scilab enterprises

Scilab para principiantes

Este documento ha sido redactado conjuntamente por Scilab Enterprises y Christine Gomez, profesora de matemáticas del Lycée Descartes de Antony, Hauts-de-Seine, en 2013.
Fue actualizada por Claude Gómez en enero de 2025.

Índice

Introducción

Acerca de este documento 3

Instalación de Scilab 3

Lista de correo e información 3

Recursos adicionales 3

Capítulo 1 - Conocer Scilab

El entorno general y la consola 4

Cálculos numéricos sencillos 5

La barra de menús 6

El editor 7

Ventana gráfica 8

Ayuda en línea 9

Gestión de ventanas y personalización del espacio de trabajo 10

Capítulo 2 - Programación

Variables, asignación y visualización 11

Bucles 14

Pruebas 16

Parcelas en 2 y 3 dimensiones 17

Más información sobre matrices y vectores 22

Algunas funciones útiles 26

Problemas de precisión 28

Resolución de ecuaciones diferenciales 29

Capítulo 3 - Funciones útiles de Scilab

Para el análisis
31
Para probabilidad y estadística
31
Para visualizar y trazar
32
Utilidades
32

Introducción

Acerca de este documento

El objetivo de este documento es guiarle paso a paso a través de las distintas funciones básicas del software Scilab para un usuario que nunca antes haya utilizado un software de cálculo. Esta presentación se limita deliberadamente a lo esencial para facilitar la familiarización con Scilab.

Los cálculos, gráficos e ilustraciones están realizados con Scilab 2025.0.0. Por lo tanto, puede reproducir todos los comandos presentados a partir de esta versión.

Instalación de Scilab

Scilab es un programa de cálculo numérico que cualquiera puede descargar gratuitamente. Disponible para Windows, Linux y MacOS, Scilab puede descargarse desde la siguiente dirección: http://www.scilab.org/

Lista de correo e información

Para facilitar los intercambios entre los usuarios de Scilab, se han creado listas de correo para ellos (lista francesa, lista para el mundo de la educación, lista internacional en inglés). El principio es sencillo: los suscriptores pueden comunicarse entre sí por correo electrónico (preguntas, respuestas, intercambio de documentos, comentarios, etc.).

Para consultar las listas disponibles e inscribirse, visite https://www.scilab.org/about/community/mailing-lists.

Recursos adicionales

Puede acceder al sitio web de Scilab para obtener tutoriales sobre el uso de Scilab en https://www.scilab.org/tutorials.

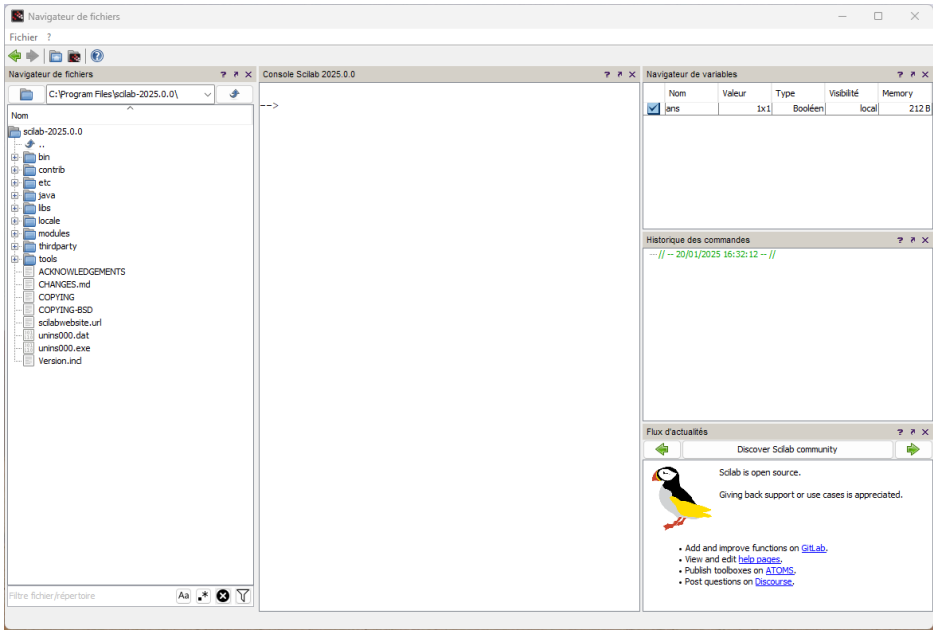
Capítulo 1 - Conocer Scilab

El espacio de trabajo útil en Scilab se compone de varias ventanas:

- La consola para hacer cálculos,
- El editor para escribir programas,
- Ventanas gráficas para mostrar gráficos,
- Ayuda.

El entorno general y la consola

Tras hacer doble clic en el icono de Scilab para iniciar el software, el entorno Scilab predeterminado muestra las siguientes ventanas acopladas: consola, exploradores de archivos y variables, historial de comandos (véase "Gestión de ventanas y personalización del espacio de trabajo", página 10):



En la consola, después del símbolo del sistema "**-->**", basta con escribir un comando y pulsar Intro (Windows y Linux) o Retorno (MacOS) en el teclado para obtener el resultado correspondiente.

```
--> 57/4
 años =
    14.25
--> (2+9)^5
 años =
    161051.
```

Tenga en cuenta que
Delante del resultado, **ans** significa

Puedes volver atrás en cualquier momento, utilizando las flechas del teclado ← ↑ → ↓ o el ratón, con las teclas izquierda y derecha que te permiten modificar instrucciones, y las teclas arriba y abajo que te dan la opción de volver a un comando ejecutado anteriormente.

Cálculos numéricos sencillos

Todos los cálculos realizados por Scilab son numéricos. Scilab calcula con matrices (ver Capítulo 2, página 22).

+^Las operaciones se escriben como " " para la suma, "**−**" para la resta, "*****" para la multiplicación, "**/**" para la división y " " para los exponentes. El punto decimal en los números decimales se marca con un punto. Por ejemplo :

```
-->2+3.4
 años =
    5.4
```

Es importante distinguir entre mayúsculas y minúsculas para que los cálculos funcionen correctamente. Por ejemplo, con el comando **sqrt**, que calcula la raíz cuadrada :

```
-->sqrt(9)
 años =
    3.
```

mientras que :

```
-->SQRT(9)
Variable no definida: SQRT
```

Números especiales

%e y **%pi** representan e y π respectivamente:

```
--> %e
 %e =
    2.7182818
```

```
--> %pi
 %pi =
    3.1415927
```

%i representa la variable compleja **i** en la entrada y muestra **i** en la salida:

```
--> 2+3*%i
 años =
    2. + 3.i
```

Para no mostrar el resultado

Si se añade un punto y coma ";" al final de una línea de comandos, se realiza el cálculo pero no se muestra el resultado.

```
-->(1+sqrt(5))/2;
```

```
--> (1+sqrt(5))/2
 años =
    1.6180340
```

Para recordar el nombre de una función

Los nombres de las principales funciones se resumen en el capítulo 3 de este documento (página 31). Por ejemplo :

```
--> exp(10)/factorial(10)
 años =
    0.0060699
```

Atención
Las funciones disponibles también se enumeran en la sección de ayuda, a la que se

Puede utilizar el tabulador → del teclado para completar el nombre de una función o variable de la que haya dado las primeras letras.

Por ejemplo, si escribe :

```
-->Hecho
```

y pulsas la tecla tabulador, aparece una pequeña ventana que te permite elegir entre todas las funciones y nombres de variables que empiezan por **`fact`**, como **`factorial`** y **`factor`**. Basta con hacer doble clic en la función deseada o seleccionarla con el ratón o las teclas ↑ ↓ y pulsar Intro (Windows y Linux) o Retorno (MacOS) para insertarla.

La barra de menús

Utilizará, en particular, los menús que se indican a continuación.

