

1 Vetor Gradiente e Matriz Hessiana

Considere a função

$$N_t = f(t) = \frac{KN_0}{N_0 + (K - N_0) \exp \{-rt\}} \quad (1)$$

onde N_t é o tamanho da população no tempo t , e N_0 é o tamanho inicial. Considere $N_0 = 2$, e use como função objetivo a perda quadrática.

$$Q(r, K) = \sum_{i=1}^m (N_i - f_{r,K}(t_i))^2 \quad (2)$$

1.1 Vetor Gradiente

O vetor gradiente da função Q dos parâmetros r e K é dada por.

$$\nabla Q(r, K) = \left[\frac{\partial Q}{\partial K}(r, K), \frac{\partial Q}{\partial r}(r, K) \right] \quad (3)$$

Vamos então começar encontrando o segundo elemento do vetor gradiente

$$\begin{aligned} \frac{\partial Q}{\partial r}(r, K) &= \frac{\partial}{\partial r} \left[\sum_{i=1}^m (N_i - f_{r,K}(t_i))^2 \right] = \sum_{i=1}^m \left[\frac{\partial}{\partial r} (N_i - f_{r,K}(t_i))^2 \right] \\ &= \sum_{i=1}^m \left[\frac{\partial}{\partial r} (N_i^2 - 2N_i f_{r,K} + f_{r,K}^2) \right] = \sum_{i=1}^m \left[(-2N_i \frac{\partial f_{r,K}}{\partial r} + \frac{\partial f_{r,K}^2}{\partial r}) \right] \\ &= \sum_{i=1}^m \left[(-2N_i \frac{\partial f_{r,K}}{\partial r} + 2f_{r,K} \frac{\partial f_{r,K}}{\partial r}) \right] \end{aligned}$$

Temos que

$$\begin{aligned} \frac{\partial f_{r,K}}{\partial r} &= \frac{\partial}{\partial r} \frac{KN_0}{N_0 + (K - N_0) \exp \{-rt\}} = KN_0 \frac{\partial}{\partial r} (N_0 + (K - N_0) \exp \{-rt\})^{-1} \\ &= KN_0 (-1) (N_0 + (K - N_0) \exp \{-rt\})^{-2} ((K - N_0) \exp \{-rt\} (-t)) \\ &= \frac{KN_0 ((K - N_0) \exp \{-rt\} t)}{(N_0 + (K - N_0) \exp \{-rt\})^2} \end{aligned}$$

Portanto

$$\frac{\partial Q}{\partial r}(r, K) = \sum_{i=1}^m \left[2N_i \frac{KN_0 ((K - N_0) \exp \{-rt\} t)}{(N_0 + (K - N_0) \exp \{-rt\})^2} + 2KN_0 \frac{KN_0 ((K - N_0) \exp \{-rt\} t)}{(N_0 + (K - N_0) \exp \{-rt\})^3} \right]$$

Agora vamos encontrar o primeiro elemento do vetor gradiente.

$$\begin{aligned}\frac{\partial Q}{\partial K}(r, K) &= \frac{\partial}{\partial K} \left[\sum_{i=1}^m (N_i - f_{r,K}(t_i))^2 \right] = \sum_{i=1}^m \left[(-2N_i \frac{\partial f_{r,K}}{\partial K} + \frac{\partial f_{r,K}^2}{\partial K}) \right] \\ &= \sum_{i=1}^m \left[(-2N_i \frac{\partial f_{r,K}}{\partial K} + 2f_{r,K} \frac{\partial f_{r,K}}{\partial K}) \right]\end{aligned}$$

Temos que

$$\frac{\partial f_{r,K}}{\partial K} = \frac{\partial}{\partial K} \frac{KN_0}{N_0 + (K - N_0) \exp \{-rt\}} = \frac{N_0(-\exp \{-rt\}N_0 + N_0)}{(N_0 + (K - N_0) \exp \{-rt\})^2}$$

Portanto

$$\frac{\partial Q}{\partial K}(r, K) = \sum_{i=1}^m \left[(-2N_i \frac{N_0(-\exp \{-rt\}N_0 + N_0)}{(N_0 + (K - N_0) \exp \{-rt\})^2} + 2 \frac{N_0KN_0(-\exp \{-rt\}N_0 + N_0)}{(N_0 + (K - N_0) \exp \{-rt\})^3}) \right]$$

1.2 Matriz Hessiana

A Matriz Hessiana da função Q dos parâmetros r e K é dada por.

$$\mathbf{H}_Q(r, K) = \begin{bmatrix} \frac{\partial^2 Q}{\partial r^2}(r, K) & \frac{\partial^2 Q}{\partial K \partial r}(r, K) \\ \frac{\partial^2 Q}{\partial r \partial K}(r, K) & \frac{\partial^2 Q}{\partial K^2}(r, K) \end{bmatrix} \quad (4)$$

