ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

TESIS DE GRADO

"SIMULACION DIGITAL DE SISTEMAS DE CONTROL"

TESIS PREVIA LA OBTENCION DEL TITULO DE

INGENIERO EN ELECTRONICA Y CONTROL

FANNY YOLANDA MALDONADO BARRIONUEVO

JULIO, 1993

Dedicatoria

A mis padres quienes han sabido apoyarme en forma permanente.

Certifico que el presente trabajo ha sido elaborado en su totalidad por la Señorita Fanny Yolanda Maldonado Barrionuevo.

Ing. Patricio Burbano
Director

CONTENIDO

3. FUNDAMENTOS TEORICOS

4. RESULTADOS Y CONCLUSIONES

1. INTRODUCCION

1.1. INTRODUCCION GENERAL

1.2. SIMULACION

1.3. TIEMPO REAL

1.1 INTRODUCCION GENERAL.

Uno de los problemas más complejos con los que se encuentra el ingeniero de control es la dificultad de analizar y/o diseñar controles de sistemas de alto orden y que tienen un gran número de variables.

La dificultad radica en la inconveniencia de trabajar directamente sobre la planta real, por el riesgo que implica las pruebas sobre la operación del sistema y por el costo de dichas pruebas; por ejemplo, el diseño y construcción de diferentes controles alternativos.

En vez de trabajar sobre la planta, por las razones indicadas, se puede obtener un modelo de dicha planta y realizar las pruebas correspondientes sobre el modelo. El modelo puede ser físico o matemático. En cualquier caso en el modelo ha de considerarse las características más importantes del sistema real.

El modelo físico consiste de un prototipo de laboratorio a pequeña escala de la planta real. Entonces el análisis y diseño se puede realizar sobre el prototipo.

El modelo analítico es una representación matemática de las características más relevantes del sistema real.

El modelo matemático o analítico permite realizar simulación de la planta a través de un computador análogo o

componentes cuanto por su comportamiento. Esto limita su aplicación a problemas de bajo orden, univariables y lineales.

- La respuesta que se obtiene es únicamente en el tiempo, sin poderse obtener otro tipo, como es la respuesta de frecuencia, lugar geométrico de las raíces, etc..

- El diseño en sí se lo hace en forma manual.

Queda como alternativa la simulación digital. A continuación se hace una exposición de las principales características que presenta dicha simulación:

- Se puede realizar análisis sobre modelos de sistemas con mucho menos restricción, pues se puede procesar modelos de alto orden, no lineales, multivariables, etc., e inclusive se pueden incluir restricciones a las señales.

- Se puede realizar el diseño de controles con gran versatilidad, pues los algoritmos de diseño se pueden implementar en el computador en un esquema fuera de línea, off line, pues el computador digital se utiliza como herramienta de diseño, lo cual no es posible con el computador análogo.

- Hace posible estudiar y experimentar sobre el modelo las

complejas interacciones que ocurren en el interior de los sistemas.

- La observación detallada del sistema que se está simulando conduce a un mejor entendimiento del mismo y proporciona sugerencias para mejorarlo, que de otro modo no podría realizarse.

- La simulación digital puede utilizarse como recurso pedagógico, para la enseñanza de la teoría y la práctica de los sistemas de control, incluyendo análisis estadístico, toma de decisiones, etc..

- Una gran ventaja es su bajo costo y fácil acceso.

- La experiencia que se adquiere al realizar diseños sobre un modelo de simulación en un computador, puede ser más valiosa que la simulación en sí misma. El conocimiento que se obtiene al aplicar un diseño mediante simulación, sugiere frecuentemente cambios en el sistema de estudio. Los efectos de dichos cambios pueden probarse, a través de la simulación, antes de implementarlos en tiempo real.

- Puede implementarse para experimentar con situaciones nuevas de las cuales se tiene muy poca o ninguna información con el objeto de entrenarse en la espera de eventos imprevistos.

modelo 60 con monitor a color y coprocesador matemático. Para la implementación en tiempo real se utilizará el sistema de adquisición de datos KEITHLEY 500A. Ambos dispositivos existen en el Laboratorio de Sistemas de Control.

A continuación se hace un breve resumen del contenido del presente trabajo de tesis.

En el capítulo I se dará una descripción general sobre simulación y tiempo real. En simulación se trabajará con paquetes específicos como son: PC-MATLAB, CAD CONTROL, TUTSIM, DYNAMO, KUO; y el paquete QUICK 500, para manejar la unidad de adquisición y salida de datos en el medio ambiente del paquete QUICK BASIC.

En el segundo capítulo se realizará una aplicación de los paquetes más importantes (CAD CONTROL, PC-MATLAB) para el análisis: conversión de modelos, discretización, respuesta en el tiempo, respuesta de frecuencia, lugar geométrico de las raíces, manejo de matrices, uso de las variables de estado, observabilidad, controlabilidad, etc.. Luego se tratará el aspecto de diseño de los controles, tales como redes de compensación y control proporcional integral derivativo (PID). Como complemento de este capítulo se incluye en el apéndice un manual resumido para el manejo de la estación de adquisición y salida de datos y del manejo de los paquetes antes mencionados.

En el tercer capítulo, se dará un aporte con casos de estudio, que tienen que ver básicamente con:

- Instrumentación en tiempo real, esto involucra la utilización de la estación de adquisición y salida de datos, y de software Quick 500, para el manejo de dicha estación.

- Modelación de diferentes circuitos pasivos, activos, incluyendo el método de identificación, mediante la aplicación de paquetes de propósito específico, para el análisis de dichos modelos, incluyendo aspectos estocásticos como la presencia de ruido.

- El control de sistemas tanto en simulación, como en tiempo real.

En el cuarto capítulo se presentan los resultados de la simulación y resultados en tiempo real tanto de los macros, como de la utilización de los paquetes, cuanto de los casos de estudio.

1.2 SIMULACION

En si la simulación implica un proceso para obtener un equivalente computacional del modelo; mucho más fácil de estudiar.

La simulación ayuda a encontrar una respuesta al diseño de control siempre que se tenga un modelo, es decir la simulación digital consiste en crear un modelo abstracto de estructura similar, pero más simple que represente el sistema real y se lo debe expresar en forma que el computador, o más bien el programa a ser utilizado lo entienda.

El objetivo de conseguir un modelo es determinar uno o más aspectos del sistema o inclusive la totalidad del mismo, o sea, sirve de elemento de información, pudiendo obtenerse varios modelos para un mismo sistema, que representan diferentes aspectos del mismo.

Un modelo se lo puede obtener por modelación o por identificación de sistemas. En modelación se parte de leyes físicas, fundamentos de ingeniería para obtener una representación analítica o matemática que represente las características más relevantes del sistema.

Se entiende por identificación, la determinación de un modelo analítico en base a la información de entrada y salida de un sistema, dentro de una clase o estructura específica, tal que bajo ciertas pruebas es equivalente al sistema.

Mientras que la simulación es el establecimiento de una analogía computacional o equivalente computacional de un sistema físico, mediante la utilización o aplicación del

modelo de dicho sistema.

Los modelos a tratarse pueden ser continuos y discretos. Los sistemas en modo continuo pueden ser modelados mediante los siguientes métodos:

- Ecuaciones diferenciales.- Estas ecuaciones son descritas para sistemas continuos. Si son invariantes en el tiempo las ecuaciones diferenciales son de coeficientes constantes.

Se caracteriza porque relaciona la salida y sus derivadas, y(t), y'(t),....; a la entrada y sus derivadas, r(t), r'(t),...., de la forma:

$$y^{(n)}+a_1y^{(n-1)}+\ldots+a_ny=b_0r^{(m)}+b_1r^{(m-1)}+\ldots+b_mr$$

- Variables de estado.- La representación en el espacio de estado es:

$$x' = A\,x + B\,u$$

$$y = C\,x + D\,u$$

donde:

x, u, y son los vectores de estado, de entrada y salida respectivamente de dimensiones n, r, m. A, B, C, D son matrices constantes, para sistemas lineales e invariantes en el tiempo.

- Función de transferencia.- Relaciona la salida a la entrada en el dominio de la frecuencia compleja "s" bajo condiciones iniciales nulas, es de la forma:

$$\frac{Y(s)}{R(s)} = G(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0}$$

Al necesitar un computador para el análisis de sistemas, y pudiendo utilizar dicho computador en esquema en línea hace que sea necesario introducir el concepto de modelos discretos, los cuales pueden ser descritos mediante valores en ciertos instantes de tiempo kT; k = 0, 1, 2, ... Por facilidad de notación se omite T. Se tiene entonces los siguientes métodos de descripción; para modelos discretos:

- Ecuaciones de diferencia.- Estas ecuaciones relacionan valores anteriores de entrada y salida de la forma:

$$y(k-1) + a_1 y(k-2) + \ldots + a_n y(k-n) = b_1 u(k-1) + \ldots + b_n u(k-n)$$

- Variables de estado.- Para sistemas discretos, se tiene su descripción de la forma:

$$x(k+1) = A\, x(k) + B\, u(k)$$

$$y(k) = C\, x(k) + D\, u(k)$$

donde:

x(k), u(k), y(k) son valores a instantes discretos kT; A, B, C, son matrices de coeficientes constantes. A, B diferentes que para el caso de sistemas continuos, pues se obtienen discretizando las ecuaciones de estado de sistemas continuos.

- Función de transferencia.- De igual manera que en sistemas continuos, se puede describir a un sistema discreto mediante función de transferencia en dominio z, o función de transferencia discreta.

$$\frac{Y(z)}{R(z)} = G(z) = \frac{b_m z^m + b_{m-1} z^{m-1} + \ldots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \ldots + a_1 z + a_0}$$

donde se considera que se antepone un cero order hold (ZOH) o dispositivo de retención de orden cero para la discretización de la planta continua.

La simulación digital se reduce al uso de los paquetes de software, para procesar el modelo. Por lo que se tratará de utilizar la potencialidad que ofrecen los paquetes de uso general tanto para el análisis como para el diseño.

De acuerdo con lo señalado anteriormente, la simulación podrá hacerse para sistemas continuos o discretos, mediante ecuaciones diferenciales o ecuaciones de diferencia a variables de estado y mediante función de transferencia.

1.3 TIEMPO REAL.

Al hablar de tiempo real, se está refiriendo a la adquisición de datos (mediciones) de un sistema y a su procesamiento, en el mismo intervalo de tiempo en el que evoluciona dicho sistema.

Desde el punto de vista de control, esos datos o mediciones que se adquieren describen la dinámica del sistema, de ahí que es posible realizar modelación (identificación) y control mediante muestreo de datos a intervalos regulares dados por el período de muestreo. En este caso se tiene un sistema de control discreto y todas las tareas de adquisición de datos, cálculo de la ley de control (procesamiento) y salida del control al sistema (salida de datos) deberán ejecutarse dentro de dicho periodo de muestreo.

Así el control de un proceso en tiempo real se lo define en base a datos de entrada (adquisición), con los cuales se

realiza algún procesamiento con un algoritmo que va a generar señales de salida (control) dentro de un intervalo de tiempo determinado.

Una tarjeta de adquisición o salida de datos por sí sola no constituye una unidad de adquisición de datos y control, vendría a ser solo parte de un hardware; pues se necesita de un computador con un software apropiado para procesar y ejecutar las tareas necesarias, convirtiéndose de esta manera en una poderosa unidad de adquisición, procesamiento de datos y control en tiempo real.

Desde la aparición del computador digital como herramienta de ingeniería, se comenzó a utilizarlo como parte del control en una planta o proceso, en que se desea controlar variables de interés, como puede ser temperatura, presión, etc., con el objetivo de mantenerlas en ciertos valores denominados puntos de referencia, sobre todo en sistemas complejos, naciendo así el concepto de control de sistemas en tiempo real, ó control digital de procesos.

El control digital puede ser un control digital directo o en línea, (ON LINE), esto es, la utilización del computador como elemento activo del lazo de control; y, puede ser fuera de línea, (OFF LINE), donde se lo considera como un elemento remoto de control, como una herramienta de diseño, utilizado para simulación.

Hoy en día se tiende al control digital directo, incluyéndose al computador dentro del lazo de control, lo que ayuda a llevar un control y una instrumentación en línea. Como se aprecia en la figura 1.1. Cuando el computador está conectado en línea dentro del lazo, es necesaria la presencia de una interfaz análoga/digital y digital/análoga, que constituye el sistema de adquisición de datos y salida de datos, el que incluye una manipulación de períodos de muestreo T. De aquí nace la necesidad de trabajar con modelos discretos, pues hay que procesar la secuencia discreta del error e(kT) para obtener la secuencia discreta del control u(kT).

La modelación, también se puede realizar mediante muestreo de datos, y procesándoles mediante algún algoritmo computacional, método que como se ha indicado consiste en la identificación de sistemas, todo esto en tiempo real.

El computador constituye la unidad que procesa la información adquirida para obtener la salida (ley de control), realiza el monitoreo de señales, la identificación, etc., en base al esquema de datos muestreados en la figura 1.1

donde:

r(t) : referencia

e(t) : error

capacidad de memoria y bajo costo. A esto se debe sumar el hecho de que existe el software en el mercado para este tipo de aplicaciones, obteniéndose, un sistema con rapidez, precisión y flexibilidad. La flexibilidad se debe al aspecto del software, pues el mismo algoritmo de control se puede aplicar a diferentes plantas cambiando ciertos parámetros, ya que ahora el control se hace por software y no por hardware que es difícil de modificar. Con éste método la instrumentación es más eficiente, pudiendo considerarla como inteligente, pues da información con más detalle, incluyendo señalización, monitoreo, etc..

En esta tesis para simulación se utilizará el programa Quick Basic. Una vez realizado el software correspondiente a simulación, se aplicarán técnicas a control en tiempo real, con la estación de adquisición de datos y control KEITHLEY 500A y la ayuda del software Quick 500 que permite el enlace entre el equipo mencionado y el computador.

La ejecución de áreas en tiempo real puede hacerse en dos esquemas diferentes de trabajo: foreground o trabajo dedicado y background, o trabajo a fondo, tanto para un canal de entrada y/o salida (unicanal), como para sistemas de varias entradas y/o salidas (multicanal).

Así en cada interrupción el procesador salta desde FOREGROUND, donde ejecuta la tarea principal, a ejecutar

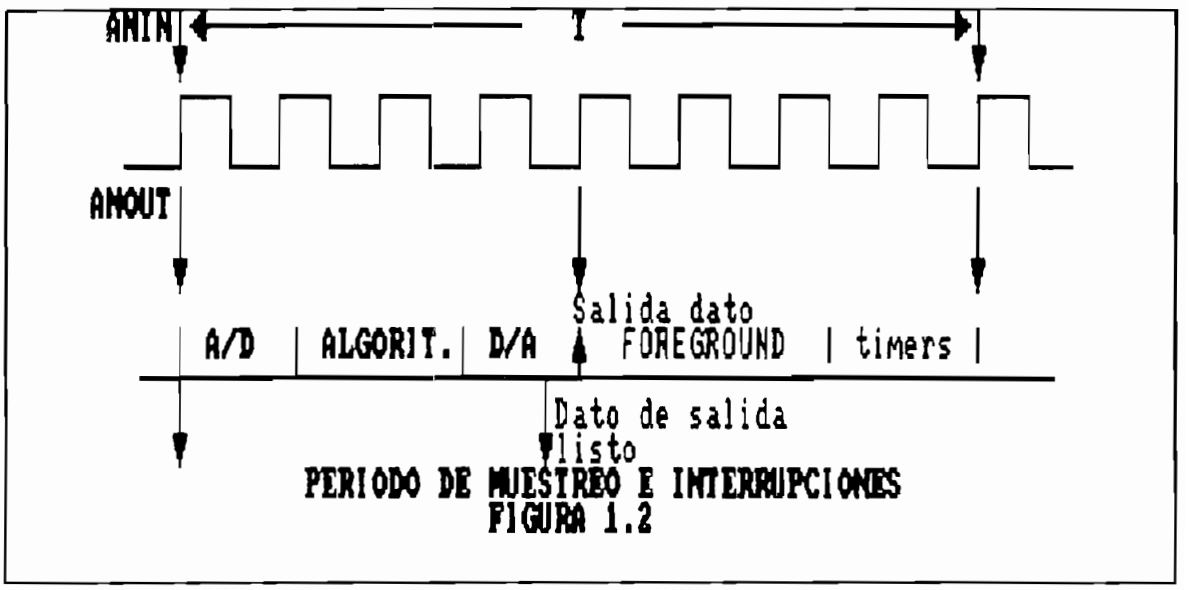
tareas de BACKGROUND, donde se realizan las tareas de fondo. Cuando estas se completan, se regresa al modo foreground, hasta la siguiente interrupción.

El sistema de adquisición y salida de datos utiliza interrupciones no mascarables (NMI), por lo que tiene la mayor prioridad. Los comandos que no son de background y los comandos ordinarios de BASIC se llaman comandos de foreground, y son ejecutados durante el tiempo en que no se atiende las interrupciones. Esta operación conjunta de los dos modos de trabajo se denomina multitarea, y si bien no se ejecutan simultáneamente, el computador lo que hace en realidad es cambiar la atención de ambas tareas.

Tanto foreground como background pueden actuar independientemente, donde además la programación de ambos es muy flexible. Mientras en background se ejecutan secuencias de adquisición de datos, el foreground se puede utilizar para monitorear estatus de las tareas de background, comunicación de periféricos, usuarios, análisis, etc.. En la figura 1.2 se esquematiza lo expuesto.

- bintv% es un período de muestreo de datos
- boutv% es un período de salida de datos

Para un bint% = 8; boutv% = 4



PERIODO DE MUESTREO E INTERRUPCIONES
FIGURA 1.2

Es importante aclarar que el período T de muestreo depende de las interrupciones. La interrupción tiene su propio período, mientras que el período de muestreo se obtiene en base al de las interrupciones, igual para el período de salida.

El período de muestreo que se impone, suele ser muy superior al tiempo que se demora entre la adquisición del dato hasta la obtención del resultado que debe ser devuelto a la planta, de ahí que mientras menor sea el retardo, menor será el error cometido. Es conveniente que en cada nuevo muestreo de entrada (siempre mayor o igual que al período de salida), se repita un ciclo, lo cual trae la necesidad de sincronizar las tareas, lo que se consigue teniendo multiplicidad exacta entre los períodos de muestreo de

entrada y el período de salida, tal como se ve en la figura 1.3

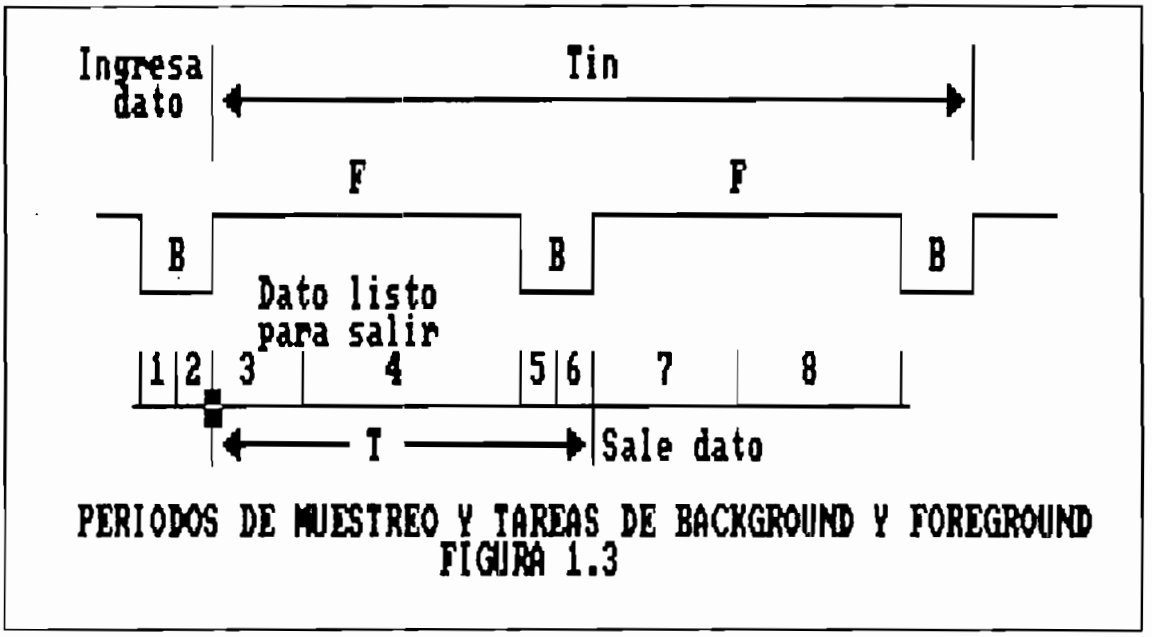
1,2 Adquisición del dato (A/D)

3,4 Algoritmo de control (control)

5,6 Salida del dato (D/A)

7 Tareas de foreground

8 Actividad de los timers



PERIODOS DE MUESTREO Y TAREAS DE BACKGROUND Y FOREGROUND
FIGURA 1.3

donde:

Tin es el período de muestreo de entrada deseado

T es el período de salida mínimo posible.

Se debe hacer un estimado del tiempo mínimo que se requiere para obtener el dato de salida (adquisición + conversión A/D + algoritmo), lo cual a su vez ayuda a saber cual es el tiempo mínimo de muestreo de entrada.

Como se ve en la figura 1.3, todo el tiempo que sobra desde el final de las tareas de foreground, hasta la nueva interrupción es llenado con la actividad de los timers (del Quick 500, que fijan el periodo de muestreo), con lo cual se asegura que todas las actividades de foreground se realicen una vez por cada dato adquirido, es decir también se sincroniza de esta manera las tareas de foreground y background.

En cuanto se refiere a la estación de adquisición de datos y control Keithley 500A, se puede mencionar, que el hardware y software con que cuenta, permite realizar la interfaz entre la planta y el computador, desde sus requerimientos y configuración, hasta su instalación.

Esta estación está construida para trabajar en un computador IMB PC, PS/2, 80286 y computadores 100% compatibles con estos, debiendo tener por lo menos un slot de expansión de longitud media, o al menos que tenga un microcanal completamente compatible con un slot de expansión.

El equipo existente, cuenta con tres módulos para

adquisición de datos análogos, salida de datos análogos y salida de datos digitales.

Los módulos mencionados son:

- AMM1: Módulo maestro de medición análoga. Es una tarjeta de 12 bits, 32 KHz para realizar medidas análogas. Este módulo combina dos funciones importantes en un solo módulo, esto es el acondicionamiento y multiplexado de señales análogas y conversión análoga digital.

Posee ocho canales de entrada, con ganancia local unitaria, las señales son aplicables a través de una bornera de tornillos y la ganancia global es manejada por software, pudiendo obtenerse ganancias x1, x2, x5 y x10.

El módulo AMM1 utiliza un conversor de 12 bits de aproximaciones sucesivas que aseguran velocidad, exactitud en las mediciones y conversión.

De los canales análogos puede obtenerse voltajes normalizados entre -10 v. y +10 v., y están predefinidos como ANLG0, ANLG1, ANLG2, ANLG3, ANLG4, ANLG5, ANLG6, ANLG7.

- AOM1: Módulo salida análoga. Ofrece una resolución de 12

bits. Tiene cinco canales de salida análoga. Tiene un conversor digital análogo, independientemente, con salida seleccionables mediante dip switches.

La conexión de la señal es hecha a través de un bornera de tornillos.

Tiene la característica de soportar dos niveles de congelamiento de datos en el conversor D/A y permite sincronizar la actualización de todos los canales análogos de salida.

Estos canales son capaces de entregar voltajes normalizados de -10 v. a +10 v. Han sido predefinidos como ANOUT0, ANOUT1, ANOUT2, ANOUT3, ANOUT4, mediante el archivo CONFIG.TBL hecho con el programa CONFIG.EXE.

El software del equipo, lo constituye el paquete Quick 500, que está orientado al entendimiento de las necesidades y requerimientos para la adquisición de datos y salida de datos de control. Está escrito para ser utilizado con la estación Keithley 500A. Ha sido desarrollado para combinar la velocidad y la programación avanzada de un lenguaje estructurado como es el Quick Basic.

Los requerimientos de software y hardawre del Quick 500 son:

- Sistema operativo DOS, versión 3.0 o mayor
- Compilador Quick Basic, versióm 4.0 o mayor
- Computador IBM PS/2 PC, 80286, o compatibles 100%
- El programa INSTALL.EXE que inicializa la cantidad de memoria ROM y RAM, verifica el hardware, y crea archivos .BAT, como el SOFT500.BAT
- Necesita 640 Kb de memoria RAM
- 1 Mb de disco duro
- Para tener ventajas gráficas deben existir tarjetas adaptadoras de gráficos como IBM CGA, Hércules COLOR, IBM EGA, Hércules monocromático, etc.
- El coprocesador matemático es un accesorio adicional.

2. ESTUDIO DE LOS PAQUETES

2.1. USO DE LOS PAQUETES

2.2. APLICACION DE LOS PAQUETES PARA ANALISIS

2.2.1. ANALISIS MEDIANTE CAD CONTROL

2.2.2. ANALISIS MEDIANTES PC-MATLAB

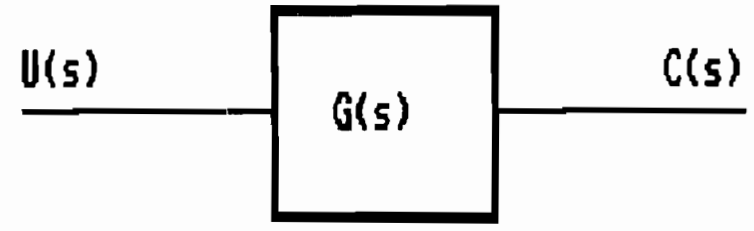
2.3. APLICACION DE LOS PAQUETES PARA DISEÑO

2.1 USO DE LOS PAQUETES.

Ver anexo A

2.2 APLICACION DE LOS PAQUETES PARA ANALISIS

Se parte de un mismo ejercicio para el análisis, mediante simulación, utilizando los paquetes CAD CONTROL y PC-MATLAB, pues estos permiten a más de obtener su respuesta en el tiempo utilizar el lugar geométrico de las raíces, respuesta de frecuencia, conversión de modelos, discretización, etc., con una muy buena resolución gráfica y ayudas. El ejercicio a utilizarse corresponde al modelo de un sistema, como el que indica en la figura 2.1



ESQUEMA DE UN SISTEMA EN LAZO ABIERTO

FIGURA 2.1

donde:

$$G(s) = \frac{35}{s^3+7s^2+14s+8}$$

2.2.1 ANALISIS MEDIANTE CAD CONTROL:

Para proceder al análisis es necesario primero el ingreso de la planta, ó función de transferencia, como ya se dijo la función de transferencia a ser analizada es:

Lazo abierto:

$$G(s) = \frac{35}{s^3+7s^2+14s+8}$$

Se procede al ingreso de la función:

CC>ENTER

Ingrese la función de transferencia Gn>g1

Ingrese el número de polinomios en el numerador>1

Ingrese el polinomio #1> 0,35

Ingrese el número de polinomios del denominador> 1

Ingrese el polinomio #1> 3,1,7,14,8

- Análisis en el tiempo:

Para cumplir este objetivo se utiliza el comando:

CC>TIME

A continuación viene como pregunta inmediata la función de transferencia que se desea ingresar:

Gi = Función de transferencia> g1

luego se ingresa el tipo, para lo cual despliega un menú de opciones, una vez escogido se digita el número de la opción deseada:

1 Respuesta en lazo cerrado con entrada de una función paso
2 Respuesta en lazo cerrado con entrada de una función impulso
3 Respuesta en lazo abierto con entrada de una función paso
4 Respuesta en lazo abierto con entrada de una función impulso
5 Respuesta en lazo abierto no causal con entrada de una impulso

Siendo la opción de interés para este caso el número 1.

Seguidamente van apareciendo una serie de condiciones para conseguir el gráfico de interés, las mismas que deberán ser contestadas para poder continuar, así para este caso en

particular se escogió:

Ingreso automático de los parámetros (Y ó N) Y

al digitar la letra Y que quiere decir (SI), es decir se quiere decir con esto que el programa se encargará de escoger las escalas correspondientes tanto para el eje x (tiempo), como para el eje y (salida del sistema).

Una vez desplegado el gráfico, en su parte inferior muestra una lista de opciones, para escoger cualquiera de las opciones es necesario digitar la letra que se desea, entre las principales se tiene:

E Opción de cambio de límites, pudiendo escogerse:

1 Ymin 2 Ymax 3 # divs

4 Tmin 5 Tmax 6 # divs 7 # ptos

para seleccionar el limite que se desea cambiar se digita el número.

P Opción que permite graficar nuevamente, es decir, por ejemplo una vez cambiado de límites es necesario que vuelva a graficar con los nuevos parámetros.

C Cursor, esta opción permite presentar en pantalla un cursor que se desplaza a lo largo de todo el gráfico, indicando en la parte inferior izquierda de la pantalla las variaciones del tiempo y la respuesta del sistema.

H Hardcopy, permite obtener el gráfico en papel a través de la impresora.

La respuesta obtenida se presenta en la figura 2.2

Del gráfico se tienen las características de respuesta (utilizando la opción cursor):

Valor final = .81, luego Ep = 19%

Máximo sobre impulso = Mp = 46.29%

Tiempo de establecimiento = ts = 5.8 s.

Cabe aclarar que estas opciones que aparecen en la parte inferior de la pantalla, están presentes en todos los gráficos, ya sean estos de frecuencia, lugar de las raíces, etc., tanto para sistemas continuos como para los discretos.

- Análisis en Frecuencia:

Para cumplir con este objetivo es necesario crear un archivo de datos, con la ayuda del comando **FREQUENCY** para cada una de las funciones de transferencia que se necesite, entonces:

CC>FREQUENCY

de la misma manera se prosigue a contestar cada una de los requerimientos necesarios, así:

Ingrese la función de transferencia= g1

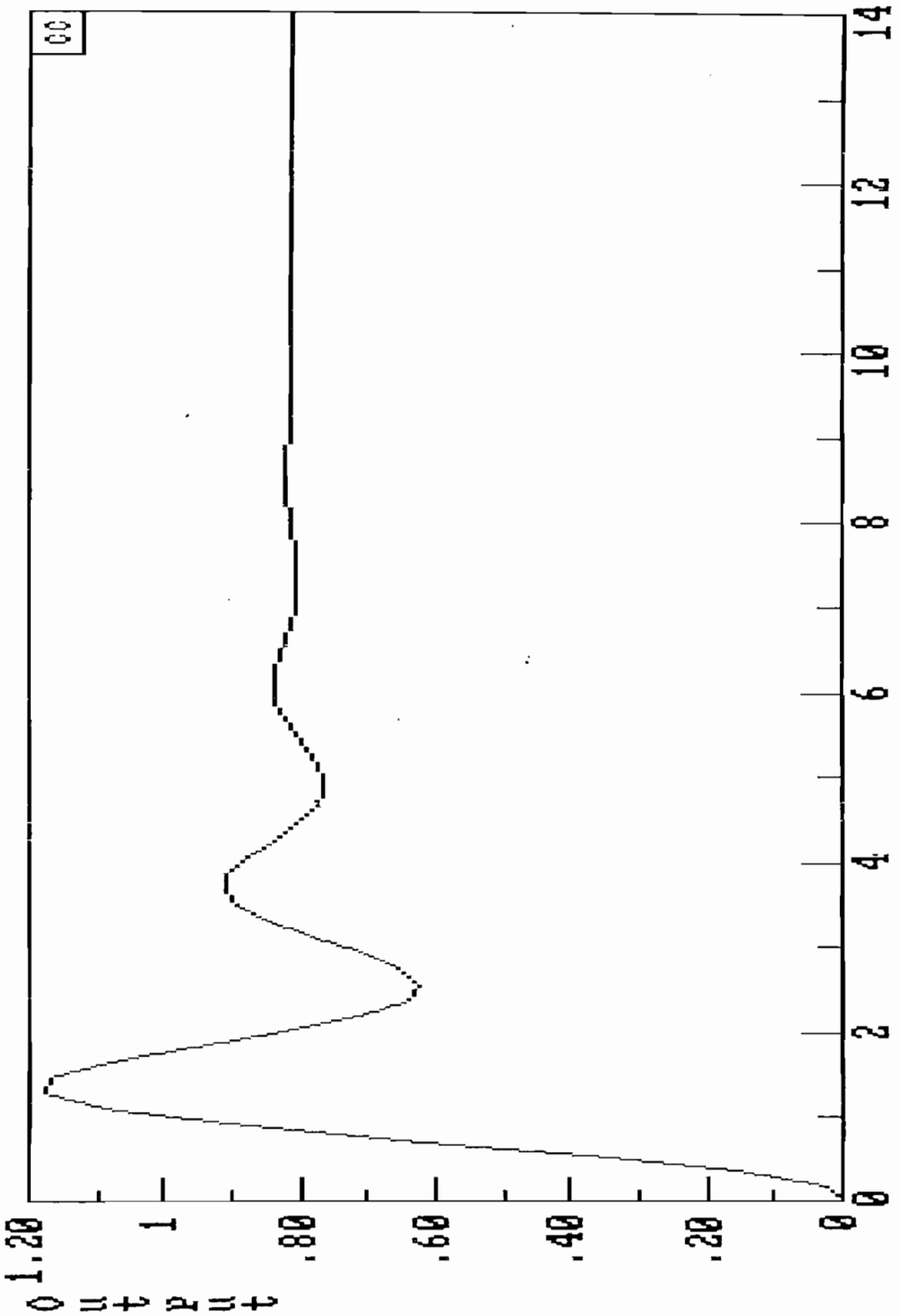
```
Ingrese la frecuencia mínima, máxima, #
puntos>.1,1000,100
Ingrese tipo de escala 0=logaritmica, 1 lineal> 0
```

Una vez creado este archivo se puede proceder a la utilización de cualquiera de los comandos de gráficos, como Bode, Nyquist, etc. El gráfico de interés es Bode.

```
CC>BODE
Parámetros:
      tipo:
1 |Gi(jw)| vs w
2 Arg |Gi(jw)| vs w
3 |Gi(jw) y Arg|Gi(jw)| vs w
4 |1 + Gi(jw)| vs w
5 |Gi(jw)| y |1 + Gi(jw)| vs w
6 |1 + 1/Gi(jw)| vs w
7 |Gi(jw)| y |1 + 1/Gi(jw)| vs w
se escoge el tipo 3
```

Seguidamente aparece la pregunta:

```
Ingreso automático de los parámetros (Y ó N): Y
Ingrese el modo:
background: 0 = líneas(default) 1 = marcas
foreground: 0 = líneas(default) 1 = marcas>b
```



RESPUESTA EN LAZO CERRADO DE LA PLANTA
FIGURA 2.2

Luego de digitar el modo, el programa despliega el gráfico correspondiente, con las mismas opciones indicadas, el mismo se lo puede visualizar en la figura 2.3. De este gráfico se tiene (utilizando la opción cursor):

Margen de ganancia = MG = 8.1 db.

Margen de fase = MF = 35.Ø1°

Ahora se realiza el mismo procedimiento pero ingresando la función en lazo cerrado para realimentación unitaria mediante el comando **BUILD.**

```
CC>BUILD
BUILD>g2=g1/(1+g1*1)
```

Su resultado es tener en la función g2, la función de transferencia de lazo cerrado.

Siguiendo los pasos anteriormente señalados para lazo abierto, pero con la función de lazo cerrado, obtenida mediante el comando build se obtiene la figura 2.4, de la que se mide:

Máximo de resonancia = Mr = 5.18 dB.

Ancho de banda = AB = 3.64 rad/s.

Al análisis en frecuencia se puede añadir el análisis en estabilidad, con el comando:

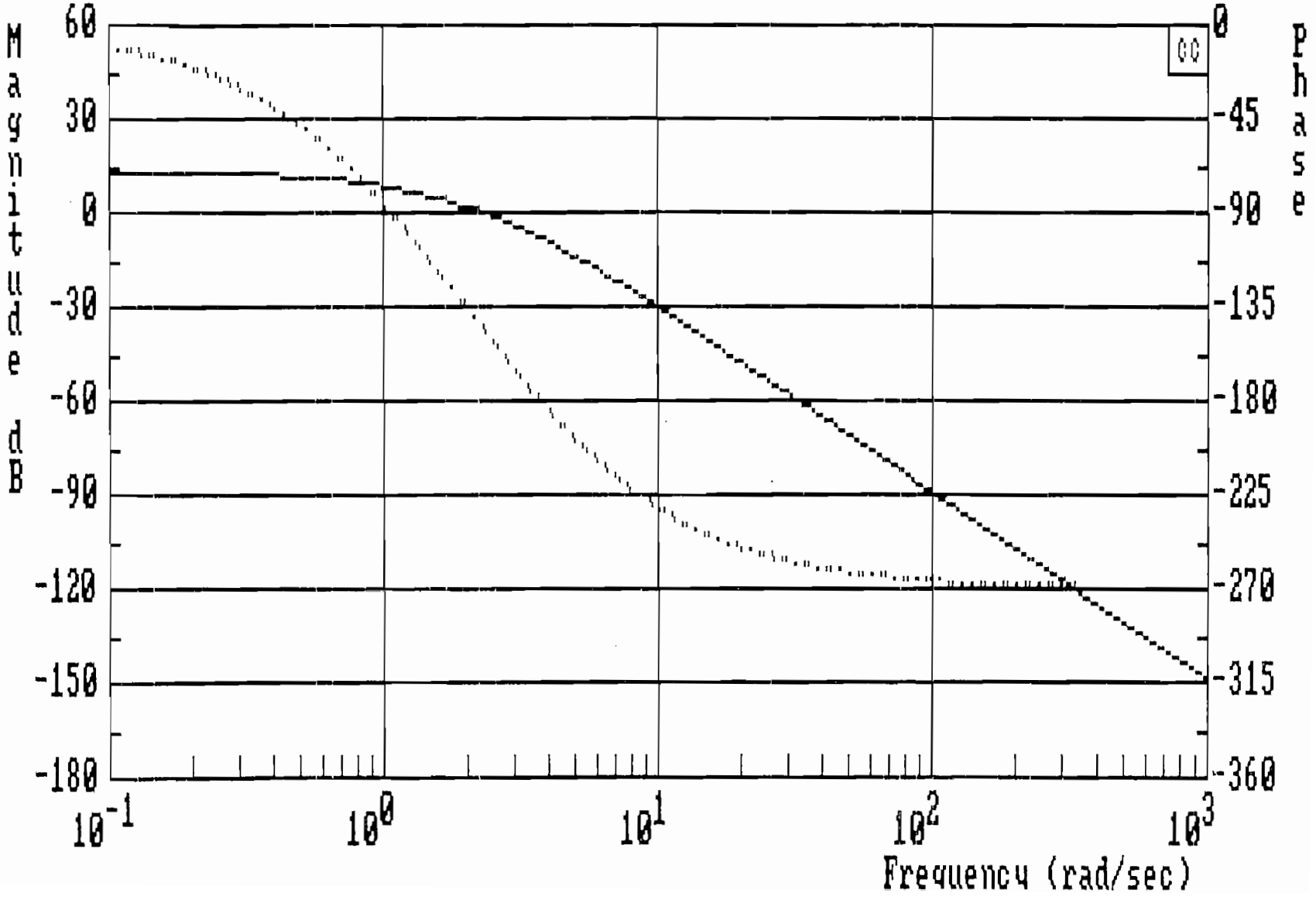


DIAGRAMA DE BODE EN LAZO ABIERTO
FIGURA 2.3

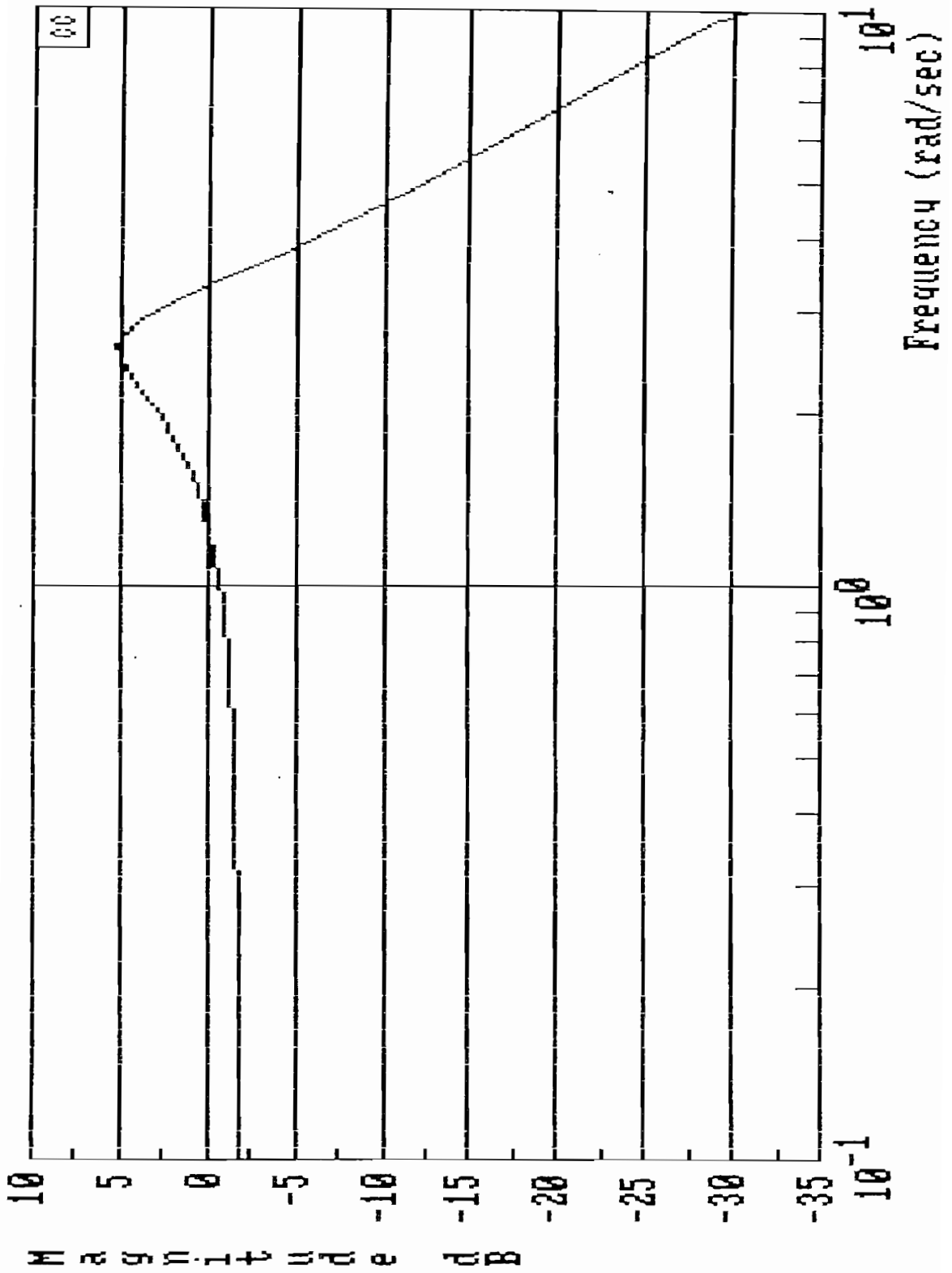


DIAGRAMA DE BODE EN LAZO CERRADO
FIGURA 2.4

CC>STABILITY

Este comando determina la estabilidad de un sistema en lazo cerrado.

