6. Modelos da Distribuição Gaussiana

Os modelos Gaussianos são modelos baseados na distribuição Normal, que designaremos por **distribuição Gaussiana** em homenagem ao grande astrônomo e matemático *Johann Carl Friedrich Gauss* (1777 – 1855).

Uma ampla gama de modelos cabe sob essa designação: modelos lineares clássicos, isto é modelos de regressão e análise de variância, modelos não lineares e modelos geoestatísticos clássicos.

Geralmente, esses modelos são apresentados em classes diferentes, mas nesse tópico procuraremos unificar a sua apresentação sob a abordagem de verossimilhança.

Conceitos

- Distribuição gaussiana: propriedades e parâmetros
- Variáveis-resposta gaussianas
- Modelos gaussianos de resposta linear da média
    - Regressão linear
    - Análise de variância
- Modelos gaussianos de resposta não-linear da média
    - Regressão não-linear
- Modelos gaussianos com resposta da variância

Tutoriais

Utilizaremos nesse tutorial dados de biomassa de árvores de *Eucalyptus saligna* do arquivo esaligna.csv.

Modelo com a média constante

Para ajustar os modelos utilizaremos a função “mle2” que está no pacote “bbmle”. Portanto é necessário “carregar” esse pacote.

```
library(bbmle)
help(mle2)
```

O modelo Gaussiano mais simples é aquele em que pretendemos apenas estimar os parâmetros média e variância, que consideramos contantes. Podemos representar este modelo assim:

$$ B \sim N(\mu = \beta , \ \sigma = \alpha) $$

Que lemos:

> "a variável B segue uma distribuição normal com parâmetros μ e σ constantes com valores β e α, respectivamente".

OU

> "A variável gaussiana B tem parâmetros constantes $\mu = \beta $ e $\sigma = \alpha$."

Vamos ajustar este modelo a dados de *biomassa total* de uma amostra de árvores de *E. saligna* (variável "total"). Sabemos que os MLEs da média e desvio-padrão de um Gaussiana são as média e desvio-padrão amostrais. Então o ajuste é simplesmente calcular estas quantidades. E com elas podemos calcular a log-verossimilhança negativa deste ajuste:

```
## Leitura dos dados
esa <- read.csv("esaligna.csv",header=T)
mean(esa$total)
[1] 93.20056
sd(esa$total)
[1] 83.51936
## Log-Verossimilhança negativa
-sum( dnorm( esa$total, mean = mean(esa$total), sd = sd(esa$total),
log=TRUE) )
[1] 209.8846
```

Portanto esse modelo tem como estimativas (MLEs) dos parâmetros: média = 93 e desvio padrão = 83,5. Já a log-verossimilhança negativa é igual a 209,9.

Objetos da Classe MLE

Mesmo sendo um modelo simples, podemos ajustá-lo com otimização computacional utilizando a função `mle2` (*Maximum Likelihood Estimator*), do pacote "bbmle" do R.

Para isso é criamos uma função em R que calcula a log-verossimilhança negativa da distribuição Gaussiana (Normal) sobre os dados que possuímos:

```
nllikGauss <- function(m=90, s=80) -sum(stats::dnorm(esa$total, m, s,
log=TRUE))
```

Para ajustar o modelo com média e desvio padrão constante entregamos esta função ao otimizador executado pela `mle2`:

```
gauss01 <- mle2( nllikGauss )
```

```
class( gauss01 )
[1] "mle2"
attr(,"package")
[1] "bbmle"
```

O objeto gerado pela função `mle2` é um objeto de classe `mle2` e pode ser utilizado diretamente em algumas funções como: `summary` e `logLik`.

```
summary(gauss01)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss)

Coefficients:
  Estimate Std. Error z value     Pr(z)
m  93.1309    13.7362  6.7800 1.202e-11 ***
s  82.4172     9.7246  8.4751 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 419.7551
```

Note que a função `summary` apresenta as *estimativas* juntamente com os respectivos *erros padrões* destas estimativas.

Verossimilhança Perfilhada

A função `profile` gera a verosimilhança perfilhada para os parâmetros do modelo, mas utiliza uma transformação da verossimilhança. Se utilizada diretamente com a função `plot`, gera gráficos da verossimilhança transformado para uma variável normal padronizada (z-score). Esta transformação usa a propriedade de normalidade assintótica dos MLEs para supor então que as estimativas dos parâmetros seguem uma distribuição normal. Sob esta premissa, é possível definir intervalos de confiança dos parâmetros, que são indicados na figura:

```
> par(mfrow=c(1,2))
> plot( profile(gauss01))
```

Intervalos de confiança são um conceito de inferência de testes de significância. Na abordagem de inferência por verossimilhança a incerteza sobre a estimativa é expressa por intervalos de plausibilidade. Estes intervalos devem convergir para os intervalos de confiança descritos acima, quando a amostra for grande o suficiente. Mas os intervalos de plausibilidade são mais gerais, pois valem mesmo quando a aproximação normal dos intervalos de confiança ainda não funciona.

A função `plotprofmle`, que está no pacote “sads”, faz as curvas de verossimilhança perfilhada e delimita os intervalos de plausibilidade para os parâmetros estimados pela função `mle2`:

```
> library(sads) # primeiro instale o pacote se não o tiver
> par(mfrow=c(1,2))                    # Múltiplos gráficos na mesma janela:
```

```
1 linha x 2 colunas
> plotprofmle( profile(gauss01) )
```

Os gráficos apresentam a log-verossimilhança negativa relativa para os dois parâmetros ajustados: média e desivo padrão. Em cada gráfico, é apresentado também o **intervalo de verossimilhança** para razão de 8 (diferença de log-verossimilhança de $\ln(8)$).

Modelo com Média como Função Linear

Nesse caso da *biomassa total* das árvores de *E. saligna* (variável `total`), faz sentido assumir que a biomassa das árvores seja uma função linear do *tamanho da árvore*, definido por uma expressão simples do diâmetro e da altura da árvore: $$ E[b_i] = \beta_0 + \beta_1 (d_i^2\ h_i)$$.

Assim, a média deixa de ser modelada como *constante* e passa a ser modelada como uma *função linear* de uma variável preditora. E nosso modelo pode ser expresso assim:

$$ B \sim N(\mu = \beta_0 + \beta_1 (d_i^2\ h_i), \ \sigma = \alpha) $$

Que lemos:

> "a variável B segue distribuição normal com parâmetro μ que é uma função linear da preditora $(d_i^2\ h_i)$ e parâmetro σ constante.

