ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE INGENIERÍA

ESTUDIO Y APLICACIÓN DE LOS ALGORITMOS GENÉTICOS PARA CONTROL

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL

SEBASTIÁN ADALBERTO GALEAS HURTADO

DIRECTOR: Prof. PATRICIO BURBANO MSc.

Quito, marzo 2006

DECLARACIÓN

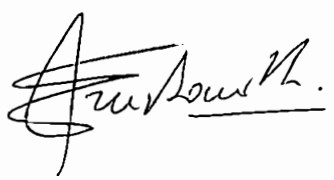
Yo, Sebastián Adalberto Galeas Hurtado, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Sebastián Adalberto Galeas Hurtado

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado en su totalidad por el señor Sebastián Adalberto Galeas Hurtado, bajo mi supervisión.



Prof. Patricio Burbano Msc

DIRECTOR DEL PROYECTO

AGRADECIMIENTOS

En primer lugar deseo darle las gracias y todo el crédito de este y todos los proyectos de mi vida a mi Dios: Creador, Señor y Amigo, a quién le debo todo lo que soy pues gracias a Él mi vida tiene sentido.

También quiero agradecer a mis padres y a mi ñaña que me dieron su apoyo y paciencia no solo en estos 5 años de estudio sino en toda mi vida.

Finalmente agradezco a mis profesores, sobre todo a mi director el Ing. Patricio Burbano por su valioso aporte y ayuda en el presente proyecto de titulación; y a mis amigos y compañeros de la Poli.

Sebastián Galeas

CONTENIDO

RESUMEN

El Proyecto de Titulación presentado en estas páginas, está estructurado en líneas generales de la siguiente manera:

En el Capítulo 1 se brinda un enfoque general de los dos aspectos más importantes dentro de este trabajo: los algoritmos genéticos y la robustez en los sistemas de control.

En el Capítulo 2 se realiza una explicación detallada del método de optimización de los algoritmos genéticos, su representación, sus operadores, su mecanismo de trabajo y un par de ejemplos sencillos.

El Capítulo 3 empieza a abordar el tema central del proyecto: el diseño de controladores robustos, aquí se incluye criterios muy importantes dentro de la robustez como la incertidumbre y las normas H.

En el Capítulo 4 se estudia la manera en cómo los algoritmos genéticos pueden ser utilizados de manera práctica en el diseño de los controladores robustos; también se introduce la idea de los controladores robustos óptimos.

En el Capítulo 5 se presentan varios casos de estudio con tres plantas distintas a las cuales se les diseñó los tipos de controladores estudiados en el presente trabajo. Se presentan los resultados obtenidos en las simulaciones con Matlab.

En el Capítulo 6 se presentan las conclusiones a las que se llegó tras la finalización del presente proyecto de titulación, además de algunas recomendaciones para quienes deseen desarrollar trabajos afines a éste.

Finalmente, en la sección de Anexos se presentan los archivos fuente (archivos de extensión *m*) escritos para Matlab para el diseño de los controladores propuestos; también se presenta un muy interesante desarrollo matemático del ISE por cortesía del Dr. Tohru Kawabe.

PRESENTACIÓN

En los años 60, John Holland concibe la idea de emplear fundamentos de la evolución de las especies para el desarrollo de programas computacionales con los objetivos de implementar sistemas con la capacidad de adaptarse dinámicamente a un entorno cambiante en el tiempo y poder diseñar sistemas artificiales que simulasen la evolución natural. Los algoritmos genéticos habían visto la luz.

Paralelamente, por esos años, la teoría de control conocida como clásica hasta el momento, iniciaba a mirar un poco más allá, para adentrarse en nuevas teorías y desarrollos (como la modelación en el espacio de estados) que tomarían el nombre de lo que hoy se le conoce como el control moderno.

En los años 70 y 80 se empezó a emplear a los algoritmos genéticos dentro del diseño de técnicas de exploración y optimización muy poderosas mientras la ciencia del control seguía avanzando y aparecía el control adaptatitvo, predictivo, robusto y otros.

De los años 80 para adelante, el número de problemas a los que se han enfrentado los algoritmos genéticos (principalmente de optimización y de inteligencia artificial) no ha parado de aumentar, aplicándolos en todo tipo de variantes y es precisamente en este punto donde los algoritmos empiezan a relacionarse con la teoría de control gracias a su flexibilidad y versatilidad.

En este trabajo se enfoca el funcionamiento de los algoritmos genéticos y cómo emplearlos dentro del control y si bien no se busca explotar toda la potencialidad de los algoritmos genéticos, sí se consigue emplearlos eficazmente como técnica de optimización que es, para resolver problemas de diseño de control robusto y control óptimo robusto.

Como es lógico, antes de hacer esto, se estudia las bases del control robusto y óptimo así como también las distintas metodologías de diseño.

El trabajo aquí desarrollado se justifica plenamente por el gran desarrollo que están teniendo las técnicas de computación evolutiva, dentro del control, tratando de dar un enfoque más práctico a esta ciencia.

CAPÍTULO 1

INTRODUCCIÓN

1.1 ALGORITMOS GENÉTICOS

Los algoritmos genéticos son un método para resolver problemas de búsqueda y optimización basados en la selección natural, emulando la evolución de los seres vivos.

Para trabajar con estos algoritmos, en una primera instancia, se debe parametrizar el problema en una serie de variables o genes ($x_i,...,x_n$) que se *codifican* en un cromosoma.

Todos los operadores empleados por los algoritmos genéticos se aplicarán sobre estos cromosomas; o mejor dicho, sobre poblaciones de éstos. Dichos operadores modifican repetidamente la población de estos cromosomas o individuos que son potenciales soluciones del problema los cuales "compiten" para ver cuál de ellos resulta ser la mejor solución.

En cada generación o etapa, el algoritmo genético selecciona una población inicial de individuos, en la mayoría de los casos al azar; a cada uno de los cromosomas o individuos de esta población, se le aplicará la función de aptitud (función objetivo del problema de optimización) a fin de saber qué tan buena es la solución que genera. Conociendo la aptitud de cada cromosoma, los algoritmos genéticos seleccionan a los que se cruzarán en la siguiente generación (se supone que se escogerá a los "mejores"), los que se convierten en "padres" de individuos "hijos" de la siguiente generación.

A lo largo de generaciones sucesivas, la población va *evolucionando* hacia la solución óptima puesto que el "medio ambiente", constituido por los otros cromosomas, ejerce una presión selectiva sobre la población, de forma que sólo los que se encuentren mejor "adaptados" (los que resuelvan mejor el problema) sobreviven o heredan su material genético a las siguientes generaciones, igual que en la evolución de las especies.

El proceso que llevan a cabo los algoritmos genéticos requiere de la aplicación de tres reglas básicas en cada paso para crear las siguientes generaciones a partir de las generaciones previas:

- Reglas de selección de los padres para las generaciones futuras
- Reglas de cruce (*crossover* en inglés) para combinar la información de los padres.
- Reglas de mutación para introducir pequeñas alteraciones en los hijos.

El pseudocódigo de un algoritmo genético simple puede expresarse como el mostrado en la figura 1.1 [3]

También puede verse un proceso un poco más general, por medio del diagrama de bloques mostrado en la figura 1.2.

Los algoritmos genéticos pueden ser aplicados para resolver una gran variedad de problemas de optimización en los que los algoritmos clásicos de optimización no pueden aplicarse incluyendo problemas cuya función (función objetivo) es discontinua, no diferenciable, estocástica o altamente no lineal. [2]

Además, debe tenerse en cuenta que un algoritmo genético es independiente del problema, lo cual lo hace un algoritmo *robusto* por ser útil para cualquier problema, pero a la vez *débil*, pues no está especializado en ninguno.

```
INICIO /* Algoritmo Genético Simple */
  Generar una población inicial.
  Computar la función de aptitud de cada individuo.
  MIENTRAS NO Terminado HACER
  INICIO /* Producir nueva generación */
    PARA Tamaño Población/2 HACER
    INICIO /* Ciclo Reproductivo */
      Seleccionar 2 individuos de la generación anterior,
      con probabilidad proporcional a la función de aptitud
      de cada individuo.
      Cruzar, con cierta probabilidad, los 2 individuos
        obteniendo 2 descendientes.
      Mutar los 2 descendientes, con cierta probabilidad.
      Calcular la función de aptitud de los 2 descendientes.
      Insertar los dos descendientes en la nueva generación.
    FIN /* Ciclo Reproductivo */
   SI La población ha convergido ENTONCES
     Terminado = VERDADERO
  FIN /* Producir nueva generación */
FIN /* Algoritmo Genético Simple */
```

Figura 1.1 Pseudocódigo de un algoritmo genético simple

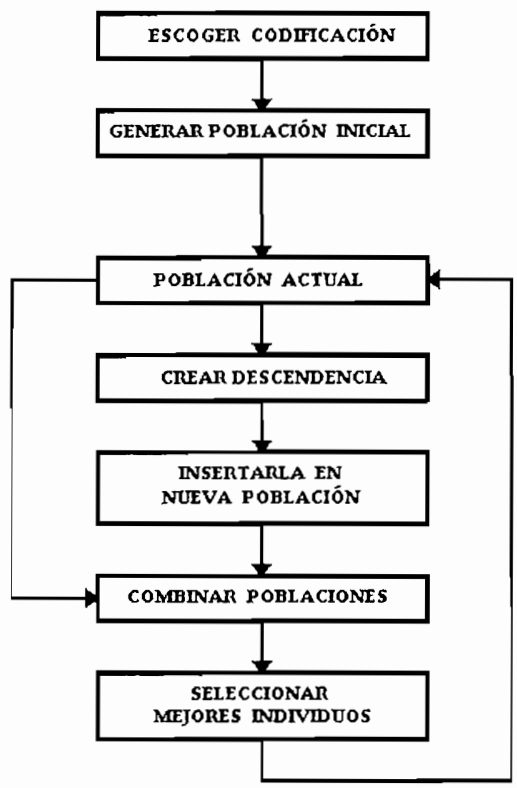


Figura 1.2 Diagrama de bloques de un algoritmo genético

1.2 CONTROLADORES ROBUSTOS

La robustez es, según el glosario estándar de la IEEE, "el grado en el que un sistema o un componente suyo, funciona correctamente en la presencia de entradas inválidas o en condiciones ambientales adversas".

El control robusto es una herramienta que apareció a partir del surgimiento de la teoría de control moderno más o menos por la década de los sesenta. Una definición muy interesante, de qué es el control robusto, es el que da Chandrasekharan [7]: "El control robusto se refiere al control de plantas desconocidas con dinámicas desconocidas bajo la influencia de perturbaciones desconocidas".

La parte clave del control robusto es, como se desprende de la anterior definición, la incertidumbre y cómo el controlador del sistema puede manejar este problema.

La incertidumbre, en un sistema de control, puede introducirse en tres puntos básicamente: puede existir incertidumbre en el modelo de la planta, están presentes también perturbaciones desconocidas en la planta misma y además el ruido que introducen los sensores de entrada al sistema. Esto puede apreciarse a breves rasgos en la figura 1.3

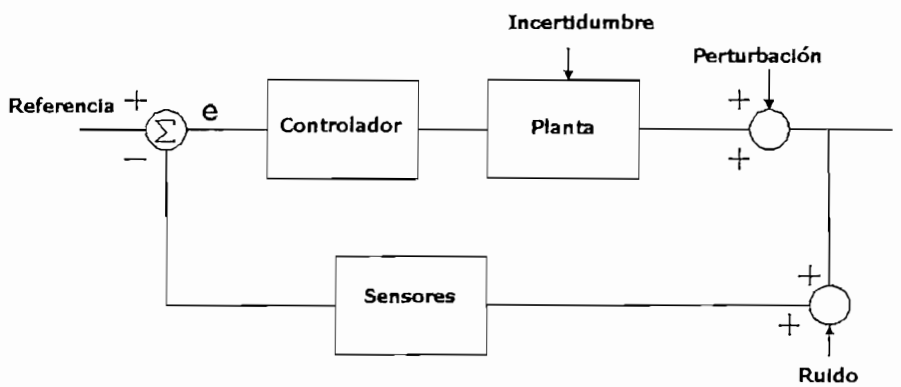


Figura 1.3 Sistema de control con incertidumbre

En el pasado, uno de los métodos más empleados para tratar la incertidumbre era el control estocástico. En el control estocástico se modela las incertidumbres del sistema como distribuciones de probabilidad, es decir trabaja con el valor esperado de la señal de control.

El control robusto en cambio busca acotar la incertidumbre antes que expresarla a manera de una distribución probabilística. Al hacer esto, el controlador permite cumplir los requerimientos de la planta en todos los casos. Viene a ser algo así como un método de análisis para "el peor de los casos" [21].

Hoy en día, el control robusto como tal, se ayuda de ciertas herramientas para su desarrollo, las cuales en su gran mayoría emplean mucho fundamento matemático. Entre estas herramientas vale la pena mencionar al control adaptativo, control difuso y H infinito.

En el control adaptativo, el controlador del sistema cambia su comportamiento de acuerdo a las modificaciones que puede sufrir la planta en su dinámica y dependiendo de las perturbaciones existentes. Por ello es que se puede emplear este control para el diseño de controladores robustos.

El control difuso se basa en la construcción de funciones representadas por grados de pertenencia las cuales permiten describir en cierta forma la incertidumbre de cada variable, razón que hace a este control aplicable al control robusto.

Las normas H infinito (llamadas así en honor a Hankel) son extensamente empleadas. Permiten medir las propiedades de un sistema de control en el dominio de la frecuencia y acerca de este tipo de control se hablará extensivamente en los siguientes capítulos junto con la ayuda de los algoritmos genéticos.

Es interesante mencionar también que los controladores pueden ser robustos en estabilidad o robustos en comportamiento, pero no se puede

conseguir en ambas formas, como en todo problema de control, existe un conflicto o un compromiso en las especificaciones de comportamiento dinámico de un sistema; [11] lo expresa de la siguiente manera: "se puede conseguir un controlador que brinde al sistema gran estabilidad pero con un mal comportamiento, o un excelente comportamiento pero sacrificando la estabilidad".

CAPÍTULO 2

ALGORITMOS GENÉTICOS

2.1 REPRESENTACIÓN DE LOS ALGORITMOS GENÉTICOS

Como se explicó en el capítulo 1, los algoritmos genéticos requieren que el conjunto de posibles soluciones, que toman el nombre de *individuos*, se codifiquen o representen en una cadena de lo que sería una especie de cromosoma el cual estará conformado por un número determinado de genes en donde cada grupo de genes representan un parámetro específico del individuo. Para comprender esto más claramente se puede asociar lo dicho con las bases biológicas de estos algoritmos, es decir; con individuos vivos como por ejemplo una mariposa.

Para fines explicativos se puede decir que el cromosoma de una mariposa está formado por 8 genes, que pueden ser representados o codificados por 8 bits. Los dos primeros bits pueden representar la forma del cuerpo del animal, el siguiente bit la longitud de las antenas, los dos siguientes la forma de la cabeza y los tres últimos la forma de las alas. (Este ejemplo emplea la *representación binaria* que más adelante se explica en profundidad).

Debe recordarse que en la mayoría de los casos una representación adecuada es la clave para tener una resolución óptima, razón por la cual debe escogerse adecuadamente esta codificación de acuerdo al problema a solucionarse. La codificación de los cromosomas, como es lógico pensar, varía de un problema a otro dependiendo de su naturaleza.

Actualmente se puede hallar representaciones binarias, representaciones con números reales, con cadenas de letras, representaciones basadas en orden

–cuando existe una relación entre un gen y el que le precede- y representaciones en ratios –cuando la suma de los genes debe ser constante-. Sin embargo las más empleadas en forma práctica son las representaciones: binaria y real.

2.1.1 REPRESENTACIÓN BINARIA [12]

Esta representación es la más común y más empleada desde que se empezó a trabajar con los algoritmos genéticos. Como su nombre lo indica, consiste en una codificación binaria en donde las variables continuas se normalizan primeramente para luego codificarlas en código binario (normalizar: hacer que tomen valores únicamente desde 0 hasta 1). Esto se hace de la siguiente manera:

Se tiene un vector $\underline{x}$ formado por *n* variables x_i con i=0,1,2,...,n. Los valores máximo y mínimo de entre todo el vector $\underline{x}$ son respectivamente: $x_{máx}$ y $x_{mín}$; entonces cada valor de x_i se normaliza empleando la ecuación 2.1:

$$x_{i_{norm}} = \frac{x_{i-} x_{mín}}{x_{máx-} x_{mín}} \qquad \text{Ecuación 2.1}$$

Luego cada x_i se transforma a un número binario c_i formado por un número de bits acorde a la precisión requerida: $\mathbf{c_i}$**=<b[1], b[2],... b[j]..., b[m]>** con b[j]=0 ó 1. Este número binario $\mathbf{c_i}$ se encuentra aplicando el siguiente pseudocódigo:

START

j = 1 y $q_j = x_{inorm}$

WHILE ($j \leq m$) **DO**

IF ($q_j - 2^{-j} \geq 0$)

b[j]=1

$q_{j+1} = q_j - 2^{-j}$

ELSE

b[j]=0

$q_{j+1} = q_j$

```
            END IF
            j=j+1
        END WHILE
END
```

2.1.2 REPRESENTACIÓN REAL [12]

Este método es más directo, simple y fácil que el anterior ya que no se requiere de una codificación propiamente dicha. Además es posible ampliar su rango de desempeño con la ayuda de ciertos operadores genéticos que permitan obtener una mayor exactitud.

La representación real es especialmente útil para problemas con restricciones. [23]

En la representación real cada individuo o cromosoma $\mathbf{c}_i$, consiste en un vector o cadena de números reales $\mathbf{x}_i$. Esta representación fue introducida cuando se empezó a trabajar con problemas con parámetros reales y con problemas que presentan algún tipo de restricciones; es decir, básicamente con problemas de optimización con variables en el dominio del tiempo.

La precisión con la que se desee operar, dependerá del computador empleado en la resolución.

Sea cual fuere la representación que se escoja, debe tenerse en cuenta el "Teorema de los Esquemas" (uno de los resultados fundamentales en la teoría de los algoritmos genéticos) el cual afirma que: "La mejor optimización (o sea aquella sobre la que los algoritmos genéticos funcionan de la mejor manera), es aquella que tiene un alfabeto de cardinalidad dos". Esto es, que el cromosoma esté formado por un número de genes igual a 2^n, siendo *n* un número entero. [17]

2.2 FUNCIÓN DE APTITUD

La función de aptitud es aquella que permite cuantificar de una manera precisa la bondad o cuán apto es un cromosoma para sobrevivir.

La aptitud o bondad (*fitness* en inglés) mide y evalúa la posibilidad de cada individuo para reproducirse y evolucionar hasta llegar a ser la solución del problema planteado; es decir si podrá sobrevivir en las generaciones que se irán creando. Por ejemplo si se desea hallar el máximo de una función cualquiera F(x,y), mientras mayor sea el valor de $F(x_i,y_i)$, mayor será la aptitud del individuo (x_i,y_i).

Esta característica de aptitud sirve para diferenciar a los mejores individuos y es la base para seleccionar la población que van a ser los "padres" de los individuos de las posteriores generaciones.

Cuando se tiene un problema con algún tipo de restricción se hace necesario la inclusión de una *penalización* dentro de la función de aptitud (esto se empleará en forma práctica en el Capítulo 5) de manera que cuando los individuos tiendan a acercarse a valores que den como resultado valores no permisibles se les castigue disminuyendo drásticamente su bondad para que no puedan sobrevivir.

Una vez evaluada la aptitud o fitness, se debe crear (como se explicó en el Capítulo 1) la nueva población teniendo en cuenta que las características óptimas de los mejores individuos se "hereden" a ésta.

Para este efecto, se debe seleccionar a una serie de individuos o cromosomas; y esta selección se puede hacer por medio de varias formas como las siguientes: [15]

- Selección por rueda de ruleta
- Selección por orden

- Selección por torneo
- Selección escalada
- Selección por rango
- Selección generacional
- Selección por estado estacionario
- Selección jerárquica

De todos estos, los métodos más empleados y representativos son los tres que a continuación se detallan.

2.2.1 SELECCIÓN DE RUEDA DE RULETA [14], [12]

Esta es la técnica de selección más común. Es conocida también como Selección Proporcional por emplear un mecanismo proporcional, de la siguiente forma:

La probabilidad de que un individuo $\mathbf{c_i}$, avance a una siguiente generación es proporcional a su aptitud $\mathbf{p_i}$; el número de descendientes o hijos $\boldsymbol{\xi}$, de $\mathbf{c_i}$ se obtiene con el producto: $\xi = p_i * \mu$, donde $\boldsymbol{\mu}$ es el número de individuos de la población. Esta probabilidad se representa gráficamente por medio de una circunferencia dividida en μ segmentos como una ruleta de la fortuna; el área de cada segmento depende proporcionalmente de la aptitud del correspondiente individuo.
Se supone que se hace girar esta ruleta y se toma el cromosoma cuyo campo "señala" la aguja. Se puede apreciar a simple vista que la probabilidad de que un individuo sea elegido depende de su aptitud. La ruleta debe "hacerse girar" el número de veces μ que equivale al número de individuos de la población.

Una vez escogidos los cromosomas, se toman parejas aleatorias de estos cromosomas y se los emparejan (aplicación de operadores genéticos) para generar "hijos".

En la figura 2.1 puede apreciarse un ejemplo muy sencillo. Aquí se tienen cinco individuos o cromosomas y puede notarse que el cromosoma más apto es el número 5 y por tanto tendrá más probabilidades de ser elegido, mientras que el cromosoma número 4 es menos apto y tendrá la menor probabilidad de ser escogido. La suma de las aptitudes de todos los individuos es siempre igual a 1.

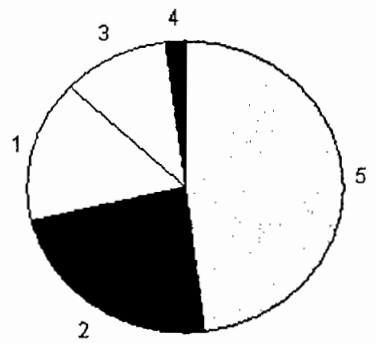


Figura 2.1 Rueda de ruleta

Con este esquema de selección, la aptitud de los individuos solamente puede ser positiva.

2.2.2 SELECCIÓN BASADA EN EL ORDEN [17]

En este esquema se conserva un porcentaje definido de la población, la mayoría generalmente, para la siguiente generación. Se coloca toda la población por orden de aptitud, y los menos aptos son eliminados y sustituidos por la descendencia de alguno de los mejores con algún otro individuo de la población.

A este esquema se le pueden aplicar otros criterios como por ejemplo, se puede crear la descendencia de uno de los mejores individuos y ésta sustituye al más parecido a éste, entre los perdedores. A esto se le conoce como "*crowding*", y fue introducido por DeJong. En realidad, para este esquema se escoge un factor, "*crowding factor*" (cf), tal que cuando nace un nuevo individuo, se seleccionan *cf* individuos de la población, y se elimina al más parecido al nuevo cromosoma.

2.2.3 SELECCIÓN DE TORNEO

Este método presenta un mejor desempeño que el de la ruleta. El esquema consiste en escoger aleatoriamente un número N de individuos de la población, y el que tiene la mayor aptitud de ellos, se reproduce. Al tamaño de este grupo N, se le conoce con el nombre de *tamaño del torneo*, y si se desea ser más exigente con la selección del más apto, se eleva en número de miembros del grupo (N); esto es basándose en el hecho probabilístico de que el vencedor de un grupo N1 es más óptimo que el de un grupo N2 si se cumple el hecho de que N1>N2. [12]

2.3 OPERADORES GENÉTICOS

Hasta el momento se ha visto la forma en que se debe escoger a los individuos de una población inicial para hacerlos reproducirse; en los siguientes párrafos se indicará como se debe realizar este paso muy importante dentro del proceso del algoritmo genético.

Una vez que se ha elegido a los individuos más aptos dentro de la población, éstos deben ser alterados de alguna forma con la esperanza de mejorar su aptitud para la siguiente generación. Existen dos estrategias básicas para llevar a cabo lo señalado. La primera y más sencilla es la mutación y la segunda es el cruce (*crossover* en inglés).

2.3.1 MUTACIÓN

De igual forma que la mutación en los seres vivos, este mecanismo se encarga de cambiar un gen aleatorio de un cromosoma por otro gen. Esto provoca pequeñas alteraciones en puntos concretos del código genético del individuo.

Así mismo, como ocurre en el mundo real, la mutación no es muy común; por esta razón es que la probabilidad de que se realice una mutación (p_m) es baja y típicamente de: [14]

- 0.001 para una población numerosa (100 individuos o más)
- 0.01 para una población reducida (30 individuos o menos)

Cuando se tiene una representación binaria, como es intuitivo pensar, la mutación invierte al bit escogido: si es 0 lo hace 1 y viceversa.

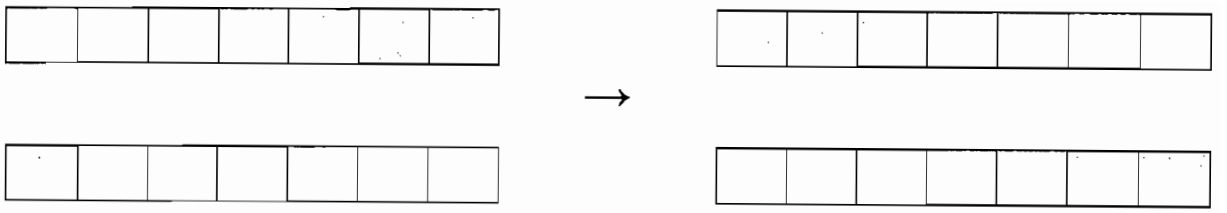
Aunque la mutación inició pensándose en la representación binaria, también existe este operador para la representación real. En este caso se puede alterar también uno o más genes del individuo por medio de una distribución de probabilidad definida dentro del rango del dominio admisible de valores para ese gen. [12]

2.3.1 CRUCE

Este operador genético implica elegir a dos individuos al azar para que intercambien entre sí segmentos de su código, produciendo una descendencia cuyos individuos son combinaciones de sus padres. Este proceso intenta simular el proceso análogo de recombinación que se da en los cromosomas durante la reproducción sexual de los seres vivos. Las formas comunes de cruzamiento pueden ser:

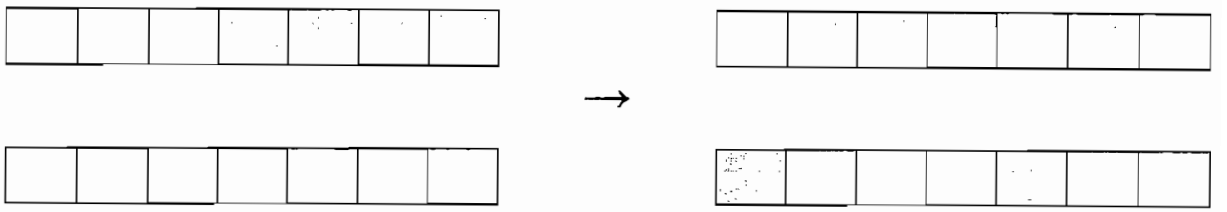
- Cruce de un punto, en el que se establece un punto de intercambio en un sitio aleatorio del cromosoma de los dos individuos; uno de los individuos contribuye todo su código anterior a ese punto y el otro individuo contribuye todo su código a partir de ese punto para producir el "hijo". Este tipo de cruce se puede apreciar en la figura 2.2

- Cruce multi-punto, en donde existen algunos puntos de cruce para intercambiar el contenido genético. Ver figura 2.3



Padres Descendencia

Figura 2.2 Cruce de un punto



Padres Descendencia

Figura 2.3 Cruce multi-punto

La probabilidad de tener un cruce (p_c) al igual que en la mutación, depende del tamaño de la población de la siguiente manera: [14]

- 0.6 para una población numerosa (100 individuos o más)
- 0.9 para una población reducida (30 individuos o menos)

A simple vista se ve que el cruzamiento es mucho más probable de ocurrir que la mutación.

