UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

GUILHERME RAFAEL GRAEFF

Exploração de técnica de visualização de dados aplicada à Dinâmica Molecular: Análise de Mapas de Correlação Cruzada Dinâmica como Redes Dinâmicas

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Prof. Dr. Márcio Dorn

Porto Alegre
2026

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Graeff, Guilherme Rafael

Exploração de técnica de visualização de dados aplicada à Dinâmica Molecular: Análise de Mapas de Correlação Cruzada Dinâmica como Redes Dinâmicas / Guilherme Rafael Graeff. – Porto Alegre: PPGC da UFRGS, 2026.

64 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2026. Orientador: Márcio Dorn.

1. Visualização de dados. 2. Redes Dinâmicas. 3. Bioinformática Estrutural. 4. Análise Espaço-Temporal. I. Dorn, Márcio. II. Título.

“This is only a foretaste of what is to come
and only the shadow of what is going to be.”
— Alan Turing

RESUMO

A Bioinformática é uma área multidisciplinar que faz uso de ferramentas computacionais para a análise de dados biológicos. Uma das áreas de aplicação da Bioinformática é a Biologia Estrutural, a qual estuda o funcionamento de fenômenos biológicos a partir das estruturas tridimensionais de biomoléculas, como de proteínas. Neste contexto, Dinâmica Molecular é a técnica que permite a simulação computacional de sistemas moleculares a partir de suas estruturas. Esta técnica é baseada na utilização de equações de mecânica clássica (campos de força), tratando cada átomo da proteína como partícula e determinando suas posições e velocidades ao longo do tempo a partir de energias potenciais. Ao longo da simulação, ocorre o registro das diferentes conformações e posições da estrutura em tempos pré-determinados. Desse modo, a Dinâmica Molecular é um método que permite o estudo da dinâmica de interação de biossistemas de forma atomística. Entretanto, esta técnica enfrenta desafios relacionados à complexidade dos dados devido à alta dimensionalidade, ao alto custo computacional para a realização das simulações destes sistemas e ao armazenamento de seu grande volume. Desta forma, faz-se necessário desenvolver métodos computacionais que facilitem a análise, integração e visualização dos dados obtidos através das simulações. Para encontrar padrões informativos nos complexos simulados, um dos métodos utilizados na análise de simulações de complexos moleculares é o Mapa Dinâmico de Correlações Cruzadas, denominado *Dynamic Cross Correlation Map* (DCCM). O DCCM analisa a média da correlação dos vetores de deslocamento cartesiano de todos contra todos entre os resíduos de aminoácidos presentes no sistema, levando em consideração as diferentes conformações adotadas durante a simulação. O método de DCCM pode ser interpretado como uma matriz de adjacência conforme descrito na teoria de Grafos. Considerando a dimensão do tempo e estados da simulação, esta matriz se torna dinâmica, caracterizando a estrutura de dados necessária para a construção de uma rede dinâmica (*Dynamic Network*). Desta forma, o objetivo deste trabalho é explorar a transformação, representação e análise de dados estruturais sob uma perspectiva computacional, de forma a aproximar as áreas de Visualização de Dados e Bioinformática Estrutural. Este estudo possui caráter exploratório, visando compreender o entendimento da representação de dados proposta, investigar maneiras de representar os dados resultantes de análise de Dinâmica Molecular utilizando redes dinâmicas e propor uma visualização integrada e informativa dos processos moleculares.

Palavras-chave: Visualização de dados. Redes Dinâmicas. Bioinformática Estrutural.

Análise Espaço-Temporal.

ABSTRACT

Bioinformatics is a multidisciplinary field that utilizes computational tools to analyze biological data. Structural Bioinformatics is one area that stems from this field, studying the workings of biological phenomena through the three-dimensional structures of biomolecules, such as proteins. Within this context, Molecular Dynamics is the technique that allows for the computational simulation of molecular systems from their structures. This method relies on using equations from classical mechanics (force fields), treating each protein atom as a particle, and determining its position and velocities over time using potential energies. During the simulation, the conformation and position of the structure are recorded at predetermined time steps. In this way, Molecular Dynamics is a method that enables the study of the interaction dynamics of biosystems at an atomic level. Nonetheless, this technique faces challenges related to data complexity resulting from high dimensionality, high computational cost to simulate these systems, and high demand for storage of this data volume. There is a need to develop computational methods that facilitate the analysis, integration, and visualization of data obtained from simulations. To find informative patterns in the simulated complex, one method utilized in the analysis of complex molecular systems is the Dynamic Cross Correlation Map (DCCM). DCCM analyzes the average correlation of the Cartesian displacement vectors of all-versus-all amino acid residues present in the system, taking into account the different conformations adopted during the simulation. The DCCM method can be interpreted as an adjacency matrix as described in Graph Theory. Considering the time dimension and simulation states, this matrix becomes dynamic, characterizing the data structure necessary for constructing a dynamic network. In this way, the objective of this work is to explore the transformation, representation, and analysis of structural data from a computational perspective, thereby bringing the fields of Data Visualization and Structural Bioinformatics closer together. This study has an exploratory character, aiming to understand the proposed data representation, investigate ways to represent data resulting from Molecular Dynamics analysis using dynamic networks, and propose an integrated and informative visualization of molecular processes.

Keywords: Data Visualization. Dynamic Networks. Structural Bioinformatics. Spatio-temporal Analysis.

LISTA DE ABREVIATURAS E SIGLAS

DCCM	Dynamic Cross Correlation Map (Mapa Dinâmico de Correlações Cruzadas)
DM	Dinâmica Molecular
DSSP	Dictionary of Secondary Structure of Proteins (Dicionário de Estrutura Secundária de Proteínas)
GPU	Graphical Processing Unit (Unidade de Processamento Gráfico)
GUI	Graphical User Interface (Interface Gráfica do Usuário)
HPC	High Performance Computing (Computação de Alta Performance)
MC1R	Melanocortin-1 Receptor (Receptor Melanocortin-1)
mDCC	Multi-modal Dynamic Cross Correlation (Correlação Cruzada Dinâmica Multimodal)
PCA	Principal Component Analysis (Análise de Componente Principal)
RGB	Red, Green, Blue (Vermelho, Verde, Azul)
RMSD	Root Mean Square Deviation (Desvio Quadrático Médio)
RMSF	Root Mean Square Fluctuation (Flutuação Quadrática Média)
TDDCC	Time Dependent Dynamic Cross Correlation (Correlação Cruzada Dinâmica Dependente do Tempo)
WebGL	Web Graphics Library (Biblioteca Web Gráfica)
WT	Wild Type (Tipo Selvagem)

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

A Bioinformática é uma área de caráter multidisciplinar, na qual o papel da Ciência da Computação neste campo do conhecimento é fundamental no desenvolvimento de ferramentas utilizadas para múltiplas tarefas que dependem desta característica que considera tanto conceitos presentes na Biologia quanto da Computação (Verli, 2014). O desenvolvimento do conhecimento científico nesta área se estabelece a partir da colaboração mútua destas áreas do conhecimento. O desenvolvimento intenso e recente da tecnologia da informação considerando o uso de Unidades de Processamento Gráfico (*Graphical Processing Unit* - GPU) influencia diversas ciências, contemplando o desenvolvimento do campo da Bioinformática Estrutural (Loukatou et al., 2014; Hollingsworth; Dror, 2018). A alta velocidade de processamento computacional culmina em um grande volume de dados que, portanto, necessitam de ferramentas que possibilitem o tratamento e análise dos mesmos. E por se tratar da intersecção do conhecimento, explorar as fronteiras existentes em ambos domínios se torna um desafio que deve ser superado a partir da conversa entre ambos os lados desta fronteira do conhecimento (Verli, 2014). Isto estabelece vínculos que definem o próprio campo da Bioinformática, permitindo o desenvolvimento de ambas as ciências.

Nesse cenário, a Computação entra em contato com a Biologia em diversas aplicações em diferentes áreas específicas da Bioinformática (Verli, 2014), como Genômica, Transcriptômica, Multiômica, Metabolômica, Dinâmica Molecular (DM) entre outras. Embora distintas em seus objetos de estudo, estas áreas convergem no objetivo de caracterizar e quantificar componentes biológicos em nível molecular. Aplicando técnicas provindas da computação para resolver problemas relacionados com a biologia, estes problemas devem ser modelados de alguma maneira para que seja possível a aplicação de diferentes algoritmos de análise e processamento dos mesmos. Existem diferentes maneiras com que a computação pode abordar estes desafios, contudo o conhecimento biológico é imprescindível para análises, pois é este que se responsabiliza pelo sentido da investigação. Neste contexto é necessário compreender, em certa medida, os tópicos relacionados às áreas que englobam o método que está sendo abordado no trabalho.

Especificamente, a Dinâmica Molecular é um método explorado pela Bioinformática Estrutural, esta aborda a simulação computacional de complexos moleculares com a finalidade de extrair informações relevantes ao contexto biológico (Hollingsworth; Dror, 2018). Esta técnica é proveniente do desenvolvimento de diferentes conceitos que englo-

bam diferentes áreas do conhecimento que interagem com a Computação, como Física e Química (Verli, 2014). Dentre as diversas formas de representações propostas aos dados resultantes destas simulações (Belghit et al., 2024), este trabalho modela as trajetórias obtidas da simulação como um grafo dinâmico (Patel; Sinha; Palermo, 2024), buscando explorar as relações entre os componentes que formam os complexos moleculares. A Visualização de Dados é a maneira escolhida para a exploração e análise, buscando ampliar a compreensão das métricas utilizadas para análise.

Para viabilizar essa compreensão, este trabalho aborda conceitos sobre Visualização de Dados aplicada, explorando a execução de técnicas de visualização em um domínio específico do conhecimento. Faz parte do desenvolvimento desta pesquisa a modelagem do problema, para que seja possível adaptar os dados e que estes sirvam de entrada para uma abordagem capaz de visualizar a informação, considerando o tempo. A abordagem investiga a utilização de um ambiente tridimensional interativo para a análise de correlações através da sobreposição de componentes, propondo vínculos visuais para a análise de dados de maneira dinâmica (Bach; Pietriga; Fekete, 2014). Os dados são provenientes de investigações de um domínio específico do conhecimento, onde o papel desta investigação é desenvolver o campo de análise destes dados considerando a perspectiva da computação.

Portanto, modelar o problema e transformar os dados faz parte do desenvolvimento da abordagem, com o foco em manipular o dado de entrada a fim de contemplar o formato necessário para que seja possível a visualização do mesmo. A solução do problema transforma a entrada que são múltiplos estados de coordenadas cartesianas em uma matriz quadrada e simétrica que contém a correlação do deslocamento cartesiano das variáveis. Para que seja possível a visualização deste dado, se faz o uso destas matrizes como matrizes de adjacências de uma rede dinâmica (Patel; Sinha; Palermo, 2024). Contemplando o uso da teoria de Grafos para a representação desta rede. Abordar o problema desta maneira garante uma base teórica para o desenvolvimento deste método. A interpretação do dado como matrizes de adjacência permite o contato com a técnica de visualização (Bach, 2016).

A Computação é responsável pela manipulação dos dados para que seja possível a extração das características necessárias para a aplicação do método. São contribuições: a modelagem do problema, transformação dos dados e produção de uma visualização dos dados, nestas estão presentes aplicações de conceitos provindos diretamente da Ciência da Computação. O trabalho abrange o uso de métodos e métricas utilizadas em diferentes

contextos, como matemática e estatística.

A visualização da informação — isto é, a representação visual da abstração utilizando uma abordagem interativa capaz de renderizar os dados processados — também constitui uma contribuição deste trabalho. Utilizando técnicas desenvolvidas para casos genéricos (Bach et al., 2017) no domínio específico definido, explorando os possíveis caminhos destes trabalhos desenvolvidos pela área computação, buscando ampliar a aplicação destas técnicas em diferentes contextos.

1.1 Objetivos gerais

O objetivo deste trabalho é explorar a técnica de visualização de dado denominada Matriz Cúbica aplicada em dados provenientes de análises de simulações de Dinâmica Molecular, explorando o método genérico aplicado ao domínio específico do conhecimento. Abordando este dado de uma maneira diferente da usual, investigando a viabilidade da ideia da utilização deste método para auxiliar as investigações de interações em diferentes sistemas simulados. Buscando preencher a lacuna presente nas fronteiras do conhecimento que formam a Bioinformática, focando na exploração do desenvolvimento de protótipo de ferramenta dedicada a análise visual de dados provenientes da Bioinformática Estrutural.

1.2 Objetivos específicos

Para alcançar o objetivo geral proposto, os seguintes objetivos específicos foram definidos para guiar as etapas de desenvolvimento e avaliação do método:

- Desenvolvimento de algoritmo capaz de considerar a trajetória de DM de maneira fatiada em relação ao tempo, pretendendo perceber correlações lineares que se ocultem na média do método usual.
- Desenvolvimento de protótipo de ferramenta de visualização de dados resultantes da aplicação de um método de análise de simulação de DM.
- Exploração da aplicação em domínio específico do conhecimento de uma técnica de visualização de dados desenvolvida de maneira genérica.
- Validar a abordagem visual proposta através da comparação com a análise de DCCM tradicional, demonstrando a capacidade do protótipo de ferramenta de revelar cor-

relações transientes e dinâmicas do sistema.

1.3 Organização

Esta dissertação está estruturada em seis capítulos: O **Capítulo 1** introduz o desafio de analisar dados de DM, apresenta os objetivos da pesquisa e a organização do texto. O **Capítulo 2** revisa os conceitos de Bioinformática Estrutural, DCCM, Teoria de Grafos e as técnicas de Visualização de Dados, como a Matriz Cúbica, que dão base ao trabalho. O **Capítulo 3** analisa publicações anteriores, contextualizando esta pesquisa ao revisar o uso de DCCM na Bioinformática Estrutural e outras abordagens de visualização de redes dinâmicas. O **Capítulo 4** detalha a metodologia de implementação, descrevendo o algoritmo em *Python/MDtraj* para o processamento segmentado dos dados e a estrutura do protótipo de ferramenta de visualização interativa em *JavaScript/Three.js*. O **Capítulo 5** avalia o protótipo de ferramenta comparando a visualização proposta com a análise de DCCM estática tradicional, demonstrando a capacidade do método em revelar correlações ocultas. Por fim, o **Capítulo 6** sumariza as contribuições, avalia o cumprimento dos objetivos e sugere caminhos para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Alguns conceitos são essenciais para um melhor entendimento do trabalho proposto, são fundamentos que definem as bases do que é discutido. Estes conceitos vêm da Bioinformática Estrutural e da Computação, a origem da ideia parte de um método utilizado para a análise de trajetórias de Dinâmica Molecular e então se mescla com conceitos de Grafos e Visualização de Dados. As seções a seguir descrevem a teoria destes conceitos.

2.1 Bioinformática Estrutural e Dinâmica Molecular

A DM descreve a variação do comportamento molecular como função do tempo (Verli, 2014). Esta simulação considera condições específicas da biologia, física e química (Verli, 2014) que definem um ambiente virtual que busca representar a interação entre os átomos. Este método é atomístico, ou seja, simula as interações entre os átomos, calculando as propriedades físicas que constroem a simulação, levando em consideração todos os átomos presentes. Esta característica torna a simulação de DM uma tarefa de alto custo computacional (Verli, 2014) que também gera um grande volume de dados, estes dados também possuem a complexidade Biológica intrínseca. As análises de DM buscam explorar as conformações adotadas pelos sistemas moleculares alvos dos estudos, estas conformações podem estar relacionadas a função biológica que aquele sistema possui (Verli, 2014). O desenvolvimento de técnicas de Computação de Alta Performance (*High Performance Computing* - HPC) contribui para o desenvolvimento de técnicas que utilizam GPU para processar dados de DM (Loukatou et al., 2014).

A bioinformática estrutural busca compreender a relação entre a estrutura tridimensional e a função das biomoléculas. O estudo de proteínas, DNA, RNA, suas interações entre si e com outras moléculas por meio de métodos computacionais, permite a compreensão de mecanismos biológicos, avanços na engenharia de enzimas, estudo de efeitos mutacionais e design de novos fármacos. Dentro desta área, a simulação por meio de dinâmica molecular cumpre o papel de descrever os movimentos das biomoléculas ao longo de um determinado tempo. A DM baseia-se em equações da mecânica clássica, atribuindo força, massa e aceleração às partículas representadas em um sistema e, assim, descreve o seu comportamento ao longo do tempo. Por exemplo, em um sistema formado por uma proteína envolta por moléculas de água em um espaço tridimensional definido,

cada átomo do sistema tem sua massa definida pela tabela periódica e a força sobre cada átomo é fornecida pela temperatura (Verli, 2014). Além disso, cada átomo de uma molécula está ligado a outro átomo sujeito a forças interatômicas e interagindo com outras moléculas por forças intermoleculares. O cálculo destas forças é fornecido por uma equação matemática denominada campo de força, responsável por reproduzir características do comportamento molecular (Hollingsworth; Dror, 2018). Sendo assim, o campo de força descreve a energia potencial intrínseca em ângulos, torções, ligações e interações atômicas (Allen, 2004). Dessa forma, a força fornecida pela temperatura e pelo campo de força aliada à massa permite a resolução das equações de movimento de Newton para a obtenção da velocidade, aceleração e energia associadas a cada partícula do sistema. Por fim, a descrição de um sistema molecular a partir da mecânica clássica e dos campos de força permite um entendimento em nível atômico do mesmo, tornando-se uma ferramenta valiosa para o entendimento de processos biológicos.

