Université catholique de Louvain
Institute of Statistics, Biostatistics and Actuarial Sciences

Statistical analysis and monitoring of time-series panels, with a particular focus on sunspot counts

Sophie Mathieu

A thesis submitted to the
Université catholique de Louvain
in partial fulfillment of the
requirements for the degree of
Doctor of Sciences

Thesis Committee:

Prof. **Rainer von Sachs**	*Université catholique de Louvain*	Supervisor
Prof. **Christian Ritter**	*Université catholique de Louvain*	Secretary
Prof. **Véronique Delouille**	*Royal Observatory of Belgium*	
Prof. **Laure Lefèvre**	*Royal Observatory of Belgium*	
Prof. **Benoit Frénay**	*Université de Namur (UNamur)*	
Prof. **Catherine Legrand**	*Université catholique de Louvain*	President

November 2021

Remember to look up at the stars and not down at your feet. Try to make sense of what you see and wonder about what makes the universe exist. Be curious. And however difficult life may seem, there is always something you can do and succeed at. It matters that you don't just give up.

Stephen Hawking

Acknowledgements

Arriving at the end of this journey, I would like to express in few words all my gratitude. First of all, I want to thank all members of the jury. I am grateful to Pr. Catherine Legrand, who accepted to be its president and gives interesting ideas for future research during the private defence. I also thank Pr. Benoit Frénay for his valuable insights and nice comments, which contribute to improve the manuscript. I am very grateful to Pr. Christian Ritter, who helps me continuously during these four years and in particular to apply the monitoring method to other data. A huge thanks to Véronique Delouille and Laure Lefèvre, who were always present whenever I had a question. I spent great time with both of you at a few conferences. You, Christian and Rainer help me so much in this thesis, especially for submitting (and reviewing) articles as well as during our scientific discussions. Finally, I would like to give a special thank to Pr. Rainer von Sachs, for his trust, kindness, patience and availability. It is wonderful to have a supervisor with such (human) qualities. I could not have learned so much otherwise, in all aspects of the thesis. So ... Danke schön! I started with a (very) limited statistical background and with your support and those of the other members of the jury, I am finally completing this thesis.

In term of scientific guidance, I would like to thank all persons who contribute to this work. A special thank to Pr. Thierry Dudok de Wit, who participates in the construction of the uncertainty model and gives us a complete review of our first paper. Pr. Frédéric Clette also gives interesting ideas and comments, which were very beneficial for our work. I want to thank Hisashi Hayakawa for his help when we were investigating the causes of deviations in several stations. Alongside with Hisashi-san, I acknowledge all observers who help us study past anomalies in the series. Many thanks to Sabrina for her support with Gitlab. I also gratefully acknowledge the Belgian Federal Science Policy Office (BELSPO) for funding this work through the BRAIN VAL-U-SUN project (BR/165/A3/VAL-U-SUN).

From a personal point of view, I want to thank all my colleagues for the wonderful moments spent together. I first thank my office co-workers Oswaldo, Mickaël (who also nicely shares the template of his thesis) and of course Rebecca. Many thanks to Nancy, Nadja, Maguy and Tatiana for their support with administrative tasks. A special thank to Sophie Malali for her help and patience as she receives many mails by errors since our names are similar! I also thank the SMCS team (in particular Vincent B., Alain, Céline, Nathalie Le., Aurélie, Séverine and Lieven) for the nice consulting projects and your coaching with them as well as for creating a warm working atmosphere. For all valuable time that we shared together such as the game nights, the RSSB meetings, the yoga and many more, I finally thank all PhD and post-doc students at the ISBA, including Anas, Alexandre, Antoine, Benjamin D., Charles-Guy, Chikêola, Emanuel, Ensiyeh, Fanny, Florian, Gilles, Hortense, Hugues, Jhon-Jhon, Joris, Kassu, Manon, Michel, Morine, Nathalie Lu., Peter, Stéfane, Stefka, Vanessa and Vincent P.

Finally, this work could not have been achieved without the presence of my family and friends. A special thank to my mother, for her unconditional support. Victor, you were always present for me during these four years, helping me when I had problems (especially in English x). It is wonderful to have a person like you in my life. Thank you for your support and encouragements. I hope to live many more adventures with you and our little Spoutnik 🐱.

Preface

Applied statistics is at the foundation of data analysis, a discipline which uses different kinds of methods to extract information from data. Hence, its two fundamental ingredients are the data and the techniques that can be used for cleaning, modelling or processing those data in order to find new evidences or support decision-making. This thesis falls in this context and has two main purposes. The first one is analysing, modelling the errors and monitoring the quality of the sunspot counts over time. Those data form the basis of the world reference for long-term solar activity: the International Sunspot Number. This index, despite being widely-used in diverse fields such as astrophysics, space weather or climatology and having gained increasing attention over the years, still lacks a proper statistical modelling. Moreover, its processing is based on a single reference observing station. It is thus exposed to any anomaly that occurs in this reference. Such a deviation arose for instance over the years 1981-2015 in the Observatory of Locarno and required the recalibration of the ISN over this whole period. The second main objective of this work is the construction of general non-parametric methods to monitor panels of time-series data, including but not limited to the sunspot counts. Those panel data are indeed common in many applications but often miss a complete and robust treatment procedure.
This thesis is organized as follows.

Chapter 1: Overview
The first chapter gives an introduction to sunspot counts data as well as a description of the main motivations behind this work. The chapter begins by introducing sunspots, their mechanisms of formation and main characteristics, which are often presented in the form of physical laws. After this small introduction more related to solar physics than to statistics, we move on to the description of the data themselves and review how they have been acquired/observed over the years. We also introduce the International Sunspot Number, the world reference for long-term so-

lar activity, which is directly related to the data that will be studied in this work. After summarizing their main features, we present the motivations and the main achievements of the thesis. The sunspot data will serve thus both as motivations and examples to demonstrate the effectiveness of the methods that are developed in the following.

Chapter 2: Uncertainty quantification in sunspot numbers
In the second chapter, we develop a comprehensive error model for the sunspot numbers in a multiplicative framework. The model decomposes the data into a physical signal, common to different observers, corrupted by three types of errors, at short-term, long-term time periods as well as during solar minima. We provide a complete analysis of the different terms of the model, including parametric fits of their distributions. This model allows us to obtain more robust estimators of the sunspot numbers and to provide errors for those data at each point in time. It also highlights the long-term deviations that occurred in the past series of several observing stations. Although specially adapted to the sunspot numbers, the model may also serve as a source of inspiration for treating other datasets with similar properties.

Chapter 3: Non-parametric monitoring of time-series panel data applied to the sunspot numbers
This third chapter describes the construction of a non-parametric monitoring procedure based on control chart and support vector machine to efficiently detect the deviations of sunspot numbers over time. The scheme is designed to work with non-normally distributed and autocorrelated processes with potential missing values. It works at different scales and does not require any parametric assumption about the data to correctly operate. This method allows us to automatically identify many deviations in the sunspot numbers, mostly unseen in previous analyses and helps us to find the root-causes of some prominent shifts. As a result of the present thesis, this control scheme will be implemented to monitor all observing stations involved in the counting process, to prevent the future build-up of large deviations over time, such as those previously observed.

Chapter 4: Neural-networks based monitoring of the sunspot numbers
In this fourth chapter, artificial neural networks are constructed for predicting the sizes and shapes of the deviations occurring in the sunspot numbers. Feed-forward but also recurrent networks, which are better suited to deal with time-series, are designed. They are then coupled with simple cut-off values in the spirit of a Shewhart chart or with an adaptive CUSUM chart to trigger alerts when sufficiently large deviations are detected in the data. Those methods are finally compared with the non-parametric monitoring scheme previously developed. They appear to outperform the previous method for identifying large or oscillating shifts, at the

expense of a greater complexity. Depending on the target, different methods or a combination of them can thus be used to solve applied monitoring problems.

Chapter 5: Application to photovoltaic production data with focus on practical computational aspects
In the fifth chapter, the non-parametric monitoring scheme is applied to the photovoltaic energy production in Belgium. This allows us to present a package, written in the programming language Python, which is associated to the monitoring method. It also solves a second applied problem of practical importance and allows us to start a new collaboration with Elia, the manager of high-voltage electricity in Belgium.

Chapter 6: Automated sunspots detection on white light images
A preliminary version of an automated procedure to extract and count sunspots from white light images is finally proposed in the sixth chapter. Contrarily to other existing algorithms, the method is designed to work on ground-based images, which share more properties with the sunspot observations than images recorded in space. The procedure works correctly on images recorded from year 2011 to year 2020 for identifying the number of spots. More research is needed to improve the method but it shows sufficient potential to be further investigated. As with other automated algorithms, the extracted numbers need to be rescaled to have a similar level as those of the observations. This procedure is thus proposed as an alternative observing method, automated and less prone to observer-related errors. It is not designed to replace totally (and at least not before several solar cycles) manual observations.

Chapter 7: Discussion and Extensions
This last chapter summarizes the achievements and limitations of this work. It also proposes several extensions and research perspectives, which close the thesis.

Contents

Notation and abbreviations

List of common symbols

N_s	Number of individual sunspots
N_g	Number of sunspot groups
N_c	Composite defined as $N_c := 10N_g + N_s$
N	Number of stations in the network
$Y_i(t)$	Observations (N_s, N_g or N_c) at time t in station i
$Z_i(t)$	Rescaled observations (N_s, N_g or N_c)
$\kappa_i(t)$	Piece-wise constant scaling factors
$s(t)$	Solar signal (actual N_s, N_g or N_c on the Sun)
$\hat{\mu}_s(i,t)$	Generic estimator for the mean of $s(t)$, used as a proxy for $s(t)$
M_t	Median of the rescaled observations ($Z_i(t)$)
$\epsilon_1(i,t)$	Short-term error (< 27 days)
$\epsilon_2(i,t)$	Long-term error (≥ 27 days & < 11 years)
$\hat{\mu}_2(i,t)$	Estimator of the mean of the long-term error (often called 'bias')
$h(i,t)$	Intrinsic level of the stations (≥ 11 years)
$\epsilon_3(i,t)$	Error at solar minima
ARL_0	In-control average run length
ARL_1	Out-of-control average run length
K	Number of nearest neighbours
$\mu_0(t)$	Mean of the $\hat{\mu}_2(i,t)$ of the pool
$\sigma_0^2(t)$	Variance of the $\hat{\mu}_2(i,t)$ of the pool
C^+	Upper CUSUM chart statistics
C^-	Lower CUSUM chart statistics

k	Allowance parameter (CUSUM chart)
δ	Shift size
δ_{tgt}	Target shift size
L	Control limit (CUSUM chart)
$\hat{\epsilon}_{\mu_2}(i,t)$	Standardised bias $((\hat{\mu}_2(i,t) - \hat{\mu}_0(t))/\hat{\sigma}_0(t))$
m	length of the input vector (SVM and NN)
ϵ	Approximation accuracy (SVM)
λ	Trade-off between misclassifications and regularisation (SVM)
δ_{cut}	Cut-off value of the networks

List of common abbreviations

ARL	Average run length
BB	Block bootstrap
BBL	Block (bootstrap) length
CBB	Circular block bootstrap
CCD	Charge-coupled device
CUSUM	Cumulative sum
CUSVM	Name of the monitoring method that is developed in Chapter 3 It comes from the contraction of 'SVM' and 'CUSUM'
DBSCAN	Density-based spatial clustering of application with noise
DSO	Distribution System Operator
EM	Expectation-maximization
GMM	Gaussian mixture model
IC	In-control
ISN	International Sunspot Number
K-NN	K nearest neighbours
MA	Moving average
MABB	Matching block bootstrap
MAPE	Mean absolute percentage error
MBB	Moving block bootstrap
MDI	Michelson Doppler Imager
MLE	Maximum likelihood estimation
MSE	Mean-squared error
MSSIM	Mean structural similarity index measure
NB	Negative binomial
NBB	Non-overlapping block bootstrap
NN	Neural network
NN-ACUSUM	(Feed-forward) neural network associated to the adaptive CUSUM chart
NN-CUT	(Feed-forward) neural network associated to simple cut-off values
OC	Out-of-control
OLS	Ordinary least-squares
PDF	Probability distribution function
RBF	Radial basis function
RNN	Recurrent neural network
RNN-ACUSUM	Recurrent neural network associated to the adaptive CUSUM chart
RNN-CUT	Recurrent neural network associated to simple cut-off values
SBB	Stationary block bootstrap

SC	Solar cycle
SE	Structuring element
SoHO	Solar and Heliospheric Observatory
SOM	Self-organising map
SPC	Statistical process control
SSIM	Structural similarity index measure
STARA	Sunspot Tracking And Recognition Algorithm
SVC	Support vector classifier
SVM	Support vector machine
SVR	Support vector regressor
t-LS	t-Location-Scale
USET	Uccle Solar Equatorial Table
WDC-SILSO	World Data Centre Sunspots Index and Long-term Solar Observations
ZA	Zero-altered

CHAPTER 1

Overview

As the title suggests, this thesis is embedded in applied statistics. This field is at the basis of data analysis and, as such, has numerous applications in almost all existing areas (economics, politics, physics, psychology, informatics, etc.). The two main ingredients of this discipline are the data, acquired via e.g. surveys or experiments, and the methods that are applied to those data for extracting information. Similarly, this thesis follows two equivalent tracks: one related to the data and the other associated to the methods. These tracks, far from being separate, are often entangled. The data serve both as motivations and examples to demonstrate the effectiveness of the methods whereas the latter should be sufficiently robust to adjust to the complex features of the former.
This thesis focusses in particular on one dataset that is related to the sunspot counts. Those data serve as proxy for the long-term solar activity and are therefore used in many fields such as space weather, climatology or astrophysics. They still lack however a proper uncertainty quantification and are subject to many deviations along time. The statistical treatment of those data will thus involve a complete error modelling as well as a quality control of the various observing stations involved in the counting process. The development of such methods will then lead to the construction of a general non-parametric monitoring, adapted to panels of time series with low signal-to-noise ratio. This general method will finally be applied to another dataset to solve a second applied problem. Starting from the sunspot numbers, we will thus progressively develop a general method that can be used to monitor many different data.

This introductory chapter begins by giving a general overview of sunspots and the underlying physics at their origin in Section 1.1. The data and their challenges are

then presented in Section 1.2. The chapter ends in Section 1.3 with a description of the main objectives and achievements of the thesis.

1.1 Solar physics interlude

Sunspots are dark areas on the Sun, observable on white (or visible) light images. They correspond to local zones that are cooler (around 4000-5000 kelvin) than their surroundings (of around 6000 kelvin). They are associated to regions of enhanced magnetic field. Their diameters vary between 1500-3500 km for the smallest spots to more than 60000 km (Thomas and Weiss, 2008). Hence the largest spots are visible with the naked eye and can be ten times larger than the Earth. They appear on the surface of the Sun, where they live between a few minutes to a few weeks before decaying and disappearing. Sunspots have been observed since the Antiquity. They were first reported in ancient China and Korea where they were believed to be related to the emperor's fortune. Regular observations of sunspots started however more recently in the seventeen centuries with the invention of the telescope by Galileo. They continue till the present days in several ground-based observing stations disseminated across the world. As such, the counting of sunspots constitutes one of the "longest-running scientific experiment" (Owens, 2013).

In Section 1.1.1, we briefly explain the main motivations behind the sunspot observations and the reasons why they are so carefully monitored today. Then, we describe the main properties of sunspots in Section 1.1.2. Those will help us later in Section 1.1.3 to understand the physical mechanisms behind the formation, evolution and decay of the spots.

1.1.1 Observations of sunspots

Regular observations of sunspots were initially driven by pure research interest. They led to major breakthroughs in the understanding of the Sun and of the mechanisms that govern the magnetic field of stars in general. Around 1610, Galileo discovered for instance the rotation of the Sun by observing the changes of positions of the sunspots over time. He estimated the rotation period of the Sun at 27 days. Closer observations reported later that the spots take around 25 days to rotate on the Sun at the equator while those at higher latitude take a longer time (around 35 days near the poles), an effect called differential rotation. This phenomenon, as we will see later, is now a key ingredient to understand the magnetic field of the stars, which is at the basis of the formation and evolution of the spots. Afterwards, George Ellery Hale observed in 1908 evidence of the presence of a magnetic field in sunspots. This discovery represents a milestone of twentieth century astronomy

(Choudhuri, 2015) since it was the first time that a magnetic field was observed in another celestial body than Earth. It appears that most of the stars and planets have magnetic fields. This finding opened therefore a new area in the history of astronomy, by offering a new way to study distant objects in space.

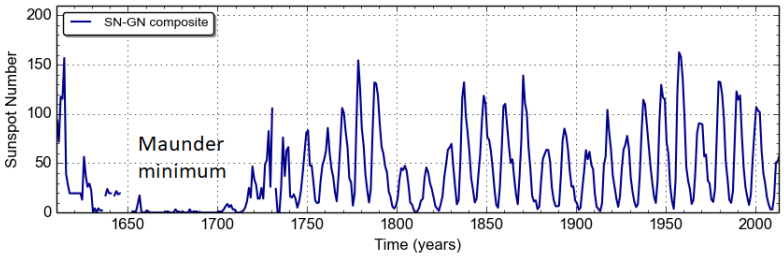


Figure 1.1: Sunspot number (yearly average) represented on the period 1610-2012. Credit to SILSO data/image, Royal Observatory of Belgium, Brussels.

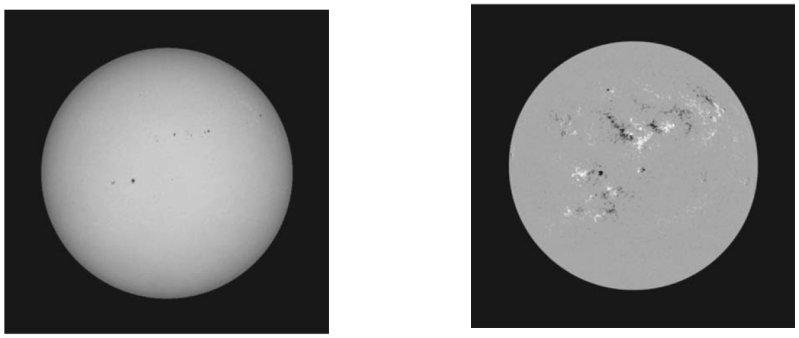
Nowadays, the Sun is monitored continuously and sunspots are studied and observed more than ever. This increase in interest is however more related to the discovery of the first solar flare in 1859 by Richard Carrington than to preceding breakthroughs. A solar flare is an explosive event that is characterized by a sudden increase in solar radiations emission. Flares are the most dramatic events that occur in our solar system. They can last for several minutes to several hours and are characterised by the intense emission of radiations in the entire electromagnetic spectrum. They are also often associated to coronal mass ejections (CME), a phenomenon where a significant amount of charged particles are accelerated and ejected from the solar atmosphere. They propagate then into space and become part of the solar wind. When those charged particles reach the Earth magnetic field, they interact with it and may cause geomagnetic storms, i.e. temporary variations in the Earth magnetic field. They produce beautiful aurorae that are harmless but can also badly damage our electronic infrastructure by inducing huge currents in the power grid.
The solar eruptions, which include flares and CMEs, can thus have drastic impacts on Earth. The CME associated to the flare detected in 1859 by Carrington produced for instance significant damage to the telegraph system, the most advanced electrical technology of that time. Another important solar eruption appeared later on March 9, 1989 and caused a power blackout of several hours on the 13th of March 1989 in Canada, close to the geomagnetic pole of the Earth. Today, large solar eruptions cause disturbances in the upper layers of the atmosphere, which may perturb the radio communication of Earth or badly affect the instruments of planes. They can also damage electronics on boards of satellites that are responsible for diverse tasks as varied as broadcasting television or providing signals to the global positioning system (GPS). Those are protected for this purpose at heavy

cost. Solar eruptions can also be harmful to the astronauts orbiting around the Earth. A major solar eruption directed toward the Earth could still trigger nowadays billions of dollars of damage by destroying the internal structures of most of our electrical equipment. In a society governed by technology, it appears thus more and more important to accurately monitor solar activity.
With the actual understanding of solar physics, the level of accuracy needed for efficient operational predictions of solar flares or CMEs has not yet been reached. We do know solar flares tend to happen mostly above regions of high magnetic fields, which are most often associated to active regions (i.e. sunspots). Those active regions are easily observable since they are more stable than the solar eruptions (their lifetime may extend up to several weeks) and appear in visible light. Therefore, one way to monitor the explosions of the Sun is to closely observe the spots. In 1855, Heinrich Schwabe discovered that the number of sunspots follows a quasi-periodic cycle of mean period equal to 11 years. This cycle, approximately regular, varies however in shape, duration and intensity as can be seen in Figure 1.1. It is a direct manifestation of the solar activity. Hence, the observations of the spots also indicate the periods where the solar eruptions are the most likely to happen: at solar maxima, when the spots are numerous and the solar activity is intense.

If solar eruptions can have a dramatic impact after few minutes or few days (depending on the type of particles that are emitted), changes in solar activity can also influence Earth over long periods. As an example, the period between 1640 and 1720 is known as the Maunder minimum, a time span where solar activity was very low with almost no spots on the surface of the Sun as can be seen in Figure 1.1. Although it is still a matter of study (Owens et al., 2017), this peculiar epoch appears to be linked to the small ice-age that happened in the late seventeen century on Earth. Multiple sources mention that this period was particularly cold, with severe winters in Europe. Therefore, a change in solar activity has also the potential to influence the climate of Earth. One hot research topic is now to understand the mechanisms behind the solar cycles and predict the strength of the next cycles.

1.1.2 Properties of sunspots

Before describing the mechanisms governing the sunspots' life, we review the main properties of the sunspots, which are often expressed in the form of physical laws. These laws appear as constraints that a physical model of the sunspots must explain. They will thus help us later to apprehend the basis of such a model.
The magnetic field of sunspots was first discovered in 1908 by Hale. It is equal to 0.3 tesla inside the largest spots. This field is quite strong for the human scale. As a comparison, the Earth magnetic field is about 3.2×10^{-5} tesla. Hale and its co-workers also found that sunspots appear in pairs of opposite polarity. Figure 1.2



a: White light image b: Magnetogram

Figure 1.2: Image of the Sun taken on 8 February 2001 by the SOHO instrument in white light (left) and its corresponding magnetogram map (right).

shows a magnetogram map of the Sun, i.e. an image that captures the magnetic field at the surface of the Sun at a particular instant. The regions of positive polarity are represented in white whereas the areas of negative polarity are displayed in dark. Both regions are concentrated around the sunspots. The grey background represents the other parts of the Sun, where the magnetic field detected by the instruments is much weaker. By looking closely at the figure, we observe that white regions are located at the right of the dark areas in the northern hemisphere whereas in the southern hemisphere, the white zones are situated at the left of the dark patches. This configuration remains the same for one complete solar cycle but reverses at the next cycle. This effect is named Hale's polarity law and is related to the 22-year magnetic cycle of the Sun. The 11-year solar cycle, which was first unravelled by counting the number of spots, is a direct manifestation of this longer cycle of 22 years. The latter was found afterwards, by observing the magnetic field of the Sun in addition to the spots.
Richard Carrington, who also observed the first solar flare, discovered later another effect linked to the drift of the sunspots. At the beginning of a solar cycle, the spots appear on the Sun at around 30° of latitude, on both sides of the equator. Then, as the time goes, the spots are created at lower and lower latitudes until appearing close to the equator at the end of the cycle. When the next cycle begins, the spots emerge once again close to 30° of latitude. This phenomenon is represented in Figure 1.3, where the latitude of the spots is displayed as a function of the time in what is called a *butterfly* diagram. It is known as Spörer's law (named after Gustav Spörer who studied this effect more systematically than Carrington). A third law was discovered afterwards by Alfred Joy, who observed that the right spot of a pair, which is called the leading spot, appears to be closer to the equator that the left spot, called the following spot. In other words, if we draw a straight

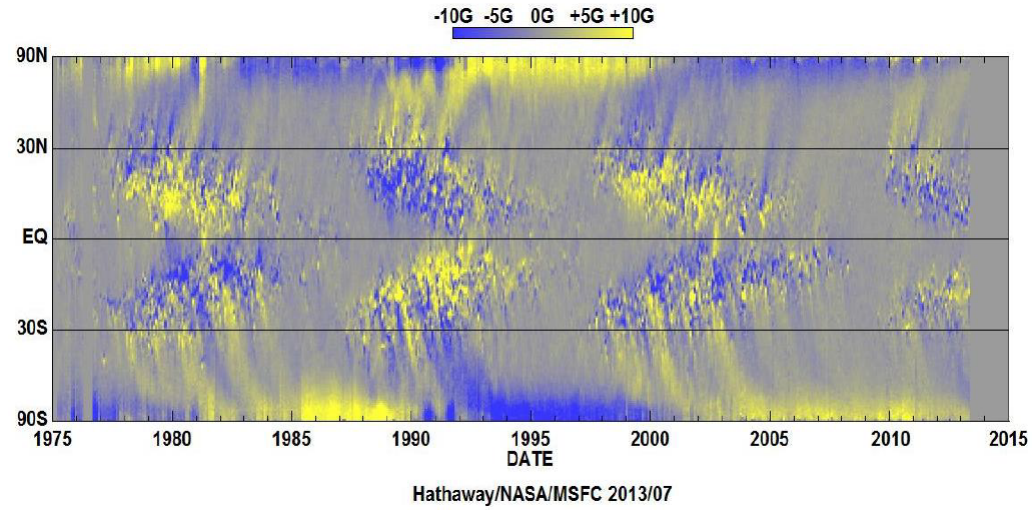


Figure 1.3: The magnetic field of the Sun by latitude represented as a function of the time. This plot is referred to as a butterfly diagram. The figure is taken from http://solarscience.msfc.nasa.gov/images/magbfly.jpg, courtesy of D. Hathaway, NASA/MSFC.

line between the centre of the following spot to those of the leading spot, the line is not perfectly aligned with the equator but is tilted by a small angle. This angle appears to increase with the latitude of the spots, following what we call Joy's law.

As stated before, these laws should be explained by a physical model of the Sun magnetic field and will guide us through the construction of such a model. Knowing the basic properties of sunspots is however already beneficial for understanding their physical nature. A sunspot is created by the emergence of magnetic field lines across the surface of the Sun, as schematically represented in Figure 1.4. The field lines pierce the surface in two different places creating a pair of spots. Where lines emerge from the surface we observe a positive polarity whereas where the lines go back to the solar interior we see a negative polarity. The sunspots emerge thus usually by pairs of opposite polarity, as first observed by Hale and its team. In these areas, the tensions of the intense magnetic field lines prevent the convection of the charged particles that come from the solar interior where the fusion takes place. The heat transport is then decreased in the sunspots with respect to other regions of the Sun where the convection is effective. This effect leads to a drop in temperature inside the spots. Since the luminosity of a region of the Sun depends quadratically on its temperature, the sunspots appear as dark areas on the surface.

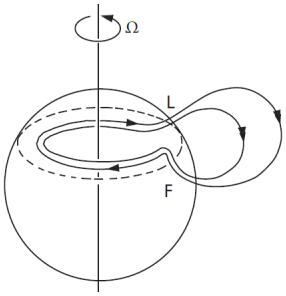


Figure 1.4: Schematic representation of the field lines that form a sunspot. The "L" denotes the leading spot whereas the "F" corresponds to the following spot. The figure is taken from Choudhuri (2015).

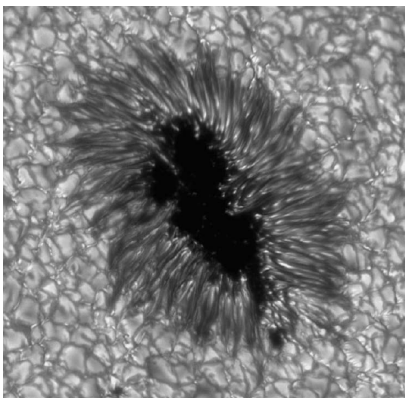


Figure 1.5: Image of a spot with an umbra and penumbra. This image was obtained in the continuum near 436 nm with the Swedish Solar Telescope on La Palma. (Courtesy of L. H. M. Rouppe van der Voort and the Royal Swedish Academy of Sciences.)

The sunspots are themselves composed of two parts: an umbra and a penumbra. Those are represented in Figure 1.5. The umbra is the darkest part of the spot, it corresponds to a region where the magnetic field lines are vertical to the surface. The penumbra is on the contrary lighter, warmer and is associated to field lines that are more inclined. Note that the smallest spots, called pores, are only composed of an umbra.
The emerging field lines can appear as two sunspots of opposite polarities. Most of the time however, the field lines emerge and create two clusters of sunspots of opposite polarities. They are called as a whole a "sunspot group" or "an active region" on the Sun. The groups may thus contain more than two sunspots and have widely different characteristics such as their number of spots, their size, the type of their spots (with or without penumbra), etc. We refer to McIntosh (1990) for illustrating the large variety of sunspot groups. Those are usually more difficult to identify than individual spots.

1.1.3 Introduction to magnetohydrodynamics

The formation, evolution and decay of sunspots are governed by the laws of physics, in particular the electromagnetism and the physics of the plasma. A plasma is the fourth state of matter. When the matter is heated to a sufficiently high temperature in the Sun, the matter becomes ionized, i.e. the electrons are extracted from their atom. The matter is then composed of a soup of electrons and ions, which is called a plasma. If the plasma is electrically neutral, it is strongly sensitive to the influence of any electric or magnetic field (a plasma is a very good conductor). The branch of physics which studies the interaction of a magnetic field inside a plasma is called magnetohydrodynamics, abbreviated by MHD. In this subsection, we briefly sketch some of the main concepts of MHD without entering in too much details and without presenting MHD complex equations.

Oscillating mechanisms

To explain the appearance of spots with opposite polarity in each hemisphere, the magnetic field lines of the Sun should be wrapped around its rotation axis and should have opposite directions in each hemisphere. Therefore the magnetic field of the Sun should have what is called a *toroidal* component. The Sun also has a *poloidal* magnetic field, with opposite polarity at each pole, that is similar to those of Earth. It was first detected in 1955 by Harold and Horace Babcok. Both fields are represented in Figure 1.6. Continuous observations reveal later that the polarity of the poloidal field reverses after a period of eleven years (i.e. the negative pole becomes positive and conversely), forming the 22-year magnetic cycle of the Sun.

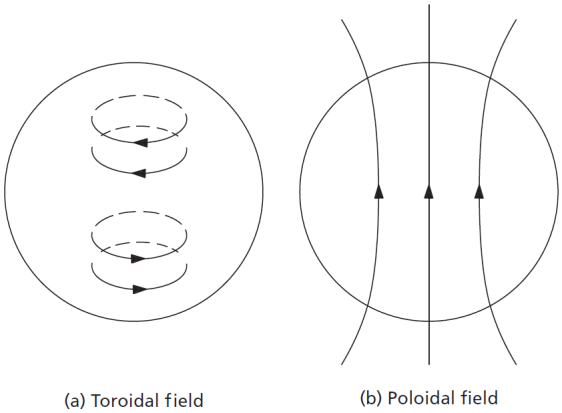


Figure 1.6: Schematic representation of a toroidal and a poloidal field. The figure is taken from Choudhuri (2015).

In 1955, Eugene Parker proposed a ground-breaking explanation for the magnetic cycle of the Sun, which also explains the formation and evolution of the spots. His theory is based on different mechanisms that convert the poloidal field into the toroidal magnetic field of the Sun and vice-versa, creating oscillations between both components of the field (Parker, 1955). These oscillations between the toroidal and poloidal fields can be "observed" a posteriori (these observations were not available for Parker in 1955) in the magnetic configuration of the Sun over time, by looking once again at the butterfly diagram of Figure 1.3. Starting at the solar minima (the period of minimal activity in a 11-year solar cycle) of 1986, we observe that the north pole has a strong negative polarity while the south pole is strongly positive. According to our intuition, the poloidal field should be large when there are few spots on the Sun, since those are rather formed by the toroidal field. The number of spots was indeed close to zero in 1986. As the time goes, the poloidal component weakens while the toroidal field grows, creating more and more spots. When the solar maxima (the period of maximal activity in a 11-year solar cycle) is reached in 1989, the poloidal field is close to zero whereas the toroidal field is maximum, producing many spots. Then the situation slowly reverses in the descending part of the solar cycle. The toroidal field diminishes, creating less and less spots whereas the poloidal field grows again, with opposite polarity with respect to the previous cycle. This continues until the next solar minima in 1997 where the toroidal field becomes once again close to zero. The poloidal field peaks then at a (local) maximum, with a positive value in the north pole and a negative value in the south. The same mechanisms are then repeated in the next cycles.

From a poloidal field toward a toroidal field

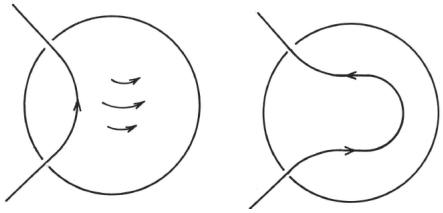


Figure 1.7: Schematic representation of an initial poloidal field (represented on the left panel) transformed into a toroidal field (represented on the right panel) in the Sun by the differential rotation. The figure is taken from Choudhuri (2015).

In Parker's theory, two mechanisms are thus involved. The first one explains how the poloidal field can be transformed into a toroidal component while the other describes how a toroidal field can produce a poloidal field.
The first mechanism in which the poloidal field of the Sun is transformed into a toroidal field is related to the differential rotation of the Sun. As explained previously, all parts of the Sun do not rotate at the same velocity around the rotation axis since the Sun is not solid but is composed of a plasma of particles. The equatorial regions rotate for instance faster than the poles. Complex observations based on helioseismology (the study of the vibrations of the Sun) demonstrated that the rotation velocity also changes within the solar interior. Therefore, as the time goes, the differential rotation progressively stretches and wraps the field lines of the poloidal field around the rotation axis to create a toroidal field, as sketched in Figure 1.7. As can be seen in the figure, the resulting toroidal field has opposite direction in each hemisphere, as expected to reproduce the Hale's polarity law of the sunspots.

The upper layer of the solar interior is the convection zone. It is a region where the heat that comes from the solar interior (where nuclear fusion occurs) is evacuated by convection, i.e. by a transport of matter. After creation by the differential rotation of the Sun, the toroidal magnetic field lines are aggregated in the form of magnetic flux tubes by their interaction with the plasma in the convection zone. These tubes have a balanced pressure with their surrounding (otherwise their cross-section would shrink or expand until such balance is reached). This pressure is only due to the pressure of the plasma outside the tubes while the pressure inside the tubes is due to both the pressure of the plasma and the magnetic pressure of the field. This implies that the internal pressure of the plasma should be lower inside the magnetic flux tubes than outside. This can happen if the density of the plasma is lower inside the tubes. Following Archimede's principle, the tubes that are immersed in a plasma of higher density become buoyant. They rise up and emerge

from the surface, creating active regions (or sunspot groups).

During their ascension, the magnetic flux tubes are also subject to the Coriolis force. This force acts on any object or body which moves on a rotating frame. The Coriolis force has for instance a strong effect on the oceanic and atmospheric circulation on Earth. It makes, amongst many other phenomena, the atmospheric cyclones rotating in opposite sense in each hemisphere. This same force deviates the magnetic flux tubes that are created near the equator — where the differential rotation of the Sun has the highest rate — such that they emerge around 30° of latitude at the beginning of the solar cycle. It also causes the tilt angle of the pairs of spots that are described by Joy's law (D'Silva and Choudhuri, 1993).

From a toroidal field toward a poloidal field

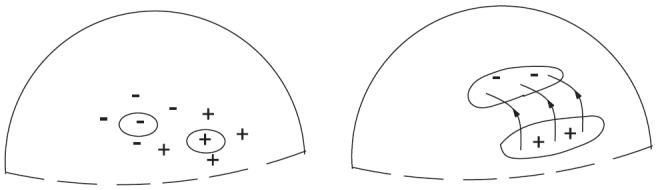
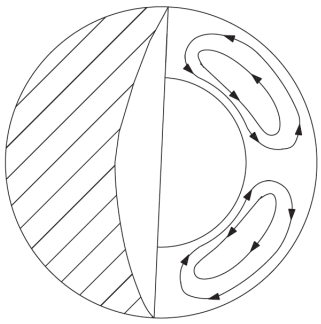


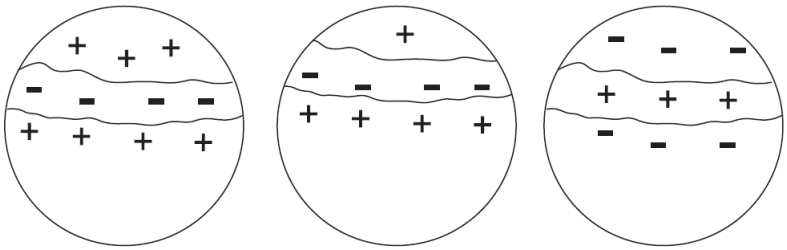
Figure 1.8: Schematic representation of the mechanism transforming an initial toroidal field into a poloidal field in the Sun. An initial pair of sunspots (represented on the left panel) is progressively dispersed by the turbulent motions of convection zone in two larger regions (represented on the right panel). The magnetic field in those regions spreads from its initial configuration in form of tubes and diffuse to its surroundings. The figure is taken from Choudhuri (2015).

To complete our sketch of the dynamo theory, we need to understand how a toroidal field gives rise to a poloidal field. Otherwise, even if the Sun had an initial poloidal magnetic field, it would eventually decay due to the differential rotation of the Sun. Therefore, a second mechanism should be involved to recreate a poloidal field from the toroidal component. This second effect, called Babcock-Leighton mechanism (Babcock, 1961; Leighton, 1969), is linked to the turbulences of the convection zone. When sunspots appear on the surface, they live for a few hours to a few weeks before decaying. The turbulent motions of the convection zone eventually disperse the magnetic field concentrated on the tubes inside the spots into larger regions, making the sunspots progressively disappear. Since a pair of spots is tilted by a small angle (following Joy's law), two regions of opposite polarity are then created, at different latitudes. The resulting field lines, after being dispersed by the turbulent motions, have a poloidal component. This is represented in Figure 1.8, where the leading (right) spot is assumed to have a positive polarity.

If several spots with similar configuration (i.e. a positive polarity at lower latitude and a negative polarity at higher latitude) decay in the same way, large bands of opposite polarity will be progressively created around the equator. Those bands will therefore produce a large-scale poloidal field that is much weaker than the magnetic field of the spots but still measurable from Earth.



a: Meridional circulation



b: Polar magnetic field reversal

Figure 1.9: The upper figure represents the meridional circulation of the Sun. The lower figure shows the transport of the bands of magnetic fields toward the poles by the meridional circulation. When a band of opposite polarity reaches the pole, the polar magnetic field reverses. The figure is taken from Choudhuri (2015).

The turbulent motions in the convection zone also create a large-scale flow of plasma close to the surface of the Sun. This flow is called the meridional circulation. It progressively carries the bands of magnetic field towards the poles like a giant conveyor belt, as can be seen in Figure 1.9a. When a band of opposite polarity reaches the pole, the Sun polar field reverses its polarity. This is illustrated in Figure 1.9b, where the pole has an initial positive polarity. When the band of negative polarity finally reaches the pole, the Sun polar field reverses and gets negative.

A part of the meridional circulation conveys the plasma to the poles. Since the plasma cannot be agglomerated to a small region forever, another flow should transport the plasma back from the pole toward the equator. This second flow was detected by the helioseismology and takes place at the bottom of the convection zone. Therefore, the meriodional circulation transports the plasma from the equator toward the pole at the top of the convection zone, near the surface. The plasma then sinks at the pole and goes back to the equator at the bottom of the convection zone where it emerges again to the surface. This is illustrated in Figure 1.9a. This journey back from the pole to the equator causes a progressive drift of the sunspots. Those first emerge around 30° of latitude at the beginning of the sunspot cycle. Then, they appear progressively at lower latitudes as the toroidal field produced by the differential rotation of the Sun is transported toward the equator by the meridional transport during the cycle. The meridional circulation thus also explains the equatoward drift of the spots described by Spörer's law (Choudhuri et al., 1995).

Complete solar cycle explained

We are now able to understand the solar cycle more deeply. Let us begin once again in the solar minima of 1986. At the beginning of the cycle, the poloidal field is maximal whereas the toroidal field is close to zero. There are few or even no spots on the Sun. As time passes, the poloidal field is progressively transformed into a toroidal component by the differential rotation of the Sun. The toroidal field gets trapped into magnetic tubes that becomes buoyant and emerge at the Surface of the Sun around 30° of latitude, creating pairs of spots. More and more spots are progressively created as the cycle continues. These spots decay due to the turbulent motions of the convection zone (Babcock-Leighton mechanism). They create gradually large bands of opposite polarity around the equator that have a poloidal field. These bands are then slowly transported toward the pole by the meridional circulation until the pole finally reverses its polarity slightly after the solar maxima of 1989, where the toroidal field is maximal and the poloidal field is close to zero. Then, the poloidal field builds up again with the decay of the spots while the toroidal field weakens. Less spots are created. They also emerge closer to the equator due to the meridional transport. The next solar minima is then reached in 1996. At this point, there are almost no spots visible on the Sun. The poloidal field is maximal again whereas the toroidal field is close to zero. The same mechanisms are then repeated over the next cycles.

This completes our small introduction to solar physics. Most of the concepts and images of this section come from the book *Nature's third cycle, a story of sunspots* from A.R. Choudhuri (Choudhuri, 2015). We strongly recommend this book to anyone who is interested in sunspots and solar physics. It is intelligible for every-

body, with or without background in physics, and contains interesting references for further reading.

1.2 Data

After this small interlude in solar physics, we may now introduce the data that will be studied in this work. Those represent the number of spots and the number of sunspot groups. They have been observed and counted since the 17th century in different observing stations around the world. The methods and the challenges associated to the observation of sunspots from Earth over such long time periods are thus first presented in Section 1.2.1. Then, we explain in Section 1.2.2 how these numbers are assembled to form the International Sunspot Number: the world reference index for long-term solar activity. The section ends with a description of the main features of the sunspot number data.

1.2.1 Observing conditions

The sunspots are observed everyday in several ground-based observing stations disseminated around the world. Those are composed of either single observers or teams and can be professional or amateurs. Note that we designate the different stations by the term "observing station" whereas the terminology "observatory" is reserved for the professional ones. To preserve the continuity of the series, the observing methods have not changed much with time. The number of sunspots N_s and sunspot groups N_g are still today manually counted on a daily basis, either by direct observation of the Sun through a telescope or based on a projected image of the Sun taken at a particular instant, by e.g. a charge-coupled device (CCD) camera.
Although the sunspot records date back to a few centuries ago, this work focusses on the most recent part of the series. The period under study extends from the mid 20th century till today. In this time-span, the data are composed of the daily observations of the World Data Centre Sunspots Index and Long-term Solar Observations (WDC-SILSO) network, which contains now around 300 stations. Earlier, this network was much smaller. Past records also contain many more observational gaps that require specific treatments. Hence, the data before the mid 20th are the subject of a dedicated analysis and will not be studied here.

Although the observation of sunspots started years ago, it still remains surprisingly difficult to arrive at an accurate determination of N_s and N_g. Three main difficulties stand out: (a) observability, (b) resolution and (c) interpretation.
(a) For ground-based observations, the Sun cannot be observed when there are

atmospheric perturbations such as clouds. Those depend on the location of the station, which may change with time. The observed series from different stations also cover different time periods. (b) The instruments often have different resolutions, which may give rise to different counts of sunspots. There may also be shifts in the observed numbers due to changes of instruments. Moreover, the counting process can be done using a single digital image of the Sun from a CCD camera or based on the direct observation of the Sun through a telescope. This latter technique has often a smaller resolution since the counting is performed over a small period of time, which is similar to take many images of the Sun into account in the counting process. (c) The sunspots have complex shapes that vary with time and they may overlap. Distinguishing sunspots and groups of sunspots therefore require experience. Even experts sometimes disagree. Moreover, different observers may vary in skill and their skills may vary over time (due to e.g. decrease of eyesight). Some stations observe thus systematically more sunspots than others.
In addition to the observer-related variations, there are also intrinsic sources of variability. Some sunspots are only visible over short periods. Their number may therefore change during the day. Furthermore, sunspot activity itself is subject to substantial variability at multiple time-scales, with the most prominent example being the 11-year solar cycle.

Those effects, either observer or solar related, induce large variabilities in the observed data. This is reflected in Table 1.1, which summarizes the properties of a subset of 21 stations that is used in the first part of this work. The table contains different features of the stations such as their location, name, type (amateur vs professional, individual vs team), observing period, percentage of missing values, and mean scaling-factor (level) with respect to the network over the period 1947-2020. It gives an overview of the various observing conditions of the network.
The mean scaling-factors may be viewed as an indication of the general level of the counts recorded by a station as compared to the median of the network. It may be related to the instrument or the counting method of the stations. level = 1 corresponds to a station that observes the same number of spots than the median of the network. The Observatory of Locarno (LO) with a level of 1.26 observes for instance around 20 % more spots than the others. The location of the stations gives an indication of the weather conditions of the stations. It might thus explain part of the missing values. Moreover, the type of stations usually impacts the quality of observations and the length of observing periods: an individual might experience less short-term variability than a team (alternating the observer from one week to another) and/or amateurs may have shorter observing periods than professionals.

ID	Name	Location	Prof. vs amateur	Team vs individual	Observing period	Level	% Obs.	% Obs. period
A3	Athens Obs.	Athens (Greece)	Prof.	team	1949-1995	1.039	30.16	44.01
BN-S	WFS Obs.*	Berlin (Germany)	Am.	team	1965-2013	1.179	23.50	32.74
CA	Catania Obs.	Catania (Italy)	Prof.	team	1950-	1.039	61.87	64.80
CRA	Cragg†	Australia	Am	indiv.	1947-2009	0.904	72.43	77.44
FU	Fujimori	Nagano (Japan)	Am	indiv.	1968-2019	1.055	45.73	67.32
HD-S	Hedewig*	Germany	Am	indiv.	1967-2013	0.931	25.42	36.96
HU	Public Observatory	Hurbanovo (Slovakia)	Am	team	1969-2019	1.004	35.452	52.80
KH	KOERI	Kandilli (Turkey)	Prof.	team	1967-	0.968	48.81	51.38
KOm	Koyama	Tokyo (Japan)	Am	indiv.	1947-1996	1.052	40.18	54.84
KS2	Kislovodsk Mountain Obs.	Kislovodsk (Russia)	Prof.	$\sim$ indiv.	1954-	1.057	85.96	95.98
KZm	University of Graz	Kanzelhohe (Austria)	Prof.	team	1944-	1.110	74.23	74.24
LFm	Luft	New York (USA)	Am	indiv.	1944-1988	0.985	34.06	54.68
LO	Specola Solare	Locarno (Switzerland)	Prof.	$\sim$ indiv.	1958-	1.260	68.27	81.68
MA	Manila Obs.	Manila (Philippines)	Prof.	team	1971-1988	1.023	20.85	78.69
MO	Mochizuki (Urawa)	Saitama (Japan)	Am	indiv.	1978-2019	1.073	35.51	66.09
PO	Observatory	Postdam (Germany)	Prof.	team	1955-1999	0.991	22.12	29.73
QU	PAGASA weather Bureau	Quezon (Philippines)	Prof.	$\sim$ indiv.	1957-2019	0.829	45.46	53.83
SC-S	Schulze*	Germany	Am	indiv.	1960-2007	0.943	23.32	33.16
SK	Skalnate Pleso Obs.	Vysoke Tatry (Slovakia)	Prof.	team	1950-2012	0.992	37.95	40.75
SM	San Miguel Obs.	Buenos Aires (Argentina)	Prof.	team	1967-2013	1.220	39.09	56.34
UC	USET	Uccle (Belgium)	Prof.	team	1949-	0.991	57.00	59.64

Table 1.1: Main characteristics on the set of 21 stations used in this study: acronym (ID), last name of observer or name of station (Name), location (Location), type of station (professional vs amateur), type of observer (individual or team), observing period (Observing period), averaged scaling-factor with respect to the network over the studied period (Level), percentage of observations on the full period studied (% Obs.) and on their observing period (% Obs. period). Note that $*$ and † symbols represent stations or observers from the SONNE and AAVSO networks respectively. They are two distinct networks of observing stations which are not members of the WDC-SILSO network and hence are not used to produce the ISN.

1.2.2 International Sunspot Number

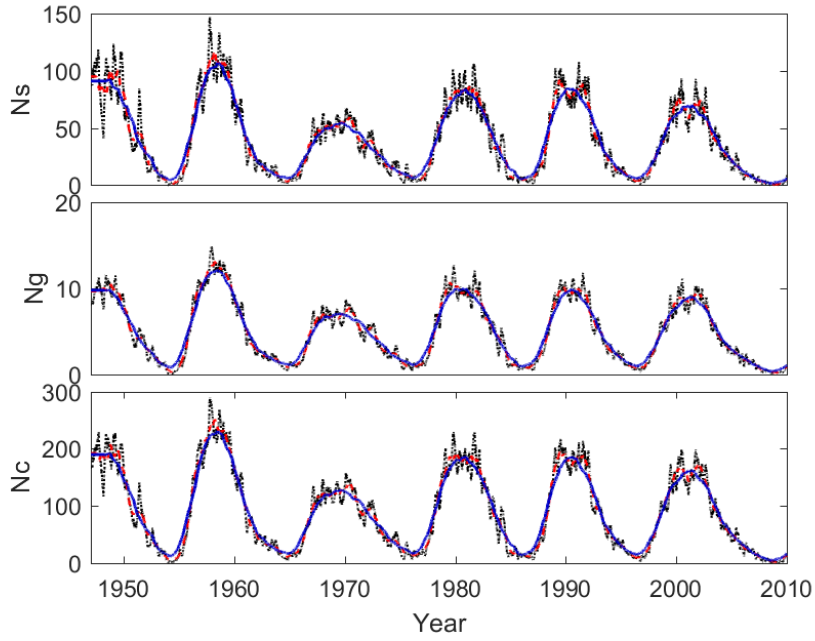


Figure 1.10: Time evolution during 1947-2013 of the median values across 21 observing stations (cf. Table 1.1) for the sunspot counts: (top) N_s, (centre) N_g, and (bottom) N_c. The data are averaged over 81 days (black dotted line), 1 year (red dashed line) and 2.5 years (blue plain line).

In addition to the number of spots and the number of groups, a third quantity will be analysed in the following. This number, denoted N_c, was introduced in 1848 by J.R. Wolf from the Zürich Observatory. It is obtained by summing up the number of sunspots with ten times the number of sunspots groups on a daily basis:

$$N_c = 10N_g + N_s. \tag{1.2.1}$$

N_c contains information about N_s and N_g, as it appears that only one of those quantities cannot fully describe the solar activity. Both numbers are thus required. The multiplication factor in (1.2.1) was introduced by J.R. Wolf to put the number of groups on the same scale as the number of spots. Indeed, during this historical period, a group contained on average ten spots (Izenman, 1985). On the contrary in recent solar cycles, the average number of spots per group is rather around six. Figure 1.10 displays smoothed averages of the median value of N_s, N_g and N_c

across the set of 21 stations presented in Table 1.1.

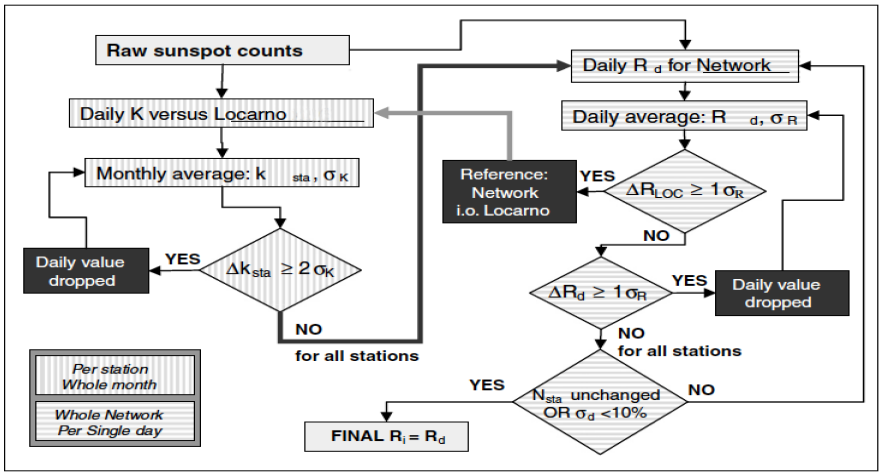


Figure 1.11: Flowchart of the WDC-SILSO data import procedure, illustrating the succession of hierarchical tests applied to raw observing reports (adapted from Clette et al. (2007)).

The Wolf number N_c, also called 'composite', is at the basis of the International Sunspot Number (ISN): the reference index for long-term solar activity. The ISN is distributed through the World Data Centre Sunspots Index and Long-term Solar Observations (WDC-SILSO)[1]. The Table 1 of Dudok de Wit et al. (2016) describes how the processing of this reference index evolved over time. This table shows for instance that between 1926-1981, the index was computed as the average of the composites (N_c) collected by several standard observers from the *same* station. When no observation could be done in this station, the index was then computed using data from auxiliary observatories. In 1981, the sunspot collection centre moved to Belgium where it still resides nowadays. The index was then computed as a weighted average over a network of *several* stations, as it is today. It was named 'International Sunspot Number' to be distinguished from previous versions. This name also emphasizes that the ISN is obtained from a whole network of stations.

Nowadays, N_s and N_g are first entered through an interface[2] and stored in a database. The N_cs from each observing station in the WDC-SILSO network are then computed and rescaled, i.e. multiplied by a factor, to compensate for the

[1] http://www.sidc.be/silso/
[2] www.sidc.be/WOLF

differing observational qualities of the stations. The rescaling uses a *single* pilot station, currently the Observatory of Locarno in Switzerland, as a reference. It compares the values obtained by a station i to the pilot station via a scaling-factor k_i, often referred as the 'k-coefficient':

$$k_i(t) = \frac{pilot(t)}{Y_i(t)}, \tag{1.2.2}$$

where $Y_i(t)$ denotes here the composite index of a station i, observed at time t (expressed in days) and $pilot(t)$ is the value of the pilot station. The monthly scaling-factors are computed from a sigma-clipping mean of (1.2.2), i.e. values differing by more than two standard deviations from the mean are eliminated from the computation process. The N_cs are then combined on a monthly basis to produce the ISN (Clette et al., 2007), which constitutes the world reference for modelling solar activity on the long-term. Its main processing is described in Clette et al. (2007) and we summarize a more recent version of it in Figure 1.11.
The ISN is now one of the most intensely used time-series in astrophysics (Hathaway, 2010). It enters in a large variety of physical models such as those of the Earth climate (Haigh, 2002; Ermolli et al., 2013) as well as in space weather predictions (Temmer et al., 2001; Wang and Colaninno, 2014).

Despite its numerous usages however, the ISN (and its processing) still suffers from its historical heritage. For instance, as an index derived from count data, N_c (or N_s and N_g) does not necessarily follow a Gaussian distribution (Dudok de Wit et al., 2016; Usoskin et al., 2003; Vigouroux and Delache, 1994). A processing based on sigma-clipping is thus not fully adapted to produce the ISN. Due to the many sources of variability that are present in the data, the single pilot station is also not fully appropriate to rescale the stations. This single pilot could for instance experience long-term deviations and cause a drift of the ISN over time. Such a drift appears for instance in the Observatory of Locarno, the actual reference and requires the recalibration of the ISN over 1981-2015 (Clette and Lefèvre, 2016).
Some part of the processing also dates back from the mid-19th century, when J.R. Wolf introduced the sunspot index. They have not been upgraded when the collection and preservation centre was moved from Zurich to Brussels in 1981 since (a) the new curators of the ISN wanted to keep the uniformity of the series and (b) the numerical tools available at that time were limited.

Thanks to recent progress in statistics and data processing, it is now possible to improve the procedures at the basis of the ISN. We can now produce, using modern techniques, an index that is more closely related to the actual solar activity while being at the same time robust against the errors of an individual station. Such a task is complex however and should be done in successive stages. The variability and the statistical nature of N_s, N_g and N_c, the building blocks of the ISN, should

be first studied. Robust estimators for the three numbers may then be constructed. Once such estimators will be constructed, an effective monitoring procedure may then be established to control the stability of these numbers over time. It is only after completing these two steps that the definition of the ISN may be properly upgraded.
The first main objectives of this work are therefore to address these two challenges, with final aim to improve the computation of the ISN.

1.2.3 Features of the data

We focus in the following on the study of N_s, N_g and N_c, which are generically designated by *sunspot numbers* in the following. Since these quantities are observed in different stations, their analysis also opens new perspectives with respect to a single-value index such as the ISN. They allow us to use, among other things, the large variety of longitudinal data analysis tools.
As previously seen, the sunspot numbers are composed of a panel of observations which have complex features. The main characteristics of the data (for N_s, N_g as well as N_c) are summarised below.

Non-Normality
As indexes derived from counts, the data are by nature non-normally distributed. They also experience an excess of zeros due to long periods of solar minima where they are few or even no spots on the Sun. Modelling the statistics of N_c is also far from trivial, since this quantity jumps from zero to eleven when a sunspot appears on the Sun ($N_s = 1, N_g = 1$, and thus $N_c = 11$).

Missing values
As Table 1.1 indicates, the data also present an important amount of missing values. The weather conditions account for most of the short-term missing values while the non-overlapping time periods of the stations are responsible for the largest gaps in the series.

Correlations
The stations are correlated across the panel. They are also correlated along the time since the spots stay from several minutes to several weeks on the Sun surface, depending on their size. The lifetime of the spots is indeed related to the strength of magnetic field lines at their origin.

No reference
Due to the observer and solar related variabilities, all stations are expected to contain various errors and biases. No single station can therefore be considered as a reliable reference for the actual number of spots, groups or composite on the Sun.

Various deviations
The data contain many errors which have various origins and differ substantially in shapes and sizes. The methods developed in this work should therefore be sufficiently robust to adapt to the low signal-to-noise ratio of the data.

1.3 Aims of the thesis

As stated previously, the ISN still suffers from a number of historical errors and inconsistencies. Some of them have been partly addressed by the recalibration of the ISN in 2015 (Clette and Lefèvre, 2016). Even the most recent part (1981-now) of the series lacks however a proper error modelling. The WDC-SILSO team is currently working on improving the ISN computation and coordinates an important community effort to correct past errors. This thesis has been created in this context, to address the specific problems of the most recent part of the series.
The main contributions of this work are therefore two-fold (one-fold for the data and one-fold for the method). Our first contribution is related to the specific dataset that is studied here: the sunspot numbers. Our main objective is to go beyond the aforementioned historical heritage and propose a thorough statistical treatment of the building blocks of the ISN: N_s, N_g and N_c. This includes the development of a comprehensive uncertainty model for the sunspot numbers as well as the construction of a robust monitoring procedure for supervising the quality of these numbers over time. Then, the second main contribution of this work focusses on the statistical methods. It is related to the development of a general non-parametric monitoring procedure, which can be applied to different panels of time-series for quality control.

To this end, we first develop in Chapter 2 a comprehensive uncertainty model for N_s, N_g and N_c. This model is motivated by studies in Dudok de Wit et al. (2016) and works within a *multiplicative* framework. It distinguishes three types of errors and includes robust estimations of the physical solar signal underlying N_s, N_g and N_c. Those can be used as references for the network and may later replace the single pilot station in the processing of the ISN. Although adapted to the sunspot numbers, the model may serve as a source of inspiration for analysing other datasets, as will be seen in Chapter 5.
The study of the long-term error developed in Chapter 2 reveals that some stations experience severe deviations over time. It highlights the need to establish an automated tool for supervising the observations continuously and maintaining the quality of the series over the years. To this end, we construct in Chapter 3 a monitoring procedure for all stations within the WDC-SILSO network. This method consists of three steps: (1) smoothing the data on multiple timescales, (2) monitoring them using block bootstrap calibrated CUSUM charts for detecting various kinds of deviations (such as sudden jumps or progressive drifts) and (3) classifying the out-of-control situations by using support vector techniques.
The monitoring is applied on past data where it allows us to investigate the causes of some prominent deviations that affect the series. Furthermore, the procedure will be soon implemented for the quasi-real time monitoring of the stations. It is designed to alert the observers when they start deviating from the network and

therefore prevent the build-up of large drifts.
In Chapter 4, other control schemes based on artificial neural networks are designed for the same purpose. After comparison with the monitoring procedure previously developed, these alternative methods appear to better perform for identifying large or oscillating deviations. The different methods could therefore be used jointly, to detect more effectively the various kinds of the deviations in the sunspot numbers.

The monitoring procedure that is developed in Chapter 3 is robust and non-parametric. It is also tailored to handle the missing values. This method can thus be applied in a large variety of situations not related to the sunspots. It is adjusted in Chapter 5 to the photovoltaic energy production in Belgium, as one example. This illustrates how the method can be applied to monitor other panels of time-series. The complete method is implemented in the Python programming language and proposed as a general package, available on Github[3]. A description of the package is thus also provided in the chapter.
As additional contributions, we propose in Chapter 6 a preliminary version of a method to automatically extract the number of spots (N_s) and groups (N_g) from ground-based images of the Sun. This method requires less human intervention and is also less variable than manual observations. Contrarily to other existing algorithms, it works on images obtained from the ground, which share many features with the observed numbers. It is designed to be used jointly with the observations and provide numbers corrupted by fewer observer-related errors. More research is needed to obtain a fully-operational method but the proposed algorithm shows sufficient potential to be further investigated.

This thesis is thus organized into seven chapters, including the introduction of Chapter 1 and the conclusion of Chapter 7. The proposed structure is to separate the more developed contributions (Chapters 2, 3, 4 and 5) from methods that are in early development (Chapter 6). The chapters are also presented by increasing degree of automation, which is defined here as all procedures that work to reduce human interventions. We first present the model of the sunspot numbers in Chapter 2. To be fully explored, this model requires many separate analyses (one for each term of the model). Those are hard to automate and take some time, especially the numerous fits of the distributions. Then, we present in Chapter 3 the monitoring method (often called "CUSVM" method), which is more computerized. It is composed of different algorithms which, once implemented, can be called as many times as wanted. A neural network (NN) based monitoring is then presented in Chapter 4 and requires even less input from the user, due to the high flexibility of the networks. It is followed by Chapter 5, presenting the package associated to the CUSVM method. The package takes one step further toward automation, by allowing an easy and compact installation and use of the method. Finally,

[3]The package is available at the following link: https://github.com/sophiano/cusvm.

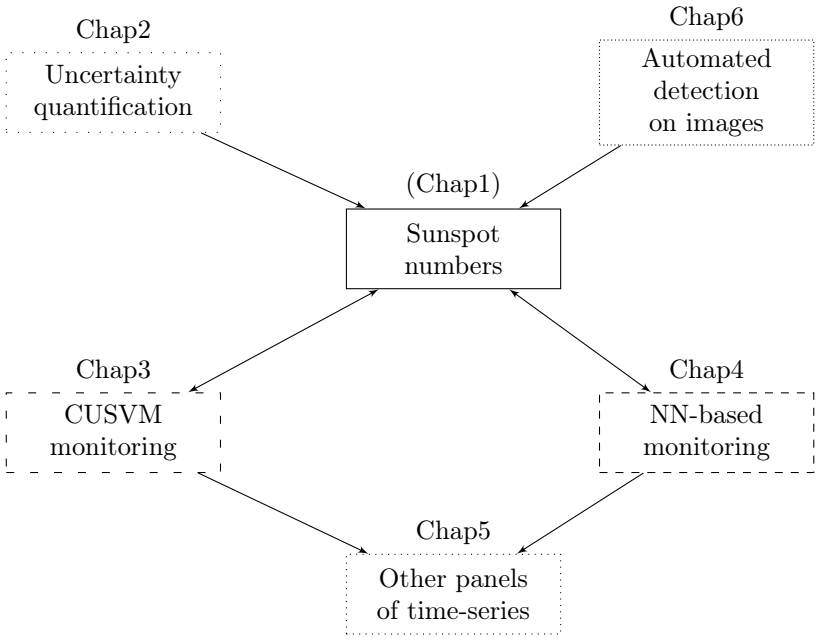


Figure 1.12: Diagram of the thesis.

Chapter 6 proposes a method to automatically retrieve the sunspot numbers from ground-based images of the Sun. It thus extends the automation not only to the monitoring methods but also to the observations themselves.
As stated at the beginning of the chapter, the thesis also follows two tracks, related to the data and methods. In this regard, Chapters 1 and 5 are related to the data whereas Chapters 2, 6 and especially 3 and 4 focus more on the methods. There are thus several ways to structure this thesis, as it is represented in Figure 1.12. By looking at the diagram from top to down, we first see the chapters related to the sunspots and then Chapter 5, associated to another dataset. The chapters which are more focussed on the data are represented in the centre of the figure whereas the others, linked to the methods, are represented on the right and left hand-sides. Finally, the increasing degrees of automation are also displayed in the figure, with a first loosely dotted block for Chapter 2 which is not much automated. It is followed by a loosely dashed block for Chapter 3, a dashed block for Chapter 4 and a dotted block for Chapter 5. Finally, Chapter 6 which extends the automation to the observations themselves is represented in a densely dotted block.

