



DEPARTAMENTO DE CIENCIAS BASICAS CICLO III/12

GUIA 4 DE LABORATORIO EN MATLAB PARA METODOS NUMERICOS SOBRE ECUACIONES DIFERENCIALES

Para obtener la solución exacta de ecuaciones diferenciales, Matlab emplea el comando “***dsolve***”. Este comando proporciona soluciones simbólicas de ecuaciones diferenciales ordinarias y además se utiliza para resolver sistemas de ecuaciones diferenciales. Las ecuaciones son especificadas por expresiones simbólicas conteniendo la letra “***D***” para denotar diferenciación de orden 1, o las expresiones: “***D2***”, “***D3***”, ..., etc, para denotar diferenciación de orden 2, 3, ..., etc. A continuación de la letra “***D***” se sitúa la variable dependiente (que suele ser Y) y cualquier letra no precedida por “***D***” es un candidato a variable independiente. Si no se especifica la variable independiente, por defecto Matlab emplea la variable “***t***”.

Se pueden especificar condiciones iniciales en ecuaciones adicionales, mediante la forma $y(a)=b$ ó $Dy(a)=b$, ..., etc. Si las condiciones iniciales no se especifican, las soluciones de las ecuaciones diferenciales contendrán constantes de integración C1, C2, ..., etc. Para obtener el valor numérico exacto se debe auxiliar del comando “***subs***”.

Además se podría generar programas basándose en las ecuaciones que definen los diferentes métodos para resolver aproximadamente los valores de la solución de ecuaciones diferenciales; además Matlab posee comandos de la familia “***ODE***” que nos permiten obtener la solución numérica de ecuaciones diferenciales así como de sistemas de ecuaciones diferenciales.

A continuación se presenta la sintaxis de los comandos:

dsolve('ecuación','v') Resuelve la ecuación diferencial siendo “v” la variable independiente (si no se especifica “v”, la variable independiente es “ t “)

dsolve('ecuación','condición_inicial','v') Resuelve la ecuación diferencial sujeta a la condición inicial especificada

dsolve('ecuación','condición1','condición2',...,'condiciónN','v')
Resuelve la ecuación diferencial sujeta a las condiciones iniciales especificadas.

dsolve('ecuación','condición1,condición2,...,condiciónN','v') Resuelve la ecuación diferencial sujeta a las condiciones iniciales especificadas.

dsolve('ecu1','ecu2',...,'ecuN','cond1','cond2',...,'condN','v') Resuelve el sistema diferencial sujeto a las condiciones iniciales especificadas.

dsolve('ecu1,ecu2,...,ecuN','cond1,cond2,...,condN','v') Resuelve el sistema diferencial sujeto a las condiciones iniciales especificadas.

| Solver | Tipo Del Problema | Orden de la exactitud | Cuándo utilizar |
|---------|----------------------|-----------------------|---|
| ode45 | No Rígido | Medio | La mayoría del tiempo. Éste debe ser el primer solver que usted debe intentar. |
| ode23 | No Rígido | Bajo | Si usa tolerancias crudas del error o para solucionar problemas moderado rígidos. |
| ode113 | No Rígido | Bajo a Alto | Si usa tolerancias rigurosas del error |
| ode15s | Rígido | Bajo a Medio | Si ode45 es lento o hay una matriz total. |
| ode23s | Rígido | Bajo | Si usa tolerancias crudas del error para solucionar sistemas tiesos o allí existe una matriz total constante. |
| ode23t | Moderadamente Rígido | Bajo | Si el problema es solamente moderado rígido |
| ode23tb | Rígido | Bajo | Si usa tolerancias crudas del error para solucionar sistemas rígidos o allí existe una matriz total. |

Solve differential equations

Syntax

[T,Y] = solver('F',tspan,y0)

[T,Y] = solver('F',tspan,y0,options)

[T,Y] = solver('F',tspan,y0,options,p1,p2...)

[T,Y,TE,YE,IE] = solver('F',tspan,y0,options)

| | |
|----------|---|
| F | Name of the ODE file, a MATLAB function of t and y returning a column vector. All solvers can solve systems of equations in the form $y' = F(t, y)$. ode15s, ode23s, ode23t, and ode23tb can solve equations of the form $My' = F(t, y)$. Of these four solvers all but ode23s can solve equations in the form $M(t)y' = F(t, y)$ |
| tspan | A vector specifying the interval of integration [t0 tfinal]. To obtain solutions at specific times (all increasing or all decreasing), use tspan = [t0,t1, ..., tfinal]. |
| y0 | A vector of initial conditions. |
| options | Optional integration argument created using the odeset function. See odeset for details. |
| p1,p2... | Optional parameters to be passed to F. |
| T,Y | Solution matrix Y, where each row corresponds to a time returned in column vector T. |

Description

[T,Y] = solver('F',tspan,y0) with tspan = [t0 tfinal] integrates the system of differential equations $y' = F(t,y)$ from time t0 to tfinal with initial conditions y0. 'F' is a string containing the name of an ODE file. Function F(t,y) must return a column vector. Each row in solution array y corresponds to a time returned in column vector t. To obtain solutions at the specific times t0, t1, . . . , tfinal (all increasing or all decreasing), use tspan = [t0 t1 ... tfinal].

[T,Y] = solver('F',tspan,y0,options) solves as above with default integration parameters replaced by property values specified in options, an argument created with the odeset function (see odeset for details). Commonly used properties include a scalar relative error tolerance RelTol (1e-3 by default) and a vector of absolute error tolerances AbsTol (all components 1e-6 by default).

[T,Y] = solver('F',tspan,y0,options,p1,p2...) solves as above, passing the additional parameters p1,p2... to the M-file F, whenever it is called. Use options = [] as a place holder if no options are set.

[T,Y,TE,YE,IE] = solver('F',tspan,y0,options) with the Events property in options set to 'on', solves as above while also locating zero crossings of an event function defined in the ODE file. The ODE file must be coded so that $F(t,y,'events')$ returns appropriate information. See odefile for details. Output TE is a column vector of times at which events occur, rows of YE are the corresponding solutions, and indices in vector IE specify which event occurred.

When called with no output arguments, the solvers call the default output function odeplot to plot the solution as it is computed. An alternate method is to set the OutputFcn property to 'odeplot'. Set the OutputFcn property to 'odephas2' or 'odephas3' for two- or three-dimensional phase plane plotting. See odefile for details.