A partir das coordenadas iniciais do sistema, o processo da DM calcula as variações conformacionais, de velocidade e de energia do mesmo. Essas variações são calculadas em intervalos de tempo curtos, para garantir a estabilidade numérica, normalmente apenas alguns femtossegundos (10^{-15}s) cada (Hollingsworth; Dror, 2018). Esses intervalos de tempo podem ser entendidos como *frames* da simulação, onde cada *frame* representa as variações do sistema naquele tempo. O conjunto desses *frames* define a trajetória do sistema. Portanto, o resultado final da simulação consiste no conjunto de coordenadas cartesianas de cada átomo do sistema ao longo do tempo da simulação. Essa trajetória é salva em arquivos do tipo .trr ou .xtc. E o intervalo de tempo de cada *frame* salvo pode ser alterado posteriormente à simulação, conforme a análise a ser realizada. Além dos arquivos de trajetória, a simulação ainda gera resultados de coordenadas (.gro), de topologia (.top ou .itp) e de energia (.edr).

Por fim, a DM permite uma avaliação em nível atomístico e dinâmico de sistemas biológicos, contribuindo não apenas para a bioinformática estrutural como também na medicina, biotecnologia, ecologia e indústria farmacêutica. Além disso, o avanço do poder computacional permitiu o aumento da complexidade dos sistemas biológicos estudados pela DM, bem como o aumento do tempo de simulação e número de réplicas. Todos esses fatores tornam a análise dos resultados de DM complexa, devido ao volume de dados gerados (Hollingsworth; Dror, 2018). Por isso, técnicas de visualização que auxiliem análises mais detalhadas são cada vez mais necessárias nesta área.

2.1.1 Análise de Trajetórias de DM

Existem, para a simulação de DM, diversas análises que permitem o estudo dos comportamentos adotados pelo sistema molecular ao longo do tempo (Baltrukevich; Podlewska, 2022). O uso de cada uma delas traz diferentes perspectivas para o dado gerado a partir da simulação, por exemplo, estabilidade de estruturas, movimentos coordenados e mudanças de estados. Desse modo, a utilização conjunta de diferentes análises permite a produção de *insights* referentes ao contexto biológico em que aquele sistema está inserido. Além disso, por serem realizadas em etapas posteriores a simulação, estas análises partem de dados de trajetória (coordenadas cartesianas) e geram novos formatos de dados, como um arquivo que contém com distância entre resíduos. Cada dado gerado pode ser abordado por diferentes técnicas computacionais no estudo da Bioinformática Estrutural, sejam métodos de visualização, estatísticos ou de aprendizado de máquina.

Dentre as análises existentes, as relevantes para a presente pesquisa são, RMSD, RMSF, PCA e DCCM, esta última que terá seu conteúdo desenvolvido na próxima subseção por conta da importância e relação com este trabalho.

O Desvio Médio Quadrático (Root Mean Square Deviation - RMSD) (Verli, 2014) é a métrica utilizada para realizar o alinhamento (superposição) de trajetórias, como feito pela função *superpose* no MDTraj (McGibbon et al., 2015). O alinhamento de estruturas busca minimizar a distância entre átomos correspondentes (como os $C\alpha$) em diferentes conformações da trajetória (Verli, 2014; Yu; Dalby, 2020). Isto é alcançado através da busca por transformações (rotação e translação) que satisfaçam o menor RMSD. Assim, ao minimizar os vetores de posição dos átomos selecionados através da distância entre eles, as conformações que mais se assemelham são efetivamente sobrepostas, facilitando a análise comparativa de sistemas moleculares. O cálculo do RMSD é da forma descrita pela equação 2.1 onde *d* é a distância entre os átomos entre as duas conformações e *n* é o número de conformações adotadas pelo sistema.

$$RMSD = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{2.1}$$

O RMSF (Root Mean Square Fluctuation), ou Flutuação Média Quadrática, é uma métrica utilizada para quantificar a mobilidade ou flexibilidade individual de cada átomo (ou resíduo) ao longo da trajetória de simulação (Verli, 2014). Diferente do RMSD, que mede o desvio global da estrutura, o RMSF calcula o desvio médio quadrático da posição

de cada átomo em relação à sua própria posição média, calculada após um alinhamento inicial de todos os estados da trajetória. Valores elevados de RMSF indicam regiões de alta flexibilidade, enquanto valores baixos são característicos de regiões estruturalmente mais rígidas, como hélices-alfa e folhas-beta. O cálculo do RMSF é da forma descrita pela equação 2.2, onde Δr_i é o vetor que representa o deslocamento cartesiano do átomo i.

$$\mathrm{RMSF}_i = \sqrt{\langle \Delta \boldsymbol{r}_i^2 \rangle} \tag{2.2}$$

A Análise de Componentes Principais (PCA), é uma técnica estatística de redução de dimensionalidade usada para identificar os movimentos coletivos e de grande amplitude em variados tipos de dados, inclusive em uma simulação de dinâmica molecular. O PCA transforma a matriz de covariância das posições atômicas (geralmente dos carbonos alfa, após o alinhamento) em um conjunto de vetores ortogonais chamados componentes principais (Greenacre et al., 2022) (ou autovetores). Por utilizar esta matriz, o método é relevante para este trabalho. Esses componentes são ordenados por importância, de modo que os primeiros (PC1, PC2, etc.) descrevem a maior parte da variância (ou seja, o "movimento essencial") do sistema (Greenacre et al., 2022). Ao projetar a trajetória nesses poucos componentes principais, é possível visualizar e analisar as transições conformacionais dominantes em um espaço de baixa dimensão.

2.1.2 Mapas Dinâmicos de Correlação Cruzada

A aplicação do método de DCCM considera apenas os Carbonos Alfas ($C\alpha$) presentes na proteína, isto permite a exploração de proteínas com tamanho considerável, esta simplificação faz parte da análise usual destes dados (Hünenberger; Mark; Gunsteren, 1995; Kasahara; Fukuda; Nakamura, 2014; Dash et al., 2022; Okamoto; Ando, 2024; Parida; Paul; Chakravorty, 2020; Wang et al., 2022; Cavatão et al., 2024). Este cálculo é definido pelo quociente entre o produto interno médio (covariância) dos vetores tridimensionais pelo produto das raízes quadradas da variância do dado (Okamoto; Ando, 2024). Isto faz com que as contribuições dos dados a partir de suas médias em direções semelhantes somem e em momentos que estas direções sejam opostas, esta contribuição é subtraída, o que denota o comportamento sinérgico dos vetores tridimensionais em relação as posições médias do dado (Okamoto; Ando, 2024). O fato de estes vetores estarem

dispostos no espaço torna a interpretação deste método mais complexa do que a utilização de um ambiente em duas dimensões, pois a terceira dimensão adiciona um fator combinatório para as correlações acontecerem, considerando que são as correlações lineares em um espaço tridimensional (Kasahara; Fukuda; Nakamura, 2014; Bernetti et al., 2024). Contudo a normalização do dado através da divisão dos produtos da variâncias garante que a correlação final capture as covariâncias totais (Okamoto; Ando, 2024) e a torne o limite superior e também inferior (Kasahara; Fukuda; Nakamura, 2014), garantindo que as covariâncias que mais contribuem se destaquem. Esta normalização, por definição, coloca os valores de correlação entre -1 e +1 (Dash et al., 2022; Wang et al., 2022; Okamoto; Ando, 2024; Cavatão et al., 2024; Patel; Sinha; Palermo, 2024), onde o -1 indica anti-correlação total dos dados e o +1 indica correlação total, ou seja, em valores negativos há correlação oposta entre as partículas, logo os vetores que calculam o método possuem direções opostas, denotando movimentos sinérgicos, isto é, as partículas se afastam ou se aproximam (Wang et al., 2022), a correlação positiva denota que os vetores em relação à sua média possuem característica que indica um movimento sinérgico na mesma direção, isto é, as partículas se movimentaram de maneira similar (Wang et al., 2022).

A fórmula que determina este cálculo é construída através da variância eq. 2.4 e covariância eq. 2.3, e por fim temos o método como um todo, o mapa dinâmico de correlações cruzadas na equação 2.5. A covariância é a comparação de dois sinais, neste caso, vetores tridimensionais já a variância é a auto-comparação do sinal. Nestas equações, $\langle ... \rangle$ significa o cálculo da média total em relação ao dado como um todo, $c(i,j)$ é a covariância entre as partículas i e j, $c(i,i)$ é a variância sobre o vetor de deslocamento cartesiano de uma partícula e por fim $C(i,j)$ é a correlação de *Pearson* ou DCCM.

$$c(i,j) = \langle \Delta \mathbf{r}_i \cdot \Delta \mathbf{r}_j \rangle \tag{2.3}$$

$$c(i,i) = \langle \Delta \mathbf{r}_i \cdot \Delta \mathbf{r}_i \rangle = \left\langle \Delta \mathbf{r}_i^2 \right\rangle \tag{2.4}$$

$$C(i,j) = \frac{c(i,j)}{[c(i,i)c(j,j)]^{1/2}} = \frac{\langle \Delta \mathbf{r}_i \cdot \Delta \mathbf{r}_j \rangle}{\left\langle \Delta \mathbf{r}_i^2 \right\rangle^{1/2} \left\langle \Delta \mathbf{r}_j^2 \right\rangle^{1/2}} \tag{2.5}$$

É importante denotar que, embora este método estatístico demonstre correlações e anti-correlações lineares, o que esta informação significa no contexto biológico é responsabilidade do cientista que analisa os dados da sua pesquisa que envolve Bioinformática

Estrutural. Aqui estamos explorando as possibilidades de visualização destas correlações considerando o tempo, onde Bach, Pietriga e Fekete (2014) utilizam redes de correlação para a visualização com Matrizes Cúbicas, no contexto da conectividade de sinais referentes a funções cerebrais, e aqui abordamos o dado sob a perspectiva da Bioinformática Estrutural.

Os trabalhos relacionados da presente dissertação ressaltam o uso do método de DCCM em contextos similares de análise de dados de simulação de DM, exemplificando modos de aplicação e contribuições do uso desta abordagem. Tendo em vista o amplo uso da técnica por trabalhos que utilizam esta estratégia, percebe-se a relevância da mesma no contexto da exploração de dados de Bioinformática Estrutural.

A análise visual e matemática desta matriz de adjacência de DCCM permite mapear as regiões da proteína cujos movimentos estão coordenados de maneira síncrona (Cavatão et al., 2024). Observa-se que valores próximos à diagonal principal indicam correlações entre resíduos que compõem um domínio da proteína (Cavatão et al., 2024), sugerindo que resíduos dentro da mesma unidade estrutural se movem em conjunto (Cavatão et al., 2024). Por outro lado, regiões distantes da diagonal que exibem anti-correlações fortes, sugerem movimentos independentes. A DCCM, sendo baseada na covariância, é inerentemente limitada à detecção de correlações lineares (Lange; Grubmüller, 2006; David; Jacobs, 2014). Isto significa que correlações não lineares ou movimentos perfeitamente coordenados em direções perpendiculares (ortogonais) podem resultar em uma correlação nula, subestimando o movimento correlacionado real (Lange; Grubmüller, 2006). A identificação de movimentos fortemente anti-correlacionados fora da diagonal é frequentemente utilizada para inferir importantes vias de comunicação alostérica ou identificar áreas funcionais, como sítios de ligação, relevantes para o comportamento dinâmico da proteína.

2.2 Computação

Para compreender o desenvolvimento deste protótipo funcional de ferramenta é necessário conhecer fundamentos vindos da computação na utilização de representações dos dados. Abordar a transformação realizada no dado provido pela área do conhecimento de domínio específico é essencial para o entendimento da ideia proposta por esta pesquisa, esta transformação é caracterizada pelo cálculo realizado nas trajetórias de DM. Este dado de variáveis que são coordenadas cartesianas que se comportam de maneira dinâmica em

relação ao tempo carregam informações topológicas referente ao seu formato.

2.2.1 Teoria de Grafos e Redes Dinâmicas no contexto da Bioinformática Estrutural

A teoria de grafos é capaz de melhorar a compreensão de fenômenos Biológicos complexos (Patel; Sinha; Palermo, 2024), auxiliando na interpretação de dinâmicas conformacionais que estas estruturas tridimensionais adotam. A representação adotada considera os átomos como nodos e as interações entre os mesmos como a arestas da rede (Patel; Sinha; Palermo, 2024), técnicas para perceber diferentes características destes grafos como centralidade, conectividade ou modularidade são importantes para a compreensão do comportamento de sistemas moleculares (Patel; Sinha; Palermo, 2024), embora estas técnicas não sejam adotadas nesta pesquisa.

Uma das representações adotadas para acomodar a informação sobre uma rede é a representação por matriz de adjacência (Wilson, 1996), onde a linha desta matriz condiz com um nodo do grafo e a coluna com outro nodo, podendo haver relação própria. A célula desta matriz condiz com a aresta entre os nodos, esta relação pode ser ponderada ou não. Neste caso fazemos o uso do modelo em que a rede é ponderada e a conexão não possui direção, isto é, o grafo pode ser representado por uma matriz simétrica.

Um método comum para análise é o da Correlação Cruzada, que consiste no cálculo da correlação de *Pearson* entre as flutuações atômicas dos $C\alpha$'s em relação as suas médias, é o método de DCCM. Nesta abordagem de Matriz Cúbica, devido à utilização da representação por matriz de adjacência, a 'aresta' da rede dinâmica não é desenhada como uma linha, mas sim como um ponto (*voxel*) cuja cor representa o peso da conexão (correlação). As arestas são ponderadas de -1 à 1 de acordo com a correlação presente entre estes nodos da rede. Considerando a característica de que trajetória de dinâmica molecular progride no tempo, é possível modelar este problema considerando diferentes momentos da simulação, segmentando o tempo e trazendo a interpretação vinda da teoria de grafos de redes dinâmicas (Gohnert et al., 2015) para este dado. Estas redes dinâmicas podem ser utilizadas para a visualização de dados (Beck et al., 2014).

Então a mescla entre estes conceitos é a base deste trabalho, abordar trajetórias de DM utilizando a representação de matrizes de adjacências traz uma perspectiva computacional para estes dados oriundos de uma área específica. O trabalho desenvolvido para demonstrar o estado da arte no estudo de visualização de redes dinâmicas feito por Kale, Sun e Papka (2023), produz uma perspectiva na visualização deste tipo de dado, onde,

aqui esta estratégia é adotada, assim como é adotada em dados provindos de outros contextos (Bach et al., 2017) como a correlação de dados sobre atividade de função cerebral (Bach; Pietriga; Fekete, 2014). Como aqui o objetivo é evidenciar as correlações lineares do movimento adotado por proteínas, isto explora a possibilidade da aplicação desta técnica em um novo contexto.

2.2.2 Técnicas de Visualização de Dados

Para a visualização dos dados, a estrutura adotada por este trabalho para a incorporação das informações de DCCM segue a taxonomia elementar de operações que consideram o espaço-tempo através de cubos proposta por Bach et al. (2017), esta técnica aqui se torna uma aplicação *web* que demonstra, de maneira prática, a utilização destes conceitos para explorar dados provindos de análises realizadas em simulação de DM. Bach et al. (2017) discute as vantagens e limitações do uso desta técnica, facilitando a descrição, visão crítica sobre os problemas abordados e a comparação de diferentes tarefas aplicadas em dados temporais com a finalidade de produzir uma visualização. A figura 2.1 exemplifica esta visualização que utiliza cubos no espaço tempo para visualizar o dado, em seguida são descritas as operações selecionadas de Bach et al. (2017) para o desenvolvimento da presente pesquisa. A escolha desta abordagem como guia do desenvolvimento do protótipo se faz por conta da possibilidade da exploração do tempo como uma dimensão adicional na representação do dado modelado utilizando matrizes de adjacência.

É através desta percepção que se identificou as necessidades do trabalho proposto, moldando o desenvolvimento do protótipo de ferramenta e propondo o desafio de trabalhar com dados de alta complexidade. O trabalho dispõe de diferentes abordagens para tratar cubos dispostos no espaço tempo. Neste contexto, a taxonomia representa a divisão das classes principais (Bach et al., 2017) de visualização de dados, sendo elas: extração da informação, achatamento da informação, transformação geométrica e transformação do conteúdo. Serão abordadas apenas aquelas classes a serem utilizadas no desenvolvimento prático da exploração proposta. Nas operações de extração é utilizada a extração de pontos e extração de um plano do volume através de corte ortogonal ao tempo. Nas operações de transformação geométrica, é abordada a translação no tempo. Na classe de transformação do conteúdo se utiliza as técnicas de rotulamento e filtragem dos dados.

A operação de translação no tempo dispõem ao visualizador mais uma maneira de

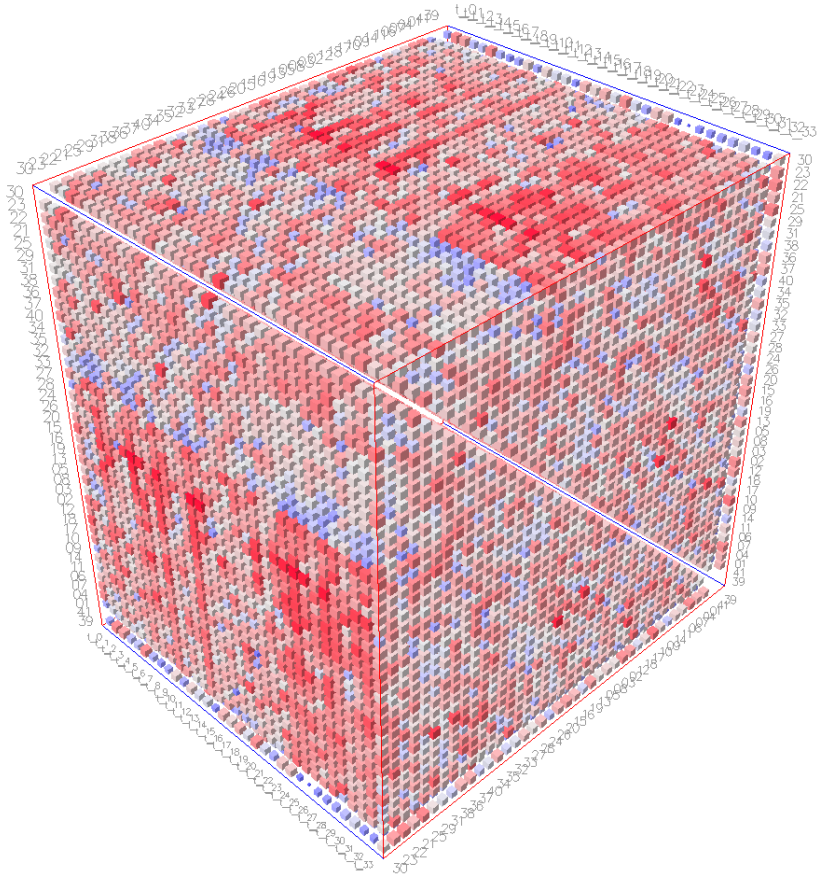

Figura 2.1 – Representação visual de Matrizes Cúbicas.
Disposição de figura que representa uma visualização de dados que utiliza a técnica de Matrizes Cúbicas.
Fonte: Adaptado de (Bach, 2016)

interagir com o dado, esta possibilidade permite que o observador perceba características e padrões a partir, também, de um vínculo visual entre os pontos vizinhos em relação ao tempo. A comparação entre os diferentes momentos da Matriz Cúbica também pode trazer uma relação de similaridade dos estados adotados por aqueles dados.

