

# TECNOLOGÍAS PARA LA AUTOMATIZACIÓN

## SISTEMAS DE CONTROL AUTOMÁTICO

### Guía N° 1: Clasificación y Representación de sistemas para la automatización Contenidos

#### SECCIÓN TEORÍA

#### 1 Introducción a los sistemas de control

- i Modelado
- ii Control. Características principales
- iii Señales
- iv Lazo abierto. Lazo cerrado

#### 2 Clasificación de los sistemas de control

- i Sistema lineal (L)
- ii Sistema invariante en el tiempo (TI)
- iii Sistema lineal invariante en el tiempo (LTI)
- iv Sistema causal
- v Sistema sin memoria
- vi Sistema inversible

### 3 Modelo de sistema en el espacio de estado

## SECCIÓN PRÁCTICA

### 4 Ejercitación

- ii Elija una opción y justifique para un sistema dado
- iii De acuerdo al siguiente enunciado, realice el modelo correspondiente.
- iv Tomamos el ejemplo del modelo de teoría de masa - resorte y codificamos con MatLab para obtener: 4.1. Forma matricial. 4.2. Solución de sistema. 4.3. Gráficas de la solución del sistema
- v Ejemplo del Planteo de un modelo matemático para sistema continuo. (Homework)

## SECCIÓN PRÁCTICA CON SOFTWARE

### 5 Introducción a la aplicación MatLab

### 6 Gráficos

## SECCIÓN PRÁCTICA COMPETENCIAS



### Ejercitación

# 1. Introducción a los sistemas de control

Los **sistemas de control** juegan un papel crucial en el mundo moderno, especialmente en los **procesos automatizados**. El conocimiento de las herramientas y la dinámica del control es esencial para garantizar el crecimiento, la automatización, la reducción de costos y el incremento de ganancias. En

definitiva, estos sistemas permiten la optimización de los recursos disponibles, lo cual es fundamental en un entorno cada vez más competitivo y tecnológicamente avanzado.

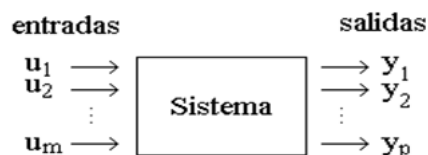
**Controlar** un sistema implica tener la capacidad de gestionar y mejorar los procesos de manera eficiente, permitiendo a las organizaciones adaptarse rápidamente a los cambios y demandas del mercado. Además, en un mundo cada vez más conectado, es vital comprender los problemas y desafíos que surgen con la integración de tecnologías avanzadas en diversos sectores.

Entender la importancia de los sistemas de control nos lleva a iniciar un estudio detallado de los mismos. Comenzaremos por explorar las **características de los sistemas y señales** y aprenderemos a describirlos mediante **modelos matemáticos**. Este conocimiento nos permitirá desarrollar y aplicar estrategias de control efectivas que optimicen el rendimiento y la eficiencia de los sistemas automatizados.

**Un sistema es una combinación de componentes que actúan interconectados, para cumplir un determinado objetivo.**

**Gráficamente podemos pensar un sistema como un rectángulo o caja negra y variables que actúan sobre el sistema. Las flechas que entran ( $u$ ) son excitaciones o entradas y las flechas que salen ( $y$ ) son variables producidas por el sistema o salidas.**

**Podemos pensar en sistemas como la relación de entradas-salidas.**



Un sistema de control es una configuración estructurada y coordinada de componentes que recibe **entradas**, **procesa** estas **entradas** y genera **salidas** para influir en el **comportamiento** de un sistema con el fin de alcanzar un **objetivo** específico, generalmente utilizando mecanismos de **retroalimentación** para mantener la **estabilidad** y compensar las **perturbaciones**.

## i. Modelado

El modelado de sistemas es una herramienta fundamental para entender y predecir el comportamiento de diversos sistemas, ya sean físicos como una placa electrónica (hardware) o abstractos como un conjunto de relaciones matemáticas. Mediante el **modelado**, podemos codificar el **funcionamiento interno de un sistema**, creando representaciones matemáticas que capturan sus **dinámicas** y **comportamientos**.

Los modelos matemáticos pueden ser implementados en software específicos facilitando su análisis y optimización a través de herramientas computacionales avanzadas.

En esencia, el **modelado** de sistemas proporciona una comprensión profunda y detallada de cómo operan los **sistemas**, permitiendo su **control y mejora continua**.

### Modelado Entrada - Salida

Uno de los enfoques de modelado más útiles para propósitos de control es el **Modelado Entrada/Salida**. Este tipo de modelado se centra en describir la relación entre los estímulos aplicados a un sistema (entradas) y las respuestas observadas (salidas).

El **Modelado de Entrada/Salida** se basa en la idea de que, para cualquier entrada dada, el sistema generará una salida específica.

Este enfoque es especialmente útil en situaciones donde la complejidad interna del sistema es elevada o donde los detalles internos no son completamente conocidos o accesibles. A través de este modelado, se puede desarrollar un entendimiento práctico y aplicable del comportamiento del sistema en su totalidad.

La **relación entre la entrada y la salida** se representa matemáticamente mediante la **Función de Transferencia** del proceso. La Función de Transferencia es una herramienta poderosa que permite analizar y diseñar sistemas de control, ya que proporciona una representación simplificada y manejable de cómo las entradas se transforman en salidas. Esta función se deriva a partir de las **ecuaciones diferenciales** que describen el sistema y se expresa en términos del **dominio de Laplace**, lo que facilita el análisis de la estabilidad y la respuesta del sistema.

En las siguientes guías de trabajo, abordaremos con profundidad el concepto de Función de Transferencia.

## ii. Control. Características principales

Controlar un proceso consiste en mantener constantes ciertas variables, prefijadas de antemano. Las variables controladas pueden ser, por ejemplo: Presión, Temperatura, Nivel, Caudal, Humedad, etc.

Un **sistema de control** es el conjunto de elementos, que hace posible que otro sistema, proceso o planta permanezca fiel a un programa establecido.

Como ejemplo de un sistema de control analicemos la temperatura de nuestro cuerpo. Si la temperatura sube por encima de  $37^{\circ}\text{C}$ , se suda, refrescando el cuerpo. Si la temperatura tiende a bajar de  $37^{\circ}\text{C}$ , el cuerpo, involuntariamente, comienza a temblar, contracción muscular que calienta nuestro cuerpo, haciendo que se normalice nuestra temperatura. Por tanto, en este caso:

- Sistema de medida o sensores -> Células nerviosas de la piel

- Señal de consigna -> 37°C
- Acción de control de la temperatura -> Sudar o temblar

Los elementos de un sistema de control trabajan juntos para **monitorear, regular y ajustar** el **comportamiento** del **sistema objetivo**, asegurando que opere de acuerdo con las **especificaciones deseadas**. Los sistemas de control son fundamentales para mantener la estabilidad, eficiencia y eficacia de los procesos automatizados, permitiendo un funcionamiento consistente y predecible.

## Características Principales

### Entradas (Inputs)

Señales o datos que representan las variables que se desean controlar o que afectan el comportamiento del sistema.

Ejemplos: temperatura, velocidad, posición, presión.

### Salidas (Outputs)

Las respuestas del sistema controlado que resultan del procesamiento de las entradas.

Ejemplos: movimiento de un motor, cambio de temperatura, ajuste de válvulas.

### Controlador

El componente del sistema de control que procesa las entradas y determina las acciones necesarias para obtener la salida deseada.

Puede ser un dispositivo físico (como un termostato) o un algoritmo/software (como un PID control).

### Sistema Controlado (Planta)

El proceso o dispositivo cuya conducta se desea controlar.

Ejemplos: un horno industrial, un vehículo, una máquina herramienta.

### Retroalimentación (Feedback)

Un mecanismo por el cual parte de la salida del sistema se devuelve a la entrada para ajustarla y corregir cualquier desviación del estado deseado.

La retroalimentación puede ser positiva o negativa, pero los sistemas de control generalmente utilizan retroalimentación negativa para mantener la estabilidad.

### Perturbaciones (Disturbances)

Factores externos que afectan el rendimiento del sistema y que deben ser compensados por el controlador.

## iii. Señales

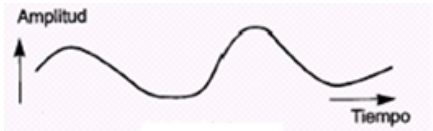


Las señales son manifestaciones físicas de procesos naturales o artificiales de diversas naturalezas.

Características comunes:

- Dependencia de Variables: Las señales son funciones de una o más variables independientes.
- Portadoras de Información: Contienen información sobre el comportamiento o la naturaleza del fenómeno físico en cuestión.

**En un sistema de control, una **señal** se puede definir como una **representación física** de una cantidad variable que transmite **información** sobre el estado o el comportamiento del sistema.**

Los dispositivos, circuitos y sistemas electrónicos manipulan estas señales eléctricas para monitorear, regular y controlar diversos procesos.

Tipo de señal eléctrica	Gráfica según tipo de señal eléctrica
<b>Señal analógica</b>  Número infinito de valores que tiene una variación continua en el tiempo.	 Una gráfica con el eje vertical etiquetado como 'Amplitud' y el eje horizontal como 'Tiempo'. Muestra una onda sinusoidal continua que oscila entre dos niveles.
<b>Señal digital</b>  Número finito de valores que tiene una variación discreta de valores en el tiempo.	 Una gráfica con el eje vertical etiquetado como 'Amplitud' y el eje horizontal como 'Tiempo'. Muestra una onda de pulso que cambia entre varios niveles discretos de amplitud.
<b>Señal digital binaria</b>  Dos valores concretos; 1 y 0. La señal eléctrica sólo puede adoptar dos niveles de tensión.	 Una gráfica con el eje vertical etiquetado como 'Amplitud' y el eje horizontal como 'Tiempo'. Muestra una onda de pulso que cambia entre dos niveles discretos de amplitud, representando los valores binarios 1 y 0.

## Señal continua

Una **señal continua** es una **función continua de una o varias variables**. Ejemplos de estas señales son los distintos tipos de electro gramas usados en medicina, que son señales unidimensionales, ya que se representan por una o varias curvas en función del tiempo. Sin embargo, los distintos tipos de radiografías son señales bidimensionales y los resultados de la tomografía axial computarizada y la resonancia nuclear magnética son señales tridimensionales.

Las señales continuas, entendidas como funciones continuas de una o más variables, son fundamentales en numerosos aspectos de estos sistemas. Estas señales representan fenómenos físicos que varían de manera continua en el tiempo y/o en el espacio, tales como temperatura, presión, velocidad, corriente eléctrica, entre otros.

Además, muchos de los **métodos matemáticos y herramientas analíticas** que se utilizan en el **control de sistemas** están específicamente desarrollados para tratar con **señales continuas**.

Al abordar señales como funciones continuas de una o más variables, se facilita el uso de técnicas avanzadas de análisis y diseño, tales como la **transformada de Laplace** y la **transformada de Fourier**, que son indispensables para la resolución de problemas en el **dominio del tiempo** y de la **frecuencia**. Estas técnicas permiten descomponer y estudiar las señales en sus componentes fundamentales, facilitando la comprensión de su comportamiento y la implementación de estrategias de control adecuadas.

### Proceso analógico y proceso digital

El proceso analógico es aquel que involucra señales analógicas, es decir señales continuas. Dichas señales se definen en un intervalo de tiempo continuo y se describen por ecuaciones diferenciales.

El proceso digital involucra señales digitales, se representa por medio de variables discretas, es decir que pueden cambiar en valores discretos de tiempo. Dichos instantes, se notan como  $kT$ , o  $kt$  ( $k=0, 1, 2, \dots$ ).  $T$  es el período de muestreo.

En el estudio de sistemas de control, es fundamental comprender y manejar tanto los procesos analógicos como los digitales. Cada uno de estos enfoques ofrece ventajas únicas y se emplea en diferentes contextos para alcanzar objetivos específicos en el control y la automatización de sistemas.

Los **procesos analógicos** se caracterizan por el uso de **señales continuas** que varían de manera continua en el tiempo. Estas señales representan directamente las magnitudes físicas, como la temperatura, la presión, la velocidad y la corriente eléctrica. Los sistemas de control analógicos operan de manera continua y suelen emplear componentes electrónicos como amplificadores operacionales, resistencias, condensadores e inductores para procesar estas señales.

El estudio de procesos analógicos es esencial porque muchos fenómenos naturales y sistemas físicos son inherentemente analógicos. Comprender cómo trabajar con estas señales permite a los ingenieros de control diseñar sistemas que interactúen de manera precisa y eficiente con el mundo real. Las técnicas de análisis mencionadas como la **transformada de Laplace** y las **ecuaciones diferenciales** se utilizan ampliamente para **modelar y diseñar sistemas de control analógicos**.

