Atmospheric Tomography Software User Manual

Laura Cartwright[1], Sangeeta Bhatia[2], Andrew Feitz[3], Andrew Zammit-Mangion[1], and Andrew Francis[4]

[1]*School of Mathematics and Applied Statistics, University of Wollongong, Wollongong, Australia*
[2]*Centre for Research in Mathematics, Western Sydney University, Sydney, Australia*
[3]*Geoscience Australia, Canberra, Australia*
[4]*Centre for Research in Mathematics, Western Sydney University, Sydney, Australia*

Contents

1 License

This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

2 About the software

The Atmospheric Tomography software is a command line tool written in python to estimate the emission rate of a point source from concentration data. It implements an extension of the Bayesian inversion method outlined in Kuske et al. (2013).

This manual walks a user through the steps needed to run the Atmospheric Tomography program. Please report any errors or omissions to Laura Cartwright (`lcartwri@uow.edu.au`).

A date file titled "simulated_data.csv" has been included. This is a data set containing 34560 simulated observations, and has been included to allow the user an opportunity to familiarise themselves with the software. Appendix A gives instructions on running the software on this data.

3 Installation

3.1 Dependencies

1. Python (version 3 or later)
2. R (version 3.5 or later recommended)
3. Inquirer (Python package)
4. PyMC (Python package)
5. Rpy2 (Python package)
6. lubridate (R package)
7. tidyr (R package)
8. dplyr (R package)
9. coda (R package)

Instructions for installing PyMC are available at `https://pymc-devs.github.io/pymc/INSTALL.html`. This page also has instructions for installing the correct version of Python. Using a pre-built distribution such as Pip (`https://pip.pypa.io/en/stable/installing/`) **for Mac & Linux users**, or Anaconda (`http://continuum.io/downloads`) **for Windows users** is highly recommended. Windows users may need to check they have a Fortran compiler installed, as well as Microsoft Visual C++ 14 (from Microsoft Visual C++ build tools).

Instructions for installing Inquirer are available at `https://python-inquirer.readthedocs.io/en/latest/`.

R can be freely downloaded from `https://www.r-project.org/`. You will need to install the following packages: dplyr, tidyr, coda, and lubridate. To install packages, type

```
install.packages("<package name>")
```

into the command line, and to load the packages, type

```
library("<package name>")
```

into the command line.

Instructions for installing Rpy2 are available at `http://rpy2.readthedocs.io/en/version_2.8.x/introduction.html`. You will need to install dplyr, tidyr, coda, and lubridate again within the Rpy2 package. The code to install these packages is as follows:

```
>>> import rpy2.robjects.packages as rpackages
>>> from rpy2.robjects.vectors import StrVector
>>> utils = rpackages.importr('utils')
>>> utils.chooseCRANmirror(ind = 1)

rpy2.rinterface.NULL

>>> packages = ('lubridate', 'tidyr', 'dplyr', 'coda')
>>> utils.install_packages(StrVector(packages))
```

3.2 Test the environment

If the installation is successful, you should be able to run Python and import the PyMC and Rpy2 modules. To test this, open the terminal utility on Mac OS X or Linux or the DOS prompt on Windows. The DOS prompt can be opened by clicking on the Windows (or Run) button on your machine and typing CMD (note you may need to open the DOS prompt with admin privileges).

On the command prompt type

```
> python --version
```

If Python installation was successful, the command should run successfully and output the version of Python on your machine.

```
> python --version

Python 3.6.4

>
```

If you get an error at this step, check if the PATH environment variable on your machine contains the path to python binaries. The Troubleshooting section of this manual contains more information on how to resolve this issue.

4 Bayes theorem applied to quantification

4.1 The atmospheric transfer model

The Bayesian approach to statistics relies on using Bayes' rule and observed evidence to update some prior belief on a parameter or parameters of interest. In our case, the parameter of

Table 1: Stability classes to which observations within the Ginninderra experiment are allocated, and the consequent values of $a_{k_i}, b_{k_i}, c_{k_i}$, and d_{k_i} allocated to each observation.

Stability Class (k_i)	**Stability Condition**	a_{k_i}	b_{k_i}	c_{k_i}	d_{k_i}
A	Very unstable	0.0383	1.281	0.495	0.873
B	Unstable	0.1393	0.9467	0.310	0.897
C	Neutral	0.0856	0.8650	0.122	0.916
D	Stable	0.1094	0.7657	0.0934	0.912
E	Very stable	0.05645	0.8050	0.0625	0.911

interest is the release rate (Q) of whatever gas was involved in the experiment of interest. The atmospheric tomography methods discussed here are largely based on those developed in Humphries et al. (2012).

The spread of methane after release is modelled using a *Gaussian plume dispersion model*, C. The model is proposed in Turner (1994) among other texts, and is given by

$$C(x_i, y_i, z_i, Q, \boldsymbol{\theta}_{k_i}) = \frac{Q}{2\pi U_i \sigma_{y_i,k_i} \sigma_{z_i,k_i}} \exp\left(-\frac{y_i^2}{2\sigma_{y_i,k_i}^2}\right)\left[\exp\left(-\frac{(z_i - H)^2}{2\sigma_{z_i,k_i}^2}\right) + \exp\left(-\frac{(z_i + H)^2}{2\sigma_{z_i,k_i}^2}\right)\right],$$

where N is the number of observations, $C(x_i, y_i, z_i, Q, \boldsymbol{\theta}_{k_i})$ is the concentration of methane at a point (x_i, y_i, z_i), Q is the methane release rate, U_i is the wind speed, H is the height of the gas source, $\boldsymbol{\theta}_{k_i} = (U_i, H, a_{k_i}, b_{k_i}, c_{k_i}, d_{k_i})'$, $k_i = A, B, C, D, E$, and $\sigma_{z_i,k_i} = a_{k_i} x^{b_{k_i}}$, $\sigma_{y_i,k_i} = c_{k_i} x^{d_{k_i}}$ are the standard deviations of the time averaged plume concentrations in the z and y directions respectively, and for $a_{k_i}, b_{k_i}, c_{k_i}, d_{k_i} \in \mathbb{R}$ (Turner, 1994). We describe how these values are chosen in further detail below. An example of this plume for the values a_E, b_E, c_E, and d_E is shown in Figure 1.

