CE043 - GAMLSS

Estimação e Inferência I: Os Algoritmos RS e CG

Silva, J.P; Taconeli, C.A.

03/08/2020

O exemplo `fish species`

Os dados se referem ao número de diferentes espécies de peixe registradas em 70 lagos ao redor do mundo. A variável resposta `fish` é uma contagem, e há apenas uma variável explicativa, a área do lago, denotata por `lake`. Os dados estão no objeto `species` do pacote `gamlss.data` que é carregado automaticamente com o pacote `gamlss`. Estes dados serão revisitados posteriomente na disicplina.

```
library(gamlss)
```

```
## Loading required package: splines
```

```
## Loading required package: gamlss.data
```

```
##
## Attaching package: 'gamlss.data'
```

```
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
## Loading required package: gamlss.dist
```

```
## Loading required package: MASS
```

```
## Loading required package: nlme
```

```
## Loading required package: parallel
```

```
##  **********   GAMLSS Version 5.1-7  **********
```

```
## For more on GAMLSS look at http://www.gamlss.com/
```

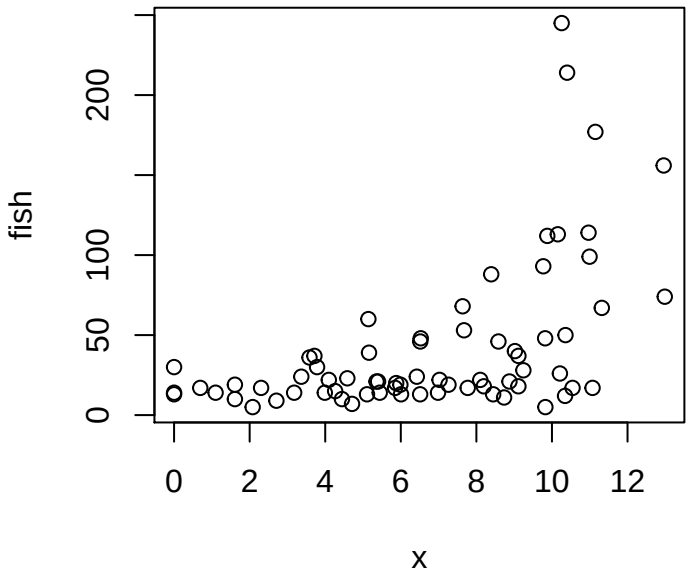
```
## Type gamlssNews() to see new features/changes/bug fixes.
```

```
data(species)
# help("species")
# help(package = "gamlss.data")
#View(species)
summary(species)
```

```
##       fish             lake
##  Min.   :  5.00   Min.   :     1.0
##  1st Qu.: 14.00   1st Qu.:    74.5
##  Median : 21.50   Median :   882.0
##  Mean   : 41.74   Mean   : 22428.8
##  3rd Qu.: 47.50   3rd Qu.: 15710.0
##  Max.   :245.00   Max.   :436000.0
```

Os dados são plotados a seguir, após transformação da variável x.

```
species <- transform(species, x=log(lake))
plot(fish~x,data=species)
```



Rigby et al. (2008) analisaram estes dados e identificaram as seguintes questões a serem respondidas.

- Como a média da variável resposta depende de x?
- A variável resposta é Poisson com superdispersão?
- Como a variância da resposta depende da média?
- Qual é a distribuição condicional da variável resposta dado x?
- Os parâmetros de escala e forma da distribuição da variável resposta dependem de x?

Ajuste Poisson

A função de probabilidade da distribuição Poisson, denotada por `PO`(μ) é dada por

$$P(Y = y|\mu) = \frac{e^{-\mu}\mu^y}{y!}, \quad \mu > 0, \quad y = 0, 1, 2, \ldots$$

- Um único parâmetro governa a média e variância, que são dados por $E(Y) = \mu$ e $Var(Y) = \mu$.

