

University of New Mexico

UNM Digital Repository

Electrical and Computer Engineering ETDs

Engineering ETDs

Spring 4-7-2022

Intra-hour solar forecasting using cloud dynamics features extracted from ground-based infrared sky images

Guillermo Terrén-Serrano
University of New Mexico

Follow this and additional works at: https://digitalrepository.unm.edu/ece_etds



Part of the [Applied Statistics Commons](#), [Artificial Intelligence and Robotics Commons](#), [Computational Engineering Commons](#), [Data Science Commons](#), [Geometry and Topology Commons](#), [Longitudinal Data Analysis and Time Series Commons](#), [Multivariate Analysis Commons](#), [Numerical Analysis and Computation Commons](#), [Oil, Gas, and Energy Commons](#), [Power and Energy Commons](#), [Probability Commons](#), [Signal Processing Commons](#), and the [Statistical Models Commons](#)

Recommended Citation

Terrén-Serrano, Guillermo. "Intra-hour solar forecasting using cloud dynamics features extracted from ground-based infrared sky images." (2022). https://digitalrepository.unm.edu/ece_etds/531

This Dissertation is brought to you for free and open access by the Engineering ETDs at UNM Digital Repository. It has been accepted for inclusion in Electrical and Computer Engineering ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.

Guillermo Terrén-Serrano

Candidate

Electrical and Computer Engineering

Department

This dissertation is approved, and it is acceptable in quality
and form for publication:

Approved by the Dissertation Committee:

Prof. Manel Martínez-Ramón, Chair

Prof. Ramiro Jordan, Member

Prof. Trilce Estrada, Member

Prof. Ali Bidram, Member

Intra-Hour Solar Forecasting using Cloud Dynamics Features Extracted from Ground-Based Infrared Sky Images

by

Guillermo Terrén-Serrano

B.Eng., Technical Industrial Engineering, University of Zaragoza, 2012

M.Sc., Electrical Engineering, University of New Mexico, 2016

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy
Engineering

The University of New Mexico

Albuquerque, New Mexico

May, 2022

Acknowledgments

This work has been supported by National Science Foundation (NSF) Established Program to Stimulate Competitive Research (EPSCoR) grant number OIA-1757207 and King Felipe VI Endowed Chair of the University of New Mexico (UNM). I would like to thank the UNM Center for Advanced Research Computing (CARC), supported in part by NFS, for providing the high-performance computing and large-scale storage resources used in this work. I appreciate the insightful discussions and advice from Prof. Andrea Mammoli and Prof. Marios Pattichis which helped guide this research. Distinctly, I want to recognize the valuable time and advice I received from my academic advisor Prof. Manel Martínez-Ramón. I would also like to thank Marie R. Fernandez for proofreading the manuscript and Prof. Matthew Fricke in helping me to efficiently run the experiments in the Xena and Wheeler high-performance computers.

Intra-Hour Solar Forecasting using Cloud Dynamics Features Extracted from Ground-Based Infrared Sky Images

by

Guillermo Terrén-Serrano

B.Eng., Technical Industrial Engineering, University of Zaragoza, 2012

M.Sc., Electrical Engineering, University of New Mexico, 2016

Ph.D., Engineering, University of New Mexico, 2022

Abstract

Due to the increasing use of photovoltaic systems, power grids are vulnerable to the projection of shadows from moving clouds. An intra-hour solar forecast provides power grids with the capability of automatically controlling the dispatch of energy, reducing the additional cost for a guaranteed, reliable supply of energy (i.e., energy storage). This dissertation introduces a novel sky imager consisting of a long-wave radiometric infrared camera and a visible light camera with a fisheye lens. The imager is mounted on a solar tracker to maintain the Sun in the center of the images throughout the day, reducing the scattering effect produced by the Sun's direct radiation. Features of the cloud dynamics are analyzed to compute the probability of the Sun intercepting air parcels in the sky images. Probabilistic and deterministic multi-task intra-hour solar forecasting algorithms are introduced, based on kernel and deep learning methods, to increase the penetration of photovoltaic systems in power grids.

Contents

1	Introduction	1
1.1	Energy Transition and Solar Energy	1
1.2	Power Output Forecasting	3
1.3	Solar Forecasting Framework	5
1.4	Contributions	7
1.4.1	Hardware and Dataset	7
1.4.2	Image and GSI Preprocessing	8
1.4.3	Geospatial Perspective Reprojection	8
1.4.4	Cloud Segmentation	8
1.4.5	Multiple Wind Velocity Field Detection and Approximation .	9
1.4.6	Multi-Task Intra-Hour Solar Forecasting	9
1.5	Publications	9
2	Girasol, Data and Acquisition System Specifications	13
2.1	Introduction	13
2.2	Data Description	15
2.3	Experimental Design, Materials and Methods	16
2.3.1	Study Area	17
2.3.2	Data Acquisition Hardware	18
2.3.3	Data Acquisition Software	23
3	Signal and Image Processing using Atmospheric Models	32

CONTENTS

3.1	Introduction	32
3.2	Methodology	33
3.2.1	Global Solar Irradiance Measurements	35
3.2.2	Infrared Radiometric Images	41
3.3	Experiments	53
3.3.1	Global Solar Irradiance Measurements Biases Models	53
3.3.2	Background Atmospheric Irradiance Parameters Model	56
3.3.3	Atmospheric Conditions Model	57
3.4	Discussion	60
3.5	Conclusion	65
4	Comparative Analysis of Methods for Cloud Segmentation	67
4.1	Introduction	67
4.2	Feature Vectors	70
4.3	Methods	71
4.3.1	Generative Models	71
4.3.2	Discriminative Models	78
4.4	Ridge Regression	79
4.5	Primal Solution for Support Vector Machines	80
4.6	Primal Solution for Gaussian Processes	80
4.7	J-Statistic	82
4.8	Experiments	83
4.9	Discussion	89
4.10	Conclusion	96
5	Geospatial Perspective Reprojections for Sky Imaging Systems	98
5.1	Introduction	98
5.2	Rectilinear Lens	100
5.3	Flat Earth Approximation	100

CONTENTS

5.4	Great Circle Approach	103
5.4.1	Reprojection of the y-axis	103
5.4.2	Reprojection of the x-axis	106
5.5	Conclusion	114
6	Detection of Clouds in Multiple Wind Velocity Fields	116
6.1	Introduction	116
6.2	Methodology	118
6.2.1	Weighted Lucas-Kanade	118
6.2.2	Maximum a Posteriori Mixture Model	121
6.2.3	Bayesian Metrics	128
6.2.4	Hidden Markov Model	129
6.3	Experiments	131
6.3.1	Image Preprocessing	131
6.3.2	WLK Parameters Cross-Validation	132
6.3.3	Mixture Models Parameters Cross-Validation	133
6.3.4	Testing Performance	134
6.4	Discussion	137
6.5	Conclusions	143
7	Visualization of Multiple Wind Velocity Fields for Solar Forecasting	145
7.1	Introduction	145
7.2	Wind Velocity Field	147
7.2.1	Motion Vectors	147
7.2.2	Velocity Vectors Selection	148
7.3	Flow Visualization	154
7.3.1	Wind Velocity Field Estimation	154
7.3.2	Support Vector Machine for Regression	155
7.3.3	Multi-Task Weighted Support Vector Machine	158

CONTENTS

7.3.4	Multi-Task Weighted Support Vector Machine with Flow Constraints	159
7.4	Wind Velocity Field Dynamics Estimation	161
7.5	Experiments	162
7.5.1	Training Data Construction	162
7.5.2	Velocity Vectors Calculation, Segmentation and Subsampling Parameters Validation	164
7.5.3	MT-WSVM-FC Parameters Validation	166
7.5.4	Wind Velocity Field Estimation with New Data	167
7.6	Discussion	170
7.7	Conclusions	173
8	Kernel Learning for Intra-Hour Solar Forecasting	175
8.1	Introduction	175
8.2	Kernel Methods	177
8.2.1	Dense Kernel Methods	178
8.2.2	Sparse Kernel Methods	182
8.2.3	Kernel Regression Chain	187
8.2.4	Multi-Task Kernel Simplification	188
8.3	Feature Extraction	189
8.3.1	Cloud Dynamics	189
8.3.2	Feature Selection	191
8.4	Experiments	197
8.4.1	Image Processing, Feature Extraction and Selection	197
8.4.2	Training and Testing Datasets	199
8.4.3	Data Preprocessing	199
8.4.4	Hyperparameters Cross-Validation	201
8.5	Discussion	205
8.6	Conclusion	208

CONTENTS

9	Deep Learning for Intra-Hour Solar Forecasting	211
9.1	Introduction	211
9.2	Methods	213
9.2.1	Deep Learning	214
9.3	Experiments	226
9.3.1	Image Processing, Feature Extraction and Selection of Pixels .	226
9.3.2	Training and Testing Datasets	228
9.3.3	Automatic Structural Hyperparameter Cross-Validation	230
9.3.4	Data Preprocessing	231
9.3.5	Deep Learning Architectures	232
9.4	Discussion	239
9.5	Conclusion	241
10	Conclusion	243
	Appendices	246
A	Unsupervised Parameterization of Velocity Vectors Methods	247
A.1	Methods	247
A.1.1	Velocity Vectors	248
A.1.2	Bayesian Optimization	256
A.1.3	Velocity Vectors Regularization	259
A.2	Experiments and Discussion	260
A.2.1	Parameters Cross-Validation	261
A.2.2	Note on the Implementation	263
B	Wavefunction Probabilities	265
C	Convolutional Layers	266
D	Software Availability	268

CONTENTS

Glossary	271
References	294

Chapter 1

Introduction

1.1 Energy Transition and Solar Energy

Technological innovations and climate change mitigation policies [303] are driving the ongoing transition toward energy generation systems that produce low-carbon emissions, increasing the penetration of renewable energies in the power grid [376]. A large power grid system operated using only renewable power in Europe could be theoretically possible by 2050 [370], and the projected solar energy share in the United States will increase to 47% by 2050 [85]. The only renewable energy sources that can produce enough power to fulfill the demand are geothermal, biomass, and solar [170]. In particular, solar energy has a prospective of becoming the primary source of power due to its availability and capability. Solar energy has the potential to fulfill the energy demand of the United States, with Photovoltaic (PV) systems covering less than 0.5% of the land [252], and 4 – 5.2% of the land in densely populated countries such as South Korea or Japan [337].

Investment in solar technology [107] is rising in response to the steadily increasing penetration of renewable energy in the power grid [370]. As a direct consequence

Chapter 1. Introduction

of research efforts [235], the manufacturing cost of PV systems has been reduced [368] while efficiency has increased [124] making solar energy a viable alternative to conventional thermal power plants [87] as well as the least expensive energy source in the market in some regions [32]. The main advantage of PV systems is their capacity to regulate energy dispatch when sufficient solar resources is available [56, 175]. New technological advances have resulted in 47.1% efficiency in solar cells [110]. However, this efficiency is still far from the feasible theoretical maximum of a solar cell [72]. In addition, the growth of PV solar power capacity has continued to increase in a steady exponential scale from 2000 [160].

Recent legislative initiatives incentivizing the use of solar power and other sustainable energy sources will increase the number of solar power plants connected to urban power grids worldwide [127]. California aims to have 100% of clean energy generation by 2045 [45]. Similar initiatives are occurring in Japan, South Africa, and the European Union, where local governments aim to generate 24% [104], 41% [250], and 32% [84] of their energy from renewable sources by 2030 respectively. Policies in the European Union aimed towards carbon neutrality are in place with more on the horizon [26]. In addition, the United States recently introduced climate policy packages to encourage the usage of renewable energy [309], and China announced measures to reduce carbon emissions [44]. Newly conducted surveys reveal that after the presentation of the European Green Deal for the decarbonization of the European Union by 2050 [302], most Europeans support introducing even more ambitious policies towards this objective [279]. In addition, recent studies show that the decarbonization of China's entire energy system using renewable energy resources is achievable by 2050 [197]. However, considering the grid's current state, power grid operators and policymakers would benefit from pursuing an upgrade in communication systems to incorporate smart technologies into power grids (i.e., Smart Grids (SG)) [62, 338].

The energy generated from PV systems is sensitive to the fluctuations in Global

Solar Irradiance (GSI) caused by the shadows of clouds moving in the troposphere [254, 10, 169]. The variations may cause irradiance to decrease or increase, and are a direct consequence of the scattering effect produced by clouds [22]. The decreases in irradiance are more severe when produced by cumulus clouds [335]. The increases in the irradiance that reach the surface of the Earth are produced when clouds are near the circumsolar area [220]. This is of great importance when a considerable percentage of the energy in a power grid is generated using large solar power plants [53], as moving clouds have an effect not only on the generation of energy from PV systems but also on solar thermal power plants [64, 208, 11].

As a consequence of irradiance variation caused by cloud shadows, power grids suffer mismatch losses [187], an effect of frequency and voltage misalignment between generators and load [88]. Moreover, the intermittency of solar radiation follows nonlinear global patterns that will decrease the reliability of PV systems as climate disruption intensifies [366]. These instabilities on the grid are quantified as ramp rates [189]. A proper configuration of PV arrays can reduce the impact of fluctuations, but even when the PV arrays in a power plant are arranged in a configuration capable of attenuating the effects of moving clouds, the interruptions in energy generation are out of the grid operator's admissible ramp rate [188]. As a result, grid operators are limiting the allowed ramp rates to energy providers, directly affecting renewable energy providers [194]. To increase the percentage of solar energy in the electrical power grid it is important to guarantee a reliable supply of energy [28], and to meet this end, it is necessary to equip SGs with power output forecasting algorithms [79, 99, 343, 362].

1.2 Power Output Forecasting

Accurate power output forecasting will provide grid operators with the technology necessary to control the energy dispatch in SGs with a high penetration of PV systems

[3, 218, 286, 344]. In particular, intra-hour power output forecasting (5 - 15 minutes in advance) can be used to stabilize the operations of energy providers that have large-scale solar resources (PV or concentrated solar power plants [64]) in their generation mix [355, 262], by managing the generation resources and scheduling transmission services [362]. Additionally, the increase in accuracy of intra-hour solar forecasting reduces the operational cost in SG since the dimensions of the energy storage (i.e., chemical or hydrolysis batteries, capacitor banks) are necessary to compensate for voltage fluctuation decreases [144, 246]. This technology, in turn, facilitates the development of distributed generation systems supported with solar energy [202], allowing small SG (i.e., microgrid) operators to control home appliances and other devices [56] to control the voltage fluctuations caused by moving clouds [336].

The power output of a PV system depends on multiple factors [270, 203], such as the configuration of PV arrays and the efficiency of the PV cells [265, 68]. In addition, PV cells and batteries degrade following unique patterns [102]. For these reasons, the predicted power output cannot be directly extrapolated among nearby PV systems connected to the same SG [92, 159, 348, 360]. In contrast, multiple GSI forecasts projected over a coordinate grid on the Earth's surface would provide the information necessary to regulate the dispatch of energy from different PV systems [233, 345]. The complexity of a forecasting algorithm may be reduced when cyclostationary components are removed from the analyzed time series [177, 211]. Like GSI, Clear Sky Index (CSI) forecasting is extrapolative while at the same time reducing the complexity of the algorithm [90, 304].

There is a documented relationship between ground measurements of direct normal irradiance and CSI [106]. The relationship holds in diverse climates and weather conditions [91, 263] when the CSI is calculated from visible and Infrared (IR) light sensors mounted on geostationary satellites [15, 128, 161, 287]. On-ground maps of solar irradiance can be derived from the CSI using geostationary satellite images

[168, 267]. However, solar forecasts that do not include information extracted from cloud images in the feature vectors are not effective in intra-hour forecasting [17, 272] and are only applicable when the forecasting horizon is in the range of hours or days ahead [38, 190, 230, 285, 66]. This evidence can be used to assert that the inclusion of cloudiness information from sky images into a statistical model for solar forecasting improves the overall performance of a intra-hour prediction [106]. In the application of intra-hour solar forecasting [379], ground-based methods found that the cloud cover and motion are highly correlated with the future irradiance [158, 93]. The cloud velocity vectors computed in the near circumsolar area [57, 43] are also effective to predict irradiance.

1.3 Solar Forecasting Framework

Presently, grid operators use solar forecasting algorithms for medium-term (i.e., ≤ 48 hours) energy resource planning and scheduling [149]. In this context, solar forecasting algorithms have two different ranges: intra-week and intra-day [61]. The first type of forecast provides information to the participants in the day-ahead energy market [223]. It uses Numerical Weather Prediction (NWP) models and mesoscale meteorology data [362]. These forecasting models are computationally expensive [238] for the resolution necessary in an intra-hours solar forecasting [2, 221, 222, 238, 264, 342]. In addition, solar forecasting models that include ground weather features from mesoscale meteorology have problems of collinearity [109]. The second type of forecast adjusts the energy generation to fulfill the demanded load, scheduling the charge and dispatch of energy from multiple generators [185]. For this application, it is necessary to implement forecasting models that combine sky condition information from satellite images with ground-based weather station measurements [12, 215, 307]. However, real-time applications of intra-hour solar forecasting using satellite imagery may not

be feasible due to communication delays [212]. The transmission of images from geostationary satellites may have a delay of up to an hour [164]. The purpose of intra-hour forecasting is to assist grid operators in controlling possible voltage fluctuations caused by moving clouds [336]. To perform an accurate intra-hour solar forecast, the most effective models analyze cloud dynamics information in the near-circumsolar area extracted from ground-based sky imagers [20, 46, 180, 278, 312, 260]. However, State-Of-the-Art (SOA) intra-hour forecasting applies end-to-end learning methods that do not take advantage of information fusion from multiple sensors or feature sources [100, 255, 313].

Sky images acquired with a Total Sky Imager (TSI) are commonly used in intra-hour solar forecasting algorithms to evaluate sky conditions [58]. The TSI is a ground-based sky imager which uses a concave reflective mirror to produce a sky image with a large Field Of View (FOV) [207]. The *all-sky* or *whole sky* imagers (i.e., skycams) are low-cost alternatives to TSIs [105, 51]. These sky imagers achieve a large FOV by attaching fisheye lenses to standard visible light skycams [43, 180]. The fisheye lens' distortion should be removed when applied to estimate the motion of clouds [54, 216]. The performances of a solar forecast are increased when ground-based sky imagers are installed on a solar tracker [59]. The main drawback of the TSI and skycam is the light scattering effect produced by the Sun when it appears in the images. This effect saturates the intensity of the pixels in the circumsolar area [54, 105, 299, 317], which removes information [116]. It is possible to reduce the scattering effect by installing a moving device that blocks the Sun's direct radiation [76, 78]. Unfortunately, this device still removes information in the circumsolar area necessary to increase the accuracy of an intra-hour forecasting algorithm [195, 301, 363]. Near-infrared filters attached to the visible light camera attenuate the scattering effect, but do not eliminate the problem [213]. However, far IR sky imagers considerably reduce the saturation of the circumsolar pixels [295].

IR sky imagers are advantageous in that they may be used in daytime and nighttime applications [73, 173]. For instance, in intra-day and intra-week solar forecasting, the observations of the clouds may include night hours before sunrise. A similar situation appears in intra-week forecasting [129]. In particular, ground-based radiometric long-wave IR imagers built out of uncooled microbolometers are low-cost and widely available [283]. They have been used to compute and analyze the statistics of the radiation emitted by gases and clouds in the atmosphere [296, 329] and to establish optical links for applications that involve Earth-space communication [248]. IR images allow for the derivation of physical features of the clouds such as temperature [91] and height, which are more interpretable for modeling physical processes. The measured temperature of clouds depends on the air temperature on the ground, and the calibration of cameras is important to perform accurate measurements [259]. IR sky imagers with large FOV are expensive and difficult to build [276], but low-cost alternatives that use multiple far IR cameras aim to incentivize their usage [214].

1.4 Contributions

1.4.1 Hardware and Dataset

A novel sky imager and dataset are introduced for intra-hour solar forecasting using a Data Acquisition System (DAQ) that simultaneously records sky images and GSI measurements, to extract features from clouds. The sky imaging system consists of a low-cost long-wave radiometric IR camera and a visible light camera with an attached fisheye lens. The cameras are installed inside of a weatherproof enclosure that is mounted on a solar tracker to maintain the Sun in the center of the images throughout the day, reducing the scattering effect produced by the Sun’s direct radiation (see Chapter 2).

1.4.2 Image and GSI Preprocessing

An efficient method for processing IR images and GSI measurements is proposed in this investigation, including how to remove cyclostationary biases and seasonal trends from GSI measurements, as well as processing the effects produced by the atmospheric diffuse radiation, the Sun’s direct radiation, and sediment on the germanium outdoor lens out of IR sky images. The processed IR sky images allow the derivation and extraction of physical cloud features to increase the overall performance of solar forecasting (see Chapter 3).

1.4.3 Geospatial Perspective Reprojection

The light beams received by the sky imager have an elevation angle with respect to the device’s normal. Thus, the pixels in the image contain information from different areas of the sky within the imaging system’s FOV. The FOV contained in the pixels increases as the elevation angle of the incident light beams decreases. This investigation formulates and compares two geospatial reprojections that transform the original euclidean frame of the sensor plane to the geospatial atmosphere cross-section where the sky imager FOV intersects the cloud layer (see Chapter 5).

1.4.4 Cloud Segmentation

Since real-time cloud segmentation in ground-based IR images reduces the noise in solar forecasting, a comparison of supervised and unsupervised methods [21] divided between discriminative and generative models is presented in the context of cloud segmentation. The performance of the different methods are analyzed using multiple features vectors (see Chapter 4).

1.4.5 Multiple Wind Velocity Field Detection and Approximation

Horizontal atmospheric wind shear causes wind velocity fields to have different directions and speeds. In images of clouds acquired using ground-based sky imagers, clouds may be moving in different wind layers. This dissertation introduces a method to detect (see Chapter 6) and visualize (see Chapter 7) multiple wind velocity fields in an image. When the streamlines are assumed equivalent to the pathlines in a sufficiently small air parcel, the cloud dynamics can be analyzed in sequences of images to compute the probability of the Sun intercepting air parcels in the sky images (i.e., voxels).

1.4.6 Multi-Task Intra-Hour Solar Forecasting

The methods introduced in this investigation fuse sky condition information from multiple sensors (i.e., pyranometer, sky imager, solar tracker, weather station) and feature sources. The probability of an air parcel intersecting with the Sun is calculated using a sequence of IR images. The proposed multi-task forecasting methods are based on kernel learning (see Chapter 8) and deep learning architectures using recurrent neural networks (see Chapter 9). The proposed multi-task kernel and deep learning methods for solar forecasting include both Bayesian [111] and deterministic approaches.

1.5 Publications

Most of the work included in this dissertation has been published in academic journals and conferences proceedings or is currently under review.

Preliminary Research. This work analyzed the necessary data acquisition technology and feature sources for performing an effective intra-hour solar forecasting:

- O. García-Hinde, G. Terrén-Serrano, M.A. Hombrados-Herrera, V. Gómez-Verdejo, S. Jiménez-Fernández, C. Casanova-Mateo, J. Sanz-Justo, M. Martínez-Ramón, and S. Salcedo-Sanz. Evaluation of dimensionality reduction methods applied to numerical weather models for solar radiation forecasting. *Engineering Applications of Artificial Intelligence*, 69:157 – 167, 2018.
- Andrea Mammoli, Guillermo Terrén-Serrano, Anthony Menicucci, Thomas P Caudell, and Manel Martínez-Ramón. An experimental method to merge far-field images from multiple longwave infrared sensors for short-term solar forecasting. *Solar Energy*, 187:254–260, 2019.

The work carried out in [109] in part of my master thesis [318], and the conclusions extracted are reiterated in this dissertation. The work developed in [214] is partly included in Chapter 2.

Accepted Journal Publications. These publications are part of the content in Chapter 2, 4 and 7:

- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Comparative analysis of methods for cloud segmentation in ground-based infrared images. *Renewable Energy*, 175:1025–1040, 2021.
- Guillermo Terrén-Serrano, Adnan Bashir, Trilce Estrada, and Manel Martínez-Ramón. Girasol, a sky imaging and global solar irradiance dataset. *Data in Brief*, page 106914, 2021.

Chapter 1. Introduction

- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Multi-layer wind velocity field visualization in infrared images of clouds for solar irradiance forecasting. *Applied Energy*, 288:116656, 2021.

Journal Publications Under Review These publications are part of the content in Chapter 3, 5, 6, 8 and 9:

- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Processing of global solar irradiance and ground-based infrared sky images for very short-term solar forecasting. *arXiv preprint arXiv:2101.08694*, 2021.
- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Geospatial perspective re-projections for ground-based sky imaging system. *arXiv preprint arXiv:2103.02066*, 2021.
- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Detection of clouds in multiple wind velocity fields using ground-based infrared sky images. *arXiv preprint arXiv:2105.03535*, 2021.
- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Review of kernel learning for intra-hour solar forecasting with infrared sky images and cloud dynamic feature extraction. *arXiv preprint arXiv:2110.05622*, 2021.
- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Deep learning for intra-hour solar forecasting with fusion of features extracted from infrared sky images, 2021. (Preprint available under request).

Published Conferences Proceedings. These conference publications are part of the content in Chapter 4 and 7:

Chapter 1. Introduction

- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Explicit basis function kernel methods for cloud segmentation in infrared sky images. *Energy Reports*, 7:442–450, 2021. 2021 The 4th International Conference on Electrical Engineering and Green Energy.
- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Segmentation algorithms for ground-based infrared cloud images. In *2021 IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pages 1–6, 2021.
- Guillermo Terrén-Serrano and Manel Martínez-Ramón. Wind flow estimation in thermal sky images for sun occlusion prediction. In *2021 IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pages 1–5, 2021.

Chapter 2

Girasol, Data and Acquisition System Specifications

2.1 Introduction

This chapter introduces a novel multisensor sky imager for intra-hour solar forecasting applications. The DAQ system nickname is *Girasol Machine* (Girasol means Sunflower in Spanish). The sky imager simultaneously records GSI measurements using a pyranometer, IR, and visible-light images using a low-cost long-wave radiometric IR camera and a visible-light images camera with a fisheye lens attached to it. IR cameras have been previously used in sky imaging but not mounted on a solar tracker [79]. A dataset was generated using this sky for solar forecasting, the specifications are shown in Table 2.1.

The dataset was originally conceived to address the problems related to intra-hour solar forecasting [55]. In intra-hour solar forecasting the cloud information in the images is useful to accurately predict when the Sun may be occluded by a cloud. In these events, a shadow is projected over a PV system producing a drop in the

Chapter 2. Girasol, Data and Acquisition System Specifications

energy supply [106, 180]. A GSI forecasting algorithm will provide the grid with the capability of managing the energy resources [56].

Table 2.1: This table shows the description of the dataset acquired using the *Girasol Machine* and the a summary of the hardware. This table also details the instruction to access the public dataset available in a Dryad repository.

Subject	Energy Engineering and Power
Specific subject area	Artificial intelligence system to forecast solar energy in Micro Grids and Smart Grids powered by photovoltaic technology [370].
Type of data	Image (.png) Table (.csv)
How data were acquired	Instruments: Infrared Camera, Visible Camera, Solar Tracker, Pyranometer sensor, Raspberry Pi, Motherboard Make and model and of the instruments used: FLIR© Lepton 2.5 infrared camera mounted on Pure Thermal 1 board manufactured by Group Gets. ELP© visible camera model 8541707515. Tracker FLIR© model PTU-E46-70. LI-COR© Pyranometer sensor model LI-200. Raspberry Pi 2B manufactured by SONY©. ASRock© model Syper Alloy J3455-ITX Quad-core.
Data format	Raw
Parameters for data collection	The sampling interval of the cameras is 15 seconds when the Sun's elevation angles is $> 15^\circ$. There are approximately 1,200 to 2,400 captures per day from each camera depending of the year day. The GSI signal has a sampling rate ranging from 4 to 6 samples per second. The Sun position files are generate out of the time recorded in the pyranometer files. The weather station data is updated every 10 minutes.
Description of data collection	The tracking system and DAQ software is currently operative. The software was programmed in Python 2.7. The system is placed on the top roof area of the Mechanical Engineering building in UNM central campus. The DAQ session can be visually monitored through a webpage. All devices are interconnected via a LAN network built by DHCP server. The weather station is located at the University of New Mexico Hospital, and both its real-time and historical data are publicly accessible ¹ .
Data source location	Institution: The University of New Mexico City/Town/Region: Albuquerque, New Mexico Country: United States of America Latitude and longitude for collected samples: 35.0821, -106.6259
Data accessibility	Repository name: Girasol, a Sky Imaging and Global Solar Irradiance Dataset Data identification number: 10.5061/dryad.zcrjdfn9m Direct URL to data and instructions for accessing these data: https://doi.org/10.5061/dryad.zcrjdfn9m
Related research article	G. Terrén-Serrano, M. Martínez-Ramón, Multi-Layer Wind Velocity Field Visualization in Infrared Images of Clouds for Solar Irradiance Forecasting, Applied Energy.

¹<https://www.wunderground.com/dashboard/pws/KNMALBUQ473>

The dataset includes images captured using two different light sensors equipped with lenses. The information provided by the sensors may be combined to increase the diversity of features extracted from the clouds [74]. Image processing is an important factor in the performance of a solar forecasting algorithm [363].

In the context of Machine Learning (ML) and image processing, the feature extraction algorithm may be adapted to different applications in solar problems [343, 109]. For instance, the solar forecasting horizon can be changed depending on the application. The dataset can be used to forecast the effect of the clouds in thermosolar energy generation systems such as concentrated solar power [64, 53].

2.2 Data Description

The repository contains recordings of the solar cycle from 242 days of 3 years. The total amount of data is 119GB. The sampling interval of the cameras is 15 seconds and the observation period is when the Sun's elevation angle is higher than 15° . There are approximately 1,200 to 2,400 captures per day from each camera depending on the day of the year.

Visible images are 16 bits with a resolution of 450×450 , intensity channel only. Approx. 240KB per frame. Between 200 MB to 400 MB per day depending on the amount of images in the directory. The images are saved in a lossless png format in the directory `*/visible`. The images are named by the UNIX time in seconds.

IR images are 16 bits with a resolution of 60×80 . Approx. 8KB per frame. 20MB per day roughly constant. Images are saved in lossless .png format in the directory `*/infrared`. The image are named by the UNIX time in seconds.

The pyranometer is sampled from 4 to 6 times per second. The measurements are saved in the directory `*/pyranometer` in .csv files named with their date

(`yyyy_mm_dd`), approx. 4,500KB to 7,500KB per file. The files contain UNIX time in the first column and GSI in W/m^2 in the second column.

The Sun position files are generated from the time recorded in the pyranometer files. The positions are saved in the directory `*/sun_position` in `.csv` files named with their date (`yyyy_mm_dd`), approx. 6,500KB to 11,500KB per file. The files contain the UNIX time in the first column, the elevation angle in the second column, and the azimuth angle in the third column, all of them computed with full precision UNIX time.

The weather station sample interval is 10 minutes. Linear interpolation was applied to match the sampling interval of the pyranometer. The weather station files are saved in the directory `*/weather_station` in `.csv` files named with their date (`yyyy_mm_dd`), approx. 14.3MB to 25.4MB per file. The files are organized by columns which contain, from left to right: UNIX time, temperature in $^{\circ}C$, dew point in $^{\circ}C$, atmospheric pressure in *mmHg*, wind direction in *radians*, wind velocity in *mile/s* and relative humidity in %.

2.3 Experimental Design, Materials and Methods

The *Girasol Machine* is composed of data acquisition hardware and software. In this section, we first summarize the specifications of the hardware. The described parts of the hardware are: the IR camera, the visible camera, the solar tracker and the pyranometer. We later describe the formulation of the algorithms in the software. These algorithms are designed to compute the Sun's position, to attenuate the noise in IR and visible images, and to fuse multiple exposure visible images together.

2.3.1 Study Area

The climate of Albuquerque, NM is arid semi-continental with little precipitation, which is more likely during the summer months. The average altitude of the city is 1,620m. Between mid May and mid June, the sky is clear or partly cloudy 80% of the time. Approximately 170 days of the year are sunny, with less than 30% cloud coverage, and 110 are partly sunny, with 40% to 80% cloud coverage. Temperatures range from a minimum of 268.71K in winter to a maximum of 306.48K in summer. Combined rainfall and snowfall are approximately 27.94 cm per year.

The weather features used in the image processing methods applied to remove cyclostationary effects from IR sky images and extract features from the clouds are P^{atm} , T^{air} , T^{dew} and relative humidity [%]. These weather features are acquired by a nearby weather station located on the roof of the University of New Mexico (UNM) Hospital (\approx 500m apart from the sky imager). The weather station records new measurements every 10 minutes, and its real-time and historical data are publicly accessible². The weather features were interpolated to match the sampling interval of the sky imager.

The sky imager acquires 10 consecutive IR images every 15 seconds, and averages them together to increase the signal-to-noise ratio. The IR camera sampling rate is 9 Hz, so the imager needs approximately 1 second to acquire 10 IR images. The sampling interval of 15 seconds is appropriate for solar nowcasting and intra-hour forecasting when it is mounted on a solar tracker. In addition, the IR spectrum is appropriate according to the feasible black body radiation of a cloud in the atmosphere (i.e. Wien's displacement law [39]). The large amount of images that need to be stored is relatively high (approx. 5 GB/year), so compression is needed for storage and public sharing purposes. The IR images are stored in `.png` with a parametrization

²<https://www.wunderground.com/dashboard/pws/KNMALBUQ473>

for lossless compression ($\times 5$ reduction achieved). The GSI pyranometer signal is processed using an analog anti-aliasing filter. The pyranometer sampling rate varies from 4 to 6 samples per second.

2.3.2 Data Acquisition Hardware

We built a system composed of visible and IR solar radiation cameras, a tracking system, and a pyranometer, to collect the data. A weatherproof enclosure contains the two USB cameras, and it is mounted on top of two servomotors. The cameras are connected to a motherboard running a solar tracking algorithm in parallel. This motherboard is placed inside of a different weatherproof enclosure together with a router, a servomotor control unit, a data storage system, and the respective power supplies. The enclosure's degree of protection is IP66. This is an international standard utilized for electronic equipment that provides protection against dust and water (see Figure 2.1-2.2).

Infrared Sensor

The IR sensor used to capture the images is a FLIR© Lepton 2.5 camera³ which is mounted on a Pure Thermal 1 board⁴ manufactured and distributed by Group Gets. The Lepton 2.5 sensor produces thermal images by measuring long-wave IR (see Figure 2.1). It captures IR radiation with a nominal response wavelength band from 8 to 14 μm .

The dimensions of a Lepton 2.5 are $8.5 \times 11.7 \times 5.6$ mm. It has 51° horizontal FOV and 63.5° diagonal FOV with type f1.1 silicon doublet lens. The resolution is 80 (horizontal) \times 60 (vertical) active pixels. The thermal sensitivity is < 50 mK. The

³<http://www.flir.com>

⁴<https://groupgets.com>

camera integrates digital thermal image processing functions that include automatic thermal environment compensation, noise filtering, non-uniformity correction and gain control. This sensor can produce or stream an image in < 0.5 seconds. It operates at 150 mW nominal power, and has a low power standby mode. The Pure Thermal 1 board adds functionalities to the IR sensor such as thermal video transmission over USB, which works with a USB Video Class (UVC) library on Windows, Linux, Mac and Android. It also includes on-board image processing without the need of an external system. The microprocessor is an STM32F411CEU6 ARM. It also includes a contactless thermopile temperature sensor to manually calibrate a Lepton module. The operational temperatures for Flat Field Correction function range from -10°C to 65°C . The captured images are digitized in Y16 format. It is a single channel format that only quantifies intensity levels.

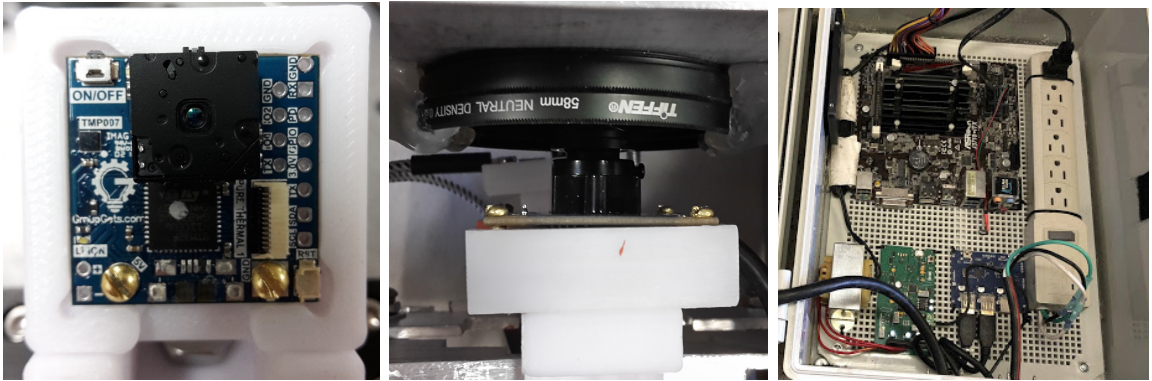


Figure 2.1: Details of IR (left) and visible (middle) cameras screwed onto their individual 3D printed support, inside the custom made enclosure, and placed in front of their respective apertures. The supports are fixed to the enclosure's structured by means of adjustable clamps. The junction box (right) contains the computer and controllers.

Visible Sensor

The visible camera is a 5 megapixel color sensor with USB 2.0 manufactured by ELP©. The sensor is an OV5640 with maximum resolution of 2592 (horizontal) \times 1944 (vertical) pixels and 170° FOV with a fisheye type lens that is adjustable within the range of 2.1 to 6 mm (see Figure 2.1). The pixel size is $1.4 \times 1.4 \mu m$, and the image area is $3673.6 \times 2738.4 \mu m$. The image stream rate is 30 frames per second at 640×480 pixel resolution. The communication protocol is UVC and its interface is a USB 2.0 high speed. The dynamic range is 68 dB, and it has a mount-in shutter that can control the frame exposure time. The camera-board has built-in functions that can be enabled for Automatic Gain Control, Automatic Exposure Control, Automatic White Balance and Automatic Black Focus. The sensor's brightness, contrast, hue, saturation, sharpness, gamma, white balance, exposure and focus are software adjustable. The power consumption in VGA resolution is 150mW. The dimensions of the camera board are 38 x 38 mm. The recommended operational temperatures for stable images range from 0°C to 60°C. The output format of the camera is YUYV. YUYV is a 3 channel format in the YUV color space, where Y represents brightness, U blue color projection and V red color projection. UVC drivers can operate in Windows, Linux, MAC and Android.

Solar Tracker

The device used for solar tracking is a commercial model manufacture by FLIR© and distributed by moviTHERM (Figure 2.2). The model is a PTU-E46-70. The tracking system is driven by two servomotors mounted to allow rotation on tilt and pan axes. The tracker is rated for a payload of 4.08 kg, (the maximum weight of the system). The rotational speed of the tracker is 60° per second, and it has a resolution of 0.003°.

The degree of freedom of the tilt servomotor allows vertical maneuvers of 78°



Figure 2.2: Germanium and neutral density lenses are attached to the IR and visible cameras, respectively, to filter the light beams and protect the cameras from weather hazards (left). The solar tracker’s servomotors are mounted on the tripod and screwed to a metallic structure which counterbalances the enclosure’s weight (middle). The sky-imaging system (right) installed on the roof of a UNM building.

with a vertical limit ranging from -47° to $+31^\circ$ with respect to the horizontal axis. The pan servomotor has more degrees of freedom. This allows the Sun to be tracked throughout the entire day. Its range is $\pm 159^\circ$. The motor speeds are adjustable up to 0.003° per second. The tracker requires an input voltage ranging from 12 to 30 VDC. The corresponding power is 13 W in maximum power mode, 6 W low-power mode and 1 W in holding power off mode. The system is controlled via Ethernet from a host computer. The control unit is connected to the servomotors via a DB-9 female connector.

Regarding the mechanical description of the tracker, the system weight is 1.36 kg, its dimensions are 7.6 cm height \times 13 cm width \times 10.17 cm depth. The control unit weighs 0.227 kg, and its dimensions are 3.18 cm height \times 8.26 cm width \times 11.43 cm depth.

Pyranometer

As part of the DAQ, a pyranometer sensor has been designed and tested to provide the device with high portability and accuracy. The circuit design for signal conditioning of the pyranometer sensor (model LI-COR LI-200) is shown in Figure 2.3. The prototype was designed to send data to a laptop, Raspberry Pi or any other computer through USB communication, to which a simple Python script needs to be added and executed. For portability purposes, the board we designed is provided with a 5 V power supply, so that a Raspberry Pi may also be connected to it.

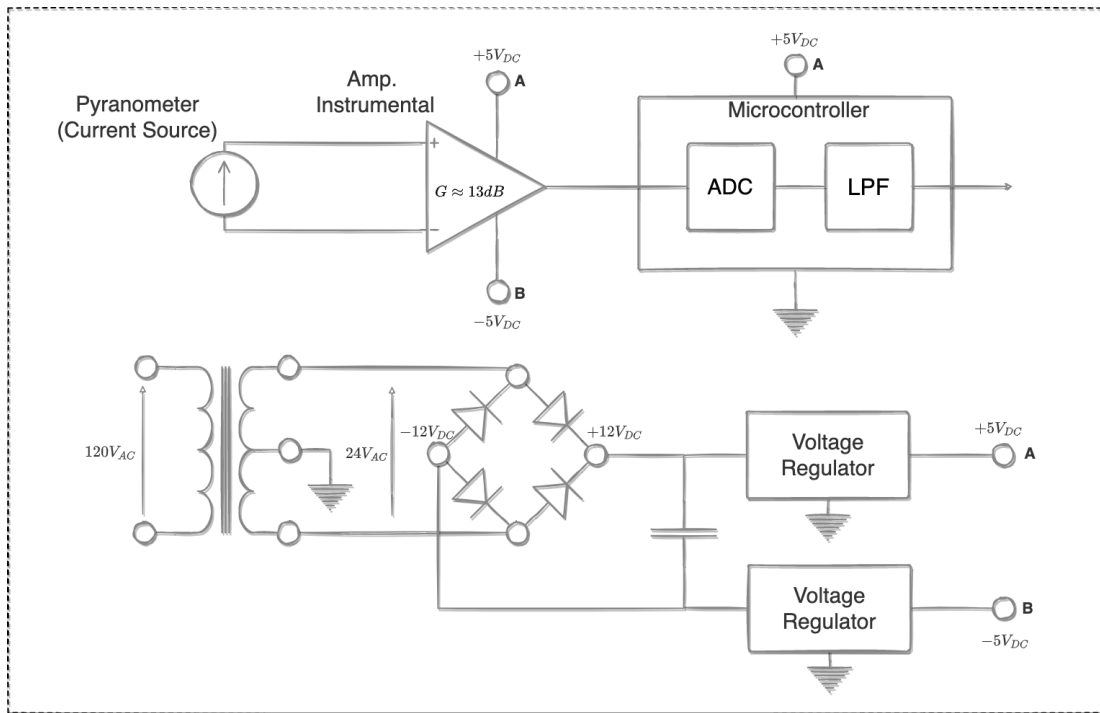


Figure 2.3: Diagram of the pyranometer circuit design. The circuit design has two parts: pyranometer signal processing and adaptation (top) and power supply (bottom). The first part has a pyranometer (current source), an instrumental amplifier, and an ADC and Low Pass Filter (LPF). The power supply (second) part has a double winding transformer, a full-bridge rectifier with a filter, and two voltage regulators that supply power at the voltage level required by the signal processing and adaptation part.

The pyranometer design objective was to condition and measure the signal of the sensor by converting it to a 12 bit digital signal and saving the data by timestamp in a `.csv` file. The system is portable, easy to setup, and reliable. The pyranometer output voltage ranges from 0 to 20 mV, where 10 mV represents a radiation power density of $1,000 \text{ W/m}^2$ (see Figure 2.5). A 120 V power outlet is used to power the device. The system is connected to the Internet and has user log-in capabilities.

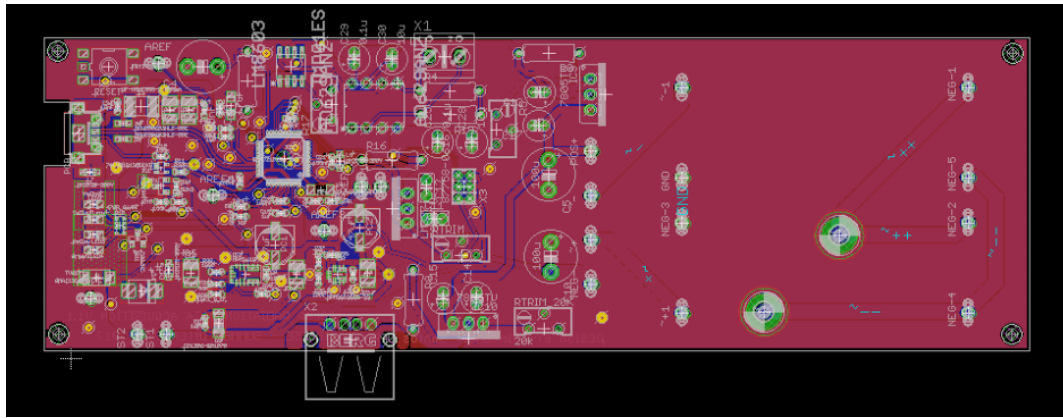


Figure 2.4: Printed circuit board layout to convert an analog signal from a pyranometer to a digital signal readable through USB. Credits to Alexander Santos-Lozano (University of Puerto Rico at Mayagüez).

Our design uses an Analog Devices AD629 instrumentation amplifier with 13 dB gain that conditions the signal to an Analog Digital Converter (ADC). An Atmel SAMD21G18 microcontroller is used to measure the signal output from the ADC and to transmit it by a USB port. The board (see Figure 2.4) is connected to a Raspberry Pi to save the signal's data in files. This Raspberry Pi is connected to the internet, so it is also able to transmit the data.

2.3.3 Data Acquisition Software

The tracking system and DAQ software is currently operative. The software was programmed in a single Python 2.7 script. The system is placed on the roof of

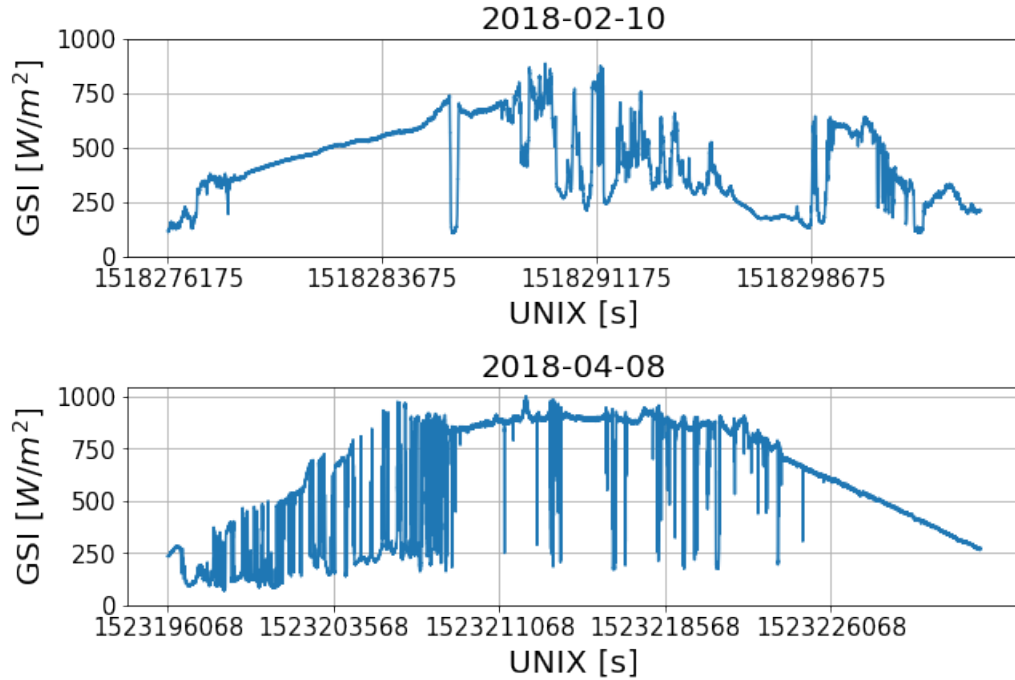


Figure 2.5: This figure shows two days of samples from the pyranometer. Day 2018-02-10 has 101510 samples and day 2018-04-08 has 131412 samples. The x axis shows the UNIX time in seconds. The y axis shows the GSI in W/m^2 .

the Electrical and Computer Engineering (ECE) building at UNM central campus (35.0821, -106.6259). The DAQ sessions can be visually monitored through a web-page. All devices were interconnected via a LAN network built by a Dynamic Host Configuration Protocol (DHCP) server.

Sun Position

The Sun position is computed to update the servomotors' position every second. The following set of equations was used to calculate azimuth and elevation angles⁵. These angles correspond to the servomotors' pan and tilt axes. The tracking system must be aligned to the true North or South of the local geographic coordinates defined

⁵<http://www.pveducation.org/>

as (λ, ϕ) , which are respectively longitude and latitude. The system must be leveled with respect to the horizontal and vertical axis. The first step is to calculate the Local Standard Time Meridian (LSTMe) is the time zone meridian with respect to the Greenwich meridian. The LSTMe, which is τ_{LSTMe} , is computed as,

$$\tau_{LSTMe} = 15^\circ \cdot \Delta\tau_{GMT}, \quad (2.1)$$

where $\Delta\tau_{GMT}$ is the difference between τ_{LT} and τ_{GMT} , which are the Local Time (LT) and Greenwich Mean Time (GMT), it is measured in hours. The Equation of Time $\mathcal{T}(\cdot)$ is an empirical equation that considers the eccentricity of earth's orbit and its axial tilt. It is expressed in minutes and is described by

$$\mathcal{T}(b) = 9.87 \cdot \sin(2b) - 7.53 \cdot \cos(b) - 1.5 \cdot \sin(b), \quad (2.2)$$

where $b = (360/365) \cdot (d - 81)$. The units of b are degrees and d corresponds to the number of days since the beginning of the year. Time Correction factor (TC), which is τ_{TC} , accounts for local effects that produce variations in the Local Solar Time (LST), defined as τ_{LST} . These effects are caused by the eccentricity of the Earth's orbit. τ_{TC} is quantified in minutes, and its equation is

$$\tau_{TC} = 4 \cdot (\lambda - \tau_{LSTMe}) + \mathcal{T}(b), \quad (2.3)$$

Constant 4 is related to the rotation of the earth, which rotates 1° every 4 minutes, and λ is the longitude. The τ_{TC} is calculated by using the previous correction and τ_{LT} . The following equation is expressed in hours

$$\tau_{LST} = \tau_{LT} + \frac{\tau_{TC}}{60}. \quad (2.4)$$

The Earth rotates 15° per hour, so each hour away from the solar noon corresponds to an angular position. The Hour Angle (HRA), defined as τ_{HRA} , converts τ_{LST} to the Sun's angle. Before noon, angles are negative and in the afternoon the angles obtained are positive. The expression for this is

$$\tau_{HRA} = 15^\circ \cdot (\tau_{LST} - 12). \quad (2.5)$$

The Sun's declination angle δ varies seasonally, and it accounts for the angular difference between the Equator and the Earth's center. δ is obtained from formula

$$\delta = 23.45^\circ \cdot \sin \left[\frac{360}{365} (d - 8) \right], \quad (2.6)$$

where d represents the day since the beginning of the year. The rotations on the tracker's pan and tilt axes are provided by the Elevation ε and Azimuth α angles. The Elevation angle quantifies the Sun's angular height, and the azimuth α is the Sun's angular position on the horizon. The angles are calculated from the following equations,

$$\begin{aligned} \varepsilon &= \sin^{-1} [\sin \delta \cdot \sin \phi + \delta \cdot \cos \phi \cdot \cos(\tau_{HRA})], \\ \alpha &= \cos^{-1} \left[\frac{\sin \delta \cdot \cos \phi - \cos \delta \cdot \sin \phi \cdot \cos(\tau_{HRA})}{\cos \xi} \right], \end{aligned} \quad (2.7)$$

where τ_{HRA} is the hour angle, ϕ is latitude, and

$$\xi = \sin^{-1} [\sin \delta \cdot \sin \phi + \delta \cdot \cos \phi \cdot \cos(\tau_{HRA})]. \quad (2.8)$$

Alternatively, ξ can be also calculated as $\xi = 90^\circ + \phi - \delta$. The zenith angle ζ is the difference between the elevation angle ε and the vertical axis, so that

$$\zeta = 90^\circ - \varepsilon. \quad (2.9)$$

The tracking system initializes one hour after the sunrise and stops one hour before sunset, so that it is inactive during the hours when the Sun's elevation is too low to generate energy by PV systems. Inverters require a minimum amount of power generation, which is reached when Sun's position is above and below these time limits. The sunrise τ_{SR} and sunset τ_{SS} are expressed in decimal hours and they are calculated in the following equations

$$\begin{aligned} \tau_{SR} &= 12 - \frac{1}{15^\circ} \cdot \cos^{-1} \left(\frac{-\sin \phi \cdot \sin \delta}{\cos \phi \cdot \cos \delta} \right) - \frac{\tau_{TC}}{60}, \\ \tau_{SS} &= 12 + \frac{1}{15^\circ} \cdot \cos^{-1} \left(\frac{-\sin \phi \cdot \sin \delta}{\cos \phi \cdot \cos \delta} \right) - \frac{\tau_{TC}}{60}. \end{aligned} \quad (2.10)$$

Noise Attenuation

The captures from the IR cameras are noisy. Therefore, we propose to attenuate the noise by averaging $N = 10$ frames taken at capture k , with a frame rate of 9 images per second. The camera frame rates are lower than the computer socket reading speed, so we need to assess whether each acquired frame read from the buffer is new or if it has been previously acquired in our set. Also, it is possible that a capture was defective. To detect these situations, we first acquire N consecutive images and we compute the sample mean image of the set $\bar{\mathbf{I}} = \frac{1}{N} \cdot \sum_{i=1}^N \mathbf{I}_i$. Then, for all images in the set, we compute the Pearson coefficient

$$\rho_{i,j} = \frac{(\mathbf{I}_i - \bar{\mathbf{I}}) \cdot (\mathbf{I}_j - \bar{\mathbf{I}})}{\sqrt{(\mathbf{I}_i - \bar{\mathbf{I}})^2} \cdot \sqrt{(\mathbf{I}_j - \bar{\mathbf{I}})^2}}, \quad \forall i, j = 1, \dots, N. \quad (2.11)$$

If a pair of frames has a coefficient equal to one, we discard one of them since the frames are the same. If an image has a coefficient less than 0.9 with respect the rest, we assume that the frame is defective, and it is also discarded. The mean is recomputed with the remaining frames and then stored (see Figure 2.6).

Exposure Switching

The scattering effect produced by solar radiation in the visible images precludes the detection of the clouds in the circumsolar region (when the exposure time is long). The light in the outer region of the image is faint when the exposure time is short. To resolve this issue, we propose to average the captured images at different exposure times. A frame is taken with four different exposure times equal to 1 ms, 4 ms, 12 ms and 28 ms. The process is repeated 10 times, and the frames with same exposure time are averaged using the procedure in Section 2.3.3, which results in 4 different frames corresponding to the four different exposures. This images are further fused using the procedure below.

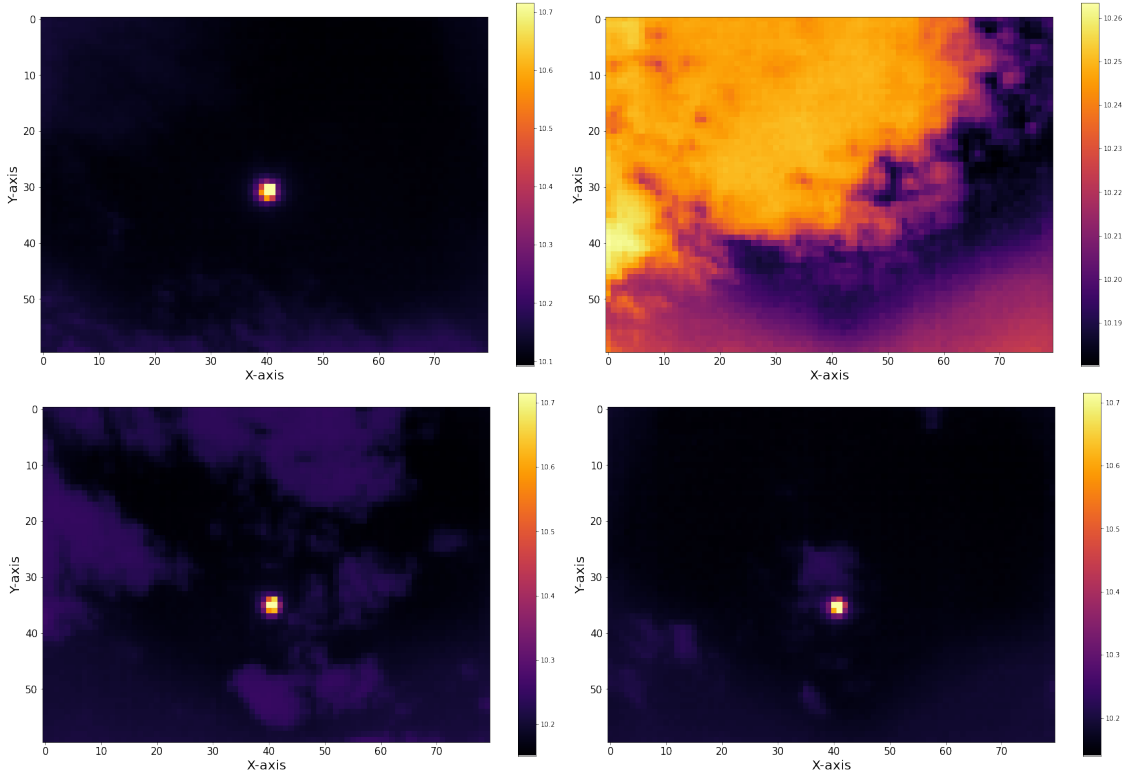


Figure 2.6: This figure shows samples (stored in the repository) of IR images that were acquired in different days. The intensity of the pixels in the images is displayed in logarithmic scale. The resolution of the images is 60×80 and their sample rate is four images per minute.

Visible Image Fusion

We introduce a method to fuse images with different exposure times, generating a high dynamic range of 16 bits, and combining cloud information from the different images. The frame is defined as $\mathbf{I}_{e,c}^k$, $1 \leq e \leq 4$ as each one of the Red, Green and Blue (RGB) components $1 \leq c \leq 3$ of image e at instant k and exposure $T_e \in \{1, 4, 12, 28\}$ ms. To merge the images we first regularized the RGB components in each exposition to avoid division by zero when light is too dim,

$$\mathbf{I}_{e,c}^k + \lambda, \quad \mathbf{I}_{e,c}^k \in \mathbb{R}^{D \times D}. \quad (2.12)$$

We convert the images to grayscale by a weighted sum of the image RGB channels with the corresponding *Luma* coding system coefficients $\beta_c = \{0.299, 0.587, 0.114\}$ [266],

$$\mathbf{I}_e^k = \sum_{c=1}^C \mathbf{I}_{e,c}^k \cdot \beta_c. \quad (2.13)$$

The proposed method for image fusion is based on the distance from a pixel to the center of the image (where the Sun is located). We know that the Sun is always centered in the frames. Therefore, we define our fusion mask as a binary valued frame function of the radial distance of the Sun to a pixel, such as

$$\mathcal{M}(r) \triangleq \{\mathbf{M}_{i,j} = \mathbb{I}((i - i_0)^2 + (j - j_0)^2 \leq r^2)\}, \quad (2.14)$$

where i and j are coordinates of a frame and $\{i_0, j_0\}$ are the coordinates of the Sun. We define a mask \mathbf{M}_e for each exposition time corresponding to a radius,

$$\mathbf{M}_e = \mathcal{M}(r_e), \quad \forall r_e = \{5, 6.25, 12.5, 25\}, \quad (2.15)$$

the set of fusion masks is then $\mathbf{M}_e = \{\mathbf{0}, \mathbf{M}_2, \dots, \mathbf{M}_E, \mathbf{1}\}$. We convolve each one of the masks with a Gaussian kernel,

$$\mathcal{N}(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{x^2 - y^2}{2\sigma^2}\right\}, \quad (2.16)$$

where x is the distance from i in the horizontal axis, y is the distance from j in the vertical axis, $\sigma = 7.5$, and has dimensions 15×15 . This operation is done to blur the mask pixels in the edges with the objective of smoothing out the transition between masked image regions in the resulting merged image. We define $\tilde{\mathbf{M}}_e$ as each one of the masks that were processed using the Gaussian filter.

The different frames are then merged with the formula

$$\mathbf{X}^k = \alpha_1^k \cdot (\tilde{\mathbf{M}}_1 \odot \mathbf{I}_1^k) + \sum_{e=2}^4 \frac{\alpha_e^k}{e} \cdot [\tilde{\mathbf{M}}_{e-1} \odot (\mathbf{1} - \tilde{\mathbf{M}}_e) \odot \mathbf{I}_e^k], \quad \mathbf{X}^k \in \mathbb{R}^{D \times D} \quad (2.17)$$

where the Hadamard or element-wise product \odot is used to create the concentric rings with radii r_{e-1} and r_e . Coefficients α_e^k are used to force the intensity of pixels of the outer edge of each ring to be equal to the intensities of the inner edge of the next ring. We obtain a set of weights $\alpha_e^k = \{1, \alpha_2^k, \dots, \alpha_E^k\}$ for each frame k by averaging the pixels of each frame outer edge as

$$\alpha_{e+1}^k = \alpha_e^k \cdot \frac{\sum_i \sum_j \mathbf{I}_e^k \odot \mathbf{R}_e^2}{\sum_i \sum_j \mathbf{I}_{e+1}^k \odot \mathbf{R}_e^1}, \quad \forall e \in \{1, \dots, E-1\}, \quad \alpha_e^k \in \mathbb{R}, \quad (2.18)$$

where \mathbf{R}_e^1 and \mathbf{R}_e^2 are rings of radius $\epsilon = \sqrt{2}$ defined as

$$\begin{aligned} \mathbf{R}_e^1 &= \mathcal{M}(r_e) \oplus \mathcal{M}(r_e + \epsilon), \quad \forall r_e = \{r_1, \dots, r_E\} \\ \mathbf{R}_e^2 &= \mathcal{M}(r_e) \oplus \mathcal{M}(r_e - \epsilon), \quad \forall r_e = \{r_1, \dots, r_E\}. \end{aligned} \quad (2.19)$$

The images are then converted to 16 bit arithmetic for further processing, and stored. Since the maximum pixel amplitude after the fusion is 225, the 8 to 16 bit conversion equation is,

$$\tilde{\mathbf{I}}^k = \left(\frac{\mathbf{X}^k}{225} \right) \cdot 2^{16}, \quad \tilde{\mathbf{I}}^k \in \mathbb{R}^{D \times D}. \quad (2.20)$$

The outer circular region in the fused image is set to an intensity value of 0, as it displays artifacts produced by its own lens holder and case (see Figure 2.7). Therefore, it does not include relevant information for the prediction.

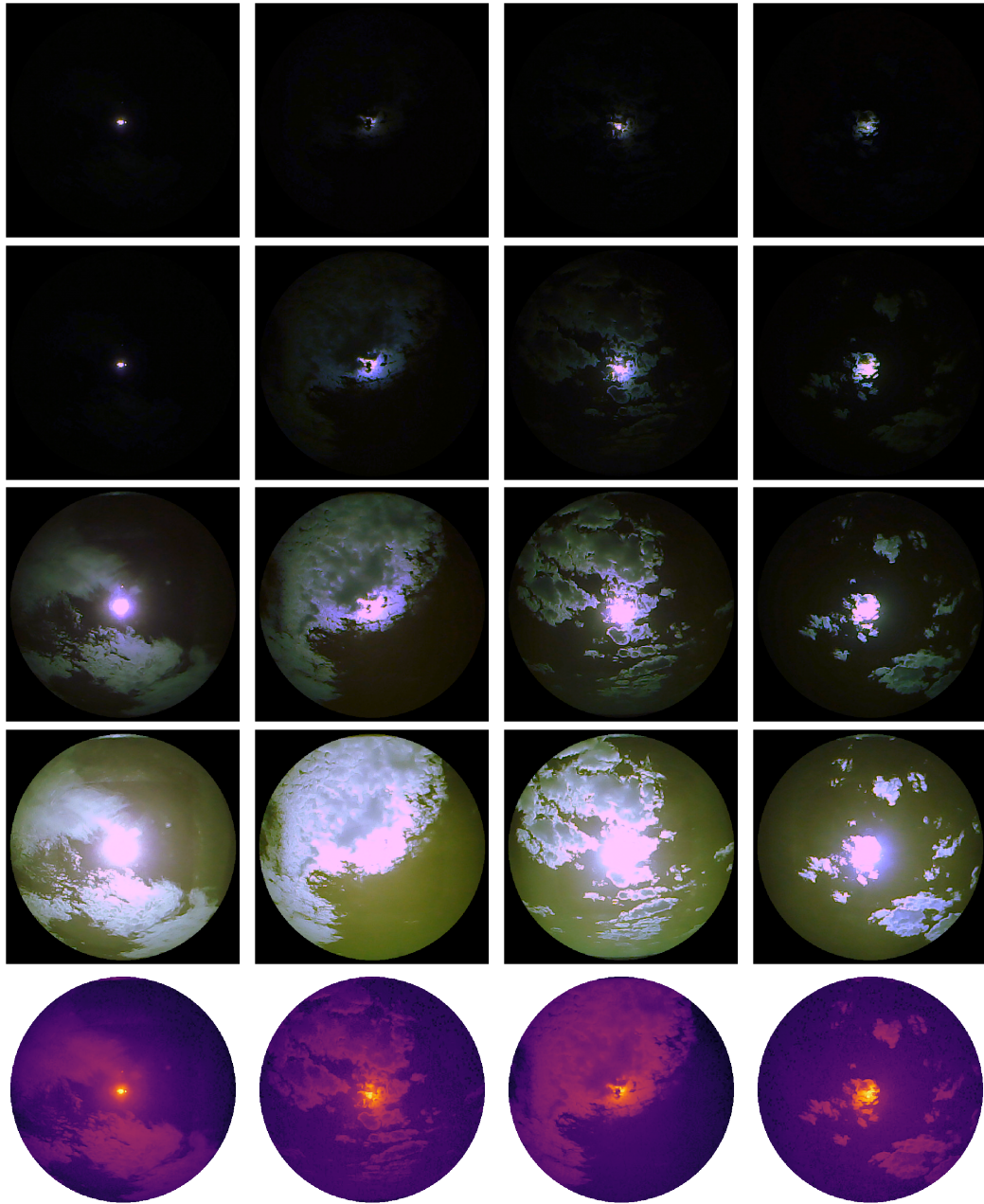


Figure 2.7: Result of the image fusion procedure. The four first rows of each column show the average of 10 sky images captured at a given exposure time (from top to bottom, 1, 4, 12 and 28 ms). The last row shows the images after completing the fusion algorithm. The resulting images are displayed in logarithmic-scale because of the high intensity of the pixels in the circumsolar region. Their size is 450×450 . Pixels without information are not displayed.

Chapter 3

Signal and Image Processing using Atmospheric Models

3.1 Introduction

An accurate segmentation of clouds enhances the performance of solar forecasting algorithms [133]. The velocity vectors may be used to know the direction and speed of clouds [91]. The velocity vectors are computed using the optical flow equation, which is based in two assumptions. The first assumption is the constant intensity of the objects in two consecutive images [96, 150, 209]. However, the intensity of clouds increases as they approach the circumsolar area in visible light imaging. The second assumption is smooth changes of intensity in the images. Sky images have an intensity gradient when the Sun is in the images. The gradient decreases as a function of the distance of a pixel to the Sun [292].

Debris such as water stains and dust particles can accumulate on sky imagers. This debris may scatter the irradiance and produce artifacts in the images that can appear similar to clouds. The camera window in the enclosure of sky imaging systems

requires routine cleaning. Sky imagers are generally installed in areas without easy access [183]. Software can reduce maintenance by removing some of these effects.

This chapter explains a method to detrend the seasonal component in GSI measurements to obtain the CSI time series [65]. This is performed optimizing the parameters of the GSI physical model to fit the GSI measurements of a pyranometer. The pyranometer is installed on a horizontal surface near to our sky imager.

Several processing methods are introduced for IR sky image application. The heights of clouds in IR sky images are approximated using the Moist Adiabatic Lapse Rate (MALR). A model that combines atmospheric irradiance scattering and the Sun's direct radiation (i.e., irradiance) in IR sky images is introduced for solar nowcasting and intra-hour forecasting applications. The parameters of the atmospheric scattering model depend on the time and the weather conditions. We propose to model the parameters of the atmospheric background irradiance model using weather features recorded from a nearby weather station.

This chapter also introduces an algorithm to model the scattering produced by debris accumulated on the outdoor window of the IR camera. This algorithm uses the last IR images that were detected as having clear sky conditions. The detection is performed with a classification model that distinguishes between four classes of sky conditions [9]. To our knowledge, no literature has been published in solar forecasting to explain how to deal with the effects of accumulated debris on a germanium outdoor lens. This problem is common within ground-based sky imaging systems.

3.2 Methodology

The methodology section is divided into the methods that are applied to the pyranometer signal (see Section 3.2.1), and the methods that are applied to the IR sky

images (see Section 3.2.2).

In Section 3.2.1, the objective is to remove cyclostationary processes from the pyranometer signal. CSI is the time series resulting after detrending a pyranometer measurement using a theoretical model. After doing this, it was found that there is an amplitude and a shifting bias between the measurements and the estimations. In this chapter, we propose to calibrate a pyranometer (via software), matching its GSI measurements with the GSI estimations from a physical model. The proposed method processes out the amplitude and shifting bias (i.e. that exists with respect to a physical model) from the pyranometer signal.

In Section 3.2.2, the objective is to process IR sky images to efficiently quantify the features from clouds. In particular, a radiometric IR camera provides thermal measurements. In the case of radiometric IR images of clouds, the temperatures can be used to estimate the height of a cloud. This is of special importance to estimate the trajectory of a cloud in a solar nowcasting or intra-hour forecasting algorithm. In Section 3.2.2, we explain how to process radiometric IR sky images to estimate the cloud height.

Another important aspect to estimate the trajectory of a cloud, is to approximate the velocity of the wind field where it is flowing. The estimation of the motion vectors in an image is sensitive to the intensity gradient (i.e. optical flow algorithms). Therefore, it is necessary to remove the gradient produced by background atmospheric irradiance. In Section 3.2.2, we explain how to process the background atmospheric irradiance from the images.

Solar nowcasting and intra-hour forecasting add the quantification of features extracted from clouds to the models. However, when IR sky images are used to extract cloud features, the radiation emitted by the outdoor camera window can produce artifacts that resemble clouds. Section 3.2.2 introduces a processing method to model

the radiation effects produced by the outdoor camera window, and explains how to normalize the resulting images to 8 bits. This is done so the 8 bit images can be used in standard optical flow algorithms for motion estimation. Finally, in Section 3.2.2, an atmospheric model is introduced with low computational cost that enables automatic learning to the algorithm.

3.2.1 Global Solar Irradiance Measurements

In this section, we explain a method to process the pyranometer GSI measurements to obtain the CSI (GSI is measured in W/m^2). First, the GSI physical model is introduced in Section 3.2.1. Second, since the theoretical estimation of GSI from the physical model does not exactly match the GSI measurements from the pyranometer, the parameters of the physical model are optimized to fit the pyranometer measurements (see Section 3.2.1). The optimization is performed with a simple minimization algorithm explained at the end of Section 3.2.1. Third, the signal from the physical model and the optimized signal are used to quantify the bias. The bias is modelled as having two different components: amplitude bias and shifting bias (see Section 3.2.1). The biases are independently modeled and finally used to correct the pyranometer measurements so they can be detrended to obtain the CSI.

Global Solar Irradiance Physical Model

The theoretical GSI \mathcal{I}_{GSI} calculated in a ground-normal horizontal surface has three components [219],

$$\mathcal{I}_{GSI} = \mathcal{I}_{Direct} + \mathcal{I}_{Diffuse} + \mathcal{I}_{Reflected} \quad (3.1)$$

The direct component of the GSI is a function of the Sun's elevation angle ε ,

$$\mathcal{I}_{Direct} = \mathcal{I}_{DN} \sin \varepsilon, \quad (3.2)$$

where the Direct Normal (DN) component is

$$\mathcal{I}_{DN} = \rho_1 \exp \left(-\frac{\rho_2}{\sin \varepsilon} \right), \quad (3.3)$$

ρ_1 is the "apparent" extraterrestrial flux and ρ_2 is the air mass coefficient.

The diffuse component of the GSI represents the solar irradiance scattering from contact with particles in the atmosphere such as the water droplets that form the clouds. The diffuse component on a ground-normal horizontal surface is

$$\mathcal{I}_{Diffuse} = \rho_3 \mathcal{I}_{DN}, \quad (3.4)$$

where the ρ_3 coefficient defines the proportion of irradiance that is scattered in an atmospheric condition.

The reflected component of GSI depends on the tilt angle of the surface. In our pyranometer, the surface is horizontal and normal to the ground, so the tilt angle is 0° . The reflected component is

$$\mathcal{I}_{Reflected} = \rho_4 \mathcal{I}_{DN} (\rho_3 + \sin \varepsilon), \quad (3.5)$$

ρ_4 is the reflective coefficient of the material where the pyranometer is placed. As the pyranometer is mounted on a black surface, the $\mathcal{I}_{Reflected}$ is assumed negligible because $\rho_4 = 0$.

When the remaining two equations are put together, \mathcal{I}_{GSI} is defined as function of the coefficients set $\boldsymbol{\rho} = \{\rho_1, \rho_2, \rho_3\} \in \mathbb{R}$ and the Sun elevation angle ε ,

$$\mathcal{I}_{GSI}(\boldsymbol{\rho}, \varepsilon) = \rho_1 \exp \left(-\frac{\rho_2}{\sin \varepsilon} \right) \left(\sin \varepsilon + \frac{\rho_3}{2} \right) \quad (3.6)$$

These three coefficients $\boldsymbol{\rho}$ can be calculated with the following set of theoretical

formulæ,

$$\begin{aligned}\rho_1 &= 1160 + 75 \sin \left(\frac{360}{N} (d - 275) \right), \\ \rho_2 &= 0.174 + 0.035 \sin \left(\frac{360}{N} (d - 100) \right), \\ \rho_3 &= 0.095 + 0.04 \sin \left(\frac{360}{N} (d - 100) \right),\end{aligned}\tag{3.7}$$

which are a function of the day of the year d and the number of days in a year N [219].

Physical Model Optimization

Assuming that D days $d = \{d \in \mathbb{N}^{[1,N]} \mid d = 1, \dots, D\}$ of clear sky GSI measurements are available, then the time series of pyranometer GSI measurements are defined as $\mathbf{y}_d = \{y_{d,k} \in \mathbb{R}^+ \mid k = 1, \dots, K_d\}$, and their corresponding elevation angles are $\boldsymbol{\varepsilon}_d = \{\varepsilon_{d,k} \in [0, \pi/2] \mid k = 1, \dots, K_d\}$, where K_d is the number of samples in a day d .

The following numerical optimization problem is solved to find the optimal set of parameters $\hat{\boldsymbol{\rho}}_d$ for each day,

$$\hat{\boldsymbol{\rho}}_d = \underset{\boldsymbol{\rho}_d}{\operatorname{argmin}} \mathcal{E}(\boldsymbol{\rho}_d), \quad \forall d = 1, \dots, D.\tag{3.8}$$

The optimal set of parameters are those that minimize the error function $\mathcal{E}(\cdot)$,

$$\mathcal{E}(\boldsymbol{\rho}_d) = \frac{1}{K_d} \sum_{k=1}^{K_d} |y_{d,k} - \mathcal{I}_{GSI}(\boldsymbol{\rho}_d, \varepsilon_{d,k})|,\tag{3.9}$$

which is the Mean Absolute Error (MAE).

The gradient of the MAE function w.r.t. each parameter $\rho_{d,i}$ in the set $\boldsymbol{\rho}$ for each day d is,

$$\frac{\partial \mathcal{E}(\boldsymbol{\rho}_d)}{\partial \rho_{d,i}} = -\frac{1}{K_d} \sum_{k=1}^{K_d} \left[\operatorname{sign}[y_{d,k} - \mathcal{I}_{GSI}(\boldsymbol{\rho}_d, \varepsilon_{d,k})] \frac{\partial \mathcal{I}_{GSI}(\boldsymbol{\rho}_d, \varepsilon_{d,k})}{\partial \rho_{d,i}} \right].\tag{3.10}$$

An optimal set of parameters $\hat{\boldsymbol{\rho}}_d$ is obtained for each day d .

Pyranometer Amplitude Bias. The amplitude of the pyranometer measurements is attenuated by a bias σ_d that is different in each day d . The amplitude bias σ_d is calculated using the theoretical GSI evaluated with the optimal coefficients $\hat{\boldsymbol{\rho}}_d$ obtained in Eq. (3.10) and the theoretical coefficient $\boldsymbol{\rho}_d$ obtained in Eq. (3.7),

$$\sigma_d = \frac{\max [\mathcal{I}_{GSI}(\hat{\boldsymbol{\rho}}_d, \boldsymbol{\varepsilon}_d)]}{\max [\mathcal{I}_{GSI}(\boldsymbol{\rho}_d, \boldsymbol{\varepsilon}_d)]}, \quad \forall d = 1, \dots, D, \quad \sigma_d \in \mathbb{R}^{[1,2]}. \quad (3.11)$$

The amplitude bias σ_d follows a periodic function with the frequency of a year. Knowing this, σ_d is modeled using a cycle-stationary periodic function,

$$\mathcal{P}e(\boldsymbol{\nu}, d) = \nu_1 \sin\left(\nu_2 + \frac{2\pi d}{N}\right) + \nu_3, \quad (3.12)$$

where N is the days in a year, d is the day of the year, and $\boldsymbol{\nu} = \{\nu_1, \nu_2, \nu_3\} \in \mathbb{R}$ is the periodic function set of parameters. The optimal set of parameters $\boldsymbol{\nu}$ are computed minimizing the error function,

$$\hat{\boldsymbol{\nu}} = \underset{\boldsymbol{\nu}}{\operatorname{argmin}} \mathcal{E}(\boldsymbol{\nu}), \quad (3.13)$$

where error of the function $\mathcal{E}(\cdot)$ is the MAE,

$$\mathcal{E}(\boldsymbol{\nu}) = \frac{1}{D} \sum_{d=1}^D |\sigma_d - \mathcal{P}e(\boldsymbol{\nu}, d)|. \quad (3.14)$$

The error function is minimized using numerical optimization. The gradient w.r.t. each parameter ν_i^2 is,

$$\frac{\partial \mathcal{E}(\boldsymbol{\nu})}{\partial \nu_i} = -\frac{1}{D} \sum_{d=1}^D \left[\operatorname{sign}[\sigma_d - \mathcal{P}e(\boldsymbol{\nu}, d)] \frac{\partial \mathcal{P}e(\boldsymbol{\nu}, d)}{\partial \nu_i} \right]. \quad (3.15)$$

The amplitude bias in a measurement is corrected as,

$$\hat{y}_{d,k} = \frac{y_{d,k}}{\sigma_d}, \quad \hat{y}_{d,k} \in \mathbb{R}^+, \quad (3.16)$$

where $\sigma_d \triangleq \mathcal{P}e(\boldsymbol{\nu}, d)$ and $y_{d,k}$ is the pyranometer measurement k of day d .

Pyranometer Shifting Bias. To correct the shifting bias in the measurement, the correlation is computed between both time series to find where the maximum is,

$$\hat{k}_d = \underset{k_d}{\operatorname{argmax}} \operatorname{corr} [\mathcal{I}_{GSI}(\boldsymbol{\rho}_d, \varepsilon_{d,k}), \hat{y}_{d,k}], \quad \forall d = 1, \dots, D, \quad (3.17)$$

where \hat{k}_d is the maximum correlation sample of day d , so $\Delta k_d = k_{d,N/2} - \hat{k}_d$ is the displacement of the time series and $\operatorname{sign}(\Delta k_d)$ is the direction of the displacement.

The theoretical GSI \mathbf{i}_d is shifted to detrend the pyranometer measurements $\hat{\mathbf{y}}_d$ using the displacement from the maximum Δk_d ,

$$i_{d,k}^{csi} = \begin{cases} \frac{\hat{y}_{d,k}}{i_{d,k-\Delta k_d}} & \operatorname{sign}(\Delta k_d) = 1 \\ \frac{\hat{y}_{d,k-\Delta k_d}}{i_{d,k}} & \operatorname{sign}(\Delta k_d) = -1 \end{cases} \quad \Delta k + 1 \leq k \leq K_d - \Delta k, \quad (3.18)$$

where \mathbf{y}_d is the CSI. When the CSI time series \mathbf{y}_d is multiplied by the theoretical GSI time series without shifting, the corrected GSI measurements $\hat{\mathbf{y}}'_d$ are obtained,

$$\hat{y}'_{d,k} = \begin{cases} y_{d,k-\Delta k_d} \cdot i_{d,k}^{csi} & \operatorname{sign}(\Delta k_d) = 1 \\ y_{d,k} \cdot i_{d,k-\Delta k_d}^{csi} & \operatorname{sign}(\Delta k_d) = -1 \end{cases} \quad \Delta k + 1 \leq k \leq K_d - \Delta k. \quad (3.19)$$

Information is lost using this correction method, since the corrected time series $\hat{\mathbf{y}}'_d$ is shorter Δk_d samples. However, this is not necessarily a problem, because we used only information from samples that have a certain elevation angle ($\varepsilon > 18^\circ$).

To approximate the shifting in the pyranometer GSI measurements, we propose a piecewise model. The set of models within the piecewise model $\mathcal{PW}(\boldsymbol{\zeta}, \boldsymbol{\delta}, d)$ are defined as horizontal lines for simplification. There are four different models. The parameters $\boldsymbol{\zeta}$ indicate a shifting depending on the day of the year of the sample. The domains of the models are defined by $\boldsymbol{\delta} = \{\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6\}$ and shiftings are

$\zeta = \{\zeta_1, \zeta_2, \zeta_3, \zeta_4, 0\}$. The proposed piecewise model is,

$$\mathcal{PW}(\zeta, \delta, d) = \begin{cases} \zeta_1 & \delta_1 \leq d < \delta_2 \\ \zeta_2 & \delta_2 \leq d < \delta_3 \\ \zeta_3 & \delta_3 \leq d < \delta_4 \\ \zeta_4 & \delta_4 \leq d < \delta_5 \\ \zeta_5 & \delta_5 \leq d \leq \delta_6 \end{cases} \quad \forall d = 1, \dots, D. \quad (3.20)$$

The optimal shifting in each domain is the sample mean,

$$\hat{\zeta}_i = \frac{1}{|\delta_i \leq d < \delta_{i+1}|} \sum_{d \in \delta_i \leq d < \delta_{i+1}} \Delta k_d, \quad \forall i = 1, \dots, 4, \quad (3.21)$$

where $|\cdot|$ denotes the cardinality of the training samples in the set.

Steepest Descent Algorithm. We propose to estimate the optimal set of parameters $\hat{\rho}$ and $\hat{\nu}$ using the steepest descent algorithm of numerical optimization [245]. The algorithm begins with the random initialization of each parameter in ρ or ν , so that $\vartheta_i^{(0)} \sim \mathcal{U}(0, K_i)$ where K_i is the maximum feasible value of a parameter ϑ_i . The set of parameters is iteratively updated using the gradient of the function,

$$\vartheta_i^{(t+1)} = \vartheta_i^{(t)} - \eta \frac{\partial \mathcal{E}(\vartheta^{(t)})}{\partial \vartheta_i^{(t)}}, \quad (3.22)$$

where $\eta \in \mathbb{R}^+$ is a very small number. Therefore, there is a series of parameter updates that converge to a function minima. The optimization algorithm stops when $\mathcal{E}(\vartheta^{(t+1)}) \geq \mathcal{E}(\vartheta^{(t)})$ and the optimal set is $\hat{\vartheta} \triangleq \vartheta^{(N)}$. The algorithm is randomly initialized N times so,

$$\hat{\vartheta} = \underset{\vartheta}{\operatorname{argmin}} \left\{ \mathcal{E}(\hat{\vartheta}^{(1)}), \dots, \mathcal{E}(\hat{\vartheta}^{(N)}) \right\}. \quad (3.23)$$

3.2.2 Infrared Radiometric Images

This section explains a method to process the IR images for solar forecasting applications. First, the method to estimate the height of a cloud is introduced in Section 3.2.2. Second, how to process the deterministic component of background atmospheric irradiance out of the IR sky images is explained (see Section 3.2.2). Third, the radiation emitted by debris on the outdoor camera window is modelled and removed from the IR images (see Section 3.2.2). Finally, an atmospheric condition model is introduced so the image processing can be implemented in an online manner (see Section 3.2.2).

Moist Adiabatic Lapse Rate

The temperature of a particle in the troposphere is a function of the height [141]. This is in contrast with the temperature in the tropopause, which is considered approximately constant [237]. The slight increase in temperature at particular heights in the stratosphere will be neglected in this chapter, since cloud phenomena are restricted to the troposphere. From the stratosphere to the exosphere the content of water vapor particles will be considered zero.

The presence of water in the troposphere implies that, in the process of convection, a rising air parcel cools down as it ascends. At a given point [240], as the temperature continues decreasing with the height, the water vapor begins to condensate, forming clouds that radiate heat [227]. This process is known as the water vapor adiabatic lapse, and while the dry adiabatic lapse rate is constant, the MALR is not [375, 148]. This fact is important because the relative humidity is nonzero in most climates. The equation for the MALR is,

$$\Gamma_{MALR} = g \frac{1 + \frac{L_r \cdot r_r}{R \cdot T^{air}}}{c_{pd} + \frac{L_p^2 r_p \epsilon}{R(T^{air})^2}} = g \frac{R_{sd}(T^{air})^2 + H_v r_v T^{air}}{c_{pd} R_{sd}(T^{air})^2 + H_v^2 r_v \epsilon} [K/m], \quad (3.24)$$

where:

- Earth's gravitational acceleration, $g = 9.8076 \text{ m/s}^2$.
- Water vaporization heat $H_v = 2501000 \text{ J/kg}$.
- Dry air specific gas constant $R_{sd} = 287 \text{ J/kgK}$.
- Water vapour specific gas constant $R_{sw} = 461.5 \text{ J/kgK}$.
- Dimensionless ratio of dry air specific gas constant to water vapor specific gas constant $\epsilon = R_{sd}/R_{sw} = 0.622$.
- Saturated air water vapor pressure, $e = \epsilon \cdot \exp([7.5T^{dew}]/[273.3 + T^{dew}]) \cdot 100$.
The original formula is in hPa but this is onen in Pa.
- Mixing ratio of water vapor mass to dry air mass $r_v = [\epsilon \cdot e]/[P^{atm} - e]$.
- Dry air specific heat at constant pressure $c_{pd} = 1003.5 \text{ J/kgK}$.

An IR image is defined as matrix \mathbf{T} with entries $T_{i,j}$, which are the IR intensity at pixel i, j in Kelvin. The ratio Γ_{MALR} is fully described knowing the air temperature T^{air} , the atmospheric pressure P^{atm} and the dew point T^{dew} . Γ_{MALR} is used to calculate the heights of the air parcels in the radiometric IR image $T_{i,j}$ pixels,

$$H_{i,j} = \frac{(T_{i,j} - T^{air})}{\Gamma_{MALR}} [m]. \quad (3.25)$$

Background Atmospheric Irradiance Model

When ground-based sky images are used for solar forecasting, analyzing the dynamics of clouds over a sequence of images is useful to anticipate when clouds will cover the Sun's direct radiation. Velocity vectors of clouds can be estimated using computer vision algorithms (i.e. dense optical flow), but these methods are sensitive to intensity

gradients in the image. To remove the gradients that may bias the motion estimation, the deterministic component (i.e. gradient produced by the Sun direct radiation and atmosphere radiation) in the IR sky images is modelled and processed out. The deterministic component is defined as the background atmospheric irradiance and it is modelled dividing it into two different components: direct and scatter irradiance. The parameters of background atmospheric irradiance are optimized using clear sky IR images. The optimal parameters of clear sky IR images are modeled to predict the optimal set of parameters in cloudy sky IR images. The predicted background atmospheric irradiance is removed from the IR images.

Solar Irradiance Models. The Sun's direct irradiance is scattered by aerosol and water particles floating in the atmosphere [186]. The scatter irradiance depends on air mass and sky conditions. The air mass varies with the altitude, because gravity attracts particles concentrating them near the surface. Sky clearness depends on the weather conditions such as wind that disperse the aerosol particles. We proposed to model the scatter and direct irradiance using parametric models to remove their effects from the IR images. The parameters of the models may be optimized to model the effects produced by scatter and direct irradiance in different atmospheric conditions. The objective is for only the radiation emitted by clouds to appear on the IR images.

The IR camera used in this chapter is installed on an azimuth-altitude mount solar tracker, which implies that the horizontal axis of the image is always parallel to the horizon. Therefore, the function proposed to model the effect of the scattered irradiance is an exponential function,

$$\mathcal{S}(j; j_0, \theta^{(1)}, \theta^{(2)}) = \theta^{(1)} \exp\left(\frac{j - j_0}{\theta^{(2)}}\right), \quad \theta^{(1)}, \theta^{(2)} \in \mathbb{R}, \quad (3.26)$$

which depends only on the vertical axis y , where θ_1 is the scale and θ_2 is the length-scale, and the function is centered in the Sun's position, with coordinates $\mathbf{x}_0 = \{i_0, j_0\}$.

The effect produced by the Sun's direct irradiance on the IR images may be

modeled by a bivariate quadratic exponential function centered in \mathbf{x}_0 . However, the Lorentzian function, which is an inverse-quadratic bivariate function, with length-scale parameter θ_4 , is more flexible in the height and tails. The Lorentzian function is

$$\mathcal{D}(i, j; \mathbf{x}_0, \theta^{(3)}, \theta^{(4)}) = \theta^{(3)} \left\{ \frac{\theta^{(4)^2}}{[(i - i_0)^2 + (j - j_0)^2 + \theta^{(4)^2}]^{\frac{3}{2}}} \right\}, \quad \theta^{(3)}, \theta^{(4)} \in \mathbb{R}. \quad (3.27)$$

Combining the function of the scatter irradiance $\mathcal{S}(\cdot)$ and the direct irradiance from the Sun $\mathcal{D}(\cdot)$, the function of background atmospheric irradiance $\mathcal{A}(\cdot)$ is obtained,

$$\mathcal{A}(i, j; \mathbf{x}_0, \boldsymbol{\theta}) = \theta^{(1)} \exp \left\{ \frac{j - j_0}{\theta^{(2)}} \right\} + \theta^{(3)} \left\{ \frac{\theta^{(4)^2}}{[(i - i_0)^2 + (j - j_0)^2 + \theta^{(4)^2}]^{\frac{3}{2}}} \right\}, \quad (3.28)$$

whose set of parameters is $\boldsymbol{\theta} = \{\theta^{(1)}, \theta^{(2)}, \theta^{(3)}, \theta^{(4)}\}$. $\mathcal{A}(i, j; \mathbf{x}_0, \boldsymbol{\theta})$ is the function used to model the deterministic component of the irradiance in the IR images.