Função de Log-Verossimilhança Negativa

Para podermos ajustar um modelo onde a média é uma função linear, precisamos definir uma nova função de Log-Verossimilhança Negativa:

```
nllikGauss2 <- function(b0=0, b1=1, s=80)
{
    m <- b0+b1*(esa$dap^2*esa$ht)
     -sum(stats::dnorm(x=esa$total, mean=m, sd=s,log=T))
 }
```

Note que temos agora três parâmetros: β_0 e β_1 definem a função linear da média, como função da variável combinada "`(esa$dap^2*esa$ht)`", enquanto que o desvio padrão (σ) permanece **constante**. Os valores *default* destes parâmetros na função "`nllikGauss2`" são usados como "palpites" iniciais pela função de ajuste "`mle`".

Ajuste e Análise do Modelo

O ajuste do modelo é realizado da mesma forma pela função "`mle`":

```
gauss02 <- mle2( nllikGauss2 )
Warning message:
NaNs produced in: dnorm(x, mean, sd, log)
```

Podemos agora analisar o modelo:

```
summary( gauss02 )
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss2)

Coefficients:
     Estimate Std. Error z value   Pr(z)
b0 11.4859246  5.9655410  1.9254 0.05418 .
b1  0.0263826  0.0013888 18.9973 < 2e-16 ***
s  24.8058770  2.9225990  8.4876 < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 333.3746

logLik(gauss02)
'log Lik.' -166.6873 (df=3)
```

E analisar as curvas de verossimilhança perfilhada:

```
par(mfrow=c(2,2))
plotprofmle( profile(gauss02) )
```

Note que quando a média é modelada como uma função linear, o valor do desvio padrão se torna bem menor. O que essa redução indica?

Comparação com o Modelo Anterior

Note que ao assumir a média como função linear do *tamanho das árvores* houve clara melhora na qualidade do modelo, conforme o valor de Log-Verossimilhança mostra:

```
logLik(gauss01)
'log Lik.' -209.8776 (df=2)

logLik(gauss02)
'log Lik.' -166.6873 (df=3)

AICtab( gauss01, gauss02, base=TRUE, logLik=TRUE )

        logLik AIC    dLogLik dAIC   df
```

```
gauss02 -166.7  339.4   43.2     0.0 3
gauss01 -209.9  423.8    0.0    84.4 2
```

Comparação com a Forma Clássica de Ajuste

As estimativas dos coeficientes de regressão que obtivemos acima são semelhantes aos obtidos pelo forma clássica de ajuste de modelos lineares de regressão:

```
>
> summary( gauss02 )
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss2)

Coefficients:
     Estimate Std. Error z value   Pr(z)
b0 11.4859246  5.9655410  1.9254 0.05418 .
b1  0.0263826  0.0013888 18.9973 < 2e-16 ***
s  24.8058770  2.9225990  8.4876 < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 333.3746


> summary( lm( total ~ I(dap^2*ht), data=esa ) )$coefficients
                Estimate  Std. Error   t value     Pr(>|t|)
(Intercept)   11.51744130 6.139609524  1.875924 6.927291e-02
I(dap^2 * ht)  0.02637721 0.001429276 18.454952 2.736869e-19
>
```

Questões:

- Quanto aos erros-padrão da estimativa (`Std. Error`), as duas formas de ajuste geram valores iguais?
- Os Intervalos de Confiança de 95% (use valor de t=2) são semelhantes aos **Intervalos de Verossimilhança** para razão 8 ? (Veja os gráficos da verossimilhança perfilhada)
- Por que a abordagem tradicional não apresenta uma estimativa de erro-padrão para o parâmetro de escala do modelo (desvio padrão)?

Modelo com Média e Desvio Padrão como Função Linear

Se observarmos a relação entre *biomassa total* e o diâmetro (`dap`) e altura das árvores (`ht`) veremos que não só a média mas também o desvio padrão cresce com o aumento do tamanho da árvore:

```
par(mfrow=c(1,1))
plot( total ~  I(dap^2*ht) , data = esa )
```

Função de Log-Verossimilhança Negativa e Ajuste

Assim podemos ajustar um modelo onde média e desvio padrão são funções do tamanho da árvore. Uma possibilidade é:

$$ B \sim N(\mu = \beta_0 + \beta_1 (d_i^2\ h_i), \ \sigma = \alpha_0 (d_i^2\ h_i)^{\alpha_1}) $$

Ou seja:

> "B é uma variável Gaussiana com parâmetro μ que é uma função linear da preditora $(d_i^2\ h_i)$, e parâmetro σ que é uma função de potência desta mesma preditora"

Para ajustar este modelo com a `mle2`, precisamos construir nova função de log-verossimilhança negativa, em que a média e desvio-padrão sejam as funções definidas da preditora:

```
nllikGauss3 <- function(b0=11.5, b1=0.0264, a0 = 1, a1 = 10)
 {
     m <- b0+b1*(esa$dap^2*esa$ht)
     s <-  exp(a0)*(esa$dap^2*esa$ht)^a1
     -sum(stats::dnorm(x=esa$total, mean=m, sd=s,log=T))
 }

gauss03 <- mle2( nllikGauss3 )

summary(gauss03)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss3)

Coefficients:
     Estimate Std. Error z value     Pr(z)
b0  8.0863364  3.1124317  2.5981  0.009375 **
b1  0.0274520  0.0016625 16.5128 < 2.2e-16 ***
a0 -0.5069408  0.8396233 -0.6038  0.545995
a1  0.4660469  0.1108618  4.2039 2.624e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

-2 log L: 317.2816

AIC(gauss03)
```

```
[1] 325.2816
```

E podemos analisar a curva de verossimilhança perfilhada para os parâmetros:

```
par(mfrow=c(2,2))
plotprofmle( profile( gauss03 ) )
```

Modelo com uma outra Função para o Desvio Padrão

Podemos analisar a importância da função que descreve o desvio padrão tentando uma outra função. No exemplo abaixo, temos uma função que relaciona linearmente o desvio padrão com a variável preditora:

$$ B \sim N(\mu = \beta_0 + \beta_1 (d_i^2\ h_i), \ \sigma = \alpha0 + \alpha_1 (d_i^2\ h_i)) $$

ou seja:

> "B é uma variável Gaussiana com parâmetros μ e σ que são funções lineares da preditora $(d_i^2\ h_i)$."

