

Caracterização de padrão espacial de doenças de Citrus utilizando a distribuição Beta-Binomial: Métodos e funcionalidades de estimação por máxima verossimilhança disponíveis em R

Elias Teixeira Krainski e Paulo Justiniano Ribeiro Junior

6 de maio de 2008

Resumo

O padrão espacial de doenças é estudado para se conhecer o mecanismo de atuação da doença e para orientar estratégias de controle. A distribuição beta-binomial é uma generalização da distribuição binomial quando assume-se que a incidência não é constante na região em estudo. Ela pode ser escrita de forma que θ seja um parâmetro de agregação espacial da doença. Porém, não há solução analítica para a estimação dos parâmetros. Dessa forma, devem ser utilizados procedimentos iterativos de maximização numérica da função de verossimilhança. É demonstrado o uso das funções `optimize()`, `optim()`, `nlm()` e `nlminb()` disponíveis em R para maximização numérica e das funções `fitdistr()` e `mle()` disponíveis para estimação por máxima verossimilhança. A metodologia foi aplicada a dados simulados, dados de Pinta Preta dos Citrus e de várias avaliações em tempos diferentes de Morte Súbita dos Citrus. Os resultados foram similares aos métodos tradicionais de análise por *quadrats*.

1 Introdução

Na análise do padrão espacial da incidência de doenças em plantas, costuma-se fazer a análise por *quadrat counts*, em que é estudado o número de plantas doentes por *quadrat*, Y . Os *quadrats* são áreas retangulares que abrangem n plantas em l linhas e c plantas na linha. Se o padrão espacial da doença no talhão é aleatório, pode-se assumir que $Y \sim \text{Bin}(n, \pi)$, com n o número de plantas por *quadrat* e π a taxa de incidência da doença no talhão. Se $n \rightarrow \infty$ e $\pi \rightarrow 0$, pode-se utilizar a distribuição de Poisson(θ), assumindo que $\theta \approx n\pi$.

O teste da hipótese de aleatoriedade espacial nesse caso, consiste em verificar se é razoável assumir uma distribuição binomial ou de Poisson para o número plantas doentes. No contexto de incidência de doenças, Madden & Hughes (1995) sugerem que um índice de dispersão D seja calculado para medir a agregação e para testar superdispersão. D é a razão entre a variância observada e a variância esperada segundo um modelo. Se a incidência tem distribuição espacial aleatória, D será igual a 1 e pode-se assumir que a contagem segue uma distribuição binomial. Se a distribuição espacial for regular, D é menor que 1. Se o padrão for agregado, D é maior que 1.

A variância observada de Y é dada por

$$V_o(Y) = \sum_i (y_i - n\pi)^2 / (N - 1) \quad (1)$$

em que y_i é o número de plantas doentes no i -ésimo *quadrat*, n é o número de plantas por *quadrats*, p é a proporção de plantas doentes no talhão e N o número de *quadrats* avaliados. A variância esperada considerando um modelo Binomial é dada por

$$V_{B_t}(Y) = n\pi(1 - \pi) . \quad (2)$$

Nota-se que n é fixo, impondo uma restrição, ou seja, o número de plantas por *quadrats* devem ser iguais. Diante disso, pode-se utilizar a variância esperada segundo o modelo de Poisson, que é a média, podendo ser calculada por

$$V_{P_t}(Y) = E(Y) = \sum_i Y_i n_i / \sum_i n_i, \quad (3)$$

em que n_i é o número de plantas no i -ésimo *quadrat*. Para o teste de significância para agregação utiliza-se a $(N - 1)D$, que tem distribuição χ^2_{N-1} .

Nos modelos binomial e de Poisson, π é constante em toda a região em estudo e o índice de dispersão é uma maneira simplista de testar isso. Em situações em que π varia, há evidências para afirmar que existe agregação espacial. Para considerar a variação de π , pode-se assumir uma distribuição $Beta(\alpha, \beta)$. A variação de π pode ser dada por α e β , ou então, pode ser feita uma reparametrização de forma que a variação de π seja expressa em um único parâmetro θ . No contexto de doenças de plantas é de grande importância, pois permite concluir se o padrão é agregado e fornece uma medida da agregação.

Na Seção 2 são implementadas as funções de densidade, de distribuição, de quantis e geradora de números aleatórios. Na Seção 3 são apresentadas as funções disponíveis em R para maximização numérica e para estimação por máxima verossimilhança. Na Seção 5 são feitas aplicações a dois conjuntos de dados de incidência de doenças em plantas.

2 Distribuição beta-binomial

A distribuição beta-binomial é obtida, segundo Skellam(1948) apud Mendonça, Bearzoti, Aquino & Ferreira (2004), compondo-se a distribuição binomial com a densidade beta para π . A função de densidade beta pode ser escrita de várias formas, uma delas é:

$$f(\pi) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \pi^{\alpha-1} (1 - \pi)^{\beta-1} , \quad (4)$$

$0 \leq \pi \leq 1$, α e β são constantes reais positivas e $\Gamma(\cdot)$ é a função gama.

A função de densidade de probabilidade da distribuição beta-binomial pode ser expressa por:

$$P(Y = y | \alpha, \beta) = \binom{n}{y} \frac{B(y + \alpha, n - y + \beta)}{B(\alpha, \beta)} , \quad (5)$$

com $y = 0, 1, \dots, n$; $\alpha \geq 0$, $\beta \geq 0$ conhecidos e B a função beta definida por $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$.

Griffiths (1973) sugere uma reparametrização implicando, segundo Mendonça et al. (2004), em simplicidade e interpretabilidade dos parâmetros. Os novos parâmetros são $p = \frac{\alpha}{\alpha+\beta}$, o valor esperado do parâmetro π da distribuição binomial e $\theta = \frac{1}{\alpha+\beta}$, uma medida da variação em π .

O valor esperado de Y e sua variância são dados por: $E(Y) = np$ e $Var(Y) = \frac{np(1-p)(1+n\theta)}{(1+\theta)}$. Assumindo-se que p varia ($\theta > 0$), pode-se observar que a variância da distribuição beta-binomial é maior que da binomial. O parâmetro θ , no contexto dos dados de incidência de doenças, pode ser interpretado como um índice de agregação espacial da doença. Se $\theta > 0$, o padrão da doença é agregado.

2.1 Definindo a função de densidade

Pode-se calcular a densidade de probabilidade definindo uma função em R. A forma mais fácil de implementar é a Equação 5. Para manter a interpretabilidade dos parâmetros no contexto de doenças em plantas, a função terá p e θ como argumentos e define internamente $\alpha = \frac{p}{\theta}$ e $\beta = \frac{(1-p)}{\theta}$.

Um detalhe que se deve acrescentar é que para valores de θ muito próximos de zero, os parâmetros α e β são muito grandes. Isso faz com que o valor da função beta seja muito próximo de zero. Exemplo:

```
> beta(99, 99)
```

```
[1] 8.879664e-61
```

Os valores de α e β não muito grandes, fazem a função se aproximar muito de zero. A escala logarítmica ameniza este problema:

```
> lbeta(999, 999)
```

```
[1] -1387.096
```

A escala logarítmica é a utilizada no cálculo da densidade:

```
> dbetabinom <- function(x, size, prob, theta, log=FALSE) {
+   # size é o tamanho do vetor x, prob e theta os parâmetros
+   # da densidade
+   alpha <- prob / theta
+   beta <- (1 - prob) / theta
+   dens <- lchoose(size,x) + lbeta(x+alpha,size-x+beta) -
+     lbeta(alpha,beta)
+   if (!log)
+     dens <- exp(dens)
+   return(dens)
+ }
```

A partir desta função pode-se estudar a densidade de probabilidade em gráficos de linhas, plotando o valor da densidade variando os parâmetros da distribuição. Na Figura 1, visualiza-se a densidade para algumas variações dos parâmetros.