Aplicaciones

- El historial de órdenes le permite encontrar todas las órdenes de sesiones anteriores y de la sesión actual.
- El navegador de variables permite encontrar todas las variables utilizadas anteriormente durante la misma sesión.

Edición

Preferencias (en el menú **Scilab** en MacOS) le permite establecer y personalizar los colores, fuentes y tamaños de fuente en la consola y en el editor, lo cual es muy útil cuando se proyecta en una pantalla.

Haga clic en **Borrar consola** para borrar todo el contenido de la consola. En este caso, el historial se conserva y los cálculos realizados durante la sesión permanecen en la memoria. Siempre se puede volver a un comando que se ha borrado utilizando las teclas de flecha del teclado.

Controlar

Para interrumpir un programa en ejecución, puede :

- **`Pause`** el programa o haga clic en **Control > Pausa** en la barra de menús (Ctrl X en Windows y Linux o Comando X en MacOS), si el programa ya se está ejecutando. En todos los casos, el símbolo del sistema "**`-->`**" cambiará a "**`-1->`**", luego a "**`-2->`**"... si se repite la operación.
- Para volver al momento en que se interrumpió el programa, escriba **`reanudar`** en la consola o haga clic en **Control > Reanudar**.
- Para detener definitivamente un cálculo sin posibilidad de retorno, escriba **`abortar`** en la consola o haga clic en **Control > Abortar** en la barra de menús.

El editor

Escribir directamente en la consola tiene dos desventajas: no es posible guardar, y si se han escrito varias líneas de instrucciones, las modificaciones no son fáciles. Para enlazar varias instrucciones, el editor es la herramienta adecuada.

Abrir el editor

 Para abrir el editor desde la consola, haga clic en el primer icono de la barra de herramientas o en **Aplicaciones > SciNotes** en la barra de menús.

El editor se abre con un archivo por defecto llamado "**`Sin título 1`**".

Escribir en el editor

Escriba el texto en el editor como lo haría en cualquier procesador de textos.

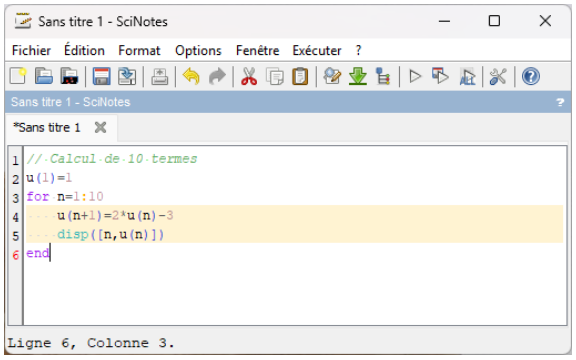
En el editor de texto, los corchetes de apertura y cierre, los comandos de fin de bucle, de función y de prueba aparecen automáticamente. Sin embargo, puede desactivar estas dos funciones en el menú **Opciones > Autocompletar**, haciendo clic en las dos entradas siguientes que están activadas por defecto:

- **(,[,...**
- **si,función,...**

En principio, debe ir a la línea siguiente a cada instrucción, pero es posible escribir varias instrucciones en la misma línea, separándolas con un punto y coma "`;`".

Cuando se inicia un bucle o una prueba, se aplica automáticamente un desplazamiento de inicio de línea denominado sangría.

(u_n)En el siguiente ejemplo, calculamos 10 términos de la secuencia definida por : $\{u_1=1 u_{n+1}=2u_n-3$



Atención
- Para escribir comentarios que no se tendrán en cuenta en los cálculos, precédalos con "`//`".
- Para cambiar el tipo de letra, haz clic en **Opciones > Preferencias**.
- Cuando escribes un programa, la sangría es automática. Si no es así, haz clic en **Formato > Corregir sangría**

Regístrese en

Puede guardar cualquier archivo haciendo clic en **Archivo > Guardar como**.

La extensión ".sce" al final del nombre del archivo iniciará automáticamente Scilab cuando se abra el archivo (excepto en Linux y MacOS).

Copiar en la consola, ejecutar el programa

Al hacer clic en Ejecutar en la barra de menús, se le ofrecen tres opciones:

- Ejecutar "**...archivo sin eco**" (Ctrl Mayús E en Windows y Linux, Cmd Mayús E en MacOS): el archivo se ejecuta sin que el programa escriba en la consola (habiendo guardado antes el archivo).
- Ejecutar **"...archivo con echo**" (Ctrl L en Windows y Linux, Cmd L en MacOS): reescribe el archivo en la consola y lo ejecuta.
- Ejecutar "**...al cursor, con eco**" (Ctrl E en Windows y Linux, Cmd E en MacOS): reescribe la selección elegida con el ratón en la consola y la ejecuta, o ejecuta los datos del fichero hasta la posición del cursor definida por el usuario.

También puedes utilizar el clásico copiar/pegar.

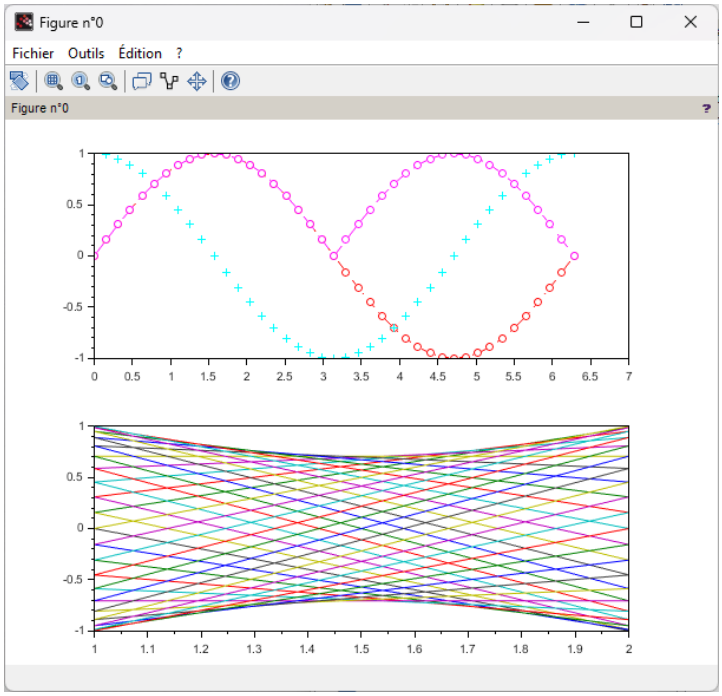
La ventana gráfica

Abrir una ventana gráfica

Se abre una ventana de gráficos para cualquier trazado. Puede trazar curvas, superficies y nubes de puntos (véase el Capítulo 2, página 17).

Se puede obtener un ejemplo de la curva tecleando :

```
-->trazar
```



Atención
- Para borrar la pista anterior, escriba ("figura clara" en inglés).
- Para abrir otra ventana gráfica, escriba **`scf;`** ("establecer figura actual").
Si tiene abiertas varias ventanas de gráficos, puede elegir en cuál desea trazar escribiendo **`scf(n)`**; donde n es el número de la ventana

Modificar una ruta

Utilice la lupa para ampliar la vista. Para ampliar en dos dimensiones, haga clic en el icono y cree con el ratón un rectángulo que formará la nueva vista ampliada. Para ampliar en tres dimensiones, haga clic en el icono y cree el paralelepípedo que formará la nueva vista ampliada. También puede hacer zoom utilizando la rueda de desplazamiento del ratón. Para volver a la pantalla inicial, pulse sobre la otra lupa .

