





Università Politecnica delle Marche  
Scuola di Laurea Magistrale in Scienze dell'Ingegneria  
Curriculum in "Environmental Engineering"

---

# **Development of a user friendly AERMOD interface and evaluation of model performance on a case study in complex environments with multiple sources**

Dissertation of:  
**Dominik Subotić**

Advisor:

**Prof. Eng. Giorgio Passerini**

Curriculum supervisor:

**PhD Simone Virgili**

A.A. 2023./2024.







Università Politecnica delle Marche  
Scuola di Laurea Magistrale in Scienze dell'Ingegneria  
Curriculum in "Environmental Engineering"

---

# **Development of a user friendly AERMOD interface and evaluation of model performance on a case study in complex environments with multiple sources**

Dissertation of:  
**Dominik Subotić**

Advisor:

**Prof. Eng. Giorgio Passerini**

Curriculum supervisor:

**PhD Simone Virgili**

A.A. 2023./2024.

---

Università Politecnica delle Marche  
*Dipartimento Di Ingegneria Industriale E Scienze Matematiche – DIISM*  
Via Brecce Bianche — 60131 - Ancona, Italy



## *ACKNOWLEDGEMENTS*

I would like to express gratitude to my advisor, prof. Giorgio Passerini, my supervisor, PhD Simone Virgili whose doctoral thesis inspired my master thesis, and UNIVPM for their guidance, material, knowledge, and opportunity to study and do my master thesis. The university staff has been nothing but outgoing, kind, dedicated and professional.

Secondly, I would like to thank EPA for developing AERMOD on which my thesis is based. Mathworks for developing MATLAB, JRSoftware for developing Inno Setup, JetBrains for developing PyCharm, which was a crucial platform for developing my application, and Microsoft for developing MS Excel.

I would finally, like to thank my family, especially my mother Vekenega, for their sacrifice and believing in my potential and vision, my friends Anđelo, Roko and Josip for always reaching out to me, being patient and inspiring me to live life to the fullest. Mia Štulić for bringing out the best in me and pushing me to pursue my dreams and goals.

Thank you all, I wouldn't be here without you.







## *ABSTRACT*

A SO<sub>x</sub> emission source (API Raffineria, Falconara Marittima, Italy) is analyzed using AERMOD air dispersion modelling system, for the purpose of demonstrating newly developed software (CAIRO for AERMOD), made to compile and visualize input, and run analyses. CAIRO for AERMOD (Compile AERMAP, AERMOD and AERPLOT Input and Run Output) is a python-based GUI aimed at streamlining the process of making complex input files, with unique syntax and running them, while obviating the need to manually write input and run the program through Windows Shell. The Windows application features automated features such as the input and conversion of coordinates to UTM through copy operations, while the input is visualized in Google Earth. Input is done through user interface and automatically compiled into correct syntax and project/file structure. It supports point or polygon sources up to 3 averaging periods, maxtable, maxifile, rankfile and plotfile, while the elevation and meteorological data must be third party.

A review of air pollution, legislative, planetary boundary layer processes and AERMOD model formulation, introduced the analysis on SO<sub>x</sub> emissions of 15 point sources in a refinery located near domestic areas. The AERMOD output and real data of 3 monitoring stations, dedicated to monitoring the plant, were further processed using multiple methods to deduce the difference of modelled values compared to actual data. The model performed with an average difference of  $-1.84 \mu\text{g SO}_x/\text{m}^3$  or 5.33%, compared to actual data. The model performance was acceptable by Normalized Mean Square Error (NMSE), Mean Fractional Bias (FB) and Mean Bias (MB) tests to comply with European legislation. Longer averaging periods (month and year) had smaller maximal deviations but on average it exhibited the same deviations as the shorter period (24h), which had up to  $\pm 78\%$  discrepancies from real data. Compared to the regulatory limit it's a relative deviation of  $\pm 3.20\%$  of the regulatory limit on average.





## ***CONTENT***

1. INTRODUCTION.....	1
2. MATERIALS AND METHODS .....	4
2.1. AIR POLLUTION.....	5
2.1.1 ENVIRONMENTAL AND HEALTH EFFECTS.....	8
2.2. ENVIRONMENTAL ASSESSMENT.....	13
2.2.1 LEGISLATIVE .....	13
2.2.2 MONITORING .....	18
2.2.3 MODELLING.....	21
2.3 EPA .....	29
2.4. PYTHON .....	32
2.4. AERMOD .....	34
2.6. GRAPHICAL USER INTERFACE .....	53
2.6.1 AERMAP INPUT FILE COMPILER .....	57
2.6.2 AERMOD INPUT FILE COMPILER.....	59
2.6.3 AERPLOT INPUT FILE COMPILER.....	65
2.7. PROGRAMMING LOGICS .....	70
2.7.1 AERMAP INPUT FILE .....	71
2.7.2 AERMOD INPUT FILE.....	79
2.7.3 AERPLOT INPUT FILE.....	102
2.7.4 “CAIRO for AERMOD” .....	112
2.7.5 COMPILING EXECUTIVE FILE .....	116
2.7.6 COMPILING INSTALLER .....	116
2.8 CASE STUDY ON MULTIPLE INDUSTRIAL SOURCES .....	119
2.8.1 AERMAP IMPLEMENTATION .....	122
2.8.2 AERMOD IMPLEMENTATION.....	127
2.8.3 AERPLOT IMPLEMENTATION.....	132

2.8.4	MATLAB POSTPROCESSING.....	136
2.8.5	MODEL VALIDATION .....	140
3.	RESULTS.....	145
3.1	AERMOD ANALYSIS WITH RECEPTOR GRID .....	146
3.2	AERPLOT POSTPROCESSING WITH RECEPTOR GRID .....	147
3.3	DATA POST PROCESSING.....	153
3.4	MODEL VALIDATION.....	158
4.	DISSCUSSION .....	164
4.1	LOCAL FACTORS.....	164
4.2	“CAIRO” PERFORMANCE .....	169
4.3	CASE STUDY .....	173
5.	CONCLUSION .....	179
6.	REFERENCES.....	182







## *LIST OF FIGURES*

Figure 1. “Modern” composition of air including criteria and other pollutants, also noted must be the average water vapor content of around 4%. Units are percent, parts per million/billion/trillion. Values correspond to natural “rural background” concentrations, ambient concentrations in metropolitan areas are substantially greater. (Fowler, et al., 2020.) .....	6
Figure 2. Long term observations of SO <sub>x</sub> , VOC, NO <sub>x</sub> , PM10, from 1940. until 2010. (PM10 monitoring only started in 1990. and led in 1970.). Interestingly the drop in all parameters co-aligns with the foundation of EPA (Environmental Protection Agency) in December of 1970 (EPA, 2010.).....	7
Figure 3. Long term observations of CO and Pb, from 1940. until 2010. (EPA, 2010.).....	8
Figure 4. Emission reduction of the main air pollutants by European union states from 2005 to 2021, in compliance with the previous “Directive 2001/81/EC” (EEA, 2022.) .....	16
Figure 5. Percentage emission reductions time plots of main air pollutants from 2005. to 2021. (EEA, 2022.).....	17
Figure 6. Variation of atmospheric stability due to vertical temperature distribution	
A) Absolute instability (strong negative temperature gradient)	
• Dry-adiabatic lapse rate unsaturated parcels cool at a rate of 10°C km <sup>-1</sup>	
B) Neutral condition or conditional stability - when the lapse rate between the dry and moist adiabatic lapse rate. Conditional instability exists when the atmosphere's stability depends on the saturation of the rising air	
• Moist Adiabatic Lapse Rate – For a saturated parcel of air, i.e., when its T=T <sub>d</sub> , then it cools at the moist adiabatic lapse rate = 6°C km <sup>-1</sup>	
• Absolute stability occurs when the ELR is less than the moist adiabatic lapse rate	

- C) Sub adiabatic: Ambient lapse rate  $<$  adiabatic. It indicates stable atmosphere, vertical motion, and mixing are suppressed. Dispersion is suppressed, and contamination is trapped.
- D) Isothermal vertical temperature distribution, indicates stratification
- E) Inversion means a hot top layer has trapped pollutants near ground.....22

Figure 7. Different types of plume behavior for various atmospheric stability conditions. The dotted lines show dry adiabatic lapse rate, the solid lines represent the actual adiabatic lapse rates for different atmospheric stability conditions. Lapse is the normal temperature fall with height and inversion is the rise of temperature with height (Geiger, et al., 1995.) .....24

Figure 8. Plume emitted by API refinery in Falconara Marittima, Italy, aloft due to PBL stratification, lightly rising due to residual ground heat flux (Cronache Ancona, 2024.).....25

Figure 9. Scheme of troposphere division into free atmosphere and boundary layer (Stull, 2012.).....37

Figure 10. Time evolution of PBL (Stull, 1988.) .....38

Figure 11. Lofting of a smoke plume occurring when the top of the plume grows upward into a neutral layer while the bottom is stopped by a stable layer .....39

Figure 12. A growing mixed layer mixes elevated smoke plumes down to the ground e.g. fumigation .....39

Figure 13. Vertical wind speed profile for CBL and SBL, in the region below  $7z_0$  (EPA, 2023.) .....44

Figure 14. Vertical wind speed profile, for CBL and SBL, in the region above  $7z_0$  (EPA, 2023.) .....44

Figure 15. Mechanical portion of the vertical turbulence in the CBL, corresponding to total vertical turbulence of SBL (EPA, 2023.) .....45

Figure 16. Convective portion of the vertical turbulence in the CBL (EPA, 2023.).....	45
Figure 17. A contaminant plume emitted from a continuous point source, with wind direction aligned with the x-axis. Profiles of concentration are given at two downwind locations, and the Gaussian shape of the plume cross-sections are shown relative to the plume centerline (Stockie, 2011.).	46
Figure 18. Terrain treatment in AERMOD, visualizing the concept of dividing streamlines and the construction of the weighting factor used in calculating total concentration (EPA, 2019.) .....	47
Figure 19. Instantaneous and corresponding ensemble-averaged plume in the CBL (EPA, 2019.) .....	48
Figure 20. AERMOD’s three plume treatments/interpretations of the CBL (EPA, 2019.).....	50
Figure 21. AERMOD’s PDF approach for plume dispersion in CBL e.g. superimposition of two Gaussian distributions, the updraft and downdraft distribution (EPA, 2019.) .....	52
Figure 22. PDF of the vertical velocity. The bi-Gaussian curve has a skewness of $S=1$ . About 60% of the $p_w$ integral is on the negative side, the rest is positive, consistent with results of numerical simulations and field observations. (EPA, 2019.).....	52
Figure 23. General flowchart of data processing .....	53
Figure 24. Detailed flowchart of data processing for the “CAIRO for AERMOD” app .....	54
Figure 25. The main window of the interface currently running AERMAP	56
Figure 26. AERMAP compiler overlaying “Google Maps” .....	58
Figure 27. Example of aermap.inp file contents created with the AERMAP input file compiler .....	58

Figure 28. Grid receptor network (with already processed sources) created with the input file from figure 19. and figure 25., visualized in Google Earth .....	59
Figure 29. AERMOD input file compiler GUI .....	60
Figure 30. AERMOD input file compiler with input information .....	61
Figure 31. Added point sources and polygon sources automatically visualized in real time using “Google Earth”.....	62
Figure 32. Manually adding polygon sources and vertices .....	63
Figure 33. Example of an “aermod.inp” file contents created with the AERMOD input file compiler .....	64
Figure 34. AERPLOT input file compiler.....	66
Figure 35. Compiled “aerplot.inp” file.....	67
Figure 36. Contents of the project folder containing the “aerplot” (1,2,3) subfolders after running the “AERPLOT input file compiler” (and previous AERMOD stages).....	67
Figure 37. Contents of one of the AERPLOT subfolders, after running AERPLOT, there are three iterations, each for one of the averaging periods .....	68
Figure 38. Concentration distribution for the 24h period visualized in Google Earth using 5 point sources over Newark, USA.....	68
Figure 39. Concentration distribution for the 24h period visualized QGIS using contour and gradient lines with 5 point sources over Newark, USA..	69
Figure 40. Concentration distribution for the 24h period visualized QGIS using the grid receptor network, contour and gradient lines with 5 point sources over Newark, USA .....	69

Figure 41. Example AERMAP input file .....	71
Figure 42. Example AERMOD input file with point sources and polygon sources added both by manually inputting UTM coordinates and by using Google Maps to interactively add vertices and visualize them in Google Earth in real time .....	81
Figure 43. Polygon and point sources during creation of an AERMOD input file. The “.kml” file, is automatically and continuously updated in Google Earth as the sources and vertices are being input .....	92
Fig 44. Example of 1st of three created AERPLOT input files created using the AERPLOT input file compiler .....	103
Figure 45. Hourly ( $350 \mu\text{g}/\text{m}^3$ ), daily ( $125 \mu\text{g}/\text{m}^3$ ), yearly and winter (1. October to 31. March, as a means of vegetation protection) ( $20 \mu\text{g}/\text{m}^3$ ) $\text{SO}_2$ limits given by the Italian legislation (ARPAM, 2010.) .....	120
Figure 46. Downloading DEM data from Copernicus browser and Copernicus GLO 30 data, with 30 m/px spatial resolution (15 m/px effective resolution, due to resolution settings) in 16 bit “TIFF” format and WGS84 UTM 33N projection. The appropriate product is selected in the browser, an area selected and downloaded. ....	122
Figure 47. Domain (Red – 20*20km, corresponding to receptor grid area) visualized over “GeoTiff” elevation data file with single band pseudo color scheme applied and Google Satellite imagery of Marche, Italy in QGIS. Coordinates are UTM (m). ....	123
Figure 48. Predetermined anchor point (SW corner) coordinates being copied from Google Maps, while in the GUI they are automatically input and converted to UTM northing, easting and zone .....	124
Figure 49. GUI with opened fully filled out “AERMAP input file compiler” window. UTM coordinates and zone were automatically filled out and converted from Google Maps data. ....	124
Figure 50. Resulting AERMAP input file for API refinery .....	125

Figure 51. AERMAP running via “CAIRO for AERMOD” after selecting the input project folder for API refinery, Falconara Marittima, Italy ..... 125

Figure 52. Snippet of “receptor.rou”, the resulting receptor grid file ..... 126

Figure 53. Domain and sources visualized over Google Terrain in QGIS. 127

Figure 54. Point sources visualized in QGIS, overlaying Google Hybrid, highlighting the proximity to urban areas (Falconara Marittima, Italy)..... 128

Figure 55. Google Hybrid view in QGIS of point sources, with labels, at API refinery in Falconara Marittima, Italy ..... 128

Figure 56. 3D view of sources created in Google Earth including labels and real heights ..... 129

Figure 57. Mean daily minimum, maximum and range temperatures, including precipitation based on data from 1993.-2023. for Falconara Marittima, Italy (Meteoblue, 2024.) ..... 130

Figure 58. Wind rose with highlighted wind speed fractionation of the predominant wind direction for Falconara Marittima, Italy (Meteoblue, 2024.)..... 130

Figure 59. Opened “.sfc” file containing MMIF surface meteorological data and the station numbers ..... 131

Figure 60. AERMOD input file compiled for the means of analyzing “API” refinery point source emissions. .... 131

Figure 61. “AERPLOT input compiler” interface filled out with the correct maximum and minimum bins, logarithmic binning, UTM zone and other generic data..... 133

Figure 62. Files in the project folder resulting in using “CAIRO for AERMOD” ..... 133

Figure 63. 24 h averaging period visualized in Google Earth..... 134

Figure 64. Monthly averaging period visualized in Google Earth .....	134
Figure 65. Annual averaging period visualized in Google Earth .....	135
Figure 69. “ARPA” map of receptor sites, with locations of the 3 used monitoring stations in Falconara Marittima, Italy, and location of “API” refinery (e.g. sources).....	140
Figure 70. Creating receptor points in QGIS from manually created “.csv” file.....	141
Figure 71. Monitoring station locations (green) and sources (white) visualized in QGIS over analyzed gradient lines (purple) .....	142
Figure 72. AERMAP input file using the “DISCCART” keyword to model real discrete industrial monitoring stations in Falconara Marittima, Italy. The keyword is coupled with the x and y UTM coordinates, base elevation and hill elevation. ....	143
Figure 73. “RANKFILE” output of AERMOD for the 24h averaging period, listing overall maximum values while omitting duplicate date/hours values .....	146
Figure 74. “RANKFILE” output of AERMOD for the monthly averaging period, listing overall maximum values while omitting duplicate date/hours values.....	146
Figure 75. Modeled 15 point source SO <sub>x</sub> emissions from ”API” refinery in Falconara Marittima, Italy, for the 24 h averaging period, displayed in QGIS. The domain is a 20*20km receptor grid (200*200 node grid with 100 m interstep), receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme. ....	147
Figure 76. Modeled 15 point source SO <sub>x</sub> emissions from ”API” refinery in Falconara Marittima, Italy, for the monthly averaging period, displayed in QGIS. The domain is a 20*20km receptor grid (200*200 node grid with 100 m interstep), receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme. ....	148



Figure 77. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the total (the year 2020.) averaging period, displayed in QGIS. The domain is a 20\*20km receptor grid (200\*200 node grid with 100 m interstep), receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme. .... 149

Figure 78. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the 24 hour averaging period, displayed in QGIS. The map is zoomed in over the refinery and the city of Falconara Marittima, highlighting the proximity of the industrial plant and the concentration over the urban area. Receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme..... 150

Figure 79. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the monthly hour averaging period, displayed in QGIS. The map is zoomed in over the refinery and the city of Falconara Marittima, highlighting the proximity of the industrial plant and the concentration over the urban area. Receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme..... 151

Figure 80. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the total (the year 2020.) averaging period, displayed in QGIS. The map is zoomed in over the refinery and the city of Falconara Marittima, highlighting the proximity of the industrial plant and the concentration over the urban area. Receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme..... 152

Figure 81. SO<sub>x</sub> concentrations for 24 hour (blue), monthly (green) and total (red - 1 year) averaging periods, visualized using boxplots and a logarithmic scale to visualize the concentration distribution..... 156

Figure 82. SO<sub>x</sub> concentration areal distribution for 24 hour (blue), monthly (green) and total (red - 1 year) averaging periods. The concentration distribution is plotted on x with a logarithmic scale and the percentage of the domain is plotted on the y axis. .... 157





## LIST OF TABLES

Table 1. Partial list of member states and their set reduction target values set by the NEC Directive, under the AAQDs (Directive (EU) 2016/2284, 2016.) The values are compared to 2005 data. ....	17
Table 2. Reduction target values set by the NEC directive, under the AAQDs (ENEA, 2021.).....	18
Table 3. The data for the point sources in the “API” refinery in Faloconara Marittima, Italy. Includes the source name (ID), base elevation, height, diameter, exit velocity and temperature, emission rate and UTM coordinates given in WSG84 UTM 33N projection system (ESPG:32633) .....	121
Table 4. The locations of sources in displayed in UTM zone, northing and easting, and converted into latitude and longitude. The projection is WSG84 UTM 33N (ESPG:32633).....	121
Table 5. Minimum and maximum values of each period gathered from the control AEROPLOT run, minimal bin was rounded to 1 due to non-relevance regarding legislation for lower values and achieving a finer discretization between concentration bins.....	132
Table 6. Table containing bin boundary values and contour line values (corresponding to lower bin boundary), with according color palette .....	135
Table 7. Locations of monitoring points in latitude and longitude and UTM. “Altitude” refers to elevation at base, receptor height is set to 1.5m .....	141
Table 8. 3D view of sources created in Google Earth including labels and real heights .....	153
Table 9. Concentration bins along with the number of corresponding nodes in the analysis, its areal coverage in percentage and km <sup>2</sup> , and color scheme for the monthly averaging period .....	154

Table 10. Concentration bins along with the number of corresponding nodes in the analysis, its areal coverage in percentage and km <sup>2</sup> , and color scheme for the total period (1 year) averaging period.....	155
Table 11. Maximum and minimum, mean, median, 25th and 75th percentile values, mean values of the maximum and minimum bin, and the mass of the pollutant in a 1 m thickness layer, over the domain, at receptor height. Data was calculated using MATLAB, and using data from the AERMOD simulation of air pollution in Falconara Marittima, Italy .....	156
Table 12. Maximum, minimum, mean, median, 25th and 75th percentile values for the Falconara Acquedotto monitoring station for the year 2020. ....	158
Table 13. Maximum and minimum, mean, median, 25th and 75th percentile values for the Falconara Scuola monitoring station for the year 2020.....	158
Table 14. Maximum and minimum, mean, median, 25th and 75th percentile values for the Falconara Alta monitoring station for the year 2020.....	159
Table 15. Difference in location of modeled to real receptors .....	159
Table 16. Tabular data of AERMOD modelled difference, comparing the AERMOD analysis with grid receptors (highest 98 <sup>th</sup> percentile excluded) compared to mean data per averaging period from 3 monitoring stations, including percentages .....	160
Table 17. Main tabular data comparing the AERMOD analysis with receptor grid (highest 98 <sup>th</sup> percentile excluded) compared to maximum data per averaging period, from 3 monitoring stations, including percentages .....	160
Table 18. Tabular data comparing the AERMOD analysis with discrete receptors (highest 98 <sup>th</sup> percentile excluded) compared to real data from 3 monitoring stations, including percentages .....	161
Table 19. Overview of differences between receptor grid modeled and real data. “Limit Fraction” is the modelled differences percentage of the SO <sub>x</sub> limit .....	162

Table 20. Overview of differences between discrete receptor modeled and real data. “Limit Fraction” is the modelled differences percentage of the SO <sub>x</sub> limit .....	162
Table 21. Detailed differences between discrete receptor modeled and real data in relation to regulatory limits. “Effective %” is the modelled differences percentage of the SO <sub>x</sub> limit .....	163
Table 22. Results of testing by Normalized Mean Square Error (NMSE), Fractional Bias (FB) and Mean Bias (MB) tests to measure the model’s results compared to observed data for compliance to European Union regulation.....	163







## ***1. INTRODUCTION***

The dynamic nature of environmental systems necessitates sophisticated models to predict and understand the interactions between various factors such as urbanization, industrialization, and climate change. The primary adverse effects of these activities include alterations in radiative balance and local heating patterns, which in turn impact micro-scale circulation and surface energy balance. This affects pollutant dispersion, distribution, chemical and physical activity. Changes in thermal behavior of regional weather due to urbanization and industrialization have well known effects on climate and air quality (Baik et al., 2000; Roth, 2000). Growing awareness of air quality deterioration, frequency of pollution events and health effects, have led to the integration of air pollution control strategies within urban planning frameworks, e.g. environmental monitoring and modeling. Making practical and effective modeling tools is at the basis of scientific research, engineering, industrial, legislative and other applications.

Environmental models give access to complex environmental processes that are difficult to study directly due to their scale, complexity, or the length of time over which they occur. And an understanding of the interactions between different components of the environment, such as the atmosphere, hydrosphere, biosphere, and lithosphere. Acting as a “virtual laboratory”, they predict possible outcomes, critical for planning and decision-making, especially in the context of climate change, pollution control, resource management, risk assessment, environmental impact assessments and compliance.

Air quality models usually incorporate parameterizations for the source, planetary boundary layer (PBL), turbulence and terrain interactions. The vertical mixing of air pollutants strongly depends on the depth and stratification of the PBL, which is governed by factors such as the PBL energy balance (heat flux), vertical motion, horizontal advection, entrainment at the boundary layer top, and time. Outputs offer quantitative predictions, scenario analysis, risk assessment and visualizations to help interpret and communicate complex data and results. More specifically, concentration maps, time series data, deposition rates, exposure and risk assessments with

the goal of regulatory compliance. The outputs of environmental air dispersion models typically include data on pollutant concentrations and other environmental parameters over time and space.

The best strategies for using environmental models are based on understanding of the scientific principles, validated data to capture the essential dynamics of the systems they represent. Choosing according to transparency and accessibility of models allows for peer review and stakeholder engagement. Integrating data from multiple sources and using an interdisciplinary approach can improve model comprehensiveness, accuracy and reliability. Regularly validating model outputs with observed data and guidelines ensures accuracy. They finally depend on the goal and specific case study.

The use of environmental models has required, and to a certain degree still requires, significant expertise in both the environmental sciences and the specific modeling software, but new advances in software development have made these tools more accessible to a broader range of users, including policymakers, educators, and community stakeholders. Models, such as AERMOD, are widely used for urban air quality forecasts due to their ability to simulate pollutant dispersion effectively under varied conditions and complex terrains. Despite scientific advancements, predicting the transport, diffusion, and transformation of airborne pollutants remains complex due to data limitations and the inherent complexities of atmospheric processes (Klausmann et al., 2003).

This study glances over air pollution, its main components and processes, environmental monitoring and modeling, focusing on the AERMOD dispersion model's formulation and the creation of a Graphical User Interface (GUI) in Python to streamline the use of AERMOD and its associate processors and input files. This GUI compiles AERMOD preprocessor, processor, and postprocessor input file and runs basic AERMOD stages, simplifying the process compared to manual command shell operations. Traditionally, running AERMOD involves manually preparing input files, executing commands in the shell, and managing output files, which can be inept and error prone or requiring expensive software options with an integrated interface.

The proposed GUI is a windows application that simplifies the process by integrating the preprocessor, processor, and postprocessor stages of AERMOD (AERMAP, AERMOD, AERPLOT), compiling input files (AERMAP, AERMOD, AERPLOT), allowing users to input data, run simulations, and view results within a single interface. Integrating a simple input interface, Google Maps and Google Earth, it enhances accessibility and usability, particularly for users who may not be familiar with command line operations.

To validate the effectiveness of the GUI, a generic analysis was conducted on real pollutant sources. The analysis involved simulating the dispersion of pollutants using the AERMOD model and comparing the results with observed data. The results were analyzed using QGIS, MATLAB and MS Excel to provide a comprehensive understanding of the pollutant distribution and its potential impacts. This practical application demonstrates the GUI's capability to facilitate detailed and accurate air quality modeling, making it a valuable tool for researchers and policymakers alike.

## ***2. MATERIALS AND METHODS***

This chapter will delve into the main problems, processes, impact, legislation, and regulations regarding air pollution, particularly the monitoring and modeling of air quality for the purpose of compliance with authoritative bodies. Focusing on the 5 key pollutants given by the NEC Directive in 2016, and other common pollutants with adverse environmental and health effects.

An overview of the air dispersion model AERMOD follows, expands on the model's formulation together with the meteorological phenomena, processes, parameters, and specifics associated with it. Most importantly, it delves into the formulation of a computer program in Python programming language and its instructions, aiming to simplify the creation and processing of AERMOD input files.

The program uses only 2 vital source types (point and polygon area sources) and allows for simple input and visual representation during operation, for instance choosing locations directly from Google Maps and visualizing multiple sources in real time in Google Earth. It is based upon using the AERMOD preprocessor AERMAP, using provided surface air and upper air AERMET output files, the processor AERMOD and post processor AERPLOT. Input file compilers are available for AERMAP, AERMOD and AERPLOT.

## *AIR POLLUTION*

The atmosphere is a complex dynamic natural gaseous system that is essential to support life on planet Earth. Its composition is crucial for sustaining life and maintaining environmental stability. Nitrogen is essential for plant growth and is a building block of proteins, while oxygen supports respiration. Trace gases, though present in minimal amounts, play significant roles; for instance, carbon dioxide and methane are critical for the greenhouse effect, which regulates the planet's temperature. The atmosphere also interacts with the hydrosphere and biosphere, driving weather patterns and climate systems, and ensuring a balanced environment that supports diverse ecosystems.

Air pollution, as defined by the World Health Organization, is “contamination of the indoor or outdoor environment by any chemical, physical or biological agent that modifies the natural characteristics of the atmosphere which has a negative effect on health and the environment” (WHO, 2024.). Emissions constitute the output of polluting substances to the atmosphere from any source, while imissions are the “reception” of the emitted pollutants and constitute what is known as “air quality” (EEA, 2024.).

Criteria pollutants are specific air contaminants regulated due to their health and environmental risks. These include ozone (O<sub>3</sub>), particulate matter (PM<sub>10</sub> and PM<sub>2.5</sub>), carbon monoxide (CO), sulfur dioxide (SO<sub>2</sub>), nitrogen oxides (NO<sub>x</sub>), and lead (Pb). Normally, Earth's atmosphere is composed of nitrogen (78%), oxygen (21%), and trace amounts of argon, carbon dioxide, and other gases.

Primary air pollutants are directly generated, while secondary air pollutants are indirectly generated through some other processes. Secondary air pollutants are created by various processes like burning of fossil fuels (electricity generation, transport, industry and households), industrial processes and solvent use (chemical and mining industries), agriculture, waste treatment, natural sources (volcanic eruptions, windblown dust, sea-salt spray and emissions of volatile organic compounds from plants), for which a well-known example is photochemical smog. CO<sub>2</sub> isn't regarded as a pollutant by legislatures, although by its role as a greenhouse gas it is regularly monitored for (Sharma et al., 2013.).

<b>Gas</b>	<b>Concentration (units as listed<sup>a</sup>)</b>
Nitrogen	78.09 %
Oxygen	20.94 %
Argon	0.93 %
Carbon Dioxide	390 ppm
Neon	18 ppm
Helium	5.2 ppm
Methane	1.7 ppm
Krypton	1.0 ppm
Hydrogen	500 ppb
Nitrous Oxide	300 ppb
Xenon	80 ppb
Criteria Pollutants <sup>b</sup>	
CO	100 ppb
O <sub>3</sub>	20 ppb
NO <sub>2</sub>	1 ppb
SO <sub>2</sub>	200 ppt
Others <sup>b</sup>	
Ammonia	10 ppb
Freon (CFC-11)	230 ppt
Hydrogen Sulfide	200 ppt
***	

<sup>a</sup> Units are percent, parts per million, parts per billion, and parts per trillion—all on a volume basis.  
<sup>b</sup> Values given here represent (approximately) natural "rural background" concentrations; ambient concentrations in urban areas are often considerably higher.  
\*\*\* A variety of other relatively stable organic and inorganic gases may exist at various concentrations, especially in urban areas.  
Values in this table were obtained from various sources.

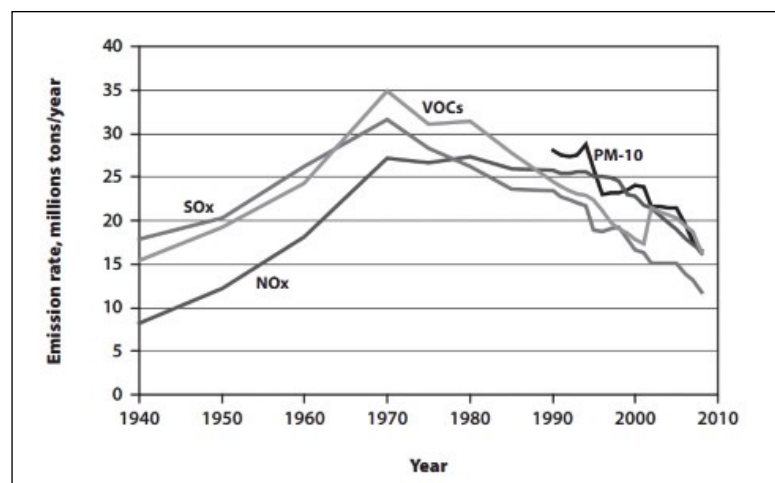
*Figure 1. "Modern" composition of air including criteria and other pollutants, also noted must be the average water vapor content of around 4%. Units are percent, parts per million/billion/trillion. Values correspond to natural "rural background" concentrations, ambient concentrations in metropolitan areas are substantially greater. (Fowler, et al., 2020.)*

Key primary air pollutants:

- Particulate matter (PM)
- Black carbon (BC)
- Sulphur oxides (SO<sub>2</sub>)
- Nitrogen oxides (NO<sub>x</sub>-NO, NO<sub>2</sub>)
- Ammonia (NH<sub>3</sub>)
- Carbon monoxide (CO)
- Methane (CH<sub>4</sub>)
- Non-methane volatile organic compounds (NMVOCs- including benzene, certain metals and polycyclic aromatic hydrocarbons including benzo[a]pyrene - BaP)

Key secondary air pollutants:

- Particulate matter (PM - key precursor gases for secondary PM are SO<sub>2</sub>, NO<sub>x</sub>, NH<sub>3</sub> and VOCs)
- Ozone (O<sub>3</sub>)
- Nitrogen dioxide (NO<sub>2</sub>)
- Oxidized volatile organic compounds (VOCs) (Sharma et al., 2013.)



*Figure 2. Long term observations of SO<sub>x</sub>, VOC, NO<sub>x</sub>, PM<sub>10</sub>, from 1940. until 2010. (PM<sub>10</sub> monitoring only started in 1990. and led in 1970.). Interestingly the drop in all parameters co-aligns with the foundation of EPA (Environmental Protection Agency) in December of 1970 (EPA, 2010.).*

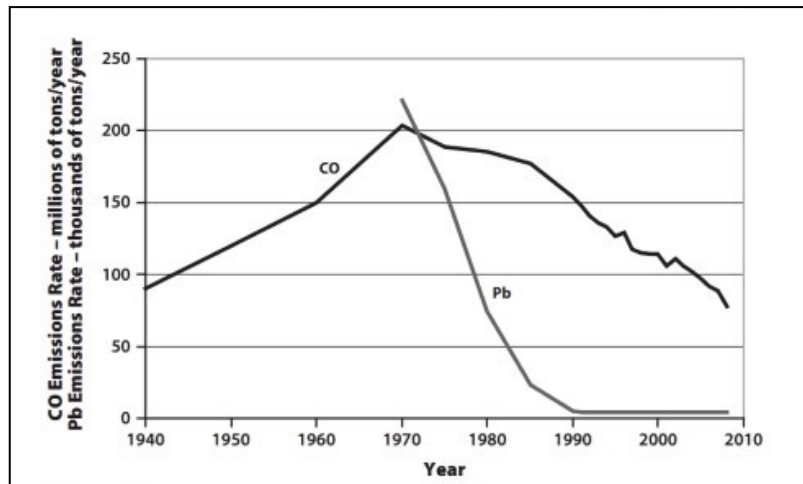


Figure 3. Long term observations of CO and Pb, from 1940. until 2010. (EPA, 2010.)

### 2.1.1 ENVIRONMENTAL AND HEALTH EFFECTS

Environmental impacts of air pollution include contamination of air and bodies of water and soil through wet and dry deposition and reactions with atmospheric components. Air pollution poses numerous threats to the environment and public health, creating a strong need for environmental assessment and legislative actions.

Wet deposition, known as acid rain, occurs when pollutants absorbed by water droplets in the atmosphere precipitate to the ground. This process lowers the pH of soil and water bodies, harming aquatic life, vegetation, and infrastructure. Acid rain can leach essential nutrients from the soil, disrupt ecosystems, and damage buildings and monuments made from limestone and marble (Singh and Agrawal, 2007).

Dry deposition is the settling of airborne pollutants onto surfaces, including soil, water, and vegetation. By means of both dry and wet deposition pollutants can enter the food chain through crops and water

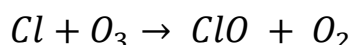


sources, leading to bioaccumulation and biomagnification (Wesely and Hicks, 2000.; Deribe et al., 2013.).

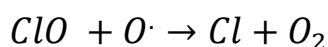
Reactions with atmospheric constituents and other pollutants can result in forming of acidic compounds, ground level ozone and fine particulate matter, of which the latter two are key components of smog.

Ozone depleting substances (ODS) mainly act by reaction of chlorine and bromine atoms with ozone ( $O_3$ ) (Reaction 1. - ozone destruction). One chlorine atom can destroy multiple thousand molecules of ozone by catalytic cycling (Reaction 2. - Ozone regeneration), where Cl acts as a catalyst and reacts with  $O_3$ , creating chlorine monoxide ( $ClO + O_2$ ) after which it can again react with O- ions, becoming a free radical, free to react further without being consumed. As ODSs are exposed to UV light in the stratosphere, they readily release chlorine and bromine atoms which react further. These substances include chlorofluorocarbons (CFC), hydrochlorofluorocarbons (HCFC), carbon tetrachloride and methyl bromide, among others (Bouare, 2009.).

Reaction 1.



Reaction 2.



Ground level ozone ( $O_3$ ) is formed photochemical and chemical processes mainly by  $NO_x$ , CO, and VOCs, who stem from exhaust and industrial emissions. It drives many chemical processes and is a pollutant itself. Ozone is a key constituent of the troposphere and an important constituent of certain regions of the stratosphere, known as the Ozone layer. It's also the third most important greenhouse gas in terms of radiative forcing, therefore accelerating climate change. At extremely high concentrations from human activities (largely the combustion of fossil fuel), it is a pollutant, and a constituent of smog (Sharma et al., 2013.; Myhre et al., 2014.).

Sulphur oxides (SO<sub>x</sub>), mainly SO<sub>2</sub> which is produced by volcanoes and in various industrial processes as coal and petroleum are often sulfur containing compounds. Further oxidation of SO<sub>2</sub>, usually in the presence of catalyst such as NO<sub>2</sub>, forms H<sub>2</sub>SO<sub>4</sub>, thus creating a component of acid rain. The highest concentrations of SO<sub>2</sub> are recorded in the vicinity of large industrial facilities. SO<sub>2</sub> emissions are also a major precursor to ambient PM<sub>2.5</sub> concentrations. SO<sub>2</sub> is also created by oxidation of H<sub>2</sub>S, which originates from combustion processes and anaerobic decay. (Sharma et al., 2013.; Lee et al., 2018.).

Nitrogen oxides (NO<sub>x</sub> - most prominently NO<sub>2</sub>) are emitted from high temperature combustion and are also produced naturally during thunderstorms by electric discharge. They can be seen as the reddish-brown haze dome above or plume downwind of cities. Most noted effects include formation of ground level ozone (through reaction with VOCs and UV light), acid deposition (HNO<sub>3</sub>), eutrophication, greenhouse gases (N<sub>2</sub>O). NO<sub>2</sub>, like SO<sub>2</sub>, is a precursor to PM<sub>2.5</sub>. (Sharma et al., 2013.; Boningari et al., 2016.; Lee et al., 2018.).

Ammonia (NH<sub>3</sub>) is mainly emitted from agricultural processes, normally in the form of a gas with a distinct odor. It serves as a precursor to nutrients and fertilizers to terrestrial organisms and as a building block for the synthesis of many pharmaceuticals. It is emitted from various industrial processes, including wastewater treatment plants. Although in wide use, ammonia is both caustic and hazardous. (Sharma et al., 2013.; Zhang et al., 2017.)

Carbon monoxide (CO) is a colorless, odorless gas. It is a byproduct of incomplete combustion of fuels. Vehicular exhaust is a major source of carbon monoxide. It is especially dangerous for organisms and human health as it binds 200-300 times more strongly than oxygen to hemoglobin (Patel, et al. 2023.).

Volatile organic compounds (VOC) are a pollutant category often divided into methane (CH<sub>4</sub>) and non-methane (NMVOCs) compounds. Methane is an extremely efficient greenhouse gas. Other hydrocarbon VOCs are also significant greenhouse gases via their role in creating ozone and in

prolonging the life of methane in the atmosphere. Common NMVOCs include aromatic compounds benzene, toluene, 1,3-butadiene and xylene, which are suspected carcinogens and may lead to leukemia through prolonged exposure. A prominent NMVOC is benzo[a]pyrene, which is a polycyclic aromatic hydrocarbon, a known carcinogen with adverse health effects. (Sharma et al., 2013.).

Particulate matter (PM) are particles in the micron range of size, of solid or liquid suspended in a gas. In contrast, aerosol refers to particles and the gas together. They are usually classified by their aerodynamic diameter. Monitored for are usually PM<sub>10</sub> (10 micron) and PM<sub>2.5</sub> (2.5 micron), sometimes down to PM<sub>1</sub> (1 micron), due to peaks in distribution patterns at around 2.5 and 10  $\mu\text{m}$  aerodynamic diameter size. (Wang et al., 2015.) Naturally, they originate from volcanoes, dust storms, forest, grassland fires, etc. Anthropologically they primarily originate from burning fossil fuels. Increased PM levels are linked to health hazards such as heart disease, altered lung function and lung cancer. (Sharma et al., 2013.) A great risk regarding PM is that they are small enough to pass the blood brain barrier and seem to increase its permeability, further increasing risk. PM<sub>2.5</sub> is regarded the most harmful pollutant, is closely associated with premature death, and can penetrate deep into lung tissue due to its small size (Oppenheim et al., 2013.; Calderon et al., 2008.).

Smog is a pollution resulting from various processes and pollutants. We discern summer and winter smog. Summer smog usually comes from transportation and industrial emissions excited by ultraviolet light, which form secondary pollutants that also combine with the primary emissions to form photochemical smog. Also called “Los Angeles smog”, summer smog results in ground ozone production, NO<sub>2</sub> and PAN (Peroxyacetyl nitrate). In winter often a temperature inversion traps pollutant near the ground. Sulfurous smog, called “London smog” is caused by high SO<sub>2</sub> concentrations and results in sulfuric acid as a secondary pollutant. Other primary pollutants include nitric oxide, hydrocarbons, carbon monoxide (Sharma et al., 2013.; Tiao and Hemming, 1975.).

Peroxyacetyl nitrate (PAN -  $C_2H_3NO_5$ ) is a secondary pollutant similarly formed, as ozone, from  $NO_x$  and VOCs. At higher temperatures, PAN decomposes into  $NO_2$  and the peroxyacetyl radical. PAN is observed in conjunction with elevated ozone concentrations.

Persistent organic pollutants (POPs) are organic compounds that are resistant to environmental degradation through chemical, biological, and photolytic processes. They are transported over great distances and bioaccumulate and biomagnify in organisms. They include substances like PCBs, DDT and dioxins. Dioxins originate from industrial processes and combustion (municipal and medical waste incineration and backyard burning of trash). PCBs have been useful in industrial applications (electrical transformers and large capacitors, such as hydraulic and cooling fluids, and in paint and lubricant production). DDT, the notorious pesticide, is still used to control mosquitoes that carry malaria in some parts of the world. (Sharma et al., 2013.; EPA 2009.)

Metal vapors originate from vehicles, ore and metal processing, are also one of the constituents of PM. Lead (Pb) is a persistent pollutant, as it deposits in soils and remobilizes in the atmosphere from traffic. Emission rates are positively related with air and blood Pb. Other sources are waste incinerators, utilities, and lead-acid battery manufacturers. The highest air concentrations of lead are usually found near lead smelters. Lead is accumulated in the bones. Transition metals Ni, V, Fe, Cu, among others, participate in redox reactions, inducing oxidative stress and worsen the impact on health on a greater scale than other PM constituents (Sharma et al., 2013.; Chen et al., 2022.; Mielke et al., 2022.)

## ***ENVIRONMENTAL ASSESSMENT***

Environmental assessment is a critical process for understanding and mitigating the impacts of human activities on the environment. A systematic and technical evaluation must be done on how various activities affect the environment and requires a detailed and multifaceted assessment to develop effective mitigation strategies. Environmental assessment is an interdisciplinary process that integrates various branches of engineering and science. A purpose and need must be defined following with a detailed analysis of cumulative impacts, unavoidable adverse impacts, mitigation measures and alternatives, under the legislative environmental policy act.

Air pollution, particularly a concern in urban and industrial areas, is a major public health concern. Vulnerable groups, children, elderly, and people with pre-existing health conditions are more susceptible to exposure. The orographic and meteorological are mainly, other than radiative forcing, the components that govern atmospheric dispersion, wind, and precipitation intensity. The terrain complexity and other topographic features usually increase the dispersive effects, while precipitative events bring the pollutants down to the ground, allowing them to leach into the ground or be able to get redispersed again. When talking about air pollution it is also important to include the various chemical processes that are part of the pollutant's complex lifecycle.

### ***2.2.1 LEGISLATIVE***

At a pan European level, air emissions are regulated by the “United Nations Economic Commission for Europe” (UNECE) and the “Convention on Long-range Transboundary Air Pollution” (Air Convention). Under the “Air Convention”, the “Gothenburg Protocol” sets emission thresholds for NO<sub>x</sub>, NMVOCs, Sulphur oxides (SO<sub>x</sub>) and NH<sub>3</sub>. Reports on emission data on numerous air pollutants are also obligatory. The EEA compiles the annual EU emission inventory report under the Air Convention, in cooperation with the EU Member States and the European Commission. The Gothenburg

Protocol to abate acidification, eutrophication and ground-level ozone was founded in Gothenburg, Sweden in 1999., and has set the emission thresholds for 2010. - 2020. It formed the basis for the NEC Directive for the next two decades (EEA, 2023.).

In the EU air pollution is governed by the “EU Ambient Air Quality Directives” (AAQDs). They oblige members to follow legislation with the aim of having air pollution reduced to levels which limit harmful effects on human health and the environment, to improve and standardize the monitoring and assessment of air quality. This should put the EU on track to achieve zero air pollution by 2050.

The first pillar comprises “Directive 2008/50/EC” (2008 AAQ Directive) on ambient air quality and cleaner air and “Directive 2004/107/EC” (2004 AAQ Directive) on arsenic, cadmium, mercury, nickel, and polycyclic aromatic hydrocarbons in ambient air adopted in 2004). The first pillar establishes standards (limit values) for air quality monitoring and modeling.

The second pillar is comprised by the “Directive (EU) 2016/2284” (NEC Directive) on the reduction of national emissions of five main atmospheric pollutants, PM<sub>2.5</sub>, Sulphur dioxide (SO<sub>2</sub>), oxides of nitrogen (NO<sub>x</sub>), non-methane volatile organic compounds (NMVOCs) and ammonia (NH<sub>3</sub>). The NEC Directive requires national air pollution control programs to be established and to achieve reductions and improvements by 2020 and 2030, depending on the criteria.

The third pillar groups several EU legislative acts regulating air pollution depending on its two main sources. Industrial emissions are covered by “Directive 2010/75/EU” (IED) on industrial emissions, “Directive (EU) 2015/2193” (MCP Directive) on the limitation of emissions of certain pollutants into the air from medium combustion plant, “Directive 2009/125/EC” (Ecodesign Directive) establishing a framework for the setting of eco-design requirements for energy-related product.

Transport emissions cover “Regulation (EC) No 715/2007” on approved emission rates for vehicles (Euro 5 and Euro 6 standards), “Regulation (EC) No 595/2009” among other points, on access to vehicle repair and maintenance information.

