Environmental Modelling & Software 27-28 (2012) 52–61

Contents lists available at SciVerse ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

openair — An R package for air quality data analysis

David C. Carslaw [a,*], Karl Ropkins [b]

[a] King's College London, Environmental Research Group, Franklin Wilkins Building, 150 Stamford Street, London SE1 9NH, UK
[b] Institute for Transport Studies, University of Leeds, LS2 9JT, UK

ARTICLE INFO

Article history:
Received 9 February 2011
Received in revised form
12 September 2011
Accepted 26 September 2011
Available online 20 October 2011

Keywords:
Air pollution
Dispersion modelling
Reproducibility
Open source software
R

ABSTRACT

openair is an R package primarily developed for the analysis of air pollution measurement data but which is also of more general use in the atmospheric sciences. The package consists of many tools for importing and manipulating data, and undertaking a wide range of analyses to enhance understanding of air pollution data. In this paper we consider the development of the package with the purpose of showing how air pollution data can be analysed in more insightful ways. Examples are provided of importing data from UK air pollution networks, source identification and characterisation using bivariate polar plots, quantitative trend estimates and the use of functions for model evaluation purposes. We demonstrate how air pollution data can be analysed quickly and efficiently and in an interactive way, freeing time to consider the problem at hand. One of the central themes of *openair* is the use of conditioning plots and analyses, which greatly enhance inference possibilities. Finally, some consideration is given to future developments.

1. Introduction

1.1. Background

Worldwide an enormous and growing amount of air pollution data is collected. In Europe for example, the *Airbase* database consists of over 7000 monitoring sites, which still only represent a fraction of the total amount of data available (http://www.eea.europa.eu/themes/air/airbase). These data are collected for many reasons including to provide information for testing compliance against air quality standards, guidelines and regulations, and for research purposes. Most data are probably collected to monitor progress towards meeting legislative limits, such as those imposed by the European Union. Much, if not most of the data are analysed in superficial ways e.g. to confirm whether an annual mean concentration is above or below a stated threshold value. This situation represents a considerable missed opportunity. Experience shows that considerably more information can be gleaned from the analysis of data through the application of innovative data analysis techniques.

There are however several barriers that prevent the widespread use of more insightful analysis. First, a coherent set of data analysis tools for air pollution purposes does not exist. Second, many users are not aware of the tools available or how to apply them. Third, even where these tools are available, they are often only accessible through a variety of expensive, specialist proprietary software, resulting in an inconsistent and fragmented approach to data analysis. If these barriers can be overcome, there are many potential benefits including: a more comprehensive evidence base to support decision making, identification of the factors controlling pollutant concentrations (including the discovery of unexpected influences), an improvement in the quality of analyses undertaken and the enhanced validation of environmental models. There are also potential economic benefits through better decision making and targeted actions to manage air pollution.

This paper describes the development of a set of air pollution analysis tools using the R statistical software (R Development Core Team, 2011). First we describe the principal aims of the software and provide some background information on how its design was approached. Second, we provide a series of examples of usage with the aim of showing how inferences can be drawn from data. These examples include importing air pollution data, discovering and characterising source characteristics, quantifying trends and applying the tools to model output to understand model performance. Other recent examples of work that does make use of air quality data analysis techniques to derive more insight includes van Velzen and Segers (2010); Appel et al. (2011). In the case of Appel et al. (2011) a range of analysis techniques have been developed to better understand regional air quality model performance, which is an issue we also address in the current paper.

* Corresponding author.
E-mail address: david.carslaw@kcl.ac.uk (D.C. Carslaw).

1364-8152/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.envsoft.2011.09.008

2. Software design and characteristics

2.1. Development aims

The software development had several aims that were to a large extent governed by the barriers discussed in Section 1.1. The first decision was to choose a development framework that would simultaneously allow a high level of numerical and graphical capability, that was cross-platform and which the tools could be made readily available internationally. Second, in order to ensure maximum uptake and participation in the project, there was a strong desire to use open-source software that could be freely distributed. The use of open-source software was important from a transparency perspective and it was thought that users and policy-makers would favour an open system where the code and techniques were open to full scrutiny. The other important aspect of using open-source software is that it is more likely that the international air pollution community would participate in its development and contribute to extending its capabilities.

2.2. The R project

R (www.r-project.org) is an open-source programming environment which is gaining rapidly in usage across a many wide range of disciplines (R Development Core Team, 2011). It is an interpreted language that offers excellent interactive analysis capabilities and is ideal for the rapid development of statistical and data analysis applications. One of the principal advantages of R for the *openair* project is that it is free and open-source; thus overcoming some of the barriers noted in Section 1.1. It is very robust and works on a wide range of platforms including Microsoft Windows, Apple OS X and Linux.

For air pollution purposes, R represents the ideal system with which to work. Core features such as effective data manipulation, data/statistical analysis, high quality graphics and visualisation lend themselves to analysing air pollution data. While R began and is probably best known as a statistical programming language, its strong capability of working with data in general means that it has a much wider impact, covering a very wide range of disciplines. The ability to develop one's own analyses, invent new approaches etc. using R means that advanced tools can be quickly developed for specific purposes. The use of R ensures that analyses and graphics are not constrained to "off the shelf" tools. These tools will often contain functionalities that are either part of the R base system or that exist through specific packages.

Another key strength of R is its package system. The base software, which is in itself highly capable (e.g. offering for example linear and generalized linear models, non-linear regression models, time series analysis, classical parametric and non-parametric tests, clustering and smoothing), has been greatly extended by additional functionality. Packages are available to carry out a wide range of analyses including: generalized additive models, linear and non-linear modelling, regression trees, Bayesian statistics etc. Currently there are over 2500 packages available and this number continues to grow. These packages are readily available through a global network of repositories called the Comprehensive R Archive Network (CRAN).

2.3. Openness and reproducibility

A key characteristic of our approach was to ensure the software was fully open to scrutiny. There were several reasons for this decision. First, it is likely in the longer term that *trustworthy* software will be developed if it is fully open to scrutiny. Second, an open system encourages other developers to contribute to the project. In the UK and Europe there is considerable reliance on proprietary software for environmental modelling and assessment. Many argue that proprietary models should be discouraged in environmental regulation, believing that those affected by decisions where environmental models and tools are used should be able to scrutinize them (NAS, 2007). The development of *openair* is very much in this spirit. The choice of R as the platform for development is also highly appropriate because R itself and all R packages are open-source.

Ensuring that all the examples in an R package run as expected is part of the system of package development and testing in R. Packages submitted to CRAN are required to go through extensive checks of both the code and documentation and to ensure all provided examples run as expected. These checks are made on a daily basis. This process ensures that all code runs exactly as intended. Package development also strongly encourages the use of comprehensive help files, thus formalising and structuring the different functions contained in a package.

