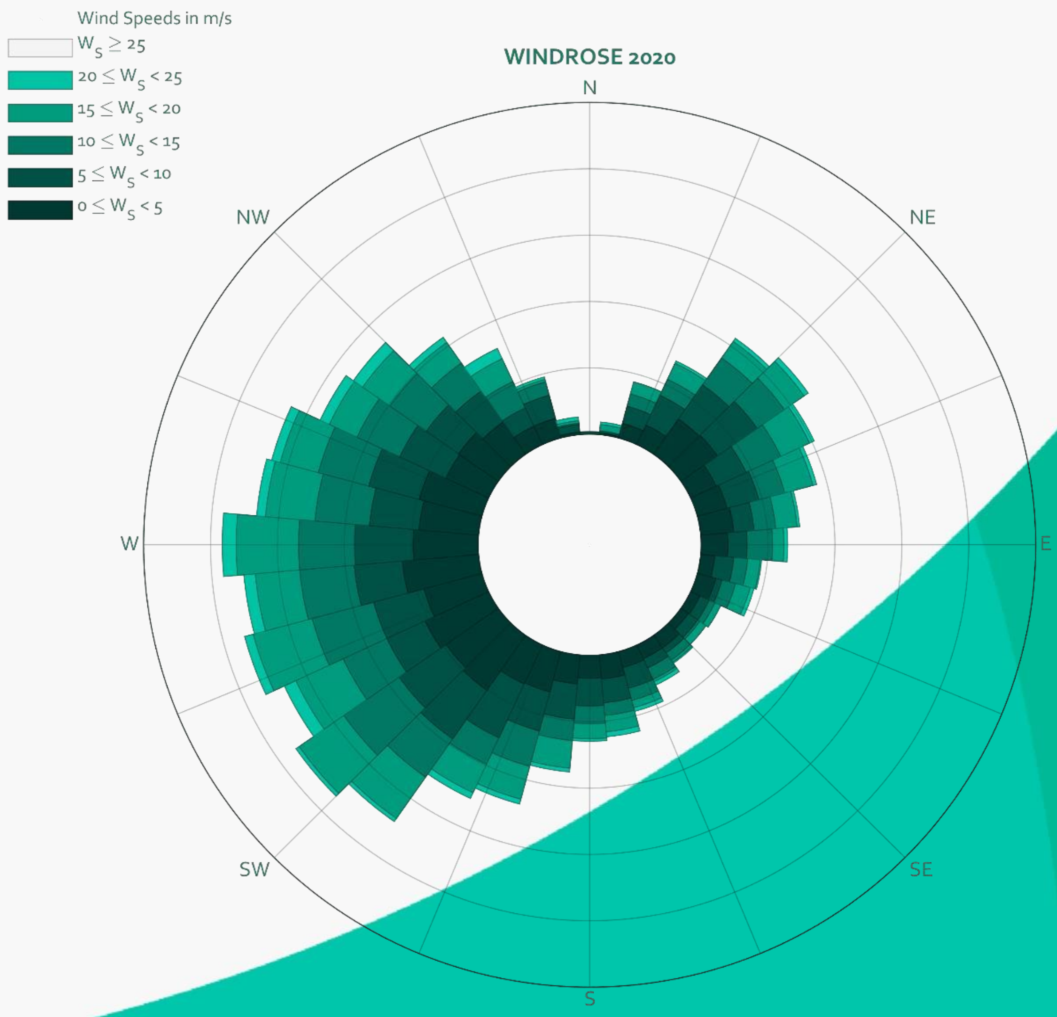


WINDROSE FOR MATLAB

20 ABR 2021

By Daniel Pereira

daniel.pereira.valades@gmail.com
dpereira.asempyme.com

Donate with PayPal

WindRose Documentation

DATA

We start from some simple data which we want to be represented in a wind rose. These data could come from data measurement (temporal series, data collection, etc.).

I have created the function WindRandomDistrib to generate a random distribution with any number of elements (8760 in this case) and a maximum wind speed (21.15 in this case).

```
clc; clear; close all;
rng(31081987);
[spd, dir] = WindRandomDistrib(8760, 21.15);
```

NOTES

This function can be called with many arguments at the same time. The obligatory input arguments are the wind directions and wind speeds (two different vectors).

The following examples have been created in order to show the effect of a particular command, but all of them can be combined in the function call, within a structure, a cell array or the common call.

It is important to mention that the properties can be called in **lowercase**, **UPPERCASE** or **mixedCASE**. In case of repating properties, the last value will be used for the function.

The following three samples show the different ways of calling the function, giving the same result.

a) With options in a cell array (the easiest if you have to call several times with same fixed options):

```
Options = {'anglenorth', 0, 'angleeast', 90, 'labels', {'N (0°)', 'E (90°)', 'S (180°)',
'W (270°)'}, 'freqlabelangle', 45};
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options);
% Now we want to add extra options for the following function call
Options1 = [Options,{'axes',gca,'legendtype',2}];
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options1);
```

b) With options in a structure (the easiest for changing only one parameter between calls):

```
Options.AngleNorth     = 0;
Options.AngleEast      = 90;
Options.Labels         = {'N (0°)', 'E (90°)', 'S (180°)', 'W (270°)'};
Options.FreqLabelAngle = 45;
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options);
% Change only one parameter for new call
Options.FreqLabelAngle = 30;
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options);
```

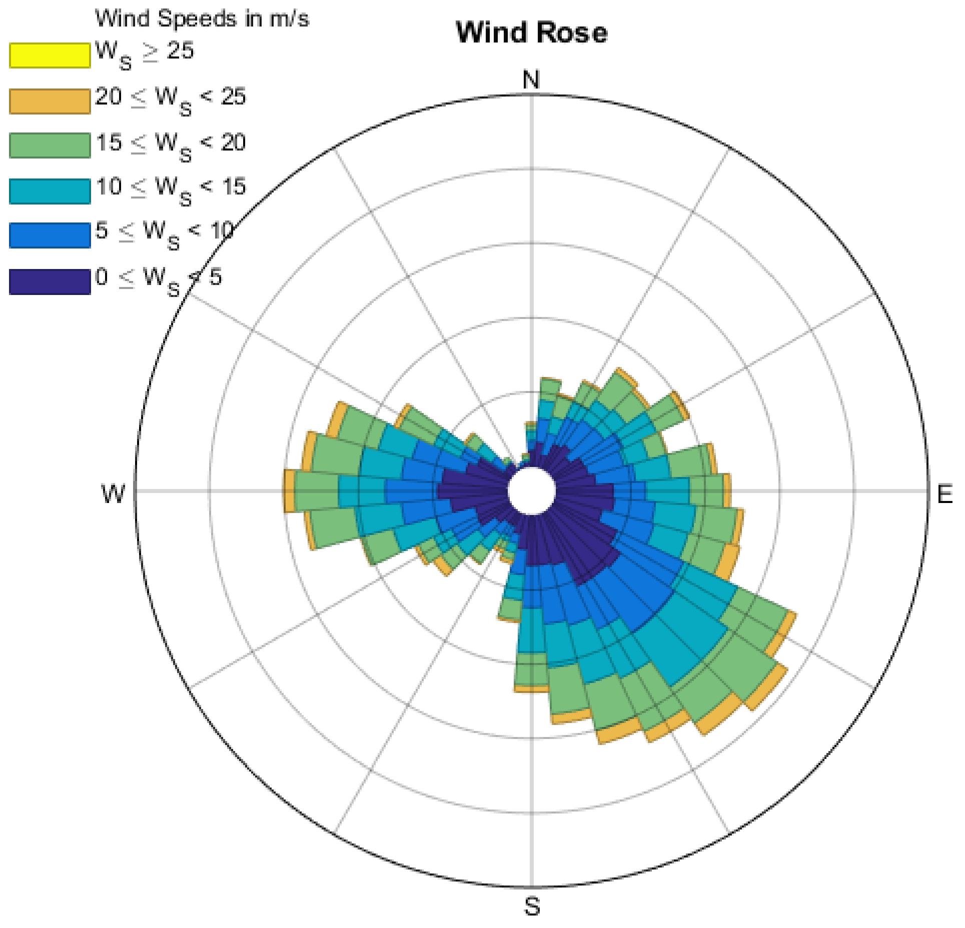
c) Usual Matlab function calling (specify everything with every function call):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'anglenorth', 0,
'angleeast', 90, 'labels', {'N (0°)', 'E (90°)', 'S (180°)', 'W (270°)'},
'freqlabelangle', 45);
```

SIMPLE USAGE

Just showing the windrose in a new figure:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd);
```



The default option represents 36 directions (with an aperture angle of 360/36 = 10º each) Note that the bins are centered in θ +0° (θ -5° to θ +5°) by default.

REFERENCE ANGLES - `'AngleNorth'` and `'AngleEast'`

Tell the function your data's references (origin and orientation).

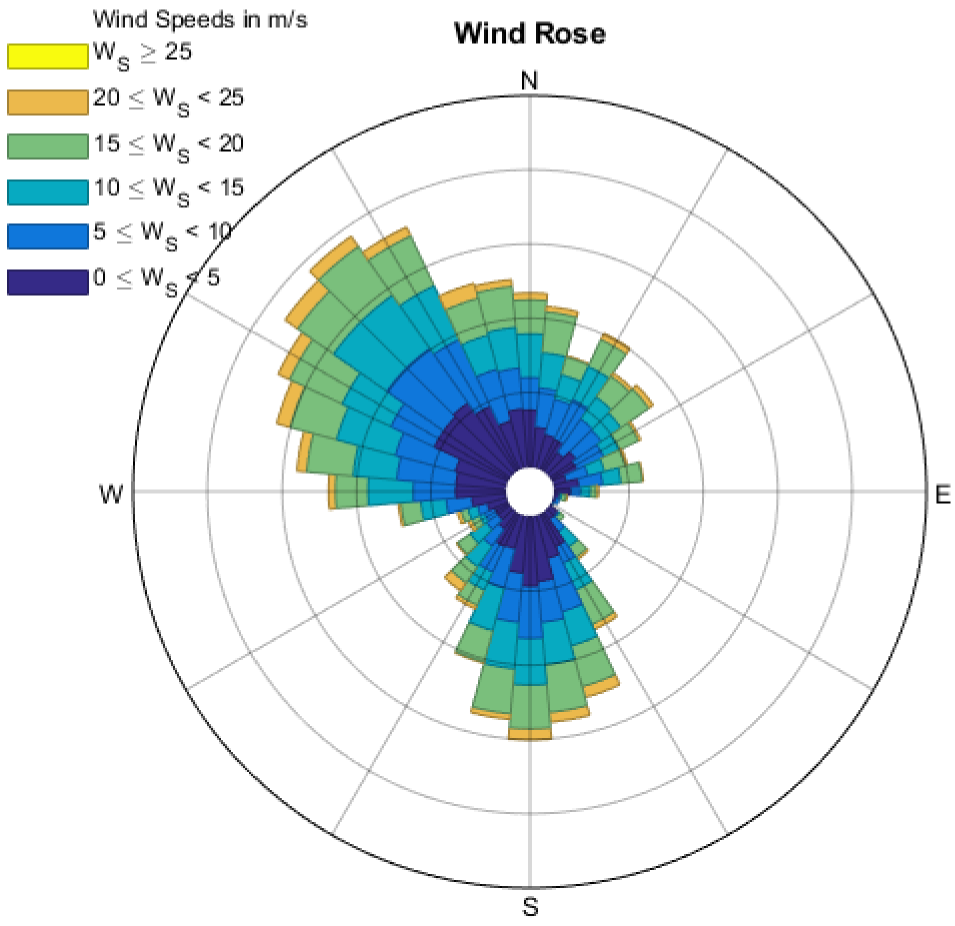
As there will always be controversy about which should be the reference angles, **this function uses by default the trigonometric convention**:

Counterclockwise, with 0° angle in the right of the circle (East).

If you want to define your own convention, use sexagesimal degrees to define which angle corresponds to North direction (up) and to East direction (right), so any user can use the desired references. These two values must differ in 90° and both values must be specified.

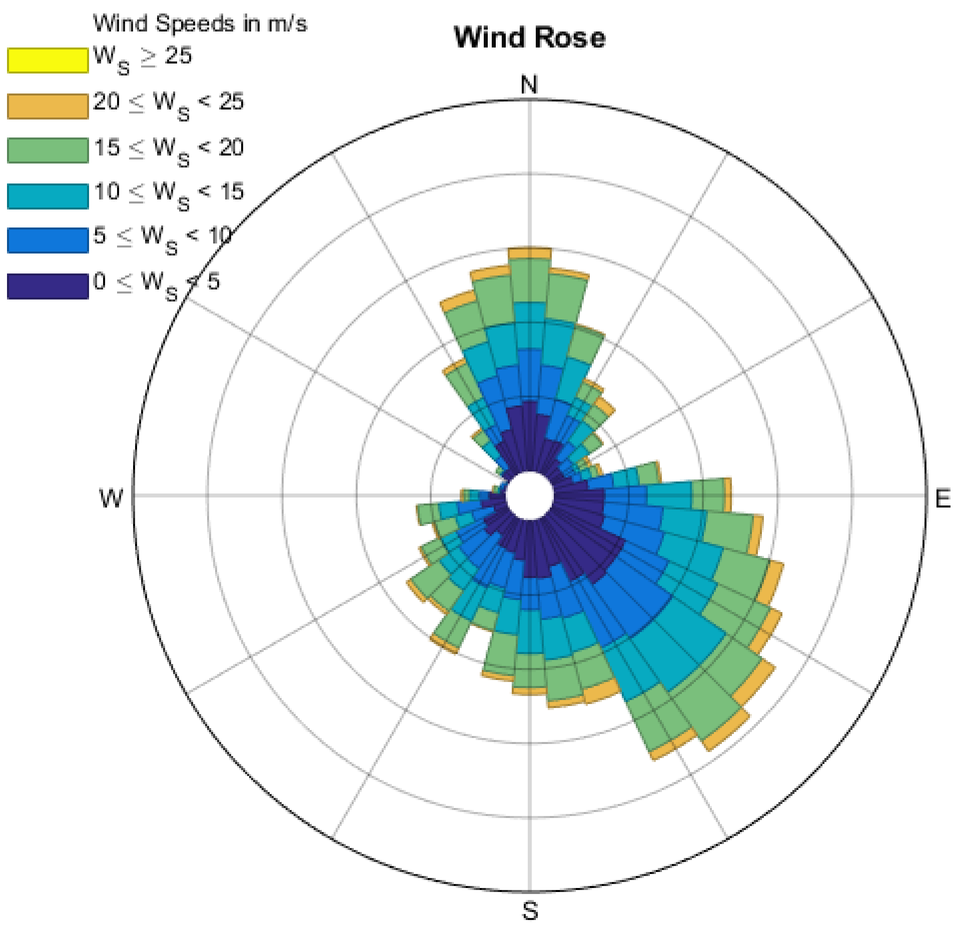
The **meteorological convention**, where North is 0° and East is 90° (we can then define origin *[N]* and orientation *[clockwise]*), is the most usual:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'AngleNorth', 0,
'AngleEast', 90);
```



Let's imagine that our data uses the noon solar convention in the northern hemisphere (0° is South, and 90° is West) → North is 180° and East is 270°.

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'AngleNorth',
180, 'AngleEast', 270);
```



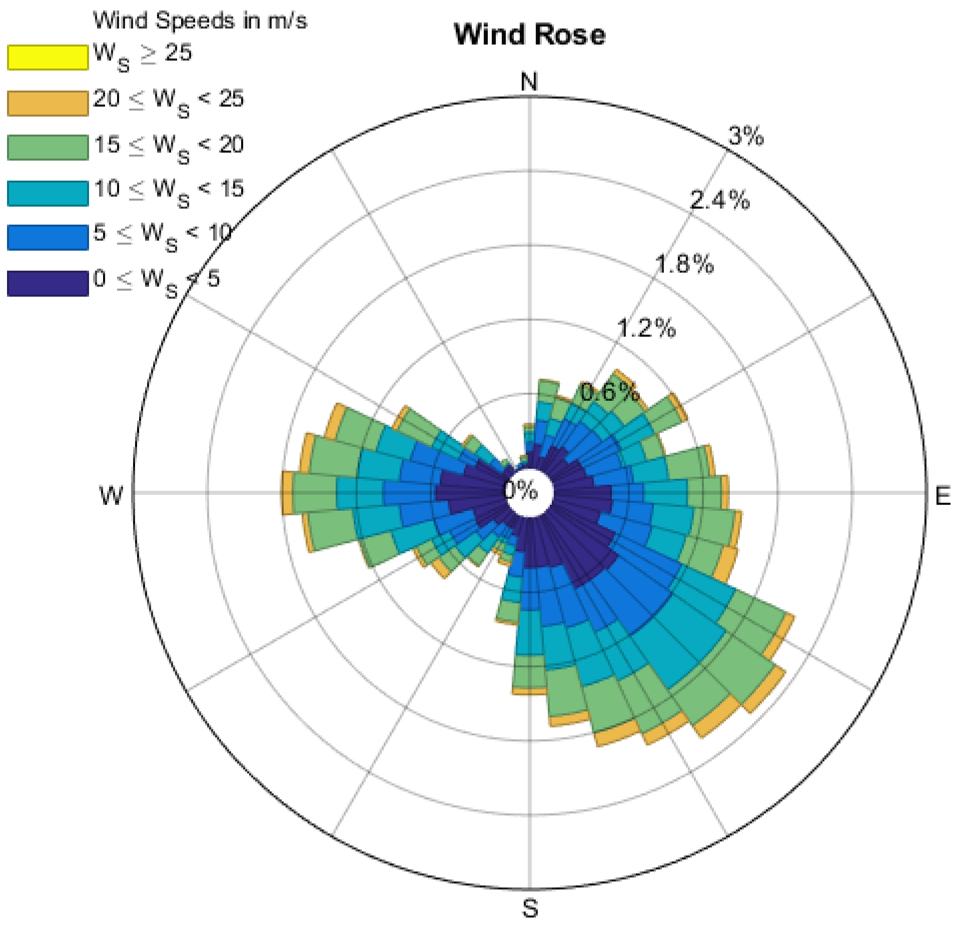
FREQUENCY LABELS - 'FreqLabelAngle'

Show Frequency label angles.

If we want to know the frequency in each direction, it is recommended to add the frequency labels, in any angle.