O ajuste da distribuição Poisson é feito a seguir, usando o método de estimação padrão, ou seja, `method=RS()`.

```
PO() #verifica a família e função de ligação padrão
```

```
##
## GAMLSS Family: PO Poisson
## Link function for mu    : log
```

```
#?PO #descrição da função PO() e funções associadas
m0 <- gamlss(fish~x, family=PO, data=species)
```

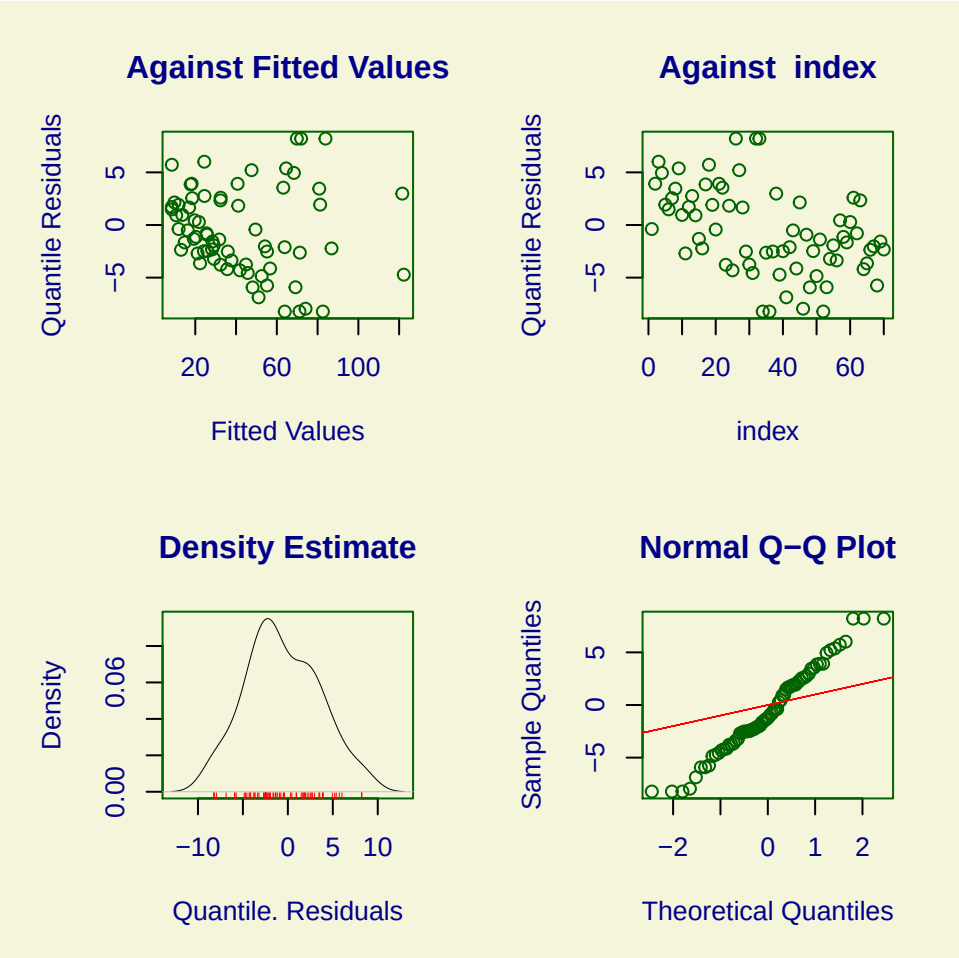
```
## GAMLSS-RS iteration 1: Global Deviance = 1896.156
## GAMLSS-RS iteration 2: Global Deviance = 1896.156
```

```
m0
```

```
##
## Family:  c("PO", "Poisson")
## Fitting method: RS()
##
## Call:  gamlss(formula = fish ~ x, family = PO, data = species)
##
## Mu Coefficients:
## (Intercept)            x
##      2.1395       0.2054
##
##  Degrees of Freedom for the fit: 2 Residual Deg. of Freedom   68
## Global Deviance:     1896.16
##             AIC:     1900.16
##             SBC:     1904.65
```

Para verificarmos a qualidade do ajuste vamos plotar os resíduos.

```
plot(m0)
```




```
## ************************************************************
```

```
##   Summary of the Randomised Quantile Residuals
##                            mean   =  -0.5842409
##                        variance   =  16.4841
##                coef. of skewness  =  0.181615
##                coef. of kurtosis  =  2.433833
## Filliben correlation coefficient  =  0.9920501
## ****************************************************************
```

Como esperado, a distribuição Poisson não apresenta um bom ajuste aos dados, o que fica evidenciado pelo comportamento não aleatório dos resíduos.

