Parte 1: "Introducción a MATLAB "

Que es MATLAB?

El nombre MATLAB proviene de *Matrix Laboratory*. Este software que fue inicialmente desarrollado para realizar operaciones con matrices muy fácilmente y ha evolucionado hasta convertirse en una herramienta muy popular en diversos campos de la ingeniería y la Ciencia. MATLAB es un lenguaje de alta performance para cálculo técnico. El mismo integra cálculo, visualización y programación en un entorno de fácil utilización en donde los problemas y las soluciones son expresadas en una notación matemática familiar [1].

Los usos más típicos incluyen:

- Cálculos Matemáticos
- Desarrollo de Algoritmos
- Modelado, simulación y prototipos
- Gráficas Científicas e Ingenieriles

I- Expresiones Fundamentales.

Trabajar en el entorno MATLAB es muy simple ya que la mayoría de los comandos son ingresados de la misma manera que se lo haría matemáticamente. Por ejemplo, tipeando:

>> a=4/3

Dará como resultado:

>> a =
1.3333

Por medio de este comando se le ha asignado el valor 1.3333 a la variable "a". MATLAB reconoce los primeros 19 caracteres de los nombres de las variables requiriendo solamente que el nombre comience con una letra [2]. Otra Particularidad es que MATLAB es sensible a las mayúsculas.
En el caso de estar interesado solamente en el resultado de la expresión y no en la asignación del mismo a alguna variable, tipeando:

>> 4/3

genera el siguiente resultado:

ans =
1.3333
donde la variable "ans" es una variable interna utilizada por MATLAB para almacenar resultados que no han sido asignado a ninguna otra variable. Notar que el contenido de "ans" cambia cada vez que una operación como la antes mencionada es realizada. Por lo tanto si Ud. considera que un resultado puede ser utilizado en subsiguientes cálculos, es conveniente guardarlo en alguna otra variable.

MATLAB tiene algunas variable predefinidas como ser:

i, j = $\sqrt{-1}$
pi = π
inf = ∞
NaN (**N**ot **a** **N**umber, ej. 0/0)

Como estas variables pueden ser sobrescritas, se recomienda usarlas con cuidado ya que por ejemplo si **i** o **j** se usan como índices de elementos de un vector o matriz y luego se desea realizar la siguiente asignación:

>> z = 1 + 2*i (notar la manera de ingresar un no. complejo)

el resultado no será el número complejo esperado. Para restablecer el valor de **i** se puede usar el comando **clear i.**

II- Comando HELP.

MATLAB tiene un help muy bien documentado. Tipeando **help**, MATLAB displaya un índice con aclaraciones que sirve para orientarse a la hora de buscar algún comando en particular. Veamos un ejemplo:

Supongamos que se desea evaluar la siguiente expresión $\mathbf{e^{1+j\,3}}$

tipeando help obtenemos:
» help

HELP topics:

matlab\mine - (No table of contents file)
toolbox\local - Local function library.
matlab\datafun - Data analysis and Fourier transform functions.
matlab\elfun - Elementary math functions.
matlab\elmat - Elementary matrices and matrix manipulation.
matlab\funfun - Function functions - nonlinear numerical methods.
matlab\general - General purpose commands.
matlab\color - Color control and lighting model functions.
matlab\graphics - General purpose graphics functions.
matlab\iofun - Low-level file I/O functions.
matlab\lang - Language constructs and debugging.
matlab\matfun - Matrix functions - numerical linear algebra.
matlab\ops - Operators and special characters.
matlab\plotxy - Two dimensional graphics.
matlab\plotxyz - Three dimensional graphics.
matlab\polyfun - Polynomial and interpolation functions.
matlab\sounds - Sound processing functions.
matlab\sparfun - Sparse matrix functions.
matlab\specfun - Specialized math functions.
matlab\specmat - Specialized matrices.
matlab\strfun - Character string functions.
matlab\dde - DDE Toolbox.
matlab\demos - The MATLAB Expo and other demonstrations.
simulink\simulink - SIMULINK model analysis and construction functions.
simulink\simdemos - SIMULINK demonstrations and samples.
simulink\blocks - SIMULINK block library.
simulink\sb2sl - SystemBuild 3.0 model import into SIMULINK.
mutools\commands - Mu-Analysis and Synthesis Toolbox.: Commands directory
mutools\subs - Mu-Analysis and Synthesis Toolbox -- Supplement
toolbox\signal - Signal Processing Toolbox.
toolbox\ident - System Identification Toolbox.
toolbox\control - Control System Toolbox.
toolbox\symbolic - Symbolic Math Toolbox.

De donde podemos inferir que la información que estamos buscando pude estar dentro de los comandos agrupados en **elementary math functions:**

» help elfun

Elementary math functions.

Trigonometric.

```
  sin        - Sine.
  sinh       - Hyperbolic sine.
  asin       - Inverse sine.
  asinh      - Inverse hyperbolic sine.
  cos        - Cosine.
  cosh       - Hyperbolic cosine.
  acos       - Inverse cosine.
  acosh      - Inverse hyperbolic cosine.
  tan        - Tangent.
  tanh       - Hyperbolic tangent.
  atan       - Inverse tangent.
  atan2      - Four quadrant inverse tangent.
  atanh      - Inverse hyperbolic tangent.
  sec        - Secant.
  sech       - Hyperbolic secant.
  asec       - Inverse secant.
  asech      - Inverse hyperbolic secant.
  csc        - Cosecant.
  csch       - Hyperbolic cosecant.
  acsc       - Inverse cosecant.
  acsch      - Inverse hyperbolic cosecant.
  cot        - Cotangent.
  coth       - Hyperbolic cotangent.
  acot       - Inverse cotangent.
  acoth      - Inverse hyperbolic cotangent.
```

Exponential.

```
  exp        - Exponential.
  log        - Natural logarithm.
  log10      - Common logarithm.
  sqrt       - Square root.
```

Complex.

```
  abs        - Absolute value.
  angle      - Phase angle.
  conj       - Complex conjugate.
  imag       - Complex imaginary part.
  real       - Complex real part.
```

Numeric.

```
  fix        - Round towards zero.
  floor      - Round towards minus infinity.
  ceil       - Round towards plus infinity.
  round      - Round towards nearest integer.
  rem        - Remainder after division.
  sign       - Signum function.
```

Donde hallamos la función **exp**. Para finalmente saber como usar dicha función uno puede tipear **help exp**.Para culminar con el ejemplo, la expresión sería:

>> exp(1+3*i)

A partir de este aquí, Ud. ya esta capacitado para buscar información sobre como usar los comandos de MATLAB. Si se desea realizar algún cálculo para el cual MATLAB no contiene una función específica, Ud. puede crear sus propias funciones. (Ver apéndice Creating Script Files y Creating Functions.)

