ESTIMADOR *BOOTSTRAP* NÃO-PARAMÉTRICO DE CURVAS DE SOBREVIVÊNCIA PARA DADOS ENTOMOLÓGICOS COM CENSURA INTERVALAR

GRAZIELA DUTRA ROCHA GOUVÊA

2006

Ficha Catalográfica Preparada pela Divisão de Processos Técnicos da Biblioteca Central da UFLA

Gouvêa, Graziela Dutra Rocha
Estimador *Bootstrap* não-paramétrico de curvas de sobrevivência para dados entomológicos com censura intervalar / Graziela Dutra Rocha Gouvêa. -- Lavras : UFLA, 2006.
61 p. : il.

Orientador: Mário Javier Ferrua Vivanco
Dissertação (Mestrado) – UFLA.
Bibliografia.

1. Censura intervalar. 1. Método *Bootstrap*. 3. Algoritmo EM. 4. Dado entomológico. 5. Estimação Não-Paramétrica. I. Universidade Federal de Lavras. II. Título.

CDD-519.54

GRAZIELA DUTRA ROCHA GOUVÊA

ESTIMADOR *BOOTSTRAP* NÃO-PARAMÉTRICO DE CURVAS DE SOBREVIVÊNCIA PARA DADOS ENTOMOLÓGICOS COM CENSURA INTERVALAR

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Curso de Mestrado em Agronomia, área de concentração em Estatística e Experimentação Agropecuária, para obtenção do título de Mestre.

Orientador

Prof. Dr. Mário Javier Ferrua Vivanco

Co-orientador

Prof. Dr. Fortunato Silva de Menezes

LAVRAS
MINAS GERAIS – BRASIL
2006

GRAZIELA DUTRA ROCHA GOUVÊA

ESTIMADOR *BOOTSTRAP* NÃO-PARAMÉTRICO DE CURVAS DE SOBREVIVÊNCIA PARA DADOS ENTOMOLÓGICOS COM CENSURA INTERVALAR

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Curso de Mestrado em Agronomia, área de concentração em Estatística e Experimentação Agropecuária, para obtenção do título de Mestre.

APROVADA em 10 de março de 2006.

Prof. Dr. Fortunato Silva de Menezes	UFLA
Prof. Dr. Júlio Neil Cassa Louzada	UFLA
Prof. Dr. Ruben Delly Veiga	UFLA

Prof. Dr. Mário Javier Ferrua Vivanco
UFLA
(Orientador)

LAVRAS
MINAS GERAIS - BRASIL

“De tudo ficarão três coisas:
a certeza de estar sempre começando,
a certeza de que é preciso continuar e
a certeza de ser interrompido antes de terminar.”

Fernando Sabino

Aos meus pais, Adimaldo Dutra Rocha e
Maria José de Gouvêa Dutra Rocha
pelo incentivo, amor incondicional
e confiança em mim depositada.

AGRADECIMENTOS

Hoje eu sei, tenho muito a agradecer...

Aos meus pais, Adimaldo e Maria José; aos meus irmãos, Sanderson, Maldon, Sandra e Adalgiza; aos meus cunhados, Pedro, Dani, Polly; e aos meus sobrinhos; Yann, Yáskara e Sanderson Filho. À minha família, que tanto me incentivou e me deu forças; que é o meu chão, a minha segurança, a minha vida; que eu tanto amo, agradeço pela compreensão e pelo apoio sem os quais eu jamais conseguiria realizar este trabalho.

À Universidade Federal de Lavras, em especial ao Departamento de Ciências Exatas, pela oportunidade de concretização deste trabalho.

Aos professores Mário Vivanco e Fortunato Menezes, pelas sugestões, críticas e ensinamentos durante o curso.

Aos professores do programa de Pós-Graduação em Estatística e Experimentação Agropecuária, especialmente ao Lucas, que tanto me ajudou no decorrer do curso.

A todos os professores do DEX, pelo incentivo e amizade demonstrados nos corredores do Departamento. Também à Selminha, à Edila e à Maria, pelo carinho e pela disposição em ajudar.

À família que encontrei em Lavras, que são meus amigos do curso de mestrado. Jamais esquecerei um minuto sequer da nossa convivência, os momentos de risos e de choro, de festas e de estudos. Vocês se tornaram muito importantes para mim: Diney, Baiana, Josi, Verônica, Rejane, Elias, Eustáquio, Lívia, Nádia, Devanil, Paulinho e Charles.

Agradeço, também, aos amigos das outras turmas de mestrado, doutorado e a todos os amigos que fiz em Lavras, especialmente Paty Campo Belo, Luciane, Luciene, Paty Ipatinga, Francisca, Adilson, Kim, Anderson,

Delly e Osvaldo, por todo o carinho, e também à Carla, pela ajuda com as abelhas.

A todos da Pensão da Dona Itinha que me receberam com tanto carinho e me fizeram sentir em casa.

Ao Jules, e também aos meus amigos de Caratinga, especialmente a Landinha e família, por compreenderem minha ausência em tantos momentos.

À CAPES e ao CNPQ, pelo apoio financeiro.

A Deus, por ter colocado todas essas pessoas no meu caminho. Eu sei que Ele as escolheu de modo que a minha vida pudesse ser mais feliz, e de maneira que eu conseguisse chegar aonde cheguei.

Muito obrigada!

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

RESUMO

GOUVÊA, Graziela Dutra Rocha. **Estimador *Bootstrap* não-paramétrico de curvas de sobrevivência para dados entomológicos com censura intervalar.** 2006. 61 p. Dissertação (Mestrado em Agronomia/Estatística e Experimentação Agropecuária) - Universidade Federal de Lavras, Lavras, MG. [1]

Este trabalho teve como objetivo estimar e comparar curvas de sobrevivência para dados entomológicos que envolvem censura intervalar. Isso com o intuito de comparar dois tipos de alimentos artificiais (solução aquosa de mel a 50% e solução aquosa de frutose a 50%) usados no suprimento da necessidade alimentar de abelhas durante a entressafra (escassez de néctar e pólen). Inicialmente, utilizou-se o algoritmo EM (Esperança-Maximização) para se alcançar tal objetivo. Foi observado que não há diferença dos saltos (*steps*) no Estimador Não-Paramétrico de Máxima Verossimilhança (ENPMV), quando se aplica esse algoritmo ou quando se determinam tais estimativas por meio de contagens. Para melhorar as estimativas obtidas por contagem, aplicou-se o método *Bootstrap* (Não-Paramétrico) para estimar os saltos e, posteriormente, construir as curvas de sobrevivência. Concluiu-se, então, que, para dados entomológicos, o método *Bootstrap* fornece boas estimativas, além de possibilitar a construção de intervalos de confiança que permitem comparar as curvas de sobrevivência sem a necessidade de se estabelecer um método específico de comparação para esse tipo de dados. Foram utilizados dados reais referentes à longevidade de abelhas da espécie *Apis mellifera* que foram coletadas no Apiário Experimental da Universidade Federal de Lavras (UFLA). Concluiu-se, também, que o alimento que proporcionou maior longevidade aos insetos foi solução aquosa de mel a 50%.

[1] Comitê orientador: Prof. Dr. Mário Javier Ferrua Vivanco – UFLA (Orientador), Prof. Dr. Fortunato Silva de Menezes – UFLA (Co-orientador).

ABSTRACT

GOUVÊA, Graziela Dutra Rocha. **Non-parametric Bootstrap Estimator of survival curves for interval censored entomological data.** 2006. 61 p. Dissertation (Master in Agronomy/Statistics and Agricultural Experimentation) - Universidade Federal de Lavras, Lavras, MG. [2]

The goal of this work was estimate and compare survival curves for interval censored entomological data. More specifically, compare the survival curves of bees fed with two artificial solutions: 50% aqua honey solution and 50% aqua fructose solution. No differences was found between the non-parametric maximum likelihood estimate obtained by the EM algorithm and the estimated obtained by the counting method. In order to improve the estimates of the previous result, the bootstrap method was applied. This method provides good estimates and confidence intervals which allow the comparison between the survival curves. The experiment was conducted in the Insect Biology Laboratory of the Entomology Department of the Universidade Federal de Lavras-Lavras, MG, Brazil , at $28 \pm 2°C$, UR 70 ± 10 % and 12-hour photophase. The final conclusion was that longer expected time life is obtained when Apis mellifera bees are fed with honey aqua solution.

[2] Guidance Committee: Dr. Mário Javier Ferrua Vivanco – UFLA (Supervisor), Dr. Fortunato Silva de Menezes – UFLA (Co-Supervisor).

1 INTRODUÇÃO

1.1 Aspectos gerais

A Análise de Sobrevivência é uma das áreas da Estatística que mais tem crescido nos últimos tempos, principalmente pela sua importância em estudos clínicos. A principal característica dos dados de sobrevivência é a presença de censura (observação parcial da resposta), que pode se apresentar em três tipos: à direita, à esquerda e intervalar.

A censura intervalar, que será tratada neste estudo, é caracterizada pelo fato de não ser conhecido o tempo exato de ocorrência de um evento, sabendo-se somente que esse ocorreu num determinado intervalo de tempo. Sendo assim, como poderiam ser estimadas as curvas de sobrevivência com esse tipo de dado?

Vários autores resolveram o questionamento acima, considerando os tempos de observação aleatórios, o que, na prática, geralmente não ocorre, uma vez que, na maioria dos casos, o próprio pesquisador fixa os tempos de observação, de maneira que isso possa facilitar o seu trabalho.

Em várias áreas do conhecimento, é possível encontrar dados com censura intervalar. Uma área em que é bem comum esse tipo de censura é a Entomologia (parte da Zoologia que estuda os insetos).

O trato com os insetos, muitas vezes, pode ser bastante difícil. Muitos deles têm um tempo de vida curto, podendo chegar a poucas horas; outros vivem dias, e, para determinar o tempo até a sua morte, devem-se realizar experimentos com visitas periódicas para se estimar o momento em que isso ocorre, já que não é possível acompanhá-lo continuamente no decorrer de sua existência.

O método utilizado pelos profissionais e estudioso da área de Entomologia tem sido criticado por muitos pesquisadores pelo fato de que, ao se realizar o experimento, na maioria das vezes, o tempo de vida do inseto é

considerado como sendo a extremidade à direita do intervalo de observação, o que pode causar perda de informações e levar a inferências inválidas.

O enfoque deste trabalho não está baseado em dados fornecidos pelas extremidades dos intervalos, mas pelas probabilidades (saltos) de que os insetos morram em um determinado intervalo de tempo. Assim, procura-se construir curvas de sobrevivência a partir desses saltos.

A abordagem aqui apresentada considera dados reais do tempo de vida de abelhas da espécie *Apis mellifera*, insetos sociais que vivem em colônias e produzem mel, própolis e cera, a qual é composta de álcoois, ácidos graxos, ésteres, hidrocarbonetos e vitamina A, muito utilizados pelo homem na indústria química, alimentícia, farmacêutica e na fabricação de produtos artesanais.

Segundo Brighenti (2003), o Brasil possui grande potencial apícola, especialmente em função de sua extensa área territorial e também pela diversidade de plantas nectaríferas e poliníferas. O potencial brasileiro de produtos apícolas está avaliado em 360 milhões de dólares anuais, podendo chegar, em curto prazo, a um bilhão, sendo, portanto, de grande importância para a economia do país.

Existem vários métodos computacionais para estimar curvas de sobrevivência na presença de censura intervalar. Entre eles, será estudado o algoritmo EM (Esperança-Maximização) e será proposto o Método *Bootstrap*, como alternativa, para tratar, exclusivamente, dados entomológicos.

O algoritmo EM é um método iterativo usado para localizar o valor de um parâmetro que maximiza a função de verossimilhança, sendo que cada iteração é constituída por dois passos, chamados E-*step* e M-*step*. Primeiramente, é calculada a esperança da variável aleatória completa em relação à distribuição condicionada à variável aleatória observada. Em seguida, ocorre a maximização da função obtida no E-*step* em relação a um parâmetro desconhecido.

O Método *Bootstrap* consiste em uma técnica de reamostragem que permite aproximar a distribuição de uma função das observações a partir da distribuição empírica dos dados. Por meio desse método, podem ser estimados as medidas de posição, de dispersão e os intervalos de confiança, com o intuito de fazer inferência sobre os parâmetros em questão.

1.2 Objetivos

Baseando-se nas probabilidades (saltos) de que os eventos ocorram em um determinado intervalo de tempo, os objetivos deste trabalho são:

(1) estimar as curvas de sobrevivência da longevidade de insetos, utilizando dados censurados intervalarmente;

(2) comparar dois tipos de alimentos artificiais que possam suprir a alimentação de abelhas durante a entressafra (escassez de néctar e pólen), em substituição a uma apicultura migratória;

(3) estudar a possibilidade de adaptação dos algoritmos utilizados no tratamento de censura intervalar com intervalos aleatórios, para o caso em que os intervalos são fixos.

.

2 REFERENCIAL TEÓRICO

Modelos de análise de sobrevivência têm por objetivo estudar dados de experimentos em que, geralmente, a variável resposta é o tempo até a ocorrência de um evento de interesse. A principal característica dos dados de sobrevivência é a presença de censura (observação parcial da resposta).