Optimal Parameters. The parameters of the background atmospheric model must be optimized in every frame, unless a function is estimated to predict the parameters of the model, so that it is not necessary to optimize the model in each new frame. To estimate this function, the parameters of the background atmospheric model are assumed variables that depend on the weather conditions, the date and the Sun's position on the horizon. And, the parameters of the background atmospheric model are optimized in frames recorded on different days of year but which have clear sky conditions (i.e. only the deterministic component of the irradiance appears on the IR images).

The raw intensity of a pixel is defined as $\mathbf{T}^{d,k}$ in frame k of day d . The MAE function is,

$$\mathcal{E}(\boldsymbol{\theta}^{d,k}) = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N \left| T_{i,j}^{d,k} - \mathcal{A}(i, j; \mathbf{x}_0^{d,k}, \boldsymbol{\theta}^{d,k}) \right|. \quad (3.29)$$

The objective is to find the optimal set of parameters $\boldsymbol{\theta}^{d,k}$ that minimizes the loss function in frame k of day d ,

$$\hat{\boldsymbol{\theta}}^{d,k} = \underset{\boldsymbol{\theta}^{d,k}}{\operatorname{argmin}} \mathcal{E}(\boldsymbol{\theta}^{d,k}), \quad \forall k = 1, \dots, K_d, \quad \forall d = 1, \dots, D. \quad (3.30)$$

The error function gradient $\mathcal{E}(\boldsymbol{\theta}^{d,k})$ w.r.t. each parameter $\theta_l^{d,k}$ in frame k of day d is,

$$\frac{\partial \mathcal{E}(\boldsymbol{\theta}^{d,k})}{\partial \theta^{(b)d,k}} = \frac{-1}{MN} \sum_{i,j=1}^{M,N} \left[\operatorname{sign} \left[T_{i,j}^{d,k} - \mathcal{A}(i, j; \mathbf{x}_0^{d,k}, \boldsymbol{\theta}^{d,k}) \right] \frac{\partial \mathcal{A}(i, j; \mathbf{x}_0^{d,k}, \boldsymbol{\theta}^{d,k})}{\partial \theta^{(b)d,k}} \right]. \quad (3.31)$$

Model of the Parameters. From the set of parameters $\hat{\boldsymbol{\theta}}^{d,k}$ optimized for each image k in a day d , we want to construct a function $\boldsymbol{\Theta}[d, k] \in \mathbb{R}^4$ that approximates these parameters as a function the azimuth $\alpha_{d,k}$, elevation $\varepsilon_{d,k}$, dew point $T_{d,k}^{dew}$, air temperature $T_{d,k}^{air}$, day d and index k with the aim of computing the optimal values of these parameters when the image shows cloudy conditions.

This set of parameters has a physical interpretation. Parameter $\hat{\theta}_1$ is the average height of the tropopause in the IR image, and parameter $\hat{\theta}_2$ defines the cyclical pattern of the tropopause curvature cross-subsection, with daily and yearly periods. The model of these parameters is not deterministic because the sphere formed by the tropopause is not perfectly spherical and its height varies along the year depending on the latitude and global climatic patterns. The feature vectors are defined as $\mathbf{u}_{d,k}^{(1)}$ for $\theta_{d,k}^{(1)}$ and $\mathbf{u}_{d,k}^{(2)}$ for $\theta_{d,k}^{(2)}$. The features in the vectors are $\mathbf{u}_{d,k}^{(1)} = \{T_{d,k}^{air}, T_{d,k}^{dew}, \varepsilon_{d,k}, \alpha_{d,k}\}$ and $\mathbf{u}_{d,k}^{(2)} = \{d, \varepsilon_{d,k}, \alpha_{d,k}\}$.

The polynomial expansion applied to the feature vectors is defined as $\varphi : \mathcal{X} \mapsto \mathcal{P}^n$, where n is the order of the polynomial expansion. The number of terms in the polynomial expansion is $\mathcal{P}^n = [(n + (d - 1))!]/[n!(d - 1)]$. After applying the expansion to a feature vector $\mathbf{u} \mapsto \varphi(\mathbf{u})$, the feature vector is transformed into a space of \mathcal{P}^n dimensions $\mathbb{R}^{\mathcal{P}^n}$. The expression of the polynomial estimator applied to the feature

vector $\mathbf{u}_{d,k}^{(1)}$ to approximate $\theta_{d,k}^{(1)}$ is,

$$\Theta^{(1)}\left(\mathbf{u}_{d,k}^{(1)}\right)=\sum_{j,l,m,p}^n w_{j,l,m,p}^{(1)} \cdot\left(T_{d,k}^{air}\right)^j \cdot\left(T_{d,k}^{dew}\right)^l \cdot\left(\varepsilon_{d,k}\right)^m \cdot\left(\alpha_{d,k}\right)^p \quad (3.32)$$

and the polynomial estimator applied to feature vector $\mathbf{u}_{d,k}^{(2)}$ to estimate $\theta_{d,k}^{(2)}$ is,

$$\Theta^{(2)}\left(\mathbf{u}_{d,k}^{(2)}\right)=\sum_{j,l,m}^n w_{j,l,m}^{(2)} \cdot(d)^j \cdot\left(\varepsilon_{d,k}\right)^l \cdot\left(\alpha_{d,k}\right)^m \quad (3.33)$$

where $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ are vectors containing the corresponding model coefficients.

The regression problem is formulated with Tikhonov's regularization [330] applied to the model coefficients, with a regularization term of the L-2 norm simplified to $\gamma^{(b)}$. This simplification is known as Ridge Regression (RR),

$$\min _{\mathbf{w}} \sum_{d=1}^D \sum_{k=1}^{K_d}\left[\theta_{d,k}^{(b)}-\mathbf{w}^{(b) \top} \varphi\left(\mathbf{u}_{d,k}^{(b)}\right)\right]^2+\gamma^{(b)}\left\|\mathbf{w}^{(b)}\right\|_2 . \quad (3.34)$$

The parameters \mathbf{w} which minimize the quadratic error function are found analytically,

$$\begin{aligned} 0 &= \frac{\partial}{\partial \mathbf{w}} \cdot\left[\left(\boldsymbol{\theta}-\mathbf{w}^{\top} \boldsymbol{\Phi}\right)^{\top}\left(\boldsymbol{\theta}-\mathbf{w}^{\top} \boldsymbol{\Phi}\right)+\gamma \cdot \operatorname{tr}\left(\mathbf{w}^{\top} \mathbf{w}\right)\right] \\ 0 &= 2 \cdot\left[\boldsymbol{\Phi}\left(\mathbf{w}^{\top} \boldsymbol{\Phi}-\boldsymbol{\theta}\right)+\gamma \mathbf{w}\right] \\ \mathbf{w} &=\left(\boldsymbol{\Phi} \boldsymbol{\Phi}^{\top}+\gamma \mathbf{I}\right)^{-1} \boldsymbol{\Phi} \boldsymbol{\theta} . \end{aligned} \quad (3.35)$$

where the dataset definition in matrix form for model $\Theta^{(b)}\left(\mathbf{u}_{d,k}^{(b)}\right)$ containing all training samples is,

$$\boldsymbol{\theta}^{(b)}=\left[\begin{array}{c} \theta_{1,1}^{(b)} \\ \vdots \\ \theta_{D, K_D}^{(b)} \end{array}\right], \quad \boldsymbol{\Phi}^{(b)}=\left[\begin{array}{ccc} \varphi\left(\mathbf{u}_{1,1}^{(b)}\right)_1 & \cdots & \varphi\left(\mathbf{u}_{1,1}^{(b)}\right)_{\mathcal{P}^n} \\ \vdots & \ddots & \vdots \\ \varphi\left(\mathbf{u}_{D, K_D}^{(b)}\right)_1 & \cdots & \varphi\left(\mathbf{u}_{D, K_D}^{(b)}\right)_{\mathcal{P}^n} \end{array}\right], \quad \mathbf{w}^{(b)}=\left[\begin{array}{c} w_1^{(b)} \\ \vdots \\ w_{\mathcal{P}^n}^{(b)} \end{array}\right] . \quad (3.36)$$

The prediction of parameters $\theta^{(1)}$ and $\theta^{(2)}$ for a new observation $\mathbf{u}^{(b)}$ is computed using the obtained optimal parameters $\mathbf{w}^{(b)}$,

$$\Theta^{(b)}\left(\mathbf{u}^{(b)}\right)=\mathbf{w}^{(b) \top} \varphi\left(\mathbf{u}^{(b)}\right), \quad b=1,2 . \quad (3.37)$$

The parameters of the Sun's direct irradiance $\theta_{d,k}^{(3)}$ and $\theta_{d,k}^{(4)}$ are considered constant because the measured Sun's irradiance does not vary throughout the day in the IR images because the Sun's black body radiation saturates the detection capabilities of our long-wave IR camera. The optimal parameters $\hat{\theta}^{(3)}$ and $\hat{\theta}^{(4)}$ are computed using the sample mean equation,

$$\Theta^{(b)} = \frac{1}{\sum_{d=1}^D K_d} \sum_{d=1}^D \sum_{k=1}^{K_d} \theta_{d,k}^{(b)}, \quad b = 3, 4. \quad (3.38)$$

Atmospheric Model Application. If an IR image contains nimbus clouds, only the scatter irradiance model $\mathcal{S}(j; y_0^{p,q}, \Theta[p, q])$ is applied since direct irradiance does not reach the IR camera. Otherwise, the background atmospheric irradiance model $\mathcal{A}(i, j; \mathbf{x}_0^{p,q}, \Theta[p, q])$ is applied, i.e.:

$$\mathbf{T}^{p,q} = \begin{cases} \mathbf{T}^{p,q} - \mathcal{S}(j; y_0^{p,q}, \Theta[p, q]) & \text{Image contains nimbus} \\ \mathbf{T}^{p,q} - \mathcal{A}(i, j; \mathbf{x}_0^{p,q}, \Theta[p, q]) & \text{Otherwise} \end{cases} \quad \mathbf{T}^{p,q} \in \mathbb{R}^{M \times N}. \quad (3.39)$$

where $\mathbf{T}^{p,q}$ is radiometric IR image, and where $1 \leq p \leq N$, 1 is the index of any day and $1 \leq q \leq K_p$ is the image index of day p .

To determine whether an image shows nimbus clouds, an atmospheric condition classification model is introduced in Section 3.2.2.

When the model $\mathcal{A}(i, j; \mathbf{x}_0^{p,q}, \Theta[p, q])$ is subtracted, artifacts may appear in the images. The artifacts are due to the low resolution of the IR imager and systematic errors introduced by the solar tracking. To eliminate artifacts produced by errors, the intensity of the pixels that are at distance less than or equal to r_0 from the Sun position \mathbf{x}_0 are interpolated using the nearest neighbors algorithm [13]. The algorithm assigns each pixel i, j at a distance less than r_0 from the Sun the intensity of the nearest pixel whose distance is higher than r_0 . In the experiments, this distance has been fixed to $r_0 = 3$.

Outdoor Germanium Camera Window Model

The majority of IR camera windows are made out of germanium, a material which allows IR radiation transmission. After rainfall, stains produced by dried water droplets appear on the exterior of the germanium camera window. Dust particles suspended in the air also accumulate on the outdoor germanium camera window over time. We assume that the sky imager is not cleaned daily, so we propose a processing method to remove from the IR images the radiation emitted by debris on the outdoor germanium camera window. This radiation emitted is modeled using the clear sky IR images detected using an atmospheric conditions classification model.

Persistent Infrared Camera Window Model. While the tracking system rotates, the particles accumulated on the surface of the window stay constant over short periods of time. For this reason, it is possible to model them. The scatter irradiance of the outdoor germanium camera window is modeled using the set \mathcal{W} of most recent clear sky IR images. An IR image $\mathbf{T}^{p,q}$ in the set \mathcal{W} , has been classified as clear sky and the background atmospheric irradiance has been removed. Therefore, the IR images in the set \mathcal{W} only have the scatter irradiance produced by artifacts on the outdoor germanium camera window. The clear sky set \mathcal{W} contains up to S images and is defined as,

$$\mathcal{W} = \{\mathbf{T}'_1, \dots, \mathbf{T}'_S\}. \quad (3.40)$$

To determine whether an image can be included in the set \mathcal{W} , it is first classified as a clear sky image (see Section 3.2.2 below). Nevertheless, the classifier will fail to detect a clear sky image if it has a large stain, which must be detected in order to remove it. In order to overcome this situation, for every image $\mathbf{T}^{p,q}$, we compute the

average absolute difference of CSI the sequence of images from $p - i + 1$ to p as

$$r^{p,q} = \frac{1}{\varrho} \sum_{i=1}^{\varrho} |1 - i_{csi}^{p-i+1,q}|. \quad (3.41)$$

where ϱ is the number of lags in the CSI time series. A cloud close enough to the Sun to appear in the image will occlude the Sun or produce scattering. Thus, the pyranometer will register a CSI different from 1. Then, $r^{p,q} < 1$. If there is a stain in the image but no clouds, the pyranometer will show a CSI $i_{csi}^{p,q} \approx 1$. Thus, $r^{p,q} \approx 0$. This discrimination using CSI has an error probability, due to the error in the CSI estimation, occurring mostly when there are small clouds. The classifier also has error. Thus, using both decisions dramatically decreases the joint error probability.

A clear sky image $\mathbf{T}^{p,q}$ is added to the set if and only if it is classified as clear sky or $r^{p,q}$ is under a threshold τ (for robustness),

$$\mathcal{W} \triangleq \begin{cases} \mathcal{W} \cup \mathbf{T}^{p,q} & \text{Image is clear sky} \vee r^{p,q} \leq \tau \\ \mathcal{W} & \text{Otherwise,} \end{cases} \quad (3.42)$$

where $\tau = 0.05$.

The camera window model (defined as \mathbf{W}) is updated when there are M images in the clear sky set \mathcal{W} to compute the median image,

$$\mathbf{W} = \text{median}(\mathcal{W}), \quad \text{iff } |\mathcal{W}| \geq M, \quad \mathbf{W} \in \mathbb{R}^{M \times N} \quad (3.43)$$

where $|\cdot|$ is the cardinality of the set. The algorithm randomly selects M elements in the set using uniform sampling to initialize the set that will be used the next day.

Window Model Application. After the subtraction of the background atmospheric model from an IR image $\mathbf{T}^{p,q}$, the result is $\mathbf{T}^{p,q}$. Then, the window model is applied to the image,

$$\Delta \mathbf{T}^{p,q} = \mathbf{T}^{p,q} - \mathbf{W}, \quad \Delta \mathbf{T}^{p,q} \in \mathbb{R}^{M \times N}, \quad (3.44)$$

and the obtained pixel intensities are the temperature difference (with respect to the tropopause) without artifacts. In order to obtain the actual temperatures instead of the temperatures differences, after removing the window model and the atmospheric model, the amplitude of the scatter irradiance model (which is equivalent to the average temperature of the tropopause) is added to the differences,

$$\mathbf{T}''^{p,q} = \Delta\mathbf{T}^{p,q} + \theta_{p,q}^{(1)}, \quad \mathbf{T}''^{p,q} \in \mathbb{R}^{M \times N}, \quad (3.45)$$

the parameter $\theta_{p,q}^{(1)}$ is obtained from the atmospheric model parameters $\Theta_{p,q}$ in Eq. (3.39).

After removing the window model of an IR image, the temperature differences $\Delta\mathbf{T}^{p,q}$ (without artifacts) are defined with 64 bit precision. We propose normalizing the image between $\{0, 1\}$ at 64 bit precision numbers within the feasible range of cloud temperatures. The normalization allows further processing to extract information about the texture of clouds, which facilitates the cloud segmentation and labelling. The normalized image can then be transformed to 8 bits to use in standard computer vision libraries, which generally require 8 bit images (i.e. reducing bit-depth).

To normalize a 16 bit IR image to 8 bit, we assume that the average distance from the sea level to the tropopause is $\sim 12\text{km}$ at 36° latitude north [256] and that Albuquerque NM is at $1,641\text{m}$ above the sea level. If the air temperature decreases by $9.8^\circ/\text{km}$ [157], the maximum feasible intensity that a cloud can have is $9.8 \cdot (11.5 - 1.6) \cdot 100 = 9.7 \times 10^3$. The normalization of the pixels intensity in images $\Delta\mathbf{T}^{p,q}$ is,

$$\mathbf{I}^{p,q} = \frac{\Delta\mathbf{T}^{p,q} - \min(\Delta\mathbf{T}^{p,q})}{9,7 \times 10^3} \cdot 2^8, \quad \mathbf{I}^{p,q} \in \mathbb{R}^{[1,2^8]}. \quad (3.46)$$

The minimum intensity $\min(\Delta\mathbf{T}^{p,q})$ can vary between consecutive images and can have a value above zero (in cloudy condition) or below zero. The signal is offset by the noise introduced by errors in the atmospheric model parameters. The normalization is only for segmentation purposes, not for feature extraction, so this is not an issue.

Atmospheric Condition Model

The aim is to classify sky conditions in images into four categories: clear sky, cumulus, stratus or nimbus cloud. The classification determines which images are clear sky, to be used in background atmospheric model application and the camera window model. Stratus and nimbus images do not need direct sun radiation to be extracted [284]. The images have been selected manually for GSI measurements and the background atmospheric model optimization.

The classification model is also used to identify which IR images require segmentation. Cumulus clouds require segmentation, but thin stratus or thick nimbus clouds that cover the entire sky do not require segmentation.

Cloud velocity vectors were found to be an important feature to increase the accuracy in the sky conditions classification model. An arbitrary set of N pairs of consecutive images is first labelled as clear sky, cumulus, nimbus or stratus. Images are normalized so the lowest pixel is set to 0 and then the pixels are divided by the maximum feasible temperature of a cloud. The cloud velocity vectors $\hat{\mathbf{V}}$ from each pair of images were computed using two consecutive images with the Lucas-Kanade (LK) algorithm [18] (see Appendix A). The mean, variance, kurtosis and skewness of the cloud velocity vector magnitudes $\text{mag}(\hat{\mathbf{V}}_i)$ are computed. and the same statistics are also computed for the raw temperatures \mathbf{T}_i of the images. The atmospheric pressure P_i^{atm} and CSI values i_i^{csi} are included in the feature vectors \mathbf{z}_i , $1 \leq i \leq N$.

A polynomial expansion is applied to the feature vectors $\mathbf{z}_i \rightarrow \varphi(\mathbf{z}_i) \in \mathbb{R}^{\mathcal{P}^n}$, where n is the polynomial expansion order. The dataset used to train the classification model is,

$$\mathbf{\Phi} = \begin{bmatrix} \varphi(\mathbf{z}_1) & \cdots & \varphi(\mathbf{z}_N) \end{bmatrix}, \quad \mathbf{\Phi} \in \mathbb{R}^{N \times \mathcal{P}^n}, \quad \mathbf{\varsigma} = \begin{bmatrix} \varsigma_1 \\ \vdots \\ \varsigma_N \end{bmatrix}, \quad \varsigma_i \in \{1, \dots, 4\}, \quad (3.47)$$

where ς_i is the label corresponding to one of the four possible classes.

A linear Support Vector for Classification (SVC) formulated in the primal is proposed to solve the classification problem due to the large number of samples [94, 353].

The standard linear SVC has two possible categories. However, our problem requires a multi-class classification model with classes $1 \leq \ell \leq L$. The one-versus-all scheme is proposed to implement a multi-class linear SVC. The labels $\tilde{\varsigma}_{i,\ell}$ for each linear SVC are,

$$\tilde{\varsigma}_{i,\ell} = \begin{cases} +1 & \varsigma_i = \ell \\ -1 & \text{Otherwise} \end{cases} \quad \forall i = 1, \dots, N, \quad \forall \ell \in L. \quad (3.48)$$

The standard one-versus-all formulation of the linear SVC in the primal for a dataset $\mathcal{D} = \{\Phi, \tilde{\mathbf{C}}\}$ requires $L - 1$ classifiers formulated as

$$\min_{\mathbf{w}_\ell} \frac{1}{2} \|\mathbf{w}_\ell\| + \mathcal{C}_\ell \sum_{i=1}^N \xi(\mathbf{w}_\ell; \varphi(\mathbf{z}_i), \tilde{\varsigma}_{i,\ell}), \quad (3.49)$$

where \mathcal{C}_ℓ are the complexity parameters and $\xi(\mathbf{w}_\ell; \varphi(\mathbf{z}_i), \tilde{\varsigma}_{i,\ell})$ is the ε -insensitive loss function. The bias is included in the weights.

The optimal parameters \mathbf{w}_ℓ are found solving the multi-class linear SVC minimization problem formulated in the primal,

$$\min_{\mathbf{w}_\ell} \frac{1}{2} \|\mathbf{w}_\ell\|_2 + \mathcal{C}_\ell \sum_{i=1}^N \left(\max [0, 1 - \tilde{\varsigma}_{i,\ell} \mathbf{w}_\ell^\top \varphi(\mathbf{z}_i)] \right)^2, \quad (3.50)$$

where the complexity parameters $\mathcal{C}_\ell > 0$ have to be cross-validated. The predicted class for a new sample \mathbf{z}_j is,

$$\hat{\varsigma}_j = \operatorname{argmax}_{\ell \in L} \operatorname{sign} [\mathbf{w}_\ell^\top \varphi(\mathbf{z}_j)]. \quad (3.51)$$

With the assumption that the transition between sky conditions is smooth, a

persistent classification is implemented to reduce errors produced by sporadic misclassifications,

$$\hat{\mathbf{s}}_j = [\hat{\zeta}_j \cdots \hat{\zeta}_{j-\varrho+1}], \quad (3.52)$$

where ϱ is the lag in time series of consecutive classifications. The frame j is classified with the most frequent class in $\hat{\mathbf{s}}_j$,

$$\hat{\zeta}'_j = \text{mode}(\hat{\mathbf{s}}_j). \quad (3.53)$$

3.3 Experiments

3.3.1 Global Solar Irradiance Measurements Biases Models

The amplitude bias model $\mathcal{P}e(\boldsymbol{\nu}, d)$ and the shifting bias model $\mathcal{P}\mathcal{W}(\boldsymbol{\xi}, \boldsymbol{\delta}, d)$ were trained using 54 days of clear sky images. 30% of the samples were left aside for testing purposes. The models were cross-validated implementing the Leave-One-Out (LOO) method. The amplitude biases are approximated using the periodic model in Eq. (3.12). The amplitude bias samples σ_d are calculated with Eq. (3.11). The shifts are approximated using the piecewise model in Eq. (3.20). The piecewise model has 5 constant levels $\boldsymbol{\zeta}$ that are computed as the mean of the shifts at the level in Eq. (3.21). The shifting samples Δk_d are computed using Eq. (3.17). The set of domains is defined as $\boldsymbol{\delta} = \{1, 72, 220, 305, 340, 365\}$. The set of shifting levels $\boldsymbol{\zeta}$ are cross-validated.

The models were trained leaving one sample out for validation. The parameters of the models were optimized for each training subset of the LOO. The parameters were randomly initialized and optimized using the gradient descent algorithm. The optimization was repeated for 5 different initializations of the parameters. The selected set of parameters is the one that achieved smaller Root Mean Square Error

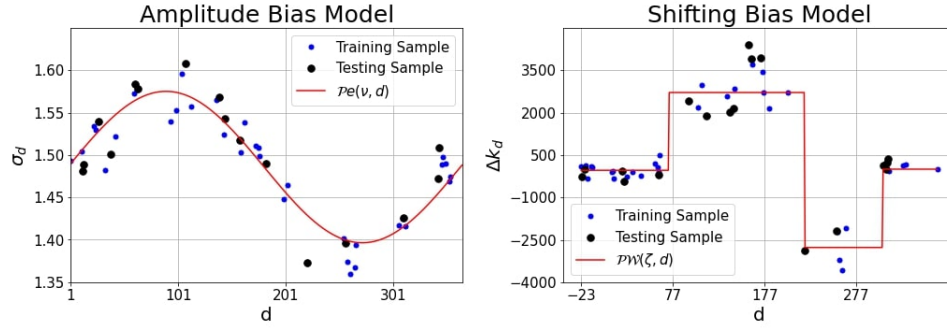


Figure 3.1: The graph on the left shows the estimation of amplitude σ_d for each day d of a year in red color using the model $\mathcal{P}e(\hat{\nu}, d)$. The graph on the right shows the shifting estimations using piecewise model $\mathcal{P}W(\hat{\zeta}, \delta, d)$ is in red. The training samples are in blue and the testing samples are in black. The amplitude bias is cyclical, but the shifting bias is not. The first day of the series is January 1st of 2018. In the shifting bias graph, the days before that year are negative, and the days after that year are greater than 365.

(RMSE). The cross-validated optimal set of parameters is the average of all the optimal parameters computed in each iteration of LOO Cross-Validation (CV) method.

The amplitude model testing error is $\text{RMSE}(\hat{\nu}) = 0.02$ and the shifting piecewise model testing error is $\text{RMSE}(\hat{\zeta}) = 440.29$. The fitting of the models is shown in Figure 3.1. The CSI obtained after detrending the pyranometer GSI measurements using the amplitude and the shifting model are shown in Figure 3.2. The error observed in Figure 3.2 after the sunrise and before the sunset is due to the scattering effects produced by small particles (pollution) and molecules (ozone) in the atmosphere. These errors are difficult to model, as they depend on the atmospheric conditions in the path travelled by light across the atmosphere.

The results show that amplitude bias varies daily reaching the maximum and the minimum in the Spring and Autumn equinox respectively (see left graph in Figure 1). The shifting bias is due to small errors in the horizontal position of the pyranometer. The methodology proposed in this chapter, corrects the pyranometer measurements avoiding to miss samples. This error is only appreciable because the resolution of

Chapter 3. Signal and Image Processing using Atmospheric Models

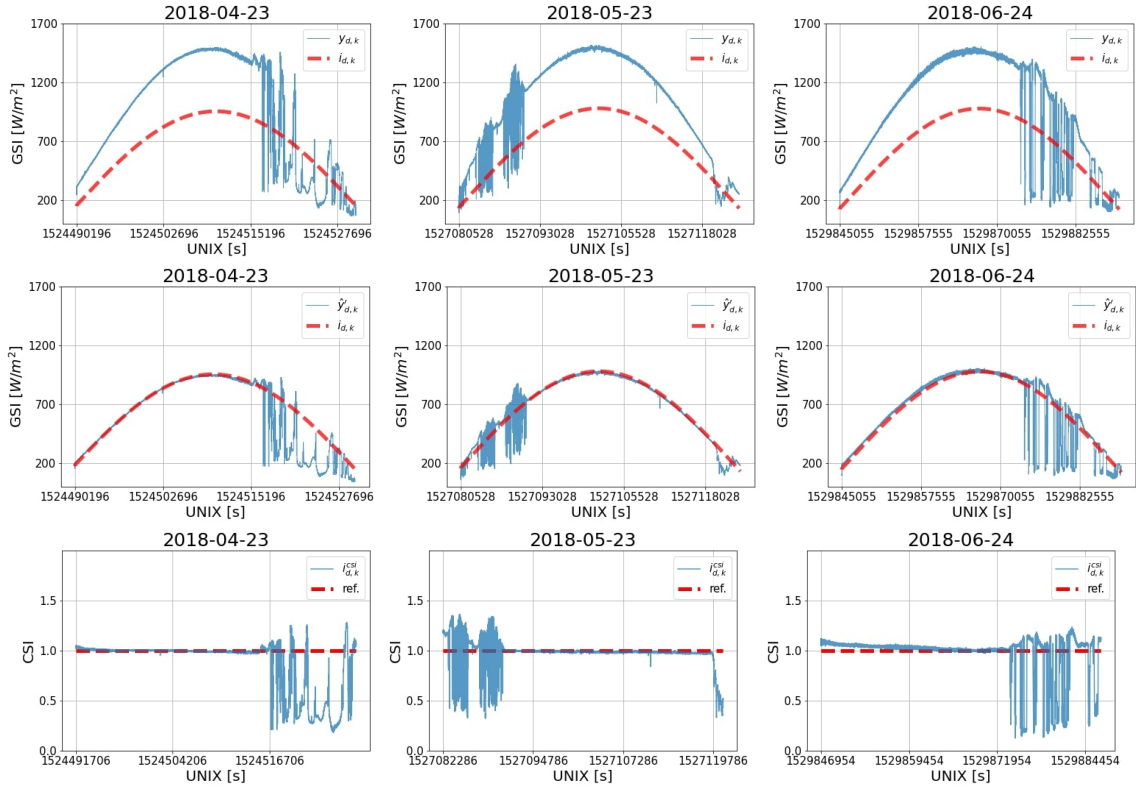


Figure 3.2: The graphs in the left column show the GSI measurements from the pyranometer (blue) and the GSI computed using the theoretical coefficients (red). The graphs in the middle column show the GSI measurements (blue) after correcting the σ_d and Δk_d biases. The graphs on the right show the CSI (blue) obtained detrending the GSI measurements using the theoretical GSI. The CSI reference for clear sky is shown in red.

the pyranometer measurements is high (6 measurements per second). The greatest shifting correction applied is approximately 5 minutes (see right graph in Figure 1). Therefore, this shifting bias will not be noticeable in time series of GSI measurements with lower resolution than 5 minutes.

3.3.2 Background Atmospheric Irradiance Parameters Model

The optimal background atmospheric irradiance parameters were acquired with a dataset of IR images and weather features from 51 days with clear sky conditions, selected out of a year's worth of samples. The number of samples available each day varies, as the number of daylight hours in each day also varies. 20% of the samples were left aside for testing purposes, and the remaining samples were used for training.

The LOO-CV method was implemented to find the optimal regularization parameter $\lambda^{(b)}$ in Eq. (3.34). The LOO-CV routine was independently implemented for the $\theta^{(1)}$ and $\theta^{(2)}$ models since the optimal feature vectors are different for each one of the parameters. In each iteration of the LOO-CV, a sample set consisting of the data for a whole day was left aside for validation purposes and the remaining were used for training the model. The best regularization for a model is the one that obtained the lowest RMSE.

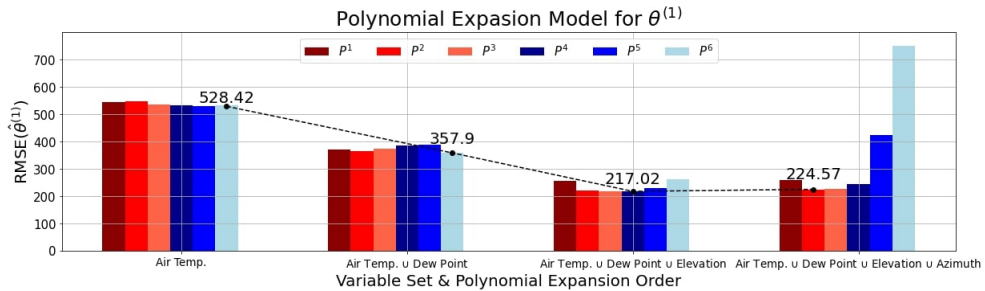


Figure 3.3: RMSE in testing achieved by the models for the optimal value of $\theta^{(1)}$. The models are grouped by the features included in the vectors. The models in a group are organized by increasing order of the polynomial expansion from left to right.

Figs. 3.3 and 3.4 show the RMSE achieved by each model experimented upon. Other variables such as relative humidity or atmospheric pressure were available, but no correlation was found with $\theta^{(1)}$ or $\theta^{(2)}$, consequently no further experimentation was carried out in this regard. Figs. 3.5 and 3.6 show the predicted $\theta^{(1)}$ and $\theta^{(2)}$ in two testing days, respectively.

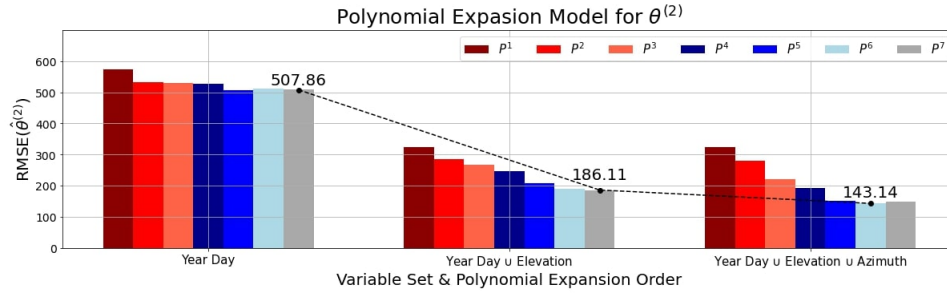


Figure 3.4: Model RMSE in testing for the optimal value of $\theta^{(1)}$. The models are grouped by the features included in the vectors. The models in a group are organized by increasing order of the polynomial expansion from left to right.

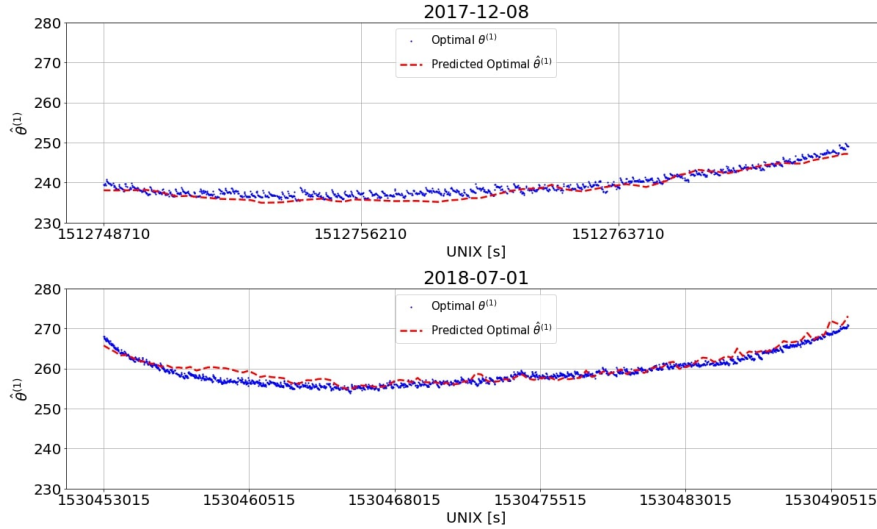


Figure 3.5: Results of the optimization of the parameter $\theta^{(1)}$ in two different days. The days are from the testing subset. The optimal parameters $\theta_{d,k}^{(1)}$ computed for each image k are in blue. The predicted optimal parameters $\hat{\theta}_{d,k}^{(1)}$ by the best $\theta^{(1)}$ model are in red.

3.3.3 Atmospheric Conditions Model

The aim of the experiments is to find the polynomial expansion order n and the combination of weather features that produces the most accurate sky condition classification. The LOO-CV method is implemented to find the optimal complexity

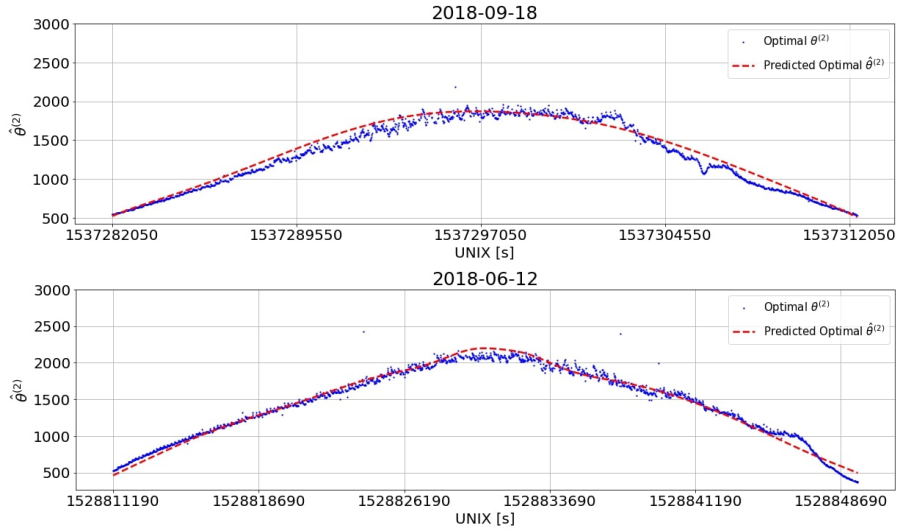


Figure 3.6: Optimal $\theta^{(2)}$ parameters in two clear sky days of the testing subset. The optimal parameters $\theta_{q,k}^{(2)}$ computed for image k are in blue. The predicted optimal parameters $\hat{\theta}_{q,k}^{(2)}$ by the best $\theta^{(2)}$ model are in red.

parameter \mathcal{C}_ℓ in Eq. (3.50). The accuracy in model validation was used to compare linear SVC models with different \mathcal{C}_ℓ .

The sequences of IR images were manually labelled. The cloud classes were determined knowing the cloud height and relative dimension with respect to the camera frame. The perspective in the IR images provides also information about the relative thickness. The sequences are from 21 nonconsecutive days and were randomly selected from the most suitable images for training, which are those that present more difficulties for a ML model (e.g. nimbus and large cumulus clouds, or effects produced by the germanium lens that provoke to confuse clear sky with a stratus or nimbus clouds). The IR images contain different sky conditions. The sequences were divided into batches with 410 samples. There are 24 batches in total, 20 (80%) were used for training and 4 (20%) were used for testing. The samples were not scrambled to preserve the time structure in the sequences. The most suitable samples were those acquired after rainfall.

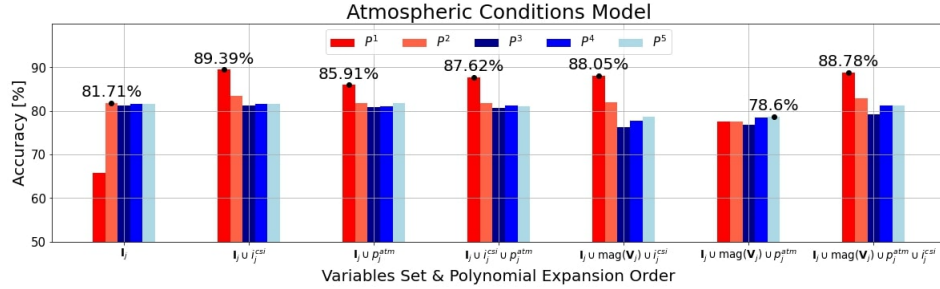


Figure 3.7: Testing classification accuracy obtained by the sky condition models. The models are grouped by the features included in the vectors. The models in a group are organized by increasing order of the polynomial expansion from left to right.

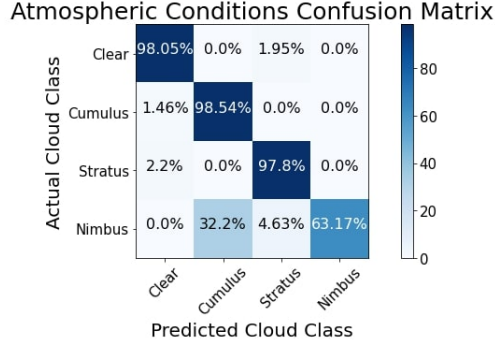


Figure 3.8: Confusion matrix achieved by the best sky condition classification model. The best model included the statistics of the temperatures and CSI $I_j \cup i_j^{csi}$ in the feature vector. No polynomial expansion was applied to the feature vector.

Figure 3.7 shows the accuracy achieved by each classification model. Figure 3.8 shows the confusion matrix of the best classification model. When pixel histogram counts are used in the feature vectors instead of pixel statistics, the models underperformed, and were consequently discarded from further experiments. The weather station features that were found to be uncorrelated with the sky condition classification are: air temperature, dew point, relative humidity, the Sun's elevation and azimuth angles.

The experiments were carried out in the Wheeler High Performance Computer (HPC) of the UNM Center for Advanced Research Computing (CARC), which uses a

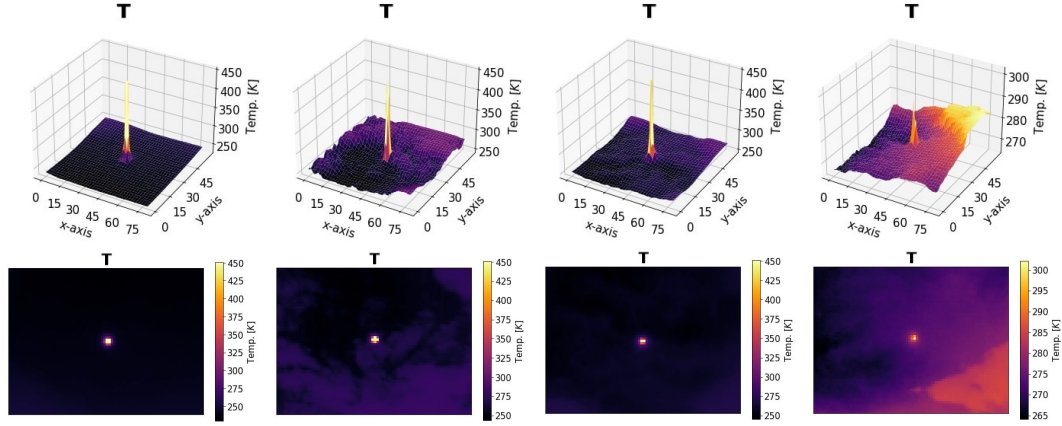


Figure 3.9: The graphs in each column correspond to an IR image from a different day of the year. The IR images were classified by the sky conditions model as: clear sky, cumulus cloud, stratus cloud and nimbus cloud. The respective radiometric temperature images are shown in 3-dimensional graphs and in color-scale images.

SGI AltixXE Xeon X5550 at 2.67GHz with 6 GB of Random Access Memory (RAM) per core, 8 cores per node, 304 nodes total, and runs at 25 theoretical peak Floating point Operations Per Second (FLOPS). Linux CentOS 7 is installed.

3.4 Discussion

The objective of detrending the GSI is to reduce the complexity of the forecasting algorithm. In this way, the forecasting model will only need to learn the stochastic component of the GSI measurements. The residual error between the theoretical GSI and actual GSI is the CSI (see Figure 3.2). The CSI quantifies the effects of clouds on GSI.

The parameters of the scatter irradiance model, $\hat{\theta}^{(1)}$ and $\hat{\theta}^{(2)}$ are variables that are modeled as functions of weather features. The model of $\hat{\theta}^{(1)}$ (which obtained the lowest error) includes the air temperature, dew point and the Sun's elevation angle (see Figure 3.3). $\hat{\theta}^{(1)}$ is the average height of the tropopause, which depends on the

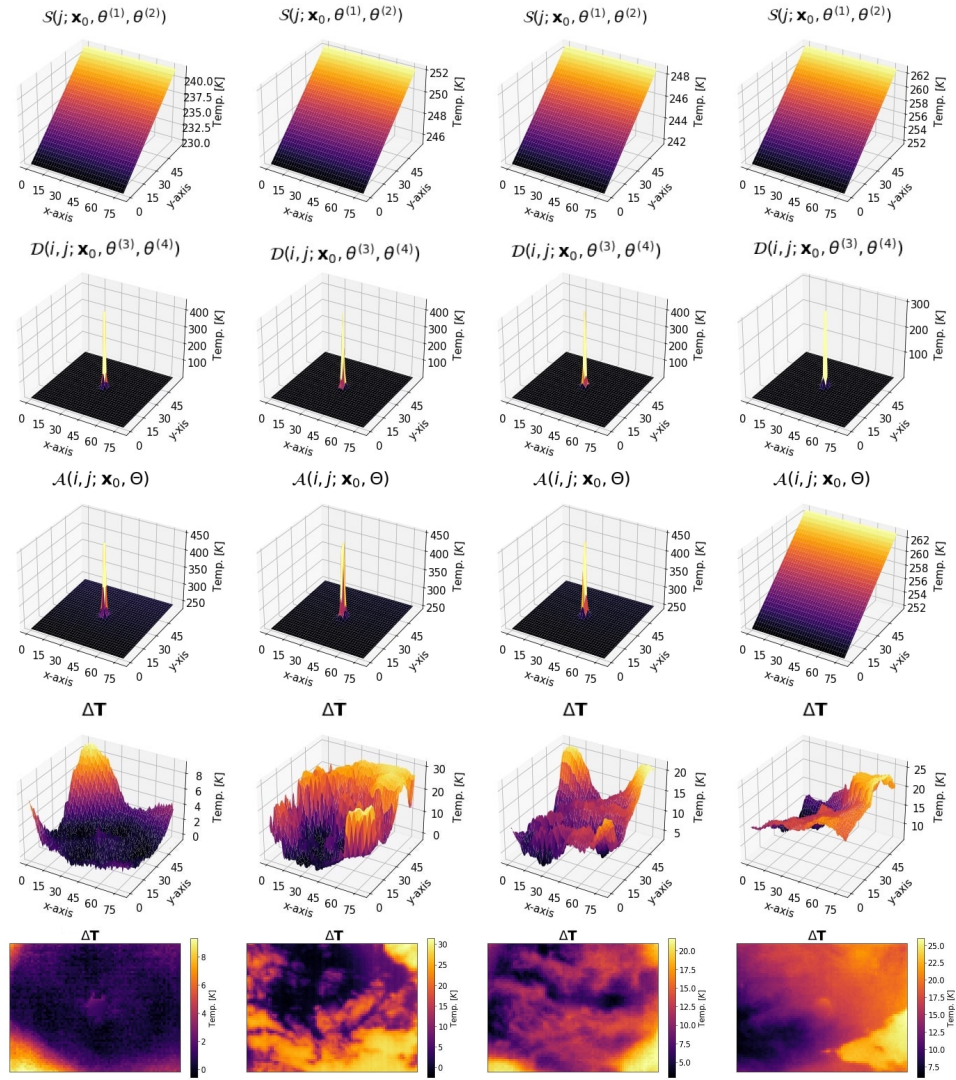


Figure 3.10: The graphs in the upper row show the irradiance scattering model $\mathcal{S}(j; j_0, \hat{\theta}^{(1)}, \hat{\theta}^{(2)})$. The graphs in the second row show the direct irradiance model $\mathcal{D}(i, j; \mathbf{x}_0, \hat{\theta}^{(3)}, \hat{\theta}^{(4)})$. The graphs in the third row show the atmospheric model $\mathcal{A}(i, j; \mathbf{x}_0, \hat{\theta}^{(3)}, \hat{\theta}^{(4)})$. Each column shows the models for a different IR image. The color-scale applied to the z-axis is different in each graph. The graphs in the last two rows show the incremental temperatures obtained after removing the atmospheric irradiance model from the radiometric temperature images in 3-dimensional graphs (fourth row) and in color-scale images (fifth row).

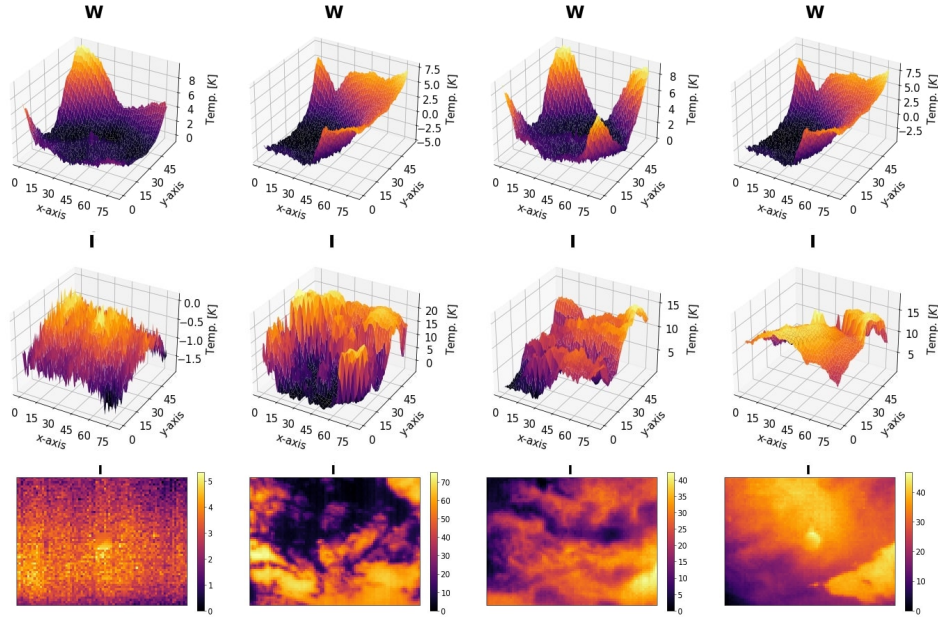


Figure 3.11: The graphs in the first row show the window effect in the radiometric measurements of temperature performed by the IR camera. The graphs in the second row show the temperature increments with respect to the tropopause temperature in the IR image (after removing the effect of the exterior camera window). The images in the third row show the normalized temperature increments.

weather conditions and the Sun's elevation in the horizon. The feature vector of the model which achieved the lowest error had a polynomial expansion of order $n = 4$. In the model of the parameter $\hat{\theta}^{(2)}$, the feature vector included the day of the year, the Sun's elevation and azimuth angles. (see in Figure 3.4). $\hat{\theta}^{(2)}$ defines the tropopause cross-subsection contained in the IR images, $\hat{\theta}^{(2)}$ depends on the time and the season. The feature vector of the model of $\hat{\theta}^{(2)}$ had a polynomial expansion of order $n = 6$. The predicted parameters $\hat{\theta}^{(1)}$ and $\hat{\theta}^{(2)}$ of the best models are in Figure 3.5 and Figure 3.6. The figures show 4 different days in the testing set.

The algorithm used in this chapter introduces a method to remove the deterministic components of solar irradiance in IR sky images. To achieve this goal, each IR image is first classified in terms of sky conditions. The raw radiometric temperatures of four

testing images are shown in Figure 3.9. These images were classified as: clear sky, cumulus cloud, stratus clouds and nimbus cloud. The parameters of the background atmospheric irradiance model were predicted to remove the atmospheric irradiance effects from the IR images. The images obtained show the temperature increments with respect to the tropopause (see results in Figure 3.10). The model of the outdoor germanium camera window was estimated using the median image of the last few IR images in the clear sky set. The effect of the germanium lens is removed and the temperature increments are normalized to the feasible range of a cloud (see Figure 3.11).

The best SVC model for the classification of all sky conditions together achieved 89.39% accuracy in testing. The optimal feature vector included the IR image's raw radiometric temperature statistics and CSI. The models that include the atmospheric pressure and the velocity vectors' magnitude statistics showed poorer performance in the testing subset (see in Figure 3.7). The feature vectors performed better without applying a polynomial expansion. The higher performing model testing confusion matrix is shown in Figure 3.8. The classification model has 98.05% accuracy in clear sky images, 98.54% in cumulus, and 97.90% in stratus cloud conditions. The model has a high rate of miss-classifications for nimbus cloud sky conditions, with 63.17% accuracy. This makes the model inefficient in determining when a cloud segmentation algorithm should be applied. However, the accuracy of detecting cloudy sky conditions is 98.78%, which is useful for updating the clear sky set of the germanium outdoor camera window model. The clear sky set of IR images in Eq. (3.40) requires $M \approx 250$ frames to learn the artifacts on the window in Eq. (3.43).

This high rate of miss-classifications is due to the cumulus and nimbus clouds forming at lower altitudes than stratus clouds, and hence having similar temperatures.

In these circumstances, the dimensions of the cloud are an important feature to differentiate cumulus from nimbus, but the sky imager only shows a portion of the

atmosphere when the Sun's elevation angle is large. In these cases, the sky portion shown in an IR camera may not have enough information to identify the size of a cloud. Therefore, the miss-classification error between nimbus and cumulus can be reduced by introducing information from a camera with a larger FOV (i.e. all-sky imager) to a classification model. In particular, nimbus clouds are only frequent during summer evenings in New Mexico, but in other locations (i.e. climate region), the frequency of nimbus clouds may increase or decrease, and vary seasonally [350, 351]. Nevertheless, the miss-classification error on Nimbus cloud does not add errors to the solar forecasting, since PV systems do not generate energy under such low irradiance circumstances.

A disadvantage of the atmospheric conditions model is that it is a supervised learning algorithm, so the method may not be extrapolable to other climate regions, and the training will require labeling a dataset from the new location. However, the feature vector includes statistical measures extracted from clouds, and if the statistical distribution of the cloud features is correlated with that in other climate regions, the model might be applicable.

The main uncertainty in the validity of the proposed models to other regions is the model of the atmospheric background model parameters. The altitude of the Tropopause varies across the latitudes, and also depends on elevation above the sea where the sky imager is localized. This is of great importance because most of the water vapor and other aerosols in the Atmosphere are within the Tropopause (i.e. Newton's gravitational law), and these are the main source of irradiance scattering.

The prediction of a parameter using the model of atmospheric background model parameters has an associated error. The error affects the resulting images after processing out the atmospheric background. Indeed, the error after processing the images might be noticeable as an offset value in the relative temperature of a cloud with respect to the Tropopause (i.e. the background temperature should be zero).

The offset is quantifiable when there are pixels in an image showing clear sky, and its quantification is of no interest when the pixels show Nimbus clouds (i.e. PV systems will not generate energy under these circumstances). The shortcoming of this approach comes to light when there are stratus clouds covering the entire image. In this case, it will not be possible to derive the water content or density of a cloud accurately.

3.5 Conclusion

This chapter introduces efficient data processing methods to remove the deterministic component of the GSI in pyranometer measurements and IR images. Adequate data processing reduces learning algorithm complexity when implemented in the application of solar forecasting. Complexity reduction increases the accuracy of the prediction and reduces the required computing time for making a prediction. This is of particular importance in real time applications such as nowcasting and intra-hour forecasting of solar energy.

Radiometric IR technology is advantageous in that it outperforms visible light sky imaging in applications with poor light conditions. In addition, debris and residue can be modeled and removed from the images. When the velocity vectors are computed using the optical flow equation, an efficient image processing algorithm is necessary. Otherwise, the computed velocity vectors will be biased by the intensity gradients of the images. In situations when the light conditions are adequate for both technologies, IR images are preferable to analyze physical processes such as clouds and solar irradiance. Thermal images facilitate the extraction of physical features which are suitable to model the underlying physical processes.

The processing of pyranometer measurements for calibration and detrending proposed in this chapter is potentially valid for irradiance measurements acquired at any geographic location. Similarly, the processing of the IR images to approximate

cloud heights using the MALR is extrapolable to other climate region, but it requires ground-based weather station measurements. The method to remove the effect produced by the germanium lens in the images is adaptable and hence potentially replicable in any IR sky imager that uses a germanium camera window in the enclosure.

This experiment has been only carried out in Albuquerque. Further evaluation of the atmospheric condition model performance in other climates is necessary to extract conclusions about the scalability of the system to other regions. The parameters of the background atmospheric irradiance model have a physical interpretation, but future research is required to develop a global model valid for any location. However, installing weather sensors in the sky imager with the same sampling interval of the pyranometer will increase the performance of the background atmospheric irradiance model. In addition to these lines of research, it will be important for the development of solar forecasting technology to determine the necessary field of view of a sky imager in relation to the forecasting horizon. Comparing the performances of an IR versus visible light sky imager will be beneficial to know the most appropriate forecasting horizons for each type of sky imager.

Chapter 4

Comparative Analysis of Methods for Cloud Segmentation

4.1 Introduction

Computer recognition of clouds is a geospatial information problem [71]. The tropopause limits the range of cloud formations, which seasonally varies across latitudes [273]. Different cloud types are expected to be found at a different range of altitudes [152]. When using features extracted from color intensity channels [198], cloud patterns inferred from data acquiesced at different latitudes may not be correlated [204]. Feature extraction methods based on Gabor filter texture analysis and statistics are more easily replicable across databases [334, 74].

Previous investigations in cloud segmentation concluded that pixel segmentation using features extracted from neighboring pixels improves performance [153, 300]. Graph models based on neighboring pixels' classification are referred to as Markov Random Field (MRF). They are a generalization of the Ising Model, first introduced in ferromagnetic problems [162], and later applied to 2-dimensional crystal lattice

problems [253]. The Iterated Conditional Modes (ICM) algorithm, developed for unsupervised training of MRFs in image processing [27], was implemented for IR satellite image cloud segmentation [258], and visible light ground-based images [199].

The superpixel approach speeds-up computing time, but produces a coarse segmentation [205]. Real-time cloud segmentation is a problem for kernel learning methods, as the Gram matrix is generally dense [317, 378]. One alternative is the use of primal formulation optimization [77]. The same problem appears with Convolutional Neural Networks (CNN) [365, 364]. The required computing time is high [74], although it is considerably reduced when using Graphical Processing Units (GPU) [83, 374]. Nevertheless, these methods require data augmentation and regularization techniques to avoid overfitting. Otherwise, the conclusions obtained are not comparable between different databases of cloud images, since the distribution of the features will vary [140]. We prove that when effective preprocessing is applied to the IR images to extract informative physical features, discriminative models are faster and have similar accuracy to generative, kernel or CNN methods.

This chapter utilizes data acquired from an innovative radiometric long-wave IR sky imager system (see Chapter 2), rather than a visible light imager system (i.e., TSI or skycams). A novelty of this work is the implementation of a preprocessing algorithm to increase the cloud segmentation performances in IR sky images. The proposed preprocessing algorithm applies two models to the IR images (see Chapter 3). The first model reproduces the scattering effect caused by debris (e.g. water stains and dust) on the outdoor germanium window of the camera. The second model reproduces the effect of direct irradiation from the Sun and scatter irradiation from the atmosphere to remove saturation in the circumsolar region, making it possible to differentiate between the Sun and clouds.

Cloud segmentation is useful to identify which pixels in an image are cloudy and which are clear-sky. This information can then be used in a solar forecasting algorithm

[30]. This chapter contributes to the field of cloud segmentation and solar forecasting through a comparative analysis of generative and discriminative models. The objective is to determine which model performs better in an IR sky-imaging system mounted on a solar tracker. The discriminative methods used in the analysis are: Ridge Regression for Classification (RRC) [297], Support Vector Classifier (SVC) [42] and Gaussian Processes for Classification (GPC) [274]. The training and testing time is drastically reduced when the RRC, SVC and GPC models are implemented in their primal formulation, because the number of dimensions obtained after mapping data to the Hilbert space is much smaller compared to the dual formulation. MRFs are part of the analyzed generative models [112]. MRF models are computationally expensive but suitable for segmentation problems [228, 201], because information from the classification of neighboring pixels is included in the prior [4]. The generative models include effective methods with low computational requirements. The training and testing computation time is improved by simplifying the covariance matrix. The Naive Bayes Classifier (NBC) and k-means clustering are simplifications of the Gaussian Discriminant Analysis (GDA) and Gaussian Mixture Model (GMM) respectively. The performances of generative models are compared between supervised (NBC, GDA and MRF) and unsupervised learning algorithms (k-means, GMM and ICM-MRF). Unsupervised learning models are less time intensive because they do not require labels to train a segmentation model, simplifying training for the user. The Simulated Annealing (SA) algorithm is implemented to perform an intelligent optimization that reduces the testing time of the MRF and ICM-MRF. A voting scheme improves the overall cloud segmentation performance of an algorithm [153]. In this chapter, the performances of the voting schemes that use all proposed methods and the optimal combination of methods are compared.

4.2 Feature Vectors

To find the optimal feature combination, we propose the validation of different physical features extracted from a pixel, and three sets of neighboring pixels, included as dependent variables in the model.

The first feature vector, $\mathbf{x}_{i,j}^1 = \{T_{i,j}, H_{i,j}\}$, contains the raw radiometric temperature of the pixels and the heights computed using the raw temperatures. The second feature vector, $\mathbf{x}_{i,j}^2 = \{T'_{i,j}, H'_{i,j}\}$, contains the temperature and height of the pixels after removing the artifacts on the IR camera's window. The third feature vector, $\mathbf{x}_{i,j}^3 = \{\Delta T_{i,j}, H''_{i,j}\}$, contains the incremental temperatures and heights after removing the Sun's direct radiation and the atmospheric scatter radiation. The fourth feature vector includes the magnitude of the cloud velocity vectors (see Appendix A), the normalized increments of temperature, and the increments of temperature; and is defined as $\mathbf{x}_{i,j}^4 = \{\text{mag}(\hat{\mathbf{v}}_{i,j}), i_{i,j}, \Delta T_{i,j}\}$.

To segment a pixel, its feature vectors and those of its neighboring pixels are introduced into the classifier. In the experiments, we define *1st order neighborhood* feature vector as the set of four pixels closest to the test pixel i, j , *2nd order neighborhood* is defined as the eight closest pixels, and term *single pixel* is used when no neighbors are included, that is:

- Single pixel: $\{\mathbf{x}_{i,j}\}, \quad \forall i, j = i_1, j_1, \dots, i_M, j_N$
- 1st order neighborhood: $\{\mathbf{x}_{i-1,j}, \mathbf{x}_{i,j-1}, \mathbf{x}_{i,j+1}, \mathbf{x}_{i+1,j}\}$.
- 2nd order neighborhood: $\{\mathbf{x}_{i-1,j}, \mathbf{x}_{i,j-1}, \mathbf{x}_{i,j+1}, \mathbf{x}_{i+1,j}, \dots$
 $\dots, \mathbf{x}_{i-1,j-1}, \mathbf{x}_{i-1,j+1}, \mathbf{x}_{i+1,j+1}, \mathbf{x}_{i+1,j-1}\}$.

4.3 Methods

The methods summarized below can be classified as generative when they have the capacity of generating new samples from a likelihood model, that is, when the model implements a density approximation of the form $p(\mathbf{x}|\mathcal{C}_k)$ where \mathcal{C}_k is the segmentation label of the pixel. Discriminative models do not have the ability to generate data since they implement a direct approximation of the posterior $p(\mathcal{C}_k|\mathbf{x})$.

4.3.1 Generative Models

Generative models are either Maximum A Posteriori (MAP) or Maximum Likelihood Estimation (MLE) methods. When generative models use an input feature structure, together with the use of an energy function for the probabilistic modeling of data (Ising model), they are generally known as MRF models. We summarize below the discriminant analysis, which applies MLE inference, GMM and k-means clustering, and supervised and unsupervised MRF methods, with MAP inference.

Discriminant Analysis

GDA and NBC are both supervised learning methods, because the training dataset input features \mathbf{x}_i are paired with a label \mathcal{C}_k . As we assume that the prior in these models is uniform, the inference applied is MLE.

Gaussian Discriminant Analysis. GDA obtains the posterior probability of $y_i = \mathcal{C}_k$ given a set of features $\mathbf{x}_i \in \mathbb{R}^D$ when applying the Bayes theorem [135], where a prior is chosen over the classes, and a Gaussian likelihood is used for the observations. The posterior of class \mathcal{C}_k , where $k \in \{1, \dots, K\}$ are possible classes, is maximized by the Bayes' rule with the expression $p(\mathbf{x}_i) \propto p(\mathcal{C}_k)p(\mathbf{x}_i | \mathcal{C}_k)$.

The corresponding means $\boldsymbol{\mu}_k \in \mathbb{R}^D$ and covariance matrices $\boldsymbol{\Sigma}_k \in \mathbb{R}^{D \times D}$ are estimated with the samples that have assigned class \mathcal{C}_k and D is the sample dimension (i.e, the number of features in vector \mathbf{x}_i).

Naive Bayes Classifier. The NBC applies the Bayes theorem, similarly to a MLE classifier, but it computes a likelihood by assuming that all features are independent. It is equivalent to a GDA where the covariance matrix of the likelihood is a diagonal matrix [239],

$$p(\mathcal{C}_k | \mathbf{x}) = \frac{p(\mathcal{C}_k) p(\mathbf{x} | \mathcal{C}_k)}{p(\mathbf{x})} \propto p(\mathcal{C}_k, x_1, \dots, x_d) \quad (4.1)$$

where x_j are the features of each sample vector \mathbf{x} .

Applying the naive conditional independent assumption between the features for simplification, is obtained that,

$$p(x_j | x_{j+1}, \dots, x_d, \mathcal{C}_k) = p(x_j | \mathcal{C}_k). \quad (4.2)$$

At this point, when the chain rule is be applied, the model can be expressed as product of factorized probabilities,

$$\begin{aligned} p(\mathcal{C}_k | x_1, \dots, x_d) &\propto p(\mathcal{C}_k, x_1, \dots, x_d) \\ &= p(\mathcal{C}_k) p(x_1 | \mathcal{C}_k) p(x_2 | \mathcal{C}_k) p(x_3 | \mathcal{C}_k) \cdots \\ &= p(\mathcal{C}_k) \prod_{j=1}^d p(x_j | \mathcal{C}_k). \end{aligned} \quad (4.3)$$

A class is assigned to a sample \mathbf{x}_i applying MLE classification criteria that is defined as,

$$\hat{y}_i = \underset{k}{\operatorname{argmax}} \prod_{j=1}^D p(x_{i,j} | \mathcal{C}_k), \quad (4.4)$$

which is equivalent to maximizing the posterior (4.3) since the prior of a class $p(\mathcal{C}_k)$ is assumed to be uniform, in the same way that GDA.

The naive classifier in our application is implemented using a normal distribution for each feature x_j in a class \mathcal{C}_k ,

$$p(x_j | \mathcal{C}_{j,k}) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp \left\{ -\frac{(x_j - \mu_{j,k})^2}{2\sigma_{j,k}^2} \right\}, \quad (4.5)$$

where $\mu_{j,k}$ and $\sigma_{j,k}$ are the sample mean and variance for the feature x_j in class \mathcal{C}_k .

Clustering

The GMM and k-means are unsupervised learning algorithms. Their respective objective functions group the samples in clusters represented by conditional likelihood functions, and then a posterior distribution for each class \mathcal{C}_k is computed with the likelihood and a prior distribution of the labels. Thereby, the inference level applied is MAP. K-means can be considered as a simplification of the GMM.

Gaussian Mixture Model. Feature distributions can be approximate by a mixture of multivariate normal distributions $\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. Under the hypothesis that a sample \mathbf{x}_i belongs to class \mathcal{C}_k , its class conditional likelihood is

$$f(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}_k|}} e^{\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)\}}, \quad (4.6)$$

where $\boldsymbol{\Sigma}_k$ is regularized to avoid an ill-conditioned covariance matrix such as $\boldsymbol{\Sigma}_k \triangleq \boldsymbol{\Sigma}_k + \varepsilon \mathbf{I}_{d \times d}$, ε is the regularization hyperparameter. The number of distributions k (i.e. clusters) is equivalent to the number of classes \mathcal{C}_k , henceforth $k = 2$ in our application (i.e. clear or cloudy pixel).

The expected complete data log-likelihood is defined as [239],

$$\mathcal{Q}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t-1)}) = \sum_{i=1}^N \sum_{k=1}^K \gamma_{i,k} \log \pi_k + \sum_{i=1}^N \sum_{k=1}^K \gamma_{i,k} \log p(\mathbf{x}_i | \boldsymbol{\theta}^{(t)}) \quad (4.7)$$

where $\gamma_{i,k} \triangleq p(y_i = k | \mathbf{x}_i, \boldsymbol{\theta}^{(t-1)})$ is the responsibility of the cluster k in the sample i .

The parameters in the clustering of multivariate normal distributions can be directly computed applying the Expectation Maximization (EM) algorithm. In the E stage of the algorithm a prior is established and then, by using the likelihood function (4.6), a posterior $\gamma_{i,k} = p(\mathcal{C}_k | \mathbf{x}_i)$ can be assigned to each sample. In the M stage, the mean and variance of each cluster that maximize the log likelihood are computed as

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^N \gamma_{i,k} \mathbf{x}_i}{\gamma_k}, \quad \boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^N \gamma_{i,k} \mathbf{x}_i \mathbf{x}_i^\top}{\gamma_k} - \boldsymbol{\mu}_k \boldsymbol{\mu}_k^\top. \quad (4.8)$$

The priors are updated as well using the posterior probabilities that are

$$\pi_k = p(\mathcal{C}_k) = \frac{1}{N} \sum_{i=1}^N \gamma_{i,k}, \quad (4.9)$$

where N is the number of available samples. A class is assigned to each sample by MAP criteria $\hat{y}_i = \underset{k}{\operatorname{argmax}} p(\mathcal{C}_k | \mathbf{x}_i, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

The theory behind mixture models, as well as the EM algorithm, is fully developed in [239].

k-means. The k-means algorithm can be seen as a particularization of the algorithm above, where the posteriors $\gamma_{i,k}$ are approximated by 1 if distance $\|\mathbf{x}_i - \boldsymbol{\mu}_k\| < \|\mathbf{x}_i - \boldsymbol{\mu}_{k'}\|$, $k \neq k'$, and zero otherwise. The mean is computed as in Eq. (4.8), and the covariance is approximated by an identity matrix.

Markov Random Fields

The energy function of a MRF is composed of two functions [200]. The function φ that is the joint distribution of a class, and the function ψ that is the potential energy of the system's configuration (a term from statistical mechanics),

$$\mathcal{E}(y_i, \mathbf{x}_i) = \sum_i \varphi(\mathbf{x}_i, y_i) + \sum_{i,j} \psi(y_i, y_j), \quad (4.10)$$

where \mathbf{x}_i is the feature vector of sample i and y_i is its class. In the graph G , a sample i has a set of neighboring pixels, and each neighboring sample j has class y_j .

Sample \mathbf{x}_i is classified using the Bayes' theorem as

$$p(y_i = \mathcal{C}_k \mid \mathbf{x}_i, \boldsymbol{\theta}_k) \propto p(\mathbf{x}_i \mid y_i = \mathcal{C}_k, \boldsymbol{\theta}_k) p(y_i = \mathcal{C}_k). \quad (4.11)$$

where the corresponding likelihood is approximated by a normal distribution $\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ of class \mathcal{C}_k , and $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ are the parameter set of the feature distribution in class \mathcal{C}_k . The log-likelihood of class \mathcal{C}_k is defined as $\varphi(\mathbf{x}_i, y_i) \triangleq \log p(\mathbf{x}_i \mid y_i = \mathcal{C}_k, \boldsymbol{\theta}_k)$ in the energy function (4.10). The prior can be expressed as,

$$p(y_i) = \frac{1}{Z} \exp(-\psi(y_i)), \quad (4.12)$$

where Z is the partition function for normalization. By applying the Hammersley–Clifford theorem [130], the potential function $\psi(y_i)$ in the exponential form can be factorized in cliques of a graph G . A clique is defined as a set of nodes that are all neighbors of each other [239]. In this way, the potential function can be independently evaluated for each clique in the factorized graph,

$$\psi(y_i) = \sum_{\ell=1}^L \sum_{i,j \in \Omega_\ell} y_i \beta y_j, \quad (4.13)$$

where the set of maximal cliques in the graph is defined as $\Omega = \Omega_1 \cup \Omega_2 \cup \dots \cup \Omega_L$, ℓ represents the order of the neighboring pixels to sample i in the graph network G , and Ω_L is the maximal clique as it cannot be made any larger without losing the clique property [239]. The cliques considered in our problem are Ω_1 and Ω_2 , which represent the 1st and 2nd order neighborhood cliques respectively. Hyperparameter β needs to be cross-validated.

By applying expression (4.12) in the logarithm of (4.11), the energy function for a pixel i of class y_i and features \mathbf{x}_i is

$$\mathcal{E}(y_i = \mathcal{C}_k \mid \mathbf{x}_i, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{2} \log |\boldsymbol{\Sigma}_k| - \frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k) + \psi(y_i). \quad (4.14)$$

plus constant terms. Similarly to Eq. (4.6), the covariance matrix Σ_k in a MRF is also regularized as $\Sigma_k \triangleq \Sigma_k + \varepsilon \mathbf{I}_{D \times D}$. Finally, probability (4.11) can be written as

$$p(y_i = \mathcal{C}_k \mid \mathbf{x}_i, \boldsymbol{\theta}_k) = \frac{\exp \mathcal{E}(y_i = \mathcal{C}_k \mid \mathbf{x}_i, \boldsymbol{\theta}_k)}{\sum_{k=1}^K \exp \mathcal{E}(y_i = \mathcal{C}_k \mid \mathbf{x}_i, \boldsymbol{\theta}_k)}. \quad (4.15)$$

A class \mathcal{C}_k is assigned to the sample \mathbf{x}_i by the MAP criterion.

Iterated Conditional Modes. Parameters $\boldsymbol{\theta}_k$ in a MRF can be inferred with the ICM algorithm [27]. The algorithm initially assigns a class to each pixel from a uniform distribution. The samples with label \mathcal{C}_k are defined within the set $S_k^{(0)}$. At iteration $t + 1$ the mean and covariance of a class are computed as

$$\begin{aligned} \boldsymbol{\mu}_k^{(t+1)} &= \frac{1}{|S_k^{(t)}|} \sum_{\mathbf{x}_{i,j,z} \in S_k^{(t)}} \mathbf{x}_{i,j,z}, \\ \Sigma_k^{(t+1)} &= \frac{1}{|S_k^{(t)}| - 1} \sum_{\mathbf{x}_{i,j,z} \in S_k^{(t)}} \left(\mathbf{x}_{i,j,z} - \boldsymbol{\mu}_k^{(t+1)} \right)^\top \left(\mathbf{x}_{i,j,z} - \boldsymbol{\mu}_k^{(t+1)} \right). \end{aligned} \quad (4.16)$$

A class is reassigned to each pixel according to the parameters computed at iteration $t + 1$ with the MAP criterion

$$y_{i,j}^{(t+1)} = \underset{k}{\operatorname{argmax}} \mathcal{E}(y_{i,j}^{(t)} \mid \mathbf{x}_{i,j}, \boldsymbol{\mu}_k^{(t+1)}, \Sigma_k^{(t+1)}), \quad (4.17)$$

when the total energy stops increasing, so that $\sum_{i,j} \mathcal{E}(y_{i,j}^{(t+1)} \mid \mathbf{x}_{i,j}, \boldsymbol{\theta}_k^{(t+1)}) \leq \sum_{i,j} \mathcal{E}(y_{i,j}^{(t)} \mid \mathbf{x}_{i,j}, \boldsymbol{\theta}_k^{(t)})$, the algorithm has converged to a stable configuration and the optimal set of parameters $\boldsymbol{\theta}_k$ have been found. The distribution of class \mathcal{C}_k is defined as $\mathcal{N}(\boldsymbol{\mu}_k^{(t)}, \Sigma_k^{(t)})$.

Simulated Annealing. The standard optimization goes through all the pixels calculating their potential and classifying them in each iteration of the algorithm. The computational cost of this method is high, but we can assume that it is not necessary to evaluate the pixels whose state has high energy, because their classification will not

change. The computation cost can be reduced by sampling the pixels that are likely to be misclassified, and applying the optimization procedure only to them.

We propose to optimize the configuration of the pixels in an IR image applying the SA algorithm [178] to the MRF models [174]. SA algorithm is applied on the implementation, after the inference of the class distributions.

The class distributions $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ were previously inferred applying a supervised or unsupervised learning algorithm. The optimization is initialized to MLE classification of the pixels $y_{i,j}^{(0)} = \underset{k}{\operatorname{argmax}} p(y_{i,j} = \mathcal{C}_k \mid \mathbf{x}_{i,j}, \boldsymbol{\theta}_k)$.

The likelihood a pixel to belong a class \mathcal{C}_k is only evaluated at the initialization of the algorithm.

The objective is to evaluate the potential function of the samples that have low energy. For that, a sample $\mathbf{x}_{i,j}$ with label $y_{i,j} = \mathcal{C}_k$ is randomly selected and its classification is changed in each iteration t , so that $\bar{y}_{i,j}^{(t)} = -y_{i,j}^{(t)}$. The probability of selecting a sample $\mathbf{x}_{i,j}$ is weighted by their energy. The weights of the samples in an image are defined as,

$$w_{i,j} = \frac{\mathcal{E}(\bar{y}_{i,j}^{(t)} \mid \mathbf{x}_{i,j}^{(t)}, \boldsymbol{\theta}_k) - \max_k \mathcal{E}(\bar{y}_{i,j}^{(t)} \mid \mathbf{x}_{i,j}^{(t)}, \boldsymbol{\theta}_k)}{\sum_{i,j} \left[\mathcal{E}(\bar{y}_{i,j}^{(t)} \mid \mathbf{x}_{i,j}^{(t)}, \boldsymbol{\theta}_k) - \max_k \mathcal{E}(\bar{y}_{i,j}^{(t)} \mid \mathbf{x}_{i,j}^{(t)}, \boldsymbol{\theta}_k) \right]}, \quad (4.18)$$

and the cumulative distribution of the weights is $\bar{w}_{n,m} = \{\{\sum_{i=1}^n \sum_{j=1}^m w_{i,j}\}_{n=1}^N\}_{m=1}^M$. Then, a sample is drawn from a uniform distribution $\hat{w} \sim \mathcal{U}(0, 1)$. The sample whose weight has the minimum distance to the drawn sample, is selected $i, j = \operatorname{argmin} |\bar{w}_{i,j} - \hat{w}|$.

The algorithm follows with Metropolis step which is computed with the energy of the changed sample $\bar{y}_{i,j}$ and the energy of the original label $y_{i,j}$, $\Delta E = \mathcal{E}(y_{i,j}^{(t)} \mid \mathbf{x}_{i,j}, \boldsymbol{\theta}_k) - \mathcal{E}(\bar{y}_{i,j}^{(t)} \mid \mathbf{x}_{i,j}, \boldsymbol{\theta}_k)$.

The new label is directly accepted $\bar{y}_{i,j}^{(t)}$ iff $\Delta E < 0$. Otherwise, it will be accepted

with probability $\rho = \exp(-\Delta E/T^{(t)})$ in an analogous way to thermodynamics with the Gibbs distribution,

$$y_{i,j}^{(t+1)} = \begin{cases} \bar{y}_{i,j}^{(t)} & \text{if } \Delta E \leq 0 \\ \bar{y}_{i,j}^{(t)} & \text{if } \Delta E > 0 \text{ and } \rho > u \\ y_{i,j}^{(t)} & \text{Otherwise} \end{cases} \quad (4.19)$$

the acceptance probability is drawn from a uniform distribution $u \sim \mathcal{U}(0, 1)$.

We propose to linearly cool down the acceptance rate through the temperature hyperparameter, so that $T^{(t+1)} = \alpha T^{(t)}$. The optimal hyperparameter α is a trade off between accuracy and speed.

4.3.2 Discriminative Models

The chosen kernel is a polynomial expansion defined as $\varphi : \mathcal{X} \mapsto \mathcal{P}^n$, where n is the order of the expansion. The dimension of the output space is defined as $\mathcal{P}^n = [(n + (D - 1))!]/[n!(D - 1)]$, so when the transformation is applied to a covariate vector $\mathbf{x}_i \mapsto \varphi(\mathbf{x}_i)$, a vector is expanded to the \mathcal{P}^n -dimension space $\varphi(\mathbf{x}_i) \in \mathbb{R}^{\mathcal{P}^n}$. The polynomial expansion of the dataset $\mathcal{D} = \{\Phi, \mathbf{y}\}$, is defined in matrix form as,

$$\Phi = \begin{bmatrix} \varphi(\mathbf{x}_1) & \dots & \varphi(\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{\mathcal{P}^n \times N}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}, \quad (4.20)$$

where $y_i \in \{0, 1\}$ which are labels for a pixel representing clear or cloudy conditions, respectively. The polynomial expansion used in the primal formulated kernel for RRC, SVC and GPC is defined as,

$$\begin{aligned} \varphi(\mathbf{x}_i) &= [a_0 \ \dots \ a_j x_j \ \dots \ a_{j,k} x_j x_k \ \dots \ a_{j,k,l} x_j x_k x_l \ \dots]^\top \in \mathbb{R}^{\mathcal{P}^n}, \\ &\forall j, k, l \dots = 1, \dots, D \end{aligned} \quad (4.21)$$

where the scalar $a_0, a_j, a_{j,k}, a_{j,k,l}, \dots \in \mathbb{R}$ is chosen so that the corresponding dot product in the space can be written

$$\varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_i) = [a_0 + \mathbf{x}_i^\top \mathbf{x}_i]^n \quad (4.22)$$

4.4 Ridge Regression

RRC is an optimization problem which aims to find the parameters that minimize the mean squared error. In this model, the parameters \mathbf{w} are regularized using the quadratic norm,

$$\min_{\mathbf{w}} \sum_{i=1}^N (\mathbf{y} - \mathbf{w}^\top \Phi)^2 + \gamma \|\mathbf{w}\|_2. \quad (4.23)$$

where γ is the regularization parameter.

This model has a quadratic loss function. Therefore, the optimal parameters $\bar{\mathbf{w}}$ can be found analytically,

$$\begin{aligned} 0 &= \frac{\partial}{\partial \mathbf{w}} \left[(\mathbf{y} - \mathbf{w}^\top \Phi)^\top (\mathbf{y} - \mathbf{w}^\top \Phi) + \gamma \cdot \text{tr}(\mathbf{w}^\top \mathbf{w}) \right] \\ 0 &= 2 [\Phi (\mathbf{w}^\top \Phi - \mathbf{y}) + \gamma \mathbf{w}] \\ \bar{\mathbf{w}} &= (\Phi \Phi^\top + \gamma \mathbf{I})^{-1} \Phi \mathbf{y}. \end{aligned} \quad (4.24)$$

In this case, as the model is for classification, a sigmoid function is applied to the prediction,

$$\begin{aligned} p(\mathcal{C}_1 \mid \varphi(\mathbf{x}_*), \mathcal{D}) &= \frac{1}{1 + \exp(-\bar{\mathbf{w}}^\top \varphi(\mathbf{x}_*))} \\ p(\mathcal{C}_2 \mid \varphi(\mathbf{x}_*), \mathcal{D}) &= 1 - p(\mathcal{C}_1 \mid \varphi(\mathbf{x}_*), \mathcal{D}). \end{aligned} \quad (4.25)$$

The result is probability between 0 and 1. The classification threshold is initially set to 0.5, thus the class with higher probability is the predicted class. Lately, it is explained the method implemented to cross-validate the threshold, so that the different models have the same objective function.

4.5 Primal Solution for Support Vector Machines

We propose to solve a SVC for binary classification in the primal to limit the complexity of the model for cloud segmentation due to the large number of pixels samples [242, 353]. For the SVC, the dataset is defined as $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^D \mid \forall i = 1, \dots, N\}$ and $\mathbf{y} = \{y_i \in \{-1, +1\} \mid \forall i = 1, \dots, N\}$. The primal formulation of the SVC is the following unconstrained optimization problem,

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\| + \mathcal{C} \sum_{i=1}^N \xi(\mathbf{w}; \mathbf{x}_i, y_i) \quad (4.26)$$

which is a maximum margin problem [94]. When the ε -loss insensitive is applied to the model, the formulation is

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_2 + \mathcal{C} \sum_{i=1}^N \left(\max [0, 1 - y_i \mathbf{w}^\top \varphi(\mathbf{x}_i)] \right)^2, \quad (4.27)$$

where \mathcal{C} is the complexity parameter

The linear SVC do not have a probabilistic output. To transform the output into a probability measure, we use the distance of a sample to the hyper-plane,

$$p(\mathcal{C}_1 \mid \varphi(\mathbf{x}_*), \mathcal{D}) = \frac{1}{1 + \exp(-\bar{\mathbf{w}}^\top \varphi(\mathbf{x}_*))} \quad (4.28)$$

$$p(\mathcal{C}_2 \mid \varphi(\mathbf{x}_*), \mathcal{D}) = 1 - p(\mathcal{C}_1 \mid \varphi(\mathbf{x}_*), \mathcal{D}),$$

and the sigmoid function (as in RRC).

4.6 Primal Solution for Gaussian Processes

When a GPC is formulated in the primal, it is commonly known as Bayesian logistic regression [163, 239, 274]. As the GPC does not have an analytical solution, the Laplace approximation is applied to solve this problem,

$$p(\mathbf{w} \mid \mathcal{D}) \propto p(\mathbf{y} \mid \Phi, \mathbf{w}) p(\mathbf{w}). \quad (4.29)$$

The likelihood function $p(y_i | \Phi, \mathbf{w}) = \prod_{i=1}^N \hat{y}_i^{y_i} (1 - \hat{y}_i)^{1-y_i}$, where $\hat{\mathbf{y}} = [\hat{y}_1 \dots \hat{y}_N]^\top$ are the predictions, is a Bernoulli distribution, and the prior $p(\mathbf{w}) \sim \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ is a Normal distribution. This combination of distributions leads to a posterior that is not Gaussian. Laplace approximation assumes that the posterior is Gaussian $q(\bar{\mathbf{w}}) \sim \mathcal{N}(\mathbf{w} | \bar{\mathbf{w}}, \boldsymbol{\Sigma}_n)$.

The Marginal Log-Likelihood (MLL) is maximized to find the optimal parameters $\bar{\mathbf{w}}$,

$$\begin{aligned} \log p(\bar{\mathbf{w}} | \mathcal{D}) = & -\frac{d}{2} \log 2\pi - \frac{1}{2} \log |\boldsymbol{\Sigma}_0| - \frac{1}{2} (\mathbf{w} - \boldsymbol{\mu}_0)^\top \boldsymbol{\Sigma}_0^{-1} (\mathbf{w} - \boldsymbol{\mu}_0) + \\ & + \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)], \end{aligned} \quad (4.30)$$

where $\hat{y}_i = \sigma(\mathbf{w}^\top \varphi(\mathbf{x}_i)) = 1/[1 + \exp(-\mathbf{w}^\top \varphi(\mathbf{x}_i))]$, is the sigmoid function.

The covariance of the posterior distribution is found analytically as the inverted Hessian of the negative log-posterior,

$$\boldsymbol{\Sigma}_n^{-1} = \boldsymbol{\Sigma}_0^{-1} + \sum_{i=1}^N y_i (1 - y_i) \varphi(\mathbf{x}_i) \varphi(\mathbf{x}_i)^\top. \quad (4.31)$$

As the convolution of a sigmoid function with a Normal distribution is intractable, the sigmoid function is approximated by a probit function. The approximated predictive distribution is,

$$\begin{aligned} p(\mathcal{C}_1 | \mathcal{D}) = & \int \sigma(\boldsymbol{\alpha}) \cdot \mathcal{N}(\boldsymbol{\alpha} | \boldsymbol{\mu}_\alpha, \boldsymbol{\sigma}_\alpha) d\boldsymbol{\alpha} \\ \approx & \sigma(\phi(\boldsymbol{\sigma}_\alpha) \cdot \boldsymbol{\mu}_\alpha), \end{aligned} \quad (4.32)$$

where $\boldsymbol{\alpha} = \bar{\mathbf{w}}^\top \Phi$, the predictive mean is $\boldsymbol{\mu}_\alpha = \bar{\mathbf{w}}^\top \Phi$, and the variance is $\boldsymbol{\sigma}_\alpha^2 = \Phi^\top \boldsymbol{\Sigma}_n \Phi$. The probit approximation of a sigmoid is $\phi(\boldsymbol{\sigma}_\alpha) = (1 + \boldsymbol{\sigma}_\alpha \pi/8)^{-1/2}$. Once the probability of class \mathcal{C}_1 is computed using Eq. (4.32), the probability of class \mathcal{C}_2 is,

$$p(\mathcal{C}_2 | \varphi(\mathbf{x}_*), \mathcal{D}) = 1 - p(\mathcal{C}_1 | \varphi(\mathbf{x}_*), \mathcal{D}). \quad (4.33)$$

4.7 J-Statistic

The Younde's j-statistic or Younde's Index is a test to evaluate the performances of a binary classification [367], that is defined as,

$$J = sensitivity + specificity - 1. \quad (4.34)$$

The entries on the confusion matrix are used to compute the sensitivity,

$$sensitivity = \frac{TP}{TP + FN}, \quad (4.35)$$

where TP and FN are the true positives and false negatives, and the specificity is,

$$specificity = \frac{TN}{TN + FP}, \quad (4.36)$$

where TN and FP are the true negatives and false positives. It is different than the accuracy score of a binary classification, which is also obtained using the entries of the confusion matrix, and that it is,

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}. \quad (4.37)$$

As the optimized loss function is different in each model, we propose to define a prior λ , which has to be cross-validated for each one of models, and has an optimal value for each classification function,

$$\begin{aligned} p(\mathcal{D} | \mathcal{C}_k) &= \frac{p(\mathcal{C}_k | \mathcal{D}) p(\mathcal{C}_k)}{p(\mathcal{D})} \\ &\propto p(\mathcal{C}_k | \mathcal{D}) p(\mathcal{C}_k) \\ &\propto p(\mathcal{C}_k | \mathcal{D}) \cdot \lambda \end{aligned} \quad (4.38)$$

so the maximized loss function is the same in all the models. The classification probabilities are defined as $p(\mathcal{D} | \mathcal{C}_1) = p(\mathcal{C}_1 | \mathcal{D}) \lambda$, and $p(\mathcal{D} | \mathcal{C}_2) = 1 - p(\mathcal{D} | \mathcal{C}_1)$. The j-statistic score is maximized finding the optimal binary classification λ threshold.

For that, the j-statistic is applied to the conventional Receiver Operating Characteristic (ROC) analysis [98], and it is computed at each point of the ROC. We propose to use the maximum value of j-statistic in the ROC curve as the optimal point.

After the cross-validation of the virtual prior λ , a class \mathcal{C}_k is assigned to a sample \mathbf{x}_* following this criteria,

$$\hat{y}_* = \operatorname{argmax}_k p(\mathcal{C}_k | \mathbf{x}_*, \mathcal{D}) \cdot \lambda, \quad (4.39)$$

which is a MAP estimation.

4.8 Experiments

The selected samples constitute the dataset used in the cross-validation and testing of the segmentation models. The samples include different days in all four seasons. The sample images include partially cloudy, fully clear-sky and fully covered sky conditions. The images were captured at different hours of the day, so the Sun's elevation and azimuth angle are different. Therefore, the atmospheric background model is different in all of the images. The dataset is composed of different types of clouds found at different heights in the tropopause. We found that the most difficult clouds for the models to classify are cirrus stratus, which are also included in the dataset. Artificially created clouds like contrails are also included in testing set. Contrails are highly difficult for the models, as the scattering effect is similar to that produced by cirrus clouds. The dataset is composed of 12 images with labels, amounting to a total of 57,600 pixels. They are organized chronologically and divided into training (earlier dates) and testing set (later dates). The training set has 7 images, which are 33,600 pixels in total. The testing set has the remaining 5 images, which are 24,000 pixels. The training set contains 5 images with clouds, 1 image with clear-sky, and another one with covered sky conditions. The testing set has 3 images with clouds, 1 with

clear-sky, and 1 with covered sky conditions.

Table 4.1: Type of clouds, percentage of cloud covered and the Sun’s position in the horizon in each IR image of the training and test sets.

	Type of Cloud	Cloud Covered [%]	Elevation [°]	Azimuth [°]
Train No. 1	Stratocumulus and Cumulus	38.67	29.69	160.91
Train No. 2	Stratocumulus	28.13	28.70	157.49
Train No. 3	Cirrocumulus and Stratocumulus	36.6	24.43	146.99
Train No. 4	Alto cumulus	4.5	31.40	183.09
Train No. 5	Cumulus	37.94	73.52	172.20
Train No. 6	Nimbus	100	29.92	164.81
Train No. 7	Clear-Sky	0	28.94	158.59
Test No. 1	Contrail	38.67	48.76	183.94
Test No. 2	Cumulus	28.13	37.53	149.34
Test No. 3	Alto cumulus	12.29	32.10	204.17
Test No. 4	Clear-Sky	0	76.58	190.5
Test No. 5	Altostratus	100	60.07	165.62

The Leave-One-Out (LOO) method is implemented in the cross-validation of the parameters. In this method, the training samples are left out for validation one at a time, while the rest of the training samples are used to fit the model. In our problem, the training samples are the images in the training set, so a training image is used for validation while the others are used for training the model.

