CE043 - GAMLSS

Família GAMLSS - Distribuições Contínuas

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

27 de agosto, 2020

Conteúdo

Introdução

Introdução

- Nesta aula, estudaremos com mais detalhes as diferentes distribuições implementadas no pacote `gamlss.dist`;

Introdução

- Nesta aula, estudaremos com mais detalhes as diferentes distribuições implementadas no pacote `gamlss.dist`;

- Em especial, abordaremos as distribuições contínuas;

Introdução

- Nesta aula, estudaremos com mais detalhes as diferentes distribuições implementadas no pacote `gamlss.dist`;
- Em especial, abordaremos as distribuições contínuas;
- Discutiremos como usar as funções de ligação dentro da função `gamlss()`;

Introdução

- Nesta aula, estudaremos com mais detalhes as diferentes distribuições implementadas no pacote `gamlss.dist`;
- Em especial, abordaremos as distribuições contínuas;
- Discutiremos como usar as funções de ligação dentro da função `gamlss()`;
- Exploraremos algumas funções auxiliares úteis para ajuste marginal de distribuições;

Introdução

- Nesta aula, estudaremos com mais detalhes as diferentes distribuições implementadas no pacote `gamlss.dist`;
- Em especial, abordaremos as distribuições contínuas;
- Discutiremos como usar as funções de ligação dentro da função `gamlss()`;
- Exploraremos algumas funções auxiliares úteis para ajuste marginal de distribuições;
- Veremos como visualizar as diferentes distribuições;

Introdução

- Nesta aula, estudaremos com mais detalhes as diferentes distribuições implementadas no pacote `gamlss.dist`;
- Em especial, abordaremos as distribuições contínuas;
- Discutiremos como usar as funções de ligação dentro da função `gamlss()`;
- Exploraremos algumas funções auxiliares úteis para ajuste marginal de distribuições;
- Veremos como visualizar as diferentes distribuições;
- Por fim, trataremos de versões censuradas das distribuições e misturas de distribuições.

A família GAMLSS de distribuições

A Família GAMLSS de Distribuições

Dentro do framework GAMLSS, podemos considerar uma função (densidade) de probabilidade bastante flexível para a resposta Y, $f(y|\boldsymbol{\theta})$, em que $\boldsymbol{\theta}^T = (\mu, \sigma, \nu, \tau)$.

Embora μ, σ, ν, τ frequentemente representem parâmetros relacionados com locação, escala, assimetria e curtose, nem sempre este será o caso.

A única restrição para a distribuição específica de Y é que $f(y|\boldsymbol{\theta})$ e sua derivada primeira (e opcionalmente a segunda derivada e a cruzada) com respeito a cada um dos parâmetros de $\boldsymbol{\theta}$ seja calculável.

As derivadas explícitas são preferíveis, embora derivadas numéricas possam ser usadas (resultando em velocidade computacional reduzida).

A Família GAMLSS de Distribuições

O tipo da distribuição a ser usada depende do tipo da variável resposta a ser modelada.

Dentro da família GAMLSS há três tipos distintos de distribuições:

1. Distribuições contínuas;
2. Distribuições discretas;
3. Distribuições mistas.

As distribuições estão implementadas no pacote `gamlss.dist` e podem ser acessadas em `?gamlss.family`. A Figura 10 ilustra os diferentes tipos de distribuições implementadas.

Famílias de Distribuições GAMLSS

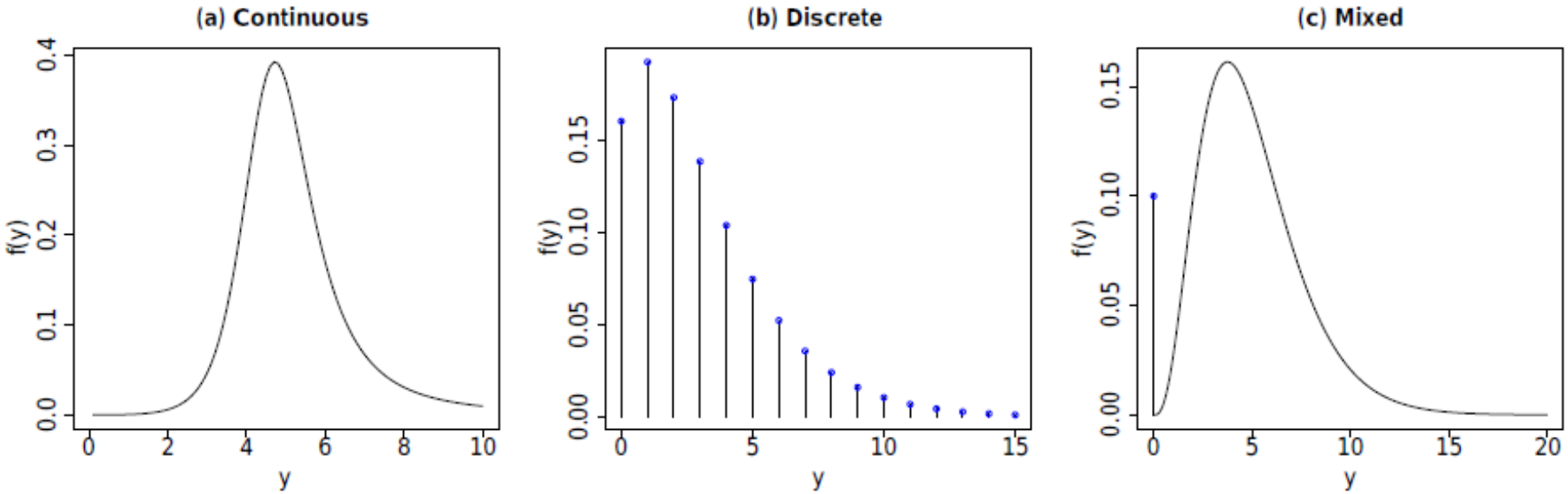
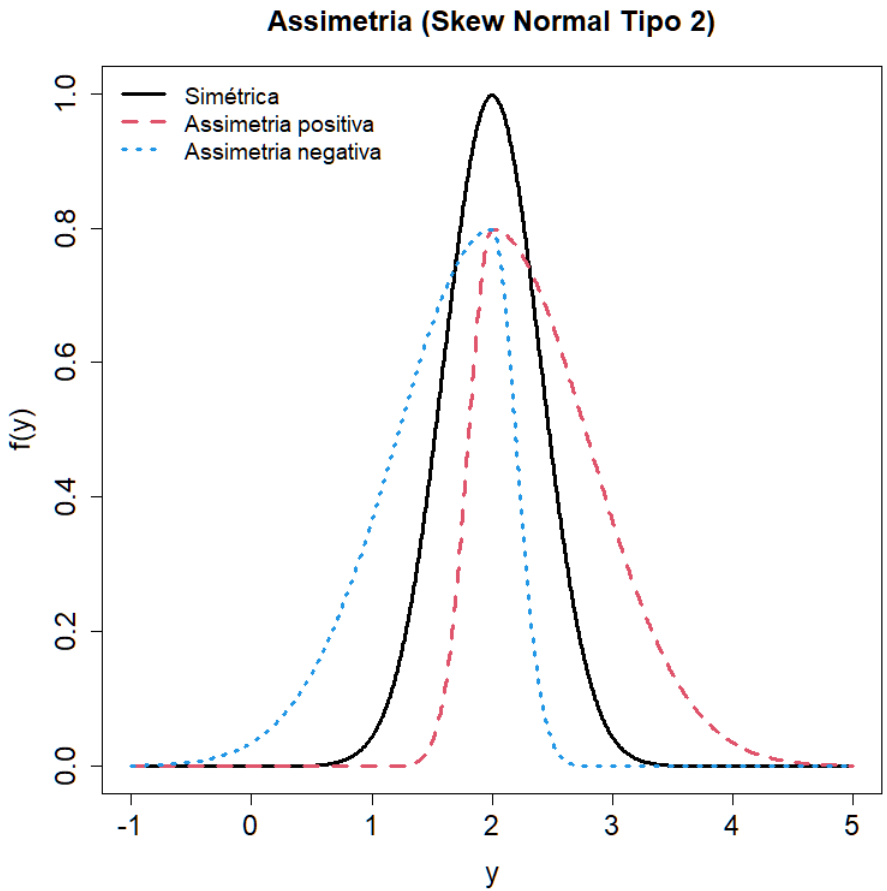


