Universidad de Alcalá

ESCUELA DE DOCTORADO

Universidad de Alcalá

Programa de Doctorado en

Investigación Espacial y Astrobiología

Application of signal processing methods to the Rover Environmental Monitoring Station data for the analysis of environmental processes on Mars

Tesis Doctoral presentada por

SOFÍA ZURITA ZURITA

2022

Universidad de Alcalá

ESCUELA DE DOCTORADO

Universidad de Alcalá

Programa de Doctorado en

Investigación Espacial y Astrobiología

Application of signal processing methods to the Rover Environmental Monitoring Station data for the analysis of environmental processes on Mars

Tesis Doctoral presentada por

SOFÍA ZURITA ZURITA

Directores:

Dr. Francisco Javier Escribano Aparicio
Dr. José Antonio Rodríguez Manfredi

Alcalá de Henares, 4 de noviembre 2022

Universidad de Alcalá

ESCUELA DE DOCTORADO

Agradecimientos

A mis directores, Francisco J. Escribano Aparicio y José Antonio Rodríguez Manfredi, por su tiempo y dedicación y por ayudarme en los momentos en los que no veía salida. Me hicisteis ver que había luz al final del túnel.

A mi familia, por su apoyo incondicional y por valorar tanto este esfuerzo realizado. Culmino hoy este duro camino junto a vosotros y os doy las gracias por haberme acompañado estos años.

Para África y Jacobo.

Abstract

The present thesis analyzes data from two of the sensors of the Mars Science Laboratory's REMS instrument, which has been active on Mars since August 2012: the Pressure Sensor and the Temperature Sensors. The information contained in the data from these sensors is very valuable and revealing for a better understanding of the meteorological agents that take place daily on Mars. It is necessary to know in depth the operation of both sensors and the data processing flow followed to transform the digital data received from Mars into tangible information. The main goal of this thesis is to use signal processing methods that help us to extract useful information contained in the sensors' data not visible at first sight. It is desired to extract information related to specific environmental processes on Mars, thus contributing to a better understanding of our neighboring planet. With all this acquired knowledge, an exhaustive research of signal processing methods has been made to find the best methods that fit our data and help us to find the information related to the environmental processes.

INDEX

List of Figures

List of Tables

List of Abbreviations

ADC	**A**nalog **D**igital **C**onversion
ADR	**A**ncillary **D**ata **R**ecords
ASCII	**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange
ASIC	**A**pplication **S**pecific **I**ntegrated **C**ircuit
ATS	**A**ir **T**emperature **S**ensor
CAB	***C**entro de **A**stro**b**iología*
CMSE	**C**onsecutive **M**ean **S**quare **E**rror
DWT	**D**iscrete **W**avelet **T**ransform
EDR	**E**xperimental **D**ata **R**ecord
EMD	**E**mpirical **M**ode **D**ecomposition
FEI	**F**ile **E**xchange **I**nterface
FIR	**F**inite **I**mpulse **R**esponse
FMI	**F**innish **M**eteorological **I**nstitute
GTS	**G**round **T**emperature **S**ensor
HHT	**H**ilbert-**H**uang **T**ransform
ICU	**I**nstrument **C**ontrol **U**nit
IMF	**I**ntrinsic **M**ode **F**unction
JPL	**J**et **P**ropulsion **L**aboratory
LMST	**L**ocal **M**ean **S**olar **T**ime
LTST	**L**ocal **T**rue **S**olar **T**ime
MA	**M**oving **A**verage
MAD	**M**edian **A**bsolute **D**eviation
MIPL	**M**ultimission **I**mage **P**rocessing **L**aboratory
MSL	**M**ars **S**cience **L**aboratory
OPGS	**O**perations **P**roduct **G**eneration **S**ubsystem
PDS	**P**lanetary **D**ata **S**ystem
QRS	**Q**uick **R**esponse **S**ystem
RDR	**R**educe **D**ata **R**ecord
REMS	**R**over **E**nvironmental **M**onitoring **S**tation
RTG	**R**adioisotope **T**hermoelectric **G**enerator
TVT	**T**hermal **V**acuum **T**est
WEB	**W**arm **E**lectronics **B**ay

Chapter 1: Motivation

Mars is probably one of the most studied planets in the solar system due to the similarities it may have had in the past with our planet Earth [1]. The search for biomarkers, the study of the climate and atmosphere, the study of the planet geology and the research for human exploration are the main pillars of study respecting our neighboring planet [2].

In many ways the climate of Mars resembles that of the Earth, particularly in its daily cycle and yearly sequence of seasons. These affinities result from the many coincidences in the celestial motions of the two planets: the Martian day, or sol, is 24 h 40 m long, a little bit longer than the terrestrial day. Mars completes an orbit around the Sun in approximately 2 terrestrial years. However, due to the eccentricity of the Martian orbit, Mars receives only about half as much sunlight as the Earth. This is the reason why the Martian surface is colder and experiences greater seasonal temperature changes and more pronounced variations throughout the day, with differences around 70 degrees or more between day and night.

The thickness and the composition of the Martian atmosphere bears no resemblance with its terrestrial counterpart [3]. It is mainly composed of 95 % carbon dioxide and Nitrogen, and Argon represents the remaining 5 %. Oxygen and water represent less than 0.2 % of the Martian atmosphere. Moreover, it is extremely tenuous, unlike the Earth atmosphere, with an average surface pressure of 6.1 mbar (600 Pa), whereas the mean surface pressure on Earth is 1013 mbar (100 KPa). The pressure on the Martian surface varies little with altitude, from the top of the highest mountains, where it drops to around 4 mbar (400 Pa), to lower areas, where it reaches 10 mbar (1000 Pa). In the case of the Earth, it can vary from 34 KPa to 100 KPa. This is certainly more than enough for liquid water to be stable provided, of course, that the temperature gets temporarily above 0 ^{0}C, which indeed occurs in regions not too far from the Equator during Summer afternoons. However, until today, and in spite of all the efforts invested in the task, the proof of the existence of liquid water in present-day Mars remains elusive, although there is circumstantial evidence which points its presence confined to a shallow layer below the surface [4][5].

Because of all this, the Mars atmosphere has attracted a huge interest in the scientific community and has been studied for centuries. The air dynamics characterization, the thermal structure, the distributions of dust, water, water clouds and carbon dioxide in the lower atmosphere are some of the main atmospheric characteristics traditionally studied.

Dust storms are probably one of the Martian atmospheric events that has attracted most attention since they were observed, for the first time, by Earth telescopes in the late nineteenth century [6], [7]. They affect the global atmospheric circulation of the planet [8], and also the transport of carbon dioxide, water and dust.

To be able to study the Red Planet, a great effort has been made, since the 1960s, in order to send orbiters [2] and land missions [9] to our neighbor planet. Consequently, information of incalculable value has already reached our hands. With this huge amount of data, the scientific community studies in depth the different research areas mentioned above. More specifically, the Curiosity Mission, landed at the crater Gale in August 2012, is equipped with a meteorological station, the Rover Environmental Monitoring Station (REMS), which continues monitoring a multiplicity of physical magnitudes from the planet, such as wind direction, wind speed, humidity, pressure, ground temperature, ambient temperature and ultraviolet radiation [10]. Thanks to all of these physic magnitudes, the scientific community has been able to analyze in more detail the Martian atmosphere, specifically in the crater Gale.

Because of all this, this thesis aims to contribute in a small way to expanding the current knowledge of the climate and atmosphere of Mars. By using several signal processing methods, this work presents the analysis of data from two sensors, the Pressure and Air Temperature Sensors, to detect environmental processes in Mars.

According to these aims, this document is organized in 6 chapters, namely:

Chapter 1 reviews the state-of-the-art of the mission, the Air Temperature and Pressure Sensors, and the data collected by these sensors, Chapter 2 explains the different signal processing methodologies used to process and analyze this kind of data, Chapter 3 explains how these methods have been applied to the Pressure Sensor, Chapter 4 explains how the method have been applied to the Air Temperature Sensor, Chapter 5 details the main conclusions of the thesis and finally, Chapter 6 states possible future research lines enabled by the findings and developments of this work.

Chapter 2: Introduction. State-of-the-art

After the first introductory chapter, which emphasizes the reasons that led us to carry out this thesis, this second chapter aims to put the reader in context. For this reason, this second chapter is devoted to explain, in detail, the Mars Science Laboratory (MSL) mission, along with its meteorological station and the associated data sent from Mars.

2.1. Mars Science Laboratory mission

The history of Mars observations began thousands of years ago. First telescopic observations of Mars were made by Galileo Galilei in 1610 [11]. Our immense interest in the study of this planet has grown in the last century may be due to the discovery of the similarities between its atmosphere and the terrestrial one. After the identification of certain characteristics of the Red Planet that could generate or have generated life, the human being has intensified his research in our neighboring planet. This is the reason why, from the decade of 1960, numerous space missions have been trying to reach the Red Planet in order to collect information and study its atmosphere and soil, and reveal whether there were past or present signs of life [12][13]. The Mars Rover Curiosity [14] (part of the Mars Science Laboratory) is one of the missions that is currently on the Martian surface. It was launched from Cape Canaveral on 26 November 2011 and landed on 6 August 2012, on Aeolis Palus, inside Gale crater on Mars. The mission goals include a serious inquiry on the Martian geology and climate. The rover is still operational and as of September 29, 2022, Curiosity has been active on Mars for 3619 sols (10 years, 65 days since its landing). Since then, the rover has been sending very valuable information about the planet, specifically related to its landing location and immediate surroundings, at the Gale crater.

Curiosity contains multiple sensors and cameras, as shown in Figure 1. Among these, the Curiosity rover is equipped with a meteorological station, the Rover Environmental Monitoring Station, better known as REMS [10][15], provided by the Spanish Ministry of Education and Science. Moreover, the rover contains, among multiples cameras, the MastCam camera, that belongs to NASA/Jet Propulsion Laboratory (JPL). We mention this camera in particular, because the opacity values, and images taken by this camera will be used to contrast the results obtained in this research.

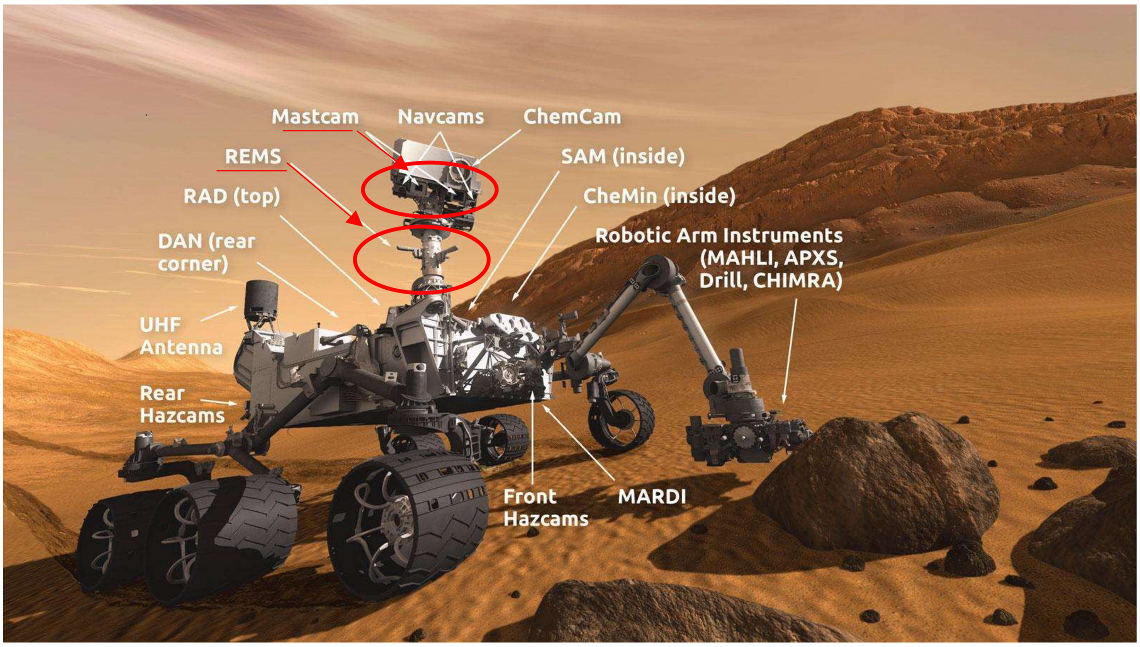

Figure 1: Artistic view of Curiosity, where instruments are identified by name, including sensors, antennas and cameras for different purposes. (Credit: NASA/JPL-Caltech)

2.2. The Rover Environmental Monitoring Station

REMS, located in the Mars Science Laboratory, has as its main goal to explore factors directly related to the Mars habitability [16][17][18]. This instrument is a meteorological package designed to provide measurements of air and ground temperature, pressure, wind speed and direction, humidity and ultraviolet radiation. All these sensors are located around three elements: two booms attached perpendicularly to the rover's mast (that house the Air and Ground Temperature Sensors, the Wind and the Humidity Sensors), the Ultraviolet Sensor (UVS) assembly located on the rover top deck, and the Instrument Control Unit (ICU) inside the rover body. Each boom contains an Application Specific Integrated Circuit (ASIC), whose function is to control and manage the sensors of each boom. Their electrical interconnection is shown, in a block diagram, in Figure 4.

Then, we explain more in detail the different sensors that make up REMS station [10]:

1. The Ground Temperature Sensor (GTS): It records the infrared (IR) brightness temperature of the Martian surface by using three thermopiles. The sensor is mounted on Boom 1, positioned in the NASA/MSL Rover mast at 1.6 m height, and it is directly connected to the Sensor Front-End electronics. To avoid local temperature effects, the GTS focuses on a large ellipsoidal ground surface area of around 100 m^2, measuring its average temperature. The sensor includes an active self-calibration system to

compensate for its potential degradation during the mission associated with dust collection on the sensor window.

2. The Air Temperature Sensor (ATS): It comprises two sensors [16], each one mounted on a different boom. Each sensor assembly consists of a small FR4 rod that has been manufactured to have low thermal conductivity and that contains 3 thermistors placed along the rod. The readings of all the thermistors, after being processed, determine the ambient temperature.

3. The Wind Sensor (WS): It comprises two sensors, each one mounted on a different boom. They measure the horizontal and vertical wind speed as well as the wind direction. They are based on hot film anemometry and each boom is equipped with three recording points. The wind data from both booms is processed to determine the estimated wind speed and direction.

4. The Pressure Sensor (PS): The sensor, located in the rover body, is composed of a Vaisala Barocap and an associated Vaisala Thermocap, both provided by the Finnish Meteorological Institute (FMI). The sensor measures the atmospheric pressure in the range of 0-1400 Pa.

5. The Humidity Sensor (HS): The sensor, located in Boom 2, is also provided by the FMI. It is composed of a Vaisala Humicap relative humidity sensor head and an associated Vaisala Thermocap. The Humidity Sensor measures the relative humidity in the 0 – 100 % range, within the temperature range of 203 K – 323 K.

6. The Ultraviolet Sensor (UVS): The sensor, located on the rover deck, is composed of six broadband photodiodes with different spectral support and a thermistor which monitors the temperature of the UVS. Each broadband photodiode shall provide a raw evaluation of the incident UV flux within its range of responsivity that can be directly compared with their equivalent measurements on Earth or on the top of the Martian atmosphere.

7. The Instrument Control Unit (ICU): This unit provides the interface with the rover in terms of data exchange, power and telemetry, receives digital data from the boom sensors, powers the sensor front-end electronics, and processes data from the HS, UVS and PS. WS, ATS and GTS instruments communicate with the ICU through the ASIC. See Figure 4 for clarification.

Figure 2 and Figure 3 show different perspectives of the rover and the location of the REMS sensors.

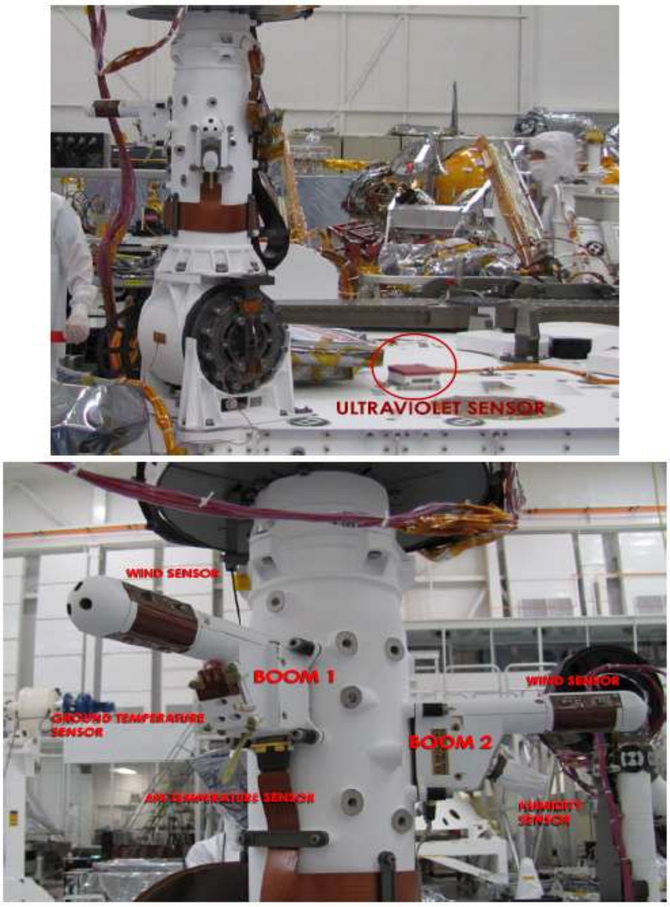

Figure 2: Ultraviolet Sensor and Close-up view of both booms. (Credit: NASA/JPL)

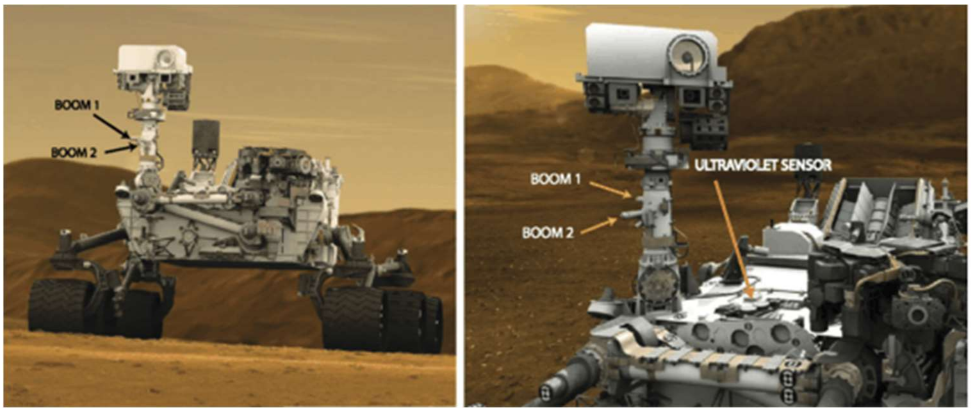

Figure 3: Artistic views of Curiosity. Positions of the REMS Booms and the Ultraviolet Sensor are shown. (Credit: Nasa/JPL-Caltech)

As mentioned before, Figure 4 shows a general scheme of the sensor electrical interconnections. ATS 1 and ATS 2 represent the Air Temperature Sensors, WS 1 and WS 2 represent the Wind Sensors, GTS represents the Ground Temperature Sensor, HS represents the Humidity Sensor, PS represents the Pressure Sensor and UV, the Ultraviolet Sensor.

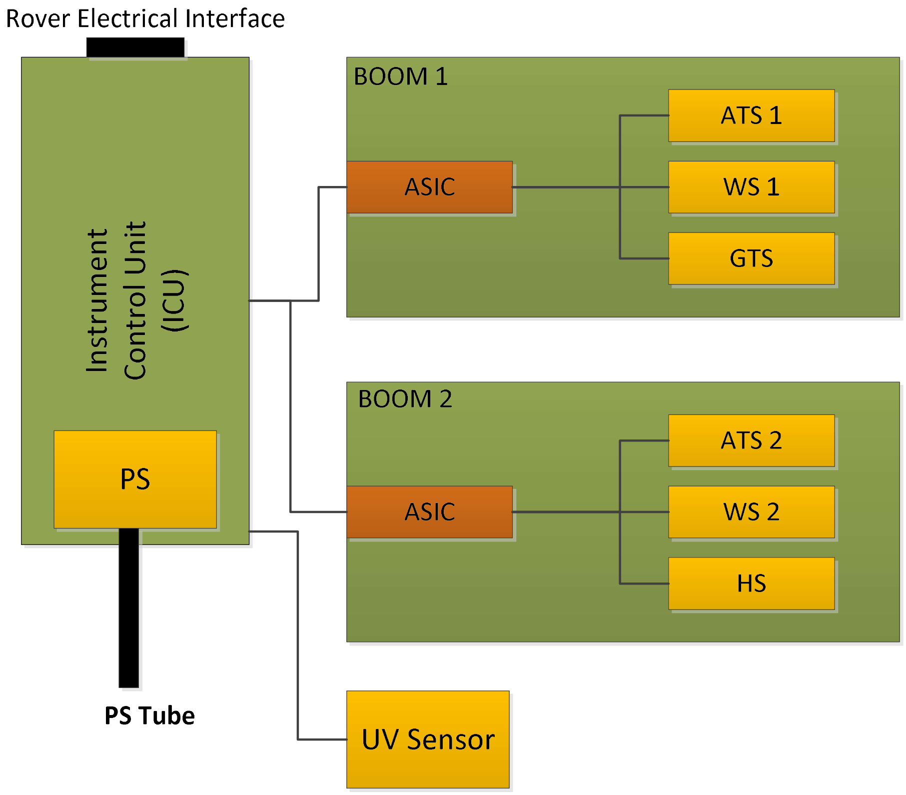

Figure 4: REMS scheme of the sensor electrical interconnections.

In the sequel, the objective sensors of this thesis, ATS and PS, are described in detail.

2.2.1 Pressure Sensor

The Pressure Sensor was designed and integrated by the Finnish Meteorological Institute (FMI). For this reason, it is not possible to describe this sensor with the same detail as the Temperature Sensor, since part of its operation, its measurement scheme and its mechanics are confidential.

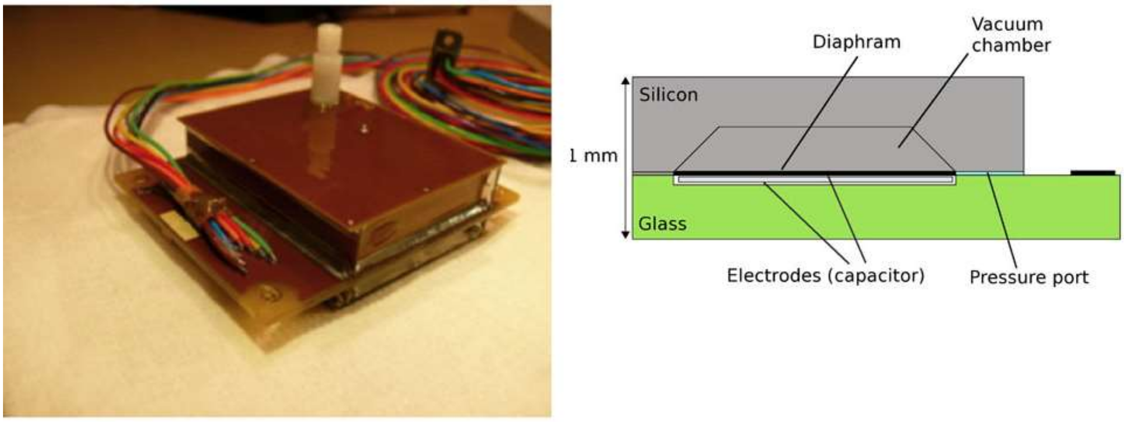

Figure 5: Left: Pressure Sensor; Right: Pressure Sensor scheme. (Credit: FMI)

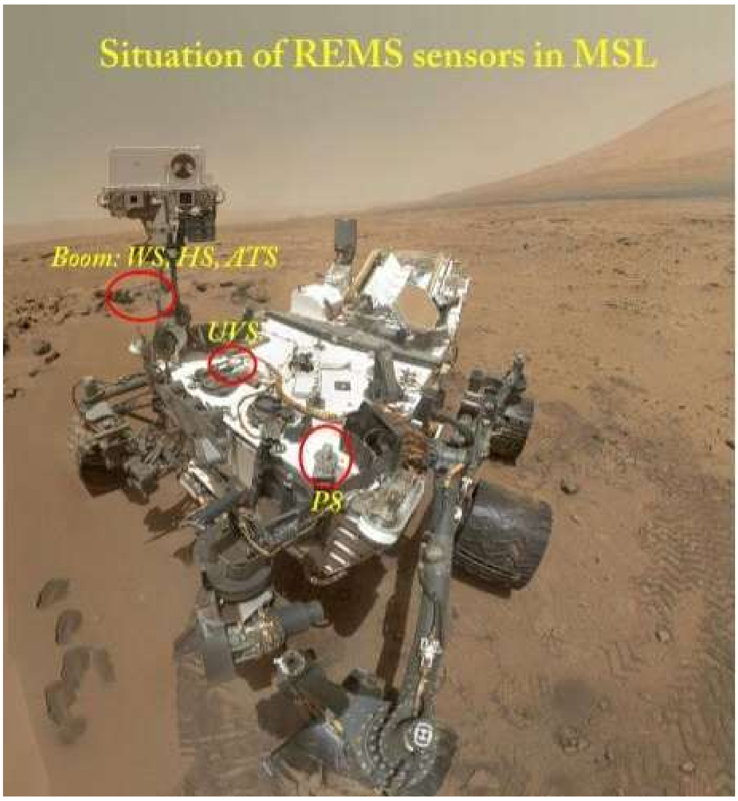

Figure 6: REMS Sensors in MSL. (Credit: NASA/JPL-Caltech/ Malin Space Science Systems)

The Pressure Sensor provides information about the atmospheric pressure around the MSL Rover. Figure 5 shows a scheme of the sensor and a real image of the sensor, and Figure 6 shows the location of the PS in the rover body, accommodated inside the ICU box. The PS uses a small tube, see Figure 4, to interface with the Martian atmosphere through a High Efficiency Particle Arrester (HEPA) filter, and it is covered with a dust protection cap.

The PS makes use of two transducers placed on a single multi-layer PCB. The transducers of the Pressure Sensor can be used in turn, thus providing some redundancy and improved reliability. Each transducer has 2 Vaisala Barocap Pressure Sensor heads and 2 Thermocap Temperature Sensors. The Barocap sensor heads are of different types: 1 is of high-stability, and 3 are of high-resolution type. Each Barocap sensor head is a single-crystal silicon micromachined device, therefore its intrinsic stability is extremely good. The measurement is based on capacitor plates (electrodes) moved by pressure, whose action changes the capacitance of the sensor head.

The Pressure Sensor measures the atmospheric pressure between 1 a 1150 Pa. The sensor has an accuracy of 2 Pa and a resolution of 0,2 Pa.

The sensor has been calibrated for a Martian pressure range of 4–12 hPa and its operational temperature is limited from −45 ^{0}C to +55 ^{0}C. The resolution of the sensor, in the case of the high-resolution sensor heads, is measured to be 0.2 Pa, while the response time of the overall pressure measurement system is approximately 1 second. The stability of the high-stability sensor head during the surface operations phase is estimated to be less than 1 Pa/year. The root-mean-square (RMS) of the absolute accuracy of the high-stability sensor head was estimated to be 3.5 Pa after the first few weeks after landing.

2.2.2. Air Temperature Sensor

REMS is equipped with two Air Temperature Sensors, called ATS 1 and ATS 2. Each REMS ATS [19][20] consists of a 35 mm long thin fin manufactured with an FR4 multilayer structure. To measure the temperature, thermistors of type Pt1000 Class A, with 1.2 mm x 1.6 mm size, are attached to the rod. The ATS 1, mounted on the so-called Boom 1, see in Figure 7, is equipped with 2 Pt1000 sensors, glued to the tip and to an intermediate position of the corresponding small FR4 rod, respectively. The Ground Temperature Sensor is placed at the base of the ATS 1, which, among other functions, provides the temperature at the base of the FR4 rod at Boom 1. The ATS 2, placed on the so-called Boom 2, has 3 Pt1000 sensors, which are glued to the tip, to an intermediate position and to the base of a similar FR4 rod, respectively. Both booms are depicted in Figure 8 and Figure 9. Each boom contains an Application-Specific Integrated Circuit (ASIC), whose function is to control and manage its different sensors, including the ATS. Figure 4 shows how they are both linked to their respective ASIC. Each ATS works at a sampling frequency of 1 sample per second, can measure temperatures within the interval 150-300 K, offers an absolute accuracy of at least 5 K and a resolution of 0.1 K.

Both booms point to directions separated 120 degrees, as we can see in Figure 2 and Figure 8, and they are placed perpendicularly on the rover main mast. They operate at 1.6 m above the Martian surface, in the lower part of the boom structure.

