Smoothing

J. Di Iorio, F. Chiaromonte

1/1/2020

Libraries

We are going to use **tidyverse** and **ggplot2**.

```
library(tidyverse) # for data manipulation and visualization
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2) # for plots
```

Data

We will try to locally regress and smooth the median duration of unemployment (uempmed) based on the **economics** dataset from **ggplot2** package. We wil focus on the latest 120 months (10 years from 2005 to 2015)

```
data(economics)
help(economics)
head(economics)
```

```
## # A tibble: 6 x 6
##   date         pce    pop psavert uempmed unemploy
##   <date>     <dbl>  <dbl>   <dbl>   <dbl>    <dbl>
## 1 1967-07-01  507. 198712    12.6     4.5     2944
## 2 1967-08-01  510. 198911    12.6     4.7     2945
## 3 1967-09-01  516. 199113    11.9     4.6     2958
## 4 1967-10-01  512. 199311    12.9     4.9     3143
## 5 1967-11-01  517. 199498    12.8     4.7     3066
## 6 1967-12-01  525. 199657    11.8     4.8     3018
```

```
dim(economics)
```

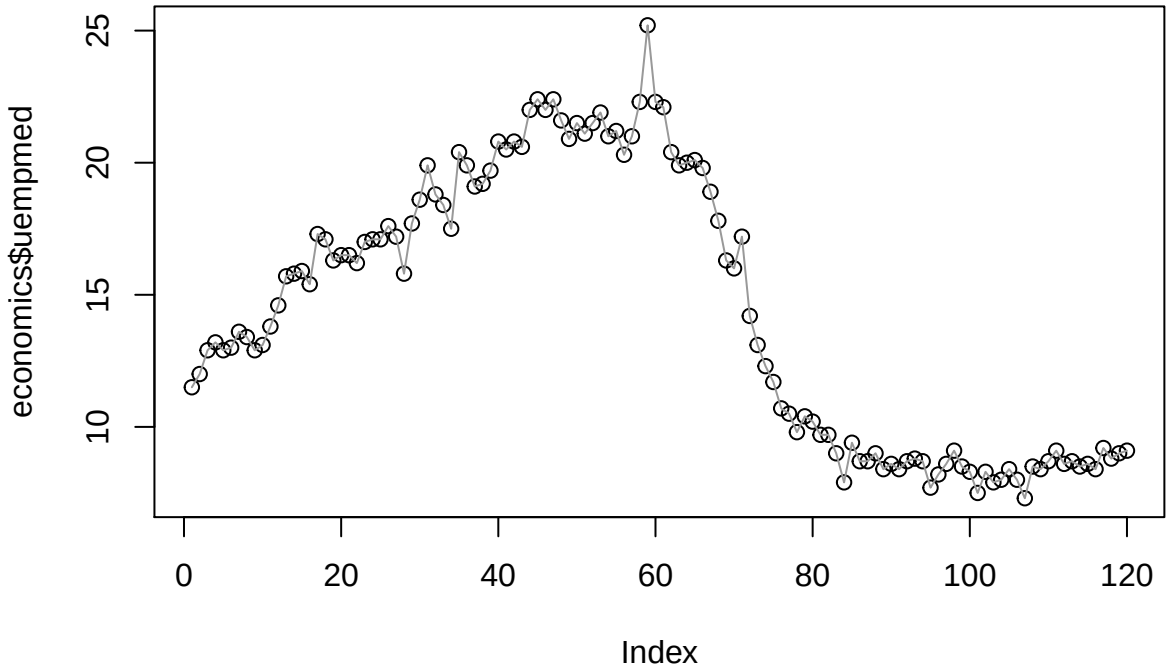
```
## [1] 574   6
```

We focus on the latest 120 months.

```
economics <- economics[dim(economics)[1]:(dim(economics)[1]-119),]
dim(economics)
```

```
## [1] 120   6
```

```
plot(economics$uempmed)
lines(economics$uempmed, col='grey60')
```



Transform the date in a index from 1 (first measurement in 2005) to 120 (latest measurement in 2015).

```
economics$index <- 1:120
```

