

Ciências e Tecnologias Espaciais

Sensores e Atuadores Espaciais

**Métodos Numéricos e Aplicações em
Clusters I – Básico**

Lista de Exercícios 5

Professor: Angelo Passaro
Aluno: Lucas Kriesel Sperotto

31 de Maio de 2012

1 – Resolver equação de Laplace para o problema de placa quadrada com condições de contorno $C_1 = 100$, $C_2 = 50$, $C_3 = 50$, $C_4 = 75$, através do método “**Fixed Random Walk**”, baseando-se nos capítulos 2 e 4 da referência [1].

Modelagem Matemática:

Como exposto em [1], a aplicação do método “Fixed Random Walk” (FRW) envolve três passos:

- 1) Obter as probabilidades transitórias resultantes da equação de diferenças finitas equivalente a equação diferencial que descreve o problema.
- 2) Usar números aleatórios juntamente com as probabilidades transitórias para direcionar vários caminhos aleatórios na região de solução e registrar o potencial final de cada percurso aleatório.
- 3) Encontrar a média estatística dos potenciais registrados no passo 2.

Para o passo 1, a modelagem matemática da equação de Laplace deve-se desenvolver a equação de forma semelhante ao método de diferenças finitas. Entretanto a grande diferença reside nos termos probabilísticos inerentes aos métodos de Monte Carlo. Tomando a equação de Laplace para material homogêneo:

$$\nabla^2 u = 0 \Rightarrow \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \quad (1)$$

Substituindo as derivadas de diferenças finitas considerando h_x e h_y iguais:

$$\frac{u_1 + u_3 + u_2 + u_4}{4} = u_P \quad (2)$$

Substituindo pelas contribuições probabilísticas de cada contribuição:

$$p_1 u_1 + p_2 u_2 + p_3 u_3 + p_4 u_4 = u_P \quad (3)$$

Onde $p_1 = p_2 = p_3 = p_4 = 1/4$. No método FRW a partícula de avaliação de u_P move-se nas quatro direções com probabilidade $1/4$. Para definir o caminho da partícula no domínio [1] faz uso de um conjunto de valores aleatórios e define quatro subconjuntos com tamanhos idênticos onde cada subconjunto de valores é responsável por deslocar a partícula em uma direção.

No momento que a partícula alcança uma aresta com condição de contorno, a probabilidade é computada juntamente com o valor da condição de contorno para contribuir no valor de u_P . Para avaliar corretamente o valor de u_P , várias partículas devem ser lançadas do ponto e suas contribuições adicionadas em u_P .

Pode-se reescrever a equação anterior como:

$$u_P = \frac{1}{N} \sum_{i=1}^N C_i \quad (4)$$

onde N é o numero total partículas e C_i é o valor da condição de contorno alcançada pela partícula.

Uma outra forma de desenvolver o raciocínio é pensar na seguinte equação:

$$p_1 C_1 + p_2 C_2 + p_3 C_3 + p_4 C_4 + \dots + p_i C_i = u_P \quad (5)$$

onde p_i é a probabilidade da partícula atingir a condição de contorno e C_i é o valor da condição de contorno alcançada pela partícula.

Esta equação pode ser reescrita na forma de um somatório:

$$u_p = \frac{1}{N} \sum_{i=1}^m N_i C_i, (6)$$

onde N é o numero total de partículas, m é o numero de faces com condição de contorno e N_i é o numero de passos para alcançar a face i . Note que N_i/N é a probabilidade de a partícula alcançar a borda com condição de contorno. Esta equação é idêntica a equação (4).

Modelagem computacional:

A modelagem em JAVA requer o uso de classes, entretanto o diagrama da Figura 1 mostra claramente uma implementação puramente procedural já que nenhuma classe apresentada se configura como um objeto. Este tipo de implementação se justifica pela rápida codificação do método e por nenhuma exigência do exercício sobre uma modelagem orientada a objetos.