Each boom provides an ambient temperature measurement by using their Pt1000 readings (together with the GTS readings for the temperature at the base of the ATS 1), which are the inputs for a model, called the m-model, which is detailed in *Annex 2.* Said model is based on the theory of the heat exchange from an infinite fin, as described in the Mueller experiment [21].

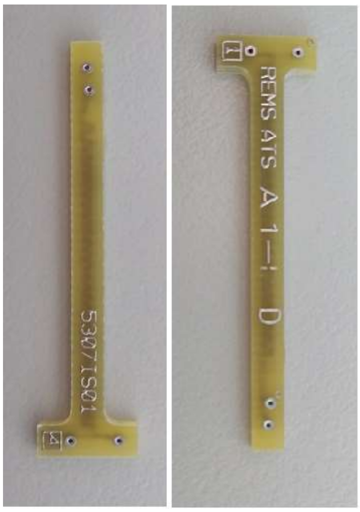

Figure 7: ATS 1 model.

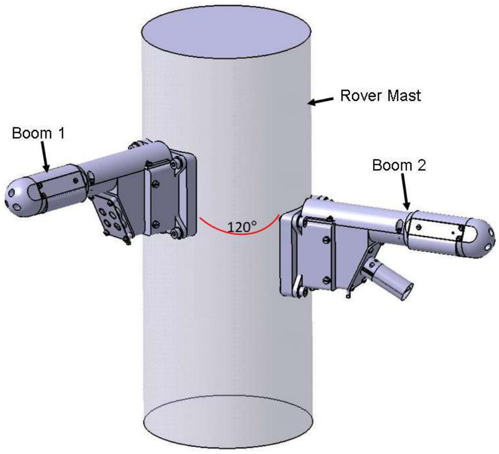

Figure 8: Arrangement of the sensor on the rover mast. (Credit: CAB-CSIC-INTA)

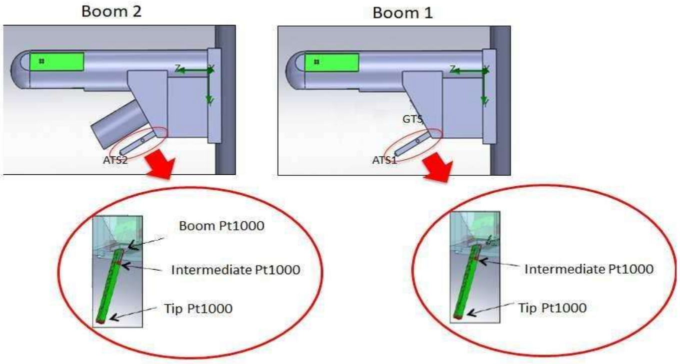

Figure 9: Detailed depiction of the ATS instrument for each boom. (Credit: CAB-CSIC-INTA)

2.3. REMS data management

This subsection is devoted to explaining how the data collected by REMS is initially processed, because this knowledge will be the primary source for the processing methods used in this thesis. Before we detail how these data are processed, we explain next the so-called Planetary Data System (PDS), which is the public database where the different mission data are stored.

The Planetary Data System (PDS) is a database that contains data from all NASA missions, organized in different processing levels, as we will explain below. All data used for this thesis can be found there. In particular, instead of resorting to raw measurements, we have used reduced data from the REMS ATS and PS measurements.

In the following, we explain the different processing levels of the data presented in the PDS.

Processing starts with the generation of the REMS Experimental Data Record (EDRs). The REMS EDR data products are generated by the Multimission Image Processing Laboratory (MIPL) at JPL, under the Operations Product Generation Subsystem (OPGS), using the telemetry processing software called MSLEdrGen, another product from JPL, which converts the binary data received as telemetry to ASCII, and which generates a separate table for each of the data types reflected in Table 1. The EDRs are then retrieved at *Centro de Astrobiología* (INTA-CSIC) by using a data file exchange interface. The EDR data products receive a first automatic processing using previously stored calibration data. The result of this is the Thermal and Electrical Reduced Data Record (TELRDR), which contains electrical and thermal magnitudes. In parallel, using ancillary data provided by JPL (such as rover location, or sun position) the Ancillary Data Records (ADRs) are generated. The EDRs, TELRDRs, ADRs, and calibration data are processed together to obtain the Environmental Magnitudes RDR (ENVRDR).

In the conversion process from EDRs to TELRDR, the specific procedure employed to calculate the electrical magnitudes from raw data, also named as *counts*, varies from sensor to sensor. These counts are the result of a digitization from an Analog/Digital Converter, so the reverse process, digital-to-analog, is done using data obtained during the calibration of the instrument. These data are included in the CALIB directory within the PDS.

As mentioned before, both ATS are handled through an ASIC and are affected by its temperature and power conditions. The ASICs status and their effects are included in the processing, in addition to the digital-to-analog conversion. ASIC information is also included in the first processing level, the EDRs.

The ENVRDRs products include environmental magnitudes with minimal corrections (mainly based on the degradation of the sensors), calculated by an algorithm that has been specifically designed for each sensor. Finally, by applying models developed by the REMS team, and by refining them with the use of confidence levels, the Models RDR (MODRDR) are created.

By using a custom software program named Quick Response System (QRS), the REMS team will generate RDR data products from the EDR data products provided by MIPL.

Figure 10 summarizes the REMS processing mentioned and their sequence. They are going to be further explained next.

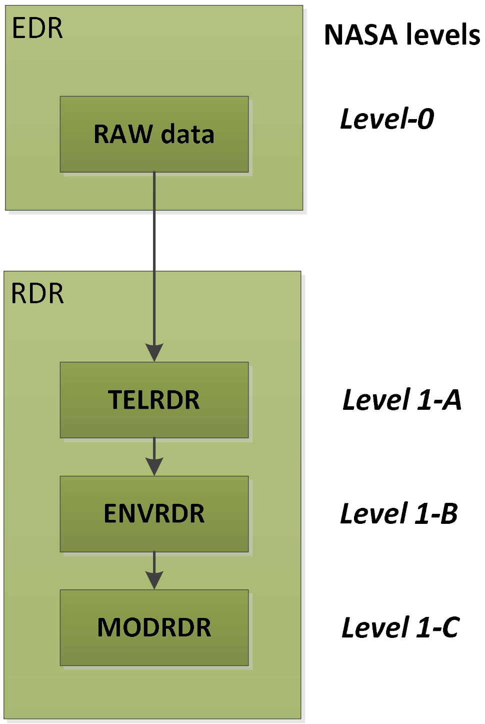

Figure 10: Processing levels of the REMS data.

2.3.1. EDR product

This dataset from REMS contains the raw unprocessed data stored in the REMS internal memory, as received from telemetry. It consists of a collection of tables that include the different sensors readings, as well as engineering data and instrument configuration parameters.

The information included in this dataset needs to be combined using calibration and other ancillary information to get the estimated physical magnitudes, in the next level. The required ancillary information can be found in the RDR archive volumes.

Each REMS EDR has a detached PDS label associated with the REMS data file. The file-naming scheme for the REMS EDR data products is detailed in Figure 11 and it is explained below.

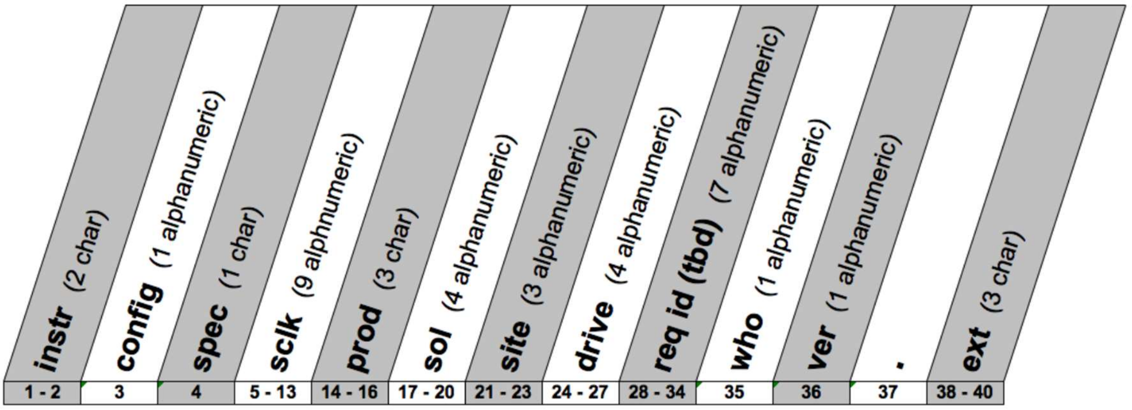

Figure 11: File name scheme of the EDR data products. (Credit: CAB-CSIC-INTA)

- instr: (the first two characters) It corresponds to the Instrument ID. It denotes the source MSL science or engineering instrument that acquired the data. The valid values for the Instrument ID's are: 'RM' – REMS.
- config: (one character) Valid values are: 'E'- Environmental.
- spec: (one character) Special Processing flag, applicable to RDRs on a case-by-case basis.
- sclk: (9 alphanumeric characters) Spacecraft Clock Start Count, in units of seconds. It is generally expected to be the time the data was acquired.
- prod: (3 character) Product type identifier. This field identifies the type of REMS EDR, and has the following rule-of-thumb:
  - ✓ REMS Science and Engineering EDRs (ASCII represented): 'ESE'
  - ✓ Any other type of REMS EDR (binary represented): 'EDR'
- sol: (4 alphanumeric characters) Sol or Mars Solar Day.

- site: (3 alphanumeric characters) Site location count, from the Rover Motion Counter (RMC).
- drive: (4 alphanumeric characters) Drive, position-within-site, location count, from the RMC.
- Seq-id: (7 alphanumeric characters) - Request ID. For REMS, this field indicates the specific type of REMS EDR. The valid values are specified in Table 1, while any leftover spaces will be padded with '_' (underscore). An all-underscores value ('_______') indicates that an EDR was created from REMS data outside the expected range. For our study, we will only use the *Sensor Acquisition* files, ACQ.

Table 1: List of REMS EDR type file, also known as raw data, with their corresponding file name.

Information containing the file	**Name of the file**
ASCII represented EDRs	
Sensor Acquisition	ACQ____
Engineering	ENG____
Humidity Sensor Regeneration Acquisition	HSREG__
Humidity Sensor Defrost Acquisition	HSDEF__
Ground Temperature Sensor Calibration	GTSCAL_
Reset	RESET__
Sleep	SLEEP__
Event Entry	EVENT__
Ground Temperature Sensor Gain Management	GTSGAIN
Error	ERROR__
System parameters	SP_____
Binary represented EDR	
DoBist	DBT____
DownloadErrInfo	DEI____
DumpEeprom	DEE____
DumpFlash	DFL____
DumpRam	DRM____
SystemByte	GET____
UtilTest	UTT____
Parms	PRM____

- who: (1 alphanumeric character) It identifies the institution that generated the product. Valid values **'M'** – MIPL (at JPL).

- ver: (1 alphanumeric character) Version identifier. The version number increments by one whenever an otherwise-identical filename would be produced.
- ext: (2 to 3 alphanumeric characters) Product type extension. Valid values for nominal operations data products:

  'TAB': Science and Engineering EDRs (ASCII represented).

  'DAT': Non-Science and Engineering EDRs (binary represented).

  'LBL': Labels from TAB files (PDS format).

Measurements information from each sensor is kept in the *Sensor Acquisition* file (ACQ), see Table 1. The rest of the files contain mainly configuration and error information about the different sensors. Data sampling is taken at 1Hz maximum, for a baseline operation of 5 minutes every hour. Additional measurements can be taken on an on-demand basis beyond those hourly observations. These additional measurements can also be triggered automatically if *event mode* is active, in which case the REMS computer will decide whether to continue operating after the 5 minutes session by comparing the measurements taken and the expected trend. The objective is to capture any ongoing transitory atmospheric event.

Each file contains one sol of activity time and data is time ordered, with a timestamp reflecting the beginning of each data acquisition.

2.3.2. RDR product

Each REMS RDR has a detached PDS label associated with the REMS data file. The file-naming scheme for the REMS RDR data products is detailed in Figure 11 and it is explained below:

- instr: (the first two characters) It corresponds to the Instrument ID. It denotes the source MSL science or engineering instrument that acquired the data. The valid values for the Instrument ID's are: 'RM' – REMS.
- config: (one character) Valid values are: 'E'- Environmental.
- spec: (one character) Special Processing flag, applicable to RDRs on a case-by-case basis.
- sclk: (9 alphanumeric characters) Spacecraft Clock Start Count, in units of seconds. It shows the time the data was acquired.
- prod: (3 characters) Product type identifier:
  - ✓ Beginning with 'E' - Type of EDR, which is the first order product with no processing applied.
  - ✓ Beginning with 'R' – Type of RDR, except for ADRs.

    - ✓ Valid values for the RDR case are listed below:

TELRDR -> 'RTL'
ENVRDR-> 'RNV'
MODRDR-> 'RMD'
ADR -> 'ADR'

- sol: (4 alphanumeric characters) Sol or Mars Solar Day.
- site: (3 alphanumeric characters) Site location count, from the Rover Motion Counter (RMC).
- drive: (4 alphanumeric characters) Drive (position-within-Site) location count, from the RMC.
- Seq-id: (7 alphanumeric characters) Request ID. For RDRs it consists of underscores '_______'.
- who: (1 alphanumeric character) It identifies the institution that generated the product. Valid values: 'P' – Principal Investigator (REMS).
- ver: (1 alphanumeric character) Version identifier. The version number increments by one whenever an otherwise-identical filename would be produced.
- ext: (2 to 3 alphanumeric characters) Product type extension. Valid values for nominal operations data products:
    - ✓ 'TAB': Science and Engineering EDRs (ASCII represented).
    - ✓ 'LBL': Labels from TAB files (PDS format).

Next, the different RDR provided are explained in detail. An effort has been made to integrate results from all sensors in each RDR, in order to facilitate data analysis. However, the complexity of data processing is not the same for all sensors, so there are a greater number of transformations between RDR types for some sensors as compared to others.

2.3.2.1. TELRDR file

The TELRDR file is the result of the first processing step. It contains data where counts recorded by the instrument have been converted to thermal and electrical values using calibration information. Temperatures for PT1000 sensors are given instead of resistances since the conversion between them is straightforward and temperatures are more helpful.

Each TELRDR product is an ASCII table containing all sensors data, accompanied by a detached PDS label. Each row contains an acquisition session, while the columns contain the sensor values. This makes it easy to load the data products to a spreadsheet or a database. Columns with related information are placed together, in the following order: REMS clock (synced with the Spacecraft Clock (SCLK) with a maximum deviation of 30 s), Local Mean Solar Time (LMST) and Local True Solar Time (LTST), Wind Sensor, Ground Temperature Sensor, Air Temperature Sensor, Ultraviolet Sensor, Humidity Sensor and Pressure Sensor. For more information about time reference, see *Annex 5*. Columns are delimited by commas and are of fixed length. Rows are time ordered and are separated by a carriage return/line feed. Each table contains a sol of measurements. Data may be set to UNK if their value is not known and it will never be (i.e., in case of saturation, or whenever a specific sensor is switched off during acquisition). They may also be set to NULL if their value is not known at the moment of the release of the dataset, but it is expected to be known in a future release.

2.3.2.2. ENVRDR file

The ENVRDR files are the second processing step. At this level, data has been converted from electrical to environmental magnitudes provided by each engineering sensor (e.g., data for each air temperature PT1000 sensor instead of a unique air temperature, or data for each ground temperature sensor thermophile instead of a unique ground temperature). Minimal corrections exist for some sensors to compensate their degradation due to exposure to Martian conditions.

Each REMS ENVRDR product in the dataset is an ASCII table containing all sensors data. Each row contains an acquisition session, and the columns contain the sensor's values.

Columns with related information are placed together, in the following order: time references (REMS clock, LMST and LTST), Wind Sensor, Ground Temperature Sensor, Air Temperature Sensor, Ultraviolet Sensor, Humidity Sensor and Pressure Sensor.

At this level, the Ground Temperature Sensor measurements are corrected by using in-flight calibration data (found in the TELRDR), and the Ultraviolet Sensor responsivity is corrected depending on the Solar Zenith Angle and the estimation of the dust over the photodiodes.

Moreover, a confidence level label for each sensor is included. This label indicates the quality of the data by codifying a string of zeroes and ones. '1' means ‘good condition’ or a '0' means ‘bad condition’. The higher the number of ones, the more reliable the measured magnitude is. The character 'X' may be present in some cases for factors whose value is not known at the moment of the data generation.

In addition to confidence level labels, some sensors include, in another column, an estimation of the uncertainty in the data provided.

Columns are delimited by commas and are of fixed length. Rows are time ordered and are separated by a carriage return/line feed. Each table contains a sol of measurements.

Data may be set to UNK if their value is not known and it will never be (such as saturation, or a specific sensor switched off during acquisition). They may also be set to NULL if their value is not known at the moment of the release of the dataset, but it is expected to be known in a future release. If data from a sensor is set to UNK or NULL, its associated confidence level code will also be set to UNK or NULL.

2.3.2.3. MODRDR file

Each REMS MODRDR product in the dataset is an ASCII table containing all sensors data. Each row contains an acquisition session, and the columns contain the different parameters measured by each sensor.

Columns with related information are placed together, in the following order: time references (REMS clock, LMST and LTST), Wind Sensor, Ground Temperature Sensor, Air Temperature Sensor, Ultraviolet Sensor, Humidity Sensor and Pressure Sensor.

As previously described for other processing levels, REMS clock is synced with the spacecraft clock (SCLK) with a maximum deviation of 30 s.

As explained in the previous level, a confidence level label for each sensor is also included. The same method is used, as previously described, to apply and understand these labels.

In addition to the confidence level label, some sensors include an estimation of the uncertainty, in a new column.

On this level, correction algorithms, specific to each sensor, are applied to present more reliable data.

As in the ENVRDR product, columns are delimited by commas and are of fixed length. Rows are time ordered and are separated by a carriage return/line feed. Each table contains a sol of measurements. Data may be set to UNK if their value is not known and it will never be (in case of saturation, or when a specific sensor is switched off during acquisition). They may also be set to NULL if their value is not known at the moment of the release of the dataset, but it is expected to be known in a future release. If data from a sensor is set to UNK or NULL, its associated confidence level code will also be set to UNK or NULL.

2.3.2.4. ADR file

The ADR contains ancillary data needed in the processing of some sensors' data, and it is obtained from sources external to the REMS instrument. For example, we can have geometry information or estimations of the signal attenuation produced by dust deposited over the Ultraviolet Sensor. This attenuation is calculated by taking images of the sensor by the rover's cameras and by applying an algorithm developed at CAB.

All these processing levels, for each sensor, are public and stored in the PDS.

Focusing on the sensors that are the subject of our study, Table 2 summarizes the different processing levels from ATS and PS, which will be explained, with more details, in Chapter 4 and 5.

Table 2: Data containing in file processing level, from ATS and PS.

NASA levels / REMS data product	**ATS**	**PS**
Level-0 / EDR or raw	Resistance counts digitized.	Pressure Sensor counts digitized, from its 8 channels.
Level 1-A / TELRDR	Temperatures, in kelvin, measured by each of the 5 Pt1000 sensors. Mean of the three thermopiles from the GTS replaces temperature at the base of the rod at Boom 1.	Pressure Sensor capacitances, in picofarad, from the 8 channels.
Level 1-B / ENVRDR	No changes from previous level except for adding the estimated uncertainties columns for each data.	Temperature from both thermocaps (K); Pressure for both barocaps (Pa), and estimated uncertainties.
Level 1-C / MODRDR	A local air temperature (K) is calculated by the m-model [20] for each boom. The cooler temperature is chosen, between both booms, as ambient temperature (K).	Pressure, in pascals, and its estimated uncertainty.

To summarize, in this chapter we have presented the mission, the sensors involved in this study, and we have explained how to obtain physical measurements from the data received from the rover.

In the next chapter, we detail the signal processing methods that will be used to analyze the data and try to characterize relevant information about the thermal processes in Mars.

Chapter 3: Methodology

This chapter is dedicated to the explanation of the theory of the signal processing methods used in this thesis. Three different methods have been mainly used in this thesis and they are going to be detailed below. Singular Spectrum Analysis is the method used for detection of given environmental processes, while wavelets and the Hilbert-Huang Transform are applied for denoising.

3.1. Singular Spectrum Analysis for environmental processes detection.

The Singular Spectrum Analysis is a recent method. The first mentions seem to be around 1989, by Rober Vautard [22]. The SSA is a technique for decomposing a time series into a sum of components, each of which has a meaningful physical interpretation. The method is based on the Singular Value Decomposition (SVD)[23] and the subspace projection and it can be applied to analyze any measured time series with a potential structure. The procedure mainly depends upon two important parameters: the window length M, the main parameter to determine the dimension of the trajectory matrix, and the number of eigenvalues/eigenvectors [24].

This technique has been intended for the extraction of information from short and noisy time series, and then provide insight into the unknown or only partially known dynamics of the underlying system that generate the series. Additionally, the SSA method, apart from decomposing the series into subcomponents, may reconstruct the signal by leaving the noise component behind.

It is a technique that works well with arbitrary statistical processes, whether linear or nonlinear, stationary or non-stationary, Gaussian or non- Gaussian [25]. This technique has been previously used for the analysis of climate time series [26]. Moreover, it has been also for Viking Lander pressure data [27].

Next, a detailed description of the method is explained [28]:

1. Let's consider two time series, *X, sol,*

$$X = (x_1, x_2, .., x_n) \qquad (1)$$
$$sol = (sol_1, sol_2, \ldots, sol_n),$$

where *sol* contains the mission sols that measured 24 observations, *X* shows their respective measured values in these sols, and *n* is the number of sols for which REMS collected a full diurnal cycle, 24 observations

2. The signal *X* is detrended by removing its mean x_{mean}. The resulting sequence of values, $X - x_{mean}$, has gaps for missing sols where there was not a full diurnal cycle of magnitudes measured.

$$X - x_{mean} = \big((x_1 - x_{mean}), (x_2 - x_{mean}), .., (x_n - x_{mean})\big). \qquad (2)$$

3. To solve the gaps problem in the vector *X*, an expanded signal is defined, $\ddot{X}$, where a 0 is added at the positions of sols where there are no 24 observations, while maintaining the main frequency characteristics [29],

$$\ddot{X} = (\ddot{x}_1, \ddot{x}_2 \ldots \ddot{x}_i, \ldots, \ddot{x}_N), \qquad \ddot{x}_i = \begin{cases} 0\ if\ SOL_i \not\subset sol \\ (x_j - x_{mean})\ if\ SOL_i \subset sol, \end{cases} \qquad (3)$$

where *j* is the position where SOL_i is in *sol*, SOL_i is a random sol from the mission and *N* is the dimension of the new vector $\ddot{X}$, corresponding to the last sol considered in the study.

4. Create the trajectory matrix *Y* with dimensions $(N - M + 1)\ x\ M$, where *M* is the temporal window covering. This converts from a one-dimensional time series $\ddot{X}$ into a multidimensional series, *Y*, with columns defined as:

$$\mathrm{Y} = (\ddot{x}_1, \ddot{x}_2 \ldots \ddot{x}_M), (\ddot{x}_2 \ldots, \ddot{x}_{M+1}), \ldots, (\ddot{x}_{N-M+1} \ldots, \ddot{x}_N). \qquad (4)$$

The result of this step is the trajectory matrix $Y = [Y_1,\ Y_2, \ldots Y_{N-M+1}]$,

$$Y = \begin{pmatrix} \ddot{x}_1 & \ddot{x}_2 & \ddot{x}_3 & \cdots & \ddot{x}_M \\ \ddot{x}_2 & \ddot{x}_3 & \ddot{x}_4 & \cdots & \ddot{x}_{M+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \ddot{x}_{N-M+1} & \ddot{x}_{N-M} & \ddot{x}_{\dots} & \cdots & \ddot{x}_N \end{pmatrix}. \tag{5}$$

The range of temporal window length, M, should be approximately $2 \leq M \leq N/2$. If M is close to $N/2$, it can make the trajectory matrix overlap its rows and columns. If the window length M is relatively small, it can cause the trajectory matrix to produce improper decomposition of mixed signal. It is advisable to take M proportional to the period of the selected time series, $\ddot{X}$, which may have a periodic component with an integer period [24].

5. Compute the matrix $S = YY^T$ (dimension $M \times M$) in order to apply the Singular Value Decomposition, SVD [23].

6. Compute the eigenvalues and the eigenvectors of the Matrix S. In this case, the cross correlation between Y and Y^T is calculated so that its eigenvalues are normalized and each eigenvalue, e_i, measures the ratio of variability associated with its eigenvector, v_i

$$v_i \; = \; (v_{i,1}, v_{i,2}, \dots, v_{i,M}). \tag{6}$$

7. There are $N - M + 1$ principal components, pc_i, each one associated with an eigenvalue, e_i, obtained as the projection of the matrix Y into the subspace associated with its eigenvector, v_i,

$$pc_i \; = \; Y^T.v_i. \tag{7}$$

8. The expanded signal, $\ddot{X}$, can be reconstructed using all the principal components, pc_i. Each component provides the contribution, r_k, to the reconstruction of $\ddot{X}$ by multiplying the component by the eigenvector as

$$r_{k,1} = pc_{k,1}.v_{k,1}$$
$$r_{k,2} = \frac{pc_{k,2}.v_{k,1} + pc_{k,1}.v_{k,2}}{2}$$

$$r_{k,j} = \frac{pc_{k,j}.v_{k,1} + pc_{k,j-1}.v_{k,2} + \cdots + pc_{k,1}.v_{k,j}}{j} \qquad j = 3, \ldots\ldots M-1 \tag{8}$$

$$\ldots\ldots.$$

$$r_{k,N-1} = \frac{pc_{k,N}.v_{k,M-1} + pc_{k,N-1}.v_{k,M}}{2}$$

$$r_{k,N} = pc_{k,N}.v_{k,M}\ .$$

The reconstruction is made with the principal components that explain 95 % of the variability of the original signal. This percentage of reconstruction has been decided by heuristic and empirical criteria. To reach the final result, we covered the data gaps and thereby reduced their influence at each transition from missing data to existing data by comparing the reconstructed signal with the zeroes in the gaps of the original $\ddot{X}$. Those zeroes were replaced with the $r_{k,i}$ to repeat the previous analysis, extracting a new set of r_k. It is then compared the new r_k values with the previous set, repeating the iteration until the difference between one reconstruction in the gaps and the next reconstruction agreed within a standard deviation of 1 %.

This general method is going be adapted and applied to both sensors data, PS and ATS. In *Annex 1,* the code to apply SSA to both sensors data is shown.

3.2. Discrete Wavelet Transform for denoising

From an historical point of view, wavelet analysis is a new method, though its theoretical mathematical foundations are subsequent to the research presented by Joseph Fourier, in the nineteenth century [30]. Fourier laid the foundations with his theories of frequency analysis, which proved to be tremendously important and influential. The attention of researchers gradually turned from frequency-based analysis to scale-based analysis when it started to become clear that an approach measuring average fluctuations at different scales might prove less sensitive to noise. The first recorded mention of what we now call a “wavelet” seems to be in 1909, when Alfred Haar, a Hungarian mathematician-scientist, introduced the idea of HAAR orthogonal system through his doctoral thesis at University of Gottingen, Germany. The concept of wavelets in its present theoretical form was first proposed by Jean Morlet and the team at the Marseille Theoretical Physics Center working under Alex Grossmann in France, in 1982 [31], [32]. The methods of wavelet analysis have been developed mainly by Y. Meyer and his team, who have ensured the dissemination of the methods [33]. The main algorithm dates back to the work of Stephane Mallat, in 1988 [34]. Since then, research on wavelets domain has become world-renowned. A lot of mathematical papers and practical researches are published every month. Such research is particularly active by the work of scientists such as Ingrid Daubechies, Ronald Coifman, and Victor Wickerhauser.

Since they have become internationally known, wavelets have been developed independently in many different fields such as mathematics, quantum physics, electrical engineering, or seismic geology. Interchanges between these fields during the last ten years have led to many new wavelet applications such as signal denoising, image compression, human vision, radar, and earthquake prediction.

From a methodological point of view, the wavelets are functions that divide data into different frequency components, and then work with each component with a resolution matched to its scale. They allow for an analysis adapted to the data scale. Wavelets are functions that satisfy certain mathematical requirements and are used in representing data and multiple functions. Wavelet algorithms process data at different scales or resolutions and this is the novelty with respect to Fourier. Wavelets are well-suited for estimating data with sharp discontinuities, and this advantage over the Fourier series is very useful for our specific data.