Ajuste Poisson Gaussiana Inversa

Podemos modelar os dados por meio de diferentes distribuições discretas. Nesta ilustração, como alternativa à distribuição Poisson, vamos usar a distribuição Poisson Gaussiana Inversa, denotada por `PIG`$(\mu;\sigma)$. A sua função de probabilidade é dada por

$$P(Y=y|\mu,\sigma)=\left(\frac{2\alpha}{\pi}\right)^{1/2}\frac{\mu^y e^{1/\alpha}K_{y-1/\alpha}(\alpha)}{y!(\alpha\sigma)^y},\quad y=0,1,2,\ldots$$

em que $\alpha^2=\sigma^{-2}+2\mu\sigma^{-1}$, $\mu>0$ e $\sigma>0$, e $K_\lambda(t)=\int_0^\infty \frac{1}{2}x^{\lambda-1}\exp\left\{-\frac{1}{2}t(x+x^{-1})\right\}dx$ é a função Bessel modificada de terceiro tipo.

- Para esta distribuição, temos $E(Y)=\mu$ e $Var(Y)=\mu+\sigma\mu^2$.
- A distribuição Poisson, `PO`(μ), é um caso limite da `PIG`(μ,σ) quando $\sigma\to 0$.
- Uma característica da distribuição `PIG` é que os parâmetros μ e σ não são ortogonais. Note que $\sigma=\left[(\mu^2+\alpha^2)^{0.5}-\mu\right]^{-1}$.

Vamos demonstrar o desempenho dos algoritmos RS e CG usando a distribuição `PIG` e modelando tanto o parâmetro de média μ como o parâmetro de dispersão σ como funções de log(`lake`).

```
PIG()#verifica a família e função de ligação padrão
```

```
##
## GAMLSS Family: PIG Poisson.Inverse.Gaussian
## Link function for mu    : log
## Link function for sigma: log
```

```
#?PIG #descrição da função PIG() e funções associadas
```

Algoritmo RS

```
m1 <- gamlss(fish~x, sigma.fo=~x, family=PIG, data=species)
```

```
## GAMLSS-RS iteration 1: Global Deviance = 617.3683
## GAMLSS-RS iteration 2: Global Deviance = 609.6741
## GAMLSS-RS iteration 3: Global Deviance = 608.8733
## GAMLSS-RS iteration 4: Global Deviance = 608.8332
## GAMLSS-RS iteration 5: Global Deviance = 608.8316
## GAMLSS-RS iteration 6: Global Deviance = 608.8315
```

```
m1
```

```
##
## Family:  c("PIG", "Poisson.Inverse.Gaussian")
## Fitting method: RS()
##
## Call:  gamlss(formula = fish ~ x, sigma.formula = ~x, family = PIG,
```

```
##     data = species)
##
## Mu Coefficients:
## (Intercept)            x
##      2.5475       0.1444
## Sigma Coefficients:
## (Intercept)            x
##     -2.0252       0.1925
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   66
## Global Deviance:     608.831
##             AIC:     616.831
##             SBC:     625.825
```

O método RS convergiu em seis iterações.

Algoritmo CG

Vamos modificar o argumento `method` para `method=CG()`.

```
m2 <- gamlss(fish~x, sigma.fo=~x, family=PIG, data=species, method=CG())
```

```
## GAMLSS-CG iteration 1: Global Deviance = 1561.463
## GAMLSS-CG iteration 2: Global Deviance = 1481.211
## GAMLSS-CG iteration 3: Global Deviance = 1407.421
## GAMLSS-CG iteration 4: Global Deviance = 1339.399
## GAMLSS-CG iteration 5: Global Deviance = 1276.056
## GAMLSS-CG iteration 6: Global Deviance = 1216.432
## GAMLSS-CG iteration 7: Global Deviance = 1159.821
## GAMLSS-CG iteration 8: Global Deviance = 1105.786
## GAMLSS-CG iteration 9: Global Deviance = 1054.036
## GAMLSS-CG iteration 10: Global Deviance = 1004.432
## GAMLSS-CG iteration 11: Global Deviance = 956.9621
## GAMLSS-CG iteration 12: Global Deviance = 911.7385
## GAMLSS-CG iteration 13: Global Deviance = 869.0208
## GAMLSS-CG iteration 14: Global Deviance = 829.2613
## GAMLSS-CG iteration 15: Global Deviance = 793.1816
## GAMLSS-CG iteration 16: Global Deviance = 761.84
## GAMLSS-CG iteration 17: Global Deviance = 736.4645
## GAMLSS-CG iteration 18: Global Deviance = 717.2406
## GAMLSS-CG iteration 19: Global Deviance = 701.5665
## GAMLSS-CG iteration 20: Global Deviance = 686.9524
```

```
## Warning in CG(): Algorithm CG has not yet converged
```

```
m2
```

```
##
## Family:  c("PIG", "Poisson.Inverse.Gaussian")
## Fitting method: CG()
##
## Call:  gamlss(formula = fish ~ x, sigma.formula = ~x, family = PIG,
##     data = species, method = CG())
##
## Mu Coefficients:
## (Intercept)            x
##      2.2177       0.1492
```

```
## Sigma Coefficients:
## (Intercept)            x
##     -6.6478       0.6648
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   66
## Global Deviance:     686.952
##             AIC:     694.952
##             SBC:     703.946
```

