FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2

[75.12] Análisis Numérico I
Segundo cuatrimestre de 2018

Alumno	Padrón	Mail
Kristal, Juan Ignacio	99779	kristaljuanignacio@gmail.com

Curso 07:

- Dr Daniel Fabian Rodriguez
- Valeria Machiunas
- Federico Balzarotti
- Michael Portocarrero

75.12 ANÁLISIS NUMÉRICO I

FACULTAD DE INGENIERIA
UNIVERSIDAD DE BUENOS AIRES

TRABAJO PRACTICO Nº 2

2do Cuatrimestre 2018

Métodos numéricos aplicados a la Ingeniería de procesos

Preparado por Ing. Federico Balzarotti

OBJETIVOS

- Experimentar con el uso de métodos numéricos la resolución de ecuaciones diferenciales no lineales.
- Aplicar métodos numéricos de diferentes temas de la materia y ensamblarlos para resolver un problema de mayor relevancia.
- Adquirir conocimientos básicos de la ingeniería de procesos industriales.

INTRODUCCIÓN

Uno de los procesos más comunes de la industria metalúrgica es el tratamiento térmico de aceros. Las múltiples aplicaciones de esta aleación Fe-C (automotriz, petróleo, tuberías, perfiles, etc.) requieren propiedades mecánicas muy diversas.

En este sentido, los tratamientos térmicos suelen ser una herramienta versátil y rentable para lograr un objetivo de resistencia mecánica y/o tenacidad.

Para una misma composición química es posible modificar las propiedades mecánicas del acero sometiendo el material a una serie de calentamientos y enfriamientos sucesivos.

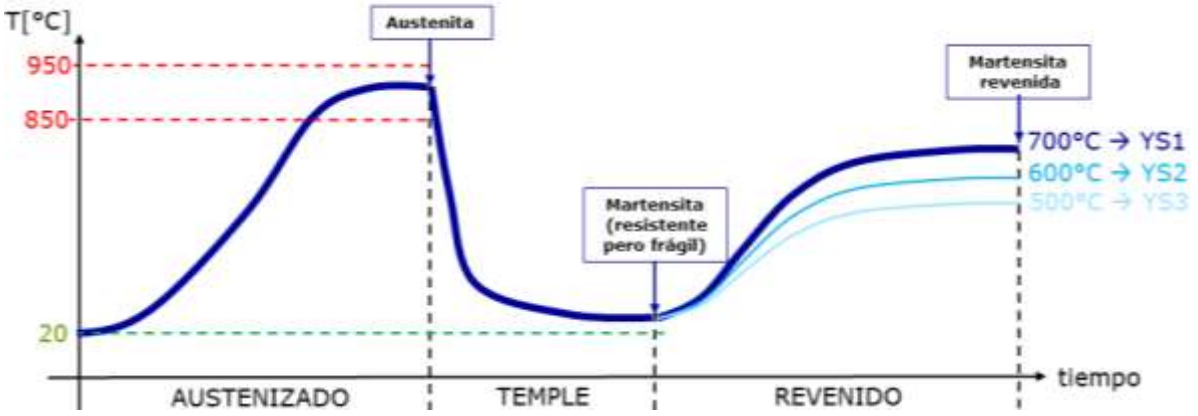
Uno de los tratamientos térmicos de gran aplicación en la industria es el *Temple y Revenido*, que consiste en las siguientes etapas:

1) *Austenizado*: Calentamiento hasta una temperatura de aproximadamente 900°C. En este proceso el material modifica completamente su microestructura (transformación alotrópica) y cambia a la fase Austenita.

2) *Temple*: Enfriamiento brusco hasta temperatura ambiente, generalmente utilizando agua. Se obtiene una nueva fase llamada Martensita. En esta condición el material tiene alta resistencia mecánica pero a la vez es muy frágil (baja tenacidad).

3) *Revenido*: Nuevo calentamiento hasta una temperatura objetivo. El rango habitual es 500-700°C. A medida que el material se calienta la Martensita disminuye su resistencia mecánica ganando tenacidad. La temperatura final definirá la combinación final de propiedades mecánicas.

En el siguiente esquema se puede visualizar el proceso completo de temple y revenido:



El presente trabajo se focalizará en modelar la evolución temporal de la tercera etapa dicho tratamiento térmico. El material a revenir serán tubos de acero destinados a la industria petrolera. La norma *API* es la encargada de definir los requerimientos de propiedades mecánicas según la profundidad y las condiciones particulares del pozo. En este contexto es necesario contar con un proceso que garantice el producto final dentro de rango de propiedades mecánicas.

Para lograr un proceso continuo de calentamiento, los tubos avanzan dentro de un horno a velocidad constante. Cada cierto intervalo de tiempo (cadencia) un tubo sale del horno a la temperatura objetivo, a la vez un nuevo tubo entra a temperatura ambiente.

Por otra parte, el horno debe tener la suficiente versatilidad para calentar tubos de distintas dimensiones, así como también lograr diferentes objetivos de temperatura. Por este motivo los hornos son diseñados con varias zonas, pudiendo establecer temperaturas independientes en cada una.

PLANTEO DEL PROBLEMA

Se deberá resolver la evolución temporal para un tubo de diámetro externo *OD*, espesor *WT* y longitud *Lt* que viaja con velocidad v_0 a través de un horno de largo *L* con dos zonas independientes. La primera mitad del horno controla a una temperatura T_1 y la segunda a una temperatura T_2.

El intercambio de calor entre un sólido y su entorno está dado por la ley de conservación de energía:

$$-mC\frac{dT}{dt} = h_c S(T - T_\infty) + \sigma\varepsilon S(T^4 - T_\infty^4)$$

Donde:

m	[kg]	masa del sólido
C	[J/kg K]	calor especifico del sólido
T	[K]	temperatura del solido
t	[s]	tiempo
hc	[W/ m^2 K]	coeficiente de convección
S	[m^2]	área de intercambio de calor por convección/radiación
T_∞	[K]	temperatura del entorno
σ	[W/ m^2 K^4]	constante de Stefan-Boltzmann
ε	[]	factor de emisividad de la superficie del tubo

El lado izquierdo de la ecuación es la tasa de energía interna almacenada en el sólido, mientras que el lado derecho incluye los intercambios instantáneos de energía por convección y radiación (notar que los términos de la ecuación tienen unidades de potencia). El modelo planteado considera la conductividad térmica del material $\lambda \rightarrow \infty$, es decir que se adopta la hipótesis de sólido de temperatura uniforme (todos los puntos materiales tienen la misma temperatura para un dado instante de tiempo).

