3
ESTIMAÇÃO DE PARÂMETROS DO MRSTAR

3.1
Introdução

Como já foi explicado no capítulo 2, o ciclo de estimação do STAR-Tree consiste em um processo iterativo de estimação de uma conFiguração da árvore, avaliação deste resultado, no qual irá se basear a decisão de continuar crescendo a árvore numa determinada direção ou não. Neste capítulo, o foco será na etapa de estimação dos parâmetros de uma dada conFiguração de árvore. Será dado foco em modelos MRSTAR onde a variável de transição é auto-regressiva de primeira, e a função de transição é a logística.

O vetor de parâmetros de um modelo MRSTAR pode ser subdividido em um conjunto de variáveis lineares e não-lineares. As não-lineares são aquelas que provêm dos nós terminais, que contém os modelos de previsão AR de ordem p. Os nós de decisão, que contém as funções de transição, possuem dois parâmetros não-lineares cada, no caso da função logística. Em um MRSTAR(n , p) com n regimes, existem n modelos AR de ordem p e (n-1) nós de decisão. Portanto, um modelo desta ordem possui um vetor de parâmetros com np parâmetros lineares e 2n-2 parâmetros não lineares, totalizando n(p+2)–2 parâmetros. A equação (3-1) descreve o vetor de parâmetros:

$$\Phi = \left[\theta \quad \phi\right], \text{ onde } \theta = \left[B_0 \cdots \quad B_n\right] \text{ e } \phi = \left[c_0\ \gamma_0 \quad \cdots \quad c_{n-1} \quad \gamma_{n-1}\right] \tag{3-1}$$

$$B_i = \begin{bmatrix} \beta_{i,0} \\ \beta_{i,1} \\ \vdots \\ \beta_{i,p} \end{bmatrix}, \ \theta \text{ é o vetor de parâmetros lineares e } \phi \text{ o de não-lineares}$$

A estimação do vetor de parâmetros é feita através da maximização da função de verossimilhança do modelo MRSTAR(n,p). Assumindo que os erros

são normais, independentes e identicamente distribuídos, a função de verossimilhança pode ser descrita pela equação (3-2):

$$L(Y \mid \Phi) = \prod_{i=1}^{T} f(y_{i-1} \mid \Phi) = \left(\frac{1}{2\pi\sigma^2}\right)^{T/2} \cdot \exp\left(-\frac{\sum_{i=1}^{T}(y_i - F(y_{i-1};\Phi))^2}{2\sigma^2}\right) \quad (3\text{-}2)$$

$$F(X) = \sum_{i=1}^{K}\left[\beta_{i,0} \quad \beta_{i,1} \quad \cdots \quad \beta_{i,1}\right] \cdot \begin{bmatrix} 1 \\ y_{t-1} \\ \vdots \\ y_{t-p} \end{bmatrix} \cdot H_i(y_{t-1};\Phi_i)$$

Onde H_i é o produtório das funções de pertinência dos nós de decisão superiores ao nó terminal em que um determinado modelo AR se encontra.

Maximizar a função acima não é uma tarefa trivial, portanto uma transformação logarítmica é aplicada a $L(Y \mid \Phi)$. Como o logaritmo é uma função contínua crescente ao ser aplicada na região da verossimilhança, os valores que a maximizam fazem o mesmo à função original. A vantagem é que a álgebra necessária para maximizar o logaritmo é muito mais simples. Portanto, temos:

$$l(Y \mid \Phi) = \ln(L(Y \mid \Phi)) = \frac{T}{2}\ln 2\pi - T\ln\sigma - \frac{\sum_{i=1}^{T}(y_i - F(y_{i-1};\Phi))^2}{2\sigma^2} \quad (3\text{-}3)$$

O vetor de parâmetros que maximiza a função acima é chamado de estimador de máxima verossimilhança.

3.2
Dificuldades na estimação

A estimação deste vetor de parâmetros é complicada em determinadas situações. Quando o parâmetro γ, fator de escala da função logística, é muito baixo, a função resultante é bastante suave. Como conseqüência, é esperado que métodos tradicionais de otimização tenham dificuldade em estimar corretamente os parâmetros nessa situação. Este fato já foi explorado em [8] pode ser entendido de duas formas. Uma é que quando o parâmetro de suavidade função logística é extremamente baixo, praticamente não há diferenciação entre os regimes, e o modelo resultante se comporta praticamente como sendo linear. Isto quer dizer que existirão inúmeras combinações de modelos aplicados aos nós terminais da árvore que poderão gerar resultados bastante similares, o que dificulta a tarefa de estimação.

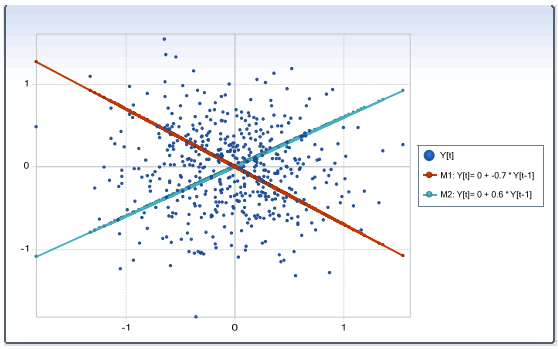


Figura 3.1: LSTAR(1) com γ=0 e c=0

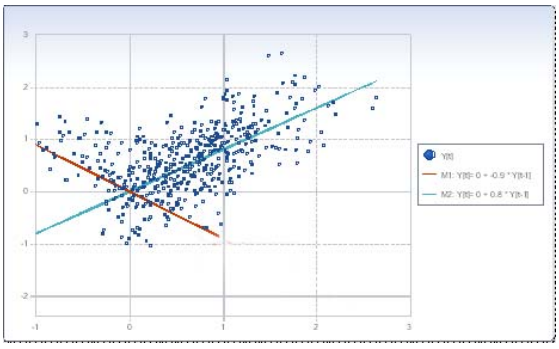


Figura 3.2: LSTAR(1) com γ=10 e c=0

As Figuras 3.1 e 3.2 representam duas séries geradas com parametrizações diferentes de γ. O eixo X representa a variável de transição (y_{t-1}), enquanto que o Y representa y_t. Ambas as séries foram geradas com 500 pontos. É possível ver que no segundo caso, onde o γ é igual a 10, os modelos são facilmente identificáveis, enquanto que no primeiro caso, onde o γ é igual a zero, não há distinção visível entre os modelos. Neste caso, o modelo LSTAR(1) funciona na prática como um único modelo AR(1), e diferentes combinações de modelos intermediários que resultam neste mesmo modelo tornam o trabalho de estimação mais complicado.

Outro fator importante é que estimar os parâmetros não lineares quando um deles, o γ, tende a zero, é uma tarefa complicada em termos de otimização. As variações na função de verossimilhança são muito suaves quando γ é baixo, o que muitas vezes inviabiliza a convergência destes métodos para o seu máximo.

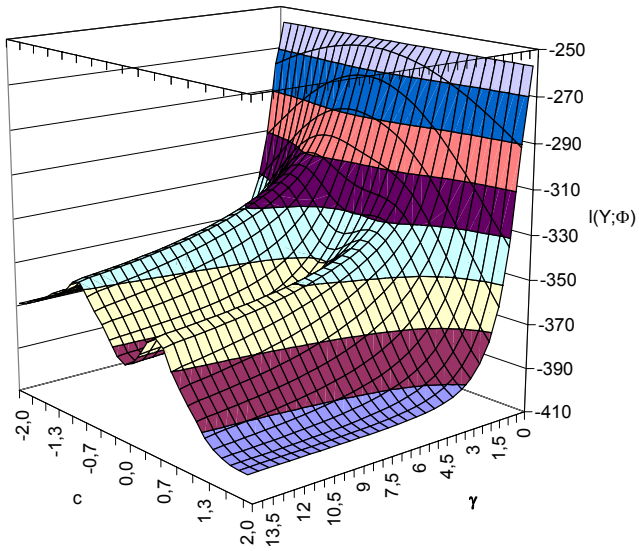


Figura 3.3: Superfície da função $l(Y \mid \Phi)$ para diferentes valores de c e γ. O modelo gerado foi um LSTAR(1) com $\gamma = 0, c = 0, B_1 = [0 \quad -.8], B_2 = [0 \quad .6]$, gerado com 1000 pontos

O exemplo mostrado na Figura 3.3 é bem claro neste aspecto. Quando γ tende a zero, não há mudança significativa da função $l(Y \mid \Phi)$ para diferentes valores de c.

Já quando o γ é muito alto, a estimação do parâmetro c se torna mais fácil. Porém, a determinação dele próprio se torna bem mais complicada. Isto acontece porque conforme γ cresce, a transição entre regimes vai se tornando abrupta, ao ponto que a pertinência da variável de transição a um regime ou a outro se torna praticamente exclusiva. A partir de um certo ponto, esta situação não se altera significativamente a cada novo incremento em γ. O que teremos na prática como resultado provável é um c bem estimado em conjunto com um γ alto, porém dificilmente estimado com precisão. A Figura 3.4 ilustra esta situação.

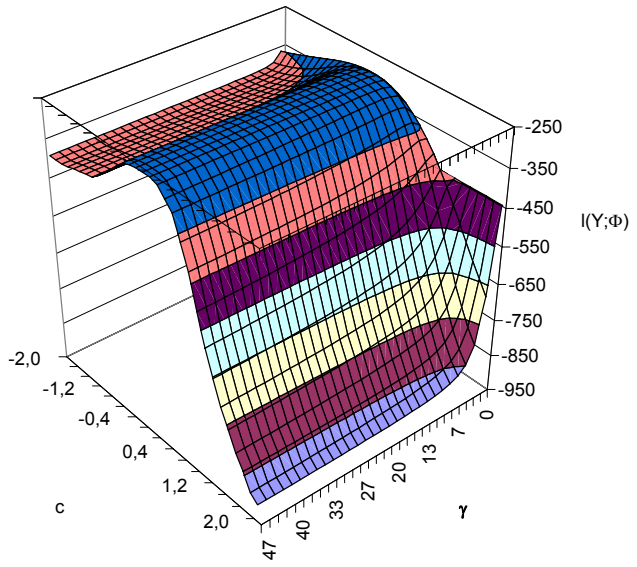


Figura 3.4: Superfície da função $l(Y \mid \Phi)$ para diferentes valores de c e γ. O modelo gerado foi um LSTAR(1) com $\gamma = 50, c = 0, B_1 = [0 \quad -.8], B_2 = [0 \quad .6]$, gerado com 1000 pontos

Nas seções seguintes serão apresentadas diferentes abordagens que foram implementadas para realizar esta tarefa de estimação de parâmetros. Nas seções 3.3, 3.4 e 3.5, serão introduzidos os métodos de estimação dos parâmetros não lineares. Na seção 3.6, será definida a estratégia de estimação de parâmetros lineares e não lineares. Na seção 3.7 será apresentada uma abordagem alternativa de estimação de todos os parâmetros através de algoritmos genéticos. Ao fim do capítulo, será feita uma comparação de desempenho entre estes métodos.

