Relatório Científico Final – Mestrado

Processo FAPESP 17/20575-9

Identificação de vias de sinalização celular baseada em repositórios de cinética de reações bioquímicas

Beneficiário: Gustavo Estrela de Matos

Responsável: Marcelo da Silva Reis

Relatório referente aos trabalhos desenvolvidos entre

10 de dezembro de 2018 e 31 de dezembro de 2019

Center of Toxins, Immune-response and Cell Signaling (CeTICS)

Laboratório de Ciclo Celular, Instituto Butantan

São Paulo, 22 de Janeiro de 2020

Conteúdo

1 Resumo do projeto proposto

A construção de modelos funcionais é uma técnica comum para se estudar vias de sinalização celular e, quando a via estudada é pouco conhecida, é possível que os modelos já propostos sejam incompletos, tornando necessário a sua modificação. Lulu Wu apresentou em 2015, em sua dissertação de mestrado, um método para modificar sistematicamente modelos funcionais, adicionando a estes interações extraídas de repositórios como KEGG. Entretanto, esta metodologia apresentou limitações: a primeira é a incompletude do banco de dados de interações criado, que extraia informações apenas do repositório KEGG; a segunda, a falta de informações sobre constantes de velocidade de interações, que podem ser extraídas de repositórios como BioNumbers; a terceira, a dinâmica do algoritmo de busca, incremental, que pode não achar o mínimo global; e a última, a penalização na complexidade dos modelos, que era feita de maneira aleatória. Propomos neste trabalho enfrentar as limitações encontradas pela metodologia de Lulu, criando um banco de dados de interações mais completo e também novas funções de custo que sejam capazes de penalizar modelos mais complexos (como critério de informação Akaike e *Bayesian inference-based modeling*); esta penalização deve induzir, em cadeias do espaço de busca, curvas em u no custo dos modelos, portanto também propomos a criação de novos algoritmos de busca que explorem essa característica da função de custo. Por fim, esperamos testar nossa metodologia na identificação de vias de sinalização celular da linhagem tumoral murina Y1.

2 Atividades desenvolvidas

2.1 Disciplinas cursadas

No período da bolsa coberto pelo primeiro Relatório Científico, o beneficiário já havia cursado três disciplinas: Tópicos em Análise de Algoritmos, Probabilidade e Inferência Estatística I, e Laboratório de Programação Extrema. Após esse período, entre os meses de

agosto e dezembro de 2019, o beneficiário também cursou a disciplina Modelagem de Banco de Dados. As notas das disciplinas cursadas estão disponíveis no histórico escolar que segue em anexo a este relatório.

2.2 Participação em cursos e conferências

No mês de janeiro de 2019, entre os dias 7 e 11, o beneficiário participou da XIV *Escuela de Verano en Matemáticas Discretas*, em Valparaíso, Chile. Nesta escola três cursos foram oferecidos, envolvendo otimização combinatória, análise de algoritmos parametrizados e problemas computacionais no estudo do cérebro. A participação do beneficiário foi financiada com reserva técnica da FAPESP e também com recursos da própria organização do evento.

Além disso, o beneficiário teve um resumo aceito para participação na ROCKY 2019, conferência que foi realizada entre os dias 5 e 7 de dezembro, na cidade Snowmass Village, Colorado, Estados Unidos. Nessa conferência, o beneficiário apresentou o trabalho desenvolvido neste projeto (o pôster apresentado segue em anexo). A participação do beneficiário nessa conferência foi financiada com reserva técnica da FAPESP e também com recursos da Pós-graduação em Computação do IME-USP.

2.3 Resumo de atividades anteriores

2.3.1 Estudo de seleção de modelos de via de sinalização celular

O desenvolvimento deste projeto se iniciou com o estudo de métodos capazes de avaliar a qualidade de um modelo de via de sinalização celular. Dado um conjunto de experimentos $\mathbf{D}$, que medem concentrações de espécies químicas, precisamos escolher uma função de custo $c(\mathbf{D}, M)$ que possa indicar a capacidade de um modelo M em reproduzir corretamente dados observados $\mathbf{D}$. O modelos de via que utilizamos é definido por um conjunto de reações químicas, produzindo um sistema de equações diferenciais, capaz de simular a dinâmica das concentrações de espécies químicas da via ao longo do tempo. Este sistema de equações

diferenciais é criado utilizando leis de cinética química, como no modelo de Michaelis-Menten, e possuem constantes de velocidade que são usualmente desconhecidas; estas constantes são parâmetros dos modelos de vias.

A função de custo escolhida deve considerar possíveis valores para as constantes de velocidade do modelo avaliado. A abordagem de Lulu Wu [1], por exemplo, utiliza um processo de *simulated annealing* para encontrar o melhor conjunto de valores de parâmetros para um modelo e conjunto de experimentos. Entretanto, esta abordagem teve limitações que podem estar associadas a falta de informação a priori sobre as constantes e também a falta de penalização apropriada a modelos mais complexos. Por conta destas limitações, decidimos implementar uma função de custo baseada em estatística Bayesiana, chamada de verossimilhança marginal; denotamos $p(\mathbf{D}|M)$ a verossimilhança marginal de um conjunto de dados $\mathbf{D}$ dado um modelo M. Esta abordagem, apresentada no mesmo contexto no trabalho de Vyshemirsky e Girolami [2], permite a definição de informações a priori sobre constantes de velocidades e também induzem a penalização automática de modelos mais complexos.

Para calcular a verossimilhança marginal, precisamos definir a função de verossimilhança, $p(\mathbf{D}|M,\theta)$, onde $\mathbf{D}$ é o conjunto de experimentos, M é o modelo de interesse, e θ é um conjunto de valores para os parâmetros (constantes de velocidade) do modelo. Seguindo a abordagem de Kolch e Girolami [3], assumimos erro Gaussiano e independente:

$$p(\mathbf{D}|M,\theta) = \prod_{i=1}^{m} p_{\mathcal{N}_{(0,\sigma^2)}}([\phi(M,\theta) - \mathbf{D}]_i), \tag{1}$$

onde $\phi(M,\theta)$ é um vetor com os valores simulados de concentrações, em cada intervalo de tempo, pelo modelo M usando parâmetros θ. A partir da função de verossimilhança, podemos obter a verossimilhança marginal com uma marginalização sobre os valores de parâmetros de modelos, ou seja, integrando a função de verossimilhança sobre o espaço paramétrico, Θ.