LOWESS

Perform LOWESS in the stats package with the loess command

```
help(loess)
```

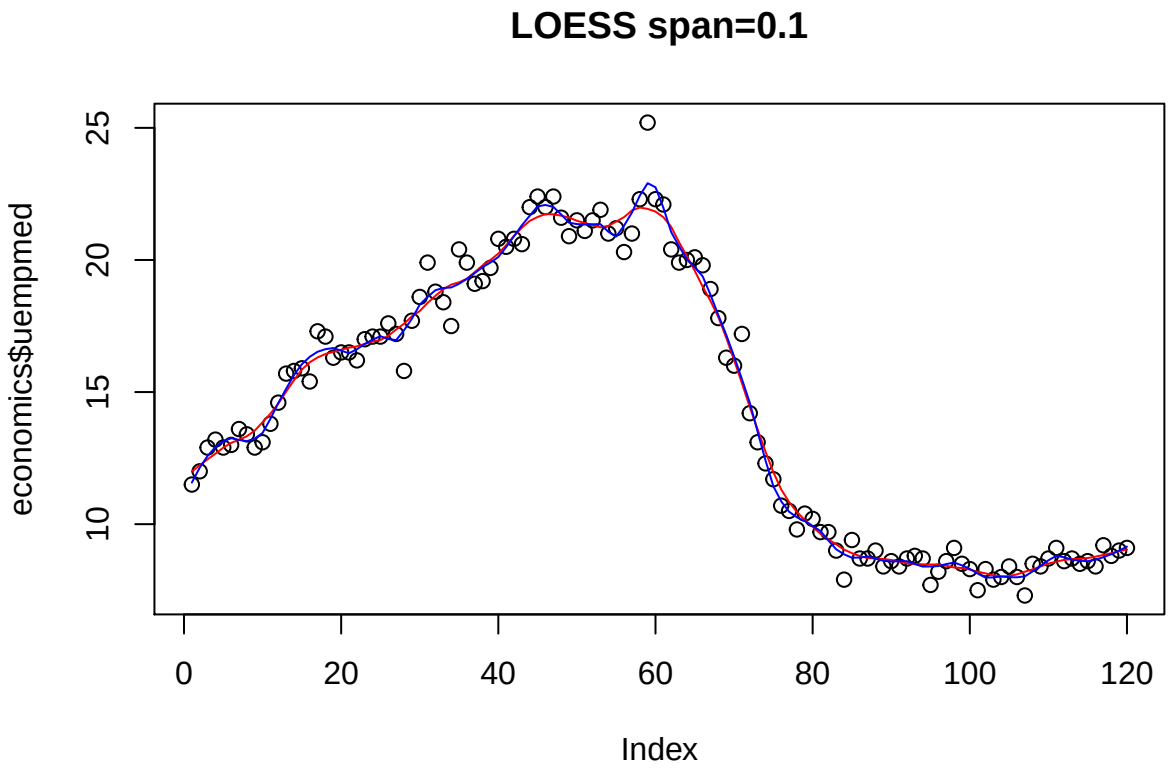
Let us focus on the following arguments:

- **formula:** a formula specifying the numeric response and one to four numeric predictors;
- **data:** the dataframe;
- **span:** the parameter which controls the degree of smoothing;
- **degree:** the degree of the polynomials to be used, normally 1 or 2;
- **family:** if gaussian fitting is by least-squares, and if symmetric a redescending estimator is used with Tukey's biweight function;

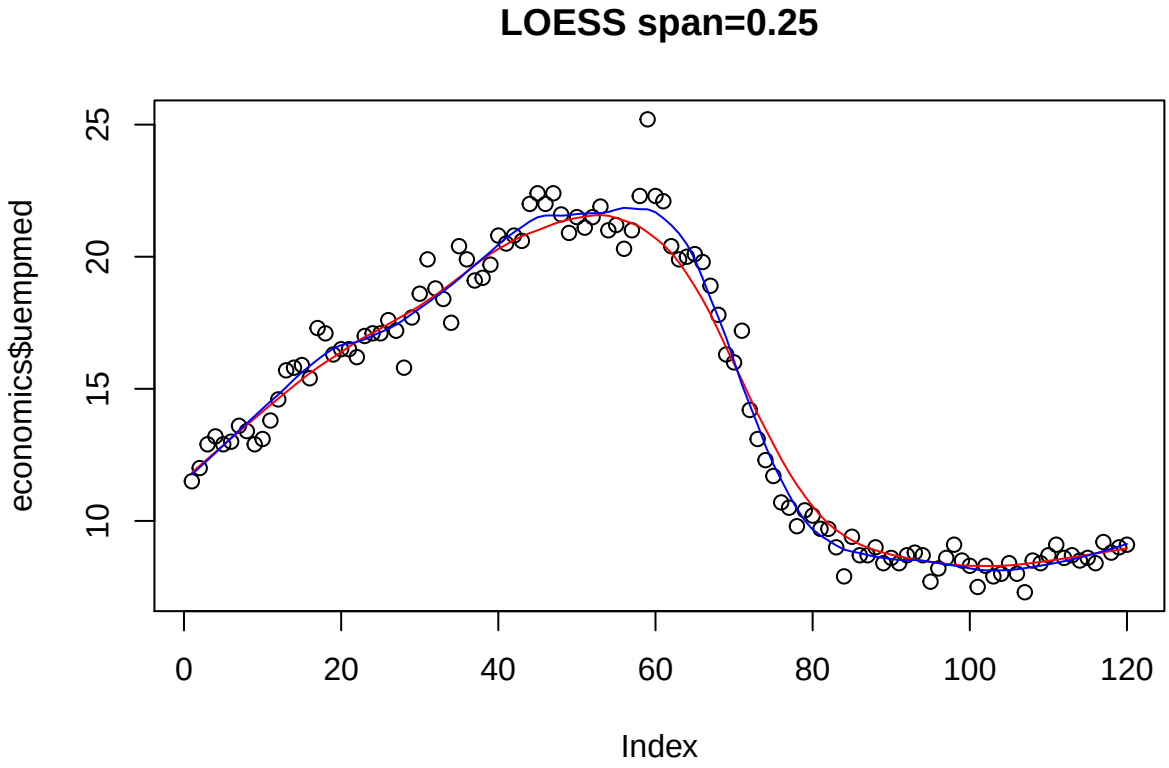
Let us try the different spans and different degrees.

```
loess1_10 <- loess(uempmed ~ index, data = economics, span = 0.1, degree=1)
loess2_10 <- loess(uempmed ~ index, data = economics, span = 0.1, degree=2)
plot(economics$uempmed, main="LOESS span=0.1")
```

