mixtureReg: A Quick Start

In this tutorial, we are going to show how to use **mixtureReg** to model data with two possible regimes.

Data

The data used for demostration purpose here is the CO2 data set from the **mixtools** package.

```
library(mixtools)
```

```
## Warning: package 'mixtools' was built under R version 3.3.2
```

```
## mixtools package, version 1.1.0, Released 2017-03-10
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518
```

```
data("CO2data")
head(CO2data)
```

```
##     GNP  CO2 country
## 1 19.02 14.7     CAN
## 2  3.67  3.9     MEX
## 3 28.20 20.8     USA
## 4 40.94  9.0     JAP
## 5 10.61  8.3     KOR
## 6 20.09 16.0     AUS
```

A simple example

The motivation of mixture of regressions is that there can be two different regimes in the data so we want to fit two lines through the data.

We can easily achieve this by putting two regression formula into a list and feed it into the **mixtureReg** function.

In this case, the message shows that the model converges in 32 iterations.

```
library(mixtureReg)

mx1 <- mixtureReg(
  regData = CO2data,
  formulaList = list(formula(CO2 ~ GNP),
                     formula(CO2 ~ GNP)),
  mixingProb = "Constant"
)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

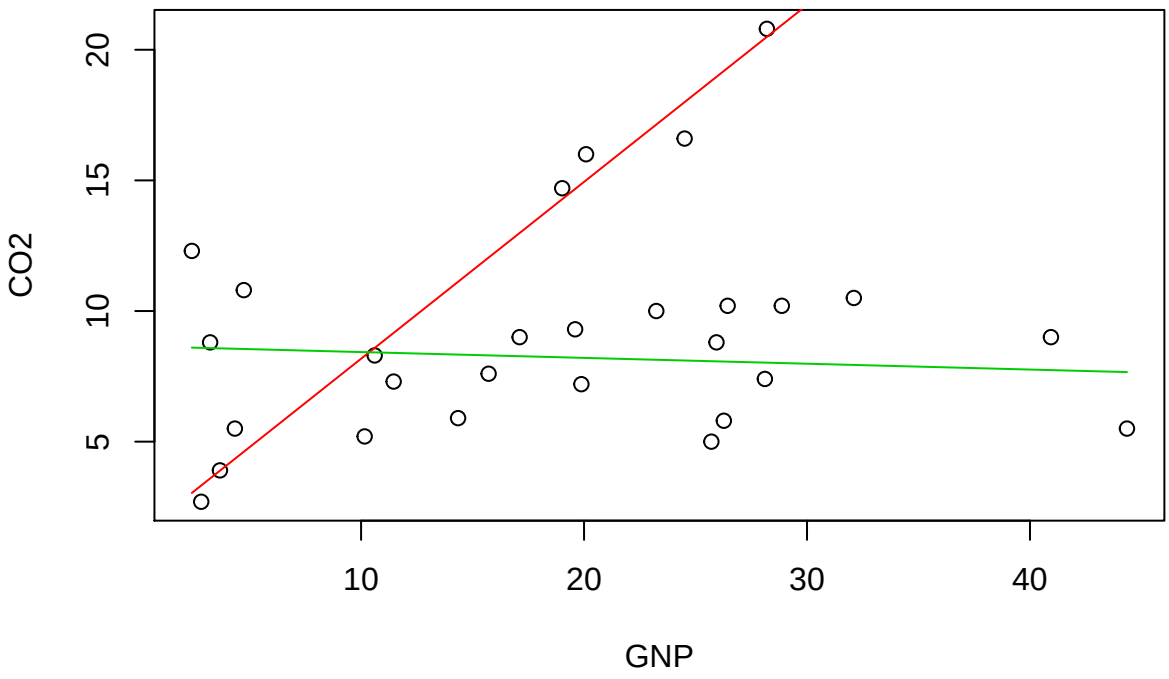
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## diff =  4.370719e-09
## iter =  32
## restart =  0
## log-likelihood =  -67.19407
```

The fit

We provide a plot method (S3 method) to visualize the predictions from the model. The circles below are the original data points and the red lines are predictions from our model.

