4
Solução Numérica da Equação de Poisson Usando SPH

4.1
Equação de Poisson

Seja Ω uma região em $\mathbb{R}^2$ limitada por uma superfície fechada $S = \partial\Omega$. Nesta seção, é apresentado um método para resolver a **equação de Poisson**

$$\Delta u\,(x,y) = f\,(x,y) \tag{4-1}$$

em Ω, onde f é uma função qualquer. A solução $u\,(x,y)$ é chamada de potencial.

A solução completa da equação de Poisson depende dos valores do potencial na fronteira do domínio em estudo. Os dois tipos mais comuns de condições de fronteira para a equação de Poisson são as condições de Dirichlet e Neumann. Quando o potencial u é explicitamente definido em todo ponto $(x,y) \in S$ tem-se a condição de fronteira de *Dirichlet*

$$u\,(x,y) = \phi\,(x,y)\ ,\ \forall\,(x,y) \in S\,. \tag{4-2}$$

Por outro lado, na condição de fronteira de *Neumann*, a variação do potencial na direção perpendicular à fronteira é conhecida

$$\nabla u\,(x,y) \cdot \eta\,(x,y) = \varphi(x,y)\,,\ \forall\,(x,y) \in S\ . \tag{4-3}$$

O vetor $\eta\,(x,y)$ é o vetor normal à fronteira S no ponto (x,y) apontando para fora da superfície.

A equação de Poisson é uma equação diferencial parcial utilizada em várias áreas, dentre elas: matemática, física e engenharia. Em eletrostática, por exemplo, pode-se escrever o campo elétrico E em termos de um potencial elétrico ϕ

$$E = -\nabla\phi$$

dado pela solução da equação de Poisson

$$\Delta\phi\,(\mathbf{r}) = -\frac{\rho\,(\mathbf{r})}{\epsilon}\,, \tag{4-4}$$

onde $\rho\,(\mathbf{r})$ é a densidade de carga em $\mathbf{r}$ e ϵ é uma constante elétrica.

Em dinâmica dos fluídos, a pressão $p\,(\mathbf{r})$ em um escoamento incom-

pressível satisfaz a equação de Poisson

$$\Delta p\left(\mathbf{r}\right) = F\left(\mathbf{r}\right) \ , \tag{4-5}$$

onde a função $F\left(\mathbf{r}\right)$ depende da velocidade do escoamento e de alguns parâmetros físicos do fluido.

Encontrar uma solução analítica para uma equação de Poisson é em geral uma tarefa díficil e, portanto, recorre-se a métodos númericos para encontrar soluções em um conjunto discreto de pontos no domínio Ω. Pode-se citar a utilização dos métodos das diferenças finitas (38, 50) e dos elementos finitos (13).

Esta seção propõe um método para resolver a equação de Poisson utilizando o método lagrangeano SPH. O operador laplaciano na equação de Poisson (4-1) será substituído pela versão discreta dada pelo operador laplaciano SPH. A aproximação em cada partícula que representa o domínio de interesse conduz, em geral, à obtenção de um sistema de grande dimensão, caracterizado por um alto grau de esparsidade. A solução desse sistema, por algum método numérico, encontra o potencial associado à equação de Poisson em cada partícula do sistema.

4.2
Equação de Poisson Discreta usando SPH

Nos métodos SPH, o domínio é discretizado por um conjunto finito de partículas. Dada uma função escalar ϕ, o operador laplaciano SPH na partícula i é dado por

$$\left(\Delta\phi\right)_h\left(\mathbf{x}_i\right) = \sum_{j\in\mathtt{V}_i}\frac{m_j}{\rho_j}\frac{2\left(\phi\left(\mathbf{x}_i\right)-\phi\left(\mathbf{x}_j\right)\right)}{\left|\mathbf{x}_{ij}\right|^2}\mathbf{x}_{ij}\cdot\nabla_i W\left(\mathbf{x}_{ij}\right) \ , \tag{4-6}$$

onde $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$.

