

SEMINARIOS REALIZADOS EN AULAS DE INFORMA´TICA CON OCTAVE/MATLAB

***ME´TODOS MATEMA´TICOS APLICADOS A LA INGENIERI´A***

Asignaturas:

Me´todos matema´ticos aplicados a la Ing. de Materiales

Me´todos nume´ricos en el Ma´ster en Ing. Industrial

A. I. Mun˜oz Montalvo Septiembre 2022

c 2022. Autora:

⃝

A. I. Mun˜oz Montalvo. Algunos derechos reservados.

Este documento se distribuye bajo la licencia internacional Creative Commons Attribution-ShareAlike 4.0 International License.

Disponible en: <http://creativecommons.org/licenses/by-sa/4.0/>

Publicado en: https://burjcdigital.urjc.es

<http://hdl.handle.net/10115/20113>

# EJERCICIOS PROPUESTOS Y SOLUCIONES

**Observaciones:** Toda l´ınea que comience por % indicar´a un comentario y lo que siga al s´ımbolo prompt del sistema, *>*, es una l´ınea de comandos.

Los c´odigos utilizados han sido creados por los profesores A.I. Mun˜oz, A. Nolla y E. Schiavi y est´an publicados en https://burjcdigital.urjc.es, siendo la mayor parte de ellos, adaptaciones de las funciones del libro *“C´alculo cient´ıfico con MATLAB y Octave”* de A. Quarteroni, F. Saleri, que se pueden obtener en https://mox.polimi.it/qs/.

## SEMINARIO 1.

1. Sea dada la funci´on

*f* (*x*) = *x*3 + 4*x*2 − 10*,*

* 1. Aplicar el algoritmo de bisecci´on para calcular todas las soluciones de la ecuaci´on *f* (*x*) = 0, trabajando con una tolerancia de 10*−*3 y un nu´mero m´aximo de 100 iteraciones.
  2. Aplicar el m´etodo de Newton para resolver la misma ecuaci´on con los mismos valores de tolerancia y nu´mero m´aximo de iteraciones.

**Soluci´on.** Escribimos en un script el siguiente texto:

% Apartado a)

*>* fecu=@(x) x.^ 3+4.\*x.^ 2-10;

* a=0;b=2;maxitera=100;errorper=1e-3;

% Comprobamos que hemos elegido bien los extremos del intervalo:

* fecu(a); fecu(b)

% y vemos que toman signos distintos, -10 y 14.

% Hacemos la llamada al c´odigo que resuelve el problema por bisecci´on,

% metbiseccion.m:

* [sol,itera]=metbiseccion(fecu,a,b,errorper,maxitera);

% Apartado b). A continuaci´on resolvemos el problema por el m´etodo de Newton-Raphson:

% s´olo queda por deﬁnir:

* dfecu=@(x) 3.\*x.^ 2+8.\*x; x0=1;

% y llamamos al c´odigo metnewton1ecu.m

* [sol2,itera2]=metnewton1ecu(fecu,dfecu,x0,...
* errorper,maxitera)

%Fin del script.

Las soluciones obtenidas son: sol=1.3643, itera=10, con el m´etodo de bisecci´on y sol2=1.352, itera2=4, con el m´etodo de Newton.

1. Sea dada la funci´on

*f* (*x*) = *ex* − 3*x*2*.*

* 1. Considerando el esquema num´erico asociado al m´etodo de Newton como un esquema de punto ﬁjo, deﬁnir la funci´on *g*(*x*) asociada al m´etodo y utilizar el algoritmo de punto ﬁjo para calcular las solu- ciones de la ecuaci´on *f* (*x*) = 0, trabajando con una tolerancia de 10*−*6 y un nu´mero m´aximo de 1000 iteraciones.

**Soluci´on.** Escribimos en un script el siguiente texto:

* f=@(x) exp(x)-3.\*x.^ 2;

% Comprobamos que en el intervalo [0,1] hay una soluci´on

* a=0;b=1; fun(a),fun(b)

% fun(a)=1, fun(b)=-0.2817

% En efecto, la funci´on toma signos distintos en 0 y en 1: 1 y -0.28172, resp.

%Deﬁnimos g(x)=x-f/f’

* g=@(x) x-(exp(x)-3.\*x.^ 2).\*(exp(x)-6.\*x).^ (-1);
* x0=0.5;errorper=1e-6;maxitera=1000;

% Hacemos la llamada al c´odigo metpuntoﬁjo.m

* [x,itera]=metpuntofijo(g,x0,errorper,maxitera)

La soluci´on obtenida es: x=0.91001 e itera=5.