Vamos então fazer o ajuste deste novo modelo no R:

```
> nllikGauss3b <- function(b0=11.5, b1=0.0264, a0 = 1, a1 = 0.5)
+ {
+     m <- b0+b1*(esa$dap^2*esa$ht)
+     s <-  a0 + a1*(esa$dap^2*esa$ht)^(1/2)
+     -sum(stats::dnorm(x=esa$total, mean=m, sd=s,log=T))
+ }
>
> gauss03b = mle2( nllikGauss3b )
Warning messages:
1: In dnorm(x, mean, sd, log) : NaNs produced
2: In dnorm(x, mean, sd, log) : NaNs produced
3: In dnorm(x, mean, sd, log) : NaNs produced
4: In dnorm(x, mean, sd, log) : NaNs produced
5: In dnorm(x, mean, sd, log) : NaNs produced
6: In dnorm(x, mean, sd, log) : NaNs produced
> summary(gauss03b)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss3b)

Coefficients:
```

```
    Estimate Std. Error z value     Pr(z)
b0 8.0513173  3.1080813  2.5904 0.0095852 **
b1 0.0274884  0.0016925 16.2412 < 2.2e-16 ***
a0 0.6623036  4.3336348  0.1528 0.8785334
a1 0.4490437  0.1293511  3.4715 0.0005175 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 317.3536

> logLik(gauss03b)
'log Lik.' -158.6768 (df=4)

> AIC(gauss03b)
[1] 325.3536
>
```

Note que nesse caso houve uma certa "*reclamação*" do otimizador. Isso acontece porque a função linear que especificamos para o desvio-padrão permite valores negativos se o otimizador tentar certas combinações de intercepto e inclinação. Como o parâmetro de desvio-padrão da normal deve ser positivo, estas tentativas devolvem valor indefinido de verossimilhança (NaNs). Mas o otimizador é robusto a estes casos. Simplesmente avisa que isso aconteceu, ignora estas combinações, e segue adiante tentando outras.

Questão: será que houve mudança marcante na qualidade do ajuste?

Modelo com Função Não-Linear para Média

Uma outra possibilidade é considerarmos que a média e o desvio-padrão são funções de potência da preditora :

$$ B \sim N(\mu = \beta_0 (d_i^2\ h_i)^{\beta_1}, \ \sigma = \alpha_0 (d_i^2\ h_i)^{\alpha_1}) $$

Como antes, para ajustar mais este modelo no R criamos uma nova função de log-verossimilhança e a minimizamos com a função `mle2`:

```
> nllikGauss4 <- function(b0=-2.4, b1=0.86, a0 = 1, a1 = 10)
+ {
+     m <- exp(b0) *(esa$dap^2*esa$ht)^b1
+     s <-  exp(a0)*(esa$dap^2*esa$ht)^a1
+     -sum(stats::dnorm(x=esa$total, mean=m, sd=s,log=T))
+ }
> gauss04 = mle2( nllikGauss4)
> summary(gauss04)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss4)
```

```
Coefficients:
    Estimate Std. Error z value     Pr(z)
b0 -2.220989   0.434689 -5.1094 3.232e-07 ***
b1  0.847536   0.052097 16.2683 < 2.2e-16 ***
a0 -0.497323   0.854650 -0.5819    0.5606
a1  0.463901   0.112893  4.1092 3.970e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 316.8159

> AIC(gauss04)
[1] 324.8159
>
> plotprofmle( profile(gauss04) )
>
```

Questões:

- O que aconteceu com o ajuste do modelo ?
- O que aconteceu com a curva de verossimilhança perfilhada das estimativas dos parâmetros na relação não-linear?

Outra Função Não-Linear para Média

A relação entre a biomassa lenhosa e as duas variáveis preditoras DAP (dap) e altura (ht) pode seguir padrões não-lineares, mas não exatamente a partir da variável combinada (dap^2 * ht). Uma outra possibilidade é uma relação da biomassa lenhosa na forma de potência com as duas variáveis:

$$ B \sim N(\mu = \beta_0 \ d^{\beta_1} \ h^{\beta_2} , \ \sigma = \alpha_0 (d^2\ h)^{\alpha_1})
$$

onde d é o DAP e h é a altura. Mantemos o parâmetro σ como função de potência da preditora já usada antes, $d^2\ h$. vamos fazer o ajuste deste novo modelo no R:

```
 nllikGauss5 <- function(b0=-1.7, b1=2.44, b2=-0.097 , a0 = 1, a1 = 10)
 {
     m <- exp(b0) * esa$dap^b1 * esa$ht^b2
     s <-  exp(a0)*(esa$dap^2*esa$ht)^a1
     -sum(stats::dnorm(x=esa$total, mean=m, sd=s,log=T))
 }
gauss05 = mle2( nllikGauss5 )
summary(gauss05)

Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss5)

Coefficients:
```

```
   Estimate Std. Error z value     Pr(z)
b0 -2.10999    0.41242 -5.1162 3.118e-07 ***
b1  2.37738    0.13322 17.8453 < 2.2e-16 ***
b2  0.11704    0.11660  1.0037    0.3155
a0  1.23524    1.08020  1.1435    0.2528
a1  0.18935    0.14320  1.3222    0.1861
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 293.326

 AIC(gauss05)
[1] 303.326

 par(mfrow=c(2,3))
 plotprofmle( profile(gauss05) )
```

Questões:

- Como foi o ajuste dessa forma não-linear para a média ?
- O que aconteceu com a curva de verossimilhança perfilhada para os parâmetros do modelo da média ?
- Alguma estimativa se mostra problemática ? Quais seriam as prováveis causas desses problemas ?

Modelo Log-Normal

O fato de tanto média como desvio padrão variarem em função do tamanho das árvores sugere que a escala apropriada de modelagem da biomassa possa ser a escala logarítmica. Ou seja, que as medidas têm uma distribuição log-normal, o que faz com que a média aumente com a variância. O gráfico na escala logarítmica sugere uma relação linear com variância aproximadamente constante:

```
> par(mfrow=c(1,1))
> plot( log(total) ~  log(dap^2*ht), data=esa )
>
```

