Programming for Data Analytics

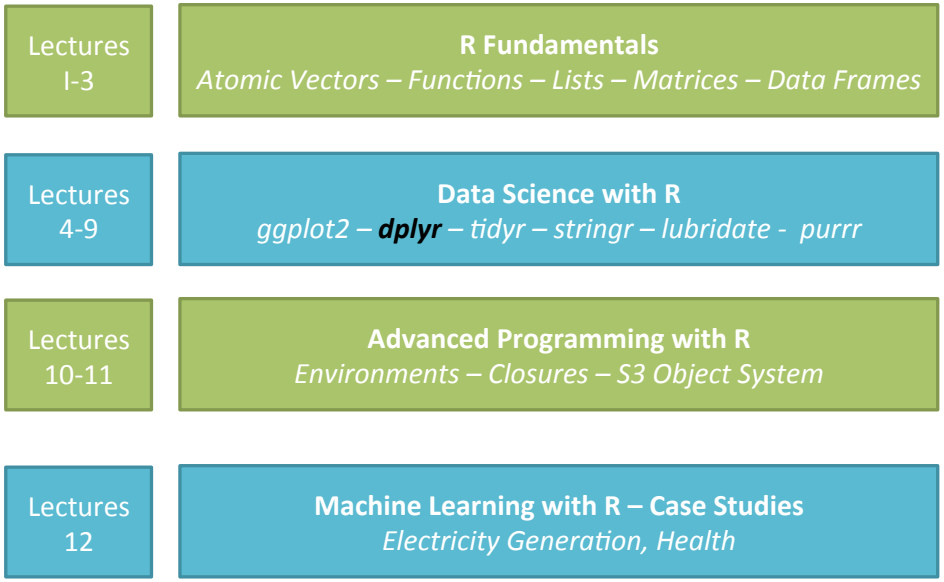
Lecture 6: Relational Data with dplyr

Dr. Jim Duggan,
School of Engineering & Informatics
National University of Ireland Galway.

https://twitter.com/_jimduggan

Lecture Overview

- Relational data in dplyr
- Joins
  - Mutating
  - Filtering
- Energy Case Study



Relational Data with dplyr

- Typically, data analysis involves many tables of data that must be combined to answer questions
- Collectively, multiple tables of data are called *relational data*
- Relations are always defined between a pair of tables

key	val_x
1	x1
2	x2
3	x3

key	val_y
1	y1
2	y2
4	y3

Keys

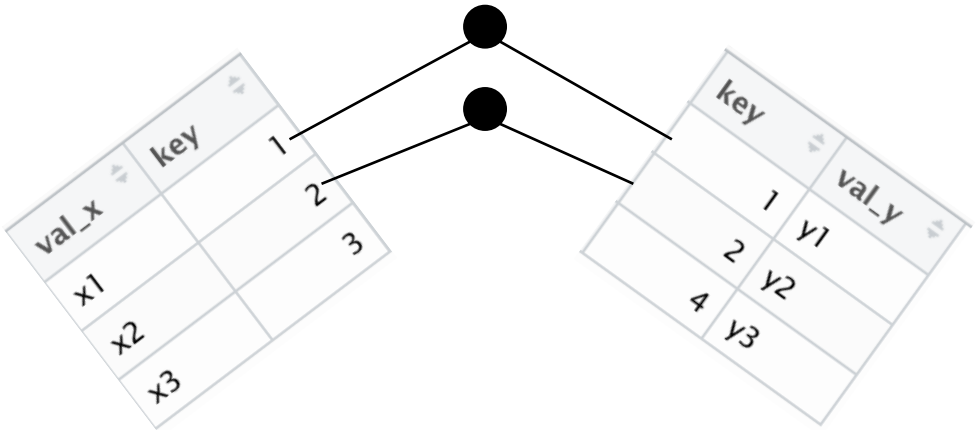
- The variables used to connect each pair of tables are called keys
- A key is a variable (or set of variables) that uniquely identifies an observation
- There are two types of keys:
  - A primary key uniquely identifies an observation in its own table
  - A foreign key uniquely identifies an observation in another table.

Mutating Joins

- Allows you to combine variables from two tables
- First matches observations by their keys, and then copies across variables from one table to another
- Similar to mutate(), the join functions add variables to the right

Join Types

- Inner Join:
  - matches pairs of observations when their keys are equal
  - Unmatched rows are not included in the result



key	val_x	val_y
1	x1	y1
2	x2	y2

```
inner_join(x,y,by="key")
```

Outer Joins

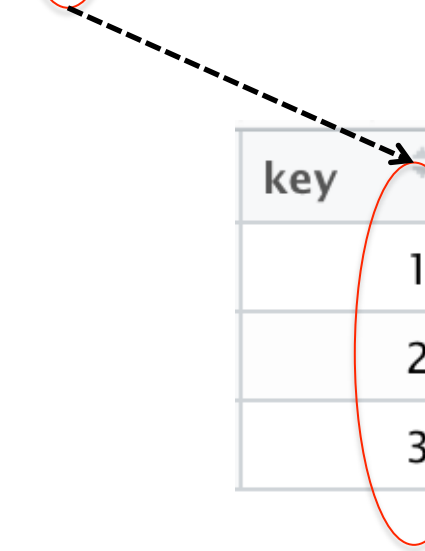
- An outer join keeps observations that appear in at least one of the tables. There are three types of outer joins (x,y)
  - A *left join* keeps all observations in x
  - A *right join* keeps all observations in y
  - A *full join* keeps all observations in x and y

Left Join

```
left_join(x,y,by="key")
```

key	val_x
1	x1
2	x2
3	x3

key	val_y
1	y1
2	y2
4	y3



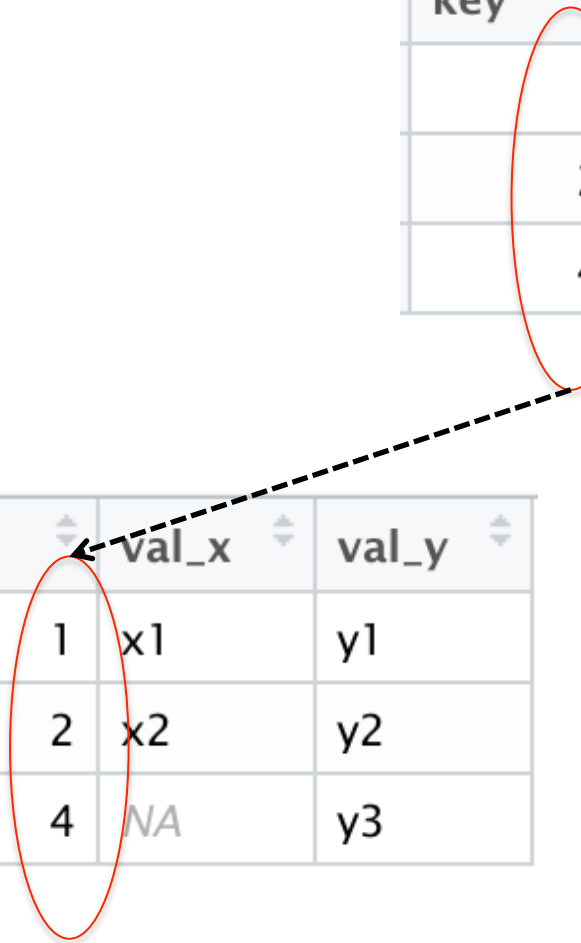
key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	*NA*

Right Join

```
right_join(x,y,by="key")
```

key	val_x
1	x1
2	x2
3	x3

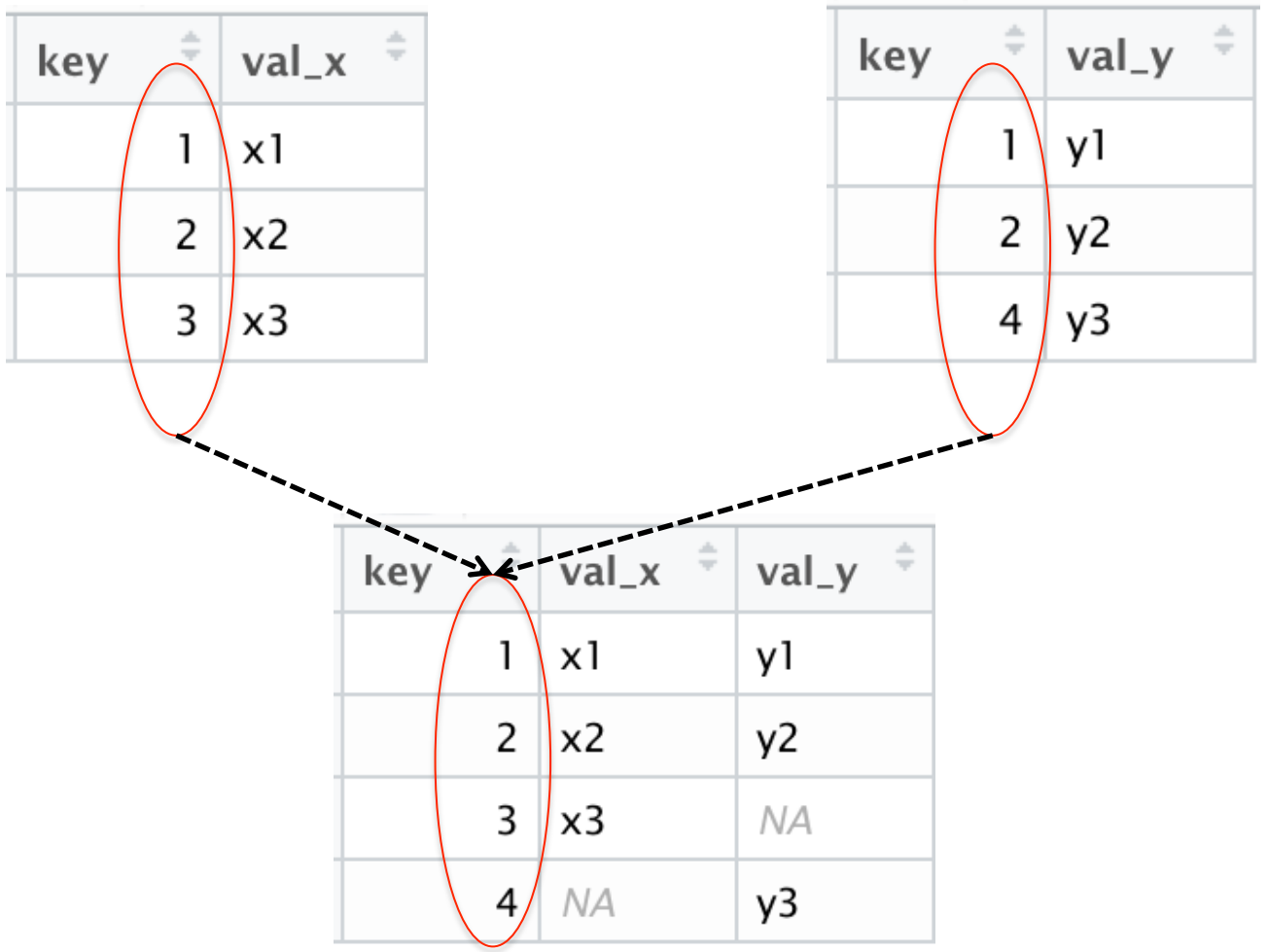
key	val_y
1	y1
2	y2
4	y3



key	val_x	val_y
1	x1	y1
2	x2	y2
4	*NA*	y3

Full Join

```
full_join(x,y,by="key")
```



key	val_x
1	x1
2	x2
3	x3

key	val_y
1	y1
2	y2
4	y3

key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	*NA*
4	*NA*	y3

Filtering Joins

- Match observations in the same way as mutating joins, but affect the observations, not the variables
- Two types:
  - semi_join(x,y) keeps all observations in x that have a match in y
  - anti_join(x,y), drops all observations in x that have a match in y.

