Universidad Central “Marta Abreu” de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Automática y Sistemas Computacionales

Trabajo de Diploma

Interfaz Gráfica en MatLab para el diseño de compensadores con Respuesta en frecuencia

Autor: Alejandro Valdés Pérez

Tutores: Msc. Iván Iglesias Navarro

Santa Clara

2012

“Año 54 de la Revolución”

Universidad Central "Marta Abreu" de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Automática y Sistemas Computacionales

Trabajo de Diploma

Interfaz Gráfica en MatLab para el diseño de compensadores con Respuesta en frecuencia

Trabajo de Diploma presentado en opción al Título Académico de Ingeniero en Automática

Autor: Alejandro Valdés Pérez

email: avperez@uclv.edu.cu

Tutores: Msc. Iván Iglesias Navarro

email: iglesias@uclv.edu.cu

Santa Clara

2012

"Año 54 de la Revolución"

Hago constar que el presente Trabajo de Diploma fue realizado en la Universidad Central “Marta Abreu” de Las Villas como parte de la culminación de estudios de la especialidad de Ingeniería en Automática, autorizando a que el mismo sea utilizado por la Institución para los fines que estime convenientes, tanto de forma parcial como total, y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

_____ _____

Alejandro Valdés Pérez Fecha

Autor

Los abajo firmantes certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Alejandro Valdés Pérez Fecha

Autor

_____ _____

Boris Luis Martínez Jiménez, Dr.C. Fecha

Jefe del Departamento

_____ _____

Responsable ICT o J` de Carrera (Dr.C., Msc. o Ing.) Fecha

Responsable de Información Científico-Técnica

PENSAMIENTO

Toda nuestra ciencia, comparada con la realidad, es primitiva e infantil…

Y sin embargo es lo más preciado que tenemos.

Albert Einstein.

DEDICATORIA

A mi abuela Nancy, por ser esa madre que siempre se ha mantenido a mi lado.

A mi hermano Idalberto, por ser siempre más que un hermano.

A mi padre, por apoyarme en todo momento de mi vida.

A Rocky, por representar tanto para mí.

A mis demás familiares y amigos.

AGRADECIMIENTOS

A mi abuela, porque desde mi infancia ha llevado sobre sus hombros el peso de mi crianza, y me ha hecho quien soy.

A mi padre, Odalis y Bryan, porque sin su apoyo y cariño no hubiese podido lograr tantas cosas, porque siempre han estado.

A mi hermano y Sayli, porque incondicionalmente se han mantenido de mi lado.

A Rockney, por todo lo que ha representado para mí este último año.

A mis amigos del cuarto y de la vida: Carlos Manuel, José Carlos (Kco), Oriel, Sandy, Danny, Yadier, Esteban, Humberto, y todos los que se sentían integrantes del # 202 U-2B, por todos los momentos felices que compartimos.

A mis amigos de clubes, textileras, ranitas, bosques, caneyes, granjitas, arcoíris y tantas noches de alcohol: Yanier, Francis, Elier, Ricardo y otro montón de trinitarios.

A mi tutor y amigo Iván Iglesias, por soportarme hasta altas horas de consultas.

A Rolo, Erick, Fide, Yule y Rocky, por la ayuda brindada en la realización de este trabajo, por todo, gracias.

A los amigos de enseñanzas anteriores, en especial a Pochy, Bryan, Denier, Eylen, Lisaris, Day y Yaimer.

A mis profesores y compañeros de aula.

A mis demás familiares y amigos.

A todos, gracias.

RESUMEN

Las interfaces gráficas se han convertido, a partir de los últimos años, en herramientas sumamente importantes para la facilitación de las más variadas aplicaciones dentro del ámbito ingenieril. Nuestra investigación está basada en la necesidad de crear una interfaz gráfica que facilite el diseño de compensadores a través del método de Respuesta en frecuencia por una vía mucho más rápida y factible que las tradicionales. Partiendo de dicha necesidad, nos hemos planteado el objetivo de crear una interfaz gráfica con el empleo de la herramienta GUIDE de MatLab que permita el diseño de redes de compensación sobre el enfoque de la Respuesta en frecuencia. En tal sentido, se realizó una exhaustiva búsqueda bibliográfica de donde se extrajeron los principales conceptos necesarios para el diseño de la interfaz DCRFtool.

Nuestra aplicación, posibilita dar simplicidad y optimizar tiempo en el proceso de diseño de compensadores, evitando así las constantes búsquedas sobre gráficos y la realización de engorrosos cálculos matemáticos en este proceso, con lo cual se podría incurrir en errores cometidos por la injerencia del factor humano. De aquí deriva, en gran medida, la relevancia y actualidad del trabajo realizado.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Las interfaces gráficas de usuario representan, en la actualidad, el punto de interconexión que permite el constante flujo de información entre el usuario común y el medio de cómputo. La necesidad de confeccionar una herramienta que posibilitara el empleo de la computadora a cualquier usuario, hizo que estas surgieran como resultado de la evolución de las complejas interfaces de líneas de comando empleadas para el manejo de los primeros sistemas operativos. Un conjunto de imágenes y objetos gráficos, tales como: botones, menús, *sliders*, campos de texto, gráficos y otros, hacen posible la representación de la información y las acciones disponibles en la interfaz, proporcionando un entorno gráfico amigable que facilita la interactividad con la máquina.

Con el objetivo de simplificar a los ingenieros el diseño de estos entornos gráficos, múltiples programas como *Visual Basic, Visual C++, MatLab* y otros, poseen diferentes controles y maneras de programar, que posibilitan la confección de las interfaces.

MatLab, con la incorporación de un complemento adicional, destinado específicamente al diseño de interfaces gráficas de usuarios: su herramienta GUIDE, se ha convertido en uno de los principales entornos de programación visual empleado en la actualidad con dicho fin para la realización y ejecución de programas, los cuales requieren del ingreso continuo de datos.

Muchos son los trabajos que en los últimos años se han realizado con la utilización de esta herramienta en la elaboración de interfaces, que faciliten el desarrollo de las más variadas aplicaciones dentro del ámbito ingenieril, permitiendo así dar

simplicidad y optimizar tiempo en aquellas que requieren, por lo general, de métodos complejos y de engorrosos cálculos matemáticos.

Ejemplo de ello es la investigación desarrollada en el campo de la ingeniería Biomédica por María Eugenia Calva, de la Universidad Autónoma Metropolitana Iztapalapa, referente al desarrollo de una interfaz gráfica para un electrocardiógrafo portátil. En este mismo campo, se encuentra el trabajo de Diploma de Saimy Rodríguez Ledesma, egresada de la Universidad Central “Marta Abreu” de Las Villas, dirigido a la creación de una interfaz para la evaluación de dosis radiológicas mediante técnicas de procesamiento digital de imágenes. En el ámbito de la ingeniería Industrial, José Francisco Escribano Molina, en la Universidad Carlos III de Madrid, desarrolló una interfaz gráfica para la aplicación de modelos de Regresión Local Polinómica. Por otra parte, Pedro Valencia Padilla, de la Universidad Tecnológica de la Mixteca, México, implementó una interfaz para el control estadístico de procesos. En la especialidad del control automático, se encontraron también algunos antecedentes, entre los que se destacan el trabajo de Diploma de Carlos Raúl Carballo Garaboto, de la Universidad Central “Marta Abreu” de Las Villas, dirigido a la creación de una interfaz gráfica que permite la identificación para plataforma de dos grados de libertad y el proyecto desarrollado por integrantes del Grupo de Automatización, Robótica y Percepción (GARP), de dicha universidad, donde se elaboró una interfaz denominada “VASmodel” para el modelo dinámico de un AUV[1]. Además, resulta relevante, en cuanto al empleo de la herramienta GUIDE de MatLab, destacar el ejemplo de la aparición, en las últimas versiones de este programa, de la reconocida interfaz “SISOTOOL”, destinada al diseño de reguladores y compensadores con el empleo del enfoque del Lugar geométrico de las raíces.

Por otro lado, una de las principales tareas que en la actualidad se realiza, dentro del campo de la ingeniería de control, es el diseño de redes de compensación que permita a un sistema dado cumplir con determinadas especificaciones, imposibles de alcanzar por sí solo. El proceso de diseño de estos compensadores, a través de métodos convencionales, sin el uso de un medio de cómputo, requiere del trabajo por

[1] Vehículo autónomo subacuático.

parte de un especialista, a través de herramientas matemáticas y continuas búsquedas sobre gráficos. Ello implica, además de un mayor consumo de tiempo en el desarrollo del proceso, la introducción de errores por parte del hombre.

Después de realizar una exhaustiva búsqueda relacionada con la elaboración de interfaces, mediante el uso del apartado GUIDE de MatLab, no se hallaron antecedentes de trabajos referentes al diseño de compensadores a través del enfoque de la Respuesta en frecuencia, que emplearan esta herramienta. Como consecuencia, se plantea la necesidad de realizar una investigación que permita confeccionar una interfaz gráfica para facilitar al usuario involucrado en el tema, el rápido y eficaz diseño de redes de compensación con Respuesta en frecuencia, sin ser preciso que este tenga conocimientos del lenguaje que se emplea en ello. De aquí deriva, en gran medida, la relevancia y actualidad de nuestra investigación.

A partir de las anteriores observaciones, se define como **situación problémica**: ¿Cómo contribuir a la creación de una interfaz gráfica en MatLab, teniendo en cuenta los principios del método de Respuesta en frecuencia que permita el diseño de compensadores?

Motivados por lo anteriormente expuesto, nos hemos planteado las siguientes **interrogantes científicas**:

¿Cuál es la situación actual que presenta el desarrollo de soluciones para el diseño de compensadores con el método de Respuesta en frecuencia a partir de la implementación de interfaces gráficas en MatLab que empleen la herramienta GUIDE?

¿Cómo elaborar una interfaz gráfica en MatLab que nos permita el diseño de redes de compensación a través del método de Respuesta en frecuencia?

¿Cómo evaluar el correcto funcionamiento de la interfaz gráfica en MatLab que permita el diseño de redes de compensación con Respuesta en frecuencia?

Para dar solución a la situación problémica, nos hemos planteado como **objetivo general**, la creación de una interfaz gráfica en MatLab que permita el diseño de

redes de compensación con Respuesta en frecuencia. En tanto, nuestros **objetivos específicos** son:

- Revisar, analizar y sintetizar la bibliografía existente sobre el tema planteado.
- Implementar la programación correspondiente al diseño de la interfaz gráfica.
- Validar los resultados obtenidos.

Como **posible impacto**, pretendemos que nuestra investigación contribuya al desarrollo de una interfaz gráfica sustentada sobre la base de la programación en MatLab, que permita el diseño de compensadores a través del método de Respuesta en frecuencia, y a su vez ofrezca una respuesta favorable a la necesidad de implementar interfaces que posibiliten la fácil obtención de redes de compensación con soluciones económicamente factibles.

Con la aplicación del presente trabajo, se facilitará a investigadores y personas interesadas en el tema, como **posibles resultados**, el diseño fácil y seguro de compensadores a través de una interfaz gráfica que permitirá adquirir el resultado deseado sin necesidad de emplear los convencionales y complejos métodos matemáticos. Además, con la ejecución de los objetivos propuestos, se abrirá el camino a futuros estudios relacionados con el tema a diferentes niveles de investigación.

En tanto, los resultados de nuestro trabajo poseen una **aplicabilidad** práctica de gran trascendencia para ingenieros vinculados a la investigación, que necesiten del diseño de redes de compensación, además de la posible utilización que se pueda dar en estudiantes de la carrera, que luego de conocer los métodos de diseño de compensadores estudiados en clases, podrán corroborar sus resultados con el *software* y hacer de este una potente herramienta para su futura utilización.

La **viabilidad** de nuestra investigación es apreciable porque contamos con los recursos materiales y el personal calificado para la elaboración del mismo. Además, no necesitamos del consumo de un presupuesto económico, ya que la interfaz para el diseño de compensadores se desarrollará sobre la base de la programación en MatLab, *software* que se encuentra al alcance de nuestro grupo investigativo.

Organización del informe

Nuestro informe final está estructurado de la siguiente manera: la presente introducción, a la que siguen, en el mismo orden, capitulario, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Los temas abordados en cada uno de nuestros capítulos son:

- Capítulo 1: La respuesta en frecuencia en el diseño de compensadores. Desarrollo de interfaz gráfica con MatLab, donde se analiza exhaustivamente la literatura crítica especializada en el tema y se abordan los principales conceptos teóricos con los que se trabajan.
- Capítulo 2: DCRFtool: interfaz gráfica para el diseño de compensadores, donde se describe la estructura de la interfaz gráfica diseñada y el algoritmo empleado en la implementación de la misma.
- Capítulo 3: Análisis de los resultados, donde se validan los resultados obtenidos para así medir la efectividad de la interfaz gráfica diseñada.

CAPÍTULO 1 LA RESPUESTA EN FRECUENCIA EN EL DISEÑO DE COMPENSADORES. DESARROLLO DE INTERFAZ GRÁFICA CON MATLAB

Las interfaces gráficas de usuario (GUI[2]) constituyen, en la actualidad, la principal herramienta que posee un usuario común para su intercomunicación con la computadora. El desarrollo de la interfaz gráfica ha proporcionado al hombre una serie de ventajas en pos de facilitar la interacción del mismo con la máquina; un sistema gráfico conformado por ventanas, íconos, menús y otras herramientas, hace posible que el usuario, en un ambiente sencillo y agradable, pueda lograr acciones deseadas sin necesidad de poseer conocimiento alguno de la programación implícita en el hecho.

Las GUIs han desempeñado un rol trascendental en la concepción de la computación moderna: con la implementación de estas se han desarrollado los que hoy conocemos como entornos de escritorio, (tal es el caso de *Windows*) y muchos de los programas que sobre estos se ejecutan.

En la actualidad existen diferentes programas que permiten la elaboración de interfaces gráficas. Entre ellos se pueden mencionar *Visual Basic, Visual C++, MatLab* y otros.

MatLab, que posee un entorno de programación de fácil uso y una herramienta específica llamada GUIDE, se ha convertido en uno de los principales programas

[2] Como se les conoce por sus siglas en inglés.

empleados en la elaboración de GUIs destinados a la realización de aplicaciones ingenieriles.

Muchas de las universidades y centros de investigación y desarrollo que en la actualidad se dedican a la elaboración de proyectos encaminados a la facilitación de disímiles aplicaciones, que dentro del ámbito ingenieril requieren de métodos de gran complejidad, lo hacen a través de la creación de interfaces confeccionadas con la herramienta GUIDE de MatLab. Ejemplo de ello son los trabajos investigativos y de diploma desarrollados por varias universidades, entre las que se encuentran: la Universidad Carlos III de Madrid, la Universidad tecnológica de la Mixteca, la Universidad Autónoma Metropolitana Iztapalapa, y la Universidad Central “Marta Abreu” de Las Villas, tales son los casos de (Escribano 2009), (Valencia 2010), (Calva 2008) y (Carballo 2011; Rodríguez 2011; Valeriano, Oria et al. 2012) respectivamente.

La presente investigación pretende desarrollar una interfaz gráfica con dicha herramienta de MatLab que posibilite el fácil diseño de redes de compensación por el método de Respuesta en frecuencia. En este capítulo, se expondrán los principales conceptos relacionados con el tema, además de una panorámica general en torno al problema que se aborda y que motiva nuestra investigación.

1.1 Respuesta en frecuencia

Cuando hablamos de Respuesta en frecuencia, nos referimos a la respuesta de un sistema en estado estable a una entrada sinusoidal. En los métodos de la Respuesta en frecuencia, la frecuencia de la señal de entrada se varía en un cierto rango, para estudiar la respuesta resultante (Ogata 1998).

Por otra parte (Smith and Corripio 1991) define la Respuesta en frecuencia como el estudio de la manera en que se comporta la razón de amplitud[3], la razón de magnitud[4], y el ángulo de fase[5] de diferentes componentes o sistemas cuando se cambia la frecuencia de entrada.

[3] Razón de la amplitud de la señal de salida respecto a la amplitud de la señal de entrada.
[4] División de la razón de amplitud entre la ganancia de estado estacionario.

El punto de comienzo para el análisis en el dominio de la frecuencia de un sistema lineal es su función de transferencia, según (Kuo 1996). La salida, en estado estable, de un sistema estable, lineal e invariante con el tiempo, ante una entrada sinusoidal con una frecuencia dada, es una función sinusoidal con la misma frecuencia que la entrada, solo que posee, por lo general, magnitud y fase diferentes a las de la señal de excitación.

La Respuesta en frecuencia de un sistema de control, según (Ogata 1998), nos brinda, en gran medida, una imagen cualitativa de la respuesta transitoria del mismo, a pesar de no existir una correlación entre ambas (excepto en el caso de los sistemas de segundo orden).

Por lo general, se usan tres representaciones gráficas de la función de transferencia de un sistema dado ante Respuesta en frecuencia: las trazas de Bode o trazas logarítmicas, la traza de Nyquist o traza polar, y la traza de magnitud logarítmica contra la fase. En nuestro trabajo, solo serán analizadas las dos primeras representaciones.

1.1.1 Trazas de Bode o trazas logarítmicas

Constituyen una representación gráfica de las funciones de transferencia sinusoidales, es decir, funciones transferenciales expresadas en el dominio de la frecuencia, donde se sustituye la variable S por su correspondiente jω. Las trazas de Bode están compuestas por dos gráficas, una que esboza la magnitud de la función de transferencia sinusoidal en términos logarítmicos, y otra con el ángulo de fase de dicha función, ambas gráficas representadas contra la frecuencia en escala logarítmica.

La magnitud logarítmica de la función transferencial sinusoidal G(jω) está dada por la expresión 20log(G(jω)) y se expresa en decibel (dB). Las trazas se representan en gráficas semilogarítmicas donde la escala logarítmica corresponde a la frecuencia, y la lineal a la magnitud (dB) y el ángulo de fase (grados).

[5] Cantidad, en grados o radianes, en que la señal de salida varía respecto a la señal de entrada.

Las trazas logarítmicas brindan una serie de ventajas que facilitan el proceso de graficar las curvas de Respuesta en frecuencia. Al trabajar las magnitudes en forma logarítmica, se simplifica el trabajo, pues esto permite que se puedan analizar cada uno de los factores que conforman la función transferencial sinusoidal por separado y construir la gráfica de magnitud como la suma de todos ellos. Otra ventaja en este sentido es, por ejemplo, que las curvas se esbozan como aproximaciones asintóticas, aunque si deseásemos conocer la curva exacta para mayores especificaciones, resulta muy fácil corregirlas.

El tipo de sistema determina la pendiente de la curva de magnitud logarítmica para las bajas frecuencias. Las constantes de error estático de posición (Kp), velocidad (Kv) o aceleración (Ka) definen el comportamiento de los sistemas de tipo 0, tipo 1 y tipo 2 respectivamente en bajas frecuencias. En un sistema definido solo es posible conocer una de las constantes de error estático.

El análisis del comportamiento de sistemas en bajas frecuencias a partir de la curva de magnitud logarítmica, nos permite conocer la existencia y magnitud del error en estado estable.

La figura 1.1 muestra, a manera de ejemplo, el diagrama de Bode de una función de primer orden, en él se esboza además de la curva de fase, la de magnitud con sus respectivas asíntotas.

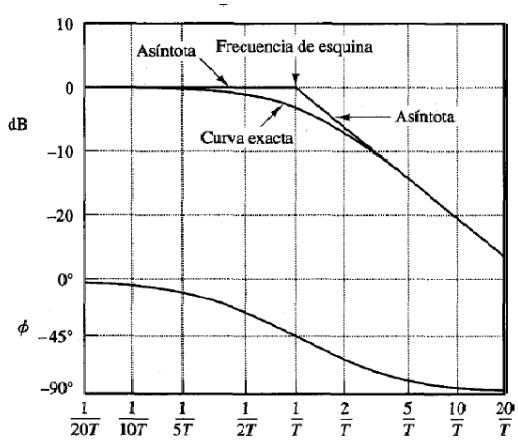


Figura 1.1. Ejemplo de una traza de Bode.

1.1.2 Trazas polares o trazas de Nyquist

La traza polar constituye otra de las representaciones gráficas para la Respuesta en frecuencia de la función transferencial sinusoidal de un sistema determinado. La traza polar es el lugar geométrico de los vectores de magnitud y ángulo de fase de la función transferencial, conforme la frecuencia varía de cero a infinito (Ogata 1998). O sea: es una gráfica donde cada punto representa la magnitud y el ángulo de fase de la función transferencial en lazo abierto para una frecuencia dada. Por otra parte, posee la ventaja de poder encerrar en un mismo gráfico los valores de magnitud y ángulo de fase del sistema en todo el dominio de la frecuencia; sin embargo, no nos proporciona una visión del aporte, en magnitud, que ofrece cada factor de la función de transferencia por separado, como lo hacen las trazas de Bode.