Wavelet transforms are broadly divided into two classes: the Continuous Wavelet Transform (CWT), which is a formal tool that provides a complete representation of a signal by letting the translation and scale parameter of the wavelets vary continuously, and the Discrete Wavelet Transform (DWT), which is any wavelet transform for which the wavelets are discretely sampled. There is a set of families, the mother wavelets, specific to each class, which have their own characteristics. When a specific wavelet shape is needed because it is physically related to the data to study, DWT is the wavelet alternative more appropriate to use. In the next paragraphs, the data processing flow based on the Discrete Wavelet Transform [33] (DWT) for denoising is explained. The specific configurations of the method will be detailed in the chapter devoted to its applications, Chapter 5:

1. The DWT is applied to the data.

   Wavelet analysis makes use of short-duration waveforms, $\psi[n]$, with zero mean and a sharp decay to zero at both ends. These short-duration waveforms are scaled and shifted to set the optimum time-frequency resolution. We use the definition and formulation of Mallat [35], where the discrete wavelet scaled by 2^j is expressed as

$$\psi_j[n] = \frac{1}{\sqrt{2^j}} \cdot \psi\left[\frac{n}{2^j}\right], \tag{9}$$

where j is the number of decomposition levels and $\psi[n]$ corresponds to a wavelet family that will be later detailed. The number of decomposition levels, j, will vary depending on the number of samples. The DWT coefficients of the signal $x[n]$ with respect to the wavelet function $\psi[n]$ can be written as

$$X_j[m] = \sum_{n=0}^{N-1} x[n] \cdot \psi_j^*[n-m], \tag{10}$$

where $\psi_j^*[n]$ is the complex conjugate of the scaled wavelet family, m is the translation parameter, and N the number of samples of the noisy signal. In similar applications [36], the choice of the mother wavelet is mainly based on visual inspection and on the correlation between the original signal and the wavelet-denoised signal.

It is worth noting that equation (10) can be expressed as a circular convolution, so that we can resort to the Fast Fourier Transform when making the corresponding computation. This approach just requires $O(N \log_2 N)$ operations for each scale, resulting in high computational efficiency.

2. Thresholding of the sub band signals with a threshold θ.

   After having decomposed the original signal, a thresholding step is applied to the coefficients of the wavelet decomposition. Its purpose is finding out which part of them should be qualified as noise, and should be consequently removed. For this purpose, the universal threshold from Donoho and Johnstone [37] has been used. Said threshold is calculated as [37]

$$\theta = \sigma \cdot \sqrt{2 \cdot \log N}, \tag{11}$$

where σ is the estimated standard deviation of the noise, which is calculated using the so-called Median Absolute Deviation (MAD) from the higher signal frequency decomposition level of the wavelet transform, $X_1[m]$, as [37]

$$\sigma = \frac{MAD}{0.6745}. \tag{12}$$

The MAD is defined as

$$MAD = med\,\{|X_1[m] - med\,\{X_1[m]\}|\}, \tag{13}$$

where *med*{} is the median value. These expressions take into account that the value of the noise standard deviation σ is often estimated from the median value of the $X_j[m]$ wavelet coefficients belonging to the first level of signal decomposition $X_1[m]$ [38], corresponding to the finest level of decomposition.

Once the value of θ is obtained, the thresholding process is performed. In the application chapter, Chapter 5, the thresholding process will be specified according to the nature of our measurements.

3. Reconstruction of the filtered output signal.

   Once the thresholding has been applied to $X_j[m]$, the output coefficients $\hat{X}_j[m]$ are obtained. They are used to produce the reconstructed signal $y^w[n]$ by means of the inverse DWT,

$$y^w\,[n] = \sum_{m=0}^{N-1}\sum_{j=1}^{J} \hat{X}_j[m] \cdot \psi_j[n-m], \tag{14}$$

   where J is the number of decomposition levels.

4. By subtracting the reconstructed signal from the original one $x[n]$, we obtain an estimation of the perturbing noise $\epsilon^w[n]$ as

$$\epsilon^w\,[n] = x[n] - y^w[n]. \tag{15}$$

3.2.1. Wavelets families considered

In this section, we present the 3 wavelets that were initially preselected in a process intended to find the one that could best fit our data. These 3 wavelet families were chosen among all possibilities taking into account the shape of our signals.

3.2.1.1. Daubechies

Ingrid Daubechies, one of the brightest people in the world of wavelet research, invented what are called compactly supported orthonormal wavelets [39], which made discrete wavelet analysis practicable. The specific Daubechies wavelet family is denoted by dbN, where N is the order, and db the surname of the wavelet. Figure 12 shows the different forms of this family:

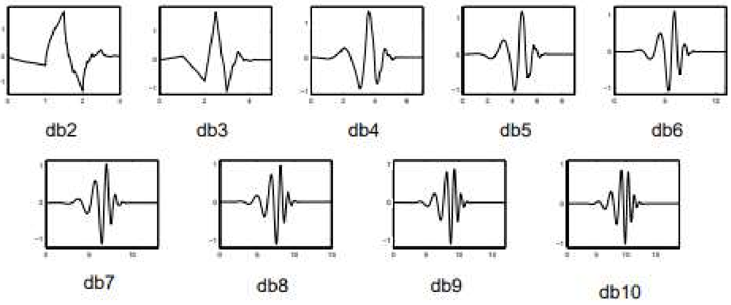

Figure 12: Daubechies family. (Credit: MATLAB toolbox)

For our purpose, we performed tests with the higher order wavelets, due to the fact that they look more similar to the signals under analysis.

3.2.1.2. Coiflets

Coiflets are discrete wavelets designed by Ingrid Daubechies [40], in order to have scaling functions with vanishing moments. Each mother wavelet in the family is near symmetric, and have 2*N* moments equal to 0, being *N* being the order. Figure 13 shows the different forms of this family. We also performed tests with the higher order wavelets, due to the fact that they look more similar to the signals under analysis.

Figure 13: Coiflets family. (Credit: MATLAB toolbox)

3.2.1.3. Symlets

The symlets group are nearly symmetrical wavelets proposed by Daubechies as modifications to the Daubechies family [39]. The properties of both wavelet families are very similar. Figure 14 shows the different forms of this family. We also performed tests with the higher order wavelets, due to the fact that they look more similar to the signals under analysis.

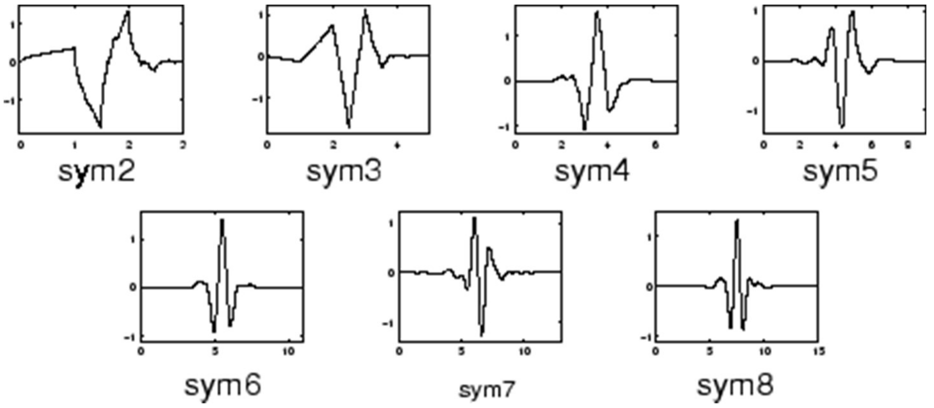

Figure 14: Symlet family. (Credit: MATLAB toolbox)

3.3. Hilbert-Huang Transform for denoising

The Hilbert-Huang Transform (HHT), was proposed by Norden E. Huang and his team in 1996 [41]. It is used in multiple areas such as biomedical applications, neuroscience, image processing, speech recognition, among others. HHT is able to extract the frequency components from possibly nonlinear and nonstationary intermittent signals.

HHT is the result of mixing the Empirical Mode Decomposition (EMD) [41], [42], a fully data-driven approach, with which any complicated dataset can be decomposed into a finite and often small number of functions that admit well-behaved Hilbert transforms, and the Hilbert Spectral Analysis (HSA)[41], which is a signal analysis method applying the Hilbert transform to compute the instantaneous frequency of a signal.

The HHT uses the EMD method to decompose a signal into so-called Intrinsic Mode Functions (IMF), and applies the HSA method to the IMFs to obtain instantaneous frequency data. Since the signal is decomposed in time domain and the length of the IMFs is the same as the original signal, the HHT preserves the characteristics of the varying frequency. This is an important advantage of HHT since real-world signal usually has multiple causes happening in different time intervals.

The denoising process based on the Hilbert-Huang Transform method, HHT, [43] is detailed in the following, according to Boudraa [44]. The method proceeds as follows:

1. The EMD is applied to the data. By using this method, a noisy signal is adaptively decomposed into oscillatory modes with variable amplitudes, the IMFs, plus a residual signal. Each component is defined as an intrinsic mode function (IMF) satisfying the following requirements:
    a. In the whole dataset, the number of extrema and the number of zero crossings must be either equal or differ at most by one.
    b. At any data point, the mean value of the envelope defined using the local maxima and the envelope defined using the local minima is zero.

2. The procedure required to extract an IMF is called *sifting*, and is defined by the following steps:
    a. Consider a dataset *x*[*n*].
    b. Extract the local maximum and local minimum from the signal.
    c. Compute upper and lower envelope.
    d. Compute the envelopes mean. Figure 15 shows an example from the envelope computation to facilitate its comprehension.
    e. Calculate the stop criterion to define the IMF:

$$SD = \sum_{n=0}^{N} \left[\frac{\left| IMF_{j,i-1}[n] - IMF_{j,i}[n] \right|^2}{\left(IMF_{j,i-1}[n] \right)^2} \right], \qquad (16)$$

where *N* is the length of *x*[*n*].

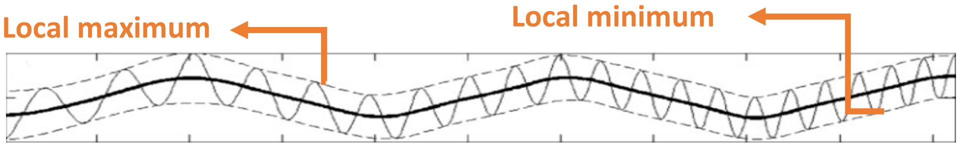

Figure 15: Local maximum and local minimum of a random signal.

f. Find an IMF, following the stop criterion. The difference between the original signal $x[n]$ and the IMF becomes the new signal, and a subsequent sifting process is performed to find the next IMF.

$$x[n] - c_1 = r_1[\mathrm{n}], \tag{17}$$

where c_1 is the first IMF found and r_1 the residual signal, which becomes the new signal.

The sifting process has to be repeated as many times as required to make the extracted signal satisfy the definition of an IMF, fulfilling the two requirements mentioned above. The sifting process can be stopped when the residual signal, $r_i[n]$, becomes a monotonic function from which no more IMF can be extracted.

3. The decomposition of the signal in M−empirical modes can be written as

$$y[n] = \sum_{m=1}^{M} c_m[n] + r[n], \tag{18}$$

where $c_m[n]$ is the IMF of the m-th decomposition level, and $r[n]$ is the residual signal. The signal $y[n]$ is the reconstruction after applying the EMD to the input data $x[n]$. Under ideal conditions, without further processing of the IMF values, the reconstructed signal should be equal to the input signal, i.e., $y[n] = x[n]$.

4. Apply the Consecutive Mean Square Error (CMSE). This step is mainly based on the idea that the main part of the meaningful signal structure has to be found in the last IMFs, which represent the lower frequencies, while the noise is usually associated to the first IMFs, which represent the higher frequencies. Accordingly, the denoising process consists in the reconstruction of the signal after discarding the initial IMF levels. Therefore,

$$y_k[n] = \sum_{m=k}^{M} c_m[n] + r[n], \tag{19}$$

is the reconstructed signal taking into account just the last $M - k + 1$ levels. We consider the CMSE as a measure of the distortion in the reconstructed signal, according to [44].

The CMSE measures the squared Euclidean distance between two consecutive reconstructions of the signal, and it is defined as

$$CSME(y_k, y_{k+1}) = \frac{1}{N}\sum_{n=0}^{N-1} |y_k\,[n] - \,y_{k+1}\,[n]|^2 = \\ = \frac{1}{N}\sum_{n=0}^{N-1} |c_{k\,[n]}|^2. \tag{20}$$

5. Apply the threshold. Apply a threshold to the lowest IMF levels. In order to explicitly filter the additive Gaussian noise, the universal threshold θ is applied here. A different θ has to be calculated for each decomposition level, since noise is found in all levels of decomposition. If we denote the threshold for the *m*-th level as θ_m, where *m* = 0, …, *j*,

$$\theta_m = \sigma_m \cdot \sqrt{2\ \cdot log\ N}\,, \tag{21}$$

where *N* is the length of the noisy signal, and σ_m the noise standard deviation of the *m*-th IMF. The estimated value for σ_m is [45], [46]:

$$\sigma_m = \frac{MAD_m}{0.6745}, \tag{22}$$

where *MADm* is the absolute median deviation of the m-th IMF, calculated as

$$MAD_m \ = \ med\ \{|c_m[n] - \ med\ \{c_m[n]\}|\}. \tag{23}$$

The index *j* that minimizes the CMSE allows to determine which IMF level represents the limit between the part of the signal where the noise can be considered negligible, and the part where the noise is dominant [46]. The index *j* is given by calculating

$$j = argmin_{1\leq k\leq M-1}\big(CMSE(y_k, y_{k+1})\big). \tag{24}$$

Before reconstruction, a soft thresholding method is applied as in [47].

6. Reconstruction of the filtered output signal. After applying the thresholding to the corresponding levels, the reconstructed signal $y^h[n]$ is obtained by adding up the thresholded IMFs, the non-thresholded IMFs and the residual signal

$$y^h\,[n] = \sum_{m=0}^{j} \hat{c}_m\,[n] + \sum_{m=j+1}^{M} c_m\,[n] + r[n]. \tag{25}$$

7. By subtracting the reconstructed signal from the original one, $x[n]$, we obtain an estimation of the perturbing noise $\epsilon^h\,[n]$ as

$$\epsilon^h\,[n] = x[n] - y^h[n]. \tag{26}$$

In this chapter, we have detailed the algorithms used in the thesis and the functionality given to them.

Chapters 4 and 5 explain the application of these methods to our data.

Chapter 4: Processing and analysis of the Pressure Sensor data

After having reviewed the methodology chosen for our data analytical purposes, this chapter focuses on the application of the required methods to the pressure data.

The Pressure Sensor data has been chosen as a target for processing and analysis because of the relevance of this magnitude in meteorology, which has been studied in Mars for years [48][49]. With all the previous studies done on Mars pressure and since pressure gives substantial information about changes in the planet atmosphere, we consider that extracting information from the REMS pressure data can help in the detection of Mars environmental processes.

Pressure is a very stable magnitude because it is a measurement of the force exerted by the atmosphere on the surface of a planet. So, its value is not characterized by sudden changes, as happens with the temperature, a fact that will be discussed in the next chapter. Because the pressure measurements are not much affected by noise, we have chosen not to apply a denoising method.

After reading bibliography related to the subject and doing an intensive study of different algorithms, such as Kalman [50], the Hilbert Spectral Analysis (HSA) [51], or Singular Spectrum Analysis (SSA) [52], this last one has been chosen as the appropriate method to decompose the pressure data and extract information relevant to characterize certain atmospheric processes.

Since the pressure has been previously studied to understand better the behavior of the dust storms [53], our study is based on the possibility to analyze these data to extract possible alarms that warn of the onset of the dust storms, what we could name as storm harbingers or precursors.

The next section is devoted to explaining the whole processing of the pressure data.

4.1. From counts to pressure in pascals

As explained in the Pressure Sensor section, in Chapter 2, much of the information about the Pressure Sensor is confidential. It is for this reason that the description of its processing is limited. Although the processing method is confidential, we have sufficient assurance of the robustness and reliability of this transforming algorithm, from counts to pascals, to be able to work with the data resulting from the application of such algorithms. The public information about the retrieval process which transforms counts (dimensionless values logged by the flight electronics) to pascals (final physical magnitude), described in Table 2, is detailed below:

1. Jet Propulsion Laboratory, JPL, through its pipelines provides the data received from Mars. Pressure Sensor measurements are received in a raw format from each of the 8 channels plus a pressure estimation channel, the latter could be used for event detection. Each channel has 3 bytes.

2. The next processing level, the REMS TELRDR dataset (see Table 2), contains processed REMS data where the counts given by the analog/digital converters from the instrument sensors have been converted to their corresponding values, taking into account calibration information, gains and other possible electrical altering factors, such as the ASIC temperature. This level contains Pressure Sensor capacities, in picofarad, for each of the 8 channels.

3. The next level contains the processing of the ENVRDRs environmental magnitudes with minimal corrections, mainly based on the degradation of the sensors. This level contains the temperature from both thermocaps, in kelvin, and pressure for both barocaps, in pascals, as well as their estimated uncertainties.

4. The last level contains the MODRDR product. At this processing level, a drift correction is adjusted by the REMS pressure team and the confident level is added. This level contains the pressure values, in pascals.

4.2. Application of the SSA to the Pressure Sensor data

In this section, we are going to explain how the SSA has been applied to the MODRDR of the pressure data, as well as the motivation behind. Our main goal behind the application of this algorithm is the detection of environmental processes in Gale. Digging into meteorological events on Mars, we have found out that dust storms dominate much of the research done about the environmental processes on Mars. There are many researches done about the pressure during the dust storms periods. It is believed that the atmospheric circulation patterns are responsible for the growth of the dust storm but their exact trigger and prediction remain a mystery [28]. Motivated by these researchers, the idea arose to search how the pressure behaves before the dust storms arrives, since there were indications in the pressure data that could be precursors of storms. Data used for this study are the most exhaustive data, in terms of accuracy and period of time measured.

As it is shown in Figure 10, data are presented in different processing levels. For our purpose, the last and more elaborated level has been chosen as input data, the MODRDR file. This type of file contains the pressure data in pascals, in its column 38. The highest level of processing has been chosen because it presents the data with the highest level of correctness and reliability and the algorithms used to obtain them have been widely tested.

By using the pressure data from the MODRDR product, we have created an intermediate file, the pressure tide file, containing the decomposition of the pressure per sol. For this purpose, we have used a standard method, the least-square technique, consisting on the decomposition of the temporal evolution of pressure signals into a harmonic series [28], represented in equation (27). The main goal of doing this preprocessing algorithm is to decompose each observed pressure cycle (one cycle per sol) into the mean pressure, its *diurnal*, *semidiurnal*, *terdiurnal*, *tetradiurnal* components, etc., containing each decomposition the pressure tide amplitudes and the pressure tide periods. The pressure tide amplitude indicates the force with which the tide arrives and the phase indicates when the tide arrives.

To decompose the pressure cycle observed over one sol into frequencies, a least-squares fitting technique is used on sols where at least 22 REMS observation windows of 5 minutes are available. If the number of observations is lower than 22 during one sol, said sol is not taken into account. The observed pressure, *p*, is fitted using the expression:

$$p = p_0 + \sum_{i=1}^{10} p_i \cdot sin\left[2\pi\left(\frac{i \cdot LTST}{24} + \varphi_i\right)\right], \tag{27}$$

where p_0 represents the mean pressure per sol, p_i are the amplitudes of the different pressure tide modes, φ_i is their phase, and LTST is Local True Solar Time in hour fraction. Each p_i is associated with a frequency corresponding to the different pressure tide modes: *diurnal*, *semidiurnal*, *terdiurnal*, *tetradiurnal, etc.* Each mode has its physical influence on the atmosphere, although studying this in detail is not our task in this thesis. In our study, we are going to use the diurnal mode, p_1. The diurnal iteration mode indicates how the diurnal tide changes over time.

More information about this intermediate file is presented in *Annex 1*, while *Annex 4* contains an extract of the tide file generated.

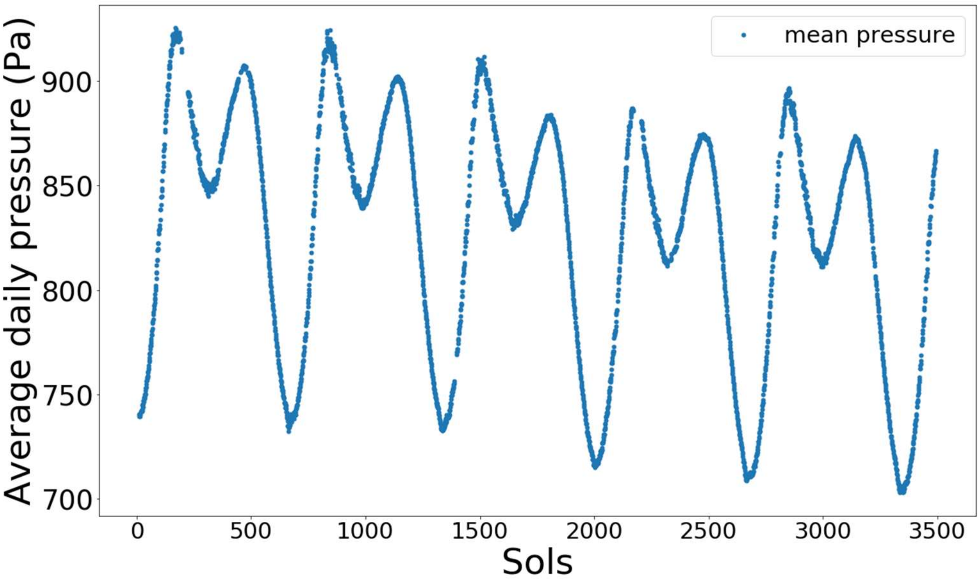

Figure 16: Mean pressure, p_0, per Martian day during the whole mission. Values obtained from equation 22.

This intermediate file provides the input values to the SSA. It contains all the pressure modes.

Figure 16 shows the mean pressure mode, p_0, calculated per Martian day, during the whole Curiosity mission. The advantages of the SSA detrending are its ability to capture the contribution of interannual changes in pressure caused by changes in altitude from a moving rover, as it can be seen in Figure 16, which the harmonic analyses cannot model without additional rover altitude information.

Given the contribution by *Tillman* [53], who noted that the diurnal pressure tide, p_1, changed when dust storms took place, it would be interesting to put the focus on this specific tide. Figure 17 represents the *diurnal* tide pressure, p_1 obtained using the decomposition method explained in equation (27) with pressure data. This figure also shows the opacity, in orange. This information, provided by the MastCam team, is obtained by calculating the radiance of images taken directly of the solar disk. This opacity informs us when a dust storm is occurring, although it cannot warn us before the storm arrives.

Because of the information contained in p_1, we have chosen these values to apply the Singular Spectrum Analysis.

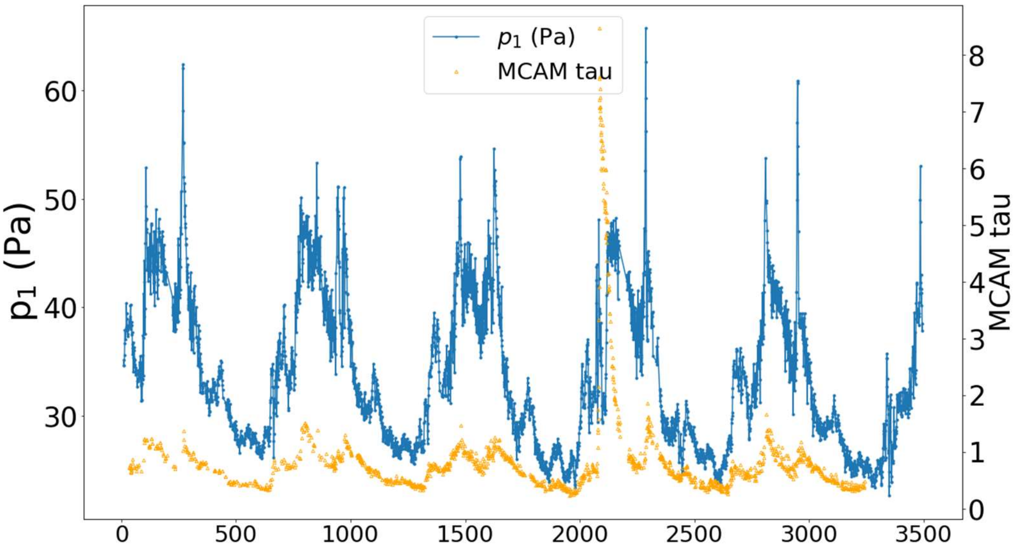

Figure 17: Diurnal pressure, p_1, during the whole mission. Values obtained from equation 22.

Next, the application of the SSA to the diurnal pressure data, p_1, is detailed:

1. Let's consider two time series, a pressure signal, *p*, and a sol series, *sol,*

$$p = (p_{0_1}, p_{0_2}, p_{0_3}, \dots, p_{0_n}) \quad (28)$$
$$sol = (sol_1, sol_2, sol_3, \dots, sol_n),$$

where *sol* contains the mission sols that measured 24 observations, *p* shows their respective measured values of pressure in these sols, and *n* is the number of sols for which REMS collected a full diurnal cycle (24 observations).

2. The pressure signal *p* is detrended by removing its mean value p_{mean}

$$p_{mean} = n^{-1} \sum_{i=1}^{n} p_{0\_i} , \tag{29}$$

where the p_{0_i} is the average pressure found, from sol *i*, in equation (27).

3. The resulting sequence of values, $p - p_{mean}$, has gaps for missing sols where there was not a full diurnal cycle of magnitudes measured.

$$p - p_{mean} = (p_1, p_2 \dots, p_n). \tag{30}$$

4. To solve the gaps problem, an expanded signal is defined, $\ddot{P}$, where a 0 is added at the positions of sols where there are no 24 observations, while maintaining the main frequency characteristics [29],

$$\ddot{P} = (P_1, P_2, \dots, P_i \dots, P_N), \;\; P_i = \begin{cases} 0 \; if \; SOL_i \not\subset sol \\ p_j \; if \; SOL_i \subset sol, \end{cases} \tag{31}$$

where *j* is the position where SOL_i is in *sol*, SOL_i is a random sol from the mission and *N* is the dimension of the new vector $\ddot{P}$, corresponding to the last sol considered in the study.

5. Create the trajectory matrix *Y* with dimensions (*N* − *M* + 1) x *M*, where *M* is the temporal window covering. The choice of M has been detailed in Chapter 3. This converts a one-dimensional pressure series $\ddot{P}$ into a multidimensional pressure series $Y = (P_1 \dots P_M), (P_2 \dots P_{M+1}), \dots, (P_{N-M+1} \dots P_N)$.

   The result of this step is the trajectory matrix $Y = [Y_1, \;\; Y_2, \dots Y_{N-M+1}]$,

$$Y = \begin{pmatrix} P_1 & P_2 & P_3 & \dots & P_M \\ P_2 & P_3 & P_4 & \dots & P_{M+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{N-M+1} & P_{N-M} & P_{\dots} & \dots & P_N \end{pmatrix}. \tag{32}$$

6. Compute the matrix $S = YY^T$ (dimension M x M) in order to apply the Singular Value Decomposition (SVD) [23].

7. Compute the eigenvalues and the eigenvectors of the Matrix S. In this case, the cross correlation between Y and Y^T is calculated so that its eigenvalues are normalized and each eigenvalue, e_i, measures the ratio of variability associated with its eigenvector, v_i

$$v_i = (v_{i,1}, v_{i,2}, \ldots., v_{i,M}). \tag{33}$$

8. There are $N - M + 1$ principal components, pc_i, each one associated with an eigenvalue, e_i, obtained as the projection of the matrix Y into the subspace associated with its eigenvector, v_i,

$$pc_i = Y^T . v_i. \tag{34}$$

9. The expanded signal, $\ddot{P}$, can be reconstructed using all the principal components, pc_i. Each component provides the contribution, r_k, to the reconstruction of $\ddot{P}$ by multiplying the component by the eigenvector as

$$\begin{gathered}
r_{k,1} = pc_{k,1}.v_{k,1} \\
r_{k,2} = \frac{pc_{k,2}.v_{k,1} + pc_{k,1}.v_{k,2}}{2} \\
r_{k,j} = \frac{pc_{k,j}.v_{k,1} + pc_{k,j-1}.v_{k,2} + \cdots + pc_{k,1}.v_{k,j}}{j} \qquad j = 3, \ldots..M-1 \\
\ldots\ldots. \\
r_{k,N-1} = \frac{pc_{k,N}.v_{k,M-1} + pc_{k,N-1}.v_{k,M}}{2} \\
r_{k,N} = pc_{k,N}.v_{k,M}\,.
\end{gathered} \tag{35}$$

The reconstruction is made with the principal components that explain 95 % of the variability. This percentage of reconstruction has been decided by heuristic and empirical criteria. This 95 % of variability corresponds to the first 8 principal components. To reach the final result, we covered the data gaps and thereby reduced their influence from missing data to existing data at each transition by comparing the reconstructed signal with the zeroes in the gaps of the original $\ddot{P}$. Those zeroes are replaced with the $r_{k,i}$ and the previous analysis is repeated, extracting a new set of r_k. It is then compared the new r_k values with the previous set, repeating the iteration until the difference between one reconstruction over the gaps and the next reconstruction agreed within a standard deviation of 1 %.

4.3. Results and discussion

In this section, we show the results of applying the SSA to the diurnal pressure data, p_1, as explained in the previous section.