Existem três tipos de censura: censura à direita, censura à esquerda e censura intervalar; e três mecanismos de censura: censura tipo I, censura tipo II e censura aleatória.

Dados com censura intervalar aparecem quando o tempo de falha T não pode ser observado, mas pode ser determinado um intervalo no qual o tempo de falha está contido (Huang & Wellner, 1996). Muitos estudos envolvendo esse tipo de censura podem ser encontrados na literatura.

O primeiro trabalho em estimação não-paramétrica da função de distribuição acumulada (FDA), ou, equivalentemente, função de sobrevivência, na presença de censura intervalar, foi atribuído a Peto (1973). Nesse trabalho, foi descrito o Algoritmo de Newton-Raphson para o Estimador Não-Paramétrico de Máxima Verossimilhança (ENPMV) da função de distribuição acumulada. Turnbull (1976) derivou o mesmo estimador que Peto (*idem*), usando um algoritmo iterativo de autoconsistência: o algoritmo EM (Esperança-Maximização).

Segundo Ng (2002), o método proposto por Peto (*ibidem*) tende a concentrar massas de probabilidade nas extremidades dos intervalos. Então, ele propôs uma modificação que supera esse problema. Giolo (2004) descreveu e ilustrou o procedimento iterativo proposto por Turnbull (1976) para estimar a função de sobrevivência que foi, por ela, implementado no software R. A autora usou, para ilustrar o método e o uso do R, um estudo apresentado por Klein & Moeschberger (1997), que foi realizado para comparar a eficiência do

tratamento, utilizando somente radioterapia contra radioterapia combinada à quimioterapia em mulheres com câncer de mama, em que o evento de interesse era o tempo até o aparecimento de retração de peito.

Vários autores estudaram o algoritmo EM, dentre eles Cox & Okes (1984), Dempster et al. (1977) e Tanner (1996), os quais apresentaram alguns exemplos em que o uso do EM é adequado, bem como as propriedades gerais desse algoritmo.

Meng & Rubin (1991) estudaram a taxa de convergência do algoritmo EM e Jamshidiam & Jennrich (1997) propuseram métodos para acelerar a convergência do algoritmo baseados no método Quase-Newton. Jamshidiam & Jennrich (2000), então, estudaram o erro-padrão produzido pelo algoritmo EM e o compararam com outros métodos de estimação.

Lindsey & Ryan (1998) introduziram um método não-paramétrico para testes de hipóteses que permite a comparação entre dois grupos com censura: à direita, à esquerda ou intervalar.

Outros métodos para comparação entre grupos com censura foram introduzidos na literatura, mas poucos são usados por causa da falta de softwares facilmente disponíveis. Devido a essa carência, uma abordagem comum é assumir que o evento aconteceu no fim (ou início, ou ponto central) de cada intervalo e, então, aplicarem-se métodos-padrão. Porém, essa abordagem pode levar a inferências inválidas que, em particular, tendem a menosprezar os erros-padrão dos estimadores dos parâmetros de interesse.

A comparação de funções de sobrevivência tem sido muito estudada por vários autores. Pepe & Fleming (1989) introduziram um teste para dados que envolvem censura à direita. Petroni & Wolfe (1994) consideram um teste similar em que os tempos de sobrevivência podem ser tomados como um número finito de valores. Fang et al. (2002) estenderam esses testes para o caso geral de dados que envolviam o caso 1 de censura intervalar, derivaram a distribuição

assintótica da estatística do teste, apresentaram um procedimento *Bootstrap* e também aplicaram o método proposto para dados com censura intervalar num estudo de câncer de mama.

Gentleman & Vandal (2001) apresentaram métodos para encontrar o ENPMV da função de distribuição, usando uma abordagem de teoria gráfica que simplifica os problemas envolvendo censura intervalar. Dentre os métodos de estimação, foram apresentados o algoritmo EM e métodos de isotonização. Hudgens (2005) também apresentou uma abordagem teórica gráfica para descrever o ENPMV para a função de distribuição acumulada para dados com censura intervalar e dados truncados à esquerda. Assim, uma condição necessária e suficiente para a existência de um ENPMV foi provada.

Huang & Wellner (1996) definiram o caso 1 e o caso 2 de censura intervalar e também um esquema geral desse tipo de censura. Os autores mostraram como calcular o ENPMV nos dois casos, bem como os modelos de regressão nos casos 1 e 2 de censura intervalar. Geskus & Groeneboom (1999) revelaram a eficiência assintótica do ENPMV para o caso 2 de censura intervalar.

Jonbloed (1998) comparou os algoritmos EM e ICM (Algoritmo Iterativo do Minorante Convexo) para o cálculo do ENPMV e concluiu que o algoritmo ICM converge mais rapidamente na presença de censura intervalar (caso 2) do que o EM. Assim, o autor propôs uma modificação do algoritmo ICM e provou que essa modificação é globalmente convergente.

Para calcular o ENPMV da função de sobrevivência na presença de censura dupla e para o caso 2 de censura intervalar, Zhang & Jamshidiam (2004) propuseram três algoritmos: EM, ICM e GR (generalização do algoritmo Rosen). Os autores, então, definiram o Estimador Não-Paramétrico de Máxima Verossimilhança da função de sobrevivência como uma função *step* que tem saltos apenas nos pontos de observação e maximizam o log-verossimilhança,

levando a um único ENPMV para os dois tipos de censura por eles considerados.

Pan (2000) sugeriu a suavização da função de sobrevivência em casos envolvendo censura intervalar pelos métodos do núcleo e *logspline*, fazendo uma comparação entre esses métodos mediante simulações. A suavização pelo núcleo é um método não-paramétrico de estimação de uma função densidade de probabilidade. As diferentes aplicações do núcleo estimador podem ser encontradas em Silverman (1986) e Simonoff (1996), dentre outros.

Rodarte (2005) estudou os intervalos de confiança para a função de distribuição na presença de censura intervalar (caso 1). Assim, o autor descreve quatro diferentes métodos para a estimação e construção desses intervalos: o ENPMV que usa um algoritmo para o cálculo de regressões isotônicas, uma variação do ENPMV suavizado via núcleo estimadores, o método *Bootstrap* e o método de imputação. O autor concluiu que o melhor método para a construção dos intervalos de confiança foi o *Bootstrap*.

O Método *Bootstrap*, inicialmente proposto por Efron (1979), consiste em uma técnica de reamostragem que permite aproximar a distribuição de uma função das observações a partir da distribuição empírica dos dados baseado em uma amostra de tamanho finito. Os conceitos básicos e muitas aplicações práticas do Método *Bootstrap* podem ser encontrados em Efron & Tibshirani (1993) e Manly (1994).

Muitos autores usaram o procedimento *Bootstrap* em dados com censura intervalar. Moreira (2001) utilizou o método na escolha entre os modelos de Cox e Logístico para dados de sobrevivência com censura intervalar, concluindo que, para a análise dos dados do experimento com linho (planta da qual se extrai o óleo de linhaça), o modelo que melhor se ajustou aos dados foi o Logístico.

Jiang & Zhou (2003) propuseram um estudo de custos médicos para o tratamento de pacientes de uma clínica de cardiologia e determinaram os intervalos de confiança para esses custos utilizando o mesmo método.

Ren (2003) exemplificou o caso 2 de censura intervalar num estudo com pacientes HIV positivos, também utilizando o procedimento *Bootstrap* para encontrar o Estimador Não-Paramétrico de Máxima Verossimilhança para esses dados.

2.1 Estimador Não-Paramétrico de Máxima Verossimilhança (ENPMV)

Segundo Groeneboom & Jongbloed (2000), a função de distribuição que maximiza a verossimilhança para todas as distribuições que têm como suporte o conjunto finito de pontos $\{x_1, x_2, ..., x_n\}$ de uma dada amostra é justamente a função de distribuição empírica. Nesse sentido, a função de distribuição empírica é o Estimador Não-Paramétrico de Máxima Verossimilhança da Função de Distribuição Acumulada.

Zhang & Jamshidiam (2004) seguiram uma convenção comum em estimação não paramétrica, na qual se define o ENPMV da função de sobrevivência como uma função *step* que tem saltos somente nos pontos de observação e maximiza o logaritmo da verossimilhança. Nesse caso, o ENPMV da função de falha pode ser determinado via seus valores $z_1, z_2, ..., z_s$, que são tempos distintos de observação extraídos de observações $\underline{Y} = (Y_1, ..., Y_n)$. O ENPMV pode ser encontrado pelos saltos $p_i = F(z_i) - F(z_{i-1})$, para $i = 1, 2, ..., s$ e $z_0 = 0$, sabendo-se que $\sum_{i=1}^{s+1} p_i = 1$.

2.2 Censura intervalar

Dados envolvendo censura intervalar ocorrem com muita freqüência, principalmente em estudos clínicos e em estudos longitudinais, nos quais indivíduos são acompanhados por um período de tempo pré-estabelecido ou observados periodicamente em um número fixo de tempos.

Na censura intervalar, o exato tempo T de falha não é observado. Sabe-se apenas que o evento de interesse ocorreu entre um ponto de observação U e um outro ponto V, ou seja, num intervalo $(U,V]$, sendo que $U < T \leq V$. Censura à direita e censura à esquerda são casos particulares de censura intervalar, nos quais são assumidos os intervalos (V,∞) e $(0,U)$, respectivamente.

Segundo Allison (1995), dados com censura intervalar podem, facilmente, ser incorporados à função de verossimilhança. A contribuição de dados envolvendo esse tipo de censura para a função de verossimilhança num determinado intervalo $(U,V]$ é $S_i(U) - S_i(V)$, em que $S_i(.)$ é a função de sobrevivência para a observação *i*.

Um caso especial de censura intervalar é o chamado *current status data* (dados no estado atual) ou caso 1, no qual o indivíduo é observado apenas uma vez para o estado de ocorrência do evento de interesse num determinado tempo de observação. Já caso geral é conhecido como caso 2 de censura intervalar.

Wellner (1995) ofereceu uma revisão dos estudos para os casos 1 e 2 de censura intervalar, na qual propõe uma aplicação desse último caso num experimento de tempo de sobrevivência envolvendo pacientes com AIDS.

Tais casos de censura intervalar têm sido estudados por vários autores e são definidos a seguir:

Caso 1. Seja $(T_1,U_1),...,(T_n,U_n)$ uma amostra de variáveis aleatórias em $\Re_+^2$, em que T_i (tempo de sobrevivência) e U_i (tempo de observação) são variáveis aleatórias não dependentes e não negativas com funções de distribuição F e G, respectivamente, supondo-se somente as observações das variáveis U_i e $\delta_i = 1_A$, em que $A = \{T_i \leq U_i\}$. O logaritmo da função de verossimilhança para F é dado como segue:

$$l(F) = \log L(F) = \sum_{i=1}^{n} \{\delta_i \log F(U_i) + (1-\delta_i)\log(1-F(U_i))\}.$$

No caso 1 de censura intervalar, o indivíduo é observado apenas uma vez no experimento e, então, é registrado (U_i, δ_i), ou seja, o tempo de observação e o indicador de censura, o qual mostra a ocorrência ou não do evento de interesse.

Caso 2. Seja $(T_1,U_1,V_1),...,(T_n,U_n,V_n)$ uma amostra de variáveis aleatórias em $\Re_+^3$, em que T_i é uma variável aleatória não negativa com função de distribuição F, em que U_i e V_i são variáveis aleatórias não negativas e independentes de T_i, com função de distribuição conjunta H e tal que $U_i \leq V_i$ com probabilidade 1. Supondo-se somente a observação das variáveis (U_1,V_1), que são os tempos de observação, e $\delta_i = 1_{\{T_i \leq U_i\}}$, $\gamma_i = 1_{\{T_i \in (U_i, V_i]\}}$, o logaritmo da função de verossimilhança, nesse caso, é o seguinte:

$$l(F) = \log L(F) = \sum_{i=1}^{n} \{\delta_i \log F(U_i) + \gamma_i \log(F(V_i) - F(U_i))$$

$+(1-\delta_i-\gamma_i)\log(1-F(V_i))\}$.

Dessa forma, para cada elemento da amostra, observam-se duas variáveis, U e V, com $U < \mathrm{V}$, que são instantes de observações, e duas variáveis indicadoras δ e γ , que revelem em qual intervalo o valor da variável de interesse T está contido, lembrando-se que a variável censurada T é o instante de ocorrência de um fenômeno de interesse, como uma infecção por um vírus ou a morte de um inseto.

Se, por exemplo, $\delta_i = 0$ e $\gamma_{\mathrm{i}} = 1$, a variável de interesse T está contida no intervalo (U_i, V_i); se $\delta_i = 1$ e $\gamma_{\mathrm{i}} = 0$, a variável T está no intervalo $(0, U_i)$; e se $\delta_i = 0$ e $\gamma_{\mathrm{i}} = 0$, a variável T está contida no intervalo (V_i, ∞); sendo que os dois últimos casos representam situações particulares de censura intervalar, isto é, censura à esquerda e censura à direita, respectivamente.