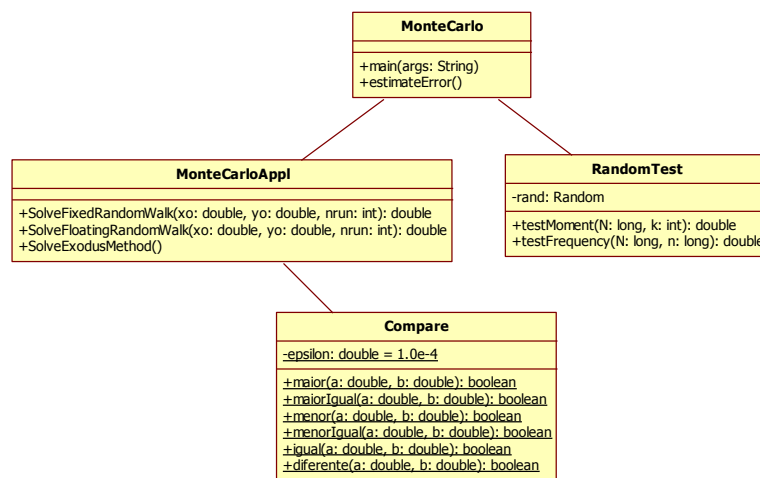


Figura 1 - Diagrama de classes.

A classe “MonteCarlo” é a classe principal, nela é gerada a malha de pontos onde a solução será calculada. Os pontos gerados são uniformemente espaçados, entretanto o método não exige essa distribuição já que o calculo do da aproximação em um ponto independe do calculo dos demais pontos.

Na classe “MonteCarloAppl” está o método que encontra o potencial para cada pondo da malha. Uma classe para escrita dos arquivos de resultados foi desenvolvida, entretanto pela semelhança com as classes já mostradas em exercícios anteriores a mesma foi suprimida do diagrama.

A classe “Compare” é uma classe utilitária usada para comparação de variáveis do tipo *double*. E por fim, a classe “RandomTest” possui métodos para testar os números aleatórios gerados pela biblioteca JAVA.

Resultados:

Para a geração de números aleatórios a biblioteca JAVA fornece a classe “Random” que gera números aleatórios usando como semente a hora da maquina. Primeiramente foi levantado medidas da uniformidade dos números gerados, para mensurar a uniformidade [1] comenta que uma das formas é executar o teste de “momento”. Na tabela 1 encontra-se o erro percentual do momento para $k = 1$ e diferentes valores de N calculado pela expressão (7).

Como esperado pela definição do Momento, o erro diminui à medida que a quantidade de números aleatórios usados aumenta.

Tabela 1 - Erro obtido para a medida de Uniformidade (Momento)

$$\frac{1}{N} \sum_{i=1}^N U_1^k = \frac{1}{k+1} \quad (7)$$

Números Gerados	Erro (%)
125	5,114
625	6,293
3.125	1,145
15.625	0,096

A distribuição das condições de contorno na placa quadrada pode ser visto na figura 2. Executado simulação para diferentes valores de N e cada simulação foi executada 10 vezes para se tomar a média das execuções como resultado final. Este passo foi usado, pois foram percebidas diferenças significativas dos valores de cada ponto entre simulações consecutivas. Na figura 3 tem-se a distribuição da temperatura para $N = 15625$.

Note que ao comparar a solução por FRW (Figura 3) com a solução por diferenças finitas da figura 4 qualitativamente há uma boa representação da distribuição de calor na placa.

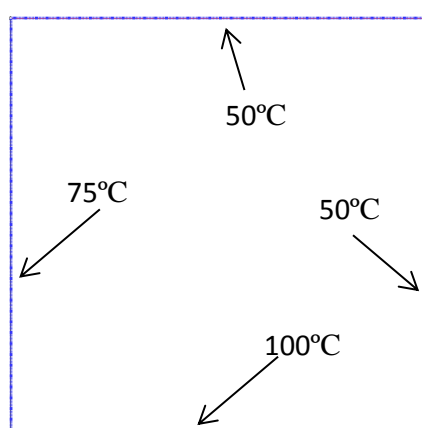


Figura 2 - Detalhes do Teste.