3.3
Método de Gradiente

Métodos de otimização são utilizados para encontrar o vetor X* que maximizam ou minimizam (dependendo do objetivo do problema) a saída de uma função $f(X) \in R$, X* sendo um vetor de ordem p, $p = 1,2,3....,n$, $X \in R^p$.

O método do Gradiente é relativamente simples. Se a função $f(X) \in R, X \in R^p$, é definida e diferenciável nas adjacências de um ponto Z qualquer, então $f(X)$ decresce mais rápido se caminhar-se deste ponto Z na direção negativa do gradiente de $f(Z)$. O gradiente de $f(Z)$ define-se por:

$$\nabla f(Z) = \begin{bmatrix} \frac{df}{dz_1} \\ \frac{df}{dz_2} \\ \vdots \\ \frac{df}{dz_p} \end{bmatrix} \tag{3-4}$$

A convergência para um vetor X* que minimiza a função f é obtida neste método através de um processo iterativo. Quando o gradiente da função não caminhar significativamente em nenhuma direção, é hora de abortar o algoritmo. A equação a seguir descreve o algoritmo:

$$X_{k+1} = X_k - \alpha \nabla f(X_k), k \geq 0 \tag{3-5}$$

X_k é o valor de X estimado na iteração k. Enquanto $X_{k+1} < X_k$, ainda há ganho e continua-se a iterar no algoritmo. Na prática, estipula-se uma tolerância , e o algoritmo não pára enquanto $\frac{|X_{k+1} - X_k|}{X_k} > \delta$. Isto se faz para evitar gastos computacionais desnecessários quando a solução começa a convergir muito lentamente.

O parâmetro α em (3-2) é chamado de passo. Seu uso é justificado porque o gradiente traz informação suficiente sobre a direção a ser percorrida para se atingir a minimização, porém não há como saber com que intensidade caminhar naquela direção. Se o passo for muito largo, é possível que nunca se atinja a convergência, pois facilmente o mínimo seria ultrapassado entre uma iteração e outra. Se for muito baixo, irá se atingir convergência, porém a um custo computacional muito alto, já que irá se caminhar muito devagar. O valor de α pode mudar de valor a cada iteração.

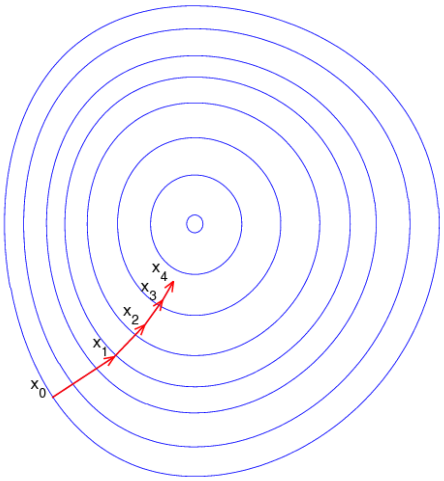


Figura 3.5: Ilustração do método do Gradiente

Na prática, a determinação do passo é uma tarefa complicada. Se o fixarmos, os resultados podem não ser satisfatório. Se o estimarmos a cada passo, o gasto computacional pode não justificar tal tarefa. O método do gradiente, porém, é bastante simples de se implementar, pois só envolve derivadas de primeira ordem.

A escolha do vetor inicial é determinante na eficácia do algoritmo. O algoritmo garante convergência para mínimos locais, não necessariamente globais das funções a serem otimizadas. Isto quer dizer que se um vetor Z qualquer for escolhido como X0, e Z for próximo a um mínimo que não seja global, o algoritmo irá convergir para ele, e não irá retornar o melhor valor possível.

3.4
Método de Newton

O método de Newton utiliza a segunda derivada como informação adicional na determinação de X*. Ele baseia-se na expansão de Taylor de ordem dois, aonde:

$$f(x+\alpha x) = f(x) + f'(x)\alpha x + \frac{1}{2} f''(x)\alpha x^2 + e \qquad (3\text{-}6)$$

A função acima é minimizada quando aplicarmos a condição de otimalidade necessária para a derivada de 1ª ordem de (3-6), ou seja:

$$f'(x)\alpha + f''(x)\alpha x = 0 \tag{3-7}$$

E a condição de otimalidade suficiente para a derivada de segunda ordem:

$$f''(x) > 0$$

O método é análogo ao do gradiente, e pode ser descrito pela equação:

$$X_{k+1} = X_k - \alpha\left[Hf(X_k)\right]^{-1}\nabla f(X_k) \tag{3-8}$$

$$k \geq 0\,,\, Hf(X_k) = \begin{bmatrix} \dfrac{df}{(dx_{1,k})^2} & \dfrac{df}{dx_{1,k}dx_{2,k}} & \cdots & \dfrac{df}{dx_{1,k}dx_{p,k}} \\ \dfrac{df}{dx_{2,k}dx_{1,k}} & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \dfrac{df}{dx_{p,k}dx_{1,k}} & \cdots & \cdots & \dfrac{df}{dx_{p,k}dx_{p,k}} \end{bmatrix}$$

A matriz $Hf(X_k)$ é de ordem p x p, e é chamada de Hessiano da função f, e contém as segundas derivadas de f em relação às variáveis xi do vetor X. Isto quer dizer que a utilização do método pressupõe que f é convexa. O Hessiano deve ser inversível e positivo definido, ou seja:

$$Y^T Hf(X)Y \geq 0, \forall Y \tag{3-9}$$

Uma vez que todas estas condições sejam satisfeitas, o método de Newton é superior e converge mais rápido que o método do gradiente [9]. O uso da matriz Hessiana na otimização do vetor X incorpora a informação de curvatura da função f, permitindo que seja tomada uma rota mais direta a seu valor ótimo. A Figura 3.6 ilustra isto melhor.

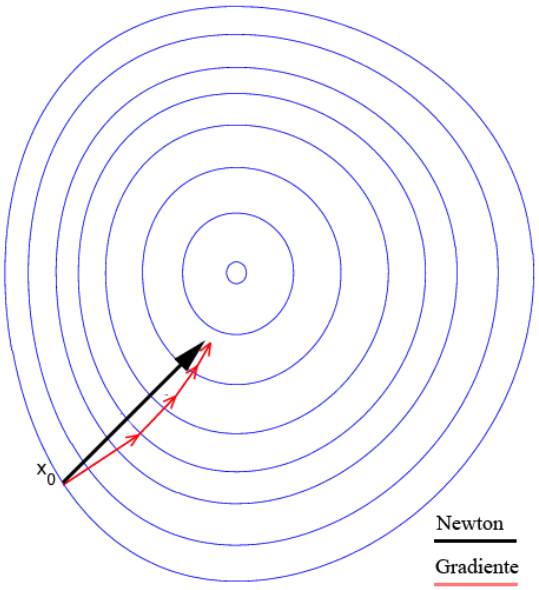


Figura 3.6: Ilustração comparando o método de Newton com o do Gradiente

O problema é que na prática, é bastante comum encontrar dificuldades na inversão do hessiano. Matrizes que não são inversíveis ou positivas-definidas irão fazer com que o método divirja da solução. Além disto, o cálculo do hessiano é uma operação custosa.

Como no método do Gradiente, a determinação do passo pode ser realizada passo a passo. Neste caso, um passo inicial é pré-determinado, e enquanto $\frac{|X_{k+1} - X_k|}{X_k} <= \delta$, o passo é reduzido à metade e X_{k+n} é recalculado.

3.5
BFGS

O Método BFGS ([3, 4, 5]) – a sigla provém dos nomes de seus criadores Broyden, Fletcher, Goldfarb e Shanno – é um método feito para resolver problemas de otimização não-linear sem restrições. Ele é derivado do método de Newton e se encaixa em uma clase de modelos que são chamados de métodos Quasi-Newton.

O método de Newton assume que a função estudada pode ser aproximada localmente como quadrática na região em torno de seu ótimo, e se utiliza de primeiras e segundas derivadas para convergir para um ponto de estacionariedade.

Nos métodos Quasi-Newton, não há necessidade de calcular a matriz Hessiana em momento algum. A informação que é trazida pelo Hessiano no método de Newton é substituída nos métodos Quasi-Newton pela análise de gradientes consecutivos. Os métodos desta classe são generalizações do método da secante para encontrar a raiz da primeira derivada em problemas multidimensionais.

Para ilustrar o caso unidimensional, o método da secante requer dois valores iniciais, que devem ser idealmente escolhidos próximos à raiz. A relação de recorrência se dá pela equação:

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n) \tag{3-10}$$

O método convergirá para a raiz, caso os valores iniciais x_0 e x_1 estejam suficientemente próximos a ela. Caso isto não aconteça, não há como garantir convergência. Se comparado com o método de Newton, o método da secante leva uma quantidade maior de iterações para convergir. Porém, enquanto o método de Newton precisa avaliar a função, sua derivada a cada passo, o da secante só avalia a própria função em cada iteração. Isto o torna na prática, mais rápido por ser menos custoso computacionalmente. Isto se torna especialmente verdadeiro no caso multidimensional. Voltando aos métodos Quasi-Newton, que são generalizações do método da secante para encontrar a raiz da primeira derivada da função estudada, o fato de não haver necessidade de calcular o Hessiano da função nem invertê-lo poupa esforço computacional. Este fator cresce de importância conforme a dimensão do problema aumenta.

