Breakpoint model

1 Single breakpoint, single slope

1.1 Theory

Consider a response variable Y and an explanatory variable X related by:

$$Y = \beta X - \beta(X - \psi)_+ + \varepsilon$$

where $(\beta, \psi) \in \mathbb{R}^2$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, and $(x)_+ = x$ if $x > 0$ and 0 otherwise. Introduce $\Theta = (\beta, \psi, \sigma^2)$, we can express the likelihood relative to n iid observations as:

$$\mathcal{L}(\Theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(Y_i - \beta X_i + \beta(X_i - \psi)_+)^2}{2\sigma^2}\right)$$

and the log-likelihood as:

$$\ell(\Theta) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \sum_{i=1}^{n} \frac{(Y_i - \beta X_i + \beta(X_i - \psi)_+)^2}{2\sigma^2}$$

Maximizing the likelihood with respect to $\Theta_\mu = (\beta, \psi)$ is equivalent to minimizing the mean squared error:

$$\ell(\Theta_\mu) = \sum_{i=1}^{n} (Y_i - \beta X_i + \beta(X_i - \psi)_+)^2$$

which is equivalent to first minimizing w.r.t. *beta*, i.e. plug-in the OLS estimator:

$$\tilde{\beta}(\psi) = \frac{\sum_{i=1}^{n} X_i Y_i - (X_i - \psi)_+ Y_i}{\sum_{i=1}^{n} (X_i + (X_i - \psi)_+)^2} = \frac{\sum_{i=1}^{n} \mathbb{1}_{X_i \leq \psi} X_i Y_i}{\sum_{i=1}^{n} \mathbb{1}_{X_i \leq \psi} X_i^2}$$

and then minimize w.r.t. ψ:

$$\ell(\psi) = \sum_{i=1}^{n} (Y_i - \tilde{\beta}(\psi) X_i + \tilde{\beta}(\psi)(X_i - \psi)_+)^2$$

So the first 'derivative' should solve:

$$
\begin{aligned}
0 =& -2\sum_{i=1}^{n}\left[\frac{\partial\tilde{\beta}(\psi)}{\partial\psi}(X_i-(X_i-\psi)_+)-\tilde{\beta}(\psi)\frac{\partial(X_i-\psi)_+}{\partial\psi}\right](Y_i-\tilde{\beta}(\psi)X_i+\tilde{\beta}(\psi)(X_i-\psi)_+) \\
0 =& \frac{\partial\tilde{\beta}(\psi)}{\partial\psi}\sum_{i=1}^{n}(X_i-(X_i-\psi)_+)(Y_i-\tilde{\beta}(\psi)X_i+\tilde{\beta}(\psi)(X_i-\psi)_+) \\
&-\tilde{\beta}(\psi)\sum_{i=1}^{n}\frac{\partial(X_i-\psi)_+}{\partial\psi}(Y_i-\tilde{\beta}(\psi)X_i+\tilde{\beta}(\psi)(X_i-\psi)_+)
\end{aligned}
$$

Assuming that the breakpoint does not coincide with any datapoint:

$$
\begin{aligned}
0 =& -\tilde{\beta}(\psi)\sum_{i=1}^{n}\mathbb{1}_{X_i\geq\psi}(Y_i-\tilde{\beta}(\psi)\psi) \\
\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{X_i\geq\psi}Y_i =& \tilde{\beta}(\psi)\psi
\end{aligned}
$$

So the breakpoint should be such that the average post-breakpoint value equal the fitted plateau.

1.2 Example

```
library(lmbreak)

set.seed(10)
df10 <- simBreak(c(1, 100), breakpoint = c(0,2,4),
                 slope = c(1,0), sigma = 0.05)
e.lmbreak10 <- lmbreak(Y ~ 0 + bp(X, pattern = "10", start = c(2)),
                       data = df11)
model.tables(e.lmbreak10)
```

```
           X duration intercept    slope
1 0.05773562 1.940351  0.000000 1.003169
2 1.99808692 1.847123  1.946501 0.000000
3 3.84520966       NA  1.946501       NA
```

OLS:

```
XX <- df10$X - pmax(df10$X-coef(e.lmbreak10),0)
solve(t(XX) %*% XX) %*% t(XX) %*% df11$Y
```

```
         [,1]
[1,] 1.003168
```