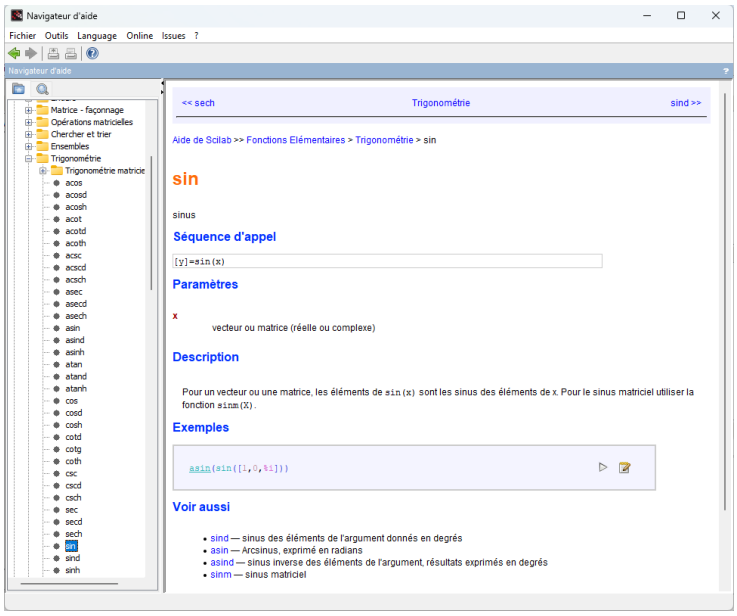
 El icono se puede utilizar para girar la figura (especialmente útil en 3D) con acciones del botón derecho del ratón que se guiarán por mensajes en la parte inferior de la ventana.

Para modificaciones más precisas, haga clic en **Edición > Propiedades de la figura** o en **Edición > Propiedades del eje.** Se abrirá una ventana llamada **`Editor gráfico`** y podrá seguir las instrucciones (esta opción aún no está disponible en MacOS).

Ayuda en línea

Para acceder a la ayuda en línea, haga clic en **? > Ayuda de Scilab** en la barra de menús, o escriba :

```
-->doc
```



Atención
Parte de la ayuda está disponible en francés, el resto en inglés.
Los ejemplos de uso pueden ejecutarse en Scilab y editarse en SciNotes utilizando los botones asociados en el

Para obtener ayuda sobre las funciones, escriba **`doc`** (antes **`help`**) y el nombre de la función que necesite en la consola. Por ejemplo, escriba :

```
>doc sin
```

mostrará la función **`sen`** (seno).

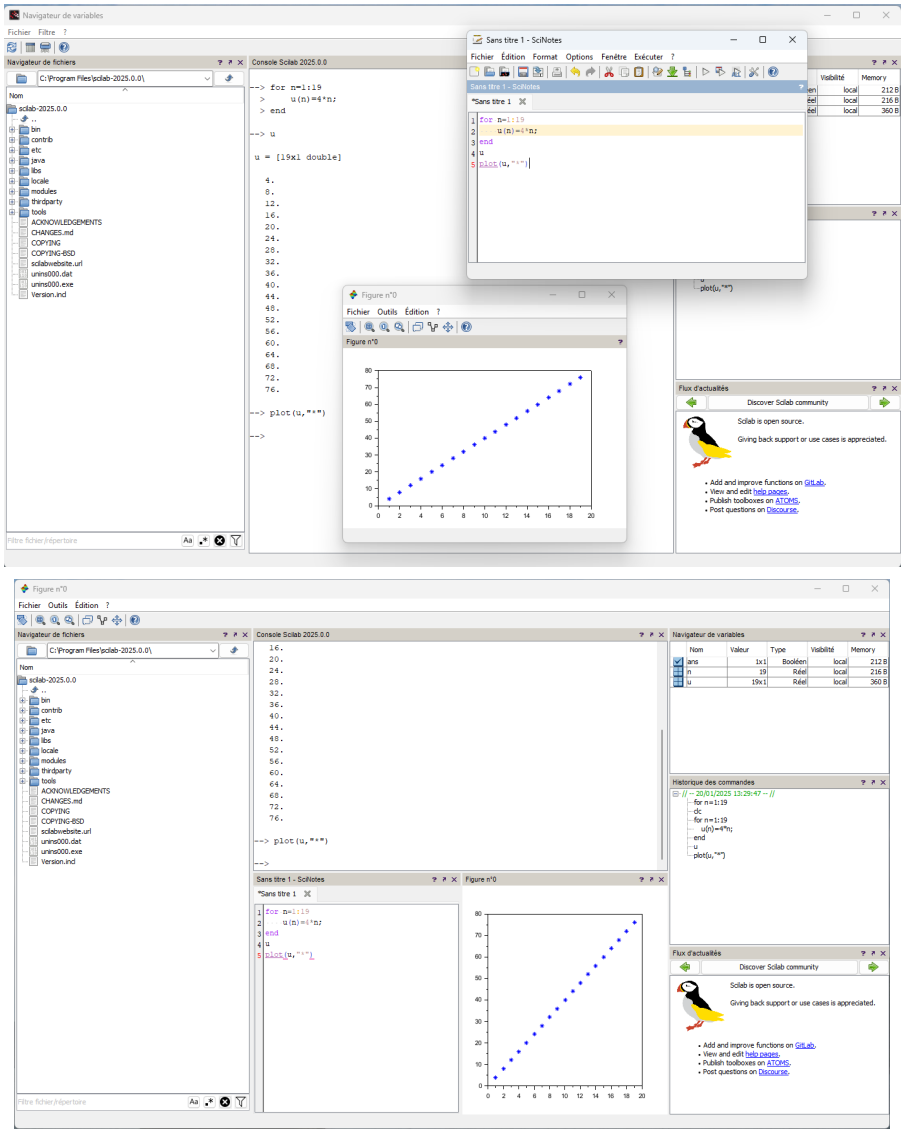
Gestión de ventanas y personalización del espacio de trabajo

Al igual que con el entorno por defecto de Scilab, que incluye las ventanas de la consola, los exploradores de archivos y variables y el historial de comandos, todas las demás ventanas de Scilab pueden reposicionarse en una sola ventana. Por ejemplo, el usuario puede elegir colocar el editor en el entorno por defecto de Scilab.

Para colocar una ventana dentro de otra, busque primero la barra horizontal azul en Windows, o la barra negra en MacOS y Linux, situada en la parte superior de la ventana, debajo de la barra de herramientas y que contiene un signo de interrogación a la derecha.

- En Windows y Linux, haga clic en esta barra con el botón izquierdo del ratón y, manteniendo pulsado el botón, mueva la flecha del ratón hasta la ventana deseada.
- En MacOS, haga clic en esta barra y, manteniendo pulsado el botón del ratón, arrástrela hasta la ventana deseada.

Aparece un rectángulo que indica la posición futura de la ventana. Cuando la posición sea la deseada, suelte el botón del ratón. Para cancelar y mostrar la ventana, haga clic en la flecha pequeña situada a la derecha de la misma barra.



Capítulo 2 - Programación

En los ejemplos dados en este documento, cualquier línea precedida por "-->" es un comando; las otras líneas son retornos (resultados de cálculos, mensajes de error, etc.). No escriba "-->" en el editor. Sólo lo hemos introducido para diferenciar las líneas de comando de los retornos de cálculo, que se muestran en la consola tras una

operación de copiar/pegar. Presentados en una tabla (sin "-->" y sin retornos de cálculo), los comandos se muestran tal y como se escribirían en el editor.

Variables, asignación y visualización

Variables

Scilab no es un software de cálculo formal. Sólo calcula con números. En realidad, todos los cálculos se hacen con matrices, pero esto puede pasar desapercibido. Aunque no se conozca la noción de matriz, utilizamos vectores y secuencias de números que son, de hecho, matrices de $1 \times n$ o $n \times 1$, del mismo modo que un número es una matriz de dimensión 1×1.

