Curso de Octave

Pablo Barrera González <pbarrera@tsc.uc3m.es>

Grupo de Usuarios de Linux

Universidad Carlos III de Madrid

Jueves, 3 de abril de 2003

Índice

¿Qué es Octave?

- Lenguaje de alto nivel para cálculos numéricos.
- Interfaz de línea de comandos para resolver tanto problemas lineales como no lineales de forma numérica.
- Extensible, ajustable con el lenguaje propio o con módulos en C, C++ y Fortran.
- Es software libre, con licencia GPL (General Public License).

¿Qué no es Octave?

- No es la solución a todos tus problemas (pero casi).
- No van a saber matemáticas por ti.
  - Aunque te va a ayudar a hacer cálculos.
- No es Matlab.
  - Aunque la utilización es prácticamente igual.
- No es un lenguaje compilado.
  - Su propósito es muy concreto.
- No es lo más rápido posible.
  - Pero si se programa muy rápido.

Breve historia de Octave

- Creado en 1988 por John W. Eaton, y otros, en el campo de los reactores químicos.
- Necesitaban una herramienta para realizar numerosos cálculos.
  - A la vez debía ser muy sencilla de manejar y lo más general posible.
- En la primavera 1992 comienza a desarrollarse en serio.
- El 4 de enero de 1993 se distribuye una versión alfa.
- El 17 de noviembre de 1994 aparece la versión 1.0.
- El nombre proviene de un profesor que hacía cálculos rápidos de cabeza.

Versiones

- En la actualidad hay dos versiones de Octave
  - La versión estable: Ver. 2.0.17, del 12 de Abril de 2002
  - La versión inestable: Ver. 2.1.44 del 3 de Febrero de 2003
- En mi opinión, la inestable solo tiene de inestable el nombre.
- Aunque la que se suele instalar por defecto es la estable.
- Existen más funciones para la versión inestable, incluidas herramientas muy interesantes (señal, proceso de imágenes, sonido, ...)
- Además es más rápida (utiliza la biblioteca ATLAS).

Otras alternativas

- Matlab
  - No es ni libre ni gratis.
- Scilab:
  - El lenguaje no es compatible con Octave.
- Numeric Python
- PDL (Perl Data Language)
- C, C++, Fortran, Bash...
  - Hay otras opciones.

Invocación del programa

Solo hay que llamar al interprete desde línea de comando:

```
pablo@tormenta:~$ octave
GNU Octave, version 2.0.16.92 (i386-pc-linux-gnu).
Copyright (C) 1996, 1997, 1998, 1999, 2000 John W. Eaton.
This is free software with ABSOLUTELY NO WARRANTY.
For details, type 'warranty'.

octave:1> exit
pablo@tormenta:~$
```

Jugando con las variables

- En Octave hay un tipo de objeto principal: la matriz.
- En las versiones más modernas también existen estructuras de datos.
- Todas las operaciones están definidas sobre matrices.
- Una variable se puede definir en cualquier momento simplemente asignándole un valor.
- El valor puede pasarse tanto de forma directa como usando la salida de una función.

Ejemplo

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

  1  2  3
  4  5  6

octave:2> B = randn(2,3)
B =

   1.09471  -0.68816   0.43064
   0.27045   0.35705  -0.35804
```

Matemáticas básicas

Operaciones básicas entre matrices:

- Asignación: =
- Suma: +
- Resta: -
- Multiplicación: *
- Potencia: ^
- Transponer: ’
- División: /

Ojo, todas estas operaciones son matriciales.

Ejemplos

```
octave:3> A+A
ans =

   2   4   6
   8  10  12

octave:4> A'*A
ans =

  17  22  27
  22  29  36
  27  36  45
```

Creación de Matrices

- Se pueden asignar a mano:
  - A = [1,2,3; 4,5,6];
- Se pueden tomar como la salida de una función:
  - A = rand(10,10);
- Las funciones típicas son:
  - ones(filas, columnas): matriz con todo 1.
  - zeros(filas, columnas): matriz con todo 0.
  - rand(filas, columnas): matriz aleatoria entre 0 y 1.
  - randn(filas, columnas): matriz aleatoria gaussiana (0,1).
  - eye(filas,columnas): matriz identidad.

Truco: Colocando un ; al final de una línea no devuelve el resultado por pantalla. El ; también sirve para separar operaciones.

Otras funciones disponibles

- diag: matriz diagonal.
- triu: matriz triangular inferior.
- tril: matriz triangular superior.
- hilb: matriz de Hilbert.
- magic: matriz mágica.
- toeplitz: matriz toeplitz.

Definir rangos

Se usan los ":"

- valor_inicial:valor_final
- valor_inicial:incremento:valor_final