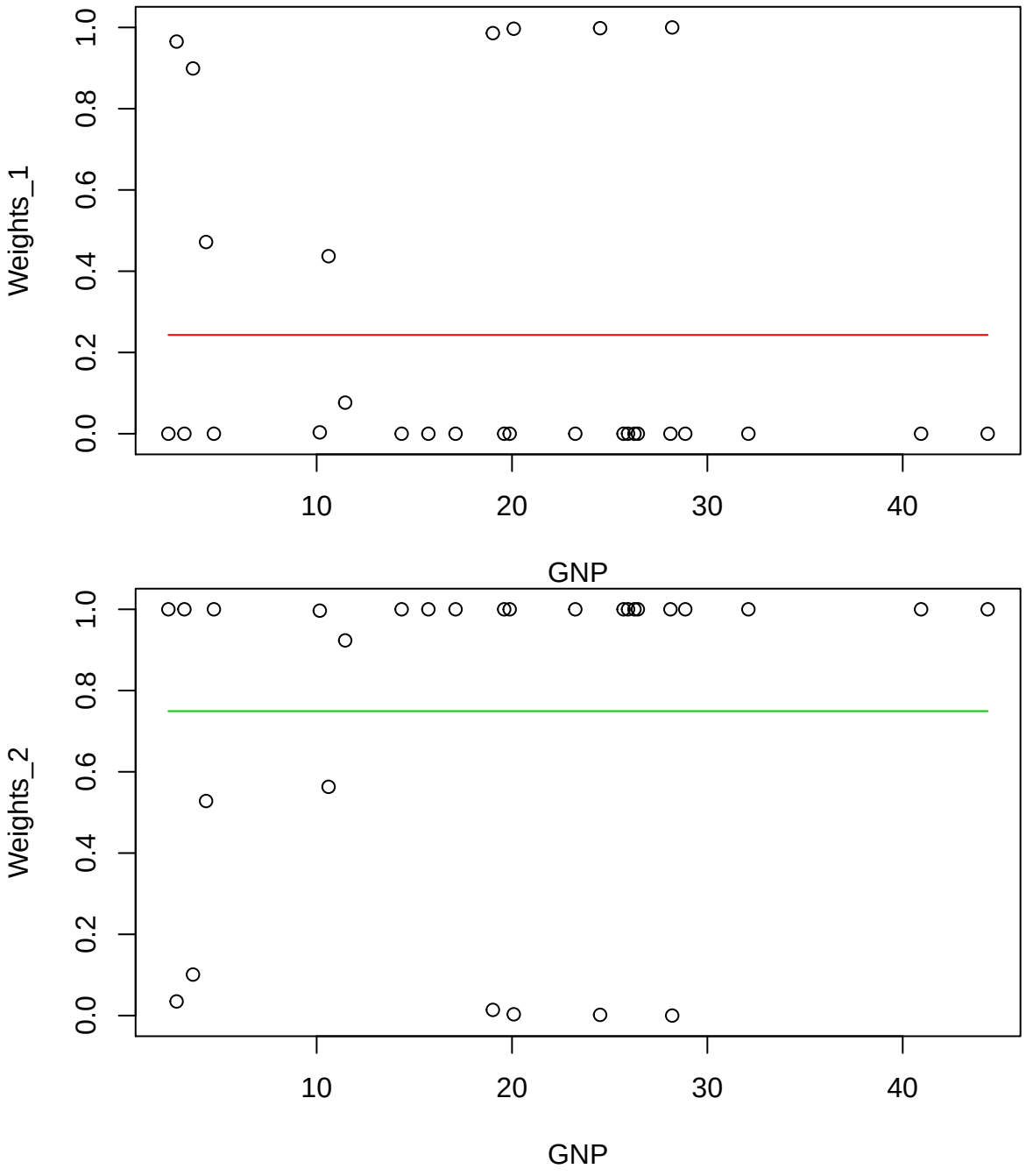
```
plot(mx1, which = 1)
```



The weights

Other than predictions, The mixture of regressions also produces weight estimates for each data point which indicate the posterior probabilities of membership to the regression lines. We provide another plot method to visualize these weights.

```
plot(mx1, which = 2)
```



The iterations

(Mainly for debugging purposes) We also provide a *monitor* component for modelers to learn more about what are happening in iterations.

```
head(mx1$monitor)
```

```
##        diff iter restart    logLik     newLL   sigma1   sigma2    ratio
## 1 1.0000000    0       0 -77.94643        NA 4.102895 4.002251 1.025147
## 2 0.1189329    1       0 -77.82750 -77.82750 4.232418 3.849434 1.099491
## 3 0.3452044    2       0 -77.48230 -77.48230 4.396608 3.601352 1.220821
## 4 0.7357961    3       0 -76.74650 -76.74650 4.571713 3.219560 1.419981
```

```
## 5 0.9742591    4       0 -75.77224 -75.77224 4.676884 2.799592 1.670559
## 6 0.8264774    5       0 -74.94576 -74.94576 4.680056 2.476780 1.889573
##     lambda1   lambda2 error_message
## 1 0.4990165 0.5009835            NA
## 2 0.4983732 0.5016268            NA
## 3 0.4957774 0.5042226            NA
## 4 0.4879779 0.5120221            NA
## 5 0.4701194 0.5298806            NA
## 6 0.4425669 0.5570970            NA
```

Flexible modeling

A nice feature of this package is that we can flexibly specify the formula as we would in **lm**.