En las trazas polares, los ángulos de fase se consideran positivos cuando se comienzan a medir a partir del eje real positivo en sentido contrario a las manecillas del reloj, y negativos si se hace a favor de las manecillas.

En la figura 1.2 se muestra un ejemplo de traza polar para una determinada función.

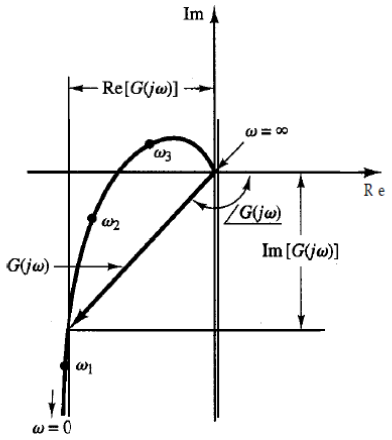


Figura 1.2. Ejemplo de una traza polar.

1.1.3 Diagrama de Nyquist

Como es sabido, para que un sistema sea estable, las raíces de la ecuación característica han de encontrarse en el semiplano izquierdo del plano complejo,

atendiendo a esto y al teorema del mapeo[6], se construye la curva cerrada en el plano GH conocida como diagrama de Nyquist: método gráfico que nos brinda la posibilidad de conocer la estabilidad absoluta de un sistema dado.

El teorema del mapeo se apoya en la condición de que a cada vector en el plano S le corresponde un vector en el plano F(s), donde F(s)= 1+G(s)H(s), para relacionar la cantidad de ceros y polos de la ecuación característica que se encuentran dentro de cierto contorno cerrado en el plano S con el número de encierros, en el sentido de las manecillas del reloj, del origen de coordenadas en el plano F(s) realizado por la curva cerrada que se forma en este plano a partir del mapeo del contorno cerrado del plano S.

Si se define el contorno cerrado del plano S como el semiplano derecho de dicho plano y se verifica, a partir del teorema del mapeo, que el número de ceros de la ecuación característica, es decir, el número de polos de la función de transferencia de lazo cerrado en esta región, es cero, entonces se puede afirmar que el sistema es estable.

El contorno que incluye todo el semiplano derecho del plano S se conoce como trayectoria de Nyquist, el mismo está formado por el eje jω, donde ω varía de -∞ a ∞ y una curva semicircular de radio infinito. Esta trayectoria se construye en el sentido de las manecillas del reloj y no puede pasar por ningún punto singular[7] de F(s), por tanto, en el caso de que la función G(s)H(s) posea algún polo en el origen del plano S o en cualquier otro punto sobre el eje jω se debe hacer una desviación en la trayectoria de Nyquist que permita esquivar dicho punto. Esto se hace con la construcción de un semicírculo con radio infinitesimal ε (donde $\varepsilon \ll 1$) que rodee al punto pero no lo incluya (contorno modificado de Nyquist).

Como se puede observar, 1+G(jω)H(jω) es la suma de vectores del vector unitario y el vector G(jω)H(jω). Por tanto, el hecho de encerrar al origen en el plano F(s) equivale a encerrar el punto (-1+j0) por el lugar geométrico de G(jω)H(jω) en el

[6] Teorema perteneciente a la teoría de la variable compleja.
[7] Cero o polo de la función transferencial del sistema.

plano GH, entonces, la estabilidad de un sistema determinado se establece a través del análisis del número de encierros del punto (-1+j0) en el plano GH(jω).

1.1.4 Criterio de estabilidad de Nyquist

Como decíamos en el subepígrafe anterior, el criterio de estabilidad de Nyquist relaciona la Respuesta en frecuencia de un sistema dado en lazo abierto con la cantidad de ceros y polos del denominador en lazo cerrado, que se encuentran en el semiplano derecho del plano S. El mismo permite, de forma gráfica, determinar la estabilidad absoluta de un sistema en lazo cerrado a partir del análisis de las curvas de Respuesta en frecuencia del sistema en lazo abierto, sin necesidad de conocer los polos en lazo cerrado.

El criterio en sí, plantea que en un sistema dado, si la función de transferencia en lazo abierto G(s)H(s) tiene k polos en el semiplano derecho del plano S, para ser estable, el lugar geométrico G(s)H(s) debe encerrar k veces el punto (-1+j0) en sentido contrario a las manecillas del reloj (Ogata 1998). O sea, para que un sistema sea estable, debe cumplirse la condición de que el número de ceros (Z) de la ecuación característica (1+G(s)H(s)) en el semiplano derecho del plano S debe ser igual al número de polos (P) de la función en lazo abierto (G(s)H(s)) en el semiplano derecho del plano S más el número de encierros (N) en el sentido de las manecillas del reloj del punto (-1+j0) (Z=P+N). Como que para que un sistema sea estable, las raíces de la ecuación característica han de encontrarse en el semiplano izquierdo del plano S, es decir, Z debe ser igual a cero, el criterio se reduce a expresar que P debe ser igual a –N, o sea, P igual al número de encierros del punto (-1+j0) en sentido contrario a las manecillas del reloj.

Las trazas de Nyquist, al igual que las trazas de Bode, constituyen representaciones gráficas de la respuesta en frecuencia de determinados sistemas. Tanto una como otra pueden resultar más convenientes para realizar una operación específica, pero en general, la misma operación puede realizarse en ambas trazas. Dado lo problemático que resulta determinar en MatLab el diagrama de Nyquist para sistemas de tipo 1 en adelante, debido a que esto implica, en el programa, dividir entre cero, lo cual conlleva a la obtención de una

traza errónea, y además teniendo en cuenta las ventajas que propicia el uso de las trazas de Bode, el diagrama de Nyquist y por ende su criterio de estabilidad, no serán incluidos en el desarrollo de la interfaz.

1.1.5 Estabilidad relativa

Además de la estabilidad absoluta, que define si un sistema es estable o no, otra de las características más importantes que presenta el comportamiento dinámico de un sistema de control es la estabilidad relativa. Una vez que se ha encontrado que el sistema es estable, es interesante determinar qué tan estable es, y este grado de estabilidad es una medida de la estabilidad relativa (Kuo 1996). Es conocido que cuando se aplica una entrada a un sistema físico determinado, la salida del mismo no sigue a la entrada de inmediato, sino que experimenta una respuesta transitoria antes de alcanzar el estado estable, en donde, por lo general, se producen armónicos amortiguados, que en dependencia de su comportamiento pueden afectar ciertos parámetros que modifican la respuesta del sistema antes de alcanzar el estado estable. Uno de estos parámetros es el máximo sobreimpulso o sobrepaso máximo (Mp), cuya cantidad, en porcentaje, indica de manera directa la estabilidad relativa del sistema (Ogata 1998).

En el caso del análisis de sistemas en Respuesta en frecuencia, la estabilidad relativa se determina en términos de ***márgenes de fase*** y de ***ganancia***. Considerando que un sistema posea realimentación unitaria, los márgenes de fase y de ganancia no son más que una medida de la proximidad del lugar geométrico G(jω) al punto (-1+j0). En caso de que nuestro sistema sea estable, cuanto más próximo esté G(jω) a dicho punto, más cerca estará del borde de la inestabilidad, y por tanto, de presentar oscilaciones sostenidas. En tanto, si el sistema es inestable, mientras más cerca esté G(jω) del punto (-1+j0), estará más próximo a estabilizarse.

1.1.5.1 Margen de fase

El margen de fase en un sistema con realimentación unitaria, es la cantidad de atraso de fase adicional, en la frecuencia en la que la magnitud de la función de

transferencia en lazo abierto G(jω) es igual a uno, que se requiere para llevar el sistema al borde de la inestabilidad (Ogata 1998).

Por su parte, (Kuo 1996) lo define como: el ángulo (en grados) que la traza G(jω) se debe rotar alrededor del origen, para que el cruce de ganancia pase por el punto (-1+j0), donde el cruce de ganancia es el punto sobre la traza G(jω) en que su magnitud es igual a uno.

1.1.5.2 Margen de ganancia

El margen de ganancia se define como el recíproco de la magnitud de la traza G(jω) en la frecuencia en la cual el ángulo de fase es -180°, o sea: $Kg=1/|G(j\omega)|$ (Ogata 1998). El mismo facilita el análisis de la estabilidad de un sistema dado, a partir de la observación de la Respuesta en frecuencia en las trazas de Bode, ya que su valor se puede expresar en dB, por tanto, se puede decir, además, que el margen de ganancia es la cantidad de ganancia en dB que se puede añadir al lazo antes de que el sistema en lazo cerrado se vuelva inestable (Kuo 1996).

Los márgenes de fase y de ganancia de un sistema nos brindan, además, la información acerca de la estabilidad absoluta del mismo, por ejemplo, podemos afirmar que en todo sistema de fase mínima[8], si los márgenes de fase y de ganancia son positivos, el sistema es estable, en caso contrario, si el sistema es de fase no mínima, para ser estable, los márgenes de fase y de ganancia han de ser negativos.

1.1.6 Consideraciones básicas sobre márgenes de fase y de ganancia

Los márgenes de fase y de ganancia resultan de gran importancia en el diseño de sistemas de control y en el ajuste de las constantes de ganancia de los sistemas. Es importante conocer que para que un sistema de fase mínima logre tener un desempeño satisfactorio, es decir, que aun cuando se varíe la ganancia de la función transferencial en lazo abierto y las constantes de tiempo de los factores

[8] Sistemas en los que la función transferencial de lazo abierto G(s) no tiene polos ni ceros en el semiplano positivo del plano S.

que la conforman en cierto grado, se garantice su estabilidad, el margen de fase debe oscilar entre 30° y 60° y el margen de ganancia debe ser mayor de 6 dB.

Según (Ogata 1998), la condición de que el margen de fase deba encontrarse entre 30° y 60° posee un significado en las trazas de Bode, tal que la pendiente de la traza en la frecuencia de cruce de ganancia[9] debe ser menos abrupta que -40 dB/década. Por lo general, cuando el cruce de la traza por 0 dB se produce con una pendiente de -20 dB/década, en la mayoría de los casos prácticos, el sistema es estable. En el caso en que la pendiente es de -40 dB/década el sistema puede ser o no estable, y si lo es, el margen de fase es muy pequeño. Por último, si la pendiente es de -60 dB/década o mayor, lo más probable es que nos encontremos frente a un sistema inestable.

1.1.7 La Respuesta en frecuencia en el diseño y la compensación de sistemas de control

Para realizar el diseño y la compensación de un sistema de control, se pueden utilizar diferentes métodos en pos de alcanzar los objetivos deseados, tal es el caso de los desarrollados a partir de los enfoques del Lugar Geométrico de las Raíces (LGR), la Respuesta en Frecuencia (RF) y el Espacio de Estado (EE).

El enfoque adecuado se determina a partir de la forma de los datos que describen las características dinámicas del sistema, es decir, de los requerimientos que deseamos cumpla el sistema analizado. Dado el caso en que los datos que describen las características dinámicas del sistema estén expresados en términos de margen de fase, margen de ganancia y otros, propios del análisis de la Repuesta en frecuencia, el enfoque que ha de aplicarse en el diseño y la compensación del sistema es el de la Respuesta en frecuencia, caso específico al que dedicaremos nuestro trabajo.

1.1.7.1 Ventajas del diseño con la Respuesta en frecuencia

El uso del enfoque de la Respuesta en frecuencia en el diseño y la compensación de sistemas de control proporciona numerosas ventajas. Según (Ogata 1998), el

[9] Frecuencia en la que la magnitud de G(jω) es de 0 dB.

diseño en el dominio de la frecuencia es sencillo y directo, la gráfica de la Respuesta en frecuencia indica, en forma clara, la manera en que debe modificarse el sistema, aunque no sea posible hacer una predicción cuantitativa exacta de las características de la respuesta transitoria. Además, la Respuesta en frecuencia facilita la obtención, de forma experimental, de las características dinámicas pertenecientes a aquellos componentes de los cuales se nos hace muy difícil conocer las ecuaciones que los controlan.

Cuando se trabaja con ruidos de alta frecuencia, el enfoque de Respuesta en frecuencia permite diseñar un sistema en el que se desprecian los efectos inconvenientes del ruido. Otra ventaja que expone (Kuo 1996) en el diseño con RF, es que en el enfoque de la frecuencia, los sistemas de orden superior no presentan mayores problemas que los inferiores, mientras que el diseño en el dominio del tiempo, ante especificaciones tales como: máximo sobreimpulso, tiempo de asentamiento y otras, solo es factible analíticamente para sistemas de segundo orden u otros que se puedan aproximar a segundo orden.

Para el diseño en el dominio de la frecuencia, se utilizan, por lo general, dos enfoques: la traza polar y la de Bode, no obstante, dada la facilidad al graficar cuando se incorporan nuevos términos a la traza original, se recomienda el uso de las trazas de Bode, pues en el caso de la traza polar, ante la misma situación, esta no conserva su forma original y se hace necesario el esbozo completo de una nueva traza. Además, en las trazas de Bode, cuando se modifica la ganancia en lazo abierto, la curva de magnitud no varía su pendiente, solo se desplaza hacia arriba o hacia abajo, y la de fase permanece intacta.

1.2 Compensadores

En cualquier proceso dado, se desea que el sistema involucrado cumpla determinados requerimientos que satisfagan las principales necesidades para alcanzar, de manera eficiente, el objetivo final. En muchas ocasiones, suele suceder que el sistema, por sí solo, es incapaz de alcanzar estos requerimientos, incluso cuando se modifica su ganancia. Por ello, se hace necesario la

compensación del mismo, es decir, su modificación en pos de conseguir los resultados deseados.

Para lograr la debida modificación de la dinámica de los sistemas, se deben instalar compensadores: dispositivos físicos que se incorporan a un sistema determinado con el objetivo de cambiar la dinámica del mismo. Ello se logra integrando la función transferencial que los caracteriza a la función transferencial propia del sistema, todo ello encaminado a satisfacer determinadas especificaciones.

Los compensadores, atendiendo a su estructura, pueden ser electrónicos, hidráulicos, neumáticos, etc. El tipo de compensador que se incorpora a un sistema, por lo general, está dado por la naturaleza de la planta que se va a controlar. En la actualidad, los mayormente usados son los electrónicos (confeccionados a base de circuitos que usan amplificadores operacionales), lo que está dado, a su vez, por lo común de convertir las señales de diferentes naturalezas en señales eléctricas, su confiabilidad, la sencillez en la transmisión, su precisión y la facilidad en la compensación (Ogata 1998).

1.2.1 Tipos de compensadores

Entre los muchos tipos de compensadores que pueden citarse, los más usados son los de adelanto, atraso y atraso-adelanto. Los nombres de estos compensadores están dados por sus respectivas características en el dominio de la frecuencia.

Los compensadores de adelanto, atraso o atraso-adelanto, presentan las características de una red de adelanto, una de atraso o una de atraso-adelanto respectivamente. La red de adelanto es aquella a la cual, aplicándosele una señal de excitación sinusoidal a la entrada, su salida en estado estable, que es también una función sinusoidal, experimenta un adelanto de fase. En tanto, la red de atraso, ante la misma entrada, entrega una señal de salida con un desfasaje en atraso. Por su parte, la red de atraso-adelanto experimenta ambos desfasajes respecto a la entrada, solo que en diferentes regiones de frecuencia: el atraso de

fase se obtiene en la región de baja frecuencia, mientras que el adelanto se produce en la región de frecuencias altas.

1.2.2 Compensador de adelanto. Metodología de diseño con RF

El papel principal que desempeña el compensador de adelanto en la compensación de un sistema de control, es que brinda un ángulo de adelanto de fase capaz de contrarrestar el atraso excesivo que proporcionan los diferentes componentes al sistema fijo. Este compensador produce un mejoramiento en la velocidad de respuesta del sistema y un pequeño cambio en la precisión en estado estable, sin embargo posee la desventaja de aumentar el ruido en altas frecuencias. En un análisis de las trazas de Bode, se puede observar con claridad que el compensador de adelanto se comporta como un filtro paso-altas, donde pasan las altas frecuencias y se atenúan las bajas.

Como mencionamos anteriormente, los compensadores más usados en la actualidad son los electrónicos, confeccionados a partir de amplificadores operacionales. La figura 1.3 presenta la configuración general de los compensadores de adelanto y de atraso electrónicos.

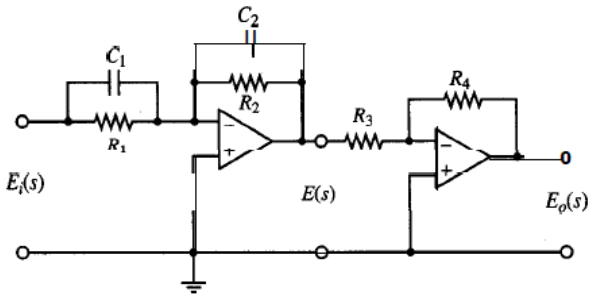


Figura 1.3. Circuito electrónico de una red de adelanto o de atraso.

Desarrollando $G_c(s) = \frac{E_0(s)}{E_1(s)}$, obtenemos la correspondiente función transferencial del compensador de adelanto.

$$G_c(s) = K_c\alpha\frac{Ts+1}{\alpha Ts+1} = K_c\frac{s+\frac{1}{T}}{s+\frac{1}{\alpha T}} \qquad (0<\alpha<1)$$

K_c: Ganancia del compensador.

α: Factor de atenuación.

T: Constante de tiempo del cero de la función de transferencia del compensador.

en donde:

$$T = R_1C_1, \qquad \alpha T = R_2C_2, \qquad K_c = \frac{R_4C_1}{R_3C_2}$$

Como podemos observar, el hecho de que $0<\alpha<1$, determina que el polo esté más alejado del eje $j\omega$ que el cero, además de la medida en la que lo estará; esto es lo que marca la diferencia entre el compensador de adelanto y el de atraso.

Antes de comenzar el diseño del compensador, han de conocerse ciertas características en frecuencia del mismo, para ello se esboza la traza polar asociada, que se presenta en la figura 1.4. En la misma se consideró $K_c = 1$. El ángulo que se forma entre el eje real positivo y la tangente a la curva polar es el ángulo de adelanto de fase máximo ϕ_m y la frecuencia del punto de tangencia, es decir, en donde se obtiene ϕ_m, se define como ω_m. Mediante arreglos algebraicos, llegamos a las expresiones:

$$\text{sen}\phi_m = \frac{1-\alpha}{1+\alpha} \qquad \text{y} \qquad \omega_m = \frac{1}{\sqrt{\alpha}T}$$

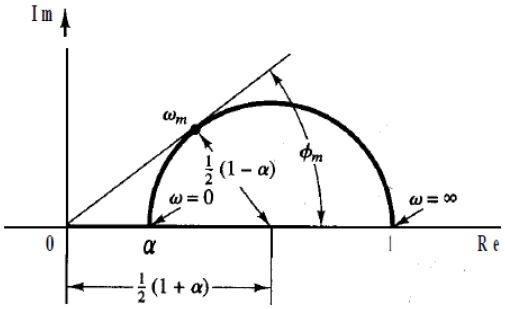


Figura 1.4. Traza polar del compensador de adelanto.

A continuación, se expone el procedimiento presentado por Ogata para el diseño de un compensador de adelanto, considerando que las especificaciones del desempeño del sistema se dan en términos de Margen de fase, Margen de ganancia, constantes de error estático, etc.

1 Suponiendo que la función transferencial de lazo abierto, incluyendo al compensador de adelanto en serie sea:

$$G_c(s)G(s) = K\frac{Ts+1}{\alpha Ts+1}G(s) = \frac{Ts+1}{\alpha Ts+1}KG(s) = \frac{Ts+1}{\alpha Ts+1}G_1(s)$$

en donde:

$$K_c\alpha = K \qquad \text{y} \qquad G_1(s) = KG(s)$$

Determinar la ganancia K que satisfaga el requerimiento sobre la constante estática de error determinada.

2 Usando la ganancia K determinada, dibujar las trazas de Bode de $G_1(j\omega)$, con la ganancia ajustada pero sin compensar. Calcular el valor del margen de fase.

3 Determinar el ángulo de fase ϕ_m necesario que se agregará al sistema.

4 Determinar el factor de atenuación α a partir de la expresión del sen ϕ_m. Establecer la frecuencia a la cual la magnitud del sistema no compensado $G_1(j\omega) = -20\log\left(\frac{1}{\sqrt{\alpha}}\right)$. Seleccionar esta como la nueva frecuencia de cruce de ganancia. Esta frecuencia corresponde a ω_m y el cambio de fase máximo ϕ_m ocurre en ella.