```
octave2.1:18> x = 1:5
x =

  1  2  3  4  5
octave2.1:19> x = 1:2:10
x =

   1   3   5   7   9
```

También es útil la función linspace.

- El anterior resultado también se conseguiría con linspace(1,10,5).

Acceder a las posiciones de una matriz

Se usa la coordenada entre paréntesis.

```
octave2.1:20> x(2)
ans = 3
```

Por supuesto se puede asignar un valor a una posición.

```
octave2.1:20> x(2) = 5
x =

  1  5  5  7  9
```

Nota: En Octave, al igual que en Matlab, la numeración comienza en 1.

Acceder a las posiciones de una matriz (II)

Si tiene dos dimensiones, se emplean dos coordenadas:

```
octave2.1:22> A = rand(3,3); A(2,1)
octave2.1:23> A(2,1)
ans = 0.052414
```

Acceder a porciones de matrices

- Hay veces que queremos extraer una parte de una matriz.
- Podemos poner un vector con los índices como coordenadas de una matriz.

```
octave2.1:24> x = 1:5;
octave2.1:25> x(1:3)
ans =

  1  2  3

octave2.1:26> x(1:2:5)
ans =

  1  3  5
```

Acceder a porciones de matrices (II)

- Los : sin número sirven para indicar que queremos todos los valores.

```
octave2.1:29> x = magic(3)
ans =

 8  1  6
 3  5  7
 4  9  2

octave2.1:30> x(1,:)
ans =

 8  1  6
```

Acceder a porciones de matrices (II)

- Los : sin número sirven para indicar que queremos todos los valores.

```
octave2.1:29> x = magic(3)
ans =

  8  1  6
  3  5  7
  4  9  2

octave2.1:30> x(1,:)
ans =

  8  1  6
```

Acceder a porciones de matrices (III)

Se pueden hacer muchas cosas interesantes:

- Dar la vuelta a un vector:

```
octave2.1:32> x=1:5;
octave2.1:33> x(length(x):-1:1)
ans =

  5  4  3  2  1
```

- Localizar un determinado valor en un vector:

```
octave2.1:34> x(find(x==3))
ans = 3
```

Añadiendo a matrices

- Para colocar un elemento delante de una matriz

```
x = [nuevo, x];
```

- Para colocarlo detrás de una matriz

```
x = [x , nuevo];
x(length(x)+1) = nuevo;
```

- Para añadir una fila

```
x = [x ; nuevo];
```

Relaciones lógicas

- Igualdad: ==
- Diferencia: ~= y !=
- Mayor que: >
- Menor que: <
- Mayor o igual que: >=
- Menor o igual que: <=

Nota: Matlab sólo soporta la relación lógica de diferencia ~=

Operadores lógicos

- Unión: & ó &&
- Opción: | ó ||
- Negación: ~ ó !

Nota: Matlab sólo soporta el operador lógico de negación ~

Operaciones por coordenadas

- Todas las operaciones que hemos visto funcionan sobre matrices.
- Para operar con coordenadas hay que anteponer un . a los signos de operación.

```
octave2.1:1> x = [1, 2];
octave2.1:2> y = [3, 4];
octave2.1:3> x*y
error: operator *: nonconformant arguments (op1 is 1x2, op2 is 1x2)
error: evaluating binary operator ‘*’ near line 3, column 2
octave2.1:3> x.*y
ans =

  3  8
```

Constantes especiales

Hay variables con unos valores predefinidos:

- pi: pi.
- e: e.
- i, j: números complejos. i = sqrt(-1).
- Inf: Infinito,
- NaN: No es un número (Not a number).
- eps: Número muy pequeño (pero no es el límite de la precisión).

Nota: algunas de estas variables pueden ser sobreescritas por el usuario.

Iteraciones para

Forma de bucle:

```
for indice = lista_de_valores_que_puede_tomar
   ...
end
```

Ejemplos de listas de valores

- 1:10
- 2:2:10
- x
- [1 5 14]
- 1:limite

Ejemplo con for

Construir una matriz de Hilbert.
En cada posición de la matriz (i,j) tenemos 1/(i+j-1).

