

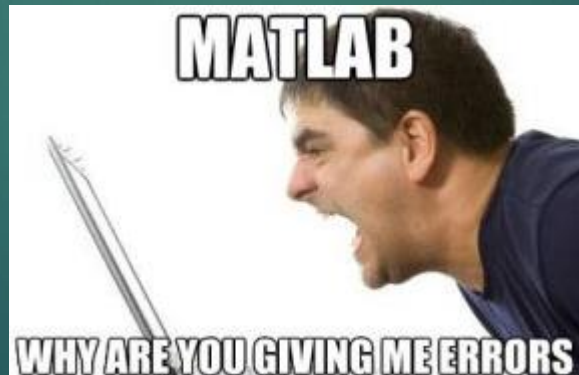


3.- Más Matlab!

DR. SERVANDO LÓPEZ AGUAYO
AGOSTO-DICIEMBRE 2020

En la clase anterior...

- ▶ Comenzamos nuestro “viaje” al aprender nociones básicas de Matlab...
- ▶ ¿Algunas primeras impresiones de Matlab?



En esta clase...

- ▶ Veremos como realizar diferentes tipos de gráficas.
- ▶ Conoceremos como importar y exportar variables.
- ▶ Manejo de errores (debugger y profiler)
- ▶ Algunas otras monerías: GUI, Mupad, Simulink,etc
- ▶ Y más más funciones! 😊

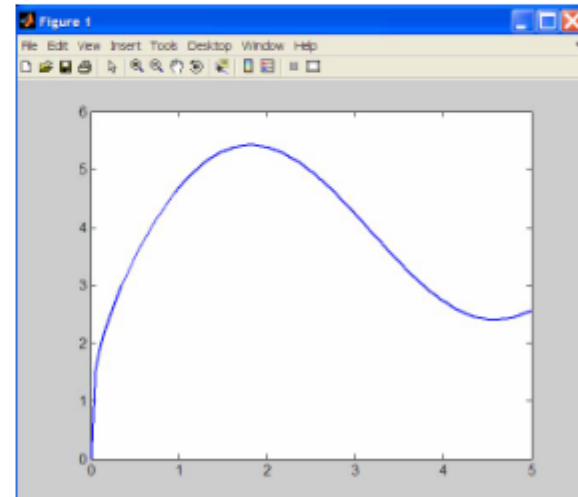
Graficando con Matlab

La función `plot(x,y)` grafica dos vectores de abscisas y ordenadas:

$$\mathbf{x} = [x_1, x_2, x_3, \dots, x_N]$$
$$\mathbf{y} = [y_1, y_2, y_3, \dots, y_N]$$

Ejemplo:

```
>> x = linspace(0,5,100);  
>> y = 3*x.^(1/4) + 2*sin(x);  
>> figure(1); plot(x,y);
```



Algunas opciones de plot

```
>> x = linspace(0,5,100);  
>> y = 3*x.^(1/4) + 2*sin(x);  
>> figure(1); plot(x,y,'r','linewidth',1.5);  
>> xlabel('Valor de las x');  
>> ylabel('Valor de las y');  
>> title('Ejemplo del plot','fontsize',14);  
>> grid on
```

```
» plot(x,y,'k.-');
```

color

marker

line-style

Mostrar varias gráficas a la vez

Opción 1: Usando la misma función plot

```
>> figure(13);  
>> plot(x, y, 'b', x, 3*exp(-(x/2).^2), 'r');  
>> legend('Función 1', 'Función 2')
```

Opción 2: Usando la función hold

```
>> figure(13);  
>> plot(x, y, 'b');  
>> hold on    % Activa el modo de congelamiento  
>> plot(x, 3*exp(-(x/2).^2), 'r');  
>> hold off;  % Desactiva el modo de congelamiento
```

Un pequeño comentario...

- ▶ En general, las gráficas de Matlab son “buenas” para visualización y “tareas”.
- ▶ Sin embargo, para cuestiones de publicación, es necesario darle un formato más estético.
Normalmente se hace con algún otro software.



Personalizando los plots

- ▶ Sin embargo, prácticamente toda la gráfica puede ser personalizada! (Aunque no es tan amigable hacerlo)

```
» plot(x,y,'--s','LineWidth',2,...  
      'Color', [1 0 0], ...  
      'MarkerEdgeColor','k',...  
      'MarkerFaceColor','g',...  
      'MarkerSize',10)
```



Gráficas en escalas logarítmicas

- ▶ Matlab nos ayuda en este aspecto...

```
» semilogx(x,y,'k');  
» semilogy(y,'r.-');  
» loglog(x,y);
```

Plots en 3d

- Para graficar en 3 d utilizamos el comando plot3:

```
» time=0:0.001:4*pi;  
» x=sin(time);  
» y=cos(time);  
» z=time;  
» plot3(x,y,z,'k','LineWidth',2);  
» zlabel('Time');
```

- Revisa los siguientes comandos:

```
» xlim, ylim, zlim
```

Y claro, hay más estilos!