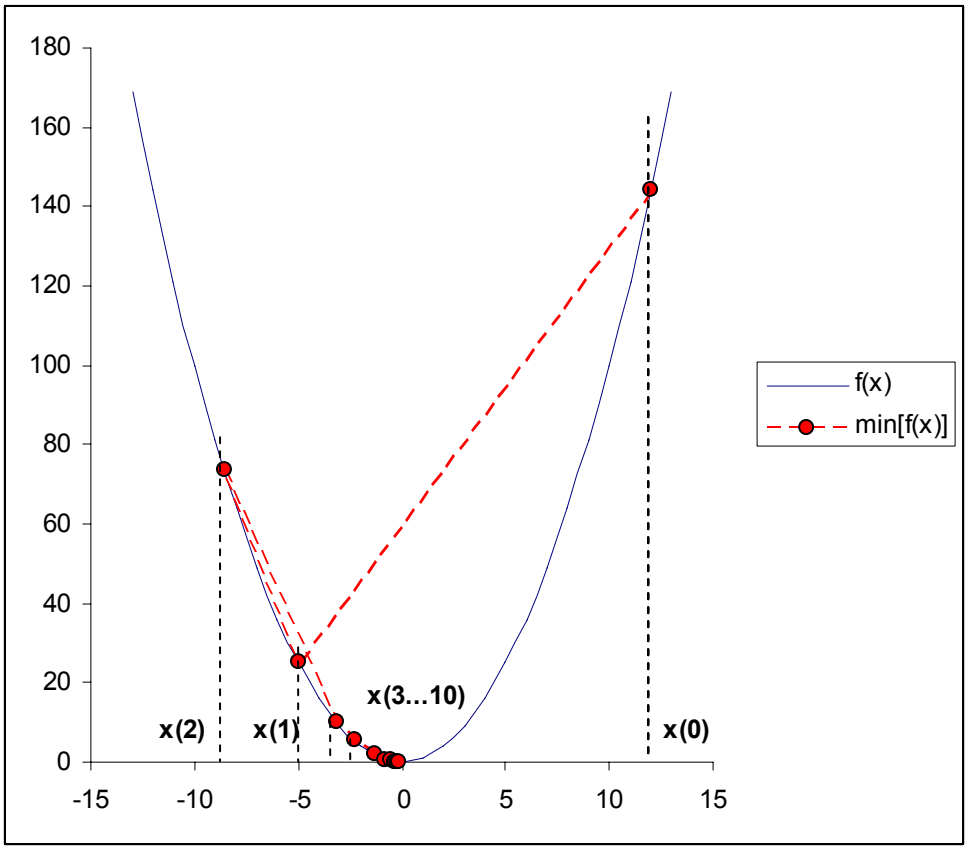


Figura 3.7: Ilustração do método da secante convergindo para o mínimo da função, partindo de dois pontos inicialmente selecionados

O primeiro método Quasi-Newton proposto chamava-se DFP (David-Fletcher-Powel), mas é raramente utilizado hoje em dia. Os algoritmos mais utilizados atualmente são o SR1 e o BFGS, proposto independentemente por Broyden, Fletcher, Goldfarb e Shanno, em 1970.

Se recordarmos a expansão de Taylor na equação (3-6) do método de Newton, e fizermos novamente uma expansão no gradiente da função, teremos:

$$f'(x_0 + \Delta x) = f'(x_0) + D\Delta x \qquad (3\text{-}11)$$

A equação acima é chamada equação da secante. Igualando $f'(x_0 + \Delta x)$ a 0 , temos $\Delta x = -D^{-1} \cdot f'(x_0)$. Ao invés de calcularmos D como o hessiano da função f, iremos calcular aproximações da mesma passo a passo com valores consecutivos de x. Novamente, existe o problema da escolha de valores iniciais. Escolher D_0=I (matriz identidade) é normalmente suficiente para alcançar rapidamente a convergência. A cada atualização de x_k pela equação principal

equivalente à de Newton, ganhamos informação para calcular o próximo valor de D_k. É neste momento que os diversos métodos quasi-Newton se diferenciam, pois são diferentes propostas para as fórmulas de atualização. O quadro a seguir exibe algumas delas:

Método	**D_{k+1}=**
DFP	$\left(I-\frac{y_k \Delta x_k^T}{y_k^T \Delta x_k}\right) D_k \left(I-\frac{\Delta x_k^T y_k^T}{y_k^T \Delta x_k}\right)+\frac{y_k y_k^T}{y_k^T \Delta x_k}$
BFGS	$D_k+\frac{y_k y_k^T}{y_k^T \Delta x_k}-\left(\frac{D_k \Delta x_k \left(D_k \Delta x_k\right)^T}{\Delta x_k^T D_k \Delta x_k}\right)$
Broyden	$D_k+\left(\frac{y_k - D_k \Delta x_k}{\Delta x_k^T \Delta x_k} \Delta x_k^T\right)$
SR1	$D_k+\left(\frac{y_k - D_k \Delta x_k \left(y_k - D_k \Delta x_k\right)^T}{\left(y_k - D_k \Delta x_k\right)^T \Delta x_k}\right)$

Figura 3.8: Diferentes métodos estimando o próximo passo da matriz D

O algoritmo pode ser descrito da seguinte forma:

Inicialização:

X_0

B_0=I

η (tolerância)

Enquanto $f(x_{k+1}) > f(x_k)$ e $\frac{|f(x_{k+1}) - f(x_k)|}{f(x_k)} > \eta$

{

$h = -D_k$

$\alpha_k = \arg\min\{f(x_k + \alpha_k \times h)\}$

$x_{k+1} = x_k + \alpha_k \times D_k \times \nabla f(x_k)$

$z_k = \nabla f(x_{k+1}) - \nabla f(x_k)$

$D_{k+1} = D_k \frac{z_k z_k^T}{z_k^T \Delta x_k} - \frac{D_k \Delta x_k (D_k \Delta x_k)^T}{\Delta x_k^T D_k \Delta x_k}$

}

3.6
Aplicação de Métodos de Otimização na Estimação do MRSTAR

Conforme visto na seção 3.1, os vetores de parâmetros de um modelo MRSTAR podem ser subdivididos em parâmetros lineares e não-lineares. Podemos simplificar o uso de métodos de otimização ao também subdividir o esforço de estimação em dois. Se os parâmetros não-lineares fossem conhecidos, a equação (3-3) poderia ser reescrita como:

$$l(Y;\theta \mid \phi) = \frac{T}{2}\ln 2\pi - T\ln\sigma - \frac{\sum_{i=1}^{T}\left(y_i - F\left(y_{i-1};\theta \mid \phi\right)\right)^2}{2\sigma^2} \tag{3-12}$$

Em (3-12) supomos que os erros do modelo são normais e I.I.D.. Quando se conhecem os parâmetros não lineares, maximizar a função de verossimilhança para os parâmetros lineares equivale a encontrá-los por meio do método de mínimos quadrados ordinários. Desta forma, encontra-se através de:

$$\theta \mid \phi = \left(W(\phi)^T W(\phi)\right)^{-1} W(\phi)^T Y \tag{3-13}$$

$$W = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{bmatrix},$$

$$Z_i = \begin{bmatrix} 1\cdot H_i\left(y_{t-(n-p)}\right) & \cdots & 1\cdot H_i\left(y_{t-2}\right) & 1\cdot H_i\left(y_{t-1}\right) \\ y_{t-(n-p)}\cdot H_i\left(y_{t-(n-p)}\right) & \cdots & y_{t-2}\cdot H_i\left(y_{t-2}\right) & y_{t-1}\cdot H_i\left(y_{t-1}\right) \\ \vdots & \vdots & \vdots & \vdots \\ y_{t-n}\cdot H_i\left(y_{t-(n-p)}\right) & \cdots & y_{t-(p+1)}\cdot H_i\left(y_{t-1}\right) & y_{t-p}\cdot H_i\left(y_{t-1}\right) \end{bmatrix}$$

$$H_i = \prod_{j=1}^{k} f(c_j;\gamma_j;y_{t-1})$$

De forma equivalente, uma vez conhecidos os parâmetros lineares, pode-se estimar condicionalmente o melhor conjunto de parâmetros não lineares que maximizam a função de verossimilhança. Para isso, qualquer um dos métodos descritos nas seções anteriores pode ser utilizado.

Este processo iterativo irá convergir quando não houver mais ganho significativo, definido por uma tolerância, que é um parâmetro de entrada do algoritmo. Ainda, para mitigar os problemas relacionados a mínimos locais na estimação de parâmetros não-lineares, o primeiro passo do algoritmo é dado a partir de uma grade de valores de γ_i e c_j. Para construir a grade, restringe-se os γ_i a valores entre um e cem, enquanto que os valores de c devem estar entre o mínimo e máximo da série Y. Quanto ao γ, não é vantajoso escolher valores baixos nem altos demais, por motivos já explicados anteriormente. No caso de c, a escolha de valores fora da amplitude total observada irá invariavelmente gerar exclusividade na pertinência de determinados modelos, o que não é interessante.

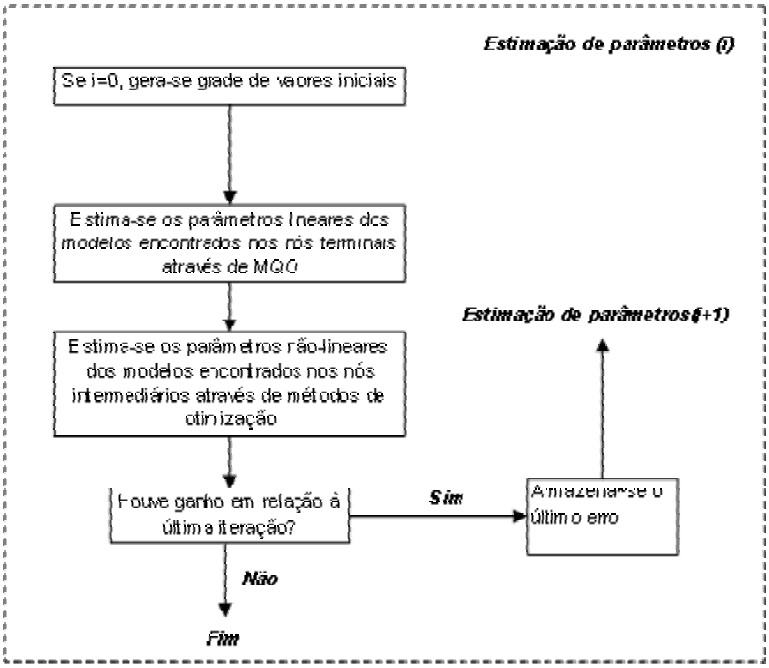


Figura 3.9: Algoritmo de estimação de parâmetros por métodos de otimização

3.7
Estimação por Algoritmos Genéticos

Esta abordagem é completamente diferente dos métodos descritos anteriormente. Ela tem inspiração no princípio de seleção natural de Darwin, e são baseados no conceito de evolução das espécies.

