Proyecto

Samuel Moreno Vahos, Juan Nicolás Sepúlveda

29 de noviembre de 2023

1. Explicación del código:

1.1. Inputs

```
%% Input

L = input("\nLongitud de la superficie de difusión: ");
t_max = input("\nTiempo total de la simulación: ");
n_x = input("\nNúmero de nodos en el espacio: ");
n_t = input("\nNúmero de nodos en el tiempo: ");
alpha = input("\nConstante del material: ");
T_1j = input("\nTemperatura del contorno: ");
T_i1 = input("\nTemperatura inicial: ");

d_x = L/(n_x - 1); % Se calcula delta de x
d_t = t_max/(n_t - 1); % Se calcula delta de t
```

Figura 1: Código para recoger inputs

1.2. Estado estable

Se halló la matriz del sistema de forma analítica y se define según sus coeficientes. También se crea el vector para el lado derecho de la igualdad y se hallan las temperaturas según este sistema.

```
%% Estado estable

fprintf("\n\n########## Estado estable ##########\n\n");

% Coeficientes para la matriz B
a = -(2 + d_x^2*alpha^2);

% Se crea la matriz B de la forma:
% a  1  0  0  0
% 1  a  1  0  0
% 0  1  a  1  0
% 0  0  1  a  1
% 0  0  0  2  a
B_1 = [zeros(1, n_x - 1); eye(n_x - 2, n_x - 1)];
B_2 = a*eye(n_x - 1);
B_3 = [zeros(n_x - 1, 1) eye(n_x - 1, n_x - 2)];
B = B_1 + B_2 + B_3;
B(n_x - 1, n_x - 2) = 2;

% Se crea el vector b de la forma:
% -T_(1,j)
%    0
%    0
%    0
%    0
sol_B = zeros(n_x - 1, 1);
sol_B(1) = -T_1j;

% Se halla la solución para graficar
T_B = resolver(B, sol_B, "escalado");
T_B = [T_1j; T_B];
disp(T_B');

figure(2)
p = plot(0:d_x:L, T_B);
p.Color = "#4DBEEE";
p.LineWidth = 2;
```

Figura 2: Código para calcular el vector solución del estado estable

1.3. Estado transitorio

Se halló la matriz del sistema de forma analítica y se define según sus coeficientes. Primero se crea el vector solución para el tiempo inicial.

```
%% Estado transitorio

fprintf("\n\n########## Estado transitorio ##########\n\n");

% Coeficientes para la matriz A y el vector b
a = d_t;
b = -(2*d_t + d_x^2*d_t*alpha^2 + d_x^2);
c = d_t;
d = -d_x^2;

% Se crea la matriz A de la forma:
% b  c  0  0  0
% a  b  c  0  0
% 0  a  b  c  0
% 0  0  a  b  c
% 0  0  0 a+c b
A_1 = [zeros(1, n_x - 1); a*eye(n_x - 2, n_x - 1)];
A_2 = b*eye(n_x - 1);
A_3 = [zeros(n_x - 1, 1) c*eye(n_x - 1, n_x - 2)];
A = A_1 + A_2 + A_3;
A(n_x - 1, n_x - 2) = A(n_x - 1, n_x - 2) + c;

% Se crea la solución inicial (t = 0) para graficar
temp = T_i1*ones(n_x - 1, 1); % Temperatura inicial
T_A = [T_1j; temp]; % Temperatura de contorno
fprintf("t = 0.00:\n");
disp(T_A');

figure(1)
p = plot(0:d_x:L, T_A);
p.LineWidth = 2;
hold on
```

Luego se factoriza la matriz para reutilizarla en las siguientes iteraciones. De nuevo, se creó el vector para el lado derecho de la igualdad y se hallan las temperaturas en cada nodo de tiempo. Estas se usan para redefinir el vector b.

```
[L_A, U_A] = factor_LU(A); % Se factoriza la matriz A

% Se crea el vector b de la forma:
% d*T_(2,j-1) - a*T_(1,j)
%       d*T_(3,j-1)
%       d*T_(4,j-1)
%       d*T_(5,j-1)
%       d*T_(6,j-1)
sol_A = d*T_i1*ones(n_x - 1, 1);
sol_A(1) = sol_A(1) - a*T_1j;

for i = 2:n_t
    % Se halla la nueva solución
    temp = solve_LU(L_A, U_A, sol_A);
    T_A = [T_1j; temp];
    fprintf("\nt = %.2f:\n", d_t*(i - 1));
    disp(T_A');

    p = plot(0:d_x:L, T_A);
    p.LineWidth = 2;

    % Se actualiza el vector b para usar en la siguiente iteración
    sol_A = d*T_A;
    sol_A(1) = sol_A(1) - a*T_1j;
end
hold off
```

Figura 3: Código para calcular los vectores solución para el estado transitorio

2. Valores ingresados:

Los valores que se ingresaron para probar el código fueron:

- Longitud de la superficie de difusión: 1 metro
- Tiempo total de la simulación: 2 segundos
- Número de nodos en el espacio: 11 nodos
- Número de nodos en el tiempo: 9 nodos
- Constante del material: 0.2
- Temperatura del contorno: 70°C
- Temperatura del ambiente o inicial: 20°C

```
>> proyecto

Longitud de la superficie de difusión:
1

Tiempo total de la simulación:
2

Número de nodos en el espacio:
11

Número de nodos en el tiempo:
9

Constante del material:
0.2

Temperatura del contorno:
70

Temperatura inicial:
20
```

