Análise Estatística de Simuladores

Leonardo Bastos
University of Sheffield

Richard Wilkinson
University of Nottingham

Prefácio

Essas notas foram produzidas para o 19^o SINAPE. Nós gostaríamos de agradecer ao projeto MUCM (*Managing Uncertainty in Complex Models*). Esse é um grande projeto de pesquisa patrocinado e baseado na Grã-Bretanha onde ambos os autores trabalharam por vários anos. Um dos resultados do projeto é um ferramental disponível na web que discute muitas questões além das que foram abordadas nessas notas. Essas notas se basearam em grande parte no conteúdo desse ferramental disponível em

`http://mucm.aston.ac.uk/MUCM/MUCMToolkit/`

Nós gostaríamos de agradecer em particular Prof. Tony O'Hagan e Dr. Jeremy Oakley, pelas direções dadas e pela supervisão nesse projeto. E também, gostaríamos de agradecer a ajuda dada por Thaís Fonseca na traduçõa dessas notas do inglês para o português.

Leo Bastos e Richard Wilkinson
Rio de Janeiro, e Nottingham, 2010.

Conteúdo

Capítulo 1

Introdução

O objetivo desse mini-curso é apresentar um tutorial sobre análise estatística para simuladores. Simuladores, também chamados modelos computacionais, são usados em quase todas as áreas da ciência e tecnologia. Um experimento computacional envolve rodar o simulador em diferentes valores de entrada (inputs) com o objetivo de aprender sobre o sistema físico. Apesar dos desafios estatísticos derivados dos experimentos físicos serem bem conhecidos, os desafios derivados dos experimentos computacionais são, de certa forma, diferentes e somente recentemente começaram a ser estudados por estatísticos. Métodos para quantificar, analisar e reduzir incerteza na aplicação de experimentos computacionais têm atraído a atenção de usuários de simulação, e nesse curso nós iremos descrever os desafios enfrentados e introduzir algumas das metodologias para lidar com essas questões.

Alguns dos desafios que iremos discutir incluem

- Limitação dos recursos computacionais - se o nosso simulador leva um tempo não trivial para rodar, nós estaremos limitados ao número de rodadas que podemos considerar (conhecido como incerteza no código)

- Administração do erro no modelo - como podemos passar de suposições sobre o modelo (i.e., 'dado que um modelo é verdadeiro, nossa previsão é...') para fazer suposições sobre o sistema físico (i.e. 'nossa previsão é...').

- Análise de incerteza/sensibilidade - se existe incerteza sobre os inputs do simulador, como isso afetará nossa incerteza sobre o output?

- Calibração - como aprendemos sobre parâmetros desconhecidos?

Uma das principais idéias consideradas nesse curso é a meta-análise. Se podemos construir uma representação estatística do simulador que é rápida de rodar, conhecida como emulador, então nós podemos usar o emulador no lugar do simulador para a análise. Processos gaussianos são a principal ferramenta usada para meta-análise. Num contexto bayesiano, esses processos fornecem um método semi-paramétrico flexível para representar nossos julgamentos sobre os outputs do simulador. Condicional num conjunto de rodadas do simulador, nós construímos um processo gaussiano que pode ser usado como um substituto do simulador. Dessa forma, alguns dos desafios citados acima podem ser solucionados sem mais rodadas do simulador.

O conteúdo do curso será dividido ao longo dos três dias da seguinte forma: no primeiro dia, nós iremos introduzir a área de experimentos computacionais e descrever como construir emuladores baseados em processos gaussianos; no segundo dia, nós iremos lidar com alguns dos desafios listados acima, incluindo análise de sensibilidade, planejamento de experimentos, e calibração; e no último dia, nós iremos apresentar alguns trabalhos em desenvolvimento nesta área, incluindo a área de computação bayesiana aproximada que têm crescido muito rapidamente.

1.1 Introdução a simuladores

Simuladores, também conhecidos como modelos computacionais, são representações matemáticas de um sitema real implementadas num computador. Assume-se que o simulador, representado por $\eta(\cdot)$, é uma função de um conjunto de inputs denotados por $\mathbf{x} = (x_1, \ldots, x_p) \in \chi \subset \Re^p$, com o output representado por $y \in \Re$. Um experimento computacional é um conjunto de rodadas do simulator em diferentes valores dos inputs, $(y_1 = \eta(\mathbf{x}_1), \ldots, y_n = \eta(\mathbf{x}_n))$. Experimentos computacionais têm sido usados para investigar sistemas do mundo real em quase todas as áreas da ciência e tecnologia. Existem situações onde o experimento computacional é possivel quando o experimento físico é impossível. Por exemplo, o número de inputs pode ser muito grande para realização do experimento físico, o custo para realizar o experimento pode ser muito alto, ou o experimento físico é anti-ético.

O uso de experimentos computacionais datam de 1940 nos Laboratórios nacionais de Los Alamos para estudar o comportamento de armas nucleares. Em 1944[1], houve uma investigação quantitativa da hidrodinâmica de

[1] Capítulo 4, Los Alamos: Revisão técnica até agosto de 1944, disponível em `http://www.fas.org/sgp/othergov/doe/lanl/00795708.pdf`

uma implosão nuclear onde máquinas de cálculo da IBM foram usadas para resolver equações diferenciais parciais da hidrodinâmica de implosões.

São muitos os exemplos de desenvolvimento científico e tecnológico que foram conduzidos usando experimentos computacionais. Em ciência climática, Zickfeld et al. (2004) apresentam um modelo de baixa ordem para a circulação térmica no Atlântico que é capaz de reproduzir muitas características do comportamento de modelos para a circulação distribuição regional e taxa de mudança climática. Randall et al. (2007) avaliou a capacidade e limitações de modelos de clima global usando o IPCC (Intergovernmental Panel on Climate Change). Em cosmologia, Benson et al. (2001) usou um modelo computacional complexo para entender o processo responsável pela formação e evolução das galáxias. Em engenharia de proteção contra incêndios, McGrattan et al. (2007) apresenta um simulador da dinâmica de incêndios usado para entender a dinâmica do fogo durante um incêndio usando as soluções númericas de uma equação Navier-Stokes apropriada para velocidades baixas, fluidos comandados por fogo e transporte de calor de incêndios.

1.2 Definindo simuladores e emuladores

Simuladores são geralmente modelos determinísticos de entrada e saída (input-output), onde rodar o simulador novamente para o mesmo valor do input sempre resulta no mesmo output. Entretanto, o valor do output é desconhecido antes de rodarmos o simulador para um conjunto de inputs particular. Num contexto bayesiano, incerteza sobre o output do simulador, também chamada de incerteza do código, pode ser expressa por um processo estocástico. O resultado é uma representação estatística do simulador, conhecida como emulador. Emuladores foram desenvolvidos no final da década de 1980 por experimentadores computacionais, e.g. Currin et al. (1988, 1991) e Sacks et al. (1989b). O'Hagan (1978) descreve como usar um processo gaussiano para representar uma função desconhecida, e os processos gaussianos são a principal ferramenta para construir um emulador que represente as nossas crenças sobre o simulador.

1.3 Emuladores gaussianos

Um emulador é uma distribuição de probabilidade que representa a incerteza do simulador, onde o simulador é visto como uma função matemática desconhecida. Apesar do simulador ser, em principio, conhecido, nós estamos

incertos sobre o output do simulador para um valor não usado do input. Sob o ponto de vista bayesiano Kimeldorf and Wahba (1970) e O'Hagan (1978) usam processos gaussianos para descrever o comportamento de uma função matemática desconhecida. Nos anos 1980, foi introduzida a idéia fundamental de construir um emulador usando processos gaussianos, num contexto não bayesiano por Sacks et al. (1989b) e num contexto bayesiano por Currin et al. (1988, 1991). Nós revisamos as principais idéias de emuladores gaussianos sob o ponto de vista bayesiano. Para o ponto de vista frequentista, veja também Santner et al. (2003, section 3.3).

Assume-se que o simulador, representado por $\eta(\cdot)$, é uma função de um conjunto de inputs denotados por $\mathbf{x} = (x_1, \ldots, x_p) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_p = \mathcal{X} \subset \mathbb{R}^p$, com o output representado por $y \in \mathbb{R}$. Com o objetivo de construir um emulador gaussiano, a incerteza sobre o simulador é descrita como um processo gaussiano com uma função de média $m(\cdot)$, e função de covariância $V(\cdot, \cdot)$. Formalmente, se $\eta(\cdot)$ é um processo gaussiano então para cada $n = 1, 2, \ldots$ a distribuição conjunta de $\eta(\mathbf{x}_1), \ldots, \eta(\mathbf{x}_n)$ é normal multivariada para todo $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$. A função de média $m(\cdot)$ pode ser qualquer função de $\mathbf{x} \in \mathcal{X}$, mas $V(\cdot, \cdot)$ deve satisfazer a propriedade de que cada matriz de covariância com elementos $\{V(\mathbf{x}_i, \mathbf{x}_j)\}$ deve ser não negativa definida.

Nossas crenças a priori sobre o simulador $\eta(\cdot)$ são representadas por um processo gaussiano com média $m_0(\cdot)$ e função de covariância $V_0(\cdot, \cdot)$. Usando essa formulação,

$$\eta(\cdot)|\beta, \sigma^2, \delta \sim GP\left(m_0(\cdot), V_0(\cdot, \cdot)\right), \tag{1.1}$$

onde a função de média $m_0(\cdot)$ é dada por

$$m_0(\mathbf{x}) \quad = \quad h(\mathbf{x})^T \beta, \tag{1.2}$$

onde $h(\cdot) : \chi \subset \mathbb{R}^p \longmapsto \mathbb{R}^q$ é uma função desconhecida dos inputs, onde q pode ser diferente da dimensão do espaço dos imputs p, e β é um vetor de coeficientes desconhecidos. A função $h(\cdot)$ deve ser escolhida para incorporar qualquer conhecimento a priori sobre a forma de $\eta(\cdot)$.

A escolha de $h(\cdot)$ depende das crenças a priori sobre o simulador. O caso mais simples é obtido quando $q = 1$ e $h(\mathbf{x}) = 1$ for all x. Então a função de média é $m(\mathbf{x}) = \beta$, onde agora β é um hiperparamêtro escalar representando uma média global desconhecida do output do simulador. Essa escolha expressa nenhum conhecimento a priori sobre como o output vai responder a variações no input. Outro exemplo simples é $h(\mathbf{x})^T = (1, \mathbf{x})$, de forma que $q = 1 + p$, onde p é o número de inputs. Então $m(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$ o que expressa uma expectativa a priori de que o

output do simulador irá mostrar uma tendência em resposta a cada um dos inputs, mas não há nenhuma informação a priori sugerindo qualquer não-linearidade nessas tendências. Se houver uma crença a priori de não-linearidade da resposta, então termos quadráticos ou polinomiais podem ser introduzidos em $h(\cdot)$. Neste texto, nossas crenças a priori para o simulador são representadas por uma média linear $m(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$.

A função de covariância $V_0(\cdot, \cdot)$ é dada por

$$V_0(\mathbf{x}, \mathbf{x}') \quad = \quad \sigma^2 C_\delta(\mathbf{x}, \mathbf{x}'), \tag{1.3}$$

onde σ^2 é um parâmetro de escala desconhecido, e $C_\delta(\cdot, \cdot)$ é uma função de correlação conhecida com vetor de parâmetros de correlação desconhecidos δ. A função de correlação escolhida $C_\delta(\cdot, \cdot)$ deve garantir que a matriz de covariância de qualquer conjunto de inputs é não negativa definida.

Funções de correlação

Uma função de correlação $C_\delta(\mathbf{x}, \mathbf{x}')$ representa a correlação entre os outputs do simulador nos inputs $\mathbf{x}$ e $\mathbf{x}'$ com paramêtro de correlação δ. Em análise estatística para simuladores é comum usar funções de correlação estacionárias. Nós dizemos que uma função de correlação $C(\cdot, \cdot)$ é estacionária se $C(\mathbf{x}, \mathbf{x}') = R(\mathbf{x} - \mathbf{x}')$, e dizemos que a função é isotrópica se $C(\mathbf{x}, \mathbf{x}') = R(||\mathbf{x} - \mathbf{x}'||)$.

Uma família de funções de correlação que é muito usada na literatura na especificação de um processo gaussiano é a função de correlação poder exponencial,

$$C_\delta(\mathbf{x}, \mathbf{x}') = \exp\{-|(\mathbf{x} - \mathbf{x}')/\delta_1|^{\delta_2}\}, \quad \text{for } \{\mathbf{x}, \mathbf{x}'\} \in \mathcal{X} \tag{1.4}$$

onde $\delta_1 > 0$, e $\delta_2 \in (0, 2]$. Se $\delta_2 = 2$ então $C_\delta(\cdot, \cdot)$ é a função de correlação exponencial quadrática, também conhecida como função de correlação gaussiana. Uma extensão dessa família é a versão separável em dimensão p da função de correlação exponencial potência,

$$C_\delta(\mathbf{x}, \mathbf{x}') = \exp\left\{-\sum_{i=1}^{p} |(x_i - x_i')/\delta_i|^{\delta_{p+i}}\right\}, \tag{1.5}$$

onde $\delta_i > 0$, e $\delta_{p+i} \in (0, 2]$ para $i = 1, \ldots, p$. Como um caso especial, a versão separável em dimensão p da função de correlação exponencial quadrática é

$$C_\delta(\mathbf{x}, \mathbf{x}') = \exp\left\{-\sum_{i=1}^{p} |(x_i - x_i')/\delta_i|^2\right\}, \tag{1.6}$$

onde os parâmetros $\delta = (\delta_1, \ldots, \delta_p)$ são chamados parâmetros de alcance da correlação. Essa fórmula mostra a função de cada parâmetro de alcance da correlação δ_i. Quanto menor o seu valor, mais próximos $\mathbf{x}_i$ e $\mathbf{x}_i'$ devem estar para que os outputs em $\mathbf{x}$ e $\mathbf{x}'$ sejam altamente correlacionados. Valores grandes (pequenos) de δ_i significam que os valores dos outputs são correlacionados em um grande (curto) intervalo do i-ésimo input x_i. Alguns autores usam uma parameterização diferente para os parâmetros de correlação, por exemplo, $\theta_i = \delta_i^{-\frac{1}{2}}$, onde θ_i é chamado parâmetro de suavidade.

Outras funções de correlação, como a linear e a cúbica são apresentadas em Currin et al. (1991). Para mais detalhes sobre funções de correlação para processos gaussianos, veja Cressie (1993), Santner et al. (2003), and Rasmussen and Willians (2006). No texto, nós usamos a versão em dimensão p da função de correlação exponencial quadrática (1.6) com parâmetros de correlação descritos pelo alcance da correlação.

Dados de treinamento

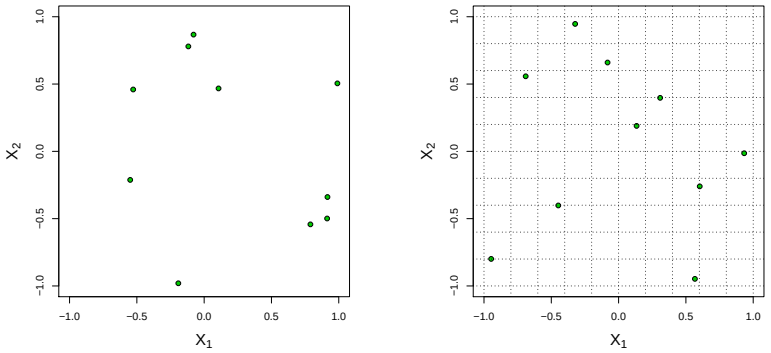
Suponha que $\mathbf{y} = [y_1 = \eta(\mathbf{x}_1), \ldots, y_n = \eta(\mathbf{x}_n)]$ contém n valores do output do simulador em pontos do planejamento $\mathbf{x}_1, \ldots, \mathbf{x}_n$ no espaço dos inputs χ. Esses dados são chamados **dados de treinamento**. Os pontos do planejamento são escolhidos de forma a cobrir todo o espaço dos inputs. Chapman et al. (1994) sugerem que o tamanho da amostra dos dados de treinamento deve ser pelo menos dez vezes a dimensão do espaço dos inputs, i.e., $n = 10d$. Recentemente, Loeppky et al. (2009) mostram que $n = 10d$ é uma regra razoável para experimentos iniciais.

Planejamento para construção de emuladores

No processo de construção de um emulador, nós precisamos resposder a seguinte questão: *Em quais valores no espaço dos inputs devemos rodar o simulador de forma a minimizar nossa incerteza com respeito ao simulador?* Para responder essa questão, vários planejamentos para modelos computacionais devem ser desenvolvidos.

O caso mais simples é uma amostra aleatória da distribuição uniforme no espaço dos inputs. Figura 1.1 (a) ilustra o planejamento aleatório simples de tamanho 10 na região $\mathcal{X} = [-1, 1]^2$. O problema com esse método é que algumas regiões do espaço dos inputs podem ser completamente não cobertas. No nosso exemplo, Figura 1.1 (a), vemos que valores pequenos de X_1 não são cobertos.

McKay et al. (1979) propõem amostragem do tipo hipercubo latino para escolher inputs para o simulador. O hipercubo latino garante que cada input é bem representado no planejamento. Figura 1.1 (b) ilustra um hipercubo latino de tamanho 10 da região $\mathcal{X} = [-1, 1]^2$. Nós vemos que marginalmente, ambos os espaços de variáveis do input são bem cobertos.



(a) (b)

Figura 1.1: Planejamentos baseados em amostragem com 10 elementos e 2 dimensões na região $\mathcal{X} = [-1, 1]^2$: (a) Amostra aleatória uniforme; (b) Amostra de hipercubo latino.

Na Seção 2.1, nós discutimos o problema de planejamento e apresentamos outros planejamentos para construção de emuladores tais como planejamentos baseados em distâncias, planejamentos ótimos, planejamentos em grade (*lattice design*) e planejamentos não aleatórios.

Atualizando o processo a priori

De acordo com (1.1) a distribuição dos outputs é normal multivariada:

$$\mathbf{y}|\beta, \sigma^2, \delta \sim N_n \left(H\beta, \sigma^2 A\right), \tag{1.7}$$

onde

$$H = [h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n)]^T, \tag{1.8}$$

e A é a matriz com elementos

$$A_{i,j} = C_\delta(\mathbf{x}_i, \mathbf{x}_j). \tag{1.9}$$

Usando propriedades conhecidas da distributição normal multivariada, pode-se mostrar que

$$\eta(\cdot)|\beta, \sigma^2, \delta, \mathbf{y} \sim GP\left(m_0^*(\cdot), V_0^*(\cdot, \cdot)\right), \tag{1.10}$$

onde

$$\begin{aligned} m_0^*(x) &= h(x)^T\beta + t_\delta(x)^T A^{-1}(\mathbf{y} - H\beta), \\ V_0^*(x,x') &= \sigma^2\left[C_\delta(x,x') - t_\delta(x)^T A^{-1} t_\delta(x')\right], \end{aligned}$$

onde $t_\delta(x) = (C_\delta(x,\mathbf{x}_1),\ldots,C_\delta(x,\mathbf{x}_n))^T$.

Usando uma priori não informativa para (β,σ^2), $p(\beta,\sigma^2)\propto\sigma^{-2}$, combinando com (1.7), usando o teorema de Bayes, a posteriori para (β,σ^2) é a distribuição gamma normal inversa, caracterizada por

$$\beta|\mathbf{y},\sigma^2,\delta \;\sim\; N\left(\widehat{\beta},\sigma^2\left(H^TA^{-1}H\right)^{-1}\right) \tag{1.11}$$

onde $\widehat{\beta} = \left(H^TA^{-1}H\right)^{-1}H^TA^{-1}\mathbf{y}$, e

$$\sigma^2|\mathbf{y},\delta \;\sim\; \text{InvGam}\left(\frac{n-q}{2},\frac{(n-q-2)\widehat{\sigma}^2}{2}\right), \tag{1.12}$$

onde $\widehat{\sigma}^2 = \dfrac{\mathbf{y}^T\left(A^{-1} - A^{-1}H\left(H^TA^{-1}H\right)^{-1}H^TA^{-1}\right)\mathbf{y}}{n-q-2}$.

Integrando β do produto (1.10) e (1.11), pode-se mostrar que

$$\eta(\cdot)|\mathbf{y},\sigma^2,\delta \sim GP\left(m_1(\cdot),V_1^*(\cdot,\cdot)\right), \tag{1.13}$$

onde

$$\begin{aligned} m_1(x) &= h(x)^T\widehat{\beta} + t_\delta(x)^TA^{-1}(\mathbf{y} - H\widehat{\beta}), &(1.14)\\ V_1^*(x,x') &= \sigma^2\left[C_\delta(x,x') - t_\delta(x)^TA^{-1}t_\delta(x')\right. \\ &+ \left.\left(h(x) - t_\delta(x)^TA^{-1}H\right)\left(H^TA^{-1}H\right)^{-1}\left(h(x') - t_\delta(x')^TA^{-1}H\right)^T\right] &(1.15)\end{aligned}$$

Emulador t-student

O emulador t-student é obtido integrando σ^2 do produto (1.12) e (1.13), e é dado por

$$\eta(\cdot)|\mathbf{y},\delta \sim \text{Processo t-student}\left(n-q,m_1(\cdot),V_1(\cdot,\cdot)\right), \tag{1.16}$$

onde

$$\begin{aligned} m_1(x) &= h(x)^T\widehat{\beta} + t_\delta(x)^TA^{-1}(\mathbf{y} - H\widehat{\beta}), &(1.17)\\ V_1(x,x') &= \widehat{\sigma}^2\left[C_\delta(x,x') - t_\delta(x)^TA^{-1}t_\delta(x')\right. \\ &+ \left.\left(h(x) - t_\delta(x)^TA^{-1}H\right)\left(H^TA^{-1}H\right)^{-1}\left(h(x') - t_\delta(x')^TA^{-1}H\right)^T\right] &(1.18)\end{aligned}$$

onde $\widehat{\beta} = \left(H^T A^{-1} H\right)^{-1} H^T A^{-1} \mathbf{y}$ e $\widehat{\sigma}^2 = \dfrac{\mathbf{y}^T \left(A^{-1} - A^{-1} H \left(H^T A^{-1} H\right)^{-1} H^T A^{-1}\right) \mathbf{y}}{n - q - 2}$.

De forma análoga ao processo gaussiano, se $\eta(\cdot)$ é um processo t-student então para cada $m = 1, 2, \ldots$ a distribuição conjunta de $\eta(\mathbf{x}_1), \ldots, \eta(\mathbf{x}_m)$ é t-student multivariada para todo $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathcal{X}$.

Inferência para o alcance da correlação

O vetor de parâmetros de correlação δ é desconhecido. Uma distribuição a priori para δ deve ser especificada. Considere $p(\delta)$ como a distribuição a priori para δ. A densidade a posteriori é obtida via

$$\begin{aligned} p(\delta|\mathbf{y}) &\propto p(\delta) \int\int p(y|\beta, \sigma^2, \delta) p(\beta, \sigma^2) d\beta d\sigma^2 \\ &\propto p(\delta)|A|^{-\frac{1}{2}} |H^T A^{-1} H|^{-\frac{1}{2}} (\hat{\sigma}^2)^{-\frac{n-q}{2}} \end{aligned} \tag{1.19}$$

onde A e $\hat{\sigma}^2$ são funções de δ. Paulo (2005) apresenta prioris de referência para os parâmetros de alcance da correlação.

Uma análise completamente bayesiana deve integrar o alcance δ do produto de densidades em (1.16) e (1.19). A distribuição a posteriori de δ é intratável, mas métodos bayesianos aproximados podem ser aplicados, tais como aproximação de Laplace (Lindley 1980). Métodos de Monte Carlo via Cadeia de Markov (MCMC) podem ser usados para uma análise completamente bayesiana como em Bayarri et al. (2007), mas esses métodos exigem métodos computacionais intensivos. Alternativamente, δ pode ser estimado da distribuição a posteriori $p(\delta|\mathbf{y})$, ou da verossimilhança $p(\mathbf{y}|\delta)$, e então procedemos a análise como se essas estimativas fossem os valores verdadeiros. Essa abordagem é chamada **the plug-in approach**. Estimativas de δ são mais frequentemente escolhidas usando um método de otimização para achar a moda a posteriori (Sacks et al. 1989b; Currin et al. 1991).

1.4 Exemplos

Exemplo 1.4.1 (Exemplo em 1D) *Considere o simulador dado pela função*

$$\eta(x) = 5 + x + \cos x.$$

Figura 1.2 apresenta a função usada como simulador com alguns dados de treinamento, os quais são usados para construir o emulador.