No es necesario declarar previamente las variables, pero toda variable debe tener un valor. Por ejemplo, solicitar el valor de la variable **a** sin haberle dado un valor de antemano produce un error :

```
-->a
Variable no definida: a
```

Si asignamos un valor a la variable **a**, ya no hay errores:

```
--> a=%pi/4
 a =
    0.7853982
--> a
 a =
    0.7853982
```

Se puede utilizar cualquier nombre de variable que no esté ya definido por el sistema:

```
--> Pisur2=%pi/2
 Pisur2 =
    1.5707963
```

Tenga en cuenta que
Al igual que las funciones Scilab, los nombres de las variables no deben contener

Si el resultado de un cálculo no se asigna a una variable, el valor se asigna automáticamente a la variable llamada **ans** (respuesta) :

```
-->3*(4-2)
 años =
    6.

-->años
 años =
    6.
```

Las funciones

Las funciones son la forma más sencilla y natural de realizar cálculos utilizando variables y obtener resultados a partir de ellas.

La definición de una función comienza con **function** y termina con **endfunction**. Por ejemplo, para transformar euros (e) en dólares (d) con un tipo de cambio (t), definimos la función **dólares**. Las variables son **e** y **t** y la imagen es **d**.

```
-->función d=dólares(e,t); d=e*t; endfunción

-->dólares(200,1.04)
 años =
    208.
```

Lo más frecuente es utilizar funciones numéricas de una sola variable. Por ejemplo, dos funciones **f** y **g** se definen utilizando los comandos siguientes:

```
-->función y=f(x); y=36/(8+exp(-x)); endfunción

-->función y=g(x); y=4*x/9+4; endfunción
```

Tenga en cuenta que
Las variables **x** e **y** son variables ficticias, por lo que sus nombres pueden reutilizarse en la

Una vez definidas las funciones, pueden utilizarse para calcular valores:

```
--> f(10)
 años =
    4.4999745
--> g(12.5)
 años =
    9.5555556
```

Solicitar la asignación de un valor

=La asignación es sencilla, utilizando el símbolo " ".

La pantalla

Escriba a

Al escribir el nombre de una variable se muestra su valor, excepto con "**;**" al final del comando.

Los ganchos

Los corchetes se utilizan para definir matrices (ver página 22). Como se dijo anteriormente, el cálculo de matrices es la base de todos los cálculos realizados en Scilab. Se utiliza un espacio o una coma para pasar de una columna a la siguiente, y un punto y coma para pasar de una fila a la siguiente.

Definir un vector columna y obtener una visualización columna :

```
-->v=[3;-2;5]
 v = [3x1 doble]
    3.
  - 2.
    5.
```

Para definir un vector de línea y obtener una visualización de línea :

```
-->v=[3 -2 5]
 v = [1x3 doble]
    3.  - 2. 5.
```

Tenga en cuenta que
También es posible escribir este comando

Para definir una matriz y visualizar una tabla :

```
-->m=[1 2 3;4 5 6;7 8 9]
 m = [3x3 doble]
    1. 2. 3.
    4. 5. 6.
    7. 8. 9.
```

Tenga en cuenta que
También puede escribir este comando de la forma:
m=[1,2,3;4,5,6;7,8,9]

La disp

La función **disp** siempre va seguida de corchetes.

Con el vector anterior **v** :

```
-->v(2)
 años =
  - 2.


-->disp(v(2))
  - 2.
```

Para mostrar una cadena de caracteres (normalmente una frase), enciérrela entre comillas :

```
-->disp("Bob ha ganado")
  Bob ganó
```

+Para mezclar palabras y valores, utilice el comando **cadena**, que transforma los valores en caracteres, y " " entre las distintas partes:

```
-->d=500;


-->disp("Bob ganó "+string(d)+"dólares")
 Bob ganó 500 dólares
```

Si la frase contiene un apóstrofo, debe duplicarse en la cadena de caracteres para que se muestre correctamente:

```
-->disp("Así es")
 Es que
```

Los bucles

Incremento de

El operador "**:**" se utiliza para definir vectores de números cuyas coordenadas están en secuencia aritmética. Se da "el primer valor: el paso: el último valor" (no necesariamente alcanzado). Si no se da el paso, su valor por defecto es 1.

Por ejemplo, para definir un vector de filas de números enteros que se incrementan en 1 entre 3 y 10 :

```
-->3:10
 ans = [1x8 doble]
    3. 4. 5. 6. 7. 8. 9. 10.
```

o incrementar en 2 entre 1 y 10 :

```
-->1:2:10
 ans = [1x5 doble]
```

```
    1. 3. 5. 7. 9.
```

o disminuir por 4 entre 20 y 2 :

```
-->u=20:-4:2
 u = [1x5 doble]
    20. 16. 12. 8. 4.
```

Para

La estructura de bucle más sencilla para un número conocido de iteraciones es **`for ... end`**, que significa "Para ... fin de for".

Ejemplo: Cálculo de 20 términos de una sucesión definida por recurrencia como : $\{u_1=4\, u_{n+1}=u_n+2n+3$

Algoritmo	Editor Scilab

Pon 4 en u(1)
Para n de 1 a 20
u(n+1) toma el valor u(n)+2n+3
Mostrar n y u(n)
Fin de para
	`u(1)=4;`
	`para n=1:20`
	`u(n+1)=u(n)+2*n+3;`
	`disp([n,u(n)])`
	`fin`

En

Si desea que el bucle se detenga cuando se alcance un objetivo determinado, utilice la forma **`while ... end`**, que significa "mientras ... fin de mientras".

Ejemplo: En 2025 replanté un árbol de Navidad de 1,20 m de altura. Crece 30 cm al año. He decidido cortarlo cuando supere los 7 m. ¿En qué año cortaré el árbol?

Algoritmo	Editor Scilab

Añada 1,2 a h (h = altura del árbol)
Pon 2025 en a (a = año)
Mientras h<7
h toma el valor h+0,3 (mi árbol está creciendo)
a toma el valor a+1 (pasa un año)
Fin de mientras
Mostrar a (el último año)
	`h=1.2;`
	`a=2025;`
	`mientras h<7`
	`h=h+0,3;`
	`a=a+1;`
	`fin`
	`disp("Cortaré mi árbol en...`
	`+string(a))`

Atención
Cuando un comando es demasiado largo para escribirlo en una sola línea, el editor pasa automáticamente a la línea siguiente. También puede poner " .. (dos puntos) antes de pasar a la línea

Las pruebas

Operadores de comparación

Comparar números o comprobar si una afirmación es verdadera o falsa son pruebas útiles. Éstos son los comandos correspondientes:

Igualdad	Diferentes	Baja	Superior	Inferior o igual a	Mayor o igual que
==	**<>**	**<**	**>**	**<=**	**=>**
Verdadero	Falso	Y	O	No	
%T	**%F**	**&**	**|**	**~**	

Tenga en cuenta lo siguiente
Tenga cuidado con la precisión. **==**Como los cálculos son aproximados, el test " " da a veces respuestas erróneas (ver problemas de precisión, página 28).

==<>Cuando desee comparar dos vectores (o dos matrices), las pruebas " " y " " comparan término a término. Por ejemplo:

```
-->X=[1,2,5]; Y=[5,3,5];

-->X==Y
 ans = [1x3 booleano]
  F F T
```

Para comprobar si dos vectores son iguales, utilizamos **isequal**, y si son diferentes, **~isequal** :

```
-->igual(X,Y)
 años =
  F

-->~igual(X,Y)
 años =
  T
```

Si...entonces

Las estructuras clásicas son las siguientes:

- **if ... then ... else ... end** ("Si...entonces...si no...fin de if"),
- **if ... then ... elseif ... then ... else ... end** ("Si...entonces...o si...entonces ... o ... fin de if").

`if ... then` debe escribirse en la misma línea, al igual que **`elseif ... then`**.

Ejemplo: Alicia lanza tres dados.

- Si obtiene tres 6, gana 20 euros,
- Si obtiene tres resultados idénticos distintos de 6, gana 10 euros,
- Si obtiene dos resultados idénticos, gana 5 euros,
- De lo contrario, no gana nada.