Figura 4: Valores ingresados

3. Resultados y gráficas obtenidas

3.1. Resultados para el estado estable

Vector Solución:

```
70.0000   69.7377   69.5032   69.2966   69.1177   68.9664   68.8427   68.7466   68.6779   68.6367   68.6230
```

Figura 5: Estado estable (resultados)

Gráfica:

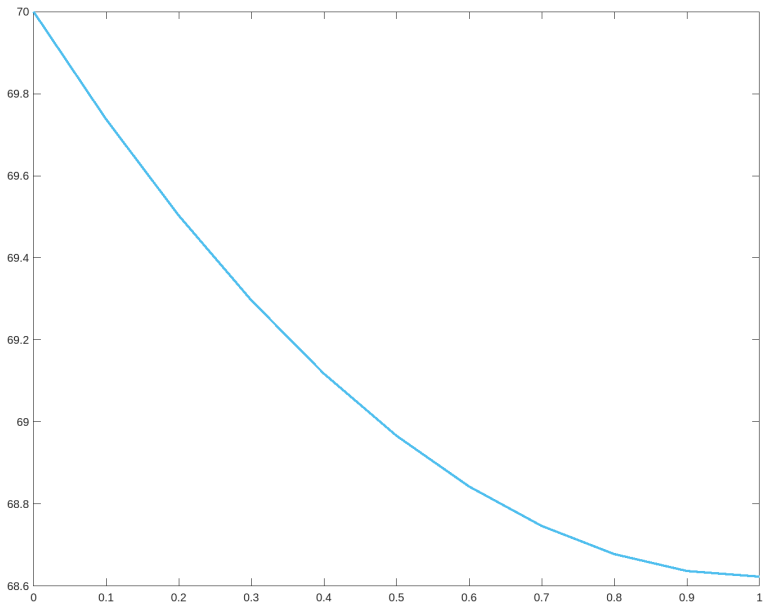

Figura 6: Estado estable (gráfica)

3.2. Resultados para el estado transitorio

Vectores Solución:

```
t = 0.00:
    70    20    20    20    20    20    20    20    20    20    20


t = 0.25:
   70.0000   61.2336   54.1410   48.4357   43.8872   40.3117   37.5649   35.5357   34.1421   33.3278   33.0600


t = 0.50:
   70.0000   66.6357   63.1636   59.7939   56.6742   53.9067   51.5616   49.6871   48.3173   47.4782   47.1915


t = 0.75:
   70.0000   68.1735   66.3013   64.4422   62.6600   61.0175   59.5731   58.3793   57.4815   56.9184   56.7222


t = 1.00:
   70.0000   68.9233   67.8311   66.7523   65.7183   64.7616   63.9148   63.2096   62.6750   62.3374   62.2190


t = 1.25:
   70.0000   69.3203   68.6410   67.9780   67.3480   66.7688   66.2584   65.8343   65.5133   65.3106   65.2395


t = 1.50:
   70.0000   69.5338   69.0768   68.6377   68.2260   67.8514   67.5241   67.2540   67.0507   66.9228   66.8781


t = 1.75:
   70.0000   69.6491   69.3121   68.9939   68.7000   68.4361   68.2079   68.0213   67.8818   67.7945   67.7641


t = 2.00:
   70.0000   69.7114   69.4392   69.1863   68.9561   68.7519   68.5774   68.4359   68.3309   68.2656   68.2430
```

Figura 7: Estado transitorio (resultados)

Gráfica:

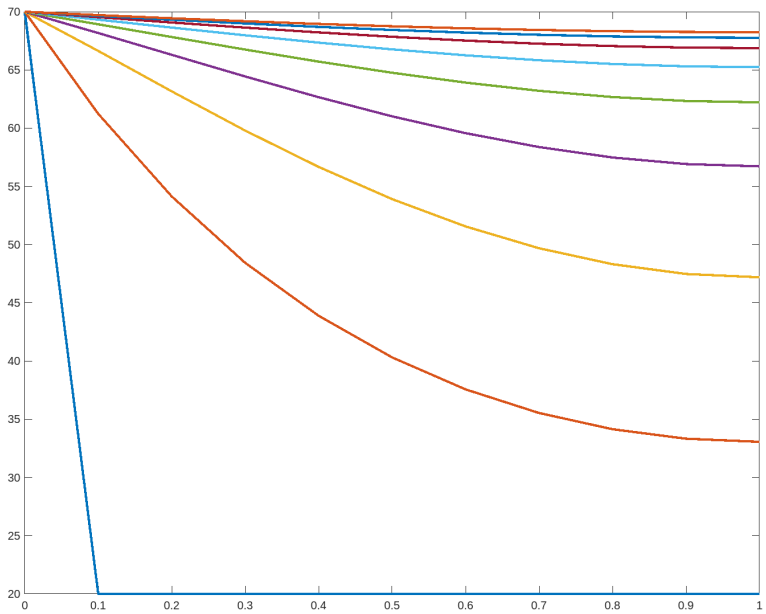

Figura 8: Estado transitorio (gráfica)