# Technical note: Averaging wind speeds and directions

**Technical Report** · June 2014

DOI: 10.13140/RG.2.1.3349.2006

Stuart K Grange
Empa - Swiss Federal Laboratories for Materials Science and Technology
**46** PUBLICATIONS **934** CITATIONS

# Technical note:
# Averaging wind speeds and directions

Stuart K. Grange*[a,b]

[a]School of Environment, the University of Auckland, Auckland, New Zealand
[b]School of Population Health, the University of Auckland, Auckland, New Zealand

June 26, 2014

## 1 Background

Wind sensors are used in many air quality, climate, weather, and meteorological monitoring applications. If wind speeds and directions are to be averaged, for example when ten-minute values are to be presented as hourly averages, they need special treatment.[1] While it is acceptable for a data user to calculate the arithmetic mean of wind speed, this cannot be done for wind direction. I have seen many situations where the handling of wind direction has been incorrect and am often asked how to correctly average wind data. Although a quick search online will find many explanations and apparent solutions, the calculation steps are not clearly laid out or explained and are often wrong, for a good example see[2]. The main issue arises because wind direction is usually reported as an angle in degrees, 0–360 (or –359) where 0 or 360 represents a wind blowing from a northerly direction. If the wind direction is blowing from the north and traverses the discontinuity at the beginning/end of the circular scale, and then the arithmetic mean is calculated, this will result in the average wind direction to be somewhere in the southern quadrant. This is clearly incorrect. To correctly deal with this scale discontinuity, trigonometric functions must be used to handle the angles.[1]

Some of the confusion comes from that two separate sensors are used to represent wind behaviour, generally a wind vane and a spinning cup anemometer.[3,4] However, wind speed and direction are two components of the same quantity, *i.e.*, wind is a vector with both magnitude and direction. Wind is rarely represented as vector quantities in databases, websites, or other forms of communication outside specialised fields however. When averages of wind directions are made, the vectors need to calculated and are called 'wind components'. There are two wind components used for horizontal wind measurement, $\vec{u}$ and $\vec{v}$ representing the east-west and north-south components respectively.[3] Most data logging applications/devices which interface with wind sensors, only deal with the sensors' data in vectors. The last step of the processing is to convert the vectors into wind

*s.grange@auckland.ac.nz

speed and direction format because it is considered more user-friendly. Therefore, in the post processing procedure, we are often repeating what has already occurred but has been discarded during the measurement process.

### 1.1 Objectives

This technical note documents some of the wind speed and averaging formulas along with a working example which can be far more useful than the equations. There are a handful of accepted techniques to average wind direction and speeds.[1,5] They are all similar, but there are some subtle variations which are the choice of the user and some are slightly more appropriate in some situations. In my opinion, the most sensible versions of wind averages are the *scalar average wind speed* and the *resultant vector average wind direction*. This note prioritises these types of averages.

## 2 The equations

There are a number of equations used for wind speed and direction averaging.[1,5] For completeness, I have included them, but simply displaying equations is often not useful.

### 2.1 Scalar functions

The scalar average wind speed ($\bar{u}$) is displayed in Eq. 1 where $u_i$ is logged or reported wind speed. This is the 'standard' arithmetic mean of wind speed.

$$\bar{u} = \frac{1}{n}\sum_{i=1}^{n} u_i \tag{1}$$

### 2.2 Vector functions

Vector functions are used to average wind direction and can be used to compute a type of average wind speed which is different to Eq. 1. Before averaging can take place, the wind components ($\vec{u}$ and $\vec{v}$) must be calculated using Eq. 2 and Eq. 3. Wind direction ($\theta_i$) must be in degrees and the units for the components are radians. There are two other things to note: ($i$) these wind components here are calculated along with wind speed ($u_i$), *i.e.* the vectors are weighted by their magnitude, and ($ii$) the negative sign negates the direction. This negation is because wind direction, by meteorological convention, is defined from where the wind is blowing from, while the vectors define the direction where the flow is heading to.[3]

$$\vec{u} = -u_i \times \sin\left[2\pi \times \frac{\theta_i}{360}\right] \tag{2}$$

$$\vec{v} = -u_i \times \cos\left[2\pi \times \frac{\theta_i}{360}\right] \tag{3}$$

