INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

Ingeniería Electrónica

Interfaz Gráfica Para El Análisis, Diseño, Simulación y Control De Sistemas Lineales Continuos Y Discretos Utilizando Scilab

---

Reporte

de Residencia

Residente:
Jesús Ernesto Velázquez López

05270323

Asesor:
M. C. Francisco Ronay López Estrada

TUXTLA GUTIÉRREZ, CHIAPAS, MÉXICO A 14 DE JUNIO DE 2010

Índice general

Capítulo 1

Generalidades

1.1. Introducción

En este trabajo se presenta el desarrollo de una Interfaz Gráfica de Usuario (GUI, del inglés Graphical User Interface) especializada en Teoría de Control. Dicha GUI tiene por objetivo brindar al usuario las herramientas para analizar, simular y controlar sistemas en tiempo continuo y discreto. Es por ello que la principal contribución será lograr que el sistema realice los complejos cálculos matemáticos que en Teoría de Control se requieren, de forma automática, con lo cual el usuario se concentrará en la parte que necesita, la de control y no en la programación del sistema en el software.

Esta Interfaz realizará aplicaciones de control tales como análisis de tipo temporal (respuesta a una entrada escalón, impulso, onda cuadrada y random). Análisis en dominio de la frecuencia (Diagramas de Bode, Nyquist y Black). Diseño de sistema de control mediante compensadores de atraso y de adelanto de fase. Diseño de sistemas de control mediante control P, PI, PD, PID en tiempo continuo y discreto. Simulación de sistemas en espacio de estados. Y análisis de estabilidad mediante gráficas de polos y lugar de las raíces.

La Interfaz se creará bajo la filosofía del software libre y el OpenSource. En este contexto, utilizaremos el Sistema Operativo Linux. Como plataforma para programar la Interfaz Gráfica, se utilizará el software matemático SciLab (de INRIA).

Acerca de Scilab podemos mencionar, que es una herramienta muy similar a MatLab (de MathWorks) el popular software matemático. Scilab es un lenguaje de programación matemático de alto nivel de cálculo científico, de uso libre, con una interfaz amigable y disponible para los sistemas operativos más populares. Scilab cuenta con herramientas que permiten realizar cálculos numéricos, cálculos simbólicos, manejo de álgebra lineal, matrices dispersas, gráficos en 2-D y 3-D, animación, polinomios y funciones racionales; permite también integrar códigos de lenguajes de programación como C, C++, Fortran y Java; además de contar con su propio simulador: Xcos.

1.2. Información general de la institución o empresa donde se desarrollo el proyecto

La presente residencia profesional se llevó a cabo en el Instituto Tecnológico de Tuxtla Gutiérrez, con domicilio en Carretera Panamericana Km. 1080, Tuxtla Gutiérrez, Chiapas, México, C.P. 29000, Apartado Postal 599.

Misión

Formar de manera integral profesionales de excelencia en el campo de la ciencia y la tecnología con actitud emprendedora, respeto al medio ambiente y apego a los valores éticos

Visión

Ser una Institución de excelencia en la educación superior tecnológica del Sureste, comprometida con el desarrollo socioeconómico sustentable de la región

Valores

- El ser humano
- El espíritu de servicio
- El liderazgo
- El trabajo en equipo
- La calidad
- El alto desempeño

1.3. Área específica relacionada directamente con el proyecto

El proyecto está destinado en primera instancia, para brindar una herramienta didáctica a los alumnos y catedráticos de clases de Teoría de Control, (Control I, Control II, Control III).

1.4. Antecedentes

Algunos estudios recientes han demostrado que el uso de software libre GPL son muy útiles para la enseñanza en las carreras de ingeniería. Desafortunadamente la documentación de este tipo de software es pobre. La carencia de este tipo de soporte es una de las razones por la cual los usuarios prefieren el uso de herramientas licenciadas como MATLAB.

Sin embargo, la comunidad de Scilab está trabajando arduamente para proveer la documentación necesaria que permita a los usuarios simular sistemas lineales en su arquitectura. Uno de los esfuerzos más importantes en este rubro es el libro "Modeling and Simulación in Scilab/Scicos", este libro provee una amplia documentación acerca del contenido de los toolboxes de SciLab, así como un resumen completo de cada uno de los comandos del software y posibles aplicaciones.

A pesar de que este libro es la principal fuente de documentación sobre SCILAB, no ofrece ninguna documentación sobre la programación de interfaces gráficas o adquisición de datos en tiempo real. En la pagina oficial de Scilab, se pueden encontrar importantes esfuerzos sobre como realizar interfaces gráficas. Desafortunadamente la mayoría de estas contribuciones son para aplicaciones en métodos numéricos y solo se puede encontrar pocos trabajos relacionados con interfaces gráficas para sistemas de control. Algunos de ellos son:

GUI RLTOOL fue desarrollado por estudiantes de postgrado de la India y, representa un importante contribución para el análisis de sistemas en tiempo continuo. Ésta GUI depende a la vez de recursos Tcl/Tk que son ejecutados en Scilab mediante un interprete de código incorporado en este último. Pero cuenta con la desventaja está basado en una de las versiones anteriores de Scilab, lo cual provocó que el GUI no sea compatible con versiones recientes, dejándolo desactualizado y no funcional, puesto que algunos de los recursos utilizados en el GUI han sido modificados o eliminados.

Control Design Tool Es un GUI simple, el cual permite obtener la respuesta de una función de transferencia de un sistema, el cual puede ser modificado mediante una ganancia ajustable, o bien mediante la aplicación de una retroalimentación, cerrando con esta el lazo del sistema.

Otros trabajos importantes son el desarrollo de toolboxes especializados (no gráficos) tales como toolbox de lógica difusa, toolbox para la adquisición de datos, toolbox para el manejo de sistemas numéricos, entre otros. A pesar de que la documentación sobre scilab es pobre, en los libros especializados pueden encontrarse métodos detallados sobre la teoría de control, por lo cual es factible lograr diseñar un toolbox especializado basándose únicamente en los métodos numéricos manejados por SCILAB y la teoría de control.

1.5. Planteamiento del problema

La Teoría de Control es un campo de estudio muy extenso, que hoy en día tiene aplicaciones en prácticamente todo los procesos. Para un estudiante de Control es necesario comprender totalmente

los conceptos, términos, expresiones formas de modelado y aplicaciones de Control. Por lo cual, se debe hacer uso de herramientas computacionales que mediante el uso de métodos numéricos resuelvan problemas matemáticos complejos.

Hoy en día se han adoptado, diversos programas matemáticos para el apoyo y comprobación de resultados, en el aspecto académico, y como herramienta esencial en el área de trabajo. Ya que, gracias a los avances en software, podemos obtener las soluciones de complejas operaciones matemáticas en un mínimo de tiempo, lo cual optimiza y facilita el estudio de los sistemas dinámicos.

Como ya se mencionó, el diseño de sistemas de control mediante software ha adquirido relevancia en los últimos años, esto debido a que los métodos empleados en Teoría de Control han evolucionado de forma tal que han demostrado que su uso tiene múltiples aplicaciones. Las aplicaciones abarcan prácticamente todas las disciplinas tales como, sistemas de control de temperatura, PH, O2 en la ingeniería bioquímica; sistemas de control de posición, velocidad, giro, señales, en automóviles, aviones, entre otros.

Derivado de lo anterior, surgen dos principales problemas:

- El alumno de Teoría de Control debe aprender términos de Control y el uso del software matemático y necesariamente las líneas de código que este requiera.

- La enorme preferencia al uso de Software de paga, el cual tiene grandes costos, trayendo consigo la piratería o la abstención a su uso.

Cuando un alumno, no está completamente concentrado en los cálculos que Control requiere por estar preocupado por la programación de su modelo en el software matemático, pueden ocurrir dos cosas: que se pierda mucho tiempo, que es algo muy considerable, o en el peor de los casos, que los cálculos no sean correctos.

Para el catedrático, la doble tarea de enseñar, tanto la Teoría como el uso de software, se convierte en pérdida de valioso tiempo que podría ser mejor empleado en abordar de una mejor manera los temas que comprende el curso.

1.6. Nombre del proyecto

Interfaz Gráfica Para El Análisis, Diseño, Simulación y Control De Sistemas Lineales Continuos Y Discretos Utilizando Scilab

1.7. Objetivos

1.7.1. General

Desarrollar una interfaz gráfica que permita analizar, simular y controlar sistemas lineales en tiempo continuo y en tiempo discreto.

1.7.2. Específicos

- Desarrollar la interfaz usando la herramienta de software libre SCILAB
- Analizar y controlar sistemas continuos
- Analizar y controlar sistemas discretos
- Analizar y controlar sistemas en espacio de estados
- Generar una herramienta de apoyo para las materias de Control I, II, III

1.8. Justificaciones del proyecto

De acuerdo con la problemática planteada con anterioridad, es muy conveniente crear una Interfaz Gráfica, con la que se puedan Analizar, Diseñar, Simular Sistemas Lineales ya sea en Tiempo Continuo o en Tiempo Discreto; pues con ella, tanto el alumno como el catedrático no necesitarán programar las ecuaciones desde líneas de código del software, sino que en un ambiente gráfico, intuitivo de fácil uso, trayendo consigo una mejor comprensión de las técnicas de Control.

Uno de los problemas al utilizar herramientas computacionales, es el elevado costo de éstas, por ello en este trabajo se optó por el uso de una herramienta de Software Libre (Free Software), el cual, no es muy popular entre las personas, pues tienen el erróneo estereotipo de que lo que es gratis no es bueno o no funciona correctamente; eso es totalmente falso, pues actualmente se puede encontrar software libre multiplataforma capaz de sustituir en un cien por cien al software licenciado.Abocándonos a Software Libre Matemático más específicos para Teoría de Control, uno de los más populares es SciLab.

SciLab es una herramienta libre al igual que MatLab esta basado en el manejo de métodos numéricos y matrices para la solución de problemas matemáticos. Tiene como principales ventajas:

- Estar bajo la filosofía de Código Libre
- Tener un diseño nada complejo
- Ser multiplataforma y sin requerir demasiados recursos de hardware

1.9. Alcances y limitaciones del proyecto

Esta Interfaz Gráfica de Usuario es capaz de manejar funciones de transferencia como sistemas de tiempo continuo y discreto. También podemos analizar sistemas en espacio de estados, de igual manera en tiempo continuo y discreto.

A la vez podrá ser distribuida mediante una página web, lo que no limitará su distribución al sector al cual está destinado.

Aunque no está diseñado en primera instancia, puede también utilizarse en otras disciplinas como Señales y Sistemas, Electrónica Analógica I, II y III.

Puede verse limitada por el hardware utilizado por el usuario, principalmente lo que respecta a la pantalla (display), ya que de su resolución dependerá como la Interfaz pueda verse.

1.10. Metodología para el desarrollo del proyecto

La investigación y desarrollo del proyecto se resume en 3 etapas de acuerdo a la complejidad. La primera se concentra en el análisis y solución de los sistemas en tiempo continuo. La segunda etapa se concentrara en los sistemas de tipo discreto. Y por último La tercera etapa en la solución de los sistemas en espacio de estados. A continuación se detallan cada una de las etapas.

Etapa 1: Sistemas en tiempo continuo.

En esta etapa se trabajó en el modelado de sistemas mediante ecuaciones diferenciales y su conversión a modelos tipo función de transferencia. Se trabajara también en la programación de la interfaz del GUI. La respuesta de los sistemas ante una entrada de onda cuadrada, senoinal, escalón, impulso, pulso. Y el estudio de estabilidad de los sistemas mediante gráficas de polos y ceros y lugar geométrico de las raíces. Y el diseño y aplicación de control por compensadores de fase y de la familia PID.

Etapa 2. Sistemas en tiempo discreto

Para diseñar sistemas de control en tiempo discreto es necesario que los sistemas en tiempo continuo normalmente representado por ecuaciones diferenciales o funciones de transferencia por transformada de Laplace, tengan que convertirse a ecuaciones en diferencias o funciones de trasferencia por transformada Z. Estas ecuaciones discretas se realizan mediante un retenedor de orden cero (ZOH). Al cambiar el tipo de representación cambian también las reglas para analizar los sistemas. Mas sin embargo, al igual que los sistemas en tiempo continuo se puede seguir trabajando con técnicas de respuesta en frecuencia y de tipo temporal, claro bajo el esquema de sistemas muestreado. Es por ello el reto en esta etapa sera lograr manejar los sistemas en tiempo discreto

y realizar las mismas pruebas de respuesta frecuencial y temporal que en los sistemas de tiempo continuo. Así, como también el diseño de controladores tipo P, PI, PD y PID.

Etapa 3: Sistemas en espacio de estados

En la vida real muchos de los sistemas tienen múltiples entrada (comando de control de velocidad, posición ángulo...) y múltiples salidas (manejo de los actuadores como válvulas de paso), estos sistemas son conocidos como sistemas tipo MIMO. Los sistemas MIMOs se majen mediante representaciones en espacio de estados. El modelado en espacio de estados maneja una representación matricial y pueden manejarse en tiempo continuo y en tiempo discreto. El reto en esta etapa sera lograr manejar sistemas mimos con “n” número de entradas y “n” número de salidas, todo manejado mediante la programación y el uso de la interfaz gráfica.

Capítulo 2

Marco Teórico

Existen dos categorías generales de software científico:

1. Los sistemas de cálculo de álgebra, que realizan cálculos simbólicos y cálculos numéricos
2. Diseñados específicamente para aplicaciones científicas

Los ejemplos más conocidos en la primer categoría son Maple, Mathematica, Maxima, Axiom y MuPAd. La segunda categoría representa un enorme mercado dominado por MATLAB, Scilab y Octave, cabe destacar que los dos últimos, son programas de distribución libre, de los cuales Scilab se encuentra dentro de los más populares.

2.1. Scilab

Scilab es un lenguaje interpretado con objetos dinámicos. Scilab se encuentra disponible en un formato binario para las principales plataformas (sistemas operativos): Unix/Linux (los principales desarrollos del software se encuentran presentados en Linux), Windows y MacOSX.

Scilab fue originalmente llamada Basile y fue diseñado por INRIA como parte del proyecto Meta2. Continuando su diseño bajo el nombre de SciLab por el grupo Scilab, un equipo de investigadores de INRIA Metalau y ENPC. Desde 2004, el diseño de Scilab ha sido coordinado por un consorcio.

Scilab puede ser usado como un lenguaje para prueba o ejecución de algoritmos o para realizar cálculos numéricos. Pero también es un lenguaje de programación, y la librería standard de Scilab contiene cerca de 2000 funciones. La sintaxis de Scilab es muy simple, y el uso de matrices, que son el objeto fundamental de cálculos científicos, es facilitado con específicas funciones y operadores. Éstas matrices pueden ser de diferentes tipos real, compleja, polinomial y racional.

Scilab es principalmente dedicado para cálculos científicos, y es provisto de enorme librería con funciones para áreas como álgebra lineal, integración numérica y optimización. Además es

simple de extender el entorno de Scilab. Cualquiera puede importar fácilmente nuevas funciones de librerías externas hacia Scilab usando enlaces estáticos o dinámicos. También es posible definir nuevos tipos de datos y nuevas funciones usando estructuras de Scilab. Numerosos toolboxes que agregan especiales funciones a Scilab, se encuentran disponibles en su página oficial.

Scilab también incluye muchas funciones de visualización incluyendo 2D, 3D, gráficas de contorno y paramétricas, y animación. Las gráficas pueden ser exportadas en varios formatos como Gif, Postscript, Postscript-Latex, y Xfig. Agregado a las funciones para interfaces de usuario, se encuentran disponibles funciones de Tcl/Tk que pueden ser utilizados para el diseño de sofisticadas GUI's (Interfaces Gráficas de Usuario).

Scilab es un enorme paquete de software que contiene aproximadamente 13,000 archivos, mas de 400,000 lineas de código fuente (en C y Fortran), 70,000 lineas de código Scilab (librerías especializadas), 80,000 líneas de ayuda en línea, y 18,000 líneas de archivos de configuración. Dichos archivos incluyen:

- Funciones elementales de cálculo científico
- Álgebra lineal, matrices dispersas
- Funciones polinomiales y racionales
- Control clásico y robusto, optimización por LMI
- Métodos no lineales (Optimización, solución de ODE y DAE, Scicos, ahora Xcos, como un híbrido modelador de sistemas dinámicos y simulador)
- Procesamiento de señales
- Muestreo aleatorio y estadístico
- Gráficas (algoritmos, visualización)
- Gráficos, animación
- Paralelismo usando PVM
- Traductor MATLAB-to-Scilab
- Un extenso número de contribuciones para varias áreas

2.1.1. Iniciando Scilab

Scilab está disponible para su descarga desde la dirección http://www.scilab.org/, encontrando versiones para usuarios Unix/Linux, Windows, MacOSX, a su vez podemos descargar versiones de instalación o versiones binarias para ejecución del código fuente y en diversos idiomas incluidos el español.