O núcleo quártico utilizado nesta tese é um núcleo esfericamente simétrico, podendo ser reescrito como

$$W\left(\mathbf{x}_{ij}, h\right) = W_1\left(\frac{\left|\mathbf{x}_{ij}\right|}{h}\right) = W_1\left(R_{ij}\right) \ .$$

Donde,

$$\nabla_i W_{ij} = \frac{\mathbf{x}_{ij}}{h|\mathbf{x}_{ij}|}\frac{\partial W_{ij}}{\partial R} \ .$$

Conseqüentemente, a equação 4-6 é reescrita como

$$\left(\Delta\phi\right)_h\left(\mathbf{x}_i\right) = \sum_{j\in\mathtt{V}_i} 2\frac{m_j}{\rho_j}\left(\phi\left(\mathbf{x}_i\right)-\phi\left(\mathbf{x}_j\right)\right)\frac{1}{h|\mathbf{x}_{ij}|}\frac{\partial W_{ij}}{\partial R} \ ,$$

ou, de uma maneira mais simplificada, dada por

$$(\Delta\phi)_h(\mathbf{x}_i) = \sum_{j\in \mathtt{V}_i} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{ij}) \;, \tag{4-7}$$

onde

$$F(\mathbf{x}_{ij}) = \frac{1}{h|\mathbf{x}_{ij}|}\frac{\partial W_{ij}}{\partial R}\,.$$

Discretizando a equação de Poisson (4-1) na partícula i, obtemos a seguinte igualdade:

$$(\Delta\phi)_h(\mathbf{x}_i) = \sum_{j\in \mathtt{V}_i} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{ij}) = f(\mathbf{x}_i) \;.$$

A equação linear

$$\sum_{j\in \mathtt{V}_i} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{ij}) = f(\mathbf{x}_i) \tag{4-8}$$

aplicada em cada partícula i do domínio do problema resulta num sistema de equações lineares

$$\begin{cases} \sum\limits_{j\in \mathtt{V}_1} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_1) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{1j}) &= f(\mathbf{x}_1) \\ \sum\limits_{j\in \mathtt{V}_2} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_2) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{2j}) &= f(\mathbf{x}_2) \\ \vdots & \\ \sum\limits_{j\in \mathtt{V}_i} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{ij}) &= f(\mathbf{x}_i) \\ \vdots & \\ \sum\limits_{j\in \mathtt{V}_n} 2\frac{m_j}{\rho_j}\left(\phi(\mathbf{x}_n) - \phi(\mathbf{x}_j)\right) F(\mathbf{x}_{nj}) &= f(\mathbf{x}_n) \end{cases} \tag{4-9}$$

onde n é o número de partículas que discretizam o domínio e

$$\mathtt{V}_i = \mathtt{V}(x_i) = \{x_j, |x_j - x_i| \leqslant \kappa h\}$$

é o conjunto de partículas vizinhas à partícula i.

O sistema linear de equações (4-9), obtido a partir da aplicação do operador laplaciano SPH em cada partícula i do domínio do problema, é representado matricialmente por

$$A \cdot \mathbf{x} = \mathbf{b}\,, \tag{4-10}$$

sendo n o número de partículas que discretizam o domínio, a dimensão da matriz A é $n \times n$, o vetor solução $\mathbf{x}$ de dimensão n é um vetor das variáveis

do sistema

$$x_i = \phi_i = \phi\left(\mathbf{x}_i\right)$$

e o vetor independente $\mathbf{b}$, também de dimensão n, é dado por

$$b_i = f\left(\mathbf{x}_i\right) .$$

A i-ésima linha da matriz A, representada pelos coeficientes

$$a_{ij} , \ j \in \{1, 2, \cdots, n\} ,$$

tem a seguinte forma

$$\begin{cases} a_{ij} & = & -2\frac{m_j}{\rho_j} F\left(\mathbf{x}_{ij}\right) , \quad j \neq i \\ \\ a_{ii} & = & -\sum\limits_{k \neq i} a_{ik} \end{cases}$$