```
for i = 1:m
    for j = 1:n
        H(i,j) = 1/(i+j-2);
    end
end
```

Iteraciones mientras

Forma del bucle

```
while condicion
   ...
end
```

Ejemplo con while

```
n = 0;
while 2^n < limite
    n = n + 1;
end
n
```

Condiciones

Forma general con múltiples ramificaciones:

```
if condicion
  ...
elseif condicion
  ...
else condicion
  ...
end
```

Manejando la interfaz

- who: Muestra las variables creadas.
- whos: Como who pero dice su tamaño
- clear: Borra una variable.
- load: Carga un archivo con datos.
- save: Guarda los datos de una sesión.
- help: Muestra la ayuda sobre un comando.

Funciones escalares

Sin comentarios...

- sin
- cos
- tan
- asin
- acos
- atan
- sinc
- exp
- log: Logaritmo natural (ojo).
- log10: Logaritmo en base 10.
- rem: Resto de la división.
- abs: Módulo.
- sqrt: Raíz cuadrada.
- sign: Signo.
- round: Redondear al entero más cercano.
- floor: Redondear hacia abajo.
- ceil: Redondear hacia arriba.

Funciones vectoriales

Aquí tampoco hay nada que comentar

- max: Máximo.
- min: Mínimo.
- sort: Ordenar.
- sum: Sumatorio.
- prod: Productorio.
- median: Mediana.
- mean: Media.
- std: Desviación típica.
- var: Varianza.
- any: Cualquier elemento es distinto de 0.
- all: Todos los elementos son distintos de 0.
- find: Devuelve los índices de los vectores distintos de 0.

Funciones matriciales

Otra tanda más de funciones

- eig: Autovalores y autovectores.
- chol: Factorización de Cholesky.
- svd: Descomposición en valores singulares.
- inv: Inversa.
- lu: Descomposición LU.
- qr: Descomposición QR.
- hess: Forma de Hessenberg.
- schur: Descomposición de Schur.
- expm: Matriz exponencial.
- sqrtm: Matriz raíz cuadrada.
- poly: Polinomio característico.
- det: Determinante.
- size: Tamaño.
- norm: Norma 1, norma 2, norma de Frobenius, norma infinito.
- cond: Número de condición en la norma 2.
- rank: Rango.

Mostrar los resultados por pantalla

Varias opciones:

- Quitar el ; detrás de una línea
- disp
- fprintf

Ejemplos

```
x=pi^2
disp("pi al cuadrado es"), disp(pi^2)
fprintf("pi al cuadrado es %f\n", pi^2)
```

Guiones con Octave

- En vez de teclear cada vez todas las expresiones podemos juntarlas en un archivo.
- Extensión .m
- Es, simplemente, una lista de comandos.
- Se llama por el nombre del fichero desde Octave.

Trucos

- Los comandos que tecleamos se guardan en $HOME/.octave_hist
- También podemos usar la función diary.

Creación de funciones

También se pueden definir funciones

```
function valores_salida = nombre_funcion(valores_entrada)
    ...

[endfunction]
```

- Se pueden guardar en un archivo llamado nombre_funcion.m

Ejemplo de una función

Generamos una matriz cuadrada con valores aleatorios entre 0 y 9.