Podemos simular un lanzamiento y calcular la ganancia de Alice, utilizando las funciones :

- **`grande`** (véase la página 21),
- **`unique(D)`**, que mantiene los valores que aparecen varias veces en **`D`** sólo una vez,
- **`length(unique(D))`** que da el tamaño del vector obtenido, es decir, 1 si los tres términos son iguales, 2 si dos términos son iguales.

Algoritmo	Editor Scilab
Ponga los tres valores en D	`D=grand(1,3,"uin",1,6);`
Si Alicia obtiene tres 6, entonces	`si D==[6,6,6] entonces`
Alice gana 20 euros	`  G=20;`
De lo contrario, si tira 3 dados idénticos, entonces	`elseif length(unique(D))==1 then`
Alice gana 10 euros	`  G=10;`
De lo contrario, si tira 2 dados idénticos, entonces	`elseif length (unique(D))==2 then`
Alice gana 5 euros	`  G=5;`
En caso contrario	`si no`
Alice no gana nada	`  G=0;`
Fin de if	`fin`
Mostrar las ganancias de Alice	`disp("Alice gana "+..`
	`cadena(G)+ " euros")`

Dibujos en 2 y 3 dimensiones

Los trazados en el plano se crean utilizando el comando **`trazar`**. Puede elegir el color y el aspecto encerrando el color y el estilo de los puntos entre comillas:

- Los colores

`"b"` = azul (por defecto), **`"k"`** = negro, **`"r"`** = rojo, **`"g"`** = verde, **`"c"`** = cian, **`"m"`** =

magenta,

"y" = amarillo, **"w"** = blanco.

- Estilos de puntada

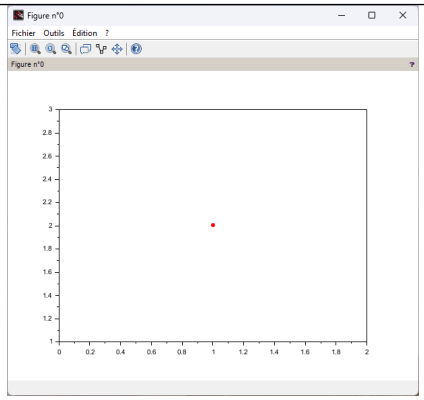
Enlazado (por defecto), o **"."**, **"+"**, **"o"**, **"x"**, **"*"**.

Otras opciones están disponibles con : **"s"**, **"d"**, **"v"**, **"<"** y **">"**.

Diseños básicos

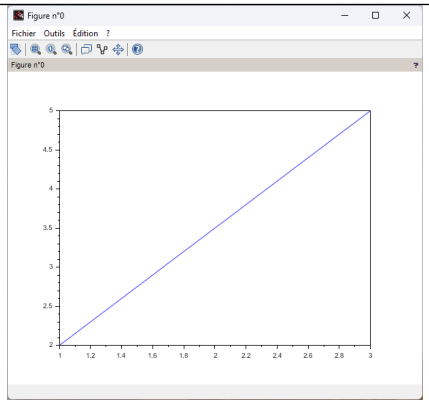
Para dibujar un punto

Dibuja el punto A(1; 2) con un punto rojo.

Editor Scilab	Ventana gráfica
`plot(1,2,".r")`	

Para dibujar un segmento

Dibuja el segmento [AB] en azul (por defecto) con A(1; 2) y B(3; 5).

Editor Scilab	Ventana gráfica
`plot([1,3],[2,5])`	

Trazar curvas planas definidas por las funciones y=f(x)

$x \to f(x)xx$Para una función, primero damos los valores de x utilizando el comando **`linspace`** escribiendo: **`x=linspace(a,b,n);`** donde **`a`** es el valor más pequeño de la variable , **`b`** es el valor más grande de la variable , y **`n`** es el número de valores que se calcularán entre **`a`** y **`b`**.

xNo olvide el "**`;`**", de lo contrario se mostrarán los n valores de.

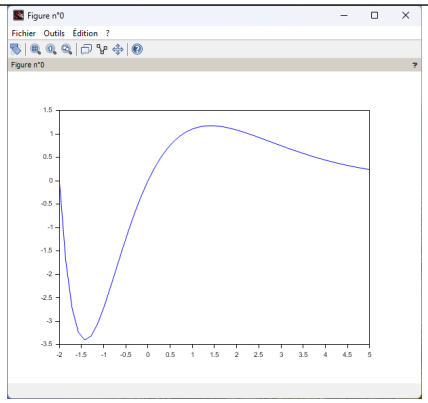
$f\,g$Por ejemplo, sean y dos funciones definidas en [-2; 5] por :

$$f(x)=(x^2+2x)e^{-x} \text{ y } g(x)=\sin\left(\frac{x}{2}\right)$$

fEl programa siguiente muestra la curva para , mostrada en azul por defecto.

Editor Scilab	Ventana gráfica

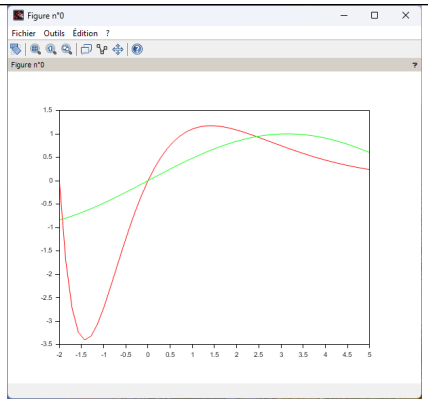
```
función y=f(x)
  y=(x^2+2*x)*exp(-x)
endfunction
x=linspace(-2,5,50);
dibujar(x,f)
```



Añadiendo el programa siguiente, obtenemos el trazado de las dos curvas, la de **`f`** en rojo y la de **`g`** en verde. El gráfico anterior se ha borrado con el comando **`clf`**.

Editor Scilab	Ventana gráfica

```
función y=g(x)
  y=sin(x/2)
endfunction
x=linspace(-2,5,50);
clf
plot(x,f, "r",x,g, "g")
```



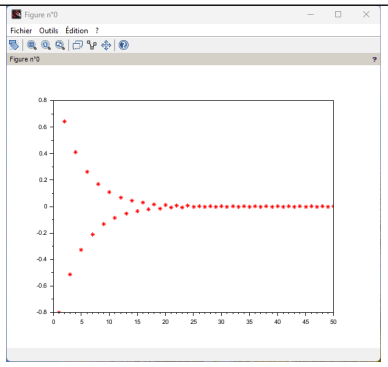
Trazado de nubes de puntos

Términos de una secuencia

$M(n,u(n))u(n)u$El caso más común es cuando se quiere trazar los puntos después de calcular las coordenadas de un vector . Escribimos **`plot(u, "*r")`** especificando la forma y el color de los puntos de la nube entre comillas. Aquí hemos elegido estrellas rojas que no están conectadas. Por defecto, los puntos son azules y están conectados.

Editor Scilab	Ventana gráfica

```
para n=1:50
  u(n)=(-0,8)^n;
fin
clf; plot(u, "*r")
```

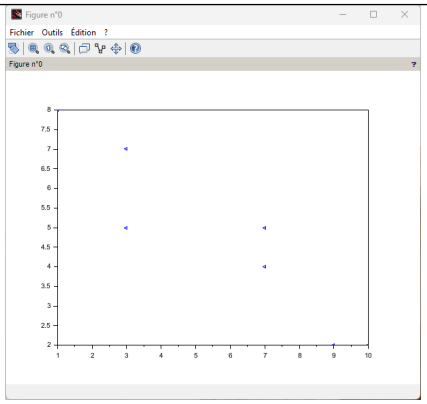


Doble estadística

$M(X_i;Y_i)$Las nubes estadísticas se dan en forma de dos vectores: llamémoslos X e Y, y escribamos **`plot(X,Y,"<")`** para trazar la nube de puntos con triángulos azules.