Observa-se na Figura 1 que quanto mais próximo θ está de zero, a densidade da distribuição beta-binomial fica mais próxima da densidade da distribuição binomial. Além disso, valores de θ , próximos de zero comprometem a precisão no cálculo da densidade:

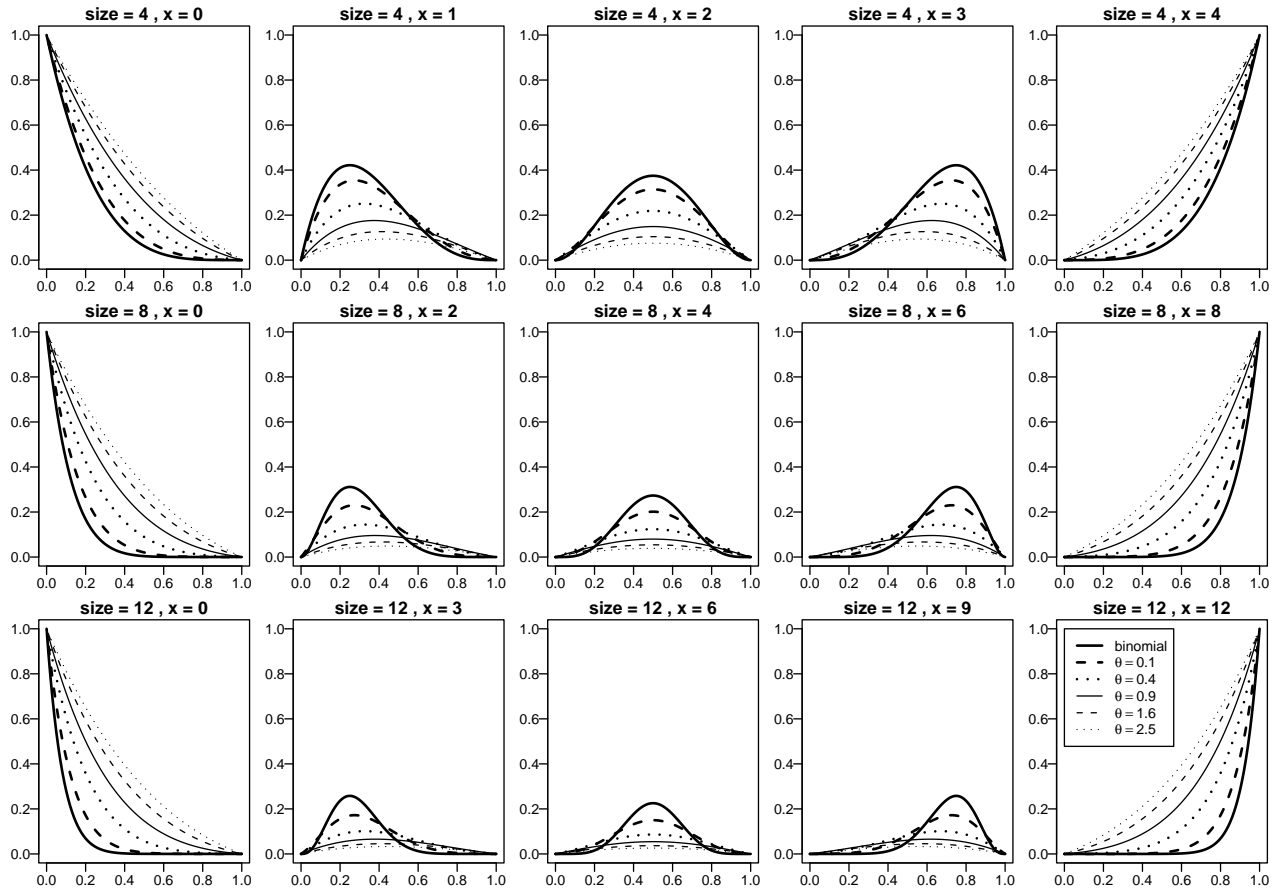


Figura 1: Densidade de probabilidade da distribuição beta-binomial. No eixo horizontal, está representado o valor do parâmetro π e no eixo vertical está representado o valor da densidade de probabilidade.

```
> sum(dbetabinom(0:5, 5, 0.5, 1e-10))
```

```
[1] 1.000000
```

A precisão pode ser melhorada considerando que a distribuição beta-binomial se aproxima da distribuição binomial quando $\theta \approx 0$. Assim, pode-se definir a função de densidade de forma que ser o valor de θ for muito próximo de zero, seja tomada a densidade da distribuição binomial. No R, `.Machine` é uma lista com as características numéricas do computador utilizado. `.Machine[['double.eps']]` é o menor número positivo tal que o computador em uso seja capaz de reconhecer como verdadeiro que $1 + .Machine[['double.eps']] > 1$:

```
> .Machine$double.eps
```

```
[1] 2.220446e-16
```

A função de densidade pode ser re-definida, usando a raiz de `.Machine[['double.eps']]` como valor mínimo para θ :

```
> dbetabinom <- function(x, size, prob, theta, log=FALSE) {
+   alpha <- prob/theta
```

```

+   beta <- (1-prob)/theta
+   dens <- lchoose(size,x) + lbeta(x+alpha,size-x+beta) -
+       lbeta(alpha,beta)
+   if (!log)
+       dens <- exp(dens)
+   # selecionando as densidades que satisfazem  $0 \leq \theta < 1,49 \times 10^{-8}$ 
+   id <- (1:length(dens))[theta>=0 & theta<sqrt(.Machine$double.eps)]
+   if (length(id)>0)
+       dens[id] <- dbinom(x, size, prob, log)[id]
+   return(dens)
+ }

```

2.2 Definindo as funções de distribuição, de quantis e de números aleatórios

A partir da função de densidade, podem ser definidas também a função de probabilidade acumulada, sua inversa e a geradora de números aleatórios.

A função de distribuição, $P(Y \leq y)$ é obtida somando-se as densidades avaliadas em $\{0, \dots, y\}$.

```

> pbetabinom <- function(q, size, prob, theta,
+   lower.tail=TRUE, log.p=FALSE) {
+   # pars é um data.frame que contém os parâmetros da função
+   pars <- data.frame(rbind(q, size, prob, theta, log.p))
+   # x[1]=q, x[2]=size, x[3]=prob, x[4]=theta e x[5]=log.p
+   p <- new("numeric",
+       sapply(pars, function(x)
+           sum(dbetabinom(0:x[1], x[2], x[3], x[4], x[5]))))
+   if (!lower.tail)
+       p <- 1-p
+   return(p)
+ }

```

No cálculo do inverso da função de distribuição acumulada, ou quantis, a partir de $\{p \in (0, 1)\}$, encontra-se o menor valor de $y = \{0, \dots, n\}$ tal que $p > P(Y \leq y)$.

```

> qbetabinom <- function (p, size, prob, theta,
+   lower.tail=TRUE, log.p=FALSE) {
+   # p é a probabilidade (F(y)) correspondente ao quantil (y) que se
+   # deseja determinar. size corresponde a posição de q em y.
+   # Ex: para y=2, size=3.
+   if (log.p)
+       p <- log(p)
+   ql <- function(z)
+       (0:z[2])[min(which(cumsum(dbetabinom(0:z[2], z[2], z[3], z[4]))>z[1]))]
+   q <- new("numeric",
+       sapply(data.frame(rbind(p, size, prob, theta)), ql))
+ }

```

```
+   if (!lower.tail)
+     q <- size-q
+   return(q)
+ }
```

A função geradora de números aleatórios para qualquer densidade de probabilidade pode ser genericamente obtida a partir de um gerador de números aleatórios da distribuição uniforme no intervalo $(0, 1)$ e da função que calcula quantis. Dessa forma, pode-se gerar amostras da distribuição $U(0, 1)$ e calcular quantis tomando essas amostras como probabilidade acumulada.