```
function matriz = entale(dimension)
   matriz = floor(10*rand(dimension))
```

Formatos de salida

- Cambia el formato de la salida de los números por pantalla.
  - Pero no cambia la representación interna de los números.

- format short
- format long
- format short e
- format long e

Octave y los gráficos

- Hace llamadas a gnuplot.
- Tiene una interfaz parecida a Matlab.
  - Pero no igual.
- Tiene la siguientes funciones:
  - plot: Gráfico en 2 dimensiones.
  - semilogx, semilogy: Papel semilogarítmico en el eje X e Y.
  - loglog: Gráfico logarítmico.
  - plot3: Líneas en 3 dimensiones.
  - mesh: Superficies en 3 dimensiones.
  - contour: Contornos de superficies.
  - bar: Diagramas de barras.
  - stairs: Escaleras.
  - polar: Gráficos polares.
  - image, imagesc: Imágenes.
  - replot: Repintar.
  - xlabel, ylabel: Etiquetas de los ejes X e Y.
  - title: Título de la figura.
  - hold on|off: Pintar sobre el último gráfico o borrar antes de pintar.

Ejemplo de un gráfico sencillo

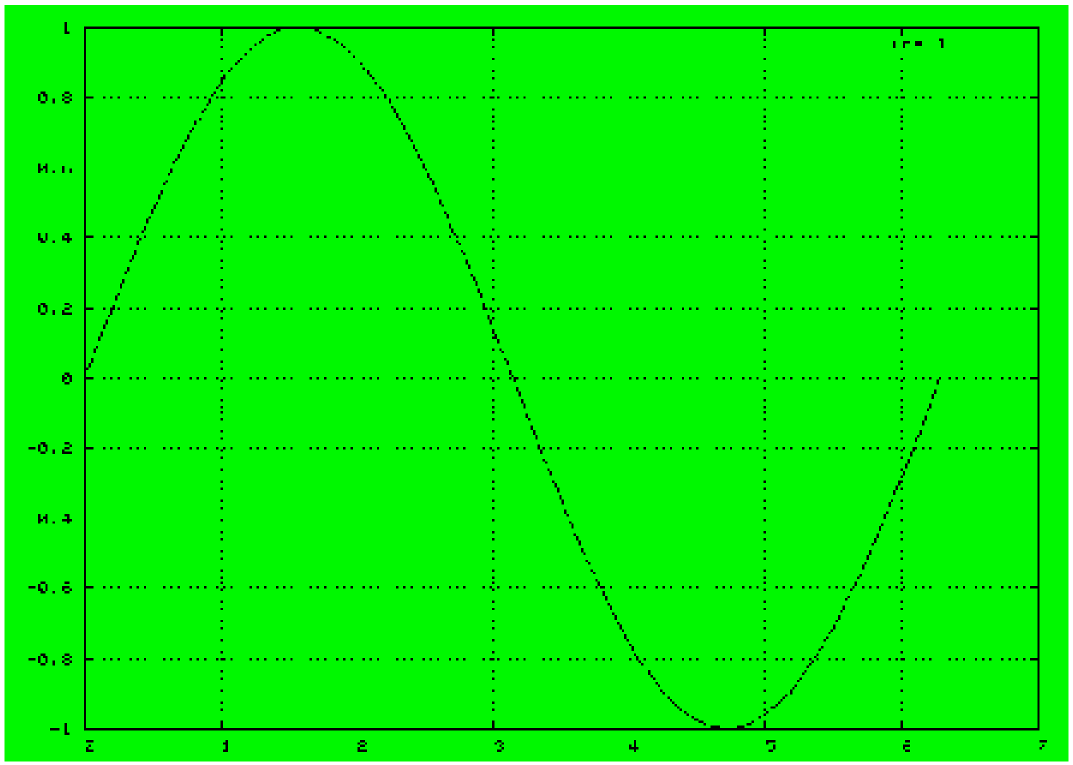
octave:15> t = linspace(0,2*pi,100);

octave:16> x = sin(t);

octave:17> plot(t,x)

octave:18> grid

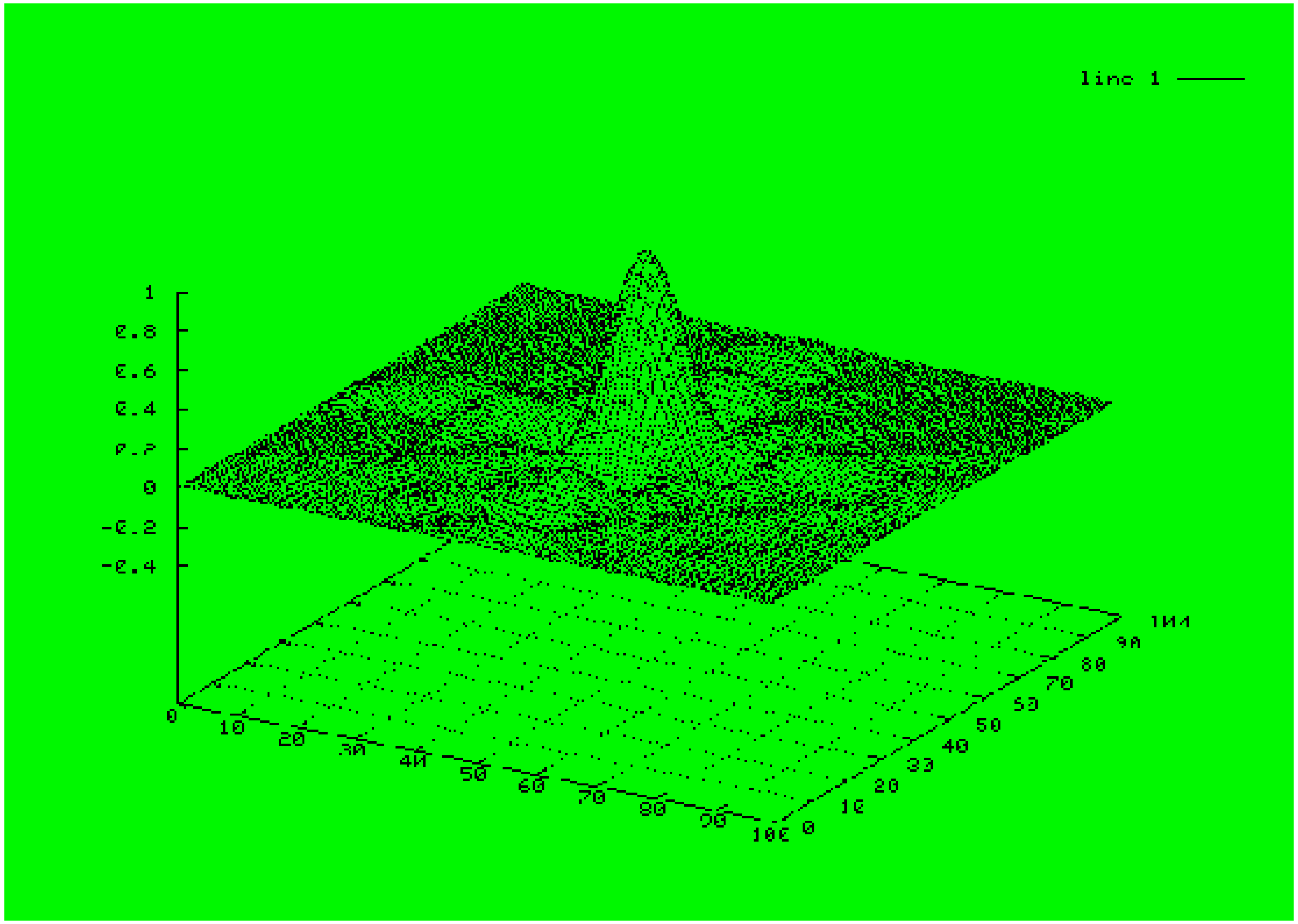
octave:19> replot



Ejemplo de un gráfico más elaborado

octave:55> t=linspace(-5,5,100);

octave:56> mesh(sinc(t)'*sinc(t))



Cómo guardar los gráficos al disco duro

Usando gnuplot. Desde Octave tecleo:

gset terminal TERMINAL

gset output "FICHERO"

replot

gset terminal X11 % para volver al modo normal

Los dos terminales más importantes:

- Postscript
- Png (de varios tamaños)

Como quitar la leyenda:

gset nokey

Algunas grupos de herramientas

- Análisis de datos: filtrado, análisis de Fourier, geometría...
- Audio:Cargar y guardar, reproducir...
- Calculo: Diferenciación, integración, EDO...
- Comunicaciones: Codificadores de bloque, codificación de fuente...
- Teoría de control: Análisis en el dominio del tiempo, análisis de frecuencias...