Our model differs from that of Humphries et al. (2012) in that they chose to use a semi-Gaussian plume dispersion model instead of the fully Gaussian model (see Section 4.4).

One aspect of this model to keep in mind is that it assumes that the x-axis is the centreline of the plume, and that the methane source is at the origin. Thus before any analysis can be performed, we must zero the location of all instruments involved in the experiment by the location of the source, and rotate the coordinates of all instruments such that the x-axis becomes the centreline of the plume. This is done automatically by the software, using the wind direction in each instance, and is explained in further detail in Section 7.1 of Cartwright (2018). It should also be noted that this model returns values in units of grams per cubic metre, while the software takes in values in ppm. The conversion is done within the software after the concentration values are produced, via the ideal gas law.

The values of $a_{k_i}, b_{k_i}, c_{k_i}$, and d_{k_i} are based on those suggested in Turner (1994). We aim to compare the concentrations we have observed to those predicted by our plume model, and so we assign values of $a_{k_i}, b_{k_i}, c_{k_i}$, and d_{k_i} to the predicted concentration based on the stability of the corresponding observed concentration. We allocate each observation to one of five stability classes, which are given in Table 1, along with the possible values of $a_{k_i}, b_{k_i}, c_{k_i}$, and d_{k_i}.

In the extremely unstable case ($k_i = E$), σ_{y_i,k_i} and σ_{z_i,k_i} are large, causing the plume itself

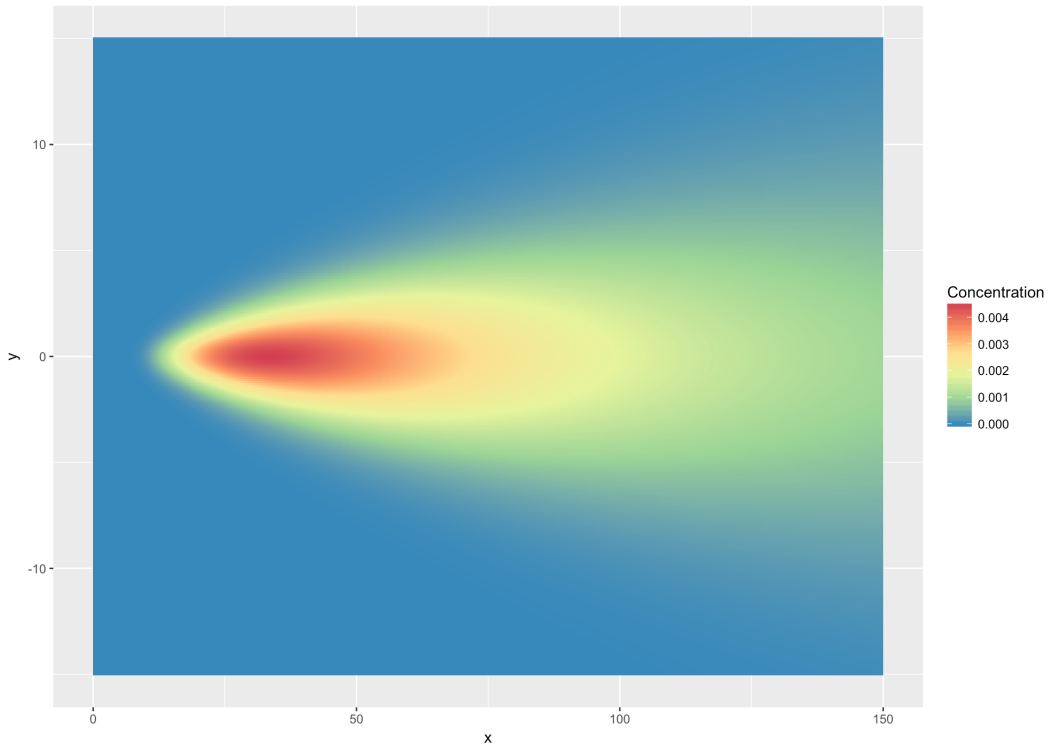

Figure 1: Plot of the Gaussian plume dispersion model at a height of 1.5 m, with $a_{k_i} = 0.05645, b_{k_i} = 0.8050, c_{k_i} = 0.0625$, and $d_{k_i} = 0.911$, and $k_i = E$.

to become more spread out, and thus allowing for the more unstable wind conditions. The stability class to which an observation is allocated is based on the Monin-Obhukov length (L value), given by

$$L = -\frac{u_*^3 \bar{\xi}_v}{kg(\overline{w'\xi_v'})_s},$$

where u_* is the frictional velocity, $\bar{\xi}_v$ is the mean virtual potential temperature, $(\overline{w'\xi_v'})_s$ is the surface virtual potential temperature flux, k is the von Kármán constant, and g is the acceleration due to gravity constant (Jacobson, 2005). Windtrax was used to determine the L-value of each observation for the Ginninderra experiment. Weather data such as wind speed, wind direction, and temperature were given to Windtrax, and the frictional velocity, mean virtual potential temperature, and surface virtual potential temperature flux were calculated, as well as the L-values (see Bhatia, 2015; Feitz et al., 2018, for more detail). For any user wishing to run the atmospheric tomography software on a new data set, these L-values will need to be determined first.