No caso de distribuições cujo número de valores possíveis é finito, o uso da função acumulada inversa pode ser evitado. É possível utilizar uma função para sortear valores a partir de um conjunto finito de valores, com probabilidade dada pela função de densidade. No R, pode-se utilizar a função `sample()` para isso.

```
> rbetabinom <- function(n, size, prob, theta) {
+   # n é o número de elementos da amostra
+   # size, prob e theta os parâmetros da densidade
+   r <- function(x)
+     sample(0:x[1], size=1, replace=TRUE,
+            prob=dbetabinom(0:x[1], size=x[1],
+                             prob=x[2], theta=x[3], log=FALSE))
+   if (length(n)>1)
+     n <- length(n)
+   return(new("numeric",
+              sapply(data.frame(rbind(size, prob, theta, rep(1,n))), r)))
+ }
```

O pacote **rmutil**, de J. K. Lindsey, contém essas funções. Porém, há um inconveniente apontado na Seção 3 que impede utilizá-las. Esse pacote, não está disponível no CRAN, mas está disponível em <http://alpha.luc.ac.be/~jlindsey/rcode.html>. Na parametrização adotada por Lindsey, é utilizado $\phi = 1/\theta$. Nesse caso ϕ é um parâmetro de dispersão de π .

3 Funções para maximização numérica

No procedimento de estimação por máxima verossimilhança é necessário encontrar as derivadas em relação aos parâmetros. No caso da estimação dos parâmetros da distribuição beta-binomial, não há expressão analítica fechada para os estimadores de máxima verossimilhança. Nessa situação, a solução que maximiza a verossimilhança deve ser encontrada iterativamente, via algoritmo de maximização numérica. Devido à estabilidade numérica, é conveniente maximizar o logaritmo da função de verossimilhança.

Para a maximização numérica, será utilizada a função `dbetabinom()` definida anteriormente, embora já esteja implementada no pacote **rmutil**. A função `dbetabinom()` do pacote **rmutil** apresenta um inconveniente que impossibilita utilizá-la na função de maximização numérica: quando os valores dos parâmetros estão fora do espaço paramétrico, a função é interrompida e não retorna valor algum. Na função `dbetabinom()` definida anteriormente, ao ser informado um valor fora do espaço paramétrico, a expressão é avaliada e é retornado zero ou `NA`, dependendo do caso. Isto permite utilizá-la no procedimento de maximização numérica.

No R, estão disponíveis funções para minimização ou maximização numérica a partir de uma função objetivo definida. A menos que seja informado em argumentos adicionais dessas funções, estas buscarão o valor que minimiza a função objetivo. Neste caso, para simplificar, as mesmas são utilizadas para encontrar a solução que minimiza o negativo do logaritmo da função de verossimilhança, (R Development Core Team 2005).

3.1 Função `optimize()`

A função `optimize()` ou seu pseudônimo `optimise()`, pode ser utilizada para encontrar o mínimo de uma função em um intervalo. Esta função usa uma tradução em C do código Fortran (Netlib) baseada no procedimento do 'Algol 60 localmin' dado na referência.

```
> args(optimize)

function (f, interval, ..., lower = min(interval), upper = max(interval),
  maximum = FALSE, tol = .Machine$double.eps^0.25)
NULL
```

Os argumentos obrigatórios da função `optimize()` são: a função e o intervalo. Os demais argumentos são alternativos. O argumento `maximum` indica se o objetivo é maximizar ou minimizar.

Para exemplificar, seja uma amostra da distribuição binomial com $n = 5$ e $\pi = 0.5$.

```
> set.seed(123)
> y1 <- rbinom(100, 5, 0.5)
> table(y1)
```

```
y1
 0  1  2  3  4  5
 2 16 35 31 14  2
```

Suponha desconhecida a expressão analítica do estimador de máxima verossimilhança de π . Pode-se maximizar numericamente a função de verossimilhança em relação a π e, para isto, deve-se defini-la de forma que o parâmetro a ser estimado seja o primeiro argumento:

```
> vero.binom <- function(p, x, n) sum(sapply(x, dbinom,
+   n, p, log = TRUE))
```

A função `vero.binom()` retorna a soma do logaritmo das densidades avaliadas em um valor de x informado, considerando os parâmetros π e n também informados.

```
> optimize(vero.binom, c(0, 1), maximum = TRUE, n = 5,
+   x = y1)

$maximum
[1] 0.4899824

$objective
[1] -146.2198
```

O valor estimado para π foi 0.4899824 e o correspondente valor do logaritmo da verossimilhança de -146.2198 . Conhecendo a expressão analítica do estimador, pode-se confirmar:

```
> mean(y1)/5
```

```
[1] 0.49
```

3.2 Função `nlm()`

A função `nlm()` pode ser utilizada para minimização ou maximização numérica de funções não-lineares. O procedimento iterativo é um algoritmo do tipo de Newton. Maiores detalhes podem ser vistos na documentação.

```
> args(nlm)
```

```
function (f, p, ..., hessian = FALSE, typsize = rep(1, length(p)),
  fscale = 1, print.level = 0, ndigit = 12, gradtol = 1e-06,
  stepmax = max(1000 * sqrt(sum((p/typsize)^2)), 1000), steptol = 1e-06,
  iterlim = 100, check.analyticals = TRUE)
```

```
NULL
```

A função a ser minimizada deve ser informada no argumento `f`. O gradiente e o hessiano da função podem ser informados como atributos da função. Quando estes argumentos não são informados, são utilizadas derivadas numéricas da função para a minimização.

A expressão do negativo do logaritmo da função de verossimilhança, considerando a distribuição beta-binomial pode ser implementada por:

```
> nlb <- function(pars, size, y) -sum(sapply(y, dbetabinom,
+   size, pars[1], pars[2], log = T))
```

Para simular 100 amostras com distribuição beta-binomial com parâmetros $\pi = 0.3$, $n = 5$ e $\theta = 0.1$ utiliza-se a função `rbetabinom()` definida anteriormente:

```
> table(y2 <- rbetabinom(100, 5, 0.3, 0.1))
```

```
 0  1  2  3  4
25 28 28 12  7
```

```
> (r2nlm <- nlm(nlb, c(0.3, 0.1), hessian = T, y = y2,
+   size = 5))
```

```
$minimum
```

```
[1] 151.8767
```

```
$estimate
```

```
[1] 0.2956661 0.1006574
```

```
$gradient
```



```
[1] 1.266429e-05 1.490079e-05
```

```
$hessian
```

```
      [,1]      [,2]  
[1,] 1769.64308 -67.98566  
[2,] -67.98566 414.92750
```

```
$code
```

```
[1] 1
```

```
$iterations
```

```
[1] 6
```

Os códigos 1 e 2 no elemento `code` indicam que, provavelmente, chegou-se à solução.

A matriz de informação de Fischer pode ser aproximada pelo hessiano numérico obtido numericamente, neste caso.

```
> solve(r2nlm$hessian)
```

```
      [,1]      [,2]  
[1,] 5.686653e-04 9.317552e-05  
[2,] 9.317552e-05 2.425326e-03
```

3.3 Função `optim()`

A função `optim()` permite obter os valores de parâmetros de uma função que minimizam essa função. Os métodos de otimização disponíveis (R versão 2.1.1) são: Nelder-Mead, Quasi-Newton, Gradiente-Conjugado, Confinamentos de Caixa e *Simulated Annealing*. Na documentação há detalhes desses métodos.

```
> args(optim)
```

```
function (par, fn, gr = NULL, ..., method = c("Nelder-Mead",  
      "BFGS", "CG", "L-BFGS-B", "SANN"), lower = -Inf, upper = Inf,  
      control = list(), hessian = FALSE)  
NULL
```

Nesta função, os argumentos obrigatórios são: o valor inicial dos parâmetros e a expressão da função. Os argumentos opcionais são: a função gradiente, o método de minimização (na documentação há detalhes sobre cada método disponível), os limites inferior e superior do espaço paramétrico se for utilizado o método 'L-BFGS-B', uma lista de parâmetros de controle e demais argumentos da função a ser minimizada ou da função gradiente.

Utiliza-se a função `optim()` para a estimação de π e θ . Neste caso o valor verdadeiro de θ é zero. Pode-se utilizar o método de otimização default:

```
> optim(c(0.3, 0.1), nlb, size = 5, y = y1)
```

```
$par
[1] 4.900283e-01 6.181747e-08
```

```
$value
[1] 146.2198
```

```
$counts
function gradient
      119      NA
```

```
$convergence
[1] 0
```

```
$message
NULL
```

Os valores retornados foram: os valores dos parâmetros com os quais encontrou-se o mínimo da função; o correspondente valor da função avaliado nos parâmetros encontrados; o número de valores avaliados na função e no gradiente, neste caso não foi avaliado o gradiente; o código indicando a convergência, zero indica que convergiu; e, alguma mensagem adicional quando houver.