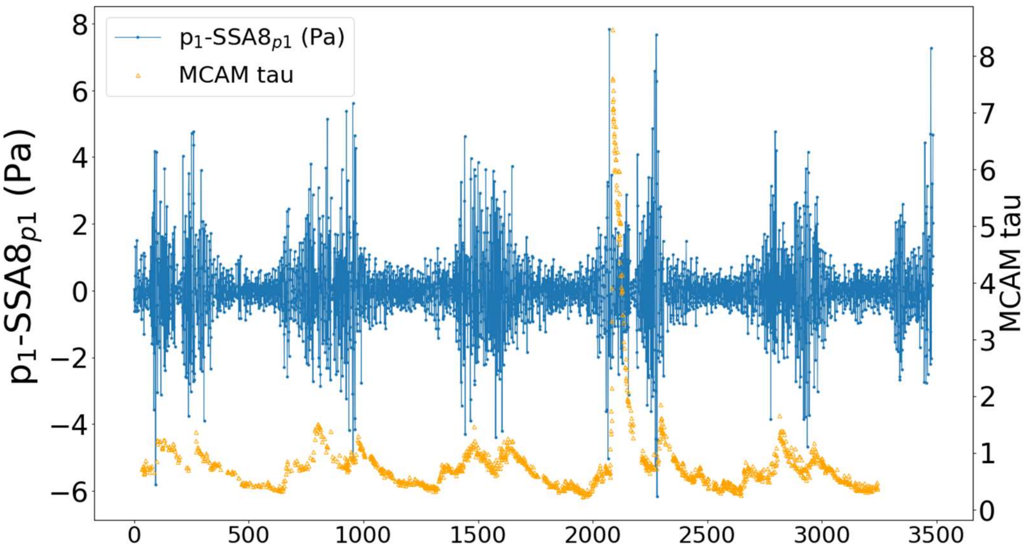

Figure 18: Diurnal pressure minus reconstruction from SSA.

Figure 18 represents the result of the subtraction between p_1, represented in blue in Figure 17, and the reconstruction of p_1 by using the first 8 eigenvectors from SSA decomposition, which represent the 95 % of the original signal. With this operation, we find sudden jumps in the signal coming earlier than the opacity peaks from the MastCam camera, represent in orange in Figure 18.

Figure 19, Figure 20, Figure 21 and Figure 22 are enlarged subperiods of the sols from Figure 18. They mainly zoom in dust storm periods. During a dust storm period, there are several subperiods where it reaches maximum opacity and others where its intensity decreases. Figure 19 shows the beginning of the third storm observed since the rover Curiosity landed in Gale crater, from sol 1350 to sol 1550. Figure 20 shows the beginning of the fourth storm, from sol 2000 to sol 2200, Figure 21 shows the second peak of the fourth dust storm, from sol 2200 to sol 2400 and Figure 22 shows the beginning of the fifth dust storm period, from sol 2700 to sol 2800. Different examples from different periods observed by the rover Curiosity have been used to support the finding we wish to demonstrate. As it can be seen in these figures, sudden jumps in the pressure are prior to MastCam opacity jumps. Studying these figures, and noticing that the behavior is repeated in each storm, we can confirm that there are changes in pressure, extracted and discovered by the SSA algorithm, prior to the storm, which warn us of its onset. Therefore, there are precursors that warn us of the dust storm. This information, although found in the pressure data, cannot be visualized in the pressure data itself. It has been discovered thanks to the analysis based on the SSA processing [28].

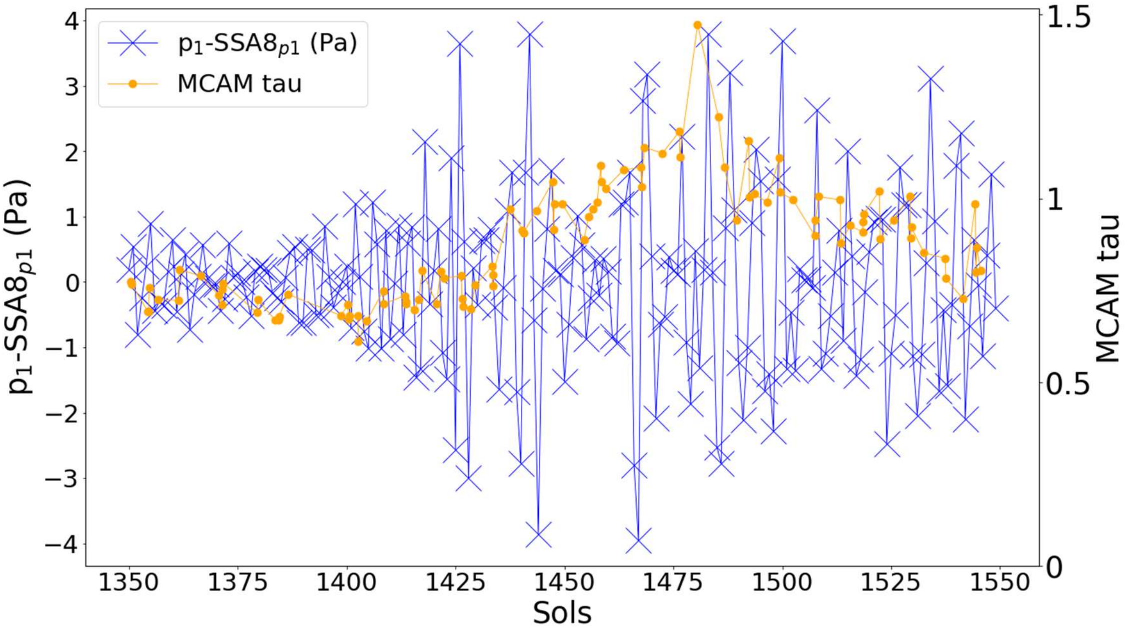

Figure 19: Zoom from Figure 13 from sol 1350 to sol 1550.

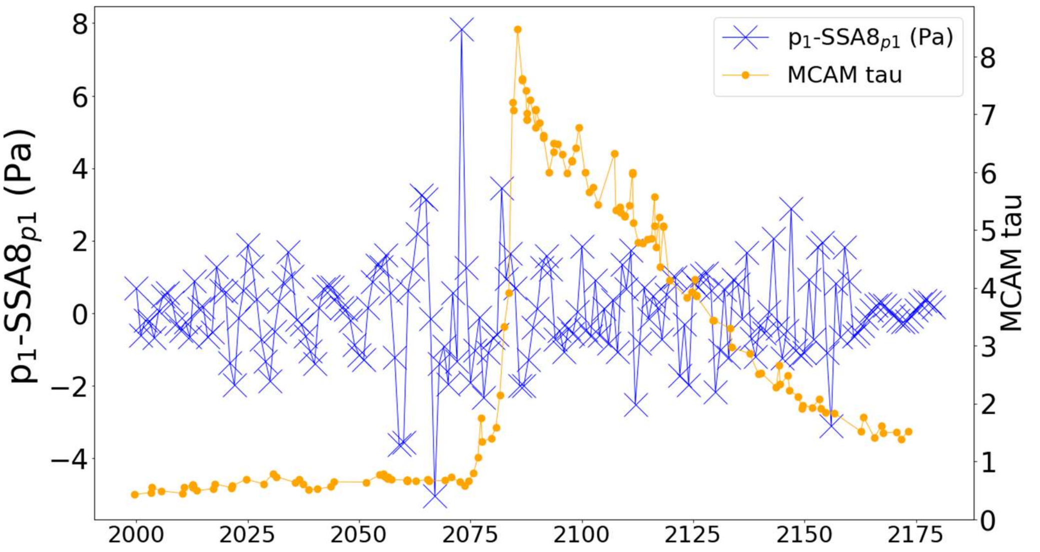

Figure 20: Zoom from Figure 13 from sol 2000 to sol 2175.

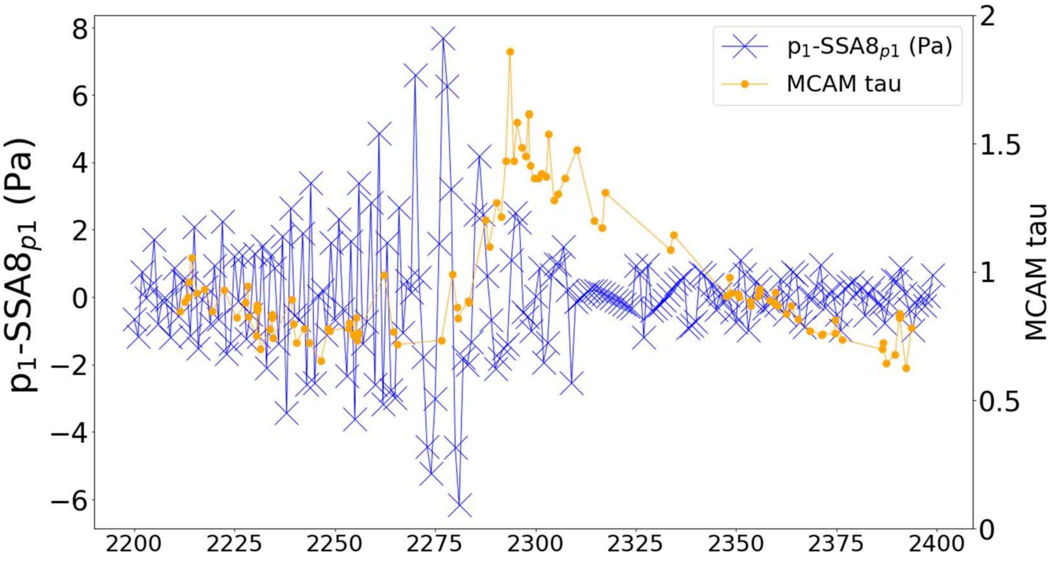

Figure 21: Zoom from Figure 13 from sol 2200 to sol 2400.

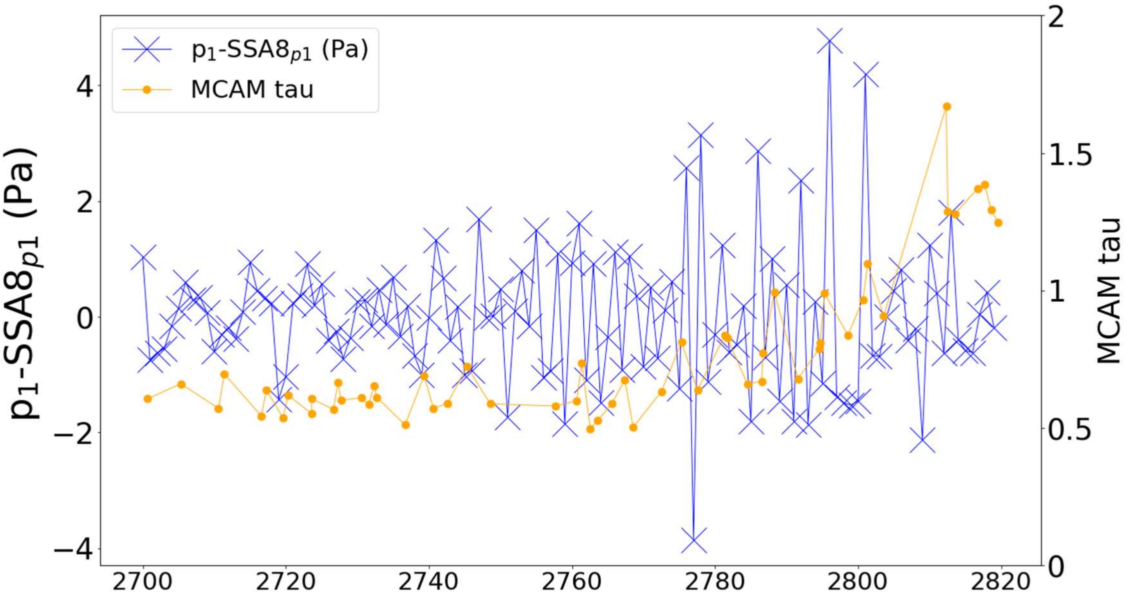

Figure 22: Zoom from Figure 13 from sol 2700 to sol 2820.

Chapter 5: Processing and analysis of the Air Temperature Sensor data

After the innovative results found in the pressure data explained in chapter 3, we decided to explore also the ATS data in depth, due to the close relationship between pressure and temperature in the context of the Martian atmospheric dynamics.

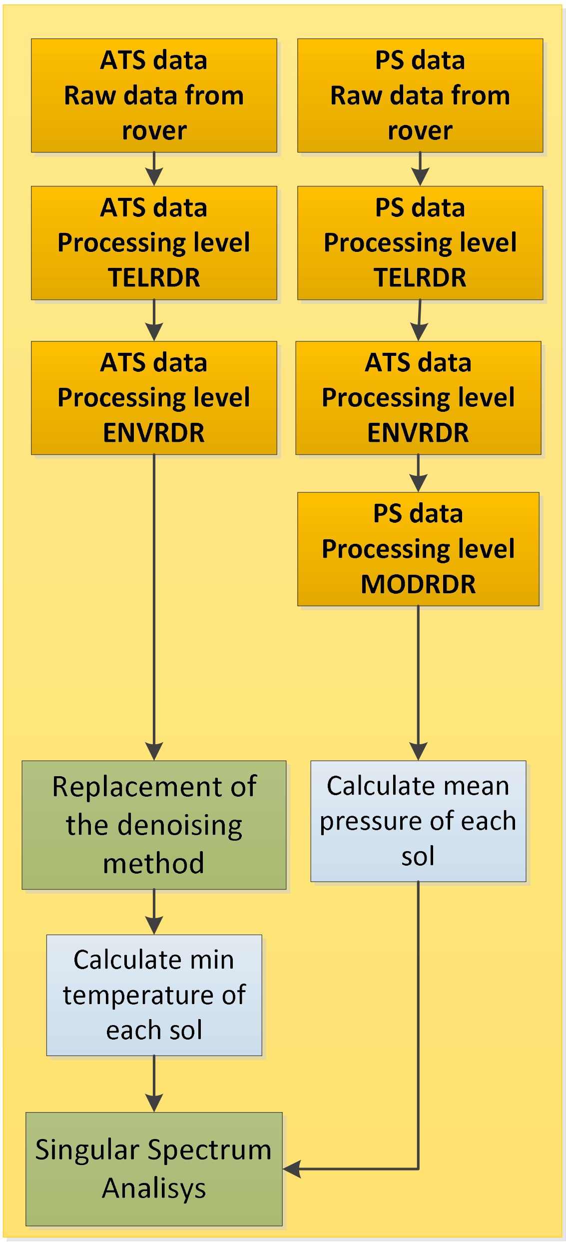

Figure 23: Scheme of the processing chains applied to ATS and PS data.

Though the working hypothesis that seemed reasonable was to look for the same precursors in the temperature, it turned out that we could not find them, so we have proceeded to look for another relationship with meteorological processes that seems to make sense and give a utility to the noise processing to improve the starting signal.

That is why this chapter is dedicated to the Temperature Sensor, to explain the algorithm that translates the data coming from the rover, in dimensionless counts, into tangible temperature values. After the data conversion process, we explain how the methods have been adapted to denoise the ATS data in order to find indicators that can be related to meteorological events.

In Figure 23, the pressure and temperature datasets and their respective processing schemes are compared.

5.1. From counts to temperature in kelvin

In Chapter 2, section *REMS data management,* we explain, in a generic way, the processing of the sensors readings from data arriving in raw format, or *counts*, until obtaining a tangible value for the given physical magnitude.

In this section, we focus on the Temperature Sensor, and we explain step by step how this process is performed to finally obtain the temperature data in kelvin.

As explained in the subsection Air *Temperature* Sensor, REMS is composed of a suite of sensors, and many of them are located within two booms, including the two ATS.

Due to their location, both ATS are constantly exposed to a number of external and internal perturbations. Externally, they are exposed to solar radiation and Martian winds, the most important external alteration factors, which can affect the readings with respect to the real magnitude, we intend to measure. Moreover, the ATS operation is mainly affected by the thermal contamination generated by the rover power supply, a Radioisotope Thermoelectric Generator (RTG), which is the warmest spot in the Curiosity rover. Internally, meaningful perturbations are related to the switching on and off of other sensors belonging to REMS, such as the WS, or the changes in configuration of the ASIC itself, which is shared with other sensors and may introduce disturbing effects to the temperature readings. Due to all this, it is difficult to provide a reliable temperature measurement without further processing.

Currently, the ATS REMS data downloaded to Earth are processed as follows [10][54]:

1. The ATS REMS data reach the Earth through eight data channels. Five channels contain information from the five Pt1000 sensors from both ATSs, and the other three channels contain information about the temperature readings from the GTS, corresponding to three different infrared wavelength channels. At this stage, the data are in a raw format, better known as *counts*, which represent the readings related to the physical magnitude corresponding to each physical sensor.

2. For the TELRDR product, the counts are translated into kelvin (K) by using the Pt1000 physical model and its conversion equations [55]. This gives the temperature measured by the sensors, and will serve as the basis for the estimation of the air temperature. The physical model takes into account certain corrections depending on other sensors' status, such as the gain of the GTS or the activation/deactivation state of the WS. The model considers six measurement signals, one from each of the five Pt1000 of both booms, while the temperature at the ATS 1 base is obtained from the three GTS readings, as mentioned before. The signals obtained from each sensor, already translated to kelvin, will be denoted from now on as $x_k[n]$, where the indexes $k = 1, 2, 3$ identify the three signals from the Temperature Sensors of Boom 1 (a Pt1000 at the tip, a Pt1000 at an intermediate point and the mean value of the three thermopiles of the GTS located at the boom base, respectively), and the indexes $k = 4, 5, 6$ identify the signals from the corresponding Pt1000 sensors of the Boom 2 (tip, intermediate point and base, respectively). The signals are considered after the analog-to-digital conversion (ADC) step, hence the discrete-time nature of the defined variables.

3. For the ENVRDR product, there is no change from the previous level, except for the addition of the estimated uncertainties for each ATS data column. This uncertainty value is obtained from the sum of the absolute value of the minimum value found and the absolute value of the maximum value found, both data searched for each sol.

$$u_k = |min\ (x_{k1}\,[n], x_{k2}[n], \ldots.x_{kN}[n])| \\ +|max\ (x_{k1}\,[n], x_{k2}[n], \ldots.x_{kN}[n]|, \tag{36}$$

   where $x_{k1}[n]$ represents the first temperature value of the signal $x_k[n]$ and N represents the number of measurements found per signal $x_k[n]$. There are six uncertainties values, one per $x_k[n]$.

4. At the last level, a the MODRDR file is generated. At this processing level, a filtering is first performed and then a specific model is applied. Both processes are detailed below:

a. A filter is applied to each of the signals in order to reduce the noise, so that we can write

$$y_k[n] = x_k[n] \ * h[n], \tag{37}$$

where $y_k[n]$ is the filtered version of the k-th signal, $h[n]$ is the impulse response of the filter and $*$ denotes the convolution operator. A Moving Average (MA) filter is the method currently applied for this task. Its application is based on the assumption that the noise is statistically independent from the temperature evolution and, therefore, it should not change the underlying structure of the signal. Under this hypothesis, averaging over a few points would effectively reduce the noise contribution. Specifically, for each data sample, it calculates the average over a predefined number of neighbors, defined by a span parameter. As it is widely known, this is an instance of a Finite Impulse Response (FIR) lowpass filter, where the span is related to the cutoff frequency. The computational cost is low but the stopband attenuation, approximately 21 dB, is often insufficient [56]. Increasing the span will reduce the cutoff frequency and the effects of the noise, at the expense of a noticeable signal distortion.

The MA filter, in fact, provides useful results for signals that are lowpass, continuous and smooth. However, when abrupt or fast changes are present, and they may arise from artifacts that may contain meaningful information, the filtering process might mask or distort them. Accordingly, this filter cannot be used if we want to get further insights about complex phenomena affecting the measurements. The calibration of the ATS and the experimental sensor response time were dominant factors to determine the span value best fitted to the nature of the ATS signals. The criterion chosen to set the MA filter was extremely conservative to avoid removing at least certain relevant artifacts, and it was decided that a reasonable trade-off for the span would be 9 samples. The coefficients used for the MA filter are constant, and the impulse response $h[n]$ is defined as follows:

$$h[n] = \frac{1}{9}\sum_{k=0}^{8} \delta\ [n-k]. \tag{38}$$

b. The retrieval model, whose details can be found in [20], extrapolates the ambient temperature using the previously processed temperature data from the three sensors at each ATS. The thermal model of the FR4 beam is based on the theory of heat transfer from a constant section fin surrounded by a fluid. In this context, a fin is a surface protruding out from an object, whose function is to increase the heat transfer rate to or from the environment by offering a maximal exposure surface area. The FR4 beam can be modeled as such a fin, transferring heat from the boom to the environment. This model, based on the principle of an infinite fin [21], is applied to give a local temperature estimation next to the tip by using all the temperature measurements from both ATSs. The estimated ambient temperature of each boom will be denoted as

$$\begin{aligned} t_1[n] &= R\{y_1[n], y_2[n], y_3[n]\} \\ t_2[n] &= R\{y_4[n], y_5[n], y_6[n]\}, \end{aligned} \tag{39}$$

where the application of the retrieval method is represented by the operator $R\{\}$, and $t_1[n]$ and $t_2[n]$ are the estimated ambient temperatures from the Boom 1 and the Boom 2, respectively.

5. At the end of the processing chain, a unique ambient temperature estimation $t[n]$ is obtained from the ATS 1 and ATS 2 estimated temperatures. Due to the low temperatures measured on Mars during the whole Curiosity mission, it is considered that the ambient temperature closest to the real value should be the coolest one among $t_1[n]$ and $t_2[n]$, specially because the rover may significantly contribute to an undesirable increase in the temperature of the air surrounding it. This can be denoted as

$$t[n] = min\ (t_1\ [n], t_2[n]\ ), \tag{40}$$

with some exceptions: It has been proven throughout the mission, that during the night, if the temperature drops too much, the ATS 2 does not work properly and shows unreliable temperatures, so it was decided, after multiple empirical tests, from 12 a.m. to 8 a.m. (LMST), the measurements will be taken exclusively from ATS 1, which provides more reliable temperatures,

$$t[n] = t1[n]. \tag{41}$$

5.2. Denoising the ATS measurements

Our initial purpose was to repeat the same process as we proceeded with pressure data, but, by comparing both dataset, it became evident that an additional denoising processing was needed before proceeding with the Singular Spectrum Analysis, as it can be seen in Figure 24, Figure 25, Figure 26 and Figure 27.

Figure 24 and Figure 25 show pressure and temperature information about sol 2000 and Figure 26 and Figure 27 show pressure and temperature information about sol 3483. Both sols were chosen randomly. The main purpose is to the variability of the pressure and temperature data. As it can be seen in both datasets, the variability of the temperature data is higher than the variability of the pressure data. The ATS provides noisier data than the Pressure Sensor, and this is mainly due to their respective locations.

Although an instance of noise filtering is currently applied to the ATS data, as described above, we consider that the current denoising method has some shortcomings that could be circumvented. In this section, it is explained how the temperature data can be denoised with more suitable alternative methods [54], before applying the SSA.

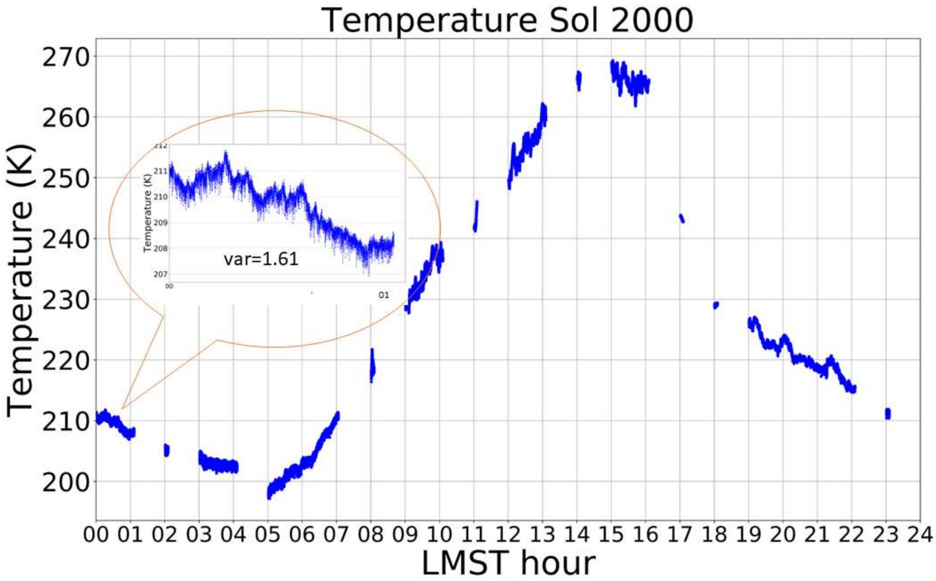

Figure 24: ATS data from sol 2000.

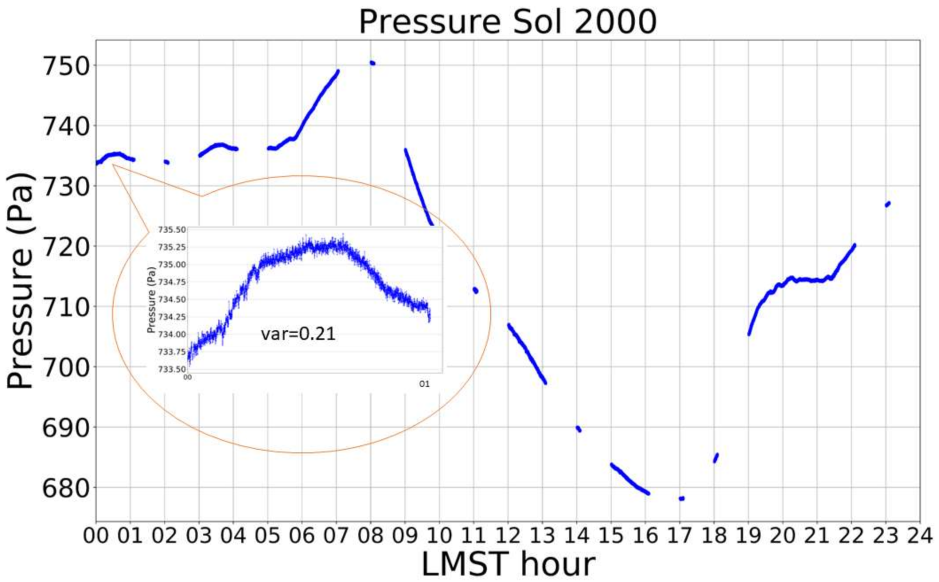

Figure 25: PS data from sol 2000.

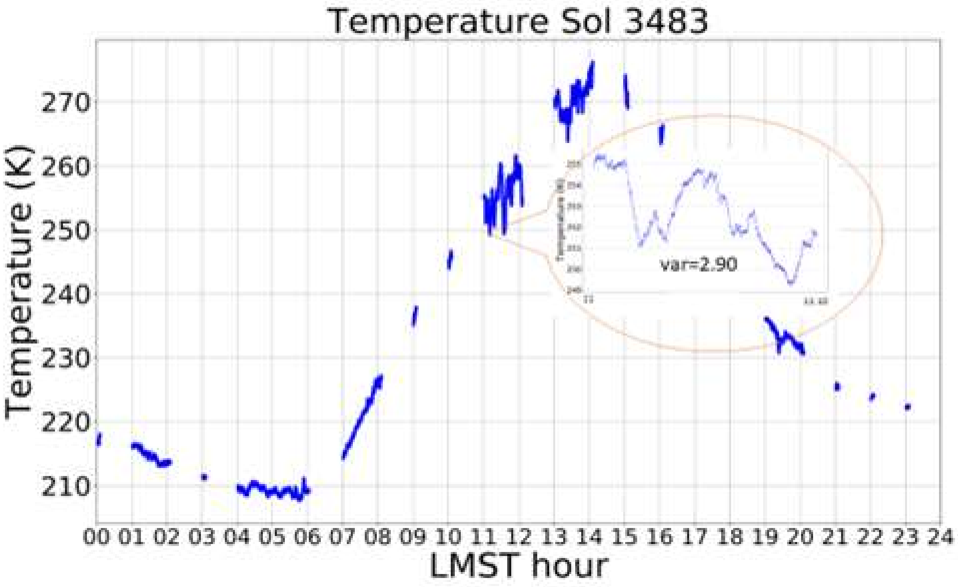

Figure 26: ATS data from sol 3483.