4.2 Bayesian inference

Let $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_N)'$ denote the N measured (observed) concentrations of methane from the experiment of interest and $\mathbf{\Theta} = (\boldsymbol{\theta}_{k_1}, \boldsymbol{\theta}_{k_2}, \ldots, \boldsymbol{\theta}_{k_N})'$, where $'$ indicates the transpose. Let τ be the precision parameter associated with the predicted methane concentrations calculated from the Gaussian plume dispersion model, which correspond to each observed concentration. We update our beliefs about the methane release rate Q by finding the posterior distribution of Q. We derive this distribution by using Bayes' rule and integrating over τ:

$$p(Q \mid \mathbf{Y}, \mathbf{\Theta}) \propto \int_0^\infty p(\mathbf{Y}, \tau \mid Q, \mathbf{\Theta}) p(Q) \, \mathrm{d}\tau = p(Q) \int_0^\infty p(\mathbf{Y} \mid \tau, Q, \mathbf{\Theta}) p(\tau) \, \mathrm{d}\tau, \qquad (1)$$

where $p(Q)$ (the prior) is our prior belief on the release rate Q before observing any evidence; $p(\mathbf{Y} \mid \tau, Q, \mathbf{\Theta})$ (the likelihood) is the likelihood that we observe what we did for some values of τ, Q, and Θ, and $p(Q \mid \mathbf{Y}, \mathbf{\Theta})$ (the posterior) is our belief on Q given certain values for $\mathbf{\Theta}$, and given that we have observed $\mathbf{Y}$.

Since the $\{Y_i \mid Q, \boldsymbol{\theta}_{k_i}\}$ are assumed to be mutually independent, we can rewrite (1) as

$$p(Q \mid \mathbf{Y}, \boldsymbol{\theta}_{k_i}) \propto p(Q) \int_0^\infty \prod_{i=1}^{N} p(Y_i \mid \tau, Q, \boldsymbol{\theta}_{k_i}) p(\tau) \, \mathrm{d}\tau. \tag{2}$$

While it is generally possible to fit "nice" functions to the prior distribution and likelihood function separately, it is often the case that the product of the two becomes rather complicated and analytically intractable. This means it is not always possible to derive our posterior distribution analytically. Markov Chain Monte Carlo (MCMC) can be used to sample from the posterior distribution, uncover the *shape* of the distribution, and hence allow us to do inference using it. From the collection of samples obtained via MCMC, we attempt to estimate the methane release rate, Q.

We model Q using a Half-Normal prior with a standard deviation, σ, of 1.5. This distribution has non-negative support, and so we choose this over simply a Normal prior as we cannot have a negative release rate. So the prior pdf of Q we use is

$$p(q) = \frac{\sqrt{2}}{\sigma\sqrt{\pi}} \exp\left(-\frac{Q^2}{2\sigma^2}\right), \quad q \in [0, \infty).$$

We model τ using a Gamma prior, with shape parameter, α, equal to 1.058, and rate parameter, β, equal to 0.621. These parameters were chosen such that the 5th and 95th percentiles of the distribution were 0.1 and 5 respectively. So the prior pdf of τ is given by

$$p(\tau) = \frac{\beta^\alpha}{\Gamma(\alpha)} \tau^{\alpha-1} e^{-\beta\tau}.$$

Each observed concentration, Y_i, is modelled using a Normal distribution with unknown precision τ. The mean of each ith observation is the predicted concentration as determined by the Gaussian plume dispersion model when conditioned on some parameter vector $\boldsymbol{\theta}_{k_i}$. So, the likelihood of each Y_i is given by

$$p(Y_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(Y_i - C_i)^2}{2\sigma^2}\right),$$

where $C_i = C(x_i, y_i, z_i, \boldsymbol{\theta}_{k_i})$, (x_i, y_i, z_i) is the location of the instrument corresponding to the ith observation, and $\boldsymbol{\theta}_{k_i} = (U_i, H, a_{k_i}, b_{k_i}, c_{k_i}, d_{k_i})'$ gives the wind speed corresponding to the ith observation, the height of the methane source, and the values of $a_{k_i}, b_{k_i}, c_{k_i}$, and d_{k_i} corresponding to the ith observation, respectively.