III- Matrices y Vectores.

Las Matrices son el principal tipo de datos que maneja MATLAB. Las matrices son ingresadas de la siguiente manera:

>> a=[1 2; 3 4]

que da el siguiente resultado

a =
1 2
3 4

Por otra parte una matriz de una sola fila es un vector:

>> b= [5 6]

b=
5 6

Para suprimir la salida en la pantalla cada vez que se asigna una variable use el operador ";"
al final de la expresión:

>> b=[5 6];

El operador ":"

>> v= [1: 0.5:3] Genera in vector con elementos de 1 a 3 con incremento de 0.5
v=
1 1.5 2 2.5 3

Las matrices así como los vectores pueden ser sometidas a operaciones elementales como suma (+), resta (-), multiplicación (*) y división(/ o \ ). Recordando, por supuesto, que algunas de estas operaciones no son conmutativas. Por ejemplo:

>>C= A\B representa la operación $A^{-1} * B$
>>C= A/B representa la operación $A * B^{-1}$

(Analice los resultados de U'*V y V*U' donde V y U son vectores de la misma dimensión y U' es el traspuesto de U)

Cada elemento de un vec. o mat. puede ser accedido por medio de índices:

>> A(2,2)=5
>> A

A=
1 2
3 5

Si se asigna un valor a un elemento que excede la dimensión de la matriz, Automáticamente la dimensión es ampliada [2]. Pruebe A(3,1)= 7.
el operador ":" también se aplica a matrices:

>> A(1:k,j) son los primeros k elementos de la columna j de A [2]

Si V es in vector de dimensión coincidente con la de A, entonces:

>> A=[A; V] añade una fila a A
>> A=[A V'] añade una columna a A
>> B= A(2:3, 1:3) genera una submatriz B con las dos ultimas filas de A

MATLAB contiene comandos para crear matrices especiales como ser la identidad, diagonal, etc. para mayor información use **help elmat**.
Aparte de las operaciones elementales, existen otras como ser el cálculo de autovalores, determinantes, rango etc. Para obtener información sobe estos comandos use **help matfun**.

IV- Polinomios

En MATLAB, los polinomios son representados como vectores fila cuyos elementos son los coeficientes del polinomio en orden decreciente de potencias. Por ejemplo:

$P(s)= s^2 + 3\,s - 5$

se expresa como:

```
>> P=[1 3 -5]
```

las principales operaciones que se pueden realizar con polinomios son:

» help polyfun

Polynomial and interpolation functions.

Polynomials.
```
  roots       - Find polynomial roots.
  poly        - Construct polynomial with specified roots.
  polyval     - Evaluate polynomial.
  polyvalm    - Evaluate polynomial with matrix argument.
  residue     - Partial-fraction expansion (residues).
  polyfit     - Fit polynomial to data.
  polyder     - Differentiate polynomial.
  conv        - Multiply polynomials.
  deconv      - Divide polynomials.
```

Sugerencias: vea los comandos del grupo general

Esta introducción esta pensada solo para proveer al alumno de conocimientos básicos. Para mayor información, consulte los manuales disponibles en la cátedra. Otra importante observación es que en el **toolbox de Control** hay muchos comandos que pueden ser útiles a lo largo del desarrollo de la materia.

Biblografía y Referencias. (Disponible en la cátedra de DSF)

[1]- MATLAB, The Language of Technical Computing- “Getting Started with MATLAB”
[2]- Ehirch, Leonard and William - “ Using MATLAB to Analize and Design Control Systems” The Benjamin/Cummings Publishing Company, Inc.

Parte 2: "Introducción a SIMULINK"

INTRODUCCIÓN

Simulink es un ambiente interactivo para modelar una amplia variedad de sistemas dinámicos, pudiendo ser estos lineales, no lineales, discretos, de tiempo continuo y sistemas mixtos. Permite realizar diagramas de bloques con operaciones click-and-drag, cambiar parámetros del modelo y visualizar resultados durante una simulación.
Es también un sistema abierto, que permite al usuario escoger, adaptar y crear componentes o subistemas. Simulink se apoya en el ambiente Técnico Computacional de MATLAB.
MATLAB y su grupo de Toolboxes ofrecen un conjunto amplio de herramientas de ingeniería y matemática para definir algoritmos, analizando datos y visualizando resultados. Juntos, SIMULINK y MATLAB proveen un entorno integrado para construir modelos versátiles y simular dinámicos, diseñando y testeando ideas nuevas.

SIMULACIÓN Y ANÁLISIS

Simulink y los Toolboxes de Matlab permiten moverse sobre varios niveles de modelado diseño y simulación. Se pueden utilizar los modelos Simulink para simulación, linealización del modelo, determinación de los puntos de equilibrio, optimización de parámetros y análisis.

SIMULACIÓN

Los diagramas de bloques Simulink facilitan un entorno interactivo para la simulación de sistemas lineales, no lineales y discretos. La simulación se puede realizar desde menúes descolgables o desde la línea de comandos de Matlab. Los resultados pueden ser vistos durante la simulación usando osciloscopios (Scopes) o bloques gráficos (Graph blocks); y grabados en un archivo o transferidos al espacio de trabajo de Matlab para su posterior análisis o procesamiento.
Simulink permite realizar el análisis de modelos cambiando los parámetros del mismo mientras se lleva a cabo la simulación.