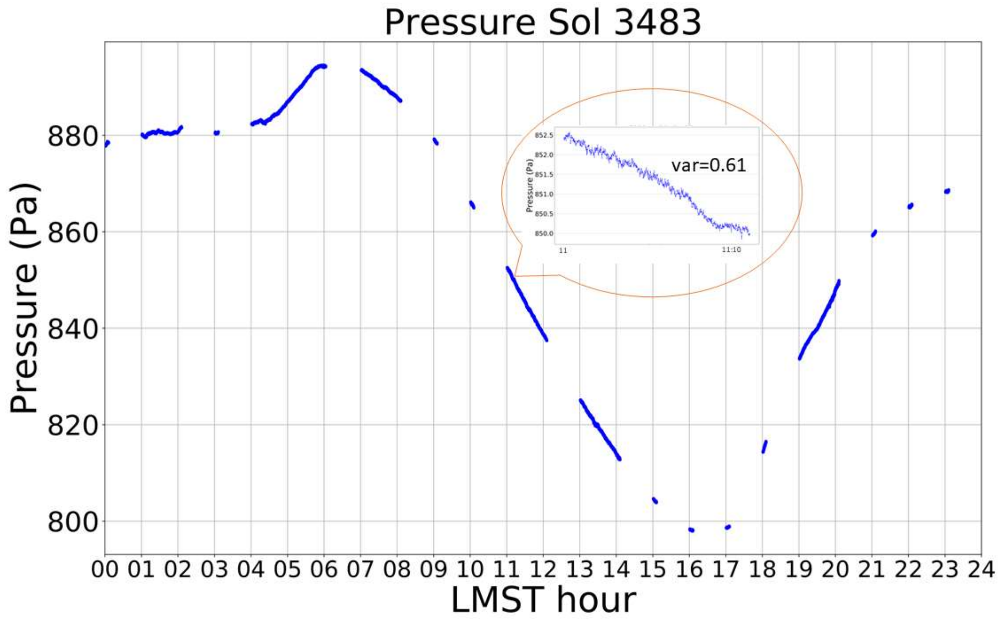

Figure 27: PS data from sol 3483.

Every day we receive very noisy data from REMS sensors and the processing explained in the previous sections is applied to such data.

As the ATS measurements are extremely noisy, due to the sensor location on the rover, it was necessary to reconsider the denoising method currently used and evaluate more efficient and powerful denoising methods to obtain higher quality data [57], before applying the SSA method.

As explained before, the methodology currently used for denoising the ATS, the Moving Average filter, MA, results in six different filtered signals, denoted as $y_k[n]$. The limitations of this method and its lack of further in-depth analysis capabilities about the results raise certain doubts about its suitability. For this reason, we have studied alternative algorithms, compared the results, and observed their potential with respect to the currently used algorithm, in order to improve the quality of the data presented to the scientific community. After having performed a thorough research, and knowing the specificity of our data, we have chosen two possible algorithms. One of the possibilities is based on the wavelets [58][59][60][61][62][63][64][65] (specifically the DWT), which comprise a family of functions that has proven to be especially useful in signal denoising. A second possibility is the Hilbert-Huang Transform [66][67], whose main feature is its high versatility for the processing of random data. In order to verify the suitability of these methods, they have been applied to two different kinds of datasets: on the one side, measurement data recorded under real conditions on Martian surface, and, on the other side, a set of test measurements theoretically unaffected by external perturbations *(*see *Annex 3* for more information*).*

In this case, we will use the ENVRDR files instead of MODRDR files because the data at this level contain the MA filtering, and we wanted to avoid this processing to propose a different denoising method.

In order to simplify the notation and without loss of generality, we have described the denoising procedure based on each of the above-mentioned methods for a generic signal $x[n]$, where we have dropped the subindex for convenience. It is to be understood that these methods will be independently applied to each of the six original signals, $x_k[n]$, $k = 1, \dots 6$.

As the current denoising filtering method has been explained and applied in detail in the previous section, only the application of the two denoising proposals is explained below.

5.2.1. Denoising with wavelets

As the wavelet algorithm was theoretically detailed in the previous section, here we only highlight the specific adaptations to our data:

1. The DWT is applied to the ATS data. The choice of the mother wavelet is mainly based on visual inspection and on the correlation between the original signal and the wavelet-denoised signal. The wavelet families Daubechies [68][69], Symlets [70] and Coiflets [40], due to the shape of their waves and the similarities with the signals we work, have been used to test which one may be better fitted to the nature of our data. We have done comparative analysis with all these bases. According to Walczak [71], the best-basis wavelet family can be defined as the basis with the minimal number of coefficients (less entropy) or maximum information for its distribution of coefficients. We have performed a test with 6000 fragments of data from 400 different sols. By using this test and the definition of best basis proposed by Walczak, we concluded that the Symlet4 family was the one determining better reconstruction results for this specific application.

2. The subband signals are thresholded with a threshold θ. After having decomposed the original signal, a thresholding step is applied to the coefficients of the wavelet decomposition. Its purpose is finding out which part of them should be qualified as

noise, and should be consequently removed. After having done deep research of the different possibilities of thresholding, a universal threshold from Donoho and Johnstone has been used [37]. This threshold has been referenced in hundreds of articles and was chosen because of its relevance in the signal processing area. In our case, this threshold is calculated by using the coefficients from the finest decomposition level. They are related to the higher signal frequencies, where the main contribution from the noise is supposed to be found. This unique threshold is used for all the coefficients of the different scales, and is calculated as [37]

$$\theta = \sigma \cdot \sqrt{2 \cdot log\, N}\,, \tag{42}$$

where N is the number of samples of the signal, and σ is the estimated standard deviation of the noise, which is calculated using the so-called Median Absolute Deviation (MAD) from the finest decomposition level of the wavelet transform, $X_1[m]$, as [37]

$$\sigma = \frac{MAD}{0.6745}, \tag{43}$$

where the MAD is defined as

$$MAD \;=\; med\,\{|X_1[m] \;-\; med\,\{X_1[m]\}|\}, \tag{44}$$

where *med*{} is the median value. These expressions take into account that the value of the noise standard deviation σ is often estimated from the median value of the $X_j[m]$ wavelet coefficients belonging to the first level of signal decomposition $X_1[m]$ [38], corresponding to the finest level of decomposition.

There is the possibility to perform hard or soft thresholding. The hard thresholding is the process of setting to zero the coefficients whose absolute values are lower than the threshold θ, while the soft thresholding acts by first setting to zero the coefficients whose absolute values are lower than the threshold θ and then shrinking the nonzero coefficients toward zero. After having done multiples tests with both alternatives, we have concluded that soft thresholding actually gives better results for our specific signals. According to this, the thresholding process is applied, following [47], for each zero-mean decomposition level, and in order to obtain the thresholded coefficients $\hat{X}_j[m]$, as

$$\hat{X}_j\,[m] = \begin{cases} X_j\,[m] - \theta, & X_j\,[m] \geq \theta \\ 0, & |X_j[m]| < \theta \\ X_j\,[m] + \theta, & X_j\,[m] \leq -\theta. \end{cases} \tag{45}$$

According to [37], the wavelet levels that do not have vanishing means should not be thresholded, since this would lead to severe distortion.

3. Reconstruction of the filtered output signal. As described, once the thresholding has been applied to $X_j[m]$, the output coefficients $\hat{X}_j[m]$ are obtained. They are used to produce the reconstructed signal $y_k^w[n]$ by means of the inverse DWT,

$$y_k^w\,[n] = \sum_{m=0}^{N-1} \sum_{j=1}^{J} \hat{X}_j[m] \cdot \psi_j[n-m], \tag{46}$$

where J is the number of decomposition levels. In our developments, the maximum decomposition level permitted has been used, according to the length of the signal $x[n]$.

4. By subtracting the reconstructed signal from the original one $x_k[n]$, we obtain an estimation of the perturbing noise $\epsilon^w[n]$ as

$$\epsilon^w\,[n] = x_k[n] - y_k^w[n]. \tag{47}$$

In *Annex 1,* the code to denoise the ATS data by using wavelet is shown.

5.2.2. Denoising with the Hilbert-Huang Transform

The denoising of the temperature data based on the Hilbert-Huang Transform, HHT, is detailed below. The exact procedure has been the result of a number of tests, where different numbers of Intrinsic Mode Functions (IMFs) and other parameters have been tuned in order to reconstruct the signal, in accordance with the nature of our data. The method proceeds as follows:

1. The EMD, explained in Chapter 3, is applied to the ATS data. The procedure consists in the extraction of IMFs (also called the *sifting* procedure):
    a. Consider the temperature dataset *x*[*n*].
    b. Extract the local maximum and local minimum from the signal.
    c. Compute upper and lower envelope.
    d. Compute the envelopes mean.
    e. Determine a criterion for the sifting process to stop and thus guarantee the IMF components retain enough physical amplitude and frequency modulations:

$$SD = \sum_{n=o}^{N} \left[\frac{\left| IMF_{j,i-1}[n] - IMF_{j,i}[n] \right|^2}{(IMF_{j,i-1}[n])^2} \right], \tag{48}$$

    where *N* is the length of *x*[*n*]. A typical value for SD is set between 0.2 and 0.3 [41]. Comparing with the two Fourier spectra, calculated by shifting only five of the 1024 points of the same dataset, can have an equivalent SD of 0.2-0,3, calculated on a point-by-point basis. Therefore, this range of values for the sifting process is a very rigorous limitation for the difference between siftings.

    f. Find an IMF following the stop criterion. The difference between the original signal *x*[*n*] and the IMF becomes a new signal, and a subsequent sifting process is performed to find the next IMF.

$$x[n] - c_1 = r_1[\mathrm{n}], \tag{49}$$

    where c_1 is the first IMF found and r_1 the residual signal, which becomes the new signal.

The sifting process has to be repeated as many times as is required to make the extracted signal satisfy the definition of an IMF. The sifting process can be stopped when the residual signal, $r_i[n]$, becomes a monotonic function from which no more IMF can be extracted.

2. The decomposition of the signal in M−empirical modes can be written as

$$y[n] = \sum_{m=1}^{M} c_m[n] + r[n], \tag{50}$$

where $c_m\,[n]$ is the IMF of the m-th decomposition level, and $r[n]$ is the residual. This latter is a monotonic function from which no more IMFs can be extracted. The signal $y[n]$ is the reconstruction obtained after applying the EMD to the input data $x[n]$. Under ideal conditions, without further processing of the IMF values, the reconstructed signal should be equal to the input signal, i.e., $y[n] = x[n]$.

3. Apply the Consecutive Mean Square Error (CMSE). This step is mainly based on the idea that the main part of the meaningful signal structure has to be found in the last IMFs, which represent the lower frequencies, while the noise is usually associated to the first IMFs, which represent the higher frequencies. Accordingly, the denoising process consists in the reconstruction of the signal after discarding the initial IMF levels. Therefore,

$$y_k[n] \;=\; \sum_{m=k}^{M} c_m[n] \;+\; r[n]\,, \tag{51}$$

is the reconstructed signal by taking into account just the last $M - k + 1$ levels. We consider the CMSE as a measure of the distortion in the reconstructed signal, according to [44]. The CMSE measures the squared Euclidean distance between two consecutive reconstructions of the signal, and it is defined as

$$\begin{aligned} CSME(y_k, y_{k+1}) &= \frac{1}{N}\sum_{n=0}^{N-1} |y_k\,[n] - \;y_{k+1}\,[n]|^2 = \\ &= \frac{1}{N}\sum_{n=0}^{N-1} |c_{k\,[n]}|^2. \end{aligned} \tag{52}$$

Therefore, the signal reconstructed using the decomposition levels from this index and the subsequent ones are judged to mainly contain the noise-free signal components. A large number of experiments have been performed and reported in the literature to support this hypothesis [44]. The IMFs from index $j + 1$ to the last one, plus the residual, should be related to the structure of the noise-free signal. However, in our case, the first IMFs could contain meaningful signal artifacts, and therefore we consider the inclusion of the first IMFs processed through thresholding.

4. Apply a threshold to the lowest IMF levels. As the noise is assumed to be distributed among the mentioned IMF levels, $IMF_{0,\dots,j}$, it is thought reasonable that a different threshold should be calculated for each of them [46]. In order to explicitly filter the

additive Gaussian noise, the universal threshold by Donoho [37] is applied here, but with a different value for each decomposition level. If we denote the threshold for the *m*-th level as θ_m, where *m* = 0, …, *j*,

$$\theta_m = \sigma_m \cdot \sqrt{2 \cdot log\, N}\,, \tag{53}$$

where N is the length of the noisy signal, and σ_m the noise standard deviation of the m-*th* IMF. The estimated value for σ_m is [45]

$$\sigma_m = \frac{MAD_m}{0.6745}, \tag{54}$$

where *MAD*$_m$ is the absolute median deviation of the m-*th* IMF, calculated as

$$MAD_m \;=\; med\,\{|c_m[n] \;-\; med\,\{c_m[n]\}|\}. \tag{55}$$

Again, a soft thresholding method is proposed to reconstruct the signal, corresponding to the expression [47]

$$\hat{c}_m[n] = \begin{cases} c_m[n] - \theta_m\,, & c_m\,[n] \geq \theta_m \\ 0, & |c_m\,[n]| < \theta_m \\ c_m[n] + \theta_m\,, & c_m\,[n] \leq -\theta_m\,. \end{cases} \tag{56}$$

The threshold depends on the decomposition level, and only the first j modes are processed.

The index j that minimizes the CMSE allows to determine which IMF level represents the limit between the part of the signal where the noise can be considered negligible, and the part where the noise is dominant [46]. The index j is given by calculating

$$j = argmin_{1 \leq k \leq M-1}\big(CMSE(y_k, y_{k+1})\big). \tag{57}$$

5. Reconstruction of the filtered output signal. After applying the thresholding to the corresponding levels, the reconstructed signal $y^h[n]$ is obtained by adding the

thresholded IMFs, the IMFs from the non-thresholded j + 1-th level and the residual, so that

$$y^h[n] = \sum_{m=0}^{j} \hat{c}_m[n] + \sum_{m=j+1}^{M} c_m[n] + r[n]. \tag{58}$$

By subtracting the reconstructed signal from the original one, $x[n]$, we obtain an estimation of the perturbing noise $\epsilon^h[n]$ as

$$\epsilon^h[n] = x[n] - y^h[n]. \tag{59}$$

In *Annex 1,* the code to denoise the ATS data by using HHT is shown.

5.2.3. Comparison of both methods

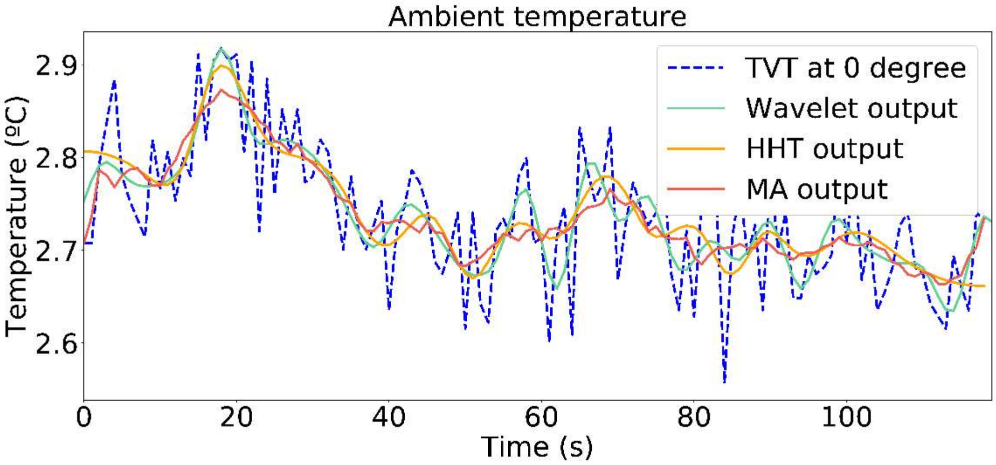

Figure 28: Results with a TVT dataset at 0 ºC. MA (span = 9), DWT (symlet4) and HHT (stop criterion: 0.05 and 0.5).

As we already know, the MA filter is the one currently used to process and make temperature data public in the PDS. In Figure 28 we have depicted data obtained from a nominal ATS TVT dataset, see *Annex 3*, where there are no abrupt changes in the evolution of the average temperature. The blue dotted line, labeled "TVT at 0 degree", represents the original signal from the test at 0 ºC, the line labeled "Wavelet output", in green, represents the filtered TVT signal after having applied the DWT based method of equations (42)-(47), the line labeled “ HHT output”, in orange, represents the filtered TVT signal after having applied the HHT based method of equations (48)-(59), and the line labeled “MA output”, in red, represents the filtered TVT signal after having applied the MA filter, corresponding to equations (37) and (38). We can thus visualize how each method performs the filtering of the noise. The MA filter is the method with the softest reconstruction since it makes use of a relatively large span, as mentioned earlier. As a consequence, in this situation, a loss of artifacts in the signal may easily occur. The HHT and the DWT based methods use more elaborated processing algorithms, and are able to follow the original signal with greater accuracy, at least by visual inspection. Nevertheless, overall, the three methods do not offer a much different behavior when processing a signal that do not contain abrupt changes.

However, when using the MA filter with a dataset containing abrupt changes in the measured temperatures, its limitations become noticeable. To confirm this, a characteristic dataset from REMS [57] has been used. The data belong to measurements in sols 68, 74, 107 and 120, where we can see how some kind of sharp temperature variations are present. Due to the ATS sensitivity, we think that these signal features might provide extra information possibly related to meteorological events occurring in Gale, also supported by other sensors, as evidenced in [72]. The plots in Figure 29, Figure 30, Figure 31 and Figure 32 show data from the previously mentioned sols. They make evident how each method behaves distinctively when processing the noise and reconstructing the filtered signal. The MA filter cannot follow the noisy signal as accurately as the HHT or DWT based filtering, possibly losing relevant features for the study of local atmospheric temperature variations or for the discrimination of signal disturbances. Therefore, we can say that the current processing method softens the abrupt signal changes excessively. The zoomed-in area shown in Figure 30 and plots in Figure 29, Figure 31 and Figure 32 highlight how the MA filter fails to follow the signal evolution, in contrast with any of the other two methods detailed. For this reason, more powerful denoising algorithms such as the ones based on the HHT or the DWT have been proposed when abrupt changes may appear, as it is very often the case with the data taken in the Martian atmosphere.

In order to characterize the performance of these methods for the discrimination between noise and the target signal, and to assess the corresponding filtering potential, we compare the results when using the ATS TVT with the results when using the ATS data taken on Mars. As it is explained in *Annex 3*, the TVT and the cruise checkout signals are supposed to be virtually free from external disturbances, and may be used as a reference to evaluate the denoising performance in conditions without noticeable signal distortion. To perform fair comparisons, it has been necessary to select datasets from

Mars measurement campaigns with similar temperatures and the same sensors configuration. To obtain the estimated noise signals in both cases, from the ATS TVT data and from the Martian ATS data, we have applied the MA, the DWT and the HHT methods as explained before.

As previously seen, the noisy part of the signal is estimated by subtracting the reconstructed signal, using the MA, DWT or the HHT, from the original one, $x[n]$. The standard deviation is used here as the comparison parameter, because by hypothesis it should be directly related to the estimated noise power. If the standard deviation values are similar when using a controlled signal dataset (i.e., TVT data) as compared to the results from a very perturbed dataset from Mars, it may be inferred that the corresponding method is effective in filtering the noise without affecting relevant signal information, because we can assume that we are essentially removing the electronic noise contribution.

According to all this, Table 3 shows the estimated standard deviation of the noise, under the conditions mentioned, at different nominal temperatures. Table 4 shows the numerical results for the signals in Figure 29, Figure 30, Figure 31 and Figure 32. The data in Table 3 and the first three rows in Table 4 can be easily compared by matching the closest nominal temperatures, so that we can see a high degree of compatibility between the standard deviation of the noise estimated with the TVT data, and the standard deviation obtained by perturbed Mars data, in the cases when we apply either the DWT or the HHT methods. Notice that, in the case of the fourth row, in Table 4, corresponding to a mean temperature around −125 ^{o}C degrees, from sol 68, there is no TVT data to compare with.

By examining both tables, we can corroborate that, when there is no perturbation, the estimated noise variance, shown in Table 3, is quite similar for any of the three methods. If this were the only figure-of-merit, any of the three methods would be equivalent. However, when a perturbation is present in the target signal, as it can be seen in Table 4, the estimated standard deviation of the noise is clearly higher in the case of the MA with respect to the values attained with HHT and DWT.

Table 3: σ with non-perturbed ATS data against the σ with TVT data, when applying MA, DWT and HHT methods.

Nominal temp ºC	**Origin from Data**	**σ noise ºC (DWT)**	**σ noise ºC (HHT)**	**σ noise ºC (MA)**
0	**TVT**	0.04	0.05	0.05
0	**ATS (Mars)**	0.06	0.04	0.07
-10	**Cruise**	0.05	0.06	0.06
-10	**ATS (Mars)**	0.07	0.05	0.07
-30	**TVT**	0.04	0.04	0.05
-30	**ATS (Mars)**	0.08	0.05	0.08
-50	**TVT**	0.04	0.04	0.04
-50	**ATS (Mars)**	0.08	0.06	0.09
-70	**TVT**	0.05	0.07	0.06
-70	**ATS (Mars)**	0.09	0.05	0.07
-90	**TVT**	0.07	0.08	0.08
-90	**ATS (Mars**	0.09	0.05	0.07

Table 4: Estimated σ of the noise from Martian ATS data with perturbations, when applying MA, DWT and HHT.

Origin from Data	**σ ºC (DWT)**	**σ ºC (HHT)**	**σ ºC (MA)**
Noise from Sol 120	0.064	0.078	0.1251
Noise from Sol 87	0.0935	0.1760	0.2865
Noise from Sol 107	0.1094	0.1700	0.2713
Noise from Sol 68	0.9465	0.9802	1.3773

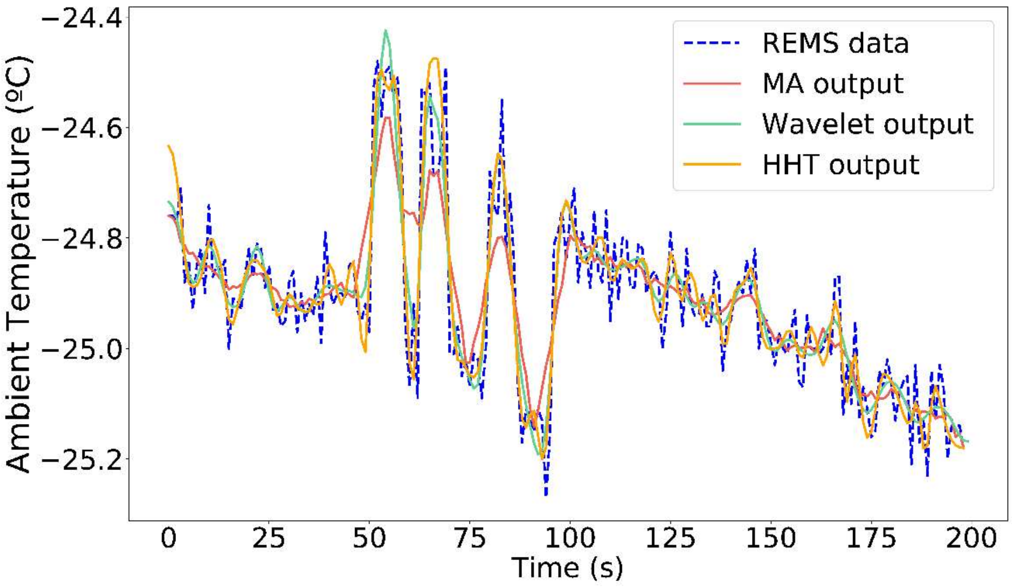

Figure 29: Artifacts on Sol 120, around 16:00.

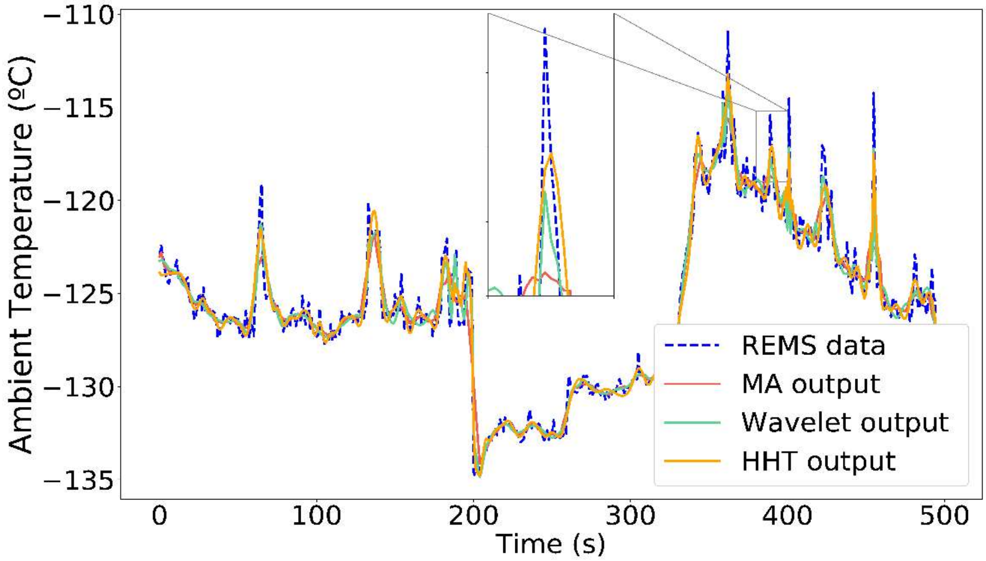

Figure 30: Artifacts on Sol 68, around 7:00.

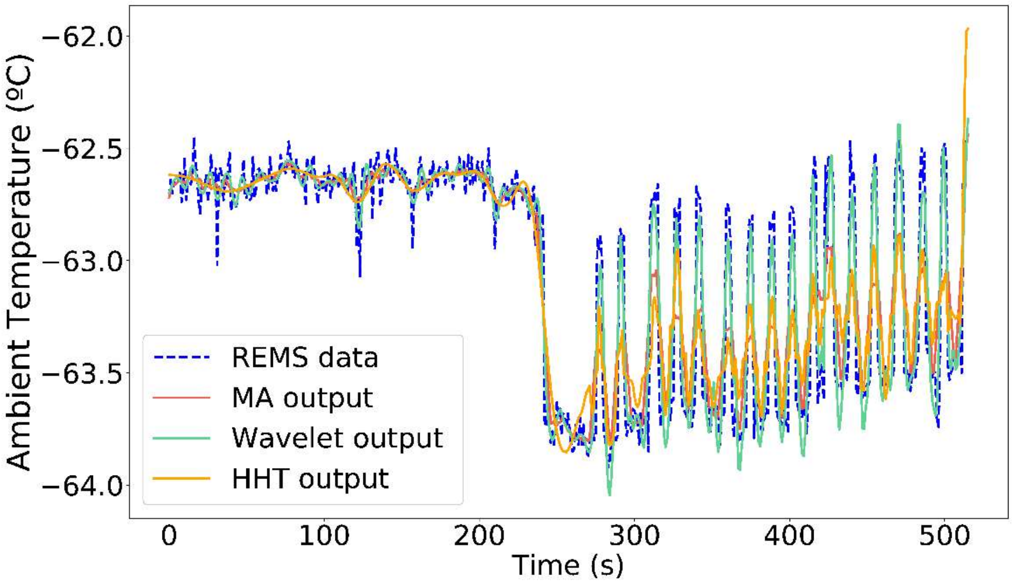

Figure 31: Artifacts on Sol 107, around 4:35.

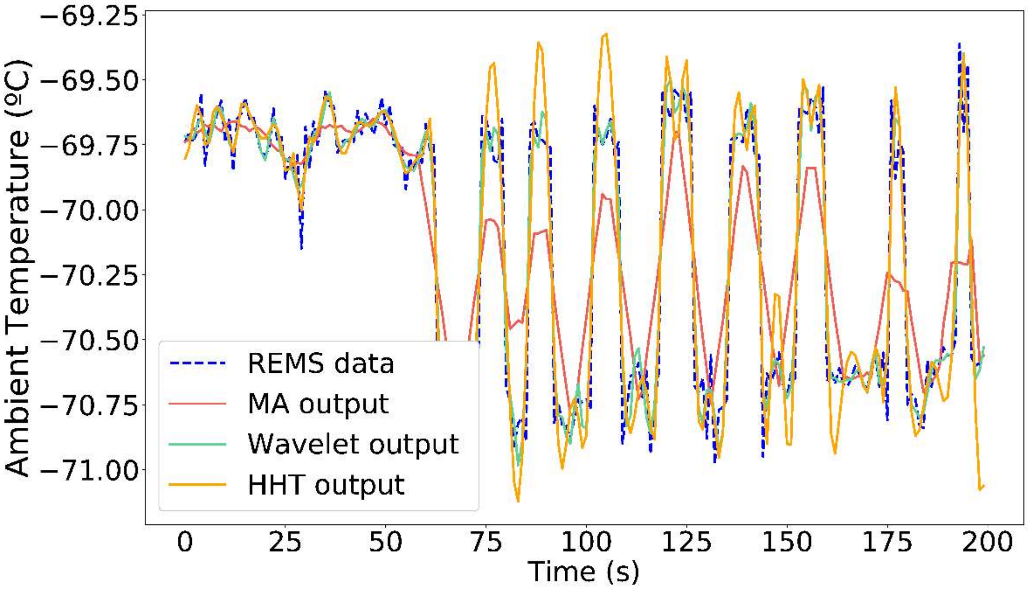

Figure 32: Artifacts on Sol 87, around 4:00.

