Proyecto nº 2

Ana Robledano, Ignacio López, Santiago Figueroa y Lucía de Angulo

17 de febrero de 2022

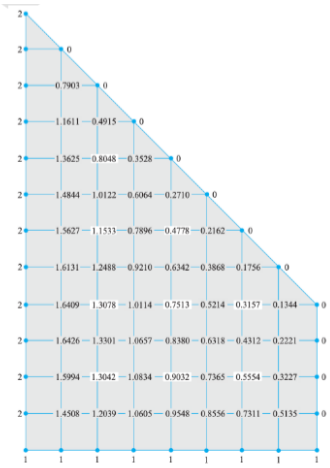


Figura 1: Placa del enunciado con temperaturas

0.1. Resumen

El método de Montecarlo fue ideado por los matemáticos Stanislaw Ulam y John Von Neumann en 1949. Se trata de un método no determinista empleado para aproximar expresiones matemáticas que son complicadas de calcular con exactitud. Se le puso ese nombre en referencia al Casino de Montecarlo en Mónaco, "la capital del juego del azar", pues la ruleta es un generador de números aleatorios. Una de las cualidades de este método es su universalidad, es decir, es aplicable a problemas muy diversos, ya sean estos planteados sobre la física o sobre las matemáticas.

0.2. Introducción

0.2.1. Descripción del problema

El objetivo de este proyecto es calcular la temperatura aproximada en cada uno de los 49 puntos indicados en la malla que se muestra al inicio de la página, sabiendo la temperatura de las paredes.dific Una posible solución al problema sería plantear un sistema de 49 ecuaciones y 49 incógnitas, pero tal cantidad de operaciones podrían dificultar los cálculos y llevar a errores, y además, no es un método óptimo. Por lo tanto, resolveremos este problema utilizando el método de Montecarlo.

0.2.2. Dificultad técnica

La dificultad técnica de este problema es media, pues los datos iniciales son sencillos. A nivel computacional, la dificultad es exponencial, ya que a mayor número de puntos y mayor aproximación del resultado exacto, más se repiten las operaciones, aumentan las iteraciones de los buclesy mayor es el tiempo de ejecución.

0.2.3. Enunciado

Se pide determinar la distribución de la temperatura de 49 puntos cuando se conocen las temperaturas en los bordes de una placa. El primer paso de este problema es representar en Matlab una figura que nos ayude a visualizar los datos del enunciado . De esta manera, viendo la figura y sus puntos en la gráfica, el problema resulta más visual y sencillo de resolver.

0.3. Resolución

A continuación, mostramos el código que hemos creado en Matlab para resolver el problema planteado. Lo que hemos hecho ha sido representar la malla del enunciado con las temperaturas de las paredes. A partir de ello, hemos creado la matriz M que contiene por columnas todos los puntos (49) que deseamos calcular. Para calcular la temperatura en cada punto, hemos creado un bucle que se repita 10,000,000 veces para cada punto, es decir, que desde dicho punto se hagan 10,000,000 paseos aleatorios hasta que lleguen a una pared. El número aleatorio será entre 0 y 1, y dependiendo de su magnitud, el punto se desplazará hacia arriba, hacia abajo, hacia la derecha o hacia la izquierda. Como la probabilidad de que toque una pared más cercana es mayor, la temperatura final será bastante precisa.

Finalmente, hemos representado la placa con las temperaturas ya calculadas y otra tabla que representa en qué partes la temperatura alcanza a ser menor que 1ºC y dónde alcanza a ser mayor que 1ºC. Hemos pensado que representando las gráficas, todo es más visual y mucho más sencillo de comprender para alguien ajeno al proyecto.