Although there are many links between some chapters, each one can be understood separately except maybe Chapter 5, which is an application of the method

developed in Chapter 3. Chapter 2 is based on a journal paper, Mathieu et al. (2019a). It is also associated to a package (`uncertainty`) with is available for the WDC-SILSO team at the Royal Observatory of Belgium (ROB). It implements the computation of all terms of the model described in the chapter in addition to special requests from the WDC-SILSO team.
Chapter 3 is the object of a second journal paper, Mathieu et al. (2021), which has been tentatively accepted and is currently under revision. Chapter 3 and Chapter 4 are both linked to a second package (`SunSpot`), which is also developed for the WDC-SILSO. It applies the monitoring methods of both chapters to the sunspot numbers, as one example (the scripts and notebooks of the package are thus adapted for the sunspot data). Finally, the method of Chapter 3 is also implemented in a general package (`cusvm`), freely available on Github. This package is described in Chapter 5 and used for monitoring the photovoltaic energy production data (also in the package), which are a bit simpler to study than the sunspots. All packages are written in the Python programming language.

CHAPTER 2

Uncertainty quantification in sunspot numbers

One main characteristic of the scientific methodology is its capacity to evaluate its own errors. Those can be measurement errors, often related to the precision of instruments. Uncertainties may also come from the (incomplete) modelling of a phenomenon ("all models are wrong but some are useful" said the famous statistician Georges Box) or from the estimation of some parameters based on a limited amount of data. Regardless of the types of errors, it is crucial to estimate them to know how reliable our data, model or predictions are. In this context, it is essential to model the uncertainties of the sunspot number data, which act as a benchmark of solar activity in a large range of physical models. An appropriate statistical modelling adapted to the time series nature of those numbers is indeed still lacking.

In this chapter, we provide the first comprehensive uncertainty quantification of sunspot counts. We study three components: the number of sunspots (N_s), the number of sunspot groups (N_g), and the composite N_c, defined as $N_c := N_s + 10N_g$. Those are reported by a network of observing stations around the world and are corrupted by various types of errors. We use a multiplicative framework to provide, for these three components, an estimation of their error distribution in various regimes (short-term, long-term and minima of solar activity). We also propose robust estimators for the underlying solar signal and fit density distributions that take into account intrinsic characteristics such as over-dispersion, excess of zeros, and multiple modes.
Although particular to the sunspot numbers, the reasoning behind the model and the estimators is general and could be applied to other panel data, as will be demonstrated in Chapter 5. The distributions underlying some terms of the model are also typical of count data with over-dispersion and surplus of zeros. Those

are encountered in various fields such as ecology, epidemiology, economy or social sciences.
More specifically, the estimation of the solar signal underlying the composite (N_c) may be seen as a robust version of the International Sunspot Number (ISN), the reference index for solar activity. Our results on N_c may therefore help characterize at the same time the uncertainty on the ISN.
Our results also paves the way for the future monitoring of the stations that will be developed in the next chapter, with the aim to alert the observers when they start deviating from the network and prevent large drifts from occurring.

2.1 Introduction

As explained in Chapter 1, the ISN and its building blocks N_s, N_g and N_c, suffer from errors due to solar- and observer-related variabilities. They also contain some inconsistencies (Clette and Lefèvre, 2016) that originate from their historical definition. An appropriate statistical modelling which takes into account the autocorrelation of the data is however still lacking, even for the most recent part (1981-now) of the series. Our goal in this chapter is therefore to develop a comprehensive uncertainty quantification model for N_s, N_g and N_c, which is adapted to their statistical nature.

2.1.1 Previous works

The study of the errors of the sunspot numbers began years ago with different models mostly based on the Poisson distribution. Long-term analyses started with models of the shape of the sunspot number series (Stewart and Eggleston, 1940; Stewart and Panofsky, 1938). They pursued the works by M. Waldmeier himself (Waldmeier, 1939) who tried to understand the solar cycle and predict upcoming cycles. Later on, Morfill et al. (1991) investigated the short-term dynamical properties of the series using a Poisson noise distribution superimposed on a mean cycle variation. Vigouroux and Delache (1994) also uses a Poisson distribution to approximate the dispersion of daily values of the sunspot numbers at different regimes of solar activity. Usoskin et al. (2003) develops later a reconstruction method for sparse daily values of the data and models the monthly number of groups corresponding to a certain level of daily values by a Poisson distribution.
The need for error bars on the AAVSO sunspot series[1] was already emphasized in Schaefer (1997) and more recent results in Dudok de Wit et al. (2016) present a first uncertainty analysis of the short-term error, through time domain errors

[1] https://www.aavso.org/category/tags/american-relative-sunspot-numbers

and dispersion errors among observing stations, still assuming a Poisson distribution. In Dudok de Wit et al. (2016) however, the authors uncover the presence of over-dispersion in the sunspot numbers and approximate the numbers by a mix of a Poisson and a Gaussian distributions, in an additive framework. Although non-Poissonian, this additive model fails to capture some of the characteristics of sunspot data. Chang and Oh (2012) uses on the contrary a multiplicative model to simulate sunspot counts, in view of assessing the dependency of correction factors on the solar cycle.

Those studies highlight thus the fact that N_s, N_g and N_c are subject to different types of errors and do not behave exactly like Poisson random variables: (1) they experience more dispersion than the Poisson distribution, (2) they are not independent from one day to another and (3) they exhibit a large number of zeros due to periods of minimal solar activity. They also demonstrate the interest of developing an error model in a multiplicative framework to reproduce the main features of the data.

2.1.2 Contributions

Our contribution is therefore two-fold. First, we develop robust estimators for the physical solar signal (denoted 'true' signal in the following) underlying N_s, N_g and N_c. We propose a model for their densities that takes into account characteristics such as over-dispersion and large number of zero counts. Our processing and estimators are also robust to missing values and do not require to fill-in missing observations, contrarily to previous studies.
Second, we propose an uncertainty model that is motivated by first studies in Dudok de Wit et al. (2016) and that works within a *multiplicative* framework. Our model distinguishes three error types. The short-term error accounts for counting errors and variable seeing conditions from one station to another (e.g. weather or atmospheric turbulence), whereas the long-term error provides an overall bias in the number of spots (e.g. gradual ageing of the instrument or observer). Finally, a third type of error aims at modelling inaccuracies occurring at solar minima and helps differentiating true from false zeros. As an illustration, the short-term variations coming from the solar variability and the observational errors are clearly visible in Figure 1.10. They are superimposed on the approximate seasonality of the eleven-year solar cycle.
Our study lays the ground for a future monitoring of all active stations within the WDC-SILSO network in quasi-real time, with the aim to: (1) define a stable reference of the network, (2) alert the observers when they start deviating from the network and (3) prevent large drifts from occurring. Furthermore, the procedures at the basis of the ISN could benefit from the robust estimators and procedures (including the rescaling of the observing stations, see Section 2.4) developed in

the following. The robust estimator for the solar signal underlying N_c could for instance replace the single pilot station in a future definition of the ISN.

This chapter is structured as follows. Section 2.2 introduces the dataset considered. The uncertainty model is presented in Section 2.3 while Section 2.4 details the preprocessing of the data. Section 2.5 provides the estimators (or proxies) for the 'true' solar signal underlying N_s, N_g and N_c as well as their densities. Finally, Section 2.6 displays our results on quantification of the different error types, as well as a first stability analysis that takes into account both short and long-term variability.

2.2 Data

Similarly to what is done in Dudok de Wit et al. (2016), we study a subset of 21 stations, whose main characteristics are listed in Table 1.1 of Chapter 1. The period under study goes from 1947 January 1 till 2013 December 31. It ranges from the maximum of solar cycle (SC) 18 until the ascending phase of SC 24[2] and covers thus almost six solar cycles. This dataset contains the daily number of spots N_s, groups N_g and the composite N_c observed in each of the 21 stations. The methods developed in this chapter can of course be applied on the whole network of stations but these results are not presented here.

2.3 Model

In this section, we present step by step the uncertainty model that we developed. It characterizes the observations of the stations in a multiplicative framework and involves different types of observing errors as well as a quantity generically denoted by $s(t)$, for N_s, N_g and N_c. $s(t)$ is a latent variable representing the 'true' solar signal, i.e. the actual number of spots, groups or composite lying on the Sun. It cannot be directly observed as the counts of the stations are corrupted by different error sources. Our goal is to estimate the distribution of the 'true' solar signal and of the errors degrading it. In particular, we are interested in the mean and the variance of these distributions but also in higher order moments since the estimated densities are far from Gaussian.

The mean of $s(t)$, denoted by $\mu_s(t)$, will be estimated in Section 2.5 based on the entire network, to be robust against errors of an individual station. It will be used

[2] https://en.wikipedia.org/wiki/List_of_solar_cycles

as a proxy for $s(t)$ in the following. Since our model is multiplicative, a good estimation of $\mu_s(t)$ is the key to get access to the multiplicative errors, cf. (2.3.4). A precise estimation of those errors is also required for a future monitoring and this depends on the accuracy of the estimation of $s(t)$.

We use a model that is conditional on the latent $s(t)$ and decomposes the observations along two regimes: when $s(t) = 0$ (solar minima) and when $s(t) > 0$ (outside periods of minima), see Section 2.3.1. This allows introducing, outside of minima, a model with short-term and long-term errors, cf. Section 2.3.2. A specific error model is then developed for periods of solar minima in Section 2.3.3, and the complete model is shown in 2.3.4. Finally, Section 2.3.5 introduces the Hurdle model in order to fit distributions exhibiting an excess of zero values, as it is the case here due to the presence of solar minima and observing errors.

2.3.1 Conditional model

The observed counts are studied in two distinct situations: when there are sunspots ($s > 0$) and when there are none ($s = 0$). This separation is motivated by the idea that the absence or the presence of sunspots are lead by a series of phenomena involving complex dynamo processes in the solar interior, which can be modelled by a latent variable with two states. Note that these two states could be related to activation of the toroidal component of the magnetic field on the Sun (which can be 'on' at solar maxima or 'off' at minima). This analysis will lead to a better understanding of the observations and allows differentiating the 'true' zeros of the counting process from the 'false' zeros that occur when a station reports zero sunspot count in presence of one or more spots on the Sun.

Let $Y_i(t)$ represent either the number of spots, groups or composite actually observed (raw, unprocessed data). The index $1 \leq i \leq N$ denotes the station, and $1 \leq t \leq T$ is the time. The probability density function (PDF) of $Y := Y_i(t)$ may be decomposed as:

$$\begin{aligned}
\mathbf{P}(Y = 0) = \quad & \overbrace{P(Y = 0|s(t) > 0)P(s(t) > 0)}^{1} \\
& + \underbrace{P(Y = 0|s(t) = 0)P(s(t) = 0)}_{2} \\
\mathbf{P}(Y \geq y) = \quad & \overbrace{P(Y \geq y|s(t) > 0)P(s(t) > 0)}^{3} \\
& + \underbrace{P(Y \geq y|s(t) = 0)P(s(t) = 0)}_{4} \text{ for } y > 0.
\end{aligned} \tag{2.3.1}$$

Terms '1' and '3' in (2.3.1) represent the short-term error in presence of one or more sunspots. Term '1' reflects a situation where no sunspots are reported while there are actually some spots on the Sun ('false' zeros or observational errors due e.g. to a bad seeing). It leads to an excess of zeros in short-term error distribution.

Term '2' captures the 'true' zeros (no sunspot and no sunspot reported) while term '4' reflects a situation where the station reports some sunspots when there are no sunspot on the Sun. Term '4' is neglected outside of solar minima periods. Together, these two terms form the distribution of the error at minima, which has an excess of 'true' zeros and a tail modelling the errors of the stations and the short-duration sunspots.

2.3.2 Short-term and long-term errors

Results in Dudok de Wit et al. (2016) evidence a short-term, rapidly evolving, dispersion error across the stations that accounts for counting errors and variable seeing conditions. We define a similar term, allowing a possible station-dependence and we denote it $\epsilon_1(i,t)$. Assuming $\mathbb{E}(\epsilon_1(i,t)) = 0$, where $\mathbb{E}$ is the expectation sign, our interest lies in modelling its variance and its tail to study the short-term variability of the stations.
Next, we introduce $\epsilon_2(i,t)$ to handle station-specific long-term errors such as systematic biases in the sunspot counting process. We want to estimate its mean, $\mu_2(i,t)$, and detect if this mean experiences sudden *jumps* or *drifts* on longer timescales.

Both $\epsilon_1(i,t)$ and $\epsilon_2(i,t)$ are multiplicative errors, as an observer typically makes larger errors when $s(t)$ is higher (Chang and Oh, 2012). Assembling these two types of errors, we propose the following noise model outside of solar minima:

$$Y_i(t) = (\epsilon_1(i,t) + \epsilon_2(i,t))s(t) \text{ when } s(t) > 0. \tag{2.3.2}$$

2.3.3 Errors at solar minima

Let ϵ_3 denote the error occurring during minima of solar activity, when there exist extended periods with no or few sunspots. We assume the error ϵ_3 to be significant when there are no sunspots ($s(t) = 0$) and otherwise negligible in order to not interfere with the errors ϵ_1 and ϵ_2. ϵ_3 captures effects like short-duration sunspots and non-simultaneity of observations between the stations. At solar minima, the model becomes:

$$Y_i(t) = \epsilon_3(i,t) \text{ when } s(t) = 0. \tag{2.3.3}$$

2.3.4 General model

Combining the three error types, we may write our uncertainty model in a compact and generic way as follows:

$$Y_i(t) = \begin{cases} (\epsilon_1(i,t) + \epsilon_2(i,t))s(t) & \text{if } s(t) > 0 \\ \epsilon_3(i,t) & \text{if } s(t) = 0. \end{cases} \tag{2.3.4}$$

We assume the random variables (r.v.) ϵ_1, ϵ_2 and ϵ_3 to be continuous, and the r.v. s, ϵ_1, ϵ_2 and ϵ_3 to be jointly independent. Although the 'true' number of counts $s(t)$ is discontinuous, its product with a continuous r.v. $(\epsilon_1 + \epsilon_2)$ remains continuous. This is consistent with the fact that, after the preprocessing, the observed data $Y_i(t)$ may be modelled by a continuous r.v.

2.3.5 Excess of zeros

All terms in (2.3.4) exhibit an excess of zeros, i.e. an unusual local peak in the density at zero due to solar minima periods. As the solar minimum is an important part of a solar cycle, the zeros must be properly treated. Specific models such as the zero-altered (ZA) or the zero-inflated (ZI) two part distributions may be used for this purpose (Colin Cameron and K. Trivedi, 2013). The main difference between both models is that the ZI distribution allows the zeros to be generated by two different mechanisms contrarily to the ZA model which treats all zeros in the same way.

As 'true' and 'false' zeros do not appear together in a single term of (2.3.4), we find appropriate to work with the ZA two-part model, also called 'Hurdle' model. Those models are commonly encountered in the literature to represent various phenomena such as abundances of species in bounded areas (F. Zuur et al., 2009). In this case, zero counts do not only represent situations where the species is not present in a site even if this site is suited for its habitat. The zeros may also represent design or sampling inconsistencies (such as the counting of a migrating species where the species is abroad or the study of a site which is actually improper for the species) or observer errors (some species are indeed difficult to see or recognize from others). The zeros are thus in excess with respect to what is expected from standard distributions of count data. Hurdle models are also appropriate to model demands for services such as health care (Pohlmeier and Ulrich, 1995) or bookings in home-sharing platforms (Biswas et al., 2020), which can have a large amount of zero values. Moreover, they are used to represent death counts associated to the Covid-19 over different cities or municipalities (Rodriguez-Villamizar et al., 2021). We denote the density of the data by $f(x)$. In the Hurdle model, the zero values are modelled by a Bernoulli distribution $f_0(x) = b^{1-x}(1-b)^x$ of parameter b. Non-zero

values follow a distribution described generically by $f_1(x)$, either another discrete distribution in case of modelling the counts $\mu_s(t)$ in Section 2.5 or a continuous distribution for ϵ_1 and ϵ_3 in Section 2.6:

$$f(x) = \begin{cases} f_0(0) = b & \text{if } x = 0 \\ (1 - f_0(0)) \frac{f_1(x)}{1 - f_1(0)} = (1 - b) \frac{f_1(x)}{1 - f_1(0)} & \text{if } x > 0. \end{cases} \tag{2.3.5}$$

The ZA distribution will be used to model the estimated densities of $\mu_s(t)$ in Section 2.5, and of $\epsilon_1(i,t)$ and $\epsilon_3(i,t)$ in Section 2.6.

2.4 Preprocessing

Due to the different characteristics of the observing conditions (telescope aperture, location, personal experience etc.), each station has its own scaling. These differences mainly impact the count of small spots, which cannot be observed with low-resolution telescopes, and the splitting of complex groups, where the personal experience of the observer matters. A preprocessing is thus needed to rescale all stations to the same level when comparing stations on the short-term and at solar minima. It is also required to compute a robust estimator of the solar signal based on the entire network. For the analysis of long-term errors however, the preprocessing will not be applied, as it would suppress long-term drifts that we want to detect. Our proposed preprocessing is robust to missing values and proceeds in two steps.

First, we compute the 'time-scale', i.e. the duration of the period where the scaling-factors are assumed to be constant. It is a trade-off between short and long periods: the former tends to standardize the observations of the stations, thereby suppressing any differences between the observers. The latter may on the contrary be too coarse to correct for important changes in observers and instruments. A statistical-driven study based on the Kruskal-Wallis test (Kruskal and Wallis, 1952) shows that the appropriate time-scale varies with the stations and with N_s, N_g, and N_c. We refer to Appendix 2.8.1 for a full description of the test. This time-scale may also evolve over time. However, to avoid introducing potential biases between the stations, we use the *same* time-scale, generically denoted by $\tau^\star$, for all stations over the entire period studied. The selected values of $\tau^\star$ are: 8 months for N_s, 14 months for N_g and 10 months for N_c. We note that that these periods are close to the twelve-months period chosen by J.R. Wolf to compute the historical version of the scaling-factors.

Second, having determined the time-scale $\tau^\star$, we compute the scaling-factors using ordinary least-squares regression (OLS) as follows. Recall that $Y_i(t)$ represents either the number of spots, groups or composite actually observed in a station i,

$1 \leq i \leq N$ at time t, $1 \leq t \leq T$ (daily values). For convenience, we re-arrange the time by an array of two indices $t = (t_1, t_2)$, where $1 \leq t_1 \leq \tau^\star$ and $1 \leq t_2 \leq T/\tau^\star$. Thus, t_1 corresponds to the index of an observation inside a block of length $\tau^\star$ and t_2 is the index of the block.

Let $\vec{Y}_{i,t_2} := [Y_i((t_1, t_2))]_{1 \leq t_1 \leq \tau^\star}$ denote the vector of the daily observations in station i on block t_2 of length $\tau^\star$ and $\vec{X}_{i,t_2} := [\underset{1 \leq i \leq N}{\text{med}} Y_i((t_1, t_2))]_{1 \leq t_1 \leq \tau^\star}$ be the vector containing the daily values of the median of the network, also of length $\tau^\star$. The scaling-factors are computed using the slope of the $OLS(\vec{Y}_{i,t_2} | \vec{X}_{i,t_2})$ regression:

$$\kappa_i(t_2) = (\vec{X}_{i,t_2}^T \vec{X}_{i,t_2})^{-1} \vec{X}_{i,t_2}^T \vec{Y}_{i,t_2}. \tag{2.4.1}$$

The new definition of the scaling-factors in (2.4.1) is a robust version of a ratio between the observations of the stations and the median of the network. It is similar to the definition of the historical k in (1.2.2), where the median of the network replaces the single pilot-station as the reference level. The rescaled data, denoted Z_i in the sequel, are defined as:

$$Z_i(t) = \frac{Y_i(t)}{\kappa_i(t)}, \tag{2.4.2}$$

where the reference appears now in the denominator. In a sense, the ratio in (1.2.2) is inverted in order to limit the problem of dividing by zero whenever the stations observe no spots. We explored other methods such as orthogonal regression (also called total least-squares) and $OLS(\vec{X}_{i,t_2} | \vec{Y}_{i,t_2})$ (Feigelson and Babu, 1992). We choose the $OLS(\vec{Y}_{i,t_2} | \vec{X}_{i,t_2})$ method since it leads to the smallest Euclidean and Total variation distances between the median of the network and the individual stations.

2.5 Solar signal estimation

In the following, we first present our estimator for the solar signal, $s(t)$, in a generic framework. Then, we compare it to data obtained from space, which are less variable than the observations (albeit biased). The distribution of the solar signal is then studied separately for N_s, N_g and N_c and fitted by appropriate statistical models. In the last part of the section, the conditional correlation between N_s and N_g is also analysed to better understand some features of those distributions.

2.5.1 Choice of the estimator

To use (2.3.4), we need an estimate of a proxy for $s(t)$. We choose this proxy to be the mean of $s(t)$, denoted $\mu_s(t)$. We propose as a robust estimator for $\mu_s(t)$ a *transformed* version of the median of the network:

$$\hat{\mu}_s(t) = T(M_t), \tag{2.5.1}$$

where $M_t = \underset{1\leq i\leq N}{\text{med}} Z_i(t)$ represents the median of the network, and T denotes a transformation composed of an Anscombe transform and a Wiener filtering (Davenport and Root, 1968). This filtering is applied in order to clean the data from very high frequencies which can lead to instabilities in the subsequent analysis. The generalized Anscombe transform stabilises the variance (Murtagh et al., 1995; Makitalo and Foi, 2013). It writes as

$$A(x) = \frac{2}{\alpha}\sqrt{\alpha x + \frac{3}{8}\alpha^2}\ . \tag{2.5.2}$$

This transformation is commonly applied in the literature to gaussianize near-Poissonnian variables. It is needed here since the Wiener filtering performs better on Gaussian data. Similarly to Dudok de Wit et al. (2016), pp. 14-15, we fix $\alpha = 4.2$ in (2.5.2). This is the optimal value found for the composite N_c. Before applying the Wiener filtering, the missing values in the median of the network are imputed using the algorithm described in Dudok de Wit (2011). Only 49 values are imputed, which represents 0.2% of the total number of values on the period studied. The Wiener filtering is then applied on the transformed and complete set of median values to suppress the highest frequencies of the signal. Finally, the imputed missing values were reset to their initial NaN ('not a number') values in $\hat{\mu}_s(t)$. The block diagram of the procedure is described in Figure 2.1.

Note that among other tested estimators (based on the mean, the median of the network or a subset of stations), the estimator proposed in (2.5.1) turns out to be the most robust to outliers.

2.5.2 Comparison with space data

To test the quality of our estimator $\hat{\mu}_s$, we compare it with a sunspot number extracted from satellite images of the Sun. We expect less variability when N_s, N_g and N_c are retrieved from satellite images using automated algorithms as the rules to count spots and groups are clearly defined. The measurements may be biased by these rules however and the most complex cases, e.g. at maxima, most often require either human intervention or a specific procedure in the algorithm. It is well established fact that a measure of the 'true' number of spots and groups does not exist.

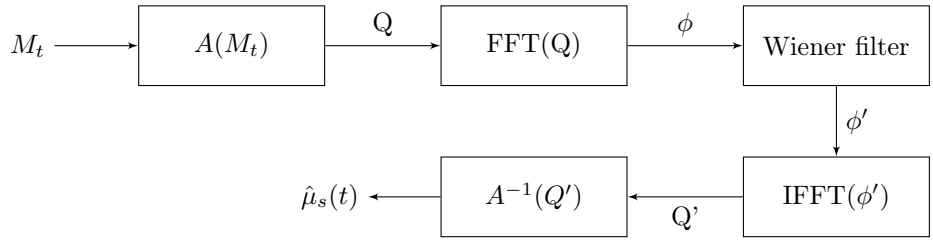


Figure 2.1: Block diagram of the T procedure defining the solar signal estimator $\hat{\mu}_s(t)$, where $M_t = \underset{1 \leq i \leq N}{\text{med}} Z_i(t)$ is the median of the network. An Anscombe transform is first applied on the median, and missing values are imputed. Then, a fast Fourier transform (FFT) is used to convert the signal to a power spectrum in the frequency domain, followed by an attenuation from a Wiener filter. A step function cancels the amplitude of the frequencies corresponding to the periods inferior to seven days (low-pass filter). The threshold at seven days is selected from (Dudok de Wit et al., 2016, Figure 5). It is the smallest visible time-scale of the signal, corresponding to the weekly shift of some observatories. Finally, an inverse FFT and an inverse Anscombe transform are applied to the signal.

As exercise for this comparison, we use the Sunspot Tracking And Recognition Algorithm (STARA) sunspot catalogue (Watson and Fletcher, 2010), regrouping observations from May 1996 to October 2010 (solar cycle 23). This number is extracted using an automated detection algorithm from the images obtained by the MDI instrument on the Solar and Heliospheric Observatory (SoHO), a spatial observatory orbiting around the Sun. The extracted number has a lower scaling than our estimator for the number of spots, $\hat{\mu}_{N_s}$, since the definition of a spot in the detection algorithm excludes the pores (spots without penumbra).

We compare three quantities on the period where STARA data are available (1996-2010): N_s (STARA), $\hat{\mu}_{N_s}$, and N_s as recorded by the Observatory of Locarno. These are shown in Figure 2.2. We test the level of variability by computing the mean value of a moving standard deviation over a window of 81 days. It is equal to 14.07 for N_s (STARA) re-scaled on $\hat{\mu}_{N_s}$ (5.38 for N_s (STARA) without scaling) against 15.68 for $\hat{\mu}_{N_s}$ and 27.13 for Locarno. As expected, N_s (STARA) experiences less variability than a single station but its variability is comparable to the one of our estimator.

Since satellite images of the Sun have only been available since 1980, the data extracted from those images cannot be traced back until the 17th century. $\widehat{\mu}_s(t)$ will thus be used as a proxy for $s(t)$ in the following. Note that gathering space observations during several decades also require the use of different satellites and instruments, as instruments age in space. These instruments need calibrations that create additional inaccuracies to the extracted numbers, which are thus also subject

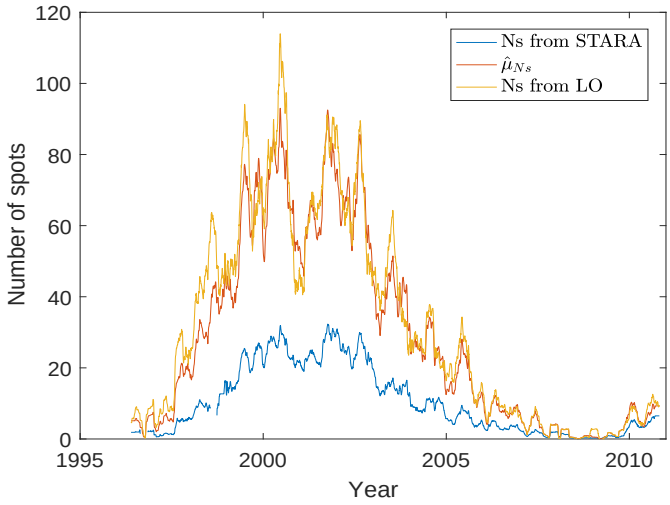


Figure 2.2: Comparison between the number of spots obtained from STARA and from our procedures, for the period May 1996 to October 2010. The number of spots obtained from the STARA catalogue is represented in blue, the actual (unprocessed) number of spots observed in Locarno (LO) in yellow, and $\widehat{\mu}_{N_s}$ is plotted in orange. The three quantities shown are averaged over 81 days.

to variations and errors.

2.5.3 Solar component for N_s and N_g

We present here the statistical modelling of the number of spots N_s and the number of groups N_g. We do this *separately* for each component since their physical origin are driven by different phenomena: the groups convey information about the dynamo-generated magnetic field in the solar interior whereas the emergence of individual spots would rather come from fragmented surface flux and agglomeration of small magnetic fields (Thomas and Weiss, 2008). Together, the analysis of the spots and groups helps us to better understand the composite N_c and the solar activity which is not satisfactorily described by only one of the two numbers (Dudok de Wit et al., 2016). In the remainder of this chapter, we define a specific notation for the generic $\hat{\mu}_s(t)$ from (2.5.1): $\hat{\mu}_{N_s}(t)$ for the number of spots, $\hat{\mu}_{N_g}(t)$ for the number of groups and $\hat{\mu}_{N_c}(t)$ for the composite.

The authors in Dudok de Wit et al. (2016) showed that the number of spots and groups experience more over-dispersion than actual Poisson variables. In order to estimate how far the distribution of the 'true' $s(t)$ departs from a Poisson distribu-

tion, we regress the conditional variance $\mathrm{Var}(Z_i(t)|\hat{\mu}_s(t) = \mu)$ versus the conditional mean $\mathbb{E}(Z_i(t)|\hat{\mu}_s(t) = \mu)$ by OLS, see Figure 2.3. Whereas in a Poisson context the slope of the fit should be close to one, for $N_s > 10$, our fit shows a slope of 1.5, with confidence interval (CI) $CI_{95\%} = [1.48, 1.51]$. This points to over-dispersion and the need for a generalization of a Poisson PDF. On the contrary, the same plot for $N_g > 0$ displays a slope of 0.96, with $CI_{95\%} = [0.93, 0.99]$, confirming the validity of a Poisson process assumption. Note that the values < 11 are excluded from the fit of N_s, as they seem to indicate a different regime. This change in the alignment may indicate the presence of a multi-modal distribution, see Figure 2.4.

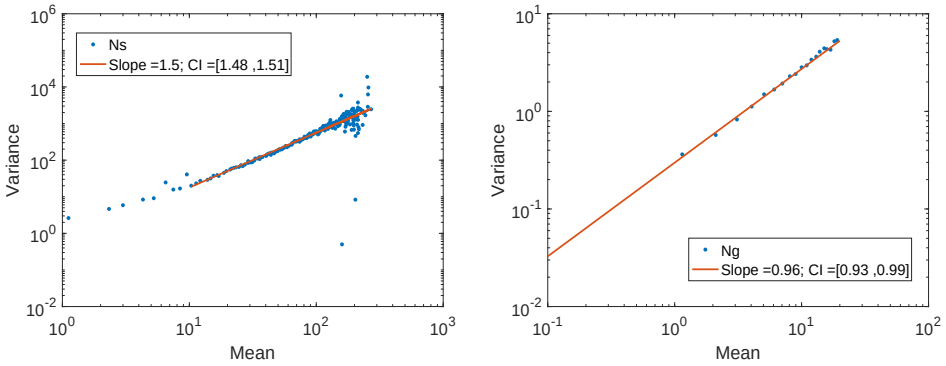


Figure 2.3: Estimation of the conditional mean-variance relationship for N_s (left) and N_g (right). The red line is a linear fit of the points (shown on a log-log scale), starting at $N_s > 9$ and $N_g > 0$, respectively. In both plots, the legend shows the value of the fitted slope together with its confidence interval at 95%. The value of the intercept is -1.21 for N_g and -0.57 for N_s. The same fit starting at $N_s > 0$ (not shown here) leads to a slope of 1.25 and $CI_{95\%} = [1.23, 1.28]$.

Count data with over-dispersion are widely modelled by the negative binomial (NB) distribution in the literature (Colin Cameron and K. Trivedi, 2013; Rodriguez, 2013) or by its generalization (Jain and Consul, 1970):

$$g(x, r, p) = \frac{\Gamma(r + x)}{\Gamma(r)\Gamma(x + 1)} p^r q^x, \tag{2.5.3}$$

where $r > 0$, $p \in (0, 1)$, $q = (1 - p)$ and Γ is the gamma function.

A visual inspection of the histogram of estimated values $\widehat{\mu}_{N_s}(t)$ in Figure 2.4(Left) reveals a local maxima in the distribution around $20 - 40$. We refer to these local maxima as *modes* in the remainder of the chapter. The underlying density of $\hat{\mu}_{N_s}(t)$ may thus be multi-modal, as suspected from Figure 2.3(Left). Such PDFs are classically modeled by a mixture model. As the density shows a typical excess of zeros as well, it requires the use of a ZA distribution defined in (2.3.5). We

thus fit the complete PDF of the estimated number of spots, $\hat{\mu}_{N_s}(t)$, by a ZA mixture of generalized NB distributions. The density at zero, $f_0(x)$, is represented by a Bernoulli distribution, whereas the density outside zero, $f_1(x)$ in (2.3.5), is identified by a mixture of NB distributions:

$$f_1(x, r_1, r_2, p_1, p_2) = w_1 g_1(x, r_1, p_1) + (1 - w_1) g_2(x, r_2, p_2), \tag{2.5.4}$$

where g_1, g_2 are NB densities and w_1 is the mixture weight.

Similarly, the histogram of $\hat{\mu}_{N_g}(t)$ exhibits a clear excess in the range $1-3$ compared to a Poisson-like distribution centred between 5 and 8. The PDF of $\hat{\mu}_{N_g}(t)$ shows thus two modes: one around $1-3$, and one around $5-8$. Such PDF may be modelled by a mixture of a NB and a Poisson distributions:

$$f(x, r_1, p_1, \mu_2) = w_1 g(x, r_1, p_1) + (1 - w_1) \frac{\mu_2^x}{x!} e^{\mu_2}, \tag{2.5.5}$$

where $\mu_2 > 0$ and, as above, w_1 is the mixture weight.

The fit of these parametric densities are shown in Figure 2.4 by a black line superimposed on the histograms. All fits in this chapter are computed using the maximum likelihood estimation (MLE). The nature of the different modes in the PDF of $\hat{\mu}_{N_s}$ and $\hat{\mu}_{N_g}$will be discussed in Section 2.5.5.

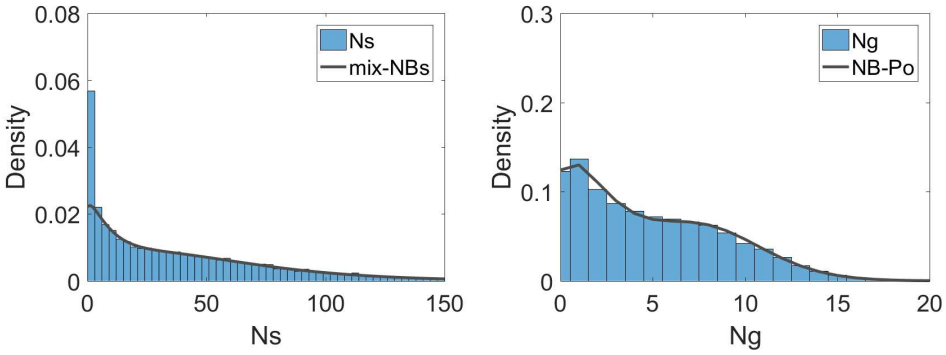


Figure 2.4: (Left) Histogram of $\hat{\mu}_{N_s}(t)$ values, computed with a bandwidth (binning) equal to 3, and estimated density for non-zero values of $\hat{\mu}_{N_s}(t)$ (shown in black line). The complete density is modelled by a ZA mixture of generalized NBs. For the zero values, the MLE value of the Bernoulli parameter is equal to $b = 0.1$. For non-zero values, the MLE values of the parameters in (2.5.4) are: $r_1 = 1.25$, $p_1 = 0.11$, $r_2 = 2.39$, $p_2 = 0.04$ and $w_1 = 0.32$. (Right) Histogram of $\hat{\mu}_{N_g}(t)$ values, computed with a bandwidth equal to one, and corresponding density fitted by MLE (shown in black line). The density is modelled by a mixture of an NB and a Poisson distributions as defined in (2.5.5). The fitted parameter values are: $\mu_2 = 8.62$, $r_1 = 1.65$, $p_1 = 0.37$, and $w_1 = 0.36$.

2.5.4 Solar component for N_c

We now use (2.5.1) to estimate the μ_{N_c}, the 'true' value of the composite $N_c = N_s + 10N_g$. By looking again at the conditional mean-variance relationship, we observe in Figure 2.5(Left) an over-dispersion with a slope of 1.29 and $CI_{95\%} = [1.27, 1.31]$ for $N_c > 20$. As a compound of both quantities, N_c experiences less over-dispersion than N_s and more than N_g. A visual inspection of the histogram of $\hat{\mu}_{N_c}(t)$ values in Figure 2.5(Right) indicates an excess of zeros and several modes, most probably coming from the modes observed in the PDFs of $\hat{\mu}_{N_g}$ and $\hat{\mu}_{N_s}$. We find thus appropriate to approximate the density of $\hat{\mu}_{N_c}(t)$ by a ZA mixture of three NB distributions, where the density outside zero values, $f_1(x)$ in (2.3.5), is identified with:

$$f_1(x, r_1, ..., r_3, p_1, .., p_3) = \sum_{i=1}^{3} w_i g_i(x, r_i, p_i), \tag{2.5.6}$$

where w_i are the mixture weights and $\sum_{i=1}^{3} w_i = 1$. The fit of f_1 is represented in Figure 2.5(Right) by a black line.

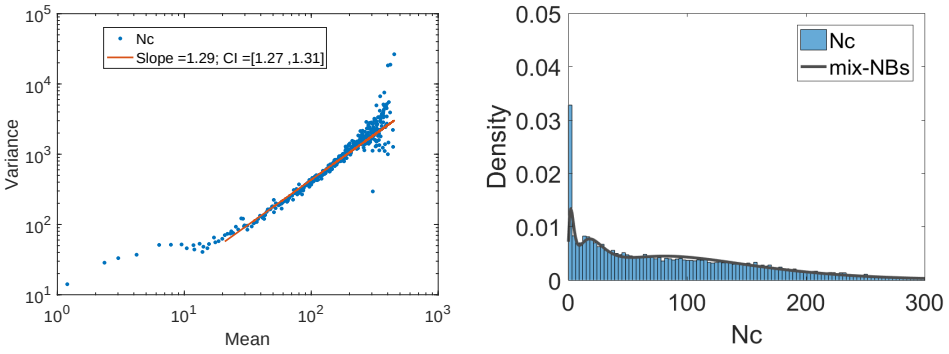


Figure 2.5: (Left) Conditional mean-variance relationship for N_c, shown on a log-log scale. (Right) Histogram of $\hat{\mu}_{N_c}(t)$ values, with a binning equal to $bw = 3$. The estimated density outside of zeros values is shown by a black line. It is modelled as a mixture of three NB distributions, see (2.5.6), with MLE parameter values equal to: $r_1 = 3.18$, $p_1 = 0.48$, $r_2 = 4.02$, $p_2 = 0.15$, $r_3 = 3.05$, $p_3 = 0.02$, $w_1 = 0.08$ and $w_2 = 0.19$. The Bernoulli parameter of the density f_0 at zero is equal to $b = 0.07$.

A statistical analysis (not presented here) shows that the distribution of the ISN is statistically close to the distribution of $\hat{\mu}_{N_c}$. The uncertainty analysis for $\hat{\mu}_{N_c}$, presented in the following, remains thus valid for the ISN.

2.5.5 Conditional Correlation

Due to the physical nature of the data, the local maxima in the densities of $\widehat{\mu}_{N_s}$ and $\widehat{\mu}_{N_g}$ are not independent. We therefore look at the conditional correlation $Corr(N_s, N_g|\widehat{\mu}_{N_c} = s)$ with the goal to better understand the nature of the modes observed in these two densities, and thus also in the density of $\hat{\mu}_{N_c}$.

Figure 2.5a shows the conditional correlation for different values of $\hat{\mu}_{N_c}$ between 0 and 400. Note that even when $\hat{\mu}_{N_c} = s$, the value of the composite $N_s + 10N_g$ for a particular station may be larger (resp. smaller) than s. Our analysis highlights three regimes of activity:

Minima $\hat{\mu}_{N_c} \in [0, 11]$. Here, the number of spots and groups oscillates between 0 or 1. As the number of spots equals exactly the number of groups, the correlation is high.

Medium activity $\hat{\mu}_{N_c} \in [12, 60]$. The correlation progressively decreases, because the number of spots increases faster than the number of groups, and then stabilizes. This regime is characterized by the development of smaller spots without penumbra or with a small penumbra. Figure 2.5b shows the bivariate boxplot of N_s and N_g when $\hat{\mu}_{N_c} = 40$. For $N_g = 1$ or $N_g = 2$, we observe values of N_s as high as 40. We clearly observe groups containing a large number of spots as well as groups, composed of fewer spots, that appear progressively as the penumbra grows and that indicate a transition toward groups with fewer but larger spots. The effect of this transition from small to larger spots is observed in Figure 5 from Clette and Lefèvre (2016).

High activity $\hat{\mu}_{N_c} > 60$. Figure 2.5c shows the bivariate boxplot of N_s and N_g when $\hat{\mu}_{N_c} = 70$. The plot has a potato-shape around ($N_s = 30$, $N_g = 4$). We now observe all kinds of groups. The correlation between groups and spots slightly increases as the number of groups begins to grow as well.

The *medium* and *high* regimes are reflected in the estimated densities of $\widehat{\mu}_{N_s}$ and $\widehat{\mu}_{N_g}$ in Figure 2.4. The first mode of $\hat{\mu}_{N_g}$ ranging from 1-3 corresponds to the *medium* regime while the second from 5-8 reflects the *high* regime. The two distinct regimes provide another justification for the use of a multi-modal distribution to characterize the PDF of $\hat{\mu}_{N_g}$. Similarly, there is also a mode in the distribution of $\widehat{\mu}_{N_s}$ around 20-40 that comes from the transition between the *medium* and *high* regime. The mode is correctly represented by a mixture model. The study of the conditional correlation constitutes the first step towards retrieving the distribution of N_c from its composites N_s and N_g. However, this task is challenging and goes beyond the scope of the work because: (1) the distributions of N_s and N_g are complex mixtures and (2) the number of spots is non-trivially correlated to the number of groups.

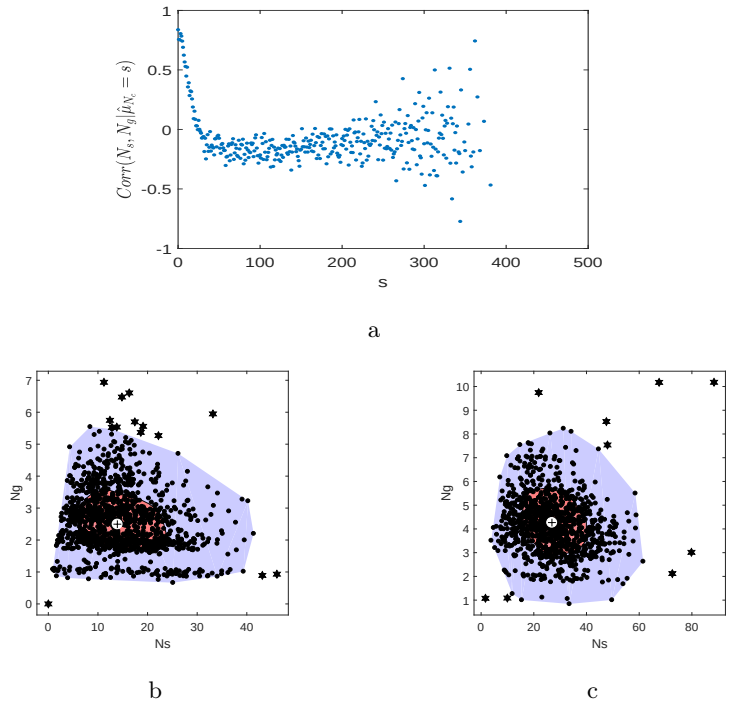


Figure 2.6: (a) Conditional correlation of N_s and N_g: $Corr(N_s, N_g|\hat{\mu}_{N_c} = s)$ for $s \in [0, 400]$. (b) Bivariate boxplot (also called 'bagplot') of N_s and N_g when $\hat{\mu}_{N_c} = 40$ and (c) when $\hat{\mu}_{N_c} = 70$. The white cross represents the depth median (Rousseeuw et al., 2012). The bag contains 50% of the observations and it is represented by a polygon in red. The fence (not represented) is obtained by inflating the bag by a factor three. The observations that are outside of the bag but inside of the fence are indicated by a light grey loop. Outliers are represented by a black star. The correlation is indicated by the orientation of the bag.

2.6 Distributions of errors

We are now in a position to analyse the error distributions in the sunspot counts, i.e. to model the distributions of ϵ_3, ϵ_1 and ϵ_2. To this end, we separate minima from non-minima regimes. We also consider two time-scales: short-term periods, that is, time-scales smaller than one solar rotation (27 days) and long-term periods. Section 2.6.1 estimates the error at solar minima, i.e. when $s(t) = 0$. Section 2.6.2 analyses the short-term variability of the pre-processed observations when $s(t) > 0$. For the study of long-term error in Section 2.6.3, we use raw data that did not undergo any preprocessing, in order to be able to detect sudden jumps

and/or large drifts in the series. The correct time-scale for the long-term period is also determined in this section, based on a statistically-driven procedure. Finally, Section 2.6.4 compares the characteristics of the different stations based on the error analysis.

2.6.1 Error at minima

The study of solar minima periods is complex as the data show a large variability and dichotomy. Observed values of the error at minima, ϵ_3, are defined as counts made by the stations when the proxy for $s(t)$, defined in (2.5.1), is equal to zero:

$$\hat{\epsilon}_3(i,t) = Z_i(t) \text{ when } \hat{\mu}_s(t) = 0, \tag{2.6.1}$$

where $Z_i(t)$ corresponds either to N_s, N_g or N_c, and where the generic $\widehat{\mu}_s(t)$ has to be replaced by $\widehat{\mu}_{N_s}(t)$ for N_s, $\widehat{\mu}_{N_g}(t)$ for N_g, and $\widehat{\mu}_{N_c}(t)$ for N_c.

A visual inspection of the histogram of N_s (resp. N_g) in Figure 2.7a (resp. Figure 2.7b) shows an important amount of 'true' zeros together with two modes around one and two. Similar modes occur around 11 and 22 in the distribution of N_c in Figure 2.7c, as expected. These modes represent short-duration sunspots. Due to the non-simultaneity of the observations between stations, the proxy for $s(t)$ might be equal to zero even if some spots appear shortly (from several minutes to several hours) on the Sun. These modes can be represented by a t-Location-Scale (t-LS) distribution, which is a generalization of the Student t-distribution. This distribution has three parameters to accommodate for asymmetry and heavy tails: the location μ, scale $\sigma > 0$, and shape $\nu > 0$ (Taylor and Verbyla, 2004; Evans et al., 2000). Its PDF is defined as:

$$g(x,\mu,\sigma,\nu)_{t-LS} = \frac{\Gamma(\frac{\nu+1}{2})}{\sigma\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\left(\frac{\nu + \frac{(x-\mu)^2}{\sigma^2}}{\nu}\right)^{-(\frac{\nu+1}{2})}. \tag{2.6.2}$$

The large proportion of zeros values for $\hat{\epsilon}_3$ requires the use of a ZA model as in (2.3.5). We choose a ZA mixture of t-LS for the complete distribution of $\hat{\epsilon}_3$. The density outside of zero, $f_1(x)$ in (2.3.5), is thus identified by such a mixture of t-LS distributions:

$$\begin{aligned} f_1(x,\mu_1,\sigma_1,\nu_1,\mu_2,\sigma_2,\nu_2) =& w_1 g(x,\mu_1,\sigma_1,\nu_1)_{t-LS} \\ &+ (1-w_1) g(x,\mu_2,\sigma_2,\nu_2)_{t-LS} \end{aligned} \tag{2.6.3}$$

where, as before, w_1 is the mixture weight. The histograms and fitted distributions for $\widehat{\epsilon}_3$ are shown in Figure 2.7. The visual closeness between the histograms and the fitted distributions was used as a criterion to select the best PDFs among a few intuitive candidates, while the parameters of the distributions are estimated

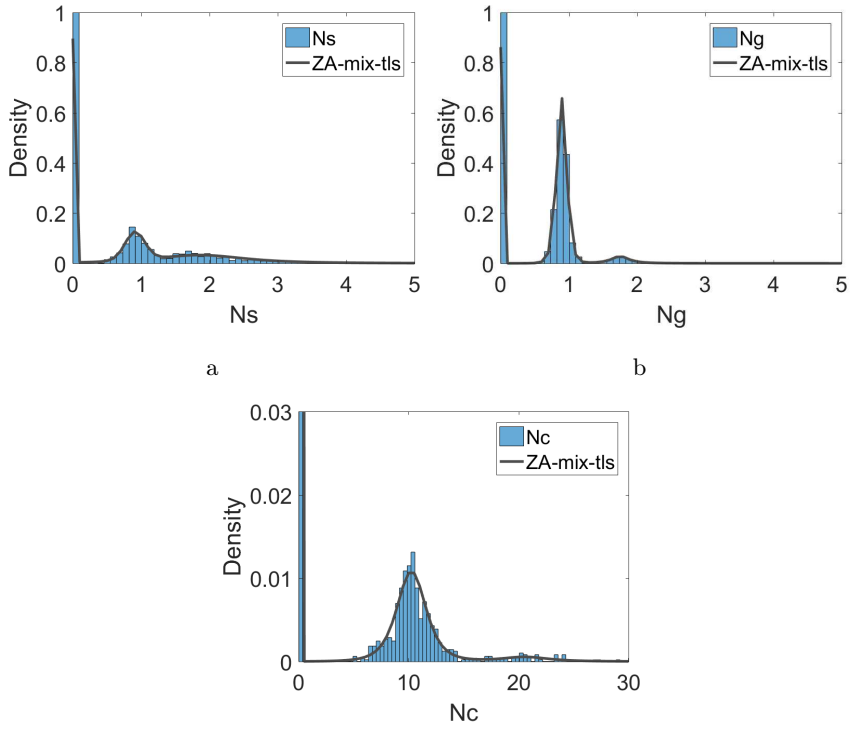


Figure 2.7: Truncated histograms of $\hat{\epsilon}_3$ for N_s (a), N_g (b) and N_c (c). The continuous line shows the fits using a ZA-mixture of t-LS distributions, defined in (2.6.3). The values of the Bernoulli parameter in (2.3.5) are equal to: (a) b=0.9, (b) b = 0.86, and (c) b= 0.96 They represent the proportion of 'true' zeros. The parameter values for the t-LS fit are: (left) for N_s: $\mu_1 = 0.91$, $\sigma_1 = 0.14$, $\nu_1 = 31.16$, $\mu_2 = 1.85$, $\sigma_2 = 0.71$, $\nu_2 = 2.09$, $w_1 = 0.6$; (center) for N_g: $\mu_1 = 0.89$, $\sigma_1 = 0.07$, $\nu_1 = 6.89$, $\mu_2 = 1.75$, $\sigma_2 = 0.14$, $\nu_2 = 1.33$, $w_1 = 0.09$; (right) for N_c: $\mu_1 = 10.24$, $\sigma_1 = 1.37$, $\nu_1 = 3.89$, $\mu_2 = 20.57$, $\sigma_2 = 2.33$, $\nu_2 = 1.93$, $w_1 = 0.08$. The bin-width ($bw = 0.0917$) is the same for the histograms of both N_s and N_g. It is related to the sample size and the data-range of N_s by a simple rule proposed by Scott (Scott, 1979). The bin-width of the histogram of N_c (c) is equal to $bw = 0.4192$ and is also computed by Scott's rule. Note that the right figure is enlarged: the value at zero is 0.96 and not 0.03.

via MLE.

In the previous figures, where the error at minima is represented for all stations combined, outliers defined as $\hat{\epsilon}_3(i,t) > 2$ are not visible for N_g and N_s. A separate

analysis (not presented here) shows that the percentage of outliers in each station are low (inferior to 0.5% for N_s). Some stations also observed a high maximal value at minima (e.g. a value of 35 was recorded in QU (Quezon, Philippines) for N_s). This extreme value for a minima may correspond to a transcription error that might be verified in the future, before being encoded in the WDC-SILSO database.

2.6.2 Short-term variability

When the proxy for $s(t)$, defined in (2.5.1) is different from zero, the short-term error $\widetilde{\epsilon}$ may be estimated using:

$$\widehat{\widetilde{\epsilon}}(i,t) = \frac{Z_i(t)}{\hat{\mu}_s(t)} \quad \text{when } \hat{\mu}_s(t) > 0. \tag{2.6.4}$$

To select the best distribution, we proceed as follows. Different densities are fitted to the values of $\widehat{\widetilde{\epsilon}}$ outside of zero, using MLE.[3] Then, the AIC criterion is used to choose the best PDF, which in this case is a t-LS distribution.

As we observe an excess of zero, we need a ZA t-LS distribution to represent the complete distribution of $\widehat{\widetilde{\epsilon}}$. Figure 2.8 shows the histogram as well as the fitted PDF of $\widehat{\widetilde{\epsilon}}$ outside of zero. For the latter, the mean is close to 1, indicating that on average the stations are aligned with $\hat{\mu}_s(t)$. The histogram exhibits a probability mass at zero representative of 'false' zeros, i.e. of stations that do not observe any sunspot when there are actually some on the Sun. The histogram also shows a tail on the right-hand side, caused by outliers. This asymmetry requires a t-LS rather than a Gaussian distribution to be fitted.

The violin plots of four different stations are shown in Figure 2.9 for the number of spots N_s, where the differences between the stations are the most visible. The mean of the Observatory of Locarno (LO), the current reference of the network, is slightly higher than the three other means (and higher than the means of all other stations), around 1.19. This results from its particular way of counting: large spots (with penumbra) count for more than small spots without penumbra.

Another characteristic feature is how the error is distributed around the mean. A violin plot may be seen as a PDF with the x-axis of the density drawn along the vertical line of the boxplot. For example, the PDF of the short-term error of LO is concentrated around the mean but the entire distribution is shifted upward unlike the PDF of Uccle (UC) which has much lower values. UC is a professional observatory. Different observers record from one week to another the number of spots, groups and composite on the Sun. As their experience and methodology slightly change, the shift of observers probably increases the short-term variability

[3] We use the function ''allfitdist.m', last modified in 2012, in Matlab R2016b.

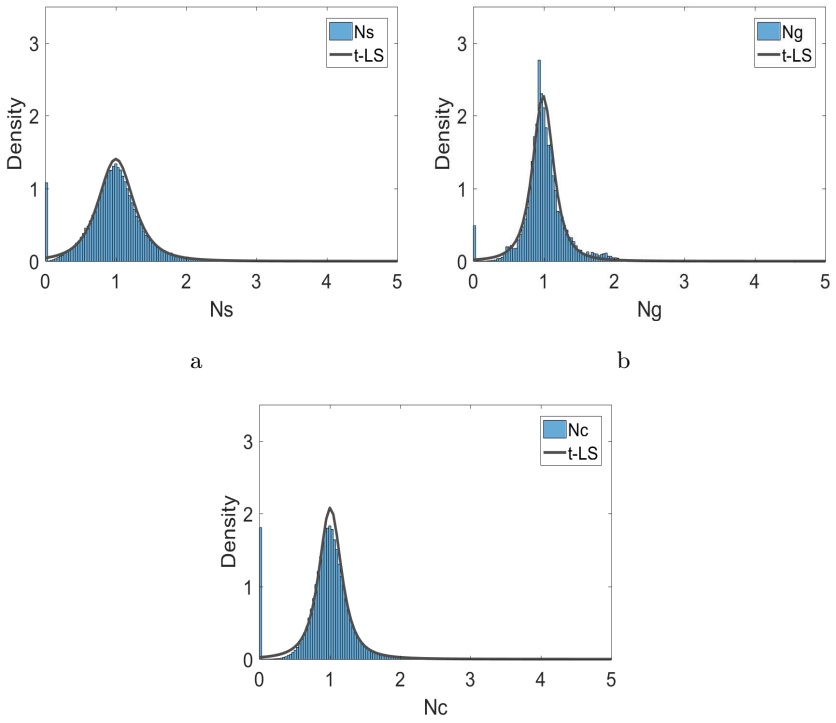


Figure 2.8: Histograms of $\hat{\tilde{\epsilon}}$ for N_s (a), N_g (b) and N_c (c). The continuous line shows the fits using a t-LS distribution defined in (2.6.2). The values of the Bernoulli parameter in (2.3.5) are equal to: (a) $b = 0.04$, (b) $b = 0.02$, (c) $b = 0.06$. They represent the proportion of false 'zeros', i.e. stations reporting no sunspot where there are some. The parameter values for the t-LS fit are: (a) for N_s: $\mu = 1$, $\sigma = 0.26$, $\nu = 2.8$; (b) for N_g: $\mu = 0.99$, $\sigma = 0.16$, $\nu = 2.33$; (c) for N_c: $\mu = 1.01$, $\sigma = 0.17$, $\nu = 2.12$. The bin-widths bw of the histograms are computed using Scott's rule. For N_s and N_g they are the same ($bw = 0.0328$) and for N_c it is equal to $bw = 0.0433$.

of the station. Usually a team of observers experience more variability than a single person, like in FU (Fujimori, Japan). This station has a remarkable short-term stability.
Similarly, the Observatory of San Miguel (SM) shows the typical shape of a professional observatory. On the other hand, the LO station shows an $\widehat{\tilde{\epsilon}}$ distribution almost characteristic of a single observer: that is because until recently LO had one main observer.

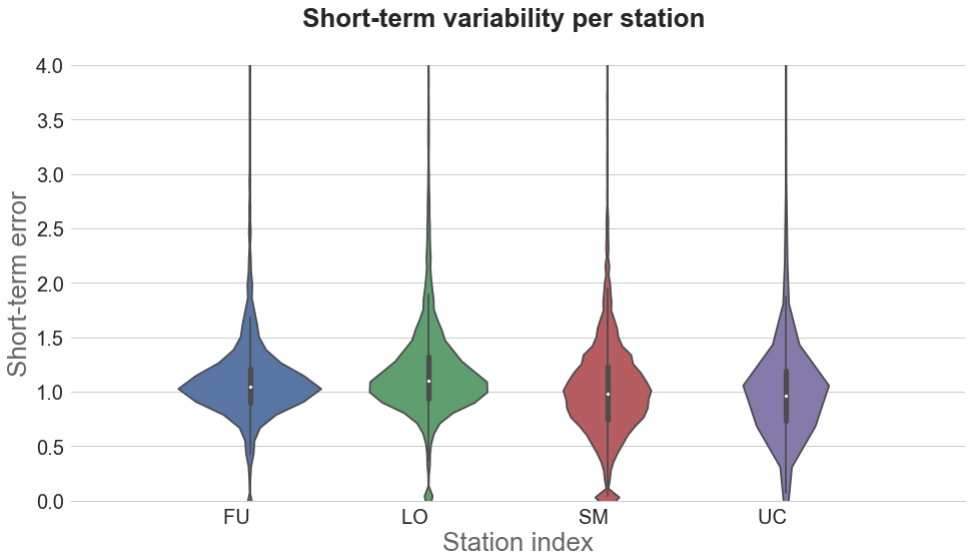


Figure 2.9: Truncated violin plots of the estimated short-term variability $\widehat{\widetilde{\epsilon}}$ for N_s in four stations (FU, LO, SM and UC). A violin plot (Hintze and D., 1998) combines a vertical box-plot with a smoothed histogram represented symmetrically to the left and right of the box. The white dot in the centre of the violin locates the mean of the distribution. The thick grey bar shows the interquartile range and the thin grey bar depicts the interdecile range. The bin-width ($bw = 0.05$) is the same for all stations and is computed with Scott's rule.

2.6.3 Long-term variability

A generic estimator for the long-term error $\epsilon_2(i,t)$ may be defined by:

$$\hat{\mu}_2(i,t) = \left(\frac{Y_i(t)}{M_t}\right)^{\star} \quad \text{when } M_t > 0, \tag{2.6.5}$$

where the $\star$ denotes the smoothing process by a moving average (MA) filter, $Y_i(t)$ are the raw observations and $M_t = \underset{1\leq i\leq N}{\text{med}}\, Z_i(t)$ is the median of the network. The T transform from (2.5.1) is not required here, as we apply a MA filter of length L.

This length L should be larger than what is considered as short-term, that is, periods inferior to one solar rotation (27 days). Long-term on the other hand is usually defined as periods above 81 days (Dudok de Wit, 2011), beyond which the effects of the solar rotation and of the sunspot's lifetime are negligible. The mid-term temporal regime corresponds to periods between 27 and 81 days. Depending on our interest in detecting long-term drifts or higher-frequency deviations such

as jumps, different window lengths may be chosen in (2.6.5) (yet above 27 days). Indeed, drifts require long smoothing periods (several months, or even years) to be observed whereas jumps might be over-smoothed by such long smoothing and hence need a smaller MA window.
Note that other smoothing processes exist in the literature. We use a MA filter here since this method is particularly suited for smoothing data which contain many missing values.

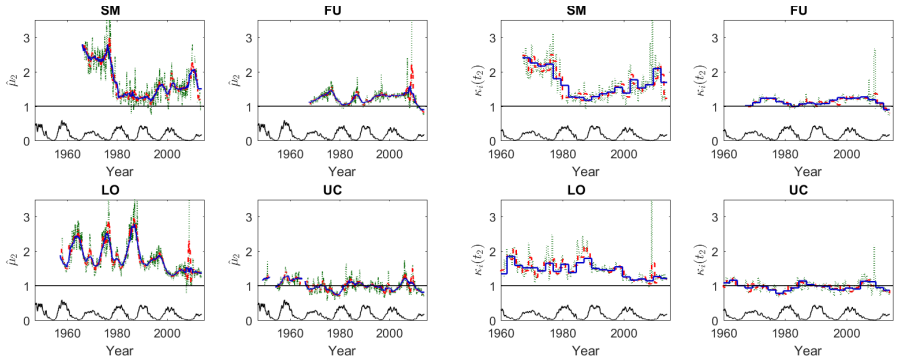


Figure 2.10: (a)-(d): Estimation of $\hat{\mu}_2(i,t)$ for N_s in four stations (SM, FU, LO and UC). $\hat{\mu}_2(i,t)$ is computed with different MA window lengths: 81 days (green dotted line), 1 year (red dashed line) and 2.5 years (blue plain line). (e)-(h): Estimation of the scaling-factors for N_s in the same stations. The $\kappa_i(t_2)$s, with $1 \leq t_2 \leq T/\tau$, are computed using the $OLS(\vec{Y}_{i,t_2}|\vec{X}_{,t_2})$ regression in (2.4.1) on a block of $\tau = 81$ days (green dotted line), 1 year (red dashed line) and 2.5 years (blue plain line). The solar cycle is represented in black at the bottom of the figures for N_s.

Figure 2.10(a)-(d) represent the $\hat{\mu}_2(i,t)$s associated to N_s in four stations starting from 1960. Figure 2.10(e)-(h) show the scaling-factors $\kappa_i(t_2)$s for the same stations used at short-term and minima regimes. We do not represent years before 1960 because FU and SM show too few observations in that period. FU and UC appear relatively stable, unlike stations LO and SM, which display severe drifts. Bias in the counting process is also larger during solar minima when there are short-duration sunspots. This effect is clearly visible in LO. Indeed, erroneous encoding of counts leads to much higher relative errors during minima than during the remaining part of the solar cycle. Some high-frequency deviations are also visible on the graphs with the smallest MA length (81 days) in green. This scale is more appropriate to observe them (although it may already suppressed the most sudden jumps), while longer scales only highlight the long-term drifts of the stations.
We emphasize here the strong link between the preprocessing and the long-term analysis. Indeed, the scaling-factors presented in Section 2.4 are a rough estimate

of the long-term error, inspired by the historical procedure of J.R. Wolf. This rough estimation is required to rescale the stations to the same level. This rescaling is used to compute the median M_t of (2.6.5). Contrarily to the piecewise constant κ_is computed in Section 2.4, the $\hat{\mu}_2(i,t)$s are smooth over time and hence are more adapted to a future monitoring of the stations.

2.6.4 Comparing stations with respect to their stability

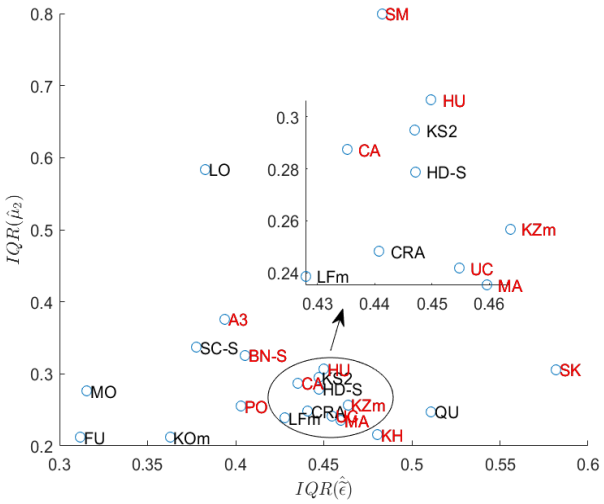


Figure 2.11: Scatter plot showing the interquantile range of the estimated short-term error $\hat{\tilde{\epsilon}}(i,t)$ and the interquantile range of the estimated long-term error $\hat{\mu}_2(i,t)$, station by station. Stations in red represent the teams of observers, the others are single observers.

In previous sections, we presented separately the estimations of the short-term error $\hat{\tilde{\epsilon}}(i,t)$, the long-term error $\hat{\mu}_2(i,t)$ and the error at minima $\hat{\epsilon}_3(i,t)$. All three types of errors are however needed to assess the quality and stability of a station. Figure 2.11 displays a visual representation of long-term against short-term error for each station. It shows the long-term versus short-term empirical interquantile range on a 2D plot and characterizes thus the stability of the stations outside of minima. Stations in red are the teams of observers. They usually experience more short-term variability than an individual. We see that MO (Mochizuki, Japan), FU and KOm (Koyama, Japan) have low-variability both on short and long-term. They correspond to long individual observers with stable observation practises. On the other hand, the LO station shows a poor long-term stability, while its short-term variability is remarkably low for a professional observatory. As mentioned earlier, this is due to the fact that there is a main observer. UC shows a large variability in

the short-term (due to many observers) but an interesting long-term stability, as already noticed in Figure 2.9. SM experiences the most severe long-term variability of the network. It has also a large short-term variability, characteristic of a team of observers. QU shows a large short-term variability and a low long-term variability level. Although it seems it is a single observer, it appears there was a move from one place to another during the observing period, and maybe a change of instrument that would impact the short-term variability. This surprising behaviour will prompt the WDC-SILSO team to ask for more metadata.