La temperatura T_∞ será la temperatura dentro del horno según la zona en la que se encuentre el tubo.

$$T_\infty(x) = \begin{cases} T_1 & x \leq \frac{L}{2} \\ T_2 & x > \frac{L}{2} \end{cases} \qquad \text{con} \qquad x = v_0 t$$

Los tubos avanzan de manera discreta por un mecanismo llamado *walking-beam* ubicándose en espacios equidistantes llamados *bolsillos*. La *cadencia* es el tiempo en segundos que tarda un tubo en pasar de un bolsillo al siguiente. De este modo, la velocidad v_0 se calcula a partir de los parámetros mencionados:

$$v_0 = \frac{L}{nbols \times cad}$$

Por último la masa *m* se puede calcular a partir de la densidad del acero ρ y del volumen del tubo:

$$m = \rho . \pi . OD . WT . \left(1 - \frac{WT}{OD}\right) L_t$$

DESARROLLO DEL PRÁCTICO

1) Resolver el modelo planteado considerando solamente el intercambio de calor por convección. Aplicar los métodos de *Euler* y *Runge Kutta de orden 4* para una condición inicial $T_0=20°C$ y paso de tiempo *h=cad*. Considerar los siguientes datos reemplazando *NP* por el número de padrón de uno de los integrantes del grupo:

Propiedades del material	ρ	7850 kg/m^3
	C	480 J/kg K
Geometría del material	*OD*	244.48 mm
	WT	13.84 mm
	Lt	12 m
Geometría del horno	*L*	50 m
	nbol	50
Parámetros de proceso	*cad*	ROUND (-10 s / 10000 * (NP - 90000) + 35 s)
	T_1	ROUND (200 °C / 10000 * (NP - 90000) + 500°C)
	T_2	ROUND (200 °C / 10000 * (NP - 90000) + 500°C)
Parámetros de la transferencia de calor	*hc*	20 W/m^{2}K
	σ	5.6703. 10^{-8} W/m^2 K^4
	ε	0.85

* Para un padrón NP=100000 los parámetros de proceso serían: *cad*=25s, T_1=700°C, T_2=700°C.
**ROUND indica redondeo simétrico al valor entero.

Notar que para este caso particular donde $T_\infty=T_1=T_2$, la solución analítica (exacta) tiene la siguiente expresión:

$$T(t) = T_\infty + (T_0 - T_\infty)e^{-\frac{h_c S}{mC}t}$$

a) Graficar la temperatura en función del tiempo superponiendo los resultados de cada método junto con la solución exacta.
b) Graficar el error relativo cometido con cada método en función del tiempo. Utilizar una escala en el eje vertical que permita visualizar con claridad ambas curvas.
c) Obtener conclusiones sobre la precisión de ambos métodos aplicados.

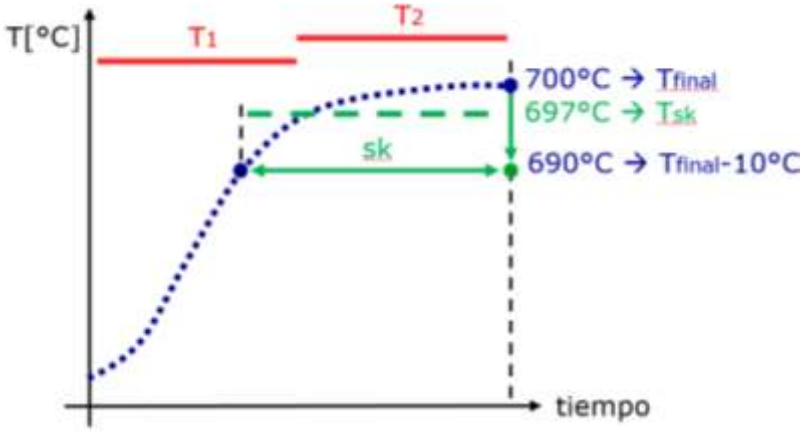
IMPORTANTE: Todos los **cálculos** de temperatura y tiempo deberán realizarse en grados Kelvin [K] y segundos [s], respectivamente. Por otro lado, los **resultados** (ya sean numéricos o gráficos) deberán expresarse en grados Celsius [°C] y minutos [min].

2) Resolver nuevamente el *ítem 1* incorporando el término de intercambio radiativo.
a) Seleccionar el método que considere más adecuado de acuerdo a los resultados *del ítem 1*. Justificar la elección.
b) Graficar los nuevos resultados superponiéndolos con la solución exacta del *ítem 1*. Decidir si el intercambio por radiación es despreciable.
c) A partir de la evolución temporal obtenida en 2a) calcular los siguientes *parámetros de salida* del proceso. Estos parámetros son de gran interés porque influyen directamente en las propiedades mecánicas finales del tubo:

> *Sk* → *Soaking*. Intervalo de tiempo para el cual el material permanece dentro del horno en el rango de temperatura más elevada. Dicho rango se define como [T_{final} -10°C, T_{final}].
>
> T_{sk} → Temperatura promedio durante el *soaking*. Representa la *temperatura objetivo* del proceso de revenido.

En el siguiente gráfico se puede visualizar cualitativamente cada parámetro:
- En color rojo se indican los *parámetros de entrada* al proceso
- En color verde los *parámetros de salida*.
- En color azul se indica el *modelo numérico desarrollado*.

Los *parámetros de salida* deberán calcularse a partir de los datos de la curva azul.



*Aclaración: Si bien en el grafico se observa que T_1 y T_2 son cualitativamente diferentes, los valores a utilizar para resolver los ítems 2a) 2b) y 2c) deben ser los de tabla del ítem 1 ($T_1=T_2$).

3) Encontrar manualmente una combinación de T_1 y T_2 para que el *Sk* sea de *10 minutos* manteniendo la T_{sk} obtenida en el *ítem 2*. La tolerancia para *Sk* es de ±*1min*.

4) Automatizar la búsqueda manual del *ítem 3* para encontrar valores de T_1 y T_2 que obtengan un T_{sk} y *sk* predeterminados. El problema se puede esquematizar de la siguiente forma:

Fijando los parámetros de salida como sk_{obj} y T_{sk-obj}, debería existir una combinación de parámetros de entrada T_1 y T_2 que satisfaga el siguiente *sistema no lineal* de ecuaciones:

$$f_1(T_1, T_2) - sk_{obj} = 0$$
$$f_2(T_1, T_2) - T_{sk-obj} = 0$$

Si bien las funciones f_1 y f_2 no se conocen explícitamente, ambas están definidas por el modelo numérico desarrollado previamente. Los parámetros T_1 y T_2 serán las raíces del sistema formado por F_1 y F_2:

$$F_1(T_1, T_2) = f_1(T_1, T_2) - sk_{obj} = 0$$
$$F_2(T_1, T_2) = f_2(T_1, T_2) - T_{sk-obj} = 0$$

De este modo, es posible plantear un esquema de punto fijo para resolver el problema de manera iterativa:

$$\begin{bmatrix} T_1^{(k+1)} \\ T_2^{(k+1)} \end{bmatrix} = \begin{bmatrix} T_1^{(k)} \\ T_2^{(k)} \end{bmatrix} - \boldsymbol{J^{-1}} \begin{bmatrix} F_1\left(T_1^{(k)}, T_2^{(k)}\right) \\ F_2\left(T_1^{(k)}, T_2^{(k)}\right) \end{bmatrix}$$

Según cual sea el jacobiano *J*, el método será *Newton-Raphson*, *Secante* o algún esquema de *Punto Fijo*. A continuación se detallan las expresiones correspondientes:

$$J_{NR} = \begin{bmatrix} \frac{\partial F_1}{\partial T_1} & \frac{\partial F_1}{\partial T_2} \\ \frac{\partial F_2}{\partial T_1} & \frac{\partial F_2}{\partial T_2} \end{bmatrix}_{T_1^{(k)}, T_2^{(k)}} \quad J_{SEC} = \begin{bmatrix} \frac{F_1^{(k)} - F_1^{(k-1)}}{T_1^{(k)} - T_1^{(k-1)}} & \frac{F_1^{(k)} - F_1^{(k-1)}}{T_2^{(k)} - T_2^{(k)}} \\ \frac{F_2^{(k)} - F_2^{(k-1)}}{T_1^{(k)} - T_1^{(k-1)}} & \frac{F_2^{(k)} - F_2^{(k-1)}}{T_2^{(k)} - T_2^{(k)}} \end{bmatrix} \quad J_{PF} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Para resolver el sistema no lineal planteado anteriormente se sugiere utilizar el siguiente *J*:

$$J = \begin{bmatrix} 0.25 & 0.75 \\ 0.75 & 0.25 \end{bmatrix}$$

Utilizar como criterio de corte 3 dígitos significativos tanto para T_1 como para T_2.

