Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Informe Tarea 1

Modelación y resolución numerica de EDP

Alumno: Joaquín Cruz
Profesores: Nancy Hitschfeld
Auxiliares: Pablo Pizarro R.
Pablo Polanco G.
Mauricion Araneda H.
Ayudantes: Joaquín Torres.
Maria José Trujillo.
Fecha de realización: 19 de abril de 2018
Fecha de entrega: 22 de Abril de 2018
Santiago, Chile

Índice de Contenidos

Lista de Figuras

Lista de Códigos

1. Introducción

El presente informe trata sobre la resolucion de la tarea 1 para el curso CC3501 Modelación y Computación Grafica para Ingenieros. El problema trata sobre la modelación del comportamiento termico que se genera al poner una planta en el litoral de central. Para esto se tiene que la temperatura de la atmosfera cumple con la ecuación de poisson:

$$\frac{\partial^2 T(x,y)}{\partial x^2} + \frac{\partial^2 T(x,y)}{\partial y^2} = \rho(x,y) \tag{1}$$

La funcion $\rho(x,y)$ era dada segun el caso del problema.
Además se requiere de hacer uso de los metodos numericos vistos durante el curso. Para generar esta modelación se requiere que el se visualize el corte transversal del litoral:

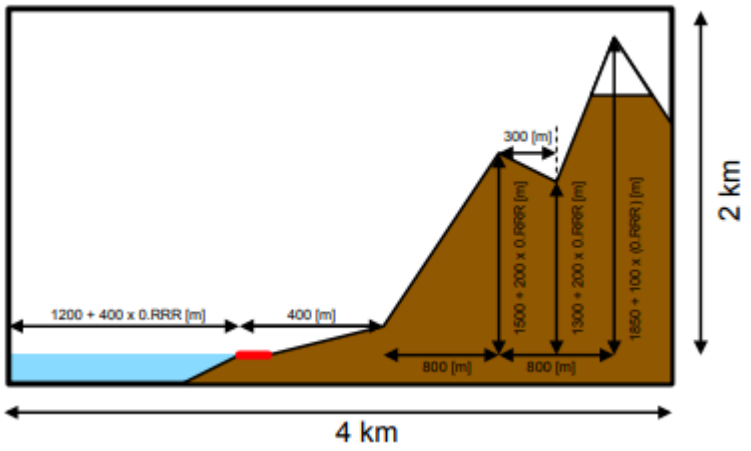

Figura 1: Terreno a generar.

En este caso, el valor de RRR es 0.692, pero es en realidad los tres ultimos digitos de la persona a utilizar el programa, que en este informe es 19.669.692-K.

2. Implementación

Para solucionar el problema, se implementó el uso de objetos de Python y el metodo de relajación sucesiva para resolver la ecuación (1). Primero se construyo un objeto llamado Planta que recibia como parametros un paso de altura (para la discretizacion del corte transversal del litoral), un digito verificador para la generación de los terrenos y tamaño del mar y por ultimo un coeficiente de relajación que fuera entre 0 y 2. Al objeto no se le da la altura debido que el problema estaba para ancho y altura de terrenos dada (4000 $[km]$ de ancho y 2000 $[km]$ de alto).
Ademas, el objeto creaba las siguientes variables internas:

1. Coordenadas a partir de el paso definido tanto para el ancho como la altura del terreno. A esta se les llamo **l** para el ancho y **h** para la altura.
2. Se creo una primera matriz para generar el resultado final llamada **matriz** cuyos valores iniciales fueron solo ceros.
3. Se creo una segunda matriz auxiliar para iterar en el metodo de relajacion sucesiva. A esta variable se le llamo **grilla** cuyos valores iniciales fueron solo ceros.
4. Se estableció una variable para pasar el dígito verificador a decimales cuyo nombre es **rrr**

Ahora a partir del objeto, se dividió el problema en los siguientes subproblemas, es decir, se planteo diferentes metodos para solucionar cierta caracteristica del problema, luego se juntan para generar una solución final a la iteración, estas se plantearon en la siguiente secuencia:

1. Primero se debe generar el terreno a partir de las condiciones dadas.
2. Una vez generado el terreno, se debe generar las condiciones de borde en cada caso.
3. Luego, se da paso al metodo de resolucion sucesiva para resolver el problema.
4. A continuacion, graficar la variable **matriz** para ver los efectos
5. Finalmente, implementar un metodo en el objeto que resuelva todo el problema uniendo las soluciones anteriores.