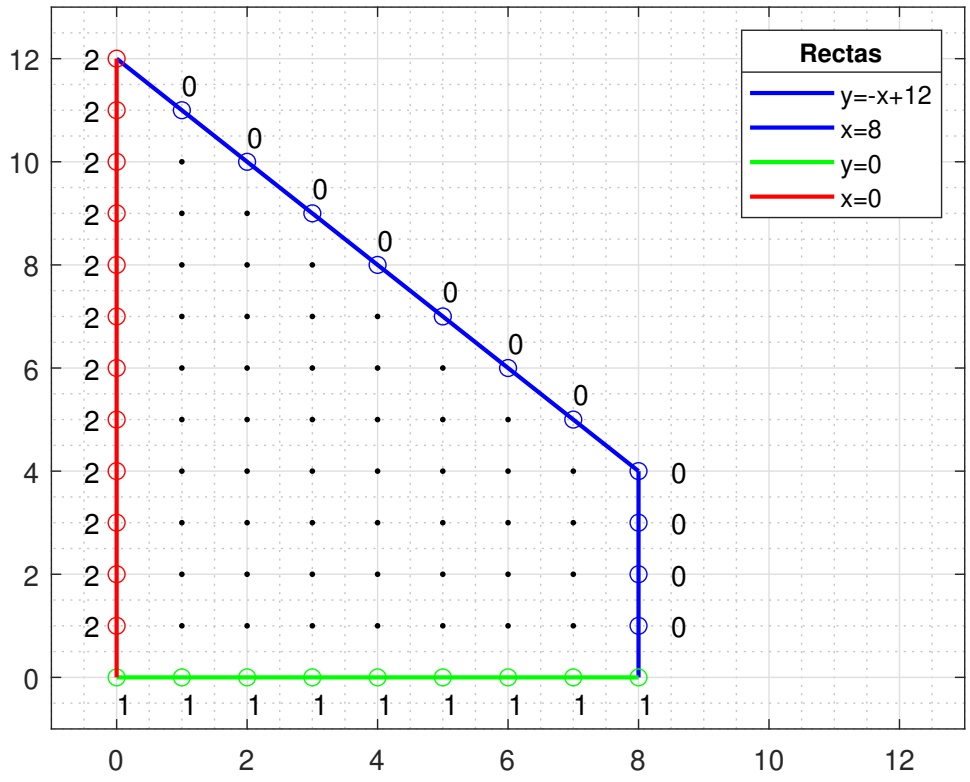
Representación del enunciado:

```
clear, clc;
%Representación de la malla
%Recta de temperatura 0
x0=0:1:8;
y0=-x0+12;
p1=plot(x0,y0,'LineWidth',1.5,'Color','b');
hold on
for a=1:7
    b=-a+12.5;
    y=-a+12;
    text(a,b,'0')
    plot(a,y,'o', 'color', 'b')
end


axis([-1 13 -1 13])
title('Figura 1: Placa del enunciado')
hold on
```

```
grid on
grid minor

%Recta de temperatura 0
y1=0:1:4;
x1=8+0*y1  ;
p2=plot(x1,y1,'LineWidth',1.5,'Color','b');
hold on
for b=1:4
    x=8;
    a=8.5;
    text(a,b,'0')
    plot(x,b,'o', 'color', 'b')
end

%Recta de temperatura 1
x3=0:1:8;
y3=0*x3;
p3=plot(x3,y3,'LineWidth',1.5,'Color','g');
hold on
for a=0:8
    y=0;
    b=-0.5;
    text(a,b,'1')
    plot(a,y,'o', 'color', 'g')
end

%Recta de temperatura 2
y2=0:1:12;
x2=0*y2;
p4=plot(x2,y2,'LineWidth',1.5,'Color','r');
hold on
for b=1:12
    x=0;
    a=-0.5;
    text(a,b,'2')
    plot(x,b,'o', 'color', 'r')
end
title(legend,'Rectas')

c=0;
%Matriz M que contiene los 49 puntos que queremos calcular
M=[];
for y=1:10
    for x=1:7
        if y<-x+12
```

```
            c=c+1;
            plot(x,y,'.', 'color', 'k')
            M(:,c)=[x y];
        end
    end
end
legend([p1 p2 p3 p4],'y=-x+12','x=8','y=0','x=0','Location','northeast')
```