To calculate the resultant vector average wind speed ($\bar{U}_{RV}$) and direction ($\bar{\Theta}_{RV}$) Eq. 4 and

Eq. 5 are used:

$$\bar{U}_{RV} = \left(\vec{u}^2 + \vec{v}^2\right)^{\frac{1}{2}} \tag{4}$$

$$\bar{\Theta}_{RV} = arctan\left(\frac{\vec{u}}{\vec{v}}\right) + \texttt{flow} \tag{5}$$

$$\begin{aligned}\texttt{flow} &= +180\ for\ arctan\left(\frac{\vec{u}}{\vec{v}}\right) < 180 \\ &= -180\ for\ arctan\left(\frac{\vec{u}}{\vec{v}}\right) > 180\end{aligned}$$

The `arctan` function takes more than one form and is better explained with a working example because the `flow` clause can be resolved in a simpler fashion.

## 3 A working example

To outline the steps needed to correctly average wind, we will use a ten-minute data file spanning two weeks from a climate station in Auckland, New Zealand maintained by NIWA (station number: 21937).[6] We will use R[7] for the processing and use the formulas displayed in Section 2.

### 3.1 The data

First we will load a pre-cleaned data file which includes meteorological data, a date column, some metadata, and has headers which should be clearly identifiable along with two helpful R packages.[8,9,10] The `head` function will display the first rows of data and `nrow` will tell us how many measurements the data file contains.

```r
# Load packages:
# An air quality data analysis package
library(openair)
# and a package which makes dates easy
library(lubridate)

## Loading required package:  methods

# Read ten-minute data file
data.alb <- read.csv('albany_cli_flo_data_december_2013.csv')
# Process the dates into the POSIXct format
data.alb$date <- ymd_hms(data.alb$date)

# Display the start of the data file
head(data.alb)

##     site                date temp rh  ws  wd rain press rad
## 1 Albany 2013-12-01 00:00:00 14.9 87 0.7 257  0.0    NA   0
## 2 Albany 2013-12-01 00:10:00 14.8 87 0.7 244  0.0    NA   0
## 3 Albany 2013-12-01 00:20:00 14.8 88 1.3 262  0.2    NA   0
## 4 Albany 2013-12-01 00:30:00 14.9 87 1.7 266  0.0    NA   0
## 5 Albany 2013-12-01 00:40:00 14.9 88 0.9 260  0.0    NA   0
## 6 Albany 2013-12-01 00:50:00 14.9 88 0.9 263  0.0    NA   0

# How many ten-minute records are there in the file?
info.alb.rows <- nrow(data.alb)
info.alb.rows

## [1] 3021
```

The first step for most analyses is to visualise the data. In this example, a standard timeseries plot is the easiest way to do this (Figure 1):