Por otro lado, los **procesos digitales** implican el uso de **señales discretas**, que **varían en pasos finitos** en el tiempo. Estas señales se representan mediante **valores binarios (0 y 1)** y son procesadas por computadoras y microcontroladores. **Los sistemas de control digital convierten las señales analógicas del mundo real en señales digitales a través de convertidores analógico-digitales (ADC), y luego procesan estas señales utilizando algoritmos implementados en software.**

El estudio de procesos digitales es crucial en la actualidad debido a la creciente adopción de tecnologías digitales en todos los aspectos de la vida moderna. Los sistemas de control digital ofrecen ventajas significativas, como la capacidad de implementar algoritmos complejos, la flexibilidad en el diseño y la facilidad de integración con otros sistemas digitales. Además, permiten la implementación de técnicas avanzadas de control, como el control predictivo y el control adaptativo.

### Integración de proceso analógico y digital

En muchos sistemas de control modernos, es común encontrar una integración de procesos analógicos y digitales. Esta integración permite aprovechar lo mejor de ambos mundos: la **precisión y continuidad de las señales analógicas**, junto con la **flexibilidad y potencia de procesamiento de las señales**

**digitales.** Por ejemplo, en un sistema de control industrial, los sensores pueden medir variables físicas de manera continua (analógica), pero el procesamiento y la toma de decisiones se realizan de manera digital.

## iv. Lazo abierto. Lazo cerrado

### Lazo abierto

La entrada se elige en base en la experiencia que se tiene con dichos sistemas para producir el valor de salida requerido, sin embargo, no se ve modificada por el cambio en las condiciones de operación externas. La principal diferencia es que no existe información que se alimente de regreso (retroalimentación).

**Ventajas.** Son más sencillos y en consecuencia de bajo costo, y con buena confiabilidad. Sin embargo, con frecuencia son inexactos, si se descomponen se reemplazan fácilmente.

Un tostador automático es un sistema de control de lazo abierto, que está controlado por un regulador de tiempo. El tiempo requerido para hacer tostadas, debe ser anticipado por el usuario, quien no forma parte del sistema. El control sobre la calidad de la tostada (salida) es interrumpido una vez que se ha determinado el tiempo, el que constituye tanto la entrada como la acción de control.



### Lazo cerrado

Se tiene una señal de retroalimentación hacia la entrada desde la salida, la cual se utiliza para modificar la entrada de modo que la salida se mantenga constante a pesar de los cambios en las condiciones de operación.

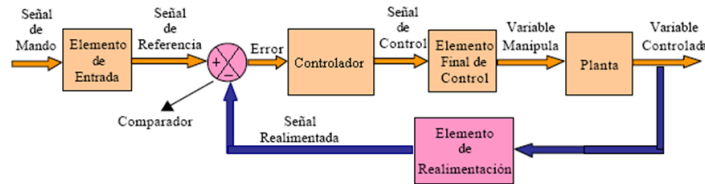
**Ventajas.** Más exactos y menos sensible a las perturbaciones y a los cambios en las características de los componentes. La velocidad de respuesta se incrementa.

Un mecanismo de piloto automático y el avión que controla, forman un sistema de control de lazo cerrado (por realimentación). Su objetivo es mantener una dirección específica del avión, a pesar de los cambios atmosféricos.

El sistema ejecutará su tarea midiendo continuamente la dirección instantánea del avión y ajustando automáticamente las superficies de dirección del mismo (timón, aletas, etc.) de modo que la dirección instantánea coincida con la especificada. El piloto u operador, quien fija con anterioridad el piloto automático, no forma parte del sistema de control.

El climatizador de un automóvil. Una variación en la salida o en otra variable, se mide, y el controlador, modifica la señal de control, para que se estabilice, el sistema, ante la nueva situación. El sistema o la planta se mide en todo momento. El control tiene información de cómo está la salida (Planta).





## 2. Clasificación de los sistemas de control

La clasificación de los sistemas de control es esencial para entender y diseñar estrategias efectivas de control que optimicen el rendimiento de los procesos automatizados. Clasificar los sistemas permite una mejor comprensión de sus características y comportamiento. Esta clasificación no solo facilita el análisis teórico de los sistemas, sino que también es crucial para seleccionar y aplicar las técnicas de control adecuadas.

### i. Sistema lineal (L)

Un sistema se define como **lineal** si cumple con dos principios fundamentales:

- Principio de **homogeneidad** (proporcionalidad o escalabilidad)
- Principio de **superposición** (o aditividad).

#### Principio de Homogenidad, Proporcionalidad o Escalabilidad

El sistema cumple con el **principio de homogeneidad** si la **entrada multiplicada por un escalar** produce una **salida del sistema multiplicada por el mismo escalar**.

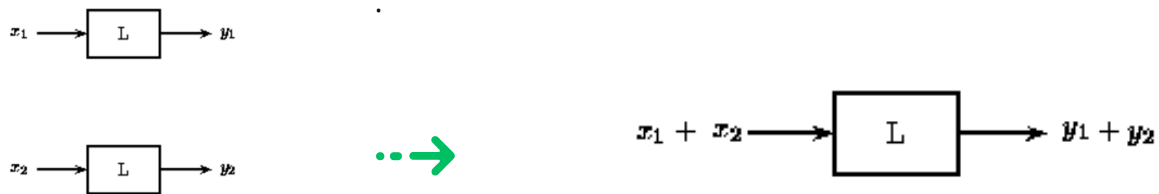


La entrada  $x$  del sistema lineal  $L$  da la salida  $y$

Si  $x$  es multiplicada por un escalar  $\alpha$  y es pasada a través del mismo sistema, la salida también será multiplicada por  $\alpha$ .

#### Principio de Superposición o Aditividad

El sistema cumple con el **principio de superposición** si **dos entradas son sumadas y pasadas a través del sistema lineal**, la **salida** será equivalente a la **suma de las dos entradas evaluadas individualmente**.



Las dos propiedades analizadas pueden enunciarse en una sola:

**Si las entradas  $x$  e  $y$  son multiplicadas por factores  $\alpha$  y  $\beta$ , respectivamente, entonces la suma de estas entradas dará la suma de las salidas individualmente.**



Las propiedades de **homogeneidad** y **superposición** que caracterizan un **sistema lineal** indican que el sistema responde de manera **proporcional a las entradas** y que la **respuesta a la suma de múltiples entradas es igual a la suma de las respuestas individuales a esas entradas**.

## Ejemplos

- Sea  $f(x) = x / 2$

- Principio de Homogeneidad

$$\begin{array}{l} x \rightarrow \boxed{L} \rightarrow x/2 \\ ax \rightarrow \boxed{L} \rightarrow a \cdot x/2 \end{array}$$

- Principio de Superposición

Análisis de la salida de cada variable por separado

$$\left. \begin{array}{l} x_1 \rightarrow \boxed{L} \rightarrow x_1/2 \\ x_2 \rightarrow \boxed{L} \rightarrow x_2/2 \end{array} \right\} x_1/2 + x_2/2 = (x_1 + x_2)/2$$

Análisis de la salida a partir de la suma de dos variables

$$x_1 + x_2 \rightarrow (x_1 + x_2)/2$$

La suma de la salida de cada entrada independiente coincide con la salida de las entradas sumadas. **EL SISTEMA ES LINEAL**

- Sea  $f(x) = x + 1$

- Principio de Homogeneidad

$$\begin{array}{lcl} x \rightarrow \boxed{\text{L}} & \rightarrow & x+1 \\ ax \rightarrow \boxed{\text{L}} & \rightarrow & a(x+1) \end{array}$$

- Principio de Superposición

Análisis de la salida de cada variable por separado

$$\begin{array}{lcl} x_1 \rightarrow \boxed{\text{L}} & \rightarrow & x_1 + 1 \\ x_2 \rightarrow \boxed{\text{L}} & \rightarrow & x_2 + 1 \end{array} \quad \left. \vphantom{\begin{array}{lcl} x_1 \rightarrow \boxed{\text{L}} & \rightarrow & x_1 + 1 \\ x_2 \rightarrow \boxed{\text{L}} & \rightarrow & x_2 + 1 \end{array}} \right\} \begin{array}{l} x_1 + 1 + x_2 + 1 = \\ (x_1 + x_2) + 2 \end{array}$$

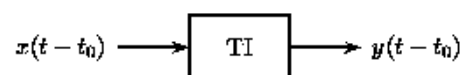
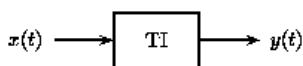
Análisis de la salida a partir de la suma de dos variables

$$x_1 + x_2 \rightarrow (x_1 + x_2) + 1$$

La suma de la salida de cada entrada independiente no coincide con la salida de las entradas sumadas.  
**EL SISTEMA NO ES LINEAL.**

## ii. Sistema invariante en el tiempo (TI)

Un sistema **invariante en el tiempo TI (Time- Invariant)** tiene la propiedad de que **cierta entrada siempre dará la misma salida, independientemente del momento en que fue aplicada al sistema.**



La entrada está definida en un tiempo  $t$

La entrada está definida en un tiempo  $t_0$  después

En un sistema **invariante en el tiempo**, ambas **salidas** serán **idénticas**, excepto que la segunda figura estará **retrasada en  $t_0$** .

Si un sistema es **invariante en el tiempo** puede ser descrito por **ecuaciones diferenciales**. Los sistemas invariantes en el tiempo son modelados con ecuaciones de **coeficientes constantes**. Una ecuación diferencial con coeficientes constantes significa que los parámetros del sistema no van cambiando a través del tiempo y que la entrada dará el mismo resultado independientemente del momento en que fue aplicado.

### Ejemplos

- Sea  $y(t) = x(t)$

Sea  $x_1(t)$  con salida  $y_1(t)$

Si  $x_2(t) = x_1(t - t_0)$ , se debe demostrar que las salidas son iguales, o sea  $y_2(t) = y_1(t - t_0)$

a. Salida  $y_1(t - t_0)$

Dada  $x_1(t)$  la salida  $y_1(t) = x_1(t)$

Si  $x_1(t - t_0)$  la salida es  $y_1(t - t_0) = x_1(t - t_0)$ .

b. Salida  $y_2(t)$

Dada  $x_2(t)$  la salida  $y_2(t) = x_2(t) = x_1(t - t_0)$ .

Por lo desarrollado en los puntos a y b, se demuestra que  $y_2(t) = y_1(t - t_0)$ . **EL SISTEMA ES INVARIANTE EN EL TIEMPO.**

- Sea  $y(t) = t \cdot x(t)$

Sea  $x_1(t)$  con salida  $y_1(t)$

Si  $x_2(t) = x_1(t - t_0)$ , se debe demostrar que las salidas son iguales, o sea  $y_2(t) = y_1(t - t_0)$

a. Salida  $y_1(t - t_0)$

Dada  $x_1(t)$  la salida  $y_1(t) = t \cdot x_1(t)$

Si  $x_1(t - t_0)$  la salida es  $y_1(t - t_0) = (t - t_0) \cdot x_1(t - t_0)$ .

b. Salida  $y_2(t)$

Dada  $x_2(t)$  la salida  $y_2(t) = t \cdot x_2(t) = t \cdot x_1(t - t_0)$ .

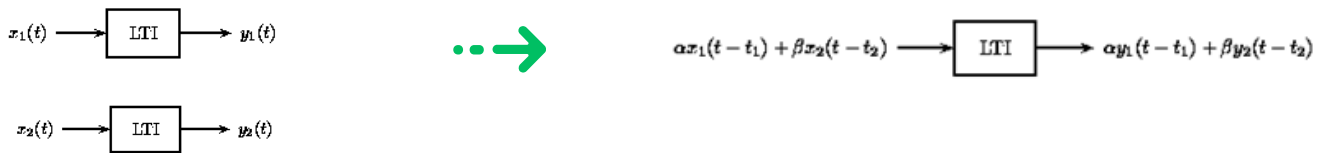
Por lo desarrollado en los puntos a y b, se demuestra que  $y_2(t) \neq y_1(t - t_0)$ . **EL SISTEMA ES VARIANTE EN EL TIEMPO.**

### iii. Sistema lineal invariante en el tiempo (LTI)



Representa una versión escalada y desplazada en el tiempo de la entrada y salida respecto de la figura anterior

Como los sistemas **LTI** son **subconjuntos de los sistemas lineales**, éstos obedecen al **principio de superposición**. El efecto de aplicar el tiempo invariante a la definición de sistema lineal de la sección anterior es el siguiente:



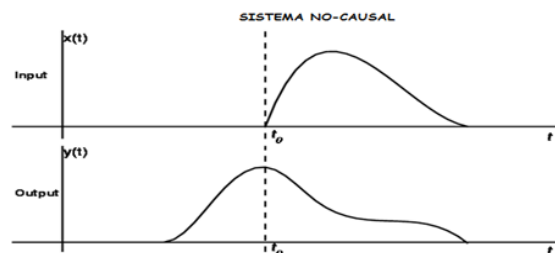
## iv. Sistema causal

Un sistema es **causal** si **no depende de valores futuros de las entradas para determinar la salida.**

Esto significa que si la **primera entrada es recibida en tiempo  $t_0$ , el sistema no deberá dar ninguna salida hasta ese tiempo.** Un ejemplo de un sistema no-causal puede ser aquél que al “detectar” que viene un entrada da la salida antes de que la entrada llegue.