Semi Join

```
semi_join(x,y,by="key")
```

key	val_x
1	x1
2	x2
3	x3

key	val_y
1	y1
2	y2
4	y3

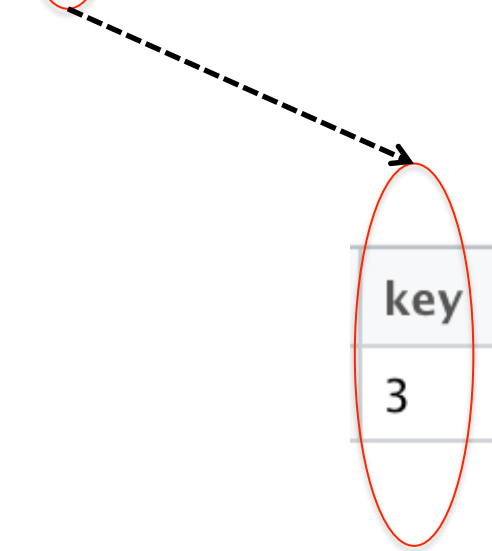
key	val_x
1	x1
2	x2

Anti Join

```
anti_join(x,y,by="key")
```

key	val_x
1	x1
2	x2
3	x3

key	val_y
1	y1
2	y2
4	y3



key	val_x
3	x3

Set Operations

- All operations work with a complete row, comparing the values of every variable
- These expect the x and y inputs to have the same variables, and treat the observations like sets
  - intersect(x,y) returns only observations in both x and y
  - union(x,y) returns unique observations in x and y
  - setdiff(x,y) returns observations in x, but not in y

union(df1,df2)

df1

x	y
1	1
2	1

df2

x	y
1	1
1	2

x	y
1	2
2	1
1	1

setdiff(df1,df2)

df1

x	y
1	1
2	1

df2

x	y
1	1
1	2

x	y
2	1

intersect(df1,df2)

df1

x	y
1	1
2	1

df2

x	y
1	1
1	2

x	y
1	1

Simple Example

name	instrument
John	guitar
Paul	bass
George	guitar
Ringo	drums
Stuart	bass
Pete	drums

name	band
John	T
Paul	T
George	T
Ringo	T
Brian	F

```
x <- data.frame(
  name = c("John", "Paul", "George", "Ringo", "Stuart", "Pete"),
  instrument = c("guitar","bass","guitar","drums","bass","drums"),
  stringsAsFactors = F
)
```

```
y <- data.frame(
  name = c("John", "Paul", "George", "Ringo", "Brian"),
  band = c(T,T,T,T,F),
  stringsAsFactors = F
)
```

Type	Action
inner	Include only rows in **both** x and y

name	instrument
John	guitar
Paul	bass
George	guitar
Ringo	drums
Stuart	bass
Pete	drums

name	band
John	T
Paul	T
George	T
Ringo	T
Brian	F

```
> inner_join(x,y)
Joining, by = "name"
    name instrument band
1   John     guitar TRUE
2   Paul       bass TRUE
3 George     guitar TRUE
4  Ringo      drums TRUE
```

Type	Action
left	Include all of x, and matching rows of y

name	instrument
John	guitar
Paul	bass
George	guitar
Ringo	drums
Stuart	bass
Pete	drums

name	band
John	T
Paul	T
George	T
Ringo	T
Brian	F

```
> left_join(x,y)
Joining, by = "name"
    name instrument band
1   John     guitar TRUE
2   Paul       bass TRUE
3 George     guitar TRUE
4  Ringo      drums TRUE
5 Stuart       bass   NA
6   Pete      drums   NA
```

Type	Action
semi	Include rows of x that match y

name	instrument
John	guitar
Paul	bass
George	guitar
Ringo	drums
Stuart	bass
Pete	drums

name	band
John	T
Paul	T
George	T
Ringo	T
Brian	F

```
>
> semi_join(x,y)
Joining, by = "name"
    name instrument
1   John     guitar
2   Paul       bass
3 George     guitar
4  Ringo      drums
```

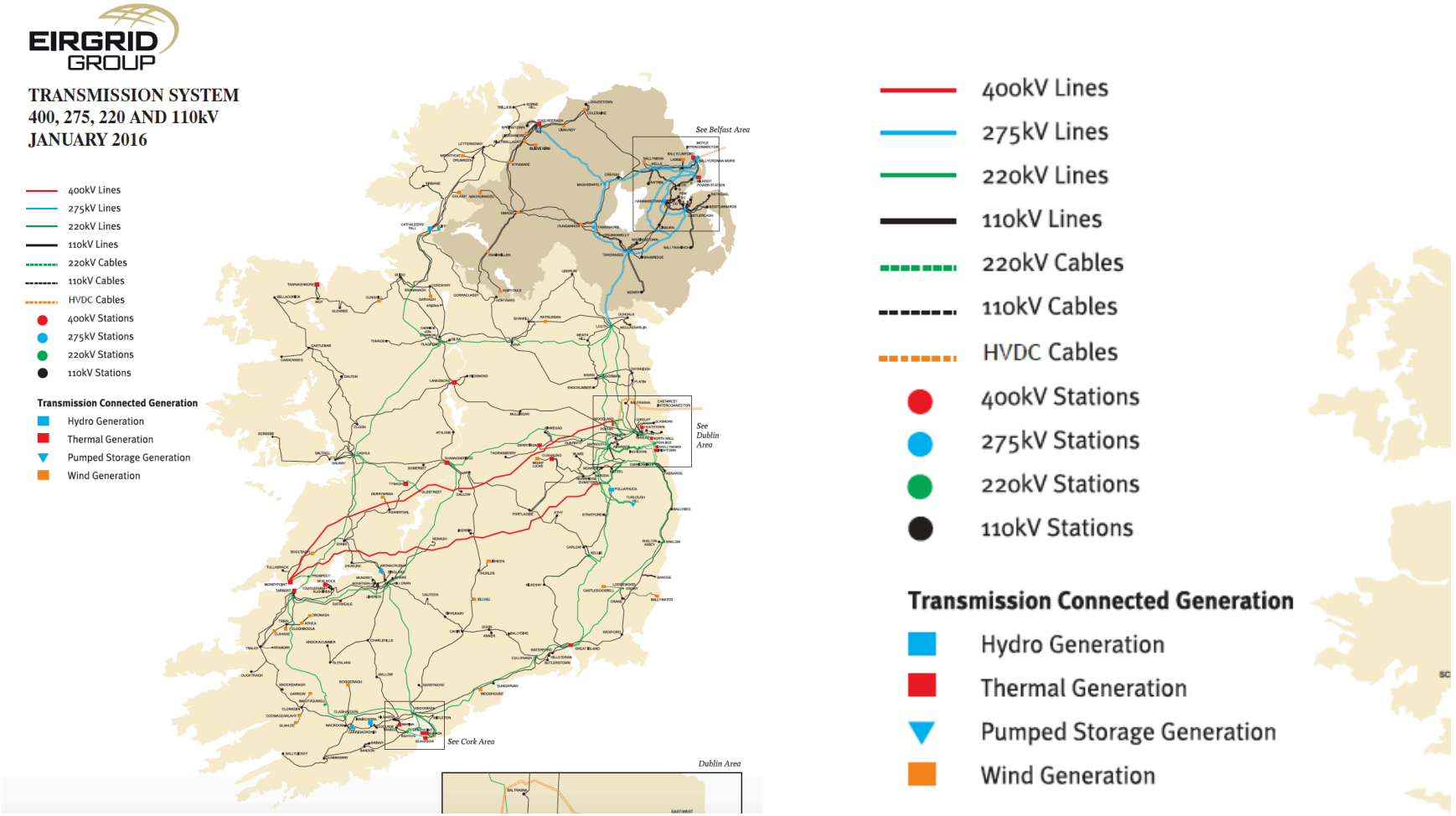
Type	Action
anti	Include rows of x that **don't** match y

name	instrument
John	guitar
Paul	bass
George	guitar
Ringo	drums
Stuart	bass
Pete	drums