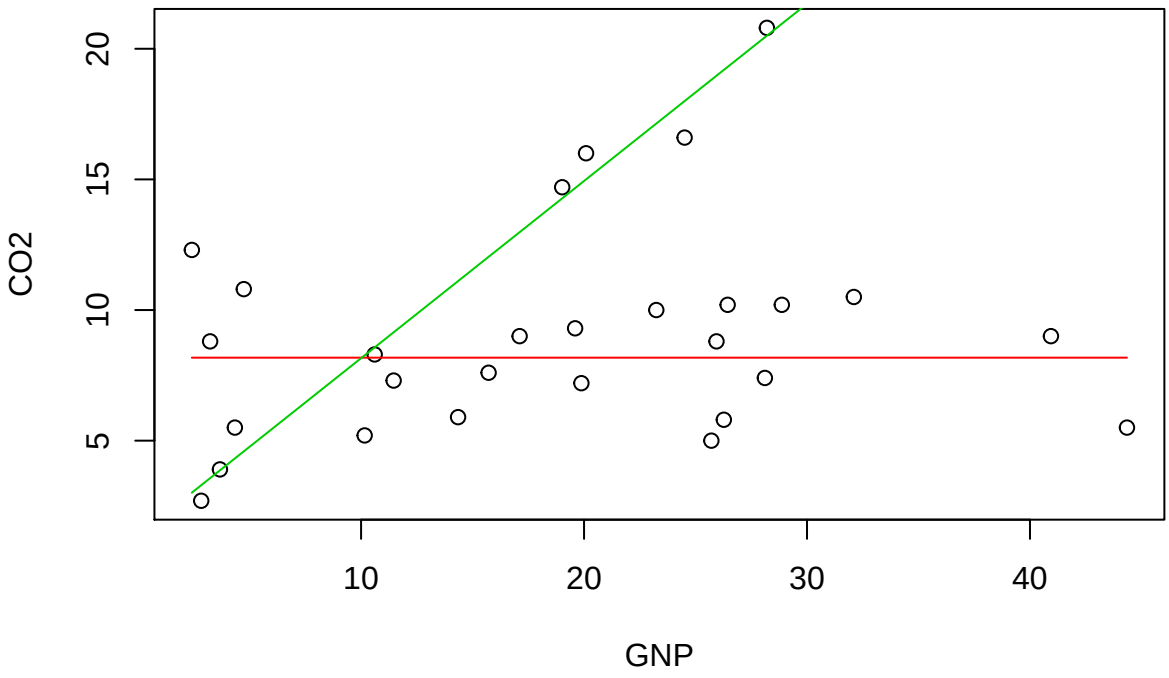
For example, we can restrict one regression line to be horizontal with no slope coefficient.

This example also helps to demonstrate the usage of "yName" and "xName" arguments in the plot method. When not specified, the plot method will search the variables in the first formula, which will not work in this case.

```
mx2 <- mixtureReg(
  regData = CO2data,
  formulaList = list(formula(CO2 ~ 1),
                     formula(CO2 ~ GNP)),
  mixingProb = "Constant"
  )
```

```
## diff =  4.990326e-09
## iter =  24
## restart =  0
## log-likelihood =  -67.31655
```

```
plot(mx2, yName = "CO2", xName = "GNP", which = 1)
```

