VETOR

Aceleração da Solução Numérica de Problemas Da Biomecânica Cardíaca Utilizando Métodos Multigrid da Biblioteca AmgX☆

Use of the AmgX Library for Solving Problems Related to Cardiac Mechanics

Jonatas Dias Machado Costa, Lucas Silva Santana, Rodrigo Weber dos Santos, Bernardo Martins Rocha, Joventino de Oliveira Campos†

FISIOCOMP - Laboratório de Fisiologia Computacional, Universidade Federal de Juiz de Fora, Juiz de Fora, Brasil
†**Autor correspondente:** joventino.campos@ufjf.br

Resumo

A resolução de sistemas lineares desempenha um papel fundamental em softwares de simulações computacionais baseadas em modelos matemáticos para o avanço de pesquisas científicas contemporâneas. Consequentemente, há uma demanda crescente por métodos numéricos e implementações eficientes para enfrentar esse desafio, em particular no contexto da engenharia biomédica onde deseja-se utilizar esses simuladores para criar gêmeos digitais de pacientes e estudar determinadas condições patológicas. Este trabalho tem como objetivo explorar e identificar técnicas eficientes para resolver sistemas lineares relacionados ao problema da biomecânica cardíaca, acelerando assim as simulações relacionadas ao intrincado sistema cardiovascular humano. Para atingir esse objetivo, foram selecionados vários métodos multigrid disponíveis na biblioteca AmgX, que foram testados e analisados em termos do seu desempenho computacional. Como um passo inicial, problemas baseados na equação de Poisson, foram resolvidos considerando geometrias simplificadas e complexas como, por exemplo, um cubo e um ventrículo humano. Esse estudo revelou vantagens distintas associadas a cada método, dependendo da complexidade e do formato dos problemas em questão.

Palavras-chave

Modelagem computacional • Sistemas lineares • Métodos multigrid • GPU

Abstract

The solution of linear systems plays a fundamental role in computer simulation software based on mathematical models to advance contemporary scientific research. Consequently, there is a growing demand for numerical methods and efficient implementations to face this challenge, particularly in the context of biomedical engineering where it is desired to use these simulators to create digital twins of patients and study certain pathological conditions. This work aims to explore and identify efficient techniques to solve linear systems related to the problem of cardiac biomechanics, thus accelerating simulations related to the intricate human cardiovascular system. To achieve this goal, several multigrid methods available in the AmgX library were selected, which were tested and analyzed in terms of their computational performance. As an initial step, problems based on Poisson's equation were solved considering simplified and complex geometries such as a cube and a human left ventricle. This study revealed distinct advantages associated with each method, depending on the complexity and format of the problems at hand.

☆Este artigo é uma versão estendida do trabalho apresentado no XXVI ENMC Encontro Nacional de Modelagem Computacional e XIV ECTM Encontro de Ciência e Tecnologia de Materiais, ocorridos em Nova Friburgo – RJ, de 25 a 27 de outubro de 2023.

Keywords

Computational modeling • Linear systems • Multigrid methods • GPU

1 Introdução

O uso de simulações computacionais tem se mostrado bastante eficiente e promissor para a investigação de condições patológicas cardíacas nos últimos anos [1]. Entretanto, a complexidade dos processos biofísicos e fisiológicos do sistema cardiovascular se traduz em modelos matemáticos e computacionais complexos, os quais são descritos por sistemas de equações diferenciais parciais (EDPs) acoplados a sistemas de equações diferenciais ordinárias (EDOs), resultando em problemas com milhões de variáveis e muitos parâmetros.

Um problema comum em diversas aplicações de interesse prático que utilizam simulações computacionais é que muitas vezes essas simulações demandam um grande esforço computacional. Em muitas situações práticas, simulações podem demorar horas ou dias para executar, ou podem até mesmo serem tão grandes em termos de dados que a memória de apenas um computador é insuficiente. Um meio de reduzir este tempo de execução ou tornar a simulação viável é a utilização de computação paralela, que consiste em dividir o problema em partes menores a serem executadas concorrentemente, por exemplo, em diferentes processadores. Em geral esse tipo de abordagem traz uma grande redução no tempo de execução, permitindo que os problemas possam ser resolvidos rapidamente ou que problemas antes intratáveis, possam ser resolvidos. No contexto da modelagem computacional da atividade eletromecânica cardíaca, diversos trabalhos já exploraram dessas técnicas [2, 3, 4]. Entretanto, com o avanço da tecnologia e dos dispositivos de computação de alto desempenho, novas técnicas aliadas a métodos numéricos robustos e eficientes ainda precisam ser exploradas, sobretudo, no contexto da solução numérica do problema mecânico cardíaco.