2.3 O algoritmo EM (Esperança-Maximização)

O algoritmo EM é um método utilizado para computar estimativas de máxima verossimilhança e tende a ser numericamente estável. Além disso, em algumas aplicações, é de fácil implementação. Esse método é iterativo, sendo que cada iteração consiste em dois passos denominados E-*step* (Esperança) e M-*step* (Maximização).

Supondo-se θ um vetor de parâmetros e Ω um espaço paramétrico, tal que $\theta \in \Omega$ e supondo, ainda, $Z = (X, Y)$, com X denotando dados não observados e Y, dados observados, Z será denotado por um vetor de dados completos. Geralmente, deseja-se estimar θ ao maximizar o logaritmo da verossimilhança dos dados incompletos, $\ell(\theta \mid y)$ com respeito a θ, o que pode

ser muito difícil de se fazer diretamente. No entanto, já que trabalhar com o log-verossimilhança dos dados completos $\ell(\theta \mid z)$ é, em geral, mais fácil do que com $\ell(\theta \mid y)$, toda a formulação do algoritmo envolve $\ell(\theta \mid z)$.

Seja $f(x, y, \theta)$ a distribuição dos dados completos. Então:

$$\ell(\theta \mid y) = \ln f(y \mid \theta) = \ln\left(\int f(x, y \mid \theta) dx\right).$$

Pode-se verificar que:

$$\ln\left(\frac{f(x, y)}{f(x \mid y)}\right) = \ln\left(\frac{f(x, y) f(y)}{f(x, y)}\right) = \ln f(y).$$

Logo,

$$\ln\left(\frac{f(y, x \mid \theta)}{f(x \mid y, \theta)}\right) = \ln f(y \mid \theta) = \ln\left(\int f(x, y \mid \theta) dx\right).$$

Portanto,

$$\ell(\theta \mid y) = \ln f(y, x \mid \theta) - \ln f(x \mid y, \theta) \ . \qquad (2.3.1)$$

Além disso, para um valor hipotético $\theta^{(0)}$ e y fixo, tem-se:

$$\ell(\theta \mid y) = \int \ell(\theta \mid y) f\left(x \mid y, \theta^{(0)}\right) dx = E_{\left(X \mid Y, \theta^{(0)}\right)}\left[\ell(\theta \mid y)\right]. \qquad (2.3.2)$$

Assim, de $(2.3.1)$, tem-se:

$$\ell(\theta \mid y) = E_{\left(X \mid Y, \theta^{(0)}\right)}\left[\ln f(y, x \mid \theta)\right] - E_{\left(X \mid Y, \theta^{(0)}\right)}\left[\ln f(x \mid y, \theta)\right].$$

O objetivo, nesse caso, é maximizar o logaritmo da verossimilhança dos dados incompletos, $\ell(\theta \mid y)$. No entanto, por simplicidade, é melhor trabalhar com $\ell(\theta \mid z) = \ln f(x, y \mid \theta)$.

Como os dados faltantes x são desconhecidos, é preciso eliminá-los antes de maximizar $\ell(\theta \mid y)$, tomando-se seus valores esperados com respeito à $\ell(x \mid y, \theta)$, como na equação $(2.3.2)$.

Denotando $Q\left(\theta, \theta^{(0)}\right) = E_{\left(X \mid Y, \theta^{(0)}\right)}\left[\ln f(y, x \mid \theta)\right]$ e

$$H\left(\theta, \theta^{(0)}\right) = E_{\left(X \mid Y, \theta^{(0)}\right)}\left[\ln f(x \mid y, \theta)\right],$$

tem-se $\ell(\theta \mid y) = Q\left(\theta, \theta^{(0)}\right) - H\left(\theta, \theta^{(0)}\right)$.

$(2.3.3)$

Para encontrar θ que maximiza $\ell(\theta \mid y)$, pode-se trabalhar com os dois termos de $(2.3.3)$. No entanto, como pode ser visto em Silva (2002), o algoritmo EM trabalha apenas com o primeiro termo $Q\left(\theta, \theta^{(0)}\right)$, desconsiderando o segundo.

O algoritmo EM consiste em se construir uma seqüência $\left\{\theta^{(k)}\right\}$, de modo que $Q\left(\theta^{(k+1)}, \theta^{(k)}\right) = \max_{\theta \in \Omega} Q\left(\theta, \theta^{(k)}\right)$. Dempster et al. (1977) mostraram que a seqüência $\left\{\theta^{(k)}\right\}$ converge para o ponto crítico de $\ell(\theta \mid y)$.

Ao se considerar Y como sendo os dados observados e X, os dados faltantes, o parâmetro θ estimado pelo algoritmo EM é definido a partir dos seguintes passos:

E-*step*: avaliação de $Q\left(\theta,\theta^{(k)}\right)=E_{\left(X|Y,\theta^{(0)}\right)}\left[\ln f\left(x\,|\,y,\theta\right)\right]$;

M-*step*: avaliação de $\theta^{(k+1)}$, o valor de θ que maximiza $Q\left(\theta,\theta^{(k)}\right)$.

A utilização do algoritmo EM para dados com censura intervalar caso 1 pode ser encontrada em Groeneboom & Jongbloed (2000), os quais detalharam o algoritmo e provaram, mostrando cada passo do algoritimo, que o ENPMV da função de distribuição é realmente encontrado.

Wellner & Zhan (1997) apresentaram um breve estudo sobre o uso do algoritmo EM para caso 2 de censura intervalar.

Considerando-se $F\in D$ uma função de distribuição, na qual D é um conjunto de funções de distribuições discretas, que é constante nos pontos W_j, $j=1,2,...,s$, a função $F\in D$ pode ser identificada pelo vetor $p=\left(p_1,p_2,...,p_{s+1}\right)^T$, em que $p_j=F\left(W_j\right)-F\left(W_{j-1}\right)$ são saltos de F em W_j para $j=1,2,...,s$ e $\sum_{i=1}^{s+1}p_j=1$.

Sendo $\mathrm{p}^{(0)}$ uma estimativa inicial de F e sendo $\mathrm{p}^{(\mathrm{m})}$ a estimativa corrente de F após m passos do algoritmo, a verossimilhança dos dados completos será:

$$\ell\left(\mathrm{p},\mathrm{X}_1,\ldots,\mathrm{X}_\mathrm{n}\right)=\log\prod_{\mathrm{i}=1}^{\mathrm{n}}\mathrm{f}\left(\mathrm{X}_\mathrm{i}\,|\,\mathrm{p}\right)=\sum_{\mathrm{j}=1}^{\mathrm{s}+1}\log\mathrm{p}_\mathrm{i}\,\#\left\{\mathrm{X}_\mathrm{i}=\mathrm{W}_{(\mathrm{j})}\right\}.$$

No E-*step* do algoritmo EM, calcula-se a esperança condicional dos dados completos por meio da expressão

$$Q\left(p \mid p^{(m)}\right) = \sum_{j=1}^{s+1}\left(\log p_j\right)\sum_{i=1}^{n} P_{F^{(m)}}\left\{X_i = W_{(j)} \mid Y_i\right\}$$

maximizada pelo M-*step* do algoritmo EM na qual $P_{F^{(m)}}$ denota a distribuição de probabilidade condicional de X dado Y quando $X \sim F^{(m)}$.

Cabe ressaltar que, até então, todos os métodos desenvolvidos pelos diversos autores mencionados anteriormente correspondem a casos em que a escolha dos intervalos de observação é aleatória e não fixada pelo pesquisador, como é o caso dos dados entomológicos.

Uma alternativa de solução para o problema no qual os intervalos são fixados *a priori*, como no caso dos dados de Entomologia, será apresentada na seção 3.2.

2.4 O Método *Bootstrap*

O *Bootstrap* é um método que pode ser implementado tanto de forma não-paramétrica quanto de forma paramétrica.

No caso não-paramétrico, a amostragem é feita com reposição da amostra original. Nesse caso, supõe-se que as observações são obtidas da função de distribuição empírica, $\hat{F}$, que designa uma massa de probabilidade igual a $1/n$ para cada ponto amostral. Já no caso paramétrico, a amostragem é feita a partir da distribuição ajustada às observações amostrais.

Considerando-se $Y_1, Y_2, ..., Y_n$ uma amostra aleatória de tamanho n com função de distribuição acumulada desconhecida, que depende de um parâmetro θ; e, considerando-se $y = (y_1, y_2, ..., y_n)$ os valores observados, pode-se estimar a função F pela função de distribuição empírica, com base na amostra de tamanho n, que é dada por:

$$\hat{F}(y) = \frac{\#(y_j \leq y)}{n}.$$

Assim, $\hat{F}$ fornece uma probabilidade $1/n$ para cada observação amostral.

Suponha-se que se queira estimar o parâmetro θ, no qual $\theta = t(F)$ baseado em Y, em que um estimador de θ seja $\hat{\theta} = s(Y)$. Como existe a necessidade de melhorar a precisão desse estimador, retiram-se amostras com reposição, $y_1^*,...,y_n^*$, utilizando-se a função de distribuição empírica estimada $\hat{F}$. Tais amostras são denominadas amostras *Bootstrap*. Dessa maneira, para a amostra *Bootstrap* $y^* = (y_1^*,...,y_n^*)$, calcula-se $\hat{\theta}^* = s(y^*)$.

Ao se obter um grande número de replicações *Bootstrap* de $y_1,...,y_n$, consegue-se a distribuição empírica de $\hat{\theta}^*$, podendo-se calcular as medidas de posição, dispersão, intervalos de confiança, dentre outras, de modo que se possa melhorar a precisão do estimador obtido inicialmente.

Quando se têm informações suficientes sobre a função de distribuição F, o estimador *Bootstrap* $\hat{\theta}^*$ de θ pode ser alcançado da mesma maneira, apenas levando-se em consideração a distribuição que se ajusta aos dados.

3 MATERIAS E MÉTODOS

3.1 Materiais

Inicialmente, foram coletadas abelhas da espécie *Apis mellifera* dos favos do ninho em colméias do Apiário Experimental da Universidade Federal de Lavras-UFLA e transportadas em gaiolas teladas para o laboratório. Em seguida, elas foram anestesiadas com dióxido de carbono, durante 120 segundos, para que se pudessem separá-las com maior facilidade para a realização do experimento.

A seguir, individualizaram-se dez insetos de cada tratamento, em gaiola de PVC de 15 cm de altura $\times 10$ cm de diâmetro com a parte superior fechada com filó e a inferior, com organza. Essa gaiola, mantida em sala climatizada a $28 \pm 2°C$, UR 70 ± 10 % e fotofase de 12 h, continha um recipiente de vidro com capacidade para 20 ml de cada um dos tratamentos.

Os recipientes possuíam tampa perfurada, por onde foi inserido um chumaço de algodão umedecido na solução para a alimentação das abelhas. Além disso, foi oferecido às abelhas um chumaço de algodão embebido em água destilada, contendo:

a) alimento 1: solução aquosa de mel a 50%;

b) alimento 2: solução aquosa de frutose a 50%.

Foram realizadas dez repetições para cada tipo de alimento e os tempos de observações eram fixos e não aleatórios, sendo avaliada a mortalidade dos insetos durante 22 tempos a cada 12 h, ou seja, 22 intervalos de tempo.

Durante esse período, foi observado e registrado o intervalo no qual ocorreu o evento; nesse caso, a morte do inseto (vide Anexo A).

Para o estudo de simulação foram geradas amostras de uma Distribuição Weibull para o tempo T até a ocorrência do evento, com as mesmas médias e

variâncias encontradas nos dados reais, a partir dos quais foram observados apenas os intervalos em que os eventos ocorreram.

Os dados experimentais apresentaram, para o tempo de sobrevivência, as seguintes médias e variâncias:

a) alimento 1: $\overline{X} = 144$ h e $\sigma_X^2 = 3312$ h^2;

b) alimento 2: $\overline{X} = 99$ h e $\sigma_X^2 = 2088$ h^2.

Os parâmetros presentes na Distribuição Weibull foram estimados por meio das seguintes relações:

$$E[X] = a^{-1/b}\Gamma\left(1 + b^{-1}\right) \qquad (3.1.1)$$

e

$$Var[X] = a^{-2/b}\left[\Gamma\left(1 + 2b^{-1}\right) - \Gamma^2\left(1 + b^{-1}\right)\right], \qquad (3.1.2)$$

nas quais a e b são parâmetros de forma e de escala, respectivamente.

Para cada tipo de alimento oferecido às abelhas, foram geradas, no software R® (R, 2005), amostras de tamanho 1000, provenientes das seguintes distribuições:

a) alimento 1: $T \sim Weibull(2,69;161,93)$;

b) alimento 2: $T \sim Weibull(2,29;111,75)$.

As funções de sobrevivência foram estimadas utilizando-se as rotinas em linguagem C, que podem ser encontradas no Anexo B.

3.2 Métodos

Para determinar o ENPMV da função de sobrevivência, inicialmente tentou-se utilizar o algoritmo EM proposto por Zhang e Jamshidiam (2004). Tal método é descrito a seguir.