Observe que o algoritmo não converge ao final de 20 iterações (valor padrão). Note a diferença nos valores estimados para os coeficientes associados com σ deste ajuste para o anterior (algoritmo RS). Mais iterações são especificadas no argumento `n.cyc`.

```
m2 <- gamlss(fish~x, sigma.fo=~x, family=PIG,
data=species, method=CG(), n.cyc=100)
```

```
## GAMLSS-CG iteration 1: Global Deviance = 1561.463
## GAMLSS-CG iteration 2: Global Deviance = 1481.211
## GAMLSS-CG iteration 3: Global Deviance = 1407.421
## GAMLSS-CG iteration 4: Global Deviance = 1339.399
## GAMLSS-CG iteration 5: Global Deviance = 1276.056
## GAMLSS-CG iteration 6: Global Deviance = 1216.432
## GAMLSS-CG iteration 7: Global Deviance = 1159.821
## GAMLSS-CG iteration 8: Global Deviance = 1105.786
## GAMLSS-CG iteration 9: Global Deviance = 1054.036
## GAMLSS-CG iteration 10: Global Deviance = 1004.432
## GAMLSS-CG iteration 11: Global Deviance = 956.9621
## GAMLSS-CG iteration 12: Global Deviance = 911.7385
## GAMLSS-CG iteration 13: Global Deviance = 869.0208
## GAMLSS-CG iteration 14: Global Deviance = 829.2613
## GAMLSS-CG iteration 15: Global Deviance = 793.1816
## GAMLSS-CG iteration 16: Global Deviance = 761.84
## GAMLSS-CG iteration 17: Global Deviance = 736.4645
## GAMLSS-CG iteration 18: Global Deviance = 717.2406
## GAMLSS-CG iteration 19: Global Deviance = 701.5665
## GAMLSS-CG iteration 20: Global Deviance = 686.9524
## GAMLSS-CG iteration 21: Global Deviance = 672.9389
## GAMLSS-CG iteration 22: Global Deviance = 659.3273
## GAMLSS-CG iteration 23: Global Deviance = 645.9871
## GAMLSS-CG iteration 24: Global Deviance = 633.1439
## GAMLSS-CG iteration 25: Global Deviance = 622.6298
## GAMLSS-CG iteration 26: Global Deviance = 616.5702
## GAMLSS-CG iteration 27: Global Deviance = 613.1547
## GAMLSS-CG iteration 28: Global Deviance = 610.9197
## GAMLSS-CG iteration 29: Global Deviance = 609.6063
## GAMLSS-CG iteration 30: Global Deviance = 609.0202
## GAMLSS-CG iteration 31: Global Deviance = 608.8564
## GAMLSS-CG iteration 32: Global Deviance = 608.8331
## GAMLSS-CG iteration 33: Global Deviance = 608.8316
## GAMLSS-CG iteration 34: Global Deviance = 608.8315
```

```
m2
```

```
##
## Family:  c("PIG", "Poisson.Inverse.Gaussian")
## Fitting method: CG()
```

```
##
## Call:  gamlss(formula = fish ~ x, sigma.formula = ~x, family = PIG,
##     data = species, method = CG(), n.cyc = 100)
##
## Mu Coefficients:
## (Intercept)            x
##      2.5476       0.1444
## Sigma Coefficients:
## (Intercept)            x
##     -2.0259       0.1926
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   66
## Global Deviance:     608.831
##             AIC:     616.831
##             SBC:     625.825
```