O rotulamento dos dados traz informações sobre o mesmo de maneira mais detalhada, tendo em vista que a visualização é tridimensional, o detalhamento específicos dos pontos agrega no entendimento do contexto que aquele se encontra. Isto informa o usuário de maneira detalhada sobre os pontos de interesse que ele pode ter por elementos específicos do dado.

Na filtragem do dado encontra-se uma maneira de diminuir a informação apresentada, com a finalidade de mitigar a sobrecarga visual da visualização e também de focar em diferentes tipos de informações que, após aplicado o filtro condicional, possa ser de interesse do observador. Como demonstrado pela figura 2.2

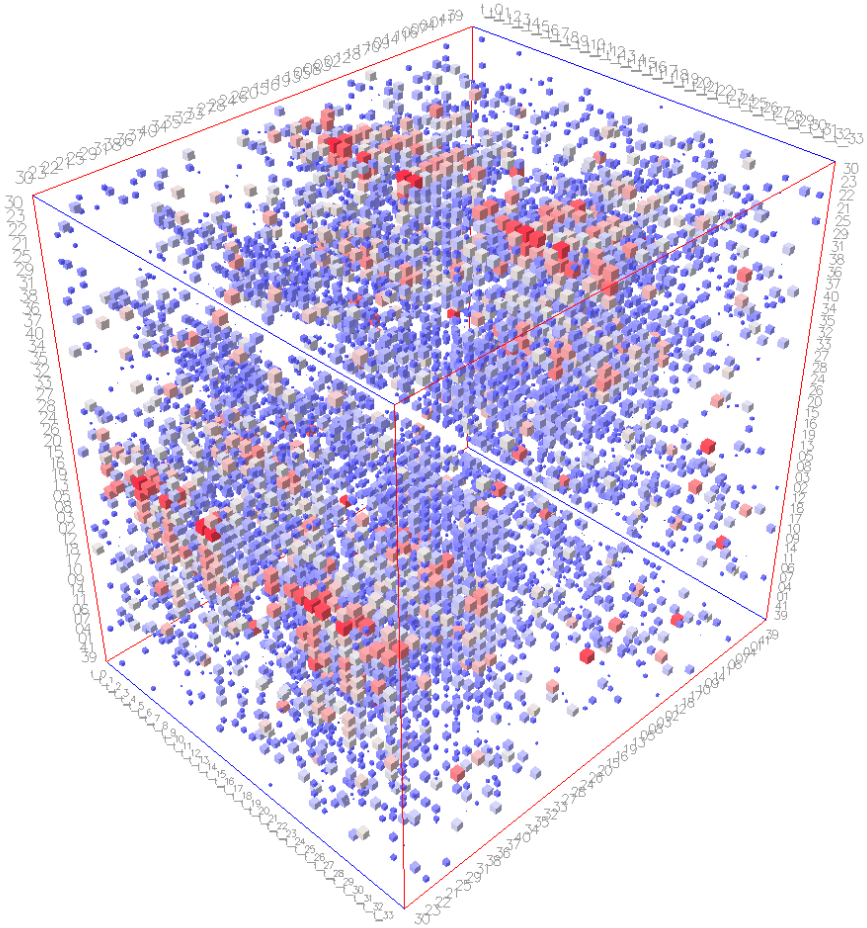

Figura 2.2 – Representação visual de Matrizes Cúbicas após a filtragem dos dados.
Disposição de figura que representa uma visualização de dados que utiliza a técnica de Matrizes Cúbicas, aplicando um filtro que destaca pontos de interesse.
Fonte: Adaptado de (Bach, 2016)

É importante ressaltar as limitações desta técnica, a estrutura discutida por Bach et al. (2017) é fundamentalmente um modelo descritivo e conceitual, servindo como uma ferramenta de pensamento para o desenvolvimento de aplicações, sem se comprometer em descrever a completude da concepção visual descrita. Ele foca na abstração das operações, não oferecendo especificidades sobre implementação técnica, a construção dos dados vindos do domínio específico do conhecimento ou a definição completa da implementação da interatividade do usuário com as ferramentas desenvolvidas. Não estabelece uma conexão direta entre as técnicas apresentadas e as tarefas ou necessidades dos usuários. Além disso, sua abrangência é limitada ao assumir meios de exibição tradicionais, estas que são abordadas por (Hong et al., 2024), e não contempla todas as possíveis complexidades e dimensionalidades dos dados temporais, sendo, portanto, uma representação intencionalmente incompleta do vasto *design space* de visualização.

Utilizando estas operações de maneira composta (Bach et al., 2017) podemos criar um ambiente interativo que permite o usuário da ferramenta explorar os dados de maneira eficiente. O que conversa com a metodologia desenvolvida por Hong et al. (2024), que descreve a área considerando a interação do usuário com a aplicação de maneira mais aprofundada.

No trabalho proposto por Hong et al. (2024), este que busca esclarecer a ideia da junção de informações de visualizações que utilizam diferente número de dimensões, por exemplo, permitindo a mescla entre um ambiente 3D e um ambiente 2D. Este mapeia o uso de diferentes maneiras de representação visual de dados que utilizam técnicas semelhantes a técnica desenvolvida por Bach (2016). Este mapeamento, realizado recentemente, guia a utilização de técnicas de visualização de dados, trazendo a superfície o padrão de *design* presente nesta pesquisa.

O artigo propõe um *design space* para classificar como as representações visuais bidimensionais e tridimensionais são combinadas e conectadas. A taxonomia se organiza em torno de três dimensões principais, investigando o "PORQUÊ", "ONDE"e "COMO"essas conexões são feitas.

A primeira dimensão é o PORQUÊ, ou motivação, detalha a razão ou o objetivo de se conectar as representações. Isso inclui a suplementação dos dados, onde as visões bidimensionais e tridimensionais fornecem informações diferentes sobre o mesmo objeto (como uma estrutura tridimensional e um gráfico bidimensional de dados abstratos). Aborda também a abstração, onde uma visão serve como uma simplificação da outra (usando técnicas como projeção, planificação ou fatiamento). Também disserta sobre o Controle, onde uma representação é usada primariamente para manipular a outra.

A segunda dimensão é ONDE (Ambiente de exibição), referindo-se ao ambiente de *hardware* e exibição onde a visualização é apresentada. As categorias incluem o *Desktop* (monitores tradicionais com mouse e teclado), a *Mixed Reality* (ambientes imersivos como Realidade Virtual ou Realidade Aumentada), a *Touch Screen* (dispositivos como *tablets*) e os Sistemas Tangíveis (uso de objetos físicos para representar ou controlar dados).

A terceira dimensão é COMO (*Layout* e Enfoque), que descreve como as representações são de fato ligadas. Ela se divide em *Layout* (arranjo espacial) e Abordagem (método de conexão interativa). O *Layout* pode ser Justaposto (lado a lado) ou Composto (ocupando o mesmo espaço, seja de forma Substituta, Sobreposta ou Embutida). A Abordagem descreve as técnicas de ligação: Conectado Visualmente (usando Cor, Posição, Forma ou Guias), conectado interativamente (com controle bidimensional para manipula-

ção de dados tridimensionais, tridimensional para manipulação de dados bidimensionais ou uma abordagem que considere ambos os tipos de controle) ou animado (usando Transformação ou Modificação do dado).

A importância desta outra taxonomia de visualização é de que ela é essencial para a construção de uma aplicação interativa, e estes conceitos se mesclam com conceitos presentes em Bach et al. (2017) que também descreve a aplicação de uma maneira de compreender o universo da Visualização Interativa de dados. Estes conceitos são aplicados durante o desenvolvimento do protótipo da ferramenta, e são fundamentais para o entendimento da presente pesquisa, embora a taxonomia escolhida seja a desenvolvida por Bach (2014).

3 TRABALHOS RELACIONADOS

Os trabalhos relacionados abordam dois pontos específicos que buscam evidenciar questões importantes sobre a modelagem do problema, sendo eles, a utilização comum do método de DCCM em trabalhos realizados na área da Bioinformática Estrutural e a utilização de análises aplicadas a redes dinâmicas. Do ponto de vista da análise de redes dinâmicas, se evidencia a utilização ou não de técnicas visuais.

3.1 Bioinformática Estrutural

O trabalho de Wang et al. (2022) utiliza uma abordagem computacional, centrada em simulações de DM, para comparar o comportamento dinâmico de uma proteína em seu estado nativo (*Wild Type* (WT)) com uma versão que contém mutação. Um dos métodos de análise pós-simulação é o DCCM. O estudo demonstra como a análise de DCCM é usada para identificar mudanças nos movimentos correlacionados e na flexibilidade geral da proteína, revelando como uma alteração pontual pode propagar efeitos dinâmicos por toda a estrutura, afetando regiões distantes.

Este trabalho é relevante para esta pesquisa, pois é um exemplo direto da metodologia que está sendo investigada. Ele aplica a análise de DCCM para o fim de comparar um sistema WT com um que possui mutação pontual (Wang et al., 2022), característica presente nos dados avaliados no presente estudo, com o intuito de entender os efeitos dinâmicos entre partes distantes da proteína. Isto demonstra necessidade de ferramentas visuais avançadas em comparação ao uso comum do método aplicado, como a desenvolvida.

O trabalho de Parida, Paul e Chakravorty (2020) usa simulações de DM para avaliar como diferentes moléculas interagem com as proteínas-alvo. Para isso, utilizam um conjunto de análises, incluindo o DCCM. O ponto central é que a pesquisa citada usa o DCCM para comparar a dinâmica da proteína simuladas com e sem o ligante (ex. fármaco). Isso permite que identifiquem quais moléculas estabilizam a proteína e quais causam perturbações em sua rede de movimentos internos (Parida; Paul; Chakravorty, 2020). O artigo aplica a análise de DCCM da forma comparativa, assim como o trabalho anterior, que também aplica o método para comparar simulações com diferentes características.

O trabalho de Okamoto e Ando (2024) explora a resposta dinâmica de uma proteína receptora quando simulada através da DM com um ligante. Como não havia uma

estrutura experimental, os investigadores utilizaram um modelo gerado por Inteligência Artificial (*AlphaFold2*). A análise principal se concentra em como a formação de uma ligação de hidrogênio específica entre o ligante e a proteína desencadeia mudanças dinâmicas. Para isso, utilizam DCCM e propõem uma nova métrica, a Correlação Cruzada Dinâmica Dependente do Tempo (Time Dependent Dynamic Cross Correlation - TDDCC) (Okamoto; Ando, 2024), para rastrear como os sinais dinâmicos se propagam. A principal descoberta é a identificação de um caminho de transferência de correlação específico, mostrando como a perturbação da ligação é transferida através de uma rede de resíduos até um local funcionalmente importante da proteína.

Este artigo se alinha ao conceito central do trabalho aqui proposto. Utilizando a análise comparativa de DCCM (no caso, apo vs. holo, sem e com ligante) para compreender esta rede de comunicação (Okamoto; Ando, 2024). O trabalho ainda pretendeu mapear a propagação do sinal ao longo do tempo (TDDCC). Isso reforça diretamente a necessidade da ferramenta de visualização de dados que considera o tempo.

O artigo desenvolvido por Kasahara, Fukuda e Nakamura (2014) também é um exemplo que busca explorar as limitações que o método de DCCM possui. Propõem um método alternativo, *multi-modal Dynamic Cross Correlation*(mDCC), que busca perceber correlações presentes no momento de transição de conformações adotadas pelo complexo molecular estudado. Esta abordagem, em partes, possui uma motivação similar à proposta neste trabalho, pois também busca evidenciar correlações que possam ter sido ocultadas pelo método padrão. Embora a abordagem aqui presente foque no uso de visualização do dado e não em um novo método matemático de análise.

O trabalho de Hünenberger, Mark e Gunsteren (1995) é um estudo clássico e fundamental que investiga a confiabilidade da própria análise de DCCM. Em vez de utilizar o DCCM para analisar um sistema, analisam o método em si. A principal conclusão é um grande alerta: os mapas de correlação não convergem rapidamente. Demonstram que, em simulações curtas (ex: 200 picossegundos), os mapas de correlação se tornam inconsistentes, mudando drasticamente o resultado do método. Os autores mostram que é preciso centenas de picossegundos, chegando a casa dos nanossegundos, para que os padrões de correlação comecem a se estabilizar. Além disso, alertam que o método de alinhamento da trajetória pode, por si só, introduzir correlações falsas.

Este artigo é importante para a investigação realizada no presente trabalho pois serve como uma grande ressalva metodológica. Ele questiona diretamente a validade de comparar mapas de diferentes simulações, informando que se as simulações não forem

longas o suficiente os mapas de correlação podem não convergir à uma representação suficientemente boa da simulação. E na investigação proposta, os dados possuem tempo de simulação superior à este limite proporcionado por Hünenberger, Mark e Gunsteren (1995). A descoberta de que os mapas podem não convergir traz uma perspectiva de que o método desenvolvido aqui consegue identificar visualmente esta convergência nos primeiros momentos da simulação.

O trabalho de Dash et al. (2022) utiliza uma abordagem computacional para investigar como mutações pontuais em uma proteína específica estudada pelo trabalho pode levar a doenças neurodegenerativas. O método central é a DM, e o DCCM é utilizado de maneira comparativa. Simularam a proteína nativa e três mutantes de alto risco e, então, calcularam os DCCM para cada um dos sistemas. Ao comparar os mapas dos mutantes com o mapa do WT identificaram como as mutações alteram os padrões de movimento. Descobriram, por exemplo, que os mutantes aumentam os movimentos anti-correlacionados entre parte funcional da proteína, o que desestabiliza a região da mesma responsável por reconhecer e se ligar a outras moléculas. Os dados utilizados pela abordagem se assemelham aos dados utilizados na presente pesquisa pois também é feita a comparação da proteína nativa com sistemas que possuem mutações, embora o complexo estudado seja completamente diferente.

O trabalho de (Cavatão et al., 2024) utiliza simulações de Dinâmica Molecular (MD) para investigar como duas mutações específicas afetam a função da proteína *Melanocortin*-1 *Receptor* (MC1R). A análise de DCCM é utilizada no estudo, para uma comparação robusta entre seis sistemas diferentes: a proteína nativa e dois outros mutantes, cada uma simulada com e sem o ligante. Ao comparar os mapas de correlação gerados, os autores identificam como as mutações alteram os padrões de movimentos correlacionados e anti-correlacionados. Descobrem então que os mutantes alteram a dinâmica de transição entre os estados ativo e inativo do sistema, um dos sistemas que possui mutação exibe um perfil de correlação que se assemelha o estado inativo do WT, mesmo quando o ligante está presente, explicando assim a perda de atividade com a presença desta mutação específica.

Este artigo é excepcionalmente relevante para a presente investigação, pois ele estuda exatamente os sistemas que são utilizados na presente pesquisa: o MC1R nativo (WT) e as mutações pontuais. O trabalho fornece estes dados para a realização da análise das correlações que acontecem nestes sistemas. A presença dos mapas de correlação cruzadas no trabalho demonstram um uso comum do método, permitindo a exploração

deste método de visualização alternativo.

3.2 Visualização da Informação

A exploração de redes e redes dinâmicas através da visualização da informação é realizada em múltiplos contextos, assim como visualização de dados de um modo geral, nesta seção estão presentes trabalhos que buscam discutir estes conceitos.

O trabalho realizado por Gohnert et al. (2015) apresenta uma técnica de visualização tridimensional de redes dinâmicas, denominada 3D DynNetVis. Esta técnica é responsável por mostrar fatias de tempo de redes dinâmicas utilizando um *layout* de grafos bidimensional, ela empilha estas fatias em uma terceira dimensão, buscando demonstrar a evolução da rede, ilustrada pela figura 3.1. É proposto uma maneira de visualizar estas redes que permite a inspeção e análise em cada fatia e também em diferentes fatias conforme a mudança de estados do grafo dinâmico, utilizando a filtragem de nodos e a aparição dos mesmos baseada em propriedades. A ideia, assim como no presente trabalho, implementa o mantra citado anteriormente e definido por Shneiderman (1996), pretendendo permitir a visão geral das redes assim como ampliação filtragem e detalhes sob demanda. Partindo desta definição é possível compreender a similaridade do trabalho com o que é feito no desenvolvimento da presente protótipo. Outro ponto que se assemelha à pesquisa proposta por Gohnert et al. (2015) é que a aplicação também é disponibilizada na *Web* e que foi desenvolvida utilizando a mesma biblioteca JavaScript chamada *Three.js*, isto justifica as escolhas realizadas no sentido de utilizar ferramentas desenvolvidas para o campo do desenvolvimento *Web* em um contexto cientifico de disponibilidade de ferramentas e divulgação cientifica.