Su respuesta es:

Polinomio característico en lazo cerrado:

$p(s) = s^3 + 7s^2 + 14s + 43$

CEROS: -5.86306027890036

-.5684698605099821 +j 2.647809641908754

-.5684698605099821 -j 2.647809641908754

El sistema análogo en lazo cerrado es ESTABLE.

- **Análisis con el lugar geométrico de las raices:**

Se digita:

CC>ROOT LOCUS

se procede a contestar cada uno de los requerimientos:

Ingrese la función de transferencia> g1

Ingrese(Mínimo real, máximo real, #divisiones)

(Mínimo imaginario, máximo imaginario, #divisiones

(AUTO, escala automática)> A

Ingrese el título: LUGAR GEOMETRICO DE LAS RAICES

Ingrese el modo:

background: 0 = líneas(default) 1 = marcas

foreground: 0 = líneas(default) 1 = marcas>b

El gráfico del lugar geométrico se puede apreciar en la

figura 2.5

Se procede al análisis mediante lugar geométrico de las raíces. Para el valor de ganancia de K = 35; los polos en lazo cerrado son:

p1 = -.56 + 2.65*j

p2 = -.56 - 2.65*j

p3 = -5.87

Obteniéndose por consiguiente un sistema en lazo cerrado aproximado a uno de segundo orden pues los polos complejos conjugados deseados son dominantes. La función de transferencia aproximada en lazo cerrado es:

$$F.T.-\frac{5.945}{s^2+1.12s+7.34}$$

Se tiene las siguientes características: Mp% = 51.48% y un ts = 5.6 s., Ep% = 19%.

A manera de complemento se puede citar el comando:

CC>ROUTH

Determina el rango de la ganancia K para que un sistema permanezca estable. En efecto este sistema es estable para el

rango de ganancia: -8 a 90

Como una acotación, el programa permite hacer operaciones (multiplicación, división, suma, resta) de funciones de transferencia, ó se puede hacer también operaciones con una función de transferencia y parámetros ya sean estos constantes ó variables (funciones de "s"). Esto lo permite el comando build, como ya se mecionó.

Como se puede apreciar, los gráficos tienen una resolución bastante buena, dando la información suficiente.

- Análisis en el espacio de estado:

Es necesario ingresar dentro del ambiente de espacio de estado, se logra digitando:

```
CC>STATE
STATE>
```

Dentro de este ambiente se tienen varias opciones, dentro de estas:

```
STATE>penter
Ingrese el espacio de estado del sistema>p1
Ingrese el #estados, #entradas, #salidas>3,1,1
La matriz A es 3 por 3
A, fila 1>0,1,0
```

```
A, fila 2>0,0,1
A, fila 3>-8,14,-7
La matriz B es 3 por 1
B, fila 1>0
B, fila 2>0
B, fila 3>35
La matriz C es 1 por 3
C, fila 1>0,0,1
La matriz D es 1 por 1
D, fila 1>0
```

Para poder observar el resultado de lo ingresado se tiene la ayuda:

```
STATE>pdisplay,p1
despliega
p: #estados = 3   #entradas = 1   #salidas = 1
A
1: 0.000000D+00      1.000000D+00      0.000000D+00
2: 0.000000D+00      0.000000D+00      1.000000D+00
3: -8.00000D+00      -14.0000D+00      -7.00000D+00
B
1: 0.000000D+00
2: 0.000000D+00
3: 35.00000D+00
C
1: 0.000000D+00      0.000000D+00      1.000000D+00
D
```

1: 0.000000D+00

Una vez ingresado el modelo a variables de estado se tiene algunas opciones:

ccf.- se obtiene una forma canónica controlable en P a partir de G(s) ingresada.

dcf.- se obtiene en P una forma canónica diagonal a partir de G(s).

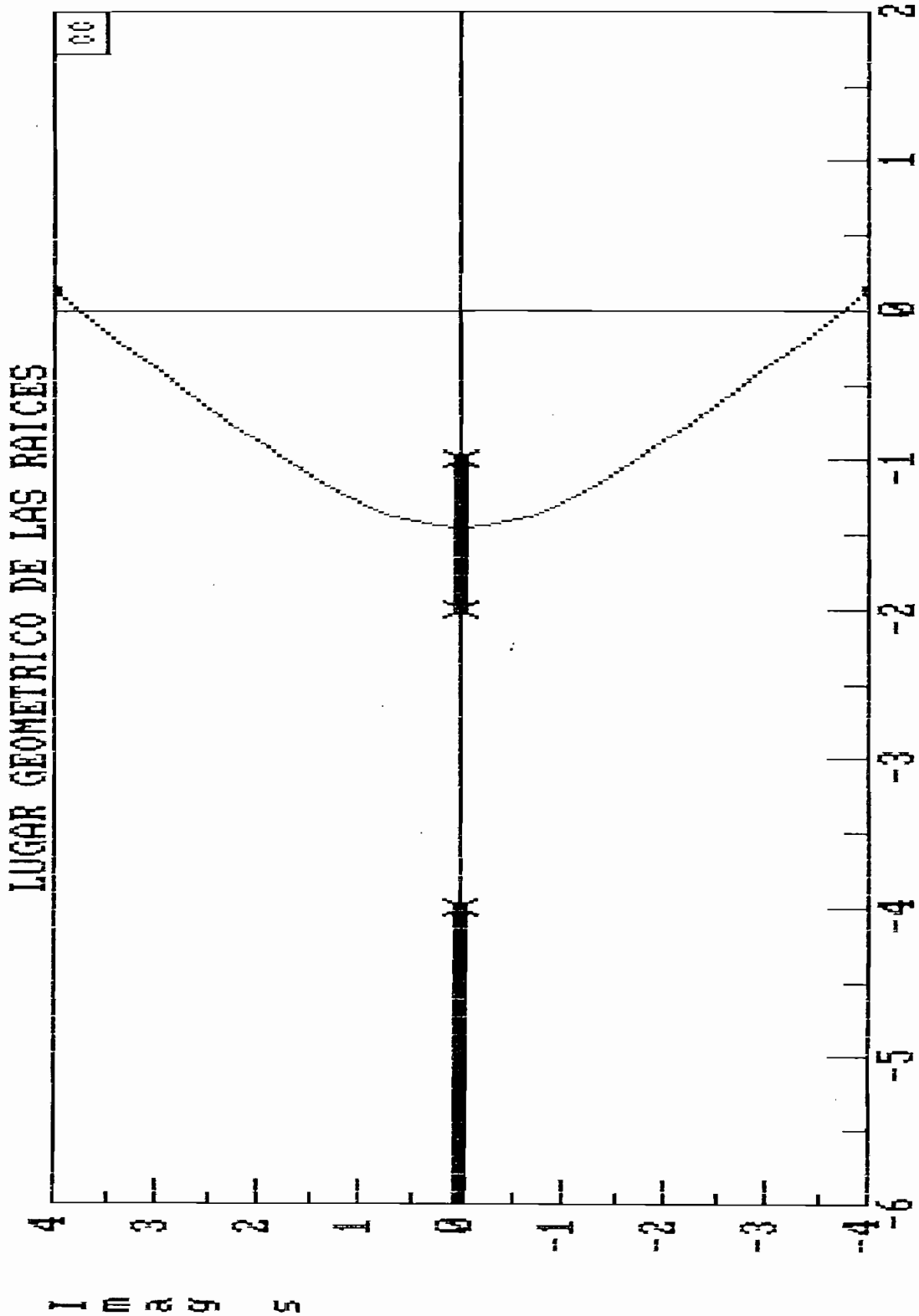
CONTROLLABILITY u OBSERVABILITY chequean observabilidad y controlabilidad.

- Discretización:

El paquete puede obtener el modelo equivalente discreto con el comando:

CC>CONVERT

Convierte del modo análogo a discreto, con requerimientos básicos como son la función de transferencia a ser cambiada G1(s), donde se va a colocar la nueva G2(z), tiempo de muestreo, y la forma de discretización existiendo algunas, sin embargo la forma que interesa es con la equivalencia del "zero orden hold" (ZOH). Para esto se sigue el siguiente procedimiento:



LUGAR GEOMETRICO DE LAS RAICES
FIGURA 2.5

CONVERT>

Ingrese la función de transferencia a ser discretizada > g1

Ingrese la función de transferencia donde se va a almacenar los valores discretos > g2

El período de muestreo > .1

Para discretizar el modelo es necesario un período de muestreo. El período de muestreo debe escogerse menor que la constante de tiempo más significativa, si los polos son reales, o menor que la frecuencia de oscilación si los polos son complejos congujados. Típicamente se escoge entre 5 y 10 veces más pequeño. El período de muestreo a ingresarse está en segundos.

Así, para la función de lazo abierto:

$$G1(s) = \frac{35}{s^3+7s^2+14s+8}$$

discretizando:

$$G2(z) = \frac{4.907479\times10^{-3)}(z^2+3.365227z+0.7047101)}{(z-.67032)(z-.8187308)(z-.9048374)}$$

Para la función en lazo cerrado:

$$G1(s) = \frac{35}{s^3+7s^2+14s+43}$$

discretizando:

$$G2(z) = \frac{4.906084x10^{-3}(z^2+3.366085z+0.705121)}{(z-.5563787)(z^2-1.823629z+.892531)}$$

La utilización de modelos discretos se ilustra en el capítulo III, en los casos de estudio.

2.2.2 ANALISIS MEDIANTE PC-MATLAB:

Como primer punto cabe indicar la gran versatilidad de este paquete, la misma que consiste en la facilidad que presenta la conversión de distintos modelos, pudiendo trabajar en cualquiera de sus modos para la utilización del resto de herramientas.

Para seguir con el esquema trazado, los modelos de conversión que interesa, por el momento son:

- tf2ss convierte sistemas de la forma de función de transferencia a variables de estado:

```
[a,b,c,d]=tf2ss(num,den)
```

como respuesta se tiene que a, b, c, d son las matrices correspondientes a variables de estado, es decir:

x' = ax + bu

y = cx + du

- **ss2tf** convierte sistemas de la forma espacio de estado a función de transferencia:

[num,den]=ss2tf(a,b,c,d,lu)

donde: como ya se dijo, a, b, c, d, son las matrices correspondientes al espacio de estado; num, den son el numerador y denominador respectivamente de la función de transferencia, y lu es la u-ava entrada del sistema.

Con esta aclaración de procede a una breve descripción de las funciones a usarse para el análisis de una función de transferencia, ó a variables de estado, indistintamente.

- **Análisis en el tiempo:**

Se hace necesario el ingreso de la función en lazo cerrado, o la utilización de un macro (archivo .m) para la obtención de la función de transferencia de lazo cerrado a partir de la función de transferencia de lazo abierto, que se explica en el anexo B. Luego con **STEP**, se calcula la respuesta paso de un sistema lineal continuo. Es necesario crear mediante un vector de valores de tiempo, en este caso 0 hasta 15 segundos.

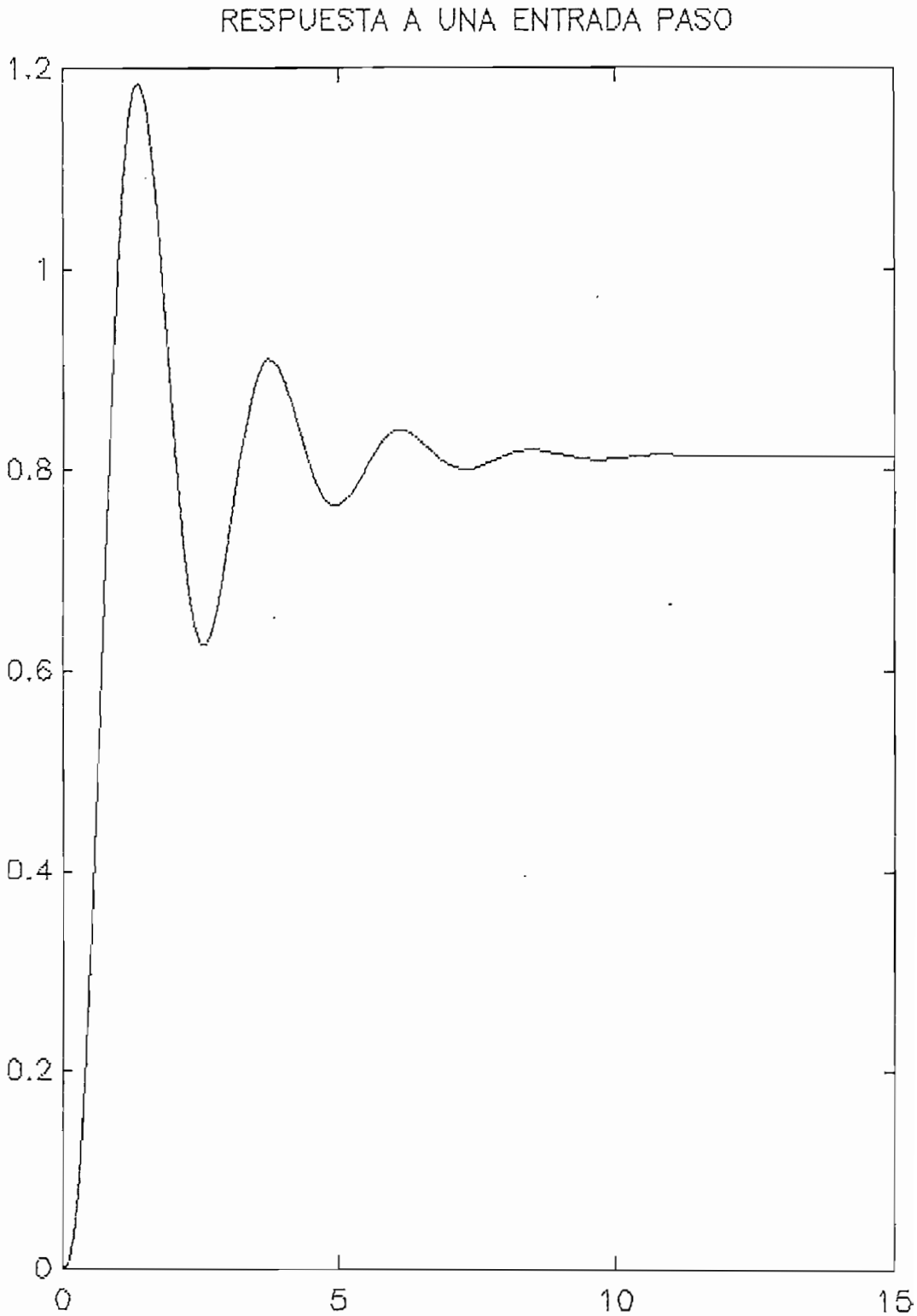
t=0:.1:15

```
num=[0,0,0,35];
den=[1,7,14,8];
function[nc,dc]=lazoc(num,den)
y = step(nc,dc,t)
plot (t,y), title('Respuesta de un sistema a una entrada
paso')
meta grafico
ó
[a,b,c,d]=tf2ss(nc,dc);
t=0:.1:15;
y=step(a,b,c,d,1,t);
plot(t,y),title('Respuesta de un sistema a una entrada
paso');
meta grafico
```

seguidamente se despliega el gráfico, el mismo que se lo puede visualizar en la figura 2.6.

Para proceder a graficar en MATLAB se crea el gráfico, luego se llama al comando **meta** y el nombre del archivo donde se quiera almacenar los datos. Al salir del programa, a nivel de DOS se procede a digitar:

```
GPP nombre del archivo /DEPSD
COPY nombre del archivo.EPS PRN /B
```



RESPUESTA EN LAZO CERRADO DE LA PLANTA
FIGURA 2.6

– Análisis en frecuencia:

El caso que interesa es el diagrama de Bode, lo que se consigue con la función **BODE**, el mismo que calcula la respuesta de frecuencia de sistemas continuos, permitiendo el análisis en frecuencia, con la obtención de margen de fase y margen de ganancia, haciéndose necesario el ingreso de la función de transferencia en lazo abierto:

```
num=35;
den=[1,7,14,8];
```

se hace necesario la creación de una tabla de valores de frecuencia con variación en escala logaritmica w:

```
w=log space[-1:2:50];
```

Seguidamente se crea una tabla completa de magnitud, ángulo

```
[a,b,c,d]=tf2ss(num,den);
[mag,phase]=bode(a,b,c,d,lu,w);
también en este caso lu=1
[mag,phase]=bode(num,den,w);
Para proceder a graficar:
loglog(w,mag),title('RESPUESTA DE LA MAGNITUD')
semilogx(w,phase),title('RESPUESTA DE LA FASE')
```

La respuesta de frecuencia se la puede ver en las figuras 2.7 y figura 2.8.

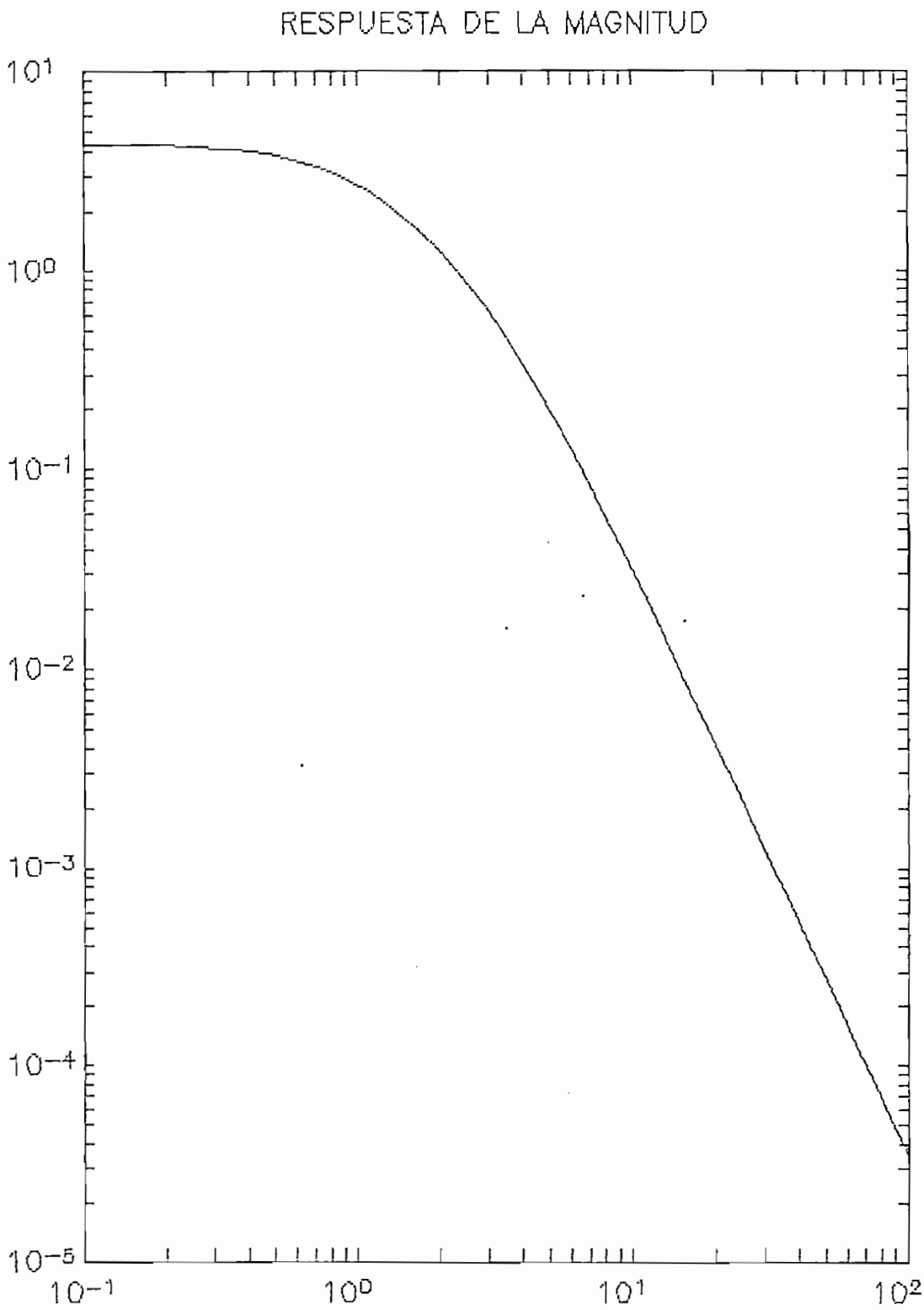


DIAGRAMA DE BODE EN LAZO ABIERTO (MAGNITUD)
FIGURA 2.7

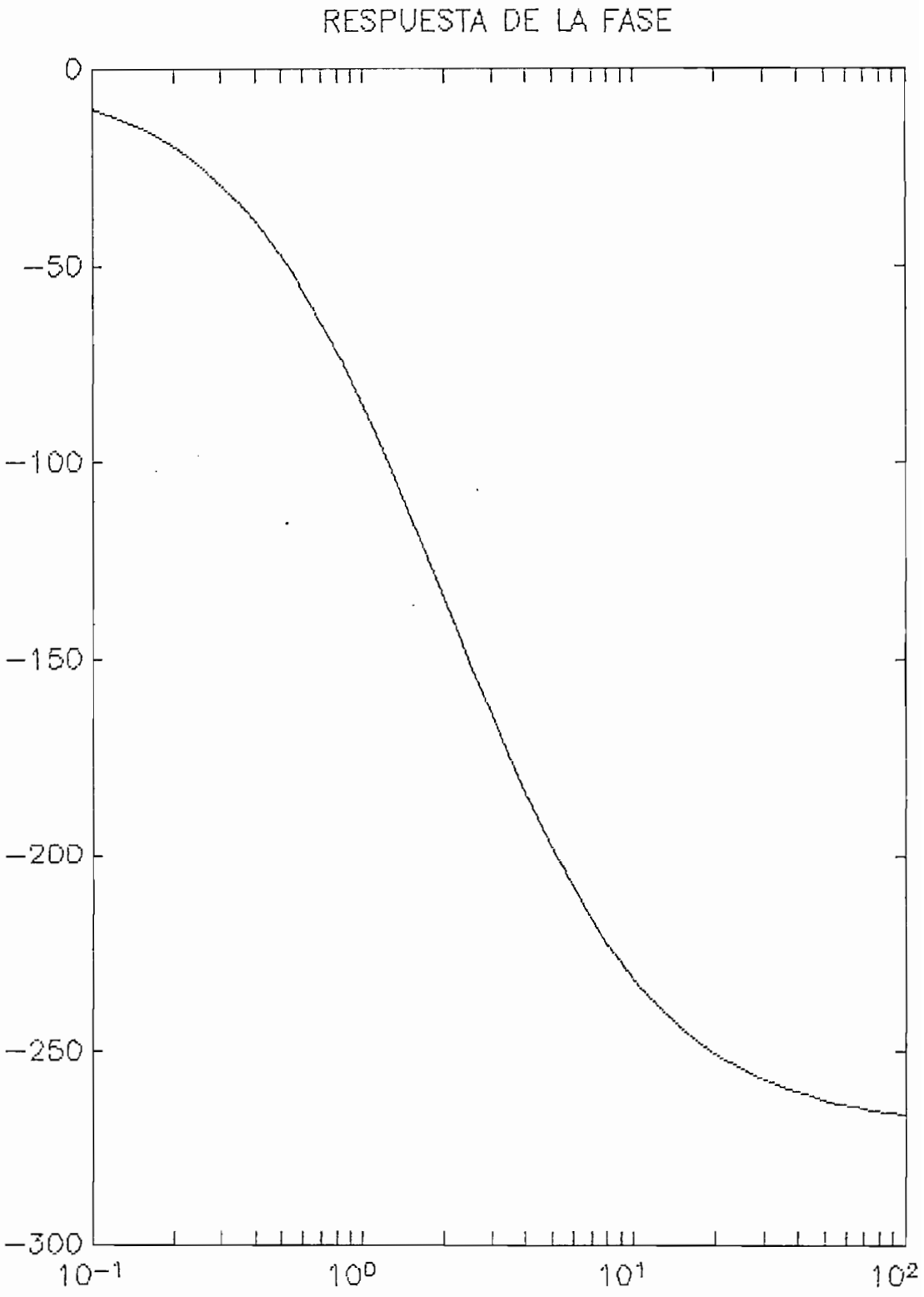


DIAGRAMA DE BODE EN LAZO ABIERTO (FASE)
FIGURA 2.8

Con el comando **margin** se obtiene el margen de ganancia y el margen de fase, con sus respectivas frecuencias:

```
w = logspace(-1,2)
[mag,phase] = bode(a,b,c,d,1,w)
[Gm,Pm,Wcg,Wcp) = margin(mag, phase, w)
```

dando como resultado:

Gm = 2.555 => 8.1 db.

Pm = 34.8407°

Wcg = 3.75 rad/s

Wcp = 2.29 rad/s

Siguiendo el mismo procedimiento, pero ahora para lazo cerrado se obtuvo la respuesta de magnitud que se puede ver en las figuras 2.9, utilizando el comando BODE.

- Análisis mediante lugar geométrico de las raíces:

RLOCUS, calcula el lugar de las raíces de Evans para sistemas lineales, el cual es usado para el estudio de los efectos de la variación de la ganancia en un sistema realimentado.

Para este caso se necesita crear una tabla de valores de ganancia "k":

```
k= logspace(-7,.5,1000);
```

seguidamente se digita, asumiendo que se ha ingresado

la función de transferencia en lazo abierto:

```
r=rlocus(num,den,k)
plot(r),title('L.G.R.'),xlabel('Real'),ylabel('Imag')
```

desplegando el gráfico que se observa en la figura 2.10

Se debe aclarar que la resolución de los gráficos anteriores, como se puede apreciar, es bastante buena, a excepción del lugar geométrico de las raíces. Para compensar este defecto una vez obtenida de una forma general, el lugar geométrico se debe poner énfasis en los lugares donde se necesita mayor exactitud, para lo cual se procede a cambiar el rango de k (tabla), para luego volver a graficar.

Como una ligera acotación, este programa permite un análisis de controlabilidad y observabilidad digitando: **CTRB** ó **OBSV**. Su nomenclatura es:

```
co=ctrb(a,b)
ob=obsv(a,c)
```

dando como resultado la característica de observabilidad y controlabilidad del sistema que está siendo estudiado.

Cabe anotar que este programa si distingue las letras mayúsculas de las minúsculas, por lo que para el ingreso de todas sus funciones se lo hará con letras minúsculas; así mismo no permite espacios en blanco en la línea de comando.

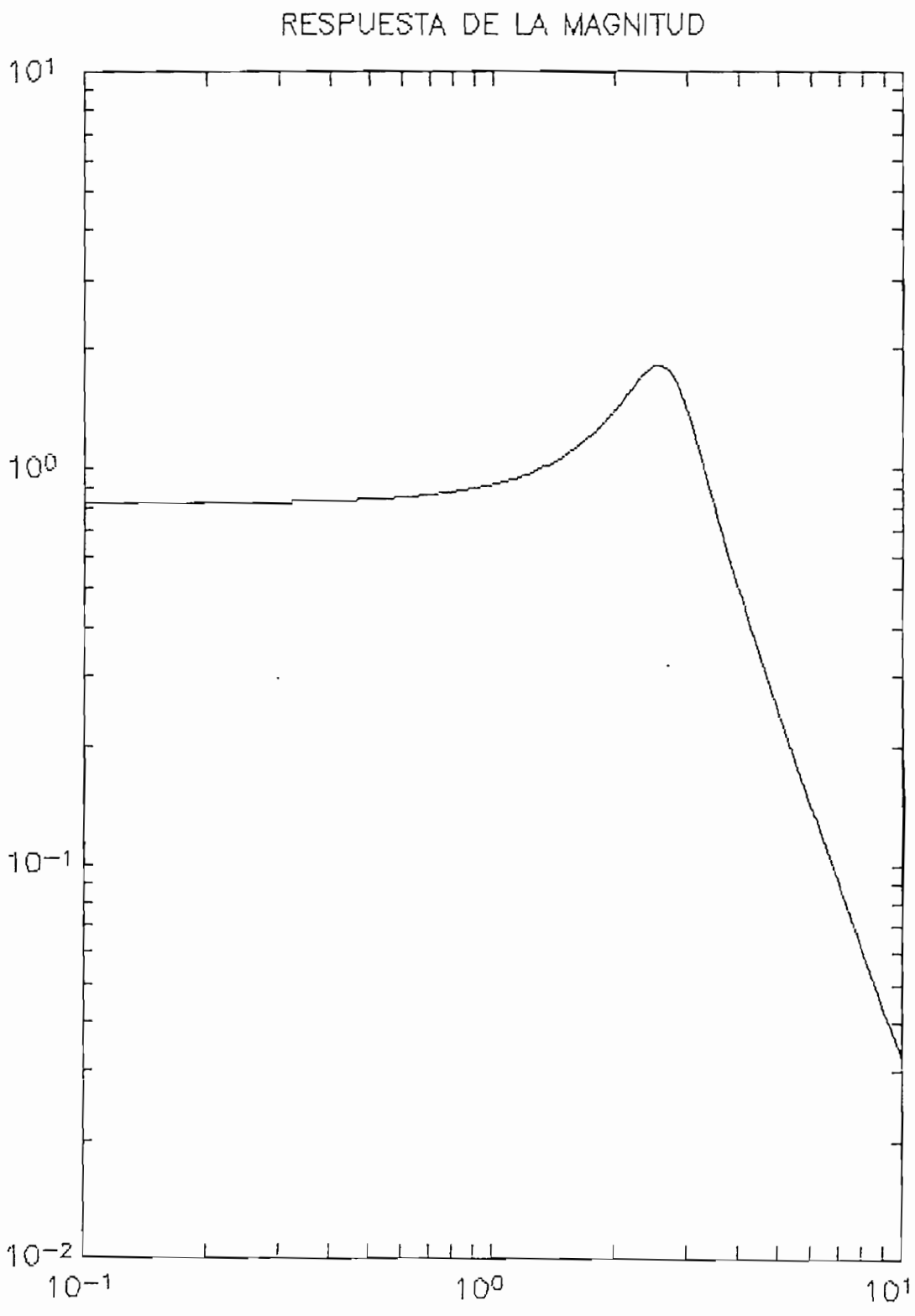
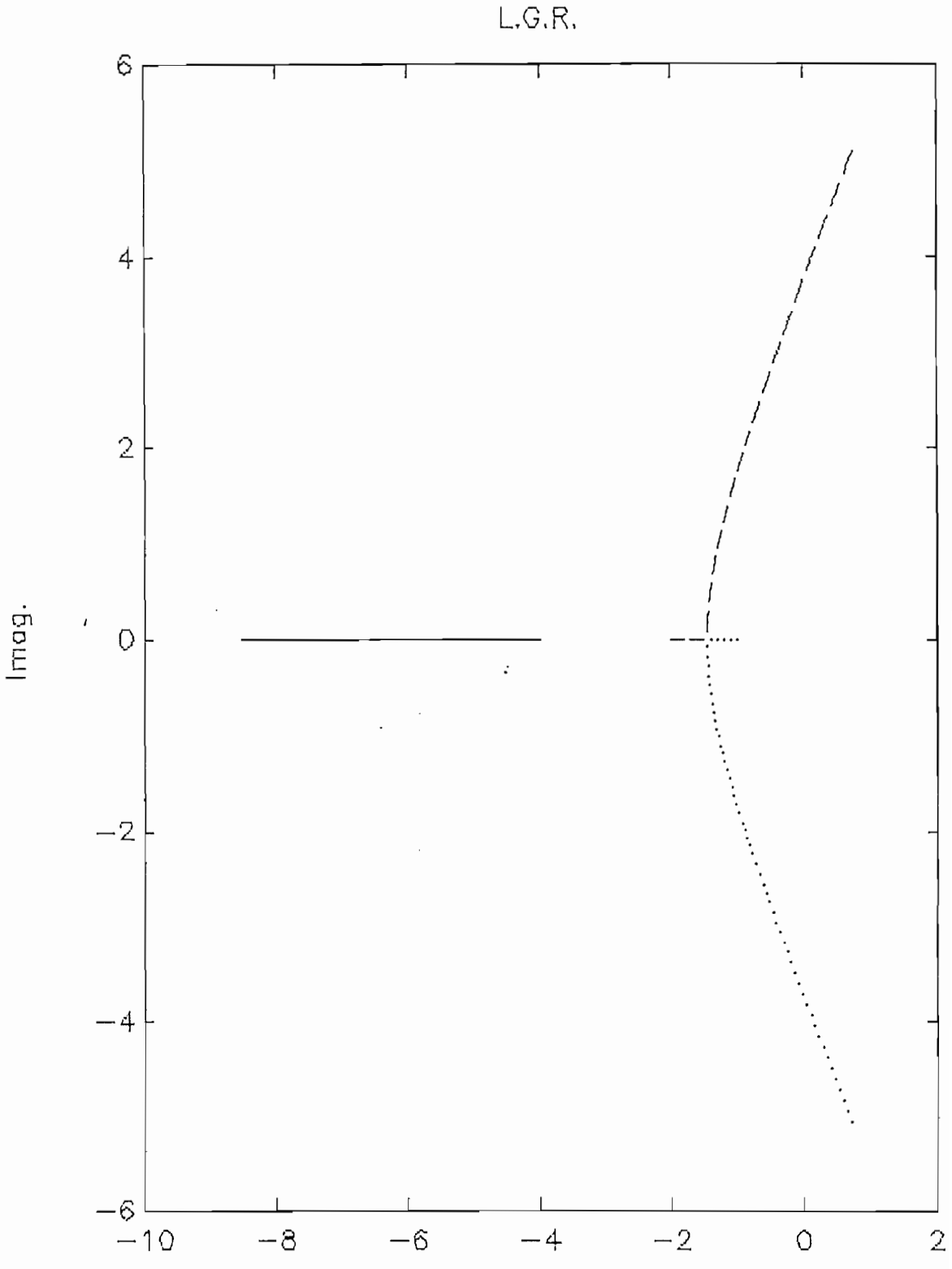


DIAGRAMA DE BODE EN LAZO CERRADO (MAGNITUD)
FIGURA 2.9



LUGAR GEOMETRICO DE LAS RAICES
FIGURA 2.10

2.3 APLICACION DE LOS PAQUETES PARA DISEÑO.

Los métodos para diseño y compensación de sistemas de control ampliamente utilizados son: el lugar geométrico de las raices, método de respuesta de frecuencia y realimentación de estado, entendiéndose por compensación el ajuste de un sistema para satisfacer condiciones dadas.

La compensación de un sistema de control se reduce al diseño de un filtro, cuyo propósito es compensar las caracteristocas indeseables e inalterables de la planta.

En forma muy general, se dice que los tipos de compensadores más usados son:

1. Constante, más bien se está hablando de un ajuste de ganancia.

2. Dinámicos, toman el nombre de redes o filtros:

   a. Red en Adelanto, el desfasaje adicional es positivo.

   b. Red en Atraso, el desfasaje adicional es negativo.

   c. Red atraso-adelanto, es una combinación de las dos redes mencionadas.

Además existen las acciones de control:

1. Control ON-OFF, es una acción no lineal de dos posiciones

2. Acciones:

a. Propocional: Gc(s) = kp

b. Integral: Gc(s) = ki/s

c. Derivativa: Gc(s) = kd*s

se puede hacer una combinación lineal de ellos, así:

- Compensador proporcional integral (PI)

- Compensador proporcional integral derivativo (PID), que es el más utilizado.

Para los métodos del lugar geométrico de las raíces y respuesta de frecuencia se considera el diseño de compensadores de atraso, adelanto o una combinación de estos, que no son más que funciones de transferencia para obtener el comportamiento deseado, es decir consiste en escoger en forma adecuada un polo y cero para alterar el lugar geométrico de las raíces o la respuesta de frecuencia de manera que se cumplan las especificaciones de funcionamiento.

El método del lugar geométrico de las raíces, es un procedimiento gráfico para determinar las ubicaciones de todos los polos de lazo cerrado a medida que varía un parámetro. Una ventaja es que resulta posible obtener información sobre la respuesta transitoria.

El efecto de agregar un polo a la función de transferencia en lazo abierto es el de desplazar el lugar geométrico hacia la derecha, tendiendo a *reducir* la

3. Si no se especifican los coeficientes de error estático se determina la ubicación del polo y cero de la red de adelanto, de manera que esta red contribuya el ángulo necesario ϕ y necesite la mínima cantidad de ganancia adicional. Si se especifica un coeficiente estático de error determinado, generalmente es más conveniente utilizar el método de respuesta de frecuencia, ó es necesario añadir una red de atraso, si solo el ajuste de ganancia no es suficiente.

4. Se determina la ganancia de lazo abierto del sistema compensado, con la condición de módulo.

Diseñado el compensador se verifica si cumple todas las características de funcionamiento, sino cumple con las especificaciones, se repite el procedimiento ajustando el polo y cero del compensador hasta que se cumpla dichas especificaciones.

Así, para el presente caso de estudio:

$$G(s) = \frac{k}{(s+4)(s+2)(s+1)}$$

Este sistema tiene las siguientes características (según el análisis realizado, página 28):

Mp % = 46.29 => ϵ = .24

ts = 5.8 s.