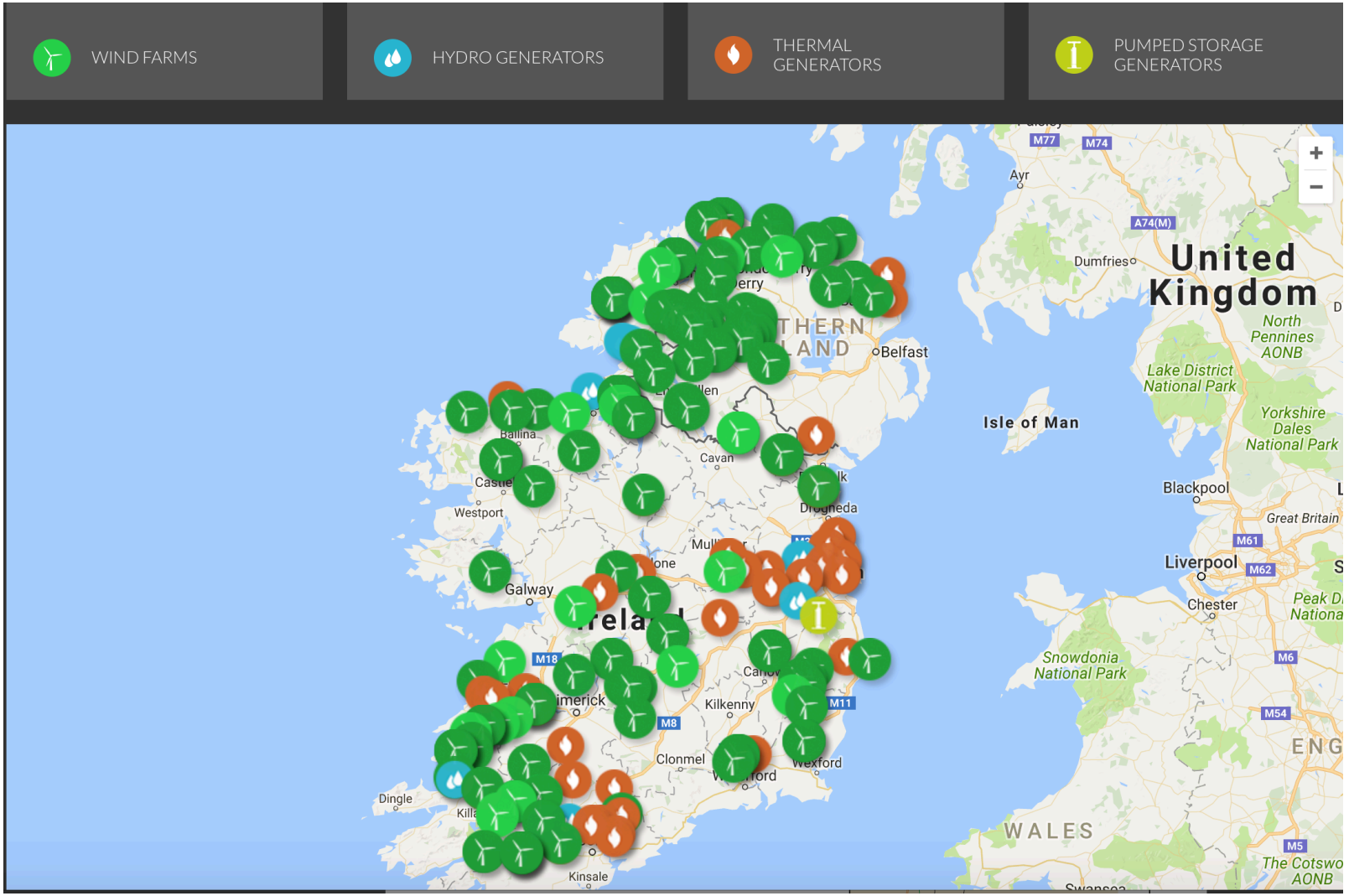
name	band
John	T
Paul	T
George	T
Ringo	T
Brian	F

```
> anti_join(x,y)
Joining, by = "name"
    name instrument
1   Pete      drums
2 Stuart       bass
```

