DATOS PERSONALES		FIRMA
Nombre: DNI:		
Apellidos:		
ESTUDIO	**ASIGNATURA**	**CONVOCATORIA**
MÁSTER UNIVERSITARIO EN INGENIERÍA MATEMÁTICA Y COMPUTACIÓN (PLAN 2016)	4391010002.- MÉTODOS NUMÉRICOS AVANZADOS EN INGENIERÍA	Extraordinaria
FECHA	**MODELO**	**CIUDAD DEL EXAMEN**
10-12/09/2021	Modelo - D	
Etiqueta identificativa		

INSTRUCCIONES GENERALES

1. Ten disponible tu documentación oficial para identificarte, en el caso de que se te solicite.
2. Rellena tus datos personales en todos los espacios fijados para ello y lee atentamente todas las preguntas antes de empezar.
3. Las preguntas se contestarán en la lengua vehicular de esta asignatura.
4. Si tu examen consta de una parte tipo test, indica las respuestas en la plantilla según las características de este.
5. Debes contestar en el documento adjunto, respetando en todo momento el espaciado indicado para cada pregunta. Si este es en formato digital, los márgenes, el interlineado, fuente y tamaño de letra vienen dados por defecto y no deben modificarse. En cualquier caso, asegúrate de que la presentación es suficientemente clara y legible.
6. Entrega toda la documentación relativa al examen, revisando con detenimiento que los archivos o documentos son los correctos. El envío de archivos erróneos o un envío incompleto supondrá una calificación de “no presentado”.
7. Durante el examen y en la corrección por parte del docente, se aplicará el Reglamento de Evaluación Académica de UNIR que regula las consecuencias derivadas de las posibles irregularidades y prácticas académicas incorrectas con relación al plagio y uso inadecuado de materiales y recursos.
8. No se permite la comunicación a lo largo del examen.
9. No se permite el uso de recursos externos en el examen.
10. En caso que se realice en domicilio, se podrá acceder a Internet **exclusivamente** para descargar el enunciado y la plantilla del examen, y cargar el examen completado en la plataforma habilitada para tal efecto.

Código de examen: 180013

11. Para facilitar la transcripción de las expresiones matemáticas, puedes utilizar una cámara de fotos, tu teléfono móvil en modo avión, conectado por cable a tu ordenador, o un escáner para incorporar las imágenes en tu examen. No se permite el uso de correo electrónico, ni de aplicaciones de mensajería (incluyendo Whatsapp Web, Teams, Discord, entre otras), ni servicios es la nube (incluye One Drive, Google Drive, Dropbox, entre otros) para realizar esta acción.

Puntuación

Preguntas

- Puntuación máxima 10.00 puntos

Vas a empezar el examen de Métodos Numéricos Avanzados en Ingeniería.

Responde a las preguntas en el espacio indicado en las páginas 5 y 14.

Encontrarás las preguntas del examen a partir de la página 15.

- Asegúrate de tener todo el material que necesites.
- Echa un vistazo a los tres problemas y comienza por el que prefieras.
- Los tres problemas tienen la misma calificación máxima.
- Todas las respuestas se deben justificar y razonar, incluyendo todos los pasos utilizados en su desarrollo hasta llegar al resultado final.
- Los resultados se deben proporcionar con 6 cifras decimales.

Dispones de 2 horas para realizar el examen. ¡Ánimo y suerte!

1. Pregunta 1 (Responder en 3 caras)

2. Pregunta 2 (Responder en 2 caras)

3. Pregunta 3 (Responder en 3 caras)

Preguntas

1. Considera el sistema de ecuaciones diferenciales

$$\begin{aligned} u''(t) + u(t) + v'(t) &= 3e^t(\cos(t) - \sin(t)), \\ v''(t) - v'(t) + v(t) &= 0, \end{aligned} \qquad t \in [0, 2],$$

sujeto a las condiciones iniciales

$$u(0) = 3, \quad u'(0) = 1, \quad v(0) = 3, \quad v'(0) = 3.$$

a) (*2 puntos*) Transforma el problema de valor inicial en un sistema de ecuaciones diferenciales de primer orden. Escribe una función `PVI.m` que implemente el sistema de ecuaciones diferenciales en Matlab. Copia el código en la hoja de respuestas del examen.