Calculating predicted concentrations along laser paths

For laser paths, C_i represents the average of the line integral of concentrations along the path between the laser and reflector. That is, for a laser-reflector path (straight line) between given points (x_1, y_1) and (x_2, y_2), parametrised by $\mathbf{k}(s) = (k_x(s), k_y(s))$, where

$$\begin{aligned} k_x(s) &= sx_2 + (1-s)x_1, \quad s \in [0, 1] \\ k_y(s) &= sy_2 + (1-s)y_1, \quad s \in [0, 1], \end{aligned}$$

we have

$$C_i = \frac{1}{L}\int_0^1 C(k_x(s), k_y(s), z_i, Q, \boldsymbol{\theta}_{k_i}) \left\| \frac{\mathrm{d}\mathbf{k}}{\mathrm{d}s} \right\| \, \mathrm{d}s,$$

where $L \in \mathbb{R}$, and $\|\cdot\|$ refers to the standard Euclidean norm. In our case, $L = \left\|\frac{\mathrm{d}\mathbf{k}}{\mathrm{d}s}\right\|$, which is no longer a function of s, and so the above simplifies to

$$C_i = \int_0^1 C(k_x(s), k_y(s), z_i, Q, \boldsymbol{\theta}_{k_i}) \, \mathrm{d}s.$$

This integral is approximated by the software. Let

$$P = \{[s_0, s_1], [s_1, s_2], \ldots, [s_{99}, s_{100}]\}$$

be a partition over $[0, 1]$, with $0 = s_0 < s_1 < \cdots < s_{99} < s_{100} = 1$, $\Delta s = s_j - s_{j-1}$, $s_j^* = \frac{s_j + s_{j-1}}{2}$, and $j = 1, 2, \ldots, 100$. Then

$$C_i \approx \sum_{j=1}^{100} C(k_x(s_j^*), k_y(s_j^*), z_i, Q, \boldsymbol{\theta}_{k_i}) \Delta s.$$

For our work, we use an equal step size of $\Delta s = 1/100$ (see Section 7 for details on changing this step size), and so our approximation becomes

$$C_i \approx \frac{1}{100} \sum_{j=1}^{100} C(k_x(s_j^*), k_y(s_j^*), z_i, Q, \boldsymbol{\theta}_{k_i}).$$

4.3 Changing the priors on Q and τ

The prior distributions over Q and τ can be changed by going to lines 148 and 149 of the file "atmospheric-tomography.py". Changing this file requires knowledge of Python. Many useful priors are made available by PyMC. PyMC documentation also contains helpful information on writing your own priors.

4.4 Other Plume Dispersion Models

Two other plume dispersion models are implemented. The first of these, which we call the semi-Gaussian model presented in Humphries et al. (2012) has the the functional form:

$$C(x, y) = Q\frac{A}{x^E} \exp\left(-\frac{B}{x^F}\right) \exp\left(-Cy^2 \frac{(D + x^G)^2}{x^H}\right).$$

A simplified version is the following function:

$$C(x, y) = Q(c_0 + c_1 x + c_2 x^2 + c_3 x^3) e^{-(y^2/Ax^E)},$$

which is called the Gaussian-polynomial hybrid function.

Comparisons between these models, as well as a Gaussian polynomial hybrid model are explored in Bhatia (2015), where it was determined that for the Ginninderra experiment (for which this software was originally developed), the best performing model was the Gaussian model. The user of the software is able to change to one of the other plume models described here (see Section 5).

The parameters for these models for various values of the Monin-Obhukov length (i.e., different stability classes) were derived by fitting the concentrations from a Lagrangian stochastic (LS) model (Thompson and Stohl, 2014) as implemented in WindTrax Version 2.0.8.8.

Note that these dispersion models also place the origin at the source and the horizontal axis along the plume centerline.

5 Modifying plume dispersion model parameters

The models implemented in the software are called 'gaussian', 'semi-gaussian' and 'gauss_poly'. The default model is the Gaussian model and its use is highly recommended.
If you want to choose a different model, open the file "atmospheric-tomography.py" in a text editor and go to lines 158 and 163:

```
conc[i] = 0 if rotcor[0] < 0 else util.predicted2(rotcor[0], rotcor[1],
z[i], initial_Q, lvals[i], w_speed[i], h_source, 'gaussian')

conc[i] = util.line_average([source_x[i], source_y[i]], [x2[i], y2[i]],
[x1[i], y1[i]], z[i], samples, initial_Q, h_source, plume_dir[i], temp[i],
pressure[i], params, 'gaussian')
```

Change the last parameter to one of the following values (including the single quotes): 'semi-gaussian' or 'gauss_poly'. Do not copy the line above from this manual since python is space sensitive. Make sure that there are no spaces around the name and that the file is saved with the extension ".py".

The parameter values for the models for each stability class are stored in the file called "params.py". To edit these parameters, open this file in a text editor and navigate to the section that has the same name as the method. For instance, the Gaussian plume dispersion model makes use of 4 parameters called $a_{k_i}, b_{k_i}, c_{k_i}$, and d_{k_i}. To edit these values, for example, go to lines 69 – 75 of "params.py":

```
gaussian = {
'A': {'a': 0.0383, 'b': 1.281, 'c': 0.495, 'd': 0.873},
'B': {'a': 0.1393, 'b': 0.9467, 'c': 0.310, 'd':0.897},
'C': {'a': 0.0856, 'b': 0.8650, 'c': 0.122, 'd':0.916},
'D': {'a': 0.1094, 'b': 0.7657, 'c': 0.0934, 'd':0.912},
'E': {'a': 0.05645, 'b': 0.8050, 'c': 0.0625, 'd':0.911}
}
```

You can change these values and save the file. Make sure the file is saved with the extension ".py" as many text editors usually add the extension ".txt" when a file is saved.