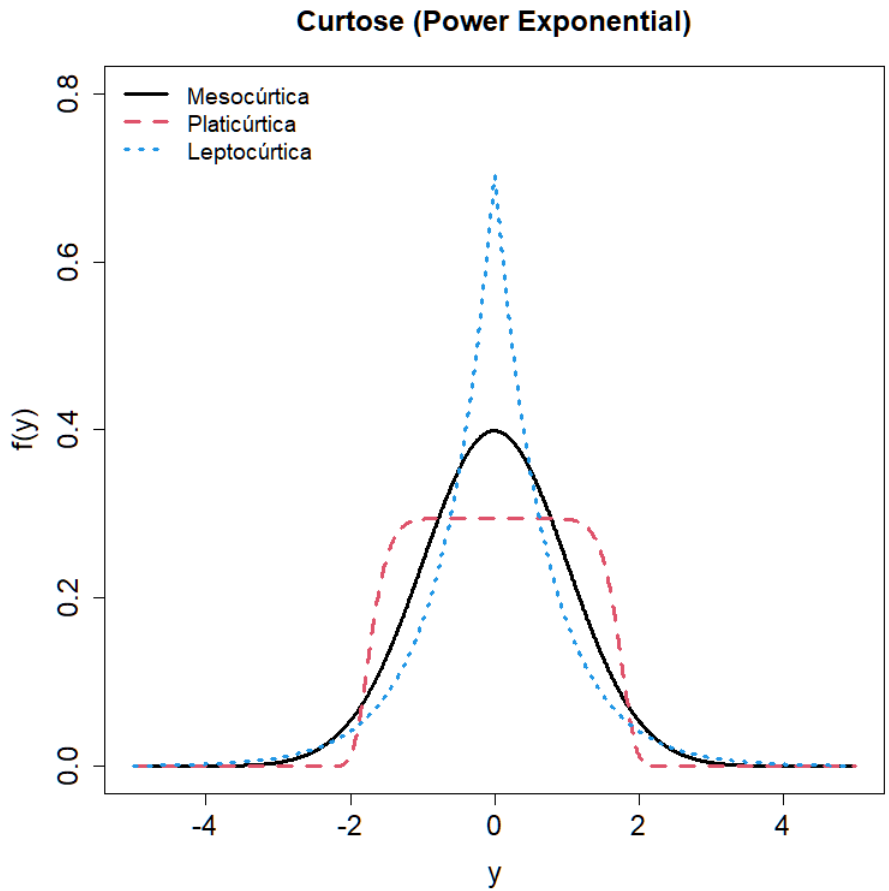
Figura 1: Diferentes tipos de distribuições (a) contínua, (b) discreta, (c) mista

Distribuições Contínuas

Famílias de Distribuições GAMLSS

As distribuições contínuas podem ser simétricas, ter assimetria negativa ou positiva, e ainda ser mesocúrticas, leptocúrticas ou platicúrticas.





Famílias de Distribuições GAMLSS

As distribuições podem ter de um a quatro parâmetros (μ, σ, ν e τ) e para algumas distribuições estes parâmetros têm interpretações diferentes. Por exemplo, para a distribuição Gama, $\texttt{GA}(\mu, \sigma)$, a média é dada por μ enquanto a variância é dada por $\sigma^2\mu^2$.

A relação entre os parâmetros da distribuição e as medidas dependentes dos momentos (média, variância, assimetria e curtose) é uma função das propriedades específicas da distribuição.

As Figuras 2 e 3 mostram todas as distribuições contínuas disponíveis.

Distribuições Contínuas

Distribution	gamlss name	Range R_Y	Parameter link functions μ	σ	ν	τ
beta	BE	$(0,1)$	logit	logit	-	-
Box–Cox Cole–Green	BCCG	$(0,\infty)$	ident.	log	ident.	-
Box–Cox Cole–Green orig.	BCCGo	$(0,\infty)$	log	log	ident.	-
Box–Cox power exponential	BCPE	$(0,\infty)$	ident.	log	ident.	log
Box–Cox power expon. orig.	BCPEo	$(0,\infty)$	log	log	ident.	log
Box–Cox t	BCT	$(0,\infty)$	ident.	log	ident.	log
Box–Cox t orig.	BCTo	$(0,\infty)$	log	log	ident.	log
exponential	EXP	$(0,\infty)$	log	-	-	-
exponential Gaussian	exGAUS	$(-\infty,\infty)$	ident.	log	log	-
exponential gen. beta 2	EGB2()	$(-\infty,\infty)$	ident.	log	log	log
gamma	GA	$(0,\infty)$	log	log	-	-
generalized beta type 1	GB1	$(0,1)$	logit	logit	log	log
generalized beta type 2	GB2	$(0,\infty)$	log	log	log	log
generalized gamma	GG	$(0,\infty)$	log	log	ident.	-
generalized inv. Gaussian	GIG	$(0,\infty)$	log	log	ident.	-
generalized t	GT	$(-\infty,\infty)$	ident.	log	log	log
Gumbel	GU	$(-\infty,\infty)$	ident.	log	-	-
inverse Gamma	IGAMMA	$(0,\infty)$	log	log	-	-
inverse Gaussian	IG	$(0,\infty)$	log	log	-	-
Johnson's SU repar.	JSU	$(-\infty,\infty)$	ident.	log	ident.	log
Johnson's SU original	JSUo	$(-\infty,\infty)$	ident.	log	ident.	log
logistic	LO	$(-\infty,\infty)$	ident.	log	-	-
logit normal	LOGITNO	$(0,1)$	ident.	log	-	-
log normal	LOGNO	$(0,\infty)$	ident.	log	-	-

Figura 2: Distribuições contínuas com funções de ligação padrão.

Distribuições Contínuas

Distribution	gamlss name	Range R_Y	Parameter link functions μ	σ	ν	τ
log normal 2	LOGNO2	$(0,\infty)$	log	log	-	-
log normal (Box–Cox)	LNO	$(0,\infty)$	ident.	log	fixed	-
NET	NET	$(-\infty,\infty)$	ident.	log	fixed	fixed
normal	NO, NO2	$(-\infty,\infty)$	ident.	log	-	-
normal family	NOF	$(-\infty,\infty)$	ident.	log	-	-
Pareto 2	PARETO2	$(0,\infty)$	log	log	-	-
Pareto 2 original	PARETO2o	$(0,\infty)$	log	log	-	-
Pareto 2 repar	GP	$(0,\infty)$	log	log	-	-
power exponential	PE	$(-\infty,\infty)$	ident.	log	log	-
reverse gen. extreme	RGE	$y > \mu - (\sigma/\nu)$	ident.	log	log	-
reverse Gumbel	RG	$(-\infty,\infty)$	ident.	log	-	-
sinh-arcsinh	SHASH	$(-\infty,\infty)$	ident.	log	log	log
sinh-arcsinh original	SHASHo	$(-\infty,\infty)$	ident.	log	ident.	log
sinh-arcsinh original 2	SHASHo2	$(-\infty,\infty)$	ident.	log	ident.	log
skew normal type 1	SN1	$(-\infty,\infty)$	ident.	log	ident.	-
skew normal type 2	SN2	$(-\infty,\infty)$	ident.	log	log	-
skew power exp. type 1	SEP1	$(-\infty,\infty)$	ident.	log	ident.	log
skew power exp. type 2	SEP2	$(-\infty,\infty)$	ident.	log	ident.	log
skew power exp. type 3	SEP3	$(-\infty,\infty)$	ident.	log	log	log
skew power exp. type 4	SEP4	$(-\infty,\infty)$	ident.	log	log	log
skew t type 1	ST1	$(-\infty,\infty)$	ident.	log	ident.	log
skew t type 2	ST2	$(-\infty,\infty)$	ident.	log	ident.	log
skew t type 3	ST3	$(-\infty,\infty)$	ident.	log	log	log
skew t type 3 repar	SST	$(-\infty,\infty)$	ident.	log	log	log-2
skew t type 4	ST4	$(-\infty,\infty)$	ident.	log	log	log
skew t type 5	ST5	$(-\infty,\infty)$	ident.	log	ident.	log
t Family	TF	$(-\infty,\infty)$	ident.	log	log	-
t Family repar	TF2	$(-\infty,\infty)$	ident.	log	log-2	-
Weibull	WEI	$(0,\infty)$	log	log	-	-
Weibull (PH)	WEI2	$(0,\infty)$	log	log	-	-
Weibull (μ the mean)	WEI3	$(0,\infty)$	log	log	-	-

