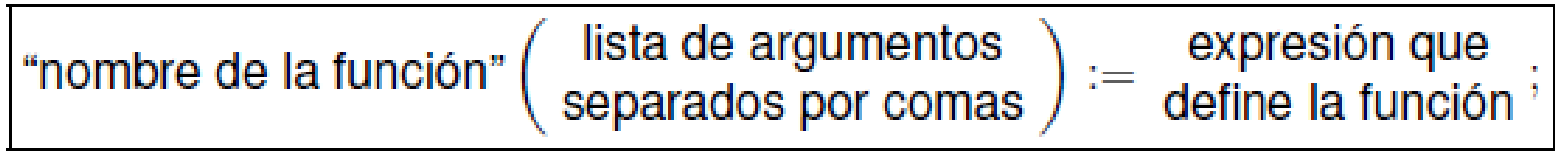
Tutoría 6
Física Computacional I
Grado en Física

Javier Carrasco Serrano, javcarrasco@madrid.uned.es

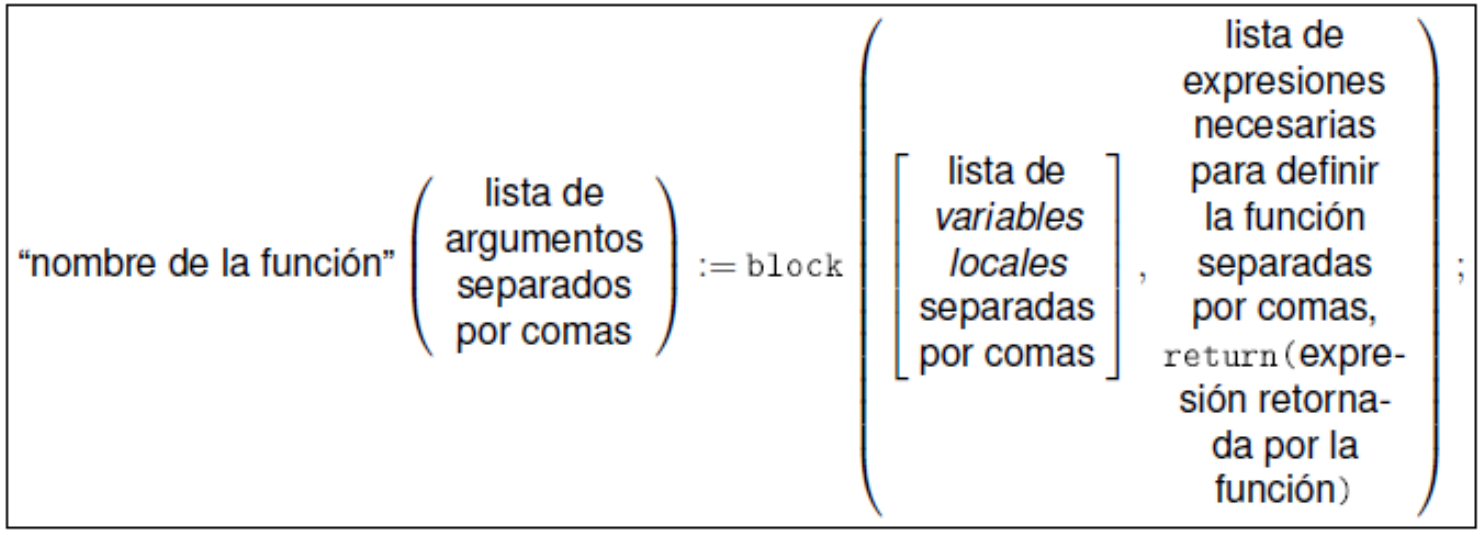
Física Computacional I, Las Tablas

Repaso funciones

- Única expresión:



- Varias expresiones:



Tema 8: Ecuaciones Diferenciales

08

Introducción

- “Ecuaciones diferenciales y sus aplicaciones”, M. Braun → teoría EDOs.
- “Análisis Numérico”, R. Burden, J. D. Faires → métodos numéricos EDOs.
- “A first course in partial differential equations”, H. F. Weinberger → EDPs.
- “Métodos Numéricos”, Infante, J.A. y Rey, J.M. → métodos numéricos EDPs.

- Ecuación diferencial: https://es.wikipedia.org/wiki/Ecuaci%C3%B3n_diferencial
- Ecuación diferencial ordinaria: https://es.wikipedia.org/wiki/Ecuaci%C3%B3n_diferencial_ordinaria
- Ecuación en derivadas parciales:

  https://es.wikipedia.org/wiki/Ecuaci%C3%B3n_en_derivadas_parciales

- Las EDPs son mucho más complejas, y se resuelven transformando la ecuación en un sistema de ecuaciones de EDOs. Muchas no tienen solución analítica.