Verificando o Modelo Gaussiano através dos Resíduos

Para verificar essa possibilidade, devemos analisar (graficamente) os resíduos do modelo ajustado. Infelizmente, não temos ainda funções que calculem automaticamente *valor ajustado* e *resíduo* para modelos ajustados pela função ”`mle`“. Então vamos fazer isto passo a passo:

```
# Primeiro construir um novo 'data.frame' para conter os valores ajustados e
resíduo

head(esa)
  arvore classe talhao  dap    ht tronco sobra folha  total
1      6      c     22 19.9 21.50 183.64 20.42  8.57 212.64
```

```
2      8      b     23 12.4 15.74  42.29  6.58  2.52  51.40
3      7      c     32 16.5 11.74  60.61 11.35 48.52 120.49
4      8      a     32  9.0  7.72  12.28  9.99 27.67  49.95
5      9      a     32  7.0  6.55  11.86  7.97  7.76  27.61
6      9      b     32 10.5  8.79  26.10  7.48 23.36  56.95
```

```
esa2 <- esa[, c("arvore","dap","ht","total")]
summary( gauss03b )
coef( gauss03b )
        b0         b1         a1
7.93781767 0.02753308 0.46753874
```

```
g3b.coef <- coef( gauss03b )
esa2$total.fit <- g3b.coef["b0"] + g3b.coef["b1"] * esa2$dap^2 * esa2$ht
esa2$total.res <- esa2$total - esa2$total.fit


# Gráfico de dispersão do resíduo contra valor ajustado

par(mfrow=c(1,1))
plot( esa2$total.fit, esa2$total.res )
abline(h=0, col="red", lty=2)
lines(lowess( esa2$total.fit, esa2$total.res ) )


# Gráfico Quantil-Quantil (para Normal) dos resíduos

qqnorm( esa2$total.res )
qqline( esa2$total.res )
```

O gráfico quantil-quantil sugere que os resíduos ainda não se comportam adequadamente sob a distribuição Gaussiana.

Ajuste do Modelo Log-Normal

Podemos ajustar o modelo Log-Normal, tomando a variável combinada (dap^2 * ht) como preditora na função do parâmetro μ da logn-normal.

> Lembre que os parâmetros μ e σ da distribuição log-normal correspondem à média e desvio-padrão dos **logaritmos** da variável.

Nosso modelo pode agora ser expresso assim:

$$ B \sim LN(\mu = \beta_0 + \beta_1 (d^2\ h), \ \sigma = \alpha) $$

Ou seja:

> "B é uma variável **log-normal** com parâmetro μ que é uma função linear de $(d^2\ h)$ e parâmetro σ que é uma constante"

Para ajustar este modelo criamos uma função de log-verossimilhança usando a função de densidade da log-normal, ao invés da densidade normal que usamos até agora:

```
> sd(log(esa2$total))
[1] 1.040294
>
> nllikLGauss <- function(b0=0, b1=1,  slog=1)
+ {
+         mlog <- b0+b1*log(esa$dap^2*esa$ht)
+         -sum(stats::dlnorm(esa$total,mlog,slog,log=T))
+ }
>
> lgauss01 <- mle2( nllikLGauss )
There were 16 warnings (use warnings() to see them)
> warnings()[1]
$`NaNs produced`
dlnorm(x, meanlog, sdlog, log)

> summary(lgauss01)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikLGauss)

Coefficients:
      Estimate Std. Error z value     Pr(z)
b0   -2.360370   0.374313 -6.3059 2.866e-10 ***
b1    0.859640   0.049362 17.4150 < 2.2e-16 ***
slog  0.334126   0.039378  8.4852 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

-2 log L: 317.4077
>
> par(mfrow=c(2,2))
> plotprofmle( profile( lgauss01 ) )
There were 50 or more warnings (use warnings() to see the first 50)
>
```

Mas podemos as variáveis usar as variáveis DAP e altura como duas variáveis preditoras na função preditora da média:

```
> nllikLGauss2 <- function(b0=0, b1=2, b2=1,  slog=1)
+ {
+         mlog <- b0+b1*log(esa$dap) +b2*log(esa$ht)
+         -sum(stats::dlnorm(esa$total,mlog,slog,log=T))
+ }
> lgauss02 = mle2( nllikLGauss2 )
There were 23 warnings (use warnings() to see them)
> summary(lgauss02)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikLGauss2)

Coefficients:
      Estimate Std. Error z value     Pr(z)
b0   -1.705500   0.323898 -5.2656 1.398e-07 ***
b1    2.438150   0.169665 14.3704 < 2.2e-16 ***
b2   -0.096743   0.204594 -0.4729    0.6363
slog  0.261749   0.030848  8.4852 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 299.8304
> par(mfrow=c(2,2))
> plotprofmle( profile( lgauss02 ) )
There were 50 or more warnings (use warnings() to see the first 50)
>
```

\

Comparação Geral dos Modelos

Façamos uma comparação geral dos modelos justados, usando o Critério de Informação de Akaike , o AIC [1)]

Nome do Modelo	Parâmetro	Função	AIC
gauss01	média	$$\mu$$	
	desvio padrão	$$\sigma$$	423.8
gauss02	média	$$\beta_0 + \beta_1 (d^2 h)$$	
	desvio padrão	$$\sigma$$	339.4
gauss03	média	$$\beta_0 + \beta_1 (d^2 h)$$	
	desvio padrão	$$\alpha_0 (d^2 h)^{\alpha_1}$$	325.3
gauss03b	média	$$\beta_0 + \beta_1 (d^2 h)$$	
	desvio padrão	$$\alpha_0 + \alpha_1 (d^2 h)^{(1/2)}$$	325.4
gauss04	média	$$ \beta_0(d^2 h)^{\beta_1} $$	
	desvio padrão	$$\alpha_0 (d^2 h)^{\alpha_1}$$	324.8

Nome do Modelo	Parâmetro	Função	AIC
gauss05	média	$$\beta_0 (d)^{\beta_1}\ (h)^{\beta_2}$$	
	desvio padrão	$$\alpha_0 (d^2 h)^{\alpha_1}$$	303.3
lgauss01	média	$$\beta_0 + \beta_1 \ln(d^2h)$$	
	desvio padrão	$$\sigma$$	323.4
lgauss02	média	$$\beta_0 + \beta_1\ln(d) + \beta_2\ln(h)$$	
	desvio padrão	$$\sigma$$	307.8

Questões:

- Qual a importância da ***forma funcional*** do modelo que descreve a média?
- Qual a importância da ***forma funcional*** do modelo que descreve o desvio padrão?
- Existe diferença quando se muda da distribuição Gaussiana para a LogNormal ?
- A diferença de distribuição independe da forma funcional do modelo da média ?

Exemplo do Princípio da Parcimônia

Trabalhamos em todos os modelos até esse ponto com duas variáveis preditoras: DAP (dap) e altura (ht). Mas qual o desempenho de um modelo mais simples com apenas o DAP como variável preditora. O poder explicativo desse modelo mais simples será tão bom quanto o do modelo com ambas variáveis?