O modelo matemático que descreve este problema geralmente é discretizado pelo método dos elementos finitos (MEF), resultando em sistemas não lineares a serem resolvidos. O método mais utilizado para resolver este sistema é o método de Newton, que requer a resolução de sistemas lineares por várias iterações até a convergência. A resolução destes sistemas lineares acaba sendo a parte mais custosa do processo de resolução. Portanto, o uso de técnicas eficientes para resolução de sistemas lineares resulta em simulações da mecânica cardíaca mais rápidas [2].

Com o avanço de aplicações 3D, como jogos que exibem imagens cada vez mais próximas da realidade, foi necessário um aprimoramento das unidades de processamento gráfico (GPUs). Observando o alto grau de paralelismo das GPUs, estas começaram a ser utilizadas na resolução de outros problemas, como simulações computacionais e aplicações na área de aprendizado de máquina [5], ambientes que estão sendo amplamente explorados no contexto atual. Outro exemplo da utilização de GPUs para a resolução de problemas complexos é a AmgX, biblioteca utilizada para a resolução de sistemas lineares criada pela NVIDIA, que será melhor explorada neste trabalho.

Neste contexto, o presente trabalho apresenta uma avaliação de métodos numéricos iterativos combinados a precondicionadores do tipo multigrid algébricos para a resolução de sistemas lineares, usando o poder computacional das GPUs.

2 Materiais e Métodos

Nesta seção são apresentados alguns conceitos fundamentais sobre os métodos multigrid e a biblioteca AmgX, que implementa diversos métodos deste tipo para a solução de sistemas lineares usando GPUs. Em seguida, apresenta-se os problemas resolvidos neste trabalho.

2.1 Multigrid

O método multigrid foi desenvolvido para a resolução de sistemas lineares na forma $\mathbf{Ax} = \mathbf{b}$, resultantes da discretização de equações diferenciais parciais. Nesta estratégia, métodos iterativos estacionários como os métodos de Jacobi e Gauss-Seidel são usados para suavizar o resíduo da solução $\mathbf{r} = \mathbf{Ax} - \mathbf{b}$. Depois de algumas iterações estes métodos não conseguem mais reduzir o resíduo, portanto, a solução é transferida para uma malha mais grosseira do problema. Nesta nova malha, as frequências do resíduo são realçadas e os métodos estacionários podem ser aplicados novamente para suavizar ainda mais o resíduo. O método produz malhas mais grosseiras até que seja possível resolver o sistema de forma direta, calculando-se o erro que será utilizado para corrigir a solução. Neste momento, inicia-se a etapa de correção, onde o erro é transferido para as malhas mais finas fazendo a correção da solução até a malha original do problema [6].

Uma versão deste método que explora a relação com a discretização do problema é denominada de multigrid geométrico. Existem diferentes estratégias de resolução e transferência de informação entre malhas, como os ciclos V, W ou F. Enquanto o ciclo V vai diretamente do nível mais fino ao nível mais grosseiro, o ciclo W alterna entre malhas finas e grosseiras até chegar no nível mais grosseiro. A Fig. 1 apresenta um exemplo da estratégia de ciclo V

com três níveis de malha. O método suaviza o resíduo na malha Ω^h e transfere a informação para a malha Ω^{2h}, então o processo é repetido para transferir as informações para o nível Ω^{4h}, onde é calculado o erro e se inicia o processo de correção até se chegar ao nível Ω^h novamente.

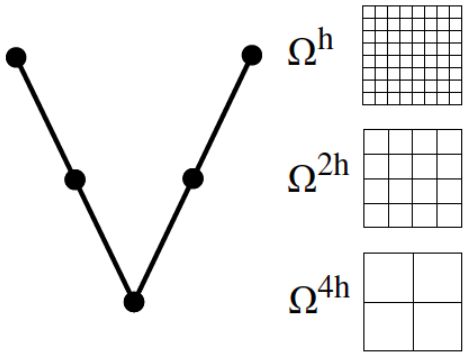


Figura 1: Ilustração do método multigrid ciclo V.