Figura 3: Distribuições contínuas com funções de ligação padrão (cont).

Especificação da Distribuição

Para ajustar uma distribuição da família GAMLSS podemos usar o comando `gamlss(y~1, family=)` em que o argumento `family` pode ser qualquer opção de `gamlss.family`.

Considere a distribuição logística, `LO`(μ, σ), que é simétrica em torno de $y = \mu$, e para a qual $E(Y) = \mu$ e $V(Y) = \pi^2\sigma^2/3$. Para ajustá-la aos dados de circunferência abdominal são equivalentes as seguintes especificações:

```
library(gamlss)
h1 <- gamlss(y~cs(x), sigma.formula=~x, family=LO, data=abdom)
h2 <- gamlss(y~cs(x), sigma.formula=~x, family=LO(), data=abdom)
h3 <- gamlss(y~cs(x), sigma.formula=~x, family="LO",
             data=abdom)
h4 <- gamlss(y~cs(x), sigma.formula=~x, family=LO(mu.link=identity,
             sigma.link=log), data=abdom)
```

Exemplo

Considere os dados de retorno da bolsa turca disponíveis no objeto `tse` do pacote `gamlss.data`. A variável de interesse `ret`, registra 2868 (log) retornos diários para o período iniciado em 1/1/1988.

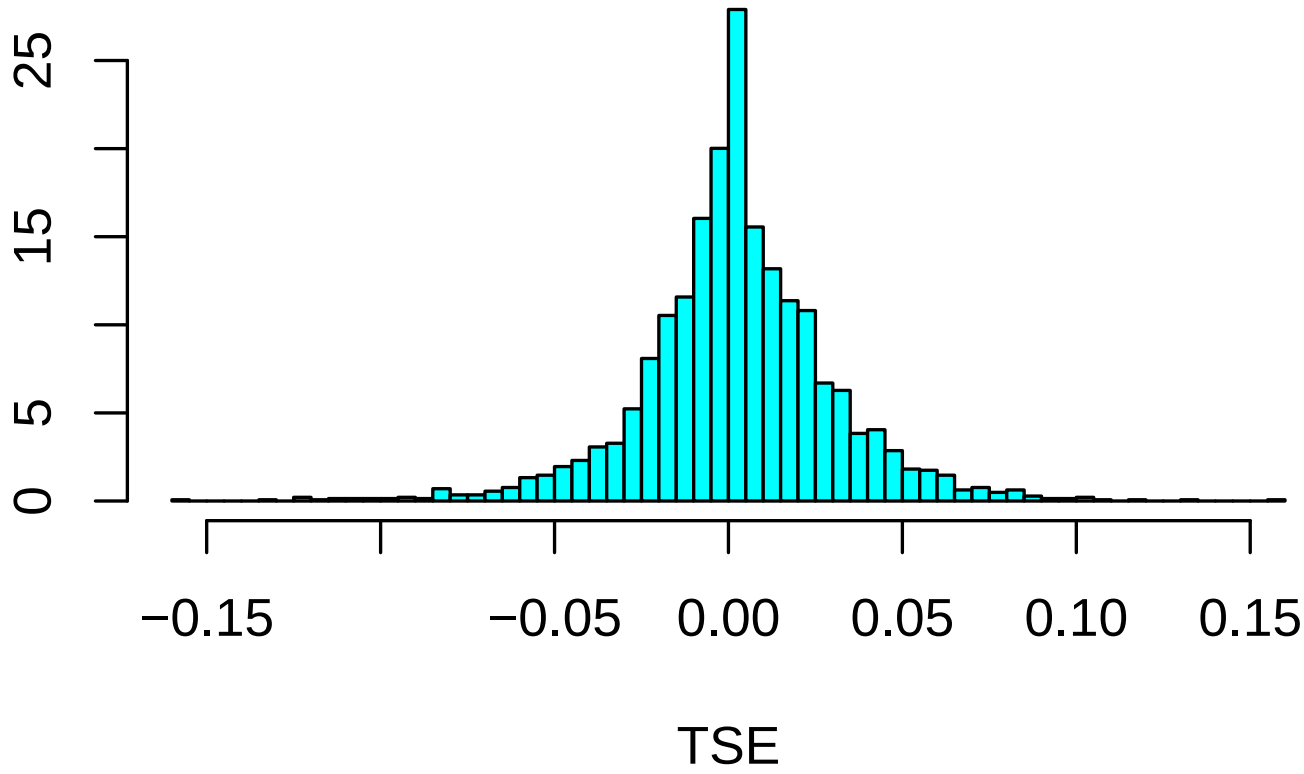
```
data(tse)
head(tse, 4)
```

```
  year month day          ret currency           tl
1 1988     1   5  0.022934971   102.32  0.009960531
2 1988     1   6  0.008660590   103.21  0.003761247
3 1988     1   7 -0.012970097   101.88 -0.005632842
4 1988     1   8 -0.008378148   101.03 -0.003638583
```

Vamos utilizar estes dados para ilustrar o uso de algumas funções do pacote `gamlss`.

A função truehist()

```
truehist(tse$ret, xlab="TSE")
```



Exemplo

Inicialmente vamos ajustar uma distribuição da família t, $\texttt{TF}(\mu, \sigma, \nu)$, que é simétrica em torno de $y = \mu$ e é adequada para modelar dados leptocúrticos, isto é, com curtose maior que a distribuição normal.

Para esta distribuição, μ, σ e ν são parâmetros de locação, escala e curtose. Mais especificamente:

- A média é μ se $\nu > 1$, e é indefinida se $\nu \leq 1$;
- A variância vale $\frac{\sigma^2\nu}{\nu-2}$ se $\nu > 2$, e ∞ se $\nu \leq 2$; e
- O excesso de curtose é dado por $\frac{6}{\nu-4}$ se $\nu > 4$, e ∞ se $\nu \leq 4$.

```
m1 <- gamlss(ret~1, data=tse, family=TF)
m1_sm <- capture.output(summary(m1))
```

Exemplo

```
cat(m1_sm[c(8:12,16:20,24:28)], sep='\n')
```

```
-------------------------------------------------------------------
Mu link function:  identity
Mu Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0021964  0.0004284   5.128 3.13e-07 ***
-------------------------------------------------------------------
Sigma link function:  log
Sigma Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.95841    0.02631  -150.4   <2e-16 ***
-------------------------------------------------------------------
Nu link function:  log
Nu Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.23115    0.07696      16   <2e-16 ***
```

A função gamlssML()