Primeiro a definição da função de log-verossimilhança negativa para o modelo com apenas o DAP baseado na distribuição Gausssiana:

```
> nllikGauss6 <- function(b0=-1.7, b1=2.44, a0 = 1, a1 = 10)
+ {
+     m <- exp(b0) * esa$dap^b1
+     s <-  exp(a0)*esa$dap^a1
+     -sum(stats::dnorm(x=esa$total, mean=m, sd=s,log=T))
+ }
>
> gauss06 = mle2( nllikGauss6 )
> summary(gauss06)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikGauss6)

Coefficients:
   Estimate Std. Error z value     Pr(z)
b0 -1.93689    0.35830 -5.4058 6.452e-08 ***
b1  2.43132    0.12336 19.7098 < 2.2e-16 ***
a0  1.15232    0.93431  1.2333   0.21745
a1  0.61590    0.37430  1.6455   0.09987 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
-2 log L: 294.9407

> par(mfrow=c(2,2))
> plotprofmle( profile(gauss06) )
>
> AICtab(gauss06, gauss05, base=TRUE, logLik=TRUE)
        logLik AIC    dLogLik dAIC   df
gauss06 -147.5  302.9    0.0     0.0 4
gauss05 -146.7  303.3    0.8     0.4 5
>
```

Vejamos agora no caso da distribuição LogNormal:

```
> nllikLGauss3 <- function(b0=0, b1=2,  slog=1)
+ {
+         mlog <- b0+b1*log(esa$dap)
+         -sum(stats::dlnorm(esa$total,mlog,slog,log=T))
+ }
> lgauss03 = mle2( nllikLGauss3)
Warning messages:
1: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
2: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
3: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
4: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
5: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
6: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
7: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
8: In dlnorm(x, meanlog, sdlog, log) : NaNs produced
> summary(lgauss03)
Maximum likelihood estimation

Call:
mle2(minuslogl = nllikLGauss3)

Coefficients:
      Estimate Std. Error z value     Pr(z)
b0   -1.795293   0.263208 -6.8208 9.052e-12 ***
b1    2.374943   0.104812 22.6591 < 2.2e-16 ***
slog  0.262563   0.030944  8.4851 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

-2 log L: 300.0533

> plotprofmle( profile(lgauss03) )
There were 50 or more warnings (use warnings() to see the first 50)
>
> AICtab(lgauss03, lgauss02, base=TRUE, logLik=TRUE)
         logLik AIC    dLogLik dAIC   df
lgauss03 -150.0  306.1    0.0     0.0 3
```

```
lgauss02 -149.9  307.8    0.1     1.8 4
```

Questão:

- Como os modelos mais simples (apenas com o DAP) se compara aos modelos análogos mais complexos?
- A variável altura (ht) é uma variável importante para a predição da biomassa nesse caso?

A Influência do Tamanho da Amostra

Nos exemplos acima, vimos que para vários modelos a curva de verossimilhança perfilhada se mostrou problemática para vários parâmetros. Esse problema pode ser função do tipo de dado ou do tamanho da amostra. Neste tutorial mostramos o efeito que o tamanho da amostra tem sobre a qualidade do ajuste, e portanto dos perfis de log-verossimilhança.

Vamos usar um exemplo com os modelos de ***relação funcional não-linear*** para média. Para isso utilizaremos os dados do arquivo "egrandis.csv" com dados de árvores de plantações de *Eucalyptus grandis* na região central do Estado de São Paulo.

```
> euca.egr = read.table("egrandis.csv", header=T, as.is=TRUE, sep=";")
> head( euca.egr )
> dim( euca.egr )
```

Ajustaremos em que o volume de madeira das árvores ("vol") é uma variável Gaussiana com média como uma função de potência do DAP ("dap") e altura ("ht") , e desvio-padrão como uma função de potência de $(dap^2 ht)$:

$$Y \ \sim \ N(\mu=\beta_0 \mathrm{dap}^{\beta_1} \mathrm{ht}^{\beta_2} , \ \sigma = \alpha_0 (dap^2 ht)^{\alpha_1} $$

Assumindo o conjunto completo do dados como uma "população", vamos gerar *sub*-conjuntos de dados, para representar amostras de diferentes tamanhos: 100, 150, 200, 500 e 1000 árvores:

```
> N = dim( euca.egr )[1]
> egr.100 = euca.egr[ sample(1:N, 100), ]
> egr.150 = euca.egr[ sample(1:N, 150), ]
> egr.200 = euca.egr[ sample(1:N, 200), ]
> egr.500 = euca.egr[ sample(1:N, 500), ]
> egr.1000 = euca.egr[ sample(1:N, 1000), ]
```

E então criamos uma função de log-verossimilhança negativa para o modelo:

```
> nllikGauss5.egr <- function(b0=-1.7, b1=2.44, b2=-0.097 , a0 = 1, a1 = 10)
+ {
+     m <- exp(b0) * dap^b1 * ht^b2
+     s <-  exp(a0)*(dap^2*ht)^a1
+     -sum(stats::dnorm(x=vol, mean=m, sd=s,log=T))
+ }
```

E ajustamos o modelo e gerando os perfis dos parâmetros para a amostra $n = 100$:

```
> dap = egr.100$dap
> ht = egr.100$ht
> vol = egr.100$vol
> gauss.100 = mle2( nllikGauss5.egr )
> gauss.100.prof = profile( gauss.100 )
```

Ajustando o modelo e gerando os perfis dos parâmetros para a amostra $n = 150$:

```
> dap = egr.150$dap
> ht = egr.150$ht
> vol = egr.100$vol
> gauss.150 = mle2( nllikGauss5.egr )
> gauss.150.prof = profile( gauss.150 )
```

Ajustando o modelo e gerando os perfis dos parâmetros para a amostra $n = 200$:

```
> dap = egr.200$dap
> ht = egr.200$ht
> vol = egr.200$vol
> gauss.200 = mle2( nllikGauss5.egr )
> gauss.200.prof = profile( gauss.200 )
```

Ajustando o modelo e gerando os perfis dos parâmetros para a amostra $n = 500$:

```
> dap = egr.500$dap
> ht = egr.500$ht
> vol = egr.500$vol
> gauss.500 = mle2( nllikGauss5.egr )
> gauss.500.prof = profile( gauss.500 )
>
```

Ajustando o modelo e gerando os perfis dos parâmetros para a amostra $n = 1000$:

```
> dap = egr.1000$dap
> ht = egr.1000$ht
> vol = egr.1000$vol
> gauss.1000 = mle2( nllikGauss5.egr )
> gauss.1000.prof = profile( gauss.1000 )
>
```

Ajustando o modelo e gerando os perfis dos parâmetros para a “população” $n = N$:

```
> dap = euca.egr$dap
> ht = euca.egr$ht
> vol = euca.egr$vol
> gauss.euca = mle2( nllikGauss5.egr )
> gauss.euca.prof = profile( gauss.euca )
```