The solvers of the ODE suite can solve problems of the form $M(t, y) y' = F(t, y)$ with a mass matrix M that is nonsingular and (usually) sparse. Use odeset to set Mass to 'M', 'M(t)', or 'M(t,y)' if the ODE file F.m is coded so that $F(t,y,'mass')$ returns a constant, time-dependent, or time-and-state-dependent mass matrix, respectively. The default value of Mass is 'none'. The ode23s solver can only solve problems with a constant mass matrix M . For examples of mass matrix problems, see fem1ode, fem2ode, or batonode.

For the stiff solvers ode15s, ode23s, ode23t, and ode23tb the Jacobian matrix $\partial F / \partial y$ is critical to reliability and efficiency so there are special options. Set JConstant to 'on' if $\partial F / \partial y$ is constant. Set Vectorized to 'on' if the ODE file is coded so that $F(t,[y1\ y2\ \dots])$ returns $[F(t,y1)\ F(t,y2)\ \dots]$. Set JPattern to 'on' if $\partial F / \partial y$ is a sparse matrix and the ODE file is coded so that $F([],[],'jpattern')$ returns a sparsity pattern matrix of 1's and 0's showing the nonzeros of $\partial F / \partial y$. Set Jacobian to 'on' if the ODE file is coded so that $F(t,y,'jacobian')$ returns $\partial F / \partial y$.

If M is singular, then $M(t) * y' = F(t, y)$ is a differential algebraic equation (DAE). DAEs have solutions only when $y0$ is consistent, that is, if there is a vector $yp0$ such that $M(t0) * yp0 = f(t0, y0)$. The ode15s and ode23t solvers can solve DAEs of index 1 provided that M is not state dependent and $y0$ is sufficiently close to being consistent. If there is a mass matrix, you can use odeset to set the MassSingular property to 'yes', 'no', or 'maybe'. The default value of 'maybe' causes the solver to test whether the problem is a DAE. If it is, the solver treats $y0$ as a guess, attempts to compute consistent initial conditions that are close to $y0$, and continues to solve the problem. When solving DAEs, it is very advantageous to formulate the problem so that M is a diagonal matrix (a semi-explicit DAE). For examples of DAE problems, see hb1dae or amp1dae.

Algorithms

ode45 is based on an explicit Runge-Kutta (4,5) formula, the Dormand-Prince pair. It is a *one-step* solver - in computing $y(t_n)$, it needs only the solution at the immediately preceding time point, $y(t_{n-1})$. In general, ode45 is the best function to apply as a "first try" for most problems.

ode23 is an implementation of an explicit Runge-Kutta (2,3) pair of Bogacki and Shampine. It may be more efficient than ode45 at crude tolerances and in the presence of moderate stiffness. Like ode45, ode23 is a one-step solver.

ode113 is a variable order Adams-Bashforth-Moulton PECE solver. It may be more efficient than ode45 at stringent tolerances and when the ODE function is particularly expensive to evaluate. ode113 is a *multistep* solver - it normally needs the solutions at several preceding time points to compute the current solution.

The above algorithms are intended to solve non-stiff systems. If they appear to be unduly slow, try using one of the stiff solvers below.

ode15s is a variable order solver based on the numerical differentiation formulas, NDFs. Optionally, it uses the backward differentiation formulas, BDFs (also known as Gear's method) that are usually less efficient. Like ode113, ode15s is a multistep solver. If you suspect that a problem is stiff or if ode45 has failed or was very inefficient, try ode15s.

ode23s is based on a modified Rosenbrock formula of order 2. Because it is a one-step solver, it may be more efficient than ode15s at crude tolerances. It can solve some kinds of stiff problems for which ode15s is not effective.

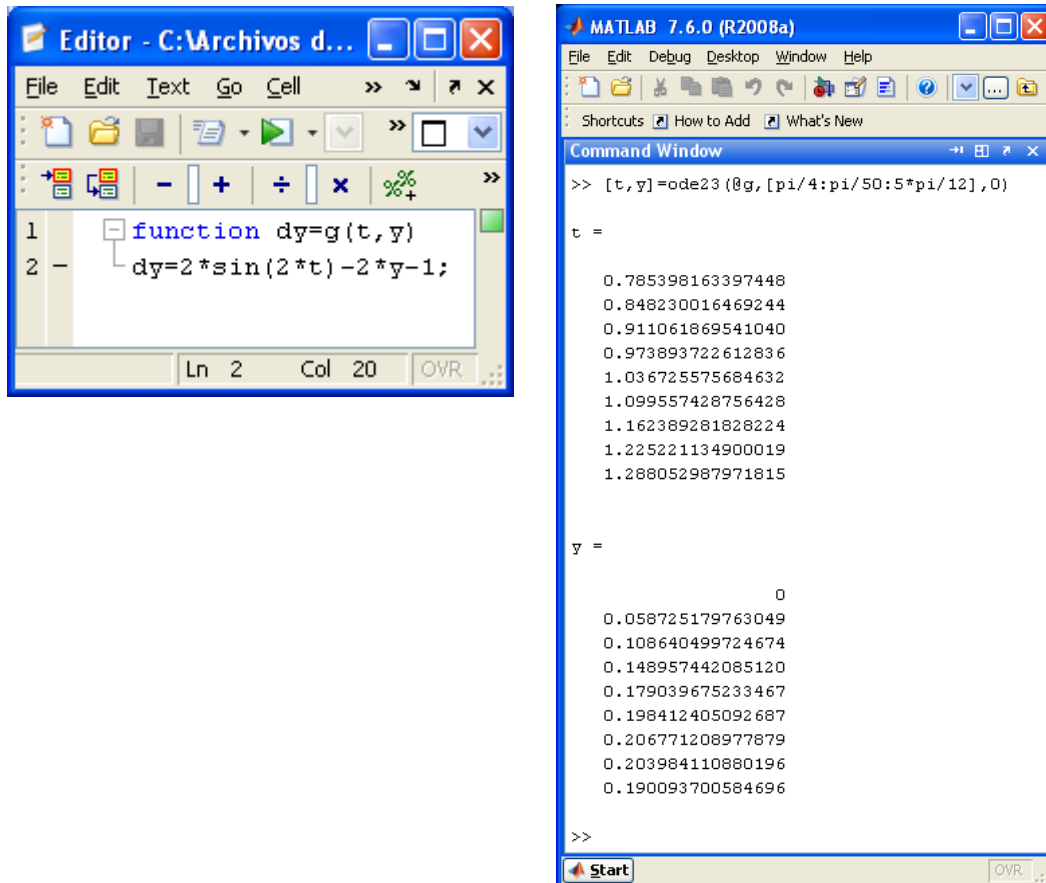
ode23t is an implementation of the trapezoidal rule using a "free" interpolant. Use this solver if the problem is only moderately stiff and you need a solution without numerical damping.

