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

21 de fevereiro de 2006

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 estimaço dos parâmetros. Dessa forma, devem ser utilizados procedimentos iterativos de minimização numérica da função de verossimilhança. É demonstrado o uso das funções `optimize()`, `optim()`, `nlm()` e nlminb() disponíveis em R para minimização 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*, onde 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 $Y \sim Bin(n, \pi)$, com n sento o número de plantas por *quadrat* e π a taxa de incidência da doença no talhão. Se $n \to \infty$ e $\pi \to 0$, pode-se utilizar a distribuição de Poisson, assumindo que $\theta \approx n * \pi$.

O teste da hipótese de aleatoriedade espacial nesse caso, consiste 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 o 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 aleatória, D será igual a 1 e pode-se assumir que a incidência segue uma distribuição binomial. Se a ditribuição 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

$$V_o(Y) = \sum_i (y_i - np)^2/(N-1) \tag{1}$$

onde 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

$$Vb_t(Y) = np(1-p)\ . \tag{2}$$

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

$$Vp_t(Y) = E(Y) = \sum_i Y_i n_i / \sum_i n_i, \tag{3}$$

onde n_i é o número de plantas no í-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 onde 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 ou 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 minimização 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 compondo-se a distribuição binomial com a densidade beta para π, (SKELLAM 1948). A função de densidade beta pode ser escrita de várias formas, uma delas é:

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

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

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

$$P(Y = y|\alpha, \beta) = \left(\begin{array}{c} n \\ y \end{array} \right) \frac{B(y+\alpha, n-y+\beta)}{B(\alpha, \beta)}\ , \tag{5}$$

onde $x = 0, 1, ..., n$.

Para uma simplicidade e interpretabilidade dos parâmetros, sugere-se uma reparametrização (GRIFFITHS 1979). Os novos parâmetros são: $p = \frac{\alpha}{\alpha+\beta}$ e $\theta = \frac{1}{\alpha+\beta}$. onde p é o valor esperado do parâmetro π da distribuição binomial e θ é uma medida da variação de p.

A função de densidade pode ser então reescrita na forma:

$$P(Y = y|p, \theta) = \begin{pmatrix} n \\ y \end{pmatrix} \frac{\prod_{i=0}^{x-1}(p + i\theta) \coprod_{i=0}^{n-x-1}(1 - p + i\theta)}{\coprod_{i=0}^{x-1}(l + i\theta)} \ . \tag{6}$$

onde $0 < p < 1$ e $\theta \leq 0$. Se $\theta = 0$ a densidade é a binomial.

O valor esperado de Y e sua variância são dados por: E(X) = np e $Var(Y) = \frac{np(1-p)(1+n\theta)}{(1+\theta)}$. Observa-se assumindo-se que p varia, isto é, se $\theta > 0$, 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 funçã será terá p e θ como argumentos e definir internamente $\alpha = \frac{\pi}{\theta}$ e $\beta = \frac{(1-\pi)}{\theta}$.

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 logarítima reduz esse problema:

```
> lbeta(999, 999)

[1] -1387.096
```

A escala logarítima será utilizada no cálculo da densidade:

```
> 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)
+     return(dens)
+ }
```