Beyond the inherent checking processes there are other options to produce documentation that is entirely reproducible. We have adopted a system called *Sweave* for the comprehensive documentation that is available for *openair* (Leisch, 2002). Sweave is a form of literate programming (Knuth, 1983) that allows R code to be embedded in latex documents. These documents must be `run' to compile a final output — usually a pdf report containing all the outputs from an analysis such as plots, tables and statistics. The purpose is to create dynamic reports, which can be updated automatically if data or analyses change. The dynamic nature of the process is particularly useful for software that is under continual development. Rather than running code to produce an analysis e.g. a plot, which is then pasted into the documentation, the master document contains the R code necessary to produce it. When run in this way, all data analysis output (tables, graphs, etc.) are created during compilation and inserted into a final latex document automatically. The report can be easily updated if data or analysis change e.g. as modifications are made to the software, which allows for truly reproducible research. We have found that this process acts as another way of ensuring quality control of the code. It also ensures that users of the software can exactly reproduce all outputs as intended.

The *openair* project itself is developed using R-Forge (Theußl and Zeileis, 2009). R-Forge is a central platform for the development of R packages, R-related software and other projects. R projects are often developed on R-Forge before being published on CRAN. One of the key advantages for project development on R-Forge is the use of the version control system subversion, which offers a wide range of source code management tools (Pilato et al., 2004). Another benefit of R-Forge is that it allows other developers to contribute to code development, as is the case with *openair*.

The *openair* website at http://www.openair-project.org/ provides more information concerning the project and a comprehensive manual that supports the package.

3. Example of *openair* usage and capability

3.1. Introduction

This section demonstrates the usage of a few *openair* functions. A summary of most *openair* functions is shown in Table 1. While it is not possible to cover many functions in depth, the examples below highlight some of the underlying principles of usage. R itself works well in an interactive way e.g. an analysis is produced and the results from it suggest a refinement or point to the use of another function. Because the feedback to the user is almost immediate,

Table 1
Summary of *openair* functions. The column 'type' shows the maximum number of conditioning variables allowed by each function. There are several other utility functions available that are not shown in this table but are described in the help files of the package and in the manual available at http://www.openair-project.org/.

Function	Purpose	Multiple pollutants	Type option
`calcFno2`	estimate primary NO_2 emissions ratio from monitoring data; see package help file for details and Carslaw and Beevers (2005)	no	no
`calendarPlot`	calendar-type view of mean values	no	no
`conditionalQuantile`	for model evaluation and air pollution forecasting evaluation	no	yes [2]
`cutData`	partition data into groups for conditioning plots and analysis	yes	yes [≥1]
`importADMS`	import outputs from the ADMS suite of dispersion models (McHugh et al., 1997)	NA	NA
`importAURN`	import hourly data from the UK air quality data archive (http://www.airquality.co.uk/data_and_statistics.php)	NA	NA
`importKCL`	import data from King's College London databases (http://www.londonair.org.uk/)	NA	NA
`kernelExceed`	bivariate kernel density estimates for exceedance statistics	no	yes [1]
`linearRelation`	explore linear relationships between variables in time	no	limited
`MannKendall`	calculate MannKendal slopes and Sen-Theil slope estimates	no	yes [2]
`modStats`	calculate a range of model evaluation statistics	no	yes [≥1]
`percentileRose`	plot multiple percentiles by wind direction	yes	yes [2]
`polarAnnulus`	polar annulus plot for temporal variations by wind direction	yes	yes [2]
`polarFreq`	alternative to wind rose/pollution rose	no	yes [2]
`polarPlot`	bivariate polar plot (Carslaw et al., 2006)	yes	yes [2]
`pollutionRose`	pollution rose	no	yes [2]
`scatterPlot`	traditional scatter plots with enhanced options	no	yes [2]
`smoothTrend`	non-parametric smooth trend estimates	yes	yes [2]
`summaryPlot`	summary view of a data frame with key statistics	yes	no
`TaylorDiagram`	taylor Diagram used for model evaluation (Taylor (2001))	no	yes [2]
`timePlot`	flexible time series plotting	yes	yes [1]
`timeVariation`	diurnal, day of week and monthly variations	yes	yes [1]
`trendLevel`	level plots with flexible conditioning	no	yes [2]
`windRose`	traditional wind rose	no	yes [2]

a very large amount of useful analysis can be conducted quickly — freeing up time to think about the problem.

3.2. Data import

Importing data into R can cause users difficulties. Sometimes this difficulty is due to data stored in an ad-hoc basis rather then in a structured relational database. For this reason *openair* provides several functions to help users import data that ensures a format consistent for use in all other *openair* functions. Currently there are two main data sources that can be imported by *openair*: data from the UK Air Quality Archive and that from the King's College London air quality networks including the London Air Quality Network (LAQN). Between them these two data sources account for the vast bulk of hourly air quality information in the UK. Data are imported through the use of two main functions:

`importAURN` and `importKCL`.

These import functions download data from servers at either AEA plc or King's College London. The data itself is stored in the `.RData` format as annual hourly time series per site. `.RData` objects are compressed and are approximately four times smaller than .csv files and typically ten times smaller than Microsoft Excel files. It is therefore quick to download data over the Internet. Compared with 'conventional' ways of downloading data e.g. through http://www.airquality.co.uk/data_and_statistics.php, the *openair* functions are much simpler and more flexible. For example, there is no limit on the number of hours of data (or sites) that can be downloaded (limited to 64,000 rows through the archive data selection tools) and it is easy to select several sites or years at once. The help files for the functions provide the full site name and site code and these are used to help users choose which sites to select. For example, to download 10 years of hourly data from the Marylebone Road site in central London (site code 'MY1'):

```
mydata <- importAURN(site = "MY1", year = 2000:2009)
```

This command will import 87,672 rows of data for 11 different pollutants.

Sites can easily be combined e.g. to also import North Kensington data (site code 'KC1'):

```
mydata <- importAURN(site = c("MY1", "KC1"), year =
2000:2009)
```

The `importKCL` function has the additional capability of importing meteorological data including measurements of wind speed, wind direction, solar radiation, rainfall, atmospheric pressure and relative humidity. These data have been specifically compiled from several meteorological sites across London and aim to represent 'typical' conditions in and close the capital. The availability of high quality meteorological data for analysis of air pollution data greatly enhances the opportunities for insightful data analysis, but unfortunately is often difficult to obtain.