6 Preparing the data files

The first step is to get the data in the format expected by the scripts. The input file should be a comma separated file with the following columns, in this order:

1. `Time` in format "YY-M-DD HH:MM";

2. `release_rate` in grams per minute;
3. `air_temp` in degrees Kelvin;
4. `air_pressure` in pascals;
5. `wind_speed` in metres per second;
6. `wind_dir` in degrees East of North;
7. `L` in metres (Monin-Obhukov length);
8. `inst_name` (name of the instrument);
9. `inst_no` (should be an integer);
10. `source_x` (x-coordinate of the gas source);
11. `source_y` (y-coordinate of the gas source);
12. `z` (Height of the instrument in metres. For laser paths, average height across path);
13. `x1` (x-coordinate of reflector or tower);
14. `y1` (y-coordinate of reflector or tower);
15. `x2` (x-coordinate of laser (NA if tower measurement));
16. `y2` (y-coordinate of laser (NA if tower measurement));
17. `Concentration` in parts per million (not background subtracted).

The order and names of the columns cannot be changed.

Often the data are stored in a wide format, rather than the long format that is required. That is, the weather data (temperature, pressure, wind speed and direction) are recorded separately and the concentration measurements are recorded as:

```
Time, Instrument_1, Instrument_2, Instrument_3...
```

meaning that some pre-processing may be required before running the software. Another consideration is that the frequency at which the weather and concentration data are recorded could be different. For instance, the meteorological data might be collected every five minutes while the concentration data could be recorded several times in a minute. In this case, it is necessary to average the concentration records to the frequency of the weather records.

Pre-processing

The author of this software carried out this pre-processing using the statistical software R. To guide the user, we outline how the pre-processing was carried out for our data. The following steps were taken to collect all of the data together into a single file:

1. Each data file containing measured methane concentrations (if more than one exist) for particular instruments, as well as associated L values for each observation, were read into `R` and a data frame for the data from each file created.

index	Time	release_rate	air_temp	air_pressure	wind_speed	wind_dir	L	inst_name	inst_no	source_x	source_y	z	x1	y1	x2	y2	Concentration
integer	*dd/mm/yy hh:mm*	*g/min*	*Kelvin*	*pascals*	*m/s*	*degrees N of E*	*m*	*character*	*integer*	*x-coordinate*	*y-coordinate*	*m*	*x-coordinate*	*y-coordintate*	*x-coordinate*	*y-coordinate*	*ppm*
1	21/5/15 16:35	5.8	285.8417	94703.1843	1.46	288	26.8	R4	4	-21.78	21.09	1.5	11.1	18.16	-45.75	16.2	2.95
2	21/5/15 16:35	5.8	285.8417	94703.1843	1.46	288	26.8	R5	5	-21.78	21.09	1.5	6.96	2.72	-45.75	16.2	1.62
3	21/5/15 16:35	5.8	285.8417	94703.1843	1.46	288	26.8	R6	6	-21.78	21.09	1.5	2.04	-12.17	-45.75	16.2	1.59
4	21/5/15 16:35	5.8	285.8417	94703.1843	1.46	288	26.8	R7	7	-21.78	21.09	1.5	-2.93	-22.27	-45.75	16.2	1.48

Figure 2: Format of the csv file to be input into the new atmospheric tomography software. The first row contains the variable names, the second row contains the units which the variable should be entered in, and final four rows contain observations. The variables `x1` and `y1` specify the locations of the lasers or point towers, and the variables `x2` and `y2` specify the locations of the reflectors where applicable.

2. The package `lubridate` was used to put the "date-time" variable into the same format for each data frame. From there, observations were matched by their date and time using the command `left_join`. At this stage, we now had a single data frame with one row for each unique date/time entry, and a column for each instrument with methane concentrations corresponding to the date/time entries.

3. The weather files were loaded into `R`, and again, the "date-time" variable was put into the same format as those we already had, and observations were matched by date/time. This meant that we now had values for the wind speed, wind direction, temperature, and air pressure for each date/time entry.

4. Observations (rows in the data frame) were removed if they did not have a value for the wind speed, wind direction, air temperature, or air pressure. This was because the inversion software cannot run without these four values.

5. The command `gather` from the package `tidyr` was used to bring the data frame from having one column of methane concentrations for each instrument (heading for the column was the name of the instrument), to having one column for all of the methane concentrations, and one column for all of the instrument names. So, for any date/time entry, there were now multiple entries with the same date/time entry; one for each instrument. This allowed for each concentration and corresponding instrument name to be recorded for that date/time. In this step, we have brought the methane concentrations from a "wide" format to a "long" format.

6. Any rows in the data frame which were missing a methane concentration value were now removed, along with any rows missing an L-value. If the inversion is to be run on data for which the methane release rate is known, any row not containing a release rate should also be removed. Again, this is because these values are needed in order to filter the data before running the inversion software.

7. It was noted upon plotting the concentrations of upwind observations that some of the upwind concentrations for our tower instruments were very large. Thus it was decided to remove any values which, for each particular tower instrument, were larger than the median concentration of that tower plus three median absolute deviations.

8. Finally, a new data frame containing only the columns needed for the csv file which is given to the atmospheric software (that in Figure 2) was created. This data frame was written to a csv file, and this csv file became the input file for the software.

7 Customizing other parameters

The following parameters are likely to remain unchanged across multiple data sets:

1. `samples`: The number of samples to be taken along the line from the laser to the mirror. The default value is 100. If the path is very long, you might want to increase this number to improve accuracy. To change this number, go to line 30 of the "atmospheric-tomography.py" script.

2. `angle`: observations which have its tower/laser-reflector path within ± angle of the plume's centreline are considered to be downwind. This value is currently set to 10 degrees. To change this value, go to line 17 of the "background-estimation.R" script.