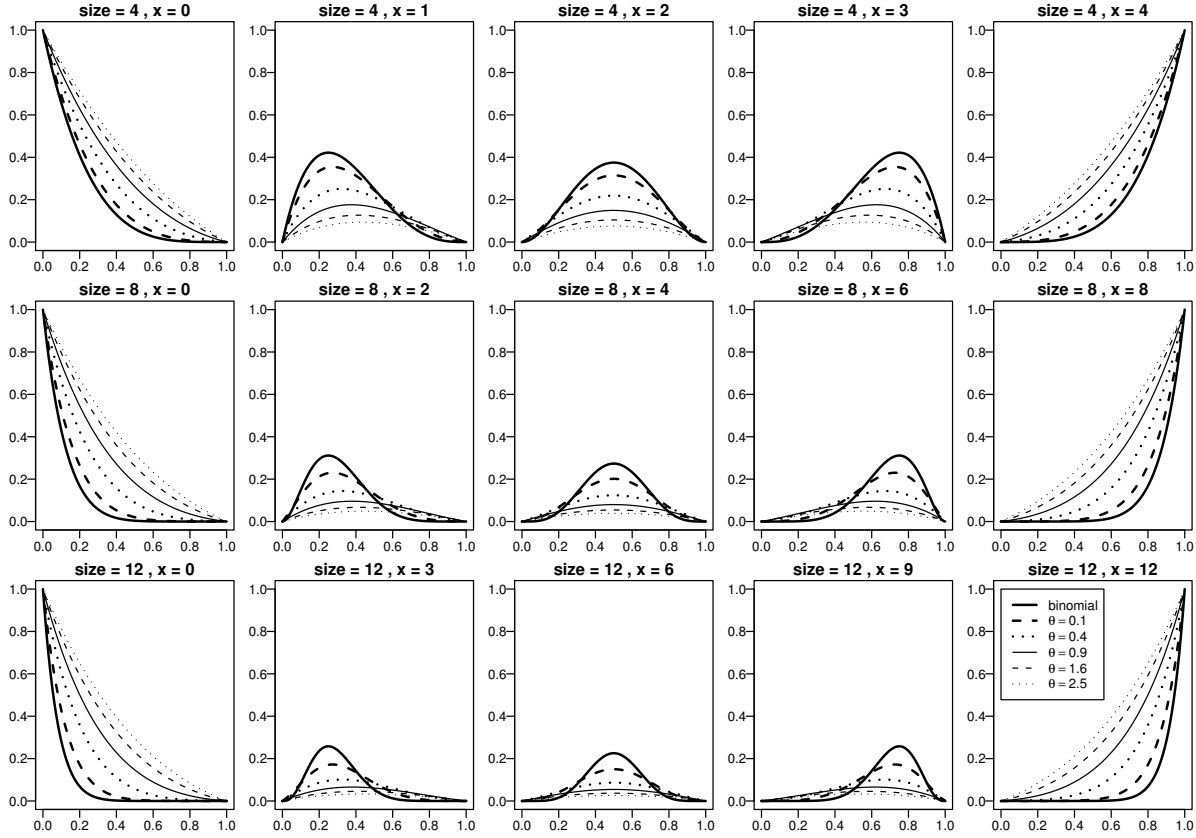

Figura 1: Densidade de probabilidade da distribuição beta-binomial. No eixo horizontal, está representado o valor do parâmetro π e no eixo vertical está representado o valor da densidade de probabilidade.

A partir dessa função podemos estudar a densidade de probabilidades em gráficos de linhas, plotando o valor da densidade variando os parâmetros da distribuição. Na Figura 1, podemos visualizar 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:

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

[1] 0.9999996
```

A precisão pode ser melhorada considerando que a distribuição beta-binomial se aproxima da distribuição binomial quando $\theta \approx 0$. Assim, podemos 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. Em 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)
+     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 geradora de números aleatórios.

A função de distribuição é obtida somando-se as densidades avaliadas em `0:size`:

```
> pbetabinom <- function(q, size, prob, theta, lower.tail = TRUE,
+     log.p = FALSE) {
+     pars <- data.frame(rbind(q, size, prob, theta, 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, calcula-se a probabilidade acumulada em $x = \{0, ..., \text{size}\}$. O resultado é o menor valor de x cuja probabilidade acumulada ultrapassa a probabilidade `p`.

```
> qbetabinom <- function(p, size, prob, theta, lower.tail = TRUE,
+     log.p = FALSE) {
+     if (log.p)
+         p <- log(p)
+     ql <- function(y) (0:y[2])[min(which(cumsum(dbetabinom(0:y[2],
+         y[2], y[3], y[4])) > y[1]))]
+     q <- new("numeric", sapply(data.frame(rbind(q, 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 podemos 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, podemos evitar o uso da função acumulada inversa. Podemos utilizar uma função para sorter valores de um conjunto finito de valores, com probabilidade dada pela função de densidade. Em R, podemos utilizar a função `sample()` para isso.