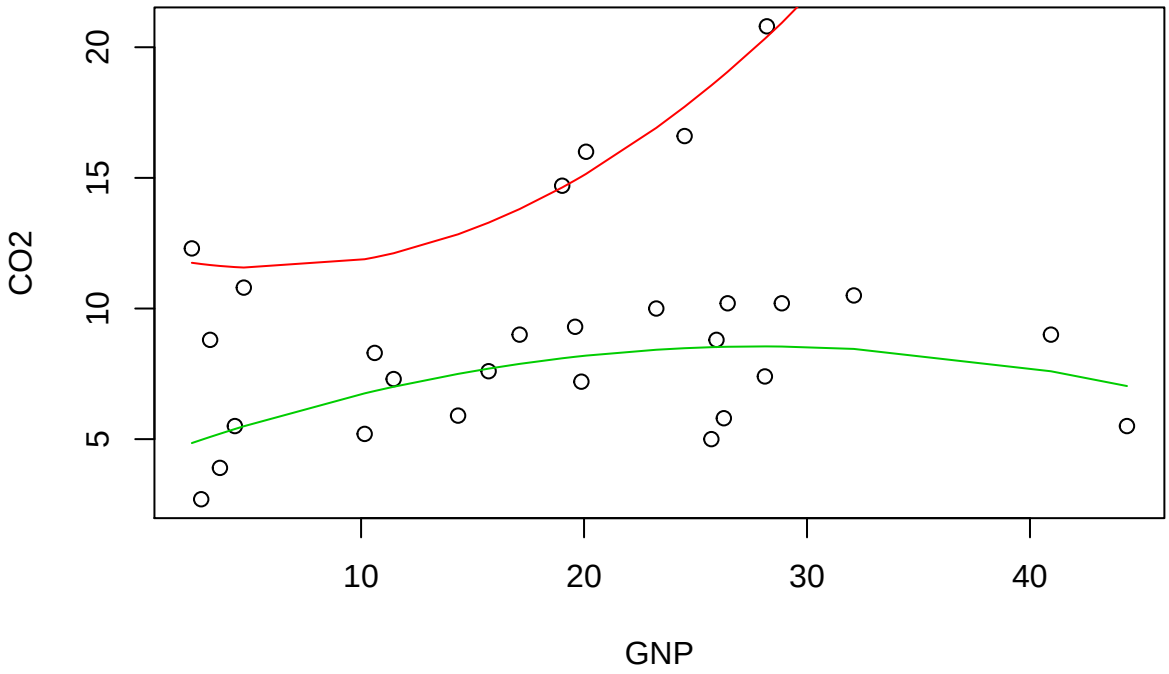


We can also specify 2nd order polynomial lines.

```
mx3 <- mixtureReg(
  regData = CO2data,
  formulaList = list(formula(CO2 ~ GNP + I(GNP^2)),
                     formula(CO2 ~ GNP + I(GNP^2))),
  mixingProb = "Constant"
  )
```

```
## diff =  -8.289391e-09
## iter =  58
## restart =  0
## log-likelihood =  -65.17028
```

```
plot(mx3, which = 1)
```

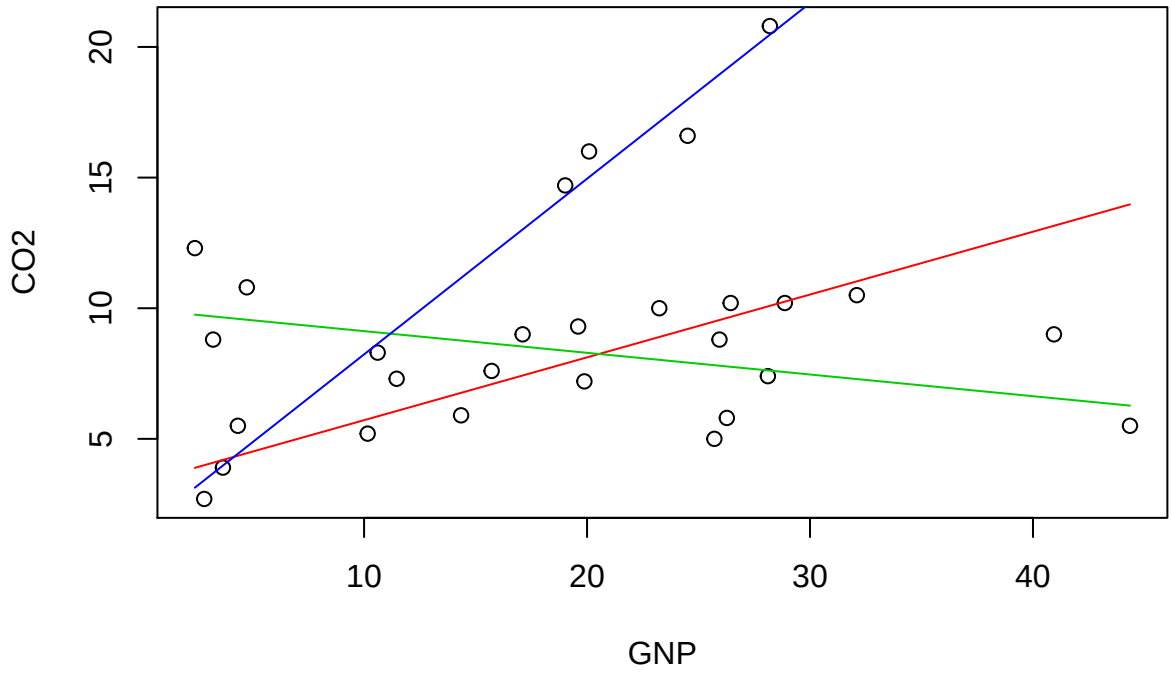


We can also fit three (or more) regressions at the same time.

```
mx4 <- mixtureReg(
  regData = CO2data,
  formulaList = list(formula(CO2 ~ GNP),
                     formula(CO2 ~ GNP),
                     formula(CO2 ~ GNP)),
  mixingProb = "Constant"
  )
```

```
## diff =  9.52474e-09
## iter =  92
## restart =  0
## log-likelihood =  -64.2842
```

```
plot(mx4, which = 1)
```



Predictor dependent mixing probabilities

Benaglia et al. (2009), proposed this modelling strategy through kernal smoothing. In the **mixtools** package this functionalities is offered in **mixtools::regmixEM.loc()**.