5 Determinar las frecuencias de esquina del compensador de adelanto del modo siguiente:

Cero del compensador de adelanto: $\omega = \frac{1}{T}$

Polo del compensador de adelanto: $\omega = \frac{1}{\alpha T}$

6 Usando el valor de K determinado en el paso 1 y el de α establecido en el paso 4, calcular la constante K_c a partir de $K_c = \frac{K}{\alpha}$.

7 Verificar el margen de ganancia para asegurarse de que es satisfactorio. De no ser así, repetir el proceso de diseño modificando la ubicación de los polos y ceros del compensador hasta obtener un resultado satisfactorio.

1.2.3 Compensador de atraso. Metodología de diseño con RF

El compensador de atraso es básicamente aquel que proporciona una atenuación en el rango de las altas frecuencias con el fin de mejorar el margen de fase del sistema, dado que su característica de atraso de fase no sirve para la compensación. El mismo mejora notablemente la precisión del sistema en estado estable, a cambio de reducir su velocidad de respuesta. Además, posee la ventaja de atenuar los efectos de ruidos en altas frecuencias, o sea, se comporta como un filtro paso-bajas.

Este compensador posee la misma configuración circuital y función transferencial que el compensador de adelanto, solo que en el compensador de atraso, al factor de atenuación α se le llama β, y el mismo posee un valor numérico siempre mayor que la unidad, lo que implica un cambio en los valores de los componentes que conforman el circuito, además de que en el plano complejo el cero estará más alejado del eje $j\omega$ que el polo.

El procedimiento propuesto por Ogata para el diseño del compensador de atraso es el siguiente:

1 Suponiendo la función transferencial de lazo abierto como:

$$G_c(s)G(s) = K\frac{Ts+1}{\beta Ts+1}G(s) = \frac{Ts+1}{\beta Ts+1}KG(s) = \frac{Ts+1}{\beta Ts+1}G_1(s)$$

en donde:

$$K_c\beta = K \quad \text{y} \quad G_1(s) = KG(s)$$

Determinar la ganancia K que satisfaga el requerimiento en la constante de error estático establecida.

2 Si el sistema no compensado $G_1(j\omega) = KG(j\omega)$ no satisface las especificaciones en los márgenes de fase y de ganancia, se debe encontrar el punto de frecuencia en el cual el ángulo de fase de la función de transferencia en lazo abierto sea igual a -180° más el margen de fase requerido, además de un margen de seguridad entre 5° y 12° que se incorpora para compensar el atraso de fase que introduce el compensador. Se selecciona esta como la nueva frecuencia de cruce de ganancia.

3 Seleccionar la frecuencia de esquina $\omega = \frac{1}{T}$ (que corresponde al cero del compensador de atraso) una década por debajo de la nueva frecuencia de cruce de ganancia.

4 Determinar la atenuación necesaria para disminuir la curva de magnitud a 0 dB en la nueva frecuencia de cruce de ganancia. Considerando que esta atenuación es de $-20\log\beta$, se determina el valor de β.

5 Obtener la otra frecuencia de esquina (que corresponde al polo del compensador de atraso) a partir de $\omega = \frac{1}{\beta T}$.

6 Usando el valor de K determinado en el paso 1 y el de β obtenido en el paso 4, calcule la constante K_c a partir de la expresión $K_c = \frac{K}{\beta}$.

7 Verificar que se satisfagan los requerimientos de diseño.

1.2.4 Compensador de atraso-adelanto. Metodología de diseño con RF

Este compensador, como su nombre lo indica, es una integración de los compensadores de atraso y de adelanto, por tanto, combina características propias de ambos. Su función transferencial está dada por la siguiente expresión:

$$G_c(s) = K_c \frac{(T_1 s + 1)(T_2 s + 1)}{\left(\frac{T_1}{\beta} s + 1\right)(\beta T_2 s + 1)} = K_c \frac{\left(s + \frac{1}{T_1}\right)\left(s + \frac{1}{T_2}\right)}{\left(s + \frac{\beta}{T_1}\right)\left(s + \frac{1}{\beta T_2}\right)}$$

en donde

$$\beta > 1$$

En este compensador, los factores que contienen a T_1 son los pertenecientes a la parte de adelanto, que es la que propicia un ángulo de adelanto de fase al sistema y, por tanto, un incremento del margen de fase en la frecuencia de cruce de ganancia. En tanto, los factores que contienen a T_2 corresponden a la parte de atraso de fase: la que proporciona una atenuación de las altas frecuencias y un aumento de la ganancia en el rango de bajas frecuencias, mejorando así el desempeño en estado estable. El compensador de atraso-adelanto se comporta como un filtro paso-banda.

La metodología de diseño del compensador de atraso-adelanto propuesta por Ogata se da a conocer a continuación:

1 Determinar el valor K que satisfaga el requerimiento de estado estacionario.

2 Determinar el margen de fase del sistema sin compensar con la ganancia K incluida.

3 Seleccionar una nueva frecuencia de cruce de ganancia en el tramo donde la pendiente de la curva sea de -40 dB/década, de manera que en esa zona se sitúe la parte del compensador que posee la pendiente de 20 dB/década y como resultado, la red compensada experimente un cruce con -20 dB/década.

4 Seleccionar la frecuencia de esquina $\omega = \frac{1}{T_2}$ del cero de la parte de atraso una década por debajo de la nueva frecuencia de cruce.

5 Determinar la frecuencia de esquina del polo de la parte de atraso. Para ello es necesario calcular β a partir de la expresión:

$$\operatorname{sen}\phi_m = \frac{1-\alpha}{1+\alpha}$$

en donde

$$\alpha = \frac{1}{\beta}$$

y

$$\phi_m = \phi_{\omega c\,\text{deseado}} - \phi_{\omega c\ \text{real}} + \text{margen de seguridad}$$

$$\phi_{\omega c\ \text{deseado}} = -180° + MF$$

6 Determinar el polo y el cero de la parte de adelanto. Se debe trazar una recta con pendiente 20 dB/década que pase por el punto (nueva frecuencia de cruce; - atenuación), donde la atenuación es la cantidad de decibeles que proporciona la nueva frecuencia de cruce y que la red compensada debe atenuar. En el punto donde esta recta choque con la zona de alta frecuencia de la red de atraso, se tiene la frecuencia de esquina del cero y en donde choque con la recta de 0 dB, se obtendrá la frecuencia de esquina del polo.

7 Comprobar si el diseño satisface los requisitos.

1.3 Interfaces gráficas

Las interfaces se pueden definir, de forma general, como el punto de interconexión entre dos entidades, sistemas, equipos, conceptos, etc (Babylon 2012). En la actualidad, existen diferentes tipos de interfaces, estas se clasifican según su naturaleza en interfaces electrónicas (aquellas por donde se envían o reciben señales de un sistema a otro, por ejemplo, el interfaz USB), interfaces de hardware (monitor, teclado, mouse), interfaces de línea de comando, interfaces gráficas y otras.

Una interfaz gráfica de usuario, conocida también como GUI, que surge como evolución de las interfaces de línea de comando utilizadas para manejar los primeros sistemas operativos, es el conjunto de métodos que permite lograr la fácil interactividad entre un usuario y una computadora (López 2009). Utilizando un conjunto de imágenes y objetos gráficos para representar la información y las acciones disponibles en la aplicación, la misma proporciona un entorno visual sencillo que facilita la interacción amigable con un medio de cómputo. El usuario común, habitualmente, realiza las acciones deseadas con la manipulación directa de la interfaz, sin necesidad de conocer la programación implícita en el hecho.

1.3.1 MatLab como herramienta en el desarrollo de aplicaciones automáticas y el diseño de interfaces gráficas

MatLab es, en la actualidad, uno de los *softwares* más utilizado, en el ámbito ingenieril, por múltiples universidades y centros de investigación y desarrollo. Debe su nombre a la abreviatura de *Matrix Laboratory* y es un programa muy potente para realizar cálculos numéricos con vectores y matrices (Escribano 2009). Este *software* trabaja con el lenguaje *M* y permite resolver muchos problemas matemáticos, específicamente aquellos que involucran vectores y matrices, en un tiempo mucho menor al requerido para escribir un programa en un lenguaje escalar no interactivo tal como *C* o *Fortran* (Esqueda 2002).

Entre sus principales prestaciones, se destacan el desarrollo de algoritmos, cálculos numéricos, modelado, simulación y prueba de prototipos, análisis de datos, exploración y visualización, graficación de datos y desarrollo de aplicaciones que requieran de una GUI (Carballo 2011).

Debido a la amplia gama de aplicaciones de tipo matemático, vinculadas al control automático (Ong and Tan 2000; DeMoyer and Mitchel 2002), que proporciona este *software*, el mismo se ha seleccionado para la realización de las principales tareas técnicas propuestas en nuestro trabajo: el diseño de compensadores a través de la Respuesta en frecuencia y la elaboración de una interfaz que facilite al usuario el rápido y fiable diseño de los compensadores anteriormente mencionados.

1.3.2 Herramienta GUIDE del MatLab

Desde los inicios de la informática, la meta general de los programadores ha sido lograr que la tarea de realizar programas para ordenadores sea cada vez lo más simple, flexible y portable posible (Francisco 2000), sobre esa base, MatLab dispone, en la actualidad, de una herramienta adicional llamada GUIDE.

GUIDE es un entorno de programación visual capaz de crear y manipular interfaces gráficas en MatLab (Mathworks 2007), presenta las características básicas de cualquier programa visual como *Visual Basic* o *Visual C++* (Barragán 2008). Esta herramienta permite al usuario ejecutar instrucciones a través del trabajo con botones, menús, *sliders*, cuadros de diálogo, etc. Cada uno de los elementos ya mencionados que conforman la interfaz, se pueden personalizar a conveniencia del diseñador de la misma, además, poseen una subrutina asociada que se ejecutará cuando se realice una acción determinada sobre el elemento en sí (Rodríguez 2011).

1.3.3 Ventajas de la herramienta GUIDE

La principal ventaja de la herramienta GUIDE es que permite realizar el diseño de una interfaz gráfica sin necesidad de recurrir a la utilización del método tradicional: la programación con líneas de código, utilizando funciones propias del MatLab (Valeriano, Oria et al. 2012). Ello conlleva a que el diseñador, empleando un

ambiente amigable, con un alto nivel de interactividad, pueda realizar mayor cantidad de operaciones en un período menor de tiempo. Otra ventaja es que permite realizar y ejecutar programas que necesiten ingreso continuo de datos.

1.4 Consideraciones finales del capítulo

En este capítulo se ha realizado un análisis introductorio al tema de las interfaces gráficas, hemos expuesto los principales conceptos relacionados tanto con las interfaces como con las herramientas que propiciarán su creación, además de abordar otros conceptos referidos al diseño de compensadores y a los métodos empleados en la determinación de los mismos.

Como resultado de dicho análisis, se evidencia la importancia de la utilización de MatLab y su herramienta GUIDE en la construcción de interfaces y el diseño de redes de compensación, así como las ventajas que propina el uso del enfoque de la RF en el diseño de los compensadores anteriormente mencionados.

En tal sentido, hemos determinado utilizar el entorno de programación visual GUIDE en la confección de la interfaz gráfica dada la necesidad, en el ámbito de la ingeniería de control, de una herramienta de *software* que posibilite el fácil y seguro diseño de compensadores; para lo cual nos hemos planteado, como basamento teórico, la utilización del método propuesto por (Ogata 1998) sustentado en el enfoque de la RF.

CAPÍTULO 2 DCRFTOOL: INTERFAZ GRÁFICA PARA EL DISEÑO DE COMPENSADORES

Dada la importancia de la utilización de MatLab en la construcción de interfaces a través de su herramienta GUIDE, evidenciada en el capítulo anterior, además del sinnúmero de prestaciones relacionadas a la ingeniería de control que brinda este *software*, y tomando como base las necesidades que nos motivaron a la realización de este trabajo, en el presente capítulo procedemos a describir el desarrollo de la interfaz nombrada DCRFtool: herramienta que nos facilitará el rápido y seguro diseño de compensadores, empleando el método de RF.

2.1 Introducción al trabajo con GUIDE

El desarrollo de una interfaz gráfica de usuario en MatLab es realizable a través de dos vías diferentes: la primera de estas es mediante la realización de un programa que al ejecutarlo genere la GUI, es decir, un *script*. En tanto, la segunda opción, consiste en utilizar el entorno de programación visual GUIDE, disponible en el paquete de MatLab para el diseño y manipulación de interfaces. Para la realización de nuestro programa, nos hemos basado en las posibilidades que ofrece esta segunda opción, es decir, el empleo del entorno GUIDE.

2.1.1 Iniciando GUIDE

Existen varias vías que nos permiten acceder a la herramienta GUIDE de MatLab para la confección de una GUI: una de ellas es tecleando el comando “guide” en la ventana de comandos. Otra es accediendo al menú “File” en la ventana principal de MatLab para seleccionar el submenú “New” y posteriormente “GUI”, o

simplemente, podemos acceder haciendo clic sobre el ícono que simboliza la herramienta, disponible en la barra de tareas.

Al utilizarse GUIDE en la fabricación de una interfaz, se obtienen dos archivos: un archivo con extensión ".fig", que contiene la descripción de los componentes que conforman la GUI y otro con extensión ".m", que contiene las funciones y los controles de la interfaz, así como el *callback*[10] asociado a cada uno de ellos.

Utilizando cualquiera de las vías mencionadas anteriormente, obtendremos la ventana que se presenta en la figura 2.1.

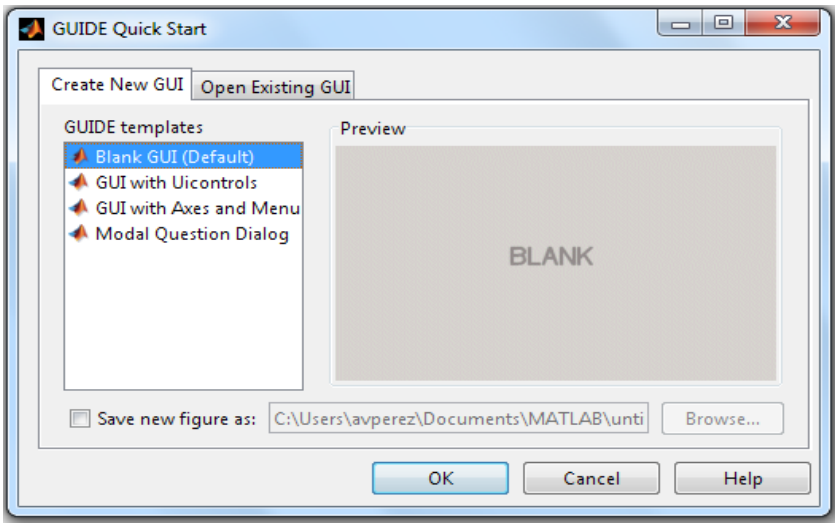


Figura 2.1. Ventana de inicio de la herramienta GUIDE de MatLab.

Para crear una nueva GUI debemos seleccionar la primera opción, señalada por defecto. Una vez realizada esta operación, en la pantalla se visualizará un área de diseño similar a la representada en la figura 2.2.

Esta ventana se encuentra conformada por la paleta de componentes, ubicada en el lateral izquierdo, donde están disponibles todos los controles a utilizar por el diseñador. Los menús y opciones de GUIDE aparecen, por otra parte, en el extremo superior de la misma. Finalmente, en la zona central de dicha ventana, donde colocaremos los componentes deseados, para conformar la GUI, encontramos el área de diseño.

[10] Función a la que se hace llamado cuando se activa un control.

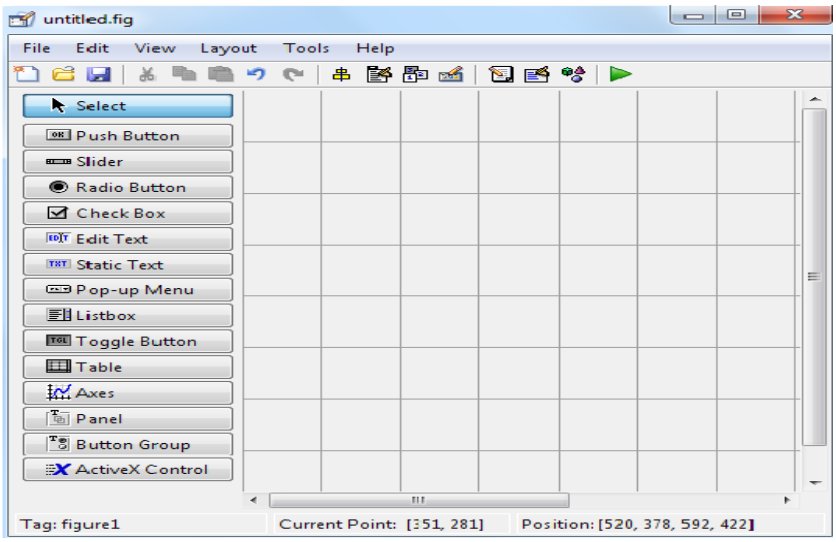


Figura 2.2. Área de diseño de una GUI.

2.1.2 Principales controles y opciones de GUIDE

Dentro de los controles y opciones de GUIDE mostrados en la figura 2.2, presentaremos a continuación las funciones de aquellos que se han empleado en la confección de nuestra GUI, además de otros utilizados frecuentemente.

Controles:

Push Button: Crea un botón que al presionarse ejecuta una acción por parte de MatLab.

Radio Button: Crea un botón de verificación *on/off*, que permite la ejecución de una determinada acción en dependencia del estado en que se encuentre el mismo.

Edit Text: Crea un campo de texto que permite al usuario introducir la cadena de caracteres deseada (es posible que esta cadena sea la representación de un vector o matriz determinado convertido a formato *string*[11]).

Static Text: Crea un determinado texto que permite exhibir símbolos, mensajes o incluso valores numéricos.

Axes: Crea un área donde se puedan introducir gráficas.

[11] Cadena de caracteres en inglés.

Panel: Crea un marco que define un área determinada donde se pueden introducir otros controles.

Opciones:

 *Align Objects*: Se utiliza para alinear de la forma deseada los componentes introducidos en la GUI.

 *Menu Editor*: Se utiliza para crear los menús deseados en la GUI.

 *M-file Editor*: Se utiliza para editar las funciones de los componentes en un archivo con extensión ".m".

 *Run*: Se utiliza para grabar y ejecutar.

 *Property Inspector*: Se utiliza para modificar las propiedades de los componentes insertados en una GUI.

2.1.3 Propiedades de los controles

Cada uno de los componentes utilizados en la confección de la interfaz gráfica, posee un conjunto de propiedades asociadas a las que se puede acceder y modificar mediante la utilización del "*Property Inspector"*; dentro de las más importantes podemos citar:

Eneable: Permite habilitar o deshabilitar un determinado control.

Visible: Permite que el control sea visible o no.

String: Es el texto que muestra el control, puede aparecer o no.

Tag: Permite agregar datos o identificar el control, es el nombre identificativo del control.

Callback: Permite ejecutar la función asociada al control una vez que este se active.

2.2 Generalidades de la aplicación DCRFtool

Antes de comenzar a describir el desarrollo de la interfaz gráfica DCRFtool debemos abordar algunos aspectos de carácter general relacionados con el *software*, que deben ser dominados por el usuario. Es válido señalar que esta interfaz tiene, como primer propósito, la facilitación del proceso de diseño de redes de compensación con el método de RF y por ende, la misma va destinada a un grupo de usuarios que obligatoriamente deben dominar los principales conceptos relacionados con el tema, es decir, está destinada al trabajo de estudiantes, profesores, técnicos y especialistas vinculados a la ingeniería de control.

Uno de los principales conceptos que debe dominar el usuario para lograr un satisfactorio aprovechamiento del *software* es el referente a la adecuada selección del método de compensación que ha de utilizarse ante determinada situación. Como es sabido, la práctica experimental nos ha llevado a conceptualizar algunos criterios respecto al comportamiento del sistema sin compensar, que nos permiten determinar el tipo de compensador que debe utilizarse, en un caso dado, para obtener los requerimientos deseados. El criterio principal que se analiza, para estos casos, es el comportamiento del sistema (en cuanto a su estabilidad en lazo cerrado) asociado a la pendiente que posee la curva de magnitud al realizarse el cruce por cero dB en el diagrama de Bode. Se puede afirmar que, por lo general, cuando en la traza de magnitud se observa un cruce largo por cero dB con pendiente de -40 dB/década, el método de compensación a utilizar es el de adelanto, por otro lado si el cruce es de igual pendiente pero con un tramo corto, se recomienda utilizar un compensador de atraso; en este segundo caso, si ocurre un cambio de pendiente a -60 dB/década muy cercano al 0 dB, es muy probable que el sistema deba compensarse a través del método de atraso-adelanto. Por otro lado, si lo que se produce es un cruce con pendiente -60 dB/década, también se recomienda la compensación de atraso-adelanto.