In *openair* there also exist a series of functions to import data from the ADMS suite of models (McHugh et al., 1997). There are two principal uses of accessing ADMS output data in *openair*. First, dispersion model predictions of concentrations can be analysed using all the techniques shown in Table 1 i.e. analysed in the same way as ambient measurements. There are several benefits of having both measurement and model predictions available. For example, techniques for source detection and characterisation applied to model output can inform dispersion modellers whether source characteristics such as plume buoyancy are correctly captured, or even whether a source is included in an emissions inventory. These types of analyses provide a greater opportunity to understand model performance beyond the use of simple numerical metrics that are most commonly used. In this way, dispersion modellers also have a greater opportunity to undertake analyses that provide information on the underlying processes that affect model performance.

Second, the output from the ADMS meteorological pre-processor provides access to many quantities that can usefully assist data

analysis — either of model output data or ambient measurements. For example, the meteorological pre-processor provides estimates of boundary layer height, the Monin-Obukov length and many other useful properties of the atmospheric boundary layer. As it will be shown in subsequent sections, conditioning plots and analyses that allow data to be partitioned in various ways for more insightful analysis are greatly enhanced by the availability of these other variables. For example, understanding the directional and wind speed dependence of a concentration dependent on boundary layer height or atmospheric stability can be very informative e.g. assisting with source identification.

3.3. Bivariate polar plots

Bivariate polar plots have proved to be extremely valuable for identifying and understanding sources of air pollution (Carslaw et al., 2006; Westmoreland et al., 2007). These plots show how the concentration of a pollutant varies by wind direction and wind speed at a receptor. Working in polar coordinates helps to understand the directional dependence of different sources; particularly when more than one monitoring site is available. The wind speed dependence of a source can provide important information concerning the source type and characteristics (Carslaw et al., 2006; Jones et al., 2010). Jones et al. (2010) considered a wide range of situations and species including chloride and nitrate concentrations as well as local-scale urban concentrations of primary pollutants and developed analytical expressions to describe the wind speed dependence. Carslaw et al. (2006) considered the specific situation of aircraft jet plumes in some detail and showed how the wind speed dependence of concentrations made it possible to detect aircraft plumes at receptors and the extent to which plume buoyancy affects dispersion. In urban situations bivariate polar plots have proved very useful in understanding the complexities of dispersion in street canyons and around junctions (Tomlin et al., 2009; Westmoreland et al., 2007). These plots are a useful first step in data analysis because they provide a good overview of different source types. A subset of information contained in these plots can be extracted and analysed using other functions e.g. to quantify trends or to understand temporal variations.

Fig. 1 shows the default bivariate polar plot for PM_{10} concentrations at the Thurrock site. The plot was created by:

```
polarPlot(tk1, pollutant = "pm10")
```

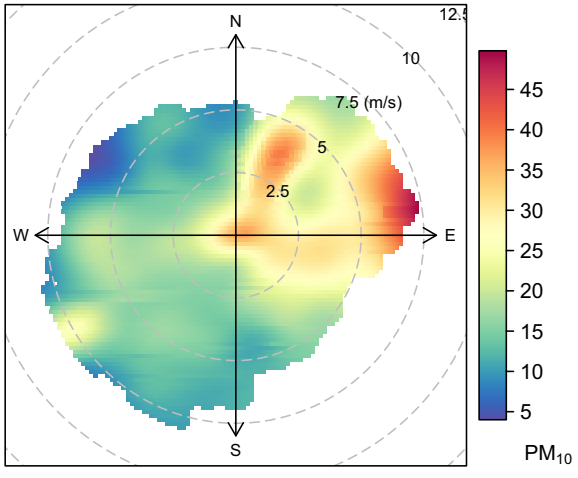

Fig. 1. Bivariate polar plot of PM_{10} concentrations (μg m^{-3}) at the Thurrock urban background site east of London. The centre of each plot represents a wind speed of zero, which increases radially outward. The concentration of PM_{10} is shown by the colour scale.

where `tk1` represents the imported data using `importKCL` over a 10 year period from 2000 to 2009 from the Thurrock background site.

Fig. 1 shows that there is evidence of increasing PM_{10} concentrations as the wind speed increases from the east and north-east. There is also evidence of a source to the NNE, for a range of wind speeds. These two features are the most obvious and could be analysed further by extracting the wind speed and wind direction ranges that they cover. Other analysis could then be undertaken to explore the characteristics of these features in more detail e.g. by considering diurnal or day of the week influences on PM_{10} concentration.

The basic bivariate polar plot has been developed further in *openair* to provide a set of functions that allow much greater flexibility of use, as will be illustrated next. An example of enhanced analysis using bivariate polar plots is shown in Fig. 2. We take the opportunity to demonstrate 'conditioning' also. The plot itself was produced by:

```
polarPlot(tk1, pollutant = "pm10", type = c("season",
"weekend"))
```

Fig. 2 shows the PM_{10} concentration dependence on season and whether the day is a weekend or weekday, using the `type = c("season", "weekend")` option of the function. This way of splitting the analysis by different variables (called conditioning) is extremely powerful and greatly extends the analysis opportunities. For this reason, we have made conditioning a core component of *openair* and also made it straightforward to use, which is discussed briefly next.

Understanding how one variable (e.g. a pollutant concentration) depends on the value of one or more other variables is essential if the aim is to understand the influences of the variable in question. In *openair* there are several in-built ways of conditioning data; two of which were used in the `polarPlot` call above ("season" and "weekend"). Other types include "hour" (for hour of the day dependence), "year" and "weekday" (all seven days of the week), for example. Moreover, as in the example used here, two conditioning variables can be used. In addition, account can be taken of any other variable in a data set. For categorical variables the categories are used directly to condition the data. If the data are numeric, *quantiles* of the data are calculated to provide four approximately equal size categories of values. This approach extends beyond using readily available directly measured quantities. For example, plotting the dependence of a concentration on variables such as atmospheric stability or boundary layer height can again provide useful insight. Many of these other variables could for example be derived from a meteorological pre-processor used as part of dispersion modelling.

So what sort of information does Fig. 2 contain? The first characteristic to note is that there is a wind speed and direction dependence on the concentration of PM_{10}. If one considers the seasonal dependence first (by considering each of the four columns), there is some interesting variability. Concentrations are generally higher for higher wind speed conditions in spring and winter — shown by the elevated PM_{10} concentrations to the east. However, the patterns of concentration are different. For high wind speeds (>about 5 m s^{-1}) there is more evidence of high PM_{10} concentrations during the springtime than winter. Subsequent analysis shows that these concentrations are strongly controlled by springtime episodes of transboundary secondary particulate matter. These features are not revealed by the basic polar plot shown in Fig. 1.

