



Modelización y Simulación Computacional de Materiales 2017

Herramientas



Problema Ingenieril o Físico

Preproceso

Preproceso

Post Proceso

“dibujo” del problema

Matricialización

Mediciones Ingenieriles

recinto de validez

Lectura de datos

Información Gráfica

Modelo Físico

Resolución

Interpretación de
resultados

Condiciones de contorno

Escritura de resultados



Problema Ingenieril o Físico

Preproceso

Proceso

Post Proceso

“dibujo” del problema

recinto de validez

Modelo Físico

Condiciones de contorno

Matricialización

Lectura de datos

Resolución

Escritura de resultados

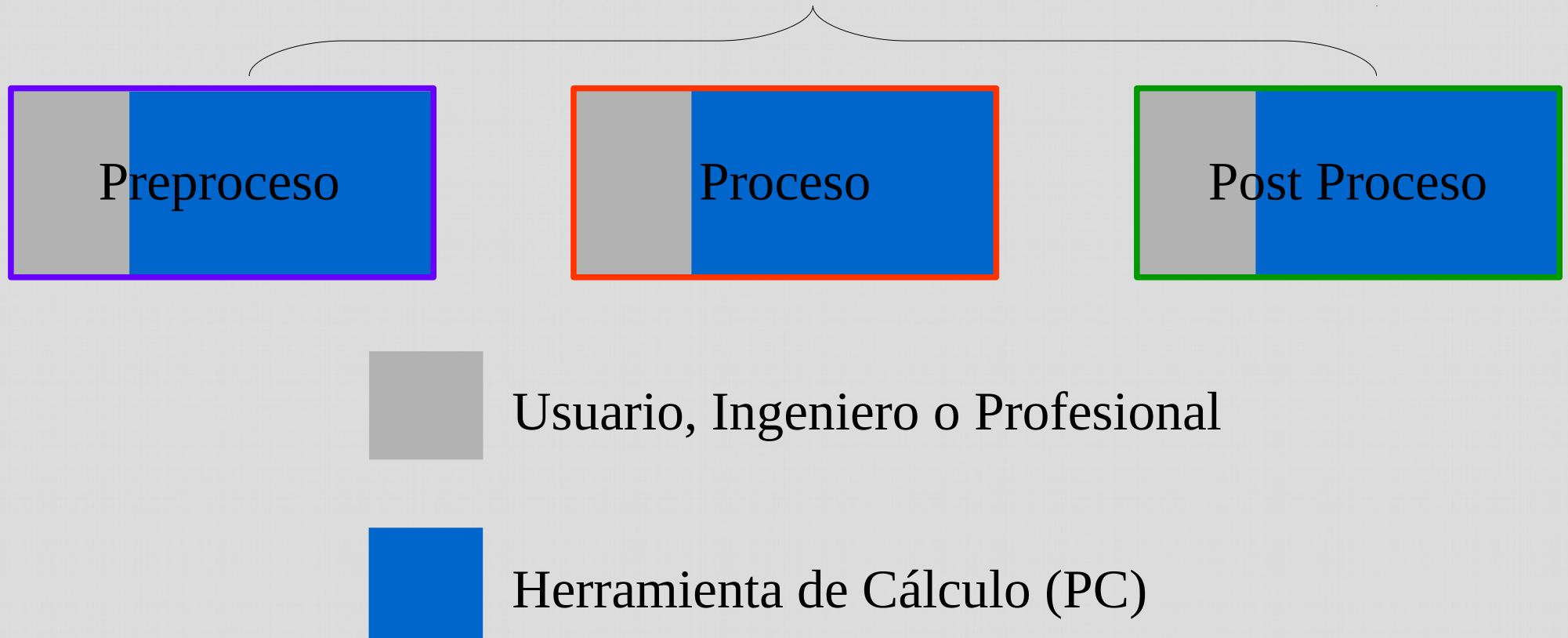
Mediciones Ingenieriles

Información Gráfica

Interpretación de
resultados



Problema Ingenieril o Físico





Uso de Herramientas

Preproceso

Proceso

Post Proceso

Modelo Físico
Condicion de
Contorno

Matricialización

Interpretación de
resultados

Lectura de datos
Resolución
Escritura de
resultados

Mediciones

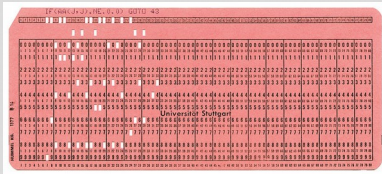
Dibujo
Discretización

Gráficos





Herramientas Alternativas



Fortran



Scilab



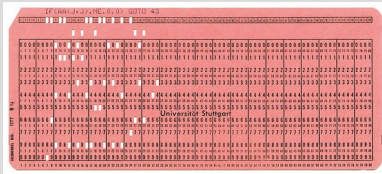