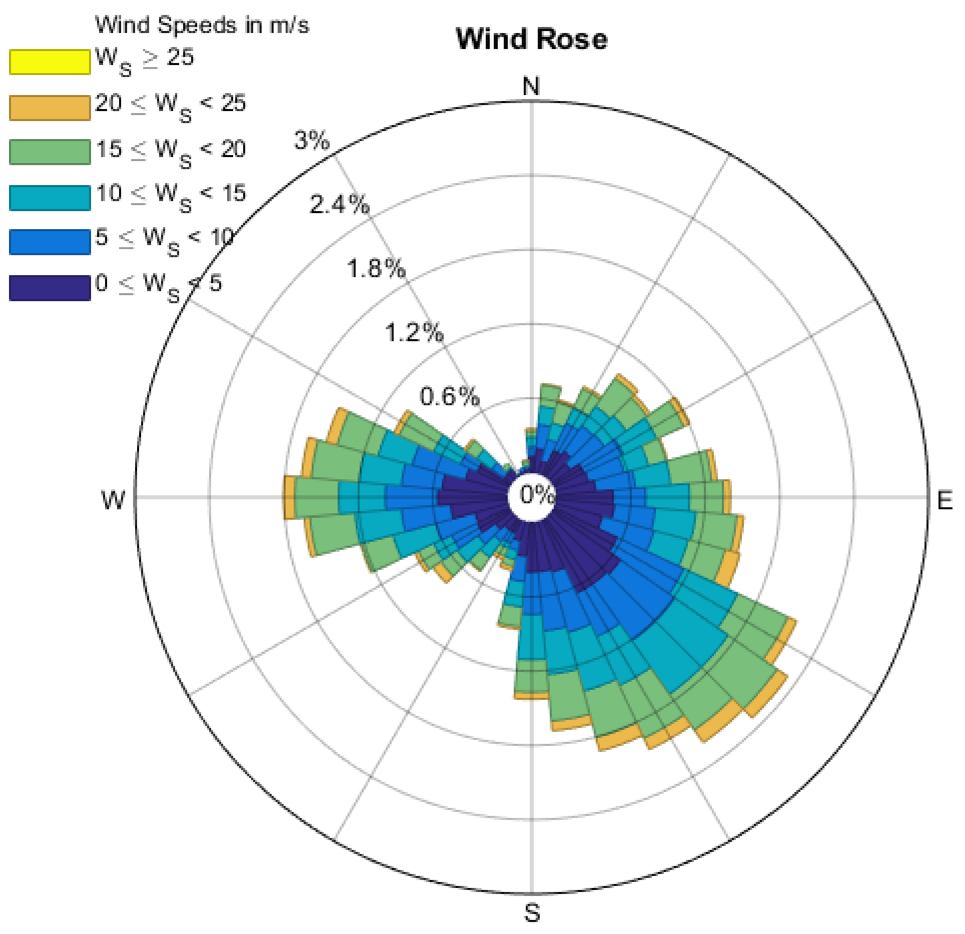
This will show the label angle at 60° (measured with the trigonometric reference):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'FreqLabelAngle',
60);
```



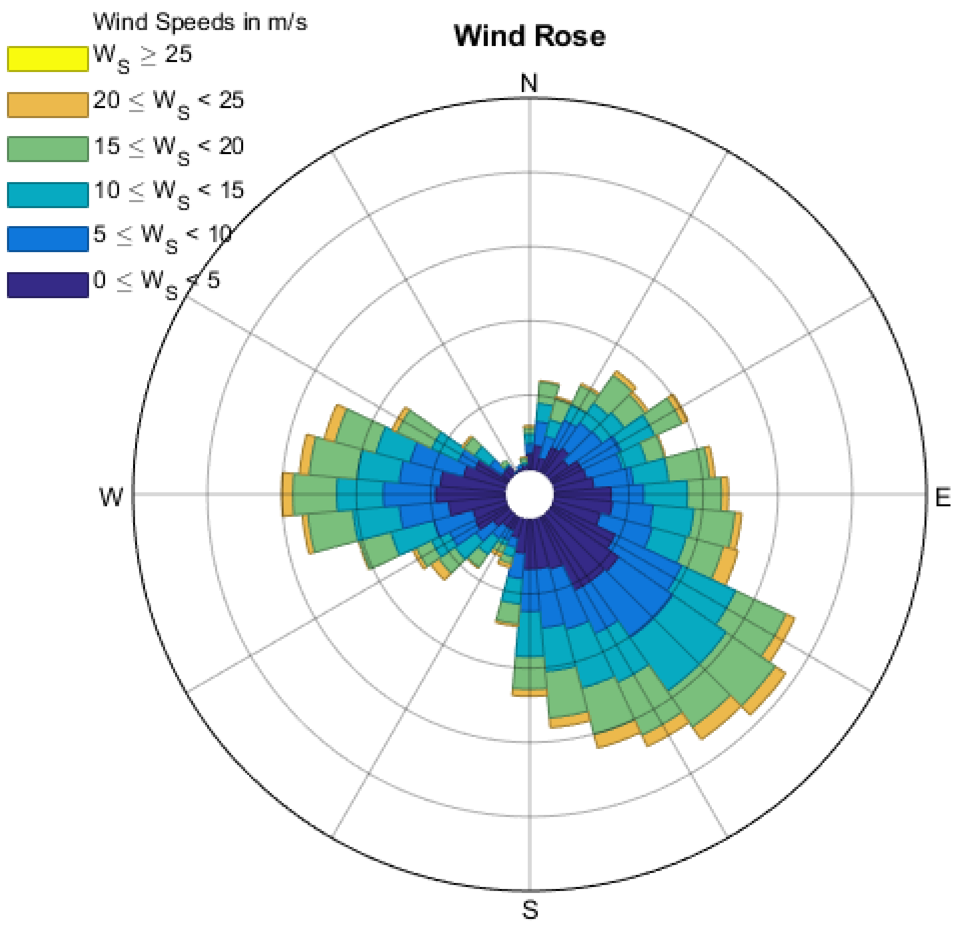
Using `'auto'` shows the frequency labels on the least frequent direction:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'FreqLabelAngle',
'auto');
```



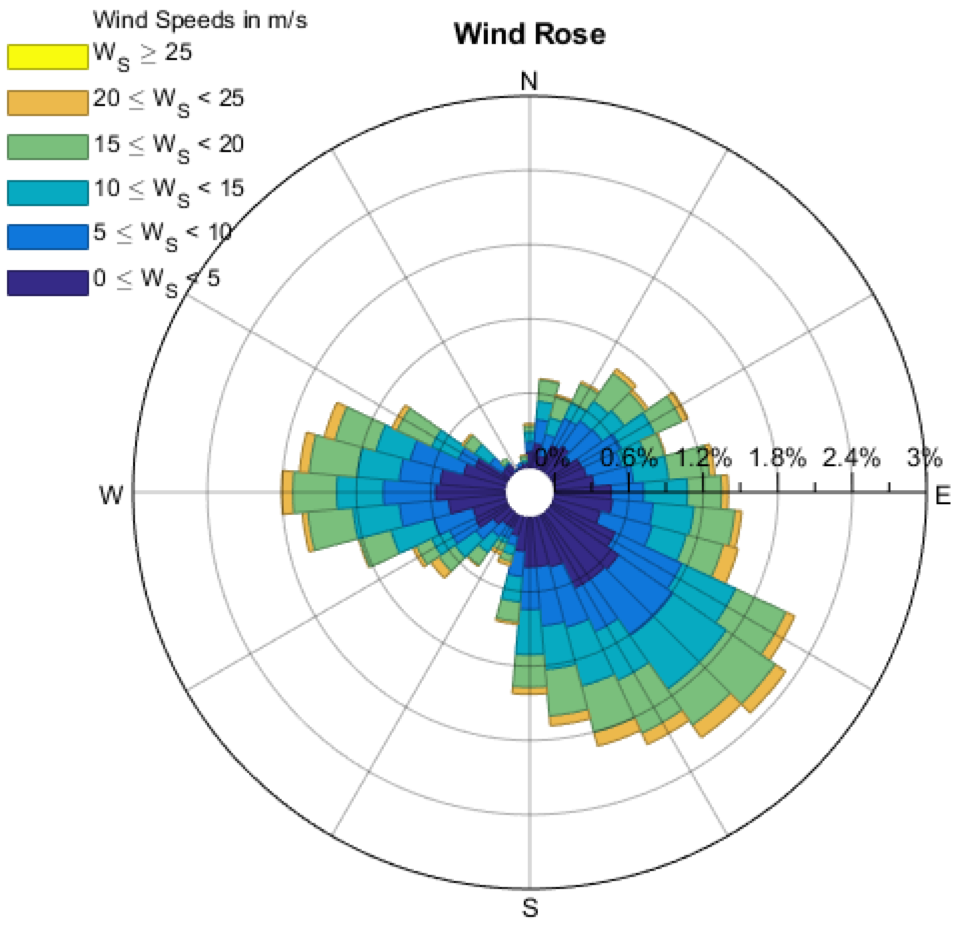
Using `'none'`, `NaN` or `[]` does not show the frequency labels:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'FreqLabelAngle',
'none');
```



Using `'ruler'`, `'rulerRight'` or `'rulerLeft'` shows the frequency labels in a horizontal ruler fashion (oriented to the right or to the left):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'FreqLabelAngle',
'ruler');
```

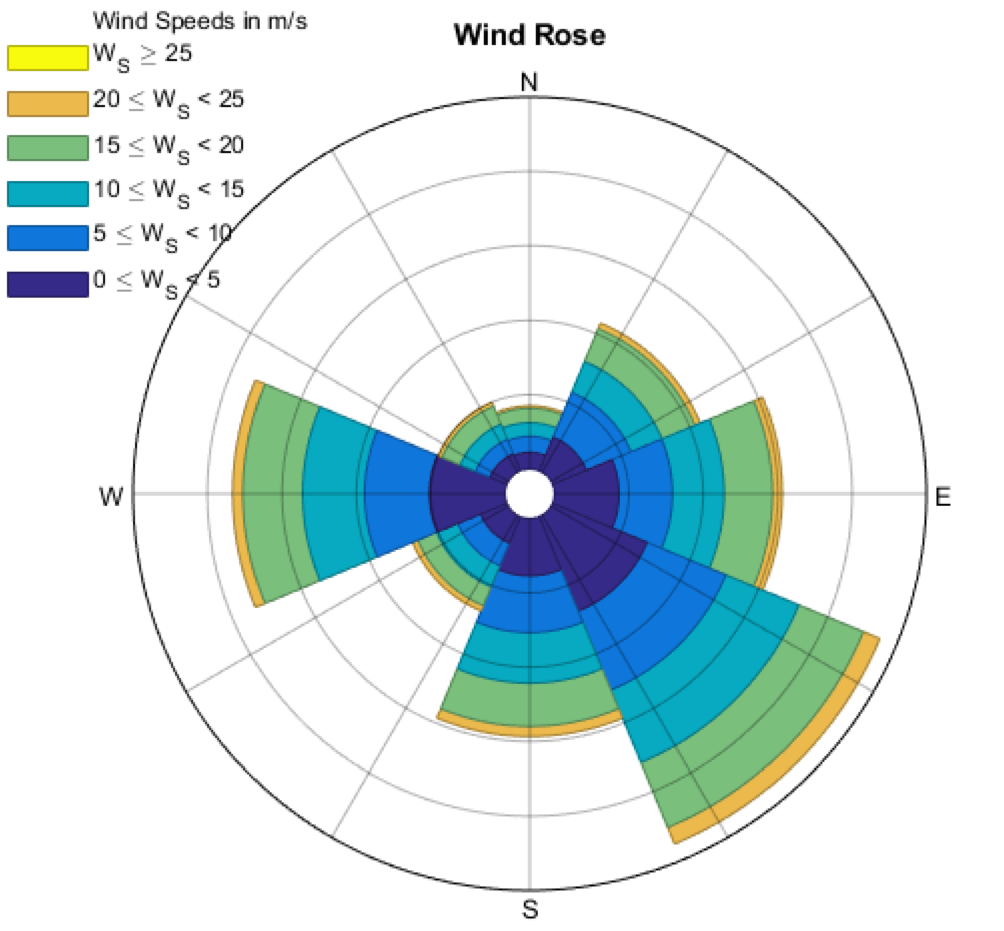

Wind Speeds in m/s
$W_S \geq 25$
$20 \leq W_S < 25$
$15 \leq W_S < 20$
$10 \leq W_S < 15$
$5 \leq W_S < 10$
$0 \leq W_S < 5$
Wind Rose
N
E
S
W
0%
0.6%
1.2%
1.8%
2.4%
3%

NUMBER OF DIRECTIONS - 'nDirections'

Set the number of separate directions bins to be calculated and shown.

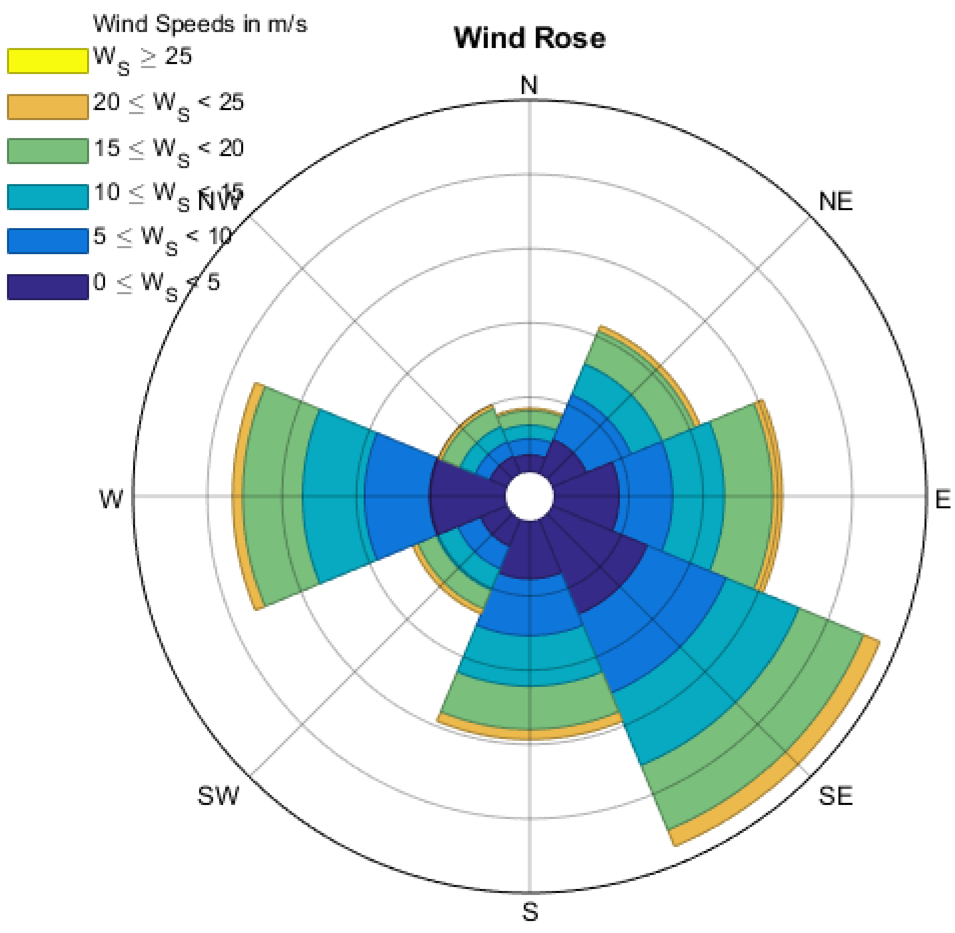
The number of directions can be changed by adding the number of directions we want to show, let's say, 8, by specifying `'ndirections'`.

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nDirections',
8);
```



We can also specify labels for these directions:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nDirections', 8,
'labels', {'N', 'NE', 'E', 'SE', 'S', 'SW', 'W', 'NW'});
```



Note that the radial grid values are changed when adding a cell array for the `'labels'`. See **Axis labels, Example 3** for more information.

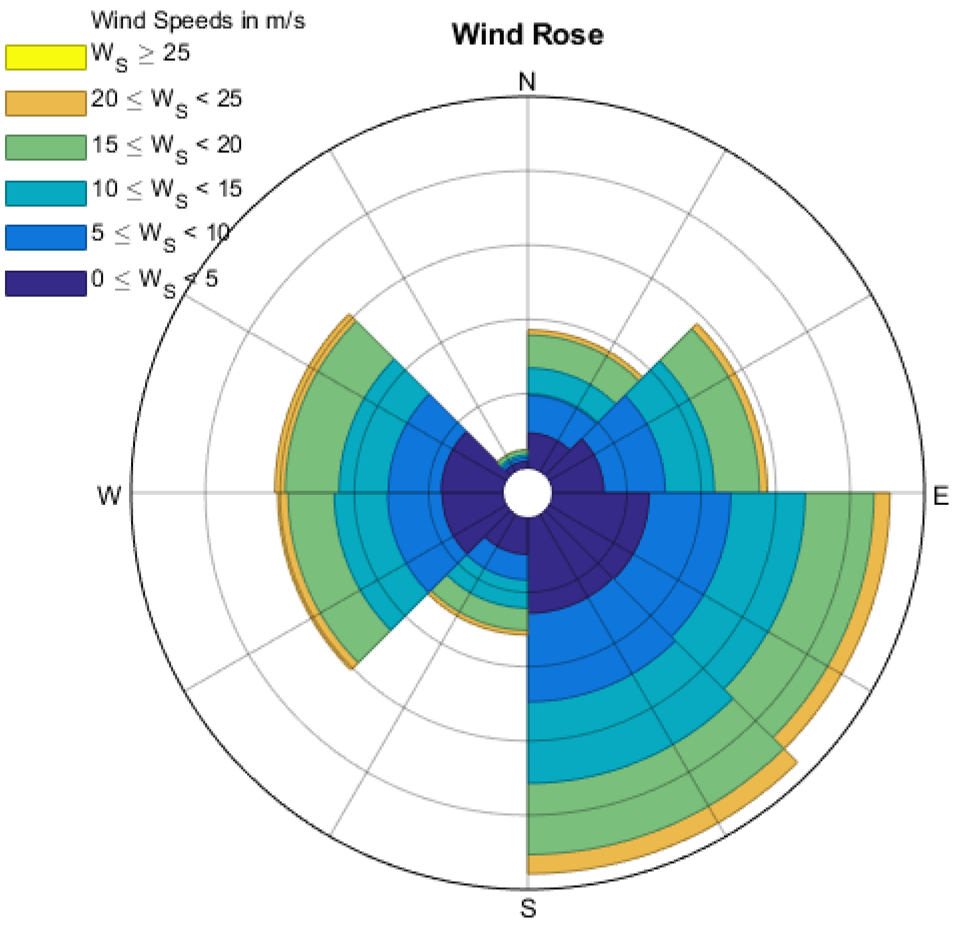
BIN ALIGNMENT - `'CenteredIn0'`

Center of bin can be the center or the origin of the direction interval.