<code>bar</code>	diagramas de barras.
<code>barh</code>	diagramas de barras horizontales.
<code>bar3</code>	diagramas de barras con aspecto 3-D.
<code>bar3h</code>	diagramas de barras horizontales con aspecto 3-D.
<code>pie</code>	gráficas con forma de “pay”.
<code>pie3</code>	gráficas con forma de “pay” y aspecto 3-D.
<code>area</code>	similar <code>plot()</code> , pero rellenando en ordenadas de 0 a y.
<code>stairs</code>	función análoga a <code>bar</code> sin líneas internas.
<code>errorbar</code>	representa en una gráfica –mediante barras– valores de errores.
<code>compass</code>	dibuja los elementos de un vector complejo como un conjunto de vectores partiendo de un origen común.
<code>feather</code>	dibuja los elementos de un vector complejo como un conjunto de vectores partiendo de orígenes uniformemente espaciados sobre el eje de abscisas.
<code>hist</code>	histogramas de un vector.

Graficar $f(x,y)$

- ▶ ¿Cómo grafico una función de dos variables?
- ▶ 1) Generar las matrices de coordenadas x & y .
- ▶ 2) Evaluar $f(x,y)$
- ▶ 3) Graficar (por ejemplo, usando surf o mesh).
- ▶ 4) Agregar detalles a la gráfica.

Un ejemplo:

Graficar $f(x,y) = xy^2$ en el dominio $-2 < x < 2$ y $-3 < y < 3$

```
>> x = linspace(-2,2,5)
```

```
x =
```

```
    -2    -1     0     1     2
```

```
>> y = linspace(3,-3,7)
```

```
y =
```

```
     3     2     1     0    -1    -2    -3
```

```
    -2    -1     0     1     2
```

Comando: meshgrid

- Utilizar el comando meshgrid:

```
>> [X,Y] = meshgrid(x,y)
```

X =

```
-2 -1 0 1 2  
-2 -1 0 1 2  
-2 -1 0 1 2  
-2 -1 0 1 2  
-2 -1 0 1 2  
-2 -1 0 1 2  
-2 -1 0 1 2
```

Y =

```
3 3 3 3 3  
2 2 2 2 2  
1 1 1 1 1  
0 0 0 0 0  
-1 -1 -1 -1 -1  
-2 -2 -2 -2 -2  
-3 -3 -3 -3 -3
```

Evaluamos y graficamos

```
>> f = X.* Y.^2  
f =
```

```
-18 -9 0 9 18
```

```
-8 -4 0 4 8
```

```
-2 -1 0 1 2
```

```
0 0 0 0 0
```

```
-2 -1 0 1 2
```

```
-8 -4 0 4 8
```

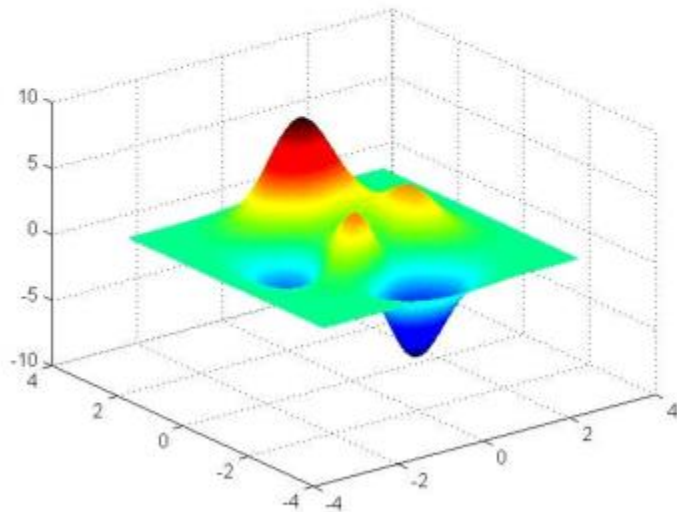
```
-18 -9 0 9 18
```

```
>> surf(X,Y,f);
```

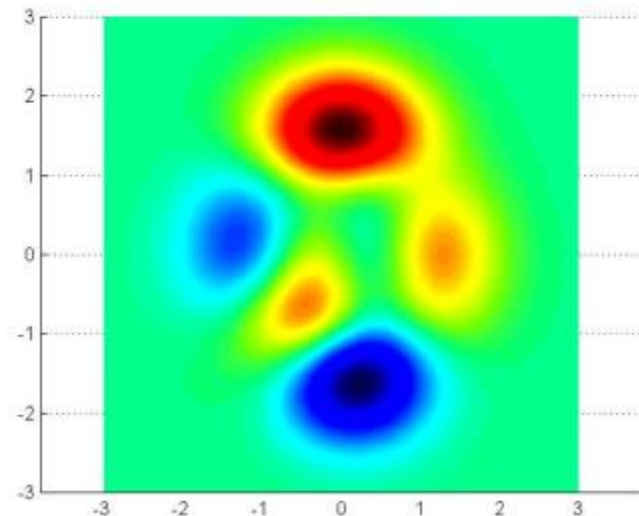


Algunas opciones de surf

```
>> figure(1); surf(x,y,z);  
>> shading interp; lighting phong;
```



```
>> view(2); axis equal;
```

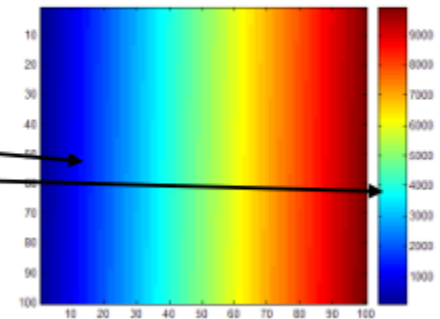


Visualización de matrices

- Cualquier matriz puede visualizarse como una imagen

» `imagesc(mat) ;`

» `colorbar`



Colormaps!