Nós assumimos que o simulador representa um processo gaussiano como em (1.1). A distribuição a priori $(\beta, \sigma^2, \delta)$ é $p(\beta, \sigma^2, \delta) \propto \sigma^{-2} p(\delta)$, onde

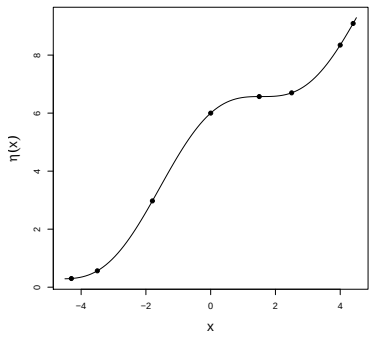
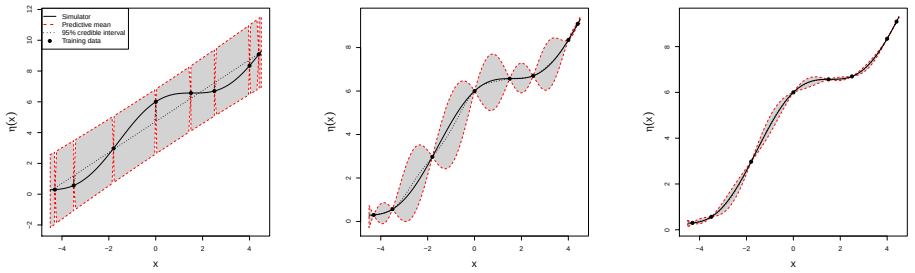


Figura 1.2: Simulador do exemplo 1.4.1 com alguns dados de treinamento dados por •.

δ segue uma distribuição log Normal com parâmetros $\mu = 0$ e $\sigma^2 = 100$. A razão da escolha desta priori log normal é que ela é uma distribuição própria porém não informativa. Portanto, condicional aos dados de treinamento (Figure 1.2) e em uma estimativa do alcance da correlação, nossas crenças sobre o simulador podem ser descritas por (1.16). A Figura 1.3 apresenta alguns emuladores t-student condicionais aos dados de treinamento e diferentes valores para o alcance da correlação, δ. Note que para todos os casos, o simulador está contido nos intervalos de credibilidade de 95%.

A figura 1.3 (a) apresenta o caso independente, i.e. quando $\delta = 0$. Note que o valor esperado do emulador avaliado em inputs não observados é representado pela reta de regressão ajustada aos dados de treinamento, e os tamanhos dos intervalos de credibilidade para a maioria dos inputs são os mesmos. Este emulador é sub-confiante. A figura 1.3 (b) apresenta o caso onde $\delta = 1$. Nós assumimos que existe alguma correlação entre outputs para inputs próximos entre si. Nós observamos que previsões para inputs próximos a dados de treinamento tem pequenos intervalos de 95% de credibilidade, por outro lado, o tamanho dos intervalos de crediidibilidade aumenta a medida que aumentamos a distância entre o ponto que queremos prever e o dado de treinamento mais próximo. Agora nós estamos mais confiantes para este emulador do que para o caso independente. A figura 1.3 (c) apresenta o caso onde $\delta = 2$. Agora, a correlação entre outputs para inputs próximos é significativamente maior. Portanto, o tamanho dos intervalos é menor. Nós estamos mais confiantes do que estavamos para os emuladores anteriores. Entretanto, nós precisamos justificar o uso de um alcance de correlação desse tamanho. Nós poderiámos ir aumentando

o alcance da correlação, mas nós iríamos ser super confiantes a respeito do comportamento do simulador e fazer algumas previsões erradas.



(a) $\delta = 0$ (b) $\delta = 1$ (c) $\delta = 2$

Figura 1.3: Emulador t de Student condicionado aos dados de treinamento e diferentes valores para o alcance da correlação.

Nós estimamos o alcance da correlação δ de sua distribuição a posteriori, equação (1.19), usando como priori uma distribuição log Normal(0,100). A moda da posteriori é dada por $\hat{\delta} = 3.857041$. A Figura 1.4 (a) mostra as distribuições a priori e a posteriori para δ. A distribuição a priori para δ é não informativa. O emulador t-student usando $\hat{\delta}$ como o valor verdadeiro do alcance da correlação é apresentado na Figura 1.4 (b). Notamos que as previsões são precisas com pequenos intervalos de credidibilidade de 95%. A diferença entre o simulador e o valor esperado do emulador é apresentada na Figura 1.4 (c). Agora, podemos ver claramente os intervalos de 95% de credibilidade. Embora as previsões do emulador pareçam perfeitas na Figura 1.4 (b), quando olhamos para a diferença entre os outputs do simulador e o valor esperado do emulador, o emulador parece ser sub confiante.

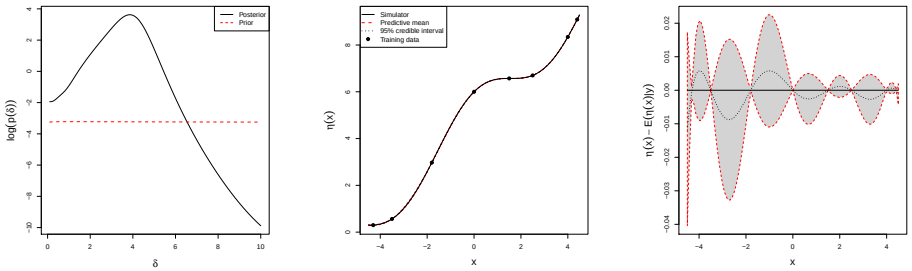
Exemplo 1.4.2 *Suponha que o simulador seja a seguinte função matemática*

$$\eta(x_1, x_2) = \left(1 - e^{-\frac{1}{2x_2}}\right)\left(\frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}\right), \qquad (1.20)$$

onde $(x_1, x_2) \in (0,1)^2$. *Este é um exemplo usado no programa GEM-SA*[2].

Usando os dados de treinamento, nós estimamos o alcance da correlação maximizando a função (1.19). As estimativas são $(\hat{\delta}_1, \hat{\delta}_2) = (0.2421, 0.4240)$, indicando que o simulator é mais suave com respeito ao segundo input do que com o primeiro.

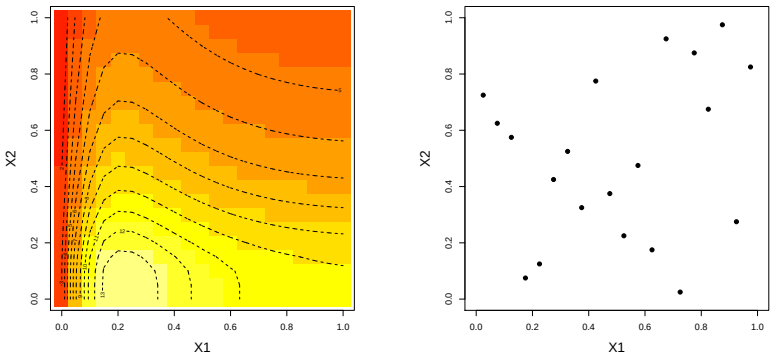
[2]O programa GEM-SA está disponível em `http://ctcd.group.shef.ac.uk/gem.html`.



(a) (b) (c)

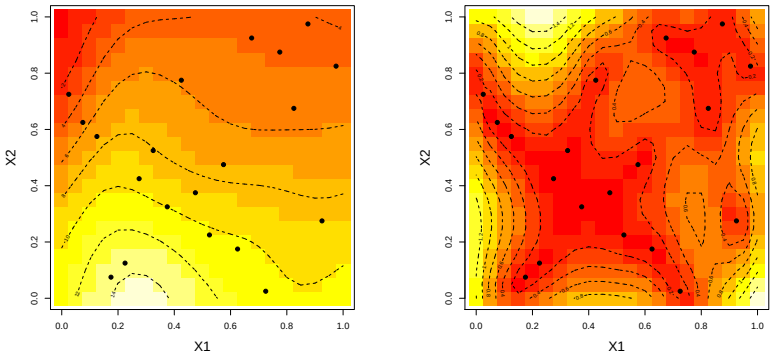
Figura 1.4: Exemplo 1.4.1: (a) Distribuição a priori e posteriori para o alcance da correlação; (b) Emulator t de Stuendt conditional aos dados de treinamente e a moda a posteriori de δ; (c) Diferença entre o simulador e o emulador esperado com intervalos de credibilidade de 95%.

Usando os dados de treinamento e as estimativas para o alcance da correlacao, nós construimos o processo t-Student, dado em (1.16), e outouts para o simulador foram preditos em uma grade com 20×20 pontos sobre o espaço de inputs. A Figura 1.6 (a) mostra o valor esperado do emulador and (b) os desvios padrões aplicados sobre a grade de pontos. Visualmente, o emulador usando 20 dados de treinamento é capaz de capturar as principais características do simulador. Note que o eixo dis outputs nas Figuras 1.5 (a) and 1.6 (a) estão na mesma escala. Os desvios padrão, como esperado, são pequenos próximos aos dados de treinamento, e maiores a medida que se distanciam dos dados de treinamento, onde existe pouca informação.



(a) (b)

Figura 1.5: Exemplo 1.4.2: (a) Simulador aplicado ao espaço de inputs; (b) Dados de treinamento.



(a) (b)

Figura 1.6: Emulador t de Student usando os dados de treinamento: (a) média do emulador; (b) desvio padrão do emulador.

Capítulo 2

Usando emuladores

Nessa seção, abordamos os desafios encontrados em experimentos computacionais. Emuladores podem ser usados em muitos desses casos para tornar a análise tratável. Boa parte do material usado neste capítulo foi tirada do excelente *toolkit* produzido pelo projeto MUCM. O endereço eletrônico das páginas relevantes é dado ao longo do texto.

2.1 Planejamentos para construir emuladores

Uma vez que o comportamento dos simuladores é desconhecido, se faz necessário obter planejamentos que trazem informação a respeito do output do simulador em qualquer ponto do espaço dos inputs. Previsões para o simulador são feitas usando emuladores que representam nossos julgamentos probabilísticos a respeito dos simuladores. Planejamento e previsão para simuladores foram inicialmente abordados na literatura por Sacks et al. (1989a,b).

Para construir um emulador, precisamos rodar o simulador em diferentes pontos do espaço dos inputs. O processo de escolher estes pontos é o problema de planejamento considerado aqui. O planejamento mais simples usa amostragem uniforme, onde amostramos de uma distribuição uniforme sobre o espaço dos inputs. Um problema com este método é que algumas regiões do espaço dos inputs podem não ser representadas. Uma maneira de resolver isso é usando amostragem estratificada, onde o espaço dos inputs é particionado em J estratos disjuntos, $\mathcal{X}_j$, e em cada estrato uma amostra aleatória uniforme de tamanho n_j é selecionada. Usando a amostragem estratificada , todas regiões do espaço dos inputs $\mathcal{X}$ estão representadas pelos $\sum_j n_j$ inputs selecionados.

O output do simulador pode, na prática, depender somente de alguns inputs, nesse caso nós queremos garantir que os valores selecionados estejam bem distribuídos nestes fatores. Um planejamento que distribui os pontos igualmente atravéz do espaço de inputs não necessariamente tem essa propriedade. O hipercubo latino é um planejamento tal que uma projeção do conjunto de pontos em uma das dimensões tem pontos uniformemente distribuídos nessa dimensão. Na próxima seção nós vamos introduzir os hipercubo latinos. Nós também apresentaremos outros planejamentos que distribuem os pontos uniformemente no espaço dos inputs chamados de planejamentos baseados em distâncias. E finalmente, nós mostraremos alguns planejamentos não aleatórios usados em integração numérica.

2.1.1 Hipercubo latino

O uso de hipercubo latinos é popular em experimentos computacionais (Santner et al. 2003). Essa popularidade é atribuída a duas razões. Primeiro, hipercubo latinos são simples de serem gerados. Segundo, em um hipercubo latino as observações são uniformemente distribuídas para cada input.

McKay et al. (1979) propôs o hipercubo latino como alternativa a amostragem aleatória simples em um estudo de Monte Carlo. Hipercubo latino garante que marginalmente, o espaço amostral de cada input é bem coberto. Stein (1987) mostrou que, assintoticamente, a variância do valor esperado do output do simulador é menor que a variância obtida usando amostragem aleatória simples. Stein também propôs um método para produzir hipercubos latinos quando os inputs são dependentes. Aqui, nós assumimos que todos inputs são independentes.

O seguinte procedimento descreve como obter um hipercubo latinos, $\mathbf{X}$.

Seja $\mathbf{B}$ uma matriz $n \times p$, onde cada coluna de $\mathbf{B}$ é uma permutação aleatória e independente de $\{1, 2, \ldots, n\}$. Seja U_{ij} $(i = 1, 2 \ldots, n,\ j = 1, 2 \ldots, p)$ np variáveis aleatórias independentes uniformemente distribuídas em (0, 1) que também são independentes de $\mathbf{B}$. Então o j-ésimo elemento do i-ésimo input é dado por

$$\mathbf{X}_{ij} = \frac{B_{ij} + U_{ij}}{n}.$$

Note que se o espaço dos inputs é $[a, b]^p$, uma transformação simples pode ser usada.

Embora o hipercubo latino represente bem todos os inputs de forma marginal, não existe garantia que tenha boas propriedades de preenchimento do espaço (space-filling).

2.1.2 Planejamentos baseados em distâncias

Planejamentos baseados em distâncias dependem de medidas de distância que quantificam quão uniformemente distribuídos estão as observações. Seja $\mathbf{X} \subset \mathcal{X}$ um planejamento qualquer consistindo dos pontos $\{x_1, x_2, \ldots, x_n\}$. Seja *dist* uma medida em $\mathcal{X}$, por exemplo, a distância euclidiana.

Para qualquer planejamento, a menor distância entre dois pontos pode ser obtida. Um planejamento que maximiza essa medida é chamada de planejamento *maximin*. Um planejamento maximin é denotado por $\mathbf{X}_{Mm}$, e é formalmente dado por

$$\mathbf{X}_{Mm} = \max_{\mathbf{X} \subset \mathcal{X}} \min_{\{x,x'\} \in \mathbf{X}} dist(x, x'), \tag{2.1}$$

onde $dist(x, x')$ é a distância euclidiana entre x e x'. Planejamentos maximin tendem a distribuir pontos próximos as bordas da regão do espaço dos inputs, o que pode não trazer informação suficiente a respeito da região central do espaço dos inputs.

Um outro planejamento baseado em distâncias é um planejamento que todo ponto do espaço de inputs $\mathcal{X}$ está próximo a algum ponto de $\mathbf{X}$. Tal planejamento é chamado de planejamento *minimax* que é denotado por $\mathbf{X}_{mM}$, e é formalmente dado por

$$\mathbf{X}_{mM} = \min_{\mathbf{X} \subset \mathcal{X}} \max_{x \in \mathcal{X}} \min_{x' \in \mathbf{X}} dist(x, x'). \tag{2.2}$$

Um problema com planejamentos minimax é que eles são difíceis de serem gerados. Consequentemente, raramente são usados.

Maximin hipercubos latinos

Dentro da classe dos hipercubo latinos, nós podemos escolher aqueles que satisfazem um critério baseado em distâncias, por exemplo a maior distância mínima. Tal planejamento é chamado de maximin hipercubo latino. Este planejamento foi estudado por Morris and Mitchell (1997). A figura 2.1 (a) apresenta 50 pontos de um maximin hipercubo latino de duas dimensões, onde 10.000 hipercubo latinos foram gerados e o planejamento com a maior distância mínima é escolhido.

2.1.3 Planejamentos não aleatórios

Planejamento em grade (*Lattice designs*). Em um *lattice design* os pontos são escolhidos em uma grade regular super imposta no espaço dos

inputs. Em experimentos computacionais *lattice designs* foram considerados por Bates et al. (1996, 1998).

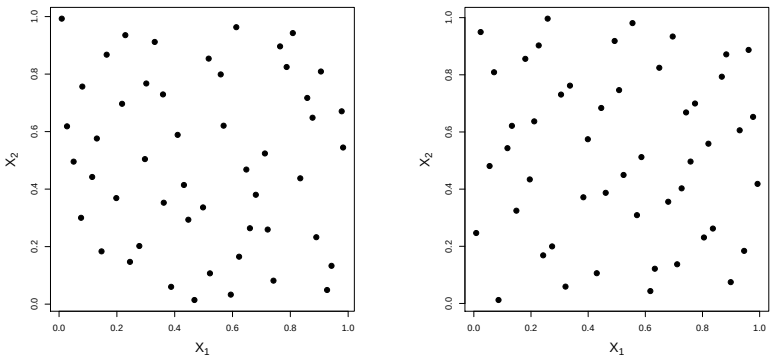
Um grande número de diferentes sequências de números com propriedade de preenchimento de espaço foram propostas. As sequências usam diferentes algoritmos de geração, mas todas têm a propriedade de serem potencialmente de tamanho infinito, pois uma nova sequência pode ser adicionada a uma sequência antiga. Um planejamento é obtido simplesmente pegando os primeiros valores dessa sequencia. Exemplos são a sequêcia de Weyl, a sequêcia de Halton e a sequêcia de Sobol (Neiderreiter 1992). A sequêcia de Weyl é similar ao *lattice design* na forma em que ela é gerada, mas os geradores são números irracionais. A sequência de Halton tem geradores inteiros primos para cada dimensão onde cada número primo gera uma sequência de frações.

A sequência de Sobol. A sequência de Sobol usa as mesmas coordenadas que a sequência de Halton com gerador 2 para cada dimensão, mas a sequência é reordenada segundo uma regra complicada. Galanti and Jung (1997) ilustra a sequência de Sobol incluindo exemplos numericos simples. Existe uma função no R para produzir sequências de Sobol, a função *runif.sobol* do pacote 'fOptions'. A figura 2.1 (b) mostra 50 pontos de uma sequência de Sobol de duas dimensões.

Uma vantagem das sequências de Sobol é que elas são computacionalmente mais baratas de serem geradas do que outras sequências. Comparadas com hipercubos latinos, sequências de Sobol podem ser construídas adicionando pontos a sequências de Sobol anteriores. Os hipercubos latinos devem ser reconstruídos se mais pontos forem necessários. Santner et al. (2003, page 160) sugerem que se for necessário mais informação sobre os parametros de correlação, então devido a maior variabilidade entre as distâncias entre pontos, planejamentos baseados em sequências de Sobol podem ser preferíveis aos hipercubo latinos.

2.2 Análise de incerteza

Uma das tarefas mais comuns de usuários de simuladores é acessar a incerteza nos outputs do simulador que é induzida pela incerteza sob os inputs. Incerteza nos inputs é uma característica presente em muitas aplicações, uma vez que simuladores requerem muitos valores de inputs a serem especificados pelo usuário e o usuário pode estar incerto sobre o melhor valor ou o valor correto de para alguns ou todos os inputs. No caso onde temos um simulador univariado $\eta : \mathcal{X} \to \mathcal{Y}$, análise de incerteza é o processo de en-



(a) (b)

Figura 2.1: Planejamentos bidimensionais de tamanho 50 usando: (a) Maximin hipercubo latinos, onde 10000 hipercubos latinos foram gerados e aquele com a maior distância mínima foi escolhido; (b) sequência de Sobol.

contrar a distribuição de $Y = \eta(X)$ induzida pela incerteza a respeito sob X. Incerteza nos inputs é caracterizada pela distribuição de probabilidade (conjunta), $\pi(X)$. Essa distribuição induz a distribuição de probabilidade (conjuta) para os outputs, $\pi(Y)$, e a análise de incerteza envolve identificar essa distribuição. Para mais detalhes e exemplos veja Oakley and O'Hagan (2002).

A distribuição a priori para X é importante na análise de incerteza, e deve ser cuidadosamente escolhida para representar a verdadeira incerteza do especialista a respeito dos inputs desconhecidos (a técnica básica de espeficificar $\pi(X)$ é conhecida como elicitação, veja O'Hagan et al. (2006) para uma introdução mais profunda). Isso pode ser feito considerando um intervalo de valores aceitáveis para X e então ajustando uma distribuição paramétrica, ou isso pode ser feito pegando o resultado de uma calibração do simulador dado algumas observações do sistema (veja seção 3.2). Um ponto importante para se ter em mente é que variáveis do model não nescessariamente coincidem com seus valores físicos se o simulador é uma representação do sistema físico que ele representa. Por exemplo, o que o modelo chama de viscosidade pode não representar a viscosidade física de um fluído, porque aproximações feitas ao discretizar o sistema de equações associado pode introduzir erros no processo.

Tarefas específicas podem ser calcular a média e a variância da distribuição da incerteza do output. A média pode ser considerada como a melhor estimativa para o output levando em consideração a incerteza, en-

quanto a variância dá uma medida do tamanho dessa incerteza. Em alguma aplicações é importante calcular a probabilidade do output ser maior ou menor que um certo limiar, i.e. $\mathbb{P}(\eta(X) > c)$. Por exemplo, qual é a probabilidade que a temperatura média global cresça mais que 4^oC de 2010 até 2100?

Uma maneira simples de calcular a distribuição de incerteza e funções dela é usando um método de Monte Carlo. Sorteie configurações aleatórias dos inputs da distribuição de incerteza dos inputs, então rode o simulador para cada configuração. O conjunto de outputs obtido corresponde a uma amostra aleatória da distribuição de incerteza do output.

Análise de incerteza via Monte Carlo

1. Para $i = 1, \ldots, N$, sorteie da priori amostras de inputs: $X_i \sim \pi(x)$.
2. Simule $Y_i = \eta(X_i)$.
3. Aproxime a distribuição de incerteza $\pi(y)$ por

$$\hat{\pi}(y) = \frac{1}{N} \sum_{i=1}^{N} \delta_{Y_i}(y)$$

   onde $\delta(\cdot)$ é a função delta Dirac (função indicadora).

4. Use $\hat{\pi}(y)$ para calcular qualquer quantidade de interesse, e.g., média

$$\mathbb{E}[Y] \approx \int y\hat{\pi}(y)\mathrm{dy} = \frac{1}{\mathrm{N}} \sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{Y_i}$$

Se uma amostra suficientemente grande for obtida, então a distribuição de incerteza e funções dela (e.g. média, variância, probabilidades, etc.) podem ser calculadas com qualquer grau de precisão. Entretanto, isso é frequentemente impraticável pois o simulador pode demorar muito para rodar. Métods de Monte Carlo requerem de 1.000 a 10.000 iterações para obter estimativas precisas para as medidas de incerteza (mais se $\dim(\mathcal{X})$ é grande), e se uma simples rodada do simulador dura mais que alguns segundos o tempo computacional da análise de incerteza pode ser muito alto.

Para o caso de simuladores determinísticos, O'Hagan et al. (1998), O'Hagan and Haylock (1997), e Haylock and O'Hagan (1996) usam o conceito de emuladores introduzidos no capítulo 1 para permitir que a análise de incerteza seja realisada de forma eficiente. Intuitivamente a idéia é simples:

Análise de incerteza usando emulador

1. Escolha um planejamento $X = \{x_1, \ldots, x_n\}$ baseado na priori $\pi(X)$.
2. Use o simulador para encontrar $Y = \{y_1 = \eta(x_1), \ldots, y_n = \eta(x_n)\}$.
3. Use (X, Y) para construir o emulador (Capítulo 2).
4. Use o emulador para fazer a análise de incerteza.

Alguns comentários

- O planejamento X deve ser escolhido levando em consideração a distribuição a priori para X, então a amplitude de X deve ser próxima ao suporte de $\pi(x)$ (note que essa regra exclue o uso de prioris impróprias para X).
- Se $\pi(X)$ é uma distribuição unimodal, então pode ser benéfico colocar mais pontos em áreas de alta probabilidade para que o emulador funcione melhor nessas áreas.
- Se um emulador gaussiano é usado, então a distribuição de incerteza resultante, $\pi(y)$, automaticamente considera na análise a incerteza introduzida pelo emulador.
- Para certas distribuições a priori, cálculos analíticos são possíveis para algumas quantidades no passo 4 (Haylock and O'Hagan (1996)), permitindonos evitar o uso de Monte Carlo.

2.3 Análise de sensitividade

Esta seção foi principalmente retirada do toolkit do MUCM. Veja
`http://mucm.aston.ac.uk/MUCM/MUCMToolkit/index.php?page=ThreadTopicSensitivityAnalysis.html`
para mais detalhes.

As várias formas de análise de sensitividade (AS) são ferramentas para estudar a relação entre os inputs e outputs do simulador (Saltelli et al. 2000). Em particular, AS é o processo de estudar quanto o output do simulador é sensitivo a mudanças em cada um dos inputs (e algumas vezes a combinac cões destes inputs) (Saltelli et al. 2000). O objetivo é dividir a variabilidade nos outputs para diferentes fontes de variação nos inputs. Análises de sensitividade são usadas para entender o comportamento do simulador, para

identificar quais inputs the mais influência nos outputs, e para aprender mais sobre a incerteza dos inputs. Ela pode ser usada como um prelúdio para simplificar o simulador, ou para construir um emulador simplificado na qual o número de inputs é reduzido.

Existem vários usos para AS. Aqui nós identificamos os quatro principais:

- **Entendimento:** Técnicas para mostrar como o output $\eta(x)$ se comporta a medida que variamos um ou mais de um dos inputs x são uma ajuda para entendermos a respeito de η. Elas podem agir como 'face validity', no sentido de que se o simulador está respondendo de forma inesperada a mudanças nos inputs, ou se alguns inputs inesperadamente parecem ter forte influência, então existem erros no modelo matemático ou erro de programação.

- **Redução de dimensão:** Se a análise de sensitividade mostrar que alguns inputs tem efeito negligenciável nos outputs, então isso abre a possibilidade de simplificação de η fixando valores para esses inputs. Embora isso não simplifique o simulador no sendido de torná-lo mais rápido ou mais fácil de avaliar $\eta(x)$ em algum particular x, a AS reduz a dimensão do espaço de inputs. Isso torna o simulador mais simples para se entender e usar. Isso é particularmente útil para simuladores com grande número de inputs. Pois executar uma análise de sensitividade em tal simulador pode ser altamente caro computacionalmente. Em geral, somente um número pequeno de inputs tem impacto significativo nos outputs, principalmente quando estamos interessados no comportamento do simulador sob uma região relativamente pequena no espaço dos inputs.