Ep % = 19%

Las características que se desea lograr son:

Mp ≤ 10%, luego ϵ ≥ .59 => ϵ = .71

ts ≤ 4s, luego wn ≥ 1.69 rad/s. => wn = 2.9 rad/s

Ep ≤ 10%

luego, el polo p_1 está en:

$p_1 = -2 + j\ 2.04 = \epsilon\ wn + j\ wn\ SQR(1 - \epsilon^2)$

$$G_c(s) = k_c \frac{(s+z)}{(s+p)}$$

$\Phi = \angle G(s) \big|_{P_D}$

$\Phi = -251.83°$

$\Phi_c = -180° + 251.83° = 71.83°$

Como el ángulo es mayor de 60° se tiene dos opciones:

- hacer un solo compensador
- dos compensadores de 35°

Con la primera opción: $\Phi_c = 72°$

Se ubica el cero en -2, entonces:

$\Phi_c = \Phi_z - \Phi_p$

$\Phi_p = 90° - 72° = 18°$

$$tg18° = \frac{2.04}{p-2}$$

luego: p = 8.3

Se procede a realizar un ajuste de ganancia:

$$G(s)G_c(s) = \frac{Kc_T}{s^3+7s^2+14s+8} \frac{s+2}{s+8.3}$$

Se procede a hacer un ajuste de ganancia con la ayuda del lugar geométrico de las raíces, obteniéndose:

$Kc_T = 41.63 \Rightarrow 42$

entonces:

$$Gc_A(s) = \frac{1.2(s+2)}{(s+8.3)}$$

De esta manera se logra modificar el lugar geométrico de las raíces para que pase por los polos deseados.

Para satisfacer el error en estado estacionario Ep ≤ 10% sin modificar apreciablemente el lugar geométrico debe

añadirse una red de atraso limitando la contribución angular de la red de retardo a un valor pequeño, para lograrlo, se ubica el polo y el cero de la red relativamente cerca origen, entonces:

$$Gc_R(s) = \frac{s+.9}{s+.09}$$

lo que da el valor de ganancia requerido (ganancia a baja frecuencia) para satisfacer el error.

Entonces:

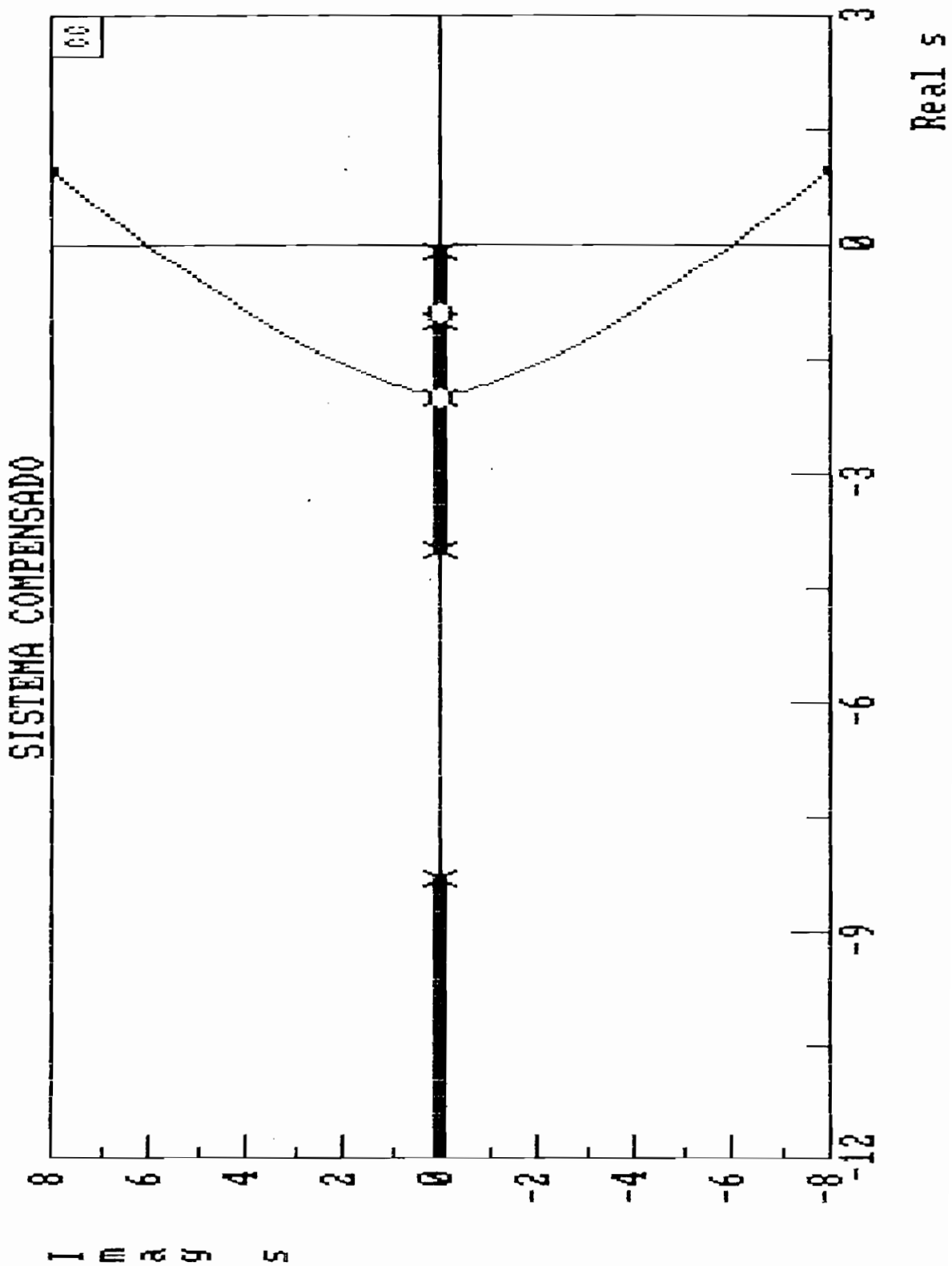
$$G_T(s) = Gc_A(s) * Gc_R(s) * G(s)$$

Las figuras 2.11 y 2.12 muestran el lugar geométrico compensado y la respuesta en el tiempo del sistema compensado como se puede apreciar se satisface las especificaciones de diseño, puesto que:

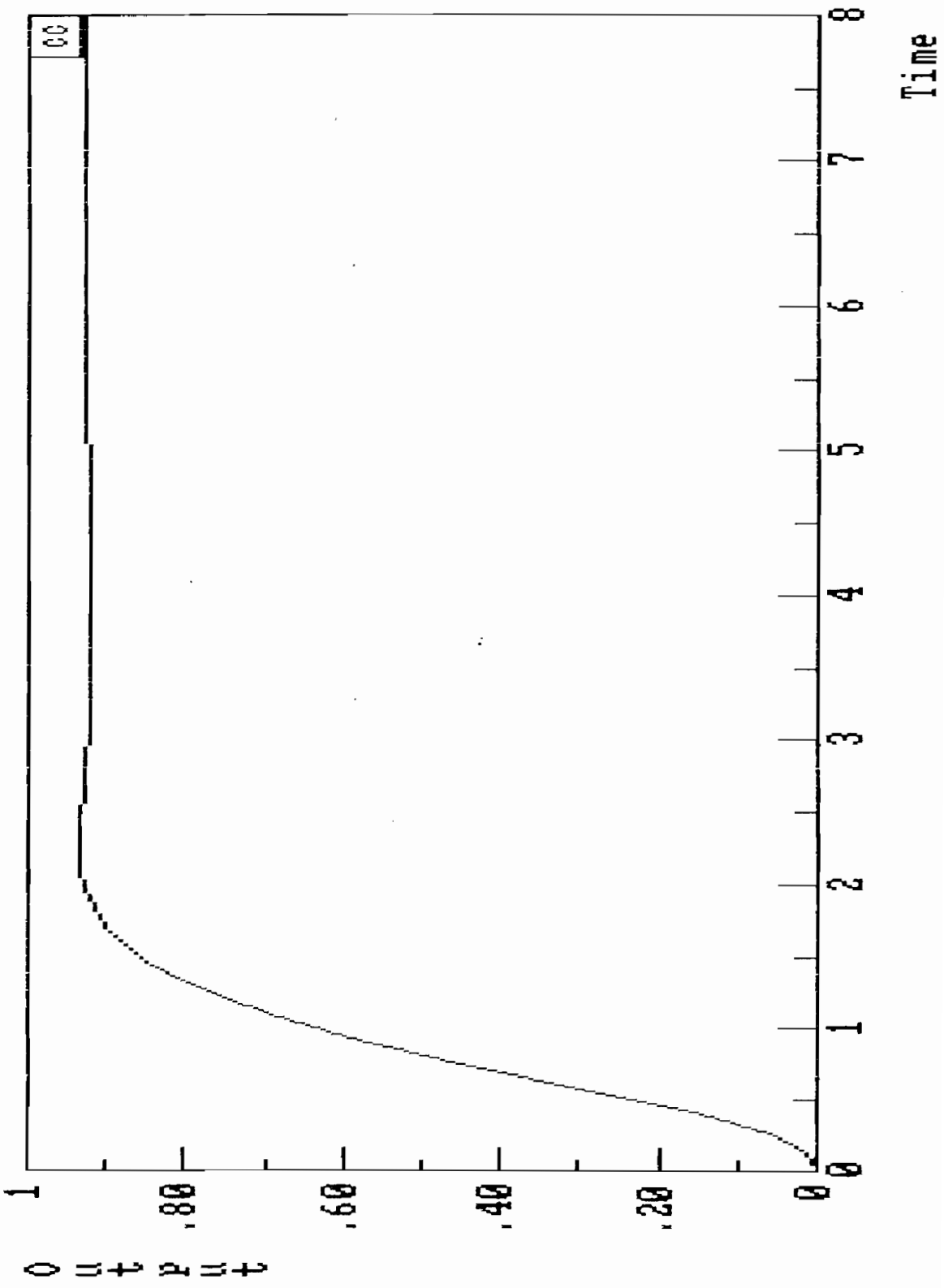
Ep% = 7.5%

Mp% = 0.8%

ts = 1.8 s.



LUGAR GEOMETRICO DE LAS RAICES DEL SISTEMA COMPENSADO
FIGURA 2.11



RESPUESTA EN LAZO CERRADO DEL SISTEMA COMPENSADO
FIGURA 2.12

Como otra opción se tiene el colocar dos compensadores en cascada de 35° cada uno, entonces:

$$\phi_p = 90° - 35° = 55°$$

Se ubica el cero en −2, entonces:

$$tg55° = \frac{2.04}{p-2}$$

luego:

$$p = 3.4$$

Se procede a realizar un ajuste de ganancia:

$$G(s)G_c(s) = \frac{Kc_T}{s^3+7s^2+14s+8}\left[\frac{s+2}{s+3.4}\right]^2$$

Se realiza el ajuste de ganancia con el método indicado anteriormente, entonces:

$$Kc_T = 19$$

donde:

$$Gc_A(s) = .54\left(\frac{s+2}{s+3.4}\right)^2$$

Se añade un red de atraso para mejorar el error de posición.

$$Gc_R(s) = \frac{s+.9}{s+.06}$$

Entonces:

$$G_T(s) = Gc_A(s) * Gc_R(s) * G(s)$$

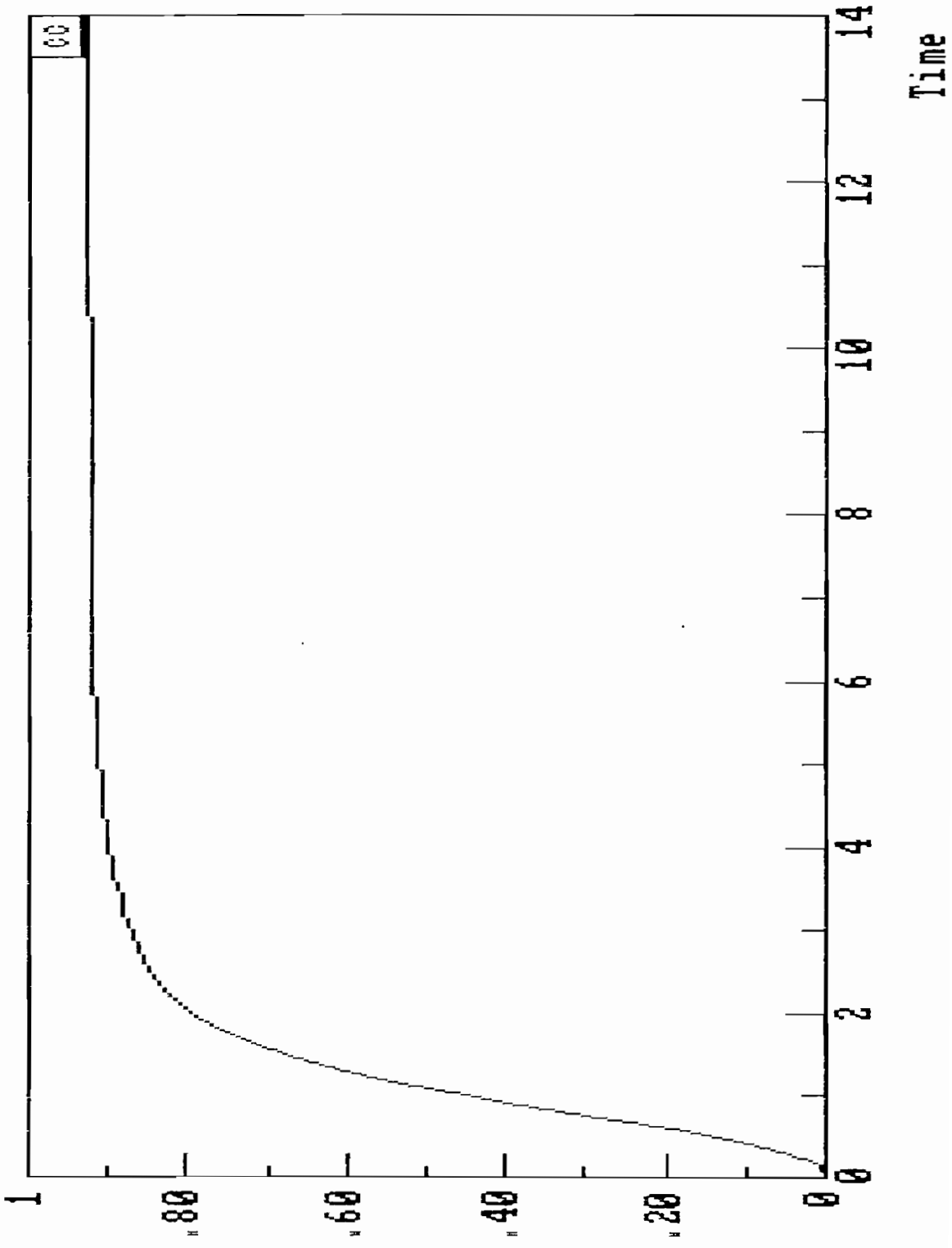
Las figuras 2.13 y 2.14 muestran el lugar geométrico compensado y la respuesta en el tiempo del sistema compensado, de donde:

Ep = 4.4 %

Mp = 0 %

ts = 4.5 s.

Por lo que se considera mejor el diseño anterior, ya que a más de acercarse a las especificaciones pedidas es de menor orden.



RESPUESTA EN LAZO CERRADO DEL SISTEMA COMPENSADO
FIGURA 2.14

- **Método de respuesta de frecuencia:**

La función primaria de un compensador de adelanto en el dominio de la frecuencia es modificar la respuesta de frecuencia para dar un suficiente ángulo de adelanto de fase para conseguir el margen de fase requerido. El procedimiento a seguir puede ser resumido del siguiente modo:

1. Determinar la ganancia del lazo abierto K para satisfacer los requerimientos de coeficiente de error.
2. Usando la ganancia K, así determinada, calcular el margen de fase del sistema no compensado.
3. Determinar el ángulo de adelanto de fase Φ necesario que debe ser agregado al sistema.
4. Determinar el factor de atenuación α utilizando la ecuación:

$$SEN\Phi_m = \frac{1-\alpha}{1+\alpha}$$

Determinar la frecuencia a la cual la magnitud del sistema no compensado es igual a $-20*\log(1/\sqrt{\alpha})$. Fijar esta frecuencia como la nueva frecuencia de transición de ganancia. Esta frecuencia corresponde a w_c y el defase máximo Φ_m se produce a esta frecuencia.
5. Determinar el polo y cero de la red de adelanto a partir de:

$$w_c = \frac{1}{T\sqrt{\alpha}}$$

$$z = \frac{1}{T}$$

$$p = \frac{1}{\alpha T}$$

Finalmente se inserta un amplificador con ganancia igual a $1/\alpha$ o se incrementa la ganancia del amplificador existente en un factor de $1/\alpha$.

Siguiendo este procedimiento se obtiene:

$$G(s) = \frac{35}{s^3+7s^2+14s+8}$$

Para satisfacer los requerimientos de diseño para un Ep $\leq$ 10%, se debe tener una ganancia K = 2, con lo cual el MF = 8.5°. Se desea como especificación llegar a un MF > 50°, para lo cual se escoge un MF = 70° y entonces el ángulo de la red de adelanto ϕ_m debe ser de 61.5°, sin embargo a este valor se le deberá sumar un ángulo extra de 5° a 10° para compensar el desplazamiento de la frecuencia de transición de ganancia, con lo cual ϕ_m = 72°.

MF = 180° − 171.45° = 8.5°

ϕ_m = 70° − 8.5° + 10° = 72°

Se calcula el valor de α:

$$sen72° = \frac{1-\alpha}{1+\alpha}$$

α = ..025

$-20 \log (1/\sqrt{.025}) = -16$

con este valor se ubica w_c, a partir del diagrama de Bode, luego la red en adelanto queda determinada como:

w_c = 7.11

z = 1.12

p = 44.97

$$GC_A(s) = \frac{1}{.025} \times \frac{s+1.12}{s+44.97}$$

$$GC_T(s) = \frac{70}{s^3+7s^2+14s+8} \frac{1}{.025} \frac{s+1.12}{s+44.97}$$

Debido a que el máximo sobreimpulso es muy alto, se debe ajustar el control mediante la técnica de ensayo y error, es decir ir ajustando la ganancia, para obtener el margen de fase deseado y disminuir el sobreimpulso.

Se mejora el máximo sobreimpulso a costa del error de posición, para contrarrestar se debe añadir una red en atraso. La característica de una red de atraso es dar alta

ganancia en baja frecuencia para dar al sistema una mejor precisión. Se llega entonces a:

$$GC_R(s) = \frac{s+.8}{s+.08}$$

$$GC_A(s) = \frac{1}{.025}\,\frac{1}{2.8}\,\frac{s+1.12}{s+44.97}$$

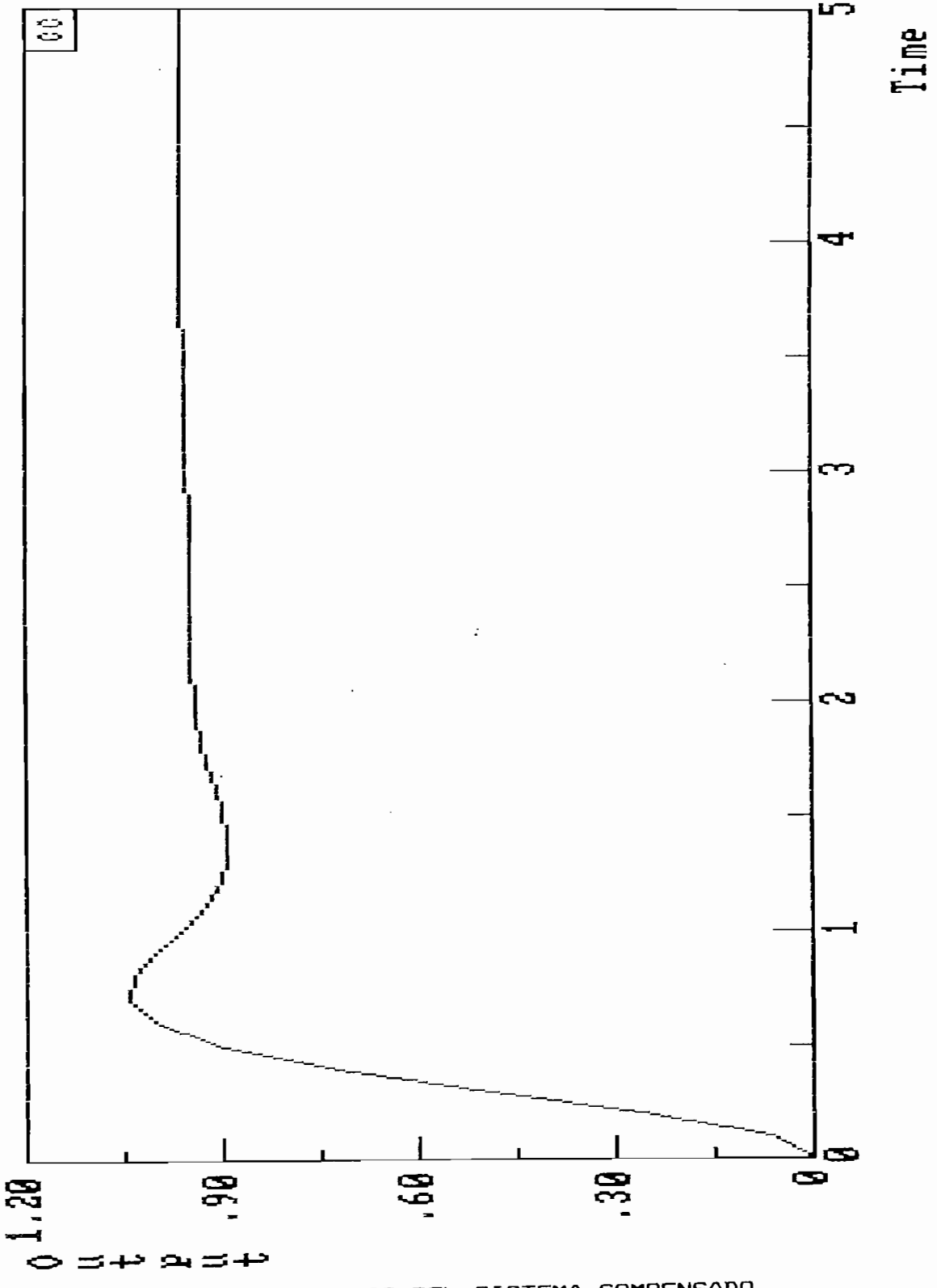
Luego:

$$G_T(s) = \frac{1000(s+.8)(s+1.12)}{(s^3+7s^2+14s+8)(s+.08)(s+44.97}$$

Obteniéndose:

Ep% = 3.3%	MF = 58.02°
ts = 2.84 s.	MG = 21.29 db.
Mp% = 7.8%	Mr = 1.027 db.
	AB = 6.098 rad/s

Las figuras 2.15, 2.16 y 2.17 muestran las respuestas en el tiempo en lazo cerrado, de frecuencia lazo abierto y lazo cerrado del sistema compensado. Como se aprecia se satisface las especificaciones de diseño.



RESPUESTA EN LAZO CERRADO DEL SISTEMA COMPENSADO
FIGURA 2.15

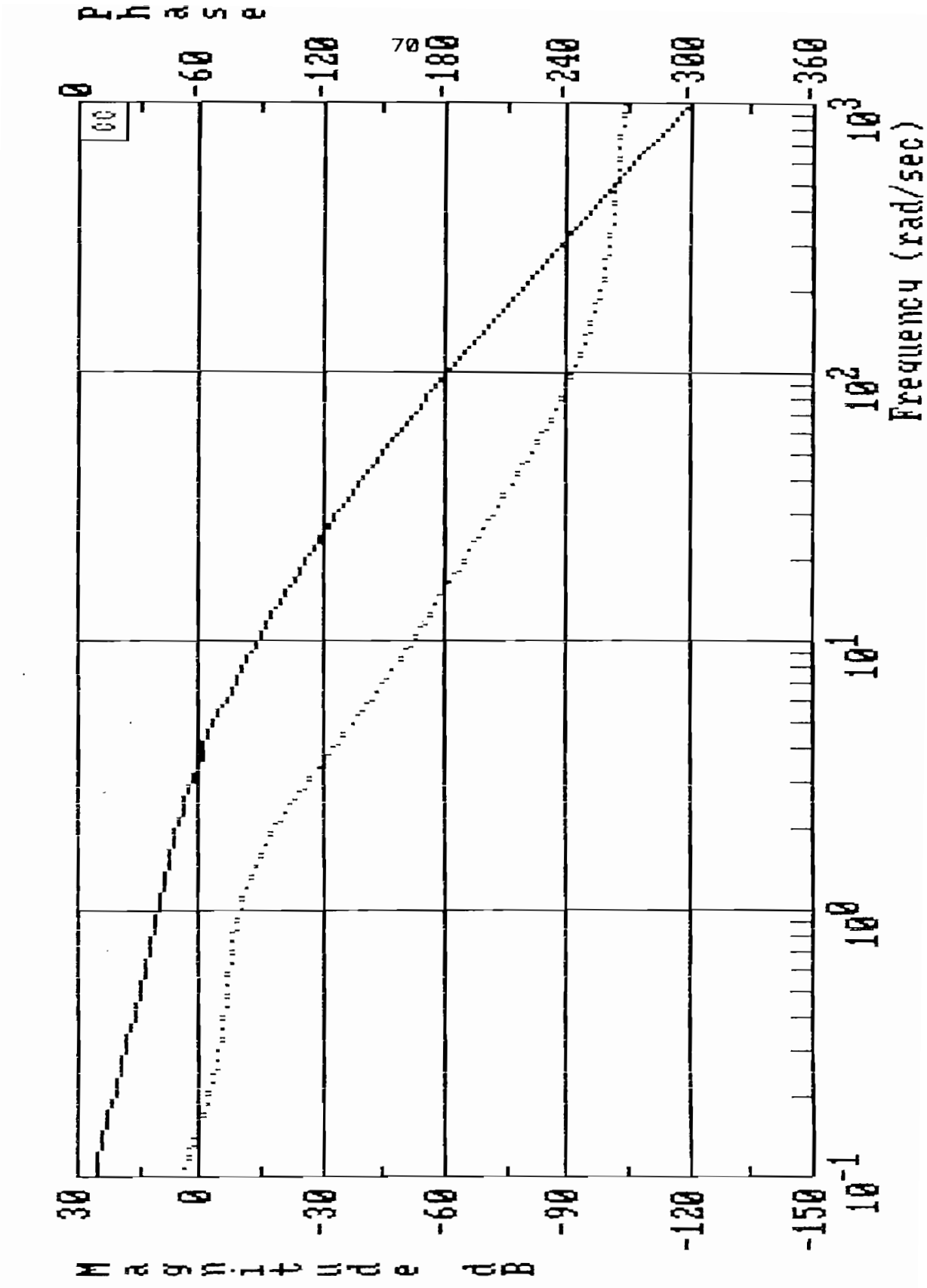


DIAGRAMA DE BODE EN LAZO ABIERTO DEL SISTEMA COMPENSADO
FIGURA 2.16

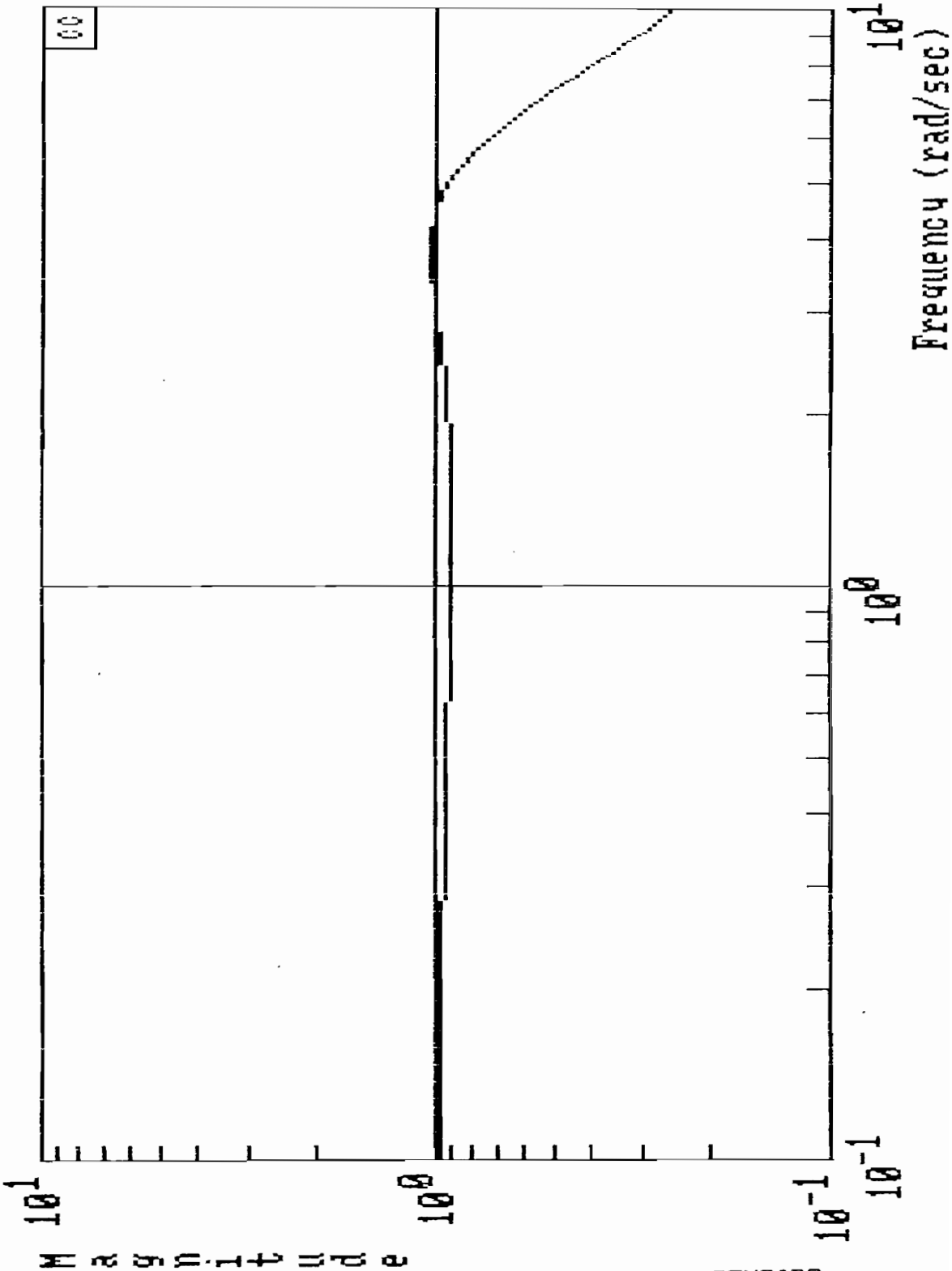


DIAGRAMA DE BODE EN LAZO CERRADO DEL SISTEMA COMPENSADO
FIGURA 2.17

- Compensador PID:

Una de las técnicas de compensación también usada es con la ayuda de las acciones de control. Dependiendo de las acciones que se efectúe sobre el error actuante (e = r - Hy) se pueden definir tres acciones de control primarias, en base a las cuales, combinándolas, se puede conseguir efectos bastante significativos.

En la acción propocional la variable manipulada (señal de control), la que se aplica a la planta es proporcional al error, es decir:

$$m - Kp * e$$

$$M(s) - kp * E(s)$$

donde:

K_P = CONSTANTE PROPORCIONAL

En la acción integral la variable manipulada es proporcional a la integral del error:

$$m - K_I \int e(t)\, dt$$

donde:

K_I = CONSTANTE INTEGRAL

$$M(s) = \frac{K_I}{s} E(s)$$

En la acción derivativa la variable manipulada es proporcional a la derivada del error:

$$m = K_D \frac{de}{dt}$$

donde:

K_D = CONSTANTE DERIVATIVA

$$M(s) = K_D s E(s)$$

La función de transferencia del controlador PID (Proporcional integral derivativo) será de la forma:

$$G_C(s) = K_P + K_D * s + \frac{K_I}{s}$$

Para el uso de estas acciones de control con mayor facilidad se hizo un macro con la ayuda del CAD CONTROL, con el nombre de PID.MAC

Para el diseño de control PID se cancela los polos más cercanos al eje "jw". Se tiene entonces:

$$G(s)Gc(s) = \frac{35}{(s+1)(s+2)(s+4)} * \frac{kd(s^2+\frac{Kp}{Kd}s+\frac{Ki}{Kd})}{s}$$

$$G(s)Gc(s) = \frac{35}{(s+1)(s+2)(s+4)} * \frac{Kd(s+2)(s+1)}{s} = \frac{35Kd}{s(s+4)}$$

entonces: $(s+1)(s+2) = s^2 + 3s + 2$

$Kp/Kd = 3$

$Ki/Kd = 2$

Sobre el nuevo lugar geométrico se realiza un ajuste de ganancia para tener un Mp = 5%, ts = 2 s., (polos deseados = -2 +/- j*2.04), con lo cual K = 8.21, entonces Kd = 8.21/35, entones:

$Kd = .2345$

$Kp = .2345 * 3 = .7035$

$Ki = .2345 * 2 = .469$

Con estos valores se realiza un redondeo de los parámetros Kp, Ki, Kd, definitivos del controlador. Obteniéndose finalmente:

Kd = .23

Kp = .7

Ki = .47

El resultado en el tiempo se muestra en la figura 2.18 de la cual se tiene:

Mp% = 5%

ts = 2.1 s.

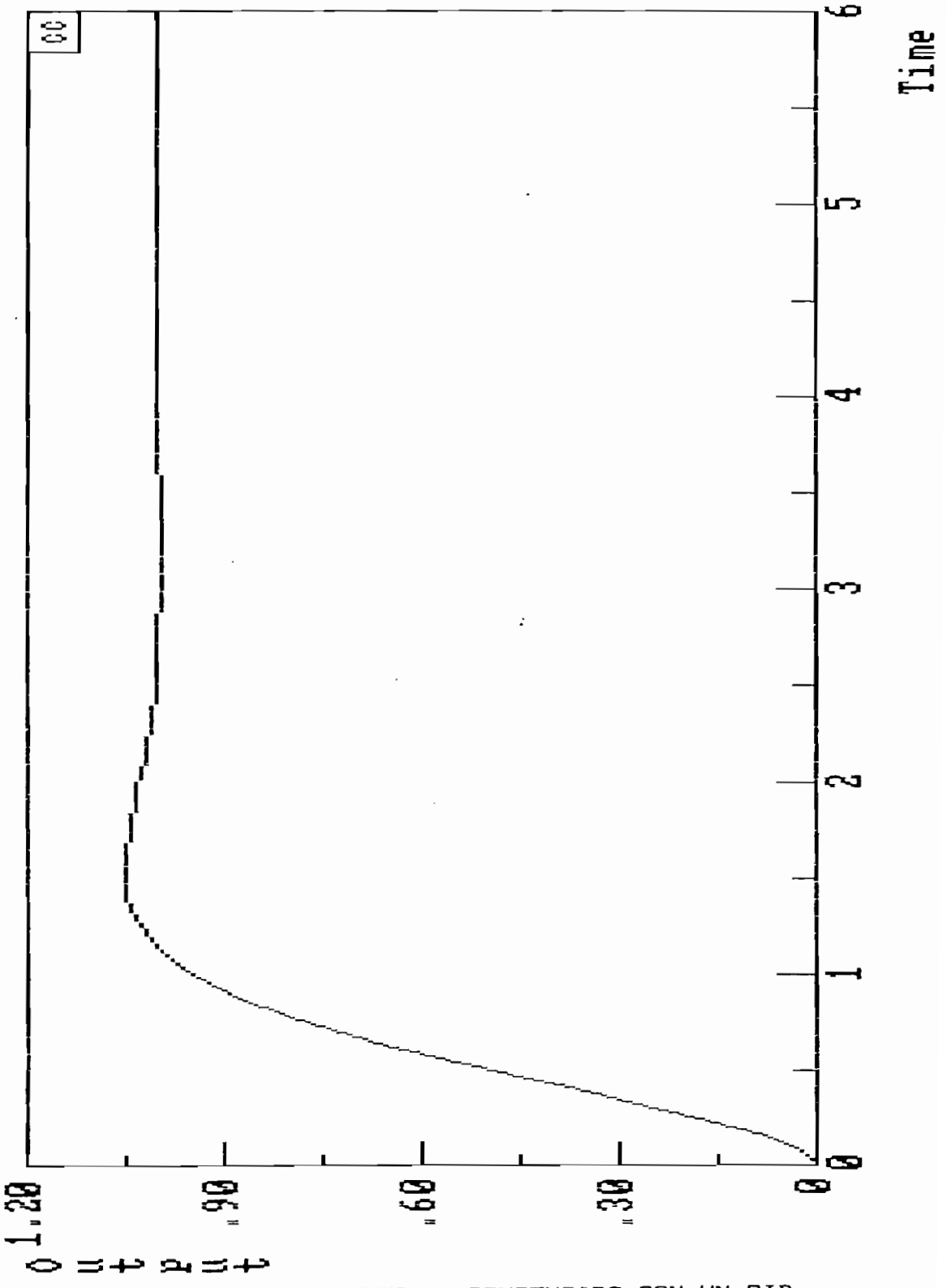
Ep% = 0%

con lo que se satisface las especificaciones dadas.

- Realimentación de estado:

La compensación por realimentación de estado implica la ubicación de polos de lazo cerrado deseado. Se escoge dos polos complejos conjugados y uno real bien alejado que se lo considere despreciable: $-1.2+1.184j$, $-1.2-1.184j$, -13. Con estos polos se satisface un Mp % < 5%, ts < 2s.

Este diseño se realiza con la ayuda, para este caso, de los paquetes CAD CONTROL y PC-MATLAB.



RESPUESTA DE UN SISTEMA COMPENSADO CON UN PID
FIGURA 2.18

PAQUETE CAD CONTROL:

Lo primero que se debe hacer es ingresar la función de transferencia. Luego se ingresa al comando STATE y se utiliza el comando CCF, cuya función es colocar en Pj la forma canónica controlable de Gi(s) (debe cumplir que el #polos ≥ #zeros).

Con el comando POLEPLACEMENT se obtiene una realimentación de estado ingresando los polos deseados. La matriz obtenida está en P = F. Para este caso

F = [28.94 20.04 8.4]

Con la ayuda de FEEDBACK se obtiene el modelo del sistema realimentado, del cual se quiere graficarla respuesta con el comando PLOT, una vez que se haya realizado la simulación SIMULATION.