Un sistema causal es un tipo de sistema en el cual la **salida** del sistema en cualquier instante de tiempo  **$t$**  depende únicamente de las **entradas presentes y pasadas**, y no de las futuras.

Esta característica es esencial en la mayoría de los sistemas de control reales, ya que la información futura generalmente no está disponible.



La gráfica representa un sistema NO causal, donde la salida se produce antes que la entrada, la cual ocurrió después en el tiempo.

### Ejemplos

- $y(n) = x(n)^2$  Es un sistema no lineal y causal.
- $y(n) = x(2n)$  Es un sistema lineal y no causal.

## v. Sistema invertible

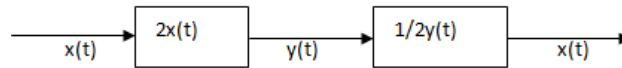
Un sistema es **invertible** si **distintas entradas producen distintas salidas.**

Dicho de otra forma, un sistema es **invertible** si al **observar su salida podemos determinar la entrada.**

Un sistema inversible es un sistema en el que, dado el conocimiento de la **salida** y el **estado del sistema**, es posible determinar de manera única la **entrada que produjo esa salida**. En otras palabras, un sistema inversible permite recuperar las entradas originales a partir de las salidas observadas.

## Ejemplos

- Sea  $y(t) = 2x(t)$ . Su sistema inverso es  $x(t) = 1/2 y(t)$ . Al interconectarlos en serie se obtiene la entrada original como salida. El sistema es inversible.



- Sea

$$y(n) = \sum_{k=0}^n x(k)$$

Para este sistema, la diferencia entre dos valores sucesivos de salida es precisamente el último valor de entrada. Por tanto, el sistema inverso es:

$$\begin{aligned} y(n) &= \sum_{k=0}^n x(k) = x(0) + x(1) + x(2) + \dots + x(n-1) + x(n) \\ y(n-1) &= \sum_{k=0}^{n-1} x(k) = x(0) + x(1) + x(2) + \dots + x(n-1) \\ \hline y(n) - y(n-1) &= x(n) \end{aligned}$$

- Sea  $y(t) = \text{seno}(x(t))$ . Esta función es periódica con período  $2\pi$ , lo que significa que para múltiples valores de  $x(t)$ , el seno puede tener el mismo valor. Por ejemplo:  $\text{seno}(\pi/2) = \text{seno}(5\pi/2) = 1$ . El sistema NO es inversible.

## vi. Sistema sin memoria

Un sistema es **sin memoria** si su **salida** para cada valor de su **variable independiente** depende sólo de la **entrada** en ese **mismo instante de tiempo**.

Un sistema **sin memoria** depende únicamente de la **entrada actual**.

Un sistema **con memoria** depende de las **entradas y/o estados anteriores además de la entrada actual**. El sistema **guarda información** sobre las entradas y/o estados anteriores, lo que le permite tener "**memoria**" de su comportamiento pasado.

Un sistema **tiene memoria** si su **salida** en el tiempo actual depende de **entradas futuras**. Aunque típicamente los sistemas con memoria dependen de entradas pasadas, en este caso específico donde la salida en el tiempo actual depende de entradas futuras, la dependencia de una entrada futura indica que el sistema necesita "**recordar**" o conocer algo que aún no ha ocurrido, lo cual implica una forma de "**memoria**" en un sentido más amplio.

## Ejemplos

- Sea  $y(t) = 2x(t)$ . Sistema sin memoria.

- Sea

$$y(n) = \sum_{k=0}^{n-1} x(k) \quad \text{Sistema con memoria.}$$

- Sea  $y(t) = 2x(t+1)$ , para  $t \geq 0$ . Sistema con memoria.

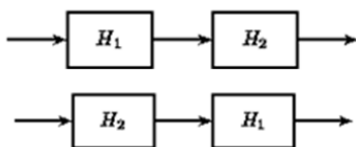
## vii. Sistema en serie y paralelo

Un sistema **en serie** representa una configuración donde la **salida de un bloque** se convierte en la **entrada del siguiente**, formando una **cadena lineal de bloques**.

Un sistema **en paralelo** tiene bloques que reciben **la misma entrada simultáneamente**, y sus **salidas** se combinan, generalmente **sumándose**, para formar la salida total del sistema.

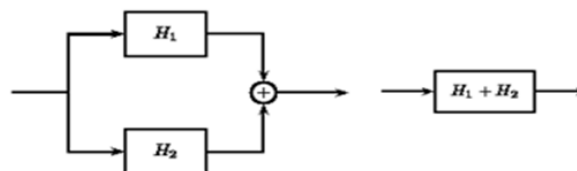
### Sistema LTI en serie

Si dos o más sistemas están en **serie** uno con otro, el **orden puede ser intercambiado** sin que se vea afectada la salida del sistema. Los sistemas en series también son llamados como sistemas en cascada.



### Sistema LTI en paralelo

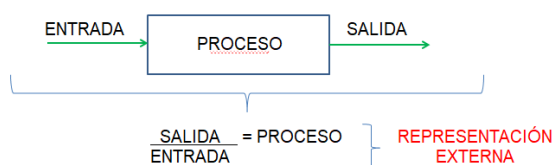
Si dos o más sistemas LTI están en **paralelo** con otro, un sistema equivalente es aquél que está definido como la **suma** de estos sistemas individuales.



## 3. Modelo de sistema en el espacio de estado

En el análisis y diseño de sistemas de control, se puede abordar la representación de los sistemas desde dos perspectivas: la **representación externa** y la **representación interna**.

La **representación externa** de un sistema se centra en la **relación directa entre las señales de entrada y salida**. Este enfoque se utiliza para comprender cómo las entradas (o estímulos) afectan las salidas (o respuestas) del sistema. La **función de transferencia** es una herramienta comúnmente empleada en esta representación, la cual proporciona una descripción matemática del sistema en el dominio de la frecuencia. La función de transferencia es particularmente útil para el análisis de la estabilidad y la respuesta en frecuencia de los sistemas lineales e invariantes en el tiempo (LTI).



La **representación interna**, por otro lado, se enfoca en la **dinámica interna del sistema** mediante el uso de **variables de estado**. Estas variables proporcionan una descripción completa del estado del sistema en cualquier momento y permiten predecir su comportamiento futuro basado en el estado actual y las entradas presentes. El modelo en espacio de estados es la herramienta principal en esta representación, expresado mediante un conjunto de **ecuaciones diferenciales de primer orden**. El modelo en espacio de estados es especialmente potente para sistemas complejos y multivariables, ya que permite una descripción detallada de la dinámica interna del sistema.

**La representación interna (representación por variables de estado) relacionará matemáticamente las salidas con las entradas a través de las variables de estado como paso intermedio.**

**Es un método que trabaja en el dominio del tiempo.**

El **estado** de un sistema dinámico se define como la **menor colección de variables** reales cuyo valor en un determinado instante, resume el pasado dinámico del sistema y es suficiente para predecir su evolución futura.

La descripción interna establece entre las señales de entrada y salida una relación indirecta, la cual se formaliza a través de las denominadas **ecuaciones de estado** y de salida que escribimos, para el caso de variable continua:

$$\left\{ \begin{array}{l} \mathbf{x}'(t) = \Phi(\mathbf{x}(t), u(t)) \\ y(t) = \psi(\Phi(\mathbf{x}(t), u(t))) \end{array} \right.$$

$\mathbf{x}$  es el **vector de estado**, es decir, la mínima colección de  $n$  variables de estado necesarias para describir completamente el sistema, dispuestas en un vector.

$\Phi$  la **función de transición de estados** que permite determinar la evolución del vector de estado en un determinado instante a partir del conocimiento del estado actual y de las entradas de control aplicadas  $u$ .

$\psi$  la **función de lectura o salida** que suministra el valor de las salidas actuales y en base a esta misma información.

En general la dimensión de los vectores  $u$  e  $y$  puede ser cualquiera. Si en particular ambos se reducen a un escalar ( $p = m = 1$ ) el sistema se denomina **SISSO** (single-input single-output). En el caso que ambas dimensiones fuesen mayores a la unidad, el sistema se denomina **MIMO** (multiple-input multiple-output).

**Las ecuaciones de  $\mathbf{x}'(t)$  se llaman ecuaciones de estado.**  
**Las ecuaciones de  $y(t)$  se llaman ecuaciones de salida.**



El conocimiento del vector de estado en un determinado instante  $t_0$  junto con las entradas aplicadas al sistema a partir de ese momento, permite conocer el estado y la salida del sistema en cualquier instante posterior  $t$ . El espacio de estados de un proceso se define como el conjunto que contiene a todos los posibles valores del vector de estado, el cual es un espacio vectorial  $n$  dimensional.

En los sistemas dinámicos lineales, las funciones de transición y de lectura son aplicaciones lineales, donde  $x$  es un vector de  $n$  variables de estado,  $u$  un vector de  $p$  entradas, y un vector de  $m$  salidas, **A** la **matriz dinámica** de dimensión  $n \times n$ , **B** la **matriz de mando o de control** de dimensión  $n \times p$ , **C** la **matriz de salida o lectura** de dimensión  $m \times n$ , y **D** la **matriz de influencia directa de la entrada sobre la salida**, de dimensión  $m \times p$  que suele ser nula en la mayoría de aplicaciones.

Si además, el sistema es invariante en el tiempo, las matrices A, B, C y D serán constantes, por lo que el sistema:

$$\begin{cases} \mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \end{cases}$$

Admite la siguiente **representación matricial**

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}; \mathbf{x}'(t) = \begin{bmatrix} x_1'(t) \\ x_2'(t) \\ \vdots \\ x_n'(t) \end{bmatrix}; \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}; \mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

$$\begin{bmatrix} x_1' \\ x_2' \\ \vdots \\ x_n' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} u(t)$$

Esta representación matricial puede escribirse como un **conjunto de ecuaciones diferenciales de primer orden**

$$\begin{cases} x_1' = a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n + b_1u(t) \\ x_2' = a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n + b_2u(t) \\ \vdots \\ x_n' = a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n + b_nu(t) \end{cases}$$

También **una ecuación diferencial de orden n**, lineal, con coeficientes constantes, puede llevarse a la **forma matricial**.

A partir de la ecuación diferencial de orden n:

$$x^{(n)}(t) + a_1x^{(n-1)}(t) + a_2x^{(n-2)}(t) + \dots + a_{n-2}x''(t) + a_{n-1}x'(t) + a_nx(t) = f(t)$$

Se define:

$$\left\{ \begin{array}{l} x(t) = x_1(t) \\ x'(t) = x_2(t) = x_1'(t) \\ x''(t) = x_3(t) = x_2'(t) \\ \vdots \\ x^{(n-1)}(t) = x_n(t) = x_{n-1}'(t) \\ x_n'(t) = f(t) - a_1 x_n(t) - a_2 x_{n-1}(t) - \dots - a_{n-1} x_2(t) - a_n x_1(t) \end{array} \right.$$

La representación matricial de este sistema es:

$$\begin{bmatrix} x_1' \\ x_2' \\ \vdots \\ x_n' \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & & & \ddots & \\ 0 & 0 & 0 & & 1 \\ -a_n & -a_{n-1} & \dots & & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} u(t)$$

Para hacer un diagrama de esta situación, consideremos:

$$x_2(t) = x_1'(t)$$

es decir,  $x_2(t)$  es la derivada de  $x_1(t)$ . En sentido inverso, es posible obtener  $x_1(t)$  pues

$$x_1(t) = \int x_2(t) dt$$

Esto significa que para recuperar  $x(t)$  a partir del conocimiento de su derivada primera, se debe integrar una vez.