Explicit OLS:

```
sum((df10$X < coef(e.lmbreak10))*df10$X*df10$Y)/sum((df10$X < coef(e.
   lmbreak10))*df10$X^2)
```

```
[1] 1.003169
```

Full likelihood:

```
calcLogLik <- function(theta){
  beta <- theta["beta"]
  gamma <- theta["gamma"]
  psi <- theta["psi"]
  sigma <- theta["sigma"]
  as.double(-NROW(df11)/2 * log(2*pi) - NROW(df11)/2 * log(sigma) - sum((
   df11$Y - beta * df11$X - gamma * pmax(df11$X - psi,0))^2)/(2*sigma))
}
theta <- c(beta = unname(coef(e.lmbreak11$model)["Us0"]),
           gamma = unname(coef(e.lmbreak11$model)["Us1"]),
           psi = 1.98454227,
           sigma = sigma(e.lmbreak11$model)^2)
calcLogLik(theta = theta)
logLik(e.lmbreak11$model)
```

```
[1] 162.2538
'log Lik.' 162.2768 (df=4)
```

Profile likelihood:

```
calcProfLik <- function(psi){ ## psi <- 1
  XX <- cbind(df11$X, pmax(df11$X-psi,0))
  OLS <- as.double(solve(t(XX) %*% XX) %*% t(XX) %*% df11$Y)
  sigma <- sum((df11$Y - XX %*% OLS)^2)/(NROW(df11)-3)
   - NROW(df11)/2 * log(2*pi) - NROW(df11)/2 * log(sigma) - (NROW(df11)-3)
    /2
}
calcProfLik(theta["psi"])

df.gridPsi <- data.frame(psi = seq(0.5,3.5,length.out=1000))
df.gridPsi$logLik <- sapply(df.gridPsi$psi,calcProfLik)
ggplot(df.gridPsi, aes(x=psi,y=logLik)) + geom_line()
```

```
[1] 162.2538
```

Score:

```
library(numDeriv)
jacobian(calcLogLik, theta)
jacobian(calcProfLik, theta["psi"])
```

```
           [,1]        [,2]        [,3]      [,4]
[1,] 0.02242403 0.006498193 0.007915766 -638.0514
            [,1]
[1,] 0.001578588
```

Hessian:

```
Hall <- hessian(calcLogLik, theta)
Iall <- solve(-Hall)

Iall[3,3] - Iall[3,1:2] %*% solve(Iall[1:2,1:2]) %*% Iall[1:2,3]

Hpsi <- hessian(calcProfLik, theta["psi"])
Ipsi <- solve(-Hpsi)

Ipsi/(Iall[3,3] - Iall[3,1:2] %*% solve(Iall[1:2,1:2]) %*% Iall[1:2,3])
```

```
             [,1]
[1,] 5.740945e-05
         [,1]
[1,] 4.928297
```

2 Single breakpoint

2.1 Theory

Consider a response variable Y and an explanatory variable X related by:

$$Y = \beta X + \gamma (X - \psi)_+ + \varepsilon$$

where $(\beta, \gamma, \psi) \in \mathbb{R}^3$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, and $(x)_+ = x$ if $x > 0$ and 0 otherwise. Introduce $\Theta = (\beta, \gamma, \psi, \sigma^2)$, we can express the likelihood relative to n iid observations as:

$$\mathcal{L}(\Theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(Y_i - \beta X_i - \gamma (X_i - \psi)_+)^2}{2\sigma^2} \right)$$

and the log-likelihood as:

$$\ell(\Theta) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \sum_{i=1}^{n} \frac{\left(Y_i - \beta X_i - \gamma (X_i - \psi)_+\right)^2}{2\sigma^2}$$

Maximizing the likelihood with respect to $\Theta_\mu = (\beta, \gamma, \psi)$ is equivalent to minimizing the mean squared error:

$$\ell(\Theta_\mu) = \sum_{i=1}^{n} (Y_i - \beta X_i - \gamma (X_i - \psi)_+)^2$$

which is equivalent to first minimizing w.r.t. (β, γ), i.e. plug-in the OLS estimator:

$$\begin{aligned}
(\widehat{\beta}, \widehat{\gamma}) &= \left(\begin{bmatrix} X \\ (X-\psi)_+ \end{bmatrix} \begin{bmatrix} X & (X-\psi)_+ \end{bmatrix} \right)^{-1} \begin{bmatrix} X \\ (X-\psi)_+ \end{bmatrix} Y \\
&= \begin{bmatrix} X^\intercal X & X^\intercal (X-\psi)_+ \\ X^\intercal (X-\psi)_+ & (X-\psi)_+^\intercal (X-\psi)_+ \end{bmatrix}^{-1} \begin{bmatrix} X^\intercal Y \\ (X-\psi)_+^\intercal Y \end{bmatrix} \\
&= \frac{\begin{bmatrix} (X-\psi)_+^\intercal (X-\psi)_+ & -X^\intercal (X-\psi)_+ \\ -X^\intercal (X-\psi)_+ & X^\intercal X \end{bmatrix} \begin{bmatrix} X^\intercal Y \\ (X-\psi)_+^\intercal Y \end{bmatrix}}{X^\intercal X (X-\psi)_+^\intercal (X-\psi)_+ - X^\intercal (X-\psi)_+ X^\intercal (X-\psi)_+}
\end{aligned}$$

where $Y = (Y_1, \ldots, Y_n)$ and $X = (X_1, \ldots, X_n)$. We therefore obtain:

$$\begin{aligned}
\tilde{\beta}(\psi) &= \frac{(X-\psi)_+^\intercal (X-\psi)_+ X^\intercal Y - X^\intercal (X-\psi)_+ (X-\psi)_+^\intercal Y}{X^\intercal X (X-\psi)_+^\intercal (X-\psi)_+ - X^\intercal (X-\psi)_+ X^\intercal (X-\psi)_+} \\
\tilde{\gamma}(\psi) &= \frac{X^\intercal X (X-\psi)_+^\intercal Y - X^\intercal (X-\psi)_+ X^\intercal Y}{X^\intercal X (X-\psi)_+^\intercal (X-\psi)_+ - X^\intercal (X-\psi)_+ X^\intercal (X-\psi)_+}
\end{aligned}$$

and then minimize w.r.t. ψ:

$$\ell(\psi) = \sum_{i=1}^{n} (Y_i - \tilde{\beta}(\psi) X_i - \tilde{\gamma}(\psi)(X_i - \psi)_+)^2$$

Its first derivative is:

$$0 = -2 \sum_{i=1}^{n} \left[\frac{\partial \tilde{\beta}(\psi)}{\partial \psi} X_i + \frac{\partial \tilde{\gamma}(\psi)}{\partial \psi} (X_i - \psi)_+ + \tilde{\gamma}(\psi) \frac{\partial (X_i - \psi)_+}{\partial \psi} \right] (Y_i - \tilde{\beta}(\psi) X_i - \tilde{\gamma}(\psi)(X_i - \psi)_+)$$

2.2 Example

```
library(lmbreak)

set.seed(10)
df11 <- simBreak(c(1, 100), breakpoint = c(0,2,4),
                 slope = c(1,0), sigma = 0.05)
e.lmbreak11 <- lmbreak(Y ~ 0 + bp(X, pattern = "11", start = c(2)),
                       data = df11)
model.tables(e.lmbreak11)
```

```
           X duration intercept      slope
1 0.05773562 1.926807  0.000000 1.00316914
2 1.98454227 1.860667  1.932913 0.01677379
3 3.84520966       NA  1.964123         NA
```

OLS:

```
XX <- cbind(df11$X, pmax(df11$X-coef(e.lmbreak11),0))
solve(t(XX) %*% XX) %*% t(XX) %*% df11$Y
```

```
           [,1]
[1,]  1.0031692
[2,] -0.9863952
```

Explicit OLS:

```
(crossprod(XX[,2]) * crossprod(XX[,1],df11$Y) - crossprod(XX[,1],XX[,2]) *
    crossprod(XX[,2],df11$Y)) / (crossprod(XX[,1]) * crossprod(XX[,2]) -
   crossprod(XX[,1],XX[,2])^2)
(crossprod(XX[,1]) * crossprod(XX[,2],df11$Y) - crossprod(XX[,1],XX[,2]) *
    crossprod(XX[,1],df11$Y)) / (crossprod(XX[,1]) * crossprod(XX[,2]) -
   crossprod(XX[,1],XX[,2])^2)
```

```
         [,1]
[1,] 1.003169
           [,1]
[1,] -0.9863952
```

