Hoja de consulta de MatLab para Cálculo - Pedro Fortuny Ayuso - Uniovi - 18 de septiembre de 2012

Básico

`[a:s:b]` — Vector de `a` a `b` de `s` en `s`.
`linspace(a,b,n)` — Vector de `n` números equidistribuidos entre `a` y `b`.
`pi, exp(1)` — Valor de π y de e (3,14159... y 2,71828...).
`clear, clear(x)` — Limpia todas las variables o solo la `x`.
`whos, who` — Describe o lista las variables definidas.
`double(x)` — Expresa `x` en coma flotante.

Operaciones aritméticas y lógicas

`+ - * /` — Operaciones básicas con números y matrices.
`^` — Elevar a una potencia: `a^b`= a^b.
`.* ./ .^` — Multiplicar, dividir y elevar *elemento a elemento*.
`& | ~` — Operadores lógicos **y** (`&`), **o** (`|`), **negación** (`~`).
`all, any` — Verifica si *todos* (`all`) o *alguno de* (`any`) los elementos de un vector cumplen una condición: `any(x>0)` ¿alguno de los elementos de `x` es > 0?.

Vectores (y matrices)

`linspace(a,b,n)` — Vector de `n` elementos equidistribuidos entre `a` y `b`.
`[a:s:b]` — Vector de números de `a` hasta `b` yendo de `s` en `s`.
`length(v)` — Longitud del vector `v`.
`size(M)` — Tamaño (`filas`×`columnas`) de `M`.
`zeros(n,m)` — Matriz llena de 0s de `n` filas y `m` columnas. Si `m` no está, matriz cuadrada `n`×`n`.
`ones(n,m)` — Matriz llena de 1s de `n` filas y `m` columnas.
`eye(n)` — Matriz cuadrada identidad de tamaño `n`.
`eye(n,m)` — Matriz `n`×`m` en la que la matriz `n`×`n` más a la izquierda es la identidad y el resto son ceros.
`diag(v)` — Matriz cuadrada cuya diagonal es el vector `v`.
`diag(v,n)` — Matriz cuadrada cuya diagonal `n` es el vector `v`. La diagonal 0 es la principal. Si `n`<0, se cuenta debajo de la diag. principal, si `n`>0, encima.
`rand(n,m)` — Matriz de números aleatorios equidistribuidos entre 0 y 1 de `n` filas y `m` columnas. Si `m` no está, matriz cuadrada.

Acceso a elementos

Si `x` es un vector

`x(r)` — El elemento `r` de `x`.
`x(r:s)` — Elementos desde el `r` hasta el `s` del vector `x`.
`x(r:end)` — Elementos desde el `r` hasta el final del vector `x`.
`x(:)` — Todos los elementos de `x`.

Si `A` es una matriz

`A(m,n)` — El elemento `m,n` de `A`.
`A(a:b,c:d)` — Los elementos de las filas `a` hasta la `b` que están en las columnas `c` hasta la `d` (es una submatriz).
`diag(A)` — Los elementos de la diagonal principal de `A`.
`A(a,:), A(:,b)` — Toda la fila `a`, toda la columna `b` de `A`.

Funciones y utilidades

`sqrt` — Raíz cuadrada. `sqrt(x)`: raíz cuadrada de cada elemento de x.
`sin, cos, tan` — Seno, coseno y tangente. P.ej. `sin(x)`, seno de cada elemento de `x`.
`asin, acos, atan` — Funciones inversas de las anteriores (arco seno, etc.)
`exp, log, log10` — Exponencial, logaritmo y logaritmo en base 10. Así, `exp(x)`: exponencial de *cada elemento de* `x`.