There is some evidence of a discrete source to the NNE in some of the panels; particularly autumn and winter, but not summer. The location of this site is close to various small industrial sources, which are potential origins of elevated PM_{10} concentrations. There are

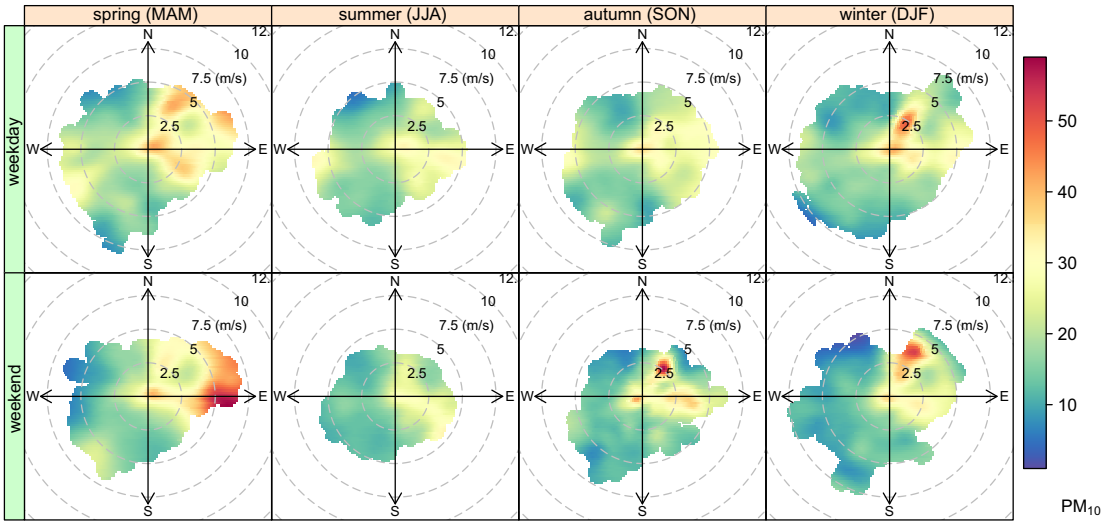

Fig. 2. Bivariate polar plot of PM_{10} concentrations (μg m^{-3}) at the Thurrock urban background site east of London. The centre of each plot represents a wind speed of zero, which increases radially outward. The concentration of PM_{10} is shown by the colour scale. The plot has been 'conditioned' by two variables: season and whether the day is a weekend or weekday.

several potential origins of the discrete PM_{10} source shown in Fig. 2 that could be investigated further. For example, a stack emission of PM_{10} could provide a high wind speed signature. Alternatively, spoil heaps and similar sources could represent a source of wind-blow suspension. The weekend-weekday differences can often reveal information about source activities e.g. whether concentrations tend to be higher during the working week than at weekends. In Fig. 2 it seems that in autumn for example, there is more evidence of a source at weekends rather than weekdays, which could be investigated further by considering potential nearby sources.

On its own, this type of analysis can help narrow-down the number of likely source origins and characteristics, but does not always provide a definitive answer. For this reason it is important to analyse the data in other ways. A subset of wind speed and direction conditions can be used to isolate various features for further analysis. For example, one might now choose to consider day of the week and diurnal variations of a subset of conditions informed by Fig. 2 — using the `timeVariation` function in *openair*. The temporal variations of the various features could help show for example that a feature tends to vary in a distinct way that strongly suggests an industrial source. The interactive nature of R works well in this respect because one analysis informs the next and so on. This also means that while a single analysis may not produce a definitive answer, a body of evidence can greatly increase the extent to which inferences can be drawn from data.

Although not shown, similar plots to Fig. 2 for other pollutants can be very revealing and help to better understand the various dependencies. Are the features shown in Fig. 2 also observed for NO_x for example, which would help to provide evidence of a combustion source? Similarly, a plot of SO_2 could narrow the analysis down further still e.g. if a PM_{10} source is also a clear source of SO_2. The extent to which these analyses can be undertaken will of course depend on the availability of measurements of other species. Sometimes it may be that a nearby site offers some further insights. For example, if a nearby site does not show similar bivariate polar plot features, then it may suggest that many of the observed sources are very local in nature. Similarly, removing background concentrations and then undertaking the analysis can considerably improve knowledge of local sources.

The `polarPlot` function (and the related functions `polarFreq` and `polarAnnulus`) have many other options not discussed here but which provide fine control over the analysis. These options include how missing data are treated, the estimate of smooth surface fitting (and the fitted surface uncertainties) etc. Full details are provided in the *openair* package help files.

3.4. Trend analysis

Understanding and quantifying trends is very important in air pollution and *openair* provides two main functions for this purpose: the `MannKendall` and `smoothTrend` functions. In this section we provide an example of the `MannKendall` function, highlighting the benefits of using R as the basis of the function development and illustrating how conditioning can provide a richer source of trend information for air quality data.

The non-parametric Mann–Kendall approach to trend detection is widely used in environmental sciences e.g. see Hirsch et al. (1982); Helsel and Hirsch (2002). The `MannKendall` function adopts the basic approach to trend detection with several enhancements. First, bootstrap techniques are used to provide and estimate of the uncertainty (Efron and Tibshirani, 1993). Because air quality data are often auto-correlated we have implemented a basic block-bootstrap technique to take account of autocorrelation (Kunsch, 1989). Second, the slope estimate and uncertainty in the slope uses the method of Sen (1968); again using bootstrap and block-bootstrap methods for quantifying slope uncertainties. R provides excellent built in facilities or add-on packages for these types of statistical calculations.

In its default use the `MannKendall` function calculates the slopes and relevent statistics for monthly mean estimates. Because many air pollutants exhibit strong seasonal variations, *openair* allows the user to apply seasonal trend decomposition to the data to remove seasonal effects before the slope estimates etc. are calculated. Use is made of the STL technique (again available in R) described by Cleveland et al. (1990).

We provide an example of the `MannKendall` applied to O_3 concentrations at the London Teddington site in a rural setting in south-west London. The commands below show how the air pollution data was imported and the plot produced. Note that meteorological data were separately added because these data are not freely available. Data have been analysed from 1996 to 2009 as annual means. We also take the opportunity to apply conditioning to understand how trends vary by wind sector. In the call below the data ted represents the hourly time series of all pollutants at the London Teddington site from 1996, imported using the importKCL

function. The `type = "wd"` option tells the function that we want to condition by wind sector i.e. have a separate panel for each sector. Finally, `avg.time = "year"` sets the averaging period to a calendar year.

```
ted <- importAURN(site = "ted", year = 1996:2010)
MannKendall(ted, pollutant = "o3", type = "wd",
avg.time = "year")
```

The results are shown in Fig. 3. The first thing to note is that *openair* recognises the conditioning variable is wind direction and sets the panels out in a logical order e.g. with the north (337.5–22.5°) at the top and so on, which aids interpretation. Each panel contains key information relating to the trend estimates. Taking the north wind sector as an example, the solid red line shows the best estimate of the Sen-Theil slope with the dashed lines showing the upper and lower 95% confidence interval estimates. The text provides a numerical summary of the slope estimate, which in this case shows the trend to be +0.86 μg m^{-3}/year with the lower and upper 95% confidence interval from −0.09–1.51 μg m^{-3}/year. The '*' symbol shows the slope estimate is statistically significant at the $p = 0.10$ confidence level.