b) (*2.5 puntos*) Resuelve el PVI para el intervalo $t \in [0, 2]$, utilziando 10 subintervalos con el método de Runge-Kutta de orden 4. Representa en una misma figura $u(t)$ en color azul y $v(t)$ en color rojo. Indica los valores de $u(t)$ y $v(t)$ para $t \in \{0, 0.4, 0.8, 1.4, 2\}$.

c) (*2.5 puntos*) Resuelve el PVI para el intervalo $t \in [0, 2]$, utilziando 10 subintervalos con el método de Adams-Bashforth de orden 4. Representa en una misma figura $u(t)$ y $v(t)$ en color verde y azul, respectivamente. Indica los valores de $u(t)$ y $v(t)$ para $t \in \{0, 0.4, 0.8, 1.4, 2\}$.

d) (*3 puntos*) Calcula una estimación del orden de convergencia de ambos métodos. Describe y muestra el proceso que has seguido para obtenerla.

2. Queremos resolver la ecuación no lineal

$$\cos(x) = xe^x - x^2.$$

Utilizaremos el método de Newton y el método NN1, cuya expresión iterativa es la siguiente:

$$\begin{aligned} y_k &= x_k - \frac{f(x_k)}{f'(x_k)}, \\ x_{k+1} &= x_k - \frac{f(x_k)}{\frac{4}{5}f'(x_k) + \frac{1}{5}f'(y_k)}, \end{aligned} \qquad k = 0, 1, 2, \ldots$$

a) (*1.5 puntos*) Representa gráficamente sobre una figura las funciones que hay a cada lado de la igualdad, tomando valores de $x \in [0, 1]$. Indica el valor aproximado en que se cortan ambas funciones.

b) (*2 puntos*) Aplica el método de Newton para obtener la solución de la ecuación no lineal, utilizando como estimación inicial $x_0 = 0$ y como criterio de parada $inc = |x_{k+1} - x_k| + |f(x_{k+1})| < 10^{-9}$ con un máximo de 50 iteraciones. Proporciona la solución de la ecuación, el número de iteraciones, el valor de *inc* obtenido en la última iteración y el valor del ACOC.

c) (*3 puntos*) Implementa en Matlab la función `NN1.m` en la que programes el método NN1. Tendrás que introducir como valores de entrada la ecuación a resolver, la estimación inicial, la tolerancia y el número máximo de iteraciones. Como valores de salida de la función, tendrás que dar la solución de la ecuación, el número de iteraciones, el valor de *inc* y el valor del ACOC. Copia el código completo del programa en la hoja del examen.

d) (*2 puntos*) Aplica el método de NN1 para obtener la solución de la ecuación no lineal, utilizando como estimación inicial $x_0 = 0$ y como criterio de parada $inc = |x_{k+1} - x_k| + |f(x_{k+1})| < 10^{-9}$ con un máximo de 50 iteraciones. Proporciona la solución de la ecuación, el número de iteraciones, el valor de *inc* obtenido en la última iteración y el valor del ACOC.

e) (*1.5 puntos*) Establece una comparativa entre los métodos de Newton y NN1 a partir de los resultados obtenidos.

Código de examen: 180013

3. Sea el problema de contorno unidimensional

$$y''(x) = y'(x) + 2(y(x) - \ln(x))^3 - \frac{1}{x}, \qquad x \in [2, 3],$$

siendo las condiciones de contorno

$$y(2) - 2y'(2) = \ln(2), \qquad y(3) = \frac{1}{3} + \ln(3).$$

Para resolver este problema utilizaremos el método de disparo no lineal con el método de la secante.

a) (*2 puntos*) Plantea el problema de valor inicial en el que alguna de las condiciones depende de t.

b) (*3 puntos*) Determina cuál es la ecuación no lineal $F(t)$ cuya solución tenemos que obtener y la expresión a partir de la cual se actualiza el valor de t_k con el método de la secante.

c) (*5 puntos*) Resuelve el problema de contorno tomando 10 subintervalos equiespaciados en $x \in [2, 3]$ y una tolerancia de 10^{-8}. Como estimación inicial de t utiliza $t_0 = \frac{1}{2}$ y como segunda estimación $t_1 = \frac{1}{4}$. Representa la solución $y(x)$. Indica en una tabla los valores de $y(x)$ para $x \in \{2, 2.5, 3\}$. Copia el código del archivo .m que has necesitado para elaborar para resolver este problema. Incluir el código es imprescindible para puntuar este apartado.