1. Resolver el sistema de ecuaciones no lineales:

{ − −

ln(*x*2 + *y*2) sin(*xy*) (ln(2) + ln(*π*)) = 0

*ex−y* + cos(*xy*) = 0*.*

utilizando el m´etodo de Newton y trabajando con una tolerancia de 10*−*6. Utilizar como semilla (*x*0*, y*0) = (2*,* 2).

## Soluci´on.

Primero deﬁnimos los scripts con las funciones fecusistema.m y jacobiana.m:

* function F=fecusistema(x,y)
* F(1,1)=log(x.^ 2+y.^ 2)-sin(x.\*y)-(log(2)+log(pi));
* F(2,1)=exp(x-y)+cos(x.\*y);
* end
* function J=jacobiana(x,y)
* J(1,1)=2.\*x.\*(x.^ 2+y.^ 2).^ (-1)-y.\*cos(x.\*y);
* J(1,2)=2.\*y.\*(x.^ 2+y.^ 2).^ (-1)-x.\*cos(x.\*y);
* J(2,1)=exp(x-y)-y.\*sin(x.\*y);
* J(2,2)=-exp(x-y)-x.\*sin(x.\*y);
* end

A continuaci´on, llamamos al c´odigo metnewtonsistemas.m, para ellos es- cribimos un nuevo script con los comandos:

* vectorx0=[2;2]; errorper=1e-6;maxitera=100;
* [x,res,niter]=metnewtonsistema(@fecusistema,...
* @jacobiana,vectorx0,errorper,maxitera)

La soluci´on obtenida es: x=1.77245, y=1.77245, itera=6.

## SEMINARIO 2.

1. Se considera el problema de valor inicial:

{ −

*y′* = *y* sin(*t*) + cos(t)*, y*(0) = 1*.*

En el intervalo temporal [0*,* 2], utilizar los esquemas num´ericos de Euler expl´ıcito, Euler impl´ıcito y Crank-Nicolson para obtener el valor aproxi- mado de la soluci´on en *t* = 1 tomando un paso de discretizaci´on *h* = 0*.*1. Sabiendo que la soluci´on exacta es *y*(*t*) = *et* + *sin*(*t*), dibujar las solu- ciones obtenidas con los tres m´etodos y la soluci´on exacta en un mismo plot. Comparar los resultados obtenidos. Escribir los valores de las dis- tintas soluciones en *t* = 0*.*5 y *t* = 1*.*5. .

**Soluci´on.** Abrimos un nuevo script para escribir el siguiente c´odigo y luego ejecutarlo:

* f=@(t,y) y-sin(t)+cos(t);
* intiempo=[0 2]; valorini=1;npasos=20;

% Hacemos una llamada a los c´odigos correspondientes:

* [solt,solye]=eulerexpicito(f,intiempo,valorini,npasos);
* [solt,solyi]=eulerimplicito(f,intiempo,valorini,npasos);
* [solt,solyc]=cranknicolson(f,intiempo,valorini,npasos);

% Observa que llamamos de forma distinta a los resultados para y.

* solexac=@(t) exp(t)+sin(t); figure;

% Dibujamos las soluciones junto con los valores de la soluci´on exacta para comparar

* plot(solt,solye,’ro’,solt,solyi,’b+’,...
* solt,solyc,’g\*’,solt,solexac(solt),’k^ )
* t1=min(find(soluciont>=0.5));

% tambi´en t1=ﬁnd(soluciont==0.5);

* t2=min(find(soluciont>=1.5));

% tambi´en t2=ﬁnd(soluci´on t==1.5);

* u1(t1),u2(t1),u3(t1), u1(t2),u2(t2),u3(t2)

Las soluciones son: para t=0.5, con EE 2.096, con EI 2.1646 y con CN 2.1283. Para t=1.5, con EE 5.2516, con EI 5.7585 y con CN, 5.4825.

10

8

6

4

2

0

0 0.5 1 1.5 2

Figure 1: Resultados obtenidos con los 3 m´etodos junto con la exacta (en c´ırculos rojos EE, en + azules EI, en \* verdes CN, y la exacta en tri´angulos negros).

1. Se considera el problema de valor inicial:

{ −

*y′* = 2*ty*2*, y*(0) = 1*.*

Aplicar los m´etodos tipo Runge-Kutta dados por los algoritmos de Heun (heun.m) y Simpson (rungekuttao3.m), para resolver el PVI anterior en el intervalo temporal [0,1], tomando como paso de discretizaci´on *h* = 0*.*01.