Python / Numpy / SciPy



Octave



Herramientas Alternativas



Fortran



Scilab



Python / Numpy / SciPy

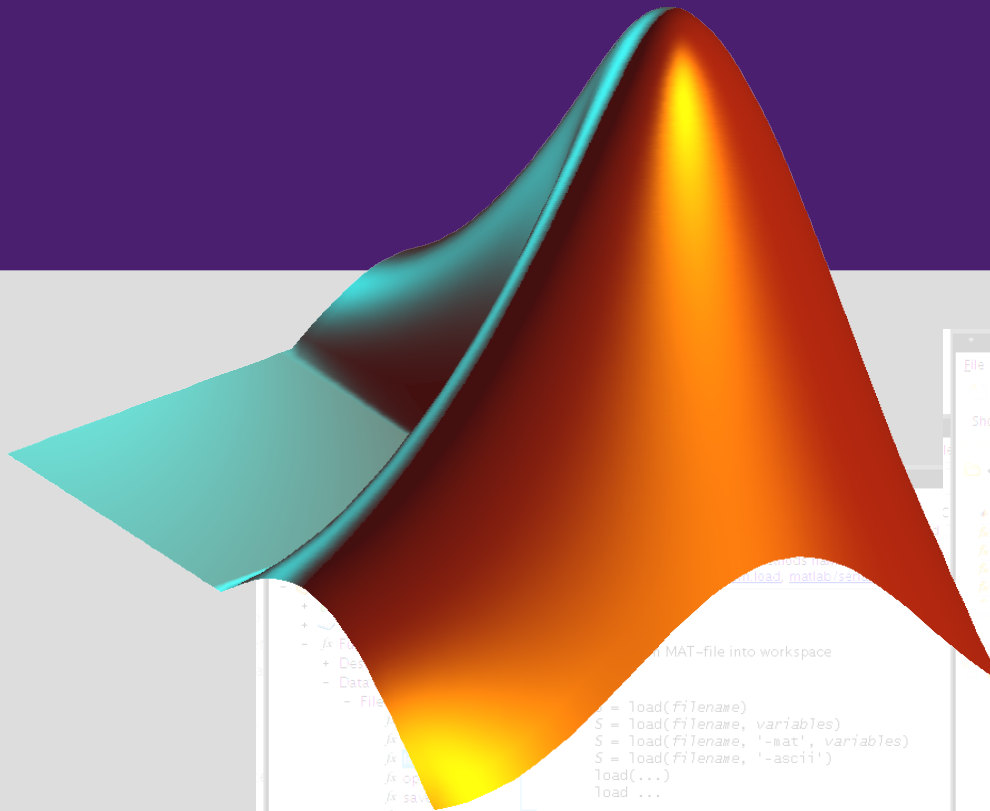


Octave



Modelización y Simulación Computacional de Materiales 2017

Matlab



The MATLAB interface displays a script for loading data from a MAT-file. The script includes comments and function calls for loading variables from a file named 'ccm-19x19-1.dat'.

```
% Load a MAT-file into workspace
% S = load(filename) loads the variables from a MAT-file into the workspace.
% S = load(filename, variables) loads the specified variables from a MAT-file.
% S = load(filename, '-mat', variables) forces load to treat the file as a MAT-file, regardless of the extension.
% S = load(filename, '-ascii') forces load to treat the file as an ASCII file, regardless of the extension.
% load(...) loads without combining MAT-file variables into a single array.

% Load the command form of the command for convenient loading from a file.
S = load(filename)
S = load(filename, variables)
S = load(filename, '-mat', variables)
S = load(filename, '-ascii')
```

Description

`S = load(filename)` loads the variables from a MAT-file into the workspace.

`S = load(filename, variables)` loads the specified variables from a MAT-file.

`S = load(filename, '-mat', variables)` forces load to treat the file as a MAT-file, regardless of the extension.

`S = load(filename, '-ascii')` forces load to treat the file as an ASCII file, regardless of the extension.

`load(...)` loads without combining MAT-file variables into a single array.

`load` is the command form of the command for convenient loading from a file.

The command window shows the following commands:

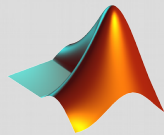
```
>> S=load('ccm-19x19-1.dat');
>> plot(S(:,2),S(:,8));
>> grid on
>> plot(S(:,2),S(:,8),'k');
>> grid on
```

The plot window displays a line graph with the x-axis ranging from 0 to 4000 and the y-axis ranging from 0 to 2. The graph shows a noisy signal that fluctuates around a mean value of approximately 0.5.

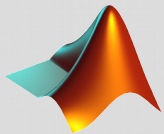


MATLAB

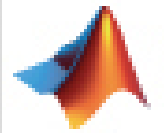
Básicos



MATrix LABoratory

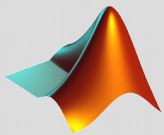


Multiplataforma (JVM)