2.7 Conclusion and future prospects

In this chapter, we propose the first comprehensive uncertainty model in a *multiplicative* framework for the number of spots, groups and composites. Our approach is robust to missing values and was applied on 66 years of data (1947-2013). We presented several parametric models for the density of the 'true' solar signal underlying N_s, N_g and N_c, as well as for the density of their error distributions at minima, short-term and long-term. This error quantification allows proposing a first classification of the subset of 21 stations selected for this analysis based on their stability. It shows that the observing stations are affected differently by the various types of errors: some are stable with respect to the network at short-term but experience large drifts and conversely. The analysis also highlights the hazards of using a single-pilot station as the unique reference of the network.

The error models presented in Section 2.6 may be used later for a parametric monitoring of all stations of the network, with a particular focus on new stations. Data from new-born observing stations can be recorded for several months. Their distributions may then be compared to the density of the short-term error (or the error at minima if we are in a minima period) of the entire network obtained in this chapter. If the stations experience similar errors, they may be included in the network. Otherwise, the stations might need to improve or correct their observing procedure before entering the WDC-SILSO network.
A non-parametric monitoring that aims at detecting in quasi real-time the long-term drifts of the stations will also be developed in the next chapter. This procedure will be designed to detect as soon as possible the various types of deviations that appear in the stations, to prevent large drift such as those observed in Figure 2.10 from occurring.
Furthermore, the work presented here enhances our comprehension of the ISN (and its errors), which is widely used in astrophysics and space weather physics. A better modelling of the ISN improves thus not only the understanding and the quality of data but also those of the models which are built upon it.

The estimators and the model developed in this chapter can also be inspirational

for analysing other datasets with similar features, as we will see later in Chapter 5. The multiplicative framework, the non-normal distributions, the different time-scales and the missing values of the data are indeed common features in several applications.

2.8 Appendix

2.8.1 Time-scales of the preprocessing

This appendix details the statistical procedure which selects the time-scales of the preprocessing described in Section 2.4. It is composed of three steps.
First, the daily scaling-factors are computing using:

$$\kappa_i((t_1,t_2)) = \frac{Y_i((t_1,t_2))}{\mathrm{med}_{1\leq i\leq N} Y_i((t_1,t_2))} \tag{2.8.1}$$

where, as in Section 2.4, we rewrite the time by an array of two indices $1 \leq t_1 \leq 30$ and $1 \leq t_2 \leq T/30$, corresponding respectively to the day and the month of the observation.

Second, the non-parametric Kruskal-Wallis test (KW) (Kruskal and Wallis, 1952) is applied on blocks of 30 factors, since the 'k-coefficients' of (1.2.2) are currently estimated on a monthly basis at the WDC-SILSO.
Let $\vec{\kappa}_{i,t_2} = [\kappa_i((t_1,t_2))]_{1\leq t_1\leq 30}$ denote the vector of the daily factors on one month. The test assesses whether the $\vec{\kappa}_{i,t_2}$s of consecutive months are statistically different. The procedure starts by comparing the distribution of the first month of the period studied, $\vec{\kappa}_{i,1}$, to the distribution of the second month, $\vec{\kappa}_{i,2}$. If the test shows that both distributions are significantly different, the two next distributions $\vec{\kappa}_{i,2}$ and $\vec{\kappa}_{i,3}$ are tested. Otherwise, the distribution of the two first months $[\vec{\kappa}_{i,1}\ \vec{\kappa}_{i,2}]$ is compared to the distribution of the third month $\vec{\kappa}_{i,3}$. The algorithm is iterated until the end of the period, for each station. Note that the KW test performs well when comparing two or more independent samples of unequal sizes. The correlation of the data is thus neglected in this procedure. Despite the presence of correlations between consecutive days, the correlation between consecutive months is low. The test provides thus a station-specific segmentation, shown in Figure 2.12 for N_s. The length of the segments indicates the number of consecutive blocks of scaling-factors that come from the same distribution. We assume that these factors are constant within each segment.

In the last step, the global time-scales for each index are defined from the segmentations of the individual stations. The length of the most frequent segment is first selected in each station. Then a global scale is estimated from the median of the most frequent lengths by station, for N_s, N_g and N_c.

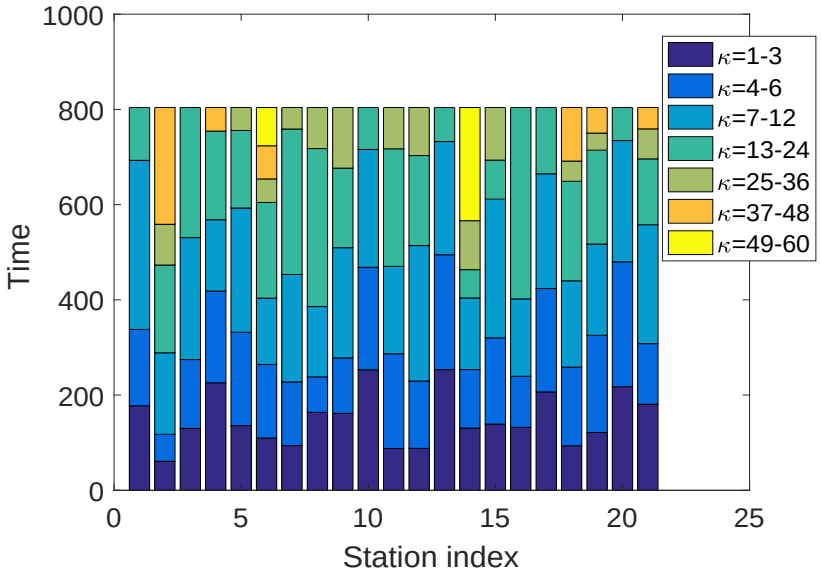


Figure 2.12: Bar-chart representing the results of the KW test applied to the scaling-factors for N_s. The x-axis represents the stations indexed from 1 to 21 and the y-axis shows the total period studied expressed in months (1 unit $\approx$ 30 days). The y-axis is not ordered in time, for readability purpose, but it is ordered with respect to the length of the segments. The colours of the chart correspond to the number of blocks that may be grouped into a single factor ('$\kappa = 5$' means that a single scaling-factor may be computed for a period of 5 months).

CHAPTER 3

Non-parametric monitoring of time-series panel data applied to the sunspot numbers

In many applications, a control procedure is required to detect the potential deviations of a process. The most trivial example being the need of constantly supervise the products quality in manufacturing industries. Many other situations can however benefit from such a procedure. Patients with predisposition to heart attacks could be saved by the early detection of any abnormality in their physiological condition. The constant evaluation of air-planes performances is also required to ensure the safety of the passengers and is thus integrated in the maintenance plans. More specifically, as previously demonstrated, the sunspot numbers recorded in some stations have irregular behaviours over time. They experience deviations than can last several years. Hence, a monitoring procedure that could early detect those deviations and send alerts to the observers would strongly improve the quality and stability of the series over the years.

In this chapter, we propose the first systematic and thorough statistical approach for monitoring the sunspot numbers. The method consists of three steps: smoothing on multiple time-scales, monitoring using block bootstrap calibrated CUSUM charts and classifying of out-of-control situations by support vector techniques. The procedure is tailored to cope with the low signal-to-noise ratio, the autocorrelation and the missing values of the data. Moreover, the problem of absence of in-control series is tackled by an intelligent exploitation of the information that is contained in the whole panel.
This approach allows us to detect a wide range of anomalies (such as sudden jumps or more progressive drifts), unseen in previous analyses. It helps us to identify the causes of major deviations, which are often observer or equipment related. Their

detection and identification will contribute to improve future observations. Their elimination or correction in past data will lead to a more precise reconstruction of the International Sunspot Number.
Furthermore, since many aspects of the method are general, the monitoring scheme can also be applied to a much wider range of problems. Examples of other applications will be developed in Chapter 5.

3.1 Introduction

As stated in Chapter 1 and described in details in Chapter 2, the sunspot numbers are subject to many kinds of deviations at different time-scales. Those can be related to the observing conditions (instruments, counting methodologies, observers, etc.) or to the solar variability (such as e.g. short-duration sunspots). We are thus facing a panel of observations with time-varying disturbances of various kinds such as jumps, drifts or oscillations. Preserving the quality of the series therefore calls for a robust and automated tool for supervising the observations in quasi real-time. This procedure should monitor the stations and send alerts when they start deviating, to prevent the build-up of large drifts.

Until recently, such a monitoring could not be developed with the available statistical methods due to the complexity of the series. Moreover, existing modelling and error quantification of the data were lacking. Advances in statistical process control (SPC) for panel data combined with the uncertainty model for the sunspot numbers that we developed in Chapter 2, allow us now to construct an effective monitoring method for these data.

3.1.1 Related works

Many different procedures have been developed in the SPC literature to monitor the mean of univariate processes (Qiu, 2013; Montgomery, 2004). Some of them such as the classical cumulative sum (CUSUM) chart (Page, 1961) are described later in Section 3.2. Those methods cannot be directly used here however. Indeed, if each station is an univariate process, its mean and variance change over time, due to e.g. the eleven year solar cycle (Hathaway, 2010). Some stations are in addition deviating in their entire observing period and hence do not have IC periods. All stable stations should therefore be used together to judge if a particular station is deviating.
To this end, we propose in the following a method based the dynamic screening system developed by Qiu and Xiang (2014). To the best of our knowledge, this method is the only one that can be adapted to the particular characteristics of the sunspot data: the non-normality, autocorrelation and non-stationarity of the data

as well as the absence of IC periods in all series. The method is composed of the two following steps. First, the regular patterns (i.e. the mean and variance) of the data are estimated on a group of non-deviating or in-control (IC) series from the panel. The data are then standardised by these patterns. In a second step, the standardised data are monitored by a CUSUM chart designed by a block bootstrap method, also reviewed in Section 3.2. This procedure constructs, without any parametric assumption, a control scheme that is valid for non-normally distributed and serially correlated data.
With this method, we can estimate the regular patterns of the data locally in time, since we have at each time point a collection of stable series at our disposal. The method can therefore detect shifts in the mean level of each series, where the means change over time. It can thus accommodate the intrinsic quasi-periodic variations in the sunspot numbers that are related to the solar cycle and are thus intrinsic to the signal.
This approach is thus partially multivariate since different parameters of the method such as the regular patterns or the control limits of the CUSUM chart are estimated on several (IC) series. The CUSUM chart is however applied on each process separately afterwards. As we are interested in supervising and analysing the deviations of each individual series, this approach appears particularly suited for our problem. Note that although the ISN is obtained by combining the observations of the panel, the final aim is not to monitor this index directly but to compute it from a subset of non-deviating series, which will be selected by the proposed method.

In the following, we use and extend the work of Qiu and Xiang (2014), to bridge the gaps between the method and the specific requirements of our problem. Those gaps are two-fold.
(1) The method of Qiu and Xiang (2014) —as all other methods that we encountered in the literature— cannot be used without knowing a priori which stations are in-control. This information is not available for our data, where *all* stations are expected to contain several kinds of deviations in their observing period.
(2) The method operates with a control chart which sends an alert when a deviation is detected, yet without providing any information about the nature of the shift. Such information is however crucial for us, since it allows to further investigate the causes of the shifts. Although several methods have been developed to automatically predict the shift size after an alert (see for instance Cheng et al. (2011) and the references therein), they are not adapted to data which are simultaneously non-normally distributed, serially correlated and contaminated by strong noise.

3.1.2 Aims

In the following, we propose a nonparametric monitoring that is tailored to the complex features of the sunspot numbers: (a) the missing values, (b) the strong

noise, (c) the complex autocorrelation structure and (d) the non-normality. Our method extensively exploits the information contained in the panel to establish a robust IC reference from the network. This allows us to monitor the stations without prior information on their stability. We complete the method by a support vector machine (SVM) procedure that efficiently predicts the size and shape of a shift once an alert has been raised. Although we could manually build a library with typical shapes and sizes to be compared to the deviations, we select the automatic SVM approach instead.
The control scheme is then applied on past observations to study the deviations of the sunspot numbers. The procedure automatically detects major deviations identified recently by hand in some stations. It also unravels many other deviations, unseen in previous analyses. In particular, small and persistent shifts that are difficult to identify manually are detected by the method. The precise information about the deviations predicted by the SVM procedures allows us to determine the causes of some prominent deviations. This sets the ground for a future enhancement of the quality of the series. Moreover, the monitoring procedure provides the possibility to be used in real-time to preserve the long-term stability of the stations. It also paves the way to a future redefinition of the International Sunspot Number based on several stations that are stable over time.

This chapter is structured as follows. A general introduction to the statistical process control is given in Section 3.2. It includes a detailed description of the CUSUM chart and its design, either for independent and identically distributed (i.i.d.) normal data or for more general non-normally distributed and autocorrelated processes. While the former is based on a simple algorithm, the latter involves in addition a block bootstrap procedure. The block bootstrap methods are thus also reviewed in the section. In Section 3.3, we present the sunspot data and the precise quantity that will be monitored in this chapter. The methods are then explained in Section 3.4, including the complete monitoring scheme as well as the SVM procedures to predict the size and shape of the deviations. In Section 3.5, we apply the proposed method on the sunspot data at different scales, in order to detect both high- and low- frequency shifts and discuss the results. In a last Section 3.6, we give some concluding remarks and perspectives. Appendices also provide some more details about the monitoring scheme as well as more examples of monitored stations.

3.2 Control charts

As stated at the beginning of the chapter, many applications are concerned about ensuring the quality of products or preserving the stability of processes over time. To that end, different methods of statistical process control (SPC) have been de-

veloped in the literature. Those can be divided into two phases. In the first stage called phase I SPC, the characteristics of the in-control (IC) or non-deviating process of interest are studied. An IC dataset is collected and used to estimate the IC distribution of the process. In the second stage called phase II SPC, a monitoring procedure is then applied to the data. When a significant deviation is detected, the process is considered to be out-of-control (OC) and an alert is triggered. In manufacturing, the production is most of the time stopped until the cause of the deviation has been found and corrected. In other applications where the process cannot be stopped (the physiological condition of patients for example), other measures can be taken to prevent further complications. The root-causes of the shift may also be studied to limit the occurrence of future deviations.

In the following, we first explain the concept of average run length (ARL), which will be used to design and assess the performances of the monitoring methods. Then, the Shewhart and CUSUM control charts, both widely-used SPC methods, are described for independent and identically distributed (i.i.d.) normal data. The block bootstrap is finally introduced, to allow the calibration of the charts for non-normally distributed and autocorrelated data.

3.2.1 Average run length

The number of observations collected from an initial arbitrary time point to the alert of the chart is defined as the run length. The performances of the control charts are usually measured using the concept of average run length (ARL). The in-control (IC) ARL, denoted by ARL_0, is the mean value of the run length when the process is IC. It represents the rate of false positives of the chart and is similar to the concept of type I error in hypothesis testing context. Large values of ARL_0 are desirable since they reduce the number of false alerts.
The out-of-control (OC) ARL, denoted ARL_1, is the mean of the run length when the process is OC. It corresponds thus to the mean value of the number of samples collected from the appearance of a shift to the alert of a chart. It embodies the detection power of the chart, such as the type II error in hypothesis testing. Smaller values of ARL_1 are thus pursued since they correspond to a greater detection power.

In practice, as with hypothesis testing, it is hardly possible to design a chart with ARL_0 as large as possible while maintaining at the same time a small value of ARL_1. This may be understood by the following argument.
If the process is IC with a probability α of triggering a false alert at each observation, the IC run length has a geometric distribution, $\mathcal{G}eom(\alpha)$. Its mean and

standard deviation are then equal to:

$$ARL_0 = \frac{1}{\alpha}, \quad \sigma^0_{RL} = \frac{\sqrt{1-\alpha}}{\alpha}. \tag{3.2.1}$$

Small values of α lead thus to high values of ARL_0.
If the process is, on the other hand, OC with a probability $1-\beta$ of triggering a (true) alert at each observation, the OC run length follows then a geometric distribution, $\mathcal{G}eom(1-\beta)$, with:

$$ARL_1 = \frac{1}{1-\beta}, \quad \sigma^1_{RL} = \frac{\sqrt{\beta}}{1-\beta}. \tag{3.2.2}$$

A chart which is designed to trigger very few false alerts is particularly cautious. It takes more time to identify a deviation that the same chart calibrated to give a higher rate of false alerts. Hence, small values of α are also related to high values of β, i.e. the probability of not giving an alert when the process is OC. Those lead in turn to large values of ARL_1 by (3.2.2).
In practice, the parameters of the chart are therefore calibrated to reach the maximal detection power for a fixed rate of false positives.
Note that for small values of α, $\sqrt{1-\alpha} \approx 1$ and thus $ARL_0 \approx \sigma^0_{RL}$. Hence, multiple runs are needed to correctly approximate the true ARL_0 values.

3.2.2 Shewhart chart

Different methods can be used to monitor a process. Among those, the most famous and widely-used are the control charts. A control chart is at its basis a powerful graphical tool for statistical process control (SPC). Each point of the chart corresponds to the value of a statistic, which is computed on samples collected from the process. The x-axis of the chart corresponds thus to the sample number and its y-axis is the value of the statistic. When this statistic exceeds certain values, called the control limits of the chart (which can have upper and lower values), the process is assumed to be out-of-control. The charting statistic must thus contain as much information about the (IC) data as possible, to rapidly detect any shift in the distribution of the process. We focus in the following on detecting a shift of size δ in the mean of a process. To this end, several control charts based on different statistics have been proposed in the literature. The first and the simple one is the Shewhart (Shewhart, 1931) chart, described in the following.

Let us assume that we want to monitor an univariate i.i.d. normal process. When the process is in-control, its mean and variance are equal to μ_0 and σ^2 (known). The distribution of the process follows thus a $\mathcal{N}(\mu_0, \sigma^2)$. When the process is out-of-control, its mean shifts to $\mu_1 = \mu_0 + \delta$. The process follows then a $\mathcal{N}(\mu_1, \sigma^2)$.

To detect this mean shift, the Shewhart chart applies an hypothesis test at each time point t:

$$H_0 : \mu_t = \mu_0 \qquad vs \qquad H_1 : \mu_t = \mu_1, \tag{3.2.3}$$

where μ_t is the true process mean at time t.
To perform this test, a sample of size m is collected at each time $\{X_{t1}, ..., X_{tm}\}$. It allows us to compute the observed mean of the process at every time-point:

$$\overline{X}_t = \frac{1}{m} \sum_{i=1}^{m} X_{ti}.$$

As known from elementary statistics, an appropriate statistic for the test in (3.2.3) is:

$$Z_t = \frac{\overline{X}_t - \mu_0}{\sigma/\sqrt{m}} \underset{H_0}{\sim} \mathcal{N}(0,1). \tag{3.2.4}$$

If the test is performed at a level α, the null hypothesis is rejected if the observed value of Z_t, denoted $Z_t^\star$, satisfies $|Z_t^\star| > z_{1-\alpha/2}$, where $z_{1-\alpha/2}$ represents the $(1-\alpha/2)$-th quantile of the standard normal distribution.
Hence, we conclude that the process is OC if:

$$\overline{X}_t > \mu_0 + z_{1-\alpha/2}\frac{\sigma}{\sqrt{m}} \quad or \quad \overline{X}_t < \mu_0 - z_{1-\alpha/2}\frac{\sigma}{\sqrt{m}}, \tag{3.2.5}$$

where $\mu_0 \pm z_{1-\alpha/2}\frac{\sigma}{\sqrt{m}}$ are the control limits of the chart.

Most of the time, μ_0 and σ are unknown. In practice, they may then be estimated on the data with the following formula:

$$\begin{aligned} \hat{\mu}_0 &= \frac{1}{n}\sum_{t=1}^{n} \overline{X}_t = \frac{1}{nm}\sum_{t=1}^{n}\sum_{i=1}^{m} X_{ti} \\ \hat{\sigma} &= \frac{1}{n}\sum_{t=1}^{n} s_t, \quad \text{where} \quad s_t = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(X_{ti} - \overline{X}_t)^2}. \end{aligned} \tag{3.2.6}$$

Note that the estimator $\hat{\sigma}$ is biased but corrections are proposed in (Kenney and Keeping, 1951, pp 65-68).

3.2.3 CUSUM chart

As can be seen in (3.2.5), the decision rules of the Shewhart chart depend only on the sample collected at time t but not on past samples. This chart is thus similar to a procedure which applies a hypothesis test at each time, regardless of the history of the process. When a shift occurs however, all data recorded thereafter contain

useful information about the deviation. The Shewhart chart, which does not use these information, is then ineffective to detect small and persistent shifts in the monitored process. To correct this effect, the cumulative sum (CUSUM) (Page, 1961) control chart was proposed few years later. This chart is described in the following in two different forms which are equivalent: the decision interval and the V-mask.
Contrarily to the Shewhart, the CUSUM chart takes into account the data history to make a decision about the status of the process. It is thus more effective to detect small and persisting deviations. Hence, in practice, the Shewhart chart is often used in the Phase I SPC to roughly identify a subset of IC data. When the IC parameters are estimated, the CUSUM chart is then preferably applied to the data in phase II SPC for the actual monitoring. Note that the Shewhart chart should be applied on batches of data. Although the CUSUM chart can also be applied on batches, it will be presented for individual data, which allows a more rapid identification of a deviation (i.e. fewer observations are needed to detect a shift).

As before, we aim at monitoring the mean of an univariate i.i.d. normal process. This mean is assumed to be equal to μ_0 when the process is IC. It shifts to μ_1 when the process becomes OC, with a shift size equal to $\delta = \mu_1 - \mu_0$. To take into account the history of the process, a simple charting statistic may be defined as:

$$C_n = \sum_{i=1}^{n} (X_i - \mu_0). \tag{3.2.7}$$

This expression is equivalent to

$$C_n = C_{n-1} + (X_n - \mu_0), \tag{3.2.8}$$

where $C_0 = 0$. C_n is thus the cumulative sum of the deviation of the data $X_1, ..., X_n$ from μ_0. It is also the sum of n i.i.d. random variables. If the process if IC, C_n follows thus a normal distribution:

$$C_n \sim \mathcal{N}(0, n\sigma^2). \tag{3.2.9}$$

Assuming that a shift δ occurs at time τ, $1 \leq \tau \leq n$, the distribution of C_n, for $n \geq \tau$, changes to:

$$C_n \sim \mathcal{N}((n - \tau + 1)\delta, n\sigma^2). \tag{3.2.10}$$

It follows from (3.2.9) and (3.2.10) that the mean of C_n is zero when the process is IC and that it starts to linearly change with a slope δ when a shift occurs. C_n would thus be a good charting statistic candidate if its variance was not also changing with n. This effect indeed complicates the identification of a linear trend in the mean of C_n.

Decision interval form

To solve the variance problem, Page (Page, 1961) introduced another statistic to detect upward shifts:

$$P_n^+ = P_{n-1}^+ + (X_n - \mu_0) - k, \tag{3.2.11}$$

where $P_0^+ = 0$ and $k > 0$ is the allowance parameter. This chart then triggers an alert if

$$P_n^+ - \min_{0 \le m < n} P_m^+ > L^+, \tag{3.2.12}$$

where L^+ is another parameter, called the control limit of the chart.

The scheme defined in (3.2.11) and (3.2.12) is equivalent to the CUSUM chart based on the following statistic:

$$C_n^+ = max(0, C_{n-1}^+ + (X_n - \mu_0) - k), \tag{3.2.13}$$

for $C_0^+ = 0$. This chart identifies an upward deviation and gives an alert when

$$C_n^+ > L^+. \tag{3.2.14}$$

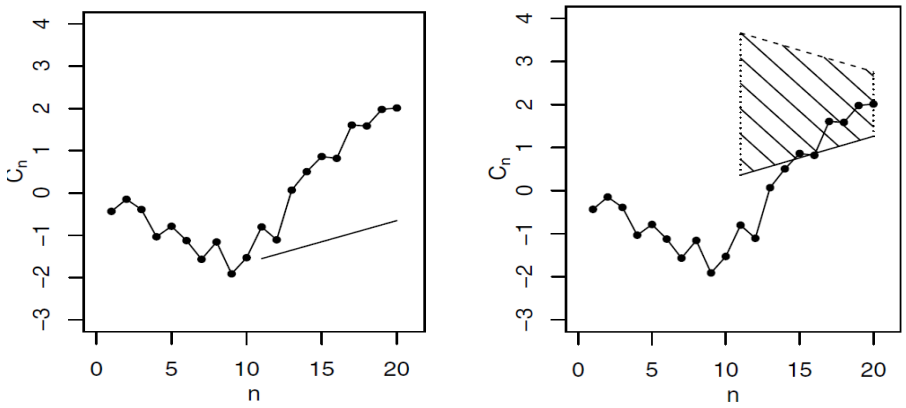
(3.2.13)-(3.2.14) corresponds to the modern decision interval form of the CUSUM chart to detect an upward shift. In this expression, the allowance constant (k) and the control limit (L^+) were introduced to take into account the increasing variance of C_n. This chart has also an explicit restarting mechanism (reset to zero), which controls its memory. The restarting occurs when $C_{n-1}^+ + (X_n - \mu_0) < k$, i.e. in cases where an upward chart is indeed very unlikely.
In practice, the data can also experience a downward shift of size $-\delta$. This negative shift can be detected by a downward CUSUM chart based on the following statistic:

$$C_n^- = min(0, C_{n-1}^+ + (X_n - \mu_0) + k), \tag{3.2.15}$$

where $C_0^- = 0$. This chart triggers an alert if:

$$C_n^- < L^-. \tag{3.2.16}$$

Both charts defined in (3.2.13)-(3.2.14) and in (3.2.15)-(3.2.16) are one-sided. To detect simultaneously upward and downward deviations, a two-sided version of the CUSUM chart can be defined by combining these two one-sided charts. The two-sided CUSUM chart detects then a (positive or negative) deviation if (3.2.14) or (3.2.16) holds. As the distribution of the data is symmetric (they are by assumption normally distributed), we have $L^+ = -L^- = L$.



a: Half-line from $\tau = 11$ b: V-mask

Figure 3.1: Values of the statistic C_n obtained from a simulated process with $\mu_0 = 0$, $\sigma^2 = 1$ and $\mu_1 = 0.2$. A shift of size $\delta = 0.2$ has occurred at time $\tau = 11$ in the data. (a) For $n > \tau$, the values of C_n are all above the half-line starting at $(\tau, C_\tau - L)$ with a slope k, for $k = \delta/2$ and $L = 3/\sqrt{5}$. (b) The V-mask form of the CUSUM chart is here applied to the same process. With this method, a deviation is detected if some values of C_n fall outside of the V-mask represented in the shaded area, as it is the case here at time $t = 15$. These figures have bee taken from (Qiu, 2013, Section 4.2).

V-mask form

The decision-interval form of the CUSUM chart is presented in (3.2.13)-(3.2.14) and (3.2.15)-(3.2.16). We present here another form of the chart which is equivalent to the decision-interval: the V-mask form. This second form is more graphical and often gives a better understanding of the different parameters of the chart (k and L).
Let assume one again that a upward shift of size δ occurs at time τ, $1 \leq \tau \leq n$, in the mean of the normal process defined earlier. As stated in (3.2.9), the statistic C_n follows a normal distribution of mean equal to zero and variance equal to $n\sigma^2$ before τ. The mean of the process changes then linearly from τ with a slope δ. Since the variance of C_n also changes with n, the detection of a linear trend in the mean is complicated. The V-mask form of the CUSUM chart was then designed to overcome this difficulty. It is explained below.

If an upward shift occurs at time τ, we expect that, for $n > \tau$, all values of C_n would be superior to the half-line starting at $(\tau, C_\tau - L)$ with a slope k (for $L > 0$ and $0 < k < \delta$). The parameters L and k have been introduced, instead of simply constructing a slope δ, to take into account the increasing variance of C_n. This is

illustrated in Figure 3.1a for a simulated process which experiences a shift of size $\delta = 0.2$ at time $\tau = 11$.
In practice, the time of the shift τ is unknown. Hence, we rather construct a half-line starting at $(n, C_n - L)$ and progressing backward with a slope equal to k. If a shift has occurred in the past, some values of C_n should then fall below the line. Similarly, another half-line starting at $(n, C_n + L)$ and progressing backward with a slope $-k$ can be constructed to detect downward shift of size $-\delta$. Both half-lines can thus be constructed at time n to detect if a negative or positive mean shift has occurred in past data. If it is the case, some values of C_n would then fall outside of the half-lines, as shown in Figure 3.1b. As can be seen, those lines form a horizontal V-mask, hence the name V-mask form of the CUSUM chart.
The time and the size of the shift are usually unknown. They can be estimated with the V-mask. If several points fall outside of the mask, $\hat{\tau}$ is chosen as the point which is the farther away from the mask, i.e. $\hat{\tau} = 11$ in Figure 3.1b. An estimation of the size of the shift can also be obtained with the following formula:

$$\hat{\delta} = \frac{C_n - C_{\hat{\tau}}}{n - \tau}, \tag{3.2.17}$$

since the mean of C_n increases linearly with a slope δ from the occurrence of a shift.
As can be seen in the figures, the distance between the last value of C_n and the half-lines is equal to L. The aperture of the V-mask is thus equal to $2L$, whereas the opening angle of the mask is determined by the parameter k. Hence, small values of k and L lead to a quicker detection of the shifts. Note that a decrease in L and/or k also gives rise to an increase in the rate of false positives of the chart, i.e. a decrease of ARL_0.

Design

As previously seen, the CUSUM chart has two interlinked parameters, k and L, which affect its performances. Those can be evaluated by the ARL: the IC ARL, ARL_0, controls the rate of false positive of the chart whereas the OC ARL, ARL_1, measures its detection power. In practice, the values of k and L are calibrated to reach a maximal detection power for a pre-specified value of ARL_0, as explained below.
A target shift size, δ_{tgt}, is first selected from scientific knowledge of the process. It could also be estimated from the OC process distribution, as explained in Appendix 3.7.1. A typical target shift size should be small enough to not be detected easily by a visual inspection of the data and at the same time be high enough to have meaningful impacts on the process. The allowance parameter is then specified to $k = \delta_{tgt}/2$. This relation is optimal for i.i.d. normal data (Moustakides, 1986). Studies in Deketelaere (2020) show that this relation still is valid for many process distributions however as long as they do not differ too much from the normal. This

relation does not hold typically for e.g. asymmetric distributions. A pre-specified value of ARL_0, denoted $ARL_0^\star$, is also chosen. Typical values are 100 ($\alpha = 0.01$), 200 ($\alpha = 0.005$) or 500 ($\alpha = 0.002$).

Algorithm 1: Pseudo-code for adjusting the value of the control limit L

```
/* Adjust the control limit of the chart for i.i.i.  normal data        */
```

Select values for:
$[L_{low}, L_{up}]$, the interval where L is searched on
$ARL_0^\star$, the desired value of the in-control ARL
ρ, the accuracy to reach ARL_0
δ_{tgt}, the target shift size
$k = \delta_{tgt}/2$

while $|ARL_0 - ARL_0^\star| > \rho$ **do**
- $L = \frac{L_{low}+L_{up}}{2}$
- **for** *b in (1, B)* **do**
  - sample data from the (IC) process distribution ($\mathcal{N}(\mu_0, \sigma^2)$)
  - compute the test statistics (C^+ and C^-) on the simulated data
  - **if** *the chart gives an alert* ($C^+ > L$ *or* $C^- < -L$) **then**
    - $RL[b]$ = time of alert
  - **else**
    - $RL[b]$ = large number
- $ARL_0 = \text{mean}(RL)$
- update L_{up} or L_{low} $\begin{cases} L_{up} = (L_{low} + L_{up})/2,\ L_{low} = L_{low} & \text{if} \quad ARL_0^\star > ARL_0 \\ L_{low} = (L_{low} + L_{up})/2,\ L_{up} = L_{up} & \text{if} \quad ARL_0^\star < ARL_0 \end{cases}$

$L = \frac{L_{low}+L_{up}}{2}$

The control limit L is then adjusted by a searching algorithm (Qiu, 2013, Section 4.2.2) until the pre-specified rate of false positives is reached at the desired accuracy. This procedure uses a bisection method shown in Algorithm 1, where L is searched in the interval $[L_{low}, L_{up}]$ which is cut in two at each iteration. Inside each iteration, data are sampled from the IC process distribution (here the normal) and the run lengths of the method are computed on a large number of runs (B). The actual ARL_0 is then evaluated over the runs. Typically, thousand or more runs are needed to accurately estimate ARL_0. If the actual ARL_0 is inferior (resp. superior) to the pre-specified $ARL_0^\star$, the control limit of the chart is then increased (resp. decreased). This algorithm is iterated until the actual ARL_0 reaches $ARL_0^\star$

at the desired accuracy.
Note that this algorithm only works if L_{low} is sufficiently small to give $ARL_0 < ARL_0^\star$ and if L_{up} is high enough to give $ARL_0 > ARL_0^\star$.

The control charts previously seen can also be used to monitor the mean of processes which follow another (known) distribution than the normal. In this case, Algorithm 1 can still be used for adjusting the limits of the CUSUM chart. The only thing that should be changed in the algorithm is the simulation of the data. They should be sampled from the specific IC process distribution instead of the normal. The relation $k = \delta/2$ can however be suboptimal if the process distribution is too different from the normal. In this case, an optimal value for k can be computed as follows. For different values of k, the control limits of the chart may be calculated using the Algorithm 1, to reach a common rate of false positives. The combination (k-L) that yields the better detection power (see Algorithm 2 below) can then be selected.
Note that Algorithm 1 with appropriate charting statistics can also be used to calibrate the limits of the Shewhart chart to monitor non-normal data.

Detection power

Algorithm 2: Pseudo-code for evaluating the ARL_1 values of the chart

```
/* Evaluate the detection power of the CUSUM chart                    */
```
Select value for:
δ_{tgt}, the shift size that we aim to detect

Compute the control limit L using Algorithm 1 for $k = \delta_{tgt}/2$
for *b in (1, B)* **do**
    // sample data from the IC process distribution
    data = sampling(IC distribution)
    // add a shift of size δ_{tgt} on those data
    data + δ_{tgt}
    compute the chart statistics C^+ and C^- on the simulated data
    if *the chart gives an alert ($C^+ > L$ or $C^- < -L$)* **then**
        $RL_1[b]$ = time of alert
    else
        $RL_1[b]$ = large number
ARL_1 = mean(RL_1)

The detection power of the charts can be evaluated using the concept of ARL_1 as proposed in Algorithm 2. With this method, the average time needed to detect a shift of size δ_{tgt} is approximated with Monte-Carlo simulations.

3.2.4 Block bootstrap

Previously, we see how the control charts can be calibrated for i.i.d. normal data. In practice however, the distribution of the process is often unknown and the data can be autocorrelated. In these cases, the control charts can still be applied to the data but they should be calibrated using a method that takes into account the actual distribution of the process and its autocorrelation. For this purpose, a popular solution is the bootstrap and its time-series version: the block bootstrap.

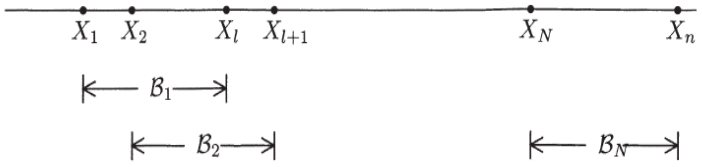
Bootstrap

The bootstrap is a non-parametric procedure that samples data with repetition from the empirical distribution of the process. In general, it is used to approximate the properties, such as the bias, variance or confidence intervals, of an estimator $\hat{\theta}$ of an unknown parameter θ based on a sample of data. Let $\hat{\theta}^*$ denotes the bootstrap version of $\hat{\theta}$. The bias of $\hat{\theta}^*$, which writes as $E(\hat{\theta}^*) - \hat{\theta}$, can for instance be used to approximate the bias of $\hat{\theta}$, $E(\hat{\theta}) - \theta$.
More specifically, the bootstrap allows us to generate here "new" data with similar distribution as those of the observations, to calibrate the control charts. With this method, the control limit of the charts can be computed using a procedure that is similar to Algorithm 1. The only step that should be modified in the algorithm is that the data are no longer sampled from a known distribution but are resampled with repetition from the IC observations.

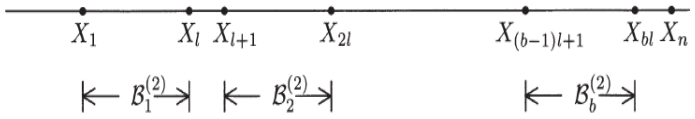
Block bootstrap methods

When the data are autocorrelated, the simple bootstrap method cannot be used to calibrate the charts since it samples data one by one. A popular solution to reproduce the autocorrelation of the process is the block bootstrap (BB). With this method, the data are randomly sampled with repetitions from the IC process distribution by blocks of consecutive observations. As many blocks as needed may be sampled from the data to reach a specific length. A "new" series of desired length may then be obtained by concatenating those blocks. This procedure preserves thus the correlation of the data inside the blocks.
Note that an alternative method to the block bootstrap is to fit the data by a parametric time-series model such as a (S)ARIMA (Brockwell and Davis, 1991) and then to apply the control chart to the uncorrelated residuals. This approach is

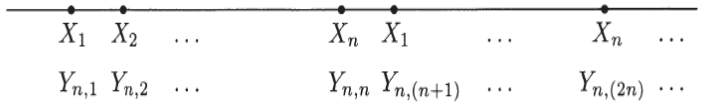
however time consuming and does not work well if the autocorrelation of the data is not well-represented by those models (which is typically the case when the data have complex autocorrelation structures). Hence, we only present the BB, which is more flexible, in the following.
Several versions of the block bootstrap have been developed in the literature: the moving block bootstrap (MBB) (Künsch, 1989; Liu and Singh, 1992), the non-overlapping block bootstrap (NBB) (Carlstein, 1986), the circular block bootstrap (CBB) (Politis and Romano, 1992) or the stationary block bootstrap (SBB) (Politis and Romano, 1994). Those methods mainly differ in the way they construct the blocks ; the number of blocks and the optimal block length are thus different in each version.



a: Blocks of MBB



b: Blocks of NBB



c: Periodic extension of the data in CBB

Figure 3.2: (a) The set $\{\mathcal{B}_1, ..., \mathcal{B}_N\}$ of overlapping blocks of the MBB method. (b) The set $\{\mathcal{B}_1^{(2)}, ..., \mathcal{B}_N^{(2)}\}$ of non-overlapping blocks of the NBB procedure. (c) Periodic extension of the data defined in CBB. These figures have bee taken from (Lahiri, 2003, Section 2).

The MBB constructs for example overlapping blocks as shown in Figure 3.2a. With $X = \{X_i, 1 \leq i \leq n\}$ a stationary and autocorrelated process of length n and ℓ

the block length, $\mathcal{B}_i = (X_i, ..., X_{i+\ell+1})$ denotes a block starting at the observation X_i of length ℓ, for $1 \leq i \leq N$ where $N = n - \ell + 1$. On the contrary, the NBB method samples non-overlapping blocks, shown in Figure 3.2b. In this case, $\mathcal{B}_i^{(2)} = (X_{(i-1)\ell+1}, ..., X_{i\ell})$ for $i = 1, ..., b$, where $b \geq 1$ is the largest integer satisfying $\ell b \leq n$.
The two last BB methods (CBB and SBB) were designed to overcome the boundary effect of the MBB, which gives more weight to the observations in the middle of the series than to those at the ends. Indeed, the first and last observations appear in fewer blocks than the other data in MBB. Note that a similar effect is observed in NBB when n is not a multiple of ℓ. To this end, CBB first defines a periodic extension of the data, $\{Y_{n,1}, ..., Y_{n,2n}\}$ illustrated in Figure 3.2c and then partitions the observations into (non-overlapping or overlapping) blocks. Hence, each observation appears exactly ℓ times in the blocks.
The SBB on the other hand samples blocks of random length using the following procedure. A first observation X_1^* is randomly sampled from the data. Then, a binary experiment with probability of success $p \in (0, 1)$, where $\mathbb{E}(\ell) = 1/p$ is realized. If the result is "success", then a second observation X_2^* is randomly sampled from the observations. Otherwise X_2^* is chosen as the next observation $X_2^* = X_{1+1}^*$. This method generates then blocks of random length and gives the same weight to all data. For more information about the different BB methods, we refer to the complete work of Lahiri (2003).
Those four methods have been compared theoretically in Lahiri (1999), which shows that BB methods using non-overlapping blocks and random block lengths are more variable than those based on overlapping blocks and constant lengths. Hence, the MBB and the CBB (using overlapping blocks) methods are often preferred for practical applications.

The previously-described BB methods randomly sample blocks of observations and create new time-series by concatenating them. The autocorrelation of the series is thus preserved inside the blocks but is destroyed outside of them. Hence, abrupt changes often appear between the blocks, which may not correspond to the behaviour of the original series if it has long correlations (i.e. correlation until high lags). To solve this problem, the matching block bootstrap (MABB)(Carlstein et al., 1998) has been designed. It creates a sequence of blocks using a transition rule, which selects blocks that are a priori more likely to be close to each other. The blocks are thus constructed using one of the previously-described BB methods and they are later sampled using a Markov chain that favours blocks whose ends match the end of the previous block (rank matching).
The MABB method can be summarized as follows. Let $B_i = \{X_{i,1}, ..., X_{i,l}\}$ denote a block of length l and $X_{i,j}$ be the j-th observation in the i-th block. Using e.g. overlapping blocks, there are thus $b = n - l + 1$ different blocks. We denote by R_i the rank of the end of the block B_i, which corresponds to the rank of the last observation $X_{i,l}$ among $X_{1,l}, ..., X_{b,l}$. A first block B_1 may be sampled from

the collection of blocks. Another block j may then be randomly selected from the $2k+1$ blocks ranked between R_1-k and R_1+k, where k is a small positive integer. The next block appended to the series is then the block that followed the block j in the original series. The process may be iterated until the series reaches a desired length.
With this method, shorter blocks may be chosen than with the previous BB methods since the long-term correlations are well-handled by the rank matching. The MABB method may however be inappropriate for time-series which experience much noise, since the MABB may create series that are smoother than the original one. The MABB method should thus reserved for time series with long autocorrelations and high signal to noise ratio.

Block length

All BB methods previously seen rely on a important parameter, the block length (or the expected block length for SBB). The choice of this parameter is similar to those of the bandwidth in non-parametric density estimation. Apart from giving the rate at which it grows with the sample size (n) however, they are few practical results in the literature to help us specify the block length. Moreover, many of them are proposed in a context where the bootstrap is used to approximate the properties of an estimator ($\hat{\theta}$) of an unknown parameter (θ). Hence, the formula that are proposed depend on this estimator and the final aim for which the bootstrap is used (estimating the bias, variance or confidence intervals of the estimator). Hall et al. (1995) proposes for example an empirical method for choosing the block length that works as follows. Let $\hat{\psi}$ denotes the bias, the variance or the distribution of $\hat{\theta}^*$, the bootstrap version of $\hat{\theta}$. The authors compute $\hat{\psi}$ for different values of the block length on subseries of length $m < n$. Then, they select the optimal block length in a subseries ($\hat{\ell}_m$) as those which yields the lowest average of the squares of the differences $|\hat{\psi}_i - \hat{\psi}|$, where $\hat{\psi}$ is computed on the complete series of length n and $\hat{\psi}_i$ in a subseries. Finally, they obtain the block length for the full series, $\hat{\ell}_n$, using $\hat{\ell}_n = (n/m)^{1/k}\hat{\ell}_m$, where $k = 3$ for estimating the bias or variance, $k = 4$ for estimating a one-sided distribution and $k = 5$ for estimating the two-sided distribution of $\hat{\theta}^*$. This method can lead to poor results however, when $\hat{\theta}$ is highly non-linear. Moreover, those rules or methods do not correspond to our problem, whose aim is simply to generate data with the same properties as the observations.

Hence, we propose another method for selecting the block length of a chosen BB method, with the aim to calibrate the limits of the control charts previously described. The procedure is described in Algorithm 3 and works as follows. For each block length over a specified range, the method resamples B series of observations using the chosen BB method. Then, it computes the mean squared error (MSE) of

Algorithm 3: Pseudo-algorithm to estimate an optimal value for the block length

```
/* Compute an appropriate block length                                    */
```

Select values for:
start, a starting value for the block length
stop, a stopping value for the block length
step, a step value for the block length
lag_max, the maximal lag up to which the autocorrelation is evaluated

for ℓ *in (start, stop, step)* **do**
  for *station in (1,* N_{IC}*)* **do**
    for b *in (1,* B*)* **do**
      // sample data from the IC series 'station'
      $boot$ = resample IC data per blocks of length ℓ
      // compute the autocorrelation of the resampled series until lag_max
      $autocorr\_boot[b, lag]$ = autocorrelation($boot$, lag_max)
    // compute the MSE of the autocorrelation in a station for each lag and take the mean over the lags
    $mse\_autocorr\_station[station]$ = mean(MSE($autocorr\_boot$), lag)
  // compute the mean of the autocorrelation over all stations
  $mse\_autocorr[\ell]$ = mean($mse\_autocorr\_station$)
$block\_length$ = knee($mse\_autocorr$)

the empirical mean, standard deviation and autocorrelation at different lags of the resampled series with respect to the original data. Small block lengths appear to represent the variance and the mean of the data properly, i.e. the MSE of the mean and the variance increases with the block length. Reversely, large block lengths better account for the autocorrelation of the data as the MSE of the autocorrelation decreases when the block length increases. The appropriate value for the block length is finally selected as the "knee" (or elbow) (Satopaa et al., 2011) of the curve, i.e. the first value such that the MSE of the autocorrelation becomes stable. This value intuitively corresponds to the smallest length which is able to represent the main part of the autocorrelation of the series.
When the BB method and its optimal block length are selected, the control limits of the charts may then be adjusted as proposed in Algorithm 1. In the algorithm, the data should simply be resampled from the IC process distribution by the block bootstrap instead of being drawn one by one from a known distribution. Hence, the control charts can be calibrated on data with similar distribution and autocorrelation structure as the observations.

3.3 Data

After this overview of SPC methods, we focus in the remaining part of the chapter on developing an effective monitoring for the sunspot numbers.
The dataset that will be studied in the following is first presented at the beginning of the section. Then, the uncertainty model of the sunspot data that was developed in Chapter 2 is introduced again, as a remainder, and is slightly modified. The precise quantity that will be monitored is then presented alongside with its estimating procedure.

3.3.1 Dataset

The period under study here embraces the most recent part of the series and extends from January 1, 1981 (when the sunspot numbers production centre moved from Zürich to Brussels) till December 31, 2019. It ranges from the descending phase of solar cycle (SC) 21 to the minimum between SC 24 and 25[1] and covers thus three complete solar cycles. The data are composed of the number of spots N_s, groups N_g and composite N_c recorded by a network of 278 observing stations across the world. They contain around 85% of missing values in this period (1981-2019). Those are mainly caused by the non-overlapping observing periods of the stations. Indeed, some stations started observing only recently while older stations stopped their activity well before 2019. The stations also contain various percentages of missing values, ranging from 15% to 75%, over their active observing period, mainly due to weather conditions. The dataset contains thus around 622000 observations. Note that we work with data containing less missing values in practice, since the data are smoothed by a moving average, see Section 3.3.3 below. This procedure imputes the smallest gaps in the data. Hence, the data smoothed on 27 days contains around 933000 values whereas those smoothed on a year are composed of 1092000 values.
In the following, we denote by t, $t \in 1, ..., T$, the date-time of the observation and represent the index of the stations by i, $i \in 1, ..., N = 278$, to keep consistency with the notations introduced in the previous chapter.

3.3.2 (New) uncertainty model

The observations have been decomposed into a common solar signal corrupted by three types of errors in (2.3.4). Among them, the short-term error (ϵ_1) is responsible for rapid fluctuations in the observed numbers, which can be removed by an appropriate filtering. Moreover, the error at minima (ϵ_3) only models errors in a

[1] https://en.wikipedia.org/wiki/List_of_solar_cycles

small part of the solar cycle. Our monitoring thus aims at the long-term error ϵ_2, which has the most significant impact on the long-term stability of the stations. As can be seen in Figure 2.10, the mean of ϵ_2, $\hat{\mu}_2(i,t)$, is not aligned with one in all stations. Some stations such as the Observatory of Locarno have an overall higher level, more around 1.5 than 1. Those levels reflect differences of instruments or counting methodologies. They may also be related to the degree of atmospheric transparency above the stations, which depends on their location. The levels are thus not directly related to the quality of the observations. Moreover, they are difficult to change in response of an alert. Those factor will thus not be taken into account in the monitoring. Otherwise, a station with a more accurate telescope than those of the other stations of the network could be systematically tagged as out-of-control (i.e. deviating).

To that end, we add an additional term in the model that we developed in Chapter 2, to explicitly untangle those levels (denoted by h) from the long-term errors. With $Y_i(t)$ representing either the number of spots, groups or composite observed in station i at time t, the more complete version of the model writes as:

$$Y_i(t) = \begin{cases} (\epsilon_1(i,t) + \epsilon_2(i,t) + h(i,t))s(t) & \text{if } s(t) > 0 \\ \epsilon_3(i,t) & \text{if } s(t) = 0. \end{cases} \tag{3.3.1}$$

This model is thus composed of the following quantities:

- $s(t)$ is a latent variable representing the actual number of spots, groups or composite of the Sun. This latent variable cannot be directly observed but its mean will be estimated based on the observations of the network and later used as a proxy for $s(t)$.

- ϵ_1 is a short-term error, which is prevailing at scales that are lower than 27 days (i.e. one solar rotation). It typically represents counting errors. We assume that $\mathbb{E}(\epsilon_1(i,t)) = 0$ where $\mathbb{E}$ denotes the expectation sign.

- ϵ_2 denotes a long-term error, which corresponds to scales between 27 days and eleven year (one solar cycle). We are interested in estimating and monitoring its mean, denoted by $\mu_2(i,t)$, which represents the bias of the stations.

- h is defined at time-scales equal to or longer than eleven years. It corresponds to the background level of the stations (accounting e.g. for differences of instruments or counting methodologies of the stations). For identification purpose, we assume that $\mathbb{E}(\epsilon_2(i,t) + h(i,t)) = 1$.

- ϵ_3 is an additive error capturing effects like short-duration sunspots during solar minima, i.e. periods of minimal activity in the eleven-year solar cycle.

The errors ϵ_1, ϵ_2 and h vary on different time-scales and are multiplicative quantities since an observer typically makes larger errors when $s(t)$ is higher (Chang and Oh, 2012). The random variables ϵ_1, ϵ_2, ϵ_3 and h are assumed to be continuous and ϵ_1, ϵ_2, ϵ_3, h and $s(t)$ to be jointly independent. Note that ϵ_1, ϵ_2 and ϵ_3 would be equal to zero and h be equal to one for a station that would be — in absence of any measurement errors —perfectly aligned with the solar signal.

3.3.3 Long-term bias

We are thus interested in estimating and monitoring the mean of the long-term error *without* levels. This quantity is assumed to better represent the bias of the stations. Hence, it will denoted by $\mu_2(i,t)$, the same notation that was used in Chapter 2 for designating the mean of the long-term error *with* levels. In the following, any further mention of the bias, the mean of the long-term error or the $\mu_2(i,t)$ will refer to this new definition (i.e. the mean of the long-term error *without* levels). To this end, we isolate the long-term error from the other components of the model following the approach described in Section 2.6.3. This step-wise procedure is described below for the more complete version of the model introduced in (3.3.1). It also serves as a remainder.

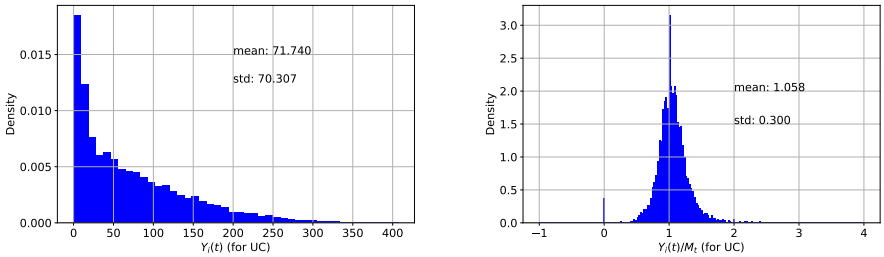
We first divide the observations by scaling factors to roughly compensate for different observing conditions: $Z_i(t) = \frac{Y_i(t)}{\kappa_i(t)}$. These piece-wise constant scaling factors $\kappa_i(t)$ are computed as the slope of the ordinary least-squares regression between the observations of the stations and the median of the observations ($\underset{1\leq i\leq N}{\text{med}} Y_i(t)$) on periods of 8 months for N_s, 14 months for N_g and 10 months for N_c. These values were selected by a statistical-driven study based on the Kruskal-Wallis test (Kruskal and Wallis, 1952), which is completely described in Section 2.8.1.
Afterwards, we compute M_t, a robust proxy for $s(t)$ based on the median of the rescaled observations:

$$M_t = \underset{1\leq i\leq N}{\text{med}} Z_i(t). \tag{3.3.2}$$

Motivated from the model in (3.3.1), the observations Y are then divided by M_t to remove the main influence of the solar signal. They are also smoothed by a moving-average (MA) filter, represented by a $\star$ in the following equation. This smoothing process untangles ϵ_2 and h, from the short-term error ϵ_1:

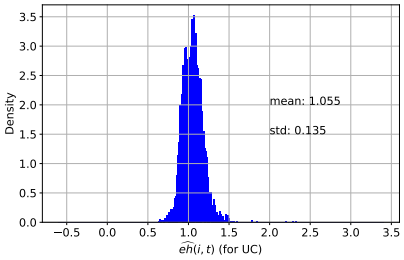
$$\hat{\epsilon}_2(i,t) + \hat{h}(i,t) \equiv \widehat{eh}(i,t) = \left(\frac{Y_i(t)}{M_t}\right)^{\star} \quad \text{when } M_t > 0. \tag{3.3.3}$$

To analyse the various deviations of the data, different MA-filter window lengths may be used in (3.3.3). Those need however to be superior than or equal to 27

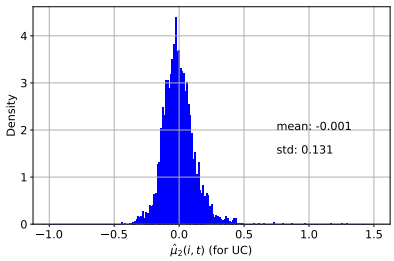


a: $Y_i(t)$ in UC

b: $Y_i(t)/M_t$ in UC



c: $\widehat{eh}(i,t)$ in UC



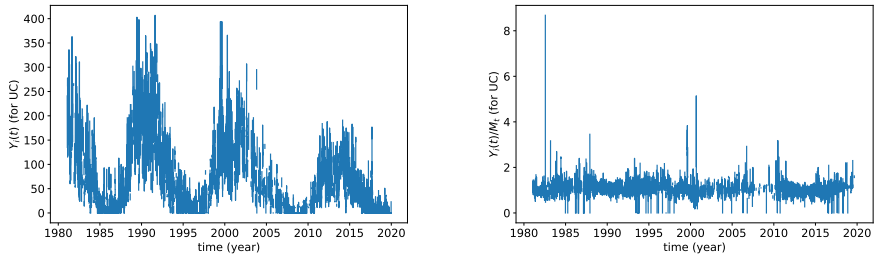
d: $\hat{\mu}_2(i,t)$ in UC

Figure 3.3: Different steps to estimate the long-term bias $\hat{\mu}_2(i,t)$ in the station Uccle (UC). (a) Histogram of the observations $Y_i(t)$ for N_c. (b) Histogram of the ratio $Y_i(t)/M_t$. This step removes the main part of the solar signal from the data. (c) Histogram of $\widehat{eh}(i,t)$. At this stage, the short-term error ϵ_1 is also removed from the observations by applying a MA filter of length equal to 27 days. (d) Histogram of the long-term bias, after removing the level h of the station by applying a MA filter of length equal to 11 years.

days to overcome the effects of the short-term regime, as stated in Section 2.6.3. We focus in the following on two different scales. The low-frequency shifts such as persisting drifts are first studied at a yearly scale (i.e. with a window length of 365 days). Then, a window of length equal to 27 days will also be used to examine the high-frequency deviations such as sudden jumps.
Finally, the levels of the stations are separated from the long-term error by applying once again a MA smoothing process denoted by $\star\star$. An estimator for the mean of the long-term error μ_2, which is used as proxy for ϵ_2, is then given by:

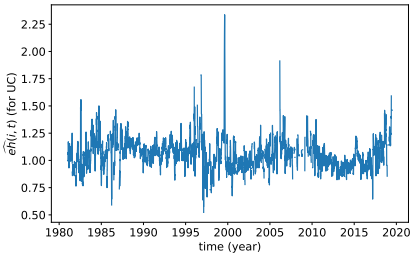
$$\hat{\mu}_2(i,t) = \widehat{eh}(i,t) - \widehat{eh}^{\star\star}(i,t), \tag{3.3.4}$$

where the MA-filter window length should be larger than those of (3.3.3). It is selected here at eleven years (one solar cycle), a physical value that is larger than the time-scales of the long-term error ϵ_2 considered here. It also seems appropriate

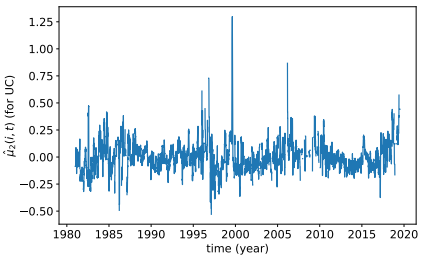


a: $Y_i(t)$ in UC

b: $Y_i(t)/M_t$ in UC



c: $\widehat{eh}(i,t)$ in UC



d: $\hat{\mu}_2(i,t)$ in UC

Figure 3.4: Different steps to estimate the long-term bias $\hat{\mu}_2(i,t)$ in the station Uccle (UC). (a) The observations $Y_i(t)$ as a function of time for N_c. (b) The ratio $Y_i(t)/M_t$ as a function of time. This step removes the main part of the solar signal from the data. (c) $\widehat{eh}(i,t)$ as a function of time. At this stage, the short-term error ϵ_1 is also removed from the observations by applying a MA filter of length equal to 27 days. (d) The long-term bias as a function of time, after removing the level h by applying a MA filter of length equal to 11 years.

since the location of the stations or their telescope are unlikely to change much over time. Since we removed the solar signal (s) from the long-term error, we assume that the ϵ_2s are independent across the stations. The main factors that impact those errors (e.g. the location of the station, the instrument or the counting methodology) are indeed intrinsic to each station.

Figures 3.3 and 3.4 represent the different stages of the estimation of the long-term bias, $\hat{\mu}_2(i,t)$. Figure 3.3 shows the histograms of all intermediate quantities and the $\hat{\mu}_2(i,t)$ whereas Figure 3.4 represents those quantities as function of time. Those results are shown for the station Uccle (UC) in Belgium, which is a typical non-deviating (in-control) station. They illustrate the computation process of the long-term bias. Those $\hat{\mu}_2$s smoothed on 27 and 365 days are also represented in

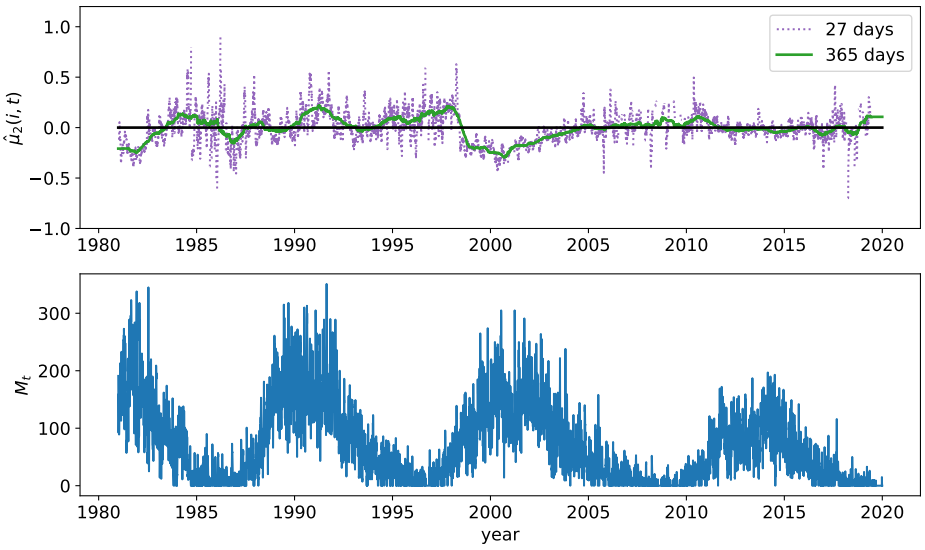


Figure 3.5: Long-term bias, $\hat{\mu}_2(i,t)$ for N_c, in the Kanzelhöhe Observatory (Austria) over its observing period. The $\hat{\mu}_2$s are smoothed on 27 days (dotted line) to allow the detection of high-frequency shifts and smoothed on 365 days (plain line) to emphasize the low-frequency deviations. M_t is also represented in the lower plot as an estimation of the actual N_c. This figure clearly shows the eleven-year solar cycle of the data.

Figure 3.5 for another station, the Kanzelhöhe Observatory in Austria.

3.4 Ingredients of the method

In this section, the complete monitoring procedure depicted in Figure 3.6 is explained. It is intentionally presented in a generic framework to allow the application of the method on the number of spots N_s, groups N_g as well as composites N_c. There are three phases: (I) estimation of the in-control (IC) parameters of the data, (II) construction and use of the monitoring procedure and (III) identification of out-of-control patterns.
The Phase I contains two steps. At first a subset of stations is selected from the panel, which follows closely the median signal, M_t defined in (3.3.2). This pool of stations is then used as a proxy for IC series in the nomenclature of Qiu and Xiang (2014). They are used to determine the IC patterns (mean and variance) of the data and to provide the basis of the block-bootstrap procedure in Phase II. After standardising all series by the IC patterns, the CUSUM control chart is

calibrated in phase II by a block bootstrap procedure from the pool of IC series. The scheme is then applied to the data for the monitoring. In Phase III, support vector machine (SVM) procedures predict the shifts size and shape on sub-series detected as out-of-control by the CUSUM, for easier problem diagnostic.

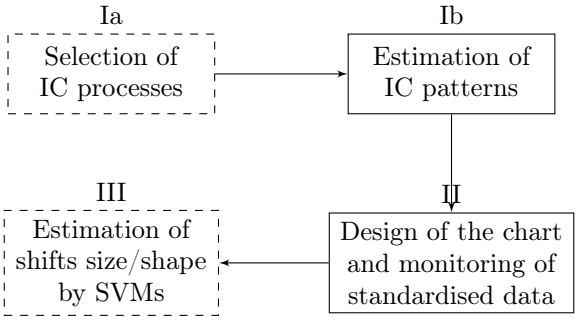


Figure 3.6: Pipeline of the procedure. It is in particular in the dashed blocks that we contribute with new ingredients of the method.

3.4.1 Phase I: Estimation of the IC longitudinal patterns

In this phase I, we automatically construct a subset of IC stations from the panel and estimate the IC patterns of the data.

Phase Ia: Selection of the IC processes

In a first stage, we need a subset (pool) of stations whose observations follow closely the median signal M_t. In order to find them, we calculate a stability criterion on each station. This criterion is based on a robust version of the mean squared error (MSE) of $\hat{\mu}_2$:

$$STB(i) = \underset{1\leq t\leq T}{\text{med}}[\hat{\mu}_2(i,t)]^2 + \underset{1\leq t\leq T}{\text{iqr}}\,\hat{\mu}_2(i,t), \tag{3.4.1}$$

where $\underset{1\leq t\leq T}{\text{iqr}}\,\hat{\mu}_2(i,t)$ and $\underset{1\leq t\leq T}{\text{med}}\,\hat{\mu}_2(i,t)$ denote respectively the interquartile range (IQR) and the median of the $\hat{\mu}_2(i,t)$ over the time for a given station i.
Note that the long-term biases of the stations, $\hat{\mu}_2$s, are thus compared in (3.4.1) to those of a hypothetical ideal station, which is perfectly aligned with the solar signal, approximated by M_t of (3.3.2). This ideal station has no long-term error, hence its $\hat{\mu}_2 = 0$. An observing station should therefore be aligned most of time with M_t and have a small variance to obtain a small STB value.
Using the STB values, we can then cluster the stations in two groups and choose

the cluster with the lowest STB values to form what we call the pool of IC stations. For this purpose, we use the k-means clustering (Lloyd, 1957; MacQueen, 1967). As the two clusters can be highly unbalanced (if e.g. a station is strongly deviating with respect to the network), the clustering in two groups is performed recursively until the smallest cluster contains at least 25% of the stations. Note that we cluster the STB criterion (one value per station) instead of the whole series since the stations have different number of observations, some of them being only composed of few values. Since we cluster 1D data, other methods based on the ordering or the distribution of the observations could be used. We prefer to use an automatic and general-purpose procedure instead. The k-means, described later in Appendix 6.9.1, also appears to classify well the stations whose stability was previously studied in Chapter 2.

