University Oldenburg

Wind Physics Measurement Project

Exercise 3 - Long Term Assessment and MCP Method

Author:
Jan Kämper
Florian Börgel

Supervisor:
Andreas Schmidt

June 7, 2016

Contents

Introduction

1 Synchronizing the data

In task 1 we were asked to synchronize the data of FINO 2 and MERRA-2. The data of FINO 2 was provided in 10 minute-averages over five years starting at January 2010. The MERRA-2 data is the so called long term data provided in one hour-averages, ranging from 1992-2016. Before we started with the actual task we had to neglect all measured data of FINO 2 after end of May 2014 and calculate one hour averages at height 90 m. We used the following routine:

```
last_timestamp = find(Fino2.time==datenum('31-May-2014 23:55:00'));
for i = 1:last_timestamp/6
    fino2_1h_v92(i,1) = datenum('01-Jan-2010 00:00:00')+(i-1)*1/24;
    range_array_v = fino2_v92((i-1)*6+1:i*6,1);
    range_array_dir = fino2_d91((i-1)*6+1:i*6,1);
    fino2_1h_v92(i,3) = nanmean(range_array_v);
    fino2_1h_v92(i,2) = nanmean(range_array_dir);
end
```

This routine calculates the last time stamp of the data and returns the corresponding line in the cell. The for loop creates a continuous time axis with one hour intervals In addition it calculates the mean of six 10-minutes intervals for wind speeds and wind directions.
Since we can only synchronize corresponding time intervals of FINO 2 and MERRA-2 we also had to calculate the correct time stamps of MERRA-2.

```
timestamps = raw_data.Var1(:,1); % MERRA-2 Data
first_timestamp = find(strcmp(timestamps(:), '01.01.2010 00:00'));
last_timestamp = find(strcmp(timestamps(:), '31.05.2014 23:00'));
```

After finding the correct time stamps we finally we able to save the data in one variable, called $connected_data$.
Synchronizing the data means that we only take time stamps into account when both measurement systems have entries. Incorrect data of FINO 2 is marked with NaN values. The data of MERRA-2 has additional status columns. 0 as an entry means correct measurement.

```
count = 0
for i = 1:length(connected_data(:,1))
    if raw_data.Var8(i) ~= 0 || raw_data.Var9(i) ~= 0 || isnan(connected_data(i,2))
    == 1 || isnan(connected_data(i,3)) == 1
        connected_data(i,2:5) = NaN;
        count = count +1;
```

```
    end
end
```

2 Sorting into 12 wind sectors

In order to sort the synchronized data in 12 sectors we decided to save all sectors in one variable. The use of only one variable instead of initializing 12 different variables helped to achieve a more structured code. Here is an abstract of our source code for Task 2.

```
for i = 1:length(connected_data(:,1))
    if ~isnan(connected_data(i,3))
        sortIndex = floor(connected_data(i,3)/360*12)+1;
        sortedCell{sortIndex*3-2} = [sortedCell{sortIndex*3-2}, connected_data(i,1)
    ]; %timestamp
        sortedCell{sortIndex*3-1} = [sortedCell{sortIndex*3-1}, connected_data(i,4)
    ]; %merra 2
        sortedCell{sortIndex*3} = [sortedCell{sortIndex*3}, connected_data(i,2)]; %
    fino 2
    end;
end;
```

The variable *sortedCell* is separated into 12 sectors. Each sector consists of 3 columns: corresponding time stamp, windspeeds of FINO 2 and windspeeds of MERRA-2.

3 Monthly averages of sectors

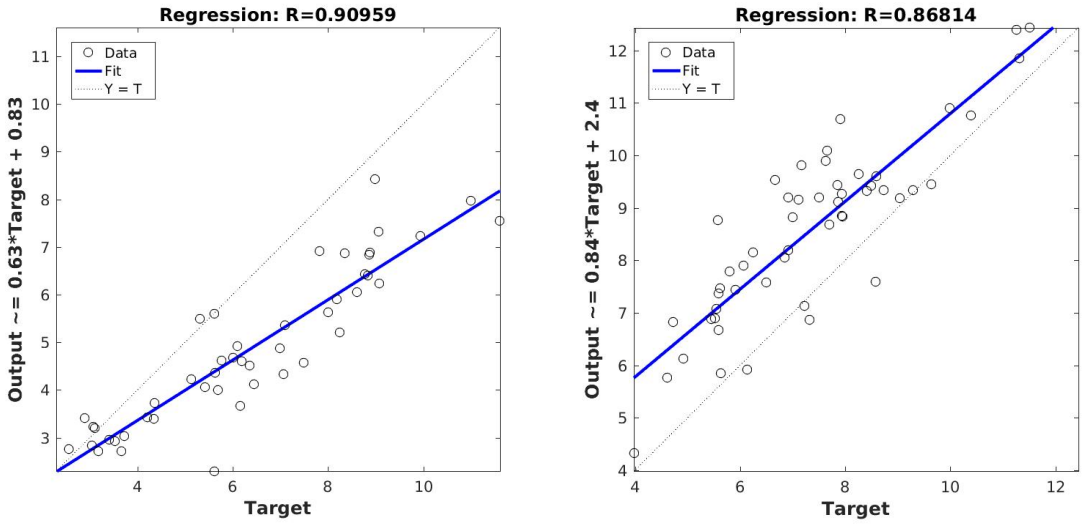
In Task 3 we calculated the monthly averages for every sector and for all data.

4 Regressions parameters and plots

In Task 4 & 5 we were asked to create regressions plots of the monthly averages of each sector. In addition we also had to calculate the corresponding regression parameters. We used the provided plot routine:

```
plotregression(avgSectorPerMonth(:,i*2), avgSectorPerMonth(:,i*2+1), 'Regression');
    [regressionParameters(i,1), regressionParameters(i,2), regressionParameters(i
    ,3)]...
     = regression(avgSectorPerMonth(:,i*2), avgSectorPerMonth(:,i*2+1), 'one');
```

Figure 1 shows the resulting plot for sector 0 - 30 deg and sector 120 - 150 deg.



(a) Sector 0 - 30 deg

(b) Sector 120 - 150 deg

Figure 1: Regressions Plots