The cross-validation is done using a HPC. Each validation sample (in the LOO routine) runs on a different Central Processing Unit (CPU), and 7 CPUs are necessary for each experiment. When the LOO routine is finished, the results are communicated to the main node, and a new set of hyperparameters and virtual prior λ are validated. This procedure is repeated until all possible combinations of hyperparameters and virtual priors are validated. The LOO routine runs in multiple experiments at the same time. Each experiment has a combination of feature vectors, neighborhoods, polynomial expansions (in the discriminative models) and cliques (in the MRF models). All CPUs are operating at full capacity and are only inactive during the waiting time (i.e. until all jobs of the LOO routine are finished).

The cross-validation is computationally expensive due to the amount of training samples, but running the LOO routine and the experiments in parallel reduces the

training time by several orders of magnitude. The testing times are obtained when running each segmentation model in a single CPU.

Exploratory results showed that the features that work best are those in vectors $\mathbf{x}_{i,j}^3$ and $\mathbf{x}_{i,j}^4$. All possible combinations were tested, but none produced any improvement in the classification performance, with the exception of those in $\mathbf{x}_{i,j}^4$. However, the original features require preprocessing to achieve reasonable performances (see Figure 4.1). This is shown in the classification results obtained by $\mathbf{x}_{i,j}^1$ and $\mathbf{x}_{i,j}^2$.

In the generative models, NBC and k-means clustering do not have hyperparameters. The GDA and GMM have the covariance matrix regularization term γ which has to be cross-validated. In the k-means clustering, the feature vectors were standardized $\bar{\mathbf{x}}_{i,j} = [\mathbf{x}_{i,j} - \mathbb{E}(\mathbf{X})]/\mathbb{V}^{1/2}(\mathbf{X})$. The rest of the models neither required normalization nor standardization of the feature vectors.

In the MRF models, the cliques potential β in Eq. (4.13) was cross-validated in all the models. The supervised MRF have the covariance matrix regularization term γ which was cross-validated. The unsupervised ICM-MRF is computationally expensive, so the regularization term of the covariance matrix was set to $\gamma = 1$. In the supervised MRF with SA in the implementation, the cross-validated parameters were the regularization term of the covariance matrix γ , and the cooling parameters α . In the unsupervised MRF trained with the ICM algorithm (using the SA algorithm in the implementation), the parameters of the regularization term of the covariance matrix and cooling were set to $\gamma = 1$ and $\alpha = 0.75$.

In the discriminative models, the RRC has the regularization γ in Eq. (4.23) that has to be cross-validated. The SVC has the complexity term \mathcal{C} of the loss function in Eq. (4.27). The hyperparameters of the GPC are the prior mean $\boldsymbol{\mu}_0$ and the covariance matrix $\boldsymbol{\Sigma}_0$. The prior mean and covariance matrix are simplified to $\boldsymbol{\mu}_0 \triangleq \mathbf{0}$ and $\boldsymbol{\Sigma}_0 \triangleq \mathbf{I}_{D \times D} \cdot \gamma$, so only the parameter γ is cross-validated.

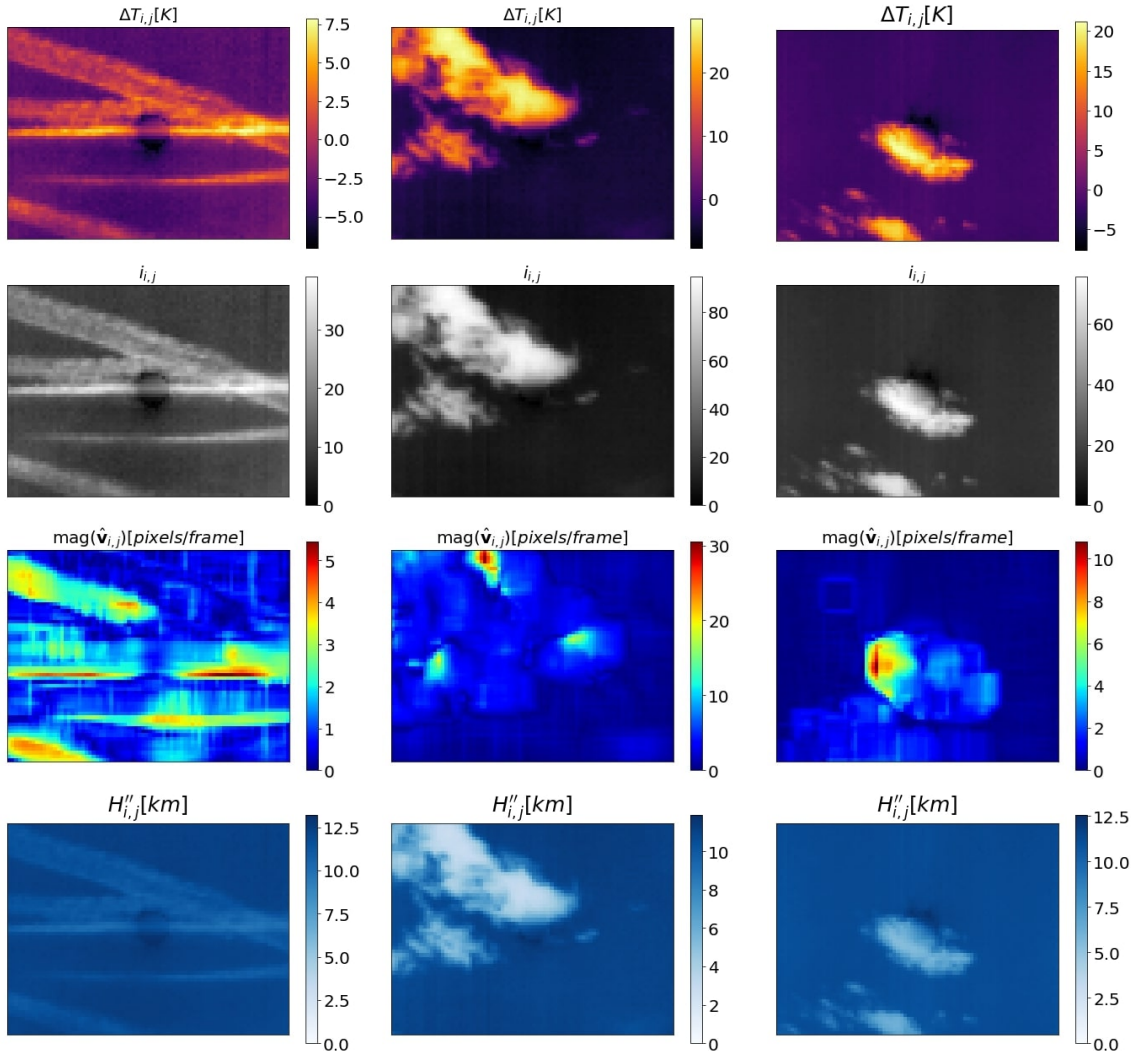


Figure 4.1: This figure shows the features extracted from three test images. The test images are organized in columns. The images in the first row show the normalized intensity of the pixels. The images in the second row show the magnitude of the velocity vectors. The images in the third row show the increments of temperature with respect to the height of the tropopause. The images in the fourth row show the height of the clouds. The last row shows the test images in which the clouds were manually segmented.

In addition to each set of hyperparameters, all models have a virtual prior λ that corrects possible class-imbalances in Eq. (4.38). The hyperparameters and the virtual

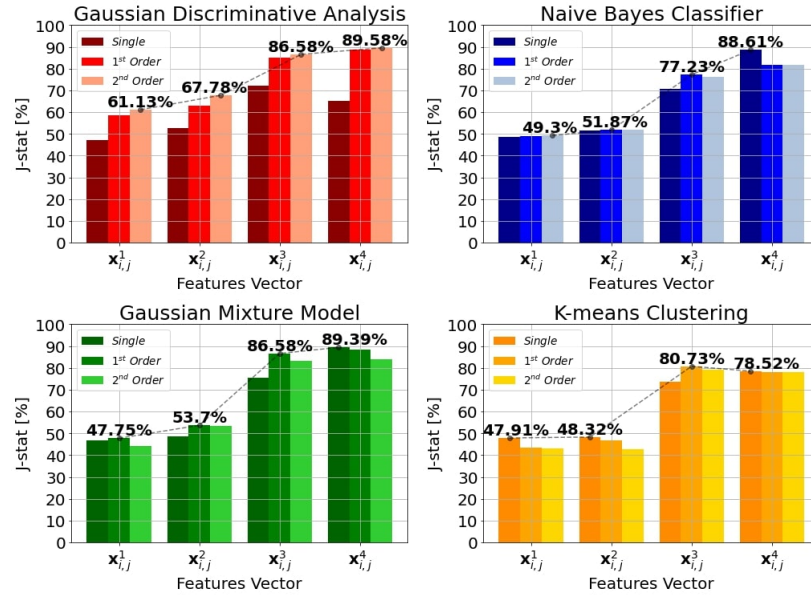


Figure 4.2: The graph shows the j-statistic achieved by the generative models. The color of the bars in the graph indicate the order of neighborhood from dark to light. The neighborhoods are organized from the left to right within the groups of bars. This corresponds with the order of the feature vectors used in the model.

prior λ have to be cross-validated. A set of hyperparameters define the ROC curve, and the virtual prior λ is used to find the optimal j-statistic along this curve with the predicted probabilities of each class for each combination. The validation j-statistic is the average of the j-statistics obtained in each LOO cross-validation loop. The model selection criteria is the highest validation j-statistic.

The experiments were carried out in the Wheeler HPC of the UNM-CARC, which uses a SGI AltixXE Xeon X5550 at 2.67GHz with 6 GB of RAM per core, 8 cores per node, 304 nodes total, and runs at 25 theoretical peak FLOPS. Linux CentOS 7 is installed.

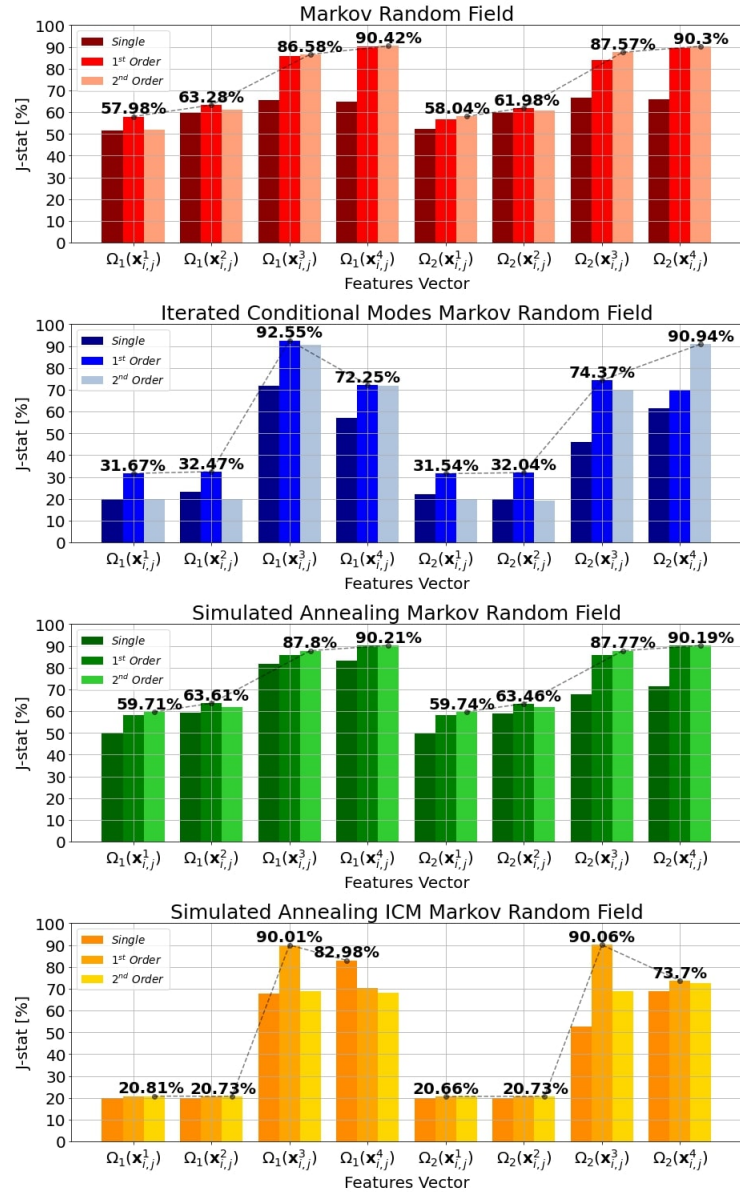


Figure 4.3: The graphs show the j-statistic achieved by the MRFs using different cliques in their potential function. The four feature vectors are organized in groups of three bars. There are two groups of feature vectors: those with a potential function of 1st order cliques $\Omega_1(\cdot)$, and those with a potential function of 2nd order cliques $\Omega_2(\cdot)$.

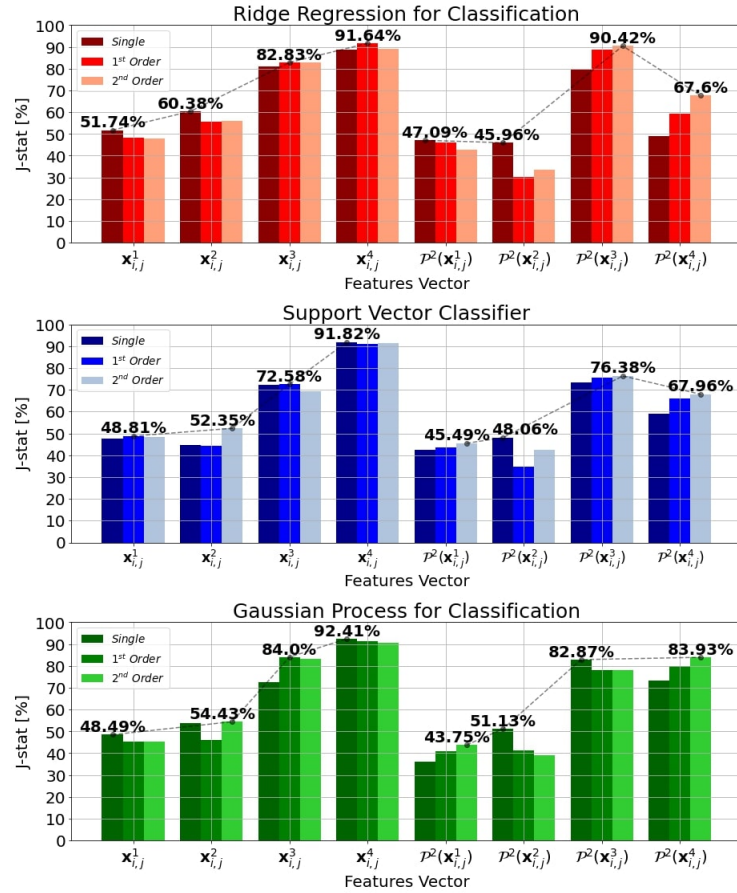


Figure 4.4: The graphs show the j-statistics achieved by the discriminative models. The feature vectors are organized in groups. The bars in the same group from dark to light are: features extracted from a single pixel, a 1st order neighborhood and a 2nd order neighborhood. When a polynomial expansion of the second order is applied to the feature vectors, it is denoted as $\mathcal{P}^2(\cdot)$.

4.9 Discussion

The segmentation performed on three testing images by the generative models are shown in Figures 4.5-4.7. The NBC and GDA are both discriminant analysis and supervised learning models (Figure 4.6). The k-means and GMM are unsupervised learning methods (Figure 4.6). The MRF and SA-MRF are supervised learning models and ICM-MRF and SA-ICM-MRF are unsupervised learning models. The

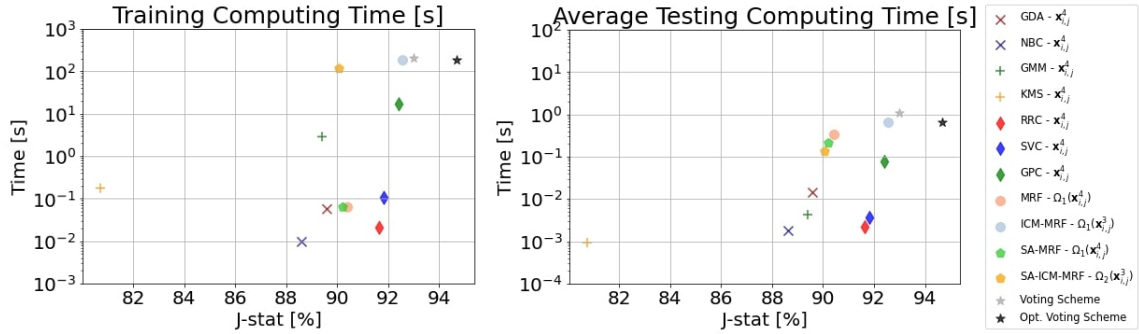


Figure 4.5: Left: Computing time of each model during training. Right: Average computing time during the segmentation in the test subset. The legend displays the optimal feature vectors, neighborhood order, polynomial expansion and cliques of each model.

SA algorithm is implemented to speed-up the MRF and ICM-MRF convergence. When MRF models use the SA algorithm, the segmentation is not so uniform (Figure 4.7). The cooling mechanism of the SA algorithm ends the optimization before the segmentation has converged to a state of higher energy. The discriminative models used are the RRC, SVC and GPC (Figure 4.8). These were solved in the primal formulation so their performances are feasible for real-time cloud segmentation (see Figure 4.5). The performances of the models are compared in terms of j-statistic vs. training computing time vs. average computing time in testing. The j-statistic is evaluated with the images in the testing subset. The computing time is measured in seconds. The time in the y-axes of the graphs shown in Figure 4.5 are displayed in logarithmic scale. The highest j-statistic is achieved by the unsupervised MRF, but the training and the average testing computing time are the largest. NBC and RRC have the lowest training times. In the implementation, the k-means, NBC and RRC have the lowest computing time. If we have considered all of this information, the most suitable model would be one of these model.

The unsupervised MRF model (ICM-MRF) achieved the highest j-statistic in testing among generative and discriminative models. The ICM-MRF model uses the

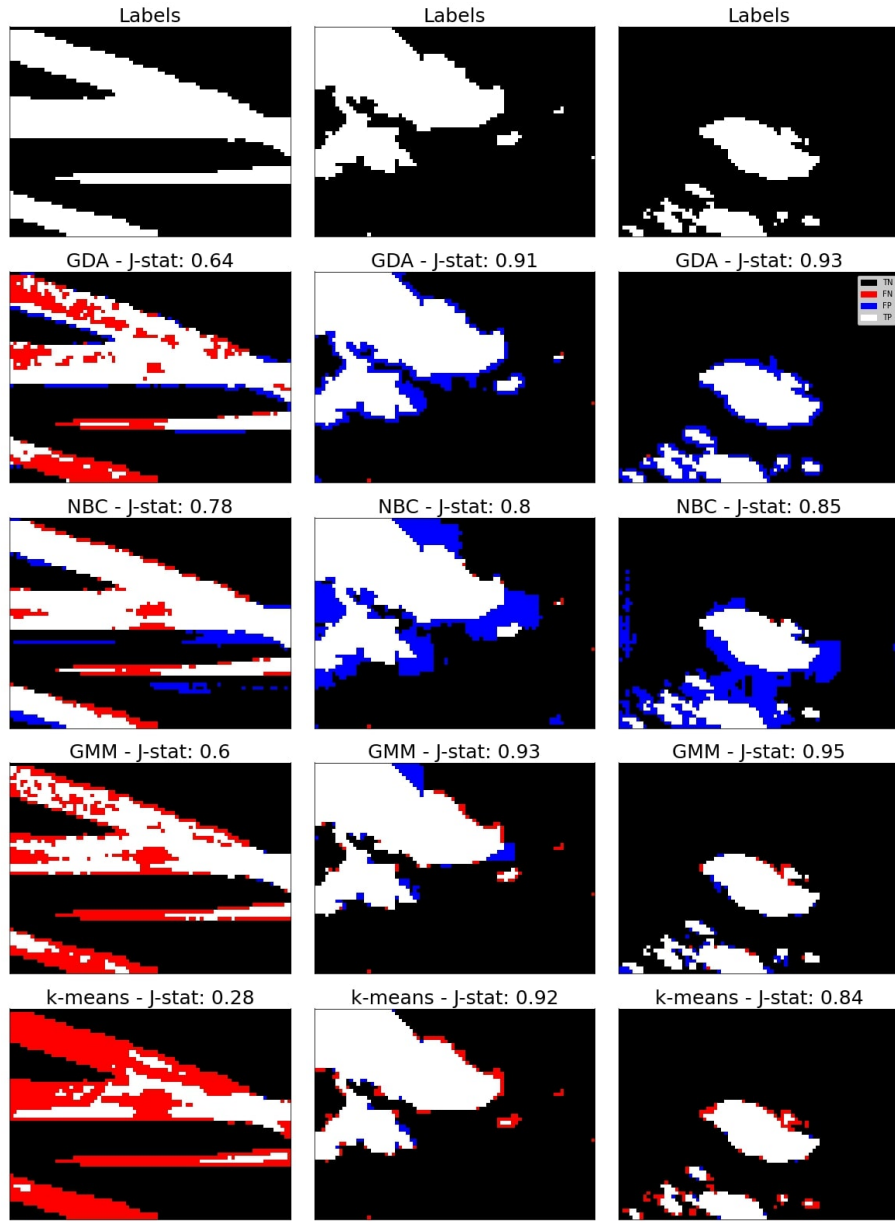


Figure 4.6: Three images from the test organized in columns. The images in each row show the segmentation performed by a generative model. The higher j-statistic was achieved by the NBC in the first image, and the GMM in the second and third images.

feature vector \mathbf{x}^3 with a 1^{st} order neighborhood and the set of cliques Ω_1 in the prior. The classification performance of the model decreased when optimized using the SA

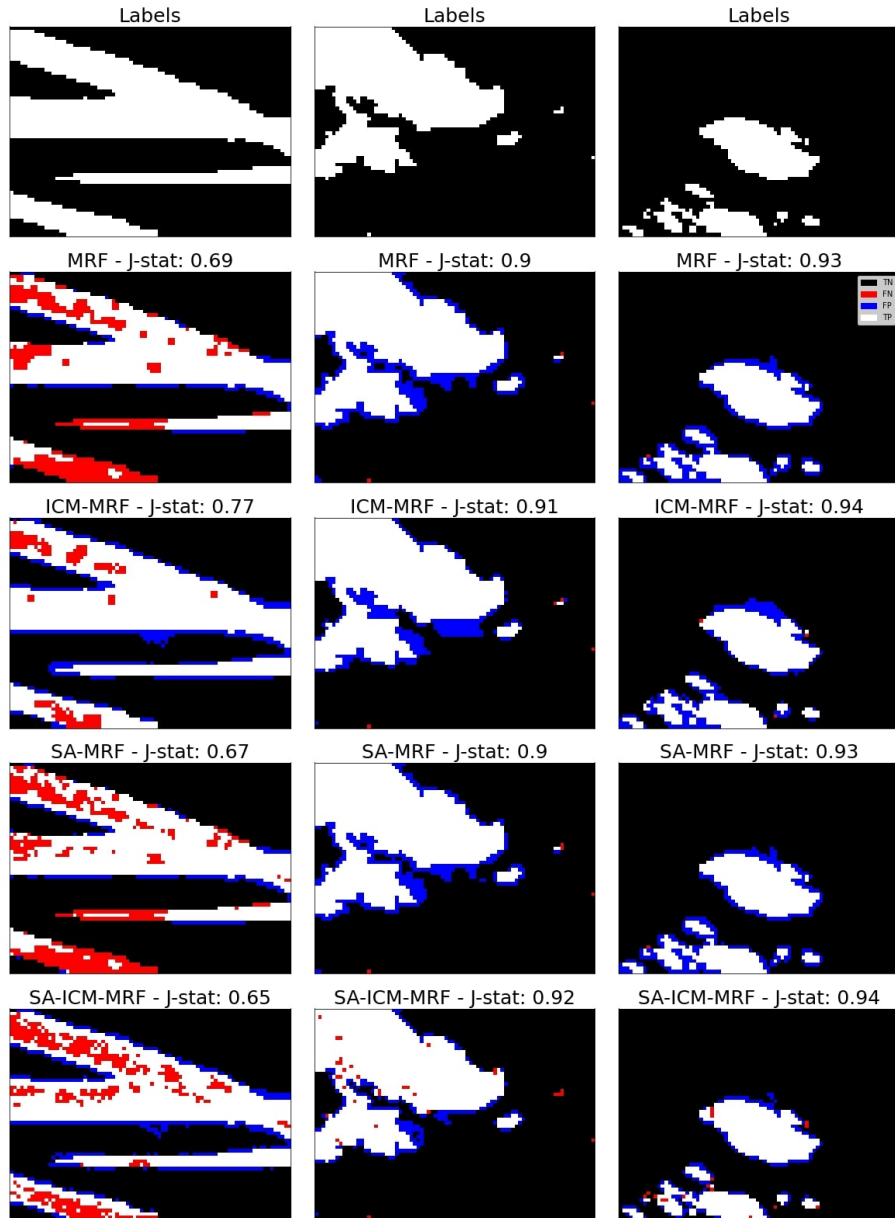


Figure 4.7: Three images from the test subset organized in columns. The images in each row show the segmentation performed by a MRF model. The highest j-statistic was achieved by ICM-MRF in the first image, SA-ICM-MRF in the second, and ICM-MRF and SA-ICM-MRF in the third image.

algorithm, but the average testing time was faster (Figure 4.3). The MRF models

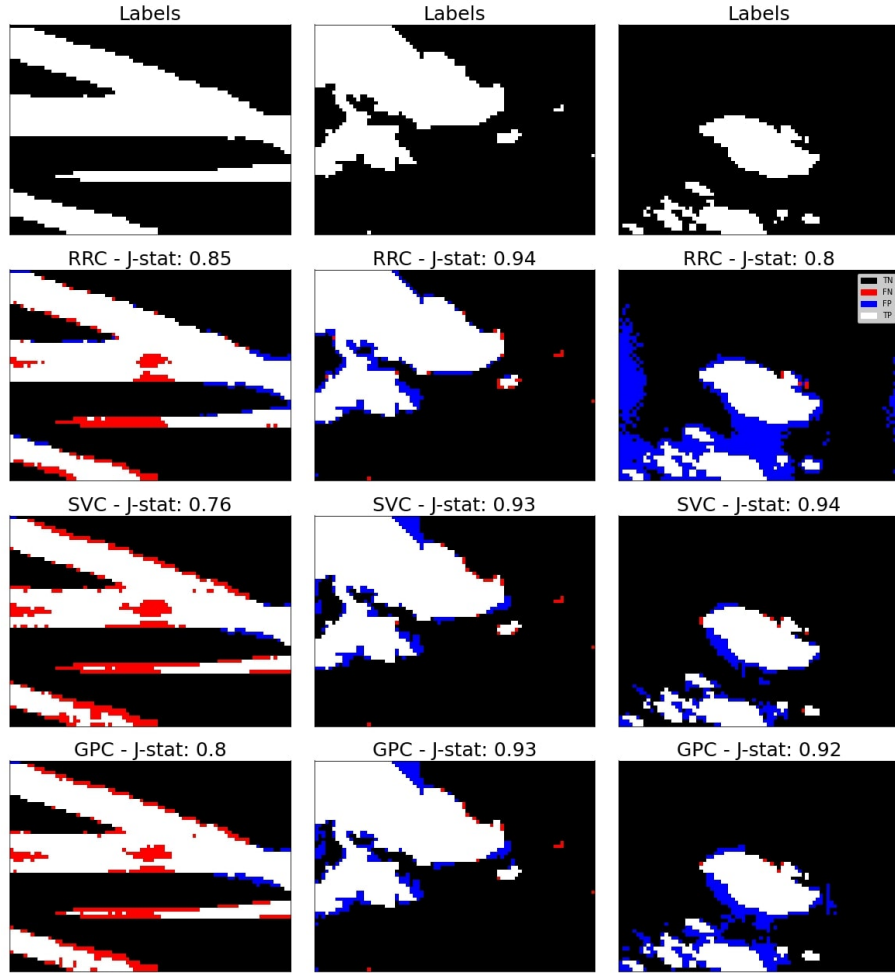


Figure 4.8: Three images from the test subset are organized in columns. The images in each row show the segmentation performed by the discriminative models. When segmenting the images, a higher j-statistic was achieved by RRC (in the first and second image), and SVC (in the third image).

that use a prior potential function lead to the largest training and average testing computational time. When only generative models without the prior potential function are considered (NBC, GDA, k-means and GMM), the GDA has the highest j-statistic with the feature vector \mathbf{x}^4 of a 2^{nd} order neighborhood (Figure 4.2). However, if the trade-off between average testing time and j-statistic is considered, the most suitable generative model is the GMM with a feature vector \mathbf{x}^4 of a single pixel neighborhood.

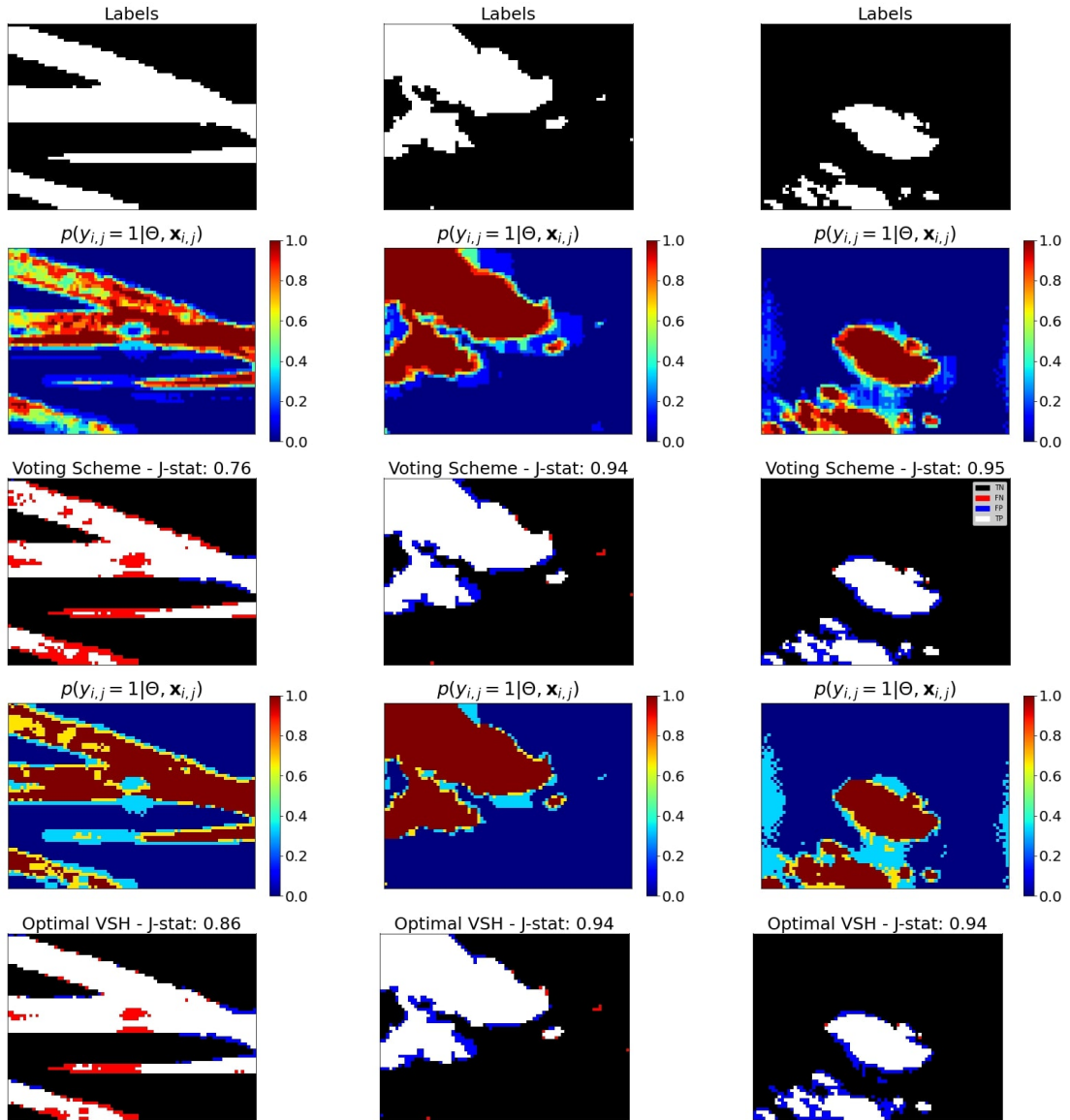


Figure 4.9: Three different test images. First and second rows: results of the voting scheme. The first row displays the probability of a pixel belonging to a cloud. The second row shows the segmentation performed by the voting scheme. Third and fourth rows: probability of a pixel belonging to a cloud and the segmentation of the optimal voting scheme (VSH).

The generative models which include a simplification of the covariance matrix and that do not use a prior potential function (NBC and k-means) yield the fastest average

testing time among all classification models (without considerably decreasing the j-statistic). However, these models have the lowest j-statistic among all the models implemented. Figure 4.4 shows the discriminative models' j-statistics. The average testing time is lower than that obtained by the generative MRF models, but the j-statistic is higher than that obtained by the generative models without the potential function. The polynomial expansion yields overfitting in all the discriminative models. The discriminative model that achieved the highest j-statistic is the linear GPC with the feature vector \mathbf{x}^4 of a single pixel neighborhood. As seen in Figure 4.5, the RRC and SVC are the most suitable methods, as they offer the best compromise between average testing times and accurate segmentation.

The results in Figures 4.2-4.4 show the importance of the feature extraction method in cloud image segmentation. The extraction of features makes it easier for the models to differentiate between cloudy and clear-sky pixels, because the distance between feature vectors of different classes increases in the feature space. Through feature extraction, the feature vectors of the same classes group together forming clusters in the feature space. Without extracting features correctly, the feature vectors from both classes (cloudy and clear-sky) are grouped in a single cluster, making it difficult to perform a classification. When the magnitude of velocity vectors are included in the feature vectors, combined with temperature increments and normalized temperature increments, the segmentation models achieved a higher j-statistic. The addition of features from neighboring pixels to the feature vectors improves the performance in some of the models.

When the raw temperature and height are used, all models have poor performance. However, when the images are preprocessed with the outdoor germanium camera window model and the atmospheric model, the ICM-MRF reaches a reasonable performance of 92.55 % at the expense of a high computational cost of 641 ms per image in testing. The performance of discriminative methods with this set of features

is lower, ranging between 72.58 % and 84 %. When velocity vectors are added to the features, the discriminative methods achieve a similar performance as the ICM-MRF, with computational times of 2.2 ms (RRC), 3.7 ms (SVC) and 77 ms (GPC). The best compromise is the SVC, which is 150 times faster than the ICM-MRF with a small difference in accuracy. The image preprocessing and feature extraction time is 0.1 ms for \mathbf{x}^1 , 4.7 ms for \mathbf{x}^2 , 99.9 ms for \mathbf{x}^3 and 1079 ms for \mathbf{x}^4 . When preprocessing time is added to the segmentation time, the average time required by the ICM-MRF is 740.9 ms. This is faster than the average time required by the discriminative models: 1081 ms (RRC), 1083 ms (SVC) and 1156 ms (GPC).

A voting scheme using the predictions from the models displayed in Figure 4.8-4.8 (not including SA-MRF and SA-ICM-MRF) achieved higher j-statistic but have a higher computing time. The j-statistic is 93 %, see Figure 4.9. The combination of the RRC, SVC and ICM-MRF lead to the best j-statistic. The optimal voting scheme reached a j-statistic of 94.68 % in testing (see Figure 4.9). The voting scheme's training and testing times are the sum of each method's respective computing times. When the models are trained and tested in parallel, the voting scheme's training and testing times are that of the slower models.

4.10 Conclusion

This chapter seeks to find the optimal methods for real-time ground-based IR cloud segmentation through image preprocessing and feature extraction. Preprocessing was applied to remove underlying cycle-stationary processes, and feature extraction was used to compute cloud height and velocity. The results show that cloud segmentation in ground-based IR images is not only feasible, but achieves high performance in real-time applications. Ground-based IR cameras perform better than visible ones in poor light conditions. We implement a preprocessing algorithm that uses physical features

extracted from IR images. The j-statistic is proposed to independently measure the accuracy of the classification in each classes.

Preprocessing the ground-based IR images using the window and atmospheric models leads to an overall performance improvement. Simplification of the covariance matrix reduces the computing time, but the j-statistic achieved is lower than that of the models using the full covariance matrix. Adding the features of neighboring pixels to the feature vectors yields an increase in segmentation performance in some cases. The discriminative models formulated in the primal result in feasible segmentation models for real-time application. MRF models remove possible outliers using cliques from neighboring pixels. This increases the overall performance of the generative models when trained with unsupervised and supervised algorithms. The optimal voting scheme achieved the best j-statistic. However, the implementation computing time might be slow for real-time applications when not run in parallel.

Further investigations may focus on segmentation in multiple layers of clouds. The clouds in each layer may be segmented into different classes. An algorithm can be trained to detect multiple layers of clouds when clouds have different heights or directions. In this way, the extraction of features may be performed independently in each one of the cloud layers. A multiple cloud layer segmentation algorithm will reduce the noise when extracting features. This algorithm may be implemented to increase the performance of ground-based intra-hour GSI forecasting.

Chapter 5

Geospatial Perspective Reprojections for Sky Imaging Systems

5.1 Introduction

The horizons of intra-hour solar forecasting depend on the FOV of the sky imager used to acquire the images. A sky imager may be composed of one or multiple visible or IR imagers, or both, and their FOV generally varies from 60° (low) to 180° (large). However, unless the sky imager is mounted on a solar tracker [213, 59], the necessary FOV to perform an accurate intra-hour solar forecast is large. TSI achieved large FOV sky images using a concave mirror to reflect light beams into a visible [58] or IR camera [276], and the camera is installed on a support at the focal distance of the mirror [116, 216]. An alternative to reflective sky imagers (in visible light sky images), is to increase the camera's FOV using a fisheye lens [199, 105, 205, 54]. These are generally known as “all sky imagers” [299, 43, 133]. Similarly, the FOV of IR sky imagers can be enlarged applying image processing techniques to merge images acquired from multiple low FOV imagers [214].

Each of these sky imagers use light beams received at an angle with respect to the imager's plane. Therefore, the produced distortion should be corrected using a geometric transformation to compute the velocity vectors of a cloud. The geometric transformation proposed by [249] transforms the Euclidean coordinate system of the pixels to a coordinate system based on the azimuth and elevation angles. This transformation was implemented by [278] for reprojecting the pixels of a TSI, in the atmosphere cross-section plane, using height measurements acquired using a nearby ceilometer. Ceilometers estimate the height of clouds and have been used to validate low-cost approaches to approximate the height of a cloud using multiple all sky imagers [243, 182]. However, this device is expensive and it is not applicable to more general operations such as a cloud speed sensor [346]. Another low-cost alternative to determine the velocity of clouds moving in the atmosphere cross-section, and thus estimating their heights, was developed using an all sky imager and a grid of sensors (i.e., pyranometers) by [347].

Nevertheless, these geometric transformations were developed for static sky imagers (i.e., TSI and all sky imager). In contrast, the geospatial reprojections introduced in this chapter not only work for static sky imagers, but are also applicable to sky imagers mounted on a solar tracker. In this last case, the perspective in the images is a function of the Sun's elevation and azimuth angles. The first approximation is a reprojection for devices that do not record low elevation angles (see Section 5.3), while the second computes accurate reprojections even when the elevation angle is low (see Section 5.4). The proposed reprojections were originally developed for a low FOV sky imager mounted on a solar tracker (see Chapter 2), however, it is possible to obtain the geospatial reprojection for any FOV and elevation angle by reparameterizing the algorithms. As a ceilometer was not available, the proposed methods applies the MALR to avoid the need for ceilometer measurements. The estimation of the error in the height approximation using this method is out of the scope of this investigation.

5.2 Rectilinear Lens

The acquired image is the light beam refraction in a converging point of the emitted black body radiation. The image resolution is defined as $M \times N$ pixels. If the radiant objects (the Sun and the clouds) are at a distance $z \rightarrow \infty$, the radiation rays converge at the focal length. Consequently,

$$\frac{1}{f} = \frac{1}{z} + \frac{1}{D} \approx \frac{1}{D}, \quad (5.1)$$

where f is the focal length and D is the distance from the lens to the converging point. The relation between the diagonal FOV and the focal length f for a rectilinear lens is

$$\tan \frac{\text{FOV}}{2} = \delta \frac{N_{diag}}{2f}, \quad (5.2)$$

where $N_{diag} = \sqrt{M^2 + N^2}$ is the number of pixels in the diagonal of the sensor and δ is the pixel size. Therefore, the focal length f of camera is,

$$f = \frac{\delta}{2} \frac{N_{diag}}{\tan \frac{\text{FOV}}{2}}. \quad (5.3)$$

5.3 Flat Earth Approximation

The *flat* Earth approximation is viable without large error (when the elevation of the Sun ε_0 is higher than 30°) because the portion of the Earth's atmosphere in the FOV of the camera is much smaller than its entire surface. With this assumption, the reprojection from the sensor plane to the atmosphere cross-section plane (in Figure 5.1) is obtained with the distance z of a cloud to the camera lens. The distance z is a function of the cloud height h and the elevation angle ε of the cloud in a pixel,

$$z = \frac{h}{\sin \varepsilon}. \quad (5.4)$$

A cloud in the sky images are segmented indicating which the pixels belong to a cloud, so that $\mathbf{B} = \{b_{i,j} \in \mathbb{B} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ is a binary image where

Chapter 5. Geospatial Perspective Reprojections for Sky Imaging Systems

0 is a clear sky pixel, and 1 is cloudy pixel (see Chapter 4). The cloud height (see Chapter 3) in a frame are computed using only in the cloudy pixels,

$$h = \frac{\sum_{i,j} H''_{i,j} \cdot \mathbb{I}(b_{i,j} = 1)}{\sum_{i,j} H''_{i,j} \cdot \mathbb{I}(b_{i,j} = 1)}, \quad (5.5)$$

where $\mathbb{I}(\cdot)$ is the indicator function.

The reprojection is computed with respect to the coordinates of each pixel i, j in the imager plane. The coordinates of a pixel in the imager plane are defined as $x_j = j\delta$ and $y_i = i\delta$. In this reprojection, we assume that the elevation angle ε_i is different in each row i and constant in each column j of pixels in an image, and the differential angle α_j (formed by the position of Sun and a pixel) is different in each column j and constant in each row i of pixels. This assumption is valid since the FOV of the individual pixels in the rectilinear lens is sufficiently small. As seen in Figure 5.1, when intersecting a cloud layer, the projection of the 3D pyramid defined by the camera FOV in a 2D plane forms a triangle. The elevation ε_i and azimuth α_j angles for each pixel i, j are,

$$\begin{aligned} \varepsilon &= \left\{ \left(\varepsilon_0 + i \frac{\nu}{2} \right) \mid \varepsilon_i \in \mathbb{R}^{(0, \pi]}, \forall i = -\frac{M}{2}, \dots, \frac{M}{2} \right\}, \\ \alpha &= \left\{ \left(\alpha_0 + j \frac{\nu}{2} \right) \mid \alpha_j \in \mathbb{R}^{(0, \frac{\alpha_x}{2}]}, \forall j = -\frac{N}{2}, \dots, \frac{N}{2} \right\}, \end{aligned} \quad (5.6)$$

where $\nu = [\text{FOV}/\sqrt{M^2 + N^2}] \cdot [\pi/180]$ is the camera ratio in radians per pixel, ε_0 is the Sun's elevation angle, and $\alpha_0 = 0$. Therefore, $\alpha_j = 0$ and $\varepsilon_i = \varepsilon_0$ represent the center of the image (since $\alpha_0 = 0$), but only when the number of pixels M and N are odd numbers. For all pixels, ν is approximated by a constant. In this way, the FOV is $\alpha_x = \nu M$ and $\alpha_y = \nu N$ in the x and y axis respectively.

The length of a row of pixels j reprojected in the atmosphere cross-section is $x'_{i,j} = x_j \cdot z_i/f$, so substituting z_i in Eq. (5.4), the coordinates of the imager plane

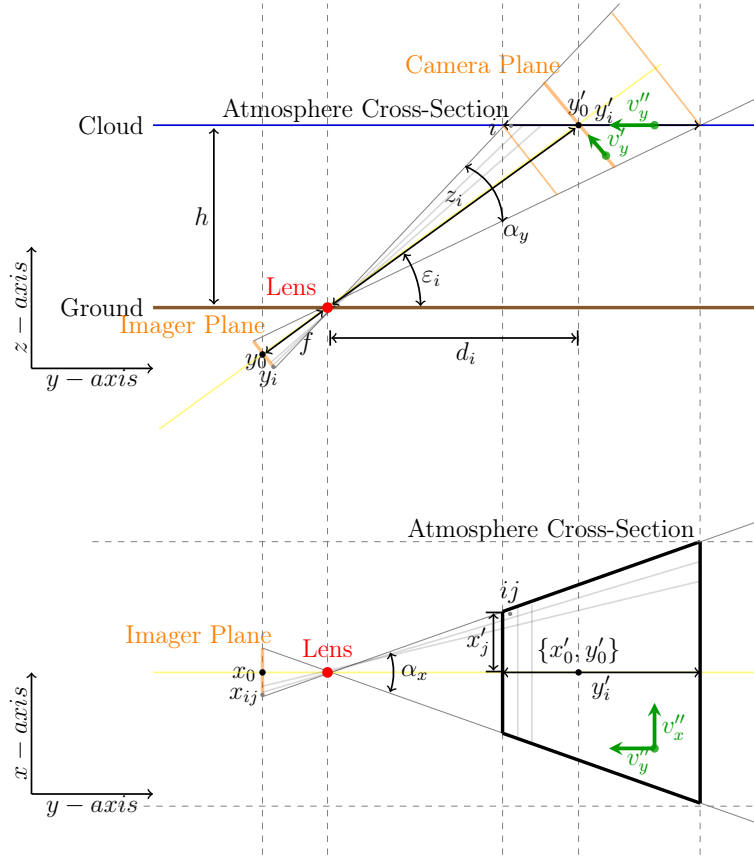


Figure 5.1: *Flat Earth* geospatial reprojection. The top one corresponds to the geospatial reprojection y-axis or side view, which shows how the reprojection depends on the distance z_i of an object to the imager, the height h and the elevation ϵ_i . The bottom one is the geospatial reprojection x-axis or top view, which shows the relation of the angular increments α used to compute the elevation angle ϵ_i of each one of the pixels ij in the image. The velocity decomposition $\mathbf{v}' = \{v'_x, v'_y\}$ shows that cloud velocity components have a perspective distortion in the x-axis and in the y-axis, due to the camera plane inclination of ϵ degrees with respect to the normal. x'_j and y'_i represent the coordinates of the pixel in the image (see in Eq. (5.7)). When the coordinate system is centered applying Eq. (5.21), $\mathbf{x}_0 = \{x'_0, y'_0\}$ represent the origin of coordinates.

reprojected in the atmosphere cross-section are,

$$\begin{aligned} x'_{i,j} &= \frac{x_j}{f} \cdot z_i = \frac{x_j}{f} \cdot \frac{h}{\sin \epsilon_i} \\ y'_i &= \frac{y_i}{f} \cdot z_i = \frac{y_i}{f} \cdot \frac{h}{\sin \epsilon_i}. \end{aligned} \tag{5.7}$$

5.4 Great Circle Approach

The atmosphere cross-section plane can be approximated more exactly using the pyramid formed by the camera FOV when intersects a cloud layer at height h in point D in Figure 5.2. The assumption is that the Earth and the cloud layer surface are two perfect spheres. The *great* circle is defined as the cloud layer surface at height h , and *small* circle is the Earth's surface. The tangent plane to the Earth's surface which intersects with the cloud layer is the chord AB (see Figure 5.2). The Earth's radius is r_{Earth} . The sagitta $\ell_i = h - v_i$ is the length from the middle of chord C_iD_i to the cloud layer, and v_i is the perpendicular distance from the *great* circle to the *small* circle. The *great* and *small* circles radii are respectively,

$$\begin{aligned} R &= r + h \\ r &= r_{Earth} + \rho \end{aligned} \tag{5.8}$$

where ρ is the altitude above the sea-level of the localization where the sky imager is installed.

The imager elevation angle ε_i defines the triangle formed by the line z_i that intersect the Earth's surface and the cloud layer as:

$$\tan \varepsilon_i = \frac{v_i}{w_i}. \tag{5.9}$$

By taking this approach, the geospatial reprojection coordinates are calculated with respect to the imager lens.

5.4.1 Reprojection of the y-axis

The sagitta ℓ_i of chord C_iD_i is related to the chord AB (Figure 5.2). The formula that describes the sagitta ℓ_i is a function of the triangle formed by the intersecting

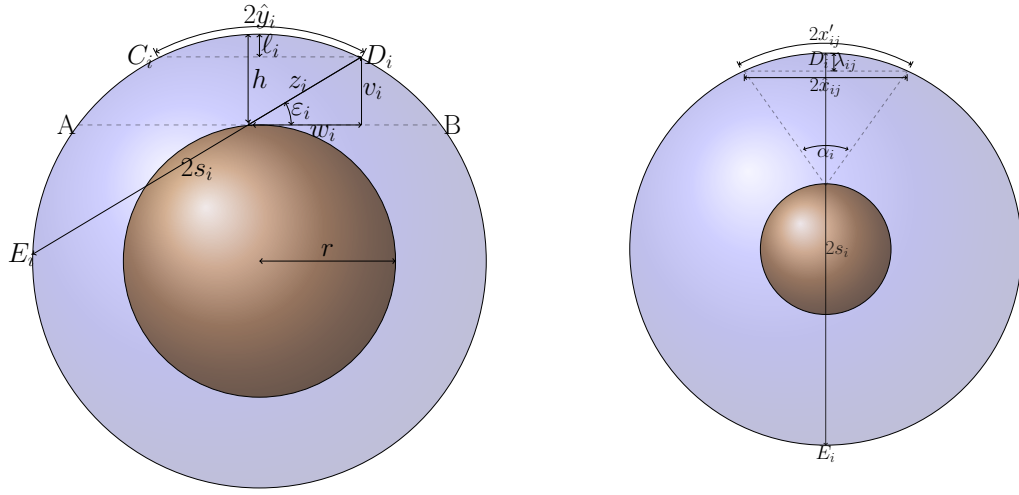


Figure 5.2: Drawing of the *great* circle (surface of a cloud layer) and the *small* circle (Earth's surface). The key in this approach is to find the relation between the chords $C_i D_i$ and AB to calculate y'_i (see right drawing, which is the imager's y-axis view). Similarly, $x'_{i,j}$ is computed for each y'_i , using the circle with diameter $2s_i$, formed by chord $D_i E_i$ (see left drawing, which is the imager's x-axis view).

line z_i that goes from AB to $C_i D_i$ with elevation angle ε_i ,

$$\begin{aligned}
 \ell_i &= R - \sqrt{R^2 - w_i^2} \\
 h - v_i &= R - \sqrt{R^2 - w_i^2} \\
 R^2 - w_i^2 &= (w_i \tan \varepsilon_i + r)^2 \\
 R^2 - w_i^2 &= w_i^2 \tan^2 \varepsilon_i + r^2 + 2rw_i \tan \varepsilon_i \\
 (r + h)^2 &= w_i^2 \tan^2 \varepsilon_i + 2rw_i \tan \varepsilon_i + w_i^2 + r^2 \\
 h^2 + 2rh &= w_i^2 (1 + \tan^2 \varepsilon_i) + 2rw_i \tan \varepsilon_i \\
 0 &= w_i^2 (1 + \tan^2 \varepsilon_i) + w_i (2r \tan \varepsilon_i) - h (h + 2r),
 \end{aligned} \tag{5.10}$$

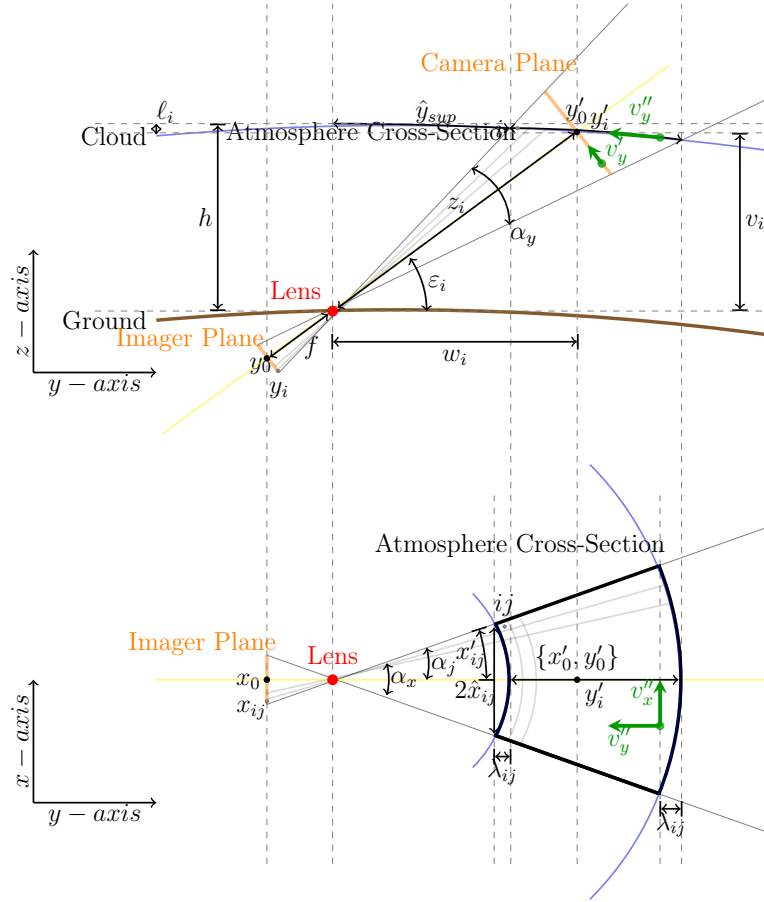


Figure 5.3: *great* circle geospatial reprojection in the y-axis or side view (top drawing) and the x-axis or top view (bottom drawing). *Imager Plane* and *Lens* are parts of the sky imager. They are physically separated by a focal length f . The distance z_i from the lens to the cloud layer is detailed in the top graph. x'_j and y'_i represent the coordinates of the pixel in the image (see in Eq. (5.20) and Eq. (5.14)).

where $\ell_i = h - v_i$, $v_i = w_i \tan \varepsilon_i$ and $R = r + h$. The quadratic equation has following coefficients,

$$\begin{aligned} a_i &= 1 + \tan^2 \varepsilon_i \\ b_i &= 2r \tan \varepsilon_i \\ c_i &= -h(h + 2r). \end{aligned} \tag{5.11}$$

The length of triangle side w_i is the result obtained solving the quadratic formula,

$$w_i = \frac{-b_i + \sqrt{b_i^2 - 4a_i c_i}}{2a_i}, \quad w_i \in \mathbb{R}^+. \quad (5.12)$$

When $r \rightarrow \infty$, $w_i \approx d$ and $v_i \approx h$, thus the *flat* approximation is equivalent to the *great* circle approach $w_i \approx h / \tan \varepsilon_i$.

The *great* circle segment \hat{y}_i is the distance from the center of the arc defined by the sagitta ℓ_i to the point D_i (Figure 5.2). The chord $2w_i$ is projected to the arc $2\hat{y}_i$ of the *great* circle by applying the arc formula:

$$\hat{y}_i = R \arcsin \frac{w_i}{R}. \quad (5.13)$$

Each pixel in an image has a different elevation angle ε_i that corresponds to a point D_i in the *great* circle. Therefore, the coordinates of the pixels relative to the imager lens in the atmosphere cross-section plane are calculated subtracting them the distance \hat{y}_{sup} which has the highest elevation (Figure 5.3, upper pane),

$$y'_i = \hat{y}_i - \hat{y}_{sup}, \quad \forall i = 1, \dots, N. \quad (5.14)$$

5.4.2 Reprojection of the x-axis

The reprojection of the sensor plane x-axis to the atmosphere cross-section is a function of the distance $z_i^2 = w_i^2 + v_i^2$ from the sensor plane to the cloud layer, and the chord $2\hat{x}_{i,j}$ of segment the $2x'_{i,j}$ formed by the angle α_j (Figure 5.3, lower pane),

$$\hat{x}_{i,j} = (z_i - \lambda_{i,j}) \tan \alpha_j, \quad \forall i = 1, \dots, M, \quad j = 1, \dots, N. \quad (5.15)$$

The diameter of the small circle $2s_i$, which is the chord $D_i E_i$ in Figure 5.2, is obtained by applying the intersecting chord theorem. In Euclid's Elements Book III, Proposition 35, (see [139]), the intersecting chords theorem is defined as $|AS| \cdot |SC| = |BS| \cdot |SD| =$

$r^2 - d^2$. When this theorem is applied to our problem the corresponding variables are $d = (R - h)$, $r = R$, $|AS| = s_i - z_i$ and $|SC| = z_i$ (see Figure 5.2 y-axis graph), so

$$\begin{aligned} (2s_i - z_i) z_i &= R^2 - (R - h)^2 \\ s_i &= \frac{2Rh - h^2}{2z_i} + \frac{z_i}{2}. \end{aligned} \quad (5.16)$$

The relation between the arc length $2x'_{i,j}$ and the chord $2\hat{x}_{i,j}$ is found through the sagitta $\lambda_{i,j}$. The formula which describes

$$\begin{aligned} (2s_i - \lambda_{i,j}) \lambda_{i,j} &= \hat{x}_{i,j}^2 \\ 2s_i &= \lambda_{i,j} + \frac{\hat{x}_{i,j}^2}{\lambda_{i,j}} \\ 2s_i \lambda_{i,j} - \lambda_{i,j}^2 &= (z_i - \lambda_{i,j})^2 \tan^2 \alpha_j \\ 0 &= \lambda_{i,j}^2 (1 + \tan^2 \alpha_j) - 2\lambda_{i,j} (z_i \tan^2 \alpha_j - s_i) + z_i^2 \tan^2 \alpha_j, \end{aligned} \quad (5.17)$$

where coefficients for solving the quadratic formula are,

$$\begin{aligned} a_{i,j} &= 1 + \tan^2 \alpha_j \\ b_{i,j} &= -2s_i - 2z_i \tan^2 \alpha_j \\ c_{i,j} &= z_i^2 \tan^2 \alpha_j. \end{aligned} \quad (5.18)$$

The sagitta $\lambda_{i,j}$ is the result obtained solving Eq. (5.17),

$$\lambda_{i,j} = \frac{-b_{i,j} - \sqrt{b_{i,j}^2 - 4a_{i,j}c_{i,j}}}{2a_{i,j}}, \quad \lambda_{i,j} \in \mathbb{R}^+. \quad (5.19)$$

When $r \rightarrow \infty$, $s_i \rightarrow \infty$, in consequence $\lambda_{i,j} \approx 0$ and $x'_{i,j} \approx \hat{x}_{i,j}$. The *flat* Earth approximation is equivalent to the *great* circle approach.

The arc length $2x'_{i,j}$ is calculated knowing the sagitta $\lambda_{i,j}$ and the small radius s_i ,

$$x'_{i,j} = s_i \arcsin \left[\frac{(z_i - \lambda_{i,j}) \tan \alpha_j}{s_i} \right] \quad (5.20)$$

where segment $x'_{i,j}$ is the projection of x-axis in the atmosphere cross-selection plane.

The origin of the coordinate system can be defined at the position of the Sun,

$$\begin{aligned} x''_{i,j} &= x'_{i,j} - x'_0 \\ y''_{i,j} &= y'_{i,j} - y'_0, \end{aligned} \tag{5.21}$$

where $\mathbf{x}'_0 = \{y'_0, x'_0\}$ are the pixel index of the Sun position in the image. These equation is applicable to both proposed perspective reprojections.

The geospatial perspective reprojections are applied to a sky imager mounted on a solar tracker that updates its pan and tilt every second, maintaining the Sun in a central position in the images throughout the day. The sky imager is located in the ECE department at the UNM central campus in Albuquerque. The climate of Albuquerque is arid semi-continental, with minimal rain, which is more likely in the summer months. The ECE department is approximately located 1,250m away (i.e., linear distance) from the city center whose elevation is 1,620m with respect to sea level.

The IR sensor is a Lepton¹ 2.5 radiometric camera with wavelength from 8 to 14 μ m. Pixel intensity within the frame is measured in centikelvin units. The resolution of an IR image is 60 \times 80 pixels. To implement the reprojection, the manufacturing specifications of the camera used are: 63.75° diagonal FOV, 51° horizontal α_x , 38.25° vertical α_y , and the size of a pixel is $\delta = 17\mu$ m. When other lenses (e.g. fisheye) are used, the camera lens affine reprojection must first be computed to know the FOV of each pixel.

The pixels in Figure 5.4 and 5.5 are displayed in the camera pixel coordinates (left) and in the atmosphere cross-section plane (right). The pixels are scaled to their actual size in the atmosphere cross-section plane. The distortion produced by the sky imager perspective causes the atmosphere cross-section plane dimensions to increase when the elevation angle decreases (see Figure 5.6).

¹<https://www.flir.com/>

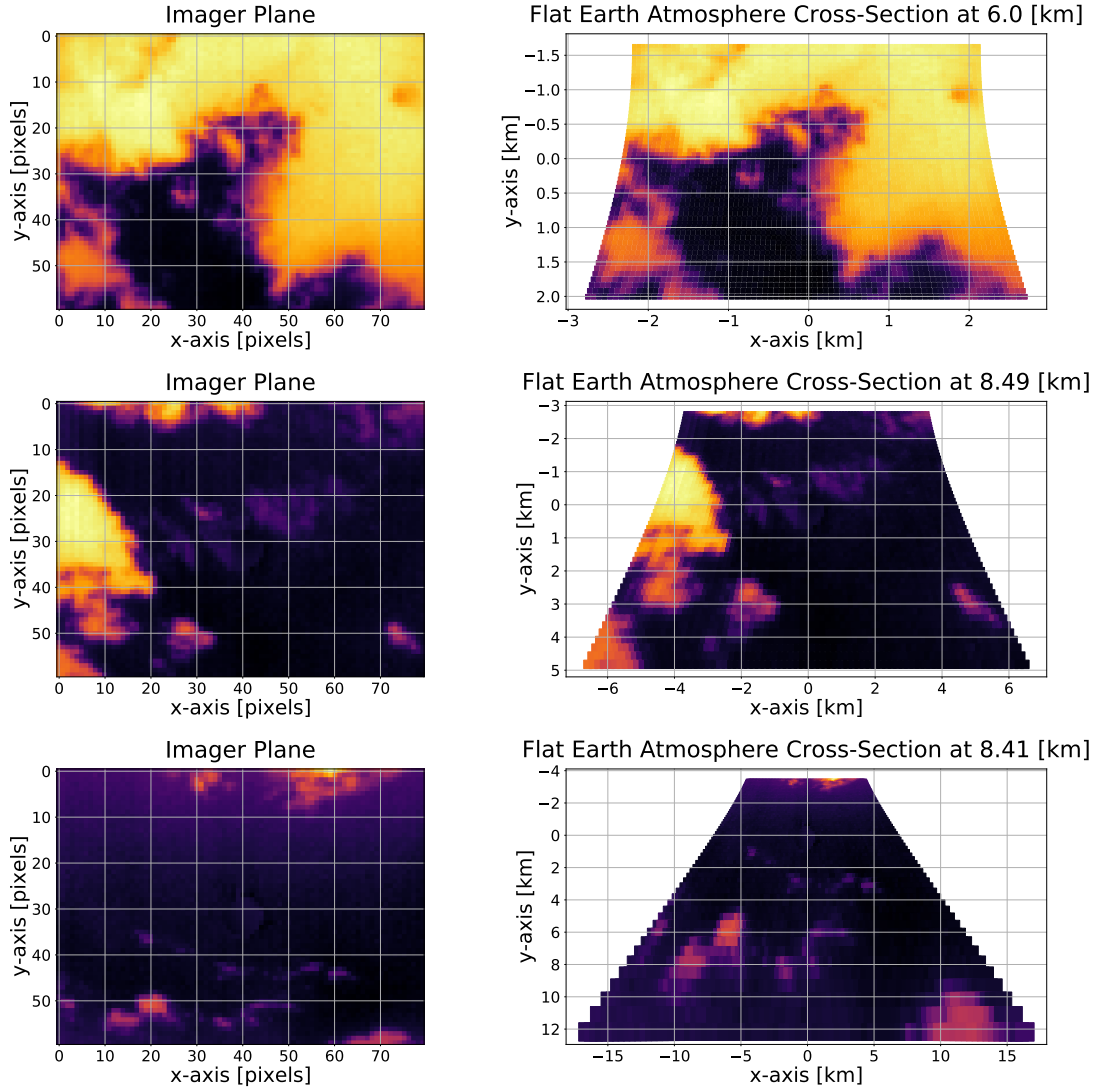


Figure 5.4: The left column shows three sky images taken at different elevation angles: 71.06° , 50.17° and 30.83° (from top to bottom). The right column shows the same three images after applying the geospatial perspective reprojection using the *flat* Earth approximation.

The difference between both geospatial reprojections is measured using RMSE. The coordinates computed using the *flat* Earth assumption are $\mathbf{X}_1, \mathbf{Y}_1$, and the coordinates computed using *great circle* approach are $\mathbf{X}_2, \mathbf{Y}_2$. The RMSE, defined as $E(\cdot)$, is calculated for each pixel averaging together the difference residuals computed

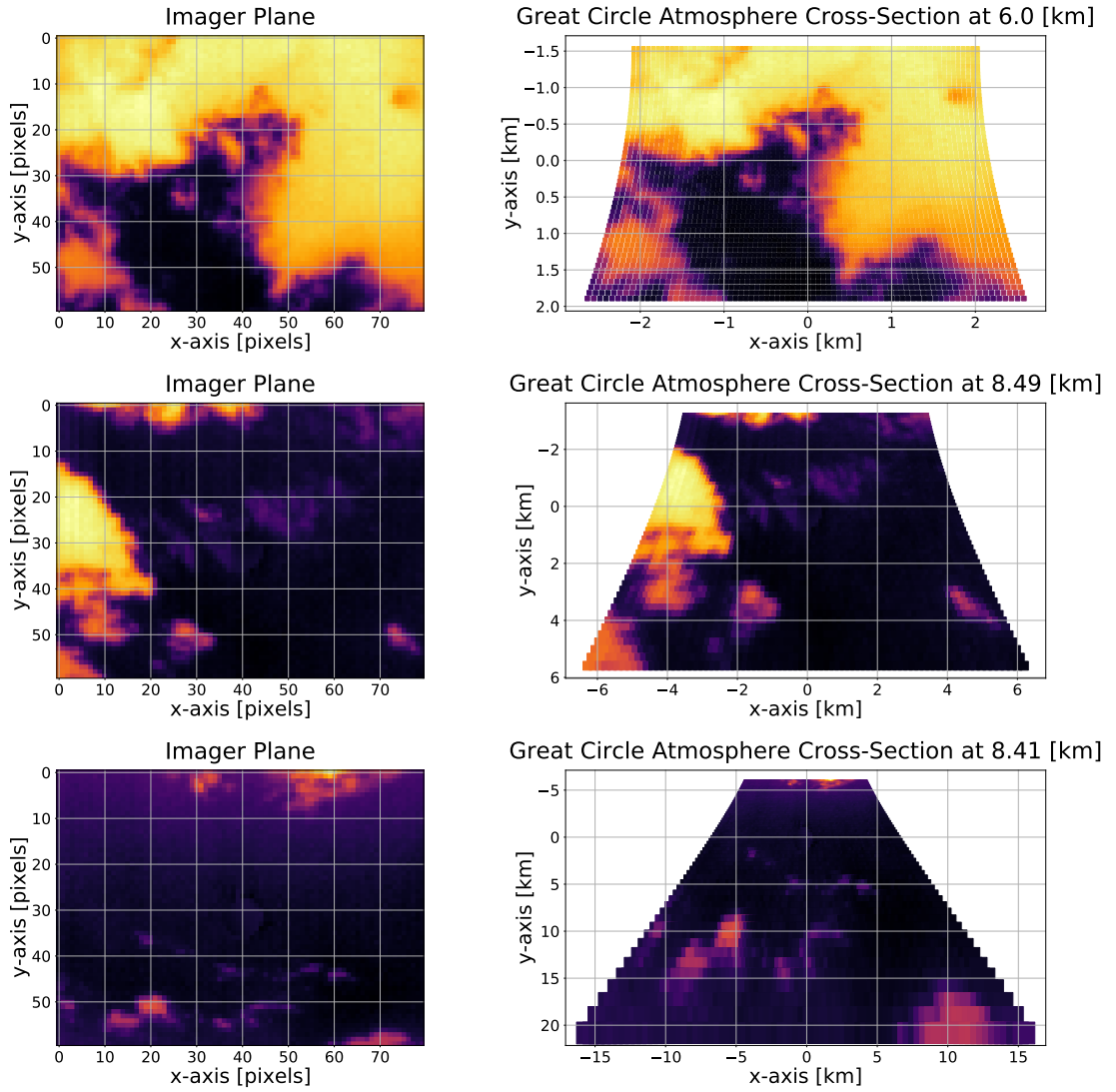


Figure 5.5: The left column shows three sky images taken at different elevation angles: 71.06° , 50.17° and 30.83° (from top to bottom). The right column shows the resulting sky images after applying the geospatial perspective reprojection using the *great circle* approach.

independently in coordinates x and y ,

$$E(\mathbf{X}_1, \mathbf{X}_2, \mathbf{Y}_1, \mathbf{Y}_2) = \sqrt{\frac{1}{2} [\mathcal{R}(\mathbf{X}_1, \mathbf{X}_2) + \mathcal{R}(\mathbf{Y}_1, \mathbf{Y}_2)]}. \quad (5.22)$$

The residuals are $\mathcal{R}(\mathbf{X}_1, \mathbf{X}_2) = (\mathbf{X}'_1 - \mathbf{X}'_2)^2$ for each coordinate.

The error maps (see Figure 5.7) show the differences between the coordinate systems approximated by both reprojections. The symmetry between both reprojections is not perfectly circular. This is because the elevation angle in *flat* Earth reprojection, was approximated as constant across the pixels in the same row.

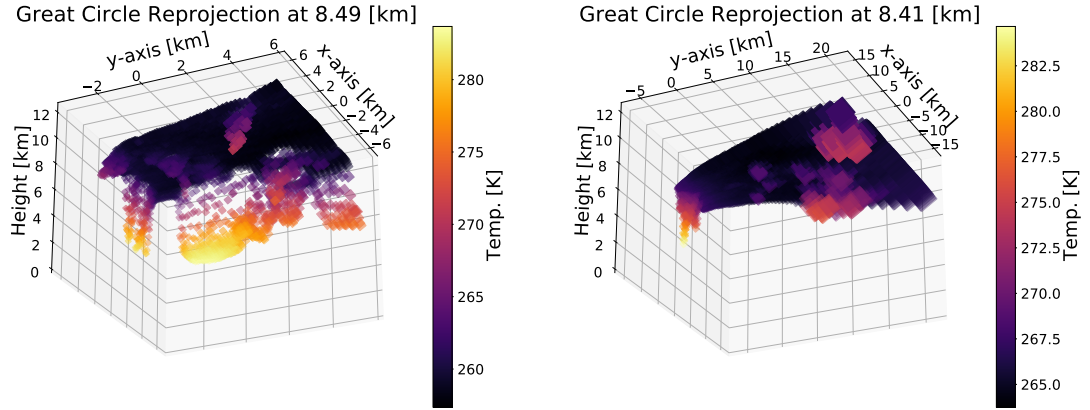


Figure 5.6: The graphs show the resulting sky images after applying the *great* circle geospatial perspective reprojection. The left graph shows the sky image taken at an elevation angle of 50.17° , and the right graph shows the sky image taken at an elevation angle of 30.83° .

The tropopause average height is approximately 10km in the latitude where the sky imager is located depending on the season. The first image in Figure 5.7 shows the error map when the camera is at the zenith. The magnitude order of the error is in meters when $\varepsilon \geq 30^\circ$. However, when the Sun's elevation angle is below $\varepsilon < 30^\circ$ the magnitude order of the error is in kilometers. Taking this into account, the geospatial reprojection that assumes that the Earth's surface is *flat*, is only adequate when the elevation angle of a sky imager pixel is above $\varepsilon \geq 30^\circ$. If the sky imager is designed to operate below $\varepsilon < 30^\circ$, the most suitable reprojection is computed using the great circle approach.

The difference between both transformations in the magnitude of the error is due

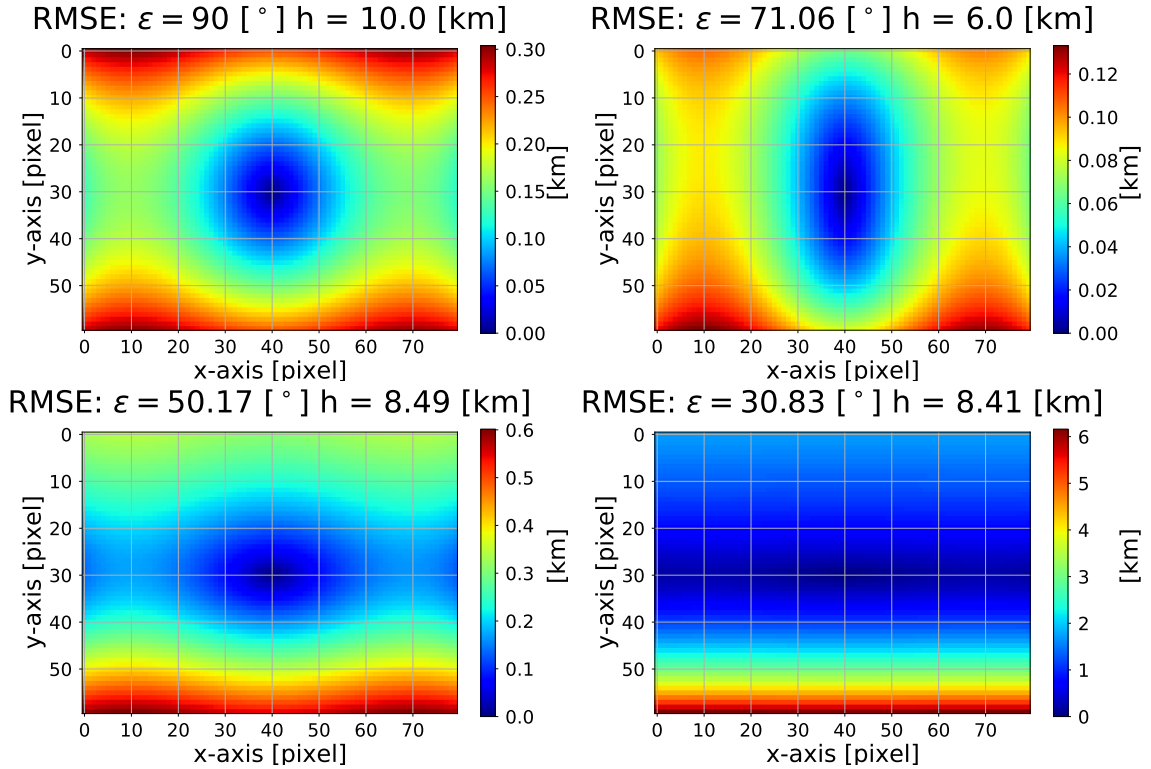


Figure 5.7: RMSE between the atmosphere cross-section coordinates approximated using the *flat* Earth assumption reprojection and the *great* circle approach reprojection. The coordinates of each reprojection are displayed in Figure 5.4 and 5.5 respectively.

to the dimensions of the region of the atmosphere that is being measured with the sky imager (see Figure 5.8). For an image at height h and elevation ε , we compute the total RMSE $E(h, \varepsilon)$ as the square root of sum of errors $E(\mathbf{X}_1, \mathbf{X}_2, \mathbf{Y}_1, \mathbf{Y}_2)$ for all coordinates of the image. Figure 5.8, left pane shows a representation of this error for various elevations as a function of h , and the right pane shows a heatmap of the total RMSE as a function of h and ε . As the elevation angle decreases, the region of the atmosphere that is measured in each pixel increases (i.e., perspective). Consequently, the great circle approach performs a more accurate approximation of the cross-section plane of the atmosphere in which the clouds are moving.

The atmosphere cross-section projected on the Earth's surface using the *great*

circle approach reprojection is shown in Figure 5.9 in Geographic Coordinates System (GCS). The GCS components are longitude and latitude and they are defined in degrees. The atmosphere cross-section plane is considerably larger when the Sun's elevation angle is low. The distance between pixels in an image increases exponentially from top to bottom.

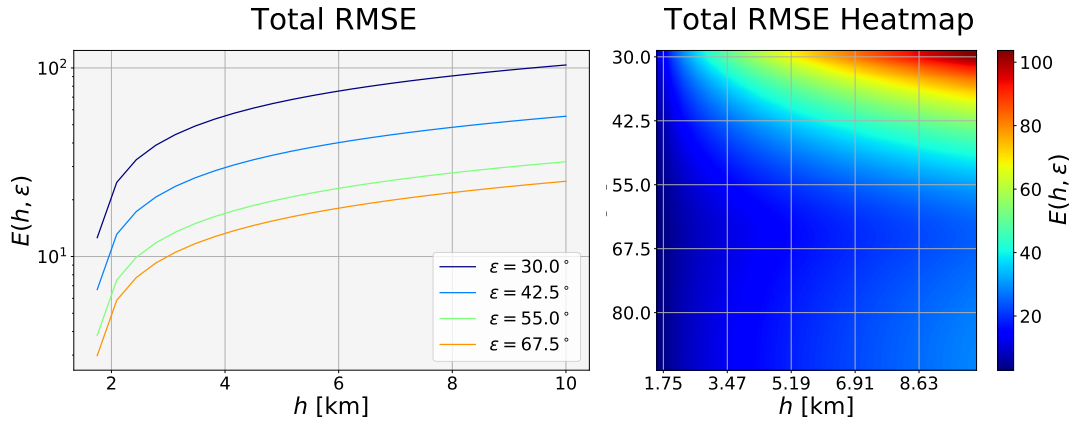


Figure 5.8: The left graph shows the increase in the quadratic total sum of error as a function of the height for 5 different elevation angles: 30° , 42.5° , 55° and 67.5° . The error function is in Eq. (5.22). The right graph shows the quadratic total sum of errors as a function of the elevation angle and the height.

The results presented in this chapter show that the proposed methodology is advantageous with respect to other methods available in the literature from a theoretical and technological point of view. The geometric reprojection proposed in [247] (i.e., voxel carving) is equivalent to the *flat* Earth approximation investigated in this chapter, and thus it does not consider the curvature of the Earth (i.e., *great* circle approach). As it is demonstrated in this chapter (see Figure 5.7), the order of magnitude of the error produced by this approximation is in the range of kilometers for high clouds (e.g., stratus) measured by pixels with elevation angles $< 30^\circ$. In addition, low-cost radiometric far IR cameras provide temperature measurements (e.g., see [193]) which can be transformed to height measurements [311] when combined with weather features measured by a simple weather station in the ground (see Chapter 3). Radiometric

IR cameras have low resolution (see Chapter 2), but their resolution is sufficient to perform accurate intra-hour solar forecasting (see Chapters 8 and 9).

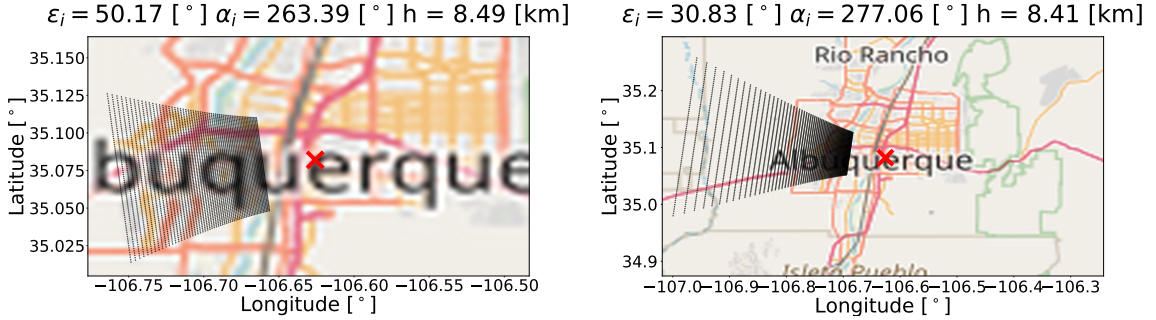


Figure 5.9: Atmosphere cross-section plane projected on the Earth's surface for elevations of 50.17° and 30.83° using the *great circle* approach (see Figure 5.6). The sky imager localization is the red dot. The sky imager pixels are in black. The coordinates of a pixel are defined by a longitude and latitude angle.

5.5 Conclusion

Intra-hour solar forecasting algorithms utilize consecutive sky images to compute cloud velocity vectors, anticipating when a cloud will occlude the Sun and produce a decrease in the GSI that reaches the Earth's surface. Velocity vectors are calculated in units of pixels per frame, but the dimensions of the pixels in sky images vary with the elevation angle. Therefore, the velocity vector accuracy used to forecast cloud occlusions of the Sun can be improved. The proposed perspective reprojection of the sensor plane to the geospatial atmosphere cross-section plane can be used to transform the pixels in sky images to the cross section coordinate system of the clouds.

When used in sky imagers, thermal images are advantageous in that cloud height can be approximated when cloud temperature is known. Radiometric IR cameras composed of microbolometers are an inexpensive technology capable of acquiring thermal sky images. When intersected by the sky imaging system field of view, the

Chapter 5. Geospatial Perspective Reprojections for Sky Imaging Systems

dimensions of the atmosphere cross-section plane can be determined using the proposed reprojections and temperatures of the objects in the images.

Chapter 6

Detection of Clouds in Multiple Wind Velocity Fields

6.1 Introduction

The formation of clouds is a phenomenon restricted by the Tropopause [171]. Different types of clouds are expected to form at different altitudes within the Troposphere [186]. The magnitude of the wind velocity field increases with the altitude in the lower atmosphere [358, 359]. The wind gradient may also change its direction due to the friction of the wind with the surface of the Earth. The planetary boundary layer (the lowest part of the Troposphere) [49] is the point where the wind shear causes low level clouds to move in a different direction and speed of that from high level clouds [36, 37].

The wind velocity field shown in a sequence of cloud images is a physical process that is assumed to have a limited complexity [277]. A sequence of IR images allows the derivation of physical features from moving clouds in a wind velocity field. These are more interpretable for modelling physical processes. The features are temperature,

velocity vectors, and height [91]. It has been found that the advantage of using unsupervised learning algorithms is that the response to a sequence of images is expected to depend on the physical process that the images represent rather than the intensity of their pixels [142]. Unsupervised learning methods, in special mixture models, infer the probability density functions of the observations without prior information [136].

Hidden Markov Models (HMM) were introduced to model linear sequences of discrete latent variables or states as a Markov process. These models are popular in computer vision and pattern recognition applications in which the current state of a system in an image is modelled in terms of previous states [41]. HMM have been used to detect and analyze temperature distributions of images acquired from IR thermal imaging systems [29]. It is possible to apply HMM to model stochastic physical processes [25, 156] in image classification [196], and object recognition [23].

The unsupervised learning algorithm proposed in this chapter does inference over the number of wind velocity fields in a sequence of images using features extracted from the clouds. IR images of clouds are obtained using an innovative DAQ system mounted in a solar tracker (see Chapter 2). The velocity vectors are computed using a weighted variation of the standard LK method [209]. The velocity vector of each pixel is computed in a weighted window of neighboring pixels. The weight of a pixel is the posterior probability of belonging to the lower or the upper cloud layer. The obtained velocity vectors for each cloud layer are averaged together weighting them by the posterior probability of each cloud layer.

A real-time probabilistic model is implemented to detect the number of layers in an IR image [308]. The proposed model is an HMM that models the hidden process of the number of wind velocity fields in a sequence of images [52, 352]. The motion of the clouds on a sequence of images is used to calculate the velocity vectors. The temperature and height of the pixels are also extracted from the cloud images. The

distributions of the features are inferred with different parametric mixture models [225, 226]. The mixture models are optimized using the EM algorithm [31, 239]. The label switching of the mixture models is solved using the average height of each distribution in the mixture model [310].

6.2 Methodology

When there are multiple layers of clouds in an image, a mixture model is expected to have multiple clusters. In order to infer the distribution of the temperatures or heights using a Beta Mixture Model (BeMM), the features are first normalized to the domain of a beta distribution $\bar{T}_{i,j,k}'' = [T_{i,j,k}'' - \min(\mathbf{T}_k'')]/[\max(\mathbf{T}_k'') - \min(\mathbf{T}_k'')]$. When the inference is performed with a Gamma Mixture Model (GaMM), the temperatures are normalized to the domain of the gamma distribution $\tilde{T}_{i,j,k}'' = T_{i,j,k}'' - \min(\mathbf{T}_k'')$. The heights (in kilometers) are within the domain of the gamma distribution. When the inference is performed using a GMM, the temperatures and heights do not require normalization.

6.2.1 Weighted Lucas-Kanade

In current computer vision literature, there are three primary methods to estimate the motion of objects in a sequence of images: LK [209], Horn-Schunck (HS) [150] and Farnebäck [96] methods. These three methods are based on the space-time partial derivatives between two consecutive frames. The techniques to estimate the motion vectors in an image are sensitive to the intensity gradient of the pixels. An atmospheric model is implemented to remove the gradient produced by the Sun's direct irradiance and the atmospheric scattered irradiance (both of which routinely appear on the images in the course of the year). A persistent model of the outdoor germanium

window of the IR camera removes debris and water spots that appear in the image (see Chapter 3). In this chapter, it is implemented a Weighted Lucas-Kanade (WLK) method.

Lucas-Kanade Method

The LK method proposes to find the solution for the optical flow (see Appendix A.1.1) equations via Least Squares (LS). In this method, the optical flow equation is solved using a local image of the pixels within a sliding window. In this chapter, the LK method is extended for multiple importance weights. A sliding window is defined with odd width $W = 2w + 1$, where w is the window size parameter, which has to be cross-validated.

Now we assume that the image may contain more than one velocity field. Then, at pixel i, j of layer ℓ , we define $1 \leq \ell \leq L$ hypothesis over the possible velocities $\mathbf{v}_{i,j,\ell}$.

If the position of the central pixel of the window is defined as i, j , then the dependent and independent variables can be defined as

$$\begin{aligned} \mathbf{x}_{i-m,j-n} &= \begin{bmatrix} \mathbf{I}'_x(i-m, j-n) \\ \mathbf{I}'_y(i-m, j-n) \end{bmatrix}, \quad \mathbf{v}_{i,j,\ell} = \begin{bmatrix} u_{i,j,\ell} \\ v_{i,j,\ell} \end{bmatrix}, \\ y_{i-m,j-n} &= -\mathbf{I}'_k(i-m, j-n), \quad -w \leq m \leq w, \quad -w \leq n \leq w. \end{aligned} \quad (6.1)$$

Each velocity vector is associated to a posterior probability $\gamma_{i,j,\ell} \triangleq p(z_{i,j} = \ell \mid T''_{i,j})$ where $z_{i,j} = \ell$ is a latent variable that indicates that the velocity field at pixel i, j is corresponds to cloud layer ℓ . These posteriors will be estimated in the next section.

We introduce a Weighted Least Squares (WLS) approach [18], whose weights are posterior probabilities $\gamma_{i,j,\ell}$. Assume an extended vector $\mathbf{y}_{i,j}$ containing all instances of $\mathbf{y}_{i-m,j-n}$ and a matrix $\mathbf{X}_{i,j}$ containing all $\mathbf{x}_{i-m,j-n}$.

Instead of minimizing the mean square error, we can maximize the expectation of

the unnormalized log-posterior

$$\begin{aligned}
& \mathbb{E} \{ \log [p(\mathbf{y}_{i,j} | \mathbf{X}_{i,j}, \mathbf{v}_{i,j,\ell}) p(\mathbf{v}_{i,j,\ell})] \} = \\
& = \mathbb{E} \left\{ \log \left[\prod_{m,n,\ell} [p(\mathbf{y}_{i-m,j-n} | \mathbf{x}_{i-m,j-n}, \mathbf{v}_{i,j,\ell})]^{\mathbb{I}(z_{i,j}=\ell)} p(\mathbf{v}_{i,j,\ell}) \right] \right\} \\
& = \sum_{m,n,\ell} \mathbb{E} [\mathbb{I}(z_{i-m,j-n} = \ell)] \log p(\mathbf{y}_{i-m,j-n} | \mathbf{x}_{i-m,j-n}, \mathbf{v}_{i,j,\ell}) + \log p(\mathbf{v}_{i,j,\ell}) \quad (6.2) \\
& = \sum_{m,n,\ell} \gamma_{i-m,j-n,\ell} \log p(\mathbf{y}_{i-m,j-n,\ell} | \mathbf{x}_{i-m,j-n,\ell}, \mathbf{v}_{i,j,\ell}) + \log p(\mathbf{v}_{i,j,\ell}) \\
& = - \sum_{m,n,\ell} \gamma_{i-m,j-n,\ell} \|\mathbf{v}_{i,j,\ell}^\top \mathbf{x}_{i-m,j-n} - \mathbf{y}_{i-m,j-n}\|^2 - \tau \|\mathbf{v}_{i,j,\ell}\|^2 + \text{constant},
\end{aligned}$$

where $\gamma_{i-m,j-n,\ell} = \mathbb{E}[\mathbb{I}(z_{i-m,j-n} = \ell)]$ is the posterior probability of the velocity field, $\mathbb{I}(\cdot)$ is the indicator function, and where we assumed that both probabilities are Gaussian distributions, where the first one is a multivariate Gaussian modelling the error, which has a given variance σ_n^2 , and the second one is a multivariate Gaussian modelling the prior over the velocities, whose covariance is an identity $\mathbf{I}_{2 \times 2}$. Thus, $\tau = \sigma_n^2$ plays the role of a regularization parameter, and it may be validated or inferred by maximizing the likelihood term [314].

By computing the gradient of the expression with respect to $\mathbf{v}_{i,j,\ell}$ and nulling it, we obtain the solution,

$$\hat{\mathbf{v}}_{i,j,\ell} = (\mathbf{X}_{i,j} \mathbf{\Gamma}_{i,j,\ell} \mathbf{X}_{i,j}^\top + \tau \mathbf{I}_{2L \times 2L})^{-1} \mathbf{X}_{i,j} \mathbf{\Gamma}_{i,j,\ell} \mathbf{y}_{i,j} \quad (6.3)$$

where $\mathbf{\Gamma}_{i,j,\ell}$ is a diagonal matrix containing all posteriors of the l cluster.