2.2 Método Multigrid Algébrico

As implementações testadas neste trabalho utilizam o método multigrid algébrico (AMG) para a obtenção de suas soluções. Diferentemente do multigrid geométrico, o AMG utiliza técnicas algébricas para construir malhas grosseiras a partir da matriz do sistema. Assim como as operações de transferências entre níveis se baseiam nesta estratégia. A principal vantagem da utilização desse método de multigrid é a sua melhor eficiência em problemas que apresentam geometrias complexas, e também sua capacidade de operar em malhas não-estruturadas, algo que não é feito em métodos de multigrid comuns [6].

Uma das estratégias relacionadas ao algoritmo é o AMG com base na agregação. Neste esquema, as variáveis da malha fina são agregadas de maneira não contínua em malhas grosseiras quando a matriz do sistema é construída. A matriz, então, é criada em uma única etapa, utilizando técnicas de correspondência de grafos [7].

Já no AMG de base clássica, a matriz é criada em duas fases. Primeiro identifica-se seu padrão de esparsidade durante o estágio de engrossamento, por exemplo, empregando a técnica *Parallel Maximal Independent Set* (PMIS). Já na interpolação, encontram-se seus valores não nulos, utilizando técnicas especializadas para esse passo. Nesse método, a interpolação pode resultar em uma configuração mais custosa, mas chega a melhores convergências durante a resolução [7].

2.3 Biblioteca AmgX

A biblioteca AmgX, desenvolvida por [7], fornece métodos iterativos precondicionados baseados no método AMG usando a plataforma CUDA (*Compute Unified Device Architecture*). A biblioteca implementa os métodos AMG clássicos e baseados em agregação, com diferentes estratégias de interpolação, além de uma variedade de suavizadores e precondicionadores. O algoritmo AMG presente na biblioteca atinge acelerações de 2 a 5 vezes em uma única GPU, quando comparada com uma implementação competitiva em processador [7].

Sobre a utilização da biblioteca AmgX em código, o primeiro passo consiste na criação de um objeto de configuração para a resolução do sistema. Nesse momento, a biblioteca AmgX apresenta grande flexibilidade para a escolha de métodos que se adequem ao tipo do problema em questão, sendo esses métodos previamente disponíveis na biblioteca ou criados por outros usuários. Esse objeto recebe as informações através de um arquivo informado no início da execução do programa.

Após isso, é criado um objeto que é passado por parâmetro para a função *solver*, sendo esse objeto um intermediário necessário para a execução da função. Essa função é responsável por aplicar o método escolhido pelo usuário na resolução do sistema linear, aproveitando a capacidade de processamento da GPU disponível. Dessa forma, é possível a obtenção de uma solução aproximada para o problema configurado.

Por fim, é representada a matriz do sistema linear na sua forma esparsa que, juntamente com o lado direito do sistema, devem ser carregados para a memória da GPU, onde os cálculos são realizados de forma eficiente. A AmgX fornece estruturas de dados para formatos específicos de matrizes esparsas, para facilitar esse processo de carregamento e manipulação dos dados na unidade de processamento gráfico.

2.4 Experimentos Computacionais

Nessa seção, serão apresentados alguns experimentos para avaliar a capacidade da biblioteca AmgX para a resolução de sistemas, sendo realizados testes dos experimentos mais simples, como o problema de poisson unidimensional, a problemas mais complexos, como o problema da mecânica cardíaca. Para isso, foi utilizada uma máquina com o processador de 12ª geração Intel Core i5-12400F de 6 núcleos, e uma GPU NVIDIA RTX 4070 com memória de 6GB.

2.4.1 Poisson 1D

Primeiramente, foram realizados testes para a solução do sistema de equações resultante da discretização da equação de Poisson em um domínio unidimensional com solução analítica dada por $u_{exata}(x) = 1 + x^2$. Esse problema é descrito pelo problema de valor de contorno dado pela Eq. (1):

$$u''(x) = 2, \quad x \in [-1, 1], \quad u(-1) = 2, \quad u(1) = 2. \tag{1}$$

Aplicando uma discretização do problema dado pela Eq. (1), chega-se ao sistema a ser resolvido apresentado pela Eq. (2), usando o método das diferenças finitas.