Apresenta-se a seguir as estimativas de máxima verossimilhança da amostra 2. Obtém-se, também, o hessiano numérico para aproximar a matriz de informação de Fisher observada colocando o argumento `hessian=TRUE`.

```
> # str: comando que exhibe a lista a seguir
> str(ropt2 <- optim(c(0.3, 0.1), nlb, hessian=TRUE, size=5, y=y2))
```

```
List of 6
 $ par      : num [1:2] 0.296 0.101
 $ value    : num 152
 $ counts   : Named int [1:2] 39 NA
 ..- attr(*, "names")= chr [1:2] "function" "gradient"
 $ convergence: int 0
 $ message   : NULL
 $ hessian   : num [1:2, 1:2] 1770.5 -67.7 -67.7 415.5
```

A matriz de informação de Fisher aproximada é obtida fazendo:

```
> solve(ropt2$hessian)

      [,1]      [,2]
[1,] 5.683560e-04 9.258861e-05
[2,] 9.258861e-05 2.421844e-03
```

3.4 Função nlminb()

A função `nlminb()` faz minimização numérica usando rotinas PORT. Também realiza minimização sob restrições no espaço paramétrico. Exemplo, com restrição no espaço paramétrico:

```
> # vetor de parâmetros: c(prob,theta), com prob restrito de 1e-10 a 1
> #                               e theta de 0 a Inf.
> str(nlminb(c(0.3, 0.1), nlb, hessian=TRUE, size=5, y=y2, lower=c(1e-10,0),
+         upper=c(1,Inf)))
```

List of 6

```
$ par          : num [1:2] 0.296 0.101
$ objective    : num 152
$ convergence  : int 0
$ message      : chr "relative convergence (4)"
$ iterations   : int 4
$ evaluations: Named int [1:2] 9 9
..- attr(*, "names")= chr [1:2] "function" "gradient"
```

4 Funções para estimação por máxima verossimilhança

A função `fitdistr()` e `mle()` são funções especificamente implementadas para estimação por máxima verossimilhança.

4.1 Função fitdistr()

A função `fitdistr()` do pacote **MASS** é específica para estimação por máxima verossimilhança (Venables & Ripley 2002).

```
> args(fitdistr)
```

```
function (x, densfun, start, ...)
NULL
```

Para utilizá-la, basta entrar com os dados, a função de densidade e os valores iniciais em formato de lista nomeada. Esta função utiliza a função `optim()` e demais argumentos podem ser passados para função `optim()` e também para a função de densidade nomeando-os, por exemplo, o argumento `size`. Essa função retorna também as estimativas dos erros padrão, obtidos através do hessiano numérico.

Na amostra 1 não foi possível utilizar a função `fitdistr()`. Sabe-se que o valor do parâmetro θ é zero e é o limite inferior do espaço paramétrico. A dificuldade aqui é encontrar o hessiano numérico, pois para isso é preciso avaliar a função na região próxima ao valor estimado na minimização numérica e para qualquer valor de θ menor que zero, a função de densidade não é definida. Neste caso, visualiza-se apenas a função desvio (ou deviance), Figura 2.

Aplicando às outras duas amostras, podemos utilizar o método “L-BFGS-B” que obrigatoriamente necessita a definição do espaço paramétrico. Para isto, informa-se os limites de cada um dos parâmetros nos argumentos `lower` e `upper`.

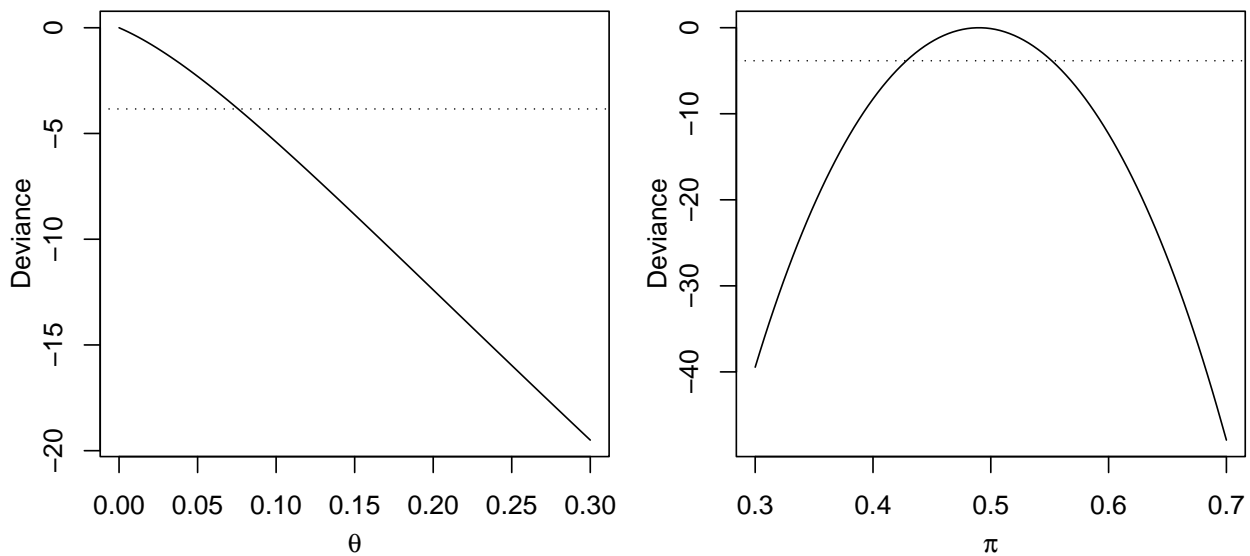


Figura 2: Função desvio para os parâmetros θ (esquerda) e π (direita) da amostra 1. A linha pontilhada corta a curva da função desvio nos limites de confiança de 95%.

```
> (r2 <- fitdistr(y2, dbetabinom, start = list(prob = 0.3,
+      theta = 1), size = 5, method = "L-BFGS-B", lower = c(0,
+      0), upper = c(1, Inf)))
```

```
      prob      theta
0.29566690 0.10065931
(0.02384172) (0.04921677)
```

4.2 Função mle()

A função `mle()` do pacote **stats4** também é específica para a estimação por máxima verossimilhança, oferecendo a possibilidade de estudar as verossimilhanças perfilhadas.

```
> args(mle)

function (minuslogl, start = formals(minuslogl), method = "BFGS",
      fixed = list(), ...)
NULL
```

Deve ser informada a expressão do negativo do logaritmo da verossimilhança e os valores iniciais para os parâmetros em forma de lista nomeada. Esta função também utiliza internamente a função `optim()`.

O resultado da função `mle()` é um objeto da classe `mle`, a partir do qual pode-se extrair a matriz de informação de Fisher, obtida através do hessiano numérico e as verossimilhanças perfilhadas para os parâmetros e com isso, construir intervalos de confiança assintóticos e visualizar as verossimilhanças perfilhadas. Esses resultados são extraídos utilizando-se os métodos

implementados para o objeto da classe `mle`, tais como `summary()`, `vcov()`, `profile()`. Ao resultado da função `profile()` podem ser aplicados os métodos `plot()` e `confint()`.