Among other criteria the EU members oblige to:

- Divide their territories into zones and agglomerations for the purposes of air quality assessment and management and report on air quality zones designated under the Ambient Air Quality Directives
- Define common methods to monitor, assess and inform on ambient air quality in the EU
- Provide information to the public
- Establish objectives for ambient air quality to avoid, prevent or reduce harmful effects on human health and the environment
- Follow a deadline to achieve compliance with limit values under certain conditions and for certain pollutants
- Assessing ambient air quality should be based on common methods and criteria for air quality monitoring and modelling.
- Improves the legal framework, providing more clarity on access damage redress and effective penalties
- Defines a minimum content of national air pollution control programs

Two specific policies involved are “Environmental Impact Assessment” and “Strategic Environmental Assessment”.

Environmental Impact Assessment (EIA) Directive amended in 2014, requires major building or development projects to first be assessed for their impact on the environment, before the project can start. The EIA assesses the direct and indirect significant impact of a project of environmental factors including population and human health, biodiversity, land, soil, water, air, climate, landscape, material assets and cultural heritage. The authority is provided with a report containing the description of the project (location, design, size), potential significant effects, reasonable alternatives, features of the project and measures to avoid, prevent, reduce, or offset likely significant impacts on the environment.

The Strategic Environmental Assessment (SEA) Directive states a procedure when assessing a plan including scoping, environmental report, reasonable alternatives, public participation, monitoring. It is applied in a range of public programs for land use, transportation, energy, waste agriculture, etc. (EC, 2024.).

In Italy an IPPC permit under State jurisdiction is required, under Annex XII, part two of Legislative Decree 152/2006, for combustion installations with a thermal input of >300 MWt, gas reprocessing plants, refineries, integrated steel plants, large chemical plants, plants located at sea, etc. (MASE, 2024.).

	NH <sub>3</sub>	NMVOCs	NO <sub>x</sub>	PM <sub>2.5</sub>	SO <sub>x</sub>
Austria	● 5%	● -29%	● -51%	● -39%	● -58%
Belgium	● -15%	● -34%	● -57%	● -47%	● -83%
Bulgaria	● -1%	● -22%	● -50%	● -23%	● -95%
Croatia	● -23%	● -38%	● -49%	● -38%	● -92%
Cyprus	● -12%	● -51%	● -44%	● -52%	● -74%
Czechia	● -10%	● -32%	● -47%	● -44%	● -67%
Denmark	● -24%	● -31%	● -55%	● -43%	● -68%
Estonia	● -2%	● -15%	● -47%	● -43%	● -84%
Finland	● -22%	● -44%	● -50%	● -45%	● -67%
France	● -13%	● -35%	● -52%	● -44%	● -81%
Germany	● -16%	● -30%	● -40%	● -38%	● -46%
Greece	● -16%	● -57%	● -54%	● -48%	● -91%
Hungary	● -4%	● -34%	● -39%	● -7%	● -67%
Ireland	● 4%	● -6%	● -43%	● -34%	● -84%
Italy	● -17%	● -35%	● -53%	● -20%	● -81%
Latvia	● 4%	● -27%	● -27%	● -35%	● -58%
Lithuania	● -2%	● -20%	● -18%	● -23%	● -59%
Luxembourg	● 6%	● -27%	● -75%	● -52%	● -70%
Malta	● -17%	● -23%	● -56%	● -52%	● -98%
Netherlands	● -21%	● 1%	● -75%	● -51%	● -69%
Poland	● -10%	● -10%	● -31%	● -8%	● -65%
Portugal	● 0%	● -18%	● -52%	● -21%	● -79%
Romania	● -18%	● -28%	● -36%	● -3%	● -89%
Slovakia	● -23%	● -34%	● -45%	● -48%	● -83%
Slovenia	● -11%	● -37%	● -53%	● -38%	● -90%
Spain	● -6%	● -25%	● -53%	● -19%	● -90%
Sweden	● -11%	● -32%	● -40%	● -49%	● -55%

● Decrease in emissions compared to 2005      ● Increase in emissions compared to 2005

Figure 4. Emission reduction of the main air pollutants by European union states from 2005 to 2021, in compliance with the previous “Directive 2001/81/EC” (EEA, 2022.)



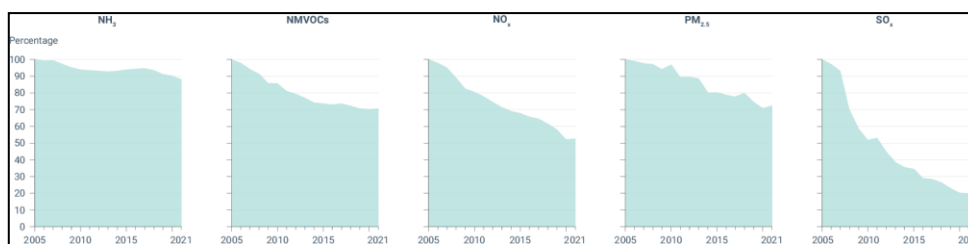


Figure 5. Percentage emission reductions time plots of main air pollutants from 2005. to 2021. (EEA, 2022.)

The national emission reduction commitments set out in the 2016 NEC Directive for years 2020-2029 and 2030 onwards are based on the estimated reduction potential of each member contained in the TSAP 16 report.

Member State	SO <sub>2</sub>		NO <sub>x</sub>		NMVOC		NH <sub>4</sub>		PM <sub>2.5</sub>	
	2020-2029	After 2030	2020-2029	After 2030	2020-2029	After 2030	2020-2029	After 2030	2020-2029	After 2030
Belgium	43%	66%	41%	59%	21%	35%	2 %	13 %	20 %	39 %
Bulgaria	78%	88%	41%	58%	21%	42%	3 %	12 %	20 %	41 %
Czech Rep.	45%	66%	35%	64%	18%	50%	7%	22%	17%	60%
Denmark	35%	59%	56%	68%	35%	37%	24%	24%	33%	55%
Germany	21%	58%	39%	65%	13%	28%	5%	29%	26%	43%
Estonia	32%	68%	18%	30%	10%	28%	1%	1%	15%	41%
Greece	74%	88%	31%	55%	54%	62%	7%	10%	35%	50%
Spain	67%	88%	41%	62%	22%	39%	3%	16%	15%	50%
France	55%	77%	50%	69%	43%	52%	4%	13%	27%	57%
Croatia	55%	83%	31%	57%	34%	48%	1%	25%	18%	55%

Table 1. Partial list of member states and their set reduction target values set by the NEC Directive, under the AAQDs (Directive (EU) 2016/2284, 2016.) The values are compared to 2005 data.

Pollutant	Italy reduction target	EU reduction target
SO <sub>2</sub>	-80%	-71%
NO <sub>X</sub>	-70%	-65%
PM <sub>2.5</sub>	-42%	-40%
NMVOCS	-50%	-46%
NH <sub>3</sub>	-17%	-16%

*Table 2. Reduction target values set by the NEC directive, under the AAQDs (ENEA, 2021.)*

Projections for Italy for the year 2030, predict reduction in SO<sub>2</sub> emissions, particularly in the maritime sector (-89% compared to 2010 values) and energy production (-59%). NO<sub>X</sub> emissions, primarily cut down in the road transport sector (-74%) and electricity generation (-46%). PM<sub>2.5</sub> will reduce due to abatement of ultrafine particulate emissions in the civil sector (-46%). Ammonia is the leading pollutant with the lowest reductions (-9% compared to 2010 values), targeting urea based fertilizers in the agricultural sector and zootechnical emissions.

### **2.2.2 MONITORING**

The main objective of environmental monitoring is to manage the impact various activities have on the environment, ensuring compliance with regulations and to mitigate risks on the environment and health. It is based on analyzing environmental monitoring data to arrive at relevant information, to be able to formulate a suitable response in a prompt manner. With the effects of emissions of polluting chemicals and industrial processes into the atmosphere, a need was created for environmental research, regulations, and air quality monitoring.

Air quality monitoring requires the integration of multiple environmental data sources, containing topographic data and meteorological, chemical, and physical parameters. As the quality of ambient air relates to the presence and concentration of substances regarded as pollutants. Air monitoring is tied to the determination of the local environmental, health and

social risk. More so, on the influence of different emission sources, pollutants, and their processes on air quality. Environmental data gathered using specialized observation tools, such as sensor networks and Geographic Information System (GIS) models, from multiple different environmental networks and institutes is integrated into air dispersion models, which combine emissions, meteorological, and topographic data to detect and predict concentration of air pollutants

From a practical point of view, the main objective of the work plan is to obtain representative samples and take removal actions. The cost of the characterization program (including the cost of the investment equipment installation, operation and maintenance and expected lifetime of the system), including the sampling period duration, number of discrete samples taken, the size of each sample, and the number of substances sampled must be defined. This allows for proper sanitation and timely response upon removal site assessment.

Sources are divided into mobile and stationary, while emissions can be steady or unsteady, and uniform or non-uniform. Sampling is done on a point area or volume. The data is assessed in two basic ways: modeling and measurement approach. Aside from meteorological factors (wind speed, direction precipitation, topography, temperature, air humidity, insolation) to take representative samples scheduling must be adapted to the variability, frequency, duration of the specific source. The best type of analysis to be performed must also be defined.

Air sampling is defined as those sampling and analytical techniques that require either off- or on-site laboratory analysis and do not provide immediate results. Air sampling techniques are more accurate in detection, identification and quantification of specific chemical compounds relative to most air monitoring technologies.

Air monitoring comprises the use of measuring instruments and other screening or monitoring equipment and techniques that provide instantaneous (real-time) data on the levels of airborne contaminants. Examples of air monitoring equipment are photoionization detectors (PID), flame ionization detectors (FID), oxygen/combustible gas detectors, and remote optical sensors.

For particulate matter filtration and impaction techniques can be used together with gravimetric analysis. High volume samplers are filtration units where all the particles above to the cut off diameter are measured. They have no preferred particle size because the measured variable is the total concentration of particles in suspension. They sample in the range of 0.8 2 m<sup>3</sup>/min of air. Photosensors are laser and optical sensors, that can also be portable, that measure light scattered from the particles which pass through the laser beam.

Absorption and adsorption methods are mainly used for gaseous pollutants, and specialized olfactometry or gas chromatography-mass spectrometry (GC-MS) for odor detection and analysis. Absorption is a sampling method comprised of the absorption of gases into a liquid medium for later analysis. This is effective for soluble gases like sulfur dioxide (SO<sub>2</sub>) and ammonia (NH<sub>3</sub>). Adsorption methods adsorb gaseous pollutants onto a solid medium, such as activated carbon, for analysis. This method is suitable for volatile organic compounds (VOCs) and other hydrocarbons). Using gas chromatography is expensive, requires technical assistance and calibration but offers speciation with low detection limit. Dynamic olfactometry is a standardized method used to measure and quantify odors in the environment. The odor concentration (OU [m<sup>-3</sup>]) is the number of times that the sample is diluted with odorless air to reach the “Odor detection threshold (ODT)” as decided by 50% of the panelist.

### 2.2.3 MODELLING

To accurately predict and describe the fate of the pollutants from the collected data a systemic approach must be taken, and many processes must be considered.

Emissions need to be accurately identified and quantified. Generally, they are described by pollutant load/emission rate ( $L$ , [ $\text{g s}^{-1}$ ]), which is equal to the product of the pollutant concentration ( $C$ , [ $\mu\text{ m}^{-3}$ ; ppm, ppb]) and gas flow rate ( $F$ , [ $\text{m}^3 \text{ s}^{-1}$ ;  $\text{L s}^{-1}$ ]).

The chemical reactions that pollutants undergo in the atmosphere, resulting in the formation of new compounds Advection, the horizontal transport of air and its pollutants, dispersion, the process by which pollutants spread from areas of higher concentration to areas of lower concentration due to vicinity of earth's complex surface and adiabatic turbulence (due to vertical temperature profile), must be considered when modelling, which has adverse consequences. This requires understanding of the wind shear, vertical velocity gradient, vertical temperature gradient, temporal changes in the Planetary boundary layer and many other meteorological parameters.

Cloud cover and insolation data is useful when calculating the mass balance of the boundary layer giving a more correct value to radiative fluxes. The stability of the lower earth's atmosphere largely depends on the vertical temperature distribution (fig. 6), meaning vertical motion is inhibited. Adiabatic lapse rate is a measurement of the rate at which a parcel of air cools as it rises, e.g. the rate of temperature change with height [ $^{\circ}\text{C km}^{-1}$ ]. Under radiation the earth's surface heats the atmosphere and produces a strong temperature gradient, thereby inducing adiabatic mixing and turbulence, where cool dense air falls to be heated, and then rises again to cool. Temperature inversions happen when the earth's surface cools, and relatively warm air sits on a cool layer above earth. This traps pollutants near the ground.

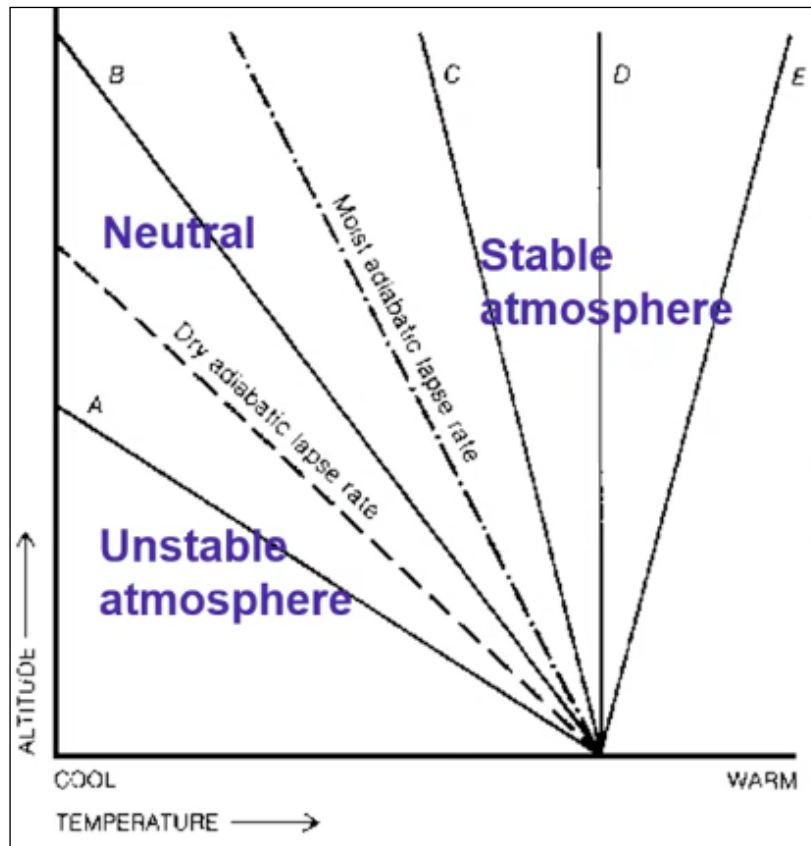


Figure 6. Variation of atmospheric stability due to vertical temperature distribution.

A) Absolute instability (strong negative temperature gradient)

- Dry-adiabatic lapse rate unsaturated parcels cool at a rate of  $10^{\circ}\text{C km}^{-1}$

B) Neutral condition or conditional stability - when the lapse rate between the dry and moist adiabatic lapse rate. Conditional instability exists when the atmosphere's stability depends on the saturation of the rising air

- Moist Adiabatic Lapse Rate – For a saturated parcel of air, i.e., when its  $T=T_d$ , then it cools at the moist adiabatic lapse rate =  $6^{\circ}\text{C km}^{-1}$
- Absolute stability occurs when the ELR is less than the moist adiabatic lapse rate.

C) Sub adiabatic: Ambient lapse rate  $<$  adiabatic. It indicates stable atmosphere, vertical motion, and mixing are suppressed. Dispersion is suppressed, and contamination is trapped.

D) Isothermal vertical temperature distribution, indicates stratification

E) Inversion means a hot top layer has trapped pollutants near ground

According to vertical temperature distributions at different times various plume behaviors form (fig. 8).

Looping Plumes occur when the emitted plume undergoes cyclical or oscillatory motion e.g. looping or meandering pattern. This happens with variations in direction or vertical wind speed profile, with high temperature gradients, associated with adiabatic convection e.g. during the day.

Coning Plumes are a behavior where the plume narrows or converges as it rises, resembling a cone shape. This can occur when the emitted substance has a higher velocity at the source and then spreads out as it rises and disperses due to pressure differences, in stable atmospheric conditions. Coning plumes are commonly observed in chimney emissions or exhaust from jet engines.

Fanning plumes occur when the wind blows across the path of the plume. The wind causes the plume to spread out horizontally, resembling a fan shape. These happen more often at night and early morning due to temperature inversion. Fanning plumes are influenced by wind direction and speed and are commonly observed in open areas or near bodies of water due to temperature inversion.

Lofting plumes exhibit significant buoyant effects and rise rapidly in the atmosphere. It is characterized by a strong upward displacement due to the temperature difference between the emitted substance and the surrounding air. Parts of the plume rise above the inversion layer. Lofting plumes are typically associated with sources that release heated gases or particles, such as wildfires, volcanic eruptions, or industrial processes involving high-temperature emissions. AERMOD also accommodates for “plume lofting”, the behavior of plumes as they rise and remain at the top of the CBL, before being mixed again.

Fumigating plumes descend and spread close to the ground, resulting in the widespread dispersal of the emitted substance. This happens in then there is a lapse, meaning fall in temperature with height in the bottom layer and an inversion in the upper layer e.g. inversion conditions with a negative upward radiative flux. They happen early in the morning or late in the afternoon. This type of plume behavior is often observed in pesticide or insecticide spraying activities, where the goal is to distribute the substance across a large area for fumigation purposes, but is very dangerous with emission near populated areas.

Trapping plumes occur when the emitted substance is confined or trapped within a specific area due to local atmospheric conditions or topographic features. This can happen when there is a stable layer of air above the plume that prevents it from dispersing vertically or horizontally. Trapping plumes are commonly observed in valleys or areas with strong temperature inversions.

Neutral plume refers to a type of plume that has minimal buoyancy effects and remains relatively stable as it disperses in the atmosphere. Neutral plumes commonly occur when the temperature of the emitted substances, such as gases or particles, is like the surrounding ambient temperature. As a result, the plume rises vertically without significant upward or downward displacement due to buoyancy forces.

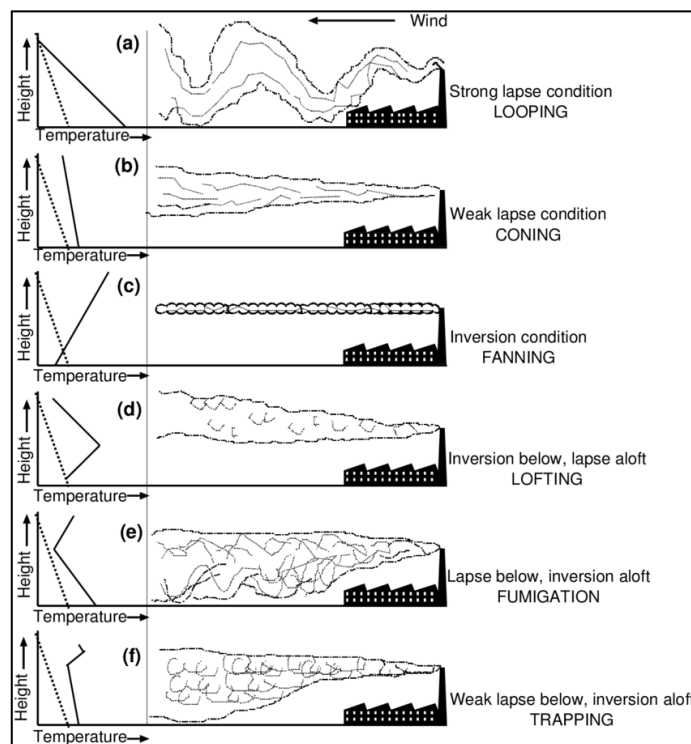


Figure 7. Different types of plume behavior for various atmospheric stability conditions. The dotted lines show dry adiabatic lapse rate, the solid lines represent the actual adiabatic lapse rates for different atmospheric stability conditions. Lapse is the normal temperature fall with height and inversion is the rise of temperature with height (Geiger, et al., 1995.)





*Figure 8. Plume emitted by API refinery in Falconara Marittima, Italy, aloft due to PBL stratification, lightly rising due to residual ground heat flux (Cronache Ancona, 2024.)*

Air dispersion/pollution modeling software is crucial for predicting concentrations of pollutants in the atmosphere. Mathematical models are indispensable in processing large quantities of different data points and producing accurate risk assessments and predictions. These tools help in assessing the potential impact of emissions from various sources, aiding in regulatory compliance, environmental impact assessments, and the development of mitigation strategies.

Air quality models use complex mathematical techniques to simulate physical and chemical processes, often in an iterative manner, that affect air pollutants as they disperse and react in the atmosphere. Air pollution modeling is based on several assumptions and simplifications, like no feedback between chemical species and flow fields (wind velocity, turbulent diffusivity, temperature). Generally, dilution and dispersion reduce concentration at given points. They consider various factors, such as meteorological conditions, topography, and the characteristics of the pollutant source. These models can simulate the transport, transformation, and deposition of pollutants over different spatial and temporal scales.

Dispersion models are mathematical tools used to simulate the distribution of air pollutants in the atmosphere, considering the aforementioned atmospheric processes. They are essential for predicting air quality and assessing the impact of emissions from various sources, based on empirical data and theoretical understanding of the processes. Some air dispersion models include chemical transport models (CTMs), describing both chemical and meteorological processes in the atmosphere, like the “Weather Research and Forecasting” (WRF) based model “WRF-Chem”.

Gaussian air pollution dispersion models are accurate and consistent, particularly in urban settings but lacking in the far field. Lagrangian models, that track pollutant parcels as they move through the atmosphere are effective in simulating the transport and dispersion of pollutants over long distances and complex terrains but can be computationally intensive, require detailed meteorological data and can suffer “numerical diffusion”. Eulerian models focus on fixed grid points and solve the advection-diffusion equation to simulate the dispersion and chemical transformation of pollutants over larger scales. Their output is limited to the discretization fineness of the grid resolution. (Atabi et al., 2016).

Gaussian (normal) distribution to describe the dispersion of pollutants in the atmosphere by assuming pollutants disperse in a bell-shaped concentration distribution profile. They are widely used for regulatory purposes and to assess the impact of point sources, such as industrial stacks. They are suitable for short-range dispersion (a few kilometers from the source).

Lagrangian models track the movement of individual pollutant particles or parcels of air due to atmospheric processes. The trajectory of each particle is calculated based on wind speed, direction, and turbulence. They shine in simulating pollutant dispersion over complex terrains and for long range transport. They can capture detailed pathways of pollutants and are often used in research and regulatory applications.

Eulerian models divide the atmosphere into a grid and calculate the concentration of pollutants in each grid cell over time. They use fixed reference points to solve the advection-diffusion equations that govern pollutant transport. Eulerian models are suitable for regional and urban air quality modeling, capturing the interactions between multiple sources and

atmospheric processes. They are used for regulatory assessments and to inform air quality management strategies.

Models combine features of Gaussian, Lagrangian, and Eulerian approaches to leverage the strengths of each method. For example, a hybrid model might use Gaussian methods for near-source dispersion and Eulerian methods for regional transport offering flexibility and accuracy for a wide range of applications, from local impact assessments to large-scale air quality forecasting.

AERMOD was developed by EPA in the 2000s and will be further discussed in this paper. It is a short range dispersion air dispersion model based on planetary boundary layer turbulence and scaling concepts, integrating terrain effects and building downwash.

CALPUFF is a non-steady-state puff dispersion model, suitable for long range transport and complex terrain developed in the late 90s. It models spatially and temporally variable meteorological conditions and is used for both regulatory and non-regulatory applications.

ADMS (Atmospheric Dispersion Modeling System) was developed in 1995. by “Cambridge Environmental Research Consultants” (CERC). It is a Gaussian plume model used for simulating pollutant dispersion from various sources, including industrial and road traffic. It accounts for complex terrain, buildings, and even deposition processes.

HYSPLIT (Hybrid Single Particle Lagrangian Integrated Trajectory) was developed in 1983. by “National Oceanic and Atmospheric Administration” (NOAA). Used for computing air parcel trajectories and dispersion or deposition of pollutants, suitable for both short and long range studies.

ISCST3 (Industrial Source Complex Short Term Model) was developed by EPA in the 80s. This Gaussian plume model predicts pollutant concentrations from industrial sources and is widely used for regulatory compliance. It served as a predecessor to AERMOD.

WRF-Chem (Weather Research and Forecasting model coupled with Chemistry) was developed in 2005. by the “National Center for Atmospheric Research” (NCAR). WRF-Chem is a regional-scale model that simulates both weather and air quality, accounting for interactions between pollutants and meteorological conditions.

Using models easily ensures that industrial and vehicular emissions meet air quality standards, the timely prediction of the potential impact of new developments on air quality can be made, areas at risk of high pollution levels are identified to implement adequate responses, and to predict the dispersion of hazardous pollutants during accidental releases. Air dispersion models face challenges with data quality and complexity (which continuously evolves to more accurately model atmospheric processes). In the past the problem was also due to computational resources, but that is becoming less of a problem today.

Future advancements in air dispersion modeling include the utilization of machine learning and artificial intelligence to improve model accuracy and efficiency. Integration of real time data to leverage real-time environmental data for dynamic and adaptive modeling. Developing more accessible software to facilitate broader use by non-specialists, which is partly what this thesis focuses on.

## *EPA*

The “United States Environmental Protection Agency” (EPA) was established in December 1970, in response to growing public concern about environmental pollution. The agency was formed through an executive order by President Richard Nixon, consolidating various federal research, monitoring, standard-setting, and enforcement activities into a single agency. The main objective is to protect human health and the environment by enforcing regulations based on laws passed by Congress. The agency's primary goal is to ensure that all Americans are protected from significant risks to human health and the environment where they live, learn, and work. EPA's headquarters are in Washington, D.C. Within EPA operating are, the “Office of Air and Radiation”, the “Office of Water”, the “Office of Chemical Safety and Pollution Prevention”, and the “Office of Enforcement and Compliance Assurance”. Each office focuses on specific aspects of environmental protection and regulation.

The “Clean Air Act” (CAA), one of the most significant environmental laws, was enacted in 1970 and has undergone several amendments. The CAA aims to regulate air emissions from stationary and mobile sources, ensuring that air quality meets health-based standards. The EPA sets and enforces these standards, known as the National Ambient Air Quality Standards (NAAQS), for key pollutants.

“Clean Water Act” (CWA) established in 1972, regulates discharges of pollutants into the waters of the United States and sets quality standards for surface waters, implements this law through pollution control programs, wastewater standards for industries, water quality standards for contaminants, etc.

Other important laws include the “Safe Drinking Water Act”, and the “Toxic Substances Control Act”, “Resource Conservation and Recovery Act”, “Endangered Species Act”, “Pollution Prevention Act”.

EPA is responsible for monitoring air quality across the nation, sets standards for key pollutants, and enforces compliance through a variety of mechanisms. This includes issuing permits for emissions, conducting inspections, and taking enforcement actions against violators. The EPA also works with state and local agencies to develop State Implementation Plans (SIPs) that outline how states will achieve and maintain compliance with NAAQS. The agency's permitting programs, such as the New Source Review (NSR) and Title V operating permits, ensure that new and modified sources of pollution comply with air quality standards.

The National Ambient Air Quality Standards (NAAQS) are central to the EPA's efforts to control air pollution. These standards are set for six key pollutants: particulate matter (PM<sub>10</sub> and PM<sub>2.5</sub>), ground-level ozone (O<sub>3</sub>), carbon monoxide (CO), sulfur dioxide (SO<sub>2</sub>), nitrogen dioxide (NO<sub>2</sub>), and lead (Pb). NAAQS are based on scientific evidence regarding the health and environmental effects of these pollutants. The EPA regularly reviews and updates the standards to reflect the latest scientific knowledge, ensuring that they provide adequate protection for public health and the environment.

The Atmospheric Environmental Research and Modeling Initiative (AERMIC) was established to enhance the EPA's capabilities in air quality modeling, charged with developing a replacement for the previous ISCST model based on state of the art science and data, like NLCD land cover data, use of NED elevation data to determine height of obstacles and detailed urban morphology data for several cities. AERMIC brought together experts from the EPA, the American Meteorological Society, and other stakeholders to develop advanced tools for predicting the dispersion of pollutants in the atmosphere. The primary objective was to create a robust, scientifically sound model that could be used for regulatory purposes and air quality assessments.

The development of the AERMOD (American Meteorological Society/Environmental Protection Agency Regulatory Model) was a collaborative effort initiated by AERMIC. AERMOD was designed to replace older models that were less accurate and less capable of handling complex environmental scenarios. The development process involved extensive research, testing, and validation to ensure that the model could provide reliable predictions of pollutant dispersion under a wide range of atmospheric

conditions. The performance of AERMOD has been extensively validated through numerous studies and evaluations. These assessments have compared AERMOD's predictions with observed pollutant concentrations in various settings, confirming its reliability and accuracy. The model's robust performance has led to its widespread acceptance and use by regulatory agencies, industry, and researchers around the world.

EPA's future directions:

- Reduce emissions that cause climate change and pollution
- Accelerate resilience and adaptation to climate change impacts
- Advance international and subnational climate efforts
- Enforce environmental laws and ensure compliance
- Ensure clean and healthy air for all communities and reduce localized pollution and health impacts
- Ensure safe drinking water and reliable water infrastructure
- Protect and restore waterbodies and watersheds
- Safeguard and revitalize communities
- Reduce waste and prevent environmental contamination
- Prepare for and respond to environmental emergencies
- Ensure safety of chemicals for people and the environment
- Chemical and pesticide safety

Future directions for AERMOD include incorporating new scientific findings, improving computational efficiency, and enhancing its ability to model complex scenarios such as urban environments and climate change impacts. Additionally, integrating AERMOD with other modeling tools and data sources can provide more comprehensive and accurate air quality assessments.

The EPA's work in air pollution management is critical to protecting public health and environment. AERMOD's development and implementation represented significant advancements in the field of atmospheric dispersion modeling, enabling more accurate and reliable predictions of pollutant behavior. As environmental challenges continue to grow, the ongoing refinement and application of models like AERMOD will be essential for developing effective regulatory strategies and ensuring clean air for all.

## ***2.4. PYTHON***

Programming has become an essential skill in the field of engineering, transforming how engineers design, analyze, and manage projects. Programming plays a critical role in various aspects of engineering, including design and simulation, data analysis, automation, and creating control systems.

Creating graphical user interfaces (GUIs) is a crucial aspect of software development, providing users with an intuitive way to interact with applications. Several programming languages are particularly well-suited for GUI development, each with its own set of advantages and drawbacks.

Java is a versatile, platform-independent language widely used in web, enterprise and mobile applications. Programs created in Java can run on any platform with a Java Virtual Machine (JVM). Java's Swing and JavaFX are two primary frameworks for creating GUIs also including Qt and wxWidgets. Swing and JavaFX provide a comprehensive set of components for building complex user interfaces. Its drawbacks include slower and less responsive apps compared to those created with some other languages.

C/C++ are object-oriented languages developed by Microsoft and offer fine-grained control over system resources and performance. It is often used in conjunction with the .NET framework, particularly Windows Presentation Foundation (WPF) and Windows Forms, to create Windows-based GUIs. It has applications in creating embedded systems, real time systems, simulation and modeling, high performance computing. Its drawbacks include a steep learning curve due to complex syntax and manual memory management and longer development time compared to higher-level languages. Also, its platform dependency is a drawback as the full capabilities of C languages are only utilized on Windows.

Python interpreted, high-level language known for its simplicity and readability. It was released in 1991, as Python 0.9.0, later with Python 1.0 in 1994. Python 3.0 was released in 2008. Popular frameworks for GUI development in Python include Tkinter, PyQt5, and Kivy. Python's



straightforward syntax makes it accessible for beginners, featuring an extensive standard library, and cross-platform compatibility. GUIs can be run on multiple operating systems with minimal changes. Python's wide range of libraries allows for rapid development and integration with other technologies.

Python is easy to learn and flexible but may suffer from performance issues. Python's interpreted nature can result in slower performance for GUI applications compared to compiled languages. The term "interpreted nature" refers to the way a programming language executes code. In an interpreted language, like Python, the source code is not directly translated into machine code (binary code) by a compiler before execution. Instead, the code is interpreted and executed line-by-line by an interpreter at runtime.

An interpreted language works in like Python in this example in the following steps:

Parsing - the interpreter reads the source code and parses it into a data structure that represents the program's logic

Execution - the interpreter then processes the parsed code, executing it one statement at a time. Each line of code is translated into machine instructions and executed on the fly.

Dynamic Typing - during execution, Python determines the data types and performs type checking dynamically, without the need for explicit type declarations.

Result - The output or result of each statement is immediately available, allowing for rapid development and testing.

Many other programming languages used in science and engineering are based on python, like MATLAB and R. The GUI development leverages Python's robust libraries for GUI design and data handling, ensuring a user-friendly and efficient tool. Since I have experience in working in these environments Python was chosen to create the graphical user interface.

## ***2.4. AERMOD***

AMS/EPA Regulatory Model (AERMOD) is a steady-state plume model that incorporates air dispersion based on planetary boundary layer (PBL) turbulence structure and scaling concepts (fig. 10.), including treatment of both surface and elevated sources, and both simple and complex terrain. The “American Meteorological Society” (AMS) and “U.S. Environmental Protection Agency” (EPA) formed the “AMS and EPA Regulatory Model Improvement Committee” (AERMIC) in 1991. and created AERMOD (US EPA, 2019.).

AERMOD is made for short range (up to 50km) dispersion modelling and assumes the concentration distribution to be Gaussian (fig. 17.) in both vertical and horizontal direction in the stratified boundary layer (SBL). In the convective boundary layer (CBL), also daytime planetary boundary layer, the horizontal distribution is assumed as Gaussian, but the vertical as bi-Gaussian (Equation 11; fig. 20; fig. 21) (EPA, 2023.).

AERMOD requires hourly surface and upper air meteorological observations for simulating pollutant dispersion, but such data is often unavailable. To overcome this, meteorological parameters are derived from high-resolution simulations using the Weather Research and Forecasting (WRF) model. An offline preprocessor has been developed to couple WRF with AERMOD, initializing the latter with hourly values derived from WRF outputs.

AERMOD consists of numerous preprocessors, processor and postprocessors which serve different functions. Some of them will be listed further.

AERMAP is a terrain preprocessor designed to work with the AERMOD dispersion model. It processes digital terrain data to generate the terrain inputs necessary for AERMOD, ensuring accurate representation of the ground elevation and terrain features around the area of interest. This preprocessing is crucial as terrain can significantly influence air pollutant dispersion patterns by affecting wind flow and atmospheric stability.

AERMAP calculates elevations and hill heights for receptors and sources, providing essential data to enhance the accuracy of AERMOD's predictions.

AERMET is the meteorological preprocessor for AERMOD, responsible for preparing the meteorological data input. It processes raw weather data, including surface and upper air observations, to generate the necessary parameters for AERMOD, such as wind speed, wind direction, temperature, and atmospheric stability.

AERMOD is an advanced air dispersion model used for predicting the dispersion of air pollutants from various sources. It incorporates state-of-the-art modeling techniques and handles both simple and complex terrain scenarios. AERMOD utilizes data from AERMAP and AERMET to simulate how pollutants disperse in the atmosphere, accounting for factors like terrain, weather conditions, and source characteristics. It is widely used in regulatory applications to assess air quality and ensure compliance with environmental standards.

AERPLOT is a post-processing tool used in conjunction with AERMOD. It helps visualize the dispersion modeling results by creating graphical representations of pollutant concentrations and distributions. AERPLOT can generate contour plots, concentration maps, and other visual aids that make it easier to interpret and communicate the results of AERMOD simulations. This visualization capability is essential for presenting findings in a clear and understandable manner, aiding in decision-making and regulatory compliance.

BPIPPRIME (Building Profile Input Program with PRIME enhancements), which is used to account for the effects of building downwash on pollutant dispersion. BPIPPRIME processes information about the physical dimensions and layout of buildings near emission sources, helping AERMOD accurately model how these structures influence air flow and pollutant spread.

AERMINUTE is another useful preprocessor that refines meteorological data by processing minute-by-minute wind data to improve the accuracy of the input for AERMET, especially in capturing short-term

variations in wind speed and direction. This level of detail is particularly important for modeling near-field dispersion in complex environments.

AERSURFACE is a preprocessor that estimates surface characteristics such as albedo, Bowen ratio, and surface roughness length, which are essential for characterizing the land use and surface features within the modeling domain. These parameters influence the atmospheric boundary layer and are critical for accurately modeling dispersion under various meteorological conditions.

AERMINUTE and AERSURFACE work together with AERMET to ensure that the meteorological inputs are as representative as possible, improving the overall reliability of AERMOD's predictions.

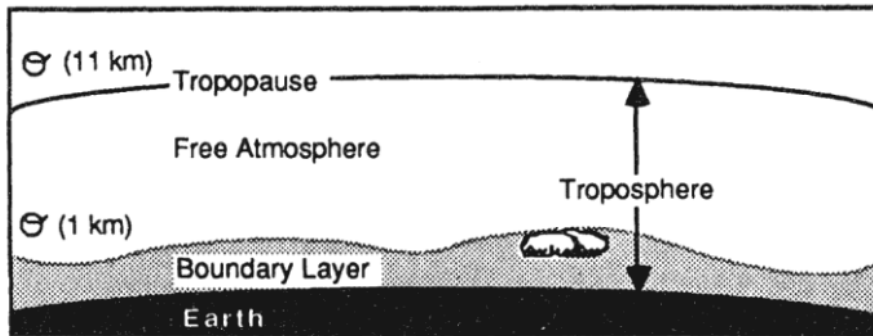
AERMAP, as previously mentioned, processes terrain data. Another related tool is AERGRID, which assists in defining receptor grids over complex terrain, ensuring that the receptors are placed accurately to capture the variations in terrain elevation effectively.

LEADPOST is a post-processor specifically designed for handling lead emissions and their dispersion, ensuring compliance with lead-specific air quality standards. It helps in interpreting the results of AERMOD simulations that involve lead, providing detailed analysis and reporting.

AERPOST is a general-purpose post-processor that can handle various types of AERMOD output, creating summaries, statistical analyses, and visual representations like tables and graphs. It simplifies the process of analyzing and communicating the results of dispersion modeling.

Some further parameters and behaviors are, as mentioned, influenced by the presence of stratification/convection in the PBL. The point of transition between the CBL and SBL (day to night) is defined as the point in time when the solar elevation angle  $\varphi = \varphi_{\text{crit}}$ . If solar radiation measurements are available AERMET determines " $\varphi_{\text{crit}}$ " from an estimate of cloud cover (EPA, 2023.).

Transport processes and interaction with the lower boundary of the troposphere (Earth's surface) modify the lowest 100 to 3000m of the atmosphere, creating the planetary boundary layer (**PBL**). Relatively high frequency of turbulent behavior is what differentiates the boundary layer from the rest of the earth's surface. A common approach in studying winds is dividing them into the "mean part" (advective contribution) and "perturbation part" (turbulent contribution), the latter being described as irregular swirls of motion called "eddies" (Stull, 2012.).



*Figure 9. Scheme of troposphere division into free atmosphere and boundary layer (Stull, 2012.)*

The boundary layer over land consists of three major parts (fig.9): Convective mixed layer (very turbulent), residual layer (less turbulent, containing former mixed layer air) and a nocturnal stable boundary layer (with sporadic turbulence). Another part is the surface layer where the wind is influenced by friction (Wyngaard, 1985.; NWS, 2024.)

The CBL is tied to solar radiation and starts about a half hour after sunrise (fig. 9). It grows in depth until the late afternoon by entraining or mixing less turbulent air from above down into it (fig. 9). The temperature profiles are adiabatic resulting in convective movement.

The residual layer forms about a half hour before sunset where turbulence decays. In the absence of advection particles will remain aloft in the residual layer during the night. It is neutrally stratified, resulting in turbulence of equal intensity in all directions. This creates dispersion rates equal in horizontal and vertical direction, creating a cone shaped plume.

The SBL is created by the contact of the residual layer with the ground. The stable air tends to suppress turbulence and vertical motion (fig. 11), while the developing nocturnal jet increases wind shears and generates turbulence, resulting in sporadic short burst turbulence.

A nocturnal jet is a fast-moving air current in the lower troposphere at nighttime. With air temperatures near the ground decreasing after sunset, an inversion layer is formed, where air temperature increases with height. Air then tends to flow horizontally and becomes less inhibited by turbulence and convection which tend to dominate during daytime when the ground surface heats up as a result of insolation. (Stull, 2012.; Davis, 2000.)

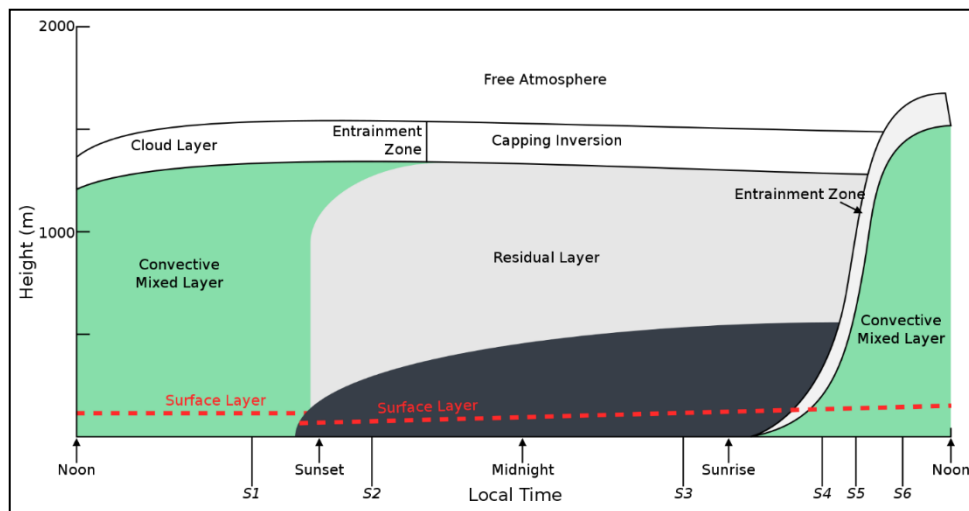


Figure 10. Time evolution of PBL (Stull, 1988.)

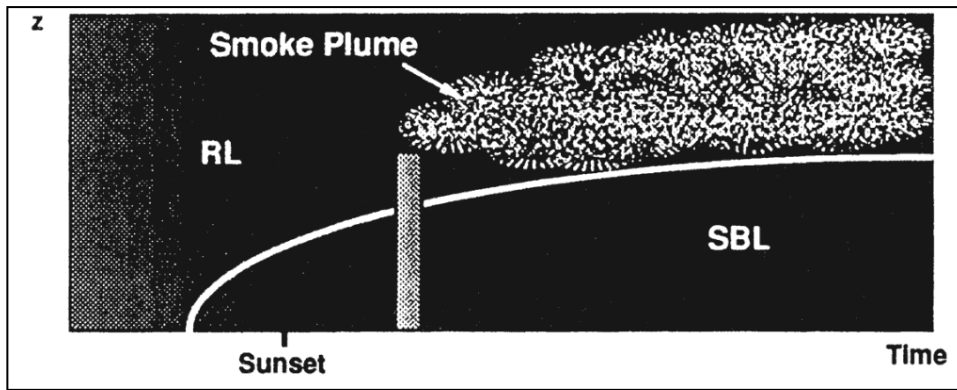


Figure 11. Lofting of a smoke plume occurring when the top of the plume grows upward into a neutral layer while the bottom is stopped by a stable layer

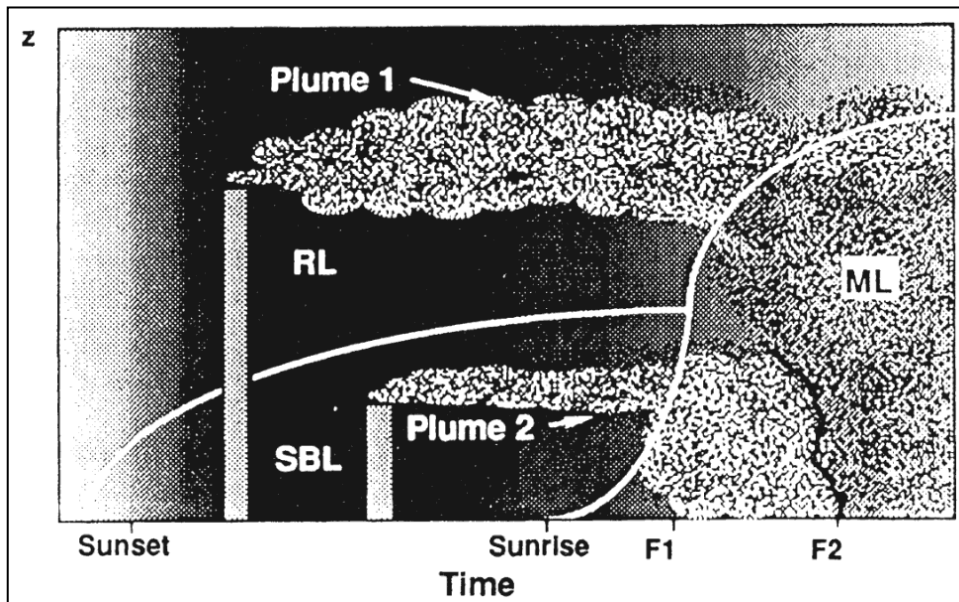


Figure 12. A growing mixed layer mixes elevated smoke plumes down to the ground e.g. fumigation

The energy balance in the PBL describes the transfer and transformation of energy in the lower part of the atmosphere. The change of parameters inside the balance influence vertical mixing and therefore pollutant dispersion. Its key components are:

**Net radiation** - which is often diverged into shortwave and longwave radiation, short wave radiation corresponds to the portion of solar radiation that reaches the earth's surface ~ 30% of total, and longwave corresponds to infrared radiation emitted by the earth's surface and atmosphere.

**Sensible heat flux** - the transfer of heat from the earth's surface to the atmosphere by means of convection and conduction.

**Latent heat flux** –the transfer of heat associated with phase change, e.g. evaporation and condensation.

**Soil heat flux** – energy gained or lost during belowground warming or cooling regarding the ground (Purdy, et al., 2016.)

They are influenced by a variety of surface and atmospheric conditions, and temporal changes (day/night). These include albedo, surface roughness, humidity, cloud cover, wind speeds, insolation and insolation length. During the CBL these parameters will be positive, while during the SBL these parameters will be negative, except for latent heat flux which will usually exhibit lower positive values, caused in part by the anomalies of water) (Mauder, et al., 2020.).