Definición de Funciones

`f = @(x) sin(x) - exp(x) .* x.^2` — Define la función `f`, de una variable, que vale $\sin(x) - e^x x^2$ (Úsense siempre `.*`, `./` y `.^`).
`f = @(x,y,z) x .* y - z.^2 .* y ./ x` — Define la función `f` de 3 variables que vale $xy - z^2 y/x$. (Úsense siempre `.*`, `./` y `.^`).
`function [z] = patata(a, b, c) ..... end` — En el fichero `patata.m`, definición de la función `patata`, que devuelve un valor `z` y requiere 3 parámetros (`a`, `b` y `c`).
`function [Sol N M] = pototo(a, b, c) ..... end` — En el fichero `pototo.m`, definición de la función `pototo`, que devuelve tres valores (`Sol`, `N` y `M`) y requiere tres parámetros (`a`, `b` y `c`).

Dibujos

`clf` — Limpia la ventana gráfica.
`hold on/hold off` — Permite (`hold on`) o impide (`hold off`) dibujar sobre la misma gráfica.
`plot(y)` — Si `y` es un vector, dibuja la secuencia de valores de `y`.
`plot(x,y)` — Si `x` e `y` son vectores de la misma longitud, dibuja la gráfica de `x` frente a `y` dada por esos valores.
`plot(x,f(x))` — Si `x` es un vector y `f` una función, dibuja los puntos `(x,f(x))`.
`subplot(n,m,i)` — Divide la pantalla gráfica en `n` filas y `m` columnas y prepara la `i`-ésima (en orden de lectura) para realizar en ella el *siguiente* `plot`.
`ezplot(f(x))` — Si `f(x)` es una función simbólica de una variable, dibuja su gráfica.
`ezplot(f(x,y))` — Si `f(x,y)` es una función simbólica de dos variables, dibuja el conjunto $f(x,y) = 0$.

Hoja de consulta de MatLab para Cálculo - Pedro Fortuny Ayuso - Uniovi - 18 de septiembre de 2012

`axis([x0 x1 y0 y1])` — Redibuja todo en el rectángulo `[x0, x1]`×`[y0, y1]`.
`xlim([x0 x1]), ylim([y0 y1])` — Redibuja una gráfica con la variable `x` entre `x0` y `x1` (o con `y` entre `y0` e `y1`).

Cálculo Simbólico

`syms x y t` — Declara las variables `x`, `y` y `t` como simbólicas.
`f=''expresión en variable syms''` — Define f como una función en las variables simbólicas que contenga.
`g=matlabFunction(f)` — Si f es una función simbólica, define g como la función anónima ("con `@`") equivalente.
`solve(expr, x)` — Si `expr` es una expresión simbólica en la variable `x`, resuelve la ecuación $expr = 0$. Devuelve un *vector columna* de soluciones, que pueden ser exactas (usar `double` para conocer su valor en coma flotante) o en coma flotante.
`limit(f,x,a,'right') - limit(f,x,a,'left')` — Límites por la derecha y por la izquierda de f cuando x tiende a a. La variable x debe haber sido declarada `syms` y `a` es un número.
`diff(f,x,n)` — Derivada $n-$ésima de f con respecto a x. La variable x debe haber sido declarada `syms` y `n` es un número entero.
`int(f,x)` — Primitiva de f respecto de la variable x. Esta variable debe haber sido declarada como simbólica.
`int(f,x,a,b)` — Integral definida de f respecto de la variable x entre a y b, es decir $\int_a^b f(x)\,dx$.
`taylor(f,x,a,n) - taylor(f,a,n) - taylor(f,n)` — Desarrollo de Taylor con respecto a la variable *simbólica* x, de orden n de f en el punto a. Por defecto, la variable es `x`, y el punto a es 0.
`symsum(1/n^2,n,3,inf)` — - suma la serie de término general $1/n^2$ desde $n = 3$ hasta $n = \infty$, es decir $\sum_{n=3}^{\infty} \frac{1}{n^2}$.

Operaciones matriciales avanzadas

`inv(M)` — Matriz inversa *de una matriz cuadrada* (si es que tiene).
`\` — Resolución de un sistema lineal: `A \ b` resuelve el sistema $Ax = b$.
`[A B] = lu(M)` — Factorización LU de la matriz `M`: Se tiene que `AB` = `M` y que `A` es triangular inferior y `B` triangular superior.

Operaciones con polinomios

Se supone que `v` es un vector `v=[`$\mathtt{v}_1, \ldots, \mathtt{v}_n$`]`.