3. `wind_speed_cutoff`: Observations with wind speeds lower than this value are removed from the data set. Low wind speeds tend to create noise and interfere with the Bayesian framework. This value is currently set to 0.5 m/s. To change this value, go to line 18 of the "background-estimation.R" script.

8 Atmospheric Tomography Scripts

Load the command line interface (Terminal utility on Mac and Linux, DOS prompt on Windows) and first ensure you are working out of the folder in which the atmospheric tomography scripts are saved. For instance, on my machines (Mac/Linux), since the source code is in the path `Home/Dropbox/atmospheric_tomography_laser_2018/src`, I run the following command in Terminal to set my working directory:

```
> cd ~/Home/Dropbox/atmospheric_tomography_laser_2018/src
```

The script to be executed is atmospheric-tomography.py. The argument to the script is a prefix for the output files (explained below). Ensure a value is set for this argument, or the software will produce an error. Below we show an example set of inputs given to the software, after which we explain each of the inputs.

```
> python src/atmospheric-tomography.py methane-inversion

Enter the number of iterations for the MCMC simulation: 100000

Enter the burn in for the MCMC simulation: 20000

Enter the thining variable for the MCMC simulation: 10

[?] Which setup?: June 8 -- June 12
April 23 -- June 7
> June 8 -- June 12
Other

How many instruments are you considering? 4

Enter the instrument numbers to consider: 20 21 22 23

[?] Which background method?: Upwind-downwind
```

```
30 min averaging
> Upwind-downwind

[1]
Processing data

[1]
Starting background estimation

[1]
25% done

[1]
50% done

[1]
75% done

[1]
Background estimation complete

[-------------    36%                  ] 1093 of 3000 complete in 0.5 sec
[-----------------69%------            ] 2083 of 3000 complete in 1.0 sec
[-----------------100%-----------------] 3000 of 3000 complete in 1.5 sec
```

As the above output shows, the script will ask the user for a number of inputs:

- The first three inputs are for the number of iterations for the MCMC simulation, the burn-in value and the thinning variable (see Section 8.1).

- The next input asks which instrument setup you want to consider. There are three options to choose from. One of the first two options should be chosen if you want to consider data from the Ginninderra experiment, and the last option should be chosen if you want to consider a different set of data.

- If the last option is chosen in the previous step, up to six further prompts will appear. The first will ask if you know the true release rate, and if so, another prompt will ask you to give the rate. Then, you will be asked to give the name of the data file (include the .csv extension, and do not put the name in quotation marks), the total number of instruments in the experiment (even those you do not wish to consider in the current instance of the inversion), the height of the gas source, and the molar mass of the gas you are considering (e.g. 16.04 g/mol for methane).

- The next two inputs ask how many instruments you want to consider in the inversion, and then what instrument numbers they are. Enter the instrument numbers with a single space between each number only, no commas, brackets, etc. If you want to consider all instruments involved in the experiment, you can alternatively type "all".

- The final input asks which background estimation method you want to consider (see Section 8.2 for details on these methods).

8.1 Number of iterations, burn-in and thinning variable

8.1.1 Number of iterations

A (Gibbs sampling) MCMC simulation proceeds by sampling the parameter of interest, one at a time from their full conditional distributions. Each parameter sample is dependent on the most recent samples of the other parameters. The number of iterations is the number of times that each parameter is sampled. The trade-off here is between efficiency and accuracy — choosing a very high value (say 10 million) would cause the script to run for a much longer time or even crash. We suggest that you use a moderately high value (we generally use 100000) and inspect the output. If the output is not satisfactory (i.e., shows high levels of auto-correlation, trace plots which appear to get "stuck" on one value, or histograms which are very odd in shape), you can re-run the analysis with a different number of iterations.

8.1.2 Burn-in

Burn-in is the number of initial samples that are discarded. There is a lot of discussion on the value to choose for this variable. In our example, we have been possibly a little overly cautious and chosen a burn-in value of 20000.

8.1.3 Thinning Variable

Thinning is used to reduce the auto-correlation between the samples from MCMC. A thinning variable set to 2, for example, instructs the software to discard every other sample. The default is 1 (no thinning). If you set this to a higher number, consider increasing the number of iterations. Alternately, you can choose to not use thinning and run the simulation longer by increasing the number of iterations. We generally set our thinning variable to 10.

8.2 Background estimation methods

The concentrations of methane measured during the experiment are of course not necessarily absent from external sources of methane, and thus the pre-existing or background concentrations of methane (that are time and wind-direction dependent) need to be estimated and subtracted off the measured concentrations. The background estimation component of the inversion is automatically carried out in R. The user needs to select one of the two available background estimation methods described below.

8.2.1 30 min averaging

This method first involves sorting all observations into 30-minute time blocks. Then, we take the average of the five lowest measured concentrations in each 30-minute block, and use this value as the estimated background concentration for all observations within that time period.

Caution should be taken when selecting this method with a low number available observations in each 30-minute block, as there is an increased risk of using measurements taken in the path of the gas plume as part of the background estimation.

8.2.2 Upwind-downwind

The overall approach to this method of background estimation is as follows:

- Determine which observations are *upwind* of the plume, and which observations are *downwind* of the plume.

- For each downwind observation, search for all upwind observations recorded by the same instrument, within a two-hour time interval centred on the downwind observation in question.
- If less than five upwind values can be found, extend the time interval to four hours, then to six, eight, and ten if necessary.
- If still five upwind observations are not found, begin the search again from the two-hour interval, using all instruments. Continue extending the time interval until at least five upwind observations are found. (If there are less that five upwind observations in the entire data set, the software will produce an error and stop.).
- The average of all upwind values found then becomes the estimated background concentration for the observation in question. This value is then subtracted from the observation, forming the new background-corrected observation to be used in the inversion.