Computando o elemento (1,1) desta matriz

$$\begin{aligned}\frac{\partial^2 Q}{\partial r^2}(r, K) &= \frac{\partial}{\partial r} \frac{\partial Q}{\partial r} = \frac{\partial}{\partial r} \sum_{i=1}^m \left[(-2N_i \frac{\partial f_{r,K}}{\partial r} + 2f_{r,K} \frac{\partial f_{r,K}}{\partial r}) \right] \\ &= \sum_{i=1}^m \left[-2N_i \frac{\partial^2 f_{r,K}}{\partial r^2} + \frac{\partial}{\partial r} \left(2f_{r,K} \frac{\partial f_{r,K}}{\partial r} \right) \right] \\ &= \sum_{i=1}^m \left[-2N_i \frac{\partial^2 f_{r,K}}{\partial r^2} + 2 \frac{\partial f_{r,K}}{\partial r} \frac{\partial f_{r,K}}{\partial r} + 2f_{r,K} \frac{\partial^2 f_{r,K}}{\partial r^2} \right] \\ &= \sum_{i=1}^m \left[-2N_i \frac{\partial^2 f_{r,K}}{\partial r^2} + 2 \left(\frac{\partial f_{r,K}}{\partial r} \right)^2 + 2f_{r,K} \frac{\partial^2 f_{r,K}}{\partial r^2} \right]\end{aligned}$$

Analogamente temos

$$\begin{aligned}\frac{\partial^2 Q}{\partial K^2}(r, K) &= \sum_{i=1}^m \left[-2N_i \frac{\partial^2 f_{r,K}}{\partial K^2} + 2 \left(\frac{\partial f_{r,K}}{\partial K} \right)^2 + 2f_{r,K} \frac{\partial^2 f_{r,K}}{\partial K^2} \right] \\ \frac{\partial^2 Q}{\partial K \partial r}(r, K) &= \sum_{i=1}^m \left[-2N_i \frac{\partial^2 f_{r,K}}{\partial K \partial r} + 2 \frac{\partial f_{r,K}}{\partial K} \frac{\partial f_{r,K}}{\partial r} + 2f_{r,K} \frac{\partial^2 f_{r,K}}{\partial K \partial r} \right]\end{aligned}$$

onde

$$\frac{\partial^2 f_{r,K}}{\partial r^2} = \frac{K \exp \{-2rt\} N_0 t^2 (-\exp \{rt\} N_0 - N_0 + K)(K - N_0)}{(N_0 + (K - N_0) \exp \{-rt\})^3}$$

$$\frac{\partial^2 f_{r,K}}{\partial K^2} = \frac{2 \exp \{-rt\} N_0 (-\exp \{-rt\} N_0 + N_0)}{(N_0 + (K - N_0) \exp \{-rt\})^3}$$

$$\frac{\partial^2 f_{r,K}}{\partial K \partial r} = \frac{\exp \{-rt\} N_0^2 t (\exp \{-rt\} N_0 - N_0 + 2K - K \exp \{-rt\})}{(N_0 + (K - N_0) \exp \{-rt\})^3}$$

Os cálculos são apresentados no apêndice B. A seguir é apresentado o gráfico de nível da função Q , podemos observar que o mínimo esta numa vizinhança de $r \in (0, 0.2)$ e $K \in (800, 1200)$.

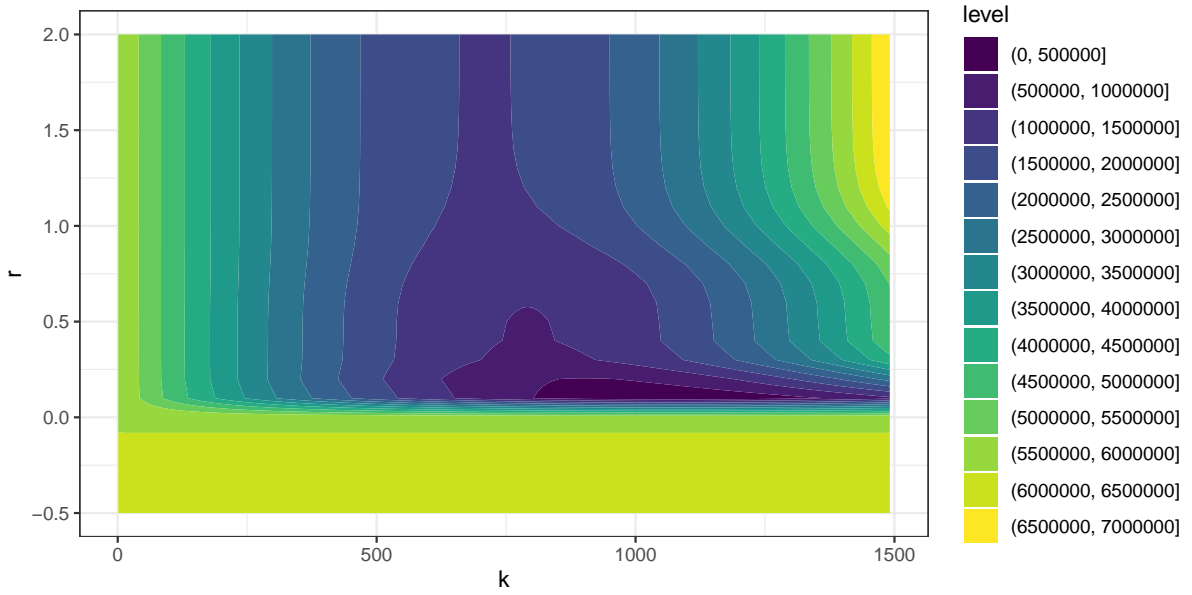


Figura 1: Curvas de nível de Q .