- **Analisando a incerteza dos outputs:** Onde existe incerteza nos inputs, existe incerteza nos outputs $\eta(x)$; quantificar essa incerteza nos outputs é tarefa da análise de incerteza, mas existe o interesse em saber o quanto da incerteza total é atribuída para um particular input ou para grupos de inputs. Em particular, o esforço para reduzir a incerteza dos outputs pode ser gasto de forma eficiente se focado naqueles inputs que mais influenciam o output.

- **Análise de decisão incluindo incerteza:** De forma geral, incerteza sobre os outputs do simulador é mais relevante quando aqueles outputs serão usados em processos de tomada de decisão (por exemplo, usando um simulador climático para fixar metas para a redução das emissões de dióxido de carbono). Novamente é útil quantificar o quanto da

> incerteza total para a tomada de decisão é atribuída a alguns particulares inputs. O esforço prioritário da pesquisa para reduzir a incerteza depende da influência dos inputs, suas atuais incertezas e da natureza da decisão.

Existem várias abordagens diferentes para a análise de sensitividade, a maioria delas é caracterizada por métodos locais ou globais. Nas duas abordagens, é comum especificar um conjunto base de inputs para ver como os outputs são afetados quando os inputs são mudados de seus valores base.

Análise de sensitividade local

A análise de sensitividade surge como resposta a preocupação com as consequências de má especificação para os valores dos inputs de um simulador. O usuário do simulador pode fornecer valores de $\hat{x}$ que podem ser considerados como as melhores estimativas para os inputs, logo $\eta(\hat{x})$ seria de alguma forma considerada a melhor estimativa para o output do simulador. Entretanto, se o valor correto para os inputs for diferente de $\hat{x}$ então o valor correto para o output será diferente de $\eta(\hat{x})$. O output seria considerado sensitivo a um particular input x_j se o output mudasse consideravelmnete quando x_j fosse pouco perturbado. Isso leva para a idéia de medir a sensitividade pela diferenciação da função. A medida de sensitividade a x_j é a derivada $\partial\eta(x)/\partial x_j$, avaliada no ponto $x = \hat{x}$.

AS baseada em deferenciação tem algumas deficiências. A medida de diferenciabilidade somente mede o impacto de mudanças infinitesimais em x_j, e por essa razão esse tipo de AS é referido como AS local. Se a resposta do output com respeito a x_j é não linear, então perturbando x_j mais que um valor pequeno pode ter um efeito que não é bem representado pela derivada.

Um problema mais sério é que a diferenciação não é invariante a unidades de medida. Se, por exemplo, nós escolhemos medir x_j em kilometros ao invés de metros, então $\partial\eta(x)/\partial x_j$ mudará em um fator de 1000. O output vai parecer ser mais sensitivo ao input x_j do que ao input x_j' porque a derivada avaliada em $\hat{x}$ é maior, mas essa ordenação pode facilmente ser invertida se nós mudassemos a escala de medida. Por estas razões, nós preferimos usar métodos globais.

Análise de sensitividade global

Na análise de sensitividade global (Saltelli et al. 2008) nós consideramos como os outputs mudam quando a variação nos inputs é grandes, refletindo de alguma forma a amplitude dos valores de interesse para aqueles inputs.

AS global é mais complexa que AS local, porque existem diferentes maneiras de medir o efeito nos outputs, e diferentes maneiras de definir como os inputs são perturbados. Análises de sensitividade global são também influenciadas pelo tamanho da variação considerada para os inputs. Nós precisamos de uma razão bem definida para a escolha das amplitudes, caso contrário as medidas de sensitividade serão arbitrárias assim como a análise de sensitividade local responde para uma escala arbitrária das medidas. Aqui, nós escolhemos descrever o caso considerado na Seção 2.2 e suponha que os inputs variem de acordo com a distribuição $\pi(x)$. Isso é conhecido como *análise de sensitividade probabilística* porque a quantidade de perturbação é definida por uma distribuição de probabilidade sobre os inputs. A interpretação usual dessa distribuição é como uma medida de incerteza que nós temos sobre o melhor valor ou valor 'verdadeiro' para esses inputs.

2.3.1 Análise de sensitividade probabilística

Análise de sensitividade probabilística requer que uma distribuição seja atribuída para os inputs do simulador. Nós, primeiramente, vamos apresentar a notação e em seguida vamos discutir a interpretação da distribuição de probabilidade e das medidas relevantes para análise de sensitividade probabilísticas. O foco da AS é a relação entre x e o output do simulador $\eta(x)$. Uma vez que a AS também tenta isolar a influência de inputs individuais, ou grupo de inputs, nos outputs, nós definimos x_j como sendo o j-ésimo elemento de x e vamos referir a ele como o j-ésimo input, para $j = 1, 2, \ldots, p$, onde p é o número de inputs. Se J é um subconjunto de índices $\{1, 2, \ldots, p\}$, então x_J denota o correspondente subconjunto de inputs.

Nós denotamos a distribuição sobre os inputs x por π. Formalmente, $\pi(x)$ é a função de densidade conjunta de probabilidade para todos os inputs. A função de densidade marginal para o input x_j é denotada por $\pi_J(x_J)$. A distribuição condicional de x_{-J} dado x_J é $\pi_{-J|J}(x_{-J} \,|\, x_J)$. Note que é natural a distribuição π ser tal que os vários inputs são estatisticamente independentes. Neste caso, a distribuição condicional $\pi_{-J|J}$ não depende de x_J e é identica a distribuição marginal π_{-J}. Note que ao atribuir distribuições de probabilidade para os inputs em AS probabilística nós formalmente tratamos estes inputs como variáveis aleatórias. É uma convenção na estatística denotar variáveis aleatórias por letras maiúsculas, e esta distinção é útil também em AS probabilística. Logo, o símbolo X denota o conjunto de inputs como aleatórios (i.e. incertos), enquanto x continua a denotar um particular conjunto de valores dos input. Similarmente, X_J representa aqueles inputs aleatórios sub escritos no conjunto J, enquanto x_J

denota o valor destes inputs.

É mais natural pensar em X como uma variável aleatória no contexto do terceiro e quarto uso da AS listados acima. Quando realmente existe incerteza a respeito dos valores a serem atribuídos aos inputs para obter os output de interesse, então X pode de fato ser interpretado como aleatório, e $\pi(x)$ é portanto a função de densidade que descreve a probabilidades associadas aos possiveis valores x de X. AS envolve entender o efeito da incerteza a respeito dos inputs na incerteza induzida involvendo os outputs $\eta(X)$.

Entretando, no contexto dos outros usos da AS pode ser menos natural pensar X como aleatório. Quando nosso objetivo é entender o comportamento dos simuladores ou identificar inputs que são mais ou menos relevantes, não é necessário tratar os inputs como incertos. De qualquer forma, é importante pensar na amplitude de valores dos inputs sob os quais desejamos obter entendimento ou redução de dimensão. Nesse caso, $\pi(x)$ é definida como sendo zero fora da amplitude de interesse, e como uma função de pesos dentro da amplitude de interesse. Nós poderíamos definir o mesmo peso para todos valores dentro do intervalo de amplitude de interesse, mas em alguns casos pode ser mais apropriado da mais peso para alguns valores do que para outros. Independente de defirnir $\pi(x)$ como uma função de densidade ou como uma função de pesos, nós podemos tomar médias sobre a região de interesse para definir medidas de incerteza.

Existem várias medidas de sensitividade diferentes. Nós vamos considerar duas, conhecidas como efeitos principais e efeitos de intereção.

Efeitos principais

O efeito principal de um input X_i é a função

$$I_i(x_i) = \mathbb{E}(\eta(X) \,|\, X_i = x_i) - \mathbb{E}\eta(X). \tag{2.3}$$

Em outras palavras, é o valor esperado do output quando o i-ésimo input é fixado menos a média. Se I_i é aproximadamente constante para a amplitude de valores aceitáveis para x_i, isso sugere que x_i não é um input importante em termos de controlar a variabilidade do output. De forma geral, podemos definir o efeito principal de um grupo de inputs como $I_J(x_J) = \mathrm{E}(f(X) \,|\, x_J) - \mathbb{E}f(X)$. O efeito principal de x_J é uma função de x_J, e portanto é mais complexa que o efeito principal de um único input. Entretanto, usando o efeito de grupos de inputs pode ser revelado mais sobre a estrutura do simulador do que usando efeitos principais dos inputs no grupo.

Efeitos de intereção

A interação entre inputs X_i e X_j é a função

$$I_{\{i,j\}}(x_i, x_j) = \mathbb{E}(\eta(X) \mid X_i = x_i, X_j = x_j) - I_i(x_i) - I_j(x_j) - \mathbb{E}\eta(X). \quad (2.4)$$

Essa equação representa o desvio no efeito conjunto ao variar dois inputs após subtrair seus efeitos principais.

Para entender porque esse termo pode ser de interesse, considere o efeito médio de mudar x_1 de x_1^0 para x_1^1, que nós denotamos por $a_1 = I_1(x_1^1) - I_1(x_1^0)$. Similarmente, seja o efeito médio de mudar x_2 de x_2^0 para x_2^1 denotado por $a_2 = I_2(x_2^1) - I_2(x_2^0)$. Agora considere mudar ambos inputs, de (x_1^0, x_2^0) para (x_1^1, x_2^1). Para um simples e suave simulador, nós podemos pensar que o efeito médio de tal mudança possa ser $a_1 + a_2$. No entanto, isso geralmente não é o caso. A verdadeira mudança seria

$$\mathbb{E}[\eta(X)|x_{12}^1] - \mathbb{E}[\eta(X)|x_{12}^0],$$

e quando isso é diferente de $a_1 + a_2$ é dito existir uma interação entre x_1 e x_2, onde nós definimos a interação usando a equação (2.4).

Comentários

Estas definições merecem explicações adicionais. Primeiro, $\mathbb{E}\eta(X)$ é a incerteza média, que é uma das quantidades tipicamente calculadas na análise de incerteza. Para os nossos propósitos ela é um ponto de referência. Se nós não especificarmos os valores de algum dos inputs, então $\mathbb{E}\eta(X)$ é a estimativa natural para $\eta(X)$.

Se, no entanto, nós especificarmos o valor de x_j, então a estimativa natural para $\eta(X)$ se torna $\mathbb{E}\eta(X)$ mais um desvio $I_j(x_j)$ do ponto de referência. Nós chamamos $I_j(x_j)$ o efeito principal de x_j.

Agora se nós especificarmos ambos x_j e $x_{j'}$, a equação (2.4) expressa a estimativa natural $\mathbb{E}(\eta(X) \mid X_i = x_i, X_j = x_j)$ como a soma de (a) o ponto de referência $\mathbb{E}\eta(X)$, (b) os dois efeitos principais $I_j(x_j)$ and $I_{j'}(x_{j'})$, e (c) o efeito de interação $I_{\{j,j'\}}(x_{\{j,j'\}})$.

Essa abordagem se extende naturalmente para definir interações de três ou mais inputs, e nós sugerimos aos leitores interessados o MUCM toolkit para mais detalhes.

Gráficos dos efeitos principais e funções do efeito de interação pode fornecer uma análise detalhada de como os outputs respondem aos inputs. Entretanto, na maioria das vezes estamos interessados em uma única figura

contendo medidas de sensitividade do output com respeito a inputs individuais ou combinações de inputs. Será que aquele particular input tem um efeito 'grande' no output? Nós consideramos uma maneira de fazer isso, chamada de análise de sensitividade basead na variância (o toolkit do MUCM contém uma discussão interessante a respeito de decisões baseadas em AS).

2.3.2 Análise de sensitividade baseada na variância

Análise de sensitividade baseada na variância é uma forma de análise de sensitividade probabilística na qual o impacto de um input ou grupo de inputs no output do simulador é medido através da variância e reduções na variância. Nós medimos a sensitividade de um particular input X_i pela quantidade de variância do output que é atribuída a aquele particular input. Nós começamos considerando a variância $\mathbb{V}\text{ar}[\eta(X)]$, que representa o total de incerteza sobre o output induzida pela incerteza dos inputs. No caso de múltiplos outputs isso seria a matriz de variâncias, cujos elementos da diagonal equivalem a sensitividade de cada dimensão do output, e as covariâncias correspondem a sensitividades conjuntas. Essa variância também surge como uma medida global de incerteza na análise de incerteza.

Variância da sensitividade

Se nós removêssemos a incerteza em X_i, nós esperaríamos que a incerteza em $\eta(X)$ reduzisse. A quantidade dessa redução esperada é

$$V_i = \mathbb{V}\text{ar}(f(X)) - \mathbb{E}[\mathbb{V}\text{ar}(f(X) \mid X_i)]. \tag{2.5}$$

Note que o segundo termo da expressão involve primeiro encontrar a variancia condicional de $\eta(X)$ dada que X_i assume algum valor x_i, que seria a incerteza que teríamos a respeito do simulator se estivéssemos certo a respeito do valor de X_i. Logo, essa seria a redução de incerteza no output. Mas nós estamos incertos a respeito de X_i, então nós tomamos a esperança da variância condicional com respeito a nossa incerteza sobre X_i. Logo, V_i é definido como a **variância da sensitividade** para o i-ésimo input. A fórmula da decomposição condicional da variância nos dá uma expressão diferente para V_i

$$V_i = \mathbb{V}\text{ar}[\mathbb{E}(f(X)|X_i)].$$

Referindo a equação (2.3) como o efeito principal $I_i(x_i)$, podemos ver que V_i é a variância do efeito principal. Isso é facilmente generalisado para um conjunto de índice J, e a variância da sensitividade para aquele conjunto de índices V_J.

A variância da sensitividade é frequentemente expressada como uma proporção da variância total $V = \mathbb{V}\text{ar}[\eta(X)]$. A razão $S_i = V_i/V$ é referida como o **índice de sensitividade** para o i-ésimo input, e os índices de sensitividade para interações são similarmente conhecidos (também chamados de índices de Sobol). Então um índice de 0.5 indica que a incerteza a respeito de x_J está associada a metade de toda incerteza associada ao output devido a incerteza em x. (Esse índice é frequentemente multiplicado por 100 e expresso como percentagem.)

Variância de interação

Nós podemos definir similarmente a variância de interação $V_{\{i,j\}}$ dos inputs X_i e X_j como a variância do efeito de interação $I_{\{i,j\}}(x_i, x_j)$. Quando a distribuição de probabilidade dos inputs é tal que eles são todos independentes, então pode ser mostrado que os efeitos principais e interações (incluindo interações de ordem altas involvendo três ou mais inputs) somados resultam na variância total $\mathbb{V}\text{ar}[f(X)]$, i.e.,

$$\mathbb{V}\text{ar}(f(X)) = \sum_{i=1}^{d} V_i + \sum_{i<j} V_{i,j} + \sum_{i<j<k} V_{i,j,k} + \ldots V_{1,2,\ldots,d}$$

onde d é a dimensão de x (Oakley and O'Hagan 2004).

Índice de sensitividade total

Outro índice de sensitividade para o i-ésimo input que é algumas vezes usado é o índice de sensitividade total:

$$T_i = \frac{\mathbb{E}[\mathbb{V}\text{ar}(\eta(X) \mid X_{-i})]}{V},$$

onde X_{-i} significa todos os inputs exceto X_i. Isso é a proporção esperada da incerteza no output model que seria deixada se removêssemos a incerteza em todos inputs exceto o i-ésimo. No caso de inputs independentes, pode-se mostar que T_i é a soma dos efeitos principais S_i e os índices das interações para todas interações involvendo X_i e um ou mais inputs, logo o uso do nome índice de sensitividade total.

2.4 Emuladores com mútiplos outputs

No capítulo 1 nós mostramos como construir um emulador de um simulador com um único output. Na prática, simuladores quase sempre produzem

múltiplos outputs, e nós frequentemente estamos interessados em responder questões relativas a mais de um output. Nessa seção, nós descrevemos vários métodos para emular simuladores com múltiplos outputs.

Uma das tarefas mais comuns é a previsão de múltiplos outputs de um simulador para inputs não observados. Isso pode ser considerado como uma questão de prever cada output separadamente, reduzindo o problema a vários problemas individuais explorados no Capítulo 1. Entretanto, previsão inevitavelmente involve incerteza e ao prever vários outputs nós estamos preocupados com a incerteza conjunta, representada pelas variâncias e covariâncias. Se nós prevemos cada output separademente, nós estamos ignorando a covariância entre outputs. Isso pode causar uma super estimação significativa de nossa incerteza, e o problema se torna mais sério a medida que a correlação entre outputs aumenta. Se os outputs forem não correlacionados (ou independentes) então usando emuladores separados teremos bons resultados.

Em geral, nós tratamos questões de simuladores com múltiplos outputs usando a mesma metodologia básica para simuladores com um único output (ou seja, nós emulamos os vários outputs e então usamos os resultados do emulador para responder as questões relevantes ao simulador). Mas existem várias alternativas nas quais podemos emular simulador com múltiplos outputs. Nós descrevemos algumas dessas alternativas abaixo.

2.4.1 Ouputs independentes

Nós dissemos que ao prever vários outputs é necessário considerar as covariâncias entre outputs, e que ao simplesmente emular os outputs individualmente usando os métodos do Capítulo 1 ignora estas covariâncias. Entretanto, em alguns casos as covariâncias podem não ser tão importantes. Suponha que nós somos capazes de emular todos os outputs de interesse de forma precisa usando emuladores do Capítulo 1, de tal forma que a incerteza em cada caso seja bem pequena. Então nós iremos saber que o valor verdadeiro de quaiquer dois outputs do simulador será muito próximo aos valores médios dos emuladores. Sabendo também a covariância permite-nos acessar a distribuição conjunta de probabilidades associada aos dois ouputs, mas quando a incerteza é muito pequena essa informação extra é de pouca importância.

Pode-se argumentar que se nós tivermos dados de treinamento suficientes para emular todos os outputs com alta precisão, no sentido de que a incerteza é pequena o suficiente para ignorá-la, então nós podemos ignorar covariâncias entre outputs. É verdade que quanto mais precisamente pode-

mos emular os outputs menos importância tem as nossas escolhas para a emulação, e assim com grandes amostras treinamento todas alternativas consideradas aqui darão essencialmente os mesmos resultados. Entretando, a realidade com simuladores complexos é que raramente nós estamos nessa posição feliz de sermos capazes de rodar o simulador para dados de treinamento um número suficiente de vezes até obter a precisação desejada, principalmente se os espaços dos inputs e dos outputs são de alta dimensão. Então quando a incerteza do emulador é importante, covariâncias entre outputs são necessárias e a escolha entre formas alternativas de emulação de múltiplos outputs importa.

Uma vantagem importante no uso de emuladores independentes para cada output é que podemos usar diferentes funções para a média e diferentes funções de correlação para cada output, e em particular, diferentes outputs podem ter diferentes tamnhos de correlação δ.

2.4.2 Outputs como dimensão extra nos inputs

Em alguns simuladores, os outputs representam valores de alguma propriedade do mundo real em diferentes localizações e/ou tempos (veja Seção 2.5). Por exemplo, um simulador de um acidente de carro pode ter outputs como velocidades e forças como uma série temporal observada em vários instantes de tempo durante o acidente simulado. Como um outro exemplo, um simulador de um derramamento de petróleo no oceano tem como output a concentração de petróleo em diferentes localizações geográficas no oceano e em diferentes instantes de tempo.

Em tais casos, uma alternativa é tratar os múltiplos outputs como um único output mas com um ou mais inputs extras. Essa abordagem é claramente muito simples, mas nós precisamos considerar quais restrições serão implicadas na estrutura de covariância em termos dos outputs originais. Em teoria, não precisa-se impor restrição alguma, pois a princípio a função de covariância de um emulador com os novos inputs pode ser definida para representar qualquer estrutura de covariância conjunta. Por outro lado, na prática as estruturas de covariâncias geralmente assumem restrições bastante fortes. Como discutido no Capítulo 1, nós invariavelmente assumimos variância constante, que implica que todos os ouputs originais têm a mesma variância. Isso pode ser razoável, mas frequentemente não é. Além disso, as escolhas usuais para a função de correlação faz fortes suposições. Por exemplo, a função de correlação gaussiana e exponencial potência têm o efeito de (a) assumir separabilidade entre inputs originais e outputs, e (b) assumir uma forma de correlação entre outputs originais que é improvável de ser

apropriada, como por exemplo, o caso de uma série temporal.

Logo a grande simplicidade de abordagem de adicionar inputs é equilibrada pelas suposições restritivas que podem levar a uma emulação pobre (ou pode requerer um número significativamente maior de amostras de treinamento para obter uma emulação adequada).

2.4.3 Processos gaussiano multivariados

O processo gaussiano univariado (PG) é uma distribuição de probabilidade para a função $\eta(x)$, na qual a distribuição conjunta de qualquer conjunto de pontos desta função é normal multivariada. O PG multivariado é uma extensão do PG univariado para o caso onde $\eta(x)$ não é uma escalar mas um vetor. Se o simulador tem r ouputs então $\eta(x)$ é $r \times 1$.

Formalmente, o PG multivariado é um modelo probabilístico sobre funções com valores multivariados. Ele é caracterizado por uma função média $m(\cdot) = \mathrm{E}[\eta(\cdot)]$ e uma função de covariância $v(\cdot,\cdot) = \mathrm{Cov}[\eta(\cdot), \eta(\cdot)]$ (Conti and O'Hagan to appear 2010). Sob esse model, a função avaliada no input x tem uma normal multivariada $\eta(x) \sim N\left(m(x), v(x,x)\right)$, onde

- $m(x)$ é o vetor $r \times 1$ de médias de $\eta(x)$
- $v(x,x)$ é a matriz de variância $r \times r$ de $\eta(x)$.

Além disso, a conjunta de $\eta(x)$ e $\eta(x')$ também é normal multivariada:

$$\begin{pmatrix} \eta(x) \\ \eta(x') \end{pmatrix} \sim N\left(\begin{pmatrix} m(x) \\ m(x') \end{pmatrix}, \begin{pmatrix} v(x,x) & v(x,x') \\ v(x',x) & v(x',x') \end{pmatrix} \right)$$

onde $v(x,x')$ é a matriz de variâncias $r \times r$ entre os elementos $\eta(x)$ and $\eta(x')$.

Função de médias multivariada

A função de médias para um conjunto de outputs é uma função vetor. Seus elementos são funções de médias para cada output, definindo a expectativa a priori para a forma da reposta dos outputs para os inputs. Se a forma linear é escolhida com o mesmo vetor $q \times 1$ $h(\cdot)$ de funções base para cada output, então o vetor da função de médias pode ser simplesmente expresso por $\beta^T h(\cdot)$, onde β é uma matriz $q \times r$ de coeficientes de regressão e r é o número de outputs.

As seguintes duas escolhas de regressores são comuns e convenientes em muitos casos:

- Para $q = 1$ e $h(x) = 1$, a função de médias se torna $m(x) = \beta$, onde β é um vetor r-dimensional de hiperparâmetros representando uma média global desconhecida para os ouputs do simulador.

- Para $h(x)^T = (1, x)$, tal que $q = 1 + p$, a função de médias é um vetor r-dimensional cujo j-ésimo elemento é um escalar: $\{m(x)\}_j = \beta_{1,j} + \beta_{2,j}\, x_1 + \ldots + \beta_{1+p,j}\, x_p$.

Essa forma linear da função de médias na qual todos outputs têm as mesmas funções base é frequentemente usada em emuladores com múltiplos outputs, onde é a escolha natural para os processos gaussianos multivariados e têm vantagens matemáticas e computacionais. No entanto, se tomarmos cuidado e tivermos o esforço necessário para construir uma boa função de médias, nós podemos eliminar muitos dos problemas encontrados ao escolher funções de covariancia. Veja Rougier et al. (2009) para um exemplo detalhado na construção de uma função de médias complexa.

Função de covariância multivariada

Em geral, o PG multivariado é caracterizado pelo fato de que se empilhássemos os vetores $\eta(x_1), \eta(x_2), \ldots, \eta(x_n)$ em um vetor com rn elementos, então este vetor segue uma distribuição normal multivariada. A única restrição formal na função de covariancia $v(.,.)$ é que a matriz de variância $rn \times rn$ para qualquer conjunto arbitrário com n outputs deve ser sempre não-negativa definida. O PG multivariado é portanto muito flexível. Entretando, encontrar uma função de covariância não-negativa apropriada é difícil, e na prática a maioria das pessoas usam formas mais restritivas para a função de covariância.

Uma simplificação é que os outputs são **separáveis** (O'Hagan 1998), o que implica que todos os outputs têm a mesma função de correlação $c(x, x')$ no espaço dos inputs, e que a função de covariância assuma a forma $\mathrm{Cov}[f_i(x), f_j(x')] = \sigma_{ij} c(x, x')$. É usual permitir que a matriz de covariância entre outputs Σ com elementos σ_{ij} seja arbitrária, embora possa assumir alguma estrutura específica. Um caso particularmente simples é onde Σ é assumido ser uma matriz diagonal, o que equivale a ter emuladores independentes para cada output, com todos tendo a mesma estrutura de correlação sobre o espaço de inputs. Isso assume ambos a separabilidade e outputs não correlacionados, e portanto pode ser restritivo demais para ser utilizada na prática. Uma estrutura mais útil para Σ poderia ser aquela na forma implicada por um model de série temporal apropriado (p.e. um modelo

auto-regressivo de primeira ordem) no caso onde os outputs representem pontos diferentes no tempo.