Del mismo modo,

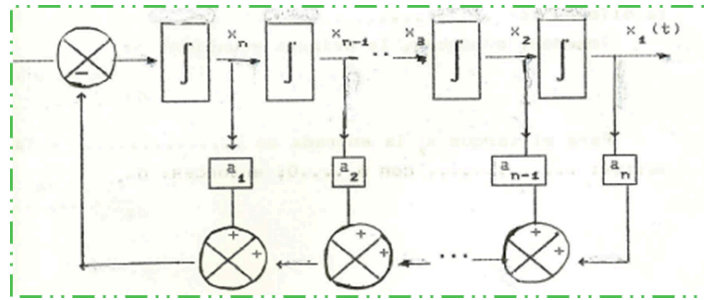
$$x_3(t) = x_2'(t) = x_1''(t)$$

Si se desea recuperar  $x_1(t)$ , se debe integrar dos veces.

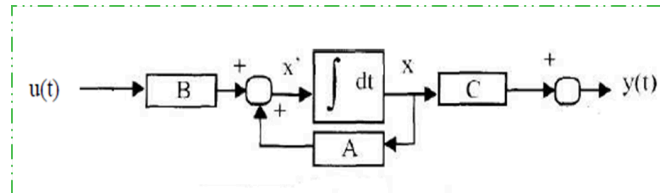
Generalizando, dada  $x^{(n)}$ , para recuperar  $x(t)$  se debe integrar  $n$  veces.

Si consideramos la ecuación homogénea, es decir,  $f(t)=0$

$$\begin{aligned} x^{(n)}(t) + a_1 x^{(n-1)}(t) + a_2 x^{(n-2)}(t) + \dots + a_{n-1} x'(t) + a_n x(t) &= 0 \\ x_n'(t) &= -[a_1 x_n(t) + a_2 x_{n-1}(t) + \dots + a_{n-1} x_2(t) + a_n x_1(t)] \end{aligned}$$



En forma más general:

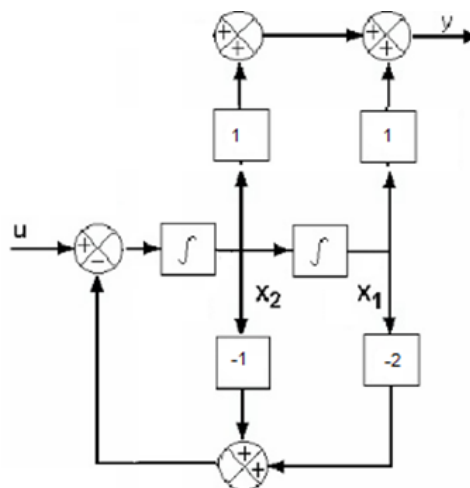


La teoría de control moderna se basa en la descripción del modelo del sistema por medio de  $n$  ecuaciones diferenciales de primer orden que pueden combinarse en una ecuación diferencial vectorial-matricial de primer orden.

La complejidad de estas ecuaciones no aumenta de forma cualitativa con el incremento del número de variables de estado, de entradas o de salidas. Además, los métodos del espacio de estado son particularmente adecuados para los cálculos en computadora debido a su representación matricial.

## Ejemplos

- Dada la siguiente representación gráfica



El sistema de ecuaciones correspondiente es:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t);$$

$$y(t) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- Ejemplo Modelo de Proyecto con Módulos en Prototipo y Producción

Una empresa de desarrollo de software cuenta con módulos que están en Prototipo, es decir, modelos en desarrollo evolutivo. Mediante el Desarrollo (incluye programación, implementación y prueba) los

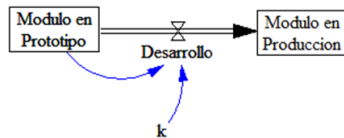
mismos pasan a Producción. Se estima que por semana el 13% de los módulos en Prototipo pasan a Producción. Inicialmente los 750 proyectos están en prototipo y no hay ningún módulo en Producción.

El modelo representado en software Vensim responde a:

$k=0.13$  - Units: 1/semanas

Modulo en Producción= INTEG (Desarrollo,0) - Units: módulos

Modulo en Prototipo= INTEG (-Desarrollo,750) - Units: módulos



Planteamos el modelo como un sistema de ecuaciones diferenciales de primer orden:

$x_1(t)$ : Módulos en Prototipo

$x_2(t)$ : Módulos en Producción

$$\begin{aligned} x_1'(t) &= -0,13x_1(t) \\ x_2'(t) &= 0,13x_1(t) \end{aligned} \quad ; \quad \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} 750 \\ 0 \end{bmatrix}$$

$$x'(t) = \begin{bmatrix} -0,13 & 0 \\ 0,13 & 0 \end{bmatrix} x(t)$$

$$x(t) = L^{-1}[(SI - A)^{-1}]x_0$$

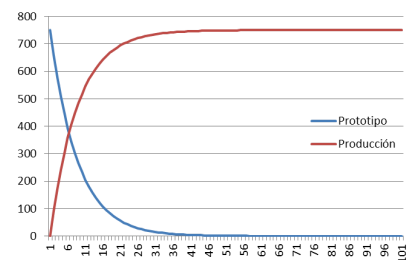
$$SI - A = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} - \begin{bmatrix} -0,13 & 0 \\ 0,13 & 0 \end{bmatrix} = \begin{bmatrix} s + 0,13 & 0 \\ -0,13 & s \end{bmatrix}$$

$$(SI - A)^{-1} = \frac{\begin{bmatrix} s & 0 \\ 0,13 & s + 0,13 \end{bmatrix}}{s(s + 0,13)} = \begin{bmatrix} \frac{1}{(s + 0,13)} & 0 \\ \frac{0,13}{s(s + 0,13)} & \frac{1}{s} \end{bmatrix}$$

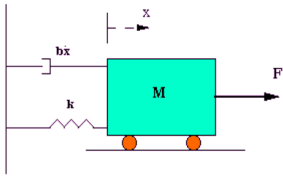
$$x(t) = L^{-1}[(SI - A)^{-1}]x_0 = L^{-1}\left\{\begin{bmatrix} \frac{1}{(s + 0,13)} & 0 \\ \frac{0,13}{s(s + 0,13)} & \frac{1}{s} \end{bmatrix} \begin{bmatrix} 750 \\ 0 \end{bmatrix}\right\} =$$

$$L^{-1}\left\{\begin{bmatrix} \frac{750}{(s + 0,13)} \\ \frac{750 \cdot 0,13}{s(s + 0,13)} \end{bmatrix}\right\} =$$

$$\begin{aligned} x_1(t) &= 750 \cdot e^{-0,13t} \\ x_2(t) &= 750(u(t) - 750 e^{-0,13t}) \end{aligned}$$



- Modelo de masa simple, resorte, y amortiguador



La ecuación diferencial de esta planta es:

$$F = Mx'' + bx' + kx$$

M: masa en (Kg)

b: constante de amortiguador

k: constante de resorte

x: desplazamiento de la masa M

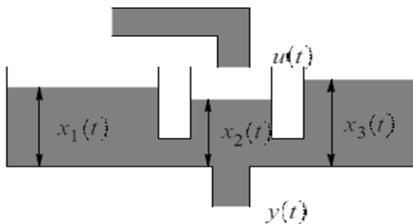
F: reacción del resorte y/o amortiguador producida por el desplazamiento de la masa

$$\begin{cases} x_1(t) = x(t) \\ x'(t) = x_2(t) = x_1'(t) \\ x''(t) = x_2'(t) = x_2'(t) \Rightarrow x_2'(t) = -\frac{b}{M}x_2 - \frac{k}{M}x_1 + \frac{F}{M} \end{cases}$$

La ecuación matricial resultante es:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{M} & -\frac{b}{M} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{F}{M} \end{bmatrix}$$

- La gráfica representa un modelo hidráulico de tanques interconectados en forma paralela



La ecuación matricial resultante es:

$$\dot{x}(t) = \begin{bmatrix} -1 & 1 & 0 \\ 1 & -3 & 1 \\ 0 & 1 & -1 \end{bmatrix} * x(t) + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} * u(t)$$

$$y(t) = [0 \quad 1 \quad 0] * x(t)$$

- Modelo de anticuerpos y bacterias

Ante la presencia de bacterias, el cuerpo fabrica anticuerpos que la destruyen. Los anticuerpos se fabrican a una tasa proporcional al número de anticuerpos y las bacterias se reproducen a una velocidad proporcional a la cantidad de bacterias que hay en ese momento. Las ecuaciones del problema son:

$x_1(t)$  = Cantidad de ANTICUERPOS en el instante t

$x_2(t)$  = Cantidad de BACTERIAS en el instante t

$$\begin{cases} \dot{x}_1(t) = ax_1(t) \\ \dot{x}_2(t) = -ax_1(t) + ax_2(t) \end{cases}$$

$$x'(t) = \begin{bmatrix} a & 0 \\ -a & a \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

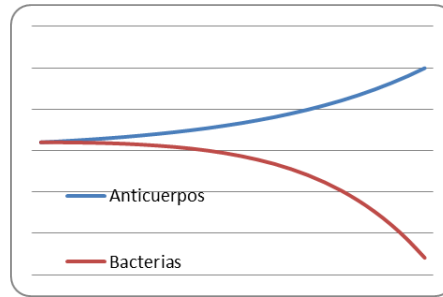
Proponemos  $a=0.1$  y  $x(0)=10$

$$A = \begin{bmatrix} 0.1 & 0 \\ -0.1 & 0.1 \end{bmatrix}$$

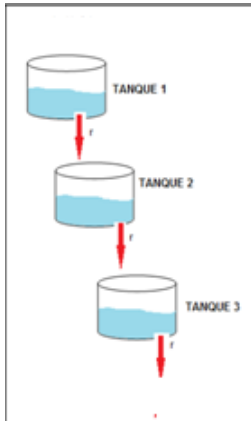
Se aplica el método de la Transformada inversa de Laplace para obtener la solución del sistema:

$$(sI - A)^{-1} = \left[ \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} - \begin{bmatrix} 0.1 & 0 \\ -0.1 & 0.1 \end{bmatrix} \right]^{-1} = \frac{\begin{bmatrix} (s-0.1) & 0 \\ 0.1 & (s-0.1) \end{bmatrix}}{(s-0.1)^2} = \begin{bmatrix} \frac{1}{(s-0.1)} & 0 \\ \frac{-0.1}{(s-0.1)^2} & \frac{1}{(s-0.1)} \end{bmatrix}$$

$$\begin{cases} x_1'(t) = e^{0.1t} \cdot 10 \\ x_2'(t) = (-0.17e^{0.1t} + e^{0.1t}) \cdot 10 \end{cases}$$



- Modelo de tres tanques



En la siguiente figura se presentan tres tanques de salmuera que contienen  $V_1$ ,  $V_2$  y  $V_3$  galones de salmuera (1 galón = 3,785 litros). En el tanque 1 fluye agua dulce, en tanto que fluye salmuera del tanque 1 al 2, del tanque 2 al 3 y fuera del tanque 3.

Sea  $x_i(t)$  la cantidad (en libras) de sal en el tanque  $i$  al tiempo  $t$  ( $i=1,2,3$ ). Si cada una de las velocidades de flujo son  $r$  galones por minuto, determinar las cantidades de sal al tiempo  $t$  en los tanques de salmuera.

$k_i = r / V_i$ . Condiciones Iniciales:  $x_1(0) = 15$ ,  $x_2(0) = x_3(0) = 0$

$V_1=20$ ,  $V_2=40$ ,  $V_3=50$ ;  $r=10$  (gal/min).

Encontrar la cantidad de sal en cada tanque en el instante  $t \geq 0$ .

Cada tanque estará representado por una variable de estado. El modelo se describirá por medio de las variables  $x_1$ ,  $x_2$  y  $x_3$  que representa la cantidad de sal en tanque 1, 2 y 3 respectivamente en el instante  $t$ .

Utilizamos la ecuación diferencial más simple para representar matemáticamente este problema, que es la primera derivada de una variable. Es de gran importancia para modelar sistemas continuos, ya que representan la manera en que muchos factores crecen o decaen en función de una tasa determinada.