$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{bmatrix}\begin{bmatrix} u_1 \\ . \\ . \\ . \\ u_{n-1} \end{bmatrix} = \begin{bmatrix} 2 - \frac{\alpha}{h^2} \\ 2 \\ \vdots \\ 2 \\ 2 - \frac{\beta}{h^2} \end{bmatrix}, \tag{2}$$

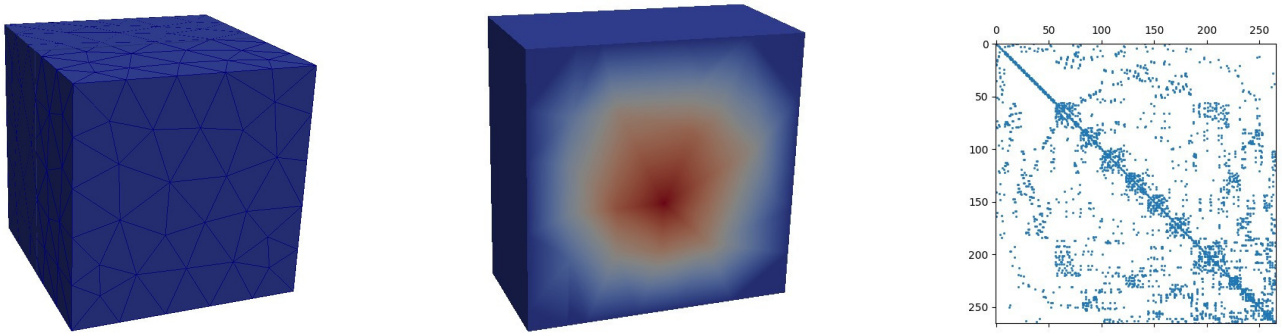
onde $\alpha = \beta = 2$ são os valores das condições de contorno do problema.

2.4.2 Poisson 3D

A resolução de sistemas lineares resultantes da discretização pelo MEF da equação de Poisson em domínios tridimensionais, dado pela Eq. (3), também foi considerada.

$$\nabla^2 u(x, y, z) = f(x, y, z), \tag{3}$$

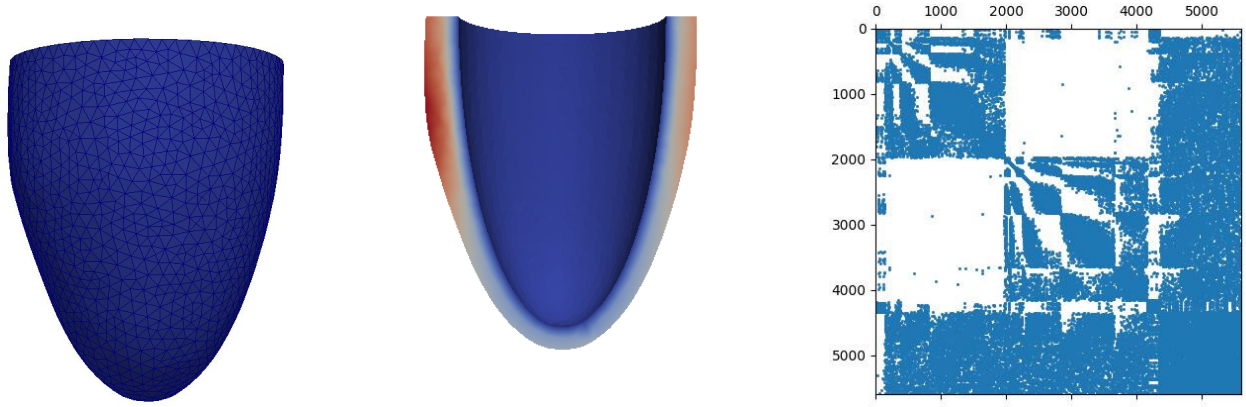
Primeiramente foi considerado um domínio no formato de um cubo unitário, discretizado com 992 elementos tetraédricos, com o seguinte termo forçante $f(x, y, z) = \sin(\pi x)\sin(\pi y)\sin(\pi z)$ e condição de contorno $u = 0$ para todo o contorno. A Fig. 2 apresenta a discretização do domínio cúbico, a solução do problema e o formato da matriz esparsa associada ao problema.



(a) (b) (c)

Figura 2: Solução da problema de Poisson 3D para o domínio no formato de cubo. (a) Malha de tetreaedros utilizada na discretização. (b) Solução do problema. (c) Representação da matriz esparsa do sistema linear associado ao problema.

