Teaching and Learning Basic Control Systems with TFI

G. Marro

Department of Electronics, Systems and Computer Science

University of Bologna, Italy

e-mail: gmarro@deis.unibo.it

The purpose of this manual is to describe the Transfer Function Interpreter, a package in Matlab [1] that was specifically developed to provide an effective interactive environment for control system design, useful both in didactics and profession.

The manual consists of two main parts:

General Information, including an *Introduction*, that presents the features of the package and its use as a lab support in Automatic Control System courses, and an *Installing, setting, and starting* section, that describes installation procedures and possible personalizations.

TFI and Its Applications, containing a detailed description of the Transfer Function Interpreter environment. For every one of the applications accessible from TFI, that are considered in alphabetical order, the presented material consists of a *Recall* section, containing some elements of the underlying theory, an *Operation* section, with the procedure(s) of use in some detail, and an *Examples* section, that for graphical applications includes numerous reproductions of screen layouts.

Upgrades of the software can be freely downloaded from the web page:

htpp://www3.deis.unibo.it/Staff/FullProf/GiovanniMarro/gm_tfi.htm

[1] Matlab is a registered trademark of The Mathworks, Inc., Cochituate Place, 24 Prime Park Way, Natick MA, 011760 USA.

Contents

Chapter 1

Introduction

1.1 What TFI is

What is TFI? The Transfer Function Interpreter (TFI) is a package of Matlab m-files that create a specific computer-aided design environment for interactive control system analysis and synthesis. Its main feature is direct definition of transfer functions with the keyboard. Transfer functions are permanently saved in the work directory of the hard disk as **.mat* files.

Hence, is TFI a Matlab toolbox? TFI is not exactly a toolbox even if it is installed in the same way, because it uses a second-level interpreter. When TFI is entered, a new prompt appears on the screen (> instead of »), to inform that the Command Window has been changed into a new one, with another syntax. However, since entering end exiting TFI from the Matlab Command Window are immediate, it may be simply considered an addition of new tools to the Matlab environment.

Is TFI simply an interpreter? No, besides direct transfer function manipulation, TFI also provides a great number of CAD functions that enable many topics of control system analysis and synthesis to be thoroughly managed. These are: inverse transformation of rational functions, linear system analysis in the time and frequency domain, stability problems and root locus, compensator and regulator design, robustness of stability in closed-loop systems, and nonlinear system analysis and design with the describing function method.

Why TFI is convenient? TFI makes the interactive approach to automatic control problems easier, since knowledge of Matlab programming language is not necessary. This feature simplifies its use both in basic didactics and in industrial design. Moreover, all the available graphic applications are provided with a friendly menu that allows very direct modification of the figure features, such as axes scales, grids, colors, and provides information useful for design.

Does TFI consider discrete-time systems? Yes, TFI recognizes and handles both continuous-time transfer functions (ratios of polynomials in s) and discrete-time transfer functions (ratios of polynomials in z), both accessible in a unique environment. Definition of a sampling time is interactively requested when entering TFI in order to match continuous and discrete. Utilities like discrete-time antitransformation, conversion from continuous to discrete and w-plane conversion, are provided for sampled-data control system design.

Does TFI consider state-space models? No, it only handles transfer functions. Strict multivariable system design is not available in the TFI environment, that has been specifically developed for introductory control system courses, while state-space analysis is usually presented in second-level courses. However, the pole assignment problem is approached in TFI environment with the Diophantine equation. This makes a complete presentation of the analytic design of regulators possible by using only transfer functions.

Is TFI a well-tested and reliable software? Yes, TFI has been developed as a teaching aid, copyrighted and distributed in Italy since September 1994 in a diskette enclosed with the most widespread textbook on Automatic Control Systems. It has been used in labs of numerous universities for several years, with continuous improvement, mainly due to suggestions coming from students.

What about numerical robustness of the computational routines? Particular attention has been given to numerical robustness in TFI. When transfer functions are stored, factorized forms are preferred to polynomial ones, so that multiple poles and zeros are robustly preserved. For instance, in conversion from continuous to discrete time a multiple pole at the origin robustly results in a unit pole with the same multiplicity, since the converted transfer function is provided in factorized form.

Are the computational methods used in TFI accessible to users? Yes, in this handbook every application is presented with a *Recall* section, that briefly reports the theory involved and the method of solution.

Are there new methods in TFI, that are not available in standard textbooks? Yes, these are the *inversion formulae* for lead and lag compensator design (see the *Recall* section of applications *regnich* and *pidnich*), that may also be taught in basic control theory courses, since they make trial-and-error design procedures faster, and *perfect tracking* in nonminimum-phase digital control systems (see the *Recall* section of application *perftra*), obtained through preview and preaction, that is now only available in technical papers, not yet in books.

Is there any TFI application particularly teaching-oriented? Yes, applications *deftf* and *regrootl*, that define transfer functions and regulators by direct allocation of their poles and zeros with the mouse, are particularly useful to easily

point out the correlation between pole/zero layout and time and frequency responses of linear dynamic systems, both continuous-time and discrete-time. They enable quick comparison of a certain number of different cases, whose pole/zero maps and time/frequency response plots are presented together in different colors. Other applications teaching-oriented are *lagc*, *leadc*, *pidc*, *pidd*, *pidnich* and *regnich*, that also enable to immediately show the effects of the free choices in trial-and-error synthesis methods.

What is the main feature of the TFI environment? It is making available powerful synthesis methods: basic applications like *regnich*, *pidnich* and *regrootl* have no equivalent in other teaching aids for automatic control systems. These are, in general, oriented towards time and frequency domain analysis and state-space synthesis, and do not include any means to speed up the fundamental trial-and error design methods based on gain and phase margins or dominant-pole location, that are the most convenient means to achieve sensitivity and insight on feedback system behavior. Furthermore, the standard analysis tools for time response, frequency response and root locus (applications *tresp*, *fresp* and *rootl*) are more complete and easily accessible than elsewhere, being provided with very unified and friendly interactive menus, that also enable a professional output in printers.

Is it possible to define transfer functions in parametric form? Yes, in the TFI environment it is possible to define transfer functions whose coefficients are any closed-term expression of one or several parameters, but their use is restricted to parametric robustness analysis (application *robpar*).

Is it possible to consider finite delays? Yes, both directly in open-loop frequency response diagrams (application *fresp*) and by using Padé approximants in the closed-loop case.

Does TFI require the Control System Toolbox? No, it uses specific computational and graphic routines.

Is there any intersection between using Simulink and TFI? No, Simulink is a very complete analysis-oriented tool, while TFI is a synthesis-oriented tool restricted to SISO systems.

Is there any difference between TFI and the Matlab's LTI systems? They are two different solutions of the same problem, i.e. to handle systems besides matrices with Matlab. TFI also perfectly works with Matlab 4, while the LTI systems are a specific feature of Matlab 5.

Is TFI compatible with the Matlab Student Edition? Yes, the two versions of TFI available (intp4 and intp5) are wholly compatible with the Matlab 4 and the Matlab 5 Prentice Hall Student Editions, respectively.

Why many commands in the TFI environment are keyboard-oriented? There are several reasons for this: *i)* we decided to make all the figures of the TFI design sessions continuously available, and use of the mouse is unadvisable when managing many Matlab-generated figures, since selection by mouse may disturb the running program; *ii)* a figure can arbitrarily be reduced or enlarged with the mouse, but the fonts of axes and text are fixed, so that its aspect is generally worsened with respect to the standard one, that in the TFI environment is full-screen for low-resolution monitors; *iii)* there are some functions, very useful to obtain or export professional figures for publications, like change of axes scales and/or graduations, or change of step when drawing a root locus, that are more easily done with keyboard than with mouse; *iv)* the Matlab's Command Window maintains a record of the session, that can be printed. Nevertheless, the mouse is used in all the design procedures that require selection of points on diagrams, in order to speed up trial-and-error methods.

1.2 TFI in a basic course on Control

The Transfer Function Interpreter has been developed for high-speed interactive analysis and design of single-input single-output automatic control systems. It is very useful in labs, specially those organized by teams, since it greatly facilitates learning by stimulating criticism. We will now see how TFI itself and the applications available in its environment are related to the topics usually considered in a standard course on Control, thus suggesting a chronological order for presenting these applications in connection with problem-solving. The reader interested in evaluating the TFI environment should check the programs in the same order as they are mentioned below.

1. Basic Concepts. Systems and mathematical models. Feedback versus feedforward. Block diagrams and signal-flow graphs. Some mathematical models of dynamic systems. Linearity and time-invariance.

TFI-based lab: TFI is introduced (by entering "tfi" from the Matlab Command Window) and applied to reduce purely algebraic block diagrams or signal-flow graphs to the minimal form. Transmission coefficient of blocks or branches are real numbers in the purely algebraic case, but it may be shown that the same reduction procedures and expressions apply when they are ratios of polynomial in s or z.

2. Time-domain analysis. Differential equations. Laplace transform, direct and inverse. The concept of transfer function. Impulse, step and ramp responses. Convolution integrals. Elementary first and second order systems.

TFI-based lab: Operations on transfer functions. Different forms of a transfer function ("gi=", factorized, "gi :", time-constant, "gi ;", pole-zero map). Application *invtr* is used to solve or to check the solutions of exercises involving the inverse

Laplace transform, and *tresp* to plot these inverse Laplace transforms versus time, to see graphs of impulse, step and ramp responses and to display the most important parameters of the step response (maximum overshoot, delay, rise and settling times, steady-state tracking error). Application *deftf* makes it possible to show the correlation between zero/pole layouts and step and frequency responses for one or more transfer functions whose zeros and poles are located with the mouse.

3. Frequency-domain analysis. The frequency response. Connection between time and frequency responses. Bode diagrams of the first and second-order elementary systems. Nichols diagrams. Nyquist (polar) diagrams.

TFI-based lab: Application *fresp*, that is the most important of the TFI environment, is used to see the various types of diagrams of the frequency response and to compare their features.

4. Stability and feedback. Definitions and theorems on stability. The Routh criterion. General properties of feedback. Steady-state tracking error and system type. The Nyquist criterion. Gain and phase margins. Constant M and N loci. Resonance peak and frequency, bandwidth. Stability and behavior of systems with pure delays.

TFI-based lab: Application *routh* directly gives the stability interval(s) in terms of gain of a feedback system, and is used to illustrate the Routh criterion. Stability of feedback systems in the most general cases (open-loop unstable systems) is studied by using *fresp* again (with the "Nyquist diagram" option), that also provides information about stability margins and closed-loop behavior (resonance, bandwidth and steady-state tracking errors). Pure delays can be directly introduced in the open-loop frequency response plots with an option of *fresp*, or taken into account by means of Padé approximants, that are provided by the application *deftf*. Stability margins may be directly computed with *gpmarg*, that uses a closed-term computational procedure.

5. The root locus. Definition of root locus. Properties. The root contour.

TFI-based lab: Application *rootl* directly plots the root locus of a given transfer function, with the asymptotes and, possibly, the constant damping coefficient loci. It also provides information on the most important features of the locus, like branching points and displays numerical parameters of the asymptotes. It also makes it possible to complete a root locus with one or several root contours drawn in different colors, and to derive the value of the gain corresponding to specific root locations with the mouse.

6. Compensator and regulator design. The most important compensators: lag, lead, lead-lag, bridget T. The standard regulators: P, I, PD, PI, PID. The analytic design of regulators (model-reference). Pole assignment and two-degree-of-freedom regulators.

TFI-based lab: The basic applications for compensator and regulator design are *regnich* and *pidnich*, that make trial-and-error procedures very fast, by directly selecting all the possible solutions with the mouse on Nichols diagrams and comparing them in the time or frequency domain. Another computer aid for design is *regrootl*, that is based on direct selection of the compensator zeros and poles locations with the mouse, while the gain is selected by clicking on the root locus of the overall system. Trial-and-error design of lag and lead compensators and PI, PD or PID regulators is also provided by *lagc*, *leadc* and *pidc*, that refer to Bode diagrams. Reference transfer functions for model-reference design are provided by *deftf* (Bessel and Butterworth filter type, with the possibility of inserting a suitable zero for type 2 regulator synthesis). In connection with model-reference design, that often requires perfect pole-zero cancellation, *factf*, *defactf* and *select* programs are very useful. Complete pole assignment by the Diophantine equation, by also considering a fixed regulator part, is provided by *regdph*. Robustness of stability of any feedback regulator in presence of plant parameter variations can be checked by using *robpar*.

7. Nonlinear systems. The describing function method. Stability criteria. Systems using relays.

TFI-based lab: Application *nlsim* provides the time response of a nonlinear system to any input (given in terms of its Laplace transform), while *descrf* deals with describing function analysis, and provides limit cycles for any linear system connected in a feedback loop with an algebraic nonlinearity of some standard types; it also allows taking a finite delay into account.

8. Sampled-data control systems. The $\mathcal{Z}$-transform of a sampled signal. The aliasing phenomenon. The transfer function of a sampled-data system. Frequency response of a discrete-time system. Nyquist stability criterion and stability margins. Root locus. Using the w-plane equivalence in design. Zero and first-order hold in discrete-to-continuous conversion.

TFI-based lab: Application *convert* provides the $\mathcal{Z}$-transform of a discrete-time signal obtained by sampling a continuous-time signal given in terms of Laplace transform. It also considers the zero-order and first-order hold equivalence. Application *wplane* provides the w-plane equivalent if applied to a discrete-time transfer function or the inverse w-plane equivalent if applied to a continuous-time transfer function. Applications *invtr*, *tresp*, *fresp*, *routh*, *rootl*, *regnich* consistently work also in the discrete-time case, while *pidd* is used instead of *pidc* for designing discrete-time PID regulators. Program *perftra* provides an interesting design method for the feedforward part of a two-degree-of-freedom digital controller that achieves almost perfect tracking also in the nonminimum-phase case.

Chapter 2

Installing, setting, and starting

2.1 How to install

The suggested installation procedure is the following.

1. Enter Matlab and type the command "matlabroot" to know the name of the Matlab root directory and "which startup" to check if the file *startup.m* already exists and, if so, to know its path. Usually the matlabroot is *c:\matlab* and *startup.m* does not exist. Exit Matlab.

2. Create the directory [matlabroot]*toolbox**intp* (where [matlabroot] denotes the information obtained at step 1) and copy into it all the files from *intp.zip*.

3a. Add the directory where the files have been copiep to the matlabpath with the path browser accessible from the Command Window or use the following alternative procedure.

3b. If the file *startup.m* does not exist, edit a file with this name in the directory [matlabroot]*toolbox**local* with the text line:

```
path([path,';',matlabroot,'\toolbox\intp'])
```

that includes the directory *intp* in the matlabpath when Matlab is entered. If *startup.m* already exists, simply add the above text line to the existing text.

NOTE. The name *intp* for the directory where the TFI files are to be copied is obligatory to obtain a correct operation.

2.2 How to get started

TFI produces numerous data files of the type **.mat* (for instance, those with the transfer functions). It is advisable, although not strictly necessary, to use a special work directory for TFI, different from that of Matlab. Suppose the Matlab work

directory is *c:\matlab\work* and that of TFI is *c:\matlab\workint* (this is an example: the names and paths of both are completely arbitrary). Of course, both these directories must exist when entering TFI for the first time. We recall that the Matlab work directory can be specified:

- as a property of the Matlab icon in the Windows environment;
- with a change of directory using the file *startup.m*.

Since setting up the icon property is far from being simple and unified, we insist upon the second way, that is very straightforward: simply add in file *startup.m*, besides the path statement considered in the above installation procedure, the text line:

```
eval(['cd ',matlabroot,\work'])
```

that changes the work directory, overriding the icon property.