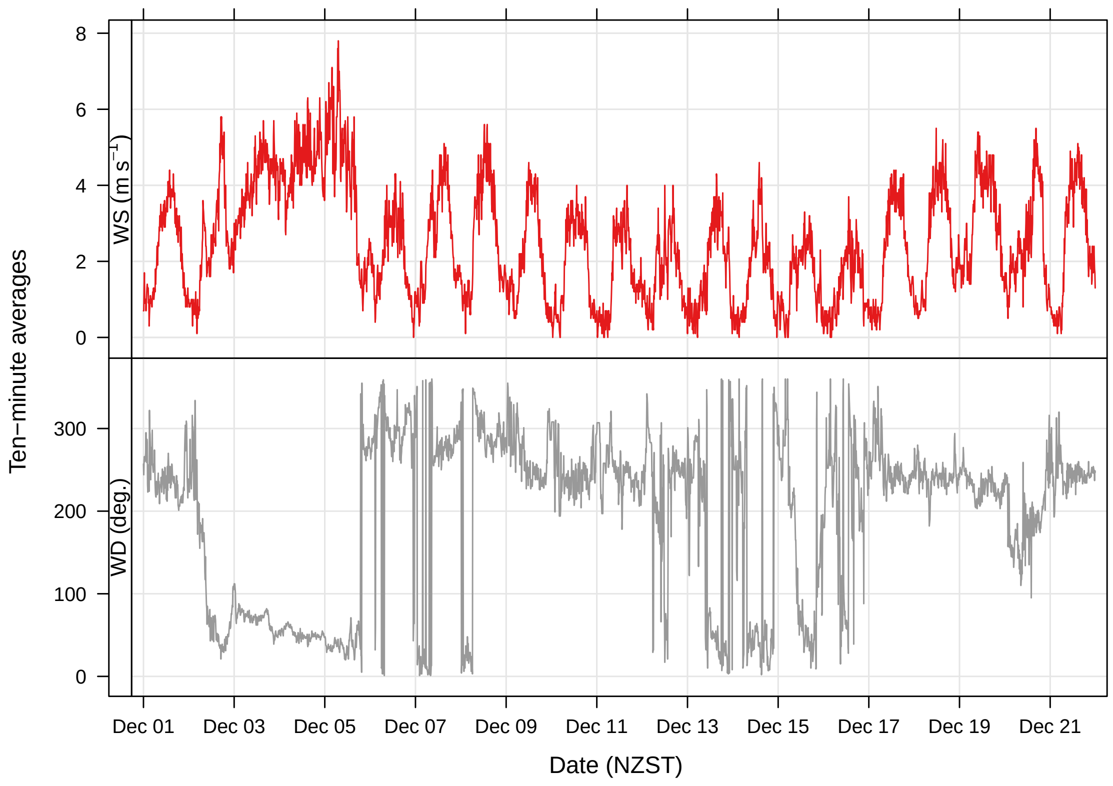


Figure 1: Ten-minute averages of wind speed and wind direction for the Albany, Auckland, New Zealand climate station for the first two weeks of December, 2013.

We can clearly see a diurnal cycle in wind speed where the daylight hours are windy while during the night, wind speed is much lower (Figure 1). We can also identify a period between December 2–6 when wind speeds were the highest in the time period and the wind direction was less variable. This is standard wind behaviour. If I was to subjectively judge what the average wind direction was during this period, I would say it was somewhere between 200 and 300 degrees. There are a number of periods that traverse the 0–360 degree scale discontinuity however, and I do not know how much this would influence the average.

Wind roses are specialised plots which are great visualisation tools to display the behaviour of wind at a monitoring site. Here, it would be useful to aggregate the wind data by day- and night-time hours to see the different patterns between these periods (Figure 2).