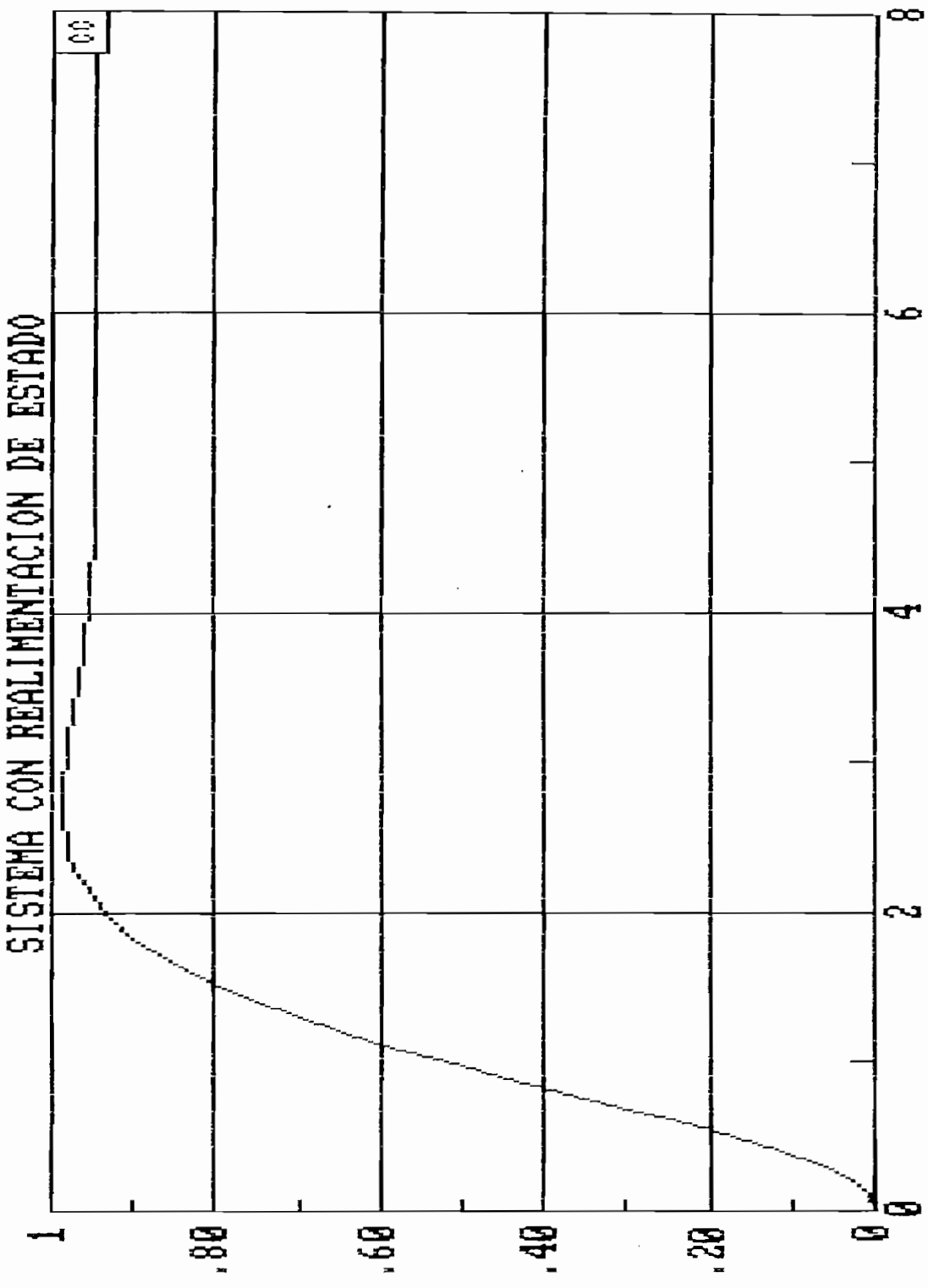
Para facilitar se creó el macro con el nombre REALI.MAC.

El gráfico obtenido se puede visualizar en la figura 2.19. Se obtiene las siguientes características:

Ep% = 5.4%

ts = 3.6 s.

Mp% = 4.1%



RESPUESTA DE UN SISTEMA CON REALIMENTACION DE ESTADO
FIGURA 2.19

PAQUETE PC-MATLAB:

Se procede al ingreso de la función de transferencia:

```
num = [0,0,0,35]
den = [1,7,14,8]
```

se realiza la transformación a variables de estado:

```
[a,b,c,d] = tf2ss(num,den)
```

se ingresa los polos deseados en P

```
P = [-1.2+1.184*j, -1.2-1.184*j, -13]
```

luego se pocede al cálculo de la matriz que contiene las ganancias de la realimentación de estado:

```
K = place(a,b,P)
```

Se obtuvo:

```
K = [8.4    20.042    28.94]
```

Al calcular la matriz de lazo cerrado, se procede a realizar un archivo .m que se llama lcsr.m, en el que se utiliza la ecuación:

```
lc = a - b*K
```

Una vez obtenida la matriz lc en lazo cerrado se pocede a graficar la salida:

```
num = [0,0,0,35];
den = [1,7,14,8];
[a,b,c,d] = tf2ss(num,den);
P = [-1.2 + 1.184*j, -1.2-1.184*j, -13];
k = place(a,b,P)
lc = lcsr[a,b,k]
t:0:.1:10
y = step(lc,b,c,d,1,t);
plot(y), title('RESPUESTA DE UN SISTEMA CON REALIMENTACION')
```

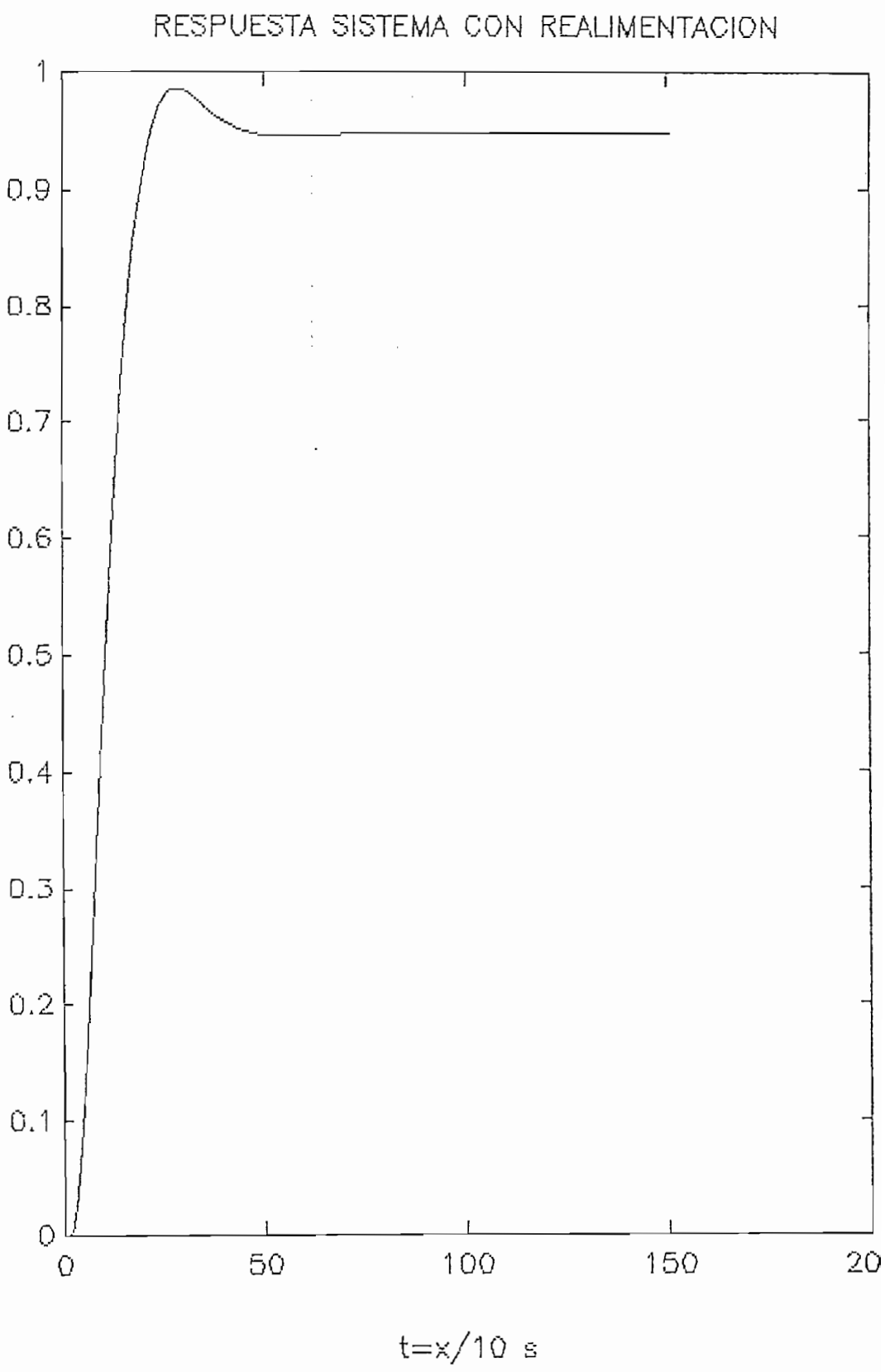
El resultado se obtiene en la figura 2.20, del cual se obtiene que:

Mp % = 4.5%

Ep % = 5.7%

ts = 4.1 s.

Como se aprecia se cumplen las especificaciones.



RESPUESTA DE UN SISTEMA CON REALIMENTACION DE ESTADO
FIGURA 2.20

3. FUNDAMENTOS TEORICOS

3.1 INTRODUCCION.

En este capítulo se analizarán las rutinas básicas de un sistema en tiempo real, como son la de adquisición de datos, que consiste en una recolección de información que describe la dinámica de una planta o proceso; así como la salida de datos que es un flujo de los mismos desde el sistema de adquisición de datos hacia el exterior (planta).

Con la ayuda de la adquisición y salida de datos conjuntamnete con un algoritmo apropiado se logrará obtener un modelo la dinámica de la planta, que reflejará lo que ha sucedido en la misma, mientras se satisfacía una condición, la misma que suele estar definida por una base uniforme de tiempo.

Las tareas que se pueden lograr con los sistemas en tiempo real se caracterizan por la habilidad de ejecutar la adquisición, salida de datos y/o control en un intervalo apropiado de tiempo.

En la rutina de identificación se trabajará con sistemas de los cuales se desea encontrar el modelo que cumpla con sus características esenciales. No se busca la información sobre el sistema por sus componentes físicos, sino la información sobre su dinámica, a través de los datos experimentales de entrada y salida de la planta. El proceso consiste en una

estimación de sus parámetros a partir de los datos de entrada y salida de la planta. La estimación permite llegar a un modelo que represente de la mejor manera posible a la dinámica real de la planta. La información necesaria se la obtiene midiendo los datos de entrada y salida.

El proceso de modelación se puede realizar fácilmente mediante un computador. Allí será necesario tener la información en forma discreta y el modelo encontrado será discreto. Para esto se requerirá de un algoritmo que asegure la convergencia y permita la obtención de la mejor manera de los parámetros con la mayor precisión posible.

Existe el inconveniente de determinar el orden del modelo que se desea construir antes de iniciar la identificación. Esto lleva a un conocimiento previo de la dinámica de la planta. A a esto se debe sumar el hecho de que se debe encontrar la mayor información tanto a la entrada como a la salida para abarcar en forma suficiente la dinámica de la planta sin alterar su funcionamiento, esto se consigue de mejor manera mediante pequeñas perturbaciones a la entrada.

Cuando se desea que un sistema cumpla con ciertos objetivos predeterminados se hace necesaria la construcción de un sistema de control realimentado que actúe sobre las entradas de la planta. Para conseguir estos objetivos la

configuración básica consta de una planta (modelo), de un controlador, de un lazo de realimentación y de un referencia.

Conociendo la referencia y la planta a través del modelo, el problema radica en el diseño del controlador, para de esta manera cumplir con ciertas características o especificaciones del sistema. Seleccionando la ley de control que se considere adecuada, se deberá estudiar la estabilidad del sistema, su respuesta transitoria, sus restricciones y robustez y con esta información llegar al diseño final del controlador.

Para este caso el controlador trabajará con señales discretas en el tiempo, por lo que trata de un controlador digital, su salida necesita de un conversor D/A para generar las señales que requiere la planta.

El computador aprecerá como parte integrante del lazo de realimentación, actuando como controlador, es decir que la ley de control aparece como un algoritmo que procesa la información del error y entrega datos de salida hacia la planta, sin la participación del usuario en el proceso.

El papel del usuario se limitará a las inicializaciones que sean necesarias sobre la ley de control.

En este trabajo de tesis se han desarrollado rutinas de:

- Instrumentación en tiempo real para adquisición y salida de datos.

- Control en tiempo real mediante controles PID y REDES.

- Identificación de sistemas en simulación y en tiempo real.

Se debe aclarar que no se realizan rutinas en tiempo real a variables de estado, solo se trabaja a variables de estado en simulación con los paquetes PC-MATLAB y CAD CONTROL, ya que el objetivo básico es trabajar a nivel de función de transferencia.

A continuación se describirán las mencionadas rutinas y algunos casos de estudio que servirán para hacer uso de dichas rutinas tanto para modelación como para control. Estos casos de estudio consisten en:

- Circuito RC pasivo de primer orden.
- Circuito RC activo de primer orden.
- Circuito activo de segundo orden.

Para cada uno de estos casos se procederá a obtener su modelo y a realizar análisis mediante la simulación, con la ayuda de los paquetes CAD CONTROL y PC-MATLAB. Se continuará

con el diseño del controlador apropiado, pudiendo ser una red de compensación o un control proporcional integral derivativo. Luego de la simulación respectiva se implementará un control digital directo y se analizará su comportamiento.

3.2 INSTRUMENTACION EN TIEMPO REAL.

La adquisición y salida de datos en la estación KEITHLEY 500A se puede realizar en dos modos diferentes de trabajo FOREGROUND y en BACKGROUND, mediante interrupciones no mascarables, es decir de máxima prioridad. En los dos casos puede hacerse adquisición de datos unicanal y multicanal.

Es de interés en este trabajo realizar la adquisición y salida de datos en forma unicanal y en background.

Se procedió a realizar un programa, llamado INSREAL, el mismo que consta de:

- Adquisición de datos
- Salida de datos
- Adquisición de datos + algoritmo + salida de datos

Su diagrama de flujo se presenta en la figura 3.1

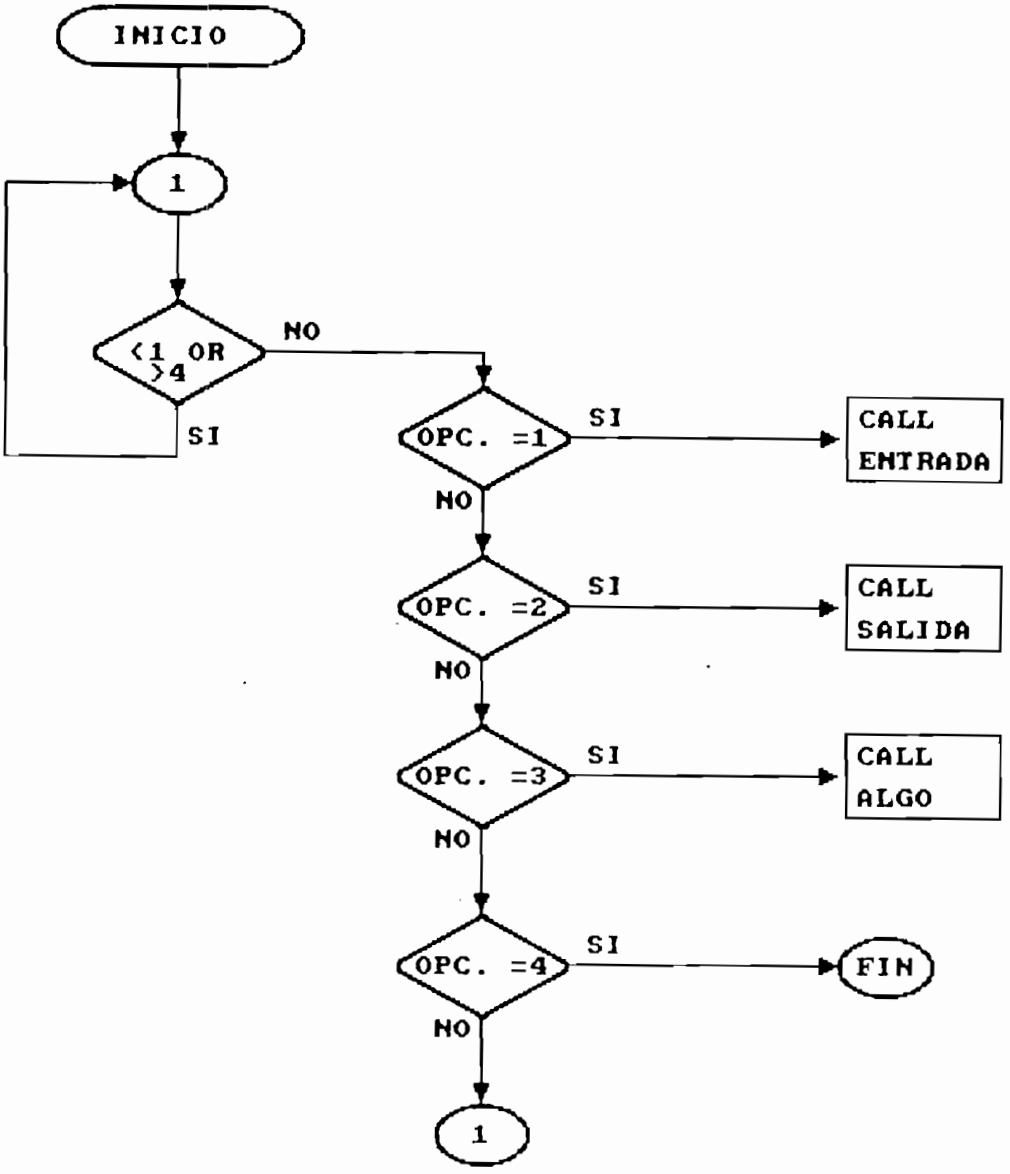


DIAGRAMA DE FLUJO DEL PROGRAMA INSREAL
FIGURA 3.1

3.2.1 Adquisición unicanal de datos análogos en background.-

El objetivo es realizar la adquisición de datos en tiempo real a través de un solo canal análogo en background. A pesar de que es una rutina sencilla será de mucha utilidad, pues se la usará para todos los restantes programas que se desarrollarán.

Es una rutina que al tiempo que realiza la adquisición de datos, efectúa tareas en foreground, como son los gráficos.

Los comandos necesarios son:

- **CALL ANIN:** Rutina que realiza el muestreo de hasta 64 canales de entrada análogos. El comando ANIN toma medidas a intervalos constantes de tiempo en un número determinado de canales y crea una área de memoria del Quick500, en la que guarda los valores adquiridos.

CALL ANIN(arn$, sn!, ion1$, bintv%, cy%, tn$, bfn$)

arn$ (string-entrada).- Nombre del arreglo creado debido a la ejecución del comando ANIN, estos arreglos son accesibles a través de los comandos de manejo de áreas solamente.

sn! (real-entrada).- Indica la profundidad o longitud del

arreglo creado, en el caso unicanal es un vector, en el caso multicanal es una matriz.

ioni$ (string-entrada).- Indica los nombres de los canales de entrada que serán muestreados por el comando ANIN. Los nombres de cada canal se separan por espacios y/o comas. El número de canales utilizados especifica el ancho del arreglo de memoria creado.

bintv% (integer-entrada).- Especifica cada que número de período de interrupción se realiza un muestreo de los canales de entrada habilitados.

cy% (integer-entrada).- Especifica el número de veces que se debe repetir esta tarea de background, si se asigna cy% = 1, esta tarea se ejecuta de manera indefinida.

tm$ (string-entrada).- Modo de disparo, ANIN, soporta los siguientes modos de disparo:

"NT" No trigger	Ejecución inmediata
"WBT" Wait background trig	Espera disparo de background
"WGO" Wait for GONOW	Espera disparo de GONOW
"BT" Background trigger	Actúa como disparador de background.

El que se utilizará en este trabajo es el de ejecución inediata (NT).

bfn$ (string-entrada).- Sirve para nombrar las tareas ejecutadas, si no se utiliza se debe ingresar un string nulo "".

- **CALL ARGETVALF**.- Permite accesar a un simple valor guardado en un arreglo Quick500, especificando su ubicación en profundidad y anchura, y expresarlo como una variable BASIC. Guarda valores en un variable real y puede expresarse en el tipo de unidades más convenientes.

CALL ARGETVALF(arn$,dep!,wid%,ion$,va!,euf%)

arn$ (string-entrada).- Identifica el área desde la que se toman datos, este arreglo es creado anteriormente, mediante los comandos ARMAKE o ANIN.

dep! (real-entrada).- Indice de la profundidad del dato dentro del arreglo. El máximo valor permisible es la longitud del arreglo, esta longitud (en su valor máximo) depende de la cantidad de memoria reservada para el almacenamiento de datos al instalar el programa (64 K, para configuración existente).

wid% (integer-entrada).- Inidica la ubicación del dato buscado respecto de la anchura del arreglo, cuando se utiliza

ion$ no se utliza **wid%**, en cuyo caso se asigna el valor de menos 1.

ion$ (string-entrada).- Se utiliza con el mismo propósito que el parámetro wid%, indicando el nombre de un canal de entrada. Si se va a utilizar **wid%**, **ion$** dede ingresarse con un string nulo "".

va! (real-salida).- Es una variable BASIC real en la que se guarda el valor tomado del arreglo Quick500, en cualquier tipo de unidades.

euf% (integer-entrada).- Esfecifica el tipo de unidades en la que se almaçenará el valor en la variable BASIC, 0 asigna a voltios, 1 a milivoltios, 2 a microvoltios.

- CALL **INTON**.- Habilita las interrupciones, a la vez que inicializa la frecuencia de las mismas de acuerdo a un valor especificado por el usuario. Las tareas de background que generalmente son inicializadas con anterioridad, no se ejecutan hasta que el comando **INTON** sea ejecutado.

```
CALL INTON(ir%, tu$)
```

ir% (integer-entrada).- Parámetro que especifica el período de interrupción. Tiene ciertos valores válidos de acuerdo a las unidades de tiempo escogidas.

tu$ (string-entrada).- Ingresa las unidades de tiempo y existen cuatro cadenas de caracteres válidos:

"HMIC"	ciento de microsegundos	ir% = 1-32767
"MIL"	milisegundos	ir% = 1-32767
"SEC"	segundos	ir% = 1-4400
"MIN"	minutos	ir% = 1-74

- **CALL INTOFF.-** Deshabilita las interrupciones que fueron habilitadas por el comando **INTON.** Este comando no borra o limpia las tareas de background, es decir que si existe un nuevo comando INTON se volverán a ejecutar estas tareas, aunque se pueda perder el sincronismo, ya que todos los timers (en software y hardware) son deshabilitados. Este comando no tiene parámetros. El diagrama de flujo de la adquisición de datos análogos unicanal en background se presenta en la figura 3.2

En el programa desarrollado se usa variables extras, así: **lp!** = **1**, para el primer dato adquirido. Con este valor, más otra variable **lip**, cuyo valor puede ser Ø ó 1, se hace una comparación, formando dos condicionantes, para setear a **l!** = Ø, por solo una vez. Esta variable servirá para ir obteniendo en pantalla sin sobreponer los resultados e ir verificando si se ha completado con el número de datos deseados. La comparación **lp!<>** **plp!** hace posible que se continúe con el proceso, cuando **lp!** = **plp!** significa que se ha completado con el número de datos deseados.

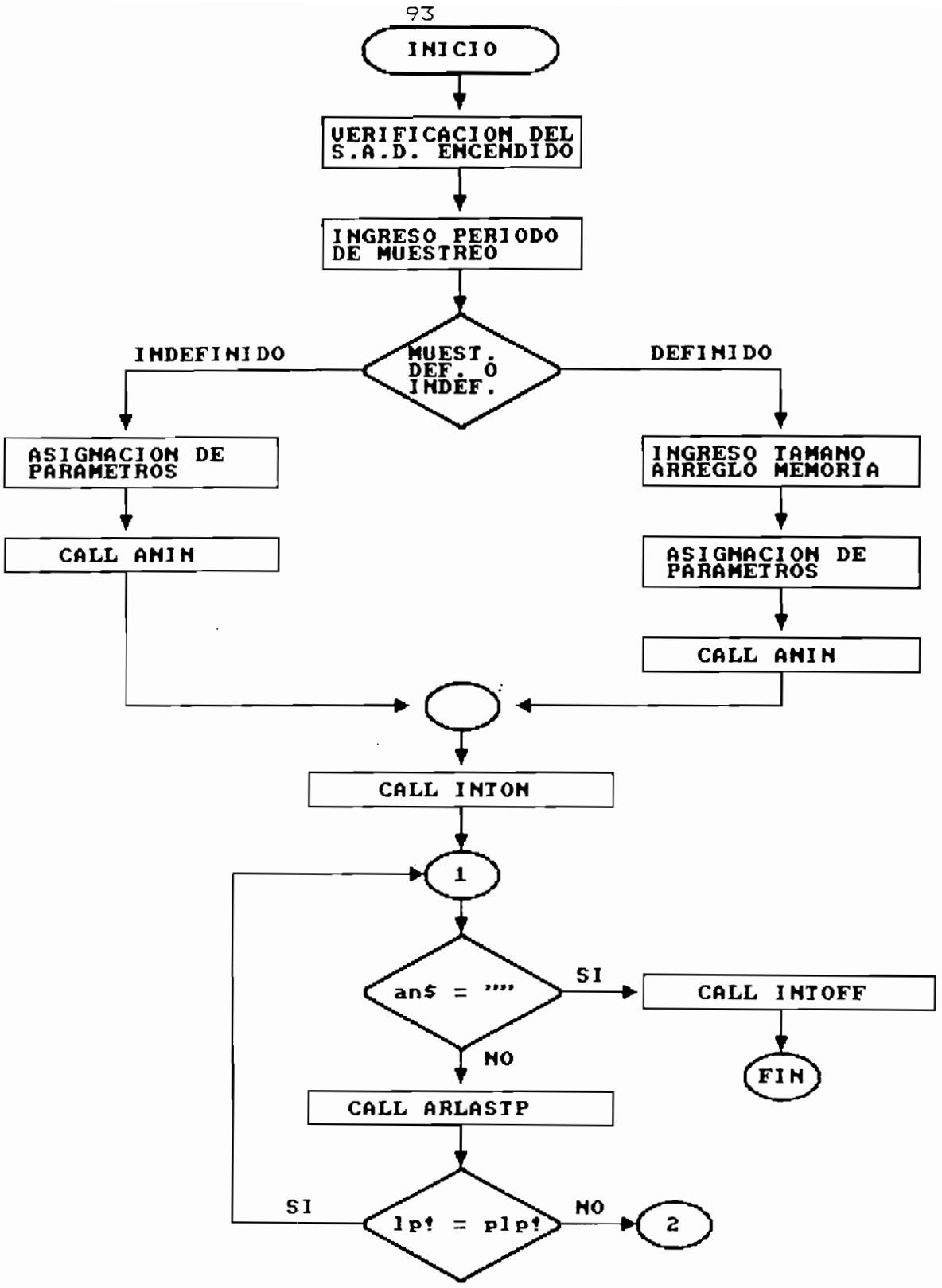
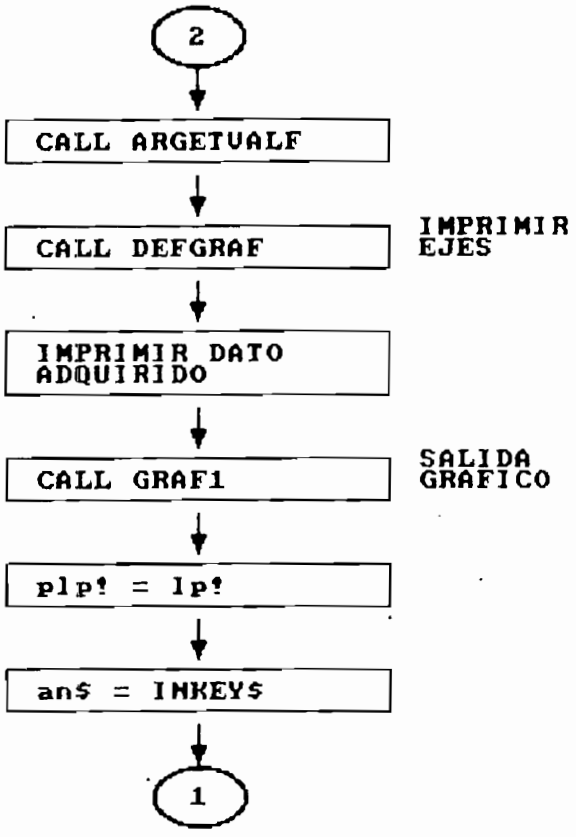


DIAGRAMA DE FLUJO DE LA SUBRUTINA DE LA ADQUISICION DE DATOS
FIGURA 3.2


2
CALL ARGETVALF
CALL DEFGRAF
IMPRIMIR EJES
IMPRIMIR DATO ADQUIRIDO
CALL GRAF1
SALIDA GRAFICO
plp! = lp!
an$ = INKEY$
1

3.2.2. Salida de datos análogos en Background.-

El objetivo es sacar datos a través de un solo canal análogo en background. Así también a pesar de ser sencilla será de gran utilidad para los próximos programas.

Por tratarse de una rutina que realiza trabajo de fondo mediante interrupciones que se presentan a intervalos iguales de tiempo, al tiempo que se realiza salida de datos, se puede realizar otras tareas de foreground como son gráficos, algoritmos de control, etc.. Estas tareas deben ser terminadas antes de que se reciba otra interrupción.

Los comandos necesarios son:

- **CALL ARMAKE.-** Crea un arreglo de memoria con el formato Quick500. Dicho arreglo puede ser de cuatro tipos: de bit, byte, palabra o un arreglo en tiempo real. Todo arreglo tiene dos dimensiones ancho y profundidad. El ancho se relaciona con el número de canales que pueden ser accesados y la profundidad es el número de valores adquiridos por cada canal.

```
CALL ARMAKE(arn$,dep!,wid%,ion1$)
```

arn$ (string-entrada).- Asigna un nombre al arreglo creado por ARMAKE, este nombre debe ser de ocho caracteres máximo,

más el caracter que indica el tipo de arreglo que se crea. Si no se especifica este caracter, el arreglo creado será real.

dep! (real-entrada).- Define la profundidad o longitud del vector.

wid% (integer-entrada).- Define el ancho del arreglo creado. Cuando se utiliza el parámetro **ion1$**, el parámetro **wid%** se ingresa con -1.

ion1$ (string-entrada).- Se utiliza como alternativa del parámetro **wid%** para definir el ancho del arreglo. Si no se utiliza se ingresa un string nulo "".

- **CALL ANOUT**.- Rutina que toma los valores de un arreglo previamente creado y lo saca a través de canales de salida análogos. El comando ANOUT saca valores a intervalos constantes pero no crea un área del Quick500, en la que se guarda los valores a ser enviados a través del conversor D/A. Este arreglo debe ser creado de antemano por ARMAKE, ANIN, etc. Puede ejecutarse por si solo o ser disparado por otro comando del Quick 500.

CALL ANOUT(arn$, ion1$, bintv%, cy%, tm$, bfn$)

arn$ (string-entrada).- Nombre del arreglo accesado por el comando ANOUT. Nótese que el comando ANOUT no crea el arreglo

de memoria, utiliza uno que debe ser creado anteriormente.

ion1$ (string-entrada).- Inidica los nombres de los canales de salida que serán utilizados por el comando ANOUT. Los nombres de cada canal se separan por espacios y/o comas. El número de canales utilizado especifica el ancho del arreglo de memoria creado.

bitnv% (integer-entrada).- Especifica cada que número de períodos de interrupción se realiza el muestreo de los canales habilitados.

cy% (integer-entrada).- Especifica el número de veces que se debe repetir esta tarea de background, si se asigna cy% = -1, esta tarea se ejecuta de manera indefinida.

tm$ (string-entrada).- Modo de disparo, ANOUT, soporta los modos de disparo:

"NT" No trigger	Ejecución inmediata
"WBT" Wait background trig	Espera disparo de background
"WGO" Wait for GONOW	Espera disparo de GONOW
"BT" Background trigger	Actúa como disparador de background

Se utiliza la opción de ejecución inmediata (NT).

bfn$ (string-entrada).- Sirve para nombrar las tareas ejecutadas, si no se utiliza debe ser ingresado un string nulo "".

- **CALL ARPUTVALF.**- Toma valores de una variable BASIC y los guarda en un arreglo Quick500. Guarda valores en una variable real y puede expresarse en el tipo de unidades más convenientes.

CALL ARPUTVALF(arn$,dep!,wid%,ion$,va!,euf%)

arn$ (string-entrada).- Indica el arreglo en que se almacenan los datos, este arreglo es creado anteriormente, mediante los comandos ARMAKE o ANIN.

dep! (real-entrada).- Inidice de profundidad del dato dentro del arreglo. El máximo valor permisible es la longitud del arreglo.

wid% (integer-entrada).- Indica la ubicación del dato buscado respecto de la anchura del arreglo, cuando se utiliza **ion$** no se utiliza **wid%**, en cuyo caso se asigna el valor de -1.

ion$ (string-entrada).- Se utiliza con el mismo propósito que el parámetro **wid%**, indicando el nombre de un canal de entrada. Si se va a utilizar **wid%**, **ion$** debe ingresarse como un string nulo "".

va! (real-entrada).- variable BASIC real que contiene el valor que se guarda en el arreglo Quick500, en cualquier tipo de unidades.

euf% (integer-entrada).- Especifica el tipo de unidades en que se almacenará el valor tomado de la variable BASIC, el 0 asigna a voltios, 1 a milivoltios, 2 microvoltios.

El diagrama de flujo de la salida de datos análogos unicanal en background se presenta en la figura 3.3

3.2.3 Adquisición de datos + Algoritmo + Salida de datos.-

Es una conjunción de las dos rutinas mencionadas anteriormente mas un algoritmo, el mismo que no es más que una multiplicación por dos de los datos de entrada.

En la figura 3.4 se presenta el diagrama secuencial de tareas de la rutina de adquisición y salida de datos en background, nótese que los procesos de tiempo real en background no dependen del algoritmo introducido.

Las variables tim(): timer y DT(): conteo de timer, se definen para hacer trabajar uno de los ocho timers con los que cuenta el sistema. Este timer es utilizado aquí para detectar el ritmo de las interrupciones de background. Su

función es detectar si el período de muestreo es insuficiente para que ejecute todo el lazo.

Dado que el timer tiene un límite de conteo, se ha añadido una pequeña subrutina para reciclar el timer.

```
dt1 = 1
CALL TIMERSTART(tim(),"nt","timer0")
DO
CALL TIMERREAD(dt())
LOOP UNTIL dt(0) / bintv% > dt1
dt1 = INT(dt(0) / bintv% + 1)
WEND
```

Para visualizar las respuestas de los algoritmos mencionados se ha tomado un circuito de primer orden RC, donde:

- Para el software de adquisición de datos se tiene la carga de un condensador en un circuito RC de primer orden, su aplicación se muestra en la figura 3.5.

- Para el software de salida de datos se tiene la generación de una onda sinusoidal, su aplicación se muestra en la figura 3.6, donde la función es SENO(2πnt)(v.); n es el número de puntos y t es el período de la onda.

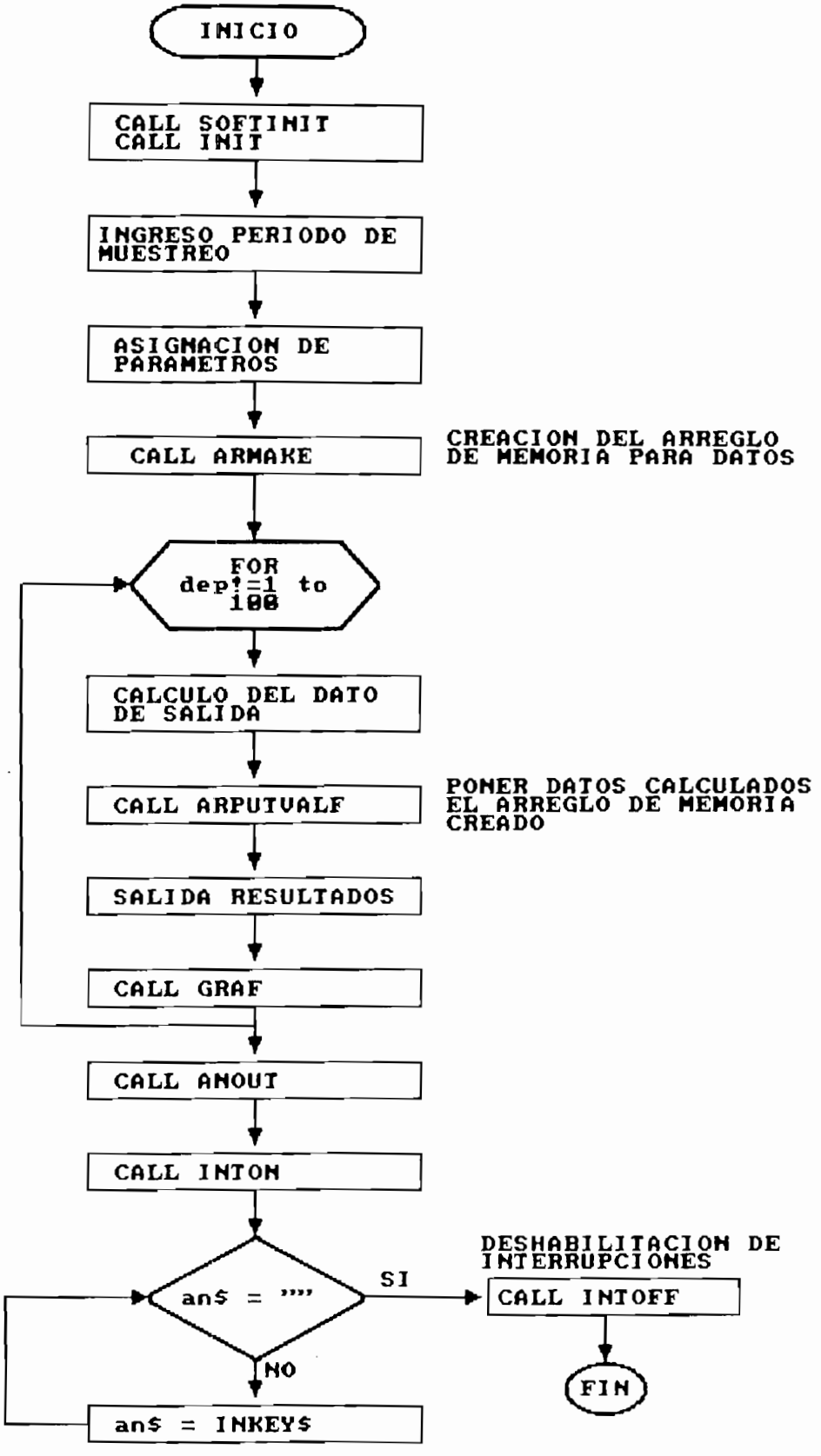


DIAGRAMA DE FLUJO DE LA SUBRUTINA DE LA SALIDA DE DATOS
FIGURA 3.3

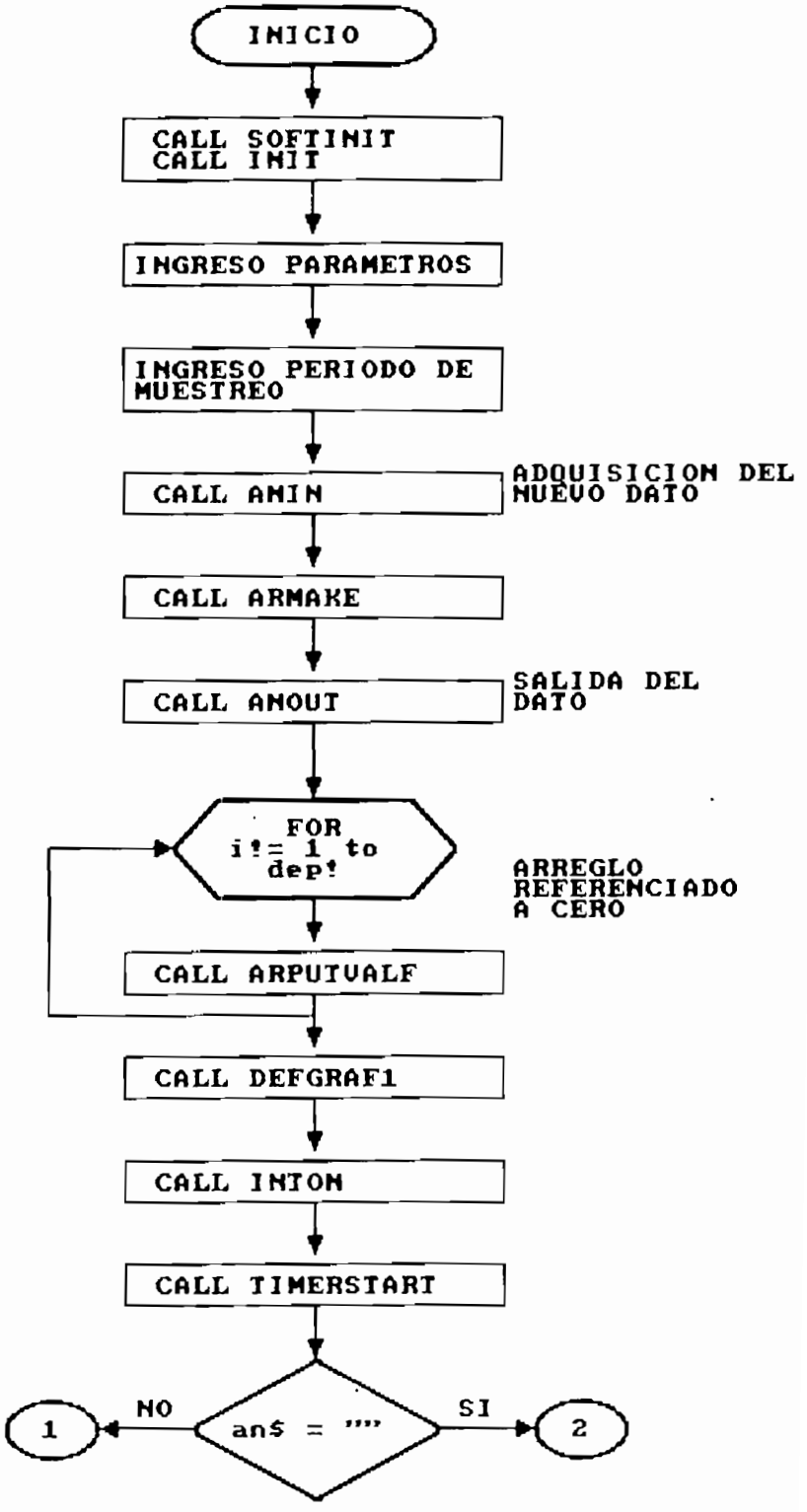
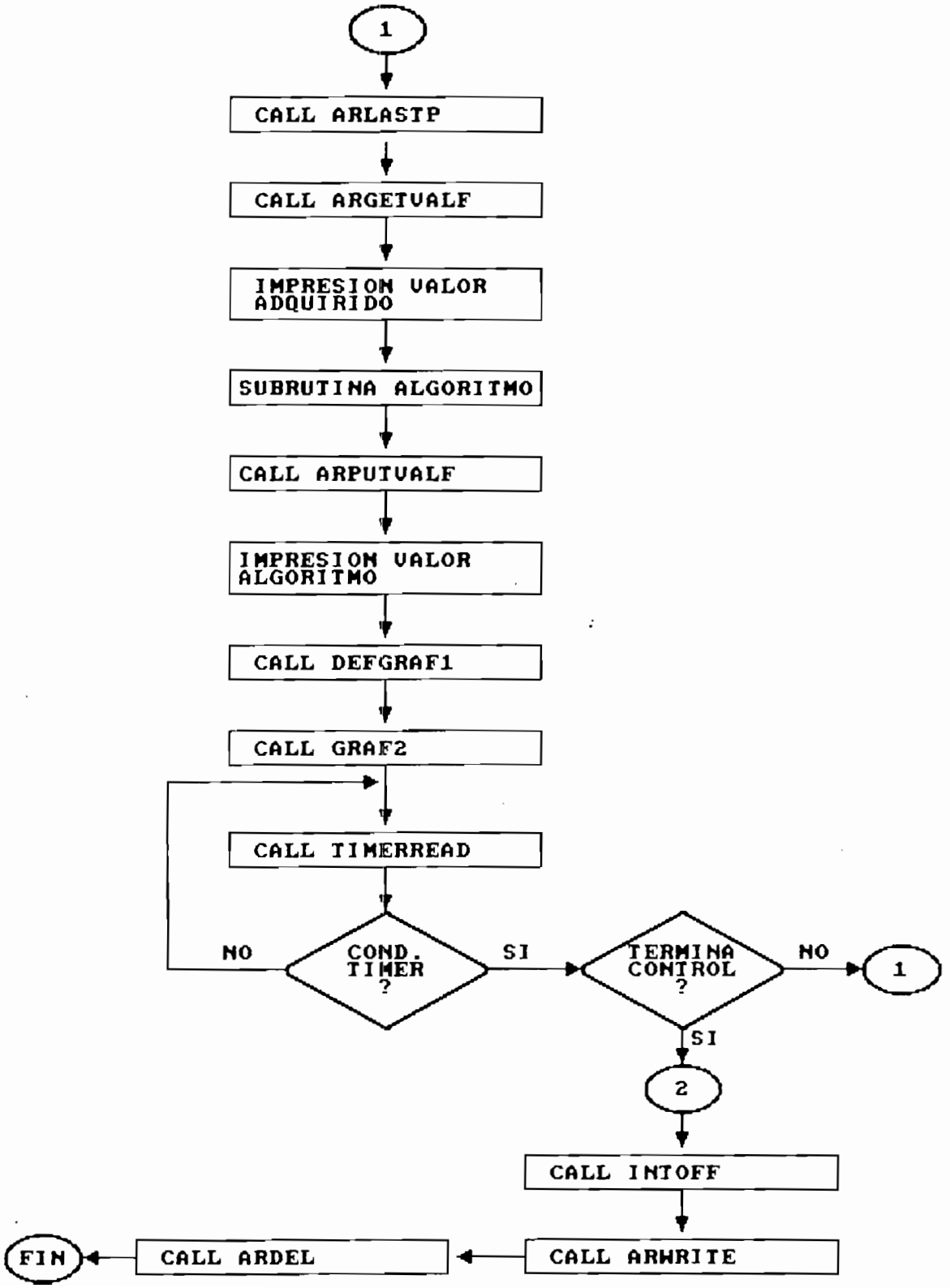