Bach et al. (2015) introduz *MultiPiles*, uma visualização que explora redes ponderadas e densas, que é baseado no empilhamento de matrizes de adjacência. O trabalho foi inicialmente desenvolvido para auxiliar neurocientistas a investigar mudanças na conectividade de redes relacionadas a atividade cerebral em centenas de instantes diferentes (Bach et al., 2015). A técnica adota estratégias para o agrupamento, que respeita a disposição temporal, destas matrizes de adjacência através da similaridade entre elas, isto permite a redução de quadros que são visualizados simultaneamente, reduzindo a sobrecarga de informações que podem ser redundantes. A implementação da ferramenta envolve profissionais da área que estuda a conectividade de redes que representam funções cerebrais, garantindo robustez no protótipo proposto. O autor, que também desenvolve a

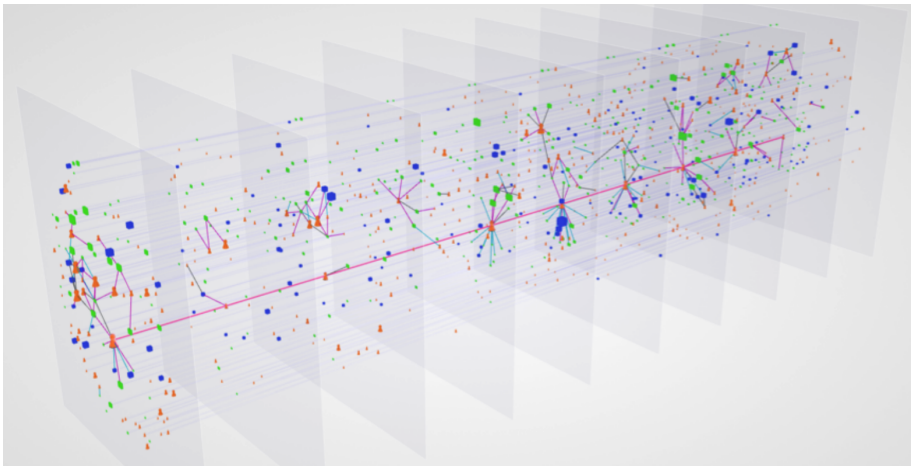

Figura 3.1 – Representação de 3D *DynNetVis*.
Visão geral em 3D da rede com arestas e nodos referentes aos dados.
Fonte: Adaptado de Gohnert et al. (2015).

ideia de Matrizes Cúbicas para a visualização (Bach; Pietriga; Fekete, 2014), investiga na técnica de *MultiPiles* a aplicação de uma visualização aplicada a um domínio específico do conhecimento enquanto propõem um novo método de análise de redes dinâmicas.

Wen et al. (2023) desenvolve *DiffSeer*, uma abordagem nova para a visualização de redes dinâmicas ponderadas. A abordagem foca na visualização da diferença entre pesos da rede em fatias de tempo adjacentes, como demonstrado pela figura 3.2, que utiliza estratégia de reorganização de nodos baseada em otimização para agrupar evoluções similares nestas redes. A visualização agrega a informação das diferenças entre pesos da rede e dispõem de maneira à oferecer uma visão geral da comparação entre dois momentos da rede e então permite o detalhamento das matrizes que compõem esta diferença. A ferramenta é desenvolvida de maneira que seja generalizável e possa ser aplicada em diversos contextos, faz isto através do estudo de caso de dois tipos de redes, do impacto do COVID-19 na rede de correlações de diferente setores no mercado de ações da China e também de uma rede de interações social online entre dois times de *Rugby*. É mostrado no trabalho a usabilidade desta ferramenta desenvolvida. Isto se relaciona com o trabalho proposto pois demonstra a possibilidade de adoção de um método de visualização de redes dinâmica em diferentes contexto de análise.

MatrixExplorer, desenvolvido por Henry e Fekete (2006), apresenta uma maneira de acoplar duas representações, a representação de grafos através de nodos e arestas visuais em conjunto com a representação de matrizes da mesma rede de maneira sincronizada, como demonstrado pela figura 3.3. A junção dos dois tipos de visualização em uma única aplicação possibilita o entendimento dos padrões visuais presentes na matriz com a per-

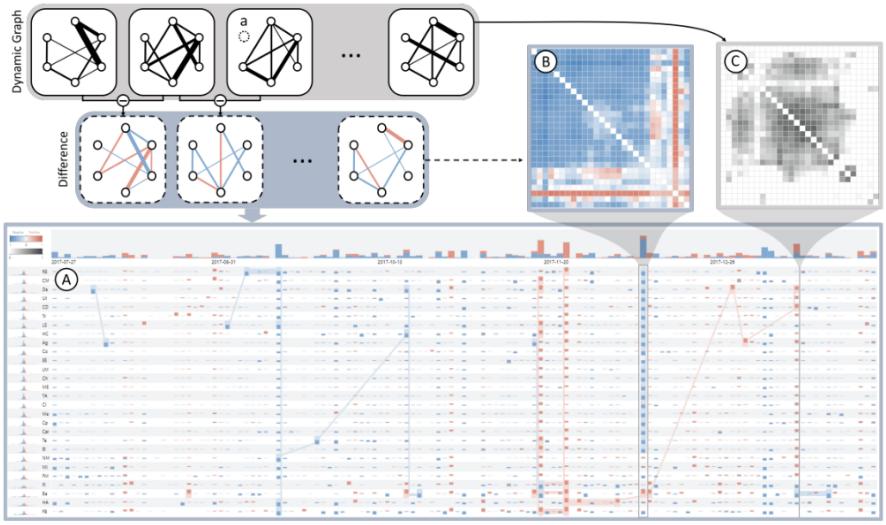

Figura 3.2 – Representação de *DiffSeer*.
Visão geral do *DiffSeer*, uma abordagem focada na visualização de diferenças entre segmentos de tempo adjacentes para analisar a evolução de grafos dinâmicos. O método utiliza um *design* matricial para resumir mudanças e destacar padrões estruturais ao longo do tempo. Onde (A) representa a visão geral que sumariza as diferenças visuais entre estados da rede. Matrizes de detalhe (B, C) para inspeção interativa do grafo sob demanda.
Fonte: Adaptado de Wen et al. (2023).

cepção da conectividade da rede que a representação por nodos e arestas dispõem. A técnica permite a interação do usuário com a rede, permitindo a ordenação das linhas e colunas da matriz com a finalidade de informar padrões presentes na rede. Embora a exploração não aborde redes dinâmicas, o entendimento sobre os padrões presentes nas representações bidimensionais da matriz de adjacência visualizada é fundamental para compreender o que estes padrões significam sobre as interações da rede, de uma maneira interativa e automatizada.

Assim como o trabalho desenvolvido por Henry e Fekete (2006), Shu et al. (2025) apresenta técnica interativa para a explicação de padrões visuais na visualização de redes, buscando ajudar na extração de informações significativas as redes exploradas (Shu et al., 2025). A técnica permite a seleção de uma área arbitrária da matriz de adjacência e automaticamente reconhece padrões no dado. O trabalho realiza um estudo quantitativo e qualitativo com a interação de participantes na ferramenta de prova de conceito de visualização de dados chamada *Pattern Explainer*, comparando sua eficácia contra o uso de *cheat sheets* (Wang et al., 2020) e explicações textuais. Os resultados demonstraram que a abordagem interativa aumentou significativamente o aprendizado da compreensão de redes, permitindo que os usuários identificassem corretamente uma quantidade superior

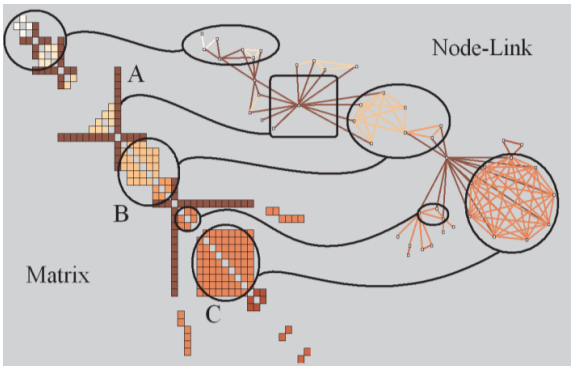

Figura 3.3 – Representação de *MatrixExplorer*.
Padrões visuais em representações matriciais e de nodos e arestas de redes sociais. (A) Representa a conexão entre diversas comunidades, (B) uma comunidade e (C) um clique (subgrafo completo).
Fonte: Adaptado de Henry e Fekete (2006).

de padrões topológicos em comparação aos métodos estáticos.

Um ponto importante a se levar em consideração no desenvolvimento de visualização de redes são as barreiras e limitações sobre esta exploração. Alkadi et al. (2023) desenvolve em seu estudo um esquema que define um fluxo de oito passos para a abordagem de exploração de redes através da Visualização da Informação, estes passos são, nesta ordem: Identificar conceitos do domínio, definir a rede, formatar o dado, importar o dado, explorar a visualização, perceber padrões visuais, interpretar os padrões visuais e interpretar conceitos da rede. Este trabalho tem o propósito de descrever um estudo que auxilia o *design* de ferramentas de visualização. O estudo estruturou-se em torno de um curso sobre visualização de redes, focado na ferramenta *Vistorian*, onde os participantes foram incentivados a utilizar seus próprios dados. O currículo evoluiu da preparação e modelagem dos dados para a exploração visual utilizando diagramas de nós e arestas, matrizes de adjacência, linhas do tempo e mapas geográficos (Alkadi et al., 2023). Este trabalho identifica possíveis problemas, e auxilia na tomada de decisões no momento de conceptualização do protótipo de ferramenta desenvolvida pelo presente trabalho.

4 DESENVOLVIMENTO DE ABORDAGEM INTERATIVA PARA ANÁLISE DE DCCM

A visualização interativa faz necessária a representação do dado, encontrar uma representação que faça com que o diálogo entre a Visualização de Dados e a Bioinformática Estrutural aconteça com sinergia é uma tarefa desafiadora que demanda a atenção do desenvolvedor do método. Neste caso o uso de um método aplicado à trajetórias de Dinâmica Molecular que pode ser interpretado como redes dinâmicas (Patel; Sinha; Palermo, 2024) é a maneira com que se busca modelar o problema, permitindo a aplicação da técnica *MatrixCube* de Visualização de Dados proposta por Bach, Pietriga e Fekete (2014).

A ideia principal da abordagem interativa é utilizar a visualização de dados tridimensional para explorar o eixo do tempo presente no método de DCCM, que é um método utilizado para a análise de simulação de Dinâmica Molecular (Hünenberger; Mark; Gunsteren, 1995; Kasahara; Fukuda; Nakamura, 2014; Dash et al., 2022; Okamoto; Ando, 2024; Parida; Paul; Chakravorty, 2020; Wang et al., 2022; Cavatão et al., 2024). O objetivo é também possibilitar a exploração deste método de uma maneira detalhada. Seguindo o Mantra de Busca de Informações Visuais (*Visual Information-Seeking Mantra*), definido por Shneiderman (1996):

> *"Overview first, zoom and filter, then details-on-demand."* (Shneiderman, 1996)

Este mantra guia o desenvolvimento de ferramentas de visualização, definindo a base para a exploração visual da informação. Primeiro uma visão geral do dado, seguido por operações interativas como ampliação e filtragem, e então os detalhes do mesmo. Isto define um fluxo importante e fundamental que direciona a elaboração da interface gráfica até os dias atuais (Cui, 2019).

O desenvolvimento deste protótipo de ferramenta de análise é realizado de maneira desacoplada, onde um código é responsável pela computação do algoritmo que processa os dados de trajetória e calcula o método e o outro pela visualização deste dado. A técnica de visualização proposta deste dado possui inspirações que se complementam para a justificativa do desenvolvimento, tendo em vista que neste trabalho aplicamos ao domínio específico aquela visualização de Matrizes Cúbicas (Bach et al., 2017) definidas por Bach (2016). A decisão de utilizar esta técnica de visualização é feita a partir da disposição desta abordagem de explorar o tempo na dimensão adicional em relação aos

dados, permitindo a percepção da mudança de estados desta rede dinâmica representada por matrizes de adjacência. A modelagem do problema envolve técnicas exploradas por trabalhos da Bioinformática Estrutural e Computação, onde a Bioinformática Estrutural abrange a exploração de complexos moleculares através do método de DCCM de análise de trajetória (Hünenberger; Mark; Gunsteren, 1995; Kasahara; Fukuda; Nakamura, 2014; Dash et al., 2022; Okamoto; Ando, 2024; Parida; Paul; Chakravorty, 2020; Wang et al., 2022; Cavatão et al., 2024) e a Computação engloba tanto o uso de grafos/matrizes de adjacência (Beck et al., 2014) para a representação de redes dinâmicas quanto o uso de técnicas de visualização de dados de maneira tridimensional (Andrienko et al., 2010; Brath, 2014; Beck et al., 2014; Bach et al., 2014; Hong et al., 2024; Shneiderman, 1996).

Os módulos da aplicação se dividem pelo processamento e visualização dos dados através da renderização, o processamento é realizado em *Python* utilizando a biblioteca *Mdtraj* (McGibbon et al., 2015) e a renderização é desenvolvida em *Javascript* utilizando a biblioteca Three.js (Cabello, 2010). A comunicação entre estes dois componentes do protótipo funcional é através de um arquivo binário que armazena os dados das matrizes de correlação e também um cabeçalho com informações relevantes, desta maneira é possível passar a informação de uma maneira compacta entre os dois componentes do protótipo.

As seguintes seções detalham e exemplificam o que foi discutido a cima. Primeiramente há uma descrição de como os dados foram tratados e então as etapas de processamento do método e de visualização dos dados são descritas, assim como a característica de reprodutibilidade e disponibilidade do protótipo de ferramenta.

4.1 Dados

Os dados utilizados nesta pesquisa são oriundos do artigo desenvolvido por Cavatão et al. (2024) que visa trazer luz aos efeitos de duas mutações distintas no mecanismo de ação da MC1R, representados de maneira ilustrativa pela figura 4.1 presente no trabalho realizado por Cavatão et al. (2024). Este mecanismo não é explorado pelo trabalho realizado aqui, o interesse desta pesquisa é apenas evidenciar as correlações considerando o tempo, visando dar foco àquelas que são ocultas na média considerada no método comumente utilizado pelas pesquisas realizadas na Bioinformática Estrutural (Hünenberger; Mark; Gunsteren, 1995; Kasahara; Fukuda; Nakamura, 2014; Dash et al., 2022; Okamoto; Ando, 2024; Parida; Paul; Chakravorty, 2020; Wang et al., 2022; Cavatão et al.,

2024).

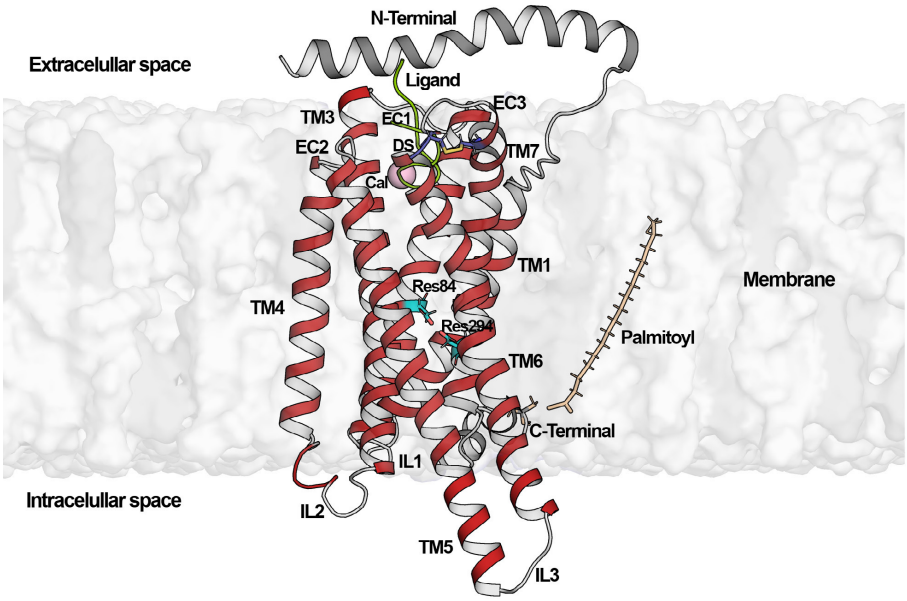

Figura 4.1 – Representação visual da MC1R.
Representação visual da MC1R, identificando domínios específicos desta proteína, estes que determinam fatores de correlação. Imagem ilustrativa de apoio para a compreensão do formato tridimensional do dado.
Fonte: Adaptado de Cavatão et al. (2024).

O *dataset* utilizado é composto por três complexos moleculares, considerando duas mutações e WT. Cada complexo molecular possui duas condições: uma considera apenas a estrutura da proteína e a outra a proteína com o ligante. Para cada condição do complexo, possuem ainda cinco réplicas de simulação de 1μs (microssegundos), 10000 *frames* cada. Totalizando 2 (condições) * 3 (complexos) * 5 (réplicas) = 30 simulações (300000 *frames*). Estes dados estão disponíveis no endereço <https://sbcb.inf.ufrgs.br/data/dbdm/MC1R/exp01/>. O uso destes dados se faz pela disponibilidade dos mesmos e com um caráter exploratório sobre a utilização do método de DCCM de maneira segmentada.

No trabalho de Cavatão et al. (2024) o DCCM gerado foi realizado a partir do procedimento usual do GROMACS (Abraham et al., 2015), que consiste em: *gmx trjcat* para concatenar as réplicas do sistema, *gmx trjconv* para gerar um arquivo de coordenadas inicial, *gmx covar* para o cálculo da matriz covariância e posterior normalização dessa matriz utilizando *MDAnalysis* (Michaud-Agrawal et al., 2011; Gowers et al., 2016). O resultado final é um DCCM calculado a partir da concatenação das réplicas referentes a um complexo molecular.