Editor Scilab	Ventana gráfica

```
X=[1,3,3,7,7,9,10];
Y=[8,7,5,5,4,2,2];
clf; plot(X,Y,"<")
```



Gráficos tridimensionales

Scilab permite dibujar superficies y curvas en el espacio, con un gran número de opciones para manejar caras ocultas, colores de caras, puntos de vista, etc. Daremos aquí sólo dos ejemplos.

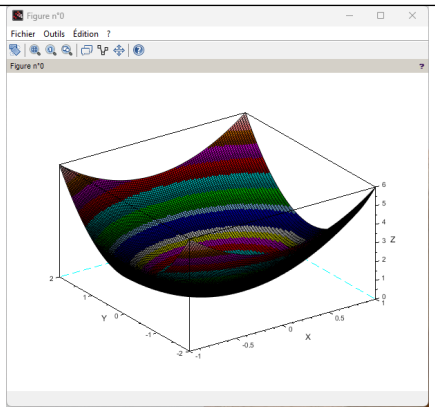
La función **`surf`** se utiliza para trazar una superficie. $mn(Ox)(Oy)n\times mz_{ij}x_iy_j$Esta función toma tres variables de entrada, **`x`**, **`y`** y **`z`**. **`x`** e **`y`** son vectores de dimensiones respectivas y correspondientes a puntos de los ejes y . **`z`** es una matriz de dimensión cuyo elemento es la dimensión del punto de la superficie con abscisa y ordenada .

$z=f(x,y)$Para trazar la superficie definida por una función del tipo , necesitamos :

- Definir la función f
- Calcular **`z=feval(x,y,f)'`**.
  $m\times nijf(x_i,y_j)$**`feval(x,y,f)`** devuelve la matriz cuyo elemento es el que transpondremos utilizando el apóstrofo " **`'`** ".
- Aplica **`surf(x,y,z)`**.

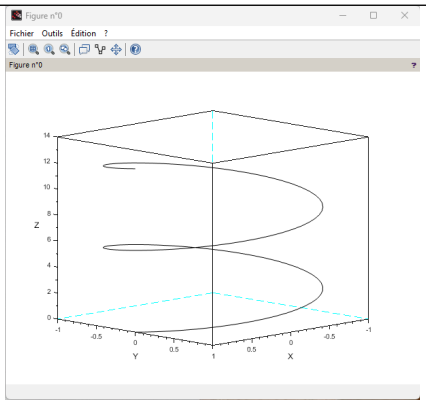
$z=2x^2+y^2$Trazado de la superficie (paraboloide elíptico):

Editor Scilab	Ventana gráfica

función z=f(x,y)
  z=2*x^2+y^2;
endfunction
x=linspace(-1,1,100);
y=linspace(-2,2,200);
z=feval(x,y,f)';
clf
surf(x,y,z)
	

(x_i, y_i, z_i)La función **param3d** se utiliza para dibujar una curva en el espacio. **param3d** toma tres argumentos, **x**, **y** y **z**, que son vectores del mismo tamaño correspondientes a los puntos de la curva.

$(x = \cos\cos(t), y = \sin\sin(t), z = t)$La trayectoria de la hélice definida por :

Editor Scilab	Ventana gráfica
`t=linspace(0,4*%pi,100);` `clf ; param3d(cos(t),sin(t),t)`	

Simulaciones y estadísticas

Dispone de una amplia gama de funciones para ejecutar simulaciones de forma rápida y eficaz.

Sorteos aleatorios con orden y descuento

- $pmnpmnm \leq n$**grand(1,p, "uin",m,n)** devuelve un vector de extracciones de enteros aleatorios tomados entre y con un entero positivo, y enteros y .

```
-->t= grand(1,4, "uin",1,6)
 t = [1x4 doble]
    3. 1. 3. 6.
```

- $pabpaba \leq b$**grand(1,p, "unf",a,b)** devuelve un vector de extracciones reales aleatorias tomadas entre y con un número entero positivo, y reales y .

```
-->tr= grand(1,2, "unf",-1,1)
 tr = [1x2 doble]
  - 0.7460264 0.9377355
```

Estadísticas

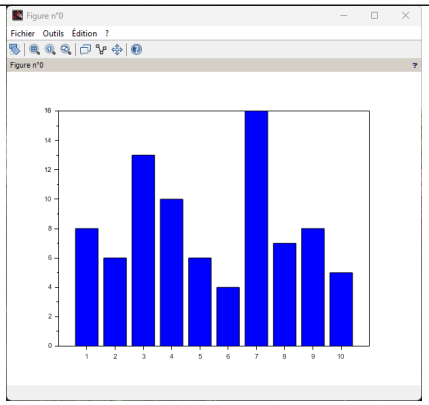
Todas las funciones estadísticas habituales se enumeran en la página 31.

Tenga esto especialmente en cuenta:

- La función **bar(x,n,color)**, que dibuja gráficos de barras:

Editor Scilab	Ventana gráfica

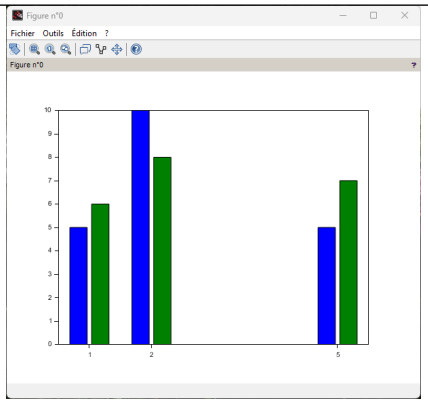
```
x=[1:10];
n=[8,6,13,10,6,4,16,7,8,5];
clf; bar(x,n)
```



- Para un diagrama de barras que representa dos series una al lado de la otra: la serie de valores X, y las dos series de números n1 y n2. Para el gráfico, n1 y n2 deben ser vectores columna, por lo que en el ejemplo siguiente, tomamos las transposiciones :

Editor Scilab	Ventana gráfica

```
X=[1,2,5];n1=[5,10,5];n2=[6,8,7];
clf; bar(X,[n1',n2'])
```



El argumento opcional **`color`** define el color como en la función **`trazar`**.

Más información sobre matrices y vectores

Acceso a los elementos

Para definir una matriz se utilizan corchetes. Se utiliza un espacio o una coma para pasar de una columna a la siguiente y un punto y coma para pasar de una fila a la siguiente.

```
-->m=[1 2 3;4 5 6]
 m = [2x3 doble]
    1. 2. 3.
    4. 5. 6.
```

Tenga en cuenta que
También es posible escribir este comando de la forma:

Los corchetes se utilizan para acceder a los elementos o modificarlos.

```
-->m(2,3)
 años =
```

```
    6.

-->m(2,3)=23
 m = [2x3 doble]
    1. 2. 3.
    4. 5. 23.
```

El operador ":" se utiliza para designar todas las filas o columnas de una matriz.