Os Algoritmos Genéticos foram introduzidos por John Holland [10] e seus colaboradores em meados da década de 70. São algoritmos de busca baseados nos mecanismos de seleção natural e genética. Os mesmos combinam a idéia evolucionária de sobrevivência dos mais aptos com uma troca de informações aleatória, simulando processos naturais, tais como: seleção, reprodução, hereditariedade, mutação e dinâmica das populações.

Seu funcionamento é baseado no fato de que os indivíduos que possuem melhores características genéticas terão maiores chances de se reproduzir, gerando indivíduos mais aptos a cada nova geração.

Essencialmente, os AGs tentam minimizar ou maximizar um valor, conforme o objetivo do problema. Inicialmente é gerada uma população aleatória de indivíduos, que podem ser vistos como possíveis soluções do problema proposto. Durante o processo evolutivo, esta população é avaliada: para cada indivíduo é dado um índice, ou nota, refletindo sua habilidade de adaptação a um determinado ambiente. A cada geração, é observado um comportamento evolutivo no algoritmo através de duas características básicas: competição e cooperação, onde os princípios de seleção e reprodução são aplicados.

Segue abaixo um pseudocódigo do funcionamento de um algoritmo genético:

```
t = 0              ; primeira geração.
inicializa P(t)    ; população inicial aleatória.

avalia P(t)        ; calcula f(i) p/ cada indivíduo.
Enquanto  não(condição_parada)
        t = t + 1                  ; próxima geração.

        seleciona P(t) de P(t-1)
        altera P(t)        ; crossover e mutação.
        avalia P(t)        ; calcula f(i) p/ cada indivíduo
```

Neste tipo de algoritmo, cada possível solução é um indivíduo em uma população de soluções. Este indivíduo pode ser descrito por uma seqüência genética que está sujeita a operações de reprodução e mutação ao longo da

evolução da população. A aptidão de um indivíduo perante o grupo é medida através de uma função de avaliação. Na estimação de séries temporais, é conveniente utilizar o somatório dos erros quadráticos do modelo como função de avaliação, pois é uma medida proporcional ao log da função de verossimilhança. Quanto menor a soma dos erros quadráticos, portanto, mais apto é o indivíduo.

Quando uma população evolui, seus melhores indivíduos (aqueles com maior aptidão), que representam uma parcela pequena do grupo, migram para a geração seguinte. A população da geração seguinte deve possuir o mesmo número de indivíduos da geração anterior. Por isso, o restante dos indivíduos é gerado ou por recombinação genética ou aleatoriamente. Um percentual pequeno da população é exposto a uma mutação genética, o que traz diversidade ao grupo, principalmente em gerações muito avançadas, quando há homogeneidade na população.

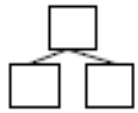
Nesta implementação, optou-se por trabalhar com seqüências binárias dos genes. Isto quer dizer que um dado vetor de parâmetros, todos números reais, deve ser traduzido para base binária. Cada número real com p casas decimais de precisão ser traduzido por uma seqüência binária de k bits, que pode ser determinado pela inequação:

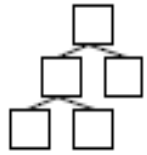
$$2^k \leq (x_{\max} - x_{\min})10^p \tag{3-11}$$

Um vetor de números reais, portanto, é traduzido para binário através da concatenação das seqüências de bits que compõem seus itens. Nesta aplicação para a estimação do modelo STAR, o vetor de números reais fornecido ao modelo continha todos os parâmetros a serem estimados, lineares e não lineares. A população gerada possuia diversos indivíduos com diferentes configurações de parametrização. A função de avaliação utilizada foi a soma dos erros quadráticos produzida por cada um dos indivíduos. Um percentual dos mais aptos evoluia para a geração seguinte, e diferentes percentuais de mutação e cruzamento dos indivíduos foram introduzidos. Estes percentuais eram passíveis de alteração a cada geração.

3.8
Resultados

Foram realizadas simulações de Monte Carlo com o fim de comparar a eficácia dos diferentes métodos de estimação de parâmetros não lineares disponíveis. Diferentes cenários foram escolhidos, com diferentes conFigurações para as séries artificiais. Os cenários escolhidos foram:

Estrutura da árvore do modelo	Ordem do modelo terminal	Variáveis de transição	Parâmetros lineares	Parâmetros não lineares
	AR(1)	Y_{t-1}	[(0;0.2); (0;0.9)] [(0;-0.5); (0;0.7)]	Gama:[1;5;10] C:[0]
	AR(2)	Y_{t-1}	[(0;0.5;-0.2); (0;-0.4;0.3)]	Gama:[1;5;10] C:[0]

Estrutura da árvore do modelo	Ordem do modelo terminal	Variáveis de transição	Parâmetros lineares	Parâmetros não lineares
	AR(1)	Y_{t-1}	[(0;-0.5); (0;0.4) ; (0;0.7)]	Gama:[1;5;10] C:[0]

Nesta simulação, fixou-se o tamanho da série em T=500 e o número de execuções em N=1000. As estatísticas e ilustrações calculadas durante os testes foram descritas a seguir:

3.8.1
Média e desvio padrão do MSE

Para cada processo simulado, e para cada um dos métodos de estimação de parâmetros utilizado, foi calculada a média e o desvio padrão dos erros quadráticos médios encontrados. Os estimadores utilizados foram:

$$\overline{X} = \frac{\sum_{j=1}^{N}\left(\frac{\sum_{i=1}^{T}(y_i - \hat{y}_i)}{T}\right)}{N}$$

$$\overline{S} = \frac{\sum_{j=1}^{N}\left(\frac{\sum_{i=1}^{T}(y_i - \overline{X}_{(j)})^2}{T-1}\right)}{N} \qquad (3\text{-}14)$$

Onde y_i representa a série de dados, e $\hat{y}_i$ a estimativa

Onde $\overline{X}_{(j)}$ é a média dos erros quadráticos médios a cada iteração n do processo de N simulações, $\overline{X}$ é a média global, e $\overline{S}$ é a variância dos erros quadráticos médios.

STAR(1) – Árvore com dois nós terminais:

$$\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$$

	C=0;γ=1	C=0;γ=5	C=0;γ=10
BFGS	$\overline{X} = 0{,}9903$ $\sqrt{\overline{S}} = 0.0597$	$\overline{X} = 0{,}9866$ $\sqrt{\overline{S}} = 0.0660$	$\overline{X} = 0{,}9834$ $\sqrt{\overline{S}} = 0.0617$
Newton	$\overline{X} = 0{,}9876$ $\sqrt{\overline{S}} = 0.0623$	$\overline{X} = 0{,}9877$ $\sqrt{\overline{S}} = 0.0624$	$\overline{X} = 0{,}9845$ $\sqrt{\overline{S}} = 0.0612$
Gradiente	$\overline{X} = 0{,}9870$ $\sqrt{\overline{S}} = 0.0639$	$\overline{X} = 0{,}9889$ $\sqrt{\overline{S}} = 0.0637$	$\overline{X} = 0{,}9841$ $\sqrt{\overline{S}} = 0.0624$
Algoritmo Genético	$\overline{X} = 1{,}0171$ $\sqrt{\overline{S}} = 0.0736$	$\overline{X} = 1{,}0163$ $\sqrt{\overline{S}} = 0.0740$	$\overline{X} = 1{,}0224$ $\sqrt{\overline{S}} = 0.0733$

$$\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$$

	C=0;γ=1	C=0;γ=5	C=0;γ=10
BFGS	$\overline{X}$ = 0,9879 $\sqrt{\overline{S}}$ =0.0626	$\overline{X}$ = 0,9895 $\sqrt{\overline{S}}$ =0.0641	$\overline{X}$ = 0,9899 $\sqrt{\overline{S}}$ =0.0634
Newton	$\overline{X}$ = 0,9887 $\sqrt{\overline{S}}$ =0.0641	$\overline{X}$ = 0,9877 $\sqrt{\overline{S}}$ =0.0624	$\overline{X}$ = 0,9867 $\sqrt{\overline{S}}$ =0.0641
Gradiente	$\overline{X}$ = 0.9899 $\sqrt{\overline{S}}$ =0.0631	$\overline{X}$ = 0.9872 $\sqrt{\overline{S}}$ =0.0617	$\overline{X}$ = 0.9882 $\sqrt{\overline{S}}$ =0.0609
Algoritmo Genético	$\overline{X}$ = 1,0425 $\sqrt{\overline{S}}$ =0.0832	$\overline{X}$ = 1,0314 $\sqrt{\overline{S}}$ = 0.0807	$\overline{X}$ = 1,0417 $\sqrt{\overline{S}}$ = 0.0830

É possível constatar que a performance entre os métodos de otimização na estimação do MSE foi parecida. As estimações por algoritmos genéticos, no entanto, consistentemente ofereceram resultados inferiores aos demais.

Conforme cresce a complexidade dos modelos, e consequentemente, a quantidade de parâmetros envolvidos, o erro na estimação também aumenta.

3.8.2
Pertinência média dos nós terminais

Para cada processo simulado, e para cada um dos métodos de estimação de parâmetros utilizado, foi calculada a média e o desvio padrão das pertinências capturadas pelos nós terminais. Os estimadores utilizados foram:

$$\overline{X} = \frac{\sum_{j=1}^{N} \left(\frac{\sum_{i=1}^{T} (p_i - \hat{p}_i)}{T} \right)}{N} \qquad (3\text{-}15)$$

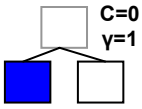
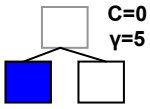
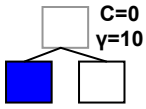
$$\overline{S} = \frac{\sum_{j=1}^{N} \left(\frac{\sum_{i=1}^{T} (p_i - \overline{P}_{(j)})^2}{T-1} \right)}{N}$$

Onde p representa cada pertinência avaliada, e $\overline{X}$, $\overline{P}_{(j)}$ e $\overline{S}$ são respectivamente a média global, a média em cada iteração n do processo de N simulações, e a variância global de cada um destas pertinências.