Preguntas

Problema 1:

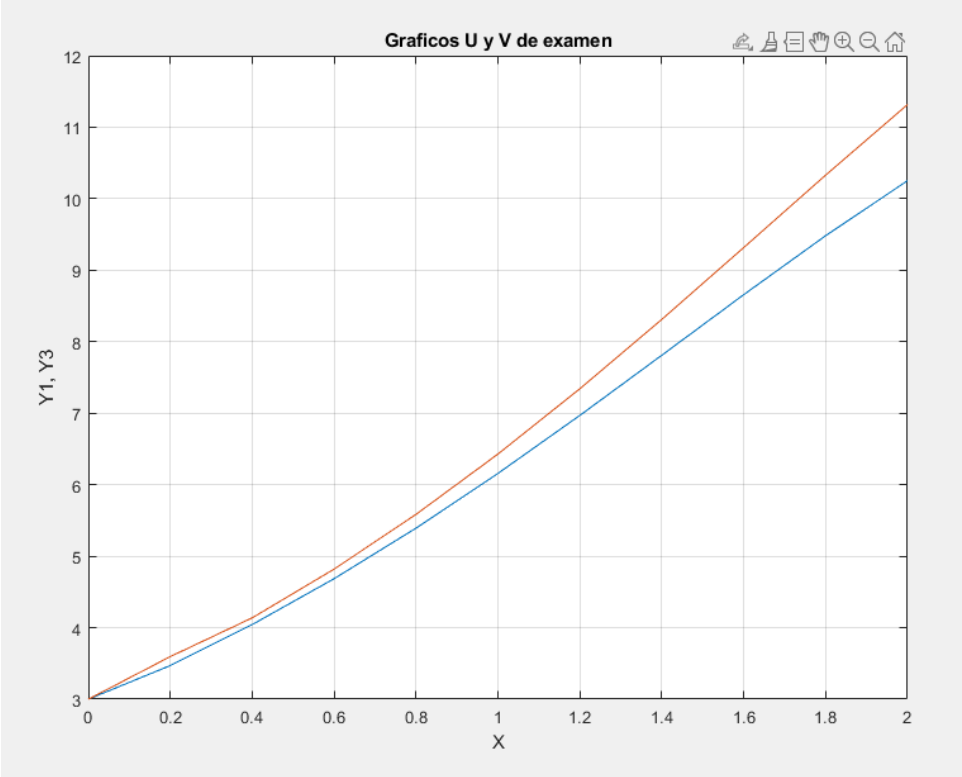
Inciso a)

```
function [du] = PVI(t,u)
    du(1) = u(2);
    du(2) = 3.*exp(t).*(cos(t)-sen(t))-u(1)-u(4);
    du(3) = u(4);
    du(4) = u(4)-u(3);
    du = du(:);
end
```

Inciso b)

Llamando a la funcion rungekutta:

```
[x1,y1] = RungeKutta_sistema('funcion_sistema',0,2,10,[3,1,3,3])
```