O algoritmo convergiu em 34 iterações. Alternativamente, poderíamos ter definido valores iniciais para os parâmetros μ e σ; por exemplo usando os valores médio e o desvio padrão observado[1]:

```
gamlss(fish~x, sigma.fo=~x, family=PIG, data=species, method=CG(),
       mu.start=mean(species$fish), sigma.start=sd(species$fish))
```

```
## GAMLSS-CG iteration 1: Global Deviance = 709.195
## GAMLSS-CG iteration 2: Global Deviance = 816.4403
## GAMLSS-CG iteration 3: Global Deviance = 733.5054
## GAMLSS-CG iteration 4: Global Deviance = 703.0179
## GAMLSS-CG iteration 5: Global Deviance = 613.0298
## GAMLSS-CG iteration 6: Global Deviance = 610.1114
## GAMLSS-CG iteration 7: Global Deviance = 608.971
## GAMLSS-CG iteration 8: Global Deviance = 608.8496
## GAMLSS-CG iteration 9: Global Deviance = 608.8349
## GAMLSS-CG iteration 10: Global Deviance = 608.8322
## GAMLSS-CG iteration 11: Global Deviance = 608.8317
```

```
##
## Family:  c("PIG", "Poisson.Inverse.Gaussian")
## Fitting method: CG()
##
## Call:  gamlss(formula = fish ~ x, sigma.formula = ~x, family = PIG,
##     data = species, method = CG(), mu.start = mean(species$fish),
##     sigma.start = sd(species$fish))
##
## Mu Coefficients:
## (Intercept)            x
##      2.5476       0.1442
## Sigma Coefficients:
## (Intercept)            x
##     -2.0249       0.1926
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   66
## Global Deviance:     608.832
##             AIC:     616.832
##             SBC:     625.826
```

[1]Note que σ não é o desvio padrão da resposta na distribuição `PIG`. Compare o ajuste e o número de iteraçoes usando um valor inicial mais realístico para σ, por exemplo, `sigma.start=0.6`.

O algoritmo convergiu em 11 iterações.

Algoritmo `mixed`

Vamos agora usar o argumento `method=mixed()`.

```
m3 <- gamlss(fish~x, sigma.fo=~x, family=PIG,
data=species, method=mixed(1,100))
```

```
## GAMLSS-RS iteration 1: Global Deviance = 617.3683
## GAMLSS-CG iteration 1: Global Deviance = 652.0548
## GAMLSS-CG iteration 2: Global Deviance = 630.8294
## GAMLSS-CG iteration 3: Global Deviance = 621.8932
## GAMLSS-CG iteration 4: Global Deviance = 616.9011
## GAMLSS-CG iteration 5: Global Deviance = 613.5051
## GAMLSS-CG iteration 6: Global Deviance = 611.1476
## GAMLSS-CG iteration 7: Global Deviance = 609.7249
## GAMLSS-CG iteration 8: Global Deviance = 609.0634
## GAMLSS-CG iteration 9: Global Deviance = 608.8648
## GAMLSS-CG iteration 10: Global Deviance = 608.8339
## GAMLSS-CG iteration 11: Global Deviance = 608.8316
## GAMLSS-CG iteration 12: Global Deviance = 608.8315
```

```
m3
```

```
##
## Family:  c("PIG", "Poisson.Inverse.Gaussian")
## Fitting method: mixed(1, 100)
##
## Call:  gamlss(formula = fish ~ x, sigma.formula = ~x, family = PIG,
##     data = species, method = mixed(1, 100))
##
## Mu Coefficients:
## (Intercept)            x
##      2.5475       0.1444
## Sigma Coefficients:
## (Intercept)            x
##     -2.0259       0.1926
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   66
## Global Deviance:     608.831
##             AIC:     616.831
##             SBC:     625.825
```

Em resumo:

- Os algoritmos RS, CG e misto RS-CG chegam na mesma solução, com três casas decimais, todos com deviance global de 608.831.

```
rbind(m1_mu=coef(m1), m2_mu=coef(m2), m3_mu=coef(m3))
```

```
##       (Intercept)         x
## m1_mu    2.547521 0.1443601
## m2_mu    2.547560 0.1443603
## m3_mu    2.547527 0.1443563
```

```
rbind(m1_sigma=coef(m1, what="sigma"), m2_sigma=coef(m2, what="sigma"),
      m3_sigma=coef(m3, what="sigma"))
```

```
##          (Intercept)         x
## m1_sigma   -2.025230 0.1925237
## m2_sigma   -2.025850 0.1925763
## m3_sigma   -2.025941 0.1925880
```

```
rbind(m1_globalDev=m1$G.deviance, m2_globalDev=m2$G.deviance,
      m3_globalDev=m3$G.deviance)
```

```
##                  [,1]
## m1_globalDev 608.8315
## m2_globalDev 608.8315
## m3_globalDev 608.8315
```