The most important TFI settings (in particular, the name of the TFI work directory) are contained in the file *set_tfi5.mat*, repeated in both the Matlab and TFI work directories, that does not exist at the first TFI call. When the Matlab work directory has been suitably set up, from Matlab (with prompt ») enter "tfi" or "intp": the following appears:

```
**** passing to the TFI environment - wait, please
**** file set_tfi#.mat has not been found
     the current work directory is : C:\MATLAB\WORK
     do you want to confirm this as a directory
     from which TFI is invoked ? (1) :
```

If we simply press the return key, we go back to the Matlab environment. On the other hand, on entering 1 we obtain:

```
**** INFORMATION ON THE TFI ENVIRONMENT
the Matlab work directory is C:\MATLAB\WORK
the TFI work directory is    C:\MATLAB\WORK
the matlabpath IS NOT REDUCED
the figure background IS BLACK
the figure locations ARE NOT PERMANENTLY STORED
legend in figures IS NOT ACTIVATED

use "startint" from TFI to change the TFI environment

**** press any key to continue
```

When any key is pressed, the file *set_tfi5.mat*, that contains the above settings, is stored in the Matlab work directory and the Command Window is switched to the TFI environment (the new prompt `>` appears). It is necessary to enter "startint"

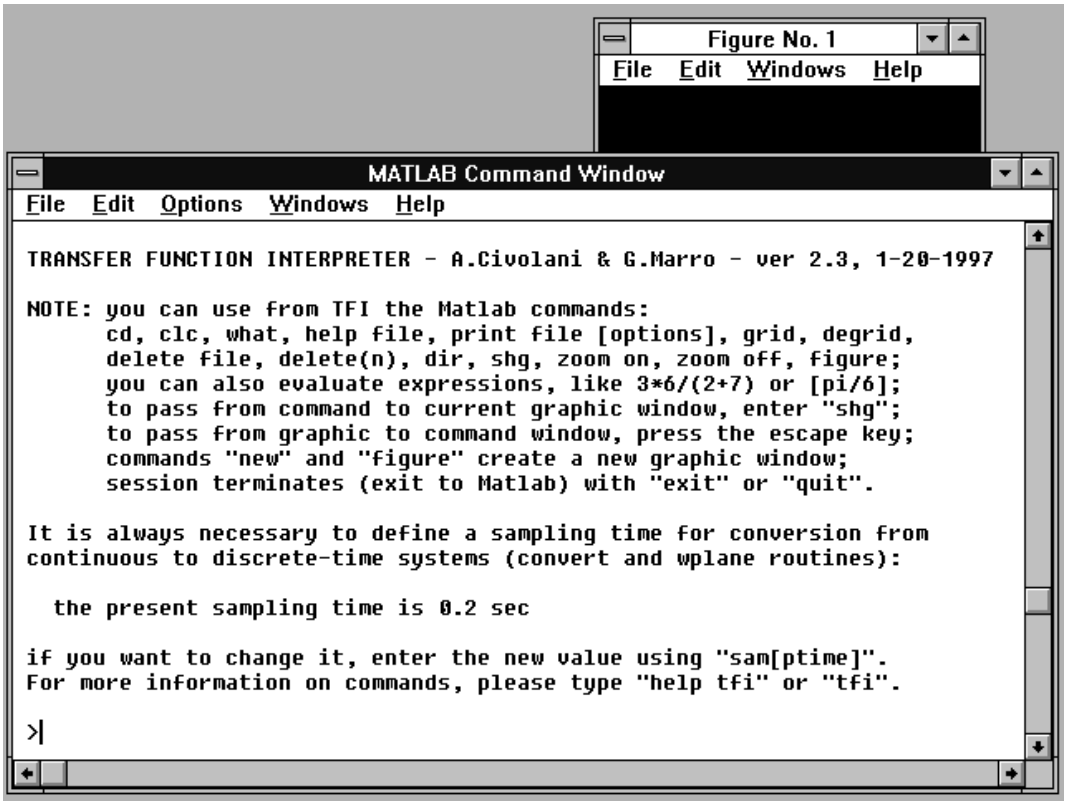

Figure 2.1: When TFI is first entered from Matlab with command "tfi" or "intp", the Command Window layout must be adjusted with the mouse as shown. This figure refers to a low-resolution monitor: in the higher-resolution cases the Command Window location may be different. The figure background color is black, but can be permanently changed with application "startint" or temporarily with command "whitebg".

immediately to change the settings, in particular to switch the work directory to *c:\matlab\workint*. This completes the installation in the case of the example referred to. It is possible to call TFI from several directories, thus obtaining as many files *set_tfi5.mat*, possibly each with a different setting. In any case the call directory, whatever it is, is recovered on exit from TFI to Matlab.

2.3 How to continue and quit

Continuing with the first-time session, or when "tfi" or "intp" is entered from Matlab another time, we should obtain the layout of the Command Window shown in Fig. 2.1. If the layout is different, it is necessary to use the mouse to change size and/or location of the Command Window, in order that the small figure window

located in the upper-right corner of the screen is partially visible. To exit from TFI to Matlab at the end of a TFI session, enter "exit" or "quit". The Matlab work directory and the complete matlabpath are automatically restored.

2.4 How to change some TFI settings

Command "startint" from TFI can be used:

- to define the work directory of TFI;
- to reduce the matlabpath in the TFI environment if necessary;
- to set the background color of the figures as black or white;
- to enable/disable storing of figure locations for next sessions;
- to activate or disactivate the legend in figures.

See the specification of this command for more information.

An important TFI setting concerns the Command Window and uicontrol fonts. In Matlab 4 they are both accessible from the Command Window control bar as **Options / Command Window Font** and **Options / Uicontrol Font**. To obtain correct display of transfer functions it is necessary to use, in the Command Window, a character with uniform letterspacing, like Fixedsys or Courier New. We suggest Fixedsys (size 9) for the Command Window and MS Sans Serif (size 8, style bold) for uicontrol. In Matlab 5 the Command Window font setting is accessible as **File / Preferences / Command Window Font**, while that of uicontrol windows is not accessible (it is set in the file *startint.m*, that is run when TFI is entered from Matlab). We suggest setting the Command Window font as Fixedsys (size 11).

Command "whitebg" from TFI can be used to change the background color of the figures, thus making it different from the default, that is set with the application *startint*. This command has a toggling effect. The default value of Matlab 4 is black, that has the following advantages:

- it emits less radiation from the screen;
- it uses pure basic colors for plots, that result in a neat trace in color prints.

Nevertheless, Matlab 5 came with a white background, whose advantages are:

- the screen appears like in other Window applications (e.g., Word);
- it is more suitable for bitmap reproductions of the screen.

Since the figure background color is a matter of taste, in the TFI environment its choice is left to the user. When it is changed, all the colors of plots and messages are accordingly slightly modified, to reinforce visibility and readability. The corresponding RGB color tables, one for plots and one for messages, both repeated for black and white backgrounds, are available as the matrices *A1*, *A2*, *A3* and *A4* in the file *coltbl.m* and should be edited and modified by the user if necessary (color shade may depend on monitor). For correct color reproduction it is necessary that the monitor driver supports at least 256 colors.

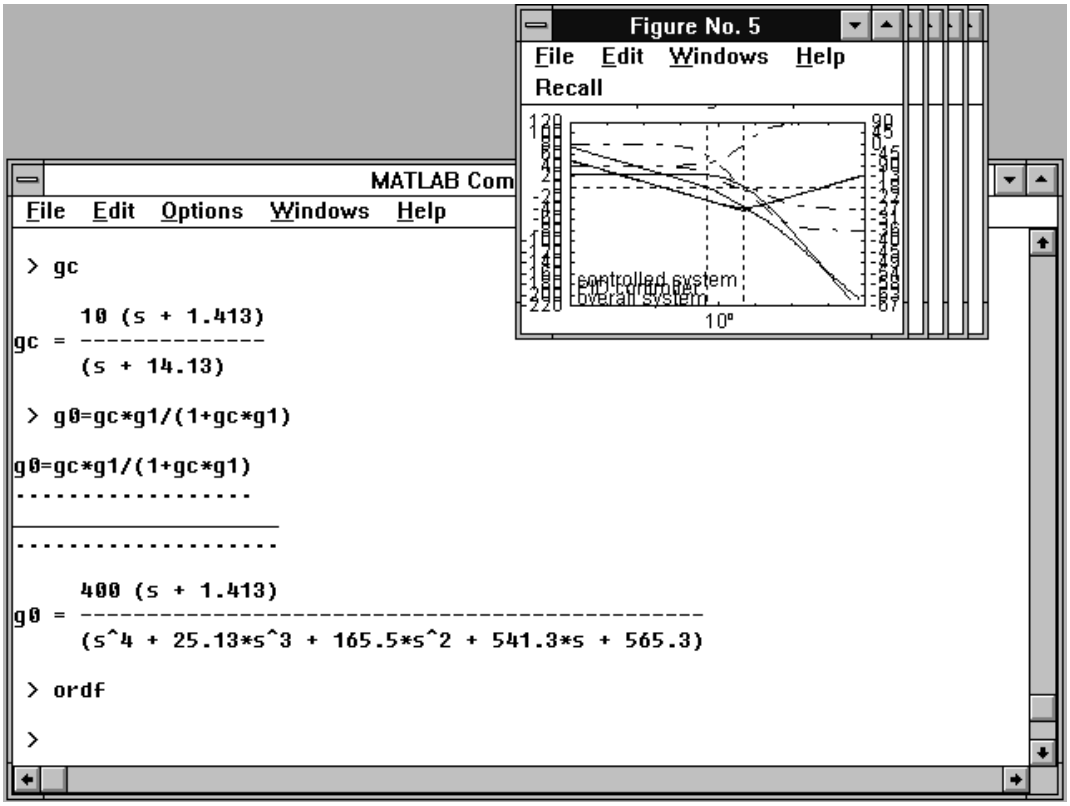

Figure 2.2: In a design session it is possible to open several graphic windows with figures by simply using the command "new". With "ordf" they are reduced to small size and arranged as shown. This enables selection and enlargement with mouse or keyboard (commands "fig1","fig2",..., "lar") to see each figure at full size again; "sma" restores previous size and location. Note that all the figures appear with white background in this case.

The file *startint.m* specifies the default font name, size and style of the text messages in figures and, separately, of the axes graduations, titles and x/y labels. In the Matlab 5 version it also defines those used in uicontrol text. This file should be edited and modified if desired (in some cases a different font size or the style bold may be preferable for text messages displayed in figures or axes graduations).

2.5 How to manage figures

When a graphic application is run from TFI, the corresponding figures (one or more) generally appear at full size, but, when it is quitted, they arev reduced as small on the upper-right corner of the screen, as shown in Fig. 2.2.

You can use the command "new" to create a new figure window, thus maintaining

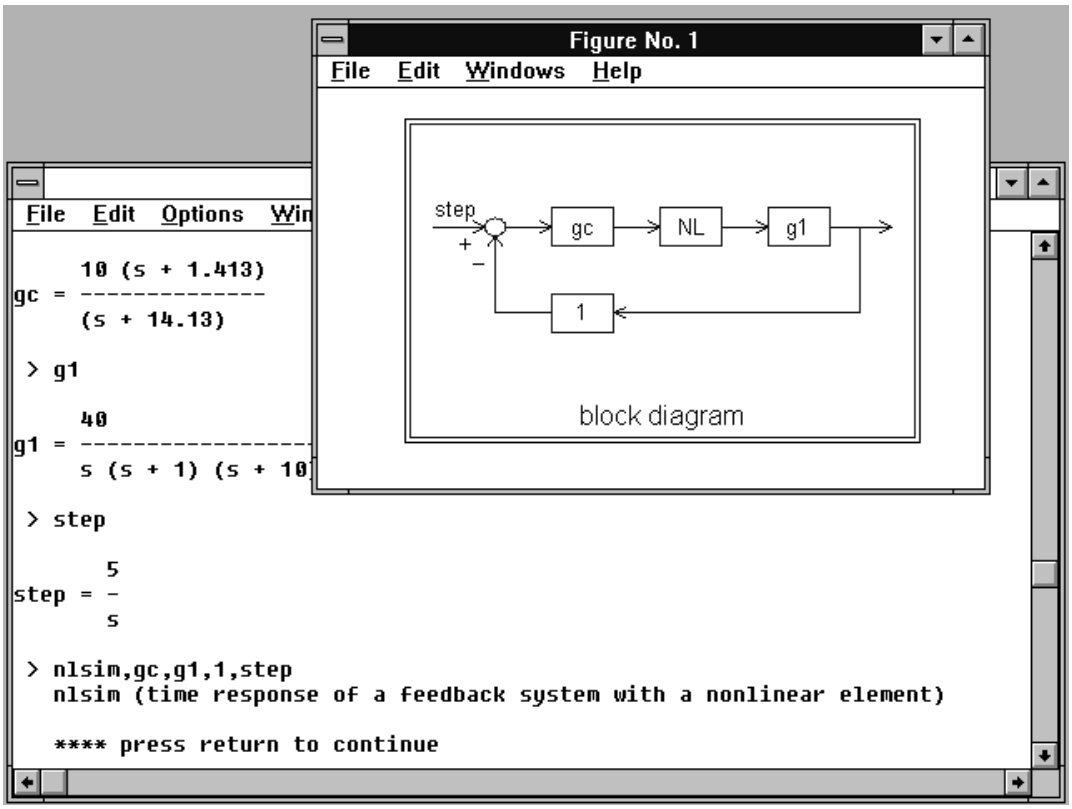

Figure 2.3: A block diagram at medium size is often temporarily shown to check the command correctness. The figure refers to application *nlsim*, that computes and plots the time response of a control loop with a nonlinear element to any input signal given through a Laplace transform.

the previously generated ones for possible comparisons, "fig1", "fig2",... to select a particular figure, "ordf" to arrange all figures like a card deck, as shown in Fig. 2.2. Commands "sma[ll]", "med[ium]" and "lar[ge]" – or, to obtain intermediate sizes, "enl[arge]" and "red[uce]" – change the size of the current figure, "last" selects the highest-numbered figure, "delf" deletes all the figures, "shg" shows the selected figure at full size, while the *Esc* key can be used to recover the Command Window from any figure window. Of course, you can also use the mouse to manage figures according to the Windows rules.

When a command is entered from TFI, a medium-size graphic window with the corresponding block diagram is often shown as in Fig. 2.3 to check if the syntax used is correct.

In most graphic applications there are some facilities added at the top command line of the figures obtained: **Recall**, that makes it possible to see the names of

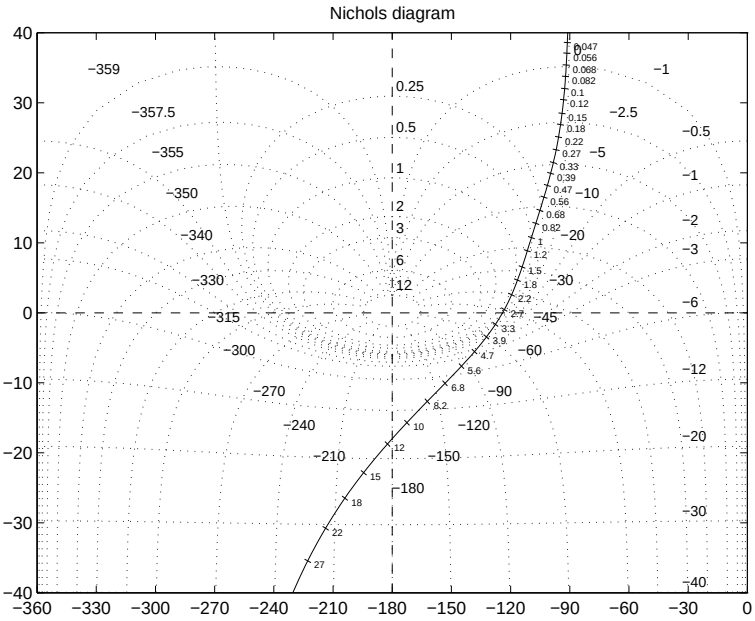

Figure 2.4. Nichols diagram exported with command "print -deps file.eps".

the transfer functions involved in the application and to temporarily display these transfer functions, superimposed on the figure in the same color as the corresponding plots, **Grid on/off**, that enables or disables the grid in the figure, and **Zoom on/off**. When using high-resolution monitors the current figure and the Command Window are contemporarily visible on the screen, and many figures may be adjusted with the mouse to appear together. Their sizes and locations are permanently saved in the hard disc, until the command "resfiglo" (reset figure location) is used.

IMPORTANT NOTICE: if when entering a graphical application the figure dimension and/or location appears to be non-correct (for instance, the figure is always "small"), this is due to improper previous exit from the same application: use the command "resfiglo" to recover the default figure settings.