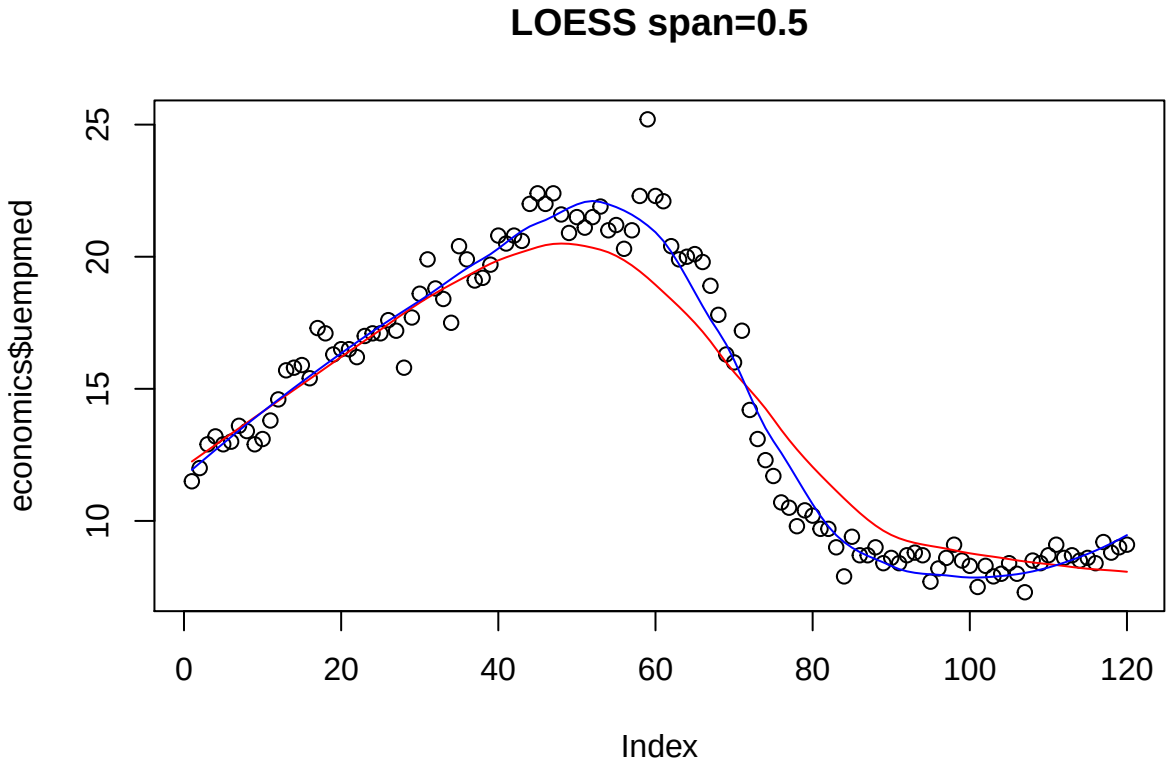
```
lines(predict(loess1_10), col='red')
lines(predict(loess2_10), col='blue')
```




```
# span=0.25
loess1_25 <- loess(uempmed ~ index, data = economics, span = 0.25, degree=1)
loess2_25 <- loess(uempmed ~ index, data = economics, span = 0.25, degree=2)
plot(economics$uempmed, main="LOESS span=0.25")
lines(predict(loess1_25), col='red')
lines(predict(loess2_25), col='blue')
```

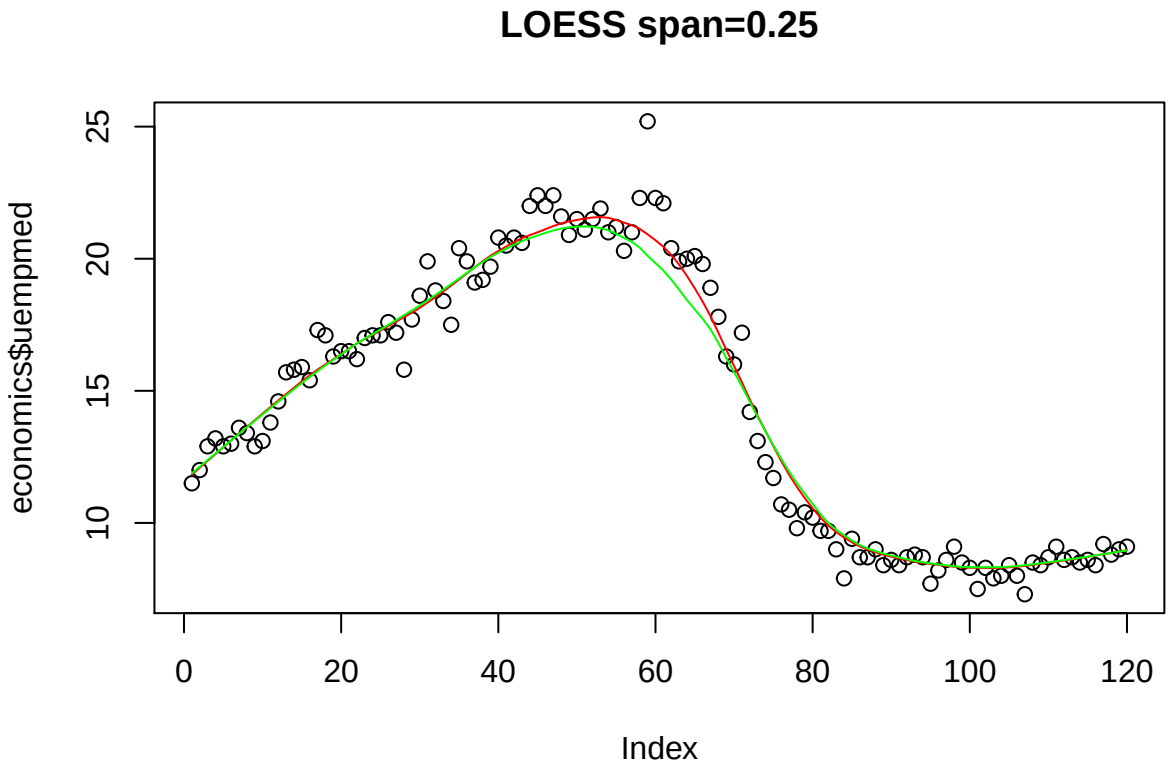



```
# span=0.5
loess1_50 <- loess(uempmed ~ index, data = economics, span = 0.5, degree=1)
loess2_50 <- loess(uempmed ~ index, data = economics, span = 0.5, degree=2)
plot(economics$uempmed, main="LOESS span=0.5")
lines(predict(loess1_50), col='red')
lines(predict(loess2_50), col='blue')
```



