Desenvolvimento de um arcabouço para inferência e seleção de modelos dinâmicos de vias de sinalização celular

Marcelo Batista De Lima Junior

Dissertação apresentada ao
Instituto de Matemática e Estatística
da Universidade de São Paulo
para obtenção do título de
Mestre em Ciências

Programa: Programa Interunidades de Pós-graduação em Bioinformática
Orientador: Prof. Dr. Marcelo da Silva Reis

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro da CNPq

São Paulo
Dezembro de 2023

Desenvolvimento de um arcabouço para inferência e seleção de modelos dinâmicos de vias de sinalização celular

Marcelo Batista De Lima Junior

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 4 de Dezembro de 2023.

Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão julgadora:

Prof. Dr. Milton Yutaka Nishiyama Junior – Instituto Butantan
Prof. Dr. Enéas de Carvalho – Instituto Butantan
Prof. Dr. Ronaldo Fumio Hashimoto – IME-USP

O conteúdo deste trabalho é publicado sob a licença CC BY 4.0
(Creative Commons Attribution 4.0 International License)

Resumo

Marcelo Batista De Lima Junior. **Desenvolvimento de um arcabouço para inferência e seleção de modelos dinâmicos de vias de sinalização celular**. Dissertação (Mestrado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2023.

Dentro do contexto de sistemas biológicos, as vias de sinalização celular são fundamentais para a sobrevivência e adaptação das células a condições externas e internas. Essas vias são compostas por uma série de reações químicas que propagam sinais e eventualmente disparam processos celulares, como a proliferação, diferenciação e morte celular. Essas reações químicas são altamente reguladas e controladas, e muitas vezes ocorrem em cascata, onde uma reação aciona a próxima em uma sequência ordenada. A complexidade dessas vias é aumentada ainda mais pelo fato de que elas se comunicam entre si, formando uma rede que coordena todas as funções celulares. No entanto, selecionar reações químicas para modelar a cinética de uma determinada via implica em desconectar essa via do restante da rede, resultando em um problema durante a inferência do modelo. Isso ocorre porque a comunicação entre vias é fundamental para que elas funcionem adequadamente e selecionar apenas algumas reações para modelar pode levar a uma perda de informação importante. Para mitigar esse problema, é preciso estimar essa comunicação faltante. Em outras palavras, é necessário modelar não apenas as reações selecionadas para a via em questão, mas também estimar como essa via interage com o restante da rede. Portanto, é necessário realizar um procedimento "aninhado", no qual cada possibilidade de seleção de reações é acompanhada da inferência de sua respectiva comunicação com o restante da rede. Neste trabalho apresentamos um arcabouço para a seleção aninhada de modelos, que acessa um banco de dados de reações para listar reações candidatas a compor um modelo e é focado na utilização de Equações Diferenciais Universais (UDEs), que combina métodos de inferência de parâmetros baseados em redes neurais com uma variedade de algoritmos de otimização tais como o gradiente descente estocástico e suas variações. Nossa abordagem também se destaca na seleção de modelos de vias de sinalização celular, na qual combinamos as métricas R^2, BIC e MAE para escolher o modelo mais adequado. Essa estratégia nos permite abordar a seleção de modelos de uma maneira abrangente e equilibrada, considerando diferentes aspectos da qualidade do ajuste. Assim, o arcabouço permite o uso integrado de bancos de reações, medidas experimentais e algoritmos de seleção de modelos, gerando resultados que são salvos em formatos padrão utilizados na comunidade de sistemas biológicos. Esperamos com este trabalho viabilizar a inferência de modelos de vias de sinalização celular que sejam mais precisos e representativos, tratando apropriadamente o problema de falta de isolamento dessas vias.

Palavras-chave: Vias de sinalização celular. Seleção de modelos. Sistemas biológicos.

Abstract

Marcelo Batista De Lima Junior. **Development of a framework for inference and selection of cell signaling pathway dynamic models**. Thesis (Master's). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2023.

Within the context of biological systems, cellular signaling pathways play a fundamental role in the survival and adaptation of cells to external and internal conditions. These pathways consist of a series of chemical reactions that transmit signals and eventually trigger cellular processes, such as proliferation, differentiation, and cell death. These chemical reactions are highly regulated and controlled, often occurring in cascades where one reaction activates the next in an ordered sequence. The complexity of these pathways is further increased by their communication with each other, forming a network that coordinates all cellular functions. However, modeling the kinetics of a specific signaling pathway faces a significant challenge. Selecting chemical reactions to model the kinetics of a pathway disconnects it from the rest of the network, resulting in a problem during model inference. This issue arises because communication between pathways is crucial for their proper functioning, and selecting only a few reactions to model can lead to significant information loss. To mitigate this problem, it is necessary to estimate this missing communication. In other words, it is essential to model not only the selected reactions but also estimate how the pathway interacts with the rest of the network. Therefore, a "nested" procedure is required, where each reaction selection possibility is accompanied by the inference of its respective communication with the rest of the network. In this work, we introduce a framework for nested model selection, which accesses a reaction database to list candidate reactions to compose a model. It is focused on the use of Universal Differential Equations (UDEs), combining parameter inference methods based on neural networks with a variety of optimization algorithms, including stochastic gradient descent and related solvers. Our approach also excels in selecting models of cellular signaling pathways, where we combine the R^2, BIC, and MAE metrics to choose the most suitable model. This strategy allows us to approach model selection comprehensively and balancedly, considering different aspects of the fitting quality. Thus, the framework enables the integrated use of reaction databases, experimental measurements, and model selection algorithms, generating results that are saved in standard formats used in the systems biology community. We expect this work contributes to the inference of cellular signaling pathway models that are more accurate and representative while appropriately addressing the problem of pathway lack of isolation.

Keywords: Cell signaling pathways. Model selection. Systems biology.

Lista de Figuras

Lista de Tabelas

Sumário

Anexos

Capítulo 1

Introdução

Todos os seres vivos, com exceção dos vírus para aqueles que os consideram como tais, são compostos por uma ou mais células, podendo chegar a trilhões em organismos complexos (Müller-Wille, 2010). As células, por sua vez, respondem a estímulos (sinais) externos: por exemplo, cada célula possui receptores distintos que reconhecem moléculas ligantes, que podem ser proteínas, hormônios ou outras substâncias que, quando se conectam a um receptor celular, transmitem um sinal que, eventualmente, desencadeia uma resposta fenotípica (e.g., leva à divisão celular) (Cooper, 2000). A transmissão do sinal de um receptor celular até os mecanismos celulares responsáveis por governar um determinado processo celular se dá através de um encadeamento de reações químicas conhecido como via de sinalização celular (Aldridge *et al.*, 2006). Desregulações em vias de sinalização estão presentes em células cancerígenas (T e P, 2017); por conta disso, essas vias vêm sendo alvo de muitas pesquisas atualmente, assim como vêm sendo criadas ferramentas que auxiliam a organizar, observar e simular o comportamento das mesmas (Shannon *et al.*, 2003). Como a propagação de sinal por essas vias se dá através da mudança de concentração das espécies químicas presentes na mesma ao longo do tempo, a modelagem matemática da cinética dessas vias é uma das ferramentas mais relevantes para estudar a biologia celular em câncer e em outros contextos.

Um método de desenhar um modelo matemático de uma via de sinalização é selecionar um subconjunto das reações químicas presentes nessa via, transcrever esse subconjunto para um sistema de equações diferenciais ordinárias (*Ordinary Differential Equations*, ODEs), e ajustar os parâmetros do modelo (i.e., constantes cinéticas e/ou concentrações iniciais) utilizando medidas experimentais e um método de inferência; exemplos desses métodos são os filtros sequenciais tal como o filtro de Kalman, cujo propósito é "utilizar medições de grandezas realizadas ao longo do tempo (contaminadas com ruído e outras incertezas) e gerar resultados que tendam a se aproximar dos valores reais das grandezas medidas e valores associados" (Wikipedia, 2019). Outras alternativas são métodos baseados em computação Bayesiana aproximada (*Approximate Bayesian Computation*, ABC), que permitem calcular a probabilidade a posteriori de um modelo sem precisar calcular a verossimilhança dos dados experimentais terem sido gerados por esse mesmo modelo (Rubin, 1984). Todavia, como o conjunto de vias de sinalização em uma célula na prática constitui uma única rede de sinalização, selecionar espécies químicas de uma via para

realizar a modelagem matemática de sua cinética implica em "desconectar" essas espécies do restante da rede; Tal falta de isolamento resulta em problemas durante a inferência dos parâmetros dos modelos, pois, durante os cálculos, uma parte da informação da via acaba sendo desconsiderada, mais especificamente, a comunicação entre as espécies recortadas da via e o restante da rede (NJ e R, 2004). Para mitigar esse problema, é necessário estimar a comunicação faltante na forma de parâmetros extras no modelo matemático, uma tarefa particularmente desafiadora devido à complexidade intrínseca das vias de sinalização celular. Recentemente, a combinação de sistemas de equações diferenciais ordinárias com redes neurais vem se tornando um método bastante utilizado para inferir a comunicação ausente nas vias de sinalização, o que pode ser feito por meio de equações diferenciais universais (*Universal Differential Equations*, UDEs) (Hartman, 2002).

Considerando o exposto, para realizar a modelagem dinâmica de uma via de sinalização por meio de modelos matemáticos com equações diferenciais universais, geralmente é necessário que os pesquisadores executem manualmente várias ferramentas independentes, cada uma responsável por uma etapa do processo, desde a importação dos dados até a inferência dos parâmetros dos modelos e sua seleção. Isso pode tornar o processo potencialmente oneroso em termos de tempo e suscetível a incorreções. Portanto, é necessário desenvolver uma abordagem mais simplificada para modelar vias de sinalização celular baseadas em UDEs. Essa abordagem deve combinar várias das etapas necessárias no processo de modelagem em uma única ferramenta e ser fácil de aplicar. Para atender a essa necessidade, propomos um novo arcabouço que integra a importação de dados de repositórios públicos de reações bioquímicas, a construção de modelos de vias de sinalização baseados em UDEs, a inferência de parâmetros e a seleção de modelos. O objetivo é possibilitar a geração de modelos de vias de sinalização celular mais precisos, que possam ajudar a entender os mecanismos dessas vias.

1.1 Objetivos

O objetivo geral deste trabalho foi o desenho e a implementação de um arcabouço para a inferência de parâmetros e seleção de modelos de vias de sinalização celular. Esse arcabouço foi projetado para proporcionar uma implementação fácil de diversos métodos de inferência, oferecendo usabilidade simplificada e e uma metodologia linear. Ele foi concebido para superar as limitações encontradas em arcabouços semelhantes, incluindo o suporte a sistemas de UDEs e a capacidade de importar facilmente dados de repositórios públicos de reações bioquímicas.

Para atingir esse objetivo, utilizamos a linguagem Julia para desenvolver uma biblioteca com funções responsáveis por realizar operações de inferência de parâmetros e seleção de modelos utilizando sistemas de UDEs. Além disso, empregamos a linguagem C++ para desenvolver uma aplicação auxiliar capaz de converter modelos bioquímicos importados da extensão de banco de dados Anguix (Montoni, Sousa *et al.*, 2022) para o formato *Systems Biology Markup Language* (SBML) (Hucka, Finney, Sauro, Bolouri, J. C. Doyle *et al.*, 2003), que é o formato padrão de modelos utilizado pela comunidade de sistemas biológicos.

Cada uma dessas linguagens oferece vantagens únicas: C++ é conhecido por seu alto

desempenho e capacidade de gerenciamento de memória, enquanto Julia é uma linguagem especificamente projetada para computação científica, com sintaxe simples e desempenho comparável ao de C++. Julia também possui suporte integrado para computação paralela e arquiteturas de memória distribuída, tornando-a adequada para lidar com grandes conjuntos de dados. Juntas, essas duas linguagens permitem que o arcabouço seja usado em uma ampla variedade de tarefas com facilidade.

Para testar as funções da biblioteca, realizamos experimentos utilizando um modelo importado e convertido do banco de dados Anguix em um notebook Jupyter (Kluyver *et al.*, 2016), Jupyter é um ambiente de desenvolvimento interativo que permite manipular arquivos que contêm código executável, equações e texto.

1.2 Organização do trabalho

Após esta introdução, no **capítulo 2** discutiremos os fundamentos teóricos deste trabalho e realizaremos uma revisão da literatura e trabalhos relacionados. Adiante no **capítulo 3** detalharemos a metodologia utilizada para desenvolver nosso arcabouço de inferência e seleção de modelos de sinalização celular, incluindo a integração de dados experimentais, a modelagem com Equações Diferenciais Universais (UDEs), os métodos de inferência de parâmetros e seleção de modelos, bem como as métricas de avaliação de desempenho. No **capítulo 4** apresentaremos os resultados obtidos ao aplicar nosso arcabouço a dados experimentais. Realizamos um exemplo de execução da inferência de parâmetros em um modelo de exemplo, no qual aplicamos dois algoritmos de inferência diferentes. Em seguida, utilizando nosso método que combina três métricas, conduzimos uma seleção entre os modelos gerados por cada algoritmo. Analisaremos cuidadosamente os resultados desse exemplo, destacando as descobertas e conclusões que surgiram dessa análise comparativa. Essa abordagem nos permitiu avaliar a eficácia do nosso arcabouço em identificar modelos mais precisos e representativos em diferentes cenários, além de demonstrar a utilidade de nosso método de seleção de modelos no contexto da modelagem de vias de sinalização celular. Por fim, no **capítulo 5** faremos uma recapitulação das principais descobertas e contribuições deste trabalho, além de abrirmos perspectivas para futuras direções de pesquisa baseadas nos resultados discutidos. Também abordaremos as limitações do arcabouço e considerações finais que ressaltam a importância desse trabalho no contexto da modelagem de vias de sinalização celular e da biologia de sistemas.

Capítulo 2

Fundamentação teórica e revisão bibliográfica

Neste capítulo, exploraremos os conceitos teóricos essenciais relacionados a este projeto. Começaremos elucidando o funcionamento das vias de sinalização molecular, destacando seu papel crucial na comunicação celular. Em seguida, vamos nos aprofundar no campo da Cinética Química, discutindo os princípios e métodos utilizados para descrever as taxas de velocidade das reações químicas. Também enfatizaremos a importância da modelagem por meio de sistemas de equações diferenciais ordinárias e mostraremos como isso funciona. Por fim, abordaremos o uso de bancos de dados de reações bioquímicas, fontes valiosas de informações fundamentais para o desenvolvimento deste projeto.

2.1 Vias de sinalização celular

As células possuem uma intrincada rede de comunicação interna que lhes permite responder a estímulos do ambiente e coordenar uma ampla gama de processos fisiológicos. Essa comunicação ocorre por meio de vias de sinalização celular, que consistem em uma série de eventos bioquímicos cuidadosamente orquestrados. As vias de sinalização celular são sistemas de transmissão de informações que permitem que as células percebam, processem e respondam a estímulos específicos, como hormônios, fatores de crescimento ou mudanças nas condições ambientais. Essas vias são compostas por proteínas, moléculas sinalizadoras e componentes estruturais, que trabalham em conjunto para transmitir o sinal desde o local de origem até o local de resposta no interior da célula (Alberts *et al.*, 2014). Uma via de sinalização celular típica começa com um ligante, como um hormônio, que se liga a um receptor na membrana celular ou no interior da célula. Essa ligação desencadeia uma série de eventos intracelulares, incluindo a ativação de proteínas sinalizadoras específicas. Essas proteínas sinalizadoras atuam como mensageiros, transmitindo o sinal para proteínas-alvo adicionais dentro da célula; É possível observar todo esse processo de forma simplificada na figura 2.1.

Uma das principais classes de proteínas sinalizadoras são as quinases, que adicionam grupos fosfato a outras proteínas em um processo chamado fosforilação (Lemmon e

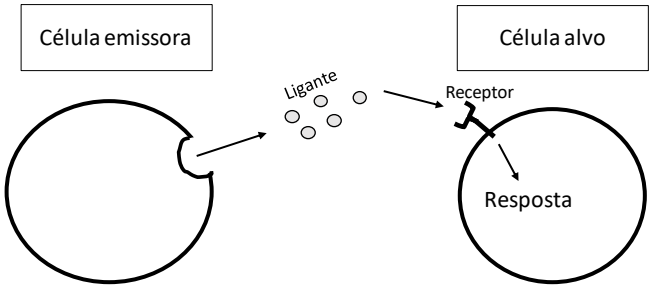

Figura 2.1: *Representação simplificada do processo de sinalização celular, com uma célula emissora liberando um ligante, que é captado pelo receptor da célula alvo e posteriormente desencadeia processos celulares. Representação do processo inspirada na figura originalmente publicada pela Khan Academy (pt.khanacademy.org/science/biology/cell-signaling/mechanisms-of-cell-signaling/a/introduction-to-cell-signaling).*

Schlessinger, 2010). A fosforilação de proteínas-alvo pode desencadear uma cascata de eventos, ativando ou inibindo diferentes proteínas e modificando sua função. Essa cascata de fosforilação é uma característica comum em muitas vias de sinalização celular.

Além da fosforilação, outros mecanismos de sinalização incluem a ativação de segundos mensageiros, como o íon cálcio, que pode agir como um intermediário crucial para amplificar e transmitir o sinal (Clapham, 2007); Ativação de genes específicos, resultando em mudanças na expressão gênica; Translocação de proteínas para diferentes compartimentos celulares, permitindo que as células controlem a localização das proteínas e, assim, sua função (Rapoport, 2007). Esses diversos mecanismos são essenciais para a transdução de sinais e a coordenação das respostas celulares específicas. As vias de sinalização celular são peças-chave em uma miríade de processos biológicos, incluindo crescimento e desenvolvimento, diferenciação celular, resposta imunológica, sobrevivência celular e muitos outros. Não é surpreendente, portanto, que disfunções nesses caminhos estejam intimamente relacionadas a uma ampla variedade de doenças humanas. Doenças como o câncer, doenças cardiovasculares, distúrbios metabólicos e doenças neurodegenerativas frequentemente têm suas raízes nas intricadas falhas das vias de sinalização celular (Lodish *et al.*, 2000).

As vias de sinalização celular são classificadas em diferentes categorias com base em sua localização e mecanismos de transdução de sinal. Uma das categorias mais comuns é a via de sinalização por receptor de superfície celular. Nesse tipo de via, o ligante sinalizador se liga a um receptor na membrana celular, desencadeando uma cascata de eventos intracelulares. Esses eventos podem incluir a ativação de enzimas, a liberação de segundos mensageiros e a transcrição de genes específicos. Exemplos conhecidos de vias de sinalização por receptor de superfície celular incluem a via do fator de crescimento epidérmico (EGF) (Carpenter, 2014) e a via do receptor de insulina (White, 2002).

Outra categoria importante é a via de sinalização intracelular, que ocorre inteiramente no interior da célula. Essas vias são ativadas por moléculas sinalizadoras que penetram na célula, como hormônios esteróides. Uma vez dentro da célula, esses hormônios se ligam a receptores nucleares, que atuam como fatores de transcrição para regular a expressão gênica. Exemplos de vias de sinalização intracelular incluem a via de sinalização do hormônio esteróide (Evans, 1988) e a via de sinalização do receptor de vitamina D (Mangelsdorf e Evans, 1995). Além dessas categorias, existem outras vias de sinalização celular menos conhecidas, como a via de sinalização mitocondrial (Green *et al.*, 2011), a via de sinalização do estresse (Ron e Walter, 2002) e a via de sinalização da morte celular programada (apoptose) (Elmore, 2007). Essas vias desempenham papéis importantes na regulação do metabolismo energético, na resposta a estresses celulares e na eliminação de células danificadas ou indesejadas.

Uma característica importante das vias de sinalização celular é a sua interconectividade e "crosstalk" (ou cruzamento de sinais) (Kholodenko, 2000). Isso significa que diferentes vias podem interagir e influenciar umas às outras, formando uma rede complexa de sinalização intracelular. Essa interconectividade permite que as células integrem múltiplos sinais e coordenem respostas adaptativas a diferentes estímulos ambientais. No entanto, também pode levar a perturbações na sinalização celular quando ocorrem disfunções em uma ou mais vias.

A compreensão das vias de sinalização celular tem sido impulsionada por avanços tecnológicos e experimentais, incluindo estudos genéticos, análises de expressão gênica, proteômica e análises computacionais. Essas abordagens têm permitido a identificação e caracterização de novos componentes das vias de sinalização, bem como a descoberta de mecanismos de regulação sofisticados. No contexto médico, o conhecimento das vias de sinalização celular tem sido fundamental para o desenvolvimento de terapias direcionadas. Terapias direcionadas são abordagens terapêuticas que visam componentes específicos de vias de sinalização que estão disfuncionais em determinadas doenças. Essas terapias têm mostrado sucesso em diversos tipos de câncer, onde a disfunção de vias de sinalização específicas está frequentemente associada à progressão da doença (Gerber, 2008).

Em conclusão, as vias de sinalização celular desempenham um papel central na regulação de processos biológicos essenciais. Sua compreensão é fundamental para avançarmos em nosso conhecimento sobre a biologia celular e para o desenvolvimento de terapias inovadoras. A contínua investigação das vias de sinalização celular e sua interconectividade promete revelar novas compreensões sobre a fisiologia celular e abrir novas perspectivas para o tratamento de doenças.