```
> rbetabinom <- function(n, size, prob, theta) {
+     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 de 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 minimização 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 logarítimo da função de verossimilhança.

No procedimento de minimização numérica, será utilizado a função `dbetabinom()` definida anteriormente, embora de já esteja implementada no pacote **rmutil**. A função `dbetabinom()` do pacote **rmutil** apresenta um inconveniente impossibilita utilizá-la na função de minimização 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 minimização numérica.

Em 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, elas buscarão o valor que minimiza a função objetivo. Neste caso, para simplificar, vamos utilizá-las para encontrar a solução que minimiza o negativo do logarítimo 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. Essa 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
```

Supoẽ-se que não conhecemos a espressão analítica do estimador de máxima verossimilhança de π. Podemos maximizar numéricamente a função de verossimilhança, em relação a π. Devemos 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 logarítimo das densidades avaliadas em dada valor de x informado, considerando o parâmetro π e n 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 logarítimo da verossimilhança de -146.2198. Conhecendo a expressão analítica do estimador, podemos 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 algorítimo 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 sem 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 logarítimo 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 $\theta = 0.1$ utilizamos 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] 1.263814e-05 1.490307e-05

$hessian
           [,1]      [,2]
[1,] 1769.64299 -67.98567
[2,]  -67.98567 414.92747

$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 9.317555e-05
[2,] 9.317555e-05 2.425327e-03
```

3.3 Função `optim()`

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

Utilizamos a função `optim()` para a estimação de π e θ. Neste caso o valor verdadeiro de θ é zero. Podemos 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 ouver.

Encontrando as estimativas de máxima verossimilhança da amostra 2. Vamos também obter o hessiano numérico para aproximar a matriz de informação de Fisher observada, colocando o argumento `hessian=TRUE`.

```
> 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
```

Obtemos a matriz de informação de Fisher aproximada, 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:

```
> str(nlminb(c(0.3, 0.1), nlb, hessian = TRUE, size = 5, y = y2,
+     lower = c(0, 0), upper = c(1, Inf)))

List of 6
 $ par        : num [1:2] 0.000 0.179
 $ objective  : num 152
 $ convergence: int 1
 $ message    : chr "false convergence (8)"
 $ iterations : int 1
 $ evaluations: Named int [1:2] 3 2
  ..- attr(*, "names")= chr [1:2] "function" "gradient"
```