Desta forma podemos escrever:

$$p(\mathbf{D}|M) = \int_{\Theta} p(\mathbf{D}|M,\theta)p(\theta|M)d\theta. \tag{2}$$

Entretanto, a integral 2 normalmente não pode ser calculada analiticamente; para isso, seria necessário determinar a distribuição de probabilidade conjunta $p(D,\theta|M)$, o que não é possível usualmente. Portanto, como é muito difícil (ou mesmo impossível) calcular a verossimilhança marginal, utilizamos um estimador desse valor como função de custo. Este estimador é construído utilizando um método conhecido como Integral Termodinâmica [4].

2.3.2 Estimação de verossimilhança marginal

O trabalho de Friel et al. [4] mostra que é possível reescrever o logaritmo da integral 2 como uma outra integral, um pouco menos simples, mas que nos permite criar estimadores para o logaritmo da verossimilhança marginal. Esta segunda forma de se escrever a verossimilhança marginal é baseada na integração de várias distribuições de probabilidade que são intermediárias entre as distribuições a priori e a posteriori das constantes de velocidade. As distribuições intermediárias são denominadas potências de posteriori.

Dada uma distribuição a priori $p(\theta|M)$ e a posteriori $p(\theta|\mathbf{D},M)$, definimos a distribuição potência de posteriori como:

$$p_{\beta}(\theta) = \frac{p(\mathbf{D}|\theta,M)^{\beta}p(\theta|M)}{z(\beta)},$$

onde

$$z(\beta) = \int_{\Theta} p(\mathbf{D}|\theta,M)^{\beta}p(\theta|M)d\theta.$$

Note que $p_0(\theta)$ é a distribuição a priori e que $p_1(\theta)$ é a distribuição a posteriori. Portanto, podemos dizer que quando variamos o valor de β entre 0 e 1 estamos produzindo distribuição

intermediárias que conectam a priori a posteriori. Friel et al. provam que é possível escrever:

$$\begin{aligned}\int_0^1 \mathbb{E}_{p_\beta(\theta)}[\ln p(\mathbf{D}|\theta, M)]d\beta &= \int_0^1 \frac{d}{d\beta} \ln z(\beta) d\beta \\ &= \Big[\ln z(\beta) \Big] \Big|_0^1 \\ &= \ln p(\mathbf{D}|M). \end{aligned} \tag{3}$$

O lado esquerdo da equação 3 recebe o nome de integral termodinâmica. Este nome se justifica pela variação do parâmetro β, que pode ser visto como um parâmetro de temperatura nas distribuições potência de posteriori. Esta integral pode ser estimada ou aproximada numericamente, permitindo acessar um valor próximo ao logaritmo da verossimilhança marginal. Para estimar ou aproximar esta integral, é necessário construir amostras de distribuições potência de posteriori para um conjunto finito de valores de β.

Em resumo, reescrevemos o logaritmo da verossimilhança marginal como uma integral que chamamos de integral termodinâmica. Esta integral pode ser aproximada numericamente ou estimada. Para ambas opções, é necessário escolher uma sequência de valores para β entre 0 e 1, e gerar amostras das distribuições potência de posteriori para os respectivos valores de β escolhidos.

2.3.3 Implementação do pacote SigNetMS

Após nossos estudos sobre seleção de modelos, decidimos implementar um pacote Python que nos providenciaria uma aproximação do logaritmo da verossimilhança marginal, usando os conceitos de integral termodinâmica. Este pacote foi implementado e recebeu o nome SigNetMS, e está disponível em um repositório público no GitHub [1]. O pacote SigNetMS recebe como entrada um arquivo no formato *Systems Biology Makup Language* (SBML) [5], com a definição das reações e constantes de velocidade da via; um arquivo *Extensible Markup Language* (XML) com resultados de experimentos; e um arquivo XML com definições de

[1] https://github.com/gustavoem/SigNetMS

distribuições a priori para constantes de velocidade das reações da via. O pacote pode devolver como resposta o valor aproximado de $\log p(\mathbf{D}|M)$ e também amostras das distribuições potência de posteriori.

O pacote SigNetMS é capaz de processar modelos no formato SBML e criar os correspondentes sistemas de equações diferenciais ordinárias. Utilizando o pacote `Scipy` e seu integrador `odeint` é possível integrar esses sistema de equações diferenciais, criando uma simulação da dinâmica das concentrações gerada pelo par modelo e parâmetros (M, θ). Esta simulação é utilizada na função de verossimilhança, implementada de acordo com a equação 1. A verossimilhança do experimento, dado um modelo e conjunto de parâmetros, é usada no processo de geração da amostra de distribuições potência de posteriori e também na aproximação da verossimilhança marginal.

Para calcular a verossimilhança marginal, o pacote SigNetMS segue uma abordagem que faz uma aproximação numérica da integral 3. Esta aproximação é simplesmente a aplicação da regra dos trapézios para integrais. Desta maneira, é necessário escolher uma sequência de valores para β, o que também determina o conjunto de potências de posteriori que serão amostradas. O pacote SigNetMS faz esta escolha de $\beta_1, \ldots, \beta_T$ da maneira recomendada por Friel et al.:

$$\beta_t = \left(\frac{t-1}{T-1}\right)^c,$$

com $T = 20$ e $c = 5$. Assim, aplicando a regra dos trapézios na integral 3, podemos escrever:

$$\log p(\mathbf{D}|M) \approx \sum_{t=0}^{T-1} (\beta_{t+1} - \beta_t) \frac{\mathbb{E}_{p_{\beta_{t+1}}(\theta)}[\log p(D|M,\theta)] + \mathbb{E}_{p_{\beta_t}(\theta)}[\log p(D|M,\theta)]}{2}.$$

Além disso, se considerarmos que a potência de posteriori β_t tem M_t parâmetros amostrados, então podemos substituir a esperança por um estimador de seu valor, produzindo a equação:

$$\log p(D|M) \approx \sum_{t=0}^{T-1} (\beta_{t+1} - \beta_t) \frac{\frac{1}{M_{t+1}} \sum_{i=1}^{M_{t+1}} \log p(D|M,\theta^{(t+1,i)}) + \frac{1}{M_t} \sum_{i=1}^{M_t} \log p(D|M,\theta^{(t,i)})}{2}, \tag{4}$$

onde $\theta^{(j,i)}$ é o i-ésimo parâmetro amostrado para a potência de posteriori $p_{\beta_j}(\theta)$. Resta agora definir como as amostras de potência de posteriori são criadas.

As amostras de potência de posteriori são criadas em três etapas que utilizam o algoritmo Metropolis-Hastings. Esse algoritmo permite gerar uma amostra de uma distribuição (geralmente desconhecida ou difícil de se amostrar) a partir de uma distribuição de proposta, com a criação de uma cadeia de Markov. Chamamos estas três etapas de burn-in, burn-in informativo e amostragem final; todas estas etapas utilizam a distribuição log-normal como distribuição de proposta para os parâmetros.