- Procesado de Imagen: Cargar y guardar, mapas de colores, control de color...
- Optimización: Ajuste de datos, minimización,...
- Proceso de señal: Diseño FIR, construcción de filtros...
- Álgebra simbólica: polinómios, calculo...
- Análisis de series temporales: análisis adaptativo, análisis multivariable.
- Vrml: Visualización en 3D y creación de objetos.

Diferencias con Matlab

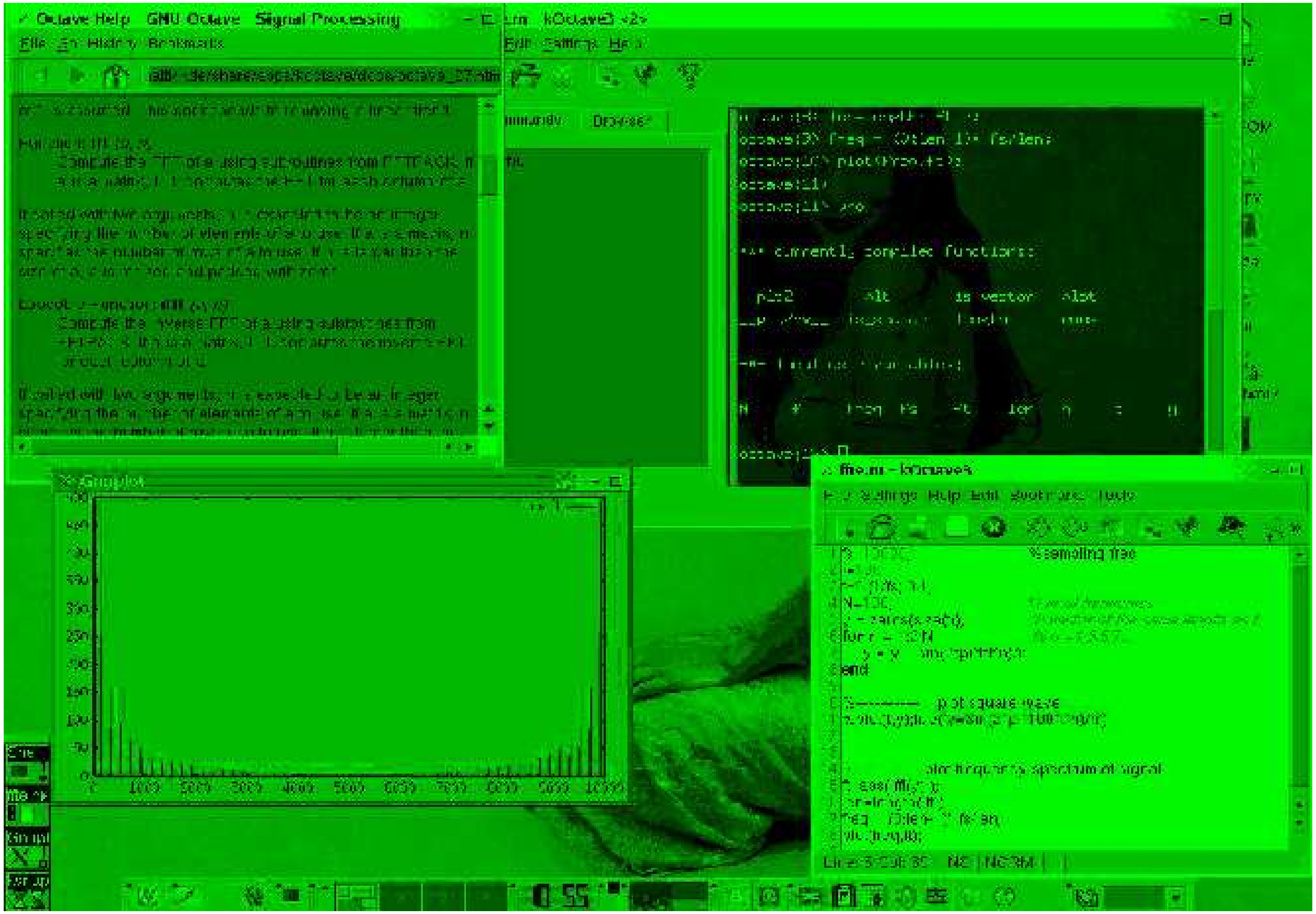
- No hay posición end.
- Octave solo trabaja con 2 dimensiones.
- Las funciones gráficas no son del todo compatibles.
- Los formatos de los archivos de datos no son compatibles.
  - Aunque se puede usar formato ascii.
- No están disponibles todas las herramientas de Matlab.
- En Octave las cadenas de caracteres se delimitan por " o por ’
- En Octave las funciones puede acabar por endfunction
- En Octave los comentarios pueden empezar por # o por %
- Los archivos MEX y OCT no son compatibles.
  - Tienen APIs diferentes.

Sabor a Matlab

