Advanced Statistical Programming with R

Some more details on location-scale regression

Hannes Riebl

April 21, 2021

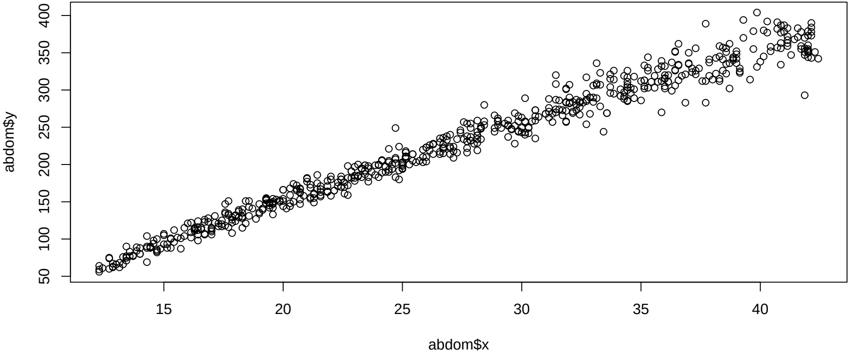
Questions?

Create your GitLab account

Please log in to https://gitlab.gwdg.de **now!**

The abdom example

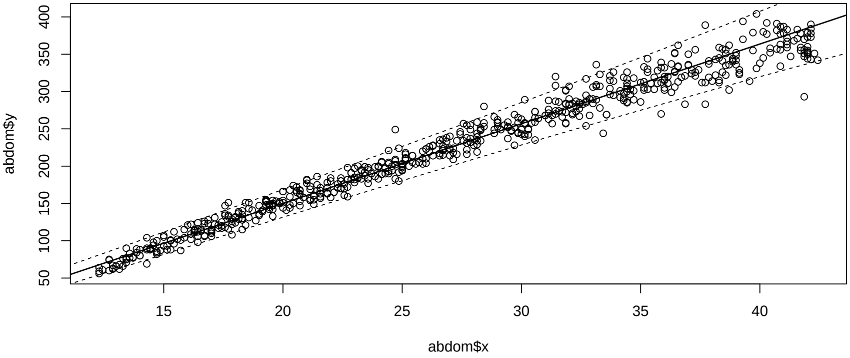
```
data(abdom, package = "gamlss.data")
plot(abdom$x, abdom$y)
```



```r
library(lslm)
mod <- lslm(location = y ~ x, scale = ~x, data = abdom)
summary(mod)
```

```
##
## Call:
## lslm(location = y ~ x, scale = ~x, data = abdom)
##
## Pearson residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -3.74000 -0.66540 -0.04960 -0.01809  0.64240  4.19900
##
## Location coefficients (identity link function):
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -63.47252    1.46549  -43.31   <2e-16 ***
## x            10.67805    0.06388  167.15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Scale coefficients (log link function):
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.386819   0.096714   14.34   <2e-16 ***
## x           0.042992   0.003388   12.69   <2e-16 ***
```

```r
plot(abdom$x, abdom$y)
abline(coef(mod, "location")[1], coef(mod, "location")[2], lwd = 1.5)
curve(f(x, "upper"), from = 5, to = 50, add = TRUE, lty = 2)
curve(f(x, "lower"), from = 5, to = 50, add = TRUE, lty = 2)
```



Location-scale regression

The location-scale regression model is defined as

$$y_i \overset{\text{ind.}}{\sim} \mathcal{N}(\boldsymbol{x}_i'\boldsymbol{\beta}, (\exp(\boldsymbol{z}_i'\boldsymbol{\gamma}))^2),$$

where

- $i = 1, \dots, n$ is the observation index,
- y_i is the response variable,
- $\boldsymbol{x}_i$ is the vector of explanatory variables **for the mean**,
- $\boldsymbol{\beta}$ is the vector of regression coefficients **for the mean**,
- $\boldsymbol{z}_i'$ is the vector of explanatory variables **for the standard deviation**, and
- $\boldsymbol{\gamma}$ is the vector of regression coefficients **for the standard deviation**.

Maximum likelihood estimation

How can we estimate the full parameter vector of the location-scale regression model

$$\boldsymbol{\vartheta} = \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix}$$

accurately, reliably, and efficiently?