If you do not want the bins (arms) to be centered in θ +0° direction, but you want them starting at that point, you can specify that bins should not be centered in 0.

This example is much clearer with a reduced number of directions:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nDirections', 8,
'CenteredIn0', false);
```



Compare the bins to the previous examples. In this new figure, bins appear in the direction 0° to 45° (360°/8), centered in 22.5°. They would normally appear between -22.5 and 22.5, centered in 0 if the parameter 'CenteredIn0' is set to true or omitted (default is true).

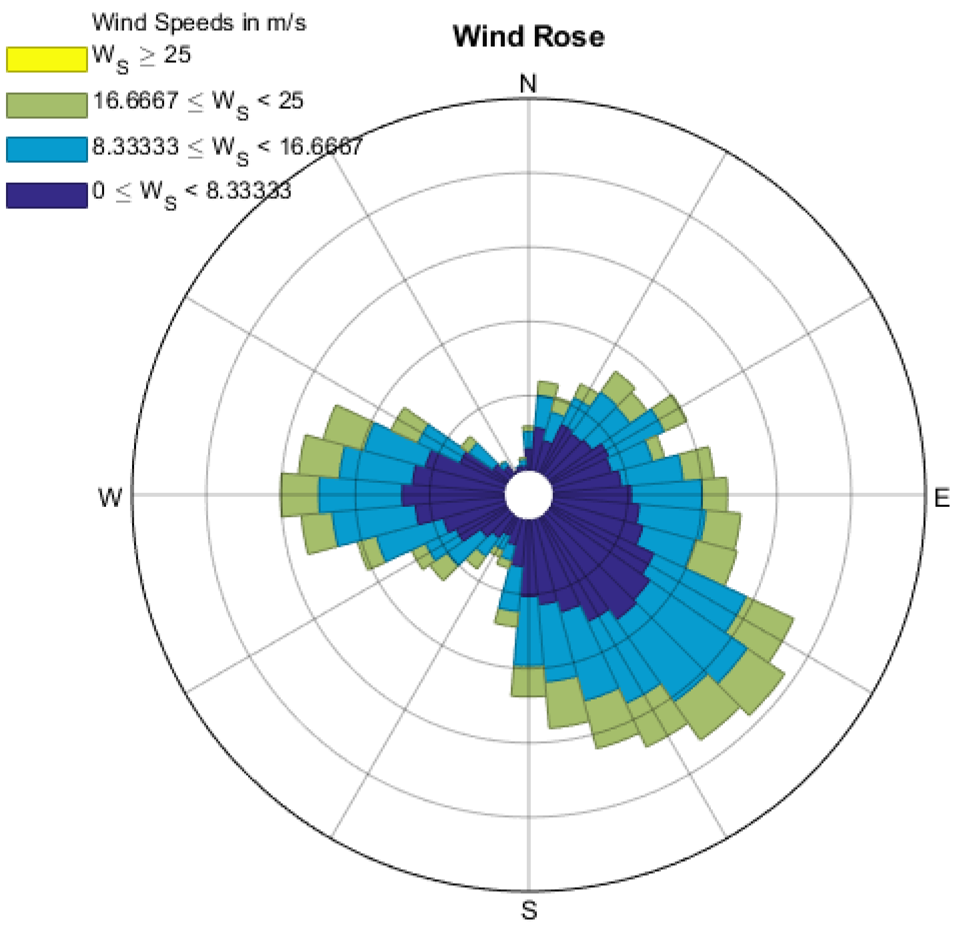
The wind rose computes the number of pointing directions in this new range, so the figure is not just rotated with respect to the original: the values are calculated in a different way, because they take into consideration different direction values.

SPEED/INTENSITY RANGES NUMBER - `'nSpeeds'`

Choose how many speed/intensity intervals should appear. An equispaced interval will be created.

Not only can the direction bins be modified. The number of bins for the intensities can be also changed:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nSpeeds', 4);
```



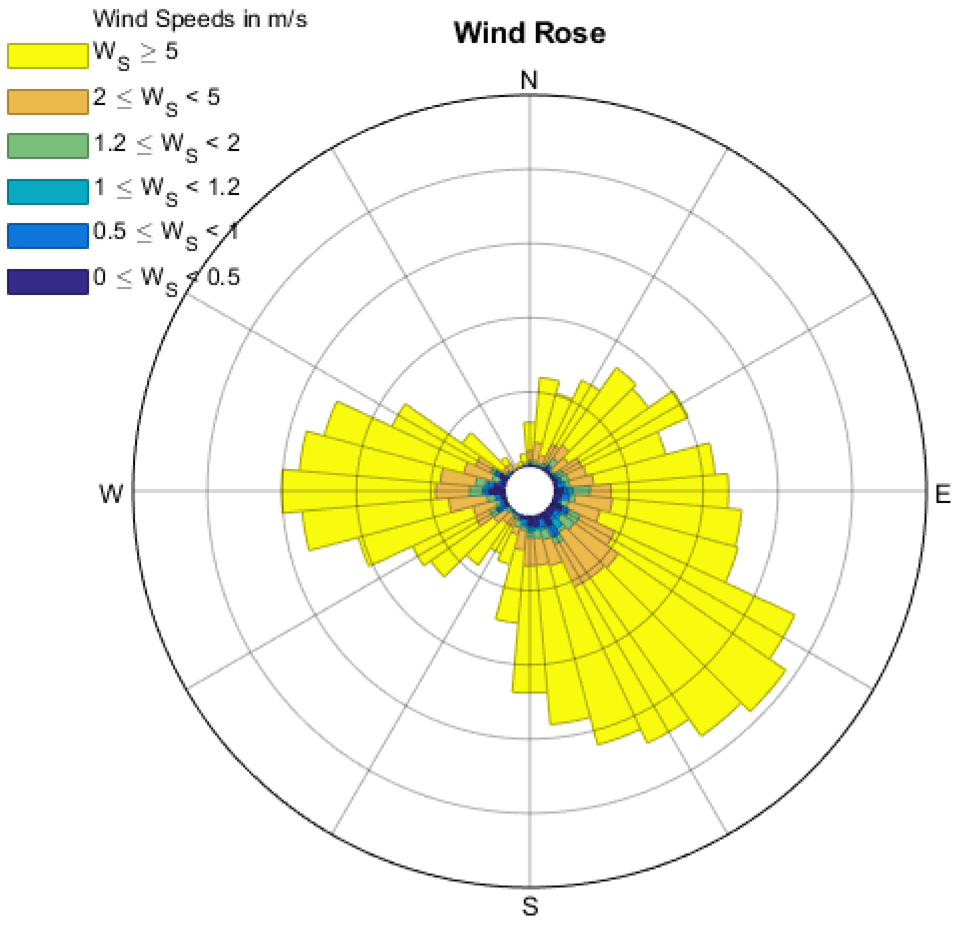
It is very common that the maximum intesity bin deos not appear in the graph (this depends on the frequency and the rounded value of the maximum).

SPEED/INTENSITY RANGES VALUES - `'vWinds'`

Choose the values of the speed/intensity intervals. Allows creting non-equispaced intervals.

If you prefer defining the speed values to create the bins, you can also do that with **`'vWinds'`**(this will omit the `'nspeeds'` command, so use whichever you need). Beware of this, since you have to know the speed ranges, in order to prevent possible errors or bins not appearing.

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'vWinds', [0 0.5
1 1.2 2 5]);
```

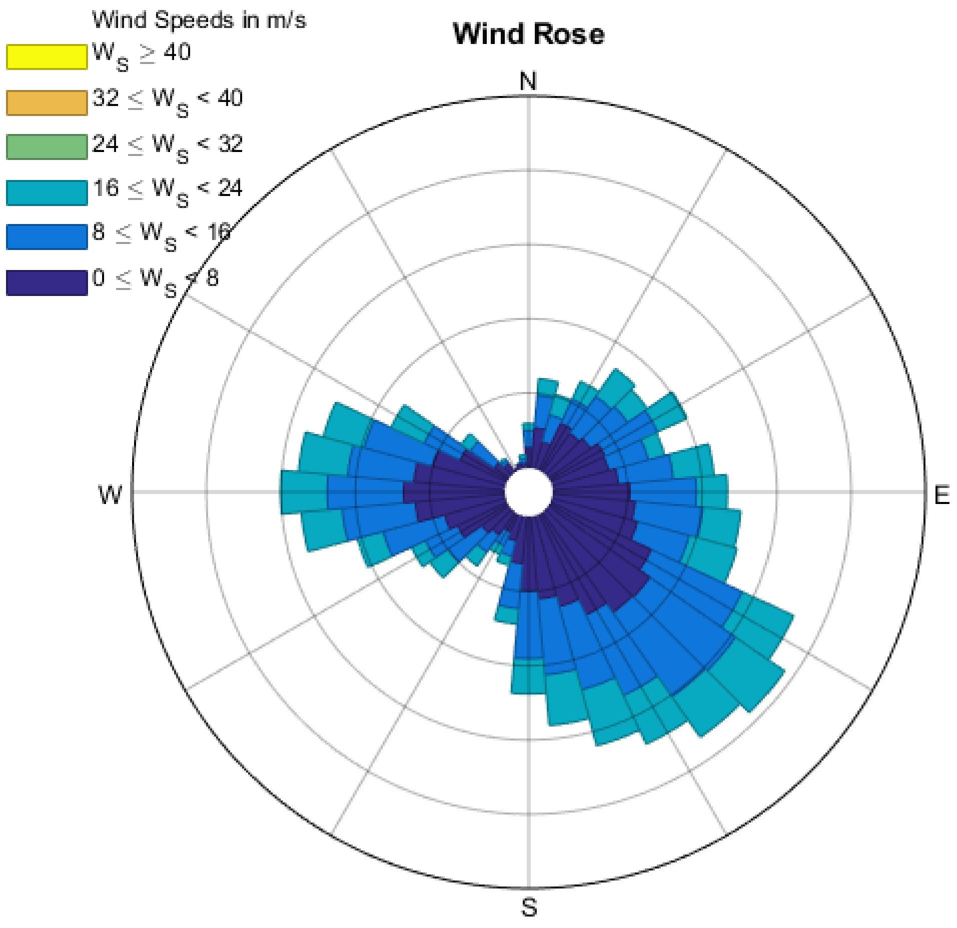


ROUND MAXIMUM SPEED/INTENSITY - 'SpeedRound'

Round the maximum speed value to a specific number.

If the maximum intensity is not defined as you wanted, change this (let's imagine that you wanted the maximum speed to be multiple of 40, with 6 intensity ranges, so all the speed ranges have integer values):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'SpeedRound', 40,
'nSpeeds', 6);
```



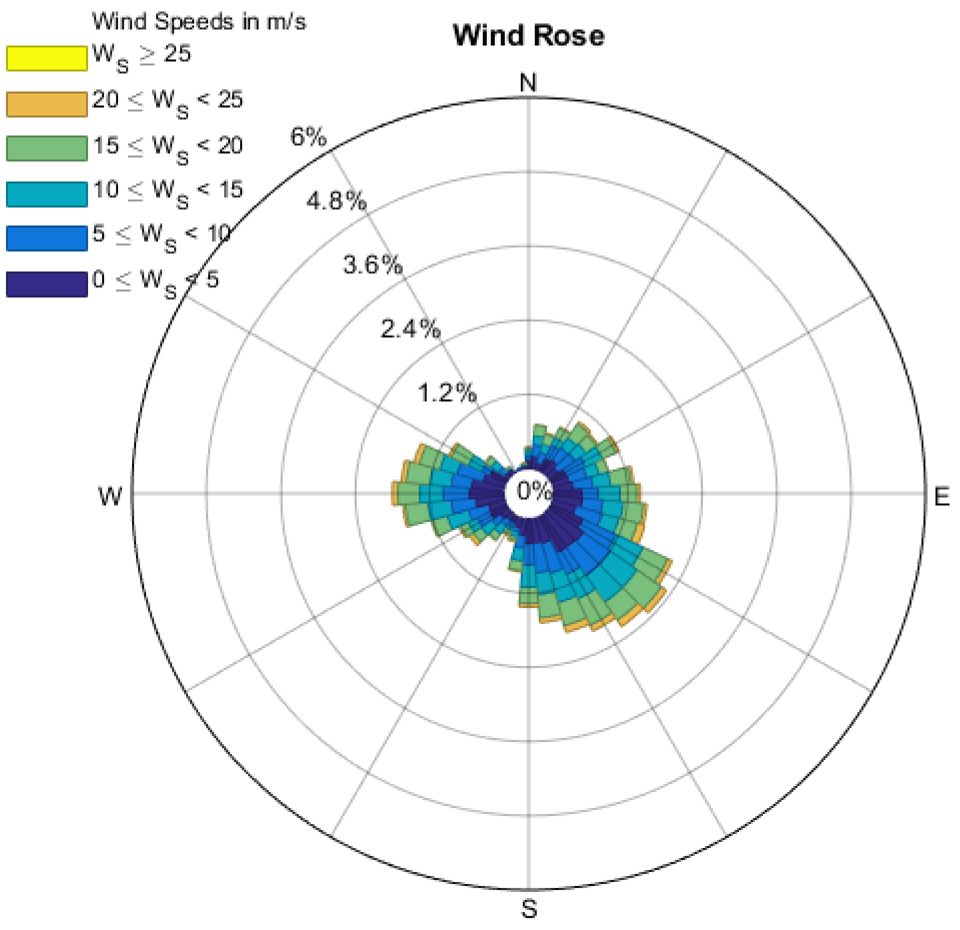
MAXIMUM FREQUENCY - `'MaxFrequency'`

Define the limit of the plot, extending the maximum frequency to this value.

The bins are drawn in a way that they fit inside the maximum circle, but, in order to compare different wind roses, it could be interesting to keep a fixed value for the maximum frequency (in percentage, 0-100).

Let's add the frequency labels too, selecting the position automatically, so we can see the frequencies that are calculated in this case and that the maximum frequency is set to 6%.

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'MaxFrequency',
6, 'FreqLabelAngle', 'auto');
```

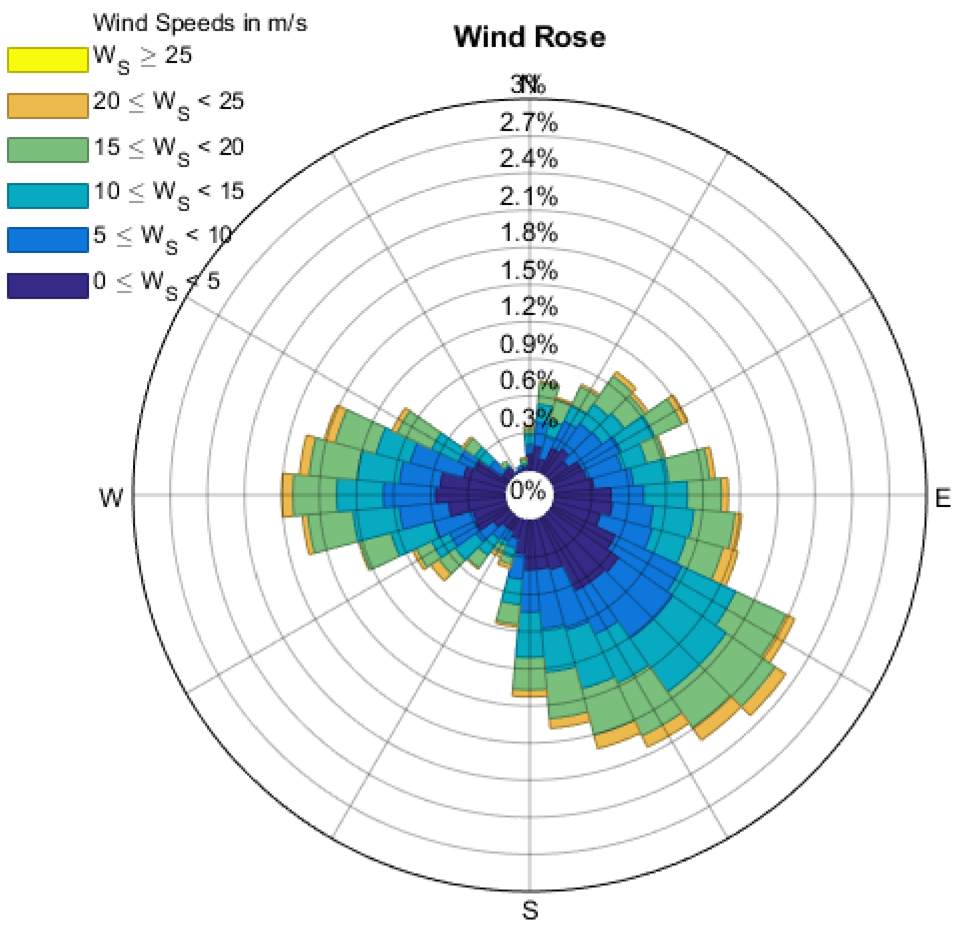


FREQUENCY/CIRCULAR GRID LINES - `'nFreq'`

Display the specified number of frequency/circular grid lines.

In order to check the **frequencies** in the final wind rose, it can be hard if data are very scattered. We can add as many frequency circles as we want. This example adds 10 frequency grid lines and the labels in the 90° (tirgonometric) line.

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nFreq', 10,
'FreqLabelAngle', 90);
```

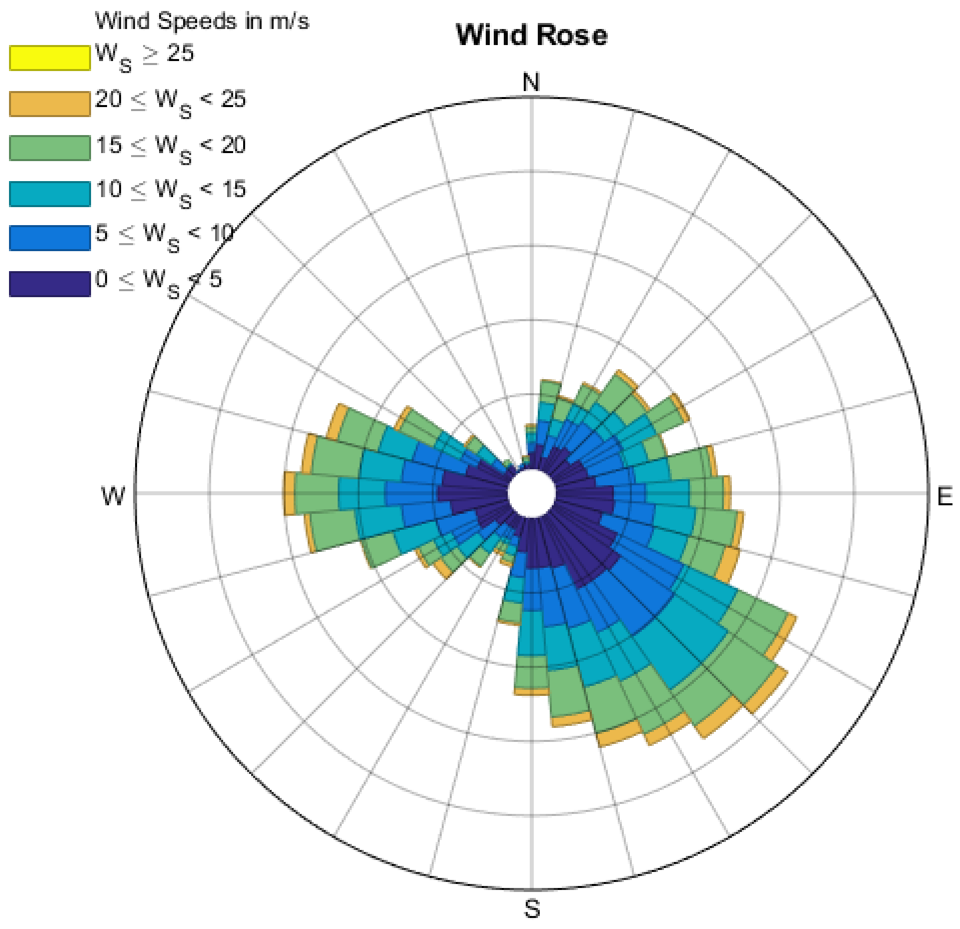


RADIAL GRID NUMBER OF DIVISIONS - `'radialGridNumber'`

Display this number of radial/sector divisions.