```
PS1 = ">> "
PS2 = ""
beep_on_error = 1.0
default_eval_print_flag = 0.0
default_save_format = "mat-binary"
define_all_return_values = 1.0
do_fortran_indexing = 1.0
empty_list_elements_ok = 1.0
fixed_point_format = 1.0
implicit_num_to_str_ok = 1.0
implicit_str_to_num_ok = 1.0
ok_to_lose_imaginary_part = 1.0
page_screen_output = 0.0
prefer_column_vectors = 0.0
prefer_zero_one_indexing = 1.0
print_empty_dimensions = 0.0
treat_neg_dim_as_zero = 1.0
warn_function_name_clash = 0.0
whitespace_in_literal_matrix = "traditional"
```

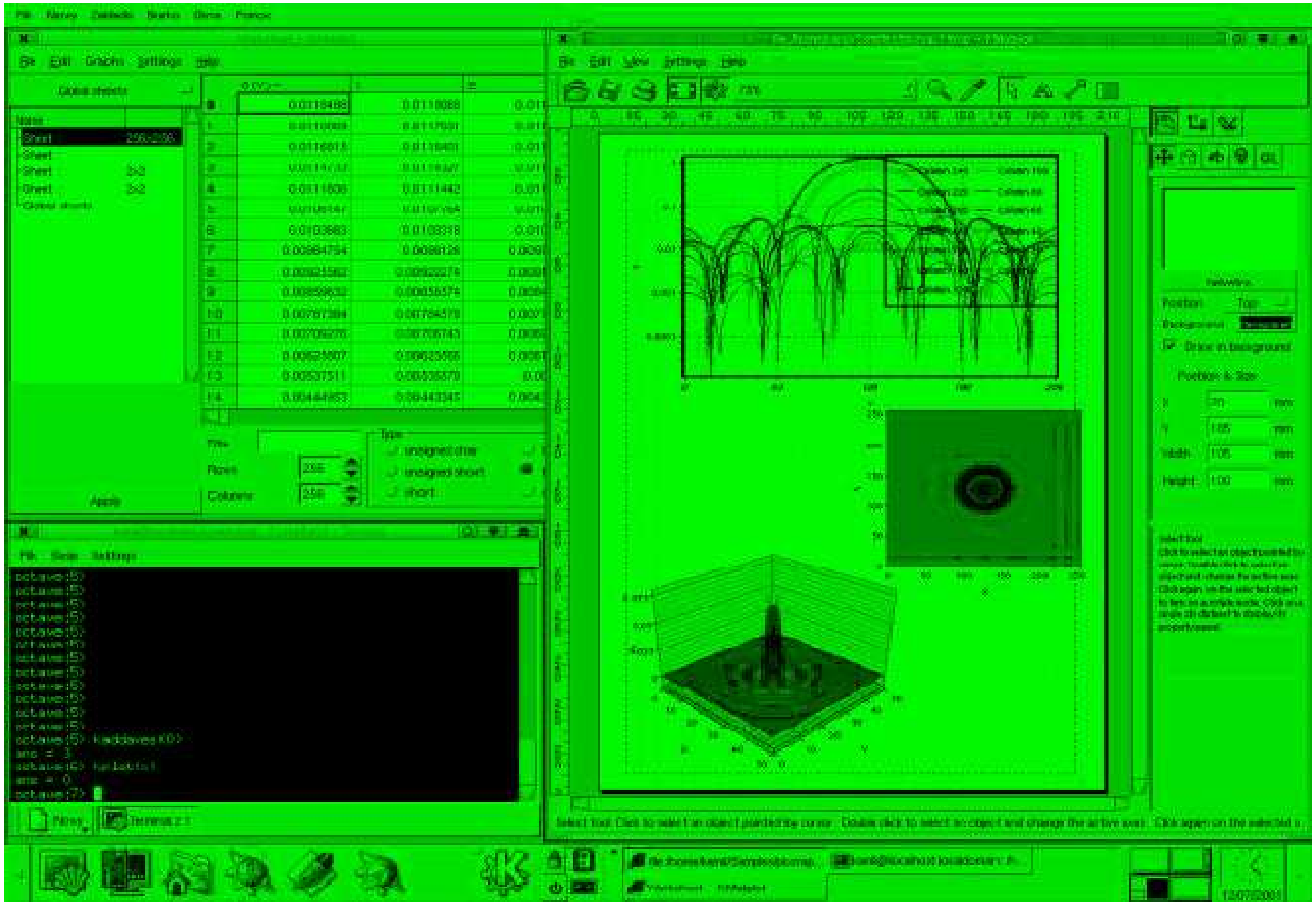
Otras aplicaciones para trabajar con Octave

Koctave



Otras aplicaciones para trabajar con Octave

Qmatplot, antes Kmatplot



Trucos para mejorar el rendimiento

- Intenta evitar los bucles for.
  - Mucha veces puedes hacer lo mismo con una multiplicación de matrices.
- Intenta reservar memoria al principio en vez de ir incrementado las matrices poco a poco.
- Borra las variables que no sirvan ya para nada.
- Guarda los índices de una matriz en vez de los valores.

Unas cuantas funciones más para nota

- eval: Ejecuta la cadena de caracteres de entrada como un comando de Octave.
- fopen, fprintf, fscanf: Permiten abrir y guardas datos en archivos.
- reshape: Cambia el tamaño de una matriz.
- exist: Existe una variable.
- fft, ifft: Transformada rápida de Fourier e su inversa.
- global: Define una variable global.
- subplot: Permite hacer regiones dentro de gnuplot.
- tic,toc: Mide tiempo desde tic hasta toc.
- pause: Hace una pausa hasta que el usuario toca una tecla.
- loadimage: Carga una imagen.
- loadaudio, record: Carga un sonido.
- menu: Permite hacer menús de texto con múltiples opciones.
- addpath: Añade un directorio a la ruta de búsqueda.
- str2mat: De cadena de caracteres a matriz.
- str2num: De cadena de caracteres a número.
- num2str: De número a cadena de caracteres.

Añadiendo funciones en C++

Es posible añadir funciones compiladas en C++.

- Enlazan dinámicamente
- La velocidad de ejecución es muy alta.
- Difíciles de hacer, la documentación es muy mala.
  - Pero no imposible.
- Todas las funciones y objetos de Octave están disponibles con la liboctave.
  - Matrices
  - Vectores
  - Matrices dispersas
  - Cadenas de caracteres
  - Estructuras
- Todos son objetos de C++.
- Es posible usar la liboctave como soporte matemático para nuestros programas en C++.
  - Aunque no está pensada para esto.

hello.cc