Aquí también se puede hacer una distinción entre cruce para una representación binaria y una representación real. Cuando se dispone de una representación binaria, el cruce se lo realiza de la forma que se explicó en el párrafo anterior.

Para la representación real [12] se tiene una variación: si **c1** y **c2** son los individuos seleccionados para reproducirse (recordar que se trata de vectores), y **c1'** y **c2'** sus hijos, los valores de éstos últimos se hallan con las siguientes fórmulas de convergencia:

$$c_1' = \lambda * c_1 + (1-\lambda) * c_2$$

$$c_2' = (1-\lambda) * c_1 + \lambda * c_2$$

Ecuación 2.2

Donde λ es un valor que puede estar entre 0 y 1 y se conoce como *parámetro de cruce*.

2.4 OPTIMIZACIÓN EMPLEANDO ALGORITMOS GENÉTICOS

A continuación se presentan dos ejemplos de optimización empleando algoritmos genéticos para poner en práctica lo que se detalló en los subtemas anteriores y poder visualizar como funcionan los algoritmos genéticos ya en la práctica. El primer ejemplo es por demás sencillo y empleará la representación binaria mientras que en el segundo se procura resolver un problema mucho más complejo con dos variables y empleando una codificación real.

2.4.1 BÚSQUEDA DEL MÁXIMO GLOBAL DE f(x)

Considérese el problema de hallar el valor del máximo global de la siguiente función:

$$\mathbf{Z} \in \{0,..,31\} \quad \rightarrow \quad \mathbf{Z}$$

$$f: x \quad \rightarrow \quad f(x)=x^2$$

Como puede verse claramente, la solución es x=31; pero se toma este ejercicio sencillo para realizar a "mano" los procedimientos que en forma automática puede ser programada una computadora personal.

Como primer paso debe darse una representación apropiada al conjunto de valores que van a ser analizados; si se escoge una representación binaria salta a la vista que puede representarse cada número como un cromosoma formado por 5 bits dado que se tiene 32 valores ($32=2^5$).

Para la selección se empleará el método de ruleta. Los operadores genéticos a aplicarse son el cruce en un punto y mutación. Se escogerá una población de tamaño μ=4. En una primera etapa, se escoge aleatoriamente 4 números y se los evalúa; según se muestra en la tabla 2.1.

Individuo #	Cromosoma	Valor "x"	$f(x)=x^2$	Aptitud
1	01101	13	169	0,14
2	11000	24	576	0,49
3	01000	8	64	0,05
4	10011	19	361	0,31

Tabla 2.1 Evaluación de 4 individuos aleatorios

En el último campo se muestra la aptitud de cada individuo. Esto se halla evaluando la función a optimizar f(x) para cada individuo y sumando todas (169+576+64+361), para luego dividir cada valor para este total y así encontrar una medida proporcional de cuán óptimo es cada cromosoma. Puede verse que de esta población el individuo #2 es el más apto y tiene una probabilidad de reproducirse (aptitud) del casi 50% mientras que el individuo #3 es el menos apto con probabilidad de sobrevivir de 5%.

Un experimento randómico (de girar la ruleta 4 veces) puede dar como resultado el que el individuo #3 "muera", los individuos #4 y #1 sean elegidos para reproducirse mientras que el individuo #2 es tomado dos veces en cuenta. En base a esto, la generación siguiente se muestra en la tabla 2.2.

Individuo #	Cromosoma	Valor "x"	"Hijos"	Valor "x"	$f(x)=x^2$	Aptitud
1	01101	13	01100	12	144	0,08
2	11000	24	11001	25	625	0,36
2	11000	24	11011	27	729	0,42
4	10011	19	10000	16	256	0,15

Tabla 2.2 Primera generación de hijos

En la tabla 2.2 se muestra que se tomó a los individuos 1-2 y 2-4 respectivamente para cruzarlos; el punto de cruce seleccionado (señalado en **negrita**) se lo hace de manera randómica.

Se ve que en ambos casos de "reproducción" uno de los hijos es mejor que ambos padres lo cual implica un aumento en su aptitud. Este proceso se sigue repitiendo: tomar cuatro individuos según su aptitud y cruzarlos; hasta que se llegue a un punto en que ya no se pueda tener un individuo mejor; entonces allí finaliza el algoritmo y devuelve el valor máximo de "x" encontrado que se sabe será 31.

2.4.2 BÚSQUEDA DEL MÍNIMO GLOBAL DE f(x,y)

La representación binaria con mecanismo de selección proporcional, con cruce de un punto y mutación por inversión de bit (condiciones más simples posibles) no es siempre el método más adecuado y eficiente para resolver un problema [15]. A continuación se empleará, para hallar el mínimo absoluto de una función de dos variables, las siguientes condiciones:

- Población: 30 individuos
- Número máximo de generaciones: 100
- Rango inicial: [-1; 1]
- Representación real
- Selección por torneo con un tamaño de torneo (N) de 2
- Parámetro de cruce (λ) de 0.5
- Mutación acorde a una distribución probabilística uniforme

La función a optimizar es la siguiente:

$$z = f(x,y) = 20 + x^2 + y^2 - 10[\cos(2\pi x) + \cos(2\pi y)] \qquad \text{Ecuación 2.3}$$

A esta función se le conoce con el nombre de la *función de Rastrigin* [16] y puede apreciarse en la figura 2.4.

Para la resolución de este problema, al tratarse de uno más complejo que el anterior, se empleará el toolbox de "Algoritmos Genéticos y Búsqueda Directa" del programa MatLab. En este ejercicio se explicará muy superficialmente cómo funciona este software para sólo centrar el estudio al trabajo del algoritmo genético. En el Capítulo 5 se profundiza en detalle la aplicación de dicho *toolbox*.

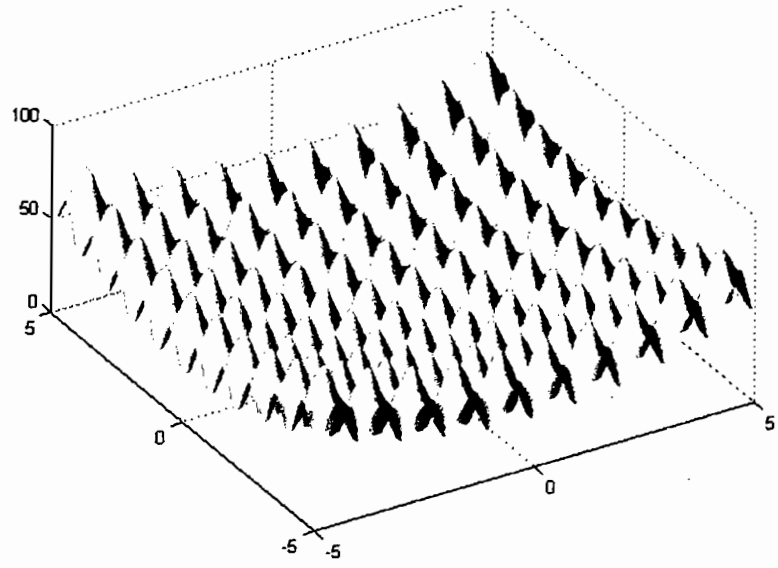


Figura 2.4 Función de Rastrigin

Una vez abierto la GUI (interfaz de usuario) de Algoritmos Genéticos del MatLab, escribiendo en la ventana de trabajo: *gatool*, se ingresa a las opciones que se desean aplicar en la resolución del problema. Este GUI se puede apreciar en la figura 2.5.

De antemano se conoce que el mínimo absoluto de esta función se encuentra en el punto (0,0) en donde el valor de la función es de 0, [16]; se verá cómo el programa se acerca bastante a estos puntos en cinco experimentos.

En la tabla 2.3 se muestra los valores finales de cada variable y su correspondiente valor de aptitud (mínimo alcanzado dentro de ese experimento) para 5 experimentos. De los cinco experimentos, el quinto presenta el mínimo valor de la función de aptitud (z = 0.000193) con los individuos: x = -0.000827; y = 0.0005384.

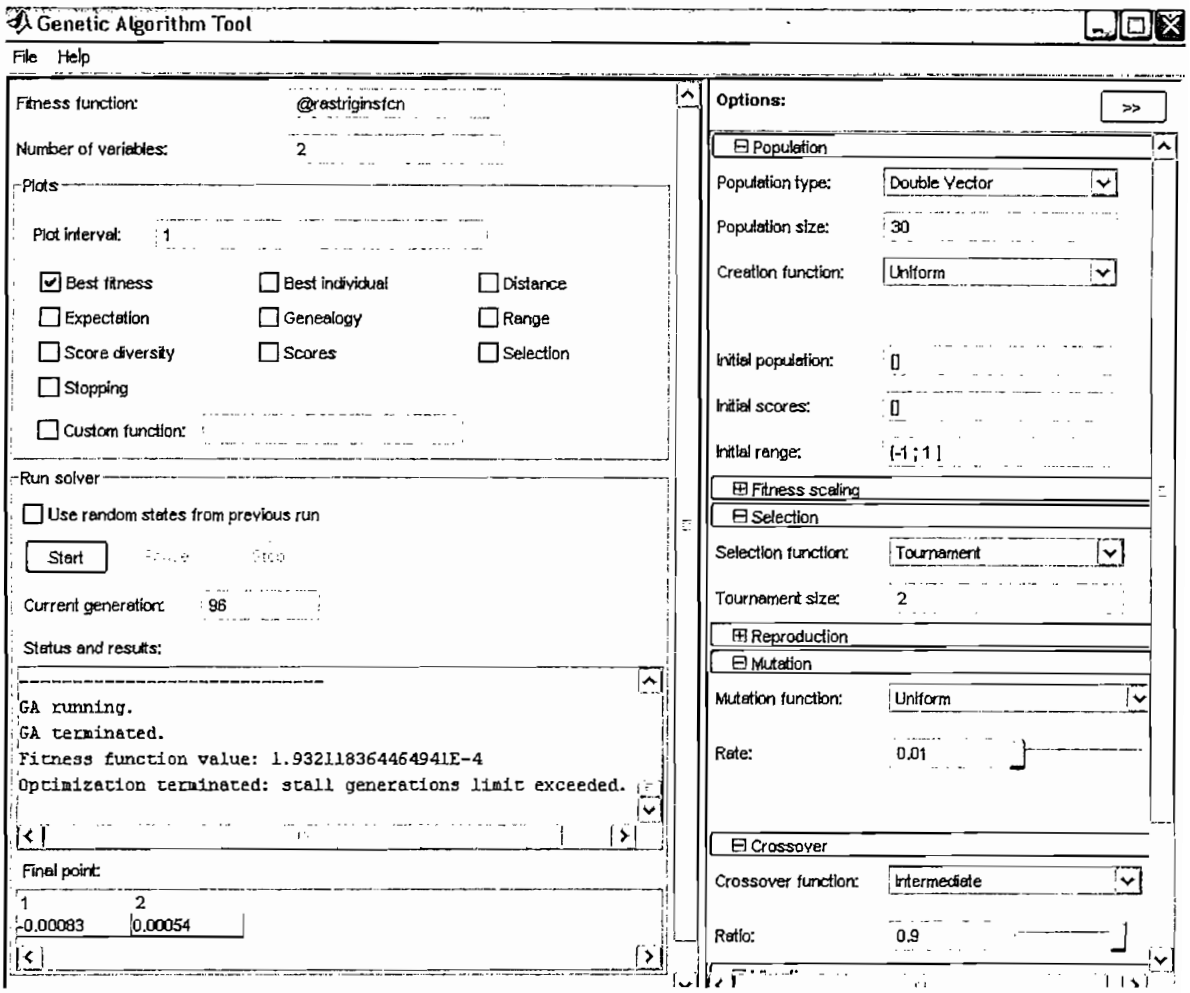


Figura 2.5 GUI del *toolbox* de Algoritmos genéticos del MatLab

Experimento	x mínimo	y mínimo	f(x,y) mínimo
1	$3*10^{-5}$	-0.032	0.201
2	$1.13*10^{-9}$	0.0048	0.0046
3	0.0553	-0.0157	.6487
4	0.0088	$1.382*10^{-8}$	0.0155
5	$-8.2701*10^{-4}$	$5.3847*10^{-4}$	$1.9321*10^{-4}$

Tabla 2.3 Resultados de optimización de la función Rastrigin

El *toolbox* empleado, permite también la posibilidad de observar algunos gráficos del desarrollo del problema. Por ejemplo en la figura 2.6 se muestra la gráfica de cómo va "evolucionando" la respuesta a lo largo de las generaciones para el caso del experimento #5. Los puntos azules representan el promedio de la aptitud de toda la población en cada generación mientras que los puntos negros representan la mejor aptitud de cada generación.

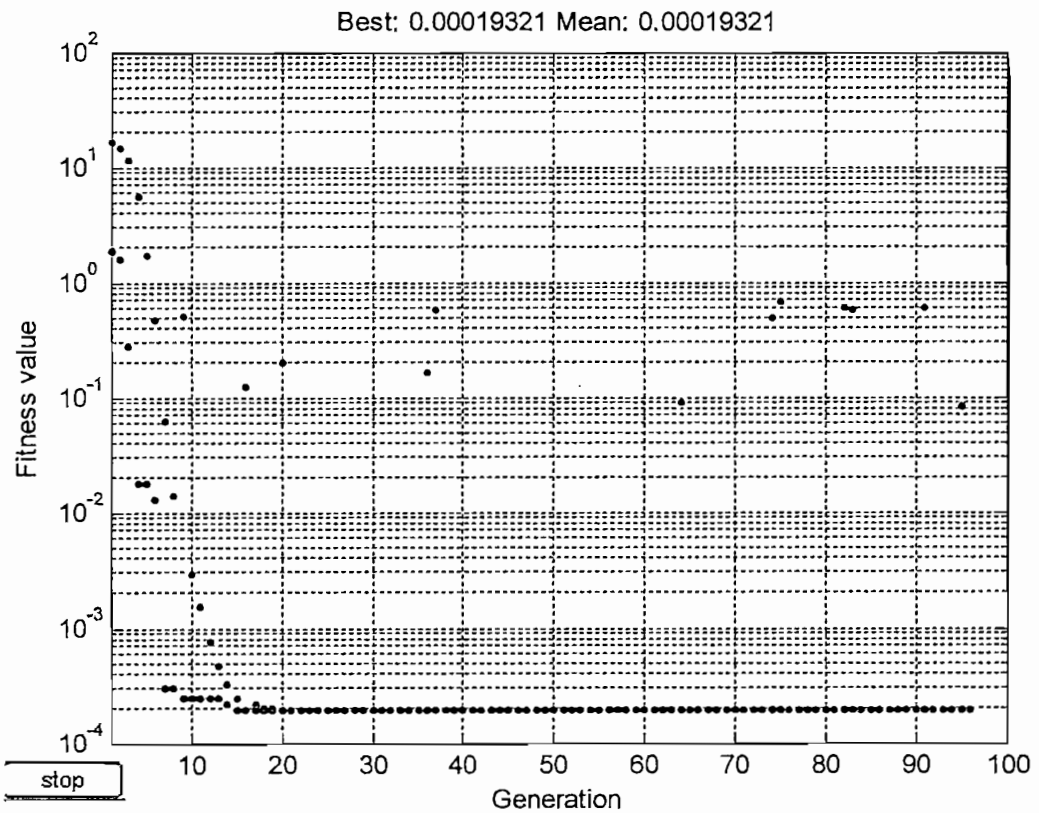


Figura 2.6 Aptitud a lo largo de las generaciones para el experimento #5

CAPÍTULO 3

DISEÑO DE CONTROLADORES

3.1 INTRODUCCIÓN

El campo del diseño del control es verdaderamente amplio; tanto en el control clásico como en el control moderno se tiene diversos métodos para diseñar controladores para mejorar el comportamiento de sistemas.

Dentro del control clásico se puede compensar una planta por medio de controladores en cascada o serie y por realimentación o lazo menor. Entre los controladores en cascada puede diseñarse redes de compensación: en adelanto si se necesita mejorar la respuesta transitoria, en atraso si se requiere mejorar el error en estado estable y en atraso-adelanto para satisfacer ambas respuestas.

Además se tiene los tradicionales y muy extendidos controladores PID que se discutirán en la Sección 3.2. La compensación por lazo menor puede ser tacométrica o realimentación de estado, técnica referida a variables de estado cuyo campo de estudio se desarrolló por la década de los 60 con el desarrollo del control moderno.

Existen varios métodos de diseño y sintonización de controladores PID, como las reglas de Ziegler-Nichols, lugar geométrico de las raíces, diseño en frecuencia, u otros métodos que no es objeto del presente trabajo estudiarlos.

Dentro de las nuevas técnicas del control avanzado, como se mencionó anteriormente, está el diseño en el espacio de estado con la realimentación y los observadores de estado y el diseño de controladores robustos, adaptativos, predictivos y otros.

Dadas las tendencias actuales del control, en este trabajo se pretende trasladar el diseño de un controlador a un problema de optimización para mejorar el comportamiento de dicho controlador. Se va a optimizar controladores PID diseñados previamente con algún método clásico y se diseñarán controladores robustos.

3.2 CONTROLADORES PID

En la actualidad más de la mitad de los controladores en procesos industriales en el mundo son del tipo PID debido a su facilidad de diseño, simplicidad de funcionamiento, por su bajo costo de implementación [18],y además por que presentan un comportamiento robusto [9]. Gracias a su gran versatilidad se puede decir que este controlador seguirá teniendo relevancia y popularidad a través de los años por lo que se justifica su estudio dentro de este proyecto.

El controlador PID es simplemente un controlador de segundo orden que tiene la función de transferencia indicada en la ecuación 3.1

$$G_C(s) = Kp + \frac{Ki}{s} + Kd\ s = \frac{s^2 Kd + s\,Kp + Ki}{s} \qquad \text{Ecuación 3.1}$$

Donde:

- K_P = ganancia proporcional
- K_I = ganancia integral
- K_D = ganancia derivativa

El controlador PID trabaja en un sistema en lazo cerrado como el de la figura 3.1 de la siguiente forma: la señal del error (*e*) ingresa al controlador, éste procesa esta información así como la derivada y la integral de esta señal. La señal de control (*u*) corresponderá a K_P veces la magnitud del error más K_I veces la integral del error más K_D veces la derivada del error. Esto se puede ver en

forma resumida en la ecuación 3.2 (notar que es simplemente la ecuación en el dominio del tiempo de la cual se tomó la transformada de Laplace para hallar la función de transferencia)

$$u = K_P * e + K_I * \int edt + K_D * \frac{de}{dt} \qquad \text{Ecuación 3.2}$$

Es importante mencionar que el controlador PID es en sí un controlador robusto gracias a las tres acciones de control que provee al sistema original. La acción proporcional (K_P) provee una contribución que depende del valor instantáneo del error. Tiene la capacidad de reducir el efecto de incertidumbres en el modelo así como también el tiempo de subida y el error en estado estable (sólo reducir, no eliminar) a más de rechazar perturbaciones (reducir el efecto de las perturbaciones).

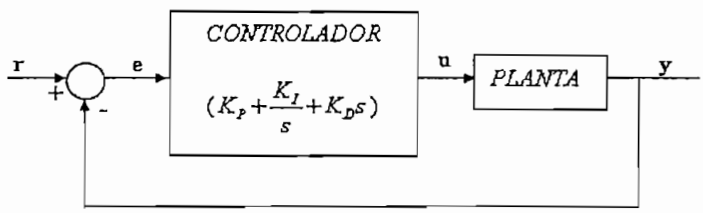


Figura 3.1 Planta y controlador PID en cascada

La acción integral, (K_I) por otro lado, brinda una salida del controlador proporcional al error acumulado lo que implica que tendrá una reacción lenta. Esta acción da un error en estado estable nulo (gracias al polo que introduce en el origen) aunque empeora la respuesta transitoria.

La acción derivativa (K_D) actúa sobre la razón de cambio del error, por lo que tendrá una reacción rápida. A esta acción se le llama también predictiva por cuanto depende de la tendencia del error; tiene el efecto de reducir el sobreimpulso (en general mejora la respuesta transitoria).

Más adelante se verá las características en frecuencia deseables de un controlador, características que tiene el controlador PID.

3.3 CONTROLADORES ROBUSTOS

Para poder diseñar controladores que sean robustos es necesario comprender qué es la incertidumbre, que fue descrita vagamente en la Sección 1.2. En esta sección se explica precisamente este concepto además de tipos de robustez, la robustez intrínseca de un sistema en lazo cerrado y un teorema útil para comprobar la robustez de sistemas.

3.3.1 INCERTIDUMBRE

El modelaje de sistemas usualmente introduce errores, los cuales son causados principalmente por:

- Diferencia entre los parámetros reales y los parámetros modelados.
- Modificaciones en la planta debidas al envejecimiento o cambio en las condiciones de trabajo o del medio ambiente.
- Simplificaciones o linealizaciones en el modelo

A estos errores se les conoce como incertidumbre del modelo; cuando se diseña controladores robustos, esta incertidumbre necesariamente debe ser considerada. Existen dos tipos de incertidumbre: la estructurada y la no estructurada.

La incertidumbre estructurada, o llamada también *incertidumbre paramétrica*, es causada por la variación de los parámetros del sistema y asume dentro del modelo a la incertidumbre, razón por la cual se provee de rangos y límites para parámetros inciertos del sistema, en otras palabras es cuando el diseñador conoce el rango de variación de un cierto parámetro, por ejemplo considérese la ecuación 3.3 que define el modelo de una planta.

$$G(s) = \frac{1}{s^2 + 2\xi s + 1}$$ Ecuación 3.3

Si se conoce que el valor de la constante ξ se encuentra en un rango conocido [$\xi_{mín}$, $\xi_{máx}$], entonces la planta G tiene una incertidumbre estructurada.

La incertidumbre no estructurada asume un menor conocimiento de la planta. Esta incertidumbre es provocada principalmente por no linealidades de la planta, modificaciones en el punto de operación o dinámicas no modeladas de la planta. En este trabajo se trabajará con la incertidumbre no estructurada por ser más crítica dentro de un sistema de control [10].

Los dos modelos más frecuentes para describir el comportamiento de una planta con incertidumbre son los modelos multiplicativo y aditivo. Para ambos modelos se tiene la siguiente notación:

- G(s) = planta
- G^(s) = planta perturbada
- Δ(s) = es un función de transferencia estable variable que representa la perturbación presente en el sistema.
- W (s) = es un función de transferencia estable fija que representa la ponderación o peso de la perturbación,

En las ecuaciones 3.4 y 3.5 se puede ver los modelos multiplicativo y aditivo respectivamente. El modelo multiplicativo es el más empleado [12] por lo que se lo empleará en el diseño futuro de los controladores, (los subíndices *m* y *a* de W son para diferenciar del modelo multiplicativo del aditivo)

$$\hat{G} = G*(1 + \Delta * Wm)$$ Ecuación 3.4

$$\hat{G} = G + \Delta * Wa$$ Ecuación 3.5

En las figuras 3.2 y 3.3 se pueden apreciar el diagrama de bloques para ambas representaciones de la incertidumbre.

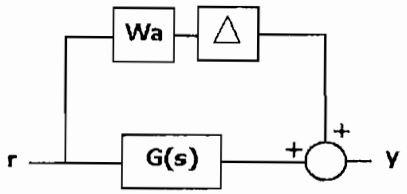


Figura 3.2 Incertidumbre aditiva

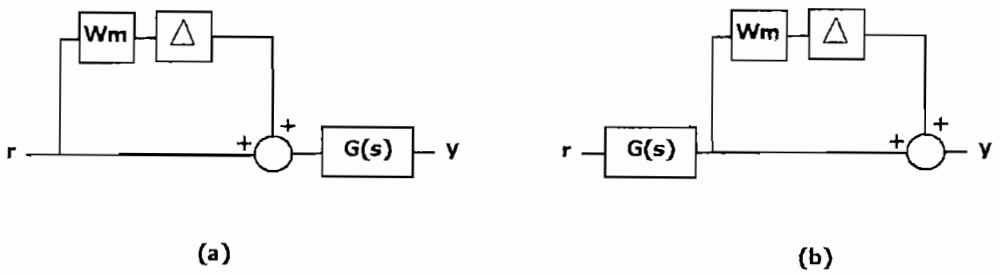


Figura 3.3 Incertidumbre multiplicativa: (a) a la entrada de la planta, (b) a la salida de la planta

La función Δ debe satisfacer la condición indicada en la ecuación 3.6 para que la perturbación sea *admisible*. En la ecuación 3.6 se presenta la norma H infinita que en la Sección 3.4 se explica en detalle. Además se asume que al formar G^ no se elimina ningún polo inestable de G, [10], [12].

$$\|\Delta\|_\infty \leq 1 \qquad \text{Ecuación 3.6}$$

Por lo tanto, la función W debe seleccionarse tal que se cumpla con la ecuación 3.7; esta expresión es muy útil pues permite dar un límite a la perturbación con la que se diseñará los controladores robustos.

$$\frac{\hat{G}}{G} - 1 = \Delta W_m$$

Ecuación 3.7

$$\left| \frac{\hat{G}}{G} - 1 \right| \leq |W_m|$$

En la figura 3.4 se ve la forma típica que presenta la incertidumbre multiplicativa [1]; a bajas frecuencias la incertidumbre es muy poca mientras que a mayores frecuencias la incertidumbre aumenta considerablemente empobreciendo el conocimiento del modelo

.

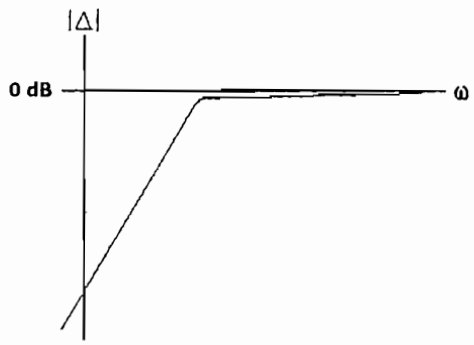


Figura 3.4 Forma típica de la incertidumbre multiplicativa

3.3.2 CONTROLADOR ROBUSTO EN ESTABILIDAD Y EN RECHAZO A PERTURBACIONES

La condición de robustez de un controlador como el de la figura 3.5 puede definirse de la siguiente manera: si la planta G(s) pertenece a una familia de plantas $G(s)$ y se considera alguna característica deseada como por ejemplo estabilidad interna del sistema, entonces se puede decir que el controlador C(s) es robusto si satisface dicha característica para todas las plantas que pertenecen a $G(s)$.

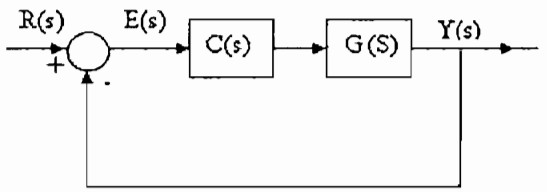


Figura 3.5 Sistema compensado en lazo cerrado

La noción de *familia de plantas* se refiere a la consideración de perturbaciones, sean estructuradas o no estructuradas, en la planta de estudio, por lo que deja de ser una sola planta y pasa a ser realmente un conjunto de plantas a considerar.

En base a lo explicado en el párrafo anterior, se dice que un controlador C(s) es establemente robusto o que provee a un sistema de robustez de estabilidad cuando brinda estabilidad interna para todas las plantas de $G(s)$.