Newton-Raphson / Fisher Scoring:

- In ML estimation, we are seeking the root of the score function $s(\vartheta)$.
- Basic idea of Newton-Raphson optimisation: Iteratively approximate $s(\vartheta)$ by a Taylor expansion (i.e. by a simple polynomial) and find the root of this approximation.
- First order Taylor approximation:

$$s(\vartheta) \approx s(\tilde{\vartheta}) - F(\tilde{\vartheta})(\vartheta - \tilde{\vartheta}).$$

- Setting the Taylor approximation to zero yields

$$\hat{\vartheta} = \tilde{\vartheta} + \frac{s(\tilde{\vartheta})}{F(\tilde{\vartheta})}.$$

- Newton-Raphson algorithm:
  - Determine a suitable starting value $\hat{\vartheta}^{(0)}$.
  - Update
$$\hat{\vartheta}^{(k+1)} = \hat{\vartheta}^{(k)} + \frac{s(\hat{\vartheta})^{(k)}}{F(\hat{\vartheta})^{(k)}}$$
    until the change in the estimate becomes small.

- In statistics, the negative second derivative (i.e. the Fisher information) is often replaced by the expected Fisher information $F^*(\vartheta)$ (Fisher scoring).
  - $F^*(\vartheta)$ often has a simpler structure since terms with expectation zero cancel.
  - The solution should not be affected too much since

$$F(\vartheta) \approx F^*(\vartheta)$$

    for large samples (and since the same score function is used).
  - In certain classes of models, one can show that the expected Fisher information is ensured to be positive which is not always for the case the Fisher information $F(\vartheta)$.

Fisher scoring with a parameter **vector**

In the location-scale regression model, ϑ is a parameter **vector**, and hence the update step of the Fisher scoring algorithm is

$$\hat{\boldsymbol{\vartheta}}^{(k+1)} = \hat{\boldsymbol{\vartheta}}^{(k)} + \left(F^*\left(\hat{\boldsymbol{\vartheta}}^{(k)}\right)\right)^{-1} s\left(\hat{\boldsymbol{\vartheta}}^{(k)}\right).$$

Blockwise parameter updates

Until convergence, repeat:

- Update $\hat{\gamma}$.
- Update $\hat{\beta}$.

Why?

- The cross-covariance matrix of $s(\beta)$ and $s(\gamma)$ is $\mathbf{0}$.
- There is an analytic ML estimator for β given γ.

Estimating β given γ

Given $\boldsymbol{\gamma}$, we know $\sigma_i = \exp(\boldsymbol{z}_i'\boldsymbol{\gamma})$ for all $i = 1, \ldots, n$. Then we can estimate β with WLS:

$$\hat{\beta}^{\text{WLS}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\boldsymbol{y},$$

where

$$\mathbf{W} = \begin{bmatrix} \sigma_1^{-2} & 0 & \ldots & 0 \\ 0 & \sigma_2^{-2} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \ldots & 0 & \sigma_n^{-2} \end{bmatrix}.$$

The WLS estimator is equivalent to the ML estimator.

Fisher scoring for γ

We need the score $s(\gamma)$ and the expected Fisher information $F^*(\gamma)$ with respect to the parameter vector $\boldsymbol{\gamma}$...

Summing up the iterative ML estimation algorithm

Until convergence, repeat:

- Update $\hat{\gamma}$ with Fisher scoring, keeping $\hat{\beta}$ fixed at its current value.
- Update $\hat{\beta}$ with WLS, assuming the current value of $\hat{\gamma}$ is the true value.

Initializing $\hat{\beta}$ with OLS

We can initialize $\hat{\beta}$ with OLS:

$$\hat{\boldsymbol{\beta}}^{\mathsf{OLS}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y}.$$

$\hat{\boldsymbol{\beta}}^{\mathsf{OLS}}$ is unbiased:

$$\mathrm{E}\left(\hat{\boldsymbol{\beta}}^{\mathsf{OLS}}\right) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{E}(\boldsymbol{y}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}.$$

Under mild regularity assumptions, $\hat{\boldsymbol{\beta}}^{\mathsf{OLS}}$ is also consistent.