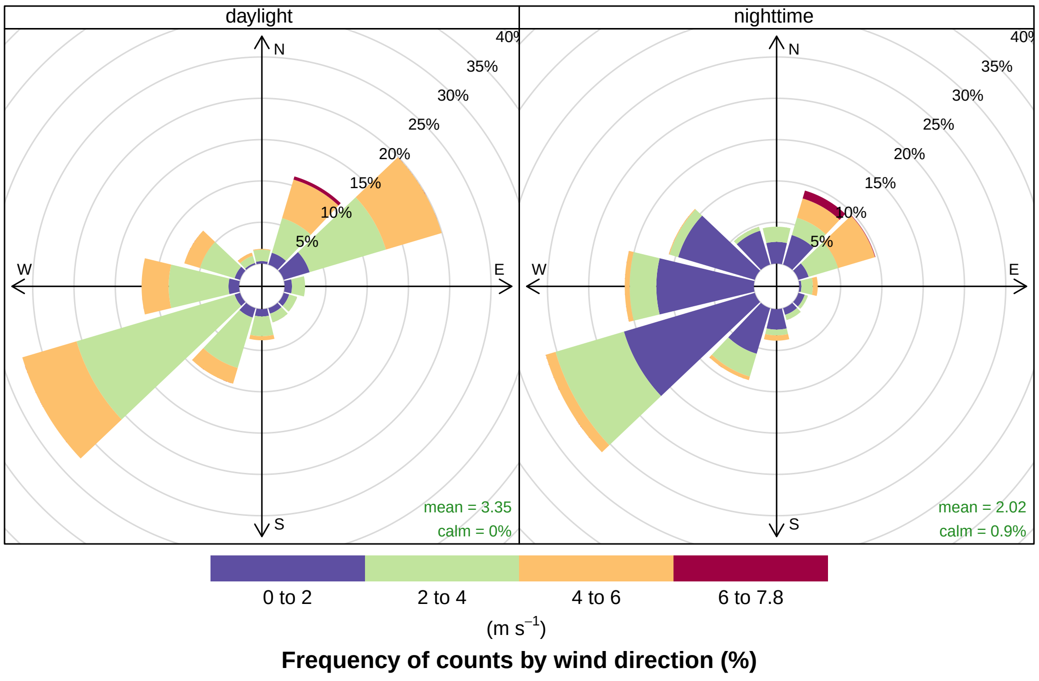


Figure 2: Wind roses for the Albany, Auckland, New Zealand climate station for the first two weeks of December, 2013. Plots are conditioned by day- and night-time hours.

It is evident in the wind rose plots (Figure 2) as well as the timeseries (Figure 1) that nighttime has a much greater proportion of low wind speeds than the daytime hours at this monitoring location.

To begin averaging wind speed and direction we need to calculate the $\vec{u}$ and $\vec{v}$ components. This is simple in R:

```r
# Calculate the u and v wind components
data.alb$u.wind <- - data.alb$ws * sin(2 * pi * data.alb$wd/360)
data.alb$v.wind <- - data.alb$ws * cos(2 * pi * data.alb$wd/360)

# Display three rows of only the wind data
head(subset(data.alb, select = c(date, ws, wd, u.wind, v.wind)), 3)

##                    date  ws  wd u.wind v.wind
## 1 2013-12-01 00:00:00 0.7 257 0.6821 0.1575
## 2 2013-12-01 00:10:00 0.7 244 0.6292 0.3069
## 3 2013-12-01 00:20:00 1.3 262 1.2873 0.1809
```

For spreadsheet software, simply replace the `ws` and `wd` with the appropriate cell references. Unlike wind direction in degrees, the wind components can be averaged to produce a valid average.[1] To calculate the average wind direction for the entire monitoring period we can combine and sum the vectors and then use the `atan2` function to average the vectors. This `atan2` is a different function to that displayed in Eq. 4 and enables the flow clause to be resolved without the extra step. The `360/2/pi` converts the radians back to degrees and the addition of 180 moves the values into the correct quadrants by altering their origin.[7] I have also included the `na.rm` argument in the `mean` function in this example because this ignores `NA` values which are almost always present in this type of monitoring data.

```
# Calculate the average wind vectors
mean.u <- mean(data.alb$u.wind, na.rm = T)
mean.v <- mean(data.alb$v.wind, na.rm = T)
# Calculate the resultant vector average wind direction with atan2
wd.average <- (atan2(mean.u, mean.v) * 360/2/pi) + 180
# Display
wd.average

## [1] 283.2

# Calculate the vector average wind speed
ws.vector.average <- ((mean.u^2 + mean.v^2)^0.5)
ws.vector.average

## [1] 0.4665

# Calculate the scalar average wind speed, the standard mean
ws.scalar.average <- mean(data.alb$ws, na.rm = T)
ws.scalar.average

## [1] 2.45
```

It is very important to note that this `atan2` function in most programming languages takes the from `atan2(u,v)`, but in most spreadsheet software, the vector arguments are reversed.[11] Ensure that you always visualise your data after manipulation so this can be checked.