MathWorks

<http://www.mathworks.com/products/matlab/>

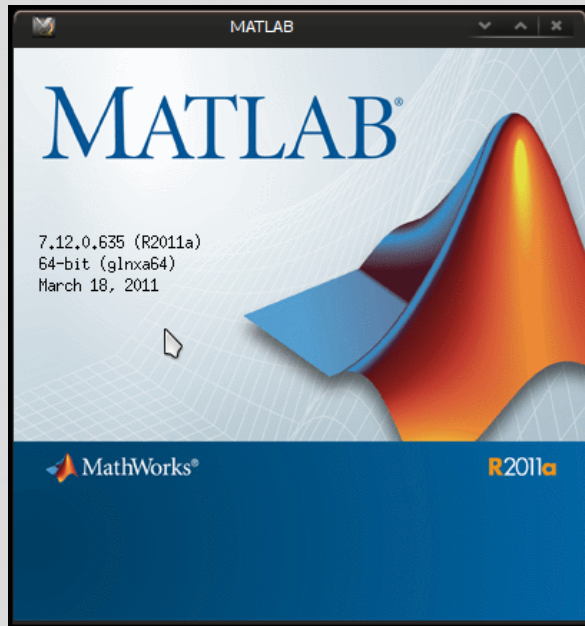


Lenguaje de programación / consola
programable /
Scripts



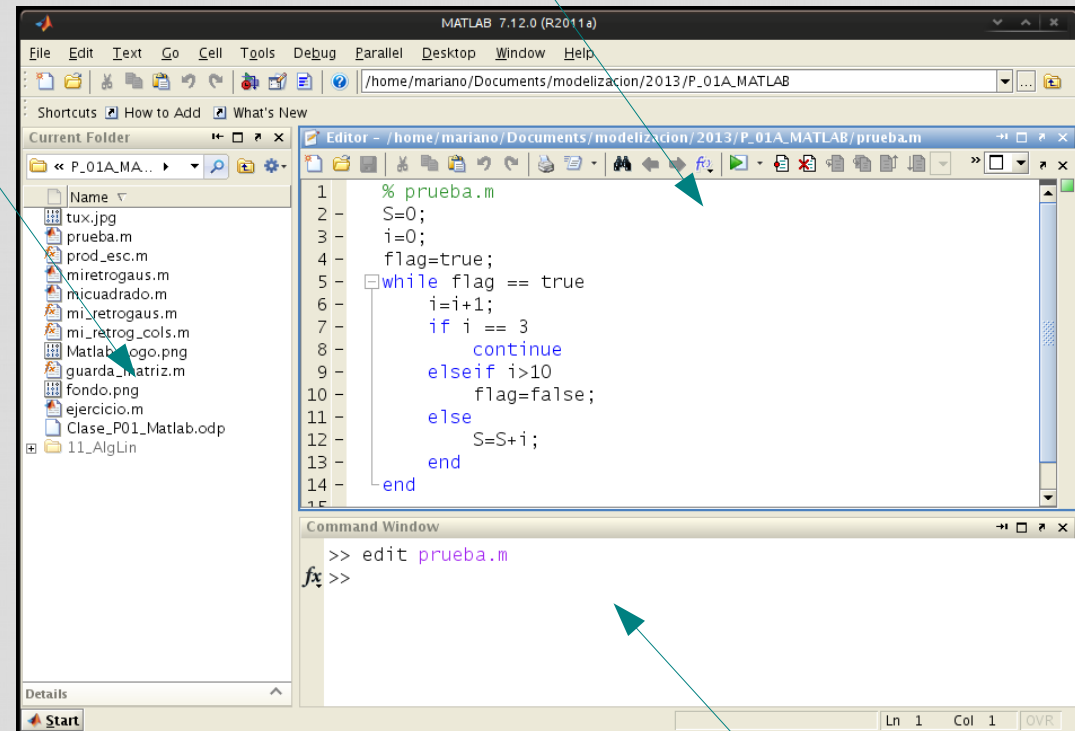
MATLAB

Escritorio - Consola



Panel Lateral

Editor



Ventana de Commandos



MATLAB

Línea de Comandos

Asignación
de variables

Transponer

prompt

```
>> A = [ 1 2 3 ; 4 5 6 ; 7 8 9 ];
```

```
>> A
```

```
A =
```

1	2	3
4	5	6
7	8	9

```
>> A'
```

```
ans =
```

1	4	7
2	5	8
3	6	9

```
>>
```



MATLAB

Indexación

Rango de filas,
Todas las columnas

Vector de índices

```
>> A(1:2,:)
ans =
```

1	2	3
4	5	6

```
>> A([1 3],:)
```

```
ans =
```

1	2	3
7	8	9

```
>> A([1 3],[2 3])
```

```
ans =
```

2	3
8	9

```
>> |
```




MATLAB

Control de Flujo

for

```
Editor - /home/mariano/Documents/modelizacion/2013/P_01A_MATLAB/prueba.m*
1 % Modelizacion de Materiales 2013
2 % Introducción a Herramientas de cálculo.
3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4 S=0;
5 for i = 1:10
6     S=S+i;
7 end
8
9 disp(S)
```

while

```
Editor - /home/mariano/Documents/modelizacion/2013/P_01A_MATLAB/prueba.m
1 % Modelizacion 2013
2 % Introducción a Herramientas de cálculo.
3 S=0;
4 i=0;
5 while i<10
6     i=i+1;
7     S=S+i;
8 end
9 disp(S)
10
```