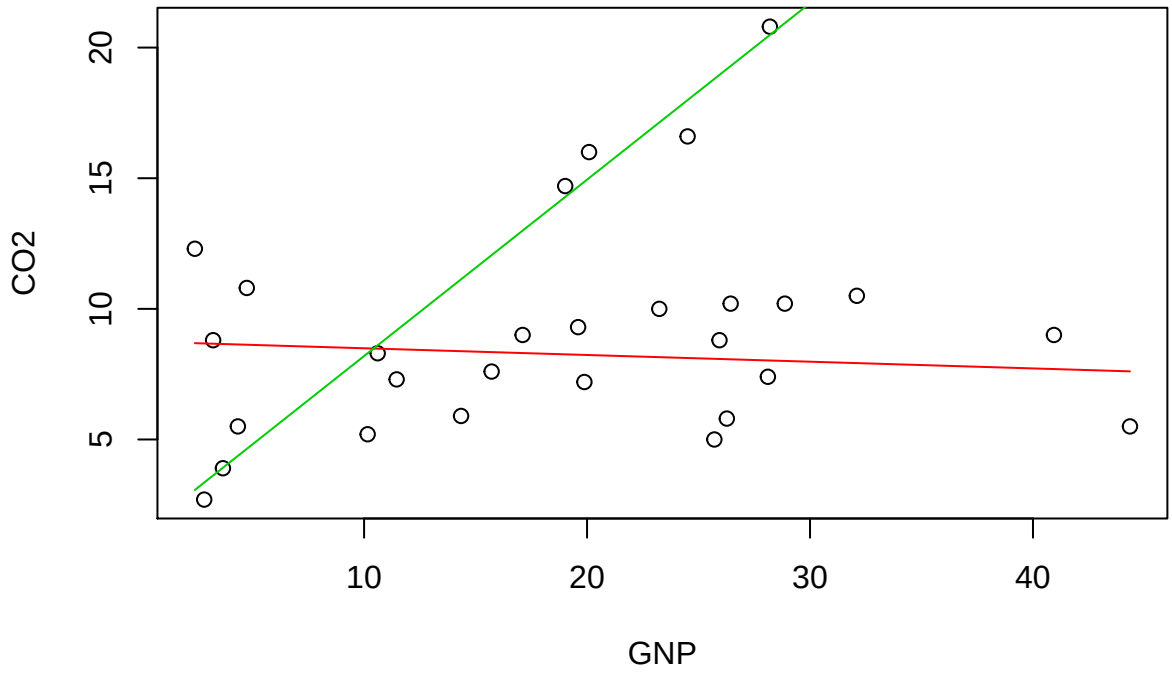
In this package we followed the same spirit but implement the smoothing based on R's own **stats::loess()**

To enable this flexibility, simply specify it in the function call.

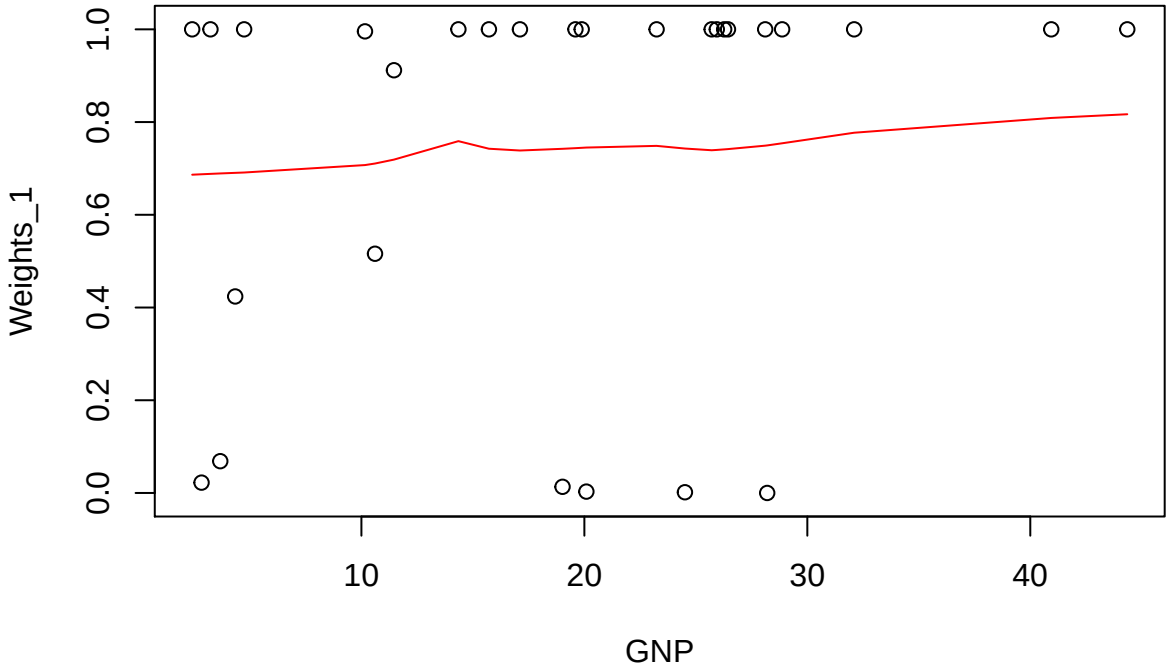
```
mx5 <- mixtureReg(
  regData = CO2data,
  formulaList = list(formula(CO2 ~ GNP),
                     formula(CO2 ~ GNP)),
  mixingProb = "loess"
  )
```