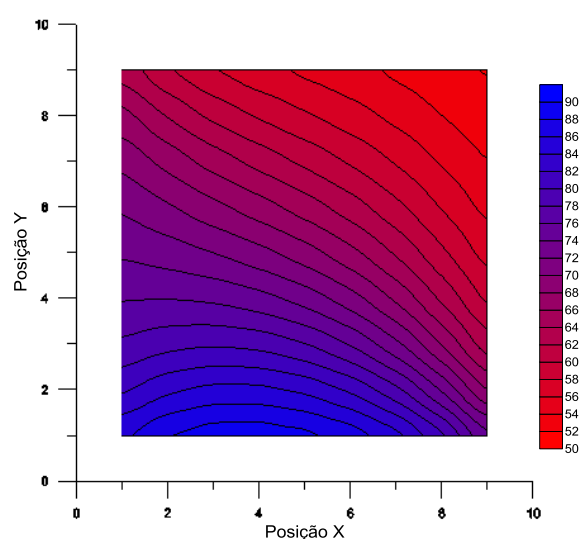


Figura 3 - Resultado Obtido com "FRW".

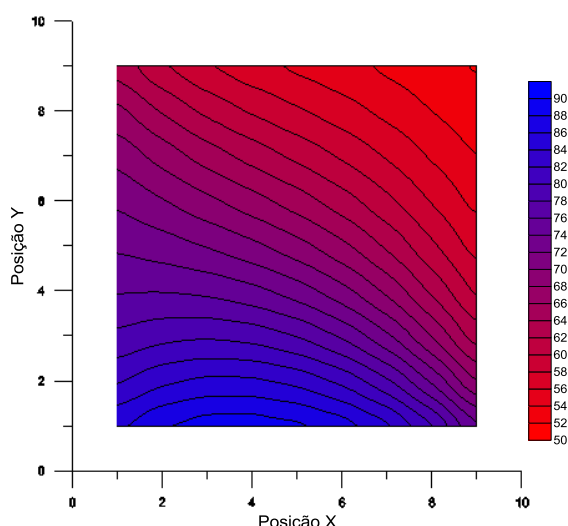


Figura 4 - Resultado Obtido com Diferenças Finitas.

Tabela 2 - Comparação entre o erro em relação à referência e o erro estimado.

Numero de passos	Erro máximo (%)	Erro máximo estimado (%)
125	1,55	2,052
625	1,05	0,951
3.125	0,53	0,364
15.625	0,48	0,207

De forma a avaliar quantitativamente os resultados dois testes foram executados. Primeiro foi calculada a estimativa de erro pela expressão (8) com intervalo de confiança de 95%, e em segunda instância calculado o erro ponto a ponto em relação à solução por diferenças finitas.

$$\varepsilon = \frac{St_{a/2;N-1}}{\sqrt{N}} \quad (8)$$

onde S é a dispersão e $t_{a/2;N-1}$ é o valor na tabela de distribuição t de Student para a confiança usada.

Na tabela 2 encontra-se o resultado para os diferentes valores de N usados, o erro máximo percentual máximo obtido em relação à solução por Diferenças Finitas e o máximo valor da estimativa de erro.

Note que com poucas partículas, o erro estimado é praticamente o dobro do erro obtido, já para um número maior de partículas o erro estimado é praticamente a metade do erro obtido. Pode-se atribuir essa diferença ao intervalo de confiança usado, acredito que este valor deva ser diferente para cada número de partículas usadas, já que para poucas partículas deve-se ter uma confiança menor e para muitas partículas uma confiança maior.

Conclusões:

O método de “Fixed Random Walk” é um método bastante interessante do ponto de vista da curiosidade. Apesar do erro em relação à solução por diferenças finitas ser pequeno, o método FRW é bastante lento se comparado a outras abordagens numéricas já estudadas na disciplina. Outro fator que considero negativo é a instabilidade da solução e ainda ter de executar a simulação mais de uma vez para retirar a média dos valores multiplica o tempo de execução que já não é pequeno dependendo do número de partículas usadas.

Claro que deve haver aplicações que necessitem deste tipo de método e que não possuam solução numérica simples. Para finalizar o trabalho gostaria de citar uma frase de Albert Einstein “Deus não joga dados...” por qual motivo os físicos o fazem?

2 – Resolver equação de Laplace para o problema de placa quadrada com condições de contorno $C_1 = 100$, $C_2 = 50$, $C_3 = 50$, $C_4 = 75$, através do método “Floating Random Walk”, baseando-se nos capítulos 2 e 5 da referência [1].