2.2 Cinética química

A cinética química é o ramo da química que estuda a velocidade das reações químicas e os fatores que as influenciam. É uma área de pesquisa essencial, pois nos permite entender como as reações ocorrem e como sua taxa de reação pode ser modificada por meio de diferentes condições experimentais. Nesta seção serão abordados os conceitos fundamentais relacionados à cinética química, fornecendo uma compreensão mais aprofundada sobre as reações químicas e seu comportamento dinâmico.

Ao investigar a cinética química, os cientistas buscam responder a perguntas cruciais, tais como: Como as moléculas colidem e interagem durante uma reação? Quais são os mecanismos pelos quais as ligações químicas são rompidas e formadas? Quais são as etapas intermediárias envolvidas no processo de reação? Esses estudos fornecem uma visão mais profunda do comportamento das reações químicas e permitem predizer e controlar suas taxas de reação.

A compreensão da cinética química é de grande importância em várias áreas da química aplicada. Por exemplo, na síntese de materiais, conhecer a velocidade das reações é essencial para otimizar os processos de produção, permitindo a fabricação de materiais com propriedades desejadas em tempo hábil. No desenvolvimento de medicamentos, compreender a cinética das reações químicas é fundamental para garantir que os compostos sejam produzidos de forma eficiente e segura, evitando a formação de subprodutos indesejados ou reações adversas. Além disso, a otimização de processos industriais depende da compreensão dos fatores que influenciam a velocidade das reações, permitindo melhorar a eficiência, reduzir custos e minimizar impactos ambientais (Marin *et al.*, 2021).

2.2.1 Reações químicas

As reações químicas constituem processos fundamentais na química, nos quais substâncias químicas interagem entre si, resultando na formação de novas substâncias. Essas transformações são descritas por meio de equações químicas, que fornecem uma representação simbólica dos reagentes, produtos, proporções e estados físicos envolvidos na reação.

Durante uma reação química, ocorrem alterações nas ligações químicas entre os átomos dos reagentes, levando à reorganização desses átomos para formar os produtos. Esse processo pode envolver o rompimento de ligações existentes e a formação de novas ligações químicas, permitindo a transformação das substâncias iniciais em produtos distintos.

É importante destacar que existem diferentes tipos de reações químicas, cada uma seguindo um mecanismo específico. Alguns exemplos comuns incluem as reações de síntese, nas quais duas ou mais substâncias se combinam para formar um produto mais complexo; as reações de decomposição, que envolvem a quebra de uma substância em produtos mais simples; as reações de substituição, nas quais um elemento ou grupo é substituído por outro em uma molécula; e as reações em equilíbrio químico, nas quais as taxas das reações direta e inversa se igualam, resultando em uma composição constante dos reagentes e produtos (Mortimer, 2008). Além disso, as reações químicas podem ser influenciadas por uma variedade de fatores, tais como a concentração dos reagentes, a temperatura, a pressão e a presença de catalisadores.

O estudo das reações químicas e de seus mecanismos é de suma importância em diversas áreas da química, desde a síntese de compostos orgânicos até a compreensão das reações bioquímicas que ocorrem nos organismos vivos. A capacidade de compreender, prever e controlar as reações químicas é essencial para o desenvolvimento de novos materiais, a fabricação de produtos químicos, a produção de medicamentos e a otimização de processos industriais.

2.2.2 Velocidade de reação

A velocidade de uma reação química refere-se à taxa de variação das concentrações dos reagentes e produtos em relação ao tempo. É expressa em termos de quantidade de reagente consumido ou produto formado por unidade de tempo. A velocidade de uma reação pode ser determinada medindo-se as mudanças nas concentrações ao longo do tempo ou através de outras técnicas experimentais.

A velocidade de uma reação química pode ser influenciada por diversos fatores. Entre os principais fatores, destacam-se:

- Concentração dos reagentes: Aumentos na concentração dos reagentes geralmente resultam em uma maior frequência de colisões entre as moléculas, o que aumenta a probabilidade de reações ocorrerem. Quanto maior a concentração dos reagentes, maior será a taxa de reação.
- Temperatura: O aumento da temperatura aumenta a energia cinética das moléculas, levando a uma maior velocidade das colisões. Com maior energia cinética, as moléculas colidem com mais força e maior frequência, resultando em uma taxa de reação mais rápida. De fato, de acordo com a Equação de Arrhenius, a velocidade de uma reação química geralmente dobra a cada aumento de 10°C na temperatura (Peleg *et al.*, 2012).
- Superfície de contato: Quando os reagentes estão em diferentes fases (sólido, líquido ou gás), uma maior área de superfície de contato entre eles aumenta a velocidade da reação. Isso ocorre porque uma maior área de contato permite um maior número de colisões efetivas entre as moléculas, proporcionando mais oportunidades para as reações ocorrerem.
- Catalisadores: Os catalisadores são substâncias que aumentam a velocidade de uma reação, fornecendo um caminho alternativo de menor energia para a reação ocorrer. Eles não são consumidos durante a reação e podem acelerar a taxa de reação em muitas ordens de magnitude. Os catalisadores facilitam a quebra de ligações químicas nos reagentes, reduzindo a energia de ativação necessária para a reação ocorrer. Isso resulta em uma maior eficiência e velocidade na formação dos produtos.

A velocidade de uma reação química pode ser expressa matematicamente pela equação da velocidade (Levine, 2019), que relaciona a taxa de variação das concentrações com as concentrações dos reagentes. Para uma reação genérica da forma:

$$aA + bB \longrightarrow cC + dD, \tag{2.1}$$

onde “A” e “B” são os reagentes, “C” e “D” são os produtos, e “a”, “b”, “c” e “d” são os coeficientes estequiométricos, a equação da velocidade pode ser escrita como:

$$v = k[A]^m[B]^n, \tag{2.2}$$

onde “k” é a constante de velocidade da reação, “[A]” e “[B]” representam as concentrações dos reagentes, e “m” e “n” são os expoentes da ordem de reação em relação a cada reagente.

2.2.3 Equações diferenciais ordinárias (ODEs) de reações químicas

Em alguns casos, a cinética de uma reação química pode ser descrita por meio de equações diferenciais ordinárias (ODEs, do inglês *Ordinary Differential Equations*), que proporcionam uma abordagem matemática poderosa para compreender a evolução temporal das concentrações de reagentes e produtos em uma reação química (Higham, 2008). Essas equações relacionam as taxas de variação das concentrações com as próprias concentrações e fornecem uma descrição mais detalhada e precisa da dinâmica das reações. Considere uma reação genérica da forma:

$$aA + bB \longrightarrow cC + dD. \tag{2.3}$$

A taxa de variação das concentrações dos reagentes em relação ao tempo pode ser expressa pelas ODEs, conforme exemplificado abaixo:

$$\frac{d[A]}{dt} = -k[A]^m[B]^n \tag{2.4a}$$

$$\frac{d[B]}{dt} = -k[A]^m[B]^n \tag{2.4b}$$

$$\frac{d[C]}{dt} = k[A]^m[B]^n \tag{2.4c}$$

$$\frac{d[D]}{dt} = k[A]^m[B]^n. \tag{2.4d}$$

Nessas equações, "d[X]/dt" representa a taxa de variação da concentração da espécie "X" em relação ao tempo "t", e as equações são determinadas pelos coeficientes estequiométricos, pela ordem de reação e pela constante de velocidade "k".

A resolução dessas equações diferenciais é um processo matemático que envolve técnicas analíticas ou numéricas, dependendo da complexidade do sistema. A solução das ODEs de reações químicas permite determinar as concentrações das espécies envolvidas ao longo do tempo, fornecendo uma descrição mais completa da dinâmica da reação. A utilização das ODEs de reações químicas é especialmente relevante em situações em que a taxa de reação é influenciada por diversos fatores, como a variação das concentrações de reagentes, a presença de catalisadores, a temperatura e a pressão. Ao incorporar essas informações nas equações diferenciais, é possível estudar a influência desses fatores na cinética da reação e prever o comportamento do sistema químico em diferentes condições experimentais.

A aplicação das ODEs de reações químicas é amplamente utilizada em diversas áreas da química. Na síntese de materiais, por exemplo, o estudo das ODEs permite otimizar as condições de reação para obter produtos com propriedades desejadas, controlando a velocidade e a seletividade das reações. Na produção de medicamentos, as ODEs são fundamentais para determinar os perfis de liberação de fármacos e a eficiência das reações de síntese. Além disso, na otimização de processos industriais, as ODEs são empregadas para maximizar a produtividade e minimizar os custos de produção.

2.3 Bancos de dados de reações

Bancos de dados de reações bioquímicas são repositórios de informações que descrevem as reações químicas que ocorrem nos sistemas biológicos. Eles desempenham um papel fundamental na pesquisa bioquímica e biológica, permitindo aos cientistas explorar e analisar a complexidade dos sistemas biológicos e compreender os mecanismos das reações bioquímicas em uma variedade de processos biológicos, como o metabolismo, a sinalização celular e a síntese de proteínas. Esses bancos de dados são construídos a partir de dados experimentais e computacionais, com a ajuda de ferramentas de modelagem e simulação. Os dados experimentais são obtidos de estudos científicos, que envolvem análises químicas e bioquímicas para medir as propriedades físicas e químicas das moléculas envolvidas nas reações biológicas. Além disso, técnicas como espectroscopia, cristalografia e microscopia são utilizadas para obter informações estruturais detalhadas das moléculas.

Por outro lado, os dados computacionais são obtidos por meio de métodos de modelagem e simulação. Essas abordagens incluem a utilização de algoritmos e software especializados para prever as propriedades químicas e físicas das moléculas, bem como para simular as reações bioquímicas em nível atômico ou molecular. Esses métodos são fundamentados em teorias e princípios químicos e físicos, como a mecânica quântica, a termodinâmica e a dinâmica molecular. A integração desses dados experimentais e computacionais é um desafio importante na construção dos bancos de dados de reações bioquímicas. É necessário desenvolver métodos e padrões para armazenar e organizar essas informações de forma consistente e acessível. Além disso, a interoperabilidade entre diferentes bancos de dados é essencial para permitir a integração e a troca de dados entre diferentes sistemas.

Existem vários bancos de dados de reações bioquímicas disponíveis atualmente, cada um com suas próprias vantagens e limitações. O Reactome (Joshi-Tope *et al.*, 2005) é um exemplo amplamente conhecido de um banco de dados abrangente baseado em grafos, que permite uma representação visual eficiente e uma busca rápida das informações sobre as reações bioquímicas. O Reactome oferece uma visão integrada de vias metabólicas, interações de proteínas e processos de sinalização celular. É importante notar que o Reactome Graph Database (RDB) é uma implementação específica dentro do ecossistema maior do Reactome, focando em dados relacionados a grafos e interações moleculares (Miller *et al.*, 2019). A interface do Reactome pode ser observada na figura 2.2. Outro exemplo é o SABIO-RK (Wittig *et al.*, 2012), que se destaca por fornecer dados quantitativos das constantes de velocidade das reações bioquímicas. Essas informações quantitativas são essenciais para a modelagem matemática e a simulação computacional de sistemas biológicos, permitindo uma compreensão mais precisa dos mecanismos de reações e uma melhor predição de comportamentos bioquímicos. A interface do banco de dados SABIO-RK pode ser observada na figura 2.3.

Além desses, há outros bancos de dados de reações bioquímicas amplamente utilizados, como o KEGG (Kanehisa *et al.*, 2016), BioCyc (Karp *et al.*, 2002) e BRENDA (Jeske *et al.*, 2021). Cada um desses bancos de dados possui sua própria abordagem de organização e apresentação dos dados, bem como suas áreas de especialização. O KEGG, por exemplo, é conhecido por sua ênfase em vias metabólicas, enquanto o BioCyc oferece uma coleção abrangente de vias metabólicas e informações genômicas. O BRENDA, por sua vez, se

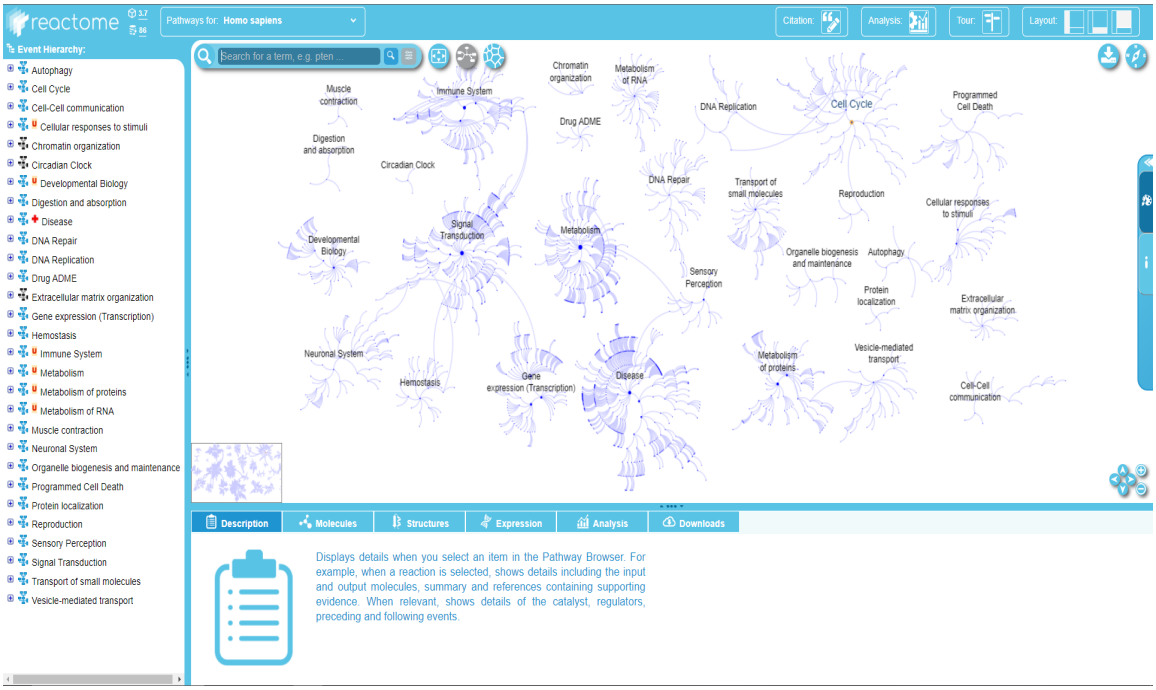

Figura 2.2: *Captura de tela da página inicial do Reactome pathway browser v.(3.7), exibindo diversas vias metabólicas e processos biológicos disponíveis para acesso. Fonte: Reactome (reactome.org/PathwayBrowser/).*

Figura 2.3: *Interface do banco de dados SABIO-RK exibindo a área de busca de informações. Fonte: SABIO-RK (sabio.h-its.org)*

concentra principalmente em dados cinéticos de enzimas. Esses bancos de dados fornecem uma fonte valiosa de informações para a comunidade científica, permitindo o avanço do conhecimento e o desenvolvimento de novas terapias e tratamentos para uma variedade de doenças. Eles são usados por pesquisadores acadêmicos, cientistas da indústria farmacêutica e médica, e até mesmo por empresas de biotecnologia.

Para este projeto em particular, a ferramenta Anguix é utilizada como uma extensão do Reactome Graph Database (Montoni, Sousa *et al.*, 2022; Montoni, De Sousa *et al.*, 2022). Essa extensão permite a importação de dados cinéticos do banco de dados SABIO-RK para o Reactome, aprimorando a precisão e a confiabilidade dos modelos gerados. Ao incorporar dados cinéticos, o Anguix oferece uma visão mais detalhada das reações bioquímicas, permitindo análises mais aprofundadas e previsões mais precisas.

Em suma, os bancos de dados de reações bioquímicas são recursos indispensáveis para a pesquisa bioquímica e biológica. Eles fornecem uma plataforma para armazenar, organizar e compartilhar informações sobre reações químicas e suas propriedades físicas e químicas. Com a contínua expansão desses bancos de dados e a melhoria das técnicas de modelagem e simulação, a compreensão dos sistemas biológicos e o desenvolvimento de terapias e tratamentos inovadores continuarão avançando em ritmo acelerado.

2.4 Inferência de parâmetros

No contexto de modelos de vias de sinalização celular, a inferência de parâmetros refere-se ao processo de estimar os valores dos parâmetros desconhecidos do modelo com base em dados experimentais. Esses parâmetros descrevem as características cinéticas das interações bioquímicas representadas no modelo, como taxas de reação, constantes de equilíbrio e afinidades de ligação. A inferência de parâmetros é uma etapa crucial na validação e refinamento dos modelos de vias de sinalização celular, pois permite comparar as previsões do modelo com os dados experimentais e ajustar os parâmetros para melhorar a concordância entre eles. Essa abordagem permite testar a capacidade do modelo em reproduzir de forma precisa e quantitativa o comportamento observado nas células.

Existem várias técnicas de inferência de parâmetros disponíveis, incluindo métodos baseados em otimização, métodos de ajuste de curvas e abordagens estatísticas (Gelman *et al.*, 2013). Essas técnicas exploram os dados experimentais, como séries temporais de concentrações de moléculas ou atividades enzimáticas, para encontrar os conjuntos de parâmetros que minimizam a diferença entre as previsões do modelo e os dados reais. Uma abordagem comumente utilizada na inferência de parâmetros de modelos de vias de sinalização celular é a minimização do erro quadrático (Hastie *et al.*, 2009), onde o objetivo é minimizar a soma dos quadrados das diferenças entre as previsões do modelo e os dados experimentais. Isso envolve a definição de uma função objetivo que quantifica o erro entre o modelo e os dados, e a utilização de algoritmos de otimização para encontrar os valores ideais dos parâmetros que minimizam essa função.

É importante ressaltar que a inferência de parâmetros está intimamente relacionada com as equações diferenciais ordinárias (ODEs) que descrevem a dinâmica das concentrações das espécies químicas ao longo do tempo. As ODEs são derivadas a partir dos modelos

de vias de sinalização celular e representam as taxas de variação das concentrações das espécies em função dos parâmetros e das interações moleculares descritas no modelo.

Portanto, a inferência de parâmetros é uma ferramenta essencial para o refinamento e validação de modelos de vias de sinalização celular, permitindo que os pesquisadores ajustem os parâmetros desconhecidos do modelo para melhor representar os dados experimentais. Essa abordagem contribui para a compreensão mais precisa e quantitativa dos processos moleculares envolvidos nas vias de sinalização celular.

2.5 Seleção de modelos

No contexto de modelos de vias de sinalização celular, a seleção de modelos refere-se ao processo de avaliar e comparar diferentes modelos matemáticos para determinar qual deles melhor descreve os dados experimentais e os conhecimentos prévios sobre a via de sinalização em estudo. A seleção de modelos desempenha um papel crucial no desenvolvimento e refinamento dos modelos de vias de sinalização celular, uma vez que diferentes abordagens podem levar a diferentes previsões e interpretações dos dados experimentais. Portanto, é fundamental identificar o modelo mais apropriado que representa de forma precisa e concisa o sistema biológico em estudo.

Existem várias estratégias e critérios para a seleção de modelos de vias de sinalização celular. Uma abordagem comumente utilizada é a comparação de modelos por meio de estatísticas de ajuste, como o erro médio quadrático (*Mean Squared Error*, MSE) (Willmott e Matsuura, 2005), o critério de informação de Akaike (*Akaike Information Criterion*, AIC) (Akaike, 1974) e o critério de informação Bayesiano (Bayesian Information Criterion, BIC) (Schwarz, 1978b). Essas estatísticas quantificam a qualidade de ajuste entre os modelos e os dados experimentais, considerando a complexidade de cada modelo.

Outra estratégia é a utilização de técnicas de validação cruzada (Kohavi, 1995), onde os dados experimentais são divididos em conjuntos de treinamento e teste. Os modelos são ajustados aos dados de treinamento e, em seguida, avaliados em relação aos dados de teste. Essa abordagem permite verificar a capacidade do modelo em fazer previsões precisas em dados não utilizados no ajuste dos parâmetros.

Além disso, a seleção de modelos também pode envolver a comparação de diferentes hipóteses biológicas e mecanismos de ação, levando em consideração a consistência com os conhecimentos prévios e as evidências experimentais disponíveis. Isso pode ser feito por meio de análises de sensibilidade, análises de identificabilidade e comparação com estudos anteriores.

É importante destacar que a seleção de modelos está diretamente relacionada com a inferência de parâmetros, uma vez que diferentes modelos podem ter diferentes conjuntos de parâmetros a serem estimados. Portanto, a seleção de modelos e a inferência de parâmetros são processos interligados e complementares no desenvolvimento e refinamento de modelos de vias de sinalização celular.

Em resumo, a seleção de modelos é uma etapa crítica no estudo de vias de sinalização celular, pois permite escolher o modelo mais adequado que melhor descreve os dados experimentais e os conhecimentos biológicos. Essa abordagem contribui para uma compreensão

mais aprofundada dos processos moleculares envolvidos nas vias de sinalização celular e facilita a identificação de alvos terapêuticos, o desenvolvimento de terapias direcionadas e a otimização de processos biotecnológicos.