In order to quantitatively assess the goodness of the proposed methods, we have included a metric, normally used in the denoising literature to measure the distortion introduced in the filtering of a signal, i.e., the percentage root mean square difference (PRD) [73], defined as

$$\mathrm{PRD} = \sqrt{\frac{\sum_{n=0}^{N-1}(x[n] - \hat{y}[n])^2}{\sum_{n=0}^{N-1}(x[n])^2}}.100, \tag{60}$$

where $x[n]$ is the original data, and $\hat{y}[n]$ is the reconstructed signal. A lower PRD represents a better reconstruction of the signal after denoising. As can be seen in Table 5, the current moving average filtering significantly increases the PRD of the signal, while the other proposed techniques improve the denoising with reduced levels of distortion, in agreement with what is seen in the plots of Figure 29, Figure 30, Figure 31 and Figure 32.

This metric provides additional evidence that supports our decision to modify the denoising filtering method, since MA distorts the reconstruction signal excessively.

For all of these reasons, we discard the MA filter method.

Table 5: PRD Metrics.

Data from	DWT	HHT	MA
Sol 120	0.2592	0.2525	0.3551
Sol 87	0.1706	0.2510	0.4084
Sol 107	0.2201	0.2711	0.3756
Sol 68	0.5568	0.6306	0.9562

5.2.4. Selection of the most appropriate denoising method

Once we have seen that the DWT and the HHT methods proposed can be equally efficient in processing the noise while keeping characteristic signal features, an additional issue to be considered in order to choose the best denoising method for our purpose is the processing time consumption. This is an important requirement for the processing of REMS ATS data due to the fact that reports of results from all REMS sensors must be delivered in a very short period of time after the arrival of the raw data to Earth. Table 6 shows the time taken by the application of the three methods to a dataset of 20 MBytes, using an Intel Core processor i7, python 3.7 software and standard libraries. We can see that the DWT based algorithm is around 50 times faster than the HHT based algorithm. For this reason, the DWT would be a better option in this particular case, where processing time matters. This does not mean that HHT could not be used at all in this context, though it will be very convenient to prioritize the method with higher time efficiency. The reason is that this will give a valuable margin that could be used to address possible unexpected outcomes and perform other side tasks (e.g., adding significant flags to the data) before presenting the final results.

Table 6: Processing time comparison, using a PC with INTEL CORE PROCESSOR I7-7500U CPU @ 2.70GHZ, 2904MHZ.

Algorithm	Processing time (s)
MA	0.0009
HHT	0.12
DWT	0.0023

5.3. Application of SSA to the Air Temperature Sensor data

In the previous subsection, it has been argued that DWT is the more appropriate method to denoise ATS data due to the faster computational time compared to HHT.

As it is shown in Figure 10, data are presented in different processing levels. As stated before, in this case, it was decided not to use the MODRDR product, as was done for the pressure, because the data at this level contained the MA filtering, and we wanted to avoid this processing because we propose a different denoising method. For our purpose, we have used the ENVRDR file, containing temperature data in columns 28, 29, 30, 35, 36 and 37. Specifically, we have used the data from the ATS 1 tip, column 30, because the tip position is the one that suffers less the thermal plume of the MSL, due to its position furthest from the mast. Moreover, it has been chosen ATS 1 instead of ATS 2 because, as explained before, temperatures from ATS 2 are not always correct during nights and have to be discarded.

As mentioned above, at the beginning of this chapter, a reasonable working hypothesis was to apply the SSA to the temperature to look for the same precursors in this magnitude, but as we could not find them, we have proceeded to look for another relationship with meteorological processes that seems to make sense and give a utility to the noise processing to improve the ATS signal.

Prior to the application of SSA to temperature data, and after multiple empirical tests, we found that the minimum temperature revealed some information that caught our attention and could be relevant to our research. We calculated the minimum value for each sol, by using all ATS 1 tip temperature data measured on each sol (ENVRDR file). We have also created an intermediate file to save all these values. This intermediate file has provided the input data for the SSA algorithm. The code to generate this file can be seen in *Annex 1* and an extract of the file is shown in *Annex 4.*

Next, we detail the application of the SSA to the minimum temperature, t_{min}:

1. Let's consider two time series, a minimum temperature signal, t_{min}, and a sol series, *sol*,

$$t_{min} = (t_{\text{min\_1}}, t_{\text{min\_2}}, t_{\text{min\_3}}, \dots, t_{\text{min\_}n}) \quad (61)$$

$$sol = (sol_1, sol_2, \dots, sol_n),$$

where *sol* contains the mission sols that measured 24 observations, t_{min} shows their respective measured values of minimum temperature in these sols, and *n* is the number of sols for which REMS collected a full diurnal cycle (24 observations).

2. The minimum temperature signal t_{min} is detrended by removing its mean value t_{mean}

$$t_{mean} = n^{-1} \sum_{i=1}^{n} t_{min}(sol_i), \quad (62)$$

3. The resulting sequence of values, $t_{min} - t_{mean}$, has gaps for missing sols where there was not a full diurnal cycle of magnitudes measured.

$$t_{min} - t_{mean} = (t_1, t_2 \dots, t_n), \quad (63)$$

4. To solve the gaps problem, an expanded signal is defined, $\ddot{T}$, where a 0 is added at the positions of sols where there are no 24 observations, while maintaining the main frequency characteristics [29],

$$\ddot{T} = (T_1, T_2, \dots, T_i \dots, T_N),\ T_i = \begin{cases} 0\ if\ SOL_i \not\subset sol \\ t_j\ \ if\ SOL_i \subset sol, \end{cases} \quad (64)$$

where *j* is the position where SOL_i is in *sol*, SOL_i is a random sol from the mission and *N* is the dimension of the new vector $\ddot{T}$, corresponding to the last sol considered in the study.

5. Create the trajectory matrix *Y* with dimensions (*N* – *M* + 1) x *M*, where *M* is the temporal window covering. The choice of M has been detailed in Chapter 3. This converts a one-dimensional minimum temperature series $\ddot{T}$ into a multidimensional minimum temperature series $Y = (T_1 \dots T_M), (T_2 \dots T_{M+1}), \dots, (T_{N-M+1} \dots T_N)$.

The result of this step is the trajectory matrix $Y = [Y_1,\ Y_2, \dots Y_{N-M+1}]$,

$$Y = \begin{pmatrix} T_1 & T_2 & T_3 & \dots & t_M \\ T_2 & T_3 & T_4 & \dots & T_{M+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T_{N-M+1} & T_{N-M} & T_{\dots} & \dots & T_N \end{pmatrix} \tag{65}$$

6. Compute the matrix $S = YY^T$ (dimension $M \times M$) in order to apply the Singular Value Decomposition [23].

7. Compute the eigenvalues and the eigenvectors of the Matrix S. In this case, the cross correlation between Y and Y^T is calculated so that its eigenvalues are normalized and each eigenvalue, e_i, measures the ratio of variability associated with its eigenvector, v_i

$$v_i \ = \ (v_{i,1}, v_{i,2}, \dots., v_{i,M}). \tag{66}$$

8. There are $N - M + 1$ principal components, pc_i, each one associated with an eigenvalue, e_i, obtained as the projection of the matrix Y into the subspace associated with its eigenvector, v_i,

$$pc_i \ = \ Y^T . v_i . \tag{67}$$

9. The expanded signal, $\ddot{T}$, can be reconstructed using all the principal components, pc_i. Each component provides the contribution, r_k, to the reconstruction of $\ddot{T}$ by multiplying the component by the eigenvector as

$$\begin{gathered} r_{k,1} = \ pc_{k,1}. v_{k,1} \\ r_{k,2} = \frac{pc_{k,2}. v_{k,1} + \ pc_{k,1}. v_{k,2}}{2} \\ r_{k,j} = \frac{pc_{k,j}. v_{k,1} + \ pc_{k,j-1}. v_{k,2} + \dots + \ pc_{k,1}. v_{k,j}}{j} \qquad j = 3, \dots.. M - 1 \\ \dots\dots. \\ r_{k,N-1} = \frac{pc_{k,N}. v_{k,M-1} + \ \ pc_{k,N-1}. v_{k,M}}{2} \\ r_{k,N} = \ pc_{k,N}. v_{k,M} \ . \end{gathered} \tag{68}$$

The reconstruction is made with the principal components associated with the eigenvectors that explain 95 % of the variability, decided by heuristic and empirical criteria. This corresponds to the first 3 eigenvectors. To reach the final result, as it has been detailed before, we covered the data gaps and thereby reduced their influence from missing data to existing data at each transition by comparing the reconstructed signal with the zeroes in the gaps of the original $\ddot{T}$. Those zeroes were replaced with the $r_{k,i}$ to repeat the previous analysis, extracting a new set of r_k. It is then compared the new r_k values with the previous set, repeating the iteration until the difference between one reconstruction over the gaps and the next reconstruction agreed within a standard deviation of 1 %.

5.4. Results and discussion

This section focuses on the results obtained after having applied the SSA to the minimum temperature values.

Figure 33 shows the minimum tip ATS 1 temperature of each Martian day data after being denoised with wavelets and before applying SSA.

Figure 34 shows, in blue, the reconstruction of the minimum temperature, after subtracting the mean, by using the first 3 eigenvectors from SSA decomposition, which represent the 95 % of the original signal. Circles in orange mark the storms period. After the storms period, the aphelion starts, which is the period of time when Mars is farthest away from the Sun. For this reason, we should expect temperatures to be coldest, as marked with circles in green. However, during these periods after the storm, we observe some humps, where the temperature rises. This pattern is remarkable because it is repeated always after the dust storms. As we can see in Figure 33, these peaks were not clearly observed before applying the SSA.

With the aim to find a physical meaning of these peaks, which the SSA algorithm has revealed in the ATS data, we find in the bibliography there have been publications about ice clouds in the Aphelion period [74]. Apparently, these clouds create a greenhouse effect that raises the temperature.

To study this hypothesis, the Nasa’s Mars Exploration Program has a website where all images for all missions are published. We have found very interesting images from science cameras, the MastCam camera, and engineering cameras, NAVCAM, both from the Curiosity rover. Figure 38, Figure 39 and Figure 40 show ice clouds during sols 1757, 2425 and 3063. As it can be seen in Figure 34 and in enlargements of this figure, Figure 35, Figure 36 and Figure 37, a kind of humps are happening during these sols. We show Figure 35, Figure 36 and Figure 37 to facilitate the visualization of these humps. So, we could conclude that these temperature increments, revealed by the SSA algorithm, seem to correspond to the greenhouse effect caused by the ice clouds. Moreover, images from Curiosity cameras help to assess this hypothesis.

After a thorough study, the correct application of signal processing methods to temperature data reveals very interesting information.

It is very rare to capture images of the sky, since these cameras mainly capture rock formations. Although it is a very important milestone to have found these clouds, which coincide with these hump periods, a deeper study and a search and measurement campaign is needed to strengthen the idea.

All this valuable information found, thanks to the use of the explained methods, seems to corroborate an initial hypothesis of finding cloud periods in the temperature data, although it needs further elaboration and study in the near future. Finding three concrete cases is really a start and a great milestone but this hypothesis requires further and systematic study. The possibility of relating atmospheric events with indicators of clouds discovered in the temperature data opens the door for further research after the completion of this thesis.

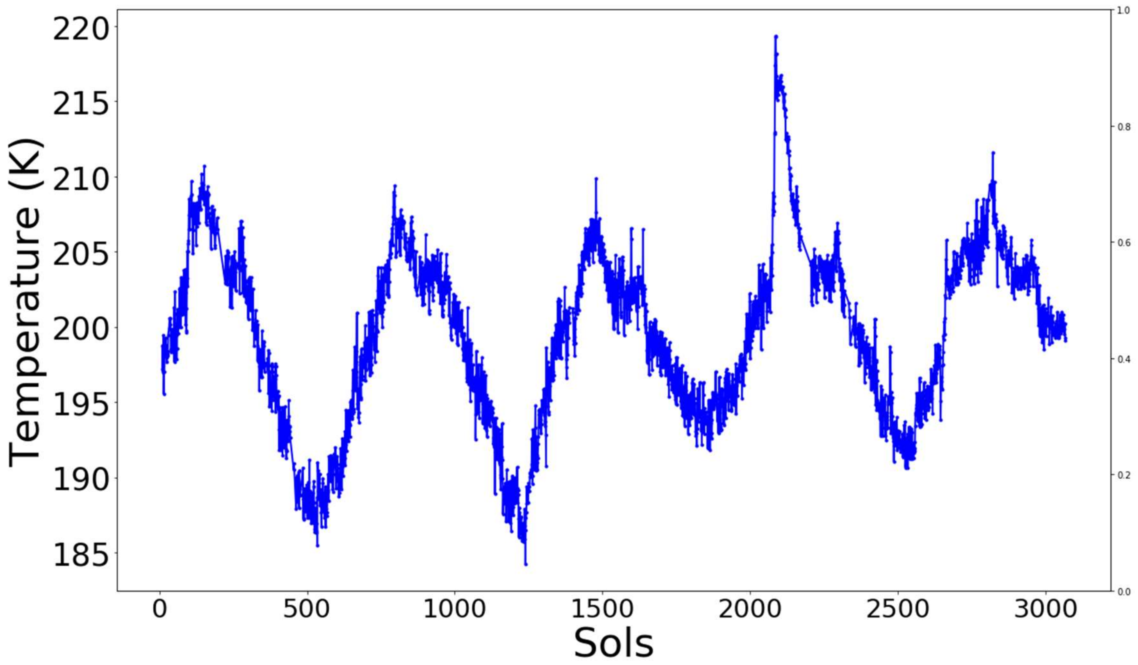

Figure 33: Minimum temperatures from the ATS 1 tip data denoised before applying the SSA.

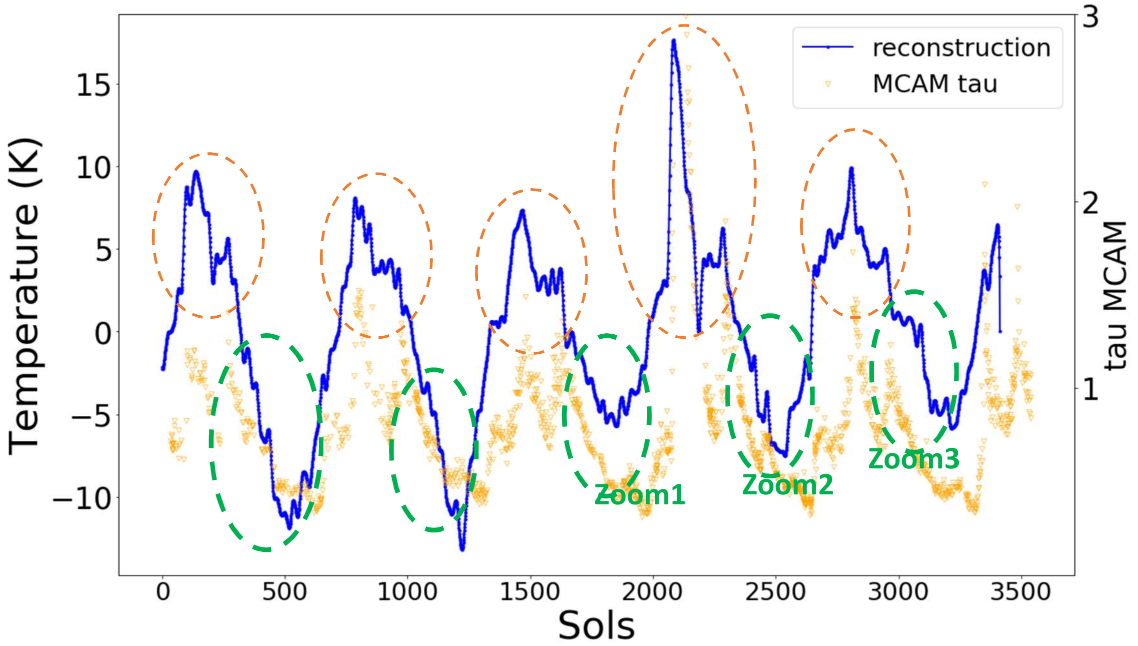

Figure 34: Reconstruction of the min temperature by using SSA for all sols of the mission. Areas marked in orange determine the dust storms periods and areas in green determine the non-dust storms periods. Mean values has been subtracted from the reconstruction.

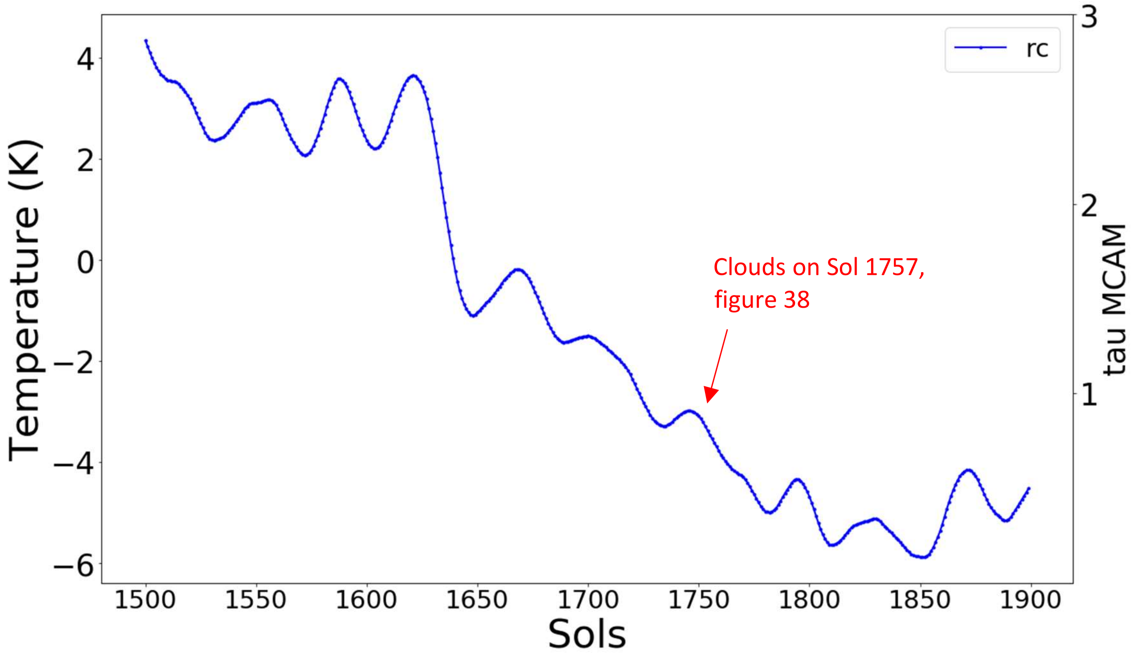

Figure 35: Zoom1, after storm 3.

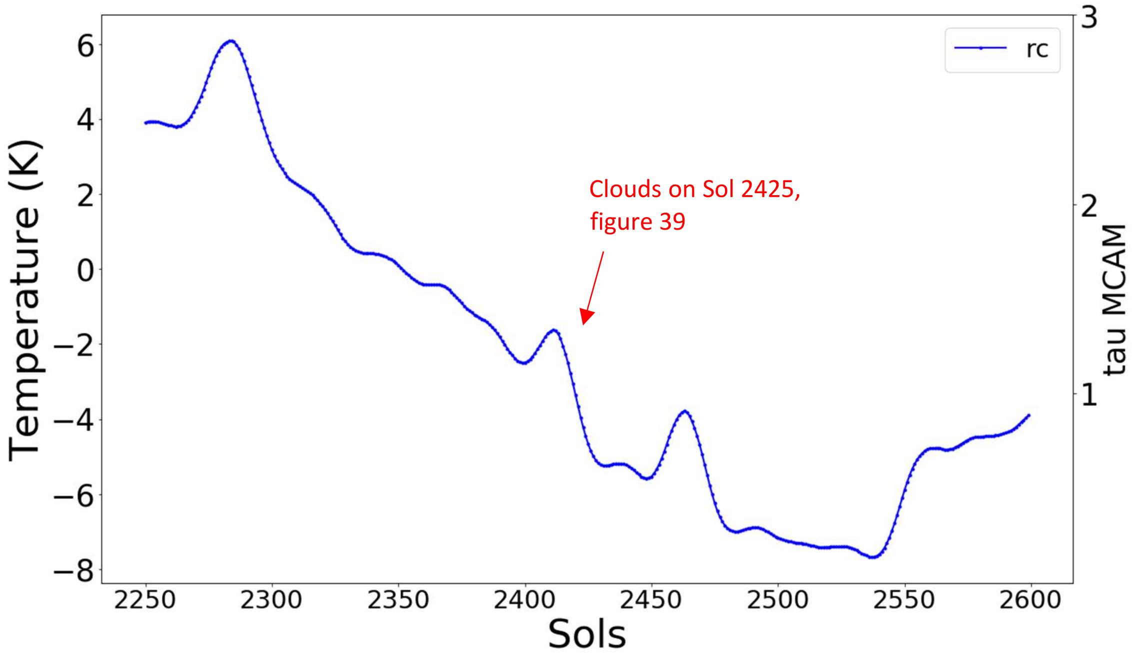

Figure 36: Zoom 2, after storm 4.

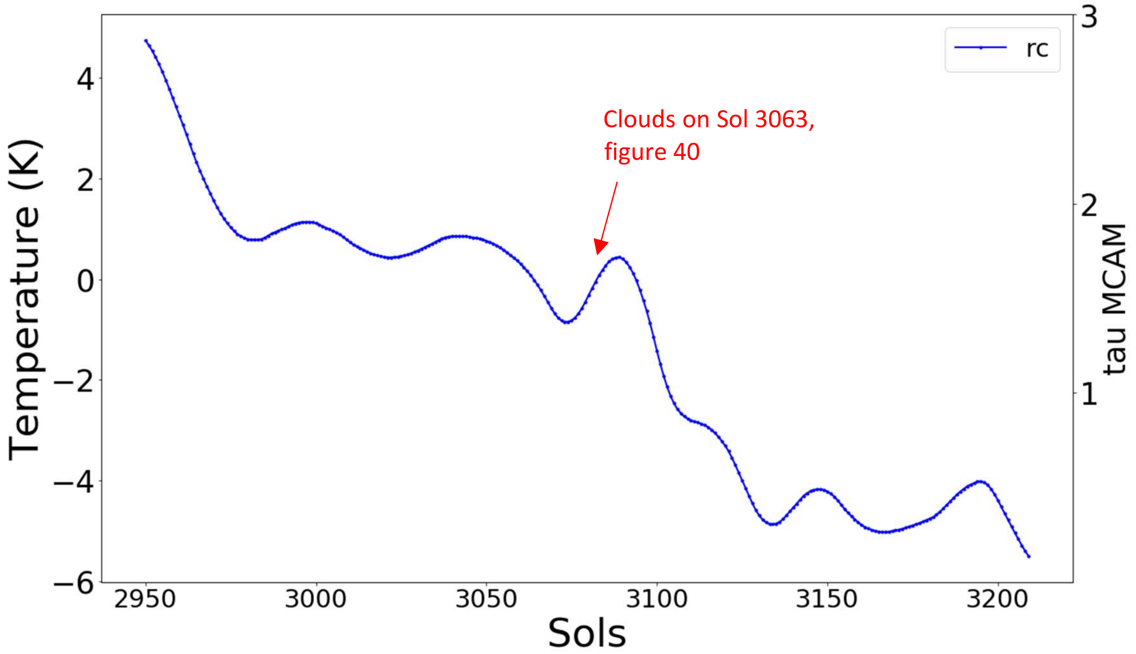

Figure 37: Zoom 3, after storm 5.

Figure 38: Clouds from sol 1757, NAVCAM camera from rover Curiosity. (Credit: NASA/JPL-Caltech)

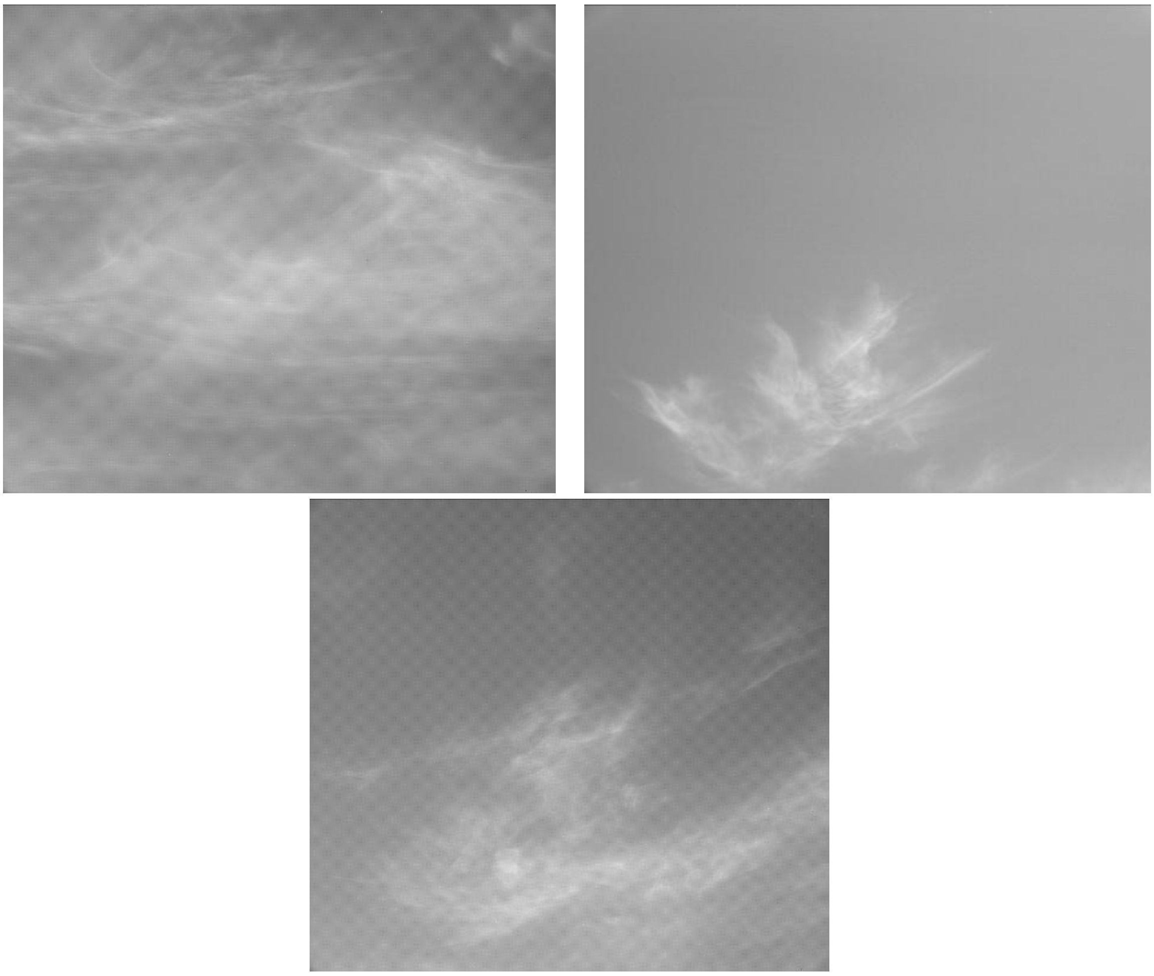

Figure 39: Clouds from sol 2425, MCAM camera from rover Curiosity. (Credit: NASA/JPL-Caltech)

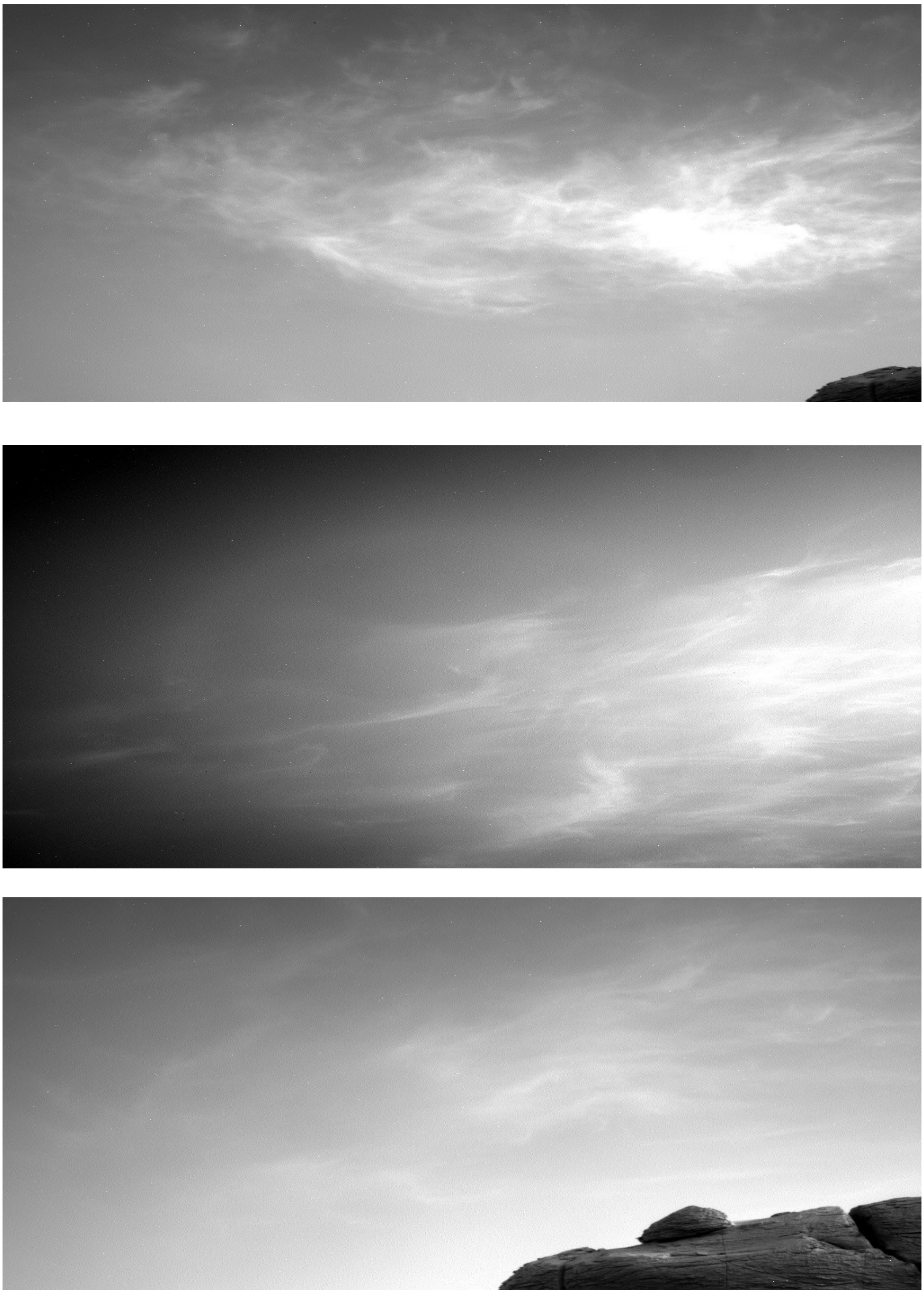

Figure 40: Set of 3 images of clouds from Sol 3063, captured by NAVCAM camera, from rover Curiosity. (Credit: NASA/JPL-Caltech)

Chapter 6: Conclusions

This research has been motivated by the processing of REMS Pressure and Temperature Sensors data, for the analysis of environmental processes occurring in Gale crater, on Mars.