Em seguida, o mesmo problema foi resolvido considerando como domínio uma geometria simplificada do ventrículo esquerdo do coração [8], como apresentada na Fig. 3. Esta geometria do ventrículo foi discretizada com 20655 elementos tetraédricos. Neste caso, a equação de Poisson foi resolvida considerando $f(x, y, z) = 1$ e $u = 0$ na superfície do endocárdio (cavidade do ventrículo).



(a) (b) (c)

Figura 3: Solução da problema de Poisson 3D para o domínio do ventrículo. (a) Malha de tetreaedros utilizada na discretização. (b) Solução do problema. (c) Representação da matriz esparsa do sistema linear associado.

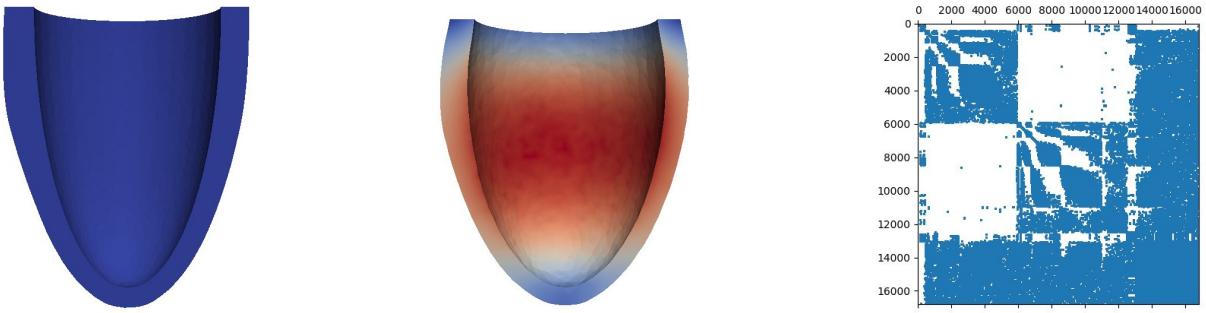
2.4.3 Mecânica Cardíaca

Por fim, foram realizados testes sobre a resolução de sistemas lineares envolvidos no processo de solução de um problema da mecânica cardíaca, o qual é apresentado na Eq. (4).

$$\begin{cases} \text{div}\boldsymbol{\sigma} &= 0, \quad \text{em } \Omega, \\ \mathbf{u} &= \overline{\mathbf{u}}, \quad \text{sobre } \partial\Omega^D, \\ \boldsymbol{\sigma}\mathbf{n} &= \overline{\mathbf{t}}, \quad \text{sobre } \partial\Omega^N. \end{cases} \tag{4}$$

A resolução deste problema busca encontrar o campo de deslocamentos $\mathbf{u}$ no domínio Ω, dados os deslocamentos prescritos no contorno $\partial\Omega^D$ e a pressão aplicada no contorno $\partial\Omega^N$. O tensor de tensões $\boldsymbol{\sigma}$ é descrito por um modelo constitutivo hiperelástico, que representa o comportamento mecânico do tecido cardíaco quando submetido a alguma carga. A discretização deste problema resulta em um sistema não linear a ser resolvido para cada incremento de carga. Os sistemas não lineares são resolvidos através do método de Newton e, portanto, a cada iteração é necessário resolver um sistema linear.

Neste experimento, foram feitos testes sobre a resolução de um sistema linear de uma das iterações do método de Newton que resolve o problema da mecânica cardíaca. O domínio considerado foi a mesma geometria do ventrículo esquerdo, apresentada anteriormente, porém, neste caso o problema tem três incógnitas por nó da malha, que são os deslocamentos em cada direção. A Fig. 4a apresenta a geometria do ventrículo na configuração indeformada, enquanto a Fig. 4b apresenta a configuração deformada, com os deslocamentos encontrados pela resolução do sistema linear. Além disso, a Fig. 4c mostra a estrutura da matriz esparsa associada ao problema resolvido.



(a) (b) (c)

Figura 4: Simulações de ventrículos relacionados ao problema da mecânica cardíaca. (a) Geometria na configuração indeformada. (b) Geometria na configuração deformada, onde as cores representam o campo de deslocamentos. (c) Estrutura da matriz esparsa associada ao problema.