loess has the option of fitting the local model with an iterative algorithm which, after fitting a model in one iteration, detects outliers and down-weight them for the next iteration. To use this option, we use the argument family="symmetric".

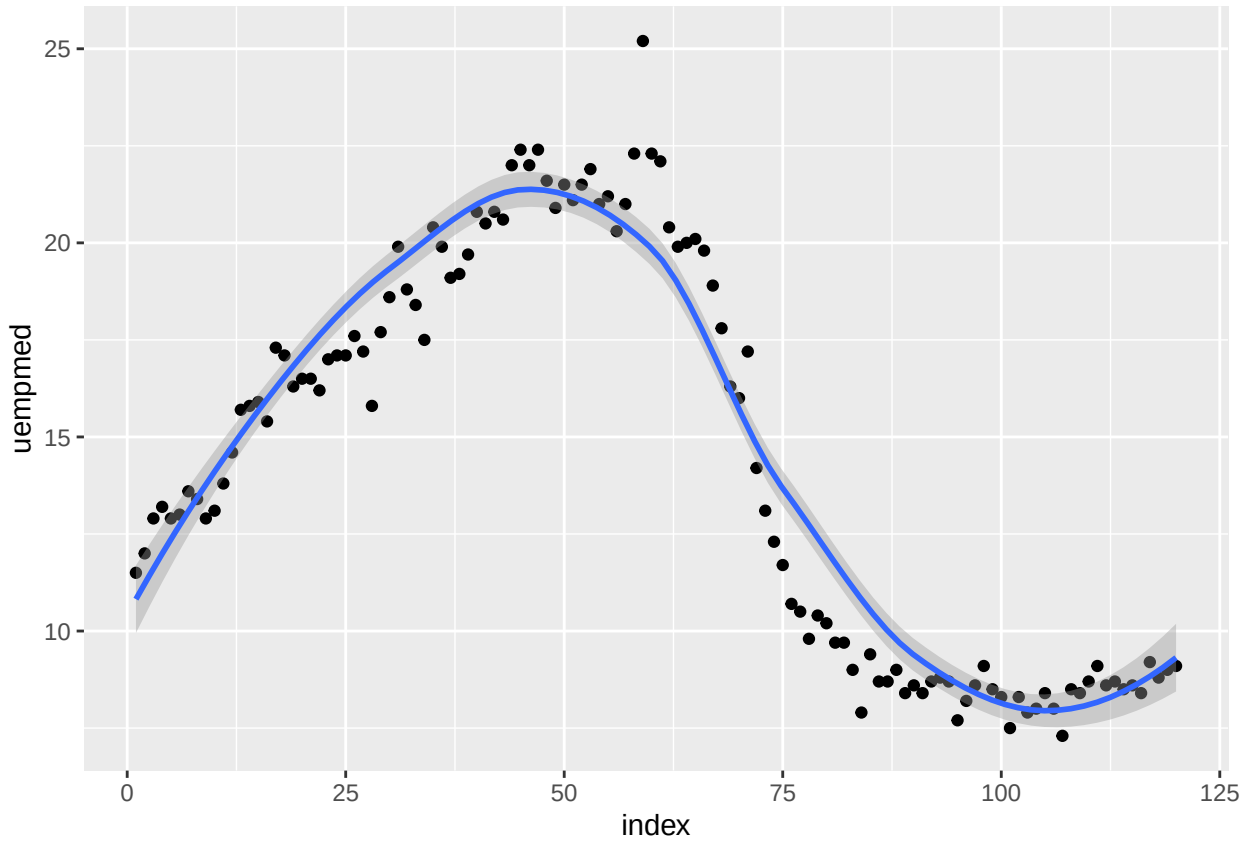
```
# span=0.25
loess1_25sim <- loess(uempmed ~ index, data = economics, span = 0.25, degree=1, family="symmetric")
plot(economics$uempmed, main="LOESS span=0.25")
lines(predict(loess1_25), col='red')
lines(predict(loess1_25sim), col='green',)
```