Existen algunos métodos para diseñar este tipo de controladores pero en este trabajo se empleará la norma H infinita pues esta técnica se presta perfectamente al empleo de los algoritmos genéticos. Este diseño se explica en la Sección 3.5.

Por otra parte, un controlador C(s) es robusto en comportamiento o rechazo a perturbaciones cuando la planta G(s) ante una perturbación $D_y(s)$ externa no sufre variaciones considerables en su salida ni en su estabilidad. Ver figura 3.6. Esto implica que el sistema cumpla tanto las especificaciones en estado estable (problema del seguimiento) como en la respuesta transitoria.

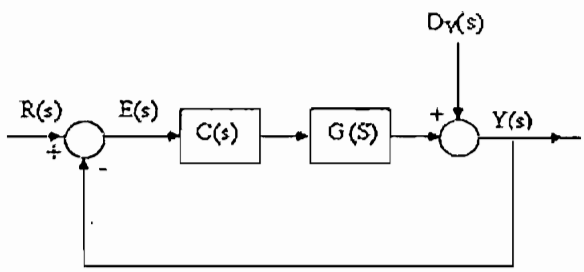


Figura 3.6 Sistema con perturbación a la salida de la planta

De igual manera, se empleará la norma H infinita para el diseño de estos controladores tal como se indica en la Sección 3.5.

3.3.3 ROBUSTEZ DEL SISTEMA EN LAZO CERRADO

Un sistema en lazo cerrado o con realimentación negativa; es en sí un sistema robusto. Este tipo de sistemas presentan dos propiedades importantes que le hacen robusto: [10]

1. Reducción de la sensibilidad del sistema a incertidumbres o variaciones de los elementos o dinámicas no modeladas.
2. Rechazo a perturbaciones, al reducir o eliminar los efectos indeseables de las perturbaciones.

Estas dos propiedades se demuestran a continuación. La sensibilidad es el cambio que sufre la salida del sistema a una pequeña variación en la función de transferencia del proceso; considerando el diagrama de bloques de la figura 3.7 y 3.8 la sensibilidad Δy es para cada caso lo indicado en las ecuaciones 3.8 y 3.9 respectivamente.

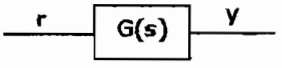


Figura 3.7 Sistema en lazo abierto

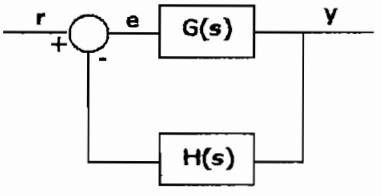


Figura 3.8 Sistema en lazo cerrado

$$\Delta y = \Delta G * r \qquad \text{Ecuación 3.8}$$

$$\Delta y = \frac{\Delta G}{(1+GH)^2} * r \qquad \text{Ecuación 3.9}$$

Las demostraciones de las expresiones se las puede desarrollar fácilmente recordando a que es igual la ganancia en lazo abierto y cerrado y el hecho de que $\Delta G << G$.

Se puede ver que en lazo abierto una variación en la planta G se refleja directamente a la salida, mientras que para el sistema en lazo cerrado su efecto se atenúa en un factor de $(1+GH)^2$; esto muestra que un sistema en lazo cerrado es mucho más insensible a perturbaciones comparado con un sistema en lazo abierto.

Ahora bien, para analizar el rechazo a perturbaciones analícese los sistemas de las figuras 3.9 y 3.10

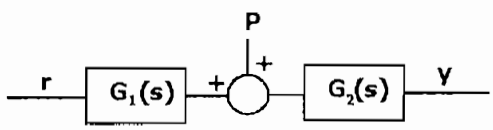


Figura 3.9 Sistema en lazo abierto

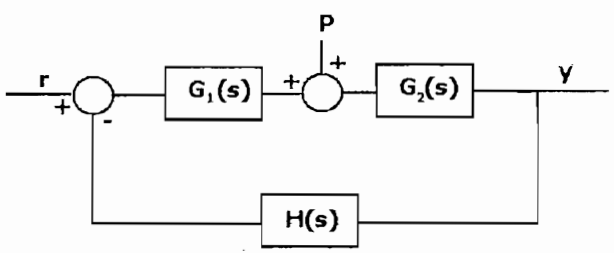


Figura 3.10 Sistema en lazo cerrado

Para ver como influye en cada caso la perturbación *P* a la salida del sistema, considérese a *P* como la entrada; con estas condiciones y sabiendo que

$G_1G_2H >> 1$ se llega a las expresiones indicadas en las ecuaciones 3.10 y 3.11, (nótese que por simplicidad se asumió que r = 0)

$$y_p = G_2 * P \qquad \text{Ecuación 3.10}$$

$$y_p = \frac{1}{G_1 H} * P \qquad \text{Ecuación 3.11}$$

Una vez más se ve que en un sistema en lazo cerrado se minimiza el efecto de perturbaciones (en un factor de G_1H) mientras que en el de lazo abierto este efecto aumenta. Esto significa que un sistema en lazo cerrado es más inmune a perturbaciones en la planta.

Para que un sistema en lazo cerrado (como el de la figura 3.8) sea robusto es deseable que su función de transferencia en lazo abierto *G* tenga la respuesta en frecuencia indicada en la figura 3.11

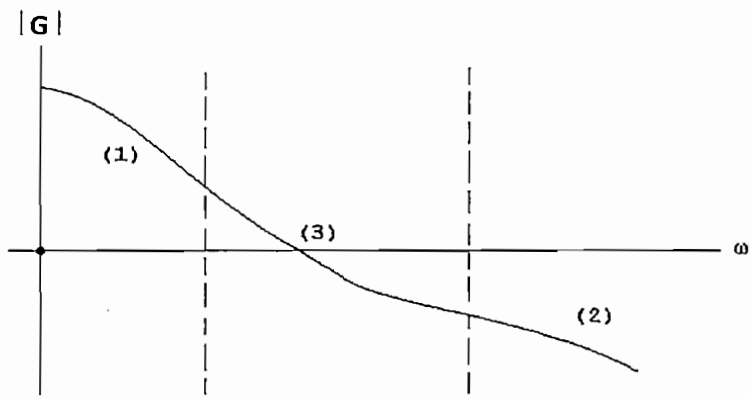


Figura 3.11 Forma deseada de la ganancia en lazo abierto

Las características de esta ganancia son: a frecuencias bajas, la ganancia en lazo abierto debe ser alta (1) para tener buen seguimiento y rechazo a perturbaciones, mientras que para las frecuencias altas esta ganancia debe ser baja (2) para atenuar el ruido y para tener estabilidad relativa [1] la pendiente con que cae la curva (3) no debe sobrepasar los -40dB siendo deseable un descenso suave de -20dB.

Se puede especificar también, en forma separada, la forma deseable de las funciones *S*(s) y T(s) del sistema. En base a la figura 3.8 se define *S* y *T* como:

$$S = \frac{1}{1+\mathrm{GH}} = \frac{\mathrm{e(s)}}{\mathrm{r(s)}} \qquad \text{Ecuación 3.12}$$

$$T = \frac{\mathrm{GH}}{1+\mathrm{GH}} = \frac{\mathrm{y(s)}}{\mathrm{r(s)}} \qquad \text{Ecuación 3.13}$$

Estas funciones reciben el nombre de sensibilidad (S) y sensibilidad complementaria (T).

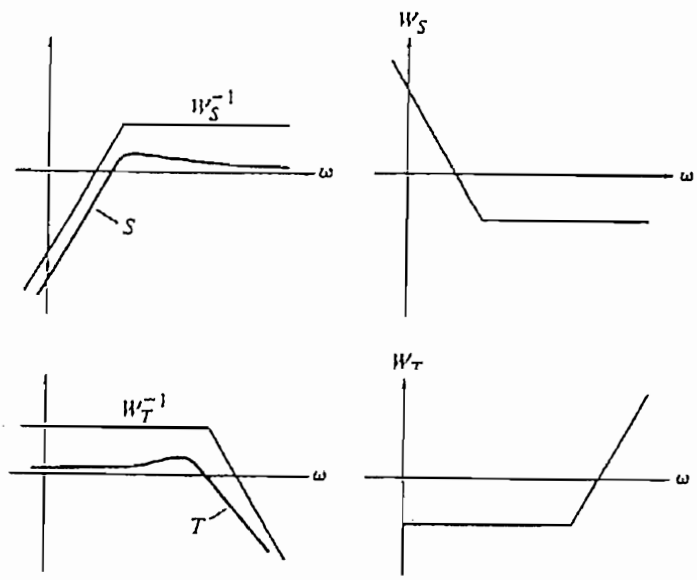


Figura 3.12 Formas deseadas (S) y (T) [1]

El comportamiento nominal de sistemas con realimentación puede ser traducido generalmente en especificaciones de sensibilidad de S(s) y T(s), por ende una caracterización de robustez implica obtener formas similares a las mostradas en la figura 3.12. En base a esto, se procura incluir compensadores o *ponderaciones*, tal que después de hallar límites adecuados, den la forma deseada a ambas funciones (sin salir de sus respectivos límites lógicamente). Estas funciones son W_T y W_S cuyas funciones inversas se muestran también en la

figura 3.12. En diagrama de bloques, se tiene el esquema indicado en la figura 3.13

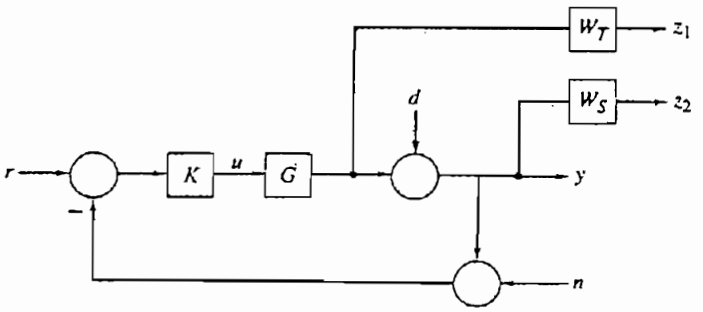


Figura 3.13 Diagrama de bloques para *moldear el lazo*

Donde:

- *d* = perturbaciones de entrada que el sistema debe ser capaz de rechazar; son perturbaciones propias de los actuadores.
- *n* = ruido generalmente introducido por sensores o mediciones

Para el diseño se emplea las expresiones 3.14 y 3.15 que pueden ser deducidas de la figura 3.12.

$$|S| \leq W_S^{-1} \quad \rightarrow \quad |W_S \; S| \leq 1 \qquad \text{Ecuación 3.14}$$

$$|T| \leq W_T^{-1} \quad \rightarrow \quad |W_T \; T| \leq 1 \qquad \text{Ecuación 3.15}$$

A todo este método se le conoce con el nombre de Moldeado del lazo, (*loop shaping* en inglés)

3.3.4 TEOREMA DE LA PEQUEÑA GANANCIA [1]

El teorema de la pequeña ganancia permite conocer si un sistema es establemente robusto. Este teorema expone que si para todo valor de frecuencia la magnitud del inverso de T(s) es mayor que la magnitud de la función de

transferencia de la incertidumbre Δ(s), entonces el sistema es establemente robusto. Esto se muestra en la ecuación 3.16.

$$|\Delta| < \frac{1}{T} \qquad \text{Ecuación 3.16}$$

Debe ponerse atención al hecho de que la condición indicada en la ecuación 3.16 es suficiente pero no necesaria, para demostrar que un sistema tiene robustez de estabilidad. Este teorema se puede demostrar partiendo del conocimiento de que, si en un sistema como el de la figura 3.8 tanto G(s) como H(s) son estables, luego el sistema en lazo cerrado será estable si se cumple la condición: | G(s) H(s) | < 1, es decir la condición que | G(s) | < 1 / | H(s) | .

Para ello es necesario simplificar el sistema con incertidumbre (figura 3.14 (a)) a una estructura de dos bloques, esto se consigue hallando la función de transferencia *vista* por la incertidumbre (b) que corresponde a la función de transferencia (figura 3.14 (c)) mostrada en la ecuación 3.17.

$$M(s) = \frac{-G(s)H(s)}{1+G(s)H(s)} \qquad \text{Ecuación 3.17}$$

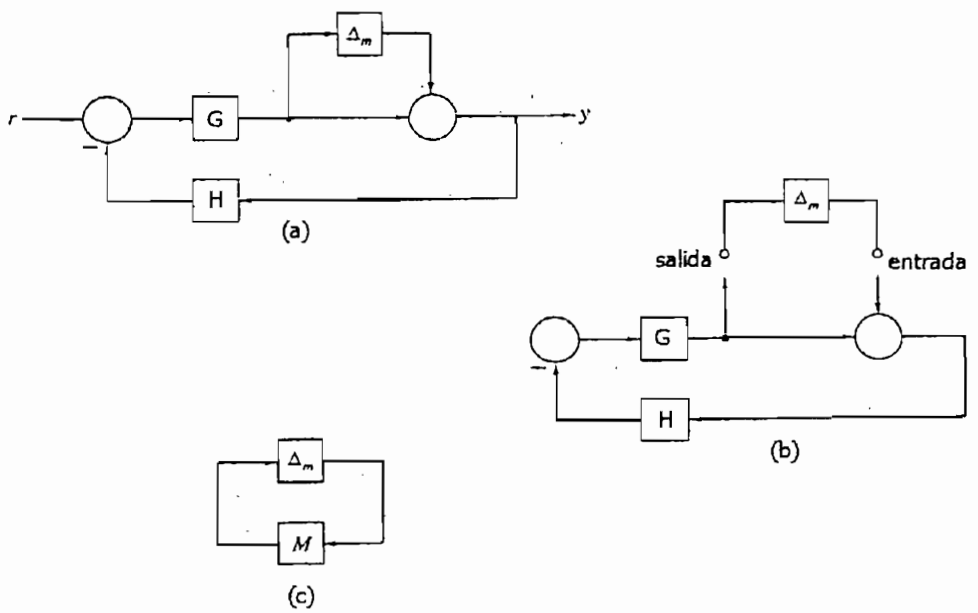


Figura 3.14 Sistema en lazo cerrado con incertidumbre multiplicativa [1]

Nótese que la función de transferencia M es simplemente el inverso de T(s) del sistema en cuestión.

3.4 NORMAS 2 E INFINITAS

Una norma es una *herramienta matemática* para cuantificar elementos de espacios tales como: vectores, matrices, funciones, sistemas, etc. Las normas son de vital importancia en el análisis matemático puesto que permiten comparar distintos objetos.

A continuación se abordará dos normas fundamentales: la norma 2 y la norma infinita. Es importante destacar que cuando el objeto al que se halla su norma es una *señal*, toma el nombre de norma L (en honor al matemático Lebesgue [12]) mientras que si se trata de un *sistema* se le conoce como norma H (en honor al matemático Hardy [12]).

Sea *u(t)* una función del tiempo, entonces la norma L_2 y L^∞ están definidas por las expresiones indicadas en las ecuaciones 3.18 y 3.19

$$\| u \|_2 := \sqrt{\int_{-\infty}^{\infty} u(t)^2 \, dt} \qquad \text{Ecuación 3.18}$$

$$\| u \|_\infty := \sup_t | u(t) | \qquad \text{Ecuación 3.19}$$

Ahora, si *G(jw)* es un sistema en función de la frecuencia, la norma H_2 y H^∞ están definidas por las expresiones indicadas en las ecuaciones 3.20 y 3.21

$$\| G \|_2 := \sqrt{\frac{1}{2\pi} \int_{-\infty}^{\infty} | G(j\omega) |^2 \, d\omega} \qquad \text{Ecuación 3.20}$$

$$\| G \|_{\infty} := \sup_{\omega} |G(j\omega)| \qquad \text{Ecuación 3.21}$$

La palabra *sup* es la abreviación de supremo y representa el valor máximo del objeto analizado incluido los límites del rango, por eso es que se le llama también el *mínimo borde superior.*

Las normas H_2 y H^{∞} son de utilidad en el control, ya que permiten dar una representación física o más tangible de las características de los sistemas. Por ejemplo se aprecia que la norma H_2 se asemeja a la expresión del valor RMS, es más, en base a algunos conceptos de densidad espectral de potencia puede inferirse que la norma H_2 de un sistema representa el valor RMS de la salida de dicho sistema cuando la excitación de entrada es ruido blanco [1].

Por otro lado, la norma H ∞ de un sistema representa la máxima amplitud en el diagrama de Bode, también representa la distancia desde el origen hasta el punto más lejano en el plano complejo del diagrama de Nyquist del sistema.

La norma H ∞ es de mucha utilidad en el diseño de controladores robustos como se verá en la Sección 3.4 y también para proporcionar una cuantificación de cuán robusto es un sistema: como el teorema de la pequeña ganancia dice que para que un sistema tenga robustez de estabilidad Δ debe ser menor o igual al inverso de la función T (referirse a la ecuación 3.16), el caso crítico será cuando T sea máximo es decir el supremo de T que no es otra cosa que la norma H infinita de T. A este valor se le conoce como *margen de estabilidad multiplicativa* (MSM), ver ecuación 3.22.

$$\text{MSM} = \frac{1}{\| T \|_{\infty}} \qquad \text{Ecuación 3.22}$$

El MSM representa entonces, el mínimo valor de incertidumbre que desestabilizará al sistema en análisis.

Un teorema muy interesante que relaciona a ambas normas, norma 2 y norma infinita, es el descrito por la ecuación 3.23.

$$\max \frac{\|y\|_2}{\|r\|_2} = \|G\|_\infty \quad ; G(s) = \frac{Y(s)}{R(s)} \qquad \text{Ecuación 3.23}$$

Este teorema es válido siempre y cuando:

- G(s) sea estable y sin polos en el eje imaginario, y
- $\|r\|_2$ sea finita.

3.5 CONTROLADORES ROBUSTOS CON NORMAS H ∞ [12] [10]

Tras haber estudiado las bases de los controladores robustos y de la norma H ∞, es posible unificar ambos criterios y analizar en qué forma estas normas expresan la robustez de un sistema y a su vez cómo pueden aplicarse para el diseño de ellos.

3.5.1 DISEÑO DE CONTROLADORES ROBUSTOS EN ESTABILIDAD

Considérese el sistema de control indicado en la figura 3.15. Como se está diseñando el controlador, no se conocen aún los valores de sus constantes por lo que su función de transferencia es C(s,k). La planta G(s) corresponde a la planta perturbada cuya planta original es $G_0(s)$, revisar Sección 3.3.1. Como se dijo anteriormente, se está empleando el modelo multiplicativo de la incertidumbre.

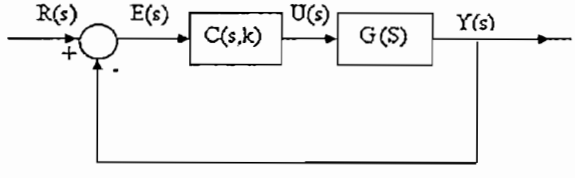


Figura 3.15 Sistema con una planta con incertidumbre estructurada y controlador de estructura fija [12]

La condición para que este sistema sea robusto en el sentido de la estabilidad es: si la función de ponderación o peso de la incertidumbre $W_m(s)$ es estable y acotada y G(s) no se formó cancelando polos inestables de la planta original $G_0(s)$; entonces, si el sistema de control nominal de control es estable sin la presencia de perturbaciones ($\Delta = 0$), el controlador C(s,k) garantiza la robustez en estabilidad si y solo si, se cumple la condición presentada en la ecuación 3.24

$$\left\| \frac{C(s,k)\ G_0(s)\ W_m(s)}{1+C(s,k)\ G_0(s)} \right\|_\infty < 1 \qquad \text{Ecuación 3.24}$$

Empleando la definición de la norma H ∞ (ecuación 3.21) se llega a la expresión indicada en la ecuación 3.25.

$$\sup \left| \frac{C(s,k)\ G_0(s)\ W_m(s)}{1+C(s,k)\ G_0(s)} \right| < 1 \qquad \text{Ecuación 3.25}$$

Reemplazando *s* por *jω* y hallando la magnitud del miembro izquierdo se llega a la ecuación 3.26.

Ecuación 3.26

$$\sup \sqrt{\frac{C(j\omega,k)\ G_0(j\omega)\ W_m(j\omega)\quad C(-j\omega,k)\ G_0(-j\omega)\ W_m(-j\omega)}{[1+C(j\omega,k)\ G_0(j\omega)]\,[1+C(-j\omega,k)\ G_0(-j\omega)]}} < 1$$

Finalmente, por facilidad, se le llama a la expresión dentro del radical como la función *α(jω,k)*, siendo entonces la ecuación 3.27 la condición de diseño que debe cumplir el controlador C(jω,k) para que el sistema sea establemente robusto.

$$\max_{\omega \in \Re^+} \sqrt{\alpha(\omega,k)} < 1 \qquad \text{Ecuación 3.27}$$

3.5.2 DISEÑO DE CONTROLADORES ROBUSTOS EN RECHAZO A PERTURBACIONES

En métodos clásicos de diseño de controladores se suele asumir perturbaciones con señales determinísticas como el escalón unitario, la rampa, una sinusoide, etc., sin embargo al emplear el criterio de la norma H ∞ las perturbaciones pueden tener cualquier tipo de señal siempre y cuando se asegure que la amplitud de dicha señal se encuentre acotada.

Considérese el sistema planteado en la figura 3.16; la perturbación es la señal $D_y(t)$ presente a la salida de la planta, C(s,k) es el controlador y la planta está descrita por su función de transferencia nominal $G_0(s)$.

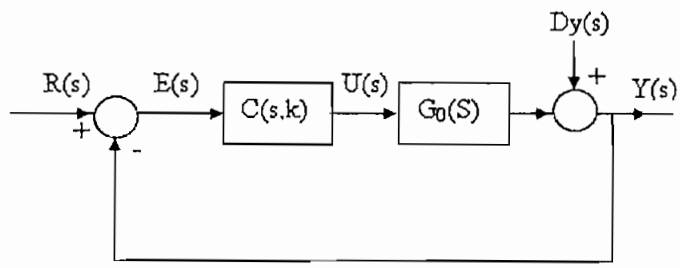


Figura 3.16 Sistema con perturbación a la salida de la planta [12]

Para analizar el efecto de la perturbación en la salida del sistema, se encuentra la función de transferencia o la ganancia de la perturbación.

$$\frac{Y(s)}{D_y(s)} = \frac{1}{1+G_0(s)C(s,k)} \qquad \text{Ecuación 3.28}$$

En la ecuación 3.28 se tiene la relación salida – entrada del sistema tomando como entrada la perturbación $D_y(t)$ y asumiendo por simplicidad que R(s)=0. A esta expresión se le puede aplicar el teorema expuesto en la ecuación 3.23. Como la máxima amplitud de la salida y(t) causada por la perturbación $D_y(t)$ no debe sobrepasar un cierto nivel deseado de atenuación [8] (ζ por ejemplo) se obtiene la expresión de la ecuación 3.29.

$$\text{máx}\ \frac{\|y\|_2}{\|d_y\|_2} = \left\| \frac{1}{1 + C(s, k)G_0(s)} \right\|_\infty < \xi \qquad \text{Ecuación 3.29}$$

$$\left\| \frac{W_d(s)}{1 + C(s, k)G_0(s)} \right\|_\infty < \xi \qquad \text{Ecuación 3.30}$$

Debe notarse que en la ecuación 3.30 se añadió además un filtro W_d (pasabajos) para atenuar la perturbación [8] cuya forma típica es la de la figura 3.4. Por lo tanto, para diseñar un controlador que sea robusto en rechazar perturbaciones, es necesario cumplir con la condición planteada en la ecuación 3.30.

Si se aplica la definición de la norma H ∞ a la ecuación 3.30 (haciendo algo similar a lo desarrollado en la Sección 3.5.1) se llega a obtener la ecuación 3.31, siendo el valor de la función *β (jω,k)* lo indicado en la ecuación 3.32.

$$\text{máx}_{\omega \in \Re^+} \sqrt{\beta(\omega, k)} < \xi \qquad \text{Ecuación 3.31}$$

Ecuación 3.32

$$\beta(j\omega k) = \frac{W_d(j\omega)\ \ W_d(-j\omega)}{[1 + C(j\omega, k)\, G_0(j\omega)]\,[1 + C(-j\omega, k)\, G_0(-j\omega)]}$$

La ecuación 3.31 es entonces la condición de diseño que debe cumplir el controlador C(jω,k) para que el sistema pueda ser robusto en rechazo a perturbaciones.

De todo lo desarrollado anteriormente se nota que para el diseño de ambos tipos de controladores robustos, el problema se reduce a encontrar el máximo de

una función con restricciones, problema que puede ser resuelto empleando la técnica de los algoritmos genéticos como se verá en el Capítulo 4 y 5.

CAPÍTULO 4

Aplicación de los Algoritmos Genéticos en los Controladores óptimos y robustos

4.1 INTRODUCCIÓN

Antes de iniciar con el estudio de cómo pueden los algoritmos genéticos emplearse en el diseño u optimización de controladores, es necesario realizar primeramente un enfoque a lo que significa *óptimo* dentro de los sistemas de control.

4.1.1 OPTIMALIDAD [24]

En la práctica nunca se podrá obtener un control perfecto, como se dijo en el Capítulo 1, se debe llegar a compromisos entre las distintas características del sistema de control [11]: entre tener gran estabilidad y buen comportamiento, buen seguimiento y adecuado rechazo a perturbaciones, etc.

Por esta razón, un buen control es un criterio subjetivo, que depende de las condiciones reales que son deseables que tengan la planta de estudio. Sin embargo; siempre se considera para cualquier diseño, los siguientes parámetros de todo el sistema de control:

- Características inherentes del sistema a controlar
- Restricciones impuestas por requerimientos operativos
- Costo de conseguir objetivos deseados de control

Si se consigue diseñar un controlador que satisfaga estas restricciones, se dice que el controlador es óptimo.

Cuando se busca la optimalidad de un sistema, se juzga normalmente su desempeño (*performance* en inglés) en base a ciertos criterios de diseño que normalmente son expresiones matemáticas que toman el nombre de: *función de costo* o *función objetivo*. Estas funciones deben dar información respecto a las restricciones del sistema en cuestión.

El diseño de un controlador óptimo se obtiene pues, resolviendo el problema de optimización que generan las restricciones del sistema; algunas restricciones típicas de los sistemas de control son las presentadas a continuación:

- Restricciones duras: máximo y mínimo valor de las entradas y salidas controladas.
- Restricciones suaves: máxima taza de cambio de las entradas
- Restricciones asociadas: limitaciones en las variables que no se controlan directamente

4.1.2 ALGORITMOS GENÉTICOS EN DISEÑO DE CONTROLADORES CON NORMAS H ∞ [8], [24]

Los métodos convencionales de diseño empleando la técnica de las normas H son bastante complicadas y muy difíciles de implementar en la práctica en la industria, por esta razón es que en las últimas décadas dentro del control han surgido y se han desarrollado con fuerza nuevas ideas y herramientas que han contribuido a facilitar este tipo de diseños. Entre algunas de estas técnicas está el control difuso, los algoritmos genéticos, la programación evolutiva, las redes neuronales y otros.

Algunos métodos clásicos de búsqueda como algoritmos basados en el gradiente o algoritmos de búsqueda randómica podrían pensarse que servirían también para el problema de minimizar las funciones que dan robustez a un sistema, sin embargo los algoritmos genéticos son mucho mejores que ellos, sobretodo por las razones detalladas a continuación:

- Los algoritmos genéticos trabajan con arreglos de individuos, es decir evalúa simultáneamente varios puntos en el espacio de búsqueda. Al realizar la búsqueda de forma paralela, los algoritmos genéticos difícilmente convergerán a un mínimo local, si no más bien al mínimo global; cosa que no ocurre con la mayoría de técnicas de búsqueda que la realizan en base a un solo punto.