The Bowen ratio ( $B_o$ ) is a parameter defined as the ratio between sensible and latent heat flux.  $B_o < 1$  indicates a wet surface, while  $B_o > 1$  indicates a dry surface. Larger values produce greater updrafts and more intense buoyancy fluxes and therefore convection fluxes (Kang, 2016.). Values of the ratio are negatively related to surface air temperature, while the effect is exacerbated in less vegetated areas (Cho, et al., 2012.). It depends upon the underlying surface (e.g., dominant land-use and soil-type, latitude, elevation, continental location, drainage basin, etc.) and the time of year (Friedrich, et al., 2000.).

In AERMOD the structure and growth of the PBL is defined by an energy balance, determined by the heat fluxes and momentum drive. For the **CBL** it is defined as following:



Equation 1.

$$H + \lambda E + G = R_n \rightarrow H = \frac{0.9R_n}{\left(1 + \frac{1}{B_o}\right)}$$

Where:

H – Sensible heat flux [ $\text{W m}^{-2}$ ]

$\lambda E$  – Latent heat flux ( $\lambda E = H/B_o$ ;  $B_o$ -Bowen ratio) [ $\text{W m}^{-2}$ ]

G – Soil heat flux (assumed  $G = 0.1 * R_n$ ) [ $\text{W m}^{-2}$ ]

$R_n$  – Net radiation [ $\text{W m}^{-2}$ ], (EPA, 2023.)

The shear velocity ( $u^*$ ) (also friction or shear stress velocity) is one of the main scaling parameters in the description processes in the PBL, specifically in describing the vertical profiles. It depends on the turbulent state of the atmosphere and has many definitions, but generally it describes the turbulence intensity in the boundary layer and the transfer of momentum to the earth's surface. It is mainly influenced by surface roughness and affects the dispersive characteristics of airflow. It is generally defined as ( $\tau$ -surface shear stress;  $\rho$ -air density), with  $\tau$  being defined as a product of the horizontal (main) and vertical wind speed components (Weber, 1999.).

The von Karman constant is a widely used scalar value approximated to 0.4 that describes the relation between the exerted forces and the drag on the boundary surface, in this case between the advective turbulent motion of the air and earth's surface (Sheppard, 1947.).

Monin-Obukhov length (L) describes the effects of buoyancy on turbulence in the atmospheric surface layer. It symbolizes the height at which turbulence is predominantly generated by buoyancy, rather than wind shear.

$L < 0$  - unstable conditions,

$L > 0$  - stable conditions,

$L \rightarrow \infty$  - neutral conditions (EPA, 2019.; Bonan, 2019.)

The shear velocity ( $u^*$ ) and Monin-Obukhov length (L) are defined for **CBL** from the energy balance equation. The final estimated value of  $u^*$  is reached through iterative calculations of the following equation:

Equation 2.

$$u_* = \frac{ku_{ref}}{\ln(z_{ref}/z_0) - \Psi_m \{z_{ref}/L\} + \Psi_m \{z_0/L\}}$$

Where:

$u_*$  - Shear velocity [ $m s^{-1}$ ]

$k$  - von Karman constant (=0.4)

$u_{ref}$  – Wind speed at reference height e.g. advection [ $m s^{-1}$ ]

$z_{ref}$  – Height at which  $u=u_{ref}$  [m]

$z_0$  – Roughness length [m]

$L$  – Monin-Obukhov length [m]

$\Psi_m$  – Stability terms based on  $z_{ref}$ ,  $z_0$  and  $L$ , (EPA, 2023.)

After the final estimate of  $u_*$ ,  $L$  is calculated by:

Equation 3.

$$L = \frac{\rho c_p T_{ref} u_*^3}{k g H}$$

Where:

$L$  – Monin-Obukhov length [L]

$\rho$  – Air density [ $kg m^{-3}$ ]

$c_p$  – Specific heat of air at constant pressure [ $J g^{-1} K^{-1}$ ]

$T_{ref}$  – Ambient temperature of surface layer [K]

- Shear velocity [ $m s^{-1}$ ]

$k$  - von Karman constant (=0.4)

$g$  – Gravitational acceleration;  $g=9.807$  [ $m s^{-2}$ ]

$H$  – Sensible heat flux [ $W m^{-2}$ ], (EPA, 2023.)

For the **SBL**, the energy balance is highly site specific, so an empirical approach is used to determine the shear velocity ( $u_*$ ) and Monon-Obukhov length ( $L$ ), rather than defining a nocturnal energy balance and deriving the terms from them. The value is based on the drag coefficient in neutral conditions as based on the work of Qian and Venkatram (2011.)

The vertical structure is defined upon at least one measurement, but preferably a surface and upper air measurement of: 1. Wind direction, 2. Wind speed, 3. Temperature, 4. Vertical potential temperature gradient, 5. Vertical turbulence ( $\sigma_w$ ), 6. Lateral turbulence ( $\sigma_v$ )  
 Vertical wind speed profiles are determined for different heights separately, to minimize calculation times. This allows the wind speed below height  $7z_0$  to drop linearly.

Equation 4.

$$u = u\{7z_0\} \left[ \frac{z}{7z_0} \right], \quad \text{for } z < 7z_0$$

Equation 5.

$$u = \frac{u_*}{k} \left[ \ln\left(\frac{z}{z_0}\right) - \Psi_m\left\{\frac{z}{L}\right\} + \Psi_m\left\{\frac{z_0}{L}\right\} \right], \quad \text{for } 7z_0 \leq z \leq z_i$$

Equation 6.

$$u = u\{z_i\}, \quad \text{for } z > z_i$$

Where:

- u – Calculated wind speed [ $\text{m s}^{-1}$ ]
- Shear velocity [ $\text{m s}^{-1}$ ]
- k - von Karman constant (=0.4)
- $\Psi_m$  – Stability terms based on z,  $z_0$  and L

For CBL:

- $z_i$  – Mixing height (top of CBL) is assumed equal to 1000m
- $z_0$  – Starting height equal to 0.1m ( $z_0 = 0.0001z_i$ )
- L – Monin-Obukhov length = -10m ( $L = -0.01z_i$ )

For SBL:

- $z_i$  – Mixing height is assumed equal to 100m
- $z_0$  – Starting height equal to 0.1m ( $z_0 = 0.001z_i$ )
- L – Monin-Obukhov length = 10m ( $L = 0.1z_i$ ), (EPA, 2023.)

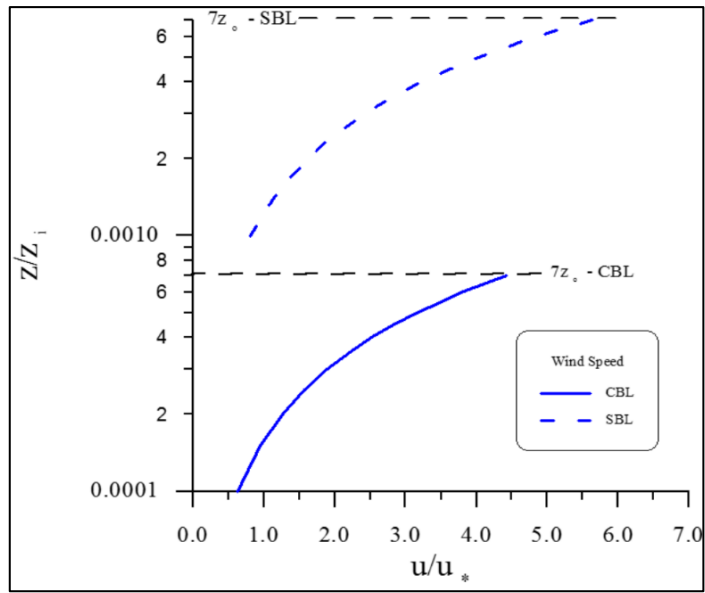


Figure 13. Vertical wind speed profile for CBL and SBL, in the region below  $7z_0$  (EPA, 2023.)

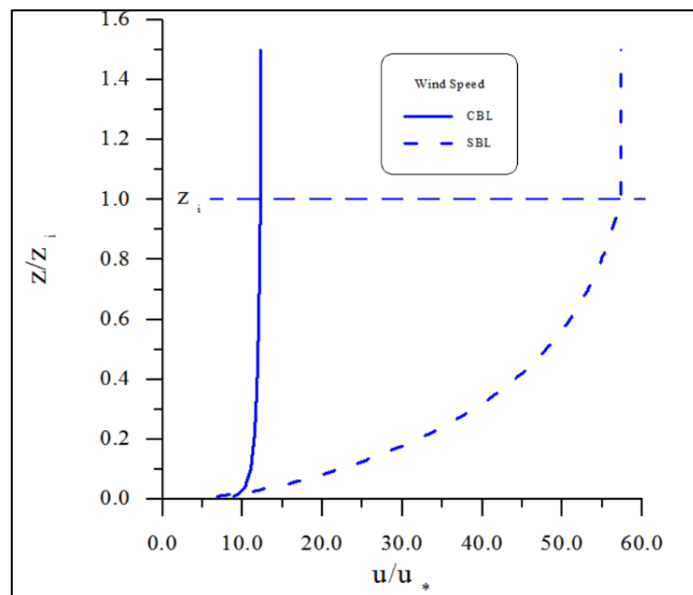


Figure 14. Vertical wind speed profile, for CBL and SBL, in the region above  $7z_0$  (EPA, 2023.)

Mixing height in the **CBL** is dependent on both mechanical (fig.8) and convective (fig.9) mixing and is determined as the greater of the two calculated heights. In the **SBL** it depends only on the mechanical mixing processes.

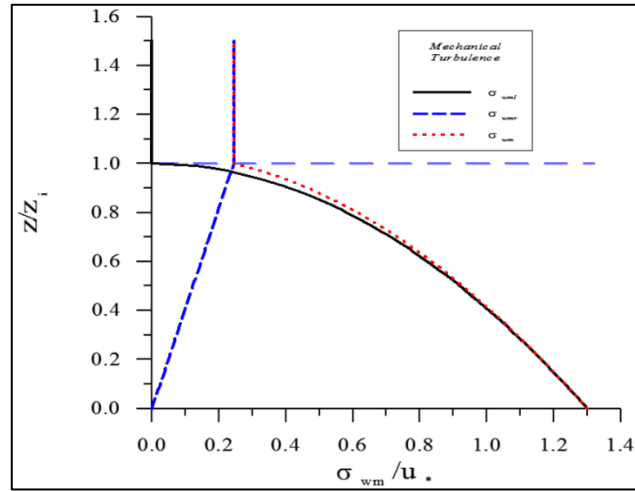


Figure 15. Mechanical portion of the vertical turbulence in the CBL, corresponding to total vertical turbulence of SBL (EPA, 2023.)

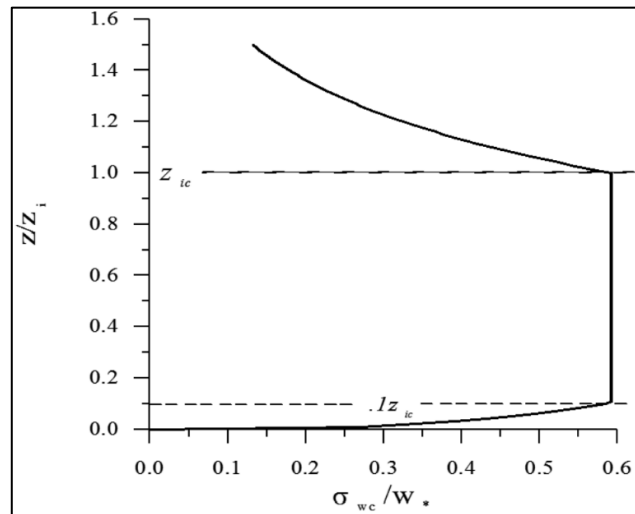
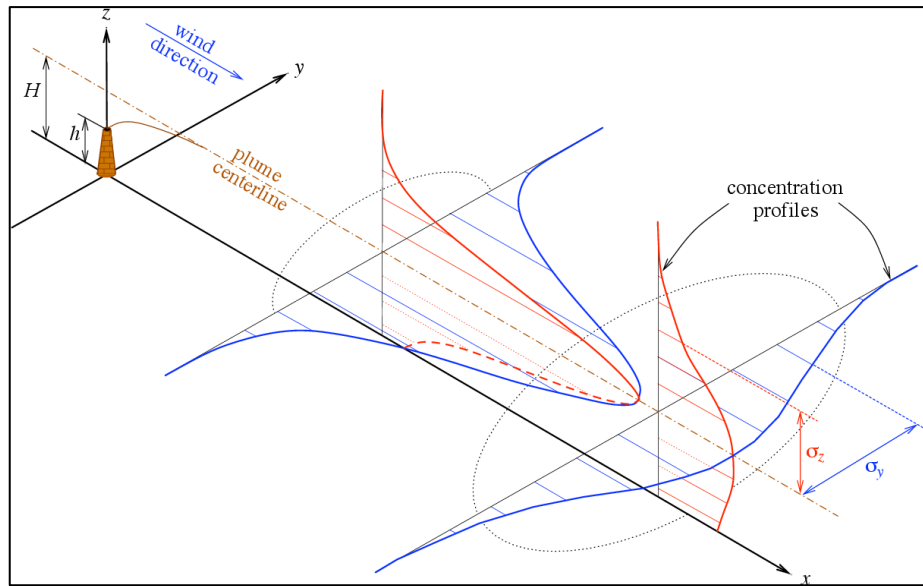


Figure 16. Convective portion of the vertical turbulence in the CBL (EPA, 2023.)



*Figure 17. A contaminant plume emitted from a continuous point source, with wind direction aligned with the x-axis. Profiles of concentration are given at two downwind locations, and the Gaussian shape of the plume cross-sections are shown relative to the plume centerline (Stockie, 2011.)*

The concept of dividing streamlines (Snyder et al., 1985.) is incorporated for flow in complex terrain, allowing the plume to be modelled either impacting or following the terrain and greatly simplifying terrain definition, and omitting the need for defining different complexities of the terrain (EPA, 2023.).

The plume is divided into “horizontal’ and “terrain following/dividing streamline parts” based on the dividing streamline height (Equation 7.; fig. 18). AERMOD calculates the concentration at the receptor position as sum of two concentrations: the concentration from horizontal plume (prominent in stable conditions) and that from terrain following plume (dominant in unstable conditions) (Venkatram, et al., 2001.).

Equation 7.

$$C = f_t \times C_h + (1 - f_t) \times C_t$$

Where:

$C$  – total concentration [ $\mu\text{g m}^{-3}$  or ppm]

$C_h$  – Concentration due to the horizontal flow component [ $\mu\text{g m}^{-3}$  or ppm]

$C_t$  – Concentration due to the terrain following/dividing streamline plumes [ $\mu\text{g m}^{-3}$  or ppm]

$f_t$  – Terrain weighing factor, (EPA, 2023.)

AERMAP, one of AERMOD's preprocessors uses gridded terrain data to calculate a representative terrain-influence height ( $h_c$ ) for each receptor with which AERMOD computes receptor specific  $H_c$  values.

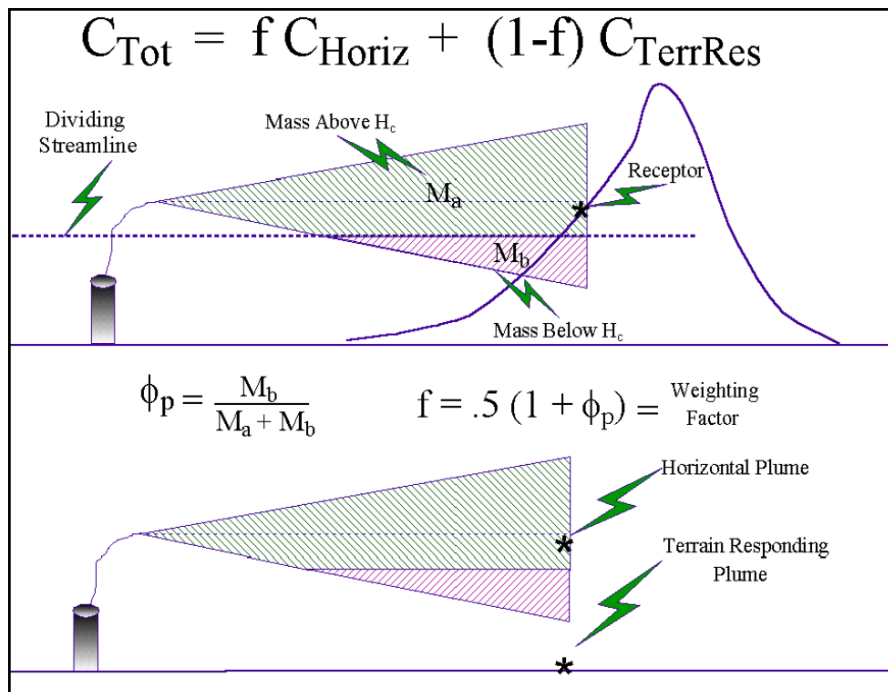
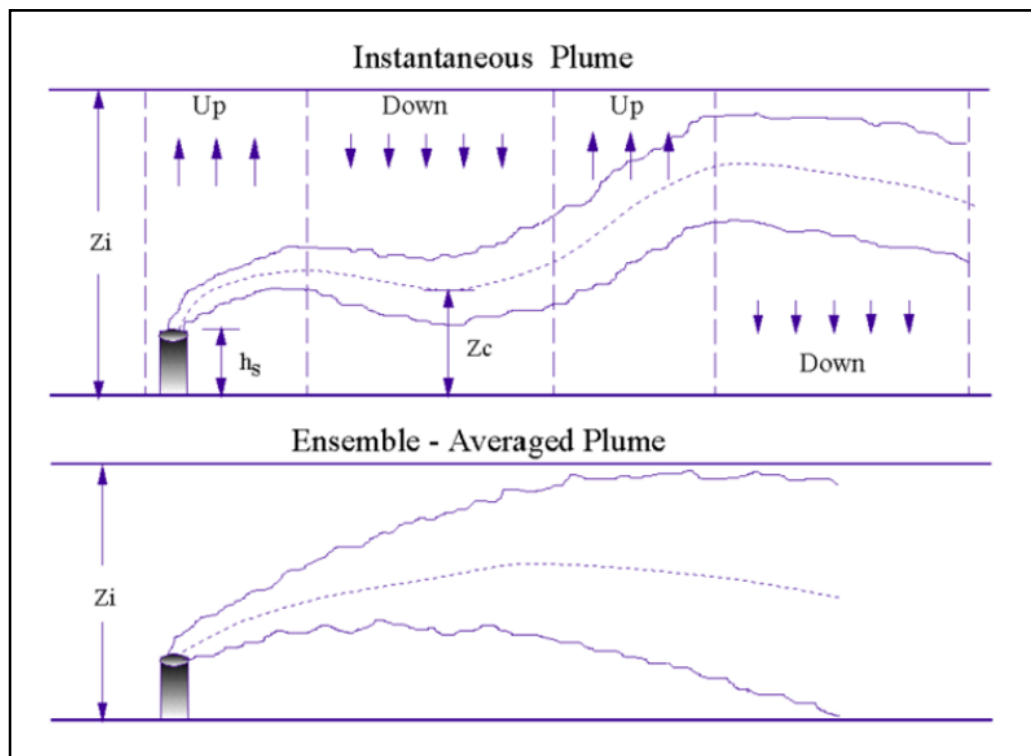


Figure 18. Terrain treatment in AERMOD, visualizing the concept of dividing streamlines and the construction of the weighting factor used in calculating total concentration (EPA, 2019.)

In the CBL, plume sections are emitted into a traveling train of convective elements, updrafts and downdrafts, moving with the mean wind. The PDF (probability density function) of the vertical velocity is positively skewed and results in a non-Gaussian vertical concentration distribution. An ensemble average of the plume volume is therefore calculated. Since a larger percentage of the instantaneous plume is affected by downdrafts, the ensemble average has a general downward trend (fig. 19) (EPA, 2019.).



*Figure 19. Instantaneous and corresponding ensemble-averaged plume in the CBL (EPA, 2019.)*

For buoyant releases, there is no “final” plume rise assumed (fig. 18). The direct transport of plume material to the ground is treated by the “direct” source located at the stack (Equation 8.). That is, the direct source treats that portion of the plume’s mass to first reach the ground and calculates all subsequent reflections of the mass. The direct plume material within the mixed layer that initially does not interact with the mixed layer lid.



For plume segments or particles initially rising in updrafts, an “indirect” or modified-image source is included (Equation 10.) (above the mixed layer) to address the initial quasi-reflection of plume material at  $z = z_i$ , e.g. at the top of the boundary layer. For the indirect source, a plume rise ( $\Delta h_i$ ) is added to delay the downward dispersion of material from the CBL top which mimics the plume’s lofting behavior.

Additionally, a “penetrated” source or plume (above the CBL top) is included to account for material that initially penetrates the elevated inversion but is subsequently re-entrained by and disperses in the growing CBL (Equation 9.). The penetrated plume material that is released in the mixed layer but due to its buoyancy, penetrates the elevated stable layer.

### Direct sources

Equation 8.

$$C_d(x, y, z) = \frac{Qf_p}{\sqrt{2\pi u}} F_y \sum_{f=1}^2 \sum_{m=0}^{\infty} \frac{\lambda_f}{\sigma_{zj}} \left[ \exp\left(-\frac{(z - \Psi_{dj} - 2mz_i)^2}{2\sigma_{zj}^2}\right) + \exp\left(-\frac{(z + \Psi_{dj} + 2mz_i)^2}{2\sigma_{zj}^2}\right) \right]$$

### Penetrated sources

Equation 9.

$$C_d(x, y, z) = \frac{Qf_p}{\sqrt{2\pi u}} F_y \sum_{f=1}^2 \sum_{m=0}^{\infty} \frac{\lambda_f}{\sigma_{zj}} \left[ \exp\left(-\frac{(z - \Psi_{dj} - 2mz_i)^2}{2\sigma_{zj}^2}\right) + \exp\left(-\frac{(z + \Psi_{dj} + 2mz_i)^2}{2\sigma_{zj}^2}\right) \right]$$

### Indirect sources

Equation 10.

$$C_d(x, y, z) = \frac{Qf_p}{\sqrt{2\pi u}} F_y \sum_{f=1}^2 \sum_{m=0}^{\infty} \frac{\lambda_f}{\sigma_{zj}} \left[ \exp\left(-\frac{(z - \Psi_{dj} - 2mz_i)^2}{2\sigma_{zj}^2}\right) + \exp\left(-\frac{(z + \Psi_{dj} + 2mz_i)^2}{2\sigma_{zj}^2}\right) \right]$$

Where:

$C_d(x,y,z)$  - Concentration due to a direct source at distance  $(x,y,z)$  [ $\mu\text{g m}^{-3}$  or ppm]

$Q$  - stack emission strength [ $\text{g s}^{-1}$ ]

$u$  - wind velocity [ $\text{m s}^{-1}$ ]

$\lambda_f$  - distribution coefficient

$\psi_{dj}$  - difference in height between the source base and plume centerline, e.g. effective source height [m]

$f_p$  - fraction of emitted contaminant that stays in the CBL ( $0 < f_p < 1$ )

$F_y$  - lateral distribution function with included meander

$z_i$  - height above the reflected surface in a stable layer [m]

$\sigma_{zp}$  - total vertical dispersion of penetrated force [m]

$\sigma_{zj}$  - vertical dispersion parameter [m]

$h_{ep}$  plume height that penetrated beyond the CBL [m]

$m$  - mass [g], (EPA, 2019.)

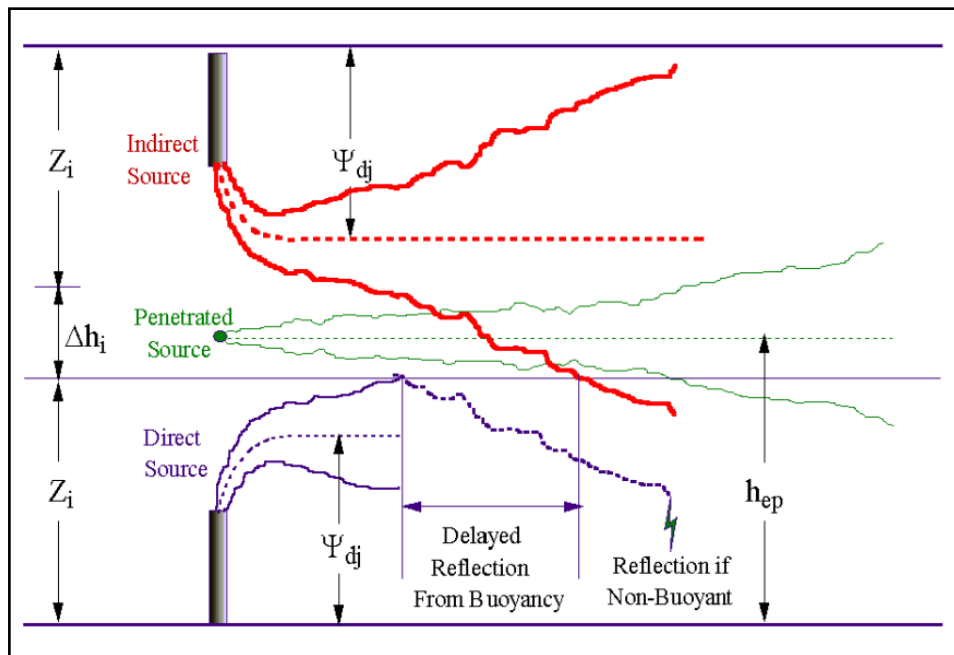


Figure 20. AERMOD's three plume treatments/interpretations of the CBL (EPA, 2019.)

For material dispersing within a convective layer, a plume embedded within a field of updrafts and downdrafts that are sufficiently large to displace the plume section within it. In the CBL a good approximation to  $p_w$  is the superposition of two Gaussian distributions. The instantaneous plume is assumed to have a Gaussian concentration distribution about its randomly varying centerline. The mean or average concentration is found by summing the concentrations due to all the random centerline displacements. This averaging process results in a skewed distribution which AERMOD represents as a bi-Gaussian PDF e.g. one for updrafts and one for downdrafts.

Equation 11.

$$p_w = \frac{\lambda_1}{\sqrt{2\pi\sigma_{w1}}} \exp\left(-\frac{(w - \bar{w}_1)^2}{2\sigma_{w1}^2}\right) + \frac{\lambda_2}{\sqrt{2\pi\sigma_{w2}}} \exp\left(-\frac{(w - \bar{w}_2)^2}{2\sigma_{w2}^2}\right)$$

Where:

$p_w$  - probability density function of the instantaneous vertical velocities

$\lambda_1$  and  $\lambda_2$  - weighting coefficients for the two distributions with  $\lambda_1 + \lambda_2 = 1$  and  $\lambda_2$  being larger (downdraft)

$w$  - random vertical velocity in the CBL [ $m s^{-1}$ ]

$\bar{w}_j$  - mean vertical velocity for the updraft ( $j = 1$ ) and downdraft ( $j = 2$ ) distributions [ $m*s^{-1}$ ]

$\sigma_w$  - vertical turbulence [ $m s^{-1}$ ], (EPA, 2019.)

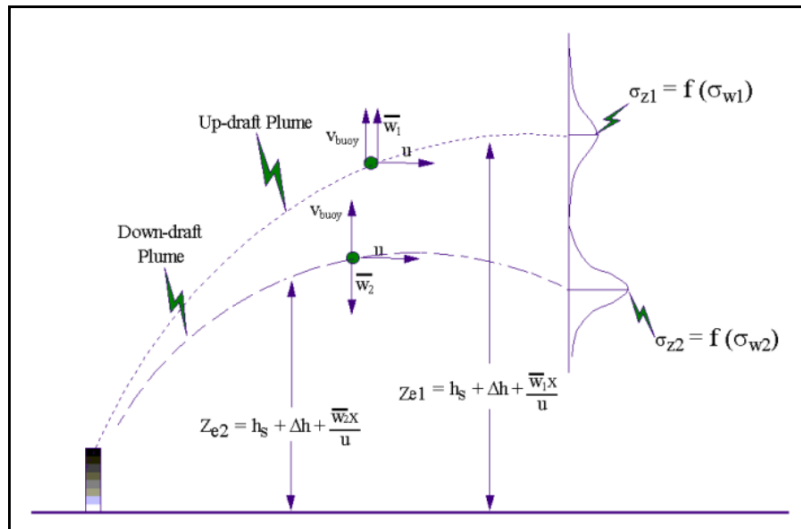


Figure 21. AERMOD's PDF approach for plume dispersion in CBL e.g. superimposition of two Gaussian distributions, the updraft and downdraft distribution (EPA, 2019.)

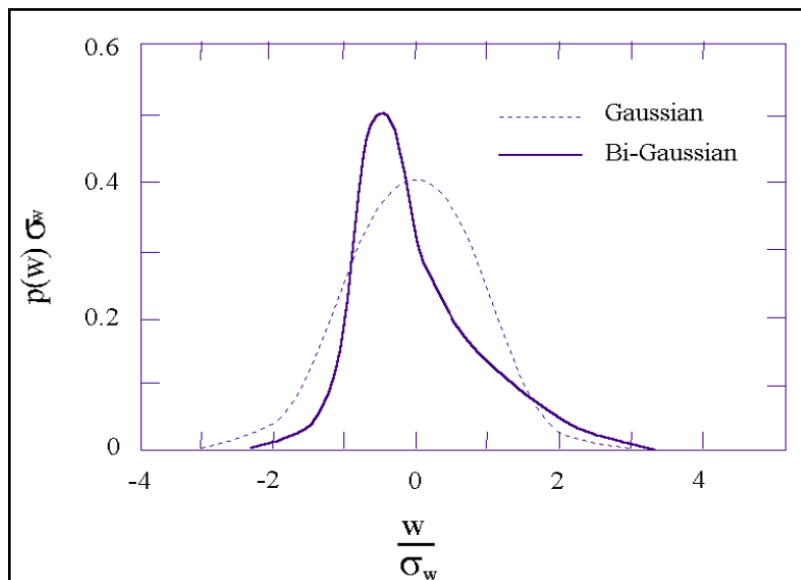


Figure 22. PDF of the vertical velocity. The bi-Gaussian curve has a skewness of  $S=1$ . About 60% of the  $p_w$  integral is on the negative side, the rest is positive, consistent with results of numerical simulations and field observations. (EPA, 2019.)

## 2.6. GRAPHICAL USER INTERFACE

The graphical interface is a python v3 based interface that enables users to create AERMAP, AERMOD and AERPLOT input files and to run different stages of AERMOD (AERMAP, AERMOD and AERPLOT) within the same program, without the use of “Command Prompt” or similar interpreters. It is suggested that the user creates a folder that will host all the data needed to run the stages. The executable files are located in their own folder and can access the input files from any location. There is no need to insert the executables or any other needed data into the folder, as it will be done automatically. Surface and upper air meteorological data were provided so there is no actual need to compile and run AERMET. The possible source types are point sources (POINT) and polygons (AREAPOLY).

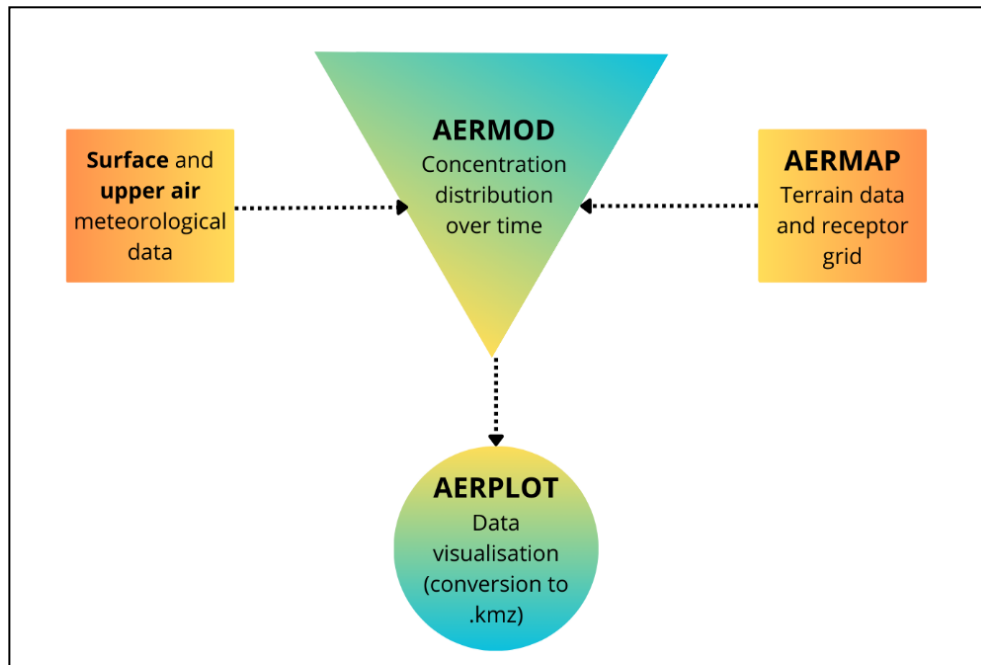


Figure 23. General flowchart of data processing

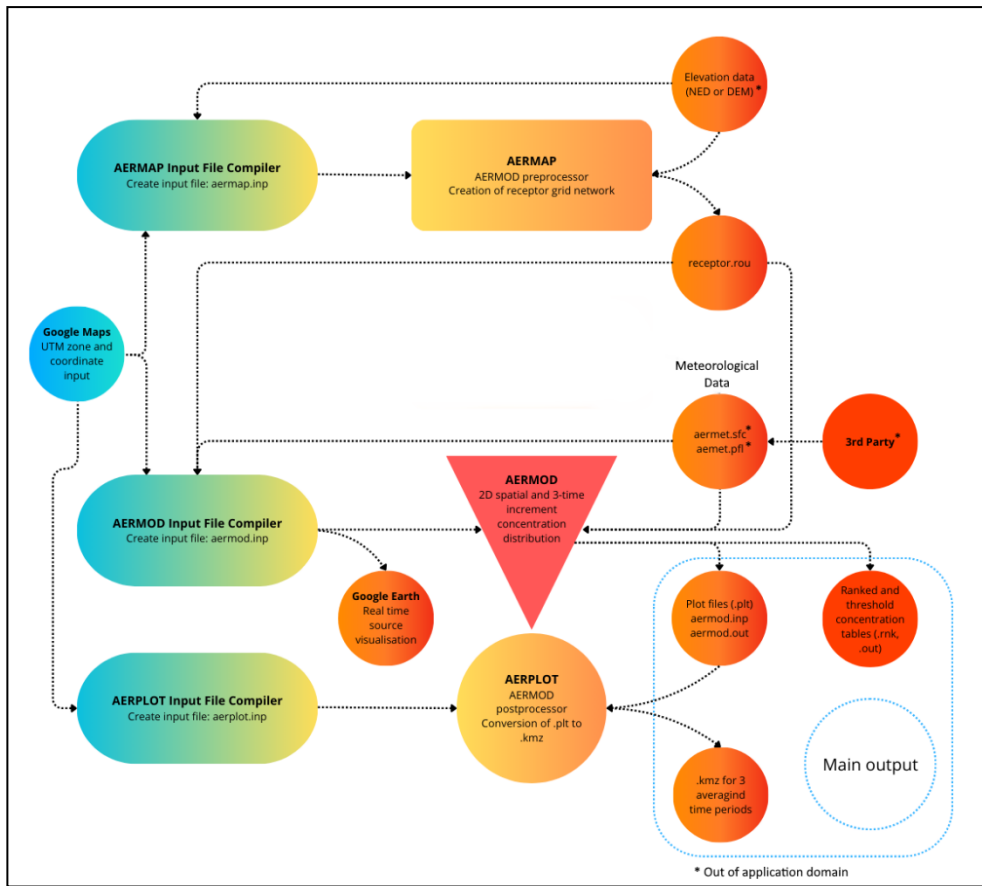


Figure 24. Detailed flowchart of data processing for the “CAIRO for AERMOD” app

AERMOD uses various input pathways to structure the necessary data for air dispersion modeling. These pathways include CO (Control), SO (Source), RE (Receptor), ME (Meteorology), and OU (Output). Each pathway has a specific function and format, contributing to the overall setup and execution of the model. When input files are compiled the pathway keyword is input, followed by specific data for each line.

The **CO (Control)** pathway defines the general settings and options for the AERMOD simulation. This includes specifying the time frame, pollutant of interest, averaging times, and other control parameters. It ensures the model is configured correctly for the specific scenario being studied.

The **SO (Source)** pathway describes the characteristics of the emission sources within the modeling domain. This includes defining the type of source (point, area, or volume), its location, emission rates, and physical parameters.

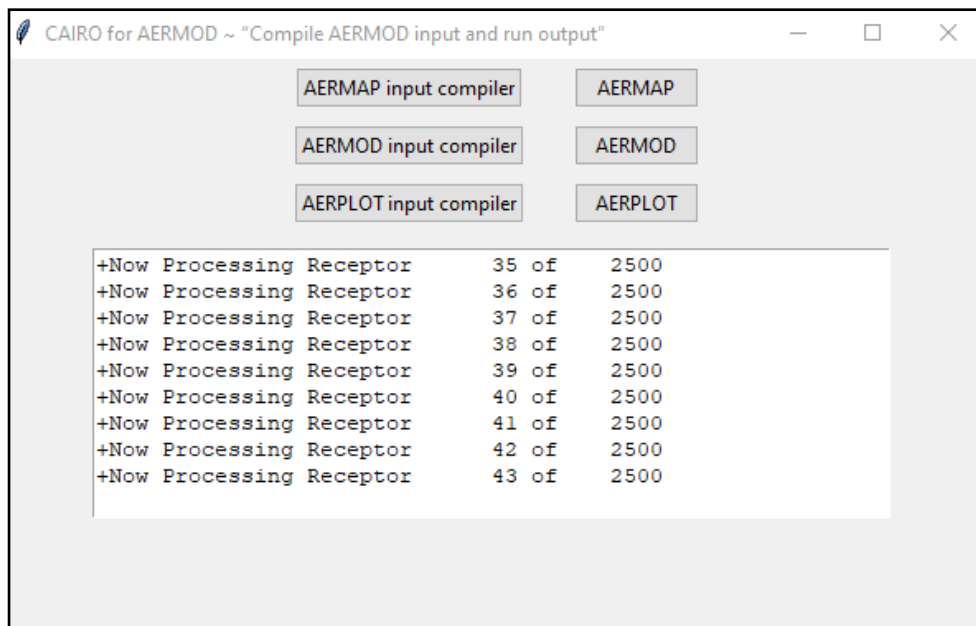
The **RE (Receptor)** pathway identifies where pollutant concentrations will be calculated. Receptors can be specified as discrete points, a grid, or along a line, and can include flagpole heights.

The **ME (Meteorology)** Meteorology pathway involves specifying the meteorological data needed for dispersion calculations. This data includes processed surface and upper air data files, which provide information on wind speed, wind direction, temperature, and atmospheric stability.

The **OU (Output)** pathway defines the types of output files and formats that AERMOD will generate. This includes specifying what concentration data to output, file formats, and any statistical processing required.

AERMOD and its pre and post processors, are usually run via the “Command Shell“. It includes manual assembly of input files, manually copying all needed data inside of a folder, then opening the “Command Shell“, navigating to the correct folder and running the application that way. It involves knowing correct AERMOD syntax, and manually relocating many files. The application commits that process by allowing the user to input raw data (with the aid of visualizations in Google Earth and Google maps), it being automatically compiled with correct syntax, needed data automatically copied to the correct path, and running the stages by defining a project folder, omitting the need of using the “Command Shell“ or manually relocating files.

The main window contains 6 buttons and an output textbox from which the whole process can be done. The first column contains buttons for opening the input file compilers. The buttons in the second column (AERMAP, AERMOD and AERPLOT) simply ask the user to navigate to the folder in which the needed input data is located, after which the corresponding executable is automatically run. AERPLOT expects that the folder contains 3 subfolders (aerplot1, aerplot2, aerplot3), which are automatically generated along with the needed input data upon running the AERPLOT compiler previously. Tooltips appear upon hovering over buttons and labels to guide the user through the process, or to explain the functionalities. Checkmarks will appear next to the buttons as the corresponding stage is done.



*Figure 25. The main window of the interface currently running AERMAP*



### ***2.6.1 AERMAP INPUT FILE COMPILER***

Originally to compile an AERMAP input file, the user has to define all grid data, with correct syntax, in a text editor and save the file as “aermap.inp”. To run AERMAP the user needs to relocate the “aermap.inp” and elevation data files in a folder containing “aermap.exe”, open the “Command Shell”, navigate to the folder and run AERMAP.

The application allows the user to input raw data inside of a GUI, it being automatically compiled with correct syntax and in correct file name and format, with the associated files mentioned in the input file, automatically copied into the project folder, omitting the need to know correct syntax or manual relocation of files. When running AERMAP, only the project folder needs to be selected, omitting the need for using the “Command Shell”, by making the “.exe” files fetch data from specific paths, not necessarily the one where the executable itself is located. This is possible for AERMET stages too, though compiling AERMET input files is not possible through the application, because most often ready surface and upper air data is used.

Upon opening the AERMAP compiler, a new window will be opened. The window contains input boxes for the generic data needed to create an AERMAP input file and tooltips guide the user about the functionality upon hovering over the labels.

When using DEM files, the FILLGAPS function is automatically inserted, but which is also not applicable for NED type files.

The “Orographic Files” button allows the user to choose multiple elevation rasters, which will be automatically copied into the project folder (the same folder the aermap.inp file will be saved in) and inputs the names of the rasters in the input file.

The “Open map” button opens “Google Maps”. By right clicking on the wanted location and left clicking on the coordinates, they are copied, automatically converted to UTM easting, northing and zone, and input into the interface as the anchor point. The user can also manually input the coordinates. The anchor point is defined as the bottom left corner of the grid.

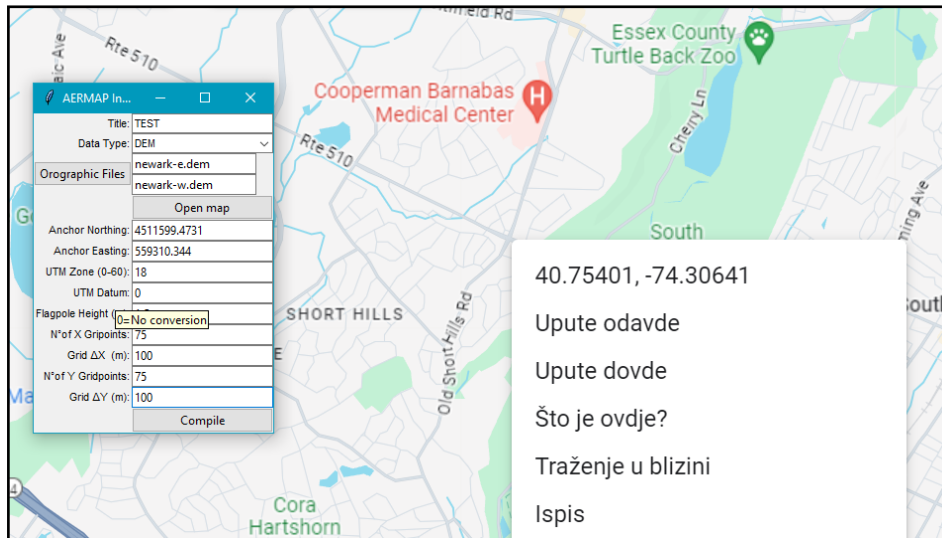


Figure 26. AERMAP compiler overlaying “Google Maps”

When the data is input, the user can compile the input file. Only the destination folder must be chosen (project folder) and the input file will be automatically named aermapi.inp, along with copying the needed elevation data into the folder and creating “RECEPT.rou” file (containing grid elevation data) and other output files.

```

CO STARTING
CO TITLEONE TEST
CO DATATYPE DEM FILLGAPS
CO DATAFILE newark-e.dem
CO DATAFILE newark-w.dem
ANCHORXY 557211.1242 4510238.0714 557211.1242 4510238.0714 18 0
FLAGPOLE 1.5
CO RUNORMOT RUN
CO FINISHED
RE STARTING
GRIDCART CART01 STA
XYINC 557211.1242 75 100 4510238.0714 75 100
GRIDCART CART01 END
RE FINISHED
OU STARTING
RECEPT RECEPT.ROU
OU FINISHED

```

Figure 27. Example of aermapi.inp file contents created with the AERMAP input file compiler

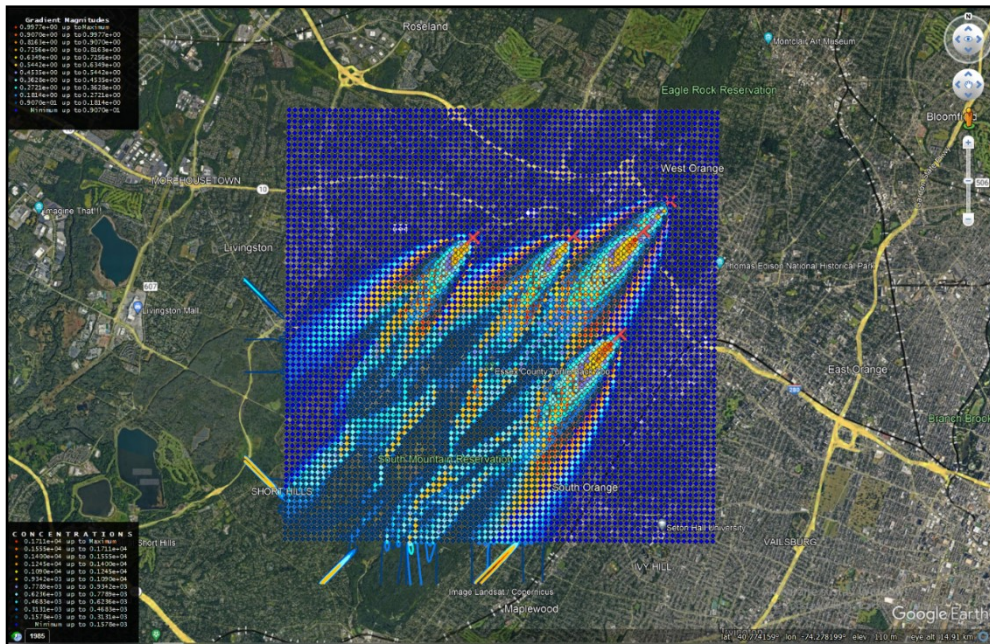


Figure 28. Grid receptor network (with already processed sources) created with the input file from figure 19. and figure 25., visualized in Google Earth

## 2.6.2 AERMOD INPUT FILE COMPILER

The process of running AERMOD is similar to that of AERMAP. Originally to compile an AERMOD input file, the user must define control, source, meteorological, receptor grid, and output data, with correct syntax, in a text editor and save the file as “aermod.inp”. The user cannot visualize sources until running a postprocessor like AERPLOT, this is only available in paid versions like “AERMOD View”. To run AERMOD the user relocates the “aermod.inp” and elevation data files in a folder containing “aermod.exe”, open the “Command Shell”, navigate to the folder and run AERMOD.

The application allows the user to input raw data inside of a GUI, it being automatically compiled with correct syntax, with the associated files mentioned in the input file, automatically copied into the project folder, omitting the need to know correct syntax or manual relocation of files.

Inputting sources is possible through Google Maps while sources are visualized in Google Earth, minimizing user error and acting as a real time

visual aid. When running AERMOD, only the project folder needs to be selected, omitting the need for using the “Command Shell”, by making the “.exe” files fetch data from specific paths, not necessarily the one where the executable itself is located.

Upon opening the AERMOD compiler, a new window will be opened. The window contains input boxes for the generic data needed to create an AERMOD input file, while tooltips guide the user about the functionality upon hovering over the labels. Navigation is done by mouse wheel (Ctrl+mouse wheel results in horizontal movement) and switching between textboxes by “Tabulator” button.

AERMOD Input File Generator

Title:

Time period 1 (h):

Time period 2 (h):

Time period 3 (h):

Pollutant:

Flagpole/Receptor Height (m):

Add Point Source

Add Polygon Area Source With Google Maps

Add Polygon Area Source Manually

Group name:

Choose Receptor File

Choose Surface Meteo Data File

Choose Meteo Profile Data File

Surface Data Station Number:

Start Year:

Upper Air Data Station Number:

Start Year (Upper Air):

Base Elevation (m):

Start Date (YYYY MM DD):

End Date (YYYY MM DD):

Rec Table (1ST 2ND 3RD):

Max Table N° of entries:

Hinum for 1st Time period Rank:

Hinum for 2nd Time period Rank:

Hinum for 3rd Time period Rank:

Threshold 1st Time period:

Threshold 2nd Time period:

Threshold 3rd Time period:

Compile

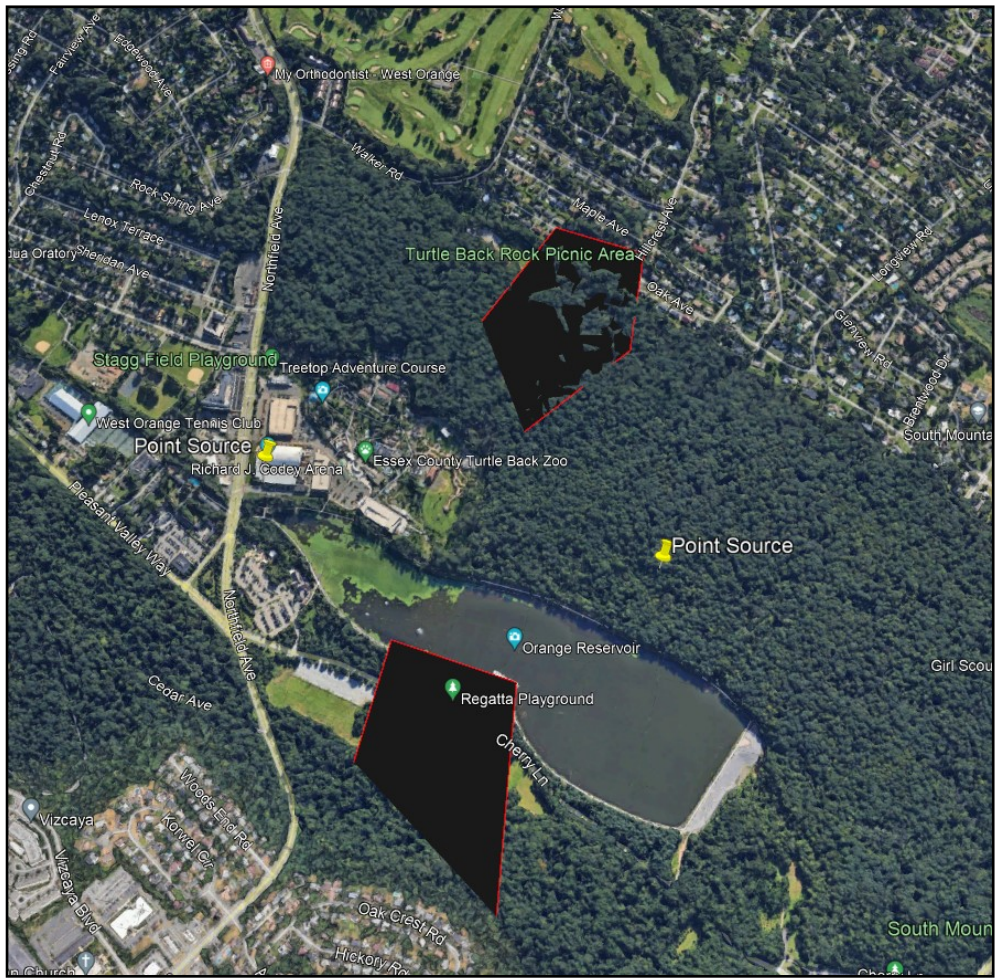
Figure 29. AERMOD input file compiler GUI

Upon addition of point sources, the frame will expand and allow for entry of physical and geometric parameters, and UTM coordinates. The coordinates can also be input using the “Choose on map” button. This action opens “Google Maps” and “Google Earth”. As coordinates are copied from “Google Maps” they are automatically converted to UTM coordinates and input into the interface, also the point source will appear in “Google Earth”, where the elevation can be conveniently read (Figure 23.). Manual input is also possible. This action is repeatable for multiple point sources. Clicking on the “Add Polygon Area Source With Google Maps” button, similarly opens “Google Maps” and “Google Earth”. As coordinates are copied from “Google Maps” they are automatically converted to UTM and added as vertices. Polygons also appear in “Google Earth” (Figure 30.).

The screenshot displays the AERMOD Input File Generator interface. It features several sections for data entry:

- Title and Time Periods:** Title: TEST; Time period 1 (h): 1; Time period 2 (h): 8; Time period 3 (h): 24.
- Pollutant and Height:** Pollutant: PM10; Flagpole/Receptor Height (m): 1.5.
- Point Sources:** Two point sources are defined with parameters like Emission Rate (g/s), Stack Height (m), Exit Temp (K), and Stack Diameter (m).
- Area Sources:** Two area sources are defined with vertices (Northing/Easting coordinates), Emission Rate (g/s), Release Height (m), and Initial Source Height (m).
- Receptor and Meteorological Data:** Fields for Group name (TEST), Receptor File (RECEPT.ROU), Surface Station Number (14737), and various meteorological parameters like Start Year, Base Elevation, and Rec Table.
- Buttons:** Multiple "Open map" buttons are present for selecting coordinates. A "Compile" button is at the bottom.

Figure 30. AERMOD input file compiler with input information



*Figure 31. Added point sources and polygon sources automatically visualized in real time using “Google Earth”.*

When manually inputting polygon coordinates, vertices are added using the “Add Vertex” button (Fig. 32).

The screenshot shows a software interface with the following components:

- Top Section:** Input fields for Title, Time period 1 (h), Time period 2 (h), Time period 3 (h), Pollutant, and Flagpole/Receptor Height (m). Below these are buttons for "Add Point Source" and "Add Polygon Area Source With Google Maps".
- Area 1 Vertices:** A grid of input fields for Northing and Easting coordinates (Northing 1-5, Easting 1-5).
- Area 2 Vertices:** A grid of input fields for Northing and Easting coordinates (Northing 1-7, Easting 1-7).
- Source 1 Parameters:** Input fields for Emission Rate (g/s) 1, Release Height (m) 1, N° Vertices 1, and Initial Source Height (m) 1. A button "Add Vertex" is positioned to the left of these fields.
- Source 2 Parameters:** Input fields for Emission Rate (g/s) 2, Release Height (m) 2, N° Vertices 2, and Initial Source Height (m) 2. A button "Add Vertex" is positioned to the left of these fields.
- Manual Addition Section:** A button "Add Polygon Area Source Manually" is located below the source parameters. Below it are input fields for Group name, Choose Receptor File, Choose Surface Meteo Data File, and Choose Meteo Profile Data File.
- Receptor and Data Parameters:** A series of input fields for Surface Data Station Number, Start Year, Upper Air Data Station Number, Start Year (Upper Air), Base Elevation (m), Start Date (YYYY MM DD), End Date (YYYY MM DD), Rec Table (1ST 2ND 3RD), Max Table N° of entries, and three Hinum (Hinimum) values for 1st, 2nd, and 3rd time periods. Below these are three Threshold input fields for 1st, 2nd, and 3rd time periods.
- Bottom Section:** A "Compile" button is located at the bottom right of the interface.

Figure 32. Manually adding polygon sources and vertices

The group name is important as it defines the name of the output files along with time periods (For example here: PLOT24H\_TEST.PLT). The receptor file, surface and upper air meteorological data are chosen in an explorer window and are automatically copied into the same folder as the compiled “aermod.inp” file (if files with the same name exist, they won’t be copied, but will still be listed in the text of the input file).

```

CO STARTING
CO TITLEONE TEST
CO MODELOPT DFAULT CONC
CO AVERTIME 1 8 24
CO POLLUTID PM10
CO FLAGPOLE 1.5
CO RUNORNOT RUN
CO FINISHED

SO STARTING
SO ELEVUNIT METERS
SO LOCATION STACK1 POINT 563808.5619 4516113.7612 172
SO LOCATION STACK2 POINT 563352.1983 4515589.5316 156
SO LOCATION STACK3 POINT 562144.4309 4515497.6694 168
SO LOCATION STACK4 POINT 562967.8961 4513810.8131 103
SO LOCATION STACK5 POINT 560413.9124 4515452.7305 157
SO SRCPARAM STACK1 85 20 305 20 0.7
SO SRCPARAM STACK2 65 15 300 15 0.85
SO SRCPARAM STACK3 125 25 320 12 0.8
SO SRCPARAM STACK4 125 20 305 15 0.75
SO SRCPARAM STACK5 98 15 315 17 0.85
SO SRCGROUP TEST STACK1 STACK2 STACK3 STACK4 STACK5
SO FINISHED

RE STARTING
RE INCLUDED RECEPT.ROU
RE FINISHED

ME STARTING
ME SURFFILE aermet.sfc
ME PROFFILE aermet.pfl
ME SURFDATA 14737 1992
ME UAIRDATA 00014735 1992
ME PROFBASE 67 METERS
ME STARTEND 1992 8 15 1992 8 17
ME FINISHED

OU STARTING
OU RECTABLE ALLAVE 1ST 2ND
OU MAXTABLE ALLAVE 100
OU RANKFILE 1 50 RANK1.RNK
OU RANKFILE 8 50 RANK8.RNK
OU RANKFILE 24 50 RANK24.RNK
OU MAXIFILE 1 TEST 540 MAX1H_TEST.OUT
OU MAXIFILE 8 TEST 85 MAX8H_TEST.OUT
OU MAXIFILE 24 TEST 12.5 MAX24H_TEST.OUT
OU PLOTFILE 1 TEST FIRST PLOT1H_TEST.PLT
OU PLOTFILE 8 TEST FIRST PLOT8H_TEST.PLT
OU PLOTFILE 24 TEST FIRST PLOT24H_TEST.PLT
OU FINISHED

```

*Figure 33. Example of an “aermod.inp” file contents created with the AERMOD input file compiler*



### ***2.6.3 AERPLOT INPUT FILE COMPILER***

Again, the process of running AERPLOT is like that of AERMAP and AERMOD. Originally to compile an AERPLOT input file, the user must define a variety of parameters (UTM zone, hemisphere, datum conversions, visualization options, etc.) and generic AERPLOT input, with correct syntax, in a text editor and save the file as “aerplot.inp”. To run AERPLOT the user relocates the “aerplot.inp” and AERMOD outputs (including plot file) in a folder containing “aerplot.exe”, open the “Command Shell”, navigate to the folder and run AERPLOT.

The application allows the user to input raw data inside of a GUI, it being automatically compiled with correct syntax, with the associated files assigned through the input action automatically copied into the project folder, omitting the need to know correct syntax or manual relocation of files. With multiple averaging periods, this is done iteratively, with all periods being analyzed at once. Inputting UTM zone is possible through Google Maps. When running AERPLOT, only the project folder needs to be selected, omitting the need for using the “Command Shell”, in this case by relocating the “.exe” file to the correct path and running it automatically.

The AERPLOT input file compiler opens in a new window and also features tooltips which guide the user through the process. The automatic opening of .kmz files in “Google Earth” is disabled.

Version should be input as 2 by default, this option exist for potential future upgrades. UTM zone can be input manually or again by clicking on the “Open map for UTM zone” button, which opens “Google Maps” and upon copying of coordinates automatically inputs the UTM zone.

The time periods and group name must be identical to the ones set in the “aermod.inp” file, as the needed files are named according to them, and will be automatically fetched and copied into the AERPLOT subfolders.

Minimum and maximum bin can be set at will or conformed to the data using the input: data. Binning methods (also for gradient) can be chosen to be linear or logarithmic, while they can also be disabled. Grid rows and columns should be set at 400, for computational ease, though they can be set at larger values for larger files. The number of smoothing iterations distorts locations as it is increased, so a value of 1 is recommended.

Upon compiling the user is asked to choose the input folder (again the project folder), which will automatically create 3 subfolders (aerplot1, aerplot2 and aerplot3) corresponding to the 3 time periods. The “aermod.inp”, “aermod.out”, “aerplot.inp”, aerplot.exe and corresponding “.plt” file will be automatically copied in each of the subfolders. Afterwards when running AERPLOT from the main window the program will expect all the files to be in place before running, otherwise an error will ensue.

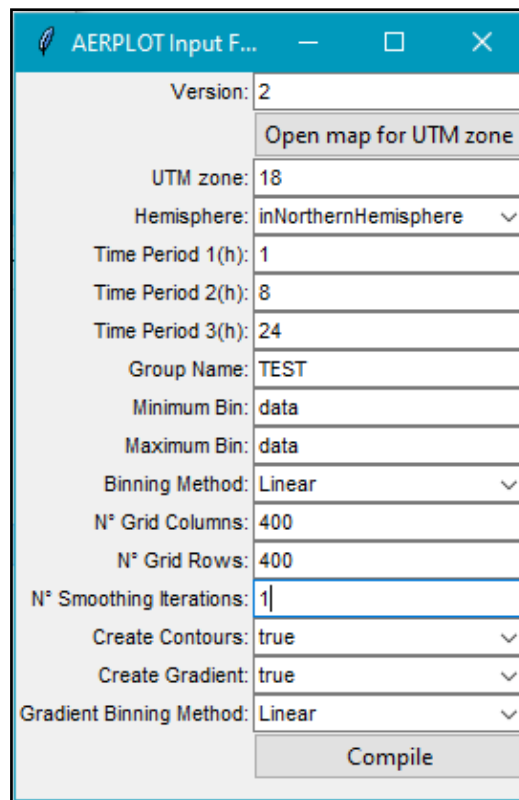


Figure 34. AERPLOT input file compiler

```

version=2
origin=UTM
eastings=0
northing=0
utmZone=18
inNorthernHemisphere=true
originLatitude =0
originLongitude =0
altitudeChoice = relativeToGround
altitude=0
PlotFileName =PLOT1H_TEST.PLT
SourceDisplayInputFileName=aermod.inp
OutputFileNameBase =PLOT1H_TEST.PLT
NameDisplayedInGoogleEarth=PLOT1H_TEST.PLT
sDisableProgressMeter = false
sDisableEarthBrowser = true
IconScale = 0.40
sIconSetChoice=redBlue
minbin=data
maxbin=data
binningChoice =Linear
customBinningElevenLevels=na
contourLegendTitleHTML =C&nbsp;O&nbsp;N&nbsp;E&nbsp;T&nbsp;R&nbsp;A&nbsp;T&nbsp;I&nbsp;O&nbsp;N&nbsp;S
numberofGridCols =400
numberofGridRows =400
numberOfTimesToSmoothContourSurface =1
makeContours =true
contourExtension = 9999999
makeGradients =true
gradientExtension= 9999999
gradientMaxBin=data
gradientMinBin=data
gradientBinningChoice=Linear
customGradBinElevenLevels=na
gradientLegendTitleHTML=Gradient&nbsp;Magnitudes
provideEvenlySpacedInterpolatedGrid = false

```

Figure 35. Compiled “aerplot.inp” file

aerplot1	16.5.2024, 22:25	Mapa s datotekama	
aerplot2	16.5.2024, 22:25	Mapa s datotekama	
aerplot3	16.5.2024, 22:25	Mapa s datotekama	
aermap	16.5.2024, 22:20	INP datoteka	1 KB
aermap	16.5.2024, 22:23	OUT datoteka	4 KB
aermet	6.5.2024, 21:02	PFL datoteka	7,839 KB
aermet	6.5.2024, 21:02	SFC datoteka	1,576 KB
aermod	16.5.2024, 19:23	INP datoteka	2 KB
aermod	16.5.2024, 22:24	OUT datoteka	1,453 KB
MAPDETAIL	16.5.2024, 22:21	OUT datoteka	7 KB
MAPPARAMS	16.5.2024, 22:21	OUT datoteka	7 KB
MAX1H_TEST	16.5.2024, 22:24	OUT datoteka	1,648 KB
MAX8H_TEST	16.5.2024, 22:24	OUT datoteka	1,306 KB
MAX24H_TEST	16.5.2024, 22:24	OUT datoteka	721 KB
newark-e	16.5.2024, 22:20	DEM datoteka	9,629 KB
newark-w	16.5.2024, 22:20	DEM datoteka	9,629 KB
PLOT1H_TEST	16.5.2024, 22:24	PLT datoteka	655 KB
PLOT8H_TEST	16.5.2024, 22:24	PLT datoteka	655 KB
PLOT24H_TEST	16.5.2024, 22:24	PLT datoteka	655 KB
RANK1	16.5.2024, 22:24	RNK datoteka	4 KB
RANK8	16.5.2024, 22:24	RNK datoteka	2 KB
RANK24	16.5.2024, 22:24	RNK datoteka	2 KB
RECEPT	16.5.2024, 22:23	ROU datoteka	161 KB

Figure 36. Contents of the project folder containing the “aerplot” (1,2,3) subfolders after running the “AERPLOT input file compiler” (and previous AERMOD stages)







	aermod	16.5.2024. 22:25	INP datoteka	2 KB
	aermod	16.5.2024. 22:25	OUT datoteka	1,453 KB
	aerplot	16.5.2024. 22:25	Aplikacija	3,928 KB
	aerplot	16.5.2024. 22:25	INP datoteka	2 KB
	PLOT24H_TEST	16.5.2024. 22:25	PLT datoteka	655 KB
	PLOT24H_TEST.PLT	16.5.2024. 22:25	KMZ	62,834 KB

Figure 37. Contents of one of the AERPLOT subfolders, after running AERPLOT, there are three iterations, each for one of the averaging periods

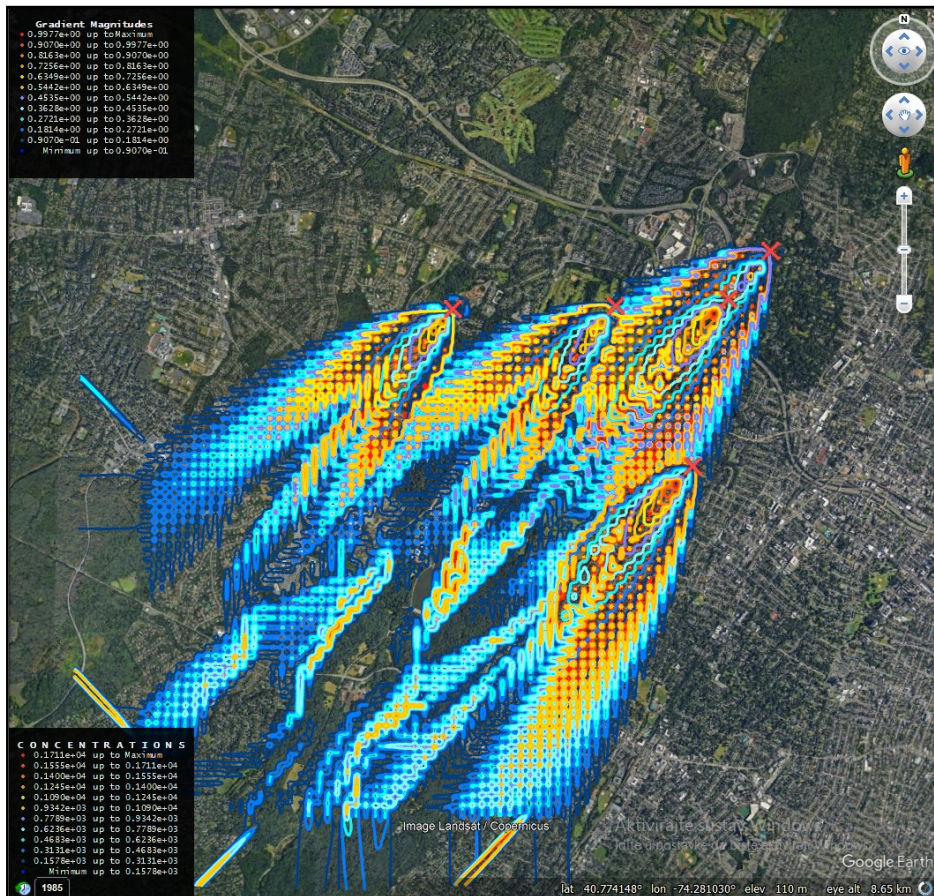
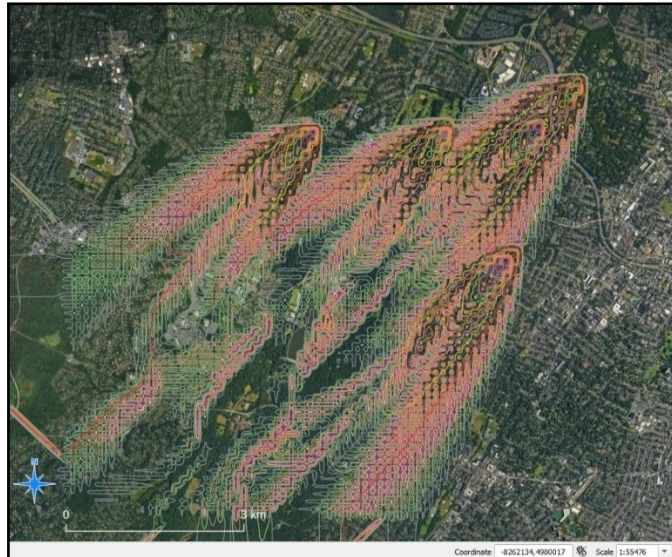
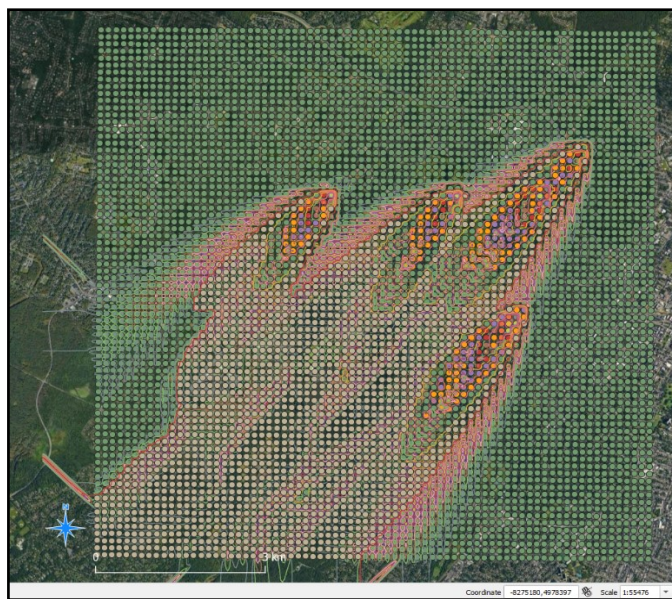


Figure 38. Concentration distribution for the 24h period visualized in Google Earth using 5 point sources over Newark, USA



*Figure 39. Concentration distribution for the 24h period visualized QGIS using contour and gradient lines with 5 point sources over Newark, USA*



*Figure 40. Concentration distribution for the 24h period visualized QGIS using the grid receptor network, contour and gradient lines with 5 point sources over Newark, USA*

## 2.7. PROGRAMMING LOGICS

This section will run through the programming logics of the 3 input file compilers for AERMAP, AERMOD and AERPLOT, and the main frame where the preprocessors AERMAP, AERMET Stage 1 and AERMET stage 2, the processor AERMOD and postprocessor AERPLOT, can be run. Their functions and options will be discussed, and the python code associated with the function included, along with necessary packages. It will go over by line of input file the implications and implementation of the needs presented when creating this software.

```
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
import os
import subprocess
import shutil
import webbrowser
import win32clipboard
import time
import threading
import utm
import simplekml
from shutil import copyfile
```

These import specific libraries and logics into python to be able to perform certain functions, while easing computing by narrowing the interpretation range and not loading unnecessary libraries.

“Tkinter” is a well-known library used for creating visually simple and non-computationally demanding graphical user interfaces.

### 2.7.1 AERMAP INPUT FILE

This section will go over, by line, the different formulations that went into this code. For visual appearance, AERMOD input file code will be written in a different font and blue color and Python code in different font and black color for the main body of the code, while both will be bordered.

```
1)CO STARTING
2)CO TITLEONE TEST
3)CO DATATYPE NED
4)CO DATAFILE UTM33_Italy.tif
5) ANCHORXY 380542.5263 4830024.6688 380542.5263 4830024.6688 33 0
6) FLAGPOLE 1.5
7)CO RUNORNOT RUN
8)CO FINISHED
9)RE STARTING
10) GRIDCART CART01 STA
11) XYINC 380542.5263 50 100 4830024.6688 50 100
12) GRIDCART CART01 END
13)RE FINISHED
14)OU STARTING
15) RECEPTOR RECEPT.ROU
16)OU FINISHED
```

Figure 41. Example AERMAP input file

When the compiler is accessed from the main window (where the stages are ran), it opens in a new window using the “tk.Toplevel” function, a option of the “ttKinter” library.

The ”class Tooltip“ defines a widget that pops up above and to the right of the described item, and disappears when you move your cursor. It contains also generic inputs for style. This feature is available for all the compilers and gives helpful messages like: "Specifies Y coordinate of bottom left grid corner" or "Choose folder where “aermap.inp”, “receptor.rou”, orographic files and other associate files will be created".

```
class Tooltip:
    def __init__(self, widget, text):
        self.widget = widget
        self.text = text
        self.tooltip = None
        self.widget.bind("<Enter>", self.enter)
        self.widget.bind("<Leave>", self.leave)

    def enter(self, event=None):
        x, y, _, _ = self.widget.bbox("insert")
        x += self.widget.winfo_rootx() + 25
        y += self.widget.winfo_rooty() + 25
        if event:
            x = event.x_root + 10
            y = event.y_root + 10
        self.tooltip = tk.Toplevel(self.widget)
        self.tooltip.wm_overrideredirect(True)
        self.tooltip.wm_geometry(f"+{x}+{y}")
        label = tk.Label(self.tooltip, text=self.text,
background="#ffffe0", relief="solid", borderwidth=1)
        label.pack()
```



```
1) CO STARTING
2) CO TITLEONE TEST
```

Here **line 1)** is written by default while in **line 2)** the last phrase is determined upon a textbox entry by the user. A root defines a new window in the GUI and contains different widgets like buttons, textboxes, multiple choice boxes. A title label and entry field are defined. A function “`def generate_output`” is defined that defines the content of the input file text by stating non variable text lines and fetching variable text lines from multiple types of sources like textboxes, multiple choice boxes, explorer windows, filenames and paths, Google maps coordinates that are automatically turned into UTM easting, northing, and zone, using the “`utm.from_latlon`” function from the Python UTM tool pack.

```
root = tk.Tk()
root.title("AERMAP Input File Generator")

# Create and pack the title entry
title_label = ttk.Label(root, text="Title:", font=('Arial',
8))
title_label.grid(row=0, column=0, sticky="e")
title_entry = ttk.Entry(root, font=('Arial', 8))
title_entry.grid(row=0, column=1, sticky="we")

def generate_output():
    output_text_content += "CO STARTING\n"

    if title_entry.get():
        output_text_content += "CO TITLEONE " +
title_entry.get() + "\n"
    ...
return output_text_content
```

The “def compile\_output” function outputs the file upon completion of data input and copies needed files for running AERMAP into the project folder by opening an explorer window and automatically copying the orographic files using the “copyfile” function and outputting the input file in the project folder. After successful compilation the window self-destructs, allowing to run AERMAP in the main window and proceed to the creation of the AERMOD input file.

```
def compile_output():
    output_text_content = generate_output()
    folder_path = filedialog.askdirectory()
    if folder_path:
        file_path = os.path.join(folder_path, "aermap.inp")
        with open(file_path, "w") as file:
            file.write(output_text_content)
        for entry, full_path in datafile_entries:
            _, file_name = os.path.split(full_path)
            destination = os.path.join(folder_path, file_name)
            if not os.path.exists(destination):
                copyfile(full_path, destination)
    root.destroy()
```

```
3)CO DATATYPE NED
```

**Line 3)** is responsible for choosing the correct topographic data format. When NED is chosen from the multiple choice box (“combobox”), “CO DATATYPE NED” is written, this is used also for “GEOTIFF” files. If DEM is chosen the line outputs “CO DATATYPE DEM FILLGAPS”, with the last keyword being added only for DEM, which is related to interpolating receptor heights at missing datapoints in the elevation data.

```
if datafile_entries:
    output_text_content += "CO DATATYPE " +
    datatype_combobox.get()
    if datatype_combobox.get() == "DEM":
        output_text_content += " FILLGAPS\n"
    else:
        output_text_content += "\n"
```

```
4)CO DATAFILE UTM33_Italy.tif
```

**Line 4)** lists the chosen topographic files under the third keyword entry, while the entries can be multiple. The “def browse\_files” function opens an explorer window, where topographic files are chosen, copied to the project folder, and their filenames added to the input file. A “datafile\_entries” list and “datafile\_frame” were created to list the filenames and paths and list them in the GUI respectively. It then opens a frame within the GUI window where the topographic files are listed.

```
for entry, full_path in datafile_entries:
    filename = os.path.basename(full_path)
    output_text_content += "CO DATAFILE " + filename + "\n"

def browse_files():
    filename = filedialog.askopenfilename()
    if filename:
        file_name = os.path.basename(filename)
        new_entry = ttk.Entry(datafile_frame)
        new_entry.grid(row=len(datafile_entries), column=1,
sticky="we")
        new_entry.insert(0, file_name)
        datafile_entries.append((new_entry, filename))

datafile_entries = []
datafile_frame = ttk.Frame(root)
datafile_frame.grid(row=2, column=1, sticky="we")
datafile_button = ttk.Button(datafile_frame, text="Orographic
Files",
                                command=lambda:
datafile_entries.append(ttk.Entry(datafile_frame)).grid(
                                row=len(datafile_entries), column=0,
sticky="we"))
for entry in datafile_entries:
    entry.grid(row=datafile_entries.index(entry), column=1,
sticky="we")
```

```
5) ANCHORXY 380542.5263 4830024.6688 380542.5263 4830024.6688 33 0
```

**Line 5)** Defines the user specified anchor point (most southwest point) with ANCHORXY and this is the default.

```
if (anchor_lat_entry.get() and
    anchor_long_entry.get() and
    utm_zone_entry.get() and
    utm_datum_entry.get()):
    output_text_content += (" ANCHORXY " +
                           anchor_long_entry.get() + " " +
                           anchor_lat_entry.get() + " " +
                           anchor_long_entry.get() + " " +
                           anchor_lat_entry.get() + " " +
                           utm_zone_entry.get() + " " +
                           utm_datum_entry.get() + "\n")
```

The Coordinates are input doubled because this function defines the relationship between the user coordinate system and the UTM coordinate system, which is in this case irrelevant as the coordinates are automatically converted to UTM coordinates. The fore last keyword is the UTM zone input and last the UTM datum conversion. The entry of UTM coordinates and zone is done either manually through textboxes or using Google Maps.

The "def get\_clipboard\_text" function fetches text from the clipboard to the application to be further processed.

```
import utm

def get_clipboard_text():
    try:
        win32clipboard.OpenClipboard()
        clipboard_data =
win32clipboard.GetClipboardData(win32clipboard.CF_TEXT)
        win32clipboard.CloseClipboard()
        return clipboard_data.decode('utf-8')
    except (UnicodeDecodeError, TypeError):
        return ""
```

The `def open_google_maps_for_anchor` creates a function that is later called by button click to open Google Maps and nests the `def monitor_clipboard`. The nested function is forwarded data from `def get_clipboard_text`, evaluates if it is in correct format (decimal latitude and longitude, separated by a comma), converts it to UTM easting, northing and zone, rounds to 4 decimal spots (AERMOD maximum) and inserts the values into textboxes and input file.

```
def open_google_maps_for_anchor(anchor_lat_entry,
                                anchor_long_entry):
    url = "https://www.google.com/maps"
    webbrowser.open(url)

    def monitor_clipboard():
        last_clipboard_text = get_clipboard_text()
        while True:
            clipboard_text = get_clipboard_text()
            if clipboard_text != last_clipboard_text:
                last_clipboard_text = clipboard_text
                try:
                    lat, lon = map(float,
clipboard_text.split(',')
                    utm_coords = utm.from_latlon(lat, lon)
                    utm_easting, utm_northing,
utm_zone_number, utm_zone_letter = utm_coords
                    if anchor_lat_entry.get() == '' and
anchor_long_entry.get() == '':
                        anchor_lat_entry.delete(0, 'end')
                        anchor_long_entry.delete(0, 'end')
                        utm_zone_entry.delete(0, 'end')
                        utm_northing_rounded = round(utm_northing, 4)
                        utm_easting_rounded = round(utm_easting, 4)
                        anchor_lat_entry.insert(0, utm_northing_rounded)
                        anchor_long_entry.insert(0, utm_easting_rounded)
                        utm_zone_entry.insert(0, utm_zone_number)
                    except ValueError:
                        print("Invalid coordinates format in clipboard")
                        time.sleep(1)

        clipboard_thread =
threading.Thread(target=monitor_clipboard)
        clipboard_thread.daemon = True
        clipboard_thread.start()
```

```
6) FLAGPOLE 1.5
```

**Line 6)** gives the receptor height from the ground up to the user's preference input through a textbox.

```
7) CO RUNORNOT RUN
8) CO FINISHED
9) RE STARTING
```

**Lines 7), 8) and 9)** are fixed and part of AERMOD's obligatory inputs.

```
10) GRIDCART CART01 STA
11) XYINC 380542.5263 50 100 4830024.6688 50 100
12) GRIDCART CART01 END
```

**Lines 10), 11), and 12.)** define the receptor grid gridding system type, which is in this case fixed to a cartesian rectangular system, by default named CART01. The numerical inputs are the coordinate of the left bottom corner, user input for number of receptors and spacing, for x and y.

```
output_text_content += " GRIDCART CART01 STA\n"
output_text_content += (" XYINC " +
    anchor_long_entry.get() + " " +
    x_n_entry.get() + " " + x_delta_entry.get() + " " +
    anchor_lat_entry.get() + " " + y_n_entry.get() + " " +
    y_delta_entry.get() + "\n")
output_text_content += " GRIDCART CART01 END\n"
```

```
13) RE FINISHED
14) OU STARTING
15) RECEPTOR RECEPT.ROU
16) OU FINISHED
```

The remaining lines are all fixed. **Line 15)** is the receptor grid network file name.

## 2.7.2 AERMOD INPUT FILE

The AERMOD input file compiler uses the receptor network created by AERMAP (receptor.rou), skips over the AERMET phase, as it has the option to directly input surface air and upper air meteorological data files (sfc. and pfl.). It is equipped with tooltips, like all the compilers. A canvas is created inside the root of the compiler window to house the scrollbar function, as the input data can get rather large for the screen. It is additionally tied to mouse wheel movement events and the inversion of controls instead of up to bottom, to left to right is achieved by the holding of the “Control” key. This opens in a new window using a “`ttk.TOPlevel`” function. The main libraries used are listed below.

```
import os
import subprocess
import shutil
from tkinter import ttk, filedialog
from tkinter import messagebox
import webbrowser
import win32clipboard
import time
import threading
import utm
import simplekml
```

The main window and scrollbar are defined here, to be able to scroll through data while compiling output files for many sources. The default vertical scrollbar is set to become a horizontal scrollbar when the “Control button” is pressed.

```

root = tk.Tk()
root.title("AERMOD Input File Generator")

root.geometry("1000x1000")

frame = ttk.Frame(root)
frame.grid(row=0, column=0, sticky="nsew")

canvas = tk.Canvas(frame)
scrollbar_y = ttk.Scrollbar(frame, orient="vertical",
command=canvas.yview)
scrollbar_x = ttk.Scrollbar(frame, orient="horizontal",
command=canvas.xview)
canvas.configure(yscrollcommand=scrollbar_y.set,
xscrollcommand=scrollbar_x.set)

content_frame = ttk.Frame(canvas)

content_frame.bind("<Configure>", lambda e:
canvas.configure(scrollregion=canvas.bbox("all")))

root.columnconfigure(0, weight=1)
root.rowconfigure(0, weight=1)
frame.columnconfigure(0, weight=1)
frame.rowconfigure(0, weight=1)

canvas.create_window((0, 0), window=content_frame,
anchor="nw")
canvas.grid(row=0, column=0, sticky="nsew")
scrollbar_y.grid(row=0, column=1, sticky="ns")
scrollbar_x.grid(row=1, column=0, sticky="ew")

def _on_mousewheel(event):
    if event.state & 0x4: # Check if Ctrl key is pressed
        canvas.xview_scroll(int(-1 * (event.delta / 120)),
"units")
    else:
        canvas.yview_scroll(int(-1 * (event.delta / 120)),
"units")

canvas.bind_all("<MouseWheel>", _on_mousewheel)

```



```

1)CO STARTING
2)CO TITLEONE TEST
3)CO MODELOPT DFAULT CONC
4)CO AVERTIME 1 8 24
5)CO POLLUTID PM10 PM2.5
6)CO FLAGPOLE 1.5
7)CO RUNORNOT RUN
8)CO FINISHED
9)SO STARTING
10)SO ELEVUNIT METERS
11)SO LOCATION STACK1 POINT 379665.006 4831615.688 5
12)SO LOCATION STACK2 POINT 377816.7838 4829885.5964 3
13)SO SRCPARAM STACK1 30 15 320 15 0.75
14)SO SRCPARAM STACK2 20 10 315 18 0.85
15)SO LOCATION POLYGON1 AREAPOLY 379772.0705 4830945.2329
16)SO LOCATION POLYGON2 AREAPOLY 378811.9662 4831448.6879
17)SO LOCATION MPOLYGON1 AREAPOLY 379772.0705 4830945.5329
18)SO SRCPARAM POLYGON1 55 58 5 42
19)SO SRCPARAM POLYGON2 70 15 6 0
20)SO SRCPARAM MPOLYGON1 15 30 4 15
21)SO AREAVERT POLYGON1 379772.0705 4830945.2329 380313.2487 4830611.4631
22)380310.2487 4830326.3713 380041.5904 4830295.3372 379667.824
23)4830601.7742
24)SO AREAVERT POLYGON2 378811.9662 4831448.6879 379407.5549 4831368.2872
25)379689.582 4830855.3209 379322.9126 4830226.2667 378657.6517 4830325.36
26)378316.8778 4830647.9185
27)SO AREAVERT MPOLYGON2 379772.0705 4830945.5329 380313.037 4830611.6631
28)380041.0904 4830295.423 379667.621 4830601.973
29)SO SRCGROUP MIXED STACK1 STACK2 POLYGON1 POLYGON2 MPOLYGON1
30)SO FINISHED
31)RE STARTING
32)RE INCLUDED RECEPTOR.ROU
33)RE FINISHED
34)ME STARTING
35)ME SURFFILE aemet.sfc
36)ME PROFFILE aemet.pfl
37)ME SURFDATA 134897 2001
38)ME UAIRDATA 0015784 2001

```

*Figure 42. Example AERMOD input file with point sources and polygon sources added both by manually inputting UTM coordinates and by using Google Maps to interactively add vertices and visualize them in Google Earth in real time*

```
1)CO STARTING
2)CO TITLEONE TEST
3)CO MODELOPT DFAULT CONC
```

These are generic AERMOD input lines, with the key word TEST in **line 2)** being the user specified name, input trough a textbox. The “`def generate_output`” function defines the input text.

```
def generate_output():
    # CO section
    output_text_content += "CO STARTING\n"
    if title_entry.get():
        output_text_content += "CO TITLEONE " +
title_entry.get() + "\n"
    output_text_content += "CO MODELOPT DFAULT CONC\n"
```

```
4)CO AVERTIME 1 8 24
```

**Line 4)** Defines the averaging time periods. The user can input 3 periods via textboxes, with the usual AERMOD keywords like 1 (hours), DAY, ANNUAL, which he is guided through using a tooltip.

```
if time1_entry.get() and time2_entry.get() and
time3_entry.get():
    output_text_content += (
        "CO AVERTIME " + time1_entry.get() + " " +
        time2_entry.get() + " " + time3_entry.get() +
"\n"
    )
time1_label = ttk.Label(content_frame, text="Time period 1
(h):", font=('Arial', 8))
time1_label.grid(row=1, column=0, sticky="e")

Tooltip(time1_label, "Defines the first averaging period,
example: 1, 1DAY, ANNUAL")

time1_entry = ttk.Entry(content_frame, width=9,
font=('Arial', 8))
time1_entry.grid(row=1, column=1, sticky="w")
```

```
5)CO POLLUTID PM10 PM2.5
```

**Line 5)** defines the pollutants used in the simulation. The user can input all AERMOD pollutant options. These are input via textbox, multiple pollutants are separated by commas in the same textbox.

```
if pollutant_entry.get():
    output_text_content += "CO POLLUTID " +
pollutant_entry.get() + "\n"
if flagpole_entry.get():
    Tooltip(pollutant_label, "SO2 CO NOX NO2 TSP PM10 PM2.5 LEAD
OTHER")
```

```
6)CO FLAGPOLE 1.5
```

**Line 6)** defines the receptor height, which is suggested to mirror the one set in AERMAP.

```
if flagpole_entry.get():
    output_text_content += "CO FLAGPOLE " +
flagpole_entry.get() + "\n"
flagpole_label = ttk.Label(content_frame,
text="Flagpole/Receptor Height (m):", font=('Arial', 8))
```

```
7)CO RUNORNOT RUN
```

```
8)CO FINISHED
```

```
9)SO STARTING
```

```
10)SO ELEVUNIT METERS
```

**Lines 7) through 10)** are generic AERMOD inputs.

```
output_text_content += "CO RUNORNOT RUN\n"
output_text_content += "CO FINISHED\n\n"

output_text_content += "SO STARTING\n"
output_text_content += "SO ELEVUNIT METERS\n"
```

## *POINT SOURCE*

```
11)SO LOCATION STACK1 POINT 379665.006 4831615.688 5
12)SO LOCATION STACK2 POINT 377816.7838 4829885.5964 3
```

**Lines 11) and 12)** define the location of the point source using the keyword POINT. STACK(i) is a generic name given to all point sources iteratively. There is no limit to the number of point sources you can add. The numbers in order are UTM easting, UTM northing and Zs (optional source elevation in meters above sea level).

A list to store point sources and associated entries is created, which are then listed inside a frame inside the “tkTOPlevel” window. The “def get\_clipboard\_text” function accesses the clipboard alphanumeric content, as for the “AERMAP input file compiler”, which fetched by the “def open\_google\_maps\_for\_pointsource” “def monitor\_clipboard” function. It expects the copied text to be latitude and longitude coordinates from Google maps (which are opened upon request via button in the web browser) in format 43.62077644022055, 13.511095661992483. It parses out latitude and longitude divided by a comma, uses the “utm.from\_latlon” function to convert it to UTM northing and easting uses the “.insert” function to automatically insert it into the GUI textbox and AERMOD input file.

```
pointsource_entries = []

def get_clipboard_text():
    try:
        win32clipboard.OpenClipboard()
        clipboard_data =
win32clipboard.GetClipboardData(win32clipboard.CF_TEXT)
        win32clipboard.CloseClipboard()
        return clipboard_data.decode('utf-8')
    except (UnicodeDecodeError, TypeError):
        return ""
```

```

## Update UTM coordinates using Google Maps and clipboard
def open_google_maps_for_pointsource(lat_entry, lon_entry):
    url = "https://www.google.com/maps"
    webbrowser.open(url)

    def monitor_clipboard():
        last_clipboard_text = get_clipboard_text()
        while True:
            clipboard_text = get_clipboard_text()
            if clipboard_text != last_clipboard_text:
                last_clipboard_text = clipboard_text
                try:
                    lat, lon = map(float,
clipboard_text.split(',')
                    utm_coords = utm.from_latlon(lat, lon)
                    utm_easting, utm_northing,
utm_zone_number, utm_zone_letter = utm_coords
                    if lat_entry.get() == '' and
lon_entry.get() == '':
                        lat, lon = map(float,
clipboard_text.split(',')
                        update_kmz(lat, lon, 'point')
                        os.startfile("updated_file.kmz")
                        lat_entry.delete(0, 'end')
                        lon_entry.delete(0, 'end')
                        utm_northing_rounded =
round(utm_northing, 4)
                        utm_easting_rounded =
round(utm_easting, 4)
                        lat_entry.insert(0,
utm_northing_rounded)
                        lon_entry.insert(0,
utm_easting_rounded)
                except ValueError:
                    print("Invalid coordinates format in
clipboard")
                time.sleep(1)

    clipboard_thread =
threading.Thread(target=monitor_clipboard)
    clipboard_thread.daemon = True
    clipboard_thread.start()

```

Additionally, the “def update\_kmz” function is activated upon copying of coordinates from Google Maps and uses the non-converted/copied latitude and longitude to automatically open Google Earth and visualizes all the point sources within one “.kml” file (also additional polygon sources added via Google Maps). The function is notified by the keywords point or polygon, upon addition of a new source, by which it creates geometry, adds points, new vertices to polygons and creates new polygons, when another source is added. Data about the stack base altitude (Zs) can be easily visualized in Google Earth where the sources are annotated.

```

##Update kmz file in Google Earth
kml = simplekml.Kml()

def update_kmz(lat, lon, geometry_type):
    if geometry_type == 'point':
        kml.newpoint(name="Point Source", coords=[(lon,
lat)])
    elif geometry_type == 'polygon':
        polygon_vertices.append((lon, lat))
        if len(polygon_vertices) >= 4:
            polygon = kml.newpolygon(name="Polygon Area",
outerboundaryis=polygon_vertices)
            polygon.style.linestyle.color = 'ff0000ff'
        kml.save("updated_file.kmz")

current_geometry_type = None

def delete_polygon_vertices():
    polygon_vertices.clear()

```

```

13)SO SRCPARAM STACK1 30 15 320 15 0.75
14)SO SRCPARAM STACK2 20 10 315 18 0.85

```

**Lines 13) and 14)** represent variables other than UTM northing and UTM easting and stack base altitude, regarding points sources, **including, emission rate, stack height, temperature, exit velocity and stack diameter.** The SO LOCATION lines (regarding source coordinates and base heights) are iteratively printed first, then after the SO SRCPARAM lines (regarding source parameters) are iteratively printed i times, equal to the

number of point sources. The entries of the point source locations and its parameters are saved in a list. “if” functions allow the file to compile in case of missing data.

```
if pointsource_entries:
    for i, entry in enumerate(pointsource_entries, start=1):
        lat, lon, ptype, rate, height, temp, vel, diameter =
entry
        if (lat.get() and lon.get()):
            point_location_content += f"SO LOCATION STACK{i}
POINT {lon.get()} {lat.get()}"
            if ptype.get():
                point_location_content += f" {ptype.get()}"
                point_location_content += "\n"

            if rate.get() and height.get() and temp.get() and
vel.get() and diameter.get():
                point_srcparam_content += (
                    f"SO SRCPARAM STACK{i} {rate.get()}
{height.get()} {temp.get()} {vel.get()}"
                    f" {diameter.get()}\n"
def add_pointsource():
    lat_label = ttk.Label(pointsource_frame, text=f"Nothing
{len(pointsource_entries) + 1}:", font=('Arial', 8))
    lat_label.grid(row=1, column=len(pointsource_entries) *
4, sticky="e")

    Tooltip(lat_label, "UTM coordinates, up to 4 decimal
spots")

    Tooltip(choose_on_map_button, "Opens Google Maps; Copied
coordinates are automatically converted to UTM "
"and input into the textboxes,
Google Earth is opened to display point sources")

# Add the new entries to the list
pointsource_entries.append((lat_entry, lon_entry,
ptype_entry,
                            rate_entry, height_entry,
temp_entry, vel_entry, diameter_entry))
```

## *POLYGON SOURCES*

```
15)SO LOCATION POLYGON1 AREAPOLY 379772.0705 4830945.2329
16)SO LOCATION POLYGON2 AREAPOLY 378811.9662 4831448.6879
17)SO LOCATION MPOLYGON1 AREAPOLY 379772.0705 4830945.5329
```

**Line 15)** and **16)** represent polygons added via Google Maps (with the default name POLYGON(j)), while **line 17)** represents polygons with manually added vertices and input coordinates (with the default name MPOLYGON(k)). AREAPOLY is an AERMOD keyword that is automatically defaulted to when a polygon area source is added. There is no limit to the number of polygon sources. It is paired with UTM coordinates of the center of the polygon or, as in this case a starting vertex, which is fetched automatically from the first vertex entry within a polygon using the “`def monitor_clipboard`” function. A list of polygon area sources is created, which are displayed, along with coordinates and parameters in its own frame inside the AERMOD input file compiler window.

```
18)SO SRCPARAM POLYGON1 55 58 5 42
19)SO SRCPARAM POLYGON2 70 15 6 0
20)SO SRCPARAM MPOLYGON1 15 30 4 15
```

**Lines 18)** and **19)** for the polygons added via Google maps and **line 20)** for manually added polygons lists the “SO SRCPARAM” keyword, then the generic source name, and parameters in order: **emission rate, altitude of the emission, number of polygon vertices and initial source height**. The parameters are input via textboxes, while altitudes are easily visualized as the polygon is updated in Google Earth. Upon addition of a polygon area source, a frame is opened where the “Open map” button is located and utilizes the “`def open_google_maps_for_polygon`” command to open Google Maps. In the browser window where Google Maps is opened, the user can right click a location, where he can click and thereby copy the coordinates to the clipboard. Upon copying of coordinates vertices are automatically added to the polygon.



```

polygon_area_source_entries = []

def add_polygon_area_source():
    polygon_entry = {"vertices": [], "rate_entry": "",
"rheight_entry": "", "nvert_entry": "", "iheight_entry": ""}
    polygon_area_source_entries.append(polygon_entry)
    choose_on_map_button = ttk.Button(polygon_area_source_frame,
text="Open map", command=lambda:
open_google_maps_for_polygon(polygon_entry),
                                style='Custom.TButton')
    Tooltip(choose_on_map_button, "Opens Google Maps; Copied
coordinates are automatically converted to UTM,create a new
vertex and insert the values. Google Earth is opened to
display the polygons")
    ...
def add_polygon_area_vertex(polygon_entry, lat, lon):
    global new_vertex_row
    polygon_entry['vertices'].append((lat_entry, lon_entry))
    ...

```

```

21)SO AREAVERT POLYGON1 379772.0705 4830945.2329 380313.2487 4830611.4631
22)380310.2487 4830326.3713 380041.5904 4830295.3372 379667.824
23)4830601.7742

24)SO AREAVERT POLYGON2 378811.9662 4831448.6879 379407.5549 4831368.2872
25)379689.582 4830855.3209 379322.9126 4830226.2667 378657.6517 4830325.36
26)378316.8778 4830647.9185

27)SO AREAVERT MPOLYGON2 379772.0705 4830945.5329 380313.037 4830611.6631
28)380041.0904 4830295.423 379667.621 4830601.973

```

**Lines 21) through 28)** contain the names of polygons and the UTM coordinates of the respective vertices. A repeated code structure, the “def get\_clipboard\_text” function accesses the clipboard content, which is then fetched by the “def open\_google\_maps\_for\_pointsource” and “def monitor\_clipboard” function. It expects the copied text to be latitude and longitude coordinates from Google maps, otherwise it will not react. The parsed out latitude and longitude divided by a comma, are used by the “utm.from\_latlon” function to convert it to UTM northing and easting and rounds to 4 decimal spots. It uses the “.insert” function to automatically insert it into the GUI textbox and AERMOD input file. As coordinates are copied, they are continuously inserted as vertices, until a new source is added, or the file is compiled.

```

def open_google_maps_for_polygon(polygon_entry):
    global current_geometry_type
    current_geometry_type = 'polygon'
    url = "https://www.google.com/maps"
    webbrowser.open(url)

    initial_source_count = len(polygon_area_source_entries)
    def monitor_clipboard(polygon_entry,
initial_source_count):
        last_clipboard_text = get_clipboard_text()
        while True:
            clipboard_text = get_clipboard_text()
            if clipboard_text != last_clipboard_text:
                last_clipboard_text = clipboard_text
                if len(polygon_area_source_entries) >
initial_source_count:
                    return
                try:
                    lat, lon = map(float,
clipboard_text.split(',')
                    update_kmz(lat, lon, 'polygon')
                    os.startfile("updated_file.kmz")
                    utm_coords = utm.from_latlon(lat, lon)
                    utm_easting, utm_northing,
utm_zone_number, utm_zone_letter = utm_coords
                    lat = round(utm_northing, 4)
                    lon = round(utm_easting, 4)
                    add_polygon_area_vertex(polygon_entry,
lat, lon)
                except ValueError:
                    print("Invalid coordinates format in
clipboard")
                    time.sleep(1)

        clipboard_thread =
threading.Thread(target=monitor_clipboard,
args=(polygon_entry, initial_source_count))
        clipboard_thread.daemon = True
        clipboard_thread.start()

kml = simplekml.Kml()
polygon_vertices = []

```

The original copied coordinates from Google Maps are fetched by the „def update\_kmz“ function, which displays the input polygons in Google Earth, in addition to possible point sources can also be visualizes within the same “.kmz” geometry file.

```
def update_kmz(lat, lon, geometry_type):
    if geometry_type == 'point':
        kml.newpoint(name="Point Source", coords=((lon,lat)))
    elif geometry_type == 'polygon':
        polygon_vertices.append((lon, lat))
        if len(polygon_vertices) >= 4:
            polygon = kml.newpolygon(name="Polygon Area",
outerboundaryis=polygon_vertices)
            polygon.style.linestyle.color = 'ff0000ff'
        kml.save("updated_file.kmz")
    current_geometry_type = None

def delete_polygon_vertices():
    polygon_vertices.clear()
```

All polygon source datasets are fetched into correct syntax and order. Appropriate lines are created for source data, vertices and parameters.

```
if polygon_area_source_entries:
    for j, polygon_entry in
enumerate(polygon_area_source_entries, start=1):
        vertices = polygon_entry["vertices"]
        if vertices: first_vertex = vertices[0]
            first_lat_entry, first_lon_entry = first_vertex
            polygon_location_content += ("SO LOCATION POLYGON{j}
f"AREAPOLY {first_lon_entry.get()}{first_lat_entry.get()}\n")
            vertices_location_content += "SO AREAVERT POLYGON{j} "
            for vertex_entry in vertices:
                lat_entry, lon_entry = vertex_entry
                vertices_location_content += {lon_entry.get()}
f"{lat_entry.get()} " vertices_location_content += "\n"
            polygon_srcparam_content += ("SO SRCPARAM POLYGON{j}
{polygon_entry['rate_entry'].get()}
"{polygon_entry['releaseheight_entry'].get()}
{polygon_entry['nvert_entry'].get()} "
f"{polygon_entry['iheight_entry'].get()}\n"
            )
```



## MANUAL POLYGON SOURCES

Manually added polygon sources and point sources aren't displayed in Google Earth, as this is not the main function of the software. They are fetched in the same way as other sources but here from user inputs into textboxes, and manually adding vertices via buttons. "if" functions allow the file to compile in case of missing data.

```
manual_polygon_area_source_entries = []
if manual_polygon_area_source_entries:
    for k, manual_polygon_entry in
enumerate(manual_polygon_area_source_entries, start=1):
    manual_vertices = manual_polygon_entry["manual_vertices"]
    if manual_vertices:
        manual_first_vertex = manual_vertices[0]
        manual_first_lat_entry, manual_first_lon_entry =
manual_first_vertex
        manual_polygon_location_content += (f"SO LOCATION
MPOLYGON{k} AREAPOLY " "{manual_first_lon_entry.get()} "
f"{manual_first_lat_entry.get()}\n")
        manual_vertices_location_content += f"SO AREAVERT
MPOLYGON{j} "
        for manual_vertex_entry in manual_vertices:
            manual_lat_entry, manual_lon_entry =
manual_vertex_entry
            manual_vertices_location_content +=
f"{manual_lon_entry.get()} {manual_lat_entry.get()} "
            manual_vertices_location_content += "\n"
            if manual_polygon_entry['m_rate_entry'].get() and
manual_polygon_entry['m_rheight_entry'].get() and
manual_polygon_entry['m_nvert_entry'].get() and \
manual_polygon_entry['m_iheight_entry'].get():
                manual_polygon_srcparam_content += (
                    f"SO SRCPARAM MPOLYGON{k}
{manual_polygon_entry['m_rate_entry'].get()} "
                    f"{manual_polygon_entry['m_rheight_entry'].get()}"
                    f" {manual_polygon_entry['m_nvert_entry'].get()} "
                    f"{manual_polygon_entry['m_iheight_entry'].get()}\n"
                )
```

All the source outputs are then gathered and rearranged for readability.

```

# Concatenate all content
output_text_content += (
    point_location_content +
    point_srcparam_content +
    polygon_location_content +
    manual_polygon_location_content +
    polygon_srcparam_content +
    manual_polygon_srcparam_content +
    vertices_location_content +
    manual_vertices_location_content
)

```

```
29)SO SRCGROUP MIXED STACK1 STACK2 POLYGON1 POLYGON2 MPOLYGON1
```

**Line 29)** defines the user defined group name, in this case „MIXED“ and lists all source names to include them in the results.

```

if group_name_entry.get():
    point_sources = ['STACK' + str(i) for i in range(1,
len(pointsource_entries) + 1)]
    polygon_sources = ['POLYGON' + str(j) for j in range(1,
len(polygon_area_source_entries) + 1)]
    manual_polygon_sources = ['MPOLYGON' + str(k) for k in
range(1,
len(manual_polygon_area_source_entries) + 1)]
    all_sources = point_sources + polygon_sources +
manual_polygon_sources
    output_text_content += ("SO SRCGROUP
{group_name_entry.get()} {' '.join(all_sources)}\n"
)

```

```

30)SO FINISHED
31)RE STARTING
32)RE INCLUDED RECEPTOR.ROU
33)RE FINISHED
34)ME STARTING

```

**Lines 30), 31), 33) and 34)** are generic AERMOD syntax, while **line 32)** provides the filename of the receptor file created by AERMAP, in this case “RECEPTOR.ROU”. The user is prompted to open an explorer window to choose the receptor file that should be included. The filename and path are fetched, to create a copy of the file in the destination folder of the AERMOD input file. If this file is already present in the folder (as it should be, because all files should be located in the project folder, but this is a safety measure, so the file is present when running the AERMOD processor) it won’t be copied.

```
output_text_content += "SO FINISHED\n\n"

output_text_content += f"RE STARTING\nRE INCLUDED
{chosen_file_entry_map_output.get()}\nRE FINISHED\n\n"
output_text_content += "ME STARTING\n"
def open_file_dialog_map_output():
    file_path_map_output = filedialog.askopenfilename()
    if file_path_map_output:
        file_name = os.path.basename(file_path_map_output)
        chosen_file_entry_map_output.delete(0, tk.END)
        chosen_file_entry_map_output.insert(0, file_name)
```

```
35)ME SURFILE aermet.sfc
36)ME PROFILE aermet.pfl
```

**Lines 35) and 36)** are necessary to include the surface air and upper air meteorological data files, usually obtained from AERMET Stage 1 and 2. The user is prompted, which opens an explorer window to select the “.sfc” and “.pfl” files separately. The two definitions copy the files and input the filenames in the same fashion as the receptor file.

```

output_text_content += "ME SURFFILE " +
chosen_file_entry_sfc_output.get() + "\n"
output_text_content += "ME PROFFILE " +
chosen_file_entry_prof_output.get() + "\n"

def open_file_dialog_sfc_output():
    file_path_sfc_output = filedialog.askopenfilename()
    if file_path_sfc_output:
        file_name = os.path.basename(file_path_sfc_output)
        chosen_file_entry_sfc_output.delete(0, tk.END)
        chosen_file_entry_sfc_output.insert(0, file_name)

def open_file_dialog_prof_output():
    file_path_prof_output = filedialog.askopenfilename()
    if file_path_prof_output:
        file_name = os.path.basename(file_path_prof_output)
        chosen_file_entry_prof_output.delete(0, tk.END)
        chosen_file_entry_prof_output.insert(0, file_name)

```

```

37)ME SURFDATA 134897 2001
38)ME UAIRDATA 0015784 2001

```

**Lines 37) and 38)** regard the station number and starting year of its dataset, which can be found in the “.sfc” file. These are user input via textboxes.

```

if station_num_entry.get() and start_year_entry.get():
    output_text_content += f"ME SURFDATA
{station_num_entry.get()} {start_year_entry.get()}\n"
if upper_air_station_num_entry.get() and
start_year_upper_air_entry.get():
    output_text_content += (f"ME UAIRDATA
{upper_air_station_num_entry.get()} "
f"{start_year_upper_air_entry.get()}\n")

```



```
39)ME PROFBASE 1 METERS
```

**Line 39)** uses the user input profile base elevation to compile this line, while the units are defaulted to meters.

```
if base_elevation_entry.get():
    output_text_content += f"ME PROFBASE
{base_elevation_entry.get()} METERS\n"
```

```
40)ME STARTEND 30 3 2001 2.4 2001
```

**Line 40)** uses the user input from textboxes to output the starting day, month and year, and ending day, month and year, of the analysis.

```
if start_date_entry.get() and end_date_entry.get():
    output_text_content += f"ME STARTEND
{start_date_entry.get()} {end_date_entry.get()}\n"
```

```
41)ME FINISHED
```

```
42)OU STARTING
```

**Lines 41) and 42)** are generic AERMOD syntax.

```
output_text_content += "ME FINISHED\n\n"
output_text_content += "OU STARTING\n"
```

```
43)OU RECTABLE ALLAVE 1ST 2ND 3RD
```

**Line 43)** uses the user input from a textbox (1<sup>ST</sup>, 2<sup>ND</sup>, 3<sup>RD</sup>, 4<sup>TH</sup>, etc.) to output a table containing the number of highest concentrations set by the user, by receptor.

```
if rec_table_entry.get():
    output_text_content += f"OU RECTABLE ALLAVE
{rec_table_entry.get()}\n"
```

```
44)OU MAXTABLE ALLAVE 100
```

**Line 44)** uses the numerical user input from a textbox to output a table containing the use set number of overall maximum values to summarize for each averaging period selected.

```
if max_table_entry.get():
    output_text_content += f"OU MAXTABLE ALLAVE
{max_table_entry.get()}\n"
```

```
46)OU RANKFILE 1 50 RANK1.RNK
47)OU RANKFILE 8 50 RANK8.RNK
48)OU RANKFILE 24 50 RANK24.RNK
```

**Lines 46), 47) and 48)** use the RANKFILE keyword that outputs values by rank for use in Q-Q (quantile) plots, for each averaging period (1, 8 and 24 hours in this case). The averaging period is fetched from the previous inputs from the three averaging time periods. The number of ranked elements is user input (in this case 50 for all periods). The name of the output is determined by the phrase “RANK”, the fetched associated averaging time period and the fetched group name. As the “RANKFILE” and later the “MAXIFILE” outputs aren’t available in AERMOD for annual and total period, averaging periods, is the keywords ANNUAL or PERIOD are input, the line is omitted.

```
if time1_entry.get() = "ANNUAL" or time1_entry.get() ==
"PERIOD":
    pass
else if time1_entry.get() and rank1_hinum_entry.get() and
rank1_hinum_entry.get():
    output_text_content += (
        f"OU RANKFILE {time1_entry.get()} "
        f"{rank1_hinum_entry.get()}
RANK{time1_entry.get()}.RNK\n"
```

```
49)OU MAXIFILE 1 MIXED 350 MAX1H_MIXED.OUT
50)OU MAXIFILE 8 MIXED 85 MAX8H_MIXED.OUT
51)OU MAXIFILE 24 MIXED 15 MAX24H_MIXED.OUT
```

**Lines 49), 50) and 51)** use the MAXIFILE keyword that outputs values of occurrences of violations of the user specified threshold value, for each averaging period (1, 8 and 24 hours in this case). The averaging period is fetched from the previous inputs from the three averaging time periods. The group name must be listed afterwards, which is also automatically fetched. The thresholds for each averaging period are user input via textboxes. The name of the output is determined by the phrase “MAX”, the fetched associated averaging time period and the fetched group name, with the addition of the time unit (H) and the last part is the group name again, with the extension “.out”. The “MAXIFILE” output isn’t available in AERMOD for annual and total period, averaging periods, is the keywords ANNUAL or PERIOD are input, the line is omitted.

```
if time1_entry.get() = "ANNUAL" or time1_entry.get() ==
"PERIOD":
    pass

else if time1_entry.get() and group_name_entry.get() and
max1_value_entry.get():
    output_text_content += (f"OU MAXIFILE {time1_entry.get()}
{group_name_entry.get()} "
                           f"{max1_value_entry.get()}
MAX{time1_entry.get()}H_{group_name_entry.get()}.OUT\n"
                           )
```

```
52)OU PLOTFILE 1 MIXED FIRST PLOT1H_MIXED.PLT
53)OU PLOTFILE 8 MIXED FIRST PLOT8H_MIXED.PLT
54)OU PLOTFILE 24 MIXED FIRST PLOT24H_MIXED.PLT
```

**Lines 52), 53) and 54)** use the PLOTFILE keyword that outputs values of concentration per points in receptor grid, for each averaging period. The averaging period is fetched from the previous inputs from the three averaging time periods. The group name must be listed afterwards, which is also automatically fetched. The keyword FIRST is generic AERMOD syntax, signifying that the first n° of hours will be averaged. The name of the output is determined by the phrase “PLOT”, the fetched associated averaging time period and the fetched group name, with the addition of the time unit (H) and the last part is the group name again, with the extension “.plt”. For annual and period analyses, the “FIRST” keywords, signifying which highest value will be output (1<sup>st</sup>, 2<sup>nd</sup>,...) is not available in AERMOD for annual and total period, averaging periods, is the keywords ANNUAL or PERIOD are input, the line is omitted.

```
if timel_entry.get() and group_name_entry.get():
    output_text_content += (
        f"OU PLOTFILE {timel_entry.get()}
{group_name_entry.get()} "
        f"{' ' if timel_entry.get() == 'ANNUAL' or
timel_entry.get() == 'PERIOD' else 'FIRST
'}PLOT{timel_entry.get()}H_{group_name_entry.get()}.PLT\n"
    )
    output_text_content += "OU FINISHED\n"

return output_text_content
```

**Line 55)** is the last line and generic AERMOD syntax. The “`def compile_output`” function is responsible for prompting the user to choose the output folder where the file “`aermod.inp`” will be automatically generated, along with associated files. It is also responsible for automatic copying of the receptor, surface data and upper air data files into the project folder so AERMOD can be ran seamlessly. After successfully compiling, the AERMOD input file compiler will automatically close, allowing you to run AERMOD from the main window.

```
output_text_content = generate_output()
folder_path = filedialog.askdirectory()
if folder_path:
    file_path = os.path.join(folder_path, "aermod.inp")
    with open(file_path, "w") as file:
        file.write(output_text_content)

    # Copy selected files to the destination folder
    for chosen_file_entry in
[chosen_file_entry_map_output, chosen_file_entry_sfc_output,
chosen_file_entry_prof_output]:
        filename = chosen_file_entry.get()
        if filename:
            source_path = os.path.join(folder_path,
filename)
            destination = os.path.join(folder_path,
filename)
            if source_path != destination:
                shutil.copy(source_path, destination)

    root.destroy()
```

### ***2.7.3 AERPLOT INPUT FILE***

AERPLOT input files, in the compiler, are defined for three different averaging periods, corresponding to the ones from AERMOD. Therefore 3 folders are automatically created upon compiling, which contain files: “aerplot.inp”, “aermod.inp”, “aermod.out”, corresponding averaging period “.plt” file (plot file) (1<sup>st</sup>, 2<sup>nd</sup> or 3<sup>rd</sup>) and aerplot.exe. Three versions of these folders will be created (aerplot1, aerplot2, aerplot3), containing the forementioned files, corresponding to the three averaging periods, will be created. This is done so AERPLOT can be run for all three folders/averaging periods, seamlessly with two mouse clicks. The three averaging periods are obligatory in the application.

```

1)version=2
2)origin=UTM
3)easting=0
4)northing=0
5)utmZone=18
6)inNorthernHemisphere=true
7)originLatitude =0
8)originLongitude =0
9)altitudeChoice = relativeToGround
10)altitude=0
11)PlotFileName =PLOT1H_MIXED.PLT
12)SourceDisplayInputFileName=aermod.inp
13)OutputFileNameBase =PLOT1H_MIXED.PLT
14)NameDisplayedInGoogleEarth=PLOT1H_MIXED.PLT
15)sDisableProgressMeter           = false
16)sDisableEarthBrowser            = true
17)IconScale           = 0.40
18)sIconSetChoice=redBlue
19)minbin=data
20)maxbin=data
21)binningChoice =Linear
22)customBinningElevenLevels=na
23)contourLegendTitleHTML
=C&nbsp;O&nbsp;N&nbsp;C&nbsp;E&nbsp;N&nbsp;T&nbsp;R&nbsp;A&nbsp;T&nbsp;I&nbsp;
s;O&nbsp;N&nbsp;S
24)numberOfGridCols           =400
25)numberOfGridRows           =400
26)numberOfTimesToSmoothContourSurface =1
27)makeContours                =true
28)contourExtension = 9999999
29)makeGradients               =true
30)gradientExtension= 9999999
31)gradientMaxBin=data
32)gradientMinBin=data
33)gradientBinningChoice=Linear
34)customGradBinElevenLevels=na
35)gradientLegendTitleHTML=Gradient&nbsp;Magnitudes

```

*Fig 44. Example of 1st of three created AERPLOT input files created using the AERPLOT input file compiler*

```
1)version=2
2)origin=UTM
3)easting=0
4)northing=0
```

**Line 1)** is a user input (which he is prompted to default to 2) regarding the AERMOD version. **Lines 2),3)** and **4)** are defaulted respectively, to UTM coordinate system, 0 correction for easting and northing.

```
output_text_content1 += "version=" + version_entry.get() +
"\n"
output_text_content1 += "origin=UTM\n"
output_text_content1 += "easting=0\n"
output_text_content1 += "northing=0\n"
```

```
5)utmZone=18
```

**Line 5)**, the UTM zone can be input manually, or by button Google Maps can be automatically opened, where copied coordinates, covert to UTM using the “`def open_google_maps_for_UTM`” function and extract the UTM zone.

```
output_text_content1 += "utmZone=" + utm_entry.get() + "\n"

choose_on_map_button = ttk.Button(root, text="Open map for
UTM zone", command=lambda:
open_google_maps_for_UTM(originlat_entry, originlon_entry))
choose_on_map_button.grid(row=2, column=1, sticky="we")
Tooltip(choose_on_map_button, "Automatically inputs UTM zone
by copying location from Google maps")
```



```

def get_clipboard_text():
    try:
        win32clipboard.OpenClipboard()
        clipboard_data =
win32clipboard.GetClipboardData(win32clipboard.CF_TEXT)
        win32clipboard.CloseClipboard()
        return clipboard_data.decode('utf-8')
    except (UnicodeDecodeError, TypeError):
        return ""

def open_google_maps_for_UTM(originlat_entry,
originlon_entry):
    url = "https://www.google.com/maps"
    webbrowser.open(url)

def monitor_clipboard():
    last_clipboard_text = get_clipboard_text()
    while True:
        clipboard_text = get_clipboard_text()
        if clipboard_text != last_clipboard_text:
            last_clipboard_text = clipboard_text
            try:
                lat, lon = map(float,
clipboard_text.split(',')
                utm_coords = utm.from_latlon(lat, lon)
                utm_easting, utm_northing,
utm_zone_number, utm_zone_letter = utm_coords
                if originlat_entry.get() == '' and
originlon_entry.get() == '':
                    utm_entry.delete(0, 'end')
                    utm_entry.insert(0, utm_zone_number)
            except ValueError:
                print("Invalid coordinates format in
clipboard")
            time.sleep(1)

    clipboard_thread =
threading.Thread(target=monitor_clipboard)
    clipboard_thread.daemon = True
    clipboard_thread.start()

```

```
6) inNorthernHemisphere=true
```

**Line 6)**, the satisfies the obligatory statement of the hemisphere. It's chosen via "Combobox" function, with the option of "inNorthernHemisphere" and "in SouthernHemisphere".

```
output_text_content1 += hemisphere_combobox.get() + "=true\n"  
  
hemisphere_combobox = ttk.Combobox(root,  
values=["inNorthernHemisphere", "inSouthernHemisphere"])
```

```
7) originLatitude =0  
8) originLongitude =0  
9) altitudeChoice = relativeToGround  
10) altitude=0
```

**Line 7)** and **8)** relate longitude and latitude to UTM and are not needed, so they are defaulted to 0. **Line 9)** sets the plotting height relative to the ground by default, and correction of altitude is set to 0 in **line 10)**.

```
output_text_content1 += "originLatitude =0\n"  
output_text_content1 += "originLongitude =0\n"  
output_text_content1 += "altitudeChoice = relativeToGround\n"  
output_text_content1 += "altitude=0\n"
```

```
11) PlotFileName =PLOT1H_MIXED.PLT  
12) SourceDisplayInputFileName=aermod.inp  
13) OutputFileNameBase =PLOT1H_MIXED.PLT  
14) NameDisplayedInGoogleEarth=PLOT1H_MIXED.PLT
```

**Line 11)** through **14)** are related to the input and output filenames. They are defined by the averaging period (this example is for the 1<sup>st</sup> averaging period, this procedure is replicated for the other two time periods) and group name, which are input by the user (and should be identical to those in AERMOD).

Again, if the averaging period keyword is “ANNUAL” or “PERIOD”, the “FIRST” keyword is omitted.

```
output_text_content1 += (  
    f"PlotFileName  
=PLOT{time1_entry.get()}H_{group_name_entry.get()}.PLT\n")  
output_text_content1 +=  
"SourceDisplayInputFileName=aermod.inp\n"  
output_text_content1 += (  
    f"OutputFileNameBase  
=PLOT{time1_entry.get()}H_{group_name_entry.get()}.PLT\n")  
output_text_content1 +=  
(f"NameDisplayedInGoogleEarth=PLOT{time1_entry.get()}H_{group  
_name_entry.get()}.PLT\n")
```

```
15) sDisableProgressMeter      = false  
16) sDisableEarthBrowser      = true  
17) IconScale                  = 0.40  
18) sIconSetChoice=redBlue
```

**Line 15) through 18)** are default values. Progress meters are enabled, automatic opening of Google Earth is disabled, icon size is set to 0.4 and color scale from red to blue.

```
output_text_content1 += "sDisableProgressMeter      =  
false\n"  
output_text_content1 += "sDisableEarthBrowser      =  
true\n"  
output_text_content1 += "IconScale                  = 0.40\n"  
output_text_content1 += "sIconSetChoice=redBlue\n"
```

```

19)minbin=data
20)maxbin=data
21)binningChoice =Linear
22)customBinningElevenLevels=na
23)contourLegendTitleHTML
=C&nbsp;O&nbsp;N&nbsp;C&nbsp;E&nbsp;N&nbsp;T&nbsp;R&nbsp;A&nbsp;T&nbsp;I&nbsp;
sp;O&nbsp;N&nbsp;S

```

**Line 19) and 20)** define the minimum and maximum bin via textbox input, in this case, its set to default to data range using the keyword “data”. **Line 21)** adjusts the bins to the data range. Features a ”combobox” with the options “Linear” and “Log“. Are default values. **Line 22) and 23)** are generic syntax regarding the binning levels and legend, set as default.

```

output_text_content1 += "minbin=" + min_bin_entry.get() +
"\n"
output_text_content1 += "maxbin=" + max_bin_entry.get() +
"\n"
output_text_content1 += "binningChoice =" +
binningchoice_combobox.get() + "\n"
output_text_content1 += "customBinningElevenLevels=na\n"
output_text_content1 += (
"contourLegendTitleHTML
=C&nbsp;O&nbsp;N&nbsp;C&nbsp;E&nbsp;N&nbsp;T&nbsp;R&nbsp;A&nbsp;
sp;"
"T&nbsp;I&nbsp;O&nbsp;N&nbsp;S\n")

```

```

24)numberOfGridCols           =400
25)numberOfGridRows          =400

```

For **lines 24) and 25)** define the user input defines the number of grid rows and columns.

```

output_text_content1 += "numberOfGridCols
=" + gridcols_entry.get() + "\n"
output_text_content1 += "numberOfGridRows
=" + gridrows_entry.get() + "\n"

```

```
27)numberOfTimesToSmoothContourSurface =1
28)makeContours =true
29)contourExtension = 9999999
```

**Line 27)** defines via user input the number of times contour surfaces are smoothed. **Line 28)** defines via „combobox“ if contours should be created or not. **Line 29)** defaults the contour extension value.

```
output_text_content1 += "numberOfTimesToSmoothContourSurface
=" + smooth_entry.get() + "\n"
output_text_content1 += "makeContours
=" + contour_combobox.get() + "\n"
output_text_content1 += "contourExtension = 9999999\n"
```

```
30)makeGradients =true
31)gradientExtension= 9999999
32)gradientMaxBin=data
33)gradientMinBin=data
34)gradientBinningChoice=Linear
35)customGradBinElevenLevels=na
36)gradientLegendTitleHTML=Gradient&nbsp;Magnitudes
37)provideEvenlySpacedInterpolatedGrid = false
```

**Line 30)** defines via „combobox“ if gradient lines should be created or not. **Line 31)** defaults on the gradient extension value. **Line 32)** and **33)** define the minimum and maximum bins for gradients via textbox input, in this case, its set to default to data range using the keyword “data”. **Line 34)** adjusts the bins the data range, featuring a ”combobox” with the options “Linear” and “Log”. **Line 35), 36)** and **37)** are generic syntax regarding the binning levels, legend and grid, set as default.

```
output_text_content1 += "makeGradients
=" + gradient_combobox.get() + "\n"
output_text_content1 += "gradientExtension= 9999999\n"
output_text_content1 += "gradientMaxBin=" +
max_bin_entry.get() + "\n"
output_text_content1 += "gradientMinBin=" +
min_bin_entry.get() + "\n"
output_text_content1 += "gradientBinningChoice=" +
gradientbinningchoice_combobox.get() + "\n"
output_text_content1 += "customGradBinElevenLevels=na\n"
output_text_content1 +=
"gradientLegendTitleHTML=Gradient&nbsp;Magnitudes\n"
output_text_content1 += "provideEvenlySpacedInterpolatedGrid
= false\n"
```

The text is then compiled 3 times, for each averaging period, set in 3 automatically created folders (aerplot1, aerplot2 and aerplot3), together with the copied files “aermod.inp2, ”aermod.out”, plot file and “aerplot.exe”. All in the folder of choice, e.g. project folder. After successful compilation the window self-destructs.

```

def compile_output():
    folder_path = filedialog.askdirectory()

    for i in range(1, 4):
        subfolder_path = os.path.join(folder_path,
f"aerplot{i}")
        os.makedirs(subfolder_path, exist_ok=True)

        shutil.copy(os.path.join(folder_path, "aermod.inp"),
subfolder_path)
        shutil.copy(os.path.join(folder_path, "aermod.out"),
subfolder_path)
        shutil.copy("C:/AERMOD/EXE_all/aerplot.exe",
subfolder_path)
        output_text_content1 = generate_output1()
        output_text_content2 = generate_output2()
        output_text_content3 = generate_output3()

        for i, output_text_content in
enumerate([output_text_content1, output_text_content2,
output_text_content3], start=1):
            with open(os.path.join(folder_path, f"aerplot{i}",
"aerplot.inp"), "w") as file:
                file.write(output_text_content)

```

The plot file names are defined from the same entries as in the AERMOD output (averaging period and group name), requiring the user to use the same input continuously to work. The required accessory inputs are automatically copied, because they were named and fetched in the same iterative way.

```

    plot_files =
[f"PLOT{time_entry.get()}H_{group_name_entry.get()}.PLT" for
time_entry in [time1_entry, time2_entry, time3_entry]]
    for i, plot_file in enumerate(plot_files, start=1):
        src_plot_path = os.path.join(folder_path, plot_file)
        dest_plot_path = os.path.join(folder_path,
f"aerplot{i}", plot_file)
        shutil.copy(src_plot_path, dest_plot_path)

    root.destroy()

```

#### ***2.7.4 “CAIRO for AERMOD”***

“CAIRO for AERMOD” (Compile AERMOD input and run output) is the main window that houses the buttons that upon choosing the project folder run the AERMOD stages (AERMAP, AERMET Stage1, AERMET Stage2, AERMOD and AERPLOT), a textbox for simulation output and buttons to launch input file compilers for AERMAP, AERMOD and AERPLOT. When running AERPLOT the software expects to find the 3 subfolders (aerplot1, aerplot2, aerplot3) to run successfully. During installation the software houses all AERMOD executive files within a predetermined folder so it can run simulations for any folder seamlessly.



Stages for running all AERMOD “.exe” files are defined, which run data from the selected project folder, every stage getting its label.

```
class AERMODGUI(tk.Toplevel):
    def __init__(self, master=None):
        super().__init__(master)
        self.main_window = master
        self.protocol("WM_DELETE_WINDOW",
self.on_close_aermodgui)
        self.title("CAIRO for AERMOD")
        self.geometry("600x450") # window size
        self.stage_labels = ["AERMAP", "AERMET Stage 1",
"AERMET Stage 2", "AERMOD", "AERPLOT"]
        self.stages = [self.run_aermap,
self.run_aermet_stage1, self.run_aermet_stage2,
self.run_aermod, self.run_aerplot]

        for z, stage_label in enumerate(self.stage_labels):
            button = ttk.Button(self.button_frame,
text=stage_label, command=lambda z=z: self.run_stage(z))
```

A text output box is added for preprocessor, processor and post processor output, to visualize the progress, or possible error messages, as well as buttons to launch the compilers.

```
self.output_text = tk.Text(self, height=10, width=60,
wrap=tk.WORD)
self.output_text.pack(pady=10)
app1_button = ttk.Button(self, text="Compile AERMAP
Input File", command=app1)
app2_button = ttk.Button(self, text="Compile AERMOD
Input File", command=app2)
app3_button = ttk.Button(self, text="Compile AERPLOT
Input File", command=app3)
padx=10)
```

The path to the executables is defined and connected to the chosen project folder using the “os.path.join” function. The input file names have no flexibility and are predetermined (aermap.inp, aermet1.inp, aermet2.inp, aermod.inp, aerplot.inp). The AERMAP, AERMET Stage 1 and 2, AERMOD and AERPLOT stages are ran simply by choosing the project folder where its

corresponding inputs are located, by automatically requesting from the “Command shell”, the corresponding folder, AERMOD stage, input file name etc.

```
def run_stage(self, stage_index):
    input_folder = self.choose_input_folder()
    if input_folder:
        if self.stage_labels[stage_index] == "AERMAP":
            executable = os.path.join("C:\\", "CAIRO",
"EXE_all", "aermap.exe")
        elif
self.stage_labels[stage_index].startswith("AERMET"):
            executable = os.path.join("C:\\", "AERMOD",
"EXE_all", "aermet.exe")
        else: executable = os.path.join("C:\\", "AERMOD",
"EXE_all", "aermod.exe")

        if self.stage_labels[stage_index] == "AERMET
Stage 1": inp_file = "aermet1.inp"
        elif self.stage_labels[stage_index] == "AERMET
Stage 2": inp_file = "aermet2.inp"
        else: inp_file =
self.stage_labels[stage_index].lower() + ".inp"

        process = subprocess.Popen([executable,
inp_file], cwd=input_folder, shell=True,
stdout=subprocess.PIPE, stderr=subprocess.STDOUT,
universal_newlines=True)

        self.output_text.insert(tk.END, f"Output for
{self.stage_labels[stage_index]}:\n")
        for line in process.stdout:
            self.output_text.insert(tk.END, line)
            self.output_text.see(tk.END)
            self.update_idletasks()

        process.wait()

def choose_input_folder(self):
    folder_path = filedialog.askdirectory()

    return folder_path
```

The AERPLOT post processor asks for the choice of project folder, where it expects to find the already described aerplot1, aerplot2 and aerplot3 subfolders, after which it runs AERPLOT for all three averaging periods. The aerplot.exe file is the only executive file that is copied into the project folder, as it would have taken a higher level of programming to facilitate to be run like the other executables from a remote predetermined folder.

```
def run_aerplot(self, input_folder, stage_index):
    input_folder = filedialog.askdirectory(title="Select
the folder containing the necessary files")

    for i in range(1, 4):
        aerplot_folder = os.path.join(input_folder,
f"aerplot{i}")
        if os.path.exists(aerplot_folder):
            os.chdir(aerplot_folder)
            subprocess.run(["aerplot"], shell=True)
            print(f"AERPLOT {i} completed successfully.")
        else:
            print(f"Error: Subfolder aerplot{i} not found
in {input_folder}.")

def on_close_aermodgui(self):
    self.destroy()
    if self.main_window:
        self.main_window.destroy()
...

main_window.mainloop()

if __name__ == "__main__":
    main()
```

### 2.7.5 COMPILING EXECUTIVE FILE

For the full functionality of the GUI many dependencies and accessory files are fetched by python. These including metadata were compiled into an “.exe” file that autonomously contains all the functionalities of the GUI. Dependencies were explicitly included using the “hiddenImports” keyword and an icon was included. This was done by running the following Windows Shell command.

```
$hiddenImports = Get-Content requirements.txt | ForEach-Object { "-  
-hidden-import=" + $_.Trim() }  
pyinstaller --onefile --noconsole --name CAIROforAERMOD --icon  
"cairoiconv2.ico" --log-level=DEBUG `  
    --paths  
"C:\Users\domin\PycharmProjects\AERMAPCOMPILER\.venv\Lib\site-  
packages" `  
    "CAIROforAERMOD.py"
```

### 2.7.6 COMPILING INSTALLER

Inno Setup is a free script-driven installation system (installer creator) for Windows programs by Jordan Russell and Martijn Laan, first released in 1997. All data is compiled into a single “EXE” file to install programs. It supports multiple platforms, compression, creation of registry and “INI” file entries, integrated scripting engine based on Pascal Script, multilingual installs, passworded and encrypted installs, etc.

For “CAIRO” to run, AERMAP, AERMOD, AERPLOT and a configuration file (config.json) are required to be in the installation folder. The configuration file is automatically created during installation, and is used by “CAIRO” to fetch the location of the AERMOD executive files. The code contains metadata, locations of files to be compiled into the installer (aermap.exe, aermod.exe, aerplot.exe, CAIROforAERMOD.exe), preferred installation folder, interface that prompts an installation location and other choices and creation of the configuration file depending on installation location. This is done by the following code.

```

[Setup]
AppName=CAIRO for AERMOD
AppVersion=1.0
AppPublisher=MSc Dominik Subotic @UNIVPM
AppPublisherURL=suboticdominik@gmail.com
DefaultDirName={userdocs}\CAIRO for AERMOD
DefaultGroupName=CAIRO for AERMOD
OutputDir=Output
OutputBaseFilename=CAIROforAERMOD_Setup
UninstallDisplayName=Uninstall_CAIROforAERMOD
UninstallDisplayIcon={app}\cairoiconv2.ico
Compression=lzma
SolidCompression=yes
PrivilegesRequired=admin
[Languages]
Name: "english"; MessagesFile: "compiler:Default.isl"
[Files]
Source:"C:\Users\domin\Documents\Masters\Thesis_AERMOD\CAIROforAERMOD_Distribution\CAIROforAERMOD.exe"; DestDir: "{app}"; Flags: ignoreversion
Source:"C:\Users\domin\Documents\Masters\Thesis_AERMOD\CAIROforAERMOD_Distribution\ermap.exe"; DestDir: "{app}"; Flags: ignoreversion
Source:"C:\Users\domin\Documents\Masters\Thesis_AERMOD\CAIROforAERMOD_Distribution\ermod.exe"; DestDir: "{app}"; Flags: ignoreversion
Source:"C:\Users\domin\Documents\Masters\Thesis_AERMOD\CAIROforAERMOD_Distribution\erplot.exe"; DestDir: "{app}"; Flags: ignoreversion
[Icons]
Name: "{group}\CAIRO for AERMOD"; Filename: "{app}\CAIROforAERMOD.exe"
Name: "{commondesktop}\CAIRO for AERMOD"; Filename: "{app}\CAIROforAERMOD.exe"; Tasks: desktopicon
[Tasks]
Name: "desktopicon"; Description: "Create a &desktop icon"; GroupDescription: "Additional icons:"
[Run]
Filename: "{app}\CAIROforAERMOD.exe"; Description: "Launch CAIRO for AERMOD"; Flags: nowait postinstall skipifsilent
[Code]
function ReplaceBackslashesWithForwardSlashes(str: string): string;
var
  i: Integer;
begin
  Result := str;
  for i := 1 to Length(Result) do
  begin
    if Result[i] = '\' then
      Result[i] := '/';
  end;
end;

procedure CurStepChanged(CurStep: TSetupStep);
var
  configFilePath: string;
  jsonContent: AnsiString;
  forwardSlashAppPath: string;
  success: Boolean;
begin
  if CurStep = ssPostInstall then
  begin
    configFilePath := ExpandConstant('{app}\config.json');

```

```
forwardSlashAppPath := ReplaceBackslashesWithForwardSlashes(ExpandConstant('{app}'));

jsonContent := '{' + #13#10 +
    '  "aermap_path": "' + forwardSlashAppPath + '/aermap.exe",' + #13#10 +
    '  "aermod_path": "' + forwardSlashAppPath + '/aermod.exe",' + #13#10 +
    '  "aerplot_path": "' + forwardSlashAppPath + '/aerplot.exe"' + #13#10 +
    '}';

success := SaveStringToFile(configFilePath, jsonContent, False);

if not success then
begin
    MsgBox('Failed to create config.json. Please ensure you have the proper permissions to
write to this directory.', mbError, MB_OK);
end
else
begin
    MsgBox('Installation is successful, at: ' + configFilePath, mbInformation, MB_OK);
end;
end;
end;
```

## **2.8 CASE STUDY ON MULTIPLE INDUSTRIAL SOURCES**

An analysis was done on a real case to test AERMOD model and “CAIRO for AERMOD” application performance in complex environments and multiple sources. SO<sub>x</sub> emissions of 15 point sources were analyzed using georeferenced emission data (figure 55.) from “API” refinery in Falconara Marittima, Italy for the first 24 h. monthly average and whole period of the year 2020. The “API” refinery is owned by “IP Gruppo API S.P.A.” and produces various products rich in hydrocarbons (special bitumen, engine lubricants, LPG, methane, vehicle, marine and jet fuel among others) (IP Gruppo API, 2024.). The importance of its emission and increased risk come from the spatial proximity (~3 km) to Falconara Marittima. A town with around 25 000 inhabitants and higher population densities during the summer due to tourism. The domain (receptor grid network) was defined as a 20\*20 km grid (100 m interstep) centered around the “API” refinery and sources.

Sulphur oxides (SO<sub>x</sub>) are naturally produced by volcanoes, for illustration, the 1991 eruption of Mount Pinatubo in the Philippines released approximately 20 million tons of SO<sub>2</sub> into the atmosphere. And anthropogenically, in various industries like smelting of metal ores, oil refining, and the production of sulfuric acid or coal (Pinatubo Volcano Observatory Team, 1991.). Oxidation of SO<sub>2</sub>, usually in the presence of catalyst such as NO<sub>2</sub>, forms H<sub>2</sub>SO<sub>4</sub>, thus creating a component of acid rain. The highest concentrations of SO<sub>2</sub> are recorded in the vicinity of large industrial facilities. SO<sub>2</sub> is a precursor to fine particulate matter (PM<sub>2.5</sub>), which poses significant health risks, including respiratory and cardiovascular diseases. PM<sub>2.5</sub> can penetrate deep into the lungs and enter the bloodstream (Sharma, et al., 2013.).

Short-term exposure to SO<sub>2</sub> can cause respiratory issues, particularly in vulnerable populations such as children, the elderly, and those with pre-existing respiratory conditions. Symptoms include throat and eye irritation, coughing, and shortness of breath.

Long-term exposure can lead to more severe health problems, including chronic bronchitis and aggravation of existing heart disease (Sharma, et al., 2013.).

The thresholds are defined by the Italian legislation for SO<sub>2</sub>. The thresholds used were 125 µg/m<sup>3</sup> for the 24 h averaging period and 20 µg/m<sup>3</sup> for the monthly and total averaging period.

Biossido di Zolfo (SO <sub>2</sub> )		
Valore di riferimento	Periodo di mediazione	Valore limite
Valore limite orario per la protezione della salute umana	1 ora	350 µg/m <sup>3</sup> da non superare per più di 24 volte per anno civile
Valore limite sulle 24 ore per la protezione della salute umana	1 giorno	125 µg/m <sup>3</sup> da non superare per più di 3 volte per anno civile
Livello critico annuale per la protezione della vegetazione	anno civile	20 µg/m <sup>3</sup>
Livello critico invernale per la protezione della vegetazione	1 ottobre - 31 marzo	20 µg/m <sup>3</sup>

*Figure 45. Hourly (350 µg/m<sup>3</sup>), daily (125 µg/m<sup>3</sup>), yearly and winter (1. October to 31. March, as a means of vegetation protection) (20 µg/m<sup>3</sup>) SO<sub>2</sub> limits given by the Italian legislation (ARPAM, 2010.)*

To run AERMOD source data must be input in correct units and format. Refinery emissions are rather high in temperature so they will experience buoyancy and plume rise. Formation of secondary pollutants from SO<sub>x</sub> is expected (VOCs, H<sub>2</sub>SO<sub>4</sub>), but they aren't modelled in AERMOD.



Source Type	ID	AERMOD Input ID	Base Elevation	Height	Diameter	Exit Velocity	Exit Temp.	Emission Rate	UTM Easting	UTM Northing
Units			[m]	[m]	[m]	[m/s]	[K]	[g/s]	[m]	[m]
POINT	E1_TOPPING	STACK1	5.74	60	2.4	3.71808456	473	12.6	369358.00	4832927.00
POINT	E13_VACUUM3	STACK2	6	59.5	2.44	1.69679505	480	4.76	369216.40	4833119.00
POINT	E2_VISBREAKI	STACK3	3.31	52.6	2.74	0.49645226	470	0.042	369178.00	4833005.00
POINT	E3_THERMAL_C	STACK4	3.41	58	1.79	5.34864777	714	0.41	369251.00	4832994.00
POINT	E5_UNIFINING	STACK5	3.61	60	1.61	1.78624693	501	0.067	369084.00	4833155.00
POINT	E9_VACUUM_1	STACK6	7.51	50	1.33	1.16829638	657	0.97	369379.00	4832891.00
POINT	E7_HDS_1	STACK7	3.6	46.2	1.45	1.99259706	561	0.31	369210.00	4833229.00
POINT	E6_	STACK8	2.39	56.5	1.6	5.97069694	463	0.0001	369115.00	4833059.00
POINT	E17_POST_COM	STACK9	1.69	40	1.21	2.75659771	1003	7.2	369386.00	4833156.00
POINT	E10_HOT_OIL	STACK10	7.14	12.8	1.27	0.23332142	553	0.001	369386.00	4832900.00
POINT	E14_HDS3	STACK11	4.25	54	2	4.09190584	587	0.04	369142.00	4833165.00
POINT	E18__BSG	STACK12	0.43	20	0.92	1.87834112	398	0.00001	369228.00	4833402.00
POINT	E26B__ASG	STACK13	4.29	49.8	2.35	2.97411809	412	0.05	369267.00	4833287.00
POINT	E26A__CCPP	STACK14	4.88	43.8	7.15	9.40720705	404	3.85	369278.00	4833257.00
FLARE	FLARE1	STACK15	0	60	1.01	0.19	1072	9.67	369280.07	4833492.74

Table 3. The data for the point sources in the “API” refinery in Faloconara Marittima, Italy. Includes the source name (ID), base elevation, height, diameter, exit velocity and temperature, emission rate and UTM coordinates given in WSG84 UTM 33N projection system (ESPG:32633)

Source Type	ID	AERMOD Input ID	UTM Easting	UTM Northing	Latitude	Longitude	UTM zone
Units			[m]	[m]	[°]	[°]	
POINT	E1_TOPPING	STACK1	369358.00	4832927.00	43.637865	13.380351	33 N
POINT	E13_VACUUM3	STACK2	369216.40	4833119.00	43.639568	13.378549	ESPG
POINT	E2_VISBREAKI	STACK3	369178.00	4833005.00	43.638535	13.378101	
POINT	E3_THERMAL_C	STACK4	369251.00	4832994.00	43.638449	13.379009	32633
POINT	E5_UNIFINING	STACK5	369084.00	4833155.00	43.639869	13.376900	
POINT	E9_VACUUM_1	STACK6	369379.00	4832891.00	43.637544	13.380620	
POINT	E7_HDS_1	STACK7	369210.00	4833229.00	43.640557	13.378444	
POINT	E6_	STACK8	369115.00	4833059.00	43.639010	13.377307	
POINT	E17_POST_COM	STACK9	369386.00	4833156.00	43.639931	13.380642	
POINT	E10_HOT_OIL	STACK10	369386.00	4832900.00	43.637627	13.380704	
POINT	E14_HDS3	STACK11	369142.00	4833165.00	43.639969	13.377616	
POINT	E18__BSG	STACK12	369228.00	4833402.00	43.642117	13.378625	
POINT	E26B__ASG	STACK13	369267.00	4833287.00	43.641089	13.379136	
POINT	E26A__CCPP	STACK14	369278.00	4833257.00	43.640821	13.379279	
FLARE	FLARE1	STACK15	369280.07	4833492.74	43.642943	13.379248	

Table 4. The locations of sources in displayed in UTM zone, northing and easting, and converted into latitude and longitude. The projection is WSG84 UTM 33N (ESPG:32633)

## 2.8.1 AERMAP IMPLEMENTATION

To compile the AERMAP input file and run it elevation data needs to be parametrized and converted through AERMAP into an AERMOD compatible format, resulting in a receptor grid file. A DEM (Digital Elevation Model) is a digital representation of a terrain. Its grid cells, associated to latitude and longitude, correspond to an altitude value in meters. Depending on the gride cell size, a DEM can be more detailed (high resolution) or less detailed (low resolution).

Elevation data was downloaded from the “Copernicus Browser” (<https://dataspace.copernicus.eu/>), using the “COP DEM GLO 30” dataset (elevation model with 30m/px resolution). An area greater than, and surrounding the grid receptor was selected. The format is “GeoTiff” (16 bit, georeferenced “.tif” file format), the appropriate projection (UTM 33N) and only raw data with added data mask were selected and downloaded (fig. 46).

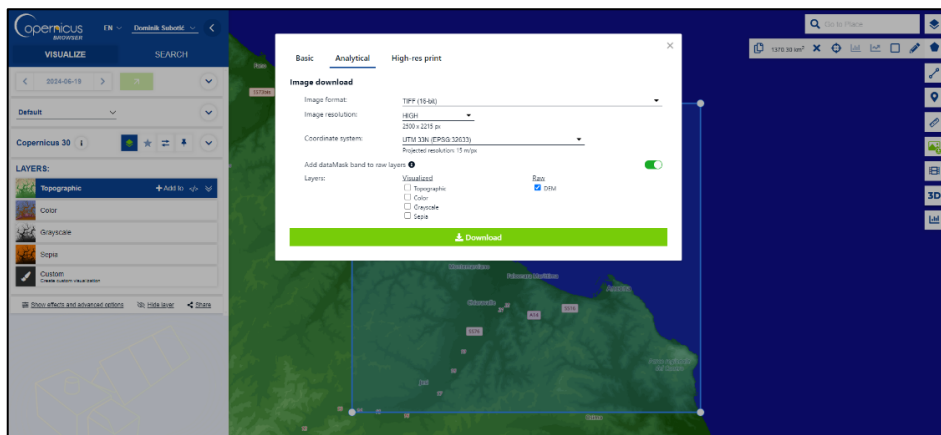
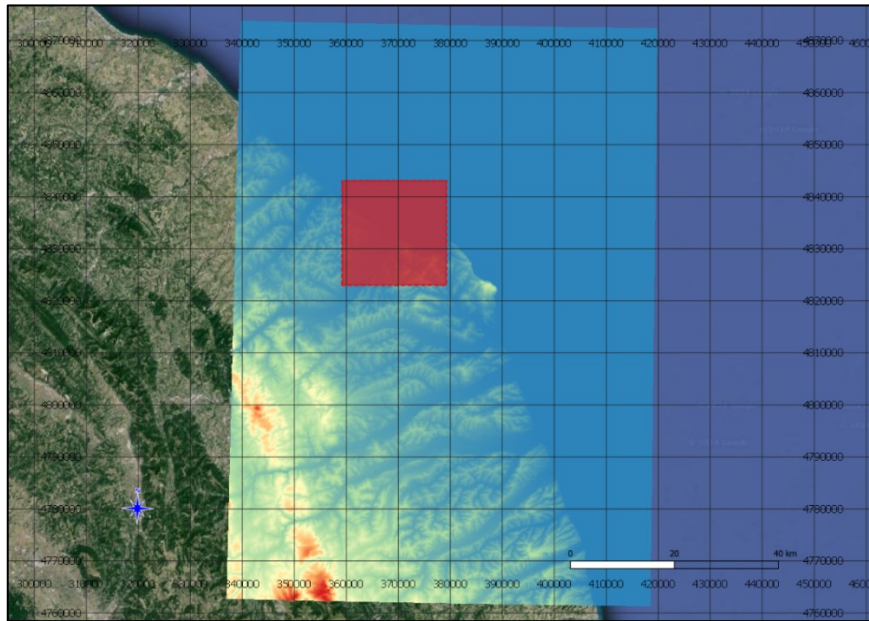


Figure 46. Downloading DEM data from Copernicus browser and Copernicus GLO 30 data, with 30 m/px spatial resolution (15 m/px effective resolution, due to resolution settings) in 16 bit “TIFF” format and WGS84 UTM 33N projection. The appropriate product is selected in the browser, an area selected and downloaded.

The elevation data, along with a shape file of the domain shape file (fig. 47) were loaded and parametrized in QGIS, to visualize the domain in which the receptor grid network will span.



*Figure 47. Domain (Red – 20\*20km, corresponding to receptor grid area) visualized over “GeoTiff” elevation data file with single band pseudo color scheme applied and Google Satellite imagery of Marche, Italy in QGIS. Coordinates are UTM (m).*

The next step is to create the AERMAP input file. The anchor point is defined as the southwest corner of the receptor grid network (fig. 47; 48). The 20\*20 km grid is defined as 200 nodes in x and y, with a step of 100 m (fig. 49).

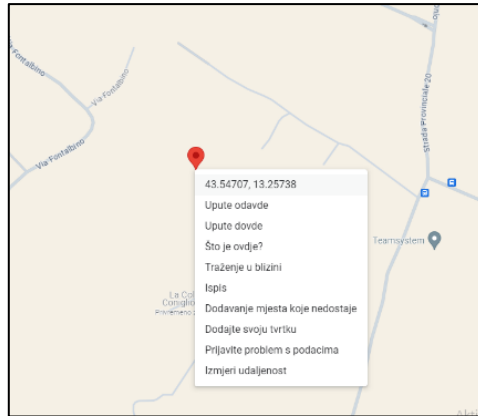


Figure 48. Predetermined anchor point (SW corner) coordinates being copied from Google Maps, while in the GUI they are automatically input and converted to UTM northing, easting and zone

AERMAP In...	
Title:	MARCHE
Data Type:	NED
Orographic Files	falconara.tif
Open map	
Anchor Northing:	4823038.1248
Anchor Easting:	359229.2483
UTM Zone (0-60):	33
UTM Datum:	0
Flagpole Height (m):	1.5
N° of X Gridpoints:	200
Grid ΔX (m):	100
N° of Y Gridpoints:	200
Grid ΔY (m):	100
Compile	

Figure 49. GUI with opened fully filled out “AERMAP input file compiler” window. UTM coordinates and zone were automatically filled out and converted from Google Maps data.

```

CO STARTING
CO TITLEONE MARCHE
CO DATATYPE NED
CO DATAFILE falconara.tif
  ANCHORXY 359229.2483 4823038.1248 359229.2483 4823038.1248 33 0
  FLAGPOLE 1.5
CO RUNORNOT RUN
CO FINISHED
RE STARTING
  GRIDCART CART01 STA
  XYINC 359229.2483 200 100 4823038.1248 200 100
  GRIDCART CART01 END
RE FINISHED
OU STARTING
  RECEPTOR RECEPT.ROU
OU FINISHED

```

Figure 50. Resulting AERMAP input file for API refinery

After compiling the AERMAP input file, the elevation data (under the line “CO DATATYPE falconara.tif”) has already been automatically copied into the project folder and the user can simply select it to run AERMAP (fig. 65).

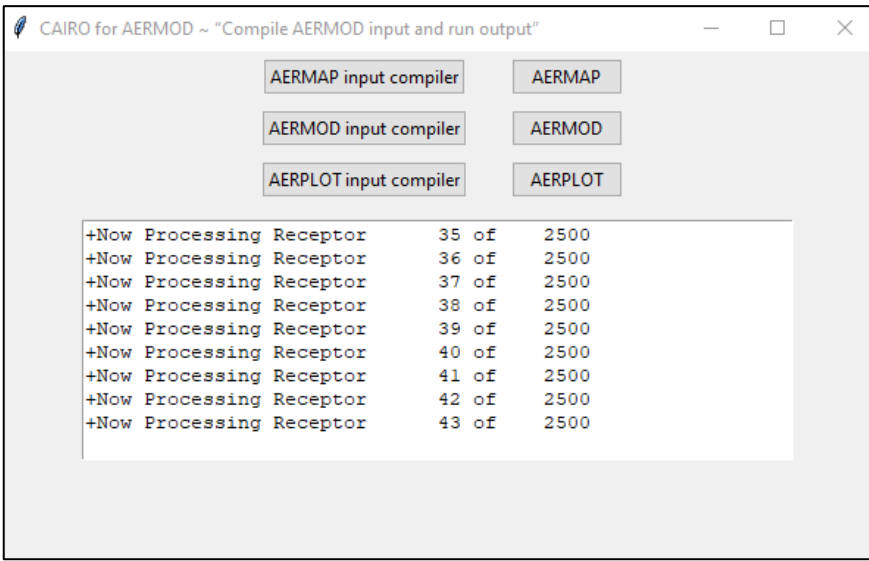


Figure 51. AERMAP running via “CAIRO for AERMOD” after selecting the input project folder for API refinery, Falconara Marittima, Italy

```

** AERMAP - VERSION 18081                                06/20/24
**                                                       00:35:41
** MARCHE
**
** A total of      1 NED files were used
** A total of  40000 receptors were processed
** No user-specified DOMAIN; all available data used
** ANCHORXY  359229.2483 4823038.1248 359229.2483 4823038.1248 33 0
** Terrain heights were extracted by default

RE ELEVUNIT METERS
  GRIDCART CART01 STA
        XYINC 359229.2483 200 100 4823038.1248 200 100
  GRIDCART CART01  ELEV  1   96.2   93.1   94.0   90.8   85.4   80.0
  GRIDCART CART01  ELEV  1   78.1   75.8   75.0   73.1   69.0   61.2
  GRIDCART CART01  ELEV  1   55.1   57.0   55.0   51.0   50.0   47.7
  GRIDCART CART01  ELEV  1   48.9   45.8   44.3   53.8   41.4   51.0
  GRIDCART CART01  ELEV  1   46.0   45.0   42.6   41.2   42.5   43.2
  GRIDCART CART01  ELEV  1   42.0   42.0   42.1   40.8   38.5   37.1
  GRIDCART CART01  ELEV  1   36.5   36.8   39.9   40.7   38.0   37.0
  GRIDCART CART01  ELEV  1   38.7   38.3   40.0   43.0   45.1   57.2
  GRIDCART CART01  ELEV  1   74.1   91.6   97.0   94.0   96.3   96.1
  GRIDCART CART01  ELEV  1   98.5  107.3  117.9  107.0  94.1   87.4

```

Figure 52. Snippet of “receptor.rou”, the resulting receptor grid file

## 2.8.2 AERMOD IMPLEMENTATION

To compile the AERMOD input file the surface and upper air meteorological data of the area was provided. The source locations were predetermined and copied from a list, while the source locations were displayed in real time in Google Earth (fig. 35). They were exported to QGIS to have greater flexibility in visualization (fig. 28, 29, 30).

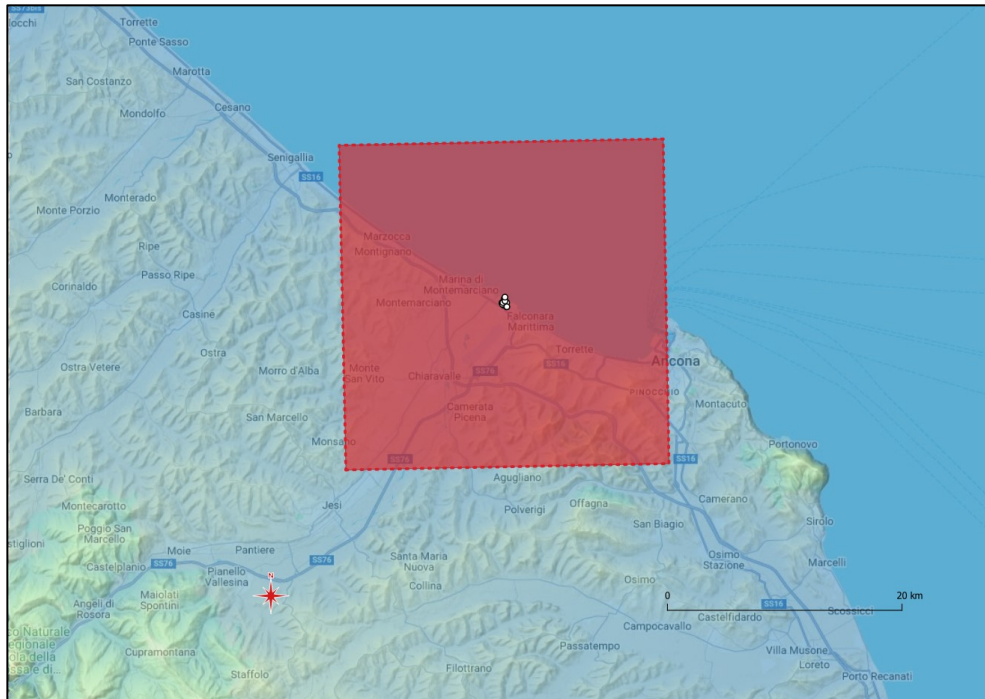


Figure 53. Domain and sources visualized over Google Terrain in QGIS



Figure 54. Point sources visualized in QGIS, overlaying Google Hybrid, highlighting the proximity to urban areas (Falconara Marittima, Italy)

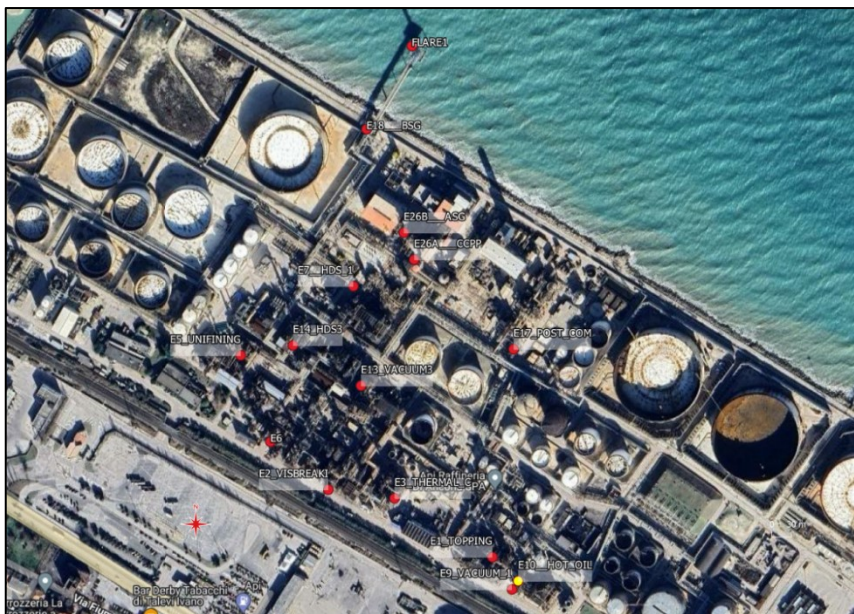


Figure 55. Google Hybrid view in QGIS of point sources, with labels, at API refinery in Falconara Marittima, Italy





*Figure 56. 3D view of sources created in Google Earth including labels and real heights*

Meteorological data provided was MMIF (Mesoscale Model Interface Program) AERMOD ready data in “.sfc” and “.pfl” format (fig. 59), for the year 2020. It was updated in the “AERMOD input file compiler, as described previously and automatically copied into the project folder along with the receptor grid file. Average yearly temperature (fig. 57) and wind rose (fig. 58) were provided underneath, for Falconara Marittima, as they coincide with dispersion mechanisms. High temperatures indicate increased convective boundary layer processes, while the dispersion pattern coincides with the wind rose, especially the longer the averaging period.

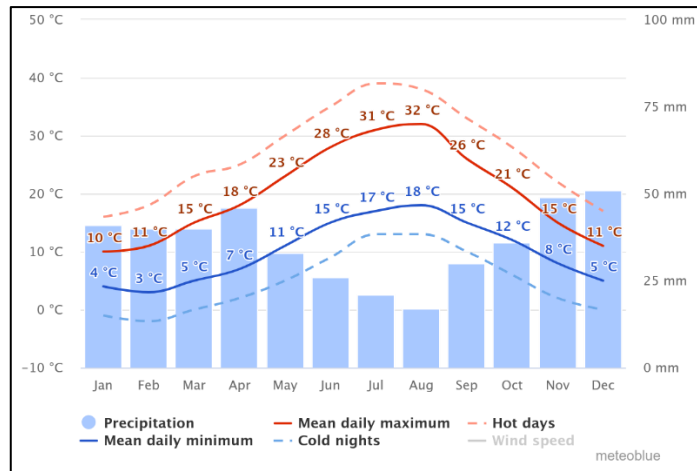


Figure 57. Mean daily minimum, maximum and range temperatures, including precipitation based on data from 1993.-2023. for Falconara Marittima, Italy (Meteoblue, 2024.)

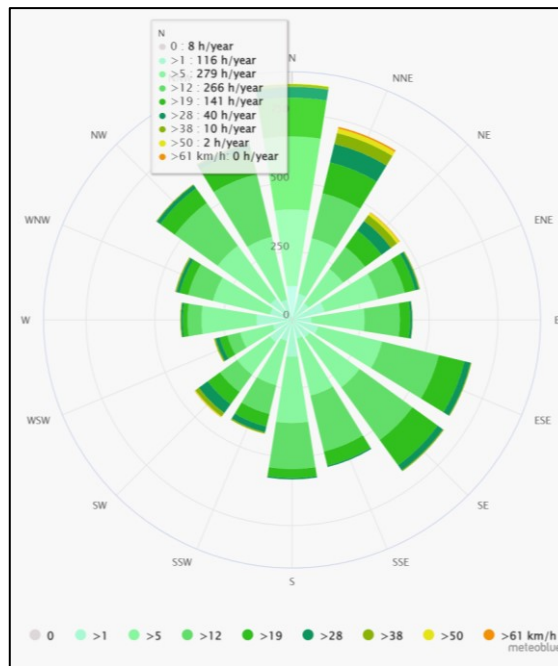


Figure 58. Wind rose with highlighted wind speed fractionation of the predominant wind direction for Falconara Marittima, Italy (Meteoblue, 2024.)

43.618N	13.380E	UA_ID:	99999	SF_ID:	99999	OS_ID:	00190015	VERSION:	15181	MMIF	VERSION	3.4.1	2019-03-11												
20	1	1	1	0.0	0.100	-9.000	-9.000	-999.	1.	8888.0	0.050000	0.00	1.00	3.38	310.1	10.0	282.9	2.0	0	0.00	53.	1025.	4	MIFF-Mod	
20	1	1	1	2	-32.4	0.150	-9.000	-9.000	-999.	49.	9.4	0.050000	4.50	1.00	2.75	304.9	10.0	280.0	2.0	0	0.00	60.	1022.	1	MIFF-Mod
20	1	1	1	3	-31.3	0.154	-9.000	-9.000	-999.	49.	10.4	0.050000	5.06	1.00	2.81	297.2	10.0	278.7	2.0	0	0.00	66.	1023.	1	MIFF-Mod
20	1	1	1	4	-38.7	0.194	-9.000	-9.000	-999.	48.	17.0	0.050000	-1.00	1.00	2.89	274.8	10.0	275.9	2.0	0	0.00	78.	1023.	0	MIFF-Mod
20	1	1	1	5	-38.6	0.200	-9.000	-9.000	-999.	48.	21.3	0.050000	-1.00	1.00	2.89	266.8	10.0	275.3	2.0	0	0.00	78.	1023.	2	MIFF-Mod
20	1	1	1	6	-39.3	0.225	-9.000	-9.000	-999.	419.	26.2	0.050000	-1.00	1.00	3.06	263.5	10.0	275.2	2.0	0	0.00	79.	1024.	7	MIFF-Mod
20	1	1	1	7	-33.4	0.214	-9.000	-9.000	-999.	419.	26.6	0.050000	-1.00	1.00	2.91	259.2	10.0	274.7	2.0	0	0.00	84.	1024.	7	MIFF-Mod
20	1	1	1	8	-27.6	0.170	-9.000	-9.000	-999.	577.	18.5	0.050000	9.06	0.60	2.58	255.8	10.0	274.6	2.0	0	0.00	87.	1024.	4	MIFF-Mod
20	1	1	1	9	-6.5	0.118	-9.000	-9.000	-999.	419.	23.6	0.050000	-1.00	0.43	1.57	251.5	10.0	277.2	2.0	0	0.00	86.	1025.	4	MIFF-Mod
20	1	1	1	10	13.4	0.113	0.491	0.005	292.	292.	-8.9	0.050000	0.73	0.33	0.77	229.4	10.0	280.2	2.0	0	0.00	79.	1025.	3	MIFF-Mod
20	1	1	1	11	32.1	0.203	0.657	0.005	293.	293.	-21.8	0.050000	0.77	0.29	1.99	116.4	10.0	282.9	2.0	0	0.00	71.	1025.	1	MIFF-Mod
20	1	1	1	12	47.8	0.332	1.038	0.006	774.	774.	-63.2	0.050000	0.71	0.28	3.79	107.7	10.0	284.0	2.0	0	0.00	64.	1025.	1	MIFF-Mod
20	1	1	1	13	52.0	0.335	1.167	0.009	1011.	1011.	-59.6	0.050000	0.70	0.29	3.75	100.5	10.0	284.2	2.0	0	0.00	60.	1025.	0	MIFF-Mod
20	1	1	1	14	37.7	0.305	1.056	0.009	1011.	1011.	-60.7	0.050000	0.57	0.32	3.42	96.5	10.0	284.0	2.0	0	0.00	60.	1025.	0	MIFF-Mod
20	1	1	1	15	0.6	0.235	0.705	0.009	1010.	1010.	-93.8	0.050000	0.22	0.39	2.75	94.6	10.0	283.5	2.0	0	0.00	62.	1025.	0	MIFF-Mod
20	1	1	1	16	-6.7	0.093	-9.000	-9.000	-999.	49.	11.7	0.050000	-1.00	0.57	1.59	86.4	10.0	282.2	2.0	0	0.00	69.	1025.	2	MIFF-Mod
20	1	1	1	17	-10.4	0.066	-9.000	-9.000	-999.	49.	2.5	0.050000	10.00	1.00	0.49	75.7	10.0	281.3	2.0	0	0.00	71.	1026.	4	MIFF-Mod
20	1	1	1	18	-6.1	0.041	-9.000	-9.000	-999.	49.	1.0	0.050000	4.59	1.00	0.84	249.8	10.0	280.9	2.0	0	0.00	73.	1026.	6	MIFF-Mod

Figure 59. Opened “.sfc” file containing MMIF surface meteorological data and the station numbers

```

CO STARTING
CO TITLEONE MARCHE
CO MODELOPT DFAULT CONC
CO AVERTIME 24 MONTH PERIOD
CO POLLUTID SOX
CO FLAGPOLE 1.5
CO RUNORNOT RUN
CO FINISHED

SO STARTING
SO ELEVUNIT METERS
SO LOCATION STACK1 POINT 369358.0196 4832927.0295 5.74
SO LOCATION STACK2 POINT 369216.362 4833119.0076 6
SO LOCATION STACK3 POINT 369177.9847 4833004.9856 3.31
SO LOCATION STACK4 POINT 369251.0383 4832994.0034 3.41
SO LOCATION STACK5 POINT 369084.0083 4833155.0372 3.61
SO LOCATION STACK6 POINT 369379.0221 4832890.955 7.51
SO LOCATION STACK7 POINT 369210.0387 4833229.0143 3.6
SO LOCATION STACK8 POINT 369114.9714 4833058.9922 2.39
SO LOCATION STACK9 POINT 369385.9688 4833156.0276 1.69
SO LOCATION STACK10 POINT 369385.9775 4832900.041 7.14
SO LOCATION STACK11 POINT 369141.9773 4833165.0144 4.25
SO LOCATION STACK12 POINT 369228.0225 4833401.9873 0.43
SO LOCATION STACK13 POINT 369267.008 4833287.0096 4.29
SO LOCATION STACK14 POINT 369277.9608 4833257.0194 4.88
SO LOCATION STACK15 POINT 369280.0627 4833492.7439 0
SO SRCPARAM STACK1 12.6 60 473 3.7181 2.4
SO SRCPARAM STACK2 4.76 59.5 480 1.6968 2.44
SO SRCPARAM STACK3 0.042 52.6 470 0.4965 2.74
SO SRCPARAM STACK4 0.41 58 714 5.3487 1.79
SO SRCPARAM STACK5 0.067 60 501 1.7863 1.61
SO SRCPARAM STACK6 0.97 50 657 1.1683 1.33
SO SRCPARAM STACK7 0.31 46.2 651 1.9926 1.45
SO SRCPARAM STACK8 0.0001 56.5 463 5.9707 1.6
SO SRCPARAM STACK9 7.2 40 1003 2.7566 1.21
SO SRCPARAM STACK10 0.001 12.8 553 0.2333 1.27
SO SRCPARAM STACK11 0.04 54 587 4.0919 2
SO SRCPARAM STACK12 0.00001 20 398 1.8783 0.92
SO SRCPARAM STACK13 0.05 49.8 412 2.9741 2.35
SO SRCPARAM STACK14 3.85 43.8 404 9.4072 7.15
SO SRCPARAM STACK15 9.67 60 1072 0.19 1.01
SO SRCGROUP API STACK1 STACK2 STACK3 STACK4 STACK5 STACK6 STACK7 STACK8 STACK9 STACK10 STACK11 STACK12 STACK13 STACK14 STACK15
SO FINISHED

RE STARTING
RE INCLUDED RECEIPT.ROU
RE FINISHED

ME STARTING
ME SURFFILE prova_high_API.sfc
ME PROFFILE prova_high_API.pf1
ME SURFDATA 99999 2020
ME UAIRDATA 99999 2020
ME PROFBASE 0 METERS
ME STARTEND 2020 1 1 2020 12 29
ME FINISHED

OU STARTING
OU RECTABLE ALLAVE 1ST 2ND
OU MAXTABLE ALLAVE 100
OU RANKFILE 24 50 RANK24.RNK
OU RANKFILE MONTH 50 RANKMONTH.RNK
OU MAXFILE 24 API 125 MAX24H_API.OUT
OU MAXFILE MONTH API 20 MAXMONTH_API.OUT
OU PLOTFILE 24 API 1ST PLOT24H_API.PLT
OU PLOTFILE MONTH API 1ST PLOTMONTH_API.PLT
OU PLOTFILE PERIOD API PLOTPERIOD_API.PLT
OU FINISHED

```

Figure 60. AERMOD input file compiled for the means of analyzing “API” refinery point source emissions.

### 2.8.3 AERPLOT IMPLEMENTATION

Firstly, AERPLOT was run for three averaging periods, with bins defaulted to linear interpretation and data range. As a result, each of the averaging periods was plotted using its own range. This was done to define overall the maximum and minimum values from the control runs of the gradient (responsible for grid receptors visualization) and contour lines bins, dependent on the concentration ( $\mu\text{g}/\text{m}^3$ ) at receptor height. The maximum and minimal values respectively for the gradient and contour lines, for all the averaging periods, were chosen as the maximum and minimum bin values for the final analysis (table 5.), to maintain continuity of representation between averaging periods. As the differences between minimal values of the averaging period is a few orders of magnitude in size, logarithmic interpretation of the bins was selected for the final run.

GRADIENT CONCENTRATION RANGES				$\mu\text{g}/\text{m}^3$	
Averaging Period	24 h	Month	Total Period	Total Range	Gradient
Maximum	<b>79.9076</b>	18.1900	8.5970	79.9076	
Minimum	1.4920	0.2610	<b>0.1474</b>	1	

*Table 5. Minimum and maximum values of each period gathered from the control AEROPLOT run, minimal bin was rounded to 1 due to non-relevance regarding legislation for lower values and achieving a finer discretization between concentration bins*

AERPLOT was run again, this time using the overall maximum concentration as the maximum bin and 1 as the minimum bin (table 5), as concentrations lower than 1 aren't relevant in terms of meeting regulations, with thresholds at 20 and 125  $\mu\text{g}/\text{m}^3$  for  $\text{SO}_x$ . It also makes for a smaller concentration range, giving a finer concentration discretization while mapping. Logarithmic scaling of the bins was done to be able to visualize short term (generally higher concentrations) with long term (generally lower concentrations) averaging periods. Provisory results were visualized in Google Earth (fig. 63-68.).





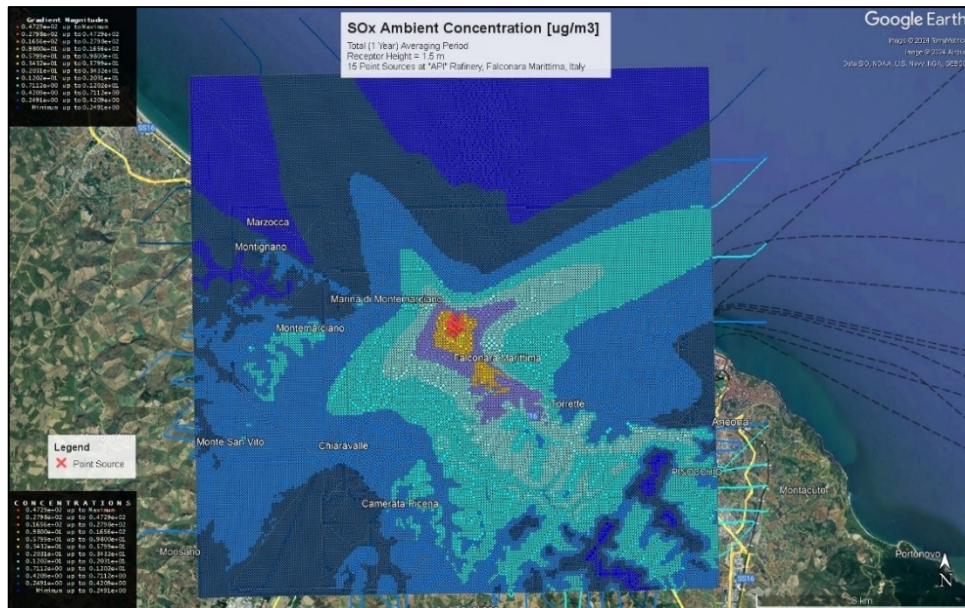


Figure 65. Annual averaging period visualized in Google Earth

In QGIS the symbology of the bins was given by rotating 30° on a HSV color scheme, for each consecutive bin accordingly (Table 6.).

Legend SO <sub>x</sub>	Gradient Bins (Logarithmic)			Contour (Logarithmic)	
Bin Minimum [µg/m <sup>3</sup> ]	Bin Maximum [µg/m <sup>3</sup> ]	Color	Value [µg/m <sup>3</sup> ]	Color	
0.1474	0.2491	Blue	0.1474	Blue	
0.2491	0.4209	Light Blue	0.2491	Light Blue	
0.4209	0.7112	Cyan	0.4209	Cyan	
0.7112	1.2020	Green	0.7112	Green	
1.2020	2.0310	Light Green	1.2020	Light Green	
2.0310	3.4320	Yellow	2.0310	Yellow	
3.4320	5.7990	Orange	3.4320	Orange	
5.7990	9.8000	Red	5.7990	Red	
9.8000	16.5600	Red-Orange	9.8000	Red-Orange	
16.5600	27.9800	Orange	16.5600	Orange	
27.9800	47.2900	Yellow-Orange	27.9800	Yellow-Orange	
47.2900	79.9076	Yellow	47.2900	Yellow	

Table 6. Table containing bin boundary values and contour line values (corresponding to lower bin boundary), with according color palette

## 2.8.4 MATLAB POSTPROCESSING

MATLAB was used to plot the areal extent of concentration levels, corresponding to different averaging periods. The minimum and maximum bins were entered to calculate the mean value of each bin, which were compared to the number of receptor grid points associated with each bin. As the area (20\*20km), resolution (200\*200 node grid with 100 m interstep) and number of receptors associated with each bin are known, the areal extent of each concentration value bin is easily calculated. The MATLAB code is bordered and in its original font for easier discernment.

A figure was created containing box plots of the averaging periods. It presents the range of concentration values modeled in the domain on a logarithmic scale, to better scale between averaging periods. The mean values of bins were assigned as the value of the bin, and the number of datapoints corresponding to each bin was input. The areal percentage and areal coverage of each bin in the domain was calculated.

```
bin_min = [0.14740, 0.24910, 0.42090, 0.71120, 1.20200, 2.03100, 3.43200,
5.79900, 9.80000, 16.56000, 27.98000, 47.29000];
bin_max = [0.24910, 0.42090, 0.71120, 1.20200, 2.03100, 3.43200, 5.79900,
9.80000, 16.56000, 27.98000, 47.29000, 79.90760];

bin_mid = (bin_min + bin_max)/2;

N_DAY = [0, 0, 0, 0, 350, 6506, 11254, 12662, 6889, 1847, 450, 42];
N_MONTH = [0, 6108, 10272, 9755, 8516, 3700, 1154, 395, 96, 4, 0, 0];
N_PERIOD = [7527, 10078, 11697, 7440, 2304, 660, 225, 69, 0, 0, 0, 0];

PERCENT_DAY = N_DAY/400;PERCENT_MONTH = N_MONTH/400;
PERCENT_PERIOD = N_PERIOD/400;
AREA_DAY = PERCENT_DAY*40/100; AREA_MONTH = PERCENT_MONTH*40/100;
AREA_PERIOD = PERCENT_PERIOD*40/100;

DAY = [];
for i = 1:length(N_DAY)
    DAY = [DAY repmat(bin_mid(i), 1, N_DAY(i))];
end
MONTH = [];
for j = 1:length(N_MONTH)
    MONTH = [MONTH repmat(bin_mid(j), 1, N_MONTH(j))];
end
PERIOD = [];
for k = 1:length(N_PERIOD)
    PERIOD = [PERIOD repmat(bin_mid(k), 1, N_PERIOD(k))];
end
```



The boxplot for each period was assigned a dataset, position, color, labels, title, font size, etc., and plotted.

```

%Concatenating all data to plot in one figure
all_data = [DAY, MONTH, PERIOD];
group = [repmat({'24h'}, 1, length(DAY)), repmat({'Month'}, 1,
length(MONTH)), repmat({'Total Period'}, 1, length(PERIOD))];

figure;
hold on;

positions = [1, 2, 3];
h = boxplot(all_data, group, 'Positions', positions, 'Whisker', 15000);

colors = {'r', 'g', 'b'};
h_boxes = findobj(gca, 'Tag', 'Box');
for j = 1:length(h_boxes)
    patch(get(h_boxes(j), 'XData'), get(h_boxes(j), 'YData'),
colors{mod(j-1, 3) + 1}, 'FaceAlpha', 0.5);
end

mean_values = [mean(DAY), mean(MONTH), mean(PERIOD)];

for i = 1:numel(mean_values)
    plot(positions(i), mean_values(i), 'k.', 'MarkerSize', 15);
end

yyaxis left; %Right axis
set(gca, 'YScale', 'log');
ylabel('SOx [\mug/m^3] (log)', 'FontSize', 20);
yyaxis right; %Right axis
set(gca, 'YScale', 'log');

yticks(bin_mid);
yticklabels(arrayfun(@num2str, bin_mid, 'UniformOutput', false));
ylim([0.1, 100]);

ylabel('SOx [\mug/m^3] (log)', 'FontSize', 20);
xlabel('Averaging Period', 'FontSize', 20);
ylabel('SOx [\mug/m^3] (log)', 'FontSize', 20);
title(sprintf('SOx Concentrations Boxplot, Falconara Marittima, Italy,
20x20km Grid, Receptor Height = 1.5m'), 'FontSize', 20);

hold off;

```

Basic statistical data was calculated about the data and its areal coverage, using basic MATLAB functions. Maximum and minimum value, median, average, 25<sup>th</sup> and 75<sup>th</sup> percentile, maximum and minimum mean bin values and pollutant mass at receptor height in a layer over the domain, with 1m thickness. They were presented in a table using the “uitable” function.

```

maximum = [79.9076, 18.1900, 8.5970];
minimum = [1.4920, 0.2610, 0.1474];
median_values = [median(DAY), median(MONTH), median(PERIOD)];
mean_values = [mean(DAY), mean(MONTH), mean(PERIOD)];
q75_values = [prctile(DAY, 75), prctile(MONTH, 75), prctile(PERIOD, 75)];
q25_values = [prctile(DAY, 25), prctile(MONTH, 25), prctile(PERIOD, 25)];
lower_whisker_values = [min(DAY), min(MONTH), min(PERIOD)];
upper_whisker_values = [max(DAY), max(MONTH), max(PERIOD)];

data = {
    'Maximum', maximum(1), maximum(2), maximum(3);
    'Minimum', minimum(1), minimum(2), minimum(3);
    'Median', median_values(1), median_values(2), median_values(3);
    'Mean', mean_values(1), mean_values(2), mean_values(3);
    '75th Percentile', q75_values(1), q75_values(2), q75_values(3);
    '25th Percentile', q25_values(1), q25_values(2), q25_values(3);
    'Maximum Bin', upper_whisker_values(1), upper_whisker_values(2),
upper_whisker_values(3);
    'Minimum Bin', lower_whisker_values(1), lower_whisker_values(2),
lower_whisker_values(3)
    'Pollutant Mass at Receptor Height [kg]', mean_values(1)*40,
mean_values(2)*40, mean_values(3)*40;
};

columnNames = {sprintf('Averaging Periods SOx[ug/m^3]'), '24h', 'Month',
'Total Period'};

t= uitable('Data', data, 'ColumnName', columnNames, 'Position', [290, 130,
400, 201], ...
    'RowName', [], 'ColumnWidth', {120});

title('Averaging Periods - SOx [\mug/m^3]');

```

Finally, the areal distribution of concentrations were plotted with the concentration range in logarithmic scale on the x axis, and area in km<sup>2</sup> on the y axis (linear).

```

figure;
hold on;

subplot(1,3,1)
plot(bin_mid, PERCENT_DAY, '-', 'LineWidth', 2, 'MarkerSize', 8, 'Color', 'b');
yyaxis right;
plot(bin_mid, AREA_DAY, '-', 'LineWidth', 2, 'MarkerSize', 8, 'Color', 'b');
xlabel('SOx Concentration Bins [\mug/m^3]', 'FontSize', 20);
ylabel('Area [km^2]', 'FontSize', 20);
ylim([0,14]);yyaxis left;
ylabel('Percentage of 20x20km Domain [%]', 'FontSize', 20);
title('A) 24h', 'FontSize', 20);
grid on;
set(gca, 'XScale', 'log'); % Set x-axis to logarithmic scale
set(gca, 'YGrid', 'on', 'XGrid', 'off');
xlim([min(bin_mid), max(bin_mid)]);ylim([0,35]);
%%%%%%%%%
subplot(1,3,2)
plot(bin_mid, PERCENT_MONTH, '-', 'LineWidth', 2, 'MarkerSize', 8, 'Color', 'g');
yyaxis right;
plot(bin_mid, AREA_MONTH, '-', 'LineWidth', 2, 'MarkerSize', 8, 'Color', 'g');
xlabel('SOx Concentration Bins [\mug/m^3]', 'FontSize', 20);
ylabel('Area [km^2]', 'FontSize', 20);
ylim([0,14]); yyaxis left;
ylabel('Percentage of 20x20km Domain [%]', 'FontSize', 20);
title('B) Month', 'FontSize', 20);
grid on;
set(gca, 'XScale', 'log'); % Set x-axis to logarithmic scale
set(gca, 'YGrid', 'on', 'XGrid', 'off'); % Only show y-axis grid
xlim([min(bin_mid), max(bin_mid)]);ylim([0,35]);
%%%%%%%%%
subplot(1,3,3)
plot(bin_mid, PERCENT_PERIOD, '-', 'LineWidth', 2, 'MarkerSize', 8, 'Color', 'r');
yyaxis right;
plot(bin_mid, AREA_PERIOD, '-', 'LineWidth', 2, 'MarkerSize', 8, 'Color', 'r');
xlabel('SOx Concentration Bins [\mug/m^3]', 'FontSize', 20);
ylabel('Area [km^2]', 'FontSize', 20);
ylim([0,14]); yyaxis left;
ylabel('Percentage of 20x20km Domain [%]', 'FontSize', 20);
title('C) Total Period (1 Year)', 'FontSize', 20);
grid on;
set(gca, 'XScale', 'log'); % Set x-axis to logarithmic scale
set(gca, 'YGrid', 'on', 'XGrid', 'off'); % Only show y-axis grid
xlim([min(bin_mid), max(bin_mid)]); ylim([0,35]);

hold off;

```

## 2.8.5 MODEL VALIDATION

To examine the validity and compliance with European Union regulations of the modeled data, a hypothesis is constructed and the data is analyzed through a series of methods to measure its difference to observed values, its relation to the set limit values, and assure compliance.

Null Hypothesis ( $H_0$ ): The difference between modeled data and real-world measurements in relation to regulatory limit is significant.

Hypothesis ( $H_1$ ): There is no significant difference between modeled and real-world data in relation to regulatory limit.

From the “ARPA” (Rete Regionale della Qualità dell’Aria) site for the region Marche, Italy, receptors were visualized to compare to modeled data. Three industrial monitoring stations were chosen, next to the “API” refinery. They contain temporal data for PM10, PM2.5, O<sub>3</sub>, SO<sub>2</sub>, NO<sub>2</sub>, C<sub>6</sub>H<sub>6</sub> (benzene), though only SO<sub>2</sub> was utilized to represent SO<sub>x</sub> emissions.

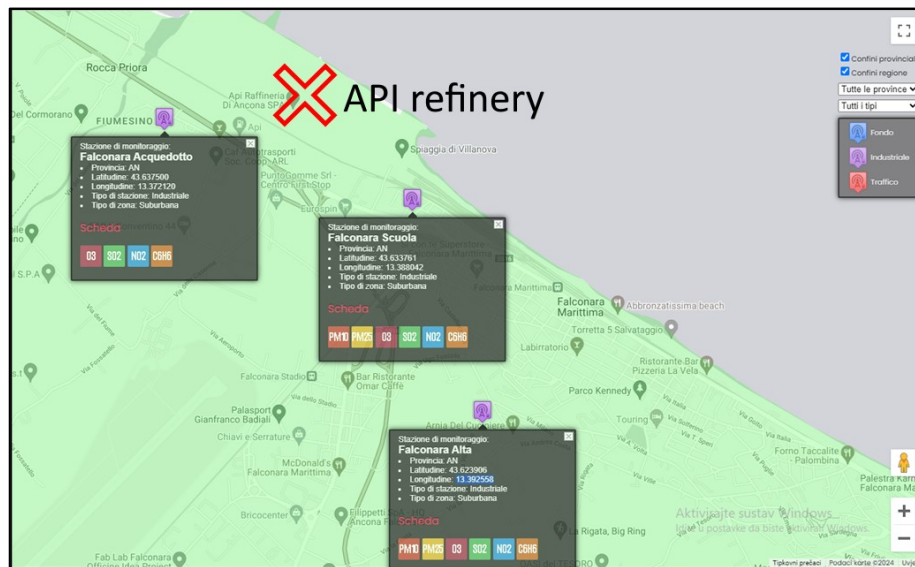
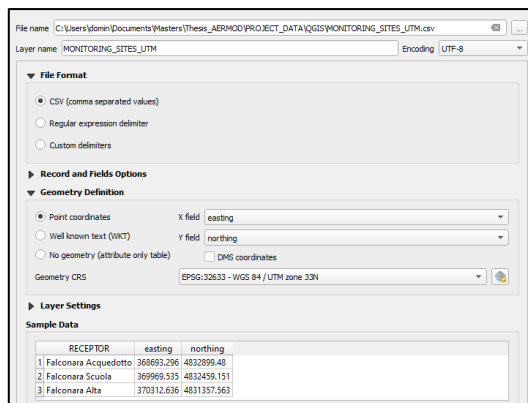


Figure 69. “ARPA” map of receptor sites, with locations of the 3 used monitoring stations in Falconara Marittima, Italy, and location of “API” refinery (e.g. sources)

The location data was put in tabular form, UTM coordinates were calculated and a “.csv” shape file was created to input the receptor data into QGIS, to get model values from the receptor locations. In QGIS a delimited text layer was added containing the coordinates and headers. The elevation of the receptors was cross referenced from multiple sources, including GPS and Google Earth, as they were not available at “ARPA”.

Industrial Receptors	Longitude	Latitude	Easting (m)	Northing (m)	Zone	Altitude (m)
Falconara Acquedotto	43.637500	13.372120	368693.296	4832899.480	N	3
Falconara Scuola	43.633761	13.388042	369969.535	4832459.151	N	6
Falconara Alta	43.623906	13.392558	370312.636	4831357.563	N	100

*Table 7. Locations of monitoring points in latitude and longitude and UTM. “Altitude” refers to elevation at base, receptor height is set to 1.5m*



*Figure 70. Creating receptor points in QGIS from manually created “.csv” file*



*Figure 71. Monitoring station locations (green) and sources (white) visualized in QGIS over analyzed gradient lines (purple)*

As data is given in hourly concentration values, MS Excel was used to determine the maximum and minimum, mean, median, 25th and 75th percentile values from the data corresponding to the three monitoring stations, to compare to AERMOD's simulated values. AERMOD outputs maximum recorded values, so maximal data from averaging periods is considered significant data. The values were compared to real data and limit values.

The monitoring station values were compared to two sets of modeled data, both by using the receptor grid as described so far, and by using discrete receptors at monitoring station locations. The process is the same, differing only in the AERMAP input file (fig. 72), where the "SO DISCCART" keyword was used instead of "GRIDCART". Differences may arise due to limitations in the spatial discretization of the receptor grid, due to representations of gradients and binning values and due to the algorithm, itself and how it handles grid or discrete receptors. Following regulations, the highest 98<sup>th</sup> percentile of the output values is excluded, by finding the 9<sup>th</sup> highest value for the daily averaging period and 2<sup>nd</sup> for the monthly, associated with the top 2% values.

```

CO STARTING
CO TITLEONE MARCHE
CO DATATYPE NED
CO DATAFILE falconara.tif
ANCHORXY 359229.2483 4823038.1248 359229.2483 4823038.1248 33 0
FLAGPOLE 1.5
CO RUNORNOT RUN
CO FINISHED
RE STARTING
DISCCART 368693.296 4832899.480 3 3
DISCCART 369969.535 4832459.151 6 6
DISCCART 370312.636 4831357.563 100 100
RE FINISHED
OU STARTING
RECEPTOR RECEPT.ROU
OU FINISHED

```

Figure 72. AERMAP input file using the “DISCCART” keyword to model real discrete industrial monitoring stations in Falconara Marittima, Italy. The keyword is coupled with the x and y UTM coordinates, base elevation and hill elevation.

To confirm the acceptability of the modeled values the data was tested through the Normalized Mean Square Error (NMSE), Mean Fractional Bias (FB) and Mean Bias (MB), which are the most used methods in the EU, this was done using MS excel.

**Normalized Mean Square Error (NMSE):**

Equation 12.

$$NMSE = \frac{\sum(C_{model} - C_{obs})^2}{\sum(C_{model} * C_{obs})}$$

Where:

NMSE - Normalized Mean Square Error, a value between 0.5 and 2.0 is acceptable, while a value of 0 is ideal, but values below 0.5 are not presumed, 1 is considered a very good value,

C<sub>model</sub> – is the modeled concentration value [µg SO<sub>x</sub>/m<sup>3</sup>],

C<sub>obs</sub> – is the observed concentration value [µg SO<sub>x</sub>/m<sup>3</sup>], (US EPA, 2017.)

### Mean Fractional Bias (FB):

Equation 13.

$$FB = \frac{2}{n} \sum_{i=1}^n \frac{(C_{model} - C_{obs})}{0.5 * (C_{model} + C_{obs})}$$

Where:

FB - Mean Fractional Bias, a value between -0.2 and 0.2 is acceptable, while a value of 0 is ideal,

$C_{model}$  – is the modeled concentration value [ $\mu\text{g SO}_x/\text{m}^3$ ]

$C_{obs}$  – is the observed concentration value [ $\mu\text{g SO}_x/\text{m}^3$ ]

n – is the number of sample (9- 3 averaging periods for 3 receptors),  
(Chang and Hanna, 2004.)

### Mean Bias (MB):

Equation 14.

$$MB = \frac{1}{n} \sum_{i=1}^n (C_{model} - C_{obs})$$

Where:

MB - Mean Bias, a value between -2. and 2.0 is acceptable, while a value of 1 is ideal

$C_{model}$  – is the modeled concentration value [ $\mu\text{g SO}_x/\text{m}^3$ ],

$C_{obs}$  – is the observed concentration value [ $\mu\text{g SO}_x/\text{m}^3$ ],

n – is the number of sample (9- 3 averaging periods for 3 receptors),  
(EC, 2011.)



### **3. RESULTS**

The results section will go through the results of the real case analysis of 15 point sources in Falconara Marittima, Italy using the AERMOD dispersion model and the “CAIRO for AERMOD” application, designed to automatize the process of compiling AERMAP, AERMOD and AERPLOT input files, and running AERMAP, AERMOD and AERPLOT for point sources and polygon sources.

The emission rates of the “API” refinery sources were analyzed using the AERMOD dispersion model using 24 hour, monthly and total period (for the year 2020.), averaging periods. The 1<sup>st</sup> highest entry at every receptor was plotted.

Model validation was done by comparing modeled concentrations of a grid receptor and of discrete receptors (replicating the real monitoring stations) to data from industrial monitoring stations. The highest 98<sup>th</sup> percentile of the modeled values per receptor were discarded to follow regulations and discard overestimations and outliers. The data was interpreted using QGIS, MS excel and MATLAB. The discrepancies were compared to regulatory limits to scale the model performance in terms of its ability to influence compliance. The model performance was also examined through Normalized Mean Square Error (NMSE), Fractional Bias (FB) and Mean Bias (MB), to determine regulatory validity of the results in the EU.

### 3.1 AERMOD ANALYSIS WITH RECEPTOR GRID

The “RANKFILE” keyword outputs concentration values by ranking them and removing duplicate date/hour occurrences. This was done for two averaging periods: 24 hours and monthly. This option isn't available for the total period (1 year in this case). The “MAXFILE” keyword was also entered but as there were no threshold violating values no entries were output. For the 24 hour averaging period the maximum value was 79.9076  $\mu\text{g}/\text{m}^3$  (threshold = 125  $\mu\text{g}/\text{m}^3$ ). For the monthly averaging period, the maximum value was 18.17858  $\mu\text{g}/\text{m}^3$  (threshold = 20  $\mu\text{g}/\text{m}^3$ ).

```

* AERMOD (23132 ): MARCHE                                06/20/24
* AERMET ( 15181):                                       11:35:20
* MODELING OPTIONS USED:  RegDFAULT CONC ELEV FLGPOL RURAL MMIF_Data
* RANK-FILE OF UP TO 50 TOP 24-HR VALUES FOR 1 SOURCE GROUPS
* INCLUDES OVERALL MAXIMUM VALUES WITH DUPLICATE DATA PERIODS REMOVED
* FORMAT: (1X,I6,1X,F13.5,1X,I8.8,2(1X,F13.5),3(1X,F7.2),2X,A8)
* RANK AVERAGE CONC DATE X Y ZELEV ZHILL ZFLAG GRP
*
1 79.90760 20091624 370729.24830 4831438.12480 63.60 112.00 1.50 API
2 69.81735 20080924 369929.24830 4831038.12480 59.90 114.00 1.50 API
3 61.50127 20080524 369529.24830 4832538.12480 5.70 5.70 1.50 API
4 61.15641 20052824 369329.24830 4833538.12480 0.00 0.00 1.50 API
5 60.59419 20080624 369429.24830 4832538.12480 3.00 3.00 1.50 API
6 55.89691 20091124 370329.24830 4831738.12480 60.10 60.10 1.50 API
7 55.06293 20072524 370329.24830 4831738.12480 60.10 60.10 1.50 API
8 54.96076 20070824 370829.24830 4830638.12480 61.00 141.00 1.50 API
9 53.13362 20091524 370029.24830 4831238.12480 67.30 114.00 1.50 API
10 47.59436 20072924 370729.24830 4831438.12480 63.60 112.00 1.50 API
    
```

Figure 73. “RANKFILE” output of AERMOD for the 24h averaging period, listing overall maximum values while omitting duplicate date/hours values

```

* AERMOD (23132 ): MARCHE                                06/20/24
* AERMET ( 15181):                                       11:35:20
* MODELING OPTIONS USED:  RegDFAULT CONC ELEV FLGPOL RURAL MMIF_Data
* RANK-FILE OF UP TO 50 TOP MONTH VALUES FOR 1 SOURCE GROUPS
* INCLUDES OVERALL MAXIMUM VALUES WITH DUPLICATE DATA PERIODS REMOVED
* FORMAT: (1X,I6,1X,F13.5,1X,I8.8,2(1X,F13.5),3(1X,F7.2),2X,A8)
* RANK AVERAGE CONC DATE X Y ZELEV ZHILL ZFLAG GRP
*
1 18.17858 20073124 369229.24830 4833238.12480 4.90 4.90 1.50 API
2 15.28295 20093024 369429.24830 4832938.12480 1.90 6.00 1.50 API
3 14.89281 20083124 369429.24830 4832938.12480 1.90 6.00 1.50 API
    
```

Figure 74. “RANKFILE” output of AERMOD for the monthly averaging period, listing overall maximum values while omitting duplicate date/hours values

### 3.2 AERPLOT POSTPROCESSING WITH RECEPTOR GRID

AERPLOT was run for all three averaging periods: 24 hours (fig. 75), monthly (fig. 76), total period (fig. 77). The data was parametrized and displayed in QGIS.

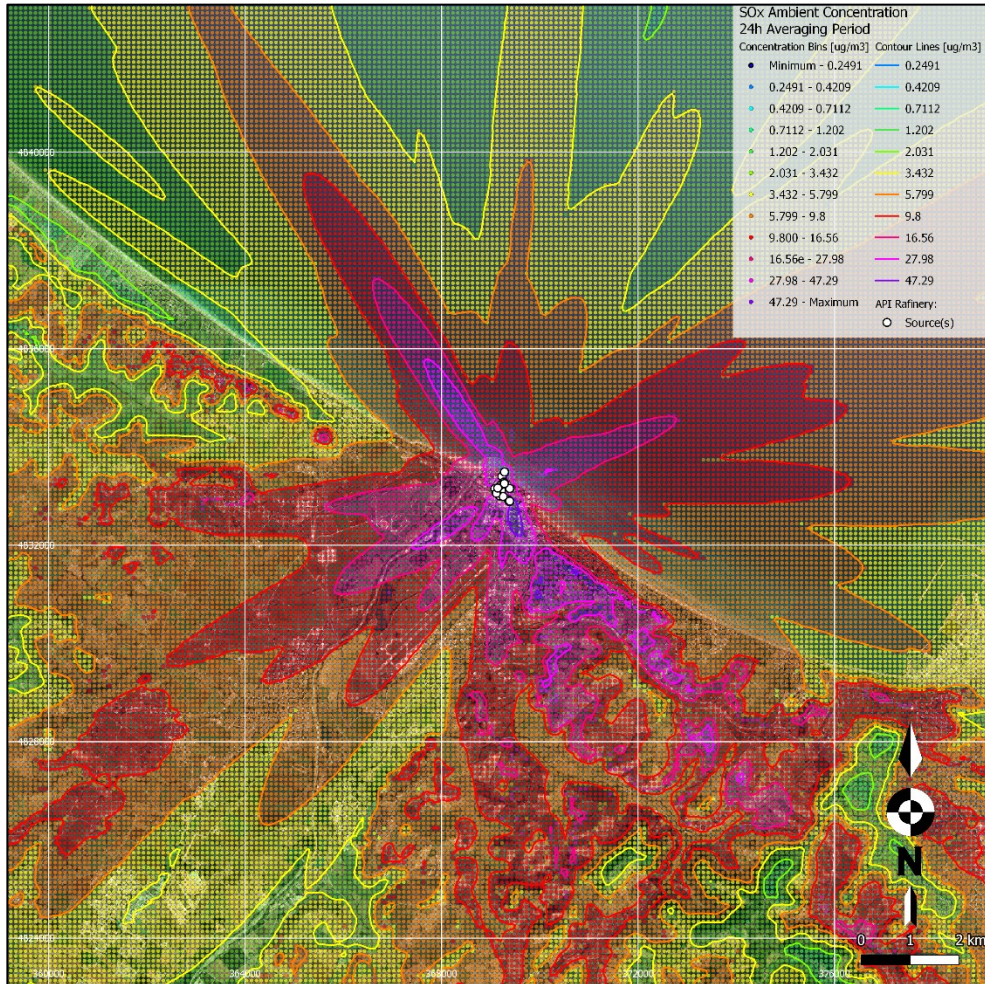
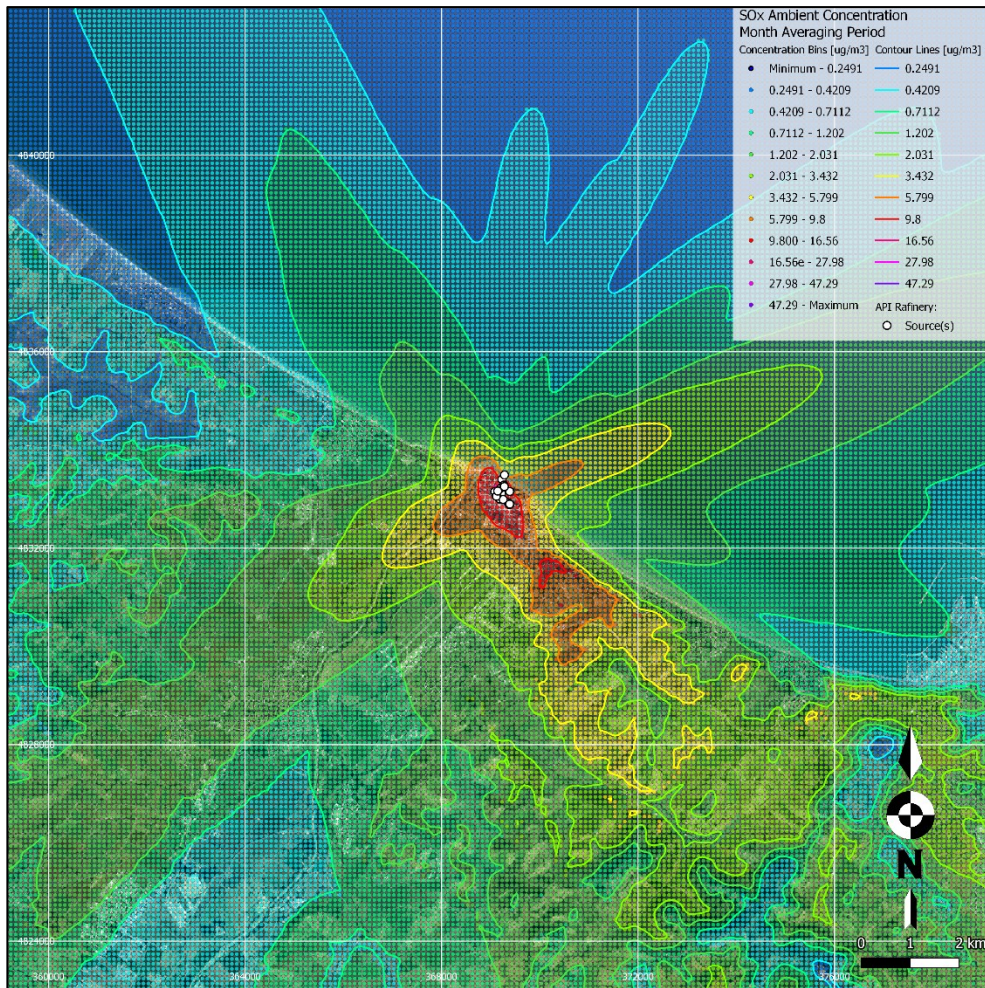


Figure 75. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the 24 h averaging period, displayed in QGIS. The domain is a 20\*20km receptor grid (200\*200 node grid with 100 m interstep), receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme.



*Figure 76. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the monthly averaging period, displayed in QGIS. The domain is a 20\*20km receptor grid (200\*200 node grid with 100 m interstep), receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme.*

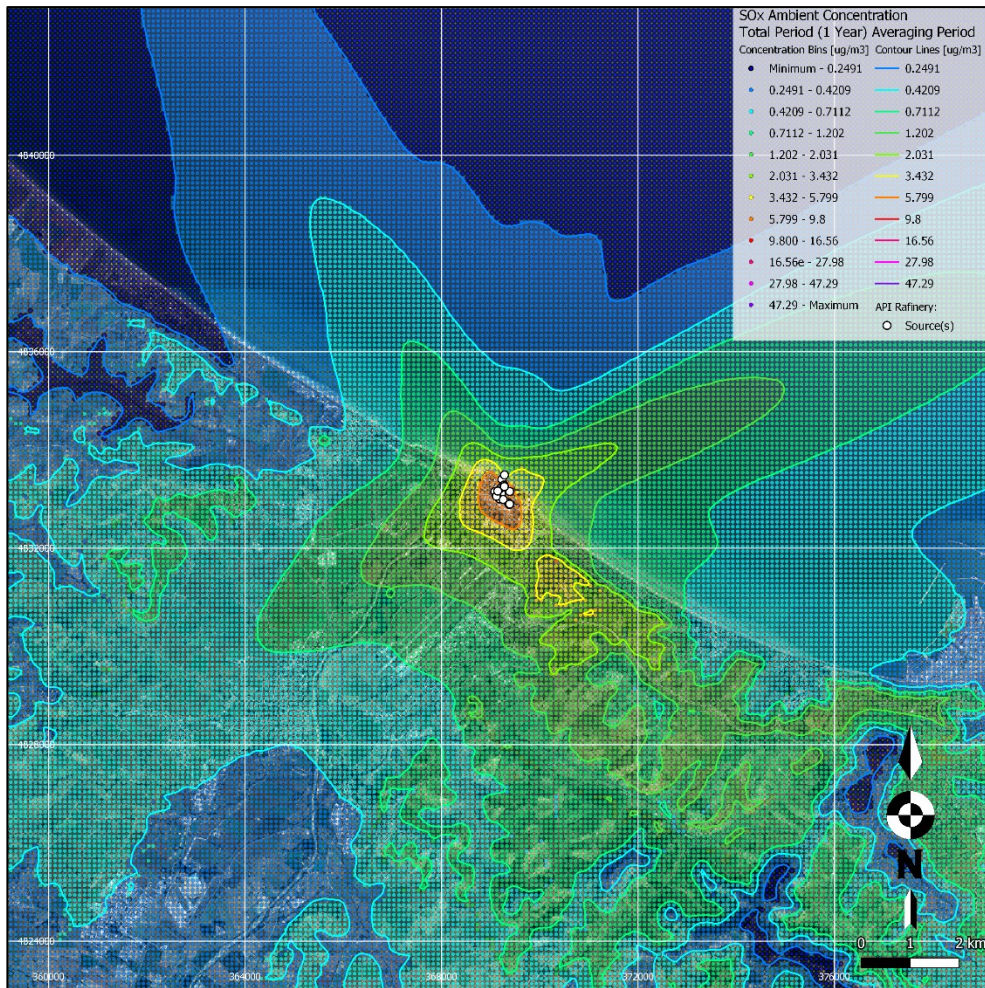
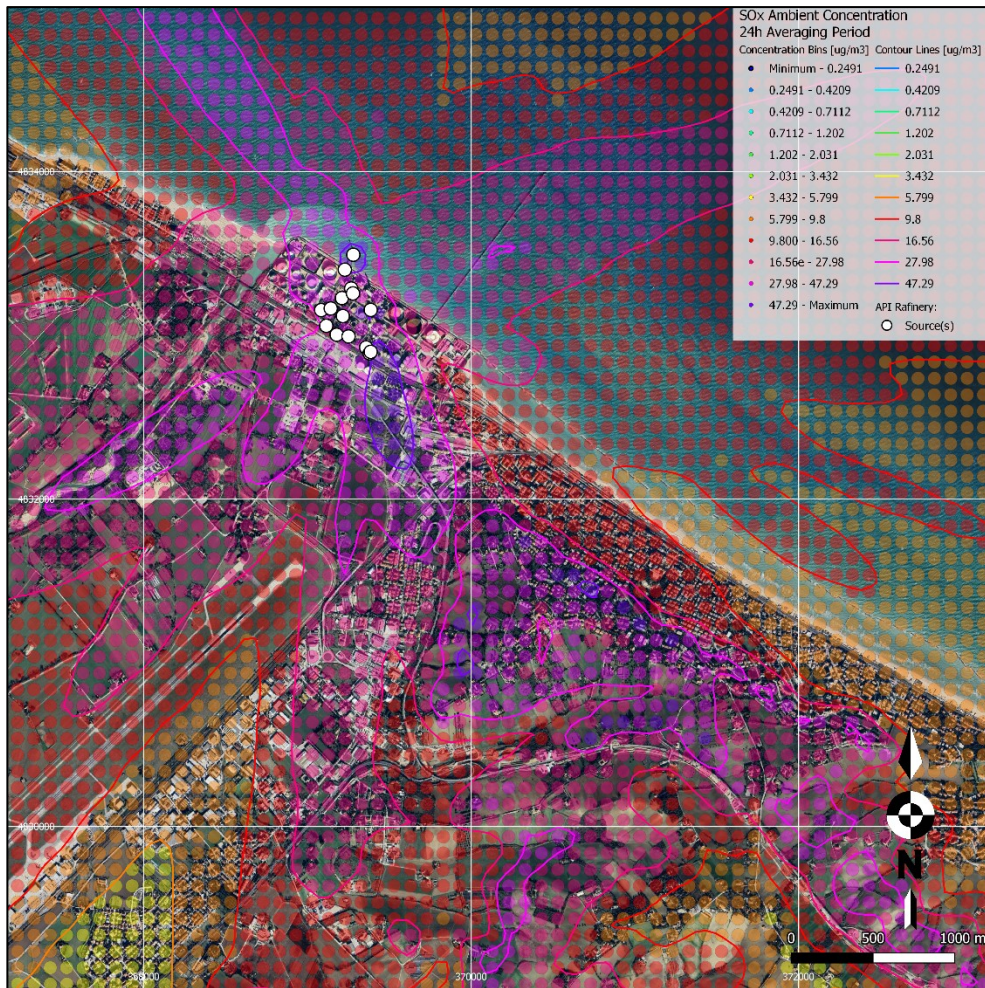
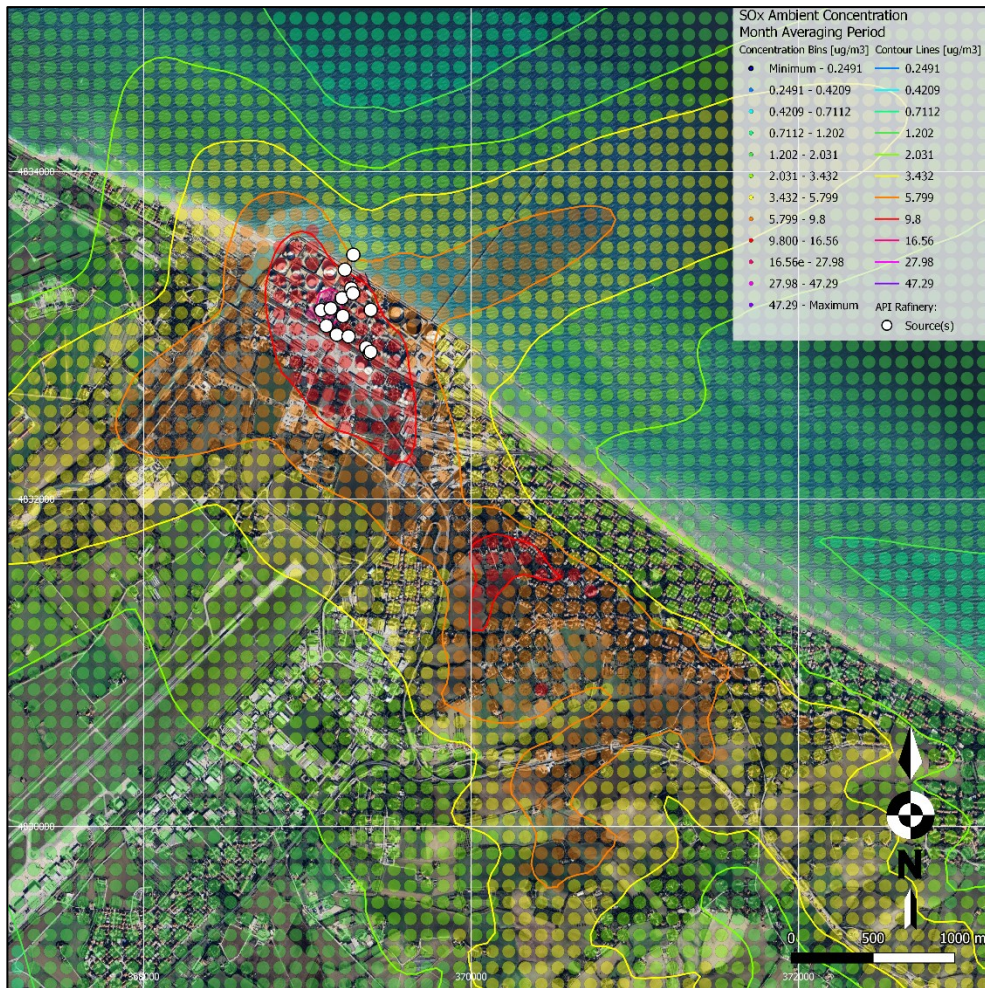


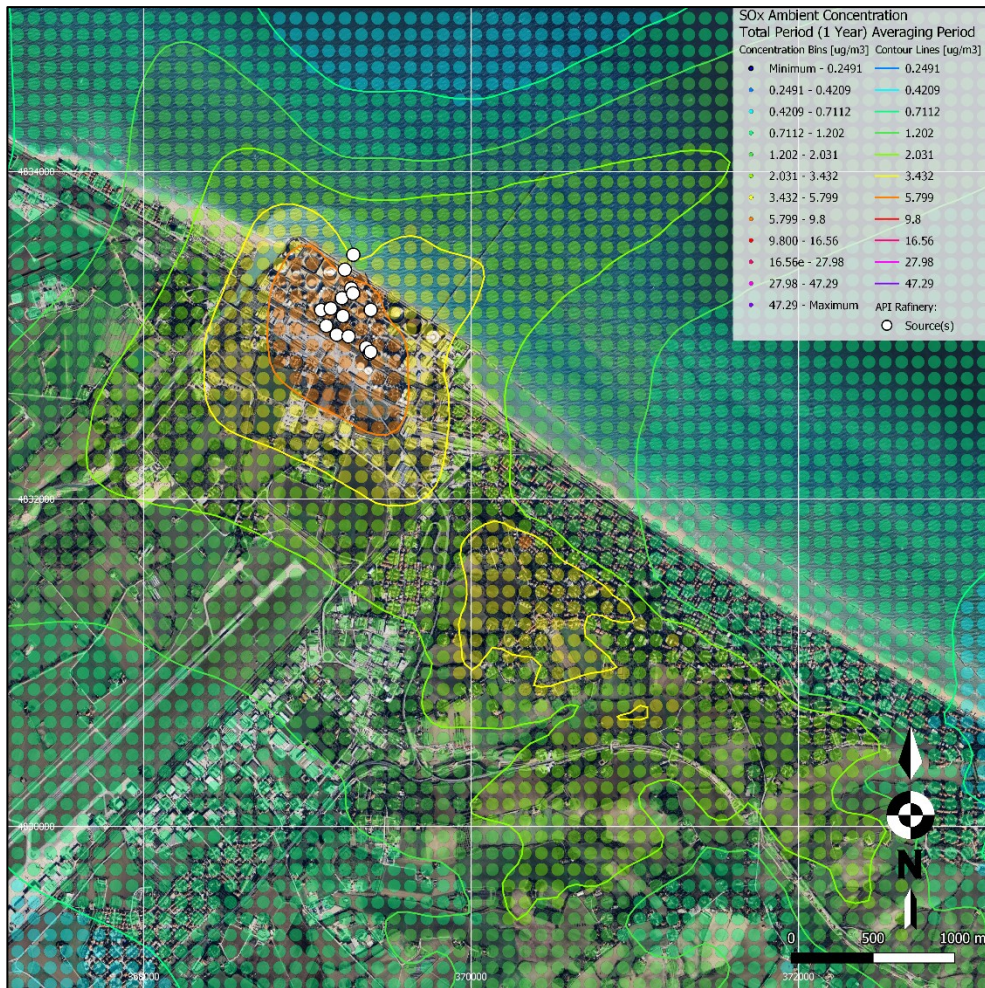
Figure 77. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the total (the year 2020.) averaging period, displayed in QGIS. The domain is a 20\*20km receptor grid (200\*200 node grid with 100 m interstep), receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme.



*Figure 78. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the 24 hour averaging period, displayed in QGIS. The map is zoomed in over the refinery and the city of Falconara Marittima, highlighting the proximity of the industrial plant and the concentration over the urban area. Receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme.*



*Figure 79. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the monthly hour averaging period, displayed in QGIS. The map is zoomed in over the refinery and the city of Falconara Marittima, highlighting the proximity of the industrial plant and the concentration over the urban area. Receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme.*



*Figure 80. Modeled 15 point source SO<sub>x</sub> emissions from "API" refinery in Falconara Marittima, Italy, for the total (the year 2020.) averaging period, displayed in QGIS. The map is zoomed in over the refinery and the city of Falconara Marittima, highlighting the proximity of the industrial plant and the concentration over the urban area. Receptor height is 1.5 m. The legend features concentration bins and their corresponding color scheme.*



### 3.3 DATA POST PROCESSING

In MATLAB, the concentration bins and their corresponding number of nodes in the receptor grid were input to calculate the areal range of each concentration, for each averaging period. The total area of the domain is 40 km<sup>2</sup>. The highest areal range for the 24 averaging period belongs to the areal coverage is of the 5.7990 - 9.8000 µg/m<sup>3</sup> bin with 31% of the domain and 12.662 km<sup>2</sup> of the surface, and 3.4320 - 5.7990 µg/m<sup>3</sup> bin with 28% of the domain and 11.254 km<sup>2</sup> of surface.

SO <sub>x</sub> 24h - Averaging Period				Total Area	40
Bin Minimum [µg/m <sup>3</sup> ]	Bin Maximum [µg/m <sup>3</sup> ]	Color	N <sup>o</sup> of Data Points	Areal Coverage [%/100]	Area [km <sup>2</sup> ]
0.1474	0.2491		0	0	0
0.2491	0.4209		0	0	0
0.4209	0.7112		0	0	0
0.7112	1.2020		0	0	0
1.2020	2.0310		350	0.00875	0.35
2.0310	3.4320		6506	0.16265	6.506
3.4320	5.7990		11254	0.28135	11.254
5.7990	9.8000		12662	0.31655	12.662
9.8000	16.5600		6889	0.172225	6.889
16.5600	27.9800		1847	0.046175	1.847
27.9800	47.2900		450	0.01125	0.45
47.2900	79.9076		42	0.00105	0.042













*Table 8. 3D view of sources created in Google Earth including labels and real heights*

The highest areal range for the monthly averaging period belongs to the areal coverage is of 0.4209 - 0.7112  $\mu\text{g}/\text{m}^3$  bin with 26% of the domain and 10.272  $\text{km}^2$  of surface, 0.7112 - 1.2020  $\mu\text{g}/\text{m}^3$  bin with 24% of the domain and 7.755  $\text{km}^2$  of the surface, and 1.2020 - 2.0310  $\mu\text{g}/\text{m}^3$  bin with 31% of the domain and 8.516  $\text{km}^2$  of the surface.

SO <sub>x</sub> Month - Averaging Period				Total Area	40
Bin Minimum [ $\mu\text{g}/\text{m}^3$ ]	Bin Maximum [ $\mu\text{g}/\text{m}^3$ ]	Color	N° of Data Points	Areal Coverage [%/100]	Area [ $\text{km}^2$ ]
0.1474	0.2491		0	0	0
0.2491	0.4209		6108	0.1527	6.108
0.4209	0.7112		10272	0.2568	10.272
0.7112	1.2020		9755	0.243875	9.755
1.2020	2.0310		8516	0.2129	8.516
2.0310	3.4320		3700	0.0925	3.7
3.4320	5.7990		1154	0.02885	1.154
5.7990	9.8000		395	0.009875	0.395
9.8000	16.5600		96	0.0024	0.096
16.5600	27.9800		4	0.0001	0.004
27.9800	47.2900		0	0	0
47.2900	79.9076		0	0	0

*Table 9. Concentration bins along with the number of corresponding nodes in the analysis, its areal coverage in percentage and  $\text{km}^2$ , and color scheme for the monthly averaging period*

The highest areal range for the total (1 year) averaging period belongs to the areal coverage is of 0.4209 - 0.7112  $\mu\text{g}/\text{m}^3$  bin with 29% of the domain and 11.697  $\text{km}^2$  of surface, and 0.2491 - 0.4209  $\mu\text{g}/\text{m}^3$  bin with 25% of the domain and 10.078  $\text{km}^2$  of the surface.

SO <sub>x</sub> Total Period (1 Year) - Averaging Period				Total Area	40
Bin Minimum [ $\mu\text{g}/\text{m}^3$ ]	Bin Maximum [ $\mu\text{g}/\text{m}^3$ ]	Color	N° of Data Points	Areal Coverage [%/100]	Area [ $\text{km}^2$ ]
0.1474	0.2491		7527	0.188175	7.527
0.2491	0.4209		10078	0.25195	10.078
0.4209	0.7112		11697	0.292425	11.697
0.7112	1.2020		7440	0.186	7.44
1.2020	2.0310		2304	0.0576	2.304
2.0310	3.4320		660	0.0165	0.66
3.4320	5.7990		225	0.005625	0.225
5.7990	9.8000		69	0.001725	0.069
9.8000	16.5600		0	0	0
16.5600	27.9800		0	0	0
27.9800	47.2900		0	0	0
47.2900	79.9076		0	0	0

*Table 10. Concentration bins along with the number of corresponding nodes in the analysis, its areal coverage in percentage and  $\text{km}^2$ , and color scheme for the total period (1 year) averaging period*

Basic statistics were done in MATLAB including maximum and minimum, mean, median, 25<sup>th</sup> and 75<sup>th</sup> percentile values, mean values of the maximum and minimum bin, and the mass of the pollutant in a 1 m thickness layer, over the domain, at receptor height (Table 11). The distribution of concentrations was visualized for each period using a boxplot, with a logarithmic scale (fig. 81) and the areal distribution of the concentrations was plotted in figure 82.

Averaging Period SO <sub>x</sub> [ $\mu\text{g}/\text{m}^3$ ]	24 h	Month	Total Period (1 Year)
Maximum	79.9076	18.19	8.597
Minimum	1.492	0.261	0.1474
Median	7.7995	0.9566	0.5661
Mean	8.0143	1.2707	0.6428
98th Percentile	53.134	15.283	8.111
75th Percentile	7.7995	1.6165	0.9566
25th Percentile	4.6155	0.5661	0.335
Maximum Bin	63.5988	22.27	7.7995
Minimum Bin	1.6165	0.335	0.1982
Total Pollutant Mass at Receptor Height [kg]	320.5736	50.8264	25.7104

Table 11. Maximum and minimum, mean, median, 25th and 75th percentile values, mean values of the maximum and minimum bin, and the mass of the pollutant in a 1 m thickness layer, over the domain, at receptor height. Data was calculated using MATLAB, and using data from the AERMOD simulation of air pollution in Falconara Marittima, Italy

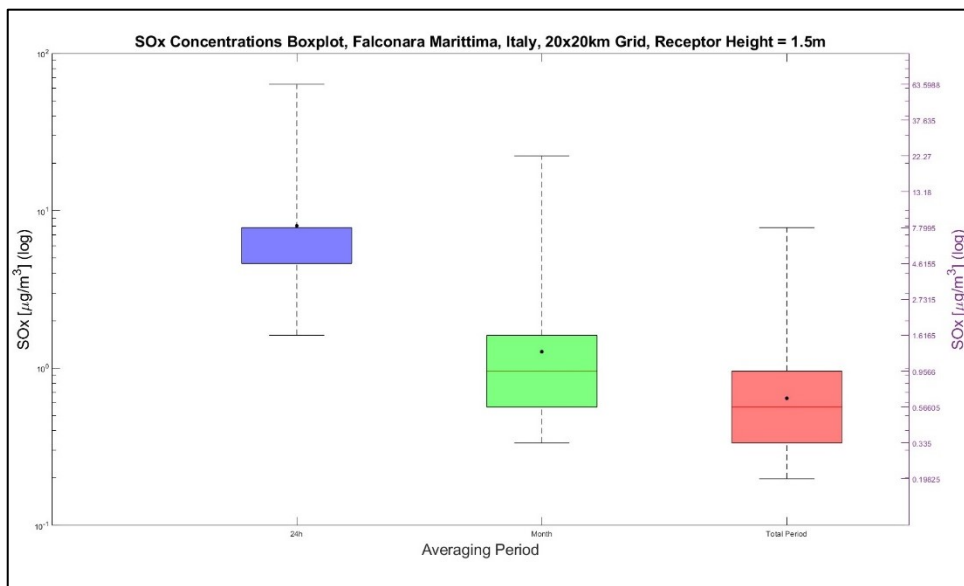


Figure 81. SO<sub>x</sub> concentrations for 24 hour (blue), monthly (green) and total (red - 1 year) averaging periods, visualized using boxplots and a logarithmic scale to visualize the concentration distribution

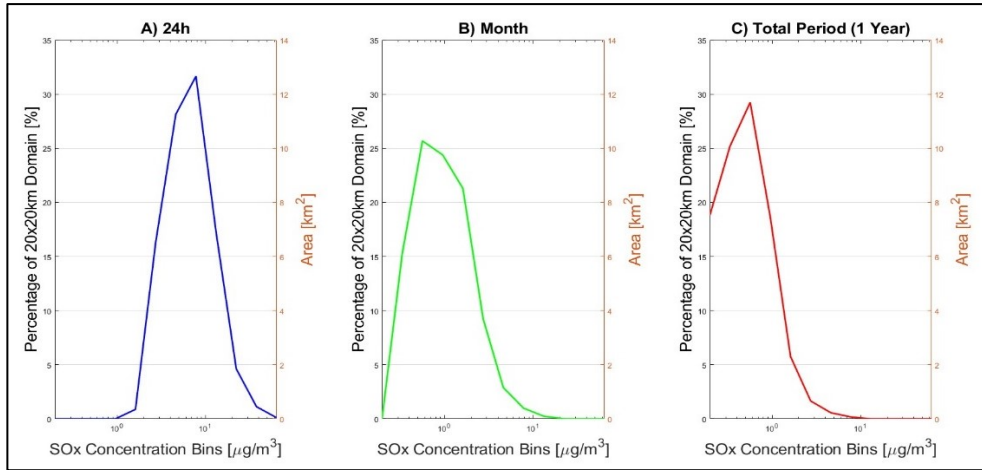


Figure 82. SO<sub>x</sub> concentration areal distribution for 24 hour (blue), monthly (green) and total (red - 1 year) averaging periods. The concentration distribution is plotted on x with a logarithmic scale and the percentage of the domain is plotted on the y axis.

### 3.4 MODEL VALIDATION

Data from “ARPA” for the year 2020. was analyzed in MS Excel to achieve values of interest and values that can be compared to AERMOD model outputs.

Receptor 1) Falconara Acquedotto Hourly data - SO <sub>2</sub> [µg/m <sup>3</sup> ] - Year 2020.			
Averaging Period	24h	Month	Period
Mean	4.54	4.52	4.53
Median	4.46	4.56	4.00
Maximum	10.67	6.24	47.00
Minimum	0.91	2.57	0.00
75 <sup>th</sup> Percentile	5.25	4.89	5.00
25 <sup>th</sup> Percentile	3.79	4.08	4.00

*Table 12. Maximum, minimum, mean, median, 25th and 75th percentile values for the Falconara Acquedotto monitoring station for the year 2020.*

Receptor 2) Falconara Scuola Hourly data - SO <sub>2</sub> [µg/m <sup>3</sup> ] - Year 2020.			
Averaging Period	24h	Month	Period
Mean	5.02	4.92	4.84
Median	4.37	4.84	4.00
Maximum	45.63	6.56	92.00
Minimum	1.09	3.38	0.00
75 <sup>th</sup> Percentile	5.46	5.73	5.00
25 <sup>th</sup> Percentile	3.33	4.11	3.00

*Table 13. Maximum and minimum, mean, median, 25th and 75th percentile values for the Falconara Scuola monitoring station for the year 2020.*

Location 3) Falconara Alta Hourly data - SO <sub>2</sub> [µg/m <sup>3</sup> ] - Year 2020.			
Averaging Period	24h	Month	Period
Mean	4.22	4.21	4.24
Median	4.33	4.32	4.00
Maximum	12.61	5.40	48.00
Minimum	1.29	2.75	1.00
75th Percentile	5.08	4.91	5.00
25th Percentile	3.04	3.53	3.00

*Table 14. Maximum and minimum, mean, median, 25th and 75th percentile values for the Falconara Alta monitoring station for the year 2020.*

The mean values of each station and for each averaging period were compared to the results of the AERMOD simulation with the highest 98<sup>th</sup> percentile excluded, the difference was calculated into the percentage of the real measured value and averaged as a total model difference (table 16).

Difference in distance from Receptor Grid Nodes (m)	
Falconara Acquedotto	52.78
Falconara Scuola	45.44
Falconara Alta	25.58

*Table 15. Difference in location of modeled to real receptors*

Average Monitoring Station Averaging Period Differences - SO <sub>x</sub> [µg/m <sup>3</sup> ]										
Analyzed Hourly Receptor Values				AERMOD DIFFERENCE - SO <sub>x</sub> [µg/m <sup>3</sup> ]						
Averaging periods				24h		Month		Period		
Receptor Name	24h	Month	Period	Rest	%	Rest	%	Rest	%	
Falconara Acquedotto	4.54	4.52	4.53	14.11	310.88	3.61	79.86	0.08	1.74	
Falconara Scuola	5.02	4.92	4.84	7.77	154.85	-0.42	-8.54	-2.15	-44.39	
Falconara Alta	4.22	4.21	4.24	18.23	432.07	2.27	53.83	-0.39	-9.14	
AERMOD F. Acquedotto	18.65392	8.12971	4.609	Total Model Difference				4.79	107.9 %	
AERMOD F. Scuola	12.79336	4.49985	2.6915	24 h Mean		Month Mean		Period Mean		
AERMOD F. Alta	22.45321	6.47621	3.8525	13.37	299.26	1.82	41.72	-0.82	-17.26	

*Table 16. Tabular data of AERMOD modelled difference, comparing the AERMOD analysis with grid receptors (highest 98<sup>th</sup> percentile excluded) compared to mean data per averaging period from 3 monitoring stations, including percentages*

The same was done using the maximal averaging periods calculated from the data from the monitoring stations (table 16.). The maximal averaging period for the whole period is not applicable here so mean values are reused for the total averaging period. As AERMOD was set to use the largest value at each receptor, comparing to the **maximal averaging period** is the more appropriate method.

Maximum Monitoring Station Averaging Period Differences - SO <sub>x</sub> [µg/m <sup>3</sup> ]										
Analyzed Hourly Receptor Values				AERMOD DIFFERENCE - SO <sub>x</sub> [µg/m <sup>3</sup> ]						
Averaging periods				24h		Month		Period		
Receptor Name	24h	Month	Period	Rest	%	Rest	%	Rest	%	
Falconara Acquedotto	10.67	6.24	4.53	7.98	74.83	1.89	30.28	0.08	1.74	
Falconara Scuola	45.63	6.56	4.84	-32.84	-71.96	-2.06	-31.40	-2.15	-44.39	
Falconara Alta	12.61	5.4	4.24	9.84	78.06	1.08	19.93	-0.39	-9.14	
AERMOD F. Acquedotto	18.65392	8.12971	4.609	Total Model Difference				-1.84	5.33%	
AERMOD F. Scuola	12.79336	4.49985	2.6915	24 h Mean		Month Mean		Period Mean		
AERMOD F. Alta	22.45321	6.47621	3.8525	-5.00	26.97	0.30	6.27	-0.82	-17.26	

*Table 17. Main tabular data comparing the AERMOD analysis with receptor grid (highest 98<sup>th</sup> percentile excluded) compared to maximum data per averaging period, from 3 monitoring stations, including percentages*



The same AERMOD analysis was done except for using discrete receptors located by replicating the three monitoring station's locations. The maximum table (table 18.) averaging periods were compared to AERMOD output, and a total difference is calculated. The highest 98<sup>th</sup> percentile was excluded from output values to follow regulations.

Discrete receptor analysis - Maximum Monitoring Station Averaging Period Differences - SO <sub>x</sub> [µg/m <sup>3</sup> ]									
Analyzed Hourly Receptor Values				AERMOD DIFFERENCE - SO <sub>x</sub> [µg/m <sup>3</sup> ]					
Averaging periods				24h		Month		Period	
Receptor Name	24h	Month	Period	Rest	%	Rest	%	Rest	%
Falconara Acquedotto	10.67	6.24	4.54	7.92	74.19	2.53	40.60	0.64	14.16
Falconara Scuola	45.63	6.56	4.84	0.00	0.00	-1.88	-28.67	-2.02	-41.82
Falconara Alta	12.61	5.4	4.24	10.22	81.02	1.28	23.79	-0.16	-3.66
AERMOD F. Acquedotto	18.58585	8.77374	5.1828	Total Model Difference				2.06	8.68%
AERMOD F. Scuola	8.46756	4.67913	2.81576	24 h Mean		Month Mean		Period Mean	
AERMOD F. Alta	22.82677	6.68443	4.08494	6.04	51.74	0.65	11.91	-0.51	-10.44

*Table 18. Tabular data comparing the AERMOD analysis with discrete receptors (highest 98<sup>th</sup> percentile excluded) compared to real data from 3 monitoring stations, including percentages*

Difference data of both gridding systems were compiled and compared to limit values, to give a sense of scale, e.g. the percentage part of the difference of modeled to real data, out of the regulatory limit for SO<sub>x</sub> for each averaging period.

Model-Limit Scaling - SO <sub>x</sub> [µg/m <sup>3</sup> ] – Receptor Grid				
	Limit	Limit Fraction	Difference	
Avg. Period	µg/m <sup>3</sup>	%	µg/m <sup>3</sup>	%
24h	125	4.00	-5.00	26.97
Month	20	1.51	0.30	6.27
Period (Year)	20	4.10	-0.82	-17.26
Total Model Difference		3.20	-1.84	5.33 %

Table 19. Overview of differences between receptor grid modeled and real data. “Limit Fraction” is the modelled differences percentage of the SO<sub>x</sub> limit

Model-Limit Scaling- SO <sub>x</sub> [µg/m <sup>3</sup> ] – Discrete Receptors				
	Limit	Limit Fraction	Difference	
Avg. Period	µg/m <sup>3</sup>	%	µg/m <sup>3</sup>	%
24h	125	4.84	-5.00	26.97
Month	20	3.23	0.30	6.27
Period (Year)	20	2.56	-0.82	-17.26
Total Model Difference		3.54	2.059	8.68

Table 20. Overview of differences between discrete receptor modeled and real data. “Limit Fraction” is the modelled differences percentage of the SO<sub>x</sub> limit

Difference between real and modelled data in relation to regulatory limits						
SOX [ $\mu\text{g}/\text{m}^3$ ] – Grid receptor						
Averaging Period			24h	Month	Year	Mean
Falconara Acquedotto	Limit fraction	%	6.387	9.449	0.395	5.410
	Model difference	$\mu\text{g}/\text{m}^3$	7.984	1.890	0.079	3.318
		%	74.826	30.284	1.744	35.618
Falconara Scuola	Limit fraction	%	26.269	10.301	0.395	12.322
	Model difference	$\mu\text{g}/\text{m}^3$	-32.837	-2.060	0.079	-11.606
		%	-71.963	-31.405	1.744	-33.875
Falconara Alta	Limit fraction	%	7.875	5.381	10.743	7.999
	Model difference	$\mu\text{g}/\text{m}^3$	9.843	1.076	-2.149	2.924
		%	78.059	19.930	44.390	17.866
Mean AERMOD difference	Limit fraction	%	4.003	1.510	4.095	3.202
	Model difference	$\mu\text{g}/\text{m}^3$	-5.003	0.302	-0.819	-1.840
		%	26.974	6.270	-17.262	5.327

Table 21. Detailed differences between discrete receptor modeled and real data in relation to regulatory limits. “Effective %” is the modelled differences percentage of the SO<sub>x</sub> limit

Model Validation			
Method	Result	Range	Acceptable
NMSE	1.015	0.5-2.0	Yes
FB	-0.0656	-0.2 - 0.2	Yes
MB	5.33%	-20% - 20%	Yes

Table 22. Results of testing by Normalized Mean Square Error (NMSE), Fractional Bias (FB) and Mean Bias (MB) tests to measure the model’s results compared to observed data for compliance to European Union regulation

## ***4. DISSCUSSION***

### ***4.1 LOCAL FACTORS***

Falconara Marittima is located in “Valle Dell'Esino”, or “Esino” River Valley, and is proclaimed by “ARPA” an “AERCA (Area a Elevato Rischio di Crisi Ambientale)”, or a high risk of environmental crisis zone, due to the high industrial and maritime pollution in the area. This designation implies the requirement for more rigorous pollution control, regulatory oversight, and stricter regulatory measures due to the complex interplay of emission sources and the region's susceptibility to high pollution loads. The “Esino” valley encompasses the cities Agugliano, Ancona, Camerata, Chiaravalle, Falconara Marittima, Jesi, Monsano, Montemarciano and Monte San Vito. This area has been recognized for its susceptibility to pollution due to the presence of multiple sources of emissions, geographical features that influence pollutant dispersion, roadways, and a dense population that further complicates air quality management. The valley structure combined with prevailing meteorological conditions makes it likely for pollutants to be trapped, especially during periods of stagnant winds or under the influence of sea breeze. The valley's topography restricts the dispersion of pollutants, resulting in periods of stagnant air, particularly during conditions characterized by low wind speeds and inversion layers. Furthermore, the breezes (N-NW) originating from the Adriatic Sea interact with these geographical features, and often drive pollutant transport inland towards residential zones, rather than allowing them to disperse effectively into the upper atmosphere and over the sea, (Regione Marche, 2024.). These site-specific conditions are well-reflected in the dispersion pattern outputs generated by AERMOD, as the concentration abruptly decreases over sea. The AERMOD modeling results indicate higher concentrations in receptor locations downwind of the refinery, particularly during the occurrence of north-western winds driving the plume towards Ancona. Pollution also gets trapped within the valley, leading to peak concentrations.

The results of the dispersion modeling, conducted using AERMOD for Falconara Marittima, highlight significant patterns of pollutant concentration that align with the regional characteristics and existing sources of pollution. This analysis focused specifically on the emissions originating from the API oil refinery, which plays a crucial role in the air quality in the surrounding area. The Ancona harbor is another significant pollution source in the vicinity that adds to the concentration of contaminants, particularly due to emissions from ships at rest with running engines, as its their only source of electricity.

The Ancona harbor also plays a significant role in pollution levels within the study area. The harbor's impact is not included in the AERMOD model, which only considers emissions from the refinery. Being Italy's major harbor on the Adriatic coast, presents a unique challenge to the local environment. In addition to the refinery, emissions from the harbor contribute significantly to overall pollution levels, especially for short-term peak periods. The study, conducted by Fileni et al. in 2019. concluded that the port of Ancona contributed significantly to the levels of carbon monoxide (CO), volatile organic compounds (VOCs), nitrogen oxides (NO<sub>x</sub>), sulfur oxides (SO<sub>x</sub>), and particulate matter (PM). The main sources of emissions include "Roll-on/roll-off" (Ro-ro) and "Roll-on/roll-off Passengers" (Ro-pax) ships as well as fishing vessels. SO<sub>x</sub> emissions are directly linked to the sulfur content of the fuel and therefore also to fuel consumption. Different fuel types are used by ships during maneuvering, cruising and "hoteling", e.g. the power required to maintain ship operations while at berth. During maneuvering ships use Marine Diesel Oil (MDO) or Marine Gas Oil (MGO), which has a low Sulphur content of around 1.5%, which have lower viscosity and are better suited for quick changes in engine speed and power, which are often required during maneuvering. During cruising Heavy Fuel Oil (HFO) is used, with higher viscosity and high Sulphur content. During hoteling ships are required by European Directive 2005/33/EC, to use Marine Gas Oil (MGO) to reduce emissions in ports, the Sulphur content in the fuel is required to be less than 0.5%. Emission is highest during maneuvering and hoteling. In Naples, 98% of NO<sub>x</sub> and SO<sub>x</sub> emissions from cruise ships were due to hoteling, with only 2% attributed to maneuvering. In Ancona 76% of PM10 emissions were during hoteling. (Fileni, et al., 2019.)

The API refinery is a primary contributor to local SO<sub>2</sub>, NO<sub>x</sub>, and particulate emissions. Falconara Marittima was defined by Decree of the Ministry of the Environment and Land Protection, as an "Area at High Risk of Environmental Crisis" (AERCA). With the Decree of the Ministry of the Environment and Protection of Land and Sea no. 308 of 28 November 2006, financial resources amounting to € 3,272,727.00 were assigned to the Falconara Marittima site. (ARPAM, 2023.)

The following areas are part of the "Falconara Marittima" National Interest Site:

- api Raffineria di Ancona S.p.A.;
- former Montedison plant;
- internal areas of Aerdorica S.p.A.;
- former Liquigas – Castellaraccia area;
- former chemical-bitumen industry;
- area of via Monti and Tognetti;
- RFI area in front of the former Montedison site
- sports field of the parish of S. Maria della Neve and S. Rocco;
- former Gattini mechanical workshop;
- former Vibrocementi;
- former R.S.U. landfill

The perimeter also includes the marine area facing the terrestrial area that extends from the Api Refinery to the former Montedison for a total surface area of approximately 1200 ha. (ARPAM, 2023.)

The Falconara Marittima National Interest Site is located in the alluvial plain near the mouth of the "Esino" River. The marine area facing this site is characterized by shallow seabed with high oceanographic dynamism, with marked seasonal and interannual variations influenced by the strong temperature range that occurs between the winter and summer seasons due to the shallow depth of the seabed and the contributions of fresh river water due to the presence of the "Esino" River that flows in the vicinity of the Api complex. (ARPAM, 2023.)

The three monitoring stations exhibit different trends. The “Acquedotto” and “Scuola” receptors, located in the vicinity of the refinery, are positioned within the direct line of pollutant fallout.

The “Acquedotto” monitoring station is directly downwind of the refinery emissions during NE winds and experiences plume trapping during NW winds and inversion. The data showed elevated concentrations of pollutants during these wind events, consistent with predicted results from the model, indicating that the plume from the API refinery heavily affects this area.

The “Scuola” monitoring station, positioned near the elementary school in “località Villanova di Falconara,” is affected during the predominating N-NW winds, which drive refinery emissions into the urban center. The modeled concentrations confirm that pollutants are transported towards the school and residential areas, underscoring the need for stringent air quality control to safeguard sensitive populations.

“Alta” station monitors emissions during northerly winds and is affected similarly. AERMOD predicts a concentration pattern that matches the expected path, confirming the accuracy of the emission data and, meteorological and terrain inputs. These stations illustrate how localized micro-conditions can dictate pollutant exposure which further emphasizes the vulnerability of residential areas. (ARPAM, 2023.)

These factors contribute to pollution and importance of environmental monitoring in Falconara Marittima, where we observe a similar pattern of pollutant distribution due to the combination of port activities, refinery operations and local meteorological conditions. And are fundamental in understanding and interpreting the AERMOD results. The presence of other pollutant sources increases pollution both quantitatively and qualitatively. Related to modeling results it justifies any concentration overestimations made by the model and diminishes underestimations, due to real-world contributions to pollution levels by other sources, as it likely under-represents actual exposure levels in Falconara Marittima. Future modeling efforts should therefore incorporate emissions from the Ancona harbor, as it provides a crucial background pollutant concentration, to provide a more comprehensive

understanding of local air quality. Local policy should consider additional emission control technologies for the refinery during periods of meteorological stability that lead to pollutant trapping, e.g. Continuous Emission Monitoring Systems (CEMS) with Predictive Control, allowing for continuous monitoring of SO<sub>x</sub> emissions. When periods of meteorological stability are predicted, CEMS can trigger operational adjustments or the activation of additional control measures. Differences observed between short-term and long-term averaging periods underscore the necessity for tailored approaches to regulatory compliance, focusing on minimizing both peak exposure and annual average concentrations. Additionally, reducing emissions from Ancona port, given the proximity of industrial and residential areas in Falconara, could be crucial in minimizing the population's exposure to harmful pollutants.



## ***4.2 “CAIRO” PERFORMANCE***

The primary goals of modeling atmospheric dispersion and creation of the “CAIRO for AERMOD” application are to assess and understand the distribution and concentration of pollutants emitted from major industrial sources, particularly the API refinery in this case study. To estimate pollutant exposure levels for residents and compare these levels against established regulatory standards for health and safety. To determine the dispersion characteristics of pollutants, provide a scientific basis for policymakers to implement mitigation measures. To understand how geographical influences affect air quality.

AERMOD was chosen over other software due to its straightforward setup and the accuracy it provides for pollution modeling. The choice of AERMOD is primarily due to its well-established credibility and straightforward application for short-scale assessments (up to 50 km). Falconara Marittima, being impacted primarily by the refinery and located within a close geographically defined area, fits well within the scope of AERMOD's capabilities. Complex terrain and surface interactions, which include coastal areas, industrial zones, and residential areas, are handled efficiently through its terrain processing algorithms, making it suitable for Falconara's mixed landscape. AERMOD includes detailed formulations to manage boundary layer physics, important for the Falconara area, which has varied surface types (water bodies, urban areas, vegetated regions). Surface meteorological data and upper air data can be easily integrated into AERMOD, while models like CALPUFF or WRF require more complex meteorological preprocessing and a longer computational setup and are suited for large scale analyses.

In the initial stages of developing the “CAIRO, Python, together with the PyCharm interface, was leveraged primarily for its versatility and the availability of numerous supporting libraries that facilitate rapid application development. Python's readability and adaptability were key to developing the initial and consecutive versions. Its capacity for combining powerful data analysis libraries allowed us to create a comprehensive air quality modeling and analysis tool with interactive components. Such as “Pandas” and “NumPy”, with user interface toolkits, “PyWin32” to handle fetching coordinates, “utm” for converting them and “simplekml”, among others, for real time visualization.

For the graphical user interface, Tkinter was used, as it a lightweight GUI toolkit for Python. Tkinter provided a quick way to create dialog windows, form entries, and other components that were needed for user interaction without having to dive into the complexities of a more sophisticated library. The simplicity of Tkinter made it easy to iterate on the user interface, which was crucial during the prototyping phase. It allowed to make modifications to suit the requirements of users.

The ”CAIRO” application allowed for all stages to be run from one window, compiled necessary input files and guided and aided throughout the modeling process by simple menus, pop-up information boxes and available video tutorial. There was no need to set up different pathways (Control, Source, Receptor...), to know correct syntax, to manipulate with folders and files, or to work through different terminals and command shells. The input of UTM coordinates and zone is possible through different inputs, from copying latitude longitude coordinates to be automatically visualized and converted, to manually inputting them. The application offers a simple workflow, containing crucial AERMOD functionalities, enabling for a simple and user-friendly interface and workflow, while still being able to achieve statistically and legislatively significant results.

After obtaining elevation data from Copernicus Services, the process of compiling input and running AERMAP was straightforward. By choosing the origin of the receptor grid via Google Maps, the zone and other accessory data is automatically compiled, the elevation data is loaded trough the interface.

The integration of Google Maps and Google Earth with AERMOD was instrumental in streamlining the input process. For instance, defining emission sources via Google Maps ensured precise geolocation and helped minimize errors related to manual data entry. Moreover, visualization in Google Earth provided immediate feedback on the receptor grid's spatial setup and the positioning of pollutant sources, ensuring the model reflected real-world scenarios more accurately and created a shapefile containing point or polygon source locations. Loading of terrain and meteorological data is conveniently done through the interface. AERPLOT fetched all necessary files during compilation and offered all basic functions.

The program had its disadvantages in requiring internet connection for full functionality, had occasional instability issues and uses a limited amount of AERMOD's functionalities. These are the other disadvantages that are planned to be resolved in future versions of "CAIRO":

- simpler interface
  - availability for other platforms
- AERMAP :
- no visual preview of elevation data or receptor domain in Google Earth or QGIS
  - only receptor grid is available, polar and discrete receptors should be included to analyze specific locations, polar grids have a higher resolution near source, which would suit the analysis in Falconara Marittima
- AERMOD:
- limited source types (point, aerapoly), line and volume should be included
  - only three averaging periods are available
  - some limitation in output, still the most important output is available
  - input of meteorological station numbers could be automated
- AERPLOT:
- custom bins are not available, but custom binning ranges are available with the choice of logarithmic or linear
  - creation of multiple iterations could be possible for every plot file (currently for every averaging period a corresponding plot is created, a possibility is to create both logarithmic and linear plots, or with different ranges and bins at once for all averaging periods)

The program excelled at its intended purpose and made the creation of a complex multisource analysis user friendly. Its real time visualization capabilities offered the greatest advantage, both by simplifying the input process, creating accessory files for post processing, offering real time confirmation of source location and automatically converting coordinates and creating vertices of polygons. Creation of multiple graphical concentration maps at once made it easier to obtain results and post process them. The "CAIRO for AERMOD" application represents a significant advancement in user-friendly PBL dispersion modeling. By integrating essential tools, real-time visualization, and straightforward data input methods, it addresses the complexities of air quality modeling with efficiency and accessibility. Though there are some limitations to the platform that will be addressed in future updates, "CAIRO" excels in meeting the practical demands of pollutant dispersion analysis, particularly in complex terrains like Falconara Marittima. This tool not only enhances the capacity for accurate and expedient environmental assessments but also equips policymakers and environmental scientists with a reliable foundation for evaluating air quality impacts and exploring mitigation measures. With ongoing development, "CAIRO" holds the potential to evolve into a robust, multi-platform tool capable of adapting to broader contexts and more diverse modeling needs.

The program has been uploaded to <https://sourceforge.net/projects/cairo-for-aermod/> and is free to download.

### ***4.3 CASE STUDY***

Elevation data was downloaded in a sufficient resolution of 30 m/px, which is either way smaller than the receptor grids 100 m resolution and the recommended size of 100\*100m to 500\*500m by EEA, for local applications, analyzing one city or one industrial plant. For urban/metropolitan scales a 1\*1km to 5\*5km grid is recommended, while for regional a 1\*1° or approximately 11\*11km grid size is recommended (EMEP, EEA, 2009.).

Receptor nodes were defined using a Cartesian grid network, ensuring comprehensive spatial coverage, particularly along critical downwind directions identified based on local meteorological patterns. A polar receptor grid could have been used to inspect concentrations more accurately near the plant. Meteorological data was supplied by the university via Co-Supervisor.

Different binning ranges and methods were applied to plot the output to represent the data suitably for each averaging period. QGIS, MATLAB and MS Excel played a crucial role in data representation, postprocessing and model validation.

As explained before, the refinery sources do not represent all pollution emission sources influencing the area, so modeled values were expected to underestimate. The final result showed a slight overestimation of 5.33%, either due to the model internally overestimating, inaccuracy in source, elevation or meteorological data, improperly setup model, or insufficient receptor replicants (real monitoring stations to compare to).

The analyzed data showed no values exceeding the regulatory limit values. Highest concentrations are found in shorter averaging periods because peaks are better represented in a shorter time frame/sample number. Also, the highest discrepancies in concentration were found in shorter averaging periods. Concentration distribution patterns strongly co-align with predominant wind directions, while entrapment due to topography is obvious in the valleys and due to NW wind driving pollution inland. The pollutant plume holds steady concentrations reaching as far as Ancona. Residential including school areas are most exposed to refinery pollution due to proximity.

Concentration distribution, observing both modelled and observed data is highly spatially variable due to complex topology and meteorological influences. Building downwash modelling, using BPIPPRM (Building Profile Input Program for PRIME) which is a preprocessor of AERMOD, could be beneficial to model in this residential and topographically complex area.

The shore south and north of Falconara exhibits lower concentration, but that picture might change if Ancona port and transit emissions were included. In the case of 24-hour averaging, the AERMOD output provides an insight into maximal pollutant impact at each receptor, which is critical for assessing health risks in vulnerable populations, such as those residing near “Scuola” and “Aquedotto”. Long term averaging gives insight into long term exposure, which often isn’t reflected in short-term averaging periods.

Statistical indicators like mean, median, percentiles, and pollutant mass are used to characterize pollution levels in depth.

Percentiles such as the 98<sup>th</sup>, 75<sup>th</sup> and 25<sup>th</sup> provide insights into the variability of the pollution concentrations, helping interpret the impact at different spatial ranges and temporal resolutions (24-hour, month, annual).

The 98th percentile values calculated in the “RANKFILE”, for specific receptor locations, provide a crucial assessment tool for regulatory compliance, that aims to represent worst-case scenarios while excluding extreme outliers that may not reflect typical conditions.

As AERMOD reports set highest values on the analyzed receptor nodes, so had the observed data been interpreted accordingly, to find the maximum values from averaging periods. Due to the discretization of the receptor grid, modeled locations were 25-52m away from their actual location. Later analysis tried to minimize the error by using discrete receptors with exact locations, but the receptor grid still showed to more accurate by 10%. This may be caused by high spatial variability in the area, inaccurate input data or inadequate number of replicants. A box analysis on receptor sites could have produced more exact values, though it might be contraindicated due to high spatial variability.

The observed data showed concentrations up to 45.63  $\mu\text{g SO}_x/\text{m}^3$  at the “Falconara Scuola” receptor, arguably the most sensitive area, occupied by children<sup>3</sup>, which is well below the regulatory limit of 125  $\mu\text{g}/\text{m}^3$ , representing approximately 36.5% of the limit, but still produces a compounding effect.

The median and mean values lie around 4-5  $\mu\text{g SO}_x/\text{m}^3$ , creating a steady background concentration and a compounded exposure to the population.

The "Falconara Aquedotto" receptor experiences lower concentration peaks at 10.67  $\mu\text{g SO}_x/\text{m}^3$ , but has a relatively high background concentration of 4.54  $\mu\text{g SO}_x/\text{m}^3$ , due to proximity to the plant, trapping of gas at lower elevations and NE winds. Modeled concentrations in the short term had a high discrepancy of up to 75%, decreasing to 30.3% and 1.74% for month and year averaging periods, respectively.

The "Falconara Scuola" receptor experiences the highest concentration peaks at 45.63  $\mu\text{g SO}_x/\text{m}^3$ , but has the highest background concentration of 4.84  $\mu\text{g SO}_x/\text{m}^3$ , due to proximity to the plant and predominant NW winds. Modeled concentrations showed the largest differences, compared to observed data. The highest underestimation at -72% for the 24h period, decreasing to -31.4% for month and again rising to -44.39% for the year averaging period. The peak of 45.63  $\mu\text{g SO}_x/\text{m}^3$ , was vastly underestimated by the model with 12.79  $\mu\text{g SO}_x/\text{m}^3$ . This shows that the specific subarea experiences a complex interplay of topography, meteorology and very proximal emission. Ground heat flux could have been underestimated by AERMOD in this case, due to presence of infrastructure, increasing the upwards heat flux. Modelling the receptor properly, would require detailed model formulation, including building downwash effect and a finer receptor grid network to conclude which features influence the complex dispersion patterns.

The "Falconara Alta" receptor experiences concentration peaks at 12.61  $\mu\text{g SO}_x/\text{m}^3$ , but has the lowest background concentration of 4.24  $\mu\text{g SO}_x/\text{m}^3$ . Lower concentrations were as expected due to higher distance to the plant, higher elevation, which promote more efficient plume dispersion. The receptor also isn't in the direct way of the predominant wind directions, reducing the measured concentration. Measured values comparable to the ones in higher proximity to the plant (Aquedotto, Scuola), indicate the magnitude of pollution in Falconara Marittima, the trapping of pollutants inland by NE winds, and possible additional sources, influencing the measured value more than the other two receptors. One possibility is the greater proximity of Ancona port to Falconara Alta. Modeled concentrations

in the short term had the highest discrepancy of 78%, decreasing to 19.93% and -9.14% for month and year averaging periods, respectively. The model calculate its highest 98th percentile value at this receptor of 22.45  $\mu\text{g SO}_x/\text{m}^3$ , while the actual data shows only 12.61  $\mu\text{g SO}_x/\text{m}^3$ . This could be due to the program underestimating ground heat flux, or the thickness of the PBL, resulting in a lower modeled dispersion and higher modeled concentration values.

Overall, the model performed very well with a discrepancy from the observed data of -1.84  $\mu\text{g SO}_x/\text{m}^3$  and 5.33%, on average. The 24-hour averaging period produced the greatest discrepancies compared to the observed data, averaging at about  $\pm 75\%$  of difference. This is largely due to model setup and could be mitigated using more complex terrain processing and more detailed meteorological data. The monthly averaging period shows smaller discrepancies up to  $\pm 30\%$ , while the yearly period has smallest discrepancies, excluding the “Scoula” receptor and its underestimation.

Discrete receptor analysis, aimed to decrease discrepancies caused by receptor location discrepancies, shows similar output, with a slightly higher discrepancy in overall predictions at -2.07  $\mu\text{g SO}_x/\text{m}^3$  and 9.68%. Move over, its showed the same high underestimation at “Scuola” receptor and overestimation at Alta receptor, indicating the receptor grid was fine enough, the up to 50 m discrepancy in receptor location isn't significant and the model setup needs further work.

Modelling results, specifically the model discrepancies, were compared to regulatory values (125  $\mu\text{g SO}_x/\text{m}^3$  for 24 h avg. period, and 20  $\mu\text{g SO}_x/\text{m}^3$  for month and year avg. periods) to scale its performance in terms of its ability to influence regulatory compliance, estimation of health hazard and provide a base for further research and legislative efforts. Discrepancies presented only up to 4.10% of the data, averaging at 3.20%. They were largest for the 24 h and yearly averaging period. Monthly averages might provide a balance between capturing significant temporal variations and minimizing the “noise” from short-term fluctuations, resulting in more reliable air quality assessments. Daily variations in weather can cause significant short-term fluctuations in pollutant levels, while yearly averages may obscure these fluctuations altogether.



Additionally, the model performance was also examined through Normalized Mean Square Error (NMSE), Fractional Bias (FB) and Mean Bias (MB), to determine regulatory validity of the results in the EU. The model acquired sufficient results and would be valid for regulatory purposes.

The results conclude that the null hypothesis (The difference between modeled data and real-world measurements in relation to regulatory limit is significant), has been disproven and the hypothesis (There is no significant difference between modeled and real-world data in relation to regulatory limit.) has been confirmed.

Falconara Marittima, designated as a high environmental risk area (AERCA), faces considerable challenges due to industrial emissions and its proximity to the Ancona harbor, both significant sources of air pollutants. Sulfur oxides (SO<sub>x</sub>) concentrations, while within regulatory limits, pose potential health risks due to the cumulative exposure effects. Long-term exposure to elevated SO<sub>x</sub> levels, even below threshold limits, is associated with respiratory issues, especially among sensitive groups like children and the elderly. Short-term peaks, which often coincide with refinery emissions and atmospheric inversions, increase the risk of acute respiratory symptoms, aggravation of asthma, and other pulmonary conditions (WHO, 2006.).

The Ancona harbor also contributes to the pollutant burden, with emissions from ship traffic compounding local air quality issues. Studies suggest that harbor-related emissions, including SO<sub>2</sub> and PM, have a marked impact on coastal and near-port communities, exacerbating the health risks associated with long-term exposure. Given these factors, the results of this study underscore the importance of stringent air quality management and regulatory compliance efforts in Falconara Marittima. Effective policies must address the cumulative impacts of industrial and port-related activities to mitigate adverse health outcomes and safeguard public well-being in this high-risk area.

In conclusion, the use of AERMOD, combined with the graphical interface CAIRO, has allowed for a robust analysis of pollutant dispersion and air quality conditions in Falconara Marittima, focusing on sulfur oxides (SO<sub>x</sub>) concentrations. Using AERMOD to simulate air pollution at using a receptor grid and at three key receptor points: Falconara Acquedotto,

Falconara Scuola, and Falconara Alta, revealed both spatial and temporal variations in pollutant concentrations.

The results align with previous studies, reinforcing the importance of integrated air quality management that includes contributions from multiple emission sources, such as industrial and harbor activities. Further work may involve refining receptor placements to capture additional micro-scale variations and extending the analysis to other pollutants of concern, such as PM<sub>10</sub> and NO<sub>x</sub>. The effectiveness of the AERMOD model, enhanced by the CAIRO interface, provided detailed insights into pollutant dispersion within Falconara Marittima. By focusing on sulfur oxide emissions from the Falconara refinery, the study highlights spatial and temporal pollutant variations that inform both regulatory compliance and environmental health impacts. Although some discrepancies between modeled and observed values were noted, particularly over shorter averaging periods, the overall model accuracy supports AERMOD's and CAIRO's applicability in regulatory and health-risk assessments, contributing to a comprehensive approach to air quality management. The findings emphasize the need for integrated mitigation measures addressing both industrial emissions and transport-related sources to protect public health and ensure regulatory compliance.

## 5. CONCLUSION

This thesis has presented a comprehensive examination of SO<sub>x</sub> emissions from a refinery in Falconara Marittima, Italy, using the AERMOD dispersion model and a newly developed application, "CAIRO for AERMOD". The development of the "CAIRO for AERMOD" software, a Python-based graphical user interface, was a key achievement, designed to streamline and automate the generation and runtime of complex input files required for AERMOD, AERMAP, and AERPLOT. The CAIRO tool simplified previously labor-intensive tasks, allowing users to compile input files with the correct syntax and structure while visualizing input data in Google Earth. Key functionalities of the software, such as automatic coordinate input, UTM conversion, handling file pathways, data formats, and guided support for different types of analysis, demonstrate its utility in environmental modeling, both for novice and experienced users.

The "CAIRO for AERMOD" application proved effective in handling input and output across all phases of the modeling process. By automating coordinate conversions, visualizing sources and receptors, and simplifying file compilation, the software provided a more efficient workflow for environmental assessments. The interface's integration with tools like Google Earth and QGIS enhanced user interaction with georeferenced data, allowing real-time visualization of point sources, receptor grids, and modeled concentrations. This feature is especially useful in environmental impact assessments, where the spatial relationship between emission sources and residential areas is critical.

The case study of the API refinery in Falconara Marittima, Italy, served to validate both the GUI's functionality and the AERMOD model's performance in simulating SO<sub>x</sub> dispersion from complex industrial sources. The study used a 20x20 km receptor grid with 100 m resolution, covering the refinery and its surrounding areas, including the nearby town of Falconara Marittima and 15 point sources stemming from the plant. Three averaging periods (24-hour, monthly, and yearly) were analyzed, providing a multi-scale perspective on the spatial distribution and concentration of SO<sub>x</sub> emissions. Modeled concentrations were compared to actual monitoring data from three industrial receptor sites around the refinery, which allowed for detailed model validation and insight into AERMOD's performance in

complex, real-world scenarios. Across all three averaging periods, AERMOD produced SOX concentration estimates that were within acceptable limits according to Italian legislative thresholds (confirmed by monitoring station data):  $125 \mu\text{g}/\text{m}^3$  for 24-hour averaging, and  $20 \mu\text{g}/\text{m}^3$  for monthly and yearly averages. AERMOD results, when compared to actual monitoring data, showed an average discrepancy of 18.5% or  $-1.088 \mu\text{g SO}_x/\text{m}^3$ , with the model performing better over longer averaging periods (1 year) and less accurately over shorter periods (24 hours). The results indicate that while AERMOD provides a valuable estimation tool for SOX emissions and highlighting potential risk zones, its precision is variable depending on the timeframe, reflecting the challenges of capturing transient atmospheric processes in environmental modeling.

These overestimations would be even greater if other major pollution sources were included, such as the road traffic and nearby Ancona harbor, whose ship emissions contribute significantly to SOX levels in the area but were not included in this model. Model validation highlighted AERMOD's tendency to produce higher discrepancies over shorter time periods (up to 78% discrepancy). The model's accuracy improved with longer averaging periods, with yearly averages closely aligning with observed data and discrepancies reduced to below 20%. In the Falconara Marittima area, additional sources such as harbor emissions may contribute to cumulative pollution levels, suggesting that AERMOD's accuracy could improve by including a broader set of emission sources. The analysis also found that the receptor grid method provided similar results to discrete receptors, although minor variations were observed due to the spatial resolution of the grid and discretization of receptor placement. A limitation of AERMOD is its inability to model chemical transformations of pollutants, such as the oxidation of  $\text{SO}_2$  into sulfate aerosols, which contribute to particulate matter ( $\text{PM}_{2.5}$ ) formation, fog and acid rain. Incorporating the Weather Research and Forecasting model coupled with Chemistry (WRF-Chem) could address this limitation.

AERMOD, supported by the CAIRO application, is a viable tool for assessing industrial air pollution in complex environments. The CAIRO for AERMOD software successfully addressed challenges in input file creation, source visualization, and receptor placement, establishing a workflow that can be applied in similar environmental modeling contexts. By addressing the

technical barriers associated with AERMOD, the "CAIRO for AERMOD" application facilitates improved compliance, scenario analysis, and decision-making. While AERMOD's performance was satisfactory over extended averaging periods, the model's limitations in capturing short-term pollution variability suggest the need for further refinement. Enhancements to CAIRO for AERMOD could include integration with meteorological and elevation data providers. By advancing both the practical application and estimation of accuracy of AERMOD modeling, this study contributes to more reliable, user-friendly and free source air quality assessment tools, supporting efforts to mitigate industrial pollution and protect public health in affected communities. This software offers a practical solution for environmental professionals by enabling efficient input handling and visualization, thereby contributing to more effective monitoring and assessment of air pollution in complex environments.

## 6. REFERENCES

World Health Organization. (n.d.). (20.5.2024.) Air Pollution. World Health Organization. <https://www.who.int/health-topics/air-pollution#>

Fowler, David, et al. "A chronology of global air quality." *Philosophical Transactions of the Royal Society A* 378.2183 (2020): 20190314.

European Environment Agency. (2024, May 27). Air Pollution. European Environment Agency's home page. <https://www.eea.europa.eu/en/topics/in-depth/air-pollution>

EPA National Emission Inventory EPA-454/R-00-003. (2010).

Singh, Anita, and Madhoolika Agrawal. "Acid rain and its ecological consequences." *Journal of Environmental Biology* 29.1 (2007): 15.

Sharma, Shyam Bihari, et al. "The effects of air pollution on the environment and human health." *Indian Journal of Research in Pharmacy and Biotechnology* 1.3 (2013): 391-396.

Lee, Chih-Sheng, Ken-Hui Chang, and Hyunook Kim. "Long-term (2005–2015) trend analysis of PM 2.5 precursor gas NO<sub>2</sub> and SO<sub>2</sub> concentrations in Taiwan." *Environmental Science and Pollution Research* 25 (2018): 22136-22152.

Oppenheim, Hannah A., et al. "Exposure to vehicle emissions results in altered blood brain barrier permeability and expression of matrix metalloproteinases and tight junction proteins in mice." *Particle and Fibre Toxicology* 10 (2013): 1-14.

Calderón-Garcidueñas, Lilian, et al. "Long-term air pollution exposure is associated with neuroinflammation, an altered innate immune response, disruption of the blood-brain barrier, ultrafine particulate deposition, and accumulation of amyloid  $\beta$ -42 and  $\alpha$ -synuclein in children and young adults." *Toxicologic pathology* 36.2 (2008): 289-310.

Tiao, G. C., G. E. P. Box, and W. J. Hamming. "Analysis of Los Angeles photochemical smog data: a statistical overview." *Journal of the Air Pollution Control Association* 25.3 (1975): 260-268.

Wesely, M. L., and B. B. Hicks. "A review of the current status of knowledge on dry deposition." *Atmospheric environment* 34.12-14 (2000): 2261-2282.

Deribe, Ermias, et al. "Biomagnification of DDT and its metabolites in four fish species of a tropical lake." *Ecotoxicology and environmental safety* 95 (2013): 10-18.

Bouare, Oumar. "Impact of Global Warming on Rural-Urban Migration and Net Emigration in Forefront Sub-Saharan Countries." Available at SSRN 1338756 (2009).

Boningari, Thirupathi, and Panagiotis G. Smirniotis. "Impact of nitrogen oxides on the environment and human health: Mn-based materials for the NO<sub>x</sub> abatement." *Current Opinion in Chemical Engineering* 13 (2016): 133-141.

Zhang, Chunlin, et al. "Emission factor for atmospheric ammonia from a typical municipal wastewater treatment plant in South China." *Environmental pollution* 220 (2017): 963-970.

Patel, Sagar, et al. "Physiology, Oxygen Transport And Carbon Dioxide Dissociation Curve." *StatPearls*, StatPearls Publishing, 27 March 2023.

Wang, Y. Q., et al. "Spatial and temporal variations of the concentrations of PM<sub>10</sub>, PM<sub>2.5</sub> and PM<sub>1</sub> in China." *Atmospheric Chemistry and Physics* 15.23 (2015): 13585-13598.

EPA. *Persistent Organic Pollutants: A Global Issue, A Global Response*. 2009., <https://www.epa.gov/international-cooperation/persistent-organic-pollutants-global-issue-global-response> , 1. June 2024.

Chen, Lung-Chi, Polina Maciejczyk, and George D. Thurston. "Metals and air pollution." *Handbook on the Toxicology of Metals*. Academic Press, 2022. 137-182.

Mielke, H.W.; Gonzales, C.R.; Powell, E.T.; Egendorf, S.P. Lead in Air, Soil, and Blood: Pb Poisoning in a Changing World. *Int. J. Environ. Res. Public Health* 2022, 19, 9500. <https://doi.org/10.3390/ijerph1915950018>. May 2024.

ENEA - Piersanti, A.; D'Elia, I.; Gualtieri, M.; Briganti, G.; Cappelletti, A.; Zanini, G.; Ciancarella, L. The Italian National Air Pollution Control Programme: Air Quality, Health Impact and Cost Assessment. *Atmosphere* 2021, 12, 196. <https://doi.org/10.3390/atmos12020196> 16. May 2024.

MASE. (n.d.). Environmental assessments and authorizations: Sea - eia - IPPC permit. MASE - Environmental Assessments and Authorizations - SEA - EIA - IPPC Permit. <https://va.mite.gov.it/en-GB/comunicazione/cittadino>

EEA, 2022, 'European Union emission inventory report under the Convention on Long-range Transboundary Air Pollution (LRTAP) — European Environment Agency', (<https://www.eea.europa.eu/publications/european-union-emissions-inventory-report>). 16. May 2024.

EU, 2016, Directive (EU) 2016/2284 of the European Parliament and of the Council of 14 December 2016 on the reduction of national emissions of certain atmospheric pollutants, amending Directive 2003/35/EC and repealing Directive 2001/81/EC, OJ L 344, 17.12.2016, p. 1–31.

US EPA,OAR. "Air Quality Dispersion Modeling - AERMOD Model Formulation | US EPA." US EPA, 22. July 2019.,

[https://gaftp.epa.gov/Air/aqmg/SCRAM/models/preferred/aermod/aermod\\_mfd.pdf](https://gaftp.epa.gov/Air/aqmg/SCRAM/models/preferred/aermod/aermod_mfd.pdf) 16. May 2024.

Stull, Roland B. An introduction to boundary layer meteorology. Vol. 13. Springer Science & Business Media, 2012.

Stull, Roland B. "Boundary layer clouds." *An Introduction to Boundary Layer Meteorology*. Dordrecht: Springer Netherlands, 1988. 545-585.

Davis, P. A. "Development and mechanisms of the nocturnal jet." *Meteorological Applications* 7.3 (2000): 239-246.



Purdy, A. J., et al. "Ground heat flux: An analytical review of 6 models evaluated at 88 sites and globally." *Journal of Geophysical Research: Biogeosciences* 121.12 (2016): 3045-3059.

Mauder, Matthias, Thomas Foken, and Joan Cuxart. "Surface-energy-balance closure over land: a review." *Boundary-layer meteorology* 177 (2020): 395-426.

Kang, Song-Lak. "Regional Bowen ratio controls on afternoon moist convection: A large eddy simulation study." *Journal of Geophysical Research: Atmospheres* 121.23 (2016): 14-056.

Cho, Jaeil, et al. "On the relationship between the Bowen ratio and the near-surface air temperature." *Theoretical and Applied Climatology* 108 (2012): 135-145.

Friedrich, K., N. Mölders, and G. Tetzlaff. "On the influence of surface heterogeneity on the Bowen-ratio: A theoretical case study." *Theoretical and Applied Climatology* 65 (2000): 181-196.

Weber, Rudolf O. "Remarks on the definition and estimation of friction velocity." *Boundary-Layer Meteorology* 93 (1999): 197-209.

Sheppard, Percival Albert. "The aerodynamic drag of the earth's surface and the value of von Karman's constant in the lower atmosphere." *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 188.1013 (1947): 208-222.

Bonan, Gordon. *Climate change and terrestrial ecosystem modeling*. Cambridge University Press, 2019.

Haby, J. (n.d.). THE PLANETARY BOUNDARY LAYER. Planetary boundary layer. [https://www.weather.gov/source/zhu/ZHU\\_Training\\_Page/clouds/planetary\\_boundary\\_layer/PBL.html](https://www.weather.gov/source/zhu/ZHU_Training_Page/clouds/planetary_boundary_layer/PBL.html) 19. May 2024.

“Simulated Historical Climate & Weather Data for Falconara Marittima.”  
Meteoblue,  
[www.meteoblue.com/en/weather/historyclimate/climatemodelled/falconara-marittima\\_italy\\_3177250](http://www.meteoblue.com/en/weather/historyclimate/climatemodelled/falconara-marittima_italy_3177250). Accessed 21 June 2024.

Wyngaard, John C. "Structure of the planetary boundary layer and implications for its modeling." *Journal of Applied Meteorology and Climatology* 24.11 (1985): 1131-1142.

EPA. User’s Guide for the AMS/EPA Regulatory Model (AERMOD). Oct. 2023.,  
[https://gaftp.epa.gov/Air/aqmg/SCRAM/models/preferred/aermod/aermod\\_userguide.pdf](https://gaftp.epa.gov/Air/aqmg/SCRAM/models/preferred/aermod/aermod_userguide.pdf), 16. May 2024.

Stockie, John M. "The mathematics of atmospheric dispersion modeling." *Siam Review* 53.2 (2011): 349-372.

Thompson, Roger S., and William H. Snyder. "Air pollution and terrain aerodynamics: a review of fluid modeling studies at the EPA fluid modeling facility." *Journal of wind engineering and industrial aerodynamics* 21.1 (1985): 1-19.

Venkatram, Akula, et al. "A complex terrain dispersion model for regulatory applications." *Atmospheric Environment* 35.24 (2001): 4211-4221.

Pinatubo Volcano Observatory Team. "Lessons from a major eruption: Mt. Pinatubo, Philippines." *Eos, Transactions American Geophysical Union* 72.49 (1991): 545-555.

Fileni, Lorenzo, et al. "Air pollution in Ancona harbour, Italy." *WIT Transactions on The Built Environment* 187 (2019): 199-208.

U.S. Environmental Protection Agency. *Guideline on Air Quality Models (40 CFR Part 51 Appendix W)*. 17 Jan. 2017, [www.epa.gov/scram/clean-air-act-permit-modeling-guidance](http://www.epa.gov/scram/clean-air-act-permit-modeling-guidance).

Chang, J.C., and S.R. Hanna. "Air Quality Model Performance Evaluation." *Meteorology and Atmospheric Physics*, vol. 87, no. 1, 2004, pp. 167-196.

European Commission. "Guidance Document on the Use of Models for the European Air Quality Directive." Joint Research Centre, 2011, [publications.jrc.ec.europa.eu/repository/handle/JRC64613](https://publications.jrc.ec.europa.eu/repository/handle/JRC64613).

"Regione Utile." Regione Marche, [www.regione.marche.it/Regione-Utile/Ambiente/Aree-ad-elevato-rischio-di-crisi-ambientale-AERCA](http://www.regione.marche.it/Regione-Utile/Ambiente/Aree-ad-elevato-rischio-di-crisi-ambientale-AERCA).

Accessed 11 Nov. 2024.

Flori, Massimo Marcelli. "Siti Di Interesse Nazionale." Siti Di Interesse Nazionale, 10 Feb. 2023, [www.arpa.marche.it/index.php/siti-di-interesse-nazionale](http://www.arpa.marche.it/index.php/siti-di-interesse-nazionale).

Etiopo, G. "EMEP/EEA air pollutant emission inventory guidebook 2009." (2009).

World Health Organization. *Air Quality Guidelines: Global Update 2005. Particulate Matter, Ozone, Nitrogen Dioxide and Sulfur Dioxide*. World Health Organization, 2006.