Figura 1: Placa del enunciado



Cálculo de las temperaturas interiores (T)

```
p=[];
a=0;
b=0;
T=[];
%Calculamos la temperatura en cada uno de los 49 puntos
for x=1:49
        a=a+1;
        p=M(:,a);
        p1=p;
        n=0;
        y=1e7;
```

```
        for m=1:y
            %Hasta que no toque una pared, el punto se moverá en horizontal
            %y vertical aleatoriamente
            while p(1)~=0 && p(2)~=0 && p(1)~=8 && p(2)~=-p(1)+12
                %Un número z aleatorio entre 0 y 1 para cada movimiento
                z=rand(1);
                  %Punto se va hacia la derecha
                  if z<0.25
                     p(1)=p(1)+1;
                  end
                  %Punto se va hacia la izquierda
                  if z>=0.25 && z<0.5
                    p(1)=p(1)-1;
                  end
                  %Punto se va hacia abajo
                  if z>=0.5 && z<0.75
                    p(2)=p(2)-1;
                  end
                  %Punto se va hacia arriba
                  if z>=0.75
                    p(2)=p(2)+1;
                  end
            end
            %Evaluamos qué pared ha tocado y sumamos la temperatura
            %correspondiente
            if p(2)==0
                n=n+1;
            end
            if p(1)==0
                n=n+2;
            end
            p=p1;
       end
        temp=n/y;
        %Creamos una matriz T 1x49 que contenga todas las temperaturas
        %interiores de izquierda a derecha y de abajo a arriba
        b=b+1;
        T(b)=[temp];
end
T
```

```
T = 1x49
        1.447         1.23        1.063        0.958
        0.884        0.742        0.529        1.593
        1.302        1.073        0.858        0.729
        0.542        0.338         1.62        1.324
        1.08        0.844        0.663        0.382
```

```
0.198       1.643       1.316       1.036
0.752       0.556       0.271        0.14
1.585       1.239       0.928        0.61
0.388       0.186       1.549        1.16
0.785       0.464       0.213       1.534
0.975       0.605       0.273       1.395
0.843       0.352       1.151       0.499
0.79
```

Representación de las temperaturas interiores (figura2)

```
legend off
%si quisiesemos redondear el resultado:
%format shortg
%t=round(T,2)

%Representación de la malla
%Recta de temperatura 0
x0=0:1:8;
y0=-x0+12;
plot(x0,y0,'LineWidth',1.5,'Color','b');
hold on
grid on
grid minor
for a=1:7
    b=-a+12.5;
    y=-a+12;
    text(a,b,'0')
    plot(a,y,'o', 'color', 'b')
end


axis([-1 13 -1 13])
title('Figura 2: Temperaturas interiores calculadas')
hold on
grid on
grid minor

%Recta de temperatura 0
y1=0:1:4;
x1=8+0*y1  ;
plot(x1,y1,'LineWidth',1.5,'Color','b');
hold on
for b=1:4
    x=8;
```

```
    a=8.5;
    text(a,b,'0')
    plot(x,b,'o', 'color', 'b')
end

%Recta de temperatura 1
x3=0:1:8;
y3=0*x3;
plot(x3,y3,'LineWidth',1.5,'Color','g');
hold on
for a=0:8
    y=0;
    b=-0.5;
    text(a,b,'1')
    plot(a,y,'o', 'color', 'g')
end

%Recta de temperatura 2
y2=0:1:12;
x2=0*y2;
plot(x2,y2,'LineWidth',1.5,'Color','r');
hold on
for b=1:12
    x=0;
    a=-0.5;
    text(a,b,'2')
    plot(x,b,'o', 'color', 'r')
end

%temperaturas solucion en los puntos
c=0;
for y=1:10
    for x=1:7
        if y<-x+12
            c=c+1;
            plot(x,y,'.', 'color', 'k')
            str={T(:,c)};
            text(x-0.23,y+0.23,str,'FontSize',7)
        end
    end
end
hold off
```