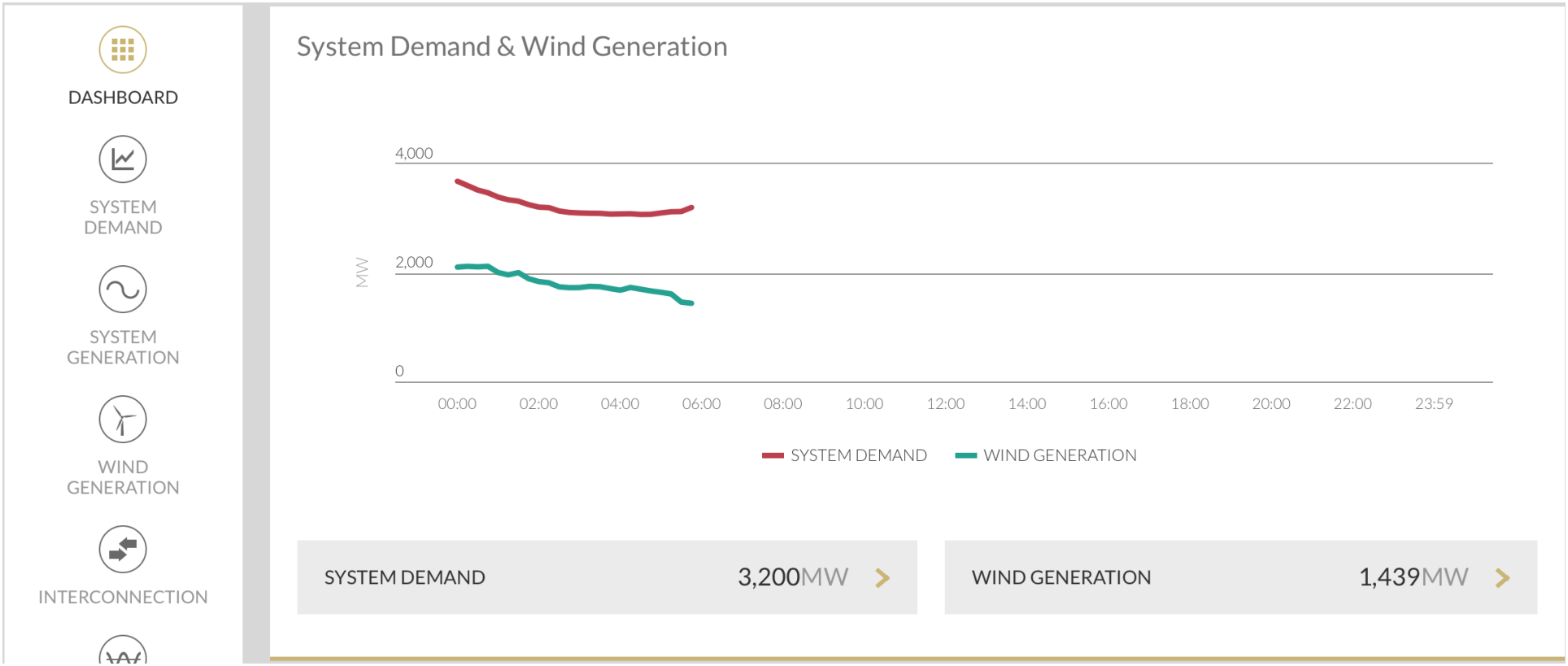
Energy Case Study



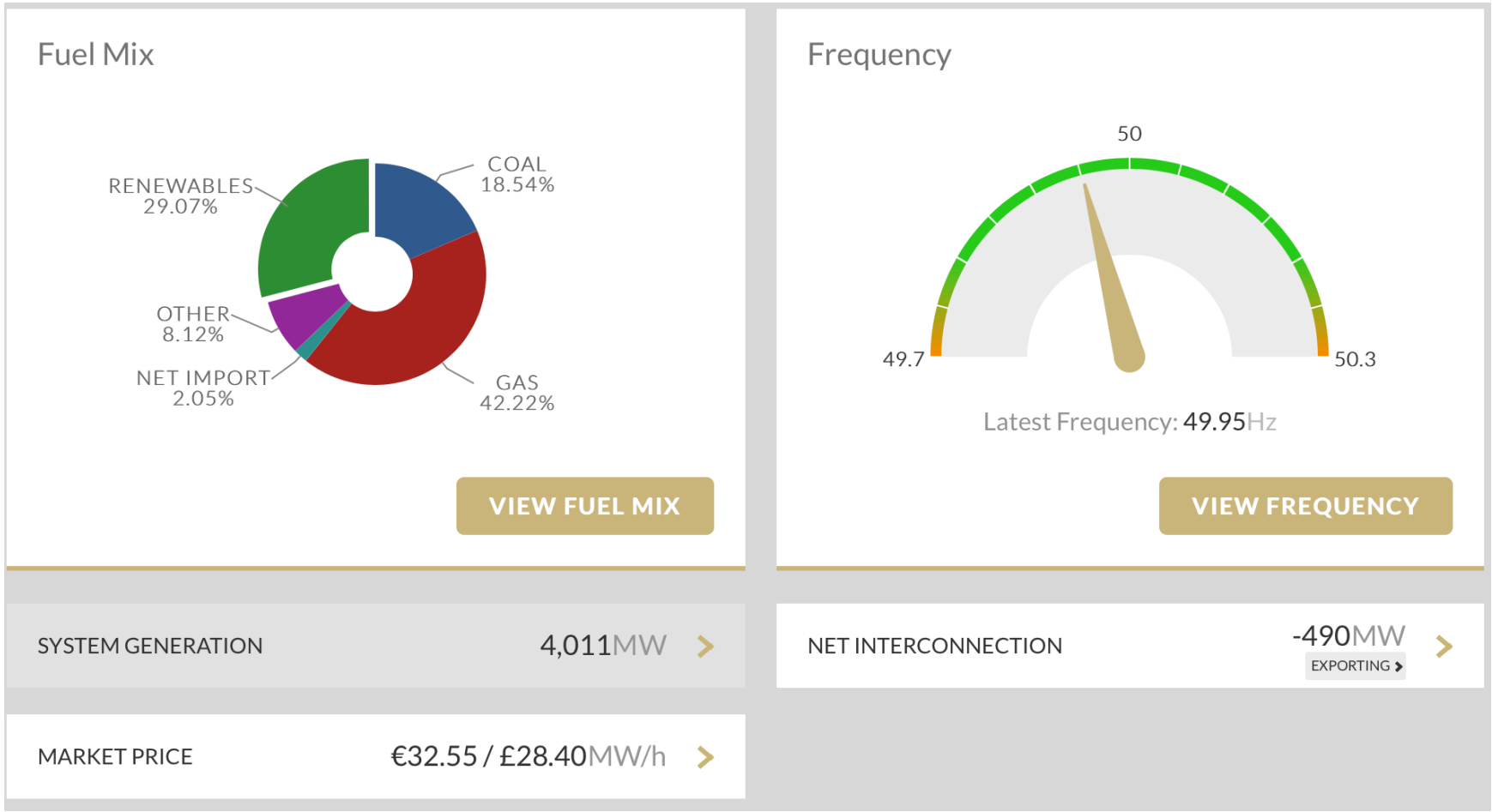
Generation Information



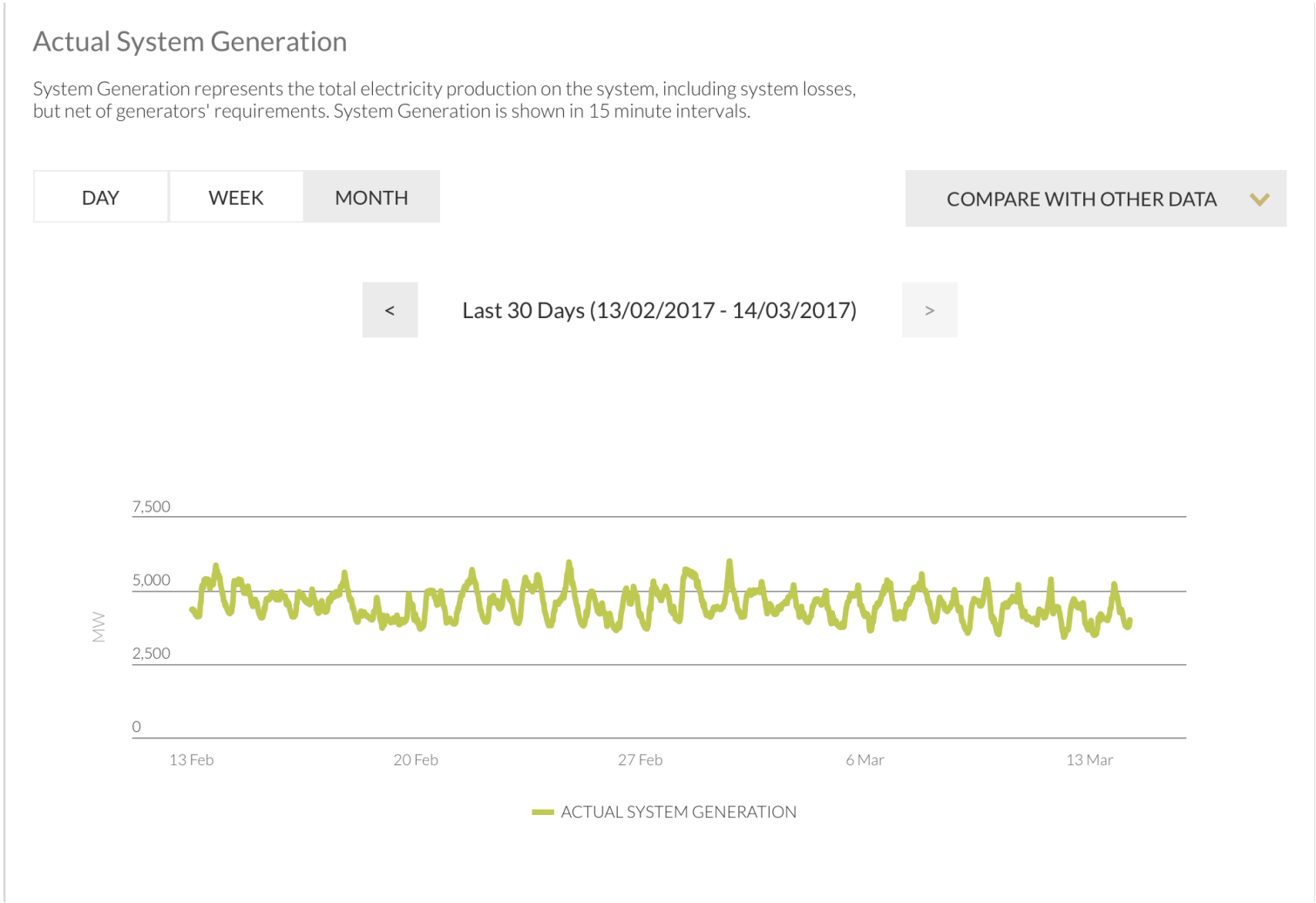
http://smartgriddashboard.eirgrid.com



Fuel Mix, Frequency & Net Flows



Generation over time



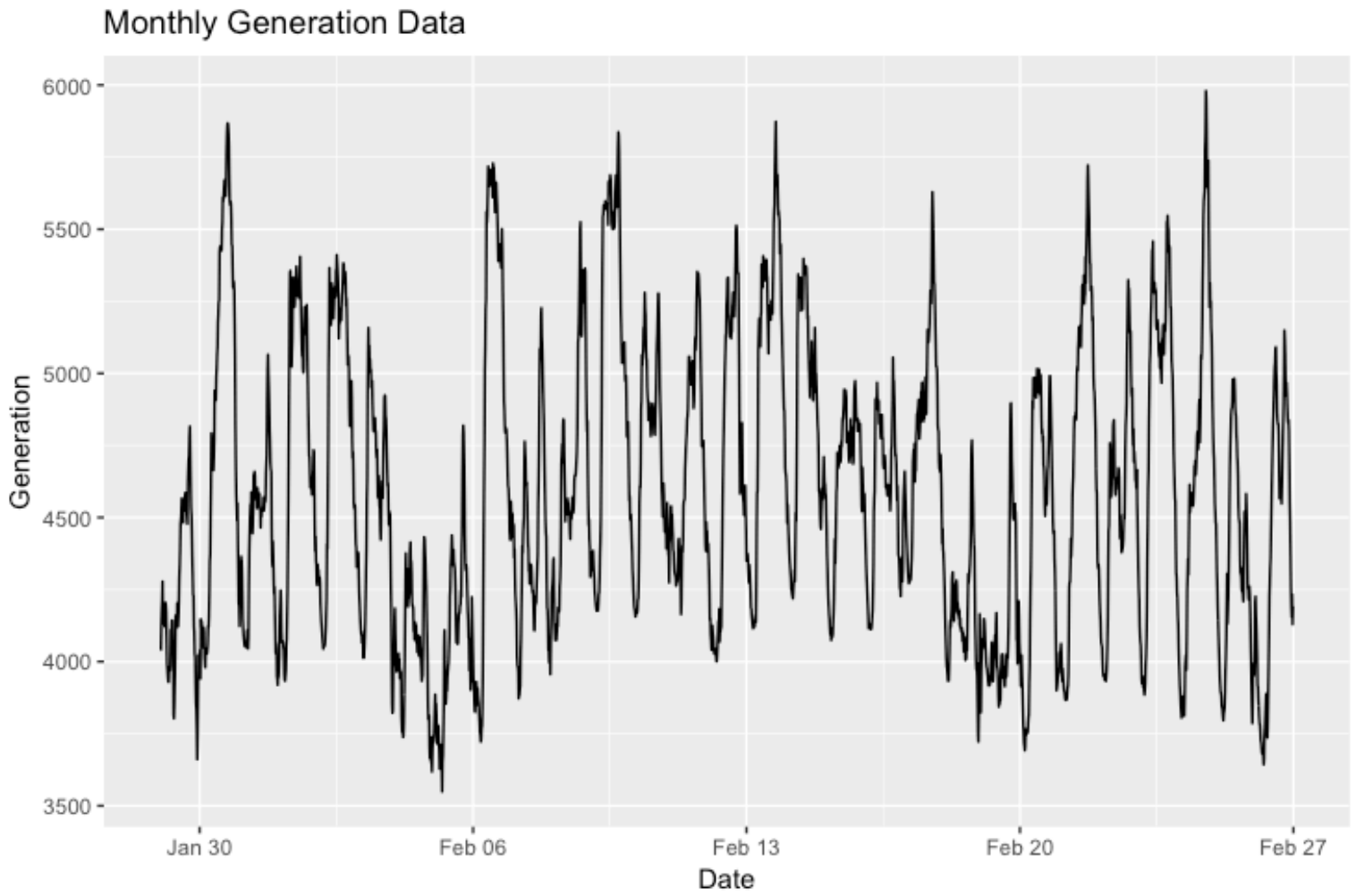
Sample Data

```
> ener
# A tibble: 2,784 x 12
             DateTime     Date HourOfDay MinuteOfDay DayOfWeek Demand Generation  Wind   CO2 NetImports  EWIC
               <dttm>    <chr>     <int>       <int>     <chr>  <int>      <int> <int> <int>      <int> <int>
 1 2017-01-29 00:00:00 29/01/17         0           0       Wed   3834       4041   449   552       -145   -33
 2 2017-01-29 00:15:00 29/01/17         0          15       Wed   3785       4041   505   548       -200  -108
 3 2017-01-29 00:30:00 29/01/17         0          30       Wed   3708       4130   521   544       -294  -183
 4 2017-01-29 00:45:00 29/01/17         0          45       Wed   3634       4181   492   543       -419  -258
 5 2017-01-29 01:00:00 29/01/17         1           0       Wed   3581       4211   538   555       -503  -333
 6 2017-01-29 01:15:00 29/01/17         1          15       Wed   3552       4278   561   531       -598  -379
 7 2017-01-29 01:30:00 29/01/17         1          30       Wed   3491       4133   484   545       -516  -374
 8 2017-01-29 01:45:00 29/01/17         1          45       Wed   3435       4143   474   551       -581  -365
 9 2017-01-29 02:00:00 29/01/17         2           0       Wed   3374       4158   442   550       -653  -373
10 2017-01-29 02:15:00 29/01/17         2          15       Wed   3329       4135   421   550       -676  -377
# ... with 2,774 more rows, and 1 more variables: Moyle <int>
```