Ejecutando Scilab se abre una ventana de comando (Figura 2.1). Ésta es una ventana interactiva donde el usuario está invitado a ingresar comandos delante al símbolo de sistema de Scilab (–>). El comando se ejecuta al presionar la tecla “enter” (retorno de carro), después Scilab ejecuta el comando y regresa el control al usuario mediante un nuevo símbolo (–>).

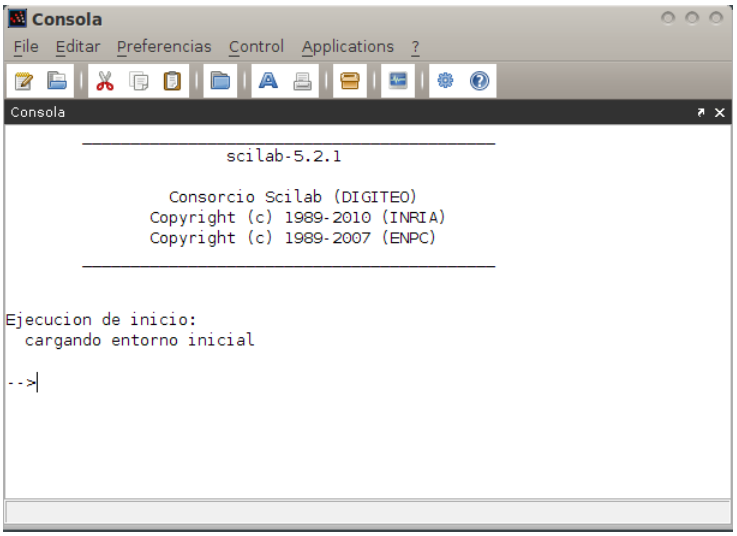


Figura 2.1: Ventana de Comandos de Scilab

La edición de funciones en la ventana de comando es muy limitada, por lo cual Scilab incluye su propio editor. Llamado desde la ventana principal mediante el comando “editor” (“edit” en inglés) o desde la barra de menú. Abriéndose el siguiente editor de texto (Figura2.2).

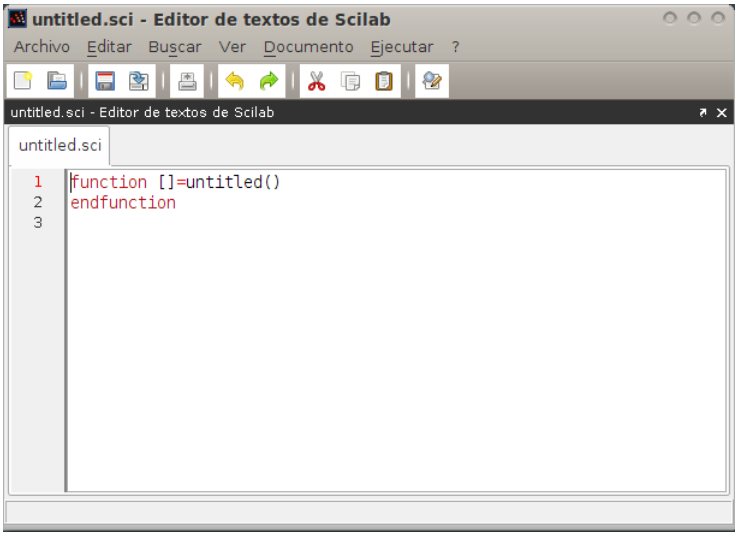


Figura 2.2: Editor de funciones

2.1.2. Documentación

Scilab cuenta con una comprensiva ayuda en su propio manual y en línea, la cuál puede ser consultada mediante comandos “help” y “apropos”. Para consultar la página del manual correspondiente a una función de Scilab, el comando “help” debe estar seguido por el nombre de la función a usar. Lo anterior abre el buscador de ayuda en la correspondiente página a la función consultada. La página del manual contiene una detallada descripción de la función y algunos ejemplos en los que ésta puede ser utilizada. Los ejemplos pueden ser copiados y ejecutados en la ventada de comandos de Scilab, en versiones recientes podemos seleccionar el ejemplo y con -click derecho- nos dará la opción de “Ejecutar la selección en Scilab”.

El buscador también puede ser llamado desde el menú en la barra de herramientas superior de la ventada principal, y contiene una lista de todas las funciones clasificadas en diferentes capítulos (Figura2.3). La página del manual de una función puede ser obtenida clickeando en su nombre.

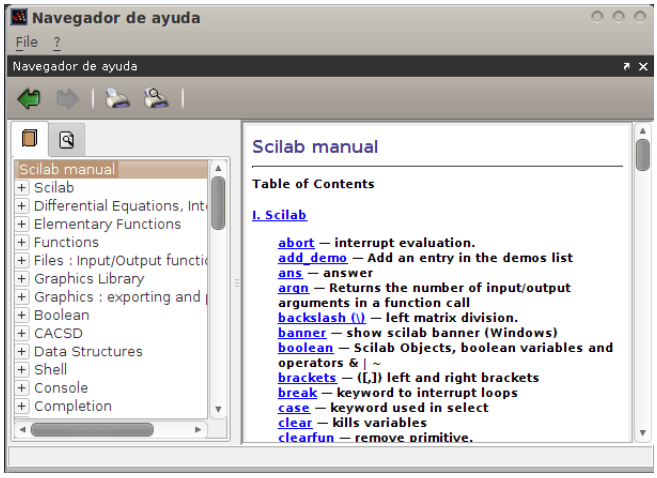


Figura 2.3: Manual de Ayuda

Los demos incluidos también son una fuente de inspiración. Éstos presentan simples ejemplos de programación de Scilab de situaciones cotidianas para los usuarios. Los demos de gráficos, por ejemplo, proporcionan una visión de lo que Scilab puede hacer en cuestiones de visualización. Los códigos fuentes de demos dan un buen punto de partida para el diseño de complejas aplicaciones.

2.2. Interfaz Gráfica de Usuario

2.2.1. Definición

La interfaz gráfica de usuario, conocida también como GUI (del inglés graphical user interface) es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes

y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso, consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo de una máquina o computador.

Habitualmente las acciones se realizan mediante manipulación directa, para facilitar la interacción del usuario con la computadora. Surge como evolución de los intérpretes de comandos que se usaban para operar los primeros sistemas operativos y es pieza fundamental en un entorno gráfico. Como ejemplos de interfaz gráfica de usuario, cabe citar los entornos de escritorio Windows, el X-Window de GNU/Linux o el de MacOSX, Aqua.

En el contexto del proceso de interacción persona-ordenador, la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

La interacción entre las personas y los ordenadores (IPO) en la actualidad se realiza principalmente a través de una interfaz gráfica de usuario, un tipo de interfaz de usuario, compuesto por metáforas gráficas inscritas en una superficie de contacto en adición de otros elementos "semánticos" como los iconosonidos (signos sonoros) o interfaces humanos (dispositivos de entrada) necesarios para posibilitar dicha interacción con los signos-objetos en la interfaz gráfica.

2.2.2. Crear una Interfaz Gráfica de Usuario mediante Scilab

Dentro de las herramientas que el software Scilab nos brinda, se encuentran recursos que nos permiten la creación de interfaces gráficas de usuario. Aunque éstas herramientas estuvieron desafortunadamente muy descuidadas en versiones anteriores, a partir de la versión 5.0, toda la interfaz gráfica de usuario ha sido mejorada y reimplementada utilizando lenguaje java.

Dentro de los recursos de Scilab más utilizados para la creación de GUI's se encuentran:

figure Consideradas como la base de la Interfaz

uicontrol Botones, entradas y/o salidas de texto, etc

uimenu Menús y submenús

Comando figure

Algorithm 2.1 Secuencia de llamado figure

```
f=figure(num);
f=figure("PropertyName1", Propertyvalue1, ...,
..., "PropertyNameN", PropertyvalueN);
```

Descripción Esta rutina crea una figura. Si un Identificador (ID) es asignado, la figura correspondiente a ese ID es creada. En otro caso, la ventana es creada con el primer ID libre, que es el menor ID actualmente no usado para otra ventana.

Parámetros

num ID de la ventana a crear. Si no es especificado, se utiliza el primer ID disponible.

PropertyName{1,...,N} Nombre de la propiedad a configurar. Las propiedades se en listan abajo.

PropertyValue{1,...,N} Valor dado para la propiedad correspondiente.

f Nombre de la variable asignada a la reciente ventana creada o manejador.

Propiedades

background En un vector real [1,3] o cadena de caracteres del color de fondo de la figura. Un color es especificado como valores Rojo, Verde y Azul. Los valores son reales dentro de [0,1]. El color puede darse como un vector real, [R,G,B] o como caracteres separados por un "|", "R|G|B".

Figure_name Cadena de caracteres, permite dar el título a la figura.

ForegroundColor En un vector real [1,3] o cadena de caracteres del color de frente de la figura. Un color es especificado como valores Rojo, Verde y Azul. Los valores son reales dentro de [0,1]. El color puede darse como un vector real, [R,G,B] o como caracteres separados por un "|", "R|G|B".

Position Permite controlar el aspecto geométrico de la figura. En un vector real [1,4] como [x y width height] donde las letras representan la posición 'x' de la esquina superior izquierda, la posición 'y' de la esquina superior izquierda, la anchura y altura de la ventana virtual de gráficos (la parte de la figura donde contiene los uicontrols y las gráficas). También puede configurar las propiedades dando una cadena de caracteres donde los parámetros se separan con "|", como sigue "x|y|width|height".

Tag Cadena de caracteres generalmente se usa para identificar la figura. Esto permite dar un nombre a la figura. Ocasionalmente se usa en conjunto con findobj()

Userdata Este puede se usado asociando algunos objetos de Scilab a la figura.

Example 1. Crear una figura dado un ID

Algorithm 2.2 Ejemplo crear figura dado un ID

```
// Crea un figura dando un ID figure_id==3
h=figure(3);
// Agregar un uicontrol de tipo texto en la figura 3
uicontrol(h, "style", "text", ...
   "string", "This is a figure", ...
   "position", [50 70 100 100], ...
   "fontsize",15);
// Create figure having figure_id==1
figure();
// Add a text uicontrol in figure 1
uicontrol("style", "text", ...
   "string", "Another figure", ...
   "position", [50 70 100 100], ...
   "fontsize", 15);
// Cerrar la última figura creada (en este caso figura 1)
close();
// cerrar figura 3
close(h);
```

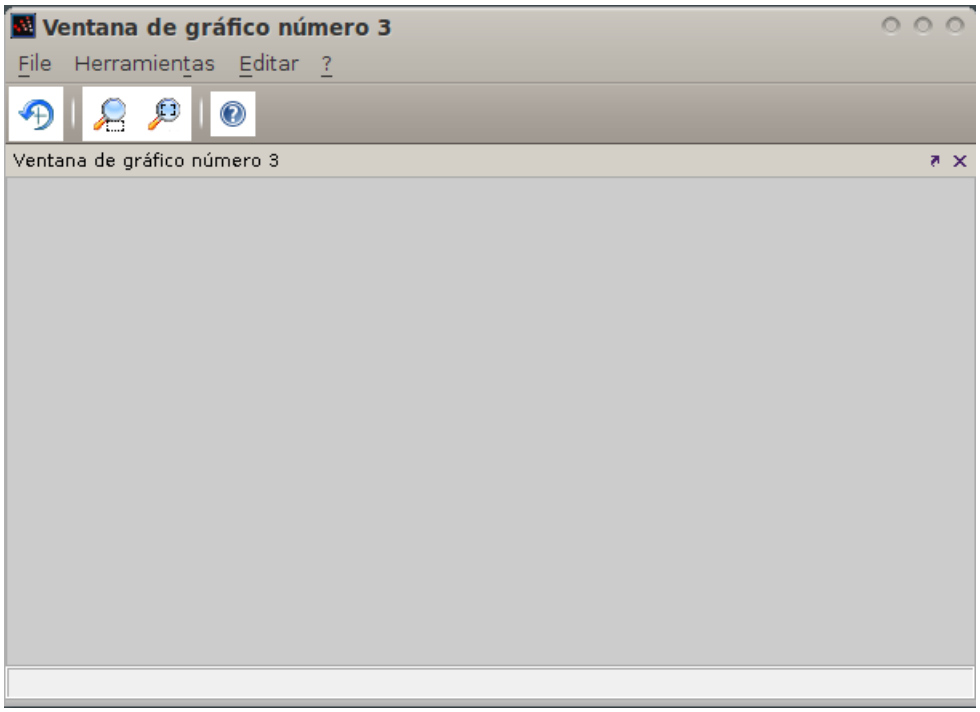


Figura 2.4: Ejemplo1: Crear una figura dado un ID

Comando uicontrol

Uicontrol crea un objeto en la Interfaz Gráfica de Usuario

Algorithm 2.3 Secuencia de llamado uicontrol

```
h = uicontrol(PropertyName,PropertyValue,...)
h = uicontrol(parent,PropertyName,PropertyValue,...)
h = uicontrol(uich)
```

Descripción Esta rutina crea un objeto dentro de un figura

Si el nombre de la figura es dado (como el primer parámetro), el uicontrol es creado en dicha figura. Si la figura no es dada, el uicontrol es creada en la última figura creada (podemos identificar ésta con el comando gcf()). Si no existe una figura creada, entonces ésta es creada antes de crear el uicontrol.

Después de creado el uicontrol las propiedades dadas como parámetros son configuradas inmediatamente con sus correspondientes valores. Esto equivale a crear el uicontrol y después modificar sus propiedades con el comando set(). Sin embargo, es más eficiente definir en un principio las propiedades. Esta es particularmente cierto en la propiedad "Style". De hecho el valor por defecto de esta propiedad es "Pushbutton". Entonces si tu no configuras esto al crear el objeto, un button será creado, pero puede ser transformado en otro uicontrol mediante la instrucción set(uicontrol,"style", ...).

h = uicontrol(PropertyName, PropertyValue,...) crea un uicontrol y asigna las específicas propiedades y valores a este. Esto asigna los valores por defecto a las propiedades no especificadas. El uicontrol por defecto es el "Pushbutton". La asociación por defecto es a la última figura creada.

h = uicontrol(parent,PropertyName, PropertyValue,...) crea un uicontrol en el objeto especificado por el manejador, parent. Si también se especifica un diferente valor para la propiedad Parent, el valor de esta toma prioridad. Parent es el manejador o nombre de la variable asignada a una figura.

h = uicontrol(uich) enfoca al uicontrol especificado por el uich.

Propiedades

Background Vector real [1,3] o cadena de caracteres. Representa el color de fondo del uicontrol, especificado en valores Rojo, Verde, Azul (RGB). Son valores reales dentro del rango [0,1]. El color puede darse como un vector real, como [R,G,B] o como una cadena donde cada valor es separado por un "|", como "R|G|B".

Callback Caracteres. Instrucción evaluada por el interprete de Scilab cuando el uicontrol es activado. (Por ejemplo al dar click en un botón).

Enable {on}|off. Activar o desactivar el uicontrol. Si esta propiedad esta configurada como "on" (por defecto), el uicontrol es puede usar, pero si esta propiedad está en "off", el uicontrol no responderá a la acción del mouse.

FontAngle {normal} | italic | oblique. Para el control del contenido de algunos 'text', esta propiedad configura la inclinación de la tipografía usada.

FontSize Escalar. Para el control del contenido de algunos 'text', esta configura el tamaño de letra en FontUnits.

FontUnits {points} | pixels | normalized. Para el control del contenido de algunos 'text', esta especifica la unidad con la cual FontSize está especificada.

FontWeight light | {normal} | demi | bold. Para el control del contenido de algunos 'text', configura el ancho de la tipografía usada.

FontName Cadena de caracteres. Usada para elegir la tipografía del texto.

ForegroundColor Vector real [1,3]o cadena de caracteres. Es el color del frente del uicontrol, especificado en valores Rojo, Verde, Azul (RGB). Son valores reales dentro del rango [0,1]. El color puede darse como un vector real, como [R,G,B] o como una cadena donde cada valor es separado por un "|", como "R|G|B".

HorizontAlalignment left | {center} | right. Configura la alineación horizontal del texto en el uicontrol. Esta propiedad solo tiene efecto en Text, Edit y CheckBoxes

ListboxTop Escalar. Para ListBox, esta propiedad especifica que valor de la lista aparecerá en la primer línea del área visible de la lista.