MODELADO

REPRESENTACIÓN DE SISTEMAS DINÁMICOS CON DIAGRAMAS DE BLOQUE

Simulink permite desarrollar modelos de sistemas dinámicos mediante ecuaciones y diagramas de bloque. Se pueden crear modelos lineales o no lineales, de tiempo discreto o continuo, o modelos híbridos utilizando "drag and drop" (arrastrar y dejar) para mover los componentes desde una biblioteca de bloques y conectándolos entre sí usando el mouse.

LIBRERÍA DE BLOQUES

La Librería de Bloques de Simulink contiene centenares de componentes agrupados de la siguiente manera:

- *Sources (fuentes),*
- *Sinks (visualizadores / salidas),*
- *Discrete (discreto),*
- *Linear (lineal),*
- *Nonlinear (no lineales),*
- *Blocksets & Toolboxes (herramientas extras).*

Los bloques de entradas y de salidas se usan para intercambiar vectores entrada-salida de simulación con el entorno MATLAB y archivos de datos. Los bloques de tiempo discreto permiten modelar y simular subsitemas con datos muestreados tales como sistemas de control digital y procesamiento de señales. Otros bloques utilizan comandos MATLAB por sus posibilidades adicionales de análisis.
MODELOS DE ELEVADA COMPLEJIDAD

Usando Simulink se pueden modelar sistemas complejos y grandes tan fácilmente como los sistemas simples, sin límites sobre la cantidad de bloques o conexiones.
Los modelos complejos se crean agrupando bloques en subsistemas. Para trabajar con subsistemas, se pueden construir los modelos usando enfoques top-down y bottom-up.

DOCUMENTACIÓN GRÁFICA

Simulink permite capturar y llevar al portapapeles de Windows sus pantallas (diagramas de bloques, gráficos) en dos formas posibles (Metafiles o Bitmap). Estas pueden ser manipuladas por cualquier procesador de textos o imágenes.
NOTAS

Los atributos de todos los bloques pueden personalizarse incluyendo parámetros internos, orientación, tamaño, color, título y fuente.

OPERABILIDAD SOBRE DISTINTAS PLATAFORMAS

Simulink corre sobre MS-Windows, plataformas Macintosh, estaciones de trabajo UNIX, y plataformas VMS que son sistemas operativos estándar usados en la industria.
Los modelos de Simulink pueden transferirse de una plataforma a otra conservando sus características y funcionalidades.
Los modelos de Simulink son compatibles con software de control estándar.

BIBLIOTECA DE BLOQUES SIMULINK

Entre los bloques más importantes de cada biblioteca se encuentran los siguientes:

Sources:

- Sine wave, ramps and squares waves (ondas senoidales, rampas y cuadradas).
- Noise (Ruido).

Sinks:

- Scopes and graph blocks (osciloscopios y bloques gráficos).
- File output (salidas hacia archivos).
- Output to the Matlab Workspace (salidas hacia el espacio de trabajo de Matlab)

Discrete:

- Transfer functions and state-space blocks (bloques de función transferencia y espacio de estados).

Linear:

- Transfer functions, state-space, gain blocks (bloques de función transferencia, espacio de estados y ganancia)

Nonlinear:

- Limiters, hysteresis blocks, and quantizers (limtadores, bloques histéresis, y cuantizadores).
- Logical and relational operators (operadores lógicos y relacionales).

Connections:

- Multiplexing and demultiplexing blocks (bloques multiplexores y demultiplexores).
- Input ports and output ports (puertos de entrada y salida).

BlockSets & Extras:

- Simulink_extras (librerías de bloques extras entre las que se encuentran bloques extras visualizadores, discretos, lineales, transformaciones, flips-flops y bloques para linealización).
- Los BlockSets que aparezcan dependerá de los Toolboxes que se encuentren instalados.

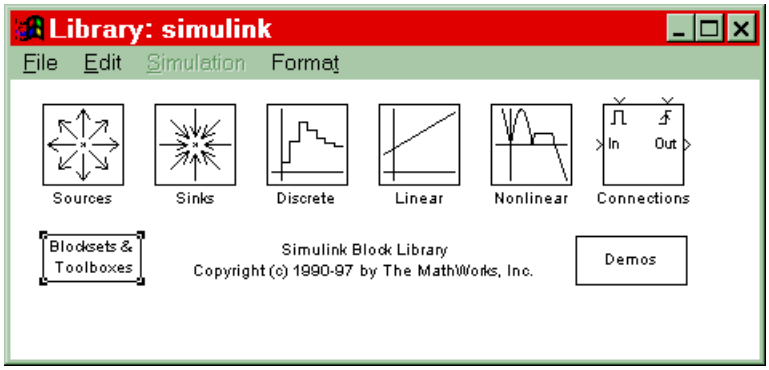
Estos últimos mencionados (Blocksets), son colecciones de bloques Simulink que se agrupan en bibliotecas separadas y se corresponden con los Toolboxes de Matlab instalados.

CONSTRUCCIÓN Y SIMULACIÓN DE UN MODELO SIMPLE

Para la utilización de Simulink es necesario familiarizarse con la manipulación de bloques y la construcción de modelos. Para esto es neceario también conocer la variedad de bloques existentes. Y por último manejar las herramientas de análisis provistas por Simulink.
La construcción y simulación de un modelo simple, descriptas de aquí en más, permiten la introducción en cada uno de estos conceptos.