Ambos separabilidade (entre inputs e outputs) e independência (entre outputs) são limitações que podem restringir a habilidade do emulador resultante representar precisamente os vários outputs do simulador. Mas ambos levam a emuladores que são relativamente fáceis de construir e usar. Relaxando as restrições irá levar a métodos mais complexos, então a escolha entre alternativas é parcialmente baseada em crenças sobre o simulador mas parcialmente baseado no equilíbrio entre complexidade e realismo.

Um aspecto dessa propriedade é que quando empilhamos os vetores $\eta(x_1), \ldots, \eta(x_n)$ em um vetor $rn \times 1$ sua matriz de covariância tem a forma do produto de Kronecker $\Sigma \otimes c(D, D)$, onde $c(D, D)$ é a matriz de correlação entre inputs com elementos $c(x_i, x_j)$. Um outro aspecto é que se ao invés de empilhar os vetores de outputs nós formássemos a matriz $r \times n$, denotada $\eta(D)$, então $\eta(D)$ segue uma distribuição normal matriz-variada com matriz de médias $m(D)$, covariancia entre linhas Σ e covariância entre colunas $c(D, D)$. Rougier (2008) mostra como esta estrutura pode ser explorada para ter ganhos computacionais.

Como discutido anteriormente nesta seção, a suposição de separabilidade impõe uma restrição que todos os outputs tenham a mesma função de correlação no espaço de inputs. Essa restrição leva a metodologia de emulação mais simples, pois nós construímos somente um emulador e seu desenvolvimento é similar ao emulador para um único output com poucas modificações. Entratanto, é consideravelmente mais restritivo com respeito a função de correlação do que emuladores independentes para cada output onde cada output tem a sua própria função de correlação (e em particular seus próprios alcances de correlação).

Vale a pena notar que outputs que são altamente correlacionados terão necessariamente funções de correlação similares. Então, na situação onde emuladores independentes são inapropriados porque esperamos os outputs serem fortemente correlacionados, o emulador com multi-outputs é mais apropriado pois a suposição de separabilidade é mais provável de ser razoável.

Uma abordagem híbrida é a de quebrar os outputs em grupos tais que os grupos sejam aproximadamente independentes mas correlação dentro dos grupos seja considerada. Então, nós emulamos cada grupo separadamente usando um dos métodos discutidos aqui.

2.4.4 Métodos de transformação

Outra abordagem é a de transformar os outputs de forma que os outputs transformados podem ser considerados independentes (ou não correlacionados), e podem ser emulados usando emuladores independentes como discutido acima. Se tal transformação pode ser encontrada, podemos proceder como descrito na Seção 2.4.1. As únicas questões ainda por responder nessa abordagem são (a) como encontrar uma transformação apropriada e (b) como acomodar incerteza na escolha da transformação.

As transformações mais simples a considerar são as lineares. Nós representamos os outputs originais como funções lineares de um número de outputs latents que são modelados como processos gaussianos independentes. Isso é conhecido na literatura de geoestatística como modelo de corregionalização (LMC) (Wackernagel 1995). Precisamos ter pelo menos tantos outputs latentes quanto outputs originais de interesse (veja discussão a seguir sobre redução de dimensão), mas podemos ter mais. Na abordagem do LMC é usual estimar todos os processos gaussianos como um exercício de estimação conjunto.

Podemos também considerar transformações lineares nas quais temos menos outputs latentes do que outputs originais. Essa possibilidade surge naturalmente quando usamos componentes principais (Seção 2.5.1), já que a essência desse método é encontrar um conjunto de outputs latentes tal que o máximo da variabilidade nos outputs originais seja explicado por poucos outputs latentes. Se os outputs latentes restantes são muito insensíveis a variações nos inputs do simulador (num intervalo explorado na amostra de treinamento), então nós podemos considerar que existe muito pouco valor em emulá-los. Na prática, não haverá garantia que componentes com menor variância serão menos estruturadas, e métodos para procurar por estruturas de correlação tais como variogramas e cross-variogramas para os outputs latentes podem ser usados para fornecer um diagnóstico qualitativo sobre a estabilidade desse modelo.

Formalmente, nós representamos os outputs latentes que não emulamos completamente como processos gaussianos com média zero e função de correlação que consiste somente de um nugget. O hiper-parâmetro de variância σ^2 é igualado em cada caso a varância estimada do componente principal.

Esse tipo de redução de dimensão pode ser uma maneira muito poderosa de administrar um emulador com um grande número de outputs, e permanece sendo o método usado frequentemente para lidar com outputs de alta dimensão. Essa forma de redução de dimensão é provável de ser efetiva quando espera-se forte correlação entre os outputs do simulador, como por

exemplo é usualmente o caso quando o output do simulador é composto de uma grade (ou outro tipo de divisão/projeção) do espaço ou espaço-tempo. De fato, usualmente existe redundância em outputs numa grade já que para o simulador ter uma resolução efetiva de uma característica espacial ele deve ser maior que 8-10 unidades da grade (dependendo dos detalhes do simulador), caso contrário, ele pode induzir a instabilidades numéricas no simulador, ou ser pobremente modelado.

A abordagem de componentes principais é explicada em detalhes na próxima seção, onde consideramos emulação de um modelo de circulação global.

2.5 Exemplo: modelo climático

Agora apresentamos uma discussão mais a fundo de um exemplo das ciências climáticas onde emuladores foram usados para fazer uma tarefa que teria sido difícil senão impossível sem o uso de emuladores. Essa descrição foi tirada de Wilkinson (2010) e mais detalhes podem ser encontrados em Ricciuto et al. (2010).

Nós usamos o modelo climático UVic ESCM (University of Victoria Earth System Climate Model) juntamente com vegetação dinâmica, ciclo de carbono terrestre e ciclo de carbono inorgânico do oceano (Meissner et al. 2003). O modelo foi construído para estudar potenciais efeitos do ciclo do carbono e para ver como afetam as provisões do clima futuro. Consideramos que o modelo tem apenas dois inputs, Q_{10} e K_c, e o output é uma série temporal de CO_2 atmosférico. O input Q_{10} controla a dependência da temperatura da respiração e pode ser considerado como controlando a fonte de carbono, enquanto K_c é a constante de Michaelis–Menton para o CO_2 e controla a sensitividade de fotossíntese e medidas de dióxido de carbono atmosférico. Nós separamos a análise em duas partes, apresentamos a teoria de componentes principais aqui, e descrevemos a calibração na Seção 3.2.

Cada rodada do modelo leva aproximadamente duas semanas de tempo computacional e tem um cenário de 47 rodadas com o qual as análises são realizadas. O output do modelo e as observações de campo são mostrados na Figura 2.2. Note que sem o uso de emuladores, somente uma análise rudimentar seria possível devido ao longo tempo para rodar o modelo.

2.5.1 Emulação por componentes principais

Nós tomamos uma direção similar a de Higdon et al. (2008) e usamos uma técnica de redução de dimensão para projetar o output do modelo com-

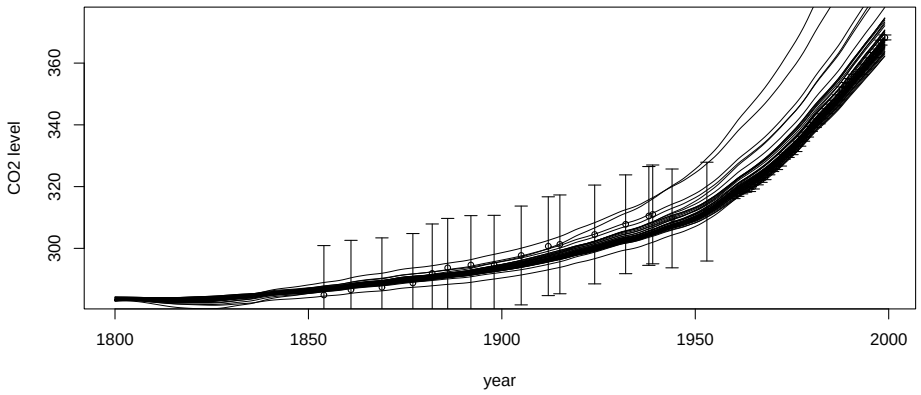


Figura 2.2: Cenário de 47 rodadas do modelo UVic para um planejamento nos dois imputs Q_{10} e K_c. O output (linha preta) é a previsão do CO_2 atmosférico para 1800-1999, e as 57 observações de campo são mostradas como círculos com barras de erro de dois desvios padrão.

putacional em um sub-espaço com um número menor de dimensões e então construímos emuladores do mapeamento do espaço dos inputs para o espaço reduzido dos outputs. A única exigência da redução de dimensão é que exista um método de reconstrução para o espaço original dos outputs. Nós usamos componentes principais aqui (também conhecido como método das funções ortogonais empíricas), já que assim garante-se que as projeções são ótimas, em termos de minimizar a média do erro de reconstrução, apesar de que poderíamos usar outra técnica de redução de dimensão desde que exista um método de reconstrução. Um gráfico esquemático dessa idéia é apresentado na Figura 2.3. O modelo computacional $\eta(\cdot)$ é uma função do espaço dos inputs Θ para o espaço dos outputs $\mathcal{Y}$. Análise de componentes principais fornece um mapeamento do espaço completo dos outputs $\mathcal{Y}$ para o espaço reduzido $\mathcal{Y}^{pc}$. Nós construímos emuladores gaussianos para mapear de Θ para $\mathcal{Y}^{pc}$ e então usamos o inverso da projeção original (também uma projeção linear) para sair de $\mathcal{Y}^{pc}$ para $\mathcal{Y}$. Isso gera um mapeamento computacionalmente barato do espaço dos inputs Θ para o espaço dos outputs $\mathcal{Y}$ que não usa o modelo $\eta(\cdot)$. Esse substituto barato, ou emulador, interpola aproximadamente todos os pontos no cenário (é aproximado devido ao erro de reconstrução da componente principal) e gera distribuições de probabilidade para o output do modelo para qualquer valor do input.

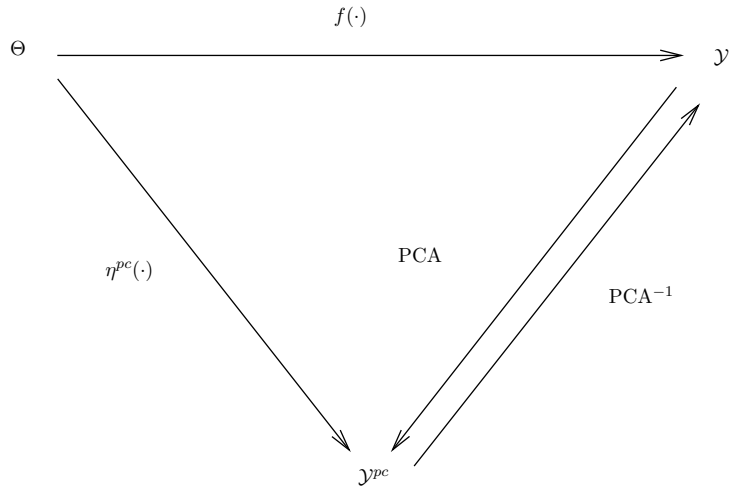


Figura 2.3: Gráfico esquemático da idéia por trás da emulação por componentes principais. Θ é o espaço dos inputs, $\mathcal{Y}$ o espaço dos outputs, e $\eta(\cdot)$ o modelo computacional. Seja $\eta^{pc}(\cdot)$ o emulador gaussiano de Θ para o sub-espaço principal $\mathcal{Y}^{pc}$.

Análise de componentes principais é uma projeção linear do dado em um sub-espaço de dimensão menor (o sub-espaço principal) tal que a variância do dado projetado é maximizada. É feita usualmente via decomposição de auto-valores da matriz de correlação, mas por razões de eficiência computacional, nós vamos usar a decomposição do valor singular. Seja Y uma matriz $N \times n$ onde na linha i temos a $i-$ésima rodada do modelo computacional, $Y_{i\cdot} = f(\theta_i)$ (lembre que o output do modelo tem dimensão n e que existem N rodadas no cenário $\mathcal{D}_{\text{sim}}$). O algoritmo de redução de dimensão pode ser descrito como:

1. Centralize a matriz. Seja μ o vetor linha de média das colunas, seja Y' a matriz Y com μ diminuido de cada linha ($Y' = Y - \mu$) de forma que a média de cada coluna de Y' é zero. Podemos escolher também escalonar a matriz, de forma que a variância de cada coluna seja um.

2. Calcule a decomposição de valor singular

$$Y' = U\Gamma V^*.$$

   V é uma matriz unitária $n \times n$ contendo as componentes principais (os auto-vetores) e V^* denota seu conjugado complexo transposto. Γ é uma matriz diagonal $N \times n$ contendo os valores principais (os auto-valores) e é ordenada de forma que a magnitude das entradas da diagonal decrescem ao longo da matriz. U é uma matriz $N \times N$ contendo os valores singulares restantes.

3. Decida a dimensão do sub-espaço principal, digamos n^* ($n^* < n$). Uma base ortonormal é dada pelas primeiras n^* colunas de V (os primeiros n^* auto-vetores) os quais denotamos V_1 (uma matriz $n \times n^*$). Seja V_2 a matriz contendo as colunas restantes de V.

4. Projete Y' no sub-espaço principal. As coordenadas do sub-espaço (os pesos dos fatores) são encontrados projetando em V_1:

$$Y^{pc} = Y'V_1.$$

   A i-ésima linha de Y^{pc} então denota a coordenada do i-ésimo membro do cenário no espaço $\mathcal{Y}^{pc}$.

Alguns comentários:

- Note que a análise de componentes principais é feita ao longo das colunas da matriz ao invés de ao longo das linhas como usual. O resultado é

que os auto-valores são da mesma dimensão do output original com os auto-valores mais importantes frequentemente tomando a forma geral do output.

- Apesar de análise de componentes principais (ACP) ser uma projeção linear, esse método pode ser usado em modelos altamente não lineares. O aspecto linear da ACP é a projeção do espaço dos outputs $\mathcal{Y}$ num espaço menor dos outputs $\mathcal{Y}^{pc}$. O mapeamento do espaço dos inputs Θ para os espço reduzido dos outputs $\mathcal{Y}^{pc}$ pode ainda ser não linear, e não será afetado pela suposição de linearidade usada na redução de dimensão. A principal exigência para o uso desse método, é que exista uma estrutura de covariância consistente entre os outputs. A projeção nas componentes principais está essencialmente assumindo que o verdadeiro número de graus de liberdade é menos que n. Se os outputs são todos independentes então a ACP não funcionará como uma técnica de redução de dimensão.

- Não há um método estabelecido para decidir a dimensão n^* do sub-espaço principal. A porcentagem de variância explicada (soma dos auto-valores correspondentes em Γ) é frequentemente usado como uma heurística, com o objetivo sendo explicar 95% ou 99% da variância. Devemos também decidir que componentes incluir em V_1. Pode-se dizer que componentes que somente explicam uma pequena porção da variância (pequenos auto-valores) são importantes preditivamente, como acontece na regressão com componentes principais (Jolliffe 2002). Um método para selecionar componentes é uso de gráficos de diagnóstico como explicado abaixo.

Isso nos deixa com as coordenadas do cenário do sub-espaço principal $\mathcal{Y}^{pc}$, com cada linha correspondendo a mesma linha do planejamento original D. Processos gaussianos podem ser usados para emular esse mapeamento. Usualmente, teremos $n^* > 1$, e então ainda precisaremos usar um emulador multivariado como o proposto por Rougier (2008). Porém, emular o mapeamento reduzido com n^* processos gaussianos independentes frequentemente tem a mesma performance de usar um emulador completamente multivariado, especialmente se o tamanho do cenário N é grande comparado com n^*. Outra ajuda computacional para a emulação é escalonar a matriz de pesos de forma que cada coluna tenha variância um. Isso ajuda com o afinamento do amostrador de MCMC para os parâmetros σ^2 da função de covariância do processo gaussiano, já que isso faz as dimensões n^* comparáveis entre si.

Para reconstruir a partir do sub-espaço $\mathcal{Y}^{pc}$ para o espaço completo $\mathcal{Y}$ a

transformação também é linear. Podemos multiplicar os pesos por V_1^T para gerar uma reconstrução determinística $Y'' = Y^{pc}V_1^T$. Porém, isso não leva em consideração o fato de que projetando num sub-espaço n^*-dimensional, nós discartamos informação na redução de dimensão. Para levar em consideração essa informação perdida adicionamos múltiplos aleatórios dos autovetores que descrevem as dimensões discartadas, V_2. Nós modelamos esses múltiplos aleatórios com distribuições gaussianas com média zero e com variâncias correspondendo aos auto-valores relevantes. Isso nos dá uma reconstrução estocástica ao invés de determinística, que acomoda erros na redução de dimensão. Em resumo, nós reconstruímos como

$$Y'' = Y^{pc}V_1^T + \Phi V_2^T$$

onde Φ é uma matriz $N \times (n - n^*)$ com $i-$ésima coluna contendo N amostras de uma distribuição $N(0, \Gamma_{n^*+i,n^*+i})$. Nós então devemos adicionar a média das colunas de Y a cada linha de Y'' para completar o emulador.

Uma ferramenta de diagnóstico útil na construção de emuladores são os gráficos de validação cruzada *leave-one-out*. Esses são obtidos deixando de fora uma das N rodadas de treinamento no cenário, usando os restante $N-1$ dados como dados de treinamento, e então prevendo os valores que ficaram de fora. O gráfico dos valores preditos, com intervalos de 95% , contra os valores verdadeiros para cada dimensão do output nos fornece informações valiosas sobre como o emulador está se comportando e permite que avaliemos o emulador. Esses gráficos podem ser usados para escolher a dimensão do sub-espaço principal e quais componentes devem ser incluídas. Também são úteis para escolher quais funções de regressão usar nas especificação da estrutura de média. Uma vez que tenhamos validado o emulador, podemos proceder para usar o modelo calibrado.

Nós usamos emulação por componentes principais para construir um substituto barato para o modelo climático de UVic introduzido anteriormente. Lembre-se que o output do modelo é uma série temporal de 200 previsões de CO_2 atmosférico. Figure 2.4 mostra gráficos de validação que usam a técnica de validação cruzada de *leave-one-out* para a seleção de quatro dos 200 pontos de output. A emulação foi feita projetando a série temporal num sub-espaço de dimensão 10 e então emulando cada mapeamento com processos gaussianos idependentes antes de reconstruir o dado de volta ao espaço de 200 valores. Uma estrutura a priori de média quadrática foi usada, $h(\theta_1, \theta_2) = (1, \theta_1, \theta_2, \theta_1^2, \theta_2^2$, já que os gráficos de validação cruzada mostraram que isso gerou uma performance superior a estrutura de média linear ou constante, com ganhos pequenos ao incluir termos de ordem maior.

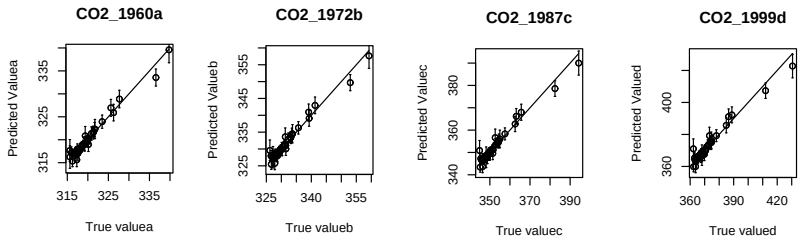


Figura 2.4: Leave-one-out cross validation plots for a selection of four of the 200 outputs. The error bars show 95% credibility intervals on the predictions. The two outliers seen in each plots are for model runs with inputs on the edge of the design. These points are predictions where we extrapolate rather than interpolate from the other model runs.

Os gráficos mostram que o emulador é acurado na predição nas rodadas mantidas fora da análise e que a incerteza nas nossas predições (mostradas pelos intervalos de 95%) fornecem uma medida razoável da nossa incerteza (com cobertura média de 91%).

Esse exemplo continua na Seção 3.2.4 onde mostramos como nosso emulador pode ser usado para calibrar o modelo para a famosa sequência de dados de Keeling.

Capítulo 3

Tópicos avançados

3.1 Validação e diagnósticos

Emuladores têm sido usados em várias áreas da ciência como aproximações estocásticas para simuladores caros, mas para construir esses emuladores algumas suposições e aproximações devem ser feitas. A não ser que o emulador represente corretamente o simulador, inferências feitas usando o emulador serão inválidas. Dessa forma, emuladores devem passar por testes de validação. Existe uma vasta literatura sobre o uso de emuladores para representação de simuladores caros, mas existe pouca pesquisa na validação de emuladores antes que estes sejam usados. Uma excessão é o trabalho de Bastos and O'Hagan (2009), que é a base para essa seção.

3.1.1 Possíveis problemas com emuladores gaussianos

Apesar do processo gaussiano ser uma classe muito flexível de distribuições para representar crenças a priori sobre o modelo computacional, o emulador gaussiano (1.16) pode fornecer predições pobres para os outputs do simulador por pelo menos duas razões. Primeiramente, a suposição de estacionaridade de um processo gaussiano com uma certa média e estrutura de covariância pode ser inapropriada. Segundo, mesmo que essa suposição seja razoável, existem vários parâmetros a serem estimados, e uma má escolha dos dados de treinamento pode sugerir valores inapropriados para esses parâmetros. No caso de parâmetros de alcance da correlação, se condicionamos em estimativas fixas ao invés de integrar usando a distribuição a posteriori completa, podemos também fazer uma escolha pobre dessas estimativas.

O simulador é representado por um processo gaussiano, então a suposição

de normalidade conjunta dos outputs do simulador deve ser um julgamento razoável. Em particular, o emulador diz que é muito improvável que o output do simulador seja mais que dois ou três desvios padrão preditivo da média preditiva, e que é tão provável estar abaixo quanto acima da média preditiva. Neste contexto, transformações podem ser úteis.

Além de assumir normalidade, formas específicas são supostas para as funções de média e covariância. Se a forma assumida para a média em (1.2) estiver errada, ou porque regressores errados foram usados em $h(\cdot)$, ou porque os coeficientes β foram mau estimados, então as previsões do emulador podem ser sitematicamente muito baixas ou muito altas em algumas regiões do espaço dos inputs.

Em (1.3), assume-se estacionariedade para a função de correlação, implicando que o output do simulador tem graus de suavidade parecidos em todos os pontos do espaço dos inputs. Assume-se que existe uma variância comum σ^2 e que a função de correlação depende somente de $(\mathbf{x} - \mathbf{x}')$. Dessa forma, variâncias desiguais ou funções de correlação que dependam da posição $(\mathbf{x}, \mathbf{x}')$ ao invés da diferença $(\mathbf{x} - \mathbf{x}')$ são falhas na hipótese de estacionariedade. Na prática, simuladores podem responder mais rapidamente a mudanças no input para algumas regiões do espaço do que em outras. No caso de não-estacionaridade, intervalos de credibilidade para predições dos emuladores podem ser muito largos em regiões de baixa variabilidade e bem apertados em regiões de resposta mais dinâmica.

Finalmente, mesmo assumindo uma forma pra função de covariância que seja apropriada, ainda pode-se ter uma estimação pobre dos parâmetros de alcance. Má estimação dos parâmetros de correlação (δ) levam a intervalos de credibilidade que são muito largos ou muito estreitos na vizinhança dos pontos de treinamento.

3.1.2 Diagnósticos para validação de emuladores gaussianos

Para validar um emulador, nossos diagnósticos serão baseados em comparações entre predições dos emuladores e rodadas do simulador para um novo conjunto de dados. Seja $\mathbf{X}^{(v)} = (\mathbf{x}_1^{(v)}, \mathbf{x}_2^{(v)}, \ldots, \mathbf{x}_m^{(v)})$ um conjunto de inputs não observados, chamados de dados de input para validação. Os outputs do simulador para os dados de input de validaçãosão dados por $\mathbf{y}^{(v)} = \eta(\mathbf{X}^{(v)})$ onde $\mathbf{y}^{(v)} = (y_1^{(v)}, \ldots, y_m^{(v)})$, e $\eta(\mathbf{X}^{(v)}) = \left(\eta(\mathbf{x}_1^{(v)}), \ldots, \eta(\mathbf{x}_m^{(v)})\right)$. Os dados de input para validação devem ser selecionados para cobrir toda a região do espaço dos inputs para a qual queremos usar o emulador. Caso contrário, pode-se validar um emulador que não representa o simulador para

um particular subconjunto não observado do espaço dos inputs.

Um diagnóstico geral $D(\cdot)$ é uma função dos outputs dos dados de validação, e nós propomos comparar $D(\mathbf{y}^{(v)})$ observado com a distribuição de referência de $D(\eta(\mathbf{X}^{(v)}))$ condicionada nos dados de treinamento. Se $D(\mathbf{y}^{(v)})$ está numa região escolhida apropriada com probabilidade baixa, então isso sugere que existe um conflito entre o emulador e o simulador. As regiões de teste para $D(\cdot)$ devem ser escolhidas de forma que $D(\mathbf{y}^{(v)})$ cair na região de teste está associado com falha na construção do emulador. Se ao longo do intervalo desses diagnósticos não há indicação de conflito, então podemos assumir que o emulador está representando a incerteza de forma apropriada.