Se deberá resolver los casos indicados en la siguiente tabla:

caso	***sk***	**Tsk**	**T_1**	**T_2**	**N° iteraciones**
A	*10*	*Tsk* del *item3*			
B	*10*	ROUND (100 °C / 10000 * (NP - 90000) + 550°C)			
C	*10*	ROUND (100 °C / 10000 * (NP - 90000) + 600°C)			

Completarla con los resultados de temperaturas T_1, T_2 y el número de iteraciones requerido. Si encuentra otro *J* que logra también la convergencia puede utilizarlo. Indicarlo claramente en el informe.

Graficar la evolución temporal de los casos A, B y C, indicando los parámetros de entrada y salida (similar al grafico cualitativo del *ítem 2*).

CONCLUSIONES

Presente sus conclusiones del trabajo práctico. En particular, comente sobre:

- Ecuaciones diferenciales lineales vs no lineales. Ventajas y desventajas de los métodos numéricos frente a los analíticos para cada caso.
- Sistemas de ecuaciones no lineales como herramienta para la resolución de problemas inversos (inferencia de datos de entrada a partir de datos de salida).
- Modificaciones y agregados al modelo planteado para acercarse más a la realidad.

Índice

1. Introducción

El trabajo práctico tiene como objetivo la resolucion de ecuaciones diferenciales mediante distintos metodos y la comparacion de los mismos asi como la resolucion de sistemas de ecuaciones no lineales. Para ello, se utilizara un modelo para el tratamiento termico de tubos de acero mediante la ley de la conservacion de la energia. La siguiente es la ecuacion diferencial a resolver:

$$-mC\frac{dT}{dt} = h_c S(T - T_\infty) + \sigma\epsilon(T^4 - T_\infty^4) \tag{1}$$

Específicamente:

- Se estimarán los valores de temperatura (por conveccion y radiacion) a lo largo del tiempo.
- Se determinará que algoritmo resulta ser mas exacto.
- Se obtendra una temperatura de soaking y un tiempo de soaking para el proceso.
- Se variará la temperatura del horno de trata en diferentes sectores para alcanzar valores deseados de soaking.
- Se automatizará dicho proceso mediante la resolucion de un SENL.

2. Desarrollo

2.1. Calculo de temperatura

Para los cálculos de temperatura se utilizaron dos metodos de resolucion de ecuaciones diferenciales. El metodo de Euler y el metodo de Runge-Kutta de cuarto orden (a partir de ahora, RK4). Se utilizo el algoritmo del libro de Hernan Gonzales [1]. Dicho calculo fue dividido en dos partes, la temperatura generada por el proceso de conveccion y la del proceso de radiacion.

2.2. Proceso de conveccion

El proceso de conveccion estaba dado por una ecuacion diferencial de variables separables por lo que su resolucion matematica era posible y sencilla. De este modo se lograron calcular estimaciones de dicha funcion con ambos metodos y se observa una comparacion con los valores exactos. Con la informacion exacta es posible determinar el error relativo de ambos metodos en base a dichos valores.

El grafico la función exacta y los valores calculados numericamente de ambos metodos pueden observarse en la figura 4

Usando la formula del calculo de errores relativos fue sencillo obtener el error de cada metodo en cada valor de tiempo particular. En la figura 5 se puede observar un grafico comparativo en escala logaritmica para una mejor apreciacion. En dicho grafico se observa claramente que el metodo de RK4 tiene un error menor y por lo tanto es el metodo que se usará para el proximo calculo. Esperable dado que el orden de error de RK4 es mayor.

2.3. Proceso de conveccion y radiacion

Con RK4 determinado como el algoritmo más exacto se resolvera mediante dicho metodo la ecuacion diferencial completa. En este caso no hay una función exacta para comparar y observar el error pero determinamos que RK4 es mas preciso que Euler.

A modo de comparación se usará la función calculada previamente con unicamente la conveccion y se observa que el intercambio por radiacion es claramente significativo en la figura 6

2.4. Temperatura y tiempo de soaking

Dado que tanto la temperatura como el tiempo de soaking son valores muy importantes para este proceso fueron calculados para los valores iniciales de temperatura. Con el horno a temperatura constante a 696 °C. Para obtener este valor de tiempo se busco el punto donde la temperatura alcanzada es de 10 °C menor a la final. Para alcanzar la temperatura de soaking se busco el promedio de temperatura durante ese intervalo de tiempo.

Luego se variaron las temperaturas para alcanzar un tiempo de soaking menor. Con el tiempo de soaking a 10 minutos fijo, se variaron las temperaturas manualmente hasta llegar a dicho tiempo y luego se planteo un sistema de ecuaciones no lineales el cual fue resuelto con el metodo de punto fijo para obtener un resultado más preciso. Se utilizaron 2 cifras decimales como condicion de corte pues el problema, por algun motivo desconocido, no lograba a conseguir errores menores para valores de temperatura menores a los 800 °C.