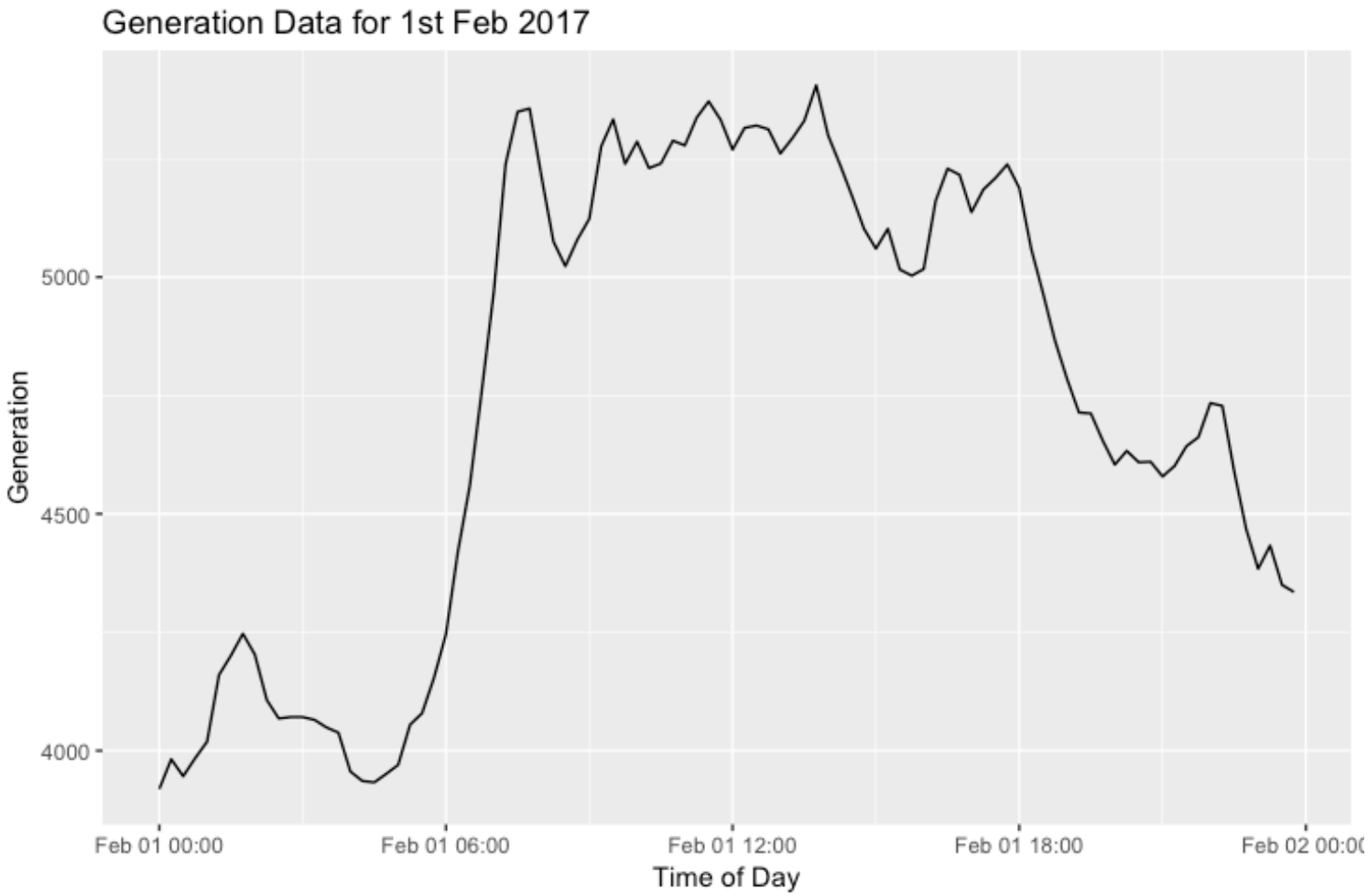
Exploring Data Relationships

- Time series:
  - Generation data over the entire month
  - Generation data over a day
- Time of Day v Net Imports
- Demand v Net Imports
- Wind Generation v CO2 Emissions
- Comparing Wind Generation with Overall Generation

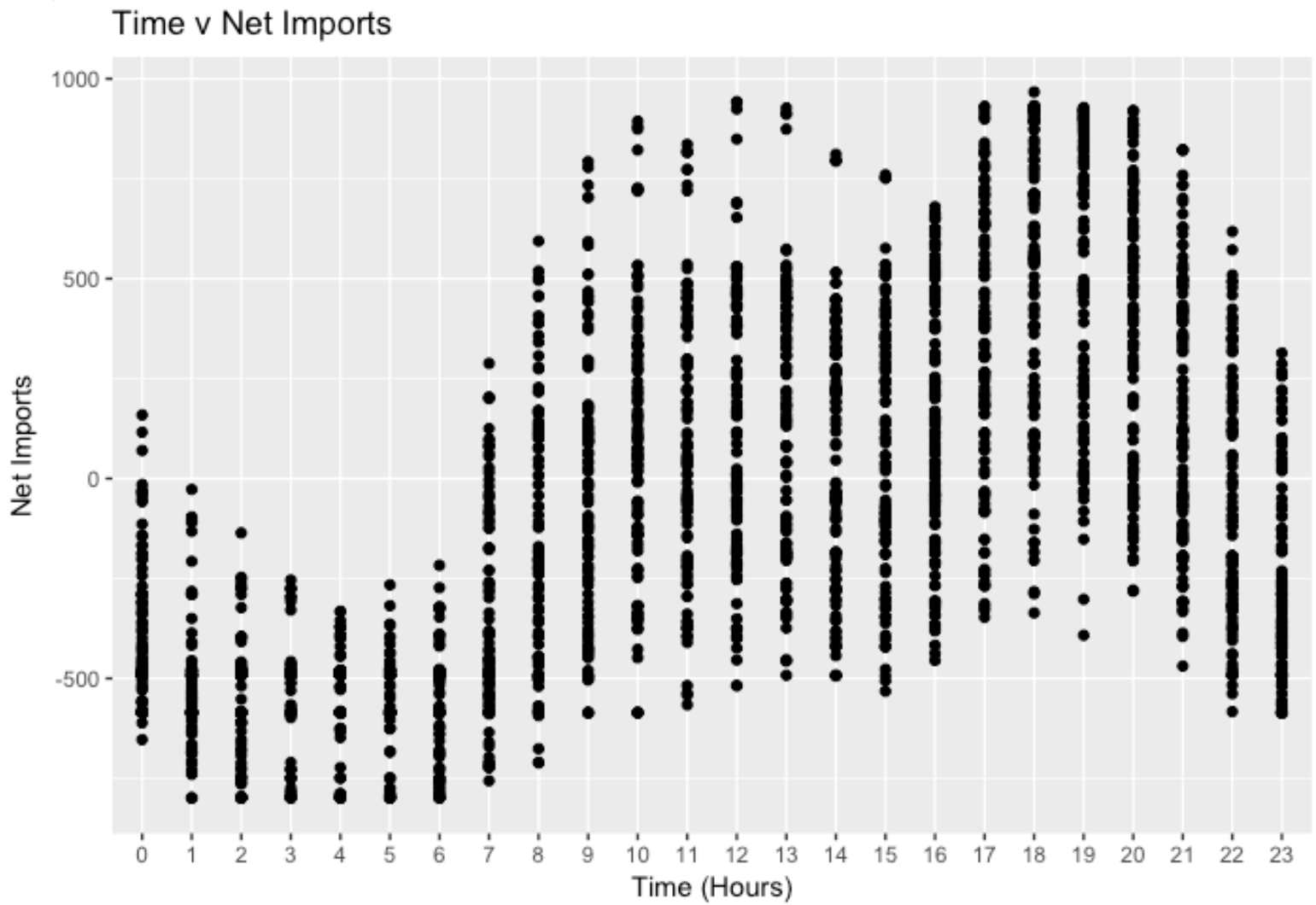
```
ggplot(data = ener) +
  geom_line(mapping = aes(x=DateTime,y=Generation)) +
  xlab("Date") + ylab("Generation") + ggtitle("Monthly Generation Data")
```



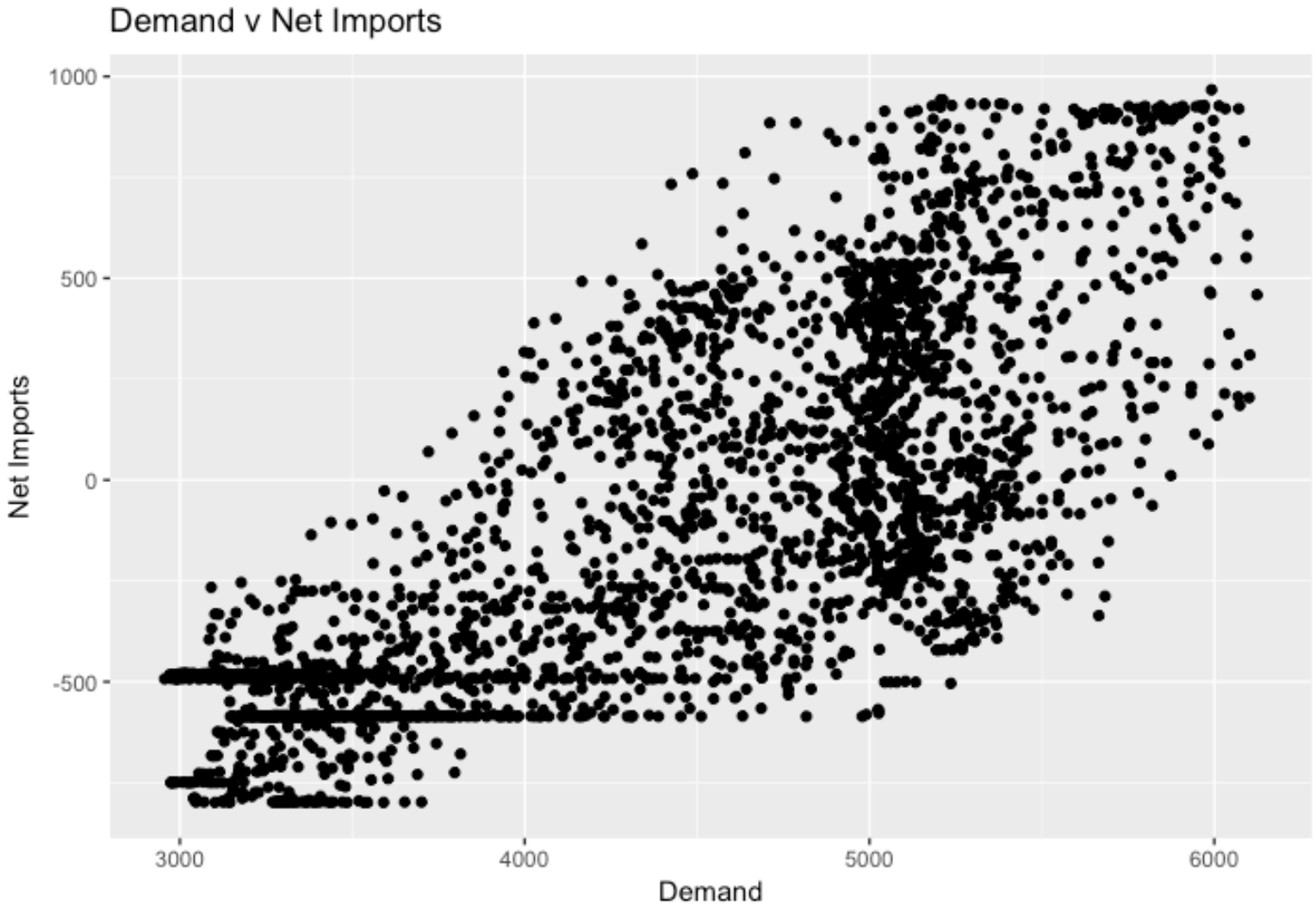
```
ggplot(data = filter(ener,Date=="2017-02-01")) +
  geom_line(mapping = aes(x=DateTime,y=Generation)) +
  xlab("Time of Day") + ylab("Generation")+
  ggtitle("Generation Data for 1st Feb 2017")
```