Na versão do SigNetMS que utilizamos até a escrita do relatório parcial, a etapa de burn-in amostrava a distribuição a posteriori (ou seja, apenas uma cadeia) de parâmetros de maneira independente, com uma distribuição de pulo com covariância diagonal. Na etapa de burn-in informativo, uma amostragem similar a primeira etapa ocorria, porém utilizando uma distribuição de pulo com matriz de covariância diagonal tal que cada variância fosse igual a variância da amostra atual. Por fim, na ultima etapa, T cadeias eram geradas, uma para cada potência de posteriori escolhida, com distribuição de pulo igual a última utilizada na etapa anterior.

2.3.4 Primeiros testes da metodologia

Ainda no primeiro ano do projeto, testamos o SigNetMS na seleção de modelos. Porém, os resultados eram satisfatórios apenas para exemplos pequenos. Com exemplos maiores e com mais parâmetros, o pacote não apresentava bons resultados. Por esse motivo, começamos o segundo período do projeto (entre dezembro de 2018 e dezembro de 2019) ajustando nossa implementação

2.4 Melhorando a estimação da posteriori

Logo após a entrega do relatório parcial, identificamos que as amostras de potência de posteriori geradas pelo SigNetMS eram muito parecidas. Isso indicava que existia uma cor-

relação grande entre as cadeias amostradas. Revisitando os trabalhos de Xu et al. [3] e de Friel et al. [4], identificamos que as duas primeiras etapas de amostragem, burn-in e burn-in informativo também deveriam ser feitas para cada potência de posteriori escolhida, e não apenas para a distribuição a posteriori. Além disso, identificamos que a distribuição de pulo da etapa de burn-in informativo poderia ter como covariância a covariância amostral do conjunto de parâmetros aceitos até o instante.

2.5 Alternativa de avaliação de modelos

Ao mesmo tempo que investigamos possíveis erros na metodologia do SigNetMS também experimentamos uma outra função de custo Bayesiana para seleção de modelos de via, chamada ABC-SMC [6]. A função ABC-SMC se baseia em um método de geração de amostras conhecido como *Approximate Bayesian Computation* (ABC). Na função de custo ABC-SMC, amostras da distribuição $p(\theta, M|\mathbf{D})$ são geradas, permitindo estimar o valor de $p(M|\mathbf{D})$.

Podemos escrever um algoritmo genérico ABC que se propõe a gerar amostras da distribuição $p(\theta, M|\mathbf{D})$ com os seguintes passos:

1. Amostre um parâmetro candidato $(\theta^*|M^*)$ da distribuição a priori $p(\theta, M)$.

2. Simule o par (θ^*, M^*) com os mesmos intervalos de tempo e para a mesma métrica do experimento $\mathbf{D}$, gerando $\phi(\theta, M) = \mathbf{D}^*$.

3. Calcule, para alguma métrica de distância d, se o valor $d(\mathbf{D}, \mathbf{D}^*)$ for menor que um ϵ pré-determinado, então adicione o par (θ^*, M^*) a amostra.

4. Repita até uma condição de parada.

O resultado deste algoritmo é uma amostra da distribuição $p(\theta, M|d(\phi(\theta, M), \mathbf{D}) \leq \epsilon)$. De acordo com Pritchard et al., quando $\epsilon \to \infty$, então o resultado será uma amostra da distribuição a priori, $p(\theta, M)$, e quando $\epsilon \to 0$, então o resultado será uma amostra da distribuição a posteriori [7], $p(\theta, M|\mathbf{D})$. Entretanto, escolher um ϵ pequeno pode ser problemático quando

a priori e a posteriori tem distribuições muito diferentes, pois neste caso os candidatos gerados são pouco prováveis a posteriori.

Para solucionar este problema, Toni et al. proporam o algoritmo ABC Sequential Monte Carlo (ABC-SMC) [8]. Este algoritmo cria uma sequência de amostras que podem ser vistas como amostras de distribuições intermediárias entre a priori e a posteriori. Dado uma sequência $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T$, este algoritmo gera amostras das distribuições:

$$p(\theta, M|d(\phi(\theta, M), \mathbf{D}) \leq \epsilon_1),\ p(\theta, M|d(\phi(\theta, M), \mathbf{D}) \leq \epsilon_2),\ \ldots,\ p(\theta, M|d(\phi(\theta, M), \mathbf{D}) \leq \epsilon_T).$$

A primeira amostra, que tem limiar ϵ_1, é gerada usando a distribuição a priori, enquanto as próximas amostras, de limiares $\epsilon_2, \ldots, \epsilon_T$, são geradas com perturbações aleatórias as amostras de limiares anteriores. O pseudocódigo 1 mostra como o ABC-SMC funciona.

Note que o desempenho e a qualidade da solução encontrada pelo algoritmo ABC-SMC dependem da sequência de limiares $\epsilon_1, \ldots, \epsilon_T$. Quanto maior o valor de T, maior é a quantidade de amostras geradas, e além disso, quanto menor é o valor do limiar, maior é a quantidade média de propostas necessárias até uma proposta satisfazer o limiar. A diferença entre dois limiares seguidos também não pode ser grande, para evitar o mesmo problema do algoritmo ABC, que falha quando a distribuição de proposta é muito diferente da distribuição de interesse. Por outro lado, o valor de ϵ_1 deve ser alto, pois a primeira amosta deve ser parecida com a prioro, enquanto o valor de ϵ_T deve ser pequeno, para que a amostra se pareça com a posteriori. Portanto, a sequência de limiares deve ter grande extensão, com valor de T pequeno, e com ϵ_T também pequeno.

2.6 Testes de seleção de modelos

Após as mudanças implementadas ao SigNetMS e o estudo da função ABC-SMC, decidimos comparar as duas abordagens em experimentos de seleção de modelos. Para testar a função ABC-SMC, usamos o programa ABC-SysBio, que implementa em Python o ABC-