This pool still contains deviations though. Those will be called *disparities* in the following, to be distinguished from the deviations that are supposed to be actually detected by the method. The *disparities* are expected to be typically smaller and less frequent than the deviations occurring in the stations not comprised in the pool. They should be included in the design of the chart otherwise the scheme would be over-sensitive.
The pool suffers in addition from deviations that are of similar magnitude as those of the out-of-control (OC) processes. To cope with this and preserve the detection power of our scheme, we also apply a Shewhart chart (see Subsection 3.2.2 for more details) with *adaptive* confidence intervals on the data. We remove the IC observations that do not fall into one standard deviation around the cross-sectional mean ($\frac{1}{N}\sum_{i=1}^{N}\hat{\mu}_2(i,t)$). This step removes around 8.8% of the IC observations. Note that we also test the method with two standard deviations instead of one and the results were similar (in this case, we only remove around 0.9% of the IC data). We emphasize that this adaptive Shewhart chart would not be a substitute for our control scheme: it only removes the largest deviations at each time without taking into account the history of the observations. Therefore, contrarily to our method, it cannot detect the small and persistent shifts.

Phase Ib: Estimation of the mean and the variance of the IC series

In Phase Ib, we compute the mean and variance of the $\hat{\mu}_2$ of the pool, which are denoted respectively by $\mu_0(t)$ and $\sigma_0^2(t)$. Those quantities can be estimated by the

empirical mean and variance using nearest neighbours (K-NN) regression method:

$$\begin{aligned}\hat{\mu}_0(t) =& \frac{1}{\Delta(t)} \sum_{t'=t-\Delta(t)/2}^{t+\Delta(t)/2} \frac{1}{N_{IC}} \sum_{i_{ic}=1}^{N_{IC}} \hat{\mu}_2(i_{ic}, t') \quad s.t. \quad K = \Delta(t) N_{IC} \\ \hat{\sigma}_0^2(t) =& \frac{1}{\Delta(t)} \sum_{t'=t-\Delta(t)/2}^{t+\Delta(t)/2} \frac{1}{N_{IC}} \sum_{i_{ic}=1}^{N_{IC}} \left(\hat{\mu}_2(i_{ic}, t') - \hat{\mu}_0(t)\right)^2 \quad s.t. \quad K = \Delta(t) N_{IC},\end{aligned} \tag{3.4.2}$$

where i_{ic} denotes the index of a station of the pool and N_{IC} is the number of stations in the pool. With this method, $\hat{\mu}_0(t)$ and $\hat{\sigma}_0^2(t)$ are estimated in moving windows across the pool but also along time. The temporal window $\Delta(t)$ can thus be adjusted to compensate the missing values of the stations, such that $\hat{\mu}_0(t)$ and $\hat{\sigma}_0^2(t)$ are always computed on the same number (K) of observations.
In practice, the number of stations in the pool, N_{IC}, is fixed by the k-means procedure previously described. The number of nearest neighbours K is then the only parameter that should be selected. Since the data must be standardised by the IC mean and variance (see (3.4.3) below) for appropriate use in the CUSUM chart statistics, K is chosen to obtain the "best" standardisation of the complete panel, in the sense that its empirical mean becomes close to zero and its empirical variance close to one. $\Delta(t)$ will thus be automatically adjusted in time direction, since $\Delta(t) = K/N_{IC}$.

3.4.2 Phase II: Monitoring

We now turn our attention to monitoring the entire panel. As a reminder, we are analysing long-term biases denoted by $\hat{\mu}_2$. Using the IC mean and standard deviation $\hat{\mu}_0(t)$ and $\hat{\sigma}_0(t)$, we standardise the (IC and OC) stations to be able to use common monitoring criteria:

$$\hat{\epsilon}_{\hat{\mu}_2}(i,t) = \frac{\hat{\mu}_2(i,t) - \hat{\mu}_0(t)}{\hat{\sigma}_0(t)}. \tag{3.4.3}$$

Let us now focus on one station (drop the index i). We would like to detect indications of patterns which may relate to problems at the station. This includes persistent or gradual deviations (shifts or trends) and oscillating patterns as they may occur when the station is used by a rotating pool of observers each of whom has their own particular way of working. As explained in Section 3.2.3, a powerful method for detecting small and gradual deviations in SPC context is the CUSUM chart. The two-sided CUSUM chart applied on the residuals writes as:

$$\begin{aligned} C_j^+ &= max(0, C_{j-1}^+ + \hat{\epsilon}_{\hat{\mu}_2}(t) - k) \\ C_j^- &= min(0, C_{j-1}^- + \hat{\epsilon}_{\hat{\mu}_2}(t) + k), \end{aligned} \tag{3.4.4}$$

where $j \geq 1$, $C_0^+ = C_0^- = 0$ and $k > 0$ is the allowance parameter.
This chart gives an alert if $C_j^+ > L^+$ or $C_j^- < L^-$, where L^- and L^+ are the control limits of the chart. Since the distribution of the residuals is almost symmetric, we use $L = L^+ = -L^-$.

High deviations may affect the series. Those lead to high values of the CUSUM statistics which may stay in alert for longer periods than the actual durations of the shifts. Therefore, in case of too high (resp. too low) values, we set the chart to a maximal value $2L$ (resp. $-2h$). Hence $|C_j^+|, |C_j^-| \leq 2L$.

Design of the chart

For practical use, the control limit and the allowance parameter of the CUSUM chart should thus be selected. As it is clear from the nature of the data, the series to be monitored have a considerable degree of autocorrelation, even when they are in control. We therefore need a method for determining the control limit of the chart that takes autocorrelation into account. As explained in Section 3.2.4, the chart can be calibrated using the block bootstrap (BB) method. This procedure constructs a bootstrap reference distribution by resampling blocks of data and thereby preserves the autocorrelation of the series.
As stated in Section 3.2.4, BB methods using non-overlapping blocks and random block lengths are more variable than those based on overlapping blocks and constant lengths. Moreover, the matching BB (MABB) is not suited here since the data have low signal to noise ratio. Hence, we select the moving BB (MBB) to obtain the best performances. Note that using the circular BB (CBB) method instead of the MBB does not change the results since there are enough observations available (i.e. the impact of having the first and last observations appearing in less blocks than the remaining part of the data is negligible).

The control limit of the chart (L) is then adjusted using the MBB method as described in Algorithm 1. In each run, a new series of the data is sampled with repetition from the IC pool by blocks of observations. This series is thus composed of random blocks of data that come from different stations, to mimic the behaviour of a new IC series. The run lengths of the chart are then computed on many of those runs. The value of L is finally adjusted, by bisection searching, until a prespecified rate of false positives is reached at the desired accuracy.
The other parameter of the chart that should be selected is the allowance parameter, which depends on the target shift size denoted by δ_{tgt}. Since we do not have prior information about the size of the deviations, we estimate it directly on the OC series as explained in Appendix 3.7.1. The allowance parameter is then fixed at $k = \delta_{tgt}/2$, as explained in Section 3.2.4.

The data also contain missing values. Among them, the large gaps prevail since the smoothing process in (3.3.3) removes the shortest gaps of the series. As the observing conditions could be different after a large amount of missing values (different weather conditions or instruments), we restart the scheme after each gap ($C_j^+ = C_j^- = 0$). Blocks composed only of missing values are not used to design the chart. This may happen when some stations contain very few observations on the period studied here, either because they are ancient and stopped observing at the beginning of the period or because they have started observing only recently.

3.4.3 Phase III: Estimation of shift sizes and shapes using SVMs

The CUSUM gives an alert when a deviation is detected in the data but does not provide information about the characteristics (shape and size) of the shift. Such information is however valuable to assign possible causes to the shift or to adapt the type of alerts that is sent back to the observers. To that end, Cheng et al. (2011) appended a support vector regression (SVR) to the CUSUM. This method is designed to predict, after each alert, the magnitude of shifts in independent and identically normally distributed data that only experience jumps. In the following, we extend Cheng et al. (2011) and design a method that is effective to detect the sizes *and* the shapes of the deviations in the sunspot number data. This is achieved by a SVM classifier (SVC) (Burges, 1998) in addition to a SVR on top of the chart.

Input vector

When an alert is triggered, the m most recent observations of the stations are fed into the SVR and SVC which then predict the size and shape of the deviation at the origin of the alert, as explained in the next subsection. In particular, the SVR prediction model writes as:

$$\hat{\delta} = f(V_{t'}) = f(\hat{\epsilon}_{\hat{\mu}_2}(t' - m + 1), \hat{\epsilon}_{\hat{\mu}_2}(t' - m + 2), ..., \hat{\epsilon}_{\hat{\mu}_2}(t')), \tag{3.4.5}$$

where t' denotes the time of the alert and $V_{t'}$ represents the input vector, i.e. a sequence containing the last m observations of the series.

The length m of the input vector should thus be sufficiently large to contain the starting point of most of the deviations while maintaining the computing efficiency of the method. Large shifts are often quickly detected by the chart (short OC run lengths) while the smallest shifts may be identified only after a certain amount of

Algorithm 4: Pseudo-code for computing the length of the input vector

```
/* Find an optimal value for the length of the input vector m          */
Select value for:
δ_tgt, the shift size that we aim to detect
Compute the control limit L using Algorithm 1 for k = δ_tgt/2

for b in (1, B) do
    // sample data with repetitions from the IC processes
    data = resample IC data per blocks
    // add a artifical shift of size δ_tgt on the resampled data
    data + δ_tgt
    compute the chart statistics C+ and C− on those data
    if the chart gives an alert (C+ > L or C− < −L) then
        RL_1[b] = time of alert
    else
        RL_1[b] = large number
if quantile is specified then
    m = quantile(RL_1, specified value)
else
    q_RL1 = quantile(RL_1, 0.5-1)
    m = knee(q_RL1)
```

time (long OC run lengths). Therefore, the latter require larger input vectors than the former. Hence, we propose to select m as an upper quantile of the OC run length distribution for a shift size equal to δ_{tgt}. This is described in Algorithm 4, which works as follows. For a target shift size δ_{tgt}, a large number (B) of IC series are sampled by the MBB procedure and are shifted by δ_{tgt}. The run lengths of the chart are then computed on these artificial series. The length of the input vector may finally be selected as an upper quantile of the run length distribution. If it is unspecified, different quantiles are then evaluated in the range [0.5, 1] and the optimal quantile is selected as the "knee" (Satopaa et al., 2011) of the curve. With this method, m should thus be sufficiently large to allow the identification of shift sizes that are superior or equal to δ_{tgt}.

As the SVM procedures do not support missing values, we have to impute them. Missing observations occurring at the beginning of $V_{t'}$ are simply replaced by the first valid observation encountered, while the "intermediate" gaps are filled by a linear interpolation. However, when there are too many of them, the analysis makes no sense. We decide to only analyse input vectors which have at least 20%

of non-missing values.

Support vector regression

The support vector machine (SVM) (Vapnik, 1998) is a supervised machine-learning procedure, here used as a robust classifier and regressor to predict the shape and size of the deviations. The method has a strong theoretical basis that takes root in the optimization theory. It is able to perform efficiently non-linear classification or regression using a kernel trick that implicitly maps the data into a high dimension where the non-linear problem becomes linear. We only introduce the SVR in the following, since the SVC can be expressed with a similar framework. Smola and Schölkopf (2004) may also be consulted for more detailed explanations.
We denote by $\{ \boldsymbol{x}^j, \delta^j | j = 1, 2, ..., M \}$ the M training pairs. $x \in \mathcal{R}^m$ represents a training input, i.e. a series of m observations that contains a deviation and $\delta \in \mathcal{R}$ is its corresponding output, the size of the deviation. The SVR aims at estimating the continuous regression function relating the deviating observations to the size of the shift, $f(\boldsymbol{x})$, based on the training pairs. This function writes as:

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x}) + b, \tag{3.4.6}$$

where ϕ is the non-linear function mapping the input data into the high dimensional feature space, where the regression may be expressed into a simpler linear problem. The coefficients $\boldsymbol{w}$ and b are then estimated during the training by solving the following optimization problem:

$$\min_{\boldsymbol{w},b} \frac{1}{2}||\boldsymbol{w}||^2 + \lambda \frac{1}{M} \sum_{j=1}^{M} L_\epsilon(\delta^j, f(\boldsymbol{x}^j)), \tag{3.4.7}$$

where

$$L_\epsilon(\delta^j, f(\boldsymbol{x}^j)) = \begin{cases} |\delta^j - f(\boldsymbol{x}^j)| - \epsilon & |\delta^j - f(\boldsymbol{x}^j)| \geq \epsilon \\ 0 & \text{otherwise.} \end{cases} \tag{3.4.8}$$

In the objective function of (3.4.7), the parameter λ represents a trade-off between misclassification and regularization whereas ϵ in the loss function of (3.4.8) is equivalent to an approximation accuracy, i.e. errors below ϵ are neglected. This optimization problem may be rewritten with Lagrange multipliers as a dual problem and easily solved in the input space thanks to the introduction of a kernel function $K(\boldsymbol{x^j}, \boldsymbol{x^k}) = \phi(\boldsymbol{x^j})\phi(\boldsymbol{x^k})$. This kernel function is thus an important hyper-parameter of the method that should be carefully selected.

λ, ϵ and the kernel function $K(\boldsymbol{x^j}, \boldsymbol{x^k})$ are thus the hyper-parameters of SVR. Those should be adjusted to minimize the prediction error evaluated on the validating set – a set of data distinct from the training and testing sets – with a

performance criterion such as the one described below in Section 3.4.3. Usually, the hyper-parameters are selected using a grid search over a finite set of values. Although they should in principle be tuned together, we perform the search over one hyper-parameter at the time and let the others to their default value in the Python package `scikit-learn` (Pedregosa et al., 2011). Those correspond to $\lambda = 1$, $\epsilon = 0.001$ and the radial basis function (RBF) kernel. This approach allows us to save some computing time. Since the performances obtained with this approach are sufficient for our monitoring problem, we do not pursue the research further. Tuning the hyper-parameters together may however increase the performances of the SVM methods. In the following, we use the RBF kernel in all SVM procedures and let $\epsilon = 0.001$, since we do not observe much changes when tuning the value of ϵ. λ will however be adjusted by direct search over a set of values at each call of the SVM methods.

Creation of the training and testing sets

The training and testing sets are constructed by simulations since only a limited amount of (unlabelled) observations are available. Note that we do not construct a validating set here. Such a set is useful to select the values of the hyper-parameters of the method. Since the training and testing sets are constructed by simulations however, we simply create multiple training and testing sets, one for each value of the hyper-parameters, and use those sets to fix the hyper-parameters.
As the SVM procedures are supposed to predict the characteristics of the shift after an alert has been raised by the CUSUM, we generate sets of series that will be first monitored by the control scheme before reaching the SVMs. When an alert will be triggered by the CUSUM, the m last values of the series will be assembled and used as an input vector for the SVMs. Hence, we create series that are initially longer than m. Those are generated from the IC data by MBB. To ensure the efficiency and the generalization of the predictions, we then add various artificial deviations (with different sizes and shapes) on top of these series.

Shift sizes The magnitudes of the shifts, δ, are first randomly sampled from two half-normal distributions (Evans et al., 2000) supported by $[-\infty, ..., -\delta_{tgt}]$ and $[\delta_{tgt}, ..., \infty]$ respectively. We select the scale parameter of the half-normals equal to 3.5, a value that is sufficiently high to reproduce the highest values/deviations observed in the data.
Shift shapes: For each δ, a series x_{ic} of length T' is generated from the IC pool by the BB. Here, we choose $T' = 500$. Three types of general deviations are then artificially constructed on top of the series:

1. jumps: $x(t) = x_{ic}(t) + \delta$;
2. drifts with varying power-law functions: $x(t) = x_{ic}(t) + \frac{\delta}{T'}(t)^a$, where a is

randomly selected in the range $[1.5, 2]$;

3. oscillating shifts with different frequencies: $x(t) = x_{ic}(t)\delta \sin(\eta\pi t)$, where η is randomly selected in the range $[\frac{\pi}{m}, \frac{3\pi}{m}]$.

These three types of deviations cover a large variety of existing shifts. Similar deviations are used in Qiu et al. (2017), to evaluate the performance of their proposed monitoring scheme on simulations. The particular values of the parameters a, T' and η are then selected here to visually correspond to the deviations observed in the data.
Time of the shift: In the data, the shifts may happen not immediately but after an initial IC period. Therefore, we also start the monitoring after a random delay in the range $[m, 3m/2]$, to train the methods at identifying shifts appearing anywhere within the input vector. Note that the SVMs as well as the control chart should be started after m observations are gathered.

The SVR is trained and tested on these constructed sets to predict the size of the deviations in the continuous range $[-\infty, ..., -\delta_{tgt}, \delta_{tgt}, ..., \infty]$. In practice, we observe that the SVR generalizes well and can make predictions on $\mathbb{R}$ even if it was only trained in a smaller range of interest. The SVC also learns on the same sets to identify three different shapes: (1) jumps, (2) drifts and (3) oscillating shifts. If a wide range of deviations are simulated, only three classes are therefore involved in the classification problem.
Note that we do not normalize the training and testing sets since the SVM methods appear to perform sufficiently well for our monitoring problem. In general however, the performances of a SVM method can be improved by normalizing the data.

Performance criteria

The SVM prediction results can then be measured with different criteria. The SVR predictive ability may be evaluated on the testing set with the mean absolute percentage error (MAPE):

$$MAPE = \frac{1}{M}\sum_{j=1}^{M}\left|\frac{|\delta^j| - |\hat{\delta}^j|}{|\delta^j|}\right| \times 100\%, \tag{3.4.9}$$

where $\hat{\delta}^j$ is the shift size predicted by the SVR (which can be positive or negative) and δ^j is the true size. Small values of MAPE are desirable since they correspond to predictions that are close to the actual shift sizes.
On the other hand, the performances of the SVC may be evaluated by the classi-

fication accuracy:

$$ACCURACY = \frac{1}{M}\sum_{j=1}^{M} \mathbb{1}_{\{\hat{\delta}_{c3}^{j}=\delta_{c3}^{j}\}} \times 100\%, \tag{3.4.10}$$

where $\hat{\delta}_{c3}^{j}$ denotes the shape of the deviation (jump, drift, or oscillating shift) predicted by the SVC, and δ_{c3}^{j} is the true shape. A high accuracy is desired since it corresponds to a close match between the predicted and the actual shapes of the shifts.
The accuracy is thus a performance measure of the classifier for all shapes. The confusion matrix may also be computed to obtain a detailed view of the performances of the classifier, class by class. The columns of this matrix represent the prediction labels (here the predicted shapes) whereas the rows are the true labels (the true shapes). Therefore, the elements of the matrix on the main diagonal correspond to the numbers of correct classification per class, while the other entries show the number and the type of classification errors.

This completes our description of the monitoring method whose main parameters are summarized in Appendix 3.7.2. In this appendix, we propose typical values of those parameters – which are also detailed in the following – for monitoring the sunspot numbers. Some simulations are also provided to illustrate the influence of some parameters, such as the block length or the size of the pool, on the performances of the method.

3.5 Monitoring the composite sunspot index Nc

In this section, we use our methodology to solve the monitoring problem of the sunspot numbers. We do so for the composite $N_c = N_s + 10N_g$ (the same approach also works for the two components, N_s and N_g and is presented in the appendix). We first study the low-frequency deviations on data that have been smoothed on a year to extract and analyse low-frequency patterns such as trends and persistent shifts. Then, we examine data that have been smoothed on 27 days (one solar rotation) to detect higher frequency patterns such as sudden jumps. The section ends with an example of a monitoring at multiple frequencies applied to a particular station.

3.5.1 Lower frequency monitoring

In the first step of the low-frequency monitoring of N_c, we smooth the long-term bias ($\hat{\mu}_2$) with a window length of one year as described in Section 3.3.3. As ex-

plained in Section 3.4.1, the network of stations is first reduced to a pool of 119 in-control (IC) stations. In the next step, we extract the IC mean and standard deviation using the K-NN regression described in Section 3.4.1. The selection mechanism finds $K = 4600$ for this step. The resulting mean and standard deviation are then used to standardise all series.
In the second stage, we use the block bootstrap method to calibrate the CUSUM chart at an average run length of $ARL_0 = 200$, as described in Section 3.4.2. This requires choosing the block length first. To do this, we use Algorithm 3 with $lag\_max = 50$ and $B = 200$ runs. We first search the block length over the range [10, 100] with a step value equal to 10. We obtain a block length value equal to 50. Then, in a second round, the block length is selected over a smaller range of values, [40, 60], with step equal to 2. With this procedure, a choice of two solar rotations (54) appears appropriate. It is longer than the lifetime of most sunspots but not too long for practical use. The control limit is then adjusted by bisection searching in the interval [0, 30] using Algorithm 1, with $B = 4000$ runs and an accuracy equal to $\rho = 2$. The calibration then leads to a control limit of $L = 18.9$ for a target shift size of $\delta_{tgt} = 1.5$.

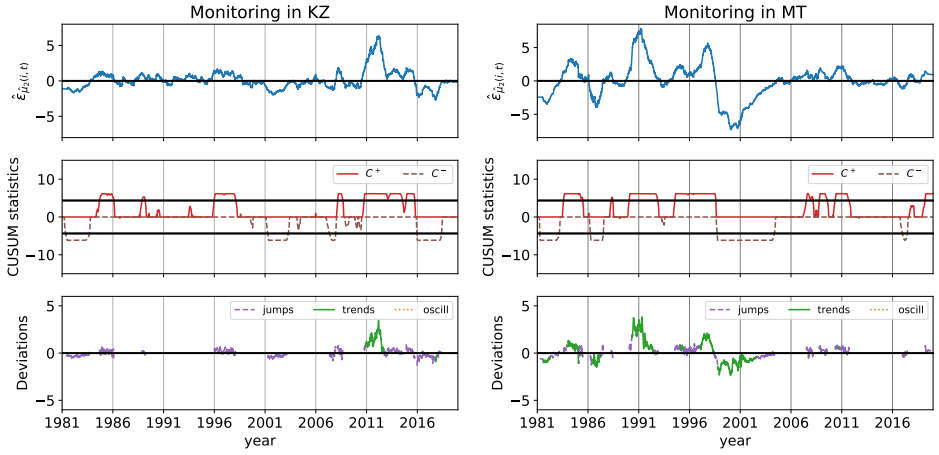
Deviation	Predicted			
True value	jump	drift	oscillating shift	
jump	4192	1	7	4200
drift	34	4166	0	4200
oscillating shift	482	1	3717	4200
	4708	4168	3724	

Table 3.1: Confusion matrix.

Finally, the support vector method for extracting and classifying out-of-control patterns is deployed. It is composed of a SVR to predict the size of the shifts and a SVC to classify the shape of the encountered deviations. We obtain them by creating a set of artificial series of 500 values generated from the IC pool by the BB. These give us series as we would observe in reality including correlations. We then artificially add jumps, trends, and oscillating shifts to them as described in Section 3.4.3. These series are then fed to the CUSUM chart which identifies the out-of-control observations. When an alert is triggered by the chart, an input vector containing the m last observations of the series is assembled. This input vector is then analysed by the SVR and SVC for predicting the characteristics of the shift. In our case, we harvested 63000 such series from the IC pool and we enriched them with artificial patterns. We then calibrated SVR and SVC models by splitting this set into a training set (80%) and a testing set (20%).

The length of the input vector is specified here at $m = 80$, using Algorithm 4 with $B = 4000$ runs. This value of 80 corresponds to the 90-th quantile of the OC run

length distribution, for a shift of size δ_{tgt}. The regularization parameters (λ) of the support vector machines are automatically selected from a searching interval over the range [0, 20] with step value equal to one, to obtain the best prediction results. Those are evaluated using the mean absolute percentage error (MAPE) for the regression and the accuracy for the classification problem, as explained in Section 3.4.3. With this method, λ is set to 13 for the classifier and regressor. The accuracy error ϵ of the SVR is also fixed at 0.001, as stated in Section 3.4.3.



a b

Figure 3.7: (a) Upper panel: the residuals $\hat{\epsilon}_{\hat{\mu}_2(i,t)}$ for N_c smoothed on one year from the KZ over the period studied (1981-2019). In addition to their disparities, the residuals also contain the actual deviations of the station, which have been removed for the design of the chart as explained in Section 3.4.1. Middle panel: the (two-sided) CUSUM chart statistics applied on the residuals in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure for MT over the same period.

After training, the SVR shows a MAPE of around 26 and the SVC has an accuracy of around 95% on the testing set. Cheng et al. (2011) present MAPE values for predicting the sizes of simple jumps in i.i.d. normal data using SVR. Those are of the same magnitude as ours. Considering the various types of deviations that are used to train the SVMs, the performances of our method are acceptable. The confusion matrix is written in Table 3.1 for the 12600 testing pairs. With a perfect classifier, the matrix would be diagonal with elements equal to 4200. As can be seen, the most frequent errors are oscillating shifts predicted as jumps. Those

account for 482 testing pairs and represent 3.8% (482/12600) of the testing set. These errors are expected since a series of consecutive jumps with different sizes may look similar to an oscillating shift. Overall, the performances of the SVMs are sufficient to achieve our goals: identify the origins of the deviations.
Note that we also examine the support vectors but they were not sparse and we did not gain further insight about the SVM methods with this analysis.

Figure 3.7 shows results for two stations: the Kanzelhöhe Observatory in Austria (KZ) and the National Observatory of Japan, in Mitaka (MT). The observatory of KZ is rather stable and belongs to the IC pool. It relies on a stable team of well trained observers. The observatory MT is generally less stable and shows a severe downward trend around 1998. The cause of this downward trend could be traced to the replacement of visual counts from the direct optical solar image by automatic computerized counts based on digital images from a CCD camera. Given the image sensor technology then available, the spatial resolution of the images was limited, and many small spots that were fully detected in earlier visual observations were not detected anymore by the new equipment.

3.5.2 Higher frequency monitoring

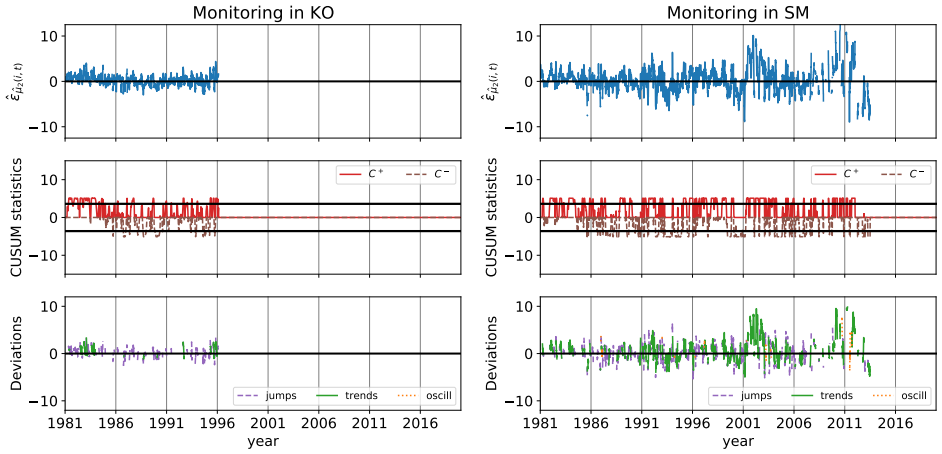
Deviation	Predicted			
True value	jump	drift	oscillating shift	
jump	4153	11	36	4200
drift	65	4135	0	4200
oscillating shift	435	10	3755	4200
	4653	4156	3791	

Table 3.2: Confusion matrix.

The method described above can also be applied to the biases ($\hat{\mu}_2$) smoothed on a shorter time window such as the duration of a solar rotation (27 days). Here the selection of the IC pool yields 100 stations and the number of nearest neighbours comes out to $K = 2400$. This number is smaller than before since we are working at a higher frequency. The block bootstrap and SVMs with the same settings as above can be used to calibrate the CUSUM chart. For $\delta_{tgt} = 1.4$, the control limit of the chart is selected at $L = 13$ to obtain an average run length of 200. The length of the input vector is fixed here at $m = 70$, which corresponds to the 90-th quantile of the OC run length distribution.

After training, the SVR shows a MAPE of around 32 and the SVC has an accuracy of around 95% on the testing set. The confusion matrix is displayed in Table 3.2

for the 12600 testing pairs. As can be seen in the Table, the most frequent errors, representing 3.5% (435/12600) of the testing set, are oscillating shifts predicted as jumps.



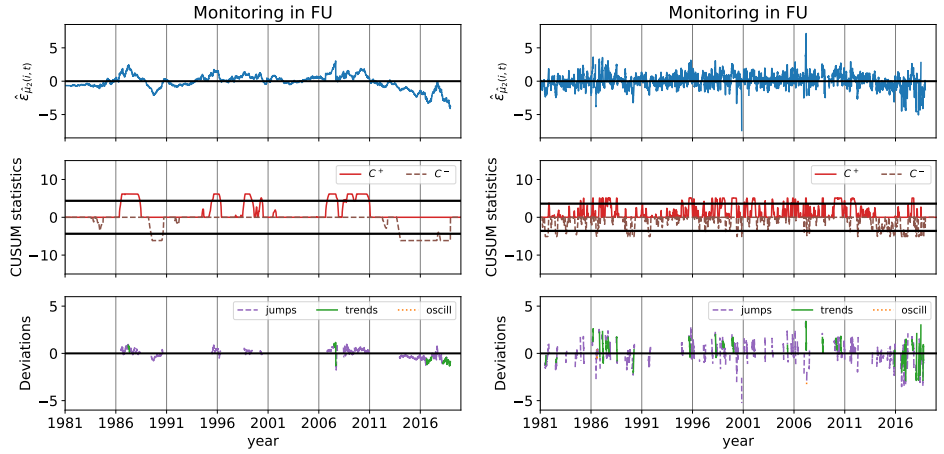
a b

Figure 3.8: (a) Upper panel: the residuals $\hat{\epsilon}_{\hat{\mu}_2(i,t)}$ for N_c smoothed on 27 days for KO over the period studied (1981-2019). In addition to their disparities, the residuals also contain the actual deviations of the station, which have been removed for the design of the chart as explained in Section 3.4.1. Middle panel: the (two-sided) CUSUM chart statistics applied on the residuals in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure for SM over the same period.

Figure 3.8 shows the methodology applied to the observer Koyama (KO) and to the observatory of San Miguel in Argentina (SM). KO was a Japanese station run by a single dedicated observer whose records (which stopped during 1996) were very stable. On the contrary, SM is a severely OC station that experiences large known deviations that were already visible in Figure 2.10. The large variations observed in SM are likely caused by the rotation of several observers involved in the counting process. In some countries, the public observatories have also an educational function. Their team of regular observers are usually small and are often completed by student or amateur astronomers that are frequently replaced, which causes large variations. Unfortunately, we could not identify more precisely the origin of the deviations since their observations stopped several years ago and we have not succeeded in contacting them yet. The lack of information is a common

problem we face when investigating past deviations in stations that are now inactive and is therefore worth mentioning.
As we see in the figure, the biases vary a lot at 27 days, a scale which is close to the short-term regime. The actual monitoring should be based on a larger scale otherwise some stations such as SM would receive almost constant alerts. If a particular deviation is detected at higher scale (such as one year), it might be interesting however to analyse it at 27 days, to better identify its origin.

3.5.3 Monitoring at multiple frequencies



a: Low-frequency monitoring b: High-frequency monitoring

Figure 3.9: (a) Upper panel: the residuals $\hat{\epsilon}_{\hat{\mu}_2(i,t)}$ for N_c smoothed on 365 days from FU over the period studied (1981-2019). In addition to their disparities, the residuals also contain the actual deviations of the station, which have been removed for the design of the chart as explained in Section 3.4.1. Middle panel: the (two-sided) CUSUM chart statistics applied on the residuals in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure for the values of $\hat{\epsilon}_{\hat{\mu}_2(i,t)}$ smoothed on 27 days in FU.

Figures 3.7 and 3.8 display instances of a stable IC station included in the pool and a typical out-of-control station for the high- and low- frequency monitoring respectively. To better grasp the motivations of a monitoring at multiple frequencies, the method is applied to the data smoothed on 27 days and one year of the observer Fujimori (FU) in Figure 3.9. The FU station is composed of a single dedicated

observer in Japan, who has observed without interruption since 1968 until today, producing one of the longest individual series. His observations are included in the IC pool but yet suffer from recent deviations. In particular, the upward deviation (which looks like a spike) reported in 2007 in FU (Clette, 2013) as well as the downward drift occurring after 2014 are well identified in Figure 3.8a. Figure 3.10 shows a zoom of Figure 3.8a on the time period from 2007 to 2008. After a progressive upward shift, the station experiences a rapid downward trend over five days. This trend, which looks like a jump in the whole period view, it thus correctly classified by the SVC. Even by taking a closer look on the figure however, it remains difficult to precisely identify the origin of the shifts on data that are smoothed on a year. By looking at a smaller scale of 27 days in Figure 3.8b, we can better characterize the shift in 2007 as a short event and pinpoint its location. After investigations, this deviation appears to be related to a small over-count that appeared in early 2007 (three groups were reported in FU while most of the network only observed two groups) while the drift might be associated to the health condition of the observer. Note that the long-term biases are not defined (i.e. set to missing values) when the median of the network is equal to zero, see (3.3.3). This regime corresponds to those of the variability at minima, represented by ϵ_3. Due to the smoothing procedure of (3.3.3), the deviations that appeared close to solar minima, such as the jump in FU, are thus particularly visible.

As shown in the figures, the monitoring and the SVM procedures can cope with a large variety of shifts ranging from small and persistent deviations to large oscillating shifts. The procedures automatically detect major deviations recently discovered by hand as mentioned above. More identified prominent deviations as well as results for other stations are shown in Appendix 3.7.3. In addition, the chart also unravels many other shifts, typically smaller, that are otherwise difficult to identify.

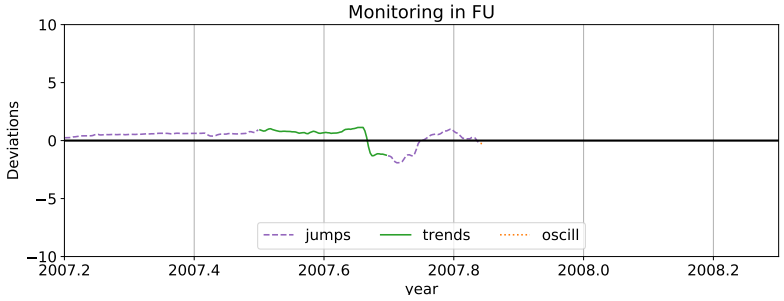


Figure 3.10: Sizes and shapes of the deviations (taken from Figure 3.9a) predicted by SVMs in FU over 2007-2008.

Note that the figures represent past observations, which have not been monitored

by any control scheme. Consequently, the stations may stay in alert for long consecutive periods. If this method is applied on future observations, any major deviation will be promptly corrected. Therefore we expect better results for data that have already been monitored.

3.6 Conclusion and perspectives

In this chapter, we construct a non-parametric control scheme to monitor the sunspot numbers across the wide network of stations. The approach allows us to deal with the missing values, autocorrelations and non-normality of the data. The procedure is based on a particular choice of methods for smoothing, robust anomaly detection, and anomaly classification. Other methods exist for these steps; yet we believe that our choices are particularly suitable for the problem at hand. The features of our approach are multi-scale smoothing, CUSUM charting, SVM classification and detailed graphical displays. They also include an automatic pre-selection of an in-control pool and the powerful calibration of the chart using block bootstrap procedures. The associated advantages are robustness, flexibility, automation, and guided interpretation of results.

The method identified a wide range of deviations, which were mostly undetected in previous analyses. As have been seen, monitoring on at least two frequencies is essential to capture these anomalies. Some patterns first attract interest on a long-term scale but it is at the short-term scale that their potential root-causes can be suggested. The method also helps us to identify the causes of major deviations that occurred in the series. Most of them have not been related to specific causes yet but will soon be investigated. Moreover, as the classification results are complex, careful analyses are required to fully interpret them, which opens new research perspectives. Further investigations are for instance required to see if the drifts may be associated in a more systematic way to the ageing of the instruments, or if the oscillations may reflect the periodic alternation of observers in large stations.

This automated method allows us and the researchers of WDW-SILSO team at the Royal Observatory of Belgium, who are in charge of producing the International Sunspot Number (ISN), to have a harmonized view across the network of stations. It also provides a way to give specific and targeted advice to the observers. Different types of alerts can for instance be designed by combining the information about the magnitude, the shape as well as the duration of the shifts. Those can be used to send immediate alerts back to the observers for the large jumps or the week-long shifts detected whereas the other deviations, which have typically a smaller impact on the quality of the series, may simply be included in an annual report containing the statistics of the shifts over the elapsed year.
The complete re-examination of past data of the whole panel has just started.

When they will be finished, these analyses will allow us to arrive at a cleaner data stream and to release an improved version of the ISN. Additionally, the implementation of the method in the continuous surveillance of future observations will lead to a faster detection and identification of inconsistencies, their elimination by better observer training or equipment maintenance, and finally to a more precise determination of the sunspot numbers in the future.

Although presented here for a special application, our methodology is general and can be adapted to other panels of time-series data such as those observed for instance in the manufacturing or financial industry. This will be shown in Chapter 5, where this method will be applied to the photovoltaic energy production in Belgium. The computational aspects of the method, which have been mostly ignored here, will also be discussed in this chapter. A tutorial for the package, which is associated to the method and written in the programming language Python, is in particular provided in Chapter 5.

3.7 Appendix

3.7.1 Selection of the target shift size

Without prior information about the deviations, the target shift size, δ_{tgt}, may be estimated on the OC series by a recursive method described in Algorithm 5. This procedure is explained below. For an initial value of δ_{tgt}, denoted by δ_0, we first compute the control limit of the chart with Algorithm 1. The chart is then applied to B series of data that are resampled from the OC series by the BB procedure. In this work, we use $B = 4000$ series if not specified otherwise. For each sample, the magnitude of the deviation is computed after the alert using the following formula (Montgomery, 2004, Section 8.5):

$$\hat{\delta} = \begin{cases} k + \frac{C_i^+}{N^+} & \text{if } C_i^+ > L \\ -k - \frac{C_i^-}{N^-} & \text{if } C_i^- < -L, \end{cases} \tag{3.7.1}$$

where $\hat{\delta}$ is the estimated shift size expressed in standard deviation units and N^+ (resp. N^-) represents the number of observations where the CUSUM statistic C_i^+ (resp. C_i^-) has been non-zero. Although valid for i.i.i. normal data, the above formula can still be used in general to provide a rough approximation of the shifts size (those will be estimated afterwards by support vector regression). A new value for δ_{tgt} is then computed as a specified quantile of the shifts size distribution and the algorithm is iterated few times until convergence. In practice, we stop the computations when the difference between the shift sizes obtained at iteration i and $i-1$ is inferior or equal to 0.1. In general, we recommend to select a quantile

around 0.4-0.5. Smaller values are not appropriate to the identification of the largest shifts and vice-versa.

Algorithm 5: Pseudo-code for estimating a target shift size

```
/* Estimate a target shift size                                    */
```

Select values for:
δ_0, an initial value of the target shift size
ρ, the accuracy of the algorithm's convergence

$\delta_{prev} = 0$; $\delta = \delta_0$
while $|\delta - \delta_{prev}| > \rho$ **do**
  Compute the control limit L using Algorithm 1 for $k = \delta/2$
  $\delta_{prev} = \delta$
  for *b in (1, B)* **do**
    `// sample data with repetitions from the OC processes`
    resample OC data per blocks
    compute the chart statistics C^+ and C^- on the resampled data
    if *the chart gives a positive alert* $(C^+ > L)$ **then**
      $shift\_size[b] = k + \frac{C^+}{N^+}$ (Montgomery's formula)
    if *the chart gives a negative alert* $(C^- < -L)$ **then**
      $shift\_size[b] = -k - \frac{C^-}{N^-}$ (Montgomery's formula)
  $\delta = quantile(shift\_size)$
$\delta_{tgt} = \delta$

3.7.2 Parameters of the monitoring

The main parameters of the monitoring method are presented in Table 3.3. In this table, two numbers are often proposed as typical values for monitoring the sunspot numbers. Those correspond to the values that should be selected to monitor respectively the data smoothed on 27 days and on one year.

Sensitivity	Symbol	Name	Default value	Typical values for SN	How to set
	-	clustering method	k-means	k-means	default works well
	K	number of nearest neighbours	-	[2400, 4600]	to obtain the best standardisation of the data (see Section 3.4.1)
	-	BB method	MBB	MBB	default works well
*	-	block length	$N^{1/3}$	54 days	Algorithm 3
**	δ_{tgt}	target shift size	1.5	[1.4, 1.5]	Algorithm 5
	k	allowance parameter	$\delta_{tgt}/2$	$\delta_{tgt}/2$	optimal formula ($\delta_{tgt}/2$)
**	L	control limit	-	[13, 18.9]	Algorithm 1
*	ARL_0	IC average run length	200	[200-500]	rate of false positives
	m	length of the input vector	-	[70, 80]	Algorithm 4
*	-	scale of half-normal distribution (construction of the training sets)	-	3.5	to reproduce the max values of the data
	ϵ	accuracy error (SVR only)	0.001	0.001	grid search
	λ	regularization parameter	1	13	grid search
	-	kernel function	RBF	RBF	default works well

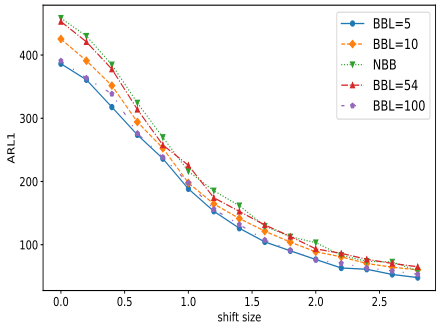
Table 3.3: Main parameters of the monitoring procedure. The table details the sensitivity, symbol, name, default value and typical values of the parameters for the sunspot numbers (SN). It also displays the method that may be used to select an appropriate value for those parameters. The first rows are related to the phase I SPC, the second set of rows corresponds to the Phase II SPC whereas the third rows are associated to the Phase III (support vector machine). In the table, a single star (*) indicates that the parameter has a moderate impact whereas the double stars (**) highlight the parameters that have the most significant influence on the performances of the method. The symbol N also corresponds to the number of observations that are available in each series.

The main parameters that influence the performances of the method are associated to a star (*) or double stars (**) in the table. Among them, the IC average run length (ARL_0), the control limit (L) and the target shift size (δ_{tgt}) are directly related to the CUSUM chart. Their impact are thus studied in many different papers such as in the work of Qiu (2013). The other parameter that has the most significant impact on the detection power of the method is the block length. To illustrate this, we present in the following some simulations which show the influence of the block length, the block bootstrap method and the number of IC stations included in the pool on the shifts detection. The Phase III parameters, which affect the estimation of the shifts characteristics but not the detection power (alerts) of the method, will not be studied here.

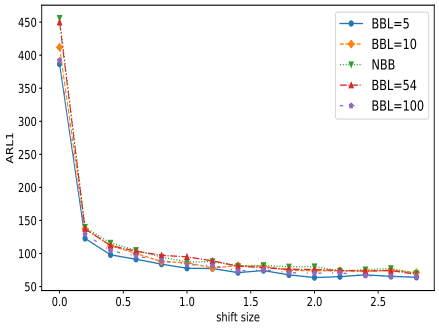
Influence of the block bootstrap

Figure 3.11 displays the out-of-control average run lengths, denoted ARL_1, of the monitoring method as a function of the shift size. The $ARL1$ represents the mean number of samples collected from the appearance of a shift to the alert of a method. For same values of IC average run length (ARL_0), a method has thus better performances if its $ARL1$ values are lower. Those ARL_1 values are computed using the moving block bootstrap method for different block lengths (BBL), which are specified in the legend. The figure also shows the ARL_1 values for the monitoring scheme calibrated using the non-overlapping block bootstrap (NBB), with a block length equal to 50. This value of 50 was selected using Algorithm 3 and is optimal for NBB. The ARL_1 values are then computed as follows. For each value of the block length and for the NBB method, the control limits of the CUSUM chart are calibrated using Algorithm 1 to reach an IC average run length equal to $ARL_0 = 200$ with an accuracy of $\rho = 2$ over $B = 2000$ runs. With this method and for a target shift size equal to $\delta_{tgt} = 1.5$, the control limits are equal to $L = 9.04$ for $BBL = 5$, $L = 13.5$ for $BBL = 10$, $L = 18.9$ for $BBL = 54$ (the proposed method), $L = 9.8$ for $BBL = 100$ and $L = 19.7$ for the NBB method with $BBL = 50$.
Then, at each run, data are sampled per blocks of 500 observations from the IC stations and are concatenated to form "new" IC series containing 2000 values (four blocks are thus selected for each series). Artificial deviations of various sizes are added on top of those IC series (x_{ic}). Finally, the average run length ARL_1 is computed over 2000 runs. In practice, we simulate three different shapes for the deviations:

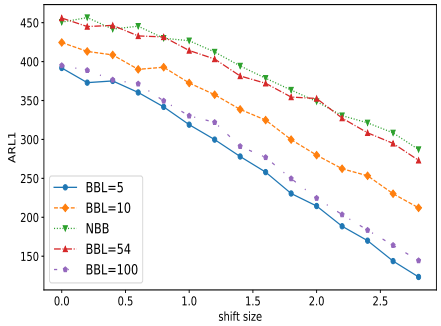
- jumps: $x(t) = x_{ic}(t) + \delta$;
- drifts: $x(t) = x_{ic}(t) + \frac{\delta}{500}(t)^a$, where a is randomly selected in the range $[1.5, 2]$;
- oscillating shifts: $x(t) = x_{ic}(t) + \sin(\eta\pi t)\delta$, where η is randomly selected in the range $[0.02, 0.2]$.



a: ARL_1 values for jumps.



b: ARL_1 values for drifts.



c: ARL_1 values for oscillations.

Figure 3.11: Out-of-control average run length, ARL_1, as a function of the shift size. The ARL_1s are computed for the moving block bootstrap method with different block lengths (BBL), which are specified in the legend. The monitoring method is also designed with the non-overlapping block bootstrap (NBB in the legend) with a block length equal to 50. The results are obtained by simulations based on IC data that are smoothed on a yearly scale.

As can be seen in Figure 3.11, using the non-overlapping BB instead of the moving BB does not change the performances of the method. This conclusion is also valid for the circular BB since we have enough observations. All three methods can therefore be used without affecting the monitoring results. On the contrary, the choice of block length affects the performances of the method. Few changes are observed for drifts, which are the easiest deviations to detect. More differences appear however for jumps and oscillating shifts.
When the block length is equal to 5 or 10, the CUSUM chart is calibrated on series

that are composed of a large number of blocks. Abrupt changes often appear from one block to another, which creates many alerts. The control limit of the chart is thus lower than with the proposed method, which has a block length equal to 54. When the block length is equal to 100, the chart is calibrated on series that contain fewer but longer blocks. Inside those blocks, the data have long positive autocorrelation since the long-term biases are smoothed on 365 days. The accumulation of small positive values inside a block leads then to many alerts and the control limit is again decreased with respect to the proposed method. The block length which is selected in this chapter is equal to 54 and is between those two regimes. It takes the main part of the autocorrelation into account while at the same time designing a chart which is not over-sensitive. Indeed, the control chart calibrated with a block length equal to 54 has the highest ARL_1 values when no shifts are simulated (i.e. the first values in the figure around 450). Hence, Algorithm 3, which was used to select the block length at 54, appears to work correctly and can be used in other applications to choose the block length.

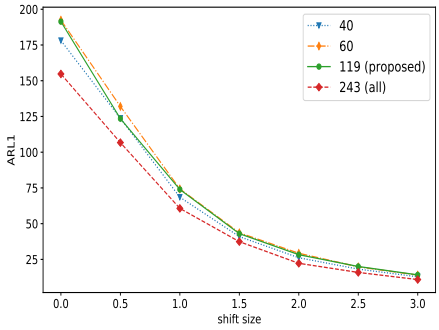
Note that when no shifts are simulated, the ARL_1 values are larger (around 400-450) than the pre-specified rate of false positive $ARL_0 = 200$. This is expected since the chart was calibrated for the different scenarios on IC series composed of blocks of smaller lengths (equal to respectively 5, 10, 50, 54 and 100) than those used here, which contain 500 values. This value of 500 is selected to compare the different scenarios on series that are similar to the actual data. If they were tested on resampled series that were composed of blocks of length equal to respectively 5, 10, 50, 54 and 100, then all methods would have shown similar performances. What interest us here is not to test that the methods designed with different block lengths are well calibrated, but to evaluate their performances on simulated shifts that are close to the actual deviations.

Influence of the pool of IC stations

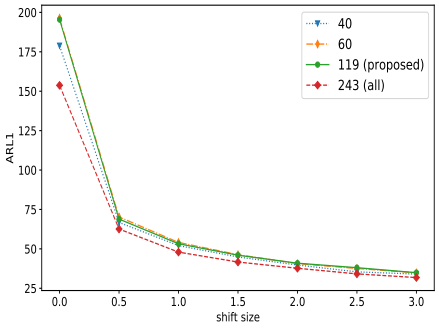
Figure 3.12 displays the ARL_1 values of the monitoring method calibrated on different number of (IC) stations as a function of the shift size. Those ARL_1 values are computed as follows. For each particular pool, the control limits of the CUSUM chart are first calibrated to reach an IC average run length equal to $ARL_0 = 200$ with an accuracy of $\rho = 2$ using Algorithm 1. For a target shift size equal to $\delta_{tgt} = 1.4$ and $B = 2000$ runs, the control limits are selected with this method at $L = 18.8$ for $N_{IC} = 40$, $L = 21.4$ for $N_{IC} = 60$, $L = 21.2$ for $N_{IC} = 119$ (the proposed method), and $L = 14.9$ for $N_{IC} = 243$. Note that same conclusions can be drawn with other values of the target shift size. $N_{IC} = 243$ corresponds to a scheme calibrated on the entire network. Although the network contains 278 stations, some stations are excluded since they contain few values on the period studied. When smoothing the data with a MA filter of length equal to one year

in (3.3.3), we impose that each moving window should contain at least 10% of observations to produce a valid value for $\hat{\mu}_2(i,t)$. After this smoothing step, we obtain stations that do not contain any value for $\hat{\mu}_2(i,t)$ on the entire period studied and remove them from the network.
To compute the ARL_1 values, we sample data per blocks of 54 observations from a subset containing $N_{IC} = 119$ stations, which corresponds to the pool selected with the procedure described in Section 3.4.1. We concatenate those blocks to form "new" IC series containing 2000 values and artificial deviations of various sizes δ are added on top of those IC series. Finally, the average run length ARL_1 is computed over 2000 runs. As in the previous simulation, we simulate three general shapes for the deviations: jumps, drifts and oscillating shifts.

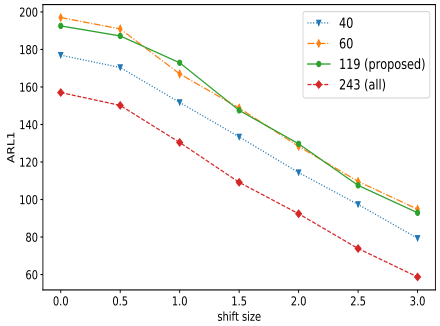
As can be seen in Figure 3.12, the detection power of the chart does not change much with different pools, except for oscillating shifts. When the method is calibrated to reach a value of $ARL_0 = 200$ on the whole network, it has a low control limit value to be robust against the numerous deviations of the data. The method has thus lower ARL_1 values when monitoring a subset of $N_{IC} = 119$ more stable series, as can be seen in the figure. When the pool is small and contains 40 stations, which represents only 5.3% of the data, we expect that the control limit of the chart would be higher than with a larger pool since the method is calibrated on stations that are particularly stable. We observe a different situation however, since some stations have a short observing period. Those are included in the pool because they have a very low variance despite the fact that they may not be aligned with the median of the network (see the clustering procedure which is applied on the stability criterion defined in (3.4.1)). When designed on those stations, the chart is thus often in alerts, which leads to lower values of the control limit with respect to a method calibrated on the proposed pool of 119 series. Its ARL_1 values are thus lower.
In between those two extremes cases, few differences are visible between different pools. The control scheme calibrated on a pool containing 60 stations (which corresponds to 10.8% of the data) has for instance similar performances as the same scheme adjusted on 119 series (29.1% of the data). There is thus some flexibility to choose the number of IC series to include in the pool. For this reason, using different clustering methods does not affect significantly the performances of the monitoring scheme.



a: ARL_1 values for jumps.



b: ARL_1 values for drifts.



c: ARL_1 values for oscillations.

Figure 3.12: Out-of-control average run length, ARL_1, as a function of the shift size. The ARL_1s are computed for the control scheme calibrated on subsets containing different numbers of IC series, which are specified in the legend. The results are obtained by simulations based on IC data that are smoothed on a yearly scale.

3.7.3 Additional results for the composite (N_c)

The developed monitoring method has been applied to the number of spots N_s, groups N_g and composites $N_c = N_s + 10N_g$ at two different scales for the 278 stations in the database. That represents a large amount of information that has not been completely analysed yet. In Section 3.5, we present results for typical in-control (IC) and out-of-control (OC) stations, for analysing the high-frequency as well as low frequency deviations in N_c. The station FU is also analysed at two different scales. In this appendix, we present more results about some prominent deviations that occurred in N_c. The persisting drifts are first displayed for $\hat{\mu}_2$

smoothed on a year. The high-frequency deviations are then shown for $\hat{\mu}_2$ smoothed on 27 days.
The figures presented in the following are composed of four panels: the upper panel exposes the $\widehat{eh}(i,t)$ of (3.3.3) for a particular station, the second panel shows the residuals defined in (3.4.3) for the station, the third panel represents the CUSUM statistics applied to the residuals in square-root scale whereas the lower panel displays the characteristics (magnitude and shape) of the shifts predicted by the support vector machine procedures.

Lower frequency monitoring

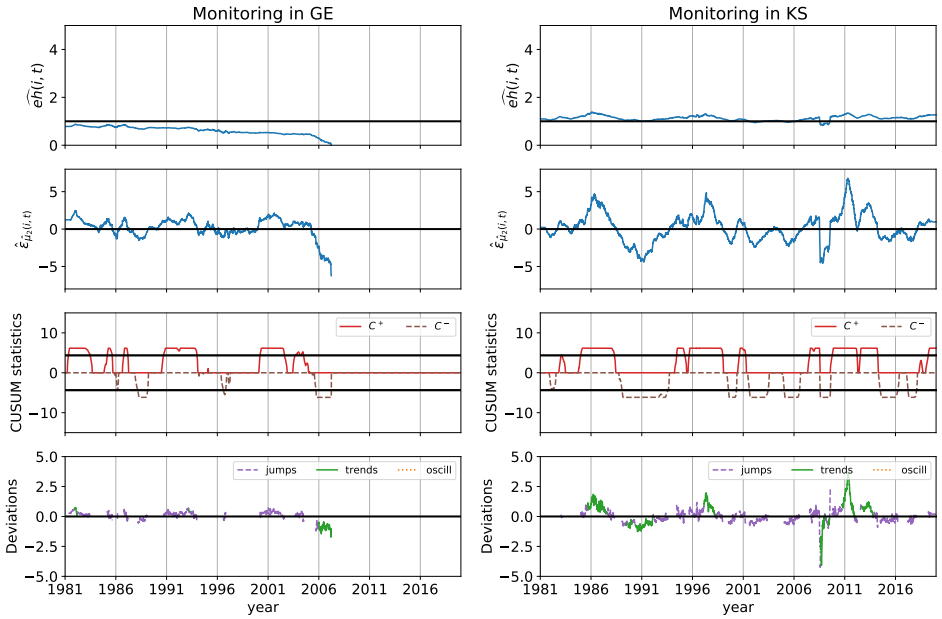


Figure 3.13: The control scheme applied on the data from observer Gerard (GE) in Belgium over the period studied (1981-2019). b) Similar figure for the station Kislovodsk (KS) in Russia over the same period.

In this subsection, we present additional examples of prominent drifts observed in the composite N_c smoothed on a year. A downward deviation is visible at the end of the data from the station GE in Figure 3.13. The station GE was composed of a single dedicated observer living in Belgium. The deviation observed in the series

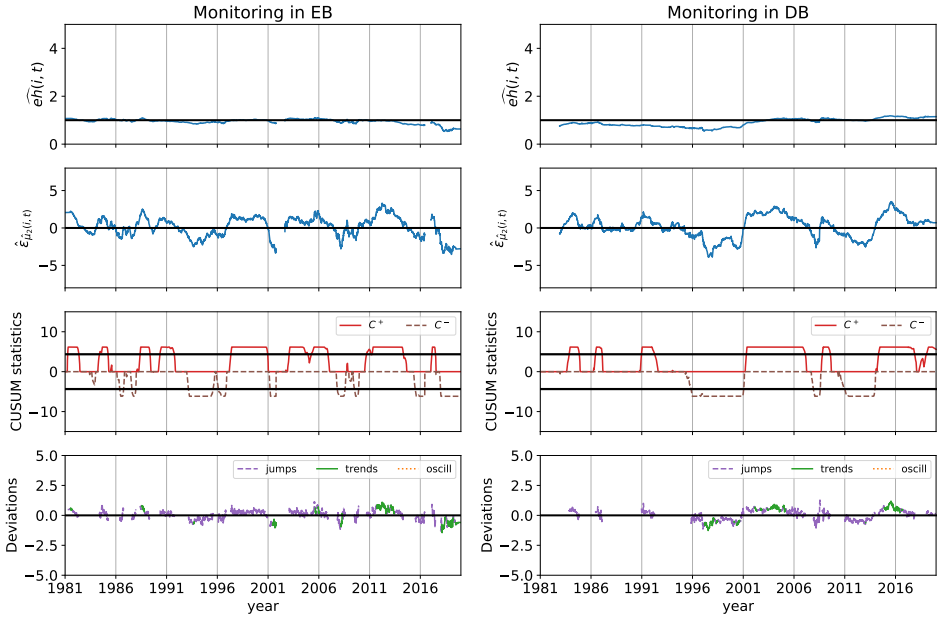


Figure 3.14: The control scheme applied on the data from the Ebro Observatory (EB) in Spain over the period studied (1981-2019). b) Similar figure for the data from observer DeBacker (DB) in Belgium over the same period.

is caused by a change of location since the observer moved to another residence at the end of the observations. The eyesight of the observer may also have declined with time, which may explain the drop in the series as well. A slight decrease is also visible in the station Kislovodsk (KS) in Russia from mid 2008 till mid 2009. This temporary downward drift may be linked to the installation of a CCD camera in 2008. In 2010, the station also switches to digital image processing, which may cause the slight increase observed afterward. In Figure 3.14, a deviation appears in the end of the data from the Ebro Observatory (EB) in Spain, which is otherwise very stable. It is probably caused by a change in the camera, which led to problems with the subsequent image treatment. The software was able to process the images again after enhancing the contrast, at the expense of the smallest details in the images. This may probably explain the decrease that we observe at the end of the series. In 1984, the single-observer station DB in Belgium switched from direct observation of the Sun to a projection method that enables the precise drawing of the sunspots. This triggers the small downward deviation observed in the second

half of the 1990s since the smallest spots were harder to see with the new procedure. In 2000, the observer bought a new filter, which allowed him to better see the smallest spots. The observer also had more personal time to carefully count the spots and groups. These changes cause a rapid upward shift around 2000-2001 that brings the level of the station back, aligned with those of the network.

Higher frequency monitoring

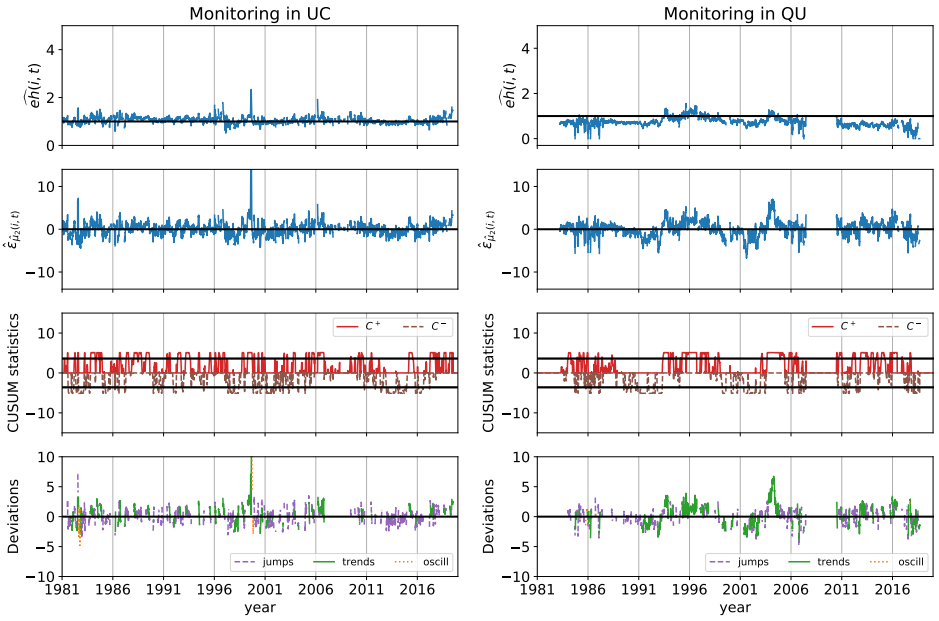


Figure 3.15: The control scheme applied on the data from the observatory of Uccle (UC) in Belgium over the period studied (1981-2019). b) Similar figure for the station Quezon (QU) in Philippines over the same period.

We present here typical examples of major jumps that occurred in the composite N_c smoothed on 27 days. Figure 3.15 shows the method applied to the data from the station Uccle (UC), in Belgium. UC is stable over time but suffers however from a large jump in 1999, already visible in previous analyses (Clette, 2013). This deviating episode is related to the intense participation of a particular observer that did not count with the same precision than the other members of the team. He was recruited at a time where there was a lack of observers but finally stopped

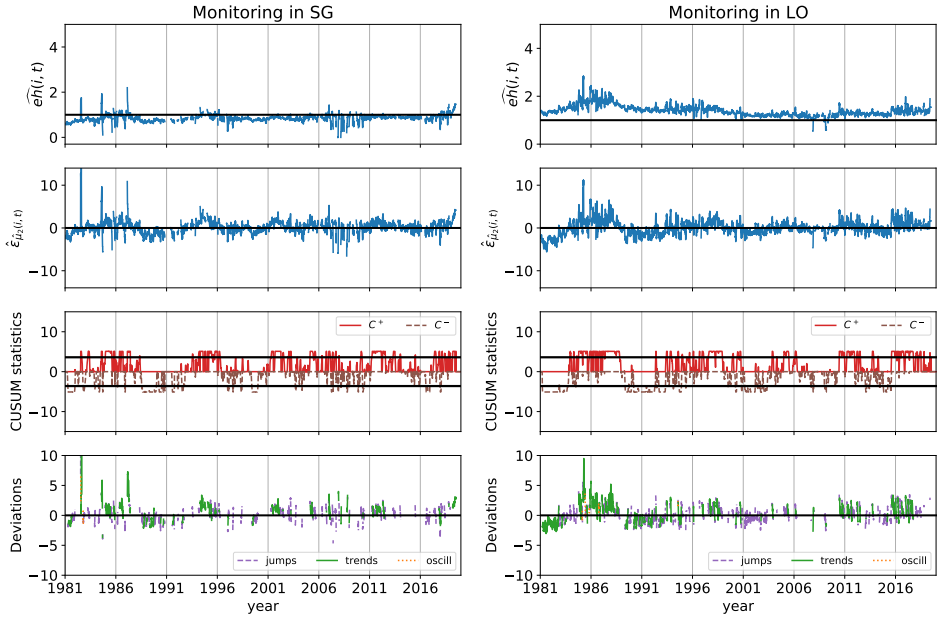


Figure 3.16: The control scheme applied on the data from the Southern Cross Observatory (SG) in Bolivia over the period studied (1981-2019). b) Similar figure for the Observatory of Locarno (LO) in Switzerland over the same period.

observing after a while. The station Quezon (QU) in Philippines experiences more variations over time. They are likely caused by an alternation of observers as in the station San-Miguel (SM) presented in Section 3.5.2. Figure 3.16 displays another type of deviations. The observer from the Southern Cross Observatory (SG) in Bolivia counts more spots than the network at the solar minima (around 1986, 1996 and 2008). This is particularly visible in the three spikes that appear at the beginning of the period studied, soon after he starts observing. Since similar deviations do not happen later in the series, the spikes are most probably related to the learning curve of the observer. We observe later around 1996 and 2008 slight excesses of spots that are also visible in other stations. They may be linked to particular errors that only occur at solar minima when there are few or even no spots on the Sun: some observers tend to over-scrutinize the Sun and may count one or two spots in excess. This effect is not observed in other parts of the solar cycles, when the sunspots are more numerous. Similar deviations are also visible in the observations of the station Locarno (LO) in Switzerland around the solar

minima of 1986 (cycle 22).