Produzindo os gráficos do verossimilhança perfilhada para todos os parâmetros, para modelos ajustados a cada amostra:

```
> par(mfrow=c(2,3))
> plotprofmle( gauss.100.prof )
> par(mfrow=c(2,3))
> plotprofmle( gauss.150.prof )
> par(mfrow=c(2,3))
> plotprofmle( gauss.200.prof )
> par(mfrow=c(2,3))
> plotprofmle( gauss.500.prof )
> par(mfrow=c(2,3))
> plotprofmle( gauss.1000.prof )
> par(mfrow=c(2,3))
> plotprofmle( gauss.euca.prof )
```

Focalizando especificamente na MLE do parâmetro β_1 (segundo parâmetro):

```
> par(mfrow=c(2,3))
> plotprofmle( gauss.100.prof , which=2); title(main="n=100")
> plotprofmle( gauss.150.prof , which=2); title(main="n=150")
> plotprofmle( gauss.200.prof , which=2); title(main="n=200")
> plotprofmle( gauss.500.prof , which=2); title(main="n=500")
> plotprofmle( gauss.1000.prof , which=2); title(main="n=1000")
> plotprofmle( gauss.euca.prof , which=2); title(main="n=N")
```

Focalizando especificamente na MLE do parâmetro β_2 (terceiro parâmetro):

```
> par(mfrow=c(2,3))
> plotprofmle( gauss.100.prof , which=3); title(main="n=100")
> plotprofmle( gauss.150.prof , which=3); title(main="n=150")
> plotprofmle( gauss.200.prof , which=3); title(main="n=200")
> plotprofmle( gauss.500.prof , which=3); title(main="n=500")
> plotprofmle( gauss.1000.prof , which=3); title(main="n=1000")
> plotprofmle( gauss.euca.prof , which=3); title(main="n=N")
```

Questões:

- Qual a influência do tamanho da amostra no valor da MLE obtida?
- Qual a influência do tamanho da amostra sobre a curva de verossimilhança perfilhada? Como isso deve ser interpretado em termos de qualidade da estimatição?
- A influência do tamanho da amostra é a mesma sobre as MLE de todos os parâmetros?

Modelo da ANOVA

Neste tutorial **opcional** descrevemos e ajustamos o modelo estatístico que está por trás da análise de variância.

Usaremos os dados do exemplo 12.1 de Zar (1999)[2)], que são medidas de concentração de cálcio

plasmático em aves machos e fêmeas, metade das quais foi sorteada para receber um tratamento com um hormônio:

```
calcium <- c(16.5,18.4,12.7,14,12.8,
             14.5,11,10.8,14.3,10.0,
             39.1,26.2,21.3,35.8,40.2,
             32.0,23.8,28.8,25.0,29.3)
hormonio <- factor(rep(c("NO","YES"),each=10))
sexo <- factor(rep(c("F","M","F","M"),c(5,5,5,5)))
```

Este experimento tem dois fatores (sexo e tratamento com hormônio), cada um com dois níveis. Na abordagem de teste de significância, uma ANOVA é usada para testar as seguintes hipóteses nulas a respeito dos efeitos dos fatores sobre a variável resposta, que é a concentração de cálcio plasmático:

1. Não há efeito do sexo da ave;
2. Não há efeito do tratamento com o hormônio;
3. Não há interação entre os dois fatores acima.

Especificando o Modelo

Como todo teste de significância, a ANOVA tem um modelo subjacente, que fica evidente se lembramos de duas de suas premissas:

1. As amostras vêm de populações estatísticas com variâncias iguais;
2. As amostras vêm de populações estatísticas de medidas com distribuição normal.

Portanto, a variável-resposta está descrita neste modelo como uma variável normal, cuja média é afetada pelos fatores, e a variância (e portanto o desvio-padrão) é constante. Como na variável normal média e desvio-padrão correspondem aos parâmetros, nosso modelo pode ser representado assim:

$$ Y \ \sim \ N(\mu=f(X_i)\ , \ \sigma=c)$$

Que lemos:

> "Y é uma variável Gaussiana, com parâmetro μ que é uma função de preditoras X_i, e parâmetro σ constante."

Modelo de Médias

Uma das maneira de representar o efeito dos fatores sobre o valor esperado da variável-resposta é substituir μ na expressão acima pela média do grupo experimental de cada observação.

Como temos um experimento com dois fatores de dois níveis cada, temos quatro grupos:

```
> intval <- interaction(hormonio,sexo)
> intval
 [1] NO.F  NO.F  NO.F  NO.F  NO.F  NO.M  NO.M  NO.M  NO.M  NO.M  YES.F YES.F
[13] YES.F YES.F YES.F YES.M YES.M YES.M YES.M YES.M
Levels: NO.F YES.F NO.M YES.M
```

Cujas médias são:

```
> medias <- tapply(calcium,intval,mean)
> medias
 NO.F YES.F  NO.M YES.M
14.88 32.52 12.12 27.78
```

Definindo $Y_{ij\cdot}$ como os valores do grupo experimental definido pelo nível i do primeiro fator e nível j do segundo fator, nosso modelo torna-se:

$$Y_{ij\cdot}\ \sim \ N(\mu=E[Y_{ij\cdot}]\ , \ \sigma=c)$$

Onde $E[Y_{ij\cdot}]$ é o valor esperado, ou a média, de Y em cada grupo. Podemos definir uma função de verossimilhança para este modelo da seguinte forma:

```
LL.m3a <- function(NO.F,YES.F,NO.M,YES.M,sigma){
  intval <- interaction(hormonio,sexo)
  media <- c(NO.F,YES.F,NO.M,YES.M)[intval]
  -sum(dnorm(calcium,mean=media,sd=sigma,log=T))
}
```

Detalhes sobre esta função estão no fim desta seção.