Debido a lo problemático que puede resultar el hecho de reconocer con qué pendiente la traza de magnitud de Bode efectúa el cruce por cero dB y la relatividad que deriva el análisis de cuándo puede ser largo o corto este cruce,

decidimos dejar a consideración del usuario la selección del tipo de compensador que debe utilizarse para lograr los propósitos deseados. No obstante, consideramos provechoso el hecho de que el usuario tome partida en la selección del tipo de compensador, dado que así, este puede efectuar el diseño utilizando cualquiera de los métodos y comparar por sí mismo cuál de ellos es el adecuado para suplir sus necesidades. Además, en caso de obtenerse los requerimientos deseados con más de un compensador, el usuario podría escoger aquel que posea mejor característica de respuesta transitoria en lazo cerrado.

Otro aspecto general que resulta de especial interés señalar en la interfaz, es el papel que desempeña el dato de margen de ganancia deseado en el diseño del compensador.

En el acápite destinado a la descripción de los pasos formulados por (Ogata 1998) para el diseño de cada uno de los compensadores, en el capítulo anterior, no se mencionó ningún procedimiento donde se tomara en cuenta el valor de margen de ganancia deseado. Sin embargo, en algunas ocasiones, los investigadores desean que el sistema cumpla con especificaciones de este tipo.

Nuestra interfaz, como se acordó al término del anterior capítulo, está confeccionada sobre la base de la metodología propuesta por (Ogata 1998), y por ende, en la misma solo se pretende lograr que el sistema compensado cumpla con los requerimientos de margen de fase y de constante de error estático deseados.

No obstante, es válido señalar que por lo general, cuando el diseñador posee experiencia, el valor de margen de ganancia que se desea alcanzar en un sistema, no se observa divorciado del valor de margen de fase deseado. A un diseñador experimentado se le hace ilógico hablar de estos términos como entes independientes, ya que existe una correlación entre ambos. En la mayoría de las ocasiones, cuando los valores deseados no son seleccionados al azar, sino que se pensó en la necesidad de la obtención de ambos a partir de un razonamiento lógico, con solo lograr que se cumpla el valor de margen de fase deseado, inferimos que se alcance el margen de ganancia deseado. A fin de cuentas, el hecho de desear que un sistema cumpla con requerimientos de este tipo, está

dado por la necesidad de que el mismo logre determinadas especificaciones en el comportamiento de su respuesta transitoria o simplemente alcance una buena estabilidad relativa, y ello es posible si el margen de ganancia posee un valor superior a 6 dB y el margen de fase se encuentra en el rango de 30° a 60°. Es la búsqueda de estos indicadores lo que nos motiva a diseñar los compensadores.

2.3 Componentes de la interfaz

La interfaz DCRFtool se encuentra conformada por dos ventanas principales y un conjunto de ventanas secundarias utilizadas para la comunicación con el usuario; díganse mensajes de error, avisos, preguntas y cuadros de diálogo.

La primera de las ventanas principales nombrada “DCRFtool”, constituye la portada de la interfaz. Se encuentra conformada por cuatro componentes, el primero de estos es un *axes* que ocupa toda el área de la ventana, utilizado para visualizar la imagen de fondo: un archivo con extensión “.jpg” nombrado “Portada” que se almacena en la misma carpeta donde se encuentran los demás archivos de la aplicación. Los restantes tres controles presentes en esta ventana son *push button* con *string* “Acerca de”, “Continuar” y “Salir” respectivamente.

El botón “Acerca de” permite, al ejecutarse, visualizar un mensaje donde se refleja la información relacionada con el diseñador del *software*. En tanto, el botón con *string* “Continuar” es el encargado de cerrar la ventana actual e iniciar la próxima ventana, mientras que el último botón es utilizado para cerrar la interfaz en caso de que el usuario no desee continuar.

La otra ventana principal de la interfaz (DCRFtoolAplic), considerada la más importante, ya que en ella se desarrolla la aplicación, es aquella a la que se accede una vez presionado el botón “Continuar” de la primera ventana. El área de la misma se encuentra segmentada por diferentes paneles que permiten agrupar los controles de acuerdo con su utilidad.

Un primer *panel*, mostrado en la figura 2.3, se destina a la selección del tipo de compensador que el usuario desee utilizar para la solución de su problema.

Figura 2.3. *Panel* 1 de la aplicación.

En tanto, un segundo *panel* (figura 2.4) es utilizado para agrupar los requerimientos de diseño, excepto el valor de constante de error estático, al cual se le asigna un *panel* individual (figura 2.5), donde no solo se insertará su valor numérico, sino que además se mostrará al usuario el valor de la ganancia del sistema ajustada para este requerimiento.

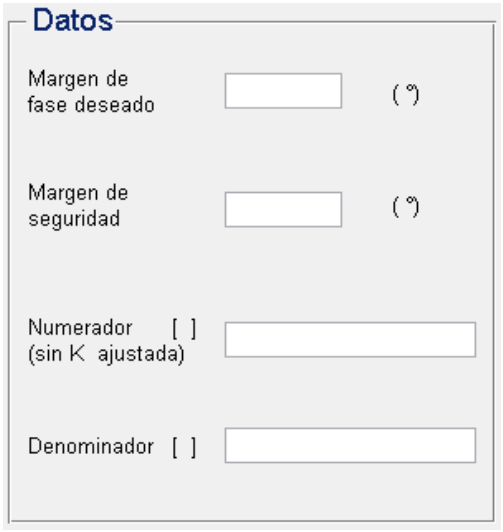


Figura 2.4. *Panel* 2 de la aplicación.

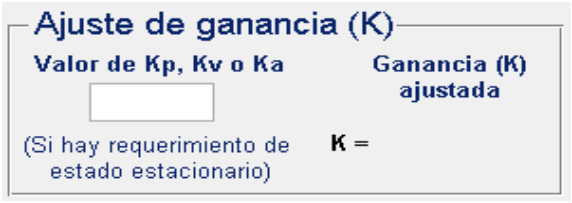


Figura 2.5 *Panel* 3 de la aplicación.

Una vez aplicado el programa, en otro *panel* se mostrarán el cero, el polo y la ganancia del compensador diseñado, además de la función transferencial del mismo, insertada a su vez en un pequeño *panel* interno. El conjunto es representado en la figura 2.6.

Figura 2.6. *Panel* 4 de la aplicación.

Los dos últimos paneles permiten visualizar las gráficas de Bode y la respuesta en lazo cerrado ante entrada paso unitario del sistema compensado.

Además de las secciones definidas por los paneles, esta ventana posee tres botones con *string* “Aplicar”, “Reset” y “Salir”. El primero diseñado para ejecutar la aplicación una vez insertados los datos necesarios, el segundo utilizado para reiniciarla y el último definido para abandonar el programa.

2.4 Paneles de la aplicación. Componentes y funcionalidad

A continuación, realizaremos una explicación más detallada de los componentes que conforman los distintos paneles de la ventana DCRFtoolAplic, así como la funcionalidad de cada uno de ellos.

2.4.1 *Panel* “Tipos de compensadores”

Como se puede apreciar en la figura 2.3, el primer *panel* nombrado “Tipos de compensadores”, destinado a la selección del método de compensación deseado por el usuario, agrupa en su interior tres controles de tipo *radio button*. Estos controles poseen como nombres identificativos “Adelanto”, “Atraso” y “Atraso-Adelanto” respectivamente.

Con el propósito de representar la función de selección del compensador, lo cual constituye nuestro principal objetivo en este *panel*, se estableció un algoritmo en cada uno de los *radio button* para deshabilitar los demás cuando uno de ellos haya sido seleccionado, ofreciendo al usuario la posibilidad de realizar la compensación únicamente por el método elegido. Además, se implementó una lógica que

permite, una vez seleccionado el compensador, habilitar todos los controles que posibilitan la inserción de los requisitos para el diseño por parte del usuario; mientras eso no acurra, y en vistas de evitar errores, el usuario no podrá insertar ningún dato a la aplicación.

Con la intención de ofrecer una mejor terminación al *software*, se desarrolló un método en cada *radio button* que posibilita reiniciar la aplicación cuando ninguno de ellos se encuentre seleccionado. Lo anterior, permite restablecer los valores iniciales en cada componente del programa en caso de que el usuario seleccione dos veces, de forma consecutiva, el mismo *radio button*.

Para devolver la aplicación a sus condiciones iniciales, se limpian primeramente todos los valores insertados en los *edit* y *static text* tanto de datos, como de respuestas. Además, se deshabilitan los *edit text* de datos, restableciendo, de esta forma, el estado para el que fueron concebidos en caso de no estar seleccionado ningún compensador. Para finalizar, con el uso de la función *newplot*, se limpian las gráficas de Bode y respuesta al paso insertadas en sendos *axes*.

Por último, con el propósito de facilitar al usuario el proceso de prueba a través de todos los métodos de compensación, cuando las especificaciones deseadas son las mismas, se desarrolló un algoritmo que posibilita mantener activos los valores de los datos insertados una vez seleccionado un *radio button*, si a continuación se selecciona otro.

2.4.2 *Panel* “Datos”

El *panel* representado en la figura 2.4 con nombre “Datos”, es el encargado de agrupar los controles donde se deben introducir los requisitos necesarios para el diseño del compensador. El mismo se encuentra conformado por cuatro *edit text*, que como ya conocemos, están destinados a la inserción de los valores necesarios en la aplicación. Dicho *panel* posee, además, seis *static text*. La función de cuatro de ellos es indicar al usuario las casillas reservadas para cada valor; los dos restantes representan las unidades de medidas en que deben ser introducidos los datos de margen de fase y margen de seguridad deseados.

Los cuatro *edit text* presentes en este *panel*, se encuentran reservados para la inserción de los dos datos mencionados anteriormente; además, incluyen los valores que representan a las matrices numerador y denominador del sistema que se desea compensar, siempre que en el mismo no se haya ajustado la ganancia ante el requerimiento de constante de error estático.

En el presente caso, los controles *edit text* del *panel* “Datos” se encuentran diseñados para capturar las cadenas de caracteres introducidas por el usuario, por lo que es válido resaltar que la inclusión de las mismas en el *software* se produce en formato *string*. Es por ello que para poder trabajar con estos valores en los cálculos efectuados, en el caso del margen de fase deseado y el margen de seguridad, se hizo necesaria la conversión a formato *double* a través de la función *str2double*[12]. Sin embargo, en el caso de los valores numerador y denominador, por tratarse de representación de matrices, se necesitó la conversión a formato *num* con el empleo de la función *str2num*[13].

2.4.2.1 Validaciones de los componentes del *panel* Datos

A continuación, se presentan las principales validaciones que se tuvieron en cuenta para diseñar los controles agrupados en el *panel* “Datos”:

- Los *edit text* destinados a la inserción de datos estarán deshabilitados y vacíos en el momento de inicializar la aplicación; solo podrán ser utilizados si se selecciona previamente el tipo de compensador que se desea diseñar.
- En los *edit text* solo se admitirán valores numéricos definidos entre los rangos permisibles de las respectivas variables. En caso de insertarse algún valor que no cumpla con estos parámetros, el usuario recibirá un aviso de error, e inmediatamente el valor erróneo desaparecerá para que este pueda introducir uno correcto.

[12] Convierte el *string* insertado a escalar siempre que sea posible.
[13] Convierte una matriz cadena a un arreglo numérico.

2.4.3 *Panel* "Ajuste de ganancia (K)"

Este *panel* se puede visualizar en la figura 2.5. El mismo agrupa en su interior cinco controles *static text*, además de un *edit text*. Como expresamos anteriormente, el componente *edit text* se encuentra diseñado para acoger el valor de constante de error estático que desee insertar el usuario, sin importar su naturaleza, es decir, se puede introducir tanto el valor de constante de posición, el de velocidad como el de aceleración.

Por su parte, los dos primeros *static text* son para mostrar al usuario que es en el espacio vacío definido por el componente *edit* donde debe ser introducido el dato y que la inserción de este no es obligatoria. Los otros dos *static text* mostrados en la mitad derecha del *panel* tienen como función representar el lugar donde se visualizará el valor de la ganancia ajustada del sistema, siempre que se introduzca el dato de constante de error estático. El último de los *static text* es invisible, la función del mismo es mostrar el valor de ganancia calculado por el *software*.

2.4.3.1 Validaciones de los componentes del *panel* "Ajuste de ganancia (K)"

- El *edit text* destinado a la introducción del dato de constante de error estático se encuentra validado para los mismos parámetros que sus homólogos del *panel* anterior. En estado inicial, y antes de que el usuario seleccione un compensador determinado, este control se comporta como un cuadro de inserción de texto vacío y deshabilitado. Además, el mismo está programado para recibir solamente valores numéricos mayores que cero, y en caso de insertar otros diferentes a este rango, se alertará al usuario del error cometido y la casilla quedará limpia.
- La inserción del valor de constante de error estático como dato es opcional: el *software* diseñado puede ejecutarse con o sin el mismo.

2.4.4 Botón "Aplicar". Validaciones y funcionalidad

El botón "Aplicar" es el encargado de ejecutar la aplicación una vez insertados los datos necesarios para el diseño del compensador deseado. A continuación se

muestran las principales validaciones tomadas en cuenta para el correcto funcionamiento de la aplicación.

2.4.4.1 Validaciones del botón “Aplicar”

- En caso de que el usuario ejecute dicho botón, sin antes haber seleccionado el tipo de compensador que se desea diseñar, se envía un mensaje de aviso explicando la necesidad de seleccionar previamente el tipo de compensador.
- Debido a que las constantes de error estático se definen para sistemas de tipo inferior o igual a dos, el *software* se encuentra validado solo para estos valores. En caso de introducirse un sistema de tipo superior a dos, al presionar el botón “Aplicar”, el usuario recibirá un aviso donde se le pide verificar si el valor del denominador insertado es correcto y se le explica la incompatibilidad del programa con estos sistemas.
- El diseño del compensador solo se logrará si los datos de margen de fase deseado y margen de seguridad insertados por el usuario, se encuentran en sus respectivos rangos permisibles. Además, debe cumplirse que las matrices numerador y denominador sean diferentes de cero y no estén vacías; por otra parte, el grado del denominador debe ser mayor que cero, y a su vez, mayor que el del numerador.
- Si se cumplen todas las especificaciones señaladas anteriormente, excepto que el grado del denominador sea menor que el del numerador o igual a cero, se alerta al usuario para que introduzca correctamente dichas matrices, y se le explican las características que debe poseer el denominador del sistema para lograr el diseño del compensador.
- Si el margen de fase del sistema sin compensar es superior al deseado, se envía un mensaje al usuario señalando que el sistema no necesita ser compensado.
- De no cumplirse alguna de las especificaciones necesarias para el diseño del compensador, se notificará al usuario que debe introducir correctamente

todos los datos; recordándole que solo el valor de constante de error estático es opcional.

- Si al compensarse el sistema se obtiene un margen de fase superior a 60°, se enviará un aviso al usuario señalando que el compensador requerido es irrealizable físicamente.
- En caso de que el margen de fase del sistema compensado se encuentre fuera de los límites del margen de fase deseado en ± 3° aparecerá un aviso al usuario donde se le indica que modifique el margen de seguridad para, de ser posible, alcanzar la especificación deseada.
- Si luego de presionar el botón Aplicar (habiendo seleccionado previamente el compensador de atraso-adelanto e insertado correctamente todos los datos necesarios para realizar el diseño), se verifica que en la traza de magnitud de Bode del sistema sin compensar no existe ninguna asíntota con pendiente de -60 dB/década antecedida por una con pendiente de -40 dB/década, se envía un mensaje al usuario donde se le explica la imposibilidad de compensar el sistema utilizando dicho método.

2.4.4.2 Funcionalidad del botón “Aplicar”

Luego de ejecutarse el botón “Aplicar” y verificarse que los datos fueron introducidos correctamente, el *software* procede a diseñar el compensador seleccionado. Primeramente, se ejecuta una subrutina que nos permite determinar la ganancia K ajustada a partir de la especificación del valor de constante de error estático insertado. Esta subrutina posee una lógica que posibilita ajustar, de forma correcta, la ganancia del sistema sin compensar, sin necesidad de que el usuario especifique la naturaleza del valor de la constante de error estático, es decir, si la constante es de posición, velocidad o aceleración. Esto se logra reconociendo previamente si el sistema es de tipo 0, tipo 1 o tipo 2.

El tipo de sistema está dado por la cantidad de columnas cero que se encuentren consecutivamente en la posición final de la matriz fila que representa al denominador del sistema sin compensar. Es por ello que primeramente se mide la cantidad de columnas de esta matriz a partir de la función [f2,c2]=size(d). Si la

última columna de la matriz posee un valor distinto de cero, estamos en presencia de un sistema de tipo 0. En caso contrario, si este valor es cero y además existen dos o más columnas y la penúltima es distinta de cero, el sistema es de tipo 1. Si la cantidad de columnas de la matriz es mayor que dos, las dos últimas son cero y la antepenúltima no lo es, el sistema es de tipo 2. Por último, en caso de que las tres últimas columnas posean valor cero y la cantidad de columnas de la matriz sea mayor que tres, nos encontramos en presencia de un sistema de tipo superior a dos.

Una vez determinado el tipo de sistema, es posible calcular el valor de K, despejando a partir de la expresión correspondiente a cada valor de constante de error estático. Las ecuaciones según el tipo de sistema se presentan a continuación:

Tipo 0: $\mathrm{K_p} = \lim_{s\to 0} \mathrm{KG(s)}$

Tipo 1: $\mathrm{K_p} = \lim_{s\to 0} \mathrm{SKG(s)}$

Tipo 2: $\mathrm{K_p} = \lim_{s\to 0} \mathrm{S^2KG(s)}$

Dado que no se encontró en MatLab ninguna función que nos permitiera determinar límites, hemos desarrollado una subrutina destinada a cada uno de los tipos de sistemas con el objetivo de solucionar este problema. A continuación mostramos solo un ejemplo representativo de dichas subrutinas:

```
% Sistema tipo 2
elseif c2 > 2 && isequal(d(1,c2),0) && isequal(d(1,c2-1),0) && const~=0 && ~isempty(n)
   d1=deconv(d,[1 0 0]);
   k= const/(polyval(n,0)/polyval(d1,0));
   set(handles.gank,'String',k);
```

En esta subrutina, primeramente se comprueba, como condición para ejecutar la sentencia, si el denominador posee más de dos columnas y si las dos últimas son cero; además, se verifica si los valores de constante de error estático y de numerador del sistema sin compensar fueron introducidos correctamente. En caso de cumplirse las condiciones anteriores, se procede a determinar la ganancia del

sistema. En este ejemplo específico, dado que nos encontramos ante un sistema de tipo 2, ajustamos la ganancia a partir de la constante de error estático de aceleración. Para ello es necesario multiplicar la función transferencial del sistema por S^2, lo que equivale a dividir el denominador de dicha función por ese mismo término. Este artificio nos brinda la posibilidad de eliminar el factor cuadrático que hace cero al denominador e indetermina la función transferencial del sistema sin compensar cuando se evalúa S en cero. Posteriormente, como se aprecia en la segunda línea de código de la sentencia, se despeja la ganancia K. El valor de la ganancia ajustada se envía, por último, al *static text* presente en el *panel* "Ajuste de ganancia (K)" destinado a la recepción del mismo.

Debido a que el valor de ganancia ajustada posee un carácter optativo en el diseño de un compensador determinado, se implementó un algoritmo que permite eliminar conflictos en el momento de redeterminar la función transferencial del sistema con la ganancia incorporada en caso de que el usuario no introduzca ningún valor de constante de error estático. Ante esta situación, se asume la ganancia K como un valor unitario, manteniendo, de esta forma, la función transferencial intacta, con el fin de no afectar el algoritmo vinculado al diseño del compensador.

Una vez determinada la ganancia del sistema sin compensar a partir de la especificación de constante de error estático, se comprueba si es necesaria la compensación del mismo, y en caso de serla se desarrolla en dependencia del tipo de compensador seleccionado.

En caso de seleccionarse el método de adelanto, aparece, como primer paso, una pequeña interfaz en forma de cuadro de diálogo nombrada "Procesando", que informa al usuario acerca del proceso de ejecución del programa. Posteriormente, se determina el ángulo de adelanto de fase necesario que se agregará al sistema, el factor de atenuación correspondiente a este ángulo y el valor de atenuación que debe ofrecer el compensador diseñado. El valor de margen de fase del sistema sin compensar con la incorporación de la ganancia ajustada, necesario en la

determinación de estos valores, ha sido calculado previamente con el fin de determinar si el sistema necesita ser compensado.