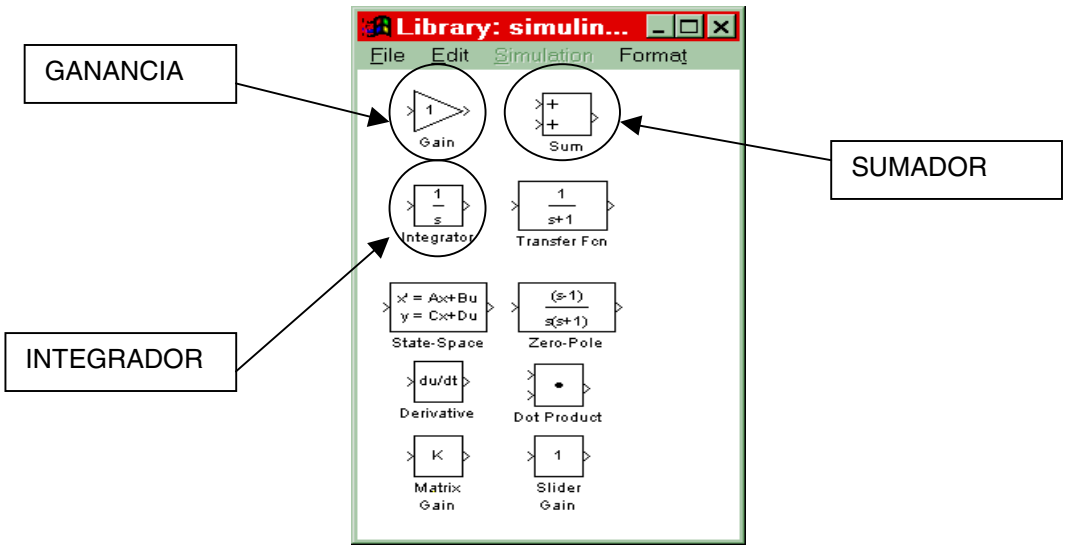
EJEMPLO:
Para ejecutar Simulink se deberá tipear *simulink* a continuación del símbolo del sistema en Matlab o bien hacer click

sobre el icono  que se encuentra en la barra de herramientas de la ventana principal de Matlab. Una vez que se cargue el programa aparecerá una ventana como la siguiente:



y además se abrirá otra ventana nombrada untitled donde se deberá armar el modelo deseado.
Cada una de las opciones que aparece contiene un grupo de bloques con los cuales se implementará el DB. Estos grupos de bloques constituyen la biblioteca de simulink.
El sistema a modelar y simular es un P_{T1} excitado con una entrada escalón. Para la realización de este ejemplo deberá procederse de la siguiente manera:

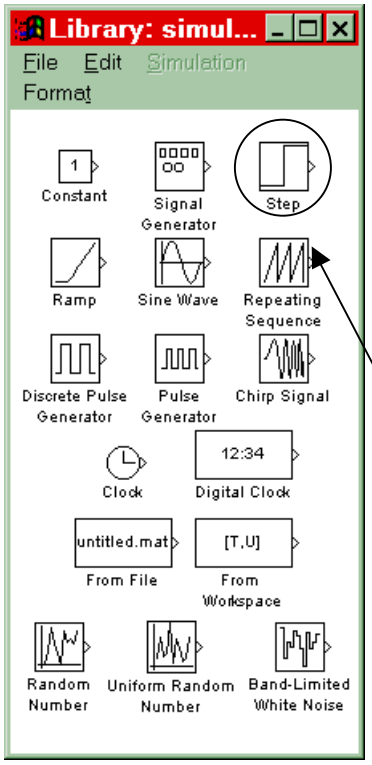
- ✓ Para comenzar con el DB se realiza doble click en el icono Linear con los cual aparecerá la siguiente pantalla.



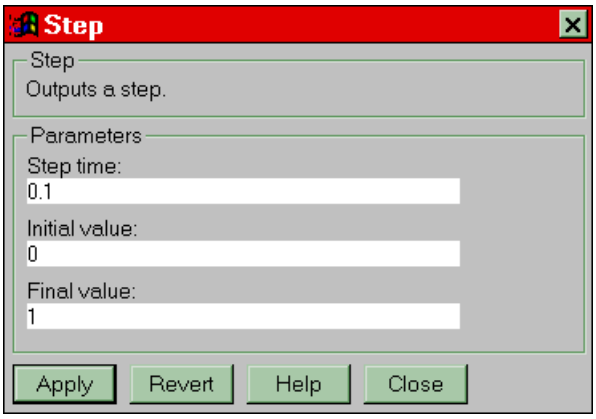
- ✓ Los bloques que se necesitan llevar a la ventana untitled, en este caso, son los bloques integrador, sumador y ganancia. Estos pueden ser copiados o simplemente arrastrados con el mouse. Para ello

cuando el puntero de este se encuentre sobre el bloque que se desea arrastrar se presiona manteniendo apretado el botón izquierdo del mouse. Con el botón apretado se desplaza el mouse hasta colocar el bloque sobre la ventana en blanco en la posición más conveniente.

- ✓ Los bloques tienen indicadas sus entradas y salidas. Estas pueden no quedar en la ubicación más cómoda, por lo cual los bloques pueden ser rotados. Para rotar un bloque se posiciona el mouse sobre el bloque y se hace un click quedando este seleccionado; luego se puede proceder de dos formas, (1) manteniendo apretada la tecla Control se pulsa la tecla R hasta conseguir la orientación deseada; (2) en el menú desplegable **Style** seleccionar la opción **Orientation** .
- ✓ Para cambiar los signos del sumador se realiza un doble click (botón izquierdo) sobre el bloque, con lo cual aparece una nueva ventana donde se encuentran los signos del sumador pudiendo ser modificados. Incluso se pueden agregar o quitar, con lo cual se aumenta o disminuye la cantidad de entradas del bloque.
- ✓ Se accede a modificar la ganancia del bloque Gain haciendo doble click sobre el mismo.
- ✓ El valor inicial del integrador puede ser seteado en forma análoga a los casos anteriores.
- ✓ Volviendo a la ventana de Simulink, se hace doble click sobre el icono **Sources**. De allí se arrastra hacia la ventana de trabajo la fuente tipo escalón (Step).



Haciendo un doble click sobre el bloque Step se puede indicar el instante en el cual se produce el escalón, el valor inicial y el valor final del mismo. En este caso el escalón se producirá en t = 0.1 seg, pasando de 0 a 1.