Full likelihood:

```
calcLogLik <- function(theta){
  beta <- theta["beta"]
  gamma <- theta["gamma"]
  psi <- theta["psi"]
  sigma <- theta["sigma"]
```

```
  as.double(-NROW(df11)/2 * log(2*pi) - NROW(df11)/2 * log(sigma) - sum((
    df11$Y - beta * df11$X - gamma * pmax(df11$X - psi,0))^2)/(2*sigma))
}
theta <- c(beta = unname(coef(e.lmbreak11$model)["Us0"]),
           gamma = unname(coef(e.lmbreak11$model)["Us1"]),
           psi = 1.98454227,
           sigma = sigma(e.lmbreak11$model)^2)
calcLogLik(theta = theta)
logLik(e.lmbreak11$model)
```

```
[1] 162.2538
'log Lik.' 162.2768 (df=4)
```

Profile likelihood:

```
calcProfLik <- function(psi){ ## psi <- 1
  XX <- cbind(df11$X, pmax(df11$X-psi,0))
  OLS <- as.double(solve(t(XX) %*% XX) %*% t(XX) %*% df11$Y)
  sigma <- sum((df11$Y - XX %*% OLS)^2)/(NROW(df11)-3)
   - NROW(df11)/2 * log(2*pi) - NROW(df11)/2 * log(sigma) - (NROW(df11)-3)
    /2
}
calcProfLik(theta["psi"])

df.gridPsi <- data.frame(psi = seq(0.5,3.5,length.out=1000))
df.gridPsi$logLik <- sapply(df.gridPsi$psi,calcProfLik)
ggplot(df.gridPsi, aes(x=psi,y=logLik)) + geom_line()
```

```
[1] 162.2538
```

Score:

```
library(numDeriv)
jacobian(calcLogLik, theta)
jacobian(calcProfLik, theta["psi"])
```

```
           [,1]        [,2]        [,3]      [,4]
[1,] 0.02242403 0.006498193 0.007915766 -638.0514
            [,1]
[1,] 0.001578588
```

Hessian:

```
Hall <- hessian(calcLogLik, theta)
Iall <- solve(-Hall)
```

```r
Iall[3,3] - Iall[3,1:2] %*% solve(Iall[1:2,1:2]) %*% Iall[1:2,3]

Hpsi <- hessian(calcProfLik, theta["psi"])
Ipsi <- solve(-Hpsi)

Ipsi/(Iall[3,3] - Iall[3,1:2] %*% solve(Iall[1:2,1:2]) %*% Iall[1:2,3])
```

```
             [,1]
[1,] 5.740945e-05
         [,1]
[1,] 4.928297
```

3 Multiple breakpoints

We now consider the more general case where:

$$Y = \beta X(\psi) + \varepsilon$$

where β is a vector of coefficients and $X(\psi)$ the design matrix depending on a vector of breakpoint ψ. Similarly to the previous derivations we need to minimize the mean square loss:

$$\sum_{i=1}^{n} (Y_i - \beta X_i(\psi))^2$$

with respect to β and ψ. For given ψ the coefficient β minimizing this loss are given by the OLS estimator:

$$\widehat{\beta} = (X^{\intercal}(\psi)X(\psi))^{-1}X^{\intercal}(\psi)Y$$

4 Proximal gradient method

One difficulty is that this objective function is not differientiable in ψ at $(X_i)_{i=1}^n$.

$\ell(\Theta_\mu)$ might not be strictly convex but it is convex. So we can try applying a proximal gradient algorithm. This means updating the estimate by:

$$\begin{aligned}\Theta_{\mu,k+1} &= \operatorname{prox}_{\alpha_k \ell}(\Theta_{\mu,k}) = \underset{\Theta_\mu \in \mathbb{R}^2}{\arg\min} \left(\ell(\Theta_\mu) + \frac{1}{2\alpha_k} ||\Theta_\mu - \Theta_{\mu,k}||^2 \right) \\ &= \underset{\Theta_\mu \in \mathbb{R}^3}{\arg\min} \left(\sum_{i=1} (Y_i - \beta X_i - \gamma(X_i - \psi)_+)^2 + \frac{(\beta-\beta_k)^2 + (\gamma-\gamma_k)^2 + (\psi-\psi_k)^2}{2\alpha_k} \right) \quad = \underset{\psi \in \mathbb{R}}{\arg\min} \left((I \right.\end{aligned}$$

where α_k is a pre-defined stricly positive real value.