- Los algoritmos genéticos necesitan solamente tener información de la calidad de la solución provista por cada controlador (individuo), o sea el valor de su función de aptitud; mientras que los métodos tradicionales requieren mayor información como existencia de diferenciabilidad o incluso la estructura global del problema y sus parámetros. Esto hace que los algoritmos genéticos sean mucho más flexibles.

Puede decirse que el diseño de controladores robustos por medio de los algoritmos genéticos viene a ser un nexo entre el fuerte contenido matemático del control robusto H con los controladores típicos y clásicos de la industria como los PID, redes de compensación, etc.

El diseño propuesto en el presente trabajo es el de hallar un controlador (o partiendo de algún diseño previo) que permita estabilizar al sistema y satisfacer características deseables, en base al cumplimiento de desigualdades restrictivas de la norma H infinita aplicada al sistema en lazo cerrado (referirse al Capítulo 3), ya sea para robustez de estabilidad o robustez en rechazo a perturbaciones.

4.2 APLICACIÓN DE LOS ALGORITMOS GENÉTICOS A LA OPTIMIZACIÓN DE CONTROLADORES PID

Para optimizar por medio de los algoritmos genéticos un controlador PID previamente diseñado se necesita, como para cualquier problema de búsqueda por este método, definir una función de aptitud que muestre o describa el

comportamiento del sistema de control compensado con el controlador PID. Esta función será función de las constantes del controlador PID, esto es: Kp, Ki y Kd.

Para poder evaluar en cada iteración la bondad de cada individuo (controlador PID compuesto por sus tres constantes), la función de aptitud debe ser capaz de dar una medida de los parámetros que deseen optimizarse, como por ejemplo el tiempo de establecimiento, el máximo sobretiro o el error de posición para que de esta manera el algoritmo genético encuentre el mínimo (o máximo) valor de la función de aptitud.

Por lo tanto lo que se hace en primera instancia es escribir la función fitness (F_T), como la indicada en la ecuación 4.1, con la que trabajará el algoritmo genético. En esta función es crucial reflejar la estabilidad relativa de la planta en lazo cerrado, pues si el controlador desestabiliza a la planta debe ser completamente rechazado, para que ese controlador no evolucione ni se reproduzca a lo largo de las generaciones.

Para comprobar la estabilidad del sistema, dadas las herramientas computacionales disponibles hoy en día, puede emplearse varios métodos, siendo el más sencillo el criterio del lugar geométrico de las raíces: si uno o más polos del sistema en lazo cerrado están en el semiplano derecho del plano complejo, entonces el sistema es inestable; este hecho debe reflejarse en la función fitness (constante M), como se muestra en la ecuación 4.1

$$F_T(kp, kd, ki) = F(kp, kd, ki) + M \qquad \text{Ecuación 4.1}$$

Además, la función F_T considera las características en estado estable y/o transitorio del sistema de control en lazo cerrado (función F). Para F(kp, kd, ki), debe escogerse en primer lugar el criterio de diseño, esto es; qué características se desea que el controlador PID satisfaga. El caso general será cuando deba cumplirse con un mínimo tiempo de establecimiento, máximo sobretiro y error de posición simultáneamente. Entonces se requiere que se evalúe en F estas

características, como se indica en la ecuación 4.2. Una vez más esto se simplifica enormemente con la ayuda de los paquetes computacionales.

$$F(kp, kd, ki) = w1 * mp + w2 * ts + w3 * ep \qquad \text{Ecuación 4.2}$$

Debe ponerse especial atención que los valores de las funciones: *mp, ts* y *ep* deben estar normalizadas para poder sumarlas y dar la misma prioridad o ponderación a las tres características. Se las normaliza dividiendo su respectivo valor para su valor base que puede ser:

- Para el tiempo de establecimiento, el tiempo total que el software emplee para graficar la respuesta temporal.
- Para el sobretiro y error de posición, el valor en estado estable para una entrada escalón unitario.

Una vez hecho esto se puede asignar valores a los pesos: *w1, w2* y *w3* según se desee dar mayor prioridad a una cierta característica. Por ejemplo si en algún caso interesa que el controlador estabilice al sistema en el menor tiempo posible sacrificando el sobretiro se puede dar valores a los pesos tales como: w1=0.5, w2=2; w3=1 (conservando la nomenclatura de la ecuación 4.1). Con esto se consigue que la aptitud de F dependa mayormente del valor de la característica que tiene mayor peso por lo que el algoritmo genético tratará de minimizar más fuertemente dicha característica.

Retomando el criterio de la estabilidad del sistema, como el algoritmo busca la minimización de la función F_T y se explicó que cualquier individuo inestable debe "morir", es lógico pensar que si el sistema es estable, M toma un valor nulo (cero), pero si es inestable, M adoptará un valor muy grande, ¿cuán grande?; como la función F está compuesta por valores normalizados (*mp, ts* y *ep*) ponderados con valores menores a 10 (dependiendo del criterio del diseñador), el valor de M cuando el sistema es inestable puede ser de cien (100) o mil (1000).

Tanto estos valores como los de los pesos (*w1, w2* y *w3*) se obtienen por medio del ensayo y error desarrollado en el Capítulo 5.

Se puede resumir todo lo explicado anteriormente por medio del algoritmo de diseño para optimizar un controlador PID indicado en la Sección 4.4.1.

Gracias al inmenso desarrollo en estos días de los paquetes computacionales especializados en la ciencia del control (MatLab y Mathematica por nombrar dos) y las interfaces de usuario (GUI), existen otras formas de optimizar controladores PID y otros tipos de controladores que son más fáciles e intuitivas, en donde el usuario no requiere de profundo conocimiento teórico (como el necesario para el desarrollo anterior) si no que tan sólo especifica gráficamente la respuesta que desea que el sistema de control tenga. Este método se explica en profundidad en la Sección 5.1.3

4.3 DISEÑO DE CONTROLADORES ÓPTIMOS ROBUSTOS

La robustez de estabilidad o robustez en rechazo a perturbaciones es un requerimiento deseado de un sistema de control, sin embargo; aun cuando la planta en lazo cerrado sea estable, no es de ninguna utilidad en la práctica si no tiene un desempeño o comportamiento adecuado, esto significa que el problema de diseño de un controlador robusto, para que sea eficiente involucra considerar la robustez de estabilidad o al rechazo a perturbaciones y también su comportamiento: en cuán bien la señal de salida sigue al set-point, el mínimo tiempo de estabilización, el máximo sobretiro, etc.

Es necesario por lo tanto proveer un mecanismo o una forma de evaluar el comportamiento de la planta para poder diseñar el controlador robusto y *convertirlo* en un controlador óptimo robusto, este mecanismo puede ser de dos tipos [24]:

1. Criterio basado en algunos puntos de la respuesta

2. Criterio basado en la respuesta completa, o criterio integral

1. Se considera parámetros del tipo tiempo de establecimiento o asentamiento, tiempo de subida, máximo sobretiro, error de seguimiento (posición), error de velocidad, etc. Este criterio es el que se emplea en la optimización del controlador PID de la Sección 4.2.

2. El criterio que considera una respuesta completa es el de los índices de comportamiento que se discuten a continuación.

4.3.1 ÍNDICES DE DESEMPEÑO O COMPORTAMIENTO

Un índice de comportamiento o desempeño puede definirse como sigue [9]: "es una medida cuantitativa del desempeño de un sistema y es escogido de tal manera que se de énfasis a las especificaciones más importantes del sistema de control."

Incluyendo este concepto dentro de lo que es un sistema óptimo se puede concluir que para que un sistema sea óptimo se requiere que sus parámetros (del controlador) minimicen algún índice de desempeño seleccionado.

Algunos de los índices de desempeño más usados son los que se basan en el error del sistema:

- ISE: integral del error al cuadrado (*integral square error* en inglés)
- ITSE: integral del tiempo multiplicado por el error al cuadrado (*integral time square error* en inglés)
- IAE: integral del valor absoluto del error (*integral absolute error* en inglés)
- ITAE: integral del tiempo multiplicado por el valor absoluto del error (*integral time absolute error* en inglés)

Las definiciones de estos cuatro índices se indican en las ecuaciones 4.3, 4.4, 4.5 y 4.6.

$$ISE = \int_0^{\infty} [e(t)]^2 \, dt \qquad \text{Ecuación 4.3}$$

$$ITSE = \int_0^{\infty} t * [e(t)]^2 \, dt \qquad \text{Ecuación 4.4}$$

$$IAE = \int_0^{\infty} |e(t)| \, dt \qquad \text{Ecuación 4.5}$$

$$ITAE = \int_0^{\infty} t * |e(t)| \, dt \qquad \text{Ecuación 4.6}$$

Nótese que para todos los casos, el valor dentro de la integral es siempre positivo. Como el error puede tomar magnitudes positivas y negativas es necesario darle una idea más realista de la verdadera influencia que éste tiene en la respuesta del sistema. Por esta razón es que se toma una medida que sea independiente del signo, elevando al cuadrado o tomando el valor absoluto del error.

Para ver una comparación gráfica entre los indices ISE, IAE e ITAE referirse a la figura 4.1.

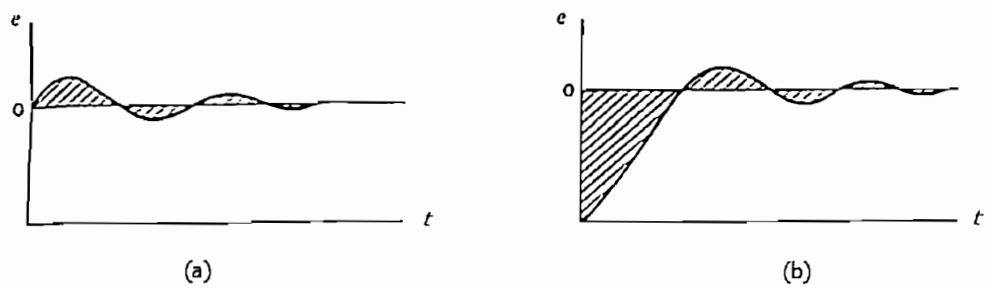


Figura 4.1 Señal de error para: (a) perturbaciones y (b) cambios de referencia [25]

En la figura 4.1 se presenta un gráfico de una señal típica de error para (a) cambios en la carga (perturbaciones) y (b) cambios en el set-point. En la figura 4.2 se aprecian las respuestas características de los índices ISE, IAE e ITAE.

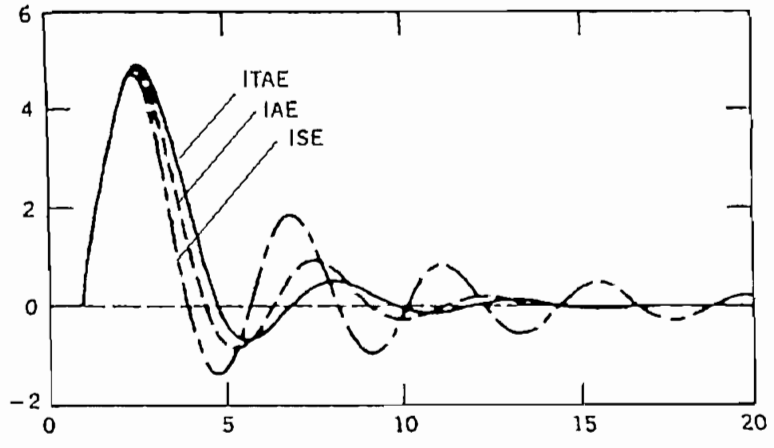


Figura 4.2 Características típicas del ITA, IAE e ISE [25]

En el presente trabajo se empleará la minimización del índice ISE debido a que puede ser adaptado fácilmente a mediciones prácticas ya que la implementación de un circuito cuadrático no presenta mayores complicaciones [9] y por "ser matemáticamente conveniente para fines analíticos y computacionales" [9].

A la estrategia de minimización del índice ISE se le conoce también como control lineal cuadrático óptimo (*linear cuadratic optimal control* en inglés).

Como se suele trabajar en el dominio de la frecuencia (s) se vuelve importante conocer la expresión correspondiente al índice ISE, la cual se indica en la ecuación 4.7

$$\mathrm{ISE} = -\frac{1}{\mathrm{j}2\pi}\int_{-\mathrm{j}\infty}^{+\mathrm{j}\infty} \mathrm{E}(\mathrm{s})\,\mathrm{E}(-\mathrm{s})\,\mathrm{ds} \qquad \text{Ecuación 4.7}$$

Siendo *E(s)* la transformada de Laplace del error *e(t)*.

4.3.2 DISEÑO DE CONTROLADORES ÓPTIMOS ROBUSTOS EN ESTABILIDAD

Teniendo en mente la idea de que el problema de diseño de un controlador robusto óptimo se interpreta como un problema de optimización del comportamiento del sistema (seguimiento) sujeto a la restricción de cumplir con robustez de estabilidad, se puede llegar al análisis que se detalla a continuación.

Considérese nuevamente el diagrama de bloques en el dominio de la frecuencia de la figura 4.3, en donde:

- R(s) es la señal de entrada al sistema o set-point
- E(s) es la señal del error que ingresa al controlador que va a ser diseñado
- C(s,k) es la función de transferencia del controlador
- U(s) señal que entra a la planta
- $G_0(s)$ es la planta nominal sin considerar incertidumbre
- Y(s) señal de salida que debe satisfacer las condiciones de diseño

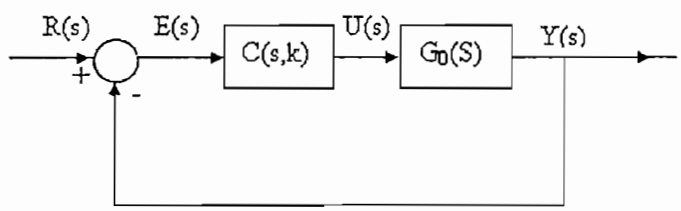


Figura 4.3 Sistema con planta con incertidumbre estructurada y controlador de estructura fija [13]

Para tener un controlador óptimo robusto en estabilidad, éste debe satisfacer las siguientes condiciones:

- Maximizar la función alfa (indicada en la ecuación 4.8) satisfaciendo la restricción de no ser mayor que uno, y
- Simultáneamente, minimizar el índice de comportamiento ISE (mostrada en la ecuación 4.7)

Ecuación 4.8

$$\alpha(\omega,k)=\frac{[C(j\omega,k)\,G_0(j\omega)W_m(j\omega)]\;\;[C(-j\omega,k)\,G_0(-j\omega)W_m(-j\omega)]}{[1+C(j\omega,k)\,G_0(j\omega)]\,[1+C(-j\omega,k)\,G_0(-j\omega)]}$$

Debe recordarse de la Sección 3.3.1 que la función W_m es una función que limita la incertidumbre en el modelo de la planta. La forma más fácil de hallar o calcular esta función es resolver el miembro izquierdo de la ecuación 3.7 y en base a esto proponer una función de transferencia cuya magnitud sea siempre mayor a dicha expresión, otra forma más teórica de hacerlo es por medio de mediciones de magnitud y fase para distintas frecuencias del sistema (este método puede consultarse en [10] y [12]).

Para calcular el ISE debe calcularse la función de transferencia del error E(s), que se indica en la ecuación 4.9 (a)

$$E(s,k)=\frac{R(s)}{1+C(s,k)\,G_0(s)} \qquad (a)$$

Ecuación 4.9

$$=\frac{D(s,k)}{A(s,k)}=\frac{\sum_{j=0}^{m} d_j s^{m-j}}{\sum_{i=0}^{n} a_i s^{n-i}} \qquad (b)$$

Que por facilidad, se expresa al error como una función racional polinomial como la mostrada en la ecuación 4.9 (b).

En la ecuación 4.9 (b) del error, se tomó como entrada un escalón unitario (R=1/s). Reemplazando este valor en la expresión del ISE se obtiene la ecuación 4.10; en el desarrollo matemático se suele emplear la letra J para designar el ISE con un subíndice que indica el orden del polinomio A(s). E(s) debe ser una fracción racional [12] para que el ISE tenga un valor finito.

Ecuación 4.10

$$J_n(k) = -\frac{1}{j2\pi}\int_{-j\infty}^{+j\infty}\left(\frac{\sum_{j=0}^{m} d_j s^{m-j}}{\sum_{i=0}^{n} a_i s^{n-i}} * \frac{\sum_{j=0}^{m} d_j (-s)^{m-j}}{\sum_{i=0}^{n} a_i (-s)^{n-i}}\right) ds$$

Por lo tanto, el diseño del controlador óptimo robusto en estabilidad se resume en la expresión de la ecuación 4.11. El algoritmo completo de diseño empleando los algoritmos genéticos se muestra en la Sección 4.4.2 y 4.4.3

Ecuación 4.11

$$\min_{k} \quad J_n(k) \qquad sujeto\ a \qquad \max_{\omega} \quad \sqrt{\alpha(\omega,k)} < 1$$

4.3.3 DISEÑO DE CONTROLADORES ÓPTIMOS ROBUSTOS EN RECHAZO A PERTURBACIONES

Considérese el diagrama de bloques en el dominio de la frecuencia de la figura 4.5, en donde:

- R(s) es la señal de entrada al sistema o set-point
- E(s) es la señal del error que ingresa al controlador que va a ser diseñado
- C(s,k) corresponde al controlador
- U(s) señal que entra a la planta
- $G_0(s)$ es la planta nominal
- Y(s) señal de salida que debe satisfacer las condiciones de diseño
- $D_y(s)$ es la perturbación externa a la salida de la planta

Para tener un controlador óptimo robusto en estabilidad, éste debe satisfacer las siguientes condiciones:

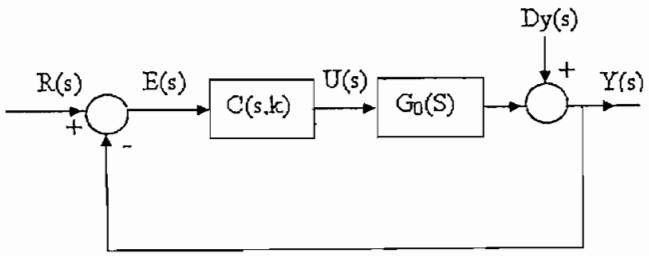


Figura 4.4 Sistema con perturbación a la salida de la planta [13]

- Maximizar la función beta (indicada en la ecuación 4.12) satisfaciendo la restricción de no ser mayor que el nivel deseado de atenuación ζ.
- Simultáneamente, minimizar el índice de comportamiento ISE (mostrada en la ecuación 4.7)

Ecuación 4.12

$$\beta(\omega,k)=\frac{W_d(j\omega)\quad W_d(-j\omega)}{[1+C(j\omega,k)\,G_0(j\omega)]\,[\,1+C(-j\omega,k)\,G_0(-j\omega)]}$$

En la ecuación 4.12, la función W_d es un filtro que brinde una atenuación apropiada para la perturbación $D_y(s)$ actuante en la planta. Por lo general las perturbaciones presentan una respuesta en frecuencia tal que a medias y altas frecuencias su efecto es considerable, por lo tanto es lógico pensar que la función $D_y(s)$ debe proveer supresión de frecuencias medias y altas, es decir un filtro pasabajos.

El valor del nivel de atenuación ζ necesariamente debe ser menor a uno para que el diseño se ajuste al requerimiento de rechazar perturbaciones, algunos autores ([10], [12], [13]) sugieren emplear un valor de 0.1 para obtener resultados adecuados.

Tanto la función de transferencia del error E(s) como la del índice ISE (Jn) son las mismas encontradas en la Sección 4.3.3 y que se indican en las ecuaciones 4.10 y 4.11 respectivamente.

Resumiendo entonces, el diseño del controlador óptimo robusto en rechazo a perturbaciones se reduce a la expresión de la ecuación 4.13. El algoritmo completo de diseño empleando los algoritmos genéticos se muestra en la Sección 4.4.2 y 4.4.3.

Ecuación 4.13

$$\min_{k} \quad J_n(k) \qquad sujeto\ a \qquad \max_{\omega} \quad \sqrt{\beta(\omega,k)} < \xi$$

4.4 ALGORITMOS DE DISEÑO

En forma general, la función $J_n(k)$ es una función altamente no lineal de (k) con muchos mínimos locales [13] y aunque todavía no exista un método completamente eficiente para obtener una solución global en el problema de minimización de las normas H, los algoritmos genéticos están siendo usados extensamente [8] para aproximarse bastante al mínimo global y resolver el problema.

En el Capítulo 2 se presentó el esquema básico de un algoritmo simple. Ahora, en base a ese esquema se desarrollan los algoritmos de diseño de los controladores robustos óptimos que son más complicados.

4.4.1 OPTIMIZACIÓN CONTROLADOR PID

A continuación se presenta un algoritmo de diseño de los algoritmos genéticos para emplearlos en la optimización de controladores PID. Esto no significa que sea el único algoritmo posible ni que sólo sirva para un controlador PID; se presenta un ejemplo empleando la estructura de un PID como se explicó en la Sección 4.2 al que es necesario remitirse para una completa comprensión de este algoritmo.

Paso a: Especificar el rango de las constantes kp, ki, kd del controlador C(s,k)

Paso b: Definir los parámetros del algoritmo genético: probabilidad de cruce, probabilidad de mutación, tamaño de la población, número de generaciones, método de selección, etc.

Paso 1: Se inicializa la población del algoritmo genético y se asigna en uno el número de generaciones (G_1) del algoritmo genético.

Paso 2: Para cada individuo k_i del algoritmo genético se encuentra la ubicación de los polos del sistema en lazo cerrado (analiza la estabilidad del sistema en lazo cerrado), si el sistema es estable M=0, si es inestable M=1000 (ver ecuación 4.14).

$$F_T(kp, kd, ki) = F(kp, kd, ki) + M \qquad \text{Ecuación 4.14}$$

Paso 3: Se calcula el tiempo de establecimiento (ts), máximo sobretiro (mp) y el error de posición (ep) y se los normaliza.

Paso 4: Se asignan valores a los pesos *w1, w2* y *w3* para hallar la función F (ver ecuación 4.15)

$$F(kp, kd, ki) = w1 * mp + w2 * ts + w3 * ep \qquad \text{Ecuación 4.15}$$

Paso 5: Se evalúa la aptitud de cada individuo por medio de la función fitness F_T (ver ecuación 4.14).

Paso 6: Se escogen individuos que deben reproducirse por medio del método de selección escogido, se los cruza y muta según la probabilidad seleccionada.

Paso 7: Si se alcanzó el número máximo de generaciones establecidas se detiene el algoritmo (se ha encontrado ya el mínimo), caso contrario se aumenta el número de la generación (G_1) en uno y se regresa al Paso 2.

4.4.2 CONTROLADOR ÓPTIMO ROBUSTO MINIMIZANDO EL ISE

En el diseño de un controlador óptimo robusto, ya sea en rechazo a perturbaciones o en estabilidad se indicó que es necesario optimizar dos funciones simultáneamente: minimizar el índice ISE (Jn) mientras a su vez se maximiza la función α (para el caso de robustez de estabilidad) y β (para rechazo a perturbaciones) con ciertas restricciones; referirse a la Sección 4.3.

Este problema se torna de mucho interés pues es necesario emplear dos algoritmos genéticos que trabajen en forma conjunta: el algoritmo genético 1 (AG_1) se usará para minimizar J_n y el algoritmo genético 2 (AG_2) para maximizar α ó β.

El algoritmo básico para diseñar controladores óptimos robustos es el mismo para cuando se busca robustez de estabilidad y robustez para rechazo a perturbaciones, por esta razón se detallará el algoritmo general para los dos casos pero haciendo notar las partes que difieren el uno del otro.

El procedimiento consiste primeramente en inicialización y configuración de parámetros y estructuras (Pasos del *a* al *c*) y el mecanismo de trabajo de los dos algoritmos en forma anidada (Pasos del *1* al *5*)

Paso a: definir la estructura del controlador *C(s,k)* (ver figuras 4.3 y 4.4 según cada caso) y de W_m (para el caso de robustez de estabilidad) o W_d (para el caso de robustez al rechazo a perturbaciones), calcular la función de transferencia del error *E(s)* (ver ecuación 4.9) y calcular la función restrictiva de robustez, α para estabilidad o β para rechazo a perturbaciones (ver ecuaciones 4.8 y 4.12 respectivamente).

Paso b: Especificar el rango de las constantes (vector *k*) del controlador C(s,k)

Paso c: Definir los parámetros de los dos algoritmos genéticos como probabilidad de cruce y mutación, tamaño de la población, número de generaciones, método de selección, etc.

Paso 1: Se inicializa la población de AG_1, es decir se escoge un primer vector k_i siendo i=1,2,...., n_1 (n_1 es el tamaño de la población). Se inicializa la población de AG_2: ω=1,2,..., n_2. Se setea en uno el número de generaciones (G_1) de AG_1.

Paso 2: Para cada individuo k_i de AG_1 se calcula, por medio de AG_2 el máximo valor de: $[\ \alpha\ (\omega,k_i)\ ]^{1/2}$ o de $[\ \beta\ (\omega,k_i)\ ]^{1/2}$ según sea el caso, de la siguiente forma:

- Paso 2.1: Se inicializa cada individuo, formado por un solo gen: ω_j siendo j=1,2,..., n_2 y se setea en uno el número de generaciones (G_2) de AG_2.
- Paso 2.2: Se evalúa a cada individuo usando la función de aptitud: $\alpha\ (\omega_j,k_i)$ o $\beta\ (\omega_j,k_i)$ según corresponda.
- Paso 2.3: Se selecciona individuos de acuerdo al método de selección escogido y se aplica los operadores genéticos seleccionados en un inicio.
- Paso 2.4: Si se alcanzó el número máximo de generaciones escogido al inicio, se pasa el valor de la aptitud del mejor individuo, es decir el máximo de: $\alpha\ (\omega_j,k_i)$ o $\beta\ (\omega_j,k_i)$ al AG_1 y sigue al Paso 3; caso contrario se incrementa en uno número de la generación (G_2) y se regresa al Paso 2.2

Paso 3: Para cada individuo k_i de AG_1 se calcula la función de aptitud F_1, ver ecuación 4.16

Paso 4: Se seleccionan individuos de AG_1 por medio del método de selección escogido, se los cruza y muta según la probabilidad seleccionada.

Paso 5: Si se alcanzó el número máximo de generaciones de AG_1 establecidas en el Paso c se detiene el algoritmo (se ha encontrado ya el mínimo), caso contrario se aumenta el número de la generación (G_1) en uno y se regresa al Paso 2.

La función de aptitud F_1 (ver ecuación 4.16) será, como es lógico pensar, el índice ISE, junto con alguna función que permita castigar o penalizar a individuos que inestabilicen al sistema o que no satisfagan la condición de restricción de α o β.

$$F_1(k_i) = J_n(k_i) + P_1(k_i) \qquad \text{Ecuación 4.16}$$

La función P_1 permite penalizar a aquellos individuos que no satisfacen con los requerimientos deseados por medio de la adición de una constante, para que a través de las generaciones no sobrevivan, de la siguiente manera:

Si un individuo es inestable, esto es; si no satisface la prueba de estabilidad aplicada a la ecuación característica del sistema (existencia de polos del sistema en lazo cerrado en el semiplano derecho), recibe una penalización muy elevada por medio de una constante muy grande: $P_1(k_i) = M$; si en cambio el individuo satisface la prueba de estabilidad pero no cumple con la condición de robustez de estabilidad (o sea se tiene que para ese individuo $\mathbf{[\alpha(\omega,k_i)]^{1/2} > 1}$ o $\mathbf{[\beta(\omega,k_i)]^{1/2} > \zeta}$) se le amonesta con una constante (aunque no tan severa como M) $P_1(k_i) = m$.

Finalmente, si el individuo es una respuesta factible y cumple con todas las condiciones no se le penaliza: $P_1(k_i) = 0$.