» `imagesc(mat)`

➤ default

jet

» `colormap(gray)`

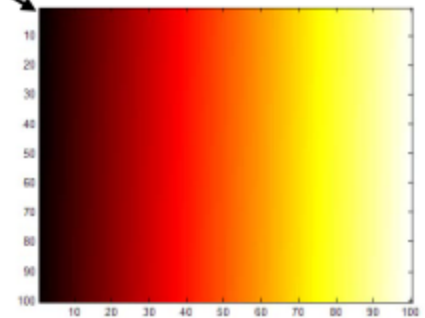
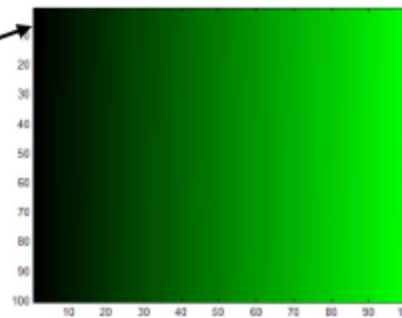
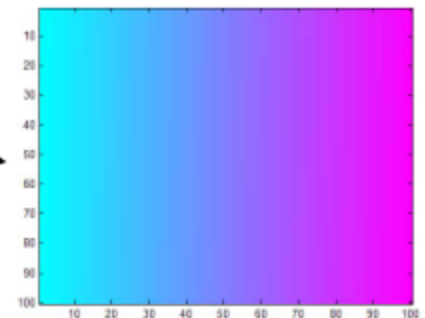
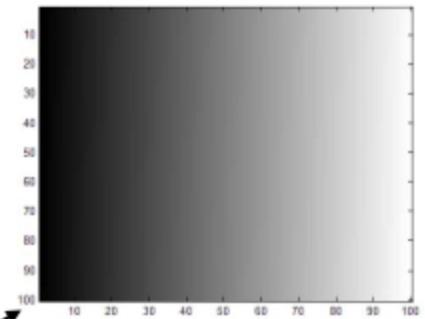
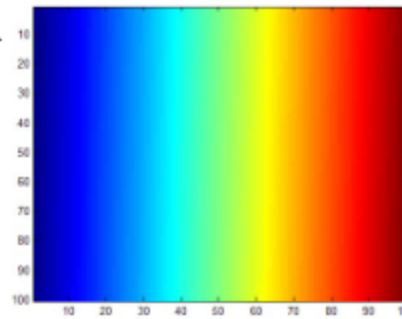
» `colormap(cool)`

» `colormap(hot(256))`

» `map=zeros(256,3);`

» `map(:,2)=(0:255)/255;`

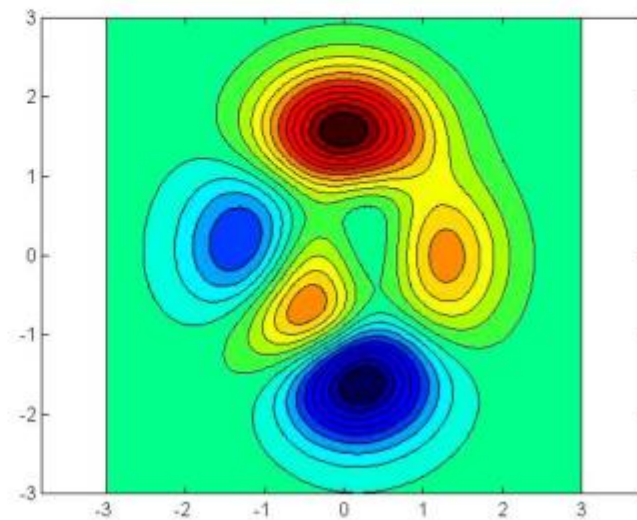
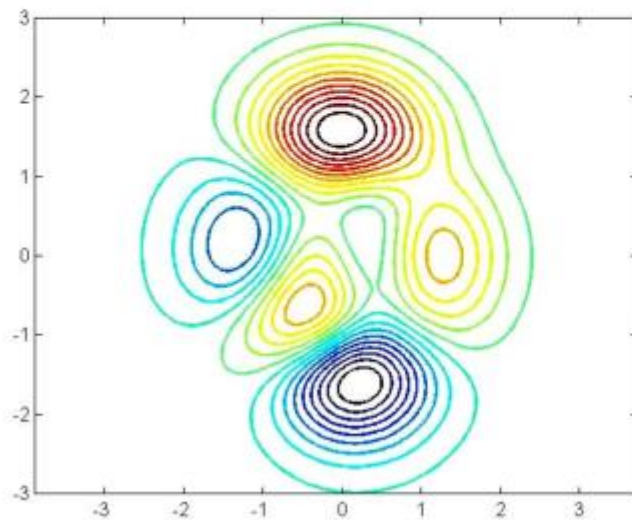
» `colormap(map);`



Otra opción: contour

```
>> figure(2); contour(x,y,z,20); axis equal
```

```
>> figure(3); contourf(x,y,z,20); axis equal
```



Actividad 1

- ▶ 1.- Crea una función que reciba como entrada seis parámetros, A_1 , A_2 , ω_1 , ω_2 , ϕ_1 y ϕ_2 : que serán, respectivamente, las amplitudes (A_1 , A_2), frecuencias angulares (ω_1 , ω_2) y fases (ϕ_1 y ϕ_2) de dos ondas senoidales.
- ▶ La función deberá mandar graficar en un figura, ambas señales senoidales, mientras que otra figura, se debe mostrar la multiplicación de ambas. No olvides darle un formato estético al resultado!