Add as many radial gridlines you want specifying the `'radialgridnumber'` parameter:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd,
'radialgridnumber', 24);
```



ROUND MAXIMUM FREQUENCY VALUE - `'FreqRound'`

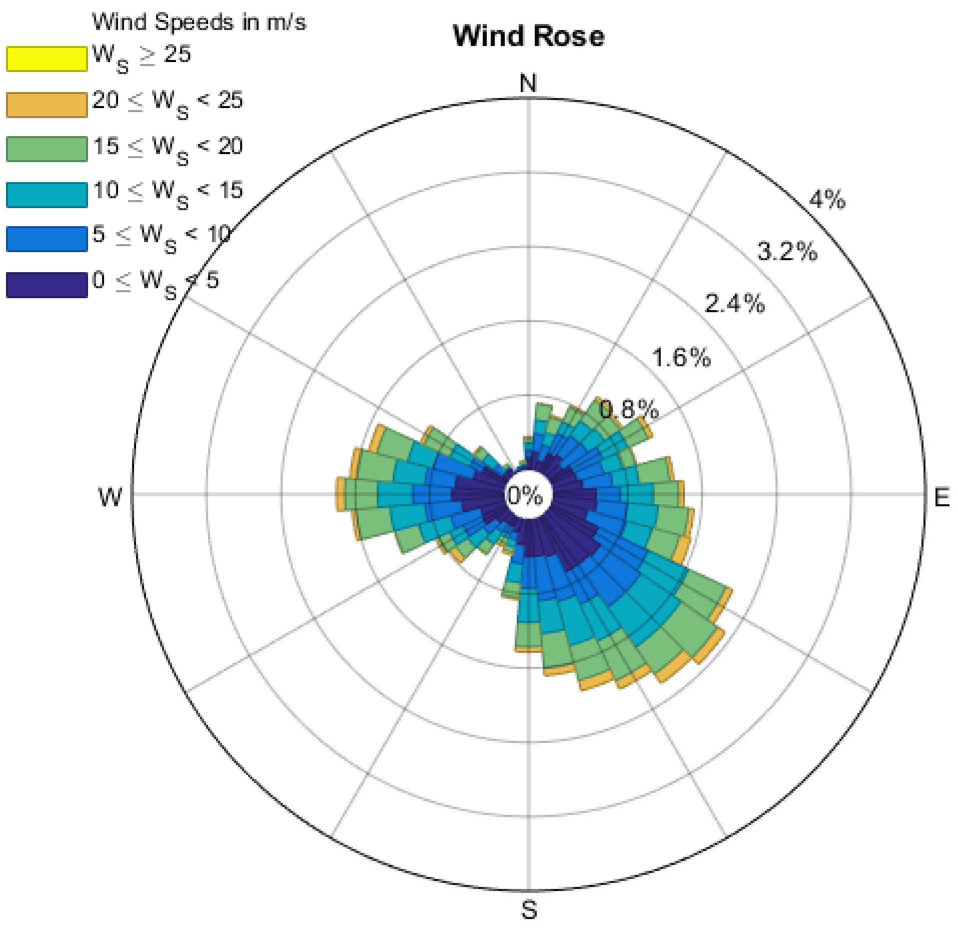
Round the maximum frequency to the input value.

The **maximum frequency value** can be rounded to a desired figure, in order to compare very different wind roses, while keeping a uniform style.

Frequency labels are shown, in order to better understand this effect.

In this example, the maximum frequency will be the lowest integer multiple of 2 which allows showing all the data:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'FreqRound', 2,
'FreqLabelAngle', 45);
```

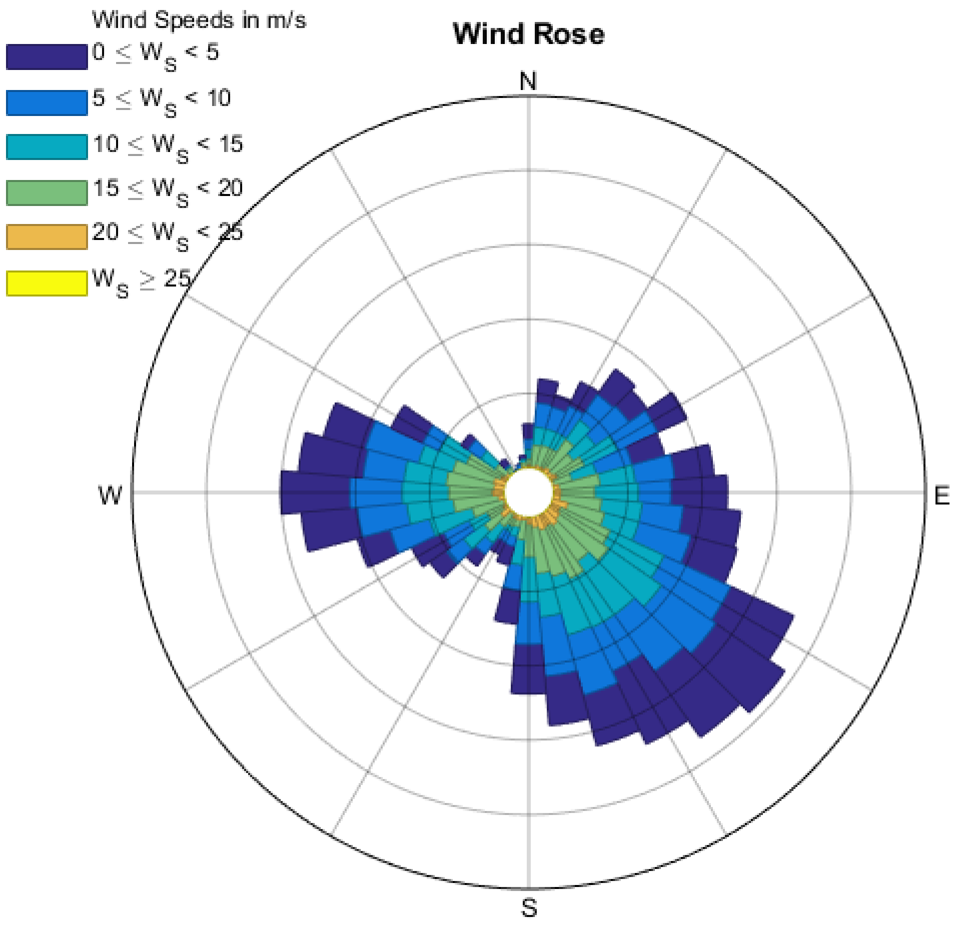


LOWEST SPEED BIN OUTSIDE - `'inverse'`

Display the lowest speed bins outside the windrose, and the highest speeds inside.

If you want to show the **lowest speeds at the outermost part** of the wind rose, there is a possibility to do this, using `'inverse', true`

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'inverse', true);
```



TITLE, LEGEND LABEL AND LEGEND MAGNITUDE - `'TitleString'`, `'LabLegend'`, `'LegendVariable'`

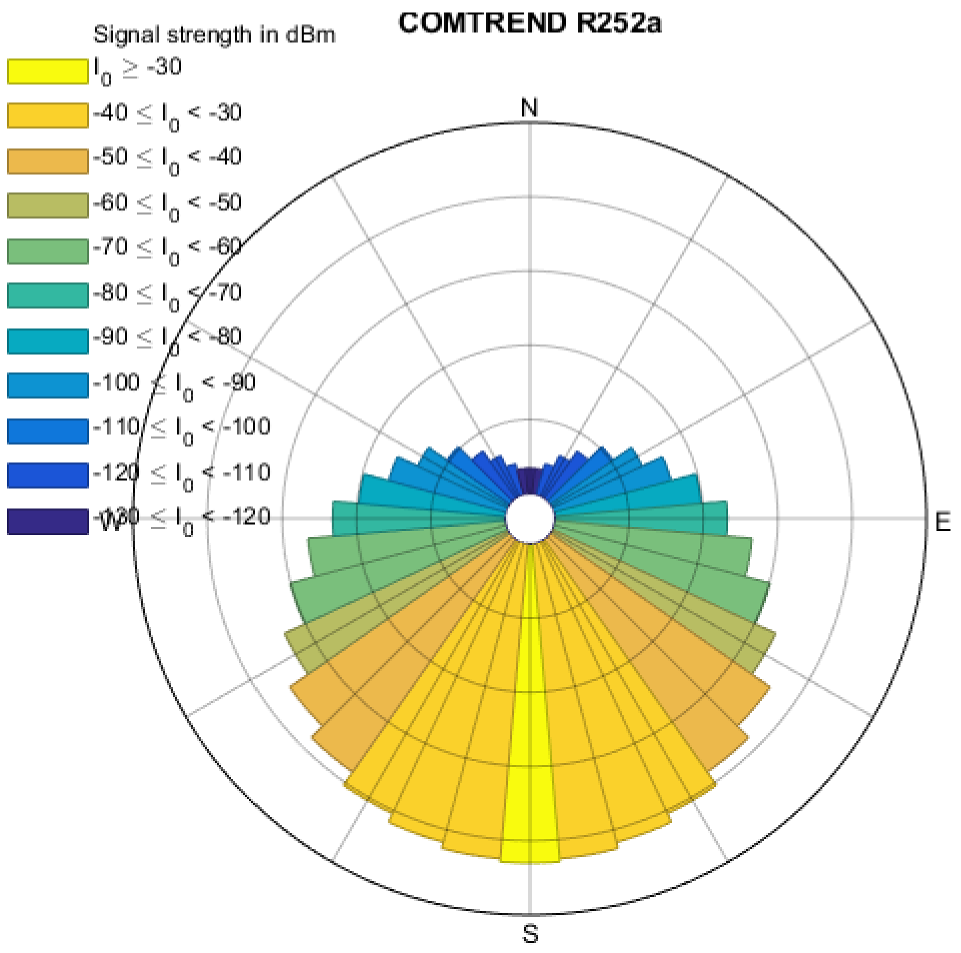
Change the default text strings in the plot, other than labels.

If you are using WindRose for other representation that does nothing to do with wind, you can change the title, the legend's label and the variable. You can use TeX strings too.

This example could be valid for a router or for a directional microphone:

```
d_r = linspace(0,350,36); int = 3*sind(-d_r)+3.5; int = int/max(int);
DIR = []; INT = []; for i=1:length(d_r); DIR = [DIR;repmat(d_r(i),round(100*int(i)),1)];
INT = [INT;repmat(100*int(i)-130,round(100*int(i)),1)]; end

[figure_handle, count, speeds, directions, Table] = WindRose(DIR, INT, 'TitleString',
'COMTREND R252a', 'LabLegend', 'Signal strength in dBm', 'LegendVariable', 'I_0'
,'zeroAndNegative', true);
```



NEGATIVE AND ZERO VALUES CONSIDERATION - `'zeroAndNegative'`

Consider negative and zero values.

Wind does not present negative values, and zeros shall be omitted from the drawing. If this is not your case, and **you want negative and zero values to appear in the wind rose**, set `'zeroAndNegative',false`. Default is true, so you don't have to worry about this.

```
[figure_handle, count, speeds, directions, Table] = WindRose(DIR, INT,
'zeroAndNegative', true);
```

MINUMUM RADIUS - `'min_radius'`

Set the minimum relative radius of the plot.

Some users do not like wind roses with a hole in the middle, but others like it a bit wider. Change this attribute by changing the min_radius value, which is relative to the circle radius.

Radius In this function is defined as:

$$r = \frac{freq + R_{min}}{1 + R_{min}}$$

$$r_{min} = \frac{R_{min}}{1 + R_{min}}$$

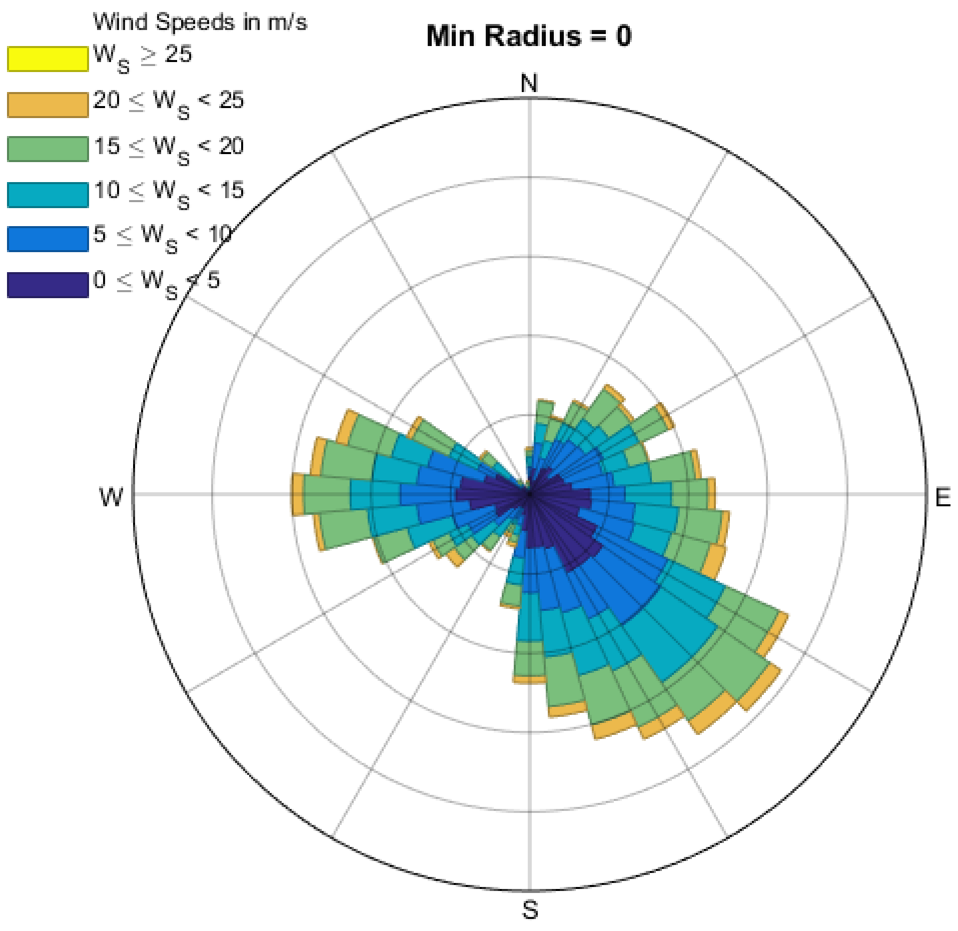
Where R_{min} is the `'min_radius'` value and r_{min} is the actual minimum radius plotted, relative to the maximum windrose radius.

In order to get a **hole 25% of the total**, use 0.25/(1-0.25) = **1/3** as min radius.

The default value for min_radius is 1/15, which implies a hole **$frac{1/15}{1+1/15} = 1/16$** of the circle).

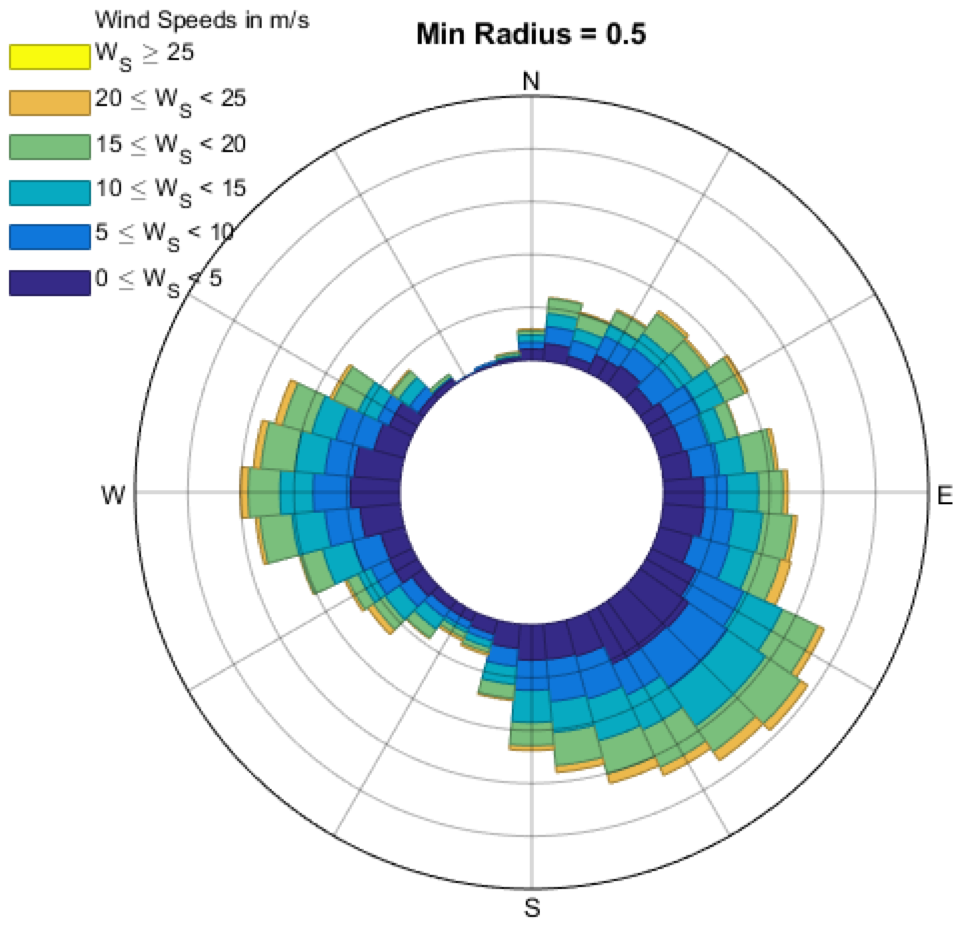
This example uses a minimum radius of 0 (no hole at the center):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'Min_Radius', 0,
'TitleString', 'Min Radius = 0');
```