4.1.1 Pré-processamento

É necessário o pré-processamento dos dados para que seja possível a utilização do método proposto. Esta etapa remove as moléculas de água presente na simulação, também seleciona apenas os átomos necessários ($C\alpha$) para a visualização proposta. Os comandos responsáveis por esta parte do processamento são executados utilizando a ferramenta GROMACS (Abraham et al., 2015) e estão disponíveis no arquivo *README.md* presente na pasta responsável por armazenar os arquivos que contém o código do processamento do dado, os arquivos necessários para esta etapa do processamento são os de trajetória (*.trr*) e *index* (*.ndx*) da simulação sendo analisada, estes são resultado do trabalho realizado na investigação da MC1R (Cavatão et al., 2024), os mesmos podem ser encontrados em Cavatão et al. (2024) e através do endereço supracitado. Como resultado do pré-processamento aplicado aos dados de simulação se tem novos arquivos que representam a trajetória dos $C\alpha$ presentes na proteína, sendo eles os arquivos *.xtc* e *.gro*. A partir destes arquivos é realizada a computação referente ao cálculo da correlação entre $C\alpha$ dos resíduos presentes na estrutura, é onde a abordagem de segmentar o DCCM para expandir a possibilidade de exploração deste método é aplicada, visando fornecer os dados necessários para representar esta informação maneira visual.

Para a realização desta etapa foi desenvolvido um código em *Python* que escreve um arquivo *.batch*, este arquivo é utilizado pela ferramenta *Slurm* que se responsabiliza por agendar os trabalhos definidos para cada uma das réplicas dos complexos moleculares. Estes trabalhos são submetidos através de um *script* de linha de comando para a finalização do pré-processamento. Permitindo então a utilização destes dados para a próxima etapa que fica responsável pelo cálculo do método.

4.2 Cálculo da correlação fatiada

A parte do protótipo da ferramenta desenvolvida em *Python* é responsável pelo processamento dos dados resultantes da simulação, fazendo o uso da biblioteca *Mdtraj* que possibilita interpretar os arquivos de entrada produzidos pelo pré-processamento. Através destes arquivos é possível extrair as informações necessárias para a construção da visualização posteriormente. Este dado possui informação topológica da estrutura da macromolécula utilizada na análise, um dos arquivos é responsável por armazenar informações de trajetória e o outro contém informações relevantes sobre o sistema.

Os dados processados e utilizados neste trabalho estão armazenados e disponíveis através do endereço <https://linkly.link/2Rfez>. O algoritmo reconhece o número de sistemas e réplicas disponíveis em uma pasta específica que contenha os dados e então os mesmos são carregados para a memória utilizando função *processTrajectory*, esta função utiliza o pacote *MDtraj* (McGibbon et al., 2015). Esta biblioteca *Python* possui o papel de permitir a manipulação, nesta linguagem, de dados provenientes de simulação de Dinâmica Molecular de maneira otimizada. Nesta função é definido o tamanho das fatias adotadas pelo método.

A partir do *MDtraj*, é possível extrair as coordenadas relativas a cada estado adotado pela simulação e armazenar em uma lista. Nesta etapa, antes de qualquer segmentação, é realizado o alinhamento global da trajetória completa em relação à estrutura de referência (primeiro *frame*) utilizando a função *superpose* da biblioteca. Este procedimento elimina os movimentos de rotação e translação do sistema, garantindo que as correlações calculadas reflitam apenas a dinâmica interna da proteína. A partir disto, o algoritmo segmenta esta lista e aplica o método comum de DCCM para cada segmento. O código 4.1 exemplifica a aplicação do método, traduzindo a fórmula presente na fundamentação teórica para um código *Python* que utiliza a biblioteca *NumPy* (Harris et al., 2020):

Listing 4.1 – Tradução da fórmula de cálculo de DCCM para python.

```
def calculateDCCM(self, traj_slice: np.ndarray) -> np.ndarray:
    n_frames, n_atoms, _ = traj_slice.shape
    coords = np.asarray(traj_slice, dtype=np.float32)
    mean_coords = np.mean(coords, axis=0)
    fluctuations = coords - mean_coords
    cov_matrix = np.einsum('tij,tkj->ik', fluctuations,
       fluctuations) / n_frames
    diag = np.diag(cov_matrix)
    diag_sqrt = np.sqrt(diag + 1e-10)
    norm_matrix = np.outer(diag_sqrt, diag_sqrt)
    dccm_matrix_cpu = cov_matrix / norm_matrix
    return dccm_matrix_cpu
```

Para cada segmento selecionado da trajetória, é aplicado o método comum de DCCM, então a função *calculateDCCM* processa um segmento de trajetória para calcular o DCCM. Seguindo a ordem que o código apresenta, inicialmente, a função determina a posição média de cada átomo ao longo das conformações do segmento. Em seguida,

ela calcula os vetores de flutuação para cada átomo em cada conformação, subtraindo a posição média da posição dos instantes específicos. O passo central é o cálculo da matriz de covariância (utilizando *np.einsum* do NumPy), que mede o quanto os movimentos dos átomos estão coordenados da seguinte forma: a notação `'tij,tkj->ik'` define a regra de redução da matriz, ela orienta o algoritmo a multiplicar as flutuações do átomo (i) pelas do átomo (k), alinhando pelo tempo (t) e pelas coordenadas espaciais (j). A falta dos índices t e j na saída (`->ik`) indica uma soma sobre esses eixos, o que efetivamente calcula o produto escalar dos vetores de deslocamento acumulado ao longo de toda a fatia da trajetória, resultando na matriz de covariância bruta. Então, o código normaliza esta matriz, extraindo a variância de cada átomo (a diagonal da matriz de covariância), calcula o desvio padrão (a raiz quadrada da variância) e divide a covariância pelo produto dos desvios padrão dos dois átomos correspondentes, resultando na matriz DCCM final com valores entre -1 e 1.

A implementação deste algoritmo dispõe de uma alternativa desenvolvida com a biblioteca CuPy (Okuta et al., 2017), visando a aceleração utilizando GPU. Embora a execução em CPU seja suficiente para o estudo de caso atual — dado o número reduzido de átomos e conformações —, a complexidade quadrática do cálculo da matriz de covariância em relação ao número de resíduos impõe um limite prático de desempenho em sistemas com um número significativo de átomos, embora esta limitação não tenha sido explorada de maneira quantitativa. Esta implementação alternativa permite a escalabilidade da abordagem interativa, permitindo que trabalhos futuros analisem sistemas de maior escala ou trajetórias de longa duração que seriam inviáveis de processar em tempo eficiente utilizando apenas a CPU.

4.2.1 Armazenamento dos dados

Posteriormente, o dado gerado é armazenado de maneira otimizada, o código abaixo demonstra a estratégia utilizada para armazenar estas informações de maneira que não ocupe espaço de armazenamento excessivo, levando em consideração que esta matriz, por ser simétrica, possui dados redundantes. Esta abordagem permite a transferência destes arquivos de uma maneira mais eficaz, auxiliando o desenvolvedor a transferir os arquivos do módulo de processamento para o módulo de visualização.

Listing 4.2 – Armazenamento otimizado da matriz DCCM em formato binário.

```
indices_triu = np.triu_indices(num_atomos)
num_elementos_triangulo = len(indices_triu[0])
dados_compactados = np.zeros((num_fatias,
   num_elementos_triangulo), dtype=DTYPE)
for i in range(num_fatias):
    dados_compactados[i] = DCCM_slices[i][indices_triu]
output_filename = os.path.join(path, f'dccm_data_{slice_size}.
   bin')
with open(output_filename, 'wb') as f:
    tipo_dado_id = 1
    header = struct.pack('<III', num_fatias, num_atomos,
       tipo_dado_id)
    f.write(header)
    f.write(encoded_names)
    f.write(dados_compactados.tobytes())
```

Dado que as matrizes de correlação são simétricas (i.e., o valor na posição (i, j) é idêntico ao da (j, i)), os dados brutos possuem alta redundância. Para eliminar o armazenamento duplicado, o método extrai apenas os elementos do triângulo superior de cada matriz (utilizando a função *np.triu_indices* da biblioteca *NumPy/CuPy* que retorna tuplas com as posições do triângulo superior para este tamanho de matriz), isto lineariza a matriz colocando a informação em uma lista de uma dimensão. Estes valores não redundantes são, então, serializados e escritos em um formato binário compacto, reduzindo significativamente o espaço em disco necessário e o tempo necessário para escrita ou leitura dos dados. Este arquivo contém um *header* e um *payload*, no *header* é informado o número de fatias e átomos e no *payload* estão os nomes dos resíduos e correlações fatiadas.

4.3 Visualização

A visualização é desenvolvida em JavaScript, utilizando a biblioteca *Three js* (Cabello, 2010). Esta biblioteca permite a montagem de um ambiente gráfico tridimensional, onde é renderizada a cena que utiliza os dados gerados no módulo de processamento. A organização do código permite a lógica de construção dos componentes visuais de maneira isolada, utilizando o sistema de funções e classes da linguagem. Esta abordagem facilita o desenvolvimento do protótipo de ferramenta, permitindo a reconstrução da cena

quando necessário e também garantindo que a aplicação não ultrapasse os limites de memória da máquina responsável pelo processamento da renderização. A aplicação realiza a captura dos dados a partir do arquivo binário do lado do servidor (*server side*) em que a mesma é hospedada, enquanto processa a visualização utilizando a memória RAM da máquina que acessa a aplicação (*client side*). Realizar as operações desta maneira garante, em partes, que o ambiente em que a aplicação esta hospedada não sofra uma sobrecarga de processamento.

4.3.1 Leitura do arquivo binário

A classe *DccmFunctions*, presente no arquivo *DccmFunctions.js*, possui as funções que realizam o processamento dos dados, interpretando o arquivo binário. A função assíncrona *loadBinaryDCCM*, código 4.3, fica responsável por extrair do arquivo binário todas as informações necessárias para a construção da projeção do dado. Estas informações são, na ordem que são coletadas no código, número de segmentos, número de átomos da proteína, nome dos resíduos referentes ao $C\alpha$ presente no método e por fim as correlações segmentadas. O restante do código é responsável pelo cálculo dos espaçamentos (bytes) entre as informações no arquivo binário. É importante ressaltar que neste momento do processamento do arquivo binário é realizada a operação inversa para captação do triângulo superior e diagonal principal, resultando em uma reconstrução do dado para então permitir a sua interpretação e modelagem da visualização.

Listing 4.3 – Carregamento dos dados a partir do arquivo binário.

```
1   async loadBinaryDCCM(url) {
2       .
3       .
4       .
5       const headerView = new DataView(arrayBuffer, 0, 12);
6       const numSlices = headerView.getUint32(0, true);
7       const numAtoms = headerView.getUint32(4, true);
8       const namesOffset = 12;
9       const namesBlockSize = numAtoms * 4;
10      const residueNames = [];
11      const textDecoder = new TextDecoder('utf-8');
12      for (let i = 0; i < numAtoms; i++) {
```

```
13              const start = namesOffset + (i * 4);
14              const nameBytesView = new DataView(arrayBuffer,
                    start, 4);
15              const name = textDecoder.decode(nameBytesView).
                    replace(/\0/g, '');
16              residueNames.push(name);
17          }
18          const dataOffset = namesOffset + namesBlockSize;
19          let rawData;
20
21          const numElementsPerSlice = (numAtoms * (numAtoms + 1))
                / 2;
22          const getDCCMValue = (sliceIndex, i, j) => {
23              if (i > j) {
24                  [i, j] = [j, i];
25              }
26              const sliceOffset = sliceIndex * numElementsPerSlice
                    ;
27              const indexInTriangle = (numAtoms * i - (i * (i - 1)
                    ) / 2) + (j - i);
28              return rawData[sliceOffset + indexInTriangle];
29          };
30
31          const getSliceAsMatrix = (sliceIndex) => {
32              const matrix = Array(numAtoms).fill(0).map(() =>
                    Array(numAtoms).fill(0));
33              for (let i = 0; i < numAtoms; i++) {
34                  for (let j = i; j < numAtoms; j++) {
35                      const value = getDCCMValue(sliceIndex, i, j)
                            ;
36                      matrix[i][j] = value;
37                      matrix[j][i] = value;
38                  }
39              }
40              return matrix;
41          };
```

```
42
43          return {
44              numSlices,
45              numAtoms,
46              residueNames,
47              rawData,
48              getDCCMValue,
49              getSliceAsMatrix
50          };
51      }
```

A função assíncrona, após interpretar o arquivo binário, retorna as informações de número de fatias geradas pelo processamento, número de átomos, uma lista com os nomes dos resíduos identificados pelo próprio índice na ordem correta em relação a proteína e por fim as matrizes de correlação contendo os valores para cada combinação de resíduo. Além destas informações, também são retornadas as *closures getDCCMValue* e *getSliceAsMatrix* responsáveis por acessar as informações coletadas, estas são utilizadas em diferentes locais do código da visualização. A *closure getDCCMValue* acessa a informação da correlação entre dois átomos em uma fatia especificada a partir dos índices dos átomos, e a *closure getSliceAsMatrix* retorna uma fatia em formato de matriz simples que é utilizada para a construção de cada uma das fatias na renderização.

4.3.2 Desenvolvimento do protótipo de ferramenta

O protótipo é desenvolvido seguindo uma organização modular onde uma função principal presente no arquivo *SceneManager.js* controla toda a cena, e os sujeitos da cena estão agrupados em uma pasta separada e são utilizados sob demanda da função principal. A estrutura *SceneManager* funciona como um controlador da aplicação, ela aplica as alterações referentes à alteração das configurações do sistema que está sendo apresentado.

A aplicação se inspira no padrão definido por Shneiderman (1996) que determina a seguinte ordem para a construção de uma visualização de dados: visão geral do dado, seguido por operações interativas como ampliação e filtragem do dado, e então os detalhes da informação (Shneiderman, 1996). Esta ordem que segue como referência para diversas ferramentas (Cui, 2019; Bach, 2016; Keim et al., 2008; Gohnert et al., 2015) de visualizações de dados pois descreve um nível de taxonomia fundamental para a área da

Visualização da Informação.

É responsabilidade fundamental da função *update* presente no arquivo *main.js* realizar o ciclo de renderização. Ao término de sua execução, ela invoca o método *renderer.render(scene, camera)*, que comanda a classe *WebGLRenderer* desenvolvida pelo *Three.js* que serve para desenhar o estado atual da cena na tela, sob a perspectiva da câmera. Esta é a operação que efetivamente exibe o quadro atualizado para o usuário.

Primeiramente a cena é criada vazia com um plano de fundo cinza na função *buildScene*, para então adicionar componentes que constroem a visualização, estes que são gerenciados pela função *SceneManager*. São adicionadas luzes de ambiente e também um indicador colorido, linhas que cobrem um segmento das retas que compõem os eixos cartesianos nas coordenadas *x*, *y* e z (0,0,0), estes segmentos de reta cortam o ponto mencionado.

O observador pode manipular fatores de rotação e translação da câmera instanciada na função *buildCamera* através da utilização do mouse que utiliza a classe *OrbitControls* provinda do Three.js e instanciada na função *buildControls*, isto permite a interação com a cena. A navegação é feita através destes parâmetros relacionados a classe *THREE.PerspectiveCamera* da mesma biblioteca, que mostra os dados através de uma perspectiva única, esta que pode ser alterada. Isto permite a exploração total do espaço tridimensional, inclusive partes internas ao cubo que possui alta densidade de pontos, isto é, o ponto orbitado pela câmera também pode sofrer alterações.

É adicionado o fundo relativo à visualização do método, este fundo é um paralelepípedo que cobre a Matriz Cúbica (Bach; Pietriga; Fekete, 2014) mantendo a distância necessária para não sobrepor visualização, esta forma geométrica renderizada é dinâmica, ou seja, o seu tamanho escala conforme o tamanho da visualização, visto que a mesma pode aumentar de profundidade devido ao número de fatias selecionadas para renderização. Os planos que compõem este fundo são visíveis da perspectiva da câmera somente se o observador está posicionado de maneira que a face sendo observada é interna ao polígono, estes planos possuem bordas sutis na cor preta definida utilizando a programação em *shader* específico, com a intenção de facilitar a percepção da caixa de visualização. Estas características permitem a exploração do dado de uma maneira que o observador não tenha sua perspectiva obstruída durante a navegação no espaço tridimensional.

É criado um painel para manipulação das configurações, constituído pela classe *GUI* que gerencia esta interface gráfica e é instanciada na função *createPanel*, estes parâmetros estão disponíveis para controlar as informações que estão sendo visualizadas. Este

painel de controle contém, nesta ordem, a seleção de limites positivos e negativos para especificar quais pontos devem ser mostrados e quais devem ser ocultados em relação às delimitações demarcadas pela normalização da correlação entre os dados, seleção de fatia específica do método segmentado de DCCM através de um *slider* e seleção do arquivo binário que representa o método aplicado, com diferentes tamanhos de janelas (múltiplos arquivos binários, um para cada tamanho de segmento adotado) para uma réplica específica de um sistema molecular específico dos dados simulados por Cavatão et al. (2024). A utilização destas configurações são detalhadas na subseção sobre operações interativas desta seção.

Após a leitura do arquivo binário, as informações são gerenciadas também pelo *SceneManager*, utilizando as *closures* definidas anteriormente na função *loadBinaryDCCM*. Durante a inicialização da cena (através da função *createSceneSubjects* e do construtor, que um dos sujeitos da cena possui, pertencente à classe *DccmSlice*) os dados brutos da simulação são carregados. A função *getSliceAsMatrix* é então invocada para realizar o processamento inicial, convertendo a matriz de correlação em *buffers* de atributos formatados para *WebGL*. Estes *buffers* são encapsulados em uma classe chamada *THREE.BufferGeometry* e instanciados na cena como um objeto denominado *THREE.Points*, estas que são funções desenvolvidas pelo Three.js, estabelecendo a visualização estática inicial.

A função *updateFromSettings* acionada por eventos da GUI gerencia a reconfiguração dinâmica da visualização. Quando parâmetros, como os limiares de correlação, são alterados pelo usuário, *getSliceAsMatrix* (ou uma lógica de filtragem) é re-executada para recalcular os atributos dos pontos, como suas cores ou visibilidade. Os *buffers* da *BufferGeometry* são atualizados e marcados para notificar o renderizador (*needsUpdate* = true), permitindo que a chamada de renderização, ao final do ciclo de atualização da cena, exiba os pontos filtrados sem a necessidade de recriar o objeto *THREE.Points* do zero.