ABC SMC $(\mathcal{M}, D)$

Defina a sequência $\epsilon_1, \ldots, \epsilon_T$.
Defina N, o tamanho das amostras.
Amostre $\{(\theta^{(1,1)}, M^{(1,1)}), (\theta^{(1,2)}, M^{(1,2)}), \ldots, (\theta^{(1,N)}, M^{(1,N)})\}$ de $p(\theta, M|\mathbf{D})$.
Seja $w^{(1,i)} = 1, \forall i \in 1, \ldots, N$.
for $t \in \{1, \ldots, T\}$ **do**
    $i \leftarrow 1$
    while $i \leq N$ **do**
        Amostre $M^* \propto p(M|\mathbf{D})$.
        Amostre $(\theta^{(t-1,k)}, M^*)$ da amostra de limiar ϵ_{t-1}, com peso $w^{(t-1,k)}$.
        Produza (θ^*, M^*) ao perturbar $\theta^{(t-1,k)}$; $\theta^* \propto K^t(\theta|\theta^{(t-1,k)})$.
        if $p(\theta^*|M^*) = 0$ **then**
            Continue para próxima iteração.
        end if
        $\mathbf{D}^* \leftarrow \phi(M^*, \theta^*)$
        if $d(\mathbf{D}^*, \mathbf{D}) \leq \epsilon_t$ **then**
            $i \leftarrow i + 1$
            $(\theta^{(t,i)}, M^{(t,i)}) \leftarrow (\theta^*, M^*)$
        end if
    end while
    Calcule os pesos da amostra atual: $w^{(t,i)} = \frac{p(\theta^{(t,i)}|M^{(t,i)})}{\sum_{j=1}^{N} w^{(t-1,j)} p_{K^t}(\theta^{(t-1,j)}, \theta^{(t,i)})}$
end for
return

Algoritmo 1: Pseudocódigo do procedimento ABC SMC.

SMC para seleção de modelos.

Para testar as duas abordagens, realizamos dois experimentos similares, em que quatro modelos são calibrados e avaliados de acordo com dados gerados por um destes modelos, chamado de modelo correto. Explicaremos a seguir esses experimentos.

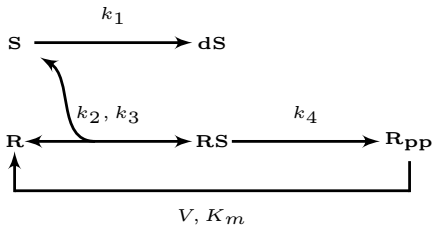
2.6.1 Primeiro experimento

No primeiro experimento que realizamos analisamos quatro modelos candidatos, de maneira similar a Vyshemirsky et al. [2]. Na figura 1 são mostrados os quatro modelos candidatos: um modelo correto; um modelo que é uma simplificação do correto; um modelo que é incompleto; e, por fim, um modelo que é uma generalização do modelo correto.

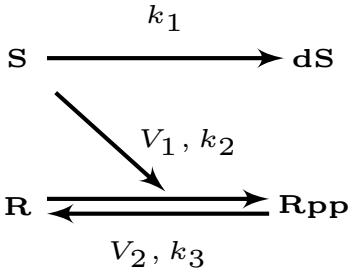
Os dados experimentais foram gerados com simulações do modelo correto, medindo as concentrações de R_{pp} nos intervalos de tempo: $2s$, $5s$, $10s$, $20s$, $40s$, $60s$, e $100s$, seguindo da adição de erro Gaussiano com média zero e variância de $0,01$, três vezes seguidas, para se gerar três conjuntos de medições. Os valores utilizados para constantes de velocidade foram: $k_1 = 0,07$, $k_2 = 0,6$, $k_3 = 0,05$, $k_4 = 0,3$, $V = 0,017$, e $K_m = 0,3$. As concentrações iniciais usadas foram: $S = 1$, $R = 1$, $dS = 0$, $RS = 0$, $R_{pp} = 0$. Por motivos de simplificação, não indicamos as unidades de medida destas constantes. Por fim, descartamos os valores dos parâmetros dos quatro modelos e buscamos inferí-los utilizando os dados experimentais produzidos e o ABC-SysBio ou o SigNetMS.

No ABC-SysBio, utilizamos a sequência de limiares gerada automaticamente para o problema e tivemos como resultado a ordenação: modelo incompleto, modelo simplificado, modelo generalizado e modelo correto. Na figura 2 mostramos simulações geradas por amostras de parâmetros de diferentes iterações no programa ABC-SysBio, considerando o modelo correto. É possível ver que ao longo das iterações os parâmetros fazem a dinâmica simulada se aproximar dos dados experimentais, porém apenas parcialmente. Isto significa que a amostra gerada pelo programa ABC-SysBio com uma sequência de limiares gerada automaticamente não se aproxima da amostra da distribuição a posteriori dos parâmetros, comprometendo a qualidade da função de avaliação de modelos.

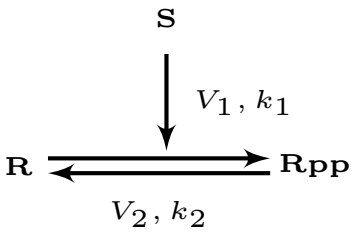
Já nos experimentos com o SigNetMS, utilizamos 15 mil iterações de burn-in e 5 mil iterações para burn-in informado e para amostragem final. Como resultado, os modelos foram ordenados da seguinte maneira: modelo correto, modelo simplificado, modelo generalizado e modelo incompleto. Este resultado é similar ao resultado original de Vyshemirsky e



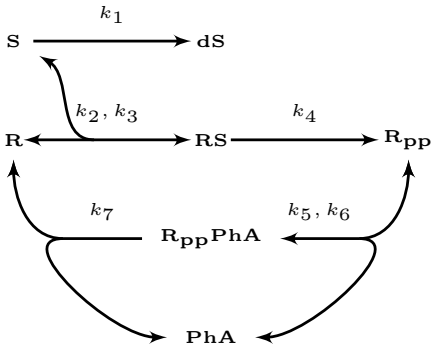
(a) Modelo correto.



(b) Modelo simplificado.