Further details of this method are discussed in Section 7 of Cartwright (2018). An immediate advantage to this method of background estimation is that we are able to remove all of the upwind observations from the data set once the background concentrations are determined. This is because these observations do not actually offer us any useful information beyond the background estimation phase, as they are not in the path of the released gas. This will speed up the MCMC component of the inversion.

8.3 Output from the software

At the end of the execution, two output files are produced: summary.csv, and plots.pdf. These files names are prefixed with the argument given to the "atmospheric-tomography.py" script. So for this example, since we entered

```
> python src/run-tomography.py methane-inversion
```

the output files are: methane-inversion-summary.csv, and methane-inversion-plots.pdf. The summary file contains the means, standard deviations, 95% highest posterior density intervals and the quantiles for the parameters Q and τ of the model. Where the upwind-downwind background estimation method has been chosen, this file also contains a total count of all upwind observations, and total count of all downwind observations. You can view these values by either opening the csv file, or by typing

```
> cat methane-inversion-summary.csv
```

into the command line.

The script also produces a pdf file containing six plots. Three are for the posterior probability distribution of the source emission rate Q (top row), and three other for the posterior probability of the precision τ (bottom row). The left panels show the histograms of the empirical posterior distributions. Where the true value of Q is known, the true value will be marked by a red dashed line on the histogram of Q. In the middle panels are the trace plots of Q and τ from the MCMC run. In the right panels are the auto-correlation plots. The initial part of the graph may display a high degree of auto correlation but it should reduce nearly to zero by the second or third lag. If you see high auto-correlation values throughout the simulation, consider running the simulation with a thinning variable set to a higher number. Figure 3 shows an example of these plots.

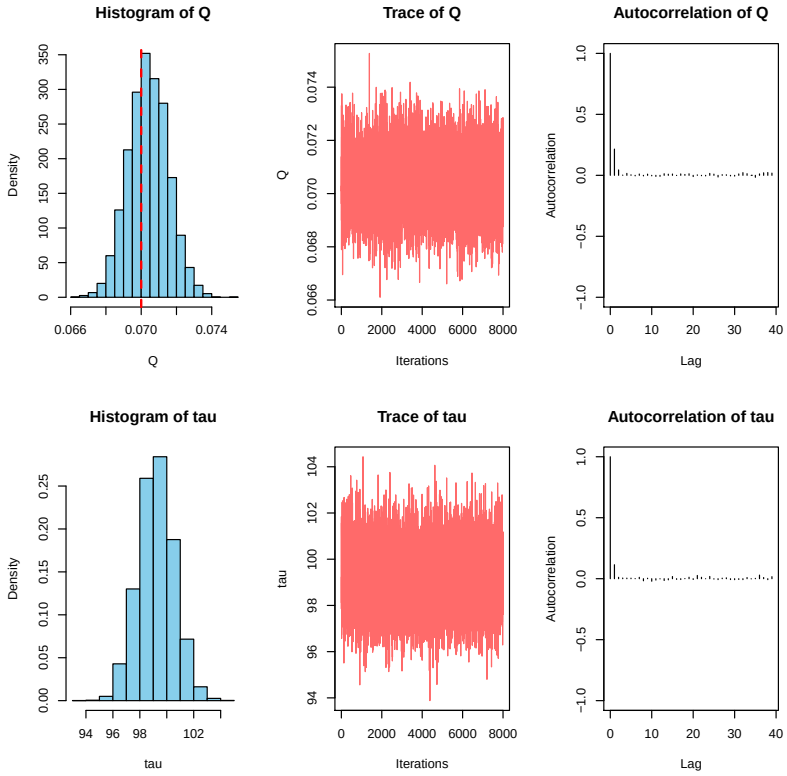

Figure 3: The six plots produced by the atmospheric tomography software. The left panels show histograms of the empirical posterior distributions for Q (top), and τ (bottom). The red dashed line is the true value of Q. The middle panels show the trace plots for Q (top), and τ (bottom). The right panels are the auto-correlation plots for Q (top), and τ (bottom).

8.4 Warning and error messages

If there are a very low number of observations to be used in the inversion (50 or less), the software will print a warning, and tell you how many observations you are using in the inversion. Similarly, if the upwind-downwind method of background estimation is selected, the software will print a warning when the total count of upwind values or total count of downwind values are low (40 or less upwinds, and 30 or less downwinds).

If, after filtering the data set for only the instrument numbers you wish to consider, there are no data values left, the software will stop running and print an error message telling you it has no data values to consider.

8.5 Further reading

There are a number of resources available which provide further information on Bayesian statistics and MCMC. We suggest Gelman et al. (2013), Gilks et al. (1995), and/or Wackerly et al. (2008).

9 Troubleshooting

1. *Cannot install Rpy2.*
   Ensure your version of R is 3.5 or higher.

2. *Cannot install python or PyMC.*
   Make sure you have administrator privileges on your machine.