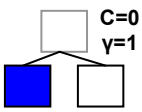
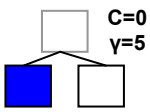
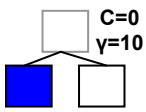
A seguir será exibido um resumo dessas estatísticas, e alguns histogramas representativos da distribuição das pertinências.

STAR(1) – Árvore com dois nós terminais:

$$\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$$

			
BFGS	$\overline{X}$ = 0,5083 $\sqrt{\overline{S}}$ = 0.2563	$\overline{X}$ = 0,3249 $\sqrt{\overline{S}}$ = 0.1950	$\overline{X}$ = 0,3014 $\sqrt{\overline{S}}$ = 0.1817
Newton	$\overline{X}$ = 0,4982 $\sqrt{\overline{S}}$ =0.2567	$\overline{X}$ = 0,3286 $\sqrt{\overline{S}}$ =0.1771	$\overline{X}$ = 0,3159 $\sqrt{\overline{S}}$ =0.1896
Gradiente	$\overline{X}$ = 0,5284 $\sqrt{\overline{S}}$ =0.2574	$\overline{X}$ = 0,3254 $\sqrt{\overline{S}}$ =0.1751	$\overline{X}$ = 0,3151 $\sqrt{\overline{S}}$ =0.1788
Algoritmo Genético	$\overline{X}$ = 0,4578 $\sqrt{\overline{S}}$ = 0.39477	$\overline{X}$ = 0,3981 $\sqrt{\overline{S}}$ = 0.3844	$\overline{X}$ = 0,4055 $\sqrt{\overline{S}}$ = 0.3978

$$\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$$

			
BFGS	$\overline{X}$ = 0,5307 $\sqrt{\overline{S}}$ = 0.2624	$\overline{X}$ = 0,3167 $\sqrt{\overline{S}}$ = 0.1547	$\overline{X}$ = 0,3172 $\sqrt{\overline{S}}$ = 0.1637
Newton	$\overline{X}$ = 0,5174	$\overline{X}$ = 0,3271	$\overline{X}$ = 0,3216

	$\sqrt{\overline{S}}$ =0.2667	$\sqrt{\overline{S}}$ =0.1527	$\sqrt{\overline{S}}$ =0.1596
Gradiente	$\overline{X}$ = 0,5169 $\sqrt{\overline{S}}$ =0.2621	$\overline{X}$ = 0,3291 $\sqrt{\overline{S}}$ =0.1464	$\overline{X}$ = 0,3092 $\sqrt{\overline{S}}$ =0.1525
Algoritmo Genético	$\overline{X}$ = 0,5040 $\sqrt{\overline{S}}$ = 0.3715	$\overline{X}$ = 0,4201 $\sqrt{\overline{S}}$ = 0.3590	$\overline{X}$ = 0,4107 $\sqrt{\overline{S}}$ = 0.3570

Quanto maior a suavidade imposta ao modelo pelo parâmetro γ, maior é a variância da pertinência estimada ao longo da estimação, o que mostra a dificuldade na estimação deste parâmetro. Conforme o mesmo parâmetro imposto ao modelo gerador cresce, mais precisa a estimação da pertinência vai se tornando. Entre os métodos de otimização, quando a suavidade é alta, a distribuição na estimação da pertinência é praticamente uniforme. Conforme a suavidade decai, esta distribuição vai ganhando forma exponencial.

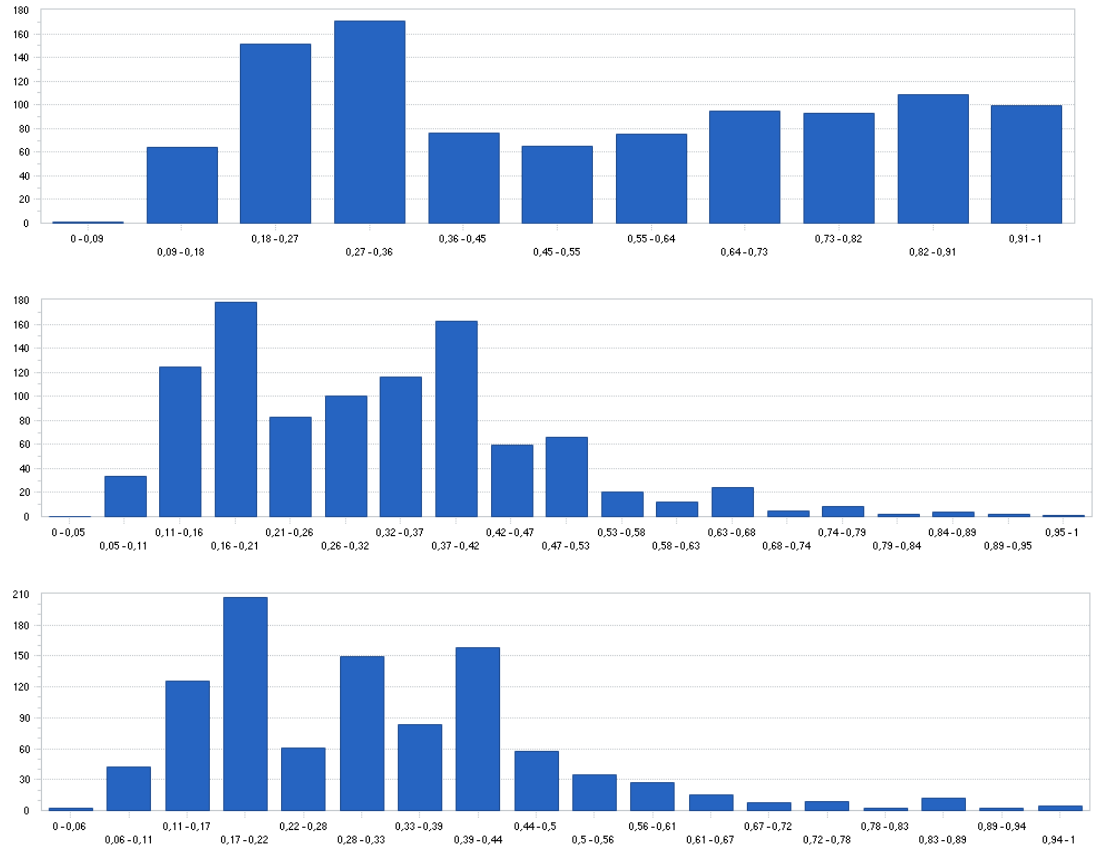


Figura 3.10: Histogramas de pertinências estimadas para um dado nó sob o método BFGS para γ=1, γ=5 e γ=10.

O mesmo não se pode dizer da estimação por algoritmos genéticos. O que se vê nesse caso é que apesar de haver uma redução gradual no desvio padrão da pertinência, os histogramas de suas distribuições mostram claramente uma alternância entre valores altos e baixos de pertinência para os nós avaliados. Isto mostra que a suavidade não é uma característica bem capturada por este tipo de estimação .

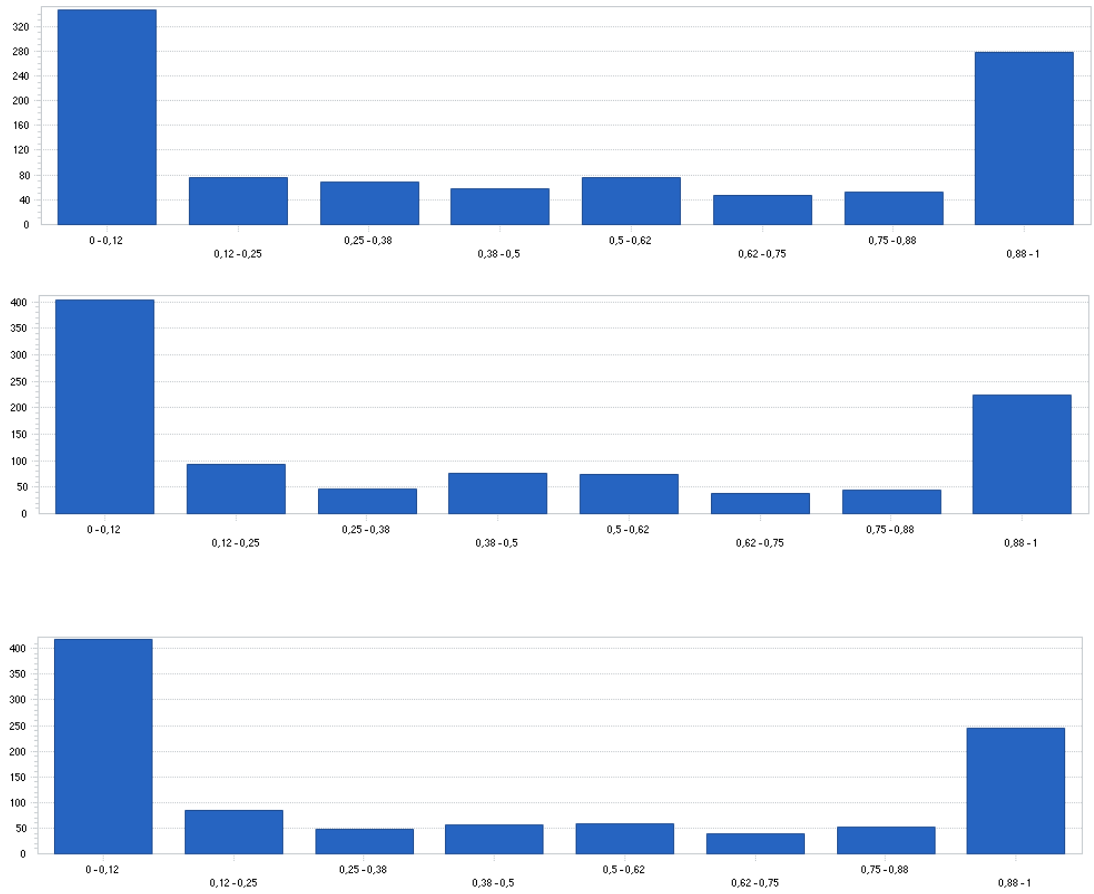


Figura 3.11: Histogramas de pertinências estimadas para um dado nó sob algoritmo genético para γ=1, γ=5 e γ=10.

Este padrão se repete para todos os modelos estudados, em menor ou maior grau.