3.7.4 Results for the number of spots (N_s) and groups (N_g)

The monitoring method has been applied to the data from station FU in Figure 3.9 for N_c. Same results are displayed here in Figure 3.17 for the number of spots and in Figure 3.18 for the groups. As stated in Section 3.5, FU is a stable station which is included in the pool. It is presented here since the few deviations of FU are particularly visible and composed of two different types: short-time and persisting shifts. As can be seen in the figures, the proposed method can cope with the different distributions and autocorrelation structures of the data and produces coherent results for N_s, N_g as well as N_c.
Since N_s, N_g and N_c are counted on the same image of the Sun captured at a particular moment of the day, similar deviating patterns are visible in the three quantities. The deviations are usually more apparent in N_s than in N_g or N_c since the groups are more robust to counting errors than the individual spots. The jump that occurs in 2007 is for instance much lower in N_g and N_c than in N_s. Note that the results presented earlier in this chapter focus on N_c, which is closer to the International Sunspot Number.
By looking separately at N_s and N_g, we may also gather more information about the deviations. The drift that appears at the end of the series is gradual in N_s and steep in N_g. It may express the fact that the observer progressively sees less spots, which after a certain time leads to a decrease in the number of groups as well. Further researches are needed to see if some types of deviations in N_c may be related to rather N_s than N_g or conversely.

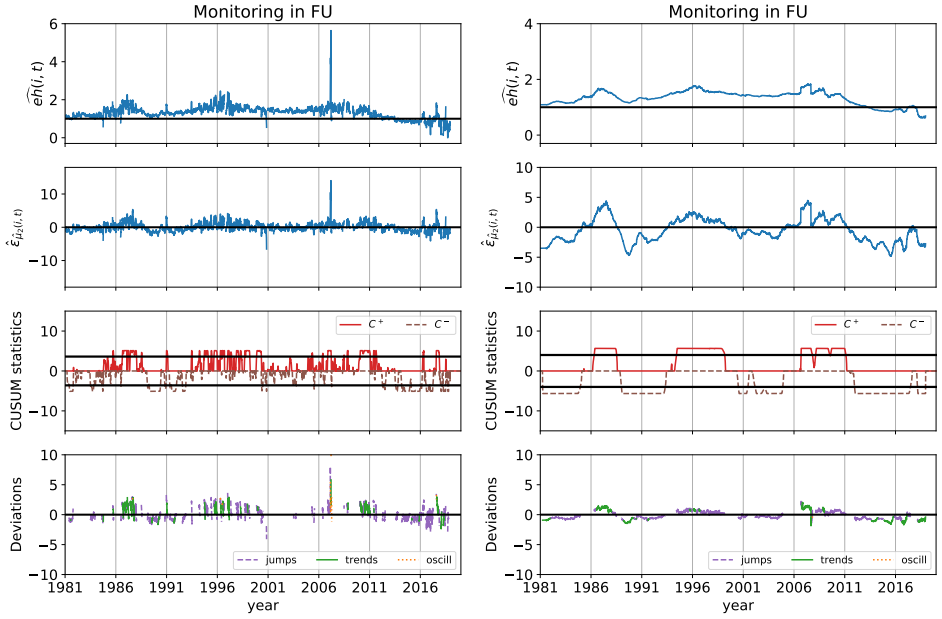


Figure 3.17: The control scheme applied on the number of spots smoothed on 27 days from station FU in Japan over the period studied (1981-2019). b) Similar figure for N_s smoothed on 365 days in FU over the same period.

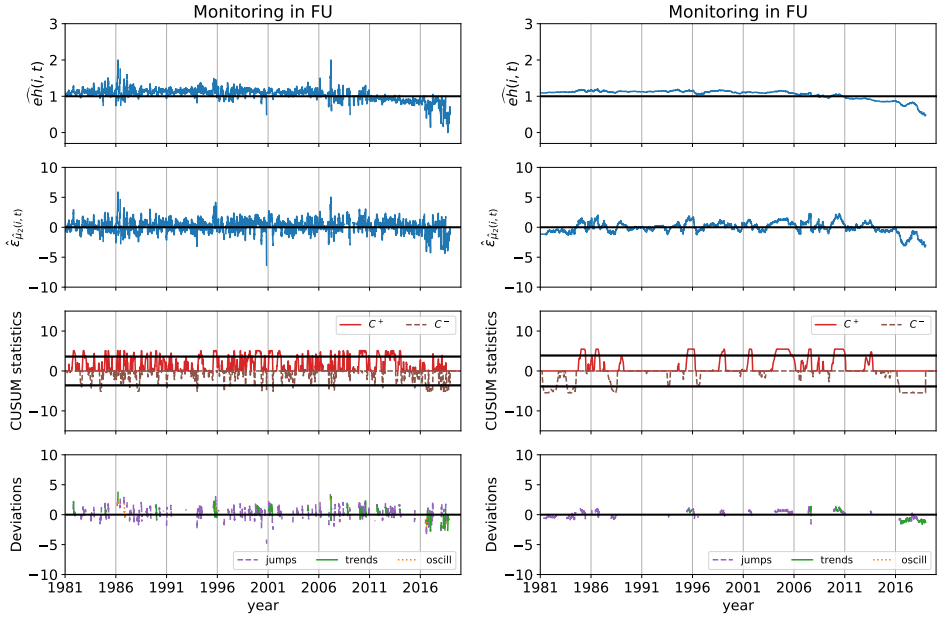


Figure 3.18: The control scheme applied on the number of groups smoothed on 27 days from station FU in Japan over the period studied (1981-2019). b) Similar figure for N_g smoothed on 365 days in FU over the same period.

CHAPTER 4

Neural-networks based monitoring of the sunspot numbers

Artificial neural networks are powerful computing systems that were initially inspired by the functioning of the brain. Due to their capacity to well represent numerous non-linear problems with a potentially large number of inputs, they are now used in many fields with growing interest. Predicting the size of the deviations occurring in the sunspot numbers and classifying their shapes are typical examples were such tools could be used and are expected to perform well. To be effective on complex tasks however, the neural networks need to be trained on a large number of labelled data, which are not available for all applications. Those methods could also be brittle against noises or targeted attacks and have a poor algorithm transparency. Hence, they require consequent post-analysis to be interpretable. For the sunspot numbers however, the problem of the absence of labelled data has already been solved in Chapter 3, by training the support vector machines on simulations. Having obtained good results with this method despite the noise of the data, we decide to construct neural networks using equivalent simulations in the following. The monitoring procedure developed in Chapter 3 will then allow us to compare our results with a fully-understood and interpretable method.

In this chapter, we first construct feed-forward neural networks for predicting the shifts sizes and classifying the shapes of the deviations in the sunspot numbers. Then, we also design recurrent neural networks, which are more adapted to deal with time series data, for the same purposes. Those recurrent networks achieve the same performances as the feed-forward networks with less parameters. Both (recurrent and feed-forward) networks are finally compared with the approach developed in Chapter 3 for various shift sizes and shapes. The neural networks-based

control schemes appear to outperform the previous method on several cases, especially to detect large or oscillating shifts. Interesting results are also obtained by analysing the shapes of the deviations predicted by the networks along time.

4.1 Introduction

The monitoring procedure developed in Chapter 3 relies on efficient and well-studied statistical procedures. The method is composed of two main parts: (1) a CUSUM chart designed by block bootstrap that is responsible for the monitoring and (2) support vector machine (SVM) procedures that predict the size and shape of the shift after an alert has been raised. This method will thus be referred to as "CUSVM", i.e. a contraction between the CUSUM and the SVM in the following. Stepping back from our proposition, we soon realize that some aspects of the CUSVM method can be improved.

1. **Numerous stages of the procedure**: The method requires many steps, from the estimation of the long-term error of the stations to the predictions of the SVMs. Although the CUSUM chart is simple to understand with respect to more elaborated charts such as those described in Qiu et al. (2017) and well known for statisticians, the successive stages of the method are difficult to track.

2. **Problem of the parameters adjustment**: The procedure contains only few parameters but some of them are difficult to adjust. We partially solve this issue by proposing several pseudo-algorithms to calibrate them. These algorithms are however time-consuming and some of them are not fully adapted to the complex features of the sunspot numbers. The target shift size (δ_{tgt}) is for instance estimated on out-of-control series using a formula (defined in (3.7.1)) that is not valid for autocorrelated data.

3. **Non-optimality of the CUSUM for all shift sizes**: The CUSUM chart is designed to optimally detect a target shift size, denoted by δ_{tgt} in Chapter 3. Hence, its allowance parameter is set to $k = \delta_{tgt}/2$ (fixed value). In practice however, the data experience numerous shifts of various sizes. The chart is thus suboptimal for identifying many deviations.

4. **"Exploding" CUSUM values**: The control limits of the CUSUM chart are not directly expressed into the units of the shift sizes (i.e. a control limit equal to 14 does not mean that the shift size needs to be larger than 14 to trigger an alert). Since the CUSUM chart accumulates the deviations of the data over time, the chart statistics can also take very high values and need to be cut at a maximal value ($|C_j^+|$, $|C_j^-| \leq 2L$) to avoid introducing latency in the detection process.

To solve some of the above-mentioned limitations, we develop in the following different procedures based on neural networks for monitoring the sunspot numbers.

The chapter is structured as follows. In Section 4.2, a brief introduction to the main concepts of artificial neural networks is presented. Then, feed-forward neural networks are constructed and trained for the monitoring in Section 4.3. The main results are then showed in Section 4.4. Those include a study of the shapes of the deviations along time. In Section 4.5, more parsimonious recurrent neural networks are also designed for the monitoring. Before concluding, the performances of the different methods (CUSVM and neural network-based control schemes) are then compared on various shift sizes and shapes.

4.2 Overview of neural networks

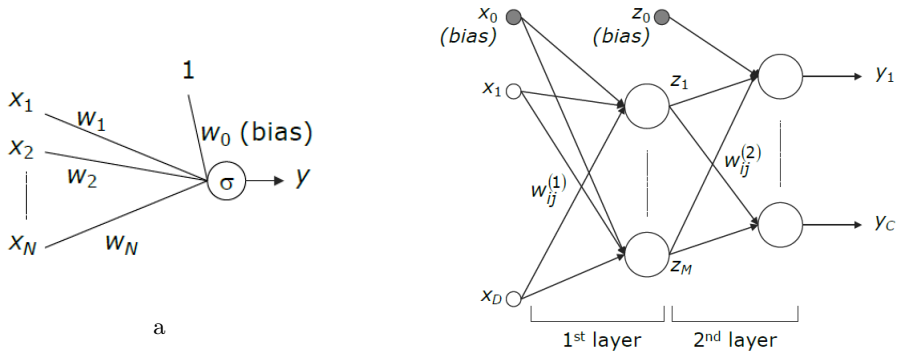


Figure 4.1: (a) Schematic represent of a single neuron or unit. (b) Drawing of a fully-connected neural network with two hidden layers. These figures come from Verleysen and Lee (2017).

Artificial neural network (NN) is a generic term that designates a wide range of models. Those were initially inspired by the neurons in the brain. The artificial neurons, most commonly named units, are the building blocks of the NNs. They are modelled by a non-linear mathematical function σ, called the activation function, which usually takes many inputs $\boldsymbol{x}$ and produces a single output y:

$$y = \sigma(\boldsymbol{w}^T \boldsymbol{x}), \tag{4.2.1}$$

where the ws are the parameters, called weights of the neuron. Most of the time,

the neurons also have an additive bias, w_0. In this case, $\boldsymbol{x}$ should be augmented by one unity, $x_0 = 1$, in (4.2.1). Figure 4.1a shows a simplified representation of a neuron/unit with bias.

The neurons are assembled into interconnected layers that form together a neural network, as schematically represented in Figure 4.1b. A network is typically composed of several layers. The input layer is the first layer of a network. It receives the input data and passes them to the network. The last layer, called the output layer, contains the output of the NN. The format of the output depends on the problem for which the network has been designed. They can be for instance single numbers or vectors, limited to specified range or expressed in probabilities. The network has also one or several hidden layers that contain the artificial neurons.

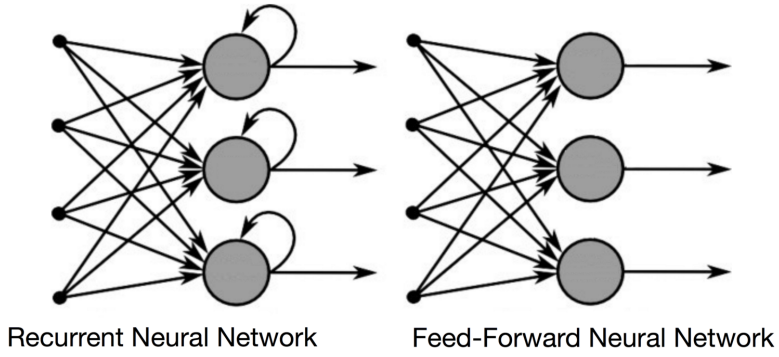


Figure 4.2: Schematic representation of recurrent and a feed-forward neural networks from De Mulder et al. (2014). Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Recurrent-versus-feedforward-neural-network_fig5_266204519 [accessed 29 Apr, 2021].

The networks are used for a large variety of applications such as function approximation or regression, classification (including visual recognition), data processing (such as filtering or compression) or even time-series analysis. In general, there are two main classes of networks: those with feed-forward or feed-back topology. In the feed-forward topology (right-hand side of Figure 4.2), the information flows from the input layer to the output layer in one direction. The outputs of the neurons are thus used as inputs for the neurons in the next layer. Those networks may be composed of different types of layers, each type more adapted to solve a particular category of tasks. The most commonly-used layer is the fully-connected,

which connects every neuron of a layer to every neuron of the next layer. Those layers are deployed in many networks that are constructed for a large variety of applications. There are also layers designed for reshaping or normalizing the input data. Others are constructed for limiting the overfitting by adding noise on the input data or by appending a regularization method such as the dropout [1] to the networks. These regularization methods are however not sufficient to deal with images or videos which involve a large number of parameters (e.g. a single HD image contains millions of pixels). Those objects are better handled by special types of sparse networks: the convolutional neural networks (CNN). They are composed of convolutional and pooling layers, which model the structure of the data with increasing complexity using different types of filters.
In feed-back topology (left-hand side of Figure 4.2), the outputs of a layer can also be used as inputs for the previous layers or for the layer itself. Those networks are called recurrent neural networks. They are composed of at least one type of recurrent layers, which allow the network to have a "memory", i.e. to retain the state of the previous calculations of the network. Hence, the recurrent NNs are mainly used in language processing (Mikolov et al., 2010) and for time-series analysis (Zemouri et al., 2003; Langkvist et al., 2014).

A neural network can be thought of as a non-linear version of the simple linear regression, with typically many more parameters. The non-linearities of the network are introduced through the activation functions (σ). These functions should be differentiable and non-linear, otherwise the network would only be composed of linear functions. It would thus be unable to approximate non-linear problems. The hyperbolic tangent ($\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$) and the sigmoid ($\sigma(x) = \frac{1}{1+e^{-x}}$) activation functions are for instance often used in shallow (i.e. less than three hidden layers) networks. Since their range is limited ($[0, 1]$ for the sigmoid and $[-1, 1]$ for the hyperbolic tangent), networks with a large number of parameters often suffer from the vanishing gradient problem[2] with those functions during the training. Hence, the rectified linear unit ($\sigma(x) = max(0, x)$) is preferred for deep (three or more hidden layers) networks and CNNs. Usually, all hidden layers share the same activation function.
The activation function of the output layer has a different role: it shapes the form of the output. Hence, no activation function ($\sigma(x) = x$) are needed for most regression problems. The binary classification often relies on the sigmoid or the hyperbolic tangent whereas multi-class classification problems depend on the soft-

[1] With the dropout, some random neurons are temporarily disabled during the training to calibrate more sparse networks (the method avoids the co-adaptation of neurons to correct mistakes of others). For more information about the dropout, we refer to Srivastava et al. (2014).

[2] During the training, the parameters of the networks are progressively adjusted by computing the gradient of a loss-function with respect to those parameters. In networks containing a large number of parameters, the gradient often becomes very close to zero, preventing further training. This effect is called the vanishing gradient problem.

max $(\sigma(\boldsymbol{x})_c = \frac{e^{x_c}}{\sum_{j=1}^{K} e^{x_j}}$, where $c = 1, ..., K$ and $\boldsymbol{x} = (x_1, ..., x_K))$. The softmax takes a vector of K inputs, corresponding to the number of different classes, and normalizes them. It returns thus K values between zero and one that sum up to one. Those represent the probability that the output belongs to each class.
For a more systematic overview of the main activation functions, we refer to Lederer (2021).

The construction of a neural network starts by defining its architecture: its number and types of layers, number of neurons in each layer and its activation functions. The parameters of the network should then be adjusted on the data as in a simple linear regression. However, since the networks are typically much more complex and non-linear, the estimation of the parameters is more complicated and requires many iterations.
In this work, the weights ($\boldsymbol{w}$) of the networks are adjusted by supervised learning. With this method, the values of the weights are gradually learned from pairs of examples, which are composed of an input vector $\boldsymbol{x}$ and its corresponding output y (also called label). Those pairs form the training set. The weights are then progressively adjusted, at each iteration, to maximize the prediction results of the network on this set of examples.
The weights of the network are first randomly initialized. The training set is also randomly partitioned into subsets, called batches, of fixed size. Note that a batch contains thus one or several examples, which are in turn composed of an input vector ($\boldsymbol{x}$) and its corresponding output (y). After a batch of examples has been presented to the network, the errors between the predicted outputs and the true values of the outputs are evaluated using a loss function (L). Then, the weights are updated to minimize the loss function:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha \frac{\partial L}{\partial \boldsymbol{w}}\bigg|_{\boldsymbol{w}(t)}, \tag{4.2.2}$$

where α is the learning rate of the algorithm. The computation of the gradient of the loss-function with respect to the weights is the most time-consuming stage of the algorithm. It is done today by a large variety of procedures which mostly rely on the back-propagation algorithm (Rumelhart et al., 1986), with some adaptations. This algorithm efficiently computes the gradient in each layer at the time, starting from the last layer and progressing backward toward the first layer. After updating the weights, the optimization proceeds with the other batches till all training examples have been seen by the network, which completes an epoch. The learning then continues for a pre-defined number of epochs.

The batch size represents thus the number of training examples that are seen by the networks in one iteration. There are three different ways to optimize the weights of the networks: the batch, stochastic and mini-batch modes. In the first mode

(batch gradient descent), the batch size is equal to the length of the training set. Hence, the parameters of the networks are updated at each iteration on the entire training set. In this mode, one iteration corresponds thus to one epoch. This takes time and also uses a lot of memory. Moreover, since the gradient is computed on all training examples, the convergence is stable but may be exposed to local minima. When designed by stochastic gradient descent, the batch size of the networks is on the contrary equal to one, meaning that the parameters are adjusted after each single example. The frequent updates of the method give thus immediate information about the performances and the rate of convergence of the algorithm. In some applications, the stochastic gradient descent can also lead to a quicker convergence since the noisy updating process prevents the networks parameters from being stuck in local minima. In others, the inaccurate estimation of the gradient causes oscillations in the design of the parameters that may overshoot the minima. The numerous iterations required by the method may also slow down the convergence when working with large datasets. In general, an optimization with a large batch size converges thus slowly with accurate estimates of the gradient of the loss function at each step whereas small batch sizes lead to a quick but noisy convergence.
Hence, the mini-batch gradient descent, which combines the advantages of both methods, is the most commonly-used algorithm in practice. In this mode, the batch size is superior to one but inferior to the length of the training set. It is usually selected as a divider of the size of the training set or as a power of two (32, 64, 128, etc.), which depends on the memory requirements of the method. With this mode, the gradient of the loss-function is computed on batches of examples, which accelerates the convergence of the algorithm with respect to the stochastic or batch gradient descent. It also provides more robust convergence than the batch mode, avoiding local minima.
Another important hyper-parameter of the optimization process is the number of epochs, which represents the number of times that the networks see the complete training set. A too small number of epochs usually leads to under-fitting, since the parameters of the networks are progressively adjusted at each iteration. A too high value may on the contrary cause over-fitting. Hence, this number should be carefully selected. This can be done by drawing the learning curve of the networks, i.e. by representing the values of the loss function evaluated on the validation set — another set of examples distinct from the training set— as a function of the number of epochs. The number of epochs may then be selected at the value where the curve stabilises and is not decreased by more epochs. Note that the values of the loss function can even increase with additional epochs, which indicates over-fitting. As this approach is slow, the number of epochs is usually set to a high value and replaced by the early-stopping method. With this method, the training is stopped when the loss-function evaluated on the validation set stabilises.

As seen in (4.2.2), the weights of the network are optimized using a loss-function.

A poor choice of loss-function could therefore lead to bad prediction results, i.e. predictions that do not correspond to the true values of the outputs. Hence, these functions should be selected properly, depending on the problem at hand.
For a regression problem, a typical loss-function is the mean-squared error. The cross-entropy is on the contrary widely used for classification purposes. The binary cross-entropy is designed for binary classification and writes as:

$$BCE(y, \hat{y}) = -\sum_i (y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)),$$

where y_i represents the true label of the example i (either zero or one) and $\hat{y}_i$ is the label predicted by the network. It is most likely a continuous value between zero and one, if an appropriate activation function is selected. The binary cross-entropy is thus computed on each training example i and the results are summed. It reaches a minimum at zero, which corresponds to a situation where all predictions are equal to the true values of the labels. With this formula, the miss-classifications are penalized and those that have the highest probability are also the most penalized.
The cross-entropy can also be adapted for multi-class classification problems through the categorical cross-entropy:

$$CCE(y, \hat{y}) = -\sum_i \sum_{c=1}^{K} (y_{i,c}\ log(\hat{y}_{i,c})),$$

where $\boldsymbol{y}_i$ represents here the true label of the example i in a categorical format: typically a vector of length K with zero everywhere except at one location corresponding to the true class of the example, where it is equal to one. $y_{i,c}$ denotes then a particular element of the vector (either 0 or 1) which corresponds to a single class. Similarly, $\widehat{\boldsymbol{y}}_i$ is the predicted output. It is a vector of length K containing the probabilities that the example belongs to each class. It usually corresponds to the outputs of the softmax activation function. $\hat{y}_{i,c}$ corresponds thus to the probability that the example belongs to a single class. Similarly to the binary cross-entropy, the categorical cross-entropy is computed over all classes and examples. It penalizes the mis-classifications and in particular those that have the highest probability. Note that other loss-functions exist in the literature but are not presented here.

When the training is completed, the prediction performances of the network should be evaluated on a third set of examples — *distinct* from the training and validation sets — called the testing set. Three different sets of data are thus used to construct a neural network. The parameters, i.e. the weights of the network are adjusted on the training set whereas the validation set is used to choose the values of the hyper-parameters such as the number of hidden layers or the number of epochs. Both sets are thus used during the learning of the method. The performances of the network are then evaluated on the testing set, which contains examples that were not used during the learning stage. The prediction capability of the network

on this testing set could be measured using different metrics: the mean-squared error (regression), the mean absolute error (regression), the accuracy (classification), etc. The metrics are thus similar to the loss-functions except that they do not need to be differentiable, contrarily to the loss-function which are used for the training. If the accuracy of the network does not prove to be sufficient, the training sets or the number of epochs could be augmented. The hyper-parameters including the architecture of the network (i.e. types of layers, number of layers and neurons) could also be modified until the desired accuracy is reached.

This closes our short overview of the artificial neural networks. For more information than the simple glance given in this section, we recommend the books of Bishop (2006) and Haykin (2009).

4.3 Feed-forward neural networks for monitoring

Now that we understand the basics about neural networks, we may explain our approach. In this part, we develop two different kinds of feed-forward networks, for regression and classification purposes. The first kind of networks is designed to predict, in a continuous range, the size of the encountered deviations. Those deviations can be close or even equal to zero if the stations are in-control. Then, we construct a second kind of networks to classify the shape of the shifts into three different classes: sudden jumps, drifts and oscillating shifts.
In the following, we first present the quantities that will be monitored by the networks. Then, we explain how the training sets may be constructed, by simulations, to obtain an efficient procedure. After designing the training data, we move on to the description of the networks. The architecture, the loss-function, activation functions and the other parameters of the networks are presented for the regression problem. Then, the networks are described for the classification problem.

4.3.1 Data

Different quantities could be monitored by the networks. In Chapter 3, we apply our monitoring method to the standardised bias, $\hat{\epsilon}_{\hat{\mu}_2}(i,t)$, defined in (3.4.3). The standardisation procedure requires however the estimation of the mean and the variance of the in-control (IC) stations using K nearest neighbours regression method, which is time-consuming. Moreover, the $\hat{\epsilon}_{\hat{\mu}_2}(i,t)$s are visually further away from the traditional scaling-factors (well-known for the data experts) than the unstandardised long-term bias, $\hat{\mu}_2(i,t)$ defined in (3.3.4). The standardisation of the long-term bias was applied earlier to match the definition of the CUSUM chart (see (3.4.4)), which requires that the IC observations have a mean equal to zero

and a variance equal to one. Here however, such a standardisation is not needed. We therefore choose to monitor the (unstandardised) long-term bias, $\hat{\mu}_2(i,t)$.
We still remove the level of the stations before the monitoring though. Those levels typically account for differences in instruments or counting methodologies of the stations. They do not correspond to actual errors but model the intrinsic counting standards of the stations. They should be untangled from the long-term bias, otherwise a station with a more accurate telescope than most stations in the network could systematically be tagged as out-of-control. Note that since we include a level in the sunspot numbers model of (3.3.1), the mean of the long-term bias is (already) close to zero, without standardisation.
In the following, we will monitor the long-term bias at different scales to detect different kinds of deviations. We study in particular the bias at a yearly scale and at 27 days, to allow the comparison with the CUSVM method developed in Chapter 3. As stated previously, the data smoothed on 27 days contains around 933000 values whereas those smoothed on a year are composed of around 1092000 values.

4.3.2 Generation of the sets

Since only a limited number of (unlabelled) observations are available, the training and testing sets are constructed by simulations. As for the SVM procedure, we do not create a validation set here. The hyper-parameters of the networks are adjusted on multiple training and testing sets, also constructed by simulations. Those sets are composed of input-output pairs. The inputs of the network are sequences of data that are simultaneously presented to the networks. In this work, they are randomly selected (with repetition) from the IC biases $\hat{\mu}_2(i_{ic},t)$ by windows of m consecutive values. Then, we add on top of these series various types of deviations with different sizes and shapes. The outputs (also called labels) corresponding to these inputs are the shift sizes for the regression and the shift shapes for the classification problem.
With this simple method inspired by the work of Brian Hwarng (2004) and our previous achievements in support vector machines in Chapter 3, the networks are trained on autocorrelated inputs. The networks are then able to take the autocorrelation of the data into account when making predictions about the characteristics of the shifts, without requiring to build complex time-series models.

The training sets are created as follows. The sizes of the shifts, $\delta \in \mathbb{R}$, are first randomly sampled from a normal distribution centred around zero. The variance of the distribution depends on the variations of the data. It is selected here at one. Then, a series x_{ic} of m consecutive observations is randomly sampled from the IC bias, for each shift size. Note that this procedure is equivalent to generate new series composed of only one block of length m using the block bootstrap.

Three types of general deviations, similar to those used in Qiu et al. (2017) and in Chapter 3, are then artificially constructed on top of the series:

1. jumps: $x(t) = x_{ic}(t) + \delta h(t)$, with $h(t) = \begin{cases} 1 & \text{if } t > \tau \\ 0 & \text{otherwise.} \end{cases}$ and τ in $[0, m]$;

2. drifts with varying power-law functions: $x(t) = x_{ic}(t) + \frac{\delta}{b}(t)^a$, where a is randomly selected in the range $[1, 2]$ and $b = 500$;

3. oscillating shifts with different frequencies: $x(t) = x_{ic}(t) + \delta \sin(\eta\pi t + \phi)$, where η is randomly selected in the range $[\frac{\pi}{2m}, \frac{2\pi}{m}]$ and ϕ in $[0, \frac{m}{4}]$.

Those deviations were designed to visually correspond to the actual deviations observed in the sunspot data. With this training set, the networks appear to correctly predict the shift sizes (regression problem). For the classification however, the networks detect mostly jumps and few drifts and oscillating shifts, which sometimes conflicts with our visual analysis. To correct this effect, we slightly modify the training set for the classification problem. The parameter b of the drifts is set to 100 and the oscillating shifts are defined in a multiplicative framework: $x(t) = x_{ic}(t) \sin(\eta\pi t + \phi)\delta$. Hence, steeper drifts and oscillating shifts with imperfect periodicity that better correspond to the data are constructed for the classification. Those appear to produce better results in practice. Note that the deviations that are simulated here are also slightly different from those used for training the support vector machines in the CUSVM procedure. Indeed, the networks should predict the shapes and sizes of all observations, not only those that are out-of-control and they are trained on unstandardised bias.
In total, 50000 examples are generated, with 40000 (4/5) being used for the training and 10000 (1/5) for the testing. The examples contain deviations that are similar to those of the stations. They are also general to ensure the efficiency of the networks on unseen shifts. After training, the methods may be applied to the real observations for the monitoring. Before being presented to the networks, the data should be prepared into sliding windows of m consecutive observations. Since the networks do not support missing values, they are trained on series x_{ic} which do not contain any missing observation. For the actual monitoring of the series however, the missing observations are imputed. The imputation procedure is the same as those used by SVM: missing observations occurring at the beginning of the input vector are simply replaced by the first valid observation encountered, while the "intermediate" gaps of the input vector are filled by a linear interpolation.

The number of inputs, m, is a parameter which affects the autocorrelation that is learned by the network. It should be sufficiently large to reflect the main part of the autocorrelation of the data without being too large to not slow down the learning. A large number of inputs also leads to a large number of network parameters, which may result in over-fitting.

The number of inputs can be treated as another hyper-parameter of the networks. It could be adjusted to obtain the best prediction results, evaluated using a particular metric. This parameter can also be selected based on the autocorrelation of the data, as follows. The autocorrelation may be computed until a high lag (lag=150 has be chosen for this study) in the IC series. Then, the lag at which the autocorrelation falls below a pre-defined threshold may be computed in those series. m may finally be selected as the mean of those lags. This method is still empirical since it requires the definition of a threshold. It allows us however to associate m to an autocorrelation threshold, which has more physical meanings than simply treating m as another hyper-parameter of the networks. The method (which is more like a rule of thumb) can also be used for different types of autocorrelation. It works for biases smoothed on 27 days and on a year, which experience different autocorrelation structures.

4.3.3 Regression problem

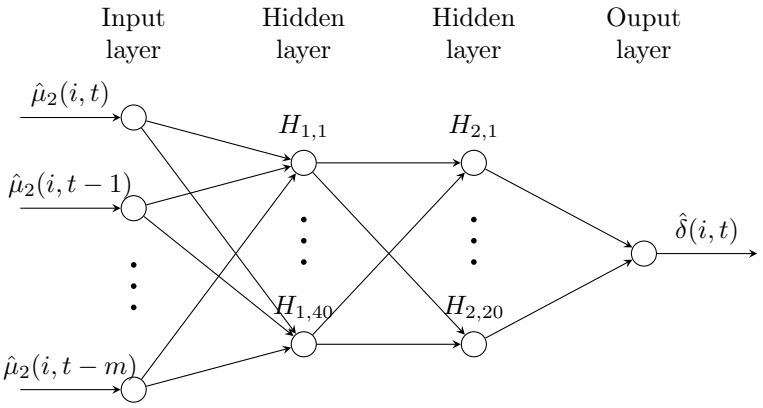
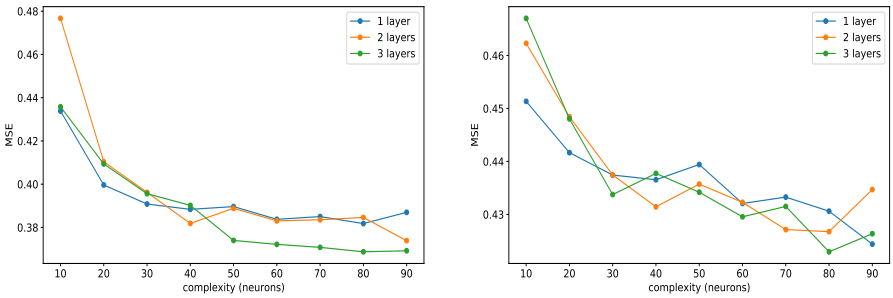


Figure 4.3: Architecture of the feed-forward neural networks designed for regression purpose.

The first neural networks that we construct will do the actual monitoring. They are designed to predict the size of the deviations encountered, which may be close or even equal to zero if the data are in-control. They are thus built for regression purpose. The architecture of those networks is represented in Figure 4.3. The input layer receives the input data and passes them to the network. Those inputs are the bias of the stations, which are thus assumed to be composed of two parts: a stable component $\hat{\mu}_2(i_{ic},t)$ and a deviating part $f(\delta)h(t)$, $\delta \in \mathbb{R}$:

$$\hat{\mu}_2(i,t) = \hat{\mu}_2(i_{ic},t) + f(\delta)h(t), \text{ for } h(t) = \begin{cases} 1 & \text{if } t > \tau \\ 0 & \text{otherwise.} \end{cases} . \tag{4.3.1}$$

The input layer has the same size as the input vector, which is fixed here at $m = 50$ for the high-frequency monitoring and at $m = 100$ for the low frequency monitoring. Those values correspond to the lag at which the autocorrelation falls below a threshold of 0.2 (for $m = 50$) and 0.75 (for $m = 100$), since the data that are smoothed on a year are more autocorrelated than those smoothed on 27 days. The networks have also a single output neuron corresponding to the predicted size of the shifts (denoted $\hat{\delta}$).



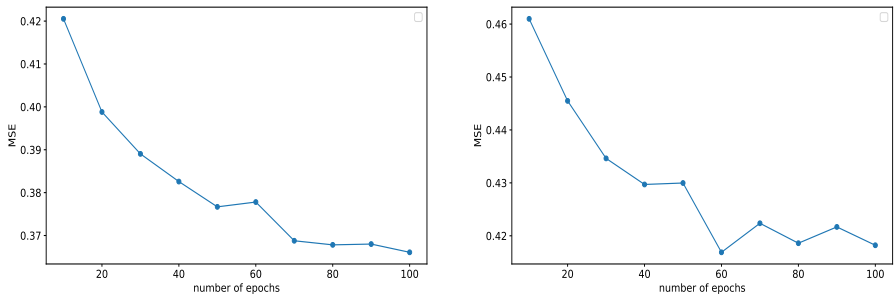
a: Number of hidden layers and neurons (data smoothed on 365 days)

b: Number of hidden layers and neurons (data smoothed on 27 days)

Figure 4.4: (a) The mean-squared error (MSE) as a function of the complexity of the network for different numbers of hidden layers. The complexity corresponds to the number of neurons in the first layer ; every subsequent layer has half the number of neurons as the preceding layer. The network is applied on data smoothed on a year. (b) Same figure for data smoothed on 27 days.

Many different numbers of hidden layers and neurons may then be selected. Unfortunately, there are no magic rules for choosing them. The best numbers depend in a complex way on the size of the training set, the activation functions, the complexity of the task that the network must approximate, the number of the inputs and outputs, etc. Usually, those numbers are thus selected with a method that is similar to those used to select the number of epochs. A simple network, with one hidden layer and only few neurons, is first designed. Then, the complexity of the network is progressively increased until the testing error stops decreasing with higher numbers of neurons/layers or when a sufficient level of precision is reached. In other words, the architecture of the network may be selected to be the simplest one, which achieves at the same time a sufficient degree of matching (evaluated using a chosen metric) between the predictions and the true values of the outputs. This method is illustrated in Figure 4.4, where the mean squared error (MSE) is represented on the testing set as a function of the number of neurons, for one, two and three hidden layers. As can be seen, the MSE has a local minima around 40

with two hidden layers, both for data smoothed on 27 days and on a year. Networks with two hidden layers containing respectively 40 and 20 neurons were then selected, since the gain in performance obtained with three hidden layers and 50 (or more) neurons is small compared to the costs, in time and complexity. Note that since the networks are designed on sets that are constructed by simulations, results vary from one training to another. The MSE values represented in Figure 4.4 are thus the mean of the values obtained over ten trainings.
With this architecture, the prediction results of the networks (which are evaluated using the MSE) fulfil our expectations. The networks also reveal no sign of over-fitting. They contain a total of respectively 4881 and 2881 trainable parameters for the low and high frequency monitoring. Note that the networks have different numbers of parameters since they operate on input vectors of different sizes (m). We use the sigmoid activation function in the hidden layers whereas no activation function is specified for the output layer, as proposed in Brian Hwarng (2004).



a: Number of epochs (data smoothed on 365 days)

b: Number of epochs (data smoothed on 27 days)

Figure 4.5: (a) The MSE as a function of the number of epochs, for a network with two hidden layers of respectively 40 and 20 neurons. The network is applied to data smoothed on 365 days. (b) Same figure for data smoothed on 27 days.

Then, we train the networks on 30 epochs using an optimizer based on the Adam algorithm (Kingma and Ba, 2015) with a batch size equal to 50. The MSE is represented in Figure 4.5 as a function of the number of epochs. It was selected here at 30 for saving some computational time, although the number of epochs could be increased without risk of over-fitting. The Adam optimization method was also selected, to speed the convergence of the training. Without entering in too much details, Adam combines the mini-batch gradient descent algorithm presented in the previous section with two other methods: momentum (Polyak, 1964) and root mean square propagation (RMSProp) (Hinton et al., 2012). In the momentum method, the weights are updated by a linear combination of both the

gradient of the loss-function at the current time and the gradient at the previous time. This method accelerates the gradient descent convergence since it favours the weights updates that are in the same direction, preventing oscillations. In the RMSProp algorithm, the learning rate is also adapted for each weight. This method further increases the convergence of the algorithm since it allows large learning rates initially. Those progressively decrease as the weights are updated to minimize the risk of overshooting the minima. The learning rate is selected initially at $\alpha = 0.001$ [3]. Note that the training of the networks are done only on CPUs using the version 3.7.9 of Python with the package `keras` (Chollet et al., 2015). During the training and the testing, the prediction errors are evaluated by the MSE (the MSE serves thus both as the loss-function and metric). With these parameters, the networks achieve a MSE around 0.41 and 0.44 on the testing set, for respectively the low and high frequency monitoring. Those values correspond to a mean absolute percentage error (MAPE), defined in Section (3.4.3), of around 45 and 56 respectively. They are thus higher than the MAPE values obtained in Chapter 3, which were equal to 26 and 32 respectively. This is expected since the networks should estimate the size of the deviations for all data (IC and OC), where the smallest shifts are often the most difficult to detect. Moreover, the MAPE criterion takes high values for errors that are close to zero, by construction, see (3.4.9). These values are nevertheless satisfactory for our monitoring problem.

4.3.4 Classification problem

We design a second type of neural networks, for classification purpose. These networks, represented in Figure 4.6, aim at predicting the shape of the input data among three different classes: (1) sudden jumps, (2) more progressive drifts and (3) oscillating shifts. The size of the input layer is selected at $m = 100$ and $m = 50$ for the low and high frequency monitoring, as previously. The networks have also three output neurons since there are three different classes. Many designs are then possible for the hidden layers. After few tests, we select networks with one fully-connected layer containing 40 neurons, to obtain an accuracy at least equal to 90% on the testing set. As can be seen in Figure 4.7, such a degree of accuracy is determined by the monitoring of data smoothed on 27 days, which experience variations on a larger scale than those smoothed on a year. The networks contain thus a total of respectively 4163 and 2163 trainable parameters for the low and high frequency monitoring. We use the sigmoid activation function in the hidden layer and the softmax for the output layer. The outputs of the network correspond then to the probabilities that the input belongs to each different class.

The parameters of the networks are then estimated using the RMSprop algorithm (i.e. mini-batch gradient descent with adaptive learning rate), a widely used opti-

[3] This is the default value for the Adam optimizer with the package `keras` (Chollet et al., 2015).

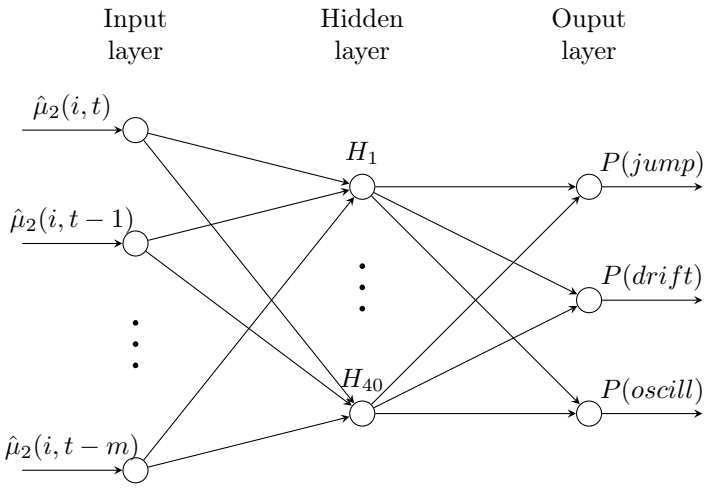
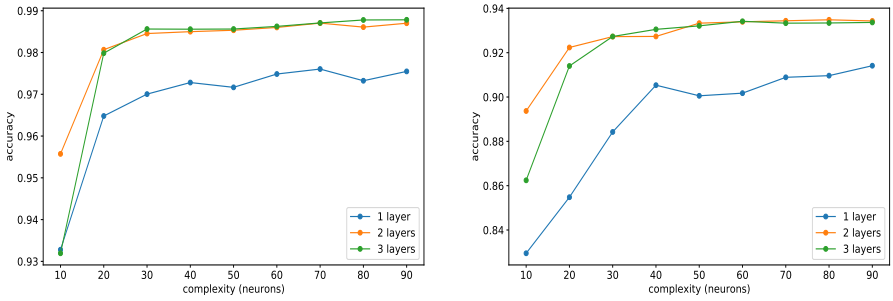


Figure 4.6: Architecture of the feed-forward neural networks designed for classification purpose. The output of the networks corresponds to the probabilities that the input belongs to each class: jump, drift and oscillating shift.

mizer for classification purpose, with an initial learning rate of 0.001. The learning is performed on 30 epochs with a batch size equal to 50. During the training,



a: Number of hidden layers and neurons (data smoothed on 365 days)

b: Number of hidden layers and neurons (data smoothed on 27 days)

Figure 4.7: (a) The accuracy as a function of the complexity of the network for different numbers of hidden layers. The complexity corresponds to the number of neurons in the first layer ; every subsequent layer has half the number of neurons as the preceding layer. The network is applied on data smoothed on a year. (b) Same figure for data smoothed on 27 days.

the prediction errors are evaluated by the categorical cross entropy loss-function whereas the prediction results are measured by the accuracy (metric) in the testing step. With these parameters, the networks achieve an accuracy of around 96% and 90% respectively, for the low and high frequency monitoring.

4.4 Results

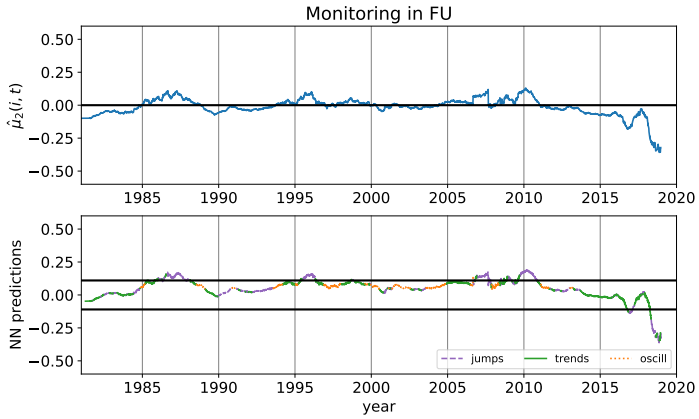
In this section, we apply the neural networks to the long-term bias, $\hat{\mu}_2(i,t)$, for predicting the sizes and shapes of the encountered deviations. Figures 4.8 and 4.9 show respectively the networks applied to biases smoothed on a yearly scale and on 27 days. As can be seen, the predictions of the shift sizes closely correspond to the pace of the long-term bias. The networks identify many types of deviations that vary over a wide range of values. The downward drift that appears in MT after 1998 (related to the installation of a CCD camera) as well as those that occurred after 2014 in FU (associated to the health condition of the observer) are well detected by the networks. The sudden jump of 1999 in UC and those that appeared around 2007 in FU are also well identified.

The outputs of the regression networks correspond to the predicted sizes of the shifts. They do not contain however information about the status (in-control or out-of-control) of the stations and cannot provide alerts. To that end, we add on top of the networks different procedures, which can warn us when the stations are out-of-control. Those are described below.

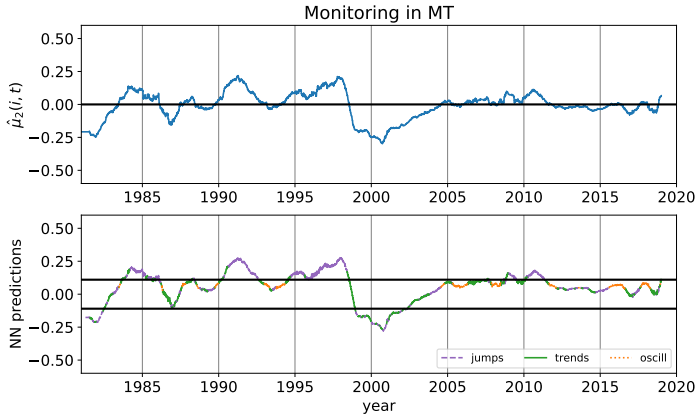
4.4.1 Classifier with four classes

One of the most simple way to give alerts is to modify the classifier previously described to include four classes instead of three: jumps, drifts, oscillating shifts and a last class which corresponds to in-control data. A series is thus considered to be out-of-control with this classifier when a deviation of one of the three first classes is identified. This new classifier could be easily trained using the sets described in Section 4.3.2, with an additional type of deviation, actually composed only of in-control data:

1. jumps: $x(t) = x_{ic}(t) + \delta h(t)$, with $h(t) = \begin{cases} 1 & \text{if } t > \tau \\ 0 & \text{otherwise.} \end{cases}$ and τ in $[0, m]$;
2. drifts with varying power-law functions: $x(t) = x_{ic}(t) + \frac{\delta}{b}(t)^a$, where a is randomly selected in the range $[1, 2]$ and $b = 500$;



a

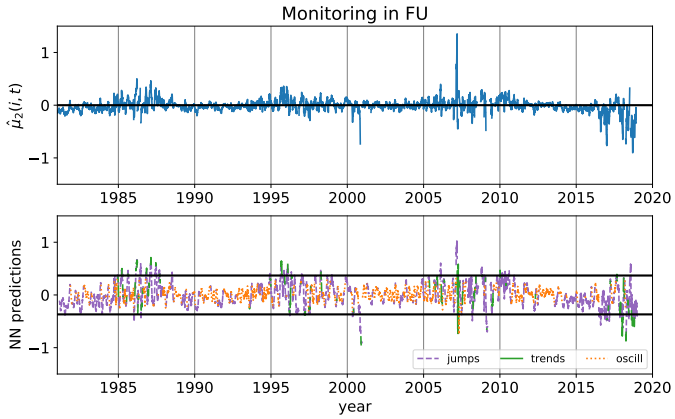


b

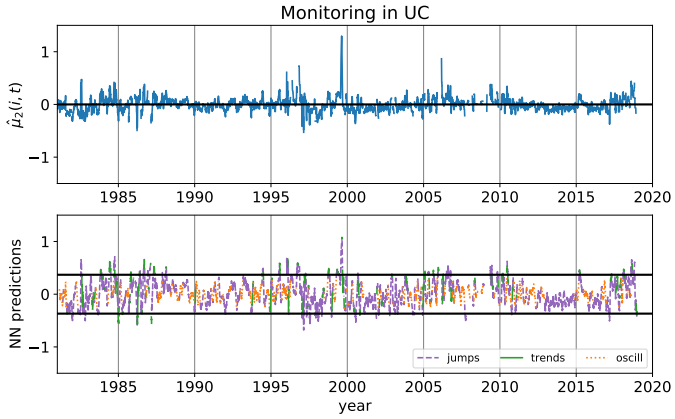
Figure 4.8: (a) Upper panel: the long-term bias ($\hat{\mu}_2(i,t)$) smoothed on a year from observer Fujimori-san (FU) in Japan over 1981-2019. Lower panel: the shift sizes and shapes predicted by the neural networks. The cutoff values ($\pm\delta_{cut}$, $\delta_{cut} = 0.11$) are represented by the two horizontal thick lines. (b) Similar figure for the observatory of Mitaka (MT) in Japan over the same period.

3. oscillating shifts with different frequencies and phases: $x(t) = x_{ic}(t)+\delta \sin(\eta\pi t + \phi)$, where η is randomly selected in the range $[\frac{\pi}{2m}, \frac{2\pi}{m}]$ and ϕ in $[0, \frac{m}{4}]$;

4. in-control: $x(t) = x_{ic}(t)$.



a



b

Figure 4.9: (a) Upper panel: the long-term bias ($\hat{\mu}_2(i,t)$) smoothed on 27 days from the station FU over 1981-2019. Lower panel: the shift sizes and shapes predicted by the neural networks. The cutoff values ($\pm\delta_{cut}$, $\delta_{cut} = 0.37$) are represented by the two horizontal thick lines. (b) Similar figure for the observatory of Uccle (UC) in Belgium over the same period.

With this method however, it is not possible to calibrate the classifier at a desired rate of false positive. Hence, we propose two others procedures in the following, which are more complex but allow us to control the rate of false positives. Those methods are composed of two distinct stages: (1) estimating the characteristics

of the shifts and (2) using a dedicated procedure to give alerts based on those estimations. Note that we thus invert here the two main steps of the CUSVM procedure, where the CUSUM chart first detects the out-of-control situations and the SVM methods estimate in a second time the characteristics of the deviations.

4.4.2 Simple cut-off values

In a first procedure inspired by the Shewhart chart (presented in Section 3.2.2), we determine simple cut-off parameters, denoted δ^+_{cut} and δ^-_{cut}, to discriminate between in-control (IC) and out-of-control (OC) data. These cuts-off are similar to the control limits of classical charts. A station is considered to be out-of-control if its shift size exceeds the cuts-off: $\hat{\delta}(i,t) > \delta^+_{cut}$ or $\hat{\delta}(i,t) < \delta^-_{cut}$, where $\hat{\delta}(i,t)$ represents the predicted shift size at time t, in station i. We assume in the following that $\delta^+_{cut} = -\delta^-_{cut} = \delta_{cut}$. The cut-off ($\delta_{cut}$) is then adjusted, after the training of the networks, using a searching algorithm based on the IC average run length. This method is similar to those used to calibrate the control limits of the CUSUM chart and is explained below.
After fixing an initial value for the cut-off, new series are generated from the IC data using the block bootstrap with a block length equal to m. These series are then arranged into moving windows of m observations and are fed to the networks which predict the size of the shifts. The run lengths are defined as the number of observations obtained until a shift size exceeds the cut-off value. The average run length is finally computed as the mean of the run lengths over a large number of runs (2000 runs are selected in this chapter). The cut-off is then progressively adjusted by bisection searching, i.e. by sub-divisions of the searching interval in half. The algorithm stops when the IC average run length reaches a pre-defined value at the desired accuracy.
Using this procedure, we calibrate the cut-off value at an average run length of 200 with an accuracy equal to $\rho = 2$. We obtain a value of $\delta_{cut} = 0.37$ and $\delta_{cut} = 0.11$ for the high and low frequency monitoring respectively. Those are represented in the lower panels of Figures 4.8 and 4.9. Note that, contrarily to the control limits of the CUSUM chart, the cut-off values are directly expressed in the units of the shift sizes.

With this method, the data are assumed to be OC when the shift size exceeds the cut-off value, while with the CUSUM chart a station is in alert when the *cumulative sum* of its deviations surpasses the control limits. Hence, the CUSUM chart is expected to be more sensitive to small and persisting shifts.
In an attempt to correct this effect, we may also adapt the cut-off with a heuristic decision procedure proposed in Brian Hwarng (2004). This method relies on two cuts-off, δ^1_{cut} and δ^2_{cut}, with $\delta^1_{cut} < \delta^2_{cut}$. A station is considered to be out-of-control with this method if (a) $\hat{\delta}(i,t) > \delta^2_{cut}$ or (b) if $\{\hat{\delta}(i,t), \hat{\delta}(i,t-1), ..., \hat{\delta}(i,t-n_{cut})\} >$

δ^1_{cut}. Hence, (a) an alert is immediately triggered for medium to large shifts (that are larger than δ^2_{cut}). (b) The series can also be in alert when a small deviation lasts n_{cut} units of time, where n_{cut} is a tunable parameter. The method is then supposed to better identify small and persisting shifts but may take more time to detect medium to large shifts, especially if they are oscillating. The main focus of this heuristic decision procedure is thus to identify small jumps and drifts. Note that δ^1_{cut} could also be used as the warning limit of the method, i.e. a value above which the series is considered to be in a warning state, yet not out-of-control.
The values of δ^1_{cut} and δ^2_{cut} can be adjusted using a searching algorithm similar to those used to calibrate δ_{cut}. For a pre-specified value of n_{cut}, δ^1_{cut} and δ^2_{cut} can be adjusted alternately, one after the other, until a pre-defined value for the IC average run length (ARL_0) is reached at the desired accuracy [4]. Here, we select a value of $ARL_0 = 200$ with an accuracy equal to $\rho = 2$. For $n_{cut} = 30$, we obtain $\delta^1_{cut} = 0.21$ and $\delta^2_{cut} = 0.08$ for the low-frequency monitoring. For the high-frequency monitoring, we compute that $\delta^1_{cut} = 0.83$ and $\delta^2_{cut} = 0.32$, for $n_{cut} = 5$. Examples of such heuristic decision procedure are shown in Figure 4.10.

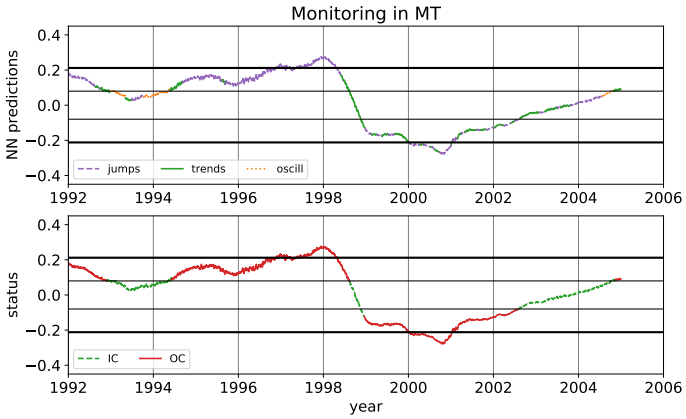
4.4.3 Adaptive CUSUM chart

The last procedure that we develop for giving alerts is based on an adaptation of the CUSUM chart. In general, the CUSUM chart is particularly efficient to detect small shifts since it agglomerates all deviations that are superior to a threshold, previously denoted by k. As explained in Chapter 3, this parameter should be set to $k = \delta/2$ to allow an optimal detection of shifts of size δ by the chart. In practice however, it is often difficult to correctly estimate δ. We use an optimal formula (see (3.7.1)) valid for iid normal data for this purpose in Chapter 3. This formula is thus not valid for the sunspot numbers, which are autocorrelated and non-normally distributed. Moreover, the formula allows us to obtain a rough estimate of δ for all stations on the period studied, while δ varies in each station and at every time-step. Hence, having already developed efficient networks to estimate the size of the shifts, we propose to use an adaptive CUSUM chart (Qiu, 2013, Section 4.5.2) to give the alerts. This chart is similar to the classical CUSUM, except that the threshold k is optimally adjusted for each new observation as a function of the estimated shift size:

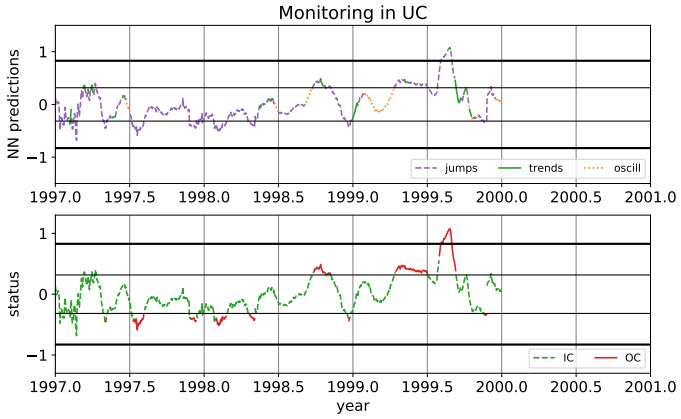
$$\begin{aligned} C_j^+ &= max(0, C_{j-1}^+ + \hat{\mu}_2(i,t) - \hat{\delta}(i,t)/2) \\ C_j^- &= min(0, C_{j-1}^- + \hat{\mu}_2(i,t) + \hat{\delta}(i,t)/2), \end{aligned} \tag{4.4.1}$$

where $j \geq 1$, $C_0^+ = C_0^- = 0$ and $\hat{\delta}(i,t)$ is the shift size predicted by the networks in station i at time t. This chart gives an alert if $C_j^+ > L(i,t)$ or $C_j^- < -L(i,t)$,

[4] Note that we can also fix a value for δ^2_{cut} and calibrate δ^1_{cut} until the pre-specified ARL_0 is achieved.



a



b

Figure 4.10: (a) Upper panel: the shift sizes and shapes predicted by the neural networks on data smoothed on 365 days from the station MT in Japan over 1992-2005. Lower panel: the status (IC or OC) of the station. The cutoff values ($\pm\delta^1_{cut}$ and $\pm\delta^2_{cut}$) for $n_{cut} = 30$ are represented by the horizontal thick lines. (b) Similar figure for data smoothed on 27 days from the station UC in Belgium over 1997-2000 ($n_{cut} = 5$).

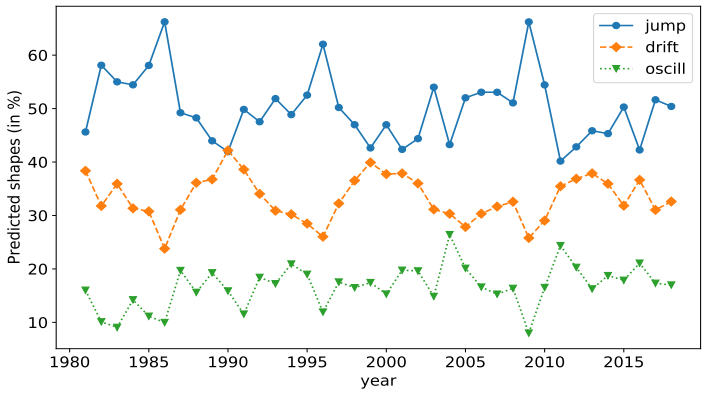
where $L(i,t)$ denotes the control limit at time t in station i. Since $\hat{\delta}(i,t)$ are continuous, the previous scheme may be viewed as a combination of an infinity of charts, with parameters $L(i,t)$ and $\hat{\delta}(i,t)$. To simplify its design, we assume that $L = L(1,1)k(1,1) = ... = L(i,t)k(i,t) = ... = L(N,T)k(N,T)$, as proposed

by Lorden (1971). The control limit L may then be adjusted using a bisection searching method similar to Algorithm 1, to reach a pre-defined value of ARL_0. With this method, the alerts are given if $|C_j^+|$, $|C_j^-| > L/k(i,t)$, which is equal to $|C_j^+|$, $|C_j^-| > 2L/\hat{\delta}(i,t)$.

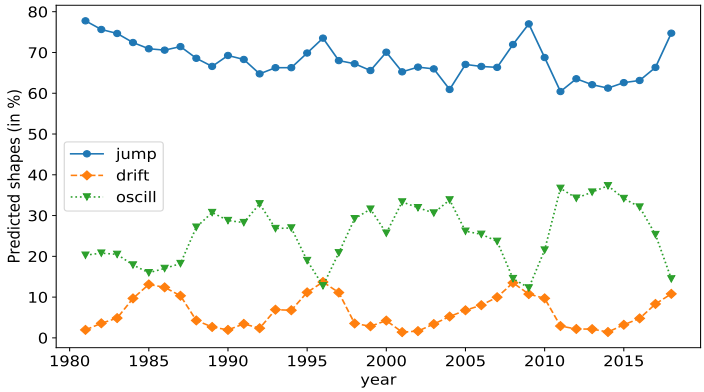
4.4.4 Analysis of the shapes of deviations

The primary aim of developing a classifier is to examine and help identify the root-causes of the deviations. As can be seen in previous figures, the shapes of the deviations are complex to analyse but appear to correspond visually to what we expect. The deviations around 2007 in FU and 1999 in UC are identified for instance as jumps whereas the shifts around 1998 in MT and after 2014 in FU are classified as (mostly) drifts.
The classification results may also be used to identify common deviating patterns in all stations. Figure 4.11 shows the percentages of predicted shapes for each year. Those percentages are calculated as the mean of the percentages of all stations in the network. As can be seen, the jumps are more numerous than the other types of deviations, both at a yearly and 27 days scales. The jumps are also more frequent during solar minima. This effect is visible at both scales, but is however more pronounced at a yearly scale. The low-frequency deviations contain least oscillations, which appear to be often localised outside of solar minima. They include more frequent trends that also mainly occur outside of minima. This may indicate that the observers stay more or less constant, which is translated by oscillations of small amplitudes, outside of minima. They can also experience trends related to changes of instrument, observing method or observers whereas they experience more jumps at minima. This may reflected the tendency of some observers to over-scrutinize the Sun in such periods. The $\hat{\mu}_2(i,t)$ may also amplify the deviations at minima since they are often computed on few values around these periods ($\hat{\mu}_2(i,t)$ are not defined when $M_t = 0$, see (3.3.1)).
The high-frequency deviations contain few drifts that mainly occurred during solar minima and slightly more oscillations that happen predominantly outside of minima. This may reflect the fact that the observers make regular errors on a rather short-term basis in one direction than another, resulting in small oscillations. Those same observers make on the contrary substantial mistakes around solar minima, which can last for several days and create rapid trends. Those results show that it might be interesting to study the shapes of the deviations along time since they appear to dependent strongly on the solar cycle.

We analyse in the same way the shapes of the deviations predicted by the support vector machines (SVM) developed in Chapter 3. As for the networks, the most common shapes are the jumps and those are more frequent around solar minima. The oscillations however are almost never detected. When there is no jump, a drift



a: Predicted shapes for low-frequency deviations



b: Predicted shapes for high-frequency deviations

Figure 4.11: (a) Types of deviations that occurred in the long-term bias ($\hat{\mu}_2(i,t)$) smoothed on a year as a function of time. The shapes of the deviations are expressed in percentages for each year over 1981-2019. The percentages of the three types of deviations (jumps, drifts and oscillating shifts) sum thus to 100%. They are computed as the mean of the percentages obtained in all stations of the network. (b) Similar figure for the shapes of the deviations that occurred in the bias smoothed on 27 days.

is then most of time identified, which gives less insights about the nature/causes of the deviations. This may be explained by the fact that the CUSUM chart is not much effective to detect oscillating shifts contrarily to NN-based control schemes,

as will be seen later in Section 4.6. Note that the SVM procedures are trained to detect the deviations when the CUSUM chart is in alert. To do this analysis however, we apply the SVM on all data, deviating or not. The SVM predictions can thus also be unreliable for non-deviating data.
Without further analyses, we cannot exclude the possibility that the networks have somehow learned the underlying cycle of the data and make wrong associations between shapes and parts of the solar cycle. It seems however unlikely since the training set is large, generated by simulation and contains windows of maximum hundred observations. Moreover, we expect that different types of errors are made at and outside of minima. With no evidence otherwise, we can thus assume that the networks correctly predict the shape of the deviations. They may provide useful information for analysing the causes of the shifts that should be further investigated.

The automated prediction of the shapes of the deviations suffers however from some limitations, which may be at least partially corrected in future works. Those limitations are not restricted to the neural networks but also affect the predictions of the SVM. First, the automated methods do not identify the shape of most of the deviations that are longer than the window lengths. These lengths are limited here to $m = 100$ and $m = 50$ for respectively the high and low frequency monitoring. Similarly, they are restricted to $m = 80$ and $m = 70$ for SVM. To correct this effect, the length of the windows may be extended at the expense of a larger number of parameters in the methods. A network with a larger number of inputs could for instance be developed but it will require regularization methods to avoid overfitting the data and will take more time to be trained. Another possibility to analyse long deviations is to smooth the bias at a even larger scale than one year. The bias at 365 or 27 days may also be smoothed by a non-parametric kernel function to remove all deviations that live on shorter scales. The classifiers might then perform better to predict the shape of the deviations that last longer than the window length since they will not be "distracted" by short-lived shifts.
Second, the methods learned to classify the shapes of the deviations into three different classes. Deviations of more than one class can however happen inside the same window length. A jump and an oscillating shift can for instance happen consecutively in a single input vector. The classes are also general and cover a large variety of shapes but many types of shifts are still not included in them. For example, polynomial shifts of order superior than 2 or exponential shifts are not part of the training set. To solve these issues, the classifiers (either the neural networks or the support vector machines) can be trained on more types of shifts and even combinations of several types of shifts. The construction of the training sets will be more complicated however and the design of the methods will also take more time. For these reasons, we focus here on only three different shapes.