Para a amostra 2, simulada com $p = 0.3$ e $\theta = 0.1$, tem-se:

```
> summary(mod2 <- mle(function(prob, theta) -sum(sapply(y2,
+      dbetabinom, 5, prob, theta, T)), start = list(prob = 0.3,
+      theta = 0.1), method = "Nelder-Mead"))
```

Maximum likelihood estimation

Call:

```
mle(minuslogl = function(prob, theta) -sum(sapply(y2, dbetabinom,
  5, prob, theta, T)), start = list(prob = 0.3, theta = 0.1),
  method = "Nelder-Mead")
```

Coefficients:

	Estimate	Std. Error
prob	0.2956434	0.02384022
theta	0.1006480	0.04921224

-2 log L: 303.7535

A função `vcov()` extrai a matriz aproximada de variância e covariância das estimativas dos parâmetros, ou seja, o inverso da informação de Fisher observada aproximada pelo hessiano numérico.

```
> vcov(mod2)
```

	prob	theta
prob	5.683560e-04	9.258861e-05
theta	9.258861e-05	2.421844e-03

As verossimilhanças perfilhadas são obtidas pela função `profile`. A partir de seu resultado, pode-se calcular o intervalo de confiança e plotar esses resultados. A verossimilhança perfilhada de um parâmetro é a análise do comportamento da função de verossimilhança, avaliada em valores próximos ao valor estimado desse parâmetro, mantendo os outros fixos nos respectivos valores estimados. O intervalo de confiança é obtido a partir das verossimilhanças perfilhadas para cada um dos parâmetros estimados.

```
> prof2 <- profile(mod2)
```

O resultado da função `profile` também pode ser visualizado em gráfico. O método `plot()` para essa classe produz um gráfico para cada parâmetro, tendo-se no eixo horizontal os valores do parâmetro e no eixo vertical, o valor da verossimilhança padronizado para $z \sim N(0, 1)$. Isto facilita a visualização da precisão da estimativa, sendo traçadas linhas de referência para intervalos de confiança. Fazendo $z = 1.96$ na linha vertical, obtém-se o valor correspondente ao limite do intervalo de confiança de 95% na linha horizontal. Na Figura 3, visualiza-se esse gráfico para os parâmetros estimados na amostra 2.

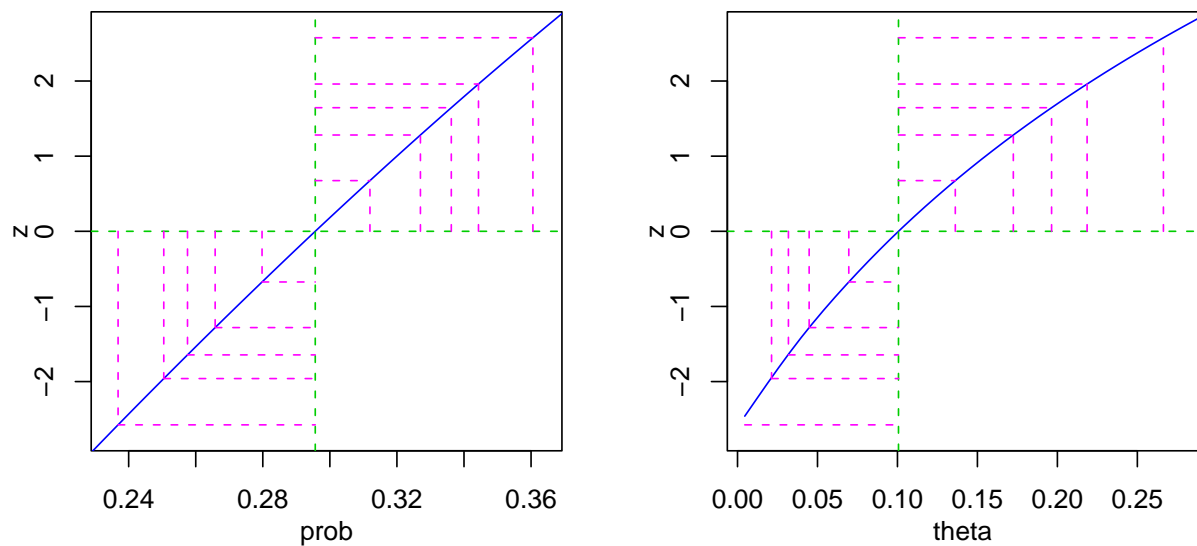


Figura 3: Verossimilhanças perfilhadas padronizadas para a amostra 2.

```
> par(mfrow = c(1, 2), mar = c(3, 3, 2, 0.5), mgp = c(2,
+ 1, 0))
> plot(prof2, absVal = FALSE)
```

No gráfico da função padronizada de verossimilhança perfilhada de θ , observa-se que não é razoável construir um intervalo de confiança simétrico para θ a partir o erro padrão obtido numericamente. Pode-se obter os valores numéricos do intervalo de confiança a partir da verossimilhança perfilhada, fazendo:

```
> confint(prof2)

      2.5 %    97.5 %
prob 0.25047609 0.3443292
theta 0.02062236 0.2186046

> confint(prof2, level = 0.99)

      0.5 %    99.5 %
prob 0.2368451 0.3605601
theta      NA 0.2662986
```

O teste de hipótese pode ser feito a partir da razão de verossimilhanças ou da diferença de log-verossimilhanças. No caso particular da distribuição beta-binomial, para testar a hipótese de que θ é igual a zero, basta comparar a verossimilhança da hipótese com a verossimilhança segundo a distribuição binomial.

```
> l1 <- sum(dbetabinom(y2, size = 5, prob = coef(mod2)[1],
+ theta = coef(mod2)[2], log = TRUE))
```

```
> l2 <- sum(dbinom(y2, size = 5, prob = coef(mod2)[1],
+ log = TRUE))
> (p.value <- pchisq(l1 - l2, 1, lower.tail = FALSE))

[1] 0.06487557
```

5 Aplicação em dados reais

A metodologia foi aplicada a dois conjuntos de dados. O primeiro contém dados de PPC em um talhão do município de Itajobi - MG e o segundo, dados de MSC em um talhão localizado no município de Comendador Gomes - MG.

Na aplicação a dados de doenças de plantas, fez-se a contagem do número de plantas doentes por *quadrat* e aplicou-se a metodologia na contagem. O **Rcitrus** é um pacote desenvolvido para análise de dados de doenças em Citrus e contém funções para análise por *quadrats*, incluindo a metodologia da seção anterior.

5.1 Dados de Pinta Preta dos Citrus

O patógeno causador da PPC pode se reproduzir por ascósporos ou por penidósporos. Da primeira forma, o esporo está na fase sexual do patógeno e a disseminação ocorre pelo vento causando um padrão espacial aleatório. Da segunda forma, relacionado à fase assexual, a disseminação ocorre por escorrimento da água da chuva ou irrigação e a contaminação de uma árvore para outra se dá pelo contato com essa água.

A metodologia de estimação dos parâmetros da distribuição beta-binomial foi aplicada em um conjunto de dados de incidência de PPC. Estes dados são de uma fazenda localizada em Itajobi, Estado de Minas Gerais. Esses dados estão disponíveis no pacote **Rcitrus**.

O mapa da incidência pode ser visto na Figura 4. Pode-se observar que existe irregularidade no número de plantas nas linhas.

```
> require(Rcitrus)
> data(Itajobi)
> geo.ita <- as.citrus(Itajobi, find.form = "geodata",
+ x = 8, y = 4)

x = 8 y = 4

> plot(geo.ita, pch = 18)
```

É preciso avaliar se é possível assumir que a incidência é constante em toda a área em estudo. A seguir utiliza-se a função `quadrat.count()` implementada no pacote **Rcitrus** para obter a contagem do número de plantas doentes por *quadrats*.

```
> (q <- quadrat.count(Itajobi, 10, 10))

$y
  [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   0   1   0   0   0   0
```