3.8.3
Estimação dos parâmetros não-lineares

A dificuldade de estimação dos parâmetros não-lineares por de todos os métodos resultou em uma pequena quantidade de valores excessivamente altos ou baixos que distorceram sua avaliação através de estatísticas como média e variância. A seguir serão exibidos alguns gráficos dos pares de parâmetros não lineares estimados para cada um dos modelos.

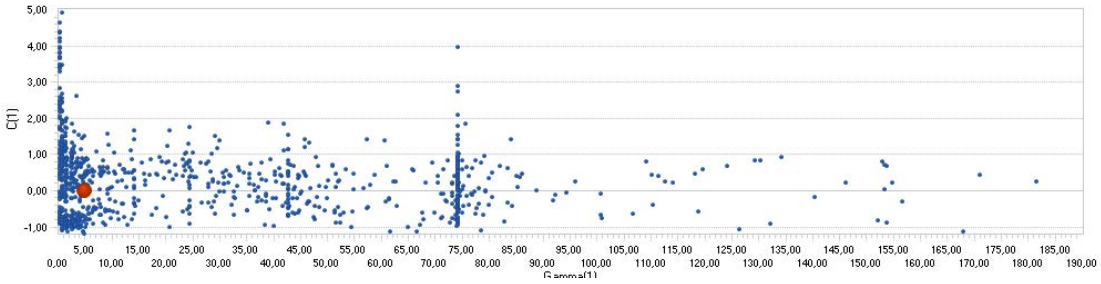


Figura 3.12: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$, estimado pelo método BFGS e γ=5.

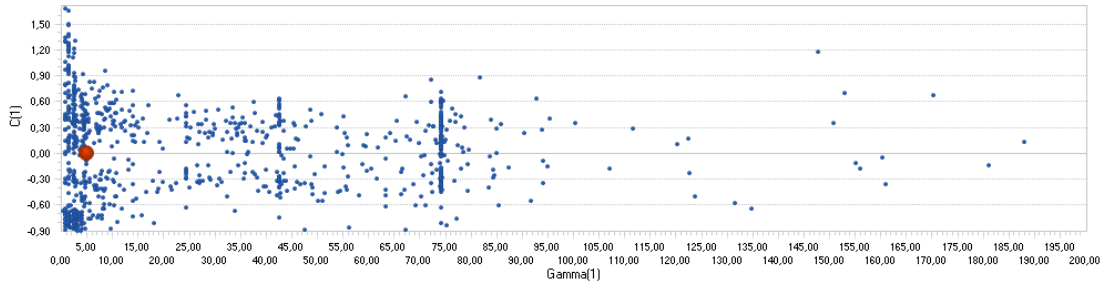


Figura 3.13: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$, estimado pelo método BFGS e γ=5.

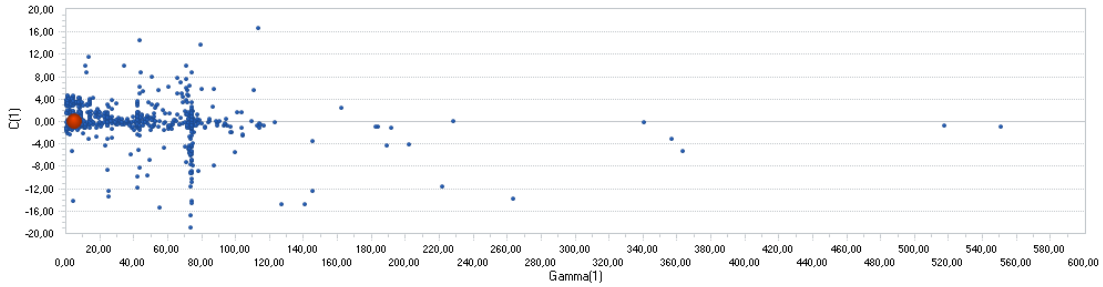


Figura 3.14: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ 0.5 \\ -0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ -0.4 \\ 0.3 \end{bmatrix}$, estimado pelo método BFGS e γ=5.

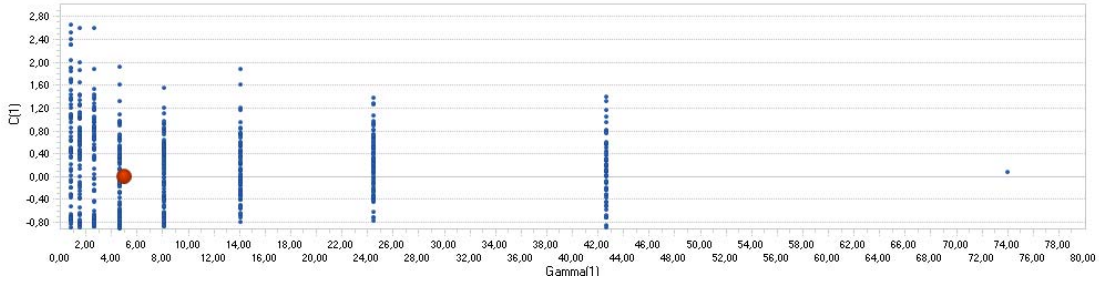


Figura 3.15: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$, estimado pelo método de Newton e γ=5.

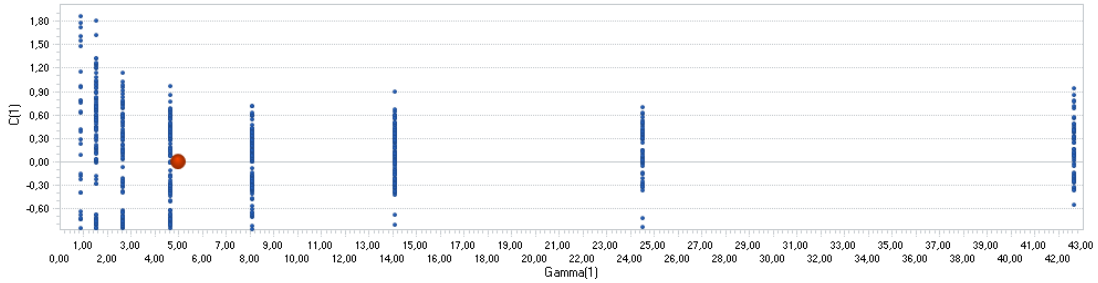


Figura 3.16: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$, estimado pelo método de Newton e γ=5.

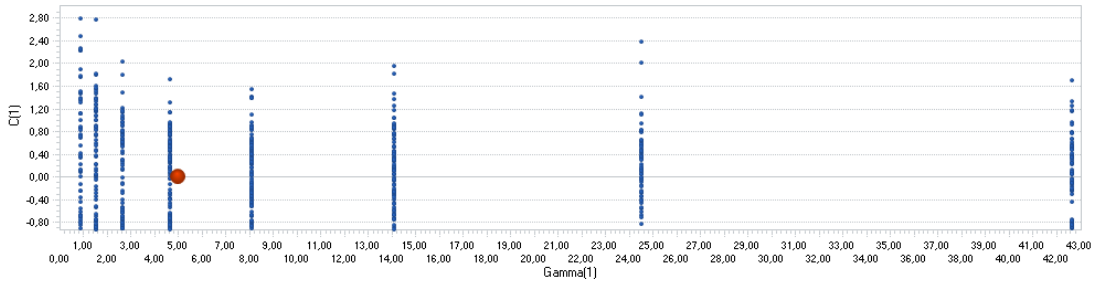


Figura 3.17: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$, estimado pelo método do Gradiente e γ=5.

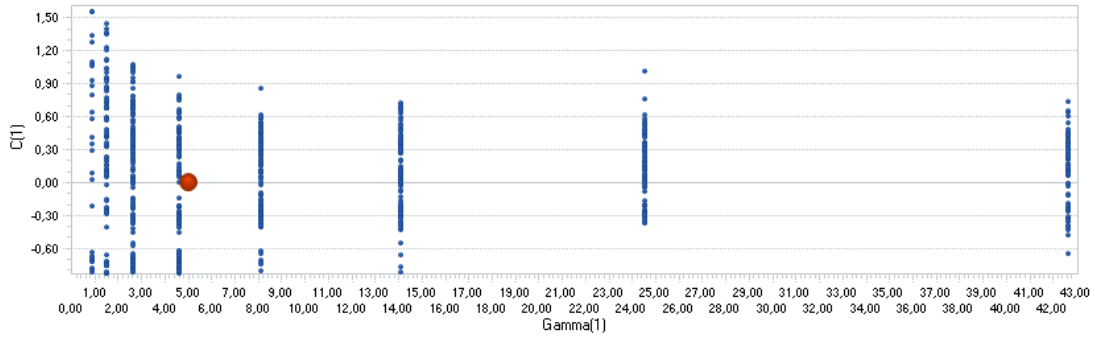


Figura 3.18: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$, estimado pelo método do Gradiente e γ=5.

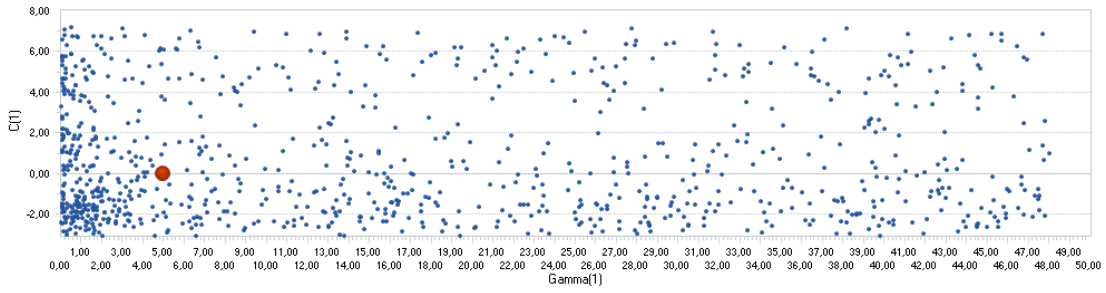


Figura 3.19: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$, estimado pelo algoritmo genético e γ=5.