- Pressure is initially chosen as the magnitude to be studied, as it is a highly influential atmospheric characteristic that models and influences the global atmospheric circulation patterns of the Mars planet.
- Singular Spectrum Analysis has been applied to the pressure data, and precursors of dust storms, one of the most studied atmospheric events on Mars due to the turbulent changes they produce on the planet, have been discovered.
- Finding precursors of dust storms on Mars is a breakthrough in science.
- Dust storms wreak havoc on many Martian mission sensors. Anticipating the storm could help future Mars missions to protect sensors before the storm arrives.
- Due to the discovery in the pressure data, we have analyzed the temperature data, because of the close relationship between both atmospheric magnitudes.
- However, temperature has undergone a prior denoising process because they are noisier data due to their location on the rover, and because they are local data.
- Several methods have been proposed for the temperature denoising: wavelets and EMD.
- Finally, a wavelet-based method has been applied to the temperature data for denoising.
- After having applied Singular Spectrum Analysis to the denoised temperature data, the results reveal humps and other interesting features in different periods of the signals.
- However, these ultimately could not be correlated with the same atmospheric processes and dynamics that control storms, in the same way that they are identified through pressure.
- On the contrary, our analysis led us to another interesting result that can be related to periods of clouds in Gale crater, creating a greenhouse effect in the area, and raising temperatures.
- These exciting results, the cloud periods in Gale, have been corroborated with cameras from the Curiosity rover, NAVCAM and MastCam.
- We consider finding cloud periods thanks to indicators in temperature data is another relevant discovery.

- The application of signal processing methods to the data has revealed information, which a priori, is not shown in the data.
- In our study, methods have helped to discover indicators of relevant atmospheric features on Mars, and in this way, we do our bit to know our neighboring planet better.
- The application of signal processing, suitably motivated and adapted to the REMS data, has served to demonstrate our initial purpose: the analysis of atmospheric processes on Mars.

Chapter 7: Future work

There are several and quite interesting open research lines as a consequence of this thesis:

- To continue investigating possible ice cloud indicators in the temperature data.
- To continue investigating possible dust storms precursors in the temperature data.
- Application of the denoising algorithms to the rest of the sensors that make up REMS and to the sensors of the Perseverance meteorological station.
- The identification of interesting features by applying the Singular Spectrum Analysis to other REMS sensors.
- Application of the proposed signal processing methods to other missions currently operating.
- Correlation of data from the different missions currently operating.

Bibliography

[1] Peters, William T. “*The Appearances of Venus and Mars in 1610*”. In: *Journal for the History of Astronomy*, vol. 15, no. 3, pp. 211–214, (1984), doi:10.1177/002182868401500306.

[2] Leovy, Conway B. and Smith, Bradford, A. and et al. “*Mariner Mars 1969: Atmospheric results*”. In: *Journal of Geophysical Research,* vol.76, (1971).

[3] Larsen, S.E. and Jørgensen, H.E. and Landberg, L. and Tillman, J.E. “*Aspects of The Atmospheric Surface Layers on Mars and Earth*”. In: *Boundary-Layer Meteorology* vol. 105, pp. 451–470, (2002). DOI: https://doi.org/10.1023/A:1020338016753.

[4] Haberle, R. M. and McKay, C. P. and Schaeffer, J. and Cabrol, N. A. and Grin, E. A. and Zent, A. P., and Quinn, R. “*On the possibility of liquid water on present-day Mars*”, In: *J. Geophys. Res.*, vol. 106, pp. 23317– 23326, (2001), doi:10.1029/2000JE001360.

[5] Orosei, R and et al. *“Radar evidence of subglacial liquid water on Mars”.* In: *Science.* Vol. 361, pp. 490-493, (2018).

[6] Cantor, Bruce A. and James, Philip B. and et al. “*Martian dust storms: 1999 Mars Orbiter Camera observations*”. In: *Journal of Geophysical Research: Planets*, vol. 106, (2001).

[7] Gierasch, Peter J. “*Martian dust storms*”. In*: Reviews of Geophysics*, vol. 12, (1974).

[8] Leovy, Conway B. and Saltzman, Barry “*The General Circulation of Mars: Models and Observations*”. In: *Elsevier, vol. 28, (1985).*

[9] Arvidson, Raymond E. and Guinness, Edward A. and Moore, Henry and Tilman, James and et al. “*Three Mars Years: Viking Lander 1 Imaging Observations*”. In: *Science, (1983).*

[10] Gómez-Elvira, Javier and et al. “*REMS, The environmental Sensor Suite for the Mars Science Laboratory Rover*”. In: Space *Science Reviews,* (2012).

[11] Westfall, Richard “Science *and Patronage: Galileo and the Telescope*”. In: *Isis*, vol. 76, no. 1, (1985).

[12] Golombek, Mathew P. “*The Mars Pathfinder Mission*”. In: *Journal of Geophysical Research,* vol. 102, (1997).

[13] Soffen, Gerald A. and Young, A. Thomas. “*The Viking Mission to Mars*”. In: *Icarus*, vol. 16, no. 1, (2012).

[14] Grotzinger, John P. and et al. “Mars Science Laboratory Mission and Science Investigation”. In: *Space Science Reviews*, (2012).

[15] Gómez-Elvira, Javier et al. "*Curiosity's rover environmental monitoring station: Overview of the first 100 sols*". In: *Journal of Geophysical Research: Planets,* vol. 119, no. 7, pp. 1680–1688, (2014). DOI: 10.1002/2013JE004576.

[16] Audouard, J. et al. "*Analysis of Curiosity surface temperature Data*". In: *European Planetary Science Congress*, EPSC2015–300, (2015).

[17] Martínez, G.M. et al. "*Likely frost events at Gale crater: Analysis from MSL/REMS measurements*". In: *Icarus*, vol. 280, pp. 93–102, (2016).

[18] Mischna, Michael et al. "*Results from the Rover Environmental Monitoring Station (REMS) on Board the Mars Science Laboratory*". In: *Fifth international workshop on the Mars atmosphere: Modelling and observations*, (2014).

[19] Zorzano, M.-P. et al. "*Near surface temperature monitoring by the Air Temperature Sensor (ATS) on the Mars Science Laboratory Rover at Gale crater*". In: *Planetary and Space Science,* (2013).

[20] Zorzano, M.-P., Lepinette, A. and et al. "*Air temperature measurements with the Rover Environmental Monitoring Station (REMS). Air Temperature Sensor for Mars*". In: *Planetary and Space Science,* (2009).

[21] Mueller, D.W. and Abu-Mulaweh, H.I. "*Prediction of the temperature in a fin cooled by natural convection and radiation*". In: *Applied Thermal Engineering,* vol. 26, no. 14, pp. 1662–1668, (2006).

[22] Vautard, R. and M. Ghil. "*Singular spectrum analysis in nonlinear dynamics, with applications to paleoclimatic time series*", In: Physica D, vol. 35, pp. 395-424, (1989).

[23] Klema, V. and Laub, A. "*The singular value decomposition: Its computation and some applications*," In: *IEEE Transactions on Automatic Control*, vol. 25, no. 2, pp. 164-176, (1980), doi: 10.1109/TAC.1980.1102314.

[24] Rui, Wang and Hong-Guang, Ma and Guo-Qing, Liu and Dong-Guang, Zuo. "*Selection of window length for singular spectrum analysis".* In: *Journal of the Franklin Institute,* vol. 352, (2015).

[25] Elsner, J.B., and Tsonis, A.A. "*Singular Spectrum Analysis, A new Tool in Time Series Analysis*", Plenum Press: New York and London, (1996).

[26] Ghil, M. et al. "*Advanced Spectral Methods for Climatic Time Series*". In: *Reviews of Geophysics*, vol. 40, no. 1, (2002).

[27] Collins, M. and Lewis, S. R. and Read, P. L., and Hourdin, F. "*Baroclinic wave transitions in the Martian atmosphere*". In: *Icarus*, vol. 120, pp. 344–357, (1996) https://doi.org/10.1006/icar.1996.0055.

[28] Zurita-Zurita, S. and de la Torre Juárez, M. and Newman, C. E. and Viúdez-Moreiras, D. and Kahanpää, H. T. and Harri, A.-M., et al. "*Mars surface pressure oscillations as precursors of large dust storms reaching Gale".* In: *Journal of Geophysical Research: Planets*, vol. 127, (2022). https://doi.org/10.1029/2021JE007005.

[29] Kondrashov, D. and Shprits, Y. and Ghil, M. “*Gap filling of solar wind data by singular spectrum analysis*”. In: *Geophysical Research Letters*, vol. 37, no. 15, (2010). https://doi.org/10.1029/2010gl044138

[30] Price, John F. “*Fourier techniques and applications”,* (1985).

[31] Morlet, J and Arens, G and Fourgeau, E. and Glard, D. “*Wave propagation and sampling theory—Part I: Complex signal and scattering in multilayered media”. In:* GEOPHYSICS, vol. 47, no. 2, pp. 203-221, (1982).

[32] Morlet, J and Arens, G and Fourgeau, E. and Glard, D. “*Wave propagation and sampling theory—Part II: Sampling theory and complex waves”. In:* GEOPHYSICS, vol. 47, no. 2, pp. 222-236, (1982).

[33] Meyer, Y. *Ondelletes et fonctions splines, Seminaire Equations aux Dérivées Partielles*, Ecole Polytechnique, Paris, France, (1986).

[34] Mallat, Stephane G. “*Multiresolution approximations and wavelet orthonormal bases of L^2 (R)*”. In: *Trans. Amer. Math. Soc*, vol. 315, pp. 69-87, (1989).

[35] Mallat, Stéphane. “*A wavelet tour of signal processing*”. In: *Elsevier*, (1999).

[36] Galvez, Maria Cecilia et al. “*Wavelet Signal Denoising Applied to Multiwavelength-Depolarization White Light LIDAR Measurement*”. In: *Proceedings of the 23rd International Laser Radar Conference*, (2006).

[37] Donoho, David and Johnstone, I. “*Ideal Spatial Adaptation by Wavelet Shrinkage”*. In: *Biometrika*, vol. 81, pp. 425–455, (1993).

[38] Somekawa, Toshihiro et al. “*Noise Reduction in White Light Lidar Signal Using a One-Dim and Two-Dim Daubechies Wavelet Shrinkage Method*”. In: *Remote Sensing,* (2013).

[39] Daubechies, I. “*Orthonormal bases of compactly supported wavelets*”. In: Communications on Pure and Applied Mathematics, vol. 41, pp. 909-996, (1988). https://doi.org/10.1002/cpa.3160410705.

[40] Shyh-Jier Huang and Cheng-Tao Hsieh, "*Coiflet wavelet transform applied to inspect power system disturbance-generated signals*". In: *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 1, pp. 204-210, (2002), doi: 10.1109/7.993240.

[41] Huang, N.E and et al. *“The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis”* In: *Royal Society*, vol. 454, (1996).

[42] Rilling, Gabriel and Flandrin, Patrick and Gonçalves, Paulo. “*On empirical mode decomposition and its algorithms”*. IEEE-EURASIP workshop, (2003).

[43] Huang, N. E. “*Hilbert-Huang transform and its applications*”, In: *World Scientific*, vol. 16, (2014).

[44] Boudraa, Abdel-O and Cexus, Jean-Christophe. “*EMD-Based Signal Filtering*”. In: *IEEE T. Instrumentation and Measurement,* vol. 56, pp. 2196–2202, (2007), DOI: 10.1109/TIM.2007.907967.

[45] Boudraa, Abdel-O and Cexus, Jean-Christophe. “*Denoising via empirical mode decomposition*”. In: *Second International Symposium on Communications, Control and Signal Processing,* (2006).

[46] Boudraa, Abdel-O, Cexus, Jean-Christophe, and Saidi, Zazia. “*EMD-Based Signal Noise Reduction*”. In: *Signal Processing,* vol. 1, (2005).

[47] Donoho, David. “*Denoising by Soft-Thresholding*”. In: *IEEE Transactions on Information Theory,* vol. 41, no. 03, pp. 613–627, (1995).

[48] Hess, S.L., Ryan, J.A., Tillman, J.E., Henry, R.M. and Leovy, C.B., “*The annual cycle of pressure on Mars measured by Viking Landers 1 and 2*”. In: *Geophys. Res. Lett.*, vol. 7, (1980), https://doi.org/10.1029/GL007i003p00197.

[49] Spiga, A. and Forget, F. and Dolla, B. and Vinatier, S. and Melchiorri R. and Drossart, P. and Gendrin, A. and Bibring, J.-P. and Langevin, Y., and Gondet, B., “*Remote sensing of surface pressure on Mars with the Mars Express/OMEGA spectrometer: 2. Meteorological maps*”, In: *J. Geophys. Res.*, vol. 112, (2007), doi:10.1029/2006JE002870.

[50] Kalman, R. E. "*A New Approach to Linear Filtering and Prediction Problems.*" In: *ASME. J. Basic Eng.*, vol. 82, (1960).

[51] Huang, Norden E. and Shen, Zheng and Long, Steven R.,” *A new view of nonlinear water waves: the Hilbert spectrum”.* Annual Review of Fluid Mechanics, vol. 31, pp. 417-457, (1999).

[52] Vautard, Robert and Yiou , Pascal and Guil, Michael. “*Singular Spectrum Analysis: A toolkit for short, noisy chaotic signal”.* In: *Physica D*, vol. 95, no. 126, (1992).

[53] Tillman, J. E. “*Mars global atmospheric oscillations: Annually synchronized transient normal-mode oscillations and the triggering of global dust storms*”. In: *J. Geophys. Research*, (1988).

[54] Zurita-Zurita, S. et al. “*Denoising Atmospheric Temperature Measurements Taken by the Mars Science Laboratory on the Martian Surface*”. In: *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–10, (2021),

DOI: 10.1109/TIM.2020.3034986.

[55] Thin Film Platinum RTDs specifications. URL: http://bit.ly/32PZxqn.

[56] Proakis, John G and Manolakis, Dimitris K. *“Digital signal processing: principles algorithms and applications*”. In: *Pearson Higher Education*, (2013).

[57] S. Zurita, “*Denoising temperature on martian surface dataset*”, (2020). [Online]. Available: http://dx.doi.org/10.21227/ban4-km41.

[58] Barros, J., Diego, R. and Apraiz, Matilde. “*Applications of Wavelet Transform for Analysis of Harmonic Distortion in Power Systems: A Review*”. In: *IEEE Transactions on Instrumentation and Measurement,* vol. 61, pp. 2604–2611, (2012).

DOI: 10.1109/TIM.2012.2199194.

[59] Driesen, J. L. J. and Belmans, R. J. M. “*Wavelet-Based Power Quantification Approaches*”. In: *IEEE Transactions on Instrumentation and Measurement,* vol. 52, no. 4, pp. 1232–1238, (2003).

[60] Ramos, Pablo and Ruisanchez, Itziar. “*Noise and background removal in Raman spectra of ancient pigments using wavelet transform*”. In: *Journal of Raman Spectroscopy*, vol.36, no. 848-856, (2005).

[61] Seok, Jong Won and Bae, Keun Sung. “*Speech enhancement with reduction of noise components in the wavelet domain*”. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing,* (1997).

[62] Tan, Hu-Wei and Brown, Steven. “*Wavelet analysis applied to removing non-constant, varying spectroscopic background in multivariate calibration*”. In: *Journal of chemometrics*, vol. 16, no. 228-240, (2002).

[63] Vasudha, N. and Sundararajan, N. “*Detection of Discontinuity in ECG using Wavelet transform*”. In: *Applied Mathematical Sciences*, vol. 6, no. 117, (2012).

[64] Yoon, Weon-Ki and Devaney, M. J. “*Power Measurement Using the Wavelet Transform*”. In: *IEEE Transactions on Instrumentation and Measurement*, vol. 47, no. 5, pp. 1205–1210, (1998).

[65] Yoon, Weon-Ki and Devaney, M. J. “*Reactive Power Measurement Using the Wavelet Transform*”. In: *IEEE Transactions on Instrumentation and Measurement,* vol. 49. no. 2, pp. 246–252, (2000).

[66] Wu, Zhaohua and Huang, Norden E. “*Ensemble Empirical Mode Decomposition: A Noise-Assisted Data Analysis Method*”. In: *Advances in Adaptive Data Analysis*, vol. 01, no. 01, pp. 1–41, (2009).

[67] Yan, R. and Gao, R. X. “*Hilbert-Huang Transform-Based Vibration Signal Analysis for Machine Health Monitoring*”. In: *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 6, pp. 2320–2329, (2006).

[68] Agbinya, J. I. "*Discrete wavelet transform techniques in speech processing,*" In: Proceedings of Digital Processing Applications (TENCON '96), vol. 2, pp. 514-519, (1996) doi: 10.1109/TENCON.1996.608394.

[69] Popov, Dmitry and Gapochkin, Artem and Nekrasov, Alexey. “*An Algorithm of Daubechies Wavelet Transform in the Final Field When Processing Speech Signals*”. In: *Electronics,* vol. 7, (2018).

[70] Thakral, S., Manhas, P. “Image Processing by Using Different Types of Discrete Wavelet Transform”. In: *Springer*, vol. 955, (2018).

[71] Walczak, B. and Massart, D.L “*Noise suppression and signal compression using the wavelet packet transform”*. In: *Elsevier,* vol. 26, (1997).

[72] Zorzano, M.-P., Soria-Salinas, Álvaro et al. “*Wind retrieval from temperature measurements from the Rover Environmental Monitoring Station, Mars Science Laboratory*”. In: *Icarus*, (2020).

[73] Sharma, L. and Dandapat, S. and Mahanta, A. *“ECG signal denoising using higher order statistics in Wavelet subbands*”. In: *Biomed. Signal Process Control*, vol. 5, pp. 214–222, (2010).

[74] Cooper, B., de la Torre Juárez, M., Mischna, M., Lemmon, M., Martínez, G., Kass, D., et al. *“Thermal forcing of the nocturnal near surface environment by martian water ice clouds”*. In: *Planets: Journal of Geophysical Research*, vol. 126, (2021).

Annex 1: Code

1. READING ENVRDR and MODRDR FILES (PS and ATS data)

```
import datetime as dt
import matplotlib
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import scale as mscale
from matplotlib import transforms as mtransforms
from matplotlib.ticker import FixedLocator
import numpy as np
import os

#esta clase se usa para poder imprimir los ejes x
class SegmentedScale(mscale.ScaleBase):
    name = 'segmented'

    def __init__(self, axis, **kwargs):
        mscale.ScaleBase.__init__(self, axis)
        self.points = kwargs.get('points',[0,1])
        self.lb = self.points[0]
        self.ub = self.points[-1]

    def get_transform(self):
        return self.SegTrans(self.lb, self.ub, self.points)

    def set_default_locators_and_formatters(self, axis):
        axis.set_major_locator(FixedLocator(self.points))

    def limit_range_for_scale(self, vmin, vmax, minpos):
        return max(vmin, self.lb), min(vmax, self.ub)

    class SegTrans(mtransforms.Transform):
        input_dims = 1
        output_dims = 1
        is_separable = True

        def __init__(self, lb, ub, points):
            mtransforms.Transform.__init__(self)
```

```
            self.lb = lb
            self.ub = ub
            self.points = points

        def transform_non_affine(self, a):
            masked = a # ma.masked_where((a < self.lb) | (a > self.ub),
a)
            return np.interp(masked, self.points,
np.arange(len(self.points)))

        def inverted(self):
            return SegmentedScale.InvertedSegTrans(self.lb, self.ub,
self.points)

    class InvertedSegTrans(SegTrans):

        def transform_non_affine(self, a):
            return np.interp(a, np.arange(len(self.points)),
self.points)
        def inverted(self):
            return SegmentedScale.SegTrans(self.lb, self.ub,
self.points)

# Now that the Scale class has been defined, it must be registered so
# that ``matplotlib`` can find it.
mscale.register_scale(SegmentedScale)


colores=['blue',
'green','red','yellow','orange','cyan','m','k','olive','navy','peru',
'black','pink', 'lime', 'violet','skyblue','coral','gold', 'purple']
solPath = 'F:/sols_2000_2249'
solPath='C:/Users/zurit/Documents/doctorado/ats_signal_processing/tesi
s/RDR-sol3483-1'
my_xticks=['00','01','02','03','04','05','06','07','08','09','10','11'
,'12','13','14','15', '16','17','18','19','20','21','22','23','24']


### READ MODRDR FILEs for PS #####
list_of_rems_files = []
list_of_rems_files_temp =[]
lastVersion = []
for root, dirs, files in os.walk(solPath):
```

```
    for file in files:
        if  'RMD' in file and '.TAB' in file and '3483' in file:
            list_of_rems_files.append(root+'/'+file)
        if  'RNV' in file and '.TAB' in file and '3483' in file:
            list_of_rems_files_temp.append(root+'/'+file)

for i in range(0,len(list_of_rems_files)):
    solstr                                                              =
      list_of_rems_files[i].split('/')[8].partition('RMD')[2][0:4]
    print(solstr)



    time  =  [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]  #  keep
information  about  if  there  measurements  each  hour.  If  not,  do  not  use
the sol
    with open ( list_of_rems_files[i], 'r') as f:

            aobt = []
            time1= []
            psdata = []

            for line in f:
                s = line.split(',')

               #
labelTime.append((s[1].partition("M")[2].partition(".")[0]))
                if s[37].find('UNK')<0:
                    psdata.append(float(s[37]))
                    aobt.append(float(s[0]))
                    if len(aobt)==1:
          pepe = s[1].partition('M')[2].partition('.')[0]
                        counter          =          int(pepe[0:2])*3600
+int(pepe[3:5])*60 + int(pepe[6:8])
                    if len(aobt)>=2:
                        if aobt[-1]-aobt[-2]==1:
                            counter = counter + 1
                        else:
                            counter = counter+int((aobt[-1]-aobt[-2]))



 time1.append(counter)
```

```
    #ONLY WORKS WELL IF THE SAME DOWNLOADED FREQUENCY IS IN THE ALL FILE. genracion del vector eje x de raw
    for l in range(len(time)):
        if time[l]==0 and l!=0:
            time[l]= l*3699


    fig, ax = plt.subplots( figsize = (20,12))
    plt.plot(time1, psdata, color = 'blue', label = 'REMS data ', linewidth = 3, marker = '.', linestyle='None')
    #plt.plot(time1, psdata, color = 'blue', label = 'REMS data ', linewidth = 1, marker = '.')
    plt.gca().set_xscale('segmented', points = time)
    plt.xticks(time, my_xticks, fontsize = 29)
    plt.yticks(fontsize = 35)
    plt.grid(True)
    plt.title('Pressure Sol %s'%solstr, fontsize = 45)
    plt.xlabel("LMST hour", fontsize = 45)
    plt.ylabel("Pressure (Pa)", fontsize = 45)
    plt.savefig('ps_sol%s.png'%solstr,  format='png',  dpi=600)




" READ RNV FILEs for ATS #####"
for i in range(0,len(list_of_rems_files_temp)):
    solstr = list_of_rems_files_temp[i].split('/')[8].partition('RNV')[2][0:4]
    print(solstr)

    time = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
    with open ( list_of_rems_files_temp[i], 'r') as f:
            timeTemp = []
            temperature = []
            aobt =[]
            for line in f:
                s = line.split(',')
                if s[29].find('UNK')<0:          ##BOOM1_TIP_AIR_TEMP
                    temperature.append(float(s[29]))
                    aobt.append(float(s[0]))
                    if len(aobt)==1:
                        pepe = s[1].partition('M')[2].partition('.')[0]
                        counter = int(pepe[0:2])*3600 + int(pepe[3:5])*60 + int(pepe[6:8])
```

```
                    if len(aobt)>=2:
                        if aobt[-1]-aobt[-2]==1:
                            counter = counter + 1
                        else:
                        counter = counter + int((aobt[-1]-aobt[-2]))


#time.append(dt.datetime.strptime(s[1].partition("M")[2].partition(".")[0],'%H:%M:%S'))
                    timeTemp.append(counter)
    for l in range(len(time)):
        if time[l]==0 and l!=0:
            time[l]= l*3699

    fig, ax = plt.subplots( figsize = (20,12))
    plt.plot(timeTemp, temperature, color = 'blue', label = 'REMS data', linewidth = 2,  marker = '.', linestyle='None')
    plt.gca().set_xscale('segmented', points = time)
    plt.xticks(time, my_xticks, fontsize = 29)
    plt.yticks(fontsize = 35)
    plt.grid(True)
    plt.title('Temperature Sol %s'%solstr, fontsize = 45)
    plt.xlabel("LMST hour", fontsize = 45)
    plt.ylabel("Temperature (K)", fontsize = 45)
    plt.savefig('temp_sol%s.png'%solstr,  format='png',  dpi=600)
```

2. Printing figures (Data after SSA processing)

```
file = open(MCAMpath,'r')
data = [esta.strip().split() for esta in file.readlines() ]
ptau = np.array(data[:-1],dtype= 'float')


f = open('filecominMaxv1.txt','r')
file = [linea.strip().split(' ')[0:4] for linea in f.readlines()]
f.close()
hours = np.array(file, dtype='float')[:,3]
sols = np.array(file, dtype='float')[:,0][hours==24]
mintemp = np.array(file, dtype='float')[:,1][hours==24]
maxtemp = np.array(file, dtype='float')[:,2][hours==24]
```

```
fig, ax = plt.subplots( figsize = (20,12))

plt.xticks(fontsize = 29)

plt.yticks(fontsize = 35)

salida = x0-rc

plt.xlabel("Sols", fontsize = 45)

plt.ylabel("Temperature (K)", fontsize = 45)

#l1= plt.plot(sols[2080:2405], pm[2080:2405], color = 'green', label =
'min-mean(min)', linewidth = 2,  marker = '.')#, linestyle='None')

#l1= plt.plot(sols[1376:1767], pm[1376:1767], color = 'green', label =
'min-mean(min)', linewidth = 2,  marker = '.')#, linestyle='None') storm
my33

#l2= plt.plot(solsv1[2250:2600], rc[2250:2600], color = 'blue', label =
'rc from min', linewidth = 2,  marker = '.')#, linestyle='None')

#l3= plt.plot(solsv1[2250:2600], rc[2250:2600], color = 'blue', label =
'rc', linewidth = 2,  marker = '.')#, linestyle='None')

#l3= plt.plot(solsv1[1500:1900], rc[1500:1900], color = 'blue', label =
'rc', linewidth = 2,  marker = '.')#, linestyle='None') storm year 33

l1= plt.plot(solsv1[2950:3210], rc[2950:3210], color = 'blue', label =
'rc', linewidth = 2,  marker = '.')#, linestyle='None') storm year 33


#l1= plt.plot(sols, pm, color = 'green', label = 'min-mean(min)',
linewidth = 2,  marker = '.')#, linestyle='None')

#l2= plt.plot(solsv1, rc, color = 'blue', label = 'reconstruction',
linewidth = 2,  marker = '.')#, linestyle='None')

#l3= plt.plot(solsv1, salida, color = 'orange', label = 'x0-rc',
linewidth = 2,  marker = '.')#, linestyle='None')

plt.xlim()

#l1 = plt.plot(sols, mintemp, color = 'blue', label = 'min temperature
', linewidth = 2,  marker = '.')#, linestyle='None')

#l11 = plt.plot(sols, maxtemp, color = 'green', label = 'max temperature
', linewidth = 2,  marker = '.', linestyle='None')

ax1 = ax.twinx()

#l3=ax1.plot(ptau[:,0],ptau[:,2], 'v', markeredgewidth = .55, markersize
= 6, fillstyle='none', color='orange', label='MCAM tau')

ax1.set_ylim(0,3)

ax1.set_ylabel("tau MCAM", fontsize = 35)

ax1.set_yticks([1,2,3])
```

```
ax1.set_yticklabels([1,2,3], fontsize=35)


 # leyenda con los datos en dos ejes distintos
lns = l1 + l2 +l3
labs = [l.get_label() for l in lns]
ax.legend(lns, labs, loc=0, fontsize=29)
plt.show()
```