El próximo paso en el proceso de diseño del compensador, es determinar la frecuencia a la que se obtiene la atenuación calculada anteriormente. Para ello, se establece la siguiente subrutina:

```
frec=0.01:0.01:300;
[m,f,W]=bode(g,frec);
diferencia=10;
for i=1:length(m)
    diferenciaactual=abs(aten-m(i));
    if diferenciaactual<diferencia
       diferencia=diferenciaactual;
       ind=i;
    end
end
Wm=W(ind);
```

Como podemos observar, el primer paso en la determinación de la misma es establecer un rango abarcador de frecuencia con un intervalo de muestreo pequeño. Esto nos permitirá determinar, con la mayor exactitud posible, las trazas de Bode, que con la función [m,f,W]=bode(g,frec), quedarán expresadas a través de arreglos numéricos: los valores de magnitud, fase y frecuencia, proporcionados respectivamente por cada valor del rango especificado por *frec* quedan recogidos, finalmente, en tres arreglos. Esta lógica nos permite establecer, a partir de un valor dado, los restantes dos que contribuyen a formar las trazas en un punto determinado.

Para concluir, la presencia de un ciclo *for* nos brinda la posibilidad de conocer la iteración en la que se encuentra el valor de magnitud más próximo a la atenuación que el compensador debe aportar al sistema. Evaluando en el arreglo de las frecuencia el valor de iteración determinado anteriormente conoceremos, sino la frecuencia a la que el compensador aportará la atenuación deseada al sistema, al menos el valor más cercano a esta. A partir de este dato se establecen las frecuencias de esquina del cero y el polo del compensador de adelanto y se calcula la ganancia del mismo.

Finalmente, una vez determinados los valores necesarios para la construcción del compensador, los mismos son enviados a sus respectivos destinos en la interfaz donde serán visualizados por el usuario. Para ello, desarrollamos una subrutina que nos permite concatenar los términos “(S+”, frecuencia de esquina y “)” con el empleo de la función *strcat*. Esto nos posibilita expresar, adecuadamente, el cero y el polo del compensador. Es válido señalar que esta función solo permite enlazar términos expresados en formato *string*, por lo que previamente los valores de frecuencias de esquina son convertidos a este formato utilizando la función *double2str*. De esta forma concluye la subrutina destinada al diseño de la red de adelanto.

En caso de compensarse un sistema dado a través del método de atraso, se ejecutará un algoritmo similar al empleado en el método anterior. Al igual que en la red de adelanto, el primer paso consiste en proyectar la interfaz que informe al usuario del proceso de ejecución del programa. Seguidamente, se determina el punto de frecuencia en el que el ángulo de fase de la función transferencial de lazo abierto es igual a la suma de -180°, el margen de fase deseado y el margen de seguridad. Con este fin se emplea un bucle semejante al utilizado en la subrutina de la red de adelanto, solo que este tendrá como objetivo establecer la iteración donde se encuentre el valor, dentro del arreglo de las fases de Bode, más próximo al ángulo de fase determinado en el paso anterior. A partir del valor de iteración, se establece la frecuencia deseada (nueva frecuencia de cruce de ganancia), el valor de atenuación que debe proporcionar el compensador y el factor de atenuación (β) del mismo. Una vez que poseemos estos datos, se determinan las frecuencias de esquina del cero y el polo, además del valor de ganancia del compensador: resultados que son enviados al usuario a través de una subrutina similar a la empleada en la red de adelanto.

Por su parte, el algoritmo asociado al método de atraso-adelanto, es considerado el más complejo. Ello está dado no solo por lo complicado que puede resultar el desarrollo del mismo, sino además porque este, a diferencia de los anteriores, posee una subrutina que permite determinar previamente si la compensación con

el empleo de dicho método es viable. En el anexo V se muestra la subrutina utilizada.

Dicha subrutina, posee una lógica enfocada a la condición de que para lograr la compensación de un sistema con el método de atraso-adelanto, su traza de magnitud de Bode debe tener una asíntota con pendiente -60 dB/década antecedida por una con pendiente de -40 dB/década.

Como se puede apreciar, el primer paso en el desarrollo de la subrutina es determinar los vectores cero y polo del sistema sin compensar. Seguidamente, con el empleo de la función *damp*, se establecen los vectores frecuencias de esquina, tanto del cero y el polo por separados, como de ambos unidos (*Wesq*[14]). A través de la función *sort*, estos vectores se organizan en orden ascendente. En el caso específico del vector *Wesq*, con la utilización del código *unique*, hacemos posible la selección de un solo valor entre las frecuencias de esquinas que se repiten. Posteriormente, un bucle que recorre todos los valores del vector *Wesq*, permite determinar, en cada iteración, cuántos valores de los vectores cero y polo coinciden con el valor de *Wesq* en la iteración analizada. Es entonces, con el empleo de la expresión matemática planteada a continuación, que se puede establecer la pendiente que caracterizará la curva de magnitud de Bode desde la frecuencia de esquina analizada hasta la próxima:

pend = pend +20*fa -20*fb

fa: Cantidad de valores de frecuencia de esquina del vector cero que coinciden con el valor de *Wesq* analizado.

fb: Cantidad de valores de frecuencia de esquina del vector polo que coinciden con el valor de *Wesq* analizado.

Como se aprecia en la expresión, al igual que en la construcción de la traza de magnitud de Bode, la variable *pend* es acumulativa: siempre, el último valor determinado, se sumará al que se esté calculando. En tanto, estos valores se diferenciarán dependiendo de la cantidad de factores que aporten los vectores

[14] Vector conformado por las frecuencias de esquinas del cero y el polo del sistema sin compensar.

cero y polo en la frecuencia de esquina analizada: cada factor presente en el vector cero tributará a aumentar la pendiente en 20 dB/década, mientras que cada factor del vector polo contribuirá a disminuirla en la misma cantidad.

Para concluir la subrutina, establecimos una lógica que permite verificar, en cada frecuencia de esquina, si se cumple la condición necesaria para desarrollar la compensación a través del método analizado. En caso de que en una iteración determinada la misma se cumpla, se abandonará el ciclo y procederá a desarrollarse el diseño del compensador. En tanto, si se recorre el ciclo completo y la condición no se cumple, el programa finalizará notificando al usuario la imposibilidad de realizar la compensación del sistema a través de dicho método.

Dado el caso de que sea posible compensar el sistema a través del método de atraso-adelanto, se procede a ejecutar una subrutina diseñada para determinar cuál de las frecuencias que conforman el tramo de recta de magnitud de Bode con pendiente de -40 dB/década es la adecuada para cumplir las especificaciones deseadas. Para ello, establecimos una lógica que permite, a través de la utilización de un ciclo *for*, realizar el diseño para todas aquellas frecuencias que pertenezcan al rango definido por los valores desde la frecuencia de esquina donde comienza el tramo hasta la próxima, siempre y cuando estas aumenten en un valor de 0.01. Una vez concluido el ciclo, escogemos aquel compensador que, a partir de determinado valor de frecuencia, haya proporcionado un margen de fase más próximo al requerido.

Como hemos procedido en los anteriores métodos, para realizar la compensación, se presenta, en un primer momento, la interfaz “Procesando”, destinada a informar al usuario sobre la ejecución del proceso. Luego, a partir de la frecuencia de cruce de ganancia que corresponda a la iteración analizada, se establece la frecuencia de esquina del cero de la parte de atraso. Además, se calcula el factor de atenuación de dicha parte y, con este valor, la respectiva frecuencia de esquina del polo. En tanto, las frecuencias de esquina de la parte de adelanto del compensador, se determinan a través de los interceptos de la recta de pendiente 20 dB/década que pasa por el punto (nueva frecuencia de cruce; - magnitud a

dicha frecuencia) con la recta cero dB y la recta definida por la zona de altas frecuencias de la red de atraso respectivamente. La recta de pendiente 20 dB/década que atraviesa el punto mencionado responde a la siguiente ecuación:

$$\mathrm{y} = 20\log 10(\mathrm{W}) - 20\log 10(\mathrm{Wcnr}) - 20\log 10(\mathrm{mWcnr})$$

y: Valor de magnitud de la recta en un punto determinado.

W: Valor de frecuencia de la recta en un punto determinado.

Wcnr: Frecuencia de cruce de ganancia.

mWcnr: Magnitud correspondiente a la frecuencia de cruce de ganancia.

A partir de esta expresión, evaluando la magnitud de la recta en cero, podemos conocer la frecuencia a la que se produce el cruce por la recta de cero dB, o sea, la frecuencia de esquina del polo de la parte de adelanto. En tanto, la frecuencia de esquina del cero se obtiene de dividir la anterior entre el factor de atenuación calculado.

Una vez diseñados los compensadores correspondientes a cada una de las frecuencias de cruces de ganancia, y luego de seleccionar el más apropiado, presentamos una subrutina similar a la utilizada en los métodos anteriores para que el usuario logre visualizar los resultados obtenidos. De esta manera, hemos planteado el algoritmo correspondiente al diseño de nuestro compensador de atraso-adelanto.

Independientemente del tipo de compensador seleccionado, una vez concluido el algoritmo de diseño del mismo, se procede a ejecutar una pequeña subrutina que posibilita cerrar la interfaz “Procesando”, además de enviar las gráficas de Bode del sistema compensado y respuesta al paso del sistema compensado en lazo cerrado a sus respectivos destinos. Para finalizar, se comprueban las validaciones referidas a los resultados obtenidos.

2.4.5 Paneles destinados a la visualización de los resultados

Como podemos apreciar en la figura 2.6, el *panel* con *string* “Compensador diseñado” está compuesto por tres *static text* nombrados “Cero”, “Polo” y

"Ganancia" respectivamente, a los que corresponden tres *edit text* destinados a la visualización de los datos referidos con anterioridad. Dichos *edit text*, se encuentran deshabilitados de manera permanente para imposibilitar al usuario la inserción de cadenas no deseadas, puesto que el objetivo para el que fueron creados es únicamente mostrar los resultados obtenidos. El *panel* "Compensador diseñado" posee en su interior, a su vez, otro más pequeño conformado por cuatro *static text* que posibilitan la representación final de la función transferencial del compensador diseñado. Estos, por su parte, fueron elaborados con *string* vacíos para que su apariencia sea invisible. En tanto, un quinto *static*, con *string* "Gc(s)=", es el encargado de señalar el lugar donde se introducirá la función una vez ejecutada la aplicación.

Por su parte, como hemos mencionado anteriormente, los gráficos obtenidos a partir de las trazas de Bode del sistema compensado y la respuesta ante entrada paso del mismo sistema en lazo cerrado, son insertados en respectivos *axes* del *panel* "Sistema compensado". Además, se realizaron una serie de ajustes en las propiedades de los *axes* con el propósito de brindar una mejor apariencia a la aplicación.

2.4.6 Botones "Reset" y "Salir"

Las acciones de reiniciar y salir, imprescindibles para muchas aplicaciones, han sido implementadas a partir de dos botones destinados, respectivamente, a cada una de ellas.

En el caso del botón "Reset", desarrollamos una subrutina que nos posibilita devolver, a las condiciones iniciales, todos los componentes que conforman la aplicación.

Para devolver la aplicación a sus condiciones iniciales, el *software* realiza un procedimiento similar al descrito anteriormente, en caso de que el usuario hubiese seleccionado, dos veces, y de manera consecutiva, el mismo tipo de compensador. En un primer momento, se limpian todos los valores insertados en los *edit* y *static text* tanto de datos, como de respuestas. Además, se deshabilitan los *edit text* de datos, restableciendo, de esta manera, la posición inicial. Por

último, se limpian las gráficas de Bode y respuesta al paso insertadas en los respectivos *axes*.

En el caso del botón “Salir”, se desarrolló un algoritmo para dar al usuario la posibilidad de verificar si realmente desea salir de la aplicación, evitando abandonar el programa en caso de que se haya ejecutado esta acción por error. Para ello, aparece un mensaje en forma de pregunta donde el usuario confirmará la salida o no del programa, dependiendo, esto último, de la respuesta que él mismo seleccione. El algoritmo descrito anteriormente, es el mismo que se empleó, para cerrar el programa, a través del botón que aparece por defecto, con dicha función, en el extremo superior derecho de cualquier ventana de *Windows*.

2.4.7 Menú “Ayuda”

Con el propósito de facilitar al usuario la utilización de la interfaz, desarrollamos además un menú con nombre “Ayuda”, encargado de representar los pasos que se deben seguir para lograr un aprovechamiento satisfactorio del *software*.

Una vez que el usuario seleccione el menú, se presentará un archivo con extensión “.pdf” nombrado “Ayuda”. Este archivo contiene las principales instrucciones para el trabajo con la aplicación y está almacenado en la carpeta donde se encuentran los demás archivos del programa.

2.5 Consideraciones finales del capítulo

Luego de haber abordado los elementos pertinentes en la realización del presente capítulo, consideramos que para el diseño de nuestra aplicación, es relevante el empleo de la herramienta GUIDE, de MatLab, puesto que la misma posee una amplia gama de funciones y controles con sus respectivas propiedades que facilitan la elaboración de esta y otras interfaces gráficas. Además, de nuestro análisis se trasluce la necesidad de que los usuarios posean un conocimiento de los temas que han sido abordados en este capítulo de manera general, y que constituyen aspectos fundamentales para llegar a dominar, eficazmente, el *software*.

CAPÍTULO 3 ANÁLISIS DE LOS RESULTADOS

Una vez descrita la estructura de la aplicación DCRFtool y el algoritmo empleado en la realización de la misma, en el presente capítulo procedemos a desarrollar una serie de pruebas con el objetivo de demostrar el buen desempeño del *software* y la correcta programación implícita en el proceso de diseño de redes de compensación con el empleo del método de RF. Con este propósito, se tomará un ejemplo tipo para cada uno de los compensadores que nos permita establecer una comparación entre los resultados obtenidos por el método convencional y los alcanzados con la implementación de la aplicación.

3.1 Validación del compensador de Adelanto

Con el propósito de validar la parte de la aplicación destinada al diseño del compensador de adelanto, realizaremos el análisis de un ejemplo tipo propuesto por (Ogata 1998), que nos permitirá comparar los resultados obtenidos a partir del desarrollo propuesto por dicho autor con los alcanzados a través del *software*. El ejemplo se muestra a continuación:

Considerando el sistema con función transferencial en lazo abierto:

$$G(s) = \frac{4}{s(s+2)}$$

Se quiere diseñar un compensador de modo que la constante de error estático de velocidad (Kv) sea de 20 seg^{-1}, el margen de fase sea de 50° y el margen de ganancia sea al menos de 10 dB.

Aplicando el método tradicional, el autor obtiene la siguiente función transferencial para el compensador de adelanto:

$$G_c(s) = 41{,}7\frac{(s + 4{,}43)}{(s + 18{,}4)}$$

Con la construcción de las trazas de Bode del sistema compensado, como se observa en la figura 3.1, los márgenes de fase y de ganancia son de cerca de 50° y +∞ dB, respectivamente. De esta manera, el sistema compensado cumple tanto con el requerimiento en estado estable como con los de la estabilidad relativa.

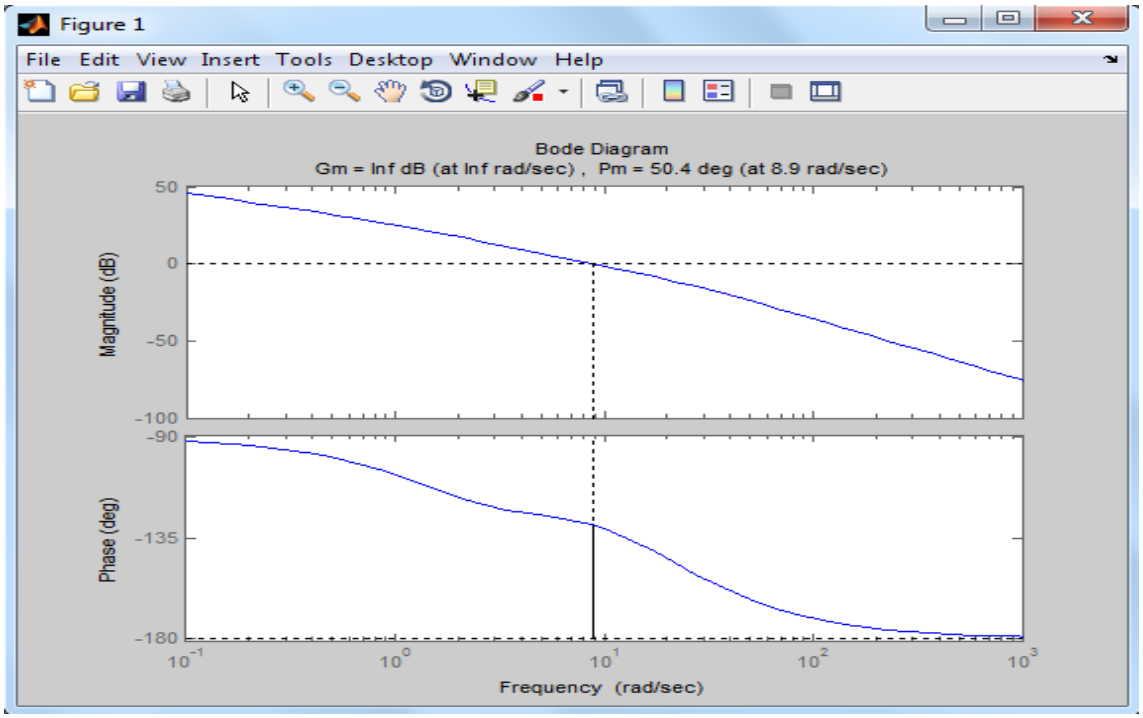


Figura 3.1. Trazas de Bode del sistema compensado con la red de adelanto.

Además, en la figura 3.2 se muestra la respuesta que experimenta el sistema compensado ante una entrada paso unitario como señal de prueba para medir el satisfactorio desempeño del sistema en respuesta a las entradas reales. En la figura se aprecia, según los parámetros de máximo sobreimpulso, tiempo de asentamiento y tiempo de subida, que definen la respuesta transitoria y la respuesta en estado estable, el correcto comportamiento dinámico del sistema.

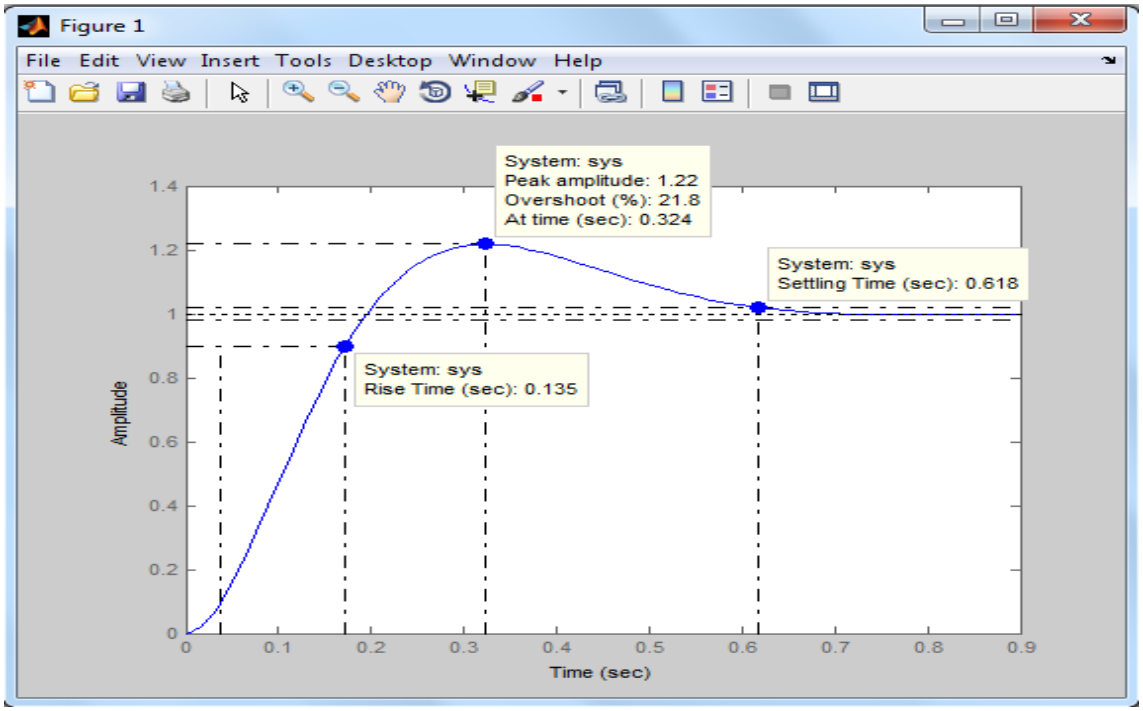


Figura 3.2. Respuesta al paso del sistema compensado con la red de adelanto.