Dado que o tempo real, T, até a morte de uma abelha não é observado, é possível saber que T está contido no intervalo (L,U). Para *n* abelhas, os intervalos (L,U) podem ser representados por n observações independentes $\mathrm{Y}=(\mathrm{Y}_1,Y_2,...,Y_n)$.

Para F, função de distribuição de T, e $T(T_1,T_2,...,T_n)$, uma amostra aleatória de T, a probabilidade de que o indivíduo *i* falhe no intervalo (L_i,U_i) será:

$$\mathrm{P}(\mathrm{Y}_i \mid F) = P(L_i < T_i < U_i) = P(T \le U_i) - P(T \le L_i) = F(U_i) - F(L_i).$$

Generalizando para o vetor Y, tem-se:

$$\begin{aligned}\mathrm{P}(\mathrm{Y}|\mathrm{F}) &= P(L_1 < T_1 < U_1) \times P(L_2 < T_2 < U_2) \times \ldots \times P(L_n < T_n < U_n) \\ &= \left[F(U_1) - F(L_1)\right] \times \left[F(U_2) - F(L_2)\right] \times \ldots \times \left[F(U_n) - F(L_n)\right] \\ &= \prod_{i=1}^{n} \left[F(U_i) - F(L_i)\right].\end{aligned}$$

A função F é desconhecida e, como F é função de parâmetros, é, portanto, também um parâmetro. Assim, é possível construir a seguinte função de verossimilhança:

$$L(F \mid Y) = \prod_{i=1}^{n} [F(U_i) - F(L_i)],$$

com a função suporte:

$$l\left(F \mid Y\right) = \sum_{i=1}^{n} \log[F(U_i) - F(L_i)].$$

Sejam $z_1, z_2, \ldots, z_s$ os "s" limites superiores distintos dos intervalos pertencentes a Y, e seja $K_i = \left(L_i, U_i\right)$ e $\delta_{ij} = \begin{cases} 1 & \mathrm{z_j} \in K_i \\ 0 & \mathrm{z_j} \notin K_i \end{cases}$, para $i = 1, 2, ..., n$ e $j = 1, 2, ..., s$, é possível expressar o parâmetro F em termos dos saltos (*steps*) $P_j = F_j - F_{j-1}$, sendo que cada salto é realizado em cada valor z_j. Ou seja, com os "s" valores de z_j, têm-se "s" saltos, como ilustrado na Figura 1.

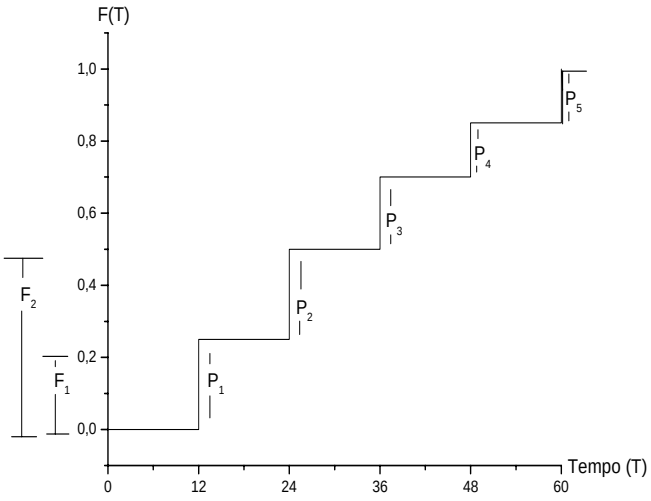

FIGURA 1 – Ilustração da função de distribuição acumulada em termos dos saltos $P_j's$

Assim, o logaritmo da função de verossimilhança, em termos de saltos, pode ser escrito como:

$$\ell\left(P \mid Y\right) = \sum_{i=1}^{n} \log\left(\sum_{j=1}^{s} P_j \delta_{ij} \right), \qquad (3.2.1)$$

em que o termo $\sum_{i=1}^{n} \log\left(\sum_{j=1}^{s} P_j \delta_{ij} \right)$ serve como um indicador dos saltos $P_j's$ que serão utilizados na maximização da função de verossimilhança.

O algoritmo EM (Esperança-Maximização), descrito a seguir, foi proposto por Zhang & Jamshidiam (2004) para determinar os saltos $P_j's$ que maximizam a expressão $(3.2.1)$:

$$\tilde{P}_j = \frac{P_j}{n} \sum_{i=1}^{n} \left(\frac{\delta_{ij}}{\sum_{j=1}^{s} P_j \delta_{ij}} \right), \qquad j = 1, 2, ..., s, \qquad (3.2.2)$$

em que $\tilde{P}_j$ é a estimativa EM, P_j é o salto na j-ésima observação, δ_{ij} é o indicador de censura intervalar e n é o tamanho da amostra.

Cabe esclarecer que o algoritmo $(3.2.2)$, tomado na sua forma original, (Zhang & Jamshidiam, 2004) apresenta valores de j que variam de 1 até $s+1$, isso devido ao fato de que se deve satisfazer a condição $\sum P_j = 1$. Ou seja, se com $"s"$ saltos não se atinge o máximo 1, cria-se mais um salto $P_{s+1} = 1 - \sum_{j=1}^{s} P_j$. Contudo, para os dados experimentais referentes ao tempo de vida de abelhas, usados neste trabalho, essa condição sempre é satisfeita com $"s"$ valores de P_j, então j sempre variará de 1 até $"s"$.

Para esse tipo de dados, nos quais se observam tempos em intervalos fixos e não aleatórios, é mostrado a seguir que não há diferença nos ENPMV dos *steps* quando se aplica o algoritmo $(3.2.2)$ ou quando se determinam tais

estimativas mediante contagens. Para os dados experimentais relativos ao tempo de vida das abelhas da espécie *Apis mellífera* (vide Anexo A), tomando-se apenas a primeira repetição do alimento mel, as estimativas dos $P_j's$ via contagem foram:

$$P_1 = \frac{1}{10},\ P_2 = \frac{2}{10},\ P_3 = \frac{1}{10},\ P_4 = \frac{3}{10},\ P_5 = \frac{1}{10},\ P_6 = \frac{1}{10},\ P_7 = \frac{1}{10}.$$

Esses valores podem ser visualizados na Figura 2, que fornece um esquema do tempo até a morte de abelhas alimentadas com mel (1ª repetição), sendo que os pontos $(\bullet)$ representam o momento da ocorrência do evento.

Exemplificando: na Figura 2, pode-se observar que uma abelha morreu no intervalo [48, 60] horas. Como eram 10 abelhas, a probabilidade de morte nesse intervalo era $P_1 = \frac{1}{10}$.

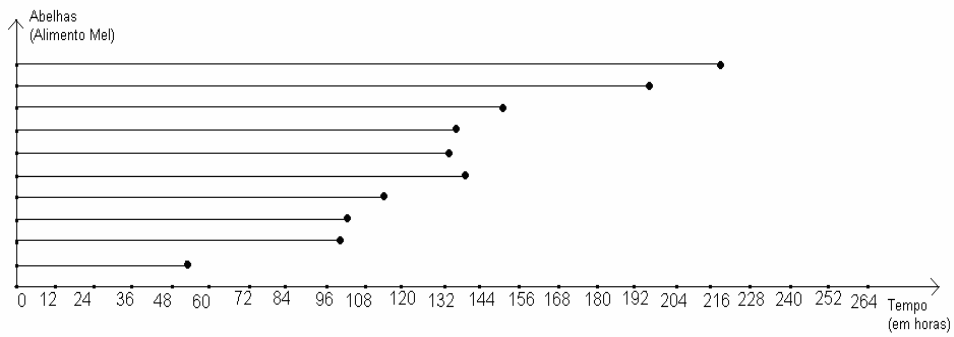

FIGURA 2 - Esquematização dos tempos até a morte das 10 abelhas

Na Figura 3, foram tomados $z_1, z_2, \ldots, z_s$ como os limites superiores dos intervalos nos quais ocorreram eventos (morte de abelhas); os valores δ_{ij} são

iguais a 1, quando $z_j \in K_i$ e $K_i = (L_i, U_i)$, para $i = 1,2,\ldots,n$ e $j = 1,2,\ldots,s$.

A partir da Figura 3, é possível determinar os valores de δ_{ij} ($i = 1,2,\ldots,10$ e $j = 1,2,\ldots,7$). Tais valores serão apresentados na Tabela 1, mais adiante.

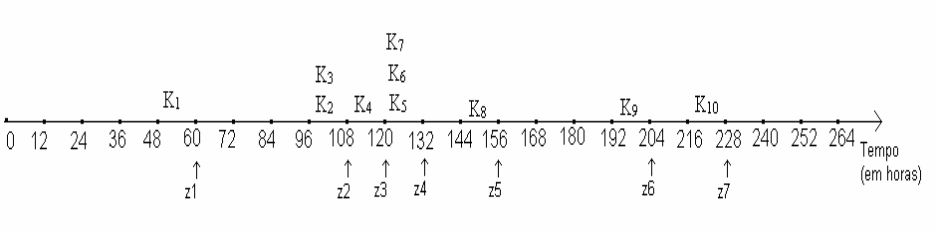

FIGURA 3 - Esquematização dos valores z_j's nos seus respectivos intervalos para obtenção dos valores δ_{ij}

Assim, a função suporte determinada segundo a equação $(3.2.1)$ será:

$$\ell(P|Y) = \log(P_1) + \log(P_2) + \log(P_2) + \log(P_3) + \log(P_4)$$

$$+ \log(P_4) + \log(P_4) + \log(P_5) + \log(P_6) + \log(P_7)$$

$$= \log P_1 + 2\log P_2 + \log P_3 + 3\log P_4 + \log P_5 + \log P_6 + \log P_7.$$

Tabela 1: Valores de δ_{ij} referentes à primeira repetição do alimento mel

$\delta_{11}=1$	$\delta_{12}=0$	$\delta_{13}=0$	$\delta_{14}=0$	$\delta_{15}=0$	$\delta_{16}=0$	$\delta_{16}=0$
$\delta_{21}=0$	$\delta_{22}=1$	$\delta_{23}=0$	$\delta_{24}=0$	$\delta_{25}=0$	$\delta_{26}=0$	$\delta_{27}=0$
$\delta_{31}=0$	$\delta_{32}=1$	$\delta_{33}=0$	$\delta_{34}=0$	$\delta_{35}=0$	$\delta_{36}=0$	$\delta_{37}=0$
$\delta_{41}=0$	$\delta_{42}=0$	$\delta_{43}=1$	$\delta_{44}=0$	$\delta_{45}=0$	$\delta_{46}=0$	$\delta_{47}=0$
$\delta_{51}=0$	$\delta_{52}=0$	$\delta_{53}=0$	$\delta_{54}=1$	$\delta_{55}=0$	$\delta_{56}=0$	$\delta_{57}=0$
$\delta_{61}=0$	$\delta_{62}=0$	$\delta_{63}=0$	$\delta_{64}=1$	$\delta_{65}=0$	$\delta_{66}=0$	$\delta_{67}=0$
$\delta_{71}=0$	$\delta_{72}=0$	$\delta_{73}=0$	$\delta_{74}=1$	$\delta_{75}=0$	$\delta_{76}=0$	$\delta_{77}=0$
$\delta_{81}=0$	$\delta_{82}=0$	$\delta_{83}=0$	$\delta_{84}=0$	$\delta_{85}=1$	$\delta_{86}=0$	$\delta_{87}=0$
$\delta_{91}=0$	$\delta_{92}=0$	$\delta_{93}=0$	$\delta_{94}=0$	$\delta_{95}=0$	$\delta_{96}=1$	$\delta_{97}=0$
$\delta_{101}=0$	$\delta_{102}=0$	$\delta_{103}=0$	$\delta_{104}=0$	$\delta_{105}=0$	$\delta_{106}=0$	$\delta_{107}=1$

Com os δ_{ij}, os valores P_j que maximizam a função suporte, determinados segundo o algoritmo EM (expressão 3.2.2), serão:

$$\hat{P}_j = \frac{1}{10}\left[\left(\frac{\delta_{1j}}{P_1}\right)P_j + \left(\frac{\delta_{2j}}{P_2}\right)P_j + \left(\frac{\delta_{3j}}{P_2}\right)P_j + \left(\frac{\delta_{4j}}{P_3}\right)P_j + \left(\frac{\delta_{5j}}{P_4}\right)P_j + \right.$$

$$\left. + \left(\frac{\delta_{6j}}{P_4}\right)P_j + \left(\frac{\delta_{7j}}{P_4}\right)P_j + \left(\frac{\delta_{8j}}{P_5}\right)P_j + \left(\frac{\delta_{9j}}{P_6}\right)P_j + \left(\frac{\delta_{10j}}{P_7}\right)P_j \right]$$

$$= \frac{P_j}{10}\left[\left(\frac{\delta_{1j}}{P_1}\right) + \left(\frac{\delta_{2j}+\delta_{3j}}{P_2}\right) + \left(\frac{\delta_{4j}}{P_3}\right) + \left(\frac{\delta_{5j}+\delta_{6j}+\delta_{7j}}{P_4}\right) + \left(\frac{\delta_{8j}}{P_5}\right) + \left(\frac{\delta_{9j}}{P_6}\right) + \left(\frac{\delta_{10j}}{P_7}\right)\right].$$

Portanto:

$$\hat{P}_1 = \frac{P_1}{10}\left[\left(\frac{1}{P_1}\right)\right] = \frac{1}{10},\ \hat{P}_2 = \frac{P_2}{10}\left[\left(\frac{2}{P_2}\right)\right] = \frac{2}{10},\ \hat{P}_3 = \frac{P_3}{10}\left[\left(\frac{1}{P_3}\right)\right] = \frac{1}{10},\ \hat{P}_4 = \frac{P_4}{10}\left[\left(\frac{3}{P_4}\right)\right] = \frac{1}{10},$$

$$\hat{P}_5 = \frac{P_5}{10}\left[\left(\frac{1}{P_5}\right)\right] = \frac{1}{10},\ \hat{P}_6 = \frac{P_6}{10}\left[\left(\frac{1}{P_6}\right)\right] = \frac{1}{10},\ \hat{P}_7 = \frac{P_7}{10}\left[\left(\frac{1}{P_7}\right)\right] = \frac{1}{10},$$

os quais são iguais aos valores dos saltos P_j obtidos por contagem, ou seja, para se obter os ENPMV da curva de sobrevivência, com os dados reais relativos ao experimento com as abelhas, é suficiente estimar os saltos P_j via contagem.