Escribir los valores obtenidos con los dos m´etodos para los tiempos *t* = 0*.*1,

*t* = 0*.*5 y *t* = 1.

**Soluci´on**. Abrimos un script con el siguiente c´odigo :

* f=@(t,y) -2.\*t.\*y.^ 2; intiempo=[0 1];
* npasos=100;valorini=1;
* [solt,soly1]=heun(f,intiempo,valorin,npasos);
* [solt,soly2]=rungekuttao3(f,intiempo,valorin,npasos);

% Dibujamos los resultados obtenidos con los dos m´etodos:

* plot(solt,soly1,’ro’,solt,soly2,’b\*’)

% Calculamos los valores num´ericos en los tiempos pedidos:

*>* t1=min(find(tt>=0.1));

*>* t2= min(find(tt>=0.5));

* t3=min(find(tt>=1));
* u2(t1),u2(t2),u2(t3) u3(t1),u3(t2),u3(t3)

Los resultados obtenidos con Heun son: en t=0.1, 0.9954, en t=0.5, 0.8518, en t=1, 0.5174. Los resultados obtenidos con Simpson son: en t=0.1, 0.9901, en t=0.5, 0.8006, en t=1, 0.5014.

## SEMINARIO 3.

1. Sea dado el Problema de Transporte Difusivo y Convectivo deﬁnido por el PVC:

*u′′* 4*u′* = 16*x*3 + 34*x* 1*, x* (0*,* 2)*, u*(0) = 4*, u*(2) = 2*,*

{ − − − − ∀ ∈

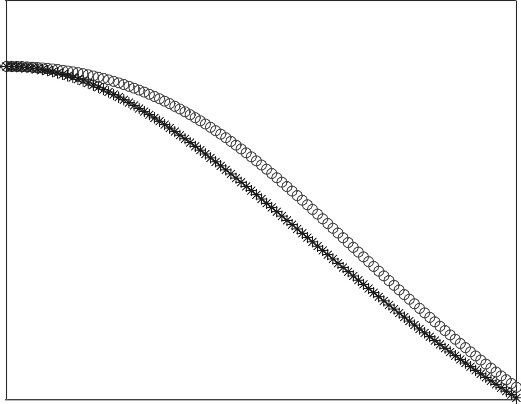
cuya soluci´on anal´ıtica es:

*sol*(*x*) = *x*4 − *x*3 − 3*.*5*x*2 + 2*x* + 4*,* siendo u la concentraci´on de un contaminante.

* 1. Aplicar el algoritmo bvpdirichlet.m para calcular la soluci´on en el

intervalo [0, 2] con paso de discretizaci´on *h* = 0*.*125.

1.1



1

0.9

0.8

0.7

0.6

0.5

0 0.2 0.4 0.6 0.8 1

Figure 2: Resultados obtenidos con el m´etodo de Heun en rojo y con Simpson en azul (queda por encima la soluci´on obtenida con Heun).

* 1. Dibujar la soluci´on anal´ıtica junto con la soluci´on num´erica.
  2. Determinar los valores m´aximos y m´ınimos de contaminaci´on as´ı como su localizaci´on utilizando la soluci´on num´erica y la anal´ıtica. Calcular los errores cometidos.
  3. Determinar la regi´on m´axima de seguridad (donde la concentraci´on del contaminante es menor que uno, es decir, *u*(*x*) *<* 1). Utilizar la soluci´on anal´ıtica.

**Soluci´on** Creamos un script con el siguiente c´odigo para luego ejecutarlo:

* a=0; b=2; D=1; V=-4; Q=0;
* f=@(x) -16.\*x.^ 3+34.\*x-1; ua=4; ub=2; N=15;
* [xh,uh]=bvpdirichlet(a,b,N,D,V,Q,f,ua,ub);
* solexac=xh.^ 4-xh.^ 3-3.5.\*xh.^ 2+2.\*xh+4;
* figure; plot(xh,uh,’r’,xh,solexac,’g’)
* nmax=max(uh),nmin=min(uh)
* emin=min(solexac),emax=max(solexac)

5

4

3

2

1

0

0 0.5 1 1.5 2

Figure 3: Resultados obtenidos con el m´etodo num´erico en rojo, y la soluci´on exacta en verde.

Los resultados son: nmax=4.3235, nmin=0.7233, emin=0.6897, emax=4.2695.