The estimated velocity components are defined as $\hat{\mathbf{U}}_\ell = \{\hat{\mathbf{u}}_{i,j,\ell} \in \mathbb{R} \mid i = 1, \dots, M, j = 1, \dots, N\}$ and $\hat{\mathbf{V}}_\ell = \{\hat{\mathbf{v}}_{i,j,\ell} \in \mathbb{R} \mid i = 1, \dots, M, j = 1, \dots, N\}$.

The obtained velocity components for each posterior l , are averaged weighting the vectors components by their posterior,

$$\hat{\mathbf{U}} = \sum_{\ell=1}^L (\mathbf{\Gamma}_\ell \odot \hat{\mathbf{U}}_\ell); \quad \hat{\mathbf{V}} = \sum_{\ell=1}^L (\mathbf{\Gamma}_\ell \odot \hat{\mathbf{V}}_\ell), \quad (6.4)$$

where \odot is the Hadamard product. The result are the velocity components for each pair of coordinates in the original frame $\hat{\mathbf{U}}, \hat{\mathbf{V}} \in \mathbb{R}^{M \times N}$. The velocity vectors defined in polar coordinates have magnitude $\mathbf{M} = (\hat{\mathbf{U}} \odot \hat{\mathbf{U}} + \hat{\mathbf{V}} \odot \hat{\mathbf{V}})^{1/2}$ and angle $\mathbf{\Omega} = \arctan2(\hat{\mathbf{U}}, \hat{\mathbf{V}})$.

6.2.2 Maximum a Posteriori Mixture Model

When the clouds are moving in multiple wind velocity fields the distributions of the temperatures, heights and velocity vectors components are expected to be distributed in multiple clusters in the feature space of the observation.

Mixture models are implemented to infer the distributions of the physical features extracted from IR cloud images. These physical features have different domain so the inference is implemented using probability functions defined in each one of these domains. Thus, the distribution of the velocity vectors defined in Cartesian coordinates is inferred with a multivariate GMM. The inference of the velocity vectors when they are defined in polar coordinates, is performed independently for each component using a GaMM and Von Mises Mixture Model (VMMM) for the magnitude and the angle respectively.

In this section, we propose the use of different mixture models to infer probability functions that approximate better the actual distribution of a features with the aim of detecting the most likely number of wind velocity fields on an image and their pixelwise posterior probabilities $\gamma_{i,j,\ell}$. The formulation of the proposed mixture models includes a prior distribution on the cluster weights in order to avoid overfitting.

Expectation-Maximization

Let us consider that \mathbf{x}_i are observations (i.e. feature vectors) which we wish to model as a mixture model, and that z_i are the corresponding latent variables of their cluster index. The optimal set of parameters in a mixture model can be computed using the EM algorithm. The implementation of the EM algorithm guarantees a smooth convergence to a local maximum following an iterative approach consisting of two steps [31]. The maximized function is the expected Complete Data Log-Likelihood (CDLL) plus the log-prior,

$$\begin{aligned}
 \mathcal{Q}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t-1)}) &\triangleq \mathbb{E} \left[\sum_{i=1}^N \log p(x_i, z_i | \boldsymbol{\theta}^{(t)}) + \log p(\boldsymbol{\pi} | \boldsymbol{\alpha}) \right] \\
 &= \sum_{i=1}^N \mathbb{E} \left[\log \left(\prod_{\ell=1}^L \left[\pi_{\ell} p(x_i | \boldsymbol{\theta}_{\ell}^{(t)}) \right]^{\mathbb{I}(z_i=\ell)} \right) \right] + \log p(\boldsymbol{\pi} | \boldsymbol{\alpha}) \\
 &= \sum_{i=1}^N \sum_{\ell=1}^L p(z_i = \ell | x_i, \boldsymbol{\theta}_{\ell}^{(t-1)}) \log \left[\pi_{\ell} p(x_i | \boldsymbol{\theta}_{\ell}^{(t)}) \right] + \log p(\boldsymbol{\pi} | \boldsymbol{\alpha}) \\
 &= \sum_{i=1}^N \sum_{\ell=1}^L \gamma_{i,\ell} \log \left[\pi_{\ell} p(x_i | \boldsymbol{\theta}_{\ell}^{(t)}) \right] + \log p(\boldsymbol{\pi} | \boldsymbol{\alpha}),
 \end{aligned} \tag{6.5}$$

where t represents the iteration of the algorithm, and the posterior probability introduced in Eq. (6.2) appears here as

$$\gamma_{i,\ell} \triangleq p(z_i = \ell | x_i, \boldsymbol{\theta}_{\ell}^{(t-1)}, \boldsymbol{\alpha}), \tag{6.6}$$

which is commonly referred to as the responsibility of cluster ℓ in sample i . A prior $p(\boldsymbol{\pi} | \boldsymbol{\alpha})$ with parameters $\boldsymbol{\alpha}$ is introduced for probabilities $\boldsymbol{\pi}$. The initialization of the EM starts by randomly assigning a set of parameters and a prior. In the expectation step of the EM algorithm, a posterior $\gamma_{i,\ell}$ is assigned to each sample using the likelihood function,

$$\gamma_{i,\ell} = \frac{\pi_{\ell} p(x_i | \boldsymbol{\theta}_{\ell}^{(t-1)})}{\sum_{\ell=1}^L \pi_{\ell} p(x_i | \boldsymbol{\theta}_{\ell}^{(t-1)})}. \tag{6.7}$$

In the maximization step, the parameters that maximize the CDLL plus the log-prior are found analytically, as it will be shown further. The prior $p(\boldsymbol{\pi}|\boldsymbol{\alpha})$ is a Dirichlet distribution $\boldsymbol{\pi} \sim \text{Dir}(\boldsymbol{\alpha})$, and $\alpha_\ell \geq 1$. The mixture weights are updated using the posterior probabilities [239],

$$\pi_\ell = \frac{\alpha_\ell - 1 + \sum_{i=1}^N \gamma_{i,\ell}}{N - L + \sum_{\ell=1}^L \alpha_\ell}, \quad (6.8)$$

when $\alpha_\ell = 1$, the prior is noninformative $p(\boldsymbol{\pi}|\boldsymbol{\alpha}) = 0$ and thus the MAP is equivalent to the MLE, $\pi_\ell = [\sum_{i=1}^N \gamma_{i,\ell}]/N$.

The E and M steps are repeated until the CDLL have converged to a local maxima. In the case of a quadratic loss function this problem has analytical solution, for instance in a GMM [239]. When the loss function has not analytical solution, it can be solved implementing a numerical optimization method based on the gradient descent.

Gamma Mixture Model

The distribution of the magnitude of velocity vectors or the heights can be approximate by mixture of Gamma distributions $X \sim \mathcal{G}(\alpha_\ell, \beta_\ell)$ which density function is,

$$f(x_i; \alpha_\ell, \beta_\ell) = \frac{x_i^{\alpha_\ell-1} e^{-\frac{x_i}{\beta_\ell}}}{\beta_\ell^{\alpha_\ell} \Gamma(\alpha_\ell)} \quad x_i > 0, \quad \alpha_\ell, \beta_\ell > 0, \quad (6.9)$$

where $\Gamma(\alpha_\ell)$ is the Gamma function.

The log-likelihood of the Gamma density function needed to compute the expected CDLL in a GaMM is,

$$\log p(x_i|\alpha_\ell, \beta_\ell) = (\alpha_\ell - 1) \log x_i - \frac{x_i}{\beta_\ell} - \alpha_\ell \log \beta_\ell - \log \Gamma(\alpha_\ell). \quad (6.10)$$

The maximization step has to be solved via numerical optimization. The gradient

w.r.t. α_ℓ is,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial \alpha_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \frac{\partial}{\partial \alpha_\ell} \log p(x_i | \alpha_\ell, \beta_\ell) \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\log x_i - \log \beta_\ell - \frac{\Gamma'(\alpha_\ell)}{\Gamma(\alpha_\ell)} \right], \end{aligned} \quad (6.11)$$

where $\Gamma'(\alpha_\ell)$ is the derivative of the Gamma function. The gradient w.r.t. β_ℓ is,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial \beta_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\partial}{\partial \beta_\ell} \log p(x_i | \alpha_\ell, \beta_\ell) \right] \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{1}{\beta_\ell} \left(\frac{x_i}{\beta_\ell} - \alpha_\ell \right) \right]. \end{aligned} \quad (6.12)$$

The generalizations of the Gamma distribution for multiple dimensions do not have an unified expression. In fact, the multivariate Gamma distribution is unknown in the exponential family [291].

Bivariate Gamma Mixture Model

The distribution of magnitude of velocity vectors and heights can be approximated by mixture of bivariate Gamma distributions $X, Y \sim \mathcal{BG}(\alpha_\ell, \beta_\ell, a_\ell)$ which density function is [291],

$$f(x_i, y_i; \alpha_\ell, \beta_\ell, a_\ell) = \frac{\beta_\ell^{\alpha_\ell} x_i^{\alpha_\ell + a_\ell - 1} y_i^{\alpha_\ell - 1}}{\Gamma(a_\ell) \Gamma(\alpha_\ell)} e^{-\beta_\ell x_i} e^{-x_i y_i} \quad x_i, y_i > 0, \quad (6.13)$$

where $\Gamma(\alpha_\ell)$ is the Gamma function, and the parameters are $\alpha_\ell, \beta_\ell, a_\ell > 0$.

The log-likelihood of the bivariate Gamma density function needed for computing the expected CDLL in a Bivariate Gamma Mixture Model (BGaMM) is,

$$\begin{aligned} \log p(x_i, y_i | \alpha_\ell, \beta_\ell, a_\ell) &= \alpha_\ell \log \beta_\ell + (\alpha_\ell + a_\ell - 1) \log x_i + \dots \\ &\dots + (a_\ell - 1) \log y_i - \beta_\ell x_i - x_i y_i - \log \Gamma(\alpha_\ell) - \log \Gamma(a_\ell). \end{aligned} \quad (6.14)$$

As the maximization of Eq. (6.5) has not analytical solution when the likelihood is a bivariate Gamma, the maximization step is solved by numerical optimization. The gradient w.r.t. α_ℓ is,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial \alpha_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \frac{\partial}{\partial \alpha_\ell} \log p(x_i, y_i | \alpha_\ell, \beta_\ell, a_\ell) \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\log \beta_\ell + \log x_i - \frac{\Gamma'(\alpha_\ell)}{\Gamma(\alpha_\ell)} \right]. \end{aligned} \quad (6.15)$$

The gradient w.r.t. β_ℓ is,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial \beta_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\partial}{\partial \beta_\ell} \log p(x_i | \alpha_\ell, \beta_\ell, a_\ell) \right] \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\alpha_\ell}{\beta_\ell} - x_i \right]. \end{aligned} \quad (6.16)$$

The gradient w.r.t. a_ℓ is,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial a_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\partial}{\partial a_\ell} \log p(x_i, y_i | \alpha_\ell, \beta_\ell, a_\ell) \right] \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\log x_i + \log y_i - \frac{\Gamma'(a_\ell)}{\Gamma(a_\ell)} \right]. \end{aligned} \quad (6.17)$$

Applying the independence assumption to each component of the Gamma model, the general form of joint density for a multivariate Gamma distribution can be derived, but it needs to be assumed that marginal density functions for each one of random variables are available [291].

Von Mises Mixture Model

The angular component of the velocity vectors is approximated by a Von Mises distribution $X \sim \mathcal{VM}(\mu_\ell, \kappa_\ell)$. The density function of this distribution is,

$$f(x_i; \mu_\ell, \kappa_\ell) = \frac{e^{\kappa_\ell \cos(x_i - \mu_\ell)}}{2\pi I_0(\kappa_\ell)}, \quad x_i, \mu_\ell \in [-\pi, \pi], \quad \kappa_\ell > 0, \quad (6.18)$$

where I_0 represents the modified Bessel function of order 0 that has this formula,

$$I_\nu(\kappa_\ell) = \left(\frac{\kappa_\ell}{2}\right)^\nu \sum_{n=0}^{\infty} \frac{\left(\frac{1}{4}(\kappa_\ell)^2\right)^n}{n! \Gamma(\nu + n + 1)}. \quad (6.19)$$

In the case of mixture of a Von Mises distribution, the data log-likelihood for each cluster is,

$$\log p(x_i | \mu_\ell, \kappa_\ell) = \kappa_\ell \cos(x_i - \mu_\ell) - \log 2\pi - \log I_0(\kappa_\ell). \quad (6.20)$$

The maximization step is solved computing the gradient w.r.t. μ_ℓ ,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial \mu_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\partial}{\partial \mu_\ell} \log p(x_i | \mu_\ell, \kappa_\ell) \right] \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} [\kappa_\ell \sin(x_i - \mu_\ell)], \end{aligned} \quad (6.21)$$

and the gradient w.r.t. κ_ℓ ,

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\theta}_\ell)}{\partial \kappa_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\partial}{\partial \kappa_\ell} \log p(x_i | \mu_\ell, \kappa_\ell) \right] \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\cos(x_i - \mu_\ell) - \frac{I_1(\kappa_\ell)}{I_0(\kappa_\ell)} \right], \end{aligned} \quad (6.22)$$

where the Bessel function of order 1 is obtained from $\partial I_0(\kappa)/\partial \kappa = I_1(\kappa)$.

An extension for the multivariate Von Mises distribution can be found in [241], other solutions to the VMMM problem are [19, 119].

Beta Mixture Model

The distribution of the normalized temperatures or heights can be approximated by mixture of beta distributions $X \sim \mathcal{B}(\alpha_\ell, \beta_\ell)$ which density function is,

$$f(x_i; \alpha_\ell, \beta_\ell) = \frac{1}{B(\alpha_\ell, \beta_\ell)} x_i^{\alpha_\ell-1} (1 - x_i)^{\beta_\ell-1}, \quad \alpha_\ell, \beta_\ell > 0, \quad (6.23)$$

where $x_i \in (0, 1)$, beta function is $B(\alpha_\ell, \beta_\ell) = [\Gamma(\alpha_\ell)\Gamma(\beta_\ell)]/[\Gamma(\alpha_\ell + \beta_\ell)]$, and $\Gamma(\alpha_\ell)$ is the Gamma function.

The log-likelihood of the beta density function, that is needed to compute the expected CDLL in the mixture model is,

$$\log p(x_i|\alpha_\ell, \beta_\ell) = (\alpha_\ell - 1) \log x_i + (\beta_\ell - 1) \log (1 - x_i) - \log B(\alpha_\ell, \beta_\ell). \quad (6.24)$$

The maximization step has to be solved by gradient descent. The gradient w.r.t. α_ℓ is,

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\theta}_\ell)}{\partial \alpha_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \frac{\partial}{\partial \alpha_\ell} \log p(x_i|\alpha_\ell, \beta_\ell) \\ &= \sum_{i=1}^N \sum_{\ell=1}^L \gamma_{i,\ell} [\log x_i - \psi(\alpha_\ell) + \psi(\alpha_\ell + \beta_\ell)], \end{aligned} \quad (6.25)$$

where $\partial B(\alpha_\ell, \beta_\ell)/\partial \alpha_\ell = B(\alpha_\ell, \beta_\ell)[\psi(\alpha_\ell) - \psi(\alpha_\ell + \beta_\ell)]$, and $\psi(\cdot)$ is the digamma function, which is $\psi(\alpha_\ell) = \Gamma'(\alpha_\ell)/\Gamma(\alpha_\ell)$. The gradient w.r.t. β_ℓ is,

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\theta}_\ell)}{\partial \beta_\ell} &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \left[\frac{\partial}{\partial \beta_\ell} \log p(x_i|\alpha_\ell, \beta_\ell) \right] \\ &= \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} [\log(-x_i) - \psi(\beta_\ell) + \psi(\alpha_\ell + \beta_\ell)]. \end{aligned} \quad (6.26)$$

In previous work carried out in the implementation of a BeMM clustering, was found that is not optimal to assign the number of clusters in these models applying Bayesian Information Criterion (BIC) [166] (see Subsection 6.2.3). The authors proposed to implement Integrated Classification Likelihood (ICL) instead.

Gaussian Mixture Model

The distribution of the velocity components in a Cartesian coordinates system can be approximate by a mixture of multivariate normal distributions $X \sim \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$. The

multivariate normal likelihood is,

$$f(\mathbf{x}_i; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell) = \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}_\ell|}} \cdot \exp \left\{ -\frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_\ell) \right\}. \quad (6.27)$$

The log-likelihood of the multivariate density function [239] for computing the expected CDLL in the GMM is,

$$\log p(\mathbf{x}_i | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell) = -\frac{D}{2} \log 2\pi - \frac{1}{2} \log |\boldsymbol{\Sigma}_\ell| - \frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_\ell). \quad (6.28)$$

In the maximization stage, the mean and variance of each cluster that maximize the log-likelihood have an analytical solution that is,

$$\begin{aligned} \boldsymbol{\mu}_\ell &= \frac{\sum_{i=1}^N \gamma_{i,\ell} \mathbf{x}_i}{\gamma_\ell} \\ \boldsymbol{\Sigma}_\ell &= \frac{\sum_{i=1}^N \gamma_{i,\ell} \mathbf{x}_i \mathbf{x}_i^\top}{\gamma_\ell} - \boldsymbol{\mu}_\ell \boldsymbol{\mu}_\ell^\top. \end{aligned} \quad (6.29)$$

The temperatures or heights can be approximated with univariate normal distribution. The extension of the GMM is the same for the case of one variable or multiple variables. The theory behind mixture models, as well as the EM algorithm, is fully developed in [239].

6.2.3 Bayesian Metrics

BIC is a metric to choose between models but penalizing the models that have higher number of parameters, and have more samples [290]. The BIC in a mixture model is,

$$\begin{aligned} \mathcal{BIC}(\hat{\boldsymbol{\theta}})_L &= \lambda \log N - 2 \log \mathcal{Q}(\hat{\boldsymbol{\theta}}) \\ &= \lambda \log N - 2 \left\{ \sum_{\ell=1}^L \sum_{i=1}^N \mathbb{I}(\gamma_{i,\ell} = \ell) \left[\log \pi_\ell + \log p(\mathbf{x}_i | \hat{\boldsymbol{\theta}}_\ell) \right] \right\}, \end{aligned} \quad (6.30)$$

where λ is the number of parameters in the model, and N is the number of samples. As a pixel is assumed to be in one wind layer or another $\mathbb{I}(\gamma_{i,\ell} = \ell) \in \{0, 1\}$.

The BIC is close related to Akaike Information Criterion (AIC) [7],

$$\mathcal{AIC}(\hat{\boldsymbol{\theta}})_L = 2\lambda - 2\log \mathcal{Q}(\hat{\boldsymbol{\theta}}). \quad (6.31)$$

In other metrics, such as the Classification Likelihood Criterion (CLC),

$$\mathcal{CLC}(\hat{\boldsymbol{\theta}})_L = 2\mathcal{H}(\hat{\boldsymbol{\theta}}) - 2\log \mathcal{Q}(\hat{\boldsymbol{\theta}}), \quad (6.32)$$

uses the entropy function $\mathcal{H}(\cdot)$ in the context of information theory. CLC is similar to the AIC [244], but applying the entropy as a penalizing factor instead of the number of parameters. The entropy in a mixture model is,

$$\mathcal{H}(\hat{\boldsymbol{\theta}})_L = \sum_{\ell=1}^L \sum_{i=1}^N \gamma_{i,\ell} \log \gamma_{i,\ell}. \quad (6.33)$$

The ICL, which is

$$\mathcal{ICL}(\hat{\boldsymbol{\theta}})_L = \mathcal{BIC}(\hat{\boldsymbol{\theta}})_L + 2\mathcal{H}(\hat{\boldsymbol{\theta}}), \quad (6.34)$$

is based on both BIC and the entropy.

The number of clusters L and the likelihood function, is different in each model \mathcal{M}_L , thus each model is expected to have a different $\mathcal{BIC}(\hat{\boldsymbol{\theta}})_L$, $\mathcal{AIC}(\hat{\boldsymbol{\theta}})_L$, $\mathcal{CLC}(\hat{\boldsymbol{\theta}})_L$, and $\mathcal{ICL}(\hat{\boldsymbol{\theta}})_L$. For all those metrics, the optimal number of clusters is when the value of the metric is the lowest.

6.2.4 Hidden Markov Model

A HMM is state space model which latent variables (i.e. system states) are discrete. In the problem of detecting the number of wind velocity fields in an image, a HMM is implemented to infer the cluster number L in a mixture model. We assume that the current state of the system L_k (i.e. the hypothesis over the number of clusters at instant k) is a Markov process conditional to the previous observed states $p(L_k \mid L_1, \dots, L_{k-1})$

[31]. For simplification, we propose to model the process as a first-order Markov chain, whose current state is only conditional to the immediately previous state $p(L_k | L_1, \dots, L_{k-1}) = p(L_k | L_{k-1})$. Henceforth, L is defined as the HMM latent variable, which represents the number of detected wind velocity fields $L \in \{1, 2\}$ in image k , and $\mathbf{x}_{i,k}$ is an observation (i.e., the feature vector).

The parameters $\boldsymbol{\theta}_{\ell,k} = \{\pi_{\ell,k}, \boldsymbol{\mu}_{\ell,k}, \boldsymbol{\Sigma}_{\ell,k}\}$ of each distribution ℓ in the mixture model, and the hidden state of the system L_k in image k are the MAP estimation obtained applying the Bayes' theorem,

$$\begin{aligned} p(\gamma_{i,k} | \mathbf{x}_{i,k}, \boldsymbol{\Theta}_k) &= \frac{p(\mathbf{x}_{i,k}, \gamma_{i,k} | \boldsymbol{\Theta}_k)}{p(\mathbf{x}_{i,k})} \\ &\propto p(\mathbf{x}_{i,k}, \gamma_{i,\ell,k} | \boldsymbol{\theta}_{\ell,k}) p(L_k | L_{k-1}) \\ &\propto p(\mathbf{x}_{i,k}, \gamma_{i,\ell,k} | \boldsymbol{\mu}_{\ell,k}, \boldsymbol{\Sigma}_{\ell,k}) p(\pi_{\ell,k} | \boldsymbol{\alpha}_{\ell}) p(L_k | L_{k-1}). \end{aligned} \quad (6.35)$$

where the set with all the parameters in the mixture model in state L_k are $\boldsymbol{\Theta}_k = \{\pi_{1,k}, \boldsymbol{\mu}_{1,k}, \boldsymbol{\Sigma}_{1,k}, \dots, \pi_{L_k,k}, \boldsymbol{\mu}_{L_k,k}, \boldsymbol{\Sigma}_{L_k,k}\}$, and the parameters of the prior distribution of the clusters weights π_{ℓ} are $\boldsymbol{\alpha}_{\ell}$.

The joint distribution $p(\mathbf{x}_{i,k}, \gamma_{i,\ell,k} | \boldsymbol{\theta}_{\ell,k}) = p(\mathbf{x}_{i,k} | \gamma_{i,\ell,k}, \boldsymbol{\theta}_{\ell,k}) p(\gamma_{i,\ell,k})$ is factorized to independently infer a mixture model of each feature in vector $\mathbf{x}_{i,k}$.

The posterior distribution of a mixture model is proportional to the joint probability of the observations $\mathbf{x}_{i,k}$ and the responsibilities $\gamma_{i,\ell,k}$ of each cluster detected in image k . The state of the system L_k , which represents the number of clusters L in the mixture models, is modelled using a HMM. Therefore, the CDLL in Eq. (6.5) can be computed using the following joint distribution:

$$p(\mathbf{X}_k, \boldsymbol{\Gamma}_{\ell,k} | \boldsymbol{\theta}_{\ell,k}, \boldsymbol{\alpha}_{\ell}) \triangleq \left\{ \sum_{i=1}^N \left[\prod_{\ell=1}^{L_k} [\pi_{\ell} p(\mathbf{x}_{i,k} | \boldsymbol{\theta}_{\ell,k})]^{\mathbb{I}(\gamma_{i,k}=\ell)} \right] \right\} p(\pi_{\ell} | \boldsymbol{\alpha}_{\ell}). \quad (6.36)$$

The transition probability on the latent variable L_k , is defined as a distribution of

the exponential family,

$$\log p(L_k|L_{k-1}) = \frac{1}{Z} \exp[-\psi(L_k, L_{k-1})], \quad (6.37)$$

where the exponent $\psi(L_k, L_{k-1})$, is a function that depends on the previous state of the system L_{k-1} ,

$$\psi(L_k, L_{k-1}) \triangleq \begin{cases} -\beta & \text{if } L_k = L_{k-1} \\ +\beta & \text{if } L_k \neq L_{k-1} \end{cases}, \quad (6.38)$$

the parameter β has to be cross-validated.

Combining Eq. (6.36) and Eq. (6.37) in Eq. (6.35), and taking logarithms and expectations with respect to $\gamma_{i,k}$, the CDLL of a image k being in state L_k are,

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1}) = & \left[\sum_{i=1}^N \sum_{\ell=1}^{L_k} \gamma_{i,\ell,k} \log \pi_{\ell,k} + \sum_{i=1}^N \sum_{\ell=1}^{L_k} \gamma_{i,\ell,k} \log p(\mathbf{x}_{i,k}|\boldsymbol{\theta}_{\ell,k}) \right] \\ & + \log p(\pi_\ell|\hat{\boldsymbol{\alpha}}_\ell) - \psi(L_k, L_{k-1}) + \text{constant}. \end{aligned} \quad (6.39)$$

It should be noted that $\psi(L_k, L_{k-1})$ is a constant with respect to $\gamma_{i,k}$, so maximizing this equation is equivalent to maximizing the original CDLL in Eq. (6.36).

After completing the inference of the mixture model parameters $\hat{\boldsymbol{\theta}}_{\ell,k}$ and $\hat{\boldsymbol{\alpha}}_\ell$ when $L_k = 1$ and $L_k = 2$, the optimal state of the system $\hat{L}_k \in \{1, 2\}$ is the MAP estimation obtained from,

$$\hat{L}_k = \underset{L_k \in \{1,2\}}{\operatorname{argmax}} \left[\sum_{i=1}^N \sum_{\ell=1}^{L_k} \log p(\gamma_{i,k}|\mathbf{x}_{i,k}, \hat{\boldsymbol{\theta}}_{\ell,k}, \hat{\boldsymbol{\alpha}}_\ell, L_k) \right] - \psi(L_k, L_{k-1}). \quad (6.40)$$

6.3 Experiments

6.3.1 Image Preprocessing

The IR images were preprocessed to remove the effects of the direct radiation from the Sun (i.e., irradiance), the scattered irradiance from the atmosphere and the radiation

emitted by particles on the germanium outdoor camera window. The image processing methods and atmospheric conditions model are fully described in Chapter 3.

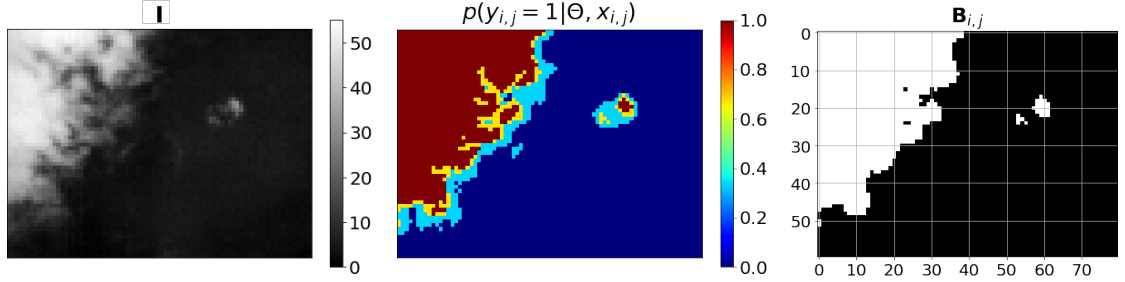


Figure 6.1: Cloud segmentation in IR images. The first image shows an IR image normalized to 8 bits, the second image shows the probabilities computed by the segmentation algorithm of pixels belonging to a cloud, and the third image shows the segmentation after applying a ≥ 0.5 threshold to the probabilities. The normalized images are also used to compute the velocity vectors in the proposed WLK.

The proposed algorithm for the detection of clouds in multiple wind velocity fields requires that pixels containing clouds be previously segmented in the images. In this way, only features from clouds containing pixels are analyzed. The cloud segmentation algorithm implemented in this chapter is a voting scheme that uses three different cloud segmentation models. The segmentation models are a Gaussian process, a support vector machine and unsupervised Markov Random Field (see Figure 6.1). The cloud segmentation models and feature extraction are explained in Chapter 4.

6.3.2 WLK Parameters Cross-Validation

A series of images with clouds flowing through different directions were simulated to cross-validate the set of parameters for each one of the mentioned methods. The WLK method was found to be the most suitable for this application. The method proposed in Appendix A was searching for a dense implementation of a motion vector method to approximate the dynamics of a cloud. The most suitable method was found to be

WLK. The optimal window size, WLS regularization, and temporal kernel amplitude are: $W = 16$ [pixels²], $\tau = 1 \times 10^{-8}$ and $\sigma = 1$.

6.3.3 Mixture Models Parameters Cross-Validation

The parameters that have to be cross-validated for each mixture model are α_ℓ and β . The parameter α_ℓ in Eq. (6.8) is the parameter of the prior distribution of the cluster weight π_ℓ in a mixture model. The parameters $\boldsymbol{\alpha}$ in a mixture model of the temperatures are cross-validated as $\boldsymbol{\alpha} \triangleq \alpha_0 \mathbf{1}_{L \times 1}$ for simplification. Equivalently, the parameters $\boldsymbol{\alpha}$ in a mixture model of the velocity vectors are cross-validated as $\boldsymbol{\alpha} \triangleq \alpha_1 \mathbf{1}_{L \times 1}$. The parameter β in Eq. (6.38) is the prior of the number of cloud layers in an image used in the sequential HMM. Both in training and in testing, the state L_k is initialized to the opposite number of cloud layers in the IR image sequence (e.g. if $L_k = 2$ in the sequence, L is initialized as $L_0 = 1$).

The parameter's cross-validation was implemented using a HPC. Even when the parameter's cross-validation is implemented in a HPC, it is still computationally expensive and the number of validation samples is prohibitive. The cross-validation used two nodes and it was distributed across sixteen cores, which corresponds with the number of possible β in the cross-validation $\beta = \{0, \dots, 1000\}$. In each core, all possible combinations of α_0 , which is the parameter of the priors of the clusters corresponding to temperature, and α_1 (velocity vector cluster priors) were cross-validated $\alpha_\ell = \{1, \dots, 1000\}$. The performance of each combination of parameters for each mixture model were evaluated in the six training sequences. The optimal combination of parameters for each mixture model is the one which achieved the highest accuracy.

The training dataset is formed by six sequences of 21 consecutive images acquired on six different days. The training sequences were captured on different seasons and

different times of the day. IR images were manually labelled as having one cloud layer $L = 1$ or two cloud layers $L = 2$. The images from three of the six days show a layer of cirrostratus in winter during the morning, altostratus in spring during the afternoon and stratocumulus in the summer during the afternoon. The other three days show two layers of altostratus and stratocumulus in winter at noon, cirrostratus and altocumulus in spring during the afternoon, and cirrocumulus and cumulus in summer during the morning. As the proposed method is an online ML algorithm, the training dataset is used only for validating the prior distribution parameters. The optimal parameters of the mixture models are computed for each new sample during the implementation.

6.3.4 Testing Performance

The testing dataset is composed of ten consecutive sequences of 30 images. The sequences were acquired at different hours of the day and in different seasons. The images in the testing dataset were acquired after the training dataset. The images in the testing dataset were manually labelled in the same way as the training set. The testing dataset includes five sequences of images that have one layer of clouds, and five sequences of images that have two layers of clouds. The clouds in the sequences with one layer are: stratocumulus on a summer morning, cumulus on a summer morning, stratus on a summer afternoon, cumulus on a fall morning, and stratocumulus on a winter morning. The clouds in the sequences with two layers are: cumulus and cirrostratus on a summer morning, cumulus and altostratus on a fall morning, cumulus and cirrus on a fall afternoon, stratocumulus and altostratus on a fall morning, and cumulus and nimbostratus on a winter afternoon.

We assume that the distribution of the velocity vectors is different in each cloud layer that appears in an IR image. In addition, we assume that a correlation exists between the height of a cloud and its velocity vectors. As the height of a cloud is a

Chapter 6. Detection of Clouds in Multiple Wind Velocity Fields

Table 6.1: Detection accuracy achieved when univariate probability functions are used in a mixture model likelihood. The detection accuracy is compared using different Bayesian metrics and a mixture model with a prior on the weights and the clusters number.

Multiple Cloud Layers Detection Accuracy [%]									
Univariate Likelihoods	\mathcal{Q}	Bayesian Metrics				Cross-Validated			MAP-HMM
		BIC	AIC	CLC	ICL	α_0	α_1	β	$\psi(L_k = L_{k-1})$
$\mathcal{B}(\hat{T}_{i,j}'')$	66.67	70.74	70.74	76.3	76.67	1	-	200	75.93
$\mathcal{G}(\hat{T}_{i,j}'')$	60	61.11	60.74	66.67	70.74	1	-	300	53.55
$\mathcal{N}(T_{i,j}'')$	74.07	73.7	71.85	79.63	81.11	100	-	500	87.41
$\mathcal{VM}(\omega_{i,j})$	57.41	60.37	61.85	58.52	61.11	-	1	500	60
$\mathcal{G}(m_{i,j})$	41.48	40.37	42.22	39.63	36.67	-	10	1000	37.78

Table 6.2: Detection accuracy achieved when multivariate probability functions are used in the likelihood of the mixture model. The detection accuracy is compared using different Bayesian metrics and a maximum a posteriori implementation of a mixture model with a prior on the weights and cluster numbers.

Multiple Cloud Layers Detection Accuracy [%]									
Multivariate Likelihoods	\mathcal{Q}	Bayesian Metrics				Cross-Validated			MAP-HMM
		BIC	AIC	CLC	ICL	α_0	α_1	β	$\psi(L_k = L_{k-1})$
$\mathcal{N}(\hat{\mathbf{v}}_{i,j})$	65.93	68.15	67.04	69.63	67.41	-	100	600	61.85
$\mathcal{N}(T_{i,j}'', \hat{\mathbf{v}}_{i,j})$	49.63	52.96	51.11	55.93	62.96	1000	-	1000	55.19
$\mathcal{BG}(\hat{T}_{i,j}'', m_{i,j})$	50	50.37	50	50.37	50.74	1	-	0	50
$\mathcal{BG}(\hat{T}_{i,j}'', m_{i,j}) \mathcal{VM}(\omega_{i,j})$	50	50	50	48.89	51.11	1	1	0	50
$\mathcal{B}(\hat{T}_{i,j}'') \mathcal{N}(\hat{\mathbf{v}}_{i,j})$	70	71.11	70.37	75.56	75.93	10	1000	650	94.44
$\mathcal{G}(\hat{T}_{i,j}'') \mathcal{N}(\hat{\mathbf{v}}_{i,j})$	61.48	61.48	62.96	55.19	57.04	1	100	450	68.15

function of its temperature, we propose to use the temperature of the pixels and the velocity vectors to infer the number of cloud layers in an IR image. The distribution of temperatures is inferred using a BeMM, GaMM and GMM, see Figure 6.2. The performances of each distribution are analyzed and compared in tables 6.1-6.2. The posterior probabilities of the temperature mixture models in Figure 6.2 are the weights used in the WLK.

The distribution of the velocity vector components is inferred using a multivariate GMM. The performance of the multivariate GMM is compared to the distribution of the velocity vectors magnitude and angle inferred factorizing the probability of the velocity vectors into two independent probability functions (see table 6.2). In

Table 6.3: Detection accuracy achieved when the mixture model likelihood is factorized in the product of independent likelihood functions for each feature. The detection accuracy is compared using different Bayesian metrics and adding a prior of the weights and the cloud layers number to the mixture model.

Multiple Cloud Layers Detection Accuracy [%]									
Factorized Univariate Likelihoods	\mathcal{Q}	Bayesian Metrics				Cross-Validated			MAP-HMM $\psi(L_k = L_{k-1})$
		\mathcal{BIC}	\mathcal{AIC}	\mathcal{CLC}	\mathcal{ICL}	α_0	α_1	β	
$\mathcal{B}(\tilde{T}_{i,j}'') \mathcal{VM}(\omega_{i,j})$	68.89	68.89	69.63	73.33	74.44	1	10	650	97.41
$\mathcal{B}(\tilde{T}_{i,j}'') \mathcal{G}(m_{i,j})$	51.48	48.89	49.26	52.96	49.63	1	1000	650	69.26
$\mathcal{B}(\tilde{T}_{i,j}'') \mathcal{VM}(\omega_{i,j}) \mathcal{G}(m_{i,j})$	55.19	55.19	55.56	57.41	56.3	10	100	1000	88.15
$\mathcal{G}(\tilde{T}_{i,j}'') \mathcal{VM}(\omega_{i,j})$	50.37	50.37	62.96	58.89	60	10	1	400	67.78
$\mathcal{G}(\tilde{T}_{i,j}'') \mathcal{G}(m_{i,j})$	62.22	62.22	50	50.37	50.74	10	10	400	54.81
$\mathcal{G}(\tilde{T}_{i,j}'') \mathcal{VM}(\omega_{i,j}) \mathcal{G}(m_{i,j})$	53.7	56.67	52.96	46.3	50.74	10	1	650	51.11
$\mathcal{N}(\tilde{T}_{i,j}'') \mathcal{VM}(\omega_{i,j})$	78.15	76.67	77.41	72.96	71.85	100	1000	600	85.93
$\mathcal{N}(\tilde{T}_{i,j}'') \mathcal{G}(m_{i,j})$	64.07	66.67	64.44	61.85	70.37	1000	100	500	80.37
$\mathcal{N}(\tilde{T}_{i,j}'') \mathcal{VM}(\omega_{i,j}) \mathcal{G}(m_{i,j})$	73.33	67.78	68.15	72.96	68.89	100	1	400	72.96
$\mathcal{VM}(\omega_{i,j}) \mathcal{G}(m_{i,j})$	51.11	48.52	48.15	42.22	43.70	-	10	1000	57.78

Figure 6.3, the distributions of the velocity angle and magnitudes are inferred using a VMMM and GaMM respectively.

When weights are not applied to the LK method, the distribution of the temperatures, velocity vector angles and magnitude can be inferred using a multivariate GMM. Similarly, a DGaMM is also proposed to infer the distribution of the temperatures and velocity vector magnitudes. The multivariate GMM and DGaMM likelihood are displayed in Figure 6.4. In this case, the probability of the velocity vector angles is factorized and inferred independently using a VMMM. The performance of these two mixture models are also shown in table 6.2.

The experiments were carried out in the Wheeler HPC of UNM-CARC, which uses SGI AltixXE Xeon X5550 at 2.67GHz with 6 GB of RAM per core, has 8 cores per node, 304 nodes total, and runs at 25 theoretical peak FLOPS. It has Linux CentOS 7 installed.

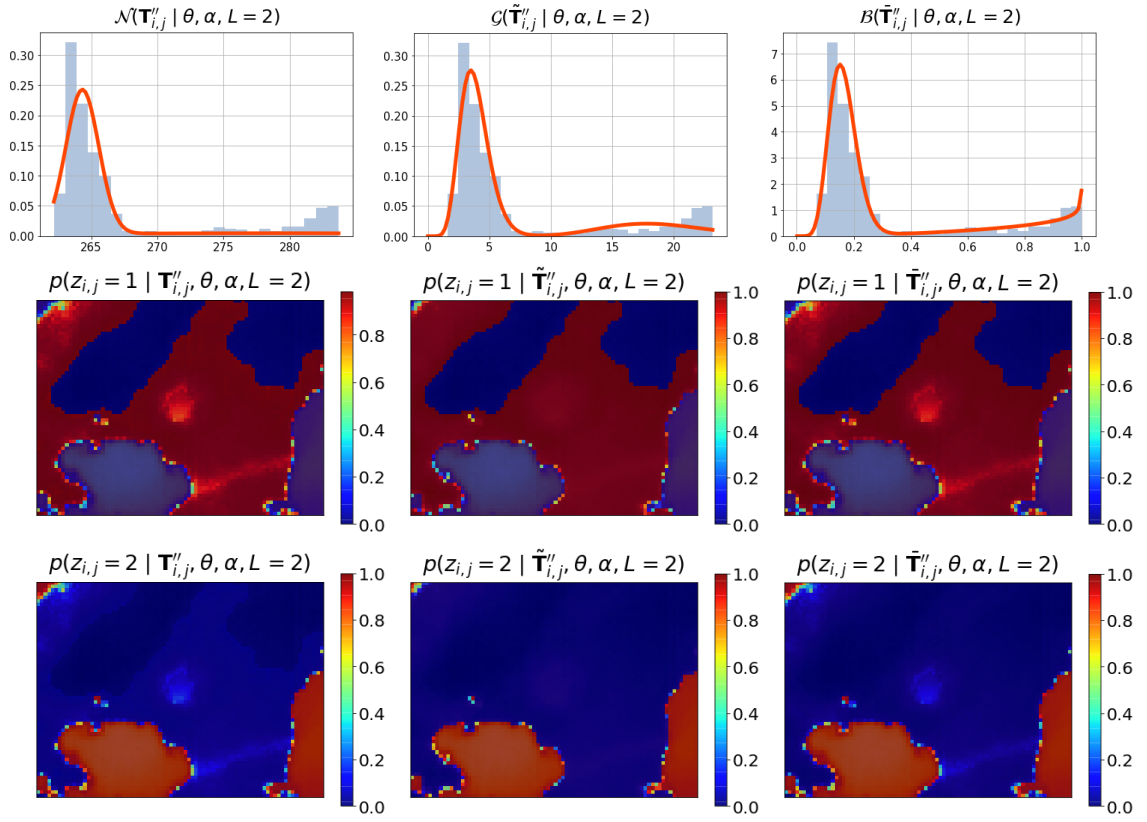


Figure 6.2: First row shows the distribution of the temperatures inferred using a GMM, GaMM and BeMM. The mixture model likelihood is evaluated with the optimal parameters. The mixture model posterior probabilities for the top cloud layer and bottom cloud layer are shown in the middle and bottom rows respectively.

6.4 Discussion

In the problem at hand, the BIC and AIC criteria do not produce an improvement on the detection accuracy with respect to accuracy achieved by MLE criterion. The best detection accuracy achieved by a model that uses the MLE criterion is 78.15%. In contrast, the same model achieved a detection accuracy of 76.67% and 77.41% when the criteria were minimum BIC and AIC respectively (see table 6.3). This mixture model has a factorized likelihood which uses a normal probability function to infer the temperature distribution and a Von Mises to infer the velocity vector angles. However,

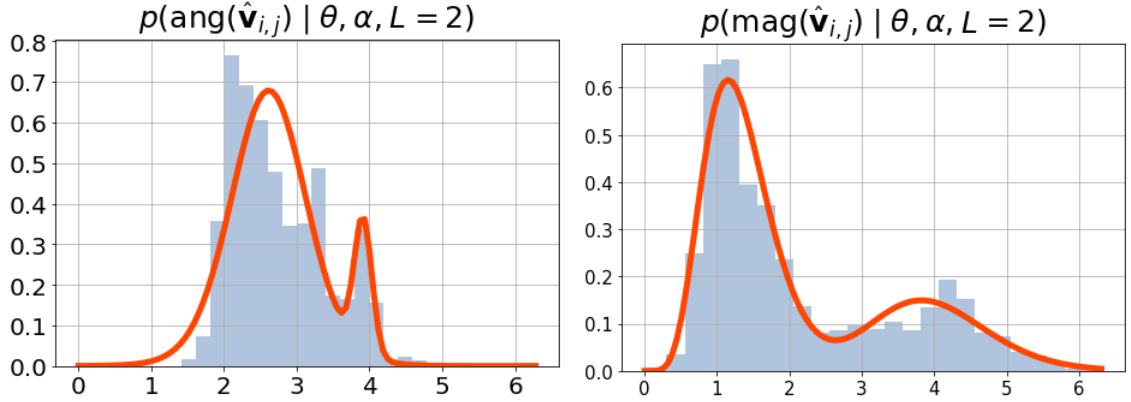


Figure 6.3: Distribution of the velocity vector's angle and magnitude. The mixture model likelihood is evaluated with the optimal parameters. The distribution of the velocity vector angle was inferred using a VMMM (left). The distribution of the velocity vector magnitude was inferred using GaMM (right).

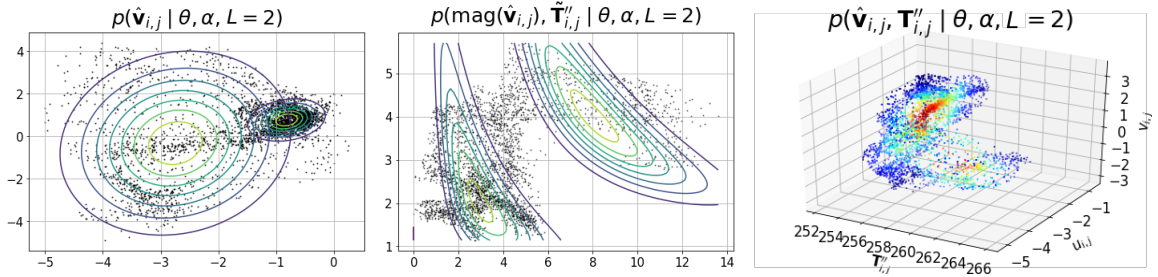


Figure 6.4: The distributions of the temperatures and the velocity vectors. The graphs show the density of mixture model likelihood evaluated with the optimal parameters. The distribution of the velocity vectors was inferred using a multivariate GMM (top left graph). The distribution of the temperatures and the velocity vector's magnitude was inferred using a BGaMM (top left graph). The distribution of the temperatures and the velocity vectors was inferred using a multivariate GMM (bottom graph).

when the minimum CLC and ICL criteria are applied, the detection accuracy improved with respect to that achieved by the MLE criterion. The detection accuracy achieved by CLC, ICL and MLE criteria were 81.11%, 79.63% and 74.07% respectively (see table 6.1). Therefore, the best detection using a Bayesian metric was performed by a mixture model with a normal likelihood on the temperatures.

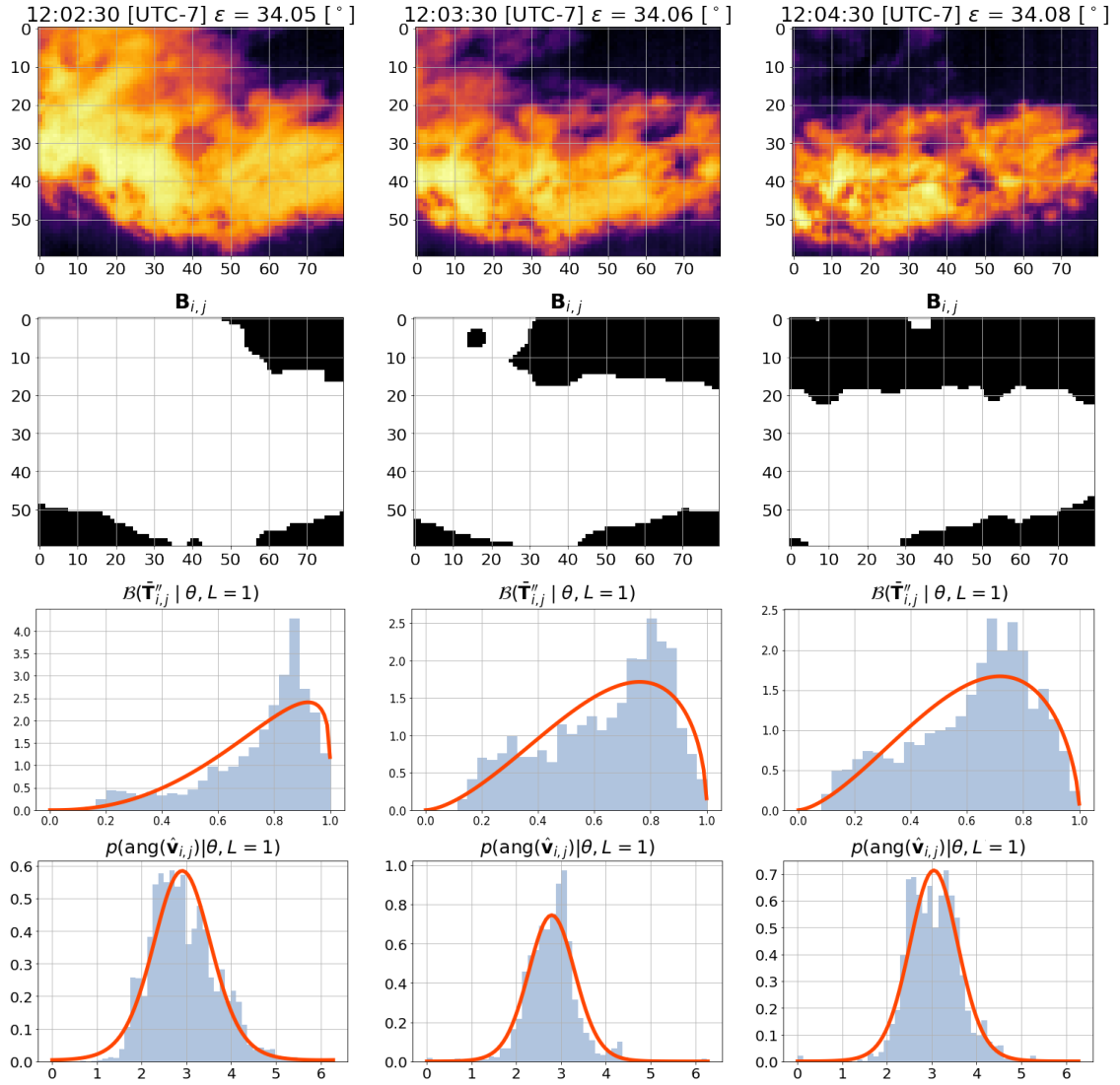


Figure 6.5: Testing sequence of consecutive IR images acquired at 1 minute interval. In the first row, the IR images show a cloud flowing in a single detected wind velocity field. The images in the second row show the cloud segmentation. The graphs in the third and fourth row show the selected model distribution of the temperatures and the velocity vector angles respectively.

The detection accuracy of the proposed algorithm increases when the mixture model includes prior distributions on the mixture weights and the number of clusters. In these mixture models, the decision criterion is MAP. Adding a prior to the mixture

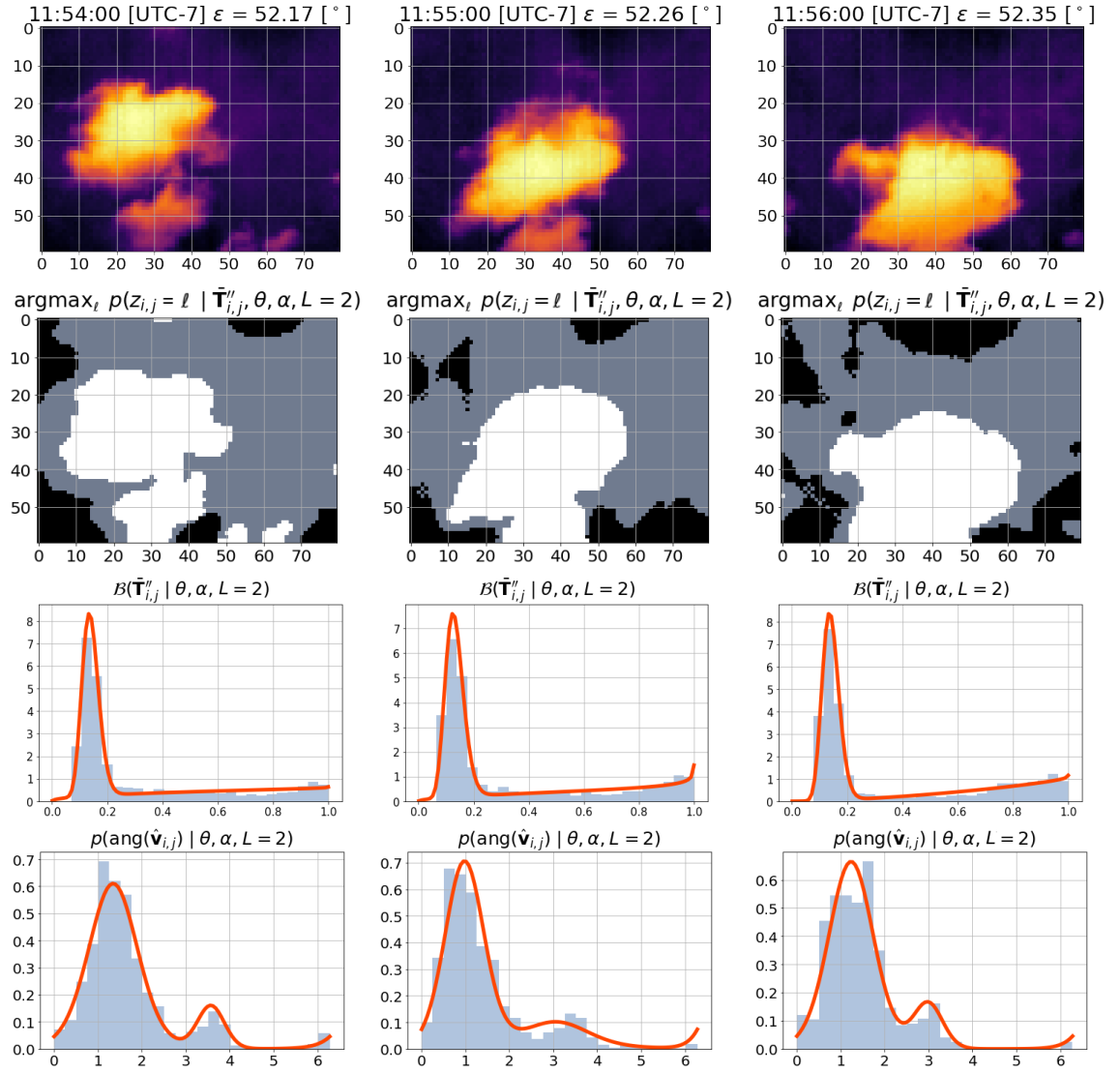


Figure 6.6: Testing sequence of IR images with two detected cloud layers (first row). The time interval between images is 1 minute. The images in the second row show the pixels that belong to the low and high temperatures, in gray and white respectively. The pixels in black were classified as not belonging to a cloud by the segmentation algorithm. The third and fourth row show the distribution of the temperature and velocity vectors of the best model.

weights and the cluster number is equivalent to regularizing the parameters. The prior adds certain known information to the model. In our problem, the prior on

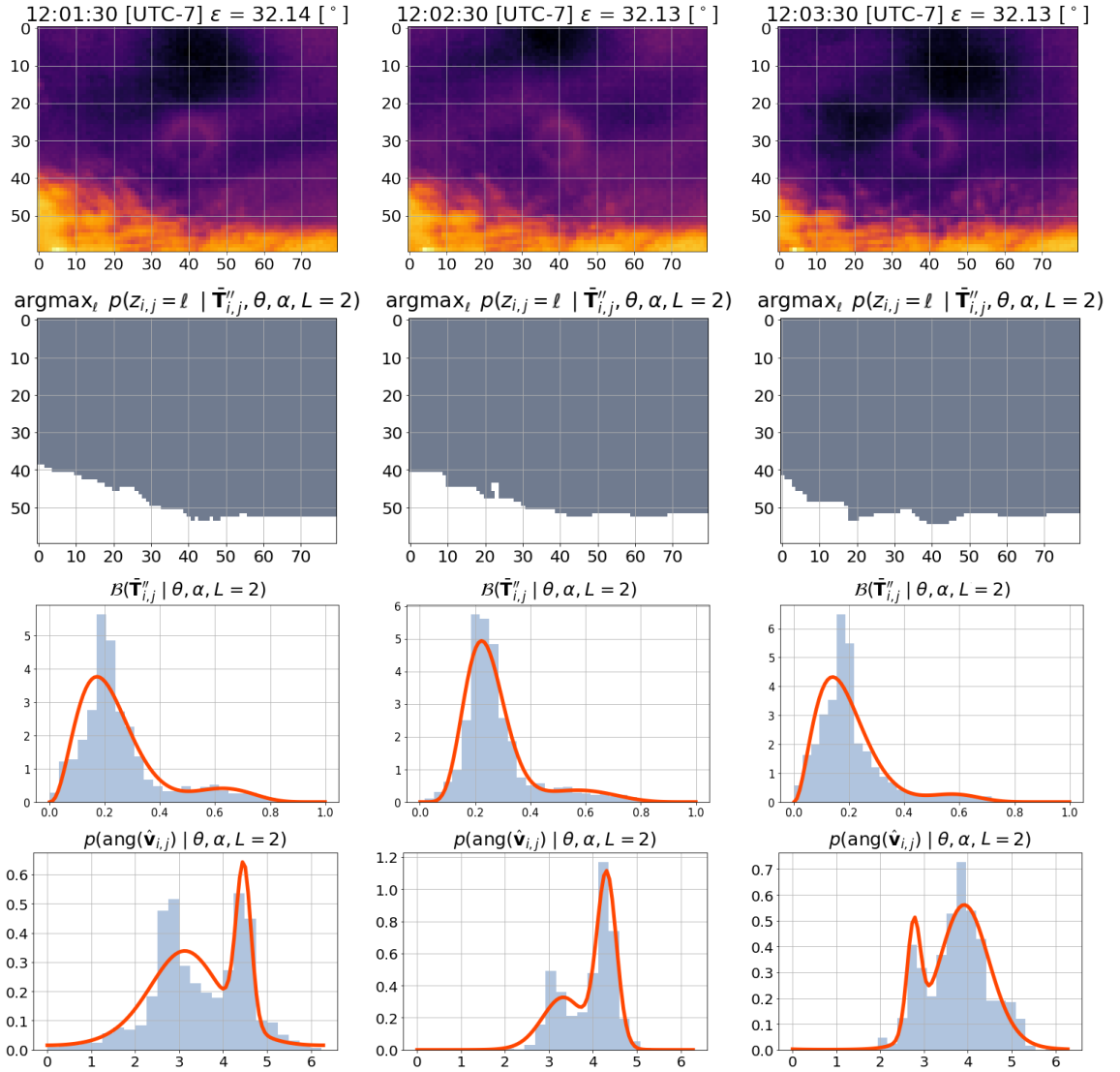


Figure 6.7: Testing image with two detected cloud layers. The time interval between the IR image is 1 minute. The consecutive IR images are shown in the first row. The pixel labels of the top (grey) or the bottom cloud layer (white) are shown in the second row. The third and fourth row show the factorized likelihood that a VMMM and GaMM uses for the velocity vector angles and magnitude respectively.

the number of clusters produces the following effect: if the previous frame had L_{k-1} clusters, the next frame is more likely to have L_k as well. Similarly, the prior on the mixture weights assures that when the likelihood of two cloud layers is inferred, the

cluster weights cannot vanish to zero. In table 6.1, when we look at the model that achieved the best detection accuracy using a Bayesian metric (ICL), the detection accuracy increased from 81.11% to 87.11%. Nevertheless, the best detection accuracy using the MAP criterion reached 97.41% (see table 6.3). The model that presents the best detection accuracy is a MAP mixture model with factorized likelihood which uses a beta probability function to infer the temperature distribution and a Von Mises distribution to infer the velocity vector angles. This validates our assumption that different cloud layers are at different heights (i.e. temperature) and hence the wind shear is also different (i.e. velocity vector angle). The proposed likelihood factorization allows us to find the optimal probability function of each feature independently.

The results show that it is feasible to identify different cloud layers in IR ground-based sky-images (see Figure 6.5-6.7). The main advantage of this algorithm is that it provides the capability of independently estimating the motion of different cloud layers in an IR image using the posterior probabilities in Figure 6.2. This is useful in predicting when different clouds will occlude the Sun. The features and dynamics can be analyzed independently to increase the performances of a solar forecasting algorithm. Another advantage of the learning algorithm proposed is that it is unsupervised, so the user does not need to provide labels, which makes the training process automatic.

As it can be seen in tables 6.1-6.3, the Bayesian metrics are not useful in this application. The highest accuracy achieved by a Bayesian metric was 81.11% with a GMM of the temperatures. The model selection was performed using minimum ICL criterion. The performances of the BGaMM are lower than the rest of the mixture models, thus it is not practical to assess the number of cloud layers in IR images. This is because the BGaMM tends to overfit even when the cluster weights are regularized using a prior distribution.

A disadvantage of the proposed unsupervised learning algorithm is that the EM requires several initializations to guarantee that the EM algorithm converges to the

best local maxima. This is problematic when the cloud layer detection algorithm is meant for real-time applications. An implementation of the algorithm feasible in real-time applications will require multiple CPUs to run different initializations in parallel.

6.5 Conclusions

This chapter proposes an online unsupervised learning algorithm to detect moving clouds in different wind velocity fields. The mixture model of the pixel temperature is used to know when a cloud is below or on top of another. The posterior probabilities of the mixture model are used to compute velocity vectors. The algorithm to compute the velocity vectors is a weighted implementation of the LK optical flow (see Appendix A.1.1 for more information). The weights are the posterior probabilities of the mixture model. The velocity vectors are computed in a scenario that assumes one cloud layer and in another scenario that assumes two cloud layers (using the cloud segmentation or the posterior probabilities respectively). The distribution of the velocity vectors and the temperatures is used to determine which one of the analyzed scenarios is the most likely. The proposed algorithm implements the MAP criterion.

The detection of clouds flowing in different wind velocity fields is useful to increase the accuracy of a forecasting algorithm that predicts the GSI that will reach a PV power plant. The prediction will aid a SG to adjust the generation mix to compensate for the decrease of energy generated by the PV panels.

In particular, the posterior probabilities of the pixel temperatures may aid the extraction of features using either image processing techniques, gradient-based learning (e.g. deep neural networks) or both. However, the posterior probabilities are only advantageous when there are multiple cloud layers in an IR image. The proposed method models a prior distribution of the cluster weights, and a prior function of each

possible scenario. The prior function of the scenarios is a temporal implementation of a hidden Markov model. This chapter shows that the proposed method increases the detection accuracy compared to the accuracy achieved by the most common Bayesian metrics used in practice.

Future work in this area will implement cloud detection algorithms in a ramp-down and intra-hour solar forecasting algorithm. The dynamics of clouds may be analyzed independently to extract features from clouds moving in different wind velocity fields. The improvement in the performance can be assessed to determine how to combine the features extracted from different clouds to model their respective influence on the GSI that will reach the surface of a PV system. Another investigation may focus on the implementation of the proposed algorithm in images acquired using ground-based all-sky imagers that are sensitive to the visible light spectrum instead of the IR. The most interesting aspect will be to fuse information acquired using visible and IR light cameras.

Chapter 7

Visualization of Multiple Wind Velocity Fields for Solar Forecasting

7.1 Introduction

This chapter aims to visualize the wind velocity field to anticipate interruptions in the supply of energy generated by PV systems [261]. The visualization of the wind velocity field requires measurements of wind velocity at a given altitude. The wind velocity increases with the altitude in the lower atmosphere [359]. The decrease of temperature along the Troposphere can be approximated by a linear function [114]. Cloud formations are feasible in a range of altitudes that varies from the ground to the Tropopause [273]. The detection of clouds in IR images allows us to indirectly measure physical magnitudes of the wind velocity field [296].

Methods of computational numerical analysis are an effective way to analyze images of clouds [48]. The direction and magnitude of cloud velocity have been estimated applying motion vector techniques to a series of consecutive frames [58]. Through image segmentation, it is possible to identify clouds and other objects in an image

[183]. The clouds' pathlines can be estimated by tracking them with a Kalman filter [105]. ML algorithms such as Artificial Neural Networks (ANN) [180], or Support Vector Machines for Regression (SVM) [74], are promising models to find space-time correlations in cloud images.

This chapter utilizes two innovations. First, an innovative sky imager for capturing radiometric long-wave IR images and GSI pyranometer measurements (see Chapter 2). As DAQ is equipped with a solar tracker that updates its pan and tilt every second to maintain the Sun in a central position in the images during the day [275], the sky images are taken at an angle from the normal position of the camera in relation to the ground. The angle is the Sun's elevation. This causes the relative distance of a given object on the horizon to increase from top to bottom in an image. To account for this effect, a second innovation is transform the velocity vectors from the original Euclidean frame of reference to a non-linear frame of reference (see Chapter 5).

This chapter also proposes and implements an online ML algorithm for predicting the streamlines of multiple wind flows in an image. An unsupervised ML algorithm infers the distribution of velocity vectors and heights of multiple layers of clouds. The velocity vectors are approximated using the WLK method, and are segmented and subsampled to reduce the noise of the approximation and the computational burden of the entire algorithm. A Multi-Task Weighted Support Vector Machine (ε -MT-WSVM) [339] visualizes the approximated velocity vectors to predict the trajectories of the clouds. The ε -MT-WSVM is modified from its original form adding flow constraints. The flow constraints are added so that the approximated streamlines are equivalent to the pathlines. The wind velocity field visualization can be used to forecast occlusion of the Sun by clouds, thereby predicting and preventing disruptions in the generation of energy from solar power plants.

7.2 Wind Velocity Field

When there are multiple layers of clouds in an image, a mixture model of the temperature of the pixels is expected to have multiple clusters. The number of clusters L is estimated by a previously trained detection algorithm that infers the number of wind velocity fields which are in an image using the processed temperatures and the angles of the cloud velocity vectors (see Chapter 6).

7.2.1 Motion Vectors

In the application of estimating the cloud velocity vectors, the most suitable method is the WLK [18], but instead of weighting the neighboring pixels with a multivariate normal distribution, the weights $\gamma_{i,j,\ell}$ are the posterior probabilities of a BeMM (see Chapter 6). For doing this, two consecutive normalized images \mathbf{I}_{k-1} and \mathbf{I}_k are used to approximate the cloud velocity vectors in each pixel i, j and each cloud layer ℓ in in image, and considering the processed temperatures $\bar{T}_{i,j}''$ of a given image (by omitting index k), a BeMM infers the distribution of the normalized processed temperatures $\bar{T}_{i,j}''$, so that $\gamma_{i,j,\ell} \triangleq p(y_{i,j} = \ell \mid \bar{T}_{i,j}'', \boldsymbol{\theta})$ is the responsibility of the cluster ℓ , with parameters $\boldsymbol{\theta} = \{\alpha_\ell, \beta_\ell\}$, in the sample i, j .

The estimation of cloud velocity vectors component in the x-axis is $\hat{\mathbf{U}}_\ell = \{\hat{u}_{i,j,\ell} \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$, and the velocity vector component in the y-axis is $\hat{\mathbf{V}}_\ell = \{\hat{v}_{i,j,\ell} \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$. The velocity vectors are in units of pixels per frame, but knowing the geometrical transformation of the frame, they can be transformed to meters per second (see Chapter 5). The geometric transformation is a function of the Sun's elevation and azimuth angles $\psi : (\varepsilon, \alpha) \mapsto \Delta \mathbf{x}_{i,j}$ in a frame, it defines the dimensions of a pixel at a given height $\Delta \mathbf{X} = \{(\Delta x, \Delta y)_{i,j} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$. This transformation connects the x,y-axis coordinates system with the height, which is the z-axis. The relation

holds even when the height is an approximation, since the components of velocity vectors are transformed with respect to the new coordinates system.

The cloud average heights in a frame are computed using the posterior probabilities $\gamma_{i,j,\ell}$ in a frame, but only in the pixels with a cloud,

$$h_\ell = \frac{\sum_{i,j} \gamma_{i,j,\ell} \cdot H''_{i,j} \cdot \mathbb{I}(b_{i,j} = 1)}{\sum_{i,j} \gamma_{i,j,\ell} \cdot \mathbb{I}(b_{i,j} = 1)}, \quad (7.1)$$

where $H''_{i,j}$ are the height computed applying the MALR to the processed temperatures $T''_{i,j}$ (see Section 3.2.2), and $\mathbb{I}(\cdot)$ is the indicator function. An image segmentation algorithm indicates which pixels belong to a cloud, so that $\mathbf{B} = \{b_{i,j} \in \mathbb{B} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ is a binary image where 0 is a clear sky pixel, and 1 is a pixel belonging to a cloud (see Chapter 4). The velocity vectors of each cloud layer are transformed such as,

$$\begin{aligned} \hat{\mathbf{u}}_{i,j} &= \frac{\delta}{f_r} \cdot \Delta x_{i,j} \sum_{\ell=1}^L h_\ell \cdot \gamma_{i,j,\ell} \cdot \hat{\mathbf{u}}_{i,j,\ell} \\ \hat{\mathbf{v}}_{i,j} &= \frac{\delta}{f_r} \cdot \Delta y_{i,j} \sum_{\ell=1}^L h_\ell \cdot \gamma_{i,j,\ell} \cdot \hat{\mathbf{v}}_{i,j,\ell} \end{aligned} \quad (7.2)$$

where f_r is the frame rate of the sequence of images, and δ is velocity vectors' scale in the WLK approximation.

7.2.2 Velocity Vectors Selection

In order to approximate the potential lines and streamlines of the wind velocity field in a frame, we propose to select the most consistent velocity vectors over a sequence of consecutive frames. The main problems with this approach are that as the vectors are selected over a sequence of images, the amount of vectors is expected to be large; also when optical flow is implemented in dense manner, it yields to noisy vectors. Because of this, we threshold the velocity vectors to reduce both the algorithm's computational burden and the variance of the noise.

Velocity Vector Segmentation

The pixel intensity difference between two consecutive frames is computed to find the pixels that show a change. The pixel normalized intensities that were used to compute the velocity vectors are $\mathbf{I} = \{i_{i,j} \in \mathbb{R}^{[0,2^8)} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$. The root squared intensity normalized difference is,

$$d_{i,j} = \frac{\sqrt{(i_{i,j,k-1} - i_{i,j,k})^2}}{\sum_{i,j} \sqrt{(i_{i,j,k-1} - i_{i,j,k})^2}}. \quad (7.3)$$

Matrix \mathbf{D} with normalized differentials $d_{i,j}$ is vectorized and sorted from the lowest to the highest, i.e., $\mathbf{d} = \text{sort}(\text{vec}(\mathbf{D}))$. A vector \mathbf{r} with the accumulated variance is computed as

$$r_m = \left\{ \sum_{p=1}^m \mathbf{d}_p \right\}_{m=1}^{M \cdot N}. \quad (7.4)$$

Then, vector \mathbf{r} is reorganized and set in the original matrix form, defined as $\mathbf{R} = \{r_{i,j} \in \mathbb{R}^{[0,1)} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$. Finally, a threshold τ is applied

$$s_{i,j} = \begin{cases} 1 & r_{i,j} \geq \tau \\ 0 & \text{Otherwise,} \end{cases} \quad (7.5)$$

where $\mathbf{S} \in \mathbb{B}$ is a binary image whose elements are 1 when a pixel is selected. The threshold velocity vectors in a frame k are $\hat{\mathbf{V}}'_k = \{\hat{\mathbf{v}}_{i,j,k} = \{\hat{u}_{i,j,k}, \hat{v}_{i,j,k}\} \wedge s_{i,j,k} = 1 \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$.

Based on the assumption that a cloud floating in the air follows a trajectory dictated by the wind velocity field, the wind velocity field is approximated using the segmented velocity vectors of ϱ last frames (i.e., time series lag). Hence, the set of

velocity vectors available to compute the wind velocity field are,

$$\hat{\mathbf{V}}_k'' = \begin{bmatrix} \hat{\mathbf{V}}_k' \\ \vdots \\ \hat{\mathbf{V}}_{k-\varrho}' \end{bmatrix} \in \mathbb{R}^{2 \times Q_k}, \quad (7.6)$$

the number of samples in $\hat{\mathbf{V}}_k''$ is Q_k , this number is not the same in each frame k .

Velocity Vector and Height Distributions

A velocity vector $\hat{\mathbf{v}}_q''$ (by omitting superindex k) in the set $\hat{\mathbf{V}}_k'' = \{\hat{\mathbf{v}}_{1,k}'' \in \mathbb{R}^2 \mid \forall q = 1, \dots, Q_k\}$ is assumed to belong to a cloud layer ℓ . The probability of a vector to belong to a cloud layer ℓ is modelled as an independent normal random variable $\hat{\mathbf{v}}_q'' \sim \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$. The function of the multivariate normal distribution is,

$$p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell) = \frac{1}{\sqrt{(2\pi)^2 |\boldsymbol{\Sigma}_\ell|}} \cdot \exp \left\{ -\frac{1}{2} (\hat{\mathbf{v}}_q'' - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\hat{\mathbf{v}}_q'' - \boldsymbol{\mu}_\ell) \right\}. \quad (7.7)$$

In the case when two cloud layers were detected, we propose to infer the probability distribution of velocity vectors' in each cloud layer with this model,

$$p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\Theta}) \propto p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)^{\lambda_q} \cdot p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)^{(1-\lambda_q)}, \quad (7.8)$$

where $\boldsymbol{\Theta} = \{\boldsymbol{\lambda}, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2\}$, and $\lambda_q \in \{0, 1\}$. $\lambda_{q,\ell}$ is defined as convex, considering that a velocity vector may belong to one or the other wind velocity layer, but no to both. Knowing the vectors that belong to the first cloud layer, the vectors that belong to the second cloud layer are also known, $\lambda_{q,2} = 1 - \lambda_{q,1}$. The lower bound of the data log-likelihood is found applying Jensen's inequality [165],

$$\log p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\Theta}) \propto \lambda_{q,1} \cdot \log p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) + \lambda_{q,2} \cdot \log p(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2), \quad (7.9)$$

so that the posterior distribution is a linear combination of the multivariate normal distributions.

The probabilistic model parameters are inferred using a fixed-point variation of the ICM [27]. The algorithm begins by randomly assigning the velocity vectors to a cloud layer, $\lambda_{q,1} \sim \mathcal{U}(0, L - 1)$. The parameters of velocity vector distributions, in Eq. (7.9) that maximize the data log-likelihood are computed in the first step of the algorithm. These same parameters are used to initialize the inference of the parameters of the height distributions in the second step in Eq. (7.12).

In the case of a multivariate normal distribution, the ICM algorithm is iteratively updates parameters. At iteration $t + 1$, the means and covariances are,

$$\boldsymbol{\mu}_\ell^{(t+1)} = \frac{\sum_q \lambda_{q,\ell}^{(t)} \cdot \hat{\mathbf{v}}_q''}{\sum_q \lambda_{q,\ell}^{(t)}}; \quad \boldsymbol{\Sigma}_\ell^{(t+1)} = \frac{\sum_q \lambda_{q,\ell}^{(t)} \cdot \left(\hat{\mathbf{v}}_q'' - \boldsymbol{\mu}_\ell^{(t+1)} \right)^\top \left(\hat{\mathbf{v}}_q'' - \boldsymbol{\mu}_\ell^{(t+1)} \right)}{\sum_q \lambda_{q,\ell}^{(t)}} \quad (7.10)$$

The vectors are re-assigned to a cloud layer at the end of each parameters update, applying the MAP criterion

$$\begin{aligned} \lambda_{q,2}^{(t+1)} &= \underset{\ell}{\operatorname{argmax}} p \left(\hat{\mathbf{v}}_q'' \mid \boldsymbol{\mu}_\ell^{(t+1)}, \boldsymbol{\Sigma}_\ell^{(t+1)} \right) - 1 \\ \lambda_{q,1}^{(t+1)} &= 1 - \lambda_{q,2}^{(t+1)}. \end{aligned} \quad (7.11)$$

After completing the inference of the velocity vectors distribution, it is possible to infer the cloud layer's height using the same method. The velocity vectors in an image were calculated using the WLK method. The algorithm approximates the velocity vector using a set of pixels inside a window. The result is that the velocity vectors do not exactly correspond to a clouds' pixels, which are in motion. Instead, the velocity vectors are assigned to nearby pixels. To identify which layer of clouds ℓ , is the highest and which one is the lowest, the height distribution of the pixels is inferred using the MAP classification of the velocity vectors $\hat{\mathbf{v}}'_{i,j}$ in a image.

The height of the pixels within the cloud are modelled as independently distributed normal random variables $H''_{i,j} \sim \mathcal{N}(\mu_\ell, \sigma_\ell^2)$. The probabilistic model to infer the distribution of heights of each cloud layer in a frame is,

$$\log p \left(H''_{i,j} \mid \boldsymbol{\Theta} \right) \propto \rho_{i,j,1} \cdot \log p \left(H''_{i,j} \mid \mu_1, \sigma_1^2 \right) + \rho_{i,j,2} \cdot \log p \left(H''_{i,j} \mid \mu_2, \sigma_2^2 \right), \quad (7.12)$$

where $\Theta = \{\mathbf{P}, \mu_1, \sigma_1, \mu_2, \sigma_2\}$, and $\rho_{i,j,\ell} \in \{0, 1\}$ is a convex variable so that $\rho_{i,j,2} = 1 - \rho_{i,j,1}$.

The ICM algorithm is also used to infer the parameters of the height distributions model. The $\rho_{i,j,\ell}$ are initialized to the MAP classification of the velocity vectors $\hat{\mathbf{v}}'_{i,j}$ using the parameters that were inferred using all the velocity vectors $\hat{\mathbf{v}}''_q$ in Eq. (7.6),

$$\begin{aligned} \rho_{i,j,2} &= \operatorname{argmax}_{\ell} p(\hat{\mathbf{v}}'_{i,j} \mid \boldsymbol{\mu}_{\ell}, \boldsymbol{\Sigma}_{\ell}) - 1 \\ \rho_{i,j,1} &= 1 - \rho_{i,j,2}. \end{aligned} \quad (7.13)$$

The parameters of the height distributions are updated using the formulas in Eq. (7.10). The algorithm eventually converges so that the pixels in the frame are segmented where a cloud appears. The segmentation is performed according to the MAP classification of height distributions,

$$\begin{aligned} \rho_{i,j,2}^{(t+1)} &= \operatorname{argmax}_{\ell} p(H''_{i,j} \mid \mu_{\ell}^{(t+1)}, \sigma_{\ell}^{2(t+1)}) - 1 \\ \rho_{i,j,1}^{(t+1)} &= 1 - \rho_{i,j,2}^{(t+1)}. \end{aligned} \quad (7.14)$$

In order to find the height of a given cloud layer, the heights are averaged with this formula,

$$\bar{h}_{\ell} = \frac{\sum_{i,j} \rho_{i,j,\ell} \cdot H''_{i,j} \cdot \mathbb{I}(b_{i,j} = 1)}{\sum_{i,j} \rho_{i,j,\ell} \cdot \mathbb{I}(b_{i,j} = 1)}. \quad (7.15)$$

The wind velocity fields are organized into upper and lower layers by average height \bar{h}_{ℓ} . In this way, each detected wind velocity field has a distribution of velocity vectors, and another distribution of heights.

Sampling

In order to reduce the computational burden of the algorithm, a subset of the velocity vectors is selected according to the estimated probability distributions of the vectors.

At each layer ℓ , we define the importance weights $w_{q,\ell,k}$ as

$$w_{q,\ell,k} \triangleq p(\hat{\mathbf{v}}''_{q,k} | \boldsymbol{\theta}_\ell), \quad w_{q,\ell,k} \in \mathbb{R}^+. \quad (7.16)$$

The weights are normalized to have the characteristics of a probability mass function such as $\sum_{q=1}^{Q_k} \hat{w}_{q,\ell,k} = 1$.

The Cumulative Probability Function (CDF) is computed as

$$\tilde{w}_{q,\ell,k} = \left\{ \sum_{m=1}^q \hat{w}_{m,\ell,k} \right\}_{q=1}^{Q_k}. \quad (7.17)$$

In order to select samples for each distribution $p(\hat{\mathbf{v}}''_{q,k} | \boldsymbol{\theta}_\ell)$, N^*/L samples are drawn from a uniform distribution,

$$z_{p,\ell,k} \sim \mathcal{U}(0, 1), \quad p = 1, \dots, \frac{N^*}{L}, \quad (7.18)$$

For each value $z_{p,\ell,k}$, a sample is selected with the criterion

$$I_{p,\ell,k} = \operatorname{argmin} \quad | \tilde{\mathbf{w}}_{\ell,k} - z_{p,\ell,k} |, \quad \forall p = 1, \dots, \frac{N^*}{L}. \quad (7.19)$$

The selected vectors are the ones whose CDF is closest to the values of the uniform samples $z_{p,\ell,k}$,

$$\hat{\mathbf{V}}_{\ell,k}^* \triangleq \hat{\mathbf{V}}''_k[I_{p,\ell,k}], \quad \forall p = 1, \dots, \frac{N^*}{L}. \quad (7.20)$$

The subset of selected velocity vectors in frame k for the cloud layer ℓ is $\hat{\mathbf{V}}_{\ell,k}^* = \{(\hat{u}, \hat{v})_{p,\ell,k}^* \in \mathbb{R}^2 \mid \forall p = 1, \dots, N^*/L\}$, the subset of Euclidean coordinate pairs of those selected vectors is $\mathbf{X}_{\ell,k}^* = \{(x, y)_{p,\ell,k}^* \in \mathbb{N}^2 \mid \forall p = 1, \dots, N^*/L\}$.

Assuming that the prior is uniform, the posterior probabilities are,

$$z_{p,\ell,k}^* \triangleq \frac{p(\hat{\mathbf{v}}_{p,k}^* | \boldsymbol{\theta}_\ell)}{\sum_{\ell=1}^L p(\hat{\mathbf{v}}_{p,k}^* | \boldsymbol{\theta}_\ell)}. \quad (7.21)$$

The sampling is repeated for as many cloud layers detected. All selected subsets of vectors, coordinate pairs, and weights form the dataset that is used to approximate the wind velocity field,

$$\mathbf{X}_k^* = \begin{bmatrix} x_{1,k}^* & y_{1,k}^* \\ \vdots & \vdots \\ x_{N^*,k}^* & y_{N^*,k}^* \end{bmatrix}, \quad \hat{\mathbf{V}}_k^* = \begin{bmatrix} \hat{u}_{1,k}^* & \hat{v}_{1,k}^* \\ \vdots & \vdots \\ \hat{u}_{N^*,k}^* & \hat{v}_{N^*,k}^* \end{bmatrix}, \quad \mathbf{Z}_k^* = \begin{bmatrix} z_{1,1,k}^* & \cdots & z_{1,L,k}^* \\ \vdots & \ddots & \vdots \\ z_{N^*,1,k}^* & \cdots & z_{N^*,L,k}^* \end{bmatrix}, \quad (7.22)$$

where $Q_k \gg N^*$.

7.3 Flow Visualization

The atmosphere is a system where the dynamics are continuously changing [171]. This fact implies that a wind velocity field exists, but we can only visualize it where clouds are present. From ground level to the Tropopause, the wind flow can have multiple layers with different velocities in each one of the layers. The wind flow may also be convective, however, for the sake of simplicity, we assume that the multi-layer flow which is observed in IR images is a multi-layer laminar flow. This analysis neither considers the z-component in the motion of a cloud (which is not observable) nor the possible inter-crossing of cloud layers.

7.3.1 Wind Velocity Field Estimation

Three methods were implemented to estimate the extrapolation function and compare their performances. The first method uses a Weighted Support Vector Machine for regression (ε -WSVM) for each one of the velocity components. The second method is a ε -MT-WSVM that estimates both velocity components. The third is an innovation which uses a ε -MT-WSVM with flow constraints (ε -MT-WSVM-FC) to estimate both velocity components. The flow constraints are used to force the extrapolated wind

flow to have zero divergence or curl, so it can be assumed that, in the approximated wind flow, streamlines are equivalent to the cloud pathlines.

The regression problem can be formulated as the optimization of a function with the form,

$$f(\mathbf{x}_i) = \mathbf{w}^\top \varphi(\mathbf{x}_i) + b, \quad \forall i = 1, \dots, N, \quad \mathbf{w}, \mathbf{x}_i \in \mathbb{R}^D, \quad b \in \mathbb{R}. \quad (7.23)$$

where $\mathbf{x}_i \triangleq \mathbf{x}_{p,k}^*$ in our problem, N is the number of training samples, D is the number of dimensions (i.e., 2 velocity components), and $\varphi(\cdot)$ is a transformation into a higher dimensional (possibly infinite) Hilbert space \mathcal{H} endowed with a dot product $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$. A function $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ is a dot product if it is a bivariate positive semi-definite function that maps $\mathbf{x}_i, \mathbf{x}_j$ into \mathbb{R} , commonly called a Mercer's kernel or simply a kernel function.

7.3.2 Support Vector Machine for Regression

Assuming $\boldsymbol{\omega}_i = \{u_i, v_i\} \triangleq \hat{\mathbf{v}}_{p,k}^*$, the regression problem in a ε -SVM is formulated with an ε -insensitive loss function, which penalizes the errors $|\varepsilon| > 0$ [82] for each one of the components in \mathbf{v}_p and for each cloud layer ℓ as

$$|u_i - f(\mathbf{x}_i)|_\varepsilon = \max[0, |u_i - f(\mathbf{x}_i)| - \varepsilon], \quad \forall i = 1, \dots, N, \quad u_i, \varepsilon \in \mathbb{R}, \quad (7.24)$$

and identically for v_i . The ε -insensitive loss function can be seen as a tube of radius ε adjusted around the regression hyper-plane via the Support Vectors (SV) [289].

The samples are weighted by their probability of belonging to wind velocity field ℓ [86],

$$\begin{aligned} z_i &\triangleq z_{i,k}^*, \quad z_i \in \mathbb{R}^{\leq 1}. \\ c_i &= z_i \cdot \frac{\mathcal{C}}{N} \end{aligned} \quad (7.25)$$

The L2-norm and ε -loss function is applied to the model weights,

$$\min_{\mathbf{w}, b, \xi, \xi^*} \quad \frac{1}{2} \|\mathbf{w}\|^2 + \frac{\mathcal{C}}{N} \sum_{i=1}^N z_i (\xi_i + \xi_i^*) \quad (7.26)$$

$$\text{s.t.} \quad \begin{cases} u_i - \mathbf{w}^\top \varphi(\mathbf{x}_i) - b & \leq \varepsilon + \xi_i \\ \mathbf{w}^\top \varphi(\mathbf{x}_i) + b - u_i & \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq 0 \end{cases} \quad i = 1, \dots, N, \quad (7.27)$$

and identically for v_i . The slack variables ξ_i were introduced to relax the constraints of the optimization problem and to deal with unfeasible optimization problems [63]. The primal objective function aims to find the trade off between the regularization term ε , the allowed errors or slack variables ξ_i and ξ_i^* , and the complexity of the model c_i per weighted sample.

The proposed kernel functions in this analysis are,

$$\begin{aligned} \mathcal{K}_L(\mathbf{x}_i, \mathbf{x}_j) &= \mathbf{x}_i^\top \mathbf{x}_j, \\ \mathcal{K}_{RBF}(\mathbf{x}_i, \mathbf{x}_j) &= \exp(-\gamma \cdot \|\mathbf{x}_i - \mathbf{x}_j\|^2), \\ \mathcal{K}_{\mathcal{P}^n}(\mathbf{x}_i, \mathbf{x}_j) &= (\gamma \cdot \mathbf{x}_i^\top \mathbf{x}_j + \beta)^n, \end{aligned} \quad (7.28)$$

where $\gamma, \beta \in \mathbb{R}$, and $n \in \mathbb{N}$ are the kernel hyperparameters that need cross-validation. [289]. These kernel functions are referred to as linear (L), Radial Basis Function (RBF) or square exponential, and Polynomial of order n (\mathcal{P}^n) respectively [297].

In order to optimize the constrained problem in functional (7.26) and constraints (7.27) a Langrangian functional is constructed with the functional and the set of constraints through a dual set of new variables [305], which leads to a solvable

Quadratic Programming problem (QP). The Lagrangian functional is

$$\begin{aligned}
 \mathcal{L}(\mathbf{w}, b, \alpha, \alpha^*, \beta, \xi, \xi^*, \varepsilon, \eta, \eta^*) &= \\
 &= \frac{1}{2} \|\mathbf{w}\|^2 + c_i \sum_{i=1}^N (\xi_i + \xi_i^*) \dots \\
 &- \sum_{i=1}^N (\eta_i \xi_i + \eta_i^* \xi_i^*) - \sum_{i=1}^N \alpha_i (\varepsilon + \xi_i - u_i + \mathbf{w}^\top \varphi(\mathbf{x}_i) + b) \dots \\
 &- \sum_{i=1}^N \alpha_i^* (\varepsilon + \xi_i^* + u_i - \mathbf{w}^\top \varphi(\mathbf{x}_i) - b), \quad \forall i = 1, \dots, N, \quad \eta_i, \eta_i^* \in \mathbb{R}.
 \end{aligned} \tag{7.29}$$

The derivatives of the primal variables $\mathbf{w}, \varepsilon, \xi_i, \xi_i^*$ yield to the following set of equations, which is a case of Karush-Kuhn-Tucker (KKT) conditions,

$$\begin{aligned}
 \mathbf{w}^\top &= \sum_{i=1}^N (\alpha_i^* - \alpha_i) \varphi(\mathbf{x}_i), \\
 0 &= \sum_{i=1}^N (\alpha_i - \alpha_i^*), \\
 0 &= c_i - \alpha_i - \eta_i, \\
 0 &= c_i - \alpha_i^* - \eta_i^*.
 \end{aligned} \tag{7.30}$$

These conditions, together with the complimentary KKT conditions (which force the product of dual parameters α_i, α_i^* with the constraints to be zero) leads to the following dual functional by substitution on the Lagrangian:

$$\begin{aligned}
 \min_{\alpha, \alpha^*} \quad & \frac{1}{2} \cdot (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*)^\top \mathbf{K} (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) + \sum_{i=1}^N (\alpha_i - \alpha_i^*) u_i + \varepsilon \cdot \mathbf{1}^\top (\boldsymbol{\alpha} + \boldsymbol{\alpha}^*) \\
 \text{s.t.} \quad & \begin{cases} \mathbf{1}^\top (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) = 0 \\ 0 \leq \alpha_i, \alpha_i^* \leq c_i \end{cases} \quad \forall i = 1, \dots, N.
 \end{aligned} \tag{7.31}$$

where $\mathbf{1}_{1 \times N} = [1, \dots, 1]^\top$, and matrix \mathbf{K} is a Gram matrix of dot product such that $\mathbf{K}_{i,j} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$. The minimal of the primal function is equivalent to the saddle point on the Lagrangian formulation. The approximated function is,

$$f(\mathbf{x}) = \sum_{i=1}^N (\alpha_i - \alpha_i^*) \cdot \mathcal{K}(\mathbf{x}_i, \mathbf{x}_i) + b, \tag{7.32}$$

where b is obtained from the complimentary KKT conditions.

7.3.3 Multi-Task Weighted Support Vector Machine

When the wind velocity field function is approximated by ε -MT-SVM, the primal regression can be formulated as

$$\boldsymbol{\omega}_i = \mathbf{W}^\top \varphi(\mathbf{x}_i) + \mathbf{b}, \quad (7.33)$$

where each one of the column vectors of primal parameter matrix \mathbf{W} approximates one of the velocities in vector $\boldsymbol{\omega}_i$. Primal parameters are a function of the dual parameters as well, but the dual parameters $\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_i^*$ are vectors in a 2-dimensional multi-output problem.