Para o caso sem covariáveis, a função **gamlssML()** do pacote **gamlss.util** usa técnicas de otimização numérica que são mais rápidas que o algoritmo usado em **gamlss()**.

```
m2 <- gamlssML(ret, data=tse, family=TF)
m2_sm <- capture.output(summary(m2))
cat(m2_sm[9:20], sep='\n')
```

```
Coefficient(s):
              Estimate   Std. Error    t value   Pr(>|t|)
eta.mu     0.002195783  0.000428278    5.12701 2.9438e-07 ***
eta.sigma -3.958731413  0.026318910 -150.41396 < 2.22e-16 ***
eta.nu     1.230441925  0.076931906   15.99391 < 2.22e-16 ***
---
```

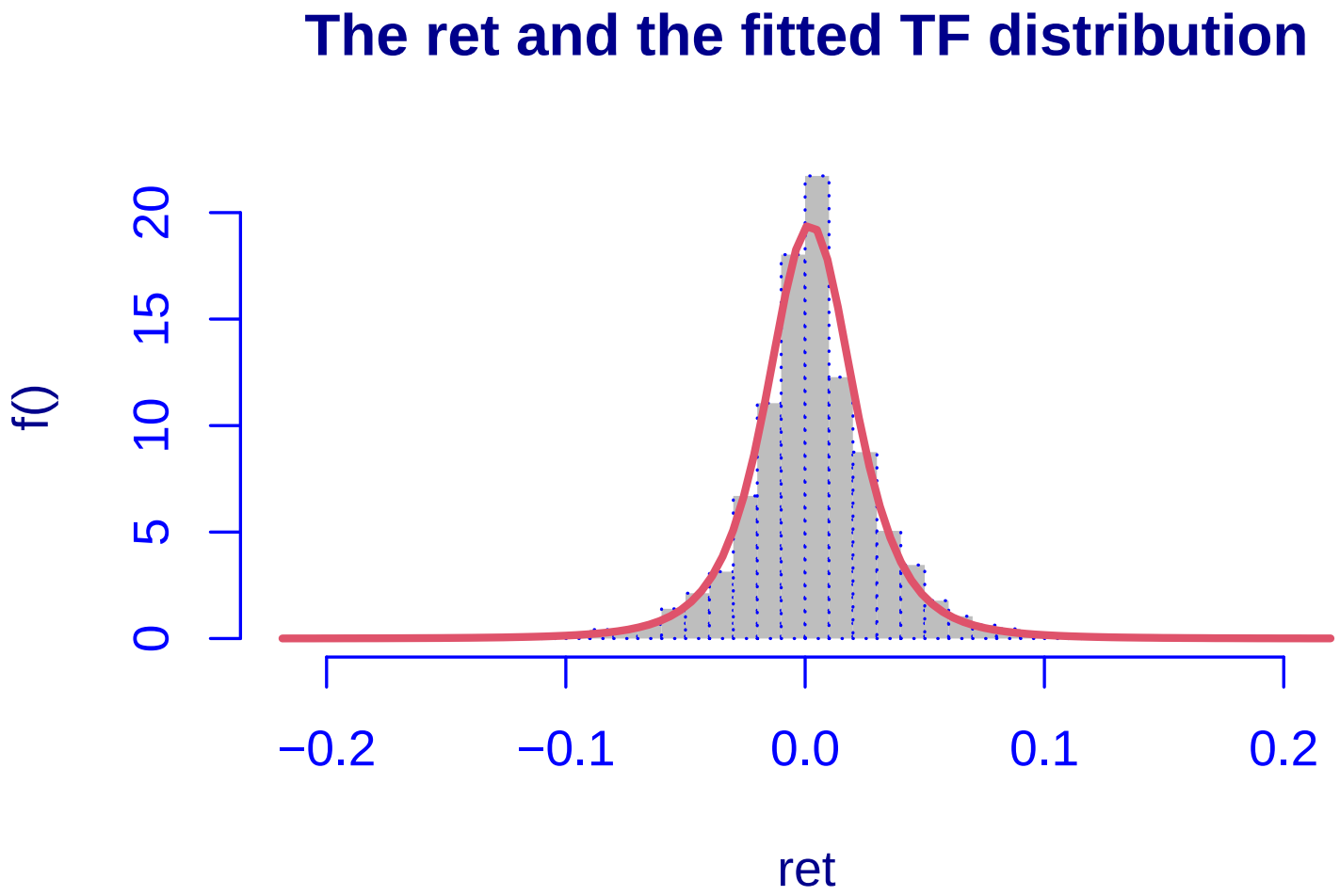
A função `histDist()`

A função `histDist()` usa `gamlssML()` como algoritmo padrão para ajustar o modelo, mas mostra ainda o histograma juntamente com a distribuição ajustada aos dados.

Esta função é apropriada para o caso sem covariáveis.

A função histDist()

```
m3 <- histDist(ret, data=tse, family=TF, nbins=30, line.wd=2.5)
```



A função `fitDist()`

A função `fitDist()` usa `gamlssML()` para ajustar um conjunto pré determinado de distribuições e escolher a *melhor* de acordo com o GAIC, com penalidade padrão de $\kappa = 2$.

Os modelos ajustados são mostrados em ordem decrescente de GAIC:

```
m5 <- fitDist(ret, data=tse, type="realline")
```

```
  Error in solve.default(oout$hessian) :
    Lapack routine dgesv: system is exactly singular: U[3,3] = 0
```

```
#m5$failed
head(m5$fits)
```

```
      SEP2       SEP1       SEP3       SEP4         PE        PE2
-12879.01 -12876.88 -12876.14 -12865.04 -12862.09 -12862.09
```

A função `fitDist()`

As quatro distribuições com melhor ajuste são as quatro diferentes versões da distribuição `SEP` (*Skew Exponential Power*).

As distribuições `SEP` têm quatro parâmetros com diferentes interpretações entre suas versões e envolve distribuições com cauda pesada e assimetria.

Distribuições `SEP` têm demonstrado grande utilidade prática para modelar dados de retornos financeiros, os quais são geralmente caracterizados por heterocedasticidade condicional e caudas mais pesadas do que as da normal.

A função `fitDist()`

```
m5$family
```

```
  [1] "SEP2"                                "Skew Exponential Power type 2"
```

```
m5$Allpar
```

```
       eta.mu     eta.sigma        eta.nu       eta.tau
 0.0001000987 -3.9679583307  0.0759356392 -0.0753877623
```

```
SEP2()
```

```
  GAMLSS Family: SEP2 Skew Exponential Power type 2
  Link function for mu    : identity
  Link function for sigma: log
  Link function for nu    : identity
  Link function for tau   : log
```

A função `fitDist()`

A distribuição `SEP2` é definida por

$$f_Y(y|\mu,\sigma,\nu,\tau) = \frac{2}{\sigma} f_{Z_1}(z)\Phi(\omega),$$

em que $z = (y-\mu)/\sigma$ e $\omega = sinal(z)|z|^{\tau/2}\nu\sqrt{2/\tau}$ e f_{Z_1} é a *pdf* de $Z_1 \sim PE2(0, \tau^{1/\tau}, \tau)$ e $\Phi(\omega)$ é a *cdf* da distribuição normal padrão avalidada em ω.