2 Algoritmo Newton-Raphson

Usando o método iterativo de Newton-Raphson

$$\hat{\theta}_{k+1} = \hat{\theta}_k - [\mathbf{H}(\hat{\theta}_k)]^{-1} \nabla Q(\hat{\theta}_k) \quad (5)$$

Onde $\theta_k = [K_k, r_k]$, Nota-se que o método não é eficiente quando o chute inicial de r é ruim, por vezes caindo em mínimos locais ou por ventura dando erro por cair em uma divisão por zero numérico. Observa-se que um bom *range* de valores iniciais para r é $(.07, .16)$. Optou-se então por testar o método para os seguintes pontos iniciais: $\theta_0^{(1)} = [900, 0.1]$, $\theta_0^{(2)} = [500, 0.1]$, $\theta_0^{(3)} = [1024, 0.15]$, $\theta_0^{(4)} = [2000, 0.15]$ e $\theta_0^{(5)} = [1024, 0.5]$.

Tabela 1: Resultados do método Newton-Raphson.

θ_0	\hat{r}	\hat{K}	iter	$Q(\hat{r}, \hat{K})$
$K_0 = 900, r = 0.1$	0.11793	1033.5623	3	83240.55
$K_0 = 500, r = 0.1$	0.11795	1033.5153	5	83240.49
$K_0 = 1024, r = 0.15$	0.11795	1033.5156	3	83240.49
$K_0 = 2000, r = 0.15$	0.11795	1033.5164	4	83240.49
$K_0 = 1024, r = 0.5$	2.18865	709.2222	7	1477923

Abaixo é plotado o gráfico de $N_t = f_{r,K}(t)$ com os valores estimados. Podemos observar um ajuste razoável.

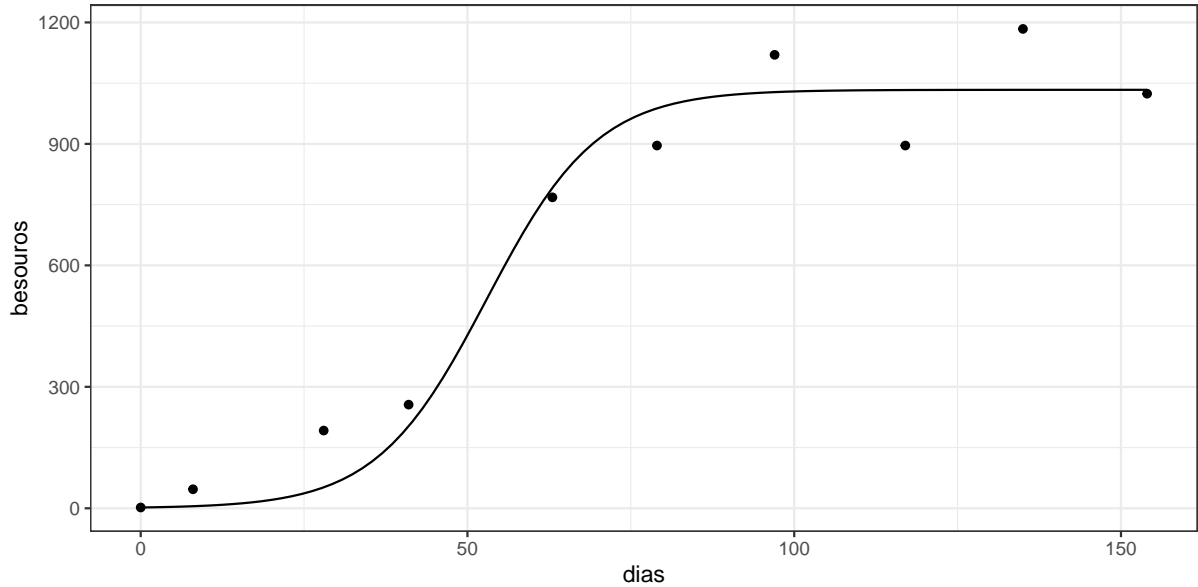


Figura 2: Curva ajustada nos dados a partir do método de Newton-Raphson ($K = 1033.56, r = 0.11795$).