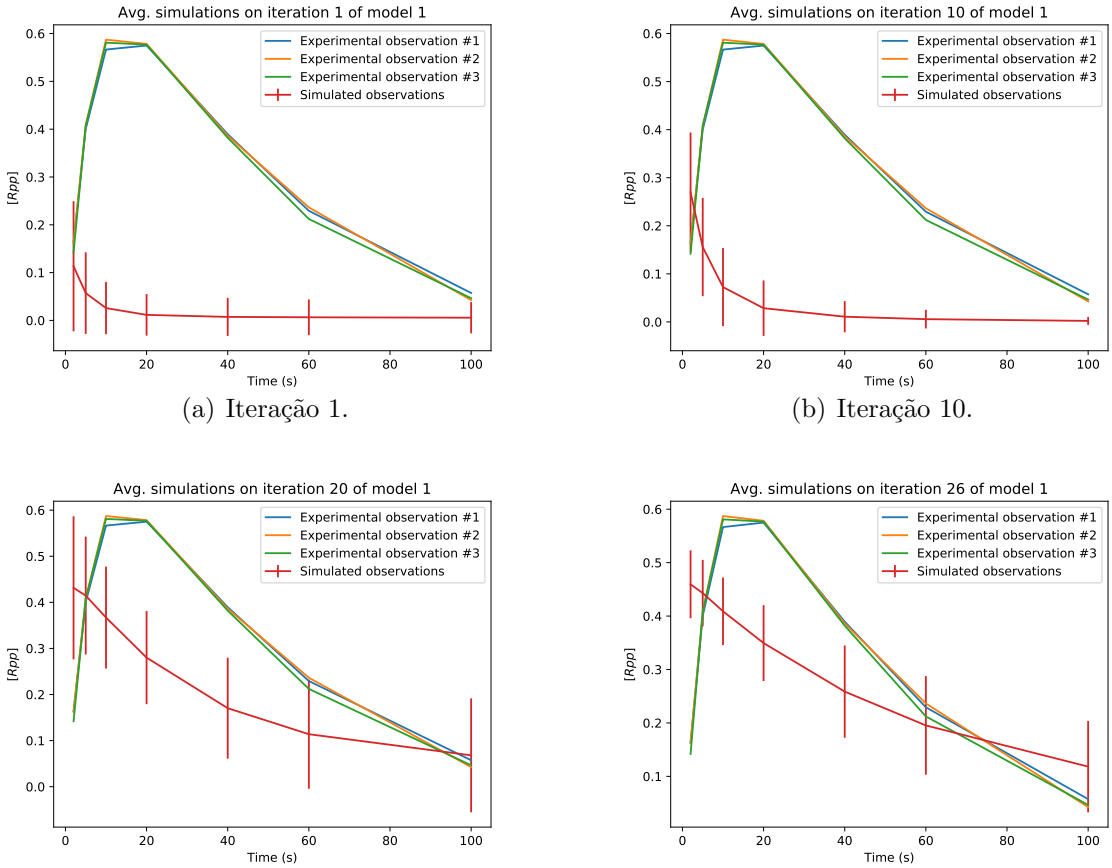
(c) Modelo incompleto.



(d) Modelo generalizado.

Figura 1: **Os quatro modelos candidatos do primeiro experimento.** *A medida de interesse na dinâmica destes modelos é a concentração da espécie química R_{pp}.*

Girolami (2007); quando comparada a ordenação que obtemos com a que é reportada nesse paper, vemos que apenas os modelos simplificado e generalizado trocaram de lugar entre eles. Na figura 3 podemos ver simulações geradas por parâmetros amostrados em diferentes potências de posteriori no programa SigNetMS, considerando o modelo correto. Vemos que com o aumento do valor de β, as amostras geradas se aproximam da distribuição a posteriori, pois a curva simulada se aproxima da curva gerada no experimento. Já na figura 4, mostramos as simulações geradas pelos parâmetros amostrados no SigNetMS da distribuição a posteriori; podemos observar que apenas o modelo incompleto, classificado como o pior, não se aproximou da dinâmica medida no experimento.



(a) Iteração 1.

(b) Iteração 10.

(c) Iteração 20.

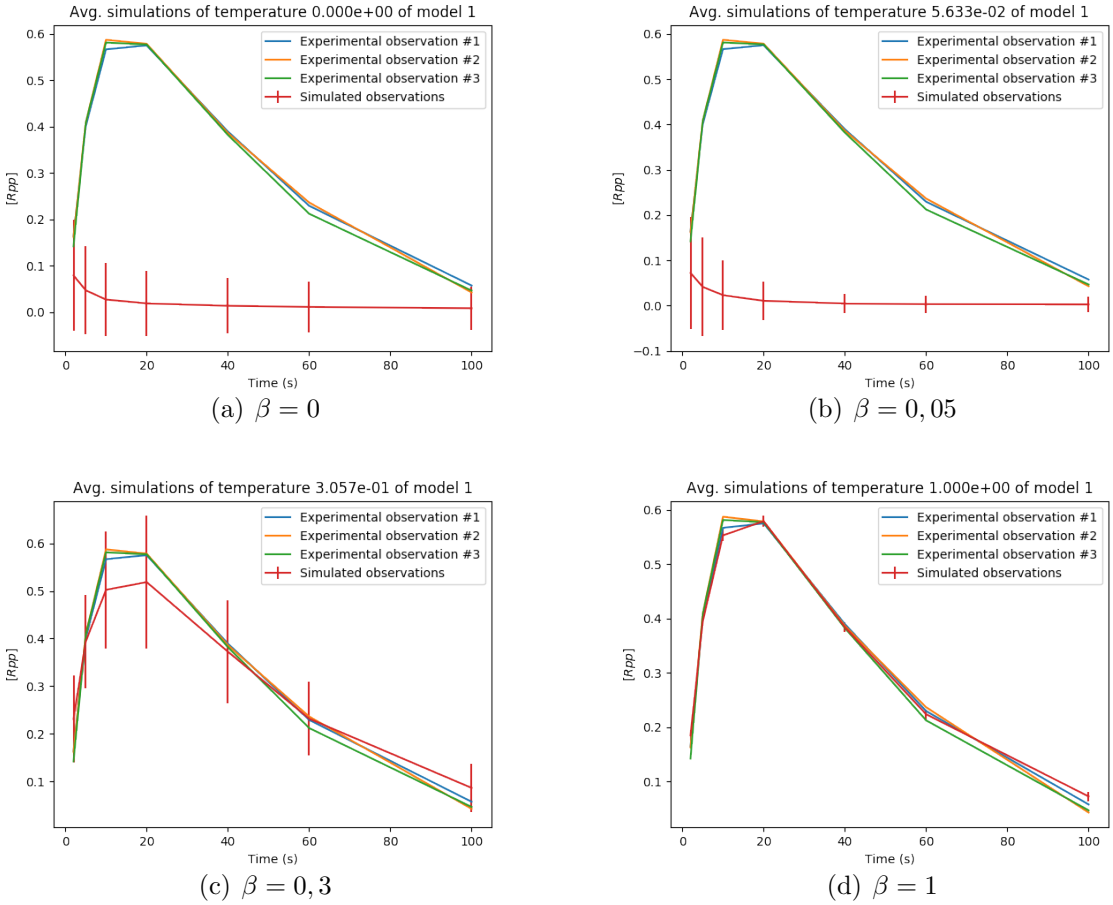
(d) Iteração 26.

Figura 2: **Estimação de parâmetros do modelo correto com o ABC-SysBio.** *Simulação média gerada pelos parâmetros amostrados na iterações 1, 10, 20 e 26 do método ABC SMC.*

2.6.2 Segundo experimento

O segundo experimento, similarmente ao primeiro, consiste em usar as duas propostas de avaliação de modelos para criar uma ordenação de quatro modelos em que um deles é o modelo correto, usado para geração de dados experimentais. Os quatro modelos são apresentados da figura 5, e são compostos, além do modelo correto, por um modelo simplificado, por um modelo generalizado e um modelo incorreto.