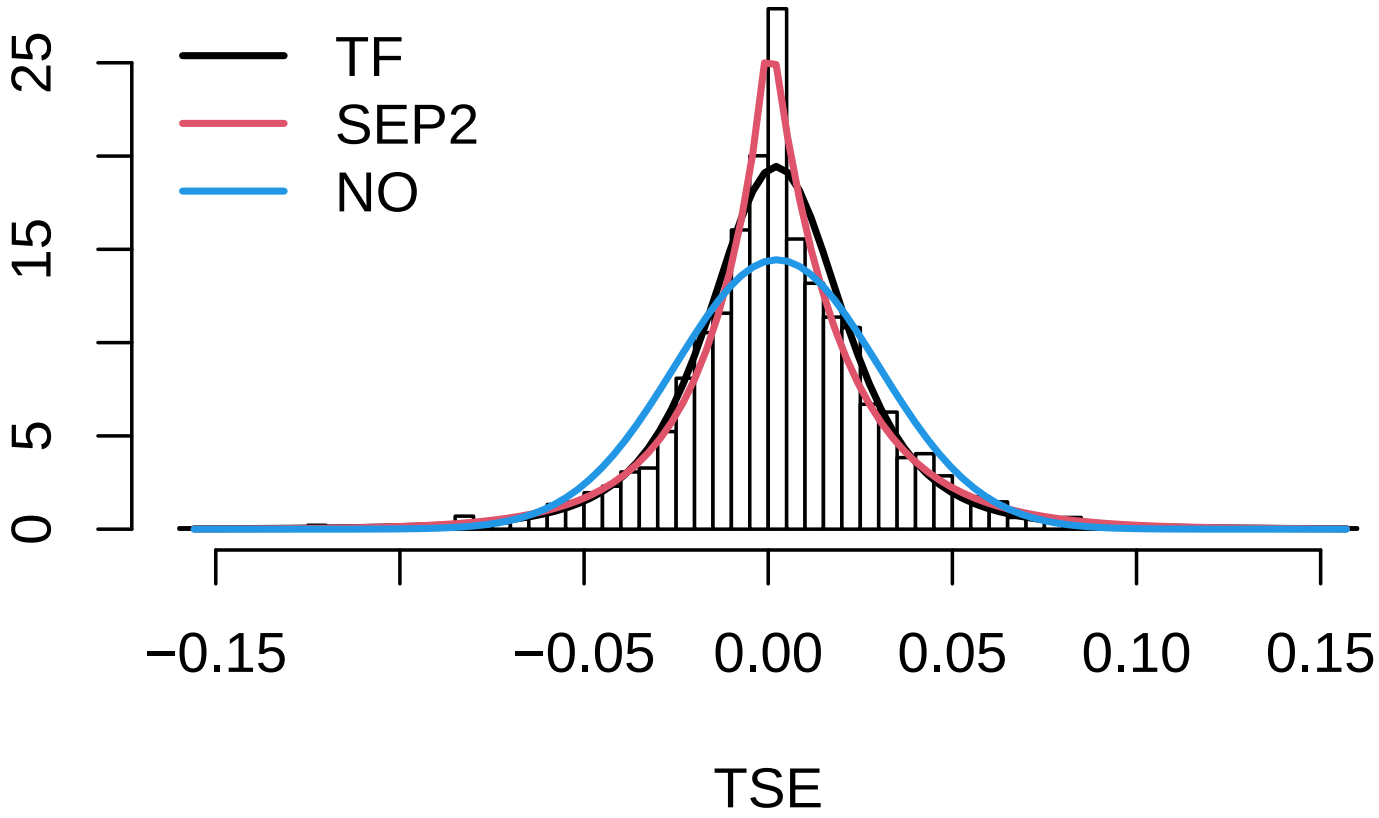
- $\nu > 0$ indica assimetria positiva e $\nu < 0$ negativa;
- $\tau > 2$ indica dados platicúrticos e $\tau < 2$ leptocúrticos;
- $E(Y) = \mu + E(Z)$
- $V(Y) = \sigma^2 V(Z) = \sigma^2 \left\{ \frac{\tau^{2/\tau}\Gamma(3\tau^{-1})}{\Gamma(\tau^{-1})} - [E(Z)]^2 \right\}$

Para $\tau = 2$ temos a distribuição skew normal tipo 1, `SN1`(μ, σ, ν), enquanto para $\nu = 0$ e $\tau = 2$ temos a distribuição normal, `NO`(μ, σ).

Comparação dos ajustes

```
truehist(tse$ret, xlab="TSE", col="white")
xx <- seq(min(tse$ret), max(tse$ret), length.out=100)
ll.TF <- dTF(xx, mu=coef(m2), sigma=exp(coef(m2, "sigma")),
             nu=exp(coef(m2, "nu")))
ll.SEP2 <- dSEP2(xx, mu=m5$mu, sigma=m5$sigma, nu=m5$nu,
                 tau=m5$tau)
m.NO <- gamlssML(ret, data=tse, family=NO)
ll.NO <- dNO(xx, mu=m.NO$mu, sigma=m.NO$sigma)
lines(xx, ll.TF, lwd=2); lines(xx, ll.SEP2, col=2, lwd=2);
lines(xx, ll.NO, col=4, lwd=2)
legend("topleft", col=c(1,2,4), legend = c("TF", "SEP2", "NO"),
       lty = 1, bty = "n", lwd = 2)
```

Comparação dos ajustes



A função `fitDistPred()`

A função `fitDistPred()` usa a função `gamlssMLpred()` para realizar os ajustes marginais mas o modelo final é selecionado pelo menor valor da *deviance* global de predição.

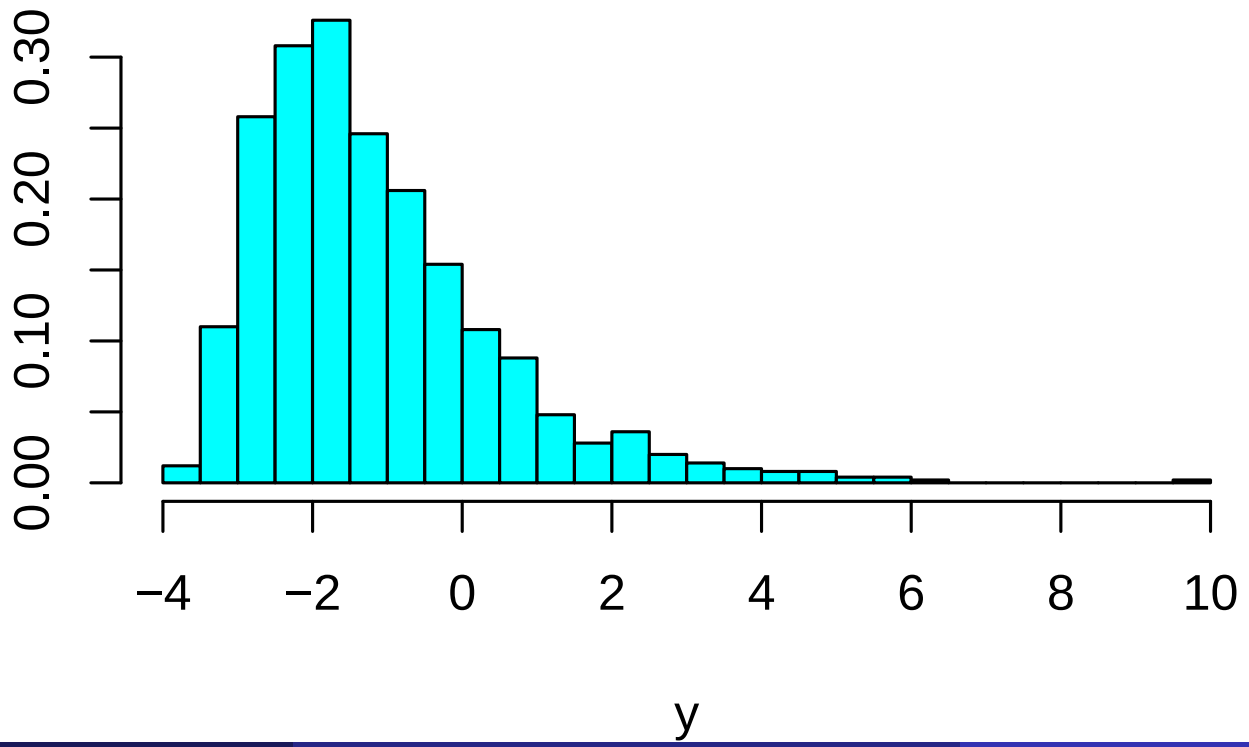
Vamos explorar esta função a dados gerados da distribuição da distribuição `JSU`(μ, σ, ν, τ). Tal distribuição é uma reparametrização da distribuição S_u de Johnson, denotada por `JSUo`(μ, σ, ν, τ).