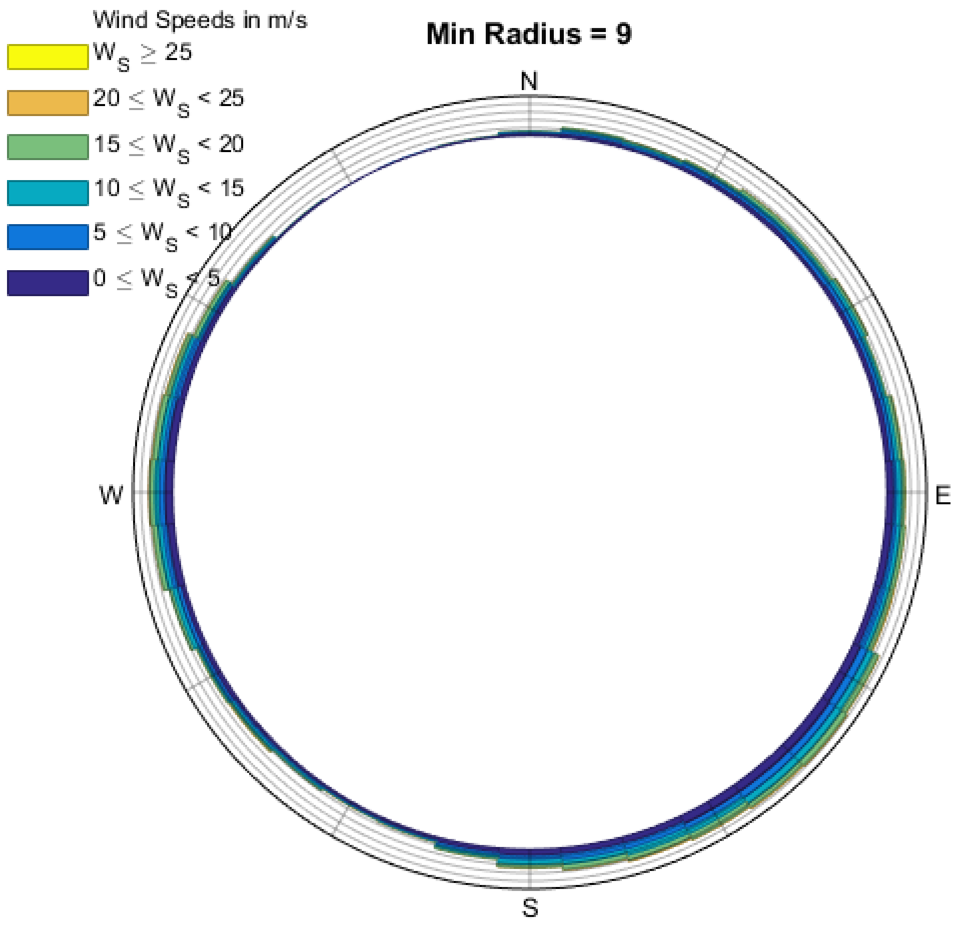
This example uses a minimum radius of 0.5 (the inner hole is 0.5/(1+0.5) = 1/3 of the radius of the windrose):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'min_radius',
0.5, 'TitleString', 'Min Radius = 0.5');
```



This example uses a minimum radius of 9 (the inner hole takes 9/(1+9) = 90% of the windrose):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'min_radius', 9,
'TitleString', 'Min Radius = 9');
```


Wind Speeds in m/s
$W_S \geq 25$
$20 \leq W_S < 25$
$15 \leq W_S < 20$
$10 \leq W_S < 15$
$5 \leq W_S < 10$
$0 \leq W_S < 5$
Min Radius = 9
N
E
S
W

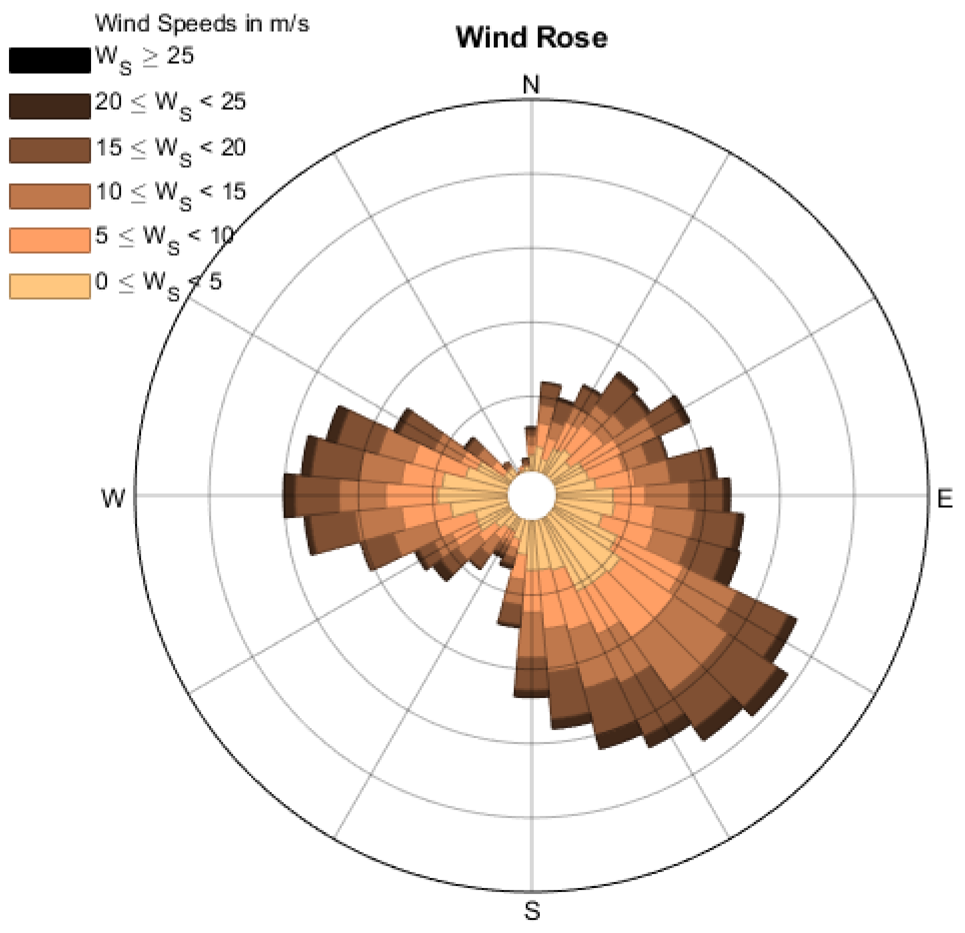
COLORMAP - `'cMap'`

Change the color scheme for this windrose.

If you hate the "parula or jet" colormaps -default, depending on whether you have parula available or not-, you can use any other built in colormap you want, even if it is defined in a Matlab function in your working paths (this means that the function you use does not necessarily be part of Matlab) You can also invert the colormap just by adding "inv" at the beginning of the colormap name.

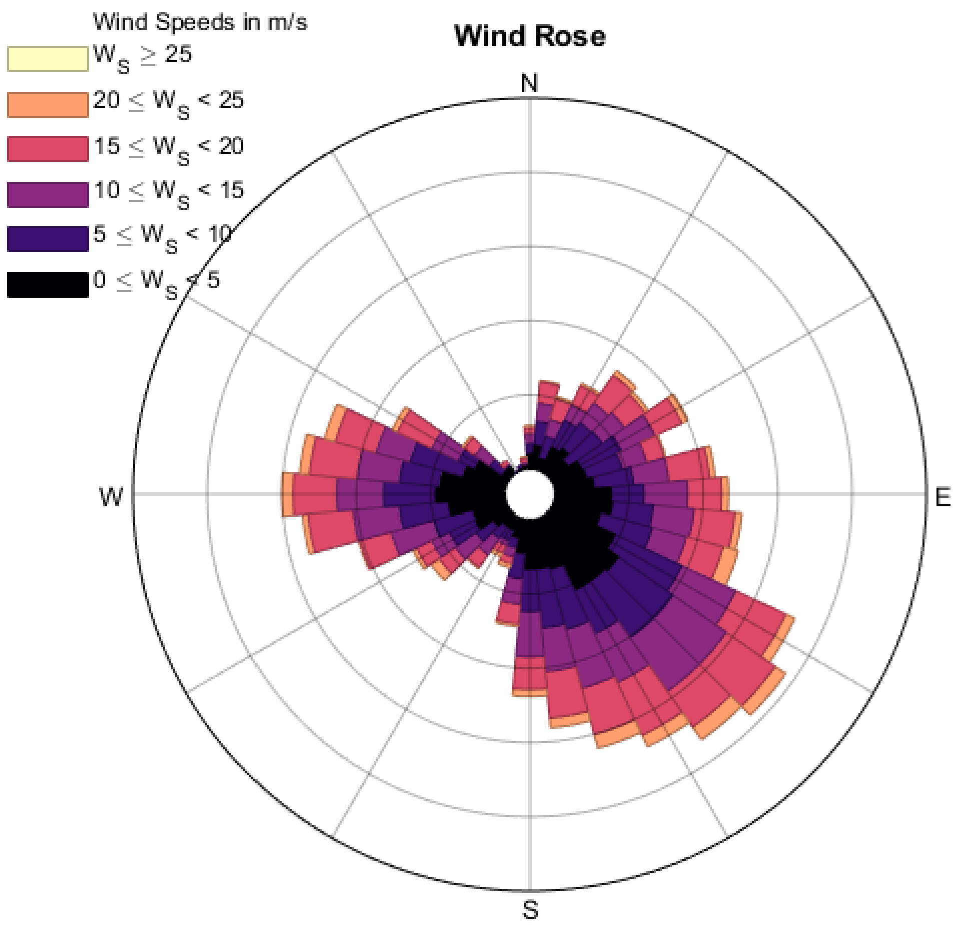
In this example, we will use the "copper" colormap but inverted:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'cMap',
'invcopper');
```



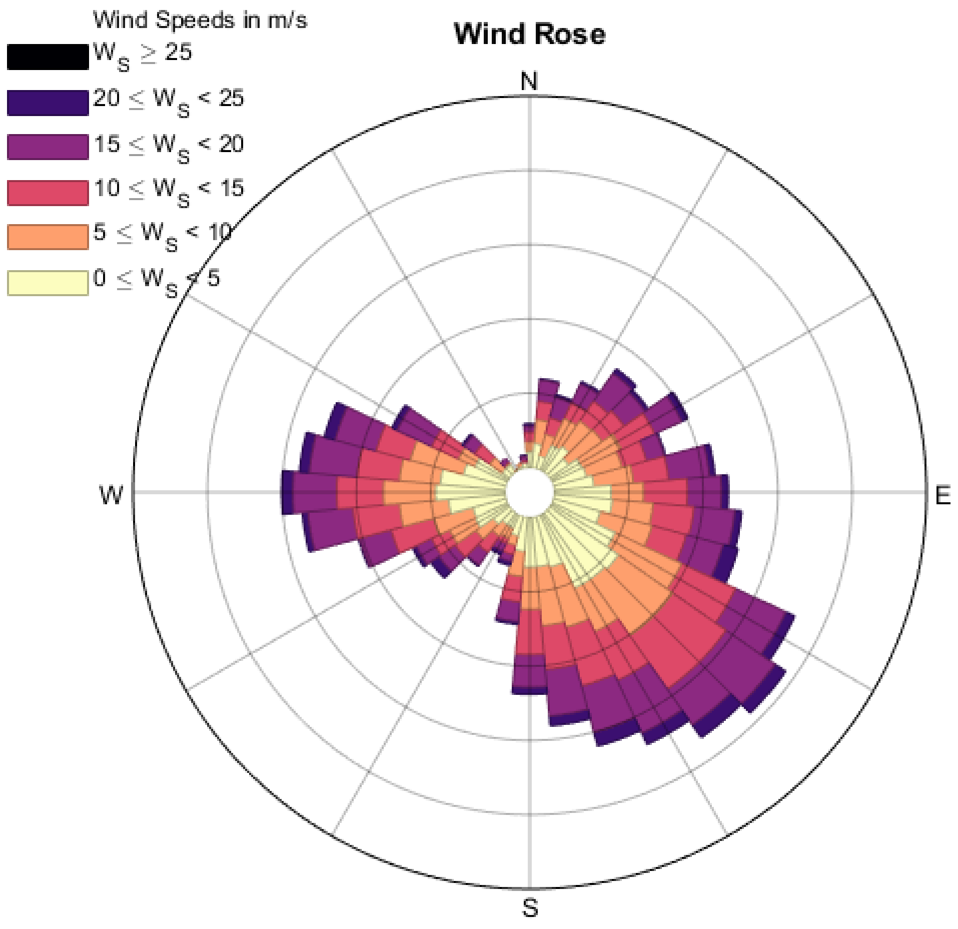
But we can also use my favourite colormap, "magma", which I have as a file in my Matlab path:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'cMap', 'magma');
```



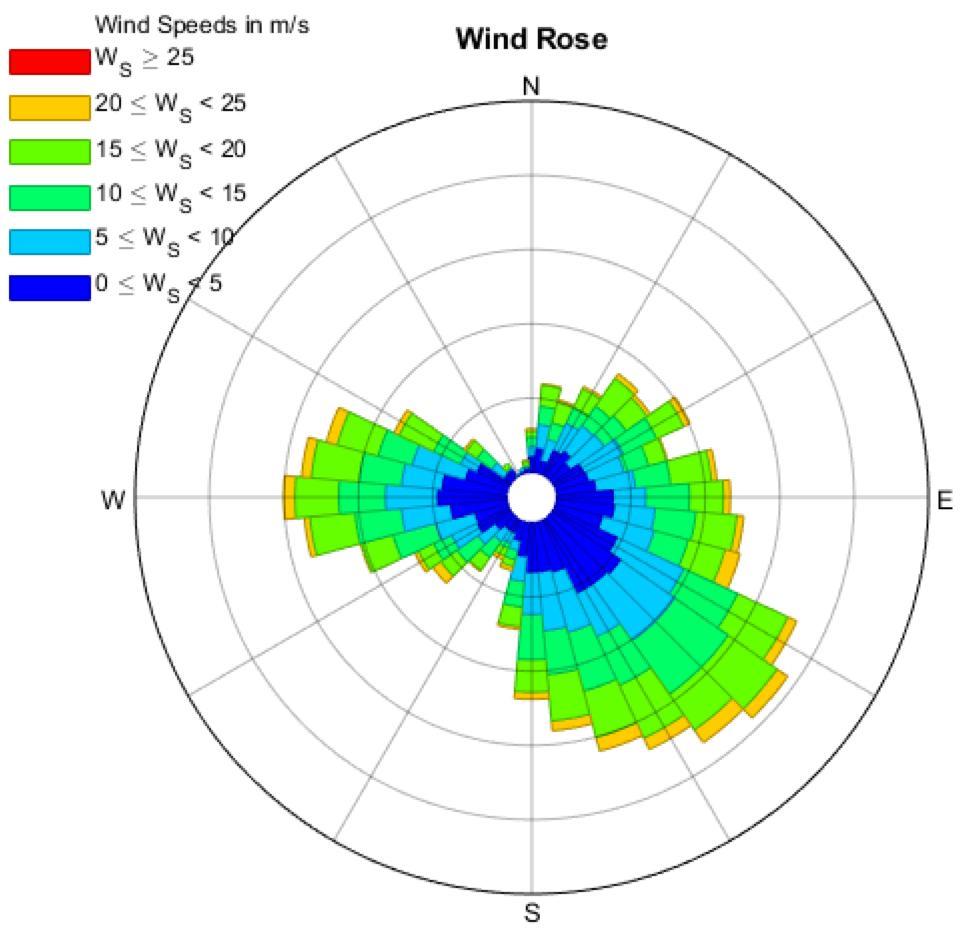
We can also invert this colormap by calling it invmagma (despite invmagma is not an actual function):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'cMap',
'invmagma');
```



You can also use a colormap of your own by using a n×3 array. This, compared to the following option, does not need setting the number of speeds, but may result in something different from what you expect, since these are inteporlated from the data you pass to the function.