Actividad 1

- ▶ 2.- Crear una función que reciba dos parámetros: A y B. Ambos parámetros definirán una función

$$f(x,y) = A \exp(-r^2/B^2) \exp(i\theta)$$

En donde $i^2 = -1$, $r^2 = x^2 + y^2$ & $\theta = \tan^{-1}(y/x)$.

Graficar en una figura la amplitud y en otra figura la fase de dicha función, con buen formato!

Reportar: códigos generados y resultados obtenidos.

Tiempo de break

- ▶ Regresamos para nuestra parte final de introducción a Matlab! (10 min!)



Comando find

- ▶ Es un comando de gran utilidad para vectorizar.
- ▶ Comparar velocidades usando for y el comando find, para saber cuales valores son positivos en la siguiente expresión: (chechar para 100, 10000 y 1000000 de puntos)

```
» x=rand(1,100);  
» inds = find(x>0.4 & x<0.6);
```

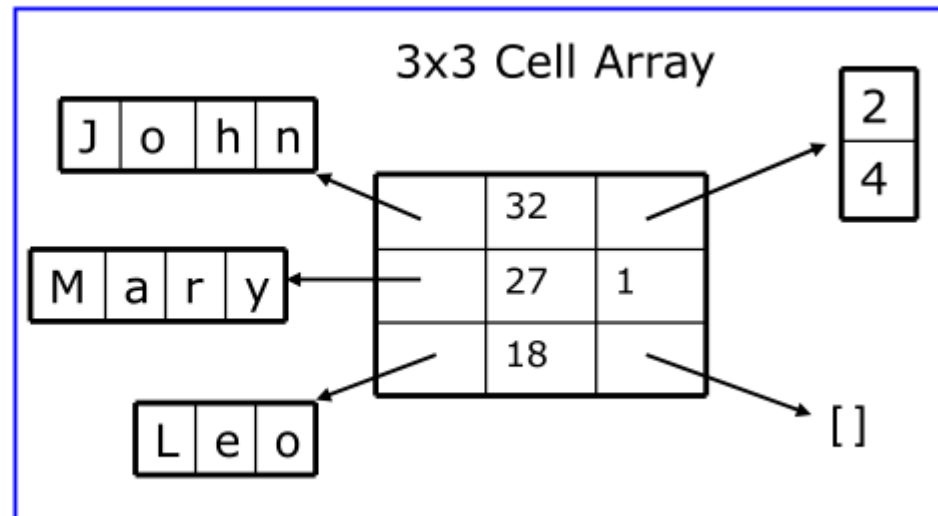
```
x= sin(linspace(0,10*pi,100)),
```

Células y estructuras

- Sirve para guardar diversos tipos de datos en una variable:

3x3 Matrix

1.2	-3	5.5
-2.4	15	-10
7.8	-1.1	4



Manejo de células

- Inicializar una célula:

```
» a=cell(3,10);
```

```
» c={'hello world',[1 5 6 2],rand(3,2)};
```

- Cómo acceder a los valores:

```
» a{1,1}=[1 3 4 -10];  
» a{2,1}='hello world 2';  
» a{1,2}=c{3};
```

Estructuras

- ▶ Inicializar una estructura

```
» s=struct ( [] ) ;
```

- ▶ Añadir valores

```
» s.name = 'Jack Bauer';  
» s.scores = [95 98 67];  
» s.year = 'G3';
```

Comando save y load

- ▶ Para salvar datos, se utiliza el comando save con la siguiente sintaxis:

```
save misdatos.mat x y u
```

- ▶ Mientras que para cargar datos se utiliza el comando load:

```
load misdatos
```