Conditioning the data by wind sector provides some useful information about O_3 trends that would not have been discovered if a single plot of all data was produced. For example, Fig. 3 shows that there is little evidence of any change in O_3 concentrations for southerly wind sectors over the past 15 or so years. By contrast there is stronger evidence that O_3 concentrations have increased for most other wind sectors (particularly for easterly and north-easterly winds). The stronger increase in positive O_3 trends for the N and NE sectors can be attributed to the location of this site in relation to London. The main bulk of the London conurbation is to the east/north-east of this site. It can be shown that concentrations of NO_x from these wind sectors have decreased. Because O_3 reacts with NO to form NO_2, the reduction in total NO_x concentrations for these wind sectors has reduced the depletion of O_3 through its reaction with NO. Therefore the reduction in NO_x concentrations has resulted in an increase in O_3 concentrations from those wind directions.

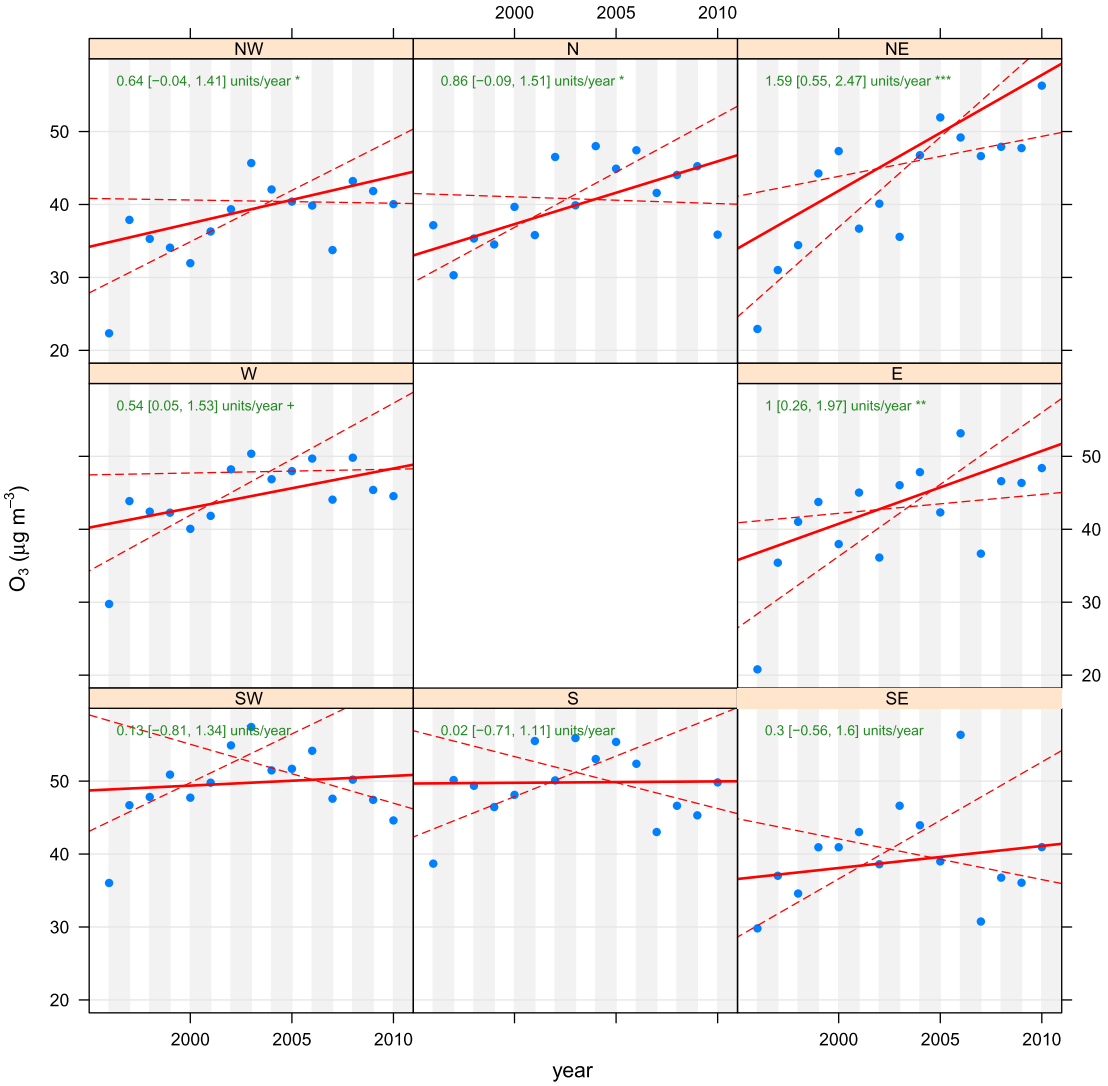

Fig. 3. Trend estimates of O_3 concentrations at the London Teddington site conditioned by wind direction (μg m^{-3}). In each panel the trend line is shown by the solid line and the 95% confidence intervals by the dashed line. The text in green shows the trend estimate together with the lower and upper 95% confidence intervals in the slope. Note also that the symbols shown next to each trend estimate relate to how statistically significant the trend estimate is: $p < 0.001 = ***$, $p < 0.01 = **$, $p < 0.05 = *$ and $p < 0.1 = +$. The absence of a symbol means that there is no statistically significant slope detected. Note that there can appear to be occasional inconsistencies between the Mann–Kendall test for trend (from which the p values are derived) and the uncertainties in the slope estimate though the Sen-Theil calculations, in part due to bootstrap re-sampling.

Although not shown in this paper, conditioning by other variables can yield further information about trends. For example, day of the week and hour of the day trends can provide information about how changes to emission sources affect trends. Seasonal trends may reveal more regional influences such as the effect of European summertime O_3 episodes or springtime northern hemispheric maxima (Monks, 2000). Similarly, in *openair* it is straightforward to consider statistics other than the mean e.g. high percentile values.

For a more flexible approach to trend estimates the `smoothTrend` function provides a non-parametric smooth trend estimate. The `smoothTrend` is useful in cases where trends in time are clearly non-linear and the shape of the trend provides useful information on the types of changes that have occurred. For example, if the concentration of a pollutant was stable over many years, underwent a decrease and then stabilised again, the `smoothTrend` function would help reveal these characteristics.