En tanto, las respuestas alcanzadas con la utilización de la aplicación DCRFtool se muestran en las figuras 3.3 y 3.4.

Figura 3.3. Función transferencial del compensador de adelanto diseñado mediante el DCRFtool.

Como se aprecia en esta figura, con la utilización del *software*, se obtiene una función de transferencia muy similar a la alcanzada por el autor; solo diferenciadas por algunas aproximaciones realizadas en el diseño mediante la vía tradicional.

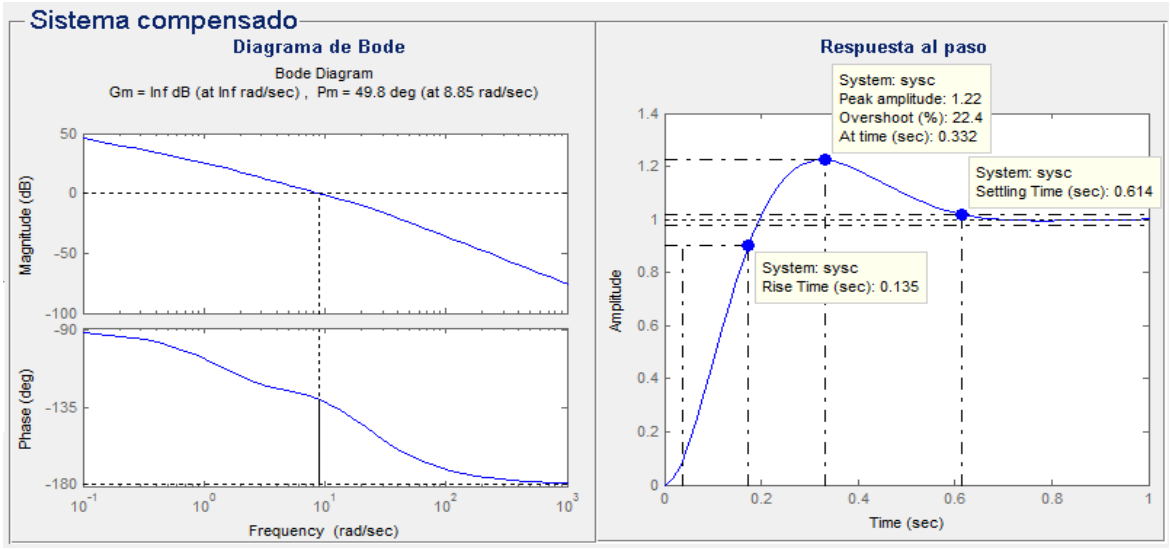


Figura 3.4. Diagrama de Bode y respuesta al paso del sistema compensado con la red de adelanto mediante el DCRFtool.

Por su parte, las características en el dominio del tiempo y la frecuencia permanecen prácticamente constantes respecto a las alcanzadas por el autor. Ello se debe a que la diferencia mostrada anteriormente en las funciones transferenciales obtenidas por ambas vías es ínfima.

3.2 Validación del compensador de Atraso

En el caso de la red de atraso, con el propósito de establecer la comparación entre los resultados obtenidos por ambas vías para un ejemplo dado, tomamos un ejercicio tipo, presente en las clases prácticas de la asignatura Ingeniería de Control I. No seleccionamos el ejemplo desarrollado por (Ogata 1998) para este compensador, debido a que en el análisis planteado por el autor para el desarrollo del ejercicio, se establecieron algunas consideraciones fuera de la metodología de diseño que provocaron resultados no esperados. El ejemplo seleccionado se presenta a continuación:

Considerando el sistema con función transferencial en lazo abierto:

$$G(s) = \frac{K}{s(s+1)(s+5)}$$

Determine el compensador que garantice un valor de constante de error estático de velocidad (Kv) de 3 seg^{-1} y margen de fase de 38°.

Utilizando la vía tradicional, la función transferencial del compensador de atraso diseñado queda expresada como:

$$G_c(s) = 5{,}04\,\frac{(s + 0{,}078)}{(s + 0{,}0262)}$$

A partir de las trazas de Bode del sistema compensado (figura 3.5), podemos comprobar que el mismo cumple con el requerimiento de margen de fase deseado.

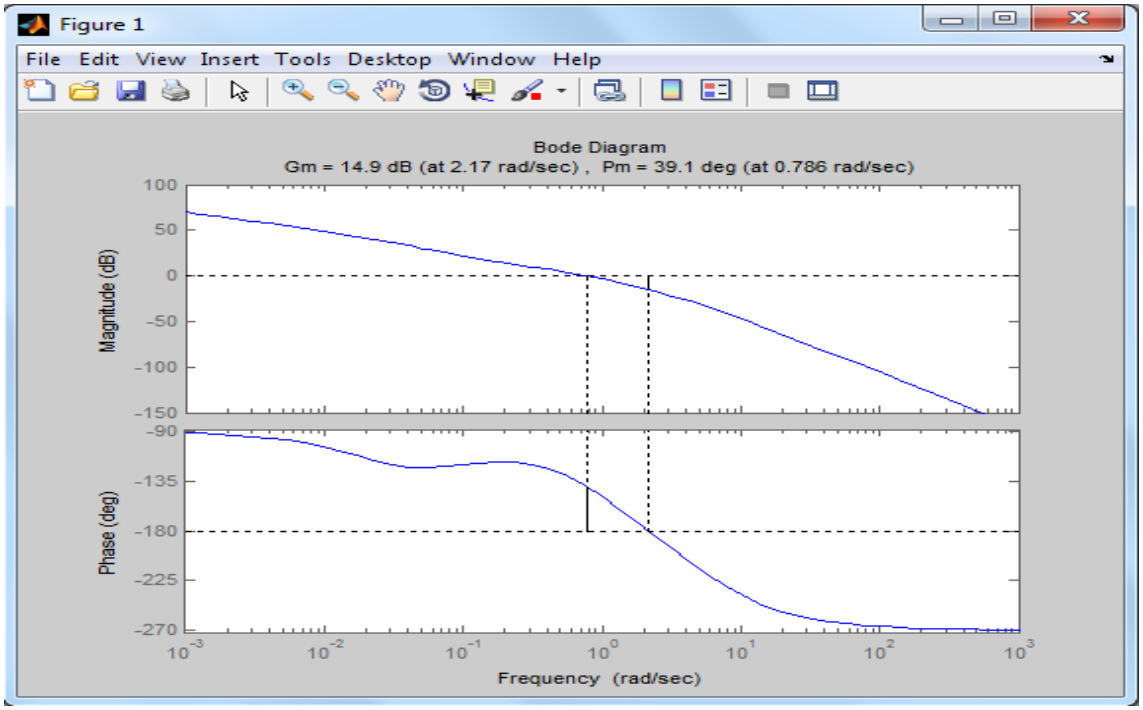


Figura 3.5. Trazas de Bode del sistema compensado con la red de atraso.

En tanto, las características del dominio del tiempo que definen el comportamiento dinámico del sistema, se muestran en la figura 3.6, a partir de la respuesta al paso del sistema en lazo cerrado con la inclusión del compensador.

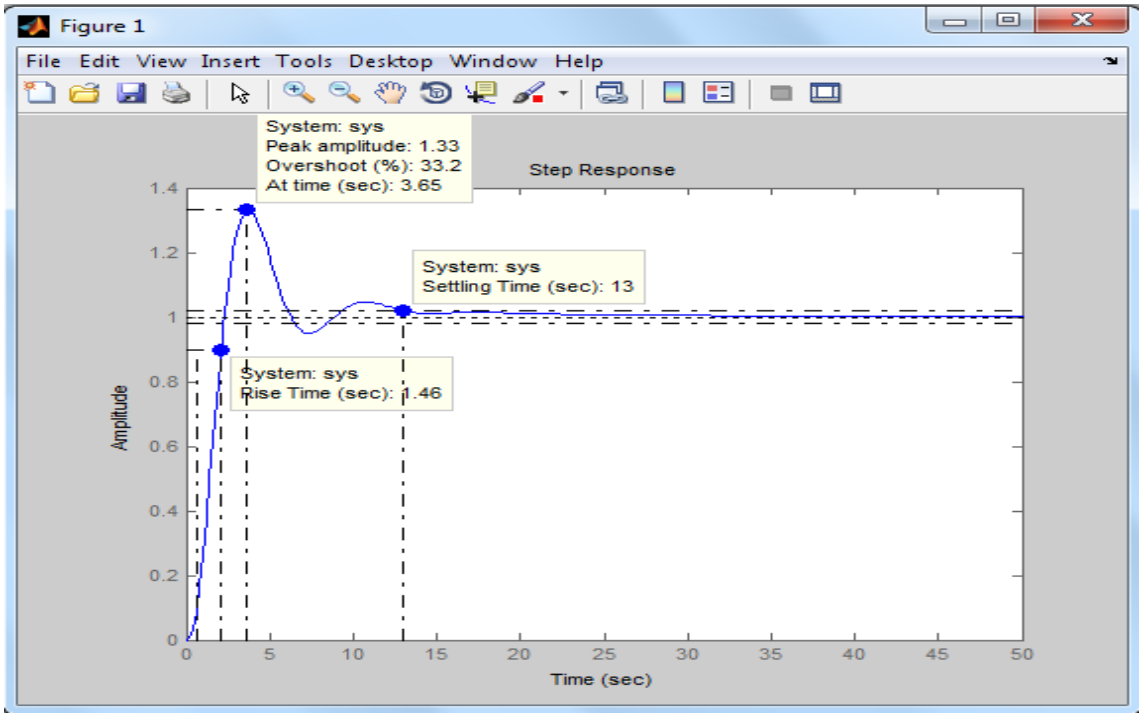


Figura 3.6. Respuesta al paso del sistema compensado con la red de atraso.

Por otra parte, con el empleo de la aplicación DCRFtool, como apreciamos en la figura 3.7, se obtiene una función transferencial muy similar a la alcanzada a través de la vía tradicional. No obstante, la obtenida con el *software* resulta más exacta, pues con el uso del método tradicional se incurre en errores a partir de las constantes búsquedas sobre gráficos realizadas para lograr el diseño del compensador.

Figura 3.7. Función transferencial del compensador de atraso diseñado mediante el DCRFtool.

Además, en la figura 3.8, con el propósito de demostrar la efectividad de la aplicación para la sección correspondiente al compensador de atraso, se presentan los gráficos de Bode y respuesta al paso del sistema compensado a través de esta red.

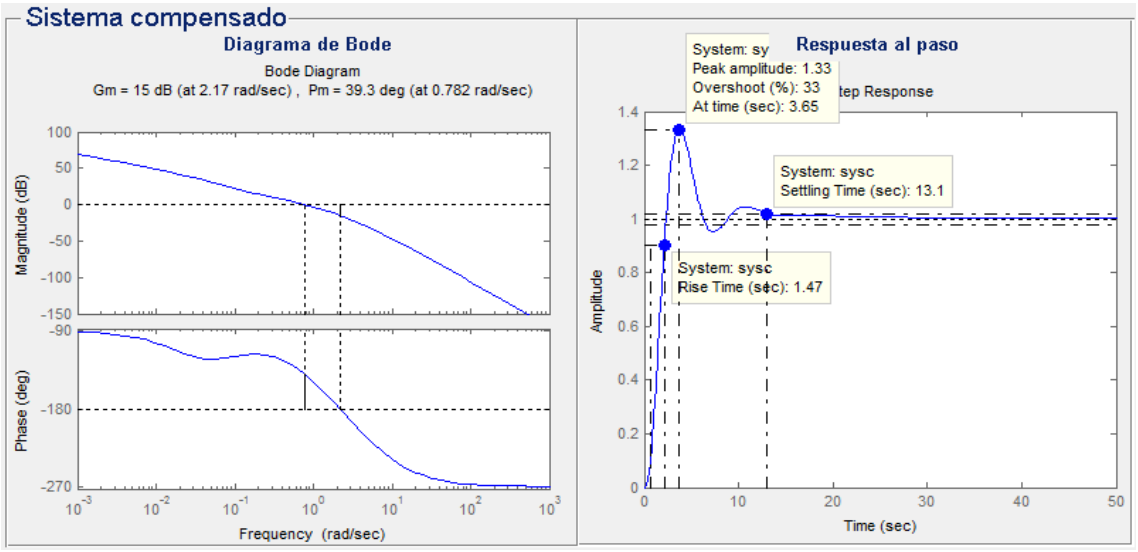


Figura 3.8. Diagrama de Bode y respuesta al paso del sistema compensado con la red de atraso mediante el DCRFtool.

3.3 Validación del compensador de Atraso-Adelanto

Por último, a fin de validar la parte de la aplicación destinada al diseño del compensador de atraso-adelanto, hemos realizado un estudiado del ejemplo tipo que presenta (Ogata 1998) en esta sección, con el propósito de establecer una comparación entre los resultados alcanzados por el autor y los que se derivan del empleo del *software* DCRFtool. A continuación se muestra el ejemplo:

Considerando el sistema con realimentación unitaria cuya función transferencial en lazo abierto es:

$$G(s) = \frac{K}{s(s+1)(s+2)}$$

Se quiere que la constante de error estático de velocidad sea de 10 seg^{-1}, que el margen de fase sea de 50° y que el margen de ganancia sea de 10 dB o más.

Solucionando este ejemplo por el método manual, la función transferencial del compensador de atraso-adelanto queda expresada de la siguiente manera:

$$G_c(s) = \frac{(s + 0{,}7)(s + 0{,}15)}{(s + 7)(s + 0{,}015)}$$

Como se puede apreciar en la figura 3.9, debido a una serie de aproximaciones realizadas por el autor, el requerimiento de margen de fase no es exactamente el deseado.

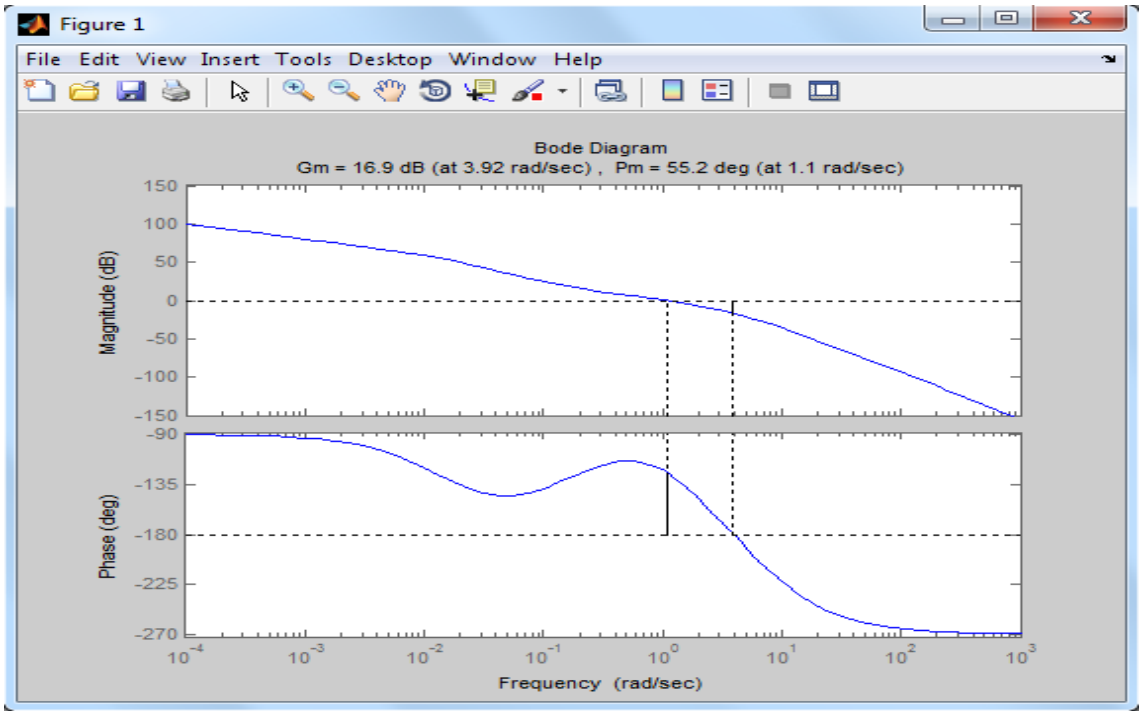


Figura 3.9. Trazas de Bode del sistema compensado con la red de atraso-adelanto.

Además, en la figura 3.10, con el propósito de establecer una comparación entre las dos vías de realización, se presenta la respuesta al paso del sistema compensado en lazo cerrado.

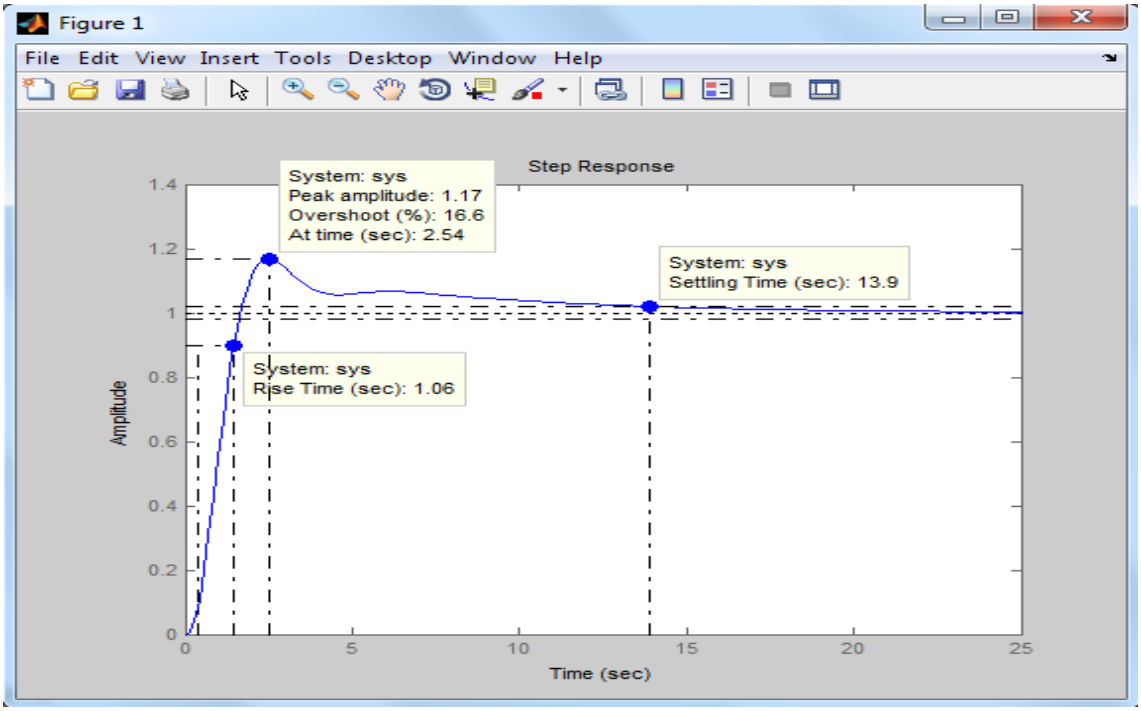


Figura 3.10. Respuesta al paso del sistema compensado con la red de atraso-adelanto.

Sin embargo, con la utilización de la aplicación DCRFtool, obtenemos un resultado más exacto. Ello está dado por la lógica empleada en el proceso de diseño del compensador, que permite determinar cuál de las frecuencias que conforman el tramo de recta de magnitud de Bode con pendiente de -40 dB/década, es la adecuada para cumplir las especificaciones deseadas.

La función transferencial obtenida a partir del *software* para la resolución de este ejemplo es mostrada en la figura 3.11.

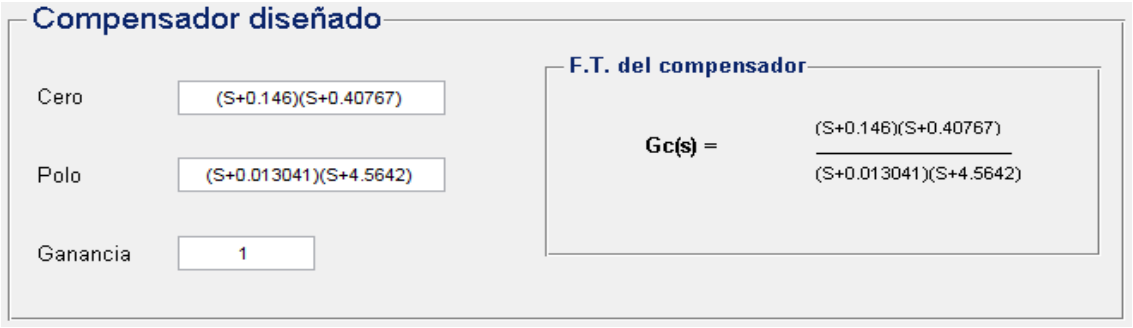


Figura 3.11. Función transferencial del compensador de atraso-adelanto diseñado mediante el DCRFtool.