ode23tb is an implementation of TR-BDF2, an implicit Runge-Kutta formula with a first stage that is a trapezoidal rule step and a second stage that is a backward differentiation formula of order two. By construction, the same iteration matrix is used in evaluating both stages. Like ode23s, this solver may be more efficient than ode15s at crude tolerances

Ejemplo:

Obtenga la solución aproximada y exacta de la ecuación diferencial:

$$2y + y' = 2\sin(2t) - 1, \quad y\left(\frac{\pi}{4}\right) = 0, \quad \frac{\pi}{4} \leq t \leq \frac{5\pi}{12}, \quad h = \frac{\pi}{50}$$



The left screenshot shows the MATLAB Editor window with the following code:

```
1 function dy=g(t,y)
2     dy=2*sin(2*t)-2*y-1;
```

The right screenshot shows the MATLAB Command Window with the following commands and output:

```
>> [t,y]=ode23(@g,[pi/4:pi/50:5*pi/12],0)

t =

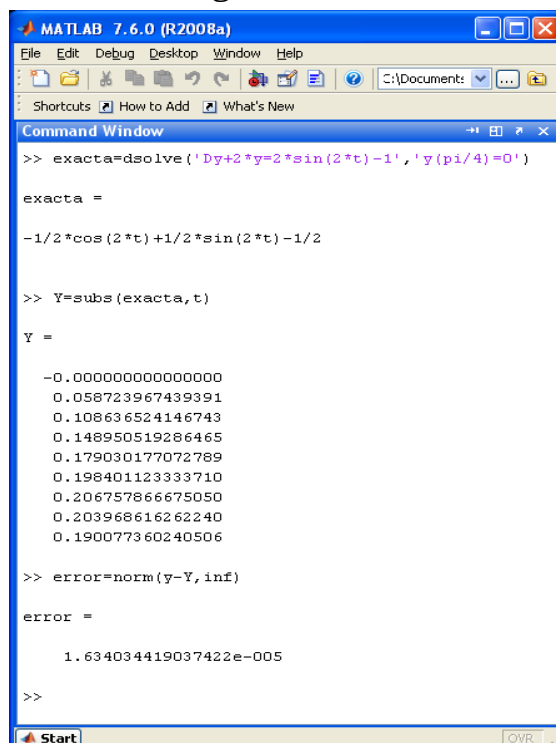
    0.785398163397448
    0.848230016469244
    0.911061869541040
    0.973893722612836
    1.036725575684632
    1.099557428756428
    1.162389281828224
    1.225221134900019
    1.288052987971815

y =

         0
    0.058725179763049
    0.108640499724674
    0.148957442085120
    0.179039675233467
    0.198412405092687
    0.206771208977879
    0.203984110880196
    0.190093700584696

>>
```

Para obtener la solución exacta de la ecuación diferencial y luego evaluarla en el intervalo de valores y obtener el error entre el valor exacto y el aproximado se realiza lo siguiente:



The MATLAB Command Window shows the following commands and output:

```
>> exacta=dsolve('Dy+2*y=2*sin(2*t)-1','y(pi/4)=0')

exacta =

-1/2*cos(2*t)+1/2*sin(2*t)-1/2

>> Y=subs(exacta,t)

Y =

-0.000000000000000
    0.058723967439391
    0.108636524146743
    0.148950519286465
    0.179030177072789
    0.198401123333710
    0.206757866675050
    0.203968616262240
    0.190077360240506

>> error=norm(y-Y,inf)

error =

    1.634034419037422e-005

>>
```

Ejemplo:

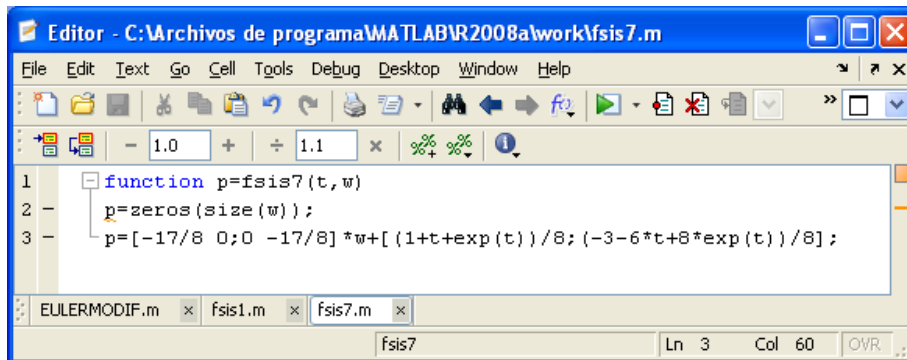
Resuelva el siguiente sistema de ecuaciones diferenciales:

$$3x' + 5x + y' - 2y = t$$

$$3x' + 6x + y' + y = e^t$$

Sujeto a las condiciones iniciales: $x(0) = 1$, $y(0) = 0$, $0 \leq t \leq 1$, $h = 0.1$

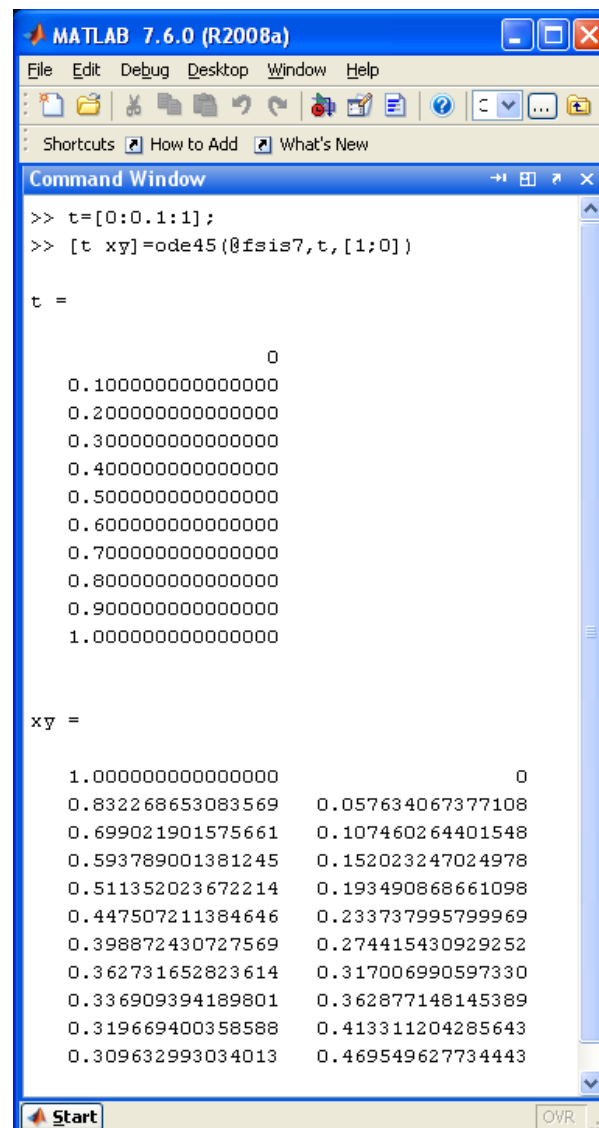
La solución aproximada se obtiene así:



```

Editor - C:\Archivos de programa\MATLAB\R2008a\work\fsis7.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1 function p=fsis7(t,w)
2     p=zeros(size(w));
3     p=[-17/8 0;0 -17/8]*w+[(1+t+exp(t))/8;(-3-6*t+8*exp(t))/8];
EULERMODIF.m x fsis1.m x fsis7.m x
Ln 3 Col 60 OVR