```
mycolormap = [0 0 1; 0 1 1; 0 1 0; 1 1 0; 1 0 0]; % Blue, Cyan, Green, Yellow, Red
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'cMap',
mycolormap);
```



Note that despite we put yellow in the colormap, it does not appear finally, since the final colormap is an interpolation of 7 colors from the 5 we specified. See the next section for a different approach.

COLORS - 'colors' + 'nSpeeds' or 'vWinds'

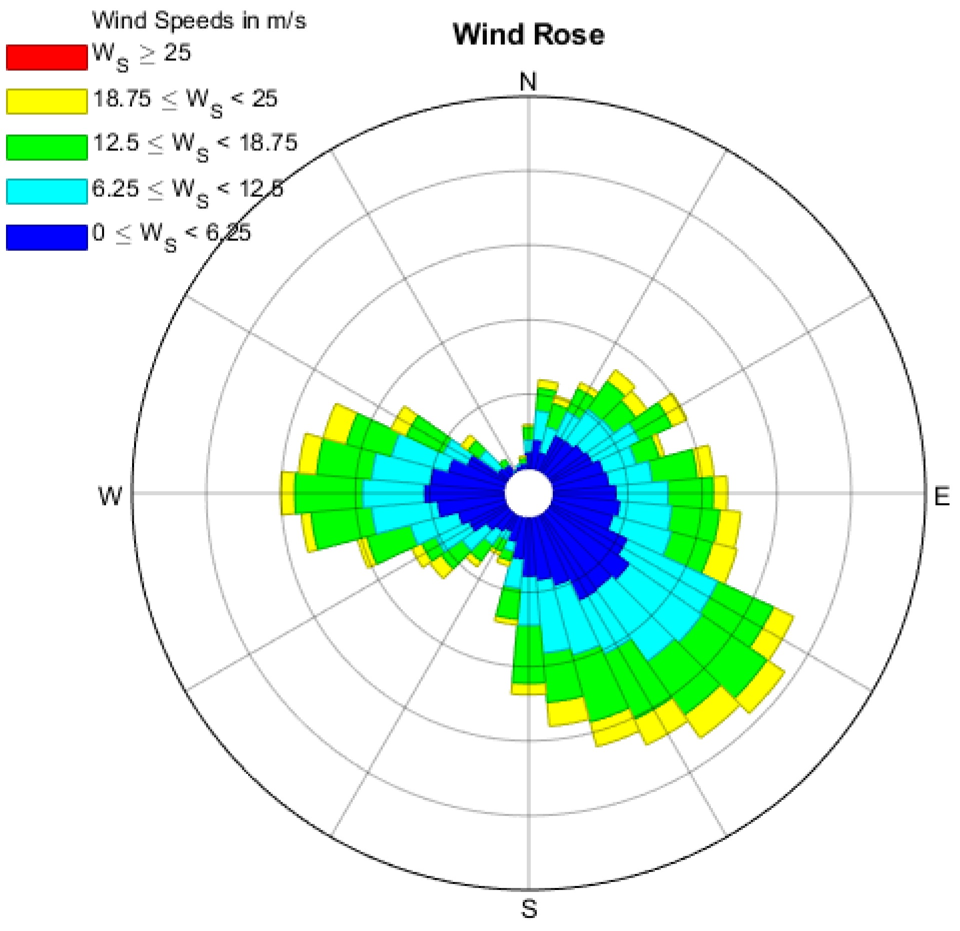
Set specific colors for specific intervals.

If you specify the `'nSpeeds'` or the `'vWinds'` argument to define the intensity intervals, a customized and fixed final colormap without interpolation can be used, specifying a numeric array for 'colors'.

It is compulsory to specify the number of speed bins, and must match the number of colors.

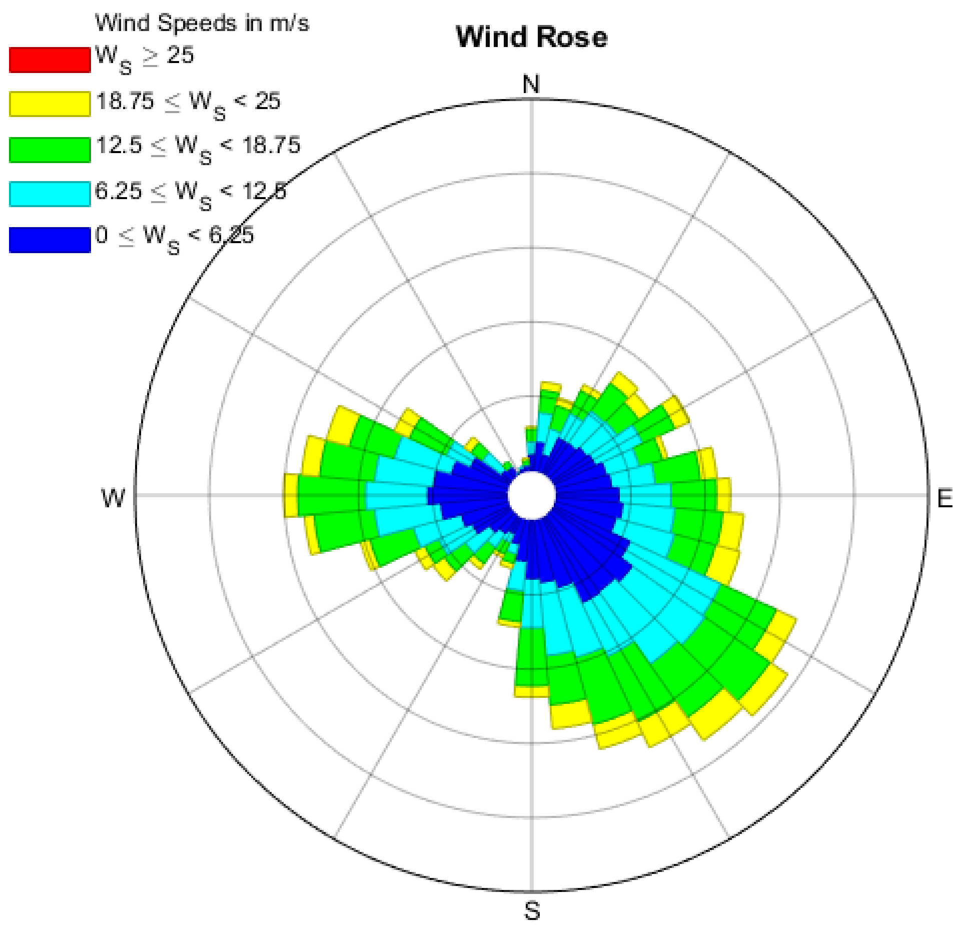
If we want the 5 colors we specified in the previous section to appear without interpolation, this is the appropriate function call:

```
mycolormap = [0 0 1; 0 1 1; 0 1 0; 1 1 0; 1 0 0]; % Blue, Cyan, Green, Yellow, Red
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nSpeeds', 5,
'colors', mycolormap);
```



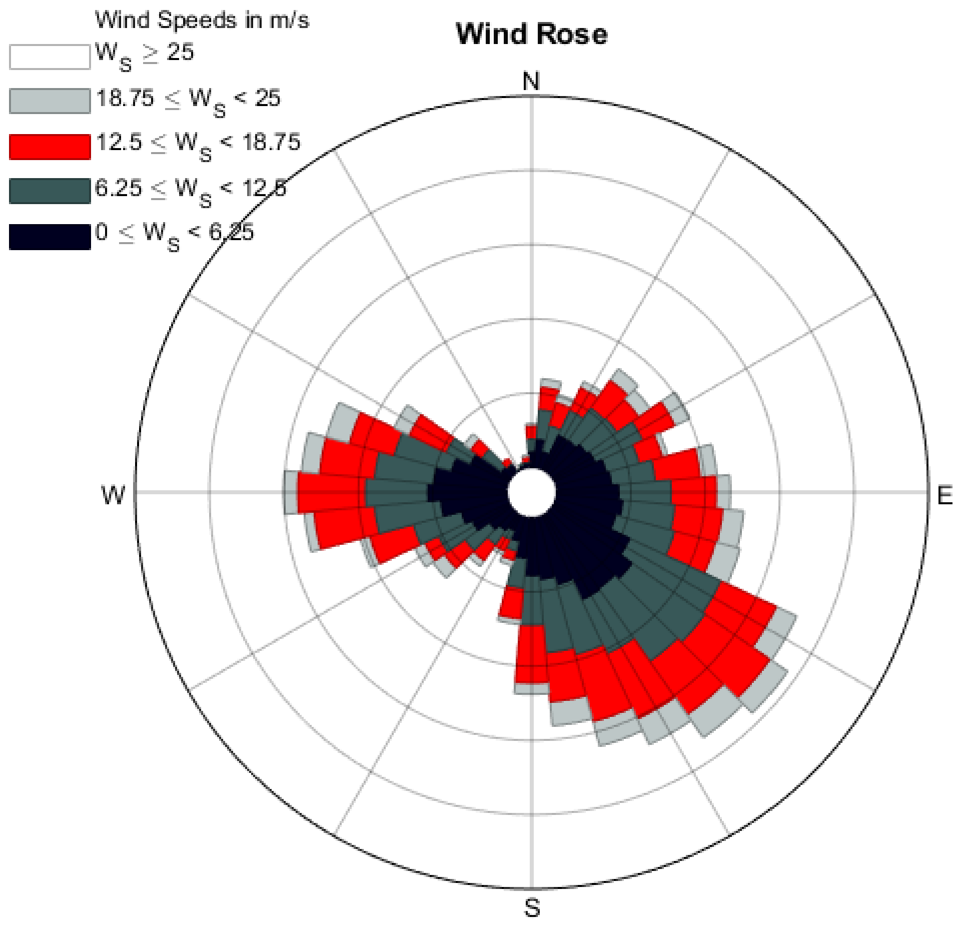
In fact, it is the same as using `'cmap'` with `'nspeeds'` instead of `'colors'` with `'nspeeds'`:

```
mycolormap = [0 0 1; 0 1 1; 0 1 0; 1 1 0; 1 0 0]; % Blue, Cyan, Green, Yellow, Red
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nSpeeds', 5,
'cMap', mycolormap);
```



This method is more interesting than using a colormap if you want to remark certain speed range. Using `‘legendtype’,1` does not make much sense for custom colors, but it does in custom colormap (previous section). See **"Changing the colormap"** in previous section for more colouring options:

```
mycolormap      = bone(5);
mycolormap(3,:) = [1 0 0];
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'nSpeeds',
5,'colors', mycolormap);
```


Wind Speeds in m/s
$W_S \geq 25$
$18.75 \leq W_S < 25$
$12.5 \leq W_S < 18.75$
$6.25 \leq W_S < 12.5$
$0 \leq W_S < 6.25$
Wind Rose
N
E
S
W

LEGEND - 'LegendType'

Change the legend type or remove the legend.

The legend can be two types: Boxes legend -default- (`'LegendType',2`) or colorbar legend (`'LegendType',1`). Since version 2021/04/20, the colorbar legend also shows the variable name. If you do not want the legend to appear, consider there is an extra type (`'LegendType',0`):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'LegendType', 1);
```

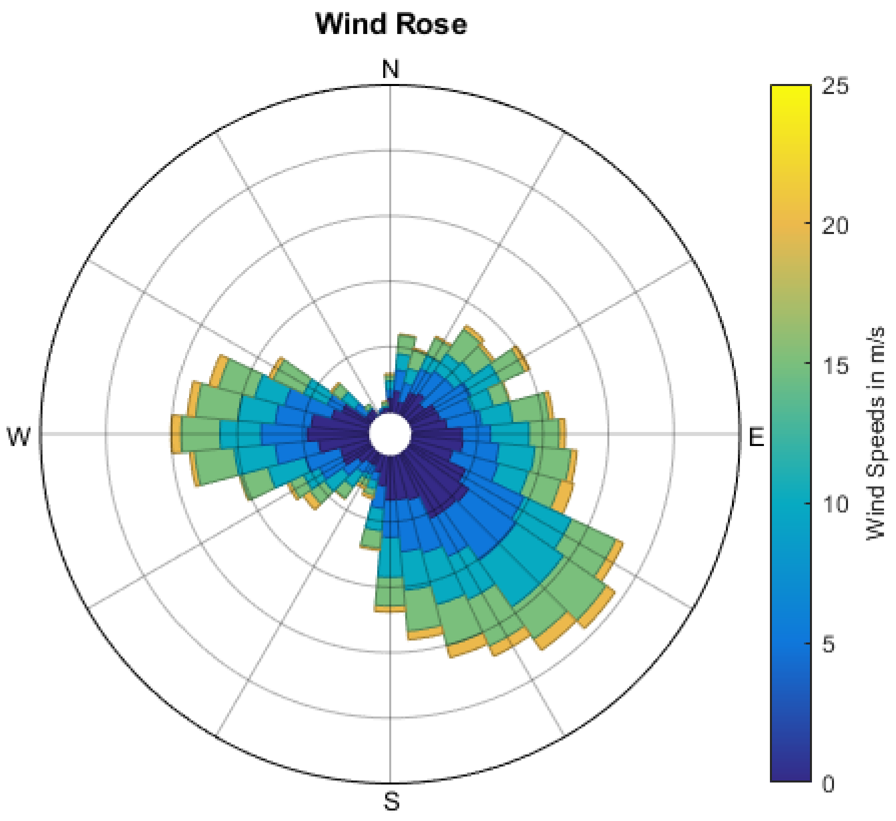
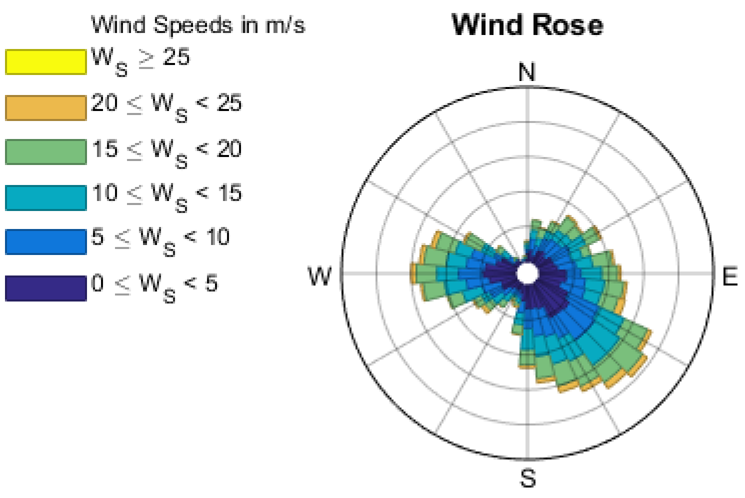


FIGURE SIZE - `'width'` and `'height'`

Change figure inner dimensions.

The default figure will be squared, with the length of the square being 2/3 of the smallest dimension of the screen (usually the height). You can specify the figure dimensions in pixels. These dimensions will be the inner figure dimensions (window border is added internally, so the axes are exactly the size you define):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'height', 256,
'width', 512);
```



SCALE FACTOR - `'scalefactor'`

Change the size of the windrose inside the figure.

If you consider that the wind rose is too big inside the figure, you can reduce its size by using a scale factor (0 to 1):

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'scalefactor',
0.5);
```

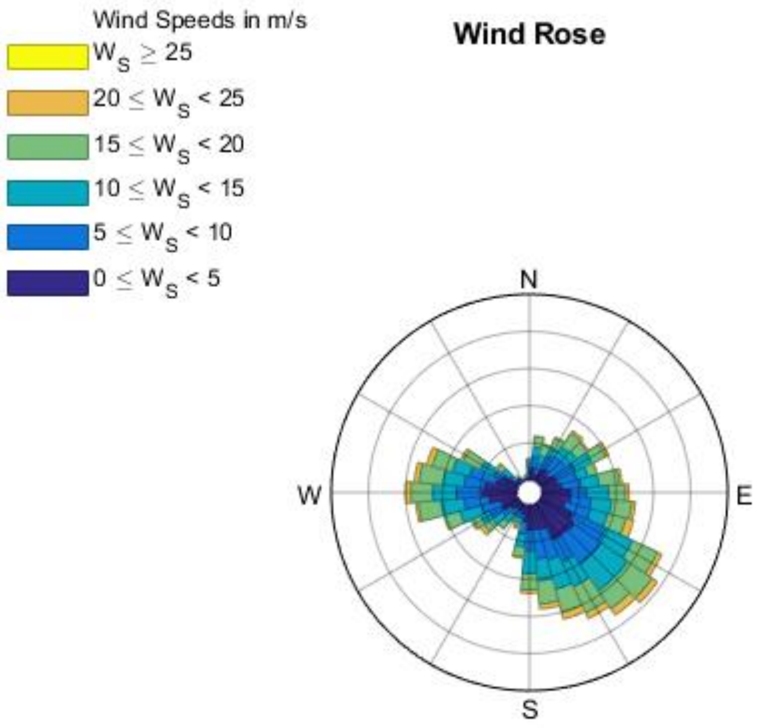


FIGURE TOOLBARS - `'menubar'` and `'toolbar'`

Change or hide the figure's menubar and toolbar.

If you want to remove the figure and menu bars, you can use the same instructions used in a normal figure:

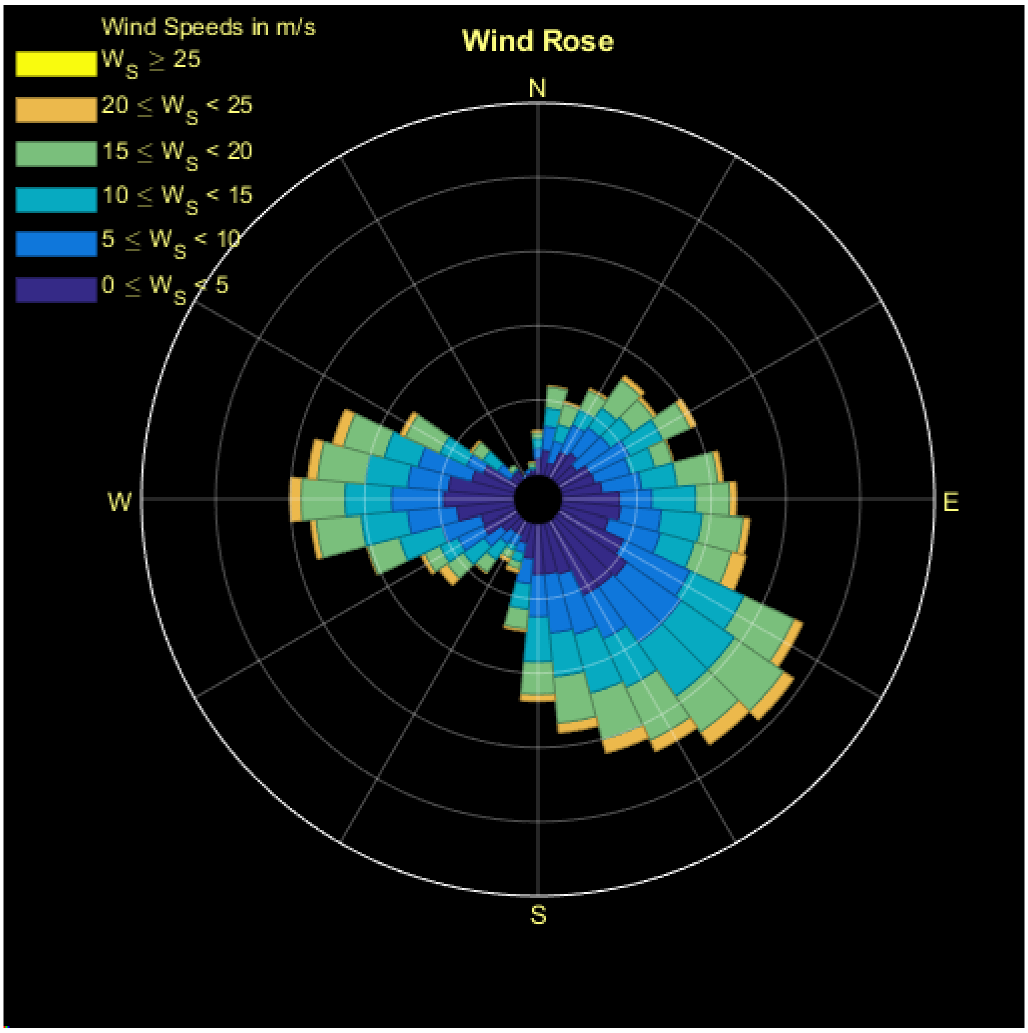
```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'menubar',
'none', 'toolbar', 'none');
```

FIGURE, TEXT AND GRID COLORS - `'figColor'`, `'textColor'`, `'gridColor'`

Change figure, text or grid colors.

