4º LABORATÓRIO de CCI-22 / 2022

CTA - ITA - IEC

Prof Juliana

Objetivo: Implementação de métodos numéricos <u>iterativos</u> para resolução de sistemas lineares.

Entregar (através do Google Classroom):

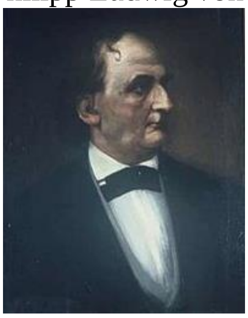
a) Códigos utilizados (não precisa compactar).

b) Relatório (em pdf) para responder as questões abaixo, mostrando resultados e conclusões. Não colocar os códigos no relatório.

c) Desconto de 1 ponto na nota por dia de atraso.

Carl Gustav Jacob **Jacobi**

D Philipp Ludwig von **Seidel**



<u>Importante</u>:

- Considere um sistema linear **`Ax = b`** de ordem n>0, com matriz de coeficientes **`A`** e termos independentes **`b`**. Considere que **`A`** $\in R^{n \times n}$ e **`b`** $\in R^{n \times 1}$.
- Implementar as funções solicitadas em arquivos separados.
- Obedeça a assinatura das funções.
- Nessas implementações não será permitido o uso de funções ou operadores do *Matlab* (ou outra ferramenta utilizada) que realizam o trabalho pedido (ou boa parte dele). Por outro lado, uma vez terminadas as implementações, você pode conferi-las com os resultados fornecidos pelas funções existentes. Nessas comparações, tenha em conta que os resultados não precisam ser rigorosamente idênticos: pode haver uma pequena diferença devido a erros de arredondamento.

<u>IMPLEMENTAÇÃO</u>: Implemente as funções abaixo.

`satisfaz = CriterioLinhas(A)`: verifica se a matriz A satisfaz o Critério das Linhas (condição suficiente para convergência do Método de Gauss-Jacobi) e retorna o resultado como a booleana **`satisfaz`**. Se **`satisfaz`** for **`true`**, o critério é satisfeito.

`[x, dr] = GaussJacobi(A, b, x0, epsilon, maxIteracoes)`: resolve o sistema **`Ax = b`** através do Método de Gauss-Jacobi, usando **`x0`** como chute inicial, **`epsilon`** como tolerância para critério de parada por erro relativo e **`maxIteracoes`** como limite máximo de iterações. O vetor coluna solução **`x`** é retornado. Também é retornado o vetor coluna **`dr`**, que contém os erros relativos de todas as iterações, de modo que **`dr(k)`** é o erro relativo calculado na iteração **`k`**. Obs: O número de iterações realizadas será justamente o tamanho do vetor **`dr`**.

`[satisfaz, beta] = CriterioSassenfeld(A)`: verifica se a matriz **`A`** satisfaz o Critério de Sassenfeld (condição suficiente para convergência do Método de Gauss-Seidel) e retorna o resultado como a booleana **`satisfaz`**. Se **`satisfaz`** for **`true`**, o critério é satisfeito. Além disso, o **`beta`** (maior dentre todos β_i, $1 \le i \le n$) do Critério de Sassenfeld é retornado.

`[x, dr] = GaussSeidel(A, b, x0, epsilon, maxIteracoes)`: resolve o sistema **`Ax = b`** através do Método de Gauss-Seidel, usando **`x0`** como chute inicial, **`epsilon`** como tolerância para critério de parada por erro relativo e **`maxIteracoes`** como limite máximo de iterações. O vetor coluna solução **`x`** é retornado. Também é retornado o vetor coluna **`dr`**, que contém os erros relativos de todas as iterações, de modo que **`dr(k)`** é o erro relativo calculado na iteração **`k`**. Obs: O número de iterações realizadas será justamente o tamanho do vetor **`dr`**.

Considere a seguinte definição de erro relativo na iteração **`k`**:

$$d_r^{(k)} = \frac{\max_{1\le i\le n} \left| x_i^{(k)} - x_i^{(k-1)} \right|}{\max_{1\le i\le n} \left| x_i^{(k)} \right|}$$

TAREFA 1: ANÁLISE SOBRE CONVERGÊNCIA

- Faça uma única tabela (como a abaixo) para indicar seus resultados. Obs:
  - Os valores apresentados são apenas exemplos.
  - Quando divergir, indicar o último **x** calculado.
  - Quando convergir, indicar a quantidade de iterações realizadas.
  - A tabela apresenta também o resultado da função Matlab A\b para você pode comparar. Obs: Aqui não precisa indicar a quantidade de iterações.
  - Pode ser útil plotar o vetor de erros relativos **dr** para ajudar a embasar suas conclusões. Só precisa fazer isso para os casos de interesse.
  - Não precisa comentar sobre cada sistema. Basta responder as perguntas colocadas a seguir.

- Utilize: **x0** como vetor nulo, **epsilon** = 0.001 e **maxIteracoes** = 100.

Sistema	Matlab A\b	Critério das Linhas	Critério Sassenfeld	Gauss-Jacobi	Gauss-Seidel
1	$x = [1\ 1\ 1.3]^T$	Não satisfaz	Satisfaz beta = 0.9	Diverge $x^{(100)} = 1.0e+36 * [\ -1.0564\ -1.4688\ \ 2.0959]^T$	Converge $x^{(23)} = [1\ 1\ 1.3]^T$
...					
7					

Obs: T significa transposta.