Since independent variables are represented in vectors $\boldsymbol{\omega}_i$, the training set is defined in a vector $\tilde{\boldsymbol{\omega}}_{1 \times 2P}$, and so are the dual parameters $\tilde{\boldsymbol{\alpha}}_{1 \times 2P}$ and $\tilde{\boldsymbol{\alpha}}_{1 \times 2P}^*$ for notation simplicity.

The gram matrix of dot products between input patterns $\varphi(\mathbf{x}_i)$ can be interpreted as the covariance matrix between variables $\boldsymbol{\omega}_i$. Indeed

$$\begin{aligned} \mathbb{E}((\boldsymbol{\omega}_i - \mathbf{b}^\top)(\boldsymbol{\omega}_j - \mathbf{b})) &= \mathbb{E}(\mathbf{W}^\top \varphi(\mathbf{x}_i) \varphi(\mathbf{x}_j)^\top \mathbf{W}) \\ &= \begin{pmatrix} \varphi(\mathbf{x}_i)^\top \boldsymbol{\Sigma}_{11} \varphi(\mathbf{x}_j) & \varphi(\mathbf{x}_i)^\top \boldsymbol{\Sigma}_{12} \varphi(\mathbf{x}_j) \\ \varphi(\mathbf{x}_i)^\top \boldsymbol{\Sigma}_{21} \varphi(\mathbf{x}_j) & \varphi(\mathbf{x}_i)^\top \boldsymbol{\Sigma}_{22} \varphi(\mathbf{x}_j) \end{pmatrix}, \end{aligned} \quad (7.34)$$

where the 2×2 covariance $\mathbb{E}(\mathbf{W}\mathbf{W}^\top)$ is interpreted as a model for the dependencies between elements in $\boldsymbol{\omega}_i$, i.e.

$$\mathbb{E}(\mathbf{W}\mathbf{W}^\top) = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}. \quad (7.35)$$

If we consider that both vertical and horizontal velocities are independent, then $\boldsymbol{\Sigma}_{12} = \boldsymbol{\Sigma}_{21} = \mathbf{0}$. If we assume further that $\boldsymbol{\Sigma}_{11} = \boldsymbol{\Sigma}_{22} = \mathbf{I}$ for simplicity, which, in

turn leads to

$$\mathbb{E}((\boldsymbol{\omega}_i - \mathbf{b}^\top)(\boldsymbol{\omega}_j - \mathbf{b})) = \begin{pmatrix} \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) & 0 \\ 0 & \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \end{pmatrix}. \quad (7.36)$$

The Gram matrix $\tilde{\mathbf{K}}_{2P \times 2P}$ in the ε -MT-SVM formulation for independent components is,

$$\tilde{\mathbf{K}} = \begin{pmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0} & \mathbf{K} \end{pmatrix}. \quad (7.37)$$

The full kernel matrix in a ε -MT-WSVM is computationally expensive, and it is not implemented in this chapter.

The extension of weights in the ε -MT-WSVM requires weighting each sample in each output [131],

$$\tilde{z}_i = [z \dots z_N \ z_1 \dots z_N]^\top, \quad \sum_{i=1}^{2N} \tilde{z}_i = 2, \quad \tilde{z}_i \in \mathbb{R}^{\leq 1}. \quad (7.38)$$

The dual formulation of the QP problem for the ε -MT-WSVM is,

$$\begin{aligned} \min_{\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\alpha}}^*} \quad & \frac{1}{2} \cdot (\tilde{\boldsymbol{\alpha}} - \tilde{\boldsymbol{\alpha}}^*)^\top \tilde{\mathbf{K}} (\tilde{\boldsymbol{\alpha}} - \tilde{\boldsymbol{\alpha}}^*) + \tilde{\boldsymbol{\omega}}^\top (\tilde{\boldsymbol{\alpha}} - \tilde{\boldsymbol{\alpha}}^*) + \varepsilon \cdot \mathbf{1}^\top (\tilde{\boldsymbol{\alpha}} + \tilde{\boldsymbol{\alpha}}^*) \\ \text{s.t.} \quad & \begin{cases} \mathbf{1}^\top (\tilde{\boldsymbol{\alpha}} - \tilde{\boldsymbol{\alpha}}^*) = 0 \\ \mathbf{0} \leq \tilde{\alpha}_i, \tilde{\alpha}_i^* \leq \tilde{c}_i \end{cases} \quad \forall i = 1, \dots, 2N, \end{aligned} \quad (7.39)$$

where the extended weighted complexity is $\tilde{c}_i = \tilde{z}_i/2N$.

7.3.4 Multi-Task Weighted Support Vector Machine with Flow Constraints

Assuming that the analyzed air parcel is sufficiently small so that the flow can be considered approximately incompressible and irrotational, a new set of flow constraints

are added to the original set of constraints with the purpose of visualizing the wind velocity field to force the divergence and the vorticity to zero:

$$\text{s.t.} \begin{cases} \left(\tilde{\omega}_{\ell,k}^\top \Delta_{x,y} \dot{\mathbf{V}} \right) \cdot \left(\tilde{\omega}_{\ell,k}^\top \Delta_{x,y} \dot{\mathbf{V}} \right)^\top = 0 \\ \left(\tilde{\omega}_{\ell,k}^\top \Delta_{x,y} \dot{\mathbf{D}} \right) \cdot \left(\tilde{\omega}_{\ell,k}^\top \Delta_{x,y} \dot{\mathbf{D}} \right)^\top = 0, \end{cases} \quad (7.40)$$

where the matrices of this expression are defined in Eq. (7.41) Eq. (7.42) and Eq. (7.43). To compute the vorticity and divergence, the differentiation of the velocity field along the x-axis, the and y-axis is implemented using operator

$$\Delta_{x,y} = \begin{bmatrix} \Delta_x & \mathbf{0} \\ \mathbf{0} & \Delta_y \end{bmatrix}_{2N \times 2N}, \quad (7.41)$$

where the differential operators Δ_x and Δ_y are defined as,

$$\Delta_x = \begin{bmatrix} -1 & 0 & \dots & 0 \\ 1 & -1 & \ddots & \vdots \\ 0 & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & -1 \\ 0 & \dots & 0 & 1 \end{bmatrix}_{N \times N}; \quad \Delta_y = \begin{bmatrix} -1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & -1 \\ 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix}_{N \times N}. \quad (7.42)$$

The operators of the velocity field's vorticity and divergence are respectively,

$$\dot{\mathbf{V}} = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \\ 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix}_{2N \times N}; \quad \dot{\mathbf{D}} = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \\ -1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & -1 \end{bmatrix}_{2N \times N}. \quad (7.43)$$

The wind velocity field is extrapolated to the entire frame using the inferred parameters in frame k

$$\tilde{\omega}_{\ell,k} = (\tilde{\alpha}_{\ell,k} - \tilde{\alpha}_{\ell,k}^*) \cdot \mathcal{K}(\mathbf{X}_k^*, \mathbf{X}) + \mathbf{b}_{\ell,k}, \quad \tilde{\omega}_{\ell,k} \in \mathbb{R}^{2 \times M \cdot N} \quad (7.44)$$

the approximate wind velocity field $\tilde{\omega}_{\ell,k} = (\tilde{\mathbf{u}}_{\ell,k}, \tilde{\mathbf{v}}_{\ell,k})$ (which is extended) is reshaped to a matrix form $\hat{\mathbf{W}} \in \mathbb{R}^{2 \times M \times N}$. The velocity components of the approximated wind velocity field are $\hat{\mathbf{U}}_{\ell,k} \triangleq \hat{\mathbf{W}}_{1,\ell,k}$, and $\hat{\mathbf{V}}_{\ell,k} \triangleq \hat{\mathbf{W}}_{2,\ell,k}$, where $\hat{\mathbf{U}}_{\ell,k}, \hat{\mathbf{V}}_{\ell,k} \in \mathbb{R}^{M \times N}$, and M, N are the original dimensions of an image (i.e., frame).

To compute the flow constraints, the velocity field has to be extrapolated to the whole frame using Eq. (7.44). The constraints in Eq. (7.40) are added to the constraints in the dual formulation of the ε -MT-SVM in Eq. (7.39).

7.4 Wind Velocity Field Dynamics Estimation

To estimate the wind velocity field dynamics, velocity vectors are approximated using the WLK method. The velocity vectors are weighted by the posterior probabilities of the cloud layers in the image, and transformed to the cloud layer plane. The velocity vectors are segmented and sampled to reduce the noise and the computation burden. The parameters of the ε -MT-WSVM-FC are cross-validated, and the model is trained to estimate the wind velocity field of the detected cloud layers. If an optimal set of parameters is available, it is possible to proceed with the training of ε -MT-WSVM-FM without performing the cross-validation. After training the ε -MT-WSVM-FC, the testing is performed to extrapolate the wind velocity field to the whole image. The streamlines are computed using Eq. (7.45) to visualize the trajectory of a cloud. The potential lines are not shown in the Figures 7.4a to 7.5f, but they are computed with Eq. (7.46).

The physical process of cloud formation and evolution over time is part of atmospheric thermodynamics and may have divergence and vorticity [186]. The air parcel in one frame is very small compared to the whole volume of air contained in the atmosphere. Within this frame we consider it feasible that there is no vorticity or divergence, and the approximated streamlines are equivalent to the pathlines. Hence-

forth, the obtained results are a numerical approximation of the actual atmospheric air parcel flow.

When we assume that a flow does not have divergence and vorticity, the stream and velocity potential functions are orthogonal, and we can apply Cauchy-Riemann equations to calculate their rates of change [120]. For the stream function we determine $d\phi = udy - vdx$ using samples of functions. The trapezoidal rule of numerical analysis is applied to solve the definite integrals [81]. The values of a streamline are,

$$\Phi_\ell = \frac{h_\ell}{2} \left[\left\{ \sum_{m=1}^i \hat{\mathbf{u}}_{m,\ell} \odot \Delta \mathbf{y}_{m,\ell} \right\}_{i=1}^M - \left\{ \sum_{m=1}^j \hat{\mathbf{v}}_{m,\ell} \odot \Delta \mathbf{x}_{m,\ell} \right\}_{j=1}^N \right], \quad (7.45)$$

where \odot denotes the element-wise matrix multiplication. This is the sum of element-wise products between each velocity component and its opposite differential increments.

The total change in the potential function is $d\psi = udx + vdy$, so we can determine the potential in each pixel $\mathbf{x} = \{x, y\}$ as,

$$\Psi_\ell = \frac{h_\ell}{2} \left[\left\{ \sum_{m=1}^j \hat{\mathbf{u}}_{m,\ell} \odot \Delta \mathbf{x}_{m,\ell} \right\}_{j=1}^N + \left\{ \sum_{m=1}^i \hat{\mathbf{v}}_{m,\ell} \odot \Delta \mathbf{y}_{m,\ell} \right\}_{i=1}^M \right]. \quad (7.46)$$

the sum of each element-wise product between the velocity components, and their differential increment.

7.5 Experiments

7.5.1 Training Data Construction

To create a data set for validation purposes we selected 21 consecutive images, per day, on six different days. The images were selected due to the presence of different types of clouds distributed across different heights. The selected images were captured

during different seasons and different times of the day. The images from three of the six days show a layer of cirrustratus in winter in the morning, altostratus in spring in the afternoon and stratocumulus in summer in the afternoon. The other three days show two layers of altostratus and stratocumulus in winter at noon, cirrustratus and altocumulus in spring in the afternoon, and cirruscumulus and cumulus in summer in the morning.

The proposed algorithm only requires the validation dataset to be labelled as it is an unsupervised online ML algorithm. The validation dataset is used to find the optimal parameters of the algorithm which segments and subsamples the velocity vectors from the last 6 consecutive frames.

The targets of the ε -WSVM are the velocity vectors computed using the WLK method. The machine is cross-validated and trained for each frame using the selected velocity vectors of the last 6 frames. The testing error is that of the ε -WSVM approximating the WLK velocity vectors.

The average height, velocity magnitude and angle of a cloud was manually calculated for each cloud layer in each sequence of images to use them as ground truth. To do this, the pathline intercepting the Sun was manually segmented. The distance that a cloud has moved in the pathline was calculated in each frame. The height of a cloud layer was measured by segmenting the clouds along the sequence of images. The calculated height, velocity magnitude and angle of each cloud layer was averaged across the validation sequences of images.

The wind velocity is a relative measure of the actual velocity in a frame. The algorithm does not need the actual wind velocity. The algorithm requires the height of the clouds to define the space of camera's FOV. The velocity vectors are transformed from pixels per frame to meters per second. Each pixel is projected to its actual dimension in the space defined by the camera's FOV. To know the distance that a

cloud will travel in a given time to occlude the Sun, the magnitude of the projected velocity vectors in the space defined by the camera's FOV is calculated. The relative measure of the wind velocity vectors (in pixels per frame) and the height of the clouds (in meters) are connected together in the frames by the geometric transformation. For this reason, the wind velocity that it is required is not the actual wind velocity but the relative wind velocity measured in the frame.

7.5.2 Velocity Vectors Calculation, Segmentation and Subsampling Parameters Validation

The parameters δ in Eq. (7.2), τ in Eq. (7.5), ϱ in Eq. (7.6) of the velocity vectors selection algorithm were validated using the six sequences of images described above. The velocity estimator was ε -WSVM with a linear kernel. The parameters of ε -WSVM, ε and \mathcal{C} , were cross-validated in the same process. The flow velocity constraints were not applied in the validation.

The average of the approximated wind velocity field height, magnitude and angle was computed, and the Mean Absolute Percentage Error (MAPE) was calculated between the measured and the averaged approximation in each frame. The MAPE was averaged and differentiated across consecutive frames. The combination of parameters that had less averaged MAPE plus total difference of MAPE between consecutive frames was selected. This added difference of MAPE was used to account for the accuracy of the selected model parameters, but also the stability of the models. This stability is optimal if good results are consistently obtained for each one of the images in the same sequence.

The optimal amplitude δ of the velocity vector in Eq. (7.2), was found to be $\delta = 2.29$. The optimal threshold τ in the segmentation of the velocity vector in Eq. (7.5), was found to be $\tau = 0.95$. The optimal number of velocity vectors from ϱ last

frames to form the dataset in Eq. (7.6), was found to be $\varrho = 6$. The optimal number of selected samples N^* by sampling algorithm in Eq. (7.22), was found that $N^* = 200$ samples are sufficient to perform a robust extrapolation of the wind velocity field to the entire frame.

The velocity vectors that were segmented in a frame with two layers of clouds are shown in Figure 7.1. The velocity vectors from the last 6 frames after applying the segmentation are shown in the upper row of Figure 7.2. In this figure, the colors represent the likelihoods conditional to the upper cloud layer (left), and lower cloud layer (right). The sampled velocity vectors in a frame with two layers of clouds are shown in the bottom row. Figure 7.3 shows the selected velocities in the bottom row of Figure 7.2 in their corresponding coordinates. In this figure, the colors represent the posterior probabilities conditional to the upper cloud layer (left), and lower cloud layer (right).

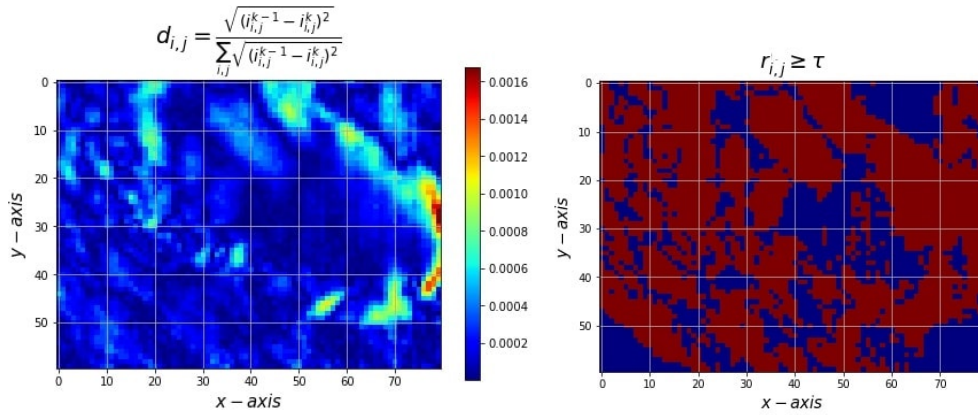


Figure 7.1: The figures illustrate the proposed method to discriminate between points which probably show a moving cloud air parcel and those that probably show a pixel without movement. The left graph shows the computation of the squared difference between two consecutive frames. The right graph shows in blue those pixels which are considered not moving as their squared difference value is less than a previously validated threshold τ .

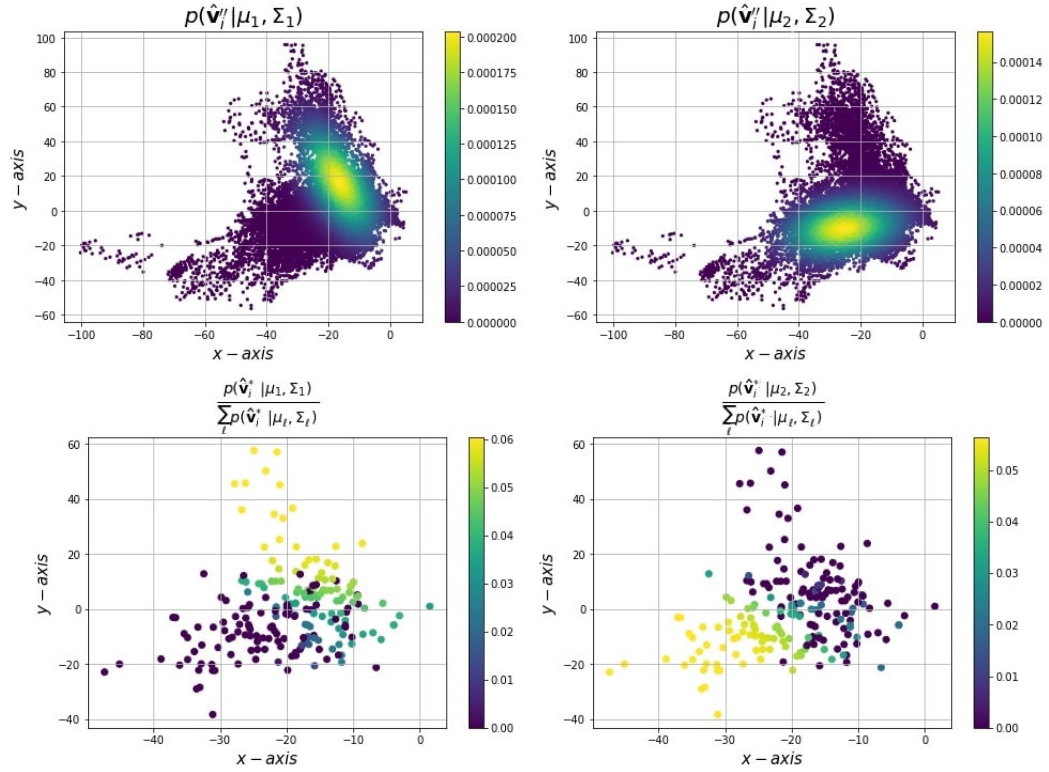


Figure 7.2: Probability distribution of the velocity vectors and the subsampling implemented to decrease the computational cost. The upper row shows the distribution of the measured velocities represented in \mathbb{R}^2 . The colormap represents the likelihood of the velocities conditional to a point belonging to the lower layer of clouds (left) and upper layer (right). The lower row shows the downsampled set of vectors (and their posterior probabilities) after applying the downsampling methodology.

7.5.3 MT-WSVM-FC Parameters Validation

After optimal values of δ , τ and ϱ have been chosen, the parameters of the proposed ε -MT-WSVM-FC are cross-validated using the validation data or an online ML approach. This means that the experiments with the ε -MT-WSVM-FC have two different setups. In the first, the parameters are cross-validated in each testing frame. In the second, the parameters are fixed to the optimal values obtained in a previous cross-validation using the validation data. This is done to check for the validity of the previously

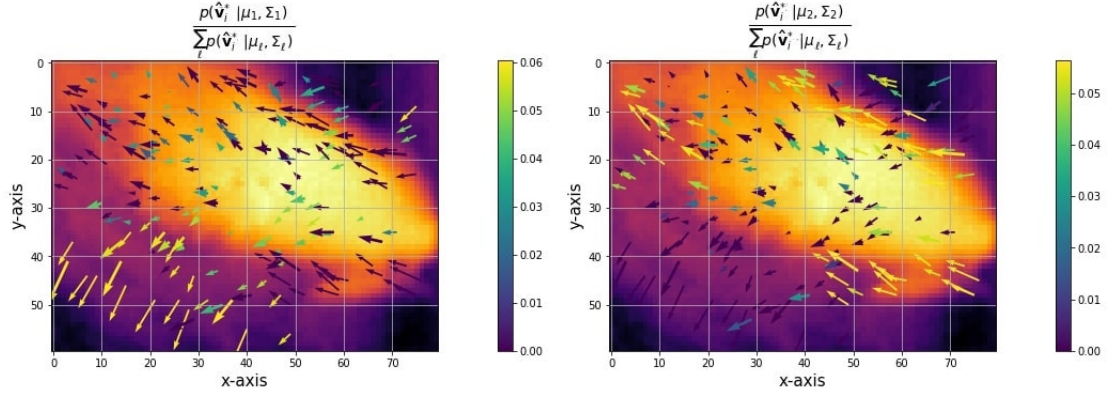


Figure 7.3: Velocities selected in Figure 7.2 in their corresponding position. The left image vectors show a color intensity corresponding to their posterior probabilities conditional to the upper cloud layer and the right image to the lower cloud layer. The left image shows in yellow the points clearly belonging to the upper layer, while right image shows in yellow the points that are clearly of the lower layer.

obtained parameters, which would significantly reduce the velocity field estimation computational burden.

The objective of the constraints is that the divergence and vorticity of approximated velocity field are zero in the clouds' plane. The velocity fields shown in Figure 7.4b to 7.5f have divergence and vorticity after the field is projected to the camera plane. This is caused by the non-linear geometric transformation applied to the velocity vectors.

7.5.4 Wind Velocity Field Estimation with New Data

Unlike the ε -MT-WSVM-FC, the experiments with the ε -WSVM and ε -MT-WSVM use only the first setup. These models are validated and trained for each testing frame. The results are compared with a Gaussian Process for Regression (GPR) for each one of the velocity components [274], a Multi-Task Ridge Regression (MT-RR) with independent components (which is a special case of Tikhonov's regularization [330]) and a Multi-Task Gaussian Process for Regression (MT-GPR) with correlation

between velocity components [33].

The testing data is composed of sequences of 28 images from 10 different days. The sequences are from different seasons and different times of the day. Five days have one velocity field layer and the other five have two layers. 75% of this data is chosen for the online training and validation of the models. The rest of data is used for testing. The testing set is from $k + \varrho$ frames ahead of the training set from frame k . The number of frames ahead is equal to the lag of the velocity vectors in the data $\varrho = 6$. The methodologies implemented in the validation are the standard grid search and 3-fold cross-validation. The parameters cross-validated in the ε -WSVM, ε -MT-WSVM and ε -MT-WSVM-FC are \mathcal{C} and ε . The MT-RR requires the cross-validation of the regularization parameter. In the GPR and MT-GPR the parameters are found by numerical gradient, optimizing the MLL. The kernel functions are: linear, RBF, polynomial of order two (\mathcal{P}^2), and polynomial of order three (\mathcal{P}^3). The optimal parameters for the ε -MT-WSVM-FC are displayed in Table 7.1.

Table 7.1: This table shows the optimal sets of parameters obtained cross-validating and training the ε -MO-WSVM-FC in each training image, the results cross-validating the parameters and training the ε -MO-WSVM-FC in each testing image, and the testing results training the ε -MO-WSVM-FC in each testing image using the optimal sets of parameters previously cross-validated in the training data.

ε -MO-WSVM-FC													
$\mathcal{K}(\mathbf{x}, \mathbf{x}^*)$	Optimal Parameters				Online Parameters Cross-Validation				Fixed Optimal Parameters				Time [s]
	\mathcal{C}	ε	γ	β	MAE	WMAE	$\nabla \cdot \vec{V}$	$\nabla \times \vec{V}$	MAE	WMAE	$\nabla \cdot \vec{V}$	$\nabla \times \vec{V}$	
\mathcal{K}_L	38.50	0.19			14.22	13.36	0.0	0.0	14.24	13.35	0.0	0.0	58.54
\mathcal{K}_{RBF}	38.52	0.35	13.92		14.55	13.53	30.96	30.86	14.12	13.05	136.97	138.80	114.71
$\mathcal{K}_{\mathcal{P}^2}$	39.72	0.24	3.78	44.8	14.36	13.48	77.22	70.85	14.48	13.59	30.44	30.77	130.92
$\mathcal{K}_{\mathcal{P}^3}$	12.88	0.22	5.61	8.34	15.34	14.34	$1.74 \cdot 10^4$	$1.66 \cdot 10^4$	45.03	44.48	$2.19 \cdot 10^6$	$1.97 \cdot 10^6$	145.50

The criteria for selecting the most suitable model and kernel function for our application is a trade-off between divergence and vorticity, Weighted Mean Absolute Error (WMAE), and the computing time. The values of these metrics are summarized for the models without constraints in Table 7.2, and for the ε -MT-WSVM-FC in Table 7.1. The experiment of the ε -MT-WSVM without flow constraints using a

\mathcal{P}^3 kernel is shown in Figure 7.4a, and that same experiment implemented with the ε -MT-WSVM-FC using a linear kernel is in Figure 7.4b. In sequences of images in which two layers of clouds were detected, the experiments of the ε -MT-WSVM-FC using a linear kernel to approximate wind velocity field are shown in the Figure 7.5a to 7.5f.

Table 7.2: The table above shows the testing results of the different kernel learning methods without flow constraints. The first method is the ε -WSVM-FC, the second is ε -MT-WSVM, the third is the GPR, the fourth is the MT-RR and the fifth is the MT-GPR. The wind velocity fields approximated by all the methods have low MAE and WMAE with high divergence and vorticity. The fastest methods are GPR and MT-GPR as the optimization of the parameters is performed via numerical gradient.

ε -WSVM					
$\mathcal{K}(\mathbf{x}, \mathbf{x}^*)$	MAE	WMAE	$\nabla \cdot \vec{V}$	$\nabla \times \vec{V}$	Time [s]
\mathcal{K}_L	13.37	12.55	$1.69 \cdot 10^3$	$2.17 \cdot 10^3$	90.01
\mathcal{K}_{RBF}	13.39	12.61	$6.25 \cdot 10^3$	$6.40 \cdot 10^3$	365.72
$\mathcal{K}_{\mathcal{P}^2}$	14.06	13.22	$1.20 \cdot 10^4$	$1.19 \cdot 10^4$	2413.79
$\mathcal{K}_{\mathcal{P}^3}$	14.90	13.95	$8.98 \cdot 10^4$	$9.48 \cdot 10^4$	3468.75
ε -MT-WSVM					
\mathcal{K}_L	13.27	12.49	$1.30 \cdot 10^3$	$1.35 \cdot 10^3$	162.70
\mathcal{K}_{RBF}	14.00	13.13	$1.21 \cdot 10^3$	$1.22 \cdot 10^3$	560.54
$\mathcal{K}_{\mathcal{P}^2}$	14.25	13.53	$1.43 \cdot 10^4$	$1.71 \cdot 10^4$	5635.31
$\mathcal{K}_{\mathcal{P}^3}$	19.29	18.12	$8.89 \cdot 10^5$	$8.92 \cdot 10^5$	7284.54
GPR					
\mathcal{K}_L	12.56	12.56	$2.62 \cdot 10^3$	$3.27 \cdot 10^3$	6.50
\mathcal{K}_{RBF}	12.89	12.88	$1.24 \cdot 10^4$	$1.27 \cdot 10^4$	6.43
$\mathcal{K}_{\mathcal{P}^2}$	12.52	12.50	$7.27 \cdot 10^3$	$9.02 \cdot 10^3$	6.44
$\mathcal{K}_{\mathcal{P}^3}$	12.67	12.68	$2.72 \cdot 10^4$	$3.11 \cdot 10^4$	6.42
MT-RR					
\mathcal{K}_L	12.62	12.58	$2.62 \cdot 10^3$	$3.31 \cdot 10^3$	6.71
\mathcal{K}_{RBF}	13.43	13.35	$3.95 \cdot 10^3$	$7.24 \cdot 10^3$	11.80
$\mathcal{K}_{\mathcal{P}^2}$	12.55	12.55	$1.53 \cdot 10^4$	$1.15 \cdot 10^4$	29.76
$\mathcal{K}_{\mathcal{P}^3}$	12.70	12.64	$3.17 \cdot 10^5$	$2.20 \cdot 10^5$	41.16
MT-GPR					
\mathcal{K}_L	12.57	12.58	$2.69 \cdot 10^3$	$3.34 \cdot 10^3$	8.07
\mathcal{K}_{RBF}	12.81	12.80	$1.21 \cdot 10^4$	$1.23 \cdot 10^4$	17.67
$\mathcal{K}_{\mathcal{P}^2}$	12.53	12.55	$1.10 \cdot 10^4$	$1.09 \cdot 10^4$	11.31
$\mathcal{K}_{\mathcal{P}^3}$	12.54	12.55	$4.12 \cdot 10^4$	$4.49 \cdot 10^4$	11.19

The experiments were carried out in the Wheeler HPC of UNM-CARC, which uses SGI AltixXE Xeon X5550 at 2.67GHz with 6 GB of RAM per core, 8 cores per node, 304 nodes total, and runs at 25 theoretical peak FLOPS. It has Linux CentOS

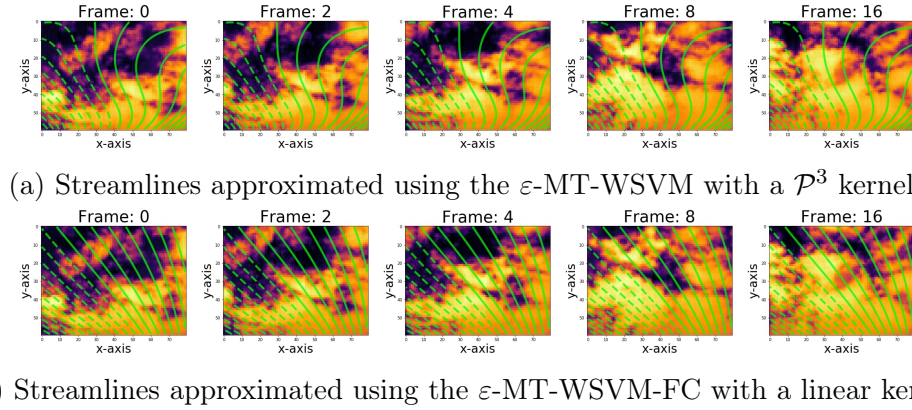


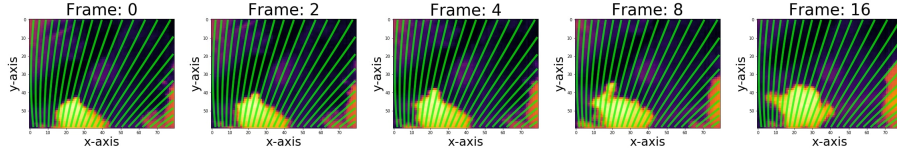
Figure 7.4: Comparison between (a) the streamlines computed by a standard Multi-output WSVM (ε -MT-WSVM) and (b) the introduced WSVM with divergence and vorticity constraints (ε -MT-WSVM-FC) in a sequence of images with elevation: 46.74° and azimuth: 174.21° . The images are organized chronologically from the left to the right. The time between frames is 15 s. The distance in the sequences across time is: 0 s, 30 s, 1 min 2 min 4 min. The sequence shows a day when a single cloud layer was detected. The top sequence visualizes a non-realistic approximation of the flow. A compression is induced to the gas in the bottom left of the frame, and an expansion is induced in the top right of the frame.

7 installed.

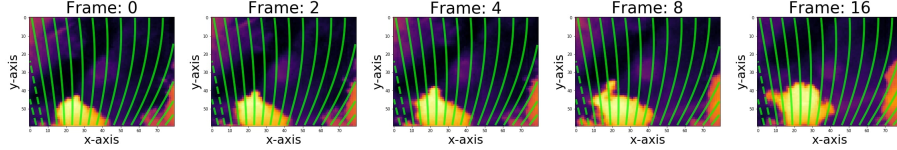
7.6 Discussion

This chapter adds new insights into the computational methods to forecast the trajectory of clouds and predict the occlusion of the Sun. The proposed method visualizes the wind velocity field using IR images of clouds. The algorithm introduced here differs from previous investigations in that it is based on fluid dynamics. The experiments show that the pathlines are equivalent to the streamlines in images where is possible to extract enough information about the wind flow from the clouds.

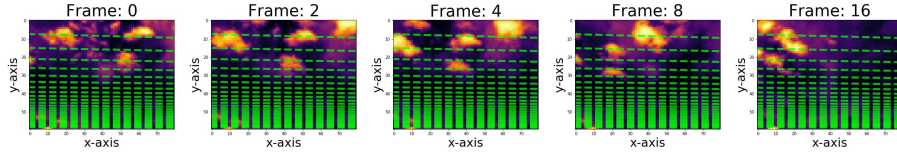
From the summary of the experiments presented in Table 7.2 and 7.1, several aspects can be highlighted. The overall performance of the ε -SVM increases when the



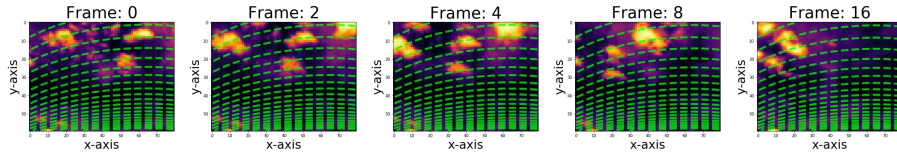
(a) Flow visualization of the upper cloud layer in day 2. Elevation: 55.66° ; azimuth: 200.35° .



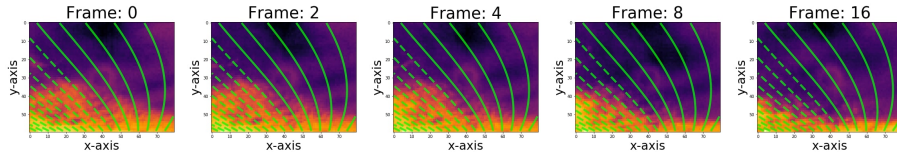
(b) Flow visualization of the lower cloud layer in day 2.



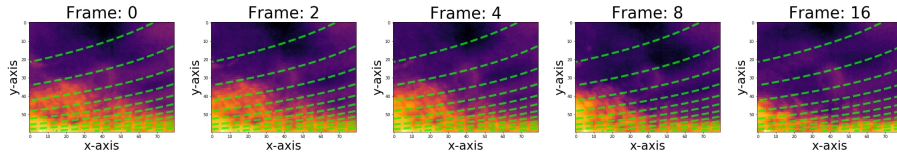
(c) Flow visualization of the upper cloud layer in day 4.



(d) Flow visualization of the lower cloud layer in day 3.



(e) Flow visualization of the upper cloud layer in day 4.



(f) Flow visualization of the lower cloud layer in day 4.

Figure 7.5: Streamlines approximated using the ε -MT-WSVM-FC with a linear kernel. Elevation: 32.15° ; azimuth: 180.29° . The sequence of IR images are organized chronologically from the left to the right similar to in Figure 7.4a and 7.4b. The displayed sequences are from days when two different cloud layers were detected. The upper layer of clouds is displayed in the top row of the sequence, the bottom row displays the lower layer.

samples are weighted, since the weights represent the probability of the vector to belong to the corresponding layer. Vectors with a very low probability do not contribute to the solution. Furthermore, the computing time of the ε -WSVM is lower than the ε -MT-WSVM as the Gram matrix dimensions are smaller. The flow divergence and vorticity are negligible when they are approximated using the ε -MT-WSVM-FC, but the computing time is larger. The results are similar between the three models but the ε -MT-WSVM and ε -MT-WSVM-FC models tend to show better performance.

The best result without cross-validation in WMAE is obtained by the ε -MT-WSVM-FC with RBF kernel (see Table 7.1). The flow approximated by this model has very low vorticity and divergence, which means that the pathlines and streamlines are approximately equivalent. When a trade-off is considered between vorticity, divergence, WMAE, and computing time, the most promising models are the ε -MT-WSVM-FC with linear kernel and RBF kernel. The computing time required for the linear kernel is lower, as the kernel does not have hyperparameters, but it is still high for a real-time application. Vorticity and divergence are removed in the approximated flow. On the other hand, the ε -WSVM with linear kernel, which has not flow constraints, is feasible in real-time application but the approximated flow is turbulent. When the pathlines begin to be the same as the streamlines, the flow constraints can be relaxed to reduce the vorticity and divergence within a feasible computing time.

In the implementation of the algorithm, the process of cross-validating the parameters of the ε -MT-WSVM-FC is computationally expensive, and the kernels may have hyperparameters which also require cross-validation. However, the optimal set of parameters is nearly identical during sort sequences. We propose to implement an exhaustive cross-validation in parallel with running the algorithm. This provides a pre-computed set of parameters for the ε -MT-WSVM-FC and the kernel function that can be used in the consecutive images until the online cross-validation is finished.

7.7 Conclusions

This chapter introduces a method to visualize wind velocity fields using physical features extracted from IR images of clouds. The images are recorded using a ground-based IR camera mounted on a solar tracker that maintains the Sun in the center of the images. The velocity vectors are transformed from the Euclidean frame of reference to the IR camera non-linear frame of reference. The wind velocity field estimation is based on unsupervised online ML methods that independently infer the distribution of the velocity vectors and the height of the clouds. Segmenting and subsampling the velocity vectors provides a computationally tractable solution. The wind velocity field is extrapolated to the entire frame using only information extracted from a cloud. This is achieved with the use of a ε -MT-WSVM which includes flow constraints in the QP problem formulation.

The methods to compute the motion vectors produce a noisy approximation of the velocity vectors in the frame. It is possible to improve the quality of the velocity vectors adding weights to the least-squares solution in the LK, and later segmenting the velocity vectors. Once the noise is reduced, a subsample of vectors is sufficient to approximate the velocity field in the entire frame. This makes a real-time implementation of the algorithm for wind flow visualization feasible. This is important, because the wind velocity field visualization predicts the pathlines of the clouds. The extrapolation of the wind velocity field to the entire frame is useful to anticipate where a cloud will be, or where it may appear in the frame. Additional constraints in the SVM yields better results, approximating the wind velocity field in the IR images.

Further research in this area may focus on predicting the occlusion of the Sun or the attenuation of irradiance using the streamline (i.e. pathline) that intercepts the Sun, and the magnitude of the wind velocity field in this streamline. The prediction of the wind velocity field distribution across space and time using Bayesian regression methods

is suitable for the selection of the most likely intercepting streamlines. Forecasting irradiance is out of the scope of this chapter. The prediction methods could use the accumulated distance of the pixels along the streamline (starting from the Sun) divided by the averaged magnitude of the approximated velocity vectors, to estimate the arrival time of the air parcel in that pixel. In the existing literature there are multiple optimization algorithms that might speed up the convergence of the ε -MT-WSVM-FC. These methods may increase the accuracy of intra-hour solar forecasting algorithms that are necessary to optimize the dispatch and storage of energy in power grids that used solar resources.

Chapter 8

Kernel Learning for Intra-Hour Solar Forecasting

8.1 Introduction

In intra-hour forecasting applications, ANNs are the most commonly used ML algorithms [14, 260]. ANNs are very efficient at handling big data problems, since the training is performed via stochastic gradient descent [192]. The main disadvantage of using a global learning approach (i.e., ANNs) is that these models might overlook local characteristics of the data space for the sake of global accuracy [132]. In these circumstances, local learning methods may provide a better understanding of the features contributions and the data space [181], leading to better performances [34]. The drawback of local learning models is that they require training multiple models [179], which is computationally expensive (involving matrix inversion). The reason behind the approach proposed in this chapter of training local models using kernel learning, is due to the relatively low number of samples. Kernel learning methods (i.e., shallow learning) have hyperparameters for fine-tuning the regularization of the

model parameters [341]. In contrast, ANNs require large amounts of data to avoid overfitting problems [361].

Kernel learning has been mainly implemented in day-ahead forecasting [231]. The methods most commonly found in the literature are the SVM and more recently the GPR [8, 154]. SVM are used in monthly [251], intra-week [75], and intra-day solar forecasting application [371] using feature vectors composed of weather features acquired in one or multiple weather stations [167]. When SVM are used for intra-hour PV power forecasting (from 15 to 300 minutes ahead) satellite images are required to estimate the motion of clouds [164]. The most effective intra-day PV power forecasting methods without sky images either apply processing techniques to the feature vectors [257] or are used in combination with measurements acquired from sensors installed in the PV systems [268]. Least-squares SVM [316] have also been previously implemented in GSI [126] and PV power forecasting [372]. Kernel Ridge Regression (KRR) has been put forward in PV power forecasting. In a particular case, the formulation of the KRR was improved for adding robustness to outliers [67]. In the most recent investigations that used kernel learning, GPRs are used in intra-week solar [280] and PV power forecasting [349].

In intra-hour solar forecasting, the direction of clouds is an important feature to determine if a cloud will occlude the Sun [80, 377]. To do this, it is necessary to analyze the dynamics of a cloud over a sequence of consecutive images [236]. This chapter introduces a method to compute the probability of a cloud intersecting the Sun, and uses it in the quantification of the cloud dynamics features extracted from a series of IR sky images. The statistics of the extracted features are computed to form the feature vectors that are utilized in the proposed intra-hour solar forecasting.

The performance of PV panels and power electronics depends on the manufacturing process and degrades following different patterns [137]. When the GSI is used as a predictor, the information obtained from solar forecasting may be shared between

nearby PV systems if the degradation pattern of each PV system is known [89, 282]. A time series model improves forecasting performances when the deterministic component is removed [177, 211]. In this chapter, the GSI measurements are detrended to obtain the CSI [90], and the performances of multi-task kernel learning models (i.e., SVM, Relevance Vector Machine (RVM), KRR and GPR) are compared when using multiple kernel functions in combination with different feature vectors. The features included in the vectors were extracted from the analyses of cloud dynamics.

8.2 Kernel Methods

A kernel method can be seen as an extension of a linear algorithm that acquires nonlinear properties through the so-called “Kernel trick”. A kernel is a definite positive function $\mathcal{K}(\cdot, \cdot)$ that maps a pair of feature vectors in a space $\mathbb{R}^{D \times D}$ into \mathbb{R} . Since this function is positive definite, the Mercer’s theorem [5, 16, 232] states that it is a dot product in a Hilbert space \mathcal{H} spanned by functions $\varphi(\mathbf{x})$ such that $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle$. The Generalized Representer Theorem [288] states that the parameters \mathbf{w} of an estimation function $f(\mathbf{x}) = \mathbf{w}^\top \varphi(\mathbf{x})$ are expressible as a linear combination $\mathbf{w} = \sum_i \alpha_i \varphi(\mathbf{x}_i)$ of the training data. For any (nonlinear) kernel, a linear algorithm has a nonlinear counterpart which can be written as

$$f(\mathbf{x}) = \sum_i \alpha_i \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}) \rangle = \sum_i \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) \quad (8.1)$$

thus being possible to generalize any linear algorithm to have nonlinear properties. In particular, SVM or GPR classification or regression algorithms, KRR or others can be endowed with nonlinear properties in a straightforward way as it will be summarized below.

Kernel regression models can be extended to multi-task kernel regression models. Multi-task models produce multiple predictors $\hat{\mathbf{y}}_i$ from an unique covariate vector

\mathbf{x}_i (i.e., feature vector). In this chapter, the kernel methods are divided into dense and sparse methods. At the same time, a deterministic and probabilistic model are proposed within the dense and sparse methods.

Kernel Functions. The proposed kernel functions in this analysis are linear (L), polynomial of order n (\mathcal{P}^n), Radial Basis Function (RBF), Rational Quadratic (RQ), and Matérn (M) [297]. Their respective functions are,

$$\begin{aligned}
 \mathcal{K}_L(\mathbf{x}_i, \mathbf{x}_j) &= \gamma \mathbf{x}_i^\top \mathbf{x}_j, \\
 \mathcal{K}_{\mathcal{P}^n}(\mathbf{x}_i, \mathbf{x}_j) &= (\gamma \mathbf{x}_i^\top \mathbf{x}_j + \beta)^n, \\
 \mathcal{K}_{RBF}(\mathbf{x}_i, \mathbf{x}_j) &= \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \\
 \mathcal{K}_{RQ}(\mathbf{x}_i, \mathbf{x}_j) &= \left(1 + \frac{1}{2\alpha} \gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)^{-\alpha}, \\
 \mathcal{K}_M^\nu(\mathbf{x}_i, \mathbf{x}_j) &= \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \cdot \gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)^\nu K_\nu \left(\sqrt{2\nu} \cdot \gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right),
 \end{aligned} \tag{8.2}$$

where $\gamma, \beta, \alpha, \nu \in \mathbb{R}^+$, and $n \in \mathbb{N}$ are the kernel hyperparameters [289]. $\Gamma(\cdot)$ is the Gamma function, and K_ν is the modified Bessel function of second kind.

The hyperparameter γ, β, α are cross-validated. The only polynomial kernel validated is when $n = 2$, since exploratory results showed that polynomial kernels of higher order leads to poor performances. The hyperparameters of the Matén Kernel are independently validated for $\nu \in \{1/2, 3/2, 5/3\}$, which are the standard values used for the smoothing parameter.

8.2.1 Dense Kernel Methods

A dense kernel method utilizes the entire training set to produce a prediction. In contrast, a sparse kernel method utilizes a subset of the training data yielding (generally) faster models in the implementation. In this chapter, sparsity is not an essential

requirement because a subset of samples is selected from the database, instead of training a model using the entire dataset.

Kernel Ridge Regression

Assume a linear regression model of the form:

$$y_i = \hat{y}_i + \varepsilon_i = \mathbf{w}^\top \varphi(\mathbf{x}_i) + \varepsilon_i. \quad (8.3)$$

The KRR is a simplification of the Tikhonov regularization [330], in which the minimized loss functions is the Mean Squared Error (MSE) with a quadratic norm regularization applied to the parameters \mathbf{w} ,

$$\min_{\mathbf{w}} \mathbb{E} [y_i - \mathbf{w}^\top \varphi(\mathbf{x}_i)] + \gamma \|\mathbf{w}\|_2. \quad (8.4)$$

where γ is the regularization hyperparameter.

Applying the representer theorem [288], which states that the dual formulation of the model parameters is $\mathbf{w} = \boldsymbol{\Phi} \boldsymbol{\alpha}$, and adding the Vapnik-Chervonenkis generalization bound [340] to the formulation, we have that,

$$\min_{\boldsymbol{\alpha}} \mathbb{E} [y_i - \boldsymbol{\alpha}^\top \boldsymbol{\Phi}^\top \varphi(\mathbf{x}_i)] + \frac{\gamma}{N} \boldsymbol{\alpha}^\top \boldsymbol{\Phi}^\top \boldsymbol{\Phi} \boldsymbol{\alpha} \quad (8.5)$$

where $\boldsymbol{\Phi} = [\varphi(\mathbf{x}_1) \cdots \varphi(\mathbf{x}_N)]$ is a matrix containing all training samples mapped into a reproducing kernel Hilbert space.

Finding the optimal regularization hyperparameter requires cross-validation. Nevertheless, the model parameters that minimized the MSE are analytically found nulling the gradient,

$$\begin{aligned} 0 &= \frac{\partial}{\partial \boldsymbol{\alpha}} \left[(\mathbf{y} - \boldsymbol{\alpha}^\top \mathbf{K}) (\mathbf{y} - \boldsymbol{\alpha}^\top \mathbf{K})^\top + \frac{\gamma}{N} \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} \right], \\ 0 &= \frac{\partial}{\partial \boldsymbol{\alpha}} \left[\mathbf{y} \mathbf{y}^\top + \boldsymbol{\alpha}^\top \mathbf{K} \mathbf{K}^\top \boldsymbol{\alpha} - 2 \boldsymbol{\alpha}^\top \mathbf{K} \mathbf{y}^\top + \frac{\gamma}{N} \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} \right], \\ 0 &= 2 \mathbf{K} \mathbf{K}^\top \boldsymbol{\alpha} - 2 \mathbf{K} \mathbf{y}^\top + \frac{2\gamma}{N} \mathbf{K} \boldsymbol{\alpha}, \\ \boldsymbol{\alpha} &= \left(\mathbf{K} + \frac{\gamma}{N} \mathbf{I} \right)^{-1} \mathbf{y}, \end{aligned} \quad (8.6)$$

where $\mathbf{K} = \Phi^\top \Phi$. A prediction for a new observation is obtained with this formula,

$$\hat{y}_* = \sum_{i=1}^N \alpha_i \varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_*) = \sum_{i=1}^N \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*). \quad (8.7)$$

The optimal kernel hyperparameters are found implementing a cross-validation method.

Multi-Task Kernel Ridge Regression (MT-KRR). The dual formulation of the MT-KRR as a MSE minimization problem is,

$$\min_{\tilde{\alpha}} \mathbb{E} [\tilde{\mathbf{y}} - \tilde{\alpha}^\top \tilde{\mathbf{K}}] + \frac{\gamma}{CN} \|\tilde{\alpha}^\top \tilde{\mathbf{K}} \tilde{\alpha}\|_2. \quad (8.8)$$

where $\tilde{\mathbf{K}} = \mathbf{\Gamma} \otimes \mathbf{K}$, the extended vector of predictors is $\tilde{\mathbf{y}} = [\mathbf{y}_1^\top \cdots \mathbf{y}_N^\top]^\top \in \mathbb{R}^{1 \times CN}$, and C is the number of forecasting outputs in the model.

The dual parameters $\tilde{\alpha}$ have an analytical solution analogous to the KRR which is $\tilde{\alpha} = (\tilde{\mathbf{K}} + \frac{\gamma}{CN} \mathbf{I}_{N \times N} \otimes \mathbf{I}_{C \times C})^{-1} \tilde{\mathbf{y}}$. The ridge regularization parameter is γ , and $\mathbf{\Gamma}$ is the matrix that contains the correlation coefficients between the multiple outputs. A prediction for a new observation is obtained from

$$\hat{\mathbf{y}}_* = \sum_{i=1}^{CN} \tilde{\alpha}_i [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)], \quad (8.9)$$

where a prediction $\hat{\mathbf{y}}_*$ is the vector of the multiple forecasting horizons $\hat{\mathbf{y}}_* = [\hat{y}_1 \cdots \hat{y}_C]^\top$.

Gaussian Process for Regression

Consider the standard model in Bayesian regression whose feature vectors are projected into a feature space [274],

$$y_i = f(\mathbf{x}_i) + \varepsilon_i = \mathbf{w}^\top \varphi(\mathbf{x}_i) + \varepsilon_i. \quad (8.10)$$

In this model, the prediction of the latent function $f(\mathbf{x}_i)$ is assumed to have a Gaussian independent and identically distributed error $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$, and the likelihood function

of the observations is $y_i \sim \mathcal{N}(\mathbf{w}^\top \varphi(\mathbf{x}_i), \sigma_n^2 \mathbf{I})$. The parameters \mathbf{w} are defined as a latent random variable whose prior is $\mathcal{N}(\mathbf{0}, \Sigma_p)$, and when applying the Representer Theorem, it is obtained that the dual representation of \mathbf{w} is

$$f(\mathbf{x}_i) = \sum_{j=1}^N \alpha_j \varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_j) = \boldsymbol{\alpha} \mathbf{K} \quad (8.11)$$

where $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \triangleq \varphi(\mathbf{x}_i)^\top \Sigma_p \varphi(\mathbf{x}_j)$ and \mathbf{K} is the matrix of dot products between training data. Therefore, the prior of the dual representation of the parameters is $p(\boldsymbol{\alpha} | \mathbf{X}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^{-1})$

The dual of parameters $\boldsymbol{\alpha}$ that maximize the posterior distribution given the training predictors y_i (i.e., MAP estimation) are proportional to the prior times the likelihood,

$$p(\boldsymbol{\alpha} | \mathbf{X}, \mathbf{y}) \propto p(\mathbf{y} | \mathbf{X}, \boldsymbol{\alpha}) p(\boldsymbol{\alpha} | \mathbf{X}). \quad (8.12)$$

The mean and covariance matrix of the posterior distribution $p(\boldsymbol{\alpha} | \mathbf{X}, \mathbf{y}) \sim \mathcal{N}(\bar{\boldsymbol{\alpha}}, \mathbf{B})$ are $\bar{\boldsymbol{\alpha}} = (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y}$ and $\mathbf{B} = \mathbf{K}^{-1} - (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1}$ respectively.

As the posterior probability of a prediction $p(f(\mathbf{x}_*) | \mathbf{X}, \mathbf{y})$ is a Gaussian distribution, it is sufficient to compute the mean and variance with respect to the posterior of $\boldsymbol{\alpha}$ [217]. Hence, the mean and the variance of a predictions are respectively

$$\begin{aligned} \bar{f}(\mathbf{x}_*) &= \mathbb{E}_{\boldsymbol{\alpha} | \mathbf{X}, \mathbf{y}} [\boldsymbol{\alpha}^\top \mathbf{k}(\mathbf{x}_*)] = \bar{\boldsymbol{\alpha}}^\top \mathbf{k}(\mathbf{x}_*), \\ \sigma_*^2 &= \mathbb{E}_{\boldsymbol{\alpha} | \mathbf{X}, \mathbf{y}} \left[\mathbf{k}(\mathbf{x}_*)^\top \boldsymbol{\alpha} \boldsymbol{\alpha}^\top \mathbf{k}(\mathbf{x}_*) \right] = \mathbf{k}(\mathbf{x}_*)^\top \mathbf{B} \mathbf{k}(\mathbf{x}_*) \\ &= \mathcal{K}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*)^\top (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{k}(\mathbf{x}_*). \end{aligned} \quad (8.13)$$

where $\mathbf{k}(\mathbf{x}_*) \triangleq \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)$ is a column vector containing all dot products.

The MLL (i.e., evidence) is used to optimize the kernel hyperparameters via gradient descent,

$$\log p(\mathbf{y} | \mathbf{X}) = -\frac{1}{2} \mathbf{y}^\top \bar{\boldsymbol{\alpha}} - \frac{1}{2} |\mathbf{K} + \sigma_n^2 \mathbf{I}| - \frac{N}{2} \log 2\pi. \quad (8.14)$$

Multi-Task Gaussian Process for Regression. Similarly to a GPR, the dual representation of parameters $\tilde{\boldsymbol{\alpha}}$ in a MT-GPR have analytical solution when the likelihood $p(\mathbf{Y}|\mathbf{X}, \tilde{\boldsymbol{\alpha}}) \sim \mathcal{N}(\tilde{\boldsymbol{\alpha}}[\mathbf{\Gamma} \otimes \mathbf{K}], \mathbf{\Sigma}_n \otimes \mathbf{I})$ and the prior $p(\tilde{\boldsymbol{\alpha}}|\mathbf{X}) \sim \mathcal{N}(\mathbf{0}, [\mathbf{\Gamma} \otimes \mathbf{K}]^{-1})$ are defined as multivariate normal distribution. The predicted mean vector and covariance matrix for a new sample are

$$\begin{aligned} \bar{f}(\mathbf{x}_*) &= \tilde{\boldsymbol{\alpha}}^\top [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)], \\ \hat{\boldsymbol{\Sigma}}_* &= \mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_*, \mathbf{x}_*) - [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)]^\top \left(\tilde{\mathbf{K}} + \mathbf{\Sigma}_n \mathbf{I} \right)^{-1} \mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*), \end{aligned} \quad (8.15)$$

where $\tilde{\boldsymbol{\alpha}} = (\tilde{\mathbf{K}} + \mathbf{\Sigma}_n \otimes \mathbf{I})^{-1} \tilde{\mathbf{y}}$, and $\tilde{\mathbf{K}} = \mathbf{\Gamma} \otimes \mathbf{K}$. The matrix that models the correlation between outputs is $\mathbf{\Gamma}$, and the matrix of noises is defined as $\mathbf{\Sigma}_n = \text{diag}(\sigma_{c,n}^2)$, where $\sigma_{c,n}^2$ is the noise of output c .

The the optimal kernel hyperparatermes in a MT-GPR, are obtained minimizing the negative MLL via gradient descent,

$$\log p(\mathbf{Y}|\mathbf{X}) = -\frac{1}{2} \tilde{\mathbf{y}}^\top \tilde{\boldsymbol{\alpha}} - \frac{1}{2} \left| \tilde{\mathbf{K}} + \mathbf{\Sigma}_n \otimes \mathbf{I} \right| - \frac{CN}{2} \log 2\pi, \quad (8.16)$$

where C is the number of forecasting horizons. The correlation between outputs matrix $\mathbf{\Gamma}$ and the noise matrix $\mathbf{\Sigma}_n$ have analytical solutions when the Expectation-Maximization algorithm is implemented in the hyperparameters optimization [33].

8.2.2 Sparse Kernel Methods

The main idea behind sparse kernel methods is that noisy vectors exist in the nonlinear transformation to the feature space $\varphi(\mathbf{x}_i)$. Therefore, the transformation can be approximated by a subset of the data. The resulting learning models are computationally faster during testing.

The sparse kernel methods proposed in this chapter are ε -SVM and RVM. The disadvantage of ε -SVM is that it does not predict a distribution. For this reason,

we implement a RVM that is a Bayesian sparse kernel method. We compare their performances in the application of solar irradiance forecasting.

Support Vector Machine for Regression

The regression problem in a ε -SVM applies an ε -insensitive loss function to the formulation for sparseness purposes [289],

$$|y_i - f(\mathbf{x}_i)|_\varepsilon = \max[0, |y_i - f(\mathbf{x}_i)| - \varepsilon], \quad \forall i = 1, \dots, N, \quad y_i, \varepsilon \in \mathbb{R}, \quad (8.17)$$

where $f(\mathbf{x}_i) = \mathbf{w}^\top \varphi(\mathbf{x}_i) + b$, $b \in \mathbb{R}$. The ε -insensitive loss function does not penalize errors that are below $|\varepsilon| > 0$ [82].

The ε -SVM aims to estimate the $f(\cdot)$ that minimizes following constrained problem,

$$\begin{aligned} \min_{\mathbf{w}, b, \xi, \xi^*} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + \mathcal{C} \sum_{i=1}^N (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & \begin{cases} y_i - \mathbf{w}^\top \varphi(\mathbf{x}_i) - b \leq \varepsilon + \xi_i \\ \mathbf{w}^\top \varphi(\mathbf{x}_i) + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \quad i = 1, \dots, N, \end{aligned} \quad (8.18)$$

in which the L2-norm is applied to the parameters \mathbf{w} to control the model complexity, the ε -loss function controls the training error through the hyperparameter \mathcal{C} , and ξ_i are the slack variables. The slack variables are introduced to relax the constraints of the optimization problem, so it is feasible to deal with non-convex problems [63].

To minimize the constrained problem in Eq. (8.18), a Lagrangian functional is defined through a set of dual variables using the functional and the set of constraints [305]. The derivatives of the functional with respect to the primal variables $\mathbf{w}, \varepsilon, \xi_i, \xi_i^*$ leads to a set of equations that are a case of KKT conditions. Substituting these equations on the Lagrangian functional, together with the complimentary KKT

condition (which forces the product of dual parameters α_i, α_i^* with the constraints to be zero) yield to the following dual functional that has the form of a solvable QP problem,

$$\begin{aligned} \min_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*} \quad & \frac{1}{2} (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*)^\top \mathbf{K} (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) + (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*)^\top \mathbf{y} + \varepsilon \mathbf{1}^\top (\boldsymbol{\alpha} + \boldsymbol{\alpha}^*) \\ \text{s.t.} \quad & \begin{cases} \mathbf{1}^\top (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) = 0 \\ 0 \leq \alpha_i, \alpha_i^* \leq \mathcal{C} \end{cases} \quad \forall i = 1, \dots, N. \end{aligned} \quad (8.19)$$

where \mathbf{K} is the Gram matrix that contains the dot products $\mathbf{K}_{i,j} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ and $\mathbf{1}_{1 \times N} = [1 \cdots 1]^\top$ is a vector of ones.

The approximated function in Eq. (8.17) evaluated for a new sample \mathbf{x}_* is,

$$f(\mathbf{x}_*) = \sum_{i=1}^N (\alpha_i - \alpha_i^*) \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*) + b, \quad (8.20)$$

where b is obtained from the complimentary KKT conditions.

Multi-Task Support Vector Machine for Regression (ε -MT-SVM). The primal ε -MT-SVM problem can be formulated as

$$\mathbf{y}_i = \tilde{\mathbf{W}}^\top [\mathbf{\Gamma} \otimes \varphi(\mathbf{x}_i)] + \mathbf{b}, \quad (8.21)$$

where the column vectors of primal parameter $\tilde{\mathbf{W}}$ and model bias $\mathbf{b} = [b_1 \cdots b_C]^\top$ approximates each one of the predictors $\mathbf{y}_i \in \mathbb{R}^C$. The matrix $\mathbf{\Gamma}$ contains the correlation parameters between outputs. Primal parameters are a function of the dual parameters $\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_i^*$ as well.

The Gram matrix $\tilde{\mathbf{K}}_{CN \times CN}$ in the ε -MT-SVM formulation for correlated outputs is $\tilde{\mathbf{K}} = \mathbf{\Gamma} \otimes \mathbf{K}$. Therefore, the dual formulation of the QP problem for the ε -MT-SVM

is,

$$\begin{aligned} \min_{\tilde{\alpha}, \tilde{\alpha}^*} \quad & \frac{1}{2} (\tilde{\alpha} - \tilde{\alpha}^*)^\top \tilde{\mathbf{K}} (\tilde{\alpha} - \tilde{\alpha}^*) + \tilde{\mathbf{y}}^\top (\tilde{\alpha} - \tilde{\alpha}^*) + \varepsilon \mathbf{1}^\top (\tilde{\alpha} + \tilde{\alpha}^*) \\ \text{s.t.} \quad & \begin{cases} \mathbf{1}^\top (\tilde{\alpha} - \tilde{\alpha}^*) = 0 \\ \mathbf{0} \leq \tilde{\alpha}_i, \tilde{\alpha}_i^* \leq \mathcal{C} \end{cases} \quad \forall i = 1, \dots, 2CN. \end{aligned} \quad (8.22)$$

A multi-task prediction is performed applying this formula,

$$\hat{\mathbf{y}}_* = [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)] (\tilde{\alpha} - \tilde{\alpha}^*)^\top + \mathbf{b}. \quad (8.23)$$

Relevance Vector Machine for Regression

This model presents as an alternative solution of the functional in a ε -SVM with the advantage of producing probabilistic predictions [331]. The Bayesian formulation of the likelihood and prior in a RVM is equivalent to a Gaussian process, but this model is endowed with an automatic relevance determination mechanisms that causes a subset of the parameters \mathbf{w} to drive to zero [332].

The likelihood function of the observations is defined as multivariate normal distribution $p(y_i | \mathbf{x}_i, \mathbf{w}, \sigma_n^{-2}) \sim \mathcal{N}(\mathbf{w}^\top \varphi(\mathbf{x}_i), \sigma_n^{-2} \mathbf{I})$, and a zero-mean Gaussian prior is set on the weights $p(w_j | \lambda_j^{-1}) \sim \mathcal{N}(w_j | 0, \lambda_j^{-1})$. Applying the Bayes theorem, we obtained that the posterior distribution of the weights is also Gaussian,

$$p(\mathbf{w} | \mathbf{y}, \mathbf{X}, \boldsymbol{\alpha}, \sigma_n^{-2}) = \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}, \boldsymbol{\Sigma}). \quad (8.24)$$

The mean and covariance matrix of the posterior distribution are,

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)} &= \left(\boldsymbol{\Lambda}^{(t)} + \sigma_n^{-2(t)} \mathbf{K}^\top \mathbf{K} \right)^{-1} \\ \boldsymbol{\mu}^{(t+1)} &= \sigma_n^{-2(t)} \boldsymbol{\Sigma}^{(t+1)} \mathbf{K}^\top \mathbf{y} \end{aligned} \quad (8.25)$$

where σ^2 is the variance of the noise, t represents the current optimization iteration, $\boldsymbol{\Lambda} = \text{diag}(\boldsymbol{\lambda})$ are the precision of the parameters, and \mathbf{K} is the Gram matrix.

The marginal likelihood is maximized to find the optimal hyperparameters,

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\lambda}, \sigma_n^{-2}) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma_n^{-2}) p(\mathbf{w}|\boldsymbol{\lambda}) d\mathbf{w}. \quad (8.26)$$

The hyperparameters that maximizes the MLL are found analytically equaling the derivatives to zero,

$$\begin{aligned} \gamma_i^{(t+1)} &= 1 - \lambda_i^{(t)} \Sigma_{i,i}^{(t+1)} \\ \lambda_i^{(t+1)} &= \frac{\gamma_i^{(t+1)}}{\mu_i^{(t+1)}} \\ \sigma_n^{2(t+1)} &= \frac{\|\mathbf{y} - \mathbf{K}\boldsymbol{\mu}^{(t+1)}\|^2}{N - \sum_{i=1}^N \gamma_i^{(t+1)}} \end{aligned} \quad (8.27)$$

where γ_i is the relevance measure of vector i . The optimization algorithm updates the parameters iteratively until reaches the convergence criterion.

The RVM characteristic sparseness arises when there is poor alignment between the direction of $\varphi(\mathbf{x}_i)$ and y_i , then $\lambda_i \rightarrow \infty$ and consequently the parameter w_i posterior distribution mean and variance will tend to zero [31]. This causes the vector $\varphi(\mathbf{x}_i)$ to be removed from the model. The remaining of the feature vectors $\varphi(\mathbf{x}_i)$ with non-zero posterior mean and variance are the so-called relevance vectors [97].

A prediction for a new sample \mathbf{x}_* is obtained as,

$$\begin{aligned} \hat{y}_* &= \sum_{i=1}^N \mu_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*), \\ \hat{\sigma}_*^2 &= \sigma_n^2 + \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)^\top \boldsymbol{\Sigma} \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*). \end{aligned} \quad (8.28)$$

The mechanism of sparsity can be exploited to derive a faster optimization of hyperparameters [333]. However, that method was not implemented in this chapter due to our low number of samples.

Multi-Task Relevance Vector Machine for Regression (MT-RVM). This model is implemented converting the predictors matrix into a vector form $\tilde{\mathbf{y}}$, and

extending the model parameters using the Kronecker product, in an analogous manner that previous multi-task models,

$$\begin{aligned}\tilde{\Sigma}^{(t+1)} &= \left(\mathbf{\Lambda}^{(t)} \otimes \mathbf{I} + \sigma_n^{-2(t)} \tilde{\mathbf{K}}^\top \tilde{\mathbf{K}} \right)^{-1}, \\ \tilde{\boldsymbol{\mu}}^{(t+1)} &= \sigma_n^{-2(t)} \tilde{\Sigma}^{(t+1)} \tilde{\mathbf{K}}^\top \tilde{\mathbf{y}},\end{aligned}\tag{8.29}$$

where $\tilde{\mathbf{K}} = \mathbf{\Gamma} \otimes \mathbf{K}$. The extension of the parameters that maximized the log-likelihood to a multi-task model are,

$$\begin{aligned}\gamma_i^{(t+1)} &= 1 - \lambda_i^{(t)} \tilde{\Sigma}_{i,i}^{(t+1)}, \\ \lambda_i^{(t+1)} &= \frac{\gamma_i^{(t+1)}}{\tilde{\mu}_i^{(t+1)}}, \\ \sigma_n^{2(t+1)} &= \frac{\|\tilde{\mathbf{y}} - \tilde{\mathbf{K}} \tilde{\boldsymbol{\mu}}^{(t+1)}\|^2}{CN - \sum_{i=1}^{CN} \gamma_i^{(t+1)}},\end{aligned}\tag{8.30}$$

where C is the number of outputs in the forecasting model.

A prediction of a new multi-task observation in a MT-RVM is,

$$\begin{aligned}\hat{\mathbf{y}}_* &= [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)] \tilde{\boldsymbol{\mu}}^\top, \\ \hat{\boldsymbol{\sigma}}_*^2 &= \sigma_n^2 + [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)]^\top \tilde{\Sigma} [\mathbf{\Gamma} \otimes \mathcal{K}(\mathbf{x}_i, \mathbf{x}_*)].\end{aligned}\tag{8.31}$$

The optimization yields to a sparse solution as well, so similarly the extended posterior mean and variance of the non-relevant vectors tends to zero.

8.2.3 Kernel Regression Chain

Another type of multi-task model proposed in this chapter are the regression chains [108]. The models in a chain are arranged in a chronological sequence [229], so that

the prediction of a model is added to the feature vectors of following models,

$$\hat{\mathbf{y}}_* = \begin{bmatrix} \hat{y}_{1*} \\ \hat{y}_{2*} \\ \hat{y}_{3*} \\ \vdots \\ \hat{y}_{C*} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1^\top \varphi(\mathbf{x}_*) \\ \mathbf{w}_2^\top \varphi(\mathbf{x}_*, \hat{y}_{1*}) \\ \mathbf{w}_3^\top \varphi(\mathbf{x}_*, \hat{y}_{1*}, \hat{y}_{2*}) \\ \vdots \\ \mathbf{w}_C^\top \varphi(\mathbf{x}_*, \hat{y}_{1*}, \hat{y}_{2*}, \dots, \hat{y}_{C-1*}) \end{bmatrix}, \quad (8.32)$$

where C is the number of sequential forecasting horizons in the regression chain.

In this chapter, the inference of the model parameters and the kernel hyperparameters is performed independently for each model in the chain. A regression chain is implemented for each one of the explained kernel methods.

8.2.4 Multi-Task Kernel Simplification

We propose to model the process $\dots, y_{k-1}, y_k, y_{k+1}, \dots$ as an autoregressive process. To do that, the matrix $\mathbf{\Gamma} \in \mathbb{R}^{C \times C}$ is defined to contain the correlation coefficients between each forecasting output C , and the correlation coefficients $\gamma_{i,j}$ require cross-validation. Unfortunately, the number of correlation coefficients $\gamma_{i,j}$ is large C^2 , so the cross-validation procedure is generally intractable in kernel learning applications. For simplification, we propose to determine the correlation coefficients modelling them as a decaying autoregressive model,

$$\gamma_{i,j} = \exp\left(\frac{C - |i - j|}{C\ell}\right), \quad \forall i, j = 1, \dots, C, \quad (8.33)$$

where ℓ is the length-scale and it is the only hyperparameter in $\mathbf{\Gamma}$ that requires cross-correlation. An advantage of this simplification is that the matrix $\mathbf{\Gamma}$ is symmetric (i.e., $\forall i, j, \gamma_{i,j} = \gamma_{j,i}$), so the computation cost is also reduced.

8.3 Feature Extraction

The IR camera used in this dissertation is an uncooled microbolometer that measures the temperature changes produced by black body radiation. The Wien's displacement law states that the black body radiation wavelength is inversely proportional to the temperature of the object [103]. Consequently, we know that the temperature of the clouds moving in the troposphere is within long-wave IR radiation [113]. This is because the black body radiation spectrum for different temperatures reaches their maxima at different wavelengths. Using consecutive IR images, the radiometric measurements of the IR camera allow us to derive physical features related to the clouds' thermal dynamics and motion.

8.3.1 Cloud Dynamics

The displacement of the pixels in consecutive IR images is analyzed to compute the cloud dynamics. The cloud velocity vectors are first computed for further extracting second order dynamics from them (i.e., curl and divergence). The following features extracted from the cloud dynamics, are included in the feature vectors used in the forecasting.

- Velocity Vectors

We define the cloud's velocity vector in the classic manner [35], which is the rate of change of an object along time,

$$\left(\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}, \frac{\partial z}{\partial t} \right) = (u, v, 0), \quad (8.34)$$

where x and y are the coordinate system of the camera plane, and t is the time variable. Variable z is the height of the cloud layer, that we assume that does not have vertical movement. The approximated cloud velocity vectors

are $\hat{\mathbf{V}} = \{(\hat{u}_{i,j}, \hat{v}_{i,j}) \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ in the x and y component respectively. The cloud velocity vectors are computed using a weighted implementation of the WLK algorithm (see Chapter 6). We propose to use the velocity vectors to extract second order features of the cloud dynamics, and their magnitude $M_{i,j} = (\hat{u}_{i,j}^2 + \hat{v}_{i,j}^2)^{1/2}$ to estimate the time instant a cloud will intersect the Sun.

- Divergence

The divergence $\mathbf{D} = \{D_{i,j} \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ of the clouds in the images, is a scalar field that describes how this is expanding or compressing at a given point. In the case of a 2-dimensions velocity field the divergence is calculated as the dot product between operator ∇ and $\hat{\mathbf{V}}$,

$$\mathbf{D} = \nabla \cdot \hat{\mathbf{V}} = \begin{pmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{pmatrix} \cdot \begin{pmatrix} u \\ v \\ 0 \end{pmatrix} = \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right). \quad (8.35)$$

- Curl

The vorticity or curl $\mathbf{V} = \{V_{i,j} \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$, is a pseudo-vector field that describes the rotation of the clouds velocity vectors $\hat{\mathbf{V}}$. The curl in an image is obtained taking the cross-product between the operator ∇ and the cloud velocity vectors $\hat{\mathbf{V}}$. In a 2-dimensions space can be calculated as

$$\mathbf{V} = \nabla \times \hat{\mathbf{V}} = \begin{vmatrix} \hat{\mathbf{i}} & \hat{\mathbf{j}} & \hat{\mathbf{k}} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ u & v & 0 \end{vmatrix} = \left(\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} \right) \hat{\mathbf{k}}. \quad (8.36)$$

8.3.2 Feature Selection

The wind velocity field in the atmosphere cross-section plane of a cloud layer $\hat{\mathbf{W}} = \{(\hat{u}_{i,j}, \hat{v}_{i,j}) \in \mathbb{R}^2 \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ is approximated using a wind flow visualization algorithm (see Chapter 7). Assuming that the air parcel in the IR image is sufficient small to consider the wind velocity field incompressible and irrotational, the streamlines are equivalent to pathlines. The streamlines and potential lines are computed using the approximated wind velocity field $\hat{\mathbf{W}}$, and are $\Phi = \{\phi_{i,j} \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ and $\Psi = \{\psi_{i,j} \in \mathbb{R} \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$ respectively.

Sun Intersecting Streamline

The Sun intersecting streamline in the potential direction can be found following an iterative connected component scheme. The proposed scheme begins at the position of the Sun $\mathbf{x}_0 = \{i_0, j_0\}$, so that $i^{(1)} = i_0$ and $j^{(1)} = j_0$, and follows this iterative position update

$$\begin{aligned} i^{(t+1)}, j^{(t+1)} &= \underset{i,j}{\operatorname{argmin}} \left(\Phi_{i,j \in \Upsilon^{(t)}} - \Phi_{i^{(t)}, j^{(t)}} \right)^2, \\ \text{s.t. } &\begin{cases} \Psi_{i^{(t+1)}, j^{(t+1)}} > \Psi_{i_0, j_0} \\ i^{(k+1)} \neq i^{(t)} \wedge j^{(t+1)} \neq j^{(k)}, i^{(t+1)}, j^{(t+1)} \in \Upsilon^{(t)}, \end{cases} \end{aligned} \quad (8.37)$$

where $\Upsilon^{(t)}$ is the set of pixels in the neighborhood of $i^{(t)}, j^{(t)}$, which is defined as $\Upsilon^{(t)} = \{(i^{(t)} + m, j^{(t)} + n) \mid m = 0 \wedge n = 0 \mid \forall m, n = -1, 0, 1\}$, and t is the iteration. The optimization scheme continues while $0 < i^{(t+1)} < M \wedge 0 < j^{(t+1)} < N$, when the streamlines connects with a pixel in the edge of the image, the algorithm stops. Therefore, the intersecting streamline $\mathcal{S} = \{(i^{(\ell)}, j^{(\ell)}) \mid \forall \ell = 1, \dots, L\}$ is the set of connected pixels from the Sun to the edge of the image.

The above constrains have to be applied to the problem in the light of finding a

feasible intersecting streamline. First, the flow potential direction has to be greater than the potential at the position of the Sun Ψ_{i_0, j_0} . Second, non-pixel coordinates can be repeated in a streamline.

Probability of a Pixel Intersecting the Sun

The space coordinates in an image are distorted by the perspective of the camera, which depends on the altitude where the cloud layer is flowing. The Euclidean coordinates system of the IR sensor plane is reprojected to the atmosphere cross-section plane of the cloud layer that is distorted by the perspective in the image (see Chapter 5). The geospatial perspective reprojection is function $\psi : (i, j; \varepsilon, \alpha, h) \mapsto \mathbf{x}_{i,j}, \Delta \mathbf{x}_{i,j}$ that maps the Euclidean coordinates to the atmosphere cross-section plane. The function depends on the Sun elevation ε and azimuth α angles, and the height of the cloud layer h . The function maps the coordinate system i, j to the atmosphere cross-section plane $\mathbf{X} = \{(x_{i,j}, y_{i,j}) \in \mathbb{R}^2 \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$, and to the dimensions of the atmosphere cross-section plane contained in a pixel $\Delta \mathbf{X} = \{(\Delta x_{i,j}, \Delta y_{i,j}) \in \mathbb{R}^2 \mid \forall i = 1, \dots, M, \forall j = 1, \dots, N\}$. The units of velocity vectors are in meters per second, and the space units are in meters.

Therefore, the pixels in the streamline \mathcal{S} have a pair of space coordinates $\mathbf{X}' = \{(x'^{(1)}, y'^{(1)}), \dots, (x'^{(L)}, y'^{(L)}) \in \mathbb{R}^2 \mid \forall \ell = 1, \dots, L\}$ in the atmosphere cross-section plane. The dimension of the pixels in the atmosphere cross-section plane are $\Delta \mathbf{X}' = \{(\Delta x'^{(1)}, \Delta y'^{(1)}), \dots, (\Delta x'^{(L)}, \Delta y'^{(L)}) \in \mathbb{R}^2 \mid \forall \ell = 1, \dots, L\}$, and their respective wind velocity field are $\hat{\mathbf{W}}' = \{(\hat{u}'^{(1)}, \hat{v}'^{(1)}), \dots, (\hat{u}'^{(L)}, \hat{v}'^{(L)}) \in \mathbb{R}^2 \mid \forall \ell = 1, \dots, L\}$.

Once we know the sequential order of the pixels in the intercepting streamline, their dimensions, and their velocity field components, we want to estimate the time $t^{(\ell)}$ when a pixel will intersect the Sun. However, the estimation of $t^{(\ell)}$ is not certain because the approximation of a velocity vector in the wind velocity field has an error.

The root mean squared error represent the uncertainty of a wind velocity vector in the u and v velocity component is $\mathbf{e} = \{e_{\hat{u}}, e_{\hat{v}}\}$. We have that the intersecting time $t^{(\ell)}$ of a pixel in the streamline is given by the classical equation with plus an uncertainty,

$$t^{(\ell)} + e_t = \left[\frac{a_x^{(\ell)} (\Delta x'^{(\ell)})^2 + a_y^{(\ell)} (\Delta y'^{(\ell)})^2}{(\hat{u}'^{(\ell)} + e_{\hat{u}})^2 + (\hat{v}'^{(\ell)} + e_{\hat{v}})^2} \right]^{1/2}, \quad \forall \ell = 1, \dots, L, \quad (8.38)$$

where $a_x^{(\ell)}$ and $a_y^{(\ell)}$ are coefficients such as $a_x^{(\ell)}, a_y^{(\ell)} \in \{0, 1\}$. Their value is 1, if the transition to a neighboring pixel in connected component algorithm involves a translation in the x-axis ($a_x^{(\ell)}$) or y-axis ($a_y^{(\ell)}$). Otherwise, the coefficient value is 0.

We draw S independent samples from $e_{u,i} \sim \mathcal{N}(0, e_u)$ and $e_{v,i} \sim \mathcal{N}(0, e_v)$ to infer the distribution of e_t . If the intercepting times $t^{(\ell)}$ along the streamlines are considered gamma random variables $t^{(\ell)} \sim \mathcal{G}(\alpha^{(\ell)}, \beta^{(\ell)})$, the MLE of gamma distribution parameters can be numerically approximated as [234],

$$\alpha^{(\ell)} \approx \frac{0.5}{\left(\log \frac{1}{S} \sum_{i=1}^S t_i^{(\ell)} \right) \left(\frac{1}{S} \sum_{i=1}^S \log t_i^{(\ell)} \right)}, \quad \beta^{(\ell)} = \frac{\frac{1}{S} \sum_{i=1}^S t_i^{(\ell)}}{\alpha^{(\ell)}}. \quad (8.39)$$

Consequently, the distributions when an air parcel will intersect the Sun is given by

$$\hat{t}^{(\ell)} \sim \mathcal{G} \left(\sum_{\ell'=1}^{\ell} \alpha^{(\ell')}, \beta^{(\ell)} \right), \quad \ell = 1, \dots, L, \quad (8.40)$$

which is the sequential cumulative sum of gamma random variables with same rate β parameter. The parameters $\beta^{(\ell)}$ are not the same in the approximated gamma distributions but they are very close to each other, so we approximated them as the same for all gamma distributions.

Therefore, the probability $p_{int}(t_c | \alpha^{(\ell)}, \beta^{(\ell)})$ of a pixel in the streamline intersecting the Sun at a time t_c is

$$p_{int}(t_c | \alpha^{(\ell)}, \beta^{(\ell)}) \triangleq w_c^{(\ell)} = \frac{\beta^{(\ell) \alpha^{(\ell)}}}{\Gamma(\alpha^{(\ell)})} t_c^{\alpha^{(\ell)} - 1} e^{-\beta^{(\ell)} t_c}, \quad (8.41)$$

where t_c is the time ahead of a forecasting horizon $t_c = \{t_1, \dots, t_C\}$. Once the uncertainty along the intersecting position in the streamline is known for each forecasting horizon t_c , the uncertainty in the intersecting position are computed in the 2-dimensional coordinate system of the atmosphere cross-section plane,

$$\begin{aligned}\mathbb{E}_{p_{int}(t_c)}[\mathbf{x}'] &= \bar{\mathbf{x}}_c \approx \sum_{\ell=1}^L \mathbf{x}'^{(\ell)} w_c^{(\ell)} \Delta \bar{t}^{(\ell)}, \\ \mathbb{C}_{p_{int}(t_c)}[\mathbf{x}'] &= \mathbf{S}_c \approx \sum_{\ell=1}^L \mathbf{x}'^{(\ell)} \mathbf{x}'^{(\ell)\top} w_c^{(\ell)} \Delta \bar{t}^{(\ell)} - \bar{\mathbf{x}}_c \bar{\mathbf{x}}_c^\top,\end{aligned}\tag{8.42}$$

where $\Delta \bar{t}^{(\ell)} = (\bar{t}^{(\ell+1)} - \bar{t}^{(\ell)})$, and $\bar{t}^{(\ell)} = \alpha^{(\ell)} \beta^{(\ell)}$ is the expected time to traverse pixel ℓ .

In addition to the uncertainty in the intersecting position in the streamline, it exists an uncertainty $e_{\Delta \mathbf{x}'}$ in the displacement due to the uncertainty in the estimation of the wind velocity field. The uncertainty of the displacement in the 2-dimensional space of the atmosphere cross-section plane for each forecasting horizon c is assumed a multivariate normally distributed random variable $e_{\Delta \mathbf{x}', c, i} \sim \mathcal{N}(\mathbf{0}, e_{\Delta \mathbf{x}', c} \mathbf{I})$. We proposed to estimated the uncertainty $e_{\Delta \mathbf{x}', c}$ numerically

$$e_{\Delta \mathbf{x}', c} \approx \frac{1}{S} \sum_{i=1}^S t_c^2 (e_{\hat{v}, i}^2 + e_{\hat{u}, i}^2),\tag{8.43}$$

drawing S independent samples from $e_{\hat{u}, i} \sim \mathcal{N}(0, e_{\hat{u}})$ and $e_{\hat{v}, i} \sim \mathcal{N}(0, e_{\hat{v}})$.

As the uncertainty in the intersecting position and the uncertainty on the displacement are both assumed i.i.d. Normal random variables, the probability of a pixel intersecting the Sun is computed given by their sum. The sum of two i.i.d. Normal random variables is the Normal distribution obtained from the convolution of the two Normal distributions, which is

$$p(\mathbf{x}_{i,j} | \bar{\mathbf{x}}_c, \bar{\mathbf{S}}_c) \sim \mathcal{N}(\bar{\mathbf{x}}_c, \mathbf{S}_c + e_{\Delta \mathbf{x}', c} \mathbf{I}),\tag{8.44}$$

and it is computed for each time t_c . Therefore, the probability for any pixel in an image $\mathbf{x}_{i,j}$ intersecting the Sun at a time t_c is

$$p(\mathbf{x}_{i,j} | \bar{\mathbf{x}}_c, \bar{\mathbf{S}}_c) = \frac{1}{(2\pi)^{1/2} |\bar{\mathbf{S}}_c|^{1/2}} \exp \left[-\frac{1}{2} (\mathbf{x}_{i,j} - \bar{\mathbf{x}}_c)^\top \bar{\mathbf{S}}_c^{-1} (\mathbf{x}_{i,j} - \bar{\mathbf{x}}_c) \right].\tag{8.45}$$

This is because the gamma distribution parameters $\alpha^{(\ell)}$ in Eq. (8.40) are sufficiently large, so the gamma distribution can be approximated by a normal distribution.

The probability of a pixel being in the intersecting streamline $p(\mathbf{x}_{i,j}|\Phi, \Psi)$ is computed using the equations in B. Finally, the probability of a pixel intersecting the Sun and being in the intersecting streamline is computed as

$$z_{i,j,c} \triangleq p(\mathbf{x}_{i,j}|\bar{\mathbf{x}}_c, \bar{\mathbf{S}}_c) \cdot p(\mathbf{x}_{i,j}|\Phi, \Psi), \quad (8.46)$$

this are the probabilities used to weight the features in each pixel.

Statistical Features and Data Structure

The features of clouds are extracted in each IR image and their first and second statistical sample moments are computed, quantifying them. The sample moments of the third (i.e., skewness) and fourth (i.e., kurtosis) order were explored but the experiments did not show any improvement in the prediction. The feature vectors obtained after the statistical quantification are added to the database.

The features of the cloud are quantified applying more importance to the pixels that are more likely to intersect the Sun. The importance weights are the previously computed probabilities $z_{i,j,c}$. The weighted sample first moment is,

$$m_c^X = \frac{\sum_{i=1}^M \sum_{j=1}^N z_{i,j,c} x_{i,j}}{\sum_{i=1}^M \sum_{j=1}^N z_{i,j,c}}. \quad (8.47)$$

where $x_{i,j,c}$ represents the value of any feature X in pixel i, j . In an analogous manner, the weighted second moment $s_{X,c}$ of any given feature X for horizon c is,

$$s_c^X = \left[\frac{\sum_{i=1}^M \sum_{j=1}^N z_{i,j,c} (x_{i,j} - m_c^X)^2}{\sum_{i=1}^M \sum_{j=1}^N z_{i,j,c}} \right]^{1/2}, \quad (8.48)$$

where $m_{X,c}$ is the weighted sample first moment of feature X for horizon c .

The exogenous features extracted from the cloudy pixels in image k are: temperature $T_{i,j}$, height $H_{i,j}$, cloud velocity vectors magnitude $M_{i,j}$, divergence $D_{i,j}$, and curl $V_{i,j}$. Their quantified statistics are the means and standard deviations,

$$\begin{aligned} \mathbf{t}_k &= \begin{bmatrix} m_{T,1} & s_{T,1} \\ \vdots & \vdots \\ m_{T,C} & s_{T,C} \end{bmatrix}; \quad \mathbf{h}_k = \begin{bmatrix} m_{H,1} & s_{H,1} \\ \vdots & \vdots \\ m_{H,C} & s_{H,C} \end{bmatrix}; \quad \mathbf{m}_k = \begin{bmatrix} m_{M,1} & s_{M,1} \\ \vdots & \vdots \\ m_{M,C} & s_{M,C} \end{bmatrix}; \\ \mathbf{d}_k &= \begin{bmatrix} m_{D,1} & s_{D,1} \\ \vdots & \vdots \\ m_{D,C} & s_{D,C} \end{bmatrix}; \quad \mathbf{v}_k = \begin{bmatrix} m_{C,1} & s_{C,1} \\ \vdots & \vdots \\ m_{C,C} & s_{C,C} \end{bmatrix} \end{aligned} \quad (8.49)$$

In addition to the statistics, the feature vector in the database include endogenous irradiance measurements,

$$\mathbf{y}_k = [y_{k-1} \ \dots \ y_{k-\varrho}]^\top \in \mathbb{R}^\varrho, \quad (8.50)$$

ϱ is defined as the lag in the time series, and the Sun elevation and azimuth angles at the time when the IR image k was recorded,

$$\mathbf{a}_k = [\varepsilon \ \alpha]^\top \in \mathbb{R}^2. \quad (8.51)$$

The different features vectors are concatenated together, so that a feature vectors k in the database is defined as,

$$\begin{aligned} \mathbf{x}_k &= [\mathbf{y}_k \ \mathbf{a}_k \ \mathbf{t}_{1,k} \ \dots \ \mathbf{t}_{C,k} \ \mathbf{h}_{1,k} \ \dots \ \mathbf{h}_{C,k} \ \dots \\ &\quad \dots \ \mathbf{m}_{1,k} \ \dots \ \mathbf{m}_{C,k} \ \mathbf{d}_{1,k} \ \dots \ \mathbf{d}_{C,k} \ \mathbf{v}_{1,k} \ \dots \ \mathbf{v}_{C,k}]^\top \in \mathbb{R}^D, \end{aligned} \quad (8.52)$$

where D represents the number of dimensions of any vector \mathbf{x}_k .

The corresponding multi-task independent variable vector to be predicted in instant k (composed of future CSI measurement) is,

$$\mathbf{y}_{k+1} = [y_1 \ \dots \ y_C]^\top \in \mathbb{R}^C, \quad (8.53)$$

this independent variable vector is also stored in the database.

Henceforth, \mathbf{y}_k is considered a stationary stochastic process defined as $\mathbf{y}_k = \{\mathbf{y}_k : k \in [1, \infty)\} \in \mathbb{R}^C$, that we aim to model using the feature vector \mathbf{x}_k formed by exogenous and endogenous variables.

8.4 Experiments

8.4.1 Image Processing, Feature Extraction and Selection

The IR images were processed to remove cyclostationary process produced by solar radiation, and particles on the germanium outdoor camera window. The heights were computed applying the MALR to the precessed temperatures (see Section 3.2.2). The method to estimate the cloud velocity vectors is a weighted implementation of the WLK algorithm (see Chapter 6). The images were normalized to avoid intensity fluctuations that may affect the accuracy of the velocity vectors (see Chapter 3 for more information about the image normalization and processing applied to the IR images). The cloud velocity vectors were used to compute the divergence and curl (see Section 8.3.1). The features extracted from IR images are shown in Figure 8.1.

The probability that a pixel in an image will intersect with the Sun is computed for various forecasting horizons using the extracted features. Two examples used in the selection algorithm are shown in Figure 8.2, which corresponds to the clouds in Figure 8.1. See more examples in Chapter 9.