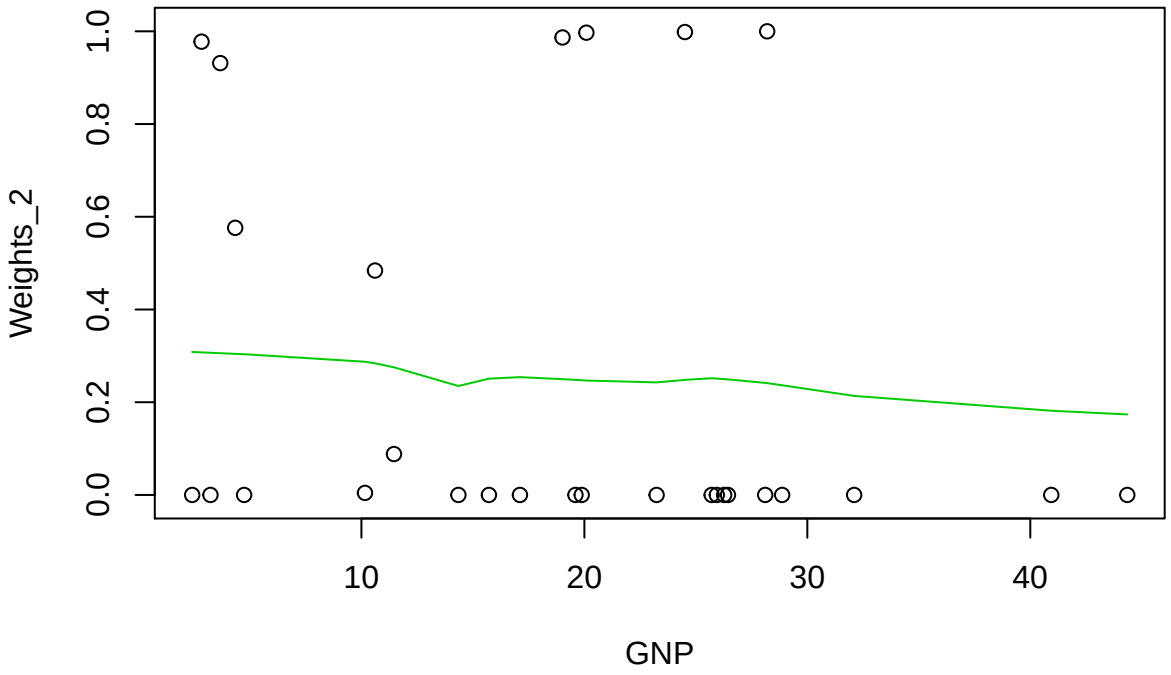
```
## diff =  4.285695e-09
## iter =  41
## restart =  0
## log-likelihood =  -66.78952
```

```
plot(mx5, which = 1)
```




```
plot(mx5, which = 2)
```