3 Algoritmo *line-search*

Para implementação do *line-search*, devemos encontrar $\gamma_{k+1}(0, \infty)$ tal que $Q(\boldsymbol{\theta}_k - \gamma_k \mathbf{p}_k)$ seja ótima em função de γ , em que $\mathbf{p}_k = [\mathbf{H}(\hat{\boldsymbol{\theta}}_k)]^{-1} \nabla Q(\hat{\boldsymbol{\theta}}_k)$

$$Q(\boldsymbol{\theta}_k - \gamma_k \mathbf{p}_k) = \sum_{i=1}^m (N_i - f_{\gamma,r,K}(t_i))^2$$

em que

$$f_{\gamma,r,K}(t_i) = \frac{(K - \gamma \mathbf{p}_1) \mathbf{N}_0}{N_0 + (K - \gamma \mathbf{p}_1 - \mathbf{N}_0) \exp \{-(\mathbf{r} - \gamma \mathbf{p}_2) \mathbf{t}\}}$$

Para tanto devemos obter o valor de γ que zera a função abaixo.

$$\frac{\partial Q}{\partial \gamma}(\gamma, r, K) = \sum_{i=1}^m \left[(-2N_i \frac{\partial f_{\gamma,r,K}}{\partial \gamma} + 2f_{\gamma,r,K} \frac{\partial f_{\gamma,r,K}}{\partial \gamma}) \right]$$

Todavia observou-se que $\partial f_{\gamma,r,K} / \partial \gamma$ possui uma forma fechada bem complexa, impossibilitando uma solução analítica, portanto optou-se por utilizar métodos numéricos já implementados no R para computar γ_k , adicionalmente foi necessário restringir $\gamma > 0.5$ pois sem essa restrição usualmente apontava um $\gamma_k \approx 0$ segundo as ressalvas citadas no método de Newton-Raphson.

Tabela 2: Resultados do método *line-search*.

θ_0	\hat{r}	\hat{K}	iter	$Q(\hat{r}, \hat{K})$
$K_0 = 900, r = 0.1$	0.11795	1033.5153	4	83240.55
$K_0 = 500, r = 0.1$	0.11795	1033.5152	4	83240.49
$K_0 = 1024, r = 0.15$	0.11795	1033.5156	3	83240.49
$K_0 = 2000, r = 0.15$	0.11795	1033.5164	3	83240.49
$K_0 = 1024, r = 0.5$	0.11795	1033.5153	28	1477923

Com a implementação do *line-search*, observou-se um ganho sob o número de interações em relação ao método anterior e adicionalmente os pontos iniciais que levaram a não convergência para o mínimo global dessa vez convergiram corretamente.

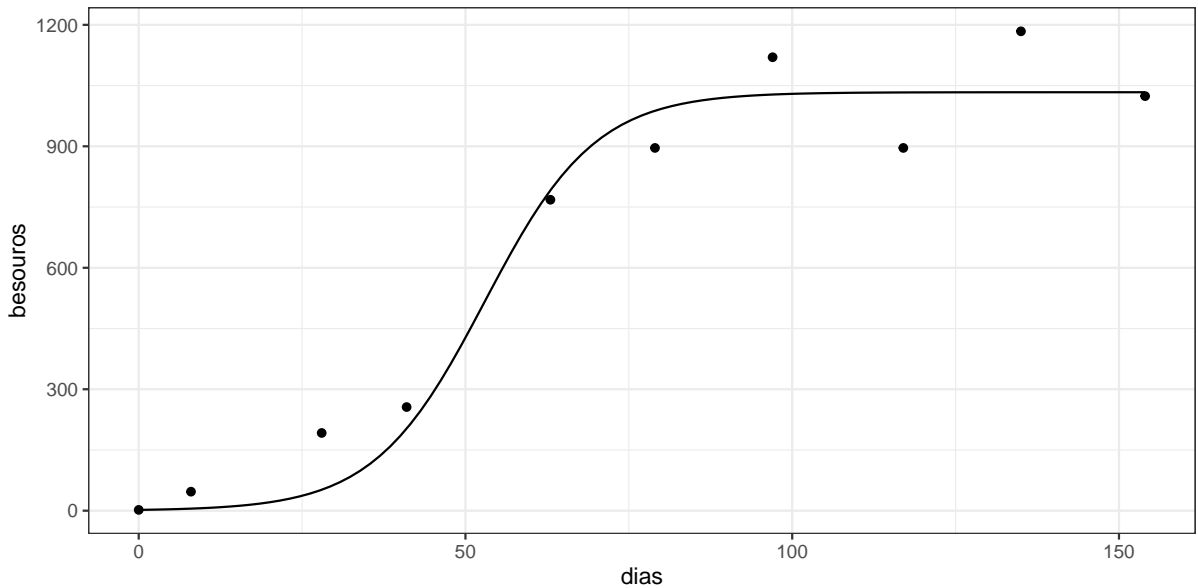


Figura 3: Curva ajustada nos dados a partir do método de *line-search* ($K = 1033.56, r = 0.11795$).