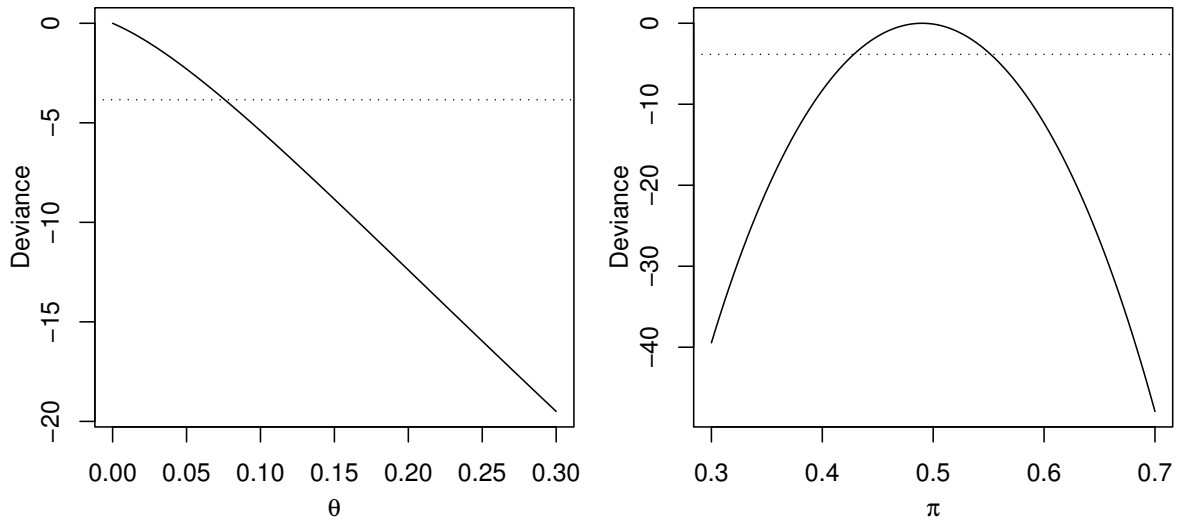

Figura 2: Função desvio para os parâmetros θ (esquerda) e π (direita) da amostra 1. A linha pontilhada corta a curva da função desvio nos limites de confiança de 95%.

4 Funções para estimação por máxima verossimilhança

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

4.1 Função `fitdistr()`

A função `fitdistr()` do pacote **MASS** é específica para estimação por maxima 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 da função de densidade nomeando-os, por exemplo o argumento `size`. Essa função retorna também as estimativas dos erros padrões, obtidos através do hessiano numérico.

Na amostra 1 não foi possível utilizar a função `fitdistr()`. Sabemos que o valor do parâmetro θ é zero e é o limite inferior do espaço paramétrico. A dificuldade aqui é em encontrar o hessiano numérico, pois para isso necessita-se 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, vamos apenas vizualizar 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 paramétrio. Para isto, informa-se os limites de cada um dos parâmetros nos argumentos `lower` e `upper`.

```
> (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 logarítmo 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 podemos extrair a matriz de informação de fischer, obtida através do hessiano numérico e as verossimilhanças perfilhadas para os parâmetros e com isso, construir intervalos de confiança assintóticos e vizualizar as verossimilhanças perfilhadas. Esses resultados são extraidos 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 $\theta = 0.1$, temos:

```
> 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 Fischer 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 resulado, 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 vizualizado 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 vizualização 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, podemos vizualizar esse gráfico para os parâmetros estimados na 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 numéricamente. Pode-se obter os vaores 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
```

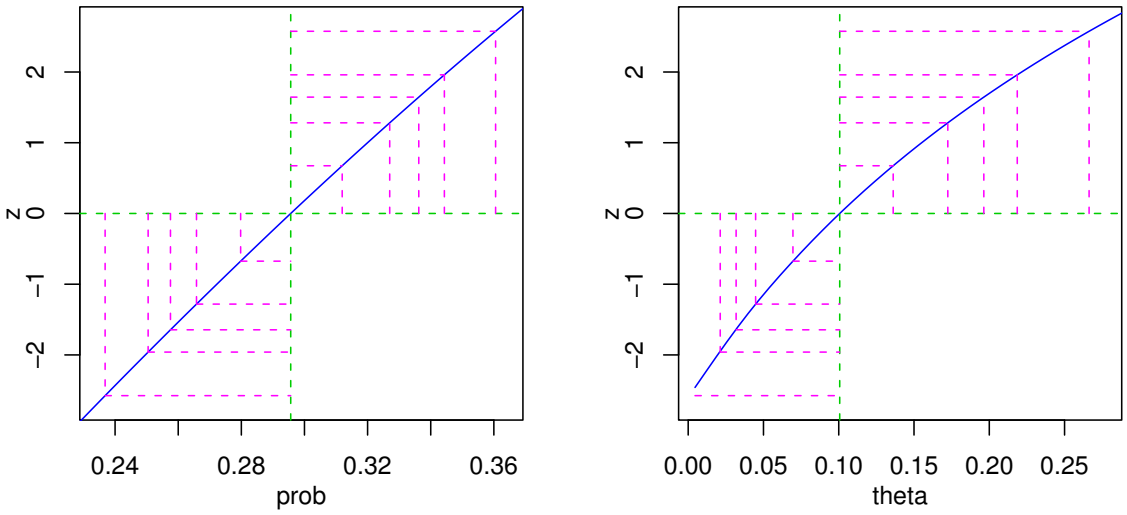

Figura 3: Verossimilhanças perfilhadas padronizadas para a amostra 2.

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 são dados de PPC em um talhão do município de Itajobi - MG. O segundo são dados de MSC em um talhão localizado no município de Comendador Gomes - MG.

Na aplicação a dados de doenças de plantas, deve ser feita a contagem do número de plantas doentes por *quadrat* e aplicar a metodologia nas contagens. 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 pcnidiósporos. 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

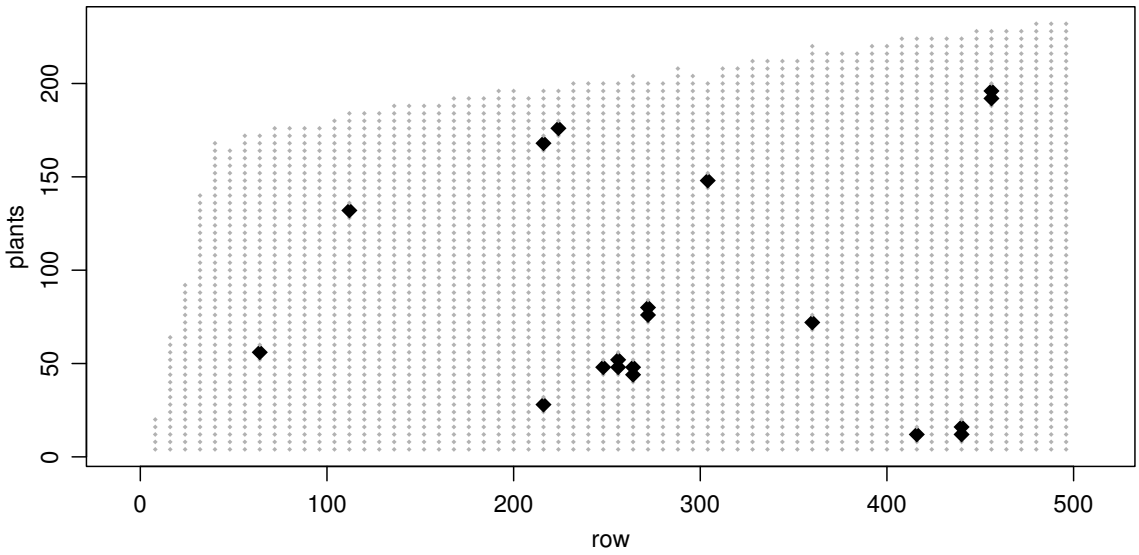

Figura 4: Incidência de Pinta Preta dos Citrus. Pontos menores indicam plantas sadias e pontos maiores indicam plantas doentes. Posições com ausência de pontos, indica ausência de plantas.

fase assexual, a disseminação ocorre por escorrimento da água da chuva ou irrigação e a contaminação de uma árvore para outra se dá pelo contado dessa 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)
```

```
[1] TRUE
```

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

```
x = 8 y = 4
```

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

Precisamos avaliar se podemos assumir que a incidência é constante em toda a área em estudo. A seguir utilizaremos 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
[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 vamos explorar as verossimilhanças perfilhadas fazendo a estimação utilizando internamente 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

> 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.