4.5 Recurrent neural networks for monitoring

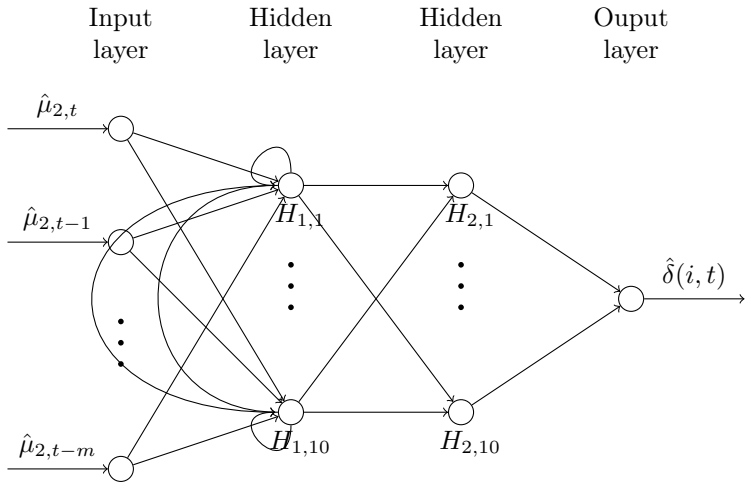


Figure 4.12: Architecture of the recurrent neural networks designed for regression purpose.

As stated in the introduction of the section, there are two main types of neural networks: feed-forward and recurrent (or feed-back) NNs. Previously, we designed feed-forward networks for monitoring the unstandardised bias. Here however, we focus on recurrent networks which are, by construction, better suited for time-series analyses. We construct in the following simple recurrent networks, only composed of fully-recurrent layers as inspired by the work of Pacella and Semeraro (2007). More complex recurrent networks exist however in the literature such as the long-short term memory (LSTM) (Hochreiter and Schmidhuber, 1997) or the gated recurrent unit (GRU) (Cho et al., 2014) networks. They are composed of different gates (layers) that control the flow of information in and out of a cell, which stores the information that the networks have chosen to retain. Those networks were initially introduced to fix the problem of the vanishing gradient that prevents the modelling of long-term dependencies in simple recurrent networks. This problem is solved in LSTM or GRU networks since the information contained in the cell flows unchanged along time, except when it is accessed or modified by the gates. A nice introduction to the recurrent networks, including the LSTM can be found in the work of Schmidt (2019). Although the LSTM or GRU networks are allowed to model longer autocorrelation structures than the simple recurrent networks, we do not implement the former as the latter achieve sufficient performances for our monitoring problem and are simpler to understand.
The simple recurrent networks that are constructed here are designed with the same training sets as those previously described in Section 4.3. We first construct

networks for the regression task. Those are represented in Figure 4.12. The inputs of the networks have the same size as before: $m = 100$ and $m = 50$ respectively, for the low and high-frequency monitoring. The networks are then composed of two layers: the first one being a fully-recurrent layer of 10 neurons and the second one a fully-connected layer of 10 neurons as well. The hyperbolic tangent activation function, which is commonly used in recurrent networks, is also selected in the first layer followed by the sigmoid for the second layer. Note that we could have chosen the hyperbolic tangent activation function in both layers, without practically changing the results. The outputs of the networks correspond then the predicted sizes of the shifts. Those networks have 1231 and 731 trainable parameters respectively, for the low and high frequency monitoring (against 4881 and 2881 for the feed-forward regression networks). This architecture was selected, after few tests, to achieve a precision (evaluated with the MSE) superior or equal to those previously obtained with the feed-forward regression networks. The recurrent networks were then trained using the same settings (optimizer, initial learning rate, number of epochs, loss function and batch size) as before. They achieve a MSE of respectively 0.37 (MAPE of 45) and 0.43 (MAPE of 54) on the testing set for the low and high frequency monitoring (versus a MSE of 0.41 and 0.44 for the feed-forward networks). In total, the recurrent networks reach thus a similar precision than the feed-forward networks with fewer parameters.
For the classification task, networks composed of a single fully-recurrent layer with 10 neurons and the hyperbolic tangent activation function were able to achieve similar degrees of accuracy than the feed-forward classification networks. Those recurrent networks have a total of 1143 and 643 parameters for respectively the low and high frequency monitoring (against 4163 and 2163 for the feed-forward classification networks). After training, they reach an accuracy of respectively 95% and 91% on the testing set for the low and high frequency monitoring, which is thus similar to the 96% and 90% obtained with the feed-forward networks.

We can thus obtain a similar accuracy with more parsimonious recurrent networks than with feed-forward networks. This can be understood by the following argumentation from Connor et al. (1992).
(a) A non-linear autoregressive (AR) model of order p writes as:

$$x_t = h(x_{t-1}, x_{t-2}, ..., x_{t-p}) + e_t, \tag{4.5.1}$$

where the function h is unknown and smooth and e_t represents random noise at time t that are independent of x_t. The best prediction of x_t is given by its conditional mean given $x_{t-1}, ..., x_{t-p}$. It can be approximated by a feed-forward neural network with one hidden layer:

$$\hat{x}_t = \hat{h}(x_{t-1}, x_{t-2}, ..., x_{t-p}) = \sum_{i=1}^{I} W_i \ \sigma(\sum_{j=1}^{p} w_{ij} x_{t-j}), \tag{4.5.2}$$

where σ is the activation function of the layer such as the sigmoid or the hyperbolic tangent and I denotes the number of hidden neurons. The matrix w contains the weights of the neurons. It is lower diagonal and allows thus no feedback. The parameters w and W are both learned by the network on the training examples.
(b) In addition, a non-linear autoregressive and moving average (ARMA) model of order p and q writes as:

$$x_t = h(x_{t-1}, x_{t-2}, ..., x_{t-p}, e_{t-1}, e_{t-2}, ..., e_{t-q}) + e_t. \tag{4.5.3}$$

The best prediction of x_t can now be approximated by a recurrent neural network:

$$\begin{aligned} \hat{x}_t &= \hat{h}(x_{t-1}, x_{t-2}, ..., x_{t-p}, \hat{e}_{t-1}, \hat{e}_{t-2}, ..., \hat{e}_{t-q}) \\ &= \sum_{i=1}^{I} W_i \ \sigma(\sum_{j=1}^{p} w_{ij} x_{t-j} + \sum_{j=1}^{q} w_{ij}^{'}(x_{t-j} - \hat{x}_{t-j})), \end{aligned} \tag{4.5.4}$$

where the random noise are estimated by $\hat{e}_{t-j} = x_{t-j} - \hat{x}_{t-j}$ and the $\hat{x}_{t-j}$, for $j = 1, ..., q$, are the results of the previous computations of the network. This model can be viewed as a special case of a fully-recurrent network:

$$\hat{x}_t = \sum_{i=1}^{I} W_i \ \sigma(\sum_{j=1}^{n} w_{ij}^{''} x_{t-j}), \tag{4.5.5}$$

where the matrix $w^{''}$ is now complete and allows feed-back.
In argument (a), a feed-forward network with one hidden layer is associated to a non linear AR model while in argument (b), a recurrent network approximates a non linear ARMA model. As it is well known for those who studied time-series, an ARMA modelling is a parsimonious alternative to autoregressive (AR) or moving average (MA) models of high orders. A recurrent network have thus similar parsimonious advantages with respect to feed-forward networks as the ARMA models have over AR models for some particular time-series.
In this work, the networks contain one or two hidden layers. They are also designed to predict the shape and size of the deviations that affect the series, not to predict the future values of the series. The analogy between our networks and non linear AR and ARMA models may however be extended. Since our data have complex autocorrelations that cannot be modelled by sparse AR or MA models, it explains why the recurrent networks perform equally or even better than the feed-forward networks with fewer parameters. The recurrent networks then offer a parsimonious alternative to monitor the sunspot data.

4.6 Comparison of the different monitoring methods

We previously developed feed-forward and recurrent neural networks for the monitoring. In this part, we compare their performances to those of the CUSVM method developed in Chapter 3. We first study the detection power of the methods, i.e. their capacity to give alerts when a deviation occurs in the data. To this end, we compute the out-of-control average run length, denoted by ARL_1, for various sizes of the shifts with the different procedures. Then, in a second stage, we also compare the performances of the methods to estimate the sizes of the shifts. The first analysis aims thus at comparing the mechanisms that give alerts such as the CUSUM chart, adaptive CUSUM chart, simple cut-off values of the networks or the classifier with four classes. The second study focusses on the contrary on evaluating the predictions of the regression networks and support vector regressor (SVR).
In the following, we describe simulations and results for monitoring the sunspot data smoothed on a yearly scale. Similar conclusions can however be drawn with data smoothed on 27 days.

4.6.1 Detection power

In this first subsection, we aim at comparing the detection power of the recurrent and feed-forward networks as well as the CUSVM method. Those procedures have however fundamental differences. To allow a fair comparison, we first thought of monitoring the standardised bias, $\hat{\epsilon}_{\hat{\mu}_2}(i,t)$ defined in (3.4.3), with all methods since the CUSVM was designed to work on standardised quantities in Chapter 3. Although we observe that changing the scale of the deviations do not affect much the performances of the networks when measured with a scale-independent metric such as the MAPE, monitoring standardised bias $\hat{\epsilon}_{\hat{\mu}_2}(i,t)$ instead of unstandardised bias $\hat{\mu}_2(i,t)$ diminishes their performances. We observe on average that the MAPE values increase by around 20 when monitoring the $\hat{\epsilon}_{\hat{\mu}_2}(i,t)$s. This is probably due to the fact that the standardisation patterns ($\mu_0(t)$ and $\sigma_0(t)$) vary locally in time. The standardisation process may then change the structure of the data. Since the hyper-parameters of the networks (architectures, number of epochs, etc.) have not be selected for monitoring standardised bias, the networks are thus less efficient for that purpose.

Hence, we decide to monitor the standardised bias $\hat{\epsilon}_{\hat{\mu}_2}(i,t)$ only with methods that are based on the CUSUM chart. Those methods correspond to the feed-forward network associated with the adaptive CUSUM (NN-ACUSUM), the recurrent network with adaptive CUSUM (RNN-ACUSUM) and the CUSVM method. Hence,

we retrain the previously described recurrent and feed-forward neural networks designed for regression purpose on the standardised bias. All training parameters remain identical except one: the variance of the normal distribution that is used to simulate the shift sizes is set here to three instead of one, since the standardised biases vary on a larger range. The feed-forward regression network combined with simple cut-off values (NN-CUT), the recurrent regression network with cut-off values (RNN-CUT) and the feed-forward classifier with 4 classes (4CLF) will on the contrary monitor the unstandardised bias, $\hat{\mu}_2(i,t)$ defined in (3.3.4), for which the networks are best performing.
For this analysis, we also keep all values in the stations, i.e. we do not remove the IC data that do not fall into one standard deviation around the cross-sectional mean as we did in Chapter 3.
Note that we also apply the NN-ACUSUM and RNN-ACUSUM to the unstandardised bias $\hat{\mu}_2(i,t)$ but they were slightly less effective in that case. Hence, we only present results when those methods are applied to the standardised bias in the following.

Design of the simulations

The methods are thus compared with the procedure explained below.
(1) First, all methods (except the classifier with four classes) are calibrated to reach a desired rate of false positive at $ARL_0 = 200$ with an accuracy equal to $\rho = 2$. The block length used to calibrate the CUSUM chart and the length m of the input vectors of the networks are set both to 100. (a) The control limit of the CUSVM method is designed using Algorithm 1 with the moving block bootstrap procedure. The target shift size is also selected at 1, which fixes the value of the allowance constant to $k = 0.5$. (b) The cut-off values of the networks are also adjusted by bisection searching as explained in Subsection 4.4.2 for the NN-CUT and RNN-CUT methods. The control limits of the *adaptive* CUSUM are finally computed following the procedure described in Subsection 4.4.3 for the NN-ACUSUM and RNN-ACUSUM.
Using those procedures with $B = 2000$ runs, we obtain a control limit equal to $L = 33.2$ for the CUSVM with $\delta_{tgt} = 1$, $L = 18.8$ for the NN-ACUSUM, $L = 16.8$ for the RNN-ACUSUM, $L = 0.106$ for the NN-CUT and $L = 0.096$ for the RNN-CUT.

(2) As stated in Chapter 3, the $ARL1$ represents the mean number of samples collected from the appearance of a shift to the alert of a method. For same values of IC average run length (ARL_0), a method has thus better performances if its $ARL1$ values are lower. In a second stage, the ARL_1 values are then computed. For each run, an IC series of length 2000, denoted $x_{ic}(t)$, is generated from the standardised or unstandardised bias depending on the method, by the block bootstrap with a

block length equal to 100. An artificial deviation of a certain shape and size is then added on top of this series. (a) In the CUSVM method, the CUSUM chart is applied directly to this series. The run length is computed as the number of values before an alert is triggered by the chart. (b) With neural networks, the series is assembled into moving windows of size $m = 100$ and fed to the networks which predict the size of the deviations at each time. An additional procedure — either the simple cut-off values or the adaptive CUSUM chart — is then used to trigger the alert. The run length of the method is then saved. (c) The classifier with four different classes (CLF4) is also applied to the series. The run length of the method corresponds here to the number of observations that are classified as 'in-control', before another type of deviation is detected.
The $ARL1$ is finally computed for all methods as the mean of the run lengths, over $B = 2000$ runs.

For this study, three types of general deviations of size δ were simulated:

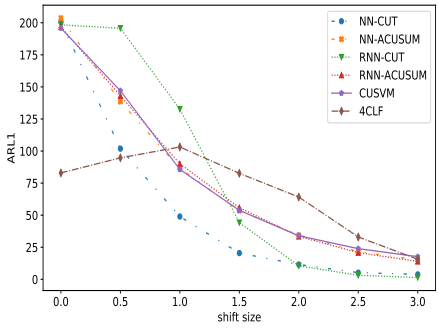
- jumps: $x(t) = x_{ic}(t) + \delta$;
- drifts: $x(t) = x_{ic}(t) + \frac{\delta}{500}(t)^a$, where a is randomly selected in the range $[1.5, 2]$;
- oscillating shifts: $x(t) = x_{ic}(t) + \sin(\eta\pi t)\delta$, where η is randomly selected in the range $[0.02, 0.2]$.

Results

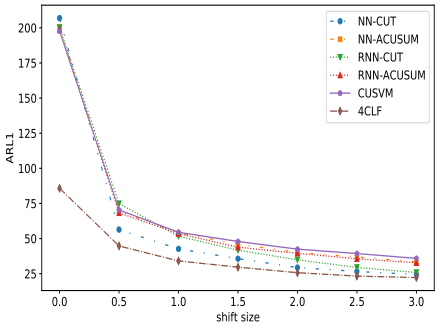
The ARL_1 values are shown in Figure 4.13 as a function of the shift sizes (δ). They are computed on data smoothed on a year but similar results are obtained on data smoothed on 27 days. Since the standardised and unstandardised biases vary on different scales, we adapt the size of the shifts in both cases. The shift sizes that are displayed in the figure correspond to those that are simulated for the standardised bias whereas those shift sizes are 21 times lower for unstandardised biases. This value of 21 was computed as the ratio between the standard deviation of the standardised and unstandardised bias ($std(\hat{\epsilon}_{\hat{\mu}_2}(i_{IC}, t))/std(\hat{\mu}_2(i_{IC}, t))$), on the IC stations. This means that an artificial deviation of size e.g. $\delta = 1$ for the CUSVM, NN-ACUSUM and RNN-ACUSUM corresponds to a shift size equal to $\delta = 0.048$ for the NN-CUT, RNN-CUT and 4CLF methods. The shift sizes are all represented on the same x-axis in the figure however, for clarity purpose.

Based on Figure 4.13, we can make the following comments:

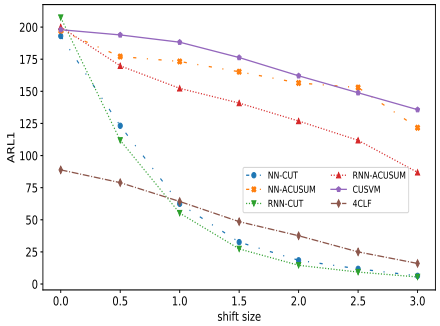
- In general, the different methods have similar performances for monitoring drifts, which are the easiest deviations to detect. More differences appear however for detecting jumps and oscillating shifts.



a: ARL_1 values for jumps.



b: ARL_1 values for drifts.



c: ARL_1 values for oscillating shifts.

Figure 4.13: Out-of-control average run lengths (ARL_1) as a function of the shift size. The ARL_1 values are computed for different methods: feed-forward network with simple cuts-off (NN-CUT), feed-forward network with adaptive CUSUM (NN-ACUSUM), recurrent networks with simple cuts-off (RNN-CUT), recurrent network with adaptive CUSUM (RNN-ACUSUM), CUSVM method and classifier with four classes (4CLF).

- The classifier (4CLF) works correctly with respect to the other methods but has a higher rate of false positives. On average, a false positive occurs after that around 85 observations are gathered instead of 200. This is expected since we cannot control its rate of false positives. Note that we only design a feed-forward classifier here but a recurrent classifier with four classes would have given similar results.

- The three methods based on the CUSUM chart, namely the CUSVM, NN-ACUSUM and RNN-ACUSUM have similar performances except for oscil-

lating shifts, where the adaptive CUSUM charts are more effective than the simple CUSUM. The NN-ACUSUM and RNN-ACUSUM also slightly outperform the CUSVM for detecting small and large jumps and drifts. This can probably be explained by the fact that the CUSUM chart of the CUSVM procedure is calibrated to be optimal to detect shift of size $\delta_{tgt} = 1$. Outside of this regime, the chart is thus less effective. It appears however to stay competitive with the procedures based on adaptive CUSUMs over a large range of values. There is thus some flexibility to choose the target shift size in the CUSVM method.

- The NN-CUT, RNN-CUT and 4CLF often outperform the CUSVM, especially for identifying large shifts and oscillations. Among them, the recurrent networks with simple cuts-off are slightly more effective to detect large deviations and oscillations than the feed-forward networks. Few differences are however visible between both types of networks, since they were designed to reach similar performances measured in MAPE values. They identify thus similar shift sizes.

For readability purpose, we do not show in the figure networks calibrated using the heuristic decision procedure described in Section 4.4.2. In general, networks combined with this heuristic procedure (based on δ^1_{cut} and δ^2_{cut}) perform slightly better than networks associated to simple cuts-off values (δ_{cut}) for identifying small jumps and drifts. They perform however worse with large shifts, as expected. This heuristic procedure should however not be used for detecting oscillating shifts, since the rapid oscillations prevent the data from staying n_L consecutive values above δ^1_{cut}. Hence, only the upper cut-off value $\delta^2_{cut} > \delta_{cut} > \delta^1_{cut}$ is actually used for triggering an alert, which diminishes the performances of the methods on oscillations.

To summarize, the methods based on the CUSUM chart (RNN-ACUSUM, NN-ACUSUM, CUSVM) exhibit similar performances except for detecting oscillating shifts, for which the adaptive CUSUM is slightly more effective than the CUSUM chart. The four classes classifier, although simpler to design, does not allow us to control the rate of false positives and has thus less practical interest than the other proposed methods. Moreover, the networks combined with simple cut-off values often outperform the other methods, including the CUSUM chart, especially to detect oscillations and large deviations.

4.6.2 Estimation of the shift sizes

In this second subsection, we compare the predictions of the networks and support vector machine to estimate the shift sizes. Those methods are compared with the following procedure. The IC biases, $\hat{\mu}_2(i_{IC}, t)$, are first partitioned in blocks of

length equal to $m = 100$ with the MBB method described in Chapter 3. For each run over 50000 examples, a block is randomly selected and an artificial deviation of certain shape and size is added on top of it. The block is then passed through the SVR and networks as an input vector, for estimation of the size of the shift. Here, we randomly sample the shift sizes δ from a standardised normal distribution. Three types of general deviations are then artificially constructed on top of the selected blocks:

1. jumps: $x(t) = x_{ic}(t) + \delta h(t)$, with $h(t) = \begin{cases} 1 & \text{if } t > \tau \\ 0 & \text{otherwise.} \end{cases}$ and τ in $[0, m]$;
2. drifts with varying power-law functions: $x(t) = x_{ic}(t) + \frac{\delta}{b}(t)^a$, where a is randomly selected in the range $[1, 2]$ and $b = 500$;
3. oscillating shifts with different frequencies and phases: $x(t) = x_{ic}(t)+\delta \sin(\eta\pi t + \phi)$, where η is randomly selected in the range $[\frac{\pi}{2m}, \frac{2\pi}{m}]$ and ϕ in $[0, \frac{m}{4}]$, for $m = 100$.

Note that those deviations have more complex shapes here than in the previous subsection, since we aim at comparing the estimations of the shift sizes and not the detection power of the methods. Same conclusions can however be drawn with previous shapes.

As proposed in Laperre et al. (2020), we also compare the networks and SVR with the predictions of a persisting model, which is used as a reference. This model simply estimates the size of the shift on each block as the last value of the block:

$$\hat{\delta}_{pst} = x(m). \tag{4.6.1}$$

It is one of the simplest model that can be used to make predictions. Contrarily to the persisting model, the SVR and networks make their predictions based on the entire block (i.e. input vector) of length equal to $m = 100$. The blocks contain thus more than one deviating observation. Hence, we also display results for the persisting model where its predictions are based on a value in the middle ($\hat{\delta}_{pst} = x(m/2)$) and at the beginning of the blocks ($\hat{\delta}_{pst} = x(1)$).

The performances of the methods evaluated using the MAPE criterion are displayed in Table 4.1. As a remainder, this criterion is defined as:

$$MAPE = \frac{1}{M} \sum_{j=1}^{M} \left| \frac{|\delta^j| - |\hat{\delta}^j|}{|\delta^j|} \right| \times 100\%,$$

where $\hat{\delta}^j$ is the shift size predicted by a method (which can be positive or negative) and δ^j is the true size. As can be seen in the table, the most effective method for detecting all types of deviations is the SVR, followed by the feed-forward and

Method	Jumps	Drifts	Oscillations	Total
SVR	7.87	56.34	50.63	38.28
NN	20.59	65.15	75.86	53.87
RNN	19.05	69.76	80.14	56.32
Pst ($\hat{\delta}_{pst} = x(1)$)	86.82	88.25	38.56	71.64
Pst ($\hat{\delta}_{pst} = x(m/2)$)	49.18	96.80	38.90	61.63
Pst ($\hat{\delta}_{pst} = x(m)$)	13.25	358.31	39.44	137

Table 4.1: Mean absolute percentage error (MAPE) values of the different methods to estimate the shift sizes. Three methods, namely a support vector regressor (SVR), a feed-forward regression network (NN) and a recurrent regresion network (RNN) are here compared with the persisting model (Pst). The performances are computed separately for the jumps, drifts and oscillating shifts as well as for all deviations combined (Total).

recurrent networks. When only considering jumps, the most powerful method remains the SVR. The persisting method also works particularly well for identifying jumps if the last observations of the blocks are used. When other values are chosen however, the persisting model is less effective. This is expected since the jumps are simulated to appear at a random position inside the blocks. Contrarily to the other procedures, the persisting model is not designed to identify shifts occurring at random positions inside a specified windows.
The persisting model is on the contrary particularly ineffective to detect drifts. This is especially true when the last observation of the block is used ($\hat{\delta}_{pst} = x(m)$), since the deviation has the time to grow. The last observations of the blocks take thus high values. For oscillating shifts however, there is practically no differences when making predictions on the first, last or middle observations, as the oscillations vary over the same range and do not grow over time. For those oscillations, the persisting model outperforms all other methods, including the SVR.
A slightly modified version of the MAPE criterion, which takes the sign of the deviations into account, can be defined as

$$MAPE(2) = \frac{1}{M}\sum_{j=1}^{M}\left|\frac{\delta^j - \hat{\delta}^j}{\delta^j}\right| \times 100\%.$$

When computed with this second version, all methods show similar performances for identifying the oscillating shifts, with MAPE(2) values equal to 100. We thus conclude that all methods have similar performances to identify oscillations and are in general more effective to detect jumps and drifts.
To summarize, the SVR, which was retrained here to identify deviations in the unstandardised bias $\hat{\mu}_2(i,t)$, appears to be the most effective predictive method.

Note that the conclusions of this section also applied for monitoring the sunspot numbers *if* the simulated shifts are representative of the actual deviations of the data. Careful visual analyses confirm that this is indeed the case. Other methods based on e.g. manual fits could also be used to analyse some deviations in details and measure in quantitative terms how far the actual deviations of the data are from the simulations. They were not applied here however, since those methods are not automated and the data experience many deviations over time. In future works, the simulations could also be improved by the following iterative procedure. A chosen predictive method such as SVM or a network can be designed and trained on a set of artificial deviations such as the one proposed in Section 4.3.2. The method can then be applied to the actual data and the number of deviations detected in each class can be evaluated. Based on these results, the simulations can be progressively improved, to contain e.g. more deviations that are often present in the data. Other types of artificial deviations (such as e.g. exponential shifts) can also be tested to better correspond to the actual shifts. This procedure is however time consuming, since it requires the design and training of the predictive method on different sets of artificial deviations. It has thus not been implemented here.

4.7 Conclusion

In this section, we construct feed-forward and recurrent networks to predict the size and shape of the deviations in the sunspot numbers. They are then associated to simple cut-off values in the spirit of a Shewhart chart or to an adaptive CUSUM chart, to trigger the alerts. As have been seen, those neural network (NNs) based procedures have several advantages with respect to the CUSVM method that was previously developed. We summarize here some solutions provided by the NN-based control schemes to the four criticisms that were addressed to the CUSVM method:

1. **Numerous stages of the procedure**: The networks combined with simple cut-off values (NN-CUT and RNN-CUT) can monitor unstandardised bias. They require thus less data processing than the CUSVM method since they do not need an IC mean and variance. This saves time but also simplifies the procedure with respect to the CUSVM, which requires thus fewer steps.

2. **Problem of the parameters adjustment**: When using NN combined with the adaptive CUSUM chart (NN-ACUSUM and RNN-ACUSUM), the allowance parameter is automatically adjusted to the predicted values of the shift size, $k = \hat{\delta}(i,t)/2$. This is valuable since the allowance parameter is difficult to choose and affects the performances of the CUSUM chart.

3. **Non-optimality of the CUSUM for all shift sizes**: This automatic adjustment of k to the predicted shift size with adaptive CUSUM charts allows

a better detection of the smallest and largest shifts as well as oscillations with respect to the CUSUM chart (of the CUSVM method).

4. **"Exploding" CUSUM values**: The NNs combined with simple cut-off values (NN-CUT and RNN-CUT) do not cumulate the deviations of the data. Hence, the predictions of the NNs are not "exploding", like the CUSUM chart statistics. There is thus no need to define upper values for those predictions contrarily to the CUSVM method where $|C_j^+|$, $|C_j^-| \leq 2L$. Additionally, the cut-off values of the networks are expressed directly into the shift size units and are thus easier to interpret than the control limits of the CUSUM or adaptive CUSUM charts.

Moreover and this is probably their greatest asset, the NN-based control schemes often outperform the CUSVM method on simulations, especially to detect large or oscillating shifts. Depending on the monitoring target, different methods may therefore be used. In the particular case of the sunspot numbers, detecting oscillations or small jumps has a limited interest since those deviations are unlikely to affect the long-term behaviour of the series. The NN-CUT or RNN-CUT may thus be selected for monitoring the high-frequency (at 27 days) deviations since those methods identify faster large jumps, which are more numerous at this scale than the drifts. Although the NN-CUT or RNN-CUT may also be used for monitoring low-frequency data at a yearly scale, the CUSVM method appears to be particularly suited for this task since it slightly outperforms NN-based schemes for identifying small drifts, which are expected to be at the basis of long-term deviations.
The NNs on the other hand work a bit like black boxes. Consequent analysis are thus required to comprehend their working. Those networks are also complicated to design since they require the construction of whole architectures, which takes time. The CUSVM method may thus be preferred for some applications, in particular those which require a deep understanding of the control scheme. An alternative monitoring method, based on SVM procedures to estimate the size and shape of the deviations and combined with simple cut-off values, could also be designed in future works. This approach would combine the advantages of the NN-based control schemes while being at the same time simpler to design. Considering the previous simulation results, it is also expected to perform well with respect to the proposed methods.

As concluding remark, the CUSVM and NN-based control schemes are flexible methods, which can be applied to other data than the sunspot numbers with few adjustments. Both methods would require a new model of the data in order to properly work on another monitoring problem. The training set of the NNs as well as those of the SVM procedures, which are generated by simulations, should also be adjusted to other datasets.

CHAPTER 5

Application to photovoltaic production data with focus on practical computational aspects

Often the packaging is as important as the product itself. The main functions of a packaging are the identification of the product and its developer, its protection (from dust or water but also from illegal reproductions) and its promotion. A good packaging is also a matter a convenience since it allows an easy manipulation and use of the product. Although obviously true for manufactured products in industry, this is also valid for pieces of computer codes. Those are often organized into a structure composed of different folders, which allows its easy installation and use: a package. Nowadays, many journals asks the authors to provide data and computer codes alongside their submission, for a better understanding and reproducibility of the results. More and more attention is thus paid to the code supporting the computational findings.

In this context, we develop a package written in the programming language Python, which implements the methodology of Chapter 3 and allows its easy handling. Although developed for analysing sunspot observations across a panel of observing stations, this method can easily be adapted to other settings of data streams which share a common principal signal. This is demonstrated in the following, for observations related to the photovoltaic production. The purposes of this work are thus to present to package as well as to identify periods and patterns of potential data acquisition or processing problems in another relevant application.

5.1 Introduction

In this chapter, we apply the monitoring method developed in Chapter 3 to another panel of data. This serves three main purposes, to illustrate the general applicability of the methodology, provide a tutorial for the software, and demonstrate its use in an example of practical importance. Although the CUSUM chart, the support vector machines and the block bootstrap procedures (which are the main ingredients of the method) are widely used in many applications, some features of the sunspot data may have overshadowed the generic nature of our methodology in Chapter 3. Here we introduce another example, the observation of photovoltaic production across Belgium, and show which parts of the methodology extend directly to this situation and which parts require adaptation.
In doing so, we present the functions of our open-source software package developed in the Python language (`https://github.com/sophiano/cusvm`), which contains the implementation of all procedures that are described in Chapter 3. The name of the package (`cusvm`) plays on the two main ingredients of the method, the CUSUM chart and SVMs. While the method has been explained in detail in Chapter 3, we thus focus here on its practical implementation.

As a result, we demonstrate how we can effectively monitor observations of the photovoltaic energy production across distribution system operators (DSOs) in Belgium. These data are collected on a quarter hourly time scale by a supplier of DC/AC converters installed in photovoltaic systems pre-processed by an intermediate and published on the web portal of Elia, the Belgian high-voltage transmission system operator. They serve in the process of load forecasting and allocation. As will be seen in the following, those data exhibit similar properties as those of the sunspot numbers, for which the monitoring method that we developed is particularly suited. Detecting deviations or inconsistencies in those data and analysing their cause is thus an interesting practical task that will be the object of a future collaboration with Elia.

This chapter is structured as follows. In Section 5.2, we briefly describe the content of the package and how it can be installed. The data and their characteristic features are then presented in Section 5.3. The CUSVM method is then calibrated on those data in Section 5.4. Detailed explanations related to the functions that should be applied for this purpose are also included in the section. The results are then presented and discussed in Section 5.5.

5.2 Setup

The package `cusvm` is written in Python 3. Before installing it, we strongly advice to set up a new Python environment, with the latest version of Python installed. Versions of Python superior than or equal to 3.6 also work. Such an environment can be easily created using the platform Anaconda[1], which is freely available for the main operating systems (Linux, Windows and macOS). Then, the easiest way to install `cusvm` is via the package manager `pip`[2]. The following command alone allows the installation of the package (main functions and data) as well as all its dependencies (i.e., the other packages that are called by `cusvm` and without which the package cannot work). The complete list of those packages and their corresponding versions can be found in the file `setup.py`, which contains all installation requirements.

```
pip install git+https://github.com/sophiano/cusvm
```

After installation, the main functions of the package can be imported (for direct use) with the following command.

```
import cusvm as cusvm
```

The data that are automatically downloaded with the package can also be retrieved as follows.

```
import pkg_resources as pkg
import pandas as pd
import numpy as np

data_path = pkg.resource_filename(pkg.Requirement.parse("cusvm"), 'data')
df = pd.read_csv(data_path + '\PVdaily.csv')
data = np.array(df)[:,1:]
data = data.astype('float')
data = data/96
```

```
import pickle

with open(data_path + '/time_daily', 'rb') as file:
    my_depickler = pickle.Unpickler(file)
    time = my_depickler.load()
```

The energy production is recorded every 15 minutes. The file `PVdaily.csv`, which may be opened with the library `pandas`, provides the daily sum of these productions. They are then converted into a `numpy` array, which allows the rapid manipulation of the data in the following. The library `numpy` is indeed specialised in numeric calculations with multidimensional arrays of numbers. Since we want to

[1] Anaconda can be downloaded from https://www.anaconda.com/.
[2] Pip package manager can be installed from https://pip.pypa.io/en/stable/.

monitor the daily average production, the data in `PVdaily.csv` are thus divided by 96 (24*4). We also provide the serialized file `time_daily`, which contains the corresponding time of the measures, expressed in fraction of years. This file can be opened with the library `pickle`.
Note that the package can also be downloaded from github in a zip file. To correctly function however, the user should manually install all dependencies of the package and work with an appropriate version of Python. All paths (for importing e.g. data, functions or pre-trained SVM models) should also be modified to match those of the user.

A package is typically composed of several files: a licence (here it is the MIT), a manifest that is used to import the data when installing the package, a setup which allows an easy installation of the package and a README. As its name suggests, the README is the first file that should be read. It contains a short description of the package and its content. It also explains how to install the package and gives further references. We provide here arXiv links to the main articles describing our method.
Our package also contains several folders. The main functions are located in the folder that has the same name as the package: `cusvm`. The folder `data` holds the data files that are also imported when the package is installed. The scripts are located inside the folder `scripts` whereas the documentation of the package is included in the folder `docs`. Those folders are common in many packages. We provide here another one, `svm_models`, which is particular to our monitoring method. It contains the pre-trained SVM models, which are saved for reproducibility purpose. Each folder also contains a README file that describes its content in more details. After installation, the functions of the folder `cusvm` and the data are installed. The other files are not imported but are available on github for helping the users to work with the package. The scripts show for instance how the main functions could be used and in which order to monitor panels of time series data. In the package, they are applied, as one example, to the data included in the folder of the same name. Jupyter notebooks are also provided inside `docs`. They contain basically the same code as the scripts but with many more explanations. Note that those notebooks can be opened online[3], without requiring a local deployment of Python.

In the following, we assume that the package is properly installed and operational. We focus on the description and application of its main functions (located inside the folder `cusvm`). Those are distributed among different files for clarity purpose. The file `bb_methods.py` contains all functions that are related to the block bootstrap (BB) procedures. Although not directly used thereafter, many functions inside the package rely on those methods (i.e. the file is internally called many

[3] Jupyter notebooks can be read online, at the following link for instance: https://jupyter.org/try.

times in the package). The file `preprocessing.py` contains a set of functions that can be used to pre-process the data. It is composed of e.g. functions to smooth or rescale the data with respect to a reference. It also contains methods to automatically select a pool of IC series from a panel and to standardise the data by K nearest neighbours regression method. `autocorrelations.py` is principally used to select an appropriate value for the block length of the BB procedures. It also contains functions to analyse the autocorrelation of series prone to missing values. `cusum_design_bb.py` is composed of different algorithms that can be used to calibrate and evaluate the performances of the CUSUM chart designed by BB. Similarly, the file `svm_training.py` holds functions to train the support vector machine methods. Finally, `alerts.py` should be called to actually apply the monitoring procedure to the data. It also includes functions to display the main results of the monitoring.

All those functions have thus one or several arguments. Those can be mandatory, meaning that the functions return an error if those arguments are not passed through the functions when they are called, or optional. The optional arguments are set to a default value in the definition of the functions and are used with their default value if not specified otherwise. The complete list of arguments can be found at the beginning of each function. This documentation also contains few explanations about the aim of the function, its implementation, the default value of its arguments and their type (optional or mandatory). In the following, we review the main arguments of the functions but seldom all of them. We thus strongly encourage the user to consult this documentation for more information.

5.3 Data

In this section, we first present the data that will be analysed in the remaining part of the chapter. Those are the photovoltaic energy production in Belgium. Then, we introduce a model for these data and isolate a relevant variable that contains potential deviations and will be monitored in the following.

5.3.1 Presentation of the data

The data related to the photovoltaic energy production in Belgium are provided by Elia[4]. Elia is the Belgian transmission system operator for high-voltage power. It manages the transport of electricity from generators to distribution system operators (DSO), working with medium and low voltage. Its primary activities are to maintain and develop the high-voltage network infrastructures and manage the

[4]The data are available at the following link: https://www.elia.be/fr.

power system. Elia is thus also responsible for keeping the balance between power production and consumption or for handling the importation and exportation of electricity toward other countries. To better manage the system, it needs to take into account the photovoltaic energy production as well. Elia thus retrieves those data from different system operators across the country and make predictions to forecast the upcoming amount of energy produced. Those allows Elia to e.g. plan ahead shortfall or surplus of energy and take appropriate measures.
The photovoltaic energy production data suffer however from inconsistencies or errors related to e.g. data transmission issues. In the remaining part of the chapter, we will therefore apply our monitoring method to those data. Our main objective is to detect past deviations and try to find their root-causes, with aim to prevent the occurrence of similar inconsistencies in the future.

In the following, the period under study extends from January 1, 2015 till January 27, 2021. In this period, both the data that are sampled every 15 minutes and their average daily values are analysed.
The data correspond to the load factors, which are defined as the ratio between the real time photovoltaic energy production measures divided by the related installed capacity, expressed in percentages. They correspond to the upstream photovoltaic energy production (before consumption) in the different installations. Those are built for professional or domestic uses and may thus vary in size. In total, the data come from around 70000 of such installations, which are spread across the country. They are provided by different DSOs, who gather the production of a particular region in Belgium. The data form thus a panel of observations, whose composition and main characteristics are summarized in Table 5.1.

Photovoltaic energy production

To better understand the data and potential causes of anomaly, we review here some basics about photovoltaic energy production. A typical photovoltaic installation contains different components, whose main elements are the solar panels. Those are in turn composed of photovoltaic cells that convert the solar energy received into electrical energy using the photovoltaic effect. Afterwards, an inverter transforms the direct current (DC) produced by the photovoltaic cells into alternating current (AC). The current generated is then either directly used in the local/domestic network (if any) or redirected into the power grid. It may also be stored into batteries for future needs. Nowadays, most of the inverters (for example those produced by the company SMA[5]) have integrated devices that allow them to record the energy produced and to communicate it to the network or/and to the holder of the installation via Internet. We are working here with such data, which

[5]Solar Technology AG (SMA) is a German producer and manufacturer of solar inverters, see https://en.wikipedia.org/wiki/SMA_Solar_Technology.

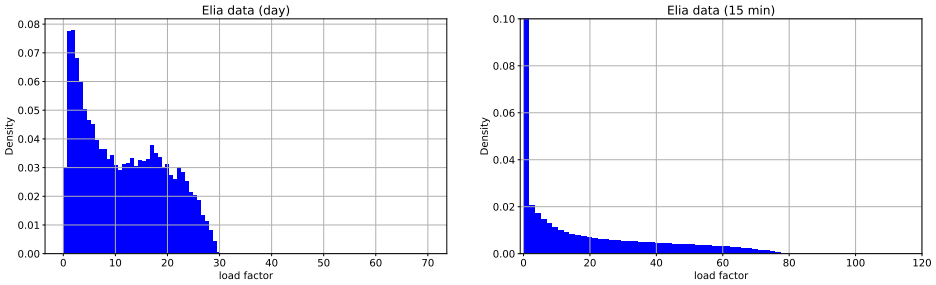
Name	Location	level
AIEG	Andenne/Gesves	102
AIESH	South of Hainaut	97
Gaselwest	Courtrai/Ypres	109
IMEA	Antwerp (Prov.)	100
IVEG	Antwerp (Prov.)	100
IVEKA	Antwerp (Prov.)	99
Imewo	East and West Flanders	105
Infrax West	East and West Flanders	109
Inter-Energa	Limburg (Prov.)	101
Intergem	East Flanders and Bever	100
Iverlek	Flemish Brabant and Antwerp	100
Ores Brabant Wallon	Walloon Brabant	101
Ores Est	Eupen	111
Ores Hainaut	Hainaut (Prov.)	103
Ores Luxembourg	Luxembourg (Prov.)	108
Ores Mouscron	Mouscron	110
Ores Namur	Namur	101
Ores Verviers	Verviers	107
PBE	Walloon and Flemish Brabants	100
Regie de Wavre	Waver	102
Resa	Liege	103
Sibelga	Brussels	100
Sibelgas	Brussels	100

Table 5.1: Main characteristics of the DSOs. The table contains the name of the operators and the regions where the data come from. The last column also represents the level of the regions on the years 2016-2020 (the year 2015 was excluded from the computation since it contains unusual deviations). It corresponds to a mean scaling-factor (obtained for daily values) expressed in percentages and represents intrinsic differences between the regions due to e.g. altitude, micro-climate, pollution level etc. Values above hundred indicate that the region produces more energy than the median of the network (we show here the mean value of χ defined in (5.3.1), see below).

may thus be affected by Internet disconnections and power cuts.

As stated before, the photovoltaic cells are the ones that produce the energy. The amount of power produced is therefore highly dependent on the solar radiations that they receive and is affected by several factors. The atmosphere is responsible for sending back around 30% of the solar energy into space. It also absorbs and scatters part of the radiations in any direction. What is remaining reaches then the

ground. The thickness and, to a lesser extent, the composition of the atmosphere impact thus the quantity of energy that is received by the photovoltaic panels. This explains why the production is weaker in the morning and afternoon than at mid-day, when the sun-rays go through a shorter distance in the atmosphere to reach the ground. For the same reason, the photovoltaic production also exhibits a yearly cycle (see Figure 5.2), since the length of the days and the elevation of the Sun on the horizon depend on the time of the year (e.g. the production is thus weaker in winters than summers). Moreover, although it may seem counter-intuitive, the energy produced diminishes when the temperature increases (Dubey et al., 2013). The latitude and the altitude of the installations can thus influence the production as well as the inclination and orientation of the panels. Finally, the energy received is also influenced by obstacles that can mask the Sun such as clouds, topographies (mountains, etc) or even neighbouring barriers such as buildings or trees.
Among those phenomenon, the altitude and local weather conditions are expected to have the strongest impacts on the Belgian production. Those effects should be minor however, since the size of the country is small.



a b

Figure 5.1: Histograms of the photovoltaic energy production for all operators/regions in the period studied. (a) The histogram is represented in (a) for daily values and in (b) for data obtained each 15 minutes.

Properties of the data

The photovoltaic energy production data have therefore complex features that are outlined below.

Non-Normality As can be seen in Figure 5.1, the data are non-normally distributed. The histogram of the daily values looks similar to those of the number of sunspot groups (N_g) obtained in Figure 2.4, which has been modelled by a mixture of a negative binomial and a Poisson distributions. The data obtained each 15

minutes resemble on the other hand more to the number of sunspots (N_s), approximated by a mixture of two negative binomials in Section 2.5. They also experience a large excess of zeros due to the nights, where the solar energy production drops to zero. Note that, this shows once again that zero-altered models are common in many applications.

Missing values

There are no missing values in the dataset. However, it might be interesting from a practical point of view to treat the zeros as missing values. Actual missing values can also happen in practice, for instance if the inverter which transmits the amount of production of a particular installation to the system operator is dysfunctional. We do not know how the DSO encode those errors (we only have the data for the entire region managed by each DSO) but it could register them either as missing values (which makes more sense) or as zeros.

Correlations

The energy productions are correlated across the panel, since the country is small. The solar energy received is therefore more or less similar in the different regions. The data are also correlated along the time since the amount of solar radiations that hits the ground depends on the incidence angle of the sun-rays. It thus progressively increases during the day till a maximum (which depends on the period of the year) before descending to zero at night. Moreover, this angle also changes throughout the year. Hence, the series are autocorrelated, in particular those that contain values each 15 minutes. The daily values also experience autocorrelations but until lower lags. Additionally, the cloud coverage is also a source of correlation across the panel and along time.

Multi-scale deviations

Different kinds of deviations may affect the data. Meteorological events (storms, etc) may perturb the production in localized regions over brief periods. They may also have long-term effects if they degrade several installations in a region. Short power or internet blackouts can disrupt the transmissions of the inverters to the DSOs. Those can last from few minutes to few days. Moreover, longer term variations can also appear in a region if the actual capacity of the installations deteriorates over time without being upgraded on a regular basis or if the monitored capacity is poorly determined/ transmitted to the DSOs. The variations in the data may thus be caused either by the weather conditions or by dysfunctions in the installation or in the communication with the network. While the former may be compared to local weather observations and ignored, the latter may hopefully be corrected to provide a better network management. In general however, the data are expected to have a rather high signal-to-noise ratio.

As a result, the photovoltaic production data have similar features (i.e. the non-normality, the autocorrelation and the multi-scale deviations) as the sunspot num-

bers. The method developed in Chapter 3 is therefore particularly adapted to monitor those data.

5.3.2 Model

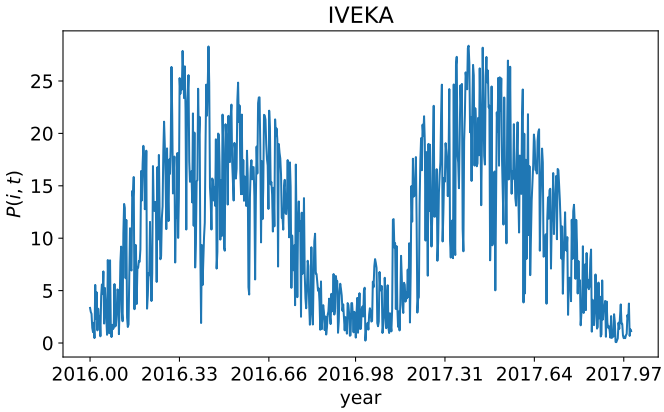


Figure 5.2: Photovoltaic energy production recorded by the DSO IVEKA over the years 2016-2017. As can be seen, the variations in the data are larger when the production is higher, which suggests a multiplicative model.

In the following, we denote by t, $t \in 1, ..., T$, the time of the measures. Those corresponds either to the day or to the quarter of an hour of the records. We also represent the index of the operators/regions by i, $i \in 1, ..., N = 23$, to keep consistency with the notations introduced before.
As seen previously, the data, denoted by $P(i, t)$, may be decomposed into different components:

$$P(i, t) = \eta(i, t)\ \chi(i, t)\ c(t). \tag{5.3.1}$$

In (5.3.1), $c(t)$ represents a common signal between all regions. This quantity models the overall amount of solar radiations that reaches the ground in Belgium. It also accounts for weather conditions that affect the whole country. This quantity depends thus on the time, since the solar energy received varies with the hour of the day and the period of the year. It is also latent since the actual amount of energy reaching the ground depends on complex phenomena in the Sun and in the atmosphere that cannot be observed. In the following, we will thus estimate the mean of $c(t)$ using the panel and employ this mean as a proxy for $c(t)$, as we did with $s(t)$ in Chapter 2.
The variable $\eta(i, t)$ of (5.3.1) accounts on the contrary for the regional variations in the data. It thus corresponds to the different amount of solar radiations that

are received in the regions due to e.g. localized weather conditions or to local malfunctions. This variable will therefore be monitored in the following to detect potential defects in the network.
Finally, another component denoted by $\chi(i,t)$ is also included in the model, to represent the level of the regions. Those $\chi(i,t)$s are piece-wise constant scaling factors that typically represent the altitude, composition of the atmosphere (e.g. the level of pollution) and micro-climate of the different areas. Their piece-wise constant nature echoes some steps in the processing of the load factors (for which we have not been granted full access yet), which are abruptly updated at regular intervals. These factors are allowed to change with time as the climate or pollution may change over the years but vary at a slower peace than the other variables. Hence, they are computed on a specified period that is larger than the sampling period of the data and take thus fewer values in the entire period studied (e.g. 2015-2020). As few disparities of altitude and climate are observed in Belgium, differences of only few percent (10-15%) are expected between the regions. We also assume that the mean of the $\chi(i,t)$s over the panel is equal to one.

The model in (5.3.1) is written in a multiplicative framework, such as those of the sunspot numbers. Those models are common in the literature to account for variations that depend on the values of the data (i.e. to model situations where large variations are expected when the data take high values). Such variations are visible here in Figure 5.2. Physically, a multiplicative model also seems appropriate since the solar energy received at the ground is the *fraction* of the total energy that passes through the atmosphere without being absorbed. The variable $c(t)$ is therefore multiplied to the other terms of the model. Similarly, since the factors $\chi(i,t)$ model the composition of the atmosphere and its degree of transparency above a defined area, they can also be multiplied to the error term, $\eta(i,t)$.
Note that additive models are also frequent in practice and can be treated by our monitoring method as well. In this case, the preprocessing of the data, explained in the next subsection, should simply be modified to correspond to an additive instead of multiplicative model.
In the following, we assume that the random variables c and η are continuous and that χ, c, and η are also jointly independent.

5.3.3 Preprocessing

To monitor η, we need to isolate it from the other terms of the model in (5.3.1). To this end, we take some inspiration from the procedure that was used to estimate the long-term bias of the sunspot numbers in Section 3.3. Note that the model of (5.3.1) is valid for daily values as well as for data collected each 15 minutes. Both will be analysed in the following but we only present the preprocessing and the calibration of the method on daily values, for clarity purpose. Results will be

shown however for both types of data.

(1) We first rescale the data to roughly compensate for the different levels of the regions: $P_{resc}(i,t) = \frac{P(i,t)}{k'(i,t)}$, where $k'(i,t)$ are piece-wise constant scaling factors computed using the procedure described in Section 2.4. This can be done by loading the file `preprocessing.py` from the package `cusvm` and calling the function `rescaling()`. This function has two mandatory arguments: `data` and `period_rescaling`. It computes the piece-wise constant scaling factors as the slope of the ordinary least-squares regression between each series and the median of the data. Those factors are calculated on a pre-specified period, which is set here to one year: `period_rescaling=365`. This value seems appropriate since the degree of pollution or the micro-climate are not expected to change much over time. The data are then divided by these factors in the function, for the actual rescaling. The function finally returns two quantities: the rescaled data (saved into the variable `data_rescaled`) and their corresponding scaling factors (saved into `k_factors`). Functions in Python can thus return several objects.

```
from cusvm import preprocessing as pre

data_rescaled, k_factors = pre.rescaling(data=data, period_rescaling=365)
```

Note that the arguments of a function can be passed by position or by keyword in Python. Both types of arguments can also be mixed in the same function, with keyword following positional arguments. The following codes are thus all equivalent:

```
data_rescaled, k_factors = pre.rescaling(data=data, period_rescaling=365)
data_rescaled, k_factors = pre.rescaling(data, 365)
data_rescaled, k_factors = pre.rescaling(data, period_rescaling=365)
```

Then, we estimate $\hat{c}(t)$, a proxy for the common component, as the point-wise median of the rescaled series along time:

$$\hat{c}(t) = \underset{1\leq i\leq N}{\text{med}} P_{resc}(i,t). \tag{5.3.2}$$

To do so, we apply the function `median()` to the rescaled data.

```
med = pre.median(data_rescaled)
```

(2) Now that we estimated c, we can remove it from the (raw) data:

$$\hat{\eta}(i,t)\hat{\chi}(i,t) = \frac{P(i,t)}{\hat{c}(t)}. \tag{5.3.3}$$

To this end, we use the function `remove_signal()`. As most of the functions in the package, its first argument is the data. The second argument of the function is set

here to `model = 'multiplicative'`. With this setting, the common signal is divided from the data whereas it would have been subtracted if `model='additive'`. This function can thus be used to remove the common component of the series in additive or multiplicative models. The third argument of the function also allows the specification of this common signal (also called "reference" of the panel). By default, it is computed as the median of the data. Here, this reference is specified as the median of the *rescaled* data instead, `ref=med`.

```
ratio = pre.remove_signal(data, model='multiplicative', ref=med)
```

Note that, as observed before with the bias of the sunspot numbers, dividing the data by c produces high values when the signal is minimal. Errors are thus enlarged around minima with regard to other parts of the cycle. A similar effect would also occur at maxima if c was subtracted from the data (when `model='additive'`).

(3) We finally obtain an estimation (denoted by μ_η) of the mean of the localized variations η, by rescaling once again the data:

$$\hat{\mu}_\eta(i,t) = \frac{P(i,t)}{\hat{c}(t)\hat{\chi}(i,t)}. \tag{5.3.4}$$

In practice, we thus apply once more time the function `rescaling()` on the previous ratio to eliminate the level (χ) of the data.

```
mu_eta, chi_factors = pre.rescaling(ratio, period_rescaling=365)
```

This function returns two quantities: the rescaled ratios $\hat{\mu}_\eta(i,t)$ (saved into the variable `mu_eta`) and their corresponding scaling factors $\hat{\chi}(i,t)$ (saved into `chi_factors`), which allows us to study the levels of the regions as well.

The $\hat{\mu}_\eta(i,t)$, the initial data and all intermediate quantities are represented in histograms and as a function of time in Appendix 5.7.1.
Note that we could have rescaled the data before removing the common signal. We do the opposite here to better study the levels of the series, whose mean values are displayed in Table 5.1. Those levels differ indeed more between the regions after removing the common signal — which varies over higher orders of magnitude than the levels — than before.
Other functions that could be used for the preprocessing of the data are also defined in `preprocessing.py`. `remove_level()` may be used to remove an additive level to the series and `smoothing()` may be applied to smooth the data by a moving average filter of a specified window length. Those functions were both used in Section 3.3.3 to estimate the long-term bias of the sunspot numbers. They are not used here however, since the localised variations, $\eta(i,t)$, can be isolated from the model of (5.3.1) without applying those functions. In general, each monitoring problem requires thus a particular preprocessing, which depends on the nature and model of the data.

5.4 CUSVM method

Previously, we present the main characteristics of the photovoltaic energy production data and propose a model for them. In the following, we will then apply the non-parametric monitoring scheme developed in Chapter 3 to those data. Although the methods will not be modified, contrarily to the model, their parameters will be adjusted to the production data.

5.4.1 Phase I: Estimation of IC longitudinal parameters

In this subsection, we first select a subset of in-control (IC) series from the panel. We then estimate the mean and variance on those IC data along time, following the methods described in Section 3.4.1. The data from the whole panel are later standardised by these parameters as proposed in Section 3.4.1.

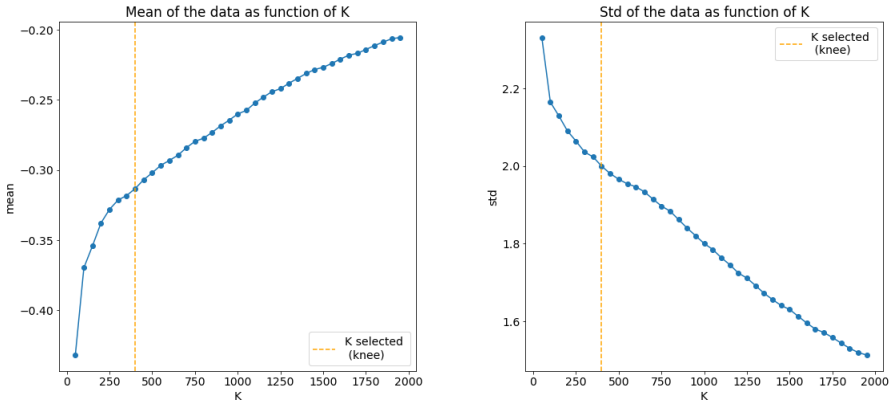
The subset of stable series is first selected using the function `pool_clustering()` from the file `preprocessing.py`. In this function, a robust version of the MSE defined in (3.4.1), is computed for each series. The series are then clustered in two groups based on their MSE value and the subset with the lowest values is selected as the IC group. The first argument of the function is the only one which is mandatory. It corresponds to the data to be clustered. The second argument of the function is optional and set here to `method = 'kmeans'` (the default value). With this method, the series are grouped using the k-means algorithm (see Appendix 6.9.1). The other clustering methods that are defined in the function are: `agg` which stands for agglomerative clustering, `ms` for mean-shift, `dbscan` for DBSCAN and `gmm` for the expectation-maximization clustering using Gaussian mixture models. Those methods are also described in Appendix 6.9.1. Two other values are also allowed for the argument `method`: `leftmed` chooses all IC series whose MSE is inferior to the median of the MSE and `fix` selects a pre-specified number of series (argument `nIC`): those whose MSE values are the lowest among the panel. Since the groups can be unbalanced, the function is designed to iterate the clustering until the smallest group contains at least 25% of the total number of series: `nIC_inf=0.25`.

```
pool = pre.pool_clustering(mu_eta, method='kmeans', nIC_inf=0.25)

#names_IC = [names[i] for i in range(n_series) if i in pool]
#['IVEKA', 'Intergem', 'Iverlek', 'Ores Brabant wallon',
# 'Ores Hainaut', 'PBE', 'Regie de Wavre', 'Sibelga', 'Sibelgas']
```

With the previous code, we obtain a pool of nine series (among the 23 energy operators in the panel). It contains the data from IVEKA, Intergem, Iverlek, Ores Brabant wallon, Ores Hainaut, PBE, Regie de Wavre, Sibelga and Sibelgas. The Flemish and Walloon Brabant together with Brussels appear thus to produce a

similar amount of energy than the median of the network. This can be understood by the fact that Belgium is more extended in longitude than in latitude. Hence, the atmospheric perturbations take more time to pass through the country from West to East (or reversely) than to navigate from South to North (and conversely). The regions located the farthest from the centre are thus expected to differ particularly much from a reference production level. They appear to be indeed excluded from the pool by the clustering method, contrarily to the regions in the centre of the country.
As one of our collaborator suspects, this may indicate that there are two components in the signal, which are related to the West and East regions. This inhomogeneity of the panel is specific to the photovoltaic production and does not appear in the sunspot numbers. Hence, it will not be treated in the following. One potential solution to take these differences into account however, would be to monitor separately the East and West regions. Another possibility would be to model the localised variations $\eta(i,t)$ in the East with an autoregressive model based on the values of $\eta(i,t)$ in the West (or conversely depending on the wind direction).
Note that since the data have a high signal-to-noise ratio, we do not remove the outliers in the IC series. This could have be done with the function `outliers_removal()` from the file `preprocessing.py`. For more information, we refer to the documentation at the beginning of the function.



a: Mean as function of K

b: Std as function of K

Figure 5.3: Mean and standard deviation (std) of the standardised data as a function of K.

Then, we compute the empirical mean and variance (denoted $\hat{\mu}_0(t)$ and $\hat{\sigma}_0(t)$) on the IC pool using K-NN regression method, which is just a box car smoothing across the panel and time. An appropriate value for the number of nearest neighbours

K is first selected using the function `choice_K()` from the file `preprocessing.py`. The two first arguments of the function are mandatory and correspond to the standardised data in the whole panel and in the IC pool only. There are thus set here to $\hat{\mu}_\eta(i,t)$ and $\hat{\mu}_\eta(i_{IC},t)$. This function selects K to obtain the "best" standardisation of the complete panel, in the sense that its empirical mean becomes close to zero and its empirical variance close to one. It works as follows. For different values of K in the range [`start`, `stop`] with a certain `step`, the function standardises the data by the IC mean and variance. The mean and the standard deviation (std) of the standardised data are then computed. Usually, the mean and std exhibit a similar behaviour: the mean comes closer to zero and the std draws near one with increasing values of K, as can be seen in Figure 5.3. K is thus finally selected as the knee (Satopaa et al., 2011) of the std curve.

```
K = pre.choice_K(mu_eta, mu_etaIC, start=50, stop=2000, step=50)

#K = 400
```

With this procedure, we select $K = 400$.

Finally, the data from the whole panel (IC and OC) are standardised by the IC mean and variance:

$$\hat{\epsilon}_\eta(i,t) = \frac{\hat{\mu}_\eta(i,t) - \hat{\mu}_0(t)}{\hat{\sigma}_0(t)}. \tag{5.4.1}$$

This can be done using the function `standardisation()`, which has only three mandatory arguments. The two first arguments of the function are the standardised data in the whole panel and in the IC pool only. They are set here to $\hat{\mu}_\eta(i,t)$ and $\hat{\mu}_\eta(i_{IC},t)$ whereas the third argument is `K`.

```
data_stn, dataIC_stn = pre.standardisation(mu_eta, mu_etaIC, K=400)
```

This function returns the standardised data in the panel and in the pool. Note that here the standardised IC series are the same in `dataIC_stn` and in `data_stn`, but this is not always the case. If we remove the outliers of the IC pool as done in Section 3.4.1, where we remove the IC observations that do not fall into one standard deviation around the cross-sectional mean, then those two quantities would be different (i.e. `data_stn` would still contain the deviations of the pool while `dataIC_stn` would not).

5.4.2 Phase II: Monitoring

Having selected an IC pool and standardised the data by the IC mean and variance, we can now move on to the second phase of the method. In this stage, we calibrate the CUSUM chart on the data using the block bootstrap (BB) procedure, as described in Section 3.4.2.

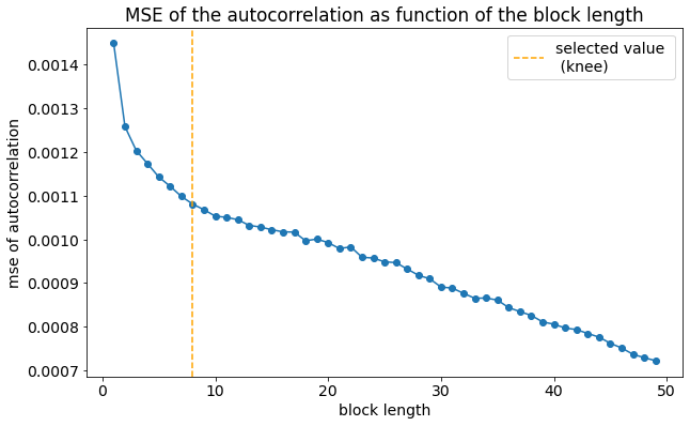


Figure 5.4: MSE of the autocorrelation of the IC series as a function of the block length.

To this end, we first select an appropriate value for the block length. This length depends on the autocorrelation of the data and can be automatically selected using the function `block_length_choice()` from the file `autocorrelations.py`. This function is based on Algorithm 3, which chooses the block length as the knee of the MSE of the autocorrelation curve. Hence, it selects the block length as the smallest value which represents the main part of the autocorrelation of the series. The first argument of the function is mandatory and corresponds thus to the standardised IC data. It is set here to $\hat{\epsilon}_\eta(i_{IC}, t)$. The three following arguments define the block length values that are tested by the method. Those are simulated in the range [`bbl_start`, `bbl_stop`] with a certain step, `bbl_step`. Another important argument of the function is `BB_method`, which allows the user to choose a particular BB procedure among those presented in Section 3.2.4. Here, the moving block bootstrap is chosen by setting `BB_method = 'MBB'`. The other BB methods that are defined in the function are: `CBB` which stands for circular BB and `NBB` for non-overlapping BB. The block length of the matched BB can also be selected with the option `NBB`, since it is implemented with non-overlapping blocks in the package. Note that the stationary block bootstrap, which has a variable block length, is not defined in the function and in the whole package.

```
from cusvm import autocorrelations as acf

bb_length = acf.block_length_choice(dataIC_stn, bbl_start=1,
            bbl_stop=50, bbl_step=1, BB_method='MBB')
```

With this method, we select a block length equal to 8. As can be seen in Figure 5.4, this value corresponds well to the knee of the curve, i.e. the point where the curve changes, specifically from a high slope to a low slope.

Then, we select a target value for the magnitude of the shifts, also generically called "shift size", using the function `shift_size()` from the file `cusum_design_bb.py`. This function is based on Algorithm 5, which estimates iteratively the shift sizes of the OC series with the formula defined in (3.7.1). The target size is then selected as a specified quantile of the shift sizes distribution. The function contains several arguments, the two first of which are mandatory. The first parameter corresponds to the standardised data from the whole panel whereas the second is the index of IC series in the panel. The quantile of the shift sizes distribution is specified here at 0.4 (`qt=0.4`). While the default value is set to 0.5 to maximise the detection of both small and large deviations, we lower a bit the value here since unusually high deviations occur in some series during 2015. Otherwise, with the default value `qt=0.5`, we would obtain a target shift size close to 3.2, which is improper for detecting the smallest shifts of the data. An initial value for the desired average run length is also defined at 200 (`ARL0_threshold=200`), which will be the standard for this chapter. This function, as all functions that are presented in the following, also contains the argument `BB_method`, which allows the selection of a particular BB procedure. It takes values among: `MBB` which stands for moving BB, `NBB` for non-overlapping BB, `CBB` for circular BB and `MABB` for matched BB. As explained previously in Chapter 3, best performances are obtained when the chart is designed with the moving block bootstrap procedure. The moving BB is thus specified here and in the following with this command: `BB_method='MBB'` (it is also the default value in the function). Moreover, we pass the block length previously chosen to the function with: `block_length=bb_length`. Both arguments (`BB_method` and `block_length`) are thus recurrent and will be set to the same values in the following functions. Finally, the initial value for the shift size is chosen at two: `delta=2`.

```
from cusvm import cusum_design_bb as chart

delta_target = chart.shift_size(data_stn, pool,
            ARL0_threshold=200, delta=2,  qt=0.4,
                        block_length=bb_length)[1]

# delta_target = 2.24
```

The function returns a value that is close to 2. Although approximate, this value gives us a rough estimation of the magnitude of typical shifts in the data. It will be used in the following to calibrate the chart, for all types of shifts (jumps, drifts and oscillations).