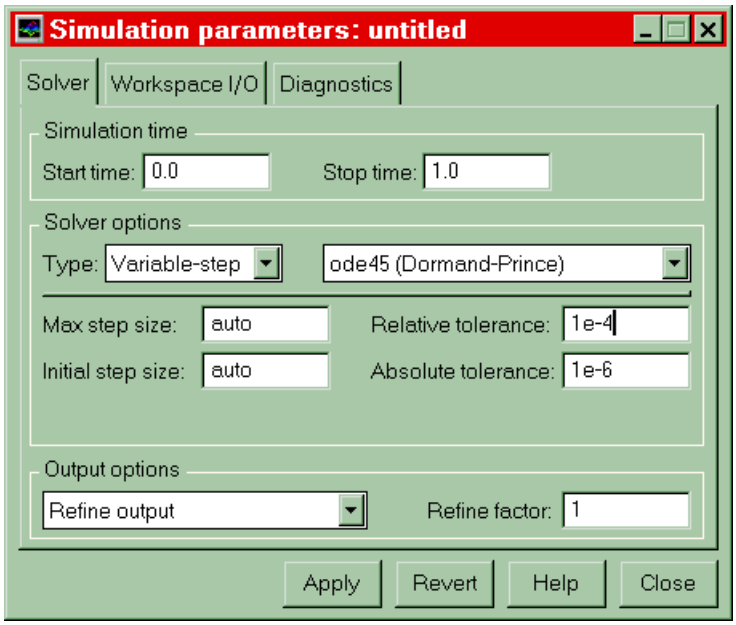


Entrada tipo

- ✓ Se debe ahora intcluir el bloque donde se visualizará el resultado de la simulación. Para ello en la ventana Simulink se hace doble click sobre el icono **Sinks**. El bloque scope es el más cómodo para utilizar ya que permite ajustar las escalas de ambos ejes en forma independiente y muy sencilla.
- ✓ Para poder realizar la simulación se deben vincular los bloques entre sí. Esto se consigue uniendo los bloques con líneas de flujo de señal. Para ello situamos el puntero del mouse sobre la salida del bloque que se desea vincular y manteniendo presionado el botón izquierdo se lo desplaza hacia la entrada del bloque correspondiente donde se suelta el botón del mouse.
- ✓ Antes de proceder con la simulación del sistema se deben definir sus parámetros; método de resolución, tiempo de simulación, paso de integración (en caso de que el método escogido lo requiera), etc. Estos se definen en el menú **Simulation**, dentro del submenú **Parameters**.

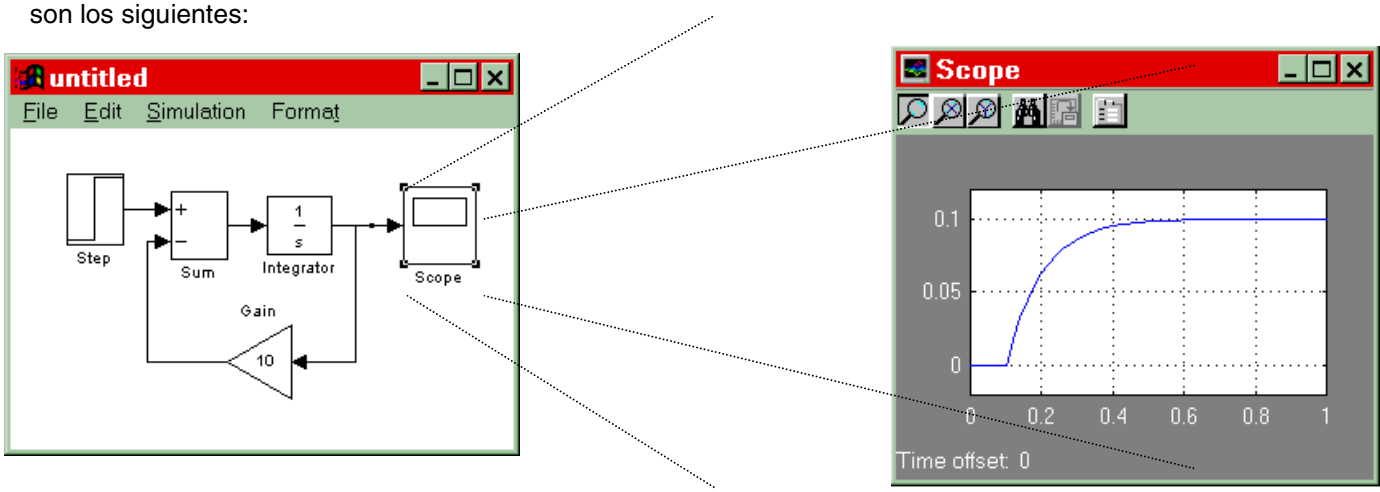
  El tiempo total de simulación debe ser suficiente para que el sistema evolucione hasta su estado estacionario, es decir de 3 a 5 veces la mayor constante de tiempo del sistema (esto si se desea ver toda la dinámica de la evolución). En el ejemplo se tiene una única constante de tiempo que es 0,1 y será tomada en cuenta al establecer el tiempo de simulación.

  En cuanto al paso de integración, dependiendo del método este puede ser fijo o variable y además puede ser acotado en forma manual o automática. Luego se explicará cada método destacando sus características principales; pero mientras tanto se utilizará para este ejemplo el método de paso variable ode45 (Dormand-Prince) con los siguientes parámetros:



Se puede observar además que se ha dejado al algoritmo la tarea de establecer el máximo y mínimo paso de integración (al escribir la palabra 'auto' en el campo correspondiente).
Además de la pestaña del solver, pueden observarse otras dos correspondientes a Workspace I/O y Diagnostics.
La primera de ellas permite asignar entradas y salidas desde y hacia matrices definidas en el espacio de trabajo (Workspace) de Matlab.
La última pestaña es la que permite definir los eventos que serán chequeados para su diagnostico, asignando además el tipo de interrupción que provocará cada evento. También permite habilitar o deshabilitar el método de detección de cruces por cero (zero crossing), que será de mucha utilidad en algunos modelos.