8.1. Breve vistazo sobre Ecuaciones Diferenciales y su clasificación general

Según el número de variables:

- EDOs: ecuaciones diferenciales ordinarias, una única variable independiente.

$$F = m\frac{d^2x}{dt^2}, \qquad F = \text{fuerza}, \ m = \text{masa}, \ x = \text{posición}, \ t = \text{tiempo}$$

- EDPs: ecuaciones en derivadas parciales, más de una variable independiente.

$$\frac{\partial T}{\partial t} = \alpha\frac{\partial^2 T}{\partial x^2}, \qquad T = \text{temperatura}, \ t = \text{tiempo}, \ \alpha = \text{difusividad térmica}, \ x = \text{posición}$$

Según el orden de la ecuación:

- Ecuación de orden *n*: la derivada más alta que aparece es de orden *n*. Habitual orden 1 y 2.

Según el tipo de condición: es necesaria una condición por cada orden.

- Condición inicial: $x(0) = x_0, \ x'(0) = x_0'$ ...
- Condición de contorno: condiciones en entornos de puntos (orden >1) → valores en varios puntos, o en un intervalo entorno a ese punto. Soluciones más complicadas.

8.1. Breve vistazo sobre Ecuaciones Diferenciales y su clasificación general

Según su linealidad:

- Ecuación lineal: las ecuaciones son funciones lineales respecto a las variables dependientes y sus derivadas (no necesariamente las independientes).
- Ecuación no lineal: las ecuaciones son funciones no lineales respecto a las variables dependientes o sus derivadas → difíciles de resolver numéricamente, muchas sin solución analítica.

Según solución analítica:

- La mayoría de ecuaciones diferenciales no tienen solución analítica. Sí tienen las lineales con coeficientes constantes. Algunas con solución analítica dan lugar a una expresión muy compleja y es más fácil aproximarlas.
- Solución numérica aproximada para el resto → métodos numéricos.

Funciones de EDOs en Maxima

- Solución analítica de EDOs:
  - EDOs lineales: *desolve(ecuaciones, variable);*
  - EDOs lineales o no lineales de hasta orden 2: *ode2(ecuaciones, variable);*
  - EDOs de orden 1 con 1 condición inicial:

    ic1(solucion_ode2, condicion_variable_independiente, condicion_variable_dependiente);
  - EDOs de orden 2 con 2 condiciones iniciales:

    ic2(soln_ode2, cond_var_indep, cond_var_dep, condicion_derivada);
  - EDOs de orden 2 con 2 condiciones de contorno:

    bc2(soln_ode2, cond_entorno_var_indep, cond_entorno_var_dep,
    cond2_entorno_var_indep, cond2_entorno_var_dep);
- Solución numérica de EDOs de orden 1:
  - Método Runge-Kutta: *rk(lista_EDOs, var_indeps, conds_iniciales, dominio);*

    → necesita cargar la librería dynamics: *load("dynamics");*

  https://es.wikipedia.org/wiki/M%C3%A9todo_de_Runge-Kutta

8.2. Funciones de Maxima que debemos aplicar

- Maxima sólo resuelve EDOs de orden 1 y 2.
- Si existe solución analítica, puede aplicar condición inicial o de contorno.
- Si no existe solución analítica, puede resolver sistemas de EDOs de orden 1.
- Problemas con solución analítica:
  - *ode2(eqn, dvar, ivar);*
  - *ic1(solution, xval, yval);*

    → *dvar(xval)=yval*
  - *ic2(solution, xval, yval, dval);*

    → *dvar(xval)=yval, dvar'(xval)=dval*
  - *bc2(solution, xval1, yval1, xval2, yval2);*

    → *dvar(xval1)=yval1, dvar(xval2)=yval2*

    → *ic1, ic2* y *bc2* funcionan sólo si *ode2* es capaz de obtener la solución general!!!

8.2. Funciones de Maxima que debemos aplicar

- Problemas y sistemas con solución numérica (EDOs de orden 1):
  - *rk(EDO, var_indep, cond_inicial, dominio);*
  - *rk(lista_EDOs, var_indeps, conds_iniciales, dominio);*
    - → necesita cargar la librería dynamics: *load("dynamics");*
    - → *dominio* indica el paso de integración para el método Runge-Kutta (por ejemplo, 0.1).

$$\frac{dx_1}{dt} = f_1(x_1, x_2, \ldots, x_n, t)$$
$$\frac{dx_2}{dt} = f_2(x_1, x_2, \ldots, x_n, t)$$
$$\ldots\ldots\ldots\ldots\ldots$$
$$\frac{dx_n}{dt} = f_n(x_1, x_2, \ldots, x_n, t)$$