Having selected the block length and the target shift size, the control limits of the CUSUM chart can finally be adjusted on the IC series by bisection searching using Algorithm 1. To this end, we call the function `limit_CUSUM()` from the file `cusum_design_bb.py`. This function has several arguments, all listed at the beginning of the function. Its first argument is the only one that is mandatory and corresponds to the IC standardised data. It is set here to $\hat{\epsilon}_\eta(i_{IC}, t)$. We

also specified the target shift size to the value previously found: `delta=2`. Since the distribution of $\hat{\epsilon}_\eta(i_{IC}, t)$ exhibits only a weak asymmetry (see Figure 5.9), the allowance parameter is let to its default value in the function: `delta/2`. Moreover, we set `BB_method='MBB'`, `ARL0_threshold=200` and `block_length=bb_length`, as previously explained. With these choices, the control limit is calibrated to reach a value of $ARL_0 = 200$ with an accuracy of $\rho = 2$ over $B = 4000$ runs. The accuracy (ρ) and the number of runs (B), which are not specified in the call of the function, are thus used with their default value (i.e. $\rho = 2$ and $B = 4000$). The control limit is then searched in the interval [`L_plus`, `L_minus`], where we define `L_plus`=20 and `L_minus`=0.

```
control_limit = chart.limit_CUSUM(dataIC_stn, delta=delta_target,
     ARL0_threshold=200, L_plus=20, L_minus=0, block_length=bb_length,
            BB_method='MBB', missing_values='omit')

#control_limit = 3.26
```

The last argument of the function that we have not yet mentioned is `missing_values` which is, as its name suggests, related to the missing data. It can take three values: `omit`, which suppresses blocks of data containing one or several missing values, `fill` which fills up the missing data with the mean of each series and `reset`. When specified to `reset`, the values of the CUSUM chart statistics are propagated through the small gaps, that are inferior than or equal to a specified number (passed though the argument `gap`). When the gaps are superior to the value of the argument `gap`, the charts statistics are then reset to zero. The option `reset` is thus particularly useful to treat data with different types of missing values or when the processes are expected to have substantially changed after long gaps and remained the same after the smallest ones. Since the daily production values do not have missing values, this argument does not impact the results. It is specified here to `omit`.
With this function, we calibrate the control limit of the chart at 3.3.

Note that the file `cusum_design_bb.py` also contains other functions to evaluate the performances of the chart. In particular, `ARL0_CUSUM()` allows us to compute the IC average run length of the chart, which corresponds to the rate of false positives (see Section 3.2.1). `ARL1_CUSUM()` can also be used to obtain the OC average run length, corresponding to the detection power of the chart, for different shift sizes and shapes.

5.4.3 Phase III: Estimation of shift sizes and shapes using SVMs

In this third and last phase of the monitoring method, the support vector machines for extracting and classifying out-of-control patterns are designed. Those methods

are composed of a SVR to predict the size of the shifts in a continuous range and a SVC to classify the shape of the encountered deviations among a pre-defined number of classes.

To this end, we first select an suitable value for the length of the input vector of the support vector machines (SVM) using Algorithm 4. This can be done automatically by calling the function `input_vector_length()` from the file `svm_training.py`. This function has three mandatory arguments. The first one is as usual the standardised IC data, which corresponds here to $\hat{\epsilon}_\eta(i_{IC}, t)$. The other ones are respectively `delta_min`: the target shift size and `L_plus`: the control limit of the chart, which were both selected in the previous subsection. Since few deviations are expected from the data (which have a high signal-to-noise ratio), we aim at detecting most of them. Hence, we also set `qt=0.95`, meaning that the length of the input vector should be selected as the 95th quantile of the OC run length distribution, for the chosen target shift size. The precise value of this quantile is not of great importance as long as an upper (≥ 0.5) quantile is selected. If unspecified, the quantile is selected by default as the knee of the upper quantiles curve.

```
from cusvm import svm_training as svm

input_length = svm.input_vector_length(dataIC_stn, delta_min=delta_target,
        L_plus=control_limit, block_length=bb_length, qt=0.95)
```

This function returns variable lengths around seven. Although seven would probably work as well, we select the length of the input vector at ten, to better detect the smallest drifts (which require large input vectors to be correctly identified).

As explained in Section 3.4.3, the SVM procedures have three main parameters: a kernel, a parameter λ which represents the trade-off between misclassification and regularization and ϵ which corresponds to an approximation error. After few tests (based on the same criteria than those used to select the parameter λ, see later), we decide to work here with the radial basis function kernel and with $\epsilon = 0.001$ (those are the default values). The parameter λ may then be automatically adjusted to the problem at hand using the function `choice_C()` from the file `svm_training.py`. This function selects λ (sometimes called C in the literature) to maximize the performance of the SVM classifier and regressor. To this end, it trains the classifier and regressor on simulated deviations for different values of λ in the range [`start`, `stop`] with a certain `step`. The function then returns the values of λ that maximize the accuracy for the classifier, the MSE and the MAPE (defined in Section 3.4.3) for the regressor. It has five mandatory arguments : `data` represents the standardised IC data, `L_plus` denotes the (positive) control limit of the chart, `delta_min` is the target shift size and `wdw_length` is the length of the input vector. The last argument, `scale`, corresponds to the scale parameter of the half-normal distribu-

tions that are used to randomly select the sizes of the artificial deviations[6]. Here, it is selected at four to reproduce the highest deviations of the series. The other (mandatory) arguments have already been chosen.

```
C_choices = svm.choice_C(data=dataIC_stn, L_plus=control_limit,
        delta=delta_target, wdw_length=input_length, scale=4,
      start = 1, stop = 10, step = 1,
         block_length=bb_length, BB_method='MBB')

#C value that minimizes the MAPE: 10
#C value that minimizes the MSE: 2
#C value that maximizes the accuracy: 7
```

With this method, we select a value for λ equal to 7, to obtain the best classification results.

Having defined the three main parameters of the SVM procedures, we can now train the classifier and regressor on artificial deviations. Since the deviations that were simulated in Chapter 3 are general, we do not modify the training and testing sets defined in Section 3.4.3. Hence, after randomly sampling series of IC data by the MBB procedure, three types of artificial deviations of size δ are added top of those series:

1. jumps: $x(t) = x_{ic}(t) + \delta$;
2. drifts with varying power-law functions: $x(t) = x_{ic}(t) + \frac{\delta}{T'}(t)^a$, where a is randomly selected in the range $[1.5, 2]$;
3. oscillating shifts with different frequencies: $x(t) = x_{ic}(t)\delta \sin(\eta\pi t)$, where η is randomly selected in the range $[\frac{\pi}{m}, \frac{3\pi}{m}]$.

Although other types of deviations such as spikes or drifts with different power-law functions could be added in the simulations, those appear to produce satisfactory results. They were designed to allow the identification of the various kinds of deviations in the sunspot numbers and appear to be generalisable to other datasets as well.
In practice, the function `training_svm()` from the file `svm_training.py` does both the training and the testing of the classifier and regressor. It has the same mandatory arguments as the function `choice_C()` that we see previously. It has also an optional argument, `C`, that allows the user to specify to the value of the parameter λ, formerly chosen at 7. We also set `delay`= `2input_length` to start the monitoring after a random delay in the range [`input_length`, `2input_length`]. This parameter has two main purposes: (1) allowing the SVM methods to identify

[6]In the function, the shift sizes are randomly sampled from two half-normal distributions (Evans et al., 2000) supported by $[-\infty, ..., -$`delta_min`$]$ and [`delta_min`$, ..., \infty]$ respectively, with a scale parameter equal to the argument `scale`.

shifts that occur anywhere within the input vector and (2) helping the methods to detect large shifts by letting the deviations build-up before being sent to the SVMs. This parameter was selected to `delay`= 3`input_length/2` in Chapter 3 (which corresponds to a random delay in the range [`input_length`, 3`input_length/2`]) but is augmented here since the input vector is much shorter. This allows the deviations to grow a bit before the start of the monitoring and helps thus the SVMs to detect high deviations.

```
reg, clf = svm.training_svm(data=dataIC_stn, L_plus=control_limit,
        delta=delta_target, wdw_length=input_length, scale=4, C=7
     delay=2*input_length, block_length=bb_length, BB_method='MBB')
```

The function returns the trained classifier and regressor, which can be saved for reproducibility purpose

5.5 Results

In the previous section, we adjust the parameters of the CUSUM chart and train the SVM methods on the photovoltaic energy production data. We will now apply the method that is fully calibrated to actually monitor those data. We first present results for the daily values and then we move on to 15-minutes data.

5.5.1 Daily values

The function `alerts_info()` from the file `alerts.py` can be used to apply the complete procedure (CUSUM chart plus SVM methods) on a particular series of the panel. It has six mandatory arguments, that were all chosen in the previous section. The first one is `data`, which represents the standardised series that we want to monitor. Then, the following arguments are respectively `L_plus`: the (positive) control limit of the chart, `delta`: the target shift size, `wdw_length`: the length of the input vector, `clf`: the trained classifier and `reg`: the trained regressor. It has also an optional argument `cut`, which is set by default to `2L_plus`. This argument defines the maximal value that the chart statistics are allowed to take, i.e. $|C_j^+|$, $|C_j^-| \leq 2L$. It thus prevents the chart to take too high values and therefore stay in alert for longer periods than the actual deviations of the series.

```
from cusvm import alerts as appl

data_indv = data_stn[:, region] #choose a series

[form_plus, form_minus, size_plus, size_minus,
C_plus, C_minus] = appl.alerts_info(data_indv, control_limit,
            delta_target, wdw_length, clf, reg)
```

Six quantities are returned by the function: the positive and negative shapes of the shifts (`form_plus` and `form_minus`), sizes of the shifts (`size_plus` and `size_minus`) and chart statistics (`C_plus` and `C_minus`).

Those can then be passed to another function, `plot_3panels()` from the file `alerts.py`, to show the result of the monitoring. This function has ten mandatory arguments. In addition to the six ones previously mentioned (`form_plus`, `form_minus`, `size_plus`, `size_minus`, `C_plus` and `C_minus`), it also contains: `data` representing the standardised series that we want to monitor, `L_plus` which is the (positive) control limit of the chart, `time` denoting the time vector and `name` corresponding to the index of the series (here it is the name of the distribution system operator (DSO)). This function returns a plot composed of three panels, which shows respectively (1) the standardised series of data which is analysed, (2) the CUSUM chart statistics ($|C^+|$, $|C^-|$) and (3) the shapes and sizes of the shifts that are predicted by the SVM when the chart is in alert. The optional arguments `time_start` and `time_stop` can also be used to display the results on a subset of the total period studied.

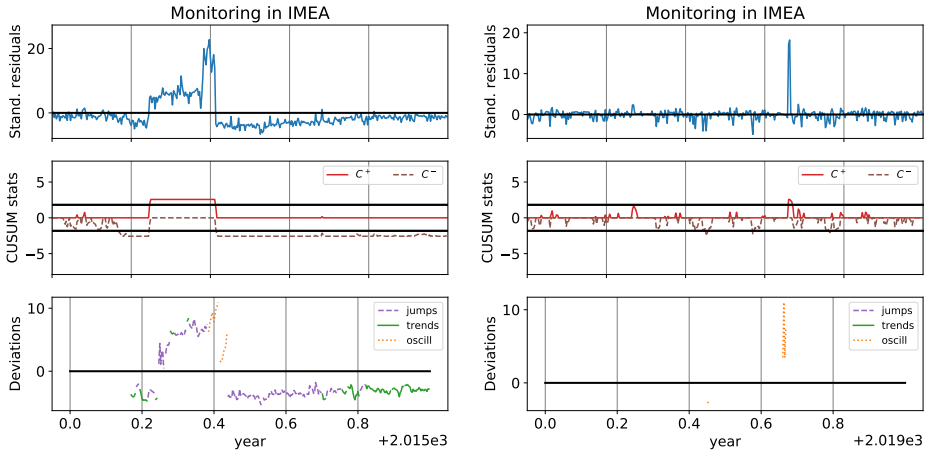
```
fig = appl.plot_3panels(data_indv, control_limit, time,
                        form_plus, form_minus, size_plus, size_minus,
                        C_plus, C_minus, names[region], time_start=2015,
                        time_stop=2016)
```

A complete script is provided in Appendix 5.7.2, to show the successive calls of functions for the whole monitoring process.

This function is applied here to the data from the DSO IMEA and the results are shown in Figure 5.5. Two main deviations are highlighted in this figure. A first unusual long-term deviation occurred on March 20, 2015 and lasted until May 31, 2015. After few investigations, it appears to be related to a problem in the estimation of the monitored capacity. Indeed, this capacity suddenly drops from 61.17 MWp[7] to 34.23 MWp on the 10th of March and then jumps once again from 34.23 MWp to 80.05 MWp during the 20th of May. As the deviation that is seen is shifted with respect to those changes, there are probably some delays between a variation in the monitored capacity and its actual effect on the load factors. Having not access to the complete processing of the load factors yet, we cannot study further the problem. Such a deviation also appeared six years ago, and as seen with the sunspot numbers, deviations that occurred in the past are often more complicated to identify than the most recent ones. Similar deviations over the same period are also visible in the data from Resa (shown in Figure 5.6a) and IVEG, which are located in different regions (Resa manages the electricity in Liege whereas IMEA

[7]MWp stands for Megawatt-Peak. It is a common unit for measuring the power capacity of an installation. It corresponds to the Megawatt produced under standard ideal conditions.

and IVEG operate in Antwerp). Hence, we suspect that those deviations are more related to transmission problems than to incorrect measurements.
Another major deviation is visible in IMEA. It is brief and lasted two days from August 30, 2019 till August 31, 2019. This deviation does not appear in the data from other DSOs and is corrected in the most recent version of the data. Hence, we cannot say if it is related to inconsistencies in the monitored capacity or in the real-time measures. After verification, it can however not be attributed to the weather conditions since both days were sunny in the whole country.

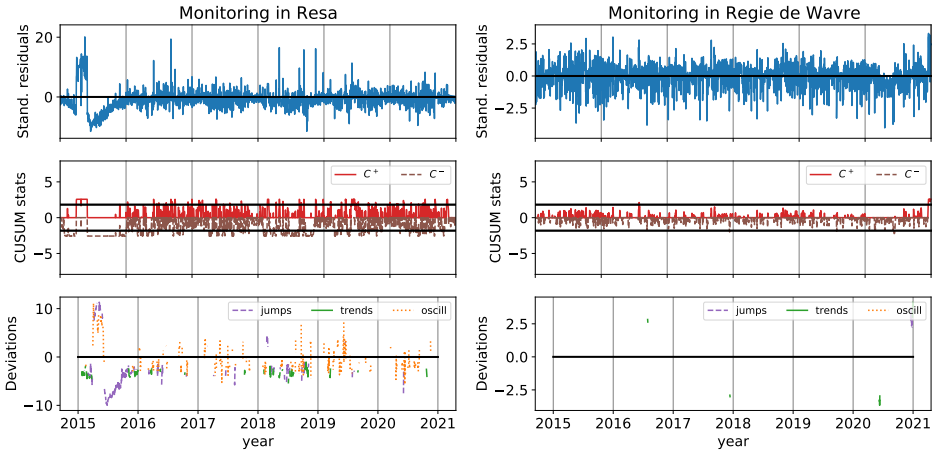


a b

Figure 5.5: (a) Upper panel: the standardised daily data $\hat{\epsilon}_\eta(i,t)$ for the DSO IMEA over the year 2015. Middle panel: the (two-sided) CUSUM chart statistics applied on the $\hat{\epsilon}_\eta(i,t)$ in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure over the year 2019.

The monitoring method is also applied to two other operators, Resa and Regie de Wavre, in Figure 5.6. As can be seen, the data from Resa experience the same kind of long-term deviation in 2015 than IMEA. It also contains several other short-lived deviations, that may be related to transient meteorological disturbances. For those reasons, both operators were classified as out-of-control by the clustering procedure of Section 5.4.1. On the contrary, the operator Regie de Wavre is a stable DSO which is included in the pool. It triggers almost no alert on the entire period studied.
As can be seen in the figures, the sizes of the deviations predicted by the support

vector machines appear to correspond well to the pace of the standardised data. The classification results are more complex to analyse. When observed in details however, they correspond most of the time to our (visual) expectations. The deviations are mostly classified as jumps and oscillations although several drifts are detected. To be better identified, the data should be monitored at a larger scale than one day. The length of the SVM input vector could also be extended to better detect the drifts.



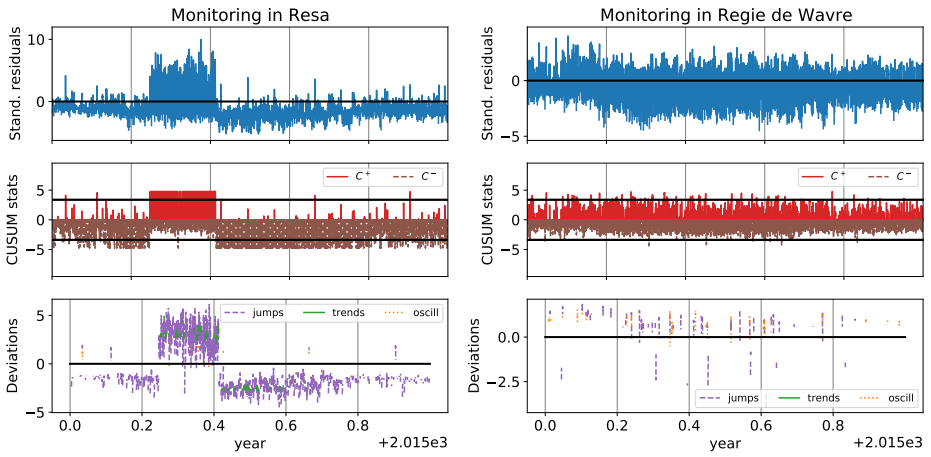
a b

Figure 5.6: (a) Upper panel: the standardised daily data $\hat{\epsilon}_\eta(i,t)$ for the DSO Resa over the period studied (2015-2021). Middle panel: the (two-sided) CUSUM chart statistics applied on the $\hat{\epsilon}_\eta(i,t)$ in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure for the DSO Regie de Wavre over the same period.

5.5.2 15 minutes values

We now turn our attention to the data obtained each quarter of an hour. A similarly procedure has been applied to these data as the one which was previously described for monitoring the daily values. We first convert all zeros (which correspond principally to the nights) into missing values to simplify the design of the method. Those zeros are related to the absence of solar radiations and may thus be considered as missing data in that sense. Then, we apply the same preprocessing as those described in Section 5.3.3, i.e. we first compute the median of the rescaled

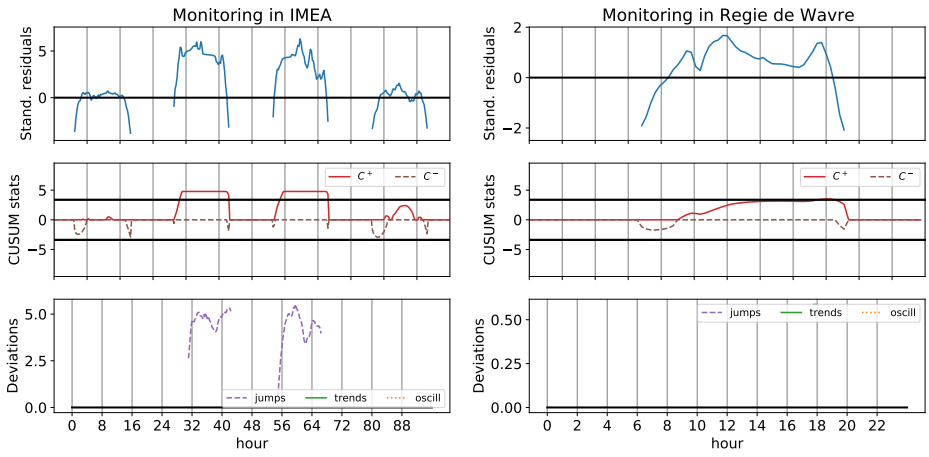
data, divide the production data by this median and apply a final rescaling on this ratio. We thus obtain an estimation of the mean of the localized variations that is monitored here.
To this end, we cluster in phase I SPC the data using the k-means algorithm and obtain an IC pool composed of 17 DSOs. The operators IMEA, IVEG, Ores Est, Ores Luxembourg, Ores Verviers and Resa were thus excluded from the pool. The data are then standardised by K nearest neighbours regression method with $K = 850$. In phase II SPC, we calibrate the CUSUM chart using the MBB method with a block length equal to 14. A longer block length is thus selected here since the quarter of an hour data experience longer-term correlations than the daily values. The correlation is indeed higher throughout a day than between consecutive days. After selecting a target shift size equal to one, the control limit of the CUSUM chart is fixed at 11.46 to reach an ARL_0 of 200. In the last phase of the procedure, the support vector machines are trained and validated on the same kinds of simulations as those used with the daily values, for an input vector of length equal to 36 and a parameter λ chosen at 10.



a b

Figure 5.7: (a) Upper panel: the standardised quarter of an hour data $\hat{\epsilon}_\eta(i,t)$ for the DSO Resa over the year 2015. Middle panel: the (two-sided) CUSUM chart statistics applied on the $\hat{\epsilon}_\eta(i,t)$ in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure for the DSO Regie de Wavre over the same period.

The results are shown in Figure 5.7 for the DSOs Resa and Regie de Wavre. As can be seen, many more jumps are detected here than with daily data, as expected. The behaviour of the data stays however essentially the same. The data from Resa, that was classified as out-of-control by the clustering procedure, experience large deviations in 2015 whereas the DSO Regie de Wavre is stable on the same year (as well as on the entire period studied).
Figure 5.8a shows the deviation that occurred on August 2019 in IMEA. As can be seen, abnormally high values were recorded both on the 30th and 31th of August. The 29th of August and the 1st of September are also represented in the figure, as a comparison. As explained above, the cause of such deviation is still unknown but is very unlikely to be related to the weather. Figure 5.8b does not show any particular deviation but a small decrease (that is too low to trigger an alert) on the 20th of March, 2015 around 10am. This small drop in production was caused by a partial solar eclipse and illustrates how sensitive the production is to changes of solar radiations. It also highlights how high frequency data can be used to gather more precise information about a shift. For practical use, we therefore recommend to monitor low frequency data such as here daily values whereas a higher frequency should be used to analyse some particular deviations in details.



a b

Figure 5.8: (a) Upper panel: the standardised quarter of an hour data $\hat{\epsilon}_\eta(i,t)$ for the DSO IMEA from August 29, 2019 till September 1, 2019. Middle panel: the (two-sided) CUSUM chart statistics applied on the $\hat{\epsilon}_\eta(i,t)$ in *square-root scale*. The control limits of the chart are represented by the two horizontal thick lines. Lower panel: the characteristics of the deviations predicted by the SVR and SVC after each alert. (b) Similar figure for the DSO Regie de Wavre on Mars 20, 2015.

Note that we monitor here the data at two different scales by averaging the quarter of an hour data into daily values. In Chapter 3, we apply a monitoring at multiple frequencies to the data by using a moving average (MA) filter with different window lengths instead. The MA approach was particularly suited for the sunspot numbers since they contain a short-term component that was excluded from the monitoring (at both scales considered). Here however, we do not have such a short-term component, which should be excluded for the control procedure. Averaging the data also reduces their number which is quite consequent in the period studied. There are indeed around 213000 values for the quarter of an hour data over 2015-2020 against around 2200 for the daily values. There are thus different ways to apply a multi-scale monitoring on a panel of data.

5.6 Conclusion

In this chapter, we successfully apply the method developed in Chapter 3 to the photovoltaic energy production. Apart from the preprocessing of the data described in Section 5.3.3, the monitoring method is used without changes as compared with the sunspot numbers. The parameters only are adjusted to the photovoltaic energy data. Indeed, the distribution and the autocorrelation of the data differ from one application to another. The types of deviations and missing values also vary with the monitoring problem at hand. The parameters of the monitoring should thus be adjusted for each particular application. For this purpose, several algorithms are proposed. Most of them are fully automated or require only few inputs from the user. They were applied here to select appropriate values for the parameters, which simplifies the design of the method. In practice, the method also appears to be robust to the particular choices of many parameters. The training sets of SVM procedures, which were expected to be particularly dependent on the deviations of the data, appear to generalise well on a completely different application.

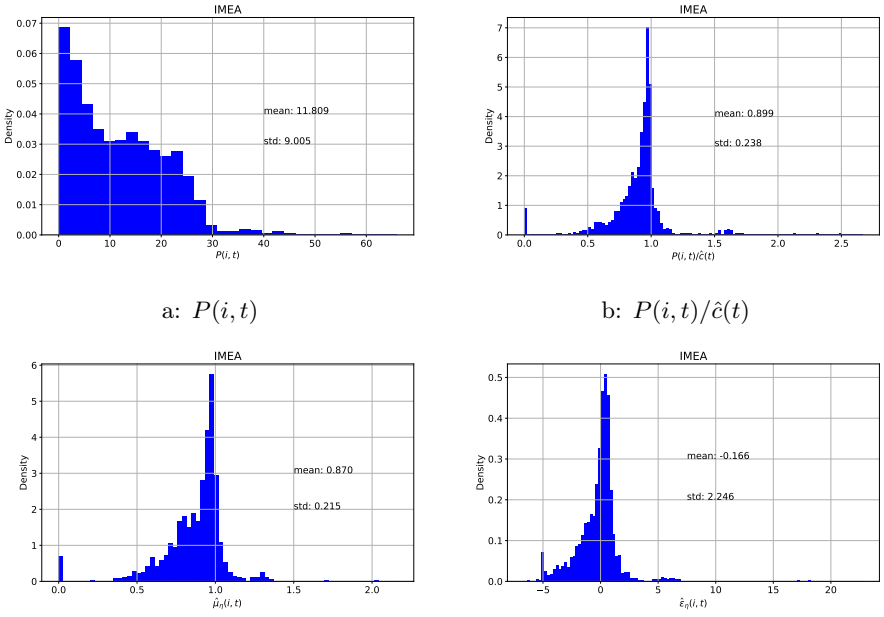
Moreover, the package that is provided with the method can be installed easily, with all its dependencies. It is composed of functions that allow the application of the CUSVM method but also the preprocessing of different data. Those functions often contain several arguments, which allow a large flexibility in the design. Four different block bootstrap methods are for instance implemented in the package. The user can also choose between three procedures to deal with the missing values and the clustering method can be selected among seven. We therefore truly hope that the method will be employed by other users in the future, since it has the potential of being applied to a large variety of problems.
Regarding the supervision of the photovoltaic energy production, two different scales were monitored and several deviations were detected in the data. Although the origin of some deviations are already identified, we will continue our investi-

gations in the following, in a closer collaboration with Elia. As the solar energy production is expected to grow in the future, an increasing supervision of the data will be required. The incoming set-up of smart meters will also allow the DSOs to acquire more information about the solar energy. The consumption of the energy produced may for instance be collected in the future and supervised as well, for a better power system management.
In the meantime, we will continue to use the method for solving other monitoring problems.

5.7 Appendix

5.7.1 Figures related to the model

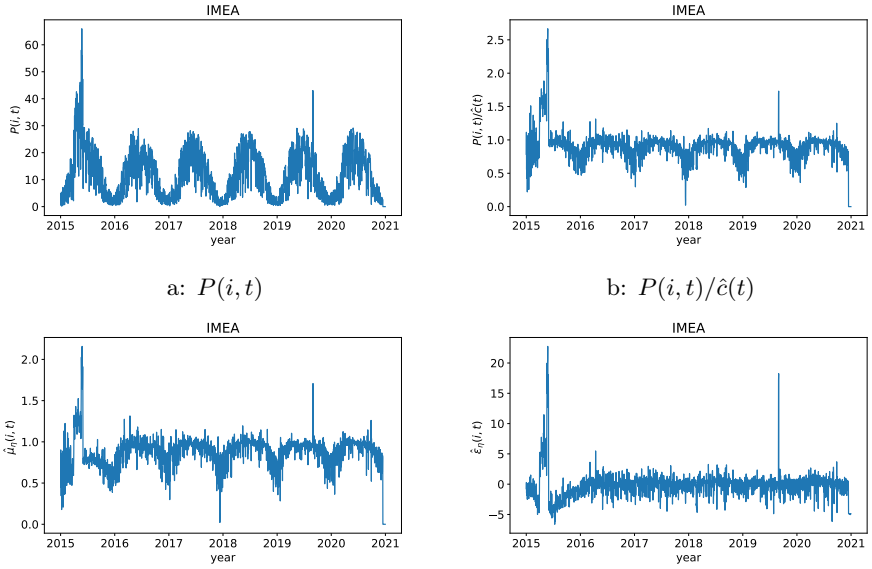
Figures 5.9 and 5.10 represent the different stages of the estimation of the localized variations of the daily photovoltaic energy production, $\hat{\mu}_\eta(i,t)$, described in Section 5.3. They also display in the last panel the standardised variations, $\hat{\epsilon}_\eta(i,t)$, obtained after applying the procedure of Section 5.4.1. Figure 5.9 shows in particular the histograms of all intermediate quantities until the standardised variations are obtained whereas Figure 5.10 represents those quantities as function of time. Those results are shown for the distribution system operator IMEA, which is typically considered as unstable or out-of-control (OC). It experiences two major deviations: a 2-days jump at the end of August 2019 and a long-term deviation that lasted around three months (March-May) in 2015.



a: $P(i,t)$ b: $P(i,t)/\hat{c}(t)$

c: $\hat{\mu}_\eta(i,t)$ d: $\hat{\epsilon}_\eta(i,t)$

Figure 5.9: Different steps to estimate the localized variations of the daily photovoltaic energy production (η of (5.3.1)) for the distribution system operator IMEA. (a) Histogram of the daily data, $P(i,t)$. (b) Histogram of the ratio $P(i,t)/\hat{c}(t)$. This step removes the main part of the solar signal from the data. (c) Histogram of the localized variations, $\hat{\mu}_\eta(i,t)$, after compensating for the different levels of the regions. (d) Histogram of the standardised variations, $\hat{\epsilon}_\eta(i,t)$.



a: $P(i,t)$ b: $P(i,t)/\hat{c}(t)$

c: $\hat{\mu}_\eta(i,t)$ d: $\hat{\epsilon}_\eta(i,t)$

Figure 5.10: Different steps to estimate the localized variations of the daily photovoltaic energy production (η of (5.3.1)) for the operator IMEA. (a) The daily values, $P(i,t)$, as a function of time. (b) The ratio $P(i,t)/\hat{c}(t)$ as a function of time. This step removes the main part of the solar signal from the data. (c) The localized variations $\hat{\mu}_\eta(i,t)$ as a function of time, after compensating for the different levels of the regions. (d) The standardised variations $\hat{\epsilon}_\eta(i,t)$ along time.

5.7.2 Script

In this section, we show a complete script that can be executed (unaltered) after installing the package as described in Section 5.2. This script applies the entire monitoring procedure to the daily energy production data and displays the result for the DSO Resa. It reproduces thus Figure 5.6a.

```
import pickle
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
plt.rcParams['font.size'] = 14
import pkg_resources as pkg

from cusvm import preprocessing as pre
from cusvm import autocorrelations as acf
from cusvm import cusum_design_bb as chart
from cusvm import svm_training as svm
from cusvm import alerts as appl

### load data (loaded automatically with package)
data_path = pkg.resource_filename(pkg.Requirement.parse("cusvm"), 'data')
df = pd.read_csv(data_path + '\PVdaily.csv')
data = np.array(df)[:,1:]
data = data.astype('float')
data = data/96
names = list(df.columns)[1:]
(n_obs, n_series) = data.shape


### load time
with open(data_path + '/time_daily', 'rb') as file:
    my_depickler = pickle.Unpickler(file)
    time = my_depickler.load() #data every day


#=============================================================
### Preprocessing
#=============================================================

### rescaling
data_rescaled, k_factors = pre.rescaling(data, period_rescaling=365)

### median
med = pre.median(data_rescaled)

### remove common signal
ratio = pre.remove_signal(data, model='multiplicative', ref=med)

mu_eta, chi_factors = pre.rescaling(ratio, period_rescaling=365)


### select an IC pool
```

```
pool = pre.pool_clustering(mu_eta, method='kmeans')
names_IC = [names[i] for i in range(n_series) if i in pool]
names_OC = [names[i] for i in range(n_series) if i not in pool]

mu_eta_IC = mu_eta[:, pool]

### standardise the data
K = pre.choice_K(mu_eta, mu_eta_IC, start=50, stop=2000, step=50) #400
data_stn, dataIC_stn = pre.standardisation(mu_eta, mu_eta_IC, K)


#=================================================================
### design of the chart
#=================================================================

### choice of the block length
bb_length = acf.block_length_choice(dataIC_stn, 1, 50, 1) #8

delta_init = 2 #intial value for the target shift size
ARL0 = 200 #pre specified ARL0 (controls the false positives rate)

### adjust the control limits
delta_target = np.round(chart.shift_size(data_stn, pool,
         delta=delta_init, ARL0_threshold=ARL0,
        block_length=bb_length,  qt=0.4, missing_values ='omit')[1]) #2


control_limit = chart.limit_CUSUM(dataIC_stn, delta=delta_target,
     L_plus=4, block_length=bb_length, missing_values='omit') #3.3


#=================================================================
### train support vector classifier and regressor
#=================================================================

### select the length of the input vector
wdw_length = svm.input_vector_length(dataIC_stn, delta_target,
        control_limit, block_length=bb_length, qt=0.95) #~7

wdw_length = 10 #fixed value

scale = 4
### find an optimal value for C (regularization parameter)
C_choices = svm.choice_C(dataIC_stn, control_limit, delta_target,
    wdw_length, scale, start = 1, stop = 11, step = 1,
    delay=wdw_length, block_length=bb_length, confusion=False) #10,2,7

C = C_choices[2] #accuracy

### train the classifier and regressor with selected C and kernel
reg, clf = svm.training_svm(dataIC_stn, control_limit, delta_target,
                wdw_length, scale, delay=wdw_length*2, C=C,
                                block_length=bb_length)
```

```
#=====================================================================
### run the control chart and plot results (with svm predictions)
#=====================================================================

region = [i for i in range(len(names)) if names[i] == 'Resa'][0]

for i in range(region, region+1):
#for i in range(n_series):

    data_indv = data_stn[:,i] #monitored series

    [form_plus, form_minus, size_plus, size_minus,
    C_plus, C_minus] = appl.alerts_info(data_indv, control_limit,
            delta_target, wdw_length, clf, reg)

    fig = appl.plot_3panels(data_indv, control_limit, time,
                      form_plus, form_minus, size_plus, size_minus,
                      C_plus, C_minus, names[i], time_start=2015,
                      time_stop=2021)
```

CHAPTER 6

Automated sunspots detection on white light images

We previously developed a general monitoring method. As a quality control procedure, this method enters into the broader field of automation, which groups all methods and technologies that tend to reduce human intervention. In general, the automated methods have several advantages over manual procedures. They often complete faster their tasks, in a more reliable and reproducible manner. They also produce more robust results, with their main disadvantages being their lack of flexibility and high initial costs (in time or money).

To this end, after constructing an automated procedure for monitoring panels of data such as the sunspot numbers, we turn our attention to the observations themselves. We propose a preliminary version of a method for automatically extracting spots and groups from white light images of the Sun. Contrarily to other existing automated procedures, the method works on images that are gathered at the ground, which are more easily obtained and at lower cost than images recorded in space. They also share more characteristics with the observations and are thus more appropriate to preserve the continuity of the series over time.

The proposed method has been primarily developed to improve the quality of the sunspot numbers by being used conjointly with the observations and not for completely replacing them. It is applied in the following to images recorded in the station Uccle, which also centralizes the sunspot numbers (N_s, N_g and N_c) from the different stations and produces the ISN. First results indicate that the method performs sufficiently well to merit future attention.

6.1 Introduction

Nowadays, the spots and groups are manually observed and counted every day in several stations around the world. This task is performed by skilled amateurs but also professional observers and requires several minutes up to several hours of their time on a daily basis. As demonstrated in Chapter 2, the counts also suffer from significant uncertainties and errors that may be directly related to the observers. The short-term error appears for example to be linked to the number of observers. It is usually smaller in stations where there is a single observer rather than a team, alternating the observations on a regular basis. Moreover, some stations have intrinsic bias that may be related to the counting method of the observers. Large spots have for instance more weight than small ones in the counting procedure applied in the Observatory of Locarno. It may thus be interesting to search for more automated methods, that are clearly defined. Although several algorithms have been developed in the literature for this purpose, they are designed to work on images recorded in space.
Hence, we investigate in the following if an automated method for extracting and counting the number of spots and groups can be designed from ground-based images. We thus develop a draft version of such a method and see how it performs on images recorded in the station Uccle, whose dataset is at the centre of this work. The proposed algorithm is then compared with manual observations and with another automated method, to better study its performances and limitations.
We emphasis that the objective here is not to propose a fully operational method but to construct a preliminary version of an automated algorithm. If it shows significant potential, this method may then be improved in future research.

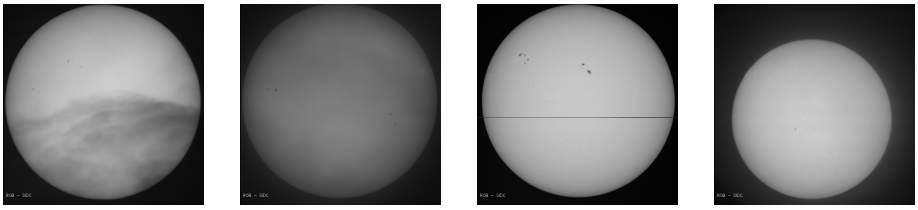
This chapter is organised as follows. In Section 6.2, we detail the main characteristics of the images that will be analysed. Some common morphological operations are also briefly reviewed in Section 6.3. Our extraction algorithm is then described in Section 6.4. The two following Sections (6.5 and 6.6) are devoted to the automated procedures to count respectively the spots and groups from the extracted areas. The complete procedure is then applied on the images from year 2003 till year 2020. The main results are presented in Section 6.7 with some comments.

6.2 Characteristics of the images

The images that are analysed in this chapter extend from year 2003 till year 2020 and are distributed through the Solar Influences Data Analysis Centre (SIDC) [1]. Those are taken from the ground with the white light telescope of the Uccle solar

[1] http://sidc.oma.be/

equatorial table (USET) in Belgium (Uccle). We analyse images with a resolution of 750×750 pixels for convenience, since they can be automatically downloaded from the SIDC website with a simple computing program. The counting and extraction methods run also faster on images of 750×750 pixels than on those with a smaller resolution.



a: Partial clouds b: Clouds c: Dead pixels d: Disk off-centered

Figure 6.1: Examples of images affected by different types of degradations.

The methods that we propose, such as all methods that have already been developed in the literature, rely on intensity contrasts. The sunspots, which are darker than their surroundings, are discriminated from the background using an intensity threshold. A simple threshold-based procedure is however not adapted to detect spots in the SIDC white light images for the following reasons. First, the quality of these images depends on the luminosity of the Sun, which varies along the year. The quality is also degraded by atmospheric perturbations such as clouds which may partially or totally conceal the Sun. They may produce irregular patterns on top of the Sun, which can be confounded with sunspots as can be seen in Figures 6.1a and 6.1b. Second, the limb darkening effect — an optical effect that is responsible for the darkening of the edges of the Sun with respect to its centre (Neckel and Dietrich, 1994) — also affects the detection of sunspots in the limits of the disk. Third, the sunspots have various shapes, sizes and intensities —varying along the sunspots' life— which complicates furthermore their detection. Moreover, few images suffer from failures in the CCD sensors which record the images as in Figure 6.1c. Some others such as those of Figure 6.1d are also off-centred due to e.g. the wrong orientation of the instruments with respect to the Sun. The method requires thus several steps to accommodate the aforementioned features/defaults of the images.

The automatic extraction algorithm that we developed is based on mathematical morphology tools. It is inspired by the Sunspot Tracking And Recognition Algorithm (STARA) developed by Watson et al. (2009), which is in turn based on the work of Curto et al. (2008). We first present common morphological operations in the following and then we explain our extraction and counting methods.

6.3 Morphological operations

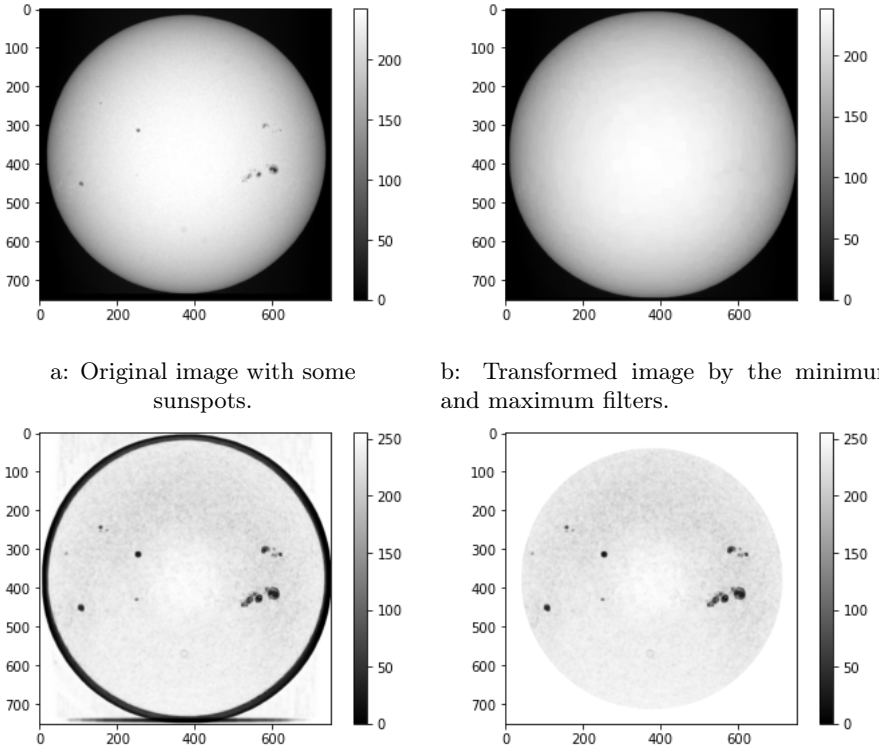
- *Erosion* $X \Theta B$:
The erosion of a grey-scale image X by a structuring element (SE) B is a new image where all pixel intensities have the minimum value of the pixel intensities in their neighbourhood. This transformation shrinks the small bright elements of an image. An erosion may be applied by using a minimum filter. This filter acts by translating a SE of a pre-defined size over the image. At each step, the pixel intensity at the centre of SE is replaced by the minimal intensity computed over its neighbouring pixels inside the SE domain.

- *Dilatation* $X \oplus B$:
The dilatation of a grey-scale image X by a SE B results in a new image where all pixel intensities have the maximum value of the pixel intensities in their neighbourhood. This operation enlarges the bright elements of an image. A dilatation may be applied by using a maximum filter. This filter displaces a SE over the image and replaces the pixel intensity at the centre of SE by the maximal intensity computed over its neighbouring pixels inside the SE domain.

- *Opening* $X \circ B = \{(X \Theta B) \oplus B\}$:
The opening of an image is defined as an erosion followed by a dilatation. This transformation suppresses the brightest regions, smaller than the SE, of the image.

- *Closing* $X \bullet B = \{(X \oplus B) \Theta B\}$:
The closing of an image is defined as a dilatation concatenated with an erosion. This transformation suppresses the darkest regions, smaller than the SE, of the image.

- *Top-hat* $X - X \bullet B$:
The top-hat transformation subtracts the closing image from the original image. The closing image is similar to the original but does not contain the darkest regions of the image. Therefore, subtracting the closing image from the original results in a new image which only contains the erased objects.



a: Original image with some sunspots.

b: Transformed image by the minimum and maximum filters.

c: Difference between the original and the transformed images (similarity index (SSIM) values).

d: Resulting image after isolating the solar disk from the background.

Figure 6.2: Preprocessing whose aims are to identify the solar disk and highlight the spots of the image.

6.4 Extraction algorithm

6.4.1 Solar disk detection and structures enhancement

The first step of the algorithm is to identify the solar disk in the images. This is not a trivial task since the apparent diameter of the Sun changes during the year. Moreover, the position of the centre of the Sun also changes from one image to another due to the orientation of the camera with respect to the Sun, as seen in Figure 6.1d. Different methods have been used in the literature for this purpose. Zharkova et al. (2003) used for instance a 'canny edge' detector while Curto et al.

(2008) applied a combination of an erosion, a gradient transformation and a threshold. Once the solar disk is located, the second step is to enhance the structures of the images with respect to their surroundings, to bring the sunspots out before applying an intensity threshold. To this end, after determining the limits of the solar disk, Watson et al. (2009) applied a top-hat transformation on the images to locally enhance the darkest regions.

Inspired by those procedures, the two aforementioned steps are mixed into a four stages procedure in our method, which is illustrated in Figure 6.2. The original image is displayed in Figure 6.2a. (1) We first apply an erosion (minimum filter) with structuring element (SE) of size equal to 10×10 pixels to eliminate the noise (i.e. the white pixels) in the image. (2) We also apply a dilatation (maximum filter) with particular SE of size superior than or equal to 10×10 pixels to remove the spots from the image. Figure 6.2b shows the resulting image after applying those filters. (3) Then, the differences between the original image and the transformed image are detected using a structural similarity index measure (SSIM) (Wang et al., 2004) [2]. The values of this index are represented in Figure 6.2c., where the darkest pixels correspond to the largest differences in the images. (4) Finally, a circle detector based on the Hough Gradient Method (Davies, 1988) is employed to automatically find the position of the centre and the radius of the circle [3]. The interior of the solar disk is then isolated in the images, as shown in Figure 6.2d.

Since the size of the sunspots varies over a wide range, we select the size of the dilating SE for each image. Indeed, large SEs are not adapted to remove local structures caused for example by clouds (which are even larger and may cover the entire image). They may transform those structures in such manner that they are confounded with spots. Whereas small SEs may only remove parts of the largest spots. Hence, we design an automated procedure for selecting the size of the dilating SE over the range 10×10 to 60×60 pixels. For each value of SE, the erosion (step 1) and the dilatation with the chosen SE size (step 2) are first applied to the image. The coordinates of the solar disk are then retrieved with the circle detector previously mentioned. Then, the spots on the interior of the Sun are extracted using an intensity threshold. To not being affected by the limb darkening effect, the spots located only within a smaller circle of radius equal to two third of the radius of the whole solar disk are taken into account. In this smaller circle, the pixels whose intensity is smaller than the mean intensity of the image minus 3 standard deviations are kept. Since the purpose of the dilation is to remove the spots from the image, the size of the SE is finally selected as those which gives the

[2] The structural similarity index is obtained with the function `structural_similarity()` which is implemented in the python package `skimage` (Van der Walt et al., 2014).

[3] The circles are detected with the function `HoughCircles()` from the python package `opencv` (Bradski, 2000).

minimal number of spots on the Sun. Once the appropriate size of the dilating SE is selected, the structural similarity index and the circle detector are then applied to the image in step 3 and 4 to finalise the preprocessing of the image.
Note that we also apply a security margin (which depends on the image) around the limits of the solar disk, to remove the darkest part of the disk. With this margin, the detection of the spots is protected against the limb darkening effect.

6.4.2 Sunspots detection

At this point of the method, the solar disk is fully identified and the prominent structures are enhanced from the background, as can be seen in Figure 6.2d. The sunspots may then be extracted using an adaptive threshold intensity similar to those described in Watson et al. (2009). In this last step, the appropriate value for the threshold is computed for each image as follows. Starting from a low value of the threshold, the pixels darker than this threshold are extracted from the image. Among those, the number of disjoint pixels are counted (a pixel in the middle of an image is assumed to have 8 direct neighbours). They correspond to potential disjoint spots. The procedure is then iterated for higher values of the threshold until the number of (disjoint) spots explodes, meaning that the background level is reached. In practice, the threshold varies between $\overline{I} - 7std(I)$ and $\overline{I} - 2std(I)$, where I is the image, $\overline{I}$ is the image mean intensity and std denotes the standard deviation. The appropriate threshold is finally selected as the first value at which the number of (disjoint) spots becomes stable.

We observe that this algorithm tends to over-detect the spots at solar minima. This effect is somehow similar to the tendency of the observers to over-scrutinize the Sun over the same periods. Indeed, the algorithm is based on an intensity threshold that depends on the image. Hence, the method may identify the darkest regions of an image as spots, even if this image does not contain any actual spot. To correct this effect, we use the mean structural similarity index measure (MSSIM) (Wang et al., 2004) over the image as a primary criterion to discriminate images that actually contain spots. This index is computed previously in the algorithm, when the original image is compared to the one that has been transformed by an erosion and a dilatation in step 3. It measures the similarity between both images. The MSSIM takes values between -1 and 1, where a score of 1 indicates that both images are identical and 0 means that the images have no structural similarity. Since a high value of the MSSIM most probably indicates that the image does not contain spots, we select the intensity threshold as $\overline{I} - 8std(I)$ when the MSSIM is larger than or equal to 0.98. This low value for the intensity threshold should then prevent the algorithm from detecting too many “false” spots. Whereas, when the MSSIM is smaller than 0.98, the intensity threshold is selected as previously explained.

6.5 Number of sunspots

Once the darkest pixels of the image have been retrieved, they should be assigned to their own sunspot. To that end, region-growing algorithms based on a neighbourhood criteria are often used in the literature (Curto et al., 2008). Similarly, we design an algorithm that browses the entire image. When a pixel that was marked as a spot candidate (when its intensity is lower than the pre-selected threshold) is detected, the pixel is associated to the spot attached to at least one of its eight direct neighbours. If the pixel neighbours do not belong to a spot or are marked as background, the pixel is attributed to a new spot. If, at some point in the algorithm, a pixel is surrounded by pixels belonging to two different spots, these spots are simply merged into a unique larger spot.
This algorithm then computes the number of disjoint spots in the images.

6.5.1 Umbra and penumbra

As stated in the introduction, large spots usually have an internal region, called the umbra, that is darker than the remaining part of the spot, called the penumbra. Such a spot is represented in Figure 1.5. Some records contain distinct information about the umbra and the penumbra but most of them consider both as a single entity. An automated algorithm could however benefit from such physical information to better identify overlapping spots. This may by particularly useful at solar maxima, when the spots are typically more numerous and larger than during other parts of the solar cycle.
To this end, Watson et al. (2011) developed a method that is able to distinguish the umbra from the penumbra based on their different intensities. They draw the histogram of the pixels intensity, which has two peaks, and identify the local minimum between these peaks, which corresponds to the intensity at the edge of the umbra.

We complete our automated counting procedure by designing a different method based on the self-organizing map (SOM) (Kohonen, 1990), a well-known dimension-reduction technique. The method may be explained as follows. A SOM is first applied on each disjoint spot to reduce its intensity into six different levels as represented in Figure 6.3. Then, we only look at the lowest level of intensity (represented in dark blue in the figure) and the number of disjoint pixels are counted for this level. Those correspond to the darkest parts of the spot that are disjoint, i.e. to potential umbrae. The number of umbrae are also counted by considering simultaneously the first two (dark and light blue in the figure), three (dark blue, light blue and green) and four (dark blue, light blue, green and orange) lowest intensity levels. The number of umbra(e) inside a spot is finally selected as the

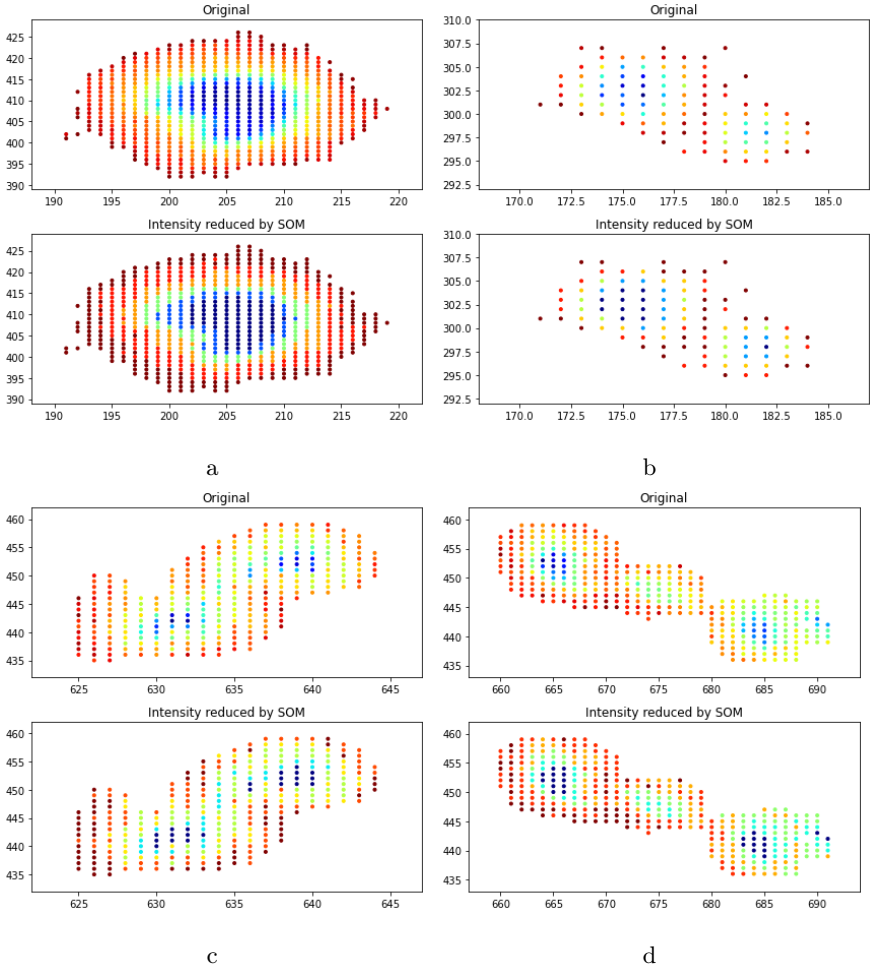
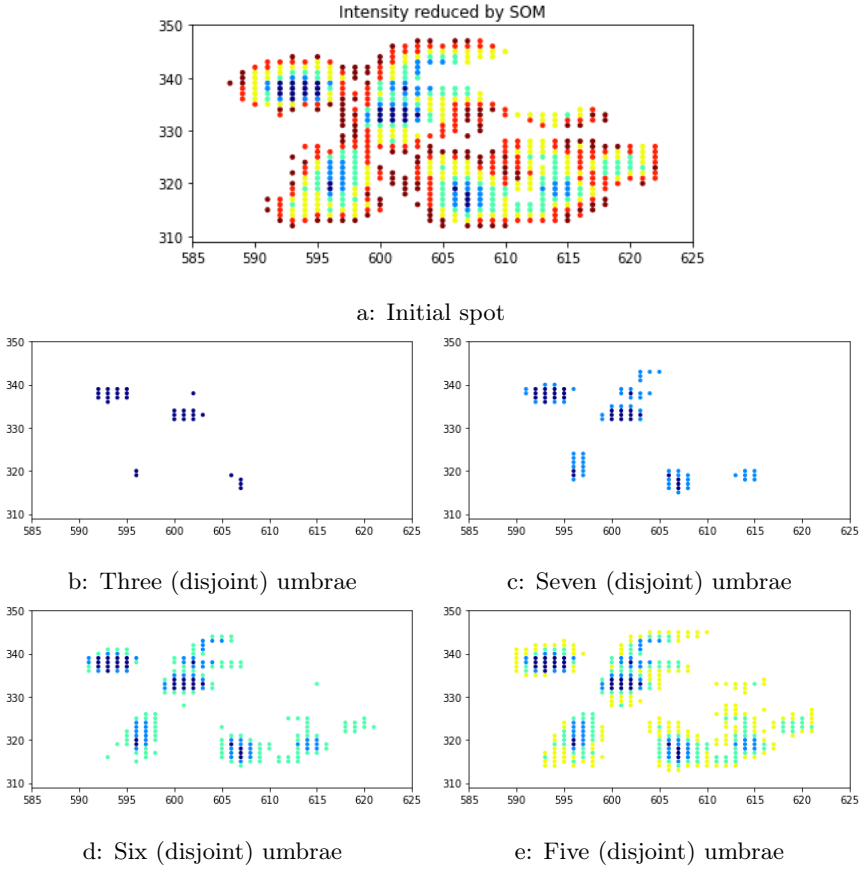


Figure 6.3: Self Organizing Map (SOM) applied to four different sunspots. The upper plots represent the original sunspots whereas the lower plots correspond to the resulting images after being reduced by the SOM into six different intensity levels. With the algorithm described in Subsection 6.5.1, one umbra is detected in the image (a), two, three and four umbrae are then counted in the subfigures (b), (c) and (d) respectively.

maximum of the number of umbrae obtained for the first, first two, first three and first four intensity levels. Note that an umbra should at least contain three pixels to be recorded in the method. This procedure is illustrated in Figure 6.4 for a complex spot (which is actually composed of seven overlapping spots).
This algorithm is a bit heuristic but appears to work well in practice. The number of umbrae obtained by the method corresponds to those that are manually counted



a: Initial spot

b: Three (disjoint) umbrae

c: Seven (disjoint) umbrae

d: Six (disjoint) umbrae

e: Five (disjoint) umbrae

Figure 6.4: (a) An initial 'spot' which is actually composed of several overlapping spots. (b) The lowest intensity level of the spot is represented in dark blue and three umbrae are counted by the method (an umbra should be at least composed of three pixels to be recorded). (c) The two lowest intensity levels of the spot are then represented in dark and light blue and seven umbrae are identified. (d) The three lowest intensity levels are showed in dark blue, light blue and green for a total of six umbrae. (e) Finally, five umbrae are registered with four intensity levels, in blue (dark and light), green and orange. With the algorithm previously described, the method identifies thus seven umbrae in the initial spot.

in most of the cases. With this method, the number of spots may be associated to the number of umbrae in the images.

6.6 Number of sunspot groups

Once we identified the number of spots in the images (either by taking into account the internal umbrae or not), we can develop an automated procedure for grouping them. In Curto et al. (2008), the authors used the same procedure — namely a region-growing algorithm based on a neighbourhood scale — to assign the spots to their groups as those used to attach pixels to their spots. Similarly to Dasgupta et al. (2011), we exploit here different clustering methods to group the spots. We concentrate on three well-know algorithms: the k-means (Lloyd, 1957; MacQueen, 1967), the expectation-maximization clustering using Gaussian mixture models (GMM) (Dempster et al., 1977) and the density-based spatial clustering of application with noise (DBSCAN) (Ester et al., 1996). Those are described in Appendix 6.9.1. The two first methods are general-purpose and may be applied to a large variety of data. k-means is particularly efficient to group data into a small numbers of clusters with regular or circular size whereas the GMMs based clustering is more suited to deal with elongated or irregularly shaped clusters, since it naturally generates groups of Gaussian forms. These procedures strongly depend on the number of clusters however, which cannot be automatically identified by those methods. Hence, we also use the DBSCAN to group the spots, as proposed in the works of Nguyen et al. (2006) and Adipranata et al. (2013). This procedure is able to automatically find the number of clusters from a neighbourhood size, another parameter which is much easier to adjust. The method is also expected to perform particularly well in case of uneven cluster sizes.
Note that the number of clusters corresponds here to the number of groups. We still carry on the discussion of the clustering algorithms however, since we are also interested to actually cluster the spots into groups (and not only to count the number of groups). This could be useful to visually control the results of the clustering methods or to analyse later some images in details.

Before describing our clustering procedure, we first explain how to select the optimal number of clusters (i.e. the number of groups) for k-means and GMM clustering. Those can be automatically chosen using two different criteria. The first one is the within-cluster sum of squares criterion, also called inertia, which writes as:

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2),$$

where x represents a spot among the N spots of the image and μ_j denotes the centre of a cluster C among G (fixed) distinct clusters. The inertia is thus a measure of the internal coherence of the clusters. The k-means algorithm used this criterion to find optimal values for the centres μ_j. Yet it can also be used to choose the optimal number of clusters (G_{opt}) for different clustering algorithms. Since the

inertia tends to zero when the number of clusters becomes close to the number of spots, the optimal value of G may be selected as the "knee"[4] of the inertia curve. The second criterion is the mean Silhouette coefficient (Rousseeuw, 1987), another widely-used measure of the clustering quality. The Silhouette coefficient is computed as the difference between the mean nearest-cluster distance and the mean intra-cluster distance, for each data sample. The mean Silhouette coefficient is then simply obtained by taking the mean of the Silhouette coefficient over all samples. It takes values between -1 and 1, where negative values indicate mis-classifications and positive values correspond to correct classifications. This criterion can be evaluated for different number of clusters: it is maximized at the optimal value.

The clustering procedure then works as follows. We first compute the centre of mass of each spot along three dimensions: intensity, x and y coordinates. The intensity of these centres of mass is defined as the mean of the pixel intensities of the spots. The coordinates of the centres of mass are then computed as the sum of the pixel coordinates of the spots weighted by their intensity (the sum are normalized). This step allows us to neglect the spatial extension of the spots. It also simplifies the computations since a spot will only be represented by a vector of three numbers in the following.
The clustering algorithm distinguishes then four different cases. (1) When there are no spots detected, zero groups are of course registered. (2) When there is a single spot, a single group is also recorded. (3) If two spots are detected on the Sun, we compute their (euclidean) distance. If it is superior to an arbitrary value fixed at 100, we consider that there are two groups on the Sun while if the distance is inferior to 100, a single group is counted. (4) When two or more spots are detected, the procedure becomes more complicated.
The inertia and the mean Silhouette coefficient [5] are first computed for different numbers of clusters starting at two till the number of spots. The optimal number of clusters with respect to both criteria is selected as previously explained for the k-means and GMM methods. The number of clusters in then selected in both methods as the minimum of the number of clusters obtained with the inertia and Silhouette. The neighbourhood size parameter of the DBSCAN is selected as well, by computing the mean distance between the spots to their first ten (or less than 10 if there are less than 10 spots on the Sun) nearest neighbours. It is then selected as the knee of the distance curve. The three different clustering methods with their optimal parameter are finally applied to the centres-of-mass of the spots, for the actual grouping. The number of groups is then simply obtained by counting the number of clusters obtained with each method.

[4] The knee of the inertia is located using the function `KneeLocator()` from the python package `kneed` (Satopaa et al., 2011).

[5] The mean Silhouette coefficient is obtained using the function `silhouette_score()` from the python package `scikit-learn` (Pedregosa et al., 2011).

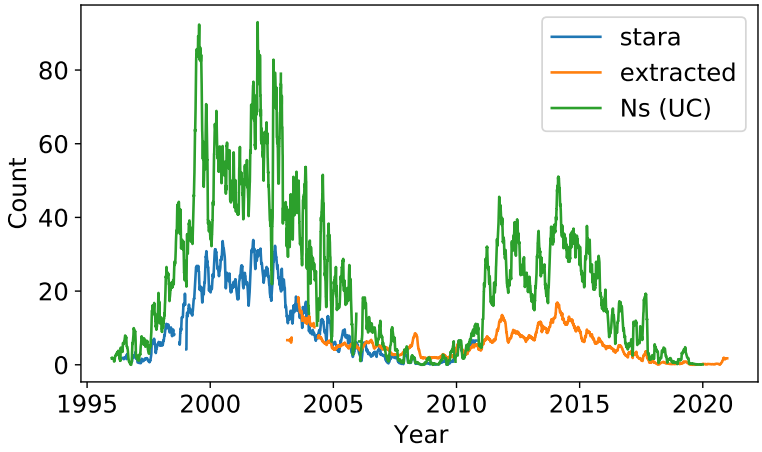
6.7 Results

In the following, we compare the number of spots and groups obtained with our extraction method to the observations. Those observations correspond here to the counts recorded in the station Uccle (UC), the same station which also registered the images that we analysed and which is responsible for producing the ISN. Note that the counts observed in UC are close to those of the median of the network.
We also inspect the STARA sunspot catalogue (Watson and Fletcher, 2010), regrouping observations from May 1996 to October 2010 (solar cycle 23). This catalogue contains the number of spots which are extracted using the automated STARA algorithm described in Watson et al. (2009) from images obtained in space. Those are acquired by the Michelson Doppler Imager (MDI) instrument on the Solar and Heliospheric Observatory (SoHO), a spatial observatory orbiting around the Sun. Figure 6.5a shows the number of spots obtained by the three different procedures. The number of groups are represented in Figure 6.5b for our extraction method and the observations only, since the STARA catalogue does not contain the number of groups. As the quality of the images are variable (e.g. some of them are particularly affected by clouds), we remove the outliers in our extracted numbers in the figures. These outliers are defined as values that do not fall into two (resp. one) standard deviations around the mean of the number of spots (resp. groups), where the mean and the standard deviation are computed in moving windows of length equal to 60 (around 2 months).
As can be seen in the figures, the number of spots detected by the automated procedures (our method and STARA) have a lower level than those obtained by visual counting. Similarly, the number of groups is two times smaller (at least during the years 2011-2016) with the extraction methods than with the observations. This is expected since the numbers that are extracted from a single image of the Sun are often inferior to those that are counted by directly observing the Sun over several minutes, a method which "integrates" many images of the Sun. The automated procedures have thus a lower resolution.