Figure is white by default; while text appears to be black. Modify them as wanted. Use Matlab built-in colors (*red* `'r'`, *green* `'g'`, *blue* `'b'`, *white* `'w'`, *black* `'k'`, *yellow* `'y'`, *magenta* `'m'`, *cyan* `'c'`) or use custom RGB vector, both for figure, text and grid lines:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'figColor', 'k',
'textColor', [1 1 0.5], 'gridColor', 'w');
```



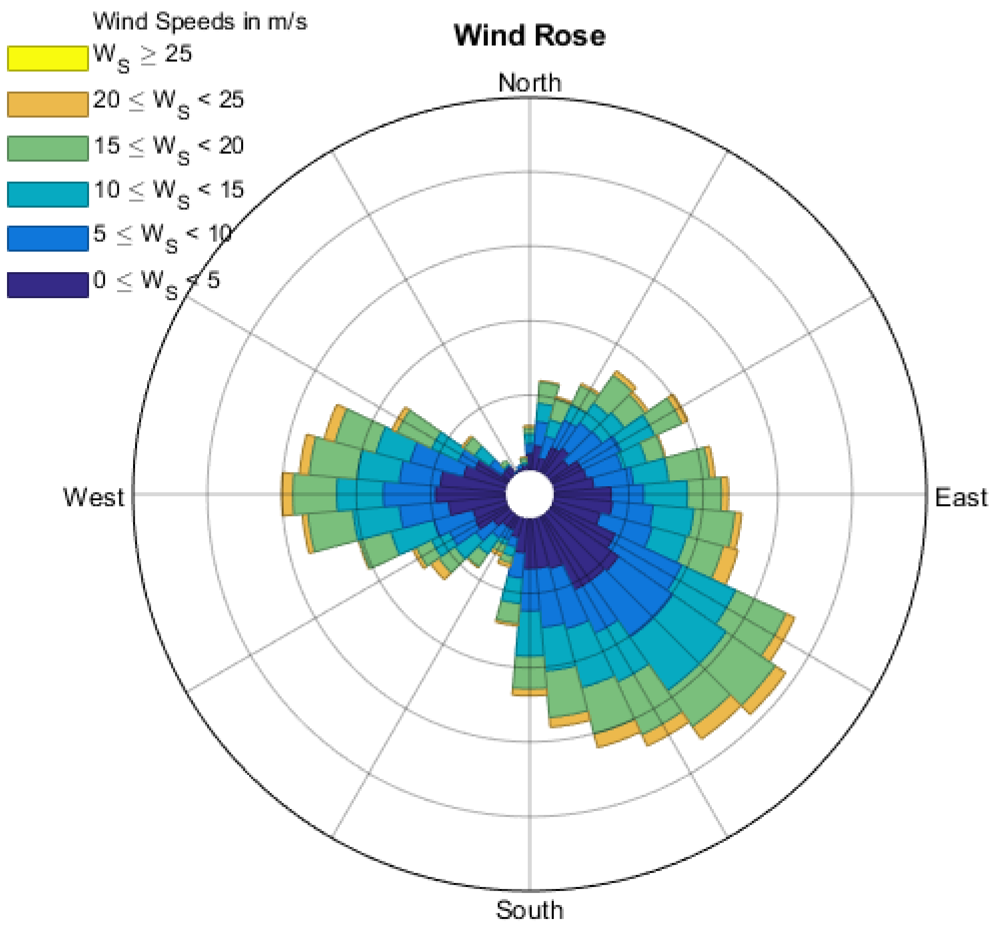
AXIS LABELS - `'labels'`

Change direction labels. Any number of labels is allowed now.

Axis labels - *default are N, E, S and W* - can be modified with separate specific properties or with a cell array `{'Label for North', 'Label for East', 'Label for South', 'Label for West'}`

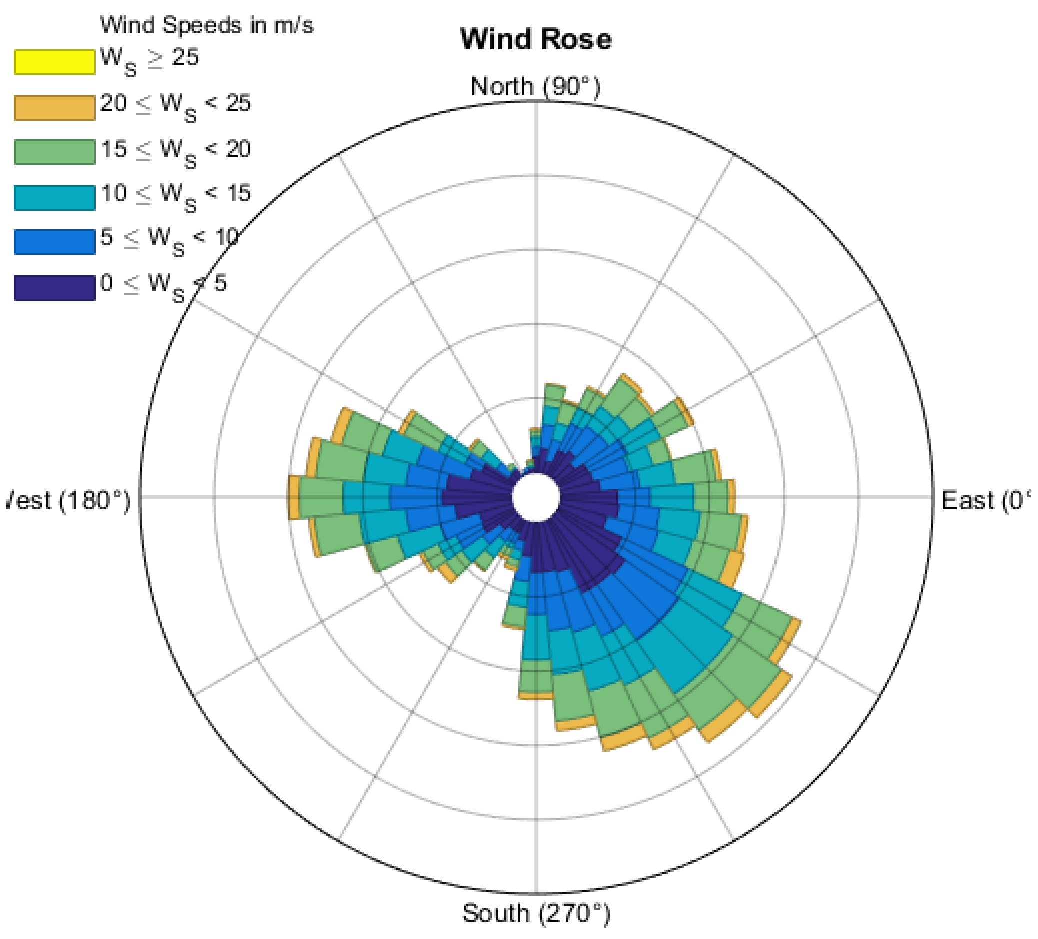
Put labels separately:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'labelNorth',
'North', 'labelSouth', 'South', 'labelEast', 'East', 'labelWest', 'West');
```



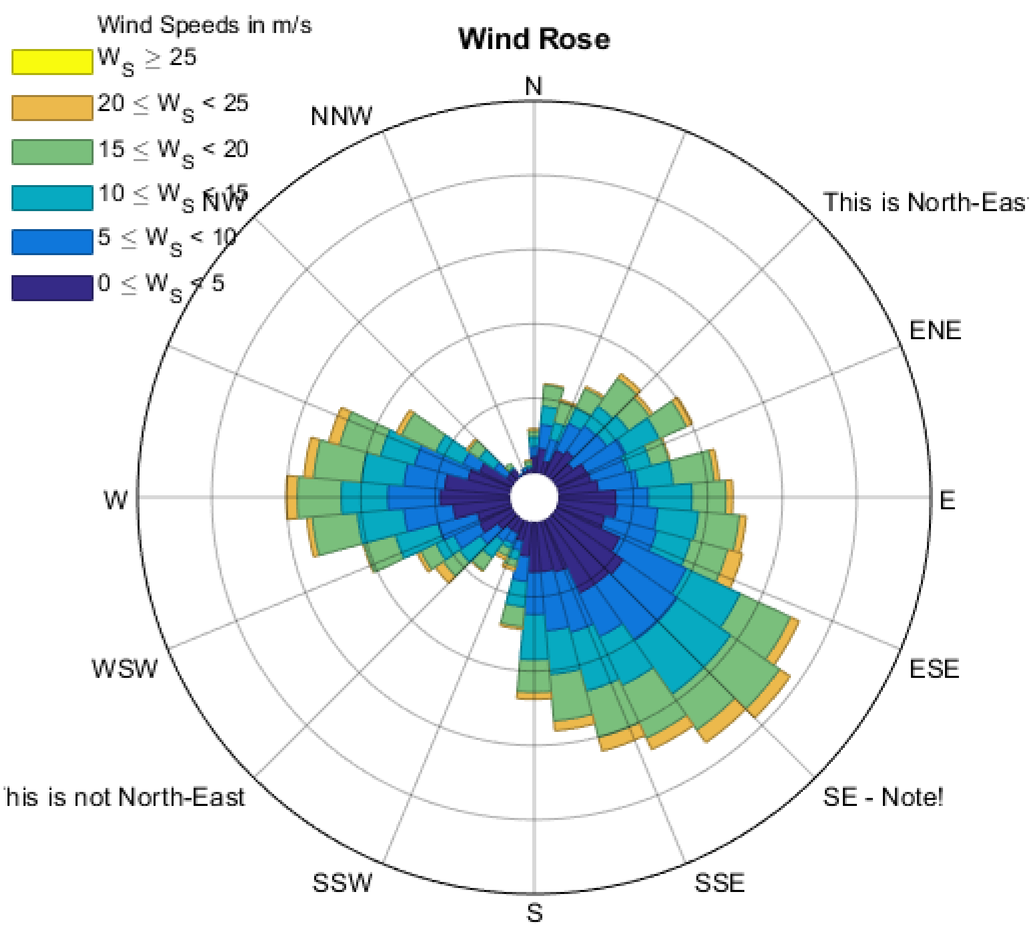
Specify labels together (North, East, South, West) ← *NOTE THAT THE ORDER IN 2015 VERSIONS WAS (North, South, East and West)*:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'labels', {'North
(90° )', 'East (0° )', 'South (270° )', 'West (180° )'});
```



If we specify **any other number of labels in the form of a cell array**, these will be distributed from the top, clockwise. Note that the radial divisions are automatically set to the number of labels (unless `'radialgridnumber'` is set to a different value). Any string (empty characters too) can be put into this cell array, thus omitting that label in the graph. See the example below:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, 'labels', {'N',
'', 'This is North-East', 'ENE',...
    'E', 'ESE', 'SE - Note!', 'SSE', 'S', 'SSW', 'This is not North-East','WSW', 'W',
'', 'NW', 'NNW'});
```



Note that **16 gridlines are automatically added for the 16 labels** specified.

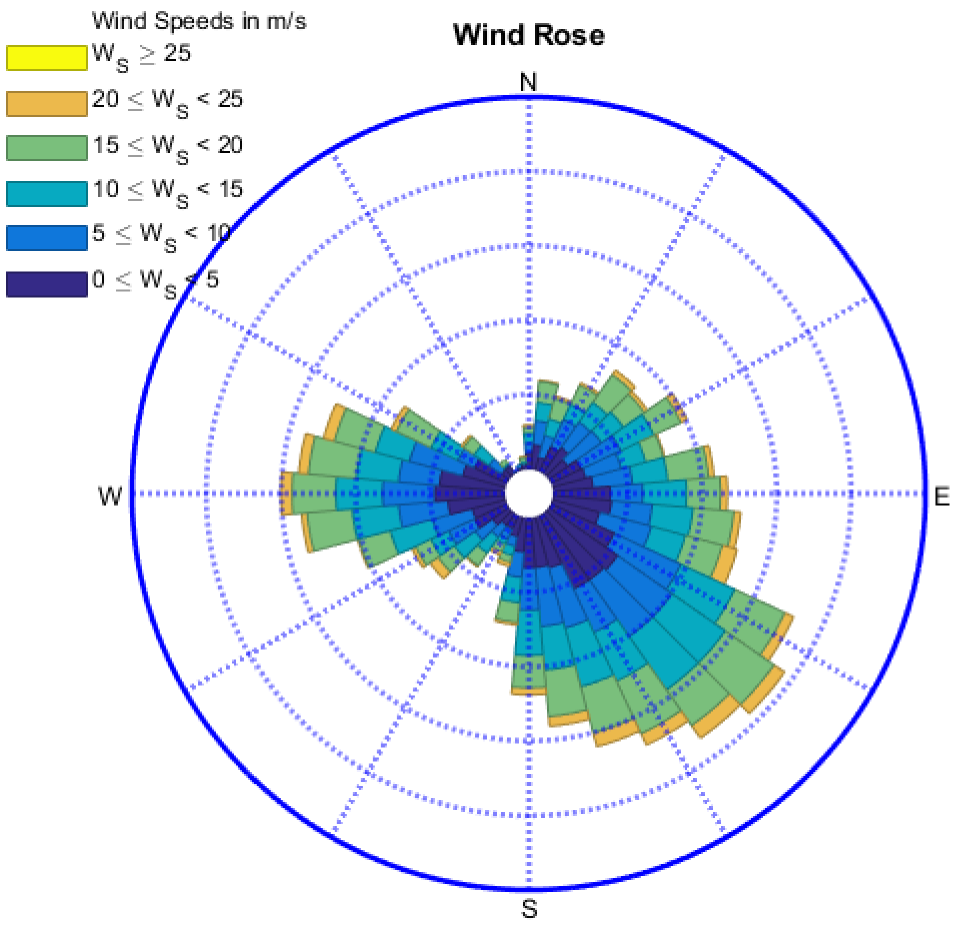
GRID LINE STYLES - 'gridStyle', 'gridColor', 'gridWidth', 'gridalpha'

Change the grid line style, color, width and opacity.

Grid lines can be styled together or separately, indicating ***line color*** (`'gridColor'`), ***line style*** (`'gridStyle'`), ***line width*** (`'gridWidth'`) and ***line alpha (opacity)*** (`'gridalpha'`) (*alpha is only available for Matlab versions 8.4 or greater*).

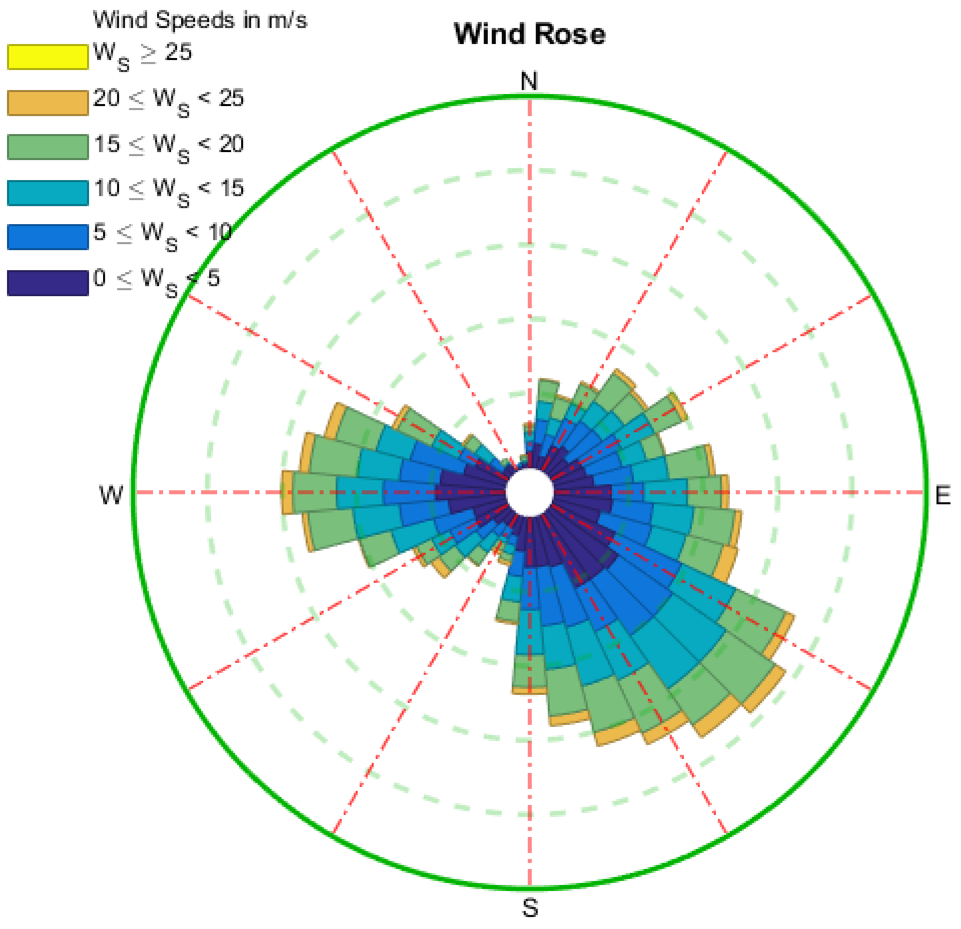
Style grid lines (radial and circular) together:

```
Options1 = {'gridstyle', ':', 'gridcolor', 'b', 'gridwidth', 2, 'gridalpha', 0.5};
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options1);
```



Or style them separately:

```
Options2 = {'radialgridstyle', '-.', 'circulargridstyle', '--', ...
    'radialgridcolor', 'r', 'circulargridcolor', [0 0.7 0]...
    , 'radialgridwidth', 1, 'circulargridwidth', 2, ...
    'radialgridalpha', 0.75, 'circulargridalpha', 0.25};
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options2);
```


Wind Speeds in m/s
$W_S \geq 25$
$20 \leq W_S < 25$
$15 \leq W_S < 20$
$10 \leq W_S < 15$
$5 \leq W_S < 10$
$0 \leq W_S < 5$
Wind Rose
N
W
E
S

SUBPLOT - 'axes'

Create wind roses in different axes (in the same or in different figures).