```
include <octave/oct.h>
include <iostream.h>
DEFUN_DLD (hello, args, ,
  "[...] = hello (...)\n\nPrint greeting followed by the values of all the arguments
passed.\nReturns all arguments in reverse order.")
{
  octave_value_list retval;
  octave_stdout << "Hello, world!\n";
  int nargin = args.length ();
  for (int i = 0; i < nargin; i++)
    {
      octave_value tmp = args (i);
      tmp.print (octave_stdout);
      retval (nargin-i-1) = tmp;
    }
  return retval;
}
```

oregonator.cc

```
include <octave/oct.h>

DEFUN_DLD (oregonator, args, ,
  "The ‘oregonator’.\n")
{
  ColumnVector dx (3);
  ColumnVector x (args(0).vector_value ());

  dx(0) = 77.27 * (x(1) - x(0)*x(1) + x(0) - 8.375e-06*pow (x(0), 2.0));
  dx(1) = (x(2) - x(0)*x(1) - x(1)) / 77.27;
  dx(2) = 0.161*(x(0) - x(2));

  return octave_value (dx);
}
```

Ejemplo práctico: guardar una matriz con el formato de las SVM

```
// $Id: curso_octave.mgp,v 1.8 2003/04/03 15:34:42 pablo Exp $
include <octave/oct.h>
include <iostream.h>
include <stdio.h>
DEFUN_DLD (write_svm_file, args, args_out,
 "write_svm_file(X,Y,file)\nGuarda en el archivo file la matriz X con las etiquetas Y.")
{
 Matrix vectors(args(0).matrix_value ());
 ColumnVector labels(args(1).vector_value ());
 string file_name(args(2).string_value());
 FILE *handle = fopen(file_name.c_str(),"w");
 for (int i=0; i<vectors.rows(); i++){
  fprintf(handle, "%i ", int(labels.elem(i)));
  for(int o=0; o<vectors.columns(); o++){
   fprintf(handle, "%i:%f ", o+1, vectors.elem(i,o));
  }
  fprintf(handle, "\n");
 }
 fclose(handle);
 return octave_value();
}
```

Referencias básicas

- http://www.octave.org
- http://octave.sourceforge.net/
- http://octave.sourceforge.net/coda/coda.html
- http://octave.sourceforge.net/new_developer.html

Tiempo para el público

¿Preguntas?

¿Comentarios?

¿Insultos?