2.1. Generación de Terreno

Dadas las condiciones del terreno, se implemento el metodo **crearTerreno** que genera solo NaN's que sera lo que represente el terreno en el modelo.
Para poder construir el algoritmo del metodo, se tiene que el borde del terreno son rectas que parten de la posición final de la planta. A partir de la pendiente y la continuidad que tiene la forma del terreno, podemos deducir su ecuación. Ahora como es el borde del terreno, esta ecuación nos dara la altura maxima en cada paso, lo que generaria que debajo de esa altura el programa deberia generar solo valores NaN. Podemos definir una ecuacion por tramos que seria:

$$f(x) = \begin{cases} \mathrm{m}_1 x + n_1 & \text{si } x \text{ esta en el tramo (1)} \\ \mathrm{m}_2 x + n_2 & \text{si } x \text{ esta en el tramo (2)} \\ \mathrm{m}_3 x + n_3 & \text{si } x \text{ esta en el tramo (3)} \\ \mathrm{m}_4 x + n_4 & \text{si } x \text{ esta en el tramo (4)} \\ \mathrm{m}_5 x + n_5 & \text{si } x \text{ esta en el tramo (5)} \end{cases} \tag{2}$$

Lo proximo es determinar el conjunto de pendientes y los coeficientes de posición.

Es facil notar que, a partir del eje O se tiene que $n_1 = 0$ y dado los datos del problema $m_1 = \frac{1}{3}$.

Por otro lado, la altura maxima del tramo 2 esta dada por: $1500 + 200 \cdot \mathbf{rrr}$, por lo cual la pendiente m_2 esta dada por:

$$m_2 = \frac{1500 + 200 \cdot \mathbf{rrr} - m_1 \cdot x}{x' - x}$$

con x la posicion final del tramo anterior y x' el final del tramo (2). Para sacar n_2, usamos la continuidad del terreno. Como el terreno es continuo, se tiene que la funcion f es continua. Asi

$$m_2 \cdot x + n_2 = m_1 \cdot x$$

$$n_2 = m_1 \cdot x - m_2 \cdot x$$

Ahora bien, para m_3 y n_3:

$$m_3 = \frac{1300 + 200 * \mathbf{rrr} - 1500 - 200 * \mathbf{rrr}}{x'' - x'} = \frac{-200}{x'' - x'}$$

En esta ecuación se tiene que x' es el final del tramo (2) y x'' es el final del tramo (3). Entonces se tiene que el n_3 se puede sacar a partir de la continuidad del terreno:

$$m_3 \cdot x + n_3 = m_2 \cdot x + n_2$$

$$n_3 = m_2 \cdot x' + n_2 - m_3 \cdot x'$$

Luego tenemos un algoritmo para sacar todas las pendientes y coeficientes de posicion.
Con esto iniciamos un bucle para rellenar con NaN. El bucle en cada paso es el siguiente:

Código 1: Bucle para generar el terreno en pseudocodigo

```
    for x in range(A,B): #A y B son el inicio y el fin del tramo en el eje x
        xd=self._xdes(x) #desplazo la coordenada x en el plano
        ymax=self._h-f(xd) #genero la altura maxima a alcanzar por el terreno, llamemosla
     ymax
        for y in range(ymax,altura):
            matriz[y][xd]=np.nan

```

2.2. Condiciones de Borde

Para las condiciones de borde, se dividió en varios metodos privados que eran para el mar,planta, terreno y borde de la matriz. Todas estas fueron utilizadas en una unica funcion que recopilo todo por tramos llamada **cb** que recibia como parametro el tiempo para luego aplicar en las dos matrices las condiciones de borde correspondientes. Esta funcion iba recorriendo cada elemento dentro de la matriz. A partir de estas posiciones de los elementos, llamabamos a funciones de condiciones de borde correspondiente dado que su ubicacion dentro de la matriz determina si es mar, atmosfera o la planta de petroleo. Ahora bien, para generar las condiciones de borde del terreno, se implemento una funcion que determinaba si el elemento estaba cerca de una NaN, lo cual implicaria que en esa ubicacion se debia implementar las condicion del terreno. Las condiciones de borde eran:

$$cb\_mar(t) = \begin{cases} 4 & \text{si } 0<t\leq 8 \\ 2{\cdot}t - 12 & \text{si } 8<t\leq 16 \\ \text{-}2{\cdot}t + 52 & \text{si } 16<t\leq 24 \end{cases} \quad \text{(Tipo Dirichlet)} \tag{3}$$

$$cb\_terreno(y) = \begin{cases} 20 & \text{si } y<1800 \\ 0 & \text{si } y\geq 1800 \end{cases} \quad \text{(Tipo Dirichlet)} \tag{4}$$

$$cb\_planta(t) = 450 \cdot \left(cos(\frac{\pi}{12} \cdot t) + 2 \right) \text{ (Tipo Dirichlet)} \tag{5}$$

$$cbatmosfera(y,t) = \frac{-6}{1000} y + cb\_mar(t) \text{ (Tipo Dirichlet)} \tag{6}$$