% Dibujamos el error cometido:

* figure; err=uh-solexac; plot(xh,err)

0.08

0.06

0.04

0.02

0

0 0.5 1 1.5 2

Figure 4: Error cometido.

% Vemos d´onde se alcanzan el valor m´aximo y el m´ınimo

* xm=find(uh>=nmax), xmin=find(uh<=nmin), xh(xm),xh(xmin)

Los valores obtenidos (componentes) son: xm=3 y xmin=14. Por tanto, el valor m´aximo se alcanza en x=0.25, y el valor m´ınimo en x=1.6250.

1. Sea dado el PVIC:



*∂u* = *∂*2 *u* + *f* (*x, t*)*,*

*∂t*

*∂x*2

donde

*u*(*x, t*) = *g*(*x, t*)*, x* = 0*, x* = 1*,*

 *u*(*x,* 0) = *u*0(*x*)*, x* ∈ (0*,* 1)

*f* (*x, t*) = 2*t* − *x*2 + 10*xt, cc*(*x, t*) = *x*2(*x* − 1)*, u*0(*x*) = 0*.*

Aplicar el algoritmo de ecucalor.m para calcular la soluci´on en el intervalo temporal [0,1], con paso de discretizaci´on espacial, ∆*x* = 0*.*05 y de dis- cretizaci´on temporal, ∆*t* = 0*.*02. Dibujar la soluci´on obtenida para *θ* = 0 (Euler Expl´ıcito). ¿Puedes justiﬁcar la gr´aﬁca obtenida ? Determinar el paso de discretizaci´on temporal m´aximo que asegura estabilidad y calcular la soluci´on para ese paso temporal. ¿Cu´anto vale la soluci´on en el punto *x* = 0.85 ?

**Soluci´on.** Abrimos un nuevo script con el siguiente c´odigo:

* intespacio=[0 1]; intiempo=[0 1];
* pasosespacio=20,pasostiempo=100;
* theta=0; c=1; u0=@(x) 0.\*x;
* cc=@(t,x) (x.^ 2).\*(x-1);
* f=@(t,x) 2.\*t-x.^ 2+10.\*x.\*t;
* [x,u]=ecucalor(intespacio,intiempo,pasosespacio,...
* pasostiempo,theta,c,u0,cc,f);
* figure; plot(x,u)

% Se puede observar que el plot no aparece puesto que hay valores

% que son del orden de 1e+11

% Esto se debe a la inestabilidad asociada al m´etodo expl´ıcito

% Para que sea inestable, debemos reducir el paso temporal

*>* x1=min(find(x>=0.85)); u(x1)

% Se tiene que x1=18 y u(18)=3.6504e+11

* dx=0.05; dt=(0.05)^ 2; pasost=1/(0.05)^ 2 ;
* pasosespacio=20; pasosespacio=800;
* [x,u1]=ecucalor(intespacio,intiempo,pasosespacio,pasostiempo,...
* theta,c,u0,cc,f);
* figure;

*>* plot(x,u1), x1=min(find(x>=0.85)), u1(x1)

El resultado obtenido es u1(18)=0.4496.

0.8

0.6

0.4

0.2

0

0 0.2 0.4 0.6 0.8 1

Figure 5: Resultados obtenidos cumpliendo el criterio de estabilidad.

# EJERCICIOS QUE DEBE REALIZAR EL ALUMNO Y ENTREGAR A LA PROFESORA

1. Sea dada la funci´on

*f* (*x*) = *ex* − 15 − *arctg*(*x*)*.*

* 1. Aplicar el algoritmo de bisecci´on para calcular la soluci´on de la ecuaci´on *f* (*x*) = 0, trabajando con una tolerancia de 10*−*3 y un nu´mero m´aximo de 100 iteraciones.
  2. Aplicar el m´etodo de Newton para resolver la misma ecuaci´on con los mismos valores de tolerancia y nu´mero m´aximo de iteraciones.

1. Sea dada la funci´on

*f* (*x*) = √*xsin*(*x*) − *x*3 + 2*.*

* 1. Considerando el esquema num´erico asociado al m´etodo de Newton como un esquema de punto ﬁjo, deﬁnir la funci´on *g*(*x*) asociada al m´etodo y utilizar el algoritmo de punto ﬁjo para calcular la soluci´on de la ecuaci´on *f* (*x*) = 0, trabajando con una tolerancia de 10*−*6 y un nu´mero m´aximo de 1000 iteraciones.
  2. Deﬁne un esquema de punto ﬁjo diferente al considerado en el apartado anterior justiﬁcando la elecci´on, para calcular la soluci´on de la ecuaci´on *f* (*x*) = 0 trabajando con una tolerancia de 10*−*6 y un nu´mero m´aximo de 1000 iteraciones.