- $E(Y) = \mu$;
- $V(Y) = \sigma^2$;
- $\nu > 0$ indica assimetria positiva e $\nu < 0$ negativa;
- τ determina a curtose.

A distribuição é leptocúrtica e se aproxima da normal quando $\tau \to \infty$.

A função fitDistPred()

```
set.seed(123)
y <- rJSU(1000, mu=-1.13, sigma=1.67, nu=11.15, tau=2)
truehist(y)
```



A função fitDistPred()

```
m1 <- fitDist(y, type="realline")
head(m1$fits)
```

```
      JSU      JSUo      EGB2      SEP4       ST5       ST2
  3470.854 3470.854 3470.959 3471.580 3473.660 3475.653
```

```
# cria dados de validação
yn <- rJSU(500, mu=-1.13, sigma=1.67, nu=11.15, tau=2)
# escolhe a distribuição que melhor se ajusta aos novos dados
p1 <- fitDistPred(y, type="realline", newdata=yn)
head(p1$fits)
```

```
      ST5       JSU      JSUo      SEP4      EGB2       ST2
  1714.689 1715.365 1715.366 1716.840 1717.873 1720.458
```

Visualização de distribuições GAMLSS

Visualização de distribuições GAMLSS: demos

Uma distribuição pode ser visualizada interativamente no R usando o pacote gamlss.demo.

```
library(gamlss.demo)
gamlss.demo()
```

Um menu será aberto, e escolhendo a opção "Demos for gamlss.family distributions", o usuário poderá visualizar as diferentes distribuições.

Alternativamente, podemos digitar demo.NAME(), em que NAME é o nome da distribuição, por exemplo demo.NO() para a distribuição normal.

Visualização de distribuições GAMLSS: `pdf.plot()`

Um método alternativo é usar a função `pdf.plot()`, útil para plotar diferentes distribuições ajustadas.

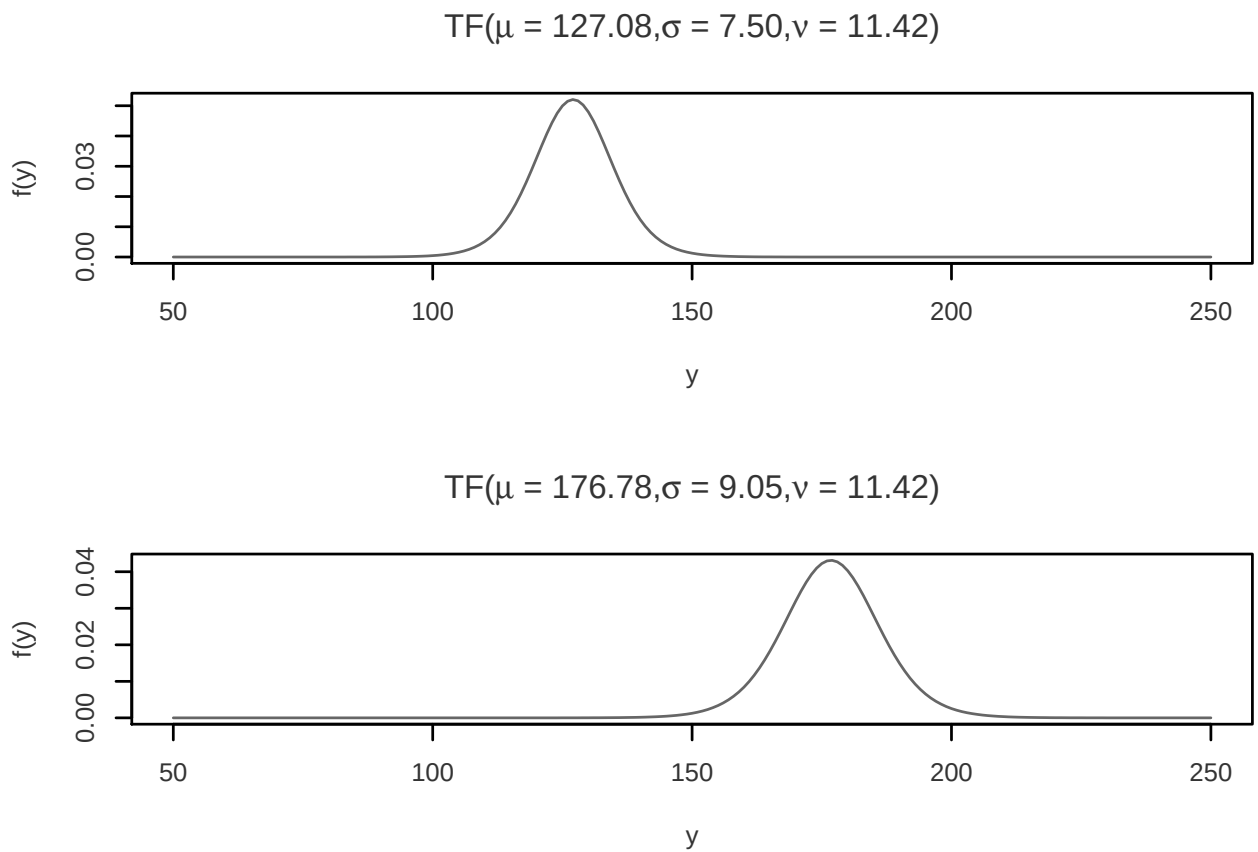
Como exemplo, considere o ajuste da distribuição `TF` aos dados `abdom`, no qual a resposta y é circunferência abdominal e a covariável x é a idade gestacional.

```
m1 <- gamlss(y~pb(x), sigma.fo=~pb(x), data=abdom, family=TF,
             trace=FALSE)
```

Como ilustração, considere as distribuições ajustadas para as observações de número 100 e 200.

Visualização de distribuições GAMLSS: pdf.plot()

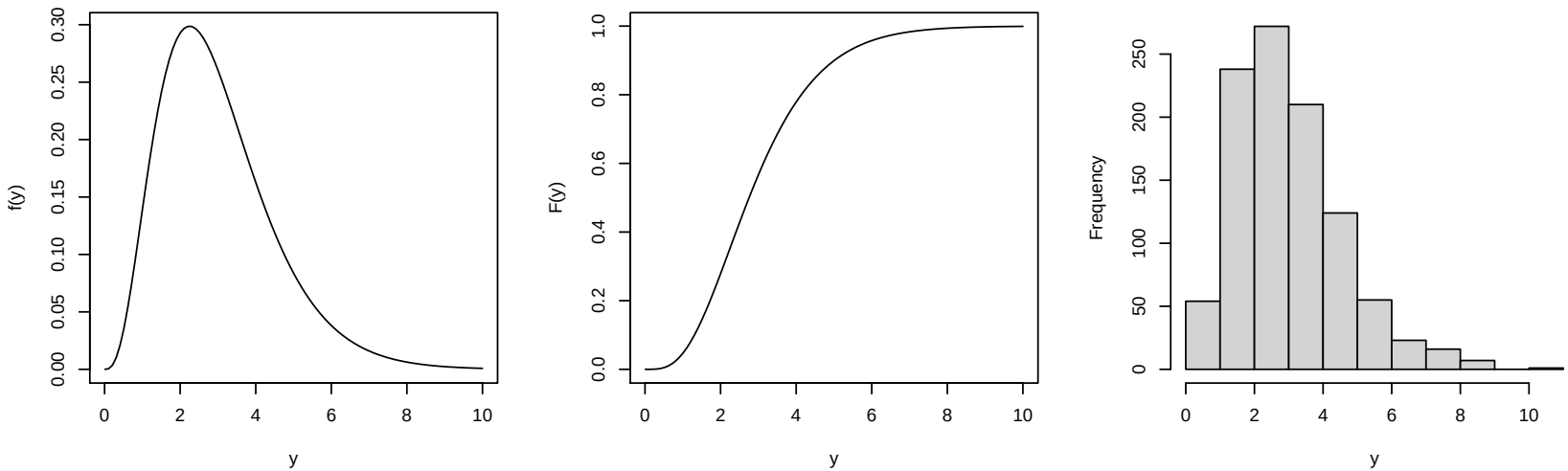
```
pdf.plot(m1, obs=c(100,200), min=50, max=250)
```



Visualização de distribuições GAMLSS

A seguir vamos usar as funções `d`, `p` e `r` para plotar a pdf, cdf e histograma de uma amostra aleatória da distribuição Gama.

```
mu=3; sigma=.5; par(mfrow=c(1,3))
curve(dGA(y, mu, sigma), 0.01, 10, xname="y",ylab="f(y)") ## pdf
curve(pGA(y, mu, sigma), 0.01, 10, xname="y", ylab="F(y)") ## cdf
y<-rGA(1000, mu, sigma) ## random sample
hist(y,col="lightgray",main="")
```



Funções de ligação

Funções de ligação

Uma função de ligação relaciona um parâmetro da distribuição a seu preditor, e.g. $g_1(\mu) = \eta_1$.