- ✓ El paso siguiente es la simulación propiamente dicha, la opción **Start** del menú **Simulation** inicia la misma. Luego haciendo doble click sobre el bloque Scope aparecerá una nueva ventana, correspondiente al bloque de visualización de gráficas que se ha incluido en el modelo, donde se realizará la gráfica de la evolución de variable de salida. El DB correspondiente y la gráfica resultante son los siguientes:



- ✓ En el caso de que existiera un lazo algebraico Simulink realizará iteraciones en cada paso de integración para determinar la solución de las ecuaciones implícitas utilizando la técnica de Newton-Raphson. Sin embargo es posible que en algún caso el método no resulte convergente a la solución, y de todas maneras en general resultará conveniente prescindir de este método utilizado por Simulink, lo cual puede realizarse dos maneras distintas: (a) en los casos que sea posible, resolver el lazo (por ej: utilizando álgebra de bloques) antes de ingresar el modelo; (b) modificar el modelo incorporando un tiempo muerto igual al paso de integración (bloque *Memory* del grupo *Nonlinear*). En este último caso se debe poner atención, estudiando cual es la ubicación más adecuada del tiempo muerto en el lazo para no alterar substancialmente el comportamiento del modelo.
- ✓ En el caso de ser necesario borrar algún bloque o línea, se sitúa el puntero del mouse sobre el componente a borrar, se lo selecciona con un click del botón izquierdo y luego se presiona de la tecla Delete.
- ✓ Para grabar el modelo se utiliza la opción **Save** el menú **File.** Este será almacenado como un archivo .mdl con el nombre que se asigne. Puede ser recuperado en el momento que se desee desde el mismo menú **File** con la opción **Open**.
- ✓ Cabe señalar que al realizar un doble click sobre un bloque, este muestra una breve explicación de su funcionamiento.

Estas notas tienen sólo un carácter introductorio, para obtener información más detallada referirse a [3]. Cabe destacar que los modelos que se realicen con Simulink 2, generalmente no se podrán leer luego con Simulink 1. Es por eso que se aconseja guardar los resultados de las simulaciones realizadas con Simulink 2 en archivos con formato ascii para luego, si no se tiene disponible una PC con esta versión instalada puedan reproducirse con cualquier graficador. Para hacer esto suponiendo que se quiera guardar un resultado que está dado por dos vectores X e Y (los cuales podrían representar por ejemplo el tiempo y la salida del DB del modelo); entonces debe construirse una matriz que contenga ambos vectores tipeando desde el workspace de Matlab lo siguiente:

>> **Z = [X ; Y] ' ;**

Con lo que se genera la matriz Z. Luego se puede proceder a grabar estos datos en formato ascii mediante el siguiente comando:

>> **save 'A: \ resultado.txt' Z –ascii –tabs**

Generándose el archivo con nombre resultado.txt en el directorio raíz de la unidad A. Luego podrán leerse estos datos y graficar los resultados con cualquier programa que lo admita (por ej.Origin, Excel, etc), inclusive con versiones anteriores de Matlab.

REFERENCIAS (disponibles en la cátedra)

[1] SIMULINK "User 's Guide", by The Math Works Inc. USA
[2] SIMULINK "The Ultimate Simulation Enviroment", 1995 by The Math Works Inc. USA
[3] SIMULINK "Using Simulink (Version 2)", by The Math Works Inc. USA
[4] SIMULINK "Simulink 2.1 New Features", by The Math Works Inc. USA

Parte 3: “Métodos Numéricos”

I- Introducción.

En esta parte del trabajo práctico se examinaran los métodos numéricos y las facilidades que emplea simulink para realizar la simulación digital de sistemas dinámicos.
Para una mejor comprensión de los puntos a tratar se recomienda la previa lectura de la publicación ‘MÉTODOS NUMÉRICOS EN LA SIMULACIÓN DIGITAL DE SISTEMAS DINÁMICOS”-A02LAB94.

Ajuste de Parámetros de Simulación y Elección del Método de Resolución (Solver).

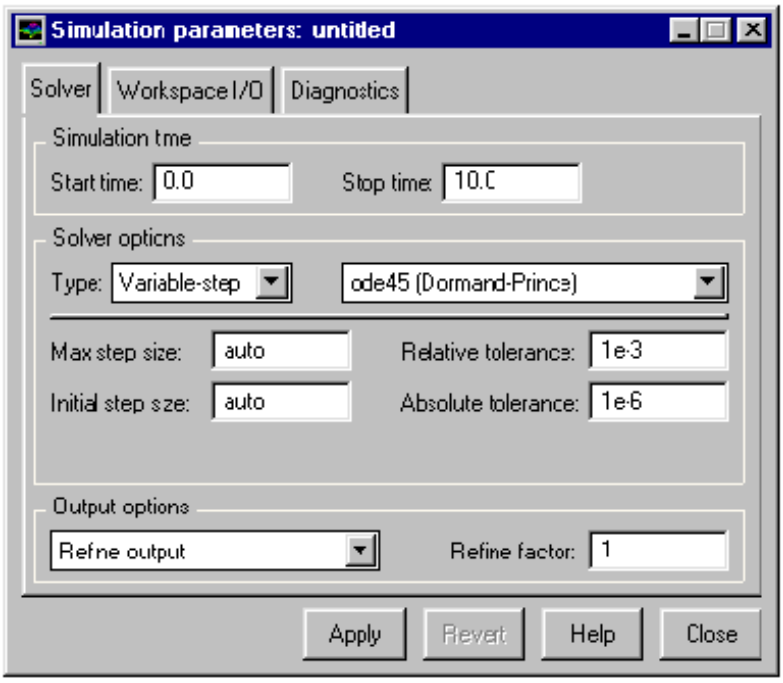
Los parámetros de simulación son accedidos a través de la opción **Parameters** del menú **Simulation** de la ventana donde se encuentra el modelo**.** SIMULINK muestra entonces la ventana de dialogo de parámetros de simulación que consta de tres paginas para el manejo de dichos parámetros:

-**Solver**: Esta página permite modificar los tiempos de inicio, parada de simulación y el método de resolución conjuntamente con sus parámetros y opciones de salida.