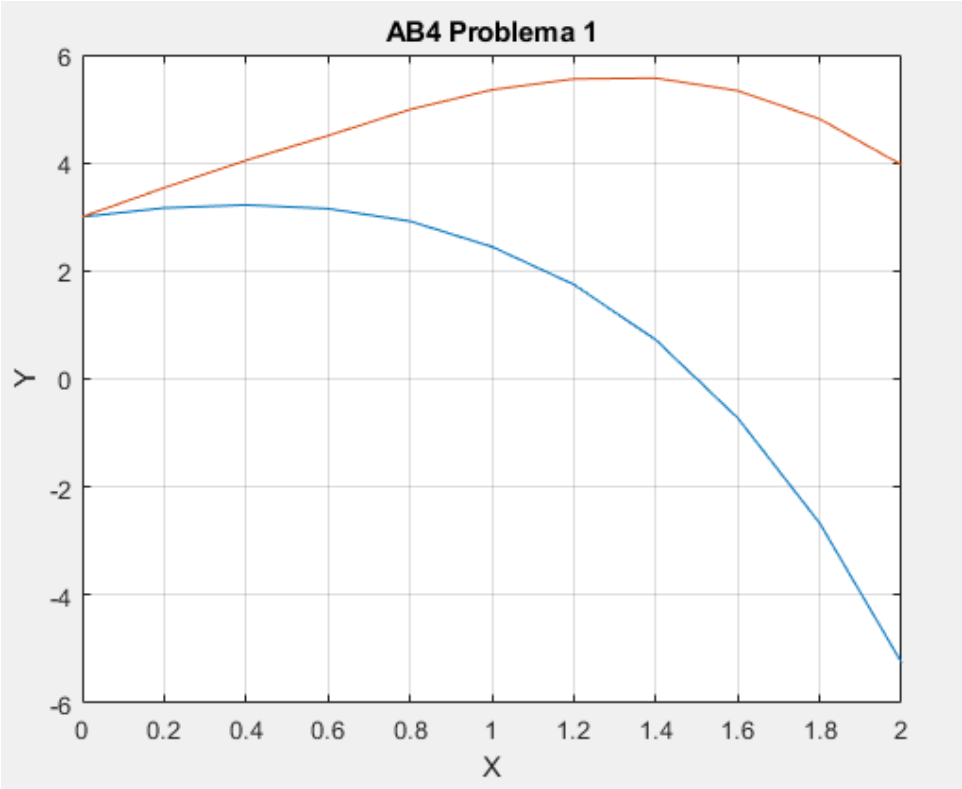
t	U(t)	V(t)
0	3	3

Código de examen: 180013

Preguntas

0.4	4.043992	4.137640
0.8	5. 392659	5.585145
1.4	7. 810634	8.311873
2	10.252846	11.317086

Llamando a la funcion adamsBashforth:



t	U(t)	V(t)
0	3	3
0.4	3.148579	4.164813
0.8	2.545830	5.754962
1.4	0.248134	5.334117
2	-5.129126	2.718531

La estimacion del orden de convergencia se da calculando el doble de iterados entre un vector y el otro.

```
[x1,y1] = RungeKutta_sistema('funcion_sistema',0,2,10,[3,1,3,3])
[x2,y2] = RungeKutta_sistema('funcion_sistema',0,2,20,[3,1,3,3])
[x3,y3] = RungeKutta_sistema('funcion_sistema',0,2,40,[3,1,3,3])


error21 = abs(y2(1:2:end,:)-y1);
error23 = abs(y3(1:2:end,:)-y2);
```