A função `make.link.gamlss()` cria todas as funções de ligação padrão dentro do pacote `gamlss.dist`.

É possível ainda criar uma função de ligação, o que não será explorado aqui.

A Figura 4 mostra todas as funções de ligação disponíveis, juntamente com o suporte usual para o parâmetro correspondentes.

Funções de ligação

Parameter range	Link functions	Formula for $g(\theta)$
$-\infty$ to ∞	`identity`	θ
0 to ∞	`log`	$\log(\theta)$
	`sqrt`	$\sqrt{\theta}$
	`inverse`	$1/\theta$
	`'1/mu^2'`	$1/\theta^2$
	`'mu^2'`	θ^2
0 to 1	`logit`	$\log[\theta/(1-\theta)]$
	`probit`	$\Phi^{-1}(\theta)$
	`cauchit`	$\tan(\pi(\theta-0.05))$
	`cloglog`	$\log(-\log(1-\theta))$
1 to ∞	`logshiftto1`	$\log(\theta-1)$
2 to ∞	`logshiftto2`	$\log(\theta-2)$
0.00001 to ∞	`logshiftto0` or `Slog`[1]	$\log(\theta-0.00001)$

[1] This function was created to avoid the value of positive parameters too close to zero.

Figura 4: Funções de ligação disponíveis.

Como mostrar as funções de ligação disponíveis

A função de ligação padrão para cada parâmetro de uma distribuição pode ser encontrada pelo nome da distribuição.

Por exemplo, para a distribuição Gama:

```
GA()
```

```
GAMLSS Family: GA Gamma
Link function for mu    : log
Link function for sigma: log
```

As funções de ligação para ambos μ e σ são log.

Como mostrar e alterar a função de ligação padrão

A função `show.link()` mostra todas as funções de ligação disponíveis.

```
show.link(GA)
```

```
$mu
c("inverse", "log", "identity", "own")

$sigma
c("inverse", "log", "identity", "own")
```

Para alterar a função de ligação de μ para identidade na distribuição Gama, especificamos `GA(mu.link=identity)` no argumento `family` da função `gamlss()`, i.e. `family=GA(mu.link=identity)`.

A opção `"own"` pode ser usada no caso de uma função de ligação criada pelo usuário.

Estendendo a família GAMLSS de distribuições

Estendendo a família GAMLSS de distribuições

Há várias formas de estender as distribuições em `gamlss.family`.

Podemos:

- Criar uma nova distribuição `gamlss.family`;
- Criar uma versão *log* ou *logit* de uma distribuição partindo de uma distribuição contínua na reta real;
- Truncar uma distribuição existente de `gamlss.family`;
- Usar uma versão censurada de uma distribuição de `gamlss.family` existente; e
- Combinar diferentes distribuições para criar uma nova mistura finita.

Os três primeiros tópicos serão estudados no Módulo 7.

Distribuições censuradas

O pacote `gamlss.cens` pode ser utilizado quando uma ou mais observações da resposta são censuradas à esquerda ou à direita (i.e. se encontram acima ou abaixo de valores conhecidos), ou de forma mais geral, se encontram em um intervalo conhecido.

Por exemplo, os respondentes de uma pesquisa relatam sua renda em um intervalo, e as respostas são da forma [\$0; \$100], (\$100; \$200], etc.

A função `gen.cens()` admite qualquer distribuição `gamlss.family` e cria uma nova função que ajusta uma resposta com observações censuradas à esquerda, à direita ou intervalo.

Distribuições censuradas

A função de verossimilhança usual, definida como

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} f(y_i|\boldsymbol{\theta})$$

é modificada, no caso de variáveis respostas intervalares independentes, para

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} [F(y_{2i}|\boldsymbol{\theta}) - F(y_{1i}|\boldsymbol{\theta})] \tag{1}$$

em que $F(\cdot)$ é a cdf e $(y_{1i}, y_{2i}]$ é o intervalo observado para a i-ésima observação.

- Para uma observação censurada à esquerda, censurada abaixo de y_{2i}, defina $y_{1i} = -\infty$, assim $F(y_{1i}|\boldsymbol{\theta}) = 0$ em (1).
- Para uma observação censurada à direita, censurada acima de y_{1i}, defina $y_{2i} = \infty$, assim $F(y_{2i}|\boldsymbol{\theta}) = 1$ em (1).

Exemplo

Considere uma distribuição Weibull que permite que uma variável resposta intervalar seja ajustada.

Os dados estão no objeto `lip` e são oriundos de um projeto de pesquisa experimental em enzimologia que objetivou desenvolver um modelo genérico de deterioração de alimentos.

A variável resposta é `y` é definida como uma variável intervalar, enquanto as variáveis explanatórias são `Tem`, `pH`, `aw`.

Exemplo

```
library(gamlss.cens)
data(lip)
help(lip)
head(lip$y)
```

```
[1]  1-       1-       1-       1-       [11, 18]  1-
```

- O valor $1-$ indica um intervalo $(1, \infty)$ não incluindo 1, enquanto $[11, 18]$ indica o intervalo $(11, 18]$ não incluindo o 11 mas incluindo o 18.
- Este conjunto de dados não possui observações censuradas à esquerda. Se tivesse, for exemplo, um valor de -5 indicaria o intervalor $(-\infty, 5]$ incluindo o 5.

Exemplo

Considere gerar uma distribuição Weibull, `WEI2`(μ, σ) censurada em intervalo, a qual permite observações censuradas à esquerda e à direta.

```
gen.cens(WEI2,type="interval")
```

```
A censored family of distributions from WEI2 has been genera
 and saved under the names:
 dWEI2ic pWEI2ic qWEI2ic WEI2ic
The type of censoring is interval
```

```
WEI2ic()
```

```
GAMLSS Family: WEI2ic interval censored Weibull type 2
Link function for mu    : log
Link function for sigma: log
```

Exemplo

```
weimi <- gamlss(y ~ poly(Tem,2) + poly(pH,2) + poly(aw,2),
       data=lip, family=WEI2ic, n.cyc=100, trace=FALSE)
weimi_sm <- capture.output(summary(weimi))
cat(weimi_sm[9:21], sep='\n')
```

```
-------------------------------------------------------------------
Mu link function:  log
Mu Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    -5.4177     0.7183  -7.542 1.31e-11 ***
poly(Tem, 2)1  36.4511     4.5706   7.975 1.42e-12 ***
poly(Tem, 2)2  -1.2301     2.0141  -0.611   0.5426
poly(pH, 2)1   21.0771     3.2225   6.541 1.90e-09 ***
poly(pH, 2)2   -4.8026     2.0584  -2.333   0.0214 *
poly(aw, 2)1   32.3043     4.3581   7.413 2.52e-11 ***
poly(aw, 2)2    1.1093     1.8584   0.597   0.5518
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Exemplo