Es necesario recalcar que los Pasos del *1* al *5* los realiza el software que se emplee en la resolución del problema escribiendo los comandos adecuados como se muestra en el Capítulo 5, mientras que los pasos del *a* al *c* deben ser realizados por el diseñador.

4.4.3 CONTROLADOR ÓPTIMO ROBUSTO MINIMIZANDO CARACTERÍSTICAS TEMPORALES

Como un controlador óptimo robusto busca satisfacer condiciones de robustez y además cumplir con especificaciones deseables, puede optarse por otro método alternativo al anterior mostrado en la Sección 4.4.2.

El método consiste en minimizar la respuesta temporal del sistema en lugar de minimizar el ISE, como se mencionó brevemente en la Sección 4.3. Este método puede verse como un híbrido entre el diseño de la optimización del controlador PID de la Sección 4.4.1 y del controlador robusto óptimo de la Sección 4.4.2.

Este problema se resuelve nuevamente empleando dos algoritmos genéticos: AG_1 minimiza las características temporales de la señal de salida mientras que AG_2 se encarga de maximizar las funciones α (para el caso de robustez de estabilidad) y β (para rechazo a perturbaciones) con ciertas restricciones; el algoritmo de diseño es el que se indica a continuación:

Paso a: definir la estructura del controlador *C(s,k)* (ver figuras 4.3 y 4.4 según cada caso) y de W_m (para el caso de robustez de estabilidad) o W_d (para el caso de robustez al rechazo a perturbaciones) y calcular la función restrictiva de robustez, α para estabilidad o β para rechazo a perturbaciones (ver ecuaciones 4.8 y 4.12 respectivamente).

Paso b: Especificar el rango de las constantes (vector *k*) del controlador C(s,k)

Paso c: Definir los parámetros de los dos algoritmos genéticos como probabilidad de cruce y mutación, tamaño de la población, número de generaciones, método de selección, etc.

Paso 1: Se inicializa la población de AG_1, es decir se escoge un primer vector k_i siendo i=1,2,…., n_1 (n_1 es el tamaño de la población). Se inicializa la población de AG_2: ω=1,2,…, n_2. Se asigna en uno el número de generaciones (G_1) de AG_1.

Paso 2: Para cada individuo k_i de AG_1 se calcula, por medio de AG_2 el máximo valor de: $[\,\alpha(\omega,k_i)\,]^{1/2}$ o de $[\,\beta(\omega,k_i)\,]^{1/2}$ según sea el caso, de la siguiente forma:

- Paso 2.1: Se inicializa cada individuo, formado por un solo gen: ω_j siendo j=1,2,…, n_2 y se setea en uno el número de generaciones (G_2) de AG_2.

- Paso 2.2: Se evalúa a cada individuo usando la función de aptitud: $\alpha(\omega_j, k_i)$ o $\beta(\omega_j, k_i)$ según corresponda.
- Paso 2.3: Se selecciona individuos de acuerdo al método de selección escogido y se aplica los operadores genéticos seleccionados en un inicio.
- Paso 2.4: Si se alcanzó el número máximo de generaciones escogido al inicio, se pasa el valor de la aptitud del mejor individuo, es decir el máximo de: $\alpha(\omega_j, k_i)$ o $\beta(\omega_j, k_i)$ al AG_1 y sigue al Paso 3; caso contrario se incrementa en uno número de la generación (G_2) y se regresa al Paso 2.2

Paso 3: Para cada individuo k_i de AG_1 se calcula la función de aptitud F_T, ver ecuación 4.17

Paso 4: Se seleccionan individuos de AG_1 por medio del método de selección escogido, se los cruza y muta según la probabilidad seleccionada.

Paso 5: Si se alcanzó el número máximo de generaciones de AG_1 establecidas en el Paso c se detiene el algoritmo (se ha encontrado ya el mínimo), caso contrario se aumenta el número de la generación (G_1) en uno y se regresa al Paso 2.

$$F_T(k) = F(k) + M + N \qquad \text{Ecuación 4.17}$$

$$F(k) = w1 * mp + w2 * ts + w3 * ep \qquad \text{Ecuación 4.18}$$

La función fitness F_T análogamente a lo visto con anterioridad, debe reflejar la estabilidad relativa, además la restricción de no sobrepasar el valor de 1 y el valor en sí de la función alfa.

Para comprobar la estabilidad del sistema se verifica que ningún polo o polos del sistema en lazo cerrado estén en el semiplano derecho del plano complejo. Si el sistema es inestable, M toma un valor muy grande (1000000 por ejemplo) [12] pero si es estable, cero.

La restricción de que el valor de la raíz cuadrada de alfa o beta (según la robustez que se busque con el diseño) no sobrepase los valores restricciones (de 1 para alfa y de un valor ζ para beta) se refleja en el valor de N: si las funciones alfa o beta resultan sobrepasar las restricciones N toma un valor de cien [12], caso contrario el valor de 0.

La función F (de la ecuación 4.18) se forma de la siguiente manera: se calcula el tiempo de establecimiento (ts), máximo sobretiro (mp) y el error de posición (ep) y se los normaliza. Se asignan valores a los pesos *w1*, *w2* y *w3* y se evalúa la expresión mostrada en la ecuación 4.18.

CAPÍTULO 5

Resultados

5.1 SELECCIÓN DEL SOFTWARE

Para el presente proyecto de titulación se consideró conveniente el empleo del paquete computacional de MatLab versión 7 para las simulaciones y desarrollos matemáticos, por las siguientes razones:

- El paquete es extensamente utilizado y estudiado en la carrera de formación académica, por lo que presenta bastante familiaridad con el estudiante y profesorado.
- Disponibilidad de interfaz gráfico para el usuario (GUI) para facilidad de manejo.
- Disponibilidad de un paquete de herramientas (*toolbox*) especializado en sistemas de control, algoritmos genéticos y optimización.

5.1.1 MATLAB

MatLab es un software matemático desarrollado por: "*The MathWorks*" que brinda muchas herramientas para el desarrollo de sistemas de control (aparte de muchas otras ramas); tanto a través del espacio de trabajo (*workspace*), como por los GUI o por medio de *Simulink* (subprograma de MatLab ideal para simulación).

En la sección de Anexos se presentan los archivos escritos para este software, sin embargo es interesante explicar y mostrar brevemente los principales comandos y opciones que se emplean en el desarrollo de este proyecto.

5.1.1.1 Comandos para Sistemas de Control

Los comandos que se utilizan para trabajar con sistemas de control son:

- tf([bn....b1 b0],[an....a1 a0]): Crea un sistema de función de transferencia de la forma: G(s)=(b0 +b1s + b2s^2+....)/(a0 +a1s + a2s^2+....)

- step(sistema): Grafica la respuesta temporal de "sistema" a una entrada paso.

- [var1 var2 var3]=damp(sistema): Asigna a "var1" los valores de las frecuencias naturales de los polos de "sistema", a , "var2" los valores de los índices de amortiguamiento de los polos del "sistema" y en "var3" los polos de "sistema" .

- [var1 var2]=step(sistema): Asigna a "var1" y "var2" los valores de la amplitud y el tiempo respectivamente, encontrados para graficar la respuesta temporal de "sistema" a una entrada paso.

- feedback(sistema,num): Realimenta negativamente a "sistema" con un valor de "num".

- rlocus(sistema): Grafica el lugar geométrico de las raíces de "sistema".

- bode(sistema): Grafica la respuesta en frecuencia (diagramas de Bode) de "sistema".

5.1.1.2 Comandos para Algoritmos Genéticos [16]

A pesar de que en este trabajo se trabaja básicamente con el GUI de los algoritmos genéticos, en algunas partes se utiliza comandos específicos como los presentados a continuación:

- [var1 var2]=ga(@funcion,n,options): Busca el mínimo valor de "funcion" de "n" variables por medio de algoritmos genéticos con opciones "options". Los valores de las variables a las que se tiene el mínimo se almacena en "var1" y el mínimo valor hallado de "funcion" en "var2".

- options = gaoptimset('PopulationSize',n): Define el tamaño de la población de individuos al valor de "n".

- options = gaoptimset('Generations',n): Define el número máximo de generaciones a desarrollarse al valor de "n".

- options = gaoptimset('StallGenLimit',t): Define el tiempo máximo permitido para finalizar la optimización (en segundos).

- options = gaoptimset('StallTimeLimit',t): Define el tiempo de espera máximo permitido para finalizar la optimización si no mejora el valor de la función (en segundos)

- options = gaoptimset('PopInitRange',[A]): Define a la matriz "A" como el rango inicial de los individuos, en la primera fila están los valores mínimos y en la segunda fila los valores máximos.

5.1.2 TOOLBOX DE ALGORITMOS GENÉTICOS

En el Capítulo 2 se hizo un breve acercamiento al GUI del *toolbox* pero ahora se profundizará en detalle cómo se plantea un problema, cuáles opciones se pueden modificar, cómo hacerlo y comprender los aspectos básicos de cómo el programa resuelve los problemas.

Para ingresar al GUI se digita en la línea de comando (*prompt*) la palabra "*gatool*". Esto abre una ventana que es en donde se harán todas las configuraciones y modificaciones pertinentes. Un bosquejo de esta ventana se

presentó en el Capítulo 2 con la figura 2.5. En las siguientes secciones se hará un análisis detallado de las distintas partes del mismo.

El toolbox de Algoritmos Genéticos trabaja con una función de aptitud (la función a minimizar) definida por el usuario la cual debe guardarse como un archivo *.m* del tipo "*function*" como el indicado en la figura 5.1

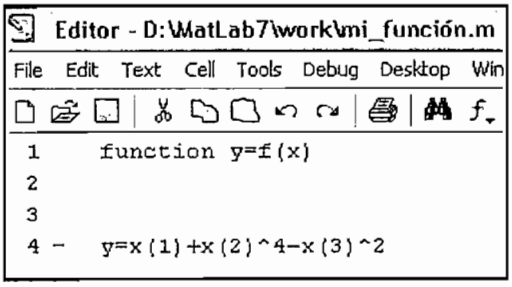


Figura 5.1 Función que depende de 3 variables nombrada "mi_función.m"

5.1.2.1 Planteamiento de problemas y opciones [16]

Para que MatLab pueda resolver un problema de optimización a través del GUI es necesario ingresar el nombre de la función de aptitud, ver figura 5.2 (1), precedido por el símbolo de arroba (@) y el número de variables de dicha función, ver figura 5.2 (2).

Se puede solicitar que el programa muestre una serie de gráficos, ver figura 5.2 (3), de la resolución del problema para ver cómo va avanzando la optimización de la función a través del tiempo o generaciones.

Una vez configuradas todas las opciones (que se discutirán un poco más adelante) se puede iniciar, ver figura 5.3 (4), el proceso de optimización. El GUI muestra los avances y otros detalles, ver figura 5.3 (5), mientras está corriendo el programa. Al finalizar, se despliega, ver figura 5.3 (6), los valores de los individuos óptimos hallados.

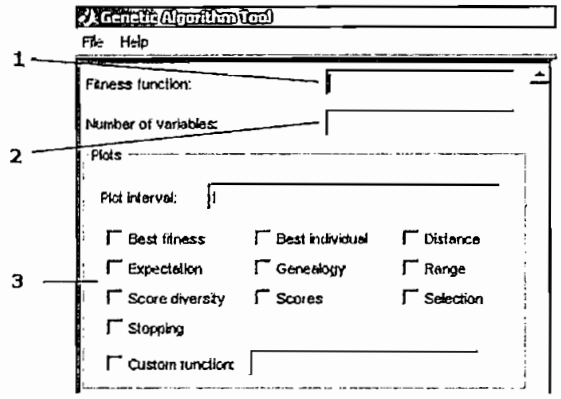


Figura 5.2 Interfaz gráfica del toolbox (parte 1) [16]

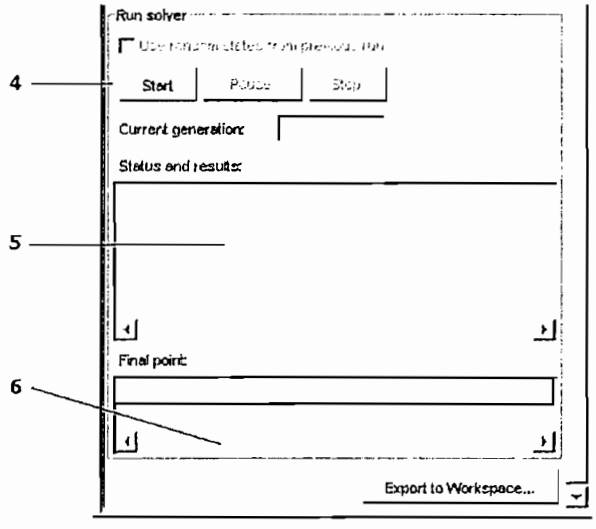


Figura 5.3 Interfaz gráfica del toolbox (parte 2) [16]

Respecto a las opciones que ofrece este toolbox (ver figura 5.4), se puede especificar, entre otras, el tamaño de la población (8). Si se desea se puede especificar los individuos con los cuales se deberá iniciar la optimización (9); o se puede especificar el rango de valores iniciales del cual se escogerá a los individuos (10).

Puede especificarse el método de selección de las poblaciones (11) entre las cuales se encuentran las mencionadas en el Capítulo 2; así como también la forma de realizar mutaciones (12) y cruces (13).

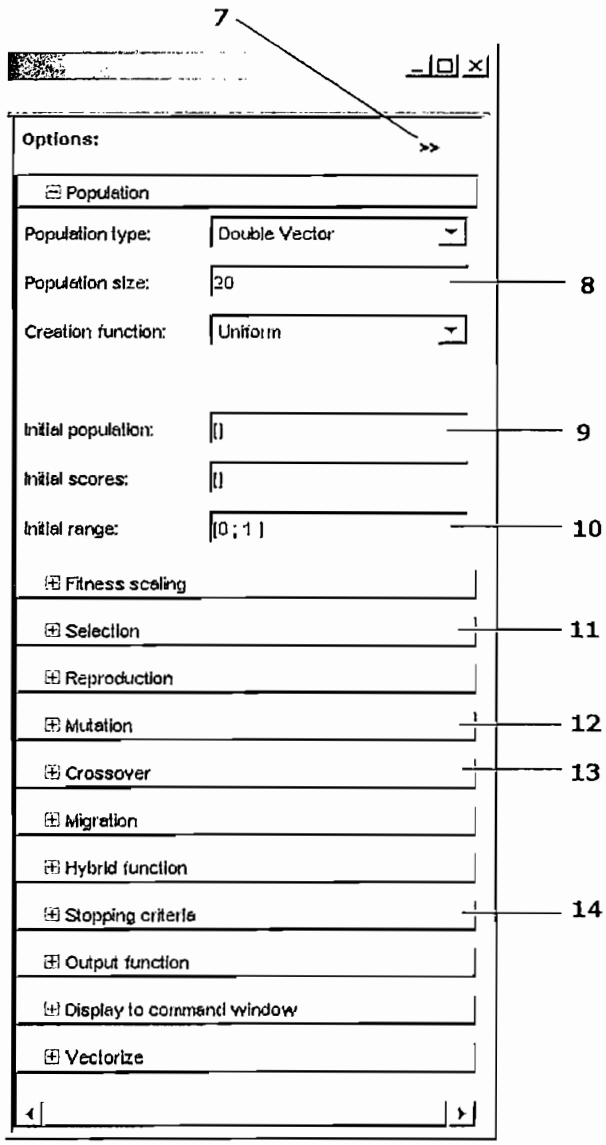


Figura 5.4 Interfaz gráfica del toolbox (parte 3) [16]

Finalmente, se puede especificar criterios de finalización de la optimización (14), configurando: tiempo y número máximo de generaciones a desarrollar, tiempo y número de generaciones a esperar si no se encuentra mejoría. Se puede acceder a breves explicaciones de cada opción (7).

5.1.2.2 Funcionamiento [16]

El algoritmo que emplea Matlab realiza básicamente los mismos pasos que se describieron en el Capítulo 1 y 2.

El algoritmo empieza creando una población inicial al azar, salvo que el usuario especifique una. Luego, el algoritmo crea una secuencia de nuevas poblaciones o generaciones; en cada etapa el algoritmo emplea los individuos de la generación actual para crear la próxima. El algoritmo crea las nuevas generaciones de la siguiente manera:

- Califica a cada miembro de la población evaluando su aptitud.
- Selecciona a los padres basándose en su aptitud.
- Crea hijos a partir de los padres por medio de cruce, mutación o ambos.
- Reemplaza la generación actual con los hijos para formar la nueva generación.

El algoritmo se detiene cuando se satisface alguna de las condiciones de finalización del algoritmo.

Es importante mencionar que MatLab busca el mínimo de la función de aptitud, por ejemplo si se tiene una función cualquiera: f(x1,x2,...) y si se desea hallar el máximo en lugar del mínimo, se debe cambiar la función a: – f(x1,x2,...) puesto que él o los puntos en los que se tiene el mínimo en -f(x) son los mismos puntos en los que f(x) es máximo [16].

5.1.3 OPTIMIZACIÓN DE RESPUESTAS CON SIMULINK

Dentro del Simulink de Matlab se encuentra un GUI llamado "Simulink Response Optimization" el cual provee una herramienta para ayudar al usuario a diseñar sistemas de control. Este programa permite calibrar cualquier parámetro dentro de un modelo de Simulink para satisfacer requerimientos o restricciones de comportamiento en el dominio del tiempo. Esto se lo hace gráficamente, de una manera fácil e intuitiva, imponiendo restricciones a la respuesta del modelo, estableciendo valores mínimos y máximos de los valores de: tiempo de subida, tiempo de establecimiento, máximo sobreimpulso y error de posición.

Para usar este GUI se necesita incluir en el diagrama de Simulink un bloque llamado "Signal constraint" (ver figura 5.5)

La optimización de respuesta de Simulink convierte las restricciones del dominio del tiempo, planteadas por el usuario, en un simple problema de optimización para resolverlo ya sea con el toolbox de Optimización o con el ya explicado toolbox de algoritmos genéticos.

Si no se consigue satisfacer las especificaciones, el software lo dice y solicita que se "suavice" las condiciones planteadas por el usuario.

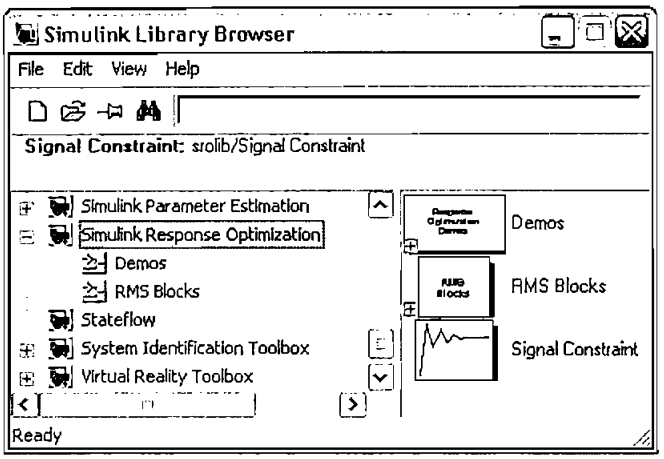


Figura 5.5 Bloque de optimización de respuesta temporal de Simulink

5.2 OPTIMIZACIÓN DE CONTROLADORES PID CON MATLAB

Es posible encontrar valores óptimos de las constantes kp, ki y kd de un controlador PID empleando MatLab. Como se acabó de explicar, dos formas de hacerlo es mediante el toolbox de algoritmos genéticos y mediante el bloque de "response optimization" del Simulink.

5.2.1 OPTIMIZACIÓN DE CONTROLADORES PID CON SIMULINK

Para poder optimizar un controlador PID, como se explicó previamente, basta con construir un modelo en Simulink como el mostrado en la figura 5.6.

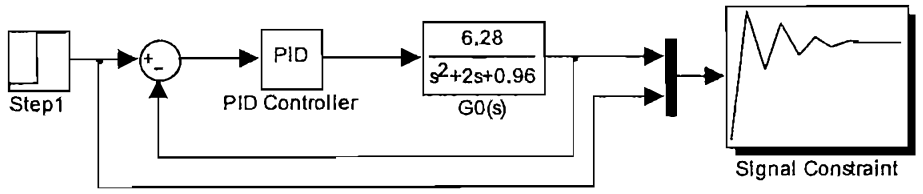


Figura 5.6 Diagrama en Simulink para optimización de respuesta temporal

En el bloque de PID se debe asignar nombres (en lugar de valores específicos) a las constantes del controlador pues estos valores es lo que se buscará optimizar.

Para que el programa pueda resolver el problema de optimización es necesario hacer previamente dos pasos fundamentales:

1. Crear en el espacio de trabajo las variables con las que se nombró a las constantes del controlador PID. Se les puede asignar valores arbitrarios.

2. Haciendo doble clic en el bloque "Signal constraint", accediendo al menú "Optimization", se escoge a través de la opción "Tuned parameters" las variables que deben ser optimizados.

Debe mencionarse que esta optimización presenta muchas opciones las cuales deben escogerse según el criterio del diseñador.

En este trabajo se empleará tres plantas para diseñarles controladores (13 en total). Las plantas son: Planta 1 definida por la ecuación 5.1 (página 73), Planta 2 definida por la ecuación 5.2 (página 77) y la Planta 3 por la ecuación 5.10 (página 96). Para cada planta se diseñará distintos tipos de controladores, así es como se presenta la siguiente notación, controladores diseñados para la *Planta 1*:

- Controlador 1
- Controlador 2
- Controlador 3

- Controlador 8
- Controlador 12
- Controlador 13

Los controladores diseñados para la *Planta 2* son los siguientes:

- Controlador 4
- Controlador 5
- Controlador 6
- Controlador 7

Y los controladores diseñados para la *Planta 3* son:

- Controlador 9
- Controlador 10
- Controlador 11

Empleando la optimización con Simulink, se procede a optimizar un controlador PID para la Planta 1, definida por la función de transferencia expresada en la ecuación 5.1.

$$G(s) = \frac{6.28}{s^2 + 2s + 0.96} \quad ; \quad H(s) = 1 \qquad \text{Ecuación 5.1}$$

Interesa observar la respuesta temporal de esta planta para compararla con las respuestas del sistema ya compensado. La respuesta temporal a una entrada paso de la planta es el indicado en la figura 5.7

Se diseñó un controlador PID por un método tradicional (Controlador 1) utilizando el lugar geométrico de las raíces seleccionando un punto de diseño y colocando ceros; y luego empleando el Simulink (Controlador 2).

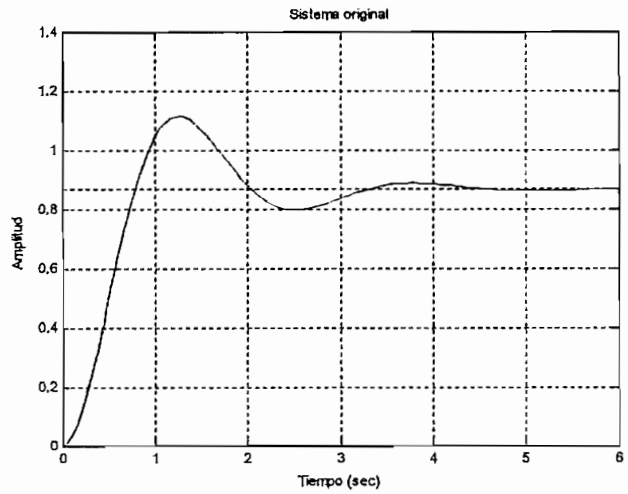


Figura 5.7 Respuesta temporal de la Planta 1 en lazo cerrado

Las constantes del controlador, diseñado por el método tradicional y por medio de la optimización de la respuesta temporal del sistema en Simulink se muestran en la tabla 5.1. El PID tradicional fue diseñado para cumplir con especificaciones de: error de posición cero, tiempo de establecimiento de 1 segundo y máximo sobretiro de 10%.

Las restricciones elegidas para la optimización con Simulink se muestran en la figura 5.8. Las respuestas temporales al primer y segundo controlador se muestran en las figuras 5.9 y 5.10 respectivamente.

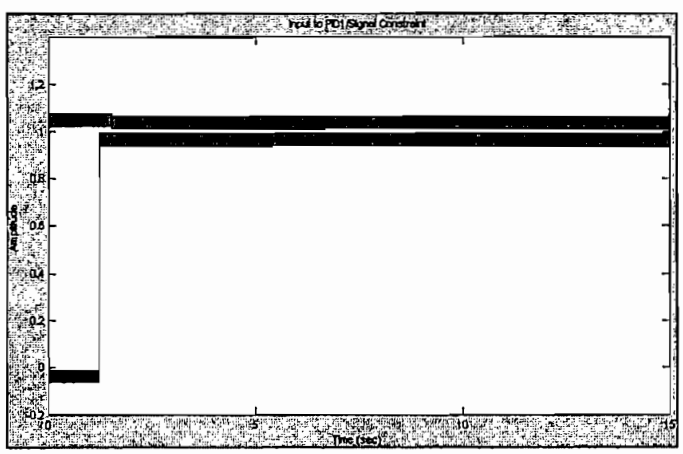


Figura 5.8 Especificaciones deseadas para el Controlador 2

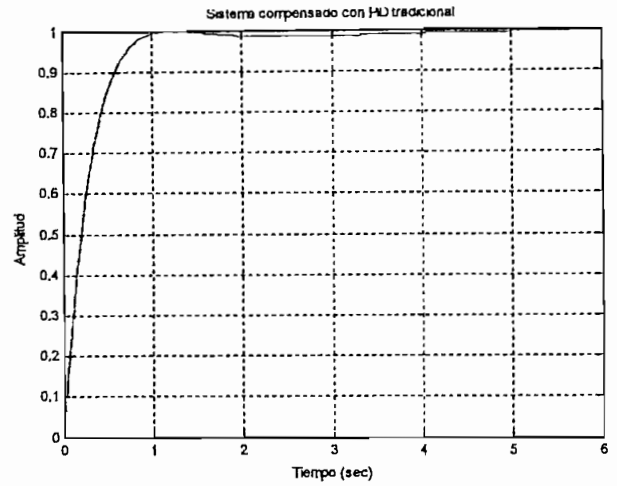


Figura 5.9 Respuesta temporal de la Planta 1 compensada con el Controlador 1

Las características de la Planta 1 sin compensación y compensada con los Controladores 1 y 2 se indican en la tabla 5.2. Aunque se especificó ciertos límites para la respuesta deseada con el Controlador 2, el software no pudo hallar valores que los satisfagan, los encontrados son los que proveen la respuesta más parecida. Los valores de estas constantes se muestran en la tabla 5.1.

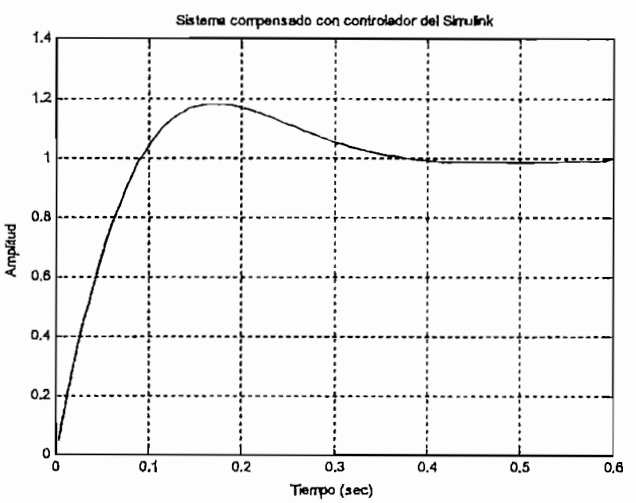


Figura 5.10 Respuesta temporal de la Planta 1 compensada con el Controlador 2

5.2.2 OPTIMIZACIÓN DE CONTROLADORES PID CON TOOLBOX DE ALGORITMOS GENÉTICOS

Para optimizar un controlador PID con el *toolbox* de Algoritmos genéticos, se escribió una función fitness de tres variables pues lo que se busca es optimizar las tres constantes del PID. En esta función se realiza los siguientes pasos:

1. Se computa la respuesta temporal del Sistema 1 compensado en lazo cerrado.
2. Se analiza la estabilidad del sistema en lazo cerrado.
3. Se calcula el tiempo de establecimiento, máximo sobretiro y el error de posición, expresándolos en por unidad (normalizar) tomando como base al tiempo total que escoge MatLab para dibujar la respuesta y el valor en estado estable de 1, (se realiza esta normalización para dar la misma influencia o peso a los tres parámetros por igual).
4. Se suma los tres parámetros asignándoles pesos según se desee dar mayor importancia a uno o más parámetros.