DIAGRAMA DE FLUJO DE LA SUBRUTINA ALGO
FIGURA 3.4


1
CALL ARLASTP
CALL ARGETVALF
IMPRESION VALOR ADQUIRIDO
SUBRUTINA ALGORITMO
CALL ARPUTVALF
IMPRESION VALOR ALGORITMO
CALL DEFGRAF1
CALL GRAF2
CALL TIMERREAD
NO
COND. TIMER ?
SI
TERMINA CONTROL ?
NO
1
SI
2
CALL INTOFF
CALL ARWRITE
CALL ARDEL
FIN

- Para el caso donde se incluye el algoritmo, se tiene que los datos de entrada (adquisición) son el voltaje de salida del circuito (Vout) y los datos de salida son el voltaje de salida del circuito multiplicado por 2. Esta aplicación se muestra en la figura 3.7

GRAFICO CON DATOS OBTENIDOS SUBRUTINA ENTRADA
FIGURA 3.5

GRAFICO CON DATOS OBTENIDOS SUBRUTINA SALIDA
FIGURA 3.6

ENTRADA Y SALIDA DE DATOS (Q.)

DIAGRAMA CON DATOS OBTENIDOS SUBRUTINA ALGO
FIGURA 3.7

3.3 CONTROL DE SISTEMAS

Para control en tiempo real, se tiene el programa CONREAL, que está integrado por dos rutinas: un control del tipo PID, en el cual se encuentran los valores de entrada a la planta (control) y salida de la planta (respuesta) incluido un despliegue gráfico. Finalmente se incluye una rutina general de ingreso de datos de un controlador discreto mediante ecuaciones de diferencias, el cual puede ser diseñado en forma discreta o puede ser diseñado en forma continua, del tipo redes de compensación y luego discretizado.

3.3.1 Control PID discreto.-

Un controlador PID implementado en tiempo real corresponde a un esquema de control que se representa en la figura 3.8, con sus diferentes variables.

El control proporcional-integral-derivativo (P.I.D.) actúa sobre la señal de error e(t), la misma que constituye la diferencia entre la señal de referencia r(t) y la señal de salida de la planta y(t). El control proporcional, simplemente multiplica la señal de error por la ganancia proporcional K_P; el control integral multiplica la ganancia integral K_I por la integral del error y el control derivativo, genera una señal, la cual es proporcional a la

derivada de la señal del error, aquí, se multiplica la derivada del error por la ganancia derivativa K_D. [KUO,B., 1980, pp. 509-514].

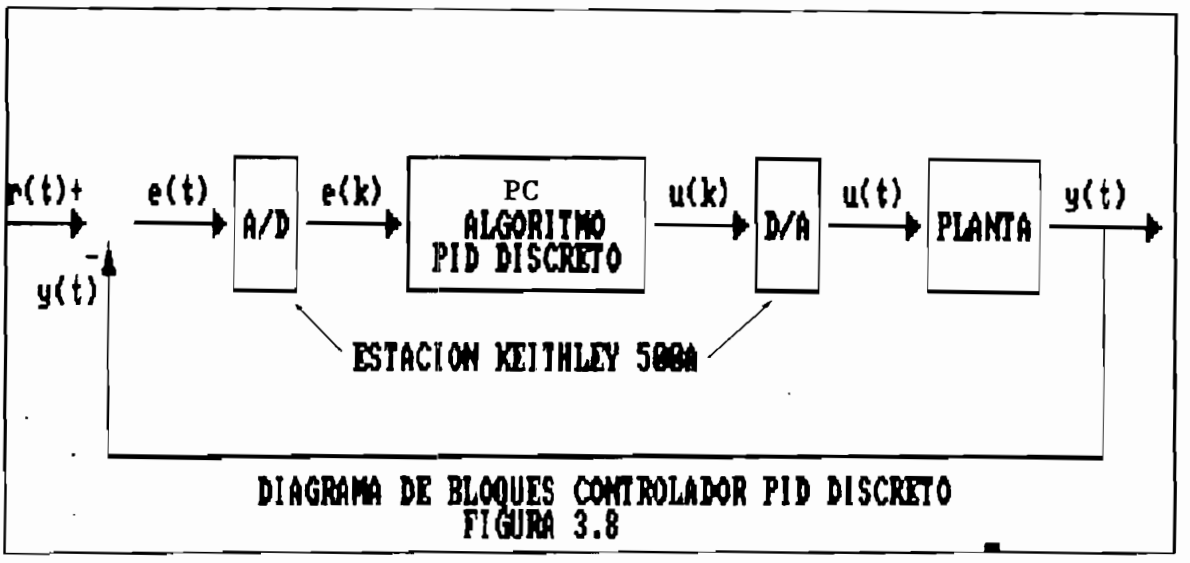


DIAGRAMA DE BLOQUES CONTROLADOR PID DISCRETO
FIGURA 3.8

La función de control integral es proveer una acción que reduce el error en estado estable, mientras que la acción derivativa provee una acción anticipatoria que mejora la respuesta transitoria.

Lo indicado se puede expresar mediante la llamada ley de control, la misma que da la relación entre el error e(t) y el control u(t) que constituyen respectivamente la entrada y la salida del controlador:

$$u(t) = K_P \times e(t) + K_I \times \int e(t) \times dt + K_D \times \frac{de(t)}{dt}$$

en la cual se tiene los siguientes parámetros:

u señal de control

e error (diferencia entre la referencia y la salida)

K_P constante proporcional

K_D constante derivativa

K_I constante integral

Al sacar la transformada de Laplace de la anterior expresión, el controlador PID responde a la ecuación:

$$U(s) = \left(K_P + \frac{K_I}{s} + K_D \times s \right) E(s)$$

Para la discretización del controlador, se toma la transformada z de la expresión anterior de la siguiente manera:

Para el integrador se utiliza el método de aproximación de una integral a la suma de los trapecios donde se tiene que:

$$u(t) = \int e(t)$$

$$u(k) - u(k-1) + \left[\frac{e(k) + e(k-1)}{2}\right] T$$

luego tomando la transformada z:

$$u(z) - u(z)\, z^{-1} + \frac{T}{2} e(z) + \frac{T}{2} e(z)\, z^{-1}$$

de donde:

$$\frac{u(z)}{e(z)} = K_I \times \frac{T(z+1)}{2(z-1)}$$

Para el derivador se utiliza la discretización mediante una aproximación de la derivada:

$$u(t) = \frac{de(t)}{dt}$$

$$u(k) = \frac{e(k) - e(k-1)}{T}$$

$$U(z) = \frac{(1 - z^{-1})}{T} E(z)$$

de donde:

$$\frac{u(z)}{e(z)} = \frac{K_D}{T} \times (1 - z^{-1})$$

En cualquier caso T es el período de muestreo. Se tiene entonces en el dominio z:

$$U(z) = \left[\frac{K_p(z-1) + K_D \frac{(z-1)^2}{Tz} + \frac{K_I T}{2}(z+1)}{(z-1)} \right]$$

Si de la expresión anterior se obtiene la transforma z inversa, se llega a la siguiente ecuación de diferencias:

$$u(k) = u(k-1) + b1 \times e(k) + b2 \times e(k-1) + b3 \times e(k-2)$$

donde:

$b1 = K_P + K_I\ T/2 + K_D/T$

$b2 = -K_P + K_I\ T/2 - 2\ K_D/T$

$b3 = K_D/T$

u(k) y **e(k)** son los valores actuales de la ley de control (salida del controlador) y el error (entrada del controlador), respectivamente.

u(k−1) y **e(k−1)** son los valores anteriores de las mismas señales.

e(k-2) es el error existente dos periodos de muestreo antes del actual.

3.3.2 Control por redes.-

Para la aplicación de control en tiempo real, mediante el diseño de redes, será necesaria la aplicación de software adicional de paquetes como el CAD CONTROL, PC-MATLAB, que ayudarán tanto para el diseño en sí de la red, como para la transformación del control a ecuación de diferencias, del tipo:

$$u(k) = -a_1 u(k-1) - a_2 u(k-2) - \ldots - a_n u(k-n) + b_0 e(k) + \ldots + b_n e(k-n)$$

de donde:

u(k) señal de control al instante k

e(k) error (diferencia entre la referencia y la salida al instante k)

a_i, b_i coeficientes de la ecuación de diferencias

Los diseños de las redes se basan en los métodos del lugar de las raíces y de respuesta de frecuencia, que se ilustraron para el caso continuo en el capítulo II.

A continuación se presenta el diagrama de flujo en la

figura 3.9, el mismo que ilustra los pasos necesarios a seguir para implementar el control en tiempo real. Entre el caso PID y REDES varía solo el algoritmo de control, en donde se aplica paso a paso las ecuaciones desarrolladas en los numerales anteriores. Así para el caso PID será necesario el ingreso de las constantes para proceder con el cálculo; mientras que para el caso de REDES se ingresará los coeficientes a_i, b_i de la ecuación de diferencias de la red.

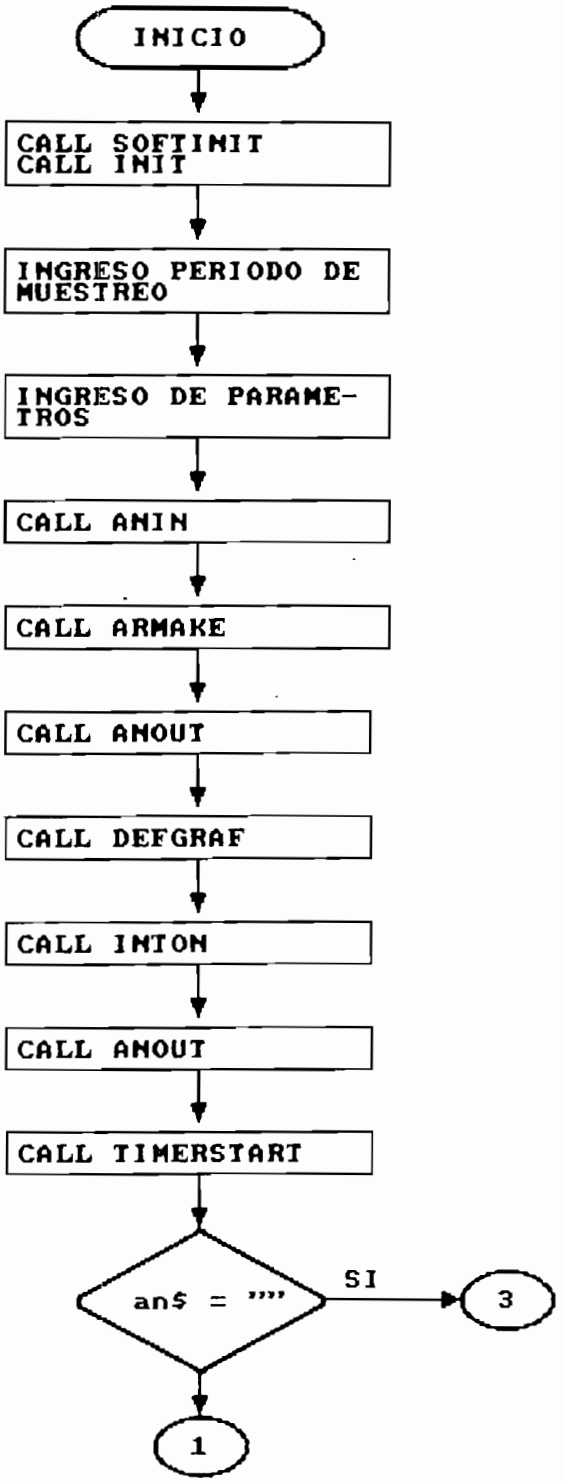
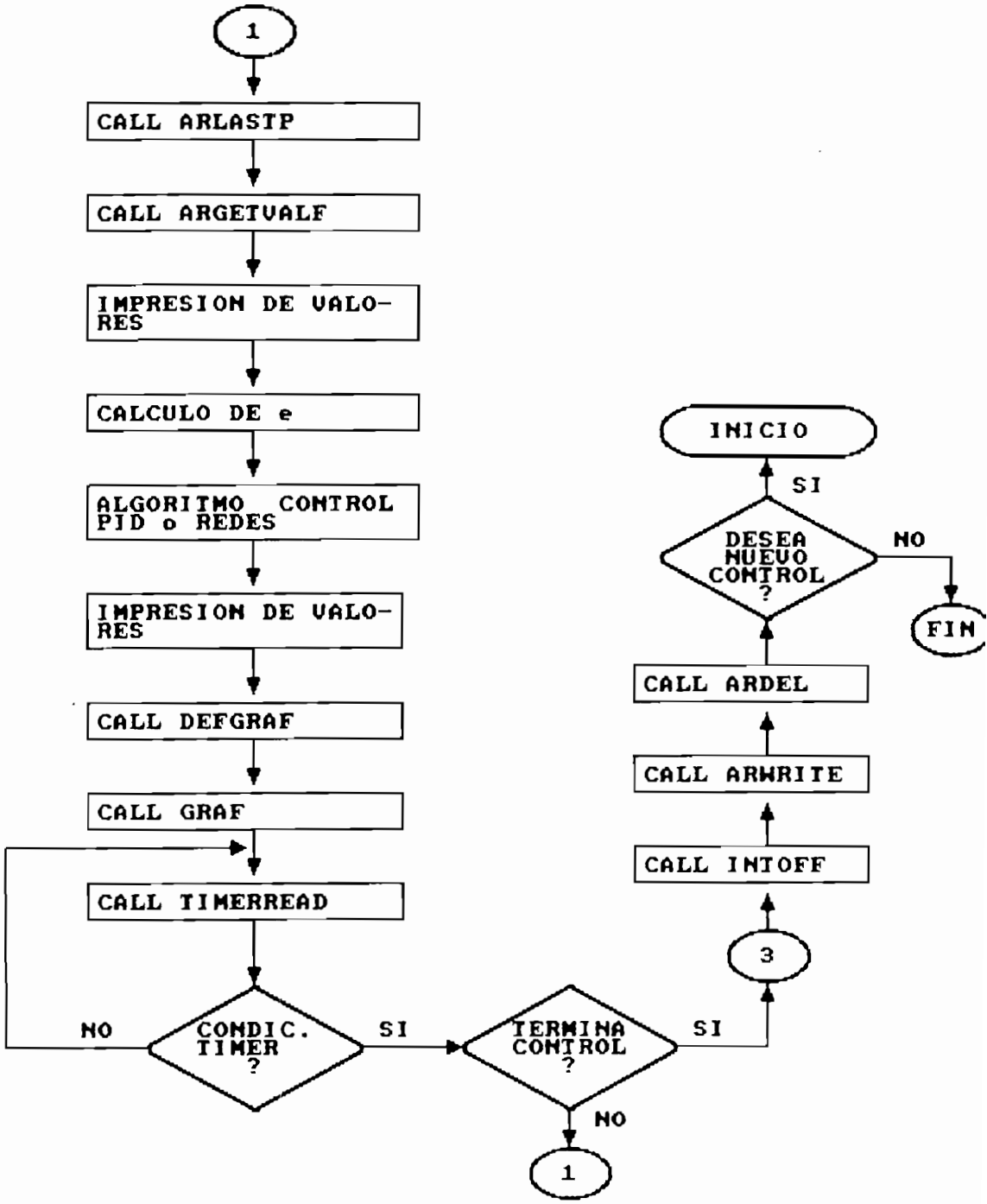


DIAGRAMA DE FLUJO DE LOS ALGORITMOS DE CONTROL EN TIEMPO REAL
FIGURA 3.9


1
CALL ARLASTP
CALL ARGETVALF
IMPRESION DE VALO-RES
CALCULO DE e
ALGORITMO CONTROL PID o REDES
IMPRESION DE VALO-RES
CALL DEFGRAF
CALL GRAF
CALL TIMERREAD
NO
CONDIC. TIMER ?
SI
TERMINA CONTROL ?
SI
NO
1
3
CALL INTOFF
CALL ARWRITE
CALL ARDEL
DESEA NUEVO CONTROL ?
SI
NO
INICIO
FIN

3.4 IDENTIFICACION DE SISTEMAS

La identificación de sistemas es la técnica de obtener modelos en base de alguna forma de análisis de datos.

La identificación según Zadeh, es definida como la determinación de un modelo analítico en base de entrada y salida de un sistema. (LEIGH, 1983, pag XI).

3.4.1 Simulación.-

La identificación consiste en la obtención de un modelo matemático que caracteriza la dinámica de la planta y puede predecir el comportamiento de la misma. Dentro del grupo de las formas canónicas, existe un modelo muy apropiado para la identificación llamado ARMA (Auto Regressive Moving Average), que consiste en una ecuación de diferencias en términos de las entradas y salidas del sistema. La salida se expresa como una combinación lineal de las salidas y las entradas anteriores, y se expresa de la siguiente manera [FRANKLIN BENJAMIN, Digital Control Systems, E.E.U.U., 1980]:

$$y(t) = -\sum_{j=1}^{n} a_j y(t-j) + \sum_{j=1}^{n} b_j u(t-j)$$

para $t \geq 0$.

En el plano z:

$$G(z) = \frac{y(z)}{u(z)} = \frac{B(z)}{1+A(z)}$$

donde:

$$A(z) = a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n^{-n}$$

$$B(z) = b_1 z^{-1} + b_2 z^{-2} + \ldots + b_n z^{-n}$$

El modelo puede ser expresado como el producto del vector de estados x(t) y de un vector de parámetros $\boldsymbol{\theta}(t)$.

$$y(t) = X(t)\boldsymbol{\theta}(t)$$

donde:

$$X(t) = [-y(t-1) - y(t-2) - \ldots - y(t-n)\, u(t-1) \ldots u(t-n)]$$

$$\boldsymbol{\theta}(t) = [a_1 a_2 \ldots a_n b_1 b_2 \ldots b_n]^T$$

El resultado de la estimación será un vector $\hat{\boldsymbol{\theta}}(t)$ de parámetros estimados que deberá ser muy cercano, sino igual al vector $\boldsymbol{\theta}(t)$ de parámetros verdaderos. El reemplazo de $\hat{\boldsymbol{\theta}}(t)$ por $\boldsymbol{\theta}(t)$ conlleva a un error que es necesario determinar y minimizar. La forma más elemental de medir este error es

encontrando la diferencia entre ambos: $\boldsymbol{\theta}(t) - \hat{\boldsymbol{\theta}}(t)$; pero esto no es posible al no conocer $\boldsymbol{\theta}(t)$, es decir los parámetros verdaderos. Se hace necesario encontrar un método que permita obtener el error a partir de los datos conocidos.

Para determinar el error de la ecuación se parte de la ecuación que describe el sistema, es decir de la expresión mediante variables de estado. Se supone además que es posible medir todos los estados, sus derivadas (o sus valores siguientes discretos) y las señales de entrada. Entonces, se calcula el error como la diferencia entre los estados reales $x(t)$ y los estados $\hat{x}(t)$ resultado de la aplicación en el modelo de los parámetros estimados $\hat{\boldsymbol{\theta}}(t)$.

Entonces la ecuación de estado es:

$$\boldsymbol{x}(t+1) = f(\boldsymbol{x}(t+1), \boldsymbol{\theta}(t))$$

para el caso lineal:

$$\boldsymbol{X}(t+1) = \boldsymbol{\phi}\boldsymbol{X}(t) + \boldsymbol{\Gamma}\boldsymbol{u}(t)$$

donde $\boldsymbol{\phi}$ es una función del vector de parámetros $\boldsymbol{\theta}(t)$:

$$\boldsymbol{\phi} = \begin{bmatrix} -a_1 & -a_2 & \dots & -a_n & b_1 & b_2 & \dots & b_n \\ 1 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} \\ 0 \\ 0 \\ \cdots \\ 1 \\ 0 \\ \cdots \\ 0 \end{bmatrix}$$

x es el vector de estado

$$x = [-y(t-1) - y(t-2) \ldots - y(t-n)\, u(t-1)\, u(t-2) \ldots u(t-n)]$$

$$X = [x_1, x_2, \ldots x_n \ldots x_{2n}]$$

Luego el error de la ecuación se define por:

$$E(t;\Theta) = X(t+1) - \hat{X}(t+1) = X(t+1) - \Phi X(t) - \Gamma u(t)$$

Resolviendo se obtiene:

$$e_1(t;\Theta) = x_1(t+1) - x^T(t)\Theta(t)$$

$$e_1(t;\Theta) = y(t) - \hat{y}(t)$$

donde:

$\Theta(t)$ representa ahora los valores estimados de los parámetros.

donde:

$\hat{y}(t)$ es el valor estimado de $y(t)$ con los parámetros $\hat{\boldsymbol{\theta}}(t)$.

también:

$$e_2(t;\boldsymbol{\theta}) = x_2(t+1) - x_1(t)$$

$$e_2(t;\boldsymbol{\theta}) = -y(t-1) + y(t-1) = 0$$

De la misma manera, para $2 < i \leq n$

$$e_i(t;\boldsymbol{\theta}) = 0$$

Por lo tanto el error de la ecuación se readuce a:

$$E(t;\boldsymbol{\theta}) = [e_1, 0 \ldots\ldots 0]^T$$

$$e_1(t;\boldsymbol{\theta}) = y(t) - \hat{y}(t)$$

Si se busca los mejores valores de $\boldsymbol{\theta}(t)$, será necesario minimizar la expresión siguiente:

$$J(\boldsymbol{\theta}) = E^T E = \sum e_j(k;\boldsymbol{\theta}) = (e_1(k;\boldsymbol{\theta}))^2$$

El error de ecuación es por tanto fácilmente adaptable al modelo ARMA pues los estados del sistema corresponden justamente a los valores de entrada y salida medidos. Esto hace posible utilizar este tipo de error para la estimación de los parámetros de la planta.

La minimización de $J(\boldsymbol{\theta})$ se realizará mediante el método de mínimos cuadrados.

Mínimos cuadrados ordinarios:

Una de las forma más utilizadas para conseguir un buen $\boldsymbol{\theta}(t)$ estimado, es decir una minimización de $J(\boldsymbol{\theta})$ es el método de mínimos cuadrados. Para la aplicación de este método se parte de un modelo ARMA donde se miden las entradas y las salidas y de la expresión de error dada por el error de ecuación.

$$e_1(t) = e(t;\boldsymbol{\theta}) = y(t) - x(t)\boldsymbol{\theta}(t)$$

donde:

$$x = [y(t-1)y(t-2)\ldots y(t-n)u(t-1)\ldots u(t-n)]$$

Para t mediciones se tiene:

$$e(0) = y(0) - x(0)\,\theta(t)$$

$$e(1) = y(1) - x(1)\,\theta(t)$$

$$e(2) = y(2) - x(2)\,\theta(t)$$

$$\ldots \quad \ldots \quad \ldots\ldots$$

$$e(t) = y(t) - x(t)\,\theta(t)$$

Con estos valores medidos, se contruyen la matriz de datos X(t) y el vector de datos Y(t) para t mediciones, de la forma:

$$X(t) = [x(0)\,x(1)\,x(2)\ldots x(t)]^T$$

$$Y(t) = [y(0)\,y(1)\,y(2)\ldots y(t)]^T$$

Según lo estudiado anteriormente, se puede generalizar para los diferentes valores de t, con:

$$E(t) = Y(t) - X(t)\,\theta(t)$$

donde:

$$E(t) = [e(0)\,e(1)\,e(2)\ldots e(t)]^T$$

Con esta expresión general del error es posible entrar a la minimización por medio del método de mínimos cuadrados:

$$J(\theta) = \sum (e(k;\theta))^2$$

$$J(\theta) = E^T E$$

$$J(\theta) = (Y(t) - X(t)\theta(t))^T (Y(t) - X(t)\theta(t))$$

Lo que más sencillamente corresponde a:

$$J = (Y - X\theta)^T (Y - \theta)$$

La minimización se realiza igualando a cero la derivada de J respecto a θ y resolviendo la ecuación correspondiente:

$$\frac{dJ}{d\theta} = \frac{d}{d\theta} [Y^T Y - Y^T X\theta - \theta^T X^T Y - \theta^T x^T X\theta] = 0$$

Usando las reglas de derivación siguientes:

$$\frac{d}{dx} [y^T x] = y$$

$$\frac{d}{dx} [x^T y] = y^T$$

se obtiene:

$$\frac{dJ}{d\theta} = -Y^T X - Y^T X - 2\theta^T X^T X$$

entonces:

$$\theta = (X^T X)^{-1} X^T Y$$

que es la solución general del problema.

Evidentemente esta solución existe si, por un lado, el modelo escogido es del tipo ARMA o de tipo canónico obteniéndose un único valor de θ, es decir que el sistema es "identificable" y si, por otro lado, la matriz X^TX no es singular de manera que se pueda calcular su inversa. Esta última propiedad se verifica cuando la señal u(k) es de "excitación persistente".

La señal de entrada u(t) puede adoptar muchas formas. Para el caso que aquí interesa, se requiere que u(t) sea de excitación persistente de manera que el algoritmo de mínimos cuadrados converja. Esto se consigue de mejor manera, cuando u(t) es una señal aleatoria, debido a que no se tiene una dependencia lineal en los valores de la secuencia.

$$x(t) = [y(t-1)y(t-2)\ldots y(t-n)u(t-1)\ldots u(t-n)]$$

Esto se consigue generando una señal aleatoria de ruido blanco que se añadirá a la señal u(t), de tipo escalón consiguiendo de esta manera un proceso estocástico.

Hasta el momento se ha podido resolver el problema de identificación en base a escoger un cierto número de datos y aplicar los mínimos cuadrados. Para una aplicación real esto significará ir obteniendo soluciones por paquetes, lo que dificulta su aplicación. Esto se resuelve encontrando un algoritmo recursivo que vaya encontrando y actualizando las soluciones. Es lo que se conoce como Mínimos Cuadrados Recursivos.

- **Mínimos cuadrados recursivos:**

El objetivo es poder obtener un valor de $\theta(t)$ en base de su valor anterior y de una correción que se iría aplicando en cada cálculo sucesivo. Así, después de un cierto número de iteraciones θ convergería al valor buscado y la corrección tendería a cero.

$$\theta(t+1) = \theta(t) + correccion$$

Se parte del cálculo de $\theta(t+1)$ y se va a tratar de ponerlo en función de $\theta(t)$. En efecto:

$$\theta(t) = (X^T(t)X(t))^{-1}X^T(t)Y(t)$$

Luego:

$$\Theta(t+1) = (X^T(t+1)X(t+1))^{-1}X^T(t+1)Y(t+1)$$

$$\Theta(t+1) = [(X(t)\,|\,x(t+1))^T(X(t)\,|\,x(t+1))]^{-1}[X(t)\,|\,x(t+1)]\begin{bmatrix}Y(t)\\ y(t+1)\end{bmatrix}$$

resolviendo se tiene:

$$\Theta(t+1) = (X^T(t)X(t)+x^T(t+1)x(t+1))^{-1}(X^T(t)Y(t)+x^T(t+1)y(t+1))$$

Se define ahora:

$$P(t) = (X^T(t)X(t))^{-1}$$

Entonces:

$$\Theta(t) = P(t)X^T(t)Y(t)$$

$$\Theta(t+1) = P(t+1)(X^T(t)Y(t)+x^T(t+1)y(t+1))$$

Aplicando el lema de inversión de matrices se tiene:

$$P(t+1) = (P^{-1}(t)+x^T(t+1)x(t+1))^{-1}$$

Haciendo reducción de términos se llega a:

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) + L(t+1)\,\epsilon(t+1)$$

donde:

$$L(t+1) = \frac{P(t)\,x^{T}(t+1)}{1 + x(t+1)\,P(t)\,x^{T}(t+1)}$$

$$P(t+1) = [I - L(t+1)\,x(t+1)]\,P(t)$$

$$\epsilon(t+1) = y(t+1) - x(t+1)\,\boldsymbol{\theta}(t)$$

De esta manera queda definido el proceso recursivo de cálculo de los parámetros $\boldsymbol{\theta}(t)$. El algoritmo general está dado por:

1.- Dar valores para $\boldsymbol{\theta}(0)$ y $P(0)$

2.- Obtener $y(0)$; iniciar con $t = 0$

3.- Obtener $u(t)$; $y(t+1)$

4.- Actualizar $x(t+1)$

5.- Calcular $L(t+1)$; $\epsilon(t+1)$

6.- Calcular $\boldsymbol{\theta}(t+1)$; $P(t+1)$

7.- Repetir desde 3 con un nuevo valor de t

El resultado final está dado por los valores de $\boldsymbol{\theta}(t)$ al

llegarse al número de iteraciones deseado.

La inicialización del algoritmo se da con valores de $\theta(0)$ y de $P(0)$. Para iniciar convenientemente el algoritmo, los valores de $\theta(0)$ pueden ser dados según la estimación intuitiva que se haga de los parámetros reales θ. Por el contrario, los valores iniciales de $P(0)$ deben permitir que el algoritmo trabaje, es decir que se tenga una convergecia rápida de los parámetros, por ello $P(0)$ debe ser suficientemente grande, por lo que normalmente se escoge $P(0) = \alpha I$, con un valor grande de $\alpha = 10.000$.

Hasta aquí se ha trabajado con parámetros constantes θ, pero en muchos casos, existen variaciones de los parámetros de la planta. Esto tiene consecuencias especialmente al realizar control en lazo cerrado en base a los parámetros identificados. Si se desea una rápida convergencia en el cálculo de los nuevos parámetros cuando hay una variación brusca pero espaciada de los parámetros, la demora en la convergencia se puede resolver con un "reset" de la matriz covarianzas a un valor igual αI, con un valor grande de α, para así acelerar la velocidad de convergencia de los nuevos parámetros.

El caso de parámetros que varían lentamente en el tiempo puede ser resuelto con la incorporación del "factor de olvido" en el algoritmo de identificación. El factor de

olvido B es un parámetro que permite ponderar en forma exponencial los datos. El valor más reciente tendrá una ponderación de 1 mientras que el n-ésimo valor estará ponderado con B^n.

El algoritmo general será el mismo solo que con las modificaciones siguientes en el cálculo de las matrices P(t) y L(t).

$$L(t+1) = \frac{P(t)x^T(t+1)}{B+x(t+1)P(t+1)x^T(t+1)}$$

$$P(t+1) = (I - \frac{P(t)x^T(t+1)x(t+1)}{B+x(t+1)P(t)x^T(t+1)}) \frac{1}{B} P(t)$$

- Minimos Cuadrados Recursivos con ruido:

En los casos prácticos casi siempre se tienen perturbaciones de diferentes tipos, muchas de las cuales son de tipo aleatorio. Por lo que para realizar el estudio de la identificación de sistemas en presencia de este tipo de perturbaciones, se hace necesaria una modificación al modelo ARMA.

Para este caso en el modelo ARMA aparece una perturbación en la ecuación de salida. El modelo será:

$$y(t) = x(t)\theta(t) + v(t)$$

siendo x(t), θ(t) los vectores antes definidos y v(t) una perturbación en forma de ruido blanco.

Si se forma la matriz Y(t) antes mencionada, en base a t valores de y(t), la ecuación anterior será:

$$Y(t) = X|t)\theta(t) + V(t)$$

donde V(t) es un vector dado por:

$$V(t) = [v(0)\, v(1)\, v(2) \ldots v(t)]^T$$

Siguiendo el mismo método de minimización del error se llega a los mismos resultados que en el caso de mínimos cuadrados recursivos sin ruido, siempre que se considere ruido blanco.

Para los dos casos, el diagrama de flujo es muy similar el único cambio que existe es el ingreso de la señal de entrada, en la cual se añade ruido blanco con la finalidad de conseguir una señal de excitación persistente, se presenta en la figura 3.10.

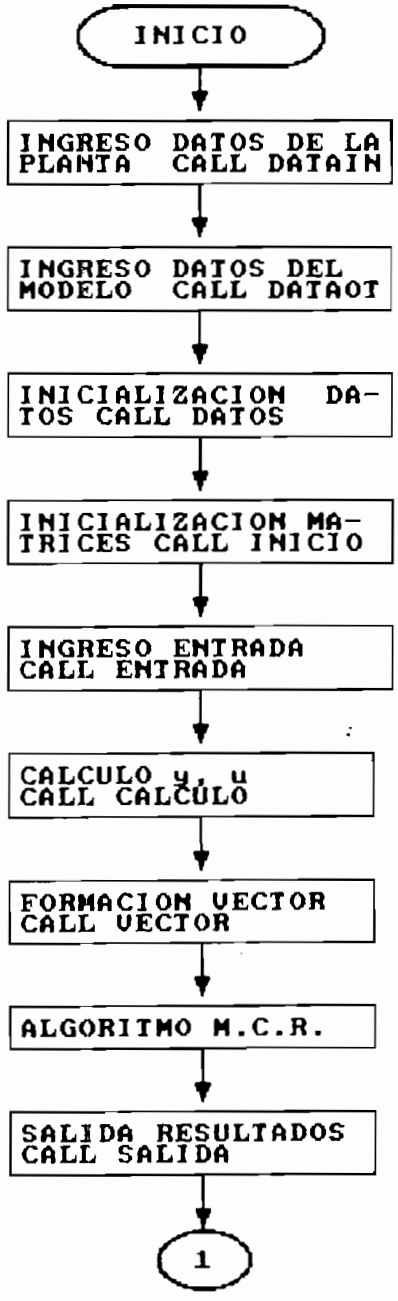
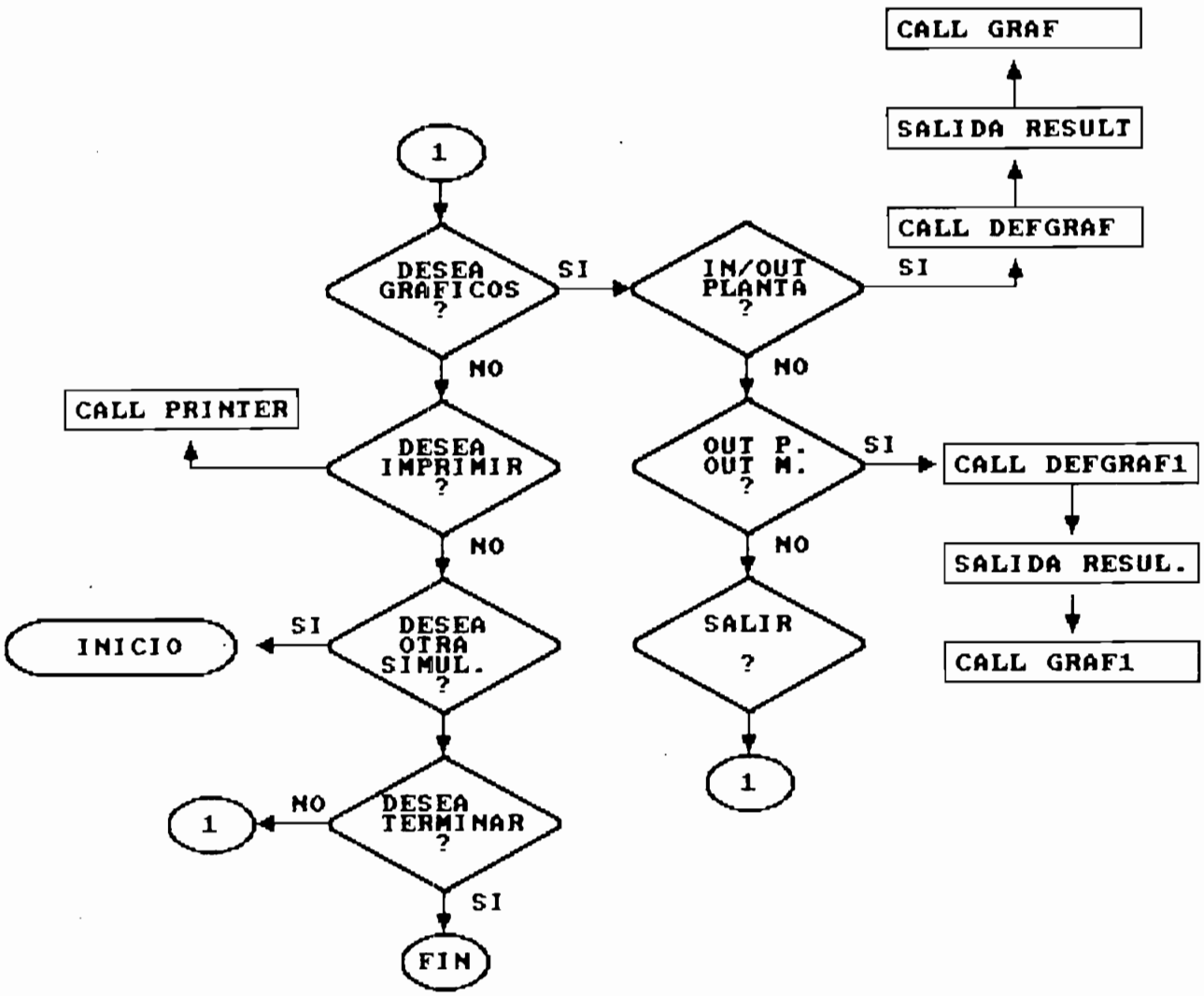


DIAGRAMA DE FLUJO DE IDENTIFICACION EN SIMULACION (IDENSIMU)
FIGURA 3.10



P. = PLANTA

M. = MODELO

3.4.2 Identificación en tiempo real.-

El proceso de identificación en tiempo real consiste fundamentalmente en la aplicación del algoritmo de identificación antes desarrollado, conjuntamente con la aplicación de las rutinas de entrada de datos del Keithley 500A. Así los datos medidos son los datos utilizados por el algoritmo para determinar los parámetros de la planta mediante el método de mínimos cuadrados recursivos. El proceso se considerará de tipo estocástico, pues se ingresará ruido blanco.