1. Resolver el sistema de ecuaciones no lineales:

{

−*x* − *y*3 + *x*2 − 2 = 0*,* 1 − 2*x* + *y* − *y*2 = 0*.*

utilizando el m´etodo de Newton y trabajando con una tolerancia de 10*−*6. Utilizar como semilla (*x*0*, y*0) = (−0*.*8*,* −0*.*4).

1. Se considera el problema de valor inicial:

{ −

*y′* = *ktβy, y*(0) = 1*,*

donde *k* = 4 y *β* = 0*.*25. Resolver el PVI en el intervalo [0*,* 4], utilizando los m´etodos de Euler expl´ıcito y Euler impl´ıcito. Analizar la estabilidad de los dos m´etodos utilizando para ello distintos pasos de discretizaci´on, por ejemplo *h*1 = 0*.*5, *h*2 = 0*.*4, *h*3 = 0*.*2 y *h*4 = 0*.*1.

1. Se considera el problema de valor inicial:

{ *y* = *,*

*′* 2*y*

*t*+1

*y*(0) = 1*.*

En el intervalo temporal [0*,* 1], utilizar los esquemas num´ericos de Euler expl´ıcito, Euler impl´ıcito y Crank-Nicholson para obtener el valor aproxi- mado de la soluci´on en *t* = 1, tomando un paso de discretizaci´on *h* = 0*.*1. Sabiendo que la soluci´on exacta es *y*(*t*) = (*t* + 1)2, dibujar las soluciones obtenidas con los tres m´etodos y la soluci´on exacta en un mismo plot. Comparar los resultados obtenidos. Escribir los valores de las distintas soluciones en *t* = 0*.*5 y *t* = 0*.*75. Encontrar los intervalos temporales para los cuales las distintas soluciones num´ericas alcanzan valores superiores a tres.

1. Se considera el problema de valor inicial:

{ − −

*y′* = *y*(*y* 1)*,*

*y*(0) = 2*.*

Aplicar los m´etodos tipo Runge-Kutta dados por los algoritmos heun.m y rungekuttao3.m, para resolver el PVI anterior en el intervalo temporal [0,1], tomando como paso de discretizaci´on *h* = 0*.*01 Escribir los valores obtenidos con los dos m´etodos para los tiempos *t* = 0*.*1, *t* = 0*.*5 y *t* = 1.

1. Sea dado el Problema de Transporte Difusivo y Convectivo deﬁnido por el PVC:

{ − − ∀ ∈

*u′′* + *u′* = 2*e−x, x* (0*,* 1)*,*

*u*(0) = 0*, u*(1) = 2*cosh*(1)*,*

cuya soluci´on anal´ıtica es:

*sol*(*x*) = *ex* + *e−x*

.

* 1. Aplicar el algoritmo bvpdirichlet.m para calcular la soluci´on en el intervalo [0, 1] con paso de discretizaci´on *h* = 0*.*01.
  2. Dibujar la soluci´on anal´ıtica junto con la soluci´on num´erica.
  3. Determinar los valores m´aximos y m´ınimos de *u* y su localizaci´on, para la soluci´on num´erica y la soluci´on anal´ıtica. Calcular los errores cometidos.

1. Sea dado el PVIC:



*∂u* = *∂*2 *u* + *f* (*x, t*)*,*

*∂t*

*∂x*2

donde

*u*(*x, t*) = *g*(*x, t*)*, x* = 0*, x* = 2*,*

 *u*(*x,* 0) = *u*0(*x*)*, x* ∈ (0*,* 2)*,*

*f* (*x, t*) = *tsin*(*x*)*, g*(*x, t*) = 0*,*

*u*0(*x*) = *x,* para 0 ≤ *x* ≤ 1 y *u*0(*x*) = 2 − *x,* para 1 *< x* ≤ 2*.*

Aplicar el algoritmo de ecucalor.m para calcular la soluci´on en el inter- valo temporal [0,1], con paso de discretizaci´on espacial ∆*x* = 0*.*05 y dis- cretizaci´on temporal, ∆*t* = 0*.*02. Dibujar la soluci´on obtenida para *θ* = 0*.*5 en *t* = 1. ¿Cu´anto vale la soluci´on en el punto *x* = 1 ?