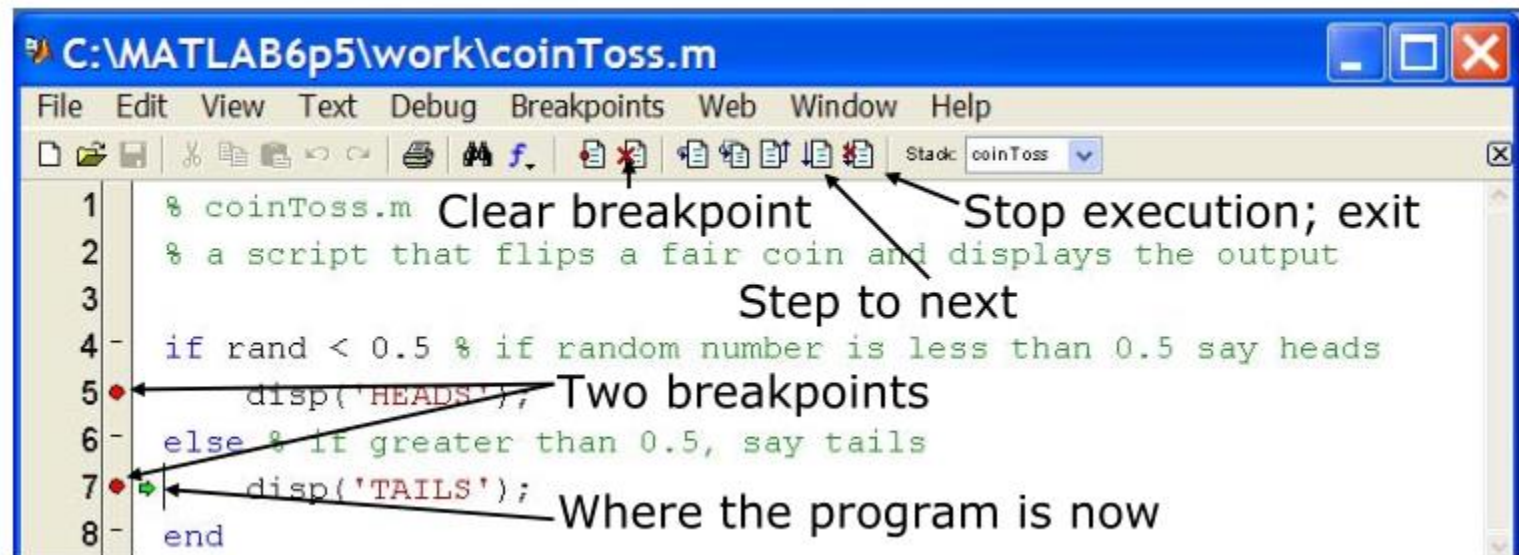
Checando nuestros códigos

- ▶ Para optimizar o verificar nuestros códigos, Matlab nos ayuda con dos herramientas: el debugger y el profile.
- ▶ Un comando útil es el disp:

```
» disp('starting loop')  
» disp('loop is over')  
» disp(strcat(['loop iteration ', num2str(n)]));
```

Debugger

- ▶ A veces será necesario ir corriendo nuestro programa paso a paso.



Profile

- Recuerden: el comando tic/toc para medir el tiempo de ejecución de un código.

```
» tic  
» CommandBlock1  
» a=toc;  
» CommandBlock2  
» b=toc;
```



Usando el profile viewer

- Es recomendable, si el código es amplio o relativamente complejo, utilizar:







```
» profile on
```

```
» profile viewer
```