Código de examen: 180013

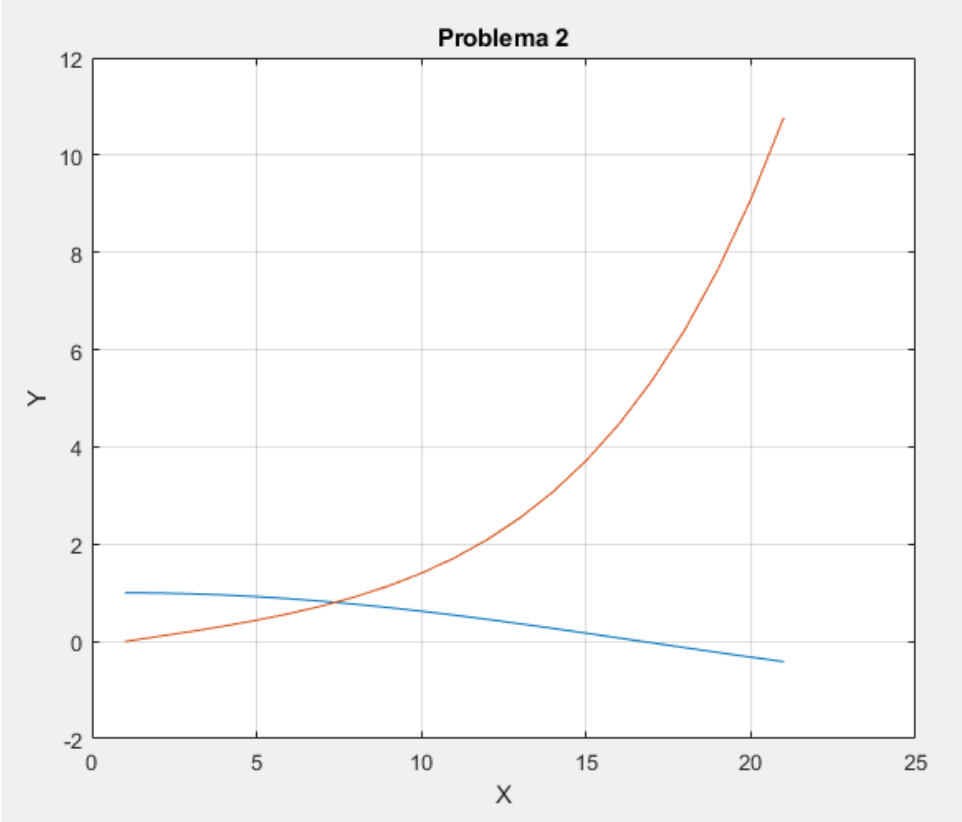
Preguntas

log2(max(error21)/max(error23))

Con tan pocos iterados, el orden de convergencia es poco preciso, sin embargo si se aumenta mucho la cantidad de iterados, puede dar una tendencia cercana a 4.

```
Orden_convergencia =    3.115997206348482
```

Problema 2:



Funcion empleada:
Funcion:

```
function [f,df] = fun1(x)
    f = x.*exp(x)-x.^2-cos(x);
    df = exp(x) - 2*x + sin(x) + x*exp(x);
end
```

Metodo Newton:

```
Digits = 200
[sol,iter,incre1,incre2,acoc] = Newton('fun1', vpa(0), 1e-9, 50)
```

Solucion: 0.6391540
Iteraciones: 6
Incre1: 4.4104e-10
Incre2: 3.6992e-19

Código de examen: 180013

Preguntas

Acoc: [0.95853, 2.1684, 2.0306, 2.0011]

Codigo metodo NN1:

```
function [sol,iter,incre1,incre2,acoc] = NN1(f,x0,tol,maxiter)
% [sol,iter,incre1,incre2,acoc] = Newton('fun1', .5, 1e-12, 40)

% digits 200
% [sol,iter,incre1,incre2,acoc] = MI_Newton('fun1', vpa(.5), 1e-12, 40)

   incre1 = tol+1;
   incre2 = incre1;
   iter = 0;

   %while incre1 > tol && incre2 > tol && iter < maxiter
   while incre1 + incre2 > tol && iter < maxiter
   %while incre2 > tol && iter < maxiter
      [fx,dfx] = feval(f,x0);
      y = x0 - fx/dfx;
      [fy, dfy] = feval(f,y);
      x1 = x0 - fx./((4/5).*dfx + (1/5).*dfy);

      %Actualizar variables
      incre1 = abs(x1-x0);

      fx1 = feval(f,x1);
      incre2 = abs(fx1);

      iter = iter + 1;
      I(iter) = incre1;
      x0 = x1;
   end

   if iter >= maxiter
      disp('Se necesitan mas iterados, no converge con este numero maximo de iteraciones');
   else
      sol = x0;
      disp('¡Bien Hecho!');
   end

   acoc = log(I(3:end)./I(2:end-1))./log(I(2:end-1)./I(1:end-2));

   acoc = vpa(acoc,5);
   incre1 = vpa(incre1,5);
   incre2 = vpa(incre2,5);
end
```

Preguntas

```
%while increl > tol && incre2 > tol && iter < maxiter
while increl + incre2 > tol && iter < maxiter
%while incre2 > tol && iter < maxiter
    [fx,dfx] = feval(f,x0);
    y = x0 - fx/dfx;
    [fy, dfy] = feval(f,y);
    x1 = x0 - fx./((4/5).*dfx + (1/5).*dfy);

    %Actualizar variables
    increl = abs(x1-x0);

    fx1 = feval(f,x1);
    incre2 = abs(fx1);

    iter = iter + 1;
    I(iter) = increl;
    x0 = x1;
end
```