ggplot uses loess in its geom_smooth function

```
economics %>% ggplot(aes(index,uempmed)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

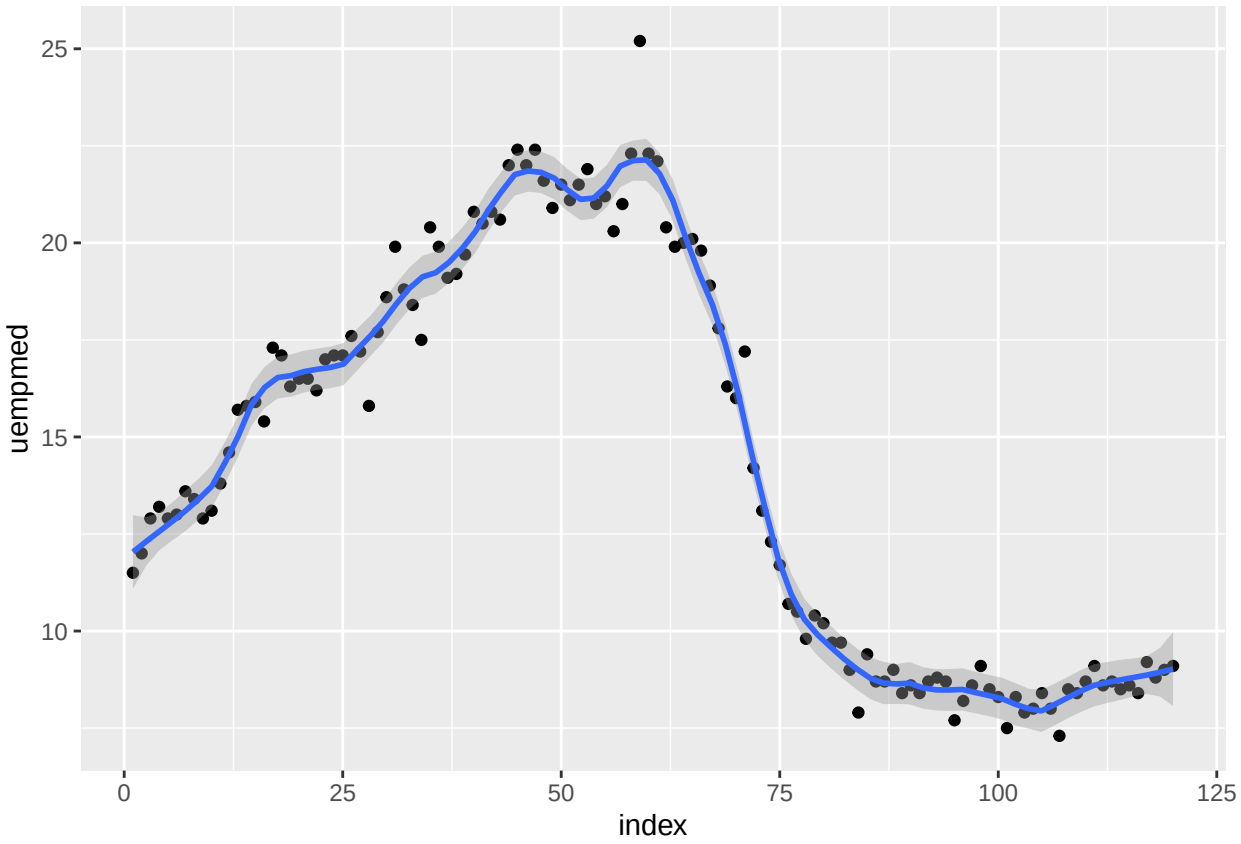


We don't know the default parameters used in geom_smooth but we can change them in this way:

```
economics %>% ggplot(aes(index,uempmed)) + geom_point() + geom_smooth(method="loess", span=0.15, methods
```

```
## Warning: Ignoring unknown parameters: methods.args
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Bin Smoothing and Kernel Smoothing

The general idea of smoothing is to group data points into strata in which the trend behind the data changes slowly. For this reason we can assumed the trend to be constant in a small window. In our case we can assume that the unemployment remained approximately the same within 3 month's time.

The assumption implies that a good estimate is the average of the values in the window (in this case: 3 month's time). By computing this mean for every point (moving the window), we form an estimate fo the underlying curve.

The command that we are going to use is **ksmooth**.

```
help(ksmooth)
```

If the mean is computed giving the same weights to the points in the window we talk about "box" kernel. The result is a list with the original **x** and the new smoothed values **y**.