No método “Fixed Random Walk” o tamanho dos passos é fixo, já no método “Floating Random Walk” o tamanho dos passos varia, e mais, o ângulo que o passo toma em relação ao ponto inicial também varia entre um passo e outro.

O Método “Floating Random Walk” parte do princípio de que a variável de estado em um dado ponto é obtida, para duas dimensões, pela integral fechada da variável de estado no limite de um círculo com raio ρ definido a partir do ponto de interesse.

Para a aplicação deste método toma-se a coordenada do ponto de interesse (x_j, y_j) e calcula-se a coordenada da partícula no final de um passo pelas seguintes expressões:

$$x_{j+1} = x_j + \rho_j \cos(\varphi) \quad (9),$$

$$y_{j+1} = y_j + \rho_j \sin(\varphi) \quad (10),$$

onde é o ρ_j raio definido como a menor distância a uma dada condição de contorno e φ é um ângulo aleatório.

Esse cálculo é repetido até que a partícula atinja a condição de contorno, várias partículas são lançadas e o valor da variável de interesse no ponto é calculado pela expressão (4) mostrada no exercício anterior.

A modelagem computacional é a mesma do apresentado no exercício anterior, apenas um método foi incluído na classe “MonteCarloAppl”.

Resultados:

Os passos para testar o método “Floating Random Walk” são exatamente os mesmos do exercício anterior. Foram escolhidos diferentes números de partículas, e para cada número

diferente de partícula, foi tomado como resultado final a média entre dez execuções consecutivas.

Tendo em mão a média de cada execução para cada ponto, pode-se calcular a variância da variável de interesse e se estimar o erro. O resultado mostrado na Tabela 3 para cada valor diferente de partícula é do máximo valor do erro entre todos os pontos, e na terceira coluna o valor do máximo erro estimado com intervalo de confiança de 95%. Uma checagem pontual mostrou que o maior erro estimado não coincide com o ponto com maior erro calculado com relação à solução por diferenças finitas.

Tabela 3 - Comparação entre o erro em relação à referência e o erro estimado.

Numero de passos	Erro máximo (%)	Erro máximo estimado (%)
125	1,89	0,676
625	0,84	0,287
3.125	0,50	0,140
15.625	0,16	0,060