2.3. Solucion Numerica

Se utilizo el metodo de relajacion sucesiva, por lo cual la solucion de la EDP numericamente esta en la ecuacion iterativa:

$$u_{ij} = u_{ij} + \omega \cdot \left(\frac{u_{(i+1)j} + u_{(i-1)j} + u_{(i(j+1)} + u_{i(j-1)} - 4 \cdot u_{ij} - h^2 \cdot \rho(x,y)}{4} \right) \tag{7}$$

En este caso, se tiene que :

$$\rho(x,y) = \frac{1}{\sqrt{x^2 + y^2 + 120}}$$

o bien

$$\rho(x,y) = 0$$

Esta solucion numerica se resolvio en dos metodos del objeto Planta que hacian iterar el programa. Una se llama ***itera*** que recibe una tolerancia, la funcion a utilizar y un numero maximo de iteraciones. La tolerancia se utiliza si el cambio en los numeros de la matriz no es tan significativo, se debe detener la iteracion. Por otro lado, el numero de iteraciones maxima se utiliza para detener las iteraciones y no entrar en un loop. Lo que en realidad hace esta funcion es llamar a otra que hace funcionar el metodo de relajacion sucesiva, esta funcion se llama ***relajacion_sucesiva*** que recibe las cordenadas de donde se esta iterando y la función para aplicar este metodo mediante la ecuacion (7).

Por lo tanto, los metodos del objeto Planta utilizadas para la solucion numerica son:

Código 2: Metodo Itera

```
def itera(self,tolerancia,rho,n_iteraciones=1000):
    e=1 #definimos primero, un epsilon alto
    iteracion=0
    while e>tolerancia and iteracion<=n_iteraciones:
        for x in range(1,self._l-1):
            for y in range(1,self._h-1):
               if np.isnan(self._grilla[y][x]):
                   break
                else:
                (self._matriz[y][x],e)=self.__relajacion_sucesiva(x,y,e,rho)
        iteracion+=1
        self._grilla=self._matriz #se actualiza la matriz de valores anteriores fijos

```

Código 3: Extracto del metodo relajacion_sucesiva

```
    if rho==0:
        r = self._w * (u_i1_j + u_1i_j + u_i_j1 + u_i_1j - 4 * self._grilla[y][x]) / 4
    else:
        xmetros=x*self._dh
        ymetros=y*self._dh
        r=self._w * ( u_i1_j + u_1i_j + u_i_j1 + u_i_1j - 4*self._grilla[y][x] - (self.
    _dh**2) * rho (xmetros,ymetros) ) / 4
    n=self._matriz[y][x]+r
    e=abs(r)
    if e<epsilon:
        epsilon=e
    return (n,epsilon)

```

Luego, para el usuario se empleo el metodo ***solve*** para unir todos lo pasos para resolver el problema planteado.

Código 4: Metodo Solve

```
def solve(self,t,epsilon,iteraciones_max,g):
    """
    Junta todas las funciones para generar solucion al problema planteado
    :param t: int
    :param epsilon: float
    :param n_iteraciones: int
    :param g: 0 o f
    :return:
    """
    self.crearTerreno()
    self.cb(t)
    self.itera(epsilon, g, iteraciones_max)
    self.plot()
```

3. Resultados

Para obtener los resultados en distintos tiempos, se utilizo el ω optimo visto en clases, un paso h de 25, una tolerancia de 0,001 un numero maximo de iteraciones de 1000. Ademas se registro la temperatura media del sistema en cada tiempo para el estado estacionario y el estado no estacionario. Luego se itero para un tiempo fijo ($t = 0$) y se midio el tiempo que demoraba el proceso al ir cambiando el factor de relajacion ω. Lo obtenido se aprecia en las siguientes subsecciones.

3.1. Resultados a las 0 horas

Para este tiempo se tuvo una temperatura media de $5,4[°C]$ en el caso estacionario y otra de $-26,21[°C]$ en el caso no estacionario.