2.6 Revisão bibliográfica

Nesta seção, faremos uma revisão da literatura, analisando aplicações desenvolvidas com propósitos semelhantes aos deste trabalho. No entanto, é importante destacar que essas aplicações apresentam significativas limitações, as quais serão mencionadas de forma mais específica nos exemplos a seguir.

2.6.1 Copasi

O COPASI (Hoops *et al.*, 2006) é um simulador que oferece uma boa usabilidade através de sua interface gráfica relativamente simples. Ele permite que o usuário edite modelos e infira parâmetros de suas vias. No entanto, o software carece de uma representação visual dos modelos fácil de entender e suporte direto para sistemas de UDEs. Além disso, pode demandar bastante tempo para se aprender a utilizar e exige um conhecimento prévio considerável em biologia celular. Vale ressaltar que o COPASI não é capaz de realizar seleção de modelos, sendo restrito apenas à inferência de parâmetros e simulações. Sua interface pode ser observada na figura 2.4.

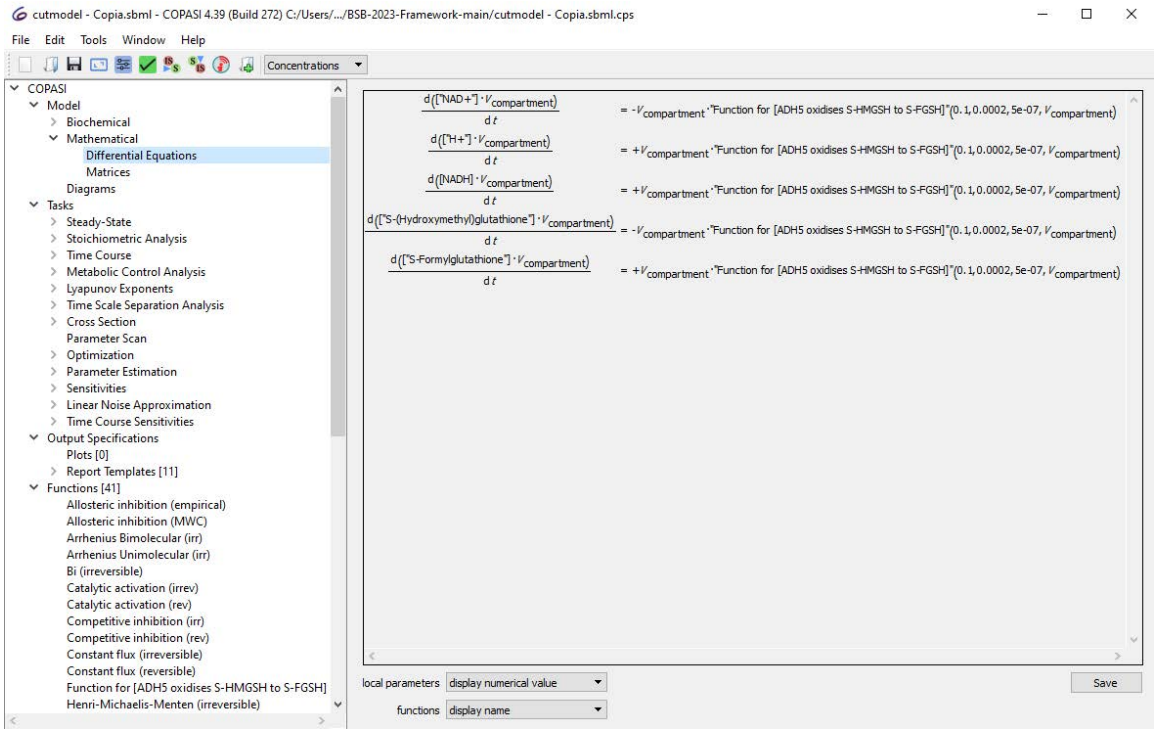

Figura 2.4: *Captura de tela do software Copasi v.(4.39), com um exemplo de modelo bioquímico tendo suas equações diferenciais definidas. Fonte: Copasi (www.copasi.org).*

2.6.2 Cell Designer

A aplicação Cell Designer (Matsuoka *et al.*, 2014) possui funcionalidades semelhantes ao COPASI, mas se destaca por oferecer uma maneira mais intuitiva de criar e manipular modelos. Neste software, é possível realizar a maior parte das ações clicando e arrastando com o mouse, enquanto se observa graficamente o modelo sendo desenhado em tempo real no padrão SBGN (*Systems Biology Graphical Notation*) (Le Novère *et al.*, 2009), como ilustrado na figura 2.5. No entanto, assim como o COPASI, o Cell Designer não possui suporte para sistemas de UDEs. Além disso, aprender a utilizar o Cell Designer também pode demandar bastante tempo e requer um conhecimento prévio considerável em biologia celular. Assim como o COPASI, o Cell Designer é limitado à inferência de parâmetros e simulações, sem a capacidade de realizar seleção de modelos.

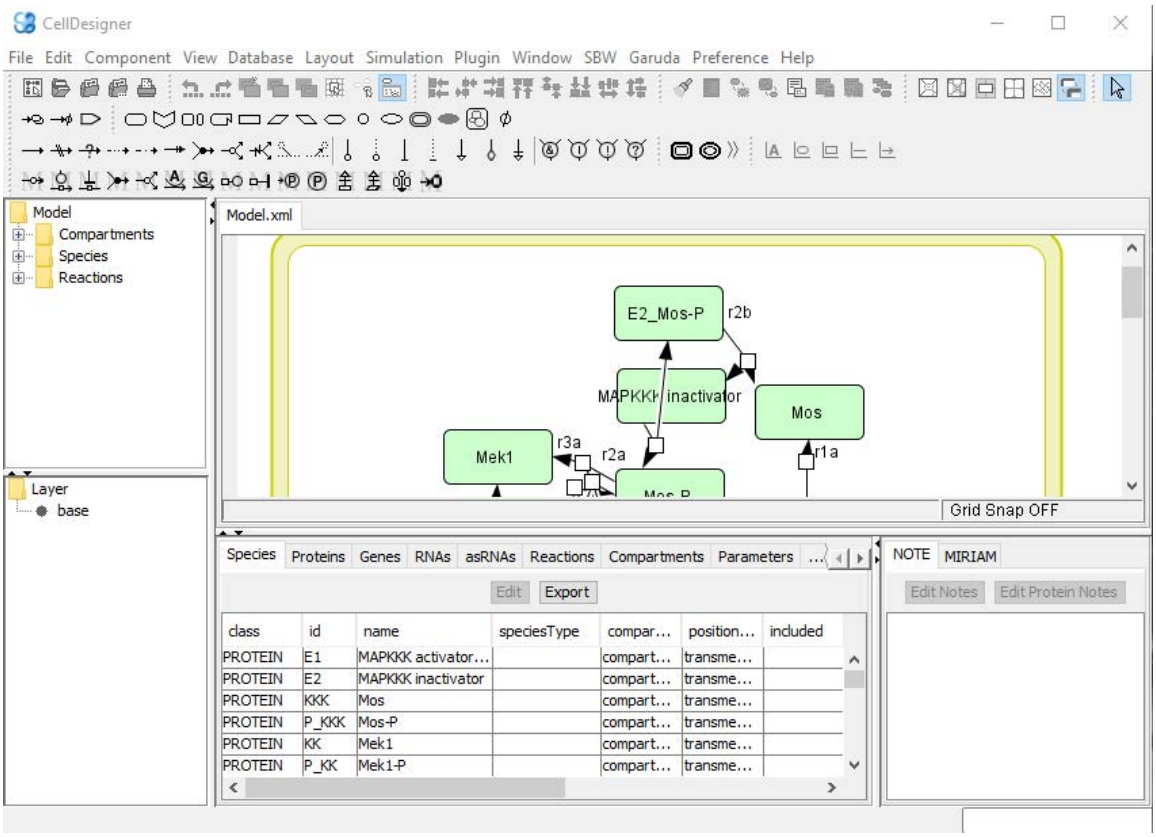

Figura 2.5: *Captura de tela do software Cell Designer v.(4.4.2), com um exemplo de modelo bioquímico sendo construído. Fonte: Cell Designer (www.celldesigner.org).*

2.6.3 Garuda e Systems Biology Workbench

Garuda é uma aplicação que tem como objetivo possibilitar o uso paralelo de outras aplicações do contexto de sistemas biológicos, se trata de uma plataforma que fornece um buscador de aplicações através do qual é possivel utilizar, por exemplo, o COPASI em uma guia e o Cell Designer em outra, compartilhando informações de arquivos entre ambos ou com demais aplicações do catálogo, como pode ser observado na figura 2.6. O Systems Biology Workbench (SBW) (Hucka, Finney, Sauro, Bolouri, J. Doyle *et al.*, 2001) funciona de maneira semelhante, com a diferença de que seu catálogo oferece um

número mais limitado de ferramentas próprias, mas que incluem uma ferramenta de simulação bastante conhecida que inclusive é acessível por algumas das aplicações do Garuda. No entanto, não existe nenhuma aplicação no catálogo dessas plataformas que possibilite a seleção de modelos, tampouco aplicações com suporte direto a sistemas de UDEs.

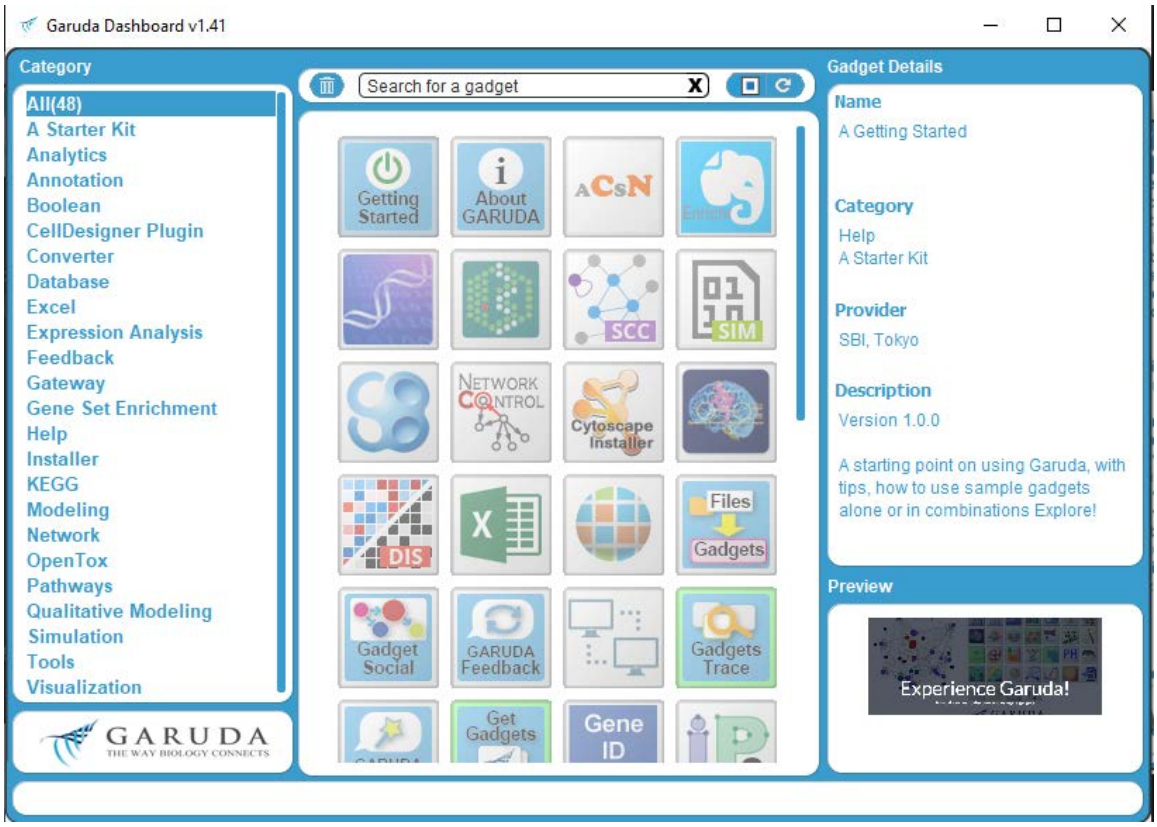

Figura 2.6: *Captura de tela do buscador de aplicativos do software Garuda v.(1.41), com diversas opções de aplicações no contexto de sistemas biológicos. Fonte: Garuda (www.garuda-alliance.org).*

2.6.4 ABC-SysBio

O ABC-SysBio é um arcabouço capaz de realizar tanto a inferência de parâmetros (utilizando um método ABC) quanto a seleção de modelos (Liepe *et al.*, 2010); porém, esse arcabouço possui uma usabilidade complicada visto que não comporta uma interface gráfica própria e funciona por linha de comando, o que o torna dificil de configurar, além disso, a aplicação foi desenvolvida especificamente para suportar modelos dinâmicos baseados em ODEs, não UDEs.

Por fim, nenhuma dessas aplicações possui uma funcionalidade capaz de lidar com o problema de falta de isolamento de recortes de vias de sinalização que foi mencionado anteriormente. Portanto, consideramos importante o desenvolvimento de uma ferramenta mais adequada para a inferência de modelos de vias de sinalização celular.

Capítulo 3

Metodologia

Neste capítulo, serão descritos em detalhes os procedimentos e técnicas utilizadas pelo arcabouço desenvolvido para importação dos dados, bem como as estratégias empregadas para a inferência e seleção de modelos de vias de sinalização celular. A metodologia proposta tem como objetivo superar desafios existentes na área, contribuindo para avanços significativos na compreensão dos processos moleculares que ocorrem nas células por meio da geração de modelos de maior qualidade. Isso permitirá abrir caminho para aplicações futuras no campo da biologia computacional.

Um fluxograma que elucida o fluxo de trabalho do arcabouço pode ser observado na figura 3.1. A primeira etapa da metodologia engloba o processo de importação dos dados. Será apresentado o procedimento utilizado para acessar o Anguix, uma extensão do banco de dados Reactome, e empregar essa ferramenta para a importação dos dados relacionados às vias de sinalização de um organismo específico. Em seguida, será descrito o conjunto de procedimentos necessários para buscar um modelo específico utilizando os dados importados. Serão detalhados os métodos empregados na extração e integração dos dados, visando assegurar a confiabilidade e qualidade das informações utilizadas. Em seguida, serão detalhadas as estratégias empregadas na inferência e seleção de modelos de vias de sinalização celular. Serão discutidos os algoritmos de aprendizado de máquina utilizados, incluindo redes neurais, que serão treinadas com os dados importados do banco de dados e serão capazes de simular dados faltantes. Por fim, o modelo de maior qualidade será determinado utilizando uma função específica do arcabouço que utiliza três métricas de seleção em conjunto.

3.1 Arcabouço proposto

O arcabouço proposto inclui uma biblioteca contendo diversas funções desenvolvidas na linguagem de programação Julia que são responsáveis pelos métodos de inferência e seleção de modelos, além de uma aplicação auxiliar desenvolvida para lidar com a importação dos dados utilizando a linguagem C++. Essa seleção estratégica das linguagens de programação foi baseada nas vantagens exclusivas oferecidas por cada uma dessas linguagens. C++ é amplamente reconhecida por sua capacidade de alto desempenho e gerenciamento eficiente de memória, tornando-a ideal para tarefas que envolvem cálculos

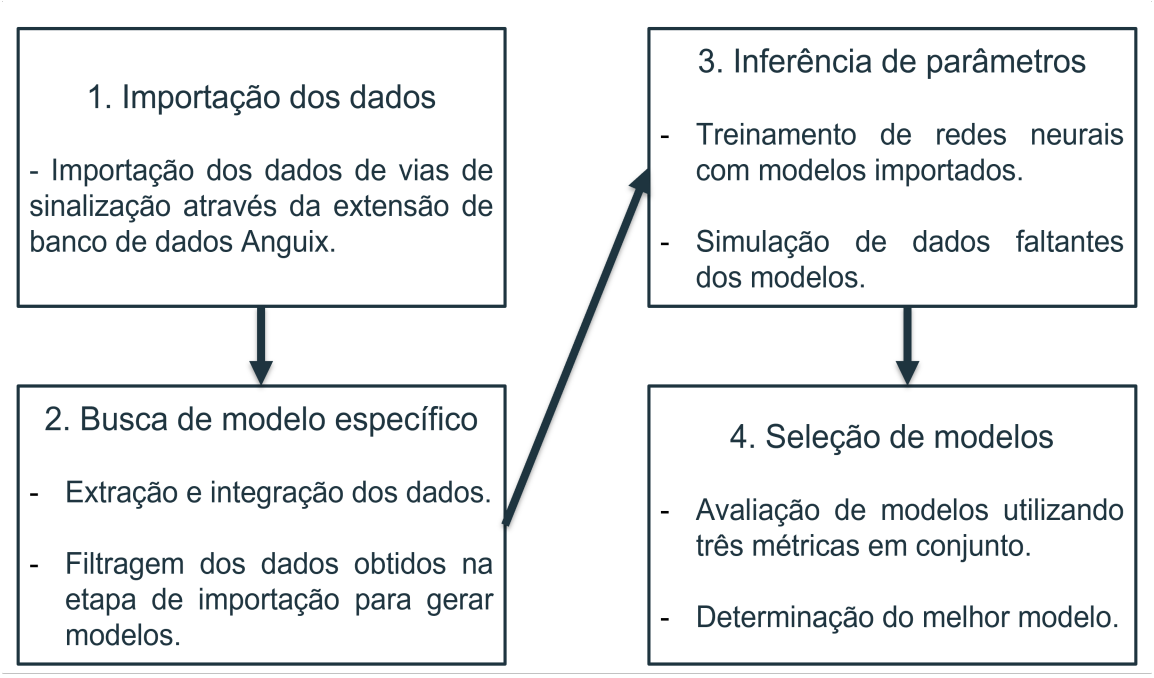

Figura 3.1: *Fluxograma exibindo o fluxo de trabalho do arcabouço desenvolvido neste projeto. A primeira etapa se trata da importação dos dados das vias de sinalização de um organismo através da extensão de banco de dados Anguix; A segunda etapa se trata da filtragem dos dados obtidos na etapa de importação para então selecionar um ou mais modelos específicos através de um subconjunto desses dados importados. A terceira etapa se trata da aplicação dos procedimentos de inferência de parametros para calcular as informações faltantes nos modelos selecionados. Por fim, na quarta etapa são aplicados métodos de seleção de modelos para calcular qual dos modelos gerados possui a maior qualidade.*

intensivos, como análises numéricas e aprendizado de máquina. Por outro lado, Julia foi especialmente projetada para computação científica, apresentando uma sintaxe de fácil compreensão e escrita, além de um desempenho comparável ao de C++. Além disso, Julia oferece suporte integrado para computação paralela e arquiteturas de memória distribuída, o que a torna apropriada para lidar com aplicações que demandam escalabilidade em conjuntos de dados extensos ou ambientes de computação distribuída. A combinação dessas duas linguagens permite que o arcabouço seja altamente eficiente e versátil, sendo capaz de lidar com uma variedade de tarefas com facilidade.

3.2 Bancos de dados suportados e importação

O arcabouço desenvolvido oferece suporte para a análise de modelos de vias de sinalização celular no formato SBML (Systems Biology Markup Language). O SBML é uma linguagem de marcação utilizada na área de biologia de sistemas para representar modelos matemáticos de redes bioquímicas e vias de sinalização. Ele foi desenvolvido como um padrão aberto e de código aberto, com o objetivo de promover a interoperabilidade e a troca de modelos entre diferentes ferramentas e plataformas de análise.

O SBML permite a representação detalhada e precisa das interações moleculares que

ocorrem em uma via de sinalização celular. Ele descreve as espécies químicas envolvidas na via, suas propriedades, como concentrações e localização subcelular, e as interações entre elas, como reações químicas e modificações pós-traducionais. Além disso, o SBML também suporta a inclusão de informações adicionais, como parâmetros cinéticos, condições experimentais e anotações semânticas. Ao utilizar o SBML como formato de modelo, o arcabouço permite que os pesquisadores importem e analisem de forma eficiente e padronizada os modelos de vias de sinalização celular. Essa padronização facilita a integração e a comparação de diferentes modelos, possibilitando uma visão mais abrangente e integrada dos processos moleculares. Além disso, o uso do SBML proporciona flexibilidade aos pesquisadores, permitindo que eles modifiquem e ajustem os modelos conforme necessário para suas análises e investigações específicas. Eles podem incorporar dados experimentais, explorar diferentes condições e cenários, e realizar simulações computacionais para compreender melhor o comportamento das vias de sinalização celular. Um exemplo simples de modelo no formato SBML pode ser observada no código 3.1.

```
<?xml version=”1.0“ encoding=”UTF-8“?>
<sbml xmlns=”http://www.sbml.org/sbml/level3/version2/core“ level=”3“ version=”2“>
  <model id=”simple_model“ name=”Simple Decay Model“>
    <listOfCompartments>
      <compartment id=”cell“ size=”1“ units=”litre“/>
    </listOfCompartments>
    <listOfSpecies>
      <species id=”A“ compartment=”cell“ initialAmount=”10“ substanceUnits=”
      mole“ hasOnlySubstanceUnits=”false“ boundaryCondition=”false“/>
      <species id=”B“ compartment=”cell“ initialAmount=”0“ substanceUnits=”
      mole“ hasOnlySubstanceUnits=”false“ boundaryCondition=”false“/>
    </listOfSpecies>
    <listOfParameters>
      <parameter id=”k1“ name=”ReactionRate“ value=”0.1“ units=”per_second“/>
    </listOfParameters>
    <listOfReactions>
      <reaction id=”decay“ reversible=”false“ fast=”false“>
        <listOfReactants>
          <speciesReference species=”A“ stoichiometry=”1“ constant=”true“/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species=”B“ stoichiometry=”1“ constant=”false“/>
        </listOfProducts>
        <kineticLaw>
          <math xmlns=”http://www.w3.org/1998/Math/MathML“>
            <apply>
              <times/>
              <ci>k1</ci>
              <ci>A</ci>
            </apply>
          </math>
        </kineticLaw>
      </reaction>
    </listOfReactions>
  </model>
</sbml>
```

(3.1)

Neste modelo, temos duas espécies químicas, A e B, localizadas no compartimento ”cell“. A espécie ”A“ decai para a espécie B com uma taxa constante ”k“. Isso é representado por uma reação chamada ”decay“ com uma equação cinética especificada na seção ”<kineticLaw>“. A espécie ”A“ começa com uma quantidade inicial de 10 moles, enquanto a espécie ”B“ começa com 0 moles. O modelo está em um único compartimento chamado ”cell“. Este é um exemplo muito simples e SBML é capaz de representar modelos muito mais complexos.