→

```
rk(ODE, var, initial, domain)

ODE : [f1, f2, ..., fn]
var : [x1, x2, ..., xn]
initial : [x10, x20, ..., xn0]
domain : [t, 0, tfin, paso]
```

$$x_1(0) = x_{10}, \quad x_2(0) = x_{20}, \quad \ldots \quad x_n(0) = x_{n0}$$

8.2.4. Cambios de variable

- Se pueden utilizar cambios de variable para resolver numéricamente EDOs de orden *n* en los problemas de contorno → se reescribe como un problema de orden 1, transformando la/s ecuación/es y condición/es inicial/es:

Cambio de variable: $u = \frac{dx}{dt}, v = \frac{dy}{dt}$

$$\frac{d^2x}{dt^2} = -w_1^2 x + \frac{y}{10} \qquad \frac{du}{dt} = -w_1^2 x + \frac{y}{10}$$

$$\frac{d^2y}{dt^2} = -w_2^2 y - \frac{x}{10} \qquad \frac{dv}{dt} = -w_2^2 y - \frac{x}{10}$$

$$x(0) = x_0, y(0) = y_0 \qquad x(0) = x_0, y(0) = y_0$$

$$x'(0) = x_0', y'(0) = y_0' \qquad \mathrm{u}(0) = x_0', v(0) = y_0'$$

8.2.4. Cambios de variable

- Ejemplo cambio variable (dentro del Ejercicio resuelto 1):

```
eq1 : -diff(x(t),t,2) + (k/m) * ( - x(t) + (y(t) - x(t)) ) - (damp/m) *
   diff(x(t),t,1);
eq2 : -diff(y(t),t,2) + (k/m) * ( - y(t) + (x(t) - y(t)) ) - (damp/m) *
   diff(y(t),t,1);
evsys : [eq1, eq2];
```

8.2.4. Cambios de variable

Sustituimos ahora el cambio de variable propuesto para convertir este sistema de 2 ecuaciones de orden 2 en un sistema de 4 ecuaciones de orden 1. Para ello sustituimos en primer lugar el cambio de variable para dx/dt $(u = dx/dt)$

```
evsys : subst(u(t), diff(x(t), t, 1), evsys);
evsys : subst(diff(u(t), t, 1), diff(x(t), t, 2), evsys);
```

sustituimos a continuación el cambio de variable para dy/dt $(v = dy/dt)$

```
evsys : subst(v(t), diff(y(t), t, 1), evsys);
evsys : subst(diff(v(t), t, 1), diff(y(t), t, 2), evsys);
```

añadimos al sistema las ecuaciones de evolución de las nuevas variables que hemos definido u y v, escrito de manera análoga a como hemos escrito las demás ecuaciones

```
evsys : append([-diff(x(t), t, 1) + u(t), -diff(y(t), t, 1) + v(t)],
   evsys);
```

8.3. Problemas resueltos

Adobe Acrobat
Document

Prueba evaluable de programación con Maxima

Criterios Evaluación

- 35% de la nota final.
- 4 ejercicios, cada uno se puntúa de 0 a 10:
  - El código realiza correctamente lo que se pide → hasta 5 puntos.
  - El código está bien estructurado y se entiende cada parte → hasta 2 puntos.
  - El código es eficiente, no utiliza variables globales → hasta 1.5 puntos.
  - El código está debidamente comentado para entender cada paso, inputs y outputs, y finalidad → hasta 1.5 puntos.
- La calificación de la Prueba de Maxima es la media de los 4 ejercicios, """"siempre y cuando cada ejercicio tenga al menos un 5""""
- Fecha límite: 7 Abril 23h55, se sube al apartado "Entrega de trabajos" del campus virtual.
- Formato: fichero de texto plano con nombre alumn@ → "Carrasco_Serrano_Javier_Maxima.mac"
- La solución de cada ejercicio es una FUNCIÓN.
- Evitar el uso de variables globales.
- Se deben guardar las soluciones a los 4 ejercicios (las 4 funciones) en el mismo fichero de texto plano.

Adobe Acrobat Document

Consejos Prueba Maxima

- La solución de cada ejercicio es una FUNCIÓN.
- Pensar en pseudocódigo, ¿cómo lo haríamos nosotros o nuestro cerebro?, ¿qué entradas necesitamos?, ¿cuál es la salida?, ¿qué operaciones intermedias necesitamos hacer → variables o expresiones auxiliares?
- Pensar qué funciones o comandos de Maxima nos ayudan a resolver cada uno de los pasos del problema que planteamos (solve, subst, plot2d, makelist, append, diff…).
- Prueba a construir una función inicial más simple, y luego añade funcionalidades.
- Comentarios para describir cada paso de la función.
- Utilizar ejemplos de inputs para comprobar que la función hace lo que se pide!!!
- Una vez hecha la función, probar a guardarla como un fichero de texto plano, cerrar la sesión, cargarla (comando batchload) y ver que funciona igual. Nos ayuda a ver si estamos utilizando alguna variable global, alguna función que no hemos cargado, o alguna librería que no hemos cargado.
  - Si la función no se carga con el comando batchload, no se corrige!
- No utilizar tildes, espacios, ni caracteres especiales (ñ, ç…) en el nombre del fichero ni en la ruta en la que se guarde.