Las cantidades iniciales de sal en los tres tanques de salmuera en libra son:  $x_1(0) = 15$ ,  $x_2(0) = x_3(0) = 0$

El sistema de ecuaciones es:

$$\begin{cases} x_1' = -k_1 x_1 \\ x_2' = k_1 x_1 - k_2 x_2 \\ x_3' = k_2 x_2 - k_3 x_3 \end{cases}$$

$$k_i = \frac{r}{V_i}; i=1, 2, 3$$

Si  $V_1=20$ ,  $V_2=40$ ,  $V_3=50$ ;  $r=10$  (gal/min) entonces:

$$k_1 = \frac{10}{20} = 0.5; \quad k_2 = \frac{10}{40} = 0.25; \quad k_3 = \frac{10}{50} = 0.2$$

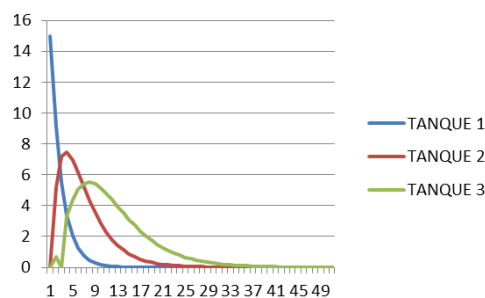
Sustituyendo los valores en las ecuaciones se obtiene:

$$\begin{cases} \dot{x}_1 = -0.5x_1 \\ \dot{x}_2 = 0.5x_1 - 0.25x_2 \\ \dot{x}_3 = 0.25x_2 - 0.2x_3 \end{cases}$$

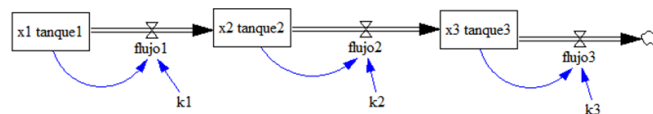
La solución de un sistema de la forma  $\dot{x}(t)=Ax(t)$ , tal como el que se ha desarrollado, se puede resolver por medio de la Transformada de Laplace.

$$\begin{cases} x_1(t) = 15e^{-0.5t} \\ x_2(t) = -30e^{-0.5t} + 30e^{-0.25t} \\ x_3(t) = 25e^{-0.5t} - 150e^{-0.25t} + 125e^{-0.2t} \end{cases}$$

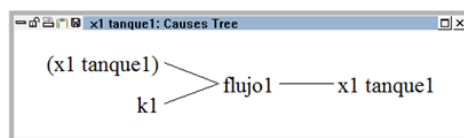
La solución indica la cantidad de sal en cada tanque para  $t>0$ . La gráfica de la evolución de cada tanque es:



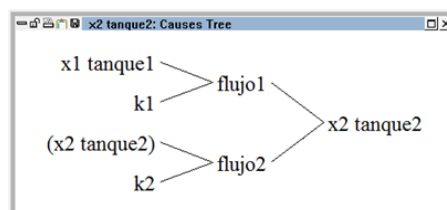
La representación de este modelo por medio del Diagrama de Simulación es:



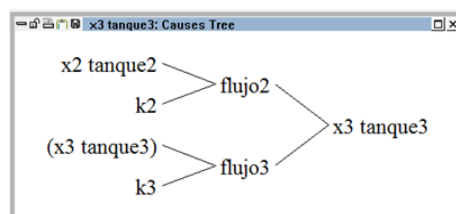
El tanque 1 es influenciado por el flujo 1, cuya ecuación queda representada por  $\dot{x}_1 = -k_1x_1$



El tanque 2 es influenciado por el flujo 1 y 2, cuya ecuación queda representada por  $\dot{x}_2 = k_1x_1 - k_2x_2$



El tanque 3 es influenciado por el flujo 2 y 3, cuya ecuación queda representada por  $\dot{x}_3 = k_2x_2 - k_3x_3$



## 4. Ejercitación

1. Tache la opción que no corresponda en cada enunciado y complete según se solicite para obtener proposiciones verdaderas.

- a. Sea  $y(n)=x(-n)$

Al evaluar la función para  $n=2$  y  $n=-2$ , se puede afirmar que el sistema es **causal/no causal**.

- b. Sea

$$y(t) = x^2(t - 2); t \geq 2$$

$y(2)$  depende de  $x(\underline{\quad})$ , esto implica que el sistema es **causal/no causal**. En cambio, si

$$y(t) = x^2(t + 2)$$

$y(2)$  depende de  $x(\underline{\quad})$ , esto implica que el sistema es **causal/no causal**.

- c. Los sistemas físicos de tiempo real son **causales/no causales** dado que el tiempo sólo se desplaza hacia adelante.

- d.

$$y(n) = \sum_{k=0}^{n-1} x(k)$$

La salida depende del valor de la entrada en                     , por lo que el sistema **con/sin memoria**.

- e.

$$T[x(t)] = 1/4 x(t) \Rightarrow T^{-1}[x(t)]$$

La función inversa de  $T$  es                     , al introducir la señal de salida del original nos devuelva la señal de entrada, es decir el sistema es                     .

- f.

$$y(n) = |x(n)|^2$$

Al introducir la señal de salida del original es **posible/no es posible** obtener la señal de entrada, es decir el sistema es                     .



g. Los sistemas \_\_\_\_\_ son aquellos cuyas señales son continuas en el tiempo y pueden describirse mediante ecuaciones **diferenciales/en diferencias**.

Los sistemas \_\_\_\_\_ son aquellos en los cuales una o más variables pueden cambiar sólo en valores discreto de tiempo y pueden describirse mediante ecuaciones \_\_\_\_\_

h. Sea  $y(n) = n \cdot x(n)$ , es un sistema **lineal/no lineal, variante/invariante en el tiempo, causal/no causal, estable/inestable, inversible, no inversible**.

i.  $y(t) = |u(t)|$

¿Es una función lineal?

2. Elija una opción y justifique. Sea:

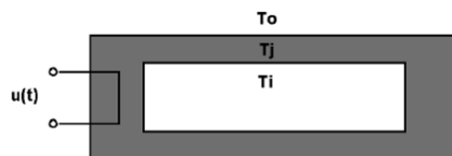
$$y(n) = 5nx^2(n)$$

El sistema es:

a) Lineal/ Causal /Invariante.

b) No Lineal/ Causal / Variante.

3. De acuerdo al siguiente enunciado, realice el modelo correspondiente.



La figura muestra un Sistema eléctrico de doble recinto -sistema térmico-. El sistema debe mantener y ajustar las temperaturas de los recintos acotándolas en ciertos valores, teniendo en cuenta algunos parámetros físicos de los recintos.

Estos recintos contienen fluidos que no afectan al modelo propuesto y sus propiedades físicas y químicas no varían y tampoco hay entrada o salida de fluido dentro de los recintos en el período de ensayo.

El sistema modelado debe mostrar la interacción de los recintos con la entrada (resistencia eléctrica), que solo afecta al recinto exterior (según figura).

Como se ve en el modelo, los dos recintos pierden temperatura, pero parte de la energía es absorbida por el otro recinto. Como se mencionó, el recinto exterior ( $x_2$ ) recibe la energía de la entrada, pero del mismo modo éste recibe la energía calórica que aporta el ambiente exterior.

Considerando constante la temperatura ambiente, se compensa este término con el valor  $-0,25T_{amb}$ .

Se tomará como salida la temperatura del recinto interior .

Se toman en cuenta los siguientes parámetros:

- $u(t)$  es la cantidad de calor de entrada al recinto .(calentador eléctrico).
- $T_0, T_j, T_i$  son las temperaturas en el exterior del horno (ambiente), en el primer recinto y en el recinto interior respectivamente.
- $C_i, C_j$  son las capacidades caloríficas del espacio interior y del exterior respectivamente.
- $h_i, h_j$  son los coeficientes de transmisión de calor para la superficie interior y exterior respectivamente.

Condiciones iniciales:

- $u(t)=0,5$
- $T_0=-0,25T_{amb}$ .
- $C_i=2; C_j=0.75$
- $h_i=0.5; h_j=2$

Obtenga:

a) Descripción del modelo mediante Ecuaciones de Estado. Aplicar condiciones iniciales y obtener el nuevo modelo.

b) Representación matricial del sistema.

**4.** Tomamos el ejemplo del modelo de teoría de masa - resorte y codificamos con MatLab para obtener:

4.1. Forma matricial,

4.2. Solución de sistema.

4.3. Gráficas de la solución del sistema

**5.** Ejemplo del Planteo de un modelo matemático para sistema continuo. (Homework)

## **A: INVESTIGACIÓN Y ANÁLISIS DEL MODELO**

*El proyecto propuesto a desarrollar es la simulación del control de temperatura de una **Placa de Video**. A continuación se definirá el término tarjeta gráfica y su función.*

*Una tarjeta gráfica, tarjeta de vídeo, placa de vídeo, tarjeta aceleradora de gráficos o adaptador de pantalla, es una tarjeta de expansión para una computadora u ordenador, encargada de procesar los datos provenientes de la CPU y transformarlos en información comprensible y representable en un dispositivo de salida, como un monitor o televisor.*

*Es habitual que se utilice el mismo término tanto a las habituales tarjetas dedicadas y separadas como a las GPU integradas en la placa base. Algunas tarjetas gráficas han ofrecido funcionalidades añadidas como captura de vídeo, sintonización de TV, etc.*

## 2-PLANTEO DEL MODELO

### B: Objetivo del modelo

*El objetivo del trabajo es el diseño de un sistema de control de temperatura aplicado a una placa de video.*

### C: Variables y parámetros

#### - Parámetros:

$K_V$ : Constante de ventilación. Equivale a  $-0,5$

$K_M$ : Constante del material del cual está hecho el chip GPU. Equivale a 1

$K_{TC}$ : Constante de temperatura del chip GPU. Equivale a  $\frac{0,02}{^{\circ}\text{C}}$

$K_{TV}$ : Constante de temperatura del ventilador. Equivale a  $\frac{0,007 \text{ RPM}}{^{\circ}\text{C} \times \text{MHZ}}$

$K_{CT}$ : Constante de control de temperatura de ventilación. Equivale a  $\frac{0,35 \text{ RPM}}{^{\circ}\text{C}}$

$K_{CF}$ : Constante de control de procesamiento del chip GPU. Equivale a  $\frac{0,15 \text{ MHZ}}{^{\circ}\text{C}}$

#### - Variables internas

$T$ : Indica la temperatura capturada del sensor instalado en la placa. [ $^{\circ}\text{C}$ ]

#### - Variables de estado

$X_1(t)$ : Indica la velocidad de procesamiento a la cual funcionara el chip GPU [MHZ]

$X_2(t)$ : Indica la velocidad a la cual está funcionando el ventilador. [RPM]

#### - Salida del sistema

$Y(t)$  = velocidad del chip GPU y velocidad del ventilador

### D: Establecer la estructura del sistema.

- La variable que se realimentara sea  $X_2$ .
- La variable que manipularemos será  $\mu_1(t)$  y  $\mu_2(t)$ .
- Como perturbación en este sistema encontramos la temperatura de ambiente. Si está excesivamente alta podría poner en peligro la placa.
- La situación de peligro a la cual estaría expuesta la placa seria de alcanzar una temperatura igual o mayor a  $130^{\circ}$ .

### E: Ecuaciones del sistema

$$X_1'(t) = (K_M - T * K_{TC}) * X_1(t) + K_{CF} * T * \mu_1(t)$$

$$X_2'(t) = (T * K_{TV}) * X_1(t) + K_V * X_2(t) + K_{CT} * T * \mu_2(t)$$

$$Y(t) = X_1(t) + X_2(t)$$

Remplazando con las constantes:

$$X'_1(t) = (1 - T * 0.02) * X_1(t) + (0.15) * T * \mu_1(t)$$

$$X'_2(t) = (T * 0.007) * X_1 - 0.5 * X_2(t) + (0.35) * T * \mu_2(t)$$

Forma matricial:

$$\begin{bmatrix} X'_1(t) \\ X'_2(t) \end{bmatrix} = \begin{bmatrix} 1 - T \times 0.02 & 0 \\ T \times 0.007 & -0.5 \end{bmatrix} \times \begin{bmatrix} X_1(t) \\ X_2(t) \end{bmatrix} + \begin{bmatrix} (0.15) \times T \\ (0.35) \times T \end{bmatrix} \times \mu(t)$$

## SECCIÓN PRÁCTICA CON SOFTWARE

# 5. Introducción a la aplicación MatLab

## i. ¿Qué es Matlab? Características Principales

(Software Matemático Con Entorno Integrado)

El software MatLab se desarrolló como un “ Laboratorio de matrices”, pues su elemento básico es una matriz. Es un sistema interactivo y un lenguaje de programación de cómputos científico y técnico en general. MATLAB es el nombre abreviado de “MATrix LABoratory”. Es un programa para realizar cálculos numéricos con vectores y matrices. Como caso particular puede también trabajar con números escalares, tanto reales como complejos.