Em especial, o arcabouço oferece suporte para a importação e análise de modelos provenientes do Anguix, que é uma extensão do banco de dados Reactome. O Anguix disponibiliza informações detalhadas sobre vias de sinalização de diversos organismos, fornecendo uma base sólida para a análise e compreensão desses processos moleculares. Além disso, o arcabouço permite a utilização de modelos em SBML baixados de outras fontes, como o Biomodels (Chelliah *et al.*, 2015), uma plataforma reconhecida internacionalmente que reúne uma vasta coleção de modelos de vias de sinalização celular provenientes de estudos científicos. Além dos modelos provenientes de bancos de dados, o arcabouço também suporta a utilização de modelos gerados manualmente. Essa funcionalidade permite aos pesquisadores desenvolver seus próprios modelos de vias de sinalização celular, customizando as análises de acordo com suas necessidades específicas e hipóteses de pesquisa.

3.2.1 Importação de dados no Anguix

Primeiramente, precisamos iniciar o aplicativo Neo4j (Lal, 2015), que é um poderoso sistema de gerenciamento de bancos de dados em grafos (Chartrand e Zhang, 2012). O Neo4j é projetado especificamente para lidar com dados altamente conectados, como as informações presentes em redes de interações moleculares, nesse caso, as vias de sinalização celular. Em termos mais técnicos, o Neo4j é um gerenciador de bancos de dados baseados no modelo de grafo, onde os dados são representados por meio de nós, relacionamentos e propriedades. Cada nó representa uma entidade, como uma proteína ou um gene, e os relacionamentos entre os nós representam as interações ou conexões entre essas entidades. As propriedades atribuídas aos nós e relacionamentos contêm informações adicionais, como nomes, descrições ou valores quantitativos.

Para importar os dados do banco de dados Anguix para o Neo4j, precisamos utilizar um executável específico, que pode ser obtido em github.com/anthraxodus/Anguix-graphical. Esse executável contém todas as funcionalidades necessárias para importar os dados do Anguix para o Neo4j de maneira fácil e intuitiva. Após realizar o download do executável, é importante seguir as instruções fornecidas no mesmo link para instalar o Neo4j e o banco de dados Reactome. Essas etapas garantem que o ambiente de trabalho esteja configurado corretamente para importar e manipular os dados do Anguix.

Uma vez que o Neo4j e o banco de dados Reactome estejam instalados, podemos iniciar o processo de importação dos dados para o Neo4j. Ao acessar o executável, seremos guiados por uma sequência de telas interativas que nos permitirão selecionar o organismo do qual desejamos importar os dados (por exemplo, *Danio rerio*). A figura 3.2 ilustra visualmente as etapas com as telas envolvidas no processo de importação dos dados do Anguix para o Neo4j, proporcionando uma visão clara e passo a passo de como realizar essa tarefa.

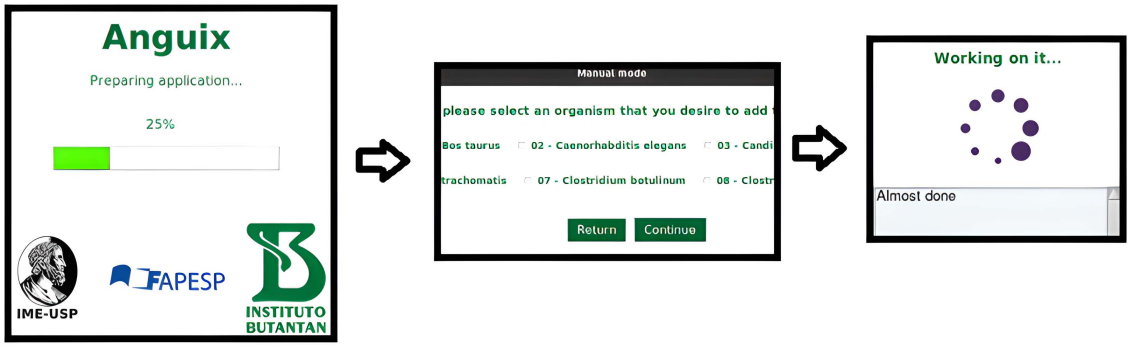

Figura 3.2: *Sequência de interfaces utilizadas para importação de dados através do Anguix.*

Por meio dessa integração entre o Neo4j e o banco de dados Anguix, torna-se possível explorar e visualizar as informações das vias de sinalização celular em formato de grafo. Essa representação gráfica permite uma compreensão mais intuitiva e abrangente das relações e interações entre os componentes das vias de sinalização, facilitando a análise dos dados.

Após a importação bem-sucedida dos dados no Neo4j, temos à nossa disposição uma poderosa ferramenta de consultas por meio da linguagem Cypher (Neo4j Developers, 2021). Essa linguagem foi especialmente desenvolvida pelos criadores do Neo4j para possibilitar a exploração detalhada de bancos de dados de grafo, nos permitindo extrair informações específicas para nossas análises e estudos de vias de sinalização celular.

Através das chamadas ”queries“ no Cypher, que são comandos ou instruções de consulta, podemos explorar uma ampla gama de modelos biológicos que foram importados no Neo4j. As queries possibilitam a especificação de critérios de busca personalizados, nos quais podemos selecionar reações específicas, identificar proteínas-chave envolvidas em processos biológicos ou investigar interações complexas entre componentes da via de sinalização. Essa flexibilidade nas consultas pelo Cypher nos permite analisar de forma minuciosa os dados importados, desvendar padrões e relações que seriam difíceis de serem identificados de outra forma e extrair informações valiosas sobre as complexas redes de interações moleculares que compõem as vias de sinalização celular. um exemplo de query utilizando Cypher em nosso contexto pode ser observado no código 3.2.

```
MATCH (n:SabioRkReaction)-[:kineticDataFor]-(k),
(n)-[:generalReactionFor]-(r), (k)-[:parameterInfo]-> (p)
RETURN n, k, r, p
```
(3.2)

Esse código específico é usado para buscar informações em um banco de dados Neo4j. A consulta começa com a cláusula MATCH, que é usada para identificar padrões no grafo de dados. Neste caso, estamos procurando por nós que atendam a certos critérios. O padrão inclui nós rotulados como ”SabioRkReaction“ conectados a outros nós através de relacionamentos com rótulos específicos, como ”kineticDataFor“ e ”generalReactionFor“. Cada nó correspondente a esses padrões é nomeado usando variáveis, como ”n“, ”k“, ”r“ e ”p“, para que possamos nos referir a eles posteriormente. Finalmente, a cláusula

”RETURN“ especifica quais nós queremos que a consulta retorne como resultado. Neste caso, estamos interessados nos nós ”n“, ”k“, ”r“ e ”p“. Essa consulta é um exemplo prático de como o Cypher é usado para recuperar dados complexos em um banco de dados de grafo, facilitando a pesquisa e análise de informações em sistemas complexos, como redes de sinalização celular.

Após realizar as queries e obter os resultados desejados, o Neo4j permite que os modelos biológicos obtidos sejam baixados para o formato CSV (*Comma-Separated Values*, valores separados por vírgula). O CSV é um formato amplamente utilizado para armazenar dados tabulares de maneira simples e eficiente. Ao baixarmos os modelos nesse formato, podemos facilmente transferi-los para outras ferramentas de análise, como planilhas eletrônicas ou softwares estatísticos, e realizar investigações mais aprofundadas, análises gráficas ou estatísticas complexas.

3.2.2 Aplicação auxiliar para importação

No contexto deste trabalho, uma aplicação auxiliar foi desenvolvida com o objetivo específico de converter os arquivos em formato CSV gerados pelo Neo4j, por meio do Anguix, para o formato SBML. Essa aplicação se mostra de extrema importância, uma vez que o formato SBML é amplamente utilizado e reconhecido pela comunidade de sistemas biológicos como uma representação padrão para modelagem e simulação de vias de sinalização celular.

Ao converter os modelos biológicos obtidos através do Neo4j para o formato SBML, os pesquisadores e cientistas têm acesso a uma representação mais padronizada e universal, o que facilita a comunicação e compartilhamento de modelos entre diferentes laboratórios e instituições. Além disso, o formato SBML permite a utilização de uma ampla variedade de ferramentas e softwares especializados para análise, simulação e visualização de vias de sinalização, fornecendo uma base sólida para estudos mais aprofundados e análises detalhadas.

A aplicação auxiliar foi desenvolvida em linguagem C++ com o objetivo de oferecer uma interface gráfica extremamente simples e amigável aos usuários. A interface consiste em um único botão, que, ao ser clicado, inicia o gerenciador de arquivos do sistema, como mostrado na figura 3.3. Por meio desse gerenciador de arquivos, o usuário pode facilmente buscar e selecionar o arquivo CSV que deseja converter para o formato SBML. Após selecionar o arquivo CSV desejado, a conversão é automaticamente iniciada pela aplicação. Todo o processo de conversão é realizado de forma rápida e eficiente, garantindo a precisão e a integridade dos dados. Uma vez concluída a conversão, o gerenciador de arquivos é aberto novamente, solicitando que o usuário selecione a pasta na qual deseja armazenar o arquivo gerado em formato SBML.

O conversor de CSV para SBML desenvolvido para integrar o arcabouço apresenta uma solução robusta e eficiente para a manipulação e conversão de dados entre os formatos CSV e SBML. Para a implementação dessa ferramenta, foram utilizadas duas bibliotecas essenciais: a LibSBML (Bornstein *et al.*, 2008) e a Csv-Parser (Wickham e Francois, 2020).

A biblioteca LibSBML, uma das principais referências para a manipulação de modelos

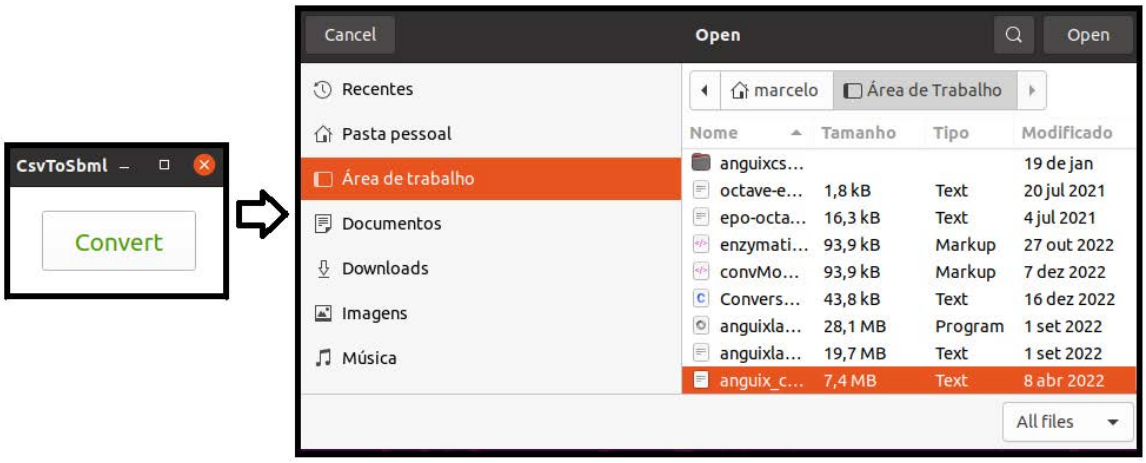

Figura 3.3: *Interfaces da aplicação auxiliar de conversão de arquivos CSV para SBML.*

em formato SBML, desempenhou um papel fundamental no desenvolvimento do conversor. Ela é uma biblioteca em C++ que permite a leitura, escrita e manipulação de modelos biológicos em SBML de forma precisa e eficaz. A LibSBML é amplamente reconhecida na comunidade científica e utilizada em diversos projetos de biologia de sistemas e modelagem computacional. Ao empregar a LibSBML no conversor de CSV para SBML, garantimos que os dados resultantes estejam em conformidade com os padrões do formato SBML, permitindo que o arcabouço possa interagir perfeitamente com outras ferramentas e recursos que também utilizam esse formato.

Por sua vez, a biblioteca Csv-Parser foi escolhida como a ferramenta ideal para a conversão dos dados do formato CSV para o formato SBML. A Csv-Parser é uma biblioteca em C++ projetada especificamente para a leitura e escrita de arquivos CSV, tornando o processo de conversão mais rápido e eficiente. Com sua interface amigável e simples, a Csv-Parser possibilitou a leitura dos dados provenientes do Neo4j através do Anguix, que estavam organizados no formato CSV, e a transformação desses dados em elementos compatíveis com o formato SBML, que é amplamente utilizado na comunidade de sistemas biológicos e oferece uma representação consistente e padronizada de modelos biológicos.

A interação conjunta entre as bibliotecas LibSBML e Csv-Parser permitiu uma integração harmoniosa entre os dois formatos, viabilizando a conversão eficiente dos dados de CSV para SBML e a criação de modelos biológicos consistentes e confiáveis no arcabouço desenvolvido. Através dessa abordagem, os usuários do arcabouço têm a vantagem de trabalhar com dados em formato CSV, que podem ser facilmente obtidos e manipulados através do Neo4j e do Anguix, e, ao mesmo tempo, utilizar esses dados em análises e simulações de modelos em formato SBML, que são mais amplamente aceitos e utilizados pela comunidade científica em estudos de vias de sinalização celular e sistemas biológicos.

O desenvolvimento do conversor de CSV para SBML representa um passo significativo na potencialização do arcabouço, uma vez que amplia suas capacidades de intercâmbio de dados e facilita a colaboração com outros pesquisadores e projetos que utilizam o formato SBML. Além disso, ao utilizar bibliotecas confiáveis e estabelecidas, como a LibSBML e a Csv-Parser, o conversor garante a qualidade e a precisão das conversões realizadas, contribuindo para a confiabilidade dos resultados obtidos através do arcabouço.

Essa simples aplicação proporciona uma abordagem simplificada que torna o processo de conversão acessível mesmo para usuários com pouca experiência técnica ou em ambientes de pesquisa com prazos curtos. A interface gráfica intuitiva elimina a necessidade de comandos complexos e proporciona uma experiência agradável e descomplicada ao usuário. Além disso, o desenvolvimento da aplicação em linguagem C++ oferece vantagens significativas em termos de desempenho e eficiência. Essa linguagem é conhecida por sua capacidade de fornecer um código altamente otimizado, permitindo que a conversão de arquivos ocorra com rapidez e sem perda de qualidade nos dados.

3.3 Biblioteca na linguagem Julia para inferência e seleção de modelos

O arcabouço desenvolvido disponibiliza uma biblioteca em linguagem Julia que permite a realização eficiente e flexível de inferência de parâmetros e seleção de modelos. Além disso, possibilita a importação dos dados de arquivos SBML, que podem ter sido gerados pelo conversor mencionado anteriormente ou provenientes de outras fontes.

A função **”import_sbml(filepath, lvl, version)“**, que pode ser observada no código 3.3, é responsável por importar um modelo SBML (Systems Biology Markup Language) de um arquivo especificado por ”filepath“. Ela utiliza o pacote ”SBMLToolkit“ para verificar a compatibilidade do arquivo SBML e, em seguida, lê o arquivo e o converte em uma estrutura interna de dados que representa o modelo SBML. O parâmetro ”lvl“ indica o nível do SBML (por exemplo, nível 2 ou 3), e o parâmetro ”version“ indica a versão correspondente do SBML (por exemplo, versão 2.4 ou 3.1). O modelo convertido é retornado como saída da função.

```
function import_sbml(filepath, lvl, version)
  SBMLToolkit.checksupport_file(filepath)
  mdl = readSBML(filepath, doc -> begin
    set_level_and_version(lvl, version)(doc)
    convert_simplify_math(doc)
    end)
  return mdl
end
```
(3.3)

No que diz respeito à inferência de parâmetros dos modelos, o arcabouço oferece a opção de utilizar um dos 13 diferentes algoritmos de otimização disponíveis no pacote Optim.jl (Mogensen *et al.*, 2018) na linguagem Julia. Esses algoritmos abrangem uma variedade de técnicas de otimização, cada uma com suas características específicas, tornando possível escolher a mais adequada para resolver diferentes tipos de problemas. Por exemplo, o gradiente descendente (Ruder, 2017) é uma opção simples e amplamente utilizada, porém pode apresentar convergência lenta ou convergir para mínimos locais. Em contraste, o algoritmo Newton-Raphson (Lu *et al.*, 2018) leva em consideração a curvatura da função objetivo, o que pode resultar em uma convergência mais rápida, mas o cálculo da matriz Hessiana pode ser computacionalmente caro em problemas com muitas variáveis. Já o BFGS (Huang *et al.*, 2017) é uma versão mais eficiente do gradiente descendente, aproximando a matriz Hessiana por meio de informações da função objetivo,

o que torna a convergência mais rápida e menos suscetível a mínimos locais. E, por sua vez, o ADAM (Kingma e Ba, 2014) é um algoritmo de otimização baseado em gradiente estocástico, que se destaca em grandes conjuntos de dados e apresenta boa capacidade de generalização.

As principais funções voltadas para a inferência de parâmetros presentes no arcabouço serão detalhadas abaixo.