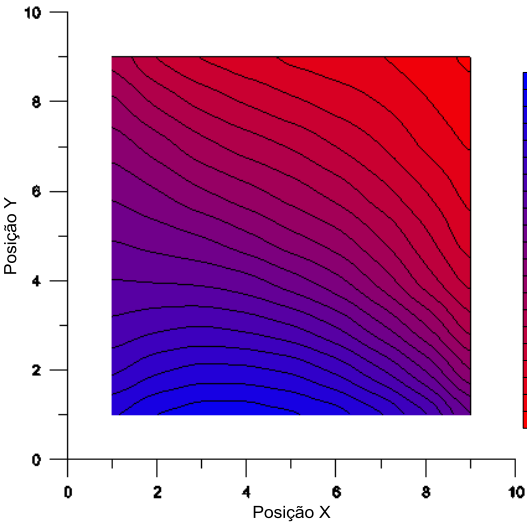


Figura 5 - Resultado Obtido com 125 partículas

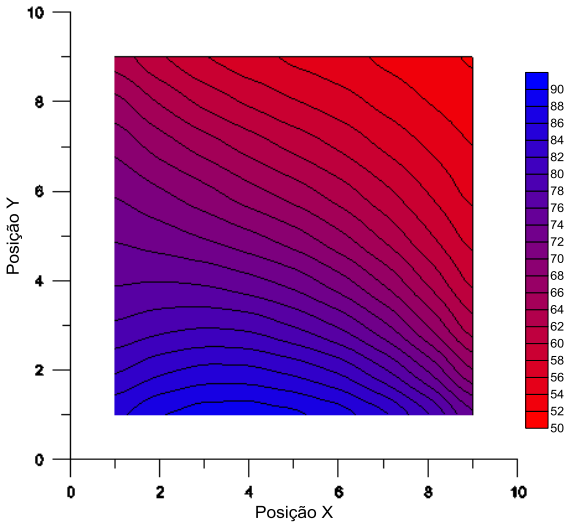


Figura 6 - Resultado Obtido com 625 partículas

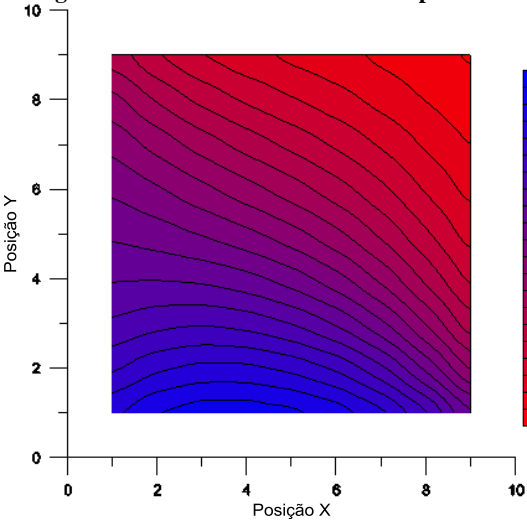


Figura 7 - Resultado Obtido com 3.125 partículas

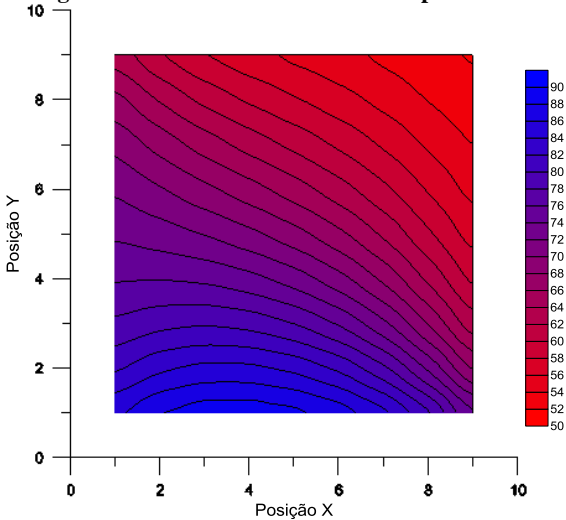


Figura 8 - Resultado Obtido com 15.625 partículas

A distribuição da temperatura calculada pelo método “Floating Random Walk” para 125, 625, 3.125, 15.625 partículas encontram-se respectivamente nas figuras 5 a 8. Note que só

é possível perceber alguma diferença no resultado da figura 5. Isso se dá pelo pequeno erro obtido.

Conclusões:

O método “Floating Random Walk” retornou um erro menor que o “Fixed Random Walk”, entretanto pode-se notar que o erro estimado está muito longe do erro obtido. Acredito que um atrativo deste método é que o usuário só precisa escolher o número de partículas a ser usado, já no “Fixed Random Walk” o usuário deve definir o tamanho do passo.

Tabela 4 - Avaliação do tempo computacional para os dois métodos usados nesta lista de exercícios.

Numero de passos	Tempo de Execução - Fixed Random Walk (ms)	Tempo de Execução - Floating Random Walk (ms)
125	112.510	551
625	611.858	2.558
3.125	3.876.362	17.316
15.625	NA	87.406

Um comparativo do tempo computacional entre os dois métodos foi efetuado (Tabela 4), porém no método “Fixed Random Walk” o tamanho do passo influencia e muito nesta medida, mas não tem grande impacto na solução.

Apesar do tempo de execução e do erro obtido para o método “Floating Random Walk” ser muito inferior se comparado ao “Fixed Random Walk”, não se pode afirmar qual realmente é o melhor. Como ambos os métodos são probabilísticos, mesmo tomando uma média de dez execuções as soluções e os erros estimados variam muito entre duas rodadas.

Gostei de realizar esta segunda etapa do exercício, pois pude perceber que os métodos de Monte Carlo permitem uma infinidade de brincadeiras e experiências na cozinha numérica, dependendo apenas do gosto do cliente.

Referências:

[1] SADIKU, M. N. O. Monte Carlo Methods for Electromagnetics. United States of America: CRC Press. 2009.

[2] http://pt.wikipedia.org/wiki/Distribui%C3%A7%C3%A3o_t_de_Student, acessado em 29 de Maio de 2012.