Usando o programa ABC-SysBio obtivemos a seguinte ordenação de modelos: modelo simplificado, modelo correto, modelo incorreto e modelo generalizado. Por outro lado, quando utilizamos o programa SigNetMS obtivemos a ordenação: modelo simplificado, modelo correto, modelo generalizado e modelo incorreto.

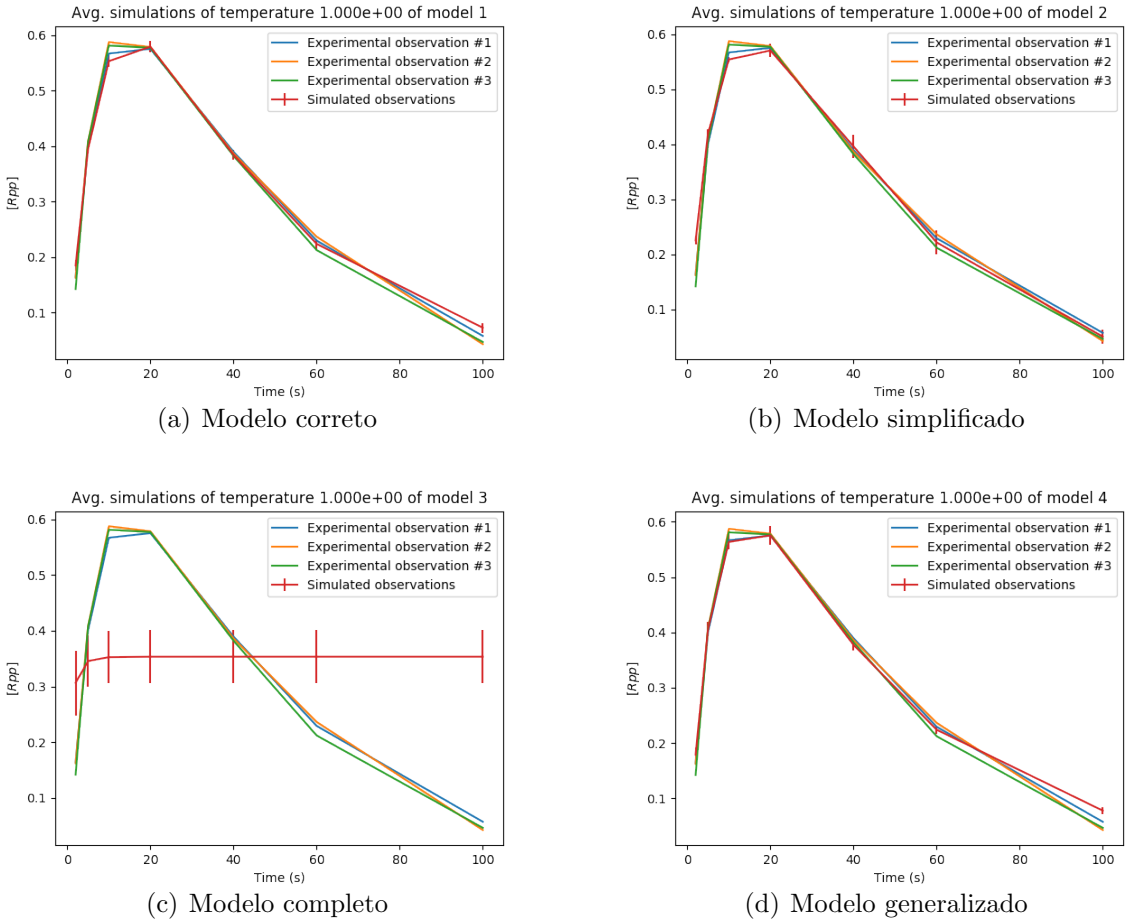


(a) $\beta = 0$ (b) $\beta = 0,05$

(c) $\beta = 0,3$ (d) $\beta = 1$

Figura 3: **Estimação de parâmetros do modelo correto com o SigNetMS.** *Simulação média gerada pelos parâmetros amostrados para potências de posteriori com* β *valendo* 0, $0,5$, $0,3$ *e* 1.

Gerando gráficos similares ao que apresentamos no último experimento, que mostram as simulações geradas por parâmetros amostrados, conseguimos analisar a ordenação gerada por ambos programas. Para os modelos correto e simplificado, ambos programas amostraram parâmetros que aproximavam a curva simulada da curva do experimento. Para o modelo incorreto nenhum dos dois programas amostrou parâmetros que aproximassem a dinâmica do experimento. Por fim, para o modelo generalizado, apenas o programa SigNetMS encontrou parâmetros que faziam a simulação aproximar os dados do experimento.



(a) Modelo correto

(b) Modelo simplificado

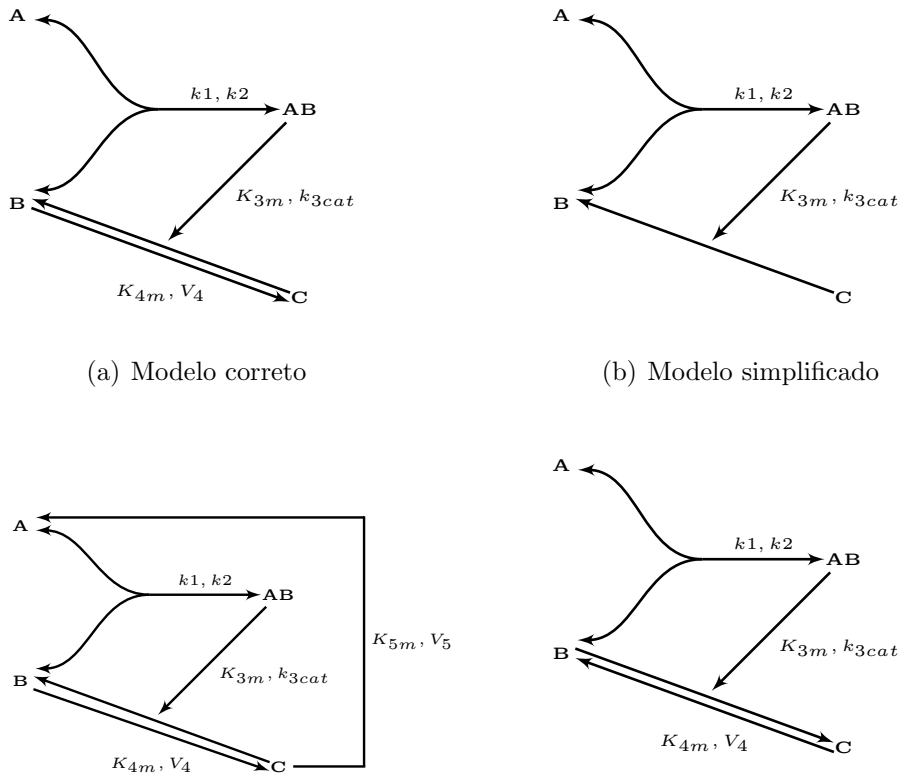
(c) Modelo completo

(d) Modelo generalizado

Figura 4: **Estimação de parâmetros dos quatro modelos com o SigNetMS.** *Simulação média gerada pela amostra da posteriori criada para cada um dos quatro modelos candidatos.*