”define_model(mdl)“: Esta função recebe o modelo SBML convertido como entrada (”mdl“) e retorna informações importantes sobre o modelo. Ela cria um sistema de reações (”rs“) a partir do modelo, converte esse sistema de reações em um sistema de equações diferenciais ordinárias (ODE) (”odesys“), e também obtém vetores que representam as reações (”reactionsvector“) e as espécies (”speciesvector“) presentes no modelo. Além disso, a função cria uma função (”ode_func“) que representa as equações diferenciais que descrevem as dinâmicas do modelo. A função pode ser observada no código 3.4.

```
function define_model(mdl)
  odesys = convert(ODESystem, rs)
  reactionsvector = reactions(rs)
  speciesvector = species(rs)
  ode_func = ODEFunction(odesys)
  return rs, odesys, reactionsvector, speciesvector, ode_func
end
```
(3.4)

”gen_timeseries(tspan, odesys, u0, model_param, method, abstol, reltol, saveat)“: Esta função é responsável por gerar séries temporais (timeseries) do modelo. Ela recebe como entrada um vetor de tempo ”tspan“, o sistema de equações diferenciais ordinárias (”odesys“), as condições iniciais ”u0“, os parâmetros do modelo ”model_param“, o método de solução (”method“) para as equações diferenciais, as tolerâncias ”abstol“ e ”reltol“ para o método de solução, e um parâmetro ”saveat“ que especifica os pontos de tempo onde os resultados serão salvos. A função resolve as equações diferenciais usando o método especificado e retorna as séries temporais resultantes (”X“) e os pontos de tempo correspondentes (”t“). A função pode ser observada no código 3.5.

```
function gen_timeseries(tspan, odesys, u0, model_param, method, abstol, reltol, saveat)
  prob = ODEProblem(odesys, u0, tspan, model_param);
  sol = solve(prob, method, abstol=abstol, reltol=reltol, saveat=saveat);
  X = Array(sol);
  t = sol.t;
  return X, t
end
```
(3.5)

”ude_dynamics!(du, u, p, nn_p, nn_st, t, ode_func, U)“: Esta função representa a dinâmica do modelo de inferência de parâmetros, que combina as equações diferenciais ordinárias (ODEs) do modelo com uma rede neural. Ela recebe como entrada os vetores de derivadas ”du“ e estados ”u“, os parâmetros do modelo ”p“, os parâmetros da rede neural ”nn_p“ e o estado da rede neural ”nn_st“, o tempo ”t“, a função de ODE (”ode_func“) que descreve as equações diferenciais do modelo, e uma função ”U“ que representa a rede

neural. A função calcula as derivadas do modelo e adiciona os efeitos da rede neural nas derivadas, retornando o resultado em du. A função pode ser observada no código 3.6.

```
function ude_dynamics!(du, u, p, nn_p, nn_st, t, ode_func, U)
  ode_func(du, u, p, t) # mechanistic model
  # here we add the neural network to the mechanistic model
  NN = U(u, nn_p, nn_st)[1]
  for i in 1:length(du)
    du[i] += NN[i]
  end
end
```
(3.6)

"predict(θ, prob_nn, method, abstol, reltol, saveat)": Esta função é responsável por fazer a previsão do modelo com base nos parâmetros inferidos ("θ"). Ela recebe como entrada os parâmetros inferidos "θ", um problema de ODE (prob_nn) que representa o modelo com a rede neural incorporada, o método de solução ("method") para as equações diferenciais, as tolerâncias abstol e reltol para o método de solução, e um parâmetro saveat que especifica os pontos de tempo onde os resultados serão salvos. A função resolve as equações diferenciais do modelo com os parâmetros inferidos e retorna as séries temporais resultantes. A função pode ser observada no código 3.7.

```
function predict(θ, prob_nn, method, abstol, reltol, saveat)
  tmp_prob = remake(prob_nn, p=θ)
  tmp_sol = solve(
    tmp_prob, method, abstol=abstol, reltol=reltol, saveat = saveat,
    sensealg = DiffEqFlux.ForwardDiffSensitivity()
  )
  return tmp_sol
end
```
(3.7)

"loss(θ, prob_nn, method, abstol, reltol, saveat, N, X, _step, selected_species)": Esta função representa a função de perda (loss function) usada para avaliar a qualidade da previsão do modelo com base nos parâmetros inferidos. Ela recebe como entrada os parâmetros inferidos "θ", o problema de ODE com a rede neural incorporada ("prob_nn"), o método de solução ("method") para as equações diferenciais, as tolerâncias "abstol" e "reltol" para o método de solução, o parâmetro "saveat" que especifica os pontos de tempo onde os resultados serão salvos, o número total de pontos de tempo N, as séries temporais observadas "X", o passo entre os pontos de tempo "_step", e as espécies selecionadas para a avaliação "selected_species". A função utiliza a previsão do modelo com base nos parâmetros inferidos e calcula a perda média absoluta (MAE) entre as séries temporais observadas e previstas. O objetivo é minimizar essa função de perda durante o processo de inferência de parâmetros para obter os melhores parâmetros que melhor se ajustam aos dados observados. Se a previsão do modelo for bem-sucedida ("sol.retcode == :Success"), a função retorna o MAE calculado. Caso contrário, ela imprime uma mensagem de falha e retorna um valor infinito para a função de perda. A função pode ser observada no código 3.8.

```
function loss(θ, prob_nn, method, abstol, reltol, saveat, N, X, _step,
selected_species)
  sol = FWmodule.predict(θ, prob_nn, method, abstol, reltol, saveat);
  if sol.retcode == :Success
    X̂ = Array(sol);
    return mae(X̂[:, 1:_step:N], X[selected_species, 1:_step:N]);
  end
  println(IJulia.orig_stdout[], ”Failed...“)
  return Inf;
end
```
(3.8)

Com relação à seleção de modelos, o arcabouço adota uma abordagem robusta e abrangente que combina métricas essenciais de avaliação: o Erro Absoluto Médio (MAE) (Willmott, 1981), o Coeficiente de Determinação (Draper e Smith, 1998) e o Critério de Informação Bayesiana (BIC) (Schwarz, 1978a). Essas métricas formam um conjunto de ferramentas valiosas para identificar o modelo mais apropriado e informativo para os dados analisados.

O Erro Absoluto Médio (MAE) é uma métrica que mensura a precisão do ajuste do modelo aos dados observados. Matematicamente, o MAE calcula a média das diferenças absolutas entre os valores previstos pelo modelo e os valores reais observados nos dados. Um MAE menor indica um ajuste mais preciso, onde as diferenças entre as previsões do modelo e os dados reais são menores em magnitude. O MAE é uma métrica intuitiva que fornece uma medida tangível da qualidade da previsão, permitindo uma avaliação direta da acurácia do modelo.

O Coeficiente de Determinação avalia a proporção da variação total nos dados observados que é explicada pelo modelo. Ele varia de 0 a 1, onde um R^2 de 1 indica um ajuste perfeito do modelo aos dados, ou seja, o modelo é capaz de explicar toda a variação observada. Por outro lado, um R^2 próximo a 0 indica que o modelo não é capaz de explicar a variação nos dados. O R^2 é uma métrica que mede a qualidade global do ajuste, oferecendo insights sobre a capacidade do modelo em capturar padrões e tendências nos dados.

O Critério de Informação Bayesiana (BIC) é um critério estatístico que busca equilibrar a qualidade do ajuste aos dados e a complexidade do modelo. Ele é particularmente valioso quando se deseja evitar modelos excessivamente complexos, que podem se ajustar bem aos dados de treinamento, mas podem não generalizar bem para novos dados. O BIC é calculado usando a verossimilhança dos dados e a penalização pela quantidade de parâmetros no modelo. Modelos mais complexos são penalizados, incentivando a escolha de modelos mais parcimoniosos que ofereçam uma explicação eficaz dos dados.

O arcabouço combina essas três métricas em uma função que calcula individualmente o MAE, o Coeficiente de Determinação e o BIC para cada modelo concorrente. Em seguida, essas métricas são usadas para gerar uma pontuação combinada para cada modelo, que é incrementada para cada métrica na qual o modelo foi superior ao seu concorrente. Essa pontuação combinada proporciona uma avaliação abrangente dos modelos, permitindo uma comparação equilibrada entre diferentes dimensões de desempenho. Para isso, o arcabouço conta com funções específicas para calcular cada uma dessas métricas, tais funções serão detalhadas abaixo.

”calculate_mae(simulated, observed)“: A função ”calculate_mae“ tem como objetivo quantificar o desempenho de um modelo ao medir a diferença média absoluta entre as previsões geradas pelo modelo e os valores reais observados nos dados. Essa métrica proporciona uma avaliação direta da acurácia das previsões, permitindo uma compreensão clara de quão bem o modelo se ajusta aos dados observados. Para calcular o MAE, a função recebe dois argumentos: ”simulated“ (valores previstos pelo modelo) e ”observed“ (valores reais observados nos dados). Primeiro, a função calcula a diferença absoluta entre cada par de valores simulados e observados. Em seguida, as diferenças absolutas são somadas e divididas pelo número total de pontos de dados para calcular a média das diferenças absolutas. O resultado é uma medida quantitativa da discrepância média entre as previsões do modelo e os valores reais, permitindo uma avaliação objetiva do ajuste do modelo. A função pode ser observada no código 3.9.

```
function calculate_mae(simulated, observed)
  return sum(abs.(simulated - observed)) / length(observed)
end
```
(3.9)

”calculate_r2(simulated, observed)“: A função ”calculate_r2“ tem como finalidade avaliar o grau em que um modelo consegue explicar a variação nos dados observados. O coeficiente de determinação (R^2) mede a proporção da variação total nos dados que é explicada pelo modelo, variando de 0 a 1. Essa métrica fornece uma avaliação global do ajuste do modelo, indicando a capacidade do modelo em capturar padrões e tendências nos dados. Para calcular o R^2, a função recebe dois argumentos: ”simulated“ (valores previstos pelo modelo) e ”observed“ (valores reais observados nos dados). A função começa calculando a média dos valores observados. Em seguida, ela determina a soma total dos quadrados (”ss_total“) e a soma dos quadrados residuais (”ss_residual“) através da diferença entre os valores observados e as previsões do modelo. A fórmula do R^2 é aplicada, subtraindo o quociente dos quadrados residuais pelo quadrado total da unidade. O resultado é uma medida da proporção da variação explicada pelo modelo, indicando a qualidade do ajuste. A função pode ser observada no código 3.10.

```
function calculate_r2(simulated, observed)
  mean_observed = mean(observed)
  ss_total = sum((observed .- mean_observed).^2)
  ss_residual = sum((observed .- simulated).^2)
  r2 = 1.0 - ss_residual / ss_total
  return abs(r2)
end
```
(3.10)

”calculate_bic(simulated, observed, num_parameters, num_data_points)“: A função ”calculate_bic“ tem como propósito avaliar a adequação de um modelo aos dados, considerando simultaneamente a complexidade do modelo. O Critério de Informação Bayesiana (BIC) é uma métrica que equilibra a qualidade do ajuste pelo número de parâmetros do modelo, favorecendo modelos que oferecem uma explicação eficaz dos dados com menor complexidade. Para calcular o BIC, a função recebe quatro argumentos: ”simulated“ (valores previstos pelo modelo), ”observed“ (valores reais observados nos dados), ”num_parameters“ (número de parâmetros do modelo) e ”num_data_points“ (número de pontos de dados nos dados observados). A função começa calculando o quadrado residual total através da

diferença entre os valores observados e as previsões do modelo. Em seguida, a fórmula do BIC é aplicada, multiplicando o logaritmo do quadrado residual pela quantidade de pontos de dados, e adicionando o logaritmo do número de pontos de dados multiplicado pelo número de parâmetros do modelo. O resultado é uma medida que avalia a qualidade do ajuste e a complexidade do modelo, fornecendo uma perspectiva equilibrada na seleção do melhor modelo. A função pode ser observada no código 3.11.

```
function calculate_bic(simulated, observed, num_parameters, num_data_points)
  n = num_data_points
  k = num_parameters
  residual_sum_of_squares = sum((observed .- simulated).^2)
  bic = n * log(residual_sum_of_squares / n) + k * log(n)
  return bic
end
```
(3.11)

”calculate_metrics(sol_nn1, sol_nn2, X, nn_p, model_param)“: A função ”calculate_metrics“ assume uma posição fundamental no arcabouço, fornecendo uma abordagem sistemática para avaliar e selecionar entre dois modelos SBML. Através da análise de métricas criteriosamente selecionadas e da pontuação combinada dessas métricas, essa função desempenha um papel essencial na determinação do modelo mais apropriado para representar os dados observados. No interior da função, as métricas são calculadas de forma meticulosa para cada um dos modelos SBML considerados. As métricas adotadas são as seguintes: Erro Absoluto Médio (MAE), Coeficiente de Determinação (R^2) e Critério de Informação Bayesiana (BIC). Essas métricas são escolhidas para avaliar a precisão, a capacidade de explicação e a adequação do modelo em relação à sua complexidade.

A função ”calculate_metrics“ recebe diversos argumentos, refletindo a abordagem comparativa. Em particular, ela recebe as soluções simuladas ”sol_nn1“ e ”sol_nn2“, correspondentes a diferentes modelos SBML. Além disso, a função leva em consideração os dados observados ”X“ e os parâmetros dos modelos ”nn_p“ e ”model_param“. Através do cálculo do MAE, o ajuste do modelo é avaliado quanto à discrepância média entre as previsões e os valores reais. O R^2 é utilizado para avaliar a habilidade do modelo em capturar a variação nos dados observados. O BIC, por sua vez, mensura a qualidade do ajuste em relação à complexidade do modelo. É feita então uma comparação dos valores de cada métrica para cada um dos modelos, e aquele que possuir uma pontuação melhor em um número maior de métricas é selecionado pela função como melhor modelo, por exemplo, se o ”modelo 1“ possuir uma pontuação melhor de acordo com as métricas R^2 e MAE, ele vai ser escolhido pela função como sendo superior ao ”modelo 2“ que obteve pontuação melhor apenas segundo a métrica BIC. Este método é utilizado para determinar uma pontuação global que reflete a qualidade do ajuste, a capacidade explicativa e a complexidade do modelo. No caso da métrica R^2, é desejável maximizar a proximidade de seu valor com 1, pois quanto mais próximo de 1 o valor de R^2, melhor o modelo está em capturar a relação entre as variáveis independentes e dependentes. Por outro lado, a minimização do MAE é desejável, pois esta métrica representa o erro médio absoluto das previsões do modelo, ou seja, desejamos que o modelo possua o menor desvio possível em relação aos dados reais. Da mesma forma, o BIC é um critério de seleção de modelos que penaliza modelos mais complexos, favorecendo modelos menos complexos, logo, um valor menor também é desejável. O modelo preferível é então determinado com base nas pontuações combinadas das métricas. Se a pontuação

combinada do primeiro modelo (”combined_score_nn“) for superior, o resultado será ”m1“. Caso contrário, se a pontuação combinada do segundo modelo (”combined_score_nn2“) for superior, o modelo escolhido será o segundo modelo, indicado pelo resultado ”m2“. A função ”calculate_metrics“ pode ser observada nos códigos 3.12, 3.13 e 3.14.

```
function calculate_metrics(sol_nn1, sol_nn2, X, nn_p, model_param)
# Calculate metrics for each SBML model
  mae_nn = calculate_mae(Array(sol_nn1), X)
  mae_nn2 = calculate_mae(Array(sol_nn2), X)
  r2_nn = calculate_r2(Array(sol_nn1), X)
  r2_nn2 = calculate_r2(Array(sol_nn2), X)
  num_parameters_nn = length(nn_p)
  num_parameters_nn2 = length(model_param)
  num_data_points = length(X)
  bic_nn = calculate_bic(Array(sol_nn1), X, num_parameters_nn, num_data_points)
  bic_nn2 = calculate_bic(Array(sol_nn2), X, num_parameters_nn2, num_data_points)
```

(3.12)

```
  # Calculate combined scores
  combined_score_nn = 0
  combined_score_nn2 = 0

  # Compare the results and choose the preferable model based on combined scores
  if mae_nn < mae_nn2
    combined_score_nn = combined_score_nn + 1
  elseif mae_nn > mae_nn2
    combined_score_nn2 = combined_score_nn2 + 1
  else
    combined_score_nn2 = combined_score_nn2 + 1
    combined_score_nn = combined_score_nn + 1
  end
  #
```

(3.13)

```
  if bic_nn < bic_nn2
    combined_score_nn = combined_score_nn + 1
  elseif bic_nn > bic_nn2
    combined_score_nn2 = combined_score_nn2 + 1
  else
    combined_score_nn2 = combined_score_nn2 + 1
    combined_score_nn = combined_score_nn + 1
  end
  #
  if r2_nn > r2_nn2
    combined_score_nn = combined_score_nn + 1
  elseif r2_nn < r2_nn2
    combined_score_nn2 = combined_score_nn2 + 1
  else
    combined_score_nn2 = combined_score_nn2 + 1
    combined_score_nn = combined_score_nn + 1
  end

  #
  if combined_score_nn > combined_score_nn2
    return "m1"
  elseif combined_score_nn2 > combined_score_nn
    return "m2"
  else
    return "Draw"
  end
end
```

(3.14)

A avaliação conjunta do MAE, R^2 e BIC oferece uma visão completa do desempenho dos modelos. O MAE destaca a precisão das previsões, o R^2 avalia a capacidade de capturar padrões e tendências, e o BIC assegura que a complexidade do modelo seja mantida sob controle. Através desse processo, o arcabouço identifica o modelo que melhor se alinha com as necessidades da análise, garantindo previsões confiáveis e interpretações sólidas nos sistemas biológicos.

Todas as funções citadas formam o núcleo do módulo "FWmodule" e desempenham um papel essencial na inferência de parâmetros e na seleção de modelos de vias de sinalização celular no arcabouço. Elas combinam técnicas de equações diferenciais ordinárias, otimização e redes neurais para oferecer uma abordagem poderosa e flexível para a análise de modelos biológicos e o entendimento de complexos processos moleculares que ocorrem nas células. Através do uso dessas funções, os pesquisadores podem realizar análises detalhadas e precisas de modelos biológicos e realizar inferências de parâmetros com base em dados experimentais, contribuindo para avanços significativos na área de biologia de sistemas e vias de sinalização celular.

Vale ressaltar que o arcabouço possui uma abordagem versátil, permitindo que novos algoritmos de seleção sejam incorporados no futuro. Por exemplo, planeja-se incluir métodos de validação cruzada (Kohavi, 1995), que são valiosos para avaliar a capacidade de generalização dos modelos e contribuir para aprimorar ainda mais a seleção do modelo ideal para os dados específicos em questão.

Com essa combinação de recursos poderosos, o arcabouço torna-se uma ferramenta extremamente útil para os pesquisadores no campo de vias de sinalização celular e biologia de sistemas, possibilitando uma abordagem abrangente e precisa na inferência de parâmetros e seleção de modelos para melhor compreensão e interpretação dos complexos processos moleculares que ocorrem nas células. Além disso, a integração das distribuições probabilísticas com os métodos de otimização da linguagem Julia confere ao arcabouço uma capacidade única de realizar análises detalhadas e confiáveis de modelos biológicos em diferentes contextos de pesquisa. Com o contínuo desenvolvimento e aprimoramento do arcabouço, espera-se que ele se torne uma ferramenta cada vez mais essencial e influente na comunidade científica, impulsionando avanços significativos nos estudos de vias de sinalização celular e sistemas biológicos como um todo.

Capítulo 4

Resultados e discussão

Neste capítulo, avançamos na exploração dos resultados obtidos por meio da aplicação de nosso arcabouço, focando na fase de inferência de parâmetros e seleção de modelos. Para tal, utilizamos um ambiente Jupyter em linguagem Julia, que nos proporciona um espaço dinâmico e interativo para exemplificar a aplicabilidade das abordagens propostas.

A base para essa análise é um modelo específico, selecionado a partir de uma gama de possibilidades previamente importadas utilizando a extensão Anguix, representando uma rede complexa de interações em uma via de sinalização celular. O modelo em questão foi submetido ao processo de inferência de parâmetros, onde os valores mais apropriados foram ajustados para que a simulação se aproximasse ao máximo dos dados experimentais. Esse procedimento é de crucial importância para garantir que o modelo esteja calibrado e pronto para representar adequadamente os eventos bioquímicos reais.

Além da inferência de parâmetros, iremos abordar a etapa de seleção de modelos, que ocorre após o ajuste dos parâmetros. Essa etapa visa comparar o desempenho do modelo resultante com outros modelos disponíveis, a fim de identificar qual oferece a melhor concordância com os dados experimentais. Isso é vital para aferir a robustez e validade das conclusões obtidas, bem como para verificar a adequação do modelo em cenários diversos.

Ao longo deste capítulo, conduziremos uma análise detalhada dos resultados, examinando tanto os gráficos gerados quanto as métricas de ajuste utilizadas. Discutiremos as implicações das escolhas feitas, como os algoritmos de otimização e as estratégias de seleção de modelos, e avaliaremos a qualidade do ajuste obtido. Além disso, destacaremos possíveis limitações do arcabouço e considerações a serem feitas para futuros refinamentos.

4.1 Execução explicativa do exemplo utilizando Jupyter notebook

Antes de iniciarmos a execução do exemplo, é crucial garantir que o ambiente Jupyter esteja corretamente configurado e os componentes necessários estejam instalados, para isso, recomendamos seguir o passo a passo presente no **anexo A**. Em seguida, vamos executar

passo a passo no notebook um exemplo demonstrativo. Inicialmente, importaremos os modelos gerados utilizando o Anguix, ”model“, que consiste de um modelo que contém duas reações representadas pelos ID’s 597 e 7489 no banco de dados SabioRK, e ”cutmodel“ que é um recorte deste último modelo contendo apenas a segunda reação, realizaremos então uma simulação simples de suas EDO’s para gerar a série temporal das espécies químicas. O código 4.1 ilustra esse processo.

```
# Gerando séries temporais com uma simulação
tspan = (0.0f0, 25.0f0); method = Rosenbrock23();
X, t = FWmodule.gen_timeseries(tspan, odesys, u0, model_param, ...
...method, abstol = 1e-12, reltol=1e-6, saveat = 0.1;
```
(4.1)

Este trecho de código é responsável por criar séries temporais para o sistema de equações diferenciais ordinárias (EDOs) fornecido, que é representado pela variável ”odesys“. Isso é feito ao resolver numericamente o sistema usando o método ”Rosenbrock23“ (Hairer e Wanner, 1993). O intervalo de tempo da simulação é definido por meio da variável ”tspan“, e as condições iniciais juntamente com os parâmetros do modelo são especificados por ”u0“ e ”model_param“, respectivamente. A resolução do problema de EDOs é realizada internamente pela função ”gen_timeseries“, mencionada no capítulo anterior, que utiliza o método escolhido junto com os parâmetros de tolerância e intervalo de tempo predefinidos. Os resultados obtidos e os pontos temporais correspondentes são armazenados nas variáveis ”X“ e ”t“, respectivamente.

A reação de ID 597 envolve a conversão de Ethanol e NAD+ em Acetaldehyde, NADH e um íon H+. Ethanol, uma molécula de álcool, é oxidado em Acetaldehyde, um composto orgânico aldeído, enquanto a coenzima NAD+ é reduzido a NADH. Esse processo resulta na transferência de hidrogênios do Ethanol para o NAD+, formando NADH, o que representa uma importante etapa na regulação do metabolismo de álcool no organismo. Já a reação de ID 7489 envolve a transformação de NAD+ e S-(Hydroxymethyl)glutathione em NADH, íons H+ e S-Formylglutathione. Nessa reação, o coenzima NAD+ é reduzido a NADH, enquanto S-(Hydroxymethyl)glutathione é convertido em S-Formylglutathione. Ocorre uma transferência de hidrogênios do S-(Hydroxymethyl)glutathione para o NAD+, formando NADH e contribuindo para a manutenção do equilíbrio de oxidação e redução no ambiente celular.