MATLAB es un lenguaje de computación técnica de alto nivel y un entorno interactivo para desarrollo de algoritmos, visualización de datos, análisis de datos y cálculo numérico. Con MATLAB, podrá resolver problemas de cálculo técnico más rápidamente que con lenguajes de programación tradicionales, tales como C, C++ y FORTRAN. Puede usarlo en una amplia gama de aplicaciones que incluyen procesamiento de señales e imágenes, comunicaciones, diseño de sistemas de control, sistemas de prueba y medición, modelado y análisis financiero y biología computacional. Los conjuntos de herramientas complementarios (colecciones de funciones para propósitos especiales, que están disponibles por separado) amplían el entorno de MATLAB permitiendo resolver problemas especiales en estas áreas de aplicación. Además, contiene una serie de funciones para documentar y compartir su trabajo. Puede integrar su código de MATLAB con otros lenguajes y aplicaciones, y distribuir los algoritmos y aplicaciones que desarrollo usando MATLAB.

### Características Principales

1. Lenguaje de alto nivel para cálculo técnico
2. Entorno de desarrollo para la gestión de código, archivos y datos

3. Herramientas interactivas para exploración, diseño y resolución de problemas iterativos
4. Funciones matemáticas para álgebra lineal, estadística, análisis de Fourier, filtraje, optimización e integración numérica
5. Funciones gráficas bidimensionales y tridimensionales para visualización de datos
6. Herramientas para crear interfaces gráficas de usuario personalizadas
7. Funciones para integrar los algoritmos basados en MATLAB con aplicaciones y lenguajes externos, tales como C/C++, FORTRAN, Java, COM y Microsoft Excel.

## **ii. Respecto de la Instalación y sus Directorios**

Las nuevas versiones son fáciles de instalar. Pero es importante tener en cuenta que el núcleo fundamental de matlab se encuentra en los subdirectorios BIN y MATLAB. En BIN se encuentran los programas ejecutables. Es necesario comentar que matlab cuenta con dos tipos básicos de funciones:

- Funciones denominadas built-in functions: Son funciones que matlab tiene incorporadas internamente y por tanto no son accesibles al usuario.
- Funciones llamadas m functions: Son funciones cuyo código es accesible. Las que se encuentran en el subdirectorio MATLAB son las básicas para el funcionamiento del sistema.

Como se desprende del árbol de directorios, los toolboxes se suelen instalar en forma de subdirectorios en el disco duro, colgando del subdirectorio TOOLBOX. En ellos se encuentran también funciones .m orientadas al control de sistemas. Además, se pueden incorporar otros toolboxes (signal processing, image processing, robust control, non-linear control, system identification, etc), e incluso funciones propias del usuario.

matlab\general - Comandos de propósito general

matlab\ops - Operadores y caracteres especiales

matlab\lang - Constructores del lenguaje de programación

matlab\elmat - Matrices elementales y manipulación matricial

matlab\elfun - Funciones matemáticas elementales

matlab\specfun - Funciones matemáticas especiales

matlab\matfun - Funciones matriciales - álgebra lineal numérica

matlab\datafun - Análisis de datos y transformada de Fourier

matlab\polyfun - Interpolación y polinomios

matlab\funfun - Funciones de funciones y métodos para ODE

matlab\sparfun - Funciones para matrices dispersas

matlab\graph2d - Gráficos en dos dimensiones

matlab\graph3d - Gráficos en tres dimensiones

matlab\specgraph - Gráficos especializados

matlab\graphics - Manipulación de gráficos

matlab\uitools - Herramientas de interfaz gráfica de usuario (GUI)

matlab\strfun - Cadenas de caracteres

matlab\iofun - Funciones para entrada/salida de ficheros

matlab\timefun - Hora y fecha

matlab\datatypes - Tipos de datos y estructuras

matlab\winfun - Ficheros de interfaz con Windows (DDE/ActiveX)

matlab\demos - Ejemplos y demostraciones

simulink\simulink - Simulink

simulink\blocks - Librería de bloques de Simulink

simulink\simdemos - Ejemplos y demostraciones de Simulink

toolbox\control - Paquete de Control de Sistemas

### iii. Misceláneas De Matlab Respecto Del Entorno De Trabajo

1. Con el comando `path` puede comprobarse cuáles son las localizaciones de los ficheros y programas con los que va a trabajar matlab, pudiendo añadirse nuevos subdirectorios (incluso personales) a conveniencia. La forma más cómoda de interactuar con dichas localizaciones es mediante la opción `File/Set-Path...` en el menú de la ventana de comandos.
2. Para poder usar cualquier función `.m`, como por ejemplo las contenidas en el paquete de control, bastará con que el camino `\matlabxx\toolbox\control` esté incluido en el `path` de matlab (cosa que ocurrirá si el paquete se instaló adecuadamente).
3. Por otro lado, matlab comienza trabajando, por defecto, en el subdirectorio `matlabxx\work`. Si queremos cambiar de directorio de trabajo en cualquier momento, podemos hacerlo con el comando `cd camino`. Puede utilizarse el nombre completo del comando si se desea: `chdir`. Cabe decir que todas las funciones `.m` que existan en el directorio de trabajo serán localizadas sin necesidad de tener que incluir dicho directorio en el `path` de matlab.
4. El comando `pwd` nos indica cuál es el directorio de trabajo actual.
5. Para mostrar el contenido del directorio de trabajo, se pueden emplear los comandos **`dir`** ó **`ls`**. El comando **`delete nombre-fichero`** puede emplearse para eliminar un archivo del directorio de trabajo. Asimismo, se pueden realizar operaciones típicas de línea de comandos del sistema operativo DOS, introduciendo el comando correspondiente precedido por el símbolo `"!"`.
6. Resulta interesante tener en cuenta que la línea de comandos de matlab posee "memoria" y podemos recuperar comandos introducidos previamente, haciendo uso de las teclas de movimiento de cursor arriba y abajo. Para una localización más eficaz de algún comando introducido previamente, podemos teclear los primeros caracteres del mismo antes de usar el cursor arriba y sólo buscará entre los comandos ya introducidos aquéllos cuyos primeros caracteres coincidan con los introducidos.
7. Otra posibilidad que se ofrece es la de introducir varios comandos en una misma línea de la ventana de comandos, separados por coma o punto y coma.
8. Puede "limpiarse" el contenido de la ventana de comandos mediante la instrucción **`clc`**. Pero si se quiere limpiar el contenido de la memoria completo incluso el de las variables creadas hasta el momento se debe utilizar **`clear all`**.
9. El símbolo **`%`** sirve para introducir comentarios. Todo lo escrito desde ese símbolo hasta el final de la línea será ignorado por el intérprete de matlab.
10. Si se quiere guardar toda la sesión en un archivo, basta usar el comando **`diary nombrearchivo`**. Dicho archivo contendrá los comandos introducidos y los correspondientes resultados. Cuando no se quiera seguir almacenando la información se introducirá **`diary off`**.

11. Si se desean almacenar todas las variables de memoria en un fichero, junto con sus valores actuales, se usa el comando **save nombre-fichero**. Esto crea un fichero binario en el directorio de trabajo actual con el nombre introducido y con extensión .mat. Si no se da el nombre del fichero, se crea uno llamado matlab.mat. En caso que se desee guardar en un fichero con formato ascii, se introducirá en el comando un modificador: **save -ascii nombre fichero**. Si sólo se quieren guardar una serie de variables, se **introducirá save nombre-fichero nombre-variables** separadas por espacios.
12. Para recuperar los ficheros generados con el comando **save** se utilizará **load nombre-fichero**.
13. La salida del sistema se efectúa al introducir **quit** ó **exit**, o simplemente cerrando la ventana de comandos.
14. Los elementos de cada fila de una matriz se pueden introducir separados por espacios o por comas, indistintamente.
15. Para separar filas de una matriz se usa “;” o un simple retorno de carro.
16. Los elementos de vectores y matrices pueden ser reales, complejos e incluso expresiones, como vemos en el caso del último elemento del vector C.
17. Si se está introduciendo un comando o conjunto de ellos cuya sintaxis sea muy larga, se puede continuar en la siguiente línea introduciendo al final de la actual tres puntos seguidos (...).
18. Las variables a las que se asignan resultados, así como las variables de entorno, se almacenan en lo que se denomina el espacio de trabajo de matlab (**workspace**).
19. Además de variables numéricas, escalares o matriciales, en matlab pueden usarse cadenas de caracteres. Para ello se delimita una secuencia de caracteres mediante apóstrofes: cadena = ‘ejemplo de cadena de caracteres’.
20. El comando help, muestra la ayuda de MatLab. Se puede utilizar help (solo) o help “lo que se desea averiguar”.

## iv. Entorno De Trabajo De Matlab

Los componentes más importantes del entorno de trabajo de MATLAB son el *editor de caminos de búsqueda* (**Path Browser**), el *editor y depurador de errores* (**Editor & Debugger**) y el *visualizador del ambiente de trabajo* (**Workspace Browser**).

Path Browser. Matlab puede llamar a una gran variedad de funciones, tanto propias como programadas por los usuarios. A veces, puede incluso haber funciones distintas que tienen el mismo nombre. Por tanto, es interesante saber cómo busca Matlab cualquier función que se le pida que ejecute. La clave es el camino de búsqueda (search path) que el programa utiliza cuando encuentra el nombre de una función. El search path es una lista de directorios que se puede ver y modificar mediante la orden path, o utilizando el Path Browser (Submenú Set Path en el menú File).

El directorio actual. El concepto de directorio actual o de trabajo es crucial en Matlab. Es el directorio donde el usuario debe guardar los diferentes archivos que genere en las sesiones, para que Matlab pueda detectarlos. El contenido de dicho directorio puede obtenerse con la orden **dir**. Para cambiar el directorio actual se utiliza la orden **cd** (Change Directory) seguido del nombre del nuevo directorio. Ejecutando **cd ..**, se sube un nivel en la jerarquía de directorios. Estos cambios también pueden hacerse de un modo gráfico.