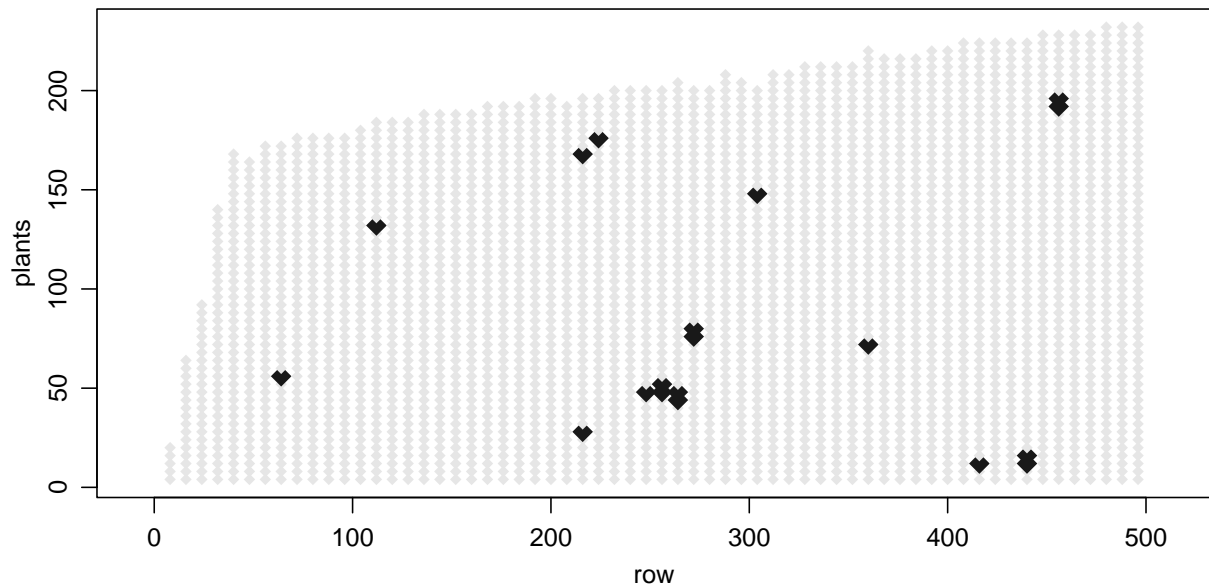


Figura 4: Incidência de Pinta Preta dos Citrus. Pontos menores indicam plantas saudáveis e pontos maiores indicam plantas doentes. Posições com ausência de pontos, indica ausência de plantas.

```
[2,] 0 0 0 1 0 0
[3,] 1 0 0 0 2 0
[4,] 0 7 0 1 0 0
[5,] 0 1 0 0 0 0
[6,] 3 0 0 0 2 0
```

```
$n
  [,1] [,2] [,3] [,4] [,5] [,6]
[1,]  95  86  73  65  17   0
[2,] 100 100 100 100  59   0
[3,] 100 100 100 100  88   0
[4,] 100 100 100 100 100   8
[5,] 100 100 100 100 100  39
[6,] 100 100 100 100 100  66
```

```
$dq
[1] 100
```

Observa-se *quadrats* incompletos, com menos de 100 plantas. Com esses dados explora-se as verossimilhanças perfilhadas fazendo a estimação que internamente utiliza a função `mle()`. Para isso utiliza-se a função `fit.betabinom()` disponível no pacote **Rcitrus**.

```
> args(fit.betabinom)

function (y, size, ini.prob = NULL, ini.theta = NULL, usage = c("fitdistr",
  "mle"), ...)
NULL
```

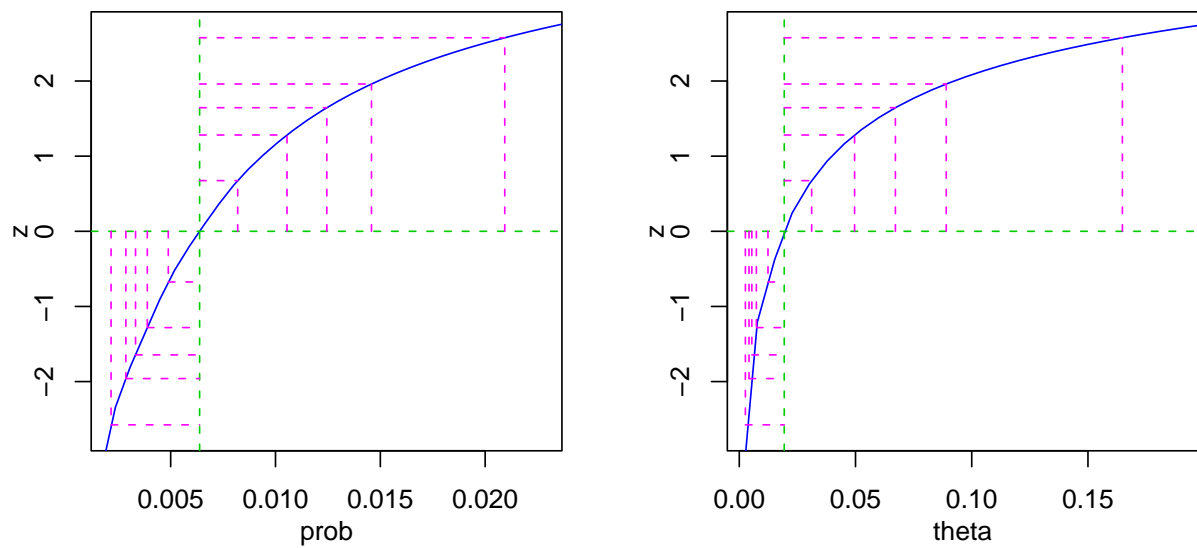



Figura 5: Verossimilhanças perfilhadas padronizadas dos parâmetros da distribuição beta-binomial ajustada aos dados de Pinta Preta dos Citrus.

```
> x <- as.numeric(q$y)
> n <- as.numeric(q$n)
> mod.ita <- fit.betabinom(x, n, usage = "mle")
> prof.ita <- profile(mod.ita)
> confint(prof.ita)
```

```
          2.5 %      97.5 %
prob 0.002884058 0.01457969
theta 0.004626649 0.08904408
```

O intervalo de confiança para o parâmetro de agregação não abrange o zero, ou seja, o modelo binomial é insuficiente para explicar a variabilidade da incidência da doença. Com isso, pode-se concluir que o padrão espacial da incidência de Pinta Preta dos Citrus nesse talhão é agregado. O gráfico das verossimilhanças perfilhadas padronizadas para essa análise estão na Figura 5.

```
> par(mfrow = c(1, 2), mar = c(3, 3, 2, 0.5), mgp = c(2,
+ 1, 0))
> plot(prof.ita, absVal = FALSE)
```

No pacote **Rcitrus**, está implementado a função `disp.quadrats()` para o teste de agregação baseado no índice de dispersão binomial e Poisson. Aplicando aos dados, fazendo a contagem por *quadrats* de tamanho 10×10 , tem-se:

```
> disp.quadrats(Itajobi, dx = 10, dy = 10)
```

```
$`10x10`
```

	n	N	nN	p	obs.var	theor.var	index	p.value	pattern
Av1	100	23	2300	0.00696	0.00025	7e-05	3.61034	0	Agregate

A variância observada é significativamente maior que a variância esperada segundo o modelo binomial e conclui-se que o padrão é agregado. Pode-se propor o modelo beta-binomial e testar se o parâmetro de agregação da distribuição beta-binomial θ é igual a zero. No argumento `model`, pode-se escolher a opção de utilizar a distribuição beta-binomial para testar a hipótese de aleatoriedade espacial. O p-valor retornado é o resultado de um teste de razão de verossimilhanças. O teste avalia a verossimilhança da distribuição beta-binomial com parâmetro θ igual ao estimado com os dados contra a verossimilhança da distribuição binomial.

```
> disp.quadrats(Itajobi, dx = 10, dy = 10, model = "beta-binomial")
```

```
$`10x10`
```

	N	n	nN	prob	theta	p.value	pattern
Av1	36	80.44444	2896	0.00638	0.01934	0.00462	Agregate

Foi avaliado um número maior de *quadrats* que na análise do índice de dispersão binomial.

5.2 Dados de Morte Súbita dos Citrus

Segundo Bassanezi, Fernandes & Yammamoto (2003) a MSC é uma nova doença dos Citrus que provoca rápido definhamento e morte de variedades de laranjas enchertadas, praticamente, apenas em limoeiro *Cravo*. O Estado de São Paulo responde por 80% da produção citrícola nacional e 85% de seus pomares apresentam laranjeiras doces enxertadas sobre limoeiro cravo. Portanto, há uma elevada vulnerabilidade da cultura à ocorrência de novas epidemias e particularmente para a MSC.

O primeiro registro oficial da doença foi realizado em fevereiro de 2001 no município de Comendador Gomes, Estado de Minas Gerais. Em 2002 a MSC atingiu o Estado de São Paulo e vários trabalhos tem sido conduzidos para a descoberta de informações a respeito. Esses trabalhos incluem a coleta de dados

O *Citrus Sudden Death Virus* - CSDV, novo vírus da família *Tymoviridae*, tem sido associado à MSC em pesquisas feitas pela empresa de biotecnologia Alellyx. Em abril de 2004 a mesma empresa anunciou a descoberta da presença do patógeno da MSC. O CSDV foi encontrado em três insetos, dois deles sendo capazes de transmitir o vírus para as plantas, (Alellyx 2004).