En tanto, la figura 3.12 presenta la respuesta al paso y el diagrama de Bode que ofrece la aplicación para este ejemplo.

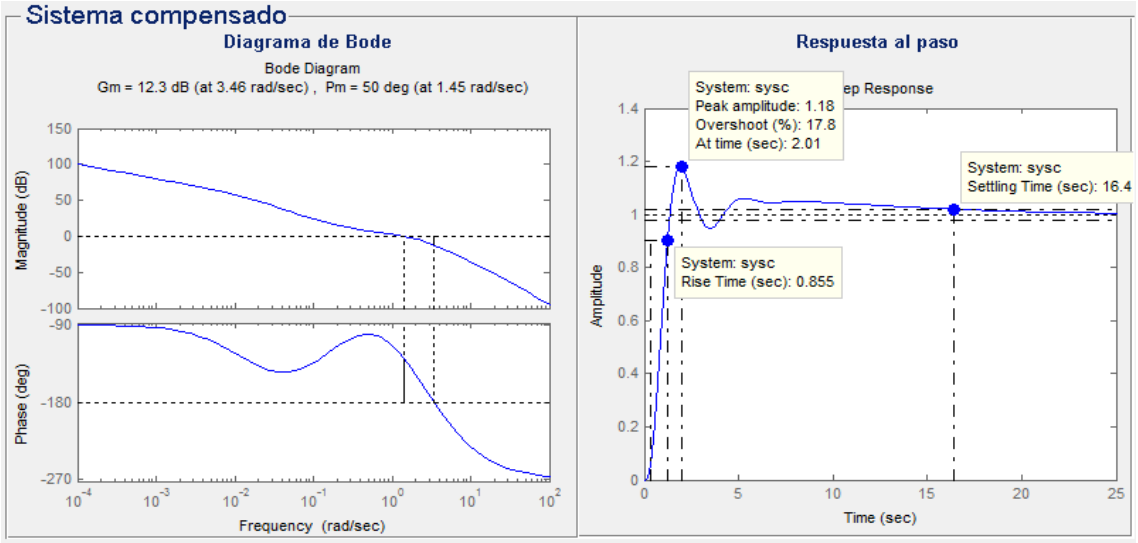


Figura 3.12. Diagrama de Bode y respuesta al paso del sistema compensado con la red de atraso-adelanto mediante el DCRFtool.

3.4 Consideraciones finales del capítulo

Llegando a este punto, podemos decir que cada una de las pruebas aplicadas con el propósito de establecer una comparación entre las dos vías analizadas en el presente capítulo para el diseño de compensadores a través del método de RF, resultaron elementos demostrativos del correcto funcionamiento de la aplicación DCRFtool y el algoritmo implementado en ella.

En otro sentido, es necesario señalar la superioridad de nuestro *software* respecto a la vía manual con relación a la exactitud en la respuesta ofrecida y la reducción del tiempo empleado para el desarrollo del proceso.

Unido a esto, podemos agregar, que durante el diseño de la interfaz se ha tenido en cuenta el aspecto visual de la misma para favorecer la buena percepción de la información y las acciones disponibles por parte de los usuarios.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Después de realizar un análisis detallado de cada uno de los resultados obtenidos en esta investigación, así como de su aplicación práctica y otros elementos significativos para el desarrollo de la misma, arribamos a las siguientes conclusiones:

- El DCRFtool es una herramienta efectiva para el diseño de compensadores, puesto que simplifica y hace más viable el trabajo de los usuarios en este sentido: dado que la interfaz reduce el tiempo de diseño del compensador deseado y garantiza la efectividad del mismo.
- La interfaz gráfica DCRFtool presenta una estructura amigable que facilita su uso durante el diseño de redes de compensación por parte de los usuarios, sin necesidad de que estos posean conocimiento alguno de los pasos implícitos en tal proceso.
- La validación del *software* DCRFtool fue efectiva para demostrar el correcto funcionamiento de la aplicación y cada uno de sus controles a partir del algoritmo empleado.

Recomendaciones

Como seguimiento a nuestro tema de investigación, proponemos las siguientes recomendaciones:

- Continuar el estudio de la herramienta GUIDE de MatLab para lograr otras versiones de la interfaz DCRFtool, con la inclusión de nuevas opciones y mejores algoritmos de programación.
- Utilizar la herramienta DCRFtool en experimentos reales que precisen del empleo de redes de compensación.
- Perfeccionar el aspecto visual de la aplicación empleando animaciones interactivas, para amenizar, de tal forma, el tiempo de espera del usuario durante la ejecución de la misma.

REFERENCIAS BIBLIOGRÁFICAS

Babylon. (2012). "Definición de interfaz." Disponible en: http://dictianary.babilon.com/interfaz.

Barragán, D. O. (2008) "Manual de interfaz gráfica de usuario en MatLab."

Calva, M. E. (2008). Desarrollo de una interfaz gráfica para un electrocardiógrafo portatil Tesis de grado, Universidad Autónoma Metropolitana Iztalapa.

Carballo, C. R. (2011). Interfaz gráfica para la identificación para la plataforma de 2 grados de libertad Tesis de grado, Universidad Central "Marta Abreu" de Las Villas.

DeMoyer, R. and E. E. Mitchel (2002). Use Matlab graphical user interface development environment for some control system applications.

Escribano, J. F. (2009). Desarrollo de una interfaz gráfica en Matlab para la aplicación de modelos de Regreción Local Polinómica Tesis de grado, Universidad Carlos III Madrid

Esqueda, J. J. (2002). Matlab e interfaces gráficas. CONATEC 2002, Instituto Tecnológico de Ciudad Madero, México.

Francisco, M. (2000). Introducción a la OOP, Grupo EIDOS.

Kuo, B. C. (1996). Sistemas de Control Automático. Prentice-Hall Hispanoamericana, México.

López, E. (2009). Interfaz Genérica para Sistemas de Adquisición basada en Software Libre Tesis de grado, UCLV.

Mathworks. (2007). "GUIDE toolbox." from http://www.mathworks.com/help/techdoc/matlab_product_page.html.

Ogata, K. (1998). Ingeniería de Control Moderna. México, Prentice-Hall Hispanoamericana.

Ong, E. K. and F. L. Tan (2000). "Menu driven graphical interface for Matlab Control Design." International Journal Engineering Education.

Rodríguez, S. (2011). Evaluación de dosis radiológicas mediante técnicas de procesamiento digital de imágenes Tesis de grado, Universidad Central "Marta Abreu" de Las Villas.

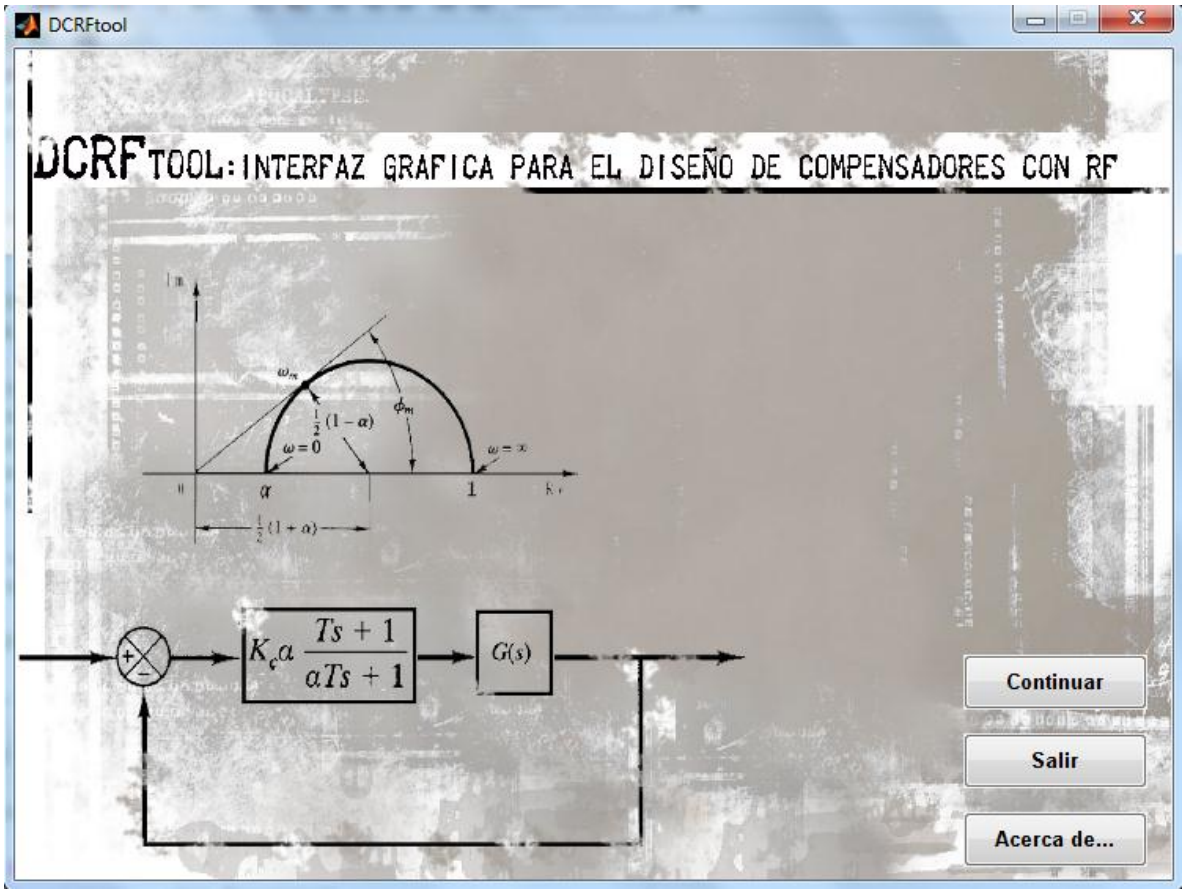
Smith, C. A. and A. B. Corripio (1991). Control Automático de Procesos. Teoría y Práctica. . México, Editorial Limusa.

Valencia, P. (2010). Desarrollo de una interfaz para el control estadístico de procesos utilizando herramientas de Matlab Tesis de grado, Universidad tecnológica de la Mixteca.

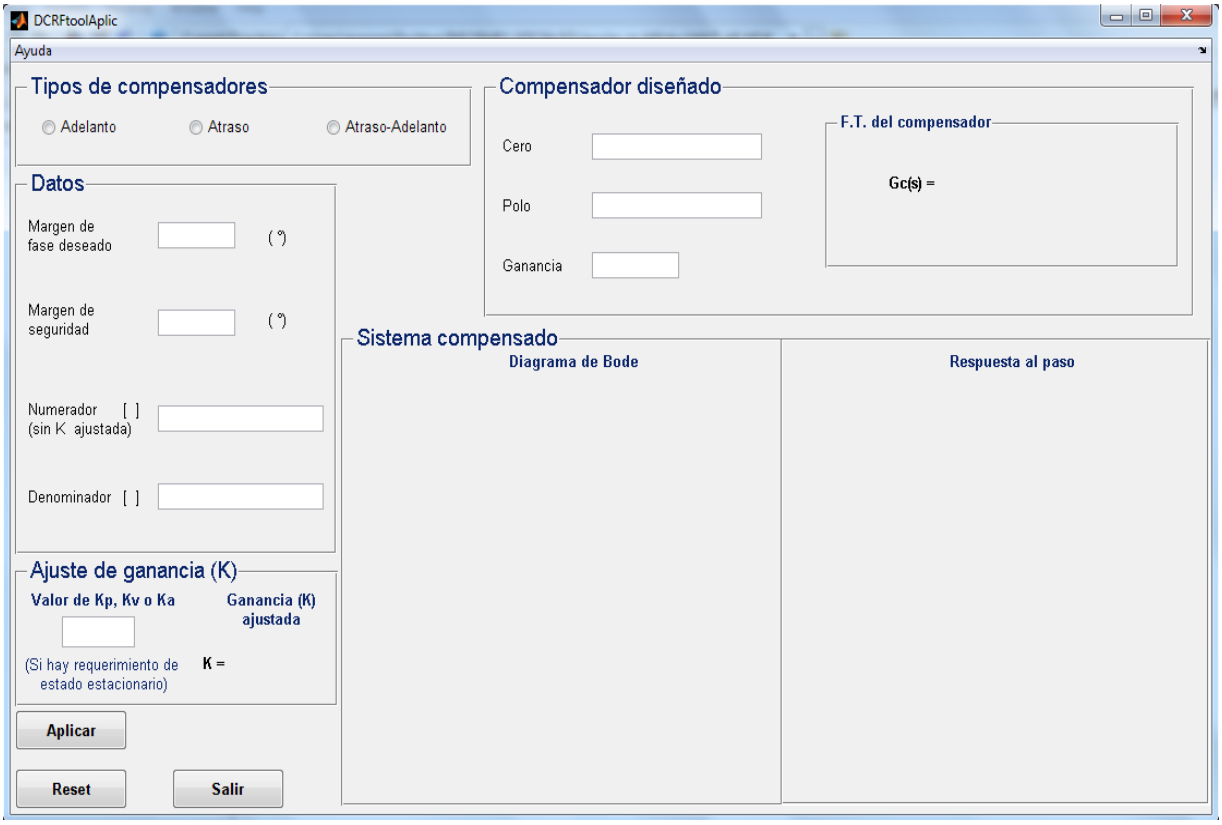
Valeriano, Y., H. J. Oria, et al. (2012). VASmodel. Interfaz gráfica para modelo dinámico de un AUV. VI Conferencia UCIENCIA 2012, Universidad de Ciencias Informáticas, Habana, Cuba.

ANEXOS

Anexo I: Portada de la interfaz DCRFtool



Anexo II: Ventana principal de la aplicación DCRFtool



Anexo III: Código de los *radio button* del *panel* "Tipo de compensador"

- Código del *radio button* "Adelanto".

```
clc;
if ~get(hObject,'Value') && ~get(findobj('Tag','radiobutton2'),'Value')
&& ~get(findobj('Tag','radiobutton3'),'Value')
   set(findobj('Tag','num'),'String','');
   set(findobj('Tag','den'),'String','');
   set(findobj('Tag','mfdeseado'),'String','');
   set(findobj('Tag','mseg'),'String','');
   set(findobj('Tag','constante'),'String','');
   set(findobj('Tag','gank'),'String','');
   set(findobj('Tag','cero'),'String','');
   set(findobj('Tag','polo'),'String','');
   set(findobj('Tag','ganancia'),'String','');
   set(findobj('Tag','cc'),'String','');
   set(findobj('Tag','pc'),'String','');
   set(findobj('Tag','kc'),'String','');
   set(findobj('Tag','raya'),'String','');
   set(findobj('Tag','mfdeseado'),'Enable','inactive');
   set(findobj('Tag','mseg'),'Enable','inactive');
   set(findobj('Tag','num'),'Enable','inactive');
   set(findobj('Tag','den'),'Enable','inactive');
   set(findobj('Tag','constante'),'Enable','inactive');
   axes(handles.axes1)
   newplot;axis off;hold off;
   axes(handles.axes2)
   newplot;axis off;hold off;
elseif get(hObject,'Value') &&~get(findobj('Tag','radiobutton2'),'Value')
&& ~get(findobj('Tag','radiobutton3'),'Value')
   handles.mfdeseado=0;
   handles.mseg=0;
   handles.num=0;
   handles.den=0;
   handles.const=0;
   set(findobj('Tag','mfdeseado'),'Enable','on');
   set(findobj('Tag','mseg'),'Enable','on');
   set(findobj('Tag','num'),'Enable','on');
   set(findobj('Tag','den'),'Enable','on');
   set(findobj('Tag','constante'),'Enable','on');
end
set(findobj('Tag','radiobutton2'),'Value',[0.0]);
set(findobj('Tag','radiobutton3'),'Value',[0.0]);
guidata(hObject,handles);
```

- Código del *radio button* “Atraso”.

```
clc;
if ~get(hObject,'Value') && ~get(findobj('Tag','radiobutton1'),'Value')
&& ~get(findobj('Tag','radiobutton3'),'Value')
   set(findobj('Tag','num'),'String','');
   set(findobj('Tag','den'),'String','');
   set(findobj('Tag','mfdeseado'),'String','');
   set(findobj('Tag','mseg'),'String','');
   set(findobj('Tag','constante'),'String','');
   set(findobj('Tag','gank'),'String','');
   set(findobj('Tag','cero'),'String','');
   set(findobj('Tag','polo'),'String','');
   set(findobj('Tag','ganancia'),'String','');
   set(findobj('Tag','cc'),'String','');
   set(findobj('Tag','pc'),'String','');
   set(findobj('Tag','kc'),'String','');
   set(findobj('Tag','raya'),'String','');
   set(findobj('Tag','mfdeseado'),'Enable','inactive');
   set(findobj('Tag','mseg'),'Enable','inactive');
   set(findobj('Tag','num'),'Enable','inactive');
   set(findobj('Tag','den'),'Enable','inactive');
   set(findobj('Tag','constante'),'Enable','inactive');
   axes(handles.axes1)
   newplot;axis off;hold off;
   axes(handles.axes2)
   newplot;axis off;hold off;
elseif get(hObject,'Value') &&~get(findobj('Tag','radiobutton1'),'Value')
&& ~get(findobj('Tag','radiobutton3'),'Value')
   handles.mfdeseado=0;
   handles.mseg=0;
   handles.num=0;
   handles.den=0;
   handles.const=0;
   set(findobj('Tag','mfdeseado'),'Enable','on');
   set(findobj('Tag','mseg'),'Enable','on');
   set(findobj('Tag','num'),'Enable','on');
   set(findobj('Tag','den'),'Enable','on');
   set(findobj('Tag','constante'),'Enable','on');
end
set(findobj('Tag','radiobutton1'),'Value',[0.0]);
set(findobj('Tag','radiobutton3'),'Value',[0.0]);
guidata(hObject,handles);
```

- Código del *radio button* “Atraso-Adelanto”.

```
clc;
if ~get(hObject,'Value') && ~get(findobj('Tag','radiobutton1'),'Value')
&& ~get(findobj('Tag','radiobutton2'),'Value')
   set(findobj('Tag','num'),'String','');
   set(findobj('Tag','den'),'String','');
   set(findobj('Tag','mfdeseado'),'String','');
   set(findobj('Tag','mseg'),'String','');
   set(findobj('Tag','constante'),'String','');
   set(findobj('Tag','gank'),'String','');
   set(findobj('Tag','cero'),'String','');
   set(findobj('Tag','polo'),'String','');
   set(findobj('Tag','ganancia'),'String','');
   set(findobj('Tag','cc'),'String','');
   set(findobj('Tag','pc'),'String','');
   set(findobj('Tag','kc'),'String','');
   set(findobj('Tag','raya'),'String','');
   set(findobj('Tag','mfdeseado'),'Enable','inactive');
   set(findobj('Tag','mseg'),'Enable','inactive');
   set(findobj('Tag','num'),'Enable','inactive');
   set(findobj('Tag','den'),'Enable','inactive');
   set(findobj('Tag','constante'),'Enable','inactive');
   axes(handles.axes1)
   newplot;axis off;hold off;
   axes(handles.axes2)
   newplot;axis off;hold off;
elseif get(hObject,'Value') &&
~get(findobj('Tag','radiobutton1'),'Value') &&
~get(findobj('Tag','radiobutton2'),'Value')
   handles.mfdeseado=0;
   handles.mseg=0;
   handles.num=0;
   handles.den=0;
   handles.const=0;
   set(findobj('Tag','mfdeseado'),'Enable','on');
   set(findobj('Tag','mseg'),'Enable','on');
   set(findobj('Tag','num'),'Enable','on');
   set(findobj('Tag','den'),'Enable','on');
   set(findobj('Tag','constante'),'Enable','on');
end
set(findobj('Tag','radiobutton1'),'Value',[0.0]);
set(findobj('Tag','radiobutton2'),'Value',[0.0]);
guidata(hObject,handles);
```

Anexo IV: Código de los controles de inserción de datos

- Código del *edit text* “Margen de fase deseado”.

```
mfde=get(hObject,'String');
if ~isempty(mfde)
   mfdes=str2double(mfde);
   if isnan(mfdes)
      warndlg('El valor de margen de fase deseado debe ser numérico.',...
'Aviso')
      set(hObject,'String','');
   elseif mfdes < 30 || mfdes > 60
      warndlg({'El margen de fase deseado debe ser','30° <= Mfd <=',...
60°.'},'Aviso');
      set(hObject,'String','');
   end
elseif isempty(mfde)
   mfdes=str2double(mfde);
end

handles.mfdeseado=mfdes;
guidata(hObject,handles);
```