-**Workspace I/O**: Esta página permite intercambiar información con espacio de trabajo de MATLAB.

- **Diagnostics**: Esta página permite modificar los niveles de los mensajes de alerta que pueden surgir durante la simulación.

La Página del Solver.



Simulation Time: Aquí se pueden modificar el tiempo inicial y final de simulación. Por defecto los valores son 0 y 10 seg. respectivamente. Es conveniente notar que el tiempo de simulación y el tiempo que tarda la computadora en realizar dicha simulación no es el mismo. Por ejemplo, si **stop time** es ajustado en 15 seg. esto no significa que la máquina estará simulando durante 15 seg. El tiempo que demanda la simulación depende generalmente de una serie de factores como ser: la complejidad del modelo, tamaño del paso de integración, método de integración, velocidad de la computadora, memoria, etc.

Solvers: Simulink provee un conjunto de diferentes métodos para resolver la simulación. Debido a la diversidad de modelos de sistemas dinámicos, algunos de estos métodos responden de manera más eficientes que otros para cada caso en particular.
Simulink presenta dos grupos de métodos de resolución:

- **Variable-step:** Son aquellos que modifican automáticamente el tamaño del paso durante la simulación; a su vez, proveen control de error y detección de cruces por cero.

- Fixed-step: Mantienen el tamaño del paso de integración a través de toda la simulación y contrariamente a los métodos de paso variable, estos no proveen control de error y detección de cruces por cero.

Métodos de Paso Variable (Variable-step Solvers).

- ode45 (ordinary differential equation solver of 4^{th} and 5^{th} order): Este es el método por defecto si simulink detecta que el modelo es de estado continuo (ver Discrete (variable-step)). ode45 es un método de un solo paso (Ver [3]) basado en la fórmula de Runge-Kutta de orden 4 y 5 (Par de Dormand-Princce). En general este método arroja resultados satisfactorios para la mayoría de los modelos continuos y resulta ser un bueno como primera aproximación cuando no se conoce mucho del sistema en estudio.

-ode23: Es un método de un solo paso basado en la fórmula de Runge-Kutta (2,3) (par de Bogacki-Shampine). Este método resulta más eficiente que el ode45 cuando las tolerancias del error no son tan exigentes y el modelo presenta un leve grado de rigidez (mild stiffness).

- ode113: Es un método multipaso (Ver [3]) de orden variable de Adams-Bashforth-Moulton. Puede resultar más eficiente que ode45 cuando las tolerancias del error se tornan muy exigentes.

- **ode15s:** Es un método multipaso de orden variable basado en fórmulas de diferenciación numérica (NDFs). Si bien las NDFs están relacionadas con las fórmulas de diferenciación por atraso (DBFs o método de Gear), estas NDFs son mucho más eficientes. Este método es recomendable si el modelo es stiff (Ver [3]) o bien el ode45 falla o resulta muy lento. El orden de este método puede variar entre uno y cinco. Cuando mayor es el orden del método, si bien mayor será la precisión de los cálculos efectuados, puede que el mismo resulte inestable. Por tal motivo, si el modelo es stiff y requiere mayor estabilidad es aconsejable reducir el orden a dos o usar ode23s. La opción para modificar el orden de este método esta disponible desde la ventana del solver una vez que ode15s fue seleccionado.

- ode23s: Es un método basado en la formula modificada de Rosenbrock de segundo orden. Al ser un método de un solo paso puede en algunos casos ser más eficiente que ode15s cuando las tolerancias no sean muy exigentes.

-discrete (paso variable)**:** Es el método por defecto que elige simulink cuando detecta un modelo de estado discreto.

Métodos de paso fijo (Fixed-step Solvers).

-ode5: Es la versión de paso fijo de ode45.
-ode4: Está basado en la formula de Runge-Kutta de cuarto orden (RK4).
-ode3: Es la versión de paso fijo de ode23.
-ode2: Está basado en el método de Heun también conocido como formula de Euler mejorada.
-ode1: Esta basado en la formula de Euler.
-discrete (paso fijo)**:** Este método no realiza integraciones por lo que es aconsejable para modelos sin estados para los cuales la detección cruces por cero y la detección de errores no es importante.

Opciones del Método Elegido (solver Options).

Maximum step size: Este parámetro es el límite superior para el tamaño que puede tomar el paso en los métodos de paso variable. Cuando la opción se encuentra en **AUTO**, el parámetro es determinado del siguiente modo: $h_{max} = (t_{stop} - t_{start}) / 50$. Por lo general este valor es suficiente pero puede haber casos que requieran una modificación del mismo para evitar que se pierdan ciertos comportamientos del sistema a causa de que el solver tome pasos muy grandes. Si la simulación demanda mucho tiempo, puede que el

paso sea muy grande para que el solver encuentre la solución. Por otra parte si el modelo presenta un comportamiento periódico o cuasi periódico, es aconsejable tomar el paso máximo como ¼ del período.

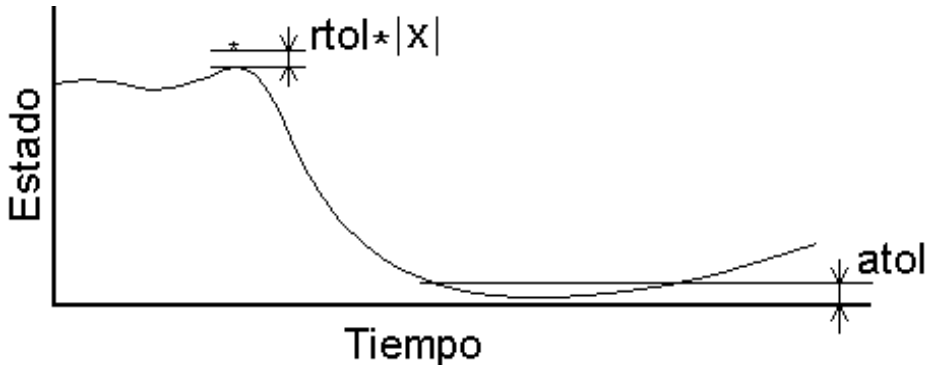
-Initial step size: Por defecto SIMULINK analiza las derivadas de los estados para determinar el tamaño inicial del paso y de este modo no perder información producto de haber comenzado con un paso demasiado grande. Este parámetro es solo un valor sugerido como tamaño para el primer paso ya que el solver reducirá automáticamente el paso si los criterios de error no son satisfechos.