```
cat(weimi_sm[23:38], sep='\n')
```

```
------------------------------------------------------------
Sigma link function:  log
Sigma Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.1386     0.1531   0.906    0.367

------------------------------------------------------------
No. of observations in the fit:  120
Degrees of Freedom for the fit:  8
      Residual Deg. of Freedom:  112
                      at cycle:  62

Global Deviance:     138.3094
            AIC:     154.3094
            SBC:     176.6093
************************************************************
```

Exemplo

Alternativamente, poderíamos fazer:

```
weimi2 <- gamlss(y ~ poly(Tem,2) + poly(pH,2) + poly(aw,2), data=lip,
           family=cens(WEI2, type="interval"), n.cyc=100, trace=FALSE)
weimi_sm2 <- capture.output(summary(weimi2))
cat(weimi_sm2[c(12:20,23,26:28)], sep='\n')
```

```
Mu Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    -5.4177     0.7183  -7.542 1.31e-11 ***
poly(Tem, 2)1  36.4511     4.5706   7.975 1.42e-12 ***
poly(Tem, 2)2  -1.2301     2.0141  -0.611   0.5426
poly(pH, 2)1   21.0771     3.2225   6.541 1.90e-09 ***
poly(pH, 2)2   -4.8026     2.0584  -2.333   0.0214 *
poly(aw, 2)1   32.3043     4.3581   7.413 2.52e-11 ***
poly(aw, 2)2    1.1093     1.8584   0.597   0.5518

Sigma Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.1386     0.1531   0.906    0.367
```

Misturas finitas

Misturas finitas podem ser ajustadas usando o pacote `gamlss.mx`. Uma mistura finita tem a forma

$$f(y|\boldsymbol{\psi}) = \sum_{\kappa=1}^{K} \pi_\kappa f_\kappa(y|\boldsymbol{\theta}_\kappa) \tag{2}$$

em que $f_\kappa(y|\boldsymbol{\theta}_\kappa)$ é a função (densidade) de probabilidade de y para o componente κ, para $\kappa = 1, 2, \ldots, K$.

Ainda, $\sum_{\kappa=1}^{K} \pi_\kappa = 1$ e $\boldsymbol{\psi} = (\boldsymbol{\theta}, \boldsymbol{\pi})$ em que $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K)$ e $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_k)$.

Qualquer combinação (contínua ou discreta) de distribuições em `gamlss.family` pode ser usada.

Misturas finitas

O modelo é ajustado usando o algoritmo EM.

As funções (densidade) de probabilidade:

- Podem ter diferentes parâmetros, ajustadas via `gamlssMX()`, ou
- Podem ter parâmetros em comum, ajustadas via `gamlssNP()`.

No primeiro caso, as probabilidades de mistura podem ser modeladas usando variáveis explicativas enquanto no segundo elas são assumidas constantes (não dependem de variáveis explicativas).

Distribuições específicas estão disponíveis em `gamlss.dist` e serão estudadas no Módulo 7.

Exemplo

A seguir, um exemplo de ajuste de uma mistura finita de duas distribuições Gumbel reversas aos dados `enzyme`.

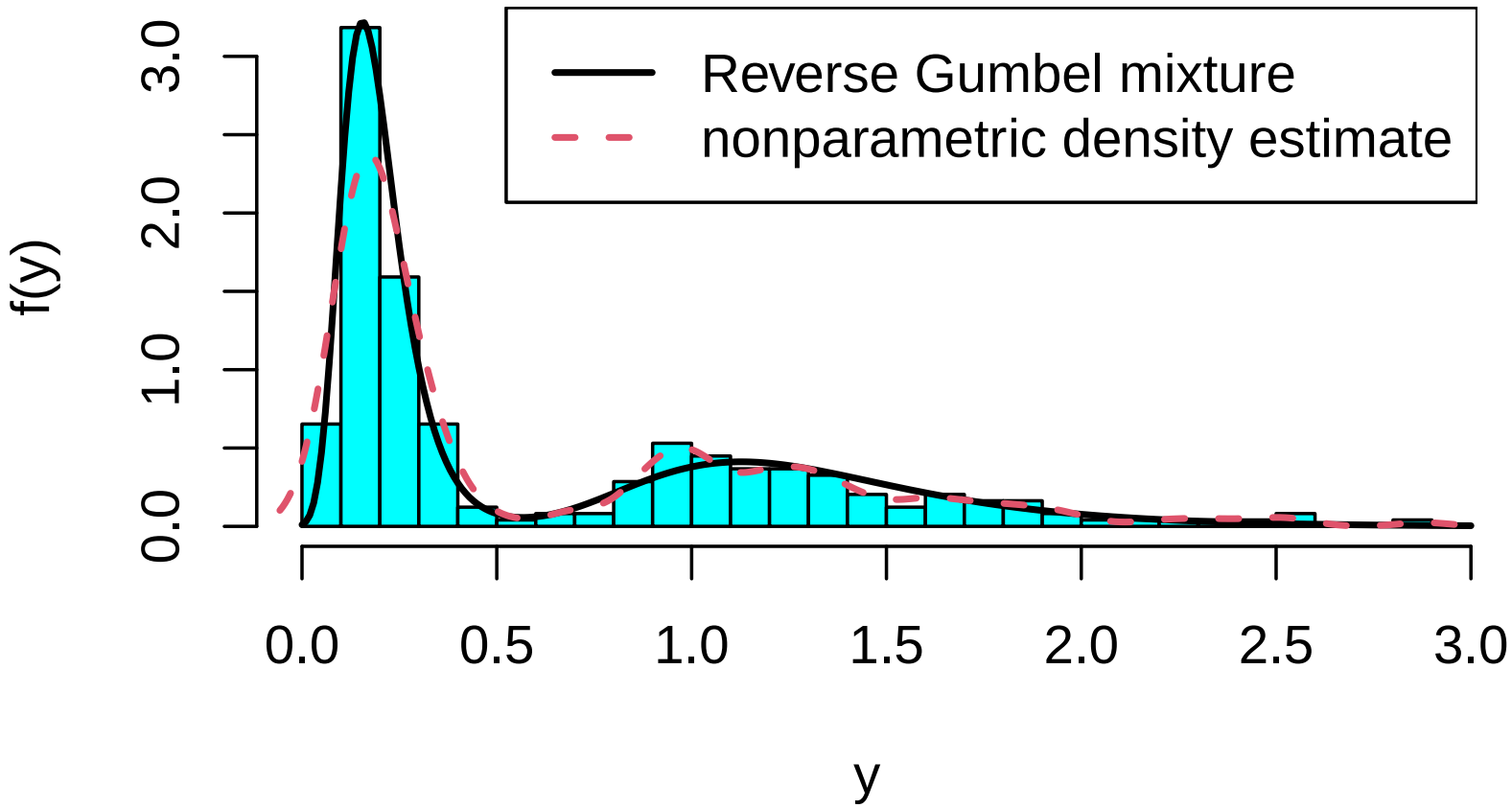
```
library(gamlss.mx)
data(enzyme)
#help("enzyme")
```

A distribuição resultante da mistura finita, obtida via `getpdfMX()`, é comparada com uma estimativa não paramétrica, obtida via `density()`.

Exemplo

```
m3 <- gamlssMX(act ~ 1, data = enzyme, family = RG, K = 2)
truehist(enzyme$act, h = 0.1, xlab="y", ylab="f(y)")
fnRG <- getpdfMX(m3, observation=1)
lines(seq(0, 3, 0.01), fnRG(seq(0, 3, 0.01)), lty = 1, lwd=2)
lines(density(enzyme$act, width ="SJ-dpi"),lty=2,lwd=2,col=2)
legend("topright", legend=c("Reverse Gumbel mixture",
"nonparametric density estimate"), lty=1:2, lwd=2,col=c(1,2))
```

Exemplo


Reverse Gumbel mixture
nonparametric density estimate
f(y)
0.0
1.0
2.0
3.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
y