The feature extraction and selection algorithm was applied to consecutive sequences of images acquired on 52 different days. The consecutive sequences show different sky conditions each day. An atmospheric condition model categorizes the IR sky images among the categories of clear sky, cumulus, stratus and nimbus clouds (see Chapter

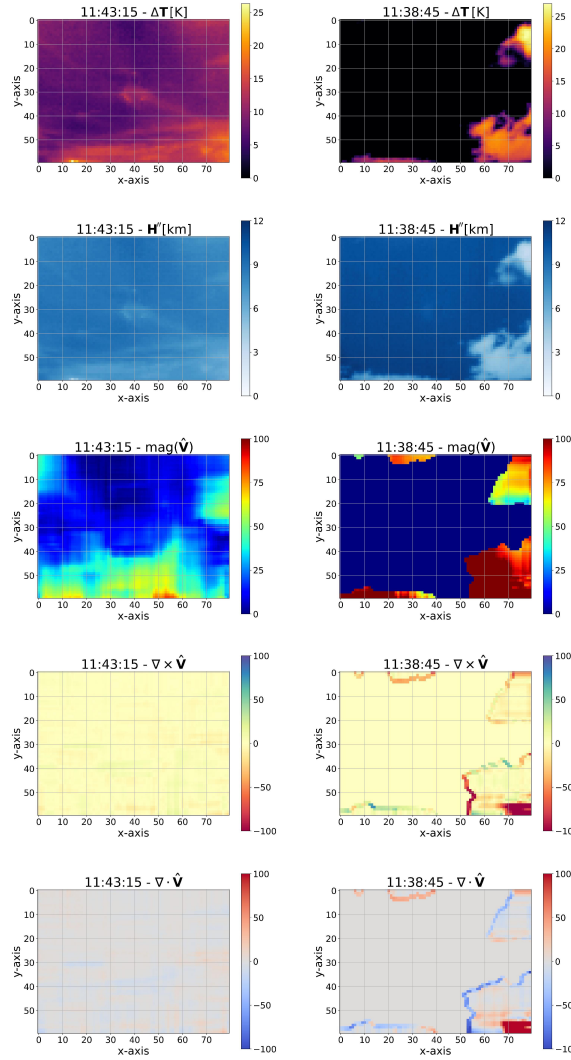


Figure 8.1: Features extracted from clouds in two different days of the same season at approximately the same hour of the day (i.e., the Sun’s elevation and azimuth angle are approximately the same). The features in the left column were extracted from a slow evolving stratus cloud. The features in the right column were extracted from a fast evolving cumulus cloud. From top to bottom, the features are: processed temperature, height, velocity vector magnitude, curl and divergence.

3). The days with sequences of images only showing clear sky or nimbus clouds were avoided in this dissertation since these are not of interest for solar energy generation.

8.4.2 Training and Testing Datasets

Out of the entire dataset, 80% percent of the data was used for training (36,932 samples) and validation purposes while the remaining 20% (9,233 samples) were used for testing. In other words, out of the 52 days in the dataset, 44 days were used for training and 8 days for testing. The samples in the dataset were grouped in the four categories of the atmospheric conditions model. The testing samples include 3,248 clear sky samples, 2,143 cumulus clouds samples, 1,596 stratus clouds samples and 2,246 nimbus clouds (i.e., large thick clouds) samples. The amount of data used in kernel learning methods is prohibitive due to the matrix inversion operation involved in GPRs, KRRs and RVMs. For comparison purposes the amount of data in the training was the same for each model (3,500 samples). Multiple output models that use the Kronecker product require the inversion of a $NC \times NC$ dimensions matrix, where N is the number of samples and C is the number of forecasting horizons. For this reason, the number of samples in the experiments carried out using the MT-KRR, MT-GPR, MT-RVM, and ε -MT-SVM is 2,500. A model is trained using only samples of the same sky condition. Therefore, there are 4 expert models per each implemented kernel learning method. For each testing sample, the class is detected first (i.e., clear sky, stratus, cumulus, or nimbus clouds), and then the corresponding expert model is used for the forecast.

8.4.3 Data Preprocessing

After dividing the training set to train the expert models in different sky condition categories, outliers in each category are removed. The method applied to detect outliers is the Local Outlier Factor (LOF) [40], based on a k-nearest neighbour algorithm, whose optimal number of neighbors was cross-validated, and found to be 3. The first 3,500 samples with higher negative outlier factor were selected.

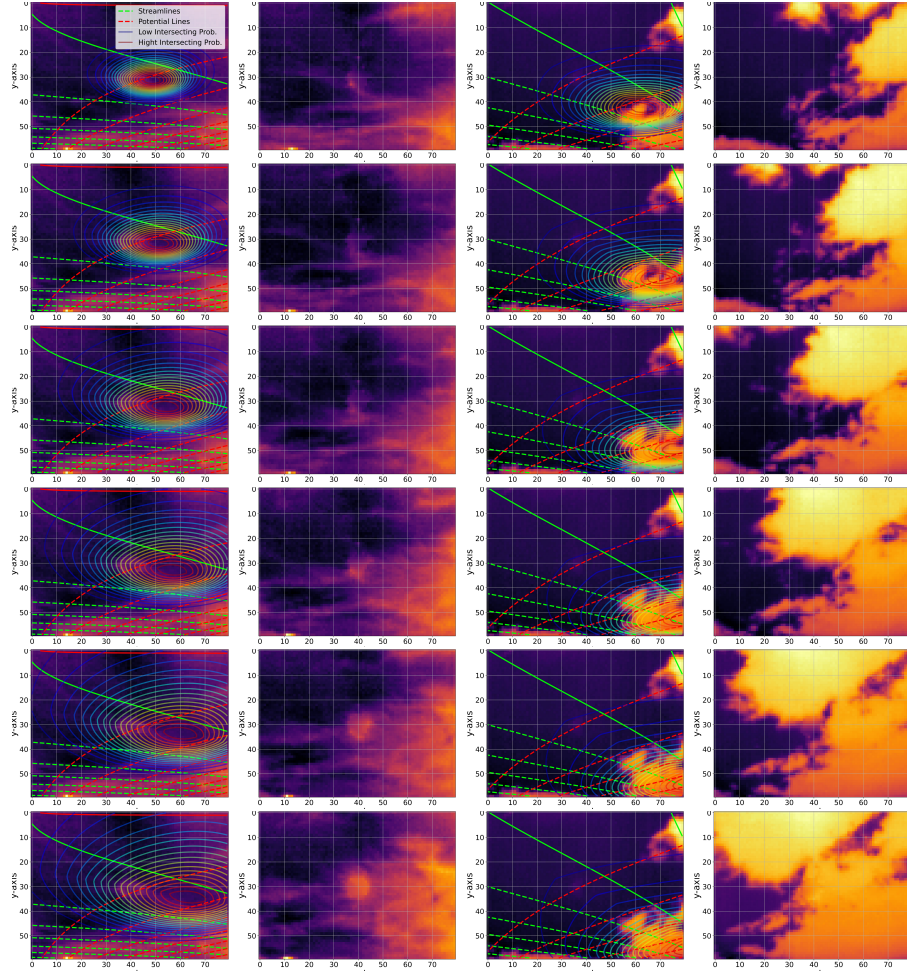


Figure 8.2: Demonstration of the feature selection algorithm in clouds moving in low and high turbulent wind flows (see the extracted cloud features in Fig. 8.1). The streamlines (green) and potential lines (red) were computed for the frame shown in the first and third columns respectively. The probability of a pixel intersecting with the Sun (contours in blue to red color gradient) is computed for the proposed forecasting horizons (3, 4, 5, 6, 7 and 8 minutes ahead). The IR images in the second and fourth columns show the frames in the sequence that correspond to the forecasting horizon displayed in the image next to it in the same row.

The feature $\mathbf{x}_{i,j}$ and predictor $\mathbf{y}_{i,j}$ vectors were standardized as $\bar{\mathbf{x}}_{i,j} = [\mathbf{x}_{i,j} - \mathbb{E}(\mathbf{X})]/\mathbb{V}^{1/2}(\mathbf{X})$ and $\bar{\mathbf{y}}_{i,j} = [\mathbf{y}_{i,j} - \mathbb{E}(\mathbf{Y})]/\mathbb{V}^{1/2}(\mathbf{Y})$. However, the feature vectors $\mathbf{x}_{i,j}$ used in a polynomial kernel did not require standardization, nor did the predictors

$\mathbf{y}_{i,j}$ used in the GPRs and RVMs (i.e., Bayesian models).

8.4.4 Hyperparameters Cross-Validation

The cross-validation routine implemented was a 3-fold validation method. The implementation of another more intensive validation method was not possible due to computational time constraints. A grid search was performed to validate all possible sets of parameters. The size of the grid was 4. The parameters of the model and hyperparameters of the kernel are cross-validated for the KRRs, RVMs and ε -SVMs. In the case of GPRs, the noise variance parameter $\sigma_{c,n}^2$ and Σ_n in Eq. (8.13) and Eq. (8.15) respectively, and the hyperparameters of the kernel (see Section 8.2) are optimized via gradient maximization of the MLL in Eq. (8.14) and Eq. (8.16). The optimal correlation matrix coefficients in Eq. (8.15) of the MT-GPR model are also found by maximizing the MLL. The forecast test results in MAPE performed by independent GPRs, chain of GPR and MT-GPR are shown in Figure 8.3. The experiments have been conducted for different combinations of features. Each feature vector used in the figure is represented by a symbol ϕ with superindexes denoting the different features as C : CSI, A : elevation and azimuth angles, T : raw temperature, T'' : processed temperature, H'' : processed height M : magnitude V : curl, and D : divergence of the velocity vectors.

The regularization parameter $\gamma_c \mathbf{I}_{N \times N}$ in the KRR in Eq. (8.5) and the kernel hyperparameters (see Subsection 8.2) require cross-validation for each forecasting horizon c . However, the regularization parameter $\gamma_c \mathbf{I}_{CN \times CN}$ in the MT-KRR in Eq. (8.8) is simplified as $\gamma_1 = \dots = \gamma_C = \gamma$ in order to reduce the computational cost of the cross-validation procedure. Similarly, the parameters in the correlation matrix $\mathbf{\Gamma}_{C \times C}$ in Eq. (8.9) are also simplified as explained in Subsection 8.2.4. Figure 8.3 shows the testing MAPE obtained by independent KRRs, a chain of KRRs and MT-KRR.

The complexity \mathcal{C} and ε parameter in Eq. (8.19), plus the hyperparameters of the kernel in the ε -SVMs require cross-validation for each forecasting horizon. However, the parameter \mathcal{C} and ε in Eq. (8.22), and the hyperparameters of the kernel in the ε -MT-SVM are the same for all forecasting horizons (see Subsection 8.2.2). Notice that the ε -MT-SVM have different biases b_c (see Eq. (8.23)) for each forecasting horizon as the independent ε -SVMs and the chain of ε -SVMs. The correlation matrix $\mathbf{\Gamma}_{C \times C}$ in Eq. (8.23), is simplified in the same way proposed for the MT-KRR. The testing results achieved by independent ε -SVMs, chain of ε -SVMs and ε -MT-SVM are in Figure 8.4.

RVMs do not require the cross-validation of any model parameters (see Subsection 8.2.2). Nevertheless, a convergence criterion has to be established to determine when the optimal relevance vectors were found. The converge criterion implemented in this chapter consists of stopping the optimization when a new minimum in the sum of squared residuals is not achieved after 25 iterations. The kernel hyperparameters do require cross-validation (see Subsection 8.2) for each forecasting horizon. The MT-RVM parameters of the correlation matrix $\mathbf{\Gamma}_{C \times C}$ in Eq. (8.31) requires cross-validation but the matrix is simplified similarly to MT-KRR and MT-SVM. Figure 8.4 shows the testing results obtained by independent RVMs, chain of RVMs and MT-RVM.

The optimal number of *sources* (i.e., sectors or density functions) included in the feature vectors of each forecast were cross-validated using the 3-Fold validation method. The cross-validation was performed using the 3,500 samples training dataset. The independent GPRs method was chosen as a forecasting model in order to avoid the burden of the model parameters and kernel hyperparameters cross-validation. The optimal indexes of *sources* c' included in the feature vectors $\mathbf{x}_{c',k}^c$ of each forecasting horizon c are: $\mathbf{x}_{c',k}^1$, $c' \in \{1, 2, 3, 4\}$; $\mathbf{x}_{c',k}^2$, $c' \in \{1, 2, 3\}$; $\mathbf{x}_{c',k}^3$, $c' \in \{2, 3, 4\}$, $\mathbf{x}_{c',k}^4$, $c' \in \{3, 4, 5\}$; $\mathbf{x}_{c',k}^5$, $c' \in \{3, 4, 5, 6\}$; and $\mathbf{x}_{c',k}^6$, $c' \in \{5, 6\}$. They were selected for performing

Chapter 8. Kernel Learning for Intra-Hour Solar Forecasting

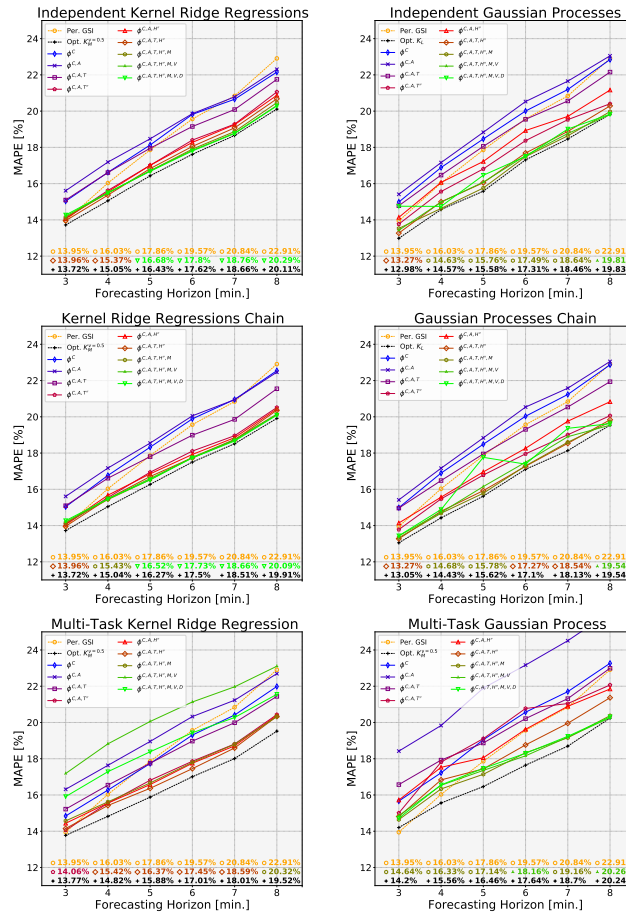


Figure 8.3: Forecasting accuracy achieved by the different multi-task dense kernel learning methods. The graphs in the right column show the results obtained using KRRs (deterministic). The graphs in the left column show the results obtained using GPRs (Bayesian). From top to bottom, the methods implemented to perform a multiple out forecast are multiple independent models, chain of models, and multi-task models. The persistence model (i.e., baseline) is displayed in orange, the colors show the models using different feature vectors (combination of the same model for each sky condition), and the optimal method is displayed in black (combination of the best model for each sky condition). The kernel used by the optimal model is denoted in the legend. In the bottom of the graphs, the MAPE of the baseline (circle in orange), best model (denoted by its corresponding color and marker shape), and optimal model (plus in black) are outlined for each forecasting horizon.

the highest MAPE after averaging the results for across the expert models. The feature vectors $\mathbf{x}_{c',k}^c$ of the multi-tasks models include the indexes of all *sources*

Chapter 8. Kernel Learning for Intra-Hour Solar Forecasting

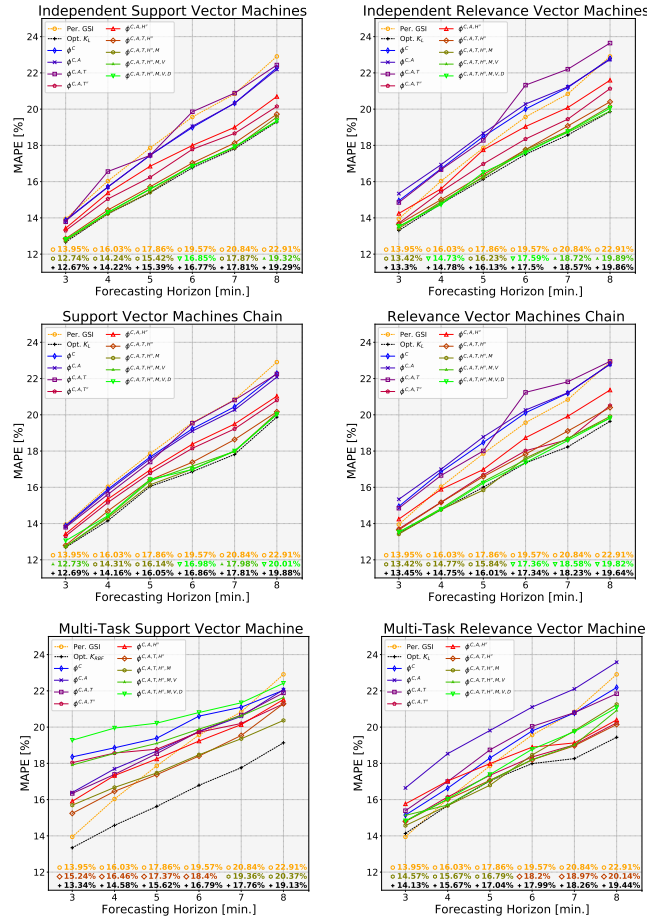


Figure 8.4: Forecasting accuracy achieved using sparse kernel learning methods. The graph on the left shows the performances of the SVMs (deterministic). The graph on the right shows the performances of the RVMs (Bayesian). From top to bottom, the kernel learning methods are graphed as independent models, chain of models, and multi-task models. The persistence model is shown in orange (i.e. baseline). The different colors show the performance for each feature vector when the same model is used in the four different sky conditions. The optimal model (composed of the best model for each sky condition) is shown in black. The kernel used by the optimal model is denoted in the legend. In the bottom of the graphs, the MAPE of the baseline (orange circle), best model (denoted by the corresponding color and marker shape), and optimal model (black plus sign) are outlined for each forecasting horizon.

$c' \in \{1, 2, 3, 4, 5, 6\}$. The feature vectors include features from the CSI, angles, raw temperature and processed height. The procedure to determine the optimal number

of neighbors in the LOF algorithm was the same, but the feature vectors include all forecasting horizons.

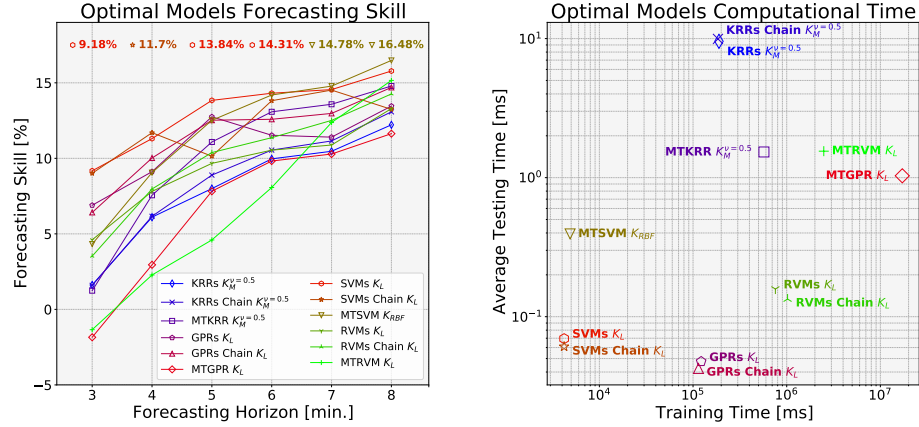


Figure 8.5: Summary of the optimal model's performances. The graph on the left shows the FS achieved by the optimal models in each forecasting horizon. The graph on the right shows the training and testing time of the optimal models. In the case of independent regressions and chain of regressions, the training time is the sum of each sky condition model's training time. The testing time is the average time of the four different models (i.e., sky conditions) when performing a forecast (i.e., 6 forecasting horizons). The kernel used by the optimal models is denoted next to the models' name. In the top of the left graph, the higher FS is outlined using the maker shape and color of the model that achieved it.

The experiments were carried out in the Wheeler HPC of the UNM-CARC, which uses a SGI AltixXE Xeon X5550 at 2.67GHz with 6GB of RAM per core, 8 cores per node, 304 nodes total, and runs at 25 theoretical peak FLOPS. Linux CentOS 7 is installed.

8.5 Discussion

When a cloud is quickly evolving, it has high curl and divergence. This is visible in the cloud dynamics displayed by the cloud in Figure 8.1 (right column). These images

Chapter 8. Kernel Learning for Intra-Hour Solar Forecasting

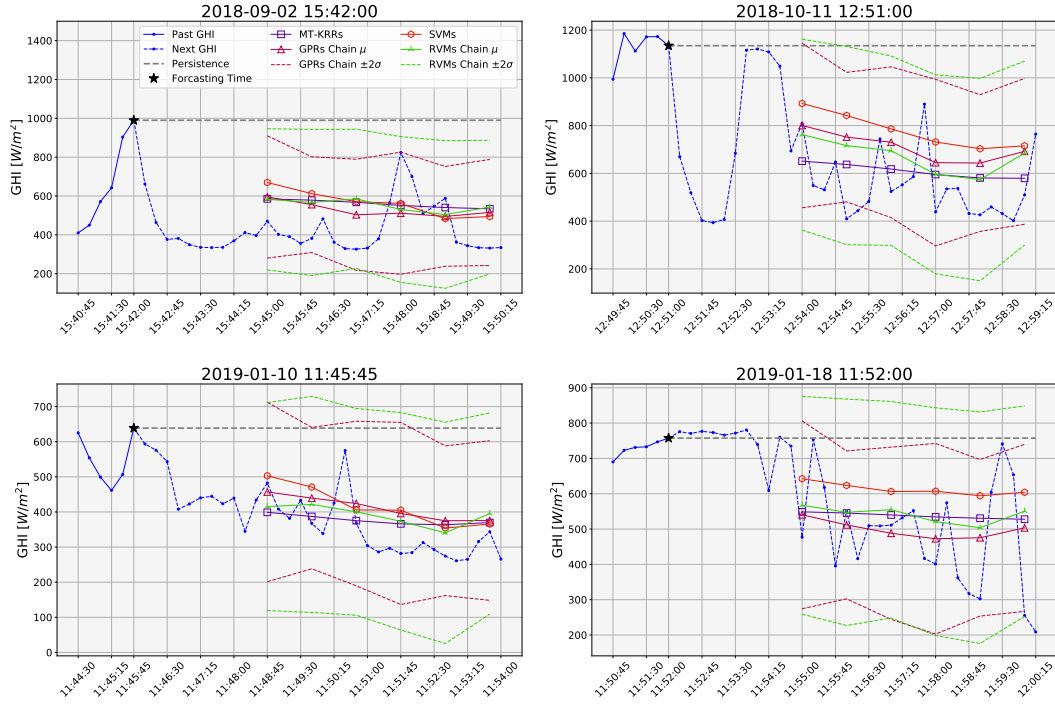


Figure 8.6: Illustration of the forecast performed by the best deterministic models (i.e., independent SVMs and MT-KRR) in purple and red, and Bayesian models (i.e., chain of GPRs and RVMs) in violet and green respectively. The graphs show four different ramp-down events. The forecasting is performed at the time marked by a bright green star. The predictions and confidence intervals correspond to the forecast performed by the different multi-task forecasting models in that ramp-down event. The presented multi-task forecasting models outperform the persistence (i.e., baseline) in anticipation of the arrival of clouds. The persistence is shown in gray dashed lines. The chain of GPRs and RVMs perform a probabilistic forecast, the 95% lower and upper confidence intervals are shown in dashed lines.

belong to the sequence shown in Figure 8.2. As seen this figure, the trajectory of the cloud may be predicted but the shape of the cloud is unpredictable. The forecasting results shown in Figure 8.3 and Figure 8.4 are consistent with this observation. The longer the forecasting horizon is, the more necessary cloud dynamics features are to increase the accuracy of the solar forecasting.

A single multi-task model is advantageous when using KRR or RVM (see Figure

8.3-8.4). When using SVMs, the best practice is to use independent models (see Figure 8.4), and the chain of models is the best multi-task approach when using GPRs (see Figure 8.3). In particular, independent SVMs perform better in shorter forecasting horizons than the chain of GPRs. GPRs is a Bayesian method, thus it is capable of predicting the uncertainty in the forecast. The $\pm 2\sigma$ confidence interval of the Bayesian methods shown in Figure 8.6 was verified in the entire testing dataset and includes $\approx 95\%$ of the forecasting samples. The examples shown in Figure 8.6 are particularly challenging (i.e., ramp-down events), and consequently the confidence intervals are wider than in safer, or more gradual events.

The models with higher Forecasting Skill (FS) are the independent SVMs and the chain of GPRs when all forecasting horizons are averaged, see left graph in Figure 8.5. The advantage of the chain of GPRs is that the forecast has a confidence interval (Figure 8.6). The SVMs require less training time than the rest of models. However, once the independent and chain of GPRs are trained, they require the same time to perform the forecast, (right graph Figure 8.5). The forecasting time required by the multiple output models that use the Kronecker product is higher, because the kernel matrix has more samples. A priori the sparse models (i.e., SVMs and RVMs) should have faster testing time than GPRs (see right graph in Figure 8.5), however the optimal GPRs use feature vectors which have less dimensions see Figure 8.3-8.4.

The FS of the proposed algorithms are 13.85% (independent SVMs) and 16.48% (MT-SVM), 5 and 8 minutes ahead respectively. A previous forecasting method found in the literature proposed 5 and 8 minutes ahead with resolution of 1 minutes and FS of 5.6% and 15.4% respectively [354]. Another method proposed 5 and 10 minutes ahead with resolution of 1 minutes and FS of 14.4% and 12.1% respectively [369]. Similar work, but not using the exact same forecasting horizons in intra-hour solar forecasting using sky images, achieved a FS of 15.7% and 10.91%, 15 and 10 minutes ahead with resolution of 1 minute and 10 minutes respectively [100, 313].

The kernel learning method proposed in this chapter improves the intra-hour solar FS with respect to other methods in the literature. We investigate several forecasting horizons between 3 to 8 minutes. The resolution of the forecast is 1 minute, but it is possible to increase the forecasting resolution to 15 seconds as well as the forecasting interval above 1 minute and below 10 minutes. The fact of forecasting the CSI instead of GSI or PV power yields better performances. Equivalently, adding features extracted from sky images also improves the performances in an intra-hour solar forecasting algorithm. In addition, the application of image processing methods to remove cyclostationary effects from the IR camera increase the performance of solar forecasting.

When approaching problems that involve handling large amounts of data, kernel learning methods are constrained by the matrix operations. The ability of a model to generalize fully depends on the minimization of the structural risk and the completeness of the dataset. Therefore, the formation of a training dataset which contains the samples that best represent the underlying function is critical. For this reason, online learning methods that will find the most adequate samples to re-train the models would be the most suitable approach to this problem. Another limitation of the method proposed in this chapter is the hardware. The FOV of the IR camera is narrow so the maximum feasible forecasting horizon is limited to 10 minutes ahead. In addition, the resolution of the camera is low, and this may affect the accuracy in the approximation of the clouds dynamics [101].

8.6 Conclusion

This chapter presents a comparison of probabilistic and deterministic multi-task intra-hour solar forecasting algorithms based on kernel learning. The forecasting horizon ranges from 3 to 8 minutes ahead with resolution of 1 minute. The forecasting

resolution is adaptable up to intervals of 15 seconds. The solar forecasting algorithm uses novel feature vectors that include previous CSI measurements (i.e., 6 lags), the position of the Sun (i.e., elevation and azimuth angles), and statistics of features extracted from the cloud dynamics (i.e., temperature, height, and velocity vector magnitude, divergence and curl). Another innovation is the introduction of a method to compute the probability of a cloud intersecting the Sun using the approximated potential and streamlines of the wind velocity field. The potential and streamlines are computed using a flow visualization algorithm that approximates the wind velocity field using sequences of consecutive IR sky images with clouds.

The image processing methods applied to raw IR images remove the cyclostationary effects produced by the atmosphere and the camera outdoor germanium window, increasing the forecasting skill of an intra-hour solar forecasting algorithm. The use of the CSI as a predictor rather than the power output or the GSI increases the accuracy of a solar forecast. At the same time, forecasting the CSI allows for the sharing of information and the extrapolation of the forecast between nearby weather stations (i.e., sky imagers).

Convolutional neural networks are the most commonly used ML method but they require optimizing thousands of free parameters. In contrast, the method proposed in this chapter uses the approximation of the wind velocity field streamlines (i.e., pathlines) to anticipate when a cloud will intercept with the Sun. This method is based on fluid mechanics and is feasible in real time. It does not require the inference of the convolutional filters' parameters and it is adaptive to different weather conditions.

The accuracy of an intra-hour solar forecast may be improved by implementing a more efficient outlier removal or sample selection in an online learning algorithm. Other ML methods such as ensemble learning or deep learning (i.e., recurrent neural networks) may be investigated to develop solar forecasting based on the wind velocity field. The comparison of the performances between a convolutional neural network

and the proposed method to anticipate the trajectory of clouds using the same dataset would be of great interest to determine which method is more promising. The most simple development would be to implement a convolutional neural network that uses sky images and GSI processed using the methods proposed in this chapter. Future work may also pursue the extension of the proposed method to sky imagers with a larger field of view to address intra-hour solar forecasting applications from 10 minutes to 1 hour ahead.

Chapter 9

Deep Learning for Intra-Hour Solar Forecasting

9.1 Introduction

In intra-hour forecasting applications, ANN are the most commonly used ML algorithms [260]. In particular, Deep Learning (DL) networks with multiple node layers (i.e., layers of neurons), as CNN, are implemented since they offer the advantage of learning the parameters of the convolutional filters used to extract features from clouds using the gradient [47]. Combining CNNs with Recurrent Neural Networks (RNN) adds the capacity of learning and forgetting (i.e., memory) to the forecasting model, and can also be optimized using the gradient [184, 347].

The most recent publications in intra-hour solar forecasting aim to develop either PV power or GSI forecasting algorithms. A recent investigation of PV power forecasting proposed a CNN to forecast PV power 15 minutes ahead using all-sky images acquired every 15 minutes [312]. The architecture is improved to combine sky images with temporal history [313]. The sampling resolution of the sky imager is 2 minutes. The

features include a series of consecutive sky images and PV output from the last 15 minutes. Another investigation proposed to use data from a sky imager that acquires all-sky visible light at multiple exposures every second. The proposed preprocessing reduces the number of images to five with four different exposure times each minute. The architecture includes a Long Short-Term Memory (LSTM), which is in series with a CNN and parallel to a Multilayer Perceptron (MLP) that uses the past PV power output. The DL model forecasts PV power at 1, 2, 5, and 10 minutes ahead [374].

In GSI forecasting, the proposed methods use data acquired from partial-sky imagers, all-sky imagers, TSI, or multiple weather stations (i.e., sensor grid). Additionally, a sky imager may use a 2-axis solar tracker. In the investigations with a solar tracker, one method uses ground-based partial-sky IR images acquired every 15 seconds. The images are the features used in a CNN, while a series of GSI measurements are the features of LSTM. The models are combined in a MLP to forecast GSI from 15 seconds to 2 minutes ahead with a 15 second resolution [6]. Another method uses a visible light all-sky imager equipped with a Sun blocking mechanism. The sky imager records images every 20 seconds. Five consecutive images are processed to extract features. Afterward, these features and a series of GSI measurements are combined to form the feature vector used in an MLP to forecast GSI from 1 to 5 minutes ahead [172].

In the investigations with static sky imagers, one method uses a series of sky images acquired every minute with an all-sky imager. The images are stacked together to form the features used by a CNN to forecast GSI from 5 to 10 minutes ahead with 1 minute resolution [354]. Another method forecasts GSI 60 minutes ahead with 10 minutes resolution using a dataset six years long, acquired using a TSI. The features used in a CNN are only sky images [100]. The investigation that proposes sensor grids uses multiple weather features from 34 different locations across Texas, USA. A CNN is implemented to find spatiotemporal patterns in the weather features, and an LTSM uses GSI time series from the location of the forecast. Both networks are combined to

forecast GSI at 1 hour ahead [369].

This chapter introduces a novel multi-task DL intra-hour solar forecasting model based on the fusion of information from multiple sensors. A single multi-task architecture instead of different architectures in parallel reduces the necessary computational resources. In addition, this chapter uses data acquired by an IR sky imager on a solar tracker instead of a TSI or an all-sky imager, achieving a higher resolution relative to the dimensions of the air parcel with less saturation in the circumsolar area. Using a thermal imager, in turn, allows for a novel extraction of features and development of a forecast based on information from multiple sources rather than using sky images as the only covariate.

In particular, we propose to extract information about the atmospheric conditions using four sensors: a pyranometer (i.e., GSI measurements), a solar tracker (i.e., Sun position in the sky), a nearby weather station (i.e., weather features), and a ground-based IR sky imager (i.e., cloud features). The information from multiple sensors allows for the extraction of different cloud features. The wind velocity field dynamics are analyzed to compute the probability of an air parcel in the IR images (represented by a pixel) obstructing the Sun’s direct radiation. The computation of an intersecting probability distribution for each forecasting horizon provides different feature sources as an outcome. This method reduces the computational cost of performing a forecast and does not require convolutional filters. The fusion of these sources of information is explored in conjunction with the SOA in DL to develop the optimal deterministic and Bayesian architectures for multi-task intra-hour solar forecasting applications.

9.2 Methods

Fundamental discoveries in the field of neuroscience [271], which describe the shape of neurons, their synaptic connection properties, and the way that these form a

network [206], motivated research efforts towards the development of a mathematical representation of the nervous activity that occurs in the brain [224]. Further insights about the brain's multi-level architecture [155] inspired the construction of computational systems (i.e., ANNs) capable of learning patterns by emulating the information processing structure in biological systems [191]. In particular, DL refers to ANNs composed of multiple layers of neural networks [239].

The organization of the methods section is as follows: the presentation of DL for a deterministic and Bayesian inference of the parameters is in Section 9.2.1, the cross-validation of the structural hyperparameters using Bayesian Optimization (BO) instead of grid search is in Appendix A.1.2, the processing applied to the IR sky images for the extraction of features is explained in Section 3.2.2, and the estimation of the intersecting probability of an air parcel and the structure of the feature vectors is in Section 8.3.2.

9.2.1 Deep Learning

A Neural Network (NN) models an independent variable as a nonlinear function of the dependent variable, this function is parameterized by a set of weights [210],

$$\mathbf{y}_k = \hat{\mathbf{y}}_k + \varepsilon_k = f(\mathbf{x}_k; \mathbf{W}^{(1)}, \dots, \mathbf{W}^{(L)}) + \varepsilon_k, \quad (9.1)$$

ε_k is the error in the approximation $\hat{\mathbf{y}}_k$ of the independent variable \mathbf{y}_k for an arbitrary sample \mathbf{x}_k . $f(\cdot)$ is the estimation function using a DL network, whose set of parameters is $\mathbf{W}^{(\ell)}$ for any given layer ℓ in a network with a total of L layers, $\ell = \{1, \dots, L\}$. In our problem at hand, the series of samples \mathbf{x}_k have a time structure.

Dense Layer

In a dense layer, a hidden variable $\mathbf{h}_k^{(\ell+1)}$, representing the *output* a node or neuron as a function of *input* \mathbf{x}_k , is fully connected to each node $\mathbf{h}_k^{(\ell)}$ of the previous layer ℓ throughout a set of parameters $\mathbf{W}^{(\ell)}$ and bias $\mathbf{b}^{(\ell)}$,

$$\begin{aligned} \mathbf{z}_k^{(\ell+1)} &= \mathbf{W}^{(\ell)} \begin{bmatrix} \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix}, \\ \mathbf{h}_k^{(\ell+1)} &= \phi \left(\mathbf{z}_k^{(\ell+1)} \right), \end{aligned} \tag{9.2}$$

where $\phi(\cdot)$ is an activation function, normally the Rectified Linear Unit (ReLU) function in Eq. (9.11), but other activations are possible, as the identity function (linear), the sigmoid in Eq. (9.9) or the hyperbolic tangent in Eq. (9.10) below. $\mathbf{W}^{(\ell)} \in \mathbb{R}^{D^{(\ell+1)} \times (D^{(\ell)}+1)}$, $\mathbf{h}_k^{(\ell)} \in \mathbb{R}^{D^{(\ell)}}$ and $\mathbf{h}_k^{(\ell+1)} \in \mathbb{R}^{D^{(\ell+1)}}$. $D^{(\ell)}$ represent the number of dimensions of the input variables of layer ℓ . In Eq. (9.2) and henceforth, the biases $\mathbf{b}^{(\ell)}$ are included inside the parameters $\mathbf{W}^{(\ell)}$. A DL model composed of only few dense layers is commonly referred to as MLP.

Recurrent Layers

The main characteristic of an RNN is that the outputs contribute to the inputs [50]. This makes them appropriate in applications where the predicted independent variables may also influence future predictions (i.e., time series analysis and natural language processing). The RNN architectures implemented in this chapter are shown in Figure 9.1.

Simple Recurrent Network (SRN). In a SRN, at each time instant, the recurrent neurons receive as the hidden vector of previous layer $\mathbf{h}_k^{(\ell)}$, and the output vector $\mathbf{h}_{k-1}^{(\ell+1)}$ from previous time instant. In this way, a SRN have a set of weights $\mathbf{W}_h^{(\ell)}$ for

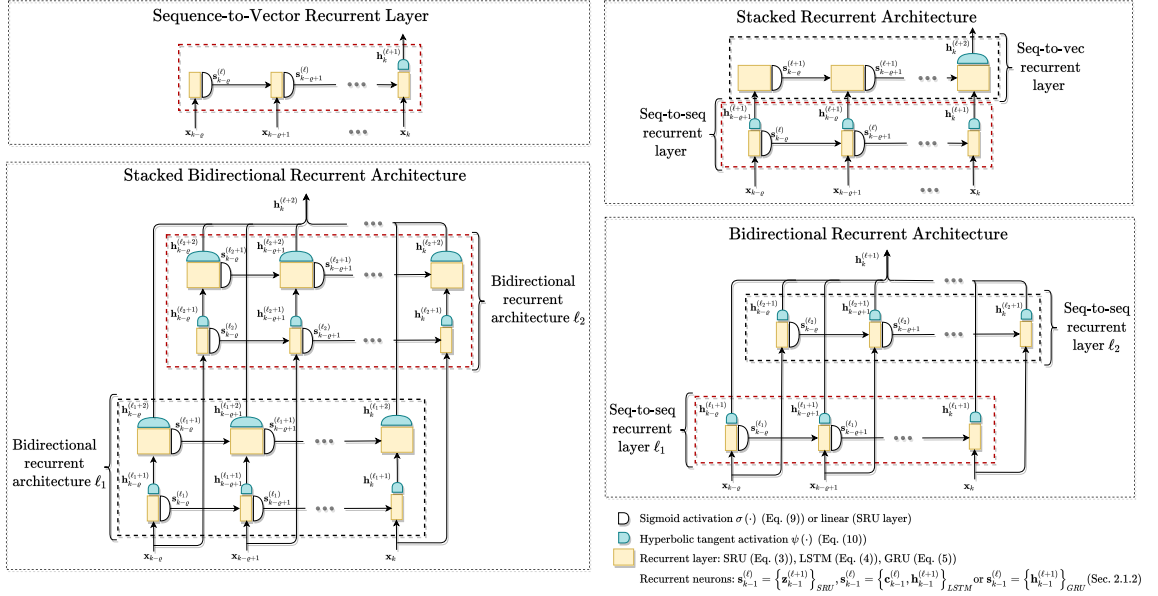


Figure 9.1: This figure shows the schematics of the RNN architectures implemented in the experiments. When there is a single RNN, the architecture used is a no stacked sequence-to-vector (seq-to-vector). When multiple RNNs are stacked, the architecture is sequence-to-sequence first (seq-to-seq) and seq-to-vector afterward. Similarly, the bidirectional architecture is seq-to-seq, and when several RNNs are stacked, the architecture is always seq-to-seq.

the output vector (from a previous time instant), and another weights $\mathbf{W}_z^{(\ell)}$ for the input vector [125],

$$\mathbf{z}_k^{(\ell+1)} = \psi \left(\mathbf{W}_h^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right) \quad (9.3)$$

$$\mathbf{h}_k^{(\ell+1)} = \mathbf{W}_z^{(\ell)} \begin{bmatrix} \mathbf{z}_{k-1}^{(\ell+1)} \\ 1 \end{bmatrix},$$

where $\psi(\cdot)$ is the hyperbolic tangent function, and set of the parameters is $\mathbf{W}^{(\ell)} = \{\mathbf{W}_z^{(\ell)} \in \mathbb{R}^{D^{(\ell+1)} \times (D^{(\ell+1)}+1)}, \mathbf{W}_h^{(\ell)} \in \mathbb{R}^{D^{(\ell+1)} \times (D^{(\ell)}+D^{(\ell+1)}+1)}\}$. The main disadvantage of the SRN is the *vanishing gradient* problem, which consists of that error gets smaller

as it is propagated backwards in the network [145]. Gated NNs (i.e., LSTM [146] and Gated Recurrent Unit (GRU) [60]) were proposed to overcome this problem.

Long Short-Term Memory. LSTM are capable of recognizing input patterns, store them in a *long-term* memory, and preserve them as long as it is necessary, in order to extract features that have a time structure [125]. The architecture of a LSTM is designed to have a main layer $\tilde{\mathbf{c}}_k$ and three gates controllers: the *forget* gate \mathbf{f}_k , the input gate \mathbf{i}_k , and the output gate \mathbf{o}_k . In this way, LSTM have four fully connected layers with their respective set of parameters: $\mathbf{W}_{\tilde{\mathbf{c}}}^{(\ell)}$, $\mathbf{W}_f^{(\ell)}$, $\mathbf{W}_i^{(\ell)}$ and $\mathbf{W}_o^{(\ell)}$,

$$\begin{aligned}
 \mathbf{f}_k^{(\ell)} &= \phi \left(\mathbf{W}_f^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right), \\
 \mathbf{i}_k^{(\ell)} &= \phi \left(\mathbf{W}_i^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right), \\
 \mathbf{o}_k^{(\ell)} &= \phi \left(\mathbf{W}_o^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right), \\
 \tilde{\mathbf{c}}_k^{(\ell)} &= \psi \left(\mathbf{W}_{\tilde{\mathbf{c}}}^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right), \\
 \mathbf{c}_k^{(\ell)} &= \mathbf{f}_k^{(\ell)} \odot \mathbf{c}_{k-1}^{(\ell)} + \mathbf{i}_k^{(\ell)} \odot \tilde{\mathbf{c}}_k^{(\ell)}, \\
 \mathbf{h}_k^{(\ell+1)} &= \mathbf{o}_k^{(\ell)} \odot \psi \left(\mathbf{c}_k^{(\ell)} \right),
 \end{aligned} \tag{9.4}$$

where \odot is the Hadamard product, and $\phi(\cdot)$ is an activation function (usually a sigmoid function). The set of the parameters in layer ℓ is $\mathbf{W}^{(\ell)} = \{(\mathbf{W}_f^{(\ell)}, \mathbf{W}_i^{(\ell)}, \mathbf{W}_o^{(\ell)}, \mathbf{W}_{\tilde{\mathbf{c}}}^{(\ell)}) \in \mathbb{R}^{D^{(\ell+1)} \times (D^{(\ell)} + D^{(\ell+1)} + 1)}\}$.

Gated Recurrent Network. The GRN is a simplified version of the LSTM in which a single controller $\mathbf{c}_k^{(\ell)}$ activates the input and forget gates. In addition, there is no output gate, but there is a *reset* gate $\mathbf{r}_k^{(\ell)}$ that controls which part of the previous state will be fed to the next layer [125],

$$\begin{aligned} \mathbf{o}_k^{(\ell)} &= \phi \left(\mathbf{W}_o^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right), \\ \mathbf{r}_k^{(\ell)} &= \phi \left(\mathbf{W}_r^{(\ell)} \begin{bmatrix} \mathbf{h}_{k-1}^{(\ell+1)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right) \odot \mathbf{h}_{k-1}^{(\ell+1)}, \\ \mathbf{c}_k^{(\ell)} &= \psi \left(\mathbf{W}_c^{(\ell)} \begin{bmatrix} \mathbf{r}_k^{(\ell)} \\ \mathbf{h}_k^{(\ell)} \\ 1 \end{bmatrix} \right), \\ \mathbf{h}_k^{(\ell+1)} &= (1 - \mathbf{o}_k^{(\ell)}) \odot \mathbf{h}_{k-1}^{(\ell+1)} + \mathbf{o}_k^{(\ell)} \odot \mathbf{c}_k^{(\ell)}, \end{aligned} \tag{9.5}$$

where the set of parameters is $\mathbf{W}^{(\ell)} = \{(\mathbf{W}_o^{(\ell)}, \mathbf{W}_r^{(\ell)}, \mathbf{W}_c^{(\ell)}) \in \mathbb{R}^{D^{(\ell+1)} \times (D^{(\ell)} + D^{(\ell+1)} + 1)}\}$.

Residual Layer

Residual learning consists in bypassing a set of layers (i.e., one or more layers) performing an element-wise addition to avoid degradation of the gradient, see Figure 9.2. This problem causes the reduction of the accuracy as the depth of a network increases [138]. In this way, the *identity mapping* allows the construction of deeper networks,

$$\mathbf{h}_k^{(\ell+R)} = f \left(\mathbf{h}_k^{(\ell)}; \mathbf{W}^{(\ell)} \dots \mathbf{W}^{(\ell+R-1)} \right) + \mathbf{W}_{sc}^{(\ell)} \mathbf{h}_k^{(\ell)}, \tag{9.6}$$

where $\mathbf{W}_{sc}^{(\ell)}$ are the shortcut parameters to match the output dimensions. $f(\cdot)$ represent the set of R layers that are being bypassed and $\mathbf{W}^{(\ell)}, \dots, \mathbf{W}^{(\ell+R-1)}$ are their set of parameters.

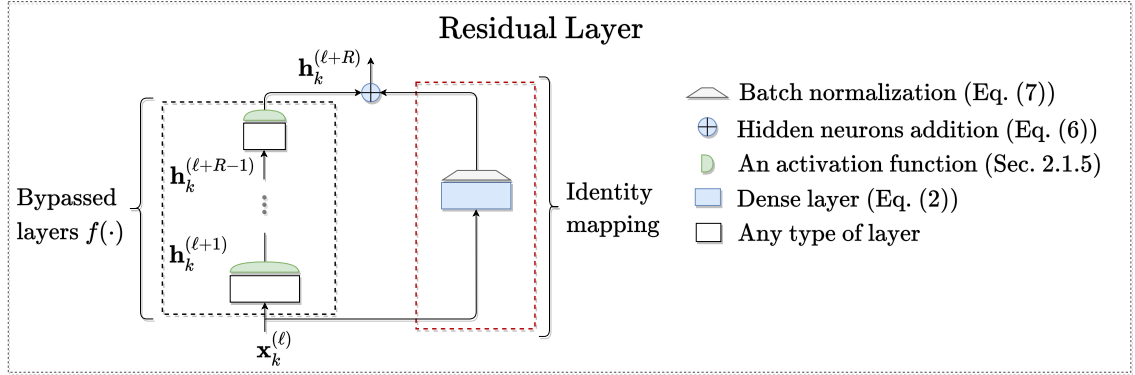


Figure 9.2: Schematic of the identity mapping. The identity mapping bypasses a sequence of consecutive layers. The residual layers include batch normalization.

The application of batch normalization, defined as

$$\mathbf{h}_k^{(\ell+1)} = \gamma \frac{\mathbf{h}_k^{(\ell)} - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2 + \epsilon}} + \boldsymbol{\beta}, \quad \gamma, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \in \mathbb{R}^{D^{(\ell)}}, \quad (9.7)$$

in a residual layer increases the overall performance of the network [69]. γ and $\boldsymbol{\beta}$ are the scale and shift parameters respectively. These parameters are learnt, however ϵ is an small constant value $\epsilon = 0.01$ used for numerical stability. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are the running mean and variance of $\mathbf{h}^{(\ell)}$ computed in each mini-batch (see for more information [69]).

Dropout

Deep NNs trained using a small dataset may suffer from *overfitting*. Dropout can reduce overfitting by randomly removing neurons in the hidden variables $\mathbf{h}^{(\ell+1)}$ [143],

$$\mathbf{w}_q^{(\ell)} = \begin{cases} \mathbf{w}_q^{(\ell)} & \delta \leq u \sim \mathcal{U}(0, 1) \\ \mathbf{0} & \text{otherwise,} \end{cases} \quad (9.8)$$

where δ is the probability that the row q of the weights $w_{q,p}^{(\ell)} \in \mathbb{R}^{D^{(\ell+1)} \times (D^{(\ell)}+1)}$ in layer ℓ is removed.

Activation Functions for Hidden Neurons

The activation functions emulate the firing (i.e., activation) of a biological neuron when it is stimulated by electrical signals received from neurons that are connected to it [147].

Sigmoid. It is the classic activation function in NNs. The sigmoid range is $(0, 1)$, which mimics the states *on* and *off*,

$$\sigma(\mathbf{z}^{(\ell)}) = \frac{e^{\mathbf{z}^{(\ell)}}}{1 + e^{\mathbf{z}^{(\ell)}}}, \quad (9.9)$$

plus it is continuously differential. However, it saturates around 0 or 1 causing the vanishing gradient problem when the network is deep.

Hyperbolic Tangent. This function presents some advantages with respect to the sigmoid,

$$\psi(\mathbf{z}^{(\ell)}) = \frac{e^{\mathbf{z}^{(\ell)}} - e^{-\mathbf{z}^{(\ell)}}}{e^{\mathbf{z}^{(\ell)}} + e^{-\mathbf{z}^{(\ell)}}}, \quad (9.10)$$

its range is $(-1, 1)$, and it is symmetric around zero, so its derivative is steeper (which is better for convergence purposes), but it still suffers vanishing gradient.

Rectified Linear Unit. ReLU activation function is most commonly used in DL,

$$\rho(\mathbf{z}^{(\ell)}) = \max(\mathbf{z}^{(\ell)}, 0), \quad (9.11)$$

the range of this function is $[0, \infty)$. Contrary to the sigmoid and the hyperbolic tangent functions, this activation function avoids vanishing gradient problems and is computationally cheaper (i.e., does not require evaluating an exponential function [115]).

Loss Function

The loss functions in DL mainly aim either the minimization of an error (i.e., distance) or a distribution cross-entropy, $\mathbb{E}[\log p(y|x)]$. The MAE and the MSE are distance measures used to evaluate the accuracy in regression problems. In classification problems, when the distribution of $p(y|x)$ is a Multinoulli distribution, the activation function used is a *softmax*, and when the distribution of $p(y|x)$ is a Bernoulli, the activation function used is a sigmoid [118]. It is worth noting that when the distribution of $p(y|x)$ is Gaussian and the activation function is linear, the minimization of the cross-entropy is equivalent to the minimization of the MSE. The percentual quantification of the MAE, known as MAPE, is an error metric commonly used to evaluate the performance of a forecast. The minimization of the MAPE is consistent with minimizing the empirical risk of a statistical model [70]. In addition, the MAPE is easily interpretable and thus facilitates the comparison of the performances between different models,

$$\mathcal{L}(y_k, \hat{y}_k) = \frac{1}{K} \sum_{k=1}^K \left| \frac{y_k - \hat{y}_k}{y_k} \right|, \quad (9.12)$$

where $\hat{y}_k = f(\mathbf{x}_k; \mathbf{W}^{(1)}, \dots, \mathbf{W}^{(L)})$ is a prediction and K is the number of samples in a set. In this chapter, the MAPE loss function implemented for a multi-task forecasting model is used,

$$\mathcal{L}(y_{k,c}, \hat{y}_{k,c}) = \frac{1}{CK} \sum_{k=1}^K \sum_{c=1}^C \left| \frac{y_{k,c} - \hat{y}_{k,c}}{y_{k,c}} \right|, \quad (9.13)$$

where C is the number of different forecasting horizons in the forecast. The weights are updated with the error gradient computed using *batch* learning and backpropagation.

Backpropagation. Determining the parameters of a NN involves solving an optimization problem that requires evaluating the derivatives of the error functions with

respect to the parameter [31]. The gradient w.r.t. parameters $w_{q,p}^{(\ell)}$ is,

$$\Delta w_{q,p}^{(\ell)} = \frac{\partial \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})}{\partial w_{q,p}^{(\ell)}}, \quad \forall q = 1, \dots, D^{(\ell)}, \quad \forall p = 1, \dots, D^{(\ell-1)}, \quad (9.14)$$

where $\mathcal{L}(\cdot)$ is the loss function and $w_{q,p}^{(\ell)}$ is the parameter corresponding to the connection q, p in layer ℓ . The gradient is computed w.r.t. the weights of each layer by applying the chain rule,

$$\frac{\partial \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})}{\partial w_{q,p}^{(\ell)}} = \frac{\partial \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})}{\partial h_q^{(\ell)}} \frac{\partial h_q^{(\ell)}}{\partial z_q^{(\ell)}} \frac{\partial z_q^{(\ell)}}{\partial w_{q,p}^{(\ell)}}. \quad (9.15)$$

The gradients have an analytical solution and they can be computed using simple matrix multiplications (see for more information [281] and [24]). In this way, the loss function may be optimized using a gradient method such as gradient descent (i.e., steepest descent).

Gradient descent is an iterative optimization algorithm based on the first order derivative. The parameters of layer ℓ are updated in each iteration t applying this formula,

$$\mathbf{W}^{(\ell)(t+1)} = \mathbf{W}^{(\ell)(t)} - \eta^{(t)} \Delta \mathbf{W}^{(\ell)(t)}, \quad t \geq 1 \quad (9.16)$$

where $\eta^{(t)}$ is an adaptive learning rate. When learning is implemented using multiple batches (i.e., stochastic gradient descent) the above partial derivatives formula is substituted by

$$\frac{\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})}{\partial w_{q,p}^{(\ell)}} = \sum_{b=1}^B \sum_{k=1}^{K_b} \frac{\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})_k}{\partial w_{q,p}^{(\ell)}} \quad (9.17)$$

where b is the batch index so that there are B batches of K samples each.

Backpropagation Through Time. In the same way that the parameters of NN are learnt using the backpropagation algorithm, a recurrent network is *unrolled* though

time to apply the backpropagation algorithm to a recurrent layer ℓ . The general formula to compute the time dependency of the gradient is

$$\frac{\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})}{\partial w_{q,p}^{(\ell)}} = \sum_{b=1}^B \sum_{k=1}^{K_b} \frac{\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})_k}{\partial \mathbf{h}_k^{(\ell+1)}} \frac{\partial \mathbf{h}_k^{(\ell+1)}}{\partial \mathbf{z}_k^{(\ell)}} \left[\frac{\partial \mathbf{z}_k^{(\ell)}}{\partial w_{q,p}^{(\ell)}} + \sum_{i=1}^{k-1} \left(\prod_{j=i+1}^k \frac{\mathbf{z}_j^{(\ell)}}{\mathbf{z}_{j-1}^{(\ell)}} \right) \frac{\partial \mathbf{z}_i^{(\ell)}}{\partial w_{q,p}^{(\ell)}} \right], \quad (9.18)$$

where k is a time instant, B is the number of batches, and K_b is the number of time instants in batch n . The gradient $(\partial \mathbf{z}_i^{(\ell)} / \partial w_{q,p}^{(\ell,r)})$ is propagated to each weight matrix in a recurrent layer ℓ applying the chain rule (see for more information [373]).

Bayesian Neural Networks (BNN)

NNs can be trained by backpropagation to approximate the maximum likelihood estimation of the parameters by optimizing the cross-entropy between the distribution of the predictions \mathbf{y}^* and the distribution of actual values \mathbf{y} . Furthermore, the parameters can be regularized by applying a prior distribution to them, which is equivalent to finding their MAP estimation. The Bayesian treatment applied to find the MAP estimation of the NN parameters $\mathbf{W} = \{\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(L)}\}$, requires an approximation of the posterior predictive distribution $p(\mathbf{y}^* | \mathbf{x}^*, \mathcal{D})$ to make a prediction. The posterior predictive distribution is calculated marginalizing the probability of a new sample $p(\mathbf{y}^* | \mathbf{x}^*, \mathbf{W}, \mathcal{D})$ over the posterior distribution of the network parameters, $p(\mathbf{y}^* | \mathbf{x}^*, \mathcal{D}) = \int p(\mathbf{y}^* | \mathbf{x}^*, \mathbf{W}, \mathcal{D}) p(\mathbf{W} | \mathcal{D}) d\mathbf{W}$. However, as DL models are highly nonlinear, the posterior distribution of \mathbf{W} is not Gaussian, so the Bayesian solution (i.e., analytical solution) is not tractable [31]. To overcome this problem, variational inference applies a factorized Gaussian approximation to the posterior distribution $p(\mathbf{W} | \mathcal{D})$.

Probabilistic Loss Function. The loss function is defined as the Negative Log-Probability (NLP) of the data given the set of weights in the network [121],

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_{k=1}^K \log p(\mathbf{y}_k | \mathbf{x}_k, \mathbf{W}^{(1)}, \dots, \mathbf{W}^{(L)}). \quad (9.19)$$

This loss function only models the *aleatory* uncertainty. The *epistemic* uncertainty of parameters is modeled using variational inference [176]. In particular, the aleatoric uncertainty quantifies the variance of the noise in the observations (i.e., likelihood), while the epistemic uncertainty quantifies the uncertainty in the model parameters (i.e., prior).

Variational Inference. As the Bayesian solution of the true posterior distribution $p(\mathbf{W}^{(\ell)} | \mathbf{h}_k^{(\ell+1)}, \mathbf{h}_k^{(\ell)})$ is not available, in variational inference, the objective is to approximate the true posterior distribution by a variational distribution $q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)})$ in each layer ℓ , where $\boldsymbol{\Theta}^{(\ell)} = \{\boldsymbol{\mu}^{(\ell)}, \boldsymbol{\Sigma}^{(\ell)}\}$ are the parameters of a multivariate normal distribution.

The optimal approximation is found by minimizing the Kullback-Leibler (KL) divergence between the variational distribution and true posterior distribution,

$$\begin{aligned} KL \left[q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)}) \parallel p(\mathbf{W}^{(\ell)} | \mathbf{h}_k^{(\ell+1)}, \mathbf{h}_k^{(\ell)}) \right] &= \\ &= \int q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)}) \log \frac{q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)})}{p(\mathbf{W}^{(\ell)} | \mathbf{h}_k^{(\ell+1)}, \mathbf{h}_k^{(\ell)})} d\mathbf{W}^{(\ell)}. \end{aligned} \quad (9.20)$$

Rearranging the terms [294], this is equivalent to minimizing the Evidence Lower Bound (ELBO) (i.e., negative variational free energy),

$$\begin{aligned} \mathcal{V}(\mathbf{h}_k^{(\ell)}, \mathbf{W}^{(\ell)}) &= \mathbb{E}_{\mathbf{W}^{(\ell)} \sim q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)})} \left[\log p(\mathbf{h}_k^{(\ell+1)} | \mathbf{h}_k^{(\ell)}, \mathbf{W}_i^{(\ell)}) \right] \\ &\quad - KL \left[q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)}) \parallel p(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}_0^{(\ell)}) \right], \end{aligned} \quad (9.21)$$

which, in turn, is equivalent to maximize the probability of the data and, at the same time, to minimize the KL divergence between the variational distribution $q(\mathbf{W}^{(\ell)} | \boldsymbol{\Theta}^{(\ell)})$

and the prior distribution $p(\mathbf{W}^{(\ell)}|\boldsymbol{\Theta}_0^{(\ell)})$, $\boldsymbol{\Theta}_0^{(\ell)} = \{\boldsymbol{\mu}_0^{(\ell)}, \boldsymbol{\Sigma}_0^{(\ell)}\}$ are the parameters of a multivariate normal distribution. The KL divergence in the ELBO is just a regularization term (i.e., complexity cost).

Nevertheless, the computation of the integral (i.e., expectation term) in the ELBO function is not analytically tractable, so it requires a Monte Carlo (MC) approximation,

$$\begin{aligned} \mathcal{V}(\mathbf{h}_k^{(\ell)}, \mathbf{W}^{(\ell)}) \approx & \frac{1}{S} \sum_{i=1}^S \left[\frac{1}{MN} \left[\log q(\mathbf{W}_i^{(\ell)}|\boldsymbol{\Theta}^{(\ell)}) - \log p(\mathbf{W}_i^{(\ell)}|\boldsymbol{\Theta}_0^{(\ell)}) \right] \right. \\ & \left. - \frac{1}{K_b} \sum_{k=1}^{K_b} \log p(\mathbf{h}_k^{(\ell+1)}|\mathbf{h}_k^{(\ell)}, \mathbf{W}_i^{(\ell)}) \right]. \end{aligned} \quad (9.22)$$

In the MC approach, S independent parameter samples $\mathbf{W}_i^{(\ell)}$ are drawn from the distribution $q(\mathbf{W}_i^{(\ell)}|\boldsymbol{\Theta}^{(\ell)})$ to compute the ELBO function. The computation of the expected value is numerical, and it is obtained by averaging the value of the ELBO function for each parameter sample.

Bayesian Backpropagation. In a BNN, unlike in backpropagation, the gradient of the loss function is computed applying the chain rule w.r.t. the set of parameters $\boldsymbol{\Theta}^{(\ell)}$ of the variational distribution. The variational inference gradients when using Gaussian distributions are [121],

$$\begin{aligned} \frac{\partial \mathcal{V}(\mathbf{h}_k^{(\ell)}, \mathbf{W}^{(\ell)})}{\partial \boldsymbol{\mu}^{(\ell)}} & \approx \frac{1}{S} \sum_{i=1}^S \left[\frac{\partial p(\mathbf{h}_k^{(\ell+1)}|\mathbf{h}_k^{(\ell)}, \mathbf{W}_i^{(\ell)})}{\partial w_{q,p,i}^{(\ell)}} \right] + \frac{\boldsymbol{\mu}_i - \boldsymbol{\mu}_0}{\boldsymbol{\Sigma}_0}, \\ \frac{\partial \mathcal{V}(\mathbf{h}_k^{(\ell)}, \mathbf{W}^{(\ell)})}{\partial \boldsymbol{\Sigma}^{(\ell)}} & \approx \frac{1}{2S} \sum_{i=1}^S \left[\frac{\partial p(\mathbf{h}_k^{(\ell+1)}|\mathbf{h}_k^{(\ell)}, \mathbf{W}_i^{(\ell)})}{\partial w_{q,p,i}^{(\ell)}} \right]^2 + \frac{1}{2} (\boldsymbol{\Sigma}_0^{-1} - \boldsymbol{\Sigma}_i^{-1}), \end{aligned} \quad (9.23)$$

similarly, S samples of $\mathbf{W}_i^{(\ell)}$ are drawn from $q(\mathbf{W}_i^{(\ell)}|\boldsymbol{\Theta}^{(\ell)})$. When implementing backpropagation using batches the drawing of parameters is performed independently for each batch $\{\mathbf{W}_i^{(\ell)}\}_{b=1}^B$. In our problem at hand, the parameters of the prior

distribution are defined as $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\Sigma}_0 = \mathbf{I}$, and the covariance matrix $\boldsymbol{\Sigma}$ of the variational distribution is the diagonal matrix $\boldsymbol{\Sigma} = \text{diag}(\boldsymbol{\sigma}^2)$ for simplification.

9.3 Experiments

9.3.1 Image Processing, Feature Extraction and Selection of Pixels

The radiation emitted by the Sun, molecules floating in the atmosphere, and sediments stacked to the sky imager's germanium outdoor window produce cyclostationary effects in the IR sky images that are processed out applying the methods described in Section 3.2.2. The approximate height of the cloudy pixels in the sky images is calculated applying the MALR to the processed temperatures (see Section 3.2.2). The cloud velocity vectors were computed using the WLK optical flow (see Chapter 6), and reprojected to the atmosphere-cross section plane in which the clouds are flowing (see Chapter 5 for the geospatial reprojections, and Chapter 7 for the explanation of how to apply the geospatial reprojection to the cloud velocity vectors). The reprojected cloud velocity vectors were used to extract second order features of the cloud dynamics [151]: velocity vectors' magnitude, divergence and curl (see Section 8.3.1). The features extracted from the clouds are shown for three different IR sky images in Figure 9.3.

The probability of an air parcel that is flowing in the intersecting streamline occluding the Sun at time t_c , is computed for the C different forecasting horizons. The probabilities were computed using the wind velocity field and the dimension of the atmosphere cross-section plane in which the clouds are flowing (see Chapter 7). The wind velocity field was approximated using the cloud velocity vectors computed over a sequence of consecutive sky images with clouds, see Figure 9.4. The demonstration of intersecting probability distribution in three different sequences of images is shown

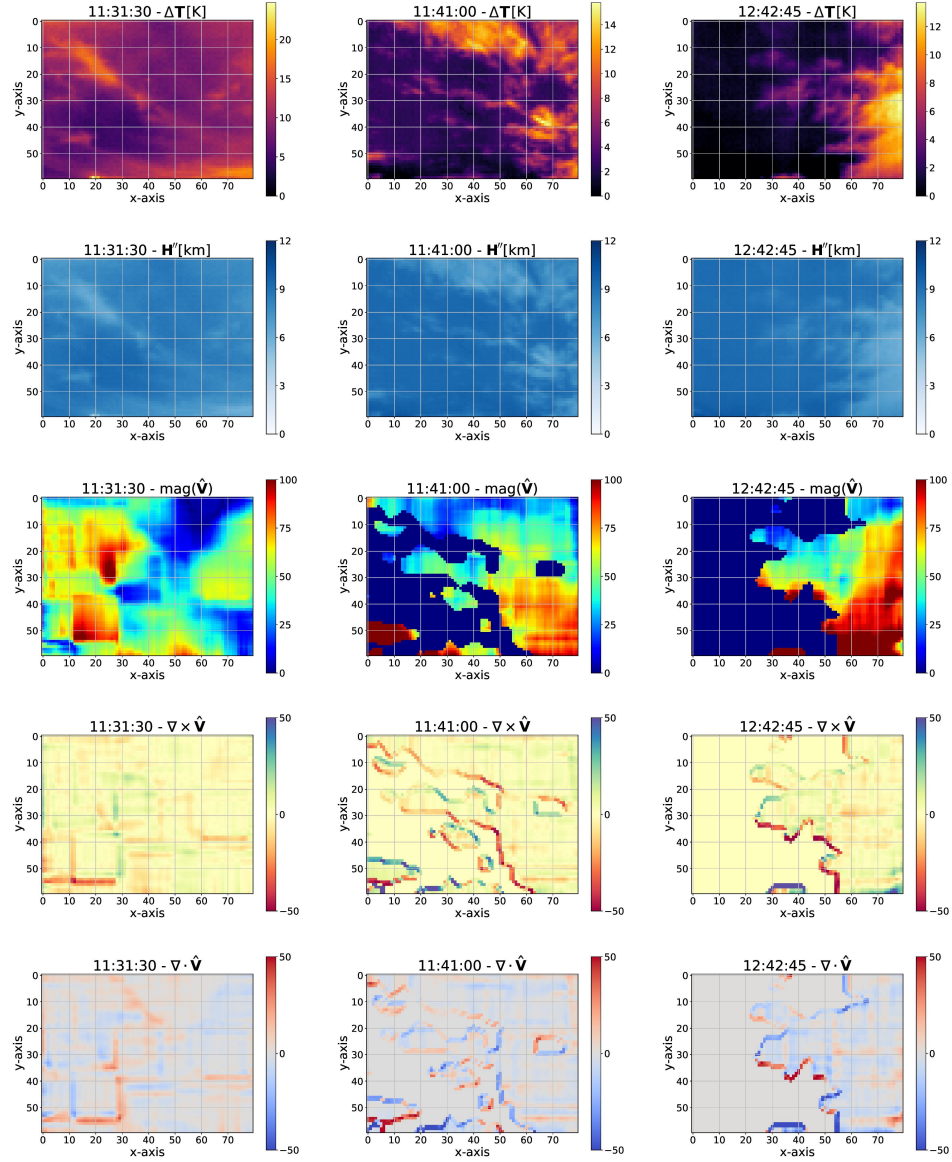


Figure 9.3: This figure shows the features extracted from the IR sky images. The IR sky images in the first, second, and third row in Figure 9.4 correspond to the features in the first, second, and third columns in this figure. The extracted features are in different rows. The first row shows the temperature, the second the height, and the third, fourth, and fifth the velocity vectors' magnitude, curl, and divergence.

in Figure 8.2. See more examples in Chapter 8.

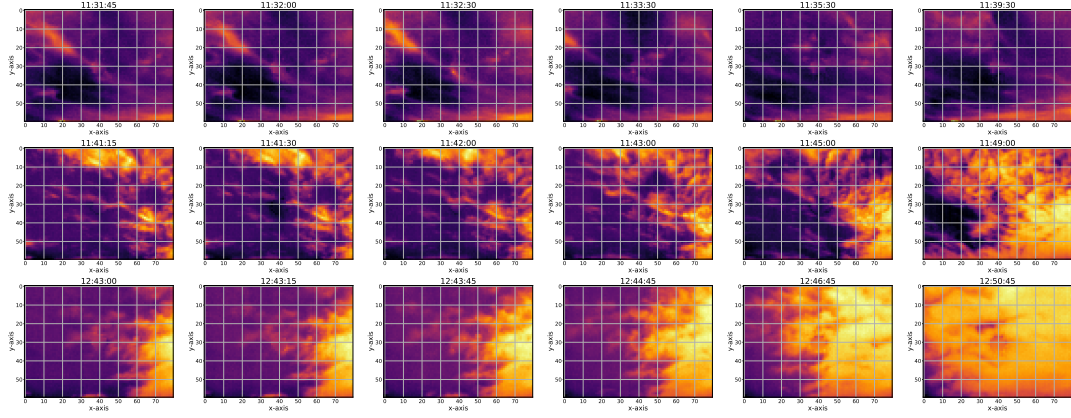


Figure 9.4: This figure shows sequences of IR sky images from three different days. The sky images are organized chronologically from left to right. The features extracted from the first sky image of each sequence are in Figure 9.3. The estimation of the wind velocity field and the probability of an air parcel intersecting the Sun in the sky images are in Figure 9.5. The sky images are shown following an exponential time structure: initial time (first column), after 15 seconds (second column), after 45 seconds (third column), after 105 seconds (fourth column), after 225 seconds (fifth column), after 465 seconds (fifth column).

The proposed methods of image processing and cloud dynamics feature extraction were applied to sequences of consecutive sky images acquired on 52 days. Sky images acquired on days displaying only clear sky or nimbus clouds were avoided since rapid fluctuations in the generation of solar energy are not caused by moving clouds under these sky conditions. The statistics of the features were computed using the probability of a pixel in the intercepting streamline occluding the Sun, $z_{i,j,c}$ in Eq. (8.46). The vectors containing the statistics of the features extracted in each sky image are set together forming the dataset used in this chapter (see Section 8.3.2).

9.3.2 Training and Testing Datasets

The dataset is divided in training 80% (36,932 samples or 44 days) and testing 20% (9,233 samples or 8 day). The training set is further divided in 90% for training

Chapter 9. Deep Learning for Intra-Hour Solar Forecasting

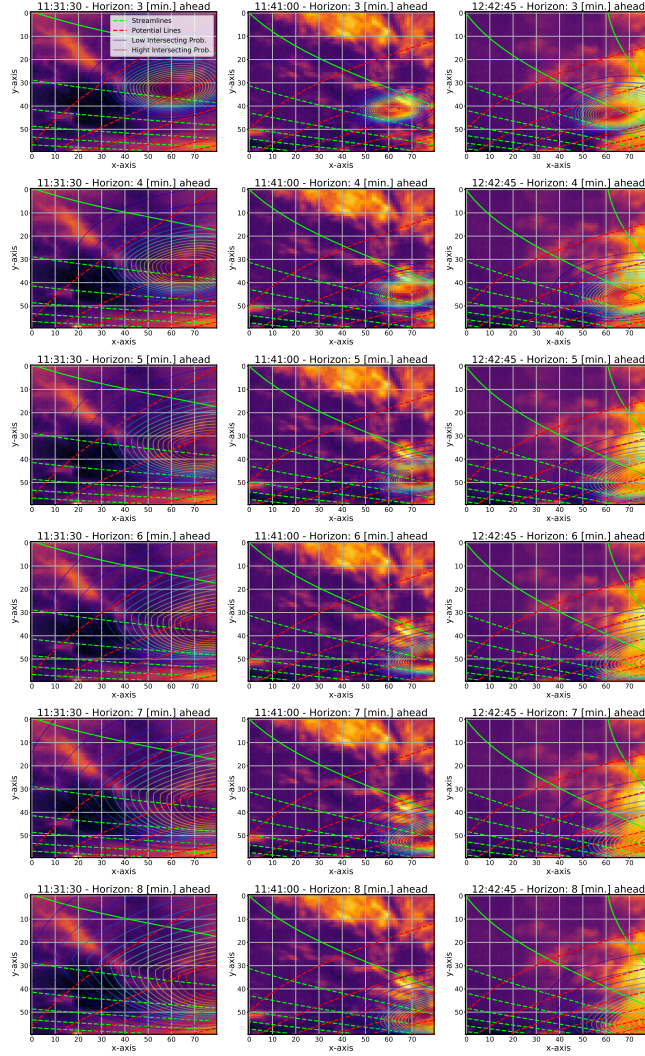


Figure 9.5: The IR images show the approximated wind velocity field streamlines (continuous green lines) and potential lines (dashed red lines). The IR images also show the probability of an air parcel intersecting the Sun in the IR image for the different forecasting horizons (The likelihood is lower as the color is darker and higher when redder). The different forecasting horizons are organized by rows 3, 4, 5, 6, 7, and 8 minutes ahead (from top to bottom row). The evolution of clouds in a sequence of images is in Figure 9.4. Each column in this figure corresponds to each one of its rows.

and 10% validation (i.e., *early stopping*). The structural hyperparameters of the NNs are cross-validated, so the partition corresponding with the training is similarly

subdivided: 90% for training and 10% validation.

The loss function is evaluated with the training set and the validation set. The training set error is propagated backwards using Eqs. (9.14), (9.18) and (9.23), while the validation error is used to assess the convergence. In this way, the validation partition is used as a criterion for early stopping of the training. The training of the NN is stopped early when no new validation loss minima have been achieved after a *patience* of 450 iterations. The parameters of a network are saved every time a minima in the validation loss function is achieved. At the end of the training, the last network saved is loaded for testing purposes.

9.3.3 Automatic Structural Hyperparameter Cross-Validation

A 3-fold validation method was implemented to cross-validate the structural hyperparameters of a network. The implementation of a more intensive validation method was not possible due to computational time constraints. The structural hyperparameters cross-validated were: number of training batches B in Eq. (9.17), learning rate of the gradient descent η in Eq. (9.16), dropout probability δ in Eq. (9.8), the number of neurons $D^{(\ell)}$ in the fully connected layers (see Eq. (9.2)) and recurrent layers (see Eq. (9.3), (9.4) and (9.5)), and the number of stacked recurrent L_R and fully connected layers L_{MLP} (which is $L = L_R + L_{MLP}$ in Eq. (9.1)).

The experiments have been conducted for different combinations of feature sources. The feature vectors used in the figures are represented as \mathbf{x} with superindexes denoting the included features as C : CSI, A : elevation and azimuth angles, T : raw temperature, T'' : processed temperature, H'' : processed height, M : magnitude, V : curl, and D : divergence.

BO is utilized as an alternative to the grid-search method to perform an efficient cross-validation of structural hyperparameters. The idea behind using BO is to avoid

cross-validating structural hyperparameters' domains that are expected to produce no improvement. To do this, the N combinations of structural hyperparameters are generated aleatory and their validation performances are evaluated using 3-fold cross-validation (i.e., validation error). The metric used to evaluate the performances is MAPE. The set of N structural hyperparameters combinations and their corresponding validation errors are used to predict which combination of structural hyperparameters is most likely to produce an improvement in the validation error (see Appendix A.1.2). The validation error achieved by the predicted structural hyperparameters is added to the set of structural hyperparameters combinations (so the set now contains $N + 1$ samples). This process is repeated until M different combinations of most likely optimal structural hyperparameters are explored.

The combination of structural hyperparameters that reach the lowest validation error is selected for training of the network. The parameters of the BO are $\nu = 1.5$ in Eq. (A.38) and $\xi = 10$ in Eq. (A.42). The number of random samples ($N = 45$) and the number of samples ($M = 65$), were selected so that the cross-validation routine and the network training are accomplished within the feasible time (≤ 48 hours).

9.3.4 Data Preprocessing

The feature vectors $\mathbf{x}_{i,j}$ and predictor vectors $\mathbf{y}_{i,j}$ were scaled applying min-max normalization as $\bar{\mathbf{x}}_{i,j} = [\mathbf{x}_{i,j} - \min(\mathbf{X})]/[\max(\mathbf{X}) - \min(\mathbf{X})]$ and $\bar{\mathbf{y}}_{i,j} = [\mathbf{y}_{i,j} - \min(\mathbf{Y})]/[\max(\mathbf{Y}) - \min(\mathbf{Y})]$, so that $\bar{\mathbf{x}}_{i,j}, \bar{\mathbf{y}}_{i,j} \in \mathbb{R}^{[0,1]}$.

The normalization parameters were computed using the training dataset and were applied to the validation and testing dataset. This process was repeated in each iteration of the 3-fold structural hyperparameters cross-validation, and in the training of the model using the cross-validated structural hyperparameters.

Table 9.1: Combination of DL architectures and features vectors analyzed in the experiments. Check symbol means that the experiment was performed, the cross symbol means that it was not performed. The recurrent models are RNNs attached to a multi-task MLP architecture. Similarly, the residual models are RNNs with residual layers attached to a multi-task MLP architecture (ResRNN). In addition, a model with multiple sources or a bidirectional multiple source architecture also includes the AR architecture.

Deep Learning		Feature Vector									
Model	Architecture	C	C,A	C,A,T	C,A,T''	C,A,H''	C,A,T,H''	C,A,T,H'',M	C,A,T,H'',M,V	C,A,T,H'',M,V,D	
MLP	Independent	✓	✓	✓	✓	✓	✓		✓	✓	
	Recursive	✓	✓	✓	✓	✓	✓		✓	✓	
	Multi-Task	✓	✓	✓	✓	✓	✓		✓	✓	
RNN	AR	✓	✓	✓	✓	✓	✓		✓	✓	
	Multiple Source	✗	✗	✗	✗	✓	✓		✓	✓	
	Bidirectional	✗	✗	✗	✗	✓	✓		✓	✓	
ResRNN	AR	✗	✗	✗	✗	✓	✓		✓	✓	
	Multiple Source	✗	✗	✗	✗	✓	✓		✓	✓	
	Bidirectional	✗	✗	✗	✗	✓	✓		✓	✓	
	Bay. Multiple Source	✗	✗	✗	✗	✓	✓		✓	✓	

9.3.5 Deep Learning Architectures

The experiments carried out in this chapter to find the most suitable architecture for intra-hour solar forecasting are summarized in Table 9.1. The performances of the different multi-task MLP architectures are first analyzed to select the most suitable architecture for our problem (see MLP architectures in Figure 9.6). The performances achieved by each MLP architecture are shown in Figure 9.7). The studied multi-task MLP architectures are independent MLPs, recursive MLPs [117], and single multi-task MLPs.

The performance of the most suitable MLP architectures are further analyzed when using different RNN architectures (see Figure 9.8) for each information source (i.e., pyranometer measurements, solar tracker, sky images). First, a backward-in-time RNN (i.e., Auto-Regressive (AR)) is used to model the time series of pyranometer measurements. The depth of RNN architectures is validated in the experiments [122] (see Figure 9.1). Second, a forward-in-time RNN (i.e., multiple source) is used

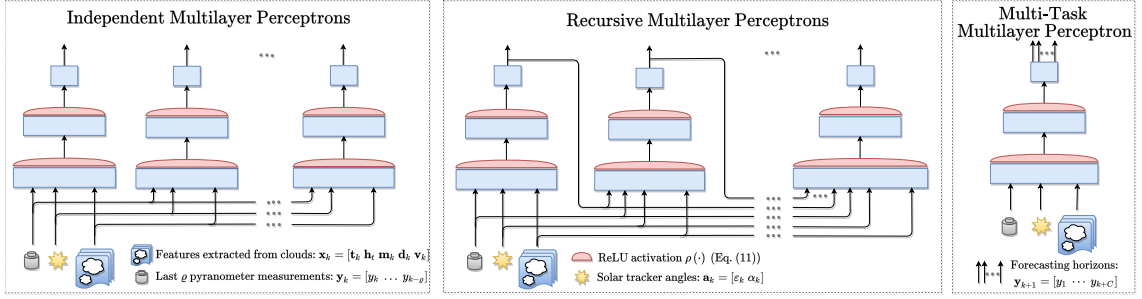


Figure 9.6: Schematics of the different multi-task architectures implemented: Independent MLPs (left), Recurrent MLPs (middle), and a single multi-task MLP (right). All architectures have the same input: cloud features extracted from sky images, the position of the solar tracker (elevation and azimuth angles), and a series of the l past pyranometer measurements. However, the recurrent architecture also includes the forecast of previous horizons.

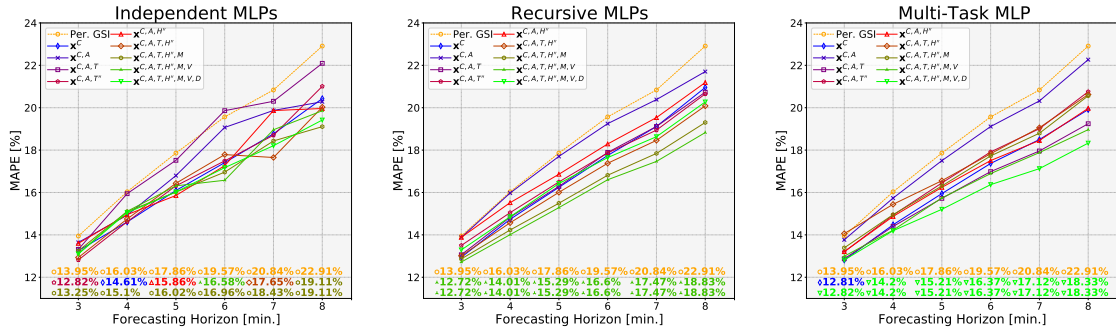


Figure 9.7: Performances of the different multi-task MLP architectures in Figure 9.6. The graph of Independent MLPs is on the left, Recursive MLPs is in the middle, and the single multi-task MLP is on the right. The results of each source of features are displayed using different marker colors. The lines of numbers at the bottom of the graphs highlight the MAPE of the persistence (orange color and circular marker), the lowest MAPE achieved in each forecasting horizon (with the marker color and shape of the best model), and the model with the lowest average MAPE (same marker color and shape).

to model the time structure in features extracted from the sky features for each forecasting horizon. Third, a Bidirectional Recurrent Neural Network (BiRNN) [123] is implemented to analyze whether a time structure exists in features extracted from the sky features (i.e., bidirectional). In each of the experiments besides the network

structural hyperparameters, the type of RNNs was also cross-validated between SRU, LSTM, and GRU. The optimal hyperparameter of the SRU used in the pyranometer measurements \mathbf{x}^C (see Figure 9.8), are used in the experiments of the multiple source and BiRNN. The experiments using the RNNs are shown in Figure 9.9.

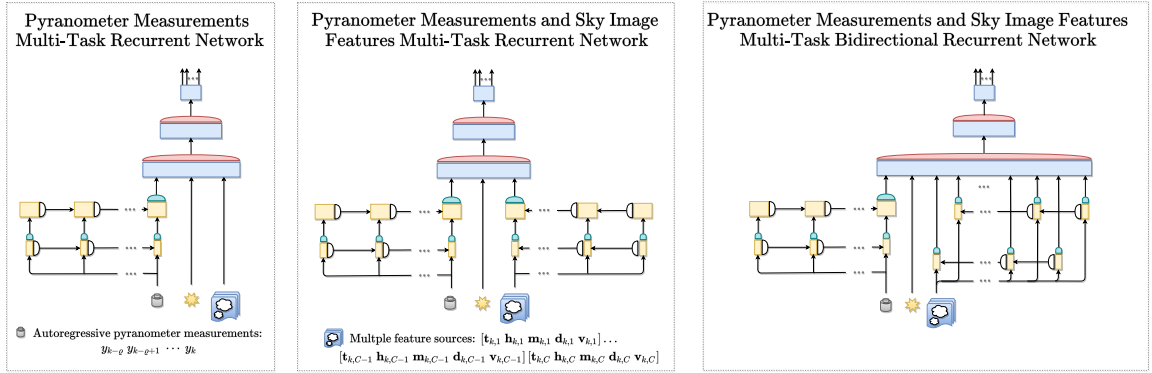


Figure 9.8: This figure shows the schematics of the different RNN architectures. The first architecture uses an RNN for the pyranometer measurements (AR model of solar radiation series). The second architecture uses an RNN for the pyranometer measurements and the features extracted from the IR sky images (multiple sources ResRNN). The third architecture uses an RNN for the pyranometer measurements and a BiRNN for the features extracted from the sky images.

An explorative analysis revealed that the angles are only useful when combined with features extracted from the sky images. However, the performances achieved by the feature vector (\mathbf{x}^C) worsen when the solar tracker angles are added to the vector ($\mathbf{x}^{C,A}$). In addition, when the feature vector ($\mathbf{x}^{C,A,T''}$) includes the processed temperatures T'' , the performance is better than when the feature vector ($\mathbf{x}^{C,A,T}$) includes the raw temperatures T . But, when the raw temperatures T are combined with heights H'' (computed using the processed temperatures T'') the feature vector ($\mathbf{x}^{C,A,T,H''}$) performs better (than $\mathbf{x}^{C,A,T'',H''}$). Therefore, the experiments presented in Figure 9.9 and 9.11 show the performances of the feature vectors which are most interesting for this application: $\mathbf{x}^{C,A,H''}$, $\mathbf{x}^{C,A,H'',T}$, $\mathbf{x}^{C,A,H'',T,M}$, $\mathbf{x}^{C,A,H'',T,M,V}$, and $\mathbf{x}^{C,A,H'',T,M,V,D}$.

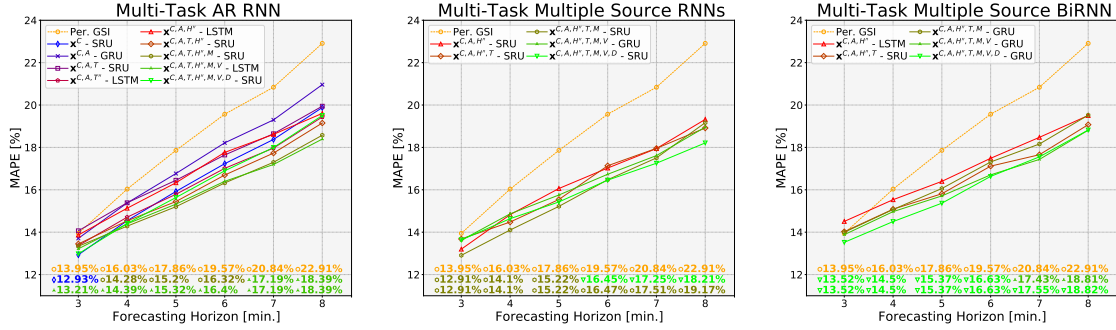


Figure 9.9: Performances of the RNN architectures in Figure 9.8. The left graph shows the MAPE when an RNN uses the pyranometer measurements. The middle graphs show the MAPE when another RNN uses multiple feature sources. And, the right graph shows the MAPE when, instead, a BiRNN uses the multiple feature sources. The MAPE achieved by the persistence in orange color with a circular marker (top line) is at the bottom of each graph, the lowest MAPE for each forecasting horizon by the color and the marker shape of the model (middle line), and the MAPE achieved by the model with the lowest average MAPE with its respective marker color and shape (bottom line).

Similarly, in the experiments with Residual RNN (ResRNN), the structural hyperparameters of the SRU with the feature vectors \mathbf{x}^C are fixed to the optimal (see Figure 9.9). The ResRNN architectures are shown in Figure 9.10. The recurrent structure in this set of experiments is the same as those shown in Figure 9.8, but with a residual layer added in each recurrent network. The results for an AR ResRNN in the pyranometer's measurements, a ResRNN in the pyranometer's measurements and in the multiple feature sources, and a Residual Bidirectional Recurrent Neural Network (ResBiRNN) in the multiple feature sources are shown in Figure 9.11. The Bayesian implementation of the ResRNN in the pyranometers measurements and in the multiple feature sources applying variational inference to the parameters of the MLP structure (see Section 9.2.1) is also shown in Figure 9.11. The error metric used to evaluate this architecture is the Root Mean Squared Percentage Error (RMSPE) [298], which is more adequate for a forecasting model trained minimizing the NLP in Eq. (9.19).

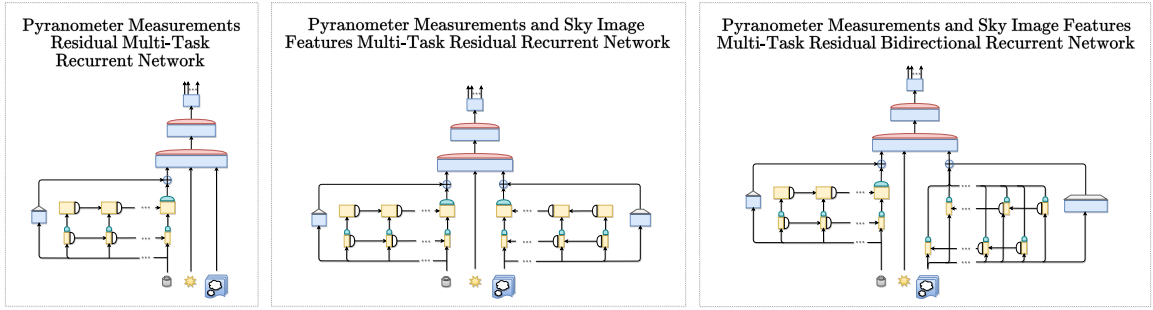


Figure 9.10: The figure shows the schematics of the proposed DL architectures with RNNs and residual layers. The RNNs are similar to those in Figure 9.8. In these architectures, the residual layer bypasses the RNN of pyranometer measurements in the left, the multiple feature sources in the middle, and the ResBiRNN in the right.