3 Resultados e Discussão

Os sistemas lineares envolvidos nos problemas descritos anteriormente foram resolvidos para todas as configurações de métodos disponíveis na biblioteca AmgX. Existem 66 configurações diferentes, que variam o método de resolução e o precondicionador utilizado. Os valores das propriedades de cada configuração, tais como tolerância para o resíduo, número de níveis e máximo de iterações, foram usados conforme disponibilizados pela biblioteca.

Nesta seção apresenta-se o desempenho dos cinco melhores métodos para cada problema, que foram ranqueados pelo menor resíduo e tempo de execução obtidos. A Tabela 1 apresenta as descrições de todos os métodos explorados nessa seção.

3.1 Resultados para o problema Poisson 1D

Para este experimento, foi utilizada uma discretização bem refinada do domínio unidimensional, resultando em um sistema com 10 milhões de incógnitas. A Tabela 2 apresenta o desempenho dos melhores métodos para a resolução do sistema resultante deste problema. Pode-se notar que o método GMRES flexível (FGMRES) combinado com o precondicionador multigrid algébrico com esquema de agregação teve o tempo e o resíduo mais baixo para este problema. Nesse caso, mesmo tendo um número elevado de incógnitas, a resolução desse sistema é mais fácil devido à sua natureza tridiagonal e foi realizada neste trabalho como passo inicial.

3.2 Resultados para o problema de Poisson 3D

A Tabela 3 apresenta o resultado da resolução dos sistemas envolvidos na solução do problema de Poisson em um domínio 3D representado por um cubo, que possui apenas 266 incógnitas. Nesse caso, todos os métodos retornaram resíduo na ordem de 10^{-17} e o método Gradiente Conjugado Precondicionado com DILU (PCG_DILU) foi o que obteve o melhor resíduo. Cabe ressaltar que novamente o FGMRES precondicionado pelo multigrid algébrico com esquema de agregação aparece entre os melhores métodos, obtendo o melhor tempo de execução e resíduo na mesma ordem de grandeza.

A Tabela 4 apresenta a solução de um novo sistema baseado na mesma equação que o experimento anterior, porém com o domínio sendo da geometria de um ventrículo. Esse sistema apresentou um total de 5602 incógnitas, onde foi possível obter soluções com resíduo de ordens entre 10^{-13} e 10^{-18}. Neste experimento, o método FGMRES_AGGREGATION_DILU obteve o menor resíduo, com um bom tempo de execução. Porém, o menor tempo foi obtido pelo método GMRES_AMG_D2, que também obteve um valor satisfatório para o resíduo final.

3.3 Resultados para o problema da Mecânica Cardíaca

O experimento envolvendo o problema da mecânica cardíaca possui um total de 16806 incógnitas, com propriedades mais difíceis para a solução via métodos numéricos iterativos. Portanto, foi necessário alterar o critério de parada dos métodos, que passou a usar o esquema de tolerância absoluta do resíduo, com valor de 10^{-5}. A Tabela 5 apresenta os resultados obtidos pelos melhores métodos, onde pode-se notar que os resíduos reportados ficaram na ordem de 10^{-6} e 10^{-7}.

Tabela 1: Descrição dos métodos da AmgX que foram utilizados.