3.5. Model evaluation

Evaluating model performance is an increasingly important issue due to the increasing complexity of models. Model evaluation can take many forms from simple numerical metrics to sophisticated approaches using advanced statistical techniques. Evaluating model performance shares many of the challenges posed by understanding air pollution measurement data and in this respect most of the tools developed in *openair* can be directly and usefully applied to model output.

The model evaluation process is described in detail by Dennis et al. (2010). They distinguish between several types of evaluation: *Operational evaluation* in which model predictions are compared with data in an overall sense using a variety of statistical measures; *Diagnostic evaluation* in which the relative interplay of chemical and physical processes captured by the model are analysed to assess if the overall operation of the model is correct; *Dynamic evaluation* in which the ability of the modelling system to capture observed changes in emissions or meteorology is analysed; and, *Probabilistic evaluation* in which various statistical techniques are used to capture joint uncertainty in model predictions and observations. R is well suited to providing tools for all these types of evaluation and some examples are available e.g. Appel and Gilliam (2008). In this context, *openair* currently provides information related mostly to operational and diagnostic evaluation and to less of an extent, dynamic evaluation.

In this section we consider the `modStats` function. The function provides straightforward, commonly used numeric model evaluation statistics. These statistics include: fraction of predictions within a factor or two (FAC2), mean bias (MB), mean gross error (MGE), normalised mean bias (NMB), normalised mean gross error (NMGE), root mean squared error (RMSE) and the correlation coefficient, *r*. As input the `modStats` function requires a vector of measured and modelled values. These statistics are defined in Appendix A.

In common with almost all other *openair* functions, full conditioning is made simple. For example, model evaluation may include models from several groups and many receptors and the summary evaluation statistics were required by these two grouping variables. In this case the function would be called like:

```
modStats(thedata, mod = "mod", obs = "o3",
  type = c("model", "receptor"))
```

which assumes there are columns representing the model used, the receptor together with observed values (`"o3"`) and modelled values (`"mod"`).

To extend the possibilities here it is again possible to use built in or other conditioning variables. For example, it may be of interest to know how well models perform at each site split by season. In this case `type = c("model", "receptor", "season")` would be chosen. Similar to other functions, any other numeric data field could be used for conditioning, which splits the variable into four quantile groups. This is useful to explore, for example, the dependence of model performance on boundary layer height or some other meteorological variable. This approach therefore helps to extend model evaluation beyond straightforward operational evaluation to diagnostic evaluation.

While the types of analysis just described provide a lot of useful information on model performance, they are limited to numerical outputs. A much richer source of information on model performance can be gained by applying many of the functions shown in Table 1. The application of many of these functions will more likely reveal *why* a model is not performing well and help point to the processes that affect poor predictions. For example, bivariate polar plots may show that the wind speed dependence for model predictions is wrong. Also, as we have experienced, these types of plots often reveal sources that are apparent from the analysis of measurements that are absent in modelled data. This situation can arise due to insufficient inventory information rather than poor model performance *per se*. Indeed, such information may provide grounds for excluding a subset of measurement data from the model evaluation (or adding a missing source to an inventory), or may be excluding certain receptors altogether. Similarly, these techniques provide valuable tools for model development.

As an example, we consider the use of the Community Multiscale Air Quality Modelling System (CMAQ, http://www.cmaq-model.org/) used to predict hourly surface O_3 concentrations at the rural Harwell site in south England for 2006. CMAQ is a sophisticated Eulerian model that is widely used internationally. In the first example we use the `modStats` function to explore how model performance varies by season. The function call is shown below and the results in Table 2.

```
modStats(CMAQ, obs = "measured", mod = "modelled")
```

Table 2 shows that the large majority of predictions are within a factor of two — ranging from 0.70 in winter to 0.85 in spring. The mean bias reveals a much stronger positive bias in the results in autumn and winter compared with spring and summer. Table 2 also shows that the correlation coefficient, *r*, is considerably higher in winter months than other months. These results can easily be explored in more detail by changing the conditioning variable through the type option e.g. to explore diurnal or day of week performance.

Recent additions to *openair* include the Taylor Diagram Taylor (2001), through the `TaylorDiagram` function, which is widely used for model evaluation purposes (Taylor, 2001). The *openair* version allows for full conditioning to easily help understand model performance in flexible ways. Additionally, `TaylorDiagram` also has the ability to show model performance changes between two model runs e.g. two different models versions or set ups.

Table 2
Summary model evaluation statistics comparing observed and modelled hourly O_3 concentrations using CMAQ applied to the Harwell measurement site in south England. These statistics include: fraction of predictions within a factor or two (FAC2), mean bias (MB), mean gross error (MGE), normalised mean bias (NMB), normalised mean gross error (NMGE), root mean squared error (RMSE) and the correlation coefficient, *r*. The results are conditioned by season.

Season	FAC2	MB	MGE	NMB	NMGE	RMSE	*r*
Spring (MAM)	0.85	4.42	18.02	0.07	0.29	23.89	0.31
Summer (JJA)	0.84	4.40	23.14	0.07	0.38	29.57	0.38
Autumn (SON)	0.80	15.86	20.33	0.32	0.41	26.05	0.34
Winter (DJF)	0.70	11.63	18.46	0.25	0.40	24.49	0.64

Model performance can easily be visualised and quantified in other ways. In this example we use the `timeVariation` function to consider the measured and modelled temporal variations. In the function call below the data (`CMAQ`) consists of a date-time column, and measured and modelled hourly O_3 concentrations.

Fig. 4 clearly shows in the monthly variation plot the tendency of the model to over-predict O_3 concentrations in the autumn and winter, which were numerically quantified in Table 2. The diurnal plot (lower left-hand plot shown in Fig. 4) shows that the peak O_3 concentration is captured well by the model at around 15:00. However, there is also evidence that nighttime concentrations are not captured very well as they are over-predicted by a significant margin. The day of the week plot (bottom right-hand plot) shows that measured O_3 concentrations tend to be higher at weekends, which at this site is mostly due to lower concentrations of NO_x. However, there is little evidence of higher weekend concentrations in predicted O_3. Fig. 4 might therefore suggest that NO_x concentrations at this receptor are important and worthy of further investigation. A further analysis of the results also suggests the meteorological model tends to over-estimate surface wind speeds and this too would explain much of the model-observation difference. Note that almost all *openair* functions output the data used to make the plots so that users can post-process the data should they wish.

```
timeVariation(CMAQ, pollutant = c("measured", "modelled"))
```

While there is a lot of useful information in Fig. 4, more insight can again be gained through conditioning. For example, splitting these results by quantiles of wind speed can help show whether deposition processes are important, or by quantiles of measured NO_x concentration, which may help reveal degraded model performance as NO_x concentrations increase. Considering concentrations of NO_x is useful because of the strong linkage between NO_x and O_3 concentrations in the atmosphere.