Max Escalar. Especifica el máximo valor que la propiedad "Value" puede tomar. Lo cual tiene diferentes aplicaciones para cada uicontrol:

- CheckBoxes: Max es el valor que la propiedad "Value" toma cuando el uicontrol es activado.
- Sliders: Máximo valor del slider.
- ListBoxes: Si (Max-Min)>1 la lista presenta múltiples selecciones, en caso contrario no.

Min Escalar. Especifica el valor más pequeño para la propiedad "Value". Lo cual tiene diferentes aplicaciones para cada uicontrol:

- CheckBoxes: Min es el valor que la propiedad "Value" toma cuando el uicontrol es desactivado.

- Sliders: Mínimo valor del slider.
- ListBoxes: Si (Max-Min)>1 la lista presenta múltiples selecciones, en caso contrario no.

Parent Manejador (Variable). Manejador o variable de la asociación del uicontrol. Cambiando esta propiedad permite mover el uicontrol de una figura a otra.

Position Vector real [1,4] o cadena de caracteres. Esta propiedad puede ser usada definir o tomar la configuración geométrica de un uicontrol. Es un vector [x y w h] donde las letras representan la posición en 'x' de la esquina inferior izquierda, la posición 'y' de la esquina inferior izquierda, la anchura (w) y altura (h) del uicontrol o una cadena de caracteres donde cada valor es separado por un "|", como "x|y|w|h". Las unidades son determinadas por la propiedad "Units".

Relief flat | groove | raised | ridge | solid | sunken. Es la apariencia del borde del uicontrol:

- Pushbutton: "raised"
- Edits: "sunken"
- Para otros estilos de uicontrol: "flat"

SliderStep Vector real [1,2] [corto grande], El paso pequeño representa el movimiento cuando se clickea sobre el slider o pulsando las teclas de desplazamiento (cuando el slider está enfocado); el paso grande es el movimiento cuando se usa Ctrl+(teclas de desplazamiento). Si el paso grande es omitido, su valor por defecto es 1/10 de la escala.

String Cadena de caracteres. Esta propiedad representa el texto que aparece en un uicontrol (excepto para estilos Frame y Slider). Para ListBoxes y PopupMenus, el valor puede ser un vector de caracteres o un vector donde los items son separados por un "|". Para uicontrols tipo Text, estos caracteres pueden contener código de texto en HTML.

Style {pushbutton} | radiobutton | checkbox | edit | text | slider | frame | listbox | popupmenu. Es el estilo del uicontrol. Aquí una breve descripción de cada uno de ellos:

Pushbutton	Un botón rectangular, regularmente usado para llamado de funciones
Radiobutton	Un botón con estados. Los RadioButtons son configurados para ser mutuamente exclusivos
Checkbox	Un botón con estados. Usados para múltiples elecciones independientes
Edit	Zona de edición o entrada de caracteres
Text	uicontrol con texto. Generalmente estático
Slider	Un control con escalas, que es una barra de desplazamiento usada para definir valores dentro de un rango con el mouse

Frame Uicontrol que representa o limita una zona usada para agrupar controles relacionados

Listbox Representa una lista de opciones que pueden ser recorridos con una barra de desplazamiento. Las opciones pueden seleccionarse con el mouse

Popupmenu Es un botón que hace aparecer un menú cuando se clickea

Tag Caracteres Esta propiedad es generalmente utilizada para identificar uicontrols. Permita asignarles un nombre. A veces se utiliza junto con la instrucción findobj()

Units {points} | pixels | normalized. Configura las unidades usadas al especificar la propiedad "Position"

Userdata Datos de Scilab. Puede ser usado para relacionar objetos de Scilab (caracteres, matrices de caracteres, matrices mxn) a un uicontrol

Value Vector o escalar. Valor del uicontrol, este depende del tipo de uicontrol

- CheckBoxes, RadioButtons: Su valor al estar activo es el definido por "Max", caso contrario es el definido por "Min"
- ListBoxes, PopupMenus: Es un vector de índices correspondientes a los índices de la entidad seleccionada en la lista. 1 es para la primer entidad en la lista
- Sliders: valor indicado por la barra de desplazamiento

Verticalalignment top | {middle} | bottom. Configura la alineación vertical del texto dentro del uicontrol. Esta propiedad solo tiene efecto en estilos Text y CheckBox

Visible {on}|off. Define la visibilidad del uicontrol. Si la propiedad está en "on" (por defecto) el uicontro es visible, pero si esta propiedad está en "off", el uicontrol no aparecerá en su correspondiente figura.

Example 2. Crear un uicontrol dentro de una figura

Algorithm 2.4 Ejemplo crear uicontrol

```
// crear una figura
f=figure();
// crear un listbox
h=uicontrol(f,'style','listbox',...
  'position', [10 10 150 160]);
 // llenar la lista
set(h, 'string', "item 1|item 2|item3");
//seleccionar el valor 1 y 3 de la lista
set(h, 'value', [1 3]);
//cerrar la figura
close(f);
```

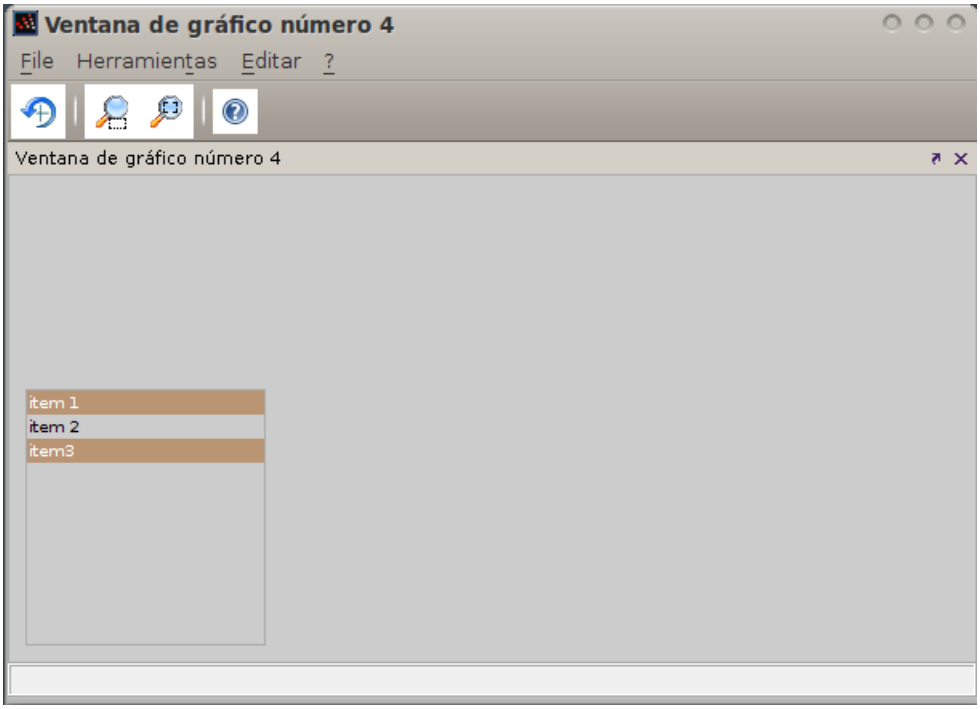


Figura 2.5: Ejemplo2: Uso del comando uicontrol

Comando uimenu

Algorithm 2.5 Secuencia de llamado uimenu

```
h=uimenu([prop1,val1] [,prop2, val2] ...)
h=uimenu(parent,[prop1, val1] [,prop2, val2] ...)
```

Descripción Esto permite crear menús dentro de una figura. Si la propiedad "parent" es una figura, entonces el menú será agregado a la barra de menús de esa figura. Si "parent" es un menú, entonces el nuevo objeto será creado dentro de ese menú, permitiendo crear submenús en cascada.

Parámetros

parent Manejador o nombre de la variable a la que se asociará el menú

porp{1,2...} Nombres de las propiedades a configurar

val{1,2...} Valores de los objetos de Scilab a afectar con la correspondiente propiedad

h Manejador o nombre de la variable del menú correspondiente

Propiedades

Callback Caracteres. Instrucción evaluada por el interprete de Scilab cuando el menú es activado. Bajo MacOSX, el callback no puede ser ejecutado por un "button menú" (un menú sin "children"), se deberá especificar por lo menos un "child"

Enable {on}|off. Activar o desactivar el menú. Si esta propiedad está en "on" (por defecto), el menú puede usarse, está funcionando, pero si esta propiedad está en "off", el menú no responderá a las acciones del mouse y estará de color gris oscuro

Checked {on}|off. Indicador de menú seleccionado. Definiendo esta propiedad en "on" (respectivamente en "off") pondrá (respectivamente removerá) una marca de selección junto al correspondiente menú. Esta opción puede ser usada para crear menús que indiquen el estado de una opción en particular

ForegroundColor Vector real [1,3] o caracteres. Color del frente del uimenu (color de tipografía). Es especificado en valores Rojo, Verde, Azul (RGB). Son valores reales dentro del rango [0,1]. El color puede darse como un vector real, como [R,G,B] o como una cadena donde cada valor es separado por un "|", como "R|G|B"

Label Caracteres. Esta propiedad representa el texto que aparece en el menú

Tag Caracteres. Esta propiedad es generalmente utilizada para identificar los menús. Permite darle un nombre. Principalmente usado junto a findobj()

Visible {on}|off. Define la visibilidad del uimenu. Si la propiedad está en "on" (por defecto) el uimenu es visible, pero si esta propiedad está en "off", el uimenu no aparecerá en su correspondiente figura.

Example 3. Crear menús y submenús

Algorithm 2.6 Ejemplo menús y submenús

```
// crear una figura
f=figure('position', [10 10 300 200]);
// crear un objeto en la barra de menú
m=uimenu(f,'label', 'windows');
//crear dos submenús en el menú "windows"
m1=uimenu(m,'label', 'operations');
m2=uimenu(m,'label', 'quit scilab', 'callback', "exit");
// crear dos submenús a "operations"
m11=uimenu(m1,'label', 'new window', 'callback',"xselect()");
m12=uimenu(m1,'label', 'clear window', 'callback',"clf()");
//Cerrar la figura
close(f);
```

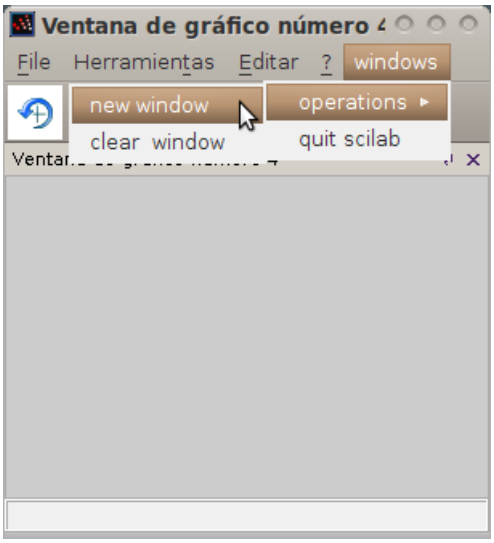


Figura 2.6: Ejemplo3: Objetos uimenu en figure

2.3. Otros conceptos importantes de Scilab

function La organización de un conjunto de líneas de código que ejecuta una acción

endfunction Finaliza la estructura de una function

global Se utiliza para declarar una variable del sistema como variable global, lo que permite ser utilizada por cualquier función que la requiere

“//” La doble diagonal invertida se utiliza para convertir una línea de código en comentario

; Al colocarlo al final de una línea de código , impide que el resultado ésta aparezca en consola

’ ’ Convierte la línea de código de instrucción a string es equivalente a “ ”

Capítulo 3

Desarrollo

3.1. Interfaz ADS_CoLiSyS

La Interfaz Gráfica ADS_CoLiSyS está compuesta por dos objetos Scilab tipo “Figure”, los cuales dan la base para la construcción de este GUI.

En ellas se adhieren los demás objetos uicontrol y uimenu, que permitirán el ingreso de los valores del sistema y nos brindarán las herramientas para el llamado de funciones y herramientas de control.

Los dos objetos figure, están definidos por las variables f , para la ventana principal y g para el área de graficación.

A continuación se presentan algunos ejemplos del código Scilab usado para la construcción de la Interfaz.

3.1.1. Construcción de la ventana Principal

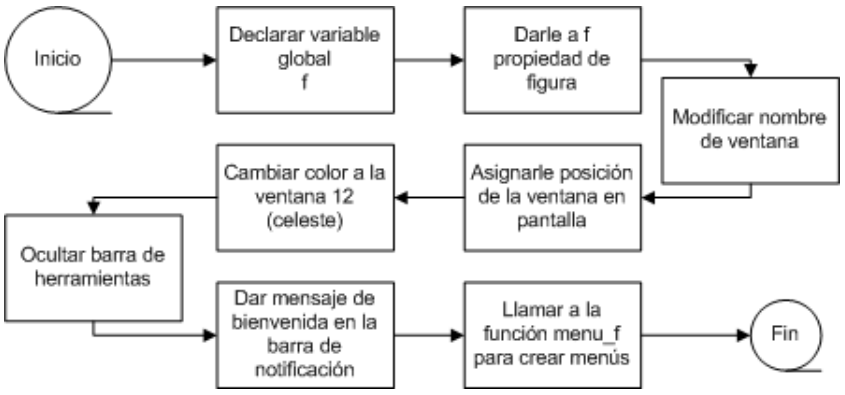


Figura 3.1: Diagrama de flujo para crear ventana principal

Algorithm 3.1 Código de Ventana principal

```
function figure_f();
  //  ------------------------------------
  //Código para generar la ventana principal
  //  ------------------------------------
  global f                                  //(1) Declarar variables global
  //Main Panel
  f=figure('Figure_name','ADS_CoLiSyS',...//(2)
  'position',[-10 0 700 200],..             //(3)
  'background',12);                         //(4)
  //'f' Extras
  toolbar(f.figure_id,'off');               // (5)
  xinfo('Welcome... Choose the type of system');//(6)
  //  Call Menus f
  menu_f();                                 //(7)
endfunction
```

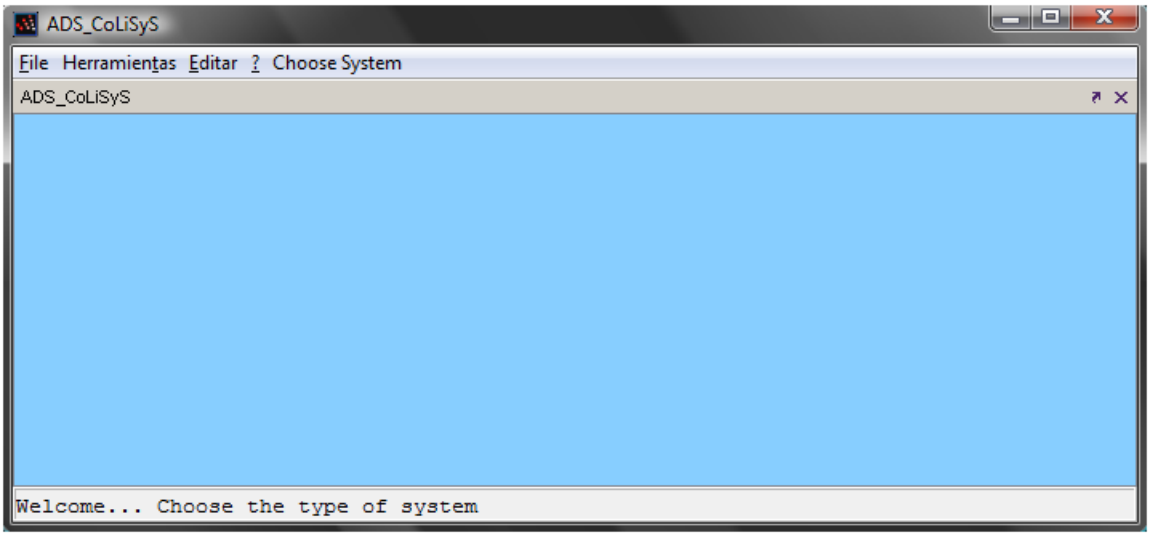


Figura 3.2: Ventana principal

3.1.2. Construcción de la ventana para Gráficas

Algorithm 3.2 Área de gráficos

```
function figure_g();
  // ------------------------------------
  //Código para generar la ventana de gráficos
  // ------------------------------------
   global g                              //(1)
  //Plot Area
  g=figure('Figure_name','Plot Area',..//(2)
  'position',[xg yg wg hg]);            //(3)
  //'g' Extras
  xinfo('Choose Plot');                 //(4)
endfunction
```

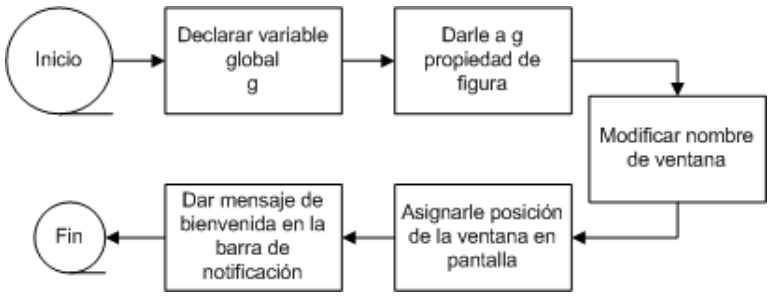


Figura 3.3: Diagrama de flujo para crear ventana de gráficas

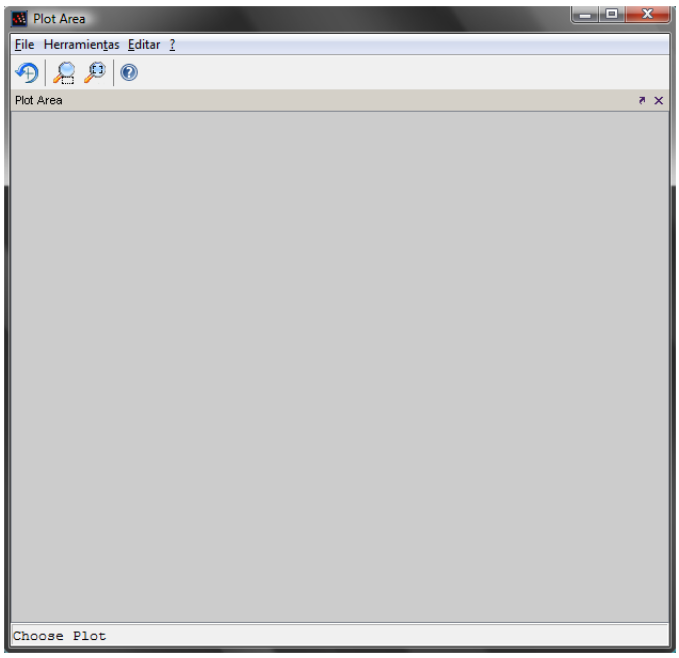


Figura 3.4: Ventana para Gráficas

3.1.3. Usando objetos uicontrol

El uso de los objetos Scilab uicontrol en esta interfaz fue fundamental. Éstos se utilizaron en casos como:

- Representación del diagrama a bloques
  - Text
  - Frame
- Elección de entradas y salidas
  - Popupmenu
- Llamado de funciones
  - Pushbutton
  - Checkbox

A continuación se presentan el código fuente para algunos de los objetos uicontrol utilizados, únicamente como referencia de éstos.