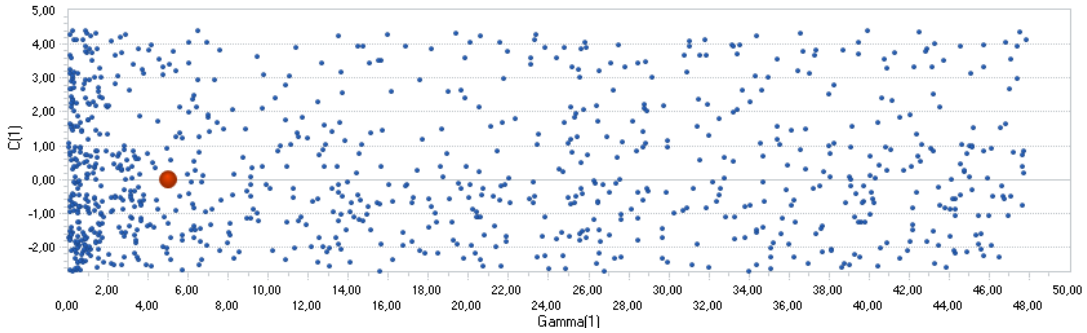


Figura 3.20: Dispersão dos parâmetros de localização e suavidade para $\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$, estimado pelo algoritmo genético e γ=5.

Os métodos de Newton e gradiente demonstraram ser menos eficientes na estimação dos parâmetros de suavidade. Fica claro pelos gráficos mostrados acima que suas estimações ficaram praticamente presas aos valores fornecidos pelos grids iniciais da estimação. Em todos os métodos de otimização utilizados (BFGS, Newton, e gradiente), ficou visível uma relação entre o valor do parâmetro de suavidade e a variância na estimação do parâmetro de localização. Quanto mais brusca a divisão estimada entre regimes, em geral menor era a variância da localização estimada. O método BFGS apresentou algumas estimações bem destoantes do parâmetro de suavidade, algumas inclusive desprezadas para a construção dos gráficos, porém pode-se considera-las desprezíveis em quantidade diante de um experimento com mil simulações para cada modelo estudado.

O algoritmo genético não demonstrou o mesmo padrão descrito acima. A relação entre suavidade e localização se mostrou bem mais dispersa. A aleatoriedade presente no processo de evolução entre gerações deste método pode explicar parte deste resultado. Outro fator importante é que, ao contrário dos métodos de otimização, o algoritmo genético aplicado aqui estimou conjuntamente parâmetros lineares e não-lineares.

3.8.4
Estimação dos parâmetros lineares

Percebe-se uma relação entre a baixa representatividade estimada pela pertinência absorvida pelos nós terminais e a qualidade da estimação dos parâmetros lineares. Em geral, nós com pertinência muito baixa tendem a ser muito mal estimados, produzindo resultados absurdos, porém irrelevantes. Isto ocorreu com freqüência em dois tipos de situação. Primeiro, quando os parâmetros lineares dos pares de nós terminais pertencentes ao mesmo nó de decisão eram bastante próximos, pois na prática, um dos nós absorvia praticamente toda a pertinência. Isto equivale a dizer que um modelo de ordem inferior seria suficiente para estimar a mesma série. Outra situação aonde foram encontradas dificuldades na detecção dos parâmetros lineares foi quando a variância na estimação da pertinência dos nós era alta. Isso acontecia particularmente com freqüência na estimação através de algoritmo genético. A Figura 3.21 mostra um exemplo de boa detecção de parâmetros:

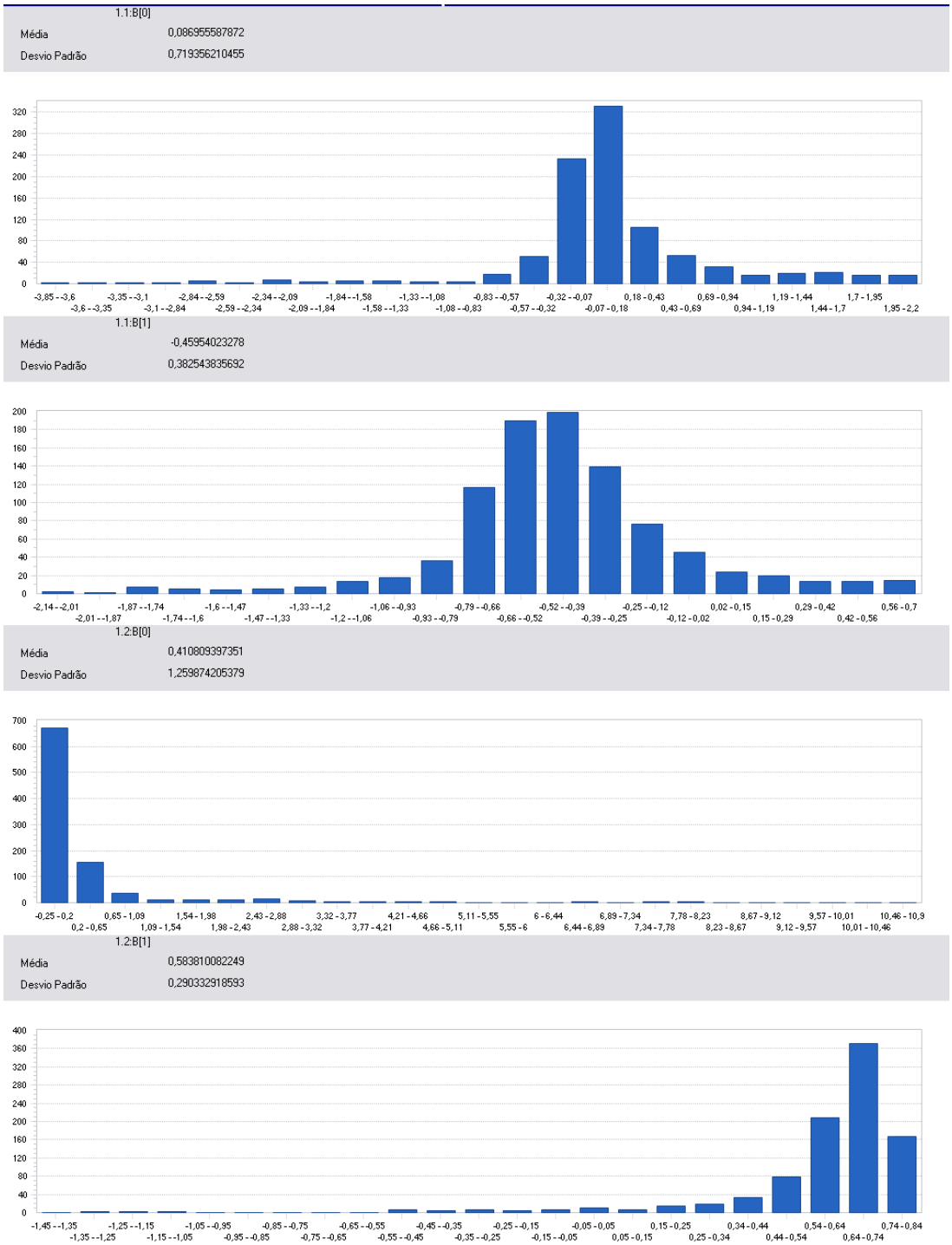


Figura 3.21: Estimação dos parâmetros lineares através do método BFGS para o modelo com $\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$ e γ=5.

É possível perceber que a estimação consistentemente se aproximou dos parâmetros do modelo gerador ao longo da simulação, pois após mil rodadas, a variância na estimação destes parâmetros foi consideravelmente baixa.

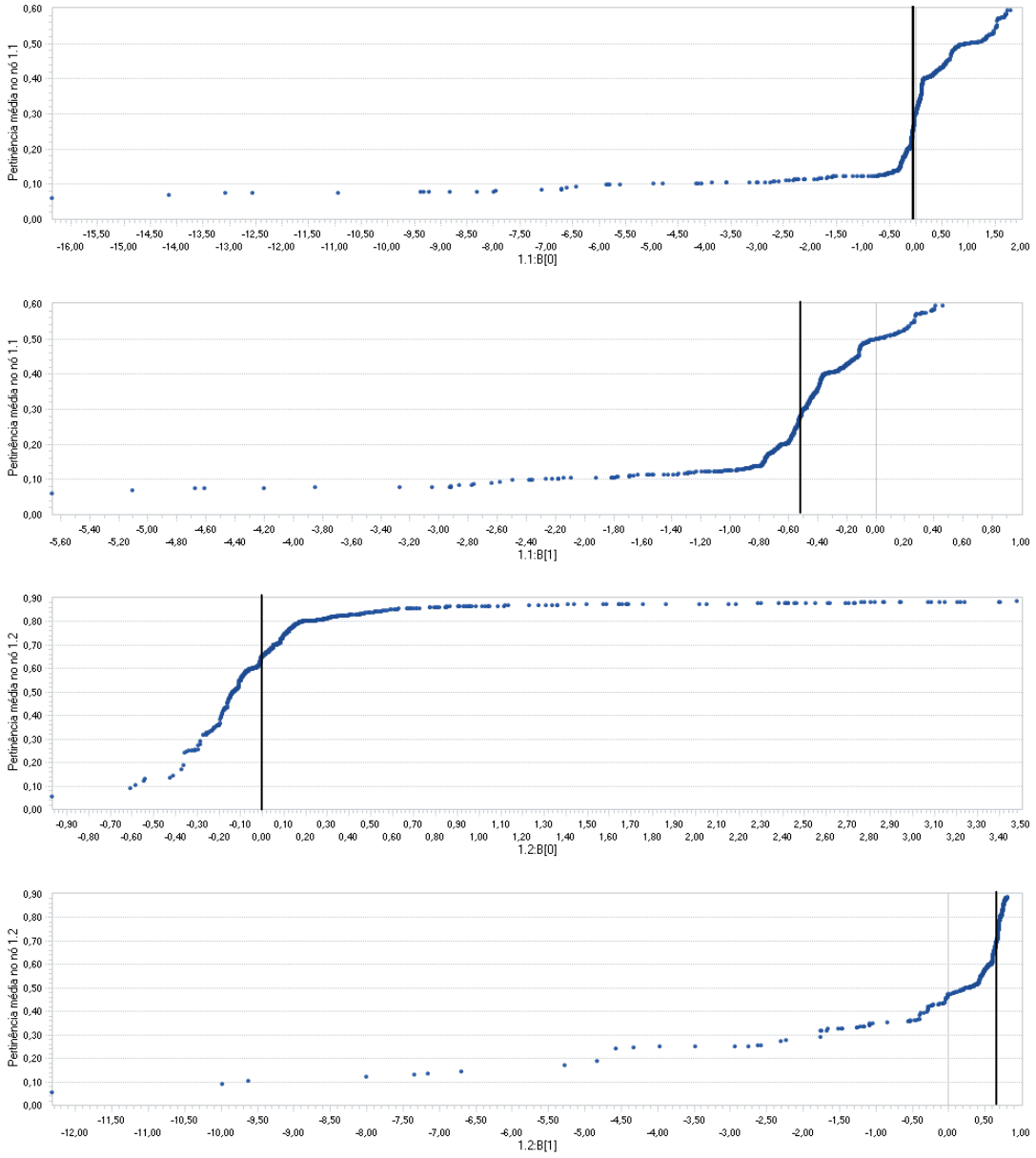


Figura 3.22: Valores detectados para os parâmetros lineares para cada uma das pertinências encontradas através do método BFGS para o modelo com $\beta_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.7 \end{bmatrix}$ e γ=5.