La figura 3.11 muestra una estructura básica de la identificación parámetrica en tiempo real. Es posible visualizar el muestreo de la entrada y salida de la planta y la utilización de esta información para determinar los parámetros de la misma.

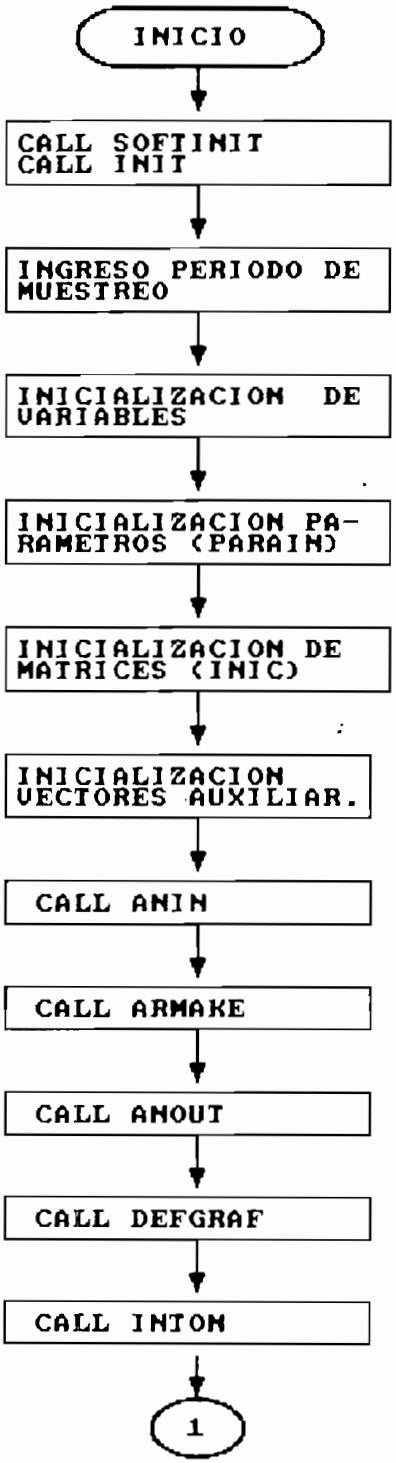
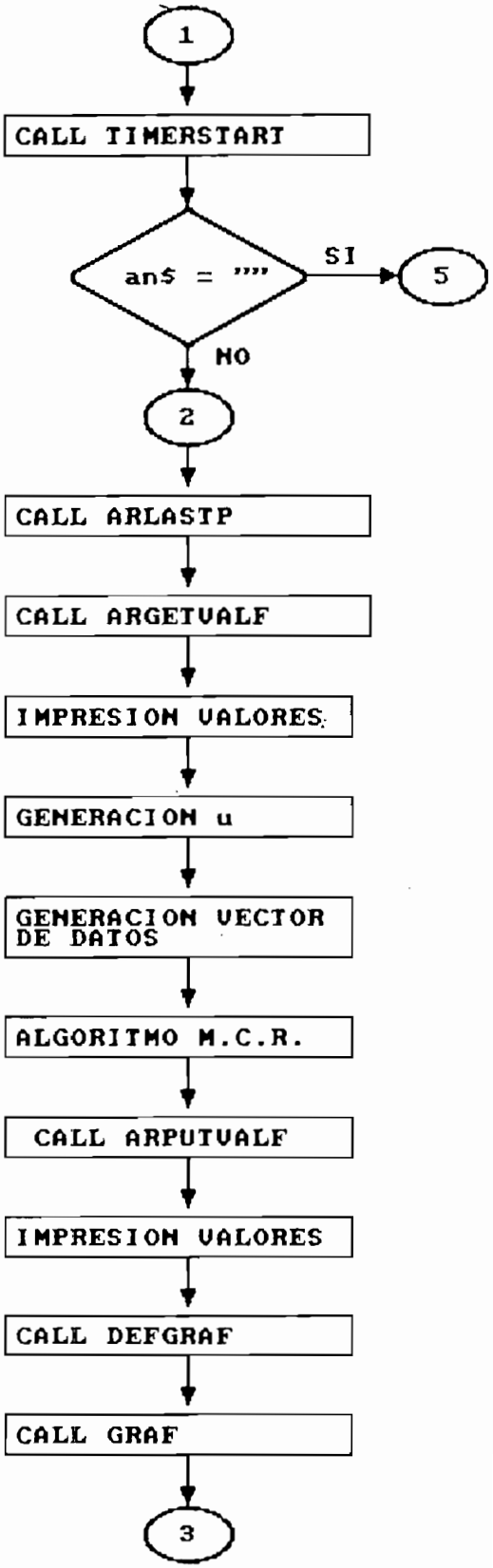
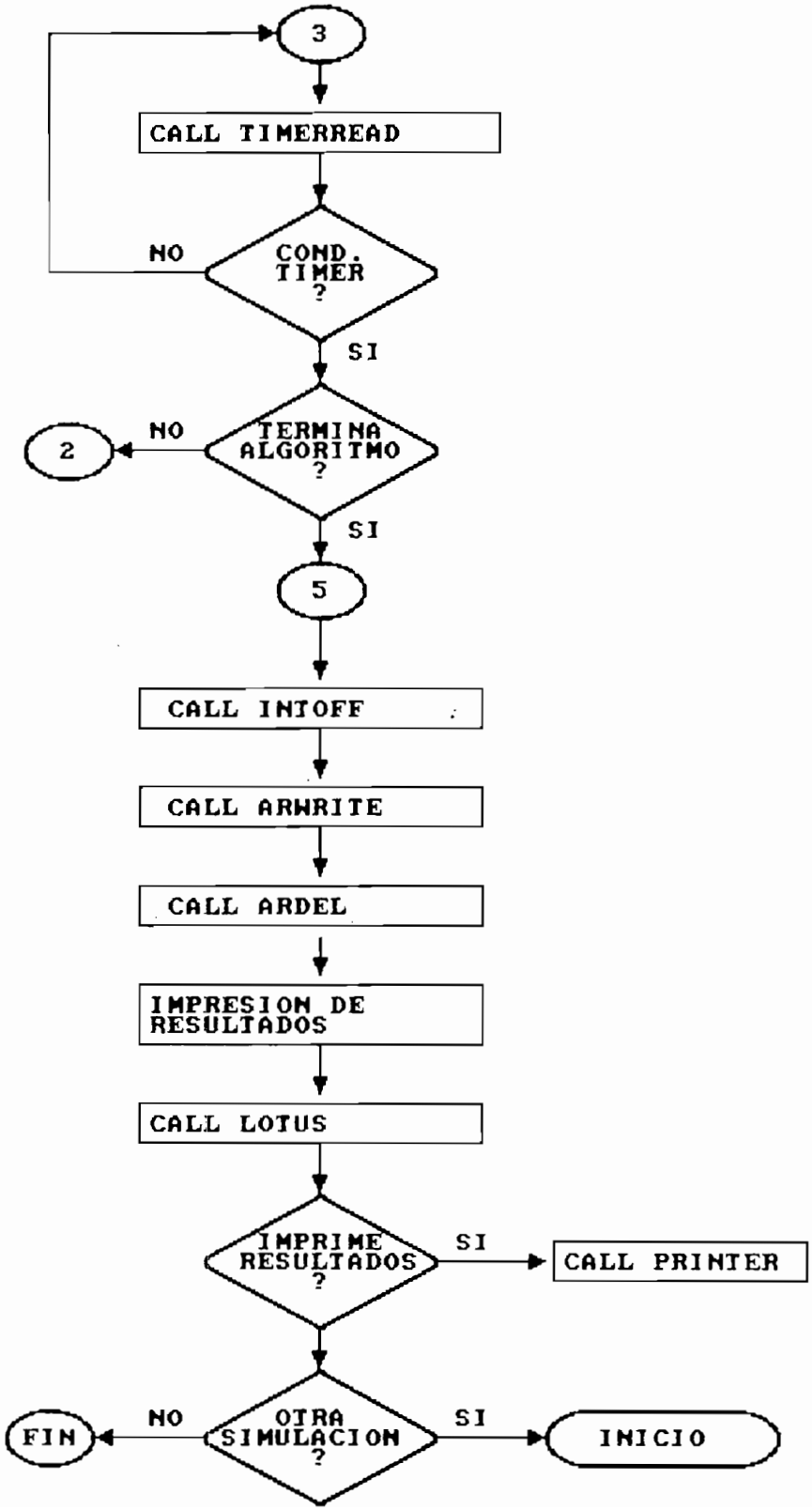


DIAGRAMA DE FLUJO DE IDENTIFICACION EN TIEMPO REAL (IDENREAL)
FIGURA 3.11


1
CALL TIMERSTART
an$ = ""
SI
5
NO
2
CALL ARLASTP
CALL ARGETVALF
IMPRESION VALORES
GENERACION u
GENERACION VECTOR DE DATOS
ALGORITMO M.C.R.
CALL ARPUTVALF
IMPRESION VALORES
CALL DEFGRAF
CALL GRAF
3


3
CALL TIMERREAD
NO
COND. TIMER ?
SI
NO
TERMINA ALGORITMO ?
2
SI
5
CALL INTOFF
CALL ARWRITE
CALL ARDEL
IMPRESION DE RESULTADOS
CALL LOTUS
IMPRIME RESULTADOS ?
SI
CALL PRINTER
NO
OTRA SIMULACION ?
SI
FIN
INICIO

4. RESULTADOS Y CONCLUSIONES

4.1. RESULTADOS

Se han realizado numerosos ejemplos en simulación y en tiempo real, tanto para análisis de sistemas continuos y discretos cuanto para la identificación de sistemas y para control mediante diferentes técnicas. Algunos de los cuales se incluyen en la presente tesis.

En los capítulos anteriores se incluyen algunos de los ejemplos mencionados, en este capítulo se presentan tres casos de estudio que corresponden a tres circuitos, uno de primer orden RC pasivo, uno de primer orden activo y uno de segundo orden subamortiguado activo.

Sobre estos sistemas se realiza la modelación mediante las técnicas convencionales, su simulación discreta con propósito de análisis, su identificación y el control mediante diferentes técnicas, tanto en simulación como en tiempo real.

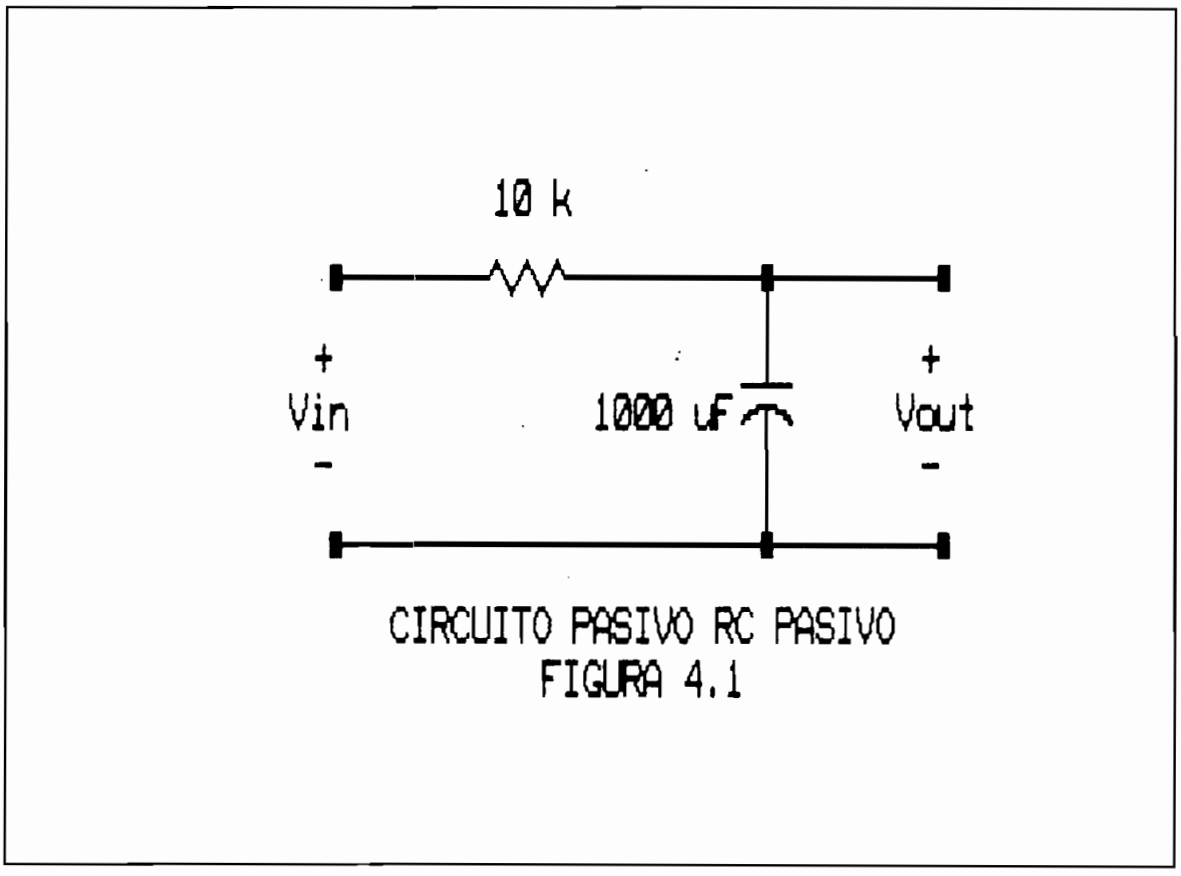
Se escogieron estos ejemplos por su aplicación didáctica a prácticas de laboratorio, implantándose en circuitos impresos los circuitos mencionados con la posibilidad de la variación de sus parámetros.

A continuación se detalla cada caso de estudio.

4.1.1. CIRCUITO RC PASIVO DE PRIMER ORDEN

4.1.1.1. Modelación y Simulación Discreta.-

La figura 4.1 muestra un circuito de primer orden pasivo, el mismo que va hacer analizado.



CIRCUITO PASIVO RC PASIVO
FIGURA 4.1

de donde:

$$G(s) = \frac{Vout}{Vin} = \frac{1}{1+sCR} = \frac{1}{10s+1}$$

Para encontrar el equivalente discreto se ha utilizado el paquete CAD CONTROL, dentro de este paquete, con el comando CONVERT, para un período de muestreo:

T = 200 ms.

$$G(z) = \frac{1.980133 * 10^{-2}}{z - .9801987}$$

luego:

$$y(k) = 1.980133 * 10^{-2} u(k-1) + .9801987 y(k-1)$$

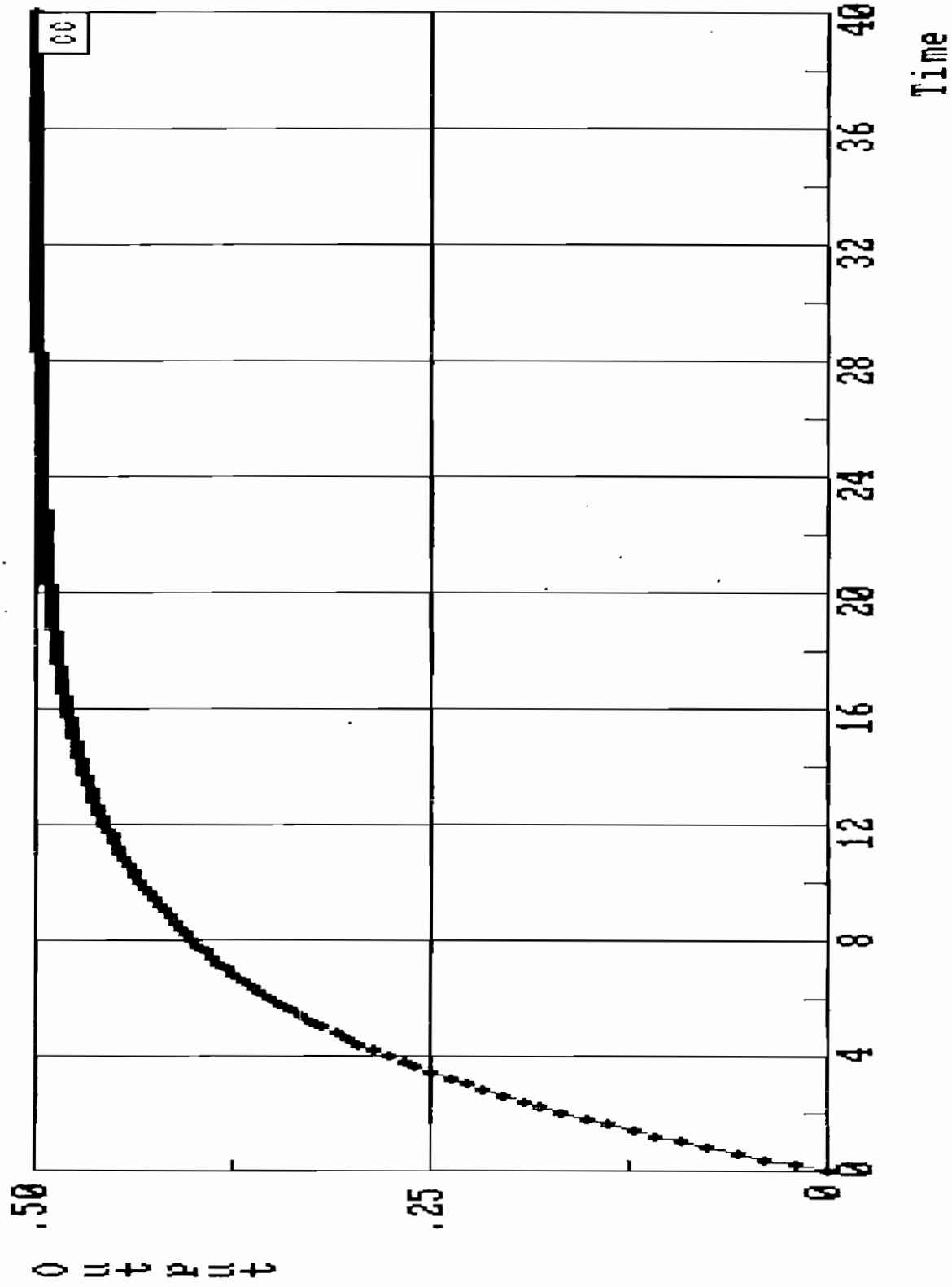
T = 1 s.

$$G(z) = \frac{0.09516259}{z - .9048374}$$

luego:

$$y(k) = 9.516259 * 10^{-2} u(k-1) + .9048374 y(k-1)$$

A continuación utilizando el mismo paquete se procede a obtener la respuesta en el tiempo del sistema discreto en lazo cerrado, para un período de muestreo de 200 ms., se muestra en la figura 4.2. Se tiene:



RESPUESTA EN EL TIEMPO DISCRETO A UNA ENTRADA PASO
FIGURA 4.2

Ep % = 50%

ts = 19.4 s.

4.1.1.2. Identificación.-

Se hace uso del programa de mínimos cuadrados recursivos para realizar la simulación, el mismo que fue desarrollado en Quick Basic con el programa IDENSIMU, obteniéndose una rápida convergencia de los parámetros, como se ve en el reporte #1.

En general, las pruebas de identificación en este trabajo consisten en la realización de la identificación en tiempo real usando el algoritmo de mínimos cuadrados recursivos, desarrollado en Quick500, el programa IDENREAL. Se pretende verificar la validez de las formulaciones teóricas. Se intentará estimar los parámetros en base a un modelo discreto, para lo cual se alimentará a la planta, con una señal de entrada u(t), la que es generada desde el computador. Esta señal u(t) puede ser igual a un escalón puro de amplitud 5 voltios ó un escalón de 2 voltios mas ruido de tipo RANDOM de amplitud 3 voltio (excitación persistente), para este caso en particular. Se tiene como resultados los que se muestran en los reportes #3 y #4.

Obteniéndose:

- Entrada sin ruido:

$$y(k) = .018821u(k-1) + .980995y(k-1)$$

RESULTADOS DE LA SIMULACION

P L A N T A

a(1) = .9801987
b(1) = 1.980133E-02

M O D E L O

a(1) = 0.9802
b(1) = 0.0198

Número de iteraciones 100

REPORTE #1

RESULTADOS DE LA SIMULACION

P L A N T A

a(1) = .9801987
b(1) = 7.128478E-02

M O D E L O

a(1) = 0.9802
b(1) = 0.0713

Número de iteraciones 100

REPORTE # 2

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(1)= 0.980995
b(1)= 0.010821

Número de iteraciones = 1002

Período de muestreo = 200 ms.

SEÑAL DE ENTRADA:
Escalón = 5
Ruido = 0

REPORTE # 3

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(1)= 0.981362
b(1)= 0.010533

Número de iteraciones = 1004

Período de muestreo = 200 ms.

SEÑAL DE ENTRADA:
Escalón = 2
Ruido = 3

REPORTE # 4

- Entrada con ruido:

$$y(k) = .018533u(k-1) + .981368y(k-1)$$

Se debe mencionar que las amplitudes tanto del escalón como del ruido dependerán de la planta, pues siempre para que el algoritmo funcione se debe evitar la saturación de la misma.

Las figuras 4.3 y 4.4 muestran los valores de la salida mediante el comando CALL ARWRITE y mediante el almacenamiento en un archivo para el procesamiento mediante LOTUS. En ambos casos se generan archivos .PRN, para el caso de la identificación con ruido. La figura 4.5 muestra el valor de entrada con ruido. Las figuras 4.6 y 4.7 la variación de los parámetros a y b. Se ha considerado un cambio de signo en los parámetros "a".

Para el caso de un período de muestreo de 1 segundo se realizó identificación obteniéndose los resultados que se muestran en el reporte 5 y las figuras 4.8 y 4.9 muestran la convergencia de los parámetros.

- Entrada con ruido:

$$y(k) = .077822u(k-1) + .920131y(k-1)$$

RESPUESTA AV DE TIMPO DE UNA PLANTA DE SEGUNDO ORDEN
CANAL ANALOGO (V.)
Tiempo (mseg)
0
0.5
1
1.5
2
2.5
3
3.5
4
0
10000
20000
30000
40000
50000
60000
70000
80000
90000
100000
110000
120000

FIGURA 4.3

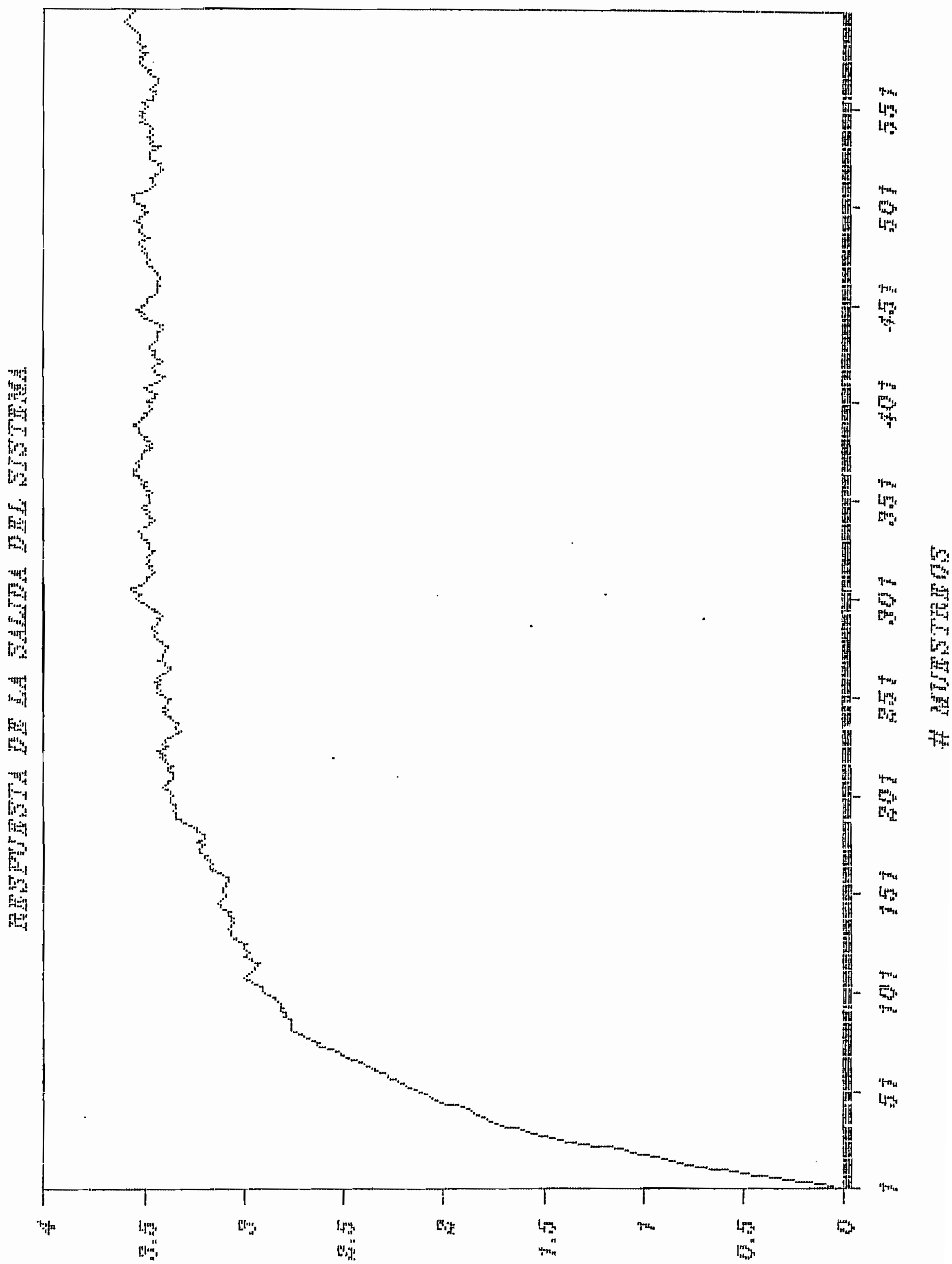

RESPUESTA DE LA SALIDA DEL SISTEMA
MUESTRAS
1
51
101
151
201
251
301
351
401
451
501
551
0
0.5
1
1.5
2
2.5
3
3.5
4

FIGURA 4.4

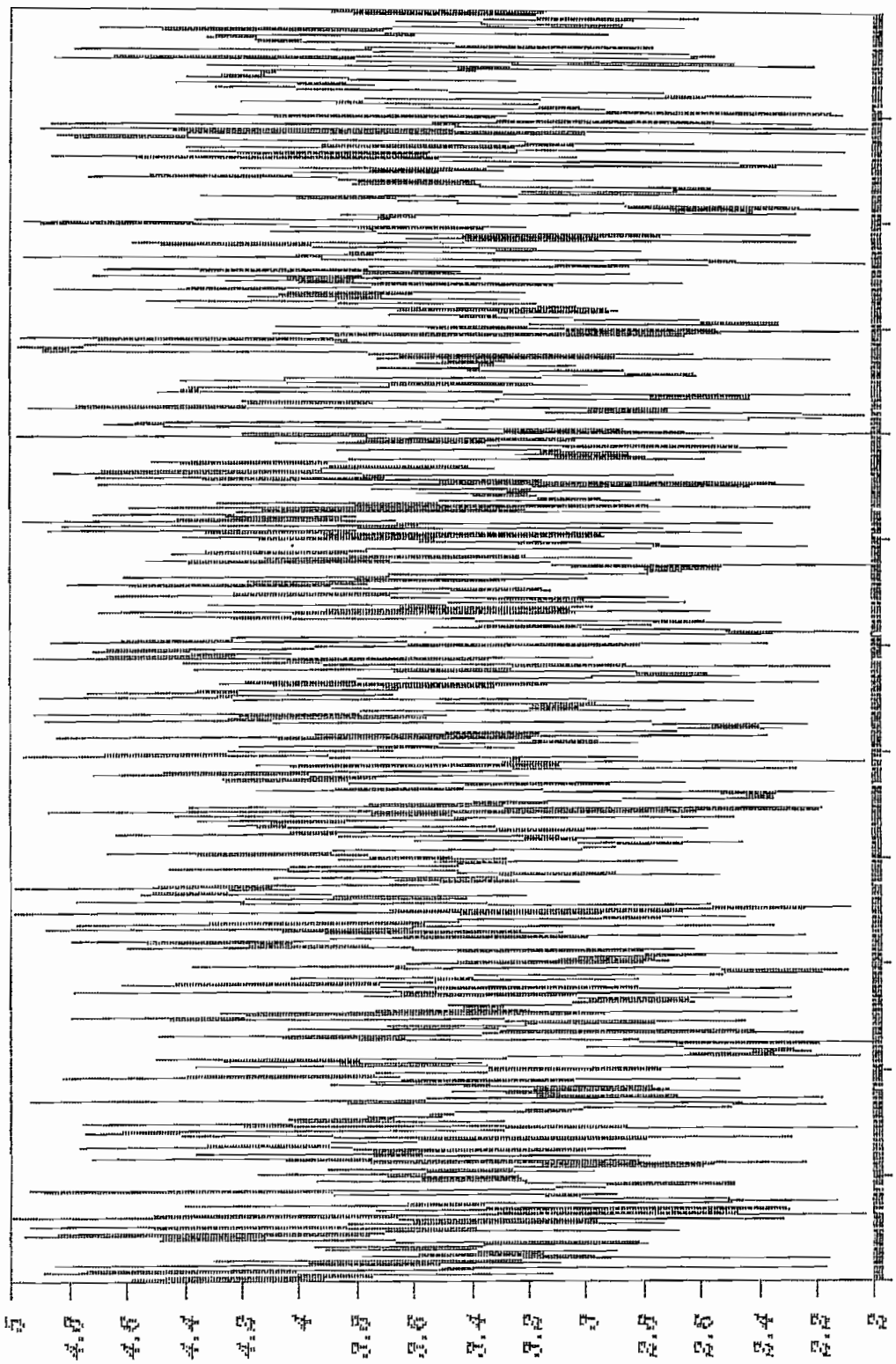


FIGURA 4.5

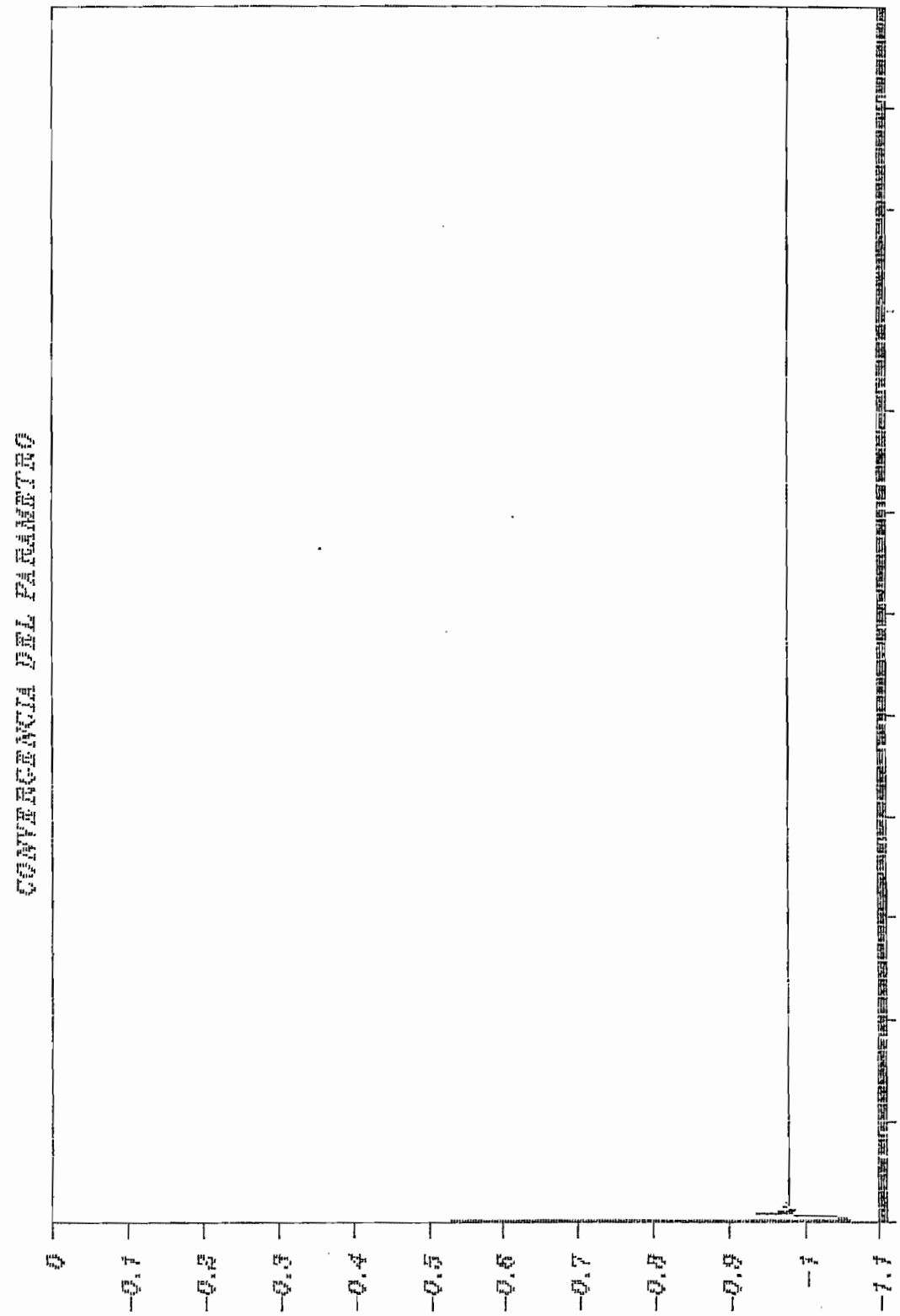

CONVERGENCIA DEL PARAMETRO
PARAMETROS α(i)
MUESTREOS
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
-0.8
-0.9
-1
-1.1
1
51
101
151
201
251
301
351
401
451
501
551

FIGURA 4.6

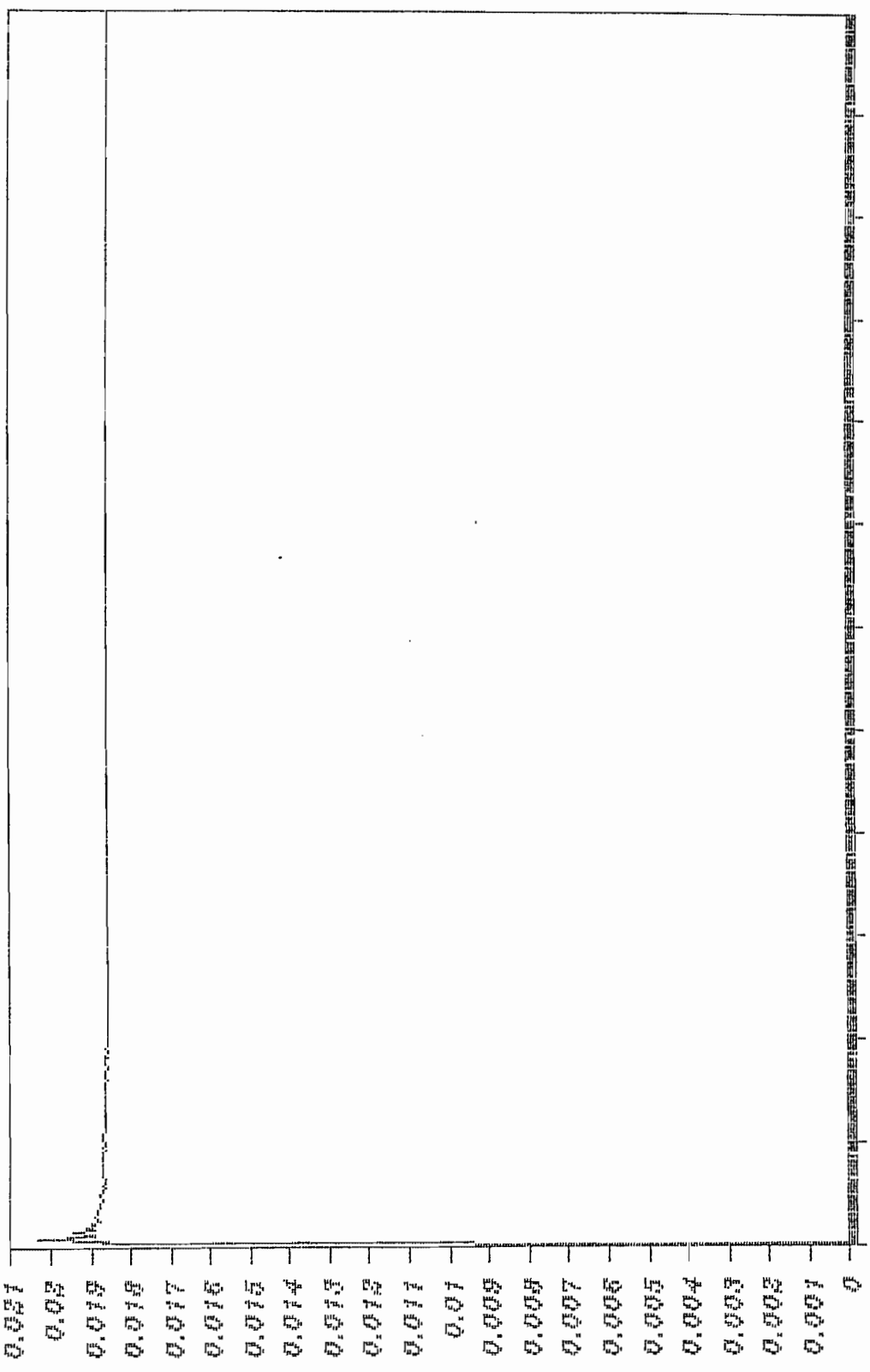

CONVERGENCIA DEL PARAMETRO
PARAMETROS Br)
0,021
0,02
0,019
0,018
0,017
0,016
0,015
0,014
0,013
0,012
0,011
0,01
0,009
0,008
0,007
0,006
0,005
0,004
0,003
0,002
0,001
0
1
51
101
151
201
251
301
351
401
451
501
551

FIGURA 4.7

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(i)= 0.920131
b(i)= 0.077822

Número de iteraciones = 1002

Período de muestreo = 1000 ms.