Profile Summary

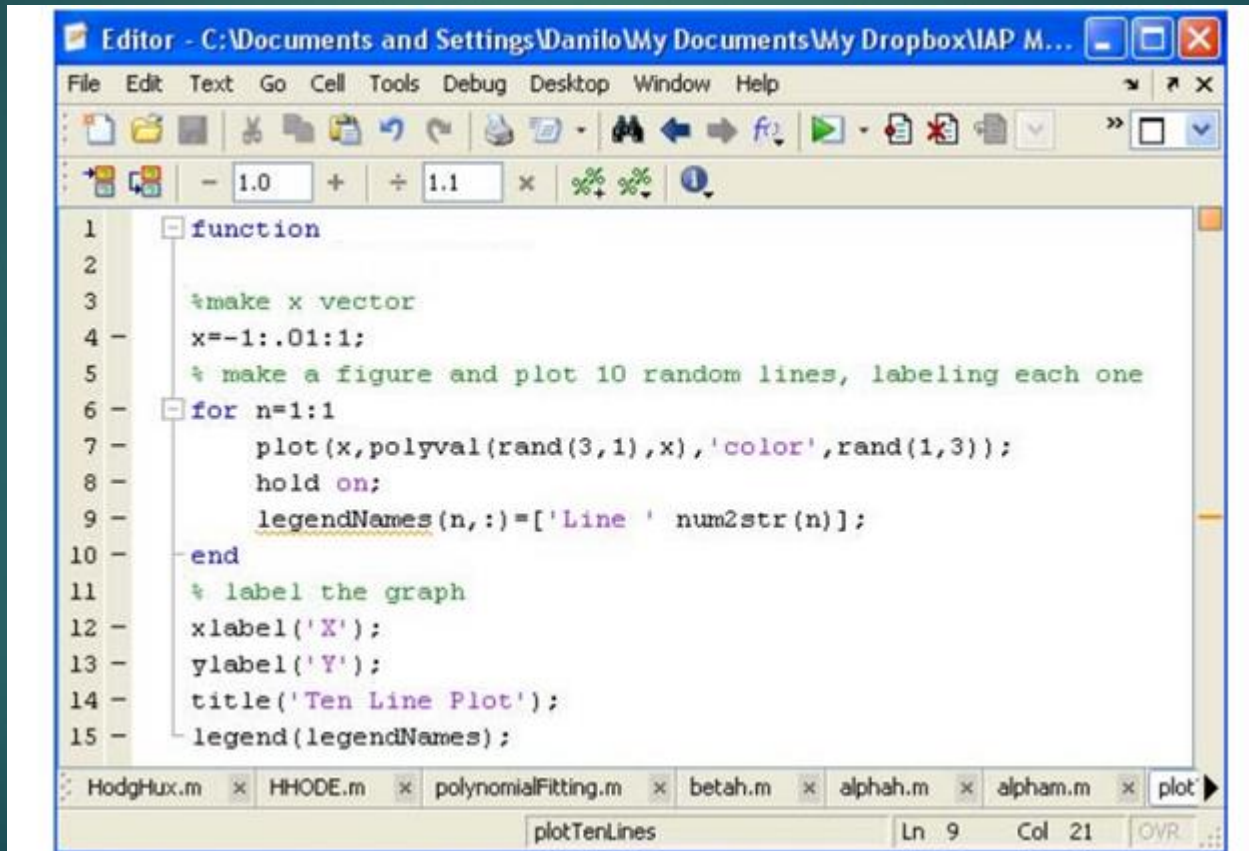
Generated 04-Jan-2006 09:53:26

Number of files called: 19

Filename	File Type	Calls	Total Time	Time Plot
newplot	M-function	1	0.802 s	
gcf	M-function	1	0.460 s	
newplot/ObserveAxesNextPlot	M-subfunction	1	0.291 s	
...matlab/graphics/private/clo	M-function	1	0.251 s	
allchild	M-function	1	0.100 s	
setdiff	M-function	1	0.050 s	

Actividad 2

- ▶ 1.- Usando el debugger, encuentra y modifica los errores en el siguiente código.



```
1 function
2
3 %make x vector
4 x=-1:.01:1;
5 % make a figure and plot 10 random lines, labeling each one
6 for n=1:1
7     plot(x,polyval(rand(3,1),x),'color',rand(1,3));
8     hold on;
9     legendNames(n,:)=['Line ' num2str(n)];
10 end
11 % label the graph
12 xlabel('X');
13 ylabel('Y');
14 title('Ten Line Plot');
15 legend(legendNames);
```

HodgHux.m x HHODE.m x polynomialFitting.m x betah.m x alphah.m x alpham.m x plot

plotTenLines Ln 9 Col 21 OVR

Actividad 2

- ▶ 2.- Corre el código anterior usando el profile viewer.
- ▶ 3- Reporta los resultandos obtenidos.



Algunas monerías de Matlab...

- Se puede utilizar variables simbólicas, mediante el comando sym:

```
» a=sym('1/3');  
» b=sym('4/5');  
» mat=sym([1 2;3 4]);
```

```
» c=sym('c','positive');
```

```
» d=a*b
```

→

d = 4/15

```
» expand((a-c)^2);
```

→

ans = 1/9-2/3*c+c^2

```
» factor(ans)
```

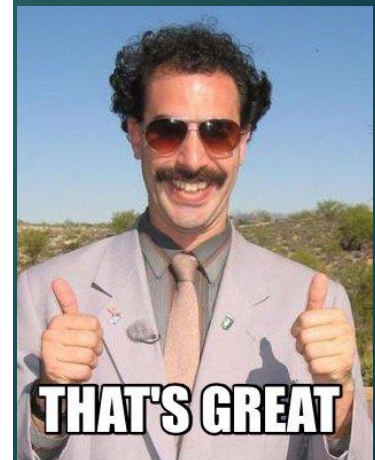
→

ans = 1/9*(3*c-1)^2

```
» matInv=inv(mat)
```

→

ans = [-2, 1] [3/2, -1/2]