Erros de predição individuais

Erros de predição individuais para os dados de validação são dados pela diferença entre o output do simulador observado e a média preditiva do output nos mesmos valores dos inputs, i.e. $(\mathbf{y}_i^{(v)} - E[\eta(\mathbf{x}_i^{(v)})|\mathbf{y}])$, para $i = 1, 2, \cdots, m$.

Nós consideramos cada erro de predição padronizado

$$D_i^I(\mathbf{y}^{(v)}) = \frac{y_i^{(v)} - E[\eta(\mathbf{x}_i^{(v)})|\mathbf{y}]}{\sqrt{V[\eta(\mathbf{x}_i^{(v)})|\mathbf{y}]}} \tag{3.1}$$

como um diagnóstico. Se o emulador pode representar o simulador de forma apropriada, os erros de predição padronizados têm distribuição t-student padrão, condicional nos dados de treinamento e nos parâmetros de correlação estimados δ. Na prática, o número de dados de treinamento é em geral grande o suficiente de forma que os graus de liberdade s ao grandes e podemos usar a distribuição normal. Então, erros individuais grandes, com valor absoluto digamos maior que 2, indicam um conflito entre o simulador e o emulador. Um outlier isolado desse tipo pode ser ignorado, ou pode indicar um problema local somente no entorno dos inputs para esse dado de validação. Isso pode ser investigado obtendo mais algumas rodadas de dados de validação nessa vizinhança.

Se há muitos erros padronizados com valores grandes, isso indica um problema mais sistemaático. Se erros grandes com o mesmo sinal aparecem em alguma parte do espaço dos inputs, isso sugere uma escolha inapropriada da função de média. Pode também indicar falha da suposição de estacionaridade.

Se erros grandes aparecem em pontos de validação que estão perto de dados de treinamento, isso indica que um ou mais parâmetros de correlação

foram super-estimados, de forma que as predições do emulador são fortemente influenciadas por dados de treinamento na vizinhança. Se não há um padrão claro na ocorrência de erros grandes, então o problema pode ser devido a falta de homocedasticidade.

Deve-se notar que padrões de erros padronizados que são menores que o esperado podem indicar conflitos complementares a aqueles que acabamos de discutir. Por exemplo, se pontos de validação perto de dados de treinamento têm erros padronizados menores do que o esperado, isso pode sugerir sub-estimação dos parâmetros de correlação. Uma maneira poderosa para identificar padrões de erros grandes ou pequenos é através de gráficos, estes serão discutidos na Seção 3.1.2.

Distância de Mahalanobis

Apesar da coleção de erros individuais padronizados $D^I(\mathbf{y}^{(v)})$ fornecerem diagnósticos muito úteis, também é importante ser capaz de resumí-los em um único diagnóstico.

Hills and Trucano (1999, 2001) usam um test qui-quadrado para comparar o simulador com o processo real numa abordagem de validação e verificação (V&V). A mesma idéia pode ser usada como um diagnóstico para comparar predições do emulador com outputs do simulador sob os mesmos inputs. O diagnóstico que eles propuseram é dado por

$$D_{\chi^2}(\mathbf{y}^{(v)}) = \sum_{i=1}^{m} D_i^I(\mathbf{y}^{(v)})^2. \tag{3.2}$$

Para um grande conjunto de dados de treinamento, i.e. quando $n \to \infty$, e com valores do output independentes, a distribuição de $D_{\chi^2}\left(\eta(\mathbf{X}^{(v)})\right)$ converge para uma distribuição qui-quadrado com m graus de liberdade. Entretanto, a suposição de independência é muito forte. Por exemplo, se o simulador é uma função suave, então outputs parecidos são esperados quando os elementos estão perto uns dos outros no espaço dos inputs. Essa correlação é capturada pelo emulador na função de covariância (1.17).

Uma extensão natural de (3.2) permitindo a correlação entre os outputs é a distância de Mahalanobis entre o emulador e os outputs do simulador no conjunto de inputs de validação, dada por

$$D_{MD}(\mathbf{y}^{(v)}) = \left(\mathbf{y}^{(v)} - E\left[\eta(\mathbf{X}^{(v)})|\mathbf{y}\right]\right)^T \left(V\left[\eta(\mathbf{X}^{(v)})|\mathbf{y}\right]\right)^{-1} \left(\mathbf{y}^{(v)} - E\left[\eta(\mathbf{X}^{(v)})|\mathbf{y}\right]\right), \tag{3.3}$$

onde os elementos do vetor de média preditiva $E\left[\eta(\mathbf{X}^{(v)})|\mathbf{y}\right]$ e a matriz de covariância preditiva $V\left[\eta(\mathbf{X}^{(v)})|\mathbf{y}\right]$ para o emulador gaussiano são dados

por (1.17) e (1.18) respectivamente. Valores extremos (grandes ou pequenos) para a distância de Mahalanobis observada ($D_{MD}(\mathbf{y}^{(v)})$) indicam um conflito entre o emulador e o simulador.

Sob normalidade do emulador, a distribuição de $D_{MD}(\eta(\mathbf{X}^{(v)}))$ condicional nos dados de treinamento e em uma estimativa do parâmetro de correlação δ é uma distribuição F-Snedecor escalonada com m e $n-q$ graus de liberdade,

$$\frac{(n-q)}{m(n-q-2)} D_{MD}(\eta(\mathbf{X}^{(v)}))|\mathbf{y}, \delta \sim F_{m,n-q}. \tag{3.4}$$

O valor esperado e a variância da distância de Mahalanobis são, respectivamente, dadas por

$$\begin{aligned} E[D_{MD}(\eta(\mathbf{X}^{(v)}))|\mathbf{y}, \delta] &= m, \\ \text{Var}[D_{MD}(\eta(\mathbf{X}^{(v)}))|\mathbf{y}, \delta] &= \frac{2m(m+n-q-2)}{n-q-4}. \end{aligned}$$

Como mencionado anteriormente, um valor não esperado pequeno ou grande de $D_{MD}(\mathbf{y}^{(v)})$ indica um conflito entre o emulador e o simulador. Se esse problema acontece, é importante explorar erros individuais, para procurar por padrões de valores pequenos ou grandes, para identificar a causa mais provável do problema. Com esse objetivo, agora iremos considerar maneiras alternativas de decompor a distância de Mahalanobis em diagnósticos individuais.

Decomposições da variância

Erros de predição individuais (3.1) são correlacionados, o que introduz alguns riscos em sua interpretação. Também, olhando para erros individuais pode não identificar efetivamente alguns conflitos entre o emulador e o simulador. Por exemplo, dois erros podem não ser individualmente grandes, mas se eles têm sinais opostos quando eles são altamente correlacionados, então isso sugere um conflito. Seja $\mathbf{G}$ uma matriz de desvio padrão tal que $V[\eta(\mathbf{X}^{(v)})|\mathbf{y}] = \mathbf{G}\mathbf{G}^T$. Então, o vetor de erros tranformados

$$D_{\mathbf{G}}(\mathbf{y}^{(v)}) = \mathbf{G}^{-1}(\mathbf{y}^{(v)} - E[\eta(\mathbf{X}^{(v)})|\mathbf{y}]) \tag{3.5}$$

São não correlacionados e têm variância unitária. Se a suposição de normalidade para os outputs é razoável, a distribuição de cada um desses erros é t-student com $(n-q)$ graus de liberdade. Nós podemos considerar isso como um conjunto alternativo de diagnósticos. Assim como com os erros

$D^I(\mathbf{y}^{(v)})$, nós procuramos por erros individuais transformados grandes e padrões de valores grandes ou pequenos. Entretanto, a estrutura de $\mathbf{G}$ nos dará diferentes interpretações para esses padrões. Essa abordagem é similar a de Houseman et al. (2004), que usa resíduos individuais rotacionados para modelos lineares com erros correlacionados.

Outra propriedade desse diagnóstico é que $D_{MD}(\mathbf{y}^{(v)}) = D_{\mathbf{G}}(\mathbf{y}^{(v)})^T D_{\mathbf{G}}(\mathbf{y}^{(v)})$. Isto é, a soma de quadrados dos elementos de $D_{\mathbf{G}}(\mathbf{y}^{(v)})$ é a distância de Mahalanobis, e então nós podemos interpretar esses diagnósticos como uma decomposição de $D_{MD}(\mathbf{y}^{(v)})$.

Existem muitas maneiras de decompor uma matriz positiva definida no produto de uma matrix raíz quadrada e sua transposta. As escolhas naturais são a decomposição de Cholesky e a decomposição de auto-valor/vetor (*eigen decomposition*) (Golub and van Loan 1996). A decomposição de auto-valor/vetor é muito popular, mas a decomposição de Cholesky é computacionalmente mais eficiente e nós veremos que ela tem também uma interpretação mais intuitiva que a decomposição de auto-valor/vetor.

decomposição de auto-valor/vetor Quando $\mathbf{G}$ é a matriz de decomposição de auto-valor/vetor, nós denotamos os elementos do vetor $D_{\mathbf{G}}(\mathbf{y}^{(v)})$ por $D_i^E(\mathbf{y}^{(v)})$, e os chamamos de erros de auto-valor/vetor. Quando um grande $D_i^E(\mathbf{y}^{(v)})$ é identificado, mais infromação pode ser obtida ao estudar quais erros individuais têm os maiores pesos na combinação linear. Se o peso for tão importante quanto os erro individual, então esse erro deve ser estudado como sugerido na seção 3.1.2. Se o peso enfatiza um subconjunto de erros de precição individuais, então isso pode indicar um problema numa região do espaço dos inputs no entorno dos inputs dos dados de validação associados, indicando um possível problema de não estacionaridade.

Decomposição de Cholesky. A decomposição de Cholesky é um caso especial quando $\mathbf{G}^T$ é a única matriz triangular superior $\mathbf{R}$ tal que $V[\eta(\mathbf{X}^{(v)})|\mathbf{y}] = \mathbf{R}^T\mathbf{R}$, e denotamos os elementos do vetor $D_{\mathbf{G}}(\mathbf{y}^{(v)})$ por $D_i^C(\mathbf{y}^{(v)})$, e os chamamos erros de Cholesky. Então $\mathbf{G}^{-1}$ é também uma matriz triangular, e $D_i^C(\mathbf{y}^{(v)})$ é a única combinação linear dos primeiros i erros de validação tal que sua variância preditiva é a variância condicional do i-ésimo erro de validação dados os outros $i-1$ erros. Apesar dessa abordagem ter o benefício de produzir erros transformados não correlacionados que ainda estão relacionados com os pontos de validação individuais (em contraste com a decomposição de auto-valor/vetor), a decomposição não é invariante a como ordenamos as observações, e padrões de valores altos e baixos não têm uma interpretação obvia.

Decomposição pivotal de Cholesky. Permutando o conjunto de dados de validação, obtemos uma decomposição de Choleski diferente. E

cada uma dessas permutações podem identificar anomalias diferentes. Entretanto, para ter os benefícios de ambas decomposições (Choleski e de autovalor/vetor), notamos que os diagnósticos mais efetivos são obtidos permutando os dados de forma que o primeiro elemento é o elemento com maior variância, o segundo elemento é o que possui maior variância condicional no primeiro elemento com maior variância, e assim por diante. Então denotamos os elementos do vetor $D_{\mathbf{G}}(\mathbf{y}^{(v)})$ por $D_i^{PC}(\mathbf{y}^{(v)})$, e os chamamos erros pivotais de Cholesky. Essa permutação pode ser obtida aplicando a decomposição pivotal de Cholesky, a qual retorna a matriz $\mathbf{P}$ e a única matriz triangular $\mathbf{R}$ tal que $\mathbf{P}^T V[\eta(\mathbf{X}^{(v)})|\mathbf{y}]\mathbf{P} = \mathbf{R}^T\mathbf{R}$. Então $\mathbf{G} = \mathbf{P}\mathbf{R}^T$.

Um grupo de grandes erros pivotais de Cholesky na primeira parte da sequência sugere não-homogeneidade, enquanto grandes ou pequenos erros na parte final da sequência indicam má estimação de δ ou uma escolha inapropriada da estrutura de correlação. Além disso, temos o benefício que cada um dos $D_i^{PC}(\mathbf{y}^{(v)})$s está associado com um particular ponto de validação, o que torna mais fácil investigar os grandes erros.

Métodos gráficos

Gráficos formam uma maneira eficiente de investigar a adequação de predições de emuladores e de checar algumas suposições feitas na construção do emulador (1.16). Nós propomos alguns métodos gráficos usando ambos erros individuais padronizados (3.1) e os erros padronizados não correlacionados (3.5).

Gráfico dos erros individuais contra as predições do emulador. Nesse diagnóstico gráfico, nós procuramos por padrões sugerindo um problema na função de média. Por exemplo, se para alguns intervalos do output, os erros são sistematicamente positivos (ou negativos) isso indica a má especificação da função de média. Heterocedasticidade dos erros individuais sugere que o simulador deve ser estudado como um processo não estacionário. Grandes erros individuais em valor absoluto podem sugerir que a variância preditiva é muito pequena, e erros individuais muito perto de zero podem sugerir uma variância muito grande.

Além de fazer o gráfico dos erros individuais $D^I(\mathbf{y}^{(v)})$ dessa maneira, podemos fazer o gráfico dos erros não correlacionados padronizados obtidos por decomposição de Cholesky ou Cholesky pivotal, porque cada erro pode ser mapeado a uma predição do emulador. Entretanto, é menos provável ver grupos de desvios sitemáticos indicando problemas na função de média. Considere, por exemplo, um grupo de erros individuais positivos em alguma parte do gráfico. Esses podem ser pontos de validação que estão relativa-

mente muito perto no espaço dos inputs. O primeiro desses a ser colocado no gráfico na sequência de Cholesky ou Cholesky pivotal $D_i^C(\mathbf{y}^{(v)})$ ou $D_i^{PC}(\mathbf{y}^{(v)})$ pode se mostrar no gráfico como um erro grande, mas os subsequentes são condicionados nos primeiros e podem aparecer normais.

Não podemos fazer gráficos dos erros não correlacionados padronizados obtidos por decomposição de auto-valor/vetor dessa maneira.

Gráfico dos erros contra os índices O significado do índice depende de qual erro usamos para fazer o gráfico. Para erros individuais $D^I(\mathbf{y}^{(v)})$ e erros de Cholesky $D_i^C(\mathbf{y}^{(v)})$, o índice i é a ordem do dado de validação. Para erros de auto-valor/vetor, o índice é a ordem de $D_i^E(\mathbf{y}^{(v)})$s que possui a maior variância preditiva. E para os erros de Cholesky pivotais, o índice é a ordem de pivotamento, que dá a ordem dos $D_i^{PC}(\mathbf{y}^{(v)})$s com as maiores variâncias condicionais preditivas. Para todos esses gráficos, é esperado que os erros flutuem em torno de zero com variância constante e sem padrão. Muitos erros grandes indicam sub-estimação da variância. Por outro lado, muitos erros pequenos indicam sobre-estimação da variância. Em ambos os casos, pode também sugerir que o simulador é um processo não-estacionário.

A decomposição de Cholesky pivotal e a decomposição de auto-valor/vetor fornecem uma interpretação extra que podemos associar com a estrutura de correlação. Em ambos os casos, se nós observamos ou erros muito grandes ou muito pequenos no começo do gráfico, i.e. do lado esquerdo, indica falha na estimação da variância preditiva, ou não estacionaridade. Se observamos erros grandes (ou muito pequenos) no final do gráfico, entretanto, i.e. no lado direito, indica que os parâmetros de alcance da correlação foram sobre (sob) estimados ou a estrutura de correlação escolhida é inapropriada.

Gráficos quantil-quantil. Sob normalidade, os erros não correlacionados $D_G(\mathbf{y}^{(v)})$ têm distribuição t-student padrão com $(n-q)$ graus de liberdade. Então, o gráfico quantil-quantil (QQ-plot) usando essa distribuição se torna um gráfico natural pra diagnóstico. Num QQ-plot, se os pontos estão perto da reta de 45 graus que passa pela origem, a normalidade dos outputs do simulador é considerada uma suposição razoável. Se os pontos se agrupam em torno de uma reta com coeficiente angular menor (ou maior) que um, a implicação é de que a variabilidade preditiva foi super-estimada (sub-estimada).

Curvatura nesse gráfico indica não normalidade, enquanto outliers no começo ou no final do gráfico sugerem problemas de ajuste ou não- estacionaridade.

A interpretação do QQ-plot usando erros não correlacionados padronizados é independente do método de decomposição, e é em geral informativo para ambas decomposição de auto-valor/vetor e decomposição de Cholesky.

Apesar da distribuição de cada erro individual padronizado $D_i^I(\mathbf{y}^{(v)})$ ser a t-student padrão, o fato de que os erros são correlacionados faz o QQ-plot mais difícil de interpretar.

Gráfico dos erros contra os inputs. Fazer o gráfico dos erros padronizados contra os valores correspondentes de cada input pode ser útil. Mais uma vez, esperamos ver uma banda horizontal contendo os erros. Esses gráficos são usados para identificar comportamentos diferentes dos erros em algumas partes do espaço dos inputs, indicando possível falha da suposição de não-estacionaridade. Esse gráfico pode também indicar que a relação entre o input e a predição não foi completamente representada na função de média. Por exemplo, nós podemos identificar um padrão que não foi incluído na função de média.

Note que não podemos fazer o gráfico dos erros de auto-valor/vetor dessa maneira. É possível fazer o gráfico dos erros de Cholesky e Cholesky pivotal, mas a ligação entre cada erro e o dado de validação, faz com que a interpretação seja complicada pelo condicionamento da mesma forma de quando fazemos o gráfico contra a média do emulador.

3.1.3 Exemplos

Nós utilizaremos os diagnósticos para o exemplo 1.4.2, onde o dado de treinamento é composto por 20 pontos selecionados usando amostragem de hipercubo latino. O dado de validação é composto por 25 pontos independentemente selecionados usando outra amostra por hipercubo latino. Figura 3.1 apresenta os dados de treinamento e validação. Usando o dado de treinamento, os parâmetros de alcance da correlação estimados são $(\hat{\delta}_1, \hat{\delta}_2) = (0.2421, 0.4240)$.

Então, nós prevemos o output para os pontos de validação usando um processo t-student (1.16) condicional nos dados de treinamento e nos alcances de correlação estimados. O diagnóstico de qui-quadrado observado, $D_{\chi^2}(\mathbf{y}^{(v)}) = 24.411$, é muito próximo do seu valor esperado $E[D_{\chi^2}(\eta(\mathbf{X}^{(v)}))] = 25$, sugerindo que o emulador é uma boa aproximação para o simulador. Entretanto, esse diagnóstico ignora o fato de que os outputs são correlacionados. Tabela 3.1 apresenta a distância de Mahalanobis observada e intervalos de credibilidade, e algumas estatísticas de suas distribuições preditivas.

A distância de Mahalanobis observada, $D_{MD}(\mathbf{y}^{(v)}) = 70.36$, é um valor extremo de sua distribuição teórica, a distribuição F escalonada com parâmetros (25,18). Isso indica conflito entre emulador e simulador. O intervalor de credibilidade diagnóstico observado é menos dramático. Vemos que 92% (23 em 25) dos outputs do simulador caem dentro dos repectivos intervalos de

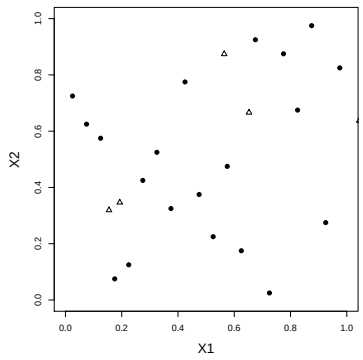


Figura 3.1: Dados de treinamento (•) e validação (△) amostrados usando independentes esquemas de amostragem por hipercubos latinos.

credibilidade marginal de 95% construídos pelo emulador. O menor quartil da distribuição de $D_{CI}(\eta(\mathbf{X}^{(v)}))$ obtido por simulação, é 0.92, e de fato não é surpreendente ter 2 valores em 25 fora dos intervalos de 95%.

Os diagnósticos de qui-quadrado e os intervalos de credibilidade sugerem que as predições dos emuladores são marginalmente satisfatórias mas a distância de Mahalanobis deixa claro que conjuntamente elas não são válidas. Isso pode estar relacionado com a má estimação dos parâmetro de alcance da correlação ou a não homogeneidade no espaço dos inputs.

Tabela 3.1: Distância de Mahalanobis observada e intervalos de credibilidade diagnósticos, com algumas estatísticas das distribuições preditivas.

	Obs.	Expected	Std. dev.	1stQ	Median	3rdQ
$D_{MD}(\cdot)$	70.360	25.000	12.404	16.016	21.778	29.438
$D_{CI}(\cdot)$	0.920	0.951	0.072	0.920	1.000	1.000

Figura 3.2 apresenta alguns diagnósticos gráficos usando erros padronizados individuais. Figura 3.2 (a) apresenta os erros padronizados individuais contra as predições do emulador dado o valor esperado. Não há nenhum padrão obvio, apesar dos dois maiores erros individuais estarem associados com valores pequenos das predições, o que pode indicar um problema na função média ou um problema de não estacionaridade. Figura 3.2 (b) é o QQ-plot dos erros individuais, e confirma o que foi indicado pelos diagnósticos de qui-quadrado e intervalos de credibilidade de que as predições parecem válidas marginalmente.

Para checar um possível problema de estacionaridade, fazemos o gráfico dos erros individuais padronizados contra cada input, Figura 3.2 (c) e (d). Os dois erros grandes são associados com valores pequenos do input 1, e valores grandes do input 2. Isso pode estar conectado com uma sub-região do espaço dos inputs não representada pelos dados de treinamento. Se possível, novas rodadas do simulador nessa sub-região podem melhor o emulador. Também, podemos notar na Figura 3.2 (c) que quanto maior o valor do input 1, menor a variabilidade dos erros individuais. Isso pode indicar não estacionaridade, ou talvez uma sobre-estimação do parâmetro de alcance da correlação δ_1. Para o segundo input, Figura 3.2 (d), não há padrão para os erros.

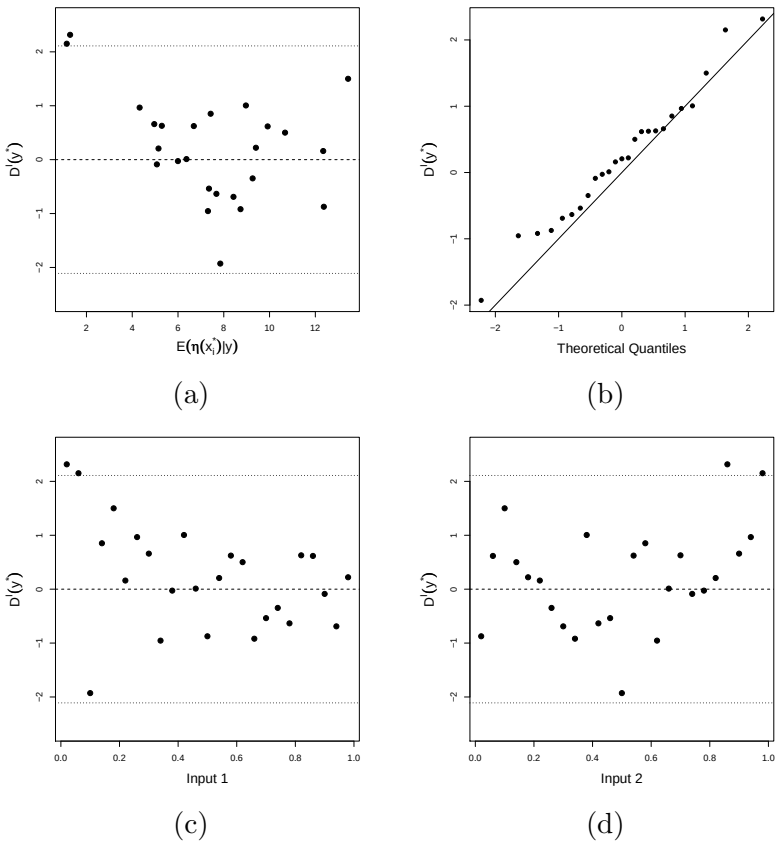


Figura 3.2: Diagnósticos gráficos para o exemplo usando erros individuais padronizados: (a) $D^I(\mathbf{y}^{(v)})$ contra as predições dos emuladores; (b) gráfico quantil-quantil; (c) $D_i(\mathbf{y}^{(v)})$ contra input 1; (d) $D_i(\mathbf{y}^{(v)})$ contra input 2.

Os diagnósticos apresentados na Figura 3.2 ignoram a estrutura de correlação dos erros, e dessa forma, não tratam o problema encontrado pela distância de Mahalanobis. Figure 3.3 apresenta disgnósticos gráficos usando erros padronizados não correlacionados.