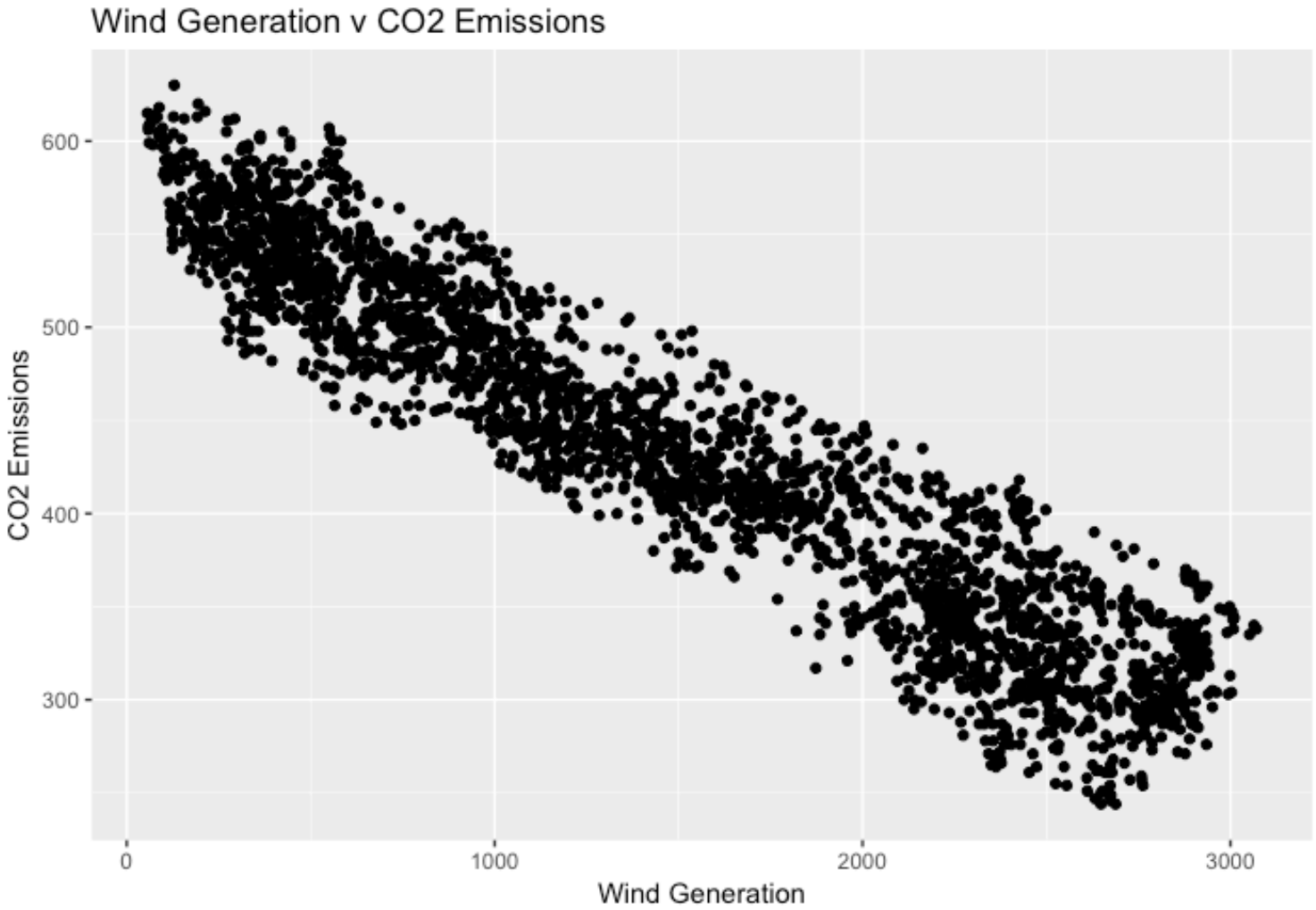
```
ggplot(data = ener) +
  geom_point(mapping = aes(x=as.factor(HourOfDay),y=NetImports)) +
  xlab("Time (Hours)") + ylab("Net Imports") +
  ggtitle("Time v Net Imports")
```



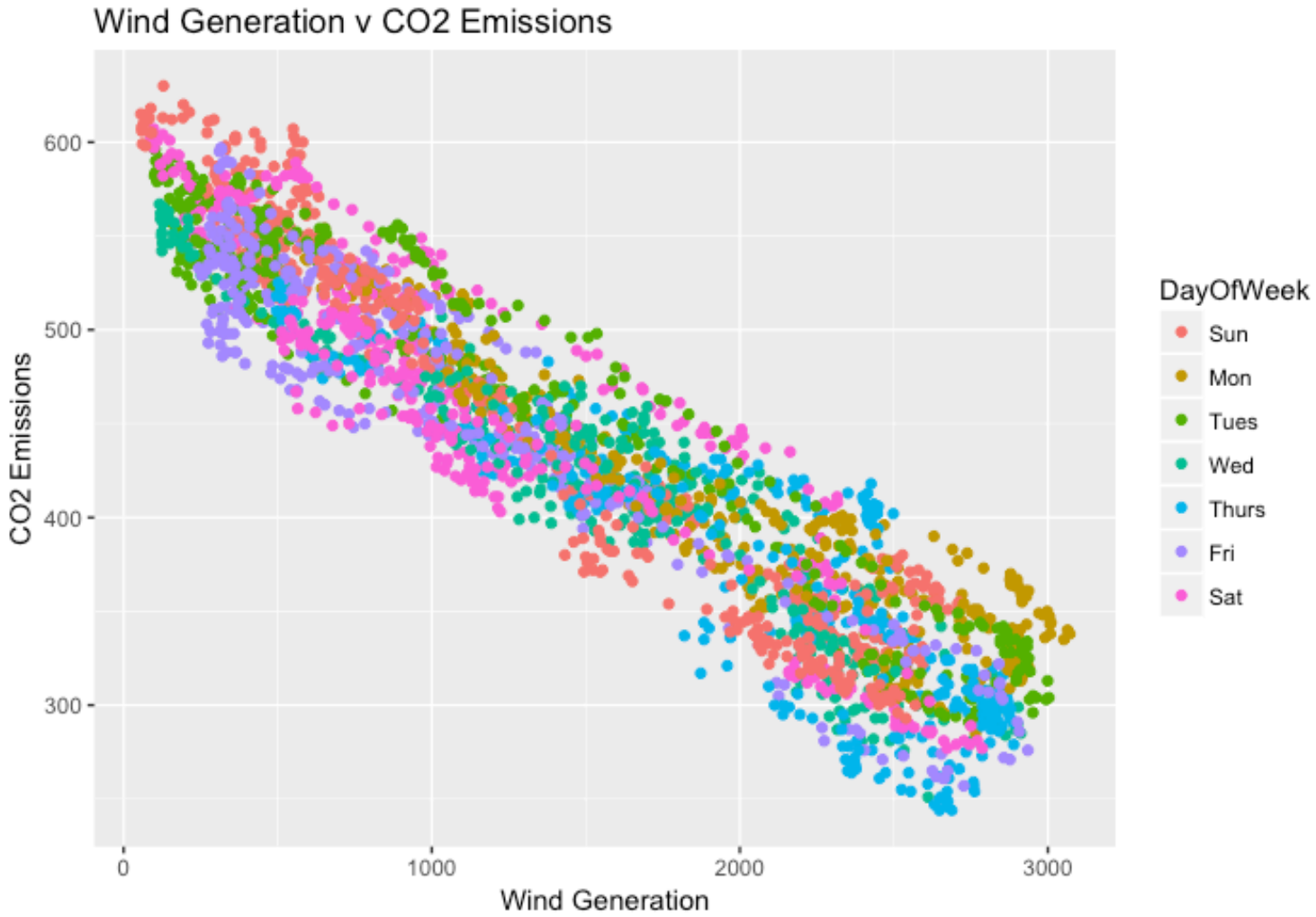
```
ggplot(data = ener) +
  geom_point(mapping = aes(x=Demand,y=NetImports)) +
  xlab("Demand") + ylab("Net Imports") +
  ggtitle("Demand v Net Imports")
```



```
ggplot(data = ener) +
  geom_point(mapping = aes(x=Wind,y=CO2)) +
  xlab("Wind Generation") + ylab("CO2 Emissions") +
  ggtitle("Wind Generation v CO2 Emissions")
```



```
ggplot(data = ener) +
  geom_point(mapping = aes(x=Wind,y=CO2,colour=DayOfWeek)) +
  xlab("Wind Generation") + ylab("CO2 Emissions") +
  ggtitle("Wind Generation v CO2 Emissions")
```



Linking *wind speed* and *wind power generation*

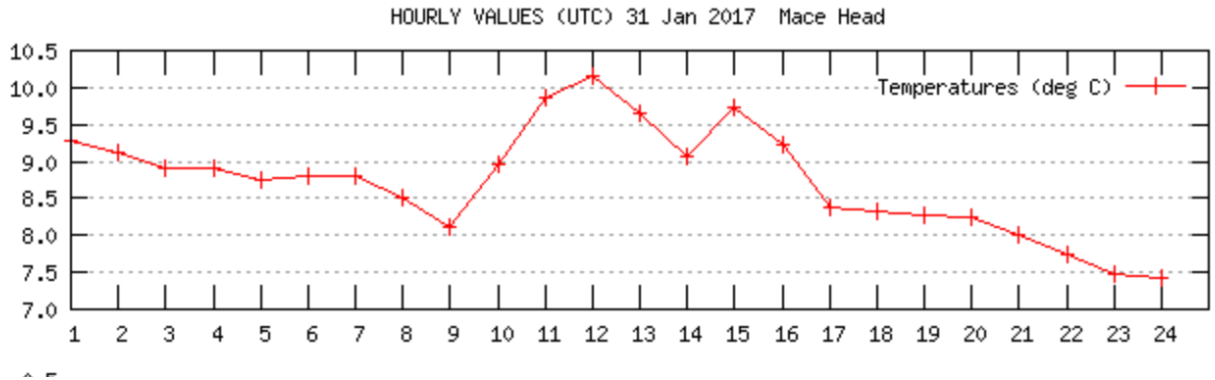
Daily Data

Weather Station Data - From 13/03/2015 to 12/03/2017

Please Select a Station and Date from the menu on the right.