Más ejemplos

» `pretty(ans)`

$$\frac{1}{9} - \frac{2}{3}c + c^2$$

» `collect(3*x+4*y-1/3*x^2-x+3/2*y)`

$$\text{ans} = 2x + 11/2y - 1/3x^2$$

» `simplify(cos(x)^2+sin(x)^2)`

$$\text{ans} = 1$$

» `subs('c^2',c,5)`

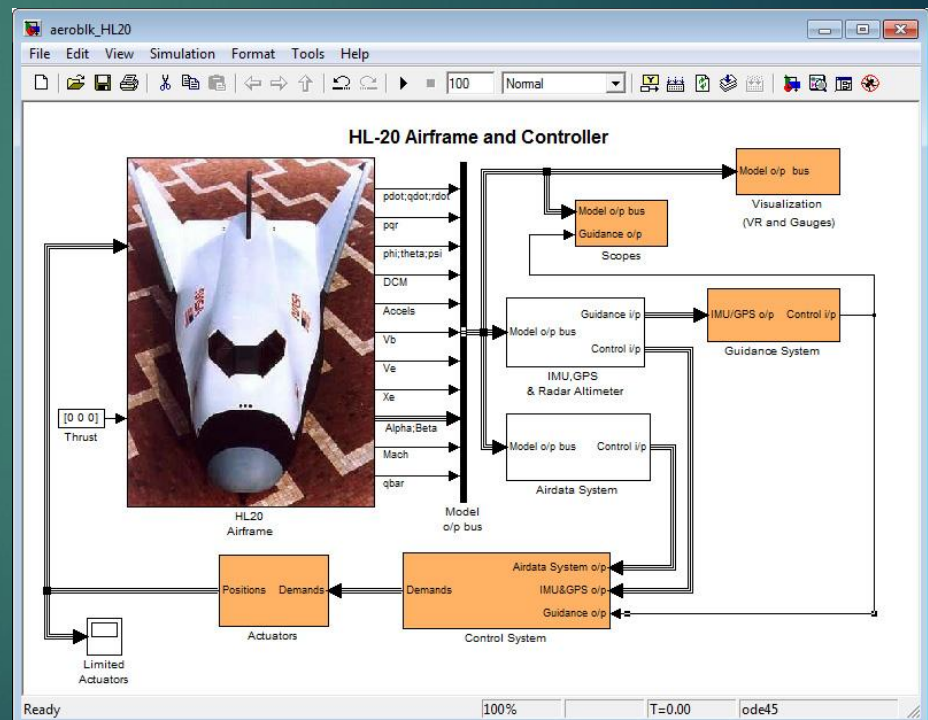
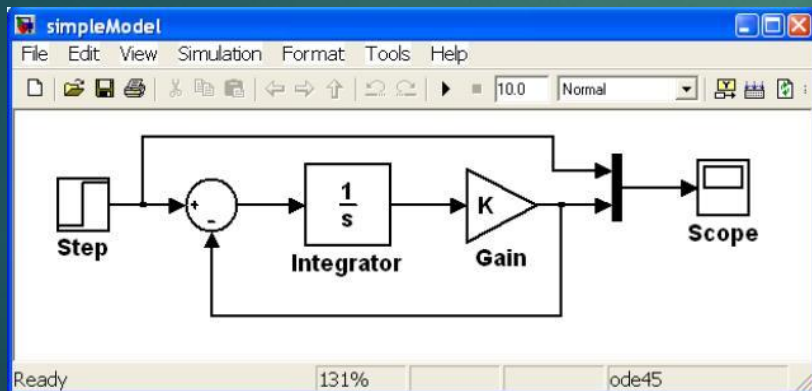
$$\text{ans} = 25$$

» `subs('c^2',c,x/7)`

$$\text{ans} = \frac{1}{49}x^2$$

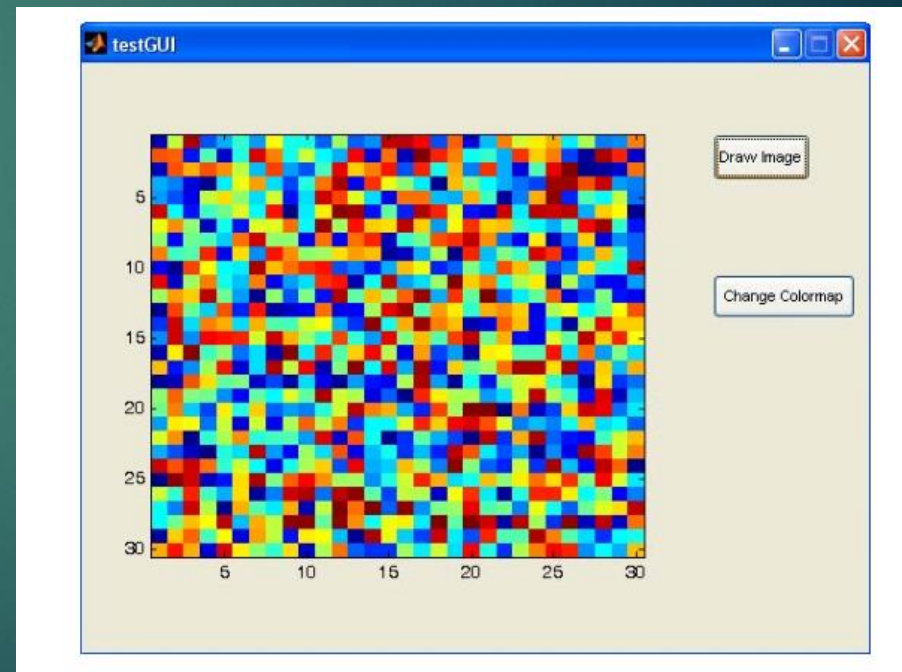
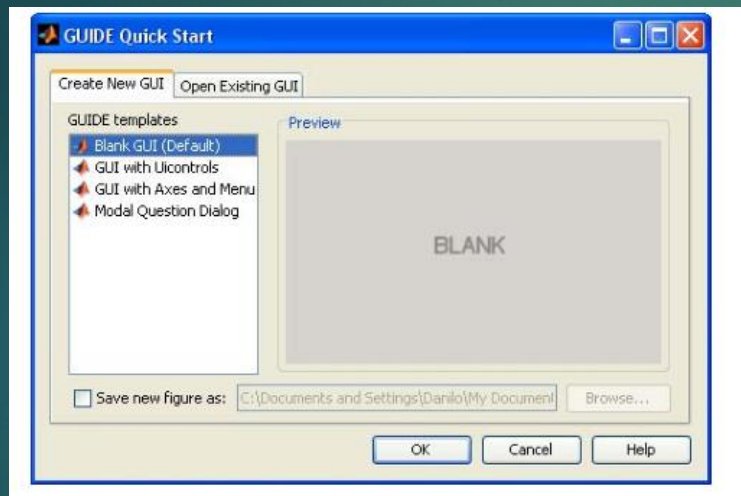
Monería: Simulink

- Ambiente interactivo basado en bloques.



Monería: GUIs

- Para la construcción de interfaces gráficas.