4. Future developments

The *openair* package to date has largely focused on data usage in the UK, but is widely used internationally despite only being available on CRAN for several months. There are clear areas where further development is needed. With respect to data import filters, we have found that the availability of easy to use, flexible functions in R greatly enhance the management of data. Among the benefits are that the format of the data are consistent with all *openair* functions and this avoids many issues of ad-hoc data input. For this reason, the extension of these facilities to other databases in Europe and further afield would be welcome. R itself has other packages such as the RNetCDF package for working with NetCDF files which are commonly used in atmospheric sciences and which work with *openair* (Michna, 2011).

In addition, it would be useful to define R *classes* for different data objects that capture other information such as the units used, the instrument type etc.; Chambers (2007) provides examples of benefits of using and defining classes. Such data could then be used to further automate plot annotations and, more importantly, provide more comprehensive information on the data itself.

However, the major area for future development is the addition of new capabilities to the package. There are clearly a very large number of techniques that could be made available. For example, in the scientific literature there are numerous methods that have been developed for source detection and characterisation, source apportionment, trend detection and so on. The availability of *openair* has also prompted the user community to suggest techniques such as meteorological normalisation i.e. accounting for meteorology in trends and more approaches to help answer policy relevant questions e.g. change detection. We would also like to make it easy to consider back trajectory analysis in a straightforward way e.g. by making it easy to import pre-calculated trajectories in a single step and to integrate existing techniques in *openair* to allow for deeper analysis based on air mass origin.

The lack of a graphical user interface (GUI) has been mentioned by some users as a disadvantage of R and *openair* in particular. Currently however, we have no plans to make a GUI available. While many users initially dislike the command line interface, in time most appreciate that there are benefits. Among the most significant benefits is that all commands can be recorded or saved

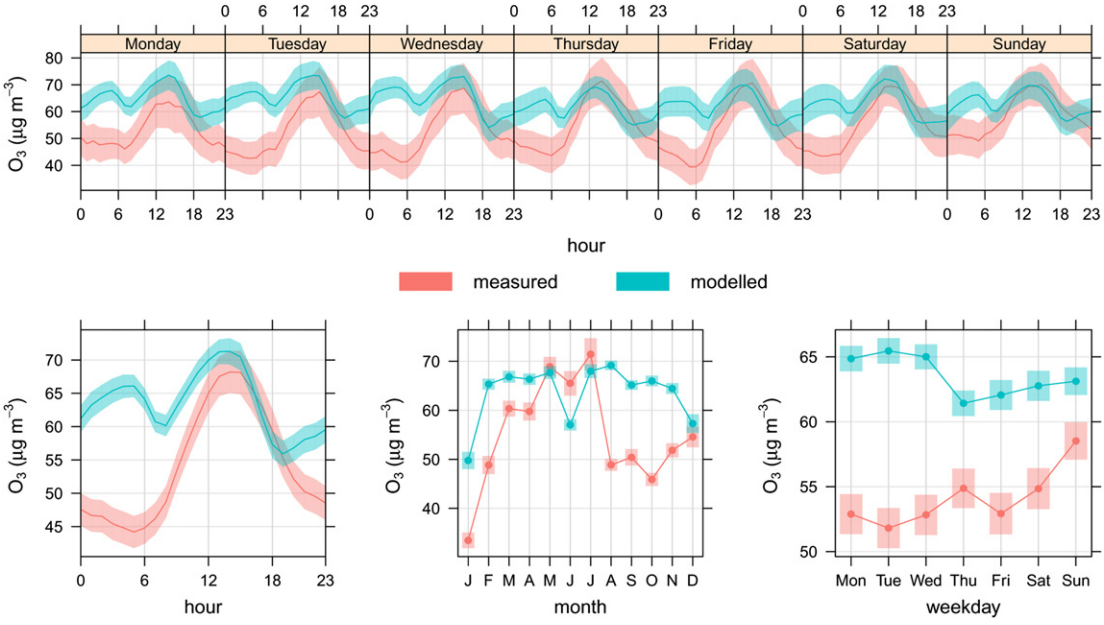

Fig. 4. Temporal variations is measured and modelled hourly O_3 concentrations at the Harwell measurement site in south England. The modelled concentrations are from CMAQ. the shading shows the 95% confidence intervals in the mean and assists the user in helping to understand the extend to which two or more quantities differ.

in a script file, which ensures that an analysis can be reproduced at a later date — perhaps with different data. This sort of reproducibility is not possible using a GUI. It should also be noted that *openair* functions work in a consistent way and once a user has learnt how to use one function, the rest work in a very similar fashion. Moreover, most analyses require very little user input, as demonstrated by the examples in this paper.

5. Software and data availability

The *openair* software is freely available as an R package. Details on installing R and optional packages including *openair* can be found at R Development Core Team (2011) and http://www.r-project.org. R will run on Microsoft Windows, linux and Apple Mac computers. No special hardware is required to run *openair* other than a standard desktop computer. Some large data sets or complex analyses may require a 64-bit platform.

Access to air quality data in the UK is possible through several *openair* functions and is also freely available. The data itself is stored on computer servers hosted by King's College London and AEA.

Acknowledgements

We would like to thank the Natural Environment Research Council (NERC) for funding the *openair* project (reference NE/G001081/1) and the Department for Environment and Rural Affairs (Defra) for additional support. We would also like to thank Dr Sean Beevers and Dr Nutthida Kitwiroon, King's College London for their ozone results from the CMAQ model.

Appendix A. Model performance evaluation statistics used in *openair*

In the following definitions, O_i represents the ith observed value and M_i represents the ith modelled value for a total of n observations.

Fraction of predictions within a factor or two, FAC2

The fraction of modelled values within a factor of two of the observed values are the fraction of model predictions that satisfy:

$$0.5 \leq \frac{M_i}{O_i} \leq 2.0 \tag{A.1}$$

Mean bias, MB

The mean bias provides a good indication of the mean over or under estimate of predictions. Mean bias in the same units as the quantities being considered.

$$\mathrm{MB} = \frac{1}{n}\sum_{i=1}^{N} M_i - O_i \tag{A.2}$$

Mean gross error, MGE

The mean gross error provides a good indication of the mean error regardless of whether it is an over or under estimate. Mean gross error is in the same units as the quantities being considered.

$$\mathrm{MGE} = \frac{1}{n}\sum_{i=1}^{N} |M_i - O_i| \tag{A.3}$$

Normalised mean bias, NMB

The normalised mean bias is useful for comparing pollutants that cover different concentration scales and the mean bias is normalised by dividing by the observed concentration.

$$\mathrm{NMB} = \frac{\sum_{i=1}^{n} M_i - O_i}{\sum_{i=1}^{n} O_i} \tag{A.4}$$

Normalised mean gross error, NMGE

The normalised mean gross error further ignores whether a prediction is an over or under estimate.