4 Algoritmo Escore de fisher

Assumindo $\log(N_t) \sim N(\log\{f_{r,K}(t)\}, \sigma^2)$, a verossimilhança é dada por

$$L(\boldsymbol{\theta}) = \prod_{i=1}^m \left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp \left(\frac{-1}{2\sigma^2} (\log(N_t) - \log(f_{r,K}))^2 \right) \right]$$

Sendo $\boldsymbol{\theta} = [r, K, \sigma^2]^\top$, portanto a log-verossimilhança é dada por

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^m \left[\frac{-1}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} (\log(N_t) - \log(f_{r,K}))^2 \right]$$

Vetor gradiente e informação de fisher

O vetor gradiente da função ℓ dos parâmetros r , K e σ^2 é dada por.

$$\nabla \ell(\boldsymbol{\theta}) = \left[\frac{\partial \ell}{\partial K}(\boldsymbol{\theta}), \frac{\partial \ell}{\partial r}(\boldsymbol{\theta}), \frac{\partial \ell}{\partial \sigma^2}(\boldsymbol{\theta}) \right] \quad (6)$$

em que

$$\frac{\partial \ell}{\partial r}(\boldsymbol{\theta}) = \frac{-1}{2\sigma^2} \sum_{i=1}^m \left[-2 \log(N_t) \frac{\partial \log(f_{r,K})}{\partial r} + \frac{\partial \log^2(f_{r,K})}{\partial r} \right]$$

$$\frac{\partial \ell}{\partial K}(\boldsymbol{\theta}) = \frac{-1}{2\sigma^2} \sum_{i=1}^m \left[-2 \log(N_t) \frac{\partial \log(f_{r,K})}{\partial K} + \frac{\partial \log^2(f_{r,K})}{\partial K} \right]$$

$$\frac{\partial \ell}{\partial \sigma^2}(\boldsymbol{\theta}) = \sum_{i=1}^m \left[\frac{-1}{2\sigma^2} + \frac{1}{2(\sigma^2)^2} (\log(N_t) - \log(f_{r,K}))^2 \right]$$

Temos que as componentes do vetor gradiente depende de algumas derivadas parciais, que são apresentadas abaixo.

$$\frac{\partial \log(f_{r,K})}{\partial r} = \frac{1}{f_{r,K}} \frac{\partial f_{r,K}}{\partial r}$$

$$\frac{\partial \log^2(f_{r,K})}{\partial r} = 2 \log(f_{r,K}) \frac{\partial \log(f_{r,K})}{\partial r} = 2 \log(f_{r,K}) \frac{1}{f_{r,K}} \frac{\partial f_{r,K}}{\partial r}$$

$$\frac{\partial \log(f_{r,K})}{\partial K} = \frac{1}{f_{r,K}} \frac{\partial f_{r,K}}{\partial K}$$

$$\frac{\partial \log^2(f_{r,K})}{\partial K} = 2 \log(f_{r,K}) \frac{\partial \log(f_{r,K})}{\partial K} = 2 \log(f_{r,K}) \frac{1}{f_{r,K}} \frac{\partial f_{r,K}}{\partial K}$$

Já a informação de fisher é dada pelo negativo da esperança da matriz hessiana de $\ell(\boldsymbol{\theta})$.

$$\mathcal{I}(\boldsymbol{\theta}) = -\mathbb{E}[\mathbf{H}_\ell(\boldsymbol{\theta})] \quad (7)$$

em que

$$\mathbf{H}_\ell(\boldsymbol{\theta}) = \begin{bmatrix} \frac{\partial^2 \ell}{\partial r^2}(\theta) & \frac{\partial^2 \ell}{\partial r \partial K}(\theta) & \frac{\partial^2 \ell}{\partial r \partial \sigma^2}(\theta) \\ - & \frac{\partial^2 \ell}{\partial K^2}(\theta) & \frac{\partial^2 \ell}{\partial K \partial \sigma^2}(\theta) \\ - & - & \frac{\partial^2 \ell}{\partial (\sigma^2)^2}(\theta) \end{bmatrix} \quad (8)$$