The average wind direction for the monitoring period was therefore 283 degrees. If we look at Figure 1, this seems sensible and my first impressions were correct. In this example, the two types of wind speeds, vector and scalar, have produced very different values; compare 0.47 and 2.45 $\mathrm{m\,s^{-1}}$.

The vector average wind speeds are always lower than the scalar average wind speeds because of the behaviour of vectors. In respect to wind, a good example to explain this follows. If wind was blowing from the east for 30 minutes at $5\,\mathrm{m\,s^{-1}}$ then swapped to the west for 30 minutes with the same magnitude ($5\,\mathrm{m\,s^{-1}}$) and the vector average wind speed was taken, it would equal zero because the vectors cancel each other out. In reality this rarely occurs, but if a monitoring location experiences a bimodal wind pattern because of land-sea or mountain-valley breezes, vector averaged wind speed can represent atmospheric motion rather poorly. This is why I have a preference for the scalar average wind speed.

### 3.2 Manually isolate an hour for averaging

Now we will manually isolate an hour for averaging. This is not an efficient way to do such averaging, but again, it demonstrates the steps necessary. It is important to note that when averages of time periods are taken, in air quality and meteorology applications, the date and time represents the preceding hour.[12,13] For example, if an hourly average is represented with a date such as `2013-12-05 07:00:00`, then this average should include all measurements in the time period between `2013-12-05 07:00:00` and `2013-12-05 07:59:59`. This can be counter-intuitive at small time scales because the last measurement time may seem more representative or 'sensible'. But when averages are taken over longer time periods, such as a day, it becomes clear that when discussing a date like `2013-12-05`, it will be representing all data for the preceding 24 hours (all measurements between `2013-12-05 00:00:00 - 2013-12-05 23:59:59`).

```
# Create new data frame by subsetting by date
data.alb.first.hour <- subset(data.alb, date <= ymd_hms('2013-12-01 00:59:59'))
# Calculate the average wind vectors
mean.u <- mean(data.alb.first.hour$u.wind, na.rm = T)
mean.v <- mean(data.alb.first.hour$v.wind, na.rm = T)
# Calculate average for this hour-long period
wd.mean <- (atan2(mean.u, mean.v) * 360/2/pi) + 180
wd.mean

## [1] 260.4

# Calculate the scalar average wind speed, the standard mean
ws.scalar.average <- mean(data.alb.first.hour$ws, na.rm = T)
ws.scalar.average

## [1] 1.033
```

### 3.3 Using `timeAverage`

After the examples and hopefully some understanding about the details regarding averaging of wind, we can take a shortcut and use one of my favourite functions; `timeAverage` in the **openair** package.[8,9] The `timeAverage` function works by aggregating data by almost any time period and has the ability to correctly process wind speed and direction. The first prerequisite for `timeAverage` to work is that the date column must have the header `date` (the lower case does matter) and it is of the `POSIXct` class. When we loaded the data this was immediately done with the `ymd_hms` function and can be confirmed by the `str` command:

```
# What class of data is the date column?
str(data.alb$date)

##  POSIXct[1:3021], format: "2013-12-01 00:00:00" "2013-12-01 00:10:00" ...
```