3.1.3.1. Creación del diagrama a bloques de la planta del sistema

Algorithm 3.3 Diagrama a bloques del sistema

```
function block_xxtime()
  // ------------------------------------
  //Código Scilab para construir el diagrama a bloques
  // ------------------------------------
  global numtf dentf modtf newtf Intput Output titlE modTime modIntput
  //-----Panels = Header
  titlE=uicontrol(f,'style','text',..
    'position',[250 205 200 22],...
    'background',[0.5294118 0.8078431 1],..
    'HorizontAlalignment','center',...
    'ForegroundColor',[0 0 0],..
    'fontsize',13);
  // Blocks
  line1=uicontrol(f,'style','frame',..
    'position',[70 160 560 1],...
    'background',[0 0 0],..
    'relief','groove');
  Intput=uicontrol(f,'style','popupmenu',..
    'string','Step|Impulse|Pulse|Square|Random',..
    'position',[20 150 60 23],..
    'background',[0.5294118 0.8078431 1],..
    'Foregroundcolor',[1 1 1],..
    'HorizontAlalignment','center',..
    'Verticalalignment','top',..
    'relief','groove',.. 'fontsize',11);
  modTime=uicontrol(f,'style','pushbutton',..
    'string','â†»',..
    'position',[80 162 11 11],...
    'background',[0 0.8431373 0],..
    'foreground',[1 1 1]);
modIntput=uicontrol(f,'style','pushbutton',..
    'string','â†»',..
    'position',[80 149 11 11],...
    'background',[0 0.5 0.5],..
    'foreground',[1 1 1]);
```

Algorithm 3.4 Diagrama a bloques del sistema (Cont.)

```
    numtf=uicontrol(f,'style','text',..
      'string',"",..
      'position',[290 161 150 20],..
      'background',[1 1 1],..
      'HorizontAlalignment','center',..
      'fontsize',12);
    dentf=uicontrol(f,'style','text',..
      'string',"",..
      'position',[290 140 150 20],..
      'background',[1 1 1],..
      'HorizontAlalignment','center',..
      'fontsize',12);
    modtf=uicontrol(f,'style','pushbutton',..
      'string','â†»',..
      'position',[438 181 12 12],...
      'background',[0 0.5 0.5],..
      'foreground',[1 1 1]);
    newtf=uicontrol(f,'style','pushbutton',..
      'string','â‡ª',..
      'position',[426 181 12 12],...
      'background',[1 0.8431373 0],..
      'foreground',[1 1 1]);
    Output=uicontrol(f,'style','popupmenu',..
      'string','Response|Bode|Gain|Nyquist|Black|RootLocus|Pl-Zr',..
      'position',[620 150 80 22],..
      'background',[0.5294118 0.8078431 1],..
      'foregroundcolor',[1 1 1],..
      'Verticalalignment','top',..
      'fontsize',11,..
      'relief','groove');
  endfunction
```

Numerador de retroalimentación (Text)

Dentro de los uicontrol utilizados se encuentran los de tipo text, dentro del GUI se utilizaron para representar las funciones de transferencia del sistema.

A continuación se presenta el código utilizado para construir el uicontrol que representa el numerador de la retroalimentación.

Algorithm 3.5 Uicontrol numerador de retroalimentación

```
num_fb=uicontrol(f,'style','text',.. //(1)
   'position',[275 71 150 20],..      //(2)
   'background',[1 1 1],..            //(3)
   'HorizontAlalignment','center',..  //(4)
   'fontsize',14,..                   //(5)
   'string','1',..                    //(6)
   'visible','off');                  //(7)
```

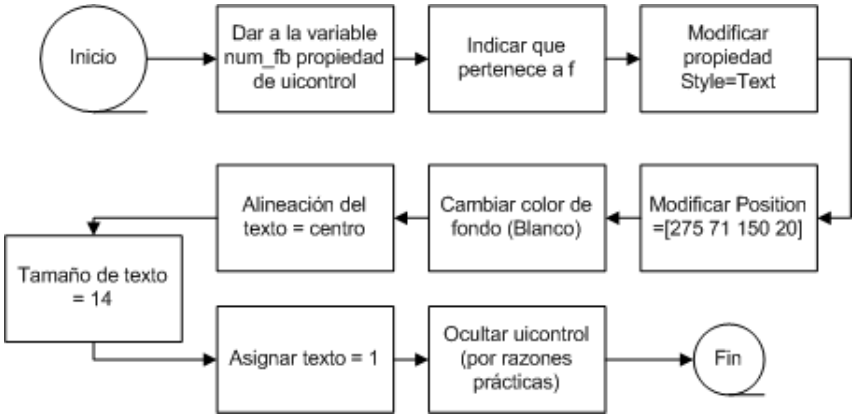


Figura 3.5: Diagrama de flujo bloque retroalimentación

Conector de Bloques (Frame)

Para unir los bloques de la función de transferencia se utilizaron objetos tipo frame, puesto que pueden ser usados para dibujar líneas.

Para ejemplificar el código de este tipo de uicontrol, se mostrará el correspondiente al frame que une a toda la planta con sus entradas, salidas e incluso con sus controladores.

Algorithm 3.6 Conector de Bloques

```
line1=uicontrol(f,'style','frame',..  //(1)
   'position',[70 160 560 1],...      //(2)
   'background',[0 0 0],...           //(3)
   'relief','groove');                //(4)
```

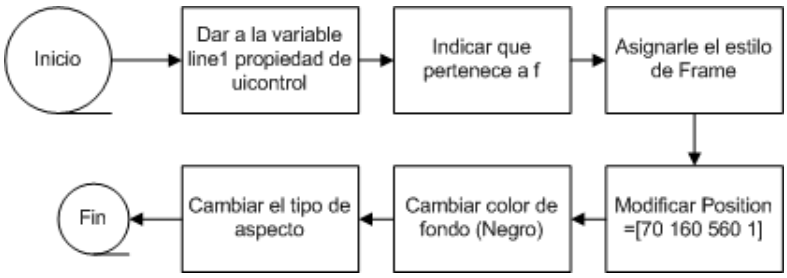


Figura 3.6: Diagrama de flujo para conector de Bloques

Elección de entradas del sistema(popupmenu)

Solo se utilizaron en dos casos, los cuales fueron para dar al usuario la posibilidad de elegir tanto entradas del sistema, como las salidas del mismo (las gráficas).

A continuación el código de Scilab utilizado para el frame de elección de entradas del sistema

Algorithm 3.7 Elección de entradas del sistema (popupmenu)

```
Intput=uicontrol(f,'style','popupmenu',..          //(1)
    'string','Step|Impulse|Pulse|Square|Random',..//(2)
    'position',[20 150 60 23],..                   //(3)
    'background',[0.5294118 0.8078431 1],..        //(4)
    'Foregroundcolor',[1 1 1],..                   //(5)
    'HorizontAlalignment','center',..              //(6)
    'Verticalalignment','top',..                   //(7)
    'relief','groove',..                           //(8)
    'fontsize',11);                                //(9)
```

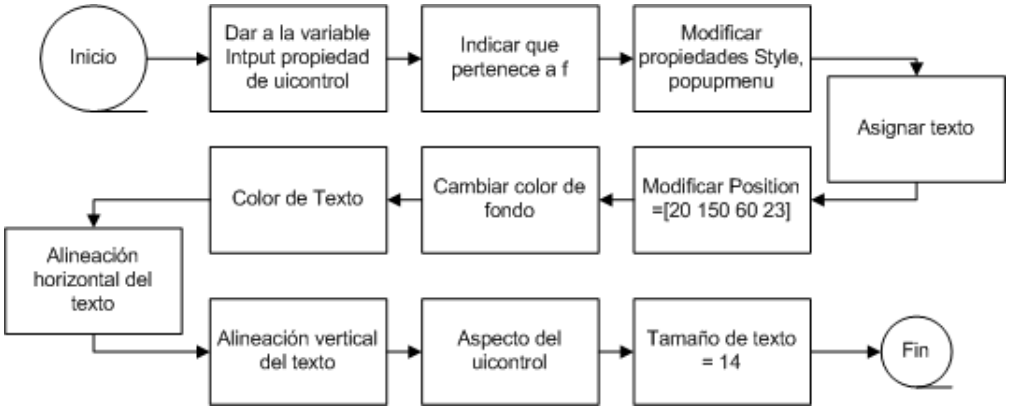


Figura 3.7: Diagrama de flujo. Entradas del sistema (Popupmenu)

Botón “Exit” (pushbutton)

Utilizados principalmente en el llamado de funciones, los pushbutton.

El siguiente es el pushbutton utilizado para llamar a la función que cierra las ventanas de la Interfaz.

Algorithm 3.8 Botón “Exit”

```
clo=uicontrol(f,..            //(1)
 'style','pushbutton',..      //(2)
 'string','Exit',..           //(3)
 'callback','closed()',..     //(4)
 'background',[0.9 0.3 0.3],..//(5)
 'ForegroundColor',[1 1 1],.. //(6)
 'fontsize',12,...            //(7)
 'position',[0 -1 30 15]);    //(8)
```

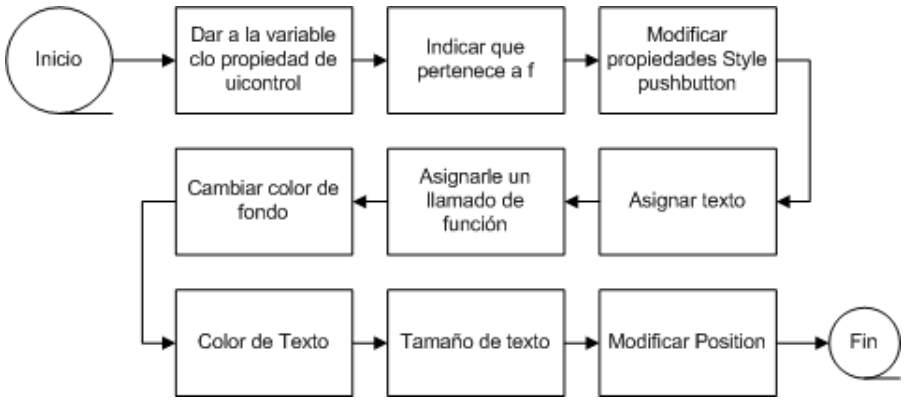


Figura 3.8: Diagrama de flujo botón “Exit”

CheckBox para aplicar retroalimentación

El único checkbox dentro de la Interfaz, es el que permite aplicar la retroalimentación al sistema, para cerrar el lazo.

Al activar un checkbox aparece en el una marca, al volver a activarlo la marca se quita, lo cual permita representar rutinas que representen la aplicación o extracción de algo.

El código Scilab utilizado en el checkbox de feedback es el que sigue.

Algorithm 3.9 CheckBox para aplicar retroalimentación

```
enable_fb=uicontrol(f,...                    //(1)
    'style','checkbox',..                    //(2)
    'position', [590 40 110 20],...          //(3)
    'background',[0.5294118 0.8078431 1],..//(4)
    'string','FeedBack',..                   //(5)
    'HorizontAlalignment','center',..        //(6)
    'ForegroundColor',[1 1 1],..             //(7)
    'fontsize',15);                          //(10)
```

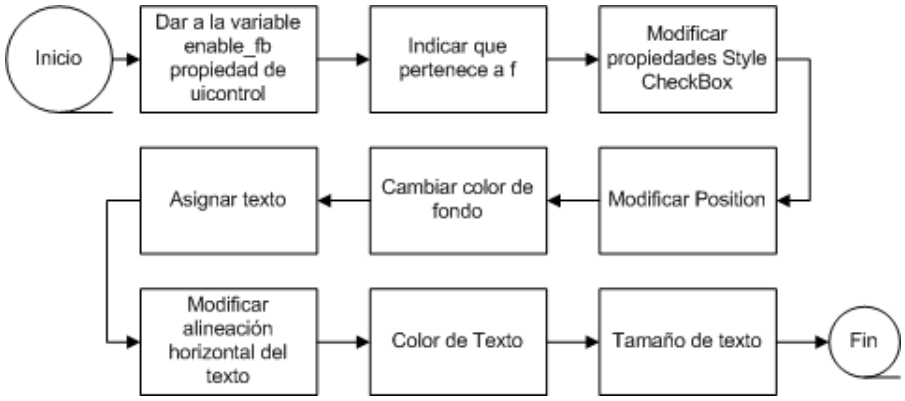


Figura 3.9: Diagrama de flujo CheckBox

3.1.4. Creación del archivo cargador de funciones

Para ejecutar las funciones de nuestra GUI, debemos construir un archivo de nombre "Loader.sce", el cual es el encargado de cargar en Scilab todas la funciones que el usuario a estructurado, declarar las variables globales que se necesitan, para que las demás funciones las puedan usar, y por último llamar a las funciones que necesiten ser ejecutadas inmediatamente.

3.2. Resultados Interfaz Gráfica ADS_CoLiSyS

Análisis, Diseño, Simulación y Control de Sistemas Lineales (por sus siglas en Inglés ADS_CoLiSyS) es un Interfaz Gráfica de Usuario, creada en el software matemático Scilab.

La interfaz se encuentra creada sobre dos objetos Scilab tipo “figure”, bajo los valores de variables f y g para la ventana principal como para el área de gráficas, respectivamente. Además se compone de diversos objetos tipo uicontrol que permiten el ingreso de valores de las funciones de transferencia, controladores, etc, la representación de éstos en el diagrama a bloques y la ejecución de algunas acciones. Por último, pero no menos importantes, los objetos de tipo uimenu, permiten agregar a las ventanas menús, que darán al usuario opciones de control, tipo de sistema, graficación y otros.

3.2.1. Ventana principal

La ventana principal de la interfaz ADS_CoLiSyS se muestra en la Figura3.10, en las secciones siguientes s detallan cada una de las partes que la componen.