No talhão 303 da fazenda Vale Verde, localizada no município de Comendador Gomes - MG, foram feitas 25 avaliações. Esses dados estão disponíveis no pacote **Rcitrus**.

```
> data(v303.geo)
```

Na Figura 6, estão os mapas da evolução da MSC ao long das primeiras 24 das 25 avaliações feitas. Não se apresenta o mapa da última avaliação por restrição de espaço, mas este pode ser gerado a partir dos dados disponíveis no **Rcitrus**.

```
> par(mfrow = c(6, 4), mar = c(0.7, 1, 0.1, 0.1), mgp = c(0.5,
+ 0.1, 0))
> for (i in 1:24) plot(v303.geo, pch = 18, eval = i)
```

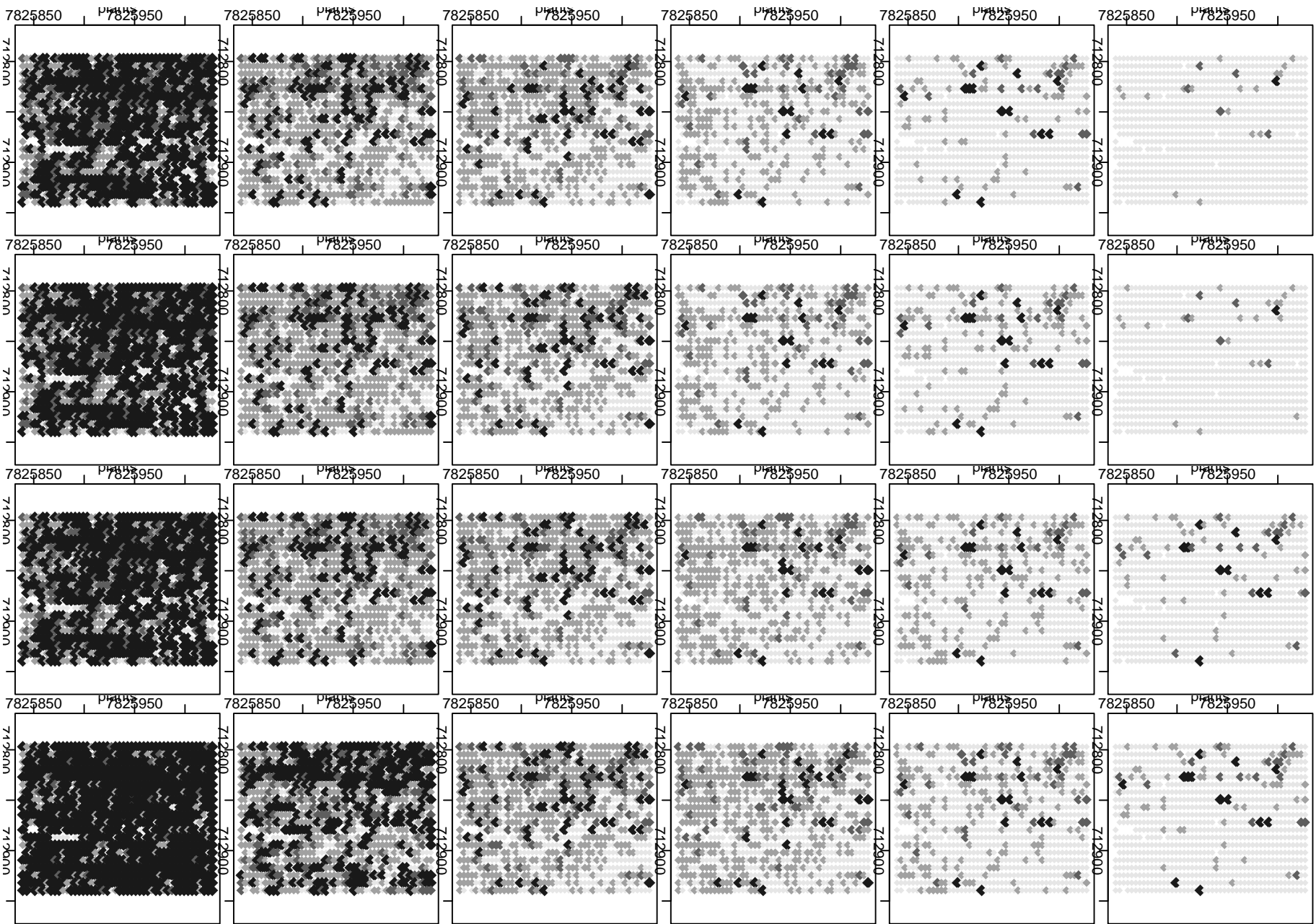


Figura 6: Mapas da evolução da Morte Súbita⁹, ao longo de 24 avaliações feitas no talhão 303 da fazenda Vale Verde.

Os códigos atribuídos à cada planta em cada avaliação são: 0 para plantas sadias, 1 para plantas com sintomas de MSC, 2 para plantas em estado avançado de MSC e 3 para plantas mortas. Considere o número de plantas em cada *status*, em cada avaliação:

```
> apply(v303.geo$data, 2, table)
```

	Av1	Av2	Av3	Av4	Av5	Av6	Av7	Av8	Av9	Av10	Av11	Av12	Av13	Av14	Av15
0	922	918	872	856	847	796	706	702	645	628	480	423	368	356	325
1	17	21	46	59	68	117	205	194	251	266	402	438	477	461	488
2	5	5	17	17	17	17	19	34	34	36	44	55	55	69	70
3	1	1	10	13	13	15	15	15	15	15	19	29	45	59	62

	Av16	Av17	Av18	Av19	Av20	Av21	Av22	Av23	Av24	Av25
0	307	224	167	163	120	69	69	69	27	27
1	492	500	554	554	385	223	217	211	125	125
2	80	108	108	112	108	110	98	100	20	20
3	66	113	116	116	332	543	561	565	773	773

O espaçamento nesse talhão é de 7.5 metros entre linhas e de 4 metros entre plantas na linha. Na análise, adota-se *quadrats* 2×4 considerando a incidência como sendo plantas identificadas pelos códigos 1, 2 e 3:

```
> (res.msc <- disp.quadrats(v303.geo, dx = 2, dy = 4, mod = "beta",
+   dea = 1:3, count = T))$res
```

```
$`2x4`
```

	N	n	nN	prob	theta	p.value	pattern
Av1	120	7.875	945	0.02424	0.08424	0.03146	Agregate
Av2	120	7.875	945	0.02852	0.05824	0.08450	Random
Av3	120	7.875	945	0.07723	0.04297	0.14315	Random
Av4	120	7.875	945	0.09417	0.07850	0.01216	Agregate
Av5	120	7.875	945	0.10362	0.07186	0.02135	Agregate
Av6	120	7.875	945	0.15712	0.09416	0.00494	Agregate
Av7	120	7.875	945	0.25270	0.07101	0.01666	Agregate
Av8	120	7.875	945	0.25699	0.06021	0.03639	Agregate
Av9	120	7.875	945	0.31778	0.06325	0.02843	Agregate
Av10	120	7.875	945	0.33571	0.07224	0.01414	Agregate
Av11	120	7.875	945	0.49247	0.08510	0.00466	Agregate
Av12	120	7.875	945	0.55336	0.10539	0.00095	Agregate
Av13	120	7.875	945	0.61161	0.15687	0.00001	Agregate
Av14	120	7.875	945	0.62393	0.16127	0.00000	Agregate
Av15	120	7.875	945	0.65652	0.18516	0.00000	Agregate
Av16	120	7.875	945	0.67488	0.19709	0.00000	Agregate
Av17	120	7.875	945	0.76230	0.12232	0.00016	Agregate
Av18	120	7.875	945	0.82294	0.13262	0.00010	Agregate
Av19	120	7.875	945	0.82767	0.14027	0.00007	Agregate
Av20	120	7.875	945	0.87312	0.13080	0.00020	Agregate
Av21	120	7.875	945	0.92705	0.10928	0.00239	Agregate
Av22	120	7.875	945	0.92705	0.10928	0.00239	Agregate

Av23	120	7.875	945	0.92705	0.10928	0.00239	Agregate
Av24	120	7.875	945	0.97131	0.10677	0.01147	Agregate
Av25	120	7.875	945	0.97131	0.10677	0.01147	Agregate

Observa-se que o padrão é aleatório nas avaliações 2 e 3 e agregado na primeira avaliação e nas avaliações 4 a 25. Esses resultados não mudam muito fazendo a análise com *quadrats* 3×5 ou 3×7 ou utilizando o índice de dispersão binomial.