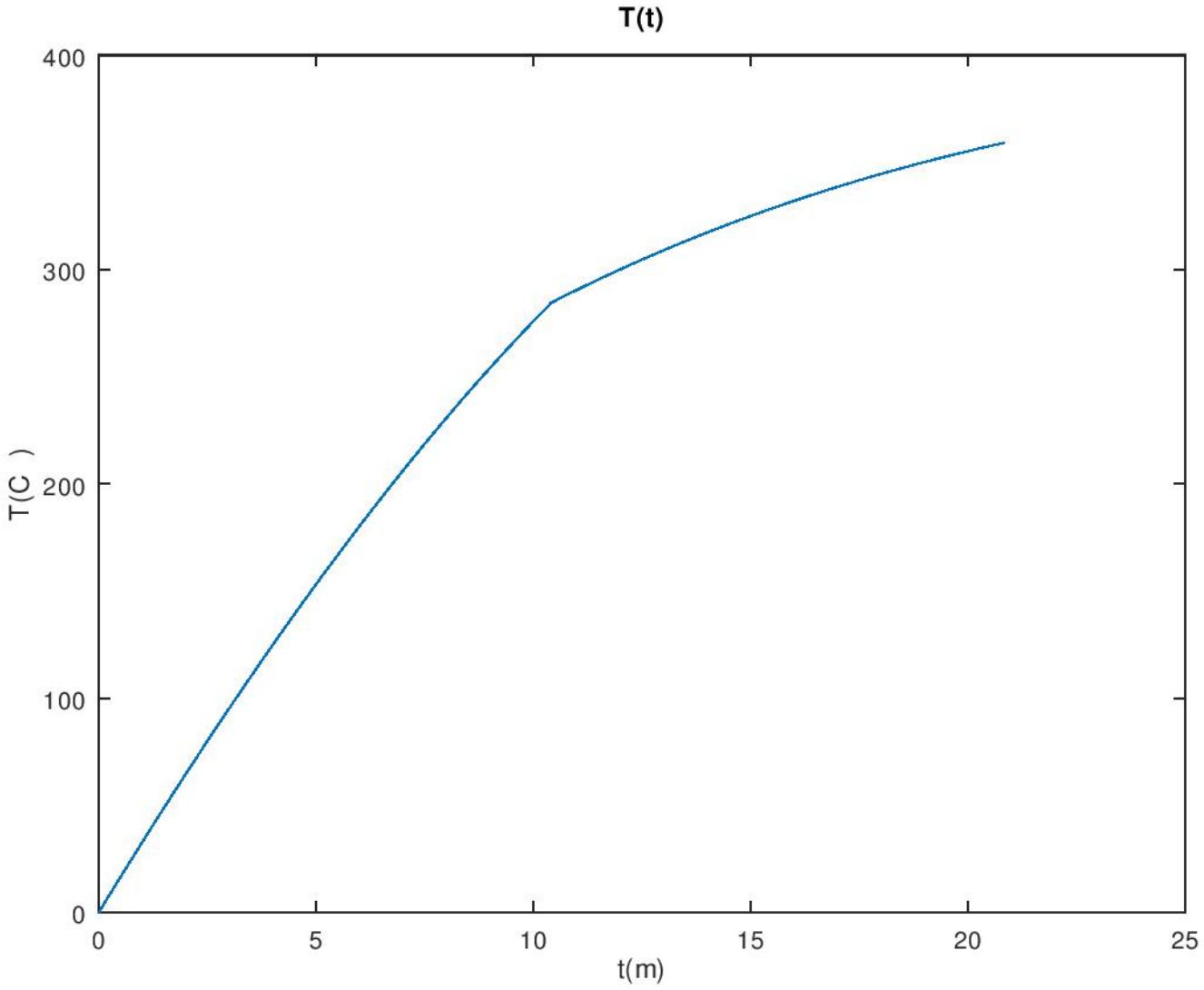


Figura 1: Evolución de 696°C

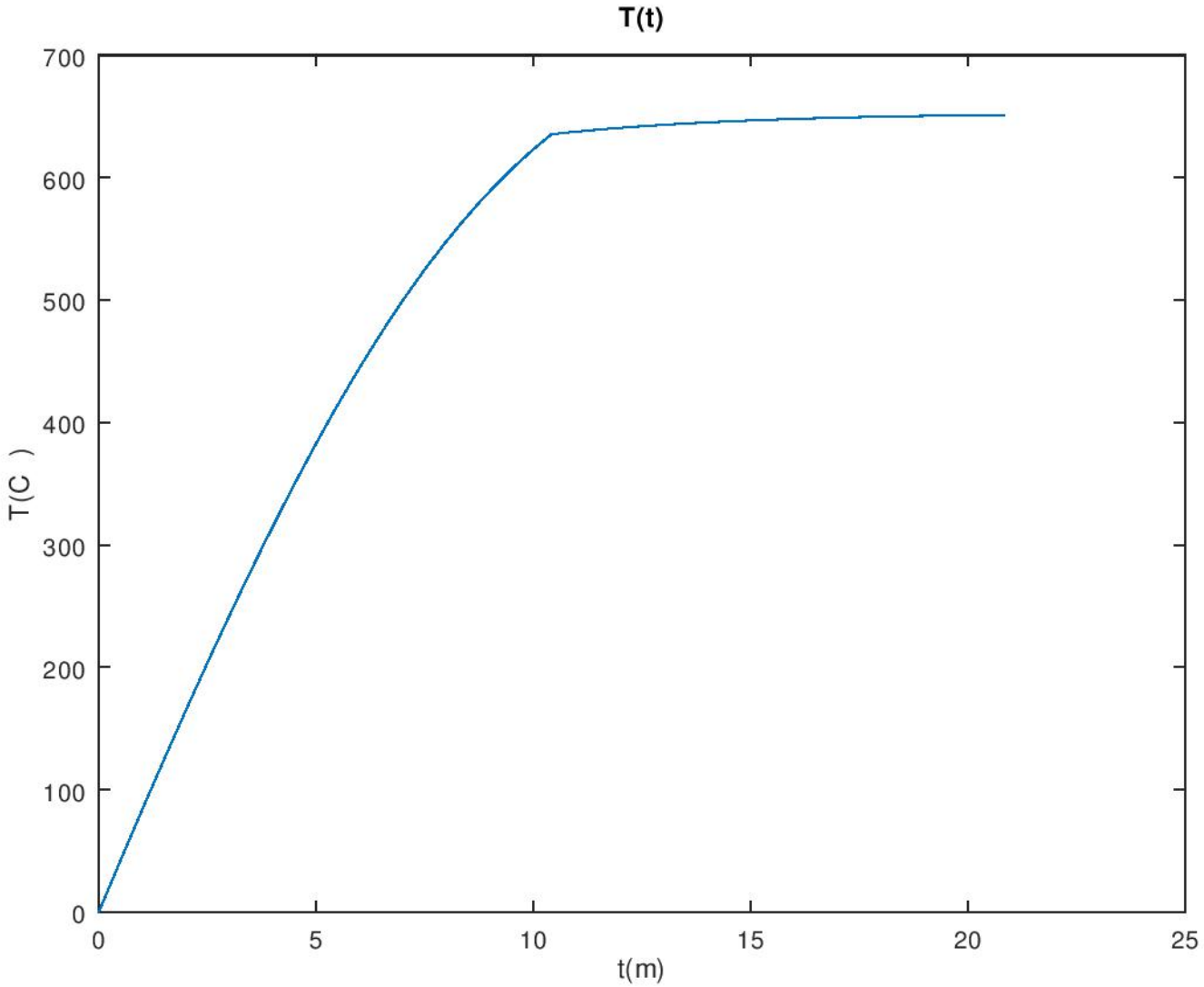


Figura 2: Evolución de 648°C

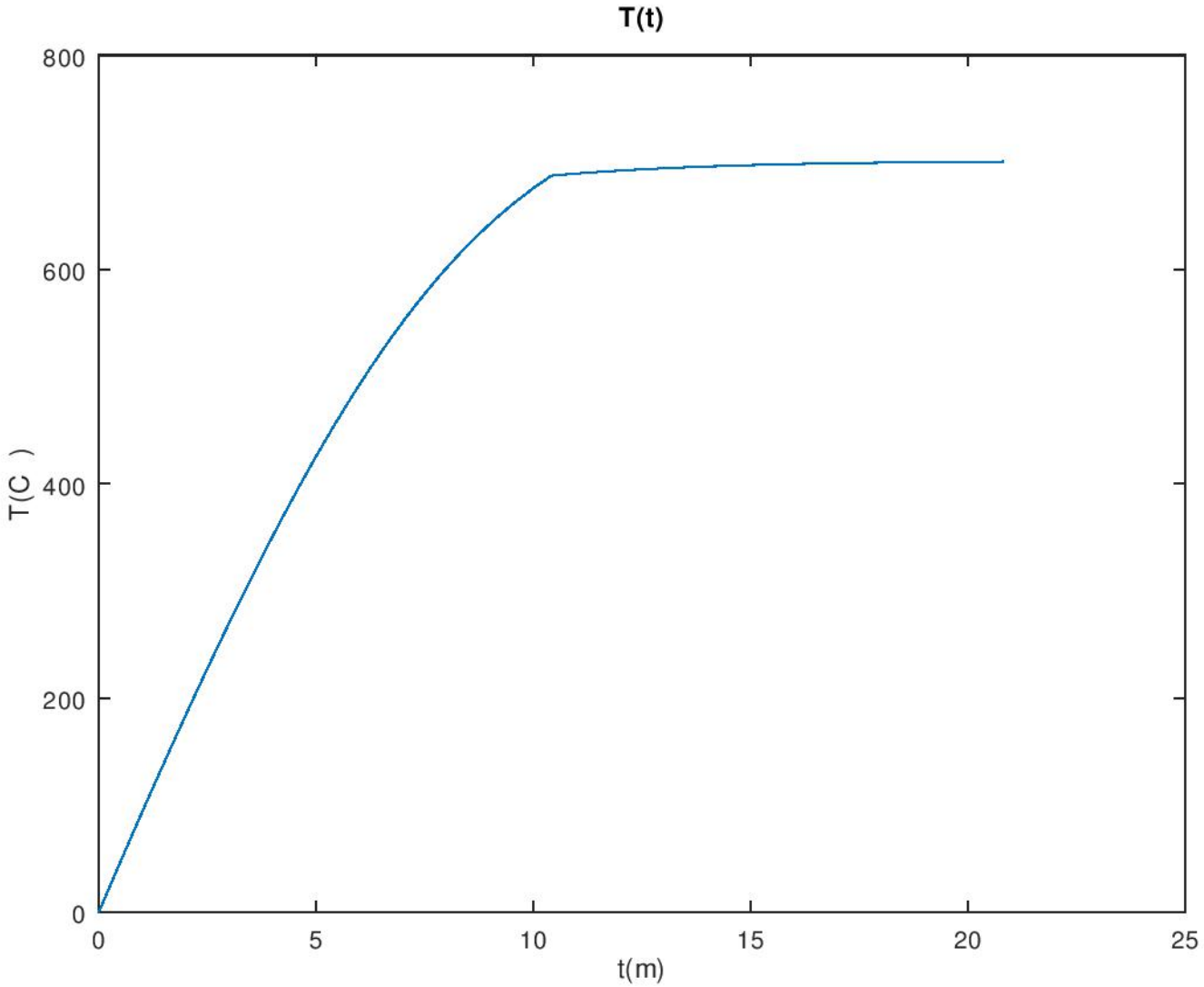


Figura 3: Evolución de 698°C

Finalmente se tomaron 3 temperaturas (696 °C, 648 °C y 698 °C) y se resolvio el sistema, graficando la evolucion de la temperatura en las figuras 1, 2 y 3 respectivamente.

3. Resultados

3.1. Graficos comparativos y evolucion de la temperatura respecto al tiempo

Utilizando ambos metodos de resolucion de ecuaciones diferenciales se llegaron a los siguientes resultados para calor por conveccion unicamente:

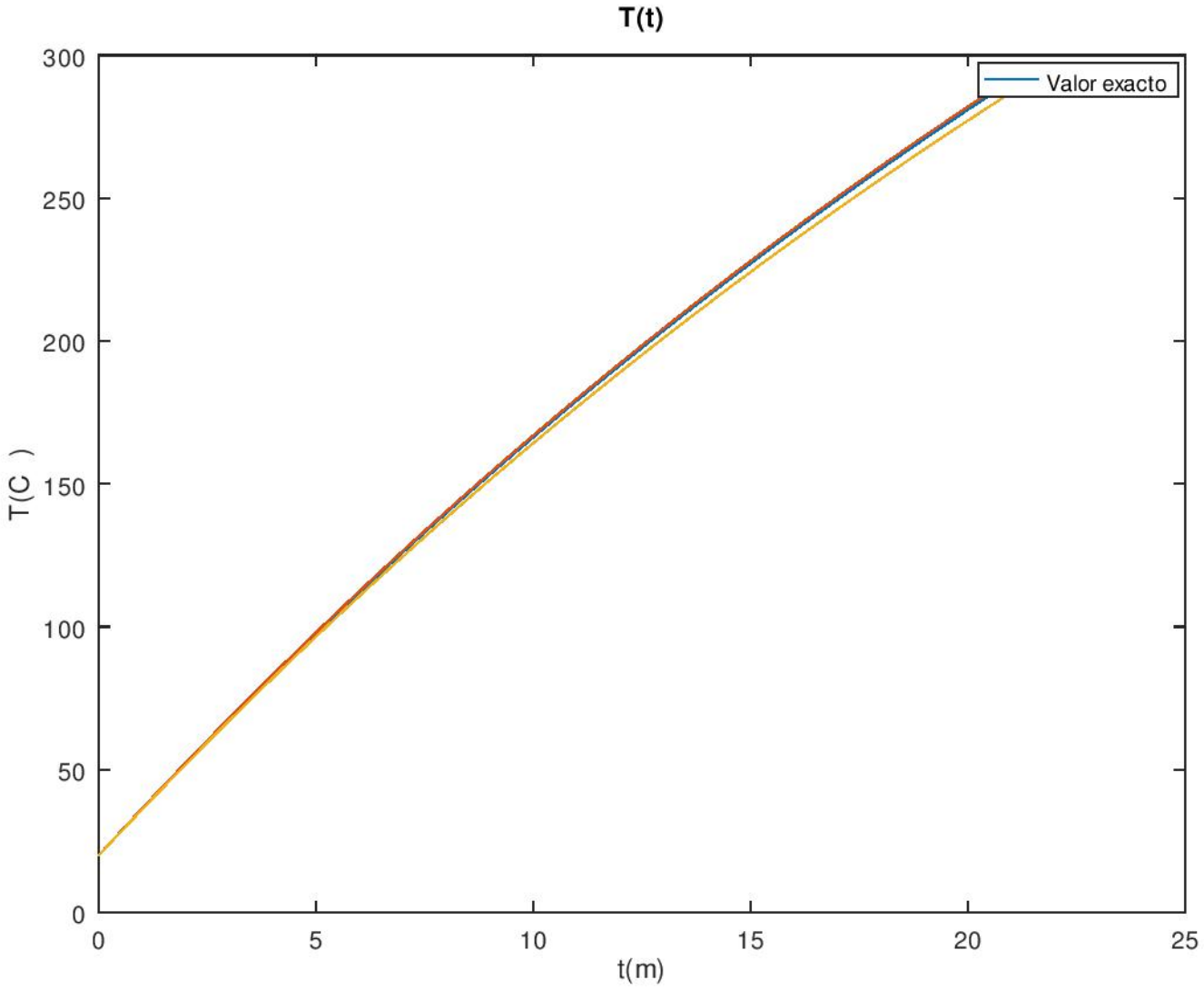


Figura 4: Métodos

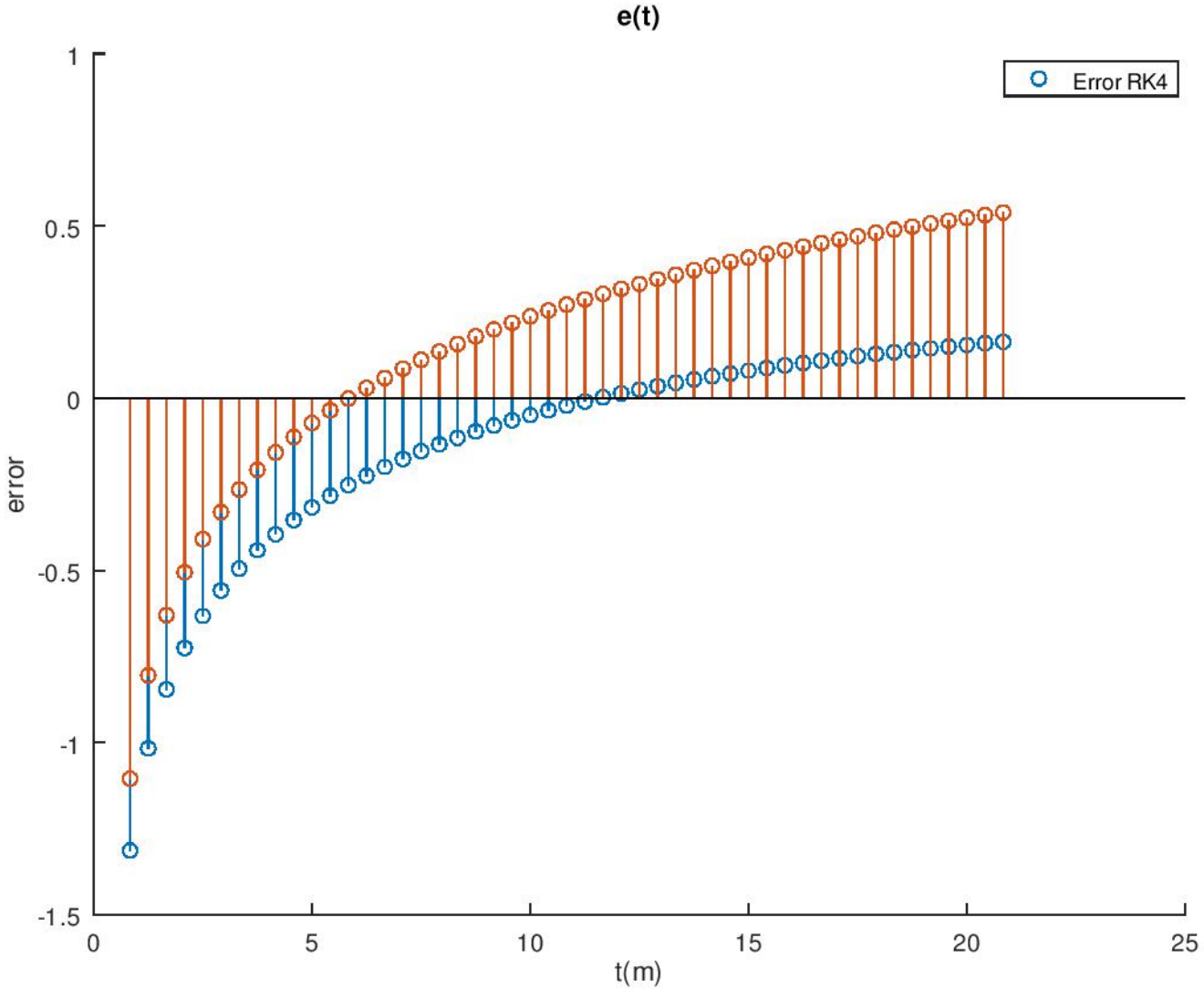


Figura 5: Errores

Utilizando el metodo de RK4 se llego a la siguiente grafica comparativa para el calor por conveccion y radiacion:

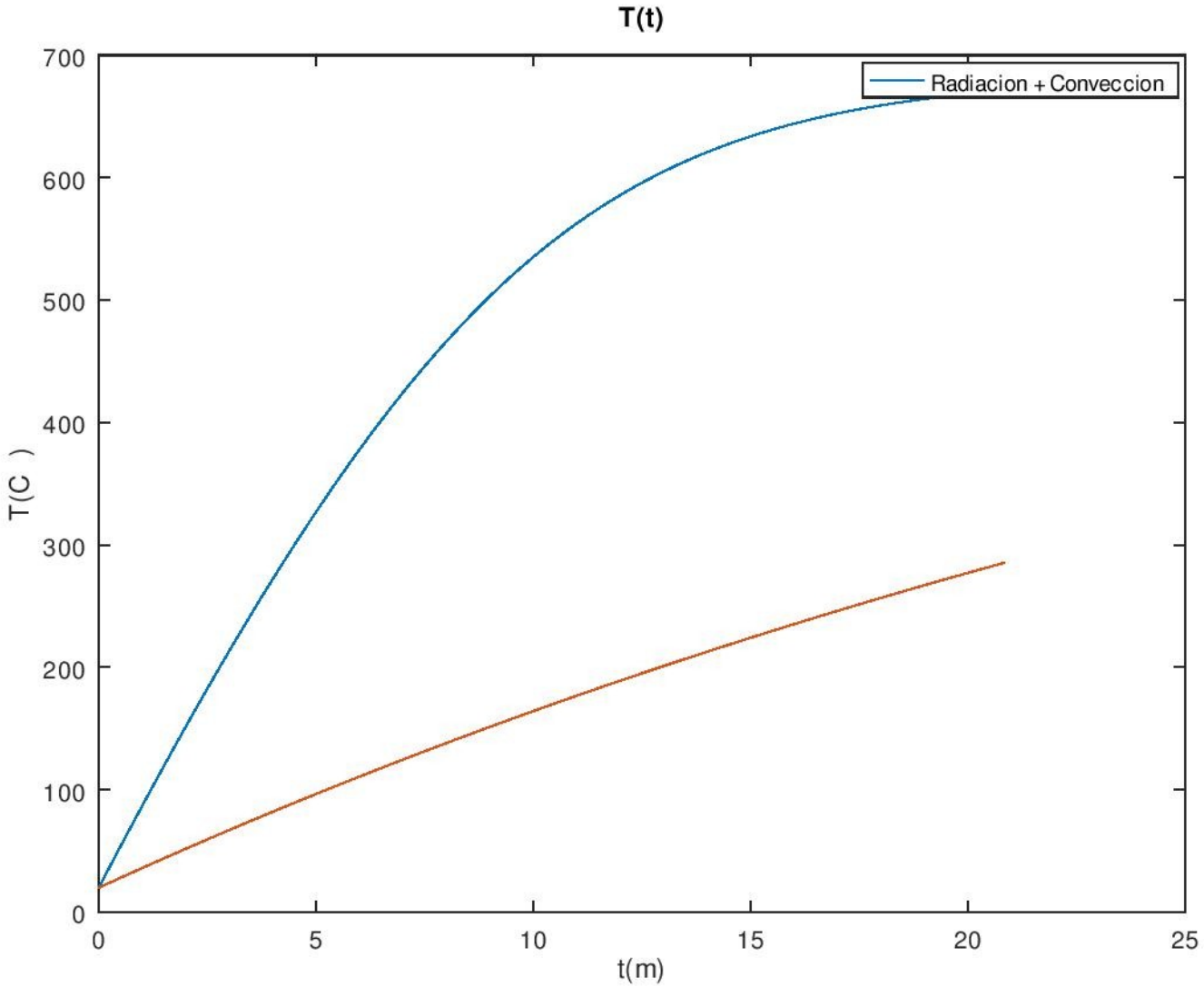


Figura 6: Comparación

Se obtuvieron:

tiempo_soaking = 2.0833 temp_soaking = 667.45

Y mediante el tanteo se aumento el tiempo de soaking a 10 minutos con los valores de temperatura:

T1 = 780 T2 = 680

Finalmente la tabla final de valores con el SENL:

Caso	sk	Tsk	T1	T2	Iteraciones
A	10min	667°C	777.41°C	684.22°C	13
B	10min	648°C	758.84°C	665.49°C	13
C	10min	698°C	792.59°C	714.23°C	13

4. Conclusiones

Las ecuaciones diferenciales resueltas por metodos numericos si bien tienen su grado de error pueden solucionar problemas incalculables. Esto es extremadamente util para casos como este donde hay ecuaciones que no son resolvibles pero un pequeño grado de error no es lo suficientemente significativo entonces los metodos como Euler o RK4 pueden optimizar el tiempo requerido para el calculo y resolucion de un problema. Se pudo observar que RK4 es un algoritmo con mayor precision pero tambien un algoritmo de mayor complejidad que Euler. En la teoria, como RK4 es un algoritmo de orden mayor de error que Euler era esperable que su precision fuese mayor. Resulta extraño que el metodo de punto fijo no haya podido converger pero haya logrado dar una precision aceptable. La condicion de corte del metodo fue reducida a una cifra decimal menos debido a esto y logro converger en todos los casos propuestos.