2.6 How to print figures

Some applications of the TFI environment produce high-quality figures and provide useful facilities, like easy change of axes scales, drawing multiple plots in different colors, and simple choice of the fonts for graduation of plots. It may be required to translate the figures into prints or files in order to include them in class notes or technical papers. To this end you can use the Matlab *print* facility from the

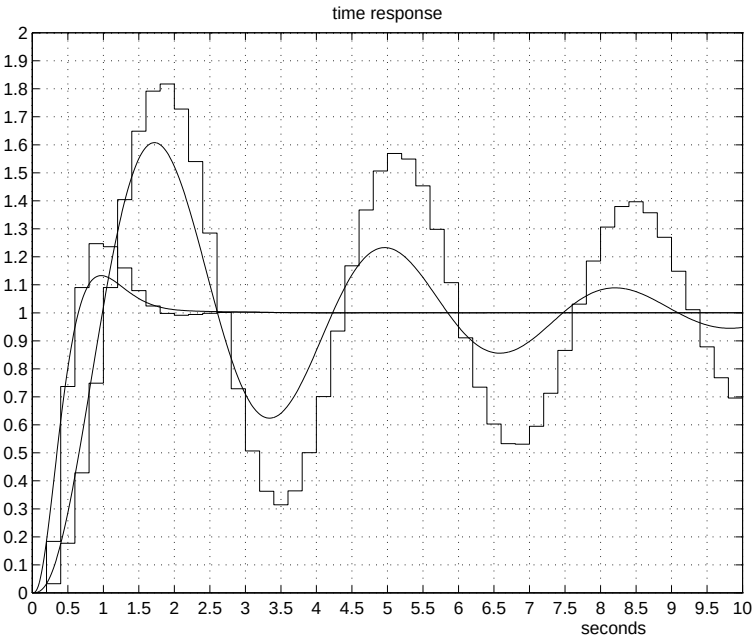

Figure 2.5. Step responses exported with command "print -deps file.eps".

Command Window (enter "help print" from TFI or Matlab to see all the options). For instance, "print -dps xx.ps" generates the PostScript file *xx.ps* that contains the current figure, while "print -dps -append xx.ps" appends the figure to others in the same file. Let us recall that the option *-dpsc* instead of *-dps* provides color printing, and *-deps*, *-depsc* produce Encapsulated PostScript, that can be used to insert figures in Latex or Microsoft Word documents. Examples are shown in Fig. 2.4 and Fig. 2.5. You can also use the Windows printing facility from the graphic window. However, the quality of the figures so obtained is worse. To pass from the Command Window to the current graphic window, enter "shg" (from TFI) or use the mouse on the **Windows** option shown at the top of the screen.

The figures of this manual, that reproduce complete screen layouts also with many windows open, came from the Matlab 4 version, set for white figure background and were exported by using the *PrtSc* key and the *Paste* facility in Microsoft Word. For better black-and-white results, the plots and messages were obtained with all the colors converted to black by simply changing the *A3* and *A4* color tables in the file *coltbl.m*. The figures with plots at full-screen size have been obtained with a low-resolution monitor (640×480).

Chapter 3

TFI and Its Applications

3.1 Tfi

The command

» `tfi` *(enter)*

from Matlab produces access to the Transfer Function Interpreter environment: a message is displayed and a new prompt appears on the screen (> instead of »), to inform that the syntax of the Command Window has been changed.

3.1.1 Operation and Examples

A transfer function is entered by typing a string of characters, where the transfer function name is first specified, followed by = and by characters denoting numbers, brackets, operators, and the symbol s or z.

When such a string is entered, TFI first operates a check on the balancing of brackets, then begins the lexical analysis. This is pointed out on the screen by a line of dots scanning the analysis, letter by letter. The dots are the same number as the characters of the string. When a mistake is encountered, the sequence of dots stops and an error message is displayed (see forward). When the lexical analysis is successfully performed, the display of a solid line denotes the beginning of syntactical-semantic analysis and a second sequence of dots points out that it is in progress. Also in this case the interpretation may be interrupted by an error message in the case of illegal operations. Let us consider some examples.

The transfer function

$$g1(s) = \frac{40}{s\,(s+1)\,(s+10)}$$

is entered by typing:

```
> g1=40/(s*(s+1)*(s+10))   (enter)
```

First, it is interpreted. Interpretation is shown as:

```
g1=40/(s*(s+1)*(s+10))
......................
--------------------
....................
```

Then, it is saved in file $g1.mat$, and displayed as:

```
     40
g1 = ------------------
     s (s + 1) (s + 10)}
```

Assume that $g1(s)$ is the transfer function of a controlled system. The transfer function $gc(s)$ of a lead compensator can be entered as:

```
> gc=10*(s+1.413)/(s+14.13)   (enter)
```

and shown in the display as:

```
     10 (s + 1.413)
gc = --------------
     (s + 14.13)
```

Then, the open-loop overall transfer function $g2(s)$ is computed by typing:

```
> g2=g1*gc   (enter)
```

Going on with the example, the overall feedback system transfer function $g0(s)$ is computed by:

```
> g0=g2/(1+g2)   (enter)
```

and displayed as:

```
     400 (s + 1.413)
g0 = ------------------------------------------------
     (s^4 + 25.13*s^3 + 165.4*s^2 + 541.3*s + 565.2)
```

TFI recognizes the symbols `+,-,*,/,^` : power refers to integer numbers only, both positive and negative.

You can have any defined transfer function displayed by simply entering its name:

`> g0` *(enter)* , or, if you want it in zero-pole form, by typing:

`> g0=` *(enter)* , that produces:

```
     400 (s + 1.413)
g0 = -----------------------------------------------
     (s + 1.746) (s + 17.25) [(s + 3.068)^2 + 3.059^2]
```

or else, in time-constant form, by typing:

`> g0:` *(enter)*

that produces:

```
     1 (0.7077*s + 1)
g0 = ------------------------------------------------------------
     (0.05798*s + 1) (0.05327*s^2 + 0.3269*s + 1) (0.5729*s + 1)
```

Finally, by typing:

`> g0;` *(enter)*

a full-size plot of the zero-pole map is obtained.

A long transfer function can be split into many lines, as follows:

`> g3=10*(s+2-g1*>` *(enter)*

`> (s+4))/((s^2+s+10)*(g1+g2))` *(enter)*

The dummy variable *ans* is automatically used if the left-hand member is not specified; for instance, entering:

`> g0*(1+g2)` *(enter)*

produces:

```
      400 (s + 1.413)
ans = ------------------------------
      s (s + 1) (s + 10) (s + 14.13)
```

The variable *ans* is saved like a transfer function, so that the request:

`> g2-ans` *enter*

produces:

```
      0
ans = -
      1
```

The defined transfer functions can be evaluated at specified values of s or z, as follows:

```
> g0(0)  (enter)

  value of g0(0): 1
  abs.value: 1; angle: 0 degrees (0 radians)

> g0(10)  (enter)

  value of g0(10): 0.07919
  abs.value: 0.07919; angle: 0 degrees (0 radians)

> g0(j)  (enter)

  value of g0(1i): 1.014-0.3078i
  abs.value: 1.06; angle: -16.89 degrees (-0.2947 radians)

> g0(10*j)  (enter)

  value of g0(10i): -0.1938-0.03008i
  abs.value: 0.1961; angle: -171.2 degrees (-2.988 radians)
```

TFI also makes it possible to perform some numerical computations with Matlab's syntax. If the string to be computed begins with a number, it can be directly entered since in this case the Matlab interpreter is automatically called in; if not, it is necessary to put the expression in square brackets. Examples:

```
> 78*9  (enter)

ans =

   702

> [sin(5*pi)]  (enter)

ans =

  6.125e-016}
```

The dummy variable *ans* used in this case comes from Matlab's interpreter, not from the TFI interpreter, and so it is temporary.

Syntax errors such as unbalanced brackets and illegal symbol sequences are pointed out by specific error messages:

```
 > g101=89/(s+125))   (enter)
 error: unbalanced brackets
```

```
 > g101=89*/(s+125)   (enter)
```

```
g101=89*/(s+125)
........
error: character after operator is illegal
```

```
 > g101=89*(s+125,8)   (enter)
```

```
g101=89*(s+125,8)
.....
error: character "," in the expression
```

You can also handle discrete-time transfer functions by simply using z instead of s while entering them. Since the conversion routines available from TFI require a sampling time T, you are asked to define it at the beginning of the first session: it is permanently saved in the special file *time#.mat* and can be changed at any time in interactive mode by typing:

```
 > samptime   (enter)
```

.

See the specification of this command for more information.

Mixing symbols s and z or continuous and discrete-time transfer functions in a single expression is illegal and causes the following error message:

```
 > g101=89*(s+1258+z)   (enter)
```

```
g101=89*(s+1258+z)
..................
error: "s" & "z" in the same expression
```

Let us see other possible types of errors:

```
 > ss2tf   enter
ss2tf.mat or application ss2tf does not exist
 > ss2tf,g1   (enter)
application ss2tf does not exist
 > g0=h2/(1+g2)   (enter)
g0=h2/(1+g2)
.....
h2.mat does not exist
```

(transfer function *h2* is not defined)

```
 > g3=g2^g1   (enter)
g3=g2^g1
........
--------------------
error: not integer number after ^ !
    > g3=g2^((s+1)/(s+2))   (enter)
g3=g2^((s+1)/(s+2))
...................
--------------------
..............error: not integer number after ^ !
```

Using the TFI transfer functions in the Matlab environment

TFI saves transfer functions of dynamic systems to files of the type *gi.mat*, hence it works with objects rather different from those used in Matlab Command Window and m-files, which usually refer to matrices. In fact, the standard representations of transfer functions in Matlab 4 are *[num,den]* and *[z,p,k]*. It is possible to convert the Matlab to the TFI form and vice versa by using two simple interface programs, named *importf.m* and *exportf.m*, that make exchanges between the two environments possible.

1 - Conversion from [num,den] or [z,p,k] to 'gi':

» `importf(num,den,'gi',[1])` *(enter)* ,
» `importf(z,p,k,'gi',[1])` *(enter)* .

Any one of the two Matlab forms of a transfer function is converted to the TFI form and saved in file *gi.mat*; option [1] allows for saving as discrete-time (default is continuous-time).

Example: the transfer functions $g1(s)$ and $gc(s)$ defined at the beginning of this section can be created in an m-file through the following program lines:

```
z1=[]; p1=[0;-1;-10]; k1=40;
importf(z1,p1,k1,'g1');
zc=-1.413; pc=-14.13; kc=10;
importf(zc,pc,kc,'gc');
```

2 - Conversion from 'gi' to [num,den] or [z,p,k]:

» `[num,den,[str]]=exportf('gi',[1])` *(enter)* ,
» `[z,p,k,[str]]=exportf('gi',[1])` *(enter)* .

The transfer function $gi(s)$ or $gi(z)$ is read from file *gi.mat* and converted to one of the Matlab forms (depending on the number of output arguments). Option [1] makes it possible to get information on transfer function type (continuous or discrete time) in string *str*, whose admissible types are $'s'$ or $'z'$.

In the Matlab 5 version of TFI the commands `sys=^{exportf1}('gi',[1])` and `{importf1}(sys,'gi',[1])` enable conversion from a transfer function *gi* to the corresponding *sys* and vice versa. The output is obtained in factorized form or, with option [1], in polynomial form.

3 - Display of a transfer function:

» `tfg('gi')` *(enter)* ,
» `tfm(num,den,'gi',[1])` *(enter)* ,
» `tfm(z,p,k,'gi',[1])` *(enter)* ,
» `tfm(z,p,k,'gi','1',[1])` *(enter)* .

Function *tfg* displays in the Command Window the transfer function $gi(s)$ or $gi(z)$ saved in file *gi.mat*, while *tfm* displays a transfer function from its Matlab form (polynomial or factorized); in this case the string *'gi'* is simply the name to denote the transfer function in display, while option [1] causes the use of symbol z instead of s, and argument '1' produces display of the time-constant form instead of the factorized form.

4 - Computations on transfer functions:

» `sumgm('gi','gj','gk',opt)` *(enter)* ,
» `prodgm('gi','gj','gk',opt)` *(enter)* ,
» `expgm('gi','gj',h)` *(enter)* .

Computation on transfer functions already available as mat-files can be performed, in the Matlab environment, by means of the above commands: *sumgm* adds ($opt = 0$) or subtracts ($opt = 1$) transfer functions $gi(s)$ and $gj(s)$ or $gi(z)$ and $gj(z)$ and saves the result in file *gk.mat*, *prodgm* multiplies ($opt = 0$) or divides ($opt = 1$) $gi(s)$ and $gj(s)$ or $gi(z)$ and $gj(z)$ and saves the result in *gk.mat*, while *expgm* raises $gi(s)$ or $gi(z)$ to the h-th power, with h integer, and saves the result in *gj.mat*.

Example: it is possible to compute $g0(s) := g1(s)\, gc(s)/(1+g1(s)\, gc(s))$, with $g1(s)$ and $gc(s)$ defined as in the above example referring to *importf*, by using the following m-file, that saves the result in *g0.mat*:

```
prodgm('g1','gc','g0',0);
importf(1,1,'gt');
sumgm('gt','g0','t1',0);
prodgm('g0','gt','g0',1);
tfg('g0') % this line displays the result
```

In the above program transfer function $gt(s)$ is used as temporary storage.

From TFI it is possible to use the following commands (many of them are Matlab commands):

> `cd` displays the name of the current directory
> `cd path` assumes the work directory specified in *path*
> `clc` clears the Command Window
> `clear` removes all the compiled functions from the TFI workspace
> `degrid` removes the grid from the current figure
> `delete file.ext` deletes *file.ext* from the work directory
> `delete(n)` deletes figure n
> `delf` deletes all the figures
> `dir` displays the file names of the work directory in Matlab format
> `enl[arge]` enlarges the current figure by 20%
> `fig`n selects figure n, with $n=1, 2, \ldots$
> `grid` adds a grid to the current figure
> `help file` displays help of *file.m*
> `lar[ge]` enlarges the current figure to full size
> `last` selects the figure with the highest number
> `med[ium]` sets the current figure to medium size
> `new` creates a new figure
> `ordf` arranges all the figures at small size
> `path` displays the current path
> `print file [options]` saves the current figure to *file*
> `red[uce]` reduces the current figure by 20%
> `res[figlo]` resets the figure locations to the default
> `shg` shows the current figure at full size
> `sma[ll]` reduces the current figure to small size
> `tfi` same as *help tfi*
> `what` lists all **.m* and **.mat* files present in the work directory
> `whitebg` changes the background color (black or white) of the figures
> `zoom` toggles the zoom status
> `zoom [on],[off]` switches the zoom facility on or off.

We also recall that the Command Window is recovered from any selected figure by simply pressing the *Esc* key.

The following CAD applications are available in the TFI environment. The part of the command in square brackets may be omitted to speed up writing.

> `con[vert],gi,gj` converts *gi* from *s* to *z* and saves the result as *gj*
> `defa[ctf],gi,gj` defactors *gi* and saves the result as *gj*
> `deft[f],gi` defines *gi* with the mouse or as Bessel, Butterworth, Padé tf
> `des[crf],gi` analyzes a nonlinear system with the describing function
> `fac[tf],gi,gj` factors *gi* and saves as *gj*
> `fre[sp],gi` plots the frequency response of *gi*
> `gpm[arg],gi` displays the (generalized) gain and phase margin of *gi*
> `inv[tr],gi` displays the inverse Laplace (or $\mathcal{Z}$-) transform of *gi*
> `lag[c],gi,gj` designs a lag compensator (with the Bode diagrams)
> `lea[dc],gi,gj` designs a lead compensator (with the Bode diagrams)
> `mak[eleg]` creates or cleans a legend in the last figure
> `nls[im],gi,gj,gk,gw` plots the time response of a nonlinear feedback system
> `per[ftra],gi,gj,gk,gw` designs a digital perfect tracking compensator
> `pidc,gi,gj` designs a PID regulator (with the Bode diagrams)
> `pidd,gi,gj` designs a discrete-time PID regulator (with the Bode diagrams)
> `pidn[ich],gi,gj` designs a PD, PI or PID regulator (with the Nichols diagram)
> `regd[ph],gi,gj,gk,gw` designs a regulator by pole assignment
> `regn[ich],gi,gj` designs a lead or lag compensator (with the Nichols diagram)
> `regr[ootl],gi,gj` designs a compensator or a regulator with the root locus
> `rob[par],gi,gj,gk,gw` analyzes robustness versus parameter variations
> `roo[tl],gi` plots the root locus of *gi*
> `rou[th],gi` displays the closed-loop stability intervals of *gi*
> `sam[ptime],T` defines the current sampling time for discrete-time systems
> `sel[ect],gi,gj` interactively selects factors from *gi* and saves as *gj*
> `sta[rtint]` defines some TFI environment settings
> `tfe[val],gi` evaluates *gi* at a particular value of *s* or *z*
> `tre[sp],gi` plots the impulse or step response of *gi*
> `wpl[ane],gi,gj` converts from *z* to *w*-plane or from *w* to *z*-plane
> `zpp[lots],gi,gj,gk,gw` plots the zero-pole maps of some transfer functions.

All the above applications are briefly presented in alphabetic order in the following pages of this manual. You can also use "help *name*" from the Command Window to obtain concise information on their use and syntax.

NOTE: It is possible to quit most TFI applications by entering "0" from any menu.

3.2 Convert

The command

`> convert` *(enter)*

converts the continuous-time transfer function $gi(s)$ to the discrete-time transfer function $gj(z)$, that is displayed and saved in the current work directory.

3.2.1 Recall

It is possible to choose one of the following three modes of conversion, that are the most important for discrete-time control system analysis and design.

$\mathcal{Z}$-transform of the sampled inverse $\mathcal{L}$-transform. The inverse $\mathcal{L}$-transform of $gi(s)$ is first obtained through partial fraction expansion (see the *Recall* of *invtr* for details). It is expressed by

$$gi(t) = K_0\,\delta(t) + \sum_{i=1}^{h}\sum_{\ell=1}^{r_i} \frac{K_{i\ell}}{(r_i-\ell)!}\,t^{r_i-\ell}\,e^{p_i t}\ ,$$

where $p_1, \dots\ p_h$ denote the distinct poles of $gi(s)$ and $r_1, \dots, r_h$ their multiplicities. Constant K_0 is zero in this case since $gi(s)$ is assumed to be strictly proper. By setting $t = kT$ $(k = 0, 1, \dots)$ we obtain the sequence, whose $\mathcal{Z}$-transform has to be determined, as a linear combination of terms of the general type $k^i\,e^{pkT}$ or $k^i\,q^k$, where $q := e^{pT}$ denotes the discrete-time pole corresponding to the continuous-time pole p.

We take the $\mathcal{Z}$-transform of this linear combination term by term by using the following table:

$$\begin{bmatrix} \dfrac{z}{z-q} \\ q\,\dfrac{z}{(z-q)^2} \\ q^2\,\dfrac{z}{(z-q)^3} \\ \vdots \\ q^n\,\dfrac{z}{(z-q)^{n+1}} \end{bmatrix} \overset{\mathcal{Z}}{\underset{\mathcal{Z}^{-1}}{\longleftrightarrow}} \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & -\dfrac{1}{2} & \dfrac{1}{2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \alpha_{n+1,1} & \alpha_{n+1,2} & \dots & \alpha_{n+1,n} \end{bmatrix} \begin{bmatrix} q^k \\ k\,q^k \\ k^2\,q^k \\ \vdots \\ k^n\,q^k \end{bmatrix} .$$

The elements of the i-th row ($i \geq 2$) of the matrix on the right are the products of $1/(i-1)!$ by the coefficients, ordered by increasing powers of x, of the polynomial $x\,(x-1)\,\ldots\,(x-i+1)$, having the integers $0, 1, \ldots, i-1$ as roots. The matrix is lower triangular with nonzero elements on the main diagonal, hence nonsingular and easily invertible.

The above relation between terms of the types $q^i z/(z-q)^{i+1}$ and $k^i q^k$ solves both the problem in hand (deriving the $\mathcal{Z}$-transform of the sampled inverse $\mathcal{L}$-transform of a given ratio of polynomials in s) and that of deriving the inverse $\mathcal{Z}$-transform of a ratio of polynomials in z. In fact, it is equivalent to the well-known binomial coefficient form of $z/(z-q)^{i+1}$ (see the *Recall* of *invtr*). However, it cannot be used for the inverse $\mathcal{Z}$-transform if $q=0$; in this case it must be replaced by

$$z^{-i} \quad \underset{\mathcal{Z}^{-1}}{\overset{\mathcal{Z}}{\longleftrightarrow}} \quad \delta(k-i) \ .$$

Zero-order hold equivalence. indexzero-order hold The above mathematical background still applies. In fact, the result is obtained as the product of $(z-1)/z$ with the $\mathcal{Z}$-transform of the sampled inverse $\mathcal{L}$-transform of $gi(s)/s$. In this case, owing to division by s, $gi(s)$ is not required to be strictly proper.

First-order hold equivalence. The result is obtained as the product of $(z-1)^2/z^2$ with the $\mathcal{Z}$-transform of the sampled inverse $\mathcal{L}$-transform of $[1/s + 1/(T\,s^2)]\,gi(s)$; $gi(s)$ is not necessarily strictly proper.

A delay $e^{-t_0 s}$ applied to $gi(s)$ can be easily taken into account during conversion. Let us write

$$e^{-t_0 s} = e^{-hT}\,e^{-\alpha T} \ ,$$

where $h \geq 1$ denotes an integer (the maximum number of sampling periods contained in t_0) and α, $0 < \alpha < 1$, a real number (corresponding to a part of the sampling period). An equivalent relation is

$$e^{-t_0 s} = e^{-(h+1)T}\,e^{mT} \ ,$$

with $m := 1-\alpha$, $0 < m < 1$. The time delay $(h+1)T$ is taken into account by simply dividing by z^{h+1} the discrete-time function $gj(z)$ obtained from $e^{mT}\,gi(s)$.

Factor e^{mT} shifts the time function $gi(t)$ in advance by mT, hence it can be taken into account by changing the generic term $t^i\, e^{pt}$ of the expansion of $gi(t)$ [derived from the partial fraction expansion of $gi(s)$] into $(t+mT)^i\, e^{p(t+mT)}$, that is sampled as

$$(k+m)^i\, T^i\, e^{-p(k+m)T} \quad (k=0,1,\dots)\ ,$$

or, setting $q{:=}e^{pT}$ as before,

$$q^m\, T^i\, (k+m)^i\, q^k \quad (k=0,1,\dots)\ ,$$

or also, by expanding the binomial $(k+m)^i$,

$$q^m\, T^i \sum_{\ell=0}^{i} \binom{i}{\ell}\, m^{i-\ell}\, k^\ell\, q^k \quad (k=0,1,\dots)\ .$$

Thus, a linear combination of terms of the general type $k^\ell\, q^k$, each easily transformable with the previously reported table, has been obtained again.

3.2.2 Operation and Examples

Let us consider the function

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ .$$

The first display after the command "convert,gi,gj" is:

```
1 - Z-transform of the sampled inverse L-transform
2 - zero-order hold equivalence
3 - first-order hold equivalence

your choice : 2

enter a possible delay - default is zero :

conversion from continuous-time to discrete-time;
the assumed sampling time is 0.2 sec
the assumed delay is 0
```

The result is:

```
     0.03279 (z + 0.1442) (z + 2.342)
gj = --------------------------------
     (z - 0.1353) (z - 0.8187) (z - 1)
```

If conversion is repeated for a discrete-time transfer function, e.g. by entering "convert,gj,gk", the following message appears:

```
**** warning: discrete-time system; conversion is not necessary
```

and function gj is displayed for a check.

If the function to be converted has negative relative degree, i.e., the degree of the numerator greater than that of the denominator, conversion is not possible and the program ends after the message:

```
**** error: m > n in gi
```

while, if it is not strictly proper with option 1, the program ends with:

```
**** error: m = n in gi with option 1
```

Note that after the mode of conversion has been chosen it is possible to introduce a delay t_0, i.e., a factor $e^{-t_0 s}$ applied to the continuous-time transfer function; for instance, a delay of 0.5 sec in function $gi(s)$ previously defined produces:

```
     0.005149 (z + 0.02337) (z + 0.5893) (z + 13.97)
gj = -----------------------------------------------
     z^3 (z - 0.1353) (z - 0.8187) (z - 1)
```

If the delay is negative, the following message appears:

```
**** error: negative delay is not allowed
```

and the request for delay is repeated.

3.3 Defactf

The command

`> defactf` *(enter)*

displays and saves in the current work directory as $gj(s)$ or $gj(z)$ the polynomial form of a given transfer function $gi(s)$ or $gi(z)$.

3.3.1 Recall

Let us consider the transfer function

$$gi(s) = \frac{100}{(s+1)\left((s-5)^2+3^2\right)^2} \ ;$$

by means of some simple polynomial manipulations (raising to a power and product), it is easily put in the ratio-of-polynomials form

$$gj(s) = \frac{100}{s^5 - 19\,s^4 + 148\,s^3 - 512\,s^2 + 476\,s + 1156} \ .$$

3.3.2 Operation

In the above case the display appears as follows:

```
     100
gi = -------------------------
     (s + 1)(s^2 - 10*s + 34)^2

     100
gj = -----------------------------------------------
     (s^5 - 19*s^4 + 148*s^3 - 512*s^2 + 476*s + 1156)
```

3.4 Deftf

The command

`> deftf` *(enter)*

defines and saves in the current work directory as $gi(s)$ or $gi(z)$ a transfer function whose zeros and poles are located with the mouse, or a transfer function of the standard types *Bessel filter*, *Butterworth filter* or *Padé delay*.

3.4.1 Recall

Zero and pole assignment with the mouse does not require any recall and will be described in detail in the following *Operation* section.

The Bessel and Butterworth filters are used in the control systems analytical design as standard references for the closed-loop transfer function to be realized. The Padé delay functions are widely used as rational approximations of the finite delay $e^{-t_0 s}$, thus making handling closed-loop systems including delays possible with the standard methods used for rational transfer functions.

Bessel filter. The Bessel filter can be viewed as an approximation of the unit delay with a rational transfer functions having no zeros. From the Maclaurin expansions of the hyperbolic cosine and sine:

$$\begin{aligned} \cosh s &= \frac{e^s + e^{-s}}{2} = 1 + \frac{s^2}{2!} + \frac{s^4}{4!} + \dots \ , \\ \sinh s &= \frac{e^s - e^{-s}}{2} = s + \frac{s^3}{3!} + \frac{s^5}{5!} + \dots \ , \end{aligned}$$

we derive

$$e^{-s} = \frac{1}{e^s} = \frac{1}{\cosh s + \sinh s} = \frac{1/\sinh s}{\cosh s/\sinh s + 1} \ ,$$

and, by using again the above Maclaurin expansions, we obtain the continued fraction

$$\operatorname{cotanh} s = \frac{\cosh s}{\sinh s} = \frac{1}{s} + \cfrac{1}{\cfrac{3}{s} + \cfrac{1}{\cfrac{5}{s} + \cfrac{1}{\cfrac{7}{s} + \dots}}} \ .$$

By breaking off this fraction after n terms we define a series of rational functions $P_n(s)/Q_n(s)$ $(n=1,2,\dots)$ whose numerators and denominators, apart from a constant $b_{n,0}$ equal for both, tend to the expansions of cosh$s$ and sinh$s$, respectively. By substituting $P_n(s)$ and $Q_n(s)$ in the above expression of e^{-s} we obtain the approximation

$$e^{-s} \simeq \frac{b_{n,0}}{P_n(s)+Q_n(s)} = \frac{b_{n,0}}{B_n(s)} \ ,$$

where $B_n(s) := P_n(s)+Q_n(s)$ is, by definition, the *Bessel polynomial* of order n and $b_{n,0}$ its constant term.

It is easily seen that $B_0(s)=1$, $B_1(s)=s+1$ and that the subsequent Bessel polynomials can be derived with the recursion formula

$$B_n(s) = (2n-1)\,B_{n-1}(s) + s^2\,B_{n-2}(s) \ .$$

Approximations of the generic delay $e^{-t_0 s}$ are obtained by substituting $t_0 s$ for s in $b_{n,0}/B_n(s)$. However, the frequency response functions $b_{n,0}/B_n(j\omega)$ do not have all the same cutoff or corner frequency (the angular frequency corresponding to the intersection of the tangents at infinity to the Bode diagram of gain). A series of filters with corner frequency equal to one is obtained with a suitable frequency scaling; let us define the new coefficients $\alpha_{n,i}$ $(i=n,\dots,0)$ by means of

$$\alpha_{n,i} = \frac{b_{n,i}}{\omega_r^{n-i}} \quad (i=0,\dots,n-1) \ , \quad \text{with} \ \ \omega_r := \sqrt[n]{b_{n,0}} \ ;$$

these are the coefficients of the denominators of the *Bessel filters* of unit corner frequency; the numerators are set equal to one. An arbitrary corner frequency ω_0 is obtained by substituting s/ω_0 for s.

Butterworth filter. Let us consider the function

$$F_n(s) := \frac{1}{s^{2n}+(-1)^n} \ ,$$

with n positive integer, whose poles, expressed by

$$p_k = e^{j\frac{2k+n-1}{n}\frac{\pi}{2}} \quad (k=1,2,\dots,2n) \ ,$$

are located on the unit circle, uniformly distributed at angular distance π/n and symmetric with respect to the real axis. Of course $F_n(s)$ is unstable, but can be factorized as $F_n(s) = F_n^-(s)\,F_n^+(s)$, where $F_n^+(s)$ has order n and all poles in the right half-plane while $F_n^-(s)$, also of order n, has all poles in the left half-plane; this latter is, by definition, the transfer function of the *Butterworth filter* of order n and unit corner frequency.

The Butterworth filter $G_n(s)$ of order n and corner frequency ω_0 is given by

$$G_n(s) := F_n^-\Big(\frac{s}{\omega_0}\Big) = \omega_0^n \prod_{k=1}^{n} \frac{1}{s - \omega_0\, p_k} \ .$$

Since the poles of $F^+(s)$ and $F^-(s)$ are symmetric with respect to the imaginary axis we have

$$|F_n^-(j\omega)| = |F_n^+(j\omega)| = \sqrt{F(j\omega)} = \frac{1}{\sqrt{1+\omega^{2n}}} \ ;$$

hence $G_n(j\omega_0) = 1/\sqrt{2}$: thus at the corner frequency the gain is -3 db, independently of the order n, so that corner frequency and conventional bandwith coincide in this case.

Padé delay. A function $P(s)/Q(s)$, where $P(s)$ and $Q(s)$ denote polynomials with generic degrees p and q, respectively, is a *Padé approximant* of function $f(s)$, Maclaurin expandable, if the sequence of powers of s obtained by division of $P(s)$ by $Q(s)$ (by using the standard polynomial division table, but with the polynomials ordered by ascending powers of s) has the first $p+q+1$ terms equal to those of the Maclaurin expansion of $f(s)$.

Let

$$\begin{aligned} P(s) &:= b_p\, s^p + b_{p-1}\, s^{p-1} + \ldots + b_0 \ , \\ Q(s) &:= a_q\, s^q + a_{q-1}\, s^{q-1} + \ldots + a_0 \ ; \end{aligned}$$

the coefficients of the Padé approximants of the exponential function

$$f(s) = e^{-s} = 1 - s + \frac{s^2}{2} - \frac{s^3}{3!} + \ldots \ ,$$

are given by

$$\begin{aligned} b_k &= \frac{(p+q-k)!\, p!}{(p+q)!\, k!\, (p-k)!}\, (-1)^k \quad (k = 0, \ldots, p) \ , \\ a_k &= \frac{(p+q-k)!\, q!}{(p+q)!\, k!\, (q-k)!} \qquad (k = 0, \ldots, q) \ . \end{aligned}$$

The approximants of $e^{-t_0 s}$ are obtained by substituting $t_0 s$ for s in $P(s)$ and $Q(s)$. It is customary to assume $p = q$ since in this case the absolute value of $P(j\omega)/Q(j\omega)$ is identically equal to one like that of the finite delay and stability of the approximant is guaranteed.

Using the Bessel and Butterworth filters as reference models

The transfer functions of the Bessel and Butterworth filters are often used as reference models in the analytical design of regulators. A brief recall of the procedure is in order. Let $G(s) = P(s)/Q(s)$ be the transfer function of the plant, that is assumed to be stable and minimum-phase: it is possible to design a regulator G_r such that the closed-loop system transfer function

$$G_0(s) = \frac{G_r(s)\,G(s)}{1 + G_r(s)\,G(s)}$$

be any given function, such that

1. $G_0(s)$ has a relative degree not less than $G(s)$.

2. If a pole of order h at the origin is requested in the regulator

to obtain asymptotically robust perfect tracking of the corresponding mode (for instance a step, a ramp, etc.), the last h terms in the numerator of $G_0(s)$ must be equal to the corresponding terms in the denominator.

From

$$G_0(s) = \frac{P_0(s)}{Q_0(s)} = \frac{G_r(s)\,G(s)}{1 + G_r(s)\,G(s)}$$

it immediately follows the well-known analytical design formula

$$G_r(s) = \frac{G_0(s)}{1 - G_0(s)}\,\frac{1}{G(s)} = \frac{P_0(s)}{Q_0(s) - P_0(s)}\,\frac{Q(s)}{P(s)}\ . \tag{3.1}$$

Restriction 2 can be introduced when $h = 2$ (type 2 regulator), by adding a suitable zero in the reference model transfer functions. Let

$$G_1(s) = \frac{1}{a_k\,s^k + \ldots + a_1\,s + 1}$$

be the transfer function of a Bessel or Butterworth filter of order k satisfying restriction 1. We define

$$G_0(s) = \frac{(a_1 + \frac{\alpha}{\omega_n})\,s + 1}{(a_k\,s^k + \ldots + a_1\,s + 1)\,(1 + \frac{\alpha}{\omega_n}\,s)}\ , \tag{3.2}$$

thus clearly satitsfying restriction 2. Increasing parameter α produces reduction of maximum overshoot to the detriment of settling time and can be interactively chosen to obtain the best compromise.

3.4.2 Operation

Let us first consider the definition of a transfer function by assigning its zero-pole map with the mouse. The input menu, appearing after the command "deftf,gi", is:

```
1) Define a zero-pole map with mouse
2) Bessel filter
3) Butterworth filter
4) Pade' expansion of finite delay

enter your choice (press return to exit) : 1
```

Upon choice of the first item of the menu, the request:

```
discrete-time ? (1) :
```

appears, that makes definition of a discrete-time transfer function possible, with the graphic environment accordingly modified. Suppose the return key is simply pressed, thus assuming continuous time. A figure appears with the push-button menu shown in Fig. 3.1 on the left and reference axes shown on the right. At the top of the figure the following message is shown:

You can introduce any zero-pole map with mouse. The gain K is set to one.
Note that it is possible to set to one the dc gain Ks instead of K. (a)

The values of **K** (the gain constant referred to the zero-pole form, initially set to one) and **Ks** (the gain constant referred to the time-constant form), and the numbers **nz** and **np** of the selected zeros and poles are shown under the menu. In the command bar at the top of the figure a **Change axes** menu is provided to adjust the automatic scaling, with the selection items: **x left**, **x right**, **y up and down** (enlarge by 1 tick, enlarge by 2, reduce by 1 tick, reduce by 2), **zoom in**, **zoom out**, and a **Grids** menu with **Grid on/off**, **Constant damping loci on/off**. In the case of Fig. 3.1 both grid and constant damping loci were turned on. If a discrete-time transfer function were selected, the constant damping loci would appear accordingly modified. Let us briefly describe the push-button menu operation.

Add zeros. This choice enables selection of a point in the figure with the mouse and causes replacement of the message (a) with:

Select a location for a new real zero or a new complex zero pair.
You can repeat selection with mouse. Press button 2 to accept. (b)

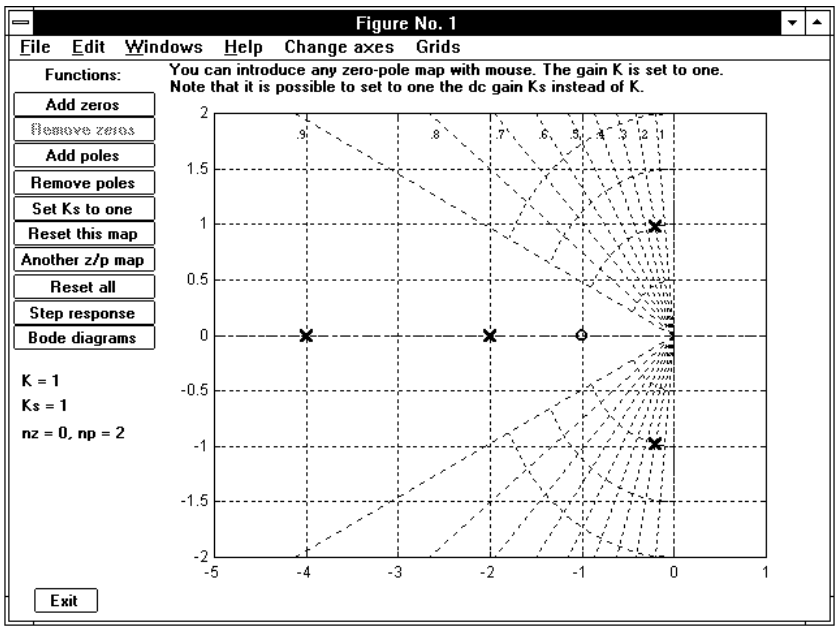

Figure 3.1. The screen layout of *deftf* with option 1.

The color of this message and of the zero-pole map first selected is green. If several zero-pole maps are considered, they will appear in different colors. Selection with the mouse of a point above the real axis defines a complex zero pair, while selection (slightly) below defines a real one. In both cases the selected zero locations are shown in green with standard symbols in the figure, while their numerical value(s) are displayed, also in green, at the left of the figure. Selection is performed and possibly modified by clicking button 1 and eventually confirmed by button 2. Definition of more zeros is obtained by repeating the whole procedure.

Remove zeros. This choice enables removal of a real zero or a complex zero pair by selection with the mouse and clicking button 1.

Add poles. Same as above for zeros, with message and symbols accordingly modified.

Remove poles. Same as above for zeros.

Set Ks to one. This option makes comparison of several time or frequency responses easier, since it unifies the steady-state values. Of course, the value of K is accordingly modified.

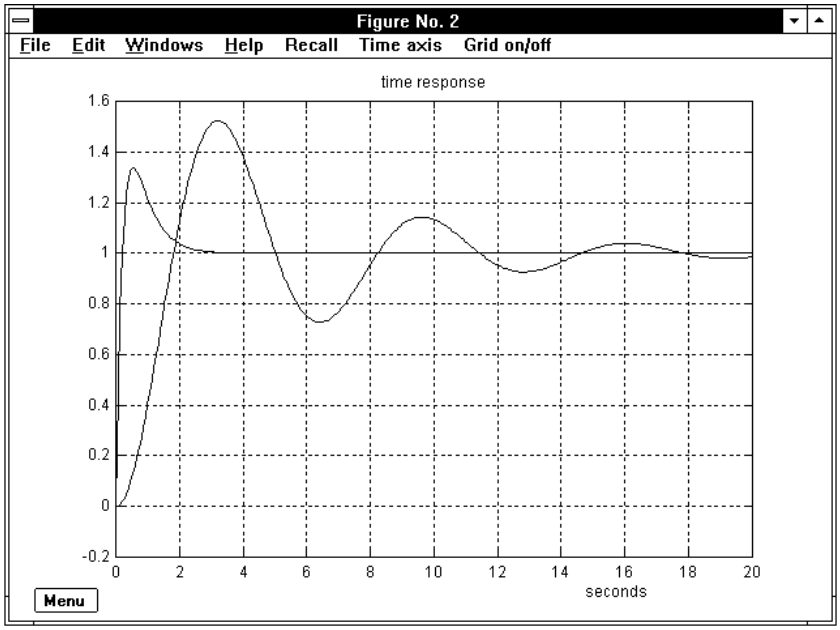

Figure 3.2. The **Step response** option of *deftf*.

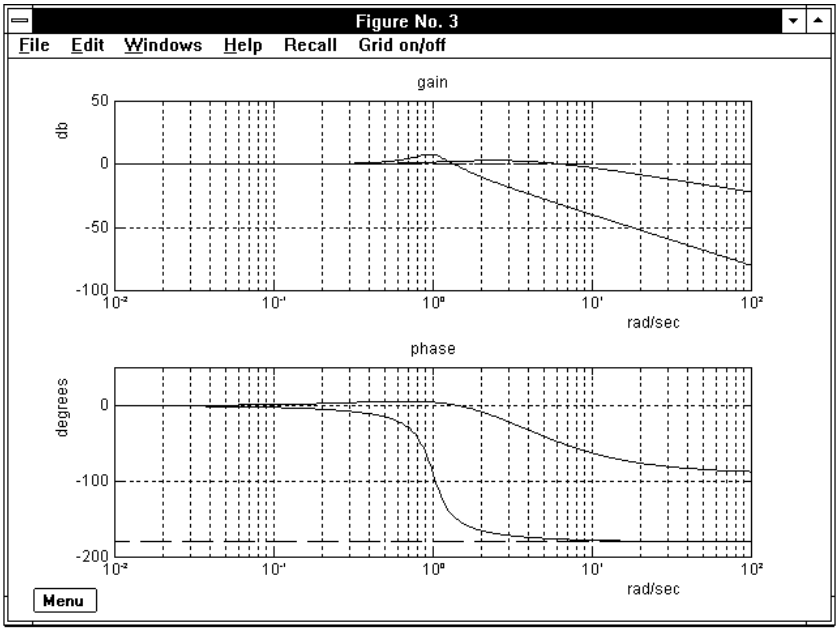

Figure 3.3. The **Bode diagrams** option of *deftf*.

Reset this map. This option removes all the zeros and poles of the current map (i.e., denoted with the same color).

Another z/p map. This choice enables introduction of another map, whose zero and pole symbols are plotted in a different color. The previously defined maps are preserved. This makes it possible to easily point out the connections between zero and pole locations and time or frequency responses.

Reset all. Self-explaining.

Step response. The step responses of the zero-pole maps are each plotted in the same color as the corresponding map, with automatic axes selection. An example is shown in Fig. 3.2 (consistent with Fig. 3.1, that reports two maps, one with a pair of real poles and a real zero, the other with a pair of complex poles, shown in different colors – green and red). In the control bar at the top of the figure a **Time axis** pop-up menu is provided to adjust the automatic one, with the selection items **zoom x 2**, **zoom x 5**, **zoom x 10**, **recover the initial axis**, and a **Grid on/off** menu. The pushbutton **Menu** at the bottom-left corner of the figure exit this window and recovers that shown in Fig. 3.1.

Bode diagrams. The Bode diagrams of the zero-pole maps are each plotted in the same color as the corresponding map, with automatic axes selection. An example is shown in Fig. 3.3, also consistent with Fig. 3.1. In the control bar at the top of the figure a **Grid on/off** pushbutton is added. In this case also an **Menu** pushbutton is provided to exit the Bode diagrams and recover the zero-pole maps with the menu.

Exit. Up to seven zero-pole maps may be added in different colors and their dynamics compared by plotting the corresponding step responses and Bode diagrams. The program is quitted with this pushbutton, that first produces the message:

PRESS RETURN TO EXIT

displayed in orange at the bottom-left corner of the figure. Pressing the return key produces exit from *deftf* and going back to the Command Window. Note that option 1 of the input menu (Define a pole-zero map with mouse) has been completely managed with the mouse, including response checks, grids and changes of scales. When the return key is pressed, we go back to keyboard management. If a single zero-pole map is defined, it is saved as transfer function *gi*, and this is shown in the Command Window.

If, on the other hand, several zero-pole maps are defined and shown in the figure in different colors, the request:

```
select function by entering color :
```

is put forward in the Command Window on exiting the program. When a letter corresponding to an existing color is entered, the corresponding zero-pole map is saved as transfer function gi, that is displayed in the Command Window.

We now consider the second and third item of the input menu (Bessel and Butterworth filter), that are managed likewise. In particular, let us refer to the case of a Bessel filter:

```
1) Define a zero-pole map with mouse
2) Bessel filter
3) Butterworth filter
4) Pade' expansion of finite delay

enter your choice (press return to exit) : 2

order of denominator : 4

corner frequency : 2
```

These choices produce creation of a medium-size figure with the step response of the filter shown in green and the message:

```
do you want a reference model for a type 2 system ? (1) :
```

Upon simple pressing of the return key the transfer function:

```
     16
gi = ------------------------------------------
     (s^4 + 6.248*s^3 + 17.57*s^2 + 25.61*s + 16)
```

is displayed and saved. On the other hand, replying 1 to the previous request produces:

```
the order of both numerator and denominator is increased by 1
parameter alpha trades overshoot for settling time
enter the value of alpha (from 1 to 100) :
```

and, when an admissible α is entered, the step response of the transfer function with a zero added according to (3.2) is shown in the same figure in a different color.

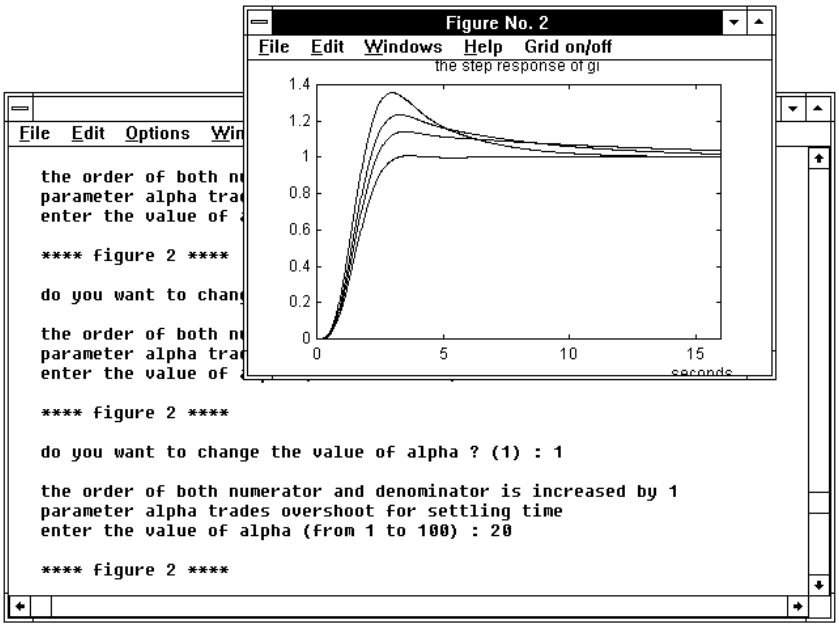

Figure 3.4. The interactive choice of α.

Upon pressing the return key the request:

```
do you want to change the value of alpha ? (1) :
```

appears, to make a different choice of α possible. Choice of α can be repeated until the step response shown is satisfactory. Fig. 3.4 reproduces the screen layout when three choices (5,10,20) have been subsequently introduced for the value of α, implying different overshoots and settling times, visible in the figure. In the case in hand, by simply pressing the return key on the above request with $\alpha = 20$ as the last introduced value, we obtain the final result:

```
     18.56 (s + 0.0862)
gi = ------------------------------------------------------
     (s^5 + 6.348*s^4 + 18.19*s^3 + 27.37*s^2 + 18.56*s + 1.6)
```

It is easy to check that $gi(s)/(1-gi(s))$, the first factor on the right of (3.1), has the double pole at the origin that characterizes a type 2 regulator.

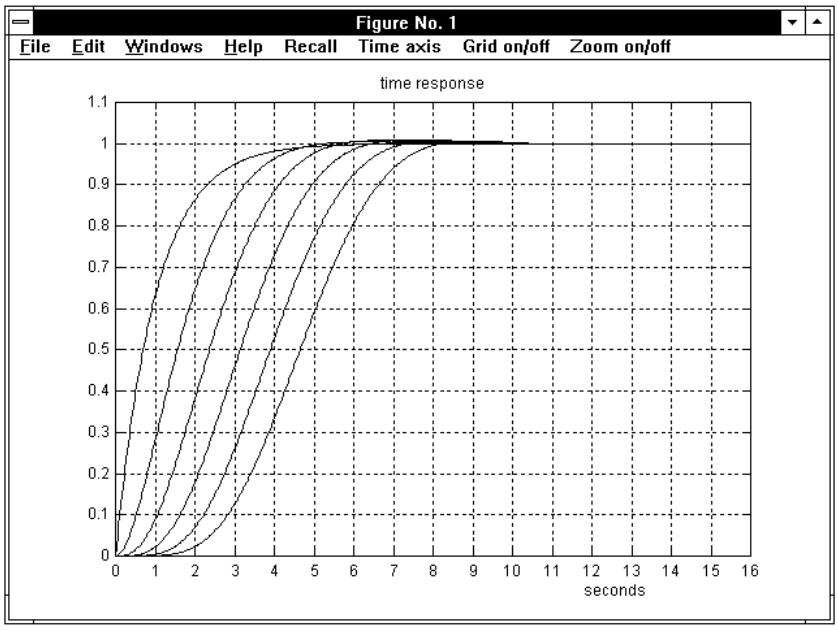

Figure 3.5. The step responses of some Bessel filters.

Figure 3.6. The step responses of some Butterworth filters.

Interactive requests are slightly different for Padé delay: in this case the order of the numerator is also requested, but, if not specified, it is set equal to that of denominator by default. As an example, let us consider:

```
1) Define a zero-pole map with mouse
2) Bessel filter
3) Butterworth filter
4) Pade' expansion of finite delay

enter your choice (press return to exit) : 4

order of denominator : 3

order of numerator (default: same as denominator) :

delay (sec) : 2
```

After this, the step response of the delay is shown in the medium-size figure, and the corresponding transfer function saved and displayed as:

```
     -1 (s^3 - 6*s^2 + 15*s - 15)
gi = ----------------------------
     (s^3 + 6*s^2 + 15*s + 15)
```

3.4.3 Examples

Fig. 3.5 shows the step responses of the Bessel filters with orders from 1 to 6 and unit corner frequency, plotted with *tresp*.

The step responses of the Butterworth filters of the same orders and unit corner frequency are shown in Fig. 3.6; less damping with respect to Bessel filters clearly appears.

Fig. 3.7 refers to the step response of the Padé approximants of the unit delay, also with orders from 1 to 6 and numerators having the same orders as denominators, while Fig. 3.8 reports the step responses of the same Padé approximants in cascade with the transfer function $g(s)=1/(1+0.5s)$; the filtering effect of $g(s)$ improves the overall system behavior within the delay time.

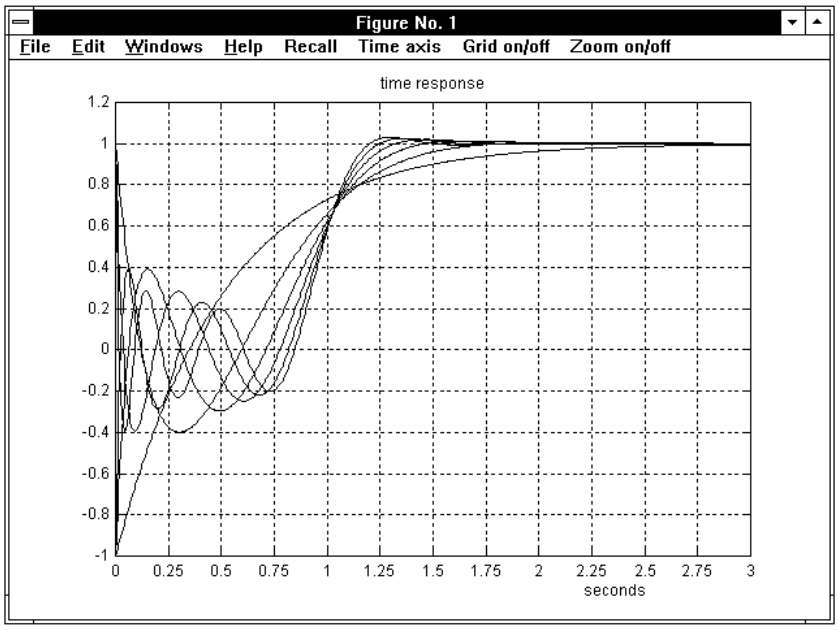

Figure 3.7. The step responses of some Padé approximants.

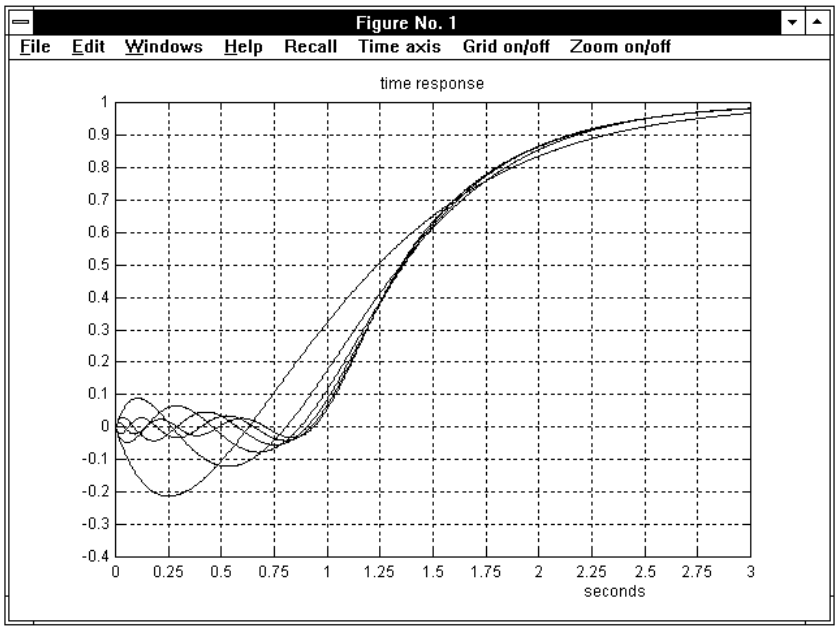

Figure 3.8. The filtered step responses of some Padé approximants.

3.5 Descrf

The command

`> descrf[,gi]` *(enter)*

if called without any argument, provides graphical and computational support to derive the describing functions $F(X)$ of some nonlinear elements, while, if called with a transfer function name gi, it determines the intersections of the polar plot of $-1/F(X)$ with that of $gi(j\omega)$ in order to detect amplitudes and angular frequencies of possible self-oscillatory behaviors of the feedback connection of the nonlinear element and the linear system.

3.5.1 Recall

Let us consider the feedback connection shown in Fig. 3.9, where a nonlinear algebraic element is connected with a linear system with transfer function $G(s)$. The input-output behavior of the nonlinear element is described by a given function $y = f(x)$, that in general is assumed to be one-valued and odd (the relay with hysteresis and backlash, described below, are exceptions). The describing function of

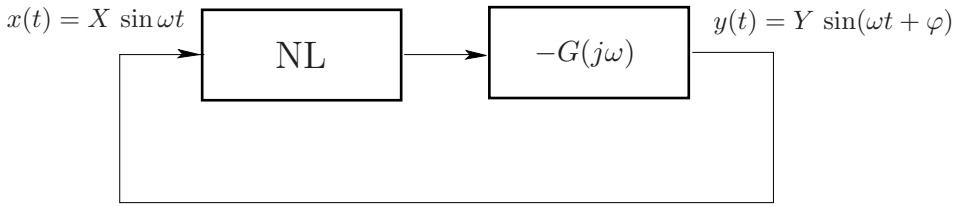

Figure 3.9. The feedback connection considered.

the nonlinear element is defined as

$$F(X) := \frac{1}{X}\, Y(X)\, e^{j\varphi(X)} \ ,$$

where Y and φ denote the amplitude and the phase shift of the first harmonic in the Fourier series expansion of the output, that are functions of the amplitude of the sine wave present at the input.

Program *descrf* considers the following types of nonlinear elements:

1. saturation
2. dead zone
3. saturation with dead zone
4. any linearly interpolated nonlinearity
5. ideal relay
6. relay with dead zone
7. relay with hysteresis
8. backlash

Selection of the type and parameters are introduced interactively. It is well-known that the describing functions of the elements from 1 to 6 can be expressed in terms of those of *unit saturation* and *relay with dead zone*, whose input-to-output functions are represented in Fig. 3.10,a and Fig. 3.10,b respectively. These are

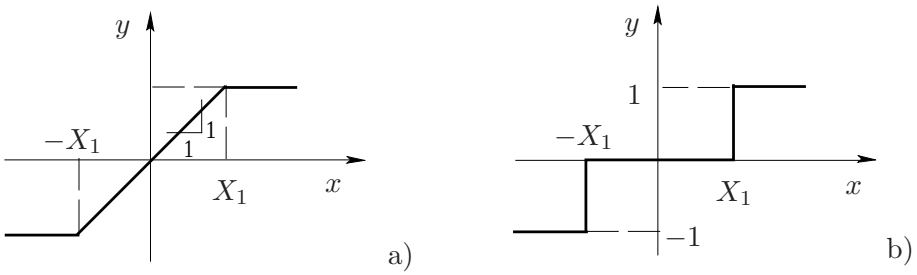

Figure 3.10. The saturation and the relay with dead zone.

$$F(X) = \begin{cases} 1 & \text{for } X \le X_1 \\ \Phi\left(\frac{X}{X_1}\right) & \text{for } X \ge X_1 \end{cases},$$

with

$$\Phi\left(\frac{X}{X_1}\right) := \frac{2}{\pi}\left(\arcsin\frac{X_1}{X} + \frac{X_1}{X}\sqrt{1-\left(\frac{X_1}{X}\right)^2}\right), \tag{3.3}$$

and

$$F(X) = \begin{cases} 0 & \text{for } X \le X_1 \\ \Psi(X, X_1) & \text{for } X \ge X_1 \end{cases} ,$$

with

$$\Psi(X, X_1) := \frac{4}{\pi X} \sqrt{1 - \Big(\frac{X_1}{X}\Big)^2} . \tag{3.4}$$

Unlike the previous ones, that are all real, the describing functions of elements 7 and 8, i.e., the relay with hysteresis and the backlash, are complex. The corresponding input-output functions are shown in Fig. 3.11,a and Fig. 3.11,b, respectively.

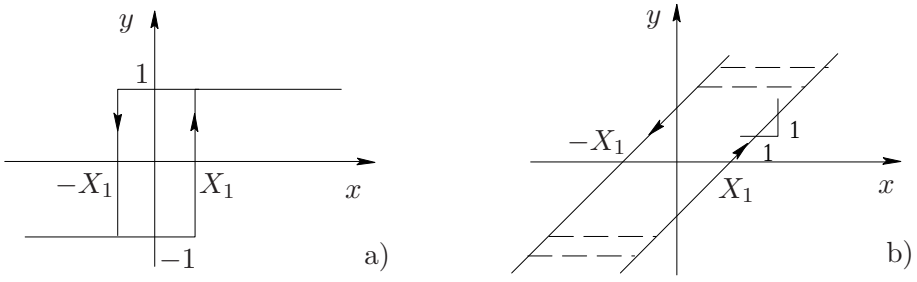

Figure 3.11. The relay with hysteresis and the backlash.

The describing function of the relay with hysteresis is

$$\begin{aligned} F(X) &= \frac{4}{\pi\, X}\Big(\sqrt{1 - \Big(\frac{X_1}{X}\Big)^2} - j\,\frac{X_1}{X}\Big) \\ &= \Psi(X, X_1) + j\frac{4\,X_1}{\pi\, X^2} \qquad\qquad \text{for } X \ge X_1 , \end{aligned}$$

not defined for $X < X_1$, while that of the backlash is

$$F(X) = \begin{cases} 0 & \text{for } X \le X_1 \\ \frac{1}{2}\left(1 + \Phi\Big(\frac{X}{X - 2\,X_1}\Big)\right) + j\,\frac{4\,X_1\,(X_1 - X)}{\pi\, X^2} & \text{for } X \ge X_1 \end{cases} .$$

Note that they are expressed in terms of the basic functions Φ and Ψ defined in (3.3) and (3.4), respectively. Program *descrf* provides the describing function both as a graph and a finite-term mathematical expression. In the second case, the expression of functions Φ and/or Ψ are recalled in display.

Describing functions of nonlinear elements are used to detect possible limit cycles (self-sustaining oscillations) of the nonlinear system shown in Fig. 3.9. To this end, it is necessary to solve the equation

$$F(X)\,G(j\omega) = -1\ ,$$

with respect to the unknown quantities X and ω. Solutions are usually derived graphically, by intersecting the polar plot of $-1/F(X)$ with the Nyquist diagram of $G(j\omega)$.

3.5.2 Operation and Examples

Let us first consider the call without any argument. Entering "descrf" causes the following input menu to be displayed:

```
Choose the nonlinear element :

 1 - saturation
 2 - dead zone
 3 - saturation with dead zone
 4 - any linearly interpolated nonlinearity
 5 - ideal relay
 6 - relay with dead zone
 7 - relay with hysteresis
 8 - backlash

enter your choice (press return to exit) :
```

If, for instance, choice 4 is entered, the following interactive requests appear (shown with possible answers):

```
enter the x break points [x1 x2 x3 ...] : [1 2 2 3 4]

enter the y break points [y1 y2 y3 ...] : [1 1 2 3 3]

**** press return to continue
```

and a medium-size figure with the input-output relation of the nonlinear element appears in the screen, as shown in Fig. 3.12.

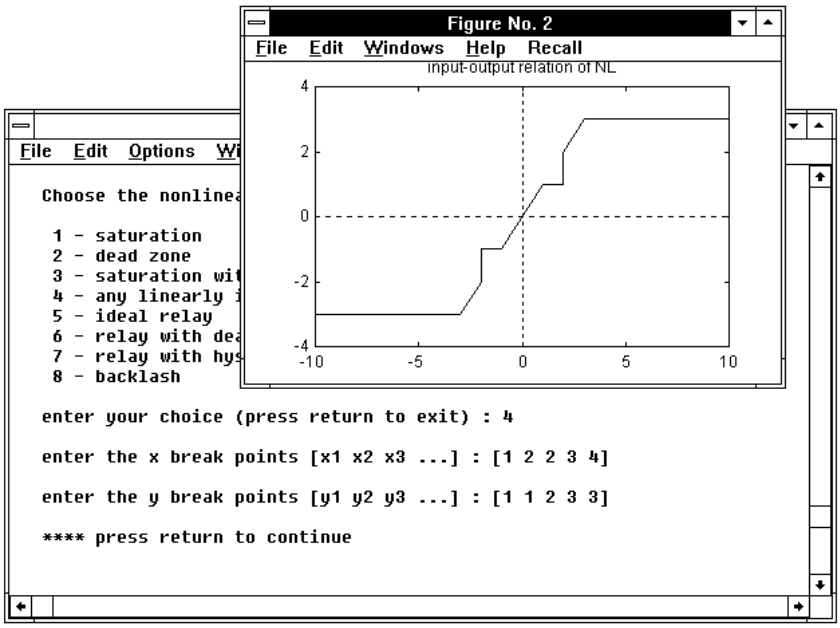

Figure 3.12. Defining a general nonlinear element by points.

On pressing return, the figure with the describing function of the selected nonlinear element is shown as in Fig. 3.13, and, when the return key is pressed again, the main menu appears as:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - compute the maximum of F(X)
4 - compute inverse values of F(X)
5 - recover the figure
6 - display the expression of F(X)

enter your choice (press return to exit) :
```

Options 1, 2 and 5 are the standard ones of all graphic programs in TFI (see, for instance, *fresp*). The subsequent ones are briefly described in the following.

Figure 3.13. The desctibing function of the nonlinear element in Fig. 3.12.

Compute the maximum of $F(X)$. This option causes the figure with the describing function to be shown again, with the maximum of $F(X)$ and the corresponding value of X (if unique) pointed out with dotted red lines. On going back to the Command Window with the return key, the corresponding values are displayed in the form:

```
[F(x)]_max    : 1

-1/[F(X)]_max : -1
```

Compute inverse values of $F(X)$. This option produces the request:

```
enter the value of F(X) :
```

Suppose the value .85 is entered. The figure with the describing function is shown again with the construction used to derive the corresponding inverse value(s) shown in cyan dotted lines. On going back to the Command Window, by pressing return the following appears:

```
inverse value(s) of F(X) = 0.85 :

X1 : 1.346
X2 : 2.252
X3 : 3.912

enter any letter for another point, press return for the menu :
```

Display the expression of $F(X)$. This produces:

```
DESCRIBING FUNCTION :

F(X) = +1                                                  for 0<=X<1
       1*Phi(X/1)                                          for 1<=X<2
       +1*Phi(X/1)-1*Phi(X/2)
       +1+1*Psi(X,2)                                       for 2<=X<3
       +1*Phi(X/1)-1*Phi(X/2)
       +1*Phi(X/3)+1*Psi(X,2)                              for 3<=X

with   Phi(X/X1):=2/pi*[asin(X1/X)+X1/X*sqrt(1-(X1/X)^2)];
       Psi(X,0):=4/(pi*X); Psi(X,X1):=4/(pi*X)*sqrt(1-(X1/X)^2).
```

Note that the describing function of the nonlinear element considered is expressed in terms of functions Φ and Ψ defined by (3.3) and (3.4), respectively. This happens as a rule when this is a linearly interpolated nonlinearity (option 4).

As previously recalled, application *descrf*, if called with an argument representing a transfer function name, determines the amplitudes X_i and angular frequencies ω_i of all the possible limit cycles of the feedback system represented in Fig. 3.9.

Let us suppose the transfer function

$$gi(s) = \frac{140}{s\,(s+1)\,(s+10)}$$

is present in the TFI work directory. The command "descrf,gi" causes the interactive input menu to be displayed again in the Command Window as follows:

```
Choose the nonlinear element :

1 - saturation
2 - dead zone
3 - saturation with dead zone
4 - any linearly interpolated nonlinearity
5 - ideal relay
6 - relay with dead zone
7 - relay with hysteresis
8 - backlash

enter your choice (press return to exit) :
```

Suppose that choice 4 is entered also in this case with the same parameters as before, i.e., with the furter interactive communication:

```
enter the x break points [x1 x2 x3 ...] : [1 2 2 3 4]

enter the y break points [y1 y2 y3 ...] : [1 1 2 3 3]

**** press return to continue
```

A medium-size figure with the input-output relation of the nonlinear element appears as in the call-without-argument case, as shown in Fig. 3.12.

On pressing return, a figure with the Nyquist diagram of $gi(j\omega)$ and the polar plot of $-1/F(X)$ shown together is displayed. The latter is drawn in blue if directed towards the origin for X increasing and in red in the opposite case, to make superimposed branches distinguishable. In the example considered, this appears as shown in Fig. 3.14, where the two plots intersect at three points. The intersections correspond to limit cycles. When the return key is pressed again, the main menu appears as:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - compute the possible limit cycles
4 - enter a finite delay
5 - recover the figure
6 - plot F(X) and display information about it

enter your choice (press return to exit) :
```

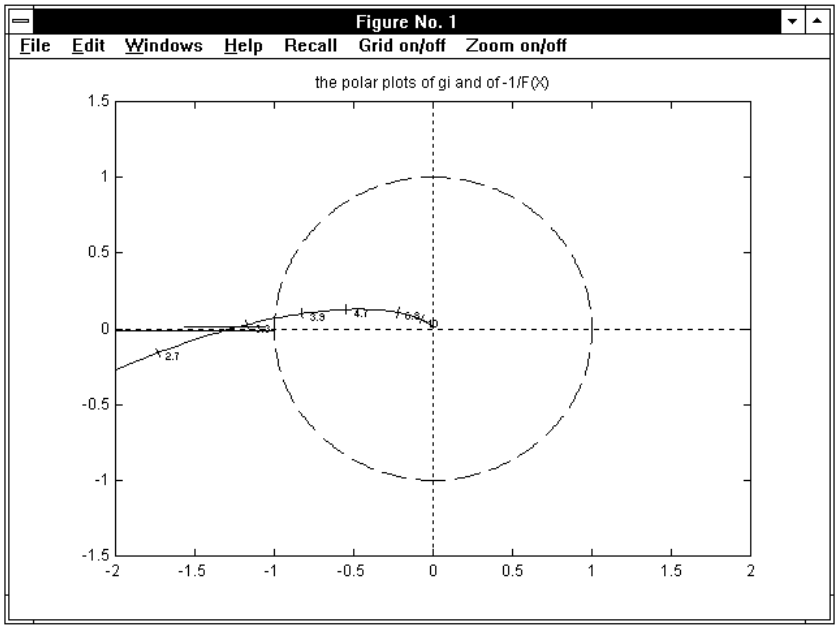

Figure 3.14. Computation of limit cycles.

Options 1, 2, 4 and 5 are the standard ones of all graphic programs in TFI, while option 3 produces:

```
INTERSECTIONS OF gi(j*omega) WITH THE NEGATIVE REAL AXIS:

intersection at Re(gi) = -1.273 (frequency: 3.162 rad/sec)

INTERSECTIONS OF gi(j*omega) WITH -1/F(X) (LIMIT CYCLES)
(only those in the plot - change the axes if necessary):

intersection for X = 1.489 (frequency: 3.162 rad/sec)
intersection for X = 2.109 (frequency: 3.162 rad/sec)
intersection for X = 4.37 (frequency: 3.162 rad/sec)
```

Option 6 simply causes switching to the previously described case when *descrf* is called without any argument, i.e. drawing of the plot of the describing function of the selected nonlinear element and access to the corresponding menu.

3.6 Factf

The command

`> factf,gi,gj` *(enter)*

displays and saves in the current work directory as $gj(s)$ or $gj(z)$ the completely factorized form (with a first degree factor for every real root and a second degree factor for every complex conjugate pair) of a given transfer function $gi(s)$ or $gi(z)$, in polynomial form or not completely factorized.

3.6.1 Recall

The factorized form of a monic polynomial $P(s)$ is expressed by

$$P(s) = \prod_{i=1}^{n} (s - r_i) \ ,$$

where $r_1, \ldots, r_n$ denote the roots. The terms corresponding to a conjugate pair of complex roots can be substituted with a single second-order term with real coefficients. Hence the factorized form is obtained through computation of the roots of the polynomials appearing at the numerator and denominator of $gi(s)$ or $gi(z)$. However, multiple roots may not be recognized, due to scarce robustness of computational routines in this case. For instance, factorization of

$$gi(s) = \frac{1}{s^6 + 12\,s^5 + 60\,s^4 + 160\,s^3 + 240\,s^2 + 192\,s + 64}$$

with the denominator roots computed by means of Matlab's *roots* routine, would provide

$$gj(s) = \frac{1}{(s + 1.993)\,(s + 2.007)\,(s^2 + 3.993\,s + 3.987)\,(s^2 + 4.007\,s + 4.013)} \ ,$$

instead of

$$gj(s) = \frac{1}{(s + 2)^6} \ ,$$

from which $gi(s)$ was obtained with "defactf,gj,gi". To avoid this, *factf* is provided with the interactive possibility of recognizing multiple roots within a specified tolerance and substituting the error-affected values with their mean: in the case in hand, a tolerance of .01 is sufficient to obtain the correct result.

3.6.2 Operation

Let us suppose that transfer function gi specified in command "factf,gi,gj" is the above one. First $gi(s)$ is shown and the following request appears:

```
You can factorize:

1) numerator
2) denominator
3) numerator and denominator

enter your choice (return to skip) : 2
```

After selection, the computed poles are displayed and a tolerance for multiplicity is requested. The poles are displayed again with the multiplicity corresponding to the specified tolerance, and, if the result is accepted by the user, the final transfer function $gj(s)$ is displayed and saved. If not, a new tolerance can be specified. In the Command Window this appears as:

```
>>> Poles <<<

  1  |  -2.0067e+000    +3.8588e-003 * j  |   1
  2  |  -2.0067e+000    -3.8588e-003 * j  |   1
  3  |  -2.0000e+000    +7.6954e-003 * j  |   1
  4  |  -2.0000e+000    -7.6954e-003 * j  |   1
  5  |  -1.9933e+000    +3.8366e-003 * j  |   1
  6  |  -1.9933e+000    -3.8366e-003 * j  |   1

specify the tolerance (default is 10^(-4)) : .01

>>> Poles with multiplicities detected <<<

  1  |  -2.0000e+000                      |   6

do you want to change the tolerance (1/0) ?:  0

    1
gi = ------------------------------------------------------
    (s^6 + 12*s^5 + 60*s^4 + 160*s^3 + 240*s^2 + 192*s + 64)

    1
gj = ---------
    (s + 2)^6
```

Options 1 and 3 of the input menu produce similar interactive sessions.

3.7 Fresp

The command

```
> fresp,gi  (enter)
```

plots the frequency response of the continuous-time or discrete-time system with transfer function $gi(s)$ or $gi(z)$. By using an interactive facility present in the program it is possible to obtain in the same figure frequency response diagrams corresponding to other transfer functions, plotted in different colors.

3.7.1 Recall

In the continuous-time case the frequency response function corresponding to $gi(s)$ is

$$F(\omega) := gi(j\omega) \ , \quad 0 \leq \omega < \infty \ ,$$

while in the discrete-time case, referring to $gi(z)$, we have

$$F(\omega) := gi(e^{j\omega T}) \ , \quad 0 \leq \omega \leq \frac{\pi}{T} \ .$$

where T denotes the sampling period.

In both cases function $F(\omega)$ is a complex-valued function of the angular frequency ω. In standard automatic control system design the following three types of frequency response representation are used.

Bode diagrams, consisting of two plots, i.e., the *Bode diagram of gain*, representing the absolute value of $F(\omega)$ versus ω, and the *Bode diagram of phase*, representing the argument of $F(\omega)$ versus ω. The standard Bode diagrams are semilogarithmic, with logarithmic scale for abscissa ω and using db (decibels), i.e. $20 \log_{10}|F(\omega)|$, for gain and degrees, i.e. $180/\pi \arg F(\omega)$, for phase.

Nichols diagram, consisting of a single representation of the absolute value versus the phase angle of $F(\omega)$, with the absolute value measured in db and the angle in degrees; every point of the graph corresponds to a different value of ω.

Nyquist diagram, consisting of a single representation of the imaginary part versus the real part of $F(\omega)$; every point of the graph corresponds to a different value of ω.

In all cases the plots are referred to suitable frequency intervals, depending on the system dynamics.

The Bode diagrams of gain and phase of continuous-time systems both admit *asymptotic approximation*, i.e., can be approximated with piecewise linear functions resulting from the sum of the asymptotic approximations of the elementary diagrams of single zeros and poles. The asymptotic approximations can easily be obtained from the transfer function in factorized form and clearly point out the locations of zeros and poles, thus characterizing the system dynamics. The elementary asymptotic diagram of gain corresponding to a single real or complex zero z_i or pole p_i consists of two half-lines: the first one horizontal at 0 db and the second one with $+20$ db/decade slope in the case of a zero and -20 db/decade slope in the case of a pole, joined at the "break point" $\omega = |z_i|$ or $\omega = |p_i|$. The elementary asymptotic diagram of phase corresponding to z_i or p_i consists of two horizontal half-lines at 0 and 90 degrees (if z_i has negative real part or p_i positive real part) or 0 and -90 degrees (if z_i has positive real part or p_i negative real part) connected by a sloping segment with mid-point at $\omega = |z_i|$ or $\omega = |p_i|$ and break points at $\omega = |z_i| / e^{\pi/2}$, $\omega = |z_i| e^{\pi/2}$ or $\omega = |p_i| / e^{\pi/2}$, $\omega = |p_i| e^{\pi/2}$. For a zero or a pole at the origin, the elementary asymptotic diagram of gain is a straight line with $+20$ or -20 db/decade slope passing through the point (1,0) of the semilogarithmic scale and that of phase is the horizontal line corresponding to $+90$ or -90 degrees. The overall asymptotic Bode diagram of gain is obtained by summing the elementary asymptotic diagrams of all zeros and poles and is vertically shifted by the gain constant of the transfer function, i.e., the value of the transfer function at $s=0$ computed neglecting all null zeros or poles (in db).

In the case of Nichols and Nyquist plots, every point of the plane represents a particular complex number c: it is possible to complete these plots with the *constant M loci*, that are lines along which $M = |c/(1+c)|$ is constant, and *constant N loci*, lines along which $N = \tan\beta$, with $\beta = \arg(c/(1+c))$, is constant. They are useful to derive some closed-loop frequency response parameters (typically resonance peak, resonance frequency and bandwidth) from the open-loop frequency response diagram and are labelled, respectively, with the values of M in db and of the main value of $\arctan N$ in degrees, i.e., the value of $\arctan N$ in the left-open interval $(-180, 180]$, that gives the argument of $c/(1+c)$ within $\pm 180\,\nu$ degrees, when ν is an arbitrary integer.

3.7.2 Operation

The input menu that appears in the Command Window after entering "fresp,gi" is:

```
1 - Bode diagram of gain
2 - Bode diagram of phase
3 - Bode diagrams of gain and phase - a single figure
4 - Bode diagrams of gain and phase - two separate figures
5 - Nichols diagram
6 - Nyquist diagram

enter your choice (0 to exit) :
```

After the choice has been performed, you must select the color of the plot according to the request:

```
choose color of plot: k=black, g=green,
b=blue, r=red, y=yellow, m=magenta, c=cyan, default is green :
```

When an admissible color is entered, the current graphic window is cleared, shown at full-screen size, and, after some delay for computations, the selected type of frequency response diagram is shown. Typical Bode diagrams of gain and phase in a single figure (option 3) are shown in Fig. 3.15, while Bode diagrams in two separate figures (option 4) are reported in Fig. 3.16, a Nichols diagram (option 5) is shown in Fig. 3.17, Nyquist diagrams (option 6) in Figs. 3.18 and 3.19. However, these figures refer to cases where some further elements were added to the first plot with the main menu.

To go back to the Command Window and to see the main menu, press the return key. The main menu partly depends on the first choice.

With choices from 1 to 4 (Bode diagrams) we have:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - information on frequency response
4 - plot another function in different color
5 - recover the figure
6 - information on plot(s) with mouse
7 - plot the asymptotic approximation (for continuous-time only)
8 - introduce a finite delay

enter your choice (press return to exit) :
```

With choice 5 (Nichols diagram) we obtain:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - information on frequency response
4 - plot another function in different color
5 - recover the figure
6 - information on plot(s) with mouse
7 - plot with constant M and N loci
8 - graduate versus omega
9 - introduce a finite delay

enter your choice (press return to exit) :
```

while choice 6 (Nyquist diagram) has the same menu, but with option 7 replaced by:

```
7 - zoom by steps
```

We now provide a brief description of the items of the above menus.

Change the reference axes and plot again. This option allows selection of ranges for both axes and of the number of divisions when the scale is linear. The default value for range is the current one while the number of divisions, if not specified, is automatically determined. A typical interactive selection of new reference axes, referring to Bode diagrams of gain and phase in two separate figures (option 4 of the input menu) is the following:

```
the frequency range is [0.1 100];
press return to maintain or type new values: [om1 om2] = [.01 1e3]

the magnitude range is [-100 50];
press return to maintain or type new values: [ym yM] = [-120 80]
number of divisions for magnitude axis: 3;
press return to plot again with automatic scaling
or type a new value: ndy = 5

the phase range is [-300 0];
press return to maintain or type new values: [ym yM] = [-270 90]
number of divisions for phase axis: 3;
press return to plot again with automatic scaling
or type a new value: ndy = 4
```

When new reference axes have been defined, the figure is drawn again. The above change of axes refers to Fig. 3.16.

Grid on. Draws the figure again with a grid referred to the axes divisions. In subsequent menus option 2 appears as:

```
2 - grid off
```

and enables the figure to be recovered without the grid. The two types of option 2 toggle. The **Grid on/off** command visible in the control bar at the top of the figure provides the grid by using the mouse instead of the keyboard.

Information on frequency response. This option provides information on the most important parameters of frequency responses that have been plotted; information appears in the Command Window.

Typical information, pertinent to frequency response of the transfer function (3.5) defined in the next *Examples* section, appears as:

```
OPEN-LOOP FREQUENCY RESPONSE :
gain margin: 2.75 (8.787 db) at frequency: 3.162 rad/sec
phase margin: 17.7 degrees at frequency: 1.861 rad/sec
abscissa of the vertical asymptote to the polar plot: -4.4

CLOSED-LOOP FREQUENCY RESPONSE :
absolute gain peak: 3.297 (10.36 db) at frequency: 1.914 rad/sec
dc gain: 1 (0 db); relative gain peak: 3.297 (10.36 db)
bandwidth (-3db): 2.961 rad/sec

**** press any key to continue
```

The above information refers to a type one system, i.e., a system with a single pole at the origin. If the system is not of type one, information on vertical asymptote is not displayed. If several plots have been drawn in different colors in the same figure (by using option 4 of the same menu in previous runs), before the information is displayed we have the following request:

```
select function by entering color :
```

The names of the transfer functions and the corresponding colors are accessible through the pop-up menu **Recall** present in the control bar at the top of the figure. By selection with the mouse, a particular transfer function can be temporarily displayed over the figure in the same color (click again on the figure to cancel it).

Plot another function in different color. This option makes possible to plot in the same figure several graphs in different colors, referring to different transfer functions. A typical use is comparing different solutions to synthesis problems in the frequency domain. The corresponding interactive request appears as follows:

```
enter transfer function : g2

choose color of plot: k=black, g=green,
b=blue, r=red, y=yellow, m=magenta, c=cyan, default is green :
```

Choice of a color already present in the figure is rejected. The added plot is referred to the previous axes, so that it is often necessary to use option 1 for the best axes selection. Mixing frequency response plots of continuous-time and discrete-time systems is allowed.

Recover the figure. This option enables the current figure to be recovered from the Command Window by using the keyboard instead of the mouse.

Information on plot(s) with mouse. This option first produces the message:

```
**** press return to enable selection
```

When the return key is pressed, the current figure is accessed from the Command Window. In cases of options from 1 to 4 (Bode diagrams), the message:

Select a point with mouse (use button 1, button 2 to exit) (a)

is displayed in the top left corner of the figure. The aim of selection is to define a value of the angular frequency. In fact, when a point is selected, a vertical bar passing through it appears and the values of both gain and phase of every frequency response plot in the figure corresponding to the intersections with the vertical bar are displayed, each in the color of the plot, in the top right corner, as shown in Fig. 23. The value of the angular frequency is also displayed, in orange. Selection with button 1 can be repeated, thus changing abscissa of the vertical bar and, consequently, the displayed values. When button 2 is pressed, the message:

PRESS RETURN TO RECOVER THE MAIN MENU

is displayed in the bottom left corner. Pressing the return key causes return to the Command Window with the main menu.

In cases of option 5 (Nichols diagram) and 6 (Nyquist diagram) operation is different. Referring to the Nichols diagram, we first obtain the layout shown in Fig. 3.21, with the message:

Choose a color or MENU to exit

in the top left corner, and a push-button menu that allows selection of a particular plot by color or exiting the information-with-mouse session (thus recovering the Command Window with the main menu). In the particular case of Fig. 3.21 there are two frequency response plots, one green and the other red. When a color, for instance red, is selected with the mouse on the push-button menu, message (a) is replaced by:

Select a point with mouse (button 1) on the red plot (button 2 to change plot or exit) (b)

and, when selection is performed, a small arrow is drawn on the corresponding plot in the same color in the direction of increasing frequency to point out location of the point selected, as shown in Fig. 25, and information on frequency and gain/phase, both open-loop and closed-loop, appears in the top right corner of the figure, while the message:

Click again button 1 to delete the displayed data (c)

is shown in the top left side. When button 1 is clicked, message (b) is displayed: thus, selection of a point on the red plot with button 1 or return to the push-button menu with button 2 are enabled again.

Plot the asymptotic approximation (for continuous-time only). In the case of Bode diagrams (choices 1–4 of the input menu) with this option it is possible to add to any diagram in the figure its asymptotic approximation. If several plots have been drawn in different colors, it is necessary to specify the desired one by answering the request:

```
select function by entering color :
```

The asymptotic approximation is drawn in the same color as the corresponding diagram. An example, referring to the asymptotic approximation of the Bode diagrams of a lead compensator, is shown in Fig. 3.15.

Plot with constant M and N loci. This option is available in the main menu only for the Nichols diagram and produces a plot with particular axes ranges (i.e., $[-360, 0]$ and $[-40, 40]$) and with constant M and N loci, as shown in Fig. 3.17. When the Command Window is recovered by pressing the return key, option 7 in the menu is replaced by:

```
7 - delete constant M and N loci
```

Zoom by steps. This option is available only for the Nyquist diagram and produces subsequent plots (typically three) with progressive enlargement, due to suitable reductions of the axes ranges. It is particularly useful when the plotted transfer function has a pole at the origin, so that the diagram asymptotically comes from a point at infinity and automatic selection of axes includes a very large domain of the complex plane to show the overall behavior of the plot. The last plot of the sequence has axes ranges $[-2, 2]$ and $[-1.5, 1.5]$ in order to represent in detail the system behavior in the neighborhood of the critical point -1.

When the last plot of the sequence has been obtained, in the next menu option 7 appears as:

```
7 - plot with constant M and N loci
```

providing these loci, that appear as circles in the Nyquist diagram (see Fig. 3.19), and toggles with:

```
7 - delete constant M and N loci
```

like in the Nichols diagram case.

Graduate versus omega. This option is available for the Nichols and Nyquist plots. It produces graduation of the diagram versus the angular frequency ω: the scale of graduation is marked with ticks, each corresponding to a displayed value of ω. It produces the interactive requests:

```
enter the font size (from 6 to 14, default 8) :

enter the density factor (from 1 to 10, default 1) :
```

The default font size usually results in a good graduation appearance, while choice of the density factor may require some trials (repeating graduation cancels the previous one). Graduation in Fig. 3.18 was obtained with density factor 1, while that in Fig. 3.23, which refers to a system with a finite delay, was obtained with density factor 4, and that in Fig. 3.24, which refers to a system with multiple resonances, with density factor 10. In any case, graduation versus angular frequency is a natural complement of Nichols and Nyquist diagrams. If several plots are present in the figure, it is first necessary to answer the request:

```
select function by entering color :
```

that allows graduation of one plot at a time.

Introduce a finite delay. This option allows taking into account the multiplication factor $e^{-t_0 s}$ in the continuous-time case, or z^{-k_0} in the discrete-time case. In the first case we obtain the request:

```
enter a finite delay (seconds) - default is zero :
```

and in the second case:

```
the current sampling time is 0.2 sec
enter a delay (number of samples) - default is zero :
```

As in the previous case, if several plots are present in the figure, it is first necessary to answer the request:

```
select function by entering color :
```

The introduction of a finite delay can be repeated for the same function. In this case the corresponding plot is drawn again.

3.7.3 Examples

Let

$$gp(s) = \frac{40}{s\,(s+1)\,(s+10)}\,,$$
$$gc(s) = \frac{10\,(s+1.413)}{s+14.13}$$

be the transfer functions of a plant and a compensator, located in a unit-feedback control loop. Denote by $gt(s) = gc(s)\,gp(s)$ the corresponding loop transfer function. Fig. 3.15 shows the Bode diagrams of $gp(j\omega)$, $gc(j\omega)$ and $gt(j\omega)$ in a single figure (option 3 of the input menu). Options 4, 1 and 6 in the main menu have been used to draw the three plots in different colors, to change the reference axes with respect to automatic scaling, and to draw the asymptotic approximation of $gc(j\omega)$. Fig. 3.16 gives the same diagrams in two separate figures (option 4 of the input menu); options 4, 1 and 6 in the main menu have been used as before. Fig. 3.17 shows the Nichols diagrams of $gp(j\omega)$ and $gt(j\omega)$ with the constant M and N loci added by means of option 7 in the main menu. Figs. 3.18 and 3.19 refer to the Nyquist diagrams of $gp(j\omega)$, $gc(j\omega)$ and $gt(j\omega)$ (option 6 of the input menu): in Fig. 3.18 option 8 of the main menu was used to graduate versus ω all the diagrams, while Fig. 3.19 shows the constant M and N loci, obtained with multiple selection of option 7. The arrows were added by using option 6. Figs. 3.20, 3.21 and 3.22 show some layouts referring to option 6. Fig. 3.23 refers to $gp(j\omega)$ with a 4 sec delay added with option 9 and graduated with option 8.

In all cases option 3 in the main menu may provide information on open-loop and closed-loop frequency response parameters corresponding to any one of the plots (whose color is specified by the user in the interactive mode); for instance, in the case in hand a phase margin increase from 17.7 degrees of $gp(j\omega)$ to 55.89 degrees of $gt(j\omega)$, due to the phase lead action of the compensator, points out a remarkable improvement in dynamic behavior, that is confirmed by the M constant loci test.

Finally, Fig. 3.24 shows the Nyquist diagram of the transfer function

$$\frac{(s+8.006)\,(s+19.67)\,(s^2-28.21\,s+216.2)\,(s^2+2.333\,s+2262)\,(s^2+3.607\,s+5284)}{(s^2+0.331\,s+8.998)\,(s^2+3.311\,s+361)\,(s^2+1.656\,s+2704)\,(s^2+4.139\,s+10410)}$$

that presents multiple resonances (thus requiring an efficient frequency-scale automatic setting), also graduated with option 8.

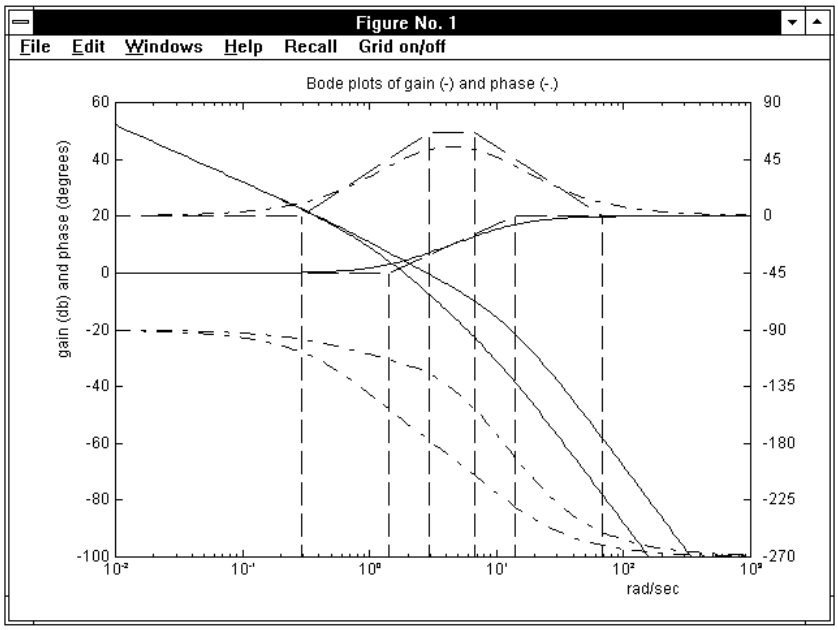

Figure 3.15. Bode diagrams in a single figure.

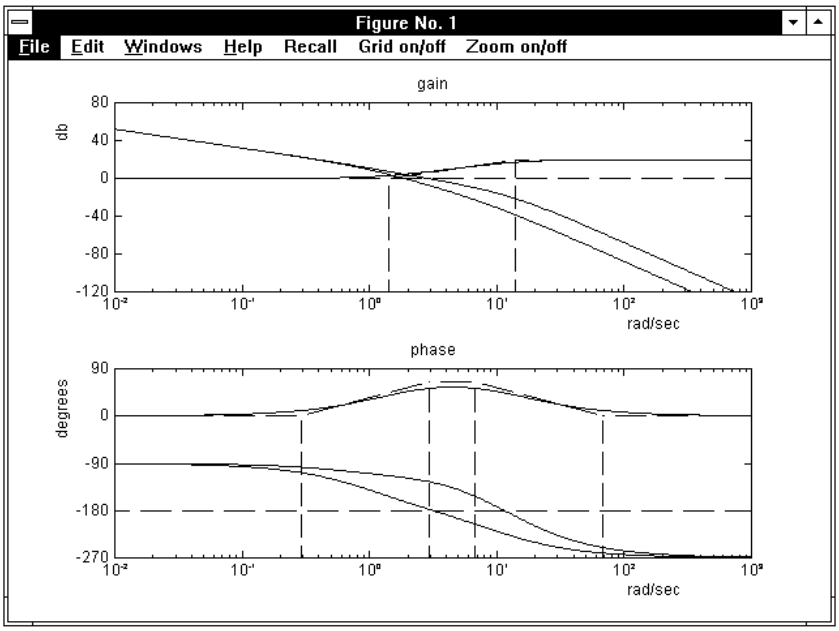

Figure 3.16. Bode diagrams in two separate figures.

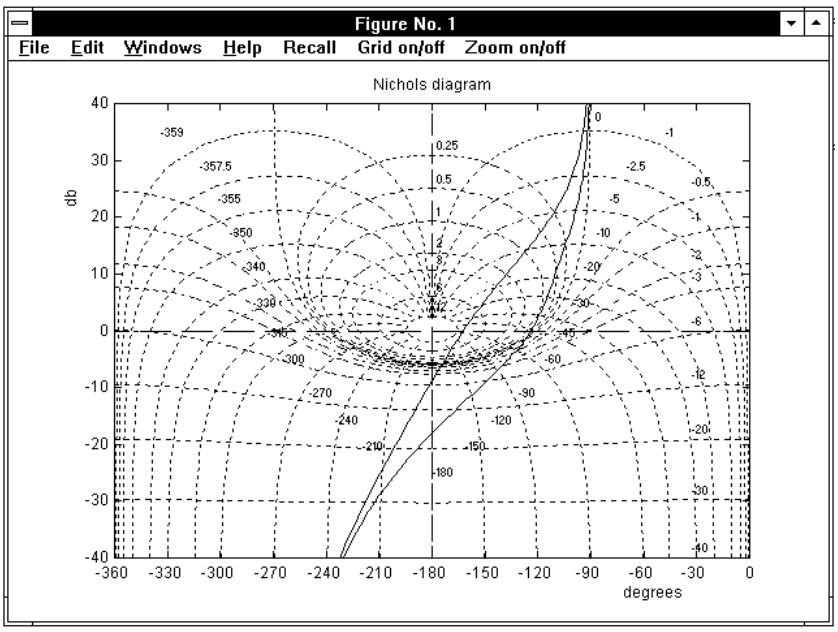

Figure 3.17. Nichols diagrams with M and N loci added.

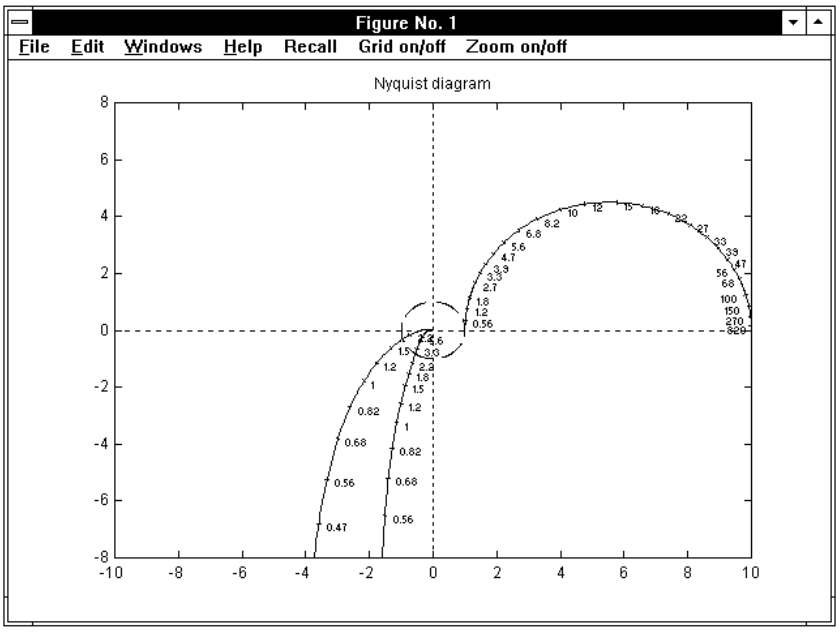

Figure 3.18. Nyquist diagrams with frequency graduation.