Estes pontos renderizados representam os valores presentes nas matrizes simétricas referentes ao dado. Para mitigar a oclusão visual em um ambiente 3D denso, onde correlações positivas podem esconder negativas (ou vice-versa), a aplicação utiliza filtragem por limiar (*treshold*), permitindo a inspeção dos pontos de interesse (correlações e anti-correlações fortes). Onde a cor dos mesmos é definida pela função *gradientColorForCorrelationForParticles* que implementa um mapeamento de cor que converte um valor escalar de correlação em RGB, utilizando uma interpolação linear segmentada. A metodologia define um gradiente de cor divergente com três pontos de controle: azul,

branco e vermelho. A função aplica a interpolação em dois segmentos distintos. Para valores negativos, a cor é calculada interpolando entre azul e branco, enquanto para valores positivos, a interpolação ocorre entre branco e vermelho. Esta função é aplicada para cada ponto da visualização, de acordo com a correlação presente na combinação dos $C\alpha$ dos resíduos e momento específico.

4.3.2.1 Visão geral do dado

Então, para a visão geral do dado, a cena posiciona a câmera instanciada na renderização com a visualização enquadrada no centro, isto fornece uma visão ampla sobre o método, como demonstrado pela figura 4.2. Outra característica da visão geral é a possibilidade de interagir com a visualização utilizando os controles disponibilizados através do uso do mouse, como descrito abaixo.

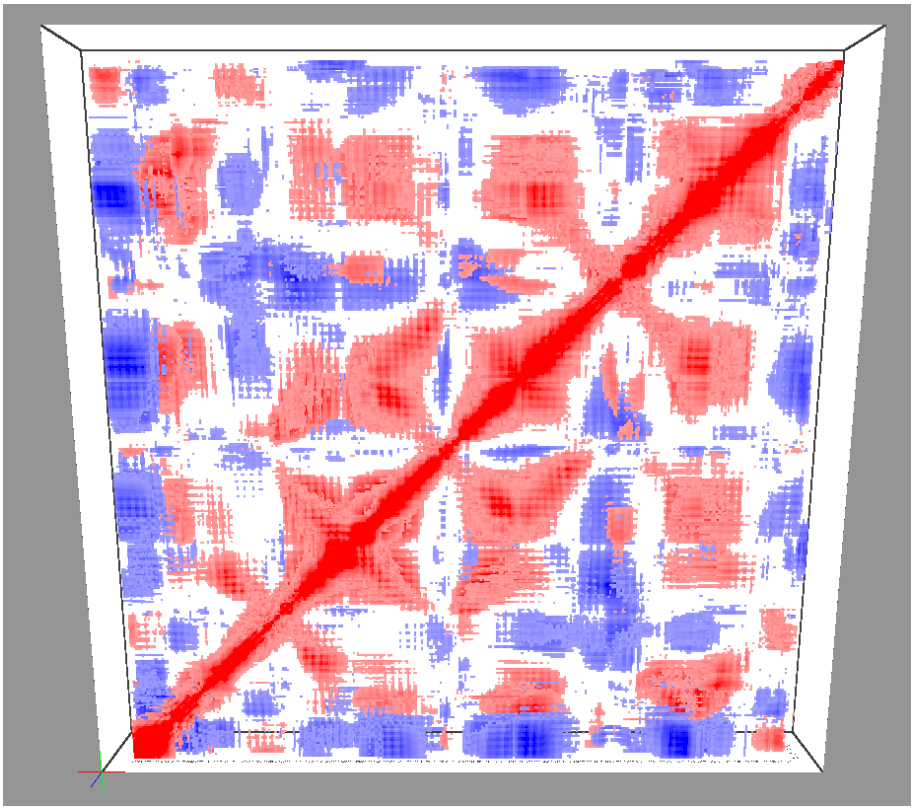

Figura 4.2 – Visão geral da visualização de DCCM segmentado.
Esta figura demonstra a visão geral centralizada da visualização do dado, utilizando a visualização de Matrizes Cúbicas aplicada ao DCCM segmentado. Em que os eixos horizontal e vertical a partir da perspectiva inicial representam os resíduos (Cα) e a profundidade representa diferentes fatias de tempo dispostas em ordem cronológica.
Fonte: Elaborado pelo autor.

4.3.2.2 Operações interativas

Estas operações são disponibilizadas para permitir que o usuário da aplicação consiga manipular a visualização seguindo a lógica proposta por Shneiderman (1996). Estas operações manipulam a câmera utilizada na aplicação e descrita anteriormente no texto, transladando e rotacionando a mesma com um ponto central não fixo na representação proposta.

Aqui a filtragem do dado é realizada de duas maneiras, em relação ao tempo (Bach et al., 2017), permitindo a seleção de uma fatia específica e podendo transladar a fatia selecionada em função do tempo através de um *slider*, a figura 4.3 exemplifica a seleção de uma fatia específica para análise. Considerando que esta interação é realizada de maneira otimizada, sem engasgos no processamento, para sistemas com número de fatias e resíduos relativamente grande, isto cria um ambiente onde a visualização da informação é possível e eficaz.

A outra maneira de filtrar a visualização é a partir dos limites definidos pela normalização da correlação dos dados. Esta funcionalidade, adotada no painel de controle da visualização, denota um limite, considerando o módulo, em formato de *slider* que se estende do valor 0 ao valor 1. O ponto da representação que não respeita a condição de filtragem não é renderizado pela aplicação, evidenciando aqueles pontos que possuem correlações mais fortes. Esta funcionalidade é demonstrada, especialmente, na figura 4.3.

4.3.2.3 Detalhes da informação

O detalhamento de cada ponto renderizado no ambiente tridimensional é feito através de uma estrutura de visualização chamada *tooltip*, esta é uma etiqueta que contém informações específicas sobre a célula que se torna visível ao observador quando o ponteiro está diretamente em cima do ponto de interesse. A operação de identificar qual ponto deve ser detalhado é realizada utilizando a classe *THREE.Raycaster*, que permite traçar uma linha e identificar qual ponto está sobrepondo o ponteiro a partir da perspectiva da câmera. Esta *label* informa o valor da correlação entre dois resíduos, quais são estes resíduos e a qual fatia este resíduo pertence, como demonstrado na imagem 4.4.

Outro detalhamento da informação é a disposição dos resíduos na base da visualização do método, os resíduos possuem identificadores e estes são renderizados na coluna correspondente a cada resíduo, como podemos ver na figura 4.4. Nesta imagem também é possível identificar uma etiqueta que identifica a janela referente a segmentação do mé-

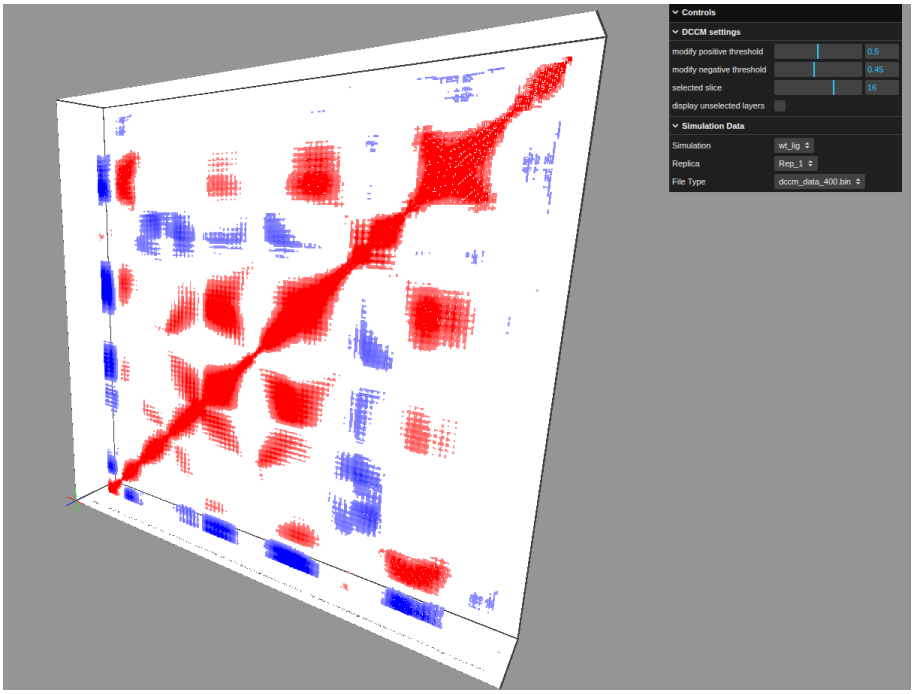

Figura 4.3 – Aplicação de filtros à visualização.
Visão que seleciona uma das fatias da representação composta com a filtragem de pontos por limites determinados pela normalização dos dados.
Fonte: Elaborado pelo autor.

todo, esta etiqueta segue a ordem temporal e denota uma ideia de em que momento da simulação aquelas correlações aconteceram.

Então, a figura 4.5 sumariza os componentes da visualização e aponta as funcionalidades possíveis para o controle da cena. A disposição da cena desta forma se faz necessária para um melhor entendimento dos conceitos propostos no desenvolvimento, buscando aproximar a descrição da ideia da visualização com o que foi desenvolvido.

4.4 Reprodutibilidade e disponibilidade

É necessário que o protótipo possua características que permitam sua reprodução em diferentes contextos, para que seja possível aprimorar, modificar e auditar o mesmo. Por conta disto, este protótipo de ferramenta é de código aberto publicada no *GitHub*, possui a licença *GNU General Public License*, permitindo a distribuição e garantindo a necessidade do desenvolvimento de *software* livre.

Para garantir ainda mais a reprodutibilidade do protótipo, o fragmento responsável pelo *Frontend* utiliza a estratégia de aplicar o *Docker*, este fica responsável por encapsular a ferramenta em imagem e contêiner próprio, permitindo então que a ferramenta se

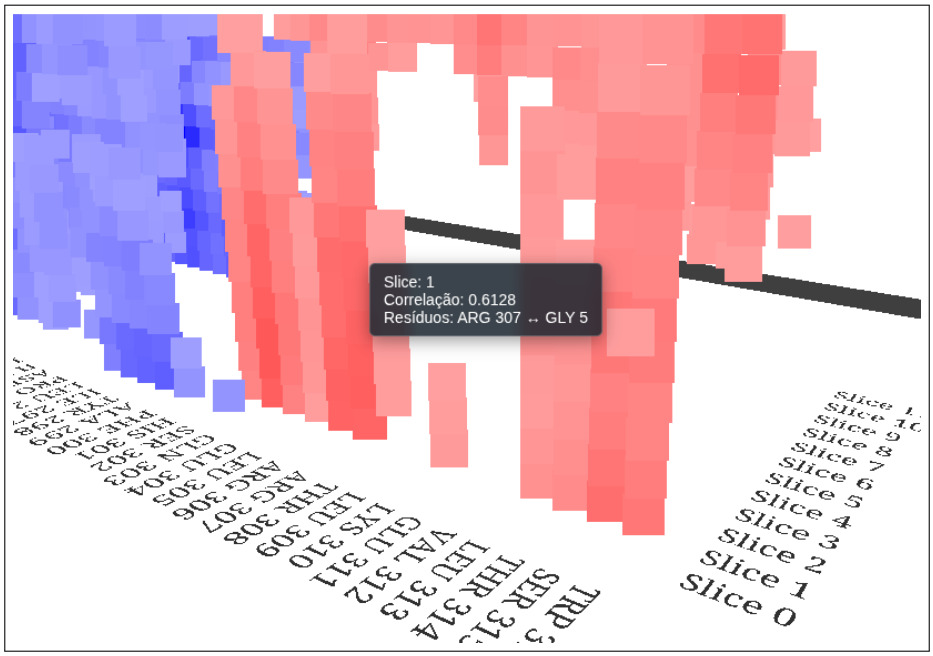

Figura 4.4 – Visão detalhada das informação através de *labels*.
Etiquetas de identificação dos resíduos com seu número de identificação na estrutura primária da proteína, juntamente com a correlação específica de um ponto na visualização.
Fonte: Elaborado pelo autor.

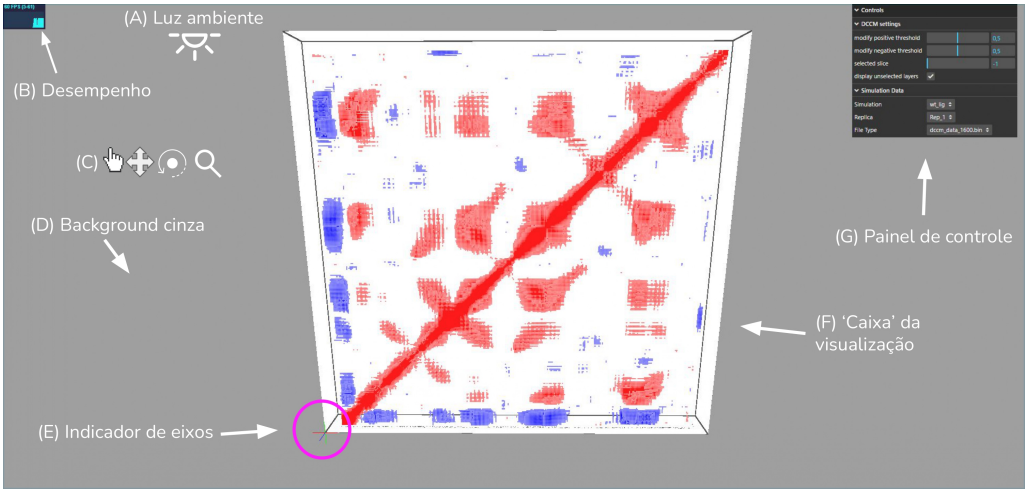

Figura 4.5 – Componentes da visualização.
Disposição dos componentes presentes na visualização, onde: (A) indica a presença de uma iluminação geral da cena, permitindo a visualização do dado, (B) aponta um gráfico de desempenho em relação ao tempo, (C) indica as operações realizadas através do cursor (translação, rotação e ampliação da visualização), (D) indica o plano de fundo cinza, (E) aponta os indicadores referentes aos eixos da visualização, (F) denota a presença da caixa de visualização e (G) aponta o painel de controle presente para a manipulação do dado.
Fonte: Elaborado pelo autor.

construa de maneira automática e garantindo as versões necessárias para a própria renderização do conteúdo desenvolvido.

Em contrapartida, o processamento dos dados por parte do *Backend* é desenvolvido para ser utilizado após os dados já preparados, permitindo apenas o processamento de trajetórias contendo apenas os átomos do $C\alpha$ da proteína presente no complexo analisado.

Outro fator que impacta o alcance dos objetivos da presente pesquisa é a disponibilidade deste estudo de caso na *Web*, tendo em vista que o acesso a este conteúdo permite que, de maneira prática, futuros investigadores consigam acessar tanto o código fonte, como evidenciado pela reprodutibilidade do código, quanto a disponibilidade da ferramenta interativa. Para alcançar este objetivo é utilizada a biblioteca *Vite* (You, 2020) que possibilita a construção de um ambiente que dispõem das funcionalidades necessárias para a publicação do *frontend* desta aplicação. Permitindo que futuros exploradores da técnica tomem posse, através do acesso ao *site* que hospeda a aplicação, <https://sbcb.inf.ufrgs.br/dccm>, e através do acesso ao código fonte presente no *GitHub*, através do endereço <https://github.com/sbcblab/DCCM-Visualization>, daquilo que é necessário para compreender os conceitos descritos no corpo do texto desta investigação. Este é um fator impactante, pois faz com que o desenvolvimento deste protótipo de ferramenta esteja disponível de maneira aberta e que seja auditável por futuros investigadores que possam encontrar, neste trabalho, a construção e utilização de ferramentas desenvolvidas pela comunidade de maneira cientifica.

5 RESULTADOS E DISCUSSÃO

Este capítulo foca na avaliação metodológica do protótipo de ferramenta desenvolvido. A principal contribuição deste trabalho é a criação de uma abordagem que supera a limitação temporal da análise de DCCM tradicional, que é baseada na média de toda a trajetória e frequentemente aplicado a todas as réplicas simuladas. O objetivo desta seção não é conduzir um estudo biológico aprofundado, mas sim provar, como conceito, que a visualização de DCCM segmentado no tempo é capaz de revelar dinâmicas e variabilidades que são ocultas pela análise da média presente no método.

Como descrito na seção de desenvolvimento da abordagem, ao descrever as operações realizadas nos dados brutos de DM, usualmente o DCCM seria realizado manualmente através dos comandos do GROMACS que executam os cálculos da análise, como em Cavatão et al. (2024). A técnica desenvolvida aqui, além de discriminar covariâncias em tempos menores da trajetória, automatiza esta abordagem da análise. Tendo em vista que para se ter o mesmo resultado a partir do método usual, haveria uma série de processos manuais de processamento do dado. Como por exemplo, para realizar o cálculo do DCCM para cada período de tempo desejado (fatias) o investigador teria que realizar um *gmx trjconv* em todas as réplicas de todos os sistemas e depois seguir com o procedimento padrão do GROMACS para calcular o DCCM para cada fatia extraída da trajetória original. Além de dificultar a comparação direta das fatias por métodos usuais de visualização de gráficos. Desse modo, este trabalho contribui para a automatização do processo para o investigador e supera os desafios do processamento manual.

5.1 Exemplo de análise utilizando o método desenvolvido

Nos exemplos a seguir, a aplicação do método proposto é explorada tendo como base de comparação o DCCM usual. Para a filtragem dos pontos na visualização, definiu-se um limiar de módulo de correlação de 0.5, visando reduzir a poluição visual e destacar apenas as correlações mais fortes.