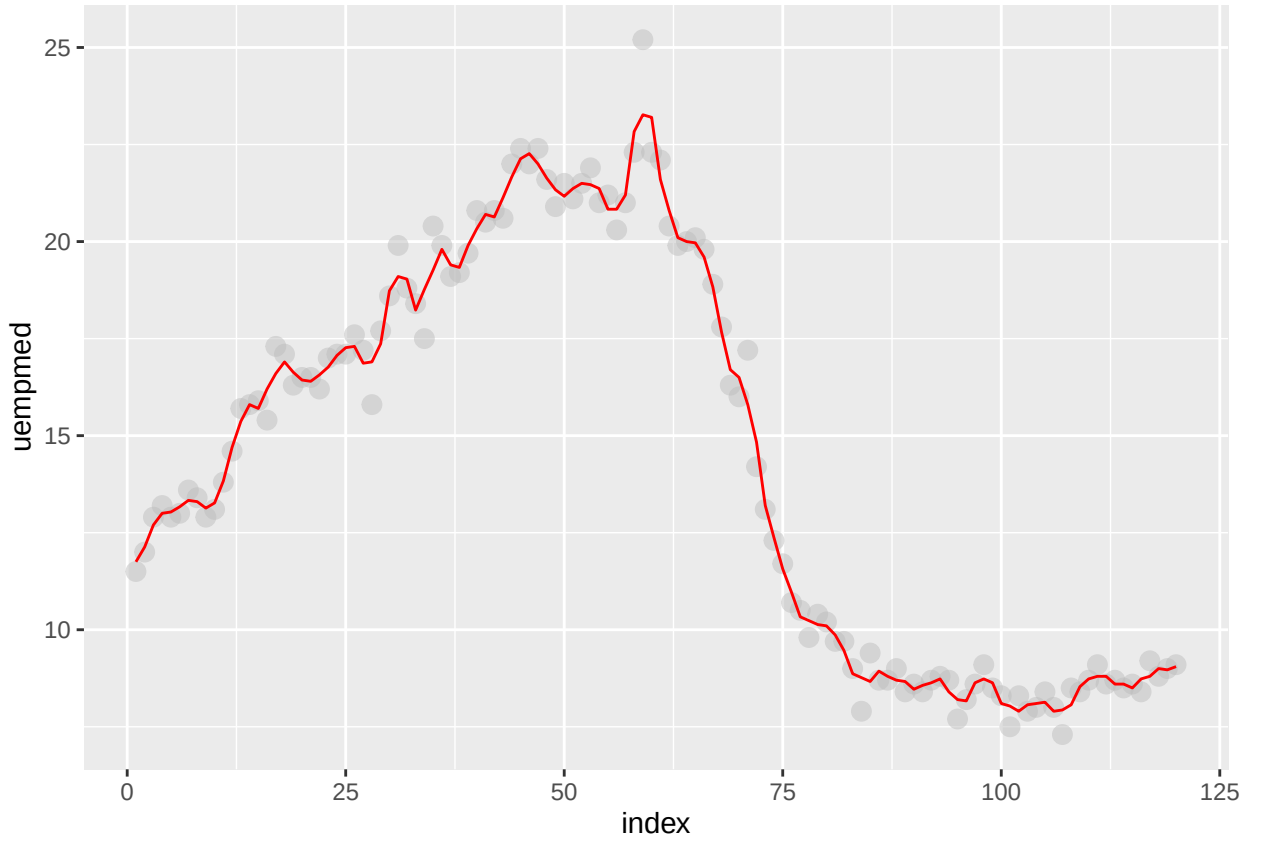
```
window <- 3
box_smooth <- ksmooth(economics$index, economics$uempmed, kernel='box', bandwidth = window)
box_smooth
```

```
## $x
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
```

```
## [109] 109 110 111 112 113 114 115 116 117 118 119 120
##
## $y
##   [1] 11.750000 12.133333 12.700000 13.000000 13.033333 13.166667 13.333333
##   [8] 13.300000 13.133333 13.266667 13.833333 14.700000 15.366667 15.800000
##  [15] 15.700000 16.200000 16.600000 16.900000 16.633333 16.433333 16.400000
##  [22] 16.566667 16.766667 17.066667 17.266667 17.300000 16.866667 16.900000
##  [29] 17.366667 18.733333 19.100000 19.033333 18.233333 18.766667 19.266667
##  [36] 19.800000 19.400000 19.333333 19.900000 20.333333 20.700000 20.633333
##  [43] 21.133333 21.666667 22.133333 22.266667 22.000000 21.633333 21.333333
##  [50] 21.166667 21.366667 21.500000 21.466667 21.366667 20.833333 20.833333
##  [57] 21.200000 22.833333 23.266667 23.200000 21.600000 20.800000 20.100000
##  [64] 20.000000 19.966667 19.600000 18.833333 17.666667 16.700000 16.500000
##  [71] 15.800000 14.833333 13.200000 12.366667 11.566667 10.966667 10.333333
##  [78] 10.233333 10.133333 10.100000  9.866667  9.466667  8.866667  8.766667
##  [85]  8.666667  8.933333  8.800000  8.700000  8.666667  8.466667  8.566667
##  [92]  8.633333  8.733333  8.400000  8.200000  8.166667  8.633333  8.733333
##  [99]  8.633333  8.100000  8.033333  7.900000  8.066667  8.100000  8.133333
## [106]  7.900000  7.933333  8.066667  8.533333  8.733333  8.800000  8.800000
## [113]  8.600000  8.600000  8.500000  8.733333  8.800000  9.000000  8.966667
## [120]  9.050000
```

Let us plot our result using ggplot (not ordinary plot).

```
economics %>% mutate(smooth = box_smooth$y) %>% ggplot(aes(index, uempmed)) +
  geom_point(size=3, alpha=0.5, color='grey') + geom_line(aes(index, smooth), color='red')
```



The result from the bin box smoother is quite wiggly. The reasons for this can be the bandwidth (too small)

or the uniform weights. We can try to change the weights by giving to the center point of the window, more weight than far away points (so the points at the edge will receive very little weight). So we are using a weighted average where the weights are provided by a normal density.

```
norm_smooth <- ksmooth(economics$index, economics$uempmed, kernel='normal', bandwidth = window)

economics %>% mutate(smooth = norm_smooth$y) %>% ggplot(aes(index, uempmed)) +
  geom_point(size=3, alpha=0.5, color='grey') + geom_line(aes(index, smooth), color='red')
```

It is still wiggly! We need to change the bandwidth.

```
window <- 6 #6 month's time
norm_smooth <- ksmooth(economics$index, economics$uempmed, kernel='normal', bandwidth = window)

economics %>% mutate(smooth = norm_smooth$y) %>% ggplot(aes(index, uempmed)) +
  geom_point(size=3, alpha=0.5, color='grey') + geom_line(aes(index, smooth), color='red')
```

Quick Idea on Kernel Density Estimator

Let us simulate a new dataset. In the data we have the sex and the weights of 400 Serperior living in the Unima region.