Deve-se observar que todo esse procedimento foi realizado com apenas 1 repetição do alimento mel. Contudo, estimar os saltos $P_j's$ via contagem com apenas uma repetição não seria confiável. Como em pesquisas com esse tipo de dados são realizadas, geralmente, com 10 repetições para cada alimento, utilizaram-se as 10 repetições para se fazerem as contagens (1 contagem para cada repetição).

Assim, com as 10 repetições, obtiveram-se 10 ENPMV via contagem para cada salto P_j. As 10 ENPMV foram consideradas como uma amostra a partir da qual se aplicou o Método *Bootstrap* para melhorar as estimativas de cada P_j.

Para a realização do procedimento *Bootstrap*, foram avaliadas as dez repetições de cada alimento. As amostras iniciais, para os alimentos mel e frutose, podem ser encontradas nos Anexos C e D, respectivamente, e referem-se às probabilidades de ocorrência de eventos em cada intervalo. Os intervalos foram numerados de 1 a 22, obtendo-se, assim, 22 conjuntos de dados contendo 10 valores cada.

Para cada conjunto que possuía pelo menos um valor diferente de zero foram retiradas 1000 amostras com reposição para se estimarem as curvas de sobrevivência por meio de saltos.

A seguir, foi estimada a média de cada uma dessas 1000 amostras, que foi denotada por $\overline{p}_{jb}$. Com os $\overline{p}_{jb}$, foi determinada a média *Bootstrap* dada pela expressão: $\overline{\overline{p}}_j = \frac{\sum_{b=1}^{1000} \overline{p}_{jb}}{1000}$.

Foi determinado, também, o desvio padrão $s_{\overline{\overline{p}}_j} = \sqrt{\frac{\sum_{b=1}^{1000} \left(\overline{p}_{jb} - \overline{\overline{p}}_j \right)}{1000-1}}$, e, em seguida, o intervalo de confiança *Bootstrap* com nível de 95%. Tal intervalo é dado por:

$$\left\{ \overline{\overline{p}}_j - s_{\overline{\overline{p}}_j} . z_{\alpha/2},\ \overline{\overline{p}}_j + s_{\overline{\overline{p}}_j} . z_{\alpha/2} \right\}.$$

Com os saltos (*Bootstrap*) $\overline{\overline{P}}_j$, foi possível construir as curvas de sobrevivência para cada tipo de alimento (mel e frutose) e seus respectivos intervalos de confiança.

4 RESULTADOS E DISCUSSÃO

4.1 Simulação via algoritmo EM (ou via contagem)

Na Figura 4, a seguir, está representada, em vermelho, uma curva de sobrevivência construída a partir de uma simulação feita com 1000 dados originados de uma Distribuição Weibull, com parâmetros de forma e escala iguais a 2,69 e 161,93, respectivamente.

Da mesma forma, na Figura 5, a seguir, está representada uma curva de sobrevivência construída a partir de uma simulação feita com 1000 dados originados de uma Distribuição Weibull, com parâmetros de forma e escala iguais a 2,29 e 111,75, respectivamente.

Os traços em preto, nessas figuras, correspondem às curvas de sobrevivência estimada via algoritmo EM (ou, equivalentemente, via contagem), como foi explicado na seção 3.2.

Os parâmetros de forma e escala foram tomados de modo que se obtivessem dados de uma distribuição Weibull que se aproximassem dos dados experimentais utilizados neste trabalho. Para tanto, foram tomadas as médias e variâncias dos dados reais referentes aos alimentos mel e frutose e, por meio das relações $(3.1.1)$ e $(3.1.2)$, foram estimados os parâmetros utilizados.

Pode-se observar que as curvas de sobrevivência estimadas se sobrepõem às curvas simuladas. Assim, verifica-se que as curvas de sobrevivência podem ser construídas a partir de saltos (probabilidades) estimados via contagens.

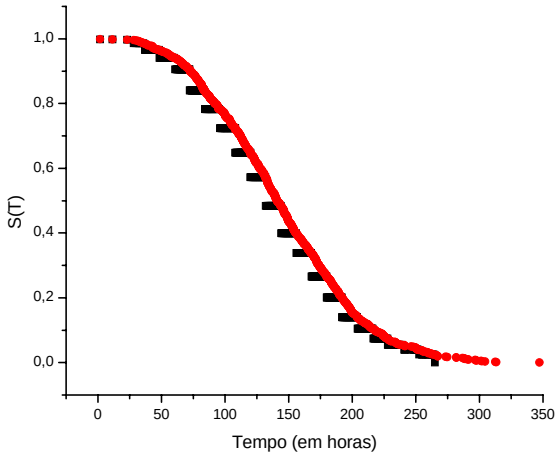

FIGURA 4 – Curva de sobrevivência, em vermelho, simulada a partir de uma Distribuição Weibull com parâmetros 2,69 (forma) e 161,93 (escala) e, em preto, estimada via algoritmo EM (ou, equivalentemente, via contagem)

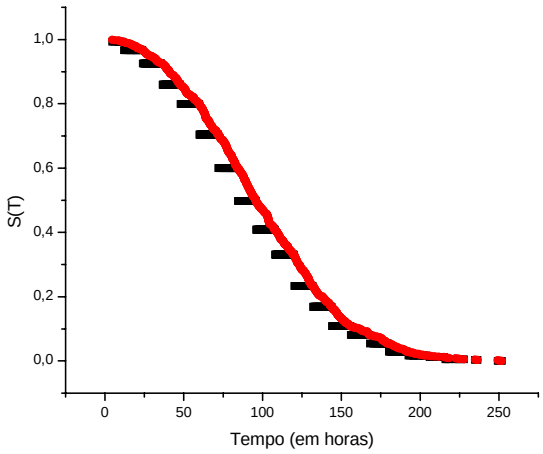

FIGURA 5 - Curva de sobrevivência, em vermelho, simulada a partir de uma Distribuição Weibull com parâmetros 2,29 (forma) e 111,75 (escala) e, em preto, estimada via algoritmo EM (ou, equivalentemente, via contagem)

4.2 Estimativa *Bootstrap* das curvas de sobrevivência

Os resultados obtidos a partir do Método *Bootstrap* (detalhado na seção 3.2) estão representados nas Figuras 6 e 7.

Na Figura 6, estão representadas as curvas de sobrevivência estimadas dos alimentos mel (solução aquosa a 50%) e frutose (solução aquosa a 50%). Pode-se observar, nessa figura, que a curva de sobrevivência do alimento mel (em preto) fica acima da curva de sobrevivência do alimento frutose (em vermelho), o que significa que as abelhas alimentadas com mel têm probabilidade maior de sobreviver a um determinado tempo "t" que as abelhas alimentadas com frutose. Isso será verificado, a seguir, na seção 4.3.

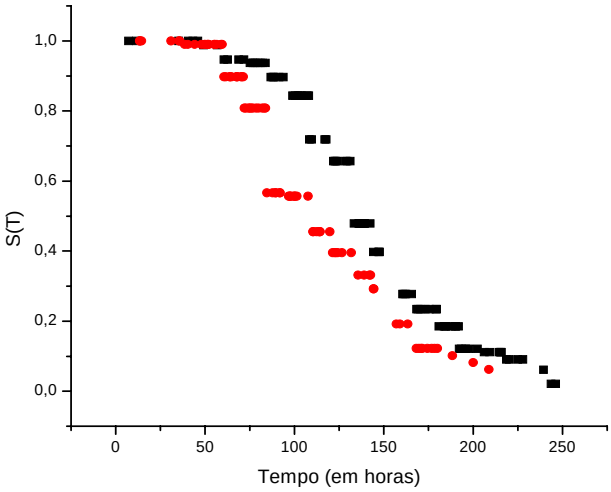

Figura 6 – Estimativa *Bootstrap* das curvas de sobrevivência para os alimentos mel (preto) e frutose (vermelho)

4.3 Comparação de curvas de sobrevivência *Bootstrap*

Na Figura 7, são apresentados os intervalos de confiança *Bootstrap* para as curvas de sobrevivência estimadas que foram apresentadas na Figura 6, como foi detalhado na seção 3.2.

Pode-se observar, na Figura 7, que os intervalos de confiança dos alimentos não se cruzam, indicando, assim, haver diferença significativa entre as duas curvas de sobrevivência.

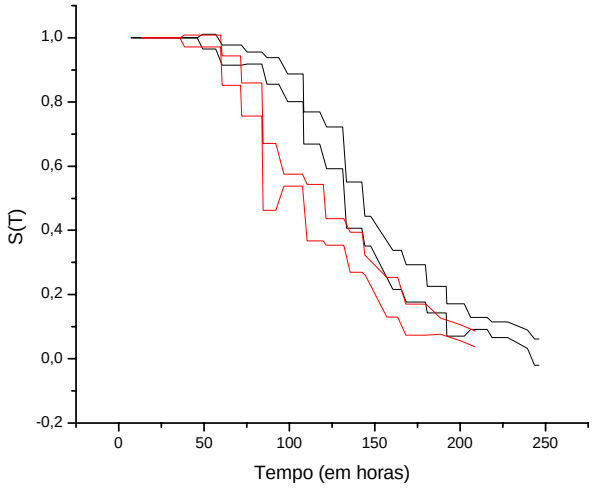

FIGURA 7 - Intervalo de confiança *Bootstrap* a 95% para as curvas de sobrevivência estimadas dos alimentos mel e frutose

Como foi verificada diferença significativa entre as duas curvas de sobrevivência, a curva do alimento mel (em preto) pode ser escolhida como aquela que prolonga mais o tempo de vida das abelhas, uma vez que está situada acima da curva do alimento frutose (em vermelho).

5 CONCLUSÕES

Para o tipo de dados usados neste trabalho (dados entomológicos), verificou-se que não há diferença no Estimador Não-Paramétrico de Máxima Verossimilhança (ENPMV) dos saltos, quando se aplica o algoritmo EM (Esperança-Maximização) ou quando se determinam tais estimativas por meio de contagens.

Embora existam vários métodos para tratar censura intervalar, pode-se concluir que, exclusivamente para o caso dos dados entomológicos tratados neste trabalho, o Método *Bootstrap* é uma boa alternativa para se estimarem curvas de sobrevivência.

Ao se compararem os dois tipos de alimentos artificiais (solução aquosa de mel a 50% e solução aquosa de frutose a 50%), verificou-se que o alimento mais indicado para suprir a alimentação das abelhas da espécie *Apis mellifera* durante a entressafra (escassez de néctar e pólen), em substituição a uma apicultura migratória, foi a solução aquosa de mel. Considerando-se apenas a longevidade dos insetos, pode-se afirmar, em termos probabilísticos, que esse alimento oferece um maior tempo de vida para as abelhas.

Enfim, a fácil construção de intervalos de confiança a partir do Método *Bootstrap* permite comparar curvas de sobrevivência para dados que envolvam censura intervalar sem a necessidade de se criar um método específico de comparação.

6 TRABALHOS FUTUROS

O desenvolvimento deste estudo, sua metodologia e resultados indicam a possibilidade de investimento em trabalhos futuros, conforme se recomenda a seguir:

1) tratar dados que envolvem censura intervalar, como os que foram utilizados neste trabalho, incluindo, também, censura à direita;

2) desenvolver um *software* para analisar dados com censura intervalar: os entomologistas forneceriam os dados e o *software* construiria as curvas de sobrevivência a serem comparadas;

3) analisar o comportamento do Método *Bootstrap* para amostras de tamanhos menores que 10 (Deve-se lembrar que, neste trabalho, as amostras são baseadas nos valores $P_j's$, ou seja, trabalhando-se com $n = 10$ indivíduos, ter-se-iam 10 $P_j's$; com $n = 5$ indivíduos, ter-se-iam 5 $P_j's$).