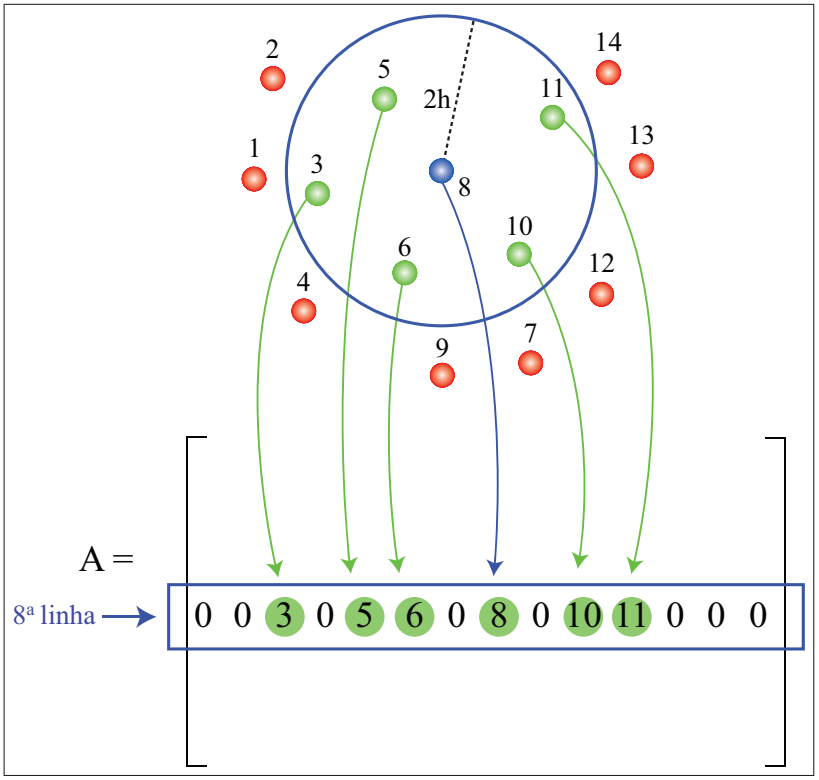

Figura 4.1: Construção da matriz A. A linha 8 da matriz tem como coeficientes não-nulos apenas as colunas, cujas partículas de mesmos índices pertencem à vizinhança $V_8 = \{3, 5, 6, 8, 10, 11\}$ da partícula 8.

Pela construção da matriz A podemos observar que na i-ésima linha da mesma o coeficiente a_{ij} na coluna j é não-nulo se, e somente se, a partícula j é vizinha da partícula i. A Figura 4.1 ilustra um simples exemplo.

Portanto, a matriz A é uma matriz com um elevado grau de esparsidade, já que em SPH o número de partículas vizinhas utilizadas no somatório da equação (4-7) pode ser consideravelmente menor do que o número de partículas usadas na discretização do domínio.

Para assegurar a eficiência numérica, torna-se imprescindível a utilização de algoritmos especialmente desenvolvidos para permitir um tratamento eficaz de matrizes esparsas. Para explorar ao máximo as características dessas matrizes e, ao mesmo tempo, garantir um bom desempenho na solução do sistema, esses algoritmos devem respeitar as seguintes regras básicas:

1. somente os coeficientes não-nulos presentes nas matrizes devem ser armazenados; e

2. efetuar somente as operações envolvendo coeficientes não-nulos, evitando cálculos redundantes envolvendo elementos nulos.

Os algoritmos que permitem trabalhar diretamente com matrizes esparsas são mais complexos do que os algoritmos utilizados no tratamento de matrizes cheias. Essa complexidade resulta não só da necessidade de se evitar sequências de operações redundantes, mas também do tipo de armazenamento utilizado. Foram utilizadas as bibliotecas `SparseLib++` e `IML++` para o armazenamento e manuseio de matrizes esparsas e para encontrar a solução do sistema linear.

4.3 Matrizes Esparsas

Nas formulações antigas de elementos finitos, as matrizes esparsas são geralmente armazenadas em semi-banda ou em perfil. No método SPH proposto, a utilização de tais estruturas de armazenamento é inadequada. A distribuição irregular das partículas faria com que qualquer uma dessas estruturas tivessem um elevado número de coeficientes nulos armazenados.

Exitem outros tipos diferentes de armazenamento de matrizes esparsas utilizados. Dentre eles, encontra-se uma forma bem simples e direta de armazenamento de matrizes esparsas, onde se utilizam três vetores com dimensão igual ao número de elementos não-nulos da matriz, denotado por nz. No primeiro vetor, $\mathbf{V}$, armazenam-se os valores dos coeficientes não-nulos existentes. Os índices correspondentes às linhas e colunas de cada um dos coeficientes não-nulos são armazenados em outros dois vetores, $\mathbf{I}$ e $\mathbf{J}$, respectivamente. Assim, para cada $k \in \{1, 2, \cdots, nz\}$ o valor não-nulo $\mathbf{VAL}(k)$ ocorre na posição $(\mathbf{I}(k), \mathbf{J}(k))$ da matriz esparsa.