MATLAB

Control de Flujo

if

```
Editor - /home/mariano/Documents/modelizaci...
1 % prueba.m
2 S=0;
3 i=0;
4 flag=true;
5 while flag == true
6     i=i+1;
7     S=S+i;
8
9     if i == 10
10         flag=false;
11     end
12
13 end
14
15 disp(S);
```

```
Editor - /home/mariano/Documents/modelizaci...
1 % prueba.m
2 S=0;
3 i=0;
4 flag=true;
5 while flag == true
6     i=i+1;
7     S=S+i;
8
9     if i == 10
10         break
11     end
12
13 end
14
15 disp(S);
```

```
Editor - /home/mariano/Docume...
1 % prueba.m
2 S=0;
3 i=0;
4 flag=true;
5 while flag == true
6     i=i+1;
7     if i == 3
8         continue
9     elseif i>10
10         flag=false;
11     else
12         S=S+i;
13     end
14 end
15
16 disp(S);
```

```
Command Window
>> prueba
55

fx >>
```

```
Command Window
>> prueba
52

fx >>
```



MATLAB

Funciones

```
Command Window
>> edit prod_esc.m
fx >> |
```

```
Editor - /home/mariano/Documents/modelizacion/2013/P_01A_MA...
1 % modelizacion de materiales y
2 % procesos 2013
3 % Funciones
4 function [ y , z ] = prod_esc(x1,x2)
5
6 y = x1 * x2';
7
8 norm1 = sqrt( x1*x1' );
9 norm2 = norm(x2);
10
11 z = y / ( norm1 * norm2 );
```

```
Command Window
>> clear all
>> a = [ 1 1 ] ;
>> b = [ 0 1 ] ;
>> [ adotb , cosab ] = prod_esc(a,b)

y =

     1

adotb =

     1

cosab =

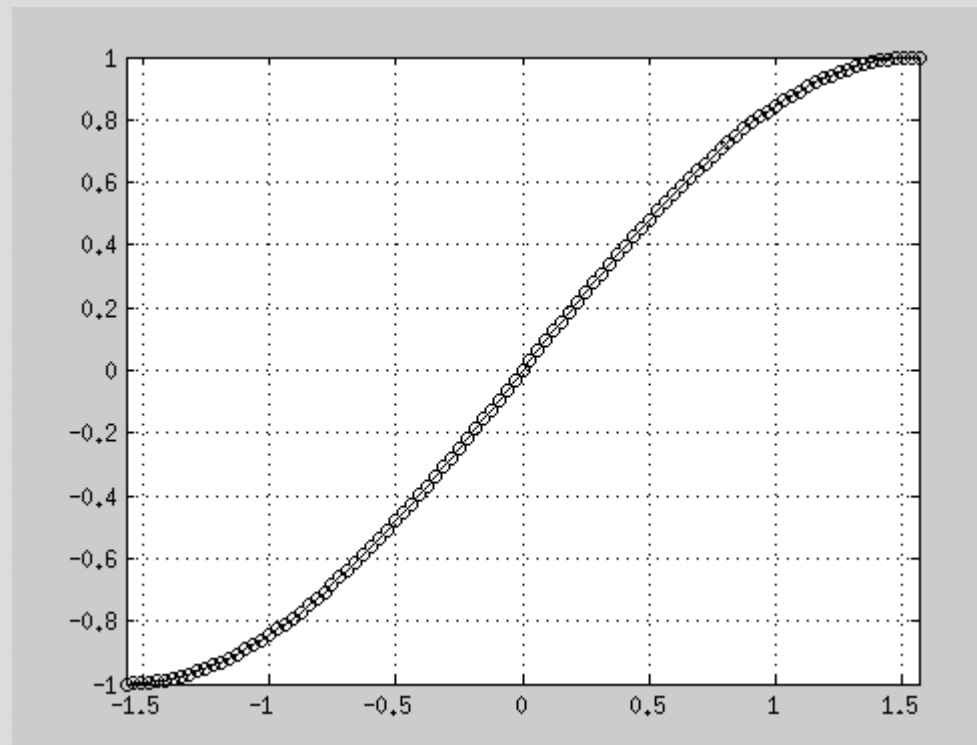
     0.7071
fx >> |
```