The other prerequisites are that the wind speed and direction columns must have the names: `ws` and `wd` (again the case matters). In our example, these prerequisites are fulfilled and can use the `timeAverage` function for various time periods immediately:

```
# Drop the wind vectors because timeAverage computes them internally
data.alb <- subset(data.alb, select = -c(u.wind, v.wind))

# Hourly averages
data.alb.hour <- timeAverage(data.alb, avg.time = 'hour')
head(data.alb.hour, 3)

##     site                date  temp    rh     ws    wd    rain rad
## 1 Albany 2013-12-01 00:00:00 14.87 87.50 1.0333 260.4 0.03333   0
## 2 Albany 2013-12-01 01:00:00 14.87 88.00 0.9833 272.6 0.00000   0
## 3 Albany 2013-12-01 02:00:00 14.58 84.67 1.2000 246.8 0.00000   0

# Daily averages
data.alb.day <- timeAverage(data.alb, avg.time = 'day')
head(data.alb.day, 3)

##     site       date  temp    rh    ws     wd     rain    rad
## 1 Albany 2013-12-01 16.12 63.61 2.291 237.13 0.001389 0.1838
## 2 Albany 2013-12-02 16.71 65.15 2.374  61.06 0.000000 0.2232
## 3 Albany 2013-12-03 17.94 78.31 4.062  69.07 0.004167 0.0666

# Monthly averages
data.alb.month <- timeAverage(data.alb, avg.time = 'month')
head(data.alb.month)

##     site       date  temp    rh   ws    wd    rain    rad
## 1 Albany 2013-12-01 17.83 73.33 2.45 283.2 0.02635 0.1636
```

Again, look at the data, but this time (only) the hourly averages:

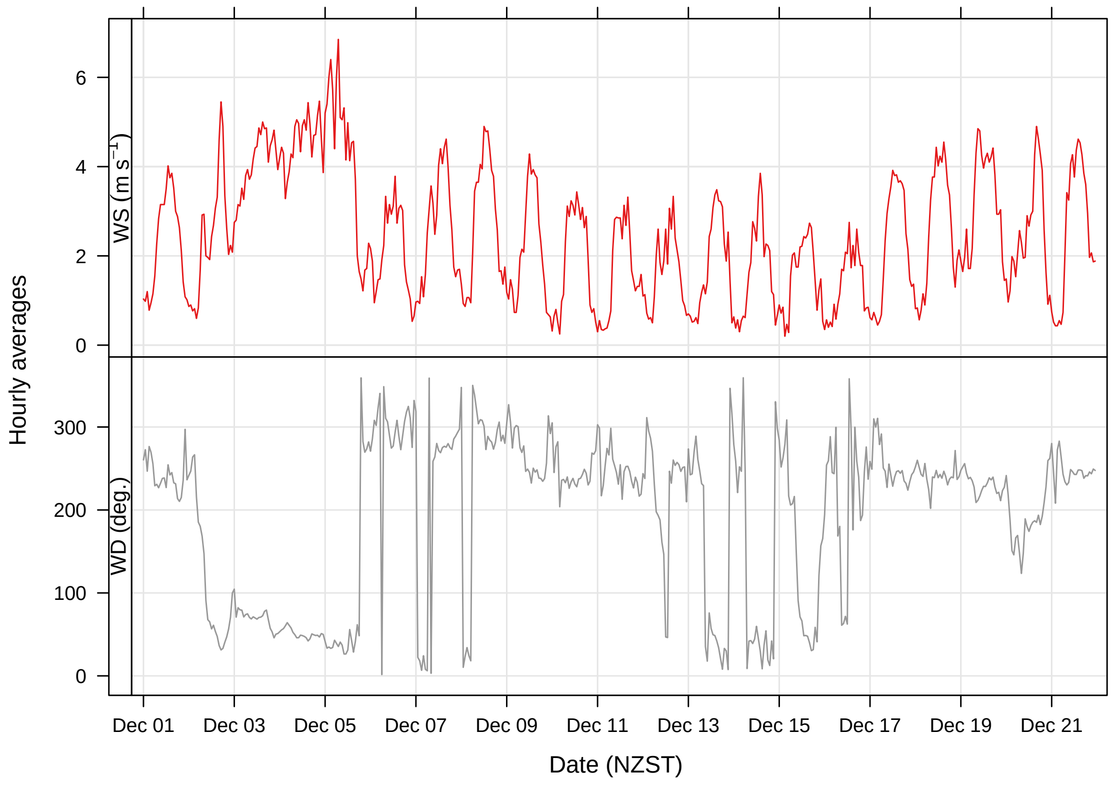


Figure 3: Hourly averages of wind speed and wind direction for the Albany, Auckland, New Zealand climate station for the first two weeks of December, 2013.