Essas reações são exemplos de processos bioquímicos cruciais que ocorrem nas vias metabólicas celulares. Ao analisar essas reações e modelá-las em um contexto mais amplo, é possível compreender melhor as interações complexas entre os componentes moleculares nas células e como essas reações desempenham um papel fundamental na regulação dos processos biológicos.

Mais adiante no notebook, é feita uma chamada a função ”ude_dynamics¡‘, citada no capítulo anterior, que executa a conversão das equações diferenciais ordinárias (EDOs) do modelo ”cutmodel“ para equações diferenciais universais (UDEs). A função ”ude_dynamics¡‘ atualiza as variáveis diferenciais ”du“ com base nos valores das variáveis de estado ”u“, nos parâmetros do modelo mecanístico ”p“, bem como nos parâmetros do modelo de rede neural ”nn_p“ e ”nn_st“. O modelo mecanístico é avaliado através do uso da função ”ode_func“, sendo que a saída da rede neural é incorporada a cada entrada de ”du“. Para isso, o notebook

cria uma arquitetura de rede neural com quatro camadas, cada uma com dez neurônios, utilizando o pacote Lux Innes *et al.*, 2021, adicionalmente, uma função é definida para combinar o modelo mecanicista com a rede neural, gerando assim uma nova versão do modelo utilizando UDEs. esses processos podem ser observados no código 4.2.

```
  # Definindo as caracteristicas da rede neural
  rng = Random.default_rng()

  U = Lux.Chain(
    Lux.Dense(7, 7, Lux.sigmoid),
    Lux.Dense(7, 7, Lux.sigmoid),
    Lux.Dense(7, 7, Lux.sigmoid),
    Lux.Dense(7, 7)
  )
  nn_p, nn_st = Lux.setup(rng, U);

  # Convertendo para UDE's
  nn_dynamics(du, u, p, t) = FWmodule.ude_dynamics(du, u, model_param2, p, nn_st, t_2, ode_func2, U)
```
(4.2)

Em sequência, é estabelecida uma função de perda para otimizar o desempenho do modelo de rede neural durante o treinamento, e um mecanismo de retorno é criado para registrar a perda após cada iteração; Esse processo pode ser observado no código 4.3.

```
# Gerando parametros das funções de perda e predição
prob_nn = ODEProblem(nn_dynamics!, u0_2, tspan2, nn_p);
method = AutoVern7(Rodas4())
abstol=1e-6; reltol=1e-6; saveat = 0.1;
X_val = view(X_2, :, 1:50)
N = size(X_2)[2]
X_train = view(X_2, :, 51:N)
N_val = size(X_val)[2]
N_train = size(X_train)[2]
_step = 1
val_losses = Float32[];

callback(θ,l) = begin # Retorno para exibição da perda durante o treinamento
  push!(losses, l)
  println(IJulia.orig_stdout[], "Current loss after $(length(losses)) iterations: $(losses[end])")

  # Avalia o modelo no conjunto de validaçãot
  val_loss = FWmodule.loss(θ, prob_nn, method, abstol, reltol, saveat, N_val, X_val, _step, selected_species2)
  push!(val_losses, val_loss)
  println(IJulia.orig_stdout[], "Current validation loss after $(length(val_losses)) iterations: $(val_losses[end])")

  false
end
```
(4.3)

Neste trecho de código, iniciamos estabelecendo os parâmetros para a função de perda e previsão. Utilizamos a função "ODEProblem" para definir "prob_nn", representando o

problema de equações diferenciais ordinárias associado ao modelo de rede neural. A seguir, definimos o método de solução e as tolerâncias de erro para as simulações. A partir dos dados disponíveis, dividimos o conjunto em treinamento e validação. Isso é feito definindo as matrizes ”X_train“ e ”X_val“ para armazenar os conjuntos de treinamento e validação, respectivamente. Também determinamos o número de pontos de dados em cada conjunto por meio das variáveis ”N_train“ e ”N_val“.

O retorno é um elemento crucial nesse contexto, pois nos permite monitorar o progresso do treinamento em tempo real. A função de retorno, definida com o nome callback, desempenha dois papéis importantes. Primeiramente, ela registra as perdas calculadas após cada iteração do treinamento, acrescentando esses valores à matriz losses. Além disso, a função avalia o desempenho do modelo no conjunto de validação, calculando a perda e a registrando em ”val_losses“. Através do mecanismo de retorno, somos capazes de observar a evolução das perdas tanto no treinamento quanto na validação. Ao imprimir esses valores durante o processo, obtemos insights valiosos sobre o ajuste e a eficácia do modelo em diferentes iterações. A utilização de funções de perda e mecanismos de retorno se combina para formar um processo de treinamento eficiente e controlado, permitindo que façamos ajustes conforme necessário para aprimorar o desempenho do modelo de rede neural.

Em seguida, concluindo a fase de inferência de parâmetros, nos deparamos com o trecho de código 4.4, que desempenha um papel essencial ao treinar o modelo da rede neural utilizando o algoritmo de otimização ADAM.

```
 # Treino com algoritmo ADAM
losses = Float32[];
maxiters = 2000;

adtype = Optimization.AutoForwardDiff()
optf = Optimization.OptimizationFunction((x,p)->(
  FWmodule.loss(x, prob_nn, method, abstol, reltol, saveat, N_train, X_train, _step, selected_species2)), adtype)
optprob = Optimization.OptimizationProblem(optf, ComponentVectorFloat64(nn_p))

res1 = Optimization.solve(
  optprob,
  ADAM(0.1),
  maxiters = maxiters,
  callback = callback,
  progress = true
)
loss_adam_end = size(losses)[1];
println(”Training loss after $(length(losses)) iterations: $(losses[end])“);
lossval_adam_end = size(val_losses)[1]
println(”Validation loss after $(length(losses)) iterations: $(losses[end])“);
```

(4.4)

Neste código, iniciamos pela definição da matriz ”losses“, que irá conter as perdas calculadas em cada iteração do treinamento. O número máximo de iterações ”maxiters“ é definido, representando o limite para a quantidade de vezes que o algoritmo de otimização será aplicado. A escolha da abordagem de diferenciação automática é feita por meio

da variável ”adtype“, especificamente ”Optimization.AutoForwardDiff()“. A função de otimização ”optf“ é definida com base na função de perda, que é adaptada para receber os parâmetros do modelo de rede neural, utilizando os dados de treinamento ”X_train“ e demais variáveis necessárias. O problema de otimização é estabelecido por meio de ”optprob“, utilizando os parâmetros do modelo de rede neural como variáveis a serem otimizadas.

A seguir, o algoritmo ADAM é aplicado por meio da função ”Optimization.solve“, onde o problema de otimização ”optprob“ é resolvido utilizando o algoritmo ADAM com uma taxa de aprendizado de 0.1. O número máximo de iterações é determinado pelo valor de ”maxiters“, e a função de retorno ”callback“ é utilizada para acompanhar o progresso do treinamento. Durante esse processo, as perdas calculadas são registradas na matriz ”losses“, permitindo que a evolução do desempenho do modelo ao longo das iterações seja acompanhada. Ao final do treinamento, os valores da perda de treinamento e validação são impressos para avaliação. Esse bloco de código encapsula a etapa crítica de treinamento da rede neural, utilizando o algoritmo ADAM para otimizar os parâmetros do modelo. Essa abordagem iterativa e controlada permite ajustar gradualmente o modelo para se aproximar melhor dos dados experimentais, resultando em um modelo de rede neural ajustado e mais eficaz.

Em seguida, procedemos à visualização das perdas por meio do código 4.5, Nesse bloco de código, inicialmente definimos o tamanho padrão do gráfico utilizando ”default(size = (900, 500))“. Em seguida, utilizamos a função ”plot“ para criar um gráfico que exibe as perdas da otimização utilizando o algoritmo ADAM. O eixo y é configurado para escala logarítmica com ”yaxis = :log10“, e o eixo x também é ajustado para escala logarítmica com ”xaxis = :log10“.

```
# Plot o gráfico de perdas (losses)

default(size = (900, 500));
plot(
  1:loss_adam_end, losses[1:loss_adam_end],
  yaxis = :log10, xaxis = :log10,
  label = ”ADAM“, color = :blue,
  title = ”Loss Function“
);
```

(4.5)

```
plot!(
  1:lossval_adam_end, val_losses[1:lossval_adam_end],
  yaxis = :log10, xaxis = :log10,
  label = ”Validation“, color = :red,
  title = ”Loss Function“
);

xlabel!(”Iterations“)
ylabel!(”Loss“)
```

No gráfico, que pode ser observado na figura 4.1 plotamos as perdas de treinamento obtidas com o algoritmo ADAM, representadas pela linha azul com a legenda ”ADAM“. O título do gráfico é definido como ”Função de Perda“. Em seguida, utilizamos a função ”plot¡‘

para adicionar ao mesmo gráfico as perdas de validação, representadas pela linha vermelha com a legenda ”Validação“. Essas duas séries de perdas são apresentadas em conjunto, permitindo a comparação entre o desempenho de treinamento e validação. As legendas das séries de dados são definidas com o uso do argumento ”label“, e as cores são configuradas com o argumento ”color“. Os eixos x e y são rotulados com as palavras ”Iterações“ e ”Perda“, respectivamente, utilizando as funções ”xlabel“ e ”ylabel“. Dessa forma, o gráfico gerado proporciona uma representação visual da evolução das perdas de treinamento e validação ao longo das iterações do algoritmo de otimização ADAM, fornecendo uma avaliação visual do progresso do treinamento e da capacidade do modelo ajustado em se adequar aos dados experimentais.

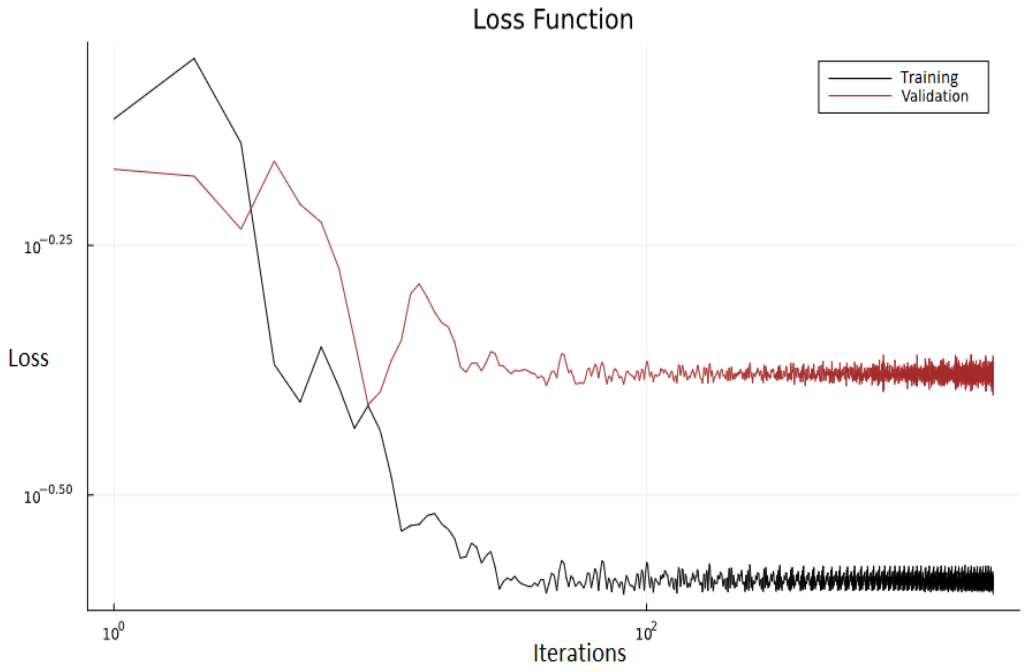

Figura 4.1: *Evolução da função de perda durante o treinamento com algoritmo ADAM. A linha preta representa o valor da função de perda no conjunto de treinamento enquanto a linha marrom representa o valor da função de perda no conjunto de validação.*

Prosseguindo, realizamos a representação gráfica do modelo treinado, conforme demonstrado no código 4.6. Nesse trecho de código, inicialmente é calculada a previsão do modelo treinado utilizando os parâmetros otimizados pelo algoritmo ADAM, armazenados na variável ”tp“. A função ”FWmodule.predict“ é utilizada para gerar as previsões do modelo treinado com base nos parâmetros otimizados.

```
# Plota o modelo treinado
tp = res1.minimizer;
X3 = Array(FWmodule.predict(tp, prob_nn, method, abstol, reltol, saveat));

default(size = (900, 500))
plot(
  t,
  X'[:, selected_species],
  title = "ADAM results",
  alpha=0.75,
  labels=permutedims(speciesvector),
  ls=permutedims([:solid,:solid,:solid,:solid,:solid,:solid,:solid,:solid,:solid,:solid]),
  color=permutedims([:red, :blue, :brown, :purple, :yellow, :pink, :green, :orange, :grey, :black])
)                                                                                                    (4.6)

xlabel!("t")
ylabel!("concentration")
speciesvector2_ = [string(s)*"'" for s in speciesvector2]
plot!(
  t,
  X3',
  alpha=0.75,
  labels=permutedims(speciesvector2_),
  ls=:dash,
  color=permutedims([:red, :blue, :brown, :purple, :yellow, :pink, :green, :orange, :grey, :black]),
  left_margin=5Plots.mm
)
```

Em seguida, é criado um gráfico que compara as concentrações simuladas pelo modelo original (linha sólida) e pelo modelo treinado (linha tracejada) ao longo do tempo. O eixo x representa o tempo ("t"), enquanto o eixo y representa as concentrações das espécies químicas selecionadas. As concentrações simuladas pelo modelo original são plotadas em linhas sólidas com cores diferentes, definidas pelo argumento "color". As concentrações simuladas pelo modelo treinado são plotadas em linhas tracejadas, com as mesmas cores do modelo original.

Além disso, os rótulos das espécies químicas selecionadas são definidos como legendas no gráfico, permitindo identificar facilmente cada linha. O título do gráfico é definido como "Resultados ADAM". Os eixos x e y são rotulados com "t" (tempo) e "concentração", respectivamente, utilizando as funções "xlabel¡' e "ylabel¡'. O gráfico gerado, que pode ser observado na Figura 4.2 proporciona uma comparação visual entre as concentrações simuladas pelo modelo original e pelo modelo treinado, permitindo avaliar a qualidade do ajuste do modelo treinado aos dados experimentais e verificar se ele consegue representar adequadamente as dinâmicas bioquímicas observadas.

Em seguida, finalizamos esta etapa aplicando um procedimento simples de seleção de modelos, conforme evidenciado no código 4.7. Neste ponto, estamos conduzindo uma comparação entre o modelo cujos parâmetros foram inferidos e otimizados usando o algoritmo ADAM, com outro modelo obtido da mesma maneira, porém usando o algoritmo Gradiente

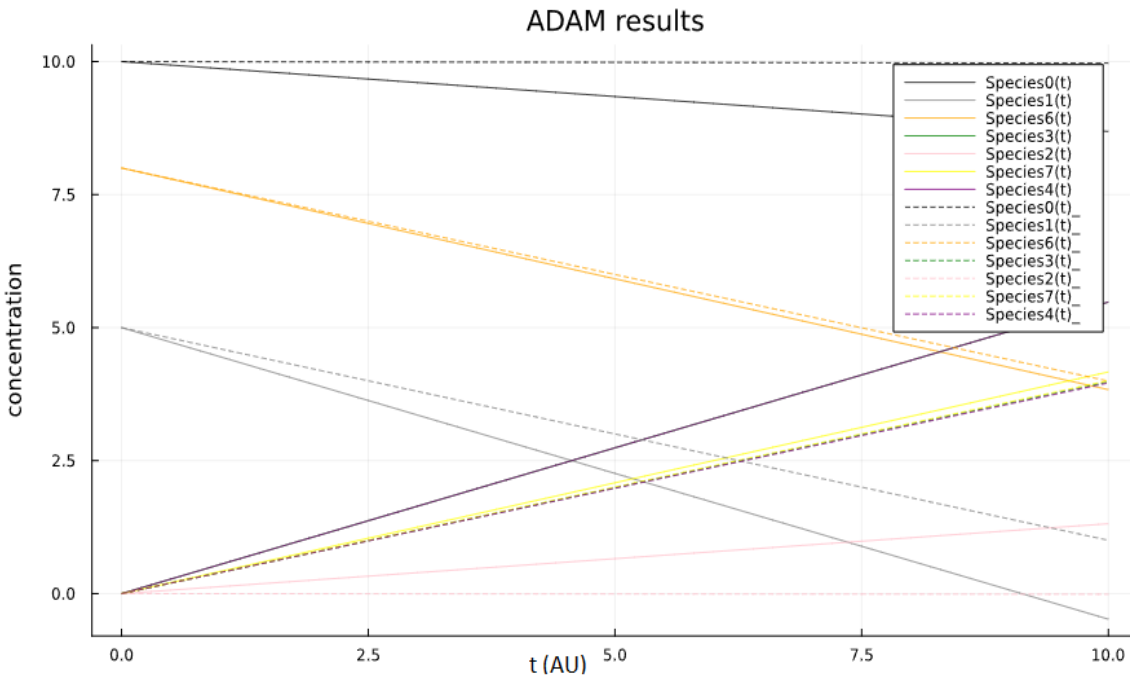

Figura 4.2: *Gráfico comparando os resultados simulados utilizando o algoritmo ADAM com as concentrações do modelo original. As linhas sólidas representam os valores das concentrações das espécies no modelo original ao longo do tempo, enquanto as linhas tracejadas representam os valores do modelo predito utilizando o algoritmo ADAM.*

Descendente Estocástico (SGD), o que pode ser alcançado simplesmente substituindo ”ADAM(0.1)“ por ”SGD(learning_rate=0.1)“ no código 4.4 . Este processo de seleção busca elucidar a eficácia de diferentes algoritmos de otimização na obtenção dos parâmetros ideais para o modelo.

```
# Calcula soluções numericamente
sol_nn = solve(prob_nn, method2, abstol=abstol2, reltol=reltol2, saveat=saveat2)
sol_sgd_nn = solve(prob_sgd_nn, method2, abstol=abstol2, reltol=reltol2, saveat=saveat2)

# Calcular metricas para o modelo treinado com a rede neural e o modelo treinado com SGD-NN
best_model = FWmodule.calculate_metrics(sol_nn, sol_sgd_nn, X_2, nn_p, model_param)

# Exibir o modelo selecionado
best_model
```
(4.7)

Nesse trecho, primeiramente, calculamos numericamente a solução ”sol_nn“, que é o resultado da resolução do problema usando o algoritmo de otimização ADAM. Em seguida, calculamos numericamente a solução ”sol_sgd_nn“, que é obtida usando o algoritmo SGD (Gradiente descendente estocástico) para otimizar os parâmetros do modelo de redes neurais. Por fim, aplicamos a função ”calculate_metrics“ para avaliar as métricas de desempenho entre os dois modelos: o treinado com redes neurais utilizando o algoritmo ADAM e o treinado com redes neurais utilizando o algoritmo SGD. O modelo que apresentar os melhores resultados em relação às métricas será exibido como o ”melhor modelo“. Isso

nos permite avaliar como diferentes algoritmos de otimização afetam o desempenho do modelo em questão; Nesse exemplo, o modelo com melhores resultados foi o treinado com o algoritmo ADAM, na seção a seguir será feita uma comparação dos resultados com mais detalhes.

4.2 Análise dos resultados obtidos

Nesta seção, daremos início à análise dos resultados obtidos por meio da comparação direta entre os modelos gerados com os algoritmos ADAM e SGD. Para isso, primeiramente, vamos examinar os gráficos de resultados e as curvas de perda associadas a esses modelos. Essa comparação inicial nos permitirá identificar possíveis diferenças de desempenho e entender como cada algoritmo de otimização influencia o ajuste do modelo aos dados experimentais. Vamos explorar tanto as semelhanças quanto as disparidades encontradas nos resultados, fornecendo uma visão abrangente do impacto da escolha do algoritmo de otimização na inferência de parâmetros e na qualidade do modelo resultante.

Após a comparação inicial entre os modelos ADAM e SGD, avançaremos para uma análise mais detalhada dos resultados. Vamos explorar os resultados da função ”calculate_metrics“, que internamente avalia os modelos com três diferentes métodos. Examinaremos individualmente os resultados obtidos por cada um desses métodos, revelando suas métricas e avaliando como contribuíram para a seleção do modelo final. Essa exploração interna nos permitirá compreender as nuances que levaram à escolha do modelo mais adequado, destacando a importância de cada método de avaliação. Dessa forma, garantiremos uma análise completa e transparente dos resultados, demonstrando o rigor do processo de seleção de modelos.