Método	Descrição
AGG_CHEB4	Multigrid geométrico com esquema de agregação Chebyshev de ordem 4.
AMG_CL_AGG_CHEB_L1_TRUNC	AMG pré-condicionado por multigrid algébrico com esquema clássico, agregação Chebyshev e truncamento L1 (interpolação truncada até a primeira ordem).
AMG_CLASSICAL_CG	CG pré-condicionado por multigrid algébrico com ciclos simples de multigrid.
AMG_CLASSICAL_CGF	CGF pré-condicionado por multigrid algébrico com ciclos completos de multigrid.
CLASSICAL_CGF_CYCLE	Método clássico Gradiente Conjugado (CG) com ciclos completos de multigrid geométrico.
FGMRES	GMRES (Generalized Minimal Residual Method) flexível pré-condicionado por multigrid algébrico.
FGMRES_AGGREGATION	GMRES flexível pré-condicionado por multigrid algébrico com esquema de agregação.
FGMRES_AGGREGATION_DILU	GMRES flexível pré-condicionado por multigrid algébrico com esquema de agregação e suavizador DILU (Diagonal Incomplete LU).
FGMRES_CL_AGG_HMIS	GMRES flexível pré-condicionado pelo AMG com esquema clássico e HMIS (Hybrid Modified Independent Set).
FGMRES_CL_AGG_PMIS	GMRES flexível pré-condicionado pelo AMG com esquema clássico e PMIS (Parallel Modified Independent Set).
GMRES_AMG_D2	GMRES pré-condicionado por AMG com interpolação de distância 2.
PBICGSTAB_AGG_W_JACOBI	Método PBiCGSTAB pré-condicionado por multigrid algébrico com esquema de agregação e relaxamento Jacobi.
PBICGSTAB_NOPREC	Método do Gradiente Bi-Conjugado Estabilizado (BiCGStab) sem precondicionador.
PBICGSTAB_W	Método PBiCGSTAB pré-condicionado por multigrid algébrico com relaxamento W.
PCG_CLASSICAL_F_JACOBI	Método PCG com esquema clássico de relaxamento Jacobi.
PCG_DILU	Método Gradiente Conjugado Pré-condicionado (PCG) com precondicionador DILU.
IDR_DILU	IDR (Induced Dimension Reduction) com precondicionador DILU

Tabela 2: Resultado dos 5 melhores métodos para o problema de Poisson 1D, apresentando número de níveis de malha usados no AMG, número de iterações, resíduo final e tempo de execução.

Método	Níveis	Iterações	Resíduo	Tempo (s)
FGMRES_AGGREGATION	17	100	4.32E-10	8.667
FGMRES_AGGREGATION_DILU	17	100	4.32E-10	8.671
CLASSICAL_CGF_CYCLE	15	23	6.49E-07	50.359
AMG_CLASSICAL_CGF	15	23	6.49E-07	50.324
AMG_CLASSICAL_CG	15	23	6.49E-07	44.974

Tabela 3: Desempenho dos 5 melhores métodos resolvendo o sistema da equacão de Poisson em um domínio 3D no formato de um cubo.

Método	Níveis	Iterações	Resíduo	Tempo (s)
PCG_DILU	0	1	2.47E-17	0.0188
FGMRES_AGGREGATION_DILU	1	1	8.23E-17	0.0217
FGMRES	1	1	8.23E-17	0.0176
FGMRES_AGGREGATION	1	1	8.23E-17	0.0209
PCG_CLASSICAL_F_JACOBI	1	1	8.74E-17	0.0234

Tabela 4: Comportamento dos métodos usados para resolver o sistema linear resultante do problema de Poisson usando como domínio a geometria de um ventrículo.

Método	Níveis	Iterações	Resíduo	Tempo (s)
FGMRES_AGGREGATION_DILU	6	11	9.08E-18	0.0439
FGMRES_AGGREGATION	6	11	9.08E-18	0.0492
IDR_DILU	0	42	3.07E-15	0.0567
GMRES_AMG_D2	8	9	2.71E-13	0.0309
FGMRES_CL_AGG_HMIS	4	13	2.71E-13	0.0323

Neste caso, todos os métodos retornaram resíduos similares, abaixo da tolerância especificada, onde o método PBICGSTAB_NOPREC obteve o menor tempo de execução. Nesta configuração não foi utilizado precondicionador, portanto, o número de níveis do AMG foi 0, o que pode ter contribuído para um tempo menor. Em segundo lugar, aparece o método AMG precondicionado com mutigrid algébrico clássico, esquema de agregação Chebyshev e truncamento L1, que apresentou o menor resíduo e tempo de execução ligeiramente maior.

Tabela 5: Desempenho dos métodos na resolução de um sistema linear envolvido na solução do problema da mecânica cardíaca, usando o domínio de um ventrículo simplificado.

Método	Níveis	Iterações	Resíduo	Tempo (s)
AMG_CL_AGG_CHEB_L1_TRUNC	5	87	9.79E-06	0.0937
FGMRES_CL_AGG_HMIS	5	74	9.88E-06	0.1225
FGMRES_CL_AGG_PMIS	5	74	9.96E-06	0.0965
PBICGSTAB_NOPREC	0	85	9.97E-06	0.0456
AGG_CHEB4	3	100	9.98E-06	0.1219

4 Conclusões

Este trabalho apresentou um estudo de desempenho da biblioteca AmgX para solução de sistema de equações lineares resultantes da discretização de equações diferenciais parciais. Em particular, focou-se em problemas e geometrias no contexto da biomecânica cardíaca. Todos os métodos da biblioteca AmgX foram avaliados nos problemas selecionados e os cinco melhores em cada caso foram classificados.