$$\mathrm{NMGE} = \frac{\sum_{i=1}^{n} |M_i - O_i|}{\sum_{i=1}^{n} O_i} \tag{A.5}$$

Root mean squared error, RMSE

The RMSE is a commonly used statistic that provides a good overall measure of how close modelled values are to predicted values.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (M_i - O_i)^2}{n}} \tag{A.6}$$

Correlation coefficient, r

The (Pearson) correlation coefficient is a measure of the strength of the linear relationship between two variables. If there is perfect linear relationship with positive slope between the two variables, $r = 1$. If there is a perfect linear relationship with negative slope between the two variables $r = -1$. A correlation coefficient of 0 means that there is no linear relationship between the variables.

$$r = \frac{1}{(n-1)}\sum_{i=1}^{n}\left(\frac{M_i - \overline{M}}{\sigma_M}\right)\left(\frac{O_i - \overline{O}}{\sigma_O}\right) \tag{A.7}$$

References

Appel, K.W., Gilliam, R.C., October 6–8, 2008. Overview of the atmospheric model evaluation tool (AMET). In: 7th Annual CMAS Models-3 Users' Conference (Chapel Hill, NC).

Appel, K.W., Gilliam, R.C., Davis, N., Zubrow, A., Howard, S.C., 2011. Overview of the atmospheric model evaluation tool (amet) v1.1 for evaluating meteorological and air quality models. Environmental Modelling and Software 26 (4), 434–443. http://www.sciencedirect.com/science/article/pii/S1364815210002653. URL.

Carslaw, D.C., Beevers, S.D., 2005. Estimations of road vehicle primary NO_2 exhaust emission fractions using monitoring data in London. Atmospheric Environment 39 (1), 167–177.

Carslaw, D.C., Beevers, S.D., Ropkins, K., Bell, M.C., 2006. Detecting and quantifying aircraft and other on-airport contributions to ambient nitrogen oxides in the vicinity of a large international airport. Atmospheric Environment 40 (28), 5424–5434.

Chambers, J.M., 2007. Software for Data Analysis: Programming with R. Springer, New York, ISBN 978-0-387-75935-7.

Cleveland, R.B., Cleveland, W.S., McRae, J., Terpenning, I., 1990. STL: a seasonal-trend decomposition procedure based on loess. Journal of Official Statistics 6 (1), 3–73.

AUG Dennis, R., Fox, T., Fuentes, M., Gilliland, A., Hanna, S., Hogrefe, C., Irwin, J., Rao, S.T., Scheffe, R., Schere, K., Steyn, D., Venkatram, A., 2010. A framework for evaluating regional-scale numerical photochemical modeling systems. Environmental Fluid Mechanics 10 (4), 471–489.

Efron, B., Tibshirani, R., 1993. An Introduction to the Bootstrap. Chapman & Hall.

Helsel, D., Hirsch, R., 2002. Statistical Methods in Water Resources. URL. US Geological Survey. http://pubs.usgs.gov/twri/twri4a3/.

Hirsch, R.M., Slack, J.R., Smith, R.A., 1982. Techniques of trend analysis for monthly water-quality data. Water Resources Research 18 (1), 107–121. iSI Document Delivery No.: NC504.

Jones, A.M., Harrison, R.M., Baker, J., 2010. The wind speed dependence of the concentrations of airborne particulate matter and nox. Atmospheric

Environment 44 (13), 1682–1690. URL. http://www.sciencedirect.com/science/article/B6VH3-4Y7P72C-2/2/f6c65e5f49ac3e9862d4c1803d4735c0.
Knuth, D.E., 1983. Literate Programming. Technical Report STAN-CS-83-981. Stanford University, Department of Computer Science.
Kunsch, H.R., 1989. The jackknife and the bootstrap for general stationary observations. Annals of Statistics 17 (3), 1217–1241.
Leisch, F., 2002. Sweave: dynamic generation of statistical reports using literate data analysis. URL. In: Härdle, W., Rönz, B. (Eds.), Compstat 2002 — Proceedings in Computational Statistics. Physica Verlag, Heidelberg, ISBN 3-7908-1517-9, pp. 575–580. http://www.stat.uni-muenchen.de/~leisch/Sweave.
McHugh, C.A., Carruthers, D.J., Edmunds, H.A., 1997. ADMS and ADMS-Urban. International Journal of Environment and Pollution 8 (3–6), 438–440.
Michna, P., 2011. RNetCDF: R Interface to NetCDF Datasets. R Package Version 1.5.2-2. URL. http://CRAN.R-project.org/package=RNetCDF.
Monks, P.S., 2000. A review of the observations and origins of the spring ozone maximum. Atmospheric Environment 34 (21), 3545–3561. iSI Document Delivery No.: 333VG.
NAS, 2007. Models in Environmental Regulatory Decision Making. Tech. Rep. National Academy of Sciences. report available from. http://www.nap.edu/catalog/11972.html URL. http://www.nap.edu/catalog/11972.html.
Pilato, C., Collins-Sussman, B., Fitzpatrick, B., 2004. Version Control with Subversion. Full book available online at. O'Reilly. http://svnbook.red-bean.com/.
R Development Core Team, 2011. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0. http://www.R-project.org/ URL.
Sen, P.K., 1968. Estimates of regression coefficient based on kendall's tau. Journal of the American Statistical Association 63 (324).
Taylor, K., 2001. Summarizing multiple aspects of model performance in a single diagram. Journal of Geophysical Research 106 (D7), 7183–7192.
Theußl, S., Zeileis, A., May 2009. Collaborative software development using R-Forge. The R Journal 1 (1), 9–14. URL. http://journal.r-project.org/2009-1/RJournal_2009-1_Theussl+Zeileis.pdf.
Tomlin, A., Smalley, R., Tate, J., Barlow, J., Belcher, S., Arnold, S., Dobre, A., Robins, A., 2009. A field study of factors influencing the concentrations of a traffic-related pollutant in the vicinity of a complex urban junction. Atmospheric Environment 43 (32), 5027–5037. URL. http://www.sciencedirect.com/science/article/B6VH3-4WNXV3Y-2/2/e71ecf623599006929c6899a22d49150.
van Velzen, N., Segers, A.J., March 2010. A problem-solving environment for data assimilation in air quality modelling. Environmental Modelling and Software 25, 277–288. URL. http://portal.acm.org/citation.cfm?id=1663658.1663969.
Westmoreland, E.J., Carslaw, N., Carslaw, D.C., Gillah, A., Bates, E., 2007. Analysis of air quality within a street canyon using statistical and dispersion modelling techniques. Atmospheric Environment 41 (39), 9195–9205.