However, $\hat{\boldsymbol{\beta}}^{\mathsf{OLS}}$ is no longer BLUE, because the homoskedasticity assumption of the Gauss-Markov theorem is violated.

Initializing $\hat{\gamma}$ with OLS

$$\sigma_i = \exp(\boldsymbol{z}_i'\boldsymbol{\gamma}) \iff \log \sigma_i = \boldsymbol{z}_i'\boldsymbol{\gamma}.$$

Hence, we are able to estimate γ with OLS if we can come up with a good estimator for $\log \sigma_i$.

The natural estimator for σ_i, $\hat{\sigma}_i = |y_i - \mu_i|$, is unbiased, but due to Jensen's inequality,

$$\mathrm{E}(\log \hat{\sigma}_i) < \log \mathrm{E}(\hat{\sigma}_i) = \log \sigma_i.$$

Deriving an unbiased estimator for $\log \sigma_i$

Is there a simple bias correction term?

Metropolis-Hastings-Algorithm:

- Define a starting value $\vartheta^{(0)}$.
- For $m = 1, \ldots, M$ generate a proposal ϑ^* from a density

$$h(\vartheta^* | \vartheta^{(m-1)})$$

depending on the previous value $\vartheta^{(m-1)}$.

- Accept it with probability

$$\alpha(\vartheta^* | \vartheta^{(m-1)}) = \min\left\{1, \frac{f(\vartheta^* | \boldsymbol{x})}{f(\vartheta^{(m-1)} | \boldsymbol{x})} \frac{h(\vartheta^{(m-1)} | \vartheta^*)}{h(\vartheta^* | \vartheta^{(m-1)})}\right\}$$

If ϑ^* is accepted, set

$$\vartheta^{(m)} = \vartheta^*.$$

Otherwise set

$$\vartheta^{(m)} = \vartheta^{(m-1)}.$$

- Remarks:
  - The normalising constant in $f(\vartheta|\boldsymbol{x})$ cancels in the acceptance probability and is therefore not required.
  - The proposal density $h(\vartheta^*|\vartheta^{(m-1)})$ can be any density with the same support as the posterior $f(\vartheta|\boldsymbol{x})$.
  - If the proposal density is close to the posterior, the acceptance probability will always be close to 1.

- If the parameter vector $\boldsymbol{\vartheta}$ is of higher dimension, additional flexibility can be achieved by decomposing $\boldsymbol{\vartheta}$ as
$$\boldsymbol{\vartheta} = (\boldsymbol{\vartheta}_1', \ldots, \boldsymbol{\vartheta}_j', \ldots, \boldsymbol{\vartheta}_k')'.$$
Then, separate proposals for each subvector $\boldsymbol{\vartheta}_j$ are defined, i.e. a proposal $\boldsymbol{\vartheta}_j^*$ is generated from
$$h_j(\boldsymbol{\vartheta}_j^* | \boldsymbol{\vartheta}_1^{(m)}, \ldots, \boldsymbol{\vartheta}_{j-1}^{(m)}, \boldsymbol{\vartheta}_j^{(m-1)}, \boldsymbol{\vartheta}_{j+1}^{(m-1)}, \ldots, \boldsymbol{\vartheta}_k^{(m-1)})$$

and accepted with probability $\alpha(\boldsymbol{\vartheta}_j^*|\boldsymbol{\vartheta}_j^{(m-1)})$ defined as

$$\min\left\{1, \frac{f(\boldsymbol{\vartheta}_j^*|\boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x})}{f(\boldsymbol{\vartheta}_j^{(m-1)}|\boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x})} \frac{h_j(\boldsymbol{\vartheta}_j^{(m-1)}|\boldsymbol{\vartheta}_1^{(m)}, \ldots, \boldsymbol{\vartheta}_{j-1}^{(m)}, \boldsymbol{\vartheta}_j^*, \boldsymbol{\vartheta}_{j+1}^{(m-1)}, \ldots, \boldsymbol{\vartheta}_k^{(m-1)})}{h_j(\boldsymbol{\vartheta}_j^*|\boldsymbol{\vartheta}_1^{(m)}, \ldots, \boldsymbol{\vartheta}_{j-1}^{(m)}, \boldsymbol{\vartheta}_j^{(m-1)}, \boldsymbol{\vartheta}_{j+1}^{(m-1)}, \ldots, \boldsymbol{\vartheta}_k^{(m-1)})}\right\}$$