The hourly averages displayed in Figure 3 are constant with the source data seen in Figure 1. It looks like the averaging process has preserved most of the data's structure but has reduced noise which is often why averaging is done.

The default behaviour of `timeAverage` is to calculate the scalar average of wind speed. However, with the use of the `vector.ws` argument, the vector average wind speed can be calculated if desired:

```
# Monthly averages with vector ws
data.alb.month <- timeAverage(data.alb, avg.time = 'month', vector.ws = T)
head(data.alb.month)

##      site       date  temp    rh     ws    wd    rain    rad
## 1 Albany 2013-12-01 17.83 73.33 0.4665 283.2 0.02635 0.1636
```

We can see that because `timeAverage` has aggregated the entire dataset together when `avg.time = "month"` because the data was all logged in December, 2012. Our manual examples are identical to `timeAverage`'s outputs which suggests our working is correct.

## 3.4 Keeping it simple

A keen observer would have noticed that this document has much redundancy. The objective of averaging wind speed and direction in our example can be handled easily and simply with a short set of R commands without the understanding of vectors and trigonometric functions:

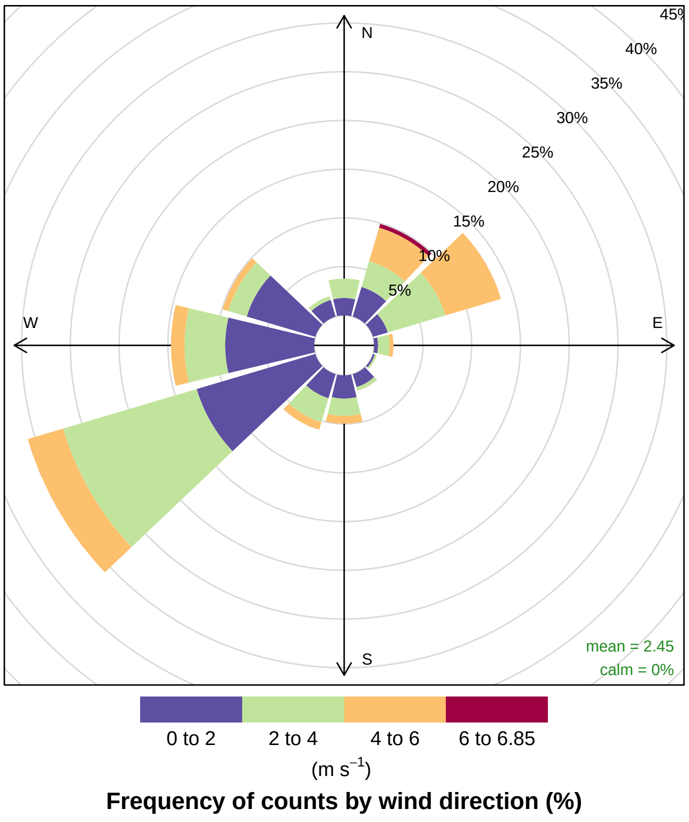
```
# Load packages:
# An air quality data analysis package
library(openair)
# and a package which makes dates easy
library(lubridate)

# Read ten-minute data file
data.alb <- read.csv('albany_cli_flo_data_december_2013.csv')
# Process the dates into the POSIXct format
data.alb$date <- ymd_hms(data.alb$date)

# Calculate hourly means
data.alb.hour <- timeAverage(data.alb, avg.time = 'hour')
head(data.alb.hour, 5)

##     site                date  temp    rh     ws    wd    rain rad
## 1 Albany 2013-12-01 00:00:00 14.87 87.50 1.0333 260.4 0.03333   0
## 2 Albany 2013-12-01 01:00:00 14.87 88.00 0.9833 272.6 0.00000   0
## 3 Albany 2013-12-01 02:00:00 14.58 84.67 1.2000 246.8 0.00000   0
## 4 Albany 2013-12-01 03:00:00 14.05 84.00 0.7833 276.5 0.00000   0
## 5 Albany 2013-12-01 04:00:00 13.20 86.17 0.9500 269.4 0.00000   0

# Display the wind rose
windRose(data.alb.hour, paddle = F)
```



```
# Now continue and plot the data, write another file for sharing, or add to a database
```