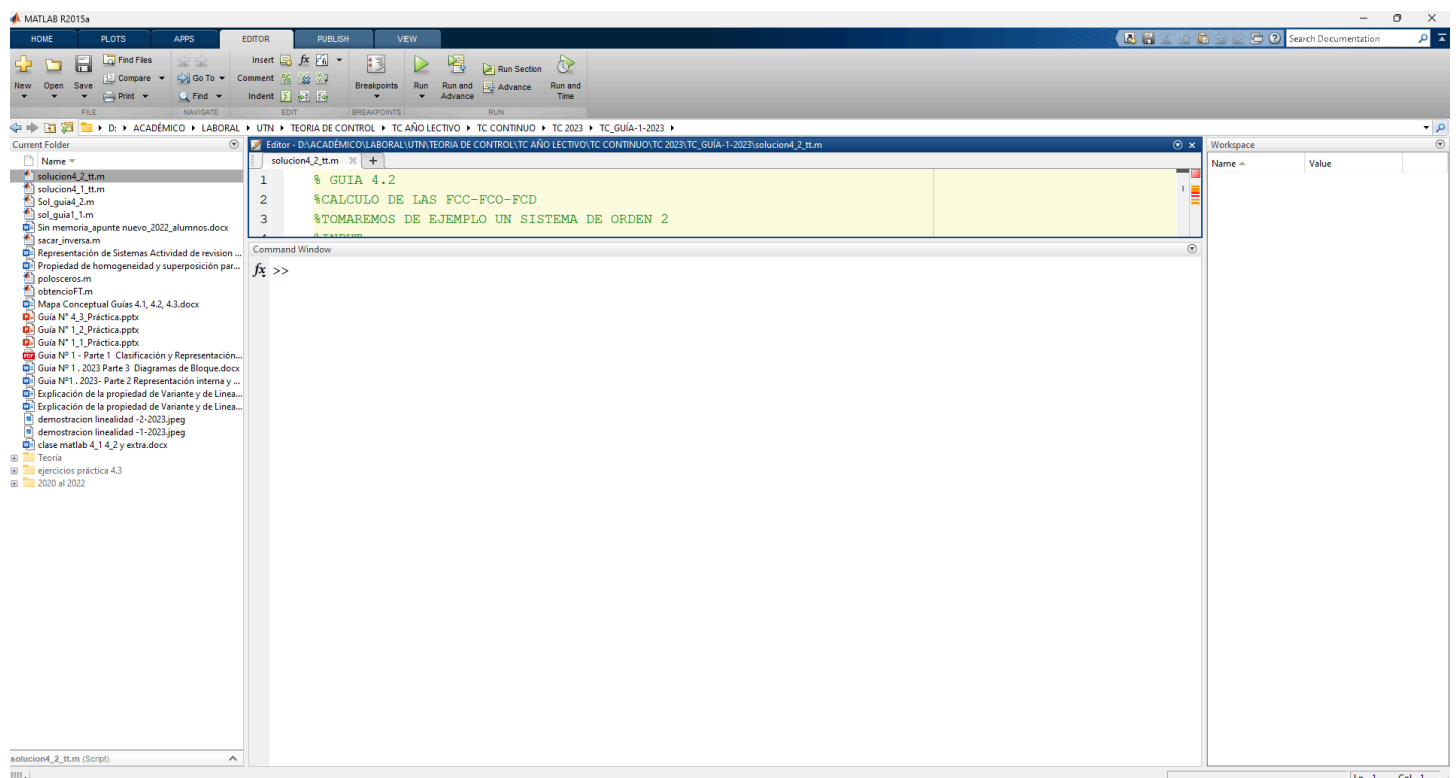


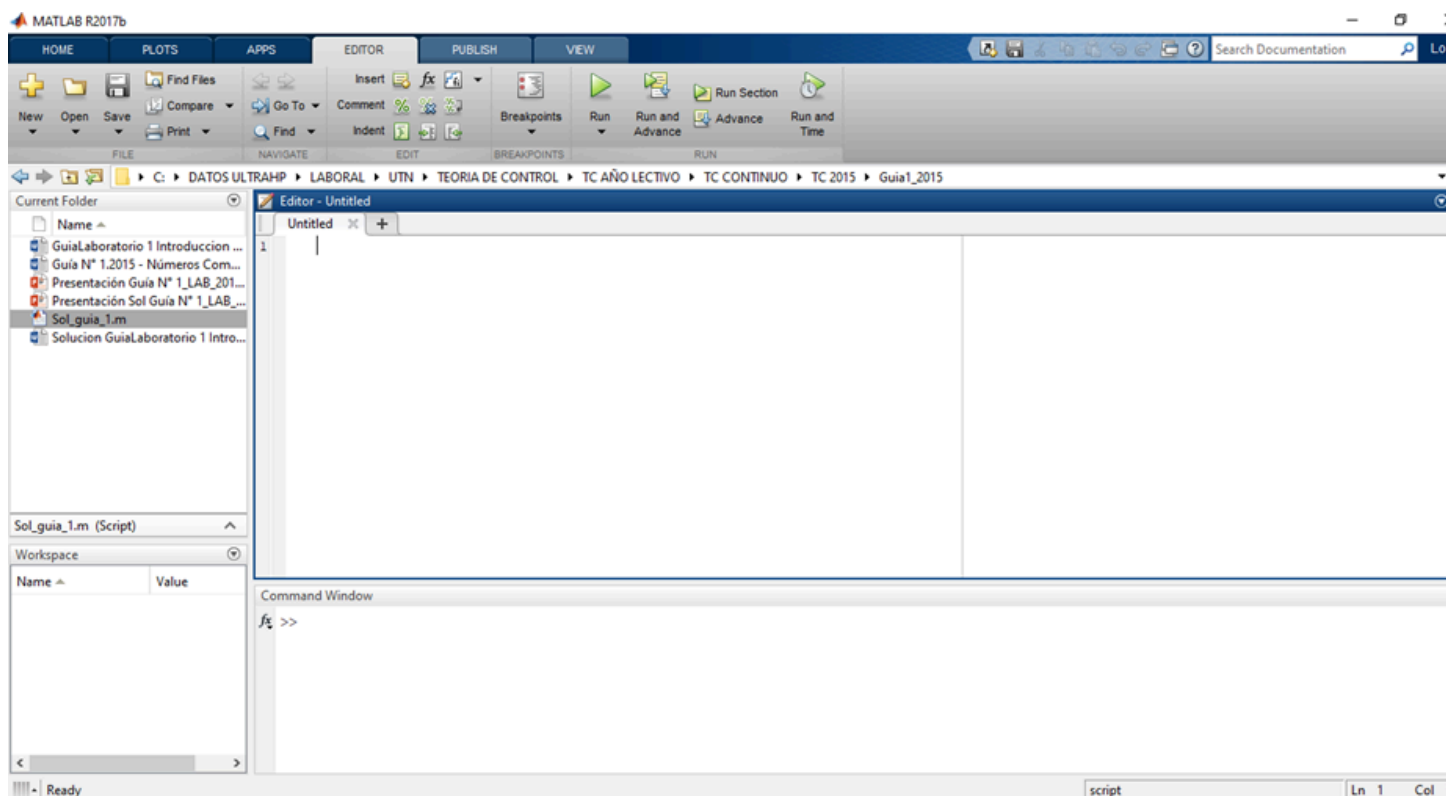
A continuación se resumen brevemente los principales elementos disponibles en esta ventana, cuyo aspecto, por defecto, se muestra en la siguiente Figura. Esta ventana puede variar de acuerdo a la versión pero en esencia es lo mismo.

Editor/Debugger. En Matlab tienen particular importancia los M-archivos, esto es, archivos con la extensión “\*.m”, los cuales son archivos de texto ASCII que contienen un cierto conjunto de órdenes de Matlab. La importancia de estos archivos es que al teclear su nombre en la línea de órdenes de Matlab, se ejecutan todas las órdenes contenidas en dicho archivo. Matlab dispone de un editor propio que permite tanto crear y modificar estos archivos (proceso de edición-Editor ), como ejecutarlos paso a paso para detectar errores (proceso de depuración-Debugger ).

Workspace Browser. El espacio de trabajo (Workspace) de Matlab es el conjunto de variables que en un determinado momento están definidas en la memoria del programa. Para obtener información sobre el workspace se pueden utilizar las órdenes who y whos. La segunda proporciona una información más detallada que la primera.

Entornos de trabajo para las versiones 2015 y 2017





## v. Operadores

Operadores:

Oper. aritméticos	
+	Suma.
-	Resta.
.*	Multiplicación.
./	División derecha.
.\	División izquierda.
:	Operador dos puntos.
.^	Potencia.
.'	Transpuesta.
'	Conjugada transpuesta.
*	Multiplicación de matrices.
/	División derecha de matrices.
\	División izquierda de matrices.
^	Potencia de matrices.

Oper. de relación	
>	Menor que
>	Mayor que
<=	Menor que o igual a
>=	Mayor que o igual a
==	Igual a
=	No igual a

Operadores lógicos.	
&	Y
	OR
~	NO

## vi. Tipos De Variables

Tipos de variables:

Clase	Ejemplo	Descripción
array	<code>[1,2;3,4]; 5+6i</code>	Datos virtual ordenado por índices cuyos componentes son datos del mismo tipo.
char	<code>'Hola'</code>	Array de caracteres (cada carácter tiene 16 bits).
celda	<code>{17, 'hola', eye(2)}</code>	Dato virtual ordenado por índices cuyos componentes son arrays de distinto tipo.
struct	<code>a.dia=1; a.mes='julio'</code>	Dato virtual para almacenar datos por campos (estructura). Cada campo es un array o celda.
objeto	<code>tf(1,[1,1])</code>	Datos definido por el usuario con base a una estructura y con funciones asociadas.

## vi. Variables especiales

NOMBRE	SIGNIFICADO	VALOR
pi	$\pi$	3.1415926 ...
i, j	Unidad imaginaria	$\sqrt{-1}$
inf	Infinito. Resultado de dividir por cero o por cálculo fuera de rango (overflow)	$\infty$
NaN	No es número, resultado de: 0.0/0.0 y inf-inf.	NaN
eps	Épsilon de la máquina	2.2204e-16

## vii. Sentencias De Control

Sentencias de control en Matlab:

- **if, else y elseif:** Ejecuta un grupo de sentencias basándose en condiciones lógicas.
- **switch, case y otherwise:** Ejecuta diferentes grupos de sentencias en función de condiciones lógicas.
- **while:** Ejecuta un número de sentencias de forma indefinida en función de una sentencia lógica.
- **for:** Ejecuta un número de sentencias un número determinado de veces.
- **try...catch:** Cambia el control de flujo en función de los posibles errores producidos.
- **break:** Termina de forma directa la realización de un bucle **for** o **while**.
- **return:** Sale de la función.

Nota: Los bucles **for** y **while** pueden ser modificados por código vectorizado para aumentar la velocidad de ejecución.

- Sentencia **break:**
  - Sirve para salir de forma automática del último bucle **while** o **for** abierto sin tener en cuenta la condición o índice de salida.
- Sentencia de control **try, catch:**
  - Formulación general,  
**try bloque-1 catch bloque-2 end**
  - Ejecuta el *bloque-1* mientras no haya un error. Si se produce un error en *bloque-1* se ejecuta *bloque-2*.
- Sentencia **return:**
  - Se sale de la función en la que se trabaja.
  - Si se llega al final de la función (**\*.m**), Matlab sale de ella automáticamente.

## viii. Principales Funciones Matemáticas

<b>FUNCIONES</b>	<b>DESCRIPCIÓN</b>	<b>EJEMPLO</b>	<b>SOLUCIÓN</b>
<b>sin, cos, tan</b>	Funciones trigonométricas	<b>sin(pi/2)</b>	<b>1</b>
<b>asin, acos, atan</b>	Funciones trigonométricas inversas	<b>acos(1)</b>	<b>0</b>
<b>sinh, cosh, tanh</b>	Funciones hiperbólicas	<b>tanh(2)</b>	<b>0.9640</b>
<b>inh, acosh, atanh</b>	Funciones hiperbólicas inversas	<b>asin(1)</b>	<b>1.5708</b>
<b>log</b>	Logaritmo natural	<b>log(2.7183)</b>	<b>1</b>
<b>log2</b>	Logaritmo en base dos	<b>log2(16)</b>	<b>4</b>
<b>log10</b>	Logaritmo en base diez	<b>log10(100)</b>	<b>2</b>
<b>exp</b>	Función exponencial	<b>exp(1)</b>	<b>2.7178</b>
<b>inv</b>	Inverso multiplicativo	<b>inv(0.2)</b>	<b>5</b>
<b>sqrt</b>	Raíz cuadrada	<b>sqrt(4)</b>	<b>2</b>
<b>abs</b>	Valor absoluto	<b>abs(-1)</b>	<b>1</b>
<b>imag, real</b>	Parte Real, Parte Imaginaria	<b>imag(i), real (1)</b>	<b>1, 1</b>

## ix. Programas

En Matlab hay dos tipos de programas: scripts y las funciones.

### Scripts.

- Es simplemente una secuencia de órdenes de Matlab.
- No tiene parámetros (argumentos) de entrada ni de salida.
- Las variables definidas en un guión son globales, es decir, después del llamado del guión, estas variables siguen existiendo.
- Tienen la extensión .m

### Funciones

- Tiene parámetros (argumentos) de entrada y de salida si se desea.
- Las variables definidas dentro de la función dejan de existir una vez finalizada la ejecución de la función.
- Se deben guardar con el mismo nombre de la función y también tienen la extensión .m.

Ejemplo: Guardar el scripts siguiente con el nombre scripts1.m y ejecutar.

```
% evalúa la función y = x^2 para valores de x entre 0 y 10 a pasos de 1.
```

```
clear all;
```

```
clc;
```

```
a = 0;b = 10;
```

```
dx = 1;
```

$x = a:dx:b;$

$y = x.^2$

RTA.