A matriz de tamanho 5×5

$$B = \begin{bmatrix} 3 & 0 & 0 & -1 & 0 \\ 0 & 5 & 1 & 0 & 0 \\ 0 & 1 & 3 & 2 & 0 \\ -1 & 0 & 2 & 4 & 0 \\ 1 & 0 & 0 & 0 & 2 \end{bmatrix} \tag{4-11}$$

pode ser armazenada da seguinte forma

$$\begin{aligned} \mathbf{V} &= [\ 5\ 4\ 3\ 3\ 2\ 2\ 2\ 1\ 1\ -1\ -1\ 1\ ] \\ \mathbf{I} &= [\ 2\ 4\ 1\ 3\ 3\ 4\ 5\ 2\ 3\ 1\ 4\ 5\ ] \\ \mathbf{J} &= [\ 2\ 4\ 1\ 3\ 4\ 3\ 5\ 3\ 2\ 4\ 1\ 1\ ] \end{aligned} \tag{4-12}$$

A estrutura de dados **V**, **I** e **J** é muitas vezes denominada de *esquema coordenado de armazenameto*. Esse tipo de armazenamento não leva em conta que os vetores tenham algum tipo de ordem, e embora seja muito simples, apresenta uma grande limitação: torna bastante pesado e moroso o processo de obtenção de todos os coeficientes situados numa mesma linha ou coluna.

Uma das estruturas de armazenamento mais utilizadas consiste em definir uma lista de vetores correspondente a cada uma das linhas, armazenados seqüencialmente. Dessa forma, a representação de matrizes esparsa é feita às custas de três vetores. O primeiro vetor **VL** é definido de maneira semelhante ao vetor **V**, exceto que todos os elementos da linha i estejam antes de qualquer elemento de qualquer linha $i+k$, $k > 0$. O segundo vetor **JL** representa a coluna associada a cada item listado no vetor **VL**. Por último, o vetor **IL** permite identificar onde iniciam e terminam, em **VL** e **JL**, os valores referentes a cada uma das linhas da matriz esparsa.

As dimensões dos vetores **VL** e **JL** são iguais ao número nz de coeficientes não-nulos da matriz esparsa, porém, a dimensão do vetor **IL** é igual ao número de linhas da matriz esparsa, geralmente menor do que nz, mais uma entrada adicional para identificar o final dos vetores **VL** e **JL**. Por exemplo, a matriz B pode então ser armazenada pelos vetores

$$\begin{aligned} \mathbf{VL} &= [\ 3\ -1\ 5\ 1\ 1\ 3\ 2\ -1\ 2\ 4\ 1\ 2\ ] \\ \mathbf{JL} &= [\ 1\ 4\ 2\ 3\ 2\ 3\ 4\ 1\ 3\ 4\ 1\ 5\ ] \end{aligned} \tag{4-13}$$

$$\mathbf{IL} = [\ 1\ 3\ 5\ 8\ 11\ 13\ ]$$

O armazenamento usando a tripla de vetores **IL**, **VL** e **JL** é conhecido como *armazenamento compacto de linhas*. Nesse processo as linhas têm de ser armazenadas por ordem, porém, dentro de uma mesma linha não é necessário

nenhuma ordenação. De acordo com a definição dos apontadores indicados pelo vetor **IL**, a linha i de uma matriz esparsa é descrita pelas posições $\mathbf{IL}(i)$ até $\mathbf{IL}(i+1)-1$ dos vetores **VL** e **JL**. Por exemplo, a terceira linha da matriz B é armazenada entre as posições

$$\mathbf{IL}(3) = 5$$

e

$$\mathbf{IL}(3+1) - 1 = 8 - 1 = 7$$

dos vetores **VL** e **JL**, isto é,

$$\begin{aligned} \mathbf{VL} &= [\ 3\ \ -1\ \ 5\ \ 1\ \ 1\ \ 3\ \ 2\ \ -1\ \ 2\ \ 4\ \ 1\ \ 2\ ] \\ \mathbf{JL} &= [\ 1\ \ 4\ \ 2\ \ 3\ \ 2\ \ 3\ \ 4\ \ 1\ \ 3\ \ 4\ \ 1\ \ 5\ ] \end{aligned}$$

Quando $\mathbf{IL}(l) = \mathbf{IL}(l+1) - 1$ a l-ésima linha da matriz esparsa não contém nenhum termo não-nulo.