3. *Script cannot find module PyMC.*
   Python finds the installed modules by referring to an environment variable called pythonpath. If you have installed the PyMC module correctly, you would find it in a directory called site-packages. Locate this directory in your filesystem and copy its path. On my machines (Mac/Linux) for example, I add the following line to the file .bash_profile (located in your home directory):

   ```
   export PYTHONPATH=$PYTHONPATH:/home/anaconda/lib/python3.6/site-packages/
   ```

   Replace the path after the colon with the location of the site-packages directory on your filesystem. The instructions for modifying environment variables on a Windows machine are available at
   `https://superuser.com/questions/949560/how-do-i-set-system-environment-variables-in-windows-10`.

4. *Cannot find Python.*
   The operating system finds the python executables using the PATH environment variable. Locate the bin folder that contains the python executables and modify the PATH variable to include it. On my machines (Mac/Linux), this is done by editing the bash_profile on Mac/Linux using the following line:

   ```
   PATH="/Library/Frameworks/Python.framework/Versions/3.4/bin:${PATH}"
   export PATH
   ```

   If you need to change the PATH variables on a Windows machine, instructions can be found here: `https://www.java.com/en/download/help/path.xml`

5. *The results look all wrong.*
   While there are many reasons this can happen, do check that the units of all parameters are correct i.e., pressure is in pascals, temperature is degree Celsius etc. Note that the units are not to be specified in the input file. Make sure the concentrations are in ppm.

10 Citing this work

You can use the following bibtex entry for citing this work:

```
misc{AT2018,
author = {Cartwright, Laura and Bhatia, Sangeeta and Feitz, Andrew
and Zammit-Mangion, Andrew and Francis, Andrew},
title = {{A}tmospheric {T}omography {S}oftware {U}ser {M}anual},
year = {2018},
howpublished = {available from \url{https://github.com/GeoscienceAustralia/
atmospheric_tomography_point_and_line}}
}
```

Alternately, you can cite it as:

Cartwright, L., Bhatia, S., Feitz, A., Zammit-Mangion, A., and Francis, A. (2018). Atmospheric Tomography Software User Manual. Available from `https://github.com/GeoscienceAustralia/atmospheric_tomography_point_and_line`.

References

Bhatia, S. (2015). Atmospheric tomography: Bayesian inversion for quantification of point source emmissions. APR Intern project report.

Cartwright, L. (2018). Application of atmospheric tomography for methane leakage rate estimation. APR Intern project report.

Feitz, A., Schroder, I., Phillips, F., Coates, T., Neghandhi, K., Day, S., Luhar, A., Bhatia, S., Edwards, G., Hrabar, S., Hernandez, E., Wood, B., Naylor, T., Kennedy, M., Hamilton, M., Hatch, M., Malos, J., Kochanek, M., Reid, P., Wilson, J., Deutscher, N., Zegelin, S., Vincent, R., White, S., Ong, C., George, S., Maas, P., Towner, S., and Griffith, D. (2018). The Ginninderra CH_4 and CO_2 release experiment: An evaluation of gas detection and quantification techniques. *International Journal of Greenhouse Gas Control*, 70:202–224.

Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., and Rubin, D. (2013). *Bayesian Data Analysis*. Chapman & Hall/CRC, Boca Raton, FL, 3rd edition.

Gilks, W., Richardson, S., and Spiegelhalter, D. (1995). *Markov Chain Monte Carlo in Practice*. Taylor and Francis Ltd, London, UK.

Humphries, R., Jenkins, C., Leuning, R., Zegelin, S., Griffith, D., Caldow, C., Berko, H., and Feitz, A. (2012). Atmospheric tomography: A Bayesian inversion technique for determining the rate and location of fugitive emissions. *Environmental Science and Technology*, 46:1739–1746.

Jacobson, M. (2005). *Fundamentals of Atmospheric Modeling*. Cambridge University Press, New York, NY, 2nd edition.

Kuske, T., Jenkins, C., Zegelin, S., Mollah, M., and Feitz, A. (2013). Atmospheric tomography as a tool for quantification of CO_2 emissions from potential surface leaks: Signal processing workflow for a low accuracy sensor array. *Energy Procedia*, 37:4065–4076.

Thompson, R. and Stohl, A. (2014). FLEXINVERT: an atmospheric Bayesian inversion framework for determining surface fluxes of trace species using an optimized grid. *Geoscientific Model Development*, 7:2223–2242.

Turner, B. (1994). *Workbook of Atmospheric Dispersion Estimates*. Lewis Publishers, Boca Raton, FL, 2nd edition.

Wackerly, D. D., Mendenhall, W., and Schaeffer, R. L. (2008). *Mathematical Statistics with Applications*. Thompson Learning, Belmont, CA.

A Bayesian inversion using the simulated data set

This data set simulates a methane-release experiment over 40 days, involving three instruments: two lasers and one tower. The full set of inputs which should be given to the software when using this data set are as follows (though either background estimation method can be chosen):

```
will-scarlett:src Laura$ python3 atmospheric-tomography.py sim-data

Enter the number of iterations for the MCMC simulation: 100000

Enter the burn in for the MCMC simulation: 20000

Enter the thining variable for the MCMC simulation: 10

[?] Which setup?: Other
April 23 -- June 7
June 8 -- June 12
> Other

[?] Is the true release rate known?: Yes
> Yes
No

Enter the true release rate in grams per second: 0.07

Enter the name of the data file: simulated_data.csv

What is the total number of instruments in the experiment? 3

Enter the height of the gas source in metres: 0.35

Enter the molar mass of the gas in grams: 16.04

How many instruments are you considering? 3

Enter the instrument numbers to consider: all

[?] Which background method?: Upwind-downwind
30 min averaging
> Upwind-downwind
```

The plots produced are those shown in Figure 3.