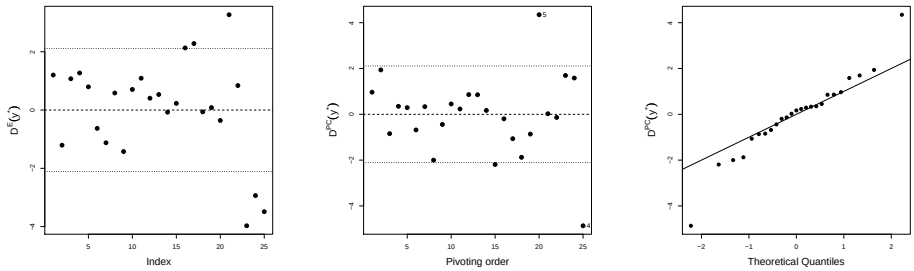
Os erros de auto-valor/vetor apresentados na Figura 3.3 (a), onde podemos observar erros grandes no final dos gráficos. Isso claramente indica um problema na estrutura de correlação, que pode ser uma má especificação da função de correlção, ou uma sobre-estimação dos parâmetros de alcance da correlação. Quando examinamos os pesos dados pelo vetor de auto-valor/vetor para os erros grandes, não podemos identificar um padrão para os componentes 16^o, 17^o e 23^o dos erros de auto-valor/vetor. Entretanto, as componentes do 21^o valor indica que o 5^o ponto de validação é muito importante para o tamanho dos erros de auto-valor/vetor. E todos os pontos de validação relacionados aos maiores pesos do 24^o e 25^o erros apresentam valores para o input 1 que são menores que 0.5.

Os erros de Cholesky pivotais são apresentados na Figura 3.3 (b). Grandes erros são novamente observados no final do gráfico nesse diagnóstico, sugerindo um problema com a estrutura de correlação. Entretanto, com esse gráfico é fácil explorar a natureza do problema, porque há apenas dois valores grandes e podemos conectar cada um com um único ponto de validação. Os seus vetores de input são (0.14, 0.58) e (0.18, 0.10). Esses dois pontos são diferentes dos dois grandes erros padronizados individuais, mas eles também são caracterizados por valores pequenos do input 1, sugerindo que as predições dos emuladores não são válidos nessa parte do espaço dos inputs.

Figura 3.3 (c) apresenta o QQ-plot dos erros de Cholesky pivotais, onde podemos notar que os pontos se agrupam em torno de uma reta com coeficiente angular ligeiramente maior que um, de forma que pode haver uma ligeira sub-estimação da variabilidade preditiva. Entretanto, os dois outliers, que são os maiores valores na Figura 3.3 (b), são a principal característica desse gráfico.

Em resumo, os diagnósticos gráficos e numéricos indicam alguns conflitos entre o emulador e o simulador. O conflito parece estar relacionado com má estimação dos parâmetros de correlação, e talvez a não-estacionaridade e também a dados de treinamento que não cobrem adequadamente a subregião do espaço dos inputs onde X_1 tem valores menores que 0.2.

Como o simulador (1.20) é uma função simples, podemos rapidamente rodá-lo mais vezes. Então, combinamos os dados de treinamento com os dados de validação, mais 5 observações selecionadas numa subregião do espaço onde os inputs são menores que 0.5. Usando o dado de treina-



(a) (b) (c)

Figura 3.3: Diagnósticos gráficos para o exemplo usando os erros padronizados não correlacionados: (a) os erros de de auto-valor/vetor $D^E(\mathbf{y}^{(v)})$, contra o autovalor; (b) os erros the Cholesky pivotais $D^{PC}(\mathbf{y}^{(v)})$, contra a ordem pivotal; (c) QQ plot dos erros de Cholesky pivotais.

mento atualizado, os parâmetros de alcance da correlação estimados são $(\hat{\delta}_1, \hat{\delta}_2) = (0.1764, 0.4116)$, que são menores que a estimativa anterior, confirmando a suspeita de sobre-estimação, especialmente δ_1.

Um novo dado de validação foi selecionado usando amostragem por hipercubo latino, e a distância de Mahalanobis e intervalos de credibilidade são apresentados na Tabela 3.2. A distância de Mahalanobis é ainda maior que esperado, mas agora sugere muito menos conflito com o simulador. O intervalo de credibilidade novamente indica que as predições do emulador para o dado de validação são individualmente razoáveis.

Tabela 3.2: Distância de Mahalanobis observada e intervalos de credibilidade, com algumas estatísticas de suas distribuições preditivas, usando o novo dado de treinamento.

	Obs.	Esperado	Desvio padrão	1^oQ	Mediana	3^oQ
$D_{MD}(\cdot)$	51.129	30	10.230	23.172	28.958	36.324
$D_{CI}(\cdot)$	0.933	0.950	0.058	0.933	0.967	1.000

Figura 3.4 (a) apresenta os erros padronizados individuais. Não há nenhum padrão obvio, e apesar de haver alguns erros fora das bandas de credibilidade, eles não são grandes o suficiente para sugerir algum conflito mais sério. Figura 3.4 (b) sugere que o emulador está sub-estimando os valores grandes do output, e talvez não esteja capturando adequadamente o pico na superfície do output.

Os erros de Cholesky pivotais são apresentados na Figura 3.4 (c). Apesar de não haver erros grandes, muitos estão fora das bandas de credibilidade. O QQ-plot dos erros de Cholesky pivotais, Figura 3.4 (d), confirma que a variabilidade predita dos emuladores é um pouco menor que a variabilidade observada.

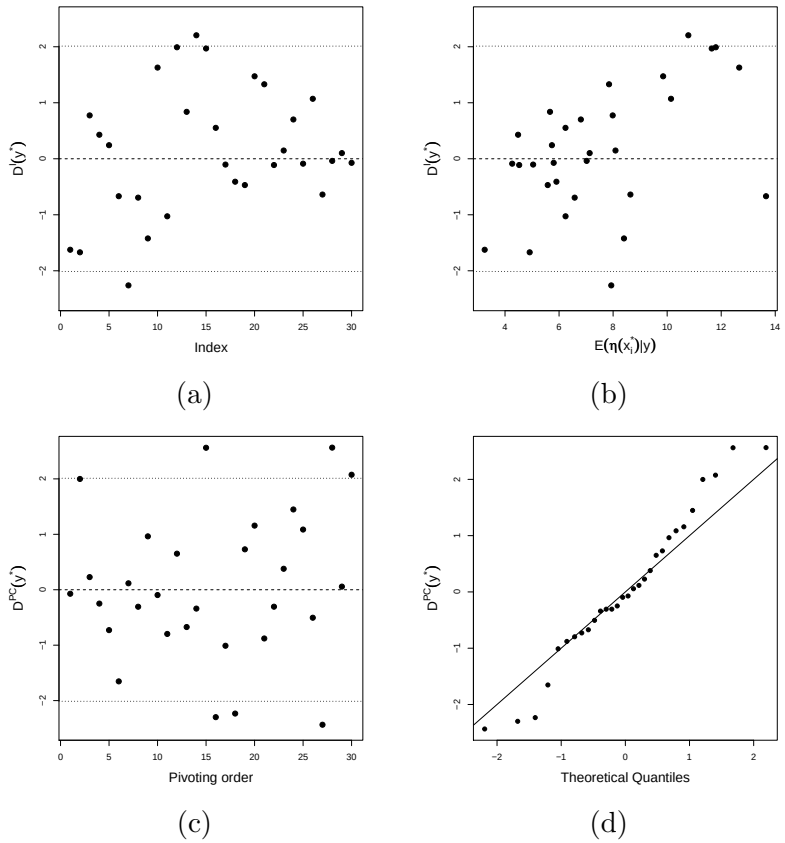


Figura 3.4: Diagnósticos gráficos para o exemplo com novo dado de treinamento: erros individuais padronizados $D^I(\mathbf{y}^{(v)})$ contra (a) a ordem de validação, e (b) predições do emulador; (c) erros de Cholesky pivotais $D^{PC}(\mathbf{y}^{(v)})$ contra a ordem de pivotamento; (d) QQ-plot dos erros de Cholesky pivotais.

Apesar dos diagnósticos indicarem que talvez ainda existam alguns problemas de validação, o emulador construído com o dado de treinamento atualizado melhorou as predições. Nesse ponto, podemos fazer uma construção final do emulador usando todas as rodadas (o dado de treinamento

original, o dado de validação original, os 5 pontos extras e o dado final de validação). Nós esperamos que o emulador final seja válido, de acordo com os poucos problemas encontrados na segunda etapa de teste de validação, e não precisaremos validá-lo novamente.

Figura 3.5 mostra a melhora no emulador. Figura 3.5 (a) mostra a média do emulador contra os dois inputs, baseando-se somente no dado de treinamento original. Figuras 3.5 (b) e (c) mostram como o emulador evolui ao incluirmos o dado de validação original e os 5 pontos extras, e então incluímos também o dado de validação adicional. O dado de treinamento para cada emulador e também o dado de validação usados para diagnóstico são ilustrados em cada gráfico. Figura 3.5 (d) apresenta o valor verdadeiro do simulador (um gráfico que em geral não esta disponível em simuladores reais). Figura 3.5 (d) mostra que essa é uma função difícil de emular, a qual é muito sensível a valores pequenos do input 1. O emulador original não captura esse comportamento, mas depois de incluir os dados de validação originais e mais 5 pontos a forma correta começa a aparecer. O emulador final na Figura 3.5 (c) fornece uma boa representação do simulador.

3.2 Calibração

Em estatística, *calibração* é o termo usado pra descrever o precesso inverso de ajustar o modelo ao dado, apesar de ser referido em modelos dinâmicos como estimação de parâmetros ou assimilação de dados. Aqui, nós consideramos o problema no qual nós temos um modelos computacional de um sistema físico juntamente com observações desse sistema. O objetivo é combinar a ciência capturada pelo modelos computacional com as observações físicas de forma a aprender sobre o valor dos parâmetros e sobre as condições iniciais do modelo. Queremos incorporar

1. o modelo computacional, $\eta(\cdot)$,
2. dados de campo do sistema físico, $\mathcal{D}_{\text{field}}$,
3. qualquer outra informação adicional.

Nós consideramos cada uma dessas fontes.

O modelo computacional

O modelo computacional, $\eta(\cdot)$, é considerado um mapeamento do espaço dos inputs Θ, para o espaço dos outputs $\mathcal{Y} \subset \mathbb{R}^n$. Os parâmetros do input

$\theta \in \Theta$ são os parâmetros de calibração que queremos estimar. Eles podem ser constantes físicas (apesar cuidado é necessario na identificação de parâmetros como constantes físicas; veja Seção 3.2.2), contantes específicas do contexto, ou parâmetros que devem ser ajustados para fazer o modelo ter uma boa performance. Esses são parâmetros que precisariam ser especificados se estivessemos fazendo um experimento físico. Modelos também frequentemente têm parâmetros a serem controlados que atuam como indicadores do contexto ou como troca de cenário. Por motivos de clareza, nós iremos ignorar esses inputs, mas note que a abordagem apresentada aqui pode ser extendida para lidar com esse caso se consideramos o espaço dos inputs como sendo $\Theta \times \mathcal{T}$, onde $\mathcal{T}$ é o espaço das variáveis de controle (veja detalhes em Kennedy and O'Hagan (2001))

O foco dessa seção é na calibração de modelos computacionais cujos tempos para rodar são muito longos. Nós mostramos como calibrar modelos estocásticos baratos na Seção 3.3. Uma consequência desse custo é que teremos poucas rodadas disponíveis do modelo. Em outras palavras, haverá um conjunto de N pontos de planejamento $D = \{\theta_i : i = 1, \ldots, N\}$ para os quais nós sabemos o output do modelo $\mathcal{D}_{\text{sim}} = \{y_i = \eta(\theta_i) : i = 1, \ldots, N\}$; Toda a informação do modelo virá dessa rodada somente. A questão de como escolher os pontos de planejamento D é discutida na Seção 2.1, e nós assumiremos que as rodadas $\mathcal{D}_{\text{sim}}$ disponíveis para análise foram bem planejadas.

Observações de campo

Nós assumimos que temos observações do sistema físico, $\mathcal{D}_{\text{field}}$, que diretamente corresponde a outputs do modelo computacional. Seja ζ_t a realidade em t, onde t é uma variável de índice tal como tempo ou localização, e assuma que o dado de compo é uma medição da realidade em t com erro aleátorio independente. Isto é

$$\mathcal{D}_{\text{field}}(t) = \zeta_t + \epsilon_t \tag{3.6}$$

onde $\epsilon_t \sim G(\cdot)$ para alguam distribuição G. Uma suposição usual é de que G é a distribuição gaussiana, $\epsilon_t \sim N(\mu_t, \sigma_t^2)$. Essa suposição não é estritamente necessária, entretanto, usar distribuições gaussianas facilita a inferência quando usamos emuladores gaussianos como veremos aqui. Outras suposições usuais são que $\mu_t = 0$ para todo t, e frequentemente temos erros homocedásticos de forma que $\sigma_t^2 = \sigma^2$ para todo t, entretanto, nenhuma dessas suposições é necessária para a análise. Nós tratamos os erros do modelo de forma separada do erro de medida por razões descritas na Seção

3.2.2. Um dos benefícios disso é que ϵ genuinamente representa o erro de medida. Como na maioria das vezes a taxa de erro de medida é conhecida, e será usualmente reportada com as medições, nós assumimos que μ_t e σ_t são constantes conhecidas neste texto. Se este não for o caso, é possível aprender sobre esses aprâmetros junto com os outros.

Outras informações relevantes

Calibração é primariamente a combinação da física no modelo com observações de campo para produzir estimativas dos parâmetros. Mas, frequentemente haverá informação adicional vinda de especialistas que deve ser introduzida no modelo. Parte dessa informação será informação a priori sobre os valores mais prováveis dos inputs, obtida através de experimentos anteriores e literatura, e será representada pela distribuição a priori $\pi(\theta)$. Os usuários do modelo podem também ter algum conhecimento a respeito de quão acuradamente o simulador representa o sistema. Como explicado em mais detalhe a seguir, quando calibramos um modelo é importante acomodar discrepâncias entre o modelo e a realidade. Desenvolvedores de modelo frequentemente são capazes de fornecer informação sobre como e onde o modelo pode estar errado. Eles podem, por exemplo, ter mais confiança em alguns outputs do modelo do que em outros, ou eles podem ter mais crença nas predições em alguns contextos do que em outros. Todas essas informações podem ser incorporadas na análise. Idealmente, informação deveria ser elicitada de especialistas antes deles terem observado rodadas do modelo ou dados de campo, entretanto, na prática esse não é o caso. Garthwaite et al. (2005) apresenta uma introdução a elicitação de crenças de especialistas.

3.2.1 Abordagem de calibração estatística

O método de calibração apresentado aqui é baseado na abordagem dada em Kennedy and O'Hagan (2001) e usa o conceito de um *melhor-input* (Goldstein and Rougier 2009). O método assume que existe um único melhor valor de θ, o qual chamamos $\hat{\theta}$, tal que o modelo rodado em $\hat{\theta}$ resulta a representação mais acurada do sistema. Note que $\hat{\theta}$ é o melhor valor somente no sentido de representar mais acuradamente os dados de acordo com a estrutura específica do erro, e como discutido anteriormente, o valor encontrado para $\hat{\theta}$ não precisa coincidir com o verdadeiro valor físico de θ. Uma consequência dessa suposição é que a rodada do modelo em seu melhor input é suficiente para a calibração, no sentido que uma vez que conhecemos $\eta(\hat{\theta})$ nós não podemos aprender nada além disso sobre a realidade a apartir do

simulador.

Uma suposição natural porém incorreta em calibração é assumir que nós observamos a predição do simulador mais perturbação aleatória independente. Se o simulador não é uma representação perfeita da realidade, essa suposição é errada e pode levar a sérios erros na análise e nas predições futuras. De forma a relacionar a predição do simulador com a realidade nós devemos permitir a existência de erro de modelo. Podemos fazer isso com um termo de erro aditivo e dizer que a realidade ζ é a melhor predição do simulador mais um erro de modelo δ:

$$\zeta = \eta(\hat{\theta}) + \delta. \tag{3.7}$$

Equações (3.6) e (3.7) descrevem completamente a forma estrutural assumida na abordagem de calibração de Kennedy and O'Hagan (2001). Combinando as duas equações resulta em

$$\mathcal{D}_{\text{field}} = \eta(\hat{\theta}) + \delta + \epsilon \tag{3.8}$$

onde todas as quantidades nessa equação são vetores. Uma consequência da abordagem do melhor input é que $\eta(\hat{\theta})$ é independente de δ nos permitindo especificar crenças sobre cada termo sem fazer referência aos outros. Uma representação esquemática da estrutura de independência condicional assumida para calibração é mostrada na Figura 3.6.

Em seções posteriores, suposições sobre distribuições são discutidas para todos os termos no lado direito da equação (3.8), mas antes consideramos brevemente o processo de inferência. Calibração é o processo de julgar quais valores dos inputs são consistentes com o dado de campo, o modelo e qualquer crença a priori. A abordagem Bayesiana para calibração é a de achar a distribuição a posteriori do parâmetro de melhor input dado essas três fontes de informação; Isto é, queremos encontrar

$$\pi(\hat{\theta}|\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}, E),$$

onde E representa a informação disponível, $\mathcal{D}_{\text{sim}}$ as rodadas do modelo, e $\mathcal{D}_{\text{field}}$ as observações de campo. A posteriori dá pesos relativos para todo $\theta \in \Theta$, e representa nossas crenças sobre o melhor input a luz da informação disponível.

Para calcular a distribuição a posteriori, nós usamos o teorema de Bayes para ver que a posteriori de $\hat{\theta}$ é proporcional a sua verossimilhança multiplicada por sua distribuição a priori:

$$\pi(\hat{\theta}|\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}, E) \propto \pi(\mathcal{D}_{\text{sim}}, \mathcal{D}_{\text{field}}|\hat{\theta}, E)\pi(\hat{\theta}|E). \tag{3.9}$$

Frequentemente, a parte mais difícil de qualquer calibração é a especificação da verossimilhança, achar a distribuição a posteriori é em teoria somente o cálculo de uma integral. Na prática, isso em geral requer aplicação cuidadosa de técnicas de integração numérica tais como o algoritmo de Monte Carlo via Cadeias de Markov (MCMC). Uma vez que tenhamos assumido distribuições para δ, ϵ e possivelmente $\eta(\hat{\theta})$, a estrutura mostrada na Figura 3.6 nos permite calcular a função de verossimilhança para o dado.

3.2.2 Erro do modelo

Existem várias razões porque simuladores são quase sempre representações imperfeitas dos sistemas físicos que querem predizer. Por exemplo, o entendimento do pesquisador sobre o sistema pode estar equivocado, ou talvez nem todos os processos físicos foram incluídos na análise, ou talvez existem inacurácias numéricas na solução das equações do modelo, e assim por diante. Se desejamos fazer predições acuradas é importante acomodar esse erro, caso contrário, podemos ter um nível de confiança não realístico nas nossas predições.

Se escolhemos não usar um termo de erro no modelo fazemos a suposição que observações são a melhor predição do simulador mais ruído branco, i.e.,

$$\mathcal{D}_{\text{field}} = f(\hat{\theta}) + \epsilon$$

onde ϵ_t é independente de ϵ_s para $t \neq s$. Frequentemente, a magnitude do erro de medida associado com o instrumento de mensuração é insuficiente para acomodar a variabilidade observada no sistema, levando a um ajuste pobre do modelo na calibração. Uma solução pode ser inflacionar a variância do erro para acomodar essa falta de variabilidade, entretanto, isso também causará problemas no nível de confiança na predição. Por outro lado, a estrutura de erro branco de ϵ, o termo de erro do modelo δ geralmente tem uma estrutura mais rica, usualmente com δ_t altamente correlacionado com δ_s para t perto de s. Usando erros correlacionados uma acurácia maior pode ser obtida.

Uma consequência de usar um simulador imperfeito na calibração, é que os parâmetros do modelo podem não corresponder aos seus correspondentes físicos. Por exemplo, num simulador da circulação no oceano podemos ter parâmetros chamados viscosidade e pode ser possível conduzir um experimento para medir o valor físico da viscosidade no laboratório. Entretanto, como o simulador é uma representação imperfeita do sitema, podemos concluir que usando o valor físico da viscosidade leva a predições mais pobres do que usando um valor determinado por calibração estaística. O fato que

parâmetros do modelo podem não ser parâmetros físicos deve ser bem enfatizado para especialistas quando elicitando a distribuição a priori $\pi(\hat{\theta}|E)$ presente na equação (3.9).

A questão de como escolher um modelo adequado para δ é uma área ativa de pesquisa estatística, e a abordagem tomada depende da quantidade de dados disponível. Em situações onde dado é farto, métodos direcionados pelos dados podem ser usados com especificação de prioris não informativas para δ. Então, por exemplo, em previsão do tempo, cada dia uma previsão é feita e no próximo dia dados são coletados e podem ser usados para validar as previsões dos dias anteriores. Em situações como essa, Kennedy and O'Hagan (2001) sugerem o uso de processos gaussianos para δ, com prioris não informativas em qualquer parâmetro em δ.

Se o dado disponível é limitado, então a opinião do especialista se torna importante. Goldstein and Rougier (2009) introduziram a idéia de um modelo concretizado (*reified model*) num experimento pensado para ajudar a elicitar crenças sobre o erro do modelo. O modelo concretizado é a versão do modelo que rodaríamos se tivéssemos recursos computacionais ilimitados. Então, por exemplo, em modelos para o clima global a superfície da Terra é dividida em uma grade e a computação assume cada célula da grade como homogênea. Se os recursos computacionais infinitos estiverem disponíveis poderiamos fazer o tamanho da célula tender a zero, resultando um contínuo de pontos no globo. Apesar de ser uma impossibilidade, pensar sobre o modelo concretizado nos ajuda a quebrar o erro do modelo em pedaços mais fácies de lidar. Nós podemos considerar a diferença entre o modelo computacional e o modelo concretizado, e então a diferença entre o modelo concretizado e a realidade. Essa abordagem pode ser uma maneira para ajudar pesquisadores a pensar mais cuidadosamente sobre $\delta(\cdot)$. Murphy et al. (2007) tomam uma direção diferente e usam cenários dos modelos. Eles olham para as predições do modelo calibrado em um coleção de diferentes modelos climáticos e os usam para acessar qual o erro do modelo em seu modelo.

3.2.3 Incerteza no código

Se o modelo computacional é rápido de rodar, então podemos assumir que seu valor é conhecido para todas as configurações possíveis do input, como em qualquer procedimento de inferência podemos avaliar o modelo sempre que seu valor for necessário. Nesse caso, o cálculo da posteriori de calibração

$$\pi(\theta|\mathcal{D}_{\text{field}}, f, E) \propto \pi(\mathcal{D}_{\text{field}}, |\theta, f, E)\pi(\theta|E), \tag{3.10}$$

onde f representa o modelo computacional, é relativamente fácil já que a abordagem de calibração (3.8) diz que

$$\mathcal{D}_{\text{field}} - f(\hat{\theta}) = \delta + \epsilon.$$

Dados as distribuições para a discrepância do modelo δ e erro de medida ϵ podemos calcular a verossimilhança do dado de campo, e então encontrar a distribuição a posteriori. Se o modelo não é rápido de rodar, então o valor do modelo é desconhecido em todos os valores do input diferentes dos valores do planejamento D. Essa incerteza sobre o output do modelo em configurações não usadas do input é usualmente chamada *incerteza do código*. Para acomodar essa fonte de incerteza na calibração usamos o método de emulação descrito no Capítulo 1 para fornecer um modelo estatístico que descreva nossas crenças sobre o output para todos os possíveis valores do input.

3.2.4 Exemplo: modelo climático (continuação)

Nós agora continuamos o exemplo de modelo climático introduzido na Seção 2.5 e descrevemos como usar o emulador de componentes principais para calibrar o modelo de UVic para a sequência de Keeling and Whorf (2005) de medições de CO_2.

Lembre que nosso objetivo é encontrar a distribuição de $\hat{\theta}$ dadas as observações e as rodadas do modelo,

$$\begin{aligned}\pi(\hat{\theta}|\mathcal{D}_{\text{field}}, \mathcal{D}_{\text{sim}}) &\propto \pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \hat{\theta})\pi(\hat{\theta}|\mathcal{D}_{\text{sim}}) \\ &\propto \pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \hat{\theta})\pi(\hat{\theta})\end{aligned}$$

onde notamos que $\pi(\mathcal{D}_{\text{sim}}|\hat{\theta}) = \pi(\mathcal{D}_{\text{sim}})$, então pode ser ignorado na distribuição a posteriori de $\hat{\theta}$, deixando $\pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \hat{\theta})$ a ser especificado para encontrarmos a posteriori. A metodologia de calibração dada na equação (3.8) contém três termos diferentes, $\eta(\hat{\theta}), \delta$ e ϵ, os quais precisamos no modelo. A abordagem do melhor input garante que o parâmetro $\hat{\theta}$ é escolhido de forma que $\eta(\hat{\theta})$ e δ são independentes para todo t (Kennedy and O'Hagan 2001), e o erro de medida ϵ também é independente de ambos os termos. Isso nos permite especificar a distribuição de cada parte da equação (3.8), e então calcular a distribuição da soma das três componentes. Se todas as três partes têm distribuição gaussiana, a soma terá também distribuição gaussiana. Escolhas de distribuições para ϵ e δ serão específicas para cada problema individual, mas frequentemente assume-se que os erros de medida têm média zero e distribuição gaussiana, usualmente com variâncias reportadas com os dados.