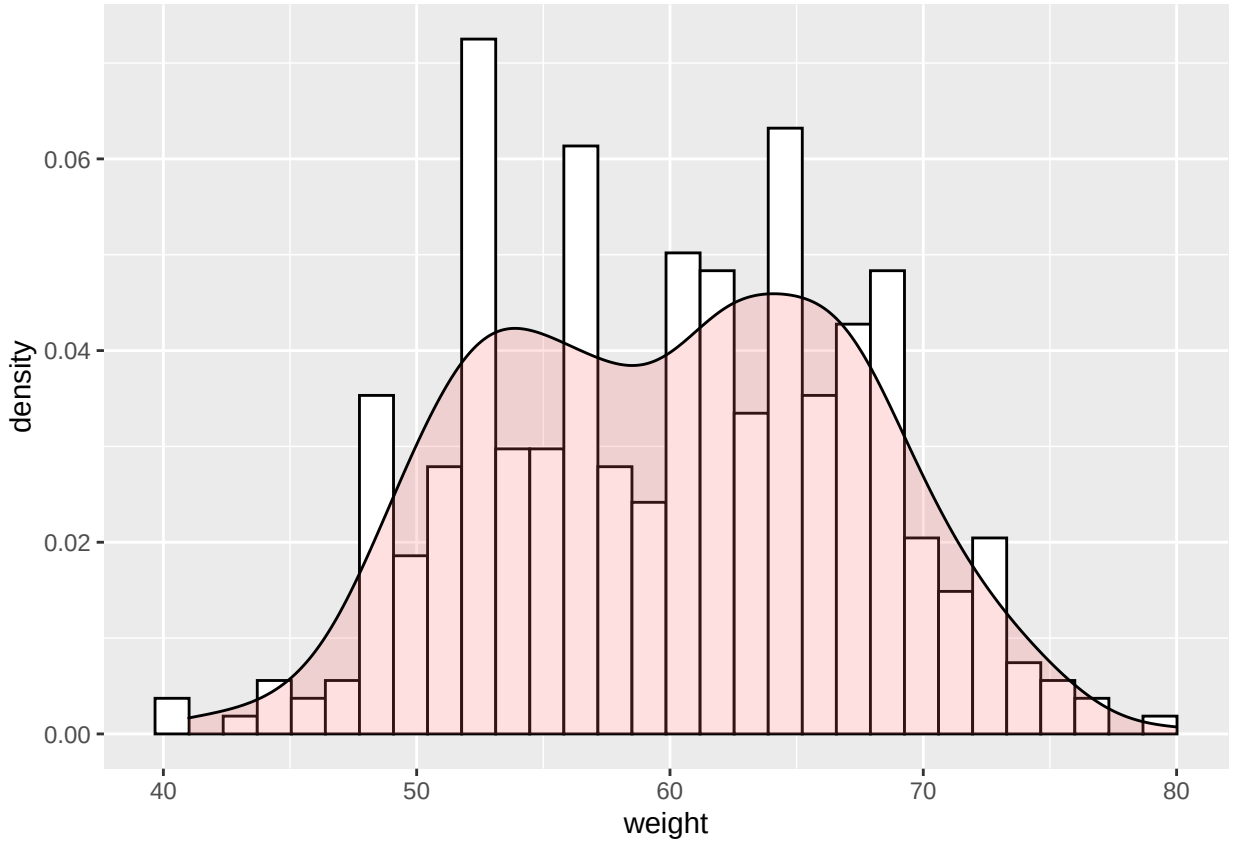
```
set.seed(1234)
df <- data.frame(
  sex=factor(rep(c("F", "M"), each=200)),
  weight=round(c(rnorm(200, mean=55, sd=5),
                 rnorm(200, mean=65, sd=5)))
)
head(df)
```

```
##   sex weight
## 1   F     49
## 2   F     56
## 3   F     60
## 4   F     43
## 5   F     57
## 6   F     58
```

Let use a histograms

```
ggplot(df, aes(x=weight)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white")+
  geom_density(alpha=.2, fill="#FF6666")
```

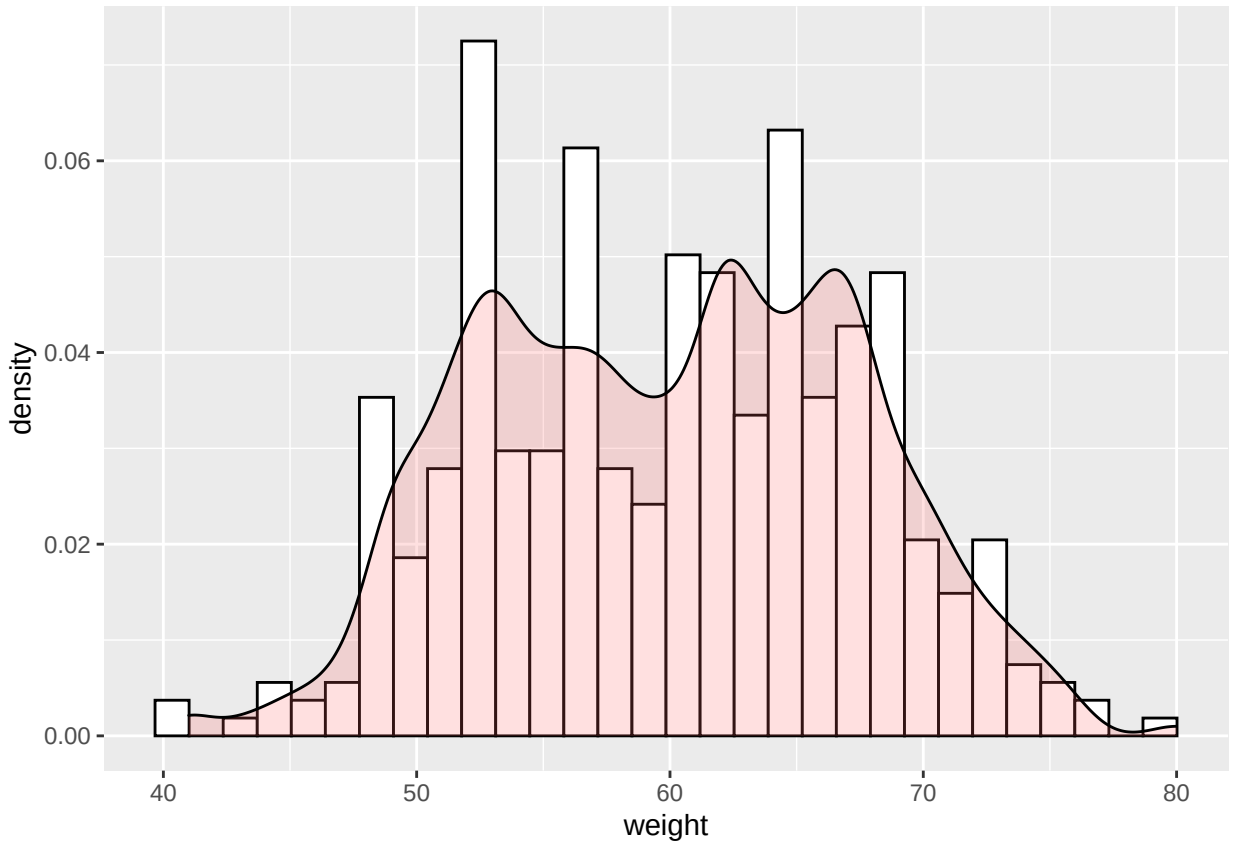
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We can adjust the default density with the adjust arguments (default is 1). What is the adjust function used for?