Agora podemos minimizar esta função com o `mle2`, usando como valores iniciais as médias dos grupos e o desvio-padrão geral:

```
library("bbmle") # carrega o pacote, desde que instalado
m3a <- mle2(LL.m3a,start=c(as.list(medias),sigma=sd(calcium)))
```

Os valores estimados das médias dos grupos correspondem aos valores iniciais, pois os MLEs das médias dos grupos são as médias amostrais:

```
> coef(m3a)
    NO.F    YES.F     NO.M    YES.M    sigma
14.88000 32.52000 12.12000 27.78000  4.28002
> medias
 NO.F YES.F  NO.M YES.M
14.88 32.52 12.12 27.78
```

Vamos inspecionar os perfis de verossimilhança destas estimativas, colocando-os todos no mesmo gráfico. Para isto baixe o código da função plot.prof.aov para seu diretório de trabalho e carregue-o em sua área de trabalho:

```
source("plot-prof-aov.r")
```

E agora podemos avaliar os perfis de verossimilhança dos valores esperados para cada grupo com

```
m3a.prof <- profile(m3a)
plot.prof.aov(m3a.prof,which=1:4)
```

Os intervalos de plausibilidade de machos e fêmeas sob o mesmo tratamento de hormônio estão sobrepostos, sugerindo que não haja efeito do sexo, apenas da aplicação do hormônio. Para avaliar isto ajustamos um novo modelo, sem efeito do sexo:

```
##função de verossimilhança
LL.m2a <- function(NO,YES,sigma){
  media <- c(NO,YES)[hormonio]
  -sum(dnorm(calcium,mean=media,sd=sigma,log=T))
}
##Ajuste
m.horm <- tapply(calcium,hormonio,mean)
m2a <- mle2(LL.m2a,start=list(NO=m.horm[1],YES=m.horm[2],sigma=sd(calcium)))
```

Os valores estimados correspondem às médias dos grupos:

```
> m.horm
   NO   YES
13.50 30.15
> coef(m2a)
      NO      YES    sigma
13.49998 30.14997  4.69880
```

Não há sobreposição nos intervalos, indicando diferenças nos valores esperados das concentrações de cálcio dos dois grupos, segundo este modelo:

```
m2a.prof <- profile(m2a)
plot.prof.aov(m2a.prof,which=1:2)
```

Vamos ainda considerar um modelo mais simples, que descreve a hipótese de ausência de efeitos dos dois fatores:

```
LL.m1a <- function(media,sigma){
  -sum(dnorm(calcium,mean=media,sd=sigma,log=T))
}
m1a <- mle2(LL.m1a,start=list(media=mean(calcium),sigma=sd(calcium)))
```

Os resultados anteriores indicam um efeito do hormônio. Mas para representar com modelos todas as hipóteses da ANOVA vamos também ajustar o modelo apenas com efeito do sexo:

```
LL.m4a <- function(F,M,sigma){
  media <- c(F,M)[sexo]
  -sum(dnorm(calcium,mean=media,sd=sigma,log=T))
}
m.sexo <- tapply(calcium,sexo,mean)
m4a <- mle2(LL.m4a,start=list(F=m.sexo[1],M=m.sexo[2],sigma=sd(calcium)))
```

E agora comparamos os quatro modelos com o AIC corrigido para pequenas amostras:

```
> AICctab(m1a,m2a,m3a,m4a,delta=T,sort=T,weights=T,
          nobs=length(calcium))
    AICc  df dAICc weight
m2a 126.2 3    0.0 0.821
m3a 129.2 5    3.1 0.179
m1a 151.8 2   25.6 <0.001
m4a 153.8 3   27.6 <0.001
```

O modelo mais plausível é o que descreve as medidas de cálcio como variáveis aleatórias com valores esperados diferentes para o grupo controle e o que recebeu hormônio. Isto indica que não há efeito do sexo nem da interação entre sexo e hormônio. O teste F aponta para a mesma conclusão:

```
> summary(aov(calcium~hormonio*sexo))
              Df  Sum Sq Mean Sq F value    Pr(>F)
hormonio       1 1386.11 1386.11 60.5336 7.943e-07 ***
sexo           1   70.31   70.31  3.0706   0.09886 .
hormonio:sexo  1    4.90    4.90  0.2140   0.64987
Residuals     16  366.37   22.90
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Um modelo que não está na ANOVA

Podemos ir além da ANOVA não só explicitando os modelos que ela usa, como também propondo novos.

Por exemplo, podemos investigar a premissa de igualdade de variâncias com um modelo em que cada nível do fator hormônio pode ter um desvio-padrão diferente:

```
## Função para calculo do MLE do desvio-padrão
sd.mle <- function(x){sqrt((var(x)*(length(x)-1))/length(x))}
## MLEs dos desvios de cada grupo
sigma.horm <- tapply(calcium,hormonio,sd.mle)
##Função de verossimilhança
LL.m5a <- function(NO,YES,sigma.NO,sigma.YES){
  media <- c(NO,YES)[hormonio]
  sigma <- c(sigma.NO,sigma.YES)[hormonio]
  -sum(dnorm(calcium,mean=media,sd=sigma,log=T))
}
## Ajuste do modelo
m5a <- mle2(LL.m5a,start=list(NO=m.horm[1],
                    YES=m.horm[2],
                    sigma.NO=sigma.horm[1],
                    sigma.YES=sigma.horm[2]))
```

Os valores estimados dos parâmetros correspondem às estimativas obtidas das amostras, que foram usadas como valores iniciais no ajuste:

```
> coef(m5a)
       NO       YES  sigma.NO sigma.YES
13.500000 30.150000  2.486367  6.162533
> m.horm
   NO   YES
13.50 30.15
> sigma.horm
      NO      YES
2.486363 6.162508
```

E os perfis de verossimilhança mostram que a precisão das estimativas das médias melhorou um pouco

```
m5a.prof <- profile(m5a)
plot.prof.aov(m5a.prof,which=1:2)
```

E que os intervalos de plausibilidade para as estimativas dos desvios-padrão estão ligeiramente sobrepostos:

```
plot.prof.aov(m5a.prof,which=3:4,legenda="bottomright")
```

Apesar disto, o AICc indica que este modelo é mais plausível do que o anteriormente selecionado:

```
> AICctab(m1a,m2a,m3a,m4a,m5a,delta=T,sort=T,weights=T,
          nobs=length(calcium))
    AICc  df dAICc weight
m5a 122.0 4    0.0 0.8668
m2a 126.2 3    4.1 0.1094
m3a 129.2 5    7.2 0.0238
m1a 151.8 2   29.8 <0.001
m4a 153.8 3   31.8 <0.001
```

Sobre as funções de verossimilhança deste tutorial

Neste tutorial criamos funções de verossimilhança no R para modelos que têm fatores como variáveis preditoras como proposto no capítulo 9 de Bolker (2008)[3].

A ideia é usar a indexação para criar um vetor que indica um valor esperado diferente para cada grupo. Nesta função, o valor do argumento `mean` da função de densidade da normal é substituído por quatro parâmetros (`NO.F`,`YES.F`,`NO.M`,`YES.M`), que representam os valores esperados de cada grupo experimental. Isto é feito na terceira linha da função:

```
media <- c(NO.F,YES.F,NO.M,YES.M)[intval]''
```

Para entender como isto funciona, veja o que ocorre com os comandos:

```
intval <- interaction(hormonio,sexo)
intval
medias <- tapply(calcium,intval,mean)
```