`polyval(v,x)` — Valor de la expresión polinomial $\mathtt{v}_1 x^{n-1} + \cdots + \mathtt{v}_{n-1} x + \mathtt{v}_n$ (el grado *decrece*).
`roots(v)` — Raíces del polinomio $\mathtt{v}_1 x^{n-1} + \cdots + \mathtt{v}_{n-1} x + \mathtt{v}_n$.
`polyderiv(v)` — Polinomio (en forma de vector) derivada de $\mathtt{v}_1 x^{n-1} + \cdots + \mathtt{v}_{n-1} x + \mathtt{v}_n$, es decir, `[`$\mathtt{v}_1/(\mathtt{n}-1), \ldots, \mathtt{v}_{n-2}/2, \mathtt{v}_{n-1}$`]`.

Operaciones numéricas

Se supone que `v` es un vector `v=[`$\mathtt{v}_1, \ldots, \mathtt{v}_n$`]`.

`sum(v), prod(v)` — Suma y producto de todos los elementos de `v`.
`max(v), min(v)` — Máximo y mínimo de los elementos de `v`.
`mean(v), var(v)` — Media y varianza de los elementos de `v`.
`diff(v)` — Vector de *diferencias sucesivas* de `v`. Es decir, $[\mathtt{v}_2 - \mathtt{v}_1, \mathtt{v}_3 - \mathtt{v}_2, \ldots, \mathtt{v}_n - \mathtt{v}_{n-1}]$. Se tiene `length(diff(v))=length(v)-1`.
`diff(v,k)` — Vector de `k`-*diferencias sucesivas* de `v`. Se tiene `length(diff(v,k))=length(v)-k`.

Interpolación

`interp1(x,y)` — Dados dos vectores `x` e `y` de misma longitud n, devuelve el polinomio interpolador de los puntos (x, y) correspondientes. Se devuelve un vector $[\mathtt{v}_1, \ldots, \mathtt{v}_n]$ (que se interpreta como polinomio de grado $n-1$).
`polyfit(x,y,n)` — Devulve el polinomio (como vector) de grado `n` que minimiza el error cuadrático respecto de la nube dada por `x` e `y` (vectores de la misma longitud).
`spline(x,y)` — Interpolante de spline cúbico por los puntos (x, y) dados por `x` e `y` (vectores de la misma longitud). Devuelve un polinomio "a trozos" que se puede evaluar con `ppval`.

Ecuaciones diferenciales & Laplace

`dsolve('ec1','ec2',...,'cond1','cond2',...,'x')` — Resuelve de manera simbólica las ecuaciones `ec1, ec2,...` con condiciones iniciales `cond1, cond2,...` donde la variable independiente es `x`, que por defecto es `t`. Devuelve una función simbólica en la variable `x`.
`D, D2, D3` — Operadores de derivación para escribir una ecuación diferencial. Ejemplo: $y'' - 3xy' + y = \cos(x)$ se escribe `D2y-3*x*Dy+y=cos(x)`.
`laplace(f,t,s)` — Obtiene la transformada de Laplace de `f` (que es una función simbólica de `t`) en la variable `s`. Pueden omitirse `t` y `s`.
`ilaplace(f,s,t)` — Obtiene la transformada inversa de Laplace de `f` (que es una función simbólica de `s`) en la variable `t`. Pueden omitirse las variables.
`ode45(@funcion, int, cond)` — Solución numérica del sistema de edo dada por `@funcion` en el intervalo de la variable `int`, con condiciones iniciales `cond`. La `@funcion` especifica el sistema de edo (ver el manual). Devuelve una trayectoria aproximada. Por ejemplo, para el sistema $\dot{x} = 3x - 2xy$, $\dot{y} = y - xy$, para $t = [0, 6]$, con c.i. $(0.2, 0.6)$ se puede utilizar así:

```
> df=@(t,dx) [3*dx(1)-2*dx(1)*dx(2);dx(2)-dx(1)*dx(2)];
> ode45(df, [0,6], [0.2;0.6]);
```