Nós devemos encontrar a distribuição de $\eta(\hat{\theta}, t)$ usando o emulador de componentes principais. Antes de considerar o mapeamento de Θ para $\mathcal{Y}$, devemos considerar a distribuição do emulador $\eta^{pc}(\cdot)$ de Θ para $\mathcal{Y}^{pc}$. Usando processos gaussianos independentes para modelar o mapeamento do espaço dos inputs para cada dimensão do subespaço principal (i.e., $\eta^{pc} = (\eta_1^{pc}, \ldots, \eta_{n^*}^{pc})$), nós temos que a distribuição a priori para $\eta_i^{pc}(\cdot)$ é

$$\eta_i^{pc}(\cdot)|\beta_i, \sigma_i^2, \lambda_i \sim GP(g_i(\cdot), \sigma_i^2 c_i(\cdot, \cdot)).$$

Se usamos para β_i uma priori uniforme imprópria $\pi(\beta_i) \propto 1$, podemos condicionar em $\mathcal{D}_{\text{sim}}$ e integrar β_i para achar

$$\eta_i^{pc}(\cdot)|\mathcal{D}_{\text{sim}}, \sigma_i^2, \lambda_i \sim GP(g_i^*(\cdot), \sigma_i^2 c_i^*(\cdot, \cdot))$$

onde

$$g_i^*(\theta) = \hat{\beta}^T h(\theta) + t(\theta)^T A^{-1}(Y_{\cdot i}^{pc} - H\hat{\beta}) \tag{3.11}$$

$$c_i^*(\theta, \theta') = c(\theta, \theta') - t(\theta)^T A^{-1} t(\theta') + (h(\theta)^T - t(\theta)^T A^{-1} H)(H^T A^{-1} H)^{-1} \times (h(\theta')^T - t(\theta')^T A^{-1} H)^T \tag{3.12}$$

e

$$\begin{aligned} \hat{\beta}_i &= (H^T A^{-1} H)^{-1} H^T A^{-1} Y_{\cdot i}^{pc} \\ t(\theta) &= (c(\theta, \theta_1), \ldots, c(\theta, \theta_N)) \\ \{A_i\}_{jk} &= \{c_i(\theta_j, \theta_k)\}_{j,k=1,\ldots,N} \\ H^T &= (h(\theta_1), \ldots, h(\theta_N)) \end{aligned}$$

assumindo que os regressores, $h(\cdot)$, são os mesmos para cada dimensão. Aqui, $Y_{\cdot i}^{pc}$ denota a i-ésima coluna da matriz Y^{pc}, e $\theta_1, \ldots, \theta_N$ são os pontos de planejamento D. A reconstrução para o espaço cheio, $\eta^e(\cdot) = \eta^{pc}(\cdot)V_1^T + \Phi V_2^T$, tem distribuição a posteriori

$$\eta^e(\theta)|\mathcal{D}_{\text{sim}}, \sigma^2, \lambda \sim N(g^*(\theta)V_1^T, \sigma^2 c^*(\theta, \theta) V_1 V_1^T + V_2 \Gamma' V_2^T)$$

onde $g^* = (g_1^*, \ldots, g_{n^*}^*)$ e $\Gamma' = \text{diag}(\Gamma_{n^*+1,n^*+1}, \ldots, \Gamma_{n,n})$. O método de empirical Bayes pode ser usado para os parâmetros de suavidade fixando-os nas estimativas de máxima verossimilhança. Nós não integramos σ^2 analiticamente por razões de tratabilidade, mas os integramos numericamente mais tarde usando MCMC.

Kennedy and O'Hagan (2001) recomendam o uso de processos gaussianos a priori para a função de discrepância δ. Enquanto isso é matematicamente

conveniente, é necessário especificar formas para a função de discrepância para cada caso separadamente. Se alguma forma não gaussiana é usada então pode ser difícil calcular a função de verossimilhança na Equação (3.10). Por clareza, assumimos que δ é um processo gaussiano. Usamos um processo auto-regressivo de ordem um para o termo de discrepância $\delta_t = \rho\delta_{t-1} + U$ onde $U \sim N(0, \sigma_\delta^2)$. Especificamos distribuições a priori para os parâmetros desconhecidos ρ e σ_δ^2 e fazemos inferência usando suas distribuições a posteriori. Usamos para ρ uma priori $\Gamma(5, 1)$ truncada no um, e para σ_δ^2 uma priori $\Gamma(4, 0.6)$

Se todas as três partes da equação (3.8) são gaussianas então podemos escrever a verossimilhança dos dados de campo condicional nos parâmetros:

$$\pi(\mathcal{D}_{\text{field}}|\mathcal{D}_{\text{sim}}, \sigma^2, \theta, \gamma_\delta)$$

one γ_δ são parâmetros necessários para a função de discrepância $\delta(t)$. Nós elicitamos as distribuições a priori para θ e γ_δ dos pesquisadores e decidimos nós mesmos as distribuições a priori para σ^2 (parâmetros dos emuladores são responsabilidade das pessoas fazendo emulação). Então usamos MCMC para encontrar as distribuições a posteriori. É possível escrever um algoritmo de Gibbs com Metropolis para acelerar os cálculos, mas não fornecemos os detalhes aqui.

Figura 3.7 mostra as distribuições a priori marginais obtidas da calibração do modelo de UVic as observações de Keeling and Whorf (2005). As cadeias de Markov têm 1000000 de iterações. As primeiras 200000 amostras foram discartadas como burn-in e as restantes foram tomadas a cada 80000. Distribuições uniformes foram usadas para Q_{10} e K_c ($Q_{10} \sim U[1, 4]$ e $K_c \sim U[0.25, 1.75]$), e $\Gamma(1.5, 6)$ foram usadas para cada variância do emulador σ^2. Testes foram feitos para chacar a sensitividade dos resultados a escolha da distribuição a priori, e a análise foi robusta a mudanças na priori para σ^2 e γ_δ, mas não a mudanças nas prioris para Q_{10} e K_c.

Isso usualmente não é o fim do processo de calibração. Os resultados retornam para os pesquisadores, que podem decidir usá-los para melhorar o modelo, antes que outra calibração seja feita. Para detalhes desse problema, e do algoritmo de Gibbs dentro do Metropolis veja Ricciuto et al. (2010).

3.3 Modelos computacionais estocásticos e inferência sem verossimilhança

Nas seções anteriores, nós mostramos como emuladores podem ser usados para calibrar modelos computacionais determinísticos caros. Mas o que

fazemos se o modelo é rápido de rodar? Então, não parece necessário fazer o esforço de emular o modelo. Entretanto, a metodologia de emulação nos forneceu uma maneira de encontrar um modelo estatístico para o output do modelo dado seus inputs, permitindo usar inferência bayesiana (via MCMC) para achar a distribuição a posteriori dos parâmetros.

Computação bayesiana aproximada (ou ABC), é o nome dado a coleção de algoritmos de Monte Carlo que têm sido desenvolvidos para fazer estimação dos parâmetros de forma rápida em modelos computacionais quando somos capazes de simular os modelos. Nessa seção, nós introduzimos essa nova área de pesquisa estatística, e descrevemos qual o atual estado da pesquisa desses algoritmos.

3.3.1 Uma curta introdução a computação bayesiana aproximada

Abordagens estatísticas para inferência em modelos estocásticos têm tradicionalmente se baseado no uso da função de verossimilhança $\mathbb{P}(\mathcal{D} \mid \theta)$. A abordagem frequentista é baseada na estimação de máxima verossimilhança na qual o objetivo é encontrar $\hat{\theta} = \arg\max_\theta \mathbb{P}(\mathcal{D}|\theta)$, enquanto a abordagem bayesiana é baseada na distribuição a posteriori $\pi(\theta|\mathcal{D}) \propto \mathbb{P}(\mathcal{D}|\theta)\pi(\theta)$. Se a verossimilhança é desconhecida, ambas abordagens podem ser impossíveis, e precisaremos fazer inferência sem verossimilhança.

Existe um grande número de modelos para os quais a função de verossimilhança é desconhecida, mas para as quais é possível simular conjuntos de dados do modelo, $\mathbb{P}(\cdot|\theta)$. Técnicas de inferência *sem verossimilhança* que podem ser aplicadas a essas situações têm sido desenvolvidas na última década e são frequentemente chamadas métodos de *Computação Bayesiana Aproximada* (ABC). O primeiro uso das idéias de ABC foi na genética, como desenvolvido por Tavaré et al. (1997) e Pritchard et al. (1999). Métodos ABC se tornaram populares nas ciências biológicas com aplicações em genética (veja, por exemplo, Siegmund et al. (2008); Foll et al. (2008)), epidemiologia (Tanaka et al. (2006)) e biologia populacional (Ratmann et al. (2007); Wilkinson and Tavaré (2009); Cornuet et al. (2008)).

O algoritmo básico é baseado no método da rejeição, mas tem sido extendido de várias maneiras. Marjoram et al. (2003) sugeriu um algoritmo de MCMC aproximado, Sisson et al. (2007) um método de amostragem por importância sequêncial, e Beaumont et al. (2002) sugeriu que a acurácia pode ser melhorada usando regressão nos outputs. Métodos ABC são, como o nome sugere, aproximados. Entretanto, pode ser mostrado que eles podem ser considerados como resultando em inferência exata, mas sob um modelo

incorretamente especificado (Wilkinson 2009).

3.3.2 Algoritmo ABC básico

Suponha que temos dados discretos $\mathcal{D}$ que assumimos ser do modelo $\mathcal{M}$, parametrizado por parâmetro θ (possivelmente multidimensional), e que θ tem distribuição a priori $\pi(\theta)$. Nosso principal objetivo aqui é encontrar a distribuição a posteriori do parâmetro dado os dados:

$$\pi(\theta|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|\theta)\pi(\theta)}{\mathbb{P}(\mathcal{D})}.$$

Para a maioria dos modelos computacionais não podemos calcular o termo da verossimilhança $\mathbb{P}(\mathcal{D}|\theta)$ ou a verossimilhança marginal $\mathbb{P}(\mathcal{D})$. Mas podemos simular conjuntos de dados $\mathcal{D}$ do modelo.

O algoritmo mais simples de computação bayesiana aproximada (Pritchard et al. 1999) é baseado no algoritmo de rejeição. Esse foi primeiro mostrado em von Neumann (1951) e é descrito como:

Algoritmo A: Algoritmo de rejeição 1

A1 Amostre θ de $\pi(\cdot)$,

A2 Aceite θ com probabilidade $r = \mathbb{P}(\mathcal{D}|\theta)$.

Como acabamos de descrever, esse algoritmo não é de muita ajuda. Para a classe de modelos que estamos interessados, a verossimilhança $\mathbb{P}(\mathcal{D}|\theta)$ é desconhecida. Entretanto, existe uma versão alternativa desse algoritmo que não depende explicitamente da verossimilhança, mas requer que simulemos do modelo:

Algoritmo B: Algoritmo de rejeição 2

B1 Amostre θ de $\pi(\cdot)$,

B2 Simule dados $\mathcal{D}'$ de $\mathbb{P}(\cdot|\theta)$,

B3 Aceite θ se $\mathcal{D} = \mathcal{D}'$.

O algoritmo B pode ser pensado como uma versão mecânica do teorema de Bayes. Para ver que esses dois algoritmos resultam numa amostra da posteriori, note que para I a função indicadora de θ é aceito ($I = 1$ se θ é aceito, e 0 caso contrário). Então,

$$\mathbb{P}(I=1) = \int \mathbb{P}(I=1|\theta)\pi(\theta)\mathrm{d}\theta = \int \mathbb{P}(\mathcal{D}|\theta)\pi(\theta)\mathrm{d}\theta = \mathbb{P}(\mathcal{D}),$$

que resulta em

$$\mathbb{P}(\theta|I=1) = \frac{\mathbb{P}(I=1|\theta)\pi(\theta)}{\mathbb{P}(I=1)} = \pi(\theta|\mathcal{D})$$

como requerido.

A taxa de aceitação de ambos algoritmos é a constante de normalização $\mathbb{P}(I=1) = \mathbb{P}(\mathcal{D})$. Então, para obter uma amostra de tamanho n temos que esperar um período de tempo que tem distribuição binomial negativa, com parâmetros n e $\mathbb{P}(\mathcal{D})$. O número médio de conjuntos de dados que precisaremos simular é $n(1-\mathbb{P}(\mathcal{D}))/\mathbb{P}(\mathcal{D})$, que é grande quando $\mathbb{P}(\mathcal{D})$ é pequeno. Isso nos leva a um problema: para problemas complexos a probabilidade que um conjunto de dados simulados, $\mathcal{D}'$, seja igual aos dados observados, $\mathcal{D}$, será muito pequena, e dessa forma Algoritmo B será extremamente lento.

Uma solução para esse problema é relaxar a exigência de igualdade no passo B3, e aceitar valores de θ quando o dado simulado é 'próximo' do dado real. Para definir 'próximo' precisamos de uma métrica ρ on espaço de estados $\mathcal{X}$, com $\rho(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, e uma *tolerância* ϵ. O algoritmo, nosso primeiro algoritmo de computação bayesiana aproximada, é dado por:

Algoritmo C: ABC 1

C1 Amostre θ de $\pi(\cdot)$,

C2 Simule dados $\mathcal{D}'$ de $\mathbb{P}(\cdot|\theta)$,

C3 Aceite θ se $\rho(\mathcal{D}, \mathcal{D}') \leq \epsilon$.

Ao contrário dos algoritmos A e B, algoritmo C não é exato. Valores aceitos de θ não formam uma amostra da distribuição a posteriori, mas sim de alguma distribuição que é uma aproximação para a posteriori. A acurácia do algoritmo (medida por alguma medida de distância) depende de ϵ de forma não trivial (cf. Seção 3.3.3). Faça $\pi_\epsilon(\theta)$ denotar a distribuição dos valores aceitos de θ quando usamos tolerância ϵ. A distribuição de $\pi_\epsilon(\theta)$ também depende da escolha da métrica, mas isso não é feito explicito na notação. Se $\epsilon = 0$, então somente aceitamos valores gerados de θ se o dado simulado é exatamente igual ao dado observado. Nesse caso, Algoritmo C é equivalente ao Algoritmo B e gera uma amostra da distribuição a posteriori,

i.e., $\pi_0(\theta) = \pi(\theta|\mathcal{D})$ e o algoritmo é exato[1]. Se $\epsilon = \infty$, todos os valores de θ são aceitos independentemente do dado simulado. Nesse caso, teremos uma amostra da distribuição a priori, i.e., $\pi_\infty(\theta) = \pi(\theta)$.

Note que a aproximação no passo C3 significa que o algoritmo pode também ser usado para modelos que simulam dados contínuos. Continuaremos usando $\mathbb{P}(\mathcal{D}|\theta)$ para denotar a verossimilhança do modelo, apesar de poder também representar uma função de densidade ao invés de probabilidade.

A tolerância ϵ representa um compromisso entre computabilidade e acurácia; valores grandes de ϵ significará mais aceitações no passo C3 acima e nos permitirá gerar amostras mais rapidamente. Porém, a distribuição obtida, $\pi_\epsilon(\theta)$, pode ser uma aproximação pobre para $\pi(\theta|\mathcal{D})$. Valores pequenos de ϵ significarão que nossa aproximação é mais acurada, mas a taxa de aceitação no passo C3 será menor e será necessário um tempo maior para gerar uma amostra de um dado tamanho. Para escolher um valor de ϵ, devemos decidir por um equilíbrio entre o tempo computacional disponível e a acurácia desejada. Se tivermos um grande cluster de computadores disponível para uso, seremos capazes de usar um valor menor de ϵ do que quando temos apenas um desktop.

3.3.3 Exemplo

Agora ilustramos a idéia básica com um exemplo da normal para o qual podemos calcular tudo analiticamente. Isso nos permitirá analisar mais de perto os efeitos do algoritmo ABC. Suponha que $X_1, \ldots, X_n \sim N(\mu, \sigma^2)$ são uma amostra independente da distribuição normal com variância conhecida σ^2. Por simplicidade suponha que a média μ tem distribuição uniforme $[a, b]$. A distribuição a posteriori de μ, dado od dados, pode ser encontrada como:

$$\begin{aligned} \pi(\mu|\mathcal{D}) &\propto \mathbb{I}_{a\leq\mu\leq b} \exp\left(-\frac{\sum(x_i-\mu)^2}{2\sigma^2}\right) \\ &\propto \mathbb{I}_{a\leq\mu\leq b} \exp\left(-\frac{n}{2\sigma^2}(\mu-\bar{x})^2\right). \end{aligned}$$

Então a distribuição a posteriori de μ é uma distribuiçõa normal truncada (truncada fora de $[a, b]$) com média $\bar{x}$ e variância σ^2/n. Se a distribuição a priori para μ é a priori não informativa imprópria $\pi(\mu) \propto 1$, $\mu \in \mathbb{R}$, então a distribuição a posteriori será $\mu|\mathcal{D} \sim \text{Normal}(\bar{x}, \frac{\sigma^2}{n})$.

[1]Note que estamos assumindo que $\rho(\cdot,\cdot)$ é uma métrica, então $\rho(\mathcal{D}, \mathcal{D}') = 0$ implica $\mathcal{D} = \mathcal{D}'$. Em geral, isso nõa será verdade, já que é necessário resumir os dados se este for de alta dimensão. Nesse caso, ρ será uma função de distância, mas não uma métrica. Veja Seção 3.3.4 para mais detalhes.

A média dos dados é suficiente para μ (cf. Seção 3.3.4 para detalhes de resumo dos dados) e então podemos comparar os conjuntos de dados comparando suas médias. Podemos assumir sem perda de generalidade que o dado tem média zero. Usando a métrica da distância absoluta $\rho(\mathbf{x}, \mathbf{x}') = |\bar{\mathbf{x}} - \bar{\mathbf{x}}'|$ resulta no algoritmo:

- Escolha μ da distribuição a priori,
- Simule $X_1', \ldots, X_n'$ da $N(\mu, \sigma^2)$,
- Aceite μ if $|\bar{x}' - 0| \leq \epsilon$.

Figura 3.8 mostra gráficos da distribuição amostral obtida desse algoritmo, juntamente com as distribuições a posteriori verdadeiras para vários valores de ϵ. As linhas tracejadas são as densidades obtidas usando estimativas dos kernels em 1000 amostragens sucessivas do algoritmo acima. Note que para valores pequenos de ϵ a aproximação é excelente, mas para valores grandes de ϵ uma aproximação mais dispersa, como esperado.

Para a distribuição normal com prioris uniformes, cálculo de $\pi_\epsilon(\theta)$ é possível. Podemos escrever a densidade aproximada numa forma quase explícita:

$$\begin{aligned}\pi_\epsilon(\mu) &= \frac{\int_{-\epsilon}^{\epsilon} \sqrt{\frac{n}{2\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}y}{\int_{-\infty}^{\infty}\int_{-\epsilon}^{\epsilon} \sqrt{\frac{n}{2\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}y\mathrm{d}\mu} \\ &= \frac{\Phi\left(\frac{\epsilon-\mu}{\sqrt{\sigma^2/n}}\right) - \Phi\left(\frac{-\epsilon-\mu}{\sqrt{\sigma^2/n}}\right)}{2\epsilon}. \end{aligned} \tag{3.13}$$

onde Φ é a função de distribuição normal padrão. Também podemos calcular a média e a variância da distribuição aproximada:

$$\begin{aligned}\mathbb{E}(\mu||\bar{x}| \leq \epsilon) &= \int_{-\infty}^{\infty} \frac{\mu \mathbb{P}(|\bar{x}| \leq \epsilon|\mu)\pi(\mu)}{\mathbb{P}(|\bar{x}| \leq \epsilon)} \mathrm{d}\mu \\ &= \frac{\int_{-\infty}^{\infty} \mu \int_{-\epsilon}^{\epsilon} \sqrt{\frac{n}{2\pi\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}y\mathrm{d}\mu}{\int_{-\infty}^{\infty}\int_{-\epsilon}^{\epsilon} \sqrt{\frac{n}{2\pi\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}y\mathrm{d}\mu} \\ &= \frac{\int_{-\epsilon}^{\epsilon}\int_{-\infty}^{\infty} \mu \sqrt{\frac{n}{2\pi\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}\mu\mathrm{d}y}{\int_{-\epsilon}^{\epsilon}\int_{-\infty}^{\infty} \sqrt{\frac{n}{2\pi\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}\mu\mathrm{d}y} \text{ pelo Teorema de Fubini} \\ &= 0 \end{aligned}$$

$$\mathbb{V}\mathrm{ar}(\mu|\rho(\mathcal{D}, \mathcal{D}') \leq \epsilon) = \frac{\int_{-\infty}^{\infty} \mu^2 \int_{-\epsilon}^{\epsilon} \sqrt{\frac{n}{2\pi\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}y\mathrm{d}\mu}{\int_{-\infty}^{\infty}\int_{-\epsilon}^{\epsilon} \sqrt{\frac{n}{2\pi\sigma^2}} e^{-\frac{n}{2\sigma^2}(y-\mu)^2} \mathrm{d}y\mathrm{d}\mu} = \frac{\sigma^2}{n} + \frac{1}{3}\epsilon^2.$$

Isso mostra que a estimativa da média a posteriori é não viciada nesse caso. Podemos ver também que a variância a posteriori aproximada cresce de forma quadrática em ϵ. essa é uma descrição matemática precisa da super-dispersão de $\pi_\epsilon(\mu)$ observada acima.

Nós podemos examinar o erro entre a aproximação e a distribuição a posteriori verdadeira usando expansão de Taylor na Equação (3.13). Seja $x = -\mu/\sqrt{\sigma^2/n}$ e $h = \epsilon/\sqrt{\sigma^2/n}$ e a expansão de Taylor para a função gaussiana em torno de x para h pequeno leva a:

$$\Phi\left(\frac{\epsilon-\mu}{\sqrt{\sigma^2/n}}\right) = \Phi(x+h) = \Phi(x) + h\Phi'(x) + \frac{h^2\Phi''(x)}{2!} + \frac{h^3\Phi'''(x)}{3!} + o(h^3)$$

so that

$$\begin{aligned}\pi_\epsilon(\mu) &= \frac{\Phi(x+h)-\Phi(x-h)}{2\epsilon} \\ &= \frac{1}{\epsilon}\left(h\Phi'(x) + \frac{h^3}{6}\Phi'''(x)\right) + o(h^2)\end{aligned}$$

Como $\Phi'(x) = \phi(x)$, i.e., a função densidade de probabilidade normal padrão, podemos ver que a aproximação de primeira ordem é exata:

$$\pi_\epsilon(\mu) = \pi(\mu|\mathcal{D}) + o(\epsilon).$$

Usando segunda ordem encontramos

$$\pi_\epsilon(\mu) = \left(1 - \frac{n\epsilon^2}{\sigma^2} + \frac{n^2\epsilon^2\mu^2}{\sigma^4}\right)\frac{1}{\sqrt{2\pi\sigma^2/n}}\exp\Big(-\frac{\mu^2 n}{2\sigma^2}\Big) + o(\epsilon^2).$$

A distância total de variação entre a distribuição a posteriori e a aproximação é definida por

$$d_{TV}(\pi_\epsilon(\mu), \pi(\mu|\mathcal{D})) = \frac{1}{2}\int |\pi(\mu|\mathcal{D}) - \pi_\epsilon(\mu)|\,\mathrm{d}\mu.$$

Podemos calcular

$$\begin{aligned}d_{TV}(\pi_\epsilon(\mu), \pi(\mu|\mathcal{D})) &= \frac{1}{2}\int_{-\infty}^{\infty}\left|\frac{n^2\epsilon^2\mu^2}{\sigma^4} - \frac{n\epsilon^2}{\sigma^2}\right|.\frac{\exp(-\frac{\mu^2 n}{2\sigma^2})}{\sqrt{2\pi\sigma^2/n}}\,\mathrm{d}\mu \\ &= \frac{n\epsilon^2}{2\sigma^2}\int_{-\infty}^{\infty}|x^2-1|\frac{\exp(-x^2/2)}{\sqrt{2\pi}}\mathrm{d}x\end{aligned}$$

$$= \frac{n\epsilon^2}{2\sigma^2}\left(\int_{-1}^{1}(1-x^2)\frac{\exp(-x^2/2)}{\sqrt{2\pi}}\mathrm{d}x + 2\int_{1}^{\infty}(x^2-1)\frac{\exp(-x^2/2)}{\sqrt{2\pi}}\mathrm{d}x\right)$$
$$= \sqrt{\frac{2}{\pi}}\exp(-1/2)\frac{\epsilon^2}{\sigma^2/n}$$

Então, a distância total de variação entre as duas distribuições é aproximadamente

$$\frac{cn\epsilon^2}{\sigma^2} + o(\epsilon^2)$$

onde $c = \sqrt{2/\pi}\exp(-1/2) \approx 1/2$. Então, para um dado tamanho do erro, o valor de tolerância que temos que usar depende do tamanho da variância a posteriori σ^2/n. Para pequenas variâncias a posteriori, devemos usar um valor menor de ϵ, enquanto que para valores grandes da variância podemos usar valores grandes de ϵ.

3.3.4 Resumo dos dados

Para problemas com muitos conjuntos de dados com alta dimensão, o Algoritmo C será impraticável, já que o dado simulado nunca será muito próximo dos dados observados. A abordagem padrão para reduzir o número de dimensões é usar estatísticas resumo (possivemente multidimensionais), digamos $S(\mathcal{D})$, as quais devem resumir as partes importantes dos dados. Então adaptamos Algoritmo C de forma que valores do parâmetro são aceitos se a estatística resumo está 'perto' o suficiente do resumo para os dados reais.