A visualização da função desvio para θ em cada avaliação pode ser feita a partir das estimativas obtidas e da contagem por *quadrats*. Na Figura 7 estão plotadas as funções de deviance para θ estimado nas 25 avaliações utilizando *quadrats* 2×4 .

6 Discussão e Resultados

A idéia de estimação por máxima verossimilhança foi apresentada em detalhes. Todos os passos envolvidos foram apresentados: problema da obtenção analítica dos estimadores, definição da função de densidade, definição da função de verossimilhança, maximização da função de verossimilhança via algoritmos de minimização numérica disponíveis em R, estimativa dos erros dos estimadores a partir do hessiano numérico, intervalos de confiança e teste de razão de verossimilhanças.

Foram apresentadas diferentes funções para maximização numérica, demonstrado o seu uso e, apontadas as particularidades de cada uma. Os procedimentos para estimação, obtenção dos erros padrão aproximados numericamente, obtenção da verossimilhança perfilhada e da função deviance foram exemplificados.

A metodologia foi exemplificada na estimação por máxima verossimilhança dos parâmetros da distribuição beta-binomial. Foi utilizada uma parametrização de forma que houvesse uma interpretação prática no contexto de análise de padrões espaciais de incidência de doenças. O teste da hipótese de aleatoriedade espacial deve ser realizado a partir do teste de razão de verossimilhanças, pois o estimador de θ não tem comportamento simétrico, quando a estimativa está próxima da borda do espaço paramétrico.

A análise do padrão espacial de doenças foi aplicada a dados de PPC e dados de MSC. Para os dados de MSC, foram analisados os dados de 25 avaliações feitas, em que se observou incidências desde 2.43% até 97.14%. A partir do teste de aleatoriedade espacial utilizando o parâmetro de agregação da distribuição beta-binomial, verificou-se que o padrão espacial da PPC é agregado no talhão analisado. Na análise dos dados de MSC, verificou-se que as evidências de padrão agregado são menores nas avaliações com incidência baixa da MSC.

Agradecimentos

Ao Laboratório de Estatística e Geoinformação (LEG), pelo ambiente computacional. Ao Fundo de Defesa da Citricultura (FUNDECITRUS) pelo financiamento do projeto em convênio com o Departamento de Estatística da Universidade Federal do Paraná, em cujo escopo estas atividades foram desenvolvidas.

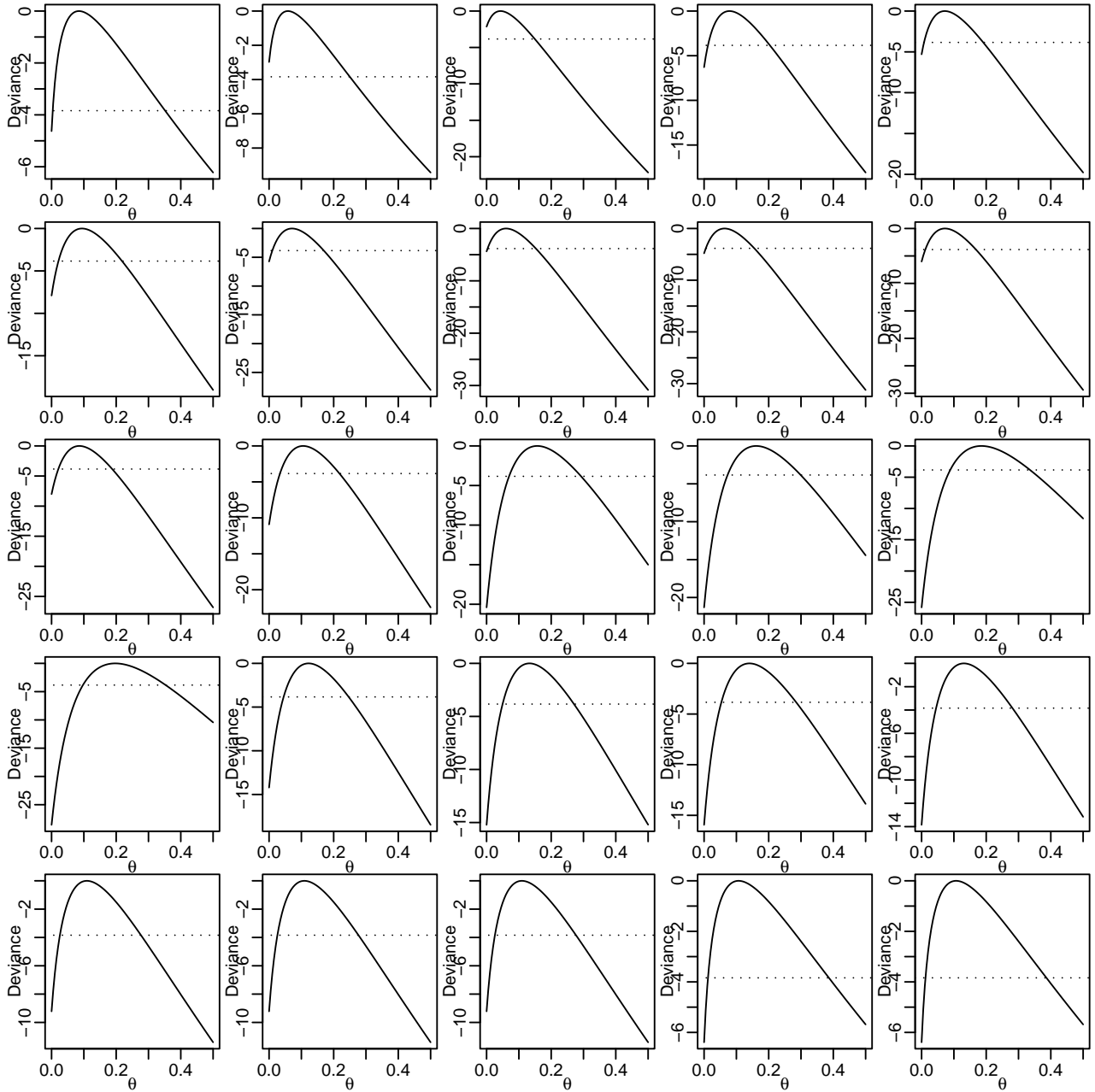


Figura 7: Função deviance das estimativas de θ obtidas para os dados das 25 avaliações de MSC feitas no talhão 303 da fazenda Vale Verde.

Referências

- Alellyx (2004). Alellyx identifica vetor da morte súbita dos citros, *Alellyx*. URL: http://www.gravena.com.br/dicas_alellyx_vetor_MSC.htm.
- Bassanezi, R. B., Fernandes, N. G. & Yammamoto, P. T. (2003). Morte súbita do citros, *Technical report*, Fundecitrus.
- Griffiths, D. A. (1973). Maximum likelihood estimation for the beta-binomial distribution and an application to household distribution of total number of a disease, *Biometrics* **29**(1): 637–

648. Washington.

- Madden, L. V. & Hughes, G. (1995). Plant disease incidence: Distributions, heterogeneity, and temporal analysis, *Phytopathology* **33**: 529–564.
- Mendonça, L. A., Bearzoti, E., Aquino, L. H. & Ferreira, D. F. (2004). Métodos de estimação por intervalo do coeficiente de correlação intraclasse, para dados com distribuição beta-binomial, *Revista Matemática Estatística* **22**: 7–21. São Paulo.
- R Development Core Team (2005). *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.
URL: <http://www.R-project.org>
- Venables, W. N. & Ripley, B. D. (2002). *Modern Applied Statistics with S*, fourth edition edn, Springer.