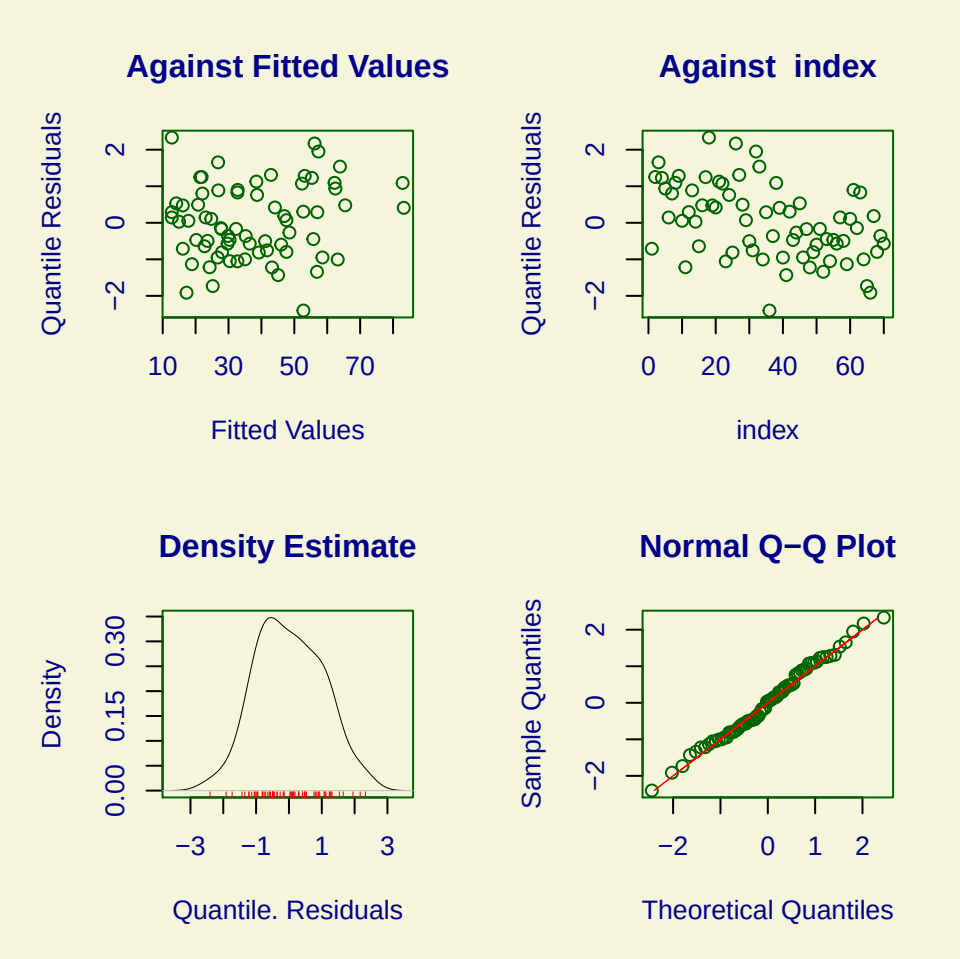
- O algoritmo RS alcança a solução em 6 iterações, o CG em 34 iterações, e o misto realiza uma iteração RS e 12 iterações CG.

```
rbind(m1_niter=m1$iter, m2_niter=m2$iter, m3_niter=m3$iter)
```

```
##          [,1]
## m1_niter    6
## m2_niter   34
## m3_niter   12
```

O diagnóstico para o modelo `PIG` no primeiro ajuste é o seguinte.

```
plot(m1)
```




```
## ******************************************************************
##   Summary of the Randomised Quantile Residuals
##                            mean   =  0.01952897
##                        variance   =  1.020841
##                coef. of skewness  =  0.1080866
##                coef. of kurtosis  =  2.44759
## Filliben correlation coefficient  =  0.9950415
## ******************************************************************
```

Como visto nos gráficos, a distribuição `PIG` modelando a dispersão apresenta um ajuste bem mais adequado aos dados que aquele obtido com a distribuição `PO`.

Por fim, vamos comparar os tempos de execução dos três modelos:

```
system.time(capture.output(m1 <- gamlss(fish~x, sigma.fo=~x, family=PIG, data=species)))
```

```
##    user  system elapsed
##    0.03    0.00    0.04
```

```
system.time(capture.output(m2 <- gamlss(fish~x, sigma.fo=~x, family=PIG, data=species,
                                        method=CG(), n.cyc=100)))
```

```
##    user  system elapsed
##     0.3     0.0     0.3
```

```
system.time(capture.output(m3 <- gamlss(fish~x, sigma.fo=~x, family=PIG, data=species,
                                        method=mixed(1,100))))
```

```
##    user  system elapsed
##    0.08    0.00    0.07
```