7 REFERÊNCIAS BIBLIOGRÁFICAS

ALLISON, P. D. **Survival analysis using the SAS® sistem:** a pratical guide. Cary, NC: SAS Institute, 1995.

BRIGHENTI, D. M. **Bioatividade do Dipel Bacillus thuringiensis var. kurstaki para Galleria mellonella (Lepidoptera: Pyralidae) e adultos de Apis mellifera Linnaeus, (Hymenoptera: Apidae).** 2003. Dissertação (Mestrado) - Universidade Federal de Lavras, Lavras, MG.

COX, D. R.; OAKS, D. **Analysis of survival data.** New York: Chapman and Hall, 1984. 201 p.

DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B. "Maximum likelihood from incomplete data via the EM algorithm", **Journal of the Royal Statistical Society, Series B,** Oxford, v. 39, n. 1, p. 1-38, 1977.

EFRON, B. Bootstrap methods: another look at the jackknife. **Annals Statistics,** Hayward, v. 7, n. 1, p. 1-26, 1979.

EFRON, B.; TIBISHIRANI, R. J. **An Introduction to the Bootstrap**. New York: Chapman & Hall, 1993. 436 p.

FANG, H. B.; SUN, J.; LEE, M. L. T. Nonparametric Survival Comparisons for Interval-Censored Continuous Data. **Statistica Sinica,** Taipei, v. 12, n. 4, p. 1073-1083, Oct. 2002.

GENTLEMAN, R.; VANDAL A. C. Computational Algorithms for Censored Data Problems Using Intersection Graphs, **Journal of Computational & Graphical Statistics,** Alexandria, v. 10, n. 3, p. 403-421, 2001.

GESKUS, R.; GROENEBOOM, P. Asymptotically Optimal Estimation of Smooth Functionals for Interval Censoring, Case 2. **The Annals of Statistical**, Haywardmd, v. 27, n. 2, p. 627-674, Apr. 1999.

GIOLO, S. R. **Turnbull's Nonparametric Estimator for Interval-Censored Data:** Technical Reports - Department of Statistics. Universidade Federal do Paraná, 2004. Disponível em: <http://www.est.ufpr.br/rt/suely04a.htm>. Acesso em: 2006.

GROENEBOOM, P.; JONGBLOED, G. Statistical methods for incomplete or indirectly observable data. **Notes for Stieltjes week on Statistics,** 2000.

JAMSHIDIAN, M.; JENNRICH, R. J. Acceleration of the EM algorithm by using Quasi-Newton Methods. **Journal of the Royal Statistical Society, Serie B,** Oxford, v. 59, n. 3, p. 569-587, 1997.

JAMSHIDIAN, M.; JENNRICH, R. J. Estandard Erros for EM Estimation. **Journal of the Royal Statistical Society, Serie B**, Oxford, v. 62, n. 2, p. 257-270, 2000.

JIANG, H.; ZHOU, X. Bootstrap Confidence Intervals for Medical Costs With Censored Observations. **Statistics in Medicine,** Sussex, v. 18, p. 3365-3376, 2004.

JONGBLOED, G. The Iterative Convex Minorant Algorithm for Nonparametric Estimation. Submitted to **Journal of Computational and Graphical Statistics**, Amsterdam, 1v. 7, n. 3, p. 310,-321, Sept. 1998.

HUANG, J.; WELLNER, J. A. Interval censored data: a review of recent progress. In: SEATTLE SYMPOSIUM IN BIOSTATISTICS, 1., 1996, New York. **Proceedings...** New York: Springer-Verlag, 1996. p. 123-169.

HUDGENS, M G. On nonparametric maximum likelihood estimation with interval censoring and left truncation. **Journal of the Royal Statistical Society Serie B**, Oxford, v. 67, Part 4, p. 573-587, Sept. 2005.

KLEIN, J. P.; MOESCHBERGER, M. **Survival analysis**. New York: Springer Verlag, 1997.

LINDSEY, J. C.; RYAN, L. M. Tutorial in biostatistics: methods for interval-censored data. **Statistics in Medicine**, Sussex, v. 17, n. 2, p. 219-238, June 1998.

MANLY, B. F. J. **Randomization, Bootstrap and Monte Carlo Methods in Biology.** 2. ed. New York: Chapman & Hall, 1994.

MENG, X. L.; RUBIN, D. B. Using EM to Obtain Asymptotic Variance-Covariance Matrices. **Journal of the American Statistical Association**, Alexandria, v. 86, n. 2. p. 899-909, June 1991.

MOREIRA, J. A. **Utilização do Método Bootstrap na escolha entre os Modelos de Cox e Logístico para Dados de Sobrevivência com Censura Intervalar.** 2001. 65 p. Tese (Doutorado) – Escola Superior de Agricultura Luiz de Queiroz, Piracicaba -

NG, M. P. Consultant's forum a Modification of Peto's Nonparametric Estimation of Survival Curves for Interval-Censored Data. **Biometrics,** Washington, v. 58, n. 2, p. 439-442, June 2002.

PAN, W. Smooth Estimation of the Survival Function for Interval Censored Data **Statistcs in Medicine,** Sussex, v. 19, n. 19, p. 2611-2624, Oct. 2000.

PEPE, M. S.; FLEMING, T. R. Weighted Kaplan-Meier statistics: A class of distance tests for censored survival data. **Biometrics,** Washington, v. 45, n. 2, 497-507, June 1989.

PETO, R. Experimental Survival Curves for Interval Censored Data. **Applied Statistics**, London, v. 22**,** n. 1, p. 86-91, 1973.

PETRONI, G. R.; WOLFE, R. A. A Two-sample Test for Stochastic Ordering with Interval Censored Data. **Biometrics**, Washington, v. 50, n. 1, p. 7-87, 1994.

R Development Core Team. **R: A language and environment for statistical computing.** Vienna, Austria: R Foundation for Statistical Computing, 2005. Disponível em: <http://www.R-project.org>. Acesso em: 2005.

REN, J. J. Goodness of Fit Tests with Interval Censored Data. **Scandinavian Journal of Statistics,** Oxford, v. 30, n. 1, p. 211-226, Mar. 2003

RODARTE, E. L. **Intervalos de confiança para a Função de Distribuição na presença de Censura Intervalar, Caso 1.** 2005. Dissertação (Mestrado) – Universidade Federal de Minas Gerais. Departamento de Estatística da UFMG, Belo Horizonte.

SILVA, C. Q. Tutorial sobre Modelos Markovianos com Estados Latentes. In: CONGRESSO NACIONAL DE MATEMÁTICA APLICADA, 15., 2002, Nova Friburgo, RJ. **Anais...** Nova Friburgo, RJ, 2002.

SILVERMAN, B. W. **Density estimation for statistics and data analysis.** London: Chapman and Hall, 1986.

SIMONOFF, J. S. **Smoothing methods in statistics**. Springer Series in Statistics, 1996.

TANNER, M. **A. Tools for statistical inference:** Methods for the Exploration of Posterior Distributions and Likelihood Functions. 3. ed. Springers Series in Statistics, 1996.

TURNBULL, B. W. The empirical Distribution Function with Arbitrarily Grouped, Censored and Truncated Data. **Journal of the Royal Statistical Society Serie B**, Oxford, v. 38, n. 3, p. 290-295, 1976.

WELLNER, J. A. Interval Censoring, Case 2: Alternative Hypotheses. **Technical Report** nº 289. University of Washington, USA, 1995.

WELLNER, J. A.; Zhan, Y. A Hybrid Algorithm for Computation of the NPMLE from Censored Data. **Journal of the American Statistical Association,** Alexandria, v. 92, n. 439, p. 945, Sept. 1997.

ZHANG, Y.; JAMSHIDIAN, M. On Algorihms for the Nonparametric Maximum Likelihood Estimator of the Failure Function With Censored Data. **Journal of Computational and Graphical Statistics**, Alexandria, v. 13, n. 1, p 123-140, Mar. 2004.

8 ANEXOS

ANEXO A

TABELA 1 A - Dados reais referentes ao tempo de vida de abelhas da espécie *Apis mellifera* coletados no Apiário Experimental da Universidade Federal de Lavras (UFLA)

		Nº de abelhas mortas	
Tempo observado (horas)	Repetição	Mel	Frutose
12	1	0	0
12	2	0	0
12	3	0	0
12	4	0	0
12	5	0	0
12	6	0	0
12	7	0	0
12	8	0	0
12	9	0	0
12	10	0	0
24	1	0	0
24	2	0	0
24	3	0	0
24	4	0	0
24	5	0	0
24	6	0	0
24	7	0	0
24	8	0	0
24	9	0	0
24	10	0	0
36	1	0	1
36	2	0	0
36	3	0	0
36	4	0	0
36	5	0	0
36	6	0	0
36	7	0	0
36	8	0	0
36	9	0	0
36	10	0	0
48	1	0	0
48	2	1	0
48	3	0	0
48	4	0	0
48	5	0	0
48	6	0	0

Tabela 1A - Continuação

		Nº de abelhas mortas	
Tempo observado (horas)	Repetição	Mel	Frutose
48	7	0	0
48	8	0	0
48	9	0	0
48	10	0	0
60	1	1	1
60	2	0	0
60	3	0	1
60	4	0	2
60	5	0	2
60	6	1	0
60	7	1	1
60	8	0	1
60	9	0	0
60	10	1	1
72	1	0	2
72	2	0	2
72	3	1	1
72	4	0	2
72	5	0	0
72	6	0	0
72	7	0	1
72	8	0	0
72	9	0	0
72	10	0	1
84	1	0	2
84	2	0	1
84	3	1	1
84	4	0	2
84	5	1	1
84	6	0	5
84	7	0	7
84	8	0	2
84	9	2	3
84	10	0	3
96	1	0	0
96	2	1	1
96	3	1	0
96	4	0	0
96	5	0	0
96	6	0	0
96	7	0	0
96	8	2	0
96	9	0	0

Tabela 1A - Continuação

		Nº de abelhas mortas	
Tempo observado (horas)	Repetição	Mel	Frutose
96	10	1	0
108	1	2	1
108	2	2	1
108	3	0	1
108	4	2	0
108	5	2	1
108	6	0	0
108	7	1	1
108	8	1	5
108	9	1	0
108	10	1	0
120	1	1	1
120	2	0	1
120	3	3	0
120	4	0	0
120	5	0	0
120	6	2	1
120	7	0	0
120	8	0	0
120	9	0	1
120	10	0	2
132	1	3	1
132	2	2	0
132	3	0	0
132	4	2	1
132	5	3	3
132	6	3	1
132	7	0	0
132	8	2	0
132	9	1	0
132	10	1	0
144	1	0	0
144	2	0	1
144	3	1	1
144	4	0	0
144	5	2	0
144	6	1	1
144	7	0	0
144	8	1	0
144	9	2	1
144	10	1	0
156	1	1	0
156	2	0	1

Tabela 1A - Continuação

		Nº de abelhas mortas	
Tempo observado (horas)	Repetição	Mel	Frutose
156	3	0	3
156	4	2	0
156	5	1	0
156	6	1	1
156	7	2	0
156	8	2	1
156	9	3	2
156	10	0	2
168	1	0	0
168	2	1	2
168	3	0	1
168	4	0	0
168	5	0	2
168	6	0	0
168	7	3	0
168	8	0	0
168	9	0	1
168	10	0	1
180	1	0	1
180	2	0	0
180	3	1	0
180	4	2	0
180	5	0	0
180	6	0	0
180	7	0	0
180	8	0	0
180	9	1	1
180	10	1	0
192	1	0	0
192	2	1	0
192	3	2	0
192	4	0	1
192	5	0	0
192	6	1	1
192	7	0	0
192	8	0	0
192	9	0	0
192	10	2	0
204	1	1	0
204	2	0	0
204	3	0	0
204	4	0	1
204	5	0	0

Tabela 1A - Continuação

		Nº de abelhas mortas	
Tempo observado (horas)	Repetição	Mel	Frutose
204	6	0	0
204	7	0	0
204	8	0	0
204	9	0	1
204	10	0	0
216	1	0	0
216	2	0	0
216	3	0	0
216	4	0	0
216	5	1	0
216	6	0	0
216	7	1	0
216	8	0	1
216	9	0	0
216	10	0	0
228	1	1	0
228	2	0	0
228	3	0	0
228	4	1	1
228	5	0	0
228	6	0	0
228	7	0	0
228	8	1	0
228	9	0	0
228	10	0	0
240	1	0	0
240	2	0	0
240	3	0	0
240	4	1	0
240	5	0	0
240	6	0	0
240	7	0	0
240	8	1	0
240	9	0	0
240	10	2	0
252	1	0	0
252	2	0	0
252	3	0	1
252	4	0	0
252	5	0	0
252	6	0	0
252	7	2	0
252	8	0	0

Tabela 1A - Continuação

		Nº de abelhas mortas	
Tempo observado (horas)	Repetição	Mel	Frutose
252	9	0	0
252	10	0	0
264	1	0	0
264	2	0	0
264	3	0	0
264	4	0	0
264	5	0	0
264	6	0	0
264	7	0	0
264	8	0	0
264	9	0	0
264	10	0	0

