T3\_CN 4ª questão

Roteiro de Pedro

1. Olá. Eu sou Pedro Mendonça e vou apresentar a 4ª questão do trabalho 3 de Cálculo Numérico.
2. A quarta questão pede para implementar algoritmos com os métodos de **Jacobi e Gauss-Seidel** para resolver os sistemas lineares **A e B**. Na **letra A,** é dado o vetor de estimativa inicial nulo, 10-3 para a tolerância, e 300 para o número máximo de iterações. Na **letra B,** o vetor de estimativa é alterado para (1,0,1) e pergunta-se o que muda nos resultados. Na **letra C**, pede-se para resolver os sistemas com funções nativas do MATLAB, justificar a escolha e comparar os resultados.
3. O método de Jacobi é um método iterativo para solucionar sistemas lineares. **Primeiramente**, escrevemos cada uma das variáveis do sistema de forma explícita, como mostrado na figura. **Em seguida**, calculamos o valor de cada variável isolada na iteração atual substituindo as demais variáveis com seus valores da iteração anterior. É definido um **vetor** de estimativa inicial, que geralmente é o vetor nulo. Como **critério de parada**, é calculado, a cada iteração, o erro relativo entre o vetor de estimativa atual e o anterior. Caso seja menor que a tolerância pré-estabelecida, o processo para, caso contrário, continua até atingir o número máximo de iterações.
4. O método de Gauss-Seidel é quase igual ao de Jacobi, **o que muda é** que, a cada iteração, as variáveis que são calculadas já são utilizadas no cálculo das demais variáveis daquela mesma iteração. Aqui temos a fórmula geral para o **primeiro elemento** do vetor, os **elementos do meio** do vetor e o **último elemento**. Da mesma forma que no método de Jacobi, também temos um **vetor de estimativa inicial** e o **critério de parada** é o mesmo.
5. Como foi pedido, usaremos a norma infinita para calcular o erro relativo, cujo cálculo está expresso na figura.
6. A convergência dos dois métodos citados é dada pela seguinte **fórmula**, aplicada à matriz dos coeficientes. **Caso** a condição seja satisfeita, a matriz é **diagonalmente dominante** e o método converge. Caso **contrário**, ainda **assim** o método pode convergir. (2’10”)
7. Implementamos o método de Jacobi como uma função, onde **A** é a matriz dos coeficientes, **b** é o vetor das constantes ou vetor resposta, **x0** é o vetor de estimativa inicial, **tol** é a tolerância, **imax** é o número máximo de iterações e **x** é o vetor solução.
8. Na validação dos parâmetros, garantimos que todos os parâmetros são **numéricos**; a **matriz A** é quadrada; o **determinante de A** é não-nulo; **na linha 25**, dim recebe a ordem de A; logo após verificamos se **b e x0** são vetores linha ou coluna e **se são** compatíveis com a ordem de A; **em seguida**, se b for vetor linha, é transformado em vetor coluna.
9. **Aqui** fazemos o mesmo com o vetor x0; e **neste** trecho do código verificamos se a matriz A é diagonalmente dominante, **conforme** a fórmula apresentada.
10. Na etapa de processamento, **temos um for** em k para cada iteração dos cálculos da estimativa atual. Na **linha 69**, soma recebe o valor de b. Logo em seguida, temos um **for em i** e **outro em j**, que irão percorrer toda a matriz A para realizar as **operações** do método de Jacobi. A **linha 74** realiza **esta** parte da equação e a **linha 77** realiza **esta outra**. Na **linha 79** calculamos o erro relativo conforme a **fórmula** apresentada considerando a **norma infinita**. Caso o erro seja **menor** que a tolerância, o laço é finalizado. Na **linha 15** atualizamos o vetor de estimativa anterior. Após o for em k, caso **k seja igual a imax**, informamos ao usuário que o **número de iterações** máximo foi atingido.
11. O método de Gauss-Seidel é praticamente idêntico. O que muda é que temos **um vetor** auxiliar que recebe o valor de x0 a cada iteração e, após cada operação em x, **atualizamos** o valor de x0, que já será utilizado no **próximo** cálculo de x. **Em seguida**, o erro relativo é calculado entre o vetor x e o vetor auxiliar. (4’07”)
12. Aqui temos um script para o item 4.a com a **declaração** dos sistemas, **declaração** dos parâmetros e chamada de funções para solucionar os sistemas **A** e **B**.
13. Para o item 4.b, mudamos apenas o **vetor** de estimativa inicial.
14. Para o item 4.c, escolhemos o linsolve como método direto, pois implementa a decomposição LU, porque foi estudada nesse trabalho e conhecemos bem o método; e o bicg como método iterativo, que implementa o método do Gradiente Biconjugado, pois apesar de não ter sido estudado, tem parâmetros semelhantes aos de Jacobi e Gauss-Seidel. **Aqui** resolvemos os sistemas com linsolve e **aqui** com bicg.