4.2.1 Comparação Gráfica dos modelos gerados

Primeiramente, enquanto examinamos os resultados, nossa atenção se volta para os gráficos de séries temporais que comparam a evolução dos valores de cada espécie química entre o modelo gerado com o algoritmo SGD e o modelo gerado com o algoritmo ADAM. Uma observação notável é que a diferença na predição dos valores de cada espécie química é mínima ao comparar o gráfico do modelo gerado com o algoritmo SGD com o gráfico do modelo gerado com o algoritmo ADAM, sendo essa discrepância quase indistinguível visualmente, como pode ser observado na figura 4.3. Isso se deve à natureza do problema de otimização subjacente. Ambos os algoritmos, ADAM e SGD, convergem eficazmente para um mínimo global ou local, indicando que ambas as abordagens são eficazes na adaptação do modelo aos dados observados. Além disso, a arquitetura da rede neural subjacente e os parâmetros do modelo permanecem essencialmente os mesmos, garantindo a uniformidade das previsões resultantes. Portanto, é natural que as previsões sejam altamente semelhantes.

Agora iremos comparar os valores reais das espécies químicas [0(t), 1(t), 2(t), 3(t), 4(t), 6(t), 7(t)] nos instantes 1, 5 e 10 com os valores preditos pelos modelos gerados com os algoritmos ADAM e SGD para essas espécies, os valores podem ser observados nas tabelas 4.1, 4.2, 4.3, 4.4, 4.5, 4.6 e 4.7. Esta análise visa avaliar quão bem os modelos treinados com

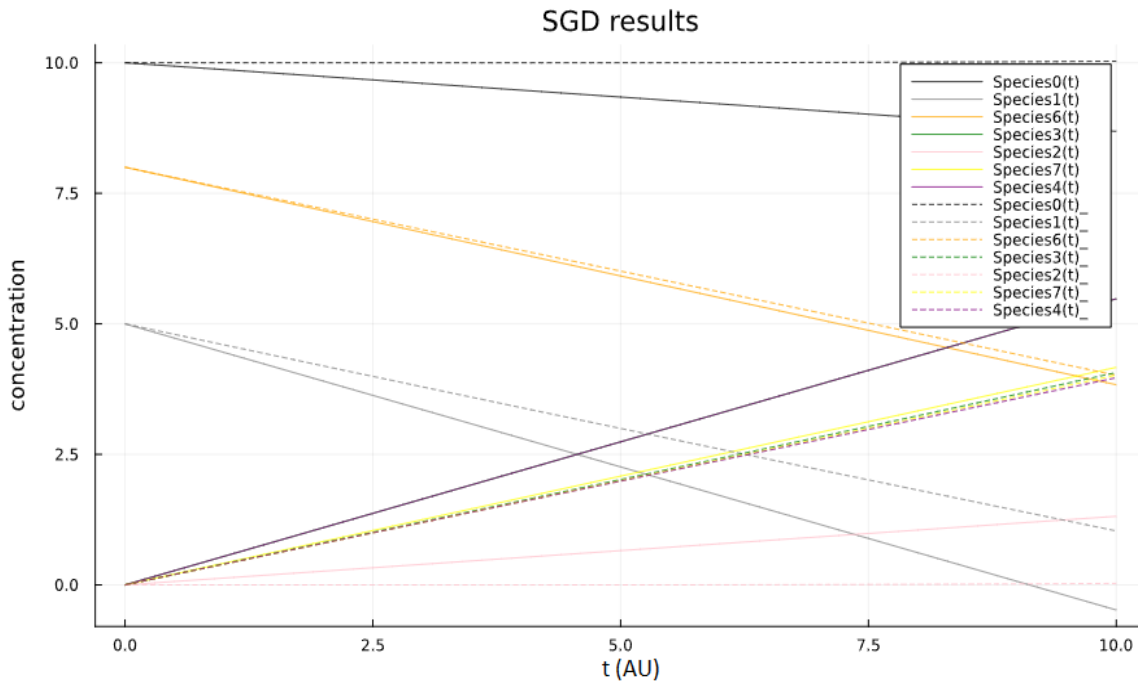

Figura 4.3: *Gráfico comparando os resultados simulados utilizando o algoritmo SGD com as concentrações do modelo original. As linhas sólidas representam os valores das concentrações das espécies no modelo original ao longo do tempo, enquanto as linhas tracejadas representam os valores do modelo predito utilizando o algoritmo SGD.*

ADAM e SGD conseguem prever os valores das espécies químicas em comparação com os resultados reais.

Instante	Real	ADAM	SGD
1	9.86862	10.00001	10.00007
5	9.34313	10.00009	10.00011
10	8.68627	10.00018	10.01133

Tabela 4.1: *Comparação dos valores da espécie 0(t) nos modelos.*

Instante	Real	ADAM	SGD
1	4.45196	4.59891	4.59712
5	2.25980	2.99459	2.97926
10	-0.48039	0.98919	0.95254

Tabela 4.2: *Comparação dos valores da espécie 1(t) nos modelos.*

Instante	Real	ADAM	SGD
1	0.13137	-2.43432	0.00033
5	0.65686	-0.00012	-0.00024
10	1.31372	-0.00024	0.01005

Tabela 4.3: *Comparação dos valores da espécie 2(t) nos modelos.*

Instante	Real	ADAM	SGD
1	0.54803	0.39931	0.40366
5	2.74019	1.99656	2.02026
10	5.48039	3.99312	4.05631

Tabela 4.4: *Comparação dos valores da espécie 3(t) nos modelos.*

Instante	Real	ADAM	SGD
1	0.54803	0.39908	0.39927
5	2.74019	1.99541	1.99351
10	5.48039	3.99082	3.97419

Tabela 4.5: *Comparação dos valores da espécie 4(t) nos modelos.*

Instante	Real	ADAM	SGD
1	7.58333	7.59865	7.60272
5	5.91666	5.99325	6.00741
10	3.83333	3.98651	4.00351

Tabela 4.6: *Comparação dos valores da espécie 6(t) nos modelos.*

Instante	Real	ADAM	SGD
1	0.41666	0.40031	0.40481
5	2.08333	2.00155	2.02245
10	4.16666	4.00310	4.04151

Tabela 4.7: *Comparação dos valores da espécie 7(t) nos modelos.*

Esses resultados indicam que, nos instantes 1, 5 e 10, os valores tanto do modelo predito com ADAM quanto do modelo predito com SGD estão muito próximos dos valores originais das espécies. As diferenças percentuais também são bastante baixas nos casos em que os valores são suficientemente grandes para que faça sentido utilizar essa métrica, sugerindo que ambos os modelos são capazes de reproduzir com precisão os valores observados.

Essa consistência notável nas previsões é um indicativo da robustez dos nossos métodos com relação à escolha do algoritmo de otimização. Isso é particularmente relevante, pois sugere que o modelo gerado é resiliente a variações nos métodos de treinamento e pode fornecer previsões confiáveis independentemente do algoritmo utilizado. Essa estabilidade é um fator crucial em aplicações práticas, onde a escolha do algoritmo de otimização pode depender de considerações computacionais ou de outros fatores práticos. Em suma, nossa análise inicial dos gráficos de resultados confirma que tanto o modelo gerado com o algoritmo ADAM quanto o modelo gerado com o algoritmo SGD produzem resultados altamente comparáveis e precisos.

Avançando para a análise dos gráficos de perda (Loss) dos modelos gerados com os algoritmos ADAM e SGD, observamos diferenças notáveis em seus comportamentos. O gráfico de Loss do modelo gerado com o ADAM exibe uma tendência de considerável variação ao longo do treinamento, como pode ser observado na figura 4.1. Isso se manifesta tanto no conjunto de testes quanto no de validação, onde a cada ponto no gráfico o valor parece hora subir e hora descer, porém, de maneira geral, há uma tendência decrescente. No entanto, essa trajetória é caracterizada por flutuações consideráveis, o que reflete uma sensibilidade do modelo a pequenas variações nos dados de entrada.

Por outro lado, o gráfico de Loss do modelo gerado com o SGD apresenta um padrão de comportamento diferente. Nele, as linhas que representam a Loss no conjunto de testes e validação são mais constantes, exibindo uma tendência mais consistente de declínio à medida que o treinamento avança, como pode ser observado na figura 4.4. Essa estabilidade sugere que o modelo do SGD é menos suscetível a oscilações abruptas durante o treinamento, o que pode ser benéfico para a convergência da otimização.

Essa diferença nos gráficos de Loss entre os dois algoritmos evidencia a influência do método de otimização na estabilidade do modelo. Enquanto o ADAM pode ser mais rápido para se adaptar a mudanças nos dados, ele também pode ser mais propenso a flutuações. Por outro lado, o SGD parece oferecer uma convergência mais gradual e previsível. Essa análise é crucial para compreendermos a performance desses modelos e selecionarmos aquele que melhor atende aos nossos objetivos de modelagem. No próximo estágio da análise, examinaremos os resultados da função ”calculate_metrics“ para uma visão mais detalhada das diferenças entre esses modelos.

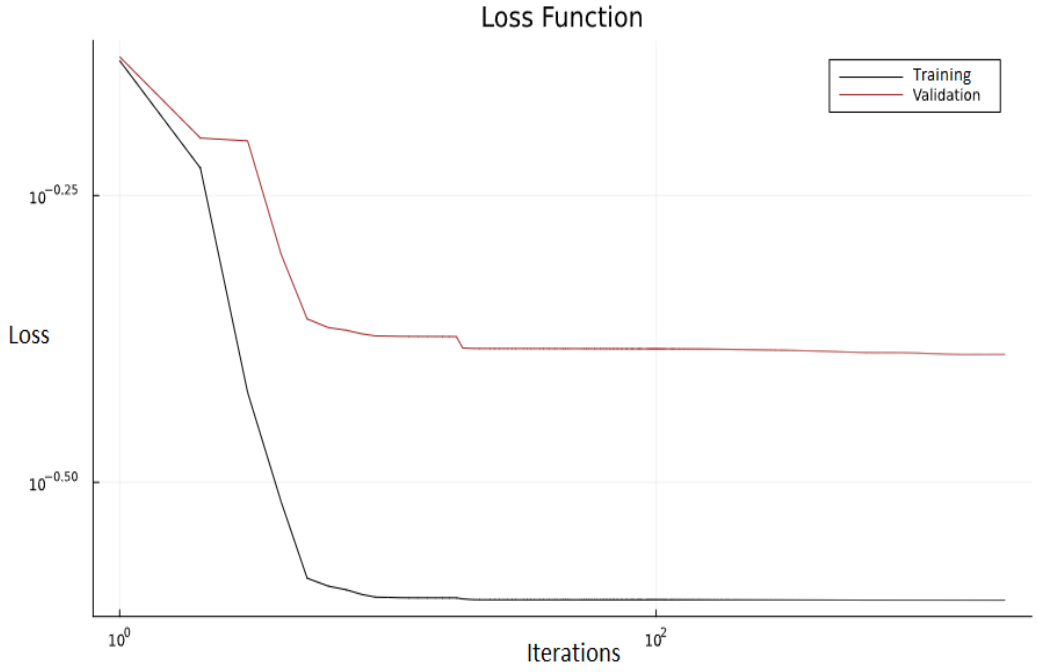

Figura 4.4: *Evolução da função de perda durante o treinamento com algoritmo SGD. A linha preta representa o valor da função de perda no conjunto de treinamento enquanto a linha marrom representa o valor da função de perda no conjunto de validação.*

4.2.2 Análise dos resultados da seleção de modelos

A análise dos resultados da função ”calculate_metrics“, cujo objetivo é realizar uma seleção de modelos, revelou que o modelo treinado com o algoritmo SGD emergiu como o modelo preferível em relação ao modelo que utilizou o algoritmo ADAM. Essa seleção foi baseada em uma avaliação abrangente das métricas MAE, R^2 e BIC, que forneceram informações muito importantes sobre o desempenho e a complexidade de ambos os modelos.

Primeiramente, ao examinarmos o R^2, uma métrica que quantifica a capacidade de explicação do modelo, observamos que o modelo treinado com o algoritmo SGD alcançou um valor notável de 0.6823, enquanto o modelo treinado com o algoritmo ADAM registrou um valor bastante inferior de 0.2614. Um R^2 mais próximo de 1 indica que o modelo é altamente capaz de explicar a variação nos dados observados, o que demonstra a habilidade superior do modelo treinado com SGD em capturar a relação entre as variáveis independentes e dependentes.

Além disso, a métrica MAE (Erro Médio Absoluto) foi empregada para medir a precisão dos modelos na previsão dos resultados. O modelo treinado com o algoritmo ADAM obteve um MAE de 1.8974, enquanto o modelo treinado com o algoritmo SGD apresentou um MAE de 1.3486. Nesse caso, uma pontuação menor é desejável, indicando menor discrepância entre as previsões do modelo e os valores reais. Portanto, o modelo treinado com SGD também demonstrou um desempenho superior em relação ao MAE.

Por fim, consideramos o BIC (Critério de Informação Bayesiano), que avalia a complexidade do modelo. O modelo treinado com o algoritmo SGD registrou um valor de BIC

de 1040.7880, enquanto o modelo treinado com o algoritmo ADAM obteve um BIC de 1624.0984. O BIC é projetado para favorecer modelos menos complexos, e uma pontuação menor indica uma complexidade reduzida. Nesse contexto, o modelo treinado com SGD apresentou uma complexidade inferior, tornando-o mais eficiente do ponto de vista do BIC.

Ao consolidar essas métricas, utilizamos uma função que adiciona uma pontuação para cada modelo baseada em cada métrica na qual esse modelo é superior ao seu concorrente. Essa abordagem nos conduz à escolha do modelo treinado com o algoritmo SGD, que seguindo essa função nesse exemplo, emerge como a opção preferencial em relação ao modelo treinado com o algoritmo ADAM, considerando todos os três métodos R^2, MAE e BIC. Dessa forma, garantimos uma métrica conjunta que atende de forma ideal aos nossos critérios de qualidade do modelo.

Capítulo 5

Conclusões e implicações

A pesquisa apresentada nesta dissertação foi conduzida com o objetivo de desenvolver um arcabouço eficiente e abrangente para a inferência de parâmetros e seleção de modelos em vias de sinalização celular. Ao longo deste estudo, abordamos a complexidade inerente à modelagem dinâmica de vias de sinalização, especialmente quando se lida com a comunicação entre espécies químicas e a necessidade de inferir parâmetros em modelos baseados em equações diferenciais universais (UDEs). Para atingir esse objetivo, utilizamos uma combinação de linguagens de programação, Julia e C++, para criar uma biblioteca e uma aplicação que permitem uma modelagem mais precisa e eficiente.

Os resultados deste estudo têm implicações significativas para a área de pesquisa em biologia celular e modelagem matemática. Primeiramente, a abordagem desenvolvida oferece uma solução simplificada e unificada para a modelagem de vias de sinalização, permitindo que pesquisadores economizem tempo e evitem incorreções durante o processo de modelagem. Essa abordagem é especialmente valiosa quando se trata de lidar com a complexidade das redes de sinalização celular, onde a desconexão de espécies químicas do restante da rede pode ser problemática.

Além disso, ao fornecer uma ferramenta que integra a importação de dados de repositórios públicos de reações bioquímicas, a construção de modelos baseados em UDEs, a inferência de parâmetros e a seleção de modelos, estamos contribuindo para a expansão do conhecimento científico na área da biologia de sistemas. Isso pode acelerar a descoberta de novos insights sobre o funcionamento das vias de sinalização celular e seu papel em processos biológicos complexos, como o câncer.

Com base nos resultados obtidos e nas análises realizadas, concluímos que nosso arcabouço representa uma abordagem promissora e eficaz para a modelagem de vias de sinalização celular. A capacidade de integrar dados de repositórios bioquímicos, realizar modelagem baseada em UDEs e realizar inferência de parâmetros e seleção de modelos é uma vantagem significativa para a comunidade de pesquisa em biologia de sistemas.

Durante a execução de nosso exemplo utilizando o Jupyter notebook, a seleção do modelo treinado com o algoritmo SGD, como preferencial em relação ao modelo treinado com o algoritmo ADAM, ilustra como nosso arcabouço pode identificar modelos mais precisos e menos complexos. A combinação das métricas R2, MAE e BIC demonstra ser

um método eficiente para escolher modelos que se destacam em termos de precisão e simplicidade.

Em última análise, a pesquisa realizada neste trabalho contribui para avançar nosso entendimento das vias de sinalização celular e fornece uma ferramenta valiosa para futuras investigações. A capacidade de integrar efetivamente dados experimentais, modelagem matemática e seleção de modelos simplifica o processo de compreensão dos mecanismos subjacentes das vias de sinalização, oferecendo uma ferramenta poderosa para pesquisadores e cientistas interessados em explorar as complexidades da biologia celular.

5.1 Limitações

Embora este arcabouço apresente uma abordagem promissora para a modelagem de vias de sinalização celular, é importante reconhecer suas limitações. Primeiramente, a precisão dos modelos gerados depende diretamente da qualidade dos dados experimentais utilizados. Ruídos ou imprecisões nos dados de entrada podem afetar a capacidade do arcabouço de inferir parâmetros com precisão. Portanto, é essencial que os experimentos sejam conduzidos rigorosamente e que os dados sejam pré-processados adequadamente para minimizar essas incertezas.

Outra limitação a ser considerada é a complexidade das vias de sinalização celular. Embora o arcabouço busque simplificar o processo de modelagem, as vias reais muitas vezes envolvem uma interconexão complexa de componentes. Em alguns casos, pode ser desafiador selecionar quais espécies químicas e reações incluir no modelo, o que pode afetar a capacidade do arcabouço de representar com precisão o sistema biológico real.

Para futuros trabalhos, é possível considerar várias melhorias e refinamentos no arcabouço. Uma área de aprimoramento potencial é a incorporação de métodos avançados de pré-processamento de dados, como técnicas de redução de dimensionalidade e detecção de outliers, para lidar com dados experimentais ruidosos de forma mais eficaz.

Além disso, o arcabouço pode ser aprimorado para incorporar diversas fontes adicionais de dados experimentais e implementar uma variedade de algoritmos de otimização e seleção de modelos. Essas melhorias proporcionariam uma flexibilidade expandida para lidar com uma gama diversificada de dados experimentais e otimizar os parâmetros dos modelos de maneira mais adaptável. A exploração de algoritmos de otimização mais rápidos, eficazes e distribuídos pode aprimorar consideravelmente a eficiência do processo de inferência de parâmetros, especialmente quando tratamos de conjuntos de dados volumosos. A integração dessas funcionalidades pode enriquecer a capacidade do arcabouço para atender a uma ampla variedade de cenários de pesquisa em biologia de sistemas.

Em última análise, o arcabouço apresentado neste trabalho serve como um ponto de partida valioso para a pesquisa em modelagem de vias de sinalização celular. Ao abordar suas limitações e continuar a buscar melhorias, podemos ampliar seu potencial e contribuir ainda mais para a compreensão dos processos biológicos subjacentes.

5.2 Contribuições deste trabalho

Este trabalho de pesquisa oferece várias contribuições significativas para o campo da modelagem de vias de sinalização celular e a inferência de parâmetros em sistemas biológicos. As principais contribuições incluem:

- **Desenvolvimento de um Arcabouço Integrado:** Este projeto resultou no desenvolvimento de um arcabouço abrangente para a inferência de parâmetros e seleção de modelos de vias de sinalização celular. Esse arcabouço não apenas fornece uma solução para a modelagem de vias de sinalização, mas também incorpora métodos avançados de aprendizado de máquina para melhorar a precisão e a eficiência do processo.
- **Integração de Dados:** Um aspecto fundamental deste trabalho é a integração de dados de reações bioquímicas de repositórios públicos. Essa integração de dados de alta qualidade aumenta a confiabilidade das simulações e modelagens resultantes, tornando o arcabouço uma ferramenta valiosa para pesquisadores em biologia computacional e sistemas biológicos.
- **Ampla Aplicabilidade:** O arcabouço desenvolvido é altamente flexível e pode ser aplicado a uma ampla variedade de vias de sinalização celular e sistemas biológicos. Ele oferece suporte a diferentes tipos de algoritmos de inferência e pode ser facilmente personalizado para atender às necessidades específicas de modelagem de cada pesquisa.
- **Potencial para Descobertas Científicas:** A capacidade de estimar parâmetros e selecionar modelos precisos abre caminho para uma melhor compreensão dos mecanismos subjacentes às vias de sinalização celular. Isso pode levar a descobertas científicas significativas e insights valiosos para o desenvolvimento de terapias e tratamentos médicos.

Essas contribuições destacam a importância deste trabalho no avanço da pesquisa em biologia computacional e modelagem de sistemas biológicos. Além disso, fornecem uma base sólida para futuros estudos e aplicações no campo da biologia de sistemas.

Além disso, é importante ressaltar que este arcabouço foi apresentado e discutido na conferência BSB (*Brazilian Symposium on Bioinformatics*) 2023, onde recebeu reconhecimento e *feedback* valiosos da comunidade científica e sendo posteriormente publicado nos anais do evento (Batista *et al.*, 2023). Para facilitar o acesso à ferramenta e promover a colaboração científica, o código-fonte completo do arcabouço está disponível, de forma livre e gratuita, no seguinte endereço:

github.com/Dynamic-Systems-Biology/BSB-2023-Framework.

5.3 Aplicações futuras

Além de sua relevância para a pesquisa em biologia de sistemas, os resultados obtidos através da utilização deste arcabouço têm potenciais aplicações em várias áreas. Uma dessas aplicações promissoras reside na área da medicina, especialmente na compreensão

e tratamento de doenças complexas, como o câncer. O arcabouço fornece a capacidade de inferir com precisão parâmetros de modelos de vias de sinalização celular, fornecendo informações valiosas sobre os mecanismos subjacentes a essas doenças. Isso, por sua vez, pode contribuir para o desenvolvimento de terapias direcionadas e personalizadas.