De esta manera el algoritmo al buscar el mínimo de la función, encontrará los valores de las constantes que minimicen el tiempo de establecimiento, máximo sobretiro y error de posición. Escribiendo la función de aptitud adecuada (ver Anexo 2) se empleó el toolbox de algoritmos genéticos para hallar las constantes del Controlador 3, que se detallan en la tabla 5.1; la respuesta del sistema con el Controlador 3 se indica en la figura 5.11.

	Controlador 1	Controlador 2	Controlador 3
Kp	1.31	31.62	33.63
Kd	0.53	2.63	21.7
Ki	0.49	0.0001	28.66

Tabla 5.1 Constantes del Controlador 1, 2 y 3 para la Planta 1

Las características de la Planta 1 sin compensación y compensada con los Controladores 1, 2 y 3 se indican en la tabla 5.2

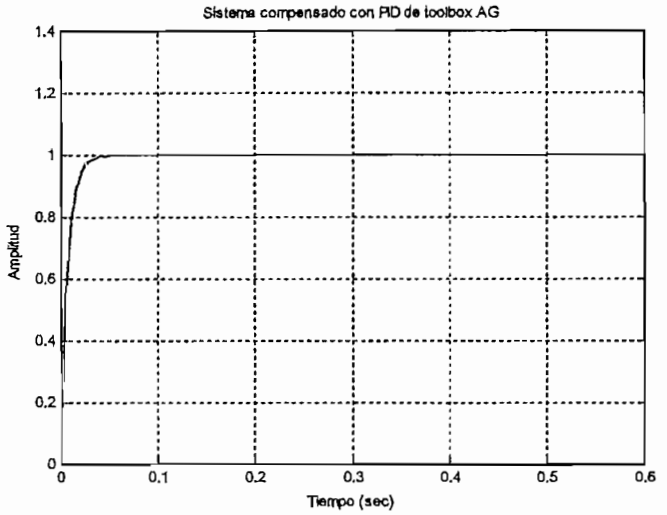


Figura 5.11 Respuesta temporal de la Planta 1 compensada con el Controlador 3

	Planta 1 original	Planta 1 con Controlador 1	Planta 1 con Controlador 2	Planta 1 con Controlador 3
Mp	28.4 %	0 %	18.2 %	0.09 %
ts	3.98 s	0.858 s	0.34 s	0.03 s
Ep	13.3 %	0 %	0 %	0 %

Tabla 5.2 Características temporales del Sistema 1 y Controladores 1, 2 y 3

Empleando este mismo método se va a hallar un controlador PID para la Planta 2, descrita por la función de transferencia mostrada en la ecuación 5.2

$$G(s) = \frac{1.8}{s^2(s+2)} \quad ; \quad H(s) = 1 \qquad \text{Ecuación 5.2}$$

La respuesta temporal de la Planta 2 a una entrada paso es el mostrado en la figura 5.12.

Se diseñó así mismo un controlador PD por métodos tradicionales (Controlador 4) con especificaciones de error cero, máximo sobretiro menor al 30% y tiempo de establecimiento menor que 8 segundos y con optimización por medio del toolbox de algoritmos genéticos (Controlador 5).

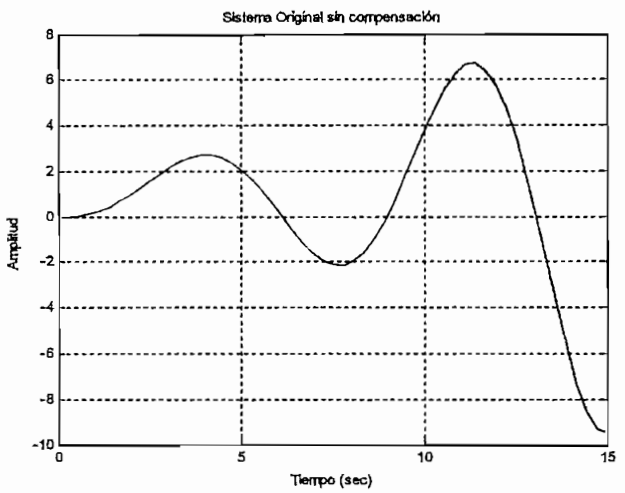


Figura 5.12 Respuesta temporal de la Planta 2 en lazo cerrado

Las constantes de los Controladores 4 y 5 se muestran en la tabla 5.3 y las respuestas temporales del Sistema 2 compensado en las figuras 5.13 y 5.14.

	Controlador 4	Controlador 5
Kp	0.47	0.01
Kd	2.35	0.88

Tabla 5.3 Constantes de los Controladores 4 y 5 para la Planta 2

La función de aptitud escrita para el Controlador 5 se basa en los mismos principios en que se escribió la función de aptitud del Controlador 3, lo único que se varía es la función de transferencia de la planta original.

Las opciones escogidas dentro del GUI para la optimización fueron las mostradas en la tabla 5.4.

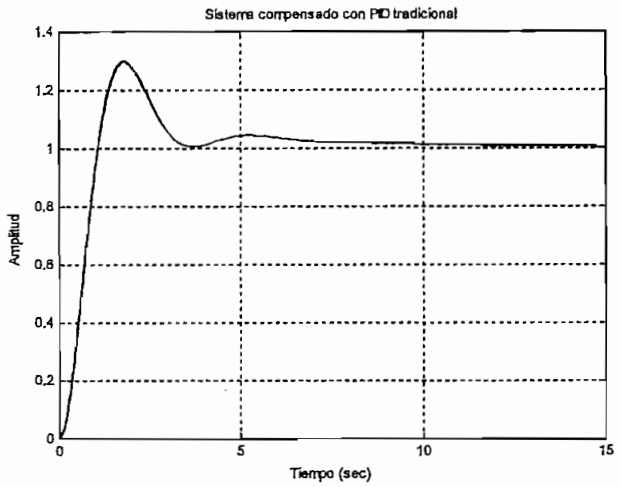


Figura 5.13 Respuesta de la Planta 2 compensada con el Controlador 4

Opción	Valor
Tamaño de la población	100
Número máximo de generaciones	100
Método de selección	Rueda de ruleta
Factor de mutación	0.01
Tipo de cruce	En un punto
Factor de cruce	0.6

Tabla 5.4 Opciones para la optimización del Controlador 5

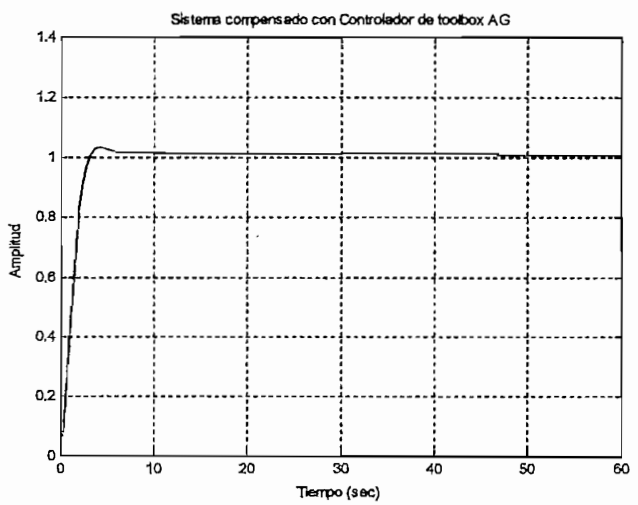


Figura 5.14 Respuesta de la Planta 2 compensada con el Controlador 5

Las características de la Planta 2 sin compensación y compensada con los Controladores 4 y 5 se indican en la tabla 5.5

	Planta 2	Planta 2 con Controlador 4	Planta 2 con Controlador 5
Mp	-	29.8 %	3.06 %
ts	-	7.48 s	5.65 s
Ep	-	0 %	0 %

Tabla 5.5 Características temporales del Sistema 2 y Controladores 4 y 5

5.3 DISEÑO DE CONTROLADORES ROBUSTOS CON MATLAB

Como se explicó en el Capítulo 3, un sistema de control puede ser robusto en el sentido de estabilidad, en el sentido de rechazo a perturbaciones o robustos óptimos en cualquiera de estos dos tipos. En esta sección se propone diseñar controladores, por medio del *toolbox* de algoritmos genéticos, que sean:

- Robustos en estabilidad
- Robustos en rechazo a perturbaciones
- Robustos óptimos en estabilidad
- Robustos óptimos en rechazo a perturbaciones

5.3.1 DISEÑO DE CONTROLADORES ROBUSTOS EN ESTABILIDAD

Para el diseño del controlador robusto en el sentido de la estabilidad se emplea la optimización (maximización) de la raíz cuadrada de la función alfa: $\alpha(w,k)$ con la restricción de que no sobrepase el valor de 1, (referirse al Capítulo 4); por lo tanto alfa será la función *fitness* a optimizar.

Recuérdese además del Capítulo 3, que es necesario proveer una incertidumbre (multiplicativa o aditiva) para poder realizar el diseño; lo que interesa es tener una incertidumbre acotada que cumpla con los requisitos que se

analizaron en la Sección 3.3.1. La manera más simple de obtener esta función es considerar la dinámica no modelada del sistema en cuestión, y calcular (con la ecuación 3.7) el valor de la incertidumbre acotada: ΔW_m, ver ecuación 5.3. Debe mencionarse que varios autores optan por simplemente escribir Δ agrupando en una sola variable el concepto de incertidumbre acotada.

Por lo tanto, para diseñar controladores robustos en estabilidad para la Planta 2 se toma en cuenta (se propone, realmente) el modelo considerando la dinámica no modelada del sistema, como se muestra en la ecuación 5.4

$$\Delta(s) * W_m(s) = \frac{0.014 s^2 + 0.472 s + 2}{10 s^2 + 0.8 s + 10} \qquad \text{Ecuación 5.3}$$

$$G'(s) = \frac{1.8}{s^2(s+2)} * \frac{10.014 s^2 + (1.27) s + 12}{10 s^2 + (0.8) s + 10} \quad ; \quad H(s) = 1 \qquad \text{Ecuación 5.4}$$

Nótese que el numerador y el denominador de la dinámica no modelada, deben ser similares, para justificar el hecho de haber despreciado esta dinámica, aproximando esta expresión a uno.

Se diseña por este medio un controlador, el Controlador 6, para la Planta 2 que se vio claramente es muy inestable. Posteriormente se probará la estabilidad de dicho sistema para los casos cuando se compensó con el PD de comportamiento óptimo (Controlador 5) y con este controlador establemente robusto (Controlador 6).

El modelo del Controlador 6 a implementar, y el rango de valores permitidos de las constantes es el indicado en la ecuación 5.5 (propuesto por Krohling [13]):

$$G_c(s) = k_1 \frac{s^2 + 2 s k_4 k_5 + k_5^{\,2}}{(s + k_2)(s + k_3)} \qquad \text{Ecuación 5.5}$$

- $1 < k_1 < 1000$
- $1 < k_2 < 100$
- $1 < k_3 < 100$
- $0 < k_4 < 1$
- $0,1 < k_5 < 100$

La función fitness empleada para este problema se puede encontrar en el Anexo 2. Las opciones escogidas dentro del GUI para la optimización fueron las mostradas en la tabla 5.6:

Opción	Valor
Tamaño de la población	80
Número máximo de generaciones	100
Método de selección	Torneo
Tamaño del torneo	5
Factor de mutación	0.01
Tipo de cruce	Doble punto
Factor de cruce	0.5

Tabla 5.6 Opciones para la optimización del Controlador 6

Al cabo de las 100 generaciones el software llegó al máximo valor de $\alpha(w,k)^{1/2}$ =0.976 que es menor a 1. Los valores de las constantes son las indicadas en la tabla 5.7.

	Controlador 6
K1	273
K2	13.61
K3	73.14
K4	0.21
K5	0.5

Tabla 5.7 Constantes del Controlador 6

En las figuras 5.15, 5.16 y 5.17 se muestra el lugar geométrico de las raíces del Sistema 2 sin compensación, compensado con el Controlador 5 y con el Controlador 6 respectivamente.

Si bien el Controlador 6 brinda estabilidad, no es un controlador muy adecuado en la práctica porque como puede verse en la figura 5.18 su comportamiento es muy poco satisfactorio. Por esta razón es que se vuelve necesario el diseño de este mismo tipo de controlador pero que a la vez tenga un buen comportamiento.

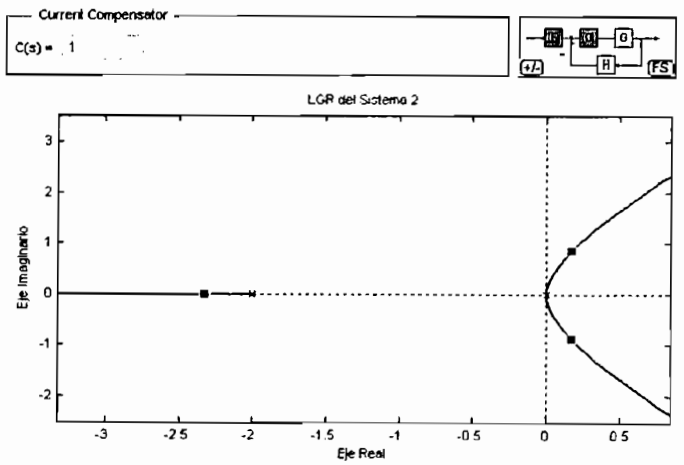


Figura 5.15 LGR en lazo abierto del Sistema 2 original

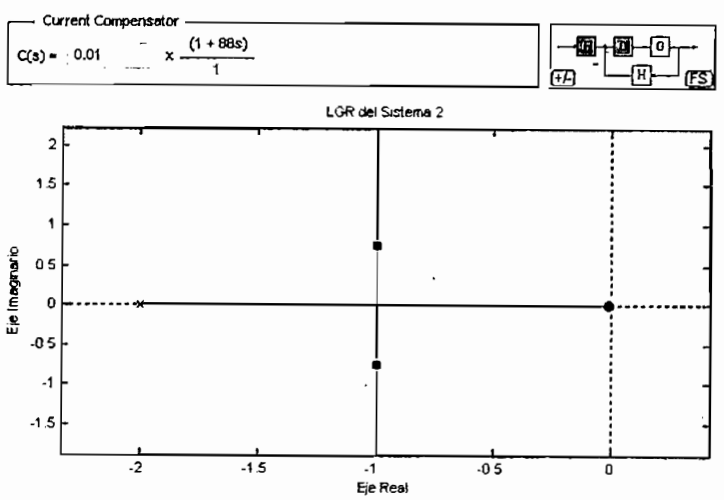


Figura 5.16 LGR en lazo abierto de la Planta 2 compensada con el Controlador 5

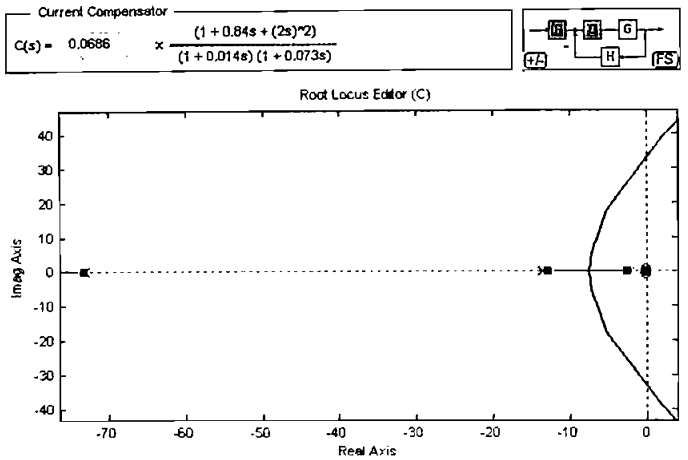


Figura 5.17 LGR en lazo abierto de la Planta 2 compensada con el Controlador 6

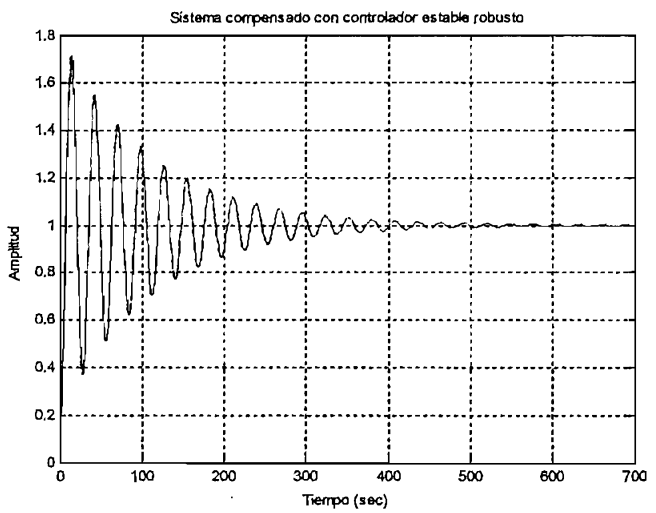


Figura 5.18 Respuesta de la Planta 2 compensada con el Controlador 6

5.3.2 DISEÑO DE CONTROLADORES ÓPTIMOS ROBUSTOS EN ESTABILIDAD

Un controlador óptimo robusto es aquel que además de su robustez de estabilidad, presenta un buen comportamiento. Este comportamiento se puede traducir en índices de error como el ISE, ITSE, ITAE y otros. Para obtener un comportamiento óptimo se busca reducir alguno de estos índices.

En la sección anterior se vio la necesidad de mejorar el comportamiento del Controlador 6, para este efecto, Jamshidi [12] sugiere emplear la minimización de la integral del cuadrado del error (ISE), que se explicó previamente en el Capítulo 4, para diseñar un nuevo controlador, el Controlador 7.

Se escribió la respectiva función de aptitud (referirse al Anexo 2) para minimizar el índice de error en base al algoritmo explicado en el Capítulo 4 y empleando el desarrollo del índice ISE para n=5 (referirse al Anexo 1). Las opciones escogidas para la optimización fueron las que se muestran en la tabla 5.8.

Opción	Valor
Tamaño de la población	25
Número máximo de generaciones de espera si no existe mejora	50
Método de selección	Torneo
Tamaño del torneo	8
Factor de mutación	0.2
Tipo de cruce	Doble punto
Factor de cruce	0.6

Tabla 5.8 Opciones para la optimización del Controlador 7

El software llegó al mínimo de $J_5(k)=0.1747$ luego de 165 generaciones. Los valores de las constantes se presentan en la tabla 5.9.

	Controlador 7
k1	2856.15
K2	27.65
K3	28.31
K4	0.85
K5	2.34

Tabla 5.9 Constantes del Controlador 7

Con este Controlador 7 se obtuvo la respuesta temporal mostrada en la figura 5.19 con las características presentadas en la tabla 5.10

	Planta 2 con Controlador 7
Mp	33.2 %
ts	1.13 s
Ep	0 %

Tabla 5.10 Características temporales de Planta 2 con Controlador 7

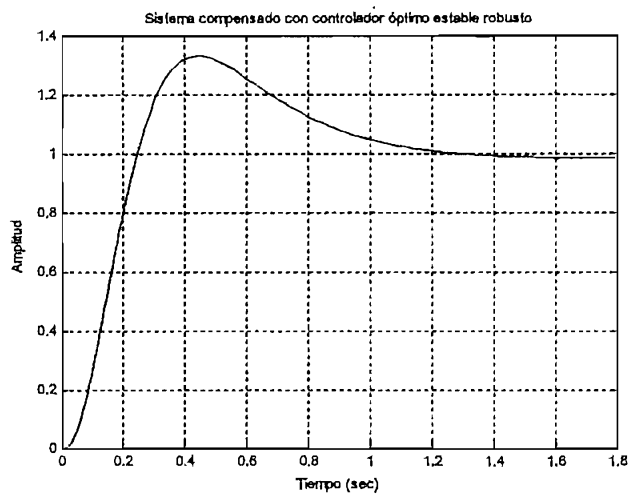


Figura 5.19 Respuesta temporal de la Planta 2 compensada con el Controlador 7

Por medio del MSM se puede ver que la planta compensada es realmente más estable que la planta sola, hecho que se observa también por medio de las gráficas de las respuestas temporales.

	Planta 2 original	Planta 2 con Controlador 7
Margen de fase	- 24.4°	83.1 °
Margen de ganancia	∞	16 dB
MSM	0.393	0.705

Tabla 5.11 Características en frecuencia de: Planta 2 original y compensada con Controlador 7

Es interesante mostrar también las respuestas en frecuencia. En la figura 5.20 se aprecia el diagrama de Bode de magnitud y fase para la Planta 2 original y en la Figura 5.21 la Planta 2 compensada con el Controlador 7. Sus correspondientes características en frecuencia se indican en la Tabla 5.11.

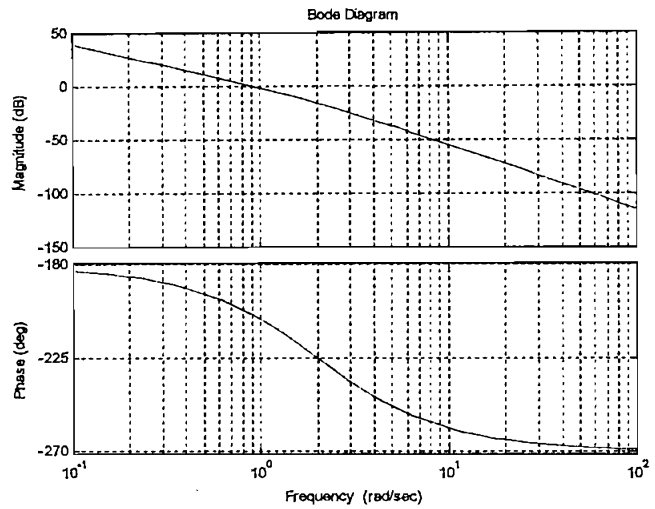


Figura 5.20 Diagrama de Bode en lazo abierto de la Planta 2

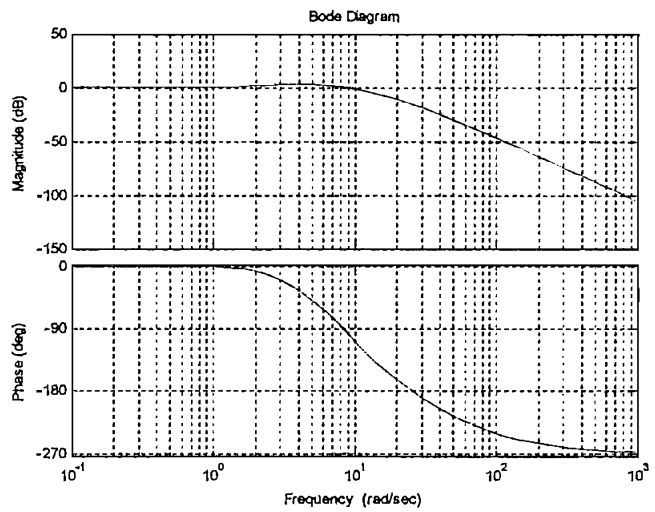


Figura 5.21 Diagrama de Bode en lazo abierto de la Planta 2 compensada con Controlador 7

Otra forma de mejorar el comportamiento de un controlador robusto para hacerlo óptimo, es empleando la función de aptitud que se introdujo en la sección 5.2.2 de este mismo capítulo (Anexo 2). Con esta función de aptitud se busca optimizar la respuesta transitoria del controlador.

Se plantea diseñar el Controlador 8 por este método para compensar la Planta 1 descrita anteriormente por la ecuación 5.1.

Nuevamente, es necesario especificar la función de incertidumbre (ver ecuación 5.6) para diseñar el controlador, con lo que la planta real sin simplificaciones queda expresada por la ecuación 5.7.

$$\Delta(s) * W_m(s) = \frac{-0.05s+1}{s^2+0.15s+9} \qquad \text{Ecuación 5.6}$$

$$G'(s) = \frac{6.28}{s^2+2s+0.96} * \frac{s^2+(0.10)s+10}{s^2+(0.15)s+9} \;;\; H(s)=1 \qquad \text{Ecuación 5.7}$$

El modelo del Controlador 8 se muestra en la ecuación 5.8. Se escogió este tipo de controlador para asegurar que no exista error de posición y mejorar la respuesta transitoria por medio de la red de compensación. El archivo de la función de aptitud para diseñar este controlador se encuentra en el Anexo 2.

$$G_c(s) = k_1 \frac{(s+k_2)}{s\,(s+k_3)} \qquad \text{Ecuación 5.8}$$

Como se desea mejorar la respuesta transitoria de la planta, la red de compensación debe ser una red en adelanto, esto significa que k3>k2, y esto se refleja en los límites escogidos para las constantes del Controlador 8, que se muestran a continuación:

- $0 < k_1 < 100$
- $0 < k_2 < 1$
- $0 < k_3 < 100$

El software corrió con las opciones detalladas en la tabla 5.13, y al cabo de 80 generaciones el mejor individuo encontrado fue el indicado en la tabla 5.12.

	Controlador 8
k1	41.32
K2	0.57
K3	241.4

Tabla 5.12 Constantes del Controlador 8

Opción	Valor
Tamaño de la población	10
Número máximo de generaciones	80
Método de selección	Ruleta
Factor de mutación	0.3
Tipo de cruce	Doble punto
Factor de cruce	0.6

Tabla 5.13 Opciones para la optimización del Controlador 8

Con la compensación provista por el Controlador 8, la Planta 1 se comporta de la forma indicada en la figura 5.22 con las características detalladas en la tabla 5.14

	Planta 1 con Controlador 8
Mp	0 %
ts	3.49 s
Ep	0 %

Tabla 5.14 Características temporales de Planta 1 con Controlador 8

En las figura 5.23 se aprecia el lugar geométrico de las raíces del sistema compensado.