Após detalhada análise das informações apresentadas nesse artigo, é possível concluir que, para a resolução eficiente do problema em foco, sendo esse o problema da mecânica cardíaca, os melhores métodos são o Gradiente Bi-Conjugado Estabilizado sem precondicionador e o AMG precondicionado com mutigrid algébrico clássico, esquema de agregação Chebyshev e truncamento L1, apresentando resíduos suficientemente pequenos com bom tempo de

execução. Além disso, o método GMRES flexível precondicionado pelo AMG com esquema de agregação aparece com bons resultados para os diferentes problemas testados.

Essas análises serão importantes para que seja facilitada a utilização dos métodos da AmgX em projetos que dependem de resoluções de grandes sistemas lineares, como por exemplo o simulador da mecânica cardíaca [9], aplicado a simulações personalizadas por paciente.

Referências

[1] K. Gillette, M. A. Gsell, A. J. Prassl, E. Karabelas, U. Reiter, G. Reiter, T. Grandits, C. Payer, D. Štern, M. Urschler *et al.*, "A framework for the generation of digital twins of cardiac electrophysiology from clinical 12-leads ECGs," *Medical Image Analysis*, vol. 71, p. 102080, 2021. Disponível em: https://doi.org/10.1016/j.media.2021.102080

[2] F. Viola, G. Del Corso, R. De Paulis, e R. Verzicco, "Gpu accelerated digital twins of the human heart open new routes for cardiovascular research," *Scientific Reports*, vol. 13, no. 1, p. 8230, 2023. Disponível em: https://doi.org/10.1038/s41598-023-34098-8

[3] J. O. Campos, R. S. Oliveira, R. W. dos Santos, e B. M. Rocha, "Lattice Boltzmann method for parallel simulations of cardiac electrophysiology using GPUs," *Journal of Computational and Applied Mathematics*, vol. 295, pp. 70–82, 2016. Disponível em: https://doi.org/10.1016/j.cam.2015.02.008

[4] B. M. Rocha, F. O. Campos, R. M. Amorim, G. Plank, R. d. Santos, M. Liebmann, e G. Haase, "Accelerating cardiac excitation spread simulations using graphics processing units," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 7, pp. 708–720, 2011. Disponível em: https://doi.org/10.1002/cpe.1683

[5] M. Pandey, M. Fernandez, F. Gentile, O. Isayev, A. Tropsha, A. C. Stern, e A. Cherkasov, "The transformational role of GPU computing and deep learning in drug discovery," *Nature Machine Intelligence*, vol. 4, no. 3, pp. 211–221, 2022. Disponível em: https://doi.org/10.1038/s42256-022-00463-x

[6] W. L. Briggs, V. E. Henson, e S. F. McCormick, *A multigrid tutorial*, 2ª ed. Philadelphia, USA: SIAM, 2000. Disponível em: https://epubs.siam.org/doi/pdf/10.1137/1.9780898719505.fm

[7] M. Naumov, M. Arsaev, P. Castonguay, J. Cohen, J. Demouth, J. Eaton, S. Layton, N. Markovskiy, I. Reguly, N. Sakharnykh, V. Sellappan, e R. Strzodka, "AmgX: A library for GPU accelerated algebraic multigrid and preconditioned iterative methods," *SIAM Journal on Scientific Computing*, vol. 37, no. 5, pp. S602–S626, 2015. Disponível em: https://doi.org/10.1137/140980260

[8] J. O. Campos, J. Sundnes, R. W. dos Santos, e B. M. Rocha, "Effects of left ventricle wall thickness uncertainties on cardiac mechanics," *Biomechanics and Modeling in Mechanobiology*, vol. 18, no. 5, pp. 1415–1427, 2019. Disponível em: https://doi.org/10.1007/s10237-019-01153-1

[9] J. O. Campos, R. W. Dos Santos, J. Sundnes, e B. M. Rocha, "Preconditioned augmented lagrangian formulation for nearly incompressible cardiac mechanics," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 34, no. 4, p. e2948, 2018. Disponível em: https://doi.org/10.1002/cnm.2948