Em relação à segmentação temporal, foram adotados tamanhos de janela de 800 e 200 *frames* para as Figuras 5.3 e 5.4, respectivamente. Estes tamanhos de janela influenciam diretamente a profundidade da visualização em relação ao tempo, onde quanto menor o número de *frames* por segmento, mais fatias representarão o dado visualmente. A escolha destes valores foi realizada de maneira exploratória com o fim de demonstrar

a flexibilidade da aplicação em lidar com diferentes níveis de granularidade temporal e ilustrar *trade-off*, referente ao possível ruído (Hünenberger; Mark; Gunsteren, 1995), presente na análise:

- **Janela de 800 *frames*:** Representa uma abordagem focada na estabilidade. Ao aumentar o tamanho da amostra por fatia, favorece-se a convergência estatística do cálculo da covariância e a suavização de ruídos, destacando tendências de comportamento mais persistentes ao longo da trajetória.
- **Janela de 200 *frames*:** Representa uma análise de alta resolução temporal, embora diminua a amostragem estatística para o cálculo da correlação de *Pearson*, podendo introduzir ruídos se a janela for curta demais para capturar a flutuação do movimento. Esta escolha visa capturar eventos transientes e mudanças conformacionais rápidas, resultando em uma visualização com maior profundidade (mais fatias), permitindo observar a características detalhadas das correlações.

O DCCM utilizado no desenvolvimento do trabalho de investigação da MC1R por Cavatão et al. (2024) é representado pela figura 5.1, o dado utilizado para gerar esta análise é o mesmo utilizado para gerar as visualizações tridimensionais da presente pesquisa. Por serem visualizações que consideram os mesmo dados de simulação, é possível comparar diretamente e evidenciar algumas características que se destacam, fortalecendo a hipótese de que é possível perceber correlações que não seriam destacadas pelo método usual.

Comparando este mesmo DCCM só que de maneira fatiada pelo método proposto pelo trabalho (figura 5.2), é possível notar a similaridade dos dados que, na visualização de Cavatão et al. (2024) o dado é demonstrado apenas através de um plano (achatado em relação a visualização tridimensional proposta). Contudo, utilizando o método proposto, é possível identificar regiões em que a correlação transita de uma correlação menor para uma correlação maior, denotada pelo volume visual gerado pelos pontos vizinhos em relação ao tempo. Este volume é demonstrado através da figura 5.3. É possível denotar as correlações e anti-correlações que acontecem simultaneamente, podendo caracterizar comportamentos que são de interesse do investigador.

A figura 5.3 demonstra o detalhamento que busca demonstrar o volume gerado pela vizinhança espacial de pontos. Isto se torna possível de visualizar após a aplicação dos filtros definidos e desenvolvidos. O detalhamento através do *tooltip* auxilia na identificação de quais resíduos compõem a região de interesse. E a característica da interatividade dá liberdade de movimento necessária para investigar estas regiões de interesse

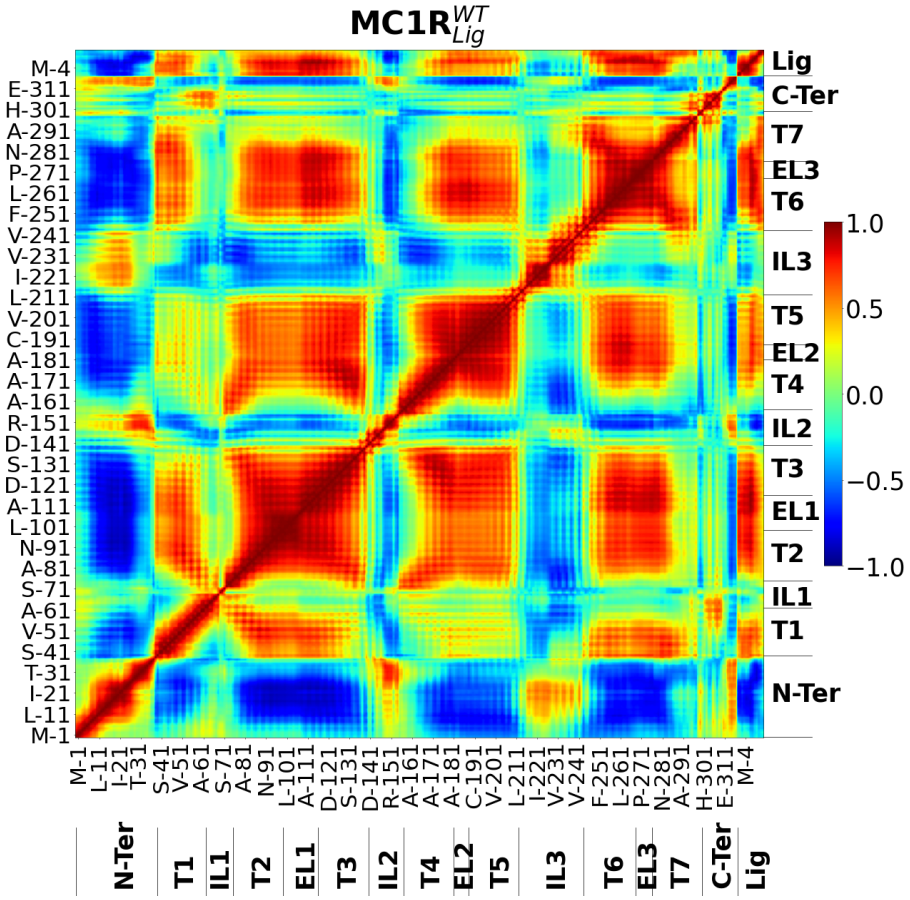

Figura 5.1 – Aplicação padrão de DCCM à uma simulação.
DCCM da proteína em sua forma WT simulada com seu ligante, onde a cor azul representa anti-correlações e a cor vermelha representa correlações.
Fonte: Adaptado de Cavatão et al. (2024).

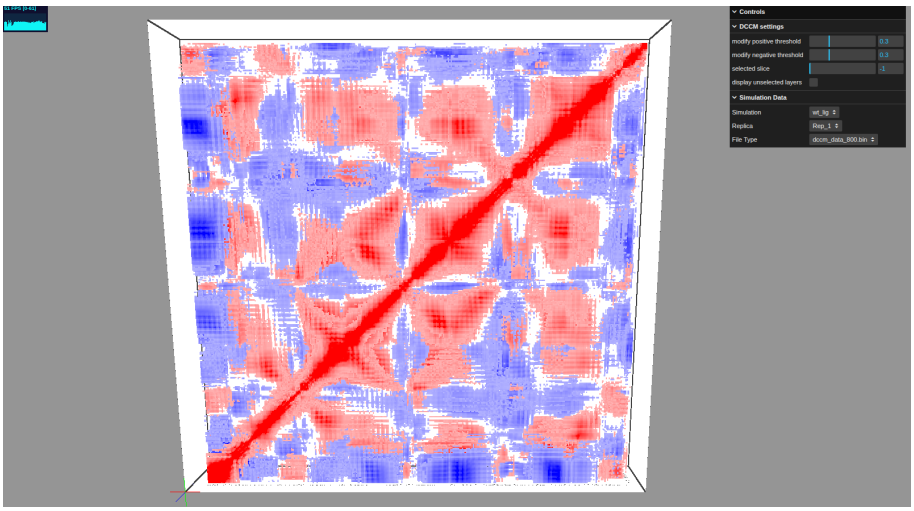

Figura 5.2 – Comparação através da visualização do método aplicado à uma simulação.
Visão geral do DCCM dinâmico em relação ao tempo, da mesma simulação utilizada pela figura 5.1, onde os pontos com a cor vermelha representam correlações positivas e a cor azul representam anti-correlações.
Fonte: Elaborado pelo autor.

de maneira detalhada e conforme o interesse de quem está investigando.

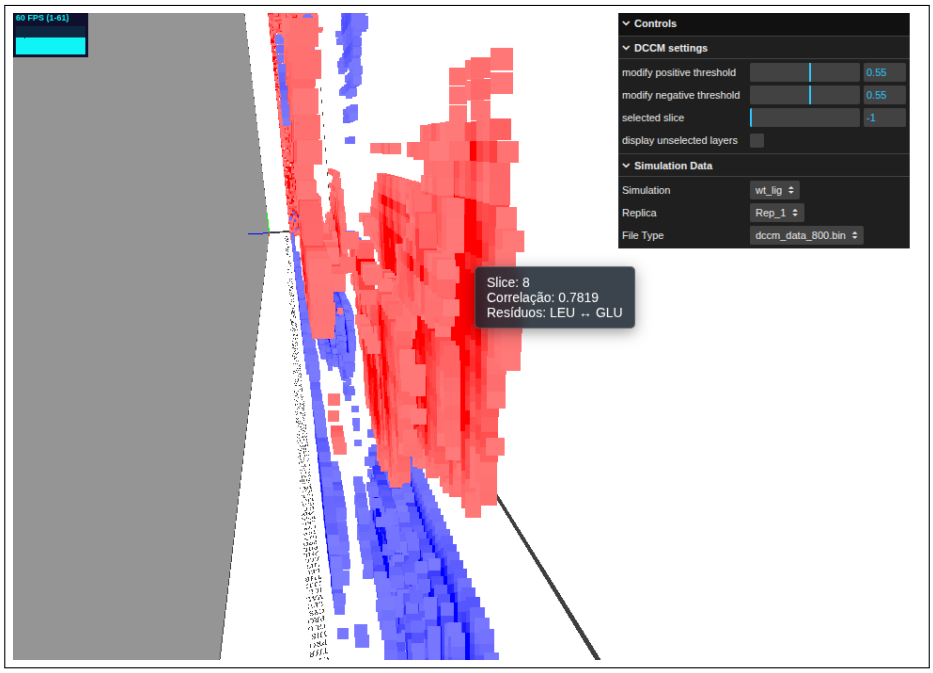

Figura 5.3 – Detalhamento do método aplicado.
Detalhes do DCCM dinâmico em relação ao tempo, em contraste com DCCM usual presenta na figura 5.2.
Fonte: Elaborado pelo autor.

Ao selecionar um número maior de fatias do método e filtrar correlações menos relevantes é possível observar transições entre regiões não correlacionadas e correlacionadas, como observável na figura 5.4, o mesmo é válido para as anti-correlações. Nesse caso específico, a trajetória de 10000 *frames* totais, foi dividido de maneira que cada janela contém 200 *frames* resultando em 50 fatias. É possível observar que a partir da fatia de número 25 até a fatia 36, há uma crescente correlação nesta região específica, denotada pela caixa formada de arestas na cor rosa presente na imagem. Este aumento repentino de correlação pode indicar um evento de acoplamento estrutural, como uma alteração da estrutura de uma parte específica da proteína ou a propagação de um determinado movimento na proteína após acomodação do ligante.

É importante destacar que esta visualização também pode auxiliar na identificação de artefatos gerados durante a simulação, isto é, algum erro que possa ter sido causado durante a preparação do sistema molecular (Verli, 2014) ou em algum outro momento da cadeia de trabalho no desenvolvimento de análises que envolvam Bioinformática Estrutural. Enquanto mudanças biológicas tendem a apresentar uma transição gradual ou sustentada (como na Figura 5.4), um artefato de simulação (como problemas de contorno periódico ou aquecimento indevido) poderia se manifestar como um '*flickering*' (cintila-

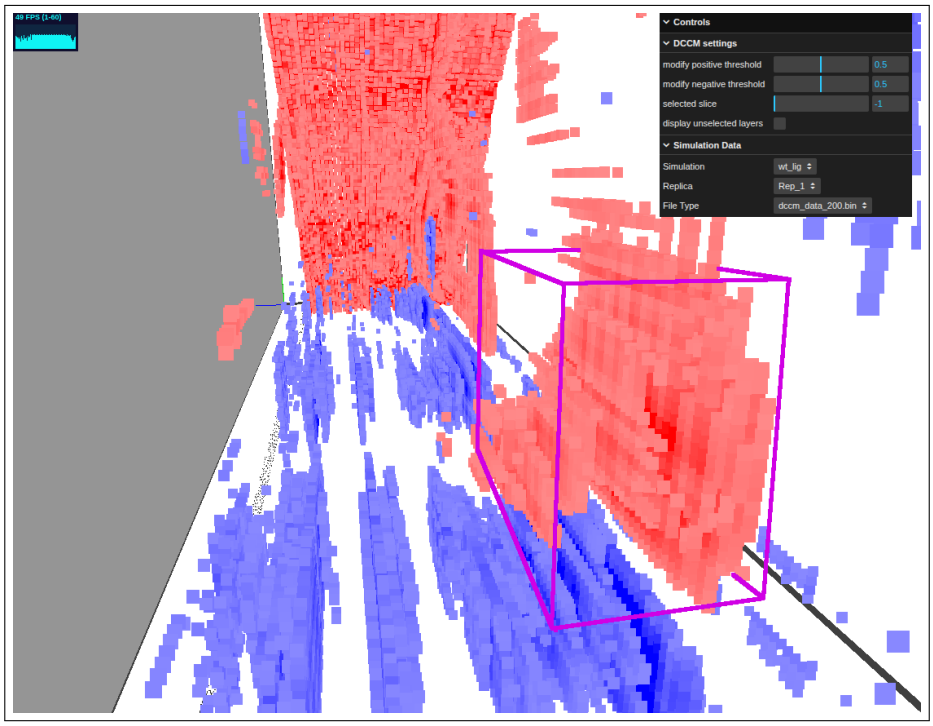

Figura 5.4 – Transição da amplitude nas correlações.
Transição de elementos não correlacionados para um momento em que possuem correlações relevantes, denotada pela caixa construída por arestas na cor rosa.
Fonte: Elaborado pelo autor.

ção) errático, faixas verticais contínuas com altos valores de correlação/anti-correlação sem sentido biológico, ou mudanças abruptas que não persistem. Estes artefatos podem impactar as pesquisas realizadas negativamente, e a percepção precoce dos mesmos auxilia no desenvolvimento de trabalhos concisos. Neste contexto, um artefato pode se manifestar por uma transição não fluída das correlações presentes no movimento do complexo molecular e cabe ao investigador compreender o que esta informação sobre as correlações significa biologicamente, fisicamente ou até mesmo quimicamente.

A partir destes exemplos é possível perceber que este método permite um maior detalhamento dos movimentos do sistema. Esta contribuição permite o investigador observar em quais momentos pode estar ocorrendo uma transição de estados ou movimentos da proteína, que podem estar relacionados com alguma função específica do sistema.

5.2 Análise de Desempenho e Escalabilidade

Um fator crítico no desenvolvimento de ferramentas de visualização *Web* para Bioinformática é a escalabilidade em relação ao tamanho do sistema molecular. Na abor-

dagem proposta, cada ponto renderizado na cena tridimensional representa uma célula da matriz de correlação. Consequentemente, o número total de pontos (P) a serem processados e desenhados cresce quadraticamente em relação ao número de resíduos (N) da proteína, tal que $P \propto N^2$.

Para o sistema MC1R analisado neste trabalho, que possui 317 resíduos, cada fatia temporal resulta em uma matriz de 317×317, gerando 100.489 pontos (considerando a redundância presente na visualização). Ao visualizar múltiplas fatias simultaneamente (por exemplo, 50 fatias), a aplicação gerencia a renderização de aproximadamente 5 milhões de pontos. Nestas condições, utilizando os filtros disponíveis, a aplicação manteve uma taxa de quadros (*frame rate*) fluida, permitindo interação em tempo real em *hardware* de consumo padrão.

Contudo, para complexos moleculares significativamente maiores, a complexidade $O(N^2)$ impõe limitações técnicas. Proteínas com mais de 1.000 resíduos gerariam milhões de pontos por fatia única, o que poderia saturar a memória RAM utilizada pelo navegador e criar um gargalo (*bottleneck*) na CPU durante as etapas de filtragem e atualização da geometria (*BufferGeometry*). Embora a biblioteca *Three.js* utilize a GPU para a rasterização eficiente, o pré-processamento dos atributos de cor e posição no lado do cliente (*client-side*) torna-se custoso. Portanto, a arquitetura atual é validada para proteínas de médio porte, sendo que a visualização de complexos massivos exigiria estratégias de otimização adicionais.

5.3 Limitações da Granularidade Temporal, Robustez Estatística e Linearidade do Método

Embora a aplicação facilite a identificação de padrões por meio de sua interface interativa, a percepção da convergência das correlações ao longo do tempo exige cautela. É fundamental analisar a robustez estatística das matrizes calculadas em janelas temporais curtas, garantindo que os padrões observados não sejam meros artefatos de uma amostragem reduzida.

A validade da correlação de *Pearson*, base do método DCCM, depende de uma amostragem suficiente do espaço conformacional para que a covariância entre os deslocamentos atômicos seja estatisticamente significante, conforme alertado pelo trabalho de Hünenberger, Mark e Gunsteren (1995) considerando o número de conformações. Ao segmentar a trajetória em fatias de alta granularidade (como janelas de 200 *frames*), assume-

se o risco de que os movimentos atômicos registrados não sejam representativos, podendo introduzir ruídos ou correlações espúrias.

O presente trabalho não aplicou métodos quantitativos de validação para garantir a significância estatística de cada fatia individualmente. Portanto, a interpretação das "correlações de transição"apresentadas deve ser realizada com cautela, entendendo a visualização como um protótipo de ferramenta exploratório. A escolha do tamanho da janela, dessa forma, torna-se um hiperparâmetro crítico que o investigador deve ajustar buscando o equilíbrio entre a resolução temporal desejada e a confiabilidade estatística dos dados.

É importante observar que o protótipo de visualização proposto, embora amplie a percepção temporal dos dados, possui restrições matemáticas que vem do método DCCM padrão. Como o algoritmo implementado é baseado na equação de correlação de *Pearson*, a análise está restringida exclusivamente à captura de características lineares entre os vetores de flutuação atômica. Consequentemente, movimentos correlacionados de natureza não-linear permanecem imperceptíveis a esta abordagem, independentemente da qualidade da representação tridimensional ou da granularidade do fatiamento temporal. O protótipo, portanto, aprimora a interpretabilidade visual das características lineares presentes nas diferentes conformações adotadas pelo complexo molecular durante a simulação.