Algoritmo D: ABC 2

D1 Amostre θ de $\pi(\cdot)$,

D2 Simule dados $\mathcal{D}'$ de $\mathbb{P}(\cdot|\theta)$,

D3 Aceite θ se $\rho(S(\mathcal{D}), S(\mathcal{D}')) \leq \epsilon$.

A estatística $S(\mathcal{D})$ é chamada uma *estatística suficiente* para θ se e somente se $\pi(\theta|\mathcal{D}) = \pi(\theta|S(\mathcal{D}))$, i.e., se a distribuição condicional de θ dado o resumo é igual a distribuição a posteriori [2]. A idéia é que se $S(\mathcal{D})$ é conhecido, então o conjunto de dados completo, $\mathcal{D}$, não irá fornecer nenhuma informação extra sobre θ.

[2] Essa é a definição bayesiana de suficiencia. Uma definição equivalente é que S é suficiente se e somente se a distribuição condicional de $\mathcal{D}$ dado $S(\mathcal{D})$ não depende de θ.

Se uma estatística suficiente está disponível, então Algoritmo D é essencialmente o mesmo que Algoritmo C (no exemplo da distribuição normal acima, nós usamos Algoritmo D). Porém, para problemas onde ambas a distribuição a posteriori e a função de verossimilhança são desconhecidas, não será possível, em geral, determinar se uma estaística é suficiente. Ao invés disso, fica a cargo da intuição e experiência do pesquisador achar uma estatística resumo que funcione bem e capture os principais aspectos do dado. Para problemas difíceis, pode ser o caso de usar tentativa e erro, ao invés de seguir determinada estratégia.

3.3.5 Vantagens do ABC

As técnicas de computação bayesiana aproximada listadas aqui e em outros referências potencialmente têm vantagens sobre outras técnicas de Monte Carlo. Nessa seção listamos algumas dessas vantagens.

A primeira coisa a ser notada é que se a função de verossimilhança não está disponível ou é muito cara de calcular não seremos capazes de usar os métodos tradicionais de Monte Carlo. Entretanto, mesmo que a verossimilhança seja conhecida, os métodos ABC podem ser úteis como um primeiro passo no processo de inferência.

Muitas das técnicas de Monte Carlo podem ser difíceis de programar e ainda dependem de afinações, e em geral requerem extensivo conhecimento estatístico para serem usados efetivamente. Algoritmos de Monte Carlo via Cadeias de Markov por exemplo, têm várias dificuldades associadas: a cadeia pode não convergir, podem haver muitos parâmetros para serem ajustados de forma a controlar a mistura das cadeias, e o output recebido é dependente. A necessidade de ajustar tantos fatores significa que usar MCMC leva em geral bastante tempo já que a cada mudança no modelo, esses fatores precisam ser revistos para obtenção de resultados satisfatórios. Por outro lado, os métodos ABC não precisam ser reajustados após mudanças no modelo, o que significa que pode ser usado nos estágios de desenvolvimento do modelo quando ele ainda passa por mudanças [3]. Outra vantagem do ABC sobre o MCMC é que ABC pode rodar em paralelo em várias máquinas sem nenhuma consideração especial.

Os métodos ABC também permitem o uso de tipos diferentes de dados,

[3] Os algoritmos ABC requerem algumas afinações já que temos que escolher uma métrica e um valor para ϵ. Porém, isso pode ser feito depois das simulações terem sido rodadas. Se salvamos todos os outputs das simulações, podemos fazer um processamento posterior do dado tentando vários valores de ϵ e ρ. Isso nos permite escolher valores ótimos com um mínimo de computação.

até mesmo aqueles com estrutura de dependência complexa. Em muitos campos tais como epidemiologia, genética de populações e biologia evolucionária, existe uma estrutura não observada de árvore, que leva a dependências complexas nos dados. Combinar diferentes tipos de dados é simples quando usamos ABC e é feito mudando a métrica usada. Finalmente, a taxa de aceitação dos algoritmos ABC fornecem uma medida natural de ajuste do modelo que pode ser usada para estimar Fatores de Bayes que podem então ser usados na seleção de modelos.

3.3.6 Desvantagens e problemas

Os métodos ABC estão apenas começando. Existem muitas questões técnicas que ainda precisam ser respondidas. Por exemplo, atualmente não sabe-se quão acurada é a aproximação $\pi_\epsilon(\theta)$, ou como a acurácia depende na escolha da métrica e de ϵ. Outro problema é que se o resumo dos dados precisam ser usados, precisamos confiar na intuição do pesquisador para a escolha de uma boa estatística resumo. Idealmente, alguma noção de suficiência aproximada é necessária juntamente com uma maneira metódica de achar resumos que são próximo de suficientes e que capturam as partes importantes dos dados.

Porém, mesmo que essas duas questões técnicas sejam resolvidas satisfatoriamente, algumas disvantagens do ABC ainda permanecem. A mais séria delas, como apontado por Sisson (2007), é que devido a amostragem da distribuição a priori em cada rodada, esses algoritmos em geral são lentos quando comparados com técnicas de MCMC, especialmente quando o número de parâmetros cresce. Marjoram et al. (2003) tentou resolver essa questão com um algoritmo de MCMC aproximado, e em Wilkinson (2007) foi mostrado como MCMC e ABC podem ser combinados para problemas onde alguma computação é possível. A metodologia mais promissora em termos de eficiência parece ser a nova onda de filtro de partículas baseadas na distribuição a priori e então filtrada através de sucessivas aplicações do algoritmo ABC reduzindo a tolerância ϵ a cada rodada até um nível satisfatório ser obtido.

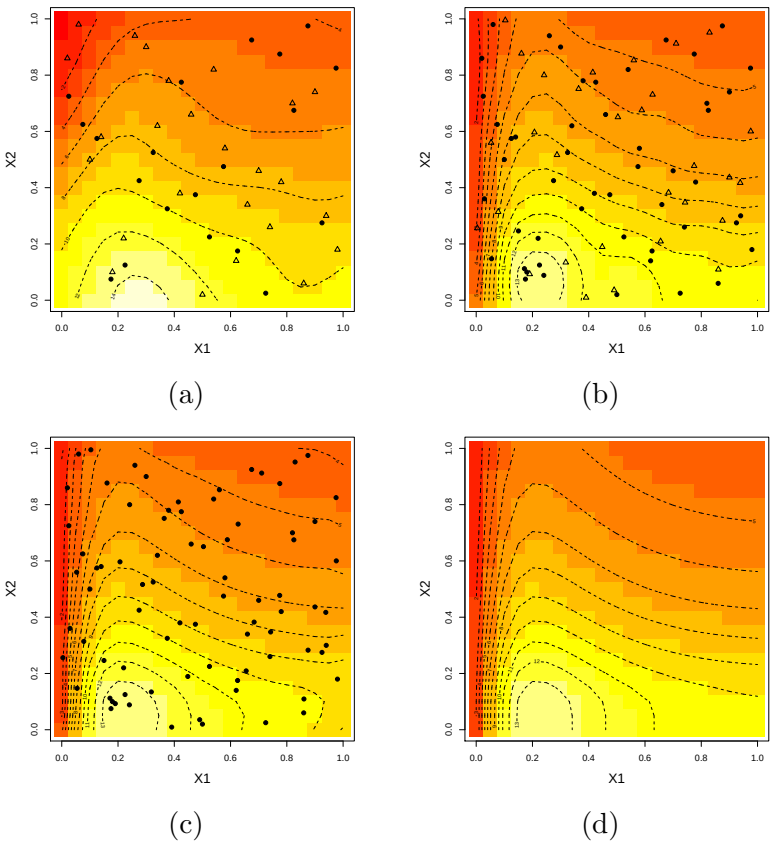


Figura 3.5: Média preditiva do emulador gaussiano construído com (a) o dado de treinamento original; (b) o dado de treinamento atualizado; (c) todas as observações como dado de treinamento; (d) Exemplo em duas dimensões no espaço dos inputs. Dados de treinamento (•) e dados de validação (△).

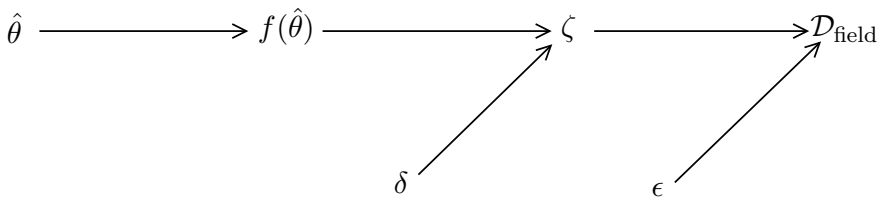


Figura 3.6: Rede de cranças Bayesiana mostrando as dependências entre as diferentes componentes no modelo estatístico. Note que a realidade separa a predição do modelo $\eta(\hat{\theta})$ das observações $\mathcal{D}_{\text{field}}$ e da discrepância do modelo δ. Veja Pearl (2000) para uma introdução em rede de cranças Bayesiana.

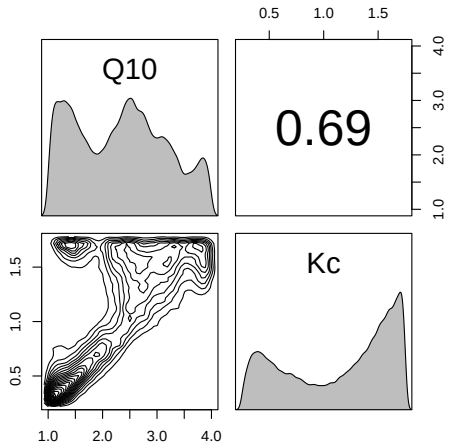


Figura 3.7: Distribuição a posteriori marginal para dois dos parâmetros de calibração, Q_{10} e K_c, no modelo climático de UVic. Os dois gráficos na diagonal principal mostram os gráficos individuais marginais. O gráfico a esquerda e abaixo mostra a distribuição marginal dois a dois, e o gráfico acima e a direita mostra a posteriori da correlação entre Q_{10} and K_c.

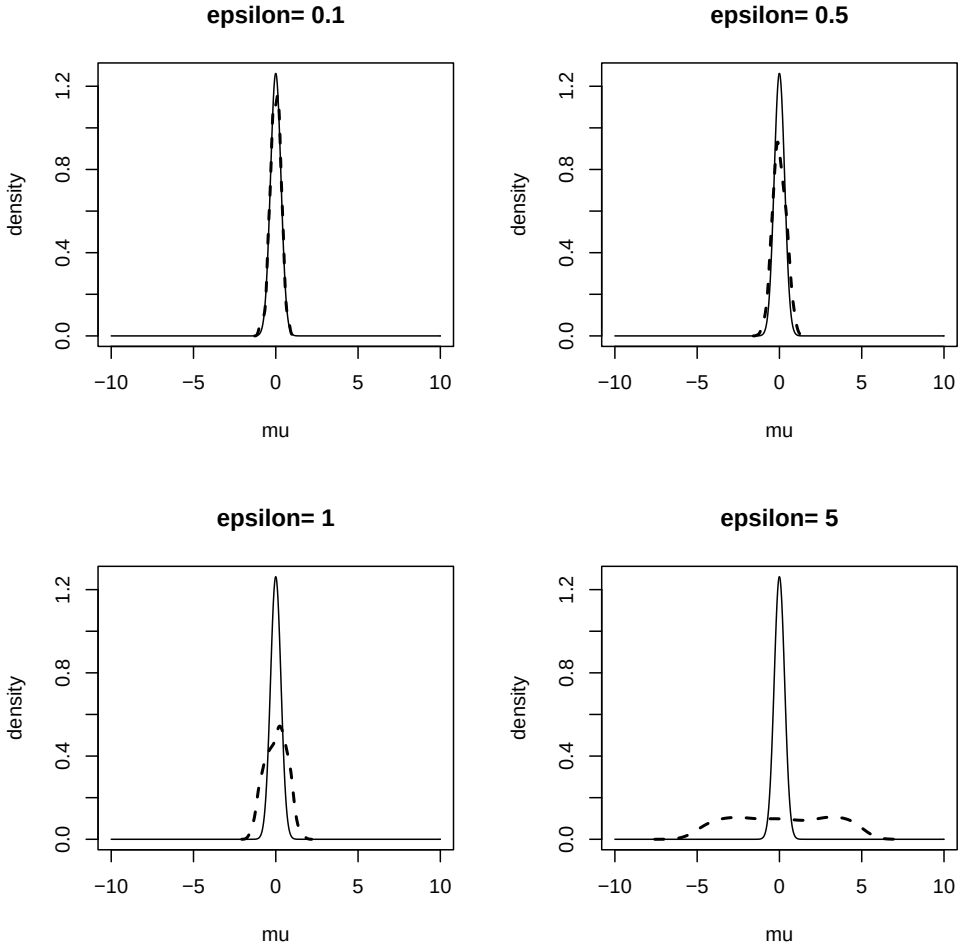


Figura 3.8: Gráficos das distribuições a posteriori verdadeiras para μ (linha cheia), e estimativa ABC usando 1000 amostras (linha tracejada). O valor de $\sigma^2 = 1$ foi usado para os quatro gráficos.

Bibliografia

Bastos, L. S. and O'Hagan, A. (2009), "Diagnostics for Gaussian process emulators," *Technometrics*, 51, 439–451.

Bates, R., Buck, R., Riccomagno, E., and Wynn, H. (1996), "Experimental design and observation for large systems," *Journal of the Royal Statistical Society B*, 58, 77–94.

Bates, R., Riccomagno, E., Schwabe, R., and Wynn, H. (1998), "The use of lattices in the design of high-dimensional experiments," *IMS Lecture Notes*, 34, 26–35.

Bayarri, M. J., Berger, J., Paulo, R., Sacks, J., Cafeo, J. A., Cavendish, J., Lin, C. H., and Tu, J. (2007), "A Framework for validation of computer models," *Technometrics*, 49, 138–154.

Beaumont, M. A., Zhang, W., and Balding, D. J. (2002), "Approximate Bayesian Computatation in Population Genetics," *Genetics*, 162, 2025–2035.

Benson, A. J., Frenk, P. C. S., Baugh, C. M., Cole, S., and Lacey, C. G. (2001), "The clustering evolution of the galaxy distribution," *Monthly Notices of the Royal Astronomical Society*.

Chapman, W. L., Welch, W. J., Bowman, K. P., Sacks, J., and Walsh, J. E. (1994), "Arctic Sea Ice Variability: Model Sensitivities and a Multidecadal Simulation," *Journal of Geophysical Research*, 99, 919–935.

Conti, S. and O'Hagan, A. (to appear 2010), "Bayesian Emulation of Complex Multi-Output and Dynamic Computer Models," *Journal of Statistical Planning and Inference*.

Cornuet, J. M., Santos, F., Beaumont, M. A., Robert, C. P., Marin, J. M., Balding, D. J., Guillemaud, T., and Estoup, A. (2008), "Inferring popu-

lation history with *DIY ABC*: a user-friendly approach to Approximate Bayesian Computation," *Bioinformatics*, 24, 2713–2719.

Cressie, N. (1993), *Statistics for Spatial Data*, New York: J. Wiley.

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1988), "A Bayesian Approach to the Design and Analysis of Computer Experiments," Tech. Rep. ORNL-6498, Oak Ridge National Laboratory.

— (1991), "Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, 86, 953–963.

Foll, M., Beaumont, M. A., and Gaggiotti, O. (2008), "An approximate Bayesian computation approach to overcome biases that arise when using amplified fragment length polymorphism markers to study population structure," *Genetics*, 179, 927–939.

Galanti, S. and Jung, A. (1997), "Low-Discrepancy Sequences: Monte Carlo Simulation of Option Prices," *Journal of Derivatives*, 63–83.

Garthwaite, P. H., Kadane, J. B., and O'Hagan, A. (2005), "Statistical methods for eliciting probability distributions," *Journal of the American Statistical Association*, 100, 680–701.

Goldstein, M. and Rougier, J. (2009), "Reified Bayesian modelling and inference for physical systems," *Journal of Statistical Planning and Inference*, 139, 1221–1239.

Golub, G. H. and van Loan, C. F. (1996), *Matrix computations*, Baltimore: Johns Hopkins University Press, 3rd ed.

Haylock, R. G. and O'Hagan, A. (1996), "On Inference for Outputs of Computationally Expensive Algorithms with Uncertainty on the Inputs," in *Bayesian Statistics 5*, ed. et al, J. M. B., Oxford University Press, pp. 629–637.

Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008), "Computer model calibration using high-dimensional output," *Journal of the American Statistical Association*, 103, 570–583.

Hills, R. G. and Trucano, T. G. (1999), "Statistical Validation of Engineering and Scientific Models: Background," Tech. Rep. SAND99-1256, Sandia National Laboratory.

— (2001), "Statistical Validation of Engineering and Scientific Models: A Maximum Likelihood Based Metric," Tech. Rep. SAND2001-1783, Sandia National Laboratory.

Houseman, E. A., Ryan, L. M., and Coull, B. A. (2004), "Cholesky Residuals for Assessing Normal Errors in a Linear Model With Correlated Outcomes," *Journal of the American Statistical Association*, 99, 383–394.

Jolliffe, I. T. (2002), *Principal Component Analysis*, Springer, 2nd ed.

Keeling, C. D. and Whorf, T. P. (2005), "Atmospheric CO_2 records from sites in the SIO air sampling network," Tech. rep., Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, U.S. Department of Energy, Tenn., U.S.A.

Kennedy, M. C. and O'Hagan, A. (2001), "A Bayesian calibration of computer models (with discussion)," *Journal of the Royal Statistical Society B*, 63, 425–464.

Kimeldorf, G. S. and Wahba, G. (1970), "A correspondence between Bayesian estimation on stochastic processes and smoothing by splines," *The Annals of Mathematical Statistics*, 41, 495–502.

Lindley, D. V. (1980), "Approximate Bayesian Methods," in *Bayesian Statistics*, eds. Bernardo, J. M., DeGroot, M. H., Lindley, D. V., and Smith, A. F. M., Valencia: Univ. Press, pp. 223–237.

Loeppky, J. L., Sacks, J., and Welch, W. J. (2009), "Choosing the Sample Size of a Computer Experiment: A Practical Guide," *Technometrics*, 51, 366–376.

Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. (2003), "Markov Chain Monte Carlo without likelihoods," *Proc. Natl. Acad. Sci. USA*, 100, 15324–15328.

McGrattan, K. B., Hostikka, S., and Floyd, J. E. (2007), "Fire Dynamics Simulator (Version 5), User's Guide," Nist special publication 1019-5, National Institute of Standards and Technology, Gaithersburg, Maryland, USA.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979), "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, 21, 239–245.

Meissner, K. J., Weaver, A. J., Matthews, H. D., and Cox, P. M. (2003), "The role of land surface dynamics in glacial inception: a study with the UVic Earth System Model," *Climate Dynamics*, 21, 515–537.

Morris, M. D. and Mitchell, T. J. (1997), "Exploratory designs for computer experiments," *Journal of Statistical Planinning and Inference*, 43.

Murphy, J. M., Booth, B. B. B., Collins, M., Harris, G. R., Sexton, D. M. H., and Webb, M. J. (2007), "A methodology for probabilitic predictions of regional climate change from perturbed physics ensembles," *Philosophical Transactions of the Royal Society A*, 1993–2028.

Neiderreiter, H. (1992), *Random Number Generation and Quasi-Monte Carlo Methods*, Philadelphia: SAIM.

Oakley, J. E. and O'Hagan, A. (2002), "Bayesian inference for the uncertainty distribution of computer model outputs," *Biometrika*, 89, 769–784.

— (2004), "Probabilistic sensitivity analysis of complex models: a Bayesian approach," *Journal of the Royal Statistical Society B*, 66, 751–769.

O'Hagan, A. (1978), "Curve fitting and optimal design for predictions," *Journal of the Royal Statistical Society B*, 40, 1–42.

— (1998), "A Markov property for covariance structures," Tech. Rep. 98-13, Nottingham University Statistics.

O'Hagan, A., Buck, C. E., Daneshkhah, A., Eiser, J. R., Garthwaite, P. H., Jenkinson, D. J., Oakley, J. E., and Rakow, T. (2006), *Uncertain Judgements: Eliciting Experts' Probabilities*, Wiley.

O'Hagan, A. and Haylock, R. G. (1997), *Bayesian uncertainty analysis and radiological protection*, Wiley: Chichester, pp. 109–128.

O'Hagan, A., Kennedy, M. C., and Oakley, J. E. (1998), "Uncertainty Analysis and other Inference Tools for Computer Codes," in *Bayesian Statistics 6*, eds. Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., Oxford University, pp. 503–524.

Paulo, R. (2005), "Default priors for Gaussian processes," *The Annals of Statistics*, 33, 556–582.

Pearl, J. (2000), *Causality: Models, Reasoning, and Inference*, Cambridge University Press.

Pritchard, J. K., Seielstad, M., Perez-Lezaun, A., and Feldman, M. W. (1999), "Population Growth of Human Y Chromosomes: A Study of Y Chromosome Microsatellites," *Molecular Biology and Evolution*, 16, 1791–1798.

Randall, D. A., Wood, R. A., Bony, S., Colman, R., Fichefet, T., Fyfe, J., Kattsov, V., Pitman, A., Shukla, J., Srinivasan, J., Stouffer, R. J., Sumi, A., and Taylor, K. E. (2007), "Climate Models and Their Evaluation," in *Climate Change 2007: the Physical Science Basis*, eds. Solomon, S., Qin, D., Manning, M., Marquis, M., Averyt, K., Tignor, M., Miller, H., and Zhenlin, C., Cambridge, United Kingdom and New York, NY, USA.: Cambridge University Press.

Rasmussen, C. E. and Willians, C. K. I. (2006), *Gaussian Processes for Machine Learning*, The MIT Press.

Ratmann, O., Jorgensen, O., Hinkley, T., Stumpf, M., Richardson, S., and Wiuf, C. (2007), "Using Likelihood-Free Inference to Compare Evolutionary Dynamics of the Protein Networks of *H. pylori* and *P. falciparum*," *PLos Comput. Biol.*, 3, 2266–2276.

Ricciuto, D., Tonkonojenkov, R., Urban, N., Wilkinson, R., Matthews, D., Davis, K., and Keller, K. (2010), "Assimilation of global carbon cycle observations into an Earth system model to estimate uncertain terrestrial carbon cycle parameters," *Global Biogeochemical Cycles*, To appear.

Rougier, J. (2008), "Efficient Emulators for Multivariate Deterministic Functions," *Journal of Computational and Graphical Statistics*, 17, 827–843.

Rougier, J., Guillas, S., Maute, A., and Richmond, A. D. (2009), "Expert Knowledge and Multivariate Emulation: The Thermosphere-Ionosphere Electrodynamics General Circulation Model (TIE-GCM)," *Technometrics*, 51, 414–424.

Sacks, J., Schiller, S. B., and Welch, W. J. (1989a), "Designs for Computer Experiments," *Technometrics*, 31, 41–47.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989b), "Design and Analysis of Computer Experiments," *Statistical Science*, 4, 409–423.

Saltelli, A., Chan, K., and Scott, M. (eds.) (2000), *Sensitivity Analysis*, New York, USA: Wiley.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (eds.) (2008), *Global Sensitivity Analysis: The Primer*, John Wiley and Sons.

Santner, T. J., Williams, B. J., and Notz, W. I. (2003), *The Design and Analysis of Computer Experiments*, New York: Springer-Verlag.

Siegmund, K. D., Marjoram, P., and Shibata, D. (2008), "Modeling DNA methylation in a population of cancer cells," *Stat. Appl. Genet. Mo. B.*, 7, article 18.

Sisson, S. A. (2007), "Genetics and stochastic simulation *do* mix!" *The American Statistician*, 61, 112–119.

Sisson, S. A., Fan, Y., and Tanaka, M. M. (2007), "Sequential Monte Carlo without Likelihoods," *Proc. Natl. Acad. Sci. USA*, 104, 1760–1765.

Stein, M. (1987), "Large Sample Properties of Simulations Using Latin Hypercube Sampling," *Technometrics*, 29, 143–151.

Tanaka, M. M., Francis, A. R., Luciani, F., and Sisson, S. A. (2006), "Using approximate Bayesian computation to estimate tuberculosis transmission parameters from genotype data," *Genetics*, 173, 1511–1520.

Tavaré, S., Balding, D. J., Griffiths, R. C., and Donnelly, P. (1997), "Inferring coalescence times for molecular sequence data," *Genetics*, 145, 505–518.

von Neumann, J. (1951), "Various Techniques used in Connection with Random Digits," *National Bureau of Standards Applied Mathematics Series*, 12, 36–38.

Wackernagel, H. (1995), *Multivariate Geostatistics*, Springer.

Wilkinson, R. D. (2007), "Bayesian inference of primate divergence times," Ph.D. thesis, University of Cambridge.

— (2009), "Approximate Bayesian computation gives exact results under the assumption of model error," *Submitted*.

— (2010), *Bayesian calibration of expensive multivariate computer experiments*, Wiley, p. To appear.

Wilkinson, R. D. and Tavaré, S. (2009), "Estimating primate divergence times by using conditioned birth-and-death processes," *Theor. Popul. Biol.*, 75, 278–285.

Zickfeld, K., Slawig, T., and Rahmstorf, S. (2004), "A low-order model for the response of the Atlantic thermohaline circulation to climate change," *Ocean Dynamics*, 54, 8–26.