Apêndice A

A.1 Derivadas parciais no R

```

1 ft <- function(param, data, N0){
2   t <- data$dias
3   k <- param[1]
4   r <- param[2]
5   output <- (k*N0) / (N0 + (k-N0)*exp(-r*t))
6   return(output)
7 }
8 # Função Objetivo
9 Q <- function(param, data, N0){
10  ti <- data[,1]
11  Ni <- data[,2]
12  frk <- ft(param, data, N0)
13  output <- sum((Ni-frk)^2)
14  return(output)
15 }
16 # Derivadas parciais de 1 e 2 ordem da função f
17 dfdr <- function(param, data, N0){
18   t <- data$dias
19   K <- param[1]
20   r <- param[2]
21   output <- (K*N0*( (K-N0)*exp(-r*t)*t )) / ((N0 + (K-N0)*exp(-r*t))^2)
22   return(output)
23 }
24
25 dfdK <- function(param, data, N0){
26   t <- data$dias
27   K <- param[1]
28   r <- param[2]
29   output <- (N0*(-exp(-r*t)*N0+N0)) / (N0+exp(-r*t)*(K-N0))^2
30   return(output)
31 }
32
33 d2fdr2 <- function(param, data, N0){
34   t <- data$dias
35   K <- param[1]
36   r <- param[2]
37   output <- (K*exp(-2*r*t)*N0*(t^2)*(-exp(r*t)*N0-N0+K)*(K-N0)) / ((N0+exp(-r*t)*(K-N0))^3)
38   return(output)
39 }
40
41 d2fdK2 <- function(param, data, N0){
42   t <- data$dias
43   K <- param[1]
44   r <- param[2]
45   output <- (2*exp(-r*t)*N0*(-exp(-r*t)*N0+N0)) / ((N0+exp(-r*t)*(K-N0))^3)
46   return(output)
47 }
48
49 d2fdkdr <- function(param, data, N0){
50   t <- data$dias
51   K <- param[1]
52   r <- param[2]
53   output <- (exp(-r*t)*N0*N0*t*(exp(-r*t)*N0-N0+2*K-K*exp(-r*t))) / ((N0+exp(-r*t)*(K-N0))^3)
54   return(output)
55 }
56
57 # Veto gradiente da função objetivo
58 dQdr <- function(param, data, N0){
59   K <- param[1]
60   r <- param[2]
61   df <- data%%
62   mutate(aux=ft(param = param, data=data, N0 = N0),
63           dQdr=-2*besouros*dfdr(param = param, data=data, N0 = N0) +
64           2*aux*dfdr(param = param, data=data, N0 = N0))
65   output <- sum(df$dQdr)
66   return(output)
67 }
68
69 dQdK <- function(param, data, N0){
70   K <- param[1]
71   r <- param[2]

```



```

72 df <- data%>%
73   mutate(aux=ft(param = param, data=data, N0 = N0),
74           dQdk=-2*besouros*dKdK(param = param, data=data, N0 = N0) +
75           2*aux*dKdK(param = param, data=data, N0 = N0))
76 output <- sum(df$dQdk)
77 return(output)
78 }
79
80 gradQ <- function(param, data, N0){
81   K <- param[1]
82   r <- param[2]
83   gradR <- dQdr(param, data, N0)
84   gradK <- dQdK(param, data, N0)
85   gradiente <- c(gradK, gradR)
86   return(gradiente)
87 }
88
89 # Matriz Hessiana da função objetivo
90 d2Qdr2 <- function(param, data, N0){
91   K <- param[1]
92   r <- param[2]
93   df <- data%>%
94     mutate(aux=ft(param = param, data=data, N0 = N0),
95             d2Qdr2=-2*besouros*d2fdr2(param, data, N0)+2*dKdK(param, data, N0)^2 +
96             2*ft(param, data, N0)*d2fdr2(param, data, N0))
97   output <- sum(df$d2Qdr2)
98   return(output)
99 }
100
101 d2Qdk2 <- function(param, data, N0){
102   K <- param[1]
103   r <- param[2]
104   df <- data%>%
105     mutate(aux=ft(param = param, data=data, N0 = N0),
106             d2Qdk2=-2*besouros*d2fK2(param, data, N0)+2*dKdK(param, data, N0)^2 +
107             2*ft(param, data, N0)*d2fK2(param, data, N0))
108   output <- sum(df$d2Qdk2)
109   return(output)
110 }
111
112 d2QdKdr <- function(param, data, N0){
113   K <- param[1]
114   r <- param[2]
115   df <- data%>%
116     mutate(aux=ft(param = param, data=data, N0 = N0),
117             d2QdKdr=-2*besouros*d2fdKdr(param, data, N0)+
118             2*dKdK(param, data, N0)*dKdK(param, data, N0)+
119             2*ft(param, data, N0)*d2fdKdr(param, data, N0))
120   output <- sum(df$d2QdKdr)
121   return(output)
122 }
123
124 hessianQ <- function(param, data, N0){
125   A11 <- d2Qdk2(param, data, N0)
126   A22 <- d2Qdr2(param, data, N0)
127   A12 <- A21 <- d2QdKdr(param, data, N0)
128   hessiana <- matrix(data = c(A11, A21, A12, A22), 2, 2)
129   return(hessiana)
130 }

```

A.2 Newton-Raphson

```

1 NR <- function(theta_k, data, N0, epsilon, criterium){
2   iter=0
3   while (criterium>epsilon) {
4     iter <- iter + 1
5     theta_k_plus_1 <- theta_k - solve(hessianQ(param=theta_k, data=data, N0=2))%*%gradQ(param=
6       theta_k, data=data, N0=2)
7     criterium <- abs(Q(param = theta_k_plus_1, data = data, N0=2)/Q(param = theta_k, data = data,
8       N0=2) - 1)
9     theta_k <- theta_k_plus_1
10  }
11  output <- list(iter=iter, theta=theta_k, Q_func=Q(param=theta_k, data, N0))
12  return(output)
13 }