ANEXO B

Rotina em linguagem “C” para se obter as estimativas *Bootstrap* das funções de sobrevivência

```
#include <stdio.h>
#include <math.h>
#include <string.h>

#include "RAN2.C"
#include "RAN3.C"


/* -------------------------- Definições de algumas variáveis ---------------- */

#define pi         3.141592653588
#define sqr(x)    ((x)*(x))
#define cub(x)   ((x)*(x)*(x))

#define N_BOOTSTRAP 1000 /* no. dados gerados no Bootstrap */
#define N_AMOS 100  /* no. de amostras */
#define LIM_T  264  /* limite tempo falha */
#define T_INTER 12  /* tamanho do intervalo */
#define N_INTER (LIM_T/T_INTER)  /* no. de intervalos */
#define N1 10
#define N_DADOS_REAL 10  /* no. dados reais */

#define N_PASSOS_MAX 100  /* no. de passos iteração */
#define N_start_aq 50 /* no. de aquecimento */

#define NPOW  N /*  N**N */

/* ------------------------------------------------------------------- */

/*==========definição das variáveis do programa =============*/


        long int idum;
        int i,j,j_aux,k,k1;
        int is,i1,k1;
```

```
        int is_aux;
        int i_prog;

        int N_dados_dif;
        int id1,id2,id3,id4,id5,id6,id7;
        int is1,is2,is3,is4,is5,is6,is7;

      long int Nt, ADELt;
        int i_aq;
        int j_loc;
        int sement_aleat;
        long int a1_ran, a1_aux_ran;
        float aux_ran;
        float data_Weib[N_AMOS+1];
        int count_block[N_INTER+1];
        int count_block_real[N_INTER+1];
        int censor_data[3+1][N_INTER+1];
        float data_real[3+1][N_DADOS_REAL+1];
        int new_block[N_INTER+1];
        int new_block_real[N_INTER+1];
        int facum[N_INTER+1];
        int delIJ[N_AMOS+1][N_INTER+2];
        int delIJ_real[N_DADOS_REAL+1][N_INTER+2];
        int INTER_GER,INTER_REAL_GER;
        float Fac_Norm[N_INTER+1][N_PASSOS_MAX+1];
        float Fac_Norm_real[N_INTER+1][N_PASSOS_MAX+1];
        float pJ[N_INTER+2][N_PASSOS_MAX+1];
        float er_pJ[N_INTER+2][N_PASSOS_MAX+1];
        float pJ_real[N_INTER+2][N_PASSOS_MAX+1];
        float er_pJ_real[N_INTER+2][N_PASSOS_MAX+1];
        float pJ_real_boot_base[N_INTER+2][N_PASSOS_MAX+2];
        float
vec_bootstrap [N_BOOTSTRAP+2][N_DADOS_REAL+2][N_INTER+2];
        float Pm_local_bootstrap[N_BOOTSTRAP+2][N_INTER+2];
        float Pm_global_bootstrap[N_INTER+2];
        float Pm_final_bootstrap[N_INTER+2];
        float Fac_final_bootstrap[N_INTER+2];
        float sigma_pM[N_INTER+2];
        float sigma_pM_novo[N_INTER+2];
        float Int_Confianca[N_INTER+2];
        float IC_novo[N_INTER+2];
        float Tz;
```

```
int janela_print;
float aux_p_alternat;

float aux_denpJ,aux_numpJ,aux_pJ;
float aux_denpJ_real,aux_numpJ_real,aux_pJ_real;
float sum_pJ,sum_pJ1;
float sum_pJ_real,sum_pJ1_real;
float sum_Pm_global_bootstrap;

int tdestat_VERT,tdestat_HORIZ;
int N1_subamo_max;
int N_del_AMOS;
int Naleat_PASSOS;
int N_passos_iter;


float p, p_teste, p_selec_det;
float P_alternat,pnovo_alternat;


int a, b;

char file_data[100];
char file_data_inX[100];
char file_data_inY[100];
char file_data_sai[100];
char file_data_sai2[100];
char file_data_sai3[100];
char file_data_sai4[100];
char file_data_sai5[100];
FILE *arquivo;
FILE *arquivo_ent;
FILE *arquivo_ent_inX;
FILE *arquivo_ent_inY;
FILE *arquivo_sai;
FILE *arquivo_sai2;
FILE *arquivo_sai3;
FILE *arquivo_sai4;
FILE *arquivo_sai5;

/*======== Fim de definicao das variaveis do programa ========== */
```

```
main ()
{

        /* Leitura de arquivo de dados da Weibull simulada */

        printf(" Entre com no. pontos VERTICAL : ");
        scanf("%d", &tdestat_VERT);

        printf(" Entre com no. pontos HORIZONTAL : ");
        scanf("%d", &tdestat_HORIZ);


        printf(" Arquivo  de Weibull (datan.dat): ");gets(file_data);
        gets(file_data);


        printf(" Arquivo  de ENTRADA pJ_real_repeticoes (datan.dat):
");gets(file_data_sai4);

        printf(" Arquivo  de ENTRADA Dados_falhas_REAIS (datan.dat):
");gets(file_data_sai2);

        printf(" Arq Fac_Weibull_EXP (datan.dat) via BOOTSTRAP:
");gets(file_data_sai3);

        printf(" Entre com a janela de impressao : Valor inteiro = ");
                scanf("%d",&janela_print);

        printf(" Entre com a semente do no. aleatorio : Valor inteiro = ");
                scanf("%d",&sement_aleat);



/********** Definição de arquivos de entrada e saída ************/

      /* arquivo_sai2=fopen(file_data_sai2,"a+"); */
                                        /* Arquivo de saída */


 /*****************************************************/
```

```
a = 3;
b = 7;

   /*########Lê dados da Weibull simulada ###### */

   arquivo_ent=fopen(file_data,"r+");
            /* Arquivo de entrada dados Weibull */

   is = 0;
   for (i=1;i<=2*tdestat_HORIZ;i++){
   for (j=1;j<=tdestat_VERT/2;j++){
            is = is + 1;
            fscanf(arquivo_ent,"%f",&data_Weib[is]);
            printf("%s %d %s %d %s %d %s %f\n",
            " i = ",i,
            " j = ",j,
            " is = ",is,
            " data_Weib = ",data_Weib[is]);
            }
   }
   fclose(arquivo_ent);


   /*###Lê dados de pJ_real_repetições DADOS EXPERIMENTAIS### */

   arquivo_sai4=fopen(file_data_sai4,"r+");
   /* Arquivo de entrada dados experimentais de repetições */

   is = 0;
   for (i=1;i<=N_INTER;i++){
   for (j=1;j<=N_DADOS_REAL;j++){
            is = is + 1;
            fscanf(arquivo_ent,"%f",&pJ_real_boot_base[i][j]);
            printf("%s %d %s %d %s %d %s %f\n",
            " i_inter = ",i,
            " j_repeticao = ",j,
            " is = ",is,
            " pJ_real_repeticoes = ",pJ_real_boot_base[i][j]);
            }
   }
   fclose(arquivo_sai4);
```

```
/* ##### Lê dados experimentais de tempos de falha ####### */

arquivo_sai2=fopen(file_data_sai2,"r+");

                    /* Arquivo de entrada dados reais */

is = 0;
for (j=1;j<=N_DADOS_REAL;j++){
          is = is + 1;
          fscanf(arquivo_sai2,"%d %f %f",
          &is1,&data_real[1][is],&data_real[2][is]);


          data_real[1][is] = data_real[1][is] + 0.5;
          data_real[2][is] = data_real[2][is] - 0.5;

          printf("%s %d %s %f %s %f\n",
          " i = ",is,
          " data_real[1] = ",data_real[1][is],
          " data_real[2] = ",data_real[2][is]);
          }
fclose(arquivo_sai2);


/* ############## Zera algumas variáveis ############## */

          for (k1=0;k1<=3;k1++){
          for (i1=0;i1<=(N_INTER);i1++){
                    censor_data[k1][i1] = 0;
                    count_block[i1] = 0;
                    new_block[i1] = 0;
                              }
          }


          for (k1=0;k1<=(N_DADOS_REAL+1);k1++){
                    count_block_real[k1] = 0;
                    new_block_real[k1] = 0;
          }

          for (k1=1;k1<=(N_INTER+1);k1++)
```

```
                for (i1=1;i1<=(N_PASSOS_MAX+1);i1++){
                        pJ[k1-1][i1-1] = 0.0;
                        Fac_Norm[k1-1][i1-1] = 0.0;
                        }

                for (k1=1;k1<=(N_INTER+1);k1++)
                for (i1=1;i1<=(N_PASSOS_MAX+1);i1++){
                        pJ_real[k1-1][i1-1] = 0.0;
                        Fac_Norm_real[k1-1][i1-1] = 0.0;
                        }

        /* #### Aloca os intervalos de teste Weibull em intervalos de ###### */
/* ######## tamanho T_INTER=LIM_T/N_INTER ########### */

                for (i1=1;i1<=(N_INTER);i1++){
                        if ( i1 == 1 ) {
                        censor_data[1][i1] = 0;
                        censor_data[2][i1] = T_INTER;
                                        }
                        else {
                        censor_data[1][i1] = censor_data[1][i1-1] +
                                                T_INTER;
                        censor_data[2][i1] = censor_data[1][i1] +
                                                T_INTER;
                                }
                        printf("%s %d %s %d %s %d\n",
                        " i1 = ",i1,
                        " c_left = ",censor_data[1][i1],
                        " c_right = ",censor_data[2][i1]);
                        }
        /* ####### Conta o no. de pontos Weibull ########### */
        /* ######### dentro de cada intervalo de tamanho T_INTER ##### */

        for (k1=1;k1<=(N_INTER);k1++){
                is = 0;
                for (i1 = 1; i1<=N_AMOS; i1++){
                        is = is + 1;
                   if ( (data_Weib[is] > censor_data[1][k1]) &
                      (data_Weib[is] <= censor_data[2][k1]) ){
                        count_block[k1] = count_block[k1] + 1;
                        }
                }
```

```
        if (count_block[k1] != 0) {
                        new_block[k1] = count_block[k1] ;
                                }
printf("%s %d %s %d %s %d\n",
" k1 = ",k1,
" Cblock = ",count_block[k1],
" Nblock = ",new_block[k1]);

} /* fecha for(k1=1;k1<=N_INTER;k1++) */

/* ####### Conta o no. de pontos reais dentro de cada intervalo #### */

for (k1=1;k1<=(N_INTER);k1++){
        is = 0;
        for (i1 = 1; i1<=N_DADOS_REAL; i1++){
                is = is + 1;
           if ( (data_real[1][is] > censor_data[1][k1]) &
                (data_real[2][is] <= censor_data[2][k1]) ){
                count_block_real[k1] = count_block_real[k1] + 1;
                }
        }
        if (count_block_real[k1] != 0) {
                        new_block_real[k1] = count_block_real[k1] ;
                                }
printf("%s %d %s %d %s %d\n",
" k1 = ",k1,
" Cblock_R = ",count_block_real[k1],
" Nblock_R = ",new_block_real[k1]);
}

/* ############ Define os valores de pJ na iteração 0 ###### */

        sum_pJ = 0;
        is = 0;
        for(k1=1;k1<=N_INTER;k1++) {

        if (count_block[k1] !=0) {
                /* is = is + 1; */
                pJ[k1][0] = new_block[k1]/(1.0*N_AMOS);
                sum_pJ = sum_pJ + pJ[k1][0];
                        }
        else {
```

```
                pJ[k1][0] = 0.0;
                }
        printf("%s %d %s %f %s %f\n",
        " k1 = ",k1,
        " pJ = ",pJ[k1][0],
        " sum_pJ = ",sum_pJ);

        }

        pJ[N_INTER+1][0] = 1.0 - sum_pJ;

        sum_pJ = sum_pJ + pJ[N_INTER+1][0];

        printf("%s %d %s %f %s %f\n",
        " k1 = ",(N_INTER+1),
        " pJ = ",pJ[N_INTER+1][0],
        " sum_pJ = ",sum_pJ);



    /* ########## Define os valores de pJ_REAL na iteração 0 ###### */

        sum_pJ_real = 0;
        is = 0;
        for(k1=1;k1<=N_INTER;k1++) {
        if (count_block_real[k1] !=0) {
                pJ_real[k1][0]                                          =
new_block_real[k1]/(1.0*N_DADOS_REAL);
                sum_pJ_real = sum_pJ_real + pJ_real[k1][0];
                        }
        else {
                pJ_real[k1][0] = 0.0;
                }
        printf("%s %d %s %f %s %f\n",
        " k1 = ",k1,
        " pJ_real = ",pJ_real[k1][0],
        " sum_pJ_real = ",sum_pJ_real);

        }

        pJ_real[N_INTER+1][0] = 1.0 - sum_pJ_real;
        sum_pJ_real = sum_pJ_real + pJ_real[N_INTER+1][0];
```

```
        printf("%s %d %s %f %s %f\n",
        " k1 = ",(N_INTER+1),
        " pJ_real = ",pJ_real[N_INTER+1][0],
        " sum_pJ_real = ",sum_pJ_real);



/* ######### Define vetor base para Bootstrap ######### */

        sum_pJ_real = 0;
        is = 0;
        N_dados_dif = 0;

        for(k1=1;k1<=N_INTER;k1++) {
        for (j=1;j<=N_DADOS_REAL;j++) {
                is = is + 1;
                vec_bootstrap[0][j][k1] = pJ_real_boot_base[k1][j];

                printf("%s %d %s %d %s %f %s %f\n",
                " k1_inter = ",k1,
                " j_repeticoes = ",is,
                " pJ_real_boot_base = ",pJ_real_boot_base[k1][j],
                " vec_boot_star = ",vec_bootstrap[0][j][k1]);
                        }

        }



/* ######### Vetor Bootstrap seguintes ################# */

/* ########### Semente no. aleatório ################ */

a1_aux_ran = sement_aleat;
a1_ran = a1_aux_ran;

/* ########## Inicia zero em vetores ################ */

        for (i=1;i<=N_BOOTSTRAP+1;i++)
        for (j=0;j<=N_INTER+2;j++) {
```

```
                Pm_local_bootstrap[i][j] = 0.0;
                Pm_global_bootstrap[j] = 0.0;
                sigma_pM[j] = 0.0;
                sigma_pM_novo[j] = 0.0;
                Int_Confianca[j] = 0.0;
                IC_novo[j] = 0.0;
                Pm_final_bootstrap[j] = 0.0;
                Fac_final_bootstrap[j] = 0.0;


        }

/* ############ Gera vetores Bootstrap  ################ */


        for (is1=1;is1<=N_INTER;is1++) {
        for (k1=1;k1<=N_BOOTSTRAP;k1++) {

                Pm_local_bootstrap[k1][is1] = 0.0;

        for (j=1;j<=N_DADOS_REAL;j++) {

         aux_ran = ran3(&a1_aux_ran);

         i1 = (int) ( N_DADOS_REAL * aux_ran + 1);

 /* i1 = inteiro aleatorio no intervalo [1,N_DADOS_REAL] = [1,10] */

    vec_bootstrap[k1][j][is1] = pJ_real_boot_base[is1][i1];


                /* printf("%s %d %s %d %s %d %s %d %s %f\n",
                " Inter = ",is1,
                " #bootstrap = ",k1,
                " j_repet = ",j,
                " #aleat = ",i1,
                " vec_bootstrap = ",vec_bootstrap[k1][j][is1]); */

         Pm_local_bootstrap[k1][is1] = Pm_local_bootstrap[k1][is1] +

vec_bootstrap[k1][j][is1];
```

```
                    } /* fecha o for (j=1;j<=N_DADOS_REAL;j++ ) */

                Pm_local_bootstrap[k1][is1]                                     =
Pm_local_bootstrap[k1][is1]/(1.0*N_DADOS_REAL);


                if ( k1%janela_print == 0) {

                printf("%s %d %s %d\n",
                        " Fazendo bootstrap: Passo ",k1,
                        " de ",N_BOOTSTRAP);

                printf("%s %d %s %d %s %f\n",
                        " Iter = ",is1,
                        " #BootStrap = ",k1,
                        " Pm_local ",Pm_local_bootstrap[k1][is1]);


                        }

                }       /* fecha for(k1=1; k1<=N_BOOTSTRAP;k1++) */

                } /* fecha for(is1=1;is1<=N_INTER;is1++) */



                for (is1 = 1; is1 <= N_INTER; is1++) {
                for (k1 = 1; k1 <= N_BOOTSTRAP; k1++) {

                Pm_global_bootstrap[is1] = Pm_global_bootstrap[is1] +
                                        Pm_local_bootstrap[k1][is1];

                        } /* fecha o for (k1=1;k1<=N_BOOTSTRAP;k1++) */


        Pm_global_bootstrap[is1]=Pm_global_bootstrap[is1]/(1.0*N_BOOTST
RAP);

                        printf("%s %d %s %f\n",
                        " Interv = ",is1,
                        " Pm_global_boot = ",Pm_global_bootstrap[is1]);
```

```
                }  /* fecha o for (is1=1;is1<=N_INTER;is1++) */


        for(is=1;is<=N_INTER;is++){
                sigma_pM[is] = 0.0;
                sigma_pM_novo[is]=0.0;
        for(k1=1;k1<=N_BOOTSTRAP;k1++){

                sigma_pM[is] = sigma_pM[is] +
                        sqr(    Pm_local_bootstrap[k1][is]      -
Pm_global_bootstrap[is] );

                sigma_pM_novo[is] = sigma_pM_novo[is] +
                        sqr(    Pm_local_bootstrap[k1][is]      -
Pm_global_bootstrap[is] );

                }   /* fecha o for (k1=1;k1<=N_BOOTSTRAP;k1++) */
         } /* fecha o for (is=1;is<=N_INTER;is++) */

        Tz = 1.96;

        sum_Pm_global_bootstrap = 0.0;

        for (i1=1;i1<=N_INTER;i1++) {

                sigma_pM[i1]=sqrt(
sigma_pM[i1]/(1.0*N_BOOTSTRAP) );

                Int_Confianca[i1] = Tz*sigma_pM[i1];

                Pm_final_bootstrap[i1] = Pm_global_bootstrap[i1];

                Fac_final_bootstrap[i1] = Fac_final_bootstrap[i1-1] +
                                Pm_final_bootstrap[i1];

        if ( Pm_global_bootstrap[i1] == 0 ) {

                Pm_final_bootstrap[i1] = Pm_global_bootstrap[i1-1];
                Fac_final_bootstrap[i1] = Fac_final_bootstrap[i1-1];
                sigma_pM[i1] = sigma_pM[i1-1];
                Int_Confianca[i1] = Int_Confianca[i1-1];
```

```
                    }


                printf("%s %d %s %f %s %f %s %f %s %f\n",
                " Inter = ",i1,
                " Pm_final_boot = ",Pm_final_bootstrap[i1],
                " Fac_final_bootstrap = ",Fac_final_bootstrap[i1],
                " sigma_pM = ",sigma_pM[i1],
                " Int_Conf = ",Int_Confianca[i1])


                    } /* fecha o for (i1=1;i1<=N_INTER;i1++) */

                    if ( Fac_final_bootstrap[i1] < 1.0 ) {

                    Pm_final_bootstrap[N_INTER+1]      =      1.0      -
Fac_final_bootstrap[N_INTER];
                    Fac_final_bootstrap[N_INTER+1]                    =
Fac_final_bootstrap[N_INTER] +

        Pm_final_bootstrap[N_INTER+1];
                    sigma_pM[N_INTER+1] = sigma_pM[N_INTER];
                    Int_Confianca[N_INTER+1]                          =
Int_Confianca[N_INTER];


                printf("%s %d %s %f %s %f %s %f %s %f\n",
                " Inter = ",(N_INTER+1),
                " Pm_final_boot = ",Pm_final_bootstrap[N_INTER+1],
                " Fac_final_bootstrap = ",Fac_final_bootstrap[N_INTER+1],
                " sigma_pM = ",sigma_pM[N_INTER+1],
                " Int_Conf = ",Int_Confianca[N_INTER+1])


                        }
                printf("%s \n",
                " Terminou o bootstrap ");


        /* ########## Início gravação resultado final############### */
        /* ####### de pJ_FINAL BOOTSTRAP no final das iterações###### */
```

```
arquivo_sai3=fopen(file_data_sai3,"w+");
                    /* Arquivo de SAIDA */

for (i=1;i<=N_AMOS;i++){
          k1 = (int) (data_Weib[i]);
          is_aux =  k1/T_INTER;
                    /* define a janela dos T_INTER */
                    /* em que data_weib[i] pertence */

          if ( is_aux == 0 ) {
                              is = is_aux + 1;
                                        }
          else {
                    is = is_aux;
                    }


          printf("%s %d %s %d %s %d\n",
          " i = ",i,
          " k1 = ",k1,
          " is = ",is);

          if ( ( data_Weib[i] >= is_aux*T_INTER ) &
          ( data_Weib[i] <  (is_aux+1)*T_INTER ) ) {

          /* printf("%s %d %s %d %s %d %s %f %s %f %s %f
%s %f\n",
          " i = ",i,
          " k1 = ",k1,
          " is = ",is,
          " x = ",data_Weib[i],
          " pJ = ",pJ[is][0],
          " Fac_Norm = ",Fac_Norm[is][N_passos_iter],
          " Sob = ",(1.0-Fac_Norm[is][N_passos_iter]) ); */

          fprintf(arquivo_sai3,"%d %f %f %f %f %f\n",
          i,
          data_Weib[i],
          Pm_final_bootstrap[is],
          Int_Confianca[is],
          Fac_final_bootstrap[is],
```

```
                    (1.0-Fac_final_bootstrap[is]) );
                        }

                }

                fclose(arquivo_sai3);

        /* #### Fim gravação resultado final de DE pJ_FINAL DE
BOOTSTRAP no final das iterações############################## */

                printf(" Terminou ....\n");
/************************************************************/


} /* fecha o programa PRINCIPAL  main() */
```