-Tolerancias de Error: Los solvers utilizan técnicas locales de control de errores en cada paso de integración. Durante cada paso, no solo se determina el valor de cada estado sino también el error local que es el error estimado de los estados calculados.
Los dígitos en el nombre de cada rutina como 2 y 3 en ode23 se refieren al orden del método usado (ver[3]). De este modo, ode23 emplea una aproximación de orden 2 y una de orden 3 y en base a la diferencia en los resultados obtenidos se estima el error local antes mencionado.
Este error local es comparado con la cota de error aceptable que es función de las tolerancias relativa y absoluta. Si el error local estimado es mayor que la cota de error aceptable para ***alguno*** de los estados calculados, entonces el solver reduce el paso y recalcula.

-Tolerancia Relativa (rtol): Mide el error relativo al valor de cada estado y representa un porcentaje del valor del dicho estado. Por defecto este parámetro es $10^{-3,}$ lo que significa que el estado calculado tendrá una precisión del 0.1%.

-Tolerancia Absoluta (atol): Representa un umbral para el valor del error. Esta tolerancia representa la cota máxima del error a medida que los estados se aproximan a cero.

El error para el i-esimo estado, e_i, debe satisfacer: $\boldsymbol{e_i \leq max(rtol * |x_i|, atol)}$



Opciones de salida.

Estas opciones permiten controlar la cantidad de puntos que genera la salida de la simulación de tres maneras posibles:

-Refine Factor: Cuando el aspecto de la salida se asemeja a una poligonal, este parámetro provee un número entero de puntos intermedios adicionales. Por ejemplo si este parámetro se fija en dos, la salida tendrá aparte de los puntos generados en cada paso un punto adicional intermedio en cada paso. Para suavizar el aspecto de la salida es más rápido usar esta opción que disminuir el paso de integración ya que el solver genera estos puntos adicionales por medio de una formula de extensión continua sin cambiar la cantidad de pasos. Esta opción se encuentra disponible solo para los métodos de paso variable. El valor por defecto es 1 con el cual no agrega ningún punto adicional.

Produce Additional output: Esta opción permite adicionar directamente tiempos en los cuales son de interés el conocimiento del estado.

Produce Specified Output Only: Esta opción provoca que solamente se generen los puntos aquí especificados.

Ejemplo.
Supóngase que la simulación arroja datos en los siguientes tiempos: 0 2.5 5 8.5 10

Refine output = 2 generará resultados en los instantes:

0, 1.25, 2.5, 3.75, 5, 6.75, 8.5, 9.25, 10

Produce adicional output = [0:10] generará resultados en los siguientes instantes de tiempo:

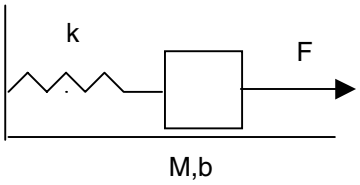
0, 1, 2, 2.5, 3, 4, 5, 6, 7, 8, 8.5, 9, 10

Produce specified output only = [0:10] generará resultados en los siguientes instantes de tiempo:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

II- Desarrollo de Trabajo Práctico.

Consideremos el siguiente sistema:



1) Obtener un DB tomando como salida la posición x de la masa M, y pasarlo a Simulink.

2) Calcular la F.T. del modelo.

3) I) Para los parámetros M=1; b=2; k=1 calcular los polos y simular utilizando paso fijo. Elegir h=.1; h=1 y h=2 y observar cada salida para entrada F(t)=μ(t).

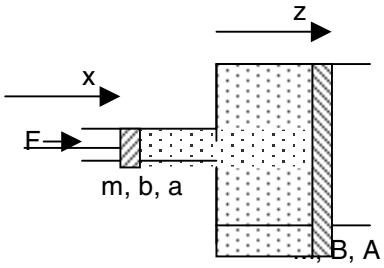
II) Modificar b=200 y recalcular los autovalores. Simular mediante paso fijo, luego utilizando paso variable y el algoritmo ODE45 y finalmente ODE15s. Graficar h vs. t (ver [4]) mediante el comando plot de Matlab.

III) Repetir el paso anterior para b=2; M=1 y k=10000.

IV) Idem I) pero con entrada F(t)=sen(100t)

Lazos algebraicos:

Considerar el siguiente sistema:



Suponiendo que el líquido es incompresible y que la presión es constante en todo el volumen, hacer un DB sin derivadores tomando dz/dt como salida.

Simular el DB en Simulink utilizando los siguientes parámetros:

A=10 a=1 m=1 M=5 b=1 bM=5 mediante un algoritmo de paso fijo, tomando h=0.01.

Resolver el lazo algebraico y simularlo nuevamente con los mismos parámetros.

Bibliografía y Referencias,

[1] The Math Works inc. Simulink - Dynamic Simulation for Matlab® 'USING SIMULINK'
[2] The Math Works inc. Simulink - Dynamic Simulation for Matlab® 'SIMULINK 2.1 NEW FEATURES'
[3] Junco, Serra 'MÉTODOS NUMÉRICOS EN LA SIMULACIÓN DIGITAL DE SISTEMAS DINÁMICOS"- A02LAB94. publicación de la cátedra de DSF.
[4] Moler C. ' GOLDEN ODE's New ordinary equation solvers for Matlab and Simulink ' Matlab Networks & Notes summer 1996. pp.11-13
[6] Press W., Flannery B., Teukolsky S., Vetterling W. 'NUMERICAL RECIPES The art of Scientific Computing.' Cambridge University Press 1986.
[7] Demidowitsch B.P., Maron I.A., Schuwalowa 'MÉTODOS NUMÉRICOS DE ANALISIS' Paraninfo Madrid 1980.