```
medias
medias[intval]
data.frame(hormonio,sexo,calcium,medias[intval])
```

Modelos de Efeitos

Uma outra maneira de parametrizar o modelo da ANOVA é definir μ como uma relação linear:

$$\mu \ = \ \beta_0 + \beta_1X_1 + \beta_2X_2 + \beta_3X_1X_2$$

Onde a variável preditora X_1 é o fator hormônio, com valor $X_1=0$ para as aves que não receberam hormônio, e $X_1=1$ para as que receberam. Da mesma forma, $X_2=0$ para fêmeas $X_2=1$ para machos.

Desta maneira, o valor esperado para as fêmeas que não receberam o hormônio será o intercepto da equação linear:

$$\mu_{F.N} \ = \ \beta_0 \ + \beta_1 \times 0 \ + \beta_2 \times 0 \ + \beta_3 \times 0 \times 0 = \beta_0$$

Do mesmo modo, o valor esperado para fêmeas que receberam hormônio será

$$\mu_{F.Y} \ = \beta_0 \ + \beta_1 \times 1 \ + \beta_2 \times 0 \ + \beta_3 \times 1 \times 0 = \beta_0 + \beta_1$$

Portanto β_1 representa a diferença entre a média das fêmeas que não receberam o hormônio e a das fêmeas que receberam. Em outras palavras, é o quanto o tratamento com hormônio acrescenta ao valor esperado, ou o **efeito** deste tratamento.

Da mesma forma, β_2 é o efeito do sexo masculino. Por fim, o parâmetro β_3 permite que o efeito dos dois fatores combinados seja diferente da soma dos efeitos de cada fator, o que configura uma interação entre fatores:

$$\mu_{M.Y} \ = \beta_0 + \beta_1 + \beta_2 + \beta_3$$

Para ajustar este modelo no R usamos:

```
## Diferencas entre as medias
d.sexo <- diff(m.sexo)
d.horm <- diff(m.horm)
##Função de verossimilhança
LL.m3b <- function(intercepto,e.horm,e.sex,int,sigma){
  media <- intercepto+e.horm*(hormonio=="YES")+e.sex*(sexo=="M")+
    int*(hormonio=="YES"&sexo=="M")
  -sum(dnorm(calcium,mean=media,sd=sigma,log=T))
}
##Ajuste do modelo
m3b <-
mle2(LL.m3b,start=list(intercepto=mean(calcium[hormonio=="NO"&sexo=="F"]),
                       e.horm=d.horm,e.sex=d.sexo,int=0,
                       sigma=sd(calcium)))
```

Neste tipo de modelo, se um intervalo de verossimilhança de um parâmetro inclui o zero, é plausível que este efeito seja nulo. Verifique isto para o modelo ajustado acima com:

```
m3b.prof <- profile(m3b)
par(mfrow=c(3,2))
plotprofmle(m3b.prof)
```

A conclusões são coerentes com as obtidas com a parametrização anterior? Vamos comparar os MLEs dos mesmos modelos com os dois tipos de parametrização:

```
> cf.m3b <- coef(m3b)
> cf.m3b
intercepto     e.horm      e.sex        int      sigma
 14.880142  17.639741  -2.760148  -1.979652   4.280016
```

Para fêmeas que receberam hormônio, o modelo de efeitos prevê um valor esperado de

```
> as.numeric(cf.m3b[1]+cf.m3b[2])
[1] 32.51988
```

Você vê a correspondência entre os dois modelos? Compare com as estimativas dos parâmetros obtidas para o mesmo modelo, na seção anterior:

```
> cf.m3a <- coef(m3a)
> cf.m3a
    NO.F    YES.F     NO.M    YES.M    sigma
14.88000 32.52000 12.12000 27.78000  4.28002
```

Efeitos em Modelos no R

A parametrização dos modelos como efeitos aditivos que descrevemos acima é a usada no R para modelos lineares com variáveis preditoras contínuas ou categóricas não ordenadas.

Verifique isto comparando os coeficientes obtidos acima para o modelo com interação com os obtidos com

```
summary(lm(calcium~hormonio*sexo))
```

Notação de Fórmula da mle2

A função `mle2` aceita a mesma notação de fórmula estatística do R, como usado no comando `lm` acima.

Compare os resultados dos ajustes do modelo com interação com este novo modelo:

```
## É preciso colocar as variaveis em um dataframe
df <- data.frame(calcium=calcium,
                 hormonio=hormonio, sexo=sexo)
```

```
m3c <- mle2(calcium~dnorm(mean=media, sd=sigma),
            parameters=list(media~hormonio*sexo,sigma~1),
            start=list(media=mean(calcium),sigma=sd(calcium)),
            data = df)
```

Exercícios

Faça os exercícios 206.1 no sistema notaR.

Recursos para Estudo

Leituras

Principal

- Standard Statistics Revisited, seções 9.1 a 9.3, capítulo 9 de: Bolker, B.M. 2008 Ecological Models and Data in R Princeton: Princeton University Press.

Complementares

- Likelihood and least squares theory. Seção 1.2.2 do Cap.1 de Burnham, K. P., & Anderson, D. R. (2002). Model Selection and Multimodel Inference: A Practical-Theoretic Approach, 2nd ed. New York, Springer-Verlag.
- Apostila de Modelos Gaussianos -- 2015: Nesta apostila, que é um pouco mais técnica que a aula, os modelos são ajustados de modo mais prático e realista, isto é, utilizando as funções **`gls`** e **`gnls`** do pacote **`nlme`** (Pinheiro & Battes). Para seguir os exemplos e fazer os exercícios você precisará dos seguintes arquivos de dados:
  - sitio-florin.csv
  - dados de biomassa de árvores de E.saligna
  - Para fazer os gráficos da Verossimilhança Perfilhada, você pode utilizar a função **`plotprofmle`** do pacote **`sads`**, ou usar a função **`plot.profmle`** do seguinte arquivo-script;
    - plot-profmle.r

Uma teoria sobre a relação entre média e variância

- Taylor, L.R. Aggregation, variance and the mean. Nature, v. 189, n. 4766, p. 732, 1961.

- Lei de Taylor na Wikipedia

1)

Veremos com detalhe o uso do AIC na unidade sobre seleção de modelos. Aqui basta saber que o AIC expressa perda de informação quando o modelo é usado para expressar os dados. Então quanto menor o AIC de um modelo melhor descrição dos dados ele será, pois reteve mais informação do que foi observado.

2)

Zar, J.H. 1999. Biostatistical Analysis. 4th Ed., Prentice Hall.

3)

Bolker, B. (2008). Ecological Models and Data in R. Princeton, Princeton University Press.