ANEXO C

Amostras iniciais relativas às 10 repetições do alimento mel

$P_1 = \{0,0,0,0,0,0,0,0,0,0\}$

$P_2 = \{0,0,0,0,0,0,0,0,0,0\}$

$P_3 = \{0,0,0,0,0,0,0,0,0,0\}$

$P_4 = \{0,1/8,0,0,0,0,0,0,0,0\}$

$P_5 = \{1/10,0,0,0,0,1/9,1/10,0,0,1/10\}$

$P_6 = \{0,0,1/10,0,0,0,0,0,0,0\}$

$P_7 = \{0,0,1/10,0,1/10,0,0,0,2/10,0\}$

$P_8 = \{0,1/8,1/10,0,0,0,0,2/10,0,1/10\}$

$P_9 = \{2/10,2/8,0,2/10,2/10,0,1/10,1/10,1/10,1/10\}$

$P_{10} = \{1/10,0,3/10,0,0,2/9,0,0,0,0\}$

$P_{11} = \{3/10,2/8,0,2/10,3/10,3/9,0,2/10,1/10,1/10\}$

$P_{12} = \{0,0,1/10,0,2/10,1/9,0,1/10,2/10,1/10\}$

$P_{13} = \{1/10,0,0,2/10,1/10,1/9,2/10,2/10,3/10,0\}$

$P_{14} = \{0,1/8,0,0,0,0,3/10,0,0,0\}$

$P_{15} = \{0,0,1/10,2/10,0,0,0,0,1/10,1/10\}$

$P_{16} = \{0,1/8,2/10,0,0,1/9,0,0,0,2/10\}$

$P_{17} = \{1/10,0,0,0,0,0,0,0,0,0\}$

$P_{18} = \{0,0,0,0,1/10,0,1/10,0,0,0\}$

$P_{19} = \{1/10,0,0,1/10,0,0,0,1/10,0,0\}$

$P_{20} = \{0,0,0,1/10,0,0,0,1/10,0,2/10\}$

$P_{21} = \{0,0,0,0,0,0,2/10,0,0,0\}$

$P_{22} = \{0,0,0,0,0,0,0,0,0,0\}$

ANEXO D

Amostras iniciais relativas às 10 repetições do alimento frutose

$P_1 = \{0,0,0,0,0,0,0,0,0,0\}$

$P_2 = \{0,0,0,0,0,0,0,0,0,0\}$

$P_3 = \{1/10,0,0,0,0,0,0,0,0,0\}$

$P_4 = \{0,0,0,0,0,0,0,0,0,0\}$

$P_5 = \{1/10,0,1/10,2/10,2/9,0,1/10,1/10,0,1/10\}$

$P_6 = \{2/10,2/10,1/10,2/10,0,0,1/10,0,0,1/10\}$

$P_7 = \{2/10,1/10,1/10,2/10,1/9,5/10,7/10,2/10,3/10,3/10\}$

$P_8 = \{0,1/10,0,0,0,0,0,0,0,0\}$

$P_9 = \{1/10,1/10,1/10,0,1/9,0,1/10,5/10,0,0\}$

$P_{10} = \{1/10,1/10,0,0,0,1/10,0,0,1/10,2/10\}$

$P_{11} = \{1/10,0,0,1/10,3/9,1/10,0,0,0,0\}$

$P_{12} = \{0,1/10,1/10,0,0,1/10,0,0,1/10,0\}$

$P_{13} = \{0,1/10,3/10,0,0,1/10,0,1/10,2/10,2/10\}$

$P_{14} = \{0,2/10,1/10,0,2/10,0,0,0,1/10,1/10\}$

$P_{15} = \{1/10,0,0,0,0,0,0,0,1/10,0\}$

$P_{16} = \{0,0,0,1/10,0,1/10,0,0,0,0\}$

$P_{17} = \{0,0,0,1/10,0,0,0,0,1/10,0\}$

$P_{18} = \{0,0,0,0,0,0,0,1/10,0,0\}$

$P_{19} = \{0,0,0,1/10,0,0,0,0,0,0\}$

$P_{20} = \{0,0,0,0,0,0,0,0,0,0\}$

$P_{21} = \{0,0,1/10,0,0,0,0,0,0,0\}$

$P_{22} = \{0,0,0,0,0,0,0,0,0,0\}$