- Código del *edit text* “Margen de seguridad”.

```
ms=get(hObject,'String');
if ~isempty(ms)
   mse=str2double(ms);
   if isnan(mse)
      warndlg('El valor de margen de seguridad debe ser insertado',...
correctamente.','Aviso');
      set(hObject,'String','');
   elseif mse > 12 || mse < 5
      warndlg({'El valor de margen de seguridad insertado debe ser',...
,'5°<= Mseg <=12°.'} ,'Aviso');
      set(hObject,'String','');
   end
elseif isempty(ms)
   mse=str2double(ms);
end
handles.mseg=mse;
guidata(hObject,handles);
```

- Código del *edit text* “Numerador”.

```
nu=get(hObject,'String');
if ~isempty(nu)
   num=str2num(nu);
   if isempty(num)|| isequal(num(1,1),0)
      errordlg('La matriz numeradorr debe ser insertada',...
correctamente.','Error');
      set(hObject,'String','');
   end
elseif isempty(nu)
   num=str2num(nu);
end
handles.num=num;
guidata(hObject,handles);
```

- Código del *edit text* “Denominador”.

```
de=get(hObject,'String');
if ~isempty(de)
   den=str2num(de);
   if isempty(den) || isequal(den(1,1),0)
      errordlg('La matriz denominador debe ser insertada',...
correctamente.','Error');
      set(hObject,'String','');
   end
elseif isempty(de)
   den=str2num(de);
end
handles.den=den;
guidata(hObject,handles);
```

- Código del edit text “Constante de error estático”

```
con=get(hObject,'String');
if ~isempty(con)
   cons=str2double(con);
   if isnan(cons)
      warndlg('En caso de insertarse un valor de constante de error,...
estático, este debe ser numérico.','Aviso');
      set(hObject,'String','');
      set(findobj('Tag','gank'),'String','');
      cons=0;
   elseif cons <= 0
      warndlg('El valor de constante de error estático insertado debe,...
ser mayor que cero.' ,'Aviso');
      set(hObject,'String','');
      cons=0;
```

```
   end
elseif isempty(con)
   cons=str2double(con);
   cons=0;
end
handles.const=cons;
guidata(hObject,handles);
```

Anexo V: Código del botón “Aplicar”

```
if ~get(findobj('Tag','radiobutton1'),'Value') &&
~get(findobj('Tag','radiobutton2'),'Value') &&
~get(findobj('Tag','radiobutton3'),'Value')
   warndlg('Para ejecutar el programa debe seleccionar primeramente',...
el tipo de compensador.','Aviso');
   return
else
   Mfd=handles.mfdeseado;
   Mseg=handles.mseg;
   n=handles.num;
   d=handles.den;
   const=handles.const;
   [f1,c1]=size(n);
   [f2,c2]=size(d);
```

- Código para el ajuste de ganancia.

```
% Ajuste de ganancia para el requerimiento de estado estacionario
   % Sistema tipo 0
   if ~isequal(d(1,c2),0) && const~=0 && ~isempty(n
      k= const/(polyval(n,0)/polyval(d,0));
      set(handles.gank,'String',k);
   % Sistema de tipo superior a 2
   elseif c2 > 3 && isequal(d(1,c2),0) && isequal(d(1,c2-1),0) &&
isequal(d(1,c2-2),0) && const~=0 && ~isempty(n)
      warndlg('El sistema es de tipo superior a dos, verifique',...
haber introducido correctamente la matriz denominador.','Aviso');
      return
   % Sistema tipo 2
   elseif c2 > 2 && isequal(d(1,c2),0) && isequal(d(1,c2-1),0) &&
const~=0 && ~isempty(n
      d1=deconv(d,[1 0 0]);
      k= const/(polyval(n,0)/polyval(d1,0));
      set(handles.gank,'String',k);
   % Sistema tipo 1
   elseif c2 >= 2 && isequal(d(1,c2),0) && ~isequal(d(1,c2-1),0) &&
const~=0 && ~isempty(n)
      d1=deconv(d,[1 0]);% Implica multiplicar por S la FT
      k=const/(polyval(n,0)/polyval(d1,0));
      set(handles.gank,'String',k);
   else
      k=1;
      set(handles.constante,'String','');
      handles.const=0;
   end
end
```

- Código de verificación de los requisitos para iniciar el diseño.

```
if 30<=Mfd && Mfd<=60 && 5<=Mseg && Mseg<=12 && ~isempty(n) &&
~isempty(d) && ~isequal(n(1,1),0) && ~isequal(d(1,1),0) && 1<c2
&& c1<=c2 && 0<=const
   g=tf([k*n],[d]);
   [mg,mf,w1,w2]=margin(g);
   if mf > Mfd
      warndlg('El sistema no es necesario compensarlo porque posee',...
un margen de fase superior al deseado.','Aviso');
   return
```

- Código para el diseño del compensador de adelanto.

```
% Diseño del compensador de adelanto
   elseif get(findobj('Tag','radiobutton1'),'Value')
      Procesando% Llama a la interfaz diseñada con el mensaje de espera
      fimax=Mfd-mf+Mseg;
      alfa=(1-sin(fimax*pi/180))/(1+sin(fimax*pi/180));
      aten=sqrt(alfa);
      frec=0.01:0.01:300;
      [m,f,W]=bode(g,frec);
      diferencia=10;
      for i=1:length(m)
          diferenciaactual=abs(aten-m(i));
          if diferenciaactual<diferencia
             diferencia=diferenciaactual;
             ind=i;
          end
      end
      Wm=W(ind);
      T=1/(sqrt(alfa)*Wm);
      nc=[T 1];
      dc=[alfa*T 1];
      gc=tf(nc,dc);
      [mgsc,mfsc,w1sc,w2sc]=margin(g*gc);
      a=1/T;
      b=1/(alfa*T);
      c=k/alfa;
      d=num2str(a);      % Se llevan los valores calculados
      e=num2str(b);      % a formato cadena
      f=num2str(c);
      h='(S+';
      k=')';
      l=strcat(h,d,k);   % Se contadenan las cadenas
      p=strcat(h,e,k);
      q='___________';
      set(handles.cero,'String',l);
      set(handles.cc,'String',l);
      set(handles.polo,'String',p);
```

```
      set(handles.pc,'String',p);
      set(handles.ganancia,'String',f);
      set(handles.kc,'String',f);
      set(handles.raya,'String',q);
```

- Código para el diseño del compensador de atraso.

```
% Diseño del compensador de atraso
   elseif get(findobj('Tag','radiobutton2'),'Value')
      Procesando
      fimax=Mfd-180+Mseg;
      frec=0.01:0.01:300;
      [m,f,W]=bode(g,frec);
      diferencia=10;
      for i=1:length(f)
          diferenciaactual=abs(fimax-f(i));
          if diferenciaactual<diferencia
             diferencia=diferenciaactual;
             ind=i;
          end
      end
      Wcn=W(ind);
      aten=m(ind);
      beta=aten;
      T=10/Wcn;
      nc=[T 1];
      dc=[beta*T 1];
      gc=tf(nc,dc);
      [mgsc,mfsc,w1sc,w2sc]=margin(g*gc);
      a=1/T;
      b=1/(beta*T);
      c=k/beta;
      d=num2str(a);
      e=num2str(b);
      f=num2str(c);
      h='(S+';
      k=')';
      l=strcat(h,d,k);
      p=strcat(h,e,k);
      q='___________';
      set(handles.cero,'String',l);
      set(handles.cc,'String',l);
      set(handles.polo,'String',p);
      set(handles.pc,'String',p);
      set(handles.ganancia,'String',f);
      set(handles.kc,'String',f);
      set(handles.raya,'String',q);
```

- Código para el diseño del compensador de atraso-adelanto.

```
% Diseño del compensador de atraso adelanto
   elseif get(findobj('Tag','radiobutton3'),'Value')
      [Z,P]=zpkdata(g,'v');
      Z=damp(Z);
      Z=sort(Z);
      P=damp(P);
      P=sort(P);
      Wesq=[damp(P);damp(Z)];
      Wesq=sort(Wesq);
      Wesq=unique(Wesq);
      pend=0;
      ind1=0;
      ind2=0;
      % Determinando la nueva frecuencia de cruce más apropiada en la
recta de -40 dB/década
      for i=1:length(Wesq);
          a=find(Z==Wesq(i));
          [fa ca]=size(a);
          b=find(P==Wesq(i));
          [fb cb]=size(b);
          pend=pend+20*fa-20*fb;
          if pend== -40
             ind1=i;
          elseif pend== -60
             ind2=i;
          end
          if ind2-ind1==1;
             Wcn1=Wesq(ind1);
             Wcn2=Wesq(ind2);
             break
          elseif i==length(Wesq)
             warndlg('El sistema no se puede compensar a través',...
del método de Atraso-Adelanto.','Aviso');
             return
          end
      end
      Procesando
      frec=Wcn1:0.01:Wcn2;
      [m,f,W]=bode(g,frec);
      diferencia=30;
      for i=1:length(W)
          Wcnr=W(i);
          mWcnr=m(i);
          fiWcnr=f(i);
          fiWcnd=Mfd-180;
          fimax=fiWcnd-fiWcnr+Mseg;
          beta=(sin(fimax*pi/180)+1)/(1-sin(fimax*pi/180));
          T2=10/Wcnr;
          T1=beta/(Wcnr*mWcnr);
          nc=conv([T2 1],[T1 1]);
```

```
            dc=conv([beta*T2 1],[T1/beta 1]);
            gc=tf(nc,dc);
            [mgsc,mfsc,w1sc,w2sc]=margin(g*gc);
            diferenciaactual=abs(Mfd-mfsc);
            if diferenciaactual<diferencia;
               diferencia=diferenciaactual;
               ind=i;
            end
         end
         Wcnr=W(ind);
         mWcnr=m(ind);
         fiWcnr=f(ind);
         fiWcnd=Mfd-180;
         fimax=fiWcnd-fiWcnr+Mseg;
         beta=(sin(fimax*pi/180)+1)/(1-sin(fimax*pi/180));
         T2=10/Wcnr;
         T1=beta/(Wcnr*mWcnr);
         nc=conv([T2 1],[T1 1]);
         dc=conv([beta*T2 1],[T1/beta 1]);
         gc=tf(nc,dc);
         [mgsc,mfsc,w1sc,w2sc]=margin(g*gc);
         cat=num2str(1/T2);
         pat=num2str(1/(beta*T2));
         cad=num2str(1/T1);
         pad=num2str(beta/T1);
         a='(S+';
         b=')';
         l=strcat(a,cat,b,a,cad,b);
         p=strcat(a,pat,b,a,pad,b);
         raya='_____________________';
         set(handles.cero,'String',l);
         set(handles.cc,'String',l);
         set(handles.polo,'String',p);
         set(handles.pc,'String',p);
         set(handles.ganancia,'String',1);
         set(handles.kc,'String','');
         set(handles.raya,'String',raya);
      end
```

- Código de finalización.

```
delete(Procesando)
   axes(handles.axes1)              % Se proyecta en los axes 1 y 2
   margin(g*gc);                    % los gráficos de Bode
   axis off;                        % y respuesta al paso
   sysc=feedback(g*gc,1);           % respectivamente
   axes(handles.axes2)
   step(sysc);
   axis off;
   if mfsc > 60
      warndlg({'El compensador determinado es','irrealizable',...
físicamente.'},'Aviso');
   elseif Mfd > mfsc+3
```

```
      warndlg({'Pruebe modificar el margen de seguridad','para,',...
de ser posible, alcanzar el margen de fase',...
deseado','5°<=Mseg<=12°.'},'Aviso');
   elseif Mfd < mfsc-3
      warndlg({'Pruebe modificar el margen de seguridad','para,',...
de ser posible, alcanzar el margen de fase',...
deseado','5°<=Mseg<=12°.'},'Aviso');
   end
% Si el grado del denominador es menor al del numerador o es igual a cero
elseif (c2<c1 || c2==1) && 30<=Mfd && Mfd<=60 && 5<=Mseg && Mseg<=12 &&
~isempty(n) && ~isempty(d) && ~isequal(n(1,1),0) && ~isequal(d(1,1),0) &&
0<=const
   warndlg('Introduzca correctamente las matrices numerador y',...
denominador. El grado del denominador debe ser mayor o igual',...
al del numerador, y a su vez mayor que cero.','Aviso')
   set(findobj('Tag','num'),'String','');
   set(findobj('Tag','d
en'),'String','');
   handles.num=0;
   handles.den=0;
% Si no se cumple algo de lo anterior, es decir, si se introdujo algún
valor fuera de los rangos permisibles o se dejó en blanco algún dato
else
   warndlg({'Introduzca correctamente los datos.','Solo puede dejar en
blanco, si no le es necesario, el valor de constante de error estático.'}
,'Aviso');
end
guidata(hObject,handles);
```

Anexo VI: Código de los botones “Reset”, “Salir” y menú “Ayuda”

- Código del botón “Reset”.

```
clc;
% Se reinicia todo, enviando caracteres vacíos a los edit y static text,
inhabilitando los edit de datos, poniendo en cero los radio button y
% limpiando las gráficas
set(findobj('Tag','num'),'String','');
set(findobj('Tag','den'),'String','');
set(findobj('Tag','mfdeseado'),'String','');
set(findobj('Tag','mseg'),'String','');
set(findobj('Tag','constante'),'String','');
set(findobj('Tag','gank'),'String','');
set(findobj('Tag','cero'),'String','');
set(findobj('Tag','polo'),'String','');
set(findobj('Tag','ganancia'),'String','');
set(findobj('Tag','cc'),'String','');
set(findobj('Tag','pc'),'String','');
set(findobj('Tag','kc'),'String','');
set(findobj('Tag','raya'),'String','');
set(findobj('Tag','radiobutton1'),'Value',[0.0]);
set(findobj('Tag','radiobutton2'),'Value',[0.0]);
set(findobj('Tag','radiobutton3'),'Value',[0.0]);
set(findobj('Tag','mfdeseado'),'Enable','inactive');
set(findobj('Tag','mseg'),'Enable','inactive');
set(findobj('Tag','num'),'Enable','inactive');
set(findobj('Tag','den'),'Enable','inactive');
set(findobj('Tag','constante'),'Enable','inactive');
axes(handles.axes1)
newplot;axis off;hold off;
axes(handles.axes2)
newplot;axis off;hold off;
guidata(hObject,handles)
```

- Código del botón “Salir”.

```
questdlg('¿Desea salir del programa?','Salir','Si','No','No');
if strcmp(ans,'Si')
   delete(findobj('Tag','fondo'));close all;
end
```

- Código del menú “Ayuda”.

```
winopen('Ayuda.pdf')
```

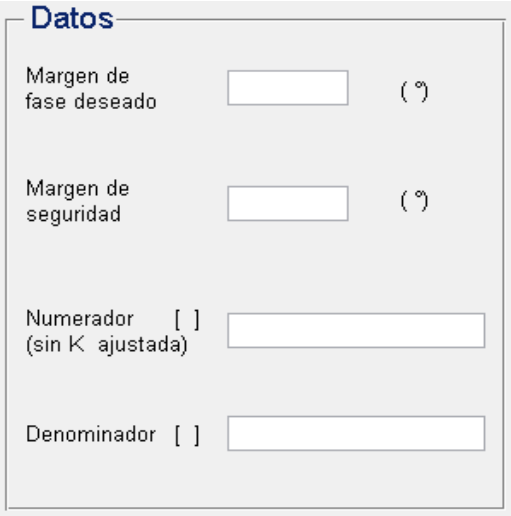
Anexo VII: Ayuda

AYUDA

Para una correcta utilización de la aplicación, siga los siguientes pasos:

1. En el panel con nombre Tipo de compensador, seleccione el método de compensación que desee aplicar.

2. Inserte, en el panel nombrado Datos, los requerimientos necesarios para ejecutar la aplicación.

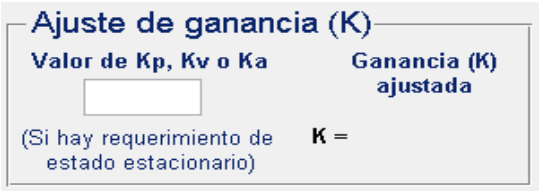


Requisitos de los datos:

- El valor de Margen de fase deseado debe estar entre 30° y 60°.

- El Margen de seguridad debe encontrarse en el rango de 5° a 12°, incluyendo estos valores.
- Las matrices numerador y denominador deben poseer valores numéricos distintos de cero.
- El grado del denominador debe ser mayor o igual al del numerador, y a su vez, mayor que cero.
- El sistema que representan las matrices numerador y denominador debe ser de tipo inferior o igual a dos.

3. Inserte, si es necesario, el valor de constante de error estático dentro de la casilla disponible en el panel Ajuste de ganancia (K).

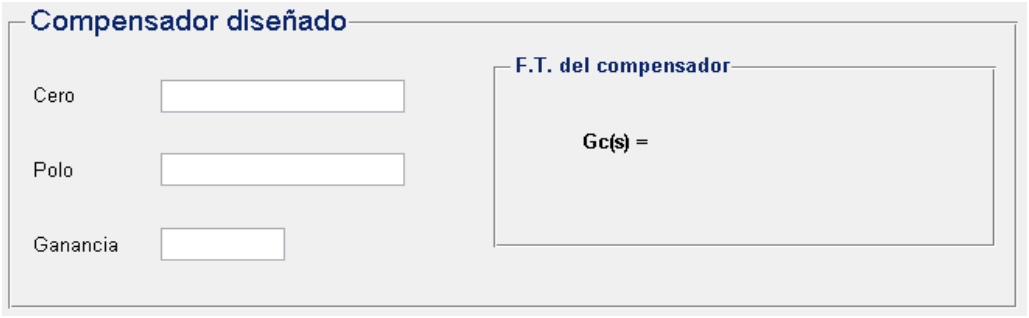


Consideraciones:

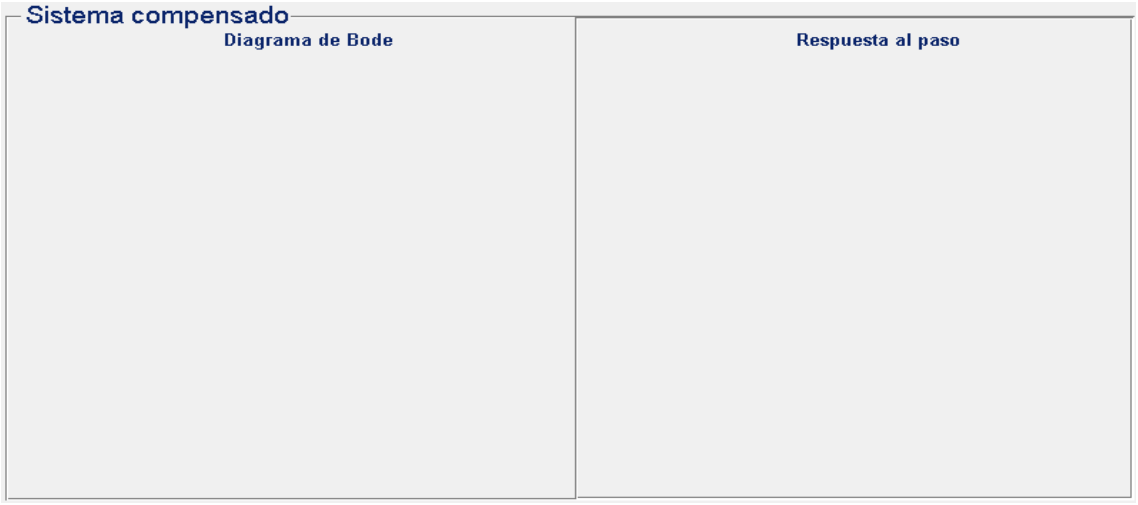
- La casilla está diseñada para acoger el valor de constante de error estático que se desee insertar, sin importar su naturaleza, es decir, pueden ser introducidos tanto el valor de constante de posición, el de velocidad como el de aceleración.
- La inserción del valor de constante de error estático como dato, se hará solo si desea alcanzar algún requerimiento de estado estacionario.

4. Presione el botón Aplicar para ejecutar la aplicación.

- Una vez aplicado el programa, en el panel Compensador diseñado se mostrarán el cero, el polo y la ganancia del compensador, además de la función transferencial del mismo, insertada a su vez en un pequeño panel interno (F.T. del compensador).



- En tanto, las trazas de Bode del sistema compensado, así como la respuesta ante entrada paso unitario del sistema compensado en lazo cerrado, serán mostradas en el panel Sistema compensado.



5. Puede utilizar el botón Reset para devolver todos los componentes de la aplicación a sus condiciones iniciales.

6. El botón Salir posibilita abandonar el programa.