2.6.3 Escolha de método de avaliação de modelos

Por conta dos resultados obtidos nos dois experimentos anteriores, decidimos focar, no restante deste projeto, no uso do método implementado no SigNeMS. É importante também ressaltar que a ferramenta ABC-SysBio pode ser mais adequada em aplicações em que uma função de verossimilhança não pode ser escrita; por outro lado, também seria interessante investigar, em trabalhos futuros, o desempenho desse método com o uso de outras sequências de limiares que não a automática, uma vez que o ABC-SysBio é altamente escalável, dispondo de suporte a GPUs para paralelização. Por outro lado, a paralelização do método implementado no SigNetMS é mais complicada e será abordada a seguir.



(a) Modelo correto

(b) Modelo simplificado

(c) Modelo generalizado

(d) Modelo incorreto

Figura 5: **Os quatro modelos candidatos no segundo experimento.** *A medida de interesse nestas vias é a concentração da espécie química C.*

2.7 Paralelização do SigNetMS

Apesar de se mostrar adequada para nossas aplicações, a versão original da ferramenta SigNetMS possuía como grande desvantagem o seu tempo de execução. Por conta disto, analisamos a ferramenta com objetivo de identificar porções de código que poderiam ser executadas em paralelo. Identificamos então que o procedimento de geração de amostras de potências de posteriori era responsável por grande parte do tempo de execução.

O procedimento de geração de amostras foi implementado com o algoritmo de Metropolis-Hastings, que é de difícil paralelização, pois um passo depende do passo anterior. Entretanto, as amostras de diferentes potências de posteriori são geradas de maneira independente nas etapas de burn-in e burn-in informado. Logo, paralelizamos estas duas etapas da amostragem.

Para implementar a paralelização utilizamos a biblioteca *pathos*[2], e a função *map*, que nos permitiu fazer com que cada potência de posteriori fosse amostrada, nas etapas de burn-in e burn-in informado, em uma thread diferente.

A última etapa do processo de amostragem não foi paralelizada, pois nela é realizado o processo de Monte Carlo Markov Chain Populacional [4], que mistura amostras de diferentes potências de posteriori, dificultando a paralelização.

2.8 Implementação eficiente de integração de sistemas de equações diferenciais

Mesmo com a paralelização do processo de amostragem, o tempo de execução do programa SigNetMS ainda era muito grande, mesmo para exemplos pequenos. Decidimos então investigar o processo de integração de sistema de equações diferenciais.

Até este momento, estávamos utilizando uma representação literal do sistema de equações diferenciais. Portanto, para fazer sua integração, usávamos uma função que interpretava as derivadas em formato texto e calculava seus respectivos valores. Propomos então utilizar uma notação simbólica do sistema de equações diferenciais, utilizando a biblioteca *SymPy*.

Ao utilizar a notação simbólica no SymPy, fomos capazes de usar funções da própria biblioteca que permitem transformar o sistema de equações diferenciais em uma função escrita em C, que é automaticamente compilada e transformada em uma função em Python. Além disso, a notação simbólica também nos permite facilmente calcular a matriz Jacobiana do sistema de equações diferenciais, o que melhora o desempenho do integrador numérico. A tabela 1 mostra o tempo de execução do integrador numérico quando usamos as duas diferentes abordagens de representação do sistema; é possível ver nela que, com a notação simbólica, o tempo de execução diminuiu consideravelmente.

[2]https://github.com/uqfoundation/pathos

	Tempo de execução em segundos	
# de integrações do sistema	Representação literal	Representação simbólica
100	31,1	5,4
200	63,3	10,5
400	125,6	20,3

Tabela 1: **Tempo de execução do integrador numérico para duas abordagens de representação do sistema de equações diferenciais.** *Para gerar estes números, o modelo correto do primeiro experimento (figura 1) foi integrado numericamente repetidas vezes (100, 200 e 400 repetições).*

2.9 Testes da metodologia em uma cadeia do espaço de busca

Com a melhora de desempenho do pacote SigNetMS, seguimos o trabalho com um experimento maior de seleção de via de sinalização celular. Este experimento consistiu em adicionar, uma a uma e de maneira aleatória, reações candidatas a um modelo inicial e analisar o comportamento da função de custo sobre sequência de modelos criada. O modelo em questão descreve a dinâmica da via de sinalização ERK, que foi ajustado com experimentos de células tumorais murinas Y1 e reportados anteriormente [9]. Do ponto de vista de seleção de características (em que reações candidatas são características), este experimento é equivalente a um passeio aleatório no espaço de busca (todos os subconjuntos de características) que sempre adiciona características, começando no conjunto vazio, correspondente ao modelo base, e terminando no conjunto completo, correspondente ao modelo base mais todas as reações candidatas.

Os dados experimentais para avaliação dos modelos foram gerados por um modelo intermediário entre o modelo base (i.e., um modelo com o mínimo de reações possíveis para descrever a cinética da via) e o modelo com o conjunto completo de reações. Na figura 6 mostramos o resultado deste experimento; nela podemos ver que a função de custo de fato penaliza modelos mais complexos, e que a função de custo não é monotônica em uma cadeia do espaço de busca, apresentando dois mínimos locais.