Metodo NN1:
Digits 200
[sol,iter,incre1,incre2,acoc] = NN1('fun1', vpa(0), 1e-9, 50)

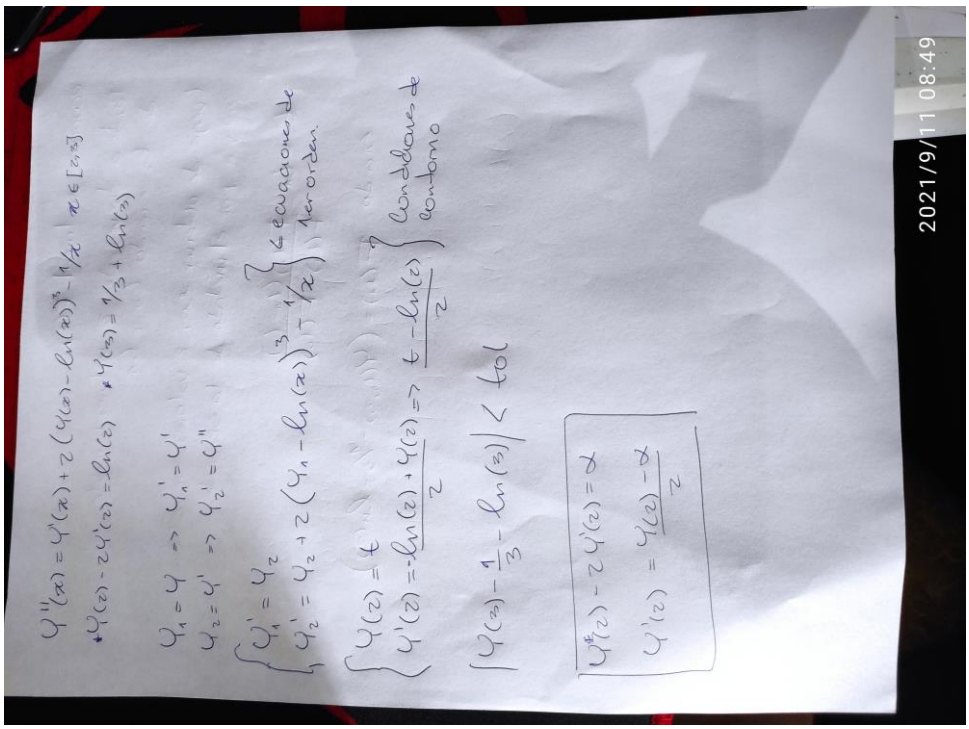
Solucion: 0.639154
Iteraciones: 5
Incre1: 1.0928e-14
Incre2: 1.3626e-28
Acoc: [1.4639, 1.9867, 2.0001]

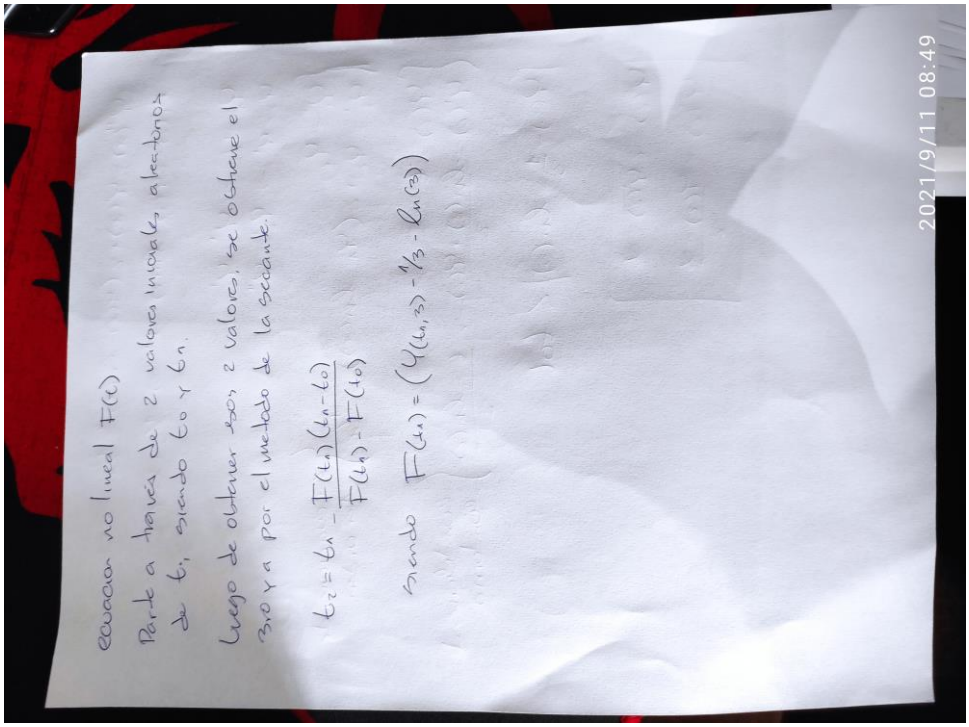
Comparando:
Ambos metodos tienden al mismo orden de convergencia segun el acoc, pero el metodo NN1 ha realizado el proceso con una iteracion menos, por lo que parece ser un poco mas eficiente en calculo. Sin embargo, el metodo NN1 tiene 2 derivadas en el denominador, lo que lo hace un poco mas complejo.