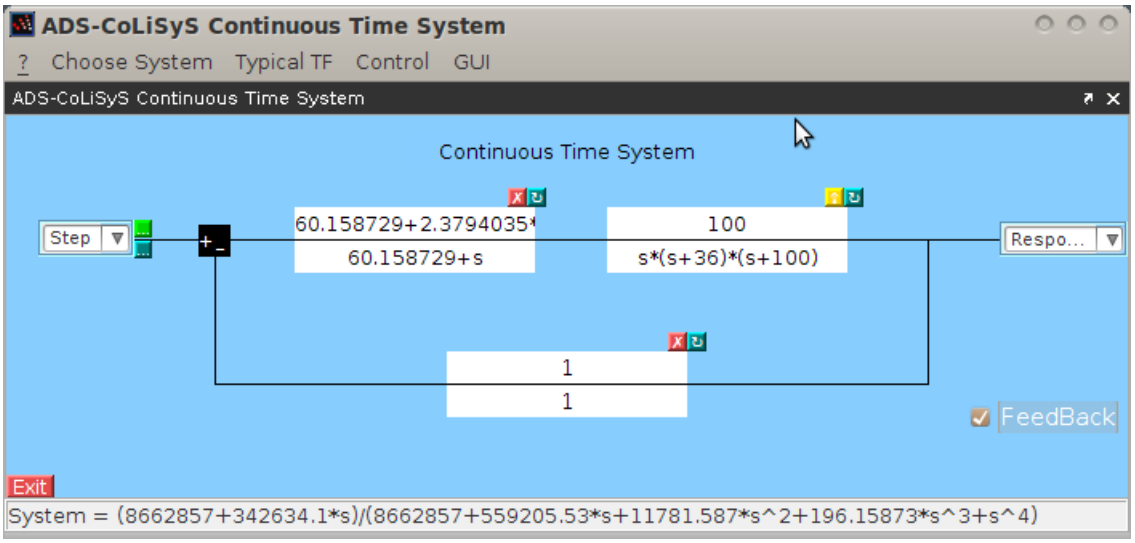


Figura 3.10: Ventana Principal

Barra de Menús

Compuesta de diferentes menús, que proporcionan al usuario elementos de control, graficación, entre otros.

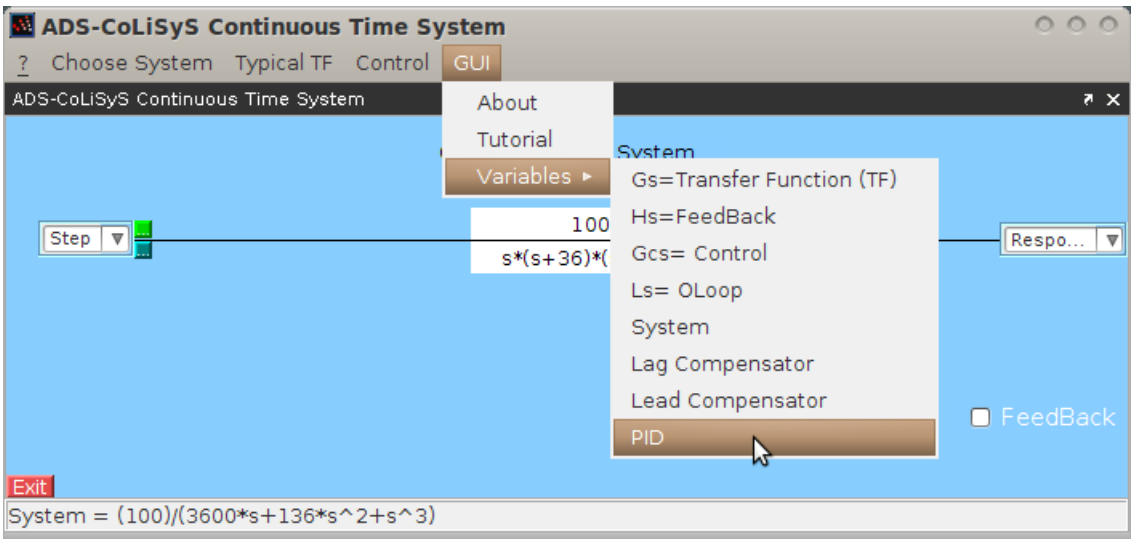


Figura 3.11: Barra de Herramientas

? Es un menú creado por defecto por Scilab al crear la ventana. En él se puede encontrar ayuda del software.

Ayuda de Scilab Al activar este submenú llamamos a la ayuda del software

Acerca de Scilab... Nos presenta datos como versión y créditos de Scilab

Choose System Permite cambiar el tipo de sistema que se trabajará en la GUI

Transfer Function Para elegir un sistema de tipo función de transferencia

Continuous Time La función de transferencia estará en tiempo continuo

Discrete Time La función de transferencia estará en tiempo discreto

State Space Se elige un sistema en Espacio de Estados

Continuous Time El sistema en espacio de estados estará en tiempo continuo

Discrete Time El sistema en espacio de estados estará en tiempo discreto

Typical TF Se encuentran en este menú, diversas funciones de transferencia predefinas, las cuales son sistemas típicos para análisis de control

Simple Filters Elegir los funciones de transferencia de Filtros Simples

Single Pole L-P Para el análisis de un filtro Pasa-Baja de un Polo

Two Pole L-P Para ingresar los parámetros de un filtro Pasa-Bajas de Dos Polos

Two Pole Notch Para un Filtro Simple Notch de Dos Polos

Compensators Con submenús para el ingreso de funciones de transferencia de compensadores, controladores

PI Para un controlador PI

Lead Controlador de adelanto de fase

Control En este menú se encuentran los diferentes controladores con los que se puede manipular la planta del sistema

Lag Compensator Para un controlador de Atraso de Fase

Desing Mediante este submenú se pedirán valores para el diseño automático de un compensador de atraso de fase

Insert Si el usuario ya tiene su compensador diseñado, puede ingresarlo mediante este submenú

Lead Compensator Para un controlador de Adelanto de Fase

Desing Para ingresar valores para el diseño automático de un Compensador de Adelanto de Fase

Insert Pide numerador y denominador de un Compensador de Adelanto de Fase previamente calculado

PID Para el ingreso o diseño de un controlador PID (Proporcional Integral Derivativo)

Insert Ingresar los valores de un controlador PID previamente diseñado

Sintonize Con el se llama a una herramienta con la que el usuario encontrará parámetros para el diseño de controladores P, PI o PID mediante el método de diseño por Oscilación Sostenida

Help Para buscar ayuda acerca de ADS_CoLiSyS

About Presenta una ventana emergente con información de créditos y versiones de la interfaz

Tutorial Abrirá este archivo, que funcionará como ayuda

Variables En él se encuentran las diferentes variables que componen el sistema en análisis, y su representación en la consola de Scilab

Gs= Transfer Function (TF) Es la planta del sistema

Hs= FeedBack Presenta la Función de Transferencia correspondiente a la Retroalimentación del sistema

Gcs= Control Es la función de transferencia del control aplicado al sistema

Ls= OLoop Sistema a Lazo Abierto

System Es el sistema actual analizado, incluyendo planta, control y retroalimentación (cuando estos últimos dos estén aplicados).

Lag Compensator Si se ya se ha aplicado, nos presenta la ecuación del controlador de Atraso de Fase

Lead Compensator Si ya se aplicó, nos presenta su ecuación

PID Presenta al controlador PID aplicado

Diagrama a Bloques

Es la representación gráfica del sistema actual. Tal como se muestra en la figura 3.12

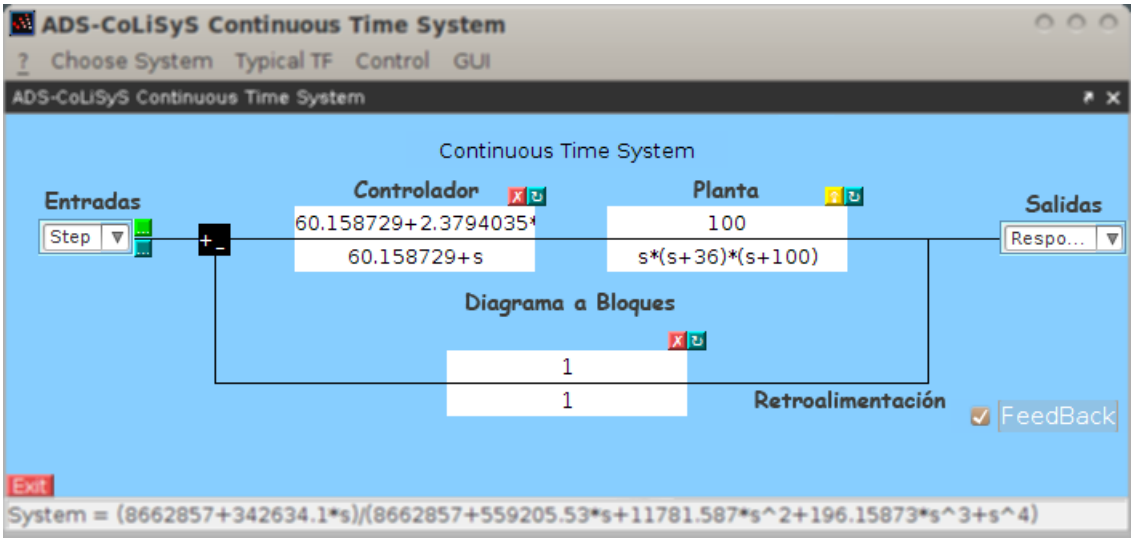


Figura 3.12: Diagrama a Bloques

Planta Es la función de transferencia que representa al sistema original.

Controlador Es la representación gráfica del controlador, aplicado a la planta.

FeedBack O retroalimentación. Compuesta de la retroalimentación del sistema de la salida del mismo hacia el sumador

Sumador Representa el bloque de unión entre el sistema a lazo abierto y su retroalimentación

Entradas Diferentes entradas que se pueden aplicar al sistema para lograr su respuesta en frecuencia

Salidas Las diferentes gráficas que se pueden obtener del sistema

Botonera Son botones que permiten el ingresar, modificar o eliminar un bloque del sistema, según sea el caso

Entradas Por medio de este objeto Scilab tipo Popupmenu, el usuario pueda elegir que tipo de entrada aplicar al Sistema en análisis (Figura 3.13).

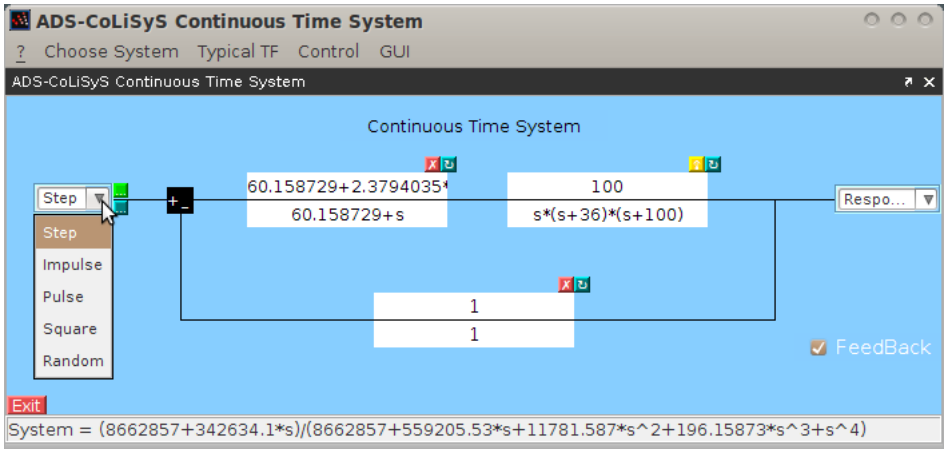


Figura 3.13: Entradas

Step Es una función de tipo escalón. Con sus parámetros:

Initial Time Tiempo en el cuál iniciará el escalón

Initial Value Valor inicial del Escalón (Amplitud)

Final Value Valor final del Escalón (Amplitud)

Impulse Función para entrada de un Impulso

Initial Value Valor del tiempo en el que inicia impulso

Final Value En que tiempo termina el impulso

Pulse Genera un pulso cuadrado, del origen hacia un valor de amplitud

Amplitude Para la amplitud de la señal

Period(secs) Periodo en segundos

Pulse Width(% of period) Porcentaje de la señal la cual no estará con valor 0

Square Señal de tipo Onda Cuadrada

Amplitude Amplitud de la onda

Period(secs) Periodo de la señal en segundos

Random Es una señal aleatoria una especie de ruido

Type

Uniform Los valores de la señal están distribuidos en un intervalo (0,1)

Normal Genera una señal con media igual a 0 y varianza 1

Salidas Al igual que las entradas, este es un objeto Scilab tipo popupmenu, en el cual se encuentran las diferentes gráficas que pueden ser obtenidas del sistema en análisis (Figura 3.14)

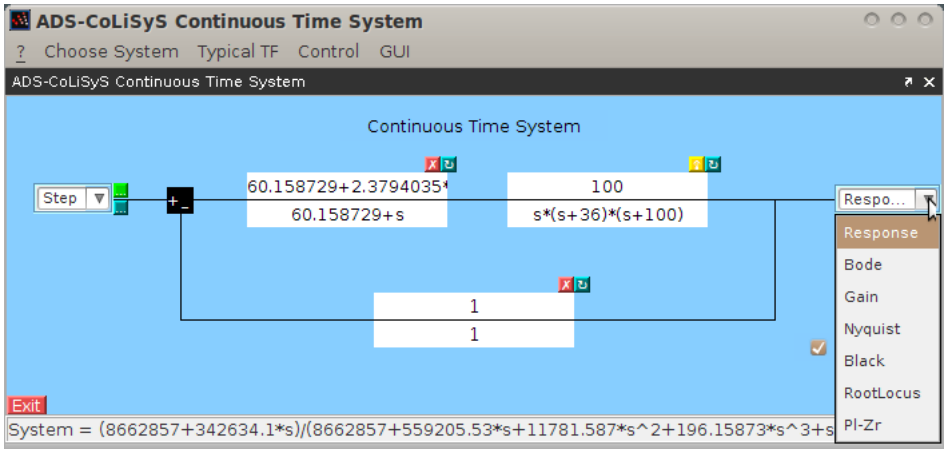


Figura 3.14: Salidas

Response Gráfica de la respuesta a la entrada aplicada

Bode Gráficas de la respuesta en frecuencia($^{rad}/_{seg}$) de la Magnitud(dB) y Fase(grados) del sistema

Gain Semejante a Bode, pero solo gráfica la magnitud(dB)

Nyquist La gráfica de la parte Imaginaria contra la parte Real de la respuesta en frecuencia del sistema

Black El diagrama Black (Nichols' Chart)

RootLocus Diagrama del lugar de las raíces

Pl-Zr Gráfica de polos-ceros del sistema lineal

Botonera Objetos Scilab tipo Pushbutton, que al ser activados permiten al usuario modificar el sistema o las entradas al mismo, dando opciones de ingresar, modificar, eliminar bloques del sistema. Se encuentran bajo una nomenclatura de símbolos y colores, los cuales corresponden a el tipo de acción que realizan.

Figura 3.15: Botonera

Botón Amarillo Es el reinicio del sistema, al ingresar una nueva función de transferencia

Botón Verde Permite modificar, el bloque actual para aplicarlo al sistema ya creado, o los parámetros de las entradas

Botón Rojo Elimina el bloque de la función de transferencia

Botón Rojo "Exit" Para salir del GUI, de la ventana principal o del área de graficado

Barra de Notificación

Es la parte inferior de la ventana. En esta se representará el sistema actual en análisis. En un formato "Numerador"+"/"+"Denominador".

3.2.2. Área de Gráficos "g"

"g" es la variable global bajo la cual se crea la ventana secundaria de ADS_CoLiSyS, la cual es el área de graficado de la salida del sistema.

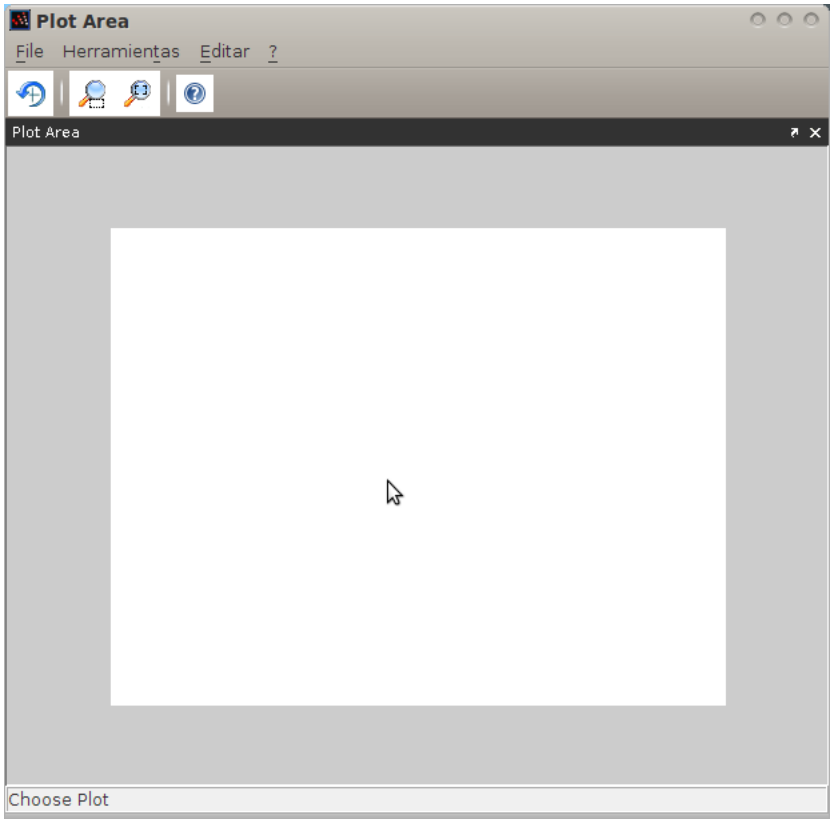


Figura 3.16: Área de Gráficos

Barra de Herramientas

La barra de herramientas de "g" no es siempre la misma, puesto que se modifica según la gráfica que se encuentre activa. Los diferentes menús que pueden ser visualizados en todos los casos se describen a continuación.