Para obtener la segunda fila de la matriz **m**, escriba :

```
-->m(2,:)
 ans = [1x3 doble]
    4. 5. 23.
```

y la tercera columna :

```
-->m(:,3)
 ans = [2x1 doble]
    3.
    23.
```

Para obtener la transposición de una matriz o un vector, utilice el apóstrofo " **'** " :

```
-->m'
 ans = [3x2 doble]
    1. 4.
    2. 5.
    3. 23.
```

Operaciones

Las operaciones "*****" y "**/**" son operaciones matriciales. Para realizar operaciones elemento a elemento, preceda el signo de la operación con un punto: "**.***", "**./**".

```
-->A=[1,2,3;4,5,6]
 A = [2x3 doble]
    1. 2. 3.
    4. 5. 6.
```

```
-->B=[1;1;2]
 B = [3x1 doble]
    1.
    1.
    2.
```

```
-->A*B                                  Multiplicación de matrices
```

ans = [2x1 doble] 9. 21.	
-->A*A Error 10 Operador *: Dimensiones erróneas para la operación [2x3] * [2x3].	Las dimensiones no son las correctas
-->A.*A ans = [2x3 doble] 1. 4. 9. 16. 25. 36.	Multiplicación elemento a elemento
-->2*(A+2) ans = [2x3 doble] 6. 8. 10. 12. 14. 16.	La operación se realiza en cada elemento porque 2 es un número
-->A/A ans = [2x2 doble] 1. 5.061D-17 -2.702-15 1.	Dar la matriz X tal que X*A = A La respuesta correcta es : 1. 0 0 1. Por razones de precisión del cálculo, el resultado puede ser ligeramente diferente dependiendo de su versión de Scilab y de su sistema operativo (ver detalles de cálculo, página 28).
-->A./A ans = [2x3 doble] 1. 1. 1. 1. 1. 1.	Da la matriz dividida elemento a elemento

En el caso de los vectores :

-->C=1:4 C = [1x4 doble] 1. 2. 3. 4.	
-->C*C Operador *: Dimensiones incorrectas para la operación [1x4] * [1x4].	Las dimensiones no son las correctas
-->C.*C ans = [1x4 doble] 1. 4. 9. 16.	
-->(1)./C ans = [1x4 doble] 1. 0.5 0.3333333 0.25	Invertir elemento por elemento Los paréntesis alrededor de 1 son necesarios para que el punto no se considere una coma que forma parte del número 1. También puede escribir **1 ./C**

con un espacio entre **1** y " **.**

Resoluciones del sistema

Para resolver el sistema lineal AX = Y, donde A es una matriz cuadrada, utilice la barra invertida "\".

X = A \ Y.

Nótese que la operación Y / A dará (siempre que las dimensiones sean correctas) otro resultado, es decir, la matriz Z tal que Z A = Y.

Para resolver el sistema : $\begin{pmatrix}1&2&3\\4&5&6\end{pmatrix}\times X=\begin{pmatrix}1\\1\end{pmatrix}$

```
-->A=[1 2 3;4 5 6];

-->Y=[1;1];

-->X=A\Y
 X = [3x1 doble]
  - 0.5000000
    0.
    0.5000000

-->A*X
 ans = [2x1 doble]
    1.000000
    1.
```

Algunas funciones útiles

Ordenar

La función **gsort** se utiliza para ordenar los elementos de un vector en orden ascendente o descendente.

```
-->v=[2,6,9,6,-4,0,2]
 v = [1x7 doble]
    2. 6. 9. 6.  - 4. 0. 2.

--> gsort(v, "g", "i")
 ans = [1x7 doble]
  - 4. 0. 2. 2. 6. 6. 9.

--> gsort(v)
 ans = [1x7 doble]
    9. 6. 6. 2. 2. 0. - 4.
```

Talla

La función **longitud** devuelve el número de coordenadas de un vector. La función **tamaño** devuelve las dimensiones (filas, columnas) de una matriz.

```
-->U=[1:10]
 U = [1x10 doble]
    1. 2. 3. 4. 5. 6. 7. 8. 9. 10.

-->longitud(U)
 años =
    10.

-->m=[1 2 3;4 5 6];

-->tamaño(m)
 ans = [1x2 doble]
    2. 3.
```

Suma y producto

Las funciones **sum** y **prod** calculan la suma y el producto respectivamente de los elementos de su argumento.

```
-->U=[1:10];

-->suma(U)
 años =
```

```
    55.

-->prod(U)
 años =
    3628800.
```

Único

La función **unique** mantiene los elementos de un vector una sola vez (aunque se repitan varias veces) y los ordena en orden ascendente. Puede resultar muy útil para realizar pruebas.

```
-->v=[2,6,9,6,-4,0,2]
 v = [1x7 doble]
    2. 6. 9. 6.  - 4. 0. 2.

-->único(v)
 ans = [1x5 doble]
  - 4. 0. 2. 6. 9.
```

Encuentre

La función **find** busca elementos en un vector o matriz y devuelve un vector que contiene los índices correspondientes.

wEncontrar todos los elementos del vector estrictamente menores que 5 :

```
-->w=[1,5,3,8,14,7,3,2,12,6]; find(w<5)
 ans = [1x4 doble]
    1. 3. 7. 8.
```

$w_1, w_3, w_7 w_8$El vector resultado (1,3,7,8) nos dice que los elementos y son menores que 5.

wEncontrar todos los elementos del vector igual a 3 :

```
-->w=[1,5,3,8,14,7,3,2,12,6]; find(w==3)
 ans = [1x2 doble]
    3. 7.
```

$w_3 w_7$El vector de resultados (3.7) indica que los elementos y son iguales a 3.

Problemas de precisión

Para el cálculo

Los números tienen valores absolutos entre aproximadamente $2{,}2\times10^{-308}$ y $1{,}8\times10^{+308}$.

El número **%eps** igual a **2,220446049D-16** da la menor precisión relativa que cabe esperar en el cálculo, es decir, unos 16 dígitos.

Ejemplo 1: Cálculo de sen(π)

```
-->sin(%pi)
 años =
1.225D-16
```

$\sin(\pi)$El valor de arriba no es 0, pero se considera cero. De hecho, comparado con el valor máximo de la función seno (es decir, 1), es igual a 0 con un error inferior a **%eps**.

√❑Ejemplo 2: Comprobemos si el triángulo de lados , 1 y 2 es rectángulo:

```
-->a=sqrt(3)
 a =
    1.7320508
```

```
-->b=1
 b =
    1.
```

```
-->c=2
 c =
    2.
```

```
-->a^2+b^2==c^2
 años =
  F
```

El programa responde falso porque el valor de a^2+b^2 es aproximado

-->abs(a^2+b^2-c^2)<%eps años = F	El programa responde falso porque la precisión solicitada es absoluta
-->abs(a^2+b^2-c^2)/c^2<%eps años = T	El programa responde verdadero porque la precisión solicitada es relativa

Para visualizar

Por defecto, los resultados se muestran con 10 caracteres, incluidos el punto decimal y el signo. La función **format** puede utilizarse para mostrar más dígitos. Para obtener 20 dígitos, escriba **format(20)**.

$a=\sqrt{\square}$Volvamos a :

-->a^2 años = 3.	Aquí hay 7 decimales, así que no se ve la precisión.
-->formato(20) -->a^2 años = 2.99999999999999956	Aquí hay 17 decimales, para que veas la precisión.

Resolución de ecuaciones diferenciales

Aquí mostramos cómo se pueden encontrar las soluciones de un sistema explícito de ecuaciones diferenciales. El principio es reducir a ecuaciones diferenciales de orden 1 :

$\{y'(t)=f(t,y(t))\, y(t_0)=y_0 t_0 t y_0 y(t)$d onde y representan el tiempo y y son vectores,

a continuación, aplique la función **ode**: y=ode(y0,t0,t,f), donde :

- n**y0**: condición inicial, vector de dimensión ,
- **t0**: tiempo inicial,
- **t**: vector de dimensión T de los tiempos en los que queremos tener la solución. Este vector debe empezar por **t0**,
- **f**: función que define el sistema de la forma :

```
función yprim=f(t,y)
    yprim(1)=...
    yprim(2)=...
    ....
    yprim(n)=...
endfunction
```

$n \times T$ La solución y es una matriz de dimensión : $\left(y_1(1)\, y_1(2) \vdots y_1(T)\, y_2(1)\, y_2(2) \vdots y_2(T) \vdots \vdots \vdots y_n(1)\, y_n(2) \ldots y_n(T)\right)$