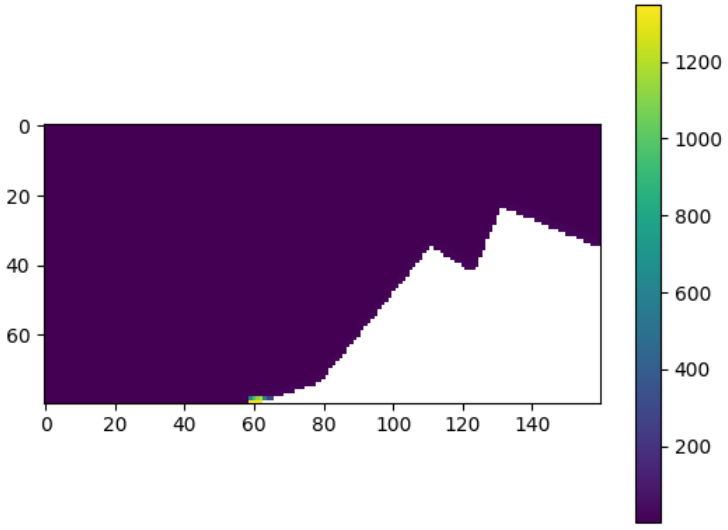

Figura 2: Resultados de la iteracion a las 0 horas y $\rho(x, y) = 0$

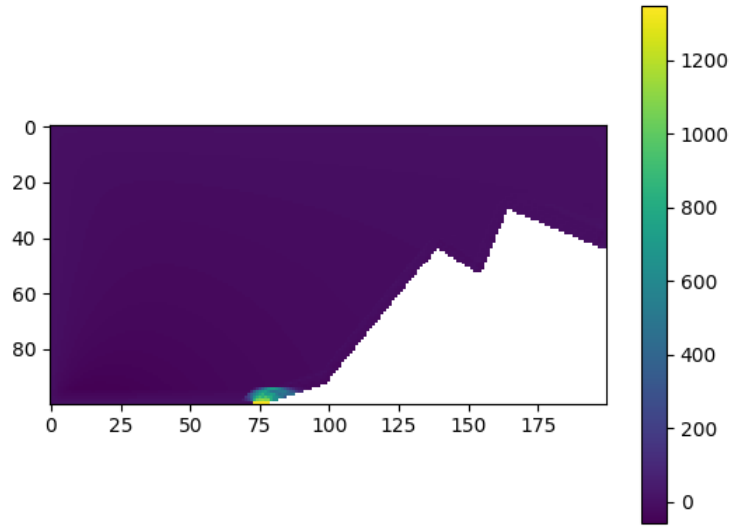

Figura 3: Resultados de la iteracion a las 0 horas y $\rho(x, y) \neq 0$

3.2. Resultados a las 8 horas

En esta hora, se obtuvo una temperatura media de $4,84[°C]$ y $-30,01[°C]$ respectivamente.

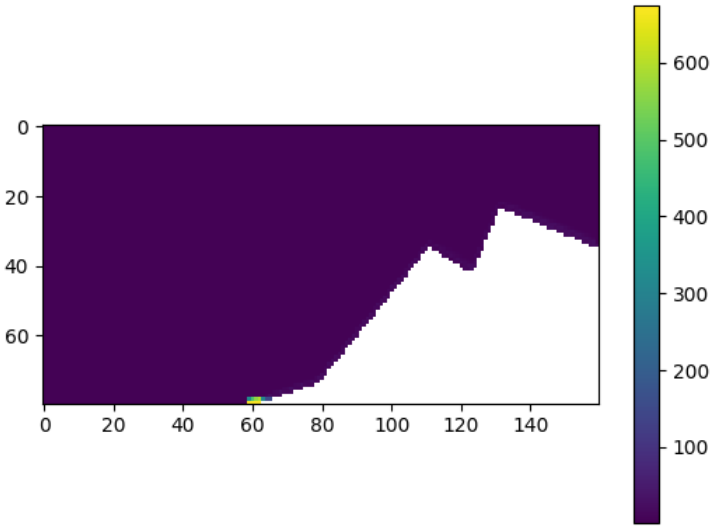

Figura 4: Resultados de la iteracion a las 8 horas y $\rho(x, y) = 0$

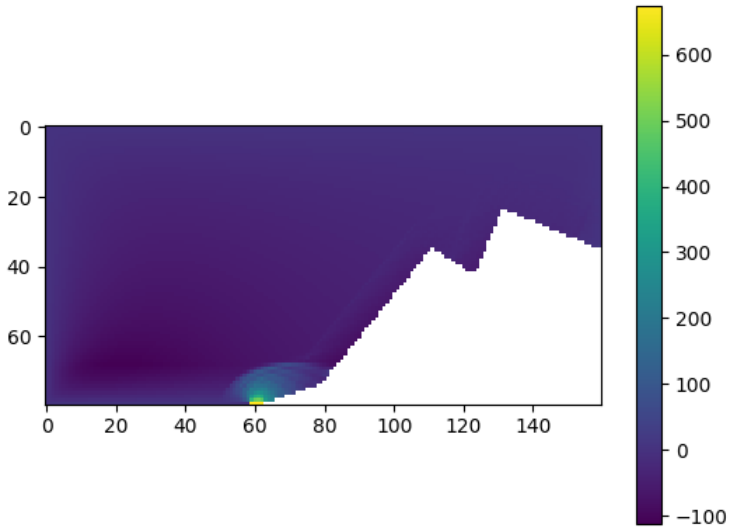

Figura 5: Resultados de la iteracion a las 8 horas y $\rho(x, y) \neq 0$

3.3. Resultados a las 12 horas

En este periodo, se obtuvo una temperatura media de $12,51[°C]$ en el primer estado y una de $-34,09[°C]$ en el segundo estado.