Os resultados também têm implicações no campo da biologia sintética, onde a engenharia de vias de sinalização celular é uma área de pesquisa ativa. O arcabouço pode ser uma ferramenta valiosa para auxiliar no projeto e otimização de sistemas biológicos personalizados, com aplicações que vão desde a produção de biocombustíveis até o desenvolvimento de organismos geneticamente modificados para aplicações industriais. Além disso, os resultados podem ser usados em estudos futuros para investigar ainda mais a complexidade das vias de sinalização celular e a interação entre diferentes vias. Isso pode levar a descobertas significativas na compreensão da biologia celular e da regulação de processos fisiológicos.

Por fim, os resultados deste arcabouço têm o potencial de impactar positivamente várias áreas de pesquisa e aplicação, abrindo novas possibilidades para o avanço do conhecimento científico e o desenvolvimento de soluções práticas para desafios complexos.

5.4 Considerações Finais

Ao longo desta dissertação, exploramos a construção de um arcabouço para inferência e seleção de modelos de sinalização celular baseados em equações diferenciais universais (UDEs). Nossos esforços nos permitiram avançar na compreensão das dinâmicas das vias de sinalização celular e na capacidade de ajustar modelos matemáticos aos dados experimentais com eficiência e precisão. Em conclusão, esta dissertação representa um passo inicial em direção a uma compreensão mais profunda das complexas vias de sinalização celular e à construção de ferramentas que podem impulsionar a pesquisa em biologia de sistemas. Esperamos que nossos esforços inspirem novas investigações e avanços na busca pelo entendimento dos segredos das células e sistemas biológicos.

Anexo A

Configuração do ambiente Jupyter e instalação dos componentes

Neste anexo, forneceremos um guia passo a passo para configurar o ambiente Jupyter e instalar os componentes essenciais necessários para executar o arcabouço de inferência de parâmetros e seleção de modelos. Essa configuração é fundamental para garantir que seja possível explorar o exemplo de aplicação de maneira correta e eficiente.

A.0.1 Instalação dos Componentes no Linux

Jupyter Notebook

Para começar, é necessário instalar o Python 3 e o Jupyter Notebook. No terminal, execute os comandos do código A.1.

```
sudo apt-get update
sudo apt-get install python3
python3 -m pip install jupyter
```
(A.1)

Julia

Em seguida, instale o Julia. No terminal, execute os comandos do código A.2.

```
sudo apt-get install julia
```
(A.2)

Abra o interpretador Julia no terminal digitando ”julia“. Em seguida, instale o pacote ”IJulia“ com os comandos do código A.3.

```
using Pkg
Pkg.add(”IJulia“)
```
(A.3)

Após a instalação dos componentes, execute o Jupyter Notebook através do terminal com o comando do código A.4.

```
jupyter notebook
```
(A.4)

A.0.2 Instalação dos Componentes no Windows

Jupyter Notebook

Para usuários do Windows, recomendamos o uso do Anaconda como ambiente Python. Baixe o instalador do Anaconda em www.anaconda.com/distribution/ e siga as instruções para a instalação.

Julia

Baixe o instalador do Julia em julialang.org/downloads/ e siga as instruções de instalação. Depois de instalado, abra o console do Julia a partir do menu Iniciar. No console do Julia, instale o pacote ”IJulia utilizando“ o comando do código A.5.

```
using Pkg
Pkg.add(”IJulia“
```
(A.5)

Após a instalação dos componentes, inicie o Anaconda a partir do menu Iniciar. Na interface do Anaconda, clique em ”Launch“ abaixo de ”Jupyter notebook“ para iniciar o ambiente. Após configurar e iniciar o ambiente Jupyter em qualquer dos sistemas operacionais, uma interface será aberta no navegador padrão. Será necessário localizar e abrir o arquivo ”Pipeline.ipynb“ no diretório de arquivos baixados para executar o exemplo presente neste trabalho.

A.0.3 Download dos arquivos do arcabouço

Os componentes essenciais para a operação do arcabouço podem ser obtidos por meio do repositório oficial do projeto, acessível através do seguinte link: github.com/Dynamic-Systems-Biology/BSB-2023-Framework. Nesse repositório, estão disponíveis os recursos fundamentais que permitem a utilização do arcabouço. Recomenda-se fazer o download dos arquivos diretamente desse repositório para assegurar o uso da versão mais atualizada e estável do arcabouço.

Referências

[Akaike 1974] Hirotugu Akaike. "A new look at the statistical model identification". Em: *IEEE transactions on automatic control* 19.6 (1974), pgs. 716–723 (citado na pg. 14).

[Alberts *et al.* 2014] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts e Peter Walter. "Molecular biology of the cell". Em: (2014) (citado na pg. 5).

[Aldridge *et al.* 2006] Bree B Aldridge, John M Burke, Douglas A Lauffenburger e Peter K Sorger. "Physicochemical modelling of cell signalling pathways". Em: *Nature cell biology* 8.11 (2006), pgs. 1195–1203 (citado na pg. 1).

[Batista *et al.* 2023] Marcelo Batista, Fabio Montoni, Cristiano Campos, Ronaldo Nogueira, Hugo A Armelin e Marcelo S Reis. "A framework for inference and selection of cell signaling pathway dynamic models". Em: *Advances in Bioinformatics and Computational Biology*. Cham: Springer Nature Switzerland, 2023, pgs. 82–93 (citado na pg. 51).

[Bornstein *et al.* 2008] Benjamin J Bornstein, Sarah M Keating, Akiya Jouraku e Michael Hucka. "LibSBML: An API Library for SBML". Em: *Bioinformatics* 24.6 (2008), pgs. 880–881 (citado na pg. 24).

[Carpenter 2014] Graham Carpenter. "Egf receptor biology". Em: *Experimental Cell Research* 333.2 (2014), pgs. 53–65 (citado na pg. 6).

[Chartrand e Zhang 2012] Gary Chartrand e Ping Zhang. "Introductory graph theory". Em: (2012) (citado na pg. 22).

[Chelliah *et al.* 2015] Vijayalakshmi Chelliah, Nick Juty, Ishan Ajmera, Raza Ali, Marine Dumousseau, Mihai Glont e Michael Hucka. "Biomodels: ten-year anniversary". Em: *Nucleic Acids Research* 43.D1 (2015), pgs. D542–D548 (citado na pg. 22).

[Clapham 2007] David E Clapham. "Calcium signaling". Em: *Cell* 131.6 (2007), pgs. 1047–1058 (citado na pg. 6).

[Cooper 2000] Geoffrey M Cooper. "The cell : a molecular approach". Em: (2000). Signaling Molecules and Their Receptors. (citado na pg. 1).

[Draper e Smith 1998] Norman R Draper e Harry Smith. "Applied regression analysis". Em: (1998) (citado na pg. 29).

[Elmore 2007] Susan Elmore. "Apoptosis: a review of programmed cell death". Em: *Toxicologic Pathology* 35.4 (2007), pgs. 495–516 (citado na pg. 7).

[Evans 1988] Ronald M Evans. "The steroid and thyroid hormone receptor superfamily". Em: *Science* 240.4854 (1988), pgs. 889–895 (citado na pg. 7).

[Gelman *et al.* 2013] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari e Donald B Rubin. "Bayesian data analysis". Em: (2013) (citado na pg. 13).

[Gerber 2008] David E Gerber. "Targeted therapies: a new generation of cancer treatments". Em: *American family physician* 77.3 (2008), pgs. 311–319 (citado na pg. 7).

[Green *et al.* 2011] Douglas R Green, Lorenzo Galluzzi e Guido Kroemer. "The pathophysiology of mitochondrial cell death". Em: *Science* 305.5684 (2011), pgs. 626–629 (citado na pg. 7).

[Hairer e Wanner 1993] Ernst Hairer e Gerhard Wanner. "Solving ordinary differential equations. ii: stiff and differential-algebraic problems". Em: *Springer-Verlag* 14 (1993) (citado na pg. 36).

[Hartman 2002] Philip Hartman. *Ordinary differential equations*. SIAM, 2002 (citado na pg. 2).

[Hastie *et al.* 2009] Trevor Hastie, Robert Tibshirani e Jerome Friedman. "The elements of statistical learning: data mining, inference, and prediction". Em: *Springer* 2.1 (2009), pgs. 9–12 (citado na pg. 13).

[Higham 2008] Desmond J Higham. "Modeling and simulating chemical reactions". Em: *SIAM review* 50.2 (2008), pgs. 347–368 (citado na pg. 10).

[Hoops *et al.* 2006] Stefan Hoops, Sven Sahle, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes e Ursula Kummer. "Copasi—a complex pathway simulator". Em: *Bioinformatics* 22.24 (2006), pgs. 3067–3074 (citado na pg. 15).

[Huang *et al.* 2017] Hsin-Ho Huang, Chien-Ming Huang, Shwu-Hua Wu e Yu-Lung Cheng. "Adaptive bfgs method for modeling physiological and pharmacological control systems". Em: *Mathematical Biosciences* 291 (2017), pgs. 73–87 (citado na pg. 26).

[Hucka, Finney, Sauro, Bolouri, J. Doyle *et al.* 2001] Michael Hucka, Andrew Finney, Herbert M Sauro, Hamid Bolouri, J Doyle e Hiroaki Kitano. "The erato systems biology workbench: enabling interaction and exchange between software tools for computational biology". Em: *Biocomputing 2002*. World Scientific, 2001, pgs. 450–461 (citado na pg. 16).

[Hucka, Finney, Sauro, Bolouri, J. C. Doyle *et al.* 2003] 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". Em: *Bioinformatics* 19.4 (2003), pgs. 524–531 (citado na pg. 2).

[Innes *et al.* 2021] Mike Innes, Alan Edelman, Kristoffer Fischer, Christopher Rackauckas e Elsie Saba. "Introducing lux: a julia package for rapid development of custom deep learning models". Em: *The Journal of Open Source Software* 6.57 (2021) (citado na pg. 37).

[Jeske *et al.* 2021] Lisa Jeske, Sandra Placzek, Ida Schomburg, Antje Chang e Dietmar Schomburg. "BRENDA in 2021: new perspectives and new tools in BRENDA". Em: *Nucleic Acids Research* 49.D1 (2021), pgs. D498–D508 (citado na pg. 11).

[Joshi-Tope *et al.* 2005] G Joshi-Tope, M Gillespie, I Vastrik, P D'Eustachio, E Schmidt, B de Bono e B Jassal. "The Reactome: a knowledge base of biologic pathways and processes." Em: *Nucleic Acids Research* 33.Database issue (2005), pgs. D428–D432 (citado na pg. 11).

[Kanehisa *et al.* 2016] Minoru Kanehisa, Yoko Sato, Masayuki Kawashima, Miho Furumichi e Mao Tanabe. "KEGG: kyoto encyclopedia of genes and genomes". Em: *Nucleic Acids Research* 44.D1 (2016), pgs. D457–D462 (citado na pg. 11).

[Karp *et al.* 2002] Peter D Karp, Suzanne Paley e Pablo Romero. "BioCyc: a collection of pathway/genome databases for comparative genomics". Em: *Nucleic Acids Research* 30.1 (2002), pgs. 56–58 (citado na pg. 11).

[Kholodenko 2000] Boris N Kholodenko. "Cell-signalling dynamics in time and space". Em: *Nature Reviews Molecular Cell Biology* 1.2 (2000), pgs. 116–126 (citado na pg. 7).

[Kingma e Ba 2014] Diederik P Kingma e Jimmy Ba. "Adam: a method for stochastic optimization". Em: *arXiv preprint arXiv:1412.6980* (2014) (citado na pg. 27).

[Kluyver *et al.* 2016] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Julien Frederic e Kyle Kelley. "Jupyter notebooks - a publishing format for reproducible computational workflows". Em: *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. Ed. por Fernando Loizides e Birgit Schmidt. IOS Press, 2016, pgs. 87–90 (citado na pg. 3).

[Kohavi 1995] Ron Kohavi. "A study of cross-validation and bootstrap for accuracy estimation and model selection". Em: *Ijcai*. Vol. 14. 2. Montreal, Canada. 1995, pgs. 1137–1145 (citado nas pgs. 14, 33).

[Lal 2015] Mahesh Lal. "Neo4j graph data modeling". Em: (2015) (citado na pg. 22).

[Le Novère *et al.* 2009] Nicolas Le Novère, Michael Hucka, Huaiyu Mi, Stuart Moodie, Falk Schreiber, Anatoly Sorokin, Emek Demir, Katja Wegner, Mirit I Aladjem, Sarala M Wimalaratne, Frank T Bergman, Ralph Gauges, Peter Ghazal, Hideya Kawaji, Lu Li, Yukiko Matsuoka, Alice Villéger, Sarah E Boyd, Laurence Calzone, Melanie Courtot, Ugur Dogrusoz, Tom C Freeman, Akira Funahashi, Samik Ghosh, Akiya Jouraku, Sohyoung Kim, Fedor Kolpakov, Augustin Luna, Sven Sahle, Esther Schmidt, Steven Watterson, Guanming Wu, Igor Goryanin, Douglas B Kell, Chris Sander, Herbert Sauro, Jacky L Snoep, Kurt Kohn e Hiroaki Kitano. "The Systems Biology Graphical Notation". eng. Em: *Nature Biotechnology* 27.8 (ago. de 2009), pgs. 735–741. ISSN: 1546-1696 (citado na pg. 16).

[Lemmon e Schlessinger 2010] Mark A Lemmon e Joseph Schlessinger. "Cell signaling by receptor tyrosine kinases". Em: *Cell* 141.7 (2010), pgs. 1117–1134 (citado na pg. 5).

[Levine 2019] Ira N Levine. "Physical chemistry". Em: (2019) (citado na pg. 9).

[Liepe *et al.* 2010] Juliane Liepe, Chris Barnes, Erika Cule, Kamil Erguler, Paul Kirk, Tina Toni e Michael PH Stumpf. "Abc-sysbio—approximate bayesian computation in python with gpu support". Em: *Bioinformatics* 26.14 (2010), pgs. 1797–1799 (citado na pg. 17).

[Lodish *et al.* 2000] Harvey Lodish, Arnold Berk, S Lawrence Zipursky, Paul Matsudaira, David Baltimore e James Darnell. "Molecular cell biology". Em: (2000) (citado na pg. 6).

[Lu *et al.* 2018] Huanxiang Lu, Xin Tong e Fan Zhang. "Newton-raphson optimization for parameter estimation in biological models: a review". Em: *Mathematical biosciences and engineering: MBE* 15.1 (2018), pgs. 41–60 (citado na pg. 26).

[Mangelsdorf e Evans 1995] David J Mangelsdorf e Ronald M Evans. "The nuclear receptor superfamily: the second decade". Em: *Cell* 83.6 (1995), pgs. 835–839 (citado na pg. 7).

[Marin *et al.* 2021] Guy B Marin, Vladimir V Galvita e Gregory S Yablonsky. "Kinetics of chemical processes: from molecular to industrial scale". Em: *Journal of Catalysis* 404 (2021), pgs. 745–759 (citado na pg. 8).

[Matsuoka *et al.* 2014] Yukiko Matsuoka, Akira Funahashi, Samik Ghosh e Hiroaki Kitano. "Modeling and simulation using celldesigner". Em: *Transcription Factor Regulatory Networks*. Springer, 2014, pgs. 121–145 (citado na pg. 16).

[Miller *et al.* 2019] Jack R Miller, Sean P Pinney, Arvind Muley, Christopher J Pinnell, Eoin Lee, Luis de la Torre-Ubieta e Dianne Finkelstein. "The reactome pathway knowledgebase: a 2019 update". Em: *Nucleic Acids Research* 47.D1 (2019), pgs. D596–D604 (citado na pg. 11).

[Mogensen *et al.* 2018] P Mogensen, A Larsen e N Städler. "Optim.jl: a mathematical optimization package for julia". Em: *Journal of Open Source Software* 3.24 (2018), pg. 615 (citado na pg. 26).

[Montoni, De Sousa *et al.* 2022] Fabio Montoni, Ronaldo N De Sousa, Marcelo B De Lima Junior, Cristiano G S Campos, Willian Wang, Vivian M Constantino, Cássia S Sanctos, Hugo A Armelin e Marcelo S Reis. "Anguix: cell signaling modeling improvement through sabio-rk association to reactome". Em: *2022 IEEE 18th International Conference on e-Science (e-Science)*. 2022, pgs. 425–426 (citado na pg. 13).

[Montoni, Sousa *et al.* 2022] Fabio Montoni, Ronaldo Sousa, Marcelo L Junior, Cristiano Campos, Vivian Constantino, Willian Wang, Cássia Sanctos, Hugo Armelin e Marcelo Reis. "Integration of sabio-rk to the reactome graph database for efficient gathering of cell signaling pathways". Em: *Anais do XVI Brazilian e-Science Workshop*. Niterói: SBC, 2022, pgs. 105–108 (citado nas pgs. 2, 13).

[Mortimer 2008] Robert G Mortimer. "Physical chemistry". Em: (2008) (citado na pg. 8).

[Müller-Wille 2010] Staffan Müller-Wille. "Cell theory, specificity, and reproduction, 1837–1870". Em: *Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences* 41.3 (2010). The cell as nexus: connections between the history, philosophy and science of cell biology, pgs. 225–231. issn: 1369-8486. url: https://www.sciencedirect.com/science/article/pii/S1369848610000427 (citado na pg. 1).

[Neo4j Developers 2021] Neo4j Developers. *Cypher Reference Manual*. neo4j.com/docs/cypher-manual/current/. Accessed: April 7, 2023. 2021 (citado na pg. 23).

[NJ e R 2004] Eungdamrong NJ e Iyengar R. "Modeling cell signaling networks". Em: *Biology of the Cell* 96(5):355-362 (2004) (citado na pg. 2).

[Peleg *et al.* 2012] Micha Peleg, Mark D Normand e Maria G Corradini. "The arrhenius equation revisited". Em: *Critical reviews in food science and nutrition* 52.9 (2012), pgs. 830–851 (citado na pg. 9).

[Rapoport 2007] Tom A Rapoport. "Protein translocation across the eukaryotic endoplasmic reticulum and bacterial plasma membranes". Em: *Nature* 450.7170 (2007), pgs. 663–669 (citado na pg. 6).

[Ron e Walter 2002] David Ron e Peter Walter. "Translational control in the endoplasmic reticulum stress response". Em: *Journal of Clinical Investigation* 110.10 (2002), pgs. 1383–1388 (citado na pg. 7).

[Rubin 1984] Donald B Rubin. "Bayesianly Justifiable and Relevant Frequency Calculations for the Applied Statistician". Em: *The Annals of Statistics* 12.4 (1984), pgs. 1151–1172 (citado na pg. 1).

[Ruder 2017] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2017. arXiv: 1609.04747 [cs.LG] (citado na pg. 26).

[Schwarz 1978a] Gideon Schwarz. "Estimating the dimension of a model". Em: *The Annals of Statistics* 6.2 (1978), pgs. 461–464 (citado na pg. 29).

[Schwarz 1978b] Gideon Schwarz. "Estimating the dimension of a model". Em: *The annals of statistics* (1978), pgs. 461–464 (citado na pg. 14).

[Shannon *et al.* 2003] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski e Trey Ideker. "Cytoscape: a software environment for integrated models of biomolecular interaction networks". Em: *Genome research* 13.11 (2003), pgs. 2498–2504 (citado na pg. 1).

[T e P 2017] Otto T e Sicinski P. "Cell cycle proteins as promising targets in cancer therapy". Em: *Nature Reviews Cancer* 17(2): 93-115 (2017) (citado na pg. 1).

[White 2002] Morris F White. "Insulin signaling in health and disease". Em: *Science* 302.5651 (2002), pgs. 1710–1711 (citado na pg. 6).

[Wickham e Francois 2020] Hadley Wickham e Romain Francois. *Csv-Parser: Read and Write CSV Files*. github.com/tidyverse/readr/blob/master/src/csv-parser.h. [Online; accessed 08-Mar-2023]. 2020 (citado na pg. 24).

[Wikipedia 2019] Wikipedia. *Filtro de Kalman — Wikipédia, a enciclopédia livre*. Alteração de 16 de abril de 2019. 2019. URL: https://pt.wikipedia.org/w/index.php?title=Filtro_de_Kalman&oldid=54840577 (citado na pg. 1).

[Willmott 1981] Cort J Willmott. "On the validation of models". Em: *Physical Geography* 2.2 (1981), pgs. 184–194 (citado na pg. 29).

[Willmott e Matsuura 2005] Cort J Willmott e Kenji Matsuura. "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance". Em: *Climate Research* 30.1 (2005), pgs. 79–82 (citado na pg. 14).

[Wittig *et al.* 2012] Ulrike Wittig, Renate Kania, Martin Golebiewski, Melanie Rey, Lina Shi, Luitzen Jong e Edita Algaa. "Sabio-rk–database for biochemical reaction kinetics". Em: *Nucleic acids research* 40.D1 (2012), pgs. D790–D796 (citado na pg. 11).