REPORTS FROM MACE HEAD (A)

Date	Rainfall (mm)	Max Temp (°C)	Min Temp (°C)	Grass Min Temp (°C)	Mean Wind Speed (knots)	Maximum Gust (if >= 34 knots)	Sunshine (hours)
31/1/2017	0	10.3	7.3	5.8	10		



Select Station & Date

Station

Mace Head (A)

Date

31/01/2017

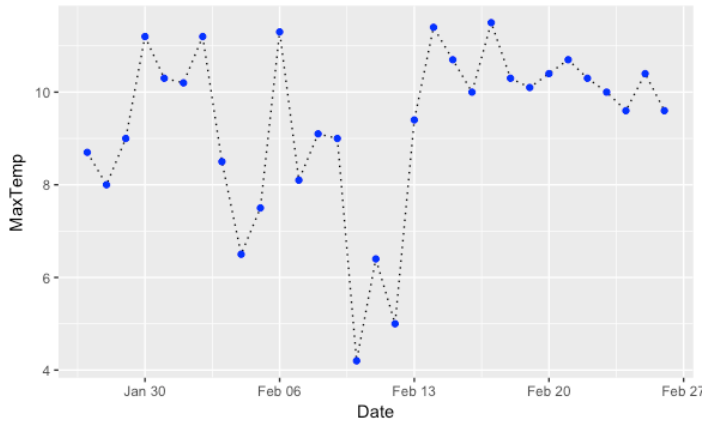
Go ->

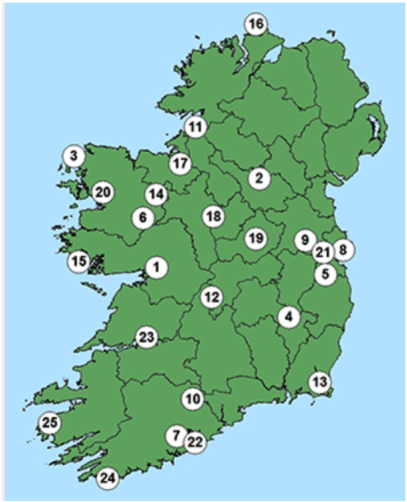
Synoptic Stations

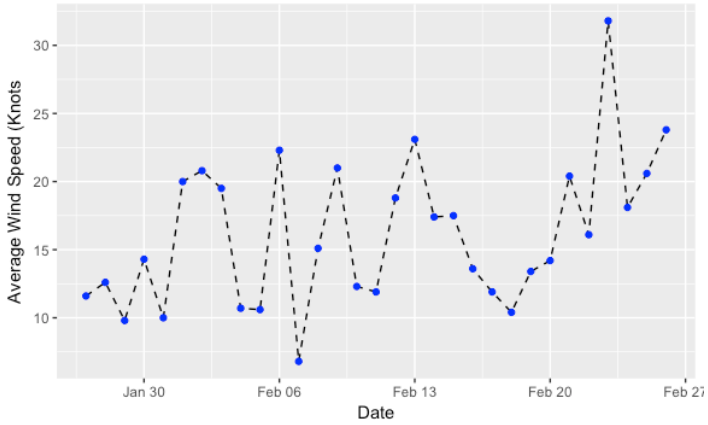
2011

http://www.met.ie

Mace Head Daily Data







Date	Rainfall	MaxTemp	MinTemp	GrassMinTemp	AVRWind	MaxWindGust
27/01/17	7.9	8.7	4.3	-0.7	11.6	
28/01/17	3.5	8	4.5	2.9	12.6	
29/01/17	4.7	9	4.9	3.7	9.8	
30/01/17	7.8	11.2	7.1	5.8	14.3	
31/01/17	0	10.3	7.3	5.8	10	
01/02/17	0.6	10.2	6.1	5.2	20	38
02/02/17	4.9	11.2	7.4	6.4	20.8	45
03/02/17	2.2	8.5	3.6	2.1	19.5	46
04/02/17	5.3	6.5	1.8	-1.3	10.7	

Exploring Data Sets:
Different time recordings of observations

```
ener <- read_csv(file = "06 rel data & models/Examples/Energy.csv")

wth <- read_csv(file = "06 rel data & models/Examples/Mac Head Wind Data.csv")
```

```
> glimpse(ener)
Observations: 2,784
Variables: 12
$ DateTime    <dttm> 2017-01-29 00:00:00,
$ Date        <chr> "29/01/17", "29/01/17
$ HourOfDay   <int> 0, 0, 0, 0, 1, 1, 1,
$ MinuteOfDay <int> 0, 15, 30, 45, 0, 15,
$ DayOfWeek   <chr> "Wed", "Wed", "Wed",
$ Demand      <int> 3834, 3785, 3708, 363
$ Generation  <int> 4041, 4041, 4130, 418
$ Wind        <int> 449, 505, 521, 492, 5
$ CO2         <int> 552, 548, 544, 543, 5
$ NetImports  <int> -145, -200, -294, -41
$ EWIC        <int> -33, -108, -183, -258
$ Moyle       <int> -112, -92, -111, -161
```

```
> glimpse(wth)
Observations: 31
Variables: 7
$ Date         <chr> "27/01/17", "28/01/17",
$ Rainfall     <dbl> 7.9, 3.5, 4.7, 7.8, 0.0
$ MaxTemp      <dbl> 8.7, 8.0, 9.0, 11.2, 10
$ MinTemp      <dbl> 4.3, 4.5, 4.9, 7.1, 7.3
$ GrassMinTemp <dbl> -0.7, 2.9, 3.7, 5.8, 5.
$ AVRWind      <dbl> 11.6, 12.6, 9.8, 14.3,
$ MaxWindGust  <int> NA, NA, NA, NA, NA, 38,
```

Need to join the data sets

```
> slice(ener,1:5)
# A tibble: 5 x 12
             DateTime     Date HourOfDay MinuteOfDay DayOfWeek Demand Generation  Wind   CO2
               <dttm>    <chr>     <int>       <int>     <chr>  <int>      <int> <int> <int>
1 2017-01-29 00:00:00 29/01/17         0           0       Wed   3834       4041   449   552
2 2017-01-29 00:15:00 29/01/17         0          15       Wed   3785       4041   505   548
3 2017-01-29 00:30:00 29/01/17         0          30       Wed   3708       4130   521   544
4 2017-01-29 00:45:00 29/01/17         0          45       Wed   3634       4181   492   543
5 2017-01-29 01:00:00 29/01/17         1           0       Wed   3581       4211   538   555
# ... with 3 more variables: NetImports <int>, EWIC <int>, Moyle <int>
>
> slice(wth,1:5)
# A tibble: 5 x 7
     Date Rainfall MaxTemp MinTemp GrassMinTemp AVRWind MaxWindGust
    <chr>    <dbl>   <dbl>   <dbl>        <dbl>   <dbl>       <int>
1 27/01/17      7.9     8.7     4.3         -0.7    11.6          NA
2 28/01/17      3.5     8.0     4.5          2.9    12.6          NA
3 29/01/17      4.7     9.0     4.9          3.7     9.8          NA
4 30/01/17      7.8    11.2     7.1          5.8    14.3          NA
5 31/01/17      0.0    10.3     7.3          5.8    10.0          NA
```

Approach: Average the power observations by day

```
> avr_daily_wind_gen <- ener %>% group_by(Date) %>%
+                                summarise(AverageWindGeneration=mean(Wind))
>
> avr_daily_wind_gen
# A tibble: 29 x 2
       Date AverageWindGeneration
      <chr>                  <dbl>
 1 01/02/17              2046.5521
 2 02/02/17              2647.0000
 3 03/02/17              1049.7500
 4 04/02/17               590.7708
 5 05/02/17               439.4688
 6 06/02/17              1970.5833
 7 07/02/17               393.7604
 8 08/02/17               970.7812
 9 09/02/17              2141.8958
10 10/02/17               548.3021
# ... with 19 more rows
```

Join the data sets on Date

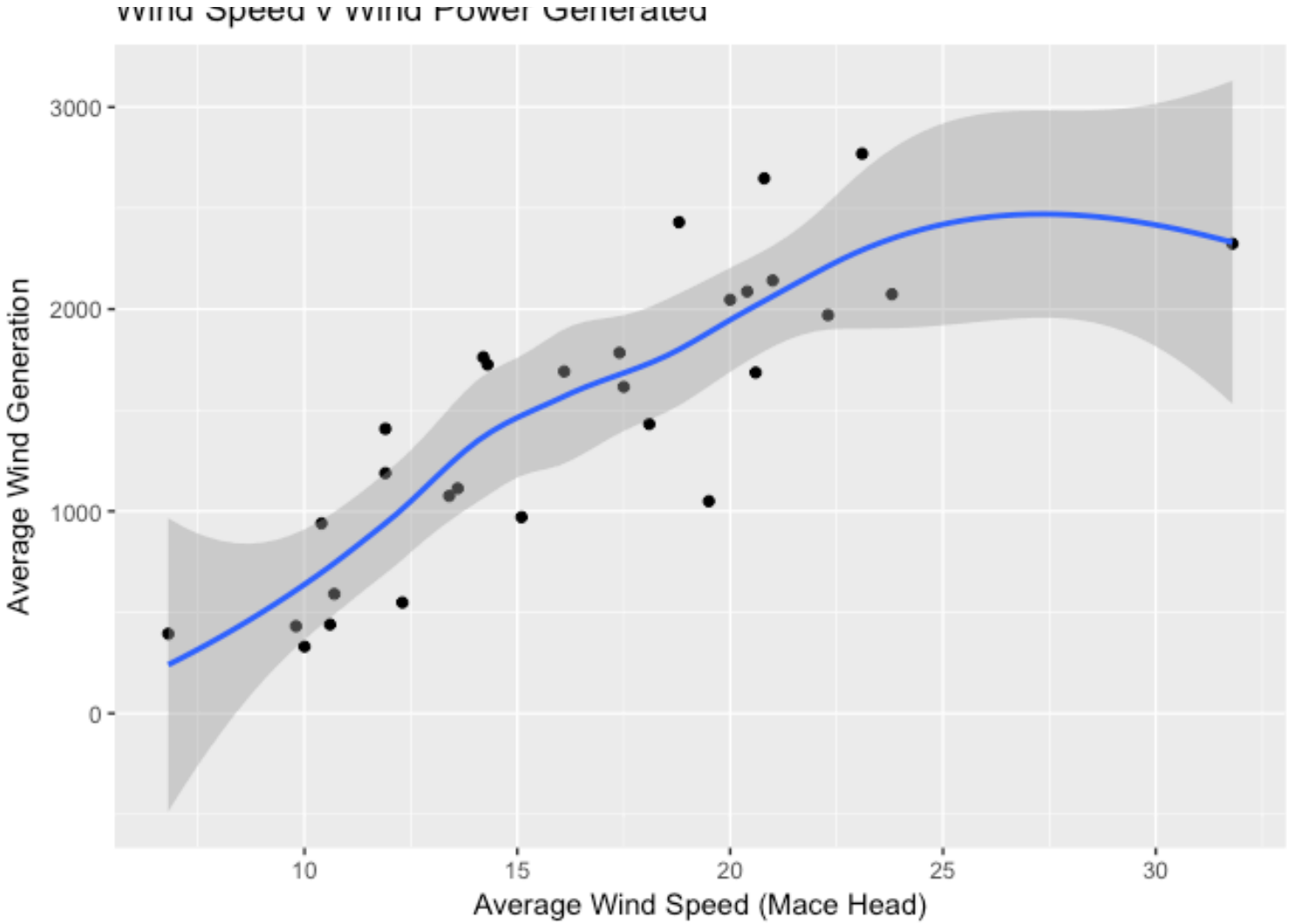
```
> select(wth,Date,AVRWind,everything())
# A tibble: 31 x 7
       Date AVRWind Rainfall MaxTemp MinTemp GrassMinTemp
      <chr>   <dbl>    <dbl>   <dbl>   <dbl>        <dbl>
 1 27/01/17    11.6      7.9     8.7     4.3         -0.7
 2 28/01/17    12.6      3.5     8.0     4.5          2.9
 3 29/01/17     9.8      4.7     9.0     4.9          3.7
 4 30/01/17    14.3      7.8    11.2     7.1          5.8
 5 31/01/17    10.0      0.0    10.3     7.3          5.8
 6 01/02/17    20.0      0.6    10.2     6.1          5.2
 7 02/02/17    20.8      4.9    11.2     7.4          6.4
 8 03/02/17    19.5      2.2     8.5     3.6          2.1
 9 04/02/17    10.7      5.3     6.5     1.8         -1.3
10 05/02/17    10.6      6.9     7.5     2.2         -1.4
# ... with 21 more rows, and 1 more variables:
#   MaxWindGust <int>
```

```
> avr_daily_wind_gen
# A tibble: 29 x 2
       Date AverageWindGeneration
      <chr>                 <dbl>
 1 01/02/17             2046.5521
 2 02/02/17             2647.0000
 3 03/02/17             1049.7500
 4 04/02/17              590.7708
 5 05/02/17              439.4688
 6 06/02/17             1970.5833
 7 07/02/17              393.7604
 8 08/02/17              970.7812
 9 09/02/17             2141.8958
10 10/02/17              548.3021
# ... with 19 more rows
```