6 CONCLUSÃO

A partir da análise do método proposto foi possível indicar caminhos possíveis para a utilização da aplicação, que, desenvolvida a partir de uma estrutura conceitual (Bach et al., 2017) que guiou os passos do desenvolvimento, toma neste trabalho a iniciativa em um dado previamente não abordado. Isto permite a exploração deste campo específico, pretendendo expandir a análise de dados de Bioinformática Estrutural no estudo dos Mapas de Correlações Cruzadas do comportamento dinâmico de sistemas moleculares simulados.

Então, o primeiro objetivo foi plenamente alcançado através do desenvolvimento de um algoritmo em *Python* que utiliza a biblioteca *MDTraj* (McGibbon et al., 2015) para segmentar trajetórias de Dinâmica Molecular em janelas temporais. Para cada fatia, o método de DCCM foi computado individualmente através de uma implementação matricial com NumPy/CuPy. Esta abordagem metodológica foi a fundação de todo o trabalho, pois gerou a estrutura de dados (uma série temporal de matrizes de correlação) essencial para a análise, sendo capaz de capturar correlações lineares, que por definição, podem ser ocultadas na média temporal do método DCCM usual.

O desenvolvimento do protótipo de ferramenta de visualização, foi concretizado como uma aplicação web interativa construída em *JavaScript* e *Three.js*. Este protótipo foi projetado especificamente para carregar e renderizar os dados resultantes do algoritmo de fatiamento. A arquitetura demonstrou ser eficaz em transformar as múltiplas matrizes de dados em um ambiente tridimensional, implementando com sucesso o mantra de (Shneiderman, 1996) (*Overview first, zoom and filter, then details-on-demand*) através de controles de câmera, filtros de GUI e *tooltip* para detalhamento. A aplicação é de código aberto, estando disponível para alterações e aprimoramentos de futuros pesquisadores.

A exploração da aplicação de uma técnica de visualização genérica em um domínio específico foi abordada. A pesquisa adaptou a taxonomia e a técnica da "Matriz Cúbica" proposta por Bach, Pietriga e Fekete (2014) ao domínio específico da Bioinformática Estrutural. Ao modelar as fatias de DCCM como uma rede dinâmica e empilhá-las ao longo de um eixo temporal, o trabalho preencheu a lacuna proposta, demonstrando que esta abstração visual genérica é uma forma viável e intuitiva de representar a evolução temporal das correlações presentes em simulações moleculares.

Finalmente, uma avaliação da abordagem visual foi demonstrada no capítulo anterior, que comparou diretamente o DCCM tradicional (a média em duas dimensões, *he-*

atmap) com a visualização em três dimensões fatiada em relação ao tempo do mesmo sistema (MC1R). Esta comparação evidenciou a principal contribuição do protótipo de ferramenta: enquanto a análise comum apresenta um quadro único e médio, a abordagem proposta revela a variabilidade e a evolução das correlações. A aplicação demonstrou ser capaz de destacar volumes de correlação que flutuam ao longo do tempo, revelando padrões dinâmicos e correlações que transicionam, estas que seriam difíceis de perceber na análise tradicional, validando assim a hipótese central do trabalho. Diferentemente de uma animação 2D (vídeo), que exige que o usuário memorize estados passados, a Matriz Cúbica apresenta o histórico completo da simulação em uma única estrutura estática, permitindo a identificação de padrões temporais contínuos de maneira visual imediatamente.

O trabalho deixa em aberto diversas questões que podem ser abordadas em trabalhos futuros, estas questões são:

A avaliação apresentada neste estudo concentrou-se na equivalência e ganho informacional dos mapas gerados em comparação ao método tradicional. Contudo, a eficácia da aplicação na detecção de padrões por terceiros não foi desenvolvida quantitativamente ou qualitativamente. Não foram conduzidos testes controlados de usabilidade ou avaliações com grupos de pesquisadores da área de Bioinformática para medir a funcionalidade, questões referentes à interface ou a diferença entre o uso padrão do DCCM e o proposto na descoberta de *insights* relevantes. Portanto, a afirmação de que o protótipo de ferramenta "facilita"a análise é baseada na disponibilidade da nova dimensão de dado (tempo) que era de difícil acesso anteriormente, caracterizando este trabalho como uma Prova de Conceito técnico. A validação prática da utilidade da ferramenta com usuários finais permanece uma etapa crucial a ser abordada em trabalhos futuros, esta que poderia ser feita através da utilização da ferramenta por pesquisadores ou profissionais da Bioinformática Estrutural, seguida de uma entrevista detalhada sobre a usabilidade do protótipo, buscando compreender quais são as necessidades do usuário final. Assim como é necessária a validação detalhada referente ao processamento necessário para calcular o método segmentado e renderizar o dado, para compreender o alcance referente ao limite de tamanho do complexo molecular à ser visualizado pelo protótipo desenvolvido.

A utilização de diferentes métodos para o processamento de DM, podendo abordar, por exemplo, a Correlação Generalizada proposta por Lange e Grubmüller (2006) que busca capturar correlações não lineares. Esta perspectiva agregaria a informação de uma maneira que as correlações que são perdidas ao calcular através do coeficiente de *Pearson* pudessem se tornar visíveis ao integrar este método com a visualização de Ma-

trizes Cúbicas. Até mesmo a aplicação da técnica de TDDCC definida por Okamoto e Ando (2024) de maneira segmentada, considerando as diferentes janelas de tempo, o que poderia agregar informações que dependem do tempo. Outra informação que poderia ser considerada é o Dicionário de Estruturas Secundária das Proteínas (DSSP, do inglês *Dictionary of Secondary Structure of Proteins*) (Gorelov et al., 2024), que, em conjunto com a atual proposta poderia auxiliar na percepção de mudanças na conformação da estrutura, indicando um caminho da exploração por parte do usuário da ferramenta.

Considerando a utilização de diferentes métodos na visualização, seria também possível a inclusão de uma representação tridimensional da simulação do complexo molecular abordado de maneira paralela, em relação ao espaço da visualização, para a identificação dos comportamentos através da inspeção pelo método segmentado de DCCM. Podendo indicar pontos de interesse em relação à estrutura tridimensional de maneira prática, indicando regiões de alta correlação ou anti-correlação com um vínculo visual entre o método e a trajetória de DM. Isto também contribuiria com a modelagem do problema, aproximando os comportamentos dos complexos moleculares com o que os mesmos representam nesta rede dinâmica, permitindo a exploração do método de maneira mais aprofundada.

Também seria um caminho possível a exploração de filtros que considera a teoria de grafos para a análise de redes e de redes dinâmicas, podendo evidenciar na visualização pontos de interesse e assim contribuindo para uma melhor exploração interativa da técnica de visualização do dado. Este trabalho seria importante e a fundamentação para o desenvolvimento da parte visual já é definida, portanto a aplicação destas abordagens possui uma taxonomia já descrita para a disposição dos componentes na visualização bastando a exploração algorítmica dos métodos e a implementação destes conceitos na visualização já desenvolvida. Estas abordagens, considerando a teoria de grafos, poderiam ser: cálculo de menor caminho, análise de centralidade, identificação de nodos críticos da simulação, dentre outras técnicas (Patel; Sinha; Palermo, 2024).

Existe também a possibilidade da integração do protótipo de ferramenta, isto é, o desenvolvimento de uma ferramenta que acopla os dois módulos implementados no trabalho, levando em consideração que esta divisão do processamento é uma das limitações do trabalho. Isto possibilitaria a investigação de diferentes complexos moleculares de uma maneira mais ágil do que a reprodução inteira do trabalho, encurtando o caminho entre os usuários (ex. pesquisadores de Bioinformática Estrutural) e a utilização da ferramenta, tendo em vista que a aplicação desenvolvida serve de demonstração da aplicação

do método em dados específicos. Isto seria de grande relevância para garantir o acesso da ferramenta de maneira difundida no contexto da Bioinformática, isto se garantida a disponibilidade da ferramenta integrada e acessível através de uma aplicação *Web*. Em conjunto com a integração, permitir o uso de dados resultantes de diferentes ferramentas de simulação, além do uso do GROMACS, é essencial para a integração total da visualização deste dado.

REFERÊNCIAS

ABRAHAM, M. J. et al. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. **SoftwareX**, v. 1-2, p. 19–25, 2015.

ALKADI, M. et al. Understanding Barriers to Network Exploration with Visualization: A Report from the Trenches. **IEEE Transactions on Visualization and Computer Graphics**, v. 29, n. 1, p. 907–917, 2023.

ALLEN, M. P. Introduction to Molecular Dynamics Simulation. In: **Computational Soft Matter: From Synthetic Polymers to Proteins**. [S.l.: s.n.], 2004. v. 23, p. 1–28.

ANDRIENKO, G. L. et al. Space, time and visual analytics. **International Journal of Geographical Information Science**, v. 24, n. 10, p. 1577–1600, 2010.

BACH, B. **Connections, changes, and cubes : unfolding dynamic networks for visual exploration**. Thesis (PhD) — Université Paris-Sud - Paris XI, May 2014.

BACH, B. Unfolding Dynamic Networks for Visual Exploration. **IEEE Computer Graphics and Applications**, v. 36, n. 2, p. 74–82, 2016.

BACH, B. et al. A Review of Temporal Data Visualizations Based on Space-Time Cube Operations. In: **Eurographics Conference on Visualization**. [S.l.: s.n.], 2014. p. 23–41.

BACH, B. et al. A Descriptive Framework for Temporal Data Visualizations Based on Generalized Space-Time Cubes. **Computer Graphics Forum**, v. 36, n. 6, p. 36–61, 2017.

BACH, B. et al. Small MultiPiles: Piling Time to Explore Temporal Patterns in Dynamic Networks. **Computer Graphics Forum**, v. 34, n. 3, p. 31–40, 2015.

BACH, B.; PIETRIGA, E.; FEKETE, J.-D. Visualizing Dynamic Networks with Matrix Cubes. In: **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. [S.l.]: Association for Computing Machinery, 2014. p. 877–886.

BALTRUKEVICH, H.; PODLEWSKA, S. From Data to Knowledge: Systematic Review of Tools for Automatic Analysis of Molecular Dynamics Output. **Frontiers in Pharmacology**, v. 13, p. 844293, 2022.

BECK, F. et al. The State of the Art in Visualizing Dynamic Graphs. In: **EuroVis - STARs**. [S.l.]: The Eurographics Association, 2014. p. 83–103.

BELGHIT, H. et al. From complex data to clear insights: visualizing molecular dynamics trajectories. **Frontiers in Bioinformatics**, v. 4, p. 1356659, 2024.

BERNETTI, M. et al. Probing allosteric communication with combined molecular dynamics simulations and network analysis. **Current Opinion in Structural Biology**, v. 86, p. 1–10, 2024.

BRATH, R. 3D InfoVis is Here to Stay: Deal with It. In: **Proceedings of the 2014 IEEE VIS International Workshop on 3DVis**. [S.l.: s.n.], 2014. p. 25–31.

CABELLO, R. **Three.js – JavaScript 3D Library**. 2010. <https://threejs.org/>. Accessed: 2025-11-13.

CAVATÃO, F. G. et al. Molecular basis of mc1r activation: Mutation-induced alterations in structural dynamics. **Proteins**, v. 92, n. 11, p. 1297–1307, 2024.

CUI, W. Visual Analytics: A Comprehensive Overview. **IEEE Access**, v. 7, p. 81555–81573, 2019.

DASH, R. et al. Dynamic insights into the effects of nonsynonymous polymorphisms (nsSNPs) on loss of TREM2 function. **Scientific Reports**, v. 12, p. 9378, 2022.

DAVID, C. C.; JACOBS, D. J. Principal Component Analysis: A Method for Determining the Essential Dynamics of Proteins. In: **Protein Dynamics: Methods and Protocols**. [S.l.: s.n.], 2014. v. 1084, p. 193–226.

GOHNERT, T. et al. 3D DynNetVis - A 3D Visualization Technique for Dynamic Networks. In: **Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining**. [S.l.: s.n.], 2015. p. 737–740.

GORELOV, S. V. et al. DSSP in GROMACS: Tool for Defining Secondary Structures of Proteins in Trajectories. **Journal of Chemical Information and Modeling**, v. 64, n. 9, p. 3593–3598, 2024.

GOWERS, R. J. et al. MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. In: **Proceedings of the 15th Python in Science Conference**. [S.l.: s.n.], 2016. p. 98–105.

GREENACRE, M. et al. Principal component analysis. **Nature Reviews Methods Primers**, v. 2, n. 1, p. 1–21, 2022.

HARRIS, C. R. et al. Array programming with NumPy. **Nature**, v. 585, n. 7825, p. 357–362, 2020.

HENRY, N.; FEKETE, J.-D. MatrixExplorer: a Dual-Representation System to Explore Social Networks. **IEEE Transactions on Visualization and Computer Graphics**, v. 12, n. 5, p. 677–684, 2006.

HOLLINGSWORTH, S. A.; DROR, R. O. Molecular Dynamics Simulation for All. **Neuron**, v. 99, n. 6, p. 1129–1143, 2018.

HONG, J. et al. A Survey of Designs for Combined 2D+3D Visual Representations. **IEEE Transactions on Visualization and Computer Graphics**, v. 30, n. 6, p. 2888–2902, 2024.

HüNENBERGER, P. H.; MARK, A. E.; GUNSTEREN, W. F. van. Fluctuation and cross-correlation analysis of protein motions observed in nanosecond molecular dynamics simulations. **Journal of Molecular Biology**, v. 252, n. 4, p. 492–503, 1995.

KALE, B.; SUN, M.; PAPKA, M. E. The State of the Art in Visualizing Dynamic Multivariate Networks. **Computer Graphics Forum**, v. 42, n. 3, p. 471–490, 2023.

KASAHARA, K.; FUKUDA, I.; NAKAMURA, H. A Novel Approach of Dynamic Cross Correlation Analysis on Molecular Dynamics Simulations and Its Application to Ets1 Dimer–DNA Complex. **PLOS ONE**, v. 9, n. 11, p. e112419, 2014.

KEIM, D. A. et al. Visual Analytics: Definition, Process, and Challenges. In: **Information Visualization: Human-Centered Issues and Perspectives**. [S.l.: s.n.], 2008. v. 4950, p. 154–175.

LANGE, O. F.; GRUBMÜLLER, H. Generalized correlation for biomolecular dynamics. **Proteins: Structure, Function, and Bioinformatics**, v. 62, n. 4, p. 1053–1061, 2006.

LOUKATOU, S. et al. Molecular dynamics simulations through GPU video games technologies. **Journal of Molecular Biochemistry**, v. 3, n. 2, p. 64–71, 2014.

MCGIBBON, R. T. et al. MDTraj: A Modern Open Library for the Analysis of Molecular Dynamics Trajectories. **Biophysical Journal**, v. 109, n. 8, p. 1528–1532, 2015.

MICHAUD-AGRAWAL, N. et al. MDAnalysis: A Toolkit for the Analysis of Molecular Dynamics Simulations. **Journal of Computational Chemistry**, v. 32, n. 10, p. 2319–2327, 2011.

OKAMOTO, C.; ANDO, K. Molecular dynamics simulation analysis of structural dynamic cross correlation induced by odorant hydrogen-bonding in mouse eugenol olfactory receptor. **Biophysics and Physicobiology**, v. 21, n. 1, p. e210007, 2024.

OKUTA, R. et al. CuPy: A NumPy-Compatible Library for NVIDIA GPU Calculations. In: **Proceedings of the Workshop on Machine Learning Systems (LearningSys) in the Thirty-first Annual Conference on Neural Information Processing Systems**. [S.l.: s.n.], 2017.

PARIDA, P. K.; PAUL, D.; CHAKRAVORTY, D. The natural way forward: Molecular dynamics simulation analysis of phytochemicals from Indian medicinal plants as potential inhibitors of SARS-CoV-2 targets. **Phytotherapy Research**, v. 34, n. 12, p. 3420–3433, 2020.

PATEL, A. C.; SINHA, S.; PALERMO, G. Graph theory approaches for molecular dynamics simulations. **Quarterly Reviews of Biophysics**, v. 57, p. e15, 2024.

SHNEIDERMAN, B. The eyes have it: a task by data type taxonomy for information visualizations. In: **Proceedings of the IEEE Symposium on Visual Languages**. [S.l.: s.n.], 1996. p. 336–343.

SHU, X. et al. Does This Have a Particular Meaning? Interactive Pattern Explanation for Network Visualizations. **IEEE Transactions on Visualization and Computer Graphics**, v. 31, n. 1, p. 677–687, 2025.

VERLI, H. (Ed.). **Bioinformática: da biologia à flexibilidade molecular**. 1. ed. São Paulo: Sociedade Brasileira de Bioquímica e Biologia Molecular, 2014. 282 p. ISBN 9788569288008.

WANG, Z. et al. Exploring the Resistance Mechanisms of Distal D835V Mutation in FLT3 to Inhibitors. **Oxidative Medicine and Cellular Longevity**, v. 2022, p. 3720026, 2022.

WANG, Z. et al. Cheat Sheets for Data Visualization Techniques. In: **Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems**. [S.l.: s.n.], 2020. p. 1–13.

WEN, X. et al. DiffSeer: Difference-Based Dynamic Weighted Graph Visualization. **IEEE Computer Graphics and Applications**, v. 43, n. 3, p. 12–23, 2023.

WILSON, R. J. **Introduction to Graph Theory**. 4th. ed. Harlow, England: Longman, 1996. ISBN 0582249937.

YOU, E. **Vite: Next Generation Frontend Tooling**. 2020. <https://vite.dev/>. Accessed: 16 Nov 2025.

YU, H.; DALBY, P. A. A beginner's guide to molecular dynamics simulations and the identification of cross-correlation networks for enzyme engineering. In: **Enzyme Engineering and Evolution: General Methods**. [S.l.: s.n.], 2020. v. 643, p. 15–49.