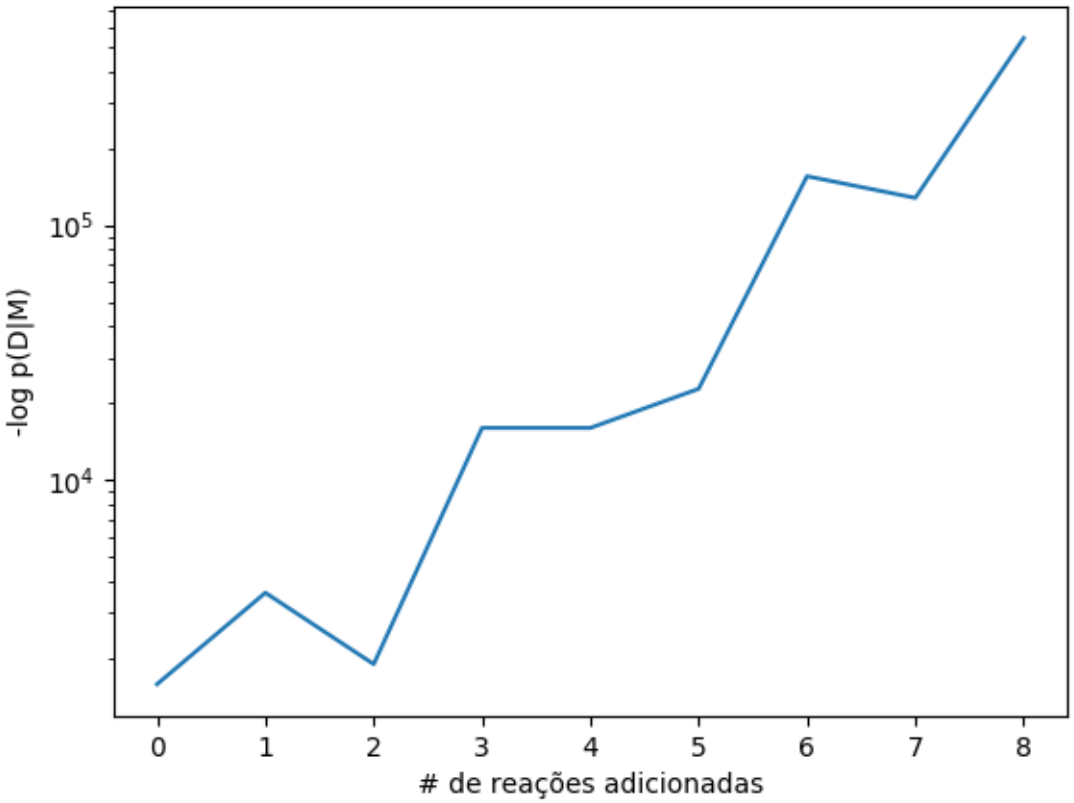


Figura 6: **Comportamento da função de custo como uma função da inclusão de reações a um modelo base.** *Para facilitar a visualização, mostramos no gráfico a verossimilhança marginal vezes a constante* -1 *- ou seja, quanto menor o valor na curva, melhor a qualidade do modelo.*

3 Atividades remanescentes

Apesar de todos os avanços que tivemos neste projeto após a entrega do relatório parcial e que foram aqui descritos, existem algumas atividades que precisam ser concluídas para o depósito da dissertação de mestrado e posterior defesa; tais atividades são listadas abaixo, com um prazo estimado para a conclusão de cada uma delas.

Fevereiro de 2020. Conclusão do experimento de verificação das cadeias do espaço de busca. No experimento completo, pretendemos repetir várias vezes o percorrimento aleatório de cadeias, tal qual mostrado na figura 6. Além disso, faremos a comparação do comportamento dessas cadeias aleatórias com o de percorrimentos que passam pela

cadeia que contém o modelo correto (i.e., o mínimo global do procedimento de busca). Se as cadeias deste último caso descreverem curvas em U, então poderíamos aplicar métodos desenvolvidos por nosso grupo para resolver o problema [10, 11].

Março de 2020. Construção de um pequeno banco de dados (essencialmente lista de reações e de constantes de velocidade) para definir o espaço de busca da seleção de modelos. Por questões práticas, vamos nos restringir a reações que envolvam ao menos uma das espécies químicas presentes no modelo da via de sinalização ERK apresentado na seção 2.9.

Abril de 2020. Experimento de seleção de modelos utilizando o banco de dados mencionado anteriormente. Finalização da dissertação de mestrado.

Maio de 2020. Depósito da dissertação de mestrado.

Junho de 2020. Defesa do mestrado.

Após a defesa do mestrado, projetamos realizar, ao longo do segundo semestre deste ano, experimentos adicionais. Além disso, pretendemos também escrever um manuscrito para ser enviado a uma conferência internacional ainda a ser definida.

Referências

[1] Lulu Wu. Um método para modificar vias de sinalização molecular por meio de análise de banco de dados de interatomas. Master's thesis, Universidade de São Paulo, 2015.

[2] Vladislav Vyshemirsky and Mark A. Girolami. Bayesian ranking of biochemical system models. *Bioinformatics*, 24(20):2421, 2008.

[3] Tian-Rui Xu, Vladislav Vyshemirsky, Amélie Gormand, Alex von Kriegsheim, Mark Girolami, George S. Baillie, Dominic Ketley, Allan J. Dunlop, Graeme Milligan, Miles D.

Houslay, and Walter Kolch. Inferring signaling pathway topologies from multiple perturbation measurements of specific biochemical species. *Science Signaling*, 3(113):ra20–ra20, 2010.

[4] N. Friel and A. N. Pettitt. Marginal likelihood estimation via power posteriors. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 70(3):589–607, 2008.

[5] Michael Hucka, Andrew Finney, Herbert M. Sauro, Hamid Bolouri, John C. Doyle, Hiroaki Kitano, Adam P. Arkin, Benjamin J. Bornstein, Dennis Bray, Athel Cornish-Bowden, et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[6] Juliane Liepe, Paul Kirk, Sarah Filippi, Tina Toni, Chris P Barnes, and Michael P H Stumpf. A framework for parameter estimation and model selection from experimental data in systems biology using approximate bayesian computation. *Nature Protocols*, 9(2):439–456, January 2014.

[7] J. K. Pritchard, M. T. Seielstad, A. Perez-Lezaun, and M. W. Feldman. Population growth of human y chromosomes: a study of y chromosome microsatellites. *Molecular Biology and Evolution*, 16(12):1791–1798, December 1999.

[8] Tina Toni, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael P.H Stumpf. Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of The Royal Society Interface*, 6(31):187–202, July 2008.

[9] Marcelo S. Reis, Vincent Noël, Matheus H Dias, Layra L Albuquerque, Amanda S Guimarães, Lulu Wu, Junior Barrera, and Hugo A Armelin. An Interdisciplinary Approach for Designing Kinetic Models of the Ras/MAPK Signaling Pathway. In *Kinase Signaling Networks*, pages 455–474. Springer, 2017.

[10] Marcelo S. Reis, Gustavo Estrela, Carlos Eduardo Ferreira, and Junior Barrera. featsel: A framework for benchmarking of feature selection algorithms and cost functions. *SoftwareX*, 6:193 – 197, 2017.

[11] Marcelo S. Reis, Gustavo Estrela, Carlos Eduardo Ferreira, and Junior Barrera. Optimal Boolean lattice-based algorithms for the U-curve optimization problem. *Information Sciences*, 471:97 – 114, 2019.