Problema 3:

(Perdon que la imagen salio torcida, si la giro se desconfigura y no puedo tomar otra por falta de tiempo, de verdad disculpas)

Preguntas






```
function [puntos, solaprox, t, iter] = disparosecante_nodirichlet(funcion, a, b, alfa, beta, N, tol, maxiter)
% [puntos, solaprox, t, iter] = disparosecante_nodirichlet('sistema', 2, 3, 1/4, 1/3+log(3), 30, 1e-7, 50)

    h = (b-a)/(N+1);
    x = a:h:b;
```

Código de examen: 180013

Preguntas

```
    t0=0;
    [x,y] = ode45(funcion, x, [alfa, t0]');
    yb0 = y(end, 1);

    t1 = (beta - alfa)/(b-a); %t1=1;
    [x, y] = ode45(funcion, x, [t1, (t1-alfa)/2]');
    yb1 = y(end,1);

    iter = 1;
    incre = tol+1;

    while incre>tol && iter<maxiter
        %Sirve para encontrar los ceros de una funcion de forma iterativa
        t = t1 - (t1-t0)*(ypb1-beta)/(ypb1-ypb0); %t = t1 - (t1-t0)*(yb1-beta)/(yb1-yb0);
        [x, y] = ode45(funcion, x, [t, (t- alfa)/2]');
        incre = abs(y(end, 1) - beta);

        iter = iter+1;
        t0 = t1;
        t1 = t;
        yb0=yb1;
        yb1=y(end,1);
    end

    if incre<tol
        puntos = x;
        solaprox = y;
    else
        disp('se necesitan mas iteraciones')
    end
end
```

Preguntas

```
function [puntos, solaprox, t, iter] = disparosecante_nodirichlet(funcion, a, b, alfa, beta, N, tol, maxiter)
% [puntos, solaprox, t, iter] = disparosecante_nodirichlet('sistema', 2, 3, 1/4, 1/3+log(3), 30, 1e-7, 50)

    h = (b-a)/(N+1);
    x = a:h:b;

    t0=0;
    [x,y] = ode45(funcion, x, [alfa, t0]');
    yb0 = y(end, 1);

    t1 = (beta - alfa)/(b-a); %t1=1;
    [x, y] = ode45(funcion, x, [t1, (t1-alfa)/2]');
    yb1 = y(end,1);

    iter = 1;
    incre = tol+1;

    while incre>tol && iter<maxiter
        %Sirve para encontrar los ceros de una funcion de forma iterativa
        t = t1 - (t1-t0)*(yb1-beta)/(yb1-yb0); %t = t1 - (t1-t0)*(yb1-beta)/(yb1-yb0);
        [x, y] = ode45(funcion, x, [t, (t- alfa)/2]');
        incre = abs(y(end, 1) - beta);

        iter = iter+1;
        t0 = t1;
        t1 = t;
        yb0=yb1;
        yb1=y(end,1);
    end

    if incre<tol
        puntos = x;
        solaprox = y;
    else
        disp('se necesitan mas iteraciones')
    end
end
```

Código de examen: 180013