MATLAB

Gráficos

```
Command Window
>> x=[ -pi/2 : pi/100 : pi / 2 ] ;
>> y = sin(x);
>> plot(x,y,'o-k');
>> grid on
>> xlim( [ -pi/2 pi/2 ] );
fx >>
```





MATLAB

Entrada - Salida

Abrir archivo,
reescribir o agregar:

Guardar un tipo de
Datos en el medio de
Una linea:

Concatenar

Imprimir en archivo
Con retorno de carro!

```
1 %Modelizacion de Materiales y Procesos 2013
2 function fid=guarda_matriz(filename,step,matriz)
3
4 if step == 1
5     fid = fopen(filename,'w');
6 else
7     fid = fopen(filename,'a');
8 end
9 line='';
10
11 [ n m ]=size(matriz);
12
13 fprintf(fid,'\n \n Matriz de %d x %d en el paso %d \n',n,n,step);
14 fprintf(fid,'===== \n');
15
16 for i = 1:n
17     for j=1:m
18         line=[line,num2str(matriz(i,j),' %10.6f ')];
19     end
20     fprintf(fid,[line,'\n']);
21 end
22 fclose(fid);
23 return;
24
```



MATLAB

Documentación

```
Editor - prod_escm Command Window
>> help quiver
QUIVER Quiver plot.
QUIVER(X,Y,U,V) plots velocity vectors as arrows with components (u,v)
at the points (x,y). The matrices X,Y,U,V must all be the same size
and contain corresponding position and velocity components (X and Y
can also be vectors to specify a uniform grid). QUIVER automatically
scales the arrows to fit within the grid.

QUIVER(U,V) plots velocity vectors at equally spaced points in
the x-y plane.

QUIVER(U,V,S) or QUIVER(X,Y,U,V,S) automatically scales the
arrows to fit within the grid and then stretches them by S. Use
S=0 to plot the arrows without the automatic scaling.

QUIVER(...,LINESPEC) uses the plot linestyle specified for
the velocity vectors. Any marker in LINESPEC is drawn at the base
instead of an arrow on the tip. Use a marker of '.' to specify
no marker at all. See PLOT for other possibilities.

QUIVER(...,'filled') fills any markers specified.

QUIVER(AX,...) plots into AX instead of GCA.

H = QUIVER(...) returns a quivergroup handle.

Example:
[x,y] = meshgrid(-2:.2:2,-1:.15:1);
z = x .* exp(-x.^2 - y.^2); [px,py] = gradient(z,.2,.15);
contour(x,y,z), hold on
quiver(x,y,px,py), hold off, axis image

See also feather, quiver3, plot.

Reference page in Help browser
doc quiver

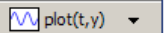
fx >> |
```

```
Command Window
>> doc quiver
fx >>
```

The screenshot shows the MATLAB Help browser interface. The left pane displays the 'Contents' tree with 'MATLAB' expanded, showing various categories like 'Getting Started', 'User's Guide', 'Functions', etc. The right pane shows the documentation for the 'quiver' function, which is part of the 'Specialized Plotting' category. The documentation includes a description of the function, a list of alternatives, the syntax for using the function, and a description of the function's behavior.

quiver
Quiver or velocity plot

Alternatives

To graph selected variables, use the Plot Selector  in the Workspace Browser, or use the Figure Palette Plot Catalog. Manipulate graphs in *plot edit* mode with the Property Editor. For details, see [Plotting Tools — Interactive Plotting](#) in the MATLAB Graphics documentation and [Creating Graphics from the Workspace Browser](#) in the MATLAB Desktop Tools documentation.

Syntax

```
quiver(x,y,u,v)
quiver(u,v)
quiver(...,scale)
quiver(...,LineStyle)
quiver(...,LineStyle,'filled')
quiver(...,'PropertyName',PropertyValue,...)
quiver(axes_handle,...)
h = quiver(...)
```

Description



MATLAB

Ejercicio