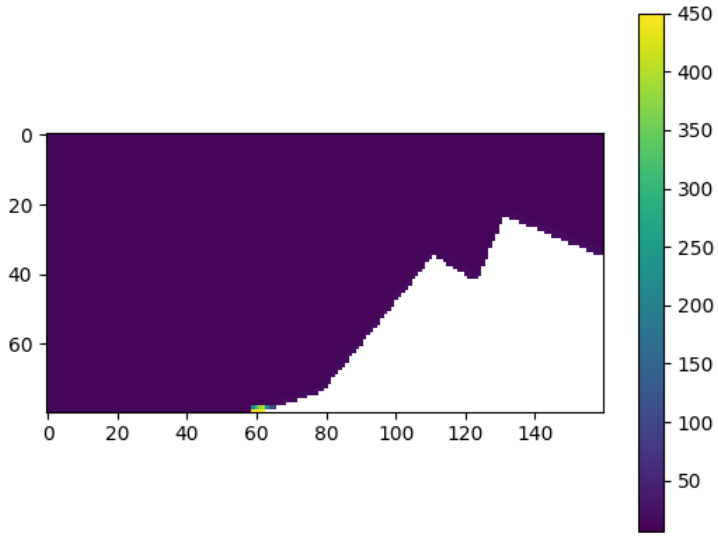

Figura 6: Resultados de la iteracion a las 12 horas y $\rho(x, y) = 0$

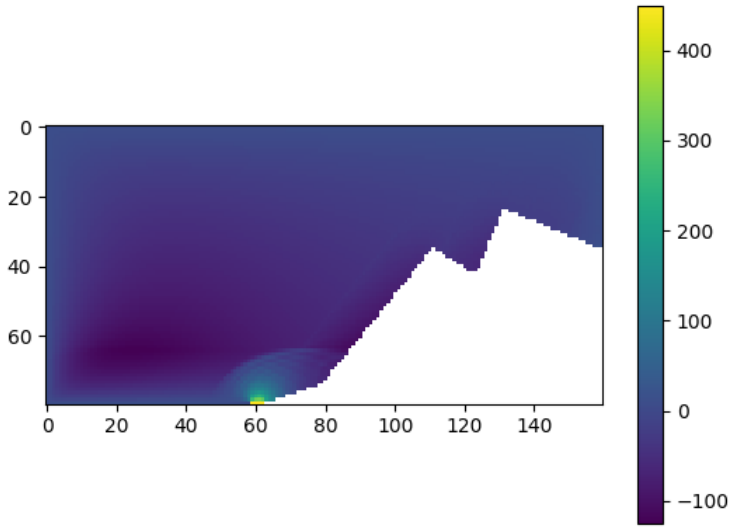

Figura 7: Resultados de la iteracion a las 12 horas y $\rho(x, y) \neq 0$

3.4. Resultados a las 16 horas

A las 16 horas se obtuvo una temperatura media de $20,52[°C]$ y de $-9,34[°C]$en el estado respectivo.