The FS is sensitive to the error metrics used to compute it. Therefore, the performance of a forecasting model depends on the loss function that is optimized. For this reason, the models are evaluated with both the MAPE and RMSE based FS. The MAPE-FS and RMSE-FS of deterministic and Bayesian multi-task multiple source ResRNN architecture, together with the MAPE-FS and RMSE-FS of the SOA are in Figure 9.12 for comparison. The SOA is two different CNN architectures that their authors named SUNSET [313], and SOLARNET [100] (see C). Since the objective is to develop an intra-hour solar forecasting algorithm to predict ramp-down and ramp-up events, the FS of the multi-task multiple source ResRNN architecture and its Bayesian implementation was analyzed as the GSI persistence MAPE and the RMSPE increases. The FS and the computational time of the proposed DL architectures using the feature vector that produced higher MAPE are in Figure 9.13. The computational cost of the SOA also appears in the figure. The type and number of layers that compose each DL architecture and the feature vector used are detailed in the legend.

The forecast performed by the multi-task multiple source ResRNN architecture and its Bayesian implementation in four different ramp-down events is in Figure 9.14. The Bayesian implementation has Confidence Intervals (CI), and it was verified that $\approx 95\%$ of the testing samples are within $\hat{\mu}_{y^*} \pm 2\hat{\sigma}_{y^*}$. The predictive standard deviation is the

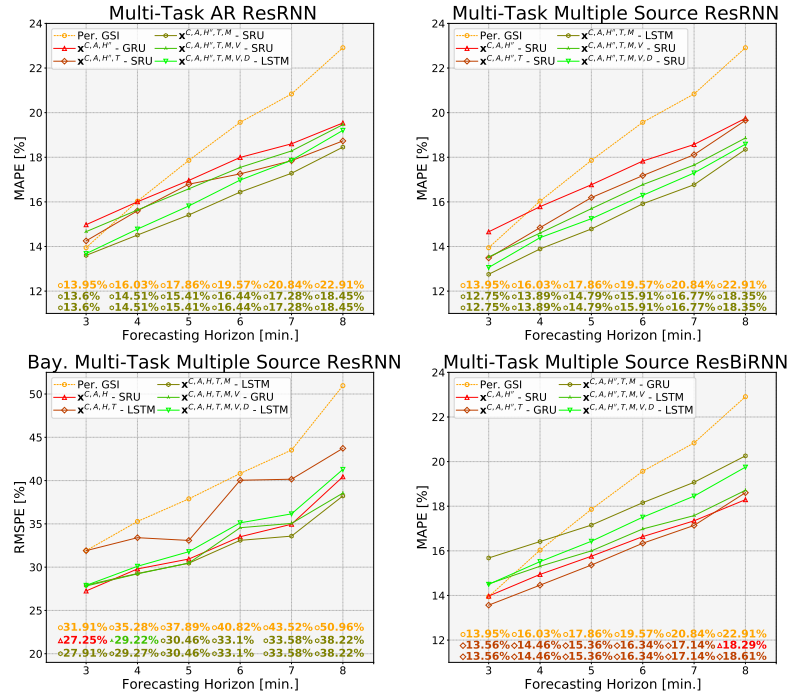


Figure 9.11: The graphs show the performances of the DL architectures with RNN and residual layers in Figure 9.10. Clockwise, the first graph shows the results when a ResRNN uses the pyranometer measurements. The second graph shows the results of the DL architectures when different ResRNN uses the pyranometer measurements and the multiple feature sources. The third graph shows the results when a ResRNN uses the pyranometer measurements, and a ResBiRNN uses the multiple feature sources. The fourth graph shows the results when the parameters of the deterministic architecture, multi-task multiple source ResRNN graph, is implemented using Bayesian inference. The MAPE or RMSPE of the persistence (in the first line with orange color and circular marker) is in the bottom of the graphs, the lowest MAPE or RMSPE achieved by one of the experiments in each forecasting horizon (in the second line using the color and marker shape of the corresponding experiment), and the architecture that archived the lowest average MAPE or RMSPE (in the third line with the color and marker shape of that experiment).

sum of the aleatoric and epistemic uncertainty $\hat{\sigma}_{\mathbf{y}^*} = \hat{\sigma}_n + \hat{\sigma}_p$. The predictive mean $\hat{\mu}_{\mathbf{y}^*}$ and standard deviation $\hat{\sigma}_p$ were computed following an MC approach. For each new observation \mathbf{x}^* , $S = 100$ predictions \mathbf{y}_i^* are sampled for numerically computing the mean $\hat{\mu}_{\mathbf{y}^*}$ and standard deviation $\hat{\sigma}_p$.

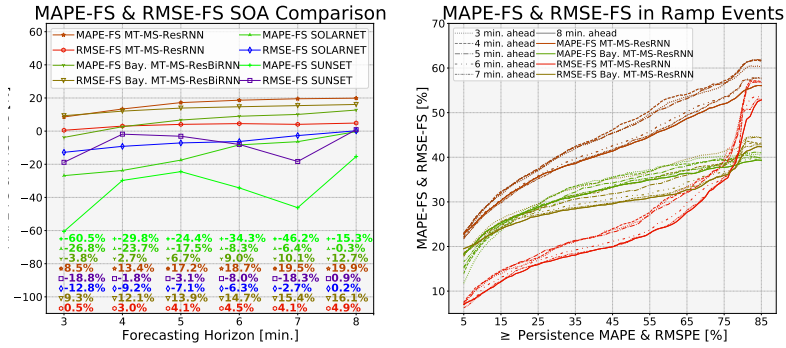


Figure 9.12: The graph on the left compares the MAPE-FS and RMSE-FS achieved by the SOA, the Multi-Task Multiple Source ResRNN (MT-MS-ResRNN), and the Bayesian implementation of this DL architecture. The graph on the right shows the performances of MT-MS-ResRNN and Bay. MT-MS-ResRNN in ramp-down and ramp-up events. As the MAPE and the RMSPE of the GSI persistence increases, the MAPE-FS and RMSE-FS of the DL models also increases (each line style indicates a different forecasting horizon).

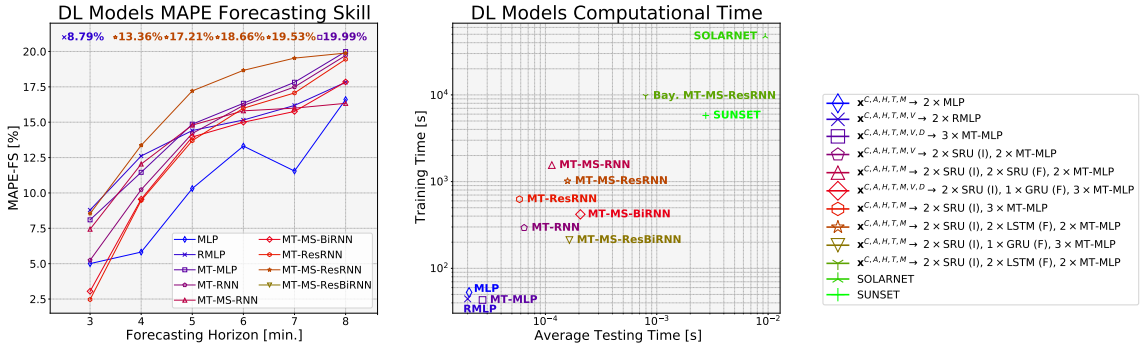


Figure 9.13: The graph on the left shows the FS of the best model of the proposed DL architectures. The one on the right shows the average computing time (in training and testing) of the proposed DL architectures and the SOA. The configuration of MLP and RNN layers and feature vectors in the architectures is in the legend. The RNN in pyranometer measurements is (I), and the RNN in the features is (F).

The experiments were carried out in the Xena HPC of the UNM-CARC, which uses an Intel Xeon CPU E5-2640 at 2.6GHz and a Nvidia Tesla K40M GPU with 64GB of RAM per node, 16 cores per node, 384 cores total and runs at 18 theoretical peak FLOPS. Scientific Linux is installed.

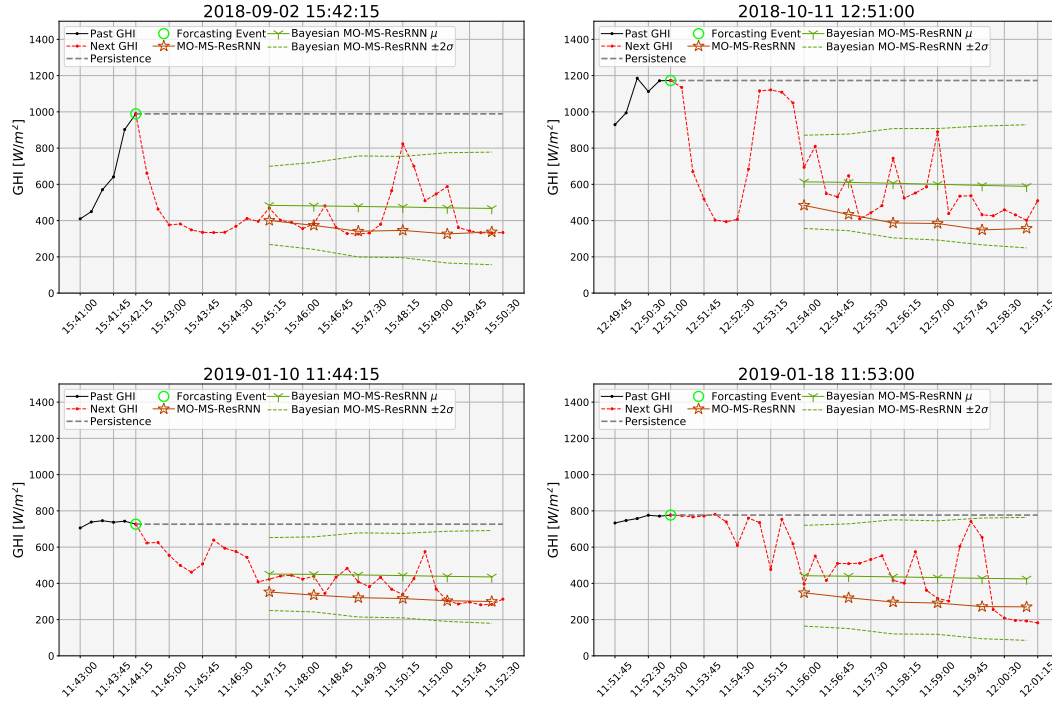


Figure 9.14: The graphs show four ramp-down forecasting events. The forecasts are from different days in the testing dataset (morning and afternoon). The time of the forecast is in bright green, the pyranometer measurements in the AR ResRNN are in black, the actual GSI measurements are in red. The GSI persistence is gray, the deterministic forecast is brown (MT-MS-RNN), and the Bayesian forecast is dark green (Bay. MT-MS-RNN). The mean is a solid line while the CI is a dashed line.

9.4 Discussion

The sequences of images in Figure 9.4 show the evolution of clouds. As the time interval between sky images increases, it is more difficult to anticipate when and which air parcels (i.e., pixel) will intersect with the Sun. The second order dynamics of the cloud velocity vectors in Figure 9.3 show which velocity vectors have a high curl and divergence, and therefore where the clouds are rapidly changing their shape and trajectory. Thus, the uncertainty of an air parcel in the streamline intersecting with the Sun increases (see Figure 9.5), and the density in the information to accurately

perform a forecast tends to have a uniform distribution.

Multi-task architectures require deeper NNs, which is computationally expensive in training but, in turn, reduces the computational cost in testing while achieving similar or even better accuracy than a recursive architecture. RNNs were combined with a multi-task architecture to model the time structure, but with decreased accuracy. However, residual layers support the RNNs in the multi-task architecture to perform better than a simple MLP (see Figure 9.13), which is probably due to the depth of the network producing vanishing gradient problems. The multiple source BiRNN architecture underperforms the RNNs with AR or multiple source architectures. It indicates that a time structure exists in the multiple features sources (see Figure 9.11). The optimal type of RNN depends on the feature vectors. No evidence of something different affecting the performances exists in the experiments (see Figure 9.9 and 9.11). However, the ResRNN architecture that achieved the lowest MAPE uses an LSTM for the multiple sources of the features extracted from the sky images and a GRU in the AR pyranometer measurements.

The SOA CNNs analyzed in this chapter achieved very high FS in the original work, 15.7% and 10.91%, for 15 and 10 minutes ahead with a resolution of 1 minute and 10 minutes SUNSET and SOLARNET respectively. However, their implementation in this chapter achieved poor performances with a remarkably high computational cost, even though we included early stopping and best-suited learning rates to compensate for the fact that the forecasting horizons differ from the original work. Other work found in the literature proposed an intra-hour solar forecasting method based on CNNs for 5 and 8 minutes ahead with a resolution of 1 minute and achieved an FS of 5.6% and 15.4% [354]. Another investigation introduced a CNN for 5 and 10 minutes ahead with a resolution of 1 minute, which achieved a FS of 14.4% and 12.1% [369]. The DL architecture forecasting performance presented in this work achieved a FS of 17.21% and 19.99% for 5 and 8 minutes ahead with a resolution of 15 seconds (in all

weather conditions see Figure 9.4 and 9.13), with much lower computation costs.

The FOV on the IR sky images limits the capabilities of the proposed solar forecast method in applications with forecasting horizons below 10 minutes, see Figure 8.2. In addition, the experiments show evidence that adding features of the cloud dynamics increases the forecast performance when the forecasting horizon also increases (see Figure 9.7 and 9.9). However, the results in Figure 9.11 obtained with residual architectures show the opposite, that the divergence and curl may not be necessary. Therefore, further experimentation should validate this hypothesis.

9.5 Conclusion

This chapter introduces a deterministic and a Bayesian multi-task deep learning that fuses information acquired from multiple sensors using independent recurrent structures to forecast solar radiation from 4 to 8 minutes ahead. The sky images are processed to remove cyclostationary artifacts and extract features from the cloud dynamics. The probability of an air parcel in the sky images intersecting with the Sun is computed for the different forecasting horizons to avoid the usage of convolutional filters. The feature vectors used in the proposed architectures include physical features from multiple sources. The results presented in this chapter show that the proposed intra-hour solar forecasting can potentially reduce the uncertainty in the energy generated by PV systems in power grids.

The analysis conducted in this chapter demonstrates that the introduction of image understanding and feature extraction methods based on the fusion of information acquired by multiple sensors, instead of using end-to-end learning (i.e., convolutional neural networks), increases the overall performances (i.e., computing time and accuracy) of an intra-hour solar forecasting algorithm. In addition, the prediction of the uncertainty in an intra-hour solar forecast conveys information necessary to reduce

the operational costs in SGs.

Convolutional neural networks are efficient for extracting complex spatial correlations and identifying patterns. However, the task of solar forecasting requires the extraction of non-linear (i.e., lens distortion and perspective depth) spatio-temporal correlations that the proposed convolutional filters are not capable of modeling. The implementation of CNNs in solar forecasting results in the optimization of thousands of filters' parameters which, in turn, limits the feasible number of structural hyperparameters to cross-validate. These practices are contrary to the fundamental principles in statistical learning for improving performance (i.e., reducing models' complexity and Occam's razor).

Future work may extend the image processing methods applied to the sky images to estimate the water content in an air parcel (i.e., pixels), which will give information about the thickness and density of clouds. The proposed methodology could be used for longer forecasting horizons when including information from the visible light spectrum using a large field of view sky imager (i.e., fisheye lens). Moreover, it is necessary to investigate efficient convolutional neural network architecture for solar forecasting by developing convolutional filters capable of exploiting the time structure in sequences of sky images. Another innovation that will likely increase the accuracy of the solar forecast is the addition of a recurrent network to the proposed neural network that includes weather features (i.e., relative humidity, air temperature, and atmospheric pressure).

Chapter 10

Conclusion

This dissertation proposes different multi-task intra-hour solar forecasting models based on kernel and deep learning. Among the proposed forecasting models are both deterministic and Bayesian methods. The approach introduced in this dissertation applies image understanding to reduce the computational time required to make a prediction. In addition, the proposed methodology for extracting physical features from cloud dynamics and fusing information from multiple sensors increases the accuracy of the state-of-the-art in intra-hour solar forecasting. It is possible to extract physical features from the cloud dynamics due to the innovative sky imager with far-infrared technology developed for this investigation.

The different solar forecasting models proposed in this investigation use data acquired with an innovative ground-based far-infrared sky imager. The sky imager is mounted on a solar tracker to maintain the Sun in the center of the images throughout the day, reducing the scattering effect produced by the Sun's direct radiation. The sky images and the pyranometer signal are processed to remove cyclostationary processes and extract physical features of the cloud dynamics. In addition, a geospatial perspective reprojection is proposed to undistort the sky image

Chapter 10. Conclusion

perspective and increase the accuracy when computing the cloud velocity vectors. The methods introduced in this investigation are capable of detecting when the clouds that appear in a sequence of consecutive images are flowing in different wind velocity fields. This information is used to independently analyze the velocity vectors of each cloud and approximate the multiple wind velocity fields. The extraction of features is performed assuming that the air parcel in a sky image is small enough that the vorticity and divergence of the wind velocity field are negligible. Under these circumstances, the streamlines are equivalent to the pathlines, so the Sun intersecting streamline is analyzed to estimate the probability of a pixel (i.e., voxel) occluding the Sun's direct radiation. Cloud features are extracted from sources independently estimated for each forecasting horizon. The extracted features are combined with information from multiple sensors to forecast solar radiation at multiple forecasting horizons.

Intra-hour solar forecasting is necessary to control energy dispatch, balancing generation with consumption in a power grid. The usage of intra-hour solar forecasting in combination with energy storage (i.e., batteries or hydrogen) has the potential to decrease the ramp rates produced by photovoltaic systems connected to a power grid as the forecast accuracy improves (i.e., forecasting skill). Consequently, intra-hour solar forecasting reduces the uncertainty in available resources, which, in turn, also reduces the amount of energy storage necessary to guarantee a stable supply of energy and the operational cost of a grid powered by solar energy. In particular, the methods introduced in this investigation considerably reduce the complexity of a solar forecasting model compared to the most common end-to-end learning methods (i.e., convolutional neural networks). In addition, the proposed multi-task forecasting models require only a single model to make a prediction, which reduces the computation time required for making a prediction and the latency in the forecast. Applying Bayesian statistics to the parameter inference in the kernel and deep learning methods, the forecasting models provide an estimation of the uncertainty on the forecast. Most remarkably, the intra-hour forecasting models presented in this dissertation are state-of-the-art

and achieved the highest forecast skill reported in a global solar irradiance forecast.

The outcomes of this investigation can be used as a roadmap for future work in intra-hour solar forecasting. One supposition when developing a solar forecasting model based on machine learning is that the model will be capable of learning and operating autonomously. However, most of the work proposed in the literature (i.e., end-to-end learning) requires a large dataset to achieve good performances. In other words, these solar forecasting methods have to be deployed ahead of time to enable the learning processes for multiple years before the machine can perform an accurate forecast. These methods assume that the training time is reduced by applying transfer learning, but the efficacy of transfer learning in solar forecasting has not been analyzed, nor has the performance of a model pre-trained with data from different locations. Intra-hour forecasting is defined as any forecast with a horizon smaller than an hour. The forecast horizon is too large in the framework of solar forecasting. The intra-hour forecast required for managing the operations in a smart grid is currently in the range of 15 minutes, but the required forecast horizon will be reduced as the penetration of photovoltaic systems increases. In the future, the forecast horizon will be in the range of 2.5 to 15 minutes. When data is acquired with large sampling intervals (i.e., 5 to 15 minutes), the cloud dynamics information is ignored, and the prediction is solely based on the previous data (i.e., experience), increasing the necessary complexity of the model. The dynamics of the clouds contain information about their trajectory and evolution that is important to forecast the formation or evaporation of a cloud. Therefore, future work should consider using data with short sample intervals to forecast at higher intervals.

Appendices

A	Unsupervised Parameterization of Velocity Vectors Methods	246
B	Wavefunction Probabilities	264
C	Convolutional Layers	265
D	Data and Software Availability	267

Appendix A

Unsupervised Parameterization of Velocity Vectors Methods

A.1 Methods

The cloud velocity vectors are used to approximate the streamlines and potential lines of the wind velocity field in a frame, as the clouds are the only elements out at our disposal for the visualization of the wind velocity field. This section aims to find the most suitable method available on the literature to compute the velocity vectors between two consecutive frames.

For that, different wind velocity fields are simulated on a sequence of images. These fields are either linear or non-linear. A sequence of IR images from a cloud evolving over time, was selected beforehand. These images are added to the simulated wind velocity fields, so that the cloud flows in the simulated velocity field. BO is implemented to optimize the parameters for each one of the methods in our analysis. The most effective method is selected looking for a trade-off between the error approximating the velocity vector, and the computing time.

A.1.1 Velocity Vectors

In computer vision, the dense approximation of velocity vectors describing the pixel-wise motion of an object in an image is performed employing a sequence of consecutive images. The predominant techniques to estimate the motion of objects are LK [209], HS [150] and Farnebäck [96] methods. Taking a different disciplinary approach, the field of experimental fluid dynamics uses research methods that are based on the statistical principle of signal cross-correlation in the frequency domain [1]. The signal Cross-Correlation (CC) can either be normalized or unnormalized. Techniques such as these are called Particle Image Velocimetry (PIV). The techniques to estimate the motion vectors in an image are sensitive to the pixels' intensity gradient. We implemented a model that removes the gradient produced by the solar direct radiation, and atmospheric scattered radiation. Both of which routinely appear on the images in the course of the year. A persistent model of the window of the camera removes sporadic artifacts that appear in the image such as water spots, or dust particles. A series of sequences of images with clouds flowing different directions were simulated to cross-validate the set of parameters for each one of the mentioned methods. The investigation was searching for a dense implementation of a motion vector method to approximate the dynamics of a cloud.

Optical Flow

The optical flow equation considers that exists a small displacement Δx and Δy in the direction of an object in an image. The object is assumed to have constant intensity \mathcal{I} between two consecutive frames \mathcal{I}_1 and \mathcal{I}_2 . The frames are separated in time by small time increment Δk ,

$$\mathcal{I}(x, y, k) = \mathcal{I}(x + \Delta x, y + \Delta y, t + \Delta k). \quad (\text{A.1})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

Assuming that the difference in intensity between neighboring pixels is smooth and that brightness of a pixel in consecutive frames is constant, the Taylor series expansion is applied and following equation obtained,

$$\mathcal{I}(x + \Delta x, y + \Delta y, k + \Delta k) = \mathcal{I}(x, y, k) + \frac{\partial \mathcal{I}}{\partial x} \Delta x + \frac{\partial \mathcal{I}}{\partial y} \Delta y + \frac{\partial \mathcal{I}}{\partial k} \Delta k. \quad (\text{A.2})$$

The factors are simplified combining the last two equation,

$$\frac{\partial \mathcal{I}}{\partial x} \Delta x + \frac{\partial \mathcal{I}}{\partial y} \Delta y + \frac{\partial \mathcal{I}}{\partial k} \Delta k = 0. \quad (\text{A.3})$$

The velocity of an object is derived dividing the terms of the displacement by the increment of time Δt ,

$$\frac{\partial \mathcal{I}}{\partial x} \frac{\Delta x}{\Delta k} + \frac{\partial \mathcal{I}}{\partial y} \frac{\Delta y}{\Delta k} + \frac{\partial \mathcal{I}}{\partial k} \frac{\Delta k}{\Delta k} = 0. \quad (\text{A.4})$$

The velocity components are defined as u and v so that,

$$\frac{\partial \mathcal{I}}{\partial x} u + \frac{\partial \mathcal{I}}{\partial y} v + \frac{\partial \mathcal{I}}{\partial k} = 0. \quad (\text{A.5})$$

This equation is known as the aperture problem,

$$\mathcal{I}_x u + \mathcal{I}_y v = -\mathcal{I}_k, \quad (\text{A.6})$$

where $\mathcal{I}_x = \partial \mathcal{I} / \partial x$, $\mathcal{I}_y = \partial \mathcal{I} / \partial y$ and $\mathcal{I}_k = \partial \mathcal{I} / \partial k$ are the derivatives for notation simplification.

The 2-dimensional derivatives are approximated using convolutional filters in a image [134]. Let us define a discrete time and space sequence of images as $\mathbf{I}_k = \{i_{i,j,k} \in \mathbb{R}^{[0,2^8]} \mid i = 1, \dots, M, j = 1, \dots, N\}$. The finite differences method is applied to compute the derivatives [95],

$$\begin{aligned} \mathbf{I}'_x &= \mathbf{I}_{k-1} \star \mathbf{K}_x \\ \mathbf{I}'_y &= \mathbf{I}_{k-1} \star \mathbf{K}_y \\ \mathbf{I}'_k &= \mathbf{I}_{k-1} \star \mathbf{K}_k + \mathbf{I}_k \star -\mathbf{K}_k \end{aligned} \quad (\text{A.7})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

where \star represent a 2-dimensional convolution, and \mathbf{I}_{k-1} and \mathbf{I}_k are the first and second consecutive frames. \mathbf{K}_x , \mathbf{K}_y and \mathbf{K}_k are the differential kernels in the x, y and t direction respectively,

$$\mathbf{K}_x = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}, \quad \mathbf{K}_y = \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{K}_k = \sigma \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad (\text{A.8})$$

the parameter σ is the amplitude of the temporal kernel. This parameter may be cross-validated when the velocity field is known.

Lucas-Kanade. This method proposes to find the solution for the optical flow equations via LS (i.e., $\mathbf{A}\mathbf{v} = \mathbf{b}$) in a sliding windows that is defined as,

$$\mathbf{x}_{i,j} = \{x_{i+m,j+n} \in \mathbb{R} \mid m = -w, \dots, +w, n = -w, \dots, +w, \} \in \mathbb{R}^{W^2}, \quad (\text{A.9})$$

where i, j refers to the pixel where the sliding windows is centered, and $W = 2w + 1$ is the size of the windows. The set of independent, and depended variables, in matrix form that is used to find the least-square solution is,

$$\mathbf{A}_{i,j} = \begin{bmatrix} \mathbf{I}'_x(x_{i,j}^{(1)}) & \mathbf{I}'_y(x_{i,j}^{(1)}) \\ \mathbf{I}'_x(x_{i,j}^{(2)}) & \mathbf{I}'_y(x_{i,j}^{(2)}) \\ \vdots & \vdots \\ \mathbf{I}'_x(x_{i,j}^{(W)}) & \mathbf{I}'_y(x_{i,j}^{(W)}) \end{bmatrix}, \quad \mathbf{v}_{i,j} = \begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}, \quad \mathbf{b}_{i,j} = \begin{bmatrix} -\mathbf{I}'_k(x_{i,j}^{(2)}) \\ -\mathbf{I}'_k(x_{i,j}^{(2)}) \\ \vdots \\ -\mathbf{I}'_k(x_{i,j}^{(W)}) \end{bmatrix}. \quad (\text{A.10})$$

The solution to the least-square problem is,

$$\begin{aligned} \mathbf{A}_{i,j}^\top \mathbf{A}_{i,j} \mathbf{v}_{i,j} &= \mathbf{A}_{i,j}^\top \mathbf{b}_{i,j} \\ \hat{\mathbf{v}}_{i,j} &= (\mathbf{A}_{i,j}^\top \mathbf{A}_{i,j})^{-1} \mathbf{A}_{i,j}^\top \mathbf{b}_{i,j}, \end{aligned} \quad (\text{A.11})$$

notice that this can be an ill-conditioned problem. When we look at the eigenvalues of the covariance matrix that are,

$$\mathbf{A}_{i,j}^\top \mathbf{A}_{i,j} = \mathbf{1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \mathbf{1}, \quad (\text{A.12})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

where $\mathbf{1}$ is an unitary matrix of size 2×2 , λ_1 and λ_2 has to be non-zero. This is equivalent to reduce the noise in the estimation of velocities applying a threshold to the eigenvalues of $\mathbf{A}_{i,j}^\top \mathbf{A}_{i,j}$ such as $\lambda_1 \geq \lambda_2 > \tau$.

Horn-Schunck. This method introduces a global Energy functional with a constrain applied to optical flow equation that has an additional regularization term. It is know as the smoothness constrain,

$$E = \iint [(\mathcal{I}_x \mathbf{u} + \mathcal{I}_y \mathbf{v} + \mathcal{I}_k)^2 + \alpha^2 (\|\nabla \mathbf{u}\|^2 + \|\nabla \mathbf{v}\|^2)] dx dy, \quad (\text{A.13})$$

where α is the parameter of the regularization term.

The aim is to minimize E via differentiating w.r.t. the variables \mathbf{u} and \mathbf{v} . The solution for a pair of second order differential equations can be computed iteratively solving the multi-dimensional Euler-Lagrange equations which are,

$$\begin{aligned} \frac{\partial E}{\partial \mathbf{u}} &= \frac{\partial}{\partial x} \frac{\partial E}{\partial \mathbf{u}} + \frac{\partial}{\partial y} \frac{\partial E}{\partial \mathbf{v}}, \\ \frac{\partial E}{\partial \mathbf{v}} &= \frac{\partial}{\partial x} \frac{\partial E}{\partial \mathbf{u}} + \frac{\partial}{\partial y} \frac{\partial E}{\partial \mathbf{v}}, \end{aligned} \quad (\text{A.14})$$

The following set of equations are obtained after differentiating,

$$\begin{aligned} \mathcal{I}_x (\mathcal{I}_x \mathbf{u} + \mathcal{I}_y \mathbf{v} + \mathcal{I}_t) - \alpha^2 \Delta \mathbf{u} &= 0, \\ \mathcal{I}_y (\mathcal{I}_x \mathbf{u} + \mathcal{I}_y \mathbf{v} + \mathcal{I}_k) - \alpha^2 \Delta \mathbf{v} &= 0, \end{aligned} \quad (\text{A.15})$$

where $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is the Laplace operator that controls the smoothness. It can be differentiated numerically so that $\Delta \mathbf{u} = 4(\bar{\mathbf{u}} - \mathbf{u})$, where $\bar{\mathbf{u}}$ represents the weighted sample mean of the pixels in the neighborhood, same for the \mathbf{v} velocity component. This operation can be implemented as a convolution using the differential operators \mathbf{K}_{HS} ,

$$\bar{\mathbf{U}} = \hat{\mathbf{U}} \star \mathbf{K}_{HS}, \quad \bar{\mathbf{V}} = \hat{\mathbf{V}} \star \mathbf{K}_{HS}, \quad \mathbf{K}_{HS} = \begin{bmatrix} \frac{1}{12} & \frac{1}{6} & \frac{1}{12} \\ \frac{1}{6} & 0 & \frac{1}{6} \\ \frac{1}{12} & \frac{1}{6} & \frac{1}{12} \end{bmatrix}. \quad (\text{A.16})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

The formula can be rearranged and express in matrix form to compute all the velocity vectors in an image at the same time,

$$\begin{aligned} \left(\mathbf{I}_x'^2 + 4\alpha^2\right) \odot \hat{\mathbf{U}} + \mathbf{I}_x' \odot \mathbf{I}_y' \odot \hat{\mathbf{V}} &= 4\alpha^2 \bar{\mathbf{U}} - \mathbf{I}_x' \odot \mathbf{I}_k', \\ \mathbf{I}_x' \odot \mathbf{I}_y' \odot \hat{\mathbf{U}} + \left(\mathbf{I}_y'^2 + 4\alpha^2\right) \odot \hat{\mathbf{V}} &= 4\alpha^2 \bar{\mathbf{V}} - \mathbf{I}_y' \odot \mathbf{I}_k', \end{aligned} \quad (\text{A.17})$$

where \odot is element-wise multiplication.

The estimation of the velocity components $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ are iteratively computed, and updated $\bar{\mathbf{U}}$ and $\bar{\mathbf{V}}$ for each pixels' neighborhood in the sliding window (similar to the approach implemented in LK method). The velocity components are defined to be 0 at the beginning of the algorithm, and are updated following these set of equations,

$$\begin{aligned} \hat{\mathbf{U}}^{(t+1)} &= \bar{\mathbf{U}}^{(t)} - \frac{\mathbf{I}_x' \odot (\mathbf{I}_x' \odot \bar{\mathbf{U}}^{(t)} + \mathbf{I}_y' \odot \bar{\mathbf{V}}^{(t)} + \mathbf{I}_t')}{4\alpha^2 + \mathbf{I}_x'^2 + \mathbf{I}_y'^2}, \\ \hat{\mathbf{V}}^{(t+1)} &= \bar{\mathbf{V}}^{(t)} - \frac{\mathbf{I}_y' \odot (\mathbf{I}_x' \odot \bar{\mathbf{U}}^{(t)} + \mathbf{I}_y' \odot \bar{\mathbf{V}}^{(t)} + \mathbf{I}_t')}{4\alpha^2 + \mathbf{I}_x'^2 + \mathbf{I}_y'^2}, \end{aligned} \quad (\text{A.18})$$

where α is the parameters of the regularization term, and t is the optimization iteration. Similarly, the sample weighted mean of the pixels in the neighborhood is updated as,

$$\bar{\mathbf{U}}^{(t+1)} = \hat{\mathbf{U}}^{(t+1)} \star \mathbf{K}_{HS}, \quad \bar{\mathbf{V}}^{(t+1)} = \hat{\mathbf{V}}^{(t+1)} \star \mathbf{K}_{HS}. \quad (\text{A.19})$$

The optimization continues until either the tolerance is $(\|\hat{\mathbf{U}}^{(t)} - \hat{\mathbf{U}}^{(t+1)}\| + \|\hat{\mathbf{V}}^{(t)} - \hat{\mathbf{V}}^{(t+1)}\|) < \tau$, or is achieved the maximum number of iteration.

Farnebäck. This method proposes to approximate a neighborhood of pixels in an image by local quadratic polynomial expansion such as,

$$\mathcal{I}_2(x, y) = \mathbf{x}^\top \mathbf{A}_1 \mathbf{x} + \mathbf{b}_1^\top \mathbf{x} + c_1, \quad (\text{A.20})$$

so that the displacement in the same neighborhood between two consecutive images is,

$$\mathcal{I}_1(x, y) = \mathcal{I}_2(x - \Delta x, y - \Delta y) = \mathbf{x}^\top \mathbf{A}_2 \mathbf{x} + \mathbf{b}_2^\top \mathbf{x} + c_2. \quad (\text{A.21})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

where $\mathbf{x} = [x \ y]$ and $\Delta\mathbf{x} = [\Delta x \ \Delta y]^\top$. The quadratic expansion coefficients $\mathbf{A}_1(\mathbf{X})$, $\mathbf{A}_2(\mathbf{X})$, $\mathbf{b}_1(\mathbf{X})$, and $\mathbf{b}_2(\mathbf{X})$ are numerically approximated using the pixels in the neighborhood of a sliding window of size $W = 2w + 1$. The pixels in the sliding window are defined as $\mathbf{X} = \{(x_i, y_i) \in \mathbf{N}^2 | \forall i, \dots, W^2\}$. Rearranging the local quadratic expansion it is obtained that,

$$\begin{aligned} \mathbf{A}(\mathbf{X}) &= \frac{\mathbf{A}_2(\mathbf{X}) - \mathbf{A}_1(\mathbf{X})}{2}, \\ \Delta\mathbf{b}(\mathbf{X}) &= -\frac{\mathbf{b}_1(\mathbf{X}) - \mathbf{b}_2(\mathbf{X})}{2}, \\ \Delta\mathbf{x}(\mathbf{X}) &= \mathbf{A}(\mathbf{X})^{-1} \Delta\mathbf{b}(\mathbf{X}), \end{aligned} \tag{A.22}$$

where $\Delta\mathbf{x}(\mathbf{X})$ is the local displacement in the neighborhood.

In order to add robustness to the displacement estimation $\Delta\mathbf{x}(\mathbf{x})$, the field is configured according to a motion model that follows theses equations,

$$\begin{aligned} \mathbf{u} &\triangleq \Delta x(x, y) = a_1 + a_2x + a_3y + a_7x^2 + a_8xy, \\ \mathbf{v} &\triangleq \Delta y(x, y) = a_4 + a_5x + a_6y + a_7xy + a_8y^2. \end{aligned} \tag{A.23}$$

The coefficients of the WLS problem $\mathbf{v} = \mathbf{S}\mathbf{p}$ for this particular motion model are,

$$\begin{aligned} \mathbf{v} &= \begin{bmatrix} \mathbf{u} & \mathbf{v} \end{bmatrix}^\top, \\ \mathbf{S} &= \begin{bmatrix} 1 & x & y & 0 & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 1 & x & y & x & y & y^2 \end{bmatrix}, \\ \mathbf{w} &= \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \end{bmatrix}^\top, \end{aligned} \tag{A.24}$$

where \mathbf{S} is the designed matrix (i.e., model) of the motion. The WLS solution of the displacement in the same neighborhood of pixel in two consecutive images is such as,

$$\hat{\mathbf{w}} = \left(\sum_{i=1}^{W^2} \gamma_i \mathbf{S}_i^\top \mathbf{A}_i^\top \mathbf{A}_i \mathbf{S}_i \right)^{-1} \sum_{i=1}^{W^2} \gamma_i \mathbf{S}_i^\top \mathbf{A}_i^\top \Delta\mathbf{b}_i. \tag{A.25}$$

The samples weights γ_i are the probabilities of the pixels on the sliding windows of size W , evaluated using a Normal distribution with standard deviation σ^2 .

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

This algorithm includes prior information to approximate small displacements, an approach based on multiple scales layers from coarse to finer, that propagates the displacement to obtain better result as the layers scale increases (see [96] for more information).

Particle Image Velocimetry

The methods that are predominant in the fluid mechanic literature [1], are based on the CC of same region in an image between two consecutive frames. The computation of the CC is expensive, so these methods are implemented as a sparse approach instead of a dense one, which is the case of the methods based on optical flow. Nevertheless, when we apply the cross-correlation theorem, which says that a convolution in the time domain becomes a dot product in the frequency domain,

$$\mathbf{R}_{i,j} = \mathcal{I}_1(i,j) \star \mathcal{I}_2(i,j) = \mathcal{F}^{-1} \left\{ \overline{\mathcal{F} \{ \mathcal{I}_1(i,j) \}} \cdot \mathcal{F} \{ \mathcal{I}_2(i,j) \} \right\}, \quad \mathcal{I}_1, \mathcal{I}_2 \in \mathbb{R}^{M \times N} \quad (\text{A.26})$$

where \star denotes a convolution. The CC computation indeed requires less time in the frequency domain. Although the solution of this problem still requires a sparse approach or a GPU in case of large resolution data.

The CC methods are computed using a sliding window similar to LK and Farnebäck, so i, j refers to the pixel where the sliding windows is centered, and $W = 2w + 1$ is the size of the windows. The index sets of the pixels in a window are $m = \{-w, \dots, +w\}$ and $n = \{-w, \dots, +w\}$.

The sliding windows that goes over the images in the frequency domain is,

$$\mathbf{W}_{i,j}^{(1)} = \mathcal{F} \{ \mathcal{I}_1(i,j) \}, \text{ and } \mathbf{W}_{i,j}^{(2)} = \mathcal{F} \{ \mathcal{I}_2(i,j) \}, \quad \mathbf{W}_{i,j}^{(1)}, \mathbf{W}_{i,j}^{(2)} \in \mathbb{R}^{W \times W} \quad (\text{A.27})$$

where \mathcal{I}_1 is the previous image, and \mathcal{I}_2 is the current image, W is the sliding window's size.

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

Spatial Cross-Correlation. The 2-dimensional CC function in the time domain is such as,

$$\mathbf{R}_{i,j} = \sum_{m=-w}^{+w} \sum_{n=-w}^{+w} \mathcal{I}_1(i, j) \cdot \mathcal{I}_2(i + m, j + n), \quad \forall i = 1, \dots, M, \forall j = 1, \dots, N, \quad (\text{A.28})$$

and when we apply the convolution theorem to the CC function between the two image's region, we obtain that it is equivalent to the following formula in frequency domain,

$$\mathbf{\Gamma}_{i,j} = \mathbf{W}_{i,j}^{(1)} \odot \mathbf{W}_{i,j}^{(2)*}, \quad \forall i = 1, \dots, M, \forall j = 1, \dots, N, \quad (\text{A.29})$$

where \odot represents element-wise multiplication, and $*$ refers to the conjugate of $\mathbf{W}_{i,j}^{(2)}$.

Normalized Cross-Correlation (NCC). An alternative to the CC is to compute the NCC. The function of NCC for a 2-dimensional convolution has following expression,

$$\hat{\mathbf{R}}_{i,j} = \frac{\sum_{m=-w}^{+w} \sum_{n=-w}^{+w} \left(\mathcal{I}_1(i, j) - \mu_{i,j}^{(1)} \right) \cdot \left(\mathcal{I}_2(i + m, j + n) - \mu_{i,j}^{(2)} \right)}{\sigma_{i,j}^{(1)}(i, j) \cdot \sigma_{i,j}^{(2)}(i + m, j + n)}, \quad (\text{A.30})$$

$$\forall i = 1, \dots, M, \forall j = 1, \dots, N,$$

where $\mu_{i,j}^{(1)}$ and $\sigma_{i,j}^{(1)}$ represents the sample mean and standard deviation of the pixels in the sliding window of image $\mathcal{I}_1(i, j)$. $\mu_{i,j}^{(2)}$ and $\sigma_{i,j}^{(2)}$ represent the same but from image $\mathcal{I}_2(i, j)$. The sample mean is computed as,

$$\mu_{i,j} = \frac{1}{W^2} \sum_{m=-w}^{+w} \sum_{n=-w}^{+w} \mathcal{I}(i + m, j + n), \quad (\text{A.31})$$

and $\sigma_{i,j}$ is the sample standard deviation of the pixels in the same window,

$$\sigma_{i,j} = \sqrt{\frac{1}{W^2} \sum_{m=-w}^{+w} \sum_{n=-w}^{+w} (\mathcal{I}(i + m, j + n) - \mu_{i,j})^2}, \quad (\text{A.32})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

where W are the pixels in the window. When we apply the CC theorem to the NCC function, the equivalent formula obtained in the frequency domain is such as,

$$\hat{\Gamma}_{i,j} = \frac{\mathbf{W}_{i,j}^{(1)} \odot \mathbf{W}_{i,j}^{(2)*}}{\left| \mathbf{W}^{(1)} \odot \mathbf{W}_{i,j}^{(2)*} \right|}, \quad \forall i = 1, \dots, M, \quad \forall j = 1, \dots, N, \quad (\text{A.33})$$

The results of computing the CC or NCC can be transform back to the time domain,

$$\mathbf{R}_{i,j} = \mathcal{F}^{-1} \{ \Gamma_{i,j} \}, \quad \mathbf{R}_{i,j} \in \mathbb{R}^{W \times W} \quad (\text{A.34})$$

$\mathbf{R}_{i,j}$ is a matrix where the entry that has higher CC or NCC corresponds to the translation occur in the window. When computing NCC, the $\hat{\Gamma}_{i,j}$ is used instead of the $\Gamma_{i,j}$.

However, if the sliding window has low resolution, it will not be possible to determine the exact maximum. We propose to fit a polynomial expansion to the window's pixels CC or NCC and implement an optimization approach via numerical gradient to find the maximum of this smooth function that is,

$$\hat{\mathbf{u}}_{i,j}, \hat{\mathbf{v}}_{i,j} = \underset{\mathbf{w}}{\operatorname{argmax}} \Theta(\mathbf{R}_{i,j}, \mathbf{w}), \quad \hat{\mathbf{U}}, \hat{\mathbf{V}} \in \mathbb{R}^{M \times N}, \quad (\text{A.35})$$

where $\hat{\mathbf{u}}_{i,j}$ and $\hat{\mathbf{v}}_{i,j}$ is the displacement for each pixel in region of interest, depending on whether it is a dense or sparse implementation, and \mathbf{U}, \mathbf{V} are the velocity vectors, $\Theta(\cdot)$ is a model with a polynomial expiation \mathcal{P}^n expansion of order n applied to the features $\mathbf{R}_{i,j}$, and \mathbf{w} are the parameters of the polynomial model.

A.1.2 Bayesian Optimization

BO is a global optimization method applied when the objective function is computationally expensive and noisy. The samples obtained from evaluating the objective function are assumed to be a collection of random variables and modeled as a GPR

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

[274]. In this way, an acquisition function uses the uncertainty in the prior distribution for searching for maxima [356].

For this optimization method to perform an efficient search, the acquisition function is updated using the Bayes' Theorem as more information is available. The selection of the parameters of the acquisition function is a compromise between exploiting the most likely maximum or exploring a new region of the posterior predictive probability of $p(y^*|x^*)$. In reinforcement learning, BO is a method used in policy search problems [315].

Gaussian Process for Regression

This regression problem with noise for an arbitrary sample m is

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \quad (\text{A.36})$$

where the independent variable y_m (which is the validation error e_m), is modeled as a function $f(\cdot)$ of the dependent variable \mathbf{x}_m (which is the set of structural hyperparameter $\boldsymbol{\omega}_m$ that produced the validation error e_m), and the estimation has an error ε_m .

The predictive distribution of e^* for a new set of structural hyperparameter $\boldsymbol{\omega}^*$ given a dataset $\mathcal{D} = \{e_m, \boldsymbol{\omega}_m\}_{m=1}^M$ composed of M samples in a GPR is,

$$e^* \sim \mathcal{N}\left(\hat{\boldsymbol{\mu}}_{e^*}, \hat{\boldsymbol{\Sigma}}_{e^*}\right), \quad (\text{A.37})$$

where $\hat{\boldsymbol{\mu}}_{e^*} = \mathbf{k}_{\Omega}^{*\top} \mathbf{K}_{\Omega}^{-1} \mathbf{e}$, and $\hat{\boldsymbol{\Sigma}}_{e^*} = k_{\Omega}^{**} - \mathbf{k}_{\Omega}^{*\top} \mathbf{K}_{\Omega}^{-1} \mathbf{k}_{\Omega}^*$. The matrices $\mathbf{K}_{\Omega} \triangleq \mathcal{K}(\boldsymbol{\omega}_n, \boldsymbol{\omega}_m)$, $\mathbf{k}_{\Omega}^* \triangleq \mathcal{K}(\boldsymbol{\omega}_n, \boldsymbol{\omega}^*)$, and the scalar $k_{\Omega}^{**} \triangleq \mathcal{K}(\boldsymbol{\omega}^*, \boldsymbol{\omega}^*)$ are given by a designed covariance function [274]. A common covariance (i.e., kernel) function is the Matérn,

$$\mathcal{K}(\boldsymbol{\omega}_n, \boldsymbol{\omega}_m) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \cdot \gamma \|\boldsymbol{\omega}_n - \boldsymbol{\omega}_m\|^2 \right)^{\nu} K_{\nu} \left(\sqrt{2\nu} \cdot \gamma \|\boldsymbol{\omega}_n - \boldsymbol{\omega}_m\|^2 \right), \quad (\text{A.38})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

where $\boldsymbol{\Omega} = \{\gamma, \nu\}$ is the set of the kernel hyperparameters, $\Gamma(\cdot)$ is the Gamma function, and K_ν is the modified Bessel function of second kind.

The optimal hyperparameters' set of the kernel is obtained by minimizing the negative marginal log-likelihood (i.e., evidence),

$$\log p(\mathbf{e}|\boldsymbol{\Omega}, \mathcal{D}) = -\frac{N}{2} \log(2\pi) - \frac{1}{2} \log |\mathbf{K}_{\boldsymbol{\Omega}}| - \frac{1}{2} \mathbf{e}^\top \mathbf{K}_{\boldsymbol{\Omega}} \mathbf{e}. \quad (\text{A.39})$$

Acquisition Function: Expected Improvement (EI)

BO aims to maximize an acquisition function to decide which sample the objective function will evaluate in next iteration [293]. Different acquisition functions may serve this purpose [357]. However, all of these functions take advantage of the posterior distributions that have been previously inferred by applying Bayesian statistics. The EI is the most common acquisition function. This function estimates the probability of the improvement $\max(0, e_{best} - e^{(t)})$ produced by the set of structural hyperparameters $\boldsymbol{\omega}^{(t)}$ for sample in the iteration t .

The expectation of the improvement produced by $\boldsymbol{\omega}^{(t)}$ is computed with respect to the predictive posterior probability,

$$\begin{aligned} EI(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D}) &= \mathbb{E}_{e^{(t)}|\boldsymbol{\omega}^{(t)}, \boldsymbol{\Omega}, \mathcal{D}} [\max(0, e_{best} - e^{(t)})] \\ &= \int_e \max(0, e_{best} - e^{(t)}) p(e^{(t)}|\boldsymbol{\omega}^{(t)}, \boldsymbol{\Omega}, \mathcal{D}) de. \end{aligned} \quad (\text{A.40})$$

The analytical solution to the integral, evaluated at $\boldsymbol{\omega}^{(t)}$, is given by this formula [269],

$$EI(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D}) = \sigma(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D}) \{ \Upsilon(\boldsymbol{\omega}^{(t)}) \Phi[\Upsilon(\boldsymbol{\omega}^{(t)})] + \phi[\Upsilon(\boldsymbol{\omega}^{(t)})] \} \quad (\text{A.41})$$

where $\Phi[\cdot]$ and $\phi[\cdot]$ are the cumulative density function and the probability density function of the standard normal distribution, and

$$\Upsilon(\boldsymbol{\omega}^{(t)}) = \frac{f(\boldsymbol{\omega}_{best}) - \mu(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D}) + \xi}{\sigma(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D})}, \quad (\text{A.42})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

where $e_{best} = f(\boldsymbol{\omega}_{best})$ is the best one of the sample already explored, $\mu(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D})$ and $\sigma(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D})$ are the predicted mean and standard deviation of sample $\boldsymbol{\omega}^{(t)}$, and ξ is the parameter (set by the user) that controls the trade-off between exploration and exploitation.

Acquisition Function Optimization

To perform an efficient optimization, the objective function is evaluated next using the dependent variables vector $\boldsymbol{\omega}^{(t+1)}$ that maximizes an acquisition function within the domain of \mathcal{W} [306],

$$\boldsymbol{\omega}^{(t+1)} = \underset{\boldsymbol{\omega} \in \mathcal{W}}{\operatorname{argmin}} EI(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D}). \quad (\text{A.43})$$

For that, the minimum of an acquisition function is found by steepest descent optimization, $\boldsymbol{\omega}^{(t+1)} = \boldsymbol{\omega}^{(t)} - \eta \cdot \nabla EI(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D})$. The gradient $\nabla EI(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D})$ is computed with the finite differences formula to numerically approximate the derivatives,

$$\nabla EI(\boldsymbol{\omega}^{(t)}) = \lim_{\epsilon \rightarrow 0} \frac{EI(\boldsymbol{\omega}^{(t)} + \epsilon; \boldsymbol{\Omega}, \mathcal{D}) - EI(\boldsymbol{\omega}^{(t)}; \boldsymbol{\Omega}, \mathcal{D})}{\epsilon}. \quad (\text{A.44})$$

A.1.3 Velocity Vectors Regularization

The dense approximation of velocity vectors between two consecutive have outlier vectors. This is because of the assumption of constant intensity, and small time increments, that these methods require in their formulation, and that is sometimes violated. Therefore, we propose to regularized the velocity vectors to remove the information provided by the outliers at each computation. The cloud velocity vectors in the image k is,

$$\hat{\mathbf{v}}_{i,j} = \{(\hat{\mathbf{u}}_{i,j}, \hat{\mathbf{v}}_{i,j}) \mid \hat{\mathbf{v}}_{i,j} \in \mathbb{R}^2, \forall i = 1, \dots, M, \forall j = 1, \dots, N\}. \quad (\text{A.45})$$

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

A lower and upper threshold is applied to find the vectors, which magnitude is so low enough that they are likely noise, or so high that are unfeasible in the real world,

$$\hat{\mathbf{v}}_{i,j} = \begin{cases} \hat{\mathbf{v}}_{i,j} & \tau_{lower} < \|\hat{\mathbf{v}}_{i,j}\| \leq \tau_{upper} \\ \mathbf{0} & \text{Otherwise} \end{cases} \quad \forall m, n = -w, \dots w, \quad (\text{A.46})$$

where $\tau_{lower} = 0.1$ and $\tau_{upper} = 10$. The information lost by deleting these velocity vectors, can be partially recovered inferring it from neighbor vectors, For that, a median filter is applied on the velocity vectors after the regularization,

$$\hat{\mathbf{v}}_{i,j} = \begin{cases} \hat{\mathbf{v}}_{i,j} & \tau_{lower} < \|\hat{\mathbf{v}}_{i,j}\| \leq \tau_{upper} \\ \text{median}(\hat{\mathbf{v}}_{i+m,j+n}) & \text{Otherwise} \end{cases} \quad \forall m, n = -w, \dots w \quad (\text{A.47})$$

The filtered velocity vectors are recovered, only in those places where were regularized to zero, in order to replace them.

A.2 Experiments and Discussion

An actual IR image sequence of a cloud with non-simulated dynamics is cropped out of its original frames. The position of reference is the cloud's mass center in the frame. The cropped image of the cloud, which is centered in the cloud's mass center, and placed in a new frame with the simulated flow. The update process to generate a trajectory is: in frame \mathcal{I}_1 the cloud is displaced to new position in frame \mathcal{I}_2 , according to the simulated velocity vector increments at the cloud's mass center coordinates in frame \mathcal{I}_1 . This is update rule is repeated along the whole sequence of images. Therefore, the trajectory of the cloud is known, it can be compared with velocity vector computed using each method. This process is repeated for 4 different simulated flows, an example is on Figure A.1.

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

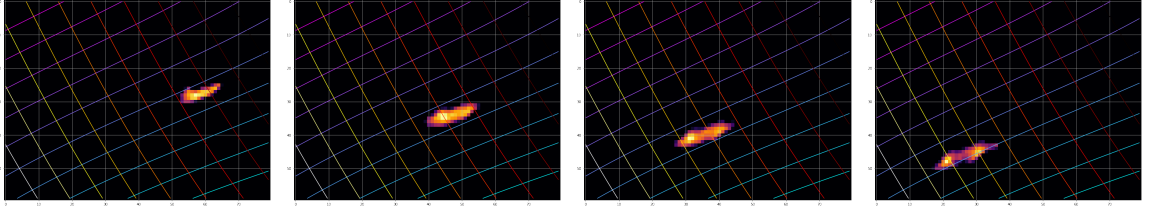


Figure A.1: One of the four sequences of images generated to cross-validate the optimal parameters for each method. In the cool colors are displayed the potential lines, in warm colors are displayed the streamlines. The generated flow displayed in the image is non-linear.

A.2.1 Parameters Cross-Validation

The methodology propose to validate the optimal set of parameters for each one of the velocity vectors techniques is to utilize them approximate a known wind velocity field on an sequence of images. The wind velocity field is generated defining beforehand the streamlines and potential lines. This simulates either linear or a non-linear flow.

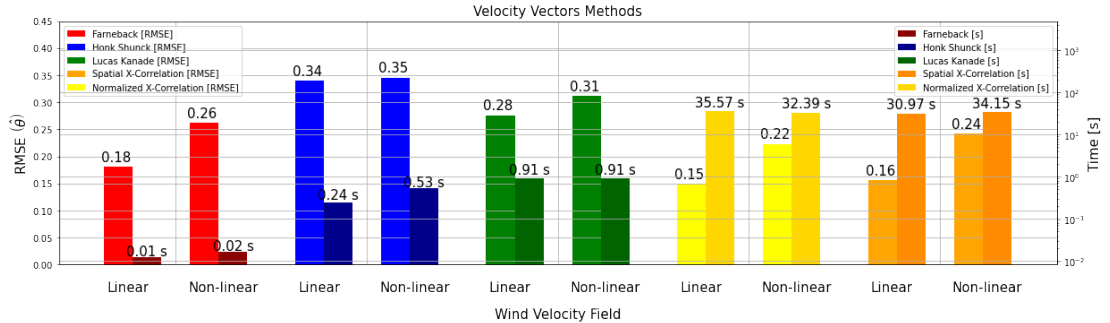


Figure A.2: This graphs shows the comparison of each method performances in RMSE and computing time for a linear flow, and a non-linear flow which has a small vorticity and divergence. The best trade-off between computing time and error is obtained by Farneback.

As the partial derivatives in our problem are not available, the optimal parameters for each technique are found by means of BO. This optimization method explores the most likely set of parameters to produce an improvement on an acquisition function

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

at each iteration. In particular, the acquisition functions aims to infer the function of the errors computing the velocity vectors by each one of methods. The function of the errors is estimated via a GPR. The optimal set of parameters for each method, are displayed in Tables A.1 to A.5.

Table A.1: This table shows the results of the LK method. The parameters validate where the size of the sliding window, and the noise threshold in the eigenvalues of the LS solution.

Flow	$\mathcal{W}_{p \times p}$	Eig. Thrs.	t [s]	RMSE
Linear	11	1×10^{-8}	0.9128	0.2759
Non-Linear	11	1×10^{-8}	0.9114	0.3115

Table A.2: The table shows the results obtained implementing the HS method with constant tolerance and a maximum number of iteration in the optimization, The parameters validated were the increments multiplayer in the optimization, and the amplitude of the differentiation kernel.

Flow	Inc. Size Opt.	Diff. Kernel Amp.	Tol.	Max. Iters.	t [s]	RMSE
Linear	13.6420	739.9022	1×10^{-5}	1000	0.2444	0.3398
Non-Linear	3.3449	244.2650	1×10^{-5}	1000	0.5328	0.3452

Table A.3: This table shows the results of the Farnebäck method. The parameters validated were the number of pyramids and their scale, the size of the sliding window, the number of interactions, and the order of the polynomial expansion and the variance of the Gaussian filter.

Flow	Pyr. Scale	No. Pyrs.	$\mathcal{W}_{p \times p}$	No. Iters.	\mathcal{P}^n	Filter σ^2	t [s]	RMSE
Linear	0.6529	7	2	11	5	1.3398	0.0120	0.1804
Non-Linear	0.2607	9	2	78	1	0.6368	0.0163	0.2620

Table A.4: This table shows the results of the CC method. The parameters validated where the size of the sliding window, and the order of the polynomial expansion implemented to approximate the cross-correlation function.

Flow	$\mathcal{W}_{p \times p}$	\mathcal{P}^n	t [s]	RMSE
Linear	22	11	30.9664	0.1557
Non-Linear	23	19	34.1549	0.2422

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

Table A.5: This table shows the results of the NCC method. The parameters validated where the size of the sliding window, and the order of the polynomial expansion implemented to approximate the normalized cross-correlation.

Flow	$\mathcal{W}_{p \times p}$	\mathcal{P}^n	t [s]	RMSE
Linear	23	14	35.5664	0.1495
Non-Linear	20	17	32.3930	0.2231

A.2.2 Note on the Implementation

After the cross-validation of the parameters for each method, these were implemented in the features extraction algorithm, see Figure A.3. Despite of the fact that the Farnebäck achieves the lowest computing time with a relative small error, it was found that produces outlier during the implementation. In fact, the region of the outliers in the image is so large that information cannot be recovered from the regularized vectors. As these outliers sometimes occur during consecutive frames, information can affect to the forecast performances. Therefore, the most suitable method is concluded to be the LK after having in consideration all facts in this analysis.

Appendix A. Unsupervised Parameterization of Velocity Vectors Methods

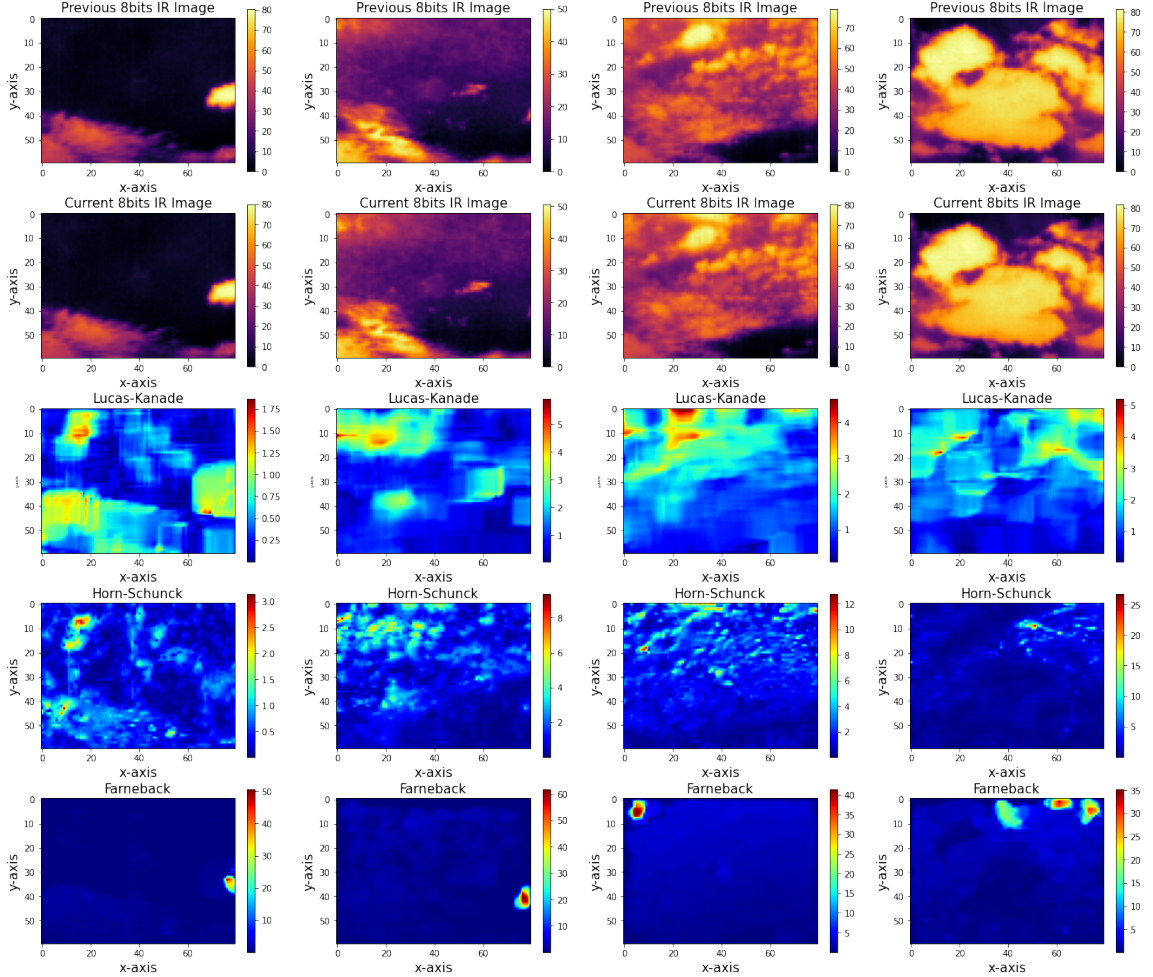


Figure A.3: The shown velocity field was computed from the images in the first row, and the images in the second row, which are organized as different sequences from left to right. The magnitude of the velocity field computed by LK, HS, and Farneback, are shown in the third, fourth and fifth row respectively. In spite of having the lowest error in the validation, the Farneback method is very sensitive to outliers during the implementation.

Appendix B

Wavefunction Probabilities

The wind velocity vector field is defined as the Riemann-Cauchy equations for a complex function $\mathbf{Z} = \Phi + \sqrt{-1} \cdot \Psi$ in each pixel $\mathbf{x}_{i,j}$. However, the maxima of the velocity vector field has to be in the Sun intersecting streamline for our purposes,

$$\begin{aligned}\Phi' &= |\Phi - \Phi_{i_0, j_0}| \\ \Phi'' &= \max[\Phi'] - \Phi'.\end{aligned}\tag{B.1}$$

The corresponding velocity vector field in complex function is $\mathbf{Z} = \Phi'' + \sqrt{-1} \cdot \Psi$, and its probability is modelled as a wave,

$$p(\mathbf{x}_{i,j}|\phi_{i,j}, \psi_{i,j}) = \frac{|\phi''_{i,j} + \sqrt{-1} \cdot \psi_{i,j}|^2}{\|\Phi'' + \sqrt{-1} \cdot \Psi\|_{\mathcal{F}}^2}\tag{B.2}$$

where $\|\cdot\|_{\mathcal{F}}$ is the Frobenius norm.

Appendix C

Convolutional Layers

The convolutional layers used in the NNs in the review of the SOA are formed combining convolutional filters and max-pooling layers. The input variables of a convolutional layer ℓ is two dimensional so that $\mathbf{H}^{(\ell)} \in \mathbb{R}^{D_x^{(\ell)} \times D_y^{(\ell)}}$.

Convolutional filters. The convolutions are preformed using filter banks of sliding windows of size $K_x^{(\ell)} \times K_y^{(\ell)}$, which parameters are learnt applying the backpropagation algorithm (see Section 9.2.1),

$$h_{q,p}^{(\ell+1)} = \rho \left(\mathbf{W}^{(\ell)} \mathbf{H}_{i+m+k,j+n+h}^{(\ell)} \right), \quad \forall i = 1, \dots, D_x^{(\ell)}, \quad \forall j = 1, \dots, D_y^{(\ell)} \quad (\text{C.1})$$

where ρ is a ReLU activation function (see Eq. (9.11)), $\mathbf{W}^{(\ell)} \in \mathbb{R}^{K_x^{(\ell)} \times K_y^{(\ell)}}$ is an squared matrix and $K_x^{(\ell)}, K_y^{(\ell)}$ are even numbers, and m, n are the index of the convolutional sliding window such as $m \in \{-K_x^{(\ell)}/2, \dots, K_x^{(\ell)}/2\}$, $n \in \{-K_y^{(\ell)}/2, \dots, K_y^{(\ell)}/2\}$. The step length between the convolutions are k, h in the x and y direction.

Max-Pooling. This layer uses convolutional filters to select the maximum value within the sliding windows,

$$h_{q,p}^{(\ell+1)} = \max \left(\mathbf{H}_{i+m+k,j+n+h}^{(\ell)} \right), \quad \forall i = 1, \dots, D_x^{(\ell)}, \quad \forall j = 1, \dots, D_y^{(\ell)}, \quad (\text{C.2})$$

Appendix C. Convolutional Layers

where m, n are the index of the convolutional sliding window, which is defined as $m \in \{-K_x^{(\ell)}/2, \dots, K_x^{(\ell)}/2\}$, and $n \in \{-K_y^{(\ell)}/2, \dots, K_y^{(\ell)}/2\}$. $K_x^{(\ell)}$ and $K_y^{(\ell)}$ are even numbers, and k, h are the step length between the convolutions in the x and y direction respectively.

Appendix D

Software Availability

The image fusion and the solar tracking software, which is explained in Chapter 2, is available in the GitHub repository https://github.com/gterren/girasol_machine. This repository also includes the pyranometer, the visible camera and far-IR reading and tracking system commanding software implemented in Python using threading¹ dependency to develop an efficient software that makes use of all CPUs available in the local computer.

The software of the implementation of both geospatial perspective reprojections introduced in Chapter 5, is available in the GitHub repository (https://github.com/gterren/geospatial_perspective_reprojection). The quadratic formula solutions are found using the SciPy² library.

The software necessary for training the parameter models, the atmospheric conditions classification model, plus computing the shifting and amplitude bias, and processing the pyranometer signal and IR camera images is this GitHub repository: https://github.com/gterren/signal_and_image_processing.

¹<https://docs.python.org/3/library/threading.html>

²<https://scipy.org>

Appendix D. Software Availability

The software implemented for the generative and discriminative cloud segmentation models (see Chapter 4) is accessible in the GitHub repository: https://github.com/gterren/cloud_segmentation. The MPI³ dependency was used to run the experiments in parallel in a HPC. The optimization and probability dependencies of SciPy are necessary.

The mixture models dependencies developed for the multiple wind velocity detection software and which was used the experiments shown in Chapter 6, is available in the GitHub repository: https://github.com/gterren/multiple_cloud_detection_and_segmentation. Additionally, the software for the wind velocity field visualization and necessary dependencies for the implementation of the ε -SVM, ε -MT-WSVM and ε -MT-WSVM-FC (Chapter 7), is accessible in the following GitHub repository: https://github.com/gterren/multiple_velocity_fields_visualization.

The software developed for the extraction of features (which are used in Chapter 4-9) and selection (which refers to the computation of probability of an air parcel occluding the Sun's direct radiation explained in Chapter 8 and 9), is available in the GitHub repository: https://github.com/gterren/feature_extraction_and_selection.

The kernel learning methods used in the solar forecasting were implemented using PyTorch⁴ and GPyTorch⁵. The developed software for the multi-task models is accessible in the GitHub repository: https://github.com/gterren/kernel_intra-hour_solar_forecasting.

The different DL architectures used in the solar forecasting experiments were implemented using Keras⁶ and TensorFlow⁷. The BO software was developed using

³<https://mpi4py.readthedocs.io>

⁴<https://pytorch.org>

⁵<https://gpytorch.ai>

⁶<https://keras.io>

⁷<https://tensorflow.org>

Appendix D. Software Availability

the GP dependency available in the Scikit-Learn⁸ library. The software of presented architectures are available in the GitHub repository: https://github.com/gterren/deep_learning_intra-hour_solar_forecasting.

⁸<https://scikit-learn.org>

Glossary

ABBREVIATIONS

ADC	Analog Digital Converter
ANN	Artificial Neural Networks
AIC	Akaike Information Criterion
AR	Auto-Regressive
BeMM	Beta Mixture Model
BGaMM	Bivariate Gamma Mixture Model
BIC	Bayesian Information Criterion
BiRNN	Bidirectional Recurrent Neural Network
BNN	Bayesian Neural Networks
BO	Bayesian Optimization
CARC	Center for Advanced Research Computing
CDF	Cumulative Probability Function
CI	Confidence Intervals
CDLL	Complete Data Log-Likelihood
CLC	Classification Likelihood Criterion
CNN	Convolutional Neural Networks
CPU	Central Processing Unit
CSI	Clear Sky Index
CV	Cross-Validation
DAQ	Data Acquisition
DHCP	Dynamic Host Configuration Protocol
DL	Deep Learning
DN	Direct Normal
ECE	Electrical and Computer Engineering
ELBO	Evidence Lower Bound

Glossary

EM	Expectation-Maximization
FLOPS	Floating point Operations Per Second
FOV	Field Of View
FS	Forecasting Skill
GaMM	Gamma Mixture Model
GCS	Geographic Coordinates System
GDA	Gaussian Discriminant Analysis
GMM	Gaussian Mixture Model
GMT	Greenwich Mean Time
GPC	Gaussian Processes for Classification
GPR	Gaussian Process for Regression
GPU	Graphical Processing Units
GRU	Gated Recurrent Unit
GSI	Global Solar Irradiance
HMM	Hidden Markov Models
HRA	Hour Angle
HS	Horn-Schunck
HPC	High Performance Computer
ICL	Integrated Classification Likelihood
ICM	Iterated Conditional Modes
IR	Infrared
KKT	Karush-Kuhn-Tucker
KL	Kullback-Leibler
KRR	Kernel Ridge Regression
LK	Lucas-Kanade
LOF	Local Outlier Factor
LOO	Leave-One-Out
LS	Least Squares
LST	Local Solar Time
LSTM	Long Short-Term Memory
LSTMe	Local Standard Time Meridian
LT	Local Time
MAE	Mean Absolute Error
MALR	Moist Adiabatic Lapse Rate
MAP	Maximum A Posteriori

Glossary

MAPE	Mean Absolute Percentage Error
MC	Monte Carlo
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MLL	Marginal Log-Likelihood
MLP	Multilayer Perceptron
MT-GPR	Multi-Task Gaussian Process for Regression
MT-KRR	Multi-Task Kernel Ridge Regression
MT-MS-ResRNN	Multi-Task Multiple Source ResRNN
MT-RR	Multi-Task Ridge Regression
MT-RVM	Multi-Task Relevance Vector Machine for Regression
MRF	Markov Random Field
MSE	Mean Squared Error
NBC	Naive Bayes Classifier
NLP	Negative Log-Probability
NN	Neural Network
NWP	Numerical Weather Prediction
PV	Photovoltaic
QP	Quadratic Programming
RAM	Random Access Memory
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
ResBiRNN	Bidirectional Recurrent Neural Network
RGB	Red, Green and Blue
RQ	Rational Quadratic
RMSE	Root Mean Square Error
RNN	Recurrent Neural Networks
ROC	Receiver Operating Characteristic
RR	Ridge Regression
RRC	Ridge Regression for Classification
RVM	Relevance Vector Machine
SA	Simulated Annealing
SOA	State-Of-the-Art
SRN	Simple Recurrent Network
SV	Support Vectors

Glossary

SVC	Support Vector for Classification
SVM	Support Vector Machines for Regression
TC	Time Correction
TSI	Total Sky Imager
UNM	University of New Mexico
UVC	USB Video Class
VMMM	Von Mises Mixture Model
WLK	Weighted Lucas-Kanade
WLS	Weighted Least Squares
WMAE	Weighted Mean Absolute Error
ε -MT-SVM	Multi-Task Support Vector Machine for Regression
ε -MT-WSVM	Multi-Task Weighted Support Vector Machine
ε -MT-WSVM-FC	ε -MT-WSVM with flow constraints

UNITS

μm	Micrometer
cK	Centikelvin
KB	Kilobite
GB	Gigabite
GHz	Gigahertz
hPa	Hectopascals
Hz	Hertz
J	Joules
K	Kelvin
kg	Kilograms
km	Kilometer
m	Meter
m^2	Square meter
min	Minute
mm	Millimeter
$mmHg$	Millimeter of mercury
ms	Millisecond
Pa	Pascals
s	Second
W	Watt

Glossary

◦ Degree

NOMENCLATURE

α	Fusion averaging weight, Sun's azimuth angle, SA cooling parameter, kernel length-scale parameter, regularization term
α_0	Sun's azimuth angle
$\alpha_\ell, \alpha^{(\ell)}$	Shape parameter of a Gamma, Bivariate Gamma and Beta distribution in cluster ℓ
α_i, α_i^*	Dual parameters of sample i
α_x	Field of view in the x-axis
α_y	Field of view in the y-axis
$\bar{\alpha}$	Mean of the posterior distribution
$\bar{\mathbf{I}}$	Image set sample mean
$\bar{\mathbf{U}}$	Matrix containing all the weighted sample mean in \mathbf{U}
$\bar{\mathbf{V}}$	Matrix containing all the weighted sample mean in \mathbf{V}
$\bar{\mathbf{x}}$	Normalized feature vector
$\bar{\mathbf{x}}_c$	Average position of intersecting air parcel in forecasting horizon
c	
$\bar{\mathbf{y}}$	Normalized target vector
$\bar{\mathcal{F}}\{\cdot\}$	Conjugate of the Fourier transform
$\bar{f}(\mathbf{x}_*)$	Mean of a prediction
\bar{h}_ℓ	Estimate height of the pixels of the velocity vectors that belong to cloud layer ℓ
$\bar{t}^{(\ell)}$	Expected time of an air parcel traversing pixel ℓ in the streamline
$\bar{T}_{i,j}''$	Normalized Temperature of a pixel between 0 and 1
$\bar{w}_{i,j}$	Normalized importance weight of sample i, j
$\bar{y}_{i,j}^{(t)}$	Inverse label of $y_{i,j}^{(t)}$
$\beta_\ell, \beta^{(\ell)}$	Rate parameter of a Gamma, Bivariate Gamma and Beta distribution in cluster ℓ
β_c	<i>Luma</i> coding system color coefficients
β	Configuration potential parameter, offset parameter of a polynomial kernel

Glossary

α	Predicted label for class \mathcal{C}_1 , azimuth angles set, Dirichlet distribution parameters of the cluster weight prior in a MAP mixture model
α, α^*	Dual parameters in a multi-task support vector machine
β	Batch normalization layer shift parameters
Δ_x	Numerical differential in the x-axis operator
Δ_y	Numerical differential in the y-axis operator
$\Delta_{x,y}$	Numerical differential operator
ϵ	Small value constant
Γ	Correlation between tasks
γ	Batch normalization layer scale parameters
$\Gamma_{i,j}$	Diagonal matrix of mixture model posterior probability within the sliding window in pixel i, j , cross-correlation $\mathbf{R}_{i,j}$ if the frequency domains
Λ	Matrix with precision parameters in the diagonal
μ	Normal distribution mean vector, mean parameter of the posterior distribution in a RVM, batch normalization layer running mean of $\mathbf{h}^{(\ell)}$
$\mu^{(\ell)}$	Mean vector of a variational distribution
μ_0	Prior distribution mean
$\mu_0^{(\ell)}$	Network layer ℓ prior distribution mean
μ_α	Predictive mean vector
μ_ℓ	Mean parameter of a Multivariate Normal Distribution in cluster ℓ
Ω	Cloud velocity vector angles in all the pixels in an image
$\Omega = \{\gamma, \nu\}$	Kernel functional hyperparameters sets
ω_i	Wind velocity vector of training sample i
ω_m	Vector of structural hyperparameters in BO
ω_{best}	Structural parameters that achieved the lowest error
Φ	Matrix containing training samples previously transformed by an explicit basis functions $\varphi(\cdot)$, wind velocity field Streamline of cloud layer ℓ
π_ℓ	Weight of cluster ℓ in a mixture model

Glossary

Ψ	Potential line of the wind velocity field of cloud layer ℓ
ρ	Coefficients of the theoretical GSI function, imager's location altitude over the sea level
Σ	Normal distribution covariance matrix, covariance of posterior distribution in a RVM
σ_{α}^2	Predictive variance vector
$\Sigma^{(\ell)}$	Covariance matrix of a variational distribution
Σ_0	Prior distribution covariance
$\Sigma_0^{(\ell)}$	Network layer ℓ prior distribution covariance
Σ_n	Laplace approximation distribution covariance
Σ_{ℓ}	Covariance parameter of a Multivariate Normal Distribution in cluster ℓ
Θ	Parameters of the cloud layer distributions
θ	Multivariate normal distribution set of parameters
$\Theta^{(\ell)}$	Parameter set of a variational distribution
$\Theta_0^{(\ell)}$	Parameter set of the prior distribution
θ_{ℓ}	Parameters set of cluster ℓ in a mixture model
Θ_k	Set of parameters in a HMM in time instant k
ε	Set elevation angles for the pixels in the x-axis
$\Delta \bar{t}^{(\ell)}$	Expected time increments
$\Delta \mathbf{X}$	Pixels dimensions given by geospatial reprojection $\psi(\cdot)$
$\Delta \mathcal{E}(\cdot, \cdot)$	Increment of energy in the configuration with inverse label
$\Delta \tau_{GTM}$	Difference between τ_{LT} and τ_{GTM}
Δk	GSI pyranometer measurements shifting bias, small displacement on the time
$\Delta T_{i,j}^{p,q}$	IR image pixel after applying background atmospheric and camera window models
$\Delta T_{i,j}$	Pixel increments temperature after atmospheric and window model application
Δx	Small displacement on the x direction
Δy	Small displacement on the y direction
Δ	Laplace operator

Glossary

δ	Sun's declination angle, piecewise shifting bias model intervals, pixel size, velocity vectors scale, probability of removing row q
$\dot{\mathbf{D}}$	Divergence operator
$\dot{\mathbf{V}}$	Vorticity operator
ℓ	Sky condition label index, neighborhood order index in graph network G , cloud layer index or wind velocity field, length-scale parameter in the simplification of the correlation matrix between task, pixel in the Sun intercepting streamline index, network layer index
ℓ'	Index of $\alpha^{(\ell')}$ cumulative sum
ℓ_i	Sagitta of the chord $C_i D_i$
ϵ	Inner and outer ring radius, infinitely small number
η	Learning rate
$\eta^{(t)}$	Learning rate at optimization iteration t
η_i, η_i^*	Lagrangian functional new variables of sample i
$\exp(\cdot)$	Exponential function
$\Gamma(\cdot)$	Gamma function
γ	Ridge regression regularization parameter, amplitude parameters of a kernel function
$\Gamma'(\cdot)$	Derivative of the Gamma function
$\gamma^{(b)}$	Regularization term of the parameters model b
$\gamma_i^{(t+1)}$	Relevance measure of training sample i
$\gamma_{i,j,\ell}$	Sample weight in the WLK for layer ℓ
$\gamma_{i,\ell}$	Responsibility of cluster ℓ in training sample i
$\gamma_{i,j}$	Responsibility of label cluster k in training sample i
Γ_{MALR}	Moist adiabatic lapse rate
$\hat{\boldsymbol{\alpha}}$	Optimal set of prior parameters
$\hat{\Gamma}_{i,j}$	Normalized cross-correlation $\hat{\mathbf{R}}_{i,j}$ if the frequency domains
$\hat{\boldsymbol{\mu}}_{\mathbf{y}^*}$	Mean in the predictive distribution of \mathbf{y}^*
$\hat{\boldsymbol{\Sigma}}_*$	Covariance of a multi-task prediction
$\hat{\boldsymbol{\Sigma}}_n$	Covariance of the error between tasks
$\hat{\boldsymbol{\sigma}}_n$	Aleatoric uncertainty in the predictive distribution of \mathbf{y}^*
$\hat{\boldsymbol{\sigma}}_p$	Epistemic uncertainty in the predictive distribution of \mathbf{y}^*

Glossary

$\hat{\sigma}_{\mathbf{y}^*}$	Uncertainty in the predictive distribution of \mathbf{y}^*
$\hat{\mu}_{e^*}$	Predictive mean of a new sample e^*
$\hat{\Sigma}_{e^*}$	Predictive Covariance of a new sample e^*
$\hat{\theta}_\ell$	Optimal parameters set of cluster ℓ in a mixture model
\hat{s}_j	Predicted labels of the last ϱ frames
$\hat{\mathbf{R}}_{i,j}$	Normalized cross-correlation between $\mathcal{I}_1(i, j)$ and $\mathcal{I}_2(i, j)$
$\hat{\mathbf{U}}_\ell$	Cloud velocity vectors x-axis components of the cluster ℓ
$\hat{\mathbf{U}}_{\ell,k}$	Wind velocity field x-components of cloud layer ℓ in frame k
$\hat{\mathbf{V}}$	Estimation of the cloud velocity vectors
$\hat{\mathbf{V}}_{\ell,k}$	Wind velocity field y-components of cloud layer ℓ in frame k
$\hat{\mathbf{V}}_{\ell,k}^*$	Selected velocity vectors
$\hat{\mathbf{V}}'$	Selected cloud velocity vectors with larges normalized increments
$\hat{\mathbf{V}}''$	Matrix of segmented velocity vectors of ϱ past time instants
$\hat{\mathbf{V}}_\ell$	Cloud velocity vectors u-axis components of the cluster ℓ
$\hat{\mathbf{v}}_{i,j,\ell} = \{\hat{\mathbf{u}}_{i,j,\ell}, \hat{\mathbf{v}}_{i,j,\ell}\}$	Estimation of the cloud velocity vectors in pixel i, j in layer ℓ using WLK
$\hat{\mathbf{v}}_{i,j} = (\hat{\mathbf{u}}_{i,j}, \hat{\mathbf{v}}_{i,j})$	Approximation of the cloud velocity vectors using LK
$\hat{\mathbf{W}}$	Approximate ion of the wind velocity field
$\hat{\mathbf{w}}$	Estimated model parameters, Laplace approximation distribution mean
$\hat{\mathbf{W}}'$	Wind velocity vectors in the Sun intercepting streamline
$\hat{\mathbf{W}}_{\ell,k}$	Wind velocity field of cloud layer ℓ in frame k
$\hat{\mathbf{y}}$	Predicted label vector
$\hat{\mathbf{y}}^*$	Multi-tasks vector of prediction for a new sample \mathbf{x}^*
$\hat{\mathbf{y}}_k$	Actual prediction from the network
$\hat{\nu}$	Optimal parameters of amplitude bias model
$\hat{\rho}$	GSI physical model optimal parameters
$\hat{\zeta}'_j$	Mode of atmospheric condition model predictions sequence $\hat{\zeta}_j \cdots \hat{\zeta}_{j-\varrho+1}$
$\hat{\zeta}_j$	Predicted label of the atmospheric condition model for a feature vector
\hat{L}_k	Optimal Number of clusters in a mixture model and latent variable of the HMM in time instant k
$\hat{t}^{(\ell)}$	Sun intercepting time of air parcel in pixel ℓ of the streamlines
$\hat{w}_{i,j}$	Sample draw from uniform distribution

Glossary

$\hat{w}_{q,\ell,k}$	Normalized probabilities $w_{q,\ell,k}$
$\hat{x}_{i,j}$	Chord form by the α_y in the circle form by great circle chord
	$E_i D_i$
\hat{y}	GSI measurements corrected with amplitude bias
\hat{y}^*	Corresponding estimation for a new sample
\hat{y}_i	Estimation of y_i
\hat{y}_i	Great circle segment
\hat{y}_{sup}	Smallest great circle segment form by the α_x
∞	Infinity
κ_ℓ	Concentration parameter of Von Mises distribution in cluster ℓ
λ	Longitude, binary classification threshold, number of parameters
	in a mixture model
λ_1, λ_2	Eigenvalues of $\mathbf{A}_{i,j}$
λ_j	Precision in the prior distribution of the primal parameter w_j of
	relevance vector machine
λ_q	Index of the cloud velocity vector q that belongs to layer 1
$\lambda_{i,j}$	Chord $\hat{x}_{i,j}$ sagitta
\lim	Value of a function when approaches to ϵ
$\log(\cdot)$	Natural logarithm function
\mathbb{B}	Binary number
$\mathbb{C}(\cdot)$	Covariance
$\mathbb{E}(\cdot)$	Expectation
$\mathbb{I}(\cdot)$	Indicator function
\mathbb{N}	Natural number
\mathbb{R}	Real number
\mathbb{R}^+	Strictly positive real number
\mathbb{R}^D	Real number with D dimension
$\mathbb{R}^{\leq 1}$	Real number strictly equal or small than 1
$\mathbb{V}(\cdot)$	Variance
$\mathbf{0}$	Matrix of zeros, vector of zeros
$\mathbf{1}$	Vector of ones
\mathbf{a}_k	Vector of Sun's position angles
$\mathbf{A}_{i,j}$	Matrix with all observations LS problem
\mathbf{B}	Covariance of the posterior distribution

Glossary

\mathbf{b}	Bias parameters of a multi-task model
$\mathbf{b}_{\ell,k}, \mathbf{b}_k^{(\ell)}$	Bias parameters of cloud layer ℓ in frame k in a multi-task model, bias parameter of the network layer ℓ
$\mathbf{b}_{i,j}$	Target vector in LS problem
$\mathbf{c}^{(\ell)}$	Cell units of layer ℓ in a recurrent network
\mathbf{D}	Divergence of the cloud velocity vectors
\mathbf{d}_k	Cloud velocity vectors divergence statistic for each forecasting horizon c
$\mathbf{e} = \{e_{\hat{u}}, e_{\hat{v}}\}$	Root mean square error in the approximation of the wind velocity field in the \hat{u} and \hat{v} component respectively
$\mathbf{f}^{(\ell)}$	Forget gate units of layer ℓ in a recurrent network
$\mathbf{H}^{(\ell)}$	Hidden units in a convolution and a max-pooling layer
$\mathbf{h}^{(\ell)}$	Hidden variables of layer ℓ
\mathbf{h}_k	height statistic for each forecasting horizon c
\mathbf{I}	Raw image, 8 bit normalized pixels intensities of an infrared im- age after applying the atmospheric and window models, identity Matrix
\mathbf{I}'_k	Finite differentials of an image in time
\mathbf{I}'_x	Finite differentials of an image in the x direction
\mathbf{I}'_y	Finite differentials of an image in the y direction
$\mathbf{i}^{(\ell)}$	Input gate units of layer ℓ in a recurrent network
\mathbf{I}_k	Normalized infrared image
\mathbf{K}	Gram matrix
$\mathbf{k}(\mathbf{x}^*)$	Vector of dot produces between a new sample and the traning samples
\mathbf{k}^*_{Ω}	Vector resulting of evaluating the kernel function with the train- ing samples and a new sample
\mathbf{k}^{**}_{Ω}	Scalar resulting of evaluating the kernel function with a new sample
\mathbf{K}_k	Differential kernel in the time
\mathbf{K}_x	Differential kernel in the x direction
\mathbf{K}_y	Differential kernel in the y direction
\mathbf{K}_{Ω}	Gram matrix evaluated with the hyperameters set Ω

Glossary

\mathbf{K}_{HS}	HS Differential kernel
\mathbf{M}	Cloud velocity vector magnitude in all the pixels in an image
\mathbf{M}_e	Mask for exposition time e
\mathbf{m}_k	Cloud velocity vectors magnitude statistic for each forecasting horizon c
$\mathbf{o}^{(\ell)}$	Output gate units of layer ℓ in a recurrent network
\mathbf{R}_e^1	Outer ring smoothing mask for exposure time e
\mathbf{R}_e^2	Inner ring smoothing mask for exposure time e
$\mathbf{r}^{(\ell)}$	Reset units of layer ℓ in a recurrent network
$\mathbf{R}_{i,j}$	Cross-correlation between $\mathcal{I}_1(i, j)$ and $\mathcal{I}_2(i, j)$
\mathbf{S}_c	Uncertainty in the position of intersecting air parcel at forecasting horizon c
\mathbf{t}_k	Temperature statistic for each forecasting horizon c
$\mathbf{u}^{(b)}$	Feature vectors of the parameters model b
\mathbf{V}	Vorticity of the cloud velocity vectors
\mathbf{v}_k	Cloud velocity vectors vorticity statistic for each forecasting horizon c
$\mathbf{v}_{i,j}$	Vector containing the finite time differentials of the pixels within the sliding window in pixel i, j
\mathbf{W}	Germanium outdoor camera window persistent model, primal parameters of a multi-task model
\mathbf{w}	Primal model parameters
$\mathbf{W}_{i,j}^{(1)}$	Fourier transform of the pixels within the sliding winding of the first image
$\mathbf{W}_{i,j}^{(2)}$	Fourier transform of the pixels within the sliding winding of the second image
$\mathbf{W}_{i,j}^{(2)*}$	Conjugate of $\mathbf{W}_{i,j}^{(2)}$
$\mathbf{W}_k^{(\ell)}$	Parameters of network layer ℓ
$\mathbf{w}^{(b)}$	Coefficients of the parameters model b
\mathbf{w}_ℓ	Coefficients of the multi-class atmospheric condition of class ℓ in the one-versus-all classifier
\mathbf{W}_h	Weights in a recurrent network
\mathbf{W}_i	Input weights

Glossary

\mathbf{W}_z	Input weights in a recurrent network
$\mathbf{W}_{\tilde{c}}$	Main weights in a recurrent network
\mathbf{W}_f	Forget weights in a recurrent network
\mathbf{W}_o	Output weights in a recurrent network
\mathbf{W}_r	Reset weights in a recurrent network
\mathbf{W}_{sc}	Shortcut weights in a residual layer
\mathbf{X}	Coordinates of the pixels in an image reprojected in the atmosphere cross-section plane
$\mathbf{x}^*, \mathbf{x}_*$	A new sample
$\mathbf{X}_{\ell,k}^*$	Coordinates of the selected velocity vectors
\mathbf{X}'	Coordinates of the pixels in the Sun intercepting streamline reprojected in the atmosphere cross-section plane
\mathbf{X}^k	Visible image after fusion
$\mathbf{x}_0 = \{i_0, j_0\}$	Sun's position image index
\mathbf{x}_i	Feature vector
\mathbf{x}_k	Vector of features extracted for sample k
$\mathbf{X}_{i,j}$	Matrix containing the finite differentials in the x and y direction of the pixels within the sliding window in pixel i, j
$\mathbf{x}_{i,j}$	Feature vector
\mathbf{y}	Label vector
\mathbf{y}^*	Model prediction of new sample \mathbf{x}^*
\mathbf{y}_k	Vector of last CSI measurements, multi-task prediction targets
$\mathbf{y}_{i,j}$	Velocity vector in pixel i, j
\mathbf{y}_{k+1}	Forecasting target for sample k
\mathbf{Z}	Wind velocity field as a complex function
\mathbf{z}	Atmospheric conditions model feature vector
$\mathbf{z}^{(\ell+1)}$	Weighted sum of connections in layer ℓ
$\mathcal{AIC}(\cdot)$	Akaike information criterion function
$\mathcal{A}(\cdot)$	Model of atmospheric radiation in the IR images
$\mathcal{BG}(\cdot)$	Bivariate Gamma Distribution
$\mathcal{BIC}(\cdot)$	Bayesian information criterion function
$\mathcal{B}(\cdot)$	Beta Distribution
$\mathcal{CLC}(\cdot)$	Classification likelihood criterion function
\mathcal{C}	SVM complexity parameter