SEÑAL DE ENTRADA:
Escalón = 2
Ruido = 3

REPORTE # 5

CONVERGENCIA DEL PARAMETRO
PARAMETRO a(k)
0
-0,1
-0,2
-0,3
-0,4
-0,5
-0,6
-0,7
-0,8
-0,9
-1
1
51
101
151
201
251
301
351
401
451
501
551
MUESTRA

FIGURA 4.8

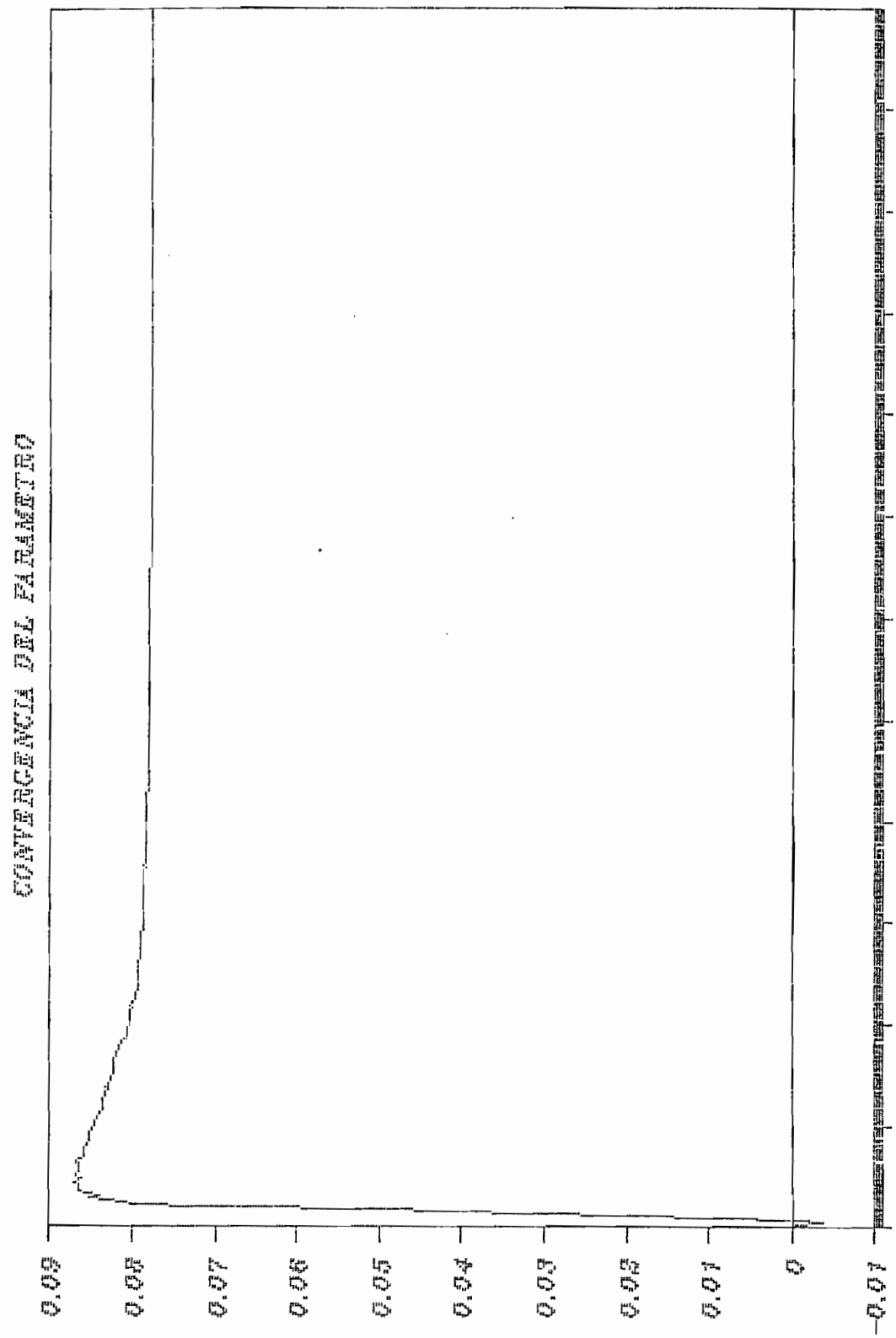

CONTRIBUCION DEL PARAMETRO
PARAMETRO Kr1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
-0.01
1
51
101
151
201
251
301
351
401
451
501
551

FIGURA 4.9

4.1.1.3. Diseño del control (Simulación).-

Se parte de un sistema con las siguientes características:

Ep% = 50%

ts = 19.4 s.

Para el diseño del control se hace necesario la utilización de los paquetes de simulación, para este caso específico el CAD CONTROL.

Se escoge un control del tipo P.I., proporcional integral.

$$Gc(s) = Kp + \frac{Ki}{s} = \frac{Ki}{s}\left[\frac{Kp}{Ki}s + 1\right]$$

de donde:

$$G_T(s) = \frac{1}{10s+1} * \frac{Ki}{s}\left[\frac{Kp}{Ki}s + 1\right]$$

Para conseguir un sistema de primer orden, se procede a cancelar el polo existente, luego:

$$\frac{Kp}{Ki}s + 1 = 10s + 1$$

de donde: Kp/Ki = 10

se escoge: Kp = 40, Ki = 4, para tener el polo deseado en lazo cerrado: p = - 4.

Luego:

$$Gc(s) = \frac{4(10s+1)}{s}$$

Entonces, la siguiente función de transferencia total en lazo abierto es:

$$G_T(s) = \frac{ki}{s} = \frac{4}{s}$$

Para la simulación del sistema compensado se utiliza el paquete CAD CONTROL, y específicamente el macro PID.mac, que requiere el ingreso de los parámetros Kd, Kp, Ki:

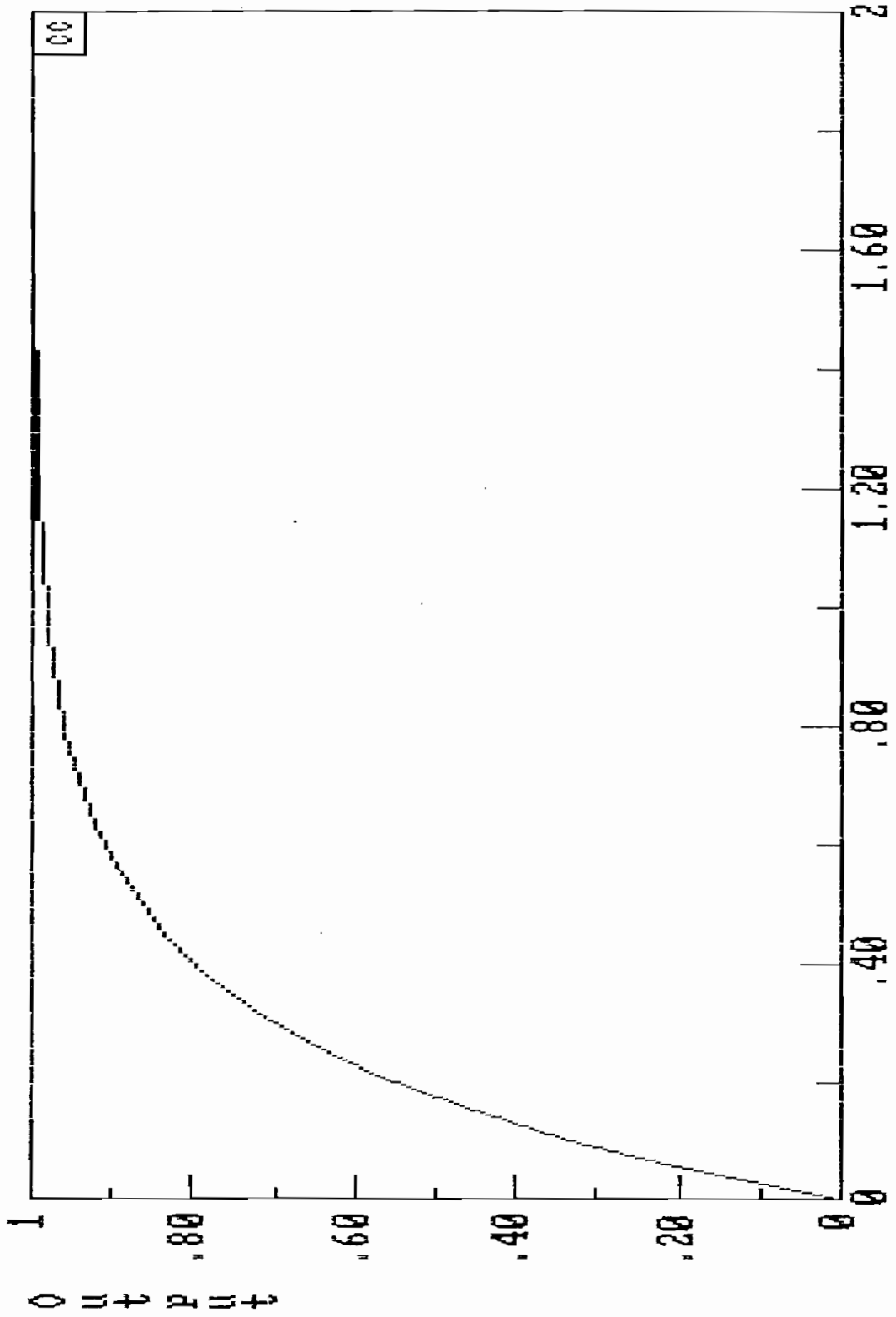
CC>@PID,kd,kp,ki, es decir

CC>@PID,0,40,4,g1

Obteneniéndose la figura 4.10, de la cual se obtiene las siguientes características:

Ep % = 0%

ts = 1 s.



RESPUESTA EN EL TIEMPO DEL SISTEMA COMPENSADO (PID)
FIGURA 4.10

4.1.1.4. Control en tiempo real.-

Para las pruebas de control se procede a utilizar los resultados obtenidos en el análisis y diseño obtenidos en los numerales anteriores, y a ingresar los parámetros del controlador utilizando el programa CONTROL, dentro del que se tiene dos opciones PID y REDES, debiéndose aclarar que para las acciones de control, es decir PID, es necesario ingresar los parámetros de ganancia, ya sea esta proporcional, integral y/o derivativa. Mientras que para el de redes se hace necesario discretizar, pues el ingreso de los parámetros es en forma discreta, de la forma:

$$u(k)+a_1u(k-1)+\ldots+a_nu(k-n)-b_0e(k)+b_1e(k-1)+\ldots+b_ne(k-n)$$

El ingreso de los parámetros es manteniendo el signo, de la ecuación anterior.

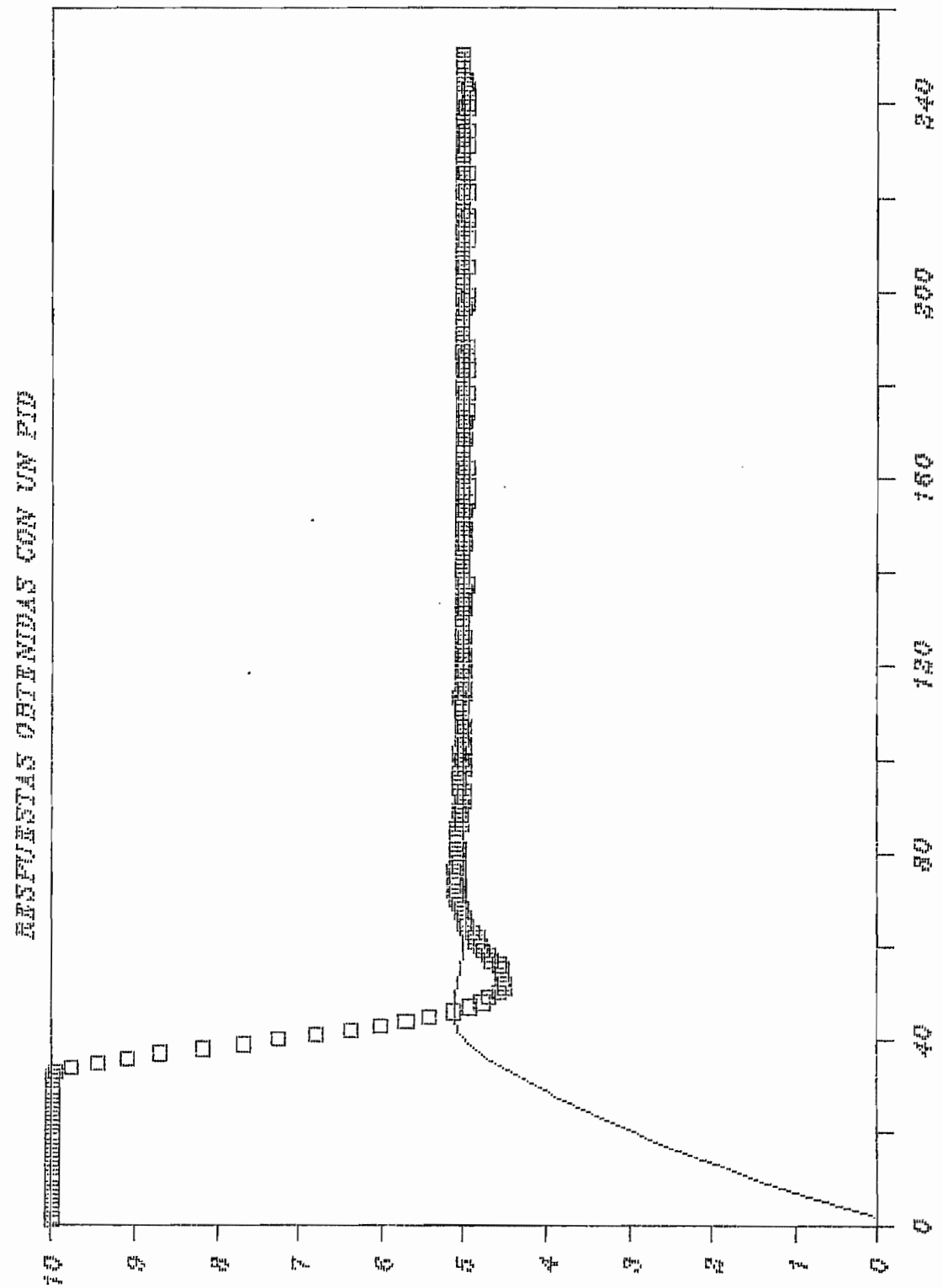
Para el primer compensador tipo PI, se hace necesario ajustar parámetros a los valores: Kp = 10 y Ki = 8, Kd = 0, r = 5 v., obteniéndose la respuesta de la figura 4.11, para una entrada de 5 voltios.

De donde:

Ep % = 0%

ts = 7.6 s.

Mp % = 2.52%

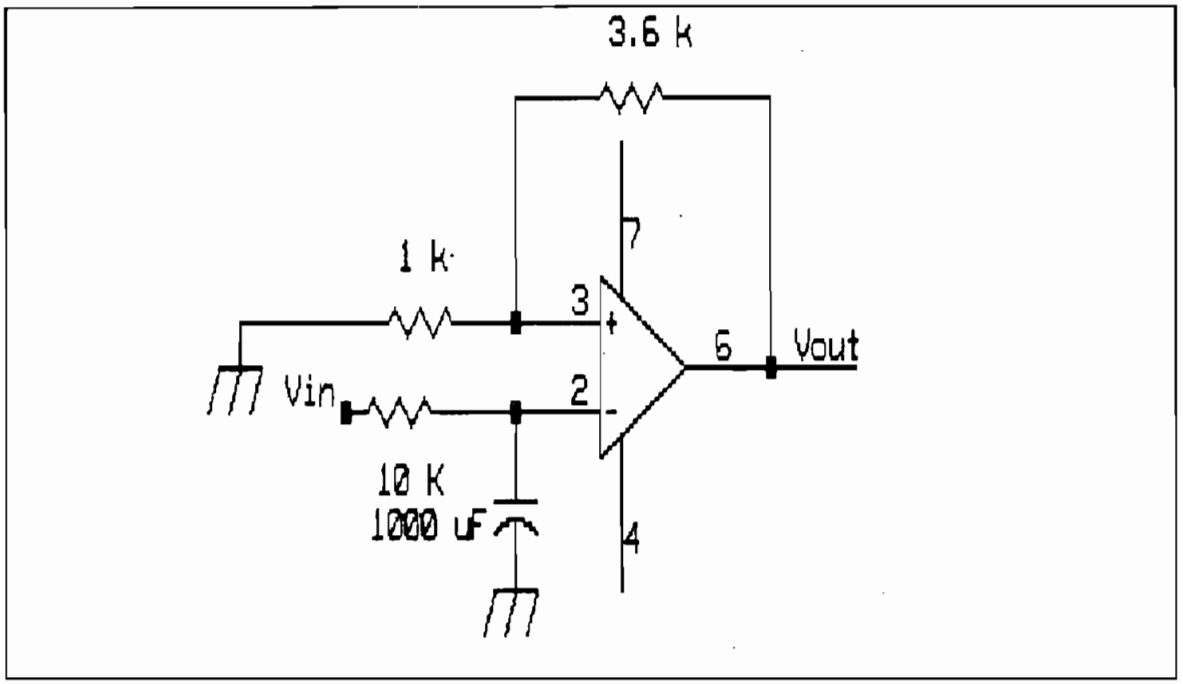


RESPUESTA SALIDA COMPENSADOR Y PLANTA EN TIEMPO REAL
FIGURA 4.11

4.1.2. CIRCUITO FILTRO ACTIVO DE PRIMER ORDEN PASA BAJOS

4.1.2.1 Modelación y Simulación Discreta.-

La figura 4.12 muestra un circuito activo de primer orden, de donde:



de donde:

$$G(s) = \frac{Vout}{Vin} = \frac{\frac{Rf}{Rs}}{sCR+1} = \frac{3.6}{10s+1}$$

Rf = 3.6 KΩ Rs = 1 KΩ

R = 10 KΩ C = 1000 μF

Con el paquete CAD CONTROL se obtiene el equivalente discreto para un período de muestreo de 200 milisegundos:

$$G(z) = \frac{0.07128478}{z - .9801987}$$

Entonces:

$$y(k) = .07128478u(k-1) + .9801987y(k-1)$$

Con la ayuda del mismo paquete se encuentra la respuesta en el tiempo del sistema discreto, para un período de 200 milisegundos, figura 4.13. Se tienen las siguientes características:

Ep % = 22%

ts = 8.1 s.

4.1.2.2. Identificación.-

Como se dijo con el programa IDENSIMU se procede a realizar la simulación, apreciándose una clara convergencia de los parámetros, como se puede ver en el reporte #2.

Mientras, que con el programa IDENREAL se realiza la identificación en tiempo real de la planta.

Aprovechando de las facilidades que presenta el programa se ingresa dos entrada diferentes, la una es un escalón puro

de magnitud de 2 voltios y la otra una escalón de 1 voltios sumada un magnitud de 2 voltios de ruido tipo random, se obtiene el modelo en los reportes #6 y #7.

- Entrada sin ruido:

$$y(k) = .086172u(k-1) + .986549y(k-1)$$

- Entrada con ruido:

$$y(k) = .086275u(k-1) + .984570y(k-1)$$

4.1.2.3. Diseño del control (Simulación).-

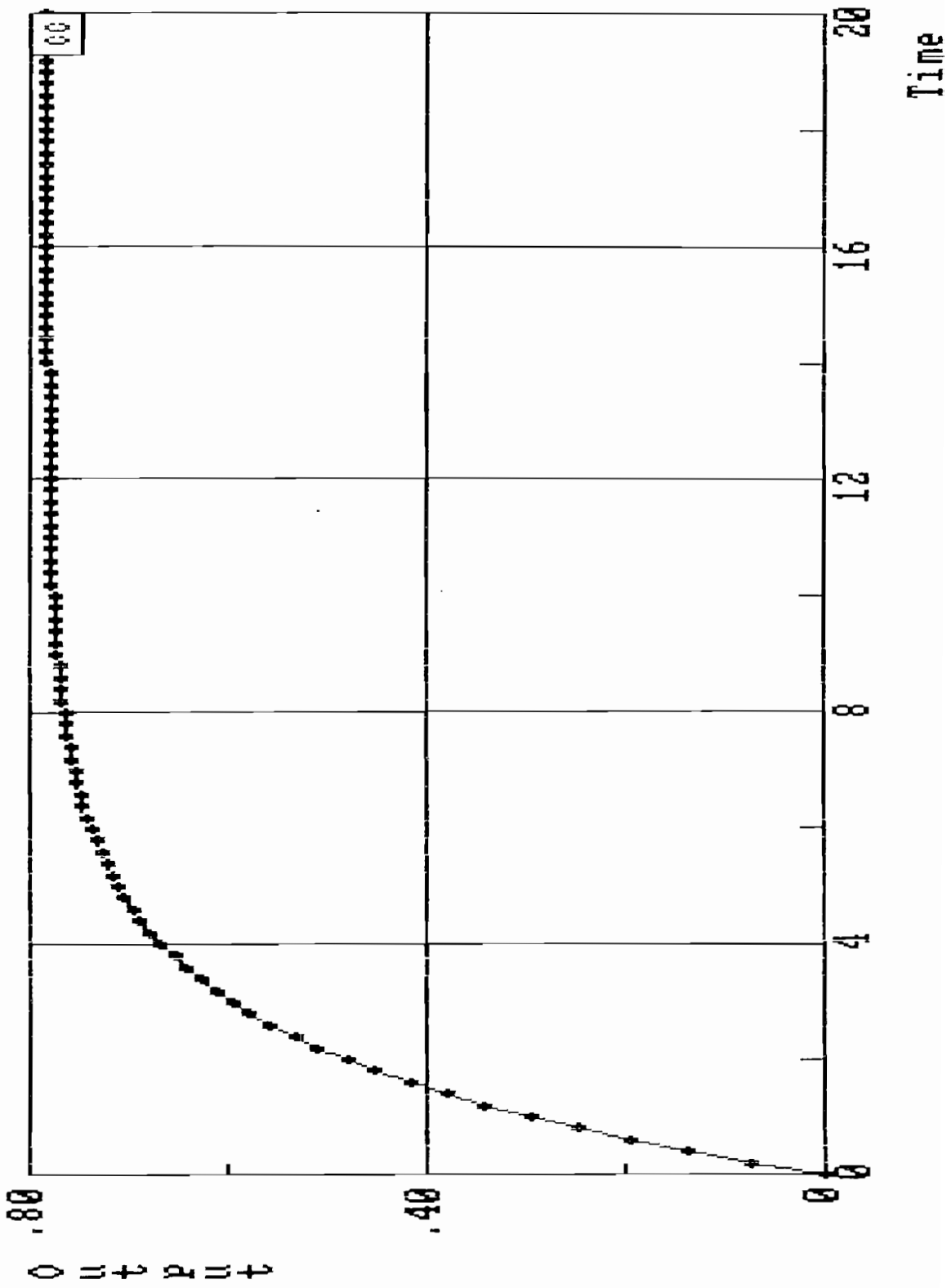
Se parte de un sistema con la siguientes características:

Ep% = 22%

ts = 8.1 s

Se considera para este caso solo un control, del tipo PI, proporcional integral.

$$Gc(s) = \frac{Ki}{s}\left[\frac{Kp}{Ki}s+1\right]$$



RESPUESTA EN EL TIEMPO DEL SISTEMA
FIGURA 4.13

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(1)= 0.981110
b(1)= 0.086172

Número de iteraciones = 1010

Período de muestreo = 200 ms.

SEÑAL DE ENTRADA:
Escalón = 2
Ruido = 0

REPORTE #6

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(1)= 0.981443
b(1)= 0.086275

Número de iteraciones = 1012

Período de muestreo = 200 ms.

SEÑAL DE ENTRADA:
Escalón = 1
Ruido = 2

REPORTE # 7

de donde:

$$G_T(s) = \frac{3.6}{10s+1} * \frac{Ki}{s} \left[\frac{Kp}{Ki}s+1\right]$$

luego:

$$\frac{Kp}{Ki} = 10$$

entonces: Kp = 10, Ki = 1, para un polo p = -3.6, con lo cual:

$$Gc(s) = \frac{10s+1}{s}$$

La función de transferencia total en lazo abierto es:

$$G_T(s) = \frac{3.6}{s}$$

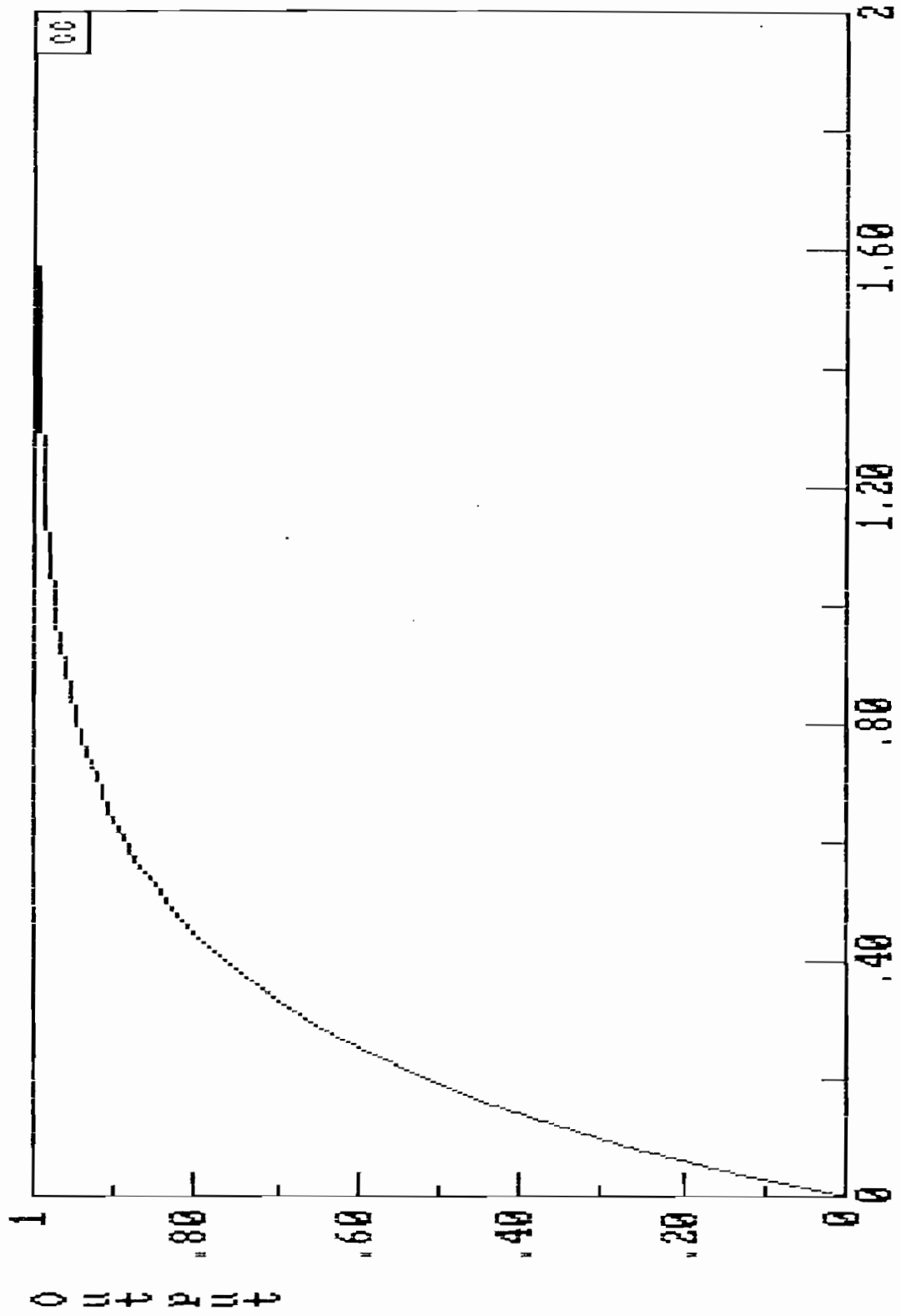
Con la ayuda del macro realizado en CAD CONTROL se obtiene la respuesta en el tiempo, figura 4.14.

CC>PID,0,10,1,gi

Obteniéndose las siguientes características:

Ep % = 0 %

ts = 1.1 s.



RESPUESTA EN EL TIEMPO DEL SISTEMA COMPENSADO (PID)
FIGURA 4.14

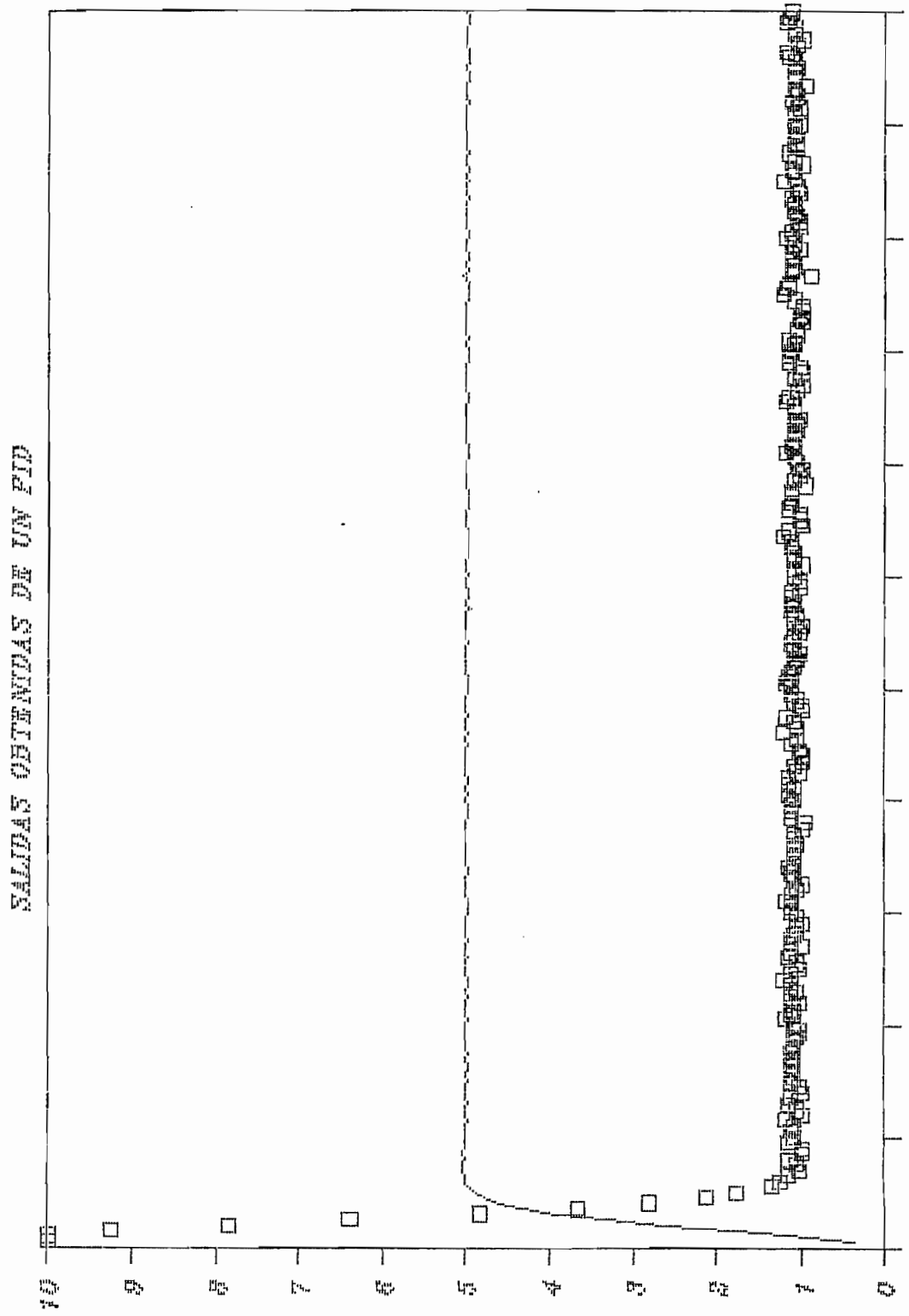
4.1.2.4. Control en tiempo real.-

Para este circuito se realizó un control PI, luego de un ajuste de parámetros se obtuvo: Kp = 5, Ki = 5 y Kd = 0. Su respuesta se puede ver en la figura 4.15, para una referencia de 5 voltios de donde:

Ep % = 0%

ts = 2 s.

Mp % = 2.25%



RESPUESTA SALIDA SISTEMA Y COMPENSADOR EN TIEMPO REAL
FIGURA 4.15

4.1.3. CIRCUITO RC ACTIVO DE SEGUNDO ORDEN

4.1.3.1. Modelación y Simulación Discreta.-

La figura 4.16 muestra el circuito de segundo orden activo, el que va a ser analizado.

A partir del circuito se tiene:

$$y''=4.7u(t)-4.7y-2.14y'$$

$$s^2y+2.14sy+4.7y=4.7u$$

Se tiene la siguiente función de transferencia en lazo abierto:

$$G(s)=\frac{4.7}{s^2+2.14s+4.7}$$

Se obtiene el equivalente discreto para un período de muestreo de 200 milisegundos:

$$G(z)=\frac{.08067191(z+.8664331)}{z^2-1.501243z+.6518114}$$

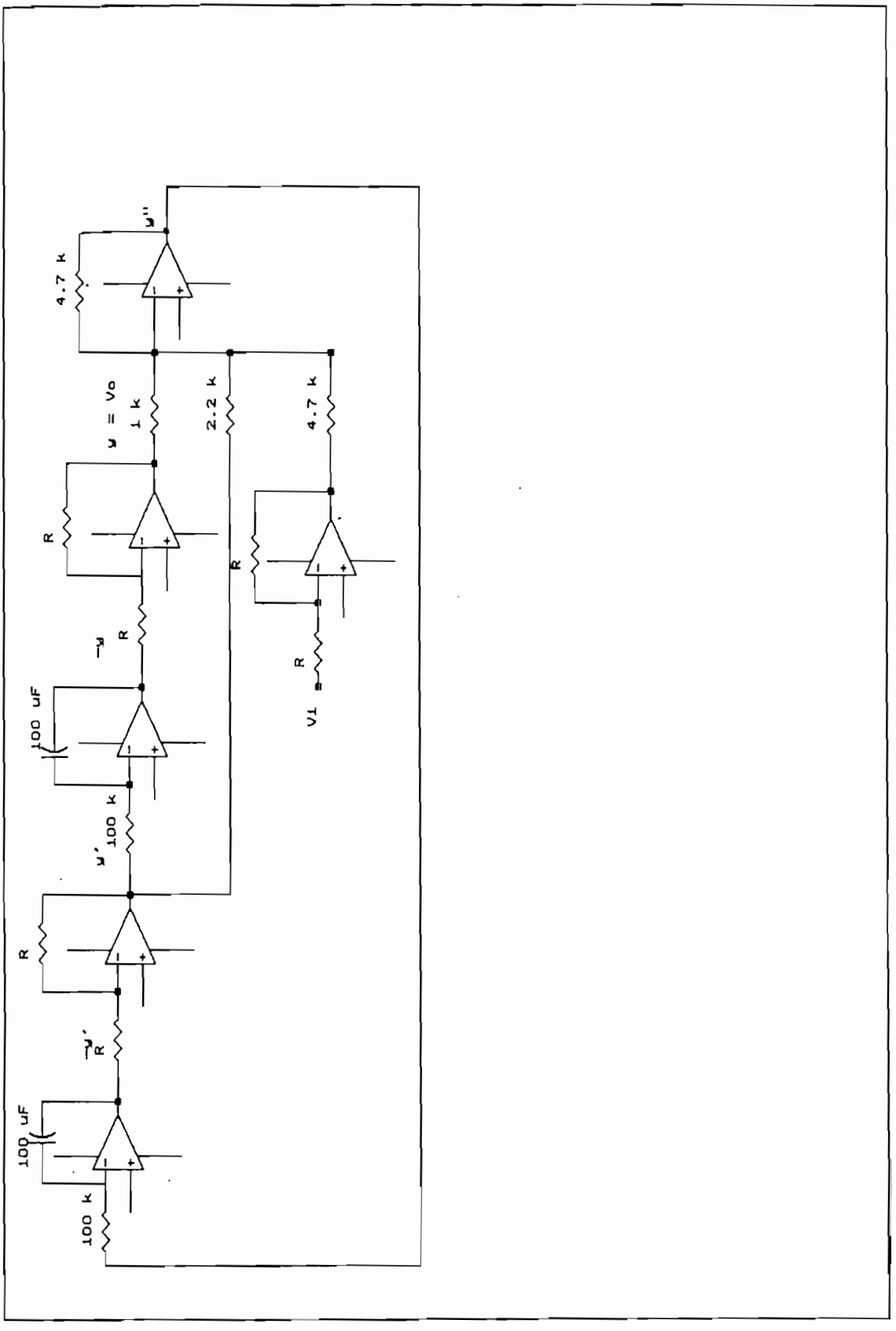


FIGURA 4.16

donde:

$$y(t) = .080671u(t-1) + .069896u(t-2) + 1.50124y(t-1) - .65181y(t-2)$$

Se escoge el período de muestreo de 200 milisegundos para encontrar la respuesta en el tiempo, figura 4.17, de la cual se obtiene:

Ep % = 50.2%

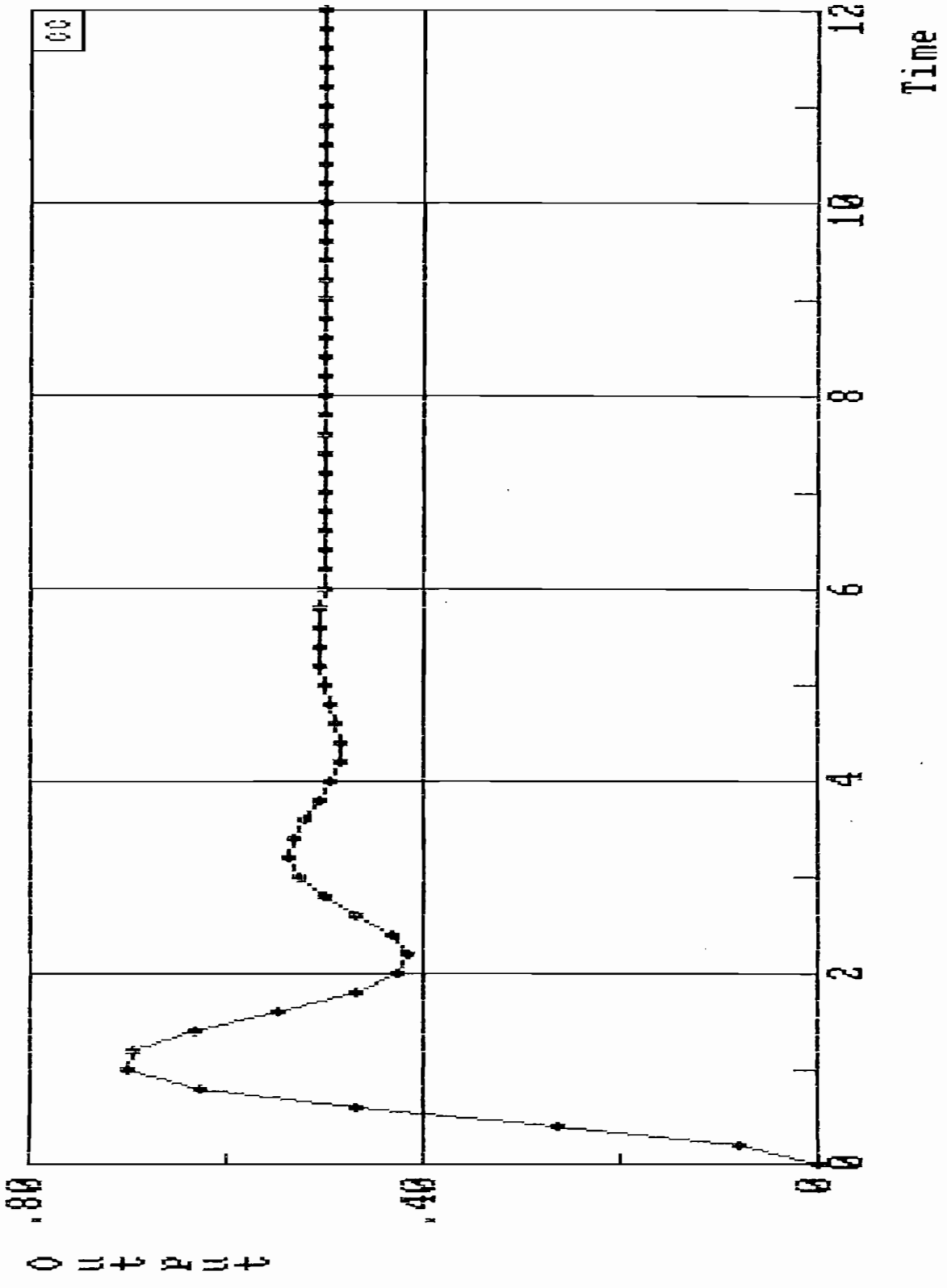
ts = 5.6 s.