Hablando mas especificamente respecto de este trabajo practico se puede concluir en que el calor por radiacion no es para nada despreciable en comparacion al de conveccion como fue observado graficamente.

Los metodos numericos son extremadamente utiles para cuando existen ecuaciones diferenciales no lineales pues son mucho mas faciles de resolver y con errores pequeños. Para las ecuaciones diferenciales lineales el error cometido puede incluso ser lo suficientemente pequeño pero teniendo la solucion exacta en general sería mejor obtener dicho valor siempre y cuando no sea demasiado costoso.

En tanto a los metodos de resolucion para SENL es evidente que es mucho más sencillo resolverlos con dichos metodos si existen multiples variables. Se pudo observar en el item 3 del trabajo practico durante el tanteo que a mi particularmente me tomo al menos unos 15 intentos hasta llegar a un resultado que siquiera llega a la precision del metodo de punto fijo. La cantidad de iteraciones del metodo para los 3 casos fueron siempre 13, no solo es mas rapido sino que tambien mas preciso. Para este tipo de problemas inversos resulta ser muy util.

Se podría aumentar la cantidad de iteraciones del metodo de punto fijo para obtener un error aun menor y acercarse aun más a la realidad, pero la relacion con respecto a su costo puede no ser rentable. La velocidad de convergencia de punto fijo puede ser lo suficientemente costosa como para optar por una menor precision con una menor cantidad de recursos.

A. Anexo I: Código Fuente

El programa utilizado es GNU Octave, versión 4.2.2 y se procuró utilizar sintaxis compatible con Matlab, teniendo como única excepción la función `fplot` que brinda Octave para graficar funciones, despues de consultar con los docentes.