3. SSA code

```
""" aplication of SSA method to Mintemp or/and MAxtemp"""

f = open(fileREMSminmax,'r')
file = [linea.strip().split(',')[0:4] for linea in f.readlines()]
f.close()
""" para rems"""
hours = np.array(file, dtype='float')[:,3]
sols = np.array(file, dtype='float')[:,0][hours==24]
mintemp = np.array(file, dtype='float')[:,1][hours==24]
maxtemp = np.array(file, dtype='float')[:,2][hours==24]



dtmin = (sols[1:]-sols[:-1]).min()

dsols = sols[((sols[1:]- sols[:-1])<=dtmin*1.001).nonzero()[0]]+0.5
t = np.arange(sols.min(), sols.max()+1,dtmin)
#dt = np.arange(dsols.min(), dsols.max()+1,dtmin) # el salto de fase solo existe cuando está un sol y el anterior

differenece= maxtemp-mintemp
pm= mintemp -mintemp.mean()
s =[]
N0 = int(sols[-1] - sols[0] + 1) #para rems
#N0 = int(sols[-1]+1) #para  MEDA

solsv1 = []
for c in range(N0): #para rems
    solsv1.append(c)
#for c in range((int(sols[-1]+1))): #para meda
#    solsv1.append(c)
```

```
dtmax = (sols[1:]-sols[:-1]).max()
M0 = 45
rc = np.zeros(N0)
l = 0
dM = int(M0/2)
adapta = 1
while l <20:
    if l>2:
        if ((s[-2]-s[-1])/s[-2])<0.001 and s[-1]<s[-3]:
            #if adapta and (M0-dM) > 2*dtmax:
            dM = int(dM/2)


    x0 = pm[0]
    j1 =0
    for k in range (1,(sols.size - 1)):
        x0 = np.hstack([x0,pm[k]])
        j1 +=1
        for j in np.arange(sols[k] + dtmin, sols[k+1], dtmin):
            x0 = np.hstack([x0,rc[j1]])
            j1 +=1
    x0 = np.hstack([x0,pm[sols.size - 1]])
    xm = x0.mean()
    x0 = x0 - xm
    matrix0 = [x0[0:M0]]
    for i in range (1,N0-M0):
        matrix0.append(x0[i:i+M0])
    matrix0 = np.array(matrix0) #PARA REMS
    #matrix0 = np.array(matrix0[0:324]) #meda
    c0 = np.corrcoef(matrix0.transpose())
    evals0, evecs0 = la.eigh(c0)
    ii = evals0.argsort()[::-1] ## indices de mayor a menor
    v = evecs0[:,ii]
    e = evals0[ii]
    numAutovect = (e.cumsum()/(e.sum())>0.95).nonzero()[0][0]# número
de autovectores que debo coger
    #cp = np.matmul(c0,v)
    cp = np.matmul(matrix0,v) #cálculo de componentes principales


    #reconstruction de la evolucion temporal proyectada sobre los
numAutovect
```

```
    r=[cp[0,0]*v[0,0]]
    nn = N0-M0
    #nn = 3016
    for i in range(1,M0):
        r.append((cp[i::-1,0]*v[:i+1,0]).sum()/(i+1))
    # pc[:,0] tiene la evolucion temporal asociada al autovalor e[0]
hasta el tiempo M (una ventana de %d puntos)\n ahora vamos con el tiempo
de M a N-M+1."%(m))
    for i in range(M0,nn):
        r.append((cp[i:i-M0:-1,0]*v[:M0,0]).sum()/M0)
        #    Ahora para los valores de N-M+1 (=%d/%d) a N (=%d)"%(nn,n0-
m,n0))
    for i in range(nn,(t.size)):
        r.append((cp[nn-1:i-M0:-1,0]*v[i-(nn-
1):,0]).sum()/np.min([i,(N0-i)]))
    r=[r]
    #      Prepara una segunda fila con el siguiente autovector
    for k in range(1,numAutovect +1):
        rk=[cp[0,k]*v[0,k]]
        for i in range(1,M0):
            rk.append((cp[i::-1,k]*v[:i+1,k]).sum()/(i+1))
        #    a[:,k] tiene la evolucion temporal asociada al autovalor
e[k] hasta el tiempo M (una ventana)\n ahora vamos con el tiempo de M a
N-M+1.
        for i in range(M0,nn):
            rk.append((cp[i:i-M0:-1,k]*v[:M0,k]).sum()/M0)
        #  Ahora para los valores de N-M+1 a N")
        for i in range(nn,(t.size)):
            rk.append((cp[nn-1:i-M0:-1,k]*v[i-(nn-
1):,k]).sum()/np.min([i,(N0-i)]))
        r=r+[rk]
    r = np.array(r)
    rc = sum(r)
    s.append(np.std(rc-x0))#[:len(rc)]))
    l +=1
```

4. **HHT code**

```
def softEMD (dataDegree):

    imf_umbralised_std = [] ; reconstructed_signal_emd =[]
    startTime = time()
```

```
    decomposer = EMD(dataDegree, nbsym=5)
    imf = decomposer.decompose()


    cmse = []
    for g in range(len(imf)-2):
        cmse.append((1/len(imf[g]))*sum(imf[g]**2))
    argMin = np.argmin(cmse)
    if argMin == 0:
        argMin = 1
    for r in range ( 0, argMin):
        sigma = np.median(np.abs(imf[r]-np.median(imf[r])))/0.6745
        thresholdEMD = sigma*math.sqrt(2*math.log(len(imf[r])))

        imf_umbralised_std.append([])
        for g in range (0,len(imf[r])):
            if abs(imf[r][g]) >= thresholdEMD:

imf_umbralised_std[r].append(np.sign(imf[r][g])*(np.abs(imf[r][g])-
thresholdEMD))
            else:
                imf_umbralised_std[r].append(0)
    """  add residual signal """
    for r in range(argMin, len(imf)):
        imf_umbralised_std.append(imf[r].tolist())

    reconstructed_signal_emd = [sum(x) for x in
zip(*imf_umbralised_std)]
    varianceEmd = np.std(reconstructed_signal_emd)

    return varianceEmd, reconstructed_signal_emd
```

5. **Wavelet code**

```
# split in 5 minutes sessions
def session_split(timestamp):
    MARGIN_SESSION = 40
  #  NUM_SAMPLES = 308
    index_start = 0
    index_ini = []
```

```
    index_fin = []


    for i in range((len(timestamp)-1)):

        if ((timestamp[i+1] - timestamp[i]) >= MARGIN_SESSION  or
i == range(len(timestamp)-1)):

            index_ini.append(index_start)

            index_start = i+1;

            index_fin.append(i)

    index_ini.append(index_start)

    index_fin.append(i+1)

    return  (index_ini,  index_fin)  #devolvemos  los  índices  de
inicio y fin


def softWavelet (dataDegree):

    n = 0

    family = 'coif4'

    base = pywt.Wavelet(family)

    coeffs = pywt.wavedec(dataDegree, base, mode = 'symmetric')#,
axis = 0)


    # Apply it to the signal and get the wavelet coefficients

    thresholdedCoeffs = copy.deepcopy(coeffs)

    """calculo del umbral """

    sigma    =    np.median(np.abs(coeffs[-1]-np.median(coeffs[-
1])))/0.6745

    threshold = sigma* math.sqrt(2*math.log(len(dataDegree)))

    """ buscar en qué listas de coeff no se aplica el threshold
"""

    for p in range(len(coeffs)-1):

        meanCoeff = np.mean(coeffs[p])

        if (np.abs(meanCoeff)> threshold/10 and len(coeffs) > 1):

            n= n + 1

    """ no se aplica el th a las primeras 'n' listas, que contienen
componente de contínua  """

    for level in range(n, len(coeffs)):

        for g in range (0,len(coeffs[level])):
```

```
            if abs(coeffs[level][g]) < threshold:
                thresholdedCoeffs[level][g] = 0
            else:
                 thresholdedCoeffs[level][g]=
np.sign(coeffs[level][g])*(np.abs(coeffs[level][g])-threshold)

  reconstructed_signal_wavelet = pywt.waverec(thresholdedCoeffs,
base, axis=0)
    return (reconstructed_signal_wavelet)
```

6. Generate the min temperature file (temperature intermediate file)

```
" Calculating min max values from ATS RNV #####"
### READ RNV FILEs for TMEP #####
list_of_rems_files = []
list_of_rems_files_temp =[]
lastVersion = []
for root, dirs, files in os.walk(solPath):
    for file in files:
        if  'RNV' in file and '.TAB' in file:
            list_of_rems_files_temp.append(root+'/'+file)

for i in range(0,len(list_of_rems_files_temp)):
    solstr =
list_of_rems_files_temp[i].split('/')[2].partition('RNV')[2][0:4
]
    isTimethere =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] # keep
information about if there measurements each hour. If not, do not
use the sol
    sols=[]
    with open ( list_of_rems_files_temp[i], 'r') as f:
            temperature = []
            token =[]
            sclkvector= []
```

```
        for line in f:
            s = line.split(',')
            if s[29].find('UNK')<0:
##BOOM1_TIP_AIR_TEMP
                sclkvector.append(float(s[0]))
                temperature.append(float(s[29]))
                lts = s[2].partition(' ')[2].split(':')
                if len(lts)== 1: # if LTST not appearing,
using timestamp and convert to ltst
                    sclk = s[0].strip()
                    et = spice.scs2e(MSL_SCID, str(sclk))
                    out = spice.sce2s(LMST_SCID, et)
                    out = out.split('/')[1].split(':')
                    positionR, lt = spice.spkpos(MSL_LS,
int(sclk),'IAU_MARS', 'NONE', 'MARS')
                    positionP = spice.reclat(positionR)
                    CuriositySitePosition = positionP[1]
                    out1 = spice.et2lst(et,
mars,CuriositySitePosition, 'PLANETOCENTRIC')
                    lts = (out[0]+' '+out1[3]).partition('
')[2].split(':')
                if isTimethere[int(lts[0])]==0:
                    isTimethere[int(lts[0])] = 1


        #split the sessions
        index1, index2 = session_split(sclkvector)


        #wavelet aplication to tip data
        wavetemperature=[]
        for h in range (len(index1)):
            out =
softWavelet(temperature[index1[h]:index2[h]])
            wavetemperature = wavetemperature + out.tolist()
```

```
            token.append(solstr)
            token.append(str(min(wavetemperature)))
            token.append(str(max(wavetemperature)))
            token.append(str(sum(isTimethere)))
            f = open ('fileMinMaxTempWithWavelet.txt','a')
            for i in token:
                f.write(i)
                f.write(',')
            f.write('\n')
            f.close()
```

7. Generate the p_tides file (pressure intermediate file)

```
import numpy as np
from scipy.optimize import leastsq
import os
import matplotlib.pyplot as plt
import spiceypy as spice
spice.furnsh("C:/Users/zurit/ownCloud/pythonCodev2/SPICE_kernels_MSL/kernels.msl_ops120808_v1")
n = 12
Ls =[]
t = 668.45
MSL_SCID = -76
LMST_SCID=-76900
MSL_LS= "MSL_LANDING_SITE"
mars = 499
sclk_sol_1 = 397535243.0
et_seconds_landing =397446666.183000

def func (p,x,n):
    total = 0
    for i in range (1, n+1):
        total = total +p[2*i1]*np.sin((x/1440*i+p[2*i])*2*np.pi)
    total = total + p[0]
```

```
    return   total



positionR = np.array((1,1,1), dtype= 'float')

lt = 1.0

positionR,  lt  =  spice.spkpos(MSL_LS,  sclk_sol_1,'IAU_MARS',
'NONE', 'MARS')

#positionR,  lt  =  spice.spkpos(MSL_LS,  sclk_sol_1,'IAU_MARS',
'NONE', 'MARS')

positionP = spice.reclat(positionR)

CuriosityLandingSitePosition = positionP[1]

solPath ='C:/Users/zurit/Documents/REms/Manuel_mareasMarte/sols'

ptidesPath = 'p_tidesversion.txt'

initialLs  =  np.array([160.948,  360.001,-10.6935,  1.24961,-
0.631483,-0.00890294])


#read MODRDR files

list_of_rems_files_all_versions = []

list_of_rems_files_last_version = []

lastVersion = []

for root, dirs, files in os.walk(solPath):

    for file in files:

        if  'RMD' in file and '.TAB' in file:

           list_of_rems_files_all_versions.append(root+'/'+file)


#  filter  from  list_of_rems_files_all_versions  only  the  last
version from each sol

i =0

solList =[]

while i<len(list_of_rems_files_all_versions):


#calcule Sol file

    solstr                                                       =
list_of_rems_files_all_versions[i].split('/')[7].partition('RMD'
)[2][0:4]

    if solstr in solList:
```

```
        i = i +1
    else:
        samesol = []
        version =[]
        solList.append(solstr)
        for j in range(len(list_of_rems_files_all_versions)):
            if
            list_of_rems_files_all_versions[j].split('/')[7].par
            tition('RMD')[2][0:4].find(solstr)>=0:
            samesol.append(list_of_rems_files_all_versions[j])
            version.append(int(list_of_rems_files_all_versions[j
            ].partition('.TAB')[0][-1:]))
        lastVersion.append(max (version))

list_of_rems_files_last_version.append(samesol[version.index(max
(version))])



for i in range(0,len(list_of_rems_files_last_version)):
    solstr                                                      =
list_of_rems_files_last_version[i].split('/')[7].partition('RMD'
)[2][0:4]
    print(solstr)
    x1 = int(solstr)/t
    #calcule Ls from Sol
    pls                                                         =
initialLs[0]+initialLs[1]*x1+initialLs[2]*(np.sin((x1+initialLs[
3])*2*np.pi))+initialLs[4]*(np.sin((2*x1+initialLs[5])*2*np.pi))
    isTimethere                                                 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]      #     keep
information about if there measurements each hour. If not, do not
use the sol
    with open ( list_of_rems_files_last_version[i], 'r') as f:
            ltstminutes = []
            psdata = []
            for line in f:
                s = line.split(',')
                if s[37].find('UNK')<0:
```

```
                    psdata.append(float(s[37]))
                    lts = s[2].partition(' ')[2].split(':')
                    if len(lts)== 1: # if LTST not appearing,
using timestamp and convert to ltst
                        sclk = s[0].strip()
                        et = spice.scs2e(MSL_SCID, str(sclk))
                        out = spice.sce2s(LMST_SCID, et)
                        out = out.split('/')[1].split(':')
                        positionR, lt = spice.spkpos(MSL_LS,
int(sclk),'IAU_MARS', 'NONE', 'MARS')
                        positionP = spice.reclat(positionR)
                        CuriositySitePosition = positionP[1]
                        out1 = spice.et2lst(et,
mars,CuriositySitePosition, 'PLANETOCENTRIC')
                        lts = (out[0]+' '+out1[3]).partition('
')[2].split(':')
                    if isTimethere[int(lts[0])]==0:
                        isTimethere[int(lts[0])] = 1
                    ltstminutes.append(int(lts[0])*60 +
int(lts[1]) + int(lts[2].replace('"',''))/60)
    p0= [800, 30, 0, -10, -0.5, -1, 0.5, 1, 0.1, 1, -0.1, 1, 0,
1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
    desv = lambda p,x,y,n:(func(p,x,n)-y)
    p = np.array(p0,dtype=float)
    x = np.array(ltstminutes,dtype=float)
    y = np.array(psdata,dtype=float)
    out, success = leastsq(desv, p , args=(x,y,n))
    Ls.append(pls)

    #update p_tides.txt file    when a new sol arrives
    outFile = open(ptidesPath, 'a')
    outFile.write(solstr + ','+ str(Ls[-1]) + ',')
    for j in range(len(out)):
        outFile.write(str(out[j])+',')
    outFile.write(str(sum(isTimethere)) + ',')
    outFile.write(str(lastVersion[i]) + ',')
```

```
outFile.write('\n')

outFile.close()
```

Annex 2: The m-model

The m-model [20] explains the model that generate the last processing level of the ATS data, the MODRDR product. Due to the similarities of the ATS to a cylinder thin fin, this Air Temperature Retrieval model is based on the Mueller theory [21], based on a thermal physics of a cylinder thin fin in equilibrium with the fluid around it. This mathematical model describes the procedure to retrieve the temperature of the fluid, T_f, around the ATS fin, using the three known temperatures T_a (temperature at the free end), T_b (temperature at the base) and T_{int} (temperature at the intermediate point) of the ATS.

The Mueller experiment explains that, for a known and fixed T_a, T_b and T_{int}, the temperature difference in the thin fin, with temperature value T_f, can be written as:

$$(T_a - T_f) = (T_b - T_f)\frac{1}{cosh\,(m)}$$

$$(T_{int} - T_f) = (T_b - T_f)\frac{cosh\, m\,\left(1 - \frac{1}{n}\right)}{cosh\, m},$$

where $m = L\sqrt{\frac{4h}{KD}}$ is an average value over the length, K is the thin fin conductivity, L and D, the length and the diameter of the cylinder and h the convection term.

This equation system is composed of two equations with two unknowns, T_f and m, which can be obtained by using the measured values of T_a, T_b and T_{int}.

Annex 3: Calibration data

Here we detail two sources of temperature measurement datasets that have been used to contrast and assess the goodness of the proposed approaches.

These two sources of data that, in principle, can be considered practically unaffected by the typical external perturbations are:

Cruise Checkout

The so-called ATS Cruise Checkout data, a vacuum health status check performed during the cruise phase to Mars, where the ambient temperature was estimated to be −10 ºC. Under these conditions, such data should be largely considered free from noticeable external perturbations and can be used for benchmarking purposes while processing the ATS noise.

Thermal Vacuum Test (TVT)

The so-called ATS TVT was performed before the mission launch. The test took place inside a vacuum chamber with a cooling screen placed at the boom base, and a surrounding cover refrigerated with liquid nitrogen. Under these conditions, intensive measurement tests were performed to check the electronic noise generated by the instrument at different target temperatures: 0 ºC, −30 ºC, −50 ºC, −70 ºC and −90 ºC.

The noise affecting the datasets mentioned above, which are processed using the previously detailed methods, stems by hypothesis from the system electronics the ATS. Under the described conditions, such data should be largely considered free from noticeable external perturbations and can be used for benchmarking purposes while processing the ATS noise.

Annex 4: Files generated

This annex shows two portions of the two intermediate files used in this thesis. The first file, the pressure file, shows the pressure modes (amplitude and phase) for each sol. The second file, the temperature file, shows the minimum and the maximum temperature values for each sol. The last column, in both files, shows the number of measurement periods taken per sol.

1. Extract from the pressure tides file:

Sol	Ls (Solar longitude)	p_0, pressure mean per day	p_1 amplitude	p_1 phase	p_2 amplitude	p_2 phase	Measurement periods per sol
1	150,816767	807,595812	35,1691542	-0,02156833	-2,45729327	-0,62591044	2
9	155,066242	765,437984	72,0019757	0,05803301	1,57072434	-0,48694841	11
10	155,601739	739,371927	34,6859153	0,06809742	12,4898258	-0,52646789	24
11	156,138206	740,133771	35,1320255	0,06444578	12,3447692	-0,51683297	24
12	156,675647	740,615911	34,645483	0,06247828	12,2472187	-0,52242216	24
13	157,214065	746,236909	42,9173483	0,0319504	17,6203556	-0,44385778	17
14	157,753462	739,465991	35,6179646	0,05561911	12,600955	-0,52252676	24
15	158,293842	740,098306	36,6588643	0,05426372	13,1045826	-0,52401879	24
16	158,835208	740,579751	37,9169257	0,05348745	13,080365	-0,51534696	24
17	159,377563	741,669954	37,0171896	0,04961273	13,7829129	-0,52379876	24
18	159,920909	767,438061	-16,9395814	0,08699567	-30,1979632	-0,73732337	12
19	160,465249	760,574599	67,4587062	0,02087528	-16,0176039	-0,82651926	12
20	161,010585	725,344767	40,0086696	0,17486875	-7,74031226	-0,69498251	15
21	161,55692	741,754443	38,1350221	0,06327448	14,110782	-0,52134194	24
22	162,104256	742,304274	40,378672	0,06755095	13,2594629	-0,51382708	24
23	162,652595	742,037214	39,4195663	0,06565386	13,7221021	-0,52198469	24
24	163,20194	742,893141	38,9125238	0,06664249	13,6964973	-0,51392713	24
25	163,752292	744,472892	38,3364381	0,06784434	13,5300421	-0,50492165	24
26	164,303653	744,850618	37,8736253	0,06232086	13,101672	-0,49782031	24
27	164,856025	743,852963	37,341603	0,06225883	12,0731803	-0,50900407	24
28	165,40941	745,273192	37,5226817	0,05892361	12,4667789	-0,51463742	24
29	165,963808	746,746181	37,9329928	0,05855668	13,019648	-0,52188918	24
30	166,519222	746,650363	38,1065776	0,05767368	13,3114616	-0,52321492	24
31	167,075653	746,686362	37,7724213	0,060884	13,5742057	-0,51306359	21
…							
3469	2024,77143	811,518006	91,7091173	-0,14618294	-74,5111314	-0,55899286	14

3470	2025,41166	840,325413	42,2692891	0,06419278	12,2663824	-0,56328035	24
3471	2026,0524	839,967318	38,6605758	0,05966608	14,090896	-0,54488202	24
3472	2026,69363	841,742002	41,4729171	0,06361415	14,6568566	-0,53295412	23
3473	2027,33535	843,046422	40,9836129	0,0615306	15,5730113	-0,53656937	23
3474	2027,97753	844,069085	39,8513361	0,06288424	14,7347197	-0,54945747	24
3475	2028,62017	845,620228	39,4158554	0,05608617	15,06797	-0,54931774	24
3476	2029,26325	845,682564	39,7153101	0,05302507	15,6017625	-0,54784823	24
3477	2029,90677	846,594197	39,6836354	0,061747	14,3385991	-0,5522516	24
3478	2030,55072	848,145316	39,4115427	0,05370898	14,7512448	-0,55470047	24
3479	2031,19507	850,564054	38,3199125	0,05151651	15,5033905	-0,55368057	24
3480	2031,83982	852,138862	40,4844263	0,05791967	15,5199052	-0,54723703	24
3481	2032,48497	854,501901	39,1401328	0,06424799	15,5782894	-0,5441407	24
3482	2033,13048	856,553955	41,2279968	0,063202	15,396396	-0,54257975	24
3483	2033,77636	855,410993	41,7605975	0,06057329	13,5840672	-0,55641117	24
3484	2034,42259	857,456845	49,8744374	0,0589836	19,1428655	-0,53356467	24
3485	2035,06916	858,542256	53,0127841	0,06548976	16,3599516	-0,53354365	24
3486	2035,71605	857,939627	47,8906657	0,06657192	15,0584253	-0,54227647	24
3487	2036,36326	860,240512	42,3050294	0,06671727	14,637008	-0,55271588	24
3488	2037,01077	861,355654	43,0147008	0,06507479	15,9020068	-0,55812811	24
3489	2037,65857	862,093409	41,678896	0,07420494	15,4091482	-0,5615148	24
3490	2038,30665	863,543314	40,1522357	0,06617729	15,1236846	-0,57159594	24
3491	2038,95499	864,425582	41,3330118	0,07104404	15,1269487	-0,55970491	24
3492	2039,60358	865,750253	38,08683	0,06037407	14,6719792	-0,58017854	22
3493	2040,25241	865,639448	37,8569168	0,06809855	15,6532494	-0,5500359	24
3494	2040,90147	866,670838	38,4642663	0,06478423	15,4280331	-0,54409357	24
3495	2041,55074	831,016779	80,3891591	-0,04159701	-19,3858942	-0,55187593	13

2. Extract from the temperature file:

Sol	Min temp	Max temp	Measurement periods per sol
1	273,110242	276,269153	2
9	207,999804	272,199453	11
10	198,768539	271,230968	24
11	197,23071	272,33678	24
12	196,984329	271,948891	24
13	199,334475	272,242046	17
14	199,438439	271,548791	24
15	195,552618	273,694644	24

16	197,013167	273,306255	24
17	197,035328	269,356696	24
18	200,330399	251,809366	12
19	212,090423	269,99745	12
20	205,219458	271,785547	15
21	199,283075	272,803033	24
22	198,828429	274,539456	24
23	198,623983	270,104109	24
24	198,13767	274,849118	24
25	197,949504	274,585416	24
26	197,704082	271,290683	24
27	198,36549	266,514823	24
28	198,228318	266,58086	24
29	198,111449	271,071681	24
30	199,481946	270,193032	24
31	199,820394	249,775985	12
32	211,436306	271,294	12
33	199,91262	271,046484	24
34	200,172568	273,783748	23
35	200,583444	271,541242	24
36	200,1875	271,879151	24
37	199,857762	272,852822	24
38	200,600317	276,171734	24
39	198,311104	275,562717	24
40	198,528923	275,006209	24
41	198,562101	275,600877	24
42	198,695429	277,964559	24
….			
3408	207,010753	263,513212	24
3409	208,273867	261,372551	21
3410	206,647206	261,893632	24
3411	207,034788	264,699726	24
3412	208,099659	261,497153	24

3413	208,919753	266,543267	24
3414	205,498365	266,866931	24
3415	204,68965	270,693003	24
3416	206,009976	272,730718	24
3417	205,818311	274,165982	24
3418	204,547925	273,322023	24
3419	205,627662	274,145694	24
3420	205,559996	278,871792	24
3421	206,294649	280,127319	24
3422	205,464832	280,833272	24
3423	205,705577	280,284262	24

Annex 5: Time Standards

Time format	Definition
SCLK	Spacecraft Clock. This is an on-board 64-bit counter, in units of nano seconds and increments once every 100 milliseconds. Time zero corresponds to midnight on 1-Jan-1980
Local Solar Time (LST).	This is the local solar time defined by the local solar days (sols) from the landing date using a 24 Martian hour clock within the current local solar day (HR:MN:SC). Since the Mars Day is 24h 37m 22s long, each unit of LST is slightly longer than the corresponding Earth unit. LST is computed using positions of the Sun and the landing site from SPICE kernels. If a landing date is unknown to the program (e.g., for calibration data acquired on Earth) then no sol number will be provided on output LST examples: SOL 12 12:00:01 SOL 132 01:22:32.498 SOL 29
Local True Solar Time (LTST)	This is related to LST, which is also known as the mean solar time. It is the time of day based on the position of the Sun, rather than the measure of time based on midnight to midnight “day”. LTST is used in all MIPL/OPGS generated products.
Local Mean Solar Time (LMST)	Specifies the local mean solar time, or LMST. It is one of two types of solar time used to express the time of day at a point on the surface of a planetary body.

	The desire to work with solar days, hours, minutes and seconds of uniform length led to the concept of the fictitious mean Sun or FMS. The FMS is defined as a point that moves over the celestial equator of a planetary body at a constant rate representing the average mean motion of the Sun over a planetary year. Local mean solar time, LMST, is defined, by analogy with local true solar time (LTST), as the difference between the areocentric right ascensions of a point on the surface and that of the FMS. The difference between LTST and LMST varies over time. The length of a mean solar day is constant and can be computed from the mean motion of the FMS and the rotation rate of a planet. The mean solar day is also called a 'sol'. Mean solar hours, minutes, and seconds are defined in the same way as the true solar units.
SOL	Solar Day Number, also known as PLANET DAY NUMBER in PDS label. This is the number of complete solar days on Mars since landing. The landing day is SOL one.