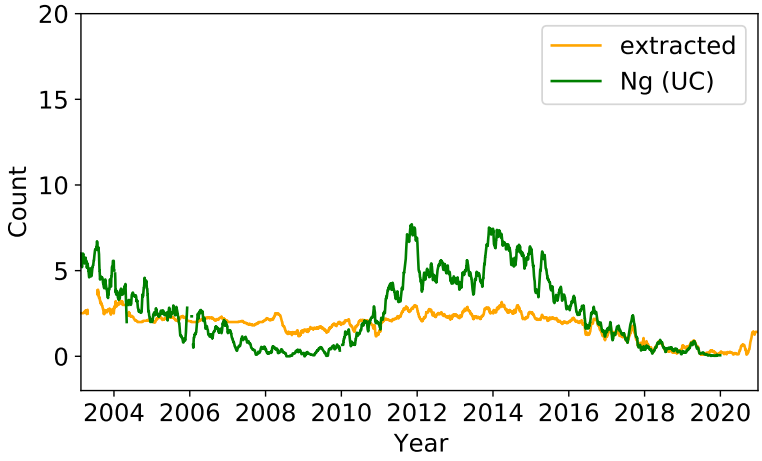
The STARA and the extraction algorithm that we developed differ however for three main reasons. First, the definition of a spot in the STARA algorithm excludes the pores, i.e. the smallest spots without penumbra. We record on the contrary all spots. Second, the resolution of the MDI instrument which recorded the images in space (used by STARA) is different from the resolution of the USET telescope, which provides the images that we analyse here. We also work with images of 750×750 pixels resolution to save some computing time, whereas the original images have a smaller resolution. The resolution of our images differs thus from the one of the STARA images. Third, the images obtained from space (SoHO) are not affected by the atmospheric conditions, contrarily to those gathered at the Earth's surface. The detection threshold is thus higher in ground-based images, to

avoid detecting too many background pixels. The CCD camera has also a longer lifetime on Earth since it is not stroked by a continuous stream of particles, which are mainly stopped by the atmosphere. Yet the images quality may be degraded by dust or water on the CCD sensors. Our method detects thus approximately the same numbers of spots as STARA, even if it excludes the pores.
Note that it could be interesting to apply our algorithm to the images obtained by the MDI instrument: to see if the differences in the number of spots recorded by STARA and our method are more related to dissimilarities in the algorithms or in the images.

To better compare the observations with the extracted numbers, we rescale the latter on the former using a parametric scaling of the form ax^b. The parameters a and b were adjusted on the observations separately for the number of spots and groups. The results are shown in Figure 6.6. The values of the rescaling parameters are equal to 2.24 (a) and 1.11 (b) for the spots and 0.52 (a) and 2.20 (b) for the groups. A quadratic scaling appears thus to perform better than a linear scaling for the number of groups. The Euclidean distance between two vectors $(x_1, x_2, ..., x_n)$ and $(y_1, y_2, ..., y_n)$ is defined as $d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2}$. Using this formula, we also compute the distance between the extracted numbers and the observations on the whole period studied, which covers the years 2003-2020. It is equal to 136 for N_g and 1090 for N_s. After rescaling, this distance drops to 99 for N_g and 529 for N_s. Since N_s and N_g do not vary on the same scale, we also compute the distance between the extracted and observed numbers of groups, when those groups are rescaled on the number of spots. In this case, we obtain a value of 834 instead of 99 for N_g, after rescaling. This illustrates that the extraction algorithm works better for identifying spots, which have an Euclidean distance equal to 529 after rescaling, than groups for which the distance is equal to 834.

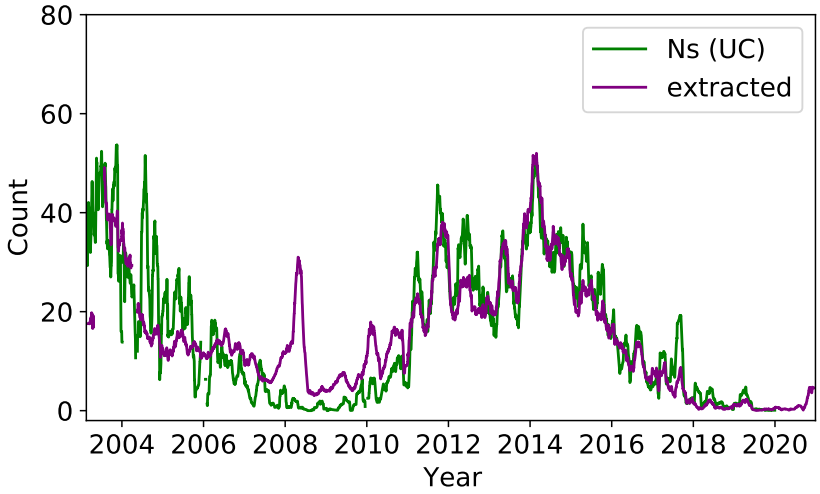


a: Number of spots

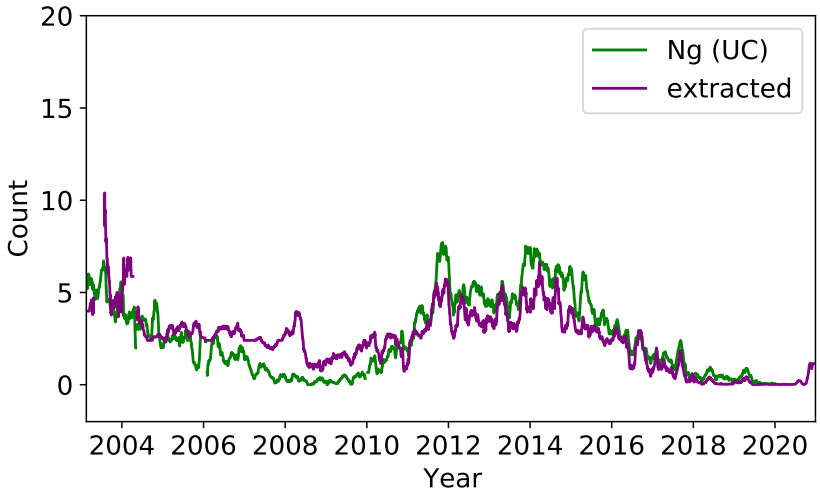


b: Number of groups

Figure 6.5: (a) The number of spots obtained by the STARA algorithm and by our developed algorithm are represented respectively in blue and in orange. The observations recorded in the station UC are also shown in green. Note that we show a longer period than those of our extracted number in the figure, to display the STARA numbers on a complete solar cycle. (b) The number of groups extracted with our algorithm is presented in orange and the observations of the station UC are in green. All numbers have been smoothed by a moving average filter on 54 days for readability purpose.



a: Number of spots

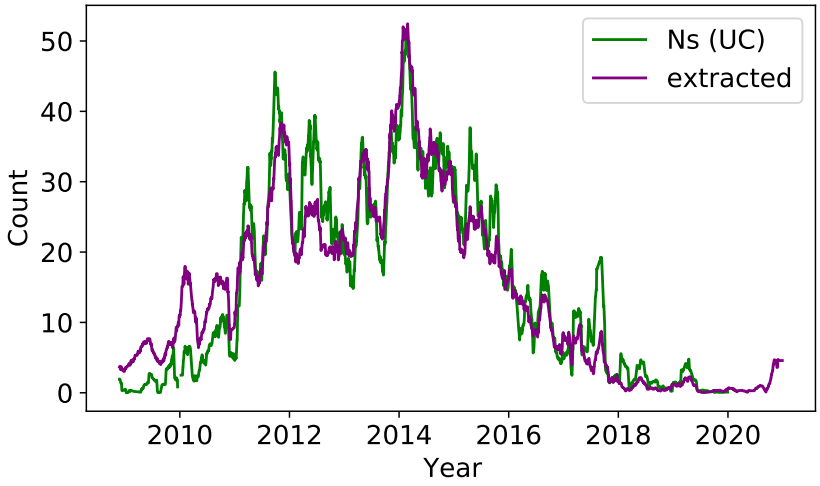


b: Number of groups

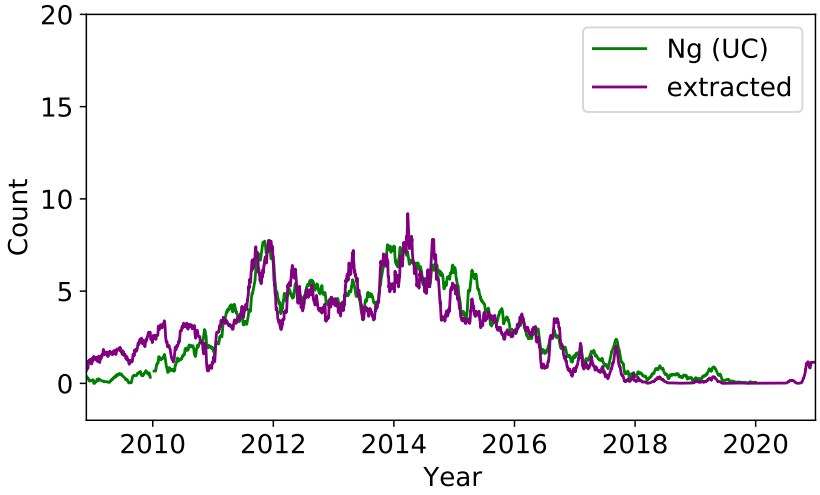
Figure 6.6: (a) The number of spots that is obtained by our extraction algorithm is represented in purple. It is rescaled on the observations of the station UC, which are shown in green. (b) Same figure for the number of groups.

As can be seen in Figure 6.6, the extraction method does not work properly around the solar minima of 2008. In this period, the algorithm detects too many spots (and thus groups) with respect to the observations. This over-count may be related to the fact that the algorithm extracts and detects the darkest parts of the images at minima, even if they do not contain actual spots. We have already tried to correct this effect in Section 6.4 by using the mean structural similarity index as a first indicator of the presence of spots but it might be insufficient. The over-count is however not visible around the next solar minima in 2019. We extend the analysis of the images until the end of 2020 and this effect still does not appear. Hence, we suspect that the over-count might be more related to the images than to the extracting procedure, which remains to be investigated.
In general, the extraction algorithm also performs better on images recorded after 2011. The Euclidean distance between the extracted number of spots and the observations drops to 250 in the period covering years 2011-2020, after rescaling (instead of 529 for the years 2003-2020). The extracted numbers are thus also presented in Figure 6.7 solely for the solar cycle 24, which extends from December 2008 till December 2019. They are better adjusted visually to the observations on this cycle than on the whole period studied.
The extracted numbers of spots that are presented in this section do not distinguish the umbra from the penumbra. When the internal umbrae are counted as separate spots, we obtain higher numbers of spots. Those can be rescaled on the observations with scaling parameters are equal to 1.81 (a) and 1.07 (b). The Euclidean distance also decreases from 1090 to 941 when counting the umbrae. After rescaling, this distance drops to 540. Taking into account the internal umbrae lead thus to numbers that are effectively closer to the observations.

The results presented for the number of groups are obtained using the Gaussian mixture models (GMM) clustering. They are close to those obtained by the k-means algorithm. The DBSCAN detects on average 20% less groups than the two other methods on the whole period studied. In general however, small variations are visible between the three different clustering methods. Hence, we only show results associated to GMM clustering. As stated before, the method for automatically counting the groups works less well than those to count the individual spots by around 40% (we obtain 529/834=0.64 with the Euclidean distances previously computed). In general, counting the number of groups is more complicated task than counting the spots, even for manual observations. More research is thus needed to obtain a better match between the extracted and the observed numbers of groups. Introducing more physical criteria such as a typical neighbouring scale for sunspots to be part of the same group (Curto et al., 2008) could be beneficial for the method. The relation between the number of spots and groups, which may be studied in past data, may also provide a physical range of values for the number of groups.
In further research, the method could also be substantially modified to incorporate



a: Number of spots



b: Number of groups

Figure 6.7: (a) The extracted number of spots is represented in purple for the solar cycle 24. It is rescaled on the observations of the station UC, which are shown in green. (b) Same figure for the number of groups.

magnetic information about the spots. A more complex procedure involving magnetograms in addition to white light images could then be designed such as in the works of Zharkov et al. (2005); Colak and Qahwaji (2008). This upgraded procedure would most probably better identify pairs of corresponding spots, which may refine the detection of overlapping spots or spots with irregular shapes. It could also improve the grouping procedure similarly to the approach taken by Colak and Qahwaji (2008), where the magnetogram images are combined with a neural network to cluster the spots into groups.

6.8 Conclusion

Our method shows that an automated procedure can be designed to count the spots and groups from ground-based white light images, such as those recorded in the station Uccle. Contrarily to images obtained in space, the ground-based images share more properties with the observations since they are both affected by the atmospheric conditions. The extracted numbers from ground-based images are thus expected to better correspond to the observations, which is valuable to preserve the continuity of the series over time. Moreover, such images are more easily obtained than those gathered in space. The instruments can for instance be installed at lower cost and have a longer lifetime on the ground since they are protected from radiations by the atmosphere. The algorithm and instruments which record the images could thus be easily integrated in the actual network, with little additional cost.
An automated counting procedure has also the advantages over manual observations of being robust against most of the observer-related variations, since the same algorithm can be implemented in all stations. Those variations have diverse origins such as differences in the counting methodology, change of observers or eyesight declines. The method also requires less interventions from the observers. The proposed algorithm has thus several advantages with respect to other existing automated methods and with respect to the observations.

The automated method developed in this section has however a lower resolution, which depends on the values of certain parameters and rules in the algorithm as well as on the resolution of the USET instrument. To solve this problem, we can rescale the extracted numbers on the observations, as we did previously. We could also apply the algorithm to several images of the Sun, obtained successively or spaced by an arbitrary interval, and record the maximum number of spots extracted from those multiple images. This approach would be more similar to the direct observations of the Sun over a certain amount of time. We do not implement it however, since those images are not yet available but they could be easily registered in the future.

As stated in the introduction, the method proposed in this section is not ready for being implemented and used. Prior to this, consequent work is required to improve the method, by taking for instance more physical information into account. This preliminary version of the algorithm demonstrates however that it is possible to extract the sunspot numbers from ground-based images. A fully-effective method remains to be developed, for extracting in particular the number of groups.
When this automated method would be operational, it may then be implemented in different stations, at least the professional observatories, and used jointly with the observations. The visual and automated counting could serve both as controls for each other. After a sufficient amount of time for thorough study and tests of the method, the extracted numbers could also participate in the construction of the International Sunspot Number as stations will hopefully less observer-related errors. This integration should however be sufficiently "smooth" to ensure the continuity of the series over time.
It is more difficult to say if an automated method such as the one that we developed here could one day replace totally the manual observations. This depends on whether the scaling between both methods changes from one solar cycle to another or if it stays more or less constant. Further analyses of the method over the next solar cycles could bring more definite answers to this question.

6.9 Appendix

6.9.1 Clustering methods

In this appendix, we give a rapid overview of different clustering methods. A clustering procedure is an unsupervised method that classifies data into specific groups. Data from the same group usually share some properties while data from different groups are more dissimilar. In the following, we present five different algorithms: the k-means (Lloyd, 1957; MacQueen, 1967), mean-shift (Comaniciu and Meer, 2002), DBSCAN (Ester et al., 1996), expectation-maximization clustering using Gaussian mixture models (Dempster et al., 1977) and the agglomerative hierarchical clustering (Anderberg, 1973). We refer to the documentation of the package `scikit-learn` (Pedregosa et al., 2011) in Python[6] for a more detailed review of the main clustering procedures, with graphical examples.

[6]The review of the clustering methods can be found at the following link: https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering.

K-means

One of the most commonly-used clustering methods is the k-means. To cluster data, this general-purpose method needs to know in advance the number of clusters, G. It then groups the data to minimize the inertia, i.e. distance between each point to the centre of its cluster:

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2), \tag{6.9.1}$$

where x represents a data point among a set of n points and μ_j denotes the centre of a cluster C from G different clusters.
The k-means is based on the Lloyd's algorithm (Lloyd, 1982), which works as follows. The algorithm starts by defining G centres, randomly selected in the data space or directly from the data themselves. These centres have thus the same dimension as the other data points. Then, the distances (Euclidean distance or maximum distance, etc.) from each point to the centres are computed. The points are then attributed to the group whose centre is the closest to them and the group centres are updated as the mean of all points in the group. These steps are then repeated several times until the group centres converge. Note that the algorithm may also be run several times with different centres initializations and the best results may be kept.

This method is simple and relatively fast. It relies however on the choice of the number of centres, G, which may be complicated to estimate (especially in high dimension). As the method varies from one initialization to another, the results may also be difficult to reproduce. Moreover, since the cluster centres are computed using the mean, k-means does not perform well when the clusters have elongated or irregular shapes (in 2D or higher dimensions). Similarly, the method is not able to distinguish concentric clusters of different radius located on the same centre as well. Hence, it is mainly used in applications where there are a few number of regular and even-sized clusters.

Mean-shift

Contrarily to the k-means, the mean-shift clustering uses a sliding-window to locate dense areas of points. It associates centre candidates to the mean of dense areas and then filters them out in a final post-processing step. The algorithm works thus as follows. First, a point c is randomly selected from the data. A circle of radius r is then drawn around c, like a kernel. At each step, the window is moved to regions of higher density by shifting the centre of the circle to the mean of the points within the window. The circle is moved until there is no direction that can increase the

number of points inside the circle. The algorithm is run several times for different starting points c, until all points belong to a group. Overlapping groups are finally merged into the group containing the highest number of points.

This method is preferably used when there are a large number of uneven-sized clusters. The centres of the groups also correspond to regions of high density, which may be desirable in some applications. Contrarily to the k-means, the mean-shift automatically finds the number of clusters in the data. It depends however on the choice of the radius r of the circular window, which may be complicated to determine.

DBSCAN

The density-based spatial clustering of application with noise (DBSCAN) is another density-based method, similar to the mean-shift. The DBSCAN considers that a cluster is a region of high density, which is separated from other clusters by regions of low density. The algorithm starts by choosing randomly an unvisited point. Then, the neighbourhood of the point (i.e. all points within a distance ϵ to the point) is analysed. If the neighbourhood does not contain enough points, the point is classified as noise. Otherwise, the point becomes the first point of a new cluster. In both cases, the point is marked as visited. All neighbours of the point are then added in the cluster as well as all their neighbours and the neighbours of their neighbours, etc. within a distance ϵ and marked as visited. The algorithm is repeated with another unvisited points until all points are visited (either classified as noise or part of a cluster).

The DBSCAN is thus able to automatically find the number of clusters. It can also differentiate noise from actual data and find clusters of arbitrary shapes. Hence, it is particularly useful to cluster data containing outliers, that should not be part of any group. It is also efficient to deal with uneven cluster sizes. Yet, the method depends on two parameters: the neighbourhood ϵ and the minimum number of points to define a cluster. These parameters may be difficult to estimate, especially when the clusters have varying densities or in high dimension.

EM clustering using GMM

As seen previously, the k-means does not perform well when the clusters are not circular (for 2D data). To overcome this problem, the expectation-maximization (EM) clustering using Gaussian mixture models (GMM), called GMM method in the following, was designed. This method uses the EM algorithm to fit Gaussian mixture models to the data. It generates thus Gaussian shaped instead of circular

clusters, which can take any elliptical shape desired (in 2D). These clusters are thus described by two parameters: the mean and the variance, which are progressively adjusted to the data by the EM algorithm.
For a pre-specified number of clusters G, the first step of the algorithm is to initialize the clusters using random parameters. For each point, we compute then the probability that it belongs to the different clusters (points closer to the mean of a cluster are more likely to belong to that cluster). The parameters of the clusters are then updated to maximize the likelihood of the data given those cluster parameters. Practically, the mean and the variance of the clusters are adjusted using a weighted sum of the data point locations, where the weights of the points are the probabilities that the points belong to each cluster. These steps are then run until convergence.

The two major advantages of the GMM method with respect to k-means are that the clusters can have variable shapes and that the data are clustered using probabilities. This method is thus often used for applications where the clusters are irregular or where a single data point may belong to several clusters, with different probabilities (mixed-membership).

Agglomerative Clustering

The last clustering procedure that we present here is a hierarchical method. In general, there are two categories of hierarchical clustering: the bottom-up and the top-down models. The agglomerative clustering is a bottom-up method, which successively merges pairs of (closest) clusters until there is only one cluster left.
This method works as follows. Each point is initially treated as a cluster. The distance between two different clusters is then evaluated using a linkage criterion, which determines the distance between two clusters from the pair-wise distances between points in the first cluster to those of the second. The linkage criterion is usually chosen as the average linkage, though other criterion such as the maximum linkage or single linkage can also be used. It corresponds to the average pair-wise distance between all points of a pair of clusters. Then, at each iteration, the pairs of clusters closest to each other are merged. The algorithm is iterated until there is only one remaining cluster. With this method, it is thus also possible to group the data into a desired number of clusters, simply by stopping the algorithm when this number is reached.

Unlike several other methods, the agglomerative clustering is robust against the choice of the distance metric (though not against the linkage criterion). All distance metrics (e.g. Euclidean distance, Manhattan distance, etc.) lead thus to similar results. Moreover, the algorithm can either cluster the data into a desired number of clusters or automatically find an appropriate number of clusters based on a

threshold distance. With this last method, the algorithm stops merging clusters if their distance is superior to that threshold. This method is thus particularly useful to cluster data which have an underlying hierarchic structure that we want to unravel. It is however pretty slow compared to the other methods.

CHAPTER 7

Discussion and Extensions

There are several ways to summarize this work. The proposed attempt is to view the whole thesis as the development of different methods that target a common objective: improving the quality of panels of time-series over time, and in particular the sunspot numbers. Although mainly related to this specific application, the approach taken is indeed general. The first step of a procedure that aims at improving the quality of a particular dataset is most of the time the development of a model for those data, as we did in Chapter 2. This allows a better comprehension and analysis of its different sources of errors. Once they have been identified, different methods can be set up to prevent the occurrence of similar inconsistencies in the future. For instance, the procedure for collecting the data can often be modified to reduce some kinds of errors, as those proposed in Chapter 6. A monitoring scheme, a powerful tool to automatically detect current as well as past deviations, is also often constructed. This allows a better identification of the deviations and helps preserving the quality of the data over time as demonstrated in Chapters 3 and 4. In a last step, the monitoring procedure is also frequently used to reconstruct past series, which remains to be done with the sunspot numbers.
Among those different steps, some of them are thus particular to each dataset, such as here the development of the model in Chapter 2 or the extraction algorithm of Chapter 6, whereas others are general and transposable to many applications, as demonstrated in Chapter 5.

Having fulfilled our objectives with the sunspot numbers, we provide now a general conclusion to this thesis. It mainly focusses on the limitations of the proposed methods and the research perspectives since the main achievements have already been described in the introductory chapter. They have also been developed in de-

tail throughout the thesis. We also believe that what has been done is (always) little compared to the vast extent of what could be or remains to be done. Hence, we first review the limitations of this work, chapter by chapter, and propose potential solutions or limited extensions of the methods. Then, in a second part, we propose broader research perspectives, related to (1) sunspot numbers and (2) the monitoring methods that have been developed.

7.1 General conclusions

We begin our journey with the development of a comprehensive uncertainty model for the sunspot numbers in Chapter 2. This model is written in a multiplicative framework and includes three types of errors, according to the latest research of data experts. We provide a complete analysis (with fit of the different distributions) for all terms of the model, including robust estimators for the actual solar signal.

A first criticism that can be addressed to this work is the absence of numerical measures of goodness-of-fit for the different fits that were presented. In practice, we often choose the best fits by a visual criterion. Although different tests were used such as the Kolmogorov-Smirnov test (Massey, 2012), the p-values were either extremely low or the methods allow no comparison with complex multi-modal distributions. Further researches are thus needed to provide numerical evaluations of the proposed parametric distributions.

As a second criticism, it could be noted that we do not really take advantage of those parametric models for the distributions. At first, we thought that they could be used for the monitoring though it soon appeared that the distribution of the long-term error varies significantly from one station to another. Hence, we choose a non-parametric approach for the long-term monitoring instead. A parametric monitoring based on the short-term error or the error at minima, as proposed in Chapter 2, remains to be developed.

Those parametric distributions could however be used to generate data similar to the observations. A first proposition of such a method was presented in Mathieu et al. (2019b). In this conference paper, the distribution of the solar signal is used to simulate new data with similar properties as the observations. This approach was however not further developed by lack of time, but it could be improved to follow more closely the model. The distribution of the errors could for instance be integrated into the procedure, to generate new data in a more diverse manner than those obtained by the block bootstrap (a method which only samples blocks of consecutive observations).

A third research perspective is related to the brief analysis of the conditional correlation that is conducted in Section 2.5.5. We could go one step further and study the conditional distribution of the number of spots (N_s) and groups (N_g) for different values of the composite (N_c), with aim to reproduce the distribution

of N_c from those of its building blocks. The analysis has not be done in this work however since the task is arduous: (a) the correlation between N_s and N_g changes with time and (b) the distribution of N_s and N_g are complex mixtures.

In Chapter 3, we construct a robust non-parametric control scheme to monitor the sunspot numbers over time. This method is composed of a CUSUM chart designed by block bootstrap to detect the deviations and support vector machines to estimate the characteristics of those shifts (it is thus called the CUSVM method). In further research, other charts may be compared with the CUSUM. The exponentially weighted moving average (EWMA) (Roberts, 1959) is for instance another popular choice of control charts that could be studied. Simulations such as those presented in (Qiu, 2013, Table 5.3) demonstrate however that both charts have similar performances. More elaborated charts such as the adaptive dynamic EWMA chart in the framework of generalized likelihood ratio testing, which is proposed in Qiu et al. (2017), could also be tested. Although they may outperform the CUSUM, those charts are more complex and less familiar to non-specialists of quality control, which decreases a bit their interest. Moreover, the CUSUM chart has an optimal relation between its allowance constant k and the target shift size δ, of the form $k = \delta/2$. This relation allows thus an easier design of the CUSUM with respect to other charts such as the EWMA.
Although valid for various distributions as studied in Deketelaere (2020), further research is however needed to see if this relation $k = \delta/2$ is also valid for other shift shapes than the jumps. Note that this remark also applies to the adaptive CUSUM chart proposed in Chapter 4. The procedure developed in this chapter also contains an automatic pre-selection of an in-control pool based on different clustering algorithms. This method could be modified in future works to take into account the parametric distributions of the data. This has not be implemented here however, since the approach may provide good results for the sunspot numbers but is more complicated to generalize to other data.

Our journey continues in Chapter 4 with the construction of feed-forward and recurrent neural networks (NN), as alternatives to predict the sizes and shapes of the deviations. The NN-based control schemes developed in this chapter offer several advantages with respect to the CUSVM method, such as a simpler processing of the data or the adjustment of the allowance parameter as a function of the predicted shift size, $k = \hat{\delta}(i,t)/2$. After comparison with the CUSVM monitoring, they also appear to detect faster large or oscillating shifts, at the expense of a greater complexity.
The architectures of those networks have however been chosen after only few tests. We design for instance networks with two times more neurons in the first layer than in the second but do not try other combinations. More research is thus needed to explore a larger number of architectures and find potentially sparser networks with

achieve at the same time similar or better degrees of accuracy. Moreover, the artificial deviations implemented in the training sets of the SVM and neural networks could also be more adjusted to the actual deviations of the data, as proposed at the end of the chapter.
Additionally, the different networks have been compared to the previously developed CUSVM method as well as with each other on simulated shifts. Those artificial deviations, although composed of various shift sizes and shapes, were designed on top of the (IC) sunspot data. Hence, they provide an interesting comparison of the methods for data which experience a high positive autocorrelation, similar to those of the sunspot numbers. More results are however needed to see if similar patterns are observed on data which have negative and/or low autocorrelations and which follow various distributions.

In Chapter 5, we apply the CUSVM method to the photovoltaic energy production in Belgium. This allows us to detect different kinds of deviations in those data and initiates our collaboration with Elia, the manager of the high-voltage electricity in Belgium, to further analyse the root-causes of those shifts. This second application also gives us the opportunity to present the package, written in the programming language Python, which implements the CUSVM method. This package is quite general and allows the user to select for instance different clustering algorithms and various block bootstrap procedures to perform the monitoring. It also contains three different ways to treat the missing values: those can simply be removed from the data or the values of the chart statistics can also be reset to zero after large gaps and propagated through the smallest ones. A third method also fills the data by the mean of each series. To complete the package, more refined techniques could thus be implemented to fill-up the gaps of the series. This could be particularly useful for monitoring panels of data which contain only a small number of observations. Those gaps could for instance by replaced by the cross-sectional median of the panel or interpolated using a more complex procedure adapted for time-series, such as those proposed in Dudok de Wit (2011). The method and package can also be applied to various applications in the future. The next application that we intend to monitor is for instance the sales of pharmaceutical products across different stores in Belgium.

In the last part of the thesis, we construct a preliminary version of an automated method to extract and count the spots and groups from ground-based images of the Sun. The method shows good results for counting the number of spots in the period covering years 2011-2020. It works however less well on the first part of the images, recorded in the years 2003-2010, and for counting the groups. More research is thus needed to obtain extracted numbers that are closer to the observations but the approach demonstrates enough potential to be further investigate, which fulfils our initial objective. In the future, several methods could then be used to improve the algorithm, such as incorporating more physical information or

magnetogram images in the processing. Those could help us to distinguish pairs of spots, especially if they overlap, and groups.
While this approach integrates more information about the solar physics underlying the spots, it is also possible to explore deep learning methods. A convolutional neural network (CNN) could for instance be developed in the future to automatically extract the number of spots and groups from ground-based images. To correctly work however, the CNN should be trained on a sufficiently large amount on data, which are not yet available. One solution to this problem would be to train the network on pre-processed images obtained with our algorithm (such as those of Figure 6.2d). These images are indeed more homogeneous and contain the sunspots which are enhanced with respect to their surrounding. Hence, we expect that fewer images would be required in the training stage of the method to obtain a desired accuracy. The extraction algorithm developed in Chapter 6 can also serve to label the number of spots in the images. It could be interesting to see how the predictions differ when using those labels with respect to the observations (N_s). Although the labels are sometimes erroneous, supervised methods have been developed to be robust against mislabelled data. Bekker and Goldberger (2016) and Sukhbaatar et al. (2015) add for instance an extra noise layer to match the outputs of the networks to the noisy labels whereas Liu et al. (2017) rely on a confidence policy to progressively depend less on the labels and even switch their values to those of the max-activated neurons, when the networks are confident enough.
Inspiration for extracting and counting the spots may also come from medical image analysis. Veta et al. (2015) provides for instance an overview of different techniques, such as CNNs or classifiers based on support vector machines and random forests, to count mitotic figures in images. Those are one of the most important markers to detect breast cancers.

7.2 Research perspectives

As stated in the introduction, this thesis follows two tracks: one related to a specific dataset: the sunspot numbers and the other linked to more general methods that can be applied to many data. Research perspectives can thus be proposed in both directions.

7.2.1 Future prospects for the sunspot numbers

Having developed an uncertainty model for the sunspot numbers, we can now employ it to upgrade the ISN processing that was presented in Section 1.2.2. With this procedure, the ISN is computed as the daily weighted average of the composites N_c, which have been previously rescaled by a scaling factor — the k-coefficient defined in (1.2.2) — evaluated on a single pilot station. This single pilot could thus be first

replaced by a more robust quantity involving several stations such as the median of the network or the transformed median defined in (2.5.1). Although the median is robust to few deviating stations if the network is sufficiently large, the pilot could also be substituted by the mean or the median over a pool of IC stations. Such a pool may be selected using e.g. the procedure described in Section 3.4.1. Similarly, the ISN, which is obtained as the daily mean of the composites N_c, could be computed either as the median of the N_cs from the whole network or the mean/median over an IC pool only, for robustness purpose. Finally, the scaling factors which are computed each month as the sigma-clipping mean of (1.2.2) could also be replaced by the long-term errors with levels, defined in (2.6.5).
Additionally, the model of (2.3.4) may also be used to provide errors for the observations at each time and in each individual station. The distribution of such errors can furnish confidence intervals (ci) for the observed numbers in each station as well. Examples of such errors and cis are already implemented in the package `uncertainty`, which is provided for the WDC-SILSO team at the Royal Observatory of Belgium. It could also be interesting to study in future works how the errors of the composites propagate to the ISN, especially if the latter is computed using a more complex procedure than a weighted average.

The quality of the series over time can also be improved using the monitoring methods that we developed in Chapters 3 and 4. A new version of the ISN may for instance be released after excluding stations experiencing high deviations from the computations. Those large deviations can be detected automatically, after eventually readjusting the control limits of the methods to only exclude the highest shifts in past series. Then, a monitoring scheme can also be set up to control the stability of the stations in quasi real-time. Different procedures could be used for this purpose: the CUSVM method proposed in Chapter 3 or another procedure based on neural networks constructed in Chapter 4 or a combination of both. Different kinds of alerts could also be designed to advice the observers. Those may depend on the type of errors but also on the station. Professional observatories may received for instance more frequent notifications than amateurs. A user-interface could also be created to allow more autonomous verifications from the users themselves. To that end, the graphical displays proposed alongside with the monitoring methods (as in e.g. Figure 3.9) may prove useful.
Such a monitoring has been applied in this work at two different scales, 27 days and one year. Data may however be supervised at other scales in the future. It might also be interesting to monitor the short-term error, ϵ_1 of (3.3.1), in incoming studies.

There are also broader research prospects related to the sunspot numbers that can be investigated. An interesting research topic is the predictions of future values of the series. Although it has not be treated in this work, this subject is in fact the

one which is the most studied in the sunspots literature. Previous works in Dudok de Wit et al. (2016) model the sunspot numbers by autoregressive processes but these series are in general poorly described by simple ARMA models. Hence, other techniques may be explored for obtaining better forecasts.
Petrovay (2020) provides for instance an overview of several methods that are currently used to predict the amplitude of the following solar maximum just after the start of a cycle. Three different types of models are presented: (1) precursors which predict the strength of the solar maximum using measures of some physical quantities at particular times, (2) model-based approaches which make predictions based on solar dynamo models and (3) methods based on time-series analysis (including but not limited to ARMA models). Among those methods, the precursors are used for the longest and outperform in general the predictions of time-series models. A popular predictor for the maximum of a cycle is for instance associated to the amplitude of the magnetic field near the pole in the minimum of the previous cycle. This precursor relies on the fact that the maximal magnitude of the poloidal field affects the following maximum of the toroidal field since the latter is produced by differential rotation on the former, as explained in Chapter 1. The model-based approaches, although more recently developed, are also gaining increasing interest. They mostly rely on simplifications (mean-field theory) of the set of coupled differential equations which describe the behaviours and time evolution of the toroidal and poloidal magnetic fields. In addition to these three types of methods, Pesnell (2020) also reviews others techniques such as neural networks for making predictions about the solar cycle 24.
Another promising approach for sunspot predictions is to use hybrid methods, which combine well-known statistical models for time-series with machine learning methods and in particular deep neural networks. The exponential smoothing and recurrent neural network (RNN) (Smyl, 2019) is a good example. This method uses an exponential smoothing to model non-stationary components of the series (such as the seasonality and linear trend) whereas the RNN learns the additional components (the non-linear trend and the stationary part). This method involves thus global parameters, which are learned over multiple series as well as local parameters, which are specific to each series, in a common framework. This method could be applied here for predicting the future values of the robust estimators of N_s, N_g and N_c, defined in Section 2.5. We refer to Lim and Zhoren (2021) for a general overview of the machine learning methods for time series forecasting, including those hybrids models.
Several methods also exist to forecast and de-noise at the same time the series. Those approaches integrate for instance de-noising layers based on multi-resolution wavelets into feed-forward or recurrent neural networks, which allows the learning of the parameters of the network and those of de-noising layers with a common procedure (Lotric and Dobnikar, 2005). Such methods could thus be particularly suited for the short-term forecasting of sunspot numbers in the individual stations, which are corrupted by much noise.

7.2.2 Future prospects for the monitoring methods

The methods that are developed in this work focus on monitoring the mean of a panel of time-series. In practice however, the mean and the variance of a process can both experience a shift. An upward shift occurring in the variance usually degrades the quality of the series whereas a downward shift tends to improve its properties. Hence, in most applications, only the upward shifts are interesting to detect. A decrease in the variance can however be associated in some applications with higher production costs and may also be monitored for this reason.
Moreover, a change in the variance often impacts the detection of shifts in the mean. As explained in Qiu (2013), the actual IC average run length (ARL_0) value of a chart designed to detect mean shifts is lower (resp. higher) than expected when the variance is shifted upward (resp. downward). The proposed methods could thus be modified to allow the monitoring of the variance of the series as well. Let us assume that X is a univariate i.i.d. process of IC mean equal to μ_0 and IC variance equal to σ_0^2, which experiences a shift from σ_0^2 to σ_1^2 in the variance only. As explained in (Qiu, 2013, Chapter 4), the following two-sided CUSUM chart can then be used to detect both upward and downward shifts in the variance:

$$\begin{aligned} C_n^+ &= max(0, C_{n-1}^+ + \left(\frac{X_n - \mu_0}{\sigma_0}\right)^2 - k) \\ C_n^- &= min(0, C_{n-1}^- + \left(\frac{X_n - \mu_0}{\sigma_0}\right)^2 - k), \end{aligned} \tag{7.2.1}$$

where $C_0^+ = C_0^- = 0$. This chart gives an alert if

$$C_n^+ > L^+ \quad \text{or} \quad C_n^- < L^-. \tag{7.2.2}$$

As for monitoring the mean, if the process is i.i.d. and normally distributed, k has an optimal value for detecting a shift from σ_0^2 to σ_1^2 in the variance:

$$k = \frac{2\log(\sigma_0/\sigma_1)}{(\sigma_0/\sigma_1)^2 - 1}. \tag{7.2.3}$$

Based on the above formula, a scheme for monitoring the variance of the series could therefore be included in the CUSVM method developed in Chapter 3. A joint monitoring can also be developed for detecting simultaneously shifts in the mean and the variance of a process. Since both chart statistics (defined in (7.2.1) for monitoring the variance and in (3.2.13) and in (3.2.15) for monitoring the mean) are usually correlated, the control limits of such a joint procedure should be calibrated together, using a method similar to those described in Algorithm 1.
Similarly, the neural networks constructed in Chapter 4 can also be trained to identify shifts in the variance of the series, after adapting their training sets. Simple cut-off values or adaptive CUSUM charts can then be designed to trigger alerts on

top of those networks when a sufficiently large shift is detected in the variance.

As stated before, a change in the variance can also affect the detection of the deviations in the mean (by modifying the actual ARL_0 value of the method). Therefore, it might be useful to stabilise the variance of the data before monitoring their mean with the methods proposed in Chapters 3 or 4. Moreover, near-Gaussian processes are often easier to monitor than data following e.g. an asymmetric distribution. Indeed, the allowance parameter k of the CUSUM chart is optimally related to the shift size δ though the formula $k = \delta/2$, when the data follow a distribution which is close to the normal. Similarly, the positive and negative control limits take the same value with opposite sign when the distribution of the data is symmetric, which allows an easier calibration of the chart.
In practice however, a process may be arbitrarily distributed. In these cases, we may first apply a transformation to stabilise the variance and Gaussianise the data before the monitoring. Different procedures have been proposed in the literature for this purpose. The Anscombe transform (applied in Section 2.5.1 for computing the transformed median of the network) may for instance be used on Poisson processes. If the process is non-negative and has a variance which is a non-decreasing function of the mean (as it is typically the case for Poisson or negative binomial variables), a data-driven Haar-Fisz transform (Fryzlewicz and Delouille, 2005) can also be applied for an automated stabilisation of the variance and Gaussianisation of the data.

Although developing a multivariate scheme was not interesting for the sunspot numbers, such a method could be construct in the future, for monitoring other panels of time-series. The quality of spacecraft engines is for instance continuously monitored though the multivariate control of several series including the temperature and the pressure of different components of those engines.
In general, two different kinds of multivariate control charts exist in the literature. Those of the first category are usually composed of univariate charts that are designed jointly to reach a common value of ARL_0 whereas the others are truly multivariate in the sense that the vector of data from the panel is monitored together. In both cases, the design is much simpler when the different processes are not correlated. If a common IC period is available in the different processes, it should then be possible to design a multivariate control scheme of any category using a block bootstrap procedure similar to those used in Chapter 3, for taking into account the actual distribution and the autocorrelations of the series. If all series contain an IC period but those periods are not overlapping, then univariate charts could still be adjusted separately for each series with a block bootstrap procedure and the control limit of the joint scheme could be designed. If, as it is the case for the sunspot numbers, all series of the panel do not possess an IC period, then more research is needed to see how such multivariate methods can be designed.

Multivariate monitoring methods based on neural networks may also be developed. Hundman et al. (2018) construct for instance a recurrent neural network based on long short-term memory (LSTM) layers (Hochreiter and Schmidhuber, 1997), which is used to predict the incoming values of the series. The prediction errors of those networks are then passed through a non-parametric dynamic threshold to determine if an anomaly occurred in the data.

To conclude, the work that has begun here can be extended in several directions. The methods developed in this thesis, either the error modelling or the quality control, remain to be applied on the processing of the International Sunspot Number and its building blocks. The non-parametric monitoring methods can also be further developed to include a multivariate monitoring of the mean and variance of panels of time-series. Machine learning based procedures also appear to provide interesting starting points for many research prospects.

References

R. Adipranata, G.S. Budhi, and B. Setiahadi. Automatic classification of sunspot groups for space weather analysis. *International Journal of Multimedia and Ubiquitous Engineering*, 8(3):41–54, 2013.

M.R. Anderberg. *Cluster Analysis for Applications*. Academic, 1973.

H.W. Babcock. The topology of the Sun's magnetic field and the 22-year cycle. *Astrophysical Journal*, 133:572–587, 1961. URL http://articles.adsabs.harvard.edu/full/1961ApJ...133..572B.

A.J. Bekker and J. Goldberger. Training deep neural-networks based on unreliable labels. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2682–2686, 2016. doi: 10.1109/ICASSP.2016.7472164.

C Bishop. *Pattern Recognition and Machine Learning*. Springer, 1st edition, 2006.

B. Biswas, P. Sengupta, and D. Chatterjee. Examining the determinants of the count of customer reviews in peer-to-peer home-sharing platforms using clustering and count regression techniques. *Decision Support Systems*, 135, 2020. doi: https://doi.org/10.1016/j.dss.2020.113324.

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

H. Brian Hwarng. Detecting process mean shift in the presence of autocorrelation: a neural-network based monitoring scheme. *International Journal of Production Research*, 42(3):573–595, 2004. doi: https://doi.org/10.1080/0020754032000123614.

P. Brockwell and P. Davis. *Time Series: Theory and Methods*. Springer, 2nd edition, 1991.

C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–267, 1998.

E. Carlstein. The use of subseries methods for estimating the variance of a general statistic from a stationary time series. *The Annals of Statistics*, 14:1171–1179, 1986.

E. Carlstein, K. Do, P. Hall, T.C. Hesterberg, and H. R. Künsch. Matched-block bootstrap for dependent data. *Bernoulli*, 4(3):305–328, 1998.

H.-Y. Chang and S.-J. Oh. Does correction factor vary with solar cycle? *Journal of Astronomy and Space Sciences*, 29(2):97–101, June 2012. doi: 10.5140/JASS.2012.29.2.097.

C-S. Cheng, P-W. Chen, and K-K. Huang. Estimating the shift size in the process mean with support vector regression and neural network. *Expert Systems with Applications*, 38(8):10624–10630, 2011. doi: https://doi.org/10.1016/j.eswa.2011.02.121.

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv*, 2014. arXiv:1406.1078.

François Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

A.R. Choudhuri. *Nature's Third Cycle: a story of sunspots*. Oxford University Press, 1st edition, 2015.

A.R. Choudhuri, M. Schussler, and M. Dikpati. The solar dynamo with meridional circulation. *Astronomy and Astrophysics*, 303:29–32, 1995.

F. Clette. Private Communication: Talk presented at 3rd Sunspot Number Workshop (Tucson, USA), 2013.

F. Clette and L. Lefèvre. The new sunspot number: assembling all corrections. *Solar Physics*, 291(9-10):2629–2651, November 2016. doi: 10.1007/s11207-016-1014-y.

F. Clette, D. Berghmans, P. Vanlommel, R. A. M. Van der Linden, A. Koeckelenbergh, and L. Wauters. From the Wolf number to the International Sunspot Index: 25 years of SIDC. *Advances in Space Research*, 40(7):919–928, 2007. doi: 10.1016/j.asr.2006.12.045.

T. Colak and R. Qahwaji. Automated McIntosh-based classification of sunspot groups using MDI images. *Solar Physics*, 248, 04 2008. doi: 10.1007/978-0-387-98154-3_7.

A. Colin Cameron and Pravin. K. Trivedi. *Regression Analysis of Count Data*. Cambridge University Press, 2nd edition, 2013.

D. Comaniciu and P. Meer. Maximum likelihood from incomplete data via the EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (5):603–619, 2002. doi: 10.1109/34.1000236.

J. Connor, L.E. Atlas, and D.R. Martin. Recurrent networks and NARMA Modeling. *Proceedings of the 4th International Conference on Neural Information Processing Systems*, pages 301–308, 1992.

J.J. Curto, M. Blanca, and E. Martinez. Automatic sunspots detection on full-disk solar images using mathematical morphology. *Solar Physics*, 250:411–429, 2008. doi: 10.1007/s11207-008-9224-6.

U. Dasgupta, S. Singh, and V. Jewalikar. Sunspot number calculation using clustering. In *2011 Third National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics*, pages 171–174, 2011. doi: 10.1109/NCVPRIPG.2011.43.

W. B. Davenport and W. L. Root. *Random Signals and Noise*. McGraw-Hill, 1968.

E.R. Davies. A modified Hough scheme for general circle location. *Pattern Recognition Letters*, 7(1):37–44, 1988.

W. De Mulder, S. Bethard, and M.F. Moens. A survey on the application of recurrent neural networks to statistical language modeling. *Computer Speech and Language*, 30, January 2014. doi: 10.1016/j.csl.2014.09.005.

B. Deketelaere. Control chart monitoring of non-normally distributed time series data". Master's thesis, Faculté des sciences, Université catholique de Louvain, 2020. URL http://hdl.handle.net/2078.1/thesis:27506.

A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B(Methodological)*, 39(1):1–38, June 1977.

S. D'Silva and A.R. Choudhuri. A theoretical model for tilts of bipolar magnetic regions. *Astronomy and Astrophysics*, 272:621–633, 1993. URL http://adsabs.harvard.edu/full/1993A&A...272..621D.

S. Dubey, J.N. Sarvaiya, and B. Seshadri. Temperature dependent photovoltaic (PV) efficiency and its effect on PV production in the world - a review. *Energy Procedia*, 33:311–321, 2013. doi: https://doi.org/10.1016/j.egypro.2013.05.072.

T. Dudok de Wit. A method for filling gaps in solar irradiance and solar proxy data. *Astronomy & Astrophysics*, 533:A29, September 2011. doi: 10.1051/0004-6361/201117024.

T. Dudok de Wit, L. Lefèvre, and F. Clette. Uncertainties in the sunspot numbers: estimation and implications. *Solar Physics*, 291(9-10):2709–2731, November 2016. doi: 10.1007/s11207-016-0970-6.

K. Ermolli, K. Matthes, T. Dudok de Wit, N.A. Krivova, K. Tourpali, M. Weber, Y.C. Unruh, L. Gray, U. Langematz, P. Pilewskie, E. Rozanov, W. Schmutz, A. Shapiro, S.K. Solanki, and T.N. Woods. Recent variability of the solar spectral irrandiance and its impact on climate modelling. *EGU publication : Atmospheric Chemistry and Physics*, 13:3945–3977, 2013. doi: https://doi.org/10.5194/acp-13-3945-2013.

M. Ester, H.P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In A. Simoudis, J. Han, and U. fayyad, editors, *KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, August 1996.

M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. Wiley, New-York, 3rd edition, 2000.

A. F. Zuur, E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith. *Mixed Effects Models and Extensions in Ecology with R*. Springer, 2009.

E. Feigelson and G. Babu. Linear regression in astronomy II. *The Astrophysical Journal*, 397(1):55–67, 1992.

P. Fryzlewicz and V. Delouille. A data-driven Haar-Fisz transform for multiscale variance stabilization. *IEEE Workshop on Statistical Signal Processing Proceedings*, 2005:539–544, 08 2005. doi: 10.1109/SSP.2005.1628654.

J. Haigh. The effects of solar variability on the Earth's climate. *Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences*, 361(1802):95–111, 2002. doi: https://doi.org/10.1098/rsta.2002.1111.

P. Hall, J. Horowitz, and B-Y. Jing. On blocking rules for the bootstrap with dependent data. *Biometrika*, 82(3):1561–574, 1995. doi: 10.2307/2337534.

D. H. Hathaway. The solar cycle. *Living Reviews in Solar Physics*, 7:1, March 2010. doi: 10.12942/lrsp-2010-1.

S Haykin. *Neural Networks and Learning Machines*. Pearson, 3rd edition, 2009.

G. Hinton, N. Srivastava, and K. Swersky. *Neural networks for machine learning lecture 6a overview of mini-batch gradient descent*. 2012.

J.L. Hintze and Nelson R. D. Violin plots: a box plot-density trace synergism. *The American Statistician*, 52(2):181–184, 1998. doi: 10.2307/2685478. URL http://www.jstor.org/stable/2685478.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. *Proceedings of 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 387–395, 07 2018. doi: 10.1145/3219819.3219845.

A.J. Izenman. J.R. Wolf and the Zürich sunspot relative numbers. *The Mathematical Intelligencer*, 7(1):27–33, 1985. doi: 10.1007/BF03023002.

G. Jain and P. Consul. A generalized negative binomial distribution. *SIAM Journal on Applied Mathematics*, 21(4):501–513, 1970. doi: https://doi.org/10.1137/0121056.

J. Kenney and E. Keeping. *Mathematics of Statistics, Part Two*. Van Nostrand, 2nd edition, 1951.

D. Kingma and J. Ba. Adam: a method for stochastic optimization. In *2015 3rd International Conference for Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6980.

T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990. doi: 10.1109/5.58325.

W. Kruskal and W. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952. doi: 10.1191/1471082X04st068oa. URL https://www.jstor.org/stable/2280779.

H. R. Künsch. The jackknife and the bootstrap for general stationary observations. *The Annals of Statistics*, 17(3):1217–1241, 1989. URL https://www.jstor.org/stable/2241719.

S.N. Lahiri. Theoretical comparisons of block bootstrap methods. *The Annals of Statistics*, 27(1):386–404, 1999. doi: https://doi.org/10.1007/978-1-4757-3803-2_5.

S.N. Lahiri. *Resampling Methods for Dependent Data*. Springer, 1st edition, 2003.

M. Langkvist, L. Karlsson, and A. Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11–24, 2014. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2014.01.008. URL https://www.sciencedirect.com/science/article/pii/S0167865514000221.

B. Laperre, J. Amaya, and G. Lapenta. Dynamic time warping as a new evaluation for Dst forecast with machine learning. *Frontiers in Astronomy and Space Sciences*, 7:39, 2020. ISSN 2296-987X. doi: 10.3389/fspas.2020.00039. URL https://www.frontiersin.org/article/10.3389/fspas.2020.00039.

J. Lederer. Activations functions in artificial neural networks: a systematic overview. *arXiv*, January 2021. arXiv:2101.09957.

R.B. Leighton. A Magneto-kinematic model of the solar cycle. *Astrophysical Journal*, 156:1–26, 1969. URL http://adsabs.harvard.edu/full/1969ApJ...156...1L.

B. Lim and S. Zhoren. Time-series forecasting with deep learning: a survey. *Philosophical transactions of the royal society A*, 2021. doi: https://doi.org/10.1098/rsta.2020.0209.

R. Y. Liu and K. Singh. Moving blocks jackknife and bootstrap capture weak dependence. In R Lepage and L. Billard, editors, *Exploring the Limits of Bootstrap*, pages 225–248. Wiley, New-York, 1992.

X. Liu, S. Li, M. Kan, S. Shan, and X. Chen. Self-error-correcting convolutional neural network for learning with noisy labels. In *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, pages 111–117, 2017. doi: 10.1109/FG.2017.22.

S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, June 1982.

S.P. Lloyd. Least squares quantization in PCM. Technical Report RR-5497 5497, Bell Lab, 1957.

G. Lorden. Procedures for reacting to a change in distribution. *The Annals of Mathematical Statistics*, 42:1897–1908, 1971.

U. Lotric and A. Dobnikar. Predicting time series using neural networks with wavelet-based denoising layers. *Neural Computing and Applications*, 14:11–17, 2005.

J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L.M. Le Cam and J. Neyman, editors, *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297. University of California Press, California, United States, 1967.

M. Makitalo and A. Foi. Optimal inversion of the generalized Anscombe transformation for Poisson-gaussian noise. *IEEE Transactions on Image Processing*, 22 (1):91–103, 2013. doi: 10.1109/TIP.2012.2202675.

F. Massey. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 2012.

S. Mathieu, R. von Sachs, V. Delouille, L. Lefèvre, and C. Ritter. Uncertainty quantification in sunspot counts. *The Astrophysical Journal*, 886(1):7, 2019a. doi: https://doi.org/10.3847/1538-4357/ab4990.

S. Mathieu, R. von Sachs, V. Delouille, L. Lefèvre, and C. Ritter. Modelisation et estimation du nombre de taches solaires. In *Colloque du Groupement de Recherche En Traitement du Signal et des Images (GRETSI)*, August 2019b.

S. Mathieu, L. Lefèvre, R. von Sachs, V. Delouille, C. Ritter, and F. Clette. Nonparametric monitoring of sunspot number observations. *Journal of Quality Technology*, 2021. Tentatively accepted.

Patrick S. McIntosh. The classification of sunspot groups. *Solar Physics*, 125(2): 251–267, September 1990. doi: 10.1007/BF00158405.

T. Mikolov, M. Karafiàt, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. *Proceedings of the 11th Annual Conference of the International Speech Communication Association, INTERSPEECH 2010*, 2:1045–1048, 01 2010.

D Montgomery. *Introduction to Statistical Quality Control*. Wiley, New-York, 5th edition, 2004.

G.E. Morfill, H. Scheingraber, W. Voges, and C.P. Sonett. Sunspot number variations -stochastic or chaotic. In C.P. Sonett, M.S. Giampapa, and Matthews, editors, *The Sun in Time*, pages 30–58. University of Arizona Press, Tucson, United States, January 1991.

G. Moustakides. Optimal stopping times for detecting changes in distributions. *The Annals of Statistics*, 14(4):1379–1387, 1986.

F. Murtagh, J.-L. Starck, and A. Bijaoui. Image restoration with noise suppression using a multiresolution support. *Astronomy and Astrophysics Supplement*, 112: 179–189, 1995. URL http://adsbit.harvard.edu//full/1995A%26AS..112..179M/0000181.000.html.

H. Neckel and L. Dietrich. Solar limb darkening 1986-1990 ($\lambda\lambda$303 to 1099 nm). *Solar Physics*, 153:91–114, 1994.

T. T. Nguyen, C. P. Willis, D. J. Paddon, and H. S. Nguyen. A hybrid system for learning sunspot recognition and classification. In *2006 International Conference on Hybrid Information Technology*, volume 2, pages 257–264, 2006. doi: 10.1109/ICHIT.2006.253620.

B. Owens. Long-term research: slow science. *Nature*, 495(7441):300, March 2013. doi: 10.1038/495300a.

M. Owens, M. Lockwood, E. Hawkins, I. Usoskin, G. Jones, L. Barnard, A. Schurer, and J. Fasullo. The Maunder minimum and the Little Ice Age: an update from recent reconstructions and climate simulations. *J. Space Weather Space Clim.*, 7:A33, 2017. doi: 10.1051/swsc/2017034. URL https://doi.org/10.1051/swsc/2017034.

M. Pacella and Q. Semeraro. Using recurrent neural networks to detect changes in autocorrelated processes for quality monitoring. *Computers and Industrial Engineering*, 52(4):502–520, 2007. doi: https://doi.org/10.1016/j.cie.2007.03.003.

E. S. Page. Cumulative sum charts. *Technometrics*, 3(1):1–9, 1961. URL https://www.jstor.org/stable/1266472.

E. N. Parker. Hydromagnetic dynamo models. *Astrophysical Journal*, 122:293–314, 1955.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12:2825–2830, 2011.

W. D. Pesnell. Lessons learned from predictions of Solar Cycle 24. *J. Space Weather Space Clim.*, 10(60), 2020. URL https://www.swsc-journal.org/articles/swsc/full_html/2020/01/swsc200057/swsc200057.html.

K. Petrovay. Solar cycle prediction. *Living Reviews in Solar Physics*, 17(2), 2020. URL https://link.springer.com/content/pdf/10.1007/s41116-020-0022-z.pdf.

W. Pohlmeier and V. Ulrich. An econometric codel of the two-part decisionmaking process in the demand for health care. *The Journal of Human Resources*, 30(2):339–361, 1995. doi: https://doi.org/10.2307/146123.

D. N. Politis and J. P. Romano. A circular block-resampling procedure for stationary data. In R Lepage and L. Billard, editors, *Exploring the Limits of Bootstrap*, pages 263–270. Wiley, New-York, 1992.

D. N. Politis and J. P. Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89:1303–1313, 1994.

B. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

P. Qiu. *Introduction to Statistical Process Control*. CRC Press, Taylor and Francis Inc, 1st edition, 2013.

P. Qiu and D. Xiang. Univariate dynamic screening system: an approach for identifying individuals with irregular longitudinal behaviour. *Technometrics*, 56(2):248–260, 2014. ISSN 0040-1706. doi: https://doi.org/10.1080/00401706.2013.822423.

P. Qiu, X. Zi, and C. Zou. Nonparametric dynamic curve monitoring. *Technometrics*, 60(3):386–397, 2017. ISSN 0040-1706. doi: https://doi.org/10.1080/00401706.2017.1361340.

S. W. Roberts. Control chart tests based on geometric moving averages. *Technometrics*, 1(3):239–250, 1959. URL https://www.jstor.org/stable/1266443.

G. Rodriguez. Models for count data with overdispersion. *https://data.princeton.edu/wws509/notes/c4addendum.pdf*, 2013. URL https://data.princeton.edu/wws509/notes/c4addendum.pdf.

L. Rodriguez-Villamizar, Belalcàzar-Ceron L.C., Fernàndez-Niño J.A., D.M. Marìn-Pineda, O.A. Rojas-Sànchez, L.A. Acuña-Merchàn, N. Ramìrez-Garcìa, S.C. Mangones-Matos, J.M. Vargas-Gonzàlez, J. Herrera-Torres, D.M. Agudelo-Castañeda, J.G. Piñeros Jimènez, N.Y. Rojas-Roa, and V.M. Herrera-Galindo. Air pollution, sociodemographic and health conditions effects on COVID-19 mortality in Colombia: An ecological study. *Science of The Total Environment*, 756, 2021. doi: https://doi.org/10.1016/j.scitotenv.2020.144020.

P. Rousseeuw, I. Ruts, and J. Tukey. The bagplot: a bivariate boxplot. *The American Statistician*, 53(4):382–387, 2012. doi: 10.1080/00031305.1999.10474494. URL https://www.tandfonline.com/doi/pdf/10.1080/00031305.1999.10474494.

P.J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. doi: https://doi.org/10.1016/0377-0427(87)90125-7.

D.E. Rumelhart, G.e. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan. Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*, pages 166–171. IEEE, June 2011. doi: 10.1109/ICDCSW.2011.20.

B. Schaefer. Improvements for the AAVSO sunspot number. *JAAVSO*, 26(1): 47–49, 1997.

Robin M. Schmidt. Recurrent neural networks (RNNs): a gentle introduction and overview. *arXiv*, 2019. arXiv:1912.05911.

D. W. Scott. On optimal and data-based histograms. *Biometrika*, 66(3):605–610, 1979. URL http://www.jstor.org/stable/2335182.

W.A. Shewhart. *Economic Control of Quality of Manufactured Product*. Van Nostrand, 1st edition, 1931.

A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

S. Smyl. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1), 2019. doi: DOI:10.1016/j.ijforecast.2019.03.017.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

J.Q. Stewart and F.C. Eggleston. The mathematical characteristics of sunspot variations. II. *Astrophysical Journal*, 91:72–82, 1940. doi: 10.1086/144147.

J.Q. Stewart and H.A.A. Panofsky. The mathematical characteristics of sunspot variations. *Astrophysical Journal*, 88:385–407, 1938. doi: 10.1086/143994.

S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus. Training convolutional networks with noisy labels. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

J. Taylor and A. Verbyla. Joint modelling of location and scale parameters of the t distribution. *Statistical Modelling*, 4(2):91–112, 2004. doi: 10.1191/1471082X04st068oa. URL https://doi.org/10.1191/1471082X04st068oa.

M. Temmer, A. Veronig, A. Hanslmeier, W. Otruba, and M. Messerotti. Statistical analysis of solar Hα flares. *Astronomy and Astrophysics*, 375(3):1049–1061, 2001. doi: https://doi.org/10.1051/0004-6361:20010908.

J. Thomas and N. Weiss. *Sunspots and Starspots*. Cambridge University Press, 1st edition, 2008.

G. Usoskin, K. Mursula, and G. A. Kovaltsov. Reconstruction of monthly and yearly group sunspot numbers from sparse daily observations. *Solar Physics*, 218(1-2):295–305, November 2003. doi: https://doi.org/10.1023/B:SOLA.0000013029.99907.97.

Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

V Vapnik. *Statistical Learning Theory*. Wiley, New-York, 1st edition, 1998.

M. Verleysen and J. Lee. ELEC2870- Machine learning: regression and dimensionality reduction. University Lecture, Université catholique de Louvain, 2017.

M. Veta, P. J. van Diest, S. Willems, H. Wang, A. Madabhushi, A. Cruz-Roa, F. Gonzalez, A. Larsen, J. Vestergaard, Dahl A., D. Cireşan, J. Schmidhuber, A. Giusti, L. Gambardella, B. Tek, T. Walter, C.-W. Wang, S. Kondo, B. Matuszewski, F. Precioso, V. Snell, J. Kittler, T. de Campos, A. Khan, N. Rajpoot, E. Arkoumani, M. Lacle, M. Viergever, and J. Pluim. Assessment of algorithms for mitosis detection in breast cancer histopathology images. *Medical Image Analysis*, 20(1):237–248, 2015. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2014.11.010. URL https://www.sciencedirect.com/science/article/pii/S1361841514001807.

A. Vigouroux and PH. Delache. Sunspot numbers uncertainties and parametric representations of solar activity variations. *Solar Physics*, 152(1):267–274, 1994.

M. Waldmeier. Die Zonenwanderung der Sonnenflecken. *Astronomische Mitteilungen der Eidgenössischen Sternwarte Zürich*, 14:470–481, January 1939.

Y.-M. Wang and R. Colaninno. Is Solar Cycle 24 producing more coronal mass ejections than Cycle 23? *The Astrophysical Journal Letters*, 784(L27):1–7, 2014. doi: https://doi.org/10.1088/2041-8205/784/2/L27.

Z. Wang, A.C. Bovik, H.R. Sheikh, and Simoncelli E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

F. Watson and L. Fletcher. Automated sunspot detection and the evolution of sunspot magnetic fileds during solar cycle 23. In D.P. Choudhary and Strassmeier K.G., editors, *Physics of Sun and Star Spots*, volume 273, pages 51–55, Cambridge, United Kingdom, September 2010. Cambridge University Press. doi: 10.1017/S1743921311014992.

F. Watson, L. Fletcher, S. Dalla, and S. Marshall. Modelling the longitudinal asymmetry in sunspot emergence: the role of the Wilson depression. *Solar Physics*, 260:5–19, 2009. doi: 10.1007/s11207-009-9420-z.

F.T. Watson, L. Fletcher, and S. Marshall. Evolution of sunspot properties during solar cycle 23. *Astronomy and Astrophysics*, 533(A14), 2011. doi: 10.1051/0004-6361/201116655.

R. Zemouri, D. Racoceanu, and N. Zerhouni. Recurrent radial basis function network for time-series prediction. *Engineering Applications of Artificial Intelligence*, 16(5):453–463, 2003. ISSN 0952-1976. doi: https://doi.org/10.1016/S0952-1976(03)00063-0. URL https://www.sciencedirect.com/science/article/pii/S0952197603000630.

S. Zharkov, V. Zharkova, S. Ipson, and A. Benkhalil. Technique for automated recognition of sunspots on full-disk solar images. *EURASIP J. Adv. Sig. Proc.*, 2005:2573–2584, 08 2005. doi: 10.1155/ASP.2005.2573.

V.V. Zharkova, S.S. Ipson, S.I. Zharkov, Benkhalil A., Aboudarham J., and Bentley R.D. A full-disk image standardisation of the synoptic solar observations at the Meudon Observatory. *Solar Physics*, 214:89–105, 2003. doi: 10.1023/A:1024081931946.