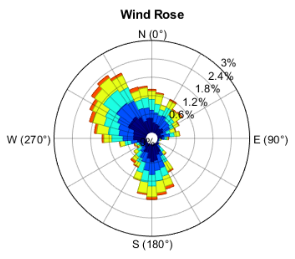
To create wind roses in **different axes** use the 'axes' argument, followed by the handle of the axes in which to plot the wind roses. Different colormaps can be used in each windrose without extra code.

```
rng(22081983);
[spd2, dir2] = WindRandomDistrib(8760, 20);
rng(02062020);
[spd3, dir3] = WindRandomDistrib(8760, 20);

Options = {'anglenorth', 0, 'angleeast', 90, 'labels', {'N (0° )', 'E (90° )', 'S (180°
)', 'W (270° )'}, 'freqlabelangle', 45, 'legendtype', 0};
```

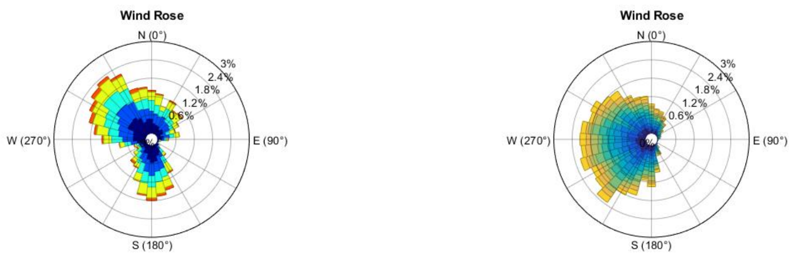
Option 1: Call the subplot and set the windrose axes to be Current axes (gca) (this should be the favorite for almost everyone).

```
subplot(2, 2, 1); set(gcf, 'units', 'normalized', 'position', [0 0 1 1]);
Options1 = [Options, {'axes', gca, 'cmap', 'jet'}]; % This is the simplest option to
concatenate cell arrays. It's the same as using Options1 = {'legendtype',0,'axes',gca};
[figure_handle, count1, speeds1, directions1, Table1] = WindRose(dir, spd, Options1);
```



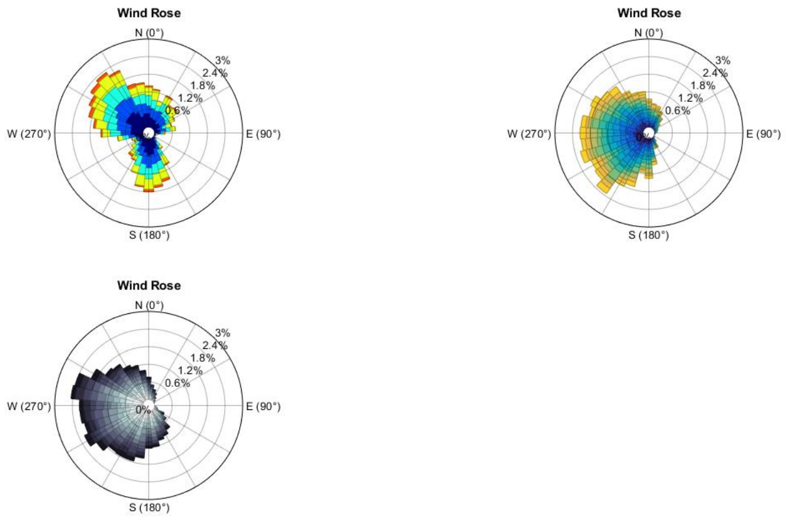
Option 2: Assign the subplot/subaxes handle to a variable.

```
axes1 = subplot(2, 2, 2);
Options2 = [Options, {'axes', axes1, 'cmap', 'parula'}];
[figure_handle, count2, speeds2, directions2, Table2] = WindRose(dir2, spd2, Options2);
```



Option3: Call the subplot inside the options directly.

```
Options3 = [Options, {'axes', subplot(2, 2, 3), 'cmap', 'invbone'}];
[figure_handle, count3, speeds3, directions3, Table3] = WindRose(dir3, spd3, Options3);
```

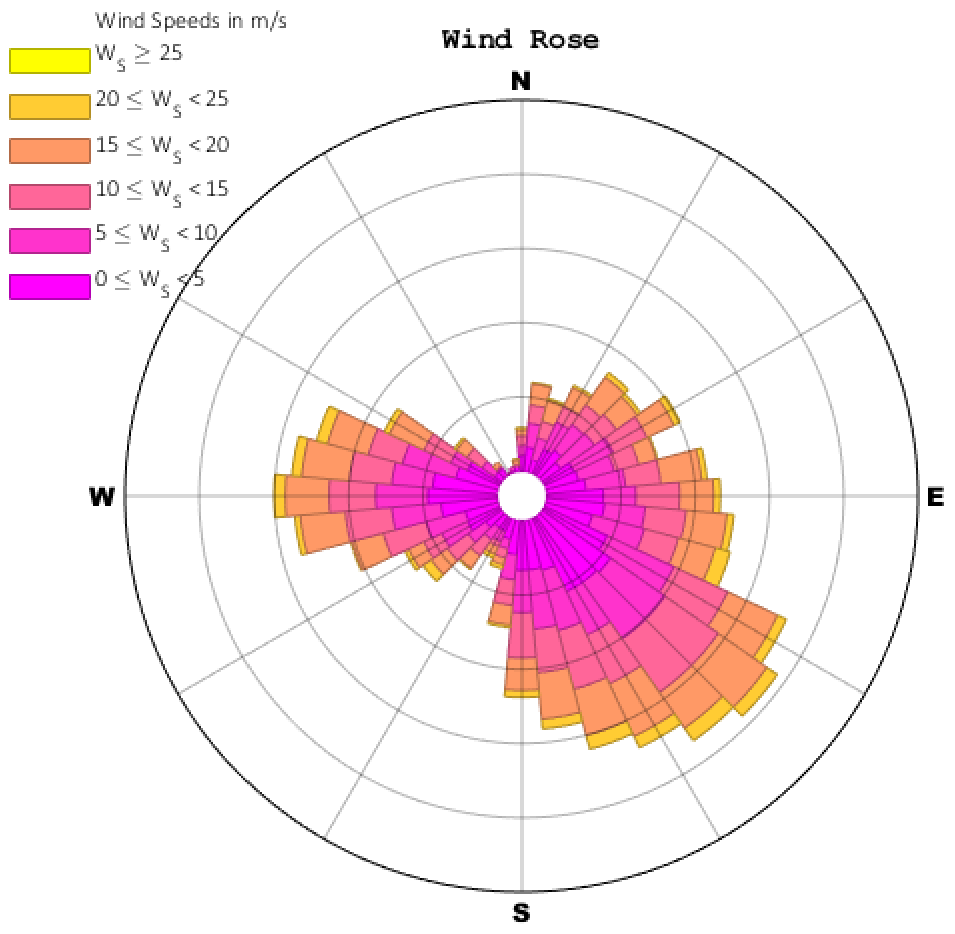


TEXT FONT FACE - 'textfontname', 'titlefontname', 'legendfontname'

Change the font face for labels, title and legend.

It is also possible to change **text fonts**, separately for labels (`'textfontname'`), title (`'titlefontname'`) and legend (`'legendfontname'`). The default font face for every text is Helvetica.

```
Options4.textfontname = 'Arial Black';
Options4.titlefontname = 'Courier New';
Options4.legendfontname = 'Calibri Light';
Options4.cmap = 'spring';
[figure_handle, count3, speeds3, directions3, Table3] = WindRose(dir, spd, Options4);
```

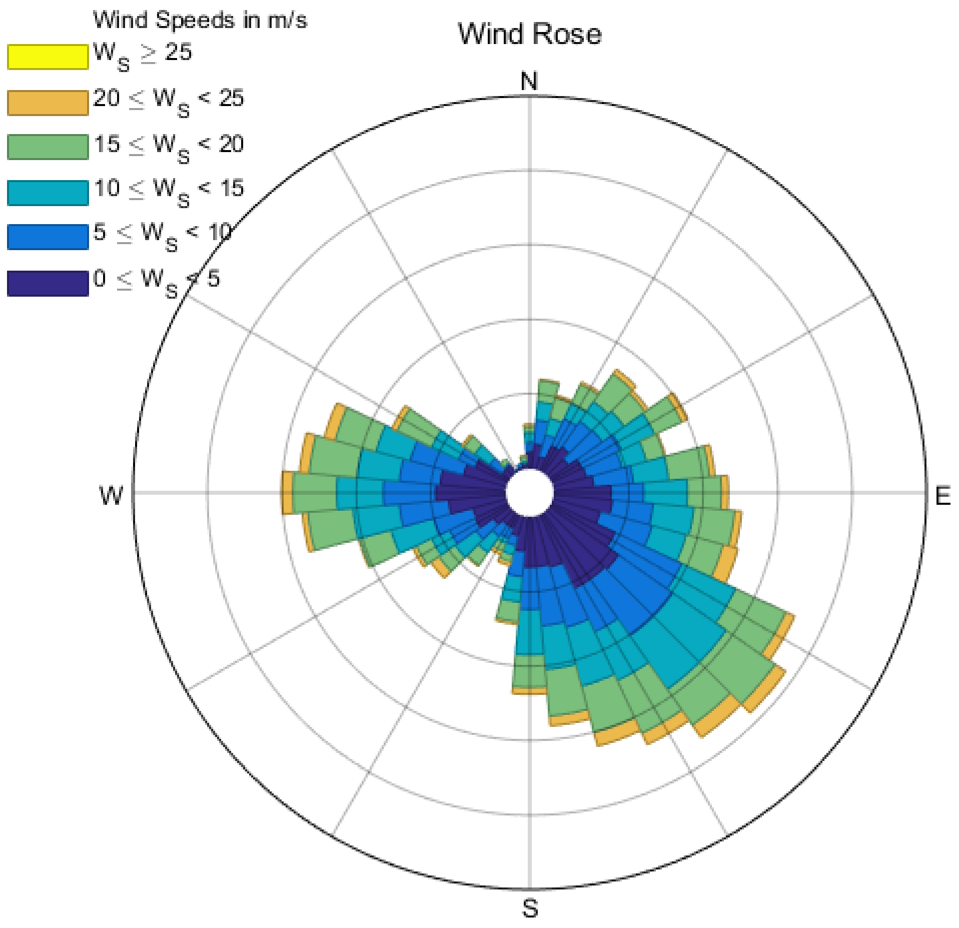


TITLE FONT WEIGHT

Change the title font weight.

If you want the title to appear with a different weight (other than bold) you can choose between `'normal'`, `'light'`, `'demi'` and `'bold'`:

```
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd,
'TitleFontWeight', 'demi');
```



PLOT WINDROSES IN CURRENT AXIS GIVEN X,Y COORDINATES - `'x','y'`

Plot a wind rose at any place, given X & Y coordinates in current axis.

If you want to plot a wind rose at any position given the center coordinates 'X', 'Y', you might want to combine this with the 'scalefactor' parameter.

Wind Roses have a radius of 1, so if you set the 'scalefactor' to be 6, the new outer radius will be 6 (Wind Rose ranging from -6 to 6).

The following example plots three wind roses at positions (5,6), (-1,-2) and (4,1) respectively. The sizes (Scale factor) will be 1, 2 and 0.75 respectively. The example displays only 1 frequency value (the highest one) and removes the labels, but we want to preserve the 12 radial grids:

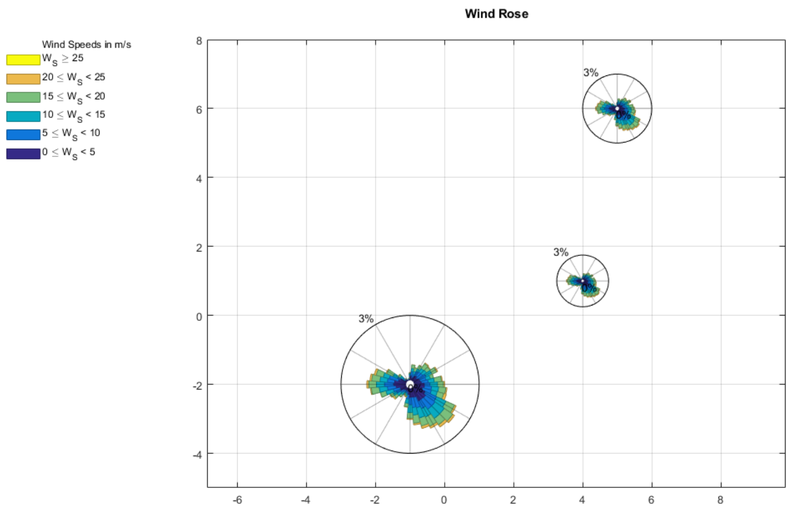
```
figure;
Options5.freqlabelangle = 'auto';
Options5.nfreq = 1;
Options5.labels = '';
Options5.radialgridnumber = 12;

Options5.X = 5;
Options5.Y = 6;
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options5);

Options5.X = -1;
Options5.Y = -2;
Options5.scalefactor = 2;
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options5);

Options5.X = 4;
Options5.Y = 1;
Options5.scalefactor = 0.75;
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options5);

grid on; box on; set(gca,'color',get(gcf,'color'));
ylim([-5 8]);
drawnow; set(gcf,'windowstate','maximized');
```



See the three wind roses plotted at the specified positions and with the specified scale factor. Note that the wind rose with scale factor of 1 is 2 units width and height (2 radii with a length of 1 each), while the wind rose with scale factor of 2 is 4 units width and height (2 radii, with a length of 2 each).

PLOT EXTRA STUFF ON A EXISTING WINDROSE

Plot anything else over an existing wind rose.

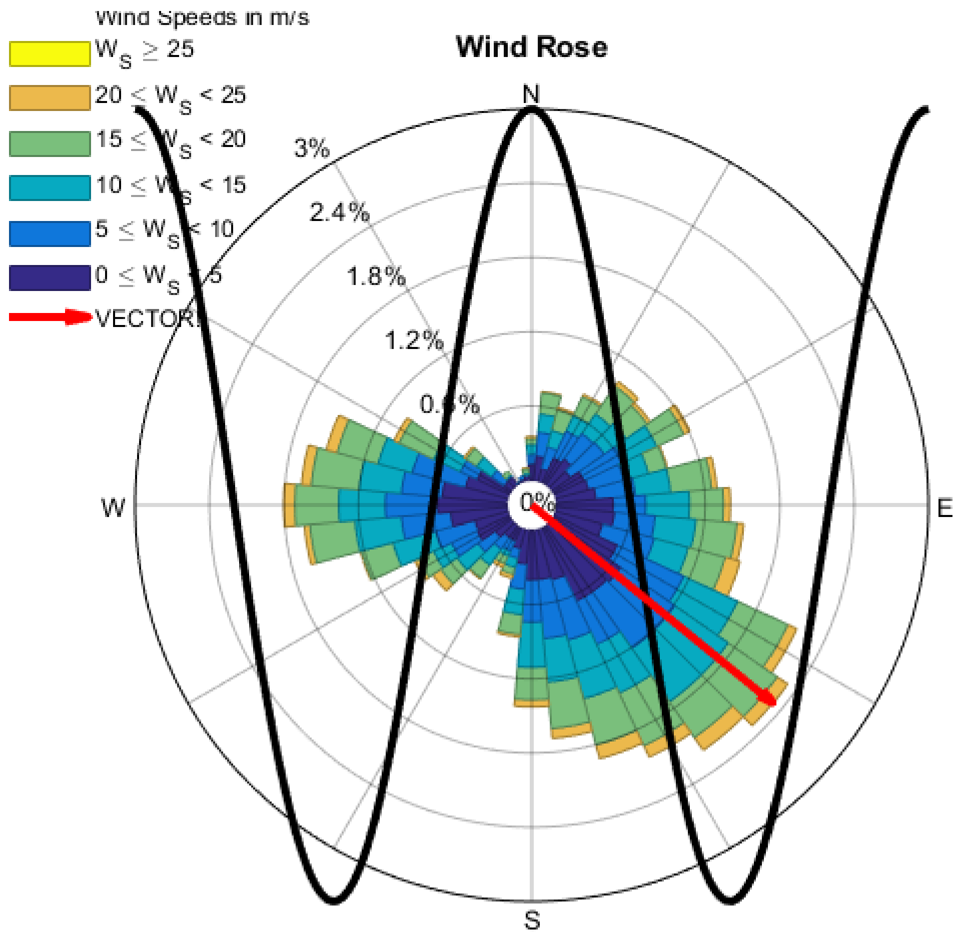
Once we have plotted our wind rose, we might need plotting extra things over it. It is important to note that the wind rose ALWAYS present a radius of 1 unit multiplied by the scale factor (which is 1, by default). Then, whatever we need to plot, must consider this wind rose size.

It is not necessary to call `'hold on'`, because it is already called inside the wind rose.

```
Options6.freqlabelangle = 'auto';
[figure_handle, count, speeds, directions, Table] = WindRose(dir, spd, Options6);

x = linspace(0,1,150)*2 - 1;
y = cos((x+1)*2*pi);
plot(x,y,'k','linewidth',3,'HandleVisibility','off'); % Do not display in the legend
('HandleVisivility','off')

quiver(0,0,0.6084,-0.4987,0,'filled','color','r','linewidth',3,'displayname','VECTOR!');
% Display a custom name in the legend ('displayname','my custom name')
```



OUTPUTS

This function has several outputs.

In order to specify dimensions of each array, the following shorthands have been used:

- **s**: Number of different speed bins shown in the windrose
- **d**: Number of different direction bins shown in the windrose

1. **figure_handle**: *DOUBLE*. Dimensions **1×1**. The first one is the figure handle, which can be used to modify its position, printing, image saving, etc...
2. **count**: *DOUBLE*. Dimensions **d×s**. count is a two dimensional array, where the frequency for each intensity (in columns) and direction (in rows) is represented.
3. **speeds**: *DOUBLE*. Dimensions **1×s**. speeds returns a 1-D array with the speeds appearing in the legends. The number of elements in this array matches the number of columns in count
4. **directions**: *DOUBLE*. Dimensions **d×1**. directions returns a 1-D array with the mean value of the directions of each "branch" in the wind rose. The number of elements in this array matches the number of rows in count
5. **Table**: *CELL*. Dimensions **4+d × 3+s**. Table is a summary table where the frequencies for each direction and each speed are shown, in an excel ready format, so this can be used for xlswrite or just prompting in Matlab's command window.

ABOUT THE AUTHOR

By **Daniel Pereira Valadés**.

Updated: 2021-04-20

daniel.pereira.valades@gmail.com

dpereira.asempyme.com

In case you need help or updates, please contact me by mail after making sure that the documentation does not present the functionality you need. I can answer in English or Spanish.

If you like my work, consider donating. With PayPal you can donate in a secure manner with your credit or debit card, with no need to have a PayPal account

Donate with PayPal

https://www.paypal.com/donate?hosted_button_id=D79DM35H7NPCW