```

A.3 line search

```

1 # Algoritmo de line search
2 ft_ls <- function(gama, param, data, N0){
3   t <- data$dias
4   k <- param[1]
5   r <- param[2]
6   p <- solve(hessianQ(param=param, data=data, N0=2))%*%gradQ(param=param, data=data, N0=2)
7   pk <- p[1]; pr=p[2]
8   output <- ((k-gama*pk)*N0) / (N0 + (k-gama*pk-N0)*exp(-(r-gama*pr)*t))
9   return(output)
10 }
11 Q_ls <- function(gama, param, data, N0){
12   ti <- data$dias
13   Ni <- data$besouros
14   frk <- ft_ls(gama, param, data, N0)
15   output <- sum((Ni-frk)^2)
16   if(gama>.5){
17     return(output)
18   }else{
19     return(Inf)
20   }
21 }
22 ls <- function(theta_k, data, N0, epsilon, criterium){
23   iter=0
24   while (criterium>epsilon) {
25     iter <- iter + 1
26     gama <- optimize(f = Q_ls, interval = c(.5, 2), param=theta_k, data=data, N0=N0)$minimum
27     theta_k_plus_1 <- theta_k - gama*solve(hessianQ(param=theta_k, data=data, N0=2))%*%gradQ(
28       param=theta_k, data=data, N0=2)
29     criterium <- abs(Q(param = theta_k_plus_1, data = data, N0=2)/Q(param = theta_k, data = data,
30       N0=2) - 1)
31     theta_k <- theta_k_plus_1
32   }
33   output <- list(iter=iter, theta=theta_k, Q_func=Q(param=theta_k, data, N0))
34   return(output)
35 }

```

A.4 Escore de fisher

```

1 llk <- function(param, data, N0){
2   K <- param[1]
3   r <- param[2]
4   sigma2 <- param[3]
5   ti <- data[,1]
6   Ni <- data[,2]
7   frk <- ft(param, data, N0)
8   output <- sum(-.5*log(2*pi*sigma2) - (1/(2*sigma2))*(log(Ni)-log(frk))^2 )
9   if(sigma2>0 & r>0 & K>0){
10     return(-output)
11   }else{
12     return(Inf)
13   }
14 }
15 }

```

```

16 dldK <- function(param, data, N0){
17   K <- param[1]
18   r <- param[2]
19   sigma2 <- param[3]
20   df <- data%%
21   mutate(aux=ft(param = param, data=data, N0 = N0),
22           dfdK=dfdK(param, data, N0),
23           dldK=(-1/(2*sigma2)) * (-2*log(besouros)*(1/aux)*dfdK + 2*log(aux)*(1/aux)*dfdK ))
24   output <- sum(df$dldK)
25   return(output)
26 }
27
28 dldr <- function(param, data, N0){
29   K <- param[1]
30   r <- param[2]
31   sigma2 <- param[3]
32   df <- data%%
33   mutate(aux=ft(param = param, data=data, N0 = N0),
34           dfdr=dfdr(param, data, N0),
35           dldr=(-1/(2*sigma2)) * (-2*log(besouros)*(1/aux)*dfdr + 2*log(aux)*(1/aux)*dfdr ))
36   output <- sum(df$dldr)
37   return(output)
38 }
39
40 dlsigma2 <- function(param, data, N0){
41   K <- param[1]
42   r <- param[2]
43   sigma2 <- param[3]
44   ti <- data[,1]
45   Ni <- data[,2]
46   frk <- ft(param, data, N0)
47   output <- sum(-1/(2*sigma2)) + (1/(2*sigma2^2))*(log(Ni)-log(frk))^2 )
48   return(output)
49 }
50
51 gradllk <- function(param, data, N0){
52   K <- param[1]
53   r <- param[2]
54   sigma2 <- param[3]
55   gradR <- dldr(param, data, N0)
56   gradK <- dldK(param, data, N0)
57   gradsigma2 <- dlsigma2(param, data, N0)
58   gradiente <- c(gradK, gradR, gradsigma2)
59   return(gradiente)
60 }
61
62 Ed2ldr2 <- function(param, data, N0){
63   K <- param[1]
64   r <- param[2]
65   sigma2 <- param[3]
66   df <- data%%
67   mutate(aux=ft(param = param, data=data, N0 = N0),
68           dfdr=dfdr(param, data, N0),
69           d2fdr2=d2fdr2(param, data, N0),
70           d2ldr2=(-1/(2*sigma2)) * (-2*log(aux)*((1/aux)*d2fdr2-(aux^(-2))*(dfdr^2)) +
71           2*(((aux^(-2))*dfdr-(aux^(-2))*dfdr*log(aux))*dfdr + log(aux)*(1/aux)*d2fdr2
72           )) )
73   output <- sum(df$d2ldr2)
74   return(output)
75 }
76
77 Ed2ldK2 <- function(param, data, N0){
78   K <- param[1]
79   r <- param[2]
80   sigma2 <- param[3]
81   df <- data%%
82   mutate(aux=ft(param = param, data=data, N0 = N0),
83           dfdK=dfdK(param, data, N0),
84           d2fdK2=d2fdK2(param, data, N0),
85           d2ldK2=(-1/(2*sigma2)) * (-2*log(aux)*((1/aux)*d2fdK2-(aux^(-2))*(dfdK^2)) +
86           2*(((aux^(-2))*dfdK-(aux^(-2))*dfdK*log(aux))*dfdK + log
87           (aux)*(1/aux)*d2fdK2) ))
88   output <- sum(df$d2ldK2)
89   return(output)
90 }