This would be my ultimate recommendation to how to deal with averaging wind speeds and directions. This demonstrates the power of R and open-source software because the authors of the package have designed, written, and shared (for free) an easy to use solution for a tricky problem which can also be freely studied.

## Data and document notes

This document was prepared with **knitr**, a report generator that straddles both LaTeX and R and allows for the data analysis and report generation to occur at the same time during compilation.[14] The climate data from Albany was retrieved from the CliFlo database which cannot be shared directly, but can be accessed with those who have a account (`http://cliflo.niwa.co.nz/`). If the source code fails to compile, you may need to install **openair** and **lubridate** with `install.packages(c("openair","lubridate"))`.

## References

[1] U.S. Environmental Protection Agency. Meteorological Monitoring Guidance for Regulatory Modeling Applications. Office of Air and Radiation. Office of Air Quality Planning and Standards. EPA-454/R-99-005. 2000. URL: http://www.epa.gov/scram001/guidance/met/mmgrma.pdf.

[2] Control.com. Calculating an average value of the wind direction. 2010. URL: http://www.control.com/thread/1026210133.

[3] Glickman, Todd S. *Glossary of Meteorology, Second Edition.* Ed. by The Council of the AMS. American Meteorological Society (AMS), 2000. URL: http://amsglossary.allenpress.com/glossary.

[4] Atmospheric Research & Technology, LLC. Vector vs. Scalar Averaging of Wind Data. 2013. URL: http://www.sodar.com/VectorScalar.htm.

[5] WebMET.com. Meteorological Data Processing/Wind Direction and Wind Speed/Vector Computations. 2002. URL: http://www.webmet.com/met_monitoring/622.html.

[6] NIWA. New Zealand's National Climate Database—CliFlo database. 2014. URL: http://cliflo.niwa.co.nz/.

[7] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing. Vienna, Austria, 2014. URL: http://www.R-project.org/.

[8] Carslaw, David C. and Ropkins, Karl. *openair* — An R package for air quality data analysis. *Environmental Modelling & Software* 27–28 (Jan. 2012), pp. 52–61. ISSN: 1364-8152. URL: http://www.sciencedirect.com/science/article/pii/S1364815211002064.

[9] Carslaw, David and Ropkins, Karl. openair: Open-source tools for the analysis of air pollution data. R package version 1.0. 2014.

[10] Grolemund, Garrett and Wickham, Hadley. Dates and Times Made Easy with lubridate. *Journal of Statistical Software* 40.3 (2011), pp. 1–25. URL: http://www.jstatsoft.org/v40/i03/.

[11] The National Center for Atmospheric Research. Wind direction quick reference. 2014. URL: https://www.eol.ucar.edu/content/wind-direction-quick-reference.

[12] Ministry for the Environment. Good Practice Guide for Air Quality Monitoring and Data Management 2009. Ministry for the Environment Ref: ME933. 2009. URL: http://www.mfe.govt.nz/publications/air/good-practice-guide-air-quality-2009/good-practice-guide-for-air-quality.pdf.

[13] Ministry for the Environment. 2011 Users' Guide to the revised National Environmental Standards for Air Quality. Publication number: ME 1068. 2011. URL: http://www.mfe.govt.nz/publications/air/2011-user-guide-nes-air-quality/2011-user-guide-nes-air-quality.pdf.

[14] Xie, Y. *knitr: A general-purpose package for dynamic report generation in R. R package.* R package version 1.2. 2013. URL: http://CRAN.R-project.org/package=knitr.