```



```

MATLAB 7.6.0 (R2008a)
File Edit Debug Desktop Window Help
Shortcuts How to Add What's New
Command Window
>> t=[0:0.1:1];
>> [t xy]=ode45(@fsis7,t,[1;0])

t =

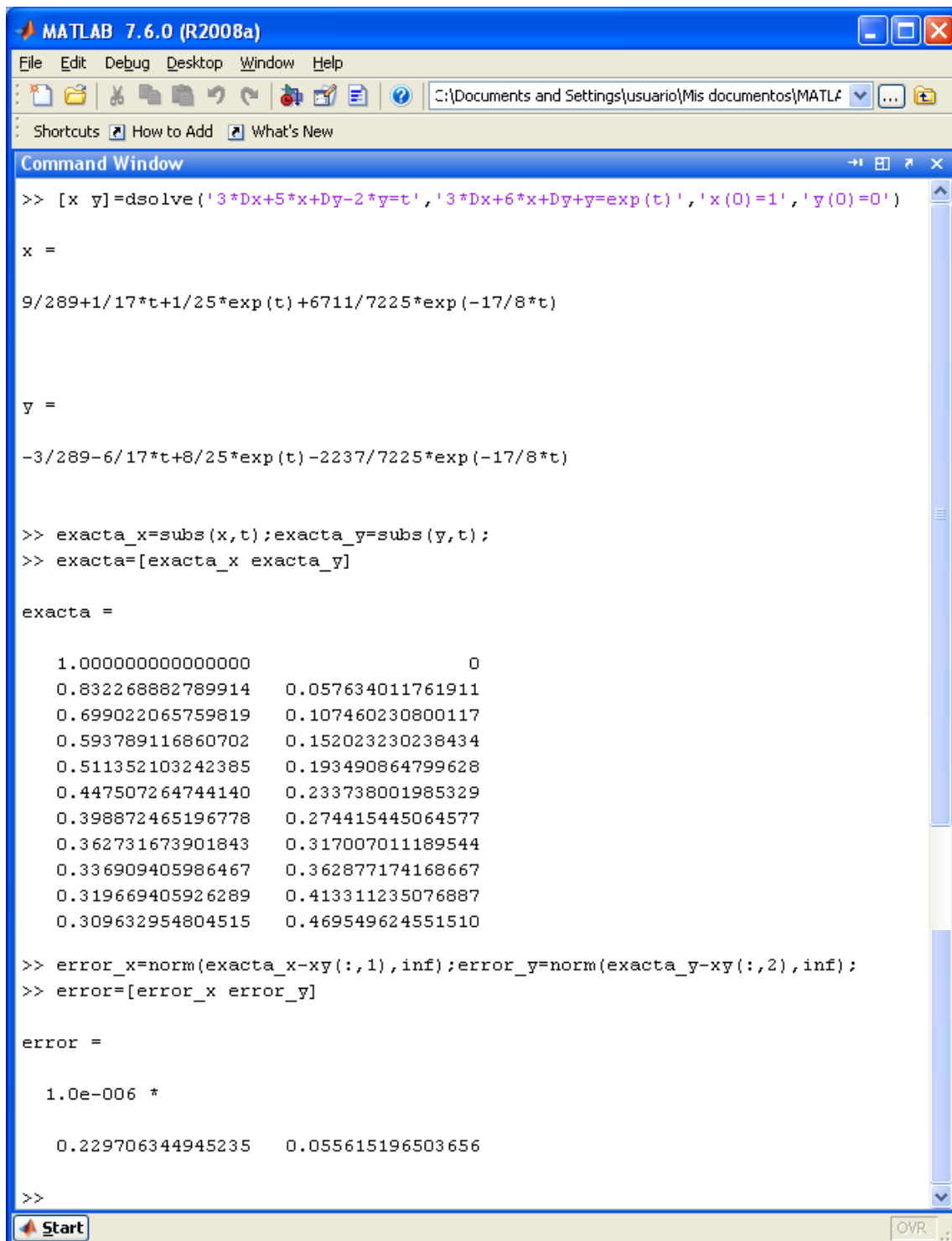
    0
0.100000000000000
0.200000000000000
0.300000000000000
0.400000000000000
0.500000000000000
0.600000000000000
0.700000000000000
0.800000000000000
0.900000000000000
1.000000000000000

xy =

1.000000000000000    0
0.832268653083569    0.057634067377108
0.699021901575661    0.107460264401548
0.593789001381245    0.152023247024978
0.511352023672214    0.193490868661098
0.447507211384646    0.233737995799969
0.398872430727569    0.274415430929252
0.362731652823614    0.317006990597330
0.336909394189801    0.362877148145389
0.319669400358588    0.413311204285643
0.309632993034013    0.469549627734443

```

La solución exacta se obtiene así:



```

MATLAB 7.6.0 (R2008a)
File Edit Debug Desktop Window Help
C:\Documents and Settings\usuario\Mis documentos\MATLAB
Shortcuts How to Add What's New

Command Window
>> [x y]=dsolve('3*Dx+5*x+Dy-2*y=t','3*Dx+6*x+Dy+y=exp(t)','x(0)=1','y(0)=0')

x =

9/289+1/17*t+1/25*exp(t)+6711/7225*exp(-17/8*t)

y =

-3/289-6/17*t+8/25*exp(t)-2237/7225*exp(-17/8*t)

>> exacta_x=subs(x,t);exacta_y=subs(y,t);
>> exacta=[exacta_x exacta_y]

exacta =

    1.000000000000000    0
    0.832268882789914    0.057634011761911
    0.699022065759819    0.107460230800117
    0.593789116860702    0.152023230238434
    0.511352103242385    0.193490864799628
    0.447507264744140    0.233738001985329
    0.398872465196778    0.274415445064577
    0.362731673901843    0.317007011189544
    0.336909405986467    0.362877174168667
    0.319669405926289    0.413311235076887
    0.309632954804515    0.469549624551510

>> error_x=norm(exacta_x-xy(:,1),inf);error_y=norm(exacta_y-xy(:,2),inf);
>> error=[error_x error_y]

error =

    1.0e-006 *

    0.229706344945235    0.055615196503656

>>