where

$$\boldsymbol{\vartheta}_{-j}^{(m)} = (\boldsymbol{\vartheta}_1^{(m)}, \ldots, \boldsymbol{\vartheta}_{j-1}^{(m)}, \boldsymbol{\vartheta}_{j+1}^{(m-1)}, \ldots, \boldsymbol{\vartheta}_k^{(m-1)})'$$

and

$$f(\boldsymbol{\vartheta}_j|\boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x})$$

is the full conditional of $\boldsymbol{\vartheta}_j$ (given all other parameters and the data $\boldsymbol{x}$).

- The full conditional $f(\boldsymbol{\vartheta}_j|\boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x})$ is proportional to the complete posterior, i.e. it can be determined by considering only those factors in $f(\boldsymbol{\vartheta}|\boldsymbol{x})$ that depend on $\boldsymbol{\vartheta}_j$.

Choice of the Proposal Density:

- If the full conditional is a known distribution, we can directly generate a proposal from this distribution that will be accepted with probability 1 (this is called a Gibbs update).
- Random walk proposal:

$$\vartheta_j^* = \vartheta_j^{(m-1)} + u_j, \quad u_j \sim \mathrm{N}(0, \tau^2)$$

where τ^2 is a tuning parameter that determines the acceptance probability.

- Locally quadratic approximation of the log-full conditional: Determine the mode

$$\boldsymbol{m}_j = \arg\max_{\boldsymbol{\vartheta}_j} \log\left(f(\boldsymbol{\vartheta}_j|\boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x})\right)$$

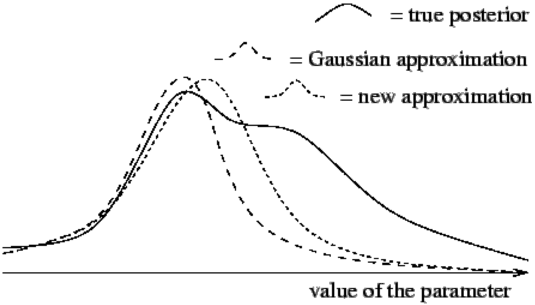
and the curvature at the mode

$$\boldsymbol{F}_j = -\frac{\partial^2}{\partial\boldsymbol{\vartheta}_j \partial\boldsymbol{\vartheta}_j'} \log\left(f(\boldsymbol{\vartheta}_j|\boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x})\right)\Bigg|_{\boldsymbol{\vartheta}_j = \boldsymbol{m}_j}$$

and propose the new state from

$$\mathrm{N}(\boldsymbol{m}_j, \boldsymbol{F}_j^{-1}).$$

Approximation of the posterior



Taken from: http://users.ics.aalto.fi/harri/thesis/valpola_thesis/node50.html

Proposal density in the `lslm` package

The `lslm` package uses the proposal density

$$\mathcal{N}\left(\boldsymbol{m}_j, \left(F^*\left(\boldsymbol{\vartheta}_j^{(m-1)} \,\middle|\, \boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x}\right)\right)^{-1}\right),$$

where $\boldsymbol{m}_j = \boldsymbol{\vartheta}_j^{(m-1)} + \left(F^*\left(\boldsymbol{\vartheta}_j^{(m-1)} \,\middle|\, \boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x}\right)\right)^{-1} s\left(\boldsymbol{\vartheta}_j^{(m-1)} \,\middle|\, \boldsymbol{\vartheta}_{-j}^{(m)}, \boldsymbol{x}\right)$.

Note that the `lslm` package assumes flat priors for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, and hence, s and F^* are simply the score and the expected Fisher information of the log-likelihood as before.

Next steps

1. Read the code of the `lslm` package.
2. Read the book "R Packages" by Hadley Wickham and Jenny Bryan.
3. Read the book "Advanced R" by Hadley Wickham.
4. Get in touch with your topic supervisors.

Questions?