main.m

```
function test = main
  padron = 99779;
  diametro_externo = 244.48 *10^-3;
  global DENSIDAD = 7850;
  global PI = 3.14159265359;
  global CALOR_ESPECIFICO = 480;
  global LARGO_TUBO = 12;
  global LARGO_HORNO = 50;
  global NBOL = 50;
  temperatura1 = round( 200 / 10000 * (padron - 90000) + 500 );
  temperatura2 = round( 200 / 10000 * (padron - 90000) + 500 );
  temperatura1 = temperatura1 + 273;
  temperatura2 = temperatura2 + 273;
  global HC = 20;
  global SIGMA = 5.6703*10^-8;
  global EPSILON = 0.85;
  global t0 = 293;
  global S = diametro_externo * LARGO_TUBO * PI
  cadencia = round( -10 / 10000 * (padron - 90000) + 35 );
  diametro_interno = 0.01384;
  velocidad0 = LARGO_HORNO / (NBOL * cadencia);
  termino1 = DENSIDAD * PI * diametro_externo * diametro_interno * LARGO_TUBO;
  termino2 = 1 - diametro_interno/diametro_externo;
  masa = termino1 * termino2;
  fin = LARGO_HORNO/velocidad0;

  tiempo = 0:cadencia:LARGO_HORNO/velocidad0;

  # Calculo de ED
  exactos = metodo_exacto(tiempo, masa, temperatura1);
  euler = metodo_euler(velocidad0, cadencia, masa, temperatura1);
  rk = conveccion_rk(velocidad0, cadencia, masa, temperatura1);

  # Grafico de errores y metodos Euler / RK4
  plotear_temps(tiempo, euler, rk, exactos);
  plotear_errores(tiempo, euler, rk, exactos);

  # Comparacion radiacion y conveccion
  r_rk4 = conveccion_radacion_rk(fin, cadencia, masa, temperatura1, t0);
  r_rk4 = r_rk4.-273;
  plotear_comparacion(tiempo, r_rk4, rk)
```

```
# Soaking
sk = calcular_indice_soaking(r_rk4);
tiempo_soaking = (fin - tiempo(sk)) / 60;
temp_soaking = calcular_temp_soaking(r_rk4, sk);
printf("El␣tiempo␣de␣soaking␣para␣T1␣=␣%d␣°C␣y␣T2␣=␣%d␣°C␣es:\n%d␣minutos\n\n",
    ↪ temperatura1, temperatura2, tiempo_soaking);
printf("Su␣temperatura␣de␣soaking␣es:\n%d␣°C\n\n", temp_soaking);

# Tanteo
t1 = 780;
t2 = 680;
rk = rk_dividido(fin, cadencia, masa, t1+273, t2+273, t0);
rk = rk.-273;
sk = calcular_indice_soaking(rk);
temp_soaking = calcular_temp_soaking(rk, sk);
tiempo_soaking = (fin - tiempo(sk)) /60;
printf("Se␣tantearon␣valores␣hasta␣llegar␣a␣T1␣=␣%d␣y␣T2␣=␣%d␣para␣conseguir␣un␣
    ↪ tiempo␣de␣soaking␣de␣10␣minutos\n", t1, t2);

# Calculo automatico
jacobiano = inv([0.75, 0.25 ; 0.25 , 0.75 ]);
tsk_obj1 = round( 100 / 10000 * (padron - 90000) + 550 );
tsk_obj2 = round( 100 / 10000 * (padron - 90000) + 600 );
tisk_obj = 10;
tempsA = obtener_temps(fin, cadencia, masa, jacobiano, [667.45;667.45], tisk_obj,
    ↪ 667.45, tiempo);
tempsB = obtener_temps(fin, cadencia, masa, jacobiano, [tsk_obj1;tsk_obj1],
    ↪ tisk_obj, tsk_obj1, tiempo);
tempsC = obtener_temps(fin, cadencia, masa, jacobiano, [tsk_obj2;tsk_obj2],
    ↪ tisk_obj, tsk_obj2, tiempo);
rkA = rk_dividido(fin, cadencia, masa, tempsA(1)+273, tempsA(2)+273, t0);
rkB = rk_dividido(fin, cadencia, masa, tempsB(1)+273, tempsB(2)+273, t0);
rkC = rk_dividido(fin, cadencia, masa, tempsC(1)+273, tempsC(2)+273, t0);
printf("\nCalculos␣automaticos␣(10␣minutos␣de␣soaking␣para␣todas):\n\n")
printf("Las␣temperaturas␣de␣inicio␣para␣una␣temperatura␣de␣soaking␣de␣%d␣°C␣son:\n
    ↪ ", temp_soaking);
printf("T1␣=␣%d␣°C\nT2␣=␣%d␣°C\n\n", tempsA(1), tempsA(2));
printf("Las␣temperaturas␣de␣inicio␣para␣una␣temperatura␣de␣soaking␣de␣%d␣°C␣son:\n
    ↪ ", tsk_obj1);
printf("T1␣=␣%d␣°C\nT2␣=␣%d␣°C\n\n", tempsB(1), tempsB(2));
printf("Las␣temperaturas␣de␣inicio␣para␣una␣temperatura␣de␣soaking␣de␣%d␣°C␣son:\n
    ↪ ", tsk_obj2);
printf("T1␣=␣%d␣°C\nT2␣=␣%d␣°C\n\n", tempsC(1), tempsC(2));
```

```
  plot(tiempo./60, rkA.-273)
  plot(tiempo./60, rkB.-273)
  plot(tiempo./60, rkC.-273)
  title("T(t)")
  xlabel("t(m)")
  ylabel("T(C)")
endfunction

function temps = obtener_temps(fin, cadencia, masa, jacobiano, v0, tisk_obj, tsk_obj
    ↪ , tiempo)
  err_individual = 1;
  i = 0;
  while err_individual > 0.5*10^-2;
    v1 = punto_fijo(fin, cadencia, masa, jacobiano, v0, tisk_obj, tsk_obj, tiempo);
    res_err = [(v1(1) - v0(1)) / v1(1); (v1(2) - v0(2)) / v1(2)];
    err_individual = norm(res_err, "inf");
    v0 = v1;
    i+=1;
    if (i == 200)
      break
    endif
  endwhile
  i = i;
  temps = v1;
endfunction

function temps = punto_fijo(fin, h, masa, jacobiano, temperaturas, tiemsk, tempsk,
    ↪ tiempo)
  rk = rk_dividido(fin, h, masa, temperaturas(1) + 273, temperaturas(2) + 273);
  rk = rk.-273;
  sk = calcular_indice_soaking(rk);
  temp_soaking = calcular_temp_soaking(rk, sk);
  tiempo_soaking = (fin - tiempo(sk)) / 60;
  v = [tiempo_soaking - tiemsk; temp_soaking - tempsk];
  temps = temperaturas - jacobiano * v;
endfunction

function rk = rk_dividido(fin, h, masa, t1, t2)
  v = conveccion_radacion_rk(fin/2, h, masa, t1, 273);
  len = columns(v);
  v2 = conveccion_radacion_rk(fin/2, h, masa, t2, v(len));
  for i = 2:len
```

```
    v = [v v2(i)];
  endfor
  rk = v;
endfunction

function void = plotear_comparacion(tiempo, r_rk4, rk)
  plot(tiempo./60, r_rk4)
  hold on
  title("T(t)")
  xlabel("t(m)")
  ylabel("T(C)")
  legend("Radiacion␣+␣Conveccion")
  plot(tiempo./60, rk.-273)
  hold off
endfunction

function indice_soaking = calcular_indice_soaking(r_rk4)
  len = columns(r_rk4);
  sk = 0;
  for i = 1:columns(r_rk4);
        if ((r_rk4(len) - r_rk4(i)) > 10)
          sk = i+1;
        endif
  endfor
  indice_soaking = sk;
endfunction

function temp_soaking = calcular_temp_soaking(r_rk4, sk)
  temp_soaking = 0;
  for i = sk:columns(r_rk4)
    temp_soaking += r_rk4(i);
  endfor
  temp_soaking = temp_soaking / (columns(r_rk4) - sk + 1);

endfunction

function void = plotear_errores(t, euler, rk, exactos)
  error_euler = calcular_error(exactos, euler);
  error_runge = calcular_error(exactos, rk);
  stem(t ./ 60, log10(error_euler))
  title("e(t)")
  xlabel("t(m)")
  ylabel("error")
```

```
  legend("Error␣Euler")
  hold on
  stem(t ./ 60, log10(error_runge))
  hold off
endfunction

function error = calcular_error(exactos, otro)
  e = [];
  exactos = exactos .- 273;
  otro = otro .- 273;
  err = 0;
  #e = [e err]
  for i = 1:columns(exactos);
    err = (exactos(i) - otro(i)) / exactos(i);
    e = [e err];
  endfor
  error = e;
endfunction

function void = plotear_temps(t, euler, rk, exactas)
  t = t ./ 60;
  ex = plot(t, exactas .- 273);
  title("T(t)")
  xlabel("t(m)")
  ylabel("T(C)")
  legend(ex, "Valor␣exacto")
  hold on
  eu = plot(t, euler .- 273);
  rk = plot(t, rk .-273);
  hold off
endfunction

 function temperaturas = metodo_euler(velocidad0, cadencia, masa, tinf)
  global LARGO_HORNO
  temp = 293;
  fin = LARGO_HORNO/velocidad0;
  v = [];
  for t = cadencia:cadencia:fin;
    v = [v temp];
    temp += cadencia*conveccion_diferencial(temp, masa, tinf);
  endfor
  temperaturas = [v temp];
endfunction
```

```
function cc = conveccion_diferencial(temp, masa, tinf)
  global HC CALOR_ESPECIFICO S
  termino1 = temp - tinf;
  termino2 = -masa * CALOR_ESPECIFICO;
  cc = HC * S * termino1 / termino2;
endfunction

function exactas = metodo_exacto(t, masa, tinf)
  for tiempo = t;
    valor = conveccion_exacta(t,masa, tinf);
  endfor
  exactas = valor;
endfunction

function cc = conveccion_exacta(t, masa, tinf)
  global HC t0 CALOR_ESPECIFICO S
  termino1 = -HC*S*t;
  termino2 = masa * CALOR_ESPECIFICO;
  exponente = termino1/termino2;
  a = exp(termino1/termino2);
  cc = tinf + (t0 - tinf)*exp(exponente);
endfunction

function rk = conveccion_rk(velocidad0, cadencia, masa, tinf)
global LARGO_HORNO
  temp = 293;
  fin = LARGO_HORNO/velocidad0;
  v = [];
  for t = cadencia:cadencia:fin;
    v = [v temp];
    temp += (cadencia/6)*rk4(temp, masa, cadencia, tinf);
  endfor
  rk = [v temp];
endfunction

function k = rk4(temp, masa, h, tinf)
  k1 = conveccion_diferencial(temp, masa, tinf);
  k2 = conveccion_diferencial(temp + h/2, masa, tinf);
  k3 = conveccion_diferencial(temp + h/2, masa, tinf);
  k4 = conveccion_diferencial(temp + h, masa, tinf);
  k = k1 + 2*k2 + 2*k3 + k4;
endfunction
```

```
function rk = conveccion_radacion_rk(fin, cadencia, masa, tinf, t0)
  v = [];
  temp = t0;
  for t = cadencia:cadencia:fin;
    v = [v temp];
    temp += (cadencia/6)*r_rk4(temp, masa, cadencia, tinf);
  endfor
  rk = [v temp];
endfunction

function k = r_rk4(temp, masa, h, tinf)
  k1 = conveccion_radacion(temp, masa, tinf);
  k2 = conveccion_radacion(temp + h/2, masa, tinf);
  k3 = conveccion_radacion(temp + h/2, masa, tinf);
  k4 = conveccion_radacion(temp + h, masa, tinf);
  k = k1 + 2*k2 + 2*k3 + k4;
endfunction

function cc = conveccion_radacion(temp, masa, tinf)
  global SIGMA EPSILON CALOR_ESPECIFICO S
  divisor = - masa * CALOR_ESPECIFICO;
  numerador = SIGMA * EPSILON * S * (temp^4 - tinf^4);
  radiacion = numerador / divisor;
  cc = conveccion_diferencial(temp, masa, tinf) + radiacion;
endfunction
```

B. Anexo II: Resultados Numéricos

Bibliografía

[1] Gonzales, Hernan: *Análisis Numérico, Primer Curso* Buenos Aires: Nueva Librería, 2002.