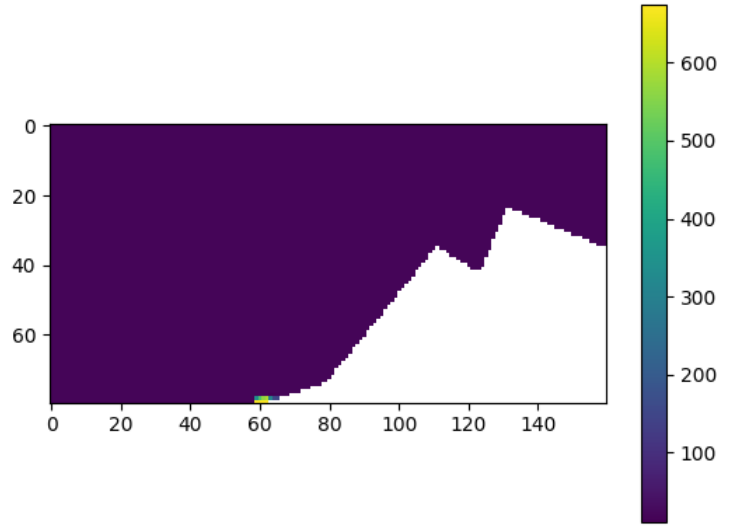

Figura 8: Resultados de la iteracion a las 16 horas y $\rho(x, y) = 0$

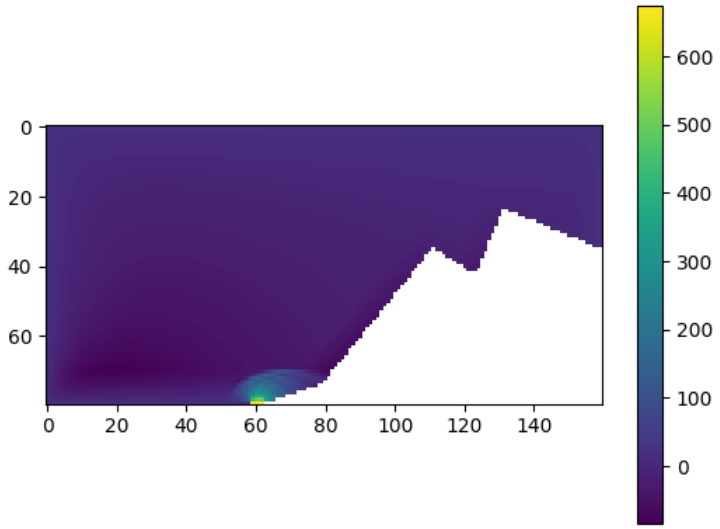

Figura 9: Resultados de la iteracion a las 16 horas y $\rho(x, y) \neq 0$

3.5. Resultados a las 20 horas

Las temperaturas medias en cada ecuacion son de $13,05[°C]$ y de $-9,56[°C]$

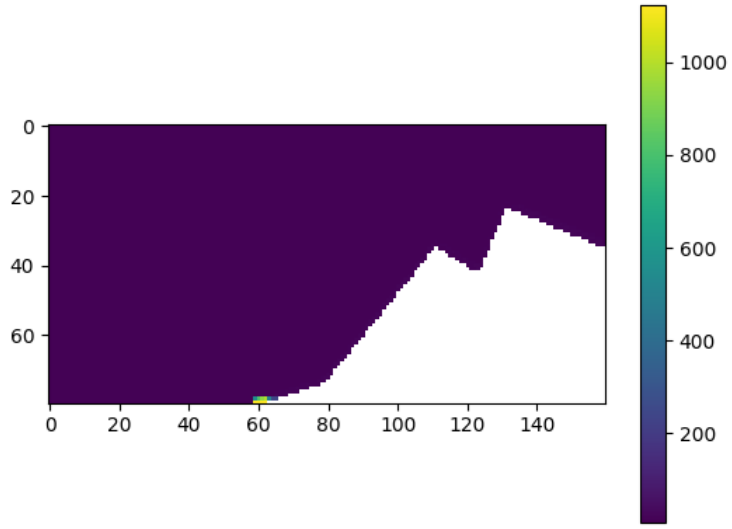

Figura 10: Resultados de la iteracion a las 20 horas y $\rho(x, y) = 0$

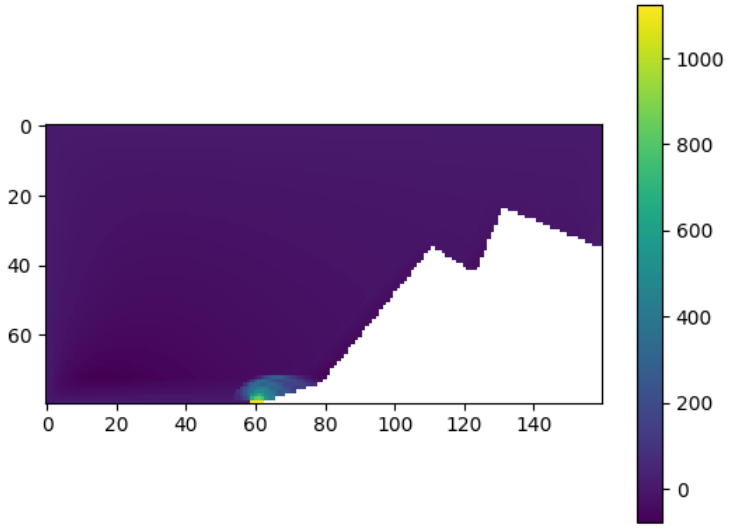

Figura 11: Resultados de la iteracion a las 20 horas y $\rho(x, y) \neq 0$

3.6. Resultados de los tiempos

En esta parte se fijo el tiempo en 0 horas y se utilizo la ecuacion estacionaria. Los resultados obtenidos son reflejados por la siguiente tabla y su grafico respectivo:

Coeficiente de relajación	Tiempo de ejecucion
0.25	0.5 $[s]$
0.5	0.51 $[s]$
0.75	0.52 $[s]$
1.0	0.52 $[s]$
1.25	0.53 $[s]$
1.5	0.55 $[s]$
1.75	0.57 $[s]$
1.93	0.62 $[s]$
2.0	0.68 $[s]$

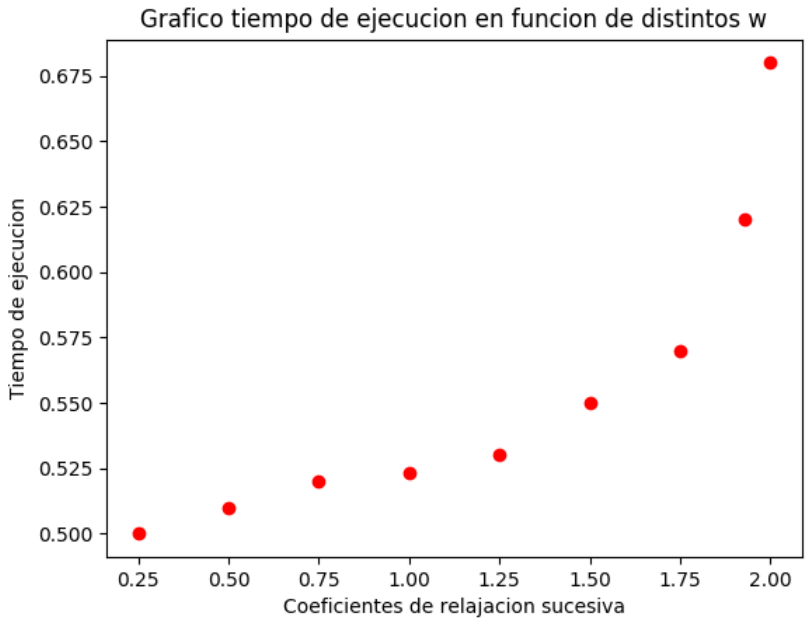

Figura 12: Graficos de la tabla 3.6

4. Discusión

A partir de los resultados obtenidos, se puede deducir que la propagacion de calor al ambiente producida por la planta en los distintos tiempos tienen sentido. Es decir, que solo afecta a las zonas cercanas a ella para el estado estacionario y estos efectos son periodicos durante el curso del dia. Esto da concordancia con la periocidad de la condicion de borde para la planta dada en la ecuacion (5) que al tener cosenos la planta debe tener una periocidad en su temperatura. Además, notemos que las temperaturas del sistema son bastante similares excepto las posiciones cercanas a la planta, que a pesar de ser casi las mismas, se puede observar un cambio en su temperatura maxima total, que tiene un maximo en $t = 0$ y un minimo en $t = 12$.
Podemos observar que ademas la temperatura media del sistema en este estado va en incremento teniendo un maximo de $20,52[°C]$ a las 16 horas del dia, pero que es cuando la temperatura maxima no es la maxima pero si es creciente, por lo tanto, debe existir un tiempo en el cual la temperatura media del sistema es maxima, pero la temperatura de la planta no es maximal.
Finalmente, no se puede notar si el mar o el terreno tienen relevancia con respecto a la temperatura del sistema, debido a las grandes temperaturas de la planta con respecto a estas condiciones de borde.

Por otro lado, si tenemos el caso no estacionario (es decir, $\rho(x, y) \neq 0$) se obtienen resultados bastante diferentes. Notemos que la periocidad de la temperatura de la planta se sigue manteniendo, pero esta vez la temperatura en las zonas altas del sistema tiene un incremento significativo. Por otro lado, la temperatura media del sistema presenta un decremento significativo, debido que para la ecuacion de relajacion sucesiva (7) el factor $\rho(x, y)$ genera un decrecimiento en la iteración que no estaba siendo considerado antes en la ecuacion de Laplace. Pero como la funcion decrece a partir de la altura y la posicion del largo del punto, las temperaturas para alturas grandes son mayores que en la ecuacion de Laplace. Notemos que para $t = 0$ y $t = 20$ vemos una capa de mayor temperatura sobre la planta de petroleo, que se va propagando a traves de la atmosfera por el incremento en la temperatura a mayor altura. Es aquí que se ve el efecto del mar y el terreno en el sistema debido a que las temperaturas más estan cercanas a estos bordes, por lo tanto, se puede hablar de un efecto generado por la presencia de estos dos componentes.