Glossary

\mathcal{C}_ℓ	Complexity term of label ℓ in the one-versus-all SVC
\mathcal{C}_k	Pixel label
\mathcal{D}	Training dataset
$\mathcal{D}(\cdot)$	Model of direct radiation in the IR images
$\mathcal{E}(\cdot)$	Error function, energy function
$\mathcal{F}\{\cdot\}$	Fourier transform
$\mathcal{F}^{-1}\{\cdot\}$	Inverse Fourier transform
$\mathcal{G}(\cdot)$	Gamma Distribution
\mathcal{H}	Hilbert space defined by a dot product
$\mathcal{H}(\cdot)$	Entropy function in information theory
$\mathcal{ICL}(\cdot)$	Integrated classification likelihood function
\mathcal{I}	Intensity of a object in an image
$\mathcal{I}(\cdot, \cdot)$	Theoretical GSI funtion
$\mathcal{I}_1(i, j)$	First image in a sequence of consecutive images
$\mathcal{I}_2(i, j)$	Second image in a sequence of consecutive images
\mathcal{I}_k	Partial derivative of \mathcal{I} with respect to k
\mathcal{I}_x	Partial derivative of \mathcal{I} with respect to x
\mathcal{I}_y	Partial derivative of \mathcal{I} with respect to y
$\mathcal{I}_{Diffuse}$	Diffuse GSI component
\mathcal{I}_{Direct}	Direct GSI component
\mathcal{I}_{DN}	Direct normal GSI
$\mathcal{I}_{Reflected}$	Reflected GSI component
$\mathcal{K}(\cdot, \cdot)$	Kernel function
$\mathcal{K}(\cdot, \cdot)_M^\nu$	Matén kernel for order ν
$\mathcal{K}(\cdot, \cdot)_{\mathcal{P}^n}$	Polynomial kernel of order n
$\mathcal{K}(\cdot, \cdot)_L$	Linear kernel
$\mathcal{K}(\cdot, \cdot)_{RBF}$	Radial basis functions kernel
$\mathcal{K}(\cdot, \cdot)_{RQ}$	Rational quadratic kernel
$\mathcal{L}(\cdot)$	Network loss function
$\mathcal{M}(\cdot)$	Mask function
$\mathcal{N}(\cdot)$	Gaussian kernel, Normal distribution
$\mathcal{PW}(\cdot, \cdot, \cdot)$	Piecewise shifting model
\mathcal{P}^n	Dimensions in a polynomial expansion of order n
$\mathcal{Pe}(\cdot, \cdot)$	Periodic amplitude model
$\mathcal{Q}(\cdot)$	Complete Data Log-Likelihood

Glossary

\mathcal{S}	Index pairs set of pixels in the Sun intercepting streamline
$\mathcal{S}(\cdot)$	Model of scattered radiation in the IR images
$\mathcal{T}(\cdot)$	Equation of time
$\mathcal{U}(\cdot, \cdot)$	Uniform distribution
$\mathcal{VM}(\cdot)$	Von Mises Distribution
$\mathcal{V}(\cdot, \cdot)$	Evidence lower bound function
\mathcal{W}	Clear sky IR images set, domain of the structural parameters ω
$B(\cdot)$	Beta function
$\text{corr}(\cdot)$	Correlation mathematical function
$\text{diag}(\cdot)$	Matrix with vector in the diagonal
$\text{Dir}(\cdot)$	Dirichlet distribution
$\text{median}(\cdot)$	Function to calculate the median value of i, j
$\text{mode}(\cdot)$	Mode of the elements in $\hat{\zeta}_j$
$\text{sign}(\cdot)$	Sign mathematical function
$\text{sort}(\cdot)$	Sorting the elements in a vector from low to high
$\text{vec}(\cdot)$	Vectorize a matrix
$\mu(\cdot)$	Predictive mean
μ_ℓ	Mean parameter of a Normal and Von Mises distribution in cluster ℓ , mean height in cloud layer ℓ
$\mu_{i,j}$	Sample mean of the pixels within the convolutional implementation of a sliding window
$\mu_{j,k}$	Mean of label k and feature j
$\nabla EI(\omega^{(t)})$	Numerical differentiation of $EI(\omega^{(t)})$
∇	Nabla operator
ν	Parameters of amplitude bias model, field of view of a pixel, smoothing parameter of a Mat��n kernel
\odot	Element-wise multiplication (Hadamard product)
Ω_ℓ	Set of cliques in neighborhood of order ℓ
\otimes	Kroneker product
$\phi(\cdot)$	Probit approximation of the sigmoid function, an activation function, probability density function of standard normal distribution
ϕ	Latitude

Glossary

$\Phi(\cdot)$	Cumulative density function of standard normal distribution
π	Number pi
π_k	Weight of label cluster k
$\psi(\cdot)$	System's configuration potential energy, Digamma Function, Geospatial perspective reprojection, Hyperbolic tangent activation function
$\rho(\cdot)$	Rectified linear unit activation function
ρ	Pearson coefficient, Simulated Annealing acceptance probability
$\rho_{i,j,1}, \rho_{i,j,2}$	Index of pixel's heights i, j that belongs to layer 1 or 2
$\sigma(\cdot)$	Sigmoid activation function, predictive standard deviation
σ	Gaussian kernel standard deviation, GSI pyranometer measurements amplitude bias, amplitude parameter of the numerical time differential kernel
σ^2	Heights in cloud layer ℓ variance
σ_*^2	Prediction variance
σ_n^2	Error variance
Σ_p	Prior distribution covariance matrix
$\Sigma_{11}, \Sigma_{12}, \Sigma_{21}, \Sigma_{22}$	Correlation between velocity components
$\sigma_{i,j}$	Sample standard deviation of the pixels within the convolutional implementation of a sliding window
$\sigma_{j,k}$	Feature j with label k standard deviation
$\sqrt{-1}$	Imaginary number
\star	Convolution operation
τ	Sky conditions classification threshold, coefficient regularization parameter of the WLS, larges normalized increments between two consecutive an image segmentation threshold, eigenvalues threshold, optimization convergence threshold
τ_{GTM}	Greenwich Mean Time
τ_{HRA}	Hour Angle
$\tau_{lower}, \tau_{upper}$	Lower and upper noisy cloud velocity vector threshold
τ_{LSTM_e}	Local Standard Time Meridian
τ_{LST}	Local Solar Time
τ_{LT}	Local Time

Glossary

τ_{SR}	Sunrise
τ_{SS}	Sunset
τ_{TC}	Time Correction factor
$\Theta(\cdot)$	Model with a polynomial expansion of order n applied to the features
$\theta^{(b)}$	Parameters of the background atmospheric model
$\Theta^{(b)}(\cdot)$	Desired function of $\mathbf{u}^{(b)}$
$\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\alpha}}^*$	Dual parameters of a multi-task model in vector form
$\tilde{\boldsymbol{\alpha}}_{\ell,k}, \tilde{\boldsymbol{\alpha}}_{\ell,k}^*$	Dual parameters of cloud layer ℓ in frame k in a multi-task model
$\tilde{\boldsymbol{\mu}}$	Mean of the primal parameter posterior distribution in a multi-task relevance vector machine
$\tilde{\boldsymbol{\omega}}$	Wind velocity vectors components in vector form
$\tilde{\boldsymbol{\omega}}_{\ell,k}$	Wind velocity components of cloud layer ℓ in frame k in a multi-task model
$\tilde{\boldsymbol{\Sigma}}$	Covariance of the primal parameter posterior in a multi-task relevance vector machine
$\tilde{\mathbf{u}}_{\ell,k}$	Wind velocity vector x-component of cloud layer ℓ in frame k in a multi-task model
$\tilde{\mathbf{v}}_{\ell,k}$	Wind velocity vector y-component of cloud layer ℓ in frame k in a multi-task model
$\tilde{\mathbf{C}}$	Matrix of labels corresponding to a training set in one-versus-all notation
$\tilde{\mathbf{c}}^{(\ell)}$	Main units of layer ℓ in a recurrent network
$\tilde{\mathbf{I}}^k$	Normalized visible image after fusion
$\tilde{\mathbf{K}}$	Gram matrix of a multi-task model
$\tilde{\mathbf{M}}_e$	Mask after applying Gassing kernel convolution
$\tilde{\mathbf{W}}$	Primal parameters in a multi-task support vector machine
$\tilde{\mathbf{y}}$	Multi-task observation in vector form
$\tilde{\zeta}_{i,\ell}$	Feature vector labels for the class ℓ in the one-versus-all in the atmospheric condition model
\tilde{c}_i	Complexity parameters of training sample i in multi-task model
$\tilde{T}_{i,j}''$	Normalized Temperature of a pixel between 0 and ∞
$\tilde{w}_{q,\ell,k}$	Cumulative density function of $\hat{w}_{q,\ell,k}$

Glossary

\tilde{z}_i	Importance weight of training sample i in multi-task model
\times	Cross-product
Υ	Set of index of the pixels in the neighborhood
ε	Sun's elevation angle, covariance matrix regularization, SVM loss function parameter
ε_0	Sun's elevation angle
ε_i	Error in the estimation of a training sample y_i
ε_k	Error in the prediction of \mathbf{y}_k
$\varphi(\cdot)$	A polynomial expansion of order n , joint distribution of a class, any explicit basis function, transformation into a Hilbert space
ϱ	Lag in a time series
ς_i	Labels of the feature vectors in the atmospheric condition model
$\vartheta^{(t)}$	Steepest descent parameter being optimize in iteration t
ξ	Trade-off parameter in EI acquisition function
ξ_i, ξ_i^*	Slack variables of sample i
ζ	Sun's zenith angle, parameters of piecewise shifting bias model
a	Any polynomial expansion coefficients, quadratic equation coefficient
a_ℓ	Parameter of a Bivariate Gamma distribution in cluster ℓ
$a_x^{(\ell)}, a_y^{(\ell)}$	Air parcel transition coefficients
AB	Imager's normal plane
B	Number of training batch
b	quadratic equation coefficient, bias parameter, a training batch
$b_{i,j}$	Binary segmentation of pixel i, j
C	Number of color component, number of forecasting horizons in a multi-task model
c	Color component index, quadratic equation free coefficient, forecasting horizons index
c_i	Complexity parameter of training sample i
$C_i D_i$	Chord formed by saggita ℓ_i
D	Dimensions of a fisheye raw image, number of days with clear sky training images, feature vector \mathbf{x}_i dimensions, distance from the lens to a converging point, feature vector \mathbf{i} dimensions

Glossary

d	Day since the beginning of the year, clear sky day index
$D^{(\ell)}$	Dimensions of the hidden variables in layer ℓ
$D_i E_i$	Great circle chord form by the imager's elevation angle ε_i
d_p	Sorted from lower to largest normalized increments
$D_x^{(\ell)}$	Dimension of the hidden frame ℓ in the x-axis
$D_y^{(\ell)}$	Dimension of the hidden frame ℓ in the y-axis
$d_{i,j}$	Normalized increments between two consecutive normalized images
E	Exposures times, global energy
e	Exposure time index
$E(\cdot)$	Root mean squared error
e^*	Validation error prediction
e_m	Sample m target validation error in BO
e_t	Error in the estimation of the time
$e_{\Delta \mathbf{x}', c}$	Uncertainty in the intersection position
$e_{\hat{u}, i}$	A sample drawn from $\mathcal{N}(0, e_{\hat{u}})$
$e_{\hat{v}, i}$	A sample drawn from $\mathcal{N}(0, e_{\hat{v}})$
e_{best}	Lowest error
$EI(\cdot)$	Expected Improvement acquisition function
f	Focal length
$f(\cdot)$	Desired function
f_r	Frame rate in a sequence of consecutive images
FN	False negative number
FOV	Imager's diagonal field of view
FP	False positive number
h	Cloud height
$H'_{i,j}$	Pixel height after atmospheric model application
$H''_{i,j}$	Pixel height after atmospheric and window model application
h_ℓ	Average height of the pixels in cluster ℓ
$H_{i,j}$	Raw pixel i, j height
i	Image horizontal pixel index, training sample, index sample drawn from a distribution, time series sample index before k
$i_{d,k}^{csi}$	time instant in backpropagation through time CSI measurements of day d , sample k

Glossary

$I_{i,j}^{p,q}$	8 bit normalization of $\Delta T_{i,j}^{p,q}$
i_0, j_0	Indexes of the Sun's position
$I_\nu(\cdot)$	Modified Bessel function of order ν
$i_{d,k}$	Theoretical GSI of day d , sample k
$I_{p,\ell,k}$	Indexes of samples in $\tilde{\mathbf{w}}_{\ell,k}$ closer to the samples $z_{p,\ell,k}$
J	J-statistic
j	Image vertical pixel index, arbitrary testing sample, dimension index, recurrent network index
K	number of labels, number of samples in the training time series
k	Index of image number of a given day d used for clear sky training images, time instant in a sequence, label index
K_ν	Second kind modified Bessel function of order ν
K_b	Number of samples in a training batch
K_d	Clear sky training images in day d
$K_x^{(\ell)}, K_y^{(\ell)}$	Dimension of the convolutional sliding window in the x-axis and y-axis
L	Number of sky condition labels, largest neighborhood in graph network G , number of cloud layers, number of pixels in the Sun intercepting streamline, number of layers in the network
L_k	Number of clusters in a mixture model and latent variable of the HMM in time instant k
L_R	Number of recurrent layers
L_{MLP}	Number of dense layers
M	Vertical pixels in a frame, number of samples in BO
m	Index of the vertical coordinate of pixel in the sliding window, accumulate normalized increments in an images index, pixel index to integrate applying trapezoid rule, a training sample in BO
m, n	Index of the pixel's neighborhood window centered in $i^{(t)}, j^{(t)}$
m_c^X	Average of feature x in forecasting horizon c
N	Frame acquired at instant k , Horizontal pixels in a frame, number of training samples, number of random samples in BO

Glossary

n	Index of the horizontal coordinate of pixel in the sliding window
N^*	Number of select cloud velocity vectors defined by the used
N_{diag}	Number of pixels in the diagonal
p	Day index for an arbitrary image, sorted normalized increments index
P^{atm}	Atmospheric pressure on the ground
q	Image index for an arbitrary day p
$q(\bar{\mathbf{w}})$	Laplace approximation distribution
$q(\cdot)$	Variational Distribution
q, p	Indexes of the elements on $\mathbf{W}^{(\ell)}$
Q_k	Total number of segmented velocity vectors
R	Great circle radius, number of layer bypassed by a residual layer
$r^{p,q}$	Average GSI sky index in the last t frames
r_e	Radius of the exposition time to define a mask
r_m	Accumulate normalized increments
r_{earth}	Radius of the Earth (small circle)
S	Images in the clear sky set, samples drawn from the distribution of $e_{\hat{u}}$ and $e_{\hat{v}}$ to approximate $t^{(\ell)}$, number of samples drawn from a variational distribution
s_c^X	Standard deviation of feature x in forecasting horizon c
s_i	Radius of the circle form by the sphere slice at chord $D_i E_i$
$s_{i,j}$	Binary segmentation of the pixels in an image by normalized increments
t	Steepest descent iteration, SA iteration, EM iteration, ICM iteration, network optimization iteration, global energy function optimization iteration
$t^{(\ell)}$	Time of an air parcel traverse pixel ℓ in the streamline
$T^{(t)}$	Simulated Annealing cooling mechanism temperature
$T_{i,j}^{p,q}$	$\Delta T_{i,j}^{p,q}$ after adding the temperature level of the tropopause
$T_{i,j}''$	Temperature of a pixel i, j in Kelvin after applying the atmospheric and windown radiation models
$T_{i,j}^{p,q}$	IR image pixel after applying the background atmospheric model
T^{air}	Air temperature on the ground

Glossary

T^{dew}	Dew temperature on the ground
t_c	Sun intersecting air parcel in forecasting horizon c
$T_{i,j}$	Pixel i, j raw radiometric measurements
TN	True negative number
TP	True positive number
u	Sample drawn from an uniform distribution, velocity component in the x direction of an object in an image
u_i	Wind velocity vector x-component of sample i
v	Velocity component in the y direction of an object in an image
v_i	Opposite side of the triangle form by the elevation angle ε_i and imager's normal plane AB , wind velocity vector y-component of sample i
W	Dimension of the convolutional implementation of a sliding window
w	Sliding window size parameter
$w_c^{(\ell)}$	Probability of a pixels in the streamline intersecting with the Sun
w_i	Adjacent side of the triangle form by the elevation angle ε_i and imager's normal plane AB
$w_{i,j}$	Sample i, j importance weight
$w_{q,\ell,k}$	Probability of cloud velocity vector q of time instant k with respect to the parameters of cloud layer ℓ
X	A random variable
x	Horizontal distance from i in a Gaussian filter, object position on the x direction
$x'_{i,j}$	Reprojection of pixel i, j dimension in the x-axis
x'_0	Sun's position in the x-axis of the atmosphere cross-section plane
x_i	A feature vector
$x_{i,j}$	Pixels set of index with the sliding window
y	Vertical distance from j in a Gaussian filter, GSI measurements, object position on the y direction
y'	GSI pyranometer measurements corrected with amplitude and shifting biases

Glossary

y_i	Sample i label, sample i target
$y'_{i,j}$	Reprojection of pixel i, j dimension in the y-axis
y'_0	Sun's position in the y-axis of the atmosphere cross-section plane
$y_{i,j}$	Pixel i, j dimension in the y-axis
$z^*_{q,\ell,k}$	Probability of the selected velocity vector q with respect to cloud layer ℓ
z_i	Distance of pixel i to the cross-section atmosphere plane, importance weight of training sample i
$z_{i,j,c}$	Sun intersecting probability of the air parcel in pixel i, j in forecasting horizon c
$z_{p,\ell,k}$	Samples from a Uniform distribution
$\ \cdot\ $	L-2 norm
$KL(\cdot\ \cdot)$	Kullback-Leibler divergence between two distribution
$\ \cdot\ _{\mathcal{F}}$	Frobenius norm

References

- [1] L. Adrian, R.J. Adrian, and J. Westerweel. *Particle Image Velocimetry*. Cambridge Aerospace Series. Cambridge University Press, 2011.
- [2] L. Mazorra Aguiar, B. Pereira, P. Lauret, F. Díaz, and M. David. Combining solar irradiance measurements, satellite-derived data and a numerical weather prediction model to improve intra-day solar forecasting. *Renewable Energy*, 97:599 – 610, 2016.
- [3] R. Ahmed, V. Sreeram, Y. Mishra, and M.D. Arif. A review and evaluation of the state-of-the-art in pv solar power forecasting: Techniques and optimization. *Renewable and Sustainable Energy Reviews*, 124:109792, 2020.
- [4] M. A. Aizerman, E. M. Braverman, and L.I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and remote Control*, 25:821–837, 1964.
- [5] Mark A Aizerman, Emmanuil M Braverman, and Lev I Rozonoer. Theoretical foundations of potential function method in pattern recognition. *Automation and Remote Control*, 25(6):917–936, 1964.
- [6] Meenu Ajith and Manel Martínez-Ramón. Deep learning based solar radiation micro forecast by fusion of infrared cloud images and radiation data. *Applied Energy*, 294:117014, 2021.
- [7] Hirotugu Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19(6):716–723, 12 1974.
- [8] Muhammad Naveed Akhter, Saad Mekhilef, Hazlie Mokhlis, and Noraisyah Mohamed Shah. Review on forecasting of photovoltaic power generation based on machine learning and metaheuristic techniques. *IET Renewable Power Generation*, 13(7):1009–1023, 2019.

References

- [9] J. Alonso, F.J. Batlles, G. López, and A. Ternero. Sky camera imagery processing based on a sky classification using radiometric data. *Energy*, 68:599 – 608, 2014.
- [10] J. Alonso-Montesinos and F.J. Batlles. The use of a sky camera for solar radiation estimation based on digital image processing. *Energy*, 90:377 – 386, 2015.
- [11] J Alonso-Montesinos, Jesús Polo, Jesús Ballestrín, FJ Batlles, and C Portillo. Impact of dni forecasting on csp tower plant power production. *Renewable Energy*, 138:368–377, 2019.
- [12] R. Alonso-Suárez, M. David, V. Branco, and P. Lauret. Intra-day solar probabilistic forecasts including local short-term variability and satellite information. *Renewable Energy*, 158:554 – 573, 2020.
- [13] H. C. Andrews and C. L. Patterson. Digital interpolation of discrete images. *IEEE Transactions on Computers*, C-25(2):196–202, 2 1976.
- [14] Javier Antonanzas, Natalia Osorio, Rodrigo Escobar, Ruben Urraca, Francisco J Martinez-de Pison, and Fernando Antonanzas-Torres. Review of photovoltaic power forecasting. *Solar Energy*, 136:78–111, 2016.
- [15] Clara Arbizu-Barrena, José A. Ruiz-Arias, Francisco J. Rodríguez-Benítez, David Pozo-Vázquez, and Joaquín Tovar-Pescador. Short-term solar radiation forecasting by advecting and diffusing msg cloud index. *Solar Energy*, 155(Supplement C):1092 – 1103, 2017.
- [16] Nachman Aronszajn and Kennan T Smith. Invariant subspaces of completely continuous operators. *Annals of Mathematics*, pages 345–350, 1954.
- [17] Kuk Yeol Bae, Han Seung Jang, and Dan Keun Sung. Hourly solar irradiance prediction based on support vector machine and its error analysis. *IEEE Transactions on Power Systems*, 32(2):935–945, 2017.
- [18] Simon Baker, Ralph Gross, Takahiro Ishikawa, and Iain Matthews. Lucas-kanade 20 years on: A unifying framework: Part 2. *International Journal of Computer Vision*, 56:221–255, 2003.
- [19] Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *J. Mach. Learn. Res.*, 6:1345–1382, December 2005.
- [20] Florian Barbieri, Sumedha Rajakaruna, and Arindam Ghosh. Very short-term photovoltaic power forecasting with cloud modeling: A review. *Renewable and Sustainable Energy Reviews*, 75:242–263, 2017.

References

- [21] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.
- [22] Jasmine S Bartlett, Áurea M Ciotti, Richard F Davis, and John J Cullen. The spectral effects of clouds on solar irradiance. *Journal of Geophysical Research: Oceans*, 103(C13):31017–31031, 1998.
- [23] Faisal I Bashir, Ashfaq A Khokhar, and Dan Schonfeld. Object trajectory-based activity classification and recognition using hidden markov models. *IEEE transactions on Image Processing*, 16(7):1912–1919, 2007.
- [24] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18, 2018.
- [25] Enrica Bellone, James P Hughes, and Peter Guttorp. A hidden markov model for downscaling synoptic atmospheric patterns to precipitation amounts. *Climate research*, 15(1):1–12, 2000.
- [26] Paolo Bertoldi. Chapter 4.3 - overview of the european union policies to promote more sustainable behaviours in energy end-users. In Marta Lopes, Carlos Henggeler Antunes, and Kathryn B. Janda, editors, *Energy and Behaviour*, pages 451–477. Academic Press, 2020.
- [27] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society B*, 48(3):48–259, 1986.
- [28] Jan Beyea. The smart electricity grid and scientific research. *Science*, 328(5981):979–980, 2010.
- [29] P. Bharadwaj and L. Carin. Infrared-image classification using hidden markov trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(10):1394–1398, 2002.
- [30] Muhammad Syazwan Rifdi Bin Mohd Rashid, Jinghong Zheng, Ernest Sng, Kurinji Malar Rajendhiran, Zhen Ye, and Li Hong Idris Lim. An enhanced cloud segmentation algorithm for accurate irradiance forecasting. *Solar Energy*, 221:218–231, 2021.
- [31] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [32] Dmitrii Bogdanov, Manish Ram, Arman Aghahosseini, Ashish Gulagi, Ayobami Solomon Oyewo, Michael Child, Upeksha Caldera, Kristina Sadovskaia,

References

- Javier Farfan, Larissa De Souza Noel Simas Barbosa, Mahdi Fasihi, Siavash Khalili, Thure Traber, and Christian Breyer. Low-cost renewable electricity as the key driver of the global energy transition towards sustainability. *Energy*, 227:120467, 2021.
- [33] Edwin V Bonilla, Kian Chai, and Christopher Williams. Multi-task gaussian process prediction. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 153–160. Curran Associates, Inc., 2008.
- [34] Léon Bottou and Vladimir Vapnik. Local learning algorithms. *Neural computation*, 4(6):888–900, 1992.
- [35] O. Boucher, D. Randall, P. Artaxo, C. Bretherton, G. Feingold, P. Forster, V.-M. Kerminen, Y. Kondo, H. Liao, U. Lohmann, P. Rasch, S. K. Satheesh, S. Sherwood, B. Stevens, and X. Y. Zhang. *Clouds and aerosols*. Cambridge University Press, Cambridge, UK, 2013.
- [36] Olivier Bousquet, Pierre Tabary, and Jacques Parent du Châtelet. On the value of operationally synthesized multiple-doppler wind fields. *Geophysical research letters*, 34(22), 2007.
- [37] Olivier Bousquet, Pierre Tabary, and Jacques Parent du Châtelet. Operational multiple-doppler wind retrieval inferred from long-range radial velocity measurements. *Journal of applied meteorology and climatology*, 47(11):2929–2945, 2008.
- [38] M. Bouzerdoun, A. Mellit, and A. Massi Pavan. A hybrid model (sarima–svm) for short-term power forecasting of a small-scale grid-connected photovoltaic plant. *Solar Energy*, 98:226 – 235, 2013.
- [39] Timothy H Boyer. Thermodynamics of the harmonic oscillator: Wien’s displacement law and the planck spectrum. *American Journal of Physics*, 71(9):866–870, 2003.
- [40] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, May 2000.
- [41] Horst Bunke and Terry Michael Caelli. *Hidden Markov models: applications in computer vision*, volume 45. World Scientific, 2001.
- [42] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

References

- [43] M. Caldas and R. Alonso-Suárez. Very short-term solar irradiance forecast using all-sky imaging and real-time irradiance measurements. *Renewable Energy*, 143:1643 – 1658, 2019.
- [44] D’Arcy Carlson, Stacy ann Robinson, Catherine Blair, and Marjorie McDonough. China’s climate ambition: Revisiting its first nationally determined contribution and centering a just transition to clean energy. *Energy Policy*, 155:112350, 2021.
- [45] California Energy Commission (CEC). Clean energy and pollution reduction act - sb 350, 2016. <https://www.energy.ca.gov/>.
- [46] M Cervantes, H Krishnaswami, W Richardson, and R Vega. Utilization of low cost, sky-imaging technology for irradiance forecasting of distributed solar generation. In *2016 IEEE Green Technologies Conference (GreenTech)*, pages 142–146. IEEE, 2016.
- [47] Hua Chai, Zhao Zhen, Kangping Li, Fei Wang, Payman Dehghanian, Miadreza Shafie-khah, and João P. S. Catalão. Convolutional auto-encoder based sky image prediction model for minutely solar pv power forecasting. In *2020 IEEE Industry Applications Society Annual Meeting*, pages 1–7, 2020.
- [48] Ming-Ching Chang, Yi Yao, Guan Li, Yan Tong, and Peter Tu. Cloud tracking for solar irradiance prediction. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 4387–4391. IEEE, 2017.
- [49] Robert J. Charlson. 7 - the atmosphere. In Michael C. Jacobson, Robert J. Charlson, Henning Rodhe, and Gordon H. Orians, editors, *Earth System Science*, volume 72 of *International Geophysics*, pages 132 – 158. Academic Press, 2000.
- [50] Eugene Charniak. *Introduction to Deep Learning*. The MIT Press, 2019.
- [51] R. Chauvin, J. Nou, S. Thil, A. Traoré, and S. Grieu. Cloud detection methodology based on a sky-imaging system. *Energy Procedia*, 2015.
- [52] Jia Chen and Chi-Keung Tang. Spatio-temporal markov random field for video denoising. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.
- [53] Xiaoyang Chen, Yang Du, Enggee Lim, Huiqing Wen, Ke Yan, and James Kirtley. Power ramp-rates of utility-scale pv systems under passing clouds: Module-level emulation with cloud shadow modeling. *Applied Energy*, 268:114980, 2020.
- [54] H.-Y. Cheng and C.-L. Lin. Cloud detection in all-sky images via multi-scale neighborhood features and multiple supervised learning techniques. *Atmospheric Measurement Techniques*, 10(1):199–208, 2017.

References

- [55] Hsu-Yung Cheng. Cloud tracking using clusters of feature points for accurate solar irradiance nowcasting. *Renewable Energy*, 104:281 – 289, 2017.
- [56] Jui-Sheng Chou and Ngoc-Son Truong. Cloud forecasting system for monitoring and alerting of energy use by home appliances. *Applied Energy*, 249:166 – 177, 2019.
- [57] Chi Wai Chow, Serge Belongie, and Jan Kleissl. Cloud motion and stability estimation for intra-hour solar forecasting. *Solar Energy*, 115:645–655, 2015.
- [58] Chi Wai Chow, Bryan Urquhart, Matthew Lave, Anthony Dominguez, Jan Kleissl, Janet Shields, and et al. Intra-hour forecasting with a total sky imager at the uc san diego solar energy testbed. *Solar Energy*, 2011.
- [59] Yinghao Chu, Mengying Li, and Carlos F.M. Coimbra. Sun-tracking imaging system for intra-hour dni forecasts. *Renewable Energy*, 96:792 – 799, 2016.
- [60] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [61] Carlos F.M. Coimbra, Jan Kleissl, and Ricardo Marquez. Chapter 8 - overview of solar-forecasting methods and a metric for accuracy evaluation. In Jan Kleissl, editor, *Solar Energy Forecasting and Resource Assessment*, pages 171–194. Academic Press, Boston, 2013.
- [62] Antonio Colmenar-Santos, Ana-Rosa Linares-Mena, Enrique-Luis Molina-Ibáñez, Enrique Rosales-Asensio, and David Borge-Diez. Technical challenges for the optimum penetration of grid-connected photovoltaic systems: Spain as a case study. *Renewable Energy*, 145:2296 – 2305, 2020.
- [63] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.
- [64] Francesco Crespi, Andrea Toscani, Paolo Zani, David Sánchez, and Giampaolo Manzolini. Effect of passing clouds on the dynamic performance of a CSP tower receiver with molten salt heat storage. *Applied Energy*, 229:224–235, 2018.
- [65] J.D. Cryer and K.S. Chan. *Time Series Analysis: With Applications in R*. Springer Texts in Statistics. Springer, 2008.
- [66] Utpal Kumar Das, Kok Soon Tey, Mehdi Seyedmahmoudian, Saad Mekhilef, Moh Yamani Idna Idris, Willem Van Deventer, Bend Horan, and Alex Stojcevski. Forecasting of photovoltaic power generation and model optimization: A review. *Renewable and Sustainable Energy Reviews*, 81:912–928, 2018.

References

- [67] P. Dash, Irani Majumder, Niranjana Nayak, and Ranjeeta Bisoi. Point and interval solar power forecasting using hybrid empirical wavelet transform and robust wavelet kernel ridge regression. *Natural Resources Research*, 29, 02 2020.
- [68] Mathieu David, Faly H Ramahatana Andriamasomanana, and Olivier Liandrat. Spatial and temporal variability of pv output in an insular grid: Case of reunion island. *Energy Procedia*, 57:1275–1282, 2014.
- [69] Soham De and Samuel L Smith. Batch normalization biases residual blocks towards the identity function in deep networks. *arXiv preprint arXiv:2002.10444*, 2020.
- [70] Arnaud de Myttenaere, Boris Golden, Bénédicte Le Grand, and Fabrice Rossi. Mean absolute percentage error for regression models. *Neurocomputing*, 192:38–48, 2016. Advances in artificial neural networks, machine learning and computational intelligence.
- [71] Michael John De Smith, Michael F Goodchild, and Paul Longley. *Geospatial analysis: a comprehensive guide to principles, techniques and software tools*. Troubador publishing ltd, 2007.
- [72] A. de Vos and H. Pauwels. On the thermodynamic limit of photovoltaic energy conversion. *Applied Physics*, 25(2):119–125, June 1981.
- [73] H.M. Deneke, A.J. Feijt, and R.A. Roebeling. Estimating surface solar irradiance from meteosat seviri-derived cloud properties. *Remote Sensing of Environment*, 112(6):3131 – 3141, 2008.
- [74] C. Deng, Z. Li, W. Wang, S. Wang, L. Tang, and A. C. Bovik. Cloud detection in satellite images based on natural scene statistics and gabor features. *IEEE Geoscience and Remote Sensing Letters*, 16(4):608–612, April 2019.
- [75] Ravinesh C. Deo, Xiaohu Wen, and Feng Qi. A wavelet-coupled support vector machine model for forecasting global incident solar radiation using limited meteorological dataset. *Applied Energy*, 168:568–593, 2016.
- [76] S. Dev, Y. H. Lee, and S. Winkler. Color-based segmentation of sky/cloud images from ground-based cameras. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(1):231–242, Jan 2017.
- [77] Soumyabrata Dev, Yee Hui Lee, and Stefan Winkler. Multi-level semantic labeling of sky/cloud images. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 636–640, 2015.

References

- [78] Soumyabrata Dev, Florian Savoy, Yee Hui Lee, and Stefan Winkler. Wahrsis: A low-cost high-resolution whole sky imager with near-infrared capabilities. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 9071, page 90711L, 05 2014.
- [79] Maimouna Diagne, Mathieu David, Philippe Lauret, John Boland, and Nicolas Schmutz. Review of solar irradiance forecasting methods and a proposition for small-scale insular grids. *Renewable and Sustainable Energy Reviews*, 27(Supplement C):65 – 76, 2013.
- [80] Lasanthika H Dissawa, Roshan I Godaliyadda, Parakrama B Ekanayake, Ashish P Agalgaonkar, Duane Robinson, Janaka B Ekanayake, and Sarath Perera. Sky image-based localized, short-term solar irradiance forecasting for multiple pv sites via cloud motion tracking. *International Journal of Photoenergy*, 2021, 2021.
- [81] Sever S Dragomir, Pietro Cerone, and Anthony Sofo. Some remarks on the trapezoid rule in numerical integration. *RGMIA research report collection*, 2(5), 1999.
- [82] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alex J. Smola, and Vladimir Vapnik. Support vector regression machines. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 155–161. MIT Press, 1997.
- [83] Johannes Dröner, Nikolaus Korfhage, Sebastian Egli, Markus Mühling, Boris Thies, Jörg Bendix, Bernd Freisleben, and Bernhard Seeger. Fast cloud segmentation using convolutional neural networks. *Remote Sensing*, 10:1782, 11 2018.
- [84] European Commission (EC). Renewable energy directive 2018/2001/eu, 2018. <https://ec.europa.eu>.
- [85] United States Energy Information Administration (EIA). Annual energy outlook 2021, 2021. www.eia.gov/outlooks/aeo/electricity/sub-topic-03.php.
- [86] Ehab E. Elattar, John Goulermas, and Q. H. Wu. Electric load forecasting based on locally weighted support vector regression. *Trans. Sys. Man Cyber Part C*, 40(4):438–447, 7 2010.
- [87] A. Elia, M. Kamidelivand, F. Rogan, and B. Ó Gallachóir. Impacts of innovation on renewable energy technology cost reductions. *Renewable and Sustainable Energy Reviews*, page 110488, 2020.

References

- [88] Bryan E. Ellis, Nathaniel Pearre, and Lukas Swan. Power ramp rates and variability of individual and aggregate photovoltaic systems using measured production data at the municipal scale. *Solar Energy*, 220:363–370, 2021.
- [89] Boudewijn Elsinga and Wilfried G.J.H.M. van Sark. Short-term peer-to-peer solar forecasting in a network of photovoltaic systems. *Applied Energy*, 206:1464–1483, 2017.
- [90] N.A. Engerer and F.P. Mills. Kpv: A clear-sky index for photovoltaics. *Solar Energy*, 105:679 – 693, 2014.
- [91] H. Escrig, F.J. Batlles, J. Alonso, F.M. Baena, J.L. Bosch, I.B. Salbidegoitia, and J.I. Burgaleta. Cloud detection, classification and motion estimation using geostationary satellite imagery for cloud cover forecast. *Energy*, 55(Supplement C):853 – 859, 2013.
- [92] Abinet Tesfaye Eseye, Jianhua Zhang, and Dehua Zheng. Short-term photovoltaic solar power forecasting using a hybrid wavelet-pso-svm model based on scada and meteorological information. *Renewable Energy*, 118:357 – 367, 2018.
- [93] Manuel Jesús Espinosa-Gavira, Agustín Agüera-Pérez, José-Carlos Palomares-Salas, Juan-José González de-la Rosa, José-María Sierra-Fernández, and Olivia Florencias-Oliveros. Cloud motion estimation from small-scale irradiance sensor networks: General analysis and proposal of a new method. *Solar Energy*, 202:276 – 293, 2020.
- [94] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, June 2008.
- [95] H. Farid and E. P. Simoncelli. Differentiation of discrete multidimensional signals. *IEEE Transactions on Image Processing*, 13(4):496–508, 2004.
- [96] Gunnar Farneback. Two-frame motion estimation based on polynomial expansion. *Image analysis*, pages 363–370, 2003.
- [97] Anita Faul and Michael Tipping. Analysis of sparse bayesian learning. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 46–53. MIT Press, 2002.
- [98] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, June 2006.

References

- [99] Cheng Feng, Yi Wang, Qixin Chen, Yi Ding, Goran Strbac, and Chongqing Kang. Smart grid encounters edge computing: opportunities and applications. *Advances in Applied Energy*, 1:100006, 2021.
- [100] Cong Feng and Jie Zhang. Solarnet: A sky image-based deep convolutional neural network for intra-hour solar forecasting. *Solar Energy*, 204:71–78, 2020.
- [101] Marc Fermigier. The use of images in fluid mechanics. *Comptes Rendus Mécanique*, 345(9):595–604, 2017. A century of fluid mechanics: 1870–1970.
- [102] Gilberto Figueiredo, Marcelo Pinho Almeida, Alex R.A. Manito, and Roberto Zilles. Assessment of an early degraded pv generator. *Solar Energy*, 189:385 – 388, 2019.
- [103] AI Fisenko and SN Ivashov. Determination of the true temperature of emitted radiation bodies from generalized wien’s displacement law. *Journal of Physics D: Applied Physics*, 32(22):2882, 1999.
- [104] Agency for Natural Resources and Energy (ANRE). Feed-in tariff for renewable electricity and solar pv auction, 2017. <http://www.enecho.meti.go.jp>.
- [105] Chia-Lin Fu and Hsu-Yung Cheng. Predicting solar irradiance with all-sky image features via regression. *Solar Energy*, 97:537 – 550, 2013.
- [106] Claudia Furlan, Amauri Pereira de Oliveira, Jacyra Soares, Georgia Codato, and João Francisco Escobedo. The role of clouds in improving the regression model for hourly values of diffuse solar radiation. *Applied Energy*, 92:240 – 254, 2012.
- [107] Yuan Gao, Jianfei Dong, Olindo Isabella, Rudi Santbergen, Hairen Tan, Miro Zeman, and Guoqi Zhang. A photovoltaic window with sun-tracking shading elements towards maximum power generation and non-glare daylighting. *Applied Energy*, 228:1454 – 1472, 2018.
- [108] Óscar García-Hinde, Vanessa Gómez-Verdejo, and Manel Martínez-Ramón. A conditional one-output likelihood formulation for multitask gaussian processes. *arXiv preprint arXiv:2006.03495*, 2021.
- [109] O. García-Hinde, G. Terrén-Serrano, M.A. Hombrados-Herrera, V. Gómez-Verdejo, S. Jiménez-Fernández, C. Casanova-Mateo, J. Sanz-Justo, M. Martínez-Ramón, and S. Salcedo-Sanz. Evaluation of dimensionality reduction methods applied to numerical weather models for solar radiation forecasting. *Engineering Applications of Artificial Intelligence*, 69:157 – 167, 2018.

References

- [110] John F. Geisz, Ryan M. France, Kevin L. Schulte, Myles A. Steiner, Andrew G. Norman, Harvey L. Guthrey, Matthew R. Young, Tao Song, and Thomas Moriarty. Six-junction iii–v solar cells with 47.1% conversion efficiency under 143 suns concentration. *Nature Energy*, 5(4), 4 2020.
- [111] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition, 2004.
- [112] Stuart Geman and Christine Graffigne. Markov random field image models and their applications to computer vision. In *Proceedings of the international congress of mathematicians*, volume 1, page 2. Berkeley, CA, 1986.
- [113] Soumitra K Ghosh, J Paul, and PE Galeski. Criteria for selection of infrared camera system. In *Proceedings of 1994 IEEE Industry Applications Society Annual Meeting*, volume 2, pages 1893–1897. IEEE, 1994.
- [114] Todd S Glickman and Walter Zenk. *Glossary of meteorology*. AMS (American Meteorological Society), 2000.
- [115] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.
- [116] M.I. Gohari, B. Urquhart, H. Yang, B. Kurtz, D. Nguyen, C.W. Chow, M. Ghonima, and J. Kleissl. Comparison of solar power output forecasting performance of the total sky imager and the university of california, san diego sky imager. *Energy Procedia*, 49:2340 – 2350, 2014. Proceedings of the SolarPACES 2013 International Conference.
- [117] Christoph Goller and Andreas Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of International Conference on Neural Networks (ICNN’96)*, volume 1, pages 347–352. IEEE, 1996.
- [118] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
- [119] Siddharth Gopal and Yiming Yang. Von mises-fisher clustering models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 154–162. PMLR, 6 2014.

References

- [120] R.A. Granger. *Fluid Mechanics*. Dover Books on Physics. Dover Publications, 1995.
- [121] Alex Graves. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.
- [122] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.
- [123] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.
- [124] Martin A. Green, Ewan D. Dunlop, Jochen Hohl-Ebinger, Masahiro Yoshita, Nikos Kopidakis, and Xiaojing Hao. Solar cell efficiency tables (version 58). *Progress in Photovoltaics: Research and Applications*, 29(7):657–667, 2021.
- [125] Aurlien Gron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O’Reilly Media, Inc., 2017.
- [126] Mawloud Guermoui, Kacem Gairaa, Arrif Toufik, and John Boland. A novel hybrid model for solar radiation forecasting using support vector machine and bee colony optimization algorithm: Review and case study. *Journal of Solar Energy Engineering*, 143:1–53, 07 2020.
- [127] Arda Halu, Antonio Scala, Abdulaziz Khiyami, and Marta C González. Data-driven modeling of solar-powered urban microgrids. *Science advances*, 2(1), 2016.
- [128] A. Hammer, D. Heinemann, E. Lorenz, and B. Lückehe. Short-term forecasting of solar radiation: a statistical approach using satellite data. *Solar Energy*, 1999.
- [129] Annette Hammer, Jan Kühnert, Kailash Weinreich, and Elke Lorenz. Short-term forecasting of surface solar irradiance based on meteosat-seviri data using a nighttime cloud index. *Remote Sensing*, 7(7):9070–9090, 2015.
- [130] J. M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. Unpublished, 1971.
- [131] Xixuan Han and Line Clemmensen. On weighted support vector regression. *Quality and Reliability Engineering*, 10 2014.

References

- [132] David J. Hand and Veronica Vinciotti. Local versus global models for classification problems: Fitting models where it matters. *The American Statistician*, 57(2):124–131, 2003.
- [133] M. Hasenbalg, P. Kuhn, S. Wilbert, B. Nouri, and A. Kazantzidis. Benchmarking of six cloud segmentation algorithms for ground-based all-sky imagers. *Solar Energy*, 201:596 – 614, 2020.
- [134] Anders Hast. Simple filter design for first and second order derivatives by a double filtering approach. *Pattern Recognition Letters*, 42:65 – 71, 2014.
- [135] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., 2001.
- [136] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Unsupervised learning*. Springer, 2009.
- [137] Takatoshi Hayashi, Tomoya Nagayama, Tadashi Tanaka, and Yoshitaka Inui. Influence of degradation in units of pv modules on electric power output of pv system. *Journal of International Council on Electrical Engineering*, 8(1):119–127, 2018.
- [138] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [139] Thomas Little Heath et al. *The thirteen books of Euclid’s Elements*. Courier Corporation, 1956.
- [140] A. Heinle, A. Macke, and A. Srivastav. Automatic cloud classification of whole sky images. *Atmospheric Measurement Techniques*, 3(3):557–567, 2010.
- [141] S.L. Hess. *Introduction to Theoretical Meteorology*. Holt-Dryden book. Holt, 1959.
- [142] Geoffrey E Hinton, Terrence Joseph Sejnowski, Tomaso A Poggio, et al. *Unsupervised learning: foundations of neural computation*. MIT press, 1999.
- [143] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
- [144] Lion Hirth, Falko Ueckerdt, and Ottmar Edenhofer. Integration costs revisited—an economic framework for wind and solar variability. *Renewable Energy*, 74:925–939, 2015.

References

- [145] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.
- [146] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [147] Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–544, 1952.
- [148] James R. Holton and Gregory J. Hakim. *Chapter 2 - Basic Conservation Laws*. Academic Press, fifth edition, 2013.
- [149] T. Hong, P. Pinson, Y. Wang, R. Weron, D. Yang, and H. Zareipour. Energy forecasting: A review and outlook. *IEEE Open Access Journal of Power and Energy*, 7:376–388, 2020.
- [150] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.
- [151] R.A. Houze. *Cloud Dynamics*. Cloud Dynamics. Academic Press, 1993.
- [152] Robert A Houze Jr. *Cloud dynamics*. Academic press, 2014.
- [153] X. Hu, Y. Wang, and J. Shan. Automatic recognition of cloud images by using visual saliency features. *IEEE Geoscience and Remote Sensing Letters*, 12(8):1760–1764, Aug 2015.
- [154] Chao Huang, Zhenyu Zhao, Long Wang, Zijun Zhang, and Xiong Luo. Point and interval forecasting of solar irradiance with an active gaussian process. *IET Renewable Power Generation*, 14(6):1020–1030, 2020.
- [155] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.
- [156] James P Hughes, Peter Guttorp, and Stephen P Charles. A non-homogeneous hidden markov model for precipitation occurrence. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 48(1):15–30, 1999.
- [157] JR Hummel and WR Kuhn. Comparison of radiative-convective models with constant and pressure-dependent lapse rates. *Tellus*, 33(3):254–261, 1981.

References

- [158] Shin ichi Inage. Development of an advection model for solar forecasting based on ground data first report: Development and verification of a fundamental model. *Solar Energy*, 153:414 – 434, 2017.
- [159] Ken ichi Shimose, Hideaki Ohtake, Joao Gari da Silva Fonseca, Takumi Takashima, Takashi Oozeki, and Yoshinori Yamada. Improvement of the japan meteorological agency meso-scale model for the forecasting the photovoltaic power production: Modification of the cloud scheme. *Energy Procedia*, 57:1346 – 1353, 2014. 2013 ISES Solar World Congress.
- [160] International Energy Agency (IEA). Snapshot of global photovoltaic markets, 2015. <http://www.iea-pvps.org>.
- [161] R. Ineichen, P. and Perez. Derivation of cloud index from geostationary satellites and application to the production of solar irradiance and daylight illuminance data. *Theoretical and Applied Climatology*, 1999.
- [162] Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift fur Physik*, 31(1):253–258, February 1925.
- [163] Tommi S. Jaakkola and Michael I. Jordan. A variational approach to Bayesian logistic regression models and their extensions. In David Madigan and Padhraic Smyth, editors, *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, volume R1 of *Proceedings of Machine Learning Research*, pages 283–294. PMLR, 04–07 Jan 1997.
- [164] H. S. Jang, K. Y. Bae, H. S. Park, and D. K. Sung. Solar power prediction based on satellite images and support vector machine. *IEEE Transactions on Sustainable Energy*, 2016.
- [165] Johan Ludwig William Valdemar Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30:175–193, 1906.
- [166] Yuan Ji, Chunlei Wu, Ping Liu, Jing Wang, and Kevin R Coombes. Applications of beta-mixture models in bioinformatics. *Bioinformatics*, 21(9):2118–2122, 2005.
- [167] He Jiang and Yao Dong. A nonlinear support vector machine model with hard penalty function based on glowworm swarm optimization for forecasting daily global solar radiation. *Energy Conversion and Management*, 126:991–1002, 2016.
- [168] Hou Jiang, Ning Lu, Guanghui Huang, Ling Yao, Jun Qin, and Hengzi Liu. Spatial scale effects on retrieval accuracy of surface solar radiation using satellite data. *Applied Energy*, 270:115178, 2020.

References

- [169] Markku Järvelä, Kari Lappalainen, and Seppo Valkealahti. Characteristics of the cloud enhancement phenomenon and pv power plants. *Solar Energy*, 196:137–145, 2020.
- [170] Ehsanul Kabir, Pawan Kumar, Sandeep Kumar, Adedeji A. Adelodun, and Ki-Hyun Kim. Solar energy: Potential and future prospects. *Renewable and Sustainable Energy Reviews*, 82:894 – 900, 2018.
- [171] Eugenia Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, 2003.
- [172] Jane Oktavia Kamadinata, Tan Lit Ken, and Tohru Suwa. Sky image-based solar irradiance prediction methodologies using artificial neural networks. *Renewable Energy*, 134:837–845, 2019.
- [173] T. Kato, Y. Manabe, T. Funabashi, K. Yoshiura, M. Kurimoto, and Y. Suzuoki. A study on several hours ahead forecasting of spatial average irradiance using nwp model and satellite infrared image. In *2016 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, 2016.
- [174] Zoltan Kato and Ting-Chuen Pong. A markov random field image segmentation model using combined color and texture features. In Władysław Skarbek, editor, *Computer Analysis of Images and Patterns*, pages 547–554. Springer Berlin Heidelberg, 2001.
- [175] Kanyawee Keeratimahat, Anna Bruce, and Iain MacGill. Analysis of short-term operational forecast deviations and controllability of utility-scale photovoltaic plants. *Renewable Energy*, 2020.
- [176] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.
- [177] Mehdi Khashei and Mehdi Bijari. An artificial neural network (p, d, q) model for timeseries forecasting. *Expert Systems with applications*, 37(1):479–489, 2010.
- [178] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
- [179] Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. Online learning with kernels. *IEEE transactions on signal processing*, 52(8):2165–2176, 2004.
- [180] Weicong Kong, Youwei Jia, Zhao Yang Dong, Ke Meng, and Songjian Chai. Hybrid approaches based on deep whole-sky-image learning to photovoltaic generation forecasting. *Applied Energy*, 280:115875, 2020.

References

- [181] Leon Kopitar, Leona Cilar, Primož Kocbek, and Gregor Stiglic. Local vs. global interpretability of machine learning models in type 2 diabetes mellitus screening. In Mar Marcos, Jose M. Juarez, Richard Lenz, Grzegorz J. Nalepa, Slawomir Nowaczyk, Mor Peleg, Jerzy Stefanowski, and Gregor Stiglic, editors, *Artificial Intelligence in Medicine: Knowledge Representation and Transparent and Explainable Systems*, pages 108–119. Springer International Publishing, 2019.
- [182] P. Kuhn, M. Wirtz, N. Killius, S. Wilbert, J.L. Bosch, N. Hanrieder, B. Nouri, J. Kleissl, L. Ramirez, M. Schroedter-Homscheidt, D. Heinemann, A. Kazantzidis, P. Blanc, and R. Pitz-Paal. Benchmarking three low-cost, low-maintenance cloud height measurement systems and ecmwf cloud heights against a ceilometer. *Solar Energy*, 168:140–152, 2018. Advances in Solar Resource Assessment and Forecasting.
- [183] Pascal Kuhn, Stefan Wilbert, Christoph Prah, David Schüler, Thomas Haase, Tobias Hirsch, Michael Wittmann, Lourdes Ramirez, Luis Zarzalejo, Angela Meyer, et al. Shadow camera system for the generation of solar irradiance maps. *Solar Energy*, 157:157–170, 2017.
- [184] Pratima Kumari and Durga Toshniwal. Long short term memory–convolutional neural network based deep hybrid approach for solar irradiance forecasting. *Applied Energy*, 295:117061, 2021.
- [185] Jan Kühnert, Elke Lorenz, and Detlev Heinemann. Chapter 11 - satellite-based irradiance and power forecasting for the german energy market. In Jan Kleissl, editor, *Solar Energy Forecasting and Resource Assessment*, pages 267–297. Academic Press, Boston, 2013.
- [186] D. Lamb and J. Verlinde. *Physics and Chemistry of Clouds*. Cambridge University Press, 2011.
- [187] Kari Lappalainen, Anssi Mäki, and Seppo Valkealahti. Effects of the sharpness of shadows on the mismatch losses of PV generators under partial shading conditions caused by moving clouds. In *Proceedings of 28th European photovoltaic solar energy conference*, pages 4081–4086, 2013.
- [188] Kari Lappalainen and Seppo Valkealahti. Output power variation of different PV array configurations during irradiance transitions caused by moving clouds. *Applied Energy*, 190:902 – 910, 2017.
- [189] Kari Lappalainen, Guang C. Wang, and Jan Kleissl. Estimation of the largest expected photovoltaic power ramp rates. *Applied Energy*, 278:115636, 2020.

References

- [190] Philippe Lauret, Cyril Voyant, Ted Soubdhan, Mathieu David, and Philippe Poggi. A benchmarking of machine learning techniques for solar radiation forecasting in an insular context. *Solar Energy*, 112:446 – 457, 2015.
- [191] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.
- [192] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [193] Peter M Lewis, Howard Rogers, and Rafe H Schindler. A radiometric all-sky infrared camera (rasicam) for des/ctio. In *Ground-based and Airborne Instrumentation for Astronomy III*, volume 7735, page 77353C. International Society for Optics and Photonics, 2010.
- [194] Binghui Li and Jie Zhang. A review on the integration of probabilistic solar forecasting in power systems. *Solar Energy*, 210:68–86, 2020. Special Issue on Grid Integration.
- [195] H. Li, F. Wang, H. Ren, H. Sun, C. Liu, B. Wang, J. Lu, Z. Zhen, and X. Liu. Cloud identification model for sky images based on otsu. In *International Conference on Renewable Power Generation (RPG 2015)*, pages 1–5, Oct 2015.
- [196] Jia Li, Amir Najmi, and Robert M Gray. Image classification by a two-dimensional hidden markov model. *IEEE transactions on signal processing*, 48(2):517–533, 2000.
- [197] Mingquan Li, Edgar Virguez, Rui Shan, Jialin Tian, Shuo Gao, and Dalia Patiño-Echeverri. High-resolution data shows china’s wind and solar energy resources are enough to support a 2050 decarbonized electricity system. *Applied Energy*, 306:117996, 2022.
- [198] Qingyong Li, Weitao Lu, and Jun Yang. A hybrid thresholding algorithm for cloud detection on ground-based color images. *Journal of Atmospheric and Oceanic Technology*, 28(10):1286–1296, 2011.
- [199] Qingyong Li, Weitao Lyu, Jun Yang, and James Wang. Thin cloud detection of all-sky images using markov random fields. *IEEE Geoscience and Remote Sensing Letters*, 9:417–421, 05 2012.
- [200] Stan Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer-Verlag, Berlin, Heidelberg, 2001.

References

- [201] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.
- [202] William Lilley, Jennifer Hayward, and Luke Reedman. Chapter 7 - realizing the potential of renewable and distributed generation. In Fereidoon P. Sioshansi, editor, *Smart Grid*, pages 161–183. Academic Press, Boston, 2012.
- [203] N. Lindsay, Q. Libois, J. Badosa, A. Migan-Dubois, and V. Bourdin. Errors in pv power modelling due to the lack of spectral and angular details of solar irradiance inputs. *Solar Energy*, 197:266 – 278, 2020.
- [204] Lei Liu, Xuejin Sun, Feng Chen, Shijun Zhao, and Taichang Gao. Cloud Classification Based on Structure Features of Infrared Images. *Journal of Atmospheric and Oceanic Technology*, 28(3):410–417, 03 2011.
- [205] S. Liu, L. Zhang, Z. Zhang, C. Wang, and B. Xiao. Automatic cloud detection for all-sky images using superpixel segmentation. *IEEE Geoscience and Remote Sensing Letters*, 12(2):354–358, Feb 2015.
- [206] Rodolfo R Llinás. The contribution of santiago ramón y cajal to functional neuroscience. *Nature Reviews Neuroscience*, 4(1):77–80, 2003.
- [207] CN Long, DW Slater, and Tim P Tooman. *Total sky imager model 880 status and testing results*. Pacific Northwest National Laboratory Richland, Wash, USA, 2001.
- [208] Francis M Lopes, Hugo G Silva, Rui Salgado, Afonso Cavaco, Paulo Canhoto, and Manuel Collares-Pereira. Short-term forecasts of ghi and dni for solar energy systems operation: assessment of the ecmwf integrated forecasting system in southern portugal. *Solar Energy*, 170:14–30, 2018.
- [209] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI’81, page 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.
- [210] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, 2002.
- [211] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. Statistical and machine learning forecasting methods: Concerns and ways forward. *PloS one*, 13(3):e0194889, 2018.

References

- [212] Lakshmi Mallika I, D. Venkata Ratnam, Saravana Raman, and G. Sivavaraprasad. Machine learning algorithm to forecast ionospheric time delays using global navigation satellite system observations. *Acta Astronautica*, 173:221 – 231, 2020.
- [213] A. Mammoli, A. Ellis, A. Menicucci, S. Willard, T. Caudell, and J. Simmins. Low-cost solar micro-forecasts for pv smoothing. In *2013 1st IEEE Conference on Technologies for Sustainability (SusTech)*, pages 238–243, 2013.
- [214] Andrea Mammoli, Guillermo Terrén-Serrano, Anthony Menicucci, Thomas P Caudell, and Manel Martínez-Ramón. An experimental method to merge far-field images from multiple longwave infrared sensors for short-term solar forecasting. *Solar Energy*, 187:254–260, 2019.
- [215] Franco Marchesoni-Acland and Rodrigo Alonso-Suárez. Intra-day solar irradiation forecast using rls filters and satellite images. *Renewable Energy*, 161:1140 – 1154, 2020.
- [216] Ricardo Marquez and Carlos F.M. Coimbra. Intra-hour dni forecasting based on cloud tracking image analysis. *Solar Energy*, 91:327 – 336, 2013.
- [217] M. Martínez-Ramón, A. Gupta, J.L. Rojo-Álvarez, and C. Christodoulou. *Machine Learning Applications in Electromagnetics and Antenna Array Processing*. Artech House, 2021.
- [218] Luis Martín, Luis F. Zarzalejo, Jesús Polo, Ana Navarro, Ruth Marchante, and Marco Cony. Prediction of global solar irradiance based on time series analysis: Application to solar thermal power plants energy production planning. *Solar Energy*, 84(10):1772 – 1781, 2010.
- [219] G.M. Masters and Wiley online library. *Renewable and Efficient Electric Power Systems*. Wiley - IEEE Series. Wiley, 2004.
- [220] D. Mateos, M. Antón, A. Valenzuela, A. Cazorla, F.J. Olmo, and L. Alados-Arboledas. Efficiency of clouds on shortwave radiation using experimental data. *Applied Energy*, 113:1216 – 1219, 2014.
- [221] Patrick Mathiesen, Craig Collier, and Jan Kleissl. A high-resolution, cloud-assimilating numerical weather prediction model for solar irradiance forecasting. *Solar Energy*, 92:47 – 61, 2013.
- [222] Patrick Mathiesen and Jan Kleissl. Evaluation of numerical weather prediction for intra-day solar forecasting in the continental united states. *Solar Energy*, 85(5):967 – 977, 2011.

References

- [223] Nicolás Mazzi and Pierre Pinson. 10 - wind power in electricity markets and the value of forecasting. In George Kariniotakis, editor, *Renewable Energy Forecasting*, Woodhead Publishing Series in Energy, pages 259–278. Woodhead Publishing, 2017.
- [224] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- [225] Geoffrey J McLachlan and Kaye E Basford. *Mixture models: Inference and applications to clustering*, volume 38. M. Dekker New York, 1988.
- [226] Geoffrey J McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2004.
- [227] C.R. Mechoso and A. Arakawa. *NUMERICAL MODELS / General Circulation Models*. Academic Press, second edition, 2015.
- [228] Farid Melgani and Sebastiano B Serpico. A markov random field approach to spatio-temporal contextual image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 41(11):2478–2487, 2003.
- [229] Gabriella Melki, Alberto Cano, Vojislav Kecman, and Sebastián Ventura. Multi-target support vector regression via correlation regressor chains. *Information Sciences*, 415:53–69, 2017.
- [230] A. Mellit, A. Massi Pavan, and V. Lughi. Short-term forecasting of power production in a large-scale photovoltaic plant. *Solar Energy*, 105:401 – 413, 2014.
- [231] Adel Mellit, Alessandro Massi Pavan, Emanuele Ogliari, Sonia Leva, and Vanni Lughi. Advanced methods for photovoltaic output power forecasting: A review. *Applied Sciences*, 10(2), 2020.
- [232] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Proceedings of The Royal Society A: Mathematical, Physical and Engineering Sciences*, 83:69–70, 1909.
- [233] Steven D. Miller, Matthew A. Rogers, John M. Haynes, Manajit Sengupta, and Andrew K. Heidinger. Short-term solar irradiance forecasting via satellite/model coupling. *Solar Energy*, 168:102 – 117, 2018. Advances in Solar Resource Assessment and Forecasting.
- [234] Thomas P Minka. Estimating a gamma distribution. *Microsoft Research, Cambridge, UK, Tech. Rep*, 2002.

References

- [235] Aleksandar M. Mitrašinović. Photovoltaics advancements for transition from renewable to clean energy. *Energy*, 237:121510, 2021.
- [236] Román Mondragón, Joaquín Alonso-Montesinos, David Riveros-Rosas, and Roberto Bonifaz. Determination of cloud motion applying the lucas-kanade method to sky cam imagery. *Remote Sensing*, 12(16), 2020.
- [237] Iyyanki V. Muralikrishna and Valli Manickam. Chapter fourteen - air pollution control technologies. In *Environmental Management*, pages 337 – 397. Butterworth-Heinemann, 2017.
- [238] Akinobu Murata, Hideaki Ohtake, and Takashi Oozeki. Modeling of uncertainty of solar irradiance forecasts on numerical weather predictions with the estimation of multiple confidence intervals. *Renewable Energy*, 117:193 – 201, 2018.
- [239] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.
- [240] John MWallace and Peter V. Hobbs. *3 - Atmospheric Thermodynamics*. Academic Press, second edition, 2006.
- [241] Alexandre K. W. Navarro, Jes Frellsen, and Richard E. Turner. The multivariate generalised von mises distribution: Inference and applications. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI’17, pages 2394–2400. AAAI Press, 2017.
- [242] Angel Navia-Vazquez, Fernando Pérez-Cruz, Antonio Artes-Rodriguez, and Aníbal R Figueiras-Vidal. Weighted least squares training of support vector classifiers leading to compact and adaptive schemes. *IEEE Transactions on Neural Networks*, 12(5):1047–1059, 2001.
- [243] Dung Andu Nguyen and Jan Kleissl. Stereographic methods for cloud base height determination using two sky imagers. *Solar Energy*, 107:495–509, 2014.
- [244] Hien D. Nguyen and Geoffrey J. McLachlan. Laplace mixture of linear experts. *Computational Statistics & Data Analysis*, 93:177 – 191, 2016.
- [245] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, second edition, 2006.
- [246] Gilles Notton, Marie-Laure Nivet, Cyril Voyant, Christophe Paoli, Christophe Darras, Fabrice Motte, and Alexis Fouilloy. Intermittent and stochastic character of renewable energy sources: Consequences, cost of intermittence and benefit of forecasting. *Renewable and Sustainable Energy Reviews*, 87:96–105, 2018.

References

- [247] Bijan Nouri, P Kuhn, Stefan Wilbert, Natalie Hanrieder, C Prah, L Zarzalejo, A Kazantzidis, Philippe Blanc, and R Pitz-Paal. Cloud height and tracking accuracy of three all sky imager systems for individual clouds. *Solar Energy*, 177:213–228, 2019.
- [248] Paul W. Nugent, Joseph A. Shaw, and Sabino Piazzolla. Infrared cloud imaging in support of earth-space optical communication. *Opt. Express*, 17(10):7862–7872, May 2009.
- [249] Jaro Nummikoski. *Sky-image based intra-hour solar forecasting using independent cloud-motion detection and ray-tracing techniques for cloud shadow and irradiance estimation*. M.sc. thesis, University of Texas, San Antonio, 2013.
- [250] National Energy Regulator of South Africa (NERSA). Renewable energy independent power producer programme (reipp), 2011. <http://www.nersa.org.za/>.
- [251] Lanre Olatomiwa, Saad Mekhilef, Shahaboddin Shamshirband, Kasra Mohammadi, Dalibor Petković, and Ch Sudheer. A support vector machine–firefly algorithm-based model for global solar radiation prediction. *Solar Energy*, 115:632–644, 2015.
- [252] Sean Ong, Clinton Campbell, Paul Denholm, Robert Margolis, and Garvin Heath. Land-use requirements for solar power plants in the united states, 2013.
- [253] Lars Onsager. Crystal statistics. i. a two-dimensional model with an order-disorder transition. *Phys. Rev.*, 65:117–149, Feb 1944.
- [254] Kenji Otani, Jyunya Minowa, and Kosuke Kurokawa. Study on areal solar irradiance for analyzing areally-totalized pv systems. *Solar Energy Materials and Solar Cells*, 47(1):281 – 288, 1997.
- [255] Quentin Paletta, Guillaume Arbod, and Joan Lasenby. Benchmarking of deep learning irradiance forecasting models from sky images – an in-depth analysis. *Solar Energy*, 224:855–867, 2021.
- [256] L. L. Pan and L. A. Munchak. Relationship of cloud top to the tropopause and jet structure from calipso data. *Journal of Geophysical Research: Atmospheres*, 116(D12), 2011.
- [257] Mingzhang Pan, Chao Li, Ran Gao, Yuting Huang, Hui You, Tangsheng Gu, and Fengren Qin. Photovoltaic power forecasting based on a support vector machine with improved ant colony optimization. *Journal of Cleaner Production*, 277:123948, 2020.

References

- [258] C. Papin, P. Bouthemy, and G. Rochard. Unsupervised segmentation of low clouds from infrared meteosat images based on a contextual spatio-temporal labeling approach. *IEEE Transactions on Geoscience and Remote Sensing*, 40(1):104–114, Jan 2002.
- [259] Nathan J. Pust Paul W. Nugent, Joseph A. Shaw. Correcting for focal-plane-array temperature dependence in microbolometer infrared cameras lacking thermal stabilization. *Optical Engineering*, 52(6):1 – 8 – 8, 2013.
- [260] Abdul Rahim Pazikadin, Damhuji Rifai, Kharudin Ali, Muhammad Zeesan Malik, Ahmed N. Abdalla, and Moneer A. Faraj. Solar irradiance measurement instrumentation and power solar generation forecasting based on artificial neural networks (ann): A review of five years research trend. *Science of The Total Environment*, 715:136848, 2020.
- [261] Hugo T.C. Pedro and Carlos F.M. Coimbra. Assessment of forecasting techniques for solar power production with no exogenous inputs. *Solar Energy*, 86(7):2017 – 2028, 2012.
- [262] Qiao Peng, Ariya Sangwongwanich, Yongheng Yang, and Frede Blaabjerg. Grid-friendly power control for smart photovoltaic systems. *Solar Energy*, 210:115–127, 2020. Special Issue on Grid Integration.
- [263] Richard Perez, Sergey Kivalov, James Schlemmer, Karl Hemker Jr., David Renné, and Thomas E. Hoff. Validation of short and medium term operational solar radiation forecasts in the us. *Solar Energy*, 2010.
- [264] Richard Perez, Elke Lorenz, Sophie Pelland, Mark Beauharnois, Glenn Van Knowe, Karl Hemker, Detlev Heinemann, Jan Remund, Stefan C. Müller, Wolfgang Traunmüller, Gerald Steinmauer, David Pozo, Jose A. Ruiz-Arias, Vicente Lara-Fanego, Lourdes Ramirez-Santigosa, Martin Gaston-Romero, and Luis M. Pomares. Comparison of numerical weather prediction solar irradiance forecasts in the us, canada and europe. *Solar Energy*, 94:305 – 326, 2013.
- [265] O. Perpiñán and E. Lorenzo. Analysis and synthesis of the variability of irradiance and pv power time series with the wavelet transform. *Solar Energy*, 85(1):188 – 197, 2011.
- [266] Charles Poynton. *Digital Video and HD: Algorithms and Interfaces*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2 edition, 2012.
- [267] Abhnil A. Prasad, Robert A. Taylor, and Merlinde Kay. Assessment of direct normal irradiance and cloud connections using satellite data over Australia. *Applied Energy*, 143:301–311, 2015.

References

- [268] Stefan Preda, Simona-Vasilica Oprea, Adela Bâra, and Anda Belciu (Velicanu). Pv forecasting using support vector machine learning in a big data analytics context. *Symmetry*, 10(12), 2018.
- [269] Chao Qin, Diego Klabjan, and Daniel Russo. Improving the expected improvement algorithm. *arXiv preprint arXiv:1705.10033*, 2017.
- [270] Hao Quan and Dazhi Yang. Probabilistic solar irradiance transposition models. *Renewable and Sustainable Energy Reviews*, 125:109814, 2020.
- [271] Santiago Ramón y Cajal. *Histologie du système nerveux de l’homme & des vertébrés: Cervelet, cerveau moyen, rétine, couche optique, corps strié, écorce cérébrale générale & régionale, grand sympathique*, volume 2. A. Maloine, 1911.
- [272] Mashud Rana, Irena Koprinska, and Vassilios G. Agelidis. Univariate and multivariate methods for very short-term solar photovoltaic power forecasting. *Energy Conversion and Management*, 121:380–390, 2016.
- [273] William J Randel, Fei Wu, and Dian J Gaffen. Interannual variability of the tropical tropopause derived from radiosonde data and ncep reanalyses. *Journal of Geophysical Research: Atmospheres*, 105(D12):15509–15523, 2000.
- [274] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.
- [275] Ibrahim Reda and Afshin Andreas. Solar position algorithm for solar radiation applications. *Solar Energy*, 76(5):577 – 589, 2004.
- [276] Brian J. Redman, Joseph A. Shaw, Paul W. Nugent, R. Trevor Clark, and Sabino Piazzolla. Reflective all-sky thermal infrared cloud imager. *Opt. Express*, 26(9):11276–11283, Apr 2018.
- [277] Hehe Ren, Shujin Laima, Wen-Li Chen, Bo Zhang, Anxin Guo, and Hui Li. Numerical simulation and prediction of spatial wind field under complex terrain. *Journal of Wind Engineering and Industrial Aerodynamics*, 180:49 – 65, 2018.
- [278] Walter Richardson, Hariharan Krishnaswami, Rolando Vega, and Michael Cervantes. A low cost, edge computing, all-sky imager for cloud tracking and intra-hour irradiance forecasting. *Sustainability*, 9(4):482, 2017.
- [279] Marc Ringel, Nils Bruch, and Michèle Knodt. Is clean energy contested? exploring which issues matter to stakeholders in the european green deal. *Energy Research & Social Science*, 77:102083, 2021.

References

- [280] Abbas Rohani, Morteza Taki, and Masoumeh Abdollahpour. A novel soft computing model (gaussian process regression with k-fold cross validation) for daily and monthly solar radiation forecasting (part: I). *Renewable Energy*, 115:411–422, 2018.
- [281] Raúl Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag, Berlin, Heidelberg, 1996.
- [282] Antonello Rosato, Massimo Panella, and Rodolfo Araneo. A distributed algorithm for the cooperative prediction of power production in pv plants. *IEEE Transactions on Energy Conversion*, 34(1):497–508, 2019.
- [283] Niclas Roxhed, Frank Niklaus, Andreas C Fischer, Fredrik Forsberg, Linda Höglund, Per Ericsson, Björn Samel, Stanley Wissmar, Anders Elfving, Tor Ivar Simonsen, et al. Low-cost uncooled microbolometers for thermal imaging. In *Optical sensing and detection*, volume 7726, page 772611. International Society for Optics and Photonics, 2010.
- [284] Masanori Saito and Hironobu Iwabuchi. Cloud discrimination from sky images using a clear-sky index. *Journal of Atmospheric and Oceanic Technology*, 33(8):1583–1595, 2016.
- [285] S. Salcedo-Sanz, C. Casanova-Mateo, A. Pastor-Sánchez, and M. Sánchez-Girón. Daily global solar radiation prediction based on a hybrid coral reefs optimization – extreme learning machine approach. *Solar Energy*, 105:91 – 98, 2014.
- [286] Remember Samu, Martina Calais, G.M. Shafiullah, Moayed Moghbel, Md Asaduzzaman Shueb, Bijan Nouri, and Niklas Blum. Applications for solar irradiance nowcasting in the control of microgrids: A review. *Renewable and Sustainable Energy Reviews*, 147:111187, 2021.
- [287] C. Schillings, H. Mannstein, and R. Meyer. Operational method for deriving high resolution direct normal irradiance from satellite data. *Solar Energy*, 2004.
- [288] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International conference on computational learning theory*, pages 416–426. Springer, 2001.
- [289] Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Comput.*, 12(5):1207–1245, May 2000.
- [290] Gideon Schwarz. Estimating the dimension of a model. *Ann. Statist.*, 6(2):461–464, 03 1978.

References

- [291] Sumen Sen, Rajan Lamichhane, and Norou Diawara. A bivariate distribution with conditional gamma and its multivariate form. *Journal of Modern Applied Statistical Methods*, 13:169–184, 11 2014.
- [292] Peter Shaffery, Aron Habte, Marcos Netto, Afshin Andreas, and Venkat Krishnan. Automated construction of clear-sky dictionary from all-sky imager data. *Solar Energy*, 212:73 – 83, 2020.
- [293] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 1 2016.
- [294] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.
- [295] Joseph A Shaw and Paul W Nugent. Physics principles in radiometric infrared imaging of clouds in the atmosphere. *European Journal of Physics*, 34(6):S111–S121, oct 2013.
- [296] Joseph A. Shaw, Paul W. Nugent, Nathan J. Pust, Brentha Thurairajah, and Kohei Mizutani. Radiometric cloud imaging with an uncooled microbolometer thermal infrared camera. *Opt. Express*, 13(15):5807–5817, Jul 2005.
- [297] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.
- [298] Maxim Vladimirovich Shcherbakov, Adriaan Brebels, Nataliya Lvovna Shcherbakova, Anton Pavlovich Tyukov, Timur Alexandrovich Janovsky, Valeriy Anatol’evich Kamaev, et al. A survey of forecast error measures. *World Applied Sciences Journal*, 24(24):171–176, 2013.
- [299] Chaojun Shi, Yatong Zhou, Bo Qiu, Jingfei He, Mu Ding, and Shiya Wei. Diurnal and nocturnal cloud segmentation of all-sky imager (asi) images using enhancement fully convolutional networks. *Atmospheric Measurement Techniques*, 12:4713–4724, 09 2019.
- [300] Cunzhao Shi, Yu Wang, Chunheng Wang, and Baihua Xiao. Ground-based cloud detection using graph model built upon superpixels. *IEEE Geoscience and Remote Sensing Letters*, 14(5):719–723, 2017.
- [301] Janet E. Shields, Monette E. Karr, Richard W. Johnson, and Art R. Burden. Day/night whole sky imagers for 24-h cloud and sky assessment: history and overview. *Appl. Opt.*, 52(8):1605–1616, Mar 2013.

References

- [302] Alicja Sikora. European green deal—legal and financial challenges of the climate change. In *ERA Forum*, volume 21, pages 681–697. Springer, 2021.
- [303] Harsh Vijay Singh, Roberto Bocca, Pedro Gomez, Steve Dahlke, and Morgan Bazilian. The energy transitions index: An analytic framework for understanding the evolving global energy system. *Energy Strategy Reviews*, 26:100382, 2019.
- [304] Christopher J Smith, Jamie M Bright, and Rolf Crook. Cloud cover effect of clear-sky index distributions and differences between human and automatic cloud observations. *Solar Energy*, 144:10–21, 2017.
- [305] Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, August 2004.
- [306] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25*, pages 2951–2959. Curran Associates, Inc., 2012.
- [307] Sobrina Sobri, Sam Koochi-Kamali, and Nasrudin Abd. Rahim. Solar photovoltaic generation forecasting methods: A review. *Energy Conversion and Management*, 156:459–497, 2018.
- [308] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 2, pages 246–252. IEEE, 1999.
- [309] Bjarne Steffen, Florian Egli, Michael Pahle, and Tobias S. Schmidt. Navigating the clean energy transition in the covid-19 crisis. *Joule*, 4(6):1137–1141, 2020.
- [310] Matthew Stephens. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(4):795–809, 2000.
- [311] Peter H Stone and John H Carlson. Atmospheric lapse rate regimes and their parameterization. *Journal of the Atmospheric Sciences*, 36(3):415–423, 1979.
- [312] Yuchi Sun, Gergely Szűcs, and Adam R Brandt. Solar pv output prediction from video streams using convolutional neural networks. *Energy & Environmental Science*, 11(7):1811–1818, 2018.
- [313] Yuchi Sun, Vignesh Venugopal, and Adam R. Brandt. Short-term solar power forecast with deep learning: Exploring optimal input and output configuration. *Solar Energy*, 188:730–741, 2019.

References

- [314] S Sundararajan and S Sathiya Keerthi. Predictive approaches for choosing hyperparameters in gaussian processes. In *Advances in neural information processing systems*, pages 631–637, 2000.
- [315] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [316] Johan A K Suykens, Tony Van Gestel, Jos De Brabanter, Bart De Moor, and Joos Vandewalle. *Least Squares Support Vector Machines*. WORLD SCIENTIFIC, 2002.
- [317] Alireza Taravat, F. Del Frate, Cristina Cornaro, and Stefania Vergari. Neural networks and support vector machine algorithms for automatic cloud classification of whole-sky ground-based images. *IEEE Geoscience and Remote Sensing Letters*, 12, 02 2015.
- [318] Guillermo Terrén-Serrano. *Machine Learning Approach to Forecast Global Solar Radiation Time Series*. M.sc. thesis, University of New Mexico, 2016.
- [319] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Segmentation algorithms for ground-based infrared cloud images. In *2021 IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pages 1–6, 2021.
- [320] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Wind flow estimation in thermal sky images for sun occlusion prediction. In *2021 IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pages 1–5, 2021.
- [321] Guillermo Terrén-Serrano, Adnan Bashir, Trilce Estrada, and Manel Martínez-Ramón. Girasol, a sky imaging and global solar irradiance dataset. *Data in Brief*, page 106914, 2021.
- [322] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Comparative analysis of methods for cloud segmentation in ground-based infrared images. *Renewable Energy*, 175:1025–1040, 2021.
- [323] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Detection of clouds in multiple wind velocity fields using ground-based infrared sky images. *arXiv preprint arXiv:2105.03535*, 2021.
- [324] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Explicit basis function kernel methods for cloud segmentation in infrared sky images. *Energy Reports*, 7:442–450, 2021. 2021 The 4th International Conference on Electrical Engineering and Green Energy.

References

- [325] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Geospatial perspective reprojections for ground-based sky imaging system. *arXiv preprint arXiv:2103.02066*, 2021.
- [326] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Multi-layer wind velocity field visualization in infrared images of clouds for solar irradiance forecasting. *Applied Energy*, 288:116656, 2021.
- [327] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Processing of global solar irradiance and ground-based infrared sky images for very short-term solar forecasting. *arXiv preprint arXiv:2101.08694*, 2021.
- [328] Guillermo Terrén-Serrano and Manel Martínez-Ramón. Review of kernel learning for intra-hour solar forecasting with infrared sky images and cloud dynamic feature extraction. *arXiv preprint arXiv:2110.05622*, 2021.
- [329] B. Thurairajah and J. A. Shaw. Cloud statistics measured with the infrared cloud imager (ICI). *IEEE Transactions on Geoscience and Remote Sensing*, 43(9):2000–2007, Sep. 2005.
- [330] Andrey N. Tikhonov and Vasilii Y. Arsenin. *Solutions of ill-posed problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York, 1977. Translated from the Russian, Preface by translation editor Fritz John, Scripta Series in Mathematics.
- [331] Michael E. Tipping. The relevance vector machine. In *Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS’99*, page 652–658, Cambridge, MA, USA, 1999. MIT Press.
- [332] Michael E. Tipping. Sparse bayesian learning and the relevance vector machine. *J. Mach. Learn. Res.*, 1:211–244, September 2001.
- [333] Michael E. Tipping, Anita Faul, J J Thomson Avenue, and J J Thomson Avenue. Fast marginal likelihood maximisation for sparse bayesian models. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, pages 3–6, 2003.
- [334] Dan Tulpan, Cajetan Bouchard, Kristopher Ellis, and Cyrus Minwalla. Detection of clouds in sky/cloud and aerial images using moment based texture segmentation. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1124–1133, 2017.
- [335] P. Tzoumanikas, E. Nikitidou, A.F. Bais, and A. Kazantzidis. The effect of clouds on surface solar irradiance, based on data from an all-sky imaging system. *Renewable Energy*, 95:314 – 322, 2016.

References

- [336] Bryan Urquhart, Mohamed Ghonima, Dung (Andu) Nguyen, Ben Kurtz, Chi Wai Chow, and Jan Kleissl. Chapter 9 - sky-imaging systems for short-term forecasting. In Jan Kleissl, editor, *Solar Energy Forecasting and Resource Assessment*, pages 195–232. Academic Press, Boston, 2013.
- [337] Dirk-Jan Van de Ven, Iñigo Capellan-Peréz, Iñaki Arto, Ignacio Cazcarro, Carlos de Castro, Pralit Patel, and Mikel Gonzalez-Eguino. The potential land requirements and related land use change emissions of solar energy. *Scientific reports*, 11(1):1–12, 2021.
- [338] Maria Mercedes Vanegas Cantarero. Of renewable energy, energy democracy, and sustainable development: A roadmap to accelerate the energy transition in developing countries. *Energy Research & Social Science*, 70:101716, 2020.
- [339] Vladimir Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.
- [340] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.
- [341] Manik Varma and Bodla Rakesh Babu. More generality in efficient multiple kernel learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, page 1065–1072, New York, NY, USA, 2009. Association for Computing Machinery.
- [342] Remco A. Verzijlbergh, Petra W. Heijnen, Stephan R. de Roode, Alexander Los, and Harm J.J. Jonker. Improved model output statistics of numerical weather prediction based irradiance forecasts for solar power applications. *Solar Energy*, 118:634 – 645, 2015.
- [343] Cyril Voyant, Gilles Notton, Soteris Kalogirou, Marie-Laure Nivet, Christophe Paoli, Fabrice Motte, and Alexis Fouilloy. Machine learning methods for solar radiation forecasting: A review. *Renewable Energy*, 105(Supplement C):569 – 582, 2017.
- [344] C. Wan, J. Zhao, Y. Song, Z. Xu, J. Lin, and Z. Hu. Photovoltaic and solar power forecasting for smart grid energy management. *CSEE Journal of Power and Energy Systems*, Dec 2015.
- [345] Fei Wang, Zhiming Xuan, Zhao Zhen, Yu Li, Kangping Li, Liqiang Zhao, Miadreza Shafie-khah, and João P.S. Catalão. A minutely solar irradiance forecasting method based on real-time sky image-irradiance mapping model. *Energy Conversion and Management*, 220:113075, 2020.

References

- [346] Guang Wang, Ben Kurtz, and Jan Kleissl. Cloud base height from sky imager and cloud speed sensor. *Solar Energy*, 131:208–221, 2016.
- [347] Guang Chao Wang, Bryan Urquhart, and Jan Kleissl. Cloud base height estimates from sky imagery and a network of pyranometers. *Solar Energy*, 184:594–609, 2019.
- [348] Huaizhi Wang, Haiyan Yi, Jianchun Peng, Guibin Wang, Yitao Liu, Hui Jiang, and Wenxin Liu. Deterministic and probabilistic forecasting of photovoltaic power based on deep convolutional neural network. *Energy Conversion and Management*, 153:409 – 422, 2017.
- [349] Ying Wang, Bo Feng, Qing-Song Hua, and Li Sun. Short-term solar power forecasting: A combined long short-term memory and gaussian process regression method. *Sustainability*, 13(7):3665, Mar 2021.
- [350] Stephen G Warren, Carole J Hahn, and Julius London. Simultaneous occurrence of different cloud types. *Journal of Applied Meteorology and Climatology*, 24(7):658–667, 1985.
- [351] Stephen G Warren, Carole J Hahn, Julius London, Robert M Chervin, and Roy L Jenne. Global distribution of total cloud cover and cloud type amounts over land, 1986.
- [352] Zhi Wei, Hongzhe Li, et al. A hidden spatial-temporal markov random field model for network-based analysis of time course gene expression data. *The Annals of applied statistics*, 2(1):408–429, 2008.
- [353] Chih wei Hsu, Chih chung Chang, and Chih jen Lin. A practical guide to support vector classification, 2010.
- [354] Haoran Wen, Yang Du, Xiaoyang Chen, Enggee Lim, Huiqing Wen, Lin Jiang, and Wei Xiang. Deep learning based multistep solar forecasting for pv ramp-rate control using sky images. *IEEE Transactions on Industrial Informatics*, 17(2):1397–1406, 2021.
- [355] Samuel R West, Daniel Rowe, Saad Sayeef, and Adam Berry. Short-term irradiance forecasting using skycams: Motivation and development. *Solar Energy*, 110:188–207, 2014.
- [356] Aaron Wilson, Alan Fern, and Prasad Tadepalli. Using trajectory data to improve bayesian optimization for reinforcement learning. *The Journal of Machine Learning Research*, 15(1):253–282, 2014.

References

- [357] James T. Wilson, Frank Hutter, and Marc Peter Deisenroth. Maximizing acquisition functions for bayesian optimization. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 9906–9917. Curran Associates Inc., 2018.
- [358] T. Wizelius. 2.13 - design and implementation of a wind power project. In Ali Sayigh, editor, *Comprehensive Renewable Energy*, pages 391 – 430. Elsevier, Oxford, 2012.
- [359] Tore Wizelius. *Developing wind power projects: theory and practice*. Earthscan, 2007.
- [360] Björn Wolff, Jan Kühnert, Elke Lorenz, Oliver Kramer, and Detlev Heinemann. Comparing support vector regression for pv power forecasting to a physical modeling approach using measurement, numerical weather prediction, and cloud motion data. *Solar Energy*, 135:197 – 208, 2016.
- [361] Zeke Xie, Fengxiang He, Shaopeng Fu, Issei Sato, Dacheng Tao, and Masashi Sugiyama. Artificial neural variability for deep learning: On overfitting, noise memorization, and catastrophic forgetting. *Neural Computation*, pages 1–30, 05 2021.
- [362] Dazhi Yang, Jan Kleissl, Christian A Gueymard, Hugo TC Pedro, and Carlos FM Coimbra. History and trends in solar irradiance and pv power forecasting: A preliminary assessment and review using text mining. *Solar Energy*, 168:60–101, 2018.
- [363] L. Ye, Z. Cao, Y. Xiao, and Z. Yang. Supervised fine-grained cloud detection and recognition in whole-sky images. *IEEE Transactions on Geoscience and Remote Sensing*, 57(10):7972–7985, Oct 2019.
- [364] Liang Ye, Zhi-Guo Cao, and Yang Xiao. Deepcloud: Ground-based cloud image categorization using deep convolutional features. *IEEE Transactions on Geoscience and Remote Sensing*, PP:1–12, 06 2017.
- [365] Liang Ye, Zhiguo Cao, Yang Xiao, and Wei Li. Ground-based cloud image categorization using deep convolutional visual features. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 4808–4812. IEEE, 2015.
- [366] Jun Yin, Annalisa Molini, and Amilcare Porporato. Impacts of solar intermittency on future photovoltaic reliability. *Nature Communications*, 11:4781, 09 2020.
- [367] W. J. Youden. Index for rating diagnostic tests. *Cancer*, 3(1):32–35, 1950.

References

- [368] Lisa Anna Zafoschnig, Sebastian Nold, and Jan Christoph Goldschmidt. The race for lowest costs of electricity production: techno-economic analysis of silicon, perovskite and tandem solar cells. *IEEE Journal of Photovoltaics*, 10(6):1632–1641, 2020.
- [369] Haixiang Zang, Ling Liu, Li Sun, Lilin Cheng, Zhinong Wei, and Guoqiang Sun. Short-term global horizontal irradiance forecasting based on a hybrid cnn-lstm model with spatiotemporal correlations. *Renewable Energy*, 160:26–41, 2020.
- [370] William Zappa, Martin Junginger, and Machteld van den Broek. Is a 100% renewable european power system feasible by 2050? *Applied Energy*, 233-234:1027 – 1050, 2019.
- [371] Alireza Zendehboudi, M.A. Baseer, and R. Saidur. Application of support vector machine models for forecasting solar and wind energy resources: A review. *Journal of Cleaner Production*, 199:272–285, 2018.
- [372] Jianwu Zeng and Wei Qiao. Short-term solar power prediction using a support vector machine. *Renewable Energy*, 52:118–127, 2013.
- [373] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.
- [374] Jinglin Zhang, Pu Liu, Feng Zhang, and Qianqian Song. Cloudnet: Ground-based cloud classification with deep convolutional neural network. *Geophysical Research Letters*, 45(16):8665–8672, 2018.
- [375] Z. Zhang and J.C. Moore. *Mathematical and Physical Fundamentals of Climate Change*. Elsevier Science, 2014.
- [376] Ning Zhao and Fengqi You. Can renewable generation, energy storage and energy efficient technologies enable carbon neutral energy transition? *Applied Energy*, 279:115889, 2020.
- [377] Zhao Zhen, Shuaijie Pang, Fei Wang, Kangping Li, Zhigang Li, Hui Ren, Miadreza Shafie-khah, and João P. S. Catalão. Pattern classification and pso optimal weights based sky images cloud motion speed calculation method for solar pv power forecasting. *IEEE Transactions on Industry Applications*, 55(4):3331–3342, 2019.
- [378] Wen Zhuo, Zhiguo Cao, and Yang Xiao. Cloud Classification of Ground-Based Images Using Texture-Structure Features. *Journal of Atmospheric and Oceanic Technology*, 31(1):79–92, 01 2014.

References

- [379] Martin Šinko, Peter Sýkora, Patrik Kamencay, and Róbert Hudec. Development of a system for collecting and processing sky images and meteorological data used for weather prediction. *Transportation Research Procedia*, 40:1548 – 1554, 2019. 2019 13th International Scientific Conference on Sustainable, Modern and Safe Transport (TRANSCOM).