A Figura 3.22 mostra como ocorrem distorções na estimação de parâmetros quando a pertinência média é muito baixa. Quando um nível de relevância mínimo é atingido, valores razoáveis são atribuídos para os parâmetros. Onde há maior concentração de pertinências estimadas, há certa estabilidade na valoração dos parâmetros. Fora dela, a dispersão é bem maior. A sobrevaloração da pertinência também produz resultados ruins, pois quando um nó absorve quase que a

totalidade da pertinência, novamente caímos na situação em que a estimação equivale àquela feita por um modelo de ordem inferior, aonde apenas um modelo predomina. No exemplo da Figura 3.22, este efeito é visível no parâmetro de nível do nó da direita (terceiro gráfico na seqüência).

A estimação por algoritmos genéticos, como já foi descrito anteriormente, consistentemente detectou variações bruscas entre regiões de transição. Em uma mesma simulação, diferentes nós terminais absorveram a pertinência quase que na totalidade, de forma alternante. Isto fez com que a medida de média e variância dos parâmetros ao longo da simulação se tornasse absurda, pois os valores estimados para nós com importância próxima de zero produziram valores desprezíveis.

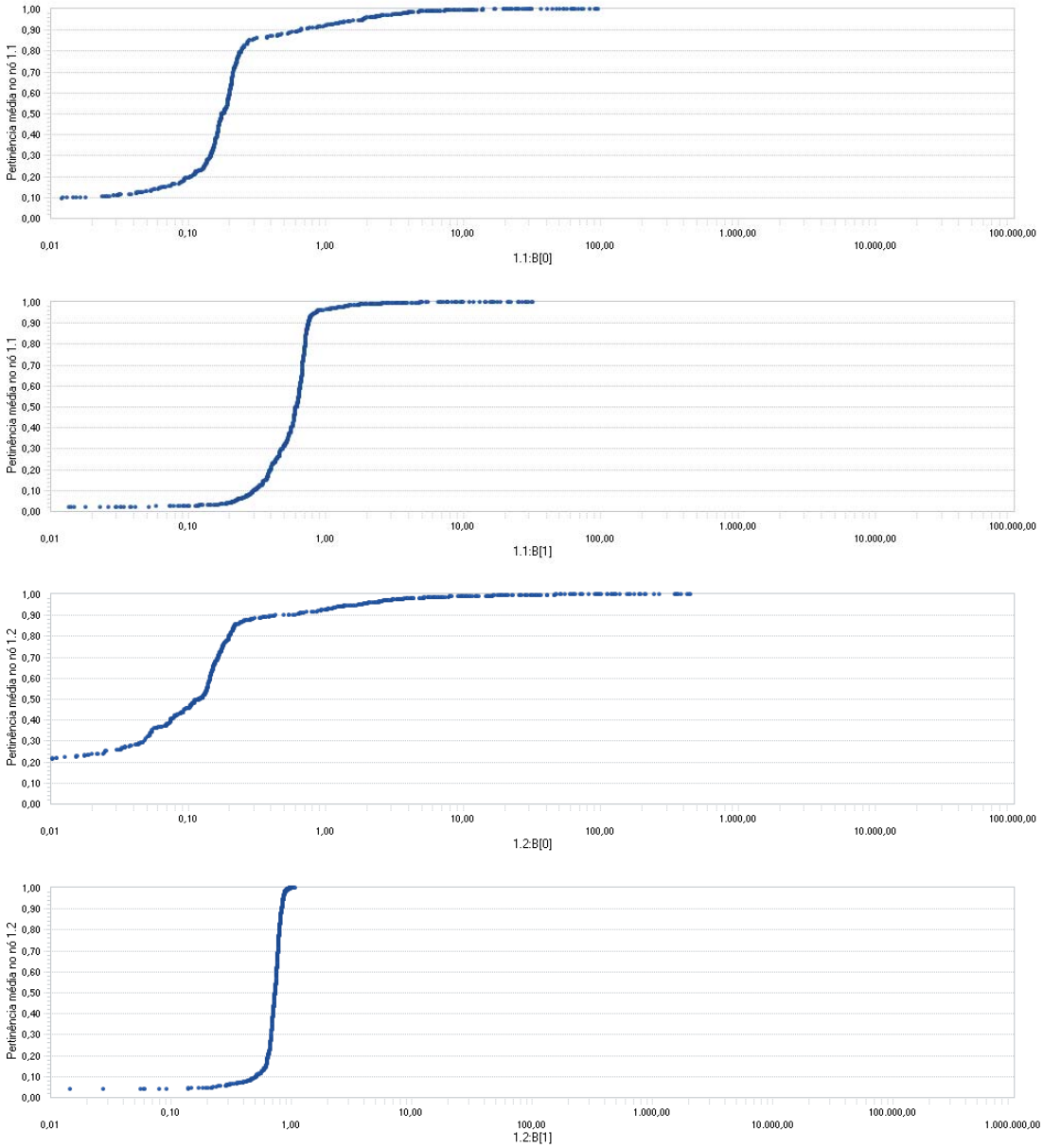


Figura 3.23: Valores detectados para os parâmetros lineares para cada uma das pertinências encontradas através de algoritmos genéticos para o modelo com $\beta_0 = \begin{bmatrix} 0 \\ 0.2 \end{bmatrix}; \beta_1 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$ e γ=5 (eixo X em escala logarítmica).

Conclusões

As simulações possibilitaram constatar que, independentemente do método utilizado, há uma relação entre a variabilidade do parâmetro de localização e o parâmetro de suavidade detectados. Conforme o parâmetro de suavidade estimado cresce, menor é a variância do parâmetro de localização. Isto acontece, pois quanto mais o parâmetro de suavidade é reduzido, mais irrelevante vai se tornando o parâmetro de localização, e incrementos ou decréscimos no mesmo não se traduzem em redução significativa do erro na estimação. Este efeito é menos visível na estimação por meio de algoritmos genéticos, pois a aleatoriedade introduzida por conceitos como cruzamento e mutação permite que uma combinação mais distinta de resultados prevaleça.

Processos muito suaves tornam difícil a estimação, independentemente do método. O resultado habitual foi encontrar uma concentração grande da pertinência estimada em apenas um dos modelos lineares, com baixa variância em seus parâmetros, e modelos desprezíveis nos demais nós, devido à baixa representatividade dos mesmos nos dados. Um resultado similar ocorre quando se gera séries a partir de modelos lineares muito próximos. O modelo estimado novamente se aproxima de um modelo linear único, capturando a maior parte da pertinência, e modelos irrelevantes nos outros nós, com parametrizações incoerentes.

Os métodos de otimização se mostraram mais eficientes do que os algoritmos genéticos numa relação custo benefício, em especial o BFGS, o mais rápido deles. O fato deste método apresentar uma alternativa ao cálculo da inversa da matriz Hessiana da função, o torna ao mesmo tempo mais robusto e rápido que os demais. Assim como o método de Newton, ele converge em menos iterações, devido à informação trazida pela pseudo-inversa calculada a cada passo. Porém, cada iteração sua é mais veloz, pois não envolve inversão nem cálculo de segunda derivada. Isso se torna mais evidente conforme a complexidade do vetor de parâmetros aumenta. O método do Gradiente se mostrou o mais ineficiente deles, por apresentar maiores dificuldades de convergência do que os demais. Sua implementação mais simples, envolvendo apenas o cálculo de gradientes, não

compensa a quantidade de iterações que necessita para atingir resultados satisfatórios.

A utilização de algoritmos genéticos apresentou vantagens e desvantagens. Para se tornar uma alternativa viável aos métodos de otimização, com resultados melhores em situações de difícil estimação, é necessário parametrizá-lo com uma quantidade muito grande de gerações e um tamanho de população também grande. Vale lembrar que cada avaliação de indivíduo da população incorre em uma aplicação do modelo sobre a série e do cálculo do somatório dos erros quadráticos, e isto pode ocorrer milhares de vezes. O problema desta abordagem é que o custo computacional envolvido pode tornar a abordagem de pouco uso prático. Além disso, a quantidade excessiva de parâmetros necessários para conFigurar uma estimação por algoritmos genéticos torna esse processo muito ad-hoc. Ainda assim, foi possível constatar que melhores resultados são obtidos quando se escolhe uma taxa alta de *cross-over* no início da estimação e baixa no final, assim como uma taxa de mutação baixa no início e alta no fim, e um nível de *steady-state* médio ao longo da estimação. Uma explicação razoável pode ser que no começo da estimação, é importante cruzar os melhores resultados para aumentar a proporção de indivíduos aptos na população. Conforme a população vai evoluindo, é interessante reduzir o percentual de cruzamento para não uniformizar a população. Ao mesmo tempo, é conveniente aumentar o percentual de mutação ao fim para introduzir aleatoriedade em uma população cada vez mais parecida, ao passo que fazer isto no início não traz grandes benefícios. O *steady-state* em um nível médio garante que uma quantidade significativa de indivíduos aptos migre de geração em geração, acelerando a convergência para um resultado próximo do ótimo.

A utilização de grades de inicialização com diferentes combinações de parâmetros não lineares se mostrou eficiente para reduzir o risco de incorrer em mínimos locais. Porém, conforme se aumenta a complexidade da árvore, a quantidade de combinações envolvidas para se iniciar a estimação se torna muito grande. A alternativa escolhida para contornar o problema foi estabelecer um número máximo de combinações a se testar, e quando esse número excedesse o limite, se amostrar do conjunto total a quantidade máxima de combinações estabelecida.