Por ultimo, se puede apreciar que mientras mayor el coeficiente de relajación, mayor se demora el programa. Esto se puede ser debido al pequeño cambio entre iteraciones, que generaria una rapida convergencia pero no necesariamente a los valores reales del problema. En cambio para los valores mayores que el optimo se tiene un mayor tiempo en el cual se puede producir mediante la superación de la cantidad maxima de iteraciones, lo cual produciria un valor mayor al real. En este caso, dio que el valor optimo obtenido por la formula (A.1) es el 1.93, que es el utilizado en los resultados anteriores.

5. Dificultades

Durante el desarrollo de la solución del problema se producieron bastantes dificultades.
El primer problema que se produjó fue la generación de terreno. Se tiene que la ecuación descrita en (2) es para un origen en la posición final de la planta y una altura medida desde la posición más abajo de la matriz. Por lo cual, al generarse el terreno en un principio, este se generaba en el lugar de comienzo de la matriz y en el extremo arriba de esta. Para solucionarlo se implementó un metodo privado en el objeto llamado ***x_des*** que hacia un desplazamiento en la coordenada x en la ubicación de un elemento. Para el otro caso, se debio restar la coordenada y a la cantidad de filas de la matriz pudiendo asi obtener la altura correcta en la generación de terreno.

Por otro lado, la identificación del borde del terreno y como hacerlo iterar debido a que la ecuación (7) falla. Como el paso de terreno es pequeño con respecto al terreno total, se hizo la aproximación de que la posición donde estaba el terreno era parecido al del lado opuesto, por ejemplo, si el NaN esta en la derecha del elemento de la matriz,el valor de $u_{(i+1)j} \approx u_{(i-1)j}$. Ahora para estos casos, se utilizó que los elementos del borde del corte transversal se guardaron en un arreglo interno del objeto Planta, además que para ver donde estaban los NaN's del terreno se creo una creo el metodo del objeto ***whereNAN*** que entrega un arreglo de Strings que contenian, en palabras como ."Arriba", las ubicaciones de los NAN's cercanos. Ahora para la variación a la ecuación (7), se implementó el siguiente algoritmo al principio del metodo *relajacion _sucesiva*:

- Primero se inician los $u_{(i+1)j}$, $u_{(i-1)j}$, $u_{i(j+1)}$ y $u_{i(j-1)}$ en el valor infinito, esto se establece gracias a la instrucción inf de la libreria numpy.
- Luego se ve si la posición de la matriz tiene NaN's cercanos. Si los tiene, con el metodo *whereNAN* creo el arreglo que me describe donde están.
- A continuación se itera para cada String en el arreglo generado, donde se reemplaza la posicion del NaN por el numero de la matriz que es opuesto a el, no el de la variable u_{kn} que es opuesta por que su valor es infinito.
- Entonces, se ve cual de los elementos sigue con valor infinito. Estos son reemplazados por sus respectivos valores.
- Finalmente, se itera como siempre.

La ultima dificultad (y la que no se pudo arreglar) fue que en los graficos de la solución al problema, como la matriz cuenta de forma inversa a las mediciones, el eje de las ordenadas esta invertido, es decir, el cero esta arriba y la coordenada maxima está abajo.

6. Conclusión

Se puede decir que, mediante el metodo de relajación sucesiva, se pudo dar solución al problema de las temperaturas en el litoral que se daban a generar por la inclusión de la planta de petroleo. Además se pudo observar distintos efectos al poner la planta debido a la ecuacion que rige la temperatura en funcion de la altura y el ancho del litoral en toda la atmosfera, generando dos modelos que los representan con resultados distintos, que hacen sentido con los datos iniciales descritos en el problema.

Pero, por otra parte, si la empresa quisiera ver los efectos del petroleo en el mar o medir la temperatura en otros sitios, que a pesar de ser problemas interesantes a estudiar, el modelo generado no los resolveria, debido que no es lo que este objeto soluciona. Este solo soluciona el problema planteado en esta situación y con estas condiciones de borde. Ahora bien, dado el estudio de la planta de petroleo, pero si lo resuelve para distintos digitos verificadores del rut de distintas personas. Aunque, ¿Se le podrá dar solución numerica al estudio de su impacto en las aguas del litoral?.

Anexo A. Formulas

1. El ω optimo viene dado por:

$$\omega = \frac{4}{2 + \sqrt{4 - \left(\cos\left(\frac{\pi}{n-1}\right) + \cos\left(\frac{\pi}{m-1}\right)\right)^2}} \tag{A.1}$$

   con n y m son el alto y el ancho de la matriz.

2. Por otro lado, el metodo plot del objeto Planta fue sacado del auxiliar que realizó Pablo Pizarro en el cual se pedia resolver el problema de los pilares en el rio. El codigo esta en https://github.com/ppizarror/CC3501-2018-1/tree/master/tareita%201 en el archivo rio.py .