```

A continuación se muestra un programa para aproximar la solución numérica de una ecuación diferencial sujeta a una condición inicial y luego se indica la manera de correr el programa:

Ejemplo:

Resuelva $y' - y = \cos(t)$ en $t \in [0, 0.45]$, si $y(0) = 0$, $h = 0.05$


```

Editor - C:\Archivos de programa\MATLAB\R2008a\work\EULERMODIF.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1 % este programa resuelve una ecuación diferencial empleando
2 % el método de Euler Modificado
3 % f es la función definida
4 % a y b son los extremos del intervalo
5 % ya es el valor inicial
6 % h es el tamaño del paso
7 function E=eulermodif(a,b,ya,h)
8 warning('off','MATLAB:dispatcher:InexactCaseMatch')
9 syms t y
10 f=y+cos(t);
11 m=(b-a)/h;
12 T=a:h:b;
13 Y(1)=ya;
14 for j=1:m
15     Y(j+1)=Y(j)+0.5*h*subs(f,(t,y),(T(j),Y(j)))+0.5*h*subs(f,(t,y),(T(j+1),Y(j)+h*subs(f,(t,y),(T(j),Y(j)))));
16 end
17 E=[T' Y'];

```

```

MATLAB 7.6.0 (R2008a)
File Edit Debug Desktop Window Help
Current Directory: C:\Documents and Settings\usuario\Mis documentos\MATLAB
Shortcuts How to Add What's New
Command Window
>> E=eulermodif(0,0.45,0,0.05)

E =

      0      0
0.050000000000000 0.051218756509874
0.100000000000000 0.104936016248324
0.150000000000000 0.161152123368000
0.200000000000000 0.219868074932462
0.250000000000000 0.281085871983849
0.300000000000000 0.344808886221066
0.350000000000000 0.411042242300627
0.400000000000000 0.479793215780850
0.450000000000000 0.551071646741011

>> exacta_y=dsolve('Dy=y+cos(t)','y(0)=0')

exacta_y =

-1/2*cos(t)+1/2*sin(t)+1/2*exp(t)

>> Y=subs(exacta_y,E(:,1))

Y =

      0
0.051250002625868
0.105000084722225
0.161250648632920
0.220002755556995
0.281258477115810
0.345021262555868
0.411296321600665
0.480091022973518
0.551415308624361

>> error=norm(Y-E(:,2),inf)

error =

3.436618833497773e-004

>>

```

Otra alternativa de construir el archivo M sería así:

Resuelva $t y' - 2 y = t^2$ en $t \in [1, 29/20]$, si $y(1) = -2$, $h = 0.05$

```

1  % este programa resuelve una ecuación diferencial empleando
2  % el método de Euler Modificado
3  % f es la función definida entre comillas
4  % g es la ecuación diferencial entre comillas
5  % e es la condición inicial entre comillas
6  % a y b son los extremos del intervalo
7  % ya es el valor inicial
8  % h es el tamaño del paso
9  function E=eulermodif2(f,g,e,a,b,ya,h)
10 - syms t y
11 - m=(b-a)/h;
12 - T=a:h:b;
13 - F=dsolve(g,e)
14 - w(1)=ya;
15 - for j=1:m+1
16 -     w(j+1)=w(j)+0.5*h*subs(f,{t,y},{T(j),w(j)})+0.5*h*subs(f,{t,y},{T(j+1),w(j)+h*subs(f,{t,y},{T(j),w(j)})});
17 - end
18 - Y=double(subs(F,t,T));
19 - fprintf('      T              Wi+1              y(t)');
20 - fprintf('\n');
21 - E=[T'      w'      Y'];

```

```

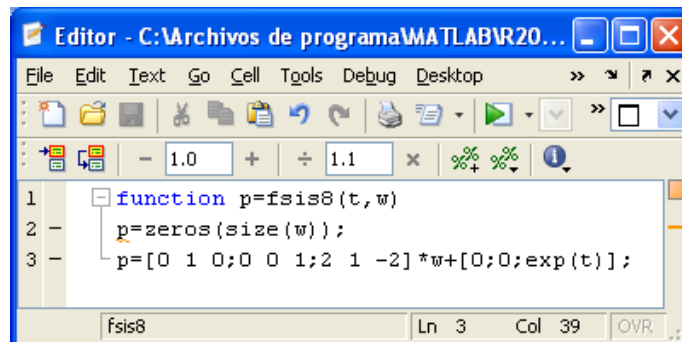
>> E=eulermodif2(' (2*y+t^2)/t ','Dy=(2*y+t^2)/t ','y(1)=-2',1,29/20,-2,0.05)
F =
(log(t)-2)*t^2
      T              Wi+1              y(t)
E =
1.000000000000000 -2.000000000000000 -2.000000000000000
1.050000000000000 -2.151130952380953 -2.151208844003201
1.100000000000000 -2.304520330859617 -2.304674682436767
1.150000000000000 -2.459935262777407 -2.460164831208852
1.200000000000000 -2.617153248060185 -2.617456958216705
1.250000000000000 -2.775961274145300 -2.776338201071547
1.300000000000000 -2.936155039575609 -2.936604393049940
1.350000000000000 -3.097538269171989 -3.098059380259258
1.400000000000000 -3.259922106927020 -3.260514416222422
1.450000000000000 -3.423124575287480 -3.423787622600704
>> Error=norm(E(:,3)-E(:,2),inf)
Error =
6.630473132238635e-004

```

Ejemplo:

Resuelva $y'''(t) + 2y''(t) - y'(t) - 2y(t) = e^t$, sujeta a las condiciones iniciales:

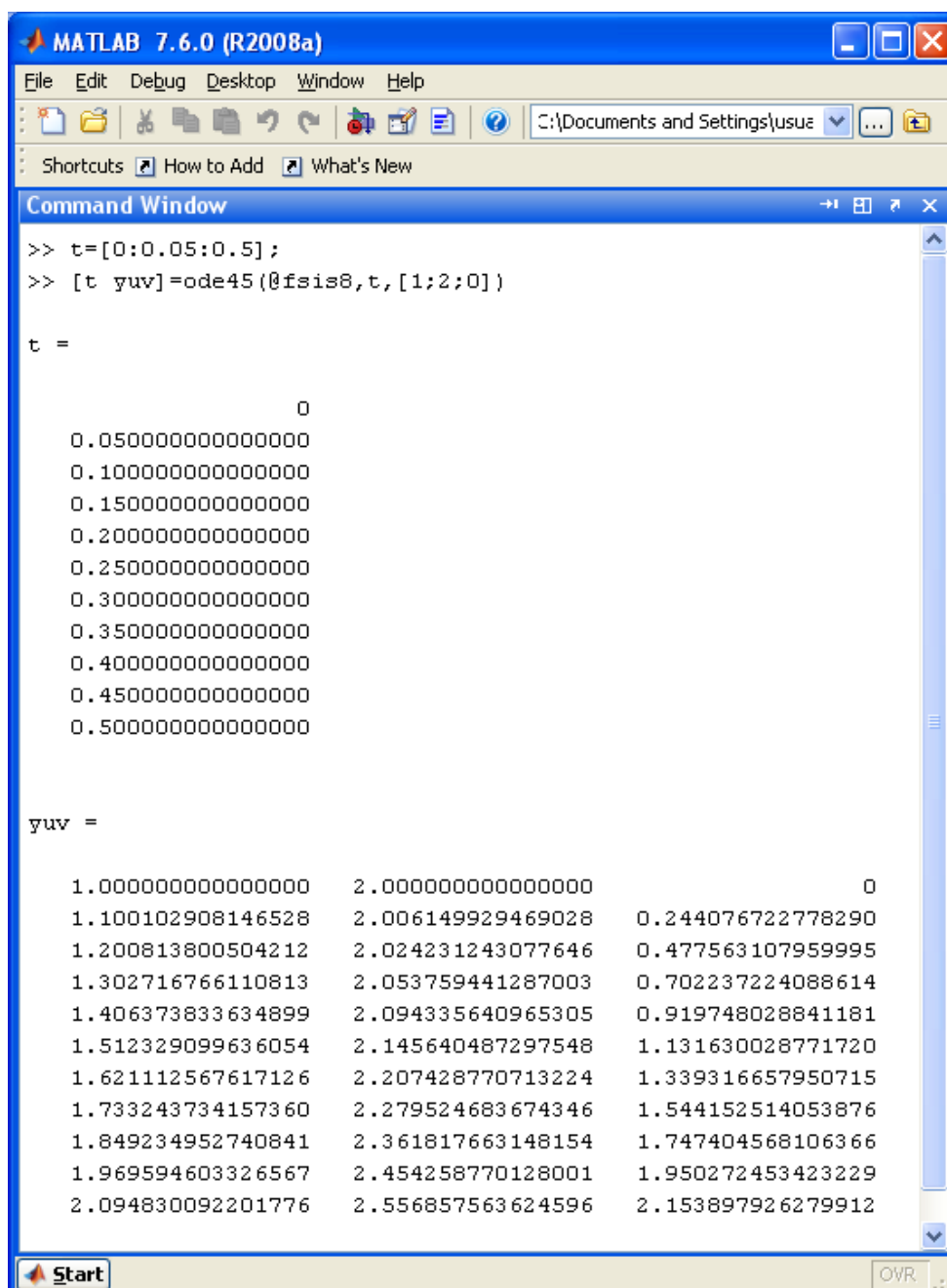
$$y(0) = 1, \quad y'(0) = 2, \quad y''(0) = 0 \quad 0 \leq t \leq 0.5, \quad h = 0.05$$



```

1 function p=fsis8(t,w)
2     p=zeros(size(w));
3     p=[0 1 0;0 0 1;2 1 -2]*w+[0;0;exp(t)];

```



```

MATLAB 7.6.0 (R2008a)
File Edit Debug Desktop Window Help
C:\Documents and Settings\usua
Shortcuts How to Add What's New

Command Window
>> t=[0:0.05:0.5];
>> [t yuv]=ode45(@fsis8,t,[1;2;0])

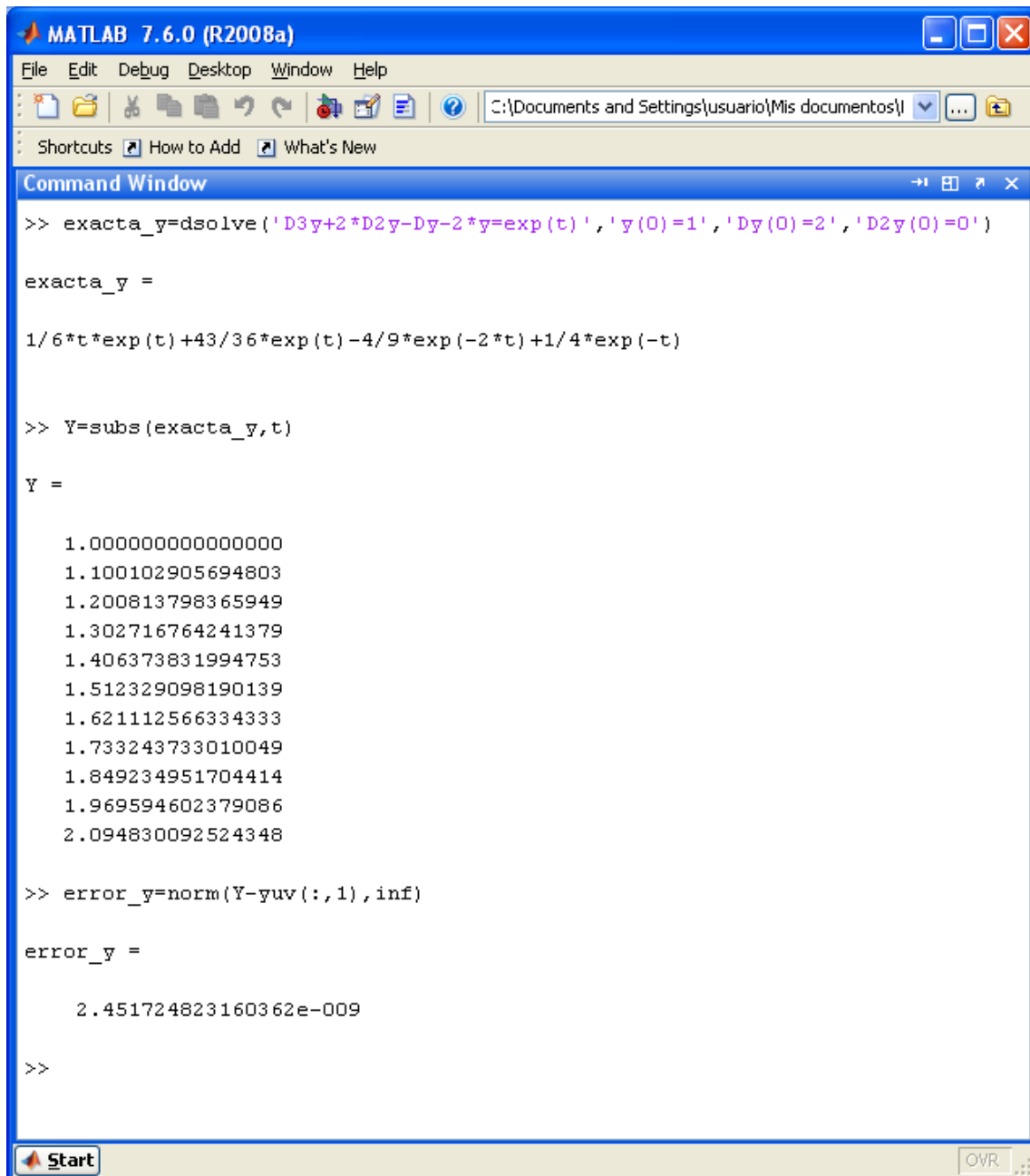
t =

    0
0.050000000000000
0.100000000000000
0.150000000000000
0.200000000000000
0.250000000000000
0.300000000000000
0.350000000000000
0.400000000000000
0.450000000000000
0.500000000000000

yuv =

    1.000000000000000    2.000000000000000    0
    1.100102908146528    2.006149929469028    0.244076722778290
    1.200813800504212    2.024231243077646    0.477563107959995
    1.302716766110813    2.053759441287003    0.702237224088614
    1.406373833634899    2.094335640965305    0.919748028841181
    1.512329099636054    2.145640487297548    1.131630028771720
    1.621112567617126    2.207428770713224    1.339316657950715
    1.733243734157360    2.279524683674346    1.544152514053876
    1.849234952740841    2.361817663148154    1.747404568106366
    1.969594603326567    2.454258770128001    1.950272453423229
    2.094830092201776    2.556857563624596    2.153897926279912