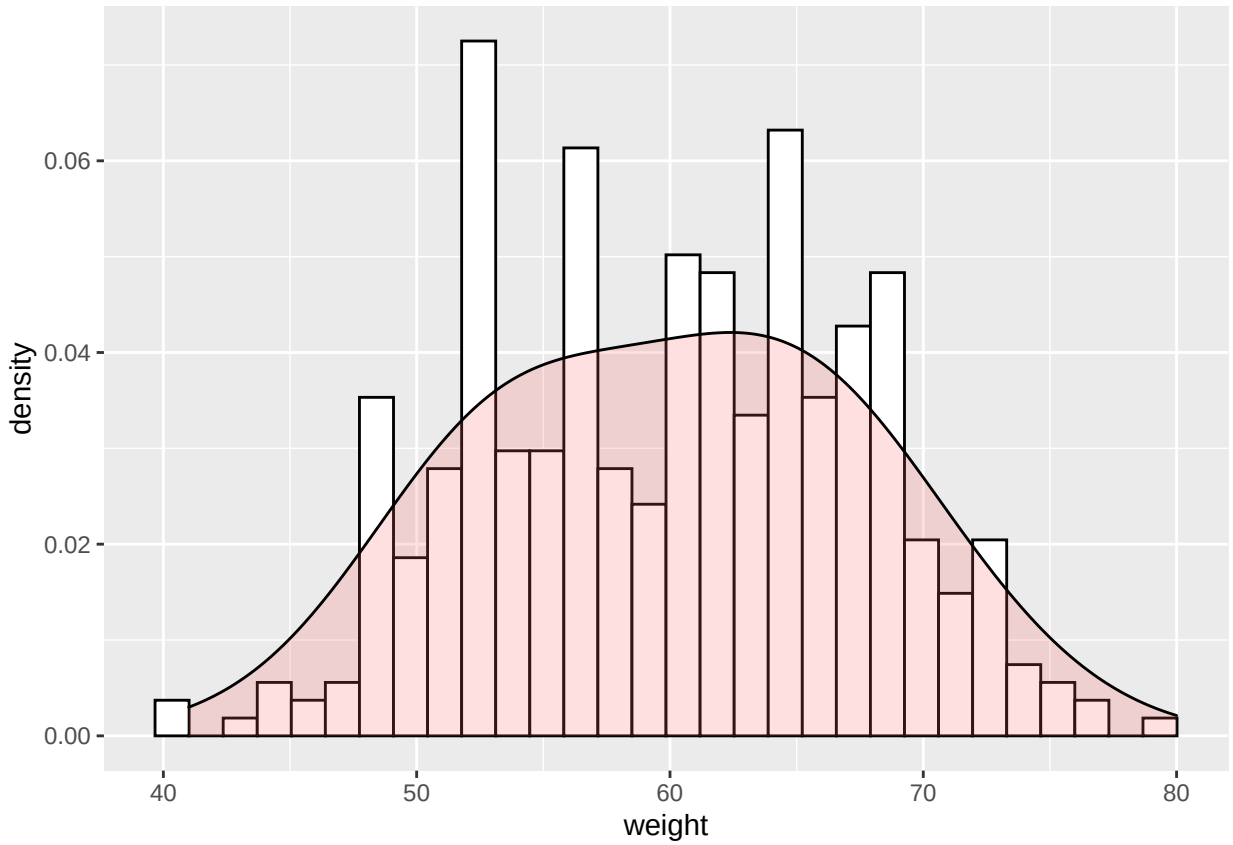
```
ggplot(df, aes(x=weight)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white")+
  geom_density(alpha=.2, fill="#FF6666", adjust=1/2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(df, aes(x=weight)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white")+
  geom_density(alpha=.2, fill="#FF6666", adjust=2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


density
0.06
0.04
0.02
0.00
40
50
60
70
80
weight