2.6. Aprendiendo Maxima

- Guardar archivos desde sesiones de trabajo: archivo → exportar → fichero por lotes maxima (*.mac).
- Es más, para trabajar es mucho mejor guardar en ficheros de texto plano funciones etc, y cargarlas en máxima para su uso.
- Guarrería: copiar y pegar.
- Menos guarrería: archivo → fichero de lotes.
- Recomendable para cargar ficheros → bathcload (filename);

  Filename es el nombre con directorio del fichero:

  C:\Users\Documents\Javier\uned\Tutor\Fisica_Computadores_I\Tutorías\Tutoría 2 - 19 Febrero\ejemplo.mac

  Por defecto busca sólo en los directorios guardados en la variable file_search_máxima.

  Se pueden añadir → append(file_search_máxima, [“mi_directorio”]);

  En la práctica es más cómodo asignar el directorio a una variable global, y llamarla cada vez que hace falta:

```
path_mec : "/home/usuario/bib/Maxima/mecanica/";
```

(obsérvese el uso de comillas dobles, a fin de que la variable `path_mec` sea de tipo *string*) y posteriormente empleamos la función de *concatenación* `concat` para generar el path completo de los archivos a cargar

```
batchload( concat( path_mec, "ec-Newton.mc") );
batchload( concat( path_mec, "ecs-Euler-Lagrange.mc") );
```

Ejemplo código bien comentado

- Cualquier ejercicio del ED que aparezca en el manual de la asignatura (por ejemplo: problemas resueltos Ud 8).
- Cualquier ejercicio de pruebas anteriores resuelto (por ejemplo: 2016-J-R.mc es muy completo).
- Cualquier ejercicio de un examen resuelto (más fáciles de leer, por ejemplo: Ejercicio 1, curso 2010-2011).
- Plantilla ejemplo código comentado → slides siguientes.

Ejemplo código bien comentado

```
/*
PRUEBA EVALUABLE DE PROGRAMACION EN WXMAXIMA
Convocatoria de junio de 2019
Física Computacional I, 1er curso del Grado en Física, UNED, curso 2018-2019
Estudiante: Javier Carrasco Serrano

Este archivo contiene el código en Maxima de las funciones pedidas en la prueba evaluable.
Para cargar el archivo desde wxMaxima se puede ejecutar:
        batchload( concat( path, "Carrasco_Serrano_Javier_Maxima.mac " ) );
donde la variable path es una variable que guarda la cadena que indica la localización del archivo
Carrasco_Serrano_Javier_Maxima.mac en el disco. Por ejemplo path podría ser algo parecido a
        path : "/home/usuario/Fisica-Computacional-1/Maxima/examen/";
*/
```

Ejemplo código bien comentado

```
/*
EJERCICIO 1: a continuación se incluye el código que resuelve el ejercicio 1.
DESCRIPCIÓN DEL EJERCICIO: QUÉ SE ESPERA QUE HAGA LA FUNCIÓN
Input:
        input1:         descripción input1
        input2:         descripción input2
        input3:         descripción input3
Output:
        ¿es una lista? ¿es una única expresión o valor?
        [elemnto1, elemnto2, elemnto3]
        elemento1:      descripción elemento1 de la lista
        elemento2:      descripción elemento2 de la lista
        elemento3:      descripción elemento3 de la lista
*/
```

Ejemplo código bien comentado

```
funcion1(input1,input2,input3) := block( [aux1,aux2],
/*descripción de las expresiones auxiliares*/
        aux1:…,
        aux2:…,
/*descripción de cómo se construyen los outputs*/
        elemento1:…,
        elemento2:…,
        elemento3:…,
/*salida de la función*/
        return([elemnto1, elemnto2, elemnto3]));
```

Prueba Maxima

- Leer enunciados y escribir algo de pseudocódigo.
- ¿Qué nos piden? ¿Inputs? ¿Outputs? ¿Funciones o variables auxiliares? ¿Paquetes auxiliares? ¿Funciones que hayamos visto que se puedan utilizar?

Adobe Acrobat Document

Pruebas Maxima anteriores

- Soluciones Pruebas Maxima años anteriores:
  - Pueden ayudar: 2016-J-R.mc, 2015-J-R.mc, 2014-S-R.mc.

Carpeta
omprimida (en zip

- PDF Tema 9 con exámenes resueltos:
  - Enunciados de los ejercicios resueltos anteriores:

Adobe Acrobat
Document

Gracias!

Javier Carrasco Serrano

Física Computacional I, Las Tablas