O armazenamento em lista de vetores foi utilizado para guardar a matriz esparsa resultante da discretização da equação de Poisson em todas as partículas que representam o domínio do problema. A biblioteca `SparseLib++` (88) foi utilizada para tal propósito. A biblioteca `SparseLib++`, além de representar uma matriz esparsa em vários formatos de armazenamento, dentre eles o armazenamento compacto de linhas, também armazena vetores e fornece as operações básicas entre matrizes esparsas ou matrizes esparsas e vetores de maneira eficiente, como por exemplo multiplicação entre uma matriz esparsa e um vetor.

4.4 Métodos iterativos

A necessidade de resolver sistemas de equações lineares aparece numa grande quantidade de problemas científicos. Vários pesquisadores têm publicado artigos que exaltam as virtudes dos métodos iterativos, em comparação aos métodos diretos para a solução de grandes sistemas de equações lineares. Muitos métodos demonstram potencialidades para solucionar tais problemas. A questão crucial é achar aquele que melhor se adapta ao problema que se tem em mãos, pois não há garantias de que um método que funciona bem para um tipo de problema irá funcionar bem para outro tipo. Vários métodos iterativos para solução de sistemas lineares e outros tópicos importantes são apresentados por Saad (91).

Os métodos iterativos trabalham pelo melhoramento contínuo da solução aproximada até que esta esteja precisa o suficiente. Esse amplo grupo de

técnicas utiliza aproximações sucessivas para chegar a soluções mais precisas a cada passo, para um dado sistema linear de equação.

A eficiência dos métodos iterativos já tem sido estudada para a solução de sistemas densos, não-simétricos. Pesquisadores como Freund e Nachtigal (34), Golub e Van Loan (40) e Hageman e Young (42) contribuíram enormemente para o estudo e desenvolvimento das técnicas iterativas.

Quando a dimensão e a esparsidade dos sistemas governantes aumentam, os métodos iterativos começam a ser competitivos. Embora a convergência do processo envolva um número indeterminado de operações, e seja necessário discutir qual o critério de parada mais adequado, a sua utilização começa a ser atrativa pela enorme economia que podem proporcionar em termos de memória envolvida no armazenamento dos coeficientes. Tipicamente, a aplicação de um método iterativo implica apenas o armazenamento da matriz dos coeficientes na sua forma inicial e de um pequeno número de vetores de dimensão igual à dimensão do sistema.

Não é só a redução da necessidade de armazenamento que torna a utilização de algoritmos iterativos atraente. Também se consegue assegurar em muitas circunstâncias melhores tempos de execução. Deve-se chamar, no entanto, a atenção para o fato de a aplicação dos algoritmos iterativos na sua forma pura não ser em geral muito eficiente. Para que tais métodos sejam competitivos, é necessário utilizar formas para acelerar a convergência.

A razão pela qual um método iterativo converge depende das propriedades do espectro da matriz dos coeficientes. Por esse motivo deve-se procurar transformar o sistema linear em outro equivalente, no sentido de possuir a mesma solução, mas com propriedades do espectro mais favoráveis. Portanto, os métodos iterativos usualmente envolvem uma segunda matriz, que transforma a matriz dos coeficientes em outra com melhores propriedades. Essa matriz de transformação é denominada pré-condicionador.

Uma apresentação mais formal dos métodos iterativos pode ser encontrada em Barrett et al. (6) e inclui os seguintes métodos iterativos.

- Gradiente Conjugado
- Resíduo Mínimo Generalizado
- Gradiente Biconjugado
- Resíduo Quase-Mínimo
- Gradiente Conjugado Quadrado
- Gradiente Biconjugado Estabilizado

O método Gradiente Conjugado, CG (Conjugate Gradient), é o método iterativo mais popular para resolver grandes sistemas de equações lineares. Só

pode ser aplicado, porém, quando a matriz relacionada ao sistema é simétrica e positiva-definida.

Na presente tese, as matrizes relacionadas à discretização da equação de Poisson, além do alto grau de esparsidade, são assimétricas. Um método iterativo aplicável a matrizes assimétricas é o de Resíduo Quase-Mínimo, QMR (Quasi-Minimal Residual). A principal idéia por trás do método QMR é a solução do sistema tridiagonal, reduzido num senso dos mínimos quadrados. Freund e Nachtigal (34) fornecem uma apresentação do método QMR com mais detalhes.

A biblioteca `IML++` (23) foi utilizada para resolver o sistema de equações lineares obtido com a equação de Poisson discreta. Vários métodos iterativos são encontrados nessa biblioteca. Além desses, alguns pré-condicionadores também são implementados, dentre eles: Diagonal de Jacobi, Cholesky e LU.