Mp % = 40.56%

4.1.3.2. Identificación.-

Con la ayuda del programa IDENSIMU, se realiza la simulación obteniéndose también una clara convergencia de los parámetros, como se ve en el reporte #8.

Mientras que, con la ayuda del programa desarrollado IDENREAL se realiza la identificación para dos diferentes tipos de entrada, un escalón puro de 5 voltios, y la otra señal de entrada está compuesta de un escalón de 3 voltios y un magnitud de ruido random de 3 voltios, para un período de muestreo de 200 ms., como se ve en los reporte #9 y #10.



RESPUESTA EN EL TIEMPO DEL SISTEMA
FIGURA 4.17

RESULTADOS DE LA SIMULACION

P L A N T A

a(1) = 1.501243
a(2) = -.6518114
b(1) = 8.067191E-02
b(2) = 6.989681E-02

M O D E L O

a(1) = 1.5012
a(2) = -0.6518
b(1) = 0.0807
b(2) = 0.0699

Número de iteraciones 200

REPORTE # 8

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(1)= 1.536140
a(2)= -0.661595
b(1)= 0.067267
b(2)= 0.054410

Número de iteraciones = 1501

Período de muestreo = 200 ms.

SEÑAL DE ENTRADA:
Escalón = 5
Ruido = 0

REPORTE # 9

RESULTADOS DE LA IDENTIFICACION EN TIEMPO REAL

M O D E L O

a(1)= 1.537422
a(2)= -0.664328
b(1)= 0.066291
b(2)= 0.056611

Número de iteraciones = 1502

Período de muestreo = 200 ms.

SEÑAL DE ENTRADA:
Escalón = 3
Ruido = 3

REPORTE # 10

CONVERGENCIA DEL PARAMETRO
PARAMETRO a(2)
MUESTREOS
0,2
0
-0,2
-0,4
-0,6
-0,8
-1
-1,2
-1,4
-1,6
-1,8
-2
1
51
101
151
201
251
301
351
401
451
501
551

FIGURA 4.18

CONVERGENCIA DEL PARAMETRO
PARAMETRO a(2)
MUESTRAS
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
1
51
101
151
201
251
301
351
401
451
501
551

FIGURA 4.19

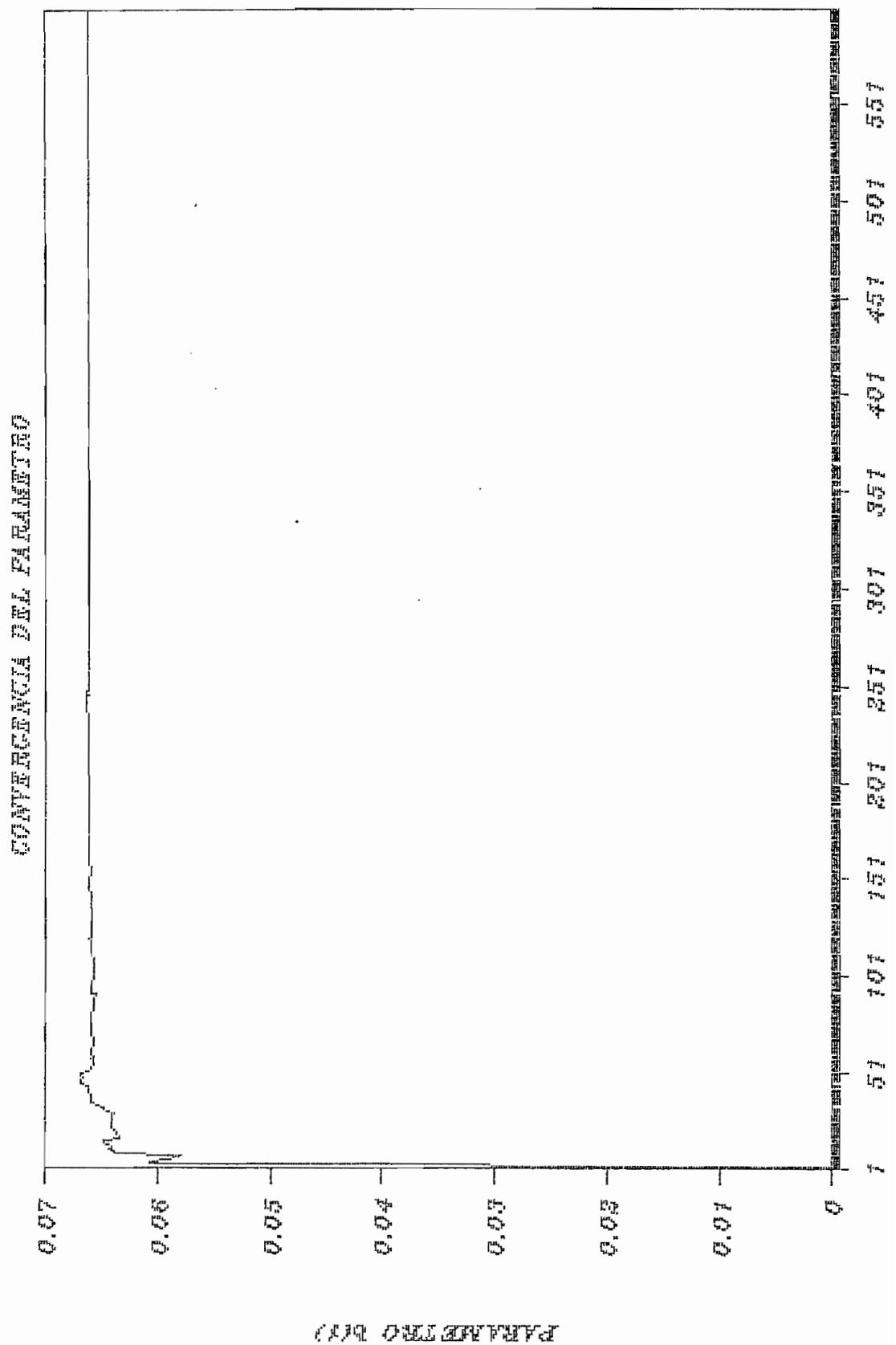

CONTRIBUCION DEL PARAMETRO
PARAMETRO Dfx
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
1
51
101
151
201
251
301
351
401
451
501
551

FIGURA 4.20

COMPORTAMIENTO DEL PARAMETRO b(3)
PARAMETRO b(3)
0,15
0,14
0,13
0,12
0,11
0,1
0,09
0,08
0,07
0,06
0,05
0,04
0,03
0,02
0,01
0
1
51
101
151
201
251
301
351
401
451
501
551
MUESTRA

FIGURA 4.21

$$Gc(s) = Kp + Kds + \frac{Ki}{s} = \frac{Kd}{s} * [s^2 + \frac{Kp}{Kd} s + \frac{Ki}{Kd}]$$

Para cancelar los polos complejos conjugados y llegar a una respuesta del tipo de primer orden se obtiene:

$$\frac{Kp}{Kd} = 2.14$$

$$\frac{Ki}{Kd} = 4.7$$

Entonces:

$$G_T(s) = Gc(s)\,G(s) = \frac{Kd}{s}$$

para tener un sistema con suficiente rapidez se escoge un Kd = .5, se tiene un Kp = 1 y Ki = 2.5 (con lo cual se ubica el polo en lazo cerrado p = −.5).

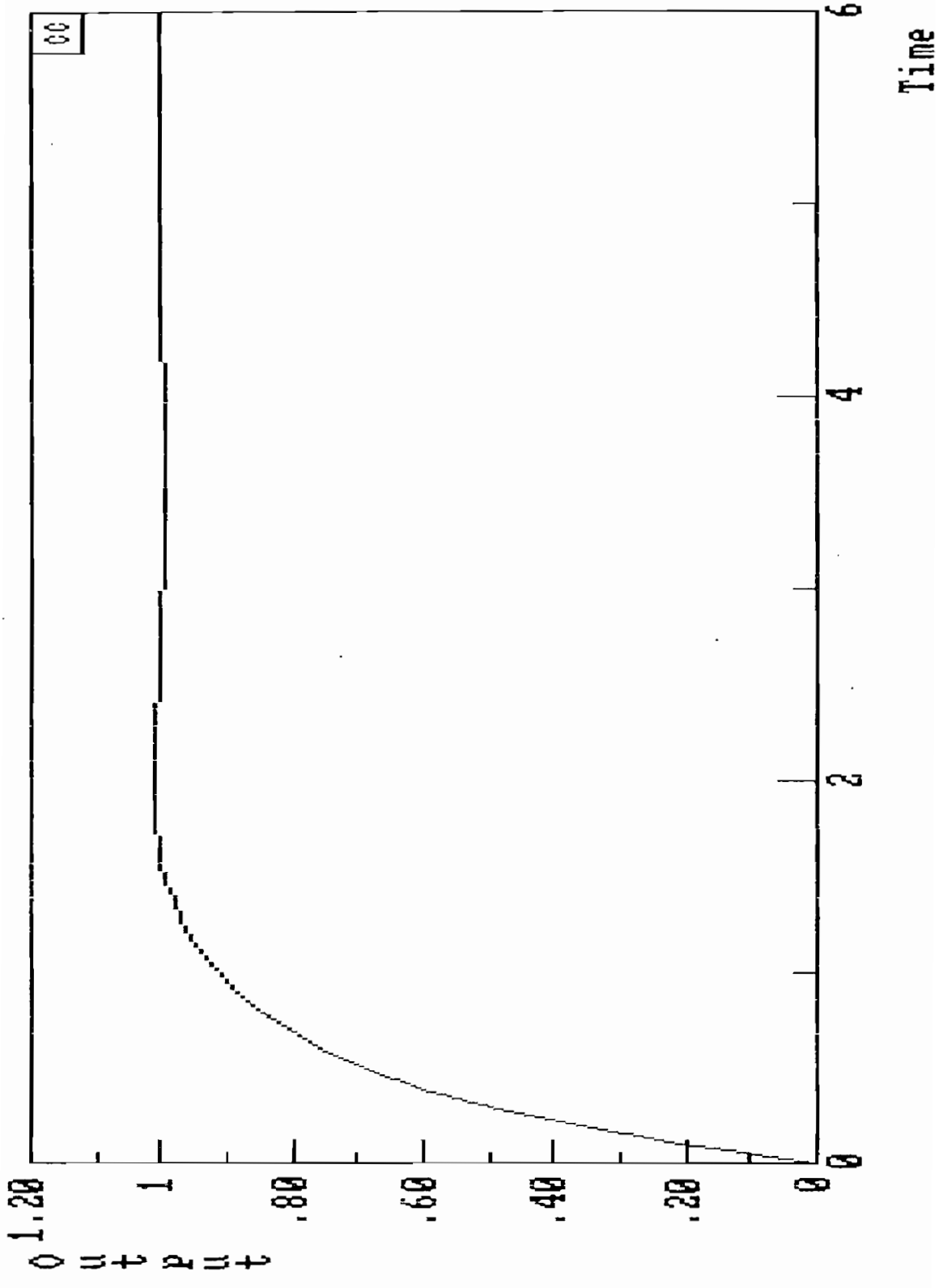
Con estos resultados se encuentra la repuesta en el tiempo, figura 4.22

CC>@PID,.5,1,2.5,G1

Se tienen las siguientes características:

Ep % = 0 %

ts = 1.8 s.



RESPUESTA EN EL TIEMPO DEL SISTEMA COMPENSADO (PID)
FIGURA 4.22

Se procede a diseñar otro tipo de compensador, esta vez se hará con el lugar geométrico de las raíces, pero discreto, figura 4.23

$$G(z) = \frac{.08067191(z+.8664331)}{z^2-1.501243z+.6518114}$$

Se escoge un compensador de tal manera que se cancele los polos complejos congujados de la función de lazo abierto original con un cero del compensador, y se ponga dos ceros uno en +.2 y otro + 1, de modo que:

$$Gc(z) = \frac{z^2-1.501243z+.6518114}{(z-.2)(z-1)}$$

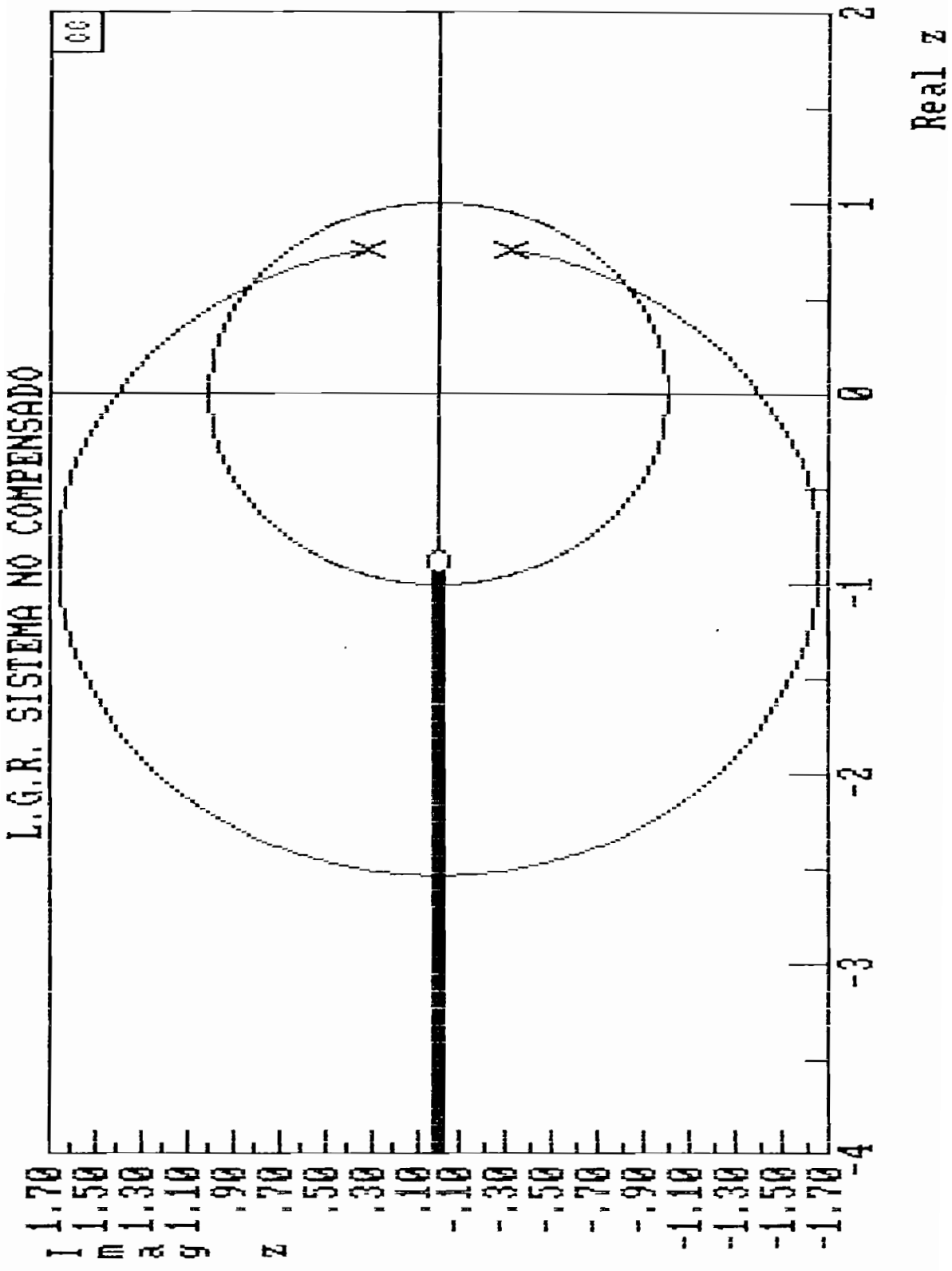
obteniéndose una función de transferencia en lazo abierto de:

$$G_T(z) = G(z) * Gc(z)$$

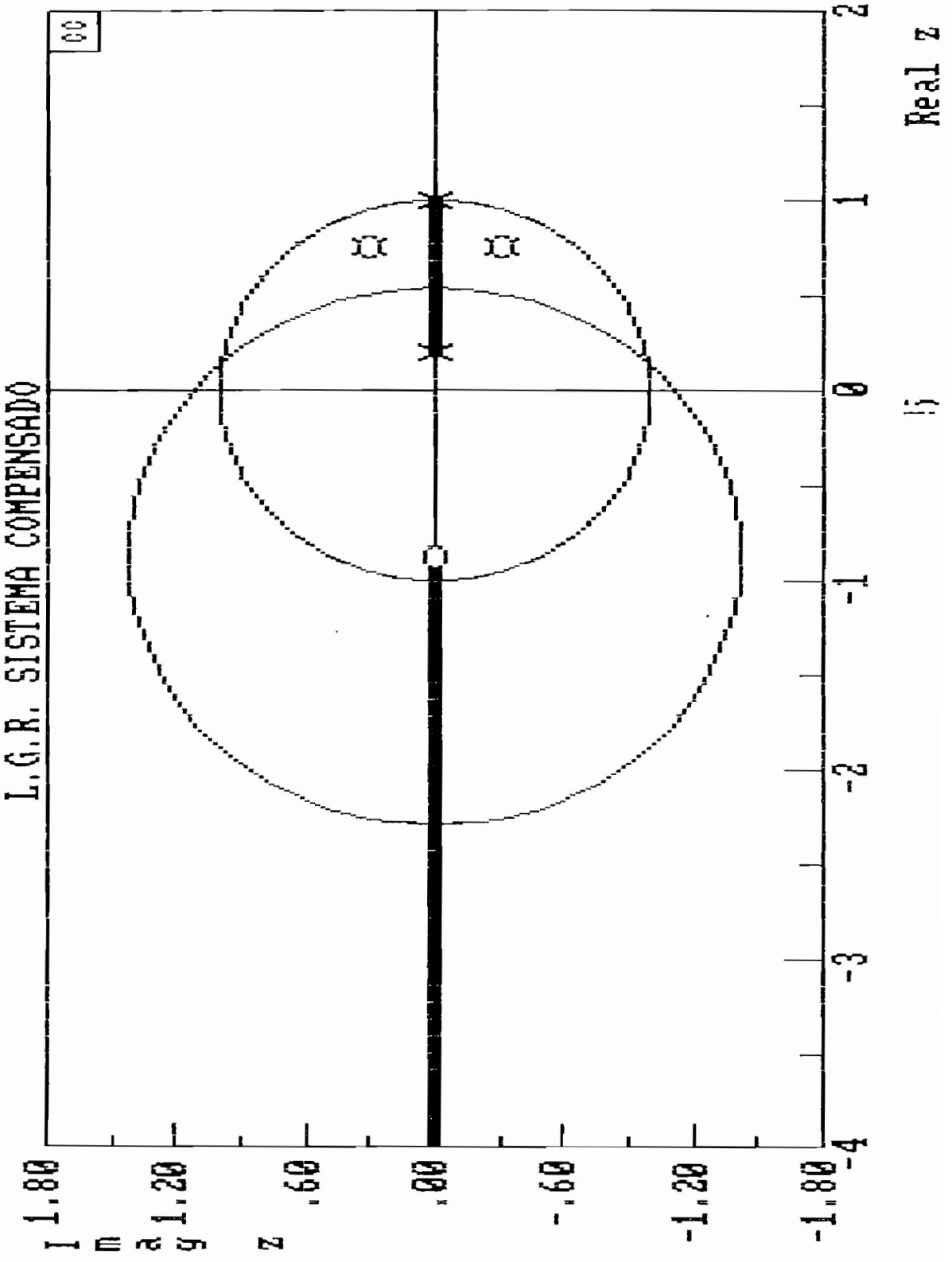
Haciendo ajuste de ganancia para satisfacer los polos deseados en: $p_1 = .37$ y $p_2 = .73$, se tiene:

$$G_T(z) = \frac{.08067191(z^2-1.501243z+.6518114)(z+.86644331)}{(z^2-1.501243z+.6518114)(z-.2)(z-1)}$$

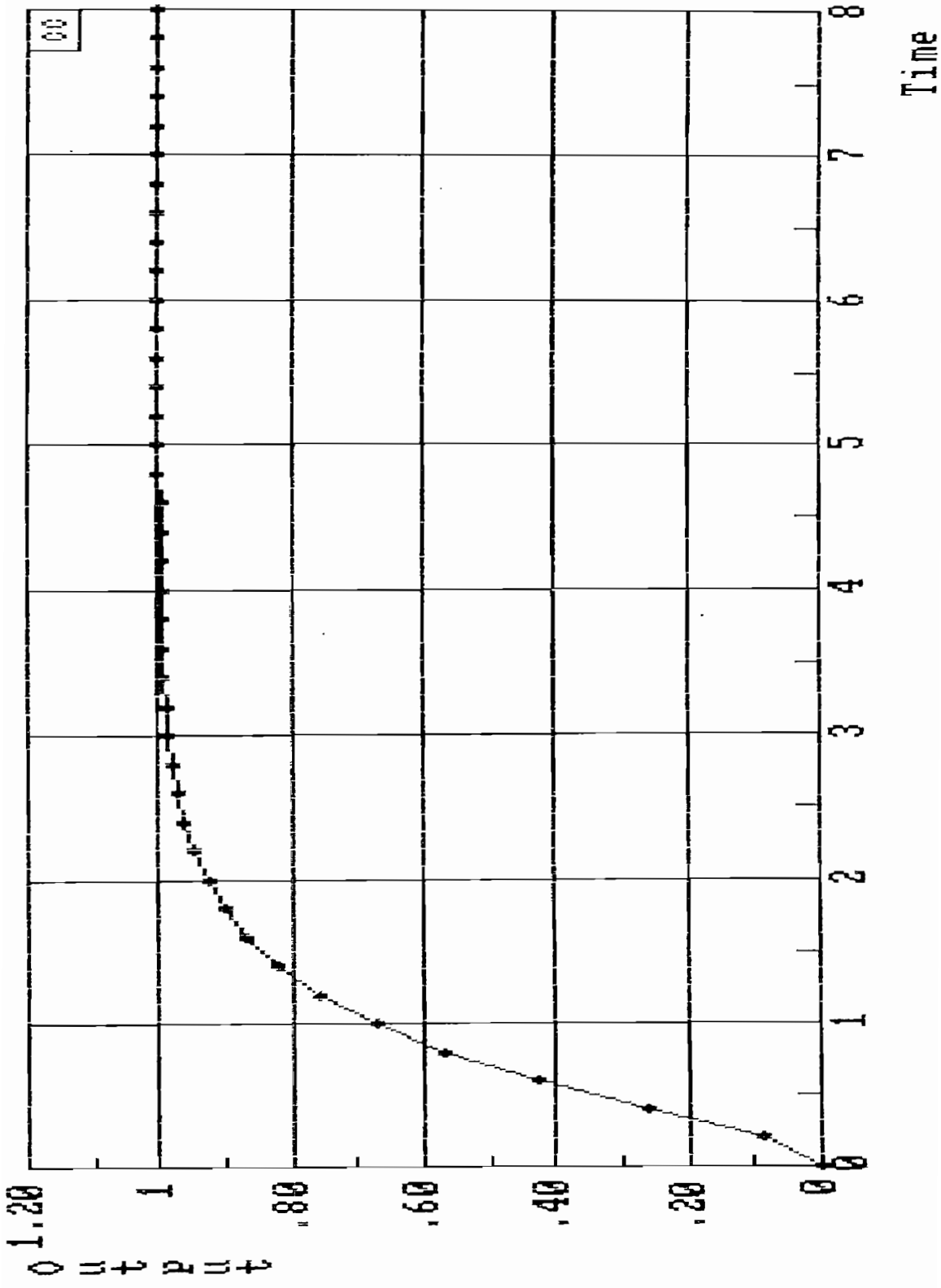
Obteniéndose el nuevo lugar geométrico de las raíces, figura 4.24 y la nueva respuesta en el tiempo en lazo cerrado, figura 4.25.



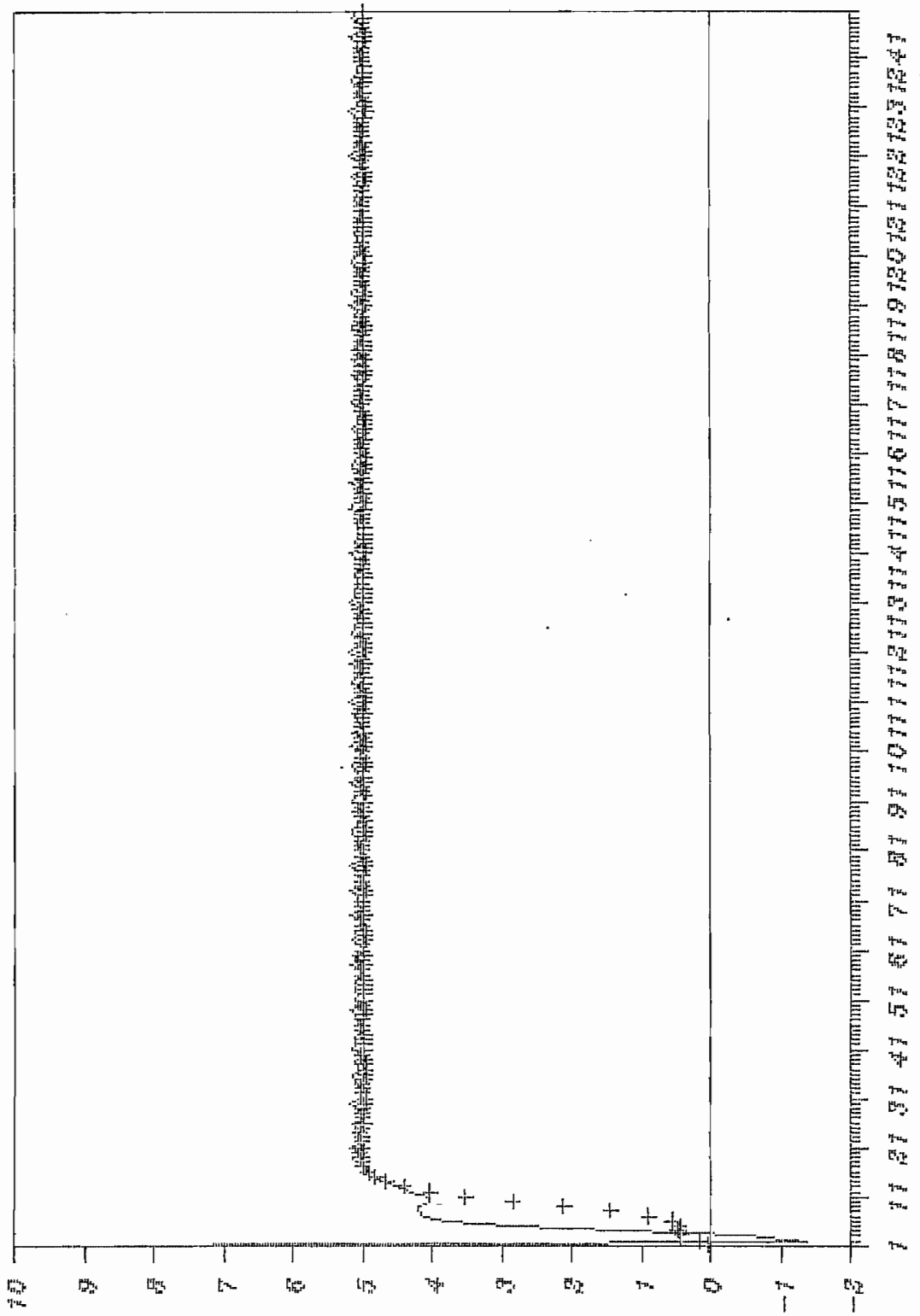
LUGAR GEOMETRICO DE LAS RAICES SISTEMA NO COMPENSADO
FIGURA 4.23



LUGAR GEOMETRICO DE LAS RAICES SISTEMA COMPENSADO
FIGURA 4.24



RESPUESTA EN EL TIEMPO SISTEMA COMPENSADO (REDES)
FIGURA 4.25



RESPUESTA SALIDA SISTEMA Y COMPENSADOR EN TIEMPO REAL (PID)
FIGURA 4.26

RESPUESTA SALIDA SISTEMA Y COMPENSADOR EN TIEMPO REAL (RED)
FIGURA 4.27

4.2. CONCLUSIONES.

Al terminar este trabajo se puede afirmar que se ha cumplido con los objetivos trazados, como son la utilización de paquetes de propósito general y software de propósito específico para el análisis y diseño de sistemas de control mediante la simulación y su aplicación en tiempo real, orientado hacia actividades docentes de laboratorio.

Los programas de propósito general estudiados son el KUO, TUTSIM, DYNAMO, CAD CONTROL, PC-MATLAB. Pudiendo concluirse que, el paquete KUO si bien tiene ayudas para el análisis de plantas dentro del estudio de sistemas de control, sin embargo su resolución gráfica es mala.

El paquete TUTSIM es bastante didáctico en el área de control, pues se reduce a la manipulación de una planta como si se utilizara un computador análogo, pero con las ventajas del computador digital. Su restricción es debido a que la única respuesta que se obtiene es en función del tiempo.

El paquemte DYNAMO es un paquete de uso más general, pues para su uso no se necesita conocer el tema en sí, tan solo obtener las ecuaciones de estado y programarlas. Su restricción en cuanto al uso dentro del área del control de sistemas es que su única respuesta está en función del

tiempo.

Se llega a la conclusión de que los paquetes de más uso en el área que compete a este trabajo son el CAD CONTROL y el PC-MATLAB. Así pues el CAD CONTROL posee una gran resolución gráfica. El PC-MATLAB ofrece una valiosa ayuda a nivel de la resolución matemática.

El software de la simulación ofrece una gran versatilidad en la variación de las condiciones de operación, tales como referencias, valores iniciales, ingreso de parámetros, elección de gráficos a ser visualizados, despliegue de resultados numéricos, resultados impresos en papel, etc..

En cuanto al software específico desarrollado, se puede decir que se divide básicamente en dos partes: la parte de simulación y en tiempo real. Los programas se desarrollaron en un computador IBM, PS60 con monitor a color y en lenguaje de programación QUICK BASIC, versión 4.5. Para trabajar en tiempo real se ha utilizado el equipo de adquisición de datos y control KEITHLEY 500A y el software especifico correspondiente para su manejo.

El software correspondiente a tiempo real, llevó al estudio de un sistema de adquisición de datos, para este caso particular la estación KEITHLEY 500A, lográndose aprender su uso en la forma más básica tanto a nivel de software como de

identificación en tiempo real. Para realizar las pruebas en tiempo real tanto para identificación como para control se utilizaron circuitos eléctricos como plantas físicas. A través del computador se genera la señal de excitación hacia la planta, mediante el sistema de adquisición y salida de datos que comunica la planta física con el computador. Para tiempo real se han utilizado dos tipos de planta, unas de primer orden, y otra de segundo orden. Las plantas de primer orden son la una pasiva formada por una resistencia y un condensador y otra activa formada por un arreglo de resistencias, capacitores y un amplificador operacional. El circuito de segundo orden está formado por resistencias, condensadores y amplificadores operacionales.

Para la identificación de parámetros de los circuitos se alimentó la planta con una señal de excitación persistente generada por el computador y enviada a la planta a través del equipo de adquisición de datos (conversor D/A). Esta señal está conformada por y un escalón, pudiendo añadirse ruido tipo RANDOM.

La señal de salida es la respuesta de la planta, esta señal es discretizada por el equipo mencionado, (conversor A/D) para ser procesada por el computador. Con esta información se procede a la identificación mediante el algoritmo de mínimos cuadrados recursivos, o al control mediante ecuaciones de diferencias o controles PID discretos.

La convergencia de los parámetros de identificación en simulación es bastante rápida y con gran exactitud, sin embargo en la identificación en tiempo real, se nota que su convergencia no es tan rápida.

En tiempo real intervienen condiciones que no aparecen en simulación. Los sistemas reales presentan variaciones, perturbaciones, no solo debido a la dinámica de la planta, sino a las interacciones con otros dispositivos como es la estación de entrada y salida de datos, ruidos indeseados en la señal, mediciones con limitaciones de las características propias de los equipos. A esto se debe sumar que cuando se obtiene matemáticamente el modelo discreto se lo hace utilizando el método del cero order hold (ZOH), que no deja de ser una aproximación. A pesar de esto el algoritmo demostró estar en la capacidad de identificar en tiempo real, pues a pesar de no llegar al valor teórico se nota que se tiene una tendencia a la convergencia de los parámetros a dichos valores.

En tiempo real resulta esencial determinar un período de muestreo que permita un control adecuado de la planta y una estabilidad de la señal de control. Lo primero se puede obtener con pequeños períodos de muestreo; mientras que un período de muestreo grande estabiliza el control, pero puede provocar que la información que llega al computador no describa totalmente la dinámica de la planta y por lo tanto

el control no sea adecuado. Un periodo muy corto hace que la planta sea muy sensible a las varaiciones de control, por lo que este último puede alcanzar oscilaciones muy grandes. No existe una regla definida para escoger el periodo de muestreo, pero una aproximación que aquí se ha utilizado es la de partir de una décima de parte de la constante de tiempo más significativa o de la frecuencia de oscilación de la respuesta de la planta. Además se debe tomar en cuenta como ya se dijo la longitud del programa.

El programa tiene limitaciones respecto a este tema, pues las plantas de dinámica muy rápida estarán limitadas por el tiempo que se demora en ejecutar el algoritmo implementado. El tiempo de duración del algoritmo se ve incrementado por la utilización de gráficos. Si se desea trabajar con este tipo de plantas se deberá optimizar el algortimo y eliminar la opción de gráficos.

Para los ejemplos en tiempo real una vez encontrado el modelo se la planta se procedió al diseño de la ley de control. Los controladores se diseñaron básicamente con la ayuda del programa cad control.

A continuación se hará un comentario para cada caso de estudio expuesto en este trabajo:

1.- Circuito RC pasivo de primer orden:

Los errores encontrados en la comparación realizada en la identificación entre la simulación y tiempo real fueron:

- Para un T = 200 ms.:

A una entrada asociada con ruido:

Ea % = .12%

Eb % = 6.84%

A una entrada sin con ruido:

Ea % = .08%

Eb % = 5.2%

- Para un T = 1 s.:

A una entrada asociada ruido:

Ea % = 1.66%

Eb % = 22.28%

En cuanto a las pruebas de control se obtuvieron los siguientes resultados:

- Simulación:

Kp = 40

Ki = 4

Oteniéndose:

Ep % = 0%

ts = 1 s.

- Tiempo real:

Kp = 10

Ki = 8

Obteniéndose:

Ep % = 0%

ts = 7.6 s.

Mp% = 2.52%

2.- Filtro activo pasa bajos de primer orden:

- Para un T = 200 ms.:

A una entrada asociada con ruido:

Ea % = .09%

Eb % = 20.88%

A una entrada sin ruido:

Ea % = .13%

Eb % = 17.37%

En control se obtuvieron los siguientes resultados:

- Simulación:

Kp = 10

Ki = 1

Oteniéndose:

Ep % = 0%

ts = 1.1 s.

- Tiempo real:

Kp = 5

Ki = 5

Obteniéndose:

Ep % = 0%

ts = 2 s.

Mp% = 2.25%

3.- Circuito activo de segundo orden:

- Para un T = 200 ms.:

A una entrada asociada con ruido:

Ea_1 % = 2.4%

Ea_2 % = 1.92%

Eb_1 % = 19.92%

Eb_2 % = 28.46%

A una entrada sin ruido:

Ea_1 % = 2.32%

Ea_2 % = 1.5%

Eb_1 % = 19.93%

Eb_2 % = 28.46%

Donde:

- Ea = error de los parámetros a_i
- Eb = error de los parámetros b_i

En control se obtuvieron los siguientes resultados:

- PID

- Simulación:

Kp = 1

Ki = 2.5

Kd = .5

Oteniéndose:

Ep % = 0%

ts = 1.8 s.

- Tiempo real:

Kp = 1

Ki = 1.5

Kd = .5

Obteniéndose:

Ep % = 0%

ts = 3.2 s.

Mp % = .36%

- REDES

Con los mismos parámetros tanto para simulación como para tiempo real se obtuvo:

- Simulación:

Ep % = 0%

ts = 3.2 s.

- Tiempo real:

Ep % = 0%

ts = 3.2 s.

Mp % = 0.8%

Como se puede observar para el primer caso, el circuito de primer orden pasivo y para el período de 200 milisegundos los errores están dentro de un margen aceptable.

Se ve que al aumentar el período de muestreo el margen de error crece, es por eso que se debe tener muy en cuenta el período de muestreo en función de los parámetros de la planta.

En control, se vio en la necesidad de bajar considerablemente las magnitudes de las constantes, esto se debe fundamentalmente a que en simulación no existe ningún tipo de restricción; mientras que para el tiempo real se debe tomar en cuenta que los límites de voltajes son +/- 10 v.

Así, al analizar se vio que para un Kp = 40 y Ki = 4 se obtuvo para el primer instante un valor pico de 40 v., el mismo que para tiempo real sería cortado a 10 v., por lo que este vendría a ser el voltaje que se suministraría a la planta. Por lo que se procede a hacer una variación de las constantes utilizando el método de ensayo y error; y se llega a tener los nuevos valores indicados.

Para el segundo caso de estudio se puede ver que los errores obtenidos en la identificación son aceptables.

Los parámetros encontrados en el diseño del control en simulación se cambiaron, consiguiendo de esta manera un mejor tiempo de respuesta.

En el tercer caso de estudio, se ve una convergencia de

los parámetros encontrados en tiempo real, a los obtenidos en simulación.

En cuanto al control se ve que para los dos casos, las características encontradas son buenas. Sin embargo para el caso PID se vio en la necesidad de hacer ajustes a los parámetros. Se procedió a bajar el máximo sobreimpulso a costa de el tiempo de establecimiento.

Para el caso de REDES no fue necesario el ajuste de ninguno de los parámetros, obteniendo una respuesta satisfactoria.

Finalmente se puede concluir que los programas desarrollados han cumplido con el objetivo de la tesis.

BIBLIOGRAFIA:

- FAUSTO ALBERTO VASCO MONCAYO, Estudio del Sistema de Adquisición de Datos Keithley 500A y Aplicaciones, E.P.N., 1991
- GERMANICO PINTO, Identificación de sistemas en tiempo real, E.P.N., 1991
- KEITHLEY, Quick 500 Data Acquisition and Control Software, 1988
- OGATA KATSUHIKO, Ingeniería de Control Moderna, Prentice-Hall, 1973
- MICROSOFT, Microsoft Quick Basic, Microsoft Corporation, 1988
- FRANKLIN G. & POWELL, Digital control of dynamic systems, Addison Wesley, 1981
- KUO BENJAMIN, Digital control systems, Halt-Samders, 1981
- STEVE BANGERT & STEVE KLEIMAN, Manual del PC-Matlab, The MathWorks Inc., 1984.
- PETER THOMPSON, User's Guide Program CC, Versión 3.0, System Technology Inc., 1985
- S.N., User's manual TUTSIM, Versión 5.0, Copyright, 1985
- BENJAMIN KUO, Software manual for accompany Automatic control systems, Prentice-Hall Inc., 1987
- S.N., Manual del DYNAMO , (Conseguido en la Fundación Nuestros Jóvenes, incompleto)