Archivo (File) Es un menú creado por Scilab para manipular la ventana

Nueva Figura... Genera una nueva ventana vacía

Cargar... Carga un archivo .scg que corresponde a una ventana guarda.

Guardar... Guarda la ventana actual con todas sus propiedades y objetos

Exportar a... Exportamos el área de trabajo de la ventana a un archivo de tipo imagen, algunas de las extensiones disponibles son .png, .ppm, .fig, .pdf, entre otras

Copiar al Portapapeles Copia el área de trabajo de la ventana activa al portapapeles

Configuración de Página... Regresa un cuadro de configuración de página para impresión

Imprimir Imprime el área de trabajo

Cerrar(I) Cierra la ventana

Herramientas Por defecto creado en la ventana por Scilab.

Mostrar/Ocultar Barra de Herramientas Muestra u oculta la barra de herramientas "extra"

Acercar Acerca la imagen del área de trabajo

Alejar Aleja la imagen del área de trabajo

Rotación 2D/3D Rota el área de trabajo en 2D O 3D según lo requiera el usuario

Editar Con opciones de edición de la ventana

Seleccionar como figura actual Selecciona la ventana actual como la última ventana creada (similar al comando scf)

Redibujar figura Redibuja la figura actual con sus mismos parámetros y objetos

Limpiar figura(c) Elimina todos los objetos agregados a la ventana, y la regresa a su estado predeterminado

Propiedades de la figura Para modificar las propiedades de la figura (posición, color, etc.)

Propiedades de los ejes Para configurar el área de trabajo de la ventana, los ejes de las gráficas, entre otros.

Empezar selector de identidades

Parar selector de identidades

? Es un menú creado por defecto por Scilab al crear la ventana. En él se puede encontrar ayuda del software.

Ayuda de Scilab Al activar este submenú llamamos a la ayuda del software

Acerca de Scilab... Nos presenta la datos como versión y créditos de Scilab

Edit (Gráfica Response) Menú de ADS_CoLiSyS para editar los parámetros de las entradas del sistema y de graficación.

Time Modifica parámetros del tiempo en el que se analiza el sistema

Step Para modificar los valores del escalón

Impulse Cambia los parámetros de la función impulso

Pulse Modifica la entrada pulso

Square Para modificar los valores del onda cuadrada

Random Cambia el tipo de entrada random

Edit (Gráfica Bode, Gain, Nyquist, Black)

Logspace Permite cambiar los parámetros del espacio logarítmico del diagrama.

Margin (Bode, Gain, Nyquist)

Phase Muestra los márgenes de fase del sistema

Gain Muestra los márgenes de ganancia del sistema

All Muestra los márgenes de fase y ganancia del sistema

To Plot Muestra en la gráfica la ubicación de los márgenes de fase y ganancia, según la gráfica que corresponda

m_circle (Nyquist)

user Permite al usuario ingresar un vector de ganancias para generar la representación de éstas en el diagrama de Nyquist

chart (Black) Agrega al diagrama de Black Nichol's el complemento Chart

Change Plot Da la lista de gráficas disponibles para cambiar a una de ellas.

Barra de Herramientas “Extra”

Es un conjunto de íconos que nos permiten hacer zoom o rotar la gráfica.

Rotar Para rotar el área de trabajo ya sea en 2D o 3D

Acercar Área Hace Zoom a la porción del área de trabajo que se seleccione

Vista Original Elimina el zoom regresando a su tamaño original

Navegador de Ayuda Abre el navegador de ayuda de Scilab

Área de Gráficos

Es el área de la ventana donde se dibujarán las diversas gráficas del sistema.

Barra de Notificación

Es la parte inferior de la ventana, se actualizará con el nombre de la gráfica representada en ese momento. Si la leyenda del área de notificación es “Choose Plot” indica al usuario que:

- No ha elegido gráfica alguna
- Se produjo un error en la gráfica actual y debe elegir otra

Si la gráfica actual es “Response”, el área de notificación también mostrará el tipo de entrada aplicada, en un formato: “Response Plot (-Entrada aplicada-)”.

3.3. Aplicaciones

Para describir algunas de las aplicaciones que la Interfaz Gráfica ADS_CoLiSyS tiene, utilizaremos algunos casos, los cuales pueden considerarse como típicos en una clase referente a teoría de control.

3.3.1. Gráfica de Bode

La Gráfica de Bode cosiste en dos gráficas: una de la amplitud logarítmica contra la frecuencia espaciada de forma logarítmica. La otra gráfica el ángulo de fase contra la frecuencia logarítmica:

Entonces dado:

$$G(j\omega) = \frac{num(j\omega)}{den(j\omega)} \tag{3.1}$$

se tiene:

$$|G(\omega)| = 20 * \log_{10}(G(j\omega)) = 20 * \log_{10}\left(\sqrt{Re^2 + Im^2}\right) \tag{3.2}$$

$$\Phi(\omega) = \tan^{-1}(G(s)) = \tan^{-1}\left(\frac{Im}{Re}\right) \tag{3.3}$$

donde:

$|G(\omega)|$ =la magnitud de $G(j\omega)$en dB

$\Phi(\omega)$ =es la fase de $G(\omega)$en grados

El diagrama de Bode de una función de transferencia puede ser obtenido por cualesquiera de los siguientes métodos:

Método 1: Separar la parte real y la parte imaginaria de una función de transferencia.

Para obtener las gráficas de Magnitud y de fase de $G(j\omega)$deben de seguir los siguientes pasos:

1. Dada una función de transferencia $G(j\omega)$,ésta se debe de manipularse de forma algebraica para separa la parte real e imaginaria.
2. Calcular la Longitud Logarítmica y el ángulo de fase variando la frecuencia de forma Logarítmica.

Example 4. Dado:

$$G(j\omega) = \frac{5}{\omega j + 2} \tag{3.4}$$

Dibujar los trazos de Bode de magnitud y de fase para $\omega = [0.01, 0.11, 1, 10, 100]$ ...

Solución: separamos las parte de real e imaginaria de $G(j\omega)$

$$G(j\omega) = \frac{5}{j\omega + 2} * \frac{2 - j\omega}{2 - j\omega} \tag{3.5}$$

$$G(j\omega) = \frac{10 - 5\omega j}{4 + \omega^2} \tag{3.6}$$

$$G(j\omega) = Re + Im = \frac{10}{4 + \omega^2} - \frac{5\omega}{4 + \omega^2} j \tag{3.7}$$

Entonces la Magnitud y la fase pueden determinarse como:

$$|G(j\omega)| = 20 * \log_{10} \sqrt{\left(\frac{10}{4+\omega^2}\right)^2 + \left(\frac{5\omega}{4+\omega^2}\right)^2}$$
$$\Phi(\omega) = \tan^{-1}\left(\frac{-\frac{5\omega}{4+\omega^2}}{\frac{10}{4+\omega^2}}\right) = \tan^{-1}\left(\frac{-\omega}{2}\right)$$

Solución utilizando Scilab La Gráfica de magnitud y la fase pueden trazarse con SciLab

Algorithm 3.10 Código Scilab para el ejemplo 4

```
clc;
clear all;
w=[0.01 0.1 1 10 100]; //dado
w=logspace(-3,4,100);
//Definimos funciones |G(w)| y Phi (w)
deff('[y]=mag(w)','y=20*log10(sqrt((10/(4+w^2))^2+((5*w)/(4+w^2))^2))');
deff('[z]=phi(w)','z=atan(-w,2)');
//Evaluando en w
magn=feval(w,mag);
phase=feval(w,phi);
phase=(phase*180/%pi); //convertimos radianes a grados
figure(1);
clf();
subplot(211);
plot2d('ln',w,magn);
xtitle('Magnitud','W hertz','Mag dB');
a=get('current_axes');
a.grid=[4,4];
subplot(212);
plot2d('ln',w,phase);
xlabel(' Frequency rad/seg'), ylabel('Phase (grade)');
a=gca();
a.grid=[4,4];
```

La gráfica solución de este problema se muestra en la Figura 3.18b, la cual es la misma que la obtenida utilizando ADS_CoLiSyS

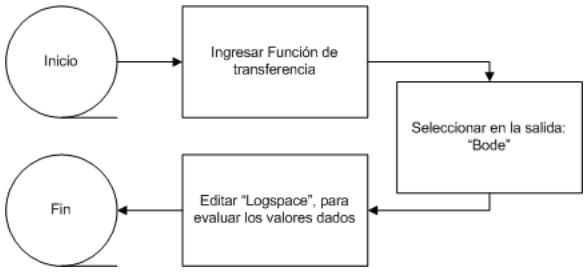
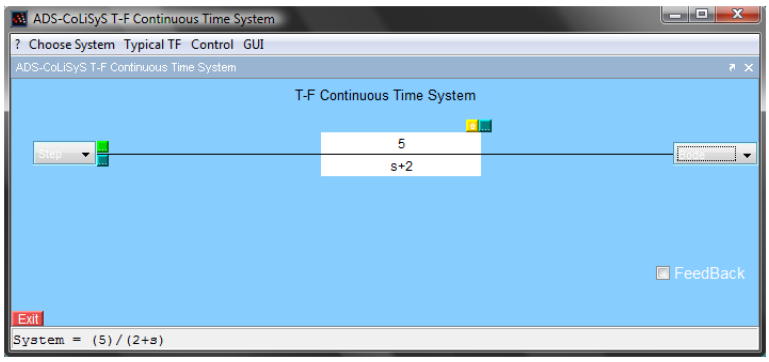
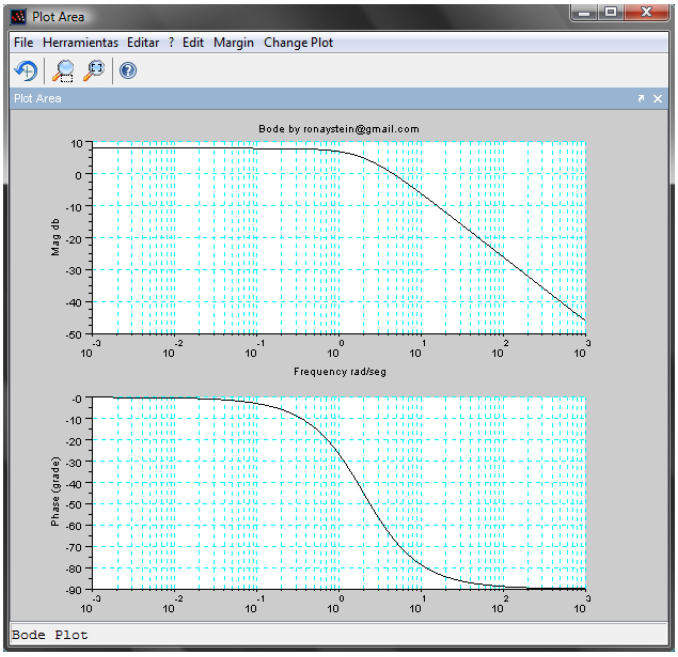


Figura 3.17: Gráfica de Bode Diagrama de Flujo ADS_CoLiSyS para el ejemplo4



(a) Ventana Principal



(b) Área de Gráficos

Figura 3.18: Gráfica de Bode utilizando ADS_CoLiSyS, ejemplo 4

Solución utilizando ADS_CoLiSyS

Método 2: Por Fórmula General

Dada la función de tranferencia de la forma:

$$G(j\omega) = \frac{K\prod_{i=1}^{l}(1+j\omega\tau_i)}{(j\omega)^N\prod_{q=1}^{Q}(1+j\omega\tau_q)\prod_{k}^{K}(1+2\zeta_K/\omega_{nk}+(j\omega/\omega_n)^2)} \tag{3.8}$$

La Ecuación 3.8, tiene una ganancia de K, i ceros, N polos en el origen, q polos en el eje real

y k polos complejos conjugados. Dada su estructura seria difícil separar la parte real y la parte imaginaria de la ecuación.

Las ecuaciones de magnitud y de fase quedan definidas como:

$$G(j\omega) = 20 * \log_{10}\left(\frac{K\prod_{i=1}^{l}(1+j\omega\tau_i)}{(j\omega)^N\prod_{q=1}^{Q}(1+j\omega\tau_q)\prod_{k}^{K}(1+2\zeta_K/\omega_{nk}+(j\omega/\omega_n)^2)}\right) \tag{3.9}$$

recurriendo a las siguientes propiedades de los logaritmos (donde $\log = \log_{10}$*);*

$$\log(A*B) = \log A + \log B \tag{3.10}$$

$$\log(\frac{A}{B}) = \log A - \log B \tag{3.11}$$

$$\log(A^N) = N\log A \tag{3.12}$$

se tiene que, mediante las propiedades dadas en las ecuaciones 3.10 y 3.11, la ecuación 3.9, se convierte en:

$$|G(\omega)| = 20\log(K) + 20\log\sum_{i=1}^{l}(\sqrt{1+(\omega\tau_i)^2}) - 20\log(\sqrt{(\omega^2)^N}) - 20\log\sum_{q=1}^{Q}(\sqrt{1+\left(\omega\tau_q\right)}) - 20\log\sum_{k=1}^{K}\left(\sqrt{(1-(\omega/\omega_n)^2)^2+(2\zeta_K/\omega_n)^2}\right) \tag{3.13}$$

Aplicando la propiedad de la ecuación 3.12 se tiene:

$$|G(\omega)| = 20\log(K) + 10\log\sum_{i=1}^{l}(1+(\omega\tau_i)^2) - 20(N)\log(\omega) - 10\log\sum_{q=1}^{Q}(1+\left(\omega\tau_q\right)) - 10\log\sum_{k=1}^{K}\left(\left(1-(\omega/\omega_n)^2\right)^2+(2\zeta_K/\omega_n)^2\right) \tag{3.14}$$

y la fase (en grados) queda definida como:

$$\Phi(\omega) = a\tan(G(\omega)) = a\tan(K) + \sum_{i=1}^{I} a\tan(1+j\omega\tau_i) + a\tan\left(\frac{1}{j\omega^N}\right) + \sum_{q=1}^{Q}\tan\left(\frac{1}{(1+j\omega\tau_q)}\right) + \sum_{k=1}^{K} a\tan\left(\frac{1}{\left(1+2\zeta_k/\omega_n-(\omega/\omega_n)^2\right)}\right)$$

$$\Phi(\omega) = 0 + \sum_{i=1}^{I} a\tan\left(\frac{\omega\tau_i}{1}\right) + (N)\,90^\circ + \sum_{q=1}^{Q} \tan\left(\frac{\omega\tau_q}{1}\right) + \sum_{k=1}^{K} a\tan\left(\frac{\left(2\zeta_k/\omega_n\right)}{\left(1-\left(\omega/\omega_n\right)^2\right)}\right) \quad (3.15)$$

Example 5. Dada la función de transferencia:

$$G(s) = \frac{5(s+2)}{s^2(s+5)(s^2+2s+6)}$$

dibujar sus trazos de Bode de magnitud y fase (Graficar con Scilab).