$y = 0 \ 1 \ 4 \ 9 \ 16 \ 25 \ 36 \ 49 \ 64 \ 81 \ 100$

## x. Actividad

### Matrices

- 1) Si se quiere introducir por ejemplo la matriz  $A = \begin{bmatrix} 4 & 2 \\ 3 & 3 \end{bmatrix}$ .
- 2) Ejemplifique las operaciones básicas con matrices:  
Adición (sustracción)  $A+B$  o  $A-B$   
Multiplicación  $A*B$   
Producto por un escalar  $\alpha*A$   
Cálculo de la inversa  $\text{inv}(A)$  o  $A^{(-1)}$   
Cálculo del determinante  $\det(A)$
- 3) Utilice el comando "help" y responda ¿Cómo se calcula el polinomio característico asociado a la matriz A del ejercicio 1? (Ayuda: Poly), Ejemplifique
- 4) Pruebe en matlab lo siguiente y diga que se genera en cada caso  
 $1:0.1:10$   
 $1:10$   
 $[1:0.1:10]$
- 5) Teniendo que, en la forma más directa, los elementos de una matriz se referencian mediante  $A(i, j)$ , donde  $i$  y  $j$  son los índices del elemento correspondiente. Podemos usar una secuencia para facilitar la indexación de múltiples elementos. Entonces en los ejemplos siguientes que resultados darán los siguientes ejemplos:  
 $A(1,2:3)$   
 $A(:,2)$

### Autovalores

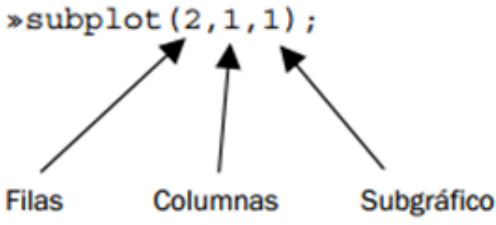
- 6) Ejemplifique los comandos utilizados para calcular los autovalores:
  - a)  $\text{roots}(p)$ : da las raíces del polinomio característico.
  - b)  $\text{eig}(A)$ : da los autovalores asociados a A.
  - c)  $\text{eigensys}(A)$ : expresa los autovalores simbólicamente.
- 7) Calcule las siguientes líneas en Matlab y resuelva. ¿Qué obtiene en cada caso?
  1.  $A = \begin{bmatrix} 4 & 2 \\ 3 & 3 \end{bmatrix}$
  2.  $[Q, D] = \text{eig}(A); Q=Q$
  3.  $[Q, D] = \text{eig}(A); D=D$
  4.  $[eves, evas] = \text{eig}(A)$
  5.  $[Q, D] = \text{eigensys}(A)$

## 6. Gráficos

### i. Algunos comandos

A continuación, se listan algunos comandos para graficar en MatLab:

Función Graficas	Significado																											
plot ()	<p>La función plot maneja varios parámetros como colores, continuidad de las líneas y marcas especiales en los puntos evaluados en el dominio.</p> <table><tr><th>COLORES</th><th>MARCAS EN EL PUNTO DOMINIO-RANGO</th><th>CONTINUIDAD DE LA LÍNEA</th></tr><tr><td>y</td><td>.</td><td>-</td></tr><tr><td>m</td><td>o</td><td>:</td></tr><tr><td>c</td><td>x</td><td>~.</td></tr><tr><td>r</td><td>+</td><td>--</td></tr><tr><td>g</td><td>s</td><td></td></tr><tr><td>b</td><td>*</td><td></td></tr><tr><td>w</td><td>^ &gt; &lt; v</td><td></td></tr><tr><td>k</td><td>d</td><td></td></tr></table>	COLORES	MARCAS EN EL PUNTO DOMINIO-RANGO	CONTINUIDAD DE LA LÍNEA	y	.	-	m	o	:	c	x	~.	r	+	--	g	s		b	*		w	^ > < v		k	d	
COLORES	MARCAS EN EL PUNTO DOMINIO-RANGO	CONTINUIDAD DE LA LÍNEA																										
y	.	-																										
m	o	:																										
c	x	~.																										
r	+	--																										
g	s																											
b	*																											
w	^ > < v																											
k	d																											
plot (x, y)	Genera una gráfica en las variables x e y.																											
plot (x, y, t)	Genera una gráfica en las variables x e y siendo t un parámetro.																											
plot3(x,y,z)	Genera una gráfica en las variables <u>x,y,z</u> .																											
ezplot(f)																												
Plot (x, y,s)	“s” es un carácter que indica la forma de la linea o curva graficada. Ver help plot.																											
figure	Las ventanas de gráficos son interpretadas por el programa como figuras, y cada una tiene asignado un número, para diferenciarlas de las demás. Permite tener en pantalla más de una ventana, cada una con un gráfico independiente.																											
	Es posible dar una numeración arbitraria a las ventanas que se abren para realizar las figuras, pasándole a “figure” el número con el cual se desea que la nueva ventana sea identificada: »figure(5); »figure(100);																											
Subplot(m,n,p)	Para tener más de un gráfico en una ventana. Este comando divide la figura en una matriz de m renglones y n columnas, por lo tanto crea mxn gráficos en una figura.																											

	 <pre> »subplot(2,1,1);  Filas      Columnas      Subgráfico </pre> <p>%Ejemplo:  %la ventana de la grafica se divide en 2 filas  t = 0:0.01:5;  y = cos(2*pi*t)+i*sin(2*pi*t);  figure;  subplot(2,1,1);  plot(t,abs(y));  subplot(2,1,2);  plot(t,angle(y)); % graficamos fase  pause  %la ventana de la grafica se divide en 2 columnas  figure;  subplot(1,2,1);  plot(t,real(y));  subplot(1,2,2);  plot(t,imag(y));  pause  %la ventana de la grafica se divide en 4 espacios  figure;  subplot(2,2,1);  plot(t,abs(y));  subplot(2,2,2);  plot(t,angle(y));  subplot(2,2,3);  plot(t,real(y));  subplot(2,2,4);  plot(t,imag(y)); </p>
<b>Para configurar el aspecto de las gráficas y ejes</b>	
title(string)	Los títulos se agregan una vez creada la figura. Establece la cadena string como título de la gráfica.
xlabel(string)	Establece la cadena string como etiqueta del eje x de la gráfica.
ylabel(string)	Establece la cadena string como etiqueta del eje y de la gráfica.
Grid on/off	Agrega una grilla al gráfico/ apaga la grilla.
Legend	Muestra la legenda en el gráfico.
Ginput	Permite tomar puntos de una función graficada y mostrarlos por pantalla.

Gtext('cadena')	Muestra la ventana gráfica, pone el cursor forma de cruz, y espera a que el botón del mouse o tecla sea presionado. La cruz se puede colocarse con el ratón (o con las teclas de flecha). Al pulsar el botón del mouse o cualquier tecla, escribe la cadena de texto en el gráfico de la ubicación seleccionada.
Hold on y off	HOLD ON mantiene el trazado actual y todas las propiedades de los ejes de modo que los comandos siguientes de los gráficos se añaden a la gráfica actual.

	HOLD OFF vuelve al modo de comandos por defecto por lo que plot restablece todas las propiedades del eje antes de dibujar nuevamente.
Axis ([xmin, xmax, ymin, ymax])	Controla la apariencia y escala de los ejes. Presenta varias opciones. Para personalizar los planteos, el modo más frecuente de hacerlo es mediante el comando axis. El comando axis cambia los ejes del diagrama actual, de modo que se muestra sólo la parte del eje que se desea. El comando axis se usa ingresando el siguiente comando justo después del comando plot (o cualquier comando que tiene un plot como una de sus salidas).

## ii. Práctica de comandos

### GINPUT, GTEXT

Un comando útil en Matlab es ginput. Ver >>help ginput (También gtext). El próximo programa dibuja  $y=x(x-1)\sin x$ , luego sobre la figura, permite tomar 6 puntos con el ratón, que al final muestra en la pantalla.

```
f='x*(x-1)*sin(x)'; ezplot(f);[x,y]=ginput(6); [x,y]
```

%Ahora con gtext

```
gtext({'This is the first line','This is the second line'})
```

```
gtext({'First line','Second line'},'FontName','Times','FontSize',12)
```

### PLOT (X, Y, S)

```
x = -pi:pi/10:pi;
```

```
y = tan(sin(x)) - sin(tan(x));
```

```
plot(x,y,'--rs','LineWidth',2,... % lineas de color rojo y cuadrados
```

```
'MarkerEdgeColor','k',... % remarcados en negro
```

```
'MarkerFaceColor','g',... %rellenos de color verde
```

```
'MarkerSize',10)
```

### GRID, XLABEL, YLABEL

```
x=0:pi/25:pi;y=sin(x); z=x.*x; plot(x,y,x,z);
```

```
grid;xlabel ('x '); ylabel ('valores ');
```



```
figure(gcf) % crea ventana y retorna control
```

## **HOLD ON, HOLD OFF**

```
x=-4:0.05:4;
```

```
y=exp(-0.5*x).*sin (5*x);
```

```
figure (1); plot(x,y, 'r');
```

```
xlabel ('eje x'); ylabel (' eje y');
```

```
hold on;
```

```
y=exp(-0.5*x).*cos (5*x);
```

```
plot(x,y, 'm');
```

```
grid; legend ('dos exponenciales...');
```

```
hold off
```

## **MESH**

```
close all % cierro figura
```

```
u=-8:0.5:8; v=u;
```

```
[U,V]=meshgrid(u,v);
```

```
R=sqrt(U.*U+V.*V)+eps;
```

```
W=sin(R)./R;
```

```
mesh(W)
```

## **SUBPLOT**

En una misma figura puede ponerse más de una línea empleando el comando subplot. El comando subplot le permite separar la figura en tantas figuras como se quiera, y ponerlas todas en una figura.

a)

```
x=[-3:0.4:3];
```

```
y=x;
```

```
close
```

```
subplot(2,2,1)
```

```
figure(gcf), fi=[-6*pi:pi/20:6*pi];
```

```
plot3(fi.*cos(fi),fi.*sin(fi),fi,'r')
```

```
[X,Y]=meshgrid(x,y);
```

b) Por ejemplo, suponga quisiera ver una senoide, un coseno , y una onda tangente graficadas en la misma figura, pero no en los mismos ejes. El siguiente código lo hará:

```
x = linspace(0,2*pi,50);  
  
y = sin(x);  
  
z = cos(x);  
  
w = tan(x);  
  
subplot(2,2,1)  
  
plot(x,y)  
  
subplot(2,2,2)  
  
plot(x,z)  
  
subplot(2,2,3)  
  
plot(x,w)
```

### **SUBPLOT**

```
x = 0:0.1:2*pi;  
  
y1 = sin(x);  
  
y2 = cos(x);  
  
subplot(2,1,1)  
  
plot(x,y1)  
  
title('y = sin(x)');  
  
subplot(2,1,2)  
  
plot(x,y2)  
  
title('y = cos(x)');
```

### **AXIS**

a) Grafique la función  $y=\exp(5t)-1$ :

```
t=0:0.01:5;  
  
y=exp(5*t)-1;  
  
plot(t,y)
```

Como puede ver, el gráfico tiende a infinito. Atendiendo al eje y (escala:  $8e10$ ), es claro que no puede verse mucho en este gráfico. Para tener una mejor idea de lo que está pasando en el ploteo, miremos el primer segundo de esta función. Escriba el siguiente comando en la ventana de comandos del Matlab.

```
axis([0, 1, 0, 50])
```

### iii. Comandos destacados

El siguiente cuadro representa un breve resumen de algunos de los comandos del software de aplicación a tener en cuenta para esta guía:

ss()	Create state-space model, convert to state-space model (modelo en el espacio de estado)
initial()	Respuesta libre del sistema. /condiciones iniciales para la respuesta temporal.
Sys	<p>Continuous-time dynamic system model (except frequency response data models). sys can represent a SISO or MIMO system, except that the 'matched' discretization method supports SISO systems only.</p> <p>sys can have input/output or internal time delays; however, the 'matched' and 'impulse' methods do not support state-space models with internal time delays.</p> <p>The following identified linear systems cannot be discretized directly:</p> <ul style="list-style-type: none"> <li>· idgrey models with FcnType is 'c'. Convert to idss model first.</li> <li>· idproc models. Convert to idtf or idpoly model first.</li> </ul> <p>For the syntax [sysd,G] = c2d(sys,Ts,opts), sys must be a state-space model.</p> <p>Crea un objeto de tipo sistema. Ejemplos:</p> <ul style="list-style-type: none"> <li>– sys = tf(num,den)</li> <li>– sys = zpk(ceros,polos,k)</li> <li>– sys = ss(A,B,C,D)</li> </ul>
Conversión entre modelos.	<pre> [num,den] = ss2tf(a,b,c,d) [z,p,k] = ss2zp(a,b,c,d) [a,b,c,d] = tf2ss(num,den) [z,p,k] = tf2zp(num,den) [a,b,c,d] = zp2ss(z,p,k) [num,den] = zp2tf(z,p,k) [r,p,k] = residue(num,den) %tf2r [num,den] = residue(r,p,k) %r2tf </pre>

## SECCIÓN PRÁCTICA COMPETENCIAS



### Ejercitación

1. Obtenga la representación en MatLab del sistema para el:

**\*\*Modelo del ejercicio de los 3 tanques**

**\*\*Modelo de masa simple, resorte, y amortiguador**

Se solicita:

a- representación matricial, solución del sistema mediante laplace y gráfica de la solución del sistema.

b- Caracterización de cada sistema

**2.** Representa la respuesta del sistema térmico de los recintos (evaluando solo a  $x_1$ ) frente a un impulso con las condiciones iniciales siguientes: (grafique)

»  $x_0 = [20 ; 100]$  % suponiendo temperaturas iniciales de 20 y 100 grados en cada recinto.

Se recomienda la utilización de: `ss()`, `initial()`.