```



The screenshot shows the MATLAB 7.6.0 (R2008a) Command Window. The user has entered the following commands and received the following outputs:

```
>> exacta_y=dsolve('D3y+2*D2y-Dy-2*y=exp(t)', 'y(0)=1', 'Dy(0)=2', 'D2y(0)=0')

exacta_y =

1/6*t*exp(t)+43/36*exp(t)-4/9*exp(-2*t)+1/4*exp(-t)

>> Y=subs(exacta_y,t)

Y =

1.000000000000000
1.100102905694803
1.200813798365949
1.302716764241379
1.406373831994753
1.512329098190139
1.621112566334333
1.733243733010049
1.849234951704414
1.969594602379086
2.094830092524348

>> error_y=norm(Y-yuv(:,1),inf)

error_y =

2.451724823160362e-009

>>
```

RESUELVA LOS SIGUIENTES EJERCICIOS:

1. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $\mathbf{y}' + \mathbf{ysec(t)} = \mathbf{sec(t)}$, $0 \leq t \leq 0.5$, $y(0) = 4$, con $h = \frac{1}{40}$. Para la solución aproximada emplee el comando **ode23**, el método de **Runge-Kutta de orden cuatro**, el método de **Punto Medio**.

2. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $(t^2 + 2y^2) dt = ty dy$, $-1 \leq t \leq -0.85$, $y(-1) = 1$, $h = \frac{3}{100}$. Para la solución aproximada emplee el comando **ode45**, el método de **Euler modificado**, el método del **Heun**.
3. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $(t + 1)y' + y = \ln(t)$, $1 \leq t \leq 1.8$, $y(1) = 10$, con $h = \frac{1}{20}$. Para la solución aproximada emplee el comando **ode113**, el método de **Adams-Bashforth de cinco pasos**, el método de **Runge-Kutta-Fehlberg de cuarto orden**.
4. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $y' + \tan(t)y = \cos^2(t)$, $0 \leq t \leq 0.2$, $y(0) = -1$, $h = \frac{1}{50}$. Para la solución aproximada emplee el comando **ode45**, el método de **Taylor de orden cuatro**, el método de **Euler**.
5. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $y' = 1 + (t - y)^2$, $2 \leq t \leq 2.3$, $y(2) = 1$, con $h = \frac{1}{50}$. Para la solución aproximada emplee el comando **ode45**, el método de **Adams-Bashforth de cuatro pasos**, el método de **Heun**.
6. Obtenga la solución **exacta y aproximada** para el siguiente sistema de ecuaciones diferenciales:

$$3x' - 5x + 3y' - 2y = \frac{7}{3}e^t - \frac{1}{3}e^{4t} + 8, \quad x(0) = 2$$

$$4x' + x + 4y' + 7y = \frac{185}{3}e^t - \frac{1}{6}e^{4t} - 28, \quad y(0) = \frac{1}{2}$$

Para los siguientes valores: $0 \leq t \leq \frac{3}{5}$, con $h = \frac{1}{20}$

Para la solución aproximada emplee el comando **ode45**, el método de **Runge-Kutta de orden cuatro**.

7. Obtenga la solución **exacta y aproximada** para el siguiente sistema de ecuaciones diferenciales:

$$x' + 3x + y' = 1$$

$$x' - x + y' - y = e^t, \quad \text{sujeto a : } x_{(0)} = 0, \quad y_{(0)} = 0$$

Para los siguientes valores: $0 \leq t \leq 1$, considere $h = \frac{1}{10}$

Para la solución aproximada emplee el comando **ode23**, el método de **Runge-Kutta de orden cuatro**.

8. Obtenga la solución **exacta y aproximada** para el siguiente sistema de ecuaciones diferenciales:

$$4x' + 4x + y' = 2$$

$$4x' + 2x + y' - y = e^t, \quad \text{sujeto a : } x_{(0)} = 2, \quad y_{(0)} = 1$$

Para los siguientes valores: $0 \leq t \leq 0.6$ considere $h = \frac{1}{20}$

Para la solución aproximada emplee el comando **ode45**, el método de **Runge-Kutta de orden cuatro**.