Solución: Primero, sustituyendo $s = j\omega$, se tiene:

$$G(j\omega) = \frac{5(j\omega+2)}{(j\omega)^2(j\omega+5)\left((j\omega)^2+2(j\omega)+6\right)}$$

Analizando $G(j\omega)$ se tiene una ganancia K=5, un cero, dos polos en el origen y un polo complejo conjugado. Por lo tanto, aplicando la ecuación 3.13 se tiene:

$$|G(\omega)| = 20\log(5) + 10\log\left(4+\omega^2\right) - 20(2)\log(\omega) - 10\log\left((50)^2+\omega^2\right) - 10\log\left(\left(36-\omega^2\right)^2+4\omega^2\right)$$

Ahora aplicamos la ecuación 3.14

$$\Phi(\omega) = 0 + a\tan\left(\tfrac{\omega}{2}\right) - 90^\circ(2) - a\tan\left(\tfrac{\omega}{50}\right) - a\tan\left(\frac{2\omega}{\left(36-\omega^2\right)}\right)$$

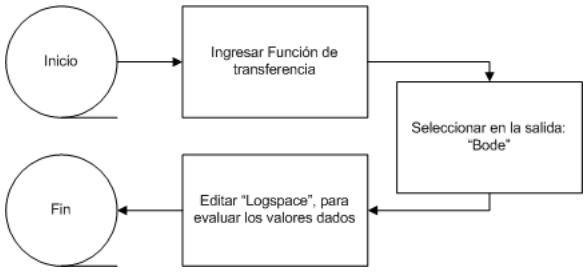
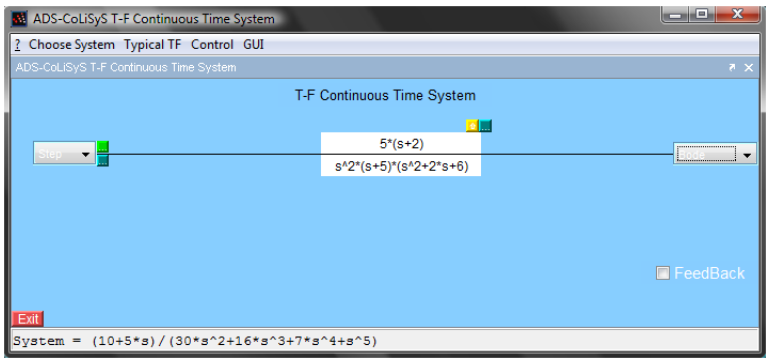
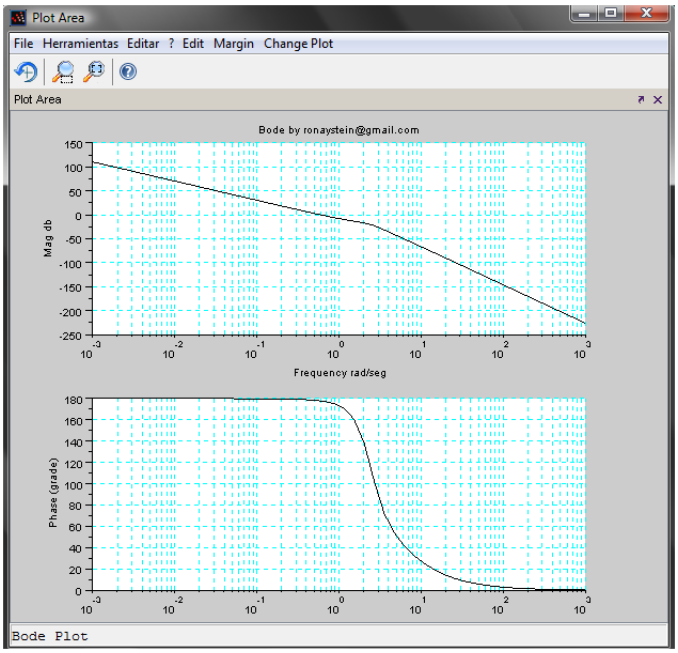


Figura 3.19: Gráfica de Bode Diagrama de Flujo ADS_CoLiSyS para el ejemplo5



(a) Ventana Principal



(b) Área de Gráficos

Figura 3.20: Gráfica de Bode utilizando ADS_CoLiSyS, ejemplo5

Solución utilizando ADS_CoLiSyS

3.3.2. Estabilidad Relativa

La estabilidad relativa de un sistema puede determinarse mediante los conceptos de margen de fase y marguen de ganancia.

El Marguen de Ganancia.

Esta definido como la magnitud del recíproco de la función de tranferencia en lazo abierto evaluado en la frecuencia $\omega_\pi$en el cual el ángulo de fase es $\pi - 180°$ (Disterfano et al.1990)

$$GainMargin = \frac{1}{|GH(\omega)|} \tag{3.16}$$

Donde $\arg GH(\omega_\pi) = -180° = -\pi$ radianes y ω_π se le conoce la frecuencia de corte de la fase

3.3.3. Criterio de Nyquist

El criterio de estabilidad de Nyquist determina la estabilidad de un sistema en lazo cerrado a partir de la respuesta en frecuencia en lazo abierto y los polos en lazo cerrado.

Para determinar la estabilidad de un sistema de control , consideraremos la siguiente ecuación característica:

$$F(s) = 1 + G(s)H(s) = 1 + L(s) = \frac{K\prod_{i=1}^{n}(s+z_i)}{\prod_{k=1}^{n}(s+p_k)} \tag{3.17}$$

Para que un sistema sea estable, todos las raíces de $F(s)$ deben permanecer en le semiplano izquierdo.

El criterio de Nyquist relaciona la respuesta en frecuencia en lazo abierto $G(j\omega)H(j\omega)$ con el número de ceros y polos de $1 + G(s)H(s)$ que se encuentran en el semiplano derecho del plano s. Este criterio permite determinar gráficamente la estabilidad absoluta del sistema en lazo cerrado a partir de las curvas de la respuesta en frecuencia en lazo abierto sin que sea necesario determinar los polos en lazo cerrado.

Example 6. Considera la función de transferencia en lazo abierto

$$G(s) = \frac{0.04798*}{z^2 - 1.81*}$$

Obtenga su diagrama de Nyquist y mostrar sus márgenes de fase y ganancia.

Algorithm 3.11 Obtener diagrama de Nyquist

```
clc;
clear all;
//Definimos funciones
z=poly(0,'z');
Gz=syslin(0.1,0.04798*z+0.0464,z^2-1.81*z+0.9048);
figure(1);
clf();
nyquist(Gs,0.01,100)
xtitle('Gráfica de Nyquist');
a=get('current_axes');
a.grid=[4,4];
[p,fr]=p_margin(Gs)
[g,fr]=g_margin(Gs)
show_margins(Gs)
```

Nyquist en ADS_CoLiSyS

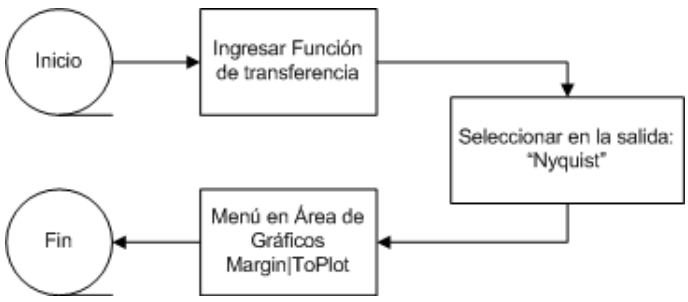
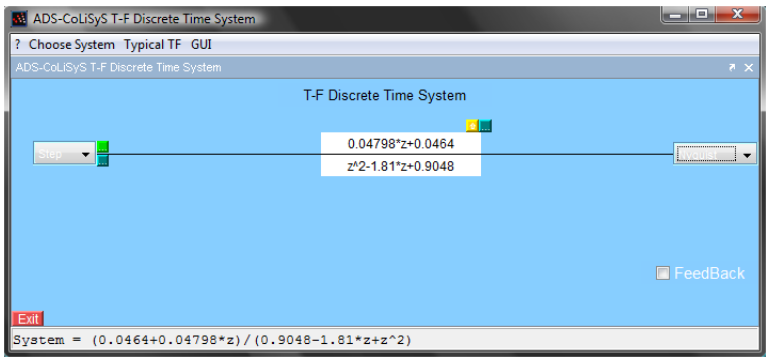
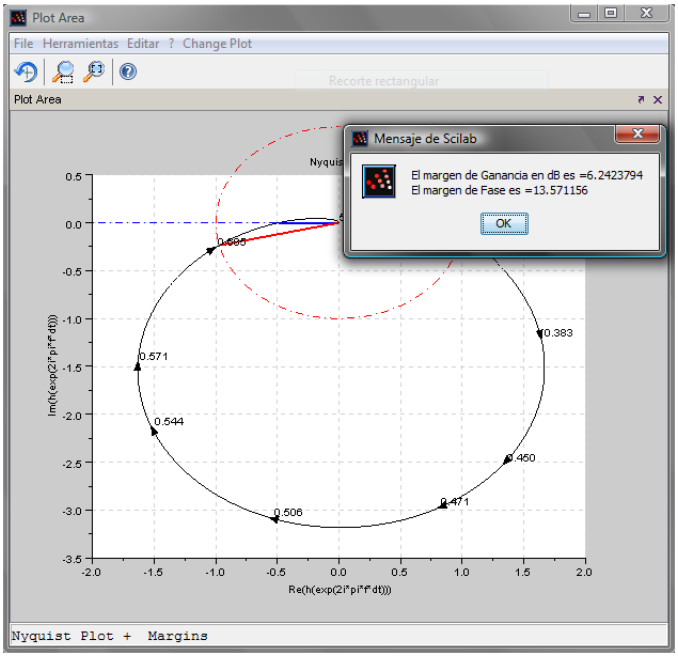


Figura 3.21: Diagrama de flujo para Gráfica de Nyquist



(a) Ventana Principal



(b) Área de Gráficos

Figura 3.22: Gráfica de Nyquist utilizando ADS_CoLiSyS para el ejemplo 6

3.3.4. Compensador de Adelanto

Respuesta en frecuencia de un compensador de adelanto

Un compensador de adelanto esta caracterizado por la siguiente ecuación

$$G_c(s) = \frac{1}{\alpha}\frac{s+\frac{1}{T}}{s+\frac{1}{\alpha T}} \tag{3.18}$$

para varios valores de α, $\alpha < 1$. Es de notar que los picos de la curva de fase varian en un

máximo ángulo y en la frecuencia a la cual ocurre el máximo.

Con el fin de diseñar un compensador de adelanto de fase y cambiar tanto el margen de fase y la frecuencia del margen de fase, es de mucha ayuda tener una expresión analítica para el máximo valor de la fase y la frecuencia a la cual ocurre el máximo valor de la fase. De la ecuación 3.18 el ángulo de fase de el compensador de adelanto ϕ_c, es:

$$\phi_c = tan^{-1}\omega T - tan^{-1}\omega\alpha T \tag{3.19}$$

Derivando con respecto a ω, se tiene

$$\frac{d\phi_c}{d\omega} = \frac{T}{1+(\omega\alpha T)^2}\frac{\alpha T}{1+(\omega\alpha T)^2} \tag{3.20}$$

Ajustando la Ecu. 3.20 igual a cero, se encuentra la frecuencia ω_{max} a la cual ocurre el máximo ángulo de fase ϕ_{max} , esto es:

$$\omega_{max} = \frac{1}{T\sqrt{\alpha}} \tag{3.21}$$

Sustituyendo la Ecu 3.20en la ecuación 3.18 con $s = j\omega_{max}$, se tiene:

$$G_c j\omega_{max} = \frac{1}{\alpha}\frac{j\omega_{max}+\frac{1}{T}}{j\omega_{max}+\frac{1}{alphaT}} = \frac{j\frac{1}{\sqrt{\alpha}+1}}{j\sqrt{\alpha}+1} \tag{3.22}$$

haciendo uso de $tan(\phi_1-\phi_2) = (tan\phi_1 - tan\phi_2)(1+tan\phi_1 tan\phi_2)$, el máximo ajuste del compensador es:

$$\phi_{max} = tan^-1\left(\frac{1-\alpha}{2\sqrt{\alpha}}\right) = sin^-1\left(\frac{1-\alpha}{1+\alpha}\right) \tag{3.23}$$

y la magnitud del compensador en $\omega_{máx}$ es:

$$G_c(j\omega_{máx}) = \frac{1}{\sqrt{\alpha}} \tag{3.24}$$

Diseño

1. Encontrar los requerimientos de ancho de banda en lazo cerrado para conocer el tiempo de estabilización, el tiempo pico o el tiempo del máximo rizo
2. Debido a que el compensador de adelanto tiene un efecto despreciable a bajas frecuencias, se debe de ajustar la ganancia K, del el sistema no compensado a un valor que satisfaga los requerimientos de error de estado estable.

3. Grafique los diagramas de Bode de magnitud y de fase para éste valor de ganancia y determine el margen de fase del sistema no compensado

4. Encuentre el margen de fase para cumplir los requerimientos de porcentaje de sobre tiro. Entonces evalué la contribución de la fase adicional requerida para el compensador.

5. Determine el valor de α para el compensador de adelanto requerida para cumplir los requerimientos de contribución en la fase.

6. Determine la magnitud del compensador en el punto pico de la curva de fase (en la fase máxima).

7. Determine la nueva frecuencia del margen de fase encontrado donde la magnitud de la curva sistema no compensado es la negativa del compensador de la magnitud del compensador de adelanto en el pico de la curva de fase del compensador

8. Diseñe el compensador de adelanto. Encuentre T y las frecuencias de corte.

9. Encuentre la ganancia del compensador $Kc = K/\alpha$

10. Verifique que el ancho de banda para asegurarse que se cumplieron los requerimientos del paso 1

11. Simular para asegurarse que se cumplieron todos los requerimientos

12. Rediseñar si es necesario en caso de que no se cumplieran los requerimientos.

Example 7. Para el sistema de control de posición mostrado en la Fig.3.23 , diseñar un compensador de adelanto de fase, sujeto a: 20 % de sobretiro, $Ts = 0.25s$, $K_v = 40s^{-1}$

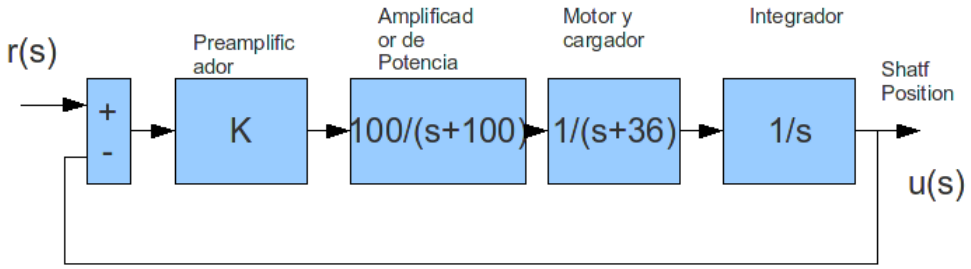


Figura 3.23: Sistema del ejemplo 7[**?**]

1. Primero se deben de calcular los requerimientos de ancho de banda. Utilizando las siguiente ecuaciones:

$$\omega_{BW} = \omega_n \sqrt{(1-2\zeta^2) + \sqrt{4\zeta^4 - 4\zeta^2 + 2}} \tag{3.25}$$

con:

$$P.O. = e^{-\zeta * \pi / \sqrt{1-\zeta^2}} \tag{3.26}$$

Por lo tanto

$$\zeta = \frac{-a}{\sqrt{\pi^2 + a^2}} \tag{3.27}$$

Donde

$$a = ln\left(\frac{P.O.}{100}\right) \tag{3.28}$$

Además si:
$\omega_n = \frac{4}{T_s \zeta} = \frac{\pi}{T_p \sqrt{1-\zeta^2}}$, dados el tiempo de estabilización T_s o el tiempo en donde se produce el pico máximo T_p.
Entonces con relación a T_s:

$$\omega_{BW} = \frac{4}{T_s \zeta} \sqrt{(1 - 2\zeta^2) + \sqrt{4\zeta^4 - 4\zeta^2 + 2}} \tag{3.29}$$

o, también con relación a T_p:

$$\omega_{BW} = \frac{\pi}{T_p \sqrt{1-\zeta^2}} \sqrt{(1 - 2\zeta^2) + \sqrt{4\zeta^4 - 4\zeta^2 + 2}} \tag{3.30}$$