Ejemplo: Resolver la ecuación diferencial

$\{ y$ =-4y y left (0 right) =3, {y} ^ {'} left (0 right) =0

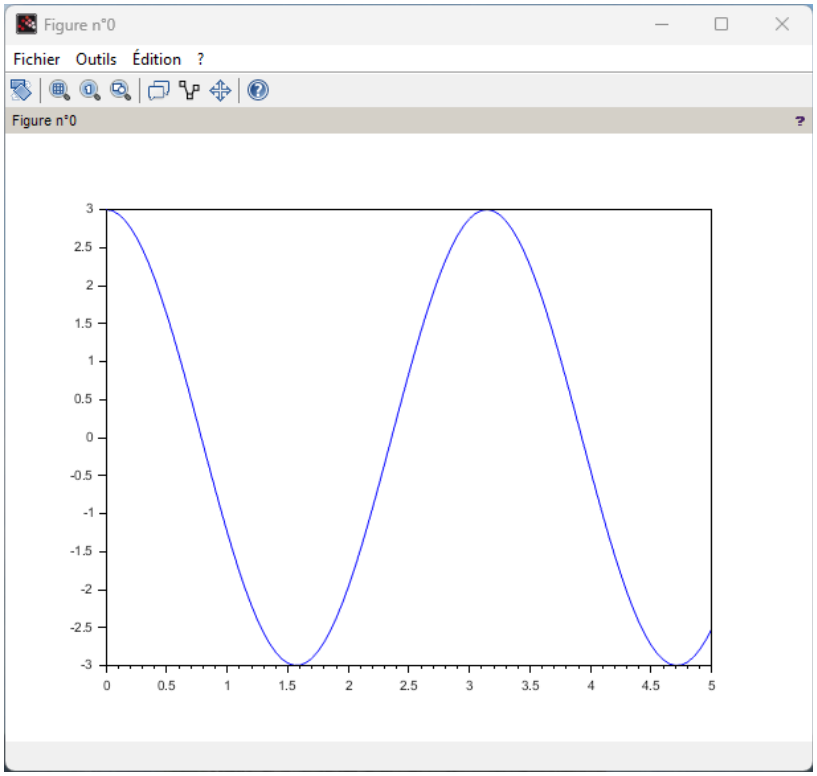
Esta ecuación de orden 2 puede reducirse a un sistema de dos ecuaciones de orden 1 planteando :

$Y = \left(Y(1)\, Y(2)\right) = \left(y\ y'\right), Yprim = \left(Yprim(1)\, Yprim(2)\right) =$ ¿ y $Yprim(1) = Y(2)\, Yprim(2) = -4 \times Y(1)$

Comentario	Editor Scilab

Definimos la función que mapea el vector y' a las dos variables t e y (que es un vector).	`función yprim=f(t,y)`
	`  yprim(1)=y(2);`
	`  yprim(2)=-4*y(1) ;`
	`endfunction`

Especificamos los valores de t para el gráfico (el propio programa elige los valores de t para su cálculo interno de la solución).	`t0=0; tmax=5;` `t=t0:0,05:tmax;`
Especificamos las condiciones iniciales	`y0=3; yprim0=0;`
Aplicamos ode	`y=ode([y0;yprim0],t0,t,f);`
Traza la curva integral de y en función de t	`clf; plot(t,y(1,:))`



Capítulo 3 - Funciones útiles de Scilab

Para el análisis

- xx**`sqrt(x)`** devuelve la raíz cuadrada de para real positivo o cero, y la raíz compleja de la parte real positiva en caso contrario.
- **`log(x)`** devuelve el logaritmo natural de x, donde x *es* un número real o complejo.
- xx**`exp(x)`** devuelve la exponencial de con un número real o complejo.
- xx**`abs(x)`** devuelve el valor absoluto del real (o el módulo si es complejo).
- x**`int(x)`** devuelve el truncamiento del real (entero antes del punto decimal).
- $xnn \leq x < n+1$**`floor(x)`** devuelve la parte entera del número real (entero como).
- $xnn-1 < x \leq n$**`ceil(x)`** para devuelve el entero tal que .

Para probabilidad y estadística

- **`factorial(n)`** devuelve el factorial de n con n un número entero positivo o cero.
- $m \leq n$**`grand(1,p, "uin",m,n)`** devuelve un vector de p extracciones

enteras entre m y n donde p es un entero positivo, m y n son enteros y .

- $a \leq b$**grand(1,p, "unf",a,b)** devuelve un vector de p extracciones reales entre a y b donde p es un entero positivo, a y b son reales y .
- n**sum(n)** devuelve la suma de los valores del vector (utilizada para calcular el número total de personas).
- n**cumsum(n)** devuelve el vector de valores acumulativos crecientes del vector (utilizado para calcular números acumulativos crecientes).
- v**length(v)** devuelve el número de coordenadas del vector .
- v**gsort(v)** devuelve el vector ordenado de números o cadenas de caracteres en orden descendente.
- v**gsort(v, "g", "i")** devuelve el vector ordenado de números o cadenas de caracteres en orden ascendente.
- v**mean(v)** devuelve la media del vector de números .
- v**stdev(v)** devuelve la desviación estándar del vector de números .
- $vnvn$**bar(v,n,color) traza** un diagrama de barras con el eje x, el eje y y vectores lineales de la misma dimensión. nPor defecto, **bar(n)** traza el gráfico de barras en azul con 1,2,3... como abscisas.
- vv**bar(v,[n1',n2']) dibuja** un gráfico de barras dobles con como eje x, n1 como eje y en azul y n2 como eje y en verde, con , n1 y n2 vectores lineales de la misma dimensión.
- $npn \times p$**rand(n,p)** con y enteros positivos, devuelve una matriz de números tomados al azar entre 0 y 1.
- **rand()** devuelve un número real tomado al azar entre 0 y 1.
- x**floor(x)** devuelve la parte entera del número real . $pp1-p$En particular, si es un número real entre 0 y 1, **floor(rand()+p)** será 1 con probabilidad y 0 con probabilidad .

Para visualizar y trazar

- **clf** significa "borrar figura" y borra la figura de la ventana de gráficos.
- puede utilizarse para dibujar curvas o nubes de puntos en dimensión 2.
- $abnnab$**linspace(a,b,n)**, con y reales y enteros, define un vector de valores espaciados regularmente entre y .
- **scf** puede utilizarse para abrir o seleccionar otras ventanas gráficas.
- **surf** puede utilizarse para dibujar superficies de dimensión 3.
- **bar(X,Y)** donde X e Y son vectores, dibuja un gráfico de barras de la serie de valores X con los valores Y como sus cuentas.
- **plot(X,Y, "*") traza** la nube de puntos con coordenadas (X(i),Y(i)) en forma de estrellas. Puede especificar el color.
- **plot(Y, "+") traza** la nube de puntos con coordenadas (i,Y(i)) como una cruz.
- **disp("Frase")** muestra lo que está escrito entre las comillas.
- **disp(A)** donde A es una matriz de números muestra la tabla de valores de A.
- x**disp("Frase "+string(x))** muestra la frase y el valor del número .

Servicios

- v**unique(v)** devuelve el vector con una única aparición de sus elementos duplicados y los ordena en orden ascendente.
- v**sum(v)** devuelve la suma de todos los elementos del vector o matriz .
- v**prod(v)** devuelve el producto de todos los elementos del vector o matriz .
- v**find(<prueba en v>)** devuelve los índices de los elementos del vector que verifican la prueba.
- **disp(x,y,...)** muestra los valores de sus argumentos en la consola.
- x**string(x)** transforma el número en una cadena de caracteres.
- nn**format(n)** donde es un entero mayor o igual que 2 establece la visualización en caracteres, incluyendo el signo y la coma.
- $n \times p$**zeros(n,p)** define una matriz que contiene sólo 0s.
- $xymnm \times n(i,j)f(x(i),y(j))$**feval(x,y,f)** donde y son vectores de tamaños y respectivamente, define la matriz cuyo elemento es .
- **doc(<función>)** abre el navegador de ayuda en la página de ayuda de la función.
- **tic** pone en marcha un cronómetro.
- **toc** detiene el cronómetro.