9. Obtenga la solución **exacta y aproximada** de cada una de las siguientes ecuaciones diferenciales:

a) $y'' - 2y' + y = te^t - t$, sujeta a $y_{(0)} = 0$, $y'_{(0)} = 0$, para los valores $0 \leq t \leq 1$ considere $h = \frac{1}{10}$.

b) $y''' + 2y'' - y' - 2y = te^t$, sujeta a: $y_{(0)} = 2$, $y'_{(0)} = 1$, $y''_{(0)} = -1$, para los valores $0 \leq t \leq 0.5$ considere $h = \frac{1}{20}$.

c) $t^3 y''' - t^2 y'' + 3ty' - 4y = 5t^3 \ln(t)$, sujeta a: $y_{(1)} = 0$, $y'_{(1)} = 1$, $y''_{(1)} = 3$ para los valores $1 \leq t \leq 2.4$ considere $h = \frac{1}{5}$

d) $y''' + 2y'' - y' - 2y = e^t$, sujeta a: $y(0) = 1$, $y'(0) = 2$, $y''(0) = 0$

para los valores $0 \leq t \leq 0.5$, considere $h = 0.05$

Para la solución aproximada emplee el comando **ode45**, el método de **Runge-Kutta de orden cuatro**.

10. Determine la carga en el capacitor y la corriente en un circuito RLC en serie, si se sabe que: $L=0.5$ H, $R=20\Omega$, $C=0.02$ F, $E(t)=25$ V, $q(0)=0$ Coulombs, $i(0)=2$ Amperes, para cada uno de los valores de t siguientes: $0 \leq t \leq 0.15$, con $h=\frac{1}{20}$. Para la solución aproximada emplee el comando **ode45**, el método de **Runge-Kutta de orden cuatro**, además obtenga la **solución exacta**.

11. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $y' = e^{3t+2y}$, $0 \leq t \leq \frac{1}{25}$, $y(0) = 0$, con $h = \frac{1}{200}$. Para la solución aproximada emplee el comando **ode45**, el método de **Taylor de orden cuatro**, el método de **Euler modificado**.

12. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $(\cos(t)\sin(t) - ty^2)dt + y(1 - t^2)dy = 0$, $0 \leq t \leq \frac{2}{5}$, $y(0) = 2$, con $h = \frac{1}{20}$. Para la solución aproximada emplee el comando **ode23**, el método de **Adams-Bashforth de cinco pasos**, el método de **Runge-Kutta- Fehlberg de cuarto orden**.

13. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $4t + 3y + 3(t + y^2)y' = 0$, $3 \leq t \leq \frac{19}{5}$, $y(3) = -1.57$, considere $h = \frac{1}{10}$. Para la solución aproximada emplee el comando **ode23**, el método de **Adams-Bashforth de cinco pasos**, el método de **Heun**.

14. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $\frac{y}{t-1}dt + (\ln(t-1) + 2y)dy = 0$, $2 \leq t \leq \frac{9}{4}$, $y(2) = 4$, con $h = \frac{1}{40}$.
Para la solución aproximada emplee el comando **ode45**, el método del **Punto Medio**, el método de **Runge-Kutta- Fehlberg de quinto orden**.
15. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $(y^2 + 3ty)dt = (4t^2 + ty)dy$, $1 \leq t \leq \frac{3}{2}$, $y(1) = 1$, con $h = \frac{1}{20}$.
Para la solución aproximada emplee el comando **ode45**, el método de **Runge-Kutta de orden cuatro**, el método del **Punto Medio**.
16. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $y' = \frac{2y}{t} + \left(\frac{2y}{t}\right)^2$, $1 \leq t \leq 1.5$, $y(1) = 2$, con $h = \frac{1}{16}$.
Para la solución aproximada emplee el comando **ode23**, el método de **Taylor de orden cuatro**, el método de **Euler Modificado**.
17. Obtenga la solución **exacta y aproximada** para la ecuación diferencial: $y' = \frac{ty + y^2}{t^2}$, $1 \leq t \leq 1.9$, $y(1) = 2$, con $h = \frac{1}{10}$.
Para la solución aproximada emplee el comando **ode113**, el método del **Punto Medio**, el método de **Heun**.
18. Dada la siguiente ecuación diferencial:
 $y' - e^y \sin(t) = 0$, $0 \leq t \leq \frac{1}{20}$, $y(0) = 0$, con $h = \frac{1}{100}$
a) Emplee el **Método de Taylor de orden 4** y el comando **ode23**, para obtener la solución aproximada en los valores de "t".
b) Además aproxime el valor de $y(0.035)$, mediante **neville**
19. Dada la siguiente ecuación diferencial:
 $t^2 dy + (y^2 + ty)dt = 0$, $1 \leq t \leq \frac{6}{5}$, $y(1) = 1$, con $h = \frac{1}{20}$

a) Determine la solución aproximada para cada valor de “t”, empleando el **Método de Heun** y el comando **ode113**.

b) Además obtenga la solución exacta para cada valor de “t”

20. Dada la siguiente ecuación diferencial:

$$(1 + \ln t + y/t)dt = (1 - \ln t)dy, \quad 1 \leq t \leq \frac{49}{40}, \quad y(1) = 2, \quad \text{con } h = \frac{1}{40}$$

a) Emplee el **Método de Adams-Bashforth de cinco pasos**, para obtener la solución aproximada en cada uno de los valores de “t”, los valores iniciales obténgalos mediante la solución exacta. Además emplee el comando **ode45**.

b) Obtenga la solución exacta y evalúela en cada valor de “t”.

21. Dada la siguiente ecuación diferencial:

$$dy - y \tan(t)dt = 2e^t dt, \quad 0 \leq t \leq \frac{1}{50}, \quad y(0) = 2, \quad \text{con } h = \frac{1}{200}$$

a) Emplee el **Método de Runge-Kutta-Fehlberg de quinto orden**, para obtener la solución aproximada en los valores de “t”. Además emplee el comando **ode23**.

b) Obtenga la solución exacta y evalúela en cada valor de “t”.

22. Obtenga la solución **exacta y aproximada** para el siguiente sistema diferencial:

$$5x' + x + 5y' + 2y = 2e^{-2t} + e^{-t} + \sin(t) + 5\cos(t), \quad x(0) = 0$$

$$4x' - 3x + 4y' - y = 4e^{-2t} + e^{-t} - 3\sin(t) + 4\cos(t), \quad y(0) = -1$$

Para los siguientes valores $0 \leq t \leq \frac{3}{20}$, con $h = \frac{1}{20}$

Para la solución aproximada emplee el comando **ode45**, el método de **Runge-Kutta de orden cuatro**

23. Emplee el **Método de Extrapolación**, con una tolerancia 10^{-12} , para obtener la solución aproximada de la siguiente ecuación diferencial:

$$(t^2 + 4)dy - tydt = 0, \quad 0 \leq t \leq \frac{1}{25}, \quad y(0) = 3, \quad \text{con } h = \frac{1}{100}$$

Además emplee el comando **ode23** y también obtenga la solución exacta