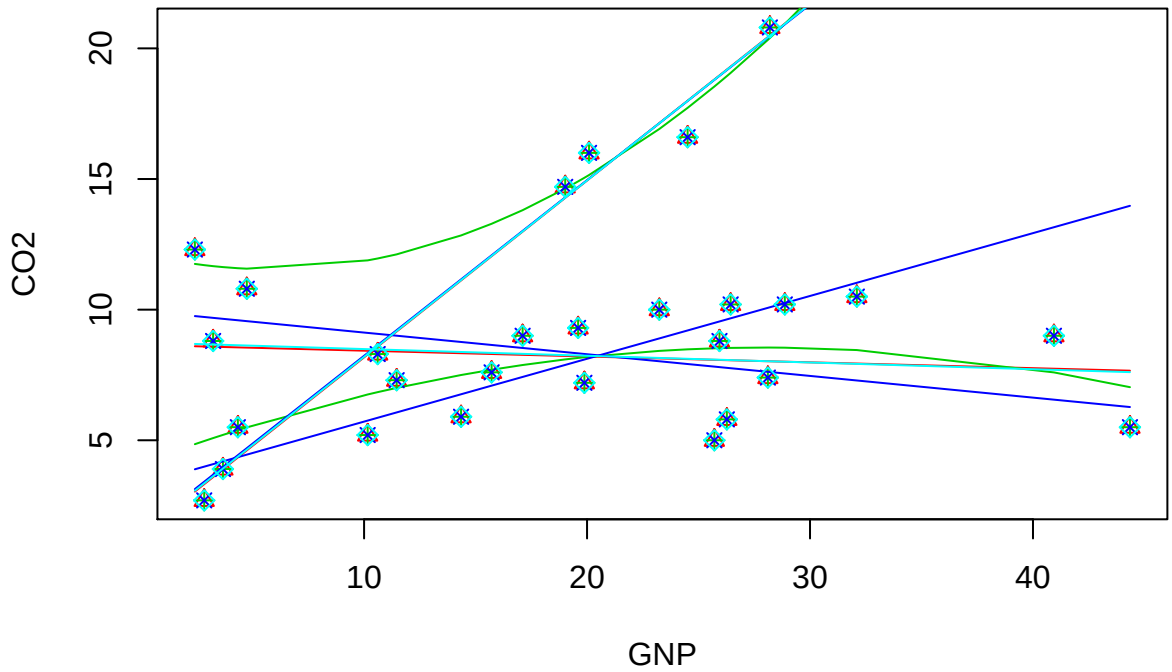


Extra plotting power

Using **plot.mixtureRegList()** function, we can compare two or more models.

For example, let's gather a few models and overlay them on top of each other.

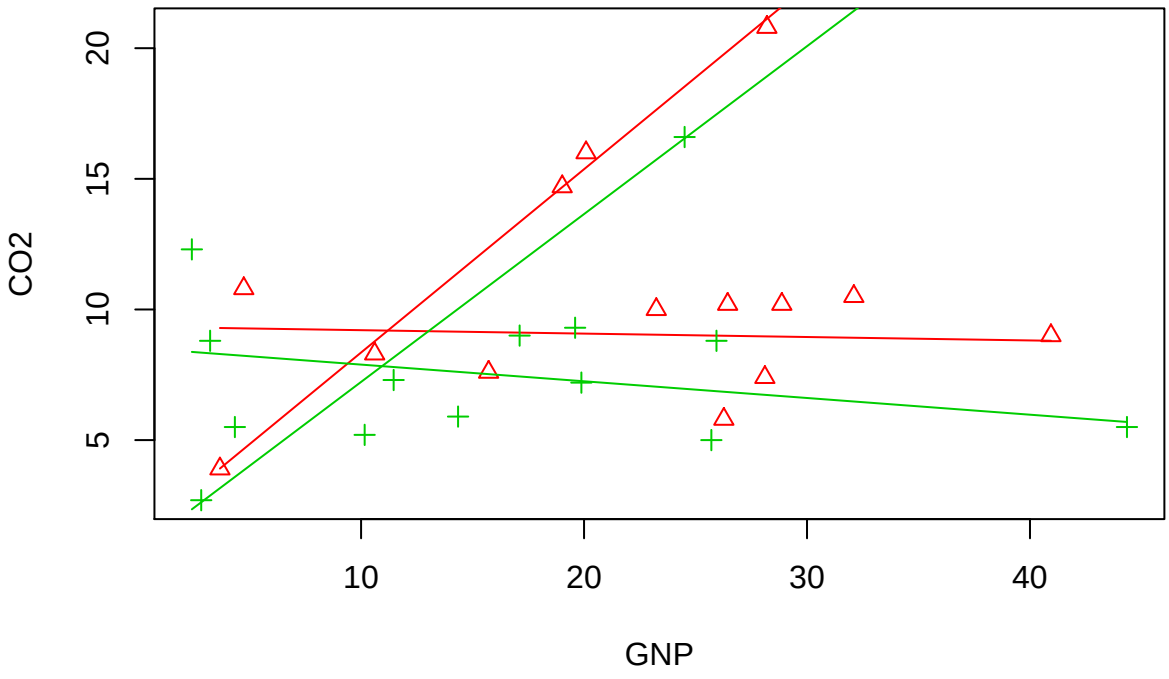
```
plot.mixtureRegList(mixtureRegList = list(mx1, mx3, mx4, mx5),
                    xName = "GNP", yName = "CO2")
```