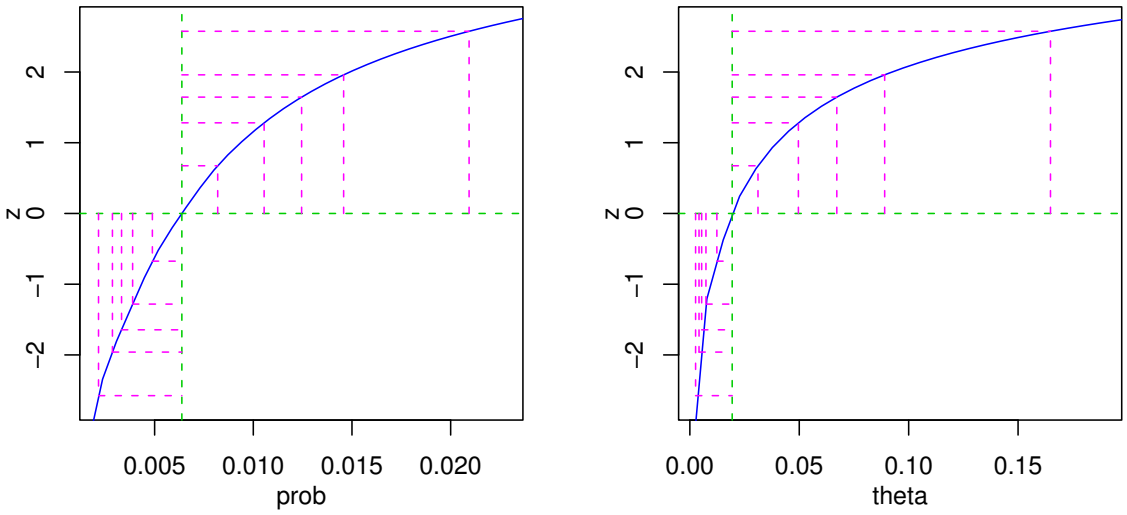

Figura 5: Verossimilhanças perfilhadas padronizadas dos parâmetros da distribuição beta-binomial ajustada aos dados de Pinta Preta dos Citrus.

```
> 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 agregaço baseado no índice de dispersão binomial e Poisson. Aplicando aos dados, fazendo a contagem por *quadrats* de tamanho 10×10, temos:

```
> 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 é avalia a verossimilhança da distribuição beta-binomial com parãmetro θ igual ao estimado com os dados e 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 apresentando 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 plotamos a última avaliação por restrição de espaço, mas esse mapa 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)
```

Os códigos atribuidos 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. Consideremos 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 Av16
0 922 918 872 856 847 796 706 702 645  628  480  423  368  356  325  307
1  17  21  46  59  68 117 205 194 251  266  402  438  477  461  488  492
2   5   5  17  17  17  17  19  34  34   36   44   55   55   69   70   80
3   1   1  10  13  13  15  15  15  15   15   19   29   45   59   62   66
  Av17 Av18 Av19 Av20 Av21 Av22 Av23 Av24 Av25
0  224  167  163  120   69   69   69   27   27
1  500  554  554  385  223  217  211  125  125
2  108  108  112  108  110   98  100   20   20
3  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, adotamos *quadrats* 2×4. Vamos considerar nessa análise a incidência como sendo plantas identificadas pelos códigos 1, 2 e 3:

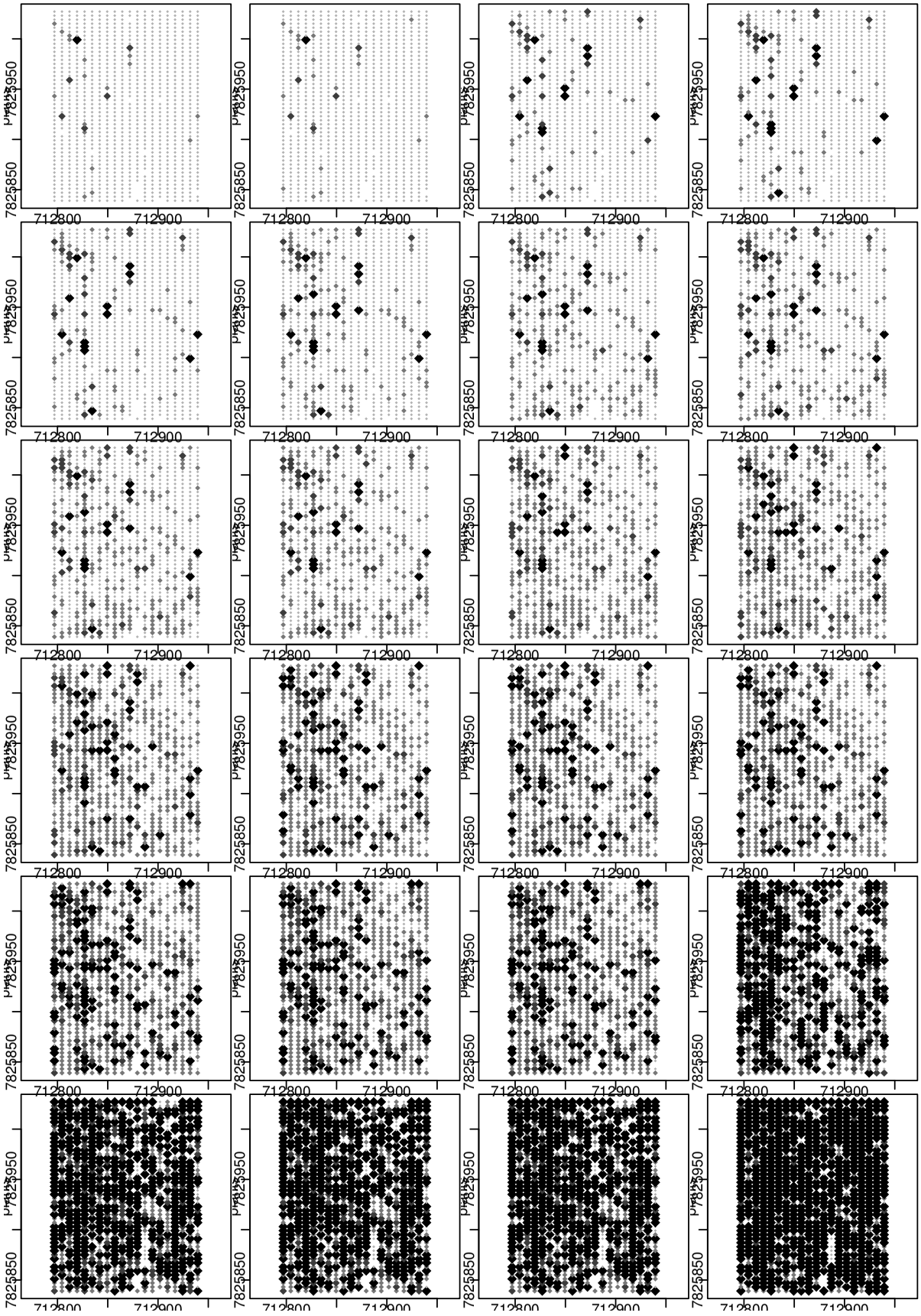

Figura 6: Mapas da evolução da Morte Súbita, ao longo de 24 avaliações feitas no talhão 303 da fazenda Vale Verde.

```
> (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 padão 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 vizualização 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 algorítmos de minimização numérica disponíveis em R, estimativa dos erros dos estimadores a partir do hessiano numérico, intervalos de confianą e teste de razão de verossimilhanças.