Para este ejemplo: $\zeta = 0.4559498$, y $\omega_{WB} = 46.320489 dB$

2. Calcular K

$$\begin{aligned} K_v &= \lim_{s \to 0} sKG(s) = \lim_{s \to 0} sK \frac{100}{s(s+100)(s+36)} = 40\backslash \\ K_v &= \frac{K(100)}{30000} = 40 \\ K &= 1440 \end{aligned}$$

por lo cual

$$G_1(s) = \frac{144000}{s(s+100)(s+36)}$$

3. La respuesta en frecuencia del sistema no compensado para $K = 144000$, se muestra en la Fig. 3.24

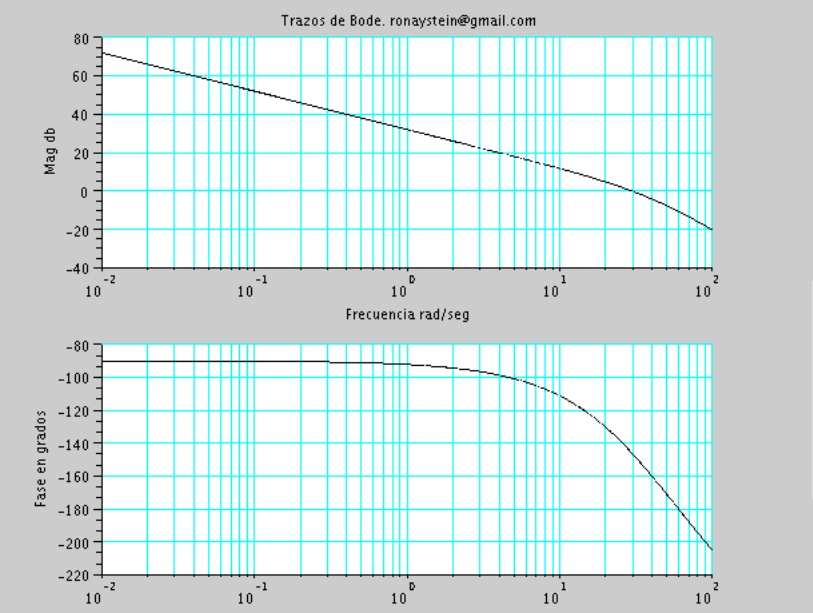


Figura 3.24: Respuesta en Frecuencia de $G_1(s)$

4. Calcular el margen de fase requerido para el porcentaje de sobre tiro dado:

$$Pm = tan^{-1}\left(\frac{2*\zeta}{\sqrt{-2\zeta^2+\sqrt{1+4\zeta^4}}}\right) \tag{3.31}$$

   Por lo cual un 20 % de sobretiro implica un margen de fase de 48.1^o. El sistema no compensado con K=1440, tiene un margen de fase de 34^o[1] en una frecuencia de margen de fase de 29.6. Para incrementar el margen de fase, se debe insertar un compensador de adelanto de fase que agregue suficiente fase al menos a 48.1^o. Considerando que la adición de un compensador de adelanto modifica la curva de magnitud de las trazas de Bode, vemos que la frecuencia de cruce de ganancia se moverá a la derecha. Por lo cual se debe compensar el atraso de fase incrementado de G_1 debido a este incremento en la frecuencia de cruce de ganancia. Como no se conoce la frecuencia de margen de fase más alta, se asume un factor de corrección de 10°. Entonces la contribución total requerida para el compensador es de $48.1° - 34° + 10 = 24.1°$. En resumen, el sistema compensado debe tener un margen de 48.1° con un ancho de banda de 46.6 rad/s. Si las características del sistema después del diseño no son aceptable, entonces debe seleccionarse un factor de corrección diferente. El Ogata sugiere un valor de $5° - 12°$.

5. Usando la Ecu. 3.23, se tiene $\alpha = 0.4202734$.

[1] El valor numérico del margen de fase puede calcularse con la primitiva "p_margin" en scilab

6. De la Ecu. 3.24, la magnitud en dB del compensador de adelanto se calcula como:

$$\begin{aligned} G_c(j\omega_{máx}) &= \frac{1}{\sqrt{\alpha}} = 1.5425315 \\ &= 20log_{10}(1.5425315) = -3.7646dB \end{aligned}$$

7. La frecuencia máxima del compensador de adelanto en -3.7646 dB es de $\omega_{máx} = 39$ rad/s[2].

8. Encontrar las frecuencias de corte del compensador. De la Ecu. 3.21 se tiene,

$$T = \frac{1}{\omega_{máx}\sqrt{\alpha}} \tag{3.32}$$

   $T = 25.2831$, y con lo cual:

$$\begin{aligned} cero &= \frac{1}{T} = 25.2831 \\ polo &= \frac{1}{\alpha T} = 60.158 \\ K_c &= \frac{1}{\alpha} = 2.3794 \end{aligned}$$

9. Las función de transferencia del compensador es dado como:

$$G_c(s) = (2.3794)\frac{s+25.2831}{s+60.158}$$

10. La función de transferencia final en lazo abierto es:

$$G_c(s)G_1(s) = (342,634.32)\frac{(s+25.2831)}{s(s+36)(s+100)(s+60.158)}$$

11. La respuesta del sistema a una entrada de tipo aleatoria es:

---

[2]

- Utilizando la función "bodeR" se puede determinar $\omega_{máx}$ de la siguiente manera: Teclear "res" en la consola (contiene un arreglo de [Phase Magnitud Frecuencia] para todo el rango de frecuencias). Localizar la magnitud de -3.7646 o el valor más cercano en la columna de enmedio. El valor de ω_{max} es el que se encuentra justo a la derecha del valor de -3.7646.

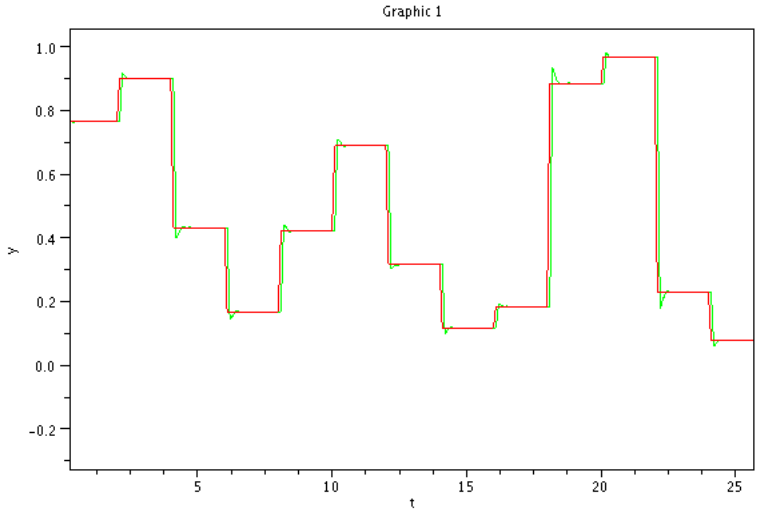


Figura 3.25: Respuesta del compensador en lazo cerrado a una entrada aleatoria. Rojo(referencia). Verde (sistema con el compensador)

Solución utilizando ADS_CoLiSyS

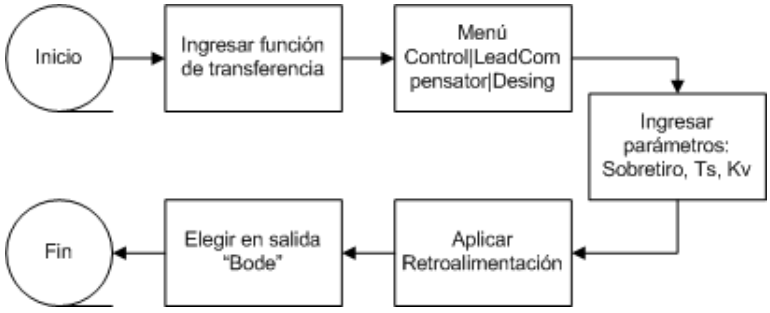
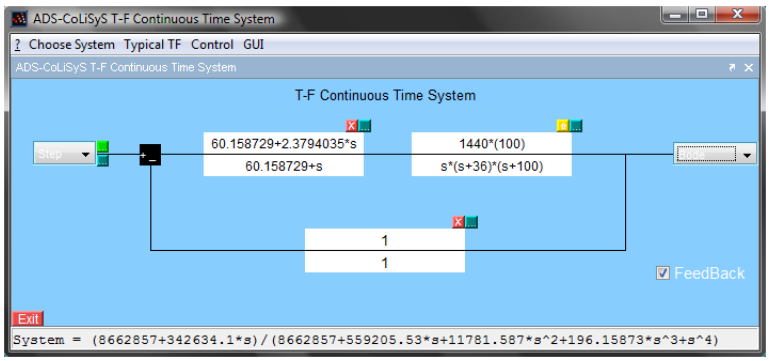
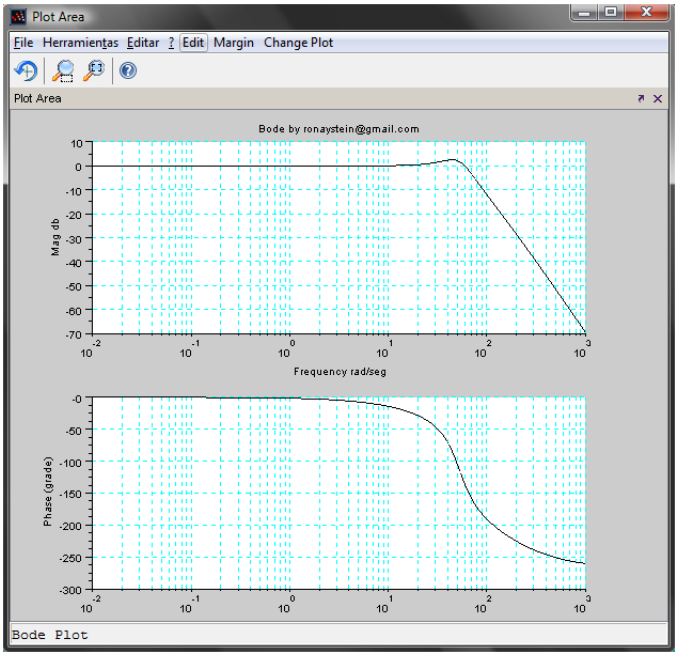


Figura 3.26: Diseño de Compensador de Adelanto Diagrama de Flujo ADS_CoLiSyS para el ejemplo 7



(a) Ventana Principal



(b) Área de Gráficos

Figura 3.27: Solución al ejemplo 7utilizando ADS_CoLiSyS

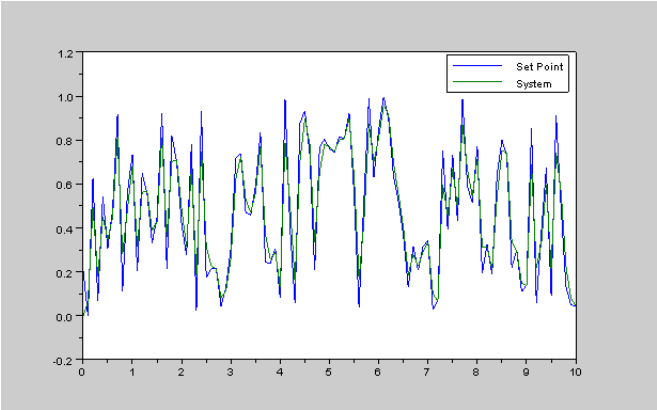


Figura 3.28: Respuesta del sistema compensado a una entrada aleatoria

Observaciones y sugerencias

Este es un proyecto ambicioso que beneficia a toda la comunidad estudiantil, pues se trata de una herramienta académica que hará más fácil el trabajo de enseñanza para los catedráticos y el aprendizaje a los alumnos que cursen materias relacionas a los conceptos de Teoría de Control, por lo que es muy conveniente sugerir:

- Continuar con el desarrollo de la Interfaz
- No abandonar el propósito de la misma
- Distribuirla a todo el que la requiera
- Contribuir al mejoramiento de la misma
- Mantener actualizada la Interfaz
    - Conceptos
    - Aplicaciones
    - Código de Scilab

Conclusiones

Se puede concluir que el desarrollo del proyecto ha sido exitoso pues se cumple con las necesidades de los estudiantes y catedráticos de Teoría de Control, además de tener aplicaciones en algunas otras materias como Electrónica Analógica (I, II, III), Señales y Sistemas, etc.

De acuerdo a las pruebas realizadas con la interfaz, es completamente funcional y no se detectaron errores. A la vez se realizaron pruebas con ejercicios resueltos, en los cuales se obtuvieron los resultados correctos.

El diseño de la Interfaz está construido de forma muy amigable, lo cual beneficia a los usuarios, los cuales podrán utilizar la interfaz casi de manera intuitiva.

Por último cabe señalar que esta Interfaz está en completo desarrollo, pues se le puede continuar agregando funciones de acuerdo a las necesidades del usuario, esto gracias a: el diseño de las funciones y la licencia de software libre.

Referencias

[1] B. Bequette. Process Control, Modeling Desing and Simulation. 2005.

[2] R. S. Burns. Advanced Control Engineering. Butterworth Heinemann. 2001.

[3] S. L. Campbell, Jean-Philippe, and R. Nikoukhah. Modeling and Simulation in Scilab/Scicos. Springer, 2006.

[4] R. C. Dorf and R. H. Bishop. Modern Control Systems. Pearson Prentice Hall, 11 edition, 2008.

[5] G. Ellis. Control System Desing Guide. Thirth edition, 2004.

[6] K. Moudgalya. Digital Control. First edition, 2007.

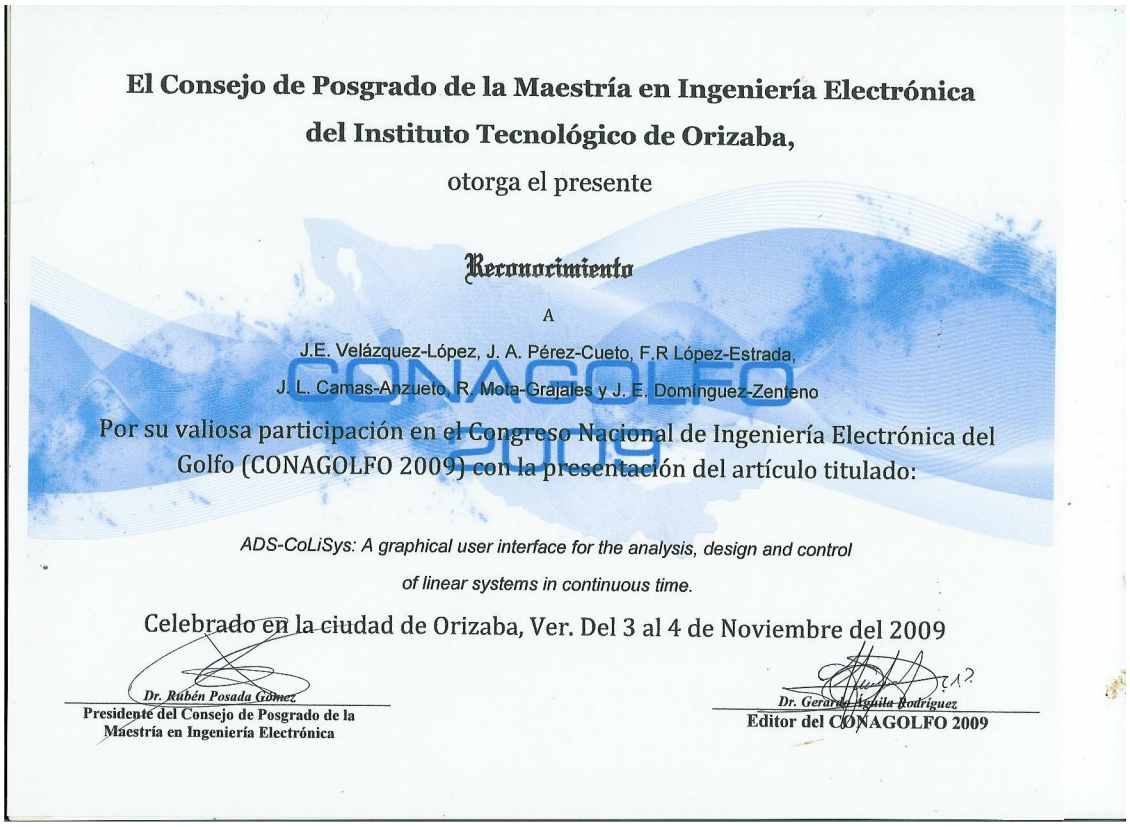
[7] N. S. Nise. Control System Engineering. Four edition, 2004.

[8] www.scilab.org

[9] Manual de Scilab

Anexos

Se anexa una constancia de participación en un Congreso Nacional en el cual el proyecto ha sido expuesto y muy bien aceptado



El Consejo de Posgrado de la Maestría en Ingeniería Electrónica
del Instituto Tecnológico de Orizaba,
otorga el presente

Reconocimiento

A

J.E. Velázquez-López, J. A. Pérez-Cueto, F.R López-Estrada,
J. L. Camas-Anzueto, R. Mota-Grajales y J. E. Dominguez-Zenteno

Por su valiosa participación en el Congreso Nacional de Ingeniería Electrónica del Golfo (CONAGOLFO 2009) con la presentación del artículo titulado:

ADS-CoLiSys: A graphical user interface for the analysis, design and control of linear systems in continuous time.

Celebrado en la ciudad de Orizaba, Ver. Del 3 al 4 de Noviembre del 2009

Dr. Rubén Posada Gómez
Presidente del Consejo de Posgrado de la Maestría en Ingeniería Electrónica

Dr. Gerardo Aguila Rodríguez
Editor del CONAGOLFO 2009

Figura 29: CONAGOLFO 2009