Another example. Here we fit two models using two subsets of the data.

```
plot.mixtureRegList(
  mixtureRegList = list(
    "mx6" = mixtureReg(
      regData = CO2data[1:14, ],
      formulaList = list(formula(CO2 ~ GNP),
                         formula(CO2 ~ GNP)),
      mixingProb = "Constant"
    ),
    "mx7" = mixtureReg(
      regData = CO2data[15:28, ],
      formulaList = list(formula(CO2 ~ GNP),
                         formula(CO2 ~ GNP)),
      mixingProb = "Constant"
    )
  ),
  xName = "GNP", yName = "CO2")
```

```
## diff =  -5.623768e-09
## iter =  23
## restart =  0
## log-likelihood =  -27.88611
## diff =  0.8874124
## iter =  372
## restart =  15
## log-likelihood =  -30.49321
```



Comparison with mixtools

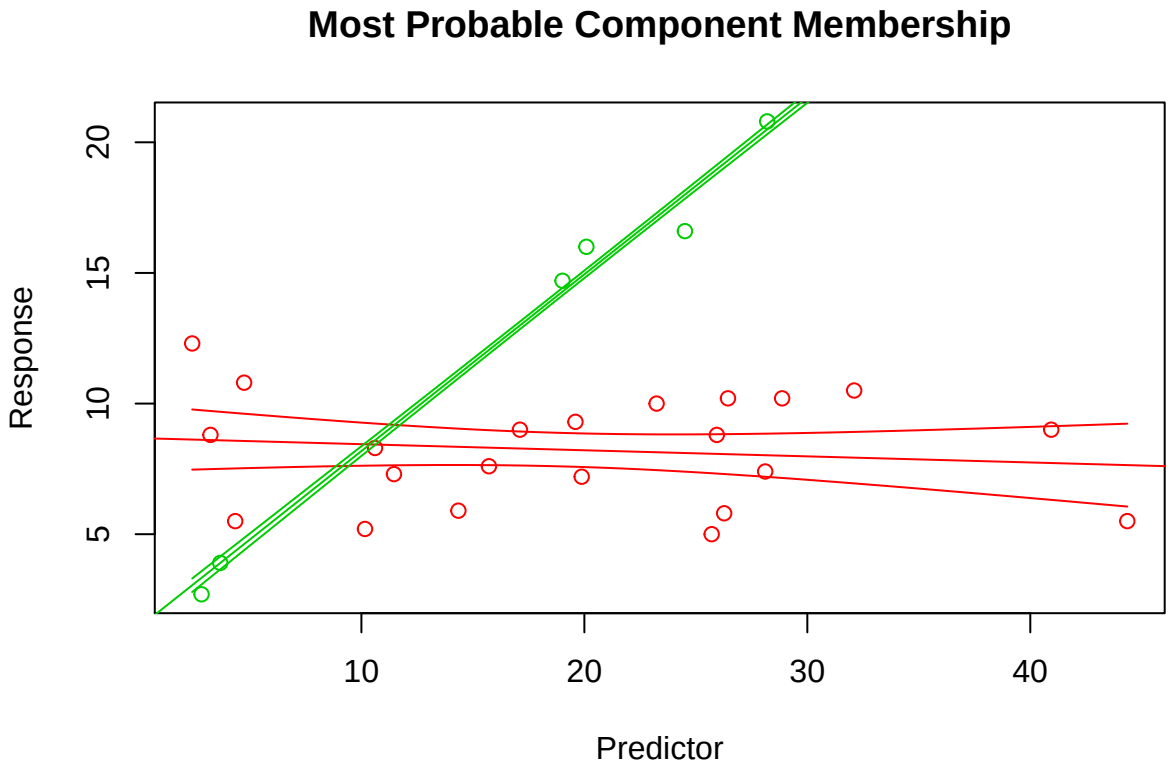
The main shortcoming of **mixtools** is that it doesn't provide easy to use options to restrict model coefficients like we do in model 2.

The following is an example from Tatiana Benaglia, Didier Chauveau, David R. Hunter, Derek Young (2009). This example produces similar results with our model 1.

```
compare1 <- mixtools::regmixEM(
  CO2data$CO2, CO2data$GNP,
  lambda = c(1/4, 3/4),
  beta = matrix(c(2, 0, 0, 1), 2, 2),
  sigma = c(1,1)
  )
```

```
## number of iterations= 18
```

```
plot(compare1, whichplots = 2)
```



References

de Veaux RD (1989). "Mixtures of Linear Regressions." Computational Statistics and Data Analysis, 8, 227-245.

Tatiana Benaglia, Didier Chauveau, David R. Hunter, Derek Young (2009). mixtools: An R Package for Analyzing Finite Mixture Models. Journal of Statistical Software, 32(6), 1-29. URL http://www.jstatsoft.org/v32/i06/.