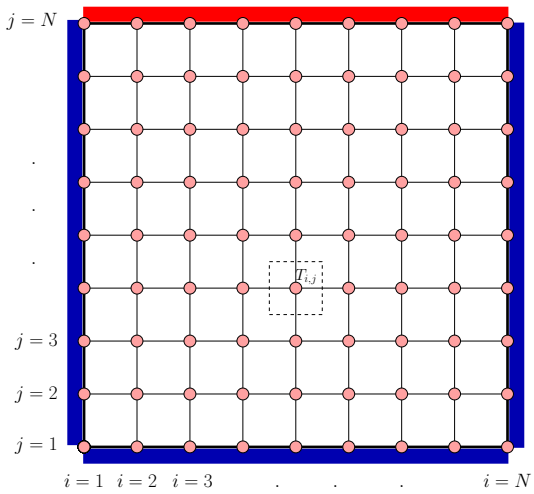
Métodos Numéricos e Computacionais I
SME0305 - Prova 9 - 06/05/2016

Nome:

Num. USP:

Considerar o problema de condução de calor num sólido de forma quadrada. O mesmo será resolvido com o método de discretização centrado nos vértices com N pontos em cada direção. Lembrando, a equação para o ponto i, j com $i, j = 2, \ldots, N-1$

$$-T_{i+1,j} - T_{i-1,j} + 4\,T_{i,j} - T_{i,j+1} - T_{i,j-1} = 0$$



Ex. 1 (3 pontos) Completar a forma vetorizada (e por tanto mais eficiente) do método de Jacobi sem matriz (*matrix free*) para resolver o problema. Lembre que na forma vetorizada não precisam ser feitos de maneira explícita os laços sobre i e j.

```
N = 21;  % Numero de pontos
% Temperaturas nas paredes verticais
Told = zeros(N,N);
for j=1:N
  Told(1,j) = 0.0;
  Told(N,j) = 0.0;
end
% Temperaturas nas paredes horizontais
for i=1:N
  Told(i,1) = 0.0;
  Told(i,N) = 20.0;
end
Tnew = Told;
for k=1:100000 %Loop de iteracoes









   if(norm(Tnew-Told,inf,'rows') < 1e-8)
    break ;
   end
  Told = Tnew;
end
```

Ex. 2 (3 pontos) Para montar a matriz do sistema precisamos asignar uma numeração global às incógnitas. Para isto, pode ser usada a função `ij2n` na sequência:

```
function IGLO = ij2n (i, j, N)
  IGLO = i + (j-1)*N;
end
```

que dados os índices `i` e `j` devolve o índice global da incógnita $T_{i,j}$.

Fazer um código de Octave **QUE UTILIZE** essa função para inserir numa matriz `A`, os coeficientes da linha correspondente à equação do nó $(i, j) = (2, 3)$, para um grid que possui `N` pontos em cada direção.

Ex. 3 (4 pontos) Completar o código do método iterativo geral, em que `A` é a matriz do sistema, `b` o lado direito, `x0` a chute inicial, `MAXIT` o máximo de iterações permitido, `TOL` a tolerância e `method` uma variável que indica o método a ser usado.

```
function x = iterativo(A, b, x0, ...
                       MAXIT, TOL,method)
[n, m] = size(A);
if(n ~= m)
  error("System is not square\n")
end

flag_grad = 0;
if(strcmp(method,"Direct")) % Direto
  M =       ; %PREENCHER
elseif(strcmp(method,"Jacobi")) % Jacobi
  M =                       %PREENCHER
elseif (strcmp(method,"Gauss-Seidel")) % Gauss-Seidel
  M =                 ; %PREENCHER
elseif(strcmp(method,"Gradients")) % Gradientes
  M =           ; %PREENCHER
  flag_grad = 1;
elseif(strcmp(method,"luinc")) % LU incompleta
  [l, u] = luinc(A, 1.0e-8);
  M =             ; %PREENCHER
else
  error("Method not recognized\n");
end

xold = x0;
beta = 1.0;
for iter=1:MAXIT
  r = A*xold - b;
  d =            ; %PREENCHER
   if(flag_grad == 1)
     beta =                    ; %PREENCHER
   end
  xnew =                       ; %PREENCHER
  if(norm(xnew-xold,inf) < TOL) break ; end
  xold = xnew;
end
x = xnew;
```