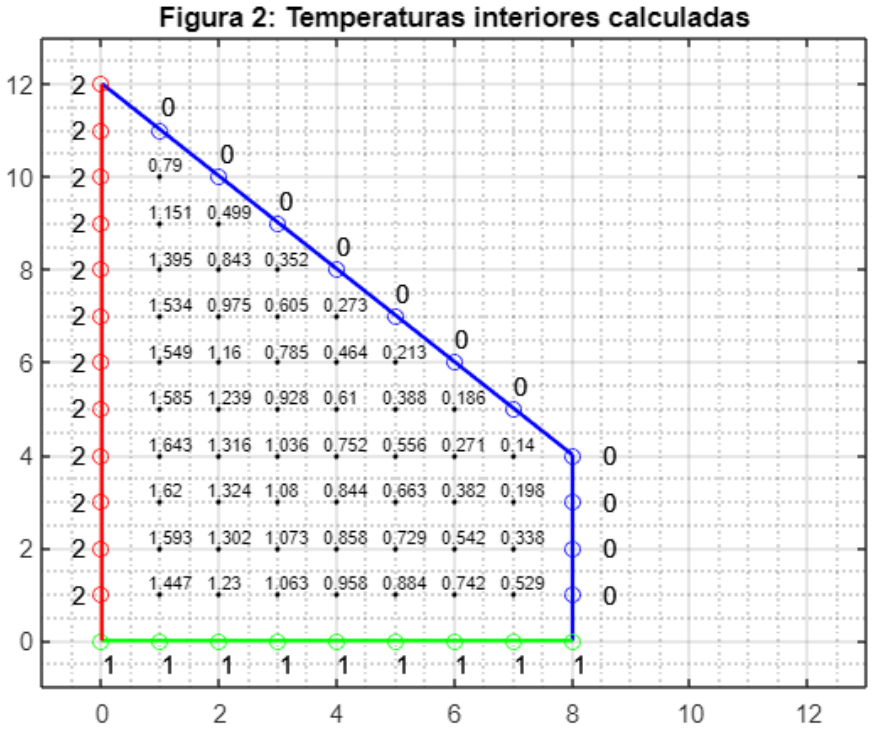


Figura 2: Placa con temperaturas interiores calculadas

0.4. Trabajo futuro

0.4.1. Otras ideas

Durante el proceso de pensar cómo resolver el problema, se nos ocurrieron varias maneras. Una de ellas fue la de crear caminos aleatorios a partir de espirales de distintas anchuras que choquen en algún momento con la pared. A nivel computacional parecía más eficiente, pero al desconocer las fórmulas de la espiral y su modificación, la idea se nos hizo demasiado complicada aunque no descartamos llevarla a cabo en un futuro para probar si funcionaría.

0.4.2. Posibles ampliaciones, mejoras...

Una posible ampliación de este trabajo, sería representar un mapa de temperaturas a color. Como se utiliza en las previsiones del tiempo. Este trabajo también nos serviría para organizar la distribución de aires acondicionados en una habitación, pues podríamos mejorar el programa permitiendo al usuario introducir cualquier punto de la placa donde desee conocer la temperatura y así ofrecer más utilidades.

Para mejorar la precisión de los resultados, bastaría con aumentar el nº de iteraciones de 1e7. Si por el contrario, no se busca precisión, se podría disminuir este número. O usar la función round para redondear los resultados obtenidos. Otro método de representar la solución de manera más visual aunque menos precisa sería con esta gráfica que hemos creado en Matlab:

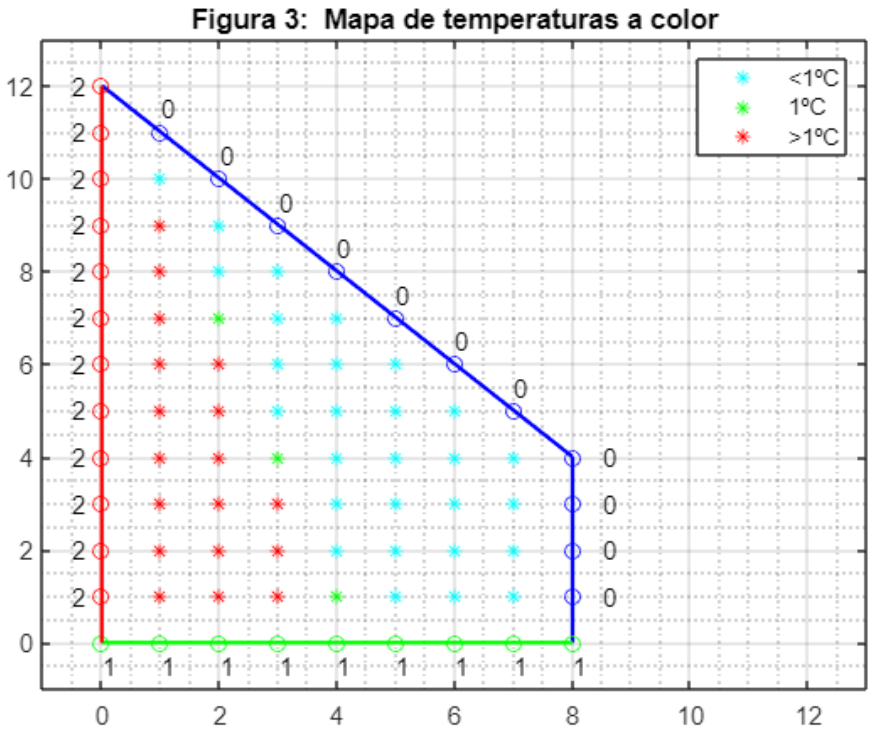


Figura 3: Placa con temperaturas a color

Representación de la Figura 3

```
c=0;
for y=1:10
    for x=1:7
        if y<-x+12
            c=c+1;
            if t(:,c)<1
                s0=plot(x,y,'*', 'color', 'c');
            end
            if t(:,c)==1
                s1=plot(x,y,'*', 'color', 'g');
            end
            if t(:,c)>1
                s2=plot(x,y,'*', 'color', 'r');
            end
            %str={t(:,c)};
            %text(x-0.23,y+0.23,str,'FontSize',7)
        end
    end
end
legend([s0 s1 s2],'<1ºC','1ºC','>1ºC','Location','northeast')
```

0.5. Conclusiones

LLegamos a la conclusión de 5 posibles métodos antes de comenzar a escribir en Matlab. Pero nos dimos cuenta de que el utilizado es el más preciso y sencillo de programar. Simplificar el problema y comenzar por averiguar la temperatura en un único punto concreto ha sido de gran ayuda para comprobar de manera rápida si el método era correcto y de esa manera lo hemos podido extrapolar luego a los 49 puntos.

El código consta de 280 líneas de las que: 229 son para guardar y representar datos. 51 son para el método de resolución del problema.

La representación de varias gráficas ha hecho el código mucho más largo pero más visual y sencillo de comprender para personas externas y además utilizando la misma figura base para todas las demás solo se ha de copiar y pegar cada vez que queramos volver a dibujarla, añadiendo algunas modificaciones.

0.5.1. Dificultades encontradas y consideraciones del trabajo en equipo

Sin duda, lo más complicado fue llegar a la idea para conseguir resolver el problema. Estuvimos planteando diversos métodos y comprobándolos, por lo que la lluvia de ideas en equipo nos benefició y gracias a eso pudimos comprender mejor el objetivo del proyecto y, por lo tanto, llevarlo a cabo.

0.6. Bibliografía

Documentación de Matlab. Documentación de Canvas. Vídeos extraídos de youtube: https://www.youtube.com/watch?v=7ESK5SaP-bc Páginas web:

colaboradores de Wikipedia. (2021, 17 octubre). Método de Montecarlo. Wikipedia, la enciclopedia libre.

EcuRed. (2006). Espiral de Arquímedes - EcuRed. ESPIRAL.

Ali, A. (2021, 20 noviembre). Número redondo en Matlab. Delft Stack.