Una monería más: MUPAD

- Es un sistema de álgebra computacional... Bueno... era! ☺

Equation:

$$\begin{aligned} \text{eq} &:= \sqrt{x+3/10} - 9/5*(x+3/10)^5 - (x+3/10)^6 \\ &+ (x-1)/(2*\sqrt{x+3/10}) = 0 \\ \sqrt{x+\frac{3}{10}} - \frac{9\left(x+\frac{3}{10}\right)^5}{5} - \left(x+\frac{3}{10}\right)^6 + \frac{x-1}{2\sqrt{x+\frac{3}{10}}} &= 0 \end{aligned}$$

General solver:

```
sol := solve(eq, x, Real)
{z1^2 - 3/10 | z1 ∈ ((-∞, -√3√10/10) ∪ (√3√10/10, ∞)) ∩ RootOf(z^13 + 9z^11 - 3z^2 + 13/20, z)}
```

DIGITS := 20: float(sol, 20)

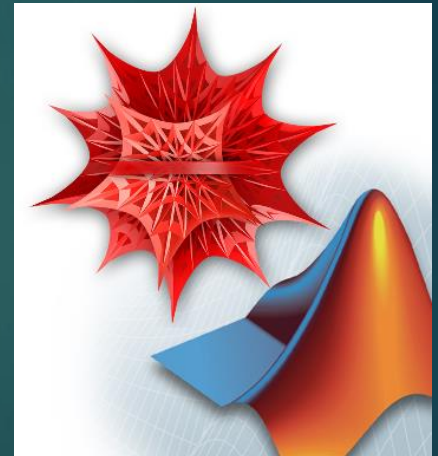
{0.12068186494007759991, 0.15249587894102346284, 0.42846617518653978967}

Numeric Solvers:

```
numeric::solve(eq, x, AllRealRoots)
{0.1524958789, 0.4284661752}
```

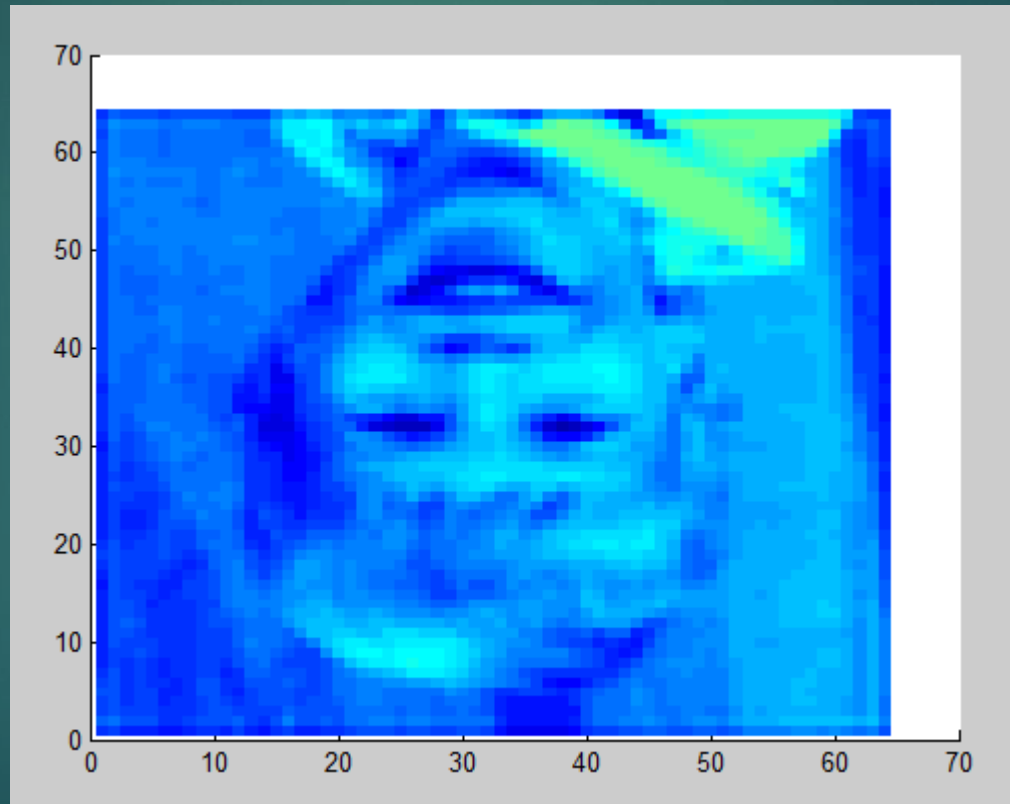
numeric::realroots(eq, x = 0 .. 1)

[[0.1484375, 0.15625], [0.421875, 0.4375]]



Y finalmente...

- Matlab tiene sus “misterios”...



Conclusiones

- ▶ Mis estimados... ahora saben lo más básico de Matlab.
- ▶ Hay mucho por aprender! Pero eso ya dependerá de cada uno de ustedes.



- ▶ Matlab definitivamente es un software muy útil para aspectos científicos!



Nos vemos la siguiente clase!

- ▶ Cuídense mucho!
- ▶ La mejor forma de aprender Matlab:
- ▶ Usar help... y programar, programar y programar!
- ▶ Nos vemos el siguiente miércoles!