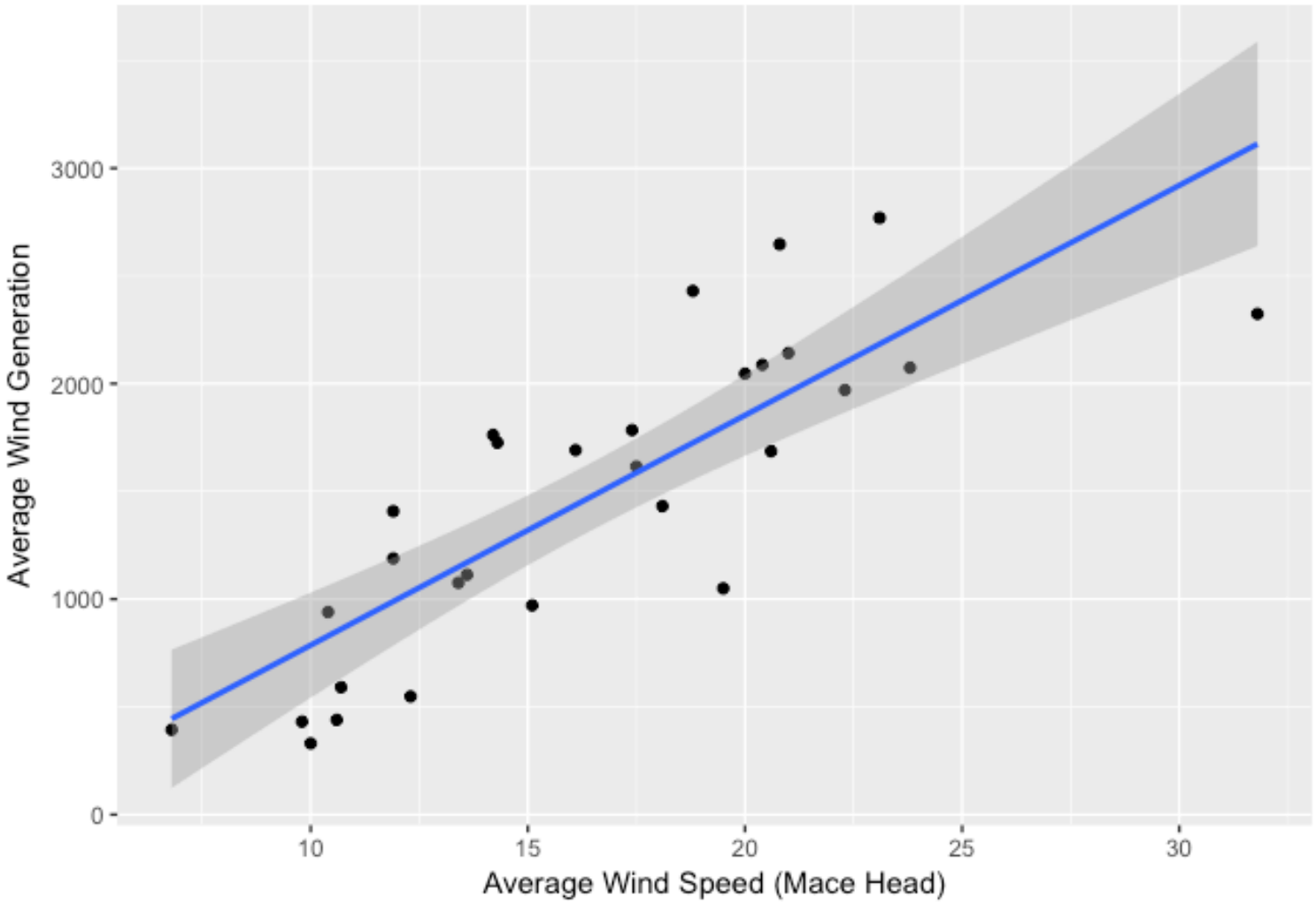
Resulting (left) join

```
> merged <- left_join(avr_daily_wind_gen,wth)
Joining, by = "Date"
>
> merged
# A tibble: 29 x 8
       Date AverageWindGeneration Rainfall MaxTemp MinTemp GrassMinTemp AVRWind
      <chr>                 <dbl>    <dbl>   <dbl>   <dbl>         <dbl>   <dbl>
 1 01/02/17             2046.5521      0.6    10.2     6.1           5.2    20.0
 2 02/02/17             2647.0000      4.9    11.2     7.4           6.4    20.8
 3 03/02/17             1049.7500      2.2     8.5     3.6           2.1    19.5
 4 04/02/17              590.7708      5.3     6.5     1.8          -1.3    10.7
 5 05/02/17              439.4688      6.9     7.5     2.2          -1.4    10.6
 6 06/02/17             1970.5833      7.6    11.3     5.9           4.8    22.3
 7 07/02/17              393.7604     13.3     8.1     2.7          -3.3     6.8
 8 08/02/17              970.7812      0.0     9.1     6.1           1.9    15.1
 9 09/02/17             2141.8958      0.0     9.0     1.3          -0.6    21.0
10 10/02/17              548.3021      0.0     4.2     0.9          -1.3    12.3
# ... with 19 more rows, and 1 more variables: MaxWindGust <int>
```

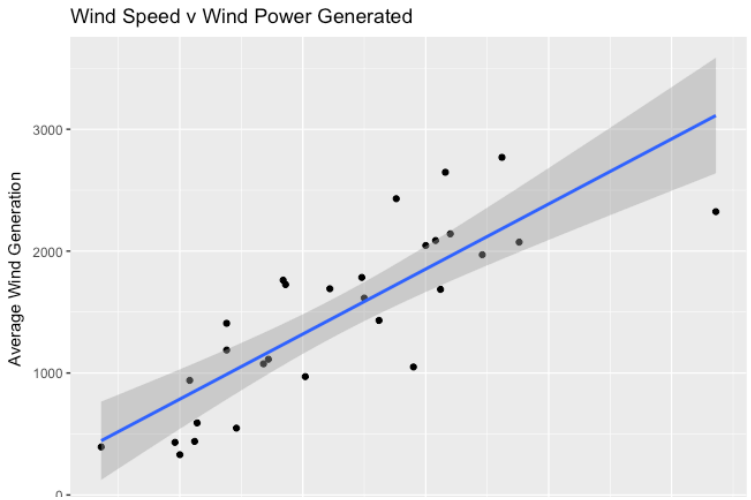
```
ggplot(data = merged,mapping = aes(x=AVRWind,y=AverageWindGeneration)) +
  geom_point() +
  geom_smooth()+
  xlab("Average Wind Speed (Mace Head)") + ylab("Average Wind Generation") +
  ggtitle("Wind Speed v Wind Power Generated")
```



```
ggplot(data = merged,mapping = aes(x=AVRWind,y=AverageWindGeneration)) +
  geom_point() +
  geom_smooth(method="lm")+
  xlab("Average Wind Speed (Mace Head)") + ylab("Average Wind Generation") +
  ggtitle("Wind Speed v Wind Power Generated")
```



Building a linear model...

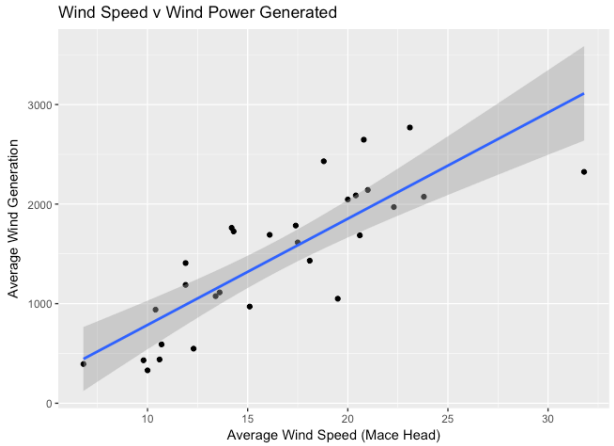



```
> mod <- lm(data = merged,AverageWindGeneration~AVRWind)
>
> mod

Call:
lm(formula = AverageWindGeneration ~ AVRWind, data = merged)

Coefficients:
(Intercept)      AVRWind
     -280.8        106.7
```

Predicting future values…




```
> p1 <- predict(mod,newdata = data.frame(AVRWind=25))
>
> p1
       1
2386.727
>
> p2 <- predict(mod,newdata = data.frame(AVRWind=30))
>
> p2
       1
2920.228
```