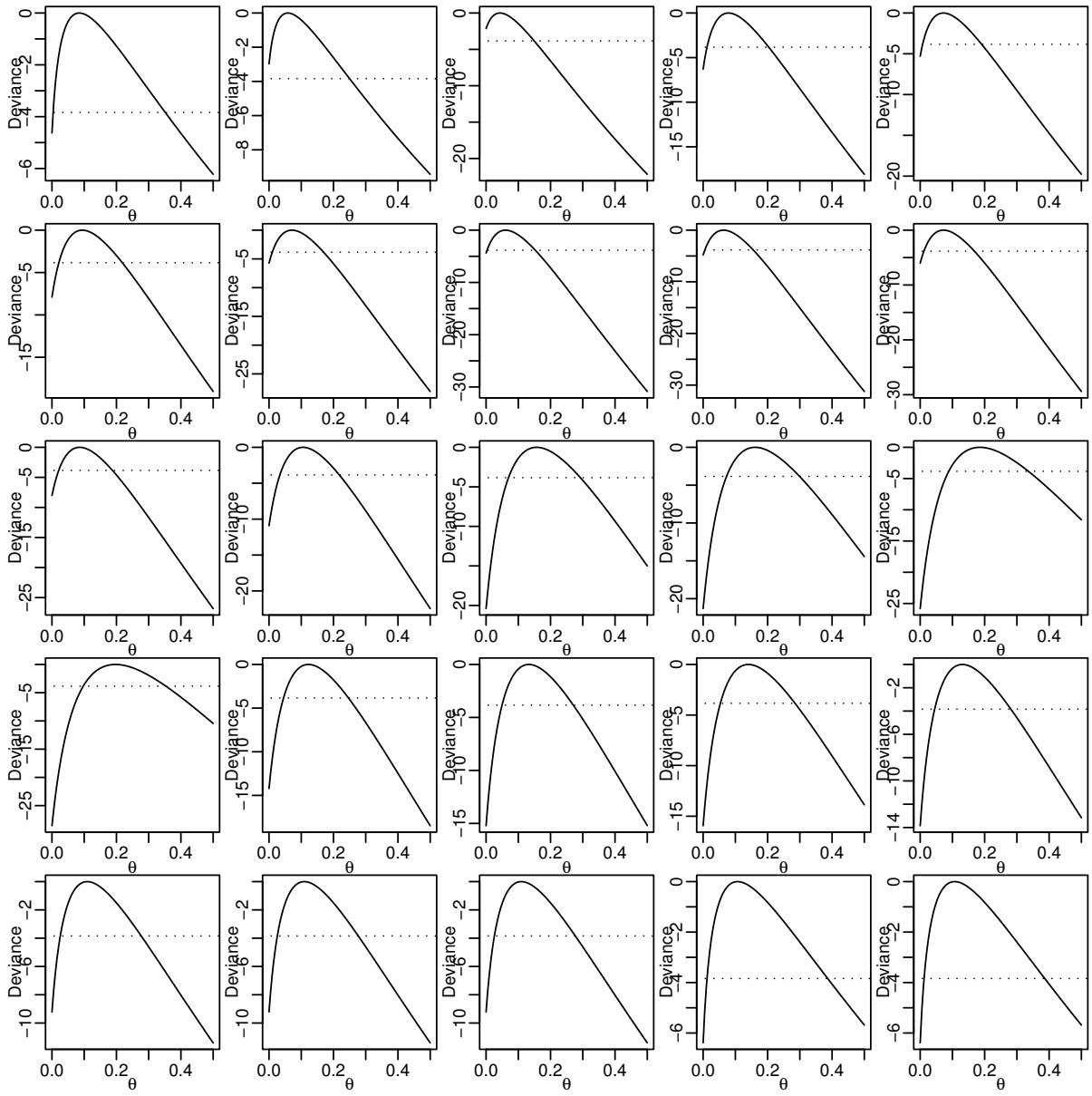

Figura 7: Função deviance das estimativas de θ obtidas para os dados das 25 avaliações de MSC feitas no talhão 303 da fazenda Vale Verde.

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 padrões aproximados numericamente, obtenção da verossmilhança 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, onde observou-se 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 padrã 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 avaliaçoes 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.

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. (1979). Maximum likelihood estimation for the beta-binomial distribution and an aplication to household distribution of total number of a disease, *Biometrics* **35**(1): 281–293. Washington.

MADDEN, L. V. & HUGHES, G. (1995). Plant disease incidence: Distributions, heterogeneity, and temporal analysis, *Phytopathology* **33**: 529–564.

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. *http://www.R-project.org

SKELLAM, J. G. A. (1948). A probability distribution derived from the binomial distribution by regarding the probability of sucess as variable between the sets of trials, *Journal of Royal Statistical Society, Series B* **10**: 257–261. Cambridge.

Venables, W. N. & Ripley, B. D. (2002). *Modern Applied Statistics with S*, fourth edition edn, Springer.