Sistema 1:
```
A = [1 3 1; 5 2 2; 0 6 8]
b = [-2; 3; -6]
```

Sistema 2:
```
A = [5 2 2; 1 3 1; 0 6 8]     %Permutação de linhas do sistema 1
b = [3; -2; -6]
```

Sistema 3:
```
A = [1 3 4; 1 -3 1; 1 1 5]
b = [8; -9; 1]
```

Sistema 4:
```
A = [1 2 -2; 1 1 1; 2 2 1]
b = [3; 0; 1]
```

Sistema 5:
```
A = [2 1 1; 1 2 1; 1 1 2]
b = [4; 4; 4]
```

Sistema 6:
```
A = [5 -1 1; 2 4 -1; -1 1 3]
b = [10; 11; 3]
```

Sistema 7:
```
A = [5 1 1; 3 4 1; 3 3 6]
b = [5; 6; 0]
```

- **Considere as afirmações abaixo. Elas são verdadeiras! Você só precisa explicá-las usando exemplos da tabela.**

  a) O Critério das Linhas é uma condição suficiente, mas não necessária, para convergência do Método Gauss-Jacobi.

  b) O Critério de Sassenfeld é uma condição suficiente, mas não necessária, para convergência do Método Gauss-Seidel.

  c) Em geral, o Método Gauss-Seidel converge com uma menor quantidade de iterações que o Método Gauss-Jacobi.

  d) Se um sistema satisfaz o Critério das Linhas, então também satisfará o Critério de Sassenfeld. A volta não é verdadeira.

  e) Às vezes basta fazer uma permutação no sistema (trocar linhas e/ou colunas) para satisfazer o(s) critérios(s) de convergência.

<u>TAREFA 2: ANÁLISE SOBRE TEMPO DE EXECUÇÃO</u>

- Investigue os tempos gastos (usando `tic` e `toc`) para resolver sistemas lineares (com diferentes tipos de matrizes) usando diferentes métodos.
  - Para os métodos iterativos, você pode usar: `x0` como vetor nulo, `epsilon` = 10*eps e `maxIteracoes` = 1000. Assim daremos mais tempo para eles trabalharem!
  - Para computar o tempo de execução, pegue a média de algumas execuções (por exemplo, umas cinco). Mas descarte as medidas iniciais (pois vimos no Lab1 que elas são inesperadas!). Ex: rode 10 vezes, mas pegue a média das últimas 5 vezes. A cada rodada, use o mesmo sistema (a seguir teremos mais detalhes).
  - Para cada caso abaixo, há um sistema de exemplo. Você pode usá-lo. Se precisar alterar algo, sem problemas! Se quiser cancerizar, pode estudar outros sistemas maiores!

- Para cada caso abaixo, preencha a tabela abaixo e depois faça a sua análise comparativa.
  - Nas colunas ‘Resíduo’ e ‘Qtde iterações’, indique o valor da última rodada. Ex: das 10 vezes que você rodou, pegue os valores da última vez.

Método	**Resíduo***	**Tempo (ms)**	**Qtde iterações****
Gauss com Pivoteamento (Lab3-questão1)	-		-
Solução com Decomposição LU (Lab3-questão2)	-		-
Gauss-Jacobi			
Gauss-Seidel			

* Resíduo é o valor do último dr. No nosso caso, só faz sentido para os métodos iterativos.

** Qtde iterações, na verdade, é o tamanho do vetor dr. No nosso caso, só faz sentido para os métodos iterativos.

a) Matriz bem condicionada grande

```
A = rand(100)+ 100*eye(100);
b = rand(100,1);

%Matrizes próximas da identidade são, em geral, bem condicionadas.
%Rode o comando cond(A,Inf) para se certificar que a matriz A é bem condicionada
%No nosso exemplo, multiplicar a diagonal principal por um fator grande (no caso, 100), pode ajudar a satisfazer o critério de convergência dos métodos iterativos.
```

> **Curiosidade**: Para matrizes bem condicionadas, a literatura indica que os métodos iterativos são melhores!

De acordo com os seus dados, verifique que a afirmação acima é verdadeira. Comente também sobre coisas estranhas que você encontrar. Ex: Um método com menos iterações que o outro, mas com tempo maior. Explique porque isso ocorre.

b) Matriz mal condicionada

```
A = hilb(8);
b = rand(8,1);

%Hilbert é um caso clássico de matriz mal condicionada!
%Rode o comando cond(A,Inf) para se certificar que a matriz A é mal condicionada
```

> **Curiosidade**: Para matrizes mal condicionadas, a literatura indica que os métodos diretos (como Eliminação de Gauss) são melhores.

De acordo com os seus dados, verifique que a afirmação acima é verdadeira. Explique porque os iterativos são ruins nesse caso.

c) Matriz esparsa grande

```
A = sprandn(1000,1000,.01)+100*eye(1000);
b = rand(1000,1);

% A função sprandn ajuda a gerar matrizes esparsas (com muitos zeros).
%Rode o comando cond(A,Inf) para se certificar que a matriz A é BEM condicionada
%No nosso exemplo, multiplicar a diagonal principal pelo fator 100 foi para ajudar a satisfazer o critério de convergência dos métodos iterativos. Caso precise ajustar esse valor para garantir convergência, fique à vontade!
```

> **Curiosidade**: Para matrizes esparsas com dimensão grande, a literatura indica que os métodos iterativos são melhores, pois utilizam a matriz inicial (sem alterar sua estrutura), o que reduz erros de arredondamento e tempo de processamento.

De acordo com os seus dados, verifique que a afirmação acima é verdadeira. Explique porque os diretos são ruins nesse caso.

Avaliação da dupla

Aluno(a)	Atividade realizadas	Percentual merecido da nota
Fulano	*Fez Q1 e revisou Q2*	*30%*
Beltrano	*Revisou Q1, fez Q2 e Q3*	*100%*

Atenção: Dividir bem as tarefas da dupla. Não vale ter uma pessoa que só escreve relatório, ou que só testa/revisa. Todos devem ‘pôr a mão na massa’!

Bom trabalho!