```

```

90 Ed2ldsigma22 <- function(param, data, N0){
91   K <- param[1]
92   r <- param[2]
93   sigma2 <- param[3]
94   ti <- data[,1]
95   Ni <- data[,2]
96   frk <- ft(param, data, N0)
97   output <- sum((1/(2*sigma2^2)) - (1/(sigma2^3))*(sigma2))
98   return(output)
99 }
100
101 Ed2ldKdr <- function(param, data, N0){
102   K <- param[1]
103   r <- param[2]
104   sigma2 <- param[3]
105   df <- data%>%
106     mutate(aux=ft(param = param, data=data, N0 = N0),
107             dfdK=dfdK(param, data, N0),
108             dfdr=dfdr(param, data, N0),
109             d2fdKdr=d2fdKdr(param, data, N0),
110             d2ldKdr=(-1/(2*sigma2)) * (-2*log(aux)*((1/aux)*d2fdKdr-(aux^(-2))*(dfdK*dfdr)) +
111               2*((aux^(-2))*dfdK-(aux^(-2))*dfdK*log(aux))*dfdK + log
112               (aux)*(1/aux)*d2fdKdr)))
113   output <- sum(df$d2ldKdr)
114   return(output)
115 }
116 Ed2ldrdsigma2 <- function(param, data, N0){
117   K <- param[1]
118   r <- param[2]
119   sigma2 <- param[3]
120   df <- data%>%
121     mutate(aux=ft(param = param, data=data, N0 = N0),
122             dfdr=dfdr(param, data, N0),
123             d2ldrdsigma2=(1/(2*sigma2^2)) * (-2*log(aux)*(1/aux)*dfdr + 2*log(aux)*(1/aux)*dfdr
124               ))
125   output <- sum(df$d2ldrdsigma2)
126   return(output)
127 }
128 Ed2ldKdsigma2 <- function(param, data, N0){
129   K <- param[1]
130   r <- param[2]
131   sigma2 <- param[3]
132   df <- data%>%
133     mutate(aux=ft(param = param, data=data, N0 = N0),
134             dfdk=dfdr(param, data, N0),
135             d2ldkdsigma2=(1/(2*sigma2^2)) * (-2*log(aux)*(1/aux)*dfdk + 2*log(aux)*(1/aux)*dfdk
136               ))
137   output <- sum(df$d2ldkdsigma2)
138   return(output)
139 }
140 fisher_inf <- function(param, data, N0){
141   A11 <- Ed2ldK2(param, data, N0)
142   A22 <- Ed2ldr2(param, data, N0)
143   A33 <- Ed2ldsigma22(param, data, N0)
144   A12 <- A21 <- Ed2ldKdr(param, data, N0)
145   A13 <- A31 <- Ed2ldKdsigma2(param, data, N0)
146   A23 <- A32 <- Ed2ldrdsigma2(param, data, N0)
147   FI <- matrix(c(A11, A21, A31, A12, A22, A32, A13, A23, A33), 3, 3)
148   return(FI)
149 }
150
151 theta_k=c(1000,.1,1)
152 data=dados[-1,]
153 N0=2
154 EF <- function(theta_k, data, N0, epsilon, criterium){
155   iter=0
156   while (criterium>epsilon) {
157     iter <- iter + 1
158     theta_k_plus_1 <- theta_k + solve(-fisher_inf(param=theta_k, data=data, N0=2))%*%gradllk(
159       param=theta_k, data=data, N0=2)
160     criterium <- abs(llk(param = theta_k_plus_1, data = data, N0=2)/llk(param = theta_k, data =
161       data, N0=2) - 1)
162     theta_k <- theta_k_plus_1

```

```

161 }
162 output <- list(iter=iter, theta=theta_k, llk=llk(param=theta_k, data, N0))
163 return(output)
164 }

```

A.4 Critério de parada

O critério de parada adotado para as otimizações foi

$$|Q(K_{k+1}, r_{k+1})/Q(K_k, r_k) - 1| < \varepsilon = 10^{-5} \quad (9)$$

Apêndice B