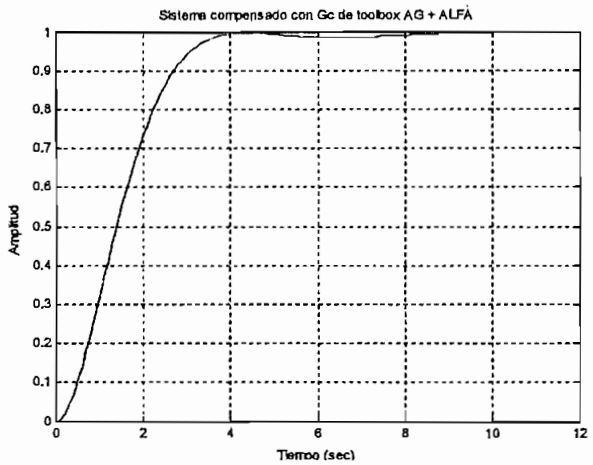


Figura 5.22 Respuesta temporal de la Planta 1 compensada con el Controlador 8

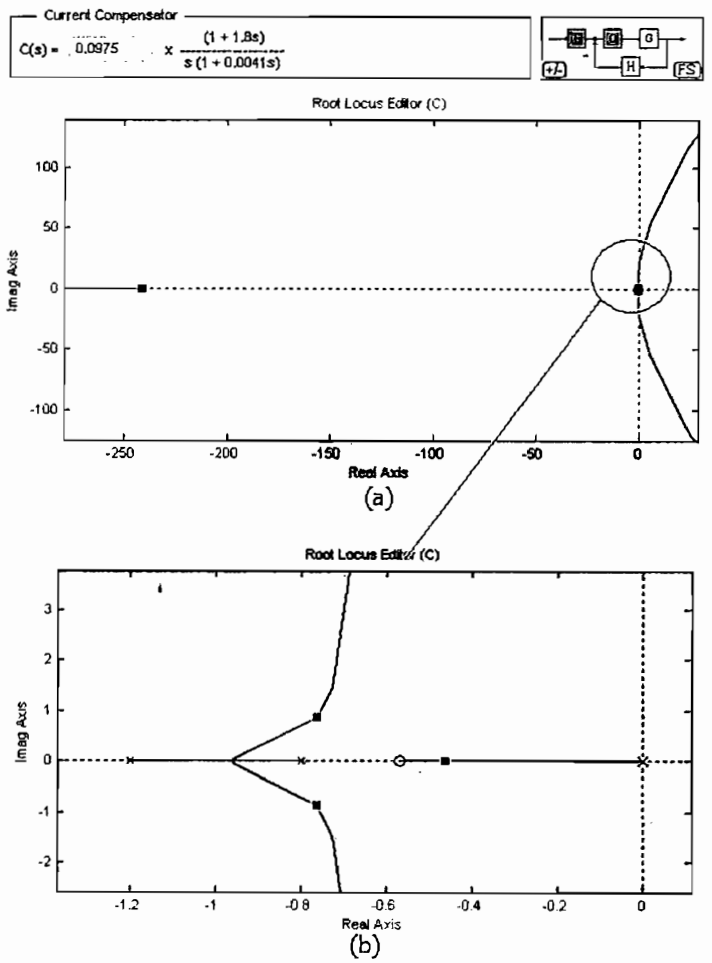


Figura 5.23 LGR en lazo abierto del Sistema 1 con el Controlador 8 (a), acercamiento o *zoom in* de la zona señalada (b)

5.3.2.1 Comprobación de estabilidad con el Teorema de la pequeña ganancia

Para comprobar la estabilidad de los Controladores 7 y 8, se emplea el teorema de la mínima ganancia explicada en el Capítulo 4.

El teorema de la pequeña ganancia asegura que si la magnitud de la función de incertidumbre multiplicativa Δ_m, (aquí se refiere al valor total de la incertidumbre, es decir; las ecuaciones 5.3 y 5.6) es siempre menor a la magnitud del inverso de la función de sensibilidad complementaria *T(s)* para cualquier frecuencia, el sistema es establemente robusto. Recordar que esta condición es suficiente pero no necesaria.

Las incertidumbres multiplicativas se indicaron ya, en las ecuaciones 5.3 y 5.6 para el Sistema 2 y 1 respectivamente y son éstas las que se grafican en las figuras 5.24 y 5.25.

En la Figura 5.24 se muestra el diagrama de magnitudes (Bode) de $\Delta_m(s)$ y de 1/T(s) para la Planta 1 con la compensación del Controlador 8 mientras que en la figura 5.25, las magnitudes de $\Delta_m(s)$ y de *1/T(s)* para la Planta 2 con la compensación del Controlador 7.

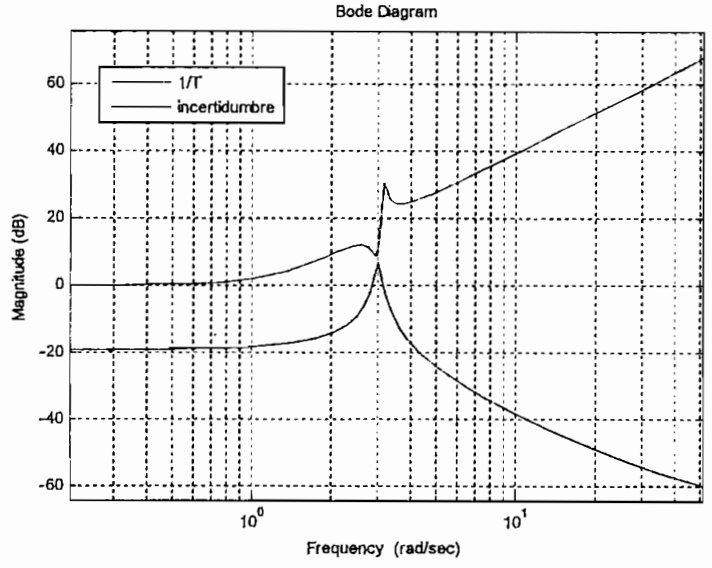


Figura 5.24 Magnitud de Δ_m y 1/T para la Planta 1 compensada con Controlador 8

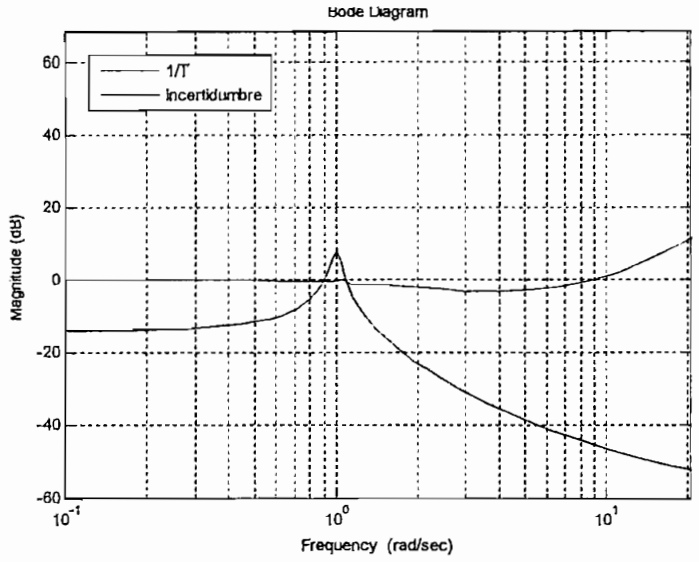


Figura 5.25 Magnitud de Δ_m y 1/T para la Planta 2 compensada con Controlador 7

Para la Planta 1, la condición suficiente se cumple con el Controlador 8, sin embargo para la Planta 2 con el Controlador 7 no es así, sin embargo esto no implica que no sea robusto como se ve en la Sección 5.3.2.2.

5.3.2.2 Incertidumbre en la planta

Para comprobar de una forma más intuitiva la robustez de un controlador, se analiza la respuesta temporal de la planta respectiva.

En la figura 5.26 se ve la respuesta temporal de la Planta 1 perturbada o real compensada con el Controlador 8 (controlador óptimo establemente robusto) y con el Controlador 1 (PID tradicional).

Se analiza también a continuación, la robustez de algunos de los controladores diseñados para compensar a la Planta 2. Se presenta nuevamente algunas figuras ya indicadas antes para visualizar mejor el comportamiento de los distintos controladores en la tabla 5.15.

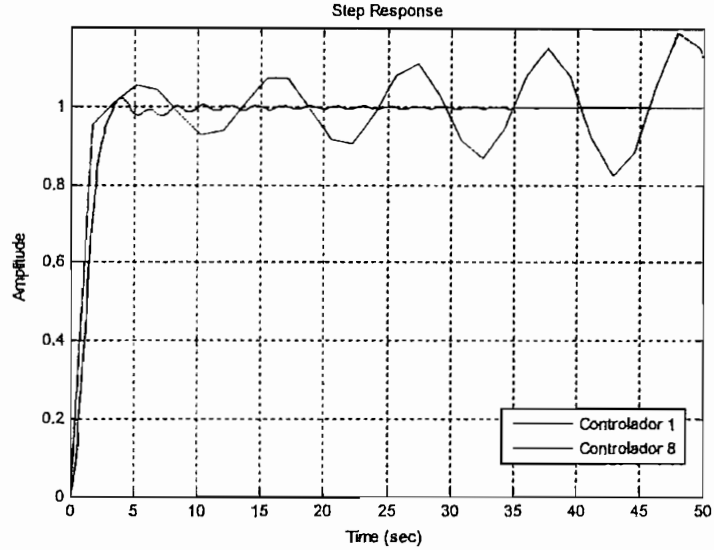
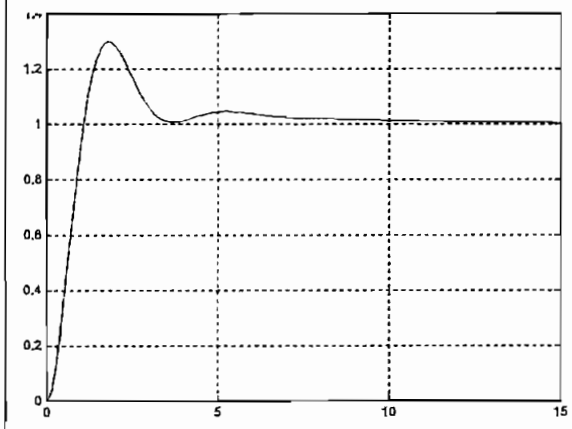
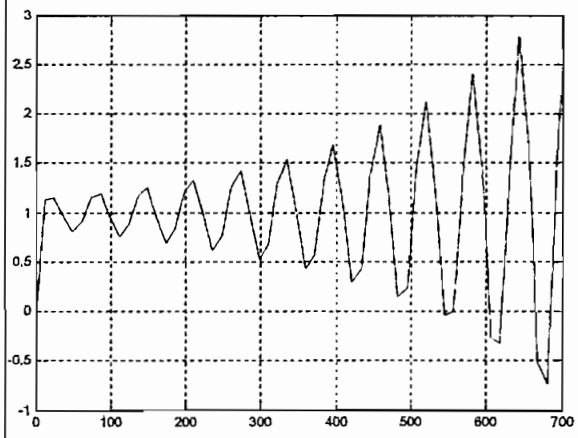
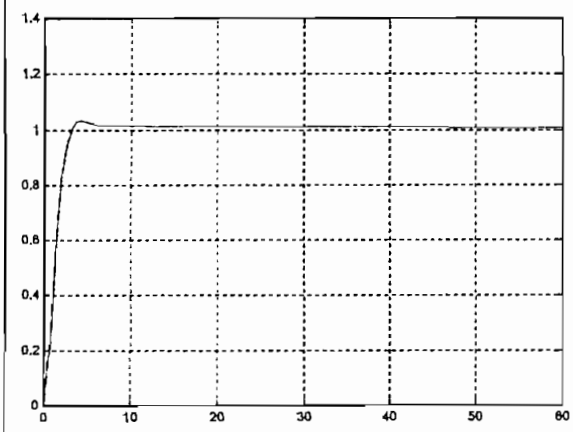
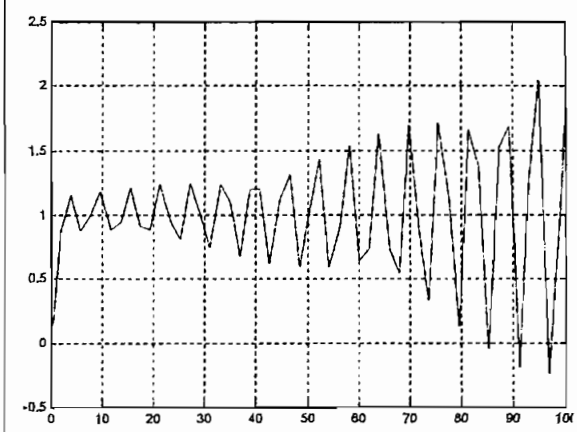
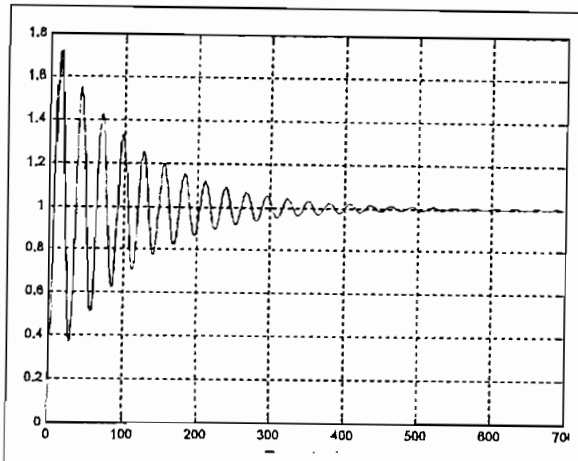
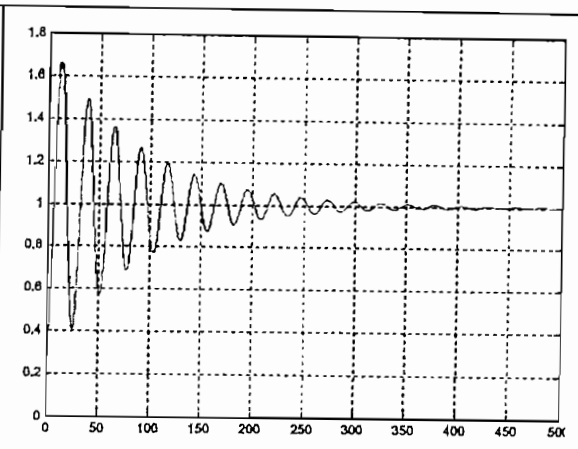


Figura 5.26 Respuesta temporal de la Planta 1 perturbada compensada con el Controlador 1 y 8

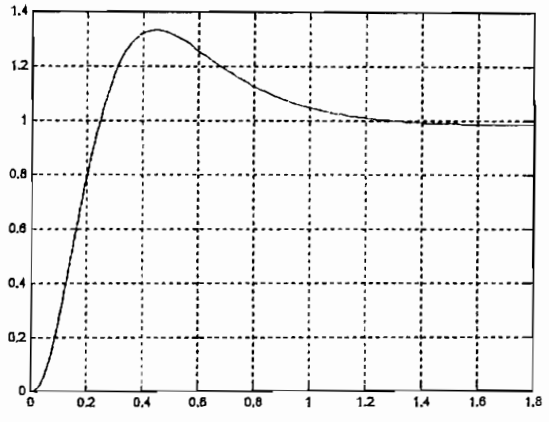
Planta 2 sin incertidumbre	Planta 2 con incertidumbre
	
PID clásico (Controlador 4)	PID clásico (Controlador 4)
	
PID de comportamiento óptimo (Controlador 5)	PID de comportamiento óptimo (Controlador 5)



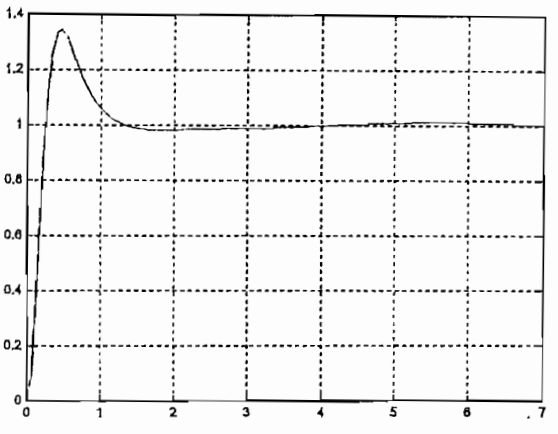
Controlador establemente robusto
(Controlador 6)



Controlador establemente robusto
(Controlador 6)



Controlador óptimo establemente robusto
(Controlador 7)



Controlador óptimo establemente robusto
(Controlador 7)

Tabla 5.15 Respuestas temporales de la Planta 1 con y sin incertidumbre para los Controladores: 4, 5, 6 y 7

Por medio de esta tabla se ve claramente la robustez de los Controladores 6 y 7 además del comportamiento óptimo de este último.

5.3.3 DISEÑO DE CONTROLADOR ROBUSTO EN RECHAZO A PERTURBACIONES

Para diseñar un controlador que sea robusto para rechazar perturbaciones es necesario utilizar la optimización (maximización) de la raíz cuadrada de la

función beta: β(w,k) con la restricción de que no sobrepase un cierto valor γ, especificado por el diseñador, (ver Capítulo 4). La función de ponderación de la incertidumbre multiplicativa es la descrita por la ecuación 5.9

$$W_m = \frac{1}{s+1} \qquad \text{Ecuación 5.9}$$

Esta función, sugerida por [8], provee el rechazo a perturbaciones a frecuencias medias y altas pues es un filtro pasabajos.

Se diseñará primeramente un controlador PID por métodos tradicionales, Controlador 9, y un controlador robusto en atenuar las perturbaciones actuantes en la planta, Controlador 10, para la Planta 3 que está definida por la función de transferencia indicada en la ecuación 5.10

$$G(s) = \frac{0.8}{s(0.5s+1)} \quad ; \quad H(s) = 1 \qquad \text{Ecuación 5.10}$$

El rango de valores iniciales para las constantes del PID se escoge entre 0 y 30 [8]; la función fitness empleada se encuentra en el Anexo 2. Las opciones escogidas para el toolbox de los algoritmos genéticos son las ya mostradas en la tabla 5.6.

Al cabo de las 100 generaciones, el individuo encontrado es el indicado en la tabla 5.16. Con estos valores, $\beta(w,k)^{1/2} = 0.09$ (que es menor que 0.1.)

	Controlador 9	Controlador 10
Kd	2.95	0.65
Kp	11.83	2.28
Ki	-	0.65

Tabla 5.16 Constantes del Controlador 9

En las figuras 5.27, 5.28 y 5.29 se muestra las respuestas temporales de: Planta 3 sin compensación, compensada con el Controlador 9 y con el Controlador 10 respectivamente. Las respectivas características temporales se presentan detalladamente más adelante en la tabla 5.19.

Aquí también ocurre lo que se esperaba (algo similar al comportamiento de los controladores robustos en estabilidad), el Controlador 10 provee una compensación razonable a la Planta 3 pero no tiene un desempeño óptimo. Por esta razón se diseña un nuevo controlador que a más de tener robustez en rechazo a perturbaciones tenga un comportamiento óptimo, como se detalla en la Sección 5.3.4.

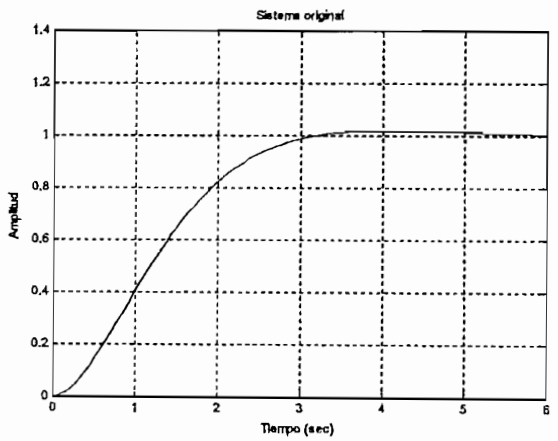


Figura 5.27 Respuesta original de la Planta 3 a una entrada paso

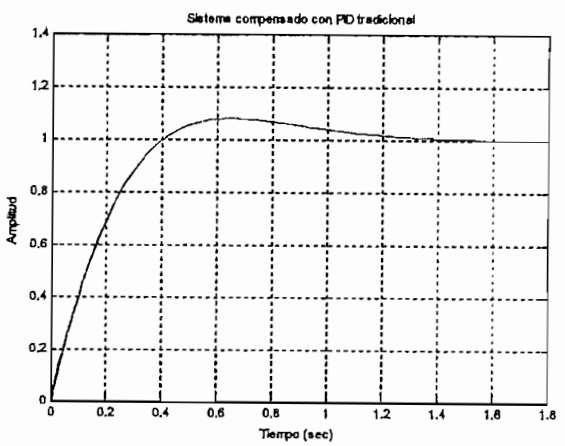


Figura 5.28 Respuesta a una entrada paso de la Planta 3 compensada con Controlador 9

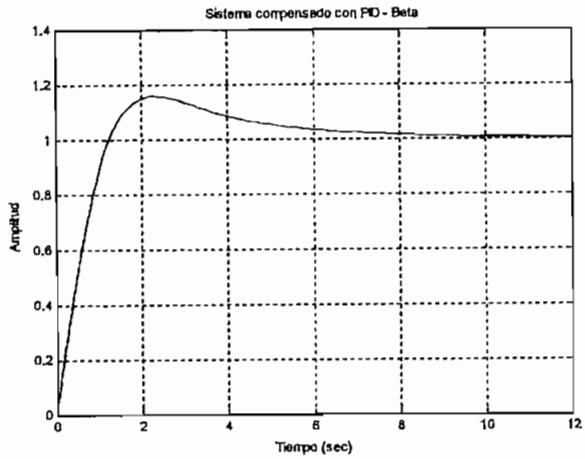


Figura 5.29 Respuesta a una entrada paso de la Planta 3 compensada con Controlador 10

5.3.4 DISEÑO DE CONTROLADOR ROBUSTO ÓPTIMO EN RECHAZO A PERTURBACIONES

Para diseñar un controlador óptimo robusto en rechazo a perturbaciones se busca reducir, como ocurrió en la sección 5.3.2, el índice ISE y/o también empleando una función fitness que busca optimizar el comportamiento transitorio.

El nuevo controlador PID que se diseña minimizando el ISE es el Controlador 11; escribiendo la respectiva función de aptitud (referirse al Anexo 2) para minimizar el índice de error en base al algoritmo explicado en el Capítulo 4 y empleando el desarrollo del índice ISE para n=3 (referirse al Anexo 1) el software encontró, después de 120 generaciones, el individuo descrito por la tabla 5.17 con el que se obtuvo $J_3(k)=0.0108$; las opciones escogidas para la optimización se muestran en la tabla 5.18.

	Controlador 11
Kp	29.7
Kd	29.8
Ki	0.15

Tabla 5.17 Constantes del Controlador 11

Opción	Valor
Tamaño de la población	35
Número máximo de generaciones de espera si no existe mejora	50
Método de selección	Torneo
Tamaño del torneo	8
Factor de mutación	0.2
Tipo de cruce	Doble punto
Factor de cruce	0.6

Tabla 5.18 Opciones para la optimización del Controlador 11

Con el Controlador 11, la respuesta temporal de la Planta 3 es la mostrada en la figura 5.30.

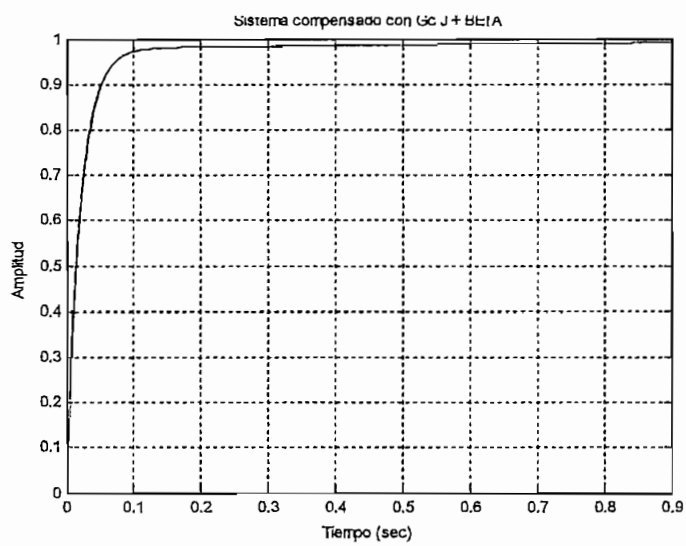


Figura 5.30 Respuesta temporal de la Planta 3 compensada con el Controlador 11

Se aprecia claramente, que el Controlador 11 tiene un mejor comportamiento que el Controlador 9 y 10. Las características temporales para los 3 casos se ven en la tabla 5.19.

	Planta 3 original	Planta 3 con Controlador 9	Planta 3 con Controlador 10	Planta 3 con Controlador 11
Mp	1.73%	7.92%	15.7%	0%
ts	2.91s	1.16s	7.32s	0.137s
Ep	0%	0%	0%	0%

Tabla 5.19 Características temporales de Planta 3

A continuación, se prueba diseñar el mismo tipo de controlador óptimo robusto, el Controlador 12, para la Planta 1 ya estudiada anteriormente pero esta vez buscando reducir las características temporales transitorias y el error.

Se opta una vez más por un controlador PID sobretodo para ver cómo varía las respuestas al comparar entre controladores del mismo tipo (PID) pero diseñados bajo distintos criterios. El archivo de la función de aptitud para diseñar este controlador se encuentra en el Anexo 2.

El software encontró, con las opciones detalladas en la tabla 5.20, al cabo de 90 generaciones, el individuo indicado en la tabla 5.21.

Opción	Valor
Tamaño de la población	10
Número máximo de generaciones	80
Método de selección	Ruleta
Factor de mutación	0.3
Tipo de cruce	Doble punto
Factor de cruce	0.6

Tabla 5.20 Opciones para la optimización del Controlador 12

	Controlador 12
Kp	578.34

Ki	0.0857
Kd	906.51

Tabla 5.21 Constantes del Controlador 12

Con la compensación provista por el Controlador 12, la Planta 1 se comporta de la forma indicada en la tabla 5.23.

5.3.4.1 Perturbación en la planta

Para poder apreciar si los controladores diseñados, son realmente robustos, es necesario incluir una perturbación a la salida de la planta, que pasará a través del filtro indicado en la ecuación 5.9. Para el caso de la Planta 3 se escoge la perturbación indicada en la ecuación 5.11.

$$d_y(t) = 2\,\mathrm{sen}(2\pi\, 0.15\, t) \qquad \text{Ecuación 5.11}$$

Para el análisis de la Planta 3 perturbada se escogió al Controlador 9 (controlador PID clásico) y el Controlador 11 (robusto óptimo); las respuestas temporales de la Planta 3 perturbada y compensada con ambos controladores, se muestra en la figura 5.31, 5.32 y 5.33 respectivamente. Sus características temporales se indican en la tabla 5.22. Las características de la Planta 3 compensada con los dos controladores se muestran en la tabla 5.23.

	Planta 3	Planta 3 con Controlador 9	Planta 3 con Controlador 11
Mp	-	-	3.2 %
ts	-	-	2.1 s
Ep	-	-	0 %

Tabla 5.22 Características temporales de Planta 3 perturbada compensada con Controladores 9 y 11

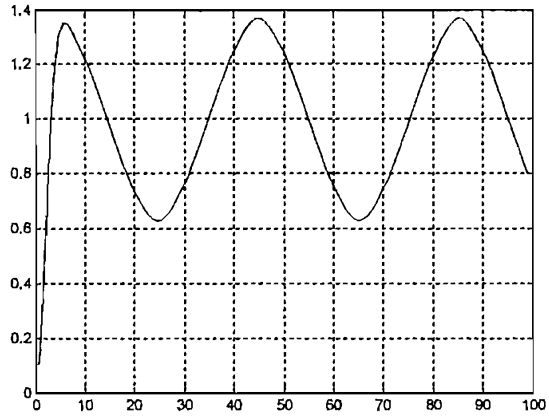


Figura 5.31 Respuesta temporal de la Planta 3 perturbada

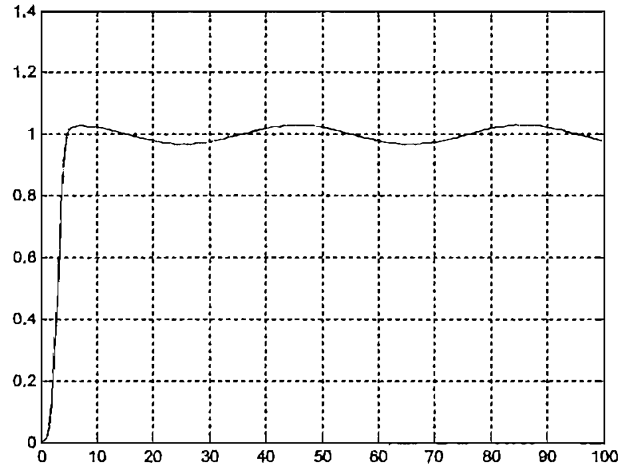


Figura 5.32 Respuesta Planta 3 perturbada compensada con Controlador 9

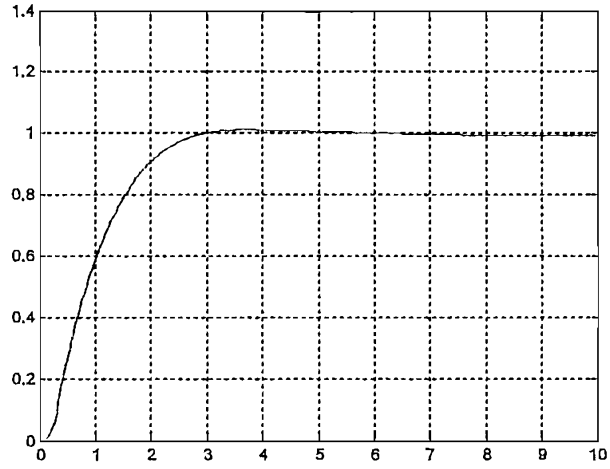


Figura 5.33 Respuesta Planta 3 perturbada compensada con Controlador 11

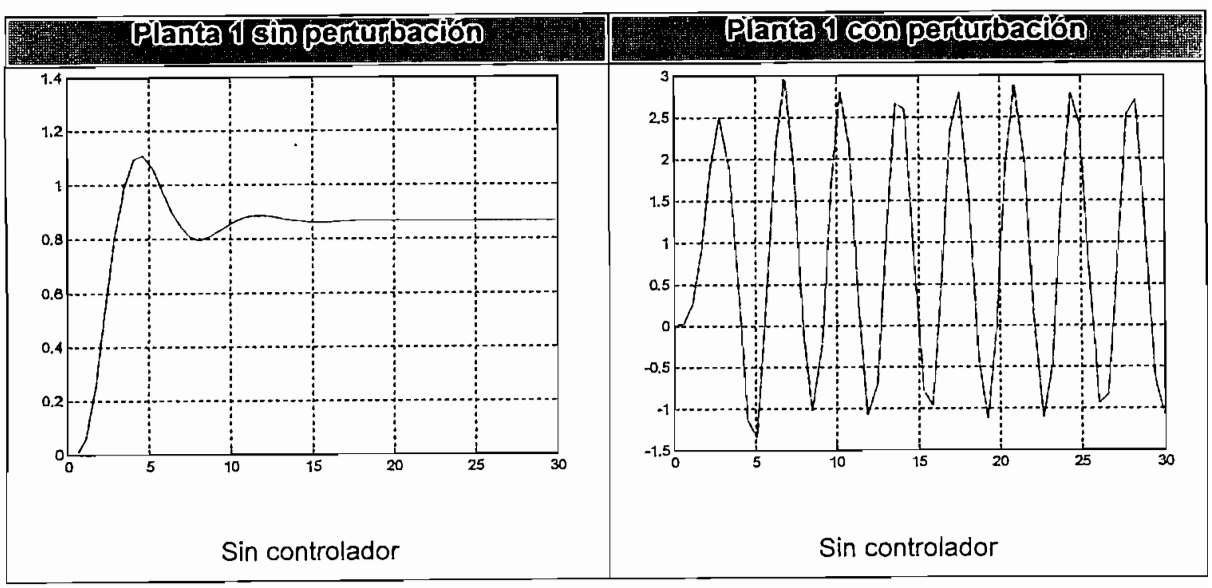
Observando estas características y los gráficos, es posible confirmar lo que se aseguró en los capítulos anteriores, el controlador PID clásico es en sí robusto pero no tanto como los controladores diseñados por medio de la optimización de las normas H infinito.

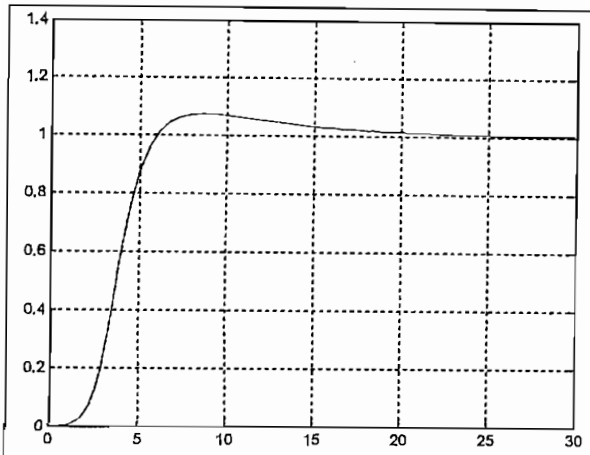
Finalmente, se muestra las respuestas de la Planta 1 bajo la influencia de la perturbación indicada en la ecuación 5.12, al ser compensada por los siguientes controladores:

- PID clásico (Controlador 1)
- Robusto en rechazo a perturbaciones (Controlador 13)
- Robusto óptimo en rechazo a perturbaciones (Controlador 12)

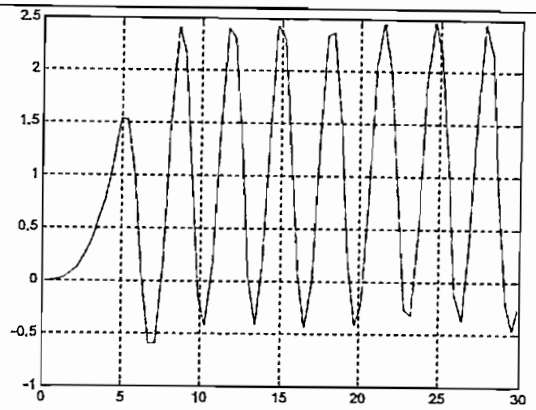
$$d_y(t) = 8\,\mathrm{sen}\,(2\pi 120\,t) \qquad \text{Ecuación 5.12}$$

Los Controladores 12 y 13 se diseñaron para un valor de atenuación de ζ=0.08 para dar mayor robustez. El Controlador 13 está formado por las constantes indicadas en la tabla 5.24.

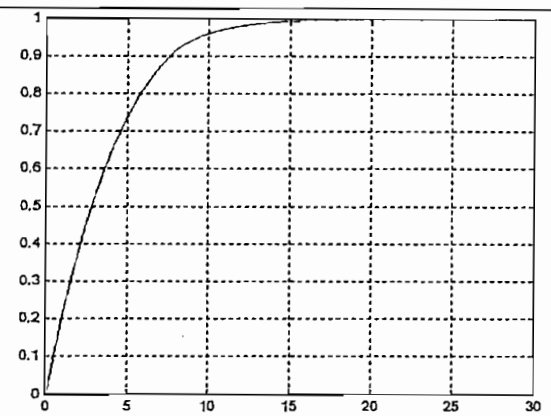




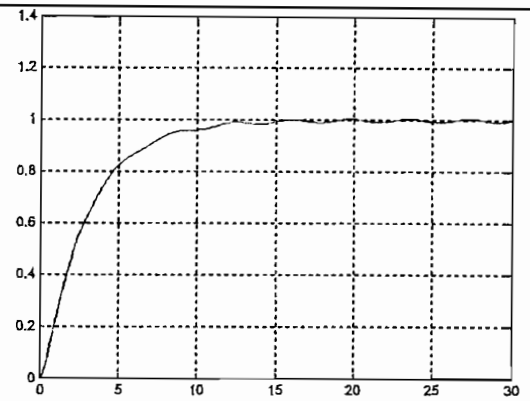
PID clásico (Controlador 1)



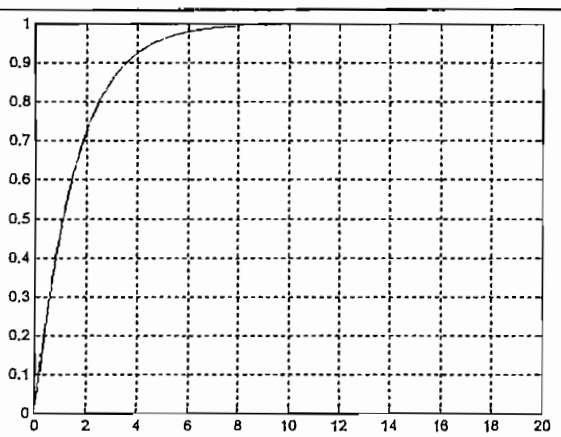
PID clásico (Controlador 1)



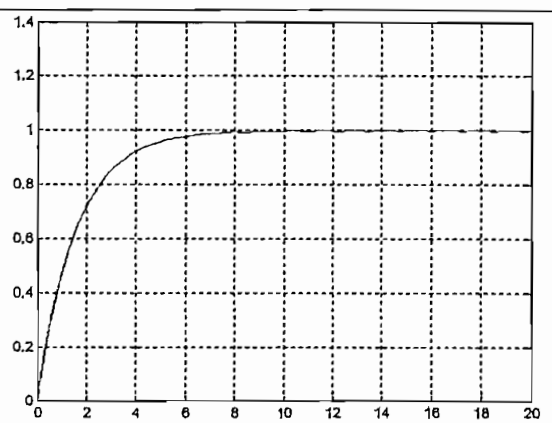
PID robusto en rechazo a perturbaciones (Controlador 13)



PID robusto en rechazo a perturbaciones (Controlador 13)



PID robusto óptimo en rechazo a perturbaciones (Controlador 12)



PID robusto óptimo en rechazo a perturbaciones (Controlador 12)

Tabla 5.23 Respuestas temporales de la Planta 1 con y sin perturbaciones para los Controladores: 1, 12 y 13

	Controlador 13
Kp	188.67
Ki	0.04
Kd	181.92

Tabla 5.24 Constantes del Controlador 13

5.4 RESUMEN

A continuación se presenta un resumen donde se indican las plantas y los controladores empleados en este capítulo. En la tabla 5.25 se pueden apreciar los modelos de las tres plantas a las cuales se les diseñó los controladores detallados en las Secciones 5.2 y 5.3.

NOTACIÓN	MODELO
Planta 1	$G(s)=\frac{6.28}{s^2+2s+0.96}$; $H(s)=1$
Planta 2	$G(s)=\frac{1.8}{s^2(s+2)}$; $H(s)=1$
Planta 3	$G(s)=\frac{0.8}{s(0.5s+1)}$; $H(s)=1$

Tabla 5.25 Modelos de las Plantas 1, 2 y 3

En la tabla 5.26 se presentan el tipo de controladores diseñados con su respectiva planta.

NOTACIÓN	PLANTA	TIPO
Controlador 1	1	Tradicional
Controlador 2	1	Optimizado con Simulink
Controlador 3	1	Óptimo

Controlador 4	2	Tradicional
Controlador 5	2	Óptimo
Controlador 6	2	Robusto en estabilidad
Controlador 7	2	Robusto óptimo en estabilidad
Controlador 8	1	Robusto óptimo en estabilidad
Controlador 9	3	Tradicional
Controlador 10	3	Robusto en rechazo a perturbaciones
Controlador 11	3	Robusto óptimo en rechazo a perturbaciones
Controlador 12	1	Robusto óptimo en rechazo a perturbaciones
Controlador 13	1	Robusto en rechazo a perturbaciones

Tabla 5.26 Tipos de controladores diseñados

En la tabla 5.27 se muestran los valores de las constantes de los trece controladores empleados. Los modelos PID, A y B se muestran en las ecuaciones 5.13, 5.14 y 5.15 respectivamente.

$$G_C(s) = \frac{s^2\,Kd + s\,Kp + Ki}{s}$$ Ecuación 5.13

$$G_c(s) = k_1 \frac{s^2 + 2s\,k_4k_5 + k_5^{\,2}}{(s+k_2)(s+k_3)}$$ Ecuación 5.14

$$G_c(s) = k_1 \frac{(s+k_2)}{s\,(s+k_3)}$$ Ecuación 5.15

NOTACIÓN	MODELO	CONSTANTES
Controlador 1	PID	Kp=1.31 Kd=0.53 Ki=0.49
Controlador 2	PID	Kp=31.62 Kd=2.63 Ki=0.0001
Controlador 3	PID	Kp=33.63 Kd=21.7 Ki=28.66
Controlador 4	PD	Kp=0.47 Kd=2.35
Controlador 5	PD	Kp=0.88 Kd=0.01

Controlador 6	A	K1=273 K2=13.61 K3=73.14 K4=0.21 K5=0.5
Controlador 7	A	K1=2856.15 K2=27.65 K3=28.31 K4=0.85 K5=2.34
Controlador 8	B	K1=41.32 K2=0.57 K3=241.4
Controlador 9	PD	Kp=2.95 Kd=11.83
Controlador 10	PID	Kp=0.65 Kd=2.28 Ki=0.65
Controlador 11	PID	Kp=29.7 Kd=29.8 Ki=0.15
Controlador 12	PID	Kp=578.34 Kd=0.0857 Ki=906.51
Controlador 13	PID	Kp=188.67 Kd=0.04 Ki=181.92

Tabla 5.27 Constantes de Controladores 1, 2,…., 13

CAPÍTULO 6

Conclusiones y Recomendaciones

6.1 CONCLUSIONES

Al finalizar el presente trabajo se concluye en primer lugar que se logró los objetivos planteados en un inicio. Se realizó un estudio de los algoritmos genéticos y de su aplicación orientada a la optimización de controladores PID y robustos. Se consiguió optimizar controladores PID empleando algoritmos genéticos así como también diseñar controladores robustos empleando dicha técnica, todo esto con la ayuda del software Matlab.

6.1.1 CONCLUSIONES GENERALES

Los algoritmos genéticos son una técnica de búsqueda (optimización) efectiva que no necesita mayor información del problema a resolver (lo que le hace muy versátil) y tampoco se requiere de que el diseñador tenga muchos conocimientos matemáticos (lo que le hace bastante accesible) a diferencia de otros métodos de optimización como el del gradiente por ejemplo.

La técnica de los algoritmos genéticos se adapta perfectamente a problemas de control puesto que en la mayoría de casos se necesita optimizar algún parámetro pero satisfaciendo alguna condición de restricción, y en este tipo de problemas los algoritmos genéticos trabajan muy bien gracias a que tienen la capacidad de reflejar cualquier restricción dentro de la función de aptitud con que trabaja.

Dentro del control siempre será necesario establecer compromisos entre las distintas características de los sistemas porque aunque se diseñe los mejores

controladores existentes, no se podrá tener nunca excelentes características transitorias con una excelente estabilidad simultáneamente, por ejemplo.

En la implementación de controladores, en la práctica, sería deseable considerar como requerimiento controladores robustos y óptimos. Un sistema es robusto cuando su respuesta no se ve sensiblemente alterada aun cuando existan perturbaciones o incertidumbres actuando sobre la planta y es óptimo cuando se consigue la mejor respuesta posible.

6.1.2 CONCLUSIONES ESPECÍFICAS

La norma H infinita es una herramienta excelente para diseñar controladores robustos en estabilidad y en rechazo a perturbaciones, y para comprobar robustez de sistemas de control. Esta norma se ve traducida en problemas de minimización de funciones (alfa y beta respectivamente).

En la optimización de los controladores PID por medio de algoritmos genéticos, se obtuvo mejor resultado cuando se escribió la función de aptitud para minimizar las características temporales del sistema que cuando se usó el bloque de optimización del Simulink, incluso en ocasiones no fue capaz de satisfacer los requerimientos de diseño y fue necesario "suavizarlos". Pero por otro lado se necesitó de mucho más conocimiento de la teoría de los algoritmos genéticos.

Para diseñar controladores óptimos es necesario minimizar algún parámetro que represente la desviación o diferencia entre el valor deseado de la respuesta y el valor real que se obtiene, este tipo de parámetros son los índices de comportamiento. Aunque también existe otra forma de hacerlo, que aunque no es tan elegante, es efectiva y consiste simplemente en calcular las características temporales iterativamente hasta que éstas sean mínimas.

Los controladores óptimos robustos se comportan efectivamente como se esperaba, mejor que los controladores que son únicamente robustos.

La función de ponderación de las incertidumbres (W) es de vital importancia dentro del diseño del controlador robusto en estabilidad pues permite dar al controlador "una idea" de la incertidumbre actuante en la planta.

El MSM permite dar una idea más amplia de margen de estabilidad, al compararlo con el margen de fase y margen de ganancia; el MSM abarca estos dos criterios en uno sólo.

Como se puede apreciar en las tablas 5.15 y 5.23, los controladores PID trabajan bien bajo condiciones nominales de operación, pero al introducir perturbaciones o incertidumbres en el modelo, éstos se deterioran visiblemente. De las mismas tablas puede concluirse que los controladores robustos resultan tener efectivamente robustez pues en ambos casos de operación, los controladores consiguen estabilizar al sistema aunque con características transitorias muy poco deseables; y es aquí donde se aprecia claramente la eficiencia de los controladores robustos óptimos que a más de mantener a la planta estable en cualquier caso de trabajo, presentan un muy buen comportamiento.

6.2 RECOMENDACIONES

Si se desea trabajar con los algoritmos genéticos para diseñar controladores robustos óptimos empleando Matlab, es conveniente usar la versión profesional, porque la versión estudiantil presenta problemas (*bugs*) cuando se trabaja con funciones bastante largas como las que se generan en el problema de la robustez óptima. Si no es posible, se puede optar por lo siguiente: en las opciones del GUI se escoge un número muy reducido de generaciones, una población muy pequeña y se corre el algoritmo las veces que se considere necesarias tomando en cada resolución como valores iniciales los valores finales de la resolución anterior.

En la optimización de los controladores PID por medio de algoritmos genéticos se recomienda escribir una función de aptitud que permita minimizar las características temporales del sistema antes que usar el bloque de optimización del Simulink

En problemas de optimización dentro de la ciencia del control es aconsejable usar los algoritmos genéticos pues son fáciles de usar y efectivos.

Se recomienda emplear la teoría del control H (norma H infinita) para diseñar controladores robustos en estabilidad y en rechazo a perturbaciones debido a su simplicidad y fuerte sustento teórico.

Antes de iniciar el diseño de cualquier controlador es recomendable definir de antemano los compromisos o acuerdos que se desea entre los distintos parámetros de la planta en estudio, por ejemplo entre comportamiento y estabilidad.

Se aconseja sobremanera para la práctica, diseñar controladores robustos óptimos en lugar de simples controladores robustos.

ANEXO 1

DESARROLLO MATEMÁTICO DEL ISE (J_n)

En el Capítulo 4 mostró la fórmula matemática del ISE pero no se la resuelve por medio de la integración sino por medio del siguiente desarrollo. Este desarrollo fue provisto en su totalidad, por gentileza del Dr. Tohru Kawabe (kawabe@cs.tsukuba.ac.jp) profesor en la Universidad de Tsukuba (Japón), autor de numerosos estudios de control robusto.

Generalmente, se puede computar el valor del ISE de orden "n" usando el determinante de Hurwitz.

Defínase:

$e(s) = A(s)/B(s)$, donde:

$A(s) = a_0*s^n + a_1*s^{(n-1)} + \ldots + a_n;$
$D(s) = B(s)*B(s) = d_0*s^{(2n-2)} + d_1*s^{(2n-4)} + \ldots + d_{n-1}.$

De esta notación:

$$ISE = \frac{1}{2\pi j}\int_{-j\infty}^{+j\infty} \frac{D(s)}{A(s)A(-s)}ds$$

Luego, la siguiente formula se desprende:

$$ISE = J_n = [(-1)^{(n-1)} * H'_n] / [\ 2*a_0*H_n]\ ;\ \text{donde:}$$

H_n es el determinante de Hurwitz y que corresponde a:

$$H_n = \begin{bmatrix} a_1 & a_3 & a_5 & . & . & 0 & 0 & 0 \\ a_0 & a_2 & a_4 & . & . & . & . & . \\ 0 & a_1 & a_3 & a_5 & . & 0 & . & . \\ . & a_0 & a_2 & . & . & a_n & . & . \\ . & 0 & . & . & . & a_{n-1} & 0 & . \\ . & . & . & . & . & a_{n-2} & a_n & . \\ . & . & . & . & . & . & a_{n-1} & 0 \\ 0 & 0 & 0 & . & a_{n-6} & a_{n-4} & a_{n-2} & a_n \end{bmatrix}$$

H'_n es el determinante de la matriz H_n cambiada la primera fila por $d_0, d_1, \ldots, d_{(n-1)}$.

ANEXO 2

CD INCLUIDO EN EL TRABAJO

Junto con este trabajo, se incluye un disco compacto (CD) con la información detallada en las 3 siguientes carpetas:

Capítulos- contiene el texto, en formato *doc* (MS Word), de todo el trabajo.

Referencias- contiene las fuentes bibliográficas empleadas en este proyecto. Esta carpeta se encuentra a su vez subdividida en carpetas según el capítulo en el que se hace referencia a la bibliografía. Los archivos se encuentran en formato *pdf* (Adobe Acrobat) y *html*.

Software- aquí se encuentran los archivos (*scripts*) escritos para trabajar con el programa MatLab® 7.

Los archivos de nombre Anexo"n" son funciones (de aptitud) de extensión .m escritas para trabajar con el toolbox de algoritmos genéticos.

Los archivos *Planta1, 2, 3* son scripts para visualizar las respuestas de las plantas respectivas, que se indican a lo largo del Capítulo 5.

Finalmente, los archivos *SignalConstarint*, *s1_per* y *s2_per*, son diagramas de Simulink para emplear el bloque de optimización de respuestas temporales (el primero) y los otros dos para comprobar el funcionamiento de los controladores robustos en rechazo a perturbaciones, perturbando la planta del sistema.

Anexo1 (función de aptitud para optimización de comportamiento de controladores pid).- escrita para optimizar el comportamiento (minimizar error de posición, máximo sobreimpulso y tiempo de establecimiento) de una planta *cualquiera* compensada con un controlador PID.

Anexo2 (función de aptitud para maximizar función $\sqrt{\alpha(\omega,k)}$).- escrita para hallar el máximo de la raíz cuadrada de la función $\alpha(\omega,k)$ con la restricción de ser menor que uno.

Anexo3 (función de aptitud para minimizar función $J_5(k)$).- función para encontrar el mínimo de la función $J_5(k)$ correspondiente al ISE de un sistema con error de orden 5, con la restricción que a la vez se maximice la raíz cuadrada de la función $\alpha(\omega,k)$ y que ésta sea menor a uno. Trabaja a su vez con otra función (para la optimización de α) llamada *AG.m*.

Anexo4.- archivo .m que calcula el valor del ISE (J_n) de un sistema de error de orden n, para n=1 hasta n=5.

Anexo5 (función de aptitud para minimizar características temporales).- busca las mínimas respuestas temporales de un sistema de orden 5, con la restricción que a la vez se maximice la raíz cuadrada de la función $\alpha(\omega,k)$ y que ésta sea menor a uno. Trabaja junto con *AG.m*.

Anexo6 (función de aptitud para maximizar función $\sqrt{\beta(\omega,k)}$).- función para hallar el máximo de la raíz cuadrada de la función $\beta(\omega,k)$ con la restricción que sea menor a 0.1.

Anexo7 (función de aptitud para minimizar función $J_3(k)$).- desarrollada para hallar el mínimo de la función $J_3(k)$, con la restricción que a la vez se maximice la raíz cuadrada de la función $\beta(\omega,k)$ y que ésta sea menor a 0.1; esto lo logra trabajando a su vez con otra función (para la optimización de β) que se llama: *BG.m*.

Anexo8 (función de aptitud para minimizar características temporales).- función fue escrita para trabajar con toolbox de AG para hallar las mínimas respuestas temporales de un sistema de orden 3, con la restricción que a la vez se maximice la raíz cuadrada de la función $\beta(\omega,k)$ y que ésta sea menor a 0.08. Trabaja a su con *BG.m*.

REFERENCIAS BIBLIOGRÁFICAS

[1] BAHRAM, Shahian; "Control system design using Matlab", Prentice Hall, E.E.U.U., 1993.

[2] BAKER, Richard; "Genetic Algorithms in Search and Optimization"; (http://www.fenews.com/fen5/ga.html)

[3] BANDA Hugo, "Seminario: Sistemas inteligentes y Redes neuronales artificiales", Rama Estudiantil IEEE, Quito, 19 al 22 Abril de 2004

[4] BASILIO, J. C. & MATOS S. R.; "Design of PI and PID Controllers With Transient Performance Specification", IEEE transactions on education, vol 45, No 4, November 2002.

[5] BODENHOFER, Ulrich; "Genetic Algorithms: Theory and Applications", Fuzzy Logic Laboratorium Linz-Hagenberg (Lecture Notes), Third Edition, Winter 2003/2004.

[6] CALISTRU, Nicolae; "Mixed H2/Hinf PID Robust Control via Genetic Algorithms and MATHEMATICA Facilities", Gheorghe Asachi Technical University of Romania Dept.of Automatic Control & Industrial Informatics.

[7] CHANDRASEKHARAN, P., C.; "Robust Control of Linear Dynamical Systems", Academic Press, 1996.

[8] CHEN, Bor-Sen, et al; "A genetic approach to mixed H2/Hinf optimal PID control", IEEE control systems magazine, 15, 5, pág:51-60, 1995

[9] DORF, Richard; "Modern control systems", Addison Wesley, 7ma Edición, E.E.U.U.

[10] DOYLE, Jhon, et all; "Feedback Control Theory", Macillan Publishing, 1990.

[11] GIOVANNINI, Leonardo; "Seminario: Control Adaptativo", ICIECA, Quito, 28 y 29 Noviembre 2005

[12] JAMSHIDI, Mo; "Robust control systems with Genetic Algorithms", CRC Press, 2003.

[13] KROHLING, Renato; "Design of Optimal Disturbance Rejection PID Controllers Using Genetic Algorithms", IEEE transactions on Evolutionary Computation, Vol 5, No. 1, February 2001.

[14] MAN K F, et all; "Genetic Algorithms", Springer, 2001.

[15] MARCZYK, Adam; "Algoritmos genéticos y computación evolutiva", 2004; (http://the-geek.org/docs/algen/)

[16] MATHWORKS; "Genetic Algorithm and Direct Search Toolbox"

[17] MERELO, Juan; "Informática evolutiva"; (http://geneura.ugr.es/~jmerelo)

[18] OGATA, Katsuhiko; "Ingeniería de control moderna", Prentice Hall, 4ta Edición, EE.UU., 2003.

[19] PRADO, Andrés; "Control con inteligencia computacional", TESIS, Escuela Politécnica Nacional, 2005.

[20] PRAKASH, A. K.; "Guaranteed Optimal Tuning of PID Robust Controllers", 5th Asian Control Conference, 2004

[21] ROLLINS, Leo; "Robust Control Theory", Carnegie Mellon University, 1999; (http://www.ece.cmu.edu/~koopman/des_s99/control_theory/#chandra96)

[22] THAM, Ming; "Introduction to robust control", University of Newcastle Upon Tyne, 2002.

[23] "Algoritmos genéticos";
(http://www.esi2.us.es/~dco/algoritm.htm#Introducción)

[24] "Notas de lecturas de: Introducción al control robusto de Ming T. Tham", 2002; Newcastle University's Chemical and Process Engineering; (http://lorien.ncl.ac.uk/ming/Dept/wordept.htm)

[25] "Métodos de Ajuste de Controladores Convencionales";
(www.eie.fceia.unr.edu.ar/.../TP%20control%202-Electronicos/ tp1/ Teor%EDa /Ajuste%20de%20controladores.pdf)

[26] kawabe@cs.tsukuba.ac.jp e-mail del Dr. Tohru Kawabe profesor en la Universidad de Tsukuba (Japón),