VIGIL 1.3.6: a tool for Automatized Probabilistic Volcanic Gas Dispersion Modelling

Fabio Dioguardi1,2 and Silvia Massaro1

1 University of Bari “Aldo Moro”, Dipartimento di Scienze della Terra e Geoambientali, Bari, Italy

2 British Geological Survey, The Lyell Centre, Edinburgh, United Kingdom

Contents

[Abstract 2](#_Toc111804005)

[1. Introduction 2](#_Toc111804006)

[2. The numerical models 3](#_Toc111804007)

[2.1 DISGAS: passive gas dispersal model 3](#_Toc111804008)

[2.2 TWODEE: dense gas dispersal model 4](#_Toc111804009)

[2.3 DIAGNO: Diagnostic Wind Model 4](#_Toc111804010)

[3. The program setup 4](#_Toc111804011)

[3.1 Dependencies and installation instructions 5](#_Toc111804012)

[3.2 Folders structure 6](#_Toc111804013)

[3.3 The VIGIL Input and Output files 7](#_Toc111804014)

[4. Running VIGIL 10](#_Toc111804015)

[4.1. weather.py 10](#_Toc111804016)

[4.2 run\_models.py 13](#_Toc111804017)

[4.3 post\_process.py 14](#_Toc111804018)

[5. Application examples 16](#_Toc111804019)

[5.1 Example 1: Solfatara volcano (Campi Flegrei, Italy) 16](#_Toc111804020)

[5.2 Example 2: La Soufrière de Guadeloupe (Lesser Antilles) 18](#_Toc111804021)

[5.3 Example 3: Mefite d’Ansanto area (Italy) 23](#_Toc111804022)

[6. Future perspectives 27](#_Toc111804023)

[Acknowledgments 27](#_Toc111804024)

[References 27](#_Toc111804025)

# Abstract

The VIGIL (automatic probabilistic VolcanIc Gas dIspersion modeLling) tool is a collection of Python scripts designed to aid the user with the usage of gas dispersion models (both passive gas dispersion and heavy gas flow) and to automatically perform probabilistic simulations. Two open-source Eulerian models are supported: DISGAS v2.5.1 (Costa et al., 2005; Costa and Macedonio, 2016) and TWODEE v2.6 (Hankin and Britter, 1999; Folch et al., 2009), which are coupled with a mass-consistent Diagnostic Wind Model (DIAGNO v1.2.3, Douglas et al., 1990). Both models are written in Fortran-90 while VIGIL is implemented through Python v3.8, an open-source, general-purpose, interpreted high-level programming language.

# 1. Introduction

Volcanic eruptions are very hazardous events which can have a devastating impact on human settlements worldwide. In the last decades, probabilistic volcanic hazard assessment methodologies have allowed the quantification of intrinsic uncertainties related to volcanic eruptions, which play a fundamental role in risk reduction strategies, making them of interest to both scientists and decision makers (e.g., Marzocchi et al., 2006; Selva et al., 2012a,b; Sandri et al., 2016). In this scenario, the study of gas dispersion is a key for the understanding the physic-chemical changes occurring in the magmatic feeding system or in the hydrothermal aquifer (i.e., Edmonds et al. 2018; Massaro et al., 2021), irrespective of magmatic and non-magmatic unrest. In last the decades, numerical modelling has investigated on the different behaviour of both heavy (e.g. Costa et al., 2008; Folch et al., 2009, 2017) and light volcanic gases (e.g., Costa et al., 2005; Chiodini et al., 2010, Granieri et al., 2013; Pedone et al., 2017) accounting for topographic effects, variation of atmospheric conditions and wind directions. For a gas denser than air, the flow behaviour over complex topography is generally described using depth-averaged variables (shallow layer approach, e.g., Hankin and Britter, 1999), while for dispersion of a diluted gas passively driven by wind advection and atmospheric turbulence, simpler advection-diffusion equations can be solved (e.g. Prabha and Mursch-Radlgruber, 1999).

Here we present the new version of VIGIL (v1.3.6), which builds on the previously presented v1.3 (Dioguardi et al. 2022) and is designed to control an automatic simulation routine and environment for DISGAS v2.5.1 (Costa et al., 2005; Costa and Macedonio, 2016) and TWODEE v2.6 (Hankin and Britter, 1999; Folch et al., 2009) coupled with a mass-consistent Diagnostic Wind Model (DIAGNO v1.2.3, Douglas et al., 1990). In the following we refer to these solvers as DISGAS, TWODEE and DIAGNO, respectively. By simplifying meteorological data collection and processing and automatizing the generation of input data, the running of the models (when possible in parallel) and the generation of outputs (also graphical), this tool can be used to provide a probabilistic volcanic gas modelling for volcanic hazard assessment purposes. In particular, DISGAS reproduces the passive gas dispersion into the atmosphere, governed by wind and atmospheric turbulence. The model calculates gas concentrations (e.g., CO2) expressed as values in excess of background gas species levels in the air at heights selected by the user in a terrain-following coordinate system. TWODEE simulates heavy gas flow in a calm ambient based on the shallow water equations for fluid depth, depth-averaged horizontal velocities and depth-averaged fluid density. The used shallow layer approach is a good compromise between the complexity of Computational Fluid Dynamics (CFD) models and the simpler integral models.

In the following we firstly describe the physical models on which the models are based, then we present the VIGIL setup, the folders structure, the Input/Output files and the instructions on how to install and run the code.

# 2. The numerical models

## 2.1 DISGAS: passive gas dispersal model

The open-source Eulerian model DISGAS (Costa et al., 2005; Costa and Macedonio, 2016) uses the passive dispersion approach for the dilute gases, solving the advection-diffusion equation in the transformed terrain-following coordinates system:

|  |  |
| --- | --- |
|  | (1) |

for the scaled concentration *C* (i.e. consistent with the transformation (1)) is:

|  |  |
| --- | --- |
|  | (2) |

Where *h* is the terrain elevation, *t* is the time, *U, V, W* are the scaled wind components along the *X*, *Y*, and *Z* directions, respectively, *Kh* and *Kz* are the diagonal scaled diffusion coefficients and *Q* is the source term. The advective terms in Eq. 1 are discretised according to the second-order Lax–Wendroff scheme (e.g., Lax and Wendroff, 1960; Ewing and Wang, 2001). Passive dispersion approach is used when the gas is diluted enough with respect to the surrounding atmosphere, and the validity of such an approximation can be assessed by estimating the Richardson number *Ri*:

|  |  |
| --- | --- |
|  | (3) |

Where *ρg* and *ρa* are the air and the gas densities, respectively, *q* is the volumetric flow rate, *R* the plume size (e.g., plume radius), and *v* is the wind velocity at the reference altitude (i.e. 10 m). For *Ri* < 0.25 transport is substantially passive whereas for *Ri* >1 is mainly density driven (Cortis and Oldenburg, 2009; Costa et al., 2013). According to this approach, gas transport is off-line coupled with the output of the wind field.

DISGAS allows to use two options for the wind field:

* “SIMILARITY”, assuming a horizontal wind profile calculated in agreement with the Monin-Obukhov similarity theory (e.g., Monin and Yaglom, 1979; Byun 1990), in which case the null divergency conditions cannot be guaranteed everywhere in the domain. It is more suitable over flat topography.
* “DIAGNO”, using the mass-consistent Diagnostic Wind Model (Douglas et al., 1990) for the wind field over a realistic topography model.

VIGIL is designed to use DISGAS with the “DIAGNO” option. However, it can still be run with the “SIMILARITY” option if the user modifies the DISGAS input file accordingly. The current version of the model does not account for the chemical reactivity of components and also neglects gas solubility in condensing H2O droplets. A full description of the physical model can be found in Costa and Macedonio (2016).

## 2.2 TWODEE: dense gas dispersal model

The open-source Eulerian TWODEE code solves a time-dependent model for the flow of a heavy gas based on the shallow layer approach. It is built on the depth-averaged equations for a gas cloud resulting from mixing a gas of density *ρg* with an ambient fluid (air) of density *ρa* (*ρg* > *ρa*). TWODEE is derived from the optimization and improvement of a previous Fortran-77 version developed by Hankin and Britter (1999). Under the assumption that *h/L* ≪ 1 (h being the gas cloud depth and L a characteristic length), the 2D shallow-layer approach allows a compromise between more realistic but computationally demanding 3D CFD models and simpler 1D integral models. Such an approach is able to describe the cloud in terms of four variables: cloud depth, two depth-averaged horizontal velocities, and depth-averaged cloud density as functions of time and position. Since real clouds do not have a definite upper surface it is necessary to define cloud depth in terms of the vertical concentration distribution (Folch et al., 2007; 2009). A full description of the physical model can be found in Folch et al. (2007, 2009). Also in this case the wind field can be evaluated by a uniform wind (SIMILARITY option) in accord to the Monin-Obukhov similarity theory (e.g., Monin and Yaglom, 1979; Byun 1990), or by a spatially variable wind which allows to incorporate terrain effects (DIAGNO option) through the DWM (Douglas et al., 1990). Additionally, the latest version of TWODEE allows the option “MET\_NC”, to use outputs of numerical weather prediction or reanalysis models in NetCDF format. This option is ideal for using high-resolution meteorology obtained by running a numerical weather prediction model.

## 2.3 DIAGNO: Diagnostic Wind Model

DIAGNO generates wind components (*u, v, w*) at several user-specified vertical levels for a specified time, incorporating local surface and upper-air wind observations, where available, and providing some information on terrain-induced air flows in regions where local observations are absent. In particular, the generation of the wind field is obtained by a two-step procedure. Step l produces a spatially varying gridded field of wind components (*u, v, w*) for each vertical layer within the modelling domain. Step 2 combines the gridded wind field generated in step l with available observational data to produce a final gridded wind field (*u, v, w*). This involves four sub-steps: (l) interpolation, (2) smoothing of the analysed field, (3) computation of a vertical velocity field, and (4) minimization of the three-dimensional divergence. Finally, a divergence-minimization procedure is iteratively applied until velocity divergence is smaller than an arbitrarily user defined threshold. The final result of DIAGNO is an approximately null-divergence wind field consistent with the observations. A complete description is provided in Douglas et al. (1990).

DIAGNO needs topography data, average wind on the computational domain, and atmospheric stability information within the scale of the domain (i.e. the temperature gradient). The approximation of a null-divergence wind field is generally applicable up to a kilometre above ground level (Dutton and Fichtl, 1969). Since we deal with dispersion in the surface layer, i.e. the lowest part of the Planetary Boundary Layer (which extents ca. up to 2-3 km in the atmosphere), this approximation is suitable to our study. Furthermore, since terrain-following coordinates are used, such an approximation still holds even if ground level varies significantly within the computational domain. Then, DISGAS and TWODEE models linearly interpolate the wind field into computational grid.

# 3. The program setup

VIGIL is composed of three scripts:

* **weather.py**: it prepares the weather data by either retrieving reanalysis data from the ECMWF ERA5 database (Copernicus Climate Change Service, 2017) or forecast data from the NOAA-NCEP Global Forecast System Numerical Weather Prediction model (<https://www.emc.ncep.noaa.gov/emc/pages/numerical_forecast_systems/gfs.php>) or from local weather stations provided by the user. The option ERA5 and weather station can be used simultaneously. The script is designed to randomly sample N days from a time interval defined by the user. If weather station data are available, the script is designed to extract weather data in the time interval specified by the user from selected weather data file. In all modes the system currently allows time intervals of up to 24 hours, which is due to the limitations of DIAGNO. Data are then organized in folders to be used by the meteorological processor DIAGNO.
* **run\_models.py**: it runs DIAGNO and successively DISGAS or TWODEE (hereafter referred to as DISGAS and TWODEE, respectively). The user can assess the gas emission sources in the computational domain by using random source locations (the source locations are selected from a probability map) or by fixed source locations (the source locations are read from a list containing the source coordinates). In both cases, the number of sources can be fixed or randomly sampled from a range defined by the user. In the same way, the associated gas fluxes can be read by a list (fixed source emission) or randomly sampled by an Empirical Cumulative Distribution Function (ECDF) (random source emission) built by the code using data provided by the user in the “flux.txt” file (see below). A combination of fixed and random emissions is also possible. This script is also optionally interfaced with Slurm Workload Manager to run DIAGNO, DISGAS and TWODEE on a cluster using the available resources with Slurm. In the latest version, an automatic scenario detection has been introduced, i.e., VIGIL calculates *Ri* (eq. 3) for each source of the domain and determines whether to run DISGAS or TWODEE for each of the source. In situations where some sources require DISGAS and other TWODEE, VIGIL run a split simulation and then merge the outputs (by summing up the gas concentrations in the computational domain) to produce a single output.
* **post\_process.py**: it reads the DISGAS or TWODEE outputs produced by run\_models.py and converts the model outputs in concentration of other gas species (e.g., CO2, H2S) based on the gas species properties made available by the user in the file gas\_properties.csv. The script can also calculate and plot the modelled concentrations at the user’s specified exceedance probabilities, timesteps and vertical layers. It can also calculate the time averaged concentrations and the persistence probability, i.e. the probability to overcome gas specie-specific concentration thresholds for specified exposure times. The user can specify the format of the processed output files; the current version only support ASCII GRD file, but other options will be introduced in the near future.

## 3.1 Dependencies and installation instructions

VIGIL assumes that the binary files of DIAGNO (diagno), DISGAS (disgas) and TWODEE (twodee) are in the system path. The latest versions of DIAGNO, DISGAS and TWODEE can be downloaded from the following repositories:

DIAGNO v1.2.3: <http://datasim.ov.ingv.it/models/diagno.html>

DISGAS v2.5.1: <http://datasim.ov.ingv.it/models/disgas.html>

TWODEE v2.6: <http://datasim.ov.ingv.it/models/twodee.html>

We refer to the user manuals of the three distributions for detailed instructions on how to install and run these tools separately.

Additionally, the following software are required before running VIGIL:

* **wgrib2**: <http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2/>. VIGIL assumes the executable is in the system PATH.
* **grib-tools**: for Windows, it is used chocolatey to install it. https://chocolatey.org/packages/grib-tools; for Linux, it is required the installation of “eccodes”. In both cases, it is required to add the binaries folder to the system PATH.
* **CDSAPI client key**: it is needed to download the ERA5 reanalysis data. The user has to register at: [https://cds.climate.copernicus.eu/cdsapp#!/home](https://cds.climate.copernicus.eu/cdsapp%23!/home). Once the registration is approved, to get the data follow the instructions at: <https://confluence.ecmwf.int/display/CKB/How+to+download+ERA5>. The user needs to install the personal key in a .cdsapirc file, to save in different locations depending on the OS.
* Additional Python modules needed are: cdsapi, matplotlib, openpyxl, pandas, pathos, utm, scipy.

It is possible to set a virtual environment with all the required dependencies specific for VIGIL by using Anaconda (<https://www.anaconda.com/>). This simplifies the installation of the different packages and the management of the Python installation in the system. An **environment.yml** file, which contains the name of the environment (“vigil” is the default) and the required modules, is available in the distribution in order to set the Conda environment quickly and easily. For setting the Conda environment, the user can use the following instructions:

1. create the environment with the needed additional packages:

conda env create -f environment.yml

1. activate the environment with:

conda activate vigil

(assuming the name “vigil” is not chanced in environment.yml)

1. to exit from the environment:

conda deactivate

## 3.2 The VIGIL Input and Output files

VIGIL requires the following mandatory files to be stored in the working folder where the Python scripts are before running:

* **diagno.inp**: it denotes the DIAGNO input file. This file has a fixed structure, which is explained in the DIAGNO user manual available in the DISGAS package. Working versions can be found in the Example folders: the user can use any working version of the diagno.inp file from the example without modifying it. weather.py modifies all the necessary entries based on the simulation setup. It is essential that the user does not alter the structure of the file manually.
* **“problemname”.inp**: it is the control file of DISGAS and/or TWODEE. “problemname” should be “disgas” or “twodee” (or both) depending on the model that has to be run. We refer the user to the DISGAS and TWODEE user manuals for details on the structure of these files. Similar to diagno.inp, the user can take any of the files in the Example folders; run\_models.py will take care of modifying these files accordingly. NOTE: it is recommended to avoid and/or delete any comments in the input files as to avoid issues when run\_models.py modifies them. Future versions of VIGIL will use built-in templates of diagno.inp, disgas.inp and twodee.inp.
* **topography.grd**: a file that describes the topography in ASCII Surfer v6 GRD format (<http://surferhelp.goldensoftware.com/topics/ascii_grid_file_format.htm>).
* **roughness.grd**: a file that describes the roughness length in ASCII Surfer v6 GRD format (<http://surferhelp.goldensoftware.com/topics/ascii_grid_file_format.htm>). In DISGAS it is possible to set a uniform value in the disgas.inp file, in which case roughness.grd is not required.

Details on the structure of these ASCII GRID files are available in the DISGAS and TWODEE user manuals.

Other optional files are required depending on the options chosen by the user. When weather data from a local weather stations are used, these should be stored into a “**weather\_stations**” folder in separate files for each weather station. In this case, the following input files are required:

* **weather\_stations\_list.txt**: it contains the list of files with the weather station data and that are stored in the folder “weather\_stations”. Mandatory only when the user run DIAGNO with local meteorological data. The format of this file is in the following:

NSTATIONS

LATITUDE LONGITUDE ELEVATION FILE\_NAME

(...)

Where NSTATIONS is the number of station files, LATITUDE and LONGITUDE are the geo-graphical coordinates, ELEVATION is the elevation of the weather station in m above the ground, FILE\_NAME is the weather station file name. These fields are tab formatted.

* **“file\_name”.txt**: it contains the meteorological data acquired by a weather station. The file has to be included in the “weather\_station” folder. Mandatory only when the user run DIAGNO with local meterological data (-WST True, see section 4.1. weather.py). The file is organized as in the following:

yyyymmdd-HH:MM,T(°C),Wind dir (°N),Wind speed(km/h),P(hPa),Tsoil(°C)

20160416-00:05,14.186667,41.819018,5.031000,857.216500,NaN

20160416-01:05,13.893333,45.242331,4.203500,856.367333,NaN

(…)

It is a comma-separated file, with each line representing a time record of the data with a 1-hour time resolution. The record Tsoil (soil temperature) should be set to NaN when not available, in which case weather.py set it equal to T, i.e. the temperature recorded by the station. When running a continuous simulation (-CS True, see section see section 4.1. weather.py), the file should include the weather data for the whole time interval defined by the start and end day of the continuous simulation interval.

Further optional files can be provided to control the emission sources. These are:

* **flux.txt**: it contains a list (in a column) of possible values of the gas source emission rate (kg s-1) that VIGIL uses to build an ECDF from which it randomly selects the number of possible source emission rates requested by the user.
* **sources\_input.txt**: a file that contains a list of fixed sources provided by the user. It is structured as in the following:

x y z prob kg/s T dx dy dur

427225 4519625 0 1 0.430844907 293 1 1 32600

(…)

Where x, y and z are the UTM coordinates (easting, northing) and elevation above the ground in m of the center of the source, respectively, prob is the probability of the emission (which in the current version of VIGIL does not play any role in the computation), kg/s is the gas emission rate, T is the gas temperature at the source in K, dx and dy are the source extension in m along the x and y direction and dur is the source duration in s. It is to note that:

* only TWODEE allows defining a gas source over a rectangular area, whilst DISGAS only allows point source; therefore dx and dy are not actually used by DISGAS. However, if the user sets a source whose area is larger than the cell size in the computational domain, if DISGAS is used then VIGIL redistribute the source gas flux across the cells that are within the source area. Therefore, dx and dy should be set to a value less than the cell size if the user wants to define a point source.
* the gas temperature of the source is also specified in **gas\_properties.csv**. However, the latter is used to initialize the gas density that is needed by TWODEE, therefore if multiple sources are present in the domain, then the single value in **gas\_properties.csv** should represent an average value of the temperature of all the source in the domain. In the case of the source\_input.txt file, each source can have a different temperature and each value is used to calculate the source Richardson number (eq. 3) in case of an automatic scenario (see section 4.2).
* **probability\_map.grd:** an ASCII Surfer v6 GRD format (<http://surferhelp.goldensoftware.com/topics/ascii_grid_file_format.htm>) file that contains the best guess probability of a gas source location, for each cell of the domain. It is a structured as a matrix with NY rows and NX columns, where NX and NY are the number of cells in the domain along the x and y directions, respectively. NX and NY must coincide with the values indicated in “problemname”.inp.

The file **gas\_properties.csv** contains geochemical data of the gas species in order to:

* convert the concentration of the gas specie tracked by TWODEE or DISGAS into the concentration of another gas specie or when the conversion from ppm (the default gas concentration units of DISGAS and TWODEE) to kg m-3 is required. Specifically, the conversion factor used to convert the gas specie from kg m-3 to ppm is calculated as follow:

|  |  |
| --- | --- |
|  | (4) |

where is the air temperature at 2 m from the ground in K and the pressure at 2 m from the ground in hPa. The molar weight of the gas specie is expressed in g mol-1. This can be used, e.g., when an abundant gas specie (e.g. H2O) is set as the gas specie in DISGAS and TWODEE and used as a tracer for less abundant species (e.g. CO2) (Tamburello et al. 2019, Massaro et al., 2021). For this conversion, two parameters are needed: the molar ratio between the species (converted / original: note, it is mandatory that the original specie coincides with the tracked specie specified by the user with the command -TS –tracking\_specie; see Section 4) and the molar weight of the new specie in g mol-1.

* optionally set up a value of the background concentration of the gas specie in the atmosphere. Note that TWODEE allows setting this value up in twodee.inp via the command “CONCENTRATION\_BG” (in ppm), but in case multiple gas species are tracked with VIGIL via the conversion procedure described above, this approach could lead to errors in the final value of the concentrations. For example, if one sets a background concentration for CO2 of 350 ppm in the twodee.inp file, TWODEE will apply this background concentration to the whole computational domain in their output files. When these are processed by VIGIL in the post processing stage, if more than one gas specie is considered (e.g., for converting from one specie to another), this background would be equally applied to any gas specie causing an error. It is therefore recommended to always set “CONCENTRATION\_BG” to 0 ppm in twodee.inp to avoid possible errors, particularly when converting from a gas specie to another, and also for consistency with DISGAS, which currently does not have an analogous command.
* provide the specific gas constant of the gas species *R* [J kg-1 K-1] and temperature *T* [K], which are used by VIGIL to calculate the gas density at the source in order to estimate the source Richardson number, which in turn is needed to set the scenario automatically (light gas with DISGAS or dense gas with TWODEE) if required by the user (eq. 3). Note that the temperature provided in **gas\_properties.csv** is only used if the random sources are activated, otherwise the temperature of each source is read from the **sources\_input.txt** file.
* provide concentration thresholds and exposure times for each species in order to calculate the persistence of the gas specie, i.e., the probability to overcome a gas concentration threshold over a specified period of time, is calculated.

This comma separated file is structured as in the following:

CO2/H2O,H2S/H2O,CT\_CO2,ET\_CO2,CT\_H2S,ET\_H2S,M\_CO2,M\_H2S,M\_H2O,BG\_CO2,T\_CO2,R\_CO2,R\_H2S

0.011,0.0115,1000,24,0.05,24,44.01,34.1,18.015,400,300,188.92,243.96

0.0111,0.0064,3500,24,0.13,1

0.0121,0.0064,5000,8,0.2,1

0.0123,0.0056,15000,1,5,1

0.0128,0.0063,30000,1,100,1

0.0143,0.0062,100000,1

0.0143,0.0064

0.015,0.0121

0.0151,0.0066

0.0153,0.0063

0.0153,0.0153

(…)

Where the columns with headers with “/” define the molar ratio. the columns with headers “M\_” + gas specie (e.g., “M\_CO2”) identify the molar weights of the species in g mol-1. Optional columns for the gas species should be provided for the gas persistence calculations: concentration thresholds (the units depends on the user choice provided in input via -U --units), which should be identified with the header “CT\_” + gas specie (e.g., “CT\_H2S”), and exposure time (in hours), which should be identified with the header “ET\_” + gas specie (e.g., “ET\_CO2”). More than one value can be provided in these columns, in which case post\_process.py link the values of concentration thresholds to exposure times row-by-row. The headers “BG\_”, “R\_”, “T\_” + gas specie (e.g., “BG\_CO2”) are used to provide the value of the background concentration in the atmosphere in ppm, specific gas constant and temperature of the gas specie, respectively.

It is to note that:

1. it is possible to provide more than one specie, as in the example above. If more than one specie is provided, all the columns with the molar ratios should be aligned to the left of the columns with the species molar weights;
2. It is possible to provide more than one value of molar ratio, which is suggested since this parameter is affected by a significant uncertainty. post\_process.py randomly select one value of the molar ratio to be used for the conversion from an ECDF built by the code based on the data provided in the gas\_properties.csv file.
3. Parameters that can have multiple values and are linked (e.g., concentration thresholds and exposure times) for each specie should be aligned one next to each other as in the example above.
4. When a tracking specie is identified, the user should provide the molar weight of this specie too.
5. The columns with concentration thresholds and exposure times for the persistence calculations are optional. If the user activates the persistence calculation and these data are not provided in the file **gas\_properties.csv**, the code simply ignores the request.
6. In the persistence calculation, VIGIL assumes that the gas concentration at each output time of TWODEE or DISGAS represents a constant concentration in the time between the previous and the current output time, i.e. if the gas dispersion simulation is setup to produce an output every 6 hours, VIGIL assumes that the concentration in these 6 hours is equal to the gas concentration at the output time under analysis. This implies that, if the exposure time for a specific concentration threshold is 1 hour and the simulations produce outputs every 6 hours, VIGIL assumes that the concentration overcomes the threshold for at least 1 hour in the 6 hours before the current output time under analysis. This approach is clearly an assumption and work is underway to improve it.

Finally, the user can provide the optional file **tracking\_points.txt** that is used by post.process.py if the option -TP --tracking\_points is activated (see Section 4.3 post\_process.py). The file contains a list of locations where the concentration fields resulting from the simulations are interpolated. It is structured as in the example below:

x y z

696945 1474693 0

696945 1474693 2

696945 1474693 10

696945 1474693 80

where x, y and z are the easting (or longitude), northing (or latitude) and elevation (m above the ground) of the tracking point. The code ignores any line in which the coordinates or the elevation do not represent realistic value and is able to recognize UTM or geographic coordinates. If tracking points are activated, the interpolated concentration at each time step is saved in an output .txt file for each simulation; these outputs are then used to create hazard curves (ECDF vs. concentration) for the location at each time step.

# 4. Running VIGIL

The workflow from weather data processing to post processing model outputs is shown in Fig. 1 and consist in running the Python scripts in the following sequence: weather.py, run\_models.py, post\_process.py. Each of these scripts has several options that can be selected by the user to activate or deactivate the different functionalities available in VIGIL. The options can be reviewed in the README.md file provided in the package and at any time in the command line by the command:

python “script” --help

where “script” is the name of the Python script.

Practical examples on the usage of VIGIL will be discussed in the next Section.

## 4.1. weather.py

This script is run to retrieve weather data from ERA5 archive in reanalysis mode or GFS forecasts in forecast mode or to process data of weather stations provided by the user as explained in Section 3.3. It also prepares the folder “simulations”, which stores all simulation input and output files, together with the original weather data files (Fig. 1). The possible arguments of this scripts are:

weather.py [-h] [-M MODE] [-RT RUN\_TYPE] [-CS CONTINUOUS\_SIMULATION] [-S START\_DATE] [-E END\_DATE] [-SY SAMPLED\_YEARS] [-SM SAMPLED\_MONTHS] [-SD SAMPLED\_DAYS] [-V VOLC] [-LAT LAT] [-LON LON] [-EL ELEV] [-NS SAMPLES] [-ERA5 ERA5] [-WST STATION] [-N NPROC] [-TD TWODEE] [-DG DISGAS]

* -h, --help

Command to show the guide to this script.

* -M MODE, --mode MODE

Possible options: reanalysis, forecast. If reanalysis, either ERA5 or WST options should be

on. If forecast, GFS data will be downloaded and processed

* -RT RUN\_TYPE, --run\_type RUN\_TYPE

Possible options: new, restart. This prepares the working folders in the proper way for run\_models.py, depending on the choice.

* -CS CONTINUOUS\_SIMULATION, --continuous\_simulation CONTINUOUS\_SIMULATION

Specify if the simulation is continuous between the specified start and end dates. Possible options are True or False. If True, weather data are sampled from a continuous interval defined by the start and end date and the number of samples (-NS) defined by the user is overwritten by the time difference in days between the end and start date. This option is possible in both forecast mode (except with weather station data) and reanalysis mode (both with model and weather station data).

* -S START\_DATE, --start\_date START\_DATE

Start date of the sampling period of the weather data. Format: DD/MM/YYYY

* -E END\_DATE, --end\_date END\_DATE

Start date of the sampling period of the weather data. Format: DD/MM/YYYY

* -SY SAMPLED\_YEARS, --sampled\_years SAMPLED\_YEARS

Optional comma-separated list of years to be sampled from the time interval defined by the start and end date.

* -SM SAMPLED\_MONTHS, --sampled\_months SAMPLED\_MONTHS

Optional comma-separated list of months to be sampled from the time interval defined by the start and end date.

* -SD SAMPLED\_DAYS, --sampled\_days SAMPLED\_DAYS

Optional comma-separated list of days to be sampled from the time interval defined by the start and end date.

* -V VOLC, --volc VOLC

This is the volcano ID based on the Smithsonian Institute IDs and is used to retrieve the coordinates and the elevation of the volcano location (if the simulation refers to a volcano) from the [BGS Eruption source parameters database](https://webapps.bgs.ac.uk/research/volcanoes/esp/search.cfc?method=viewHome).

-LAT LAT, --lat LAT

Volcano (or location) latitude, to be provided if the option -V VOLC, --volc VOLC is not used.

-LON LON, --lon LON

Volcano (or location) longitude, to be provided if the option -V VOLC, --volc VOLC is not used.

* -EL ELEV, --elev ELEV

Volcano (or location) elevation (m a.s.l.), to be provided if the option -V VOLC, --volc VOLC is not used.

* -NS SAMPLES, --samples SAMPLES

Number of days to randomly sample from the interval defined by the Start and End dates. The default number is 1.

* -ERA5 ERA5, --ERA5 ERA5

True: Use ERA5 reanalysis. False: Do not use ERA5 reanalysis. The default option if False.

* -WST STATION, --station STATION

True: Use weather station data. False: Do not use weather station data. The default option if False.

* -N NPROC, --nproc NPROC

Maximum number of allowed simultaneous processes, for parallel processing.

* -TD TWODEE, --twodee TWODEE

Option to activate the preparation of additional files for TWODEE. It can be “on” or “off”, the default option is “off”.

* -DG DISGAS, --disgas DISGAS

“on” or “off”, to prepare files for DISGAS

Additionally, weather.py creates the file “surface\_data.txt” which is needed by TWODEE when this is run with weather data from DIAGNO and the post\_process.py when conversion between gas species is requested. This file is a tab-separated file structured as in the following:

Time [HHMM] Tref[K] Tz0[K] P[Pa]

0000 273.292 273.652 94001.1

0100 273.451 273.99 94014.3

(…) (…) (…) (…)

Where Time [HHMM] defines the starting time of validity of the data in the row (the ending time of validity is the time of the following row), Tref[K] is the air temperature at the reference height specified in the twodee.inp file, Tz0[K] is the soil temperature and P[Pa] is the atmospheric pressure. Note that:

1. the reference height should be set to 2 m a.s.l. in twodee.inp when using ERA5 data; otherwise it is the height of the temperature sensor of the weather station, if weather station data are used;
2. Tz0[K] is automatically set equal to Tref[K] when this is not available. In ERA5 it is always available.

This file, together with all the other weather data files (original and processed) are temporarily stored in the folder “simulations/YYYYMMDD” where “YYYYMMDD” is the folder name of a single day of simulation (e.g. 20100524). When running with ERA5 data, the following files are generated:

* pressure\_levels.grib and surface.grib. These are the original GRIB1 files retrieved from the ERA5 archive, the former containing the data (temperature and wind) at different pressure levels (altitude), the latter containing the data at the surface (temperature, wind, soil temperature and pressure) for an area surrounding the location specified by the user.
* weather\_data\_YYYYMMDD and weather\_data\_sl\_YYYYMMDD: the GRIB2 versions of pressure\_levels.grib and surface.grib, respectively
* profile\_YYYYMMDDHH.txt and data\_location\_YYYYMMDDHH.txt are the GRIB2 data at different pressure levels and surface level interpolated at the exact location specified by the user.
* profile\_data\_YYYYMMDDHH.txt and data\_location\_data\_YYYYMMDDHH.txt are the same data as in profile\_YYYYMMDDHH.txt and data\_location\_YYYYMMDDHH.txt formatted in a user-friendly format.
* presfc.dat, preupr.dat, diagno.inp: input files for DIAGNO, edited by weather.py based on the processed weather data.
* topography.grd, a copy of the file originally stored in the working folder that will be used by DIAGNO.

If running with weather station data, all the files other than the surface ones will be missing.

It is also important to stress that, if the user does not use DIAGNO to calculate the wind field but UNIFORM option, the air temperature has to be set in Kelvin.

## 4.2 run\_models.py

This script is used to control the simulations of DIAGNO, DISGAS and TWODEE. DIAGNO is always run by run\_models.py, while the user can choose whether to run DISGAS or TWODEE or both. The possible arguments of this scripts are:

run\_models.py [-h] [-N NPROC] [-RT RUN\_TYPE] [-CS CONTINUOUS\_SIMULATION] [-RS RANDOM\_SOURCES] [-NS NSOURCES] [-SINT SOURCES\_INTERVAL] [-SLOC SOURCE\_LOCATION] [-SDX SOURCE\_DX] [-SDY SOURCE\_DY] [-SDUR SOURCE\_DUR] [-D DOMAIN] [-NX NX] [-NY NY] [-DX DX] [-DY DY] [-SEM SOURCE\_EMISSION] [-RER RANDOM\_EMISSION] [-RD RUN\_DURATION] [-OI OUTPUT\_INTERVAL] [-OH OUTPUT\_HEIGHTS] [-DI DIAGNO] [-DM DISPERSION\_MODEL] [-US USE\_SLURM] [-SP SLURM\_PARTITION] [-TS TRACKING\_SPECIE]

* -h, --help

Command to show the guide to this script.

* -N NPROC, --nproc NPROC

Maximum number of allowed simultaneous processes, for parallel processing. If the continuous simulation option is activated, this number is automatically re-set to 1 in order to allow processing days sequentially.

* -RT RUN\_TYPE, --run\_type RUN\_TYPE

Possible options: new, restart. If the run is of type “restart”, the simulations are restarted at the current day of execution starting from the outputs of the previous day.

* -CS CONTINUOUS\_SIMULATION, --continuous\_simulation CONTINUOUS\_SIMULATION

Specify if the simulation is continuous between the start and end dates specified via the weather.py script. Possible options are True or False. If True, post\_process.py runs the simulations specified in the days\_list.txt file sequentially.

* -RS RANDOM\_SOURCES, --random\_sources RANDOM\_SOURCES

“on”: randomly select NS locations from a probability map. “off”: fixed source locations. The probability map file is probability\_map.grd discussed above.

* -NS NSOURCES, --nsources NSOURCES

It specifies a number for a fixed number of sources. If “random” is specified, then the script randomly selects the number of sources from an interval defined with -SINT –sources\_interval

* -SINT SOURCES\_INTERVAL [SOURCES\_INTERVAL ...], --sources\_interval SOURCES\_INTERVAL [SOURCES\_INTERVAL ...]

This command can be used to specify the minimum and maximum number (comma-separated) of possible sources extracted by the probability map.

* -SLOC SOURCE\_LOCATION [SOURCE\_LOCATION ...], --source\_location SOURCE\_LOCATION [SOURCE\_LOCATION ...]

If one single source is needed, the user can specify its location with this command, which includes the coordinate type (UTM/GEO), latitude/northing, longitude/easting, elevation (above ground in m) sequentially. All entries should be comma-separated

* -SDX SOURCE\_DX, --source\_dx SOURCE\_DX

Extension [m] along the X direction of the single source specified above. Option valid for TWODEE only.

* -SDY SOURCE\_DY, --source\_dy SOURCE\_DY

Extension [m] along the Y direction of the single source specified above. Option valid for TWODEE only

* -SDUR SOURCE\_DUR, --source\_dur SOURCE\_DUR

Emission duration [s] of the single source specified above. Option valid for TWODEE only

* -D DOMAIN [DOMAIN ...], --domain DOMAIN [DOMAIN ...]

Command to define the computational domain. The following information should be specified sequentially as a comma-separated list: coordinates type (UTM/GEO), coordinates (latitude/northing, longitude/easting) of the bottom left corner and top right corner of the domain

* -NX NX, --nx NX

Number of grid cells along the x-direction. If not provided, the grid spacing along the x-direction must be provided.

* -NY NY, --ny NY

Number of grid cells along the y-direction. If not provided, the grid spacing along the y-direction must be provided.

* -DX DX, --dx DX

Grid spacing (in m) along the x-direction. If not provided, the number of grid cells along the x-direction must be provided.

* -DY DY, --dy DY

Grid spacing (in m) along the y-direction. If not provided, the number of grid cells along the y-direction must be provided.

* -SEM SOURCE\_EMISSION, --source\_emission SOURCE\_EMISSION

Source emission rate [kg/s]. If specified, it is assigned to all the sources in the domain

* -RER RANDOM\_EMISSION, --random\_emission RANDOM\_EMISSION

“on”: randomly assign the emission rate for each source in the domain sampled from a flux.txt file. “off”: use specified emission rate

* RD RUN\_DURATION, --run\_duration RUN\_DURATION

Run duration in hours. In the current version, fractions of hours or duration > 24 hours are not allowed, for longer runs the user should make use of the continuous simulation capability (-CS --continuous\_simulation). The default run duration is 24 hours.

* -OI OUTPUT\_INTERVAL, --output\_interval OUTPUT\_INTERVAL

Output interval in hours. In the current version, fractions of hours are not allowed. The default output interval is 1 hour.

* -OH OUTPUT\_HEIGHTS, --output\_heights OUTPUT\_HEIGHTS

Comma-separated list of output heights in m above the ground. If not provided by the users, the heights 0.0 and 10.0 m are added by default since these are mandatory in DISGAS and TWODEE.

* -DI DIAGNO, --diagno DIAGNO

Option to activate or deactivate DIAGNO. It can be “on” or “off”, the default option is “on”. It can be useful, e.g., to re-run simulations without going through the meteorological data retrieval and processing with DIAGNO.

* -DM DISPERSION\_MODEL, --dispersion\_model DISPERSION\_MODEL

Command to specify which dispersion model to use. Possible options are: twodee, disgas, automatic or none. If “automatic” is chosen, VIGIL automatically determines which model to use for each source in the computational domain based on its Richardson number via eq. 3. If “none” is specified and DIAGNO is activated, then the code runs only DIAGNO.

* -US USE\_SLURM, --use\_slurm USE\_SLURM

Option to run the simulation tools via Slurm Workload Manager if available on the system. The slurm commands are launched from within the Python scripts and the available resources are allocated based on the available nodes on the selected partition (see -SP --slurm\_partition option) and the number of simultaneous processes (via the -N --nproc command).

* -SP SLURM\_PARTITION, --slurm\_partition SLURM\_PARTITION

Name of the cluster partition where to run the simulations managed by Slurm.

* -TS TRACKING\_SPECIE, --tracking\_specie TRACKING\_SPECIE

The original emitted specie that is tracked in the dispersion simulation.

## 4.3 post\_process.py

This script is used to control the post processing of the outputs of the DISGAS and TWODEE simulations. For example, it can be used to produce probabilistic outputs like the output at a user-specified exceedance probability, convert gas species, create plots, etc.. All the processed outputs are stored in the folder post\_processing. Note that, if this folder is already present in the working directory, this will be overwritten.

All the options can be controlled via the arguments listed below:

post\_process.py [-h] [-h] [-P PLOT] [-ECDF CALCULATE\_ECDF] [-PER PERSISTENCE] [-EX EX\_PROB] [-T TIME\_STEPS] [-L LEVELS] [-D DAYS\_PLOT] [-C CONVERT] [-S SPECIES] [-TS TRACKING\_SPECIE] [-N NPROC] [-U UNITS] [-PL PLOT\_LIMITS] [-PI PLOT\_ISOLINES] [-TA TIME\_AV] [-OF OUTPUT\_FORMAT] [-PT PLOT\_TOPOGRAPHY] [-TI TOPOGRAPHY\_ISOLINES] [-PR PLOT\_RESOLUTION] [-TP TRACKING\_POINTS]

Command to show the guide to this script

* -P PLOT, --plot PLOT

“True”: Produce plots of the solutions and probabilistic outputs if activated. “False”: Do not produce plots. The default value is False.

* -ECDF, --calculate\_ecdf CALCULATE\_ECDF

“True”: Calculate the Empirical Cumulative Density Function (ECDF) of the solution and extrapolate solutions at user-defined exceedance probabilities. “False”: Do not calculate ECDF. It works only if -EX --ex\_prob is specified. The default value is False.

* -EX EX\_PROB [EX\_PROB ...], --ex\_prob EX\_PROB [EX\_PROB ...]

Comma-separated list of exceedance probabilities for probabilistic outputs.

* -PER PERSISTENCE, --persistence PERSISTENCE

If True, calculate the persistence of the gas specie, i.e., the probability to be exposed to a gas specie above specified concentration thresholds for times longer than the specified exposure times for those thresholds. Concentration thresholds and exposure times should be provided in gas\_properties.csv. If graphical outputs are requested (-P True), then persistence maps are generated.

* -T TIME\_STEPS [TIME\_STEPS ...], --time\_steps TIME\_STEPS [TIME\_STEPS ...]

Comma-separated list of time steps to plot (integer >= 0). Type “all” to plot all the time steps of the simulations.

* -L LEVELS [LEVELS ...], --levels LEVELS [LEVELS ...]

Comma-separated list of vertical levels (integer >= 1) to plot. Type “all” to plot all the levels. Note that, as stated in the previous subsection above, the level 0.0 m and 10.0 m are always added into the output heights, therefore the user should take this into account when setting the plotting levels via this commands.

* -D DAYS\_PLOT [DAYS\_PLOT ...], --days\_plot DAYS\_PLOT [DAYS\_PLOT ...]

Comma-separated list of days to plot (YYYYMMDD). Type “all” to plot all the days

* -C CONVERT, --convert CONVERT

If “True”, convert output concentration into other species listed with the command -S --species

* -S SPECIES [SPECIES ...], --species SPECIES [SPECIES ...]

Comma-separated list of gas species (e.g. CO2) into which to convert the original outputs.

* TS TRACKING\_SPECIE, --tracking\_specie TRACKING\_SPECIE

Name of the specie used as the tracking specie in DISGAS or TWODEE.

* -N NPROC, --nproc NPROC

Maximum number of allowed simultaneous processes, for parallel processing.

* -U UNITS, --units UNITS

Gas concentration units. Possible options are: “ppm”, “kg/m3”. Currently DISGAS automatically produces outputs in kg m-3 but these are converted in ppm by run\_models.py for consistency. Hence, only the conversion in kg m-3 is possible and if “ppm” is specified, no conversion is applied.

* -PL PLOT\_LIMITS [PLOT\_LIMITS ...], --plot\_limits PLOT\_LIMITS [PLOT\_LIMITS ...]

Minimum and maximum value of the concentration to display, specified as a comma-separated list. If unspecified, they are obtained from analyzing all the outputs.

* -PI PLOT\_ISOLINES, --plot\_isolines PLOT\_ISOLINES

List of gas concentrations values to be used to draw isolines. Optional

* -TA TIME\_AV, --time\_av TIME\_AV

Generate time-averaged outputs. Specify the time-averaging interval (in hours), or “0” for averaging over the whole duration

* -OF OUTPUT\_FORMAT, --output\_format OUTPUT\_FORMAT

Select format of the processed output files. Valid options are: GRD

* -PT PLOT\_TOPOGRAPHY, --plot\_topography PLOT\_TOPOGRAPHY

Plot an additional layer representing the topography elevation in the graphical outputs. This is based on the topography.grd file. Note that this add significant extra time in the post processing.

* -TI TOPOGRAPHY\_ISOLINES, --topography\_isolines TOPOGRAPHY\_ISOLINES

Topography height contour lines spatial resolution (in m a.s.l.). Used only if -PT True. The default value is 100 m.

* -PR PLOT\_RESOLUTION, --plot\_resolution PLOT\_RESOLUTION

Resolution (in dpi) of the picture. Default value is 600 dpi.

* -TP TRACKING\_POINTS, --tracking\_points TRACKING\_POINTS

“True”: Extrapolate gas concentration at locations specified in the file tracking\_points.txt

# 5. Application examples

In this section we discuss three application examples to show the possible applications of VIGIL to investigate on the space and time evolution of the dilute gas plumes and dense gas flows under different meteorological conditions in order to assess the potential hazard. The three examples are available in the package in the folder “Examples”, with input files and the commands used in this section.

In the first two examples, DISGAS is applied to simulate gas dispersion from the Solfatara volcano (Campi Flegrei, Italy) and La Soufrière de Guadeloupe volcano (Lesser Antilles). In the third example, TWODEE is applied to simulate the heavy gas emission from the Mefite d’Ansanto area (Campanian region, Italy).

All the examples can be run from the folders directly, the commands specify the relative path to the Python script without the need to copy them into the example folder.

## 5.1 Example 1: Solfatara volcano (Campi Flegrei, Italy)

Solfatara releases more than 1000 t d−1 of CO2 through soil diffuse degassing from a relatively small area (0.5-1 km2; Chiodini et al., 2001; Granieri et al., 2013). Temperature of the gases emitted from the Solfatara area is relatively high: fumaroles temperature is between 96 °C and 162 °C (Chiodini et al., 2001) and, flux weighted temperature of diffusing soil is 66 °C (Costa et al., 2005). This implies a density decrease due to temperature that almost balances the increase due to the greater molecular weight directly at the source (M[CO2]/M[air] = 44/29 while T[CO2]/T[air] ∼ 400/300). For these reasons, the passive dispersion assumption, is appropriate (Costa et al., 2005; Granieri et al., 2013).

Commands details can be found in the file **commands.txt** in the example\_1 folder. In this application we carry out the test by using the GFS forecast. The test case was run on the 16th July 2023, the user can specify a different day compatible with the time the test is run:

python ../../weather.py -S 16/07/2023 -E 16/07/2023 -V 211010 -NS 1 -M forecast -N 1 -DG on

The topography is represented by a 10 m-resolution DEM (Tarquini et al. 2007) and the resolution of the computational grid is set to 25 m. We run simulations with DISGAS using the fixed source locations (source\_input.txt) considering the CO2 diffusive contribution and some active fumaroles (Cardellini et al., 2017) in the W-NW sector of the computational domain. We directly consider the CO2 as the simulated plume component; hence no conversion is carried out in the post processing stage. The simulations were carried for 24 hours (-RD not specified, hence the default duration is 24 hours) with the following command:

python ../../run\_models.py -RS off -D UTM,4517000,426000,4522000,431000 -NX 100 -NY 100 -N 1 -DM disgas -TS CO2 -OH 0,0.5,1.5,5,15,30

Note the option -RS off that informs VIGIL to use fixed gas sources from the source\_input.txt file.

We finally carried out post-processing with the following command:

python ../../post\_process.py -P true -T all -L all -D all -C False -N 1 -U ppm -TS CO2 -S CO2 -PT True -TI 25 -PL 350,1000

In this case we decided to create graphical outputs of the results (-P true) for all time steps and vertical levels (-T all -L all). The outputs are shown in ppm (-U ppm -TS CO2 -S CO2), and the plot colour scale is bounded in the limits 350-1000 ppm (-PL 350,1000). Finally, all the plots should display the topography layer (-PT True).

Figure 2 shows some of these graphical outputs. In particular, it shows the concentration of CO2 (in ppm) at the last time step of the simulation (+6 hours from the emission start time, which coincides with the simulation start time) at four vertical layers from the ground: 0 m (Fig. 2a), 1.5 m (Fig. 2b), 5 m (Fig. 2c) and 30 m (Fig. 2d) above the ground m above the ground.

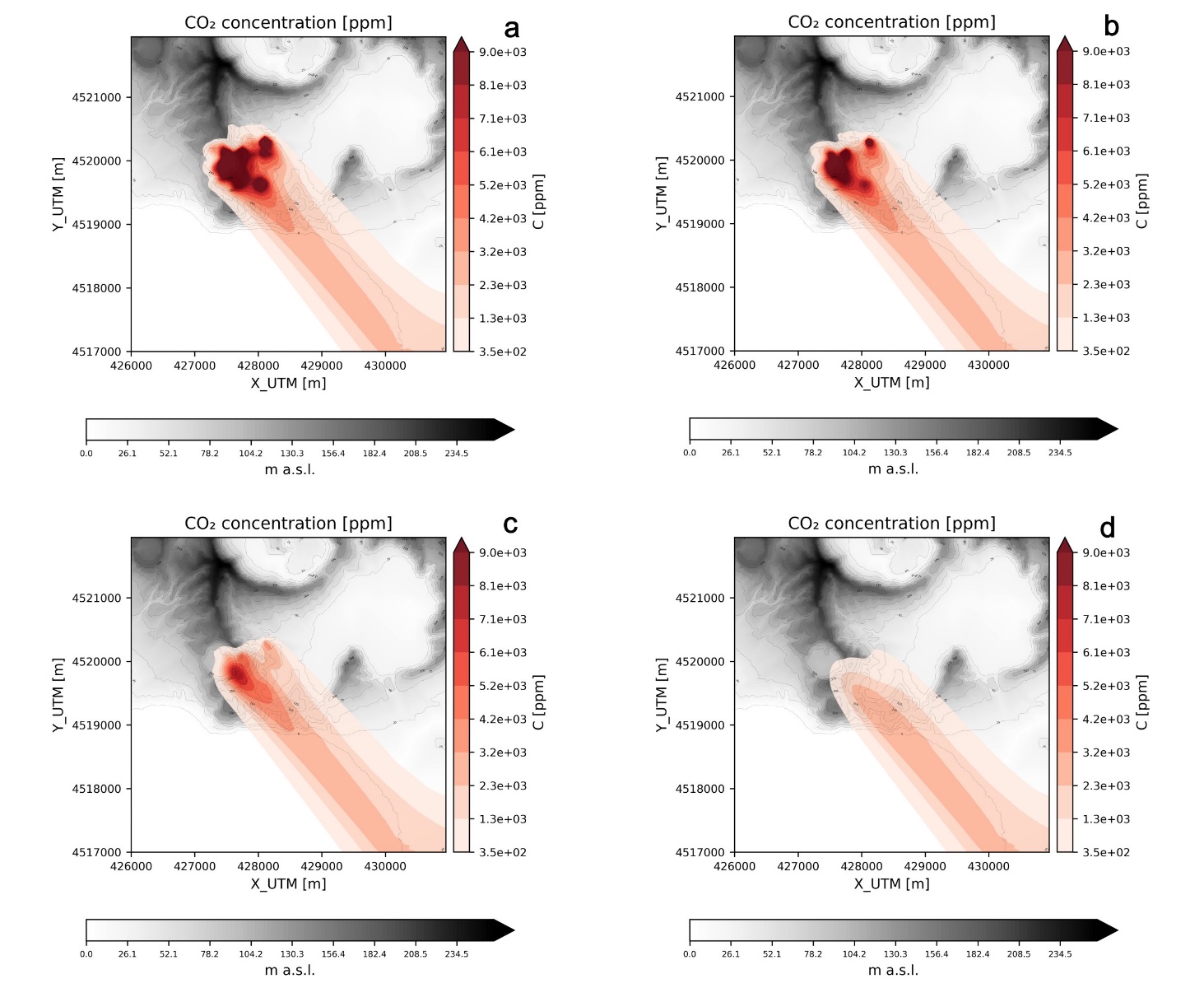


Figure 2. CO2 concentration (in ppm) 6 hours after the start of the emission at four different elevations above the ground. a) 0 m; b) 1.5 m; c) 5m; d) 30 m.

It can be seen how the concentration decreases with increasing elevation above the ground, being all the sources in this case at the ground level.

## 5.2 Example 2: La Soufrière de Guadeloupe (Lesser Antilles)

La Soufrière de Guadeloupe is currently the second most active emitter of volcanic gases in the Lesser Antilles island arc. The volcano has been in unrest since 1992 and represents the most important volcanic hydrothermal system of the region, whose present-day fumarolic activity is dominated by a passive dispersion regime. This passive condition at source is ensured since the fumaroles (mainly composed of H2O, CO2 and H2S) have a mean gas density of ca. 0.54 kg m-3 at T = 100°C, and P = 900 mbar, and the air density is ca. 1.03 kg m-3 (at T = 18 °C, P = 900 mbar) (Massaro et al., 2021).

In this application we carry out two tests by using the ERA5 reanalysis dataset (example\_2/2a) and local meteorological data (example\_2/2b) to obtain the wind field with DIAGNO. The commands that were used can be found in the **commands.txt** files in the two example folders.

The ERA5 weather data are downloaded and retrieved using the following command:

python ../../../ -S 01/01/1992 -E 01/01/2022 -V 360060 -NS 100 -ERA5 True -N 10 -DG on

Note that one hundred days (-NS 100) are randomly chosen from the period 01/01/1992 – 01/01/2022 (-S 01/01/1992 -E 01/01/2022) and requested to the Copernicus ERA5 database at groups of 10 requests per time (-N 10), since this is the limit currently set by the dataset. The volcano coordinates and summit elevation are retrieved from the BGS Eruption source parameters database by specifying the Smithsonian Institute ID of La Soufriére de Guadeloupe (-V 360060).

For the example 2b, the command to process weather data is:

python ../../../weather.py -S 16/04/2016 -E 16/04/2016 -V 360060 -NS 1 -WST True -N 1 -DG on

In this case, we select one day only (16/04/2016) specifying the same starting and end date and 1 day to sample (-S 16/04/2016 -E 16/04/2016 … -NS 1). For this day, data from one weather station ( located at 16.04 N, -61.66 W are available (-WST True) and listed in the **station\_1.txt** file stored in the folder **weather\_stations**.

In both tests, the topography is represented by a 5 m-resolution DEM (courtesy of Observatoire Volcanologique et Sismologique de la Guadeloupe - Institut de Physique du Globe de Paris) and the resolution of the computational grid was set to 15 m.

We consider the water vapour as the simulated plume component (tracking specie) since it prevails in all gas mixtures (Tamburello et al., 2019). The simulations are run with the following commands:

Case 2a: python ../../../run\_models.py -N 100 -RS on -NS 5 -SINT 2,5 -D UTM,1774000,642900,1775995,645390 -NX 166 -NY 133 -RER on -TS H2O -DM disgas -OH 0,1,2,4,10,20,30

Case 2b: python ../../../run\_models.py -N 1 -RS on -NS 5 -SINT 2,5 -D UTM,1774000,642900,1775995,645390 -NX 166 -NY 133 -RER on -TS H2O -DM disgas -OH 0,1,2,4,10,20,30

In the case 2a we run the 100 simulations for the 100 days in parallel (-N 100). We run simulations for both cases varying the daily value of the water vapour flux for each source, in order to catch the natural variability of the gas emission rate, and recording the hourly gas concentration at tracking points specified in the disgas.inp file. To do so, the values of the water vapour fluxes are randomly sampled from the values listed in flux.txt, which was built based on the data available from the literature on this observable (Massaro et al., 2021 and references therein), for five random sources (-RS on -NS 5 … -RER on). The locations of the gas sources were randomly sampled by the best guess **probability\_map.grd** file, which was created on the past literature data on the main geological structures, historical eruptive vents, past observed fumarolic activity and measurements of the present-day gas emission rates (from EUROVOLC WP12.2 deliverable, 2020 and references therein). Water vapour is the tracked specie in the dispersion simulations (-TS H2O). Figure 3 shows the probability map for this case.

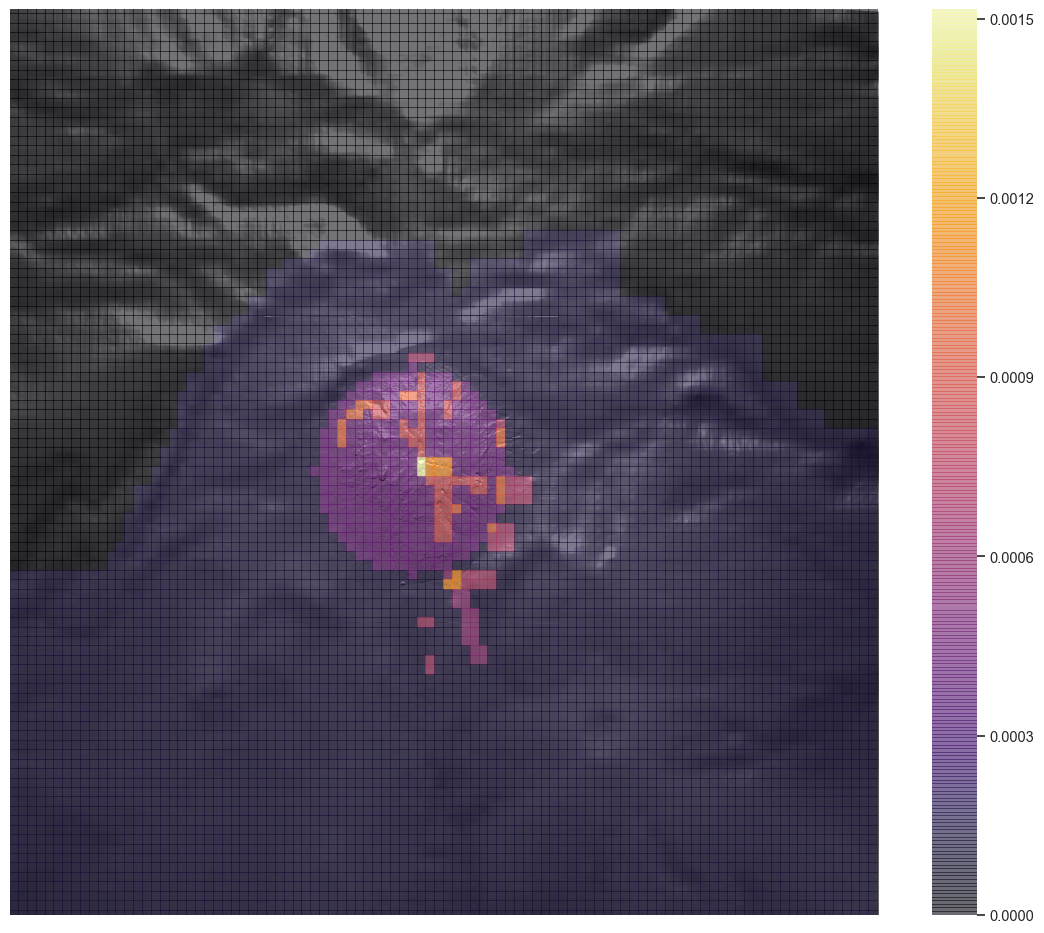
****

Figure 3. Best guess probability map for gas source opening (from EUROVOLC WP12.2 deliverable, 2020)

The post processing for the case 2a is carried out with the command:

python ../../../post\_process.py -P true -ECDF True -EX 0.5,0.05,0.01 -PER True -T 3 -L 3 -D all -C True -TS H2O -S H2S -N 100 -U ppm -PT True -TP True -PL 0.1,100

In this stage, the simulated water vapour concentration is converted into H2S concentration (-C True -TS H20 -S H2S) and displayed in ppm (-U ppm). Probability maps at exceedance probabilities of 50%, 5% and 1% are also requested (-ECDF True -EX 0.5,0.05,0.01). All the probabilistic and graphical outputs are produced for the third time step (-T 3) (+3 hours from the beginning of the simulation and emission since the time step for the output is left unspecified in the run\_models.py arguments and hence set to 1 hour in **disgas.inp**) and the third vertical level from the ground (-L 3) (2 m above the ground as specified in **disgas.inp**). Two tracking points have been used in the same location but at different elevations (-TP True):

|  |  |  |  |
| --- | --- | --- | --- |
| **Tracking point ID** | **Easting (m)** | **Northing (m)** | **Elevation (m above ground)** |
| 1 | 644000 | 1775000 | 2 |
| 2 | 644000 | 1775000 | 10 |

Figure 4 shows examples of the produced plots:

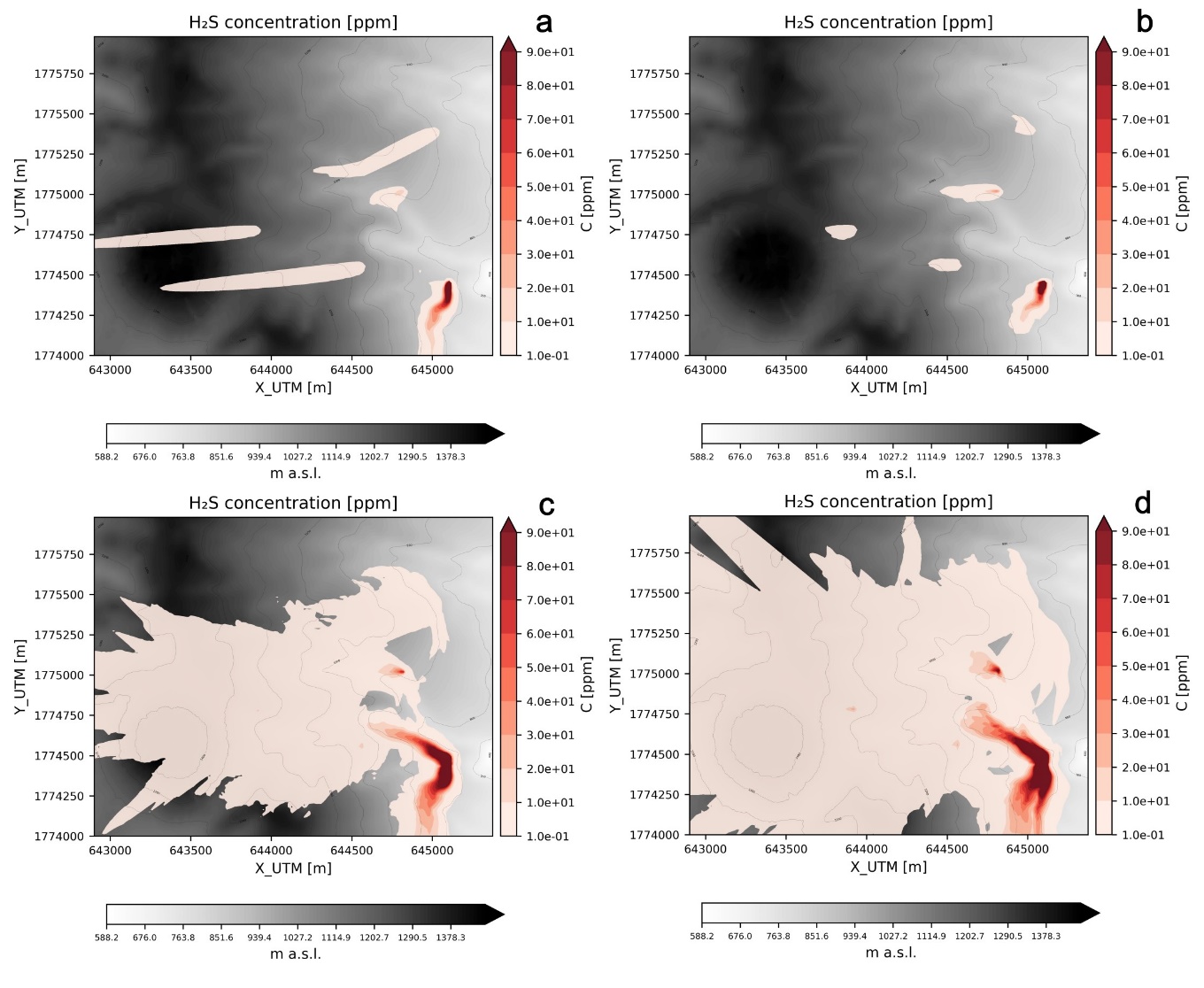


Figure 4. H2S concentration (in ppm) at 2 m above the ground 3 hours after the start of the emission. a) output from a single simulation; b) output corresponding to the 50% exceedance probability; c) output corresponding to the 5% exceedance probability; d) output corresponding to the 1% exceedance probability.

Similar to example 1, the concentration increases with decreasing exceedance probabilities, as expected. The meteorological variability also allows capturing areas of preferential dispersal, which in this area characterized by the presence of trade winds is westwards.

Fig. 5 shows the hazard curves (Exceedance probability vs. concentration) based on the 100 simulations at the two locations in the last time step of the simulation. The probability to exceed higher values of H2S concentrations are slightly higher in the tracking point 1 (2 m above the ground) compared to the other tracking point (10 m above the ground), with the highest possible concentration reaching ~2 ppm in the latter and 2.5 ppm in the former.

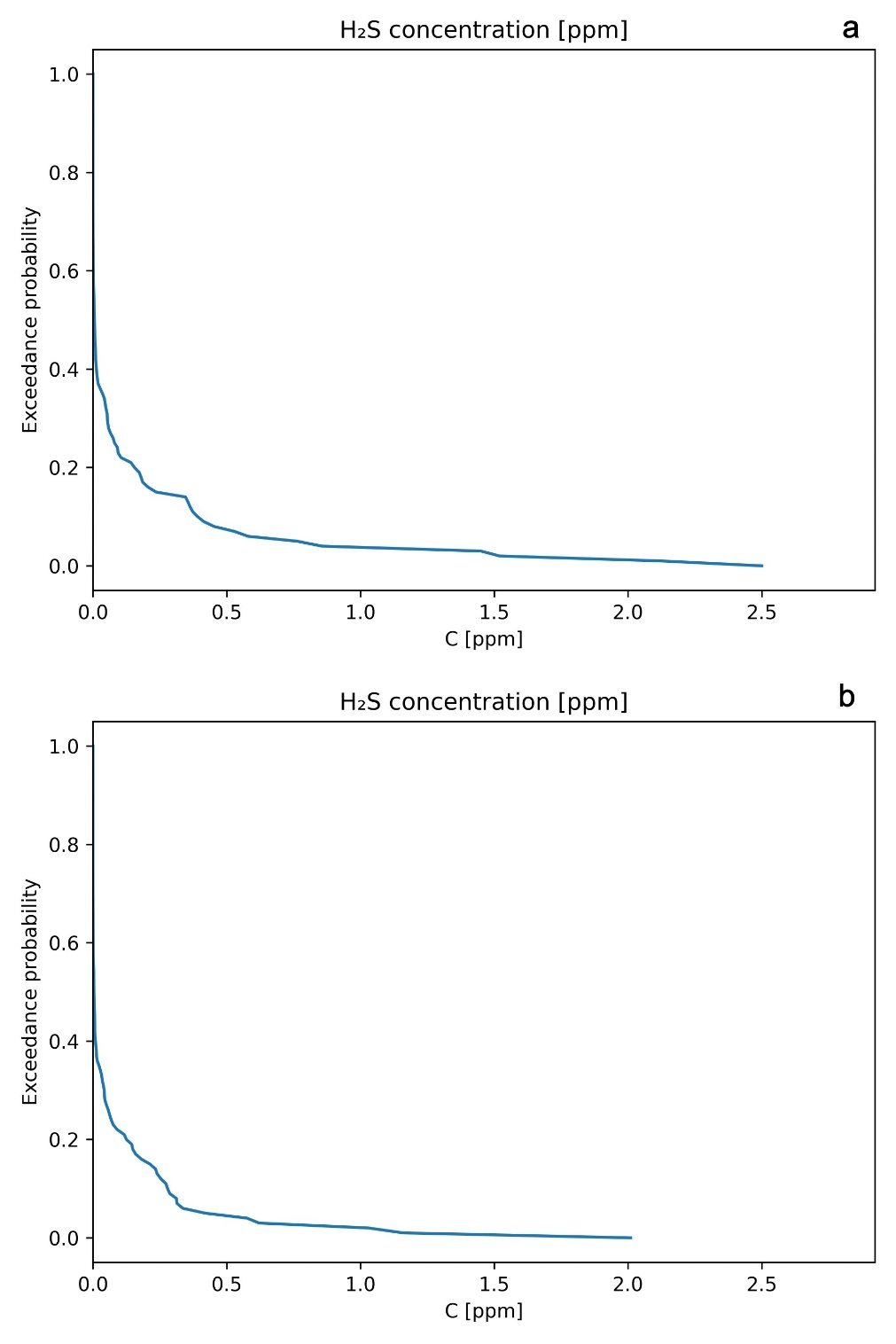


Figure 5. Hazard curve at the last time step of the simulation (+24 h from the beginning) in the tracking point 1 (a) and 2 (b).

The post processing for the case 2b is carried out with the command:

python ../../../post\_process.py -P true -T all -L 3 -D all -C True -TS H2O -S H2S -N 1 -U ppm -PT True -PL 0.1,100

Also in this case the simulated water vapour concentration is converted into H2S concentration (-C True -TS H20 -S H2S) and displayed in ppm (-U ppm). All graphical outputs are produced for all the time steps (-T all) and the third vertical level from the ground (-L 3) (2 m above the ground). Figure 6 shows examples of the produced plots:

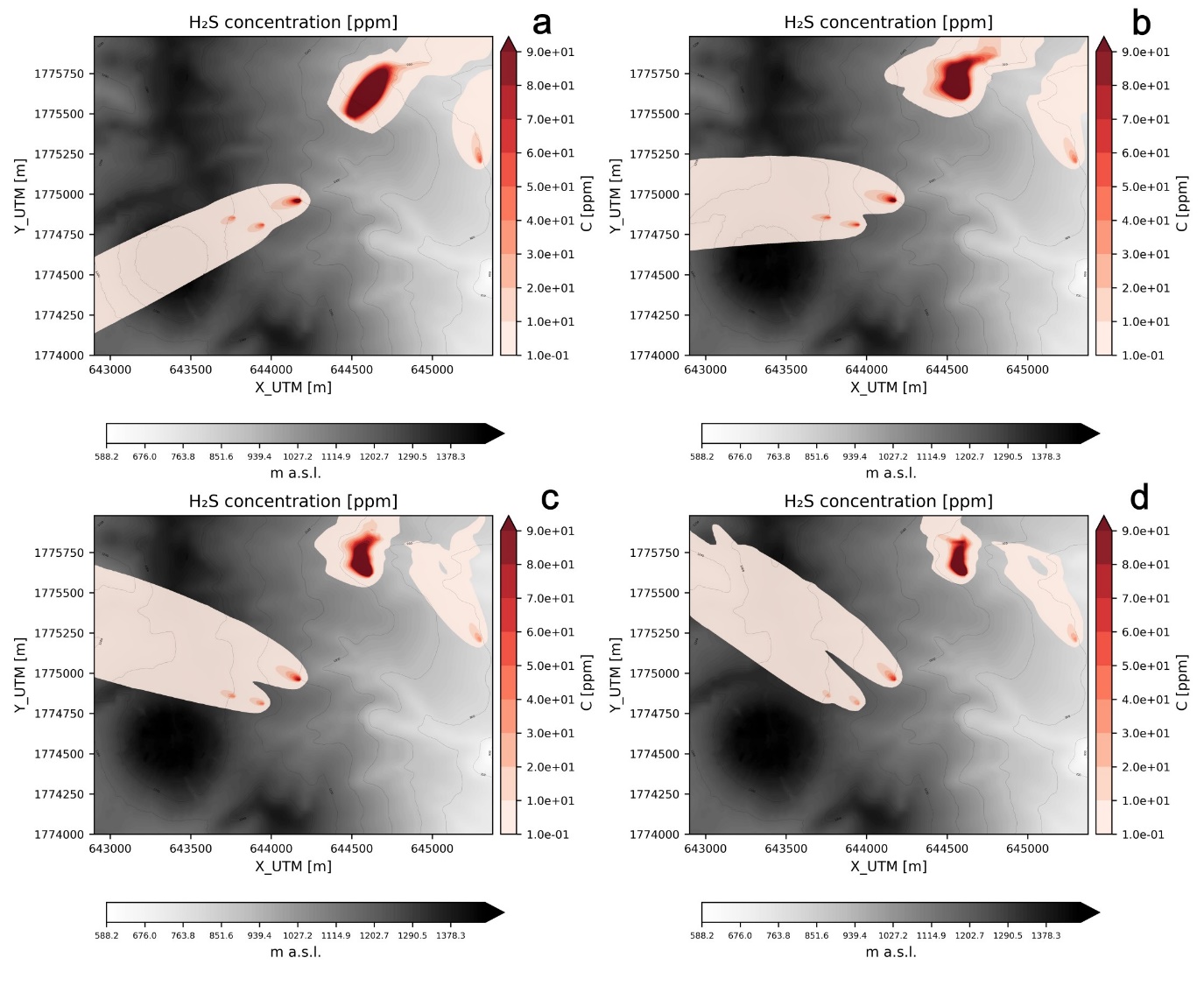


Figure 6. H2S concentration (in ppm) at 2 m above the ground at +6 hours (a), +12 hours (b), +18 hours (c) and +24 hours (d) after the start of the emission.

The progressive evolution of the gas clouds can be observed, together with changes in the propagation direction following the change of wind direction and speed.

## 5.3 Example 3: Mefite d’Ansanto area (Italy)

Mefite d’Ansanto represents the largest natural emission of low temperature CO2-rich gases from non-volcanic environment ever measured on Earth (Chiodini et al., 2010). For low wind conditions, the gas flows along a narrow natural channel producing a persistent gas river which has already killed many peoples and animals (Costa and Chiodini, 2015). In the Mefite d’Ansanto area, during periods with stable atmosphere and/or low winds, the gas, denser than the surrounding air, is channelized at the bottom of the valley forming a lethal and invisible gas river.

For simulating this scenario, we carry out two applications, with meteorology this time processed by DIAGNO using ERA5 reanalysis data:

* example 3a: we use TWODEE only for all the simulations.
* example 3b: we use the automatic scenario mode (see section 4.2), therefore VIGIL calculates the source Richardson number in each simulated day by using the source and meteorological data and, based on the obtained value, it runs DISGAS or TWODEE.

Details on the commands used in this test are in the file commands.txt in the example\_3 folder.

To process weather data in both cases, we run weather.py with the following command:

python ../../weather.py -M reanalysis -ERA5 True -S 01/01/1992 -E 01/01/2022 -LAT 40.975 -LON 15.14 -EL 600 -NS 100 -N 10 -TD on

Like in the example 2a, one hundred days (-NS 100) are randomly chosen from the period 01/01/1992 – 01/01/2022 (-S 01/01/1992 -E 01/01/2022). Unlike Example 1 and 2, in this case we are not dealing with a volcano, hence we provide the coordinate and the elevation of the reference point in the domain where to extrapolate ERA5 data (-LAT 40.975 -LON 15.14 -EL 600). Since we are running TWODEE, we need to inform VIGIL that additional TWODEE-specific weather data files are required (-TD on).

Subsequently we run the model with the command:

* example 3a: python ../../../run\_models.py -N 100 -RS off -D UTM,4535600,511300,4536110,512500 -NX 400 -NY 170 -DM twodee -RD 4 -TS CO2 -OH 0.05,0.5,1.0,1.5,2.0,2.5,3.0
* example 3b: python ../../../run\_models.py -N 100 -RS off -D UTM,4535600,511300,4536110,512500 -NX 400 -NY 170 -DM automatic -RD 4 -TS CO2 -OH 0.05,0.5,1.0,1.5,2.0,2.5,3.0

Similar to Example 2, we provide fixed sources via the sources\_input.txt file (-RS off). We turn off DISGAS by not specifying anything (-DG is off by default) and turn TWODEE on (-DM twodee) for the example 3a, while for the example 3b we specify the automatic scenario (-DM automatic).

After the simulation is complete, the post processing stage is carried out in both cases by:

python ../../../post\_process.py -P True -ECDF True -EX 0.5,0.05,0.01 -PER True -T 4 -TA 0 -L 3,6 -D all -N 100 -U ppm -TS CO2 -S CO2 -PT True -PL 400,10000

In these simulations, the tracking specie and the specie we are interested in coincide (-TS CO2 -S CO2) since CO2 is very abundant. Probability maps at exceedance probabilities of 50%, 5% and 1% are also requested (-ECDF True -EX 0.5,0.05,0.01). Persistence maps are also generated (-PER True) with the reference data available in gas\_properties.csv. All the probabilistic and graphical outputs are produced for the time step 4 (-T 4) (+4 hours from the beginning of the simulation and emission since the time step for the output is set to 1 hour, i.e., not specified and left to the default value in the run\_models.py command) and the third and sixth vertical levels from the ground (-L 3,6) (0.5 and 2 m above the ground as specified in the run\_models.py command, where, as already stated, the level at 0 m above the ground is included by default if not specified).

Sample outputs of example 3a are presented in figures 7 and 8. Specifically, fig. 7 shows the map of CO2 concentration at 5% exceedance probability averaged over the whole simulation duration (4 hours) at 0.5 m (Fig. 7a) and 2 m (Fig. 7b) above the ground. At this exceedance probability, very high concentrations (> 10,000 ppm) of CO2 are expected along the W-E valley but also in the areas close to the source and towards NE, the latter occurring when the CO2 cloud follows the prevailing south-westerly winds in the area. As expected, the concentration is higher at lower levels since the CO2 concentration is strongly vertically stratified.

Fig. 8 displays the output of the persistence calculations. For CO2, the threshold concentrations that can be tolerated by a human being for specified exposure durations, taken from NIOSH (1997), are provided in gas\_concentration.csv and reported in the following table:

|  |  |
| --- | --- |
| C threshold [ppm] | Exposure time [hours] |
| 1000 | 24 |
| 3500 | 24 |
| 5000 | 8 |
| 15000 | 1 |
| 30000 | 1 |
| 100000 | 1 |

The persistence maps calculated by VIGIL represent the probability to overcome the concentration threshold for a time equal or larger than the exposure time for that threshold. Since in this specific case the simulation lasts 4 hours, the first three rows are automatically discarded by VIGIL.

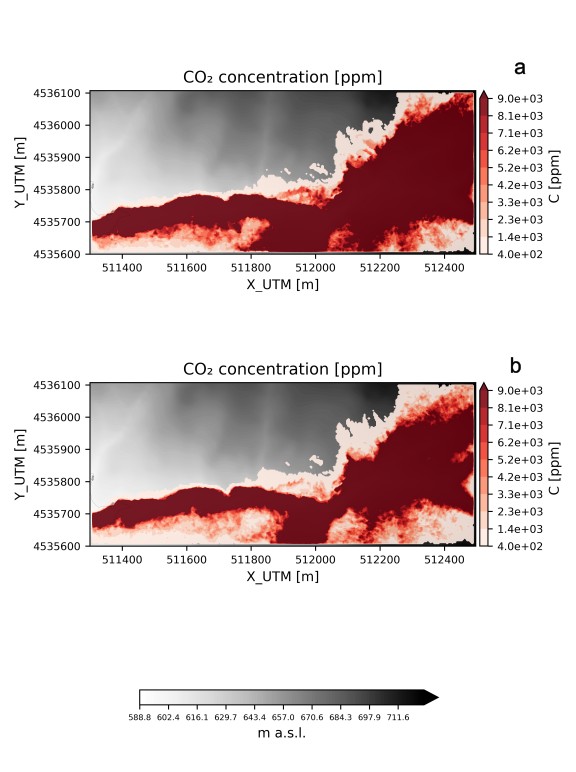


Figure 7. CO2 concentration (in ppm) at (a) 0.5 m and (b) 2 m above the ground averaged over the whole duration of the simulation (4 hours) at an exceedance probability of 5%.

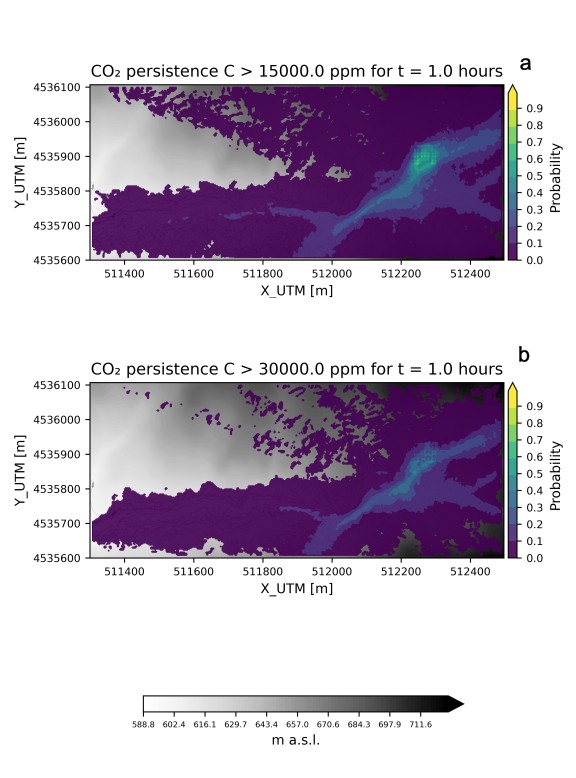


Figure 8. Persistence maps for CO2, i.e. the probability to overcome a concentration threshold of (a) 15,000 ppm for 1 hour and (b) 30,000 ppm for 1 hour. Both maps refer to a height of 2 m above the ground.

Fig. 8a shows the probability to overcome the concentration threshold of 15,000 ppm for at least 1 hour at 2 m above the ground; Fig 8b shows the same but for a concentration threshold of 30,000 ppm at 2 m above the ground. The maps inform us that there is a significant probability to overcome dangerous concentration thresholds at elevations that can affect a human being, particularly close to the source areas and along the valley. The probability is obviously higher for the threshold 15,000 ppm.

Figures 9 and 10 present sample results of the example 3b, in which DISGAS or TWODEE are automatically used by VIGIL for each simulation depending on the source Ri number. Specifically, the plots are equivalent to those of example 3a and presented in fig. 7 and 8.

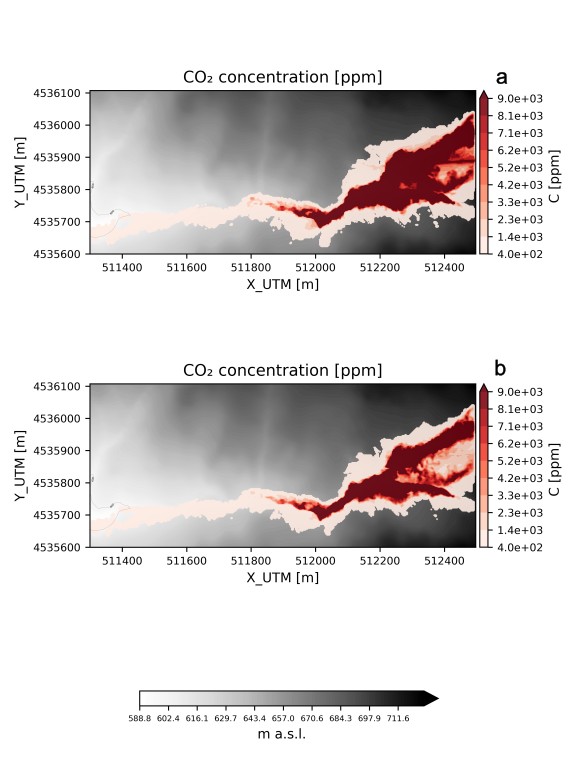


Figure 9. CO2 concentration (in ppm) at (a) 0.5 m and (b) 2 m above the ground averaged over the whole duration of the simulation (4 hours) at an exceedance probability of 5%.

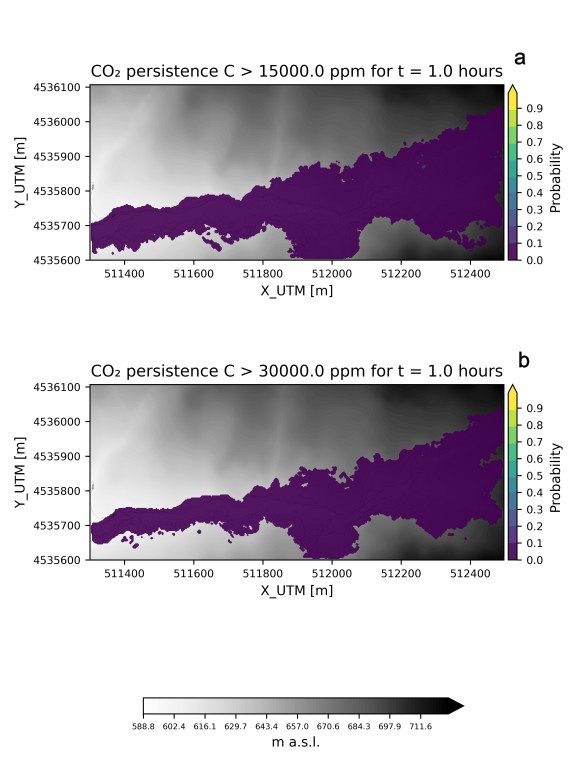


Figure 10. Persistence maps for CO2, i.e. the probability to overcome a concentration threshold of (a) 15,000 ppm for 1 hour and (b) 30,000 ppm for 1 hour. Both maps refer to a height of 2 m above the ground.

The difference with the outputs of example 3a are evident. The most striking one is the probability to generate a cold gas flow in the valley, which is much lower in the simulations of example 3b. This can be attributed to the automatic scenario (light gas with DISGAS or heavy gas with TWODEE) based on the calculation of Ri at the source. The results seem to imply that the likelihood to generate a cold gas stream in the valley is low, although not negligible, and this is consistent with the observations. In fact, Chiodini et al. (2010) simulated the cold gas flow with TWODEE in a no-wind scenario. According to our calculation for this example, a 2 m s-1 wind is enough to reach the condition Ri < 0.25, for which the light gas scenario should be taken into account. Therefore, the new automatic scenario capability shows its potential for obtaining more realistic simulations and probabilistic hazard assessments.

# 6. Future perspectives

VIGIL is under constant development and ideas for further improvements are always welcome. Possible improvements are:

* Overcome the restrictions imposed by DIAGNO on the single run duration (24 hours). Another wind pre-processor may be used.
* Usage of high-resolution meteorological data (e.g. outputs of [WRF](https://www.mmm.ucar.edu/weather-research-and-forecasting-model)).
* Improve the way to provide input data, e.g. using an input file instead of the arguments of the Python scripts.
* Create a user-friendly Graphical User Interface.

# Acknowledgments

Fabio Dioguardi has been supported from UK National Capability funding (BGS Innovation Flexible Fund). Fabio Dioguardi and Silvia Massaro have also been supported by the European Union's Horizon 2020 project EUROVOLC (grant agreement no 731070).

We thank Giovanni Macedonio, Laura Sandri, Antonio Costa, Jacopo Selva, Giancarlo Tamburello, Giovanni Chiodini and Arnau Folch for the fruitful discussions and their valuable suggestions that significantly contributed to the development of VIGIL.

# References

Byun, D.W., 1990. On the analytical solutions of flux-profile relationships for the atmospheric surface layer. Journal of Applied Meteorology, 29(7), 652-657.

Cardellini, C., Chiodini, G., Frondini, F., Avino, R., Bagnato, E., Caliro, S., and Rosiello, A., 2017. Monitoring diffuse volcanic degassing during volcanic unrests: the case of Campi Flegrei (Italy). Scientific Reports, 7(1), 1-15.

Chiodini, G., Todesco, M., Caliro, S., Del Gaudio, C., Macedonio, G., and Russo, M., 2003. Magma degassing as a trigger of bradyseismic events: The case of Phlegrean Fields (Italy). Geophysical Research Letters, 30(8).

Chiodini, G., Granieri, D., Avino, R., Caliro, S., Costa, A., Minopoli, C., and Vilardo, G., 2010. Non‐volcanic CO2 Earth degassing: Case of Mefite d'Ansanto (southern Apennines), Italy. Geophysical Research Letters, 37(11).

Cortis, A., and Oldenburg, C.M., 2009. Short-range atmospheric dispersion of carbon dioxide. Boundary-layer meteorology, 133(1), 17-34.

Costa, A., Macedonio, G., Chiodini, G., 2005. Numerical model of gas dispersion emitted from volcanic sources. Annals of Geophysics, vol. 48, n.4/5.

Costa, A., Chiodini, G., Granieri, D., Folch, A., Hankin, R.K.S., Caliro, S., and Cardellini, C., 2008. A shallow‐layer model for heavy gas dispersion from natural sources: Application and hazard assessment at Caldara di Manziana, Italy. Geochemistry, Geophysics, Geosystems, 9(3).

Costa A., Folch A., Macedonio G., 2013. Density-driven transport in the umbrella region of explosive eruptions: effects on tephra dispersion models. Geophysical Research Letters, 40, 1-5.

Costa, A. and Chiodini, G., 2015. Modelling air dispersion of CO2 from limnic eruptions. In Volcanic Lakes, 451-465. Springer, Berlin, Heidelberg.

Costa, A. and Macedonio, G., 2016. DISGAS-2.0: a model for passive DISpersion of GAS. Rapporti tecnici INGV. Istituto Nazionale Di Geofisica e Vulcanologia, Italy, 332, 2039e7941.

Dioguardi, F., Massaro, S., Chiodini, G., Costa, A., Folch, A., Macedonio, G., Sandri, L., Selva, J., Tamburello, G., 2022. VIGIL: A Python tool for automatized probabilistic VolcanIc Gas dIspersion modeLling. Annals of Geophysics, 65(1), DM107, <https://doi.org/10.4401/ag-8796>.

Douglas, S.G., Kessler, R.C., and Carr, E.L., 1990. User's guide for the Urban Airshed Model. Volume 3. User's manual for the Diagnostic Wind Model (No. PB-91-131243/XAB). Systems Applications, Inc., San Rafael, CA (USA).

Dutton, J.A., and Fichtl, G.H., 1969. Approximate equations of motion for gases and liquids. Journal of the Atmospheric Sciences, 26(2), 241-254.

Edmonds, M., Grattan, J., Michnowicz, S., 2018. Volcanic gas: silent killers. In Observing the Volcano World: Volcano Crisis Communication, 65-84. Springer, Cham.

Ewing, R., Wang, H., 2001. A summary of numerical methods for time-dependent advection-dominated partial differential equations. Journal of Computational and Applied Mathematics 128, 423–445.

Folch, A., Costa, A., and Hankin, R.K., 2007. TWODEE: Computer Code and Related Documentation. Project INGV-DPC V5 Diffuse degassing in Italy (2005-2007).

Folch, A., Costa, A., and Hankin, R.K., 2009. TWODEE: a shallow layer model for dense gas dispersion on complex topography. Computers & Geosciences, 35(3), 667-674.

Folch, A., Barcons, J., Kozono, T., and Costa, A., 2017. High-resolution modelling of atmospheric dispersion of dense gas using TWODEE.1: application to the 1986 Lake Nyos limnic eruption. Natural Hazards and Earth System Sciences.

Granieri D., Costa A., Macedonio G., Chiodini G., Bisson M., 2013. Carbon dioxide in the city of Naples: contri-bution and effects of the volcanic source. Journal of Volcanology and Geothermal Research, 260, 52-62.

Hankin, R.K.S., and Britter, R.E., 1999. Twodee: the Health and Safety Laboratory's shallow layer model for heavy gas dispersion Part 3: Experimental validation (Thorney Island). Journal of hazardous materials, 66(3), 239-261.

Lax, P.D., and Wendroff, B., 1960. Systems of conservative laws. Communications on Pure and Applied Mathematics 13, 217–237.

Massaro, S., Dioguardi, F., Sandri, L., Tamburello, G., Selva, J., Moune, S., Jessop, D.E., Moretti, R., Komorowski, J.-C., Costa, A. Testing gas dispersion modelling at La Soufrière volcano (Guadeloupe, Lesser Antilles). Journal of Volcanology and Geothermal Research, 107312. <https://doi.org/10.1016/j.jvolgeores.2021.107312>

Marzocchi, W., Zaccarelli, L., 2006. A quantitative model for the time‐size distribution of eruptions. Journal of Geophysical Research: Solid Earth, 111(B4).

Monin, A.S., Yaglom, A.M., 1979. Statistical Fluid Mechanics, vol. 1, Chapter 5.

NIOSH (1997). Pocket Guide to Chemical Hazard, DHHS (NIOSH), Publ. N.97-140, U.S. Gov. Print Office, Washington, D.C.

Pedone, M., Granieri, D., Moretti, R., Fedele, A., Troise, C., Somma, R., and De Natale, G., 2017. Improved quanti-fication of CO2 emission at Campi Flegrei by combined Lagrangian Stochastic and Eulerian dispersion modelling. Atmospheric Environment, 170, 1-11.

Prabha T., Mursch-Radlgruber E., 1999. Modeling of diffusion in a wide Alpine Valley. Theor. Appl. Climatol., 64, 93-103.

Sandri, L., Costa, A., Selva, J., Tonini, R., Macedonio, G., Folch, A., and Sulpizio, R., 2016. Beyond eruptive scenarios: assessing tephra fallout hazard from Neapolitan volcanoes. Scientific Reports, 6, 24271.

Selva, J., Orsi, G., Di Vito, M. A., Marzocchi, W., and Sandri, L., 2012a. Probability hazard map for future vent opening at the Campi Flegrei caldera, Italy. Bulletin of Volcanology, 74(2), 497-510.

Selva, J., Marzocchi, W., Papale, P., & Sandri, L., 2012b. Operational eruption forecasting at high-risk volcanoes: the case of Campi Flegrei, Naples. Journal of Applied Volcanology, 1(1), 5.

Tamburello, G., Moune, S., Allard, P., Venugopal, S., Robert, V., Rosas-Carbajal, M., Deroussi, Gaëtan-Thierry Kitou, Didier, T., Komorowski, J.C., Beauducel, F., De Chabalier J.B., Le Marchand, A., Le Friant, A., Bonifacie, M., Dessert, C., and Moretti, R., 2019. Spatio-Temporal Relationships between Fumarolic Activity, Hydrothermal Fluid Circulation and Geophysical Signals at an Arc Volcano in Degassing Unrest: La Soufrière of Guadeloupe (French West Indies). Geosciences, 9(11), 480.

Tarquini S., Isola I., Favalli M., Battistini A., 2007. TINITALY, a digital elevation model of Italy with a 10 meters cell size (Version 1.0) [Data set]. Istituto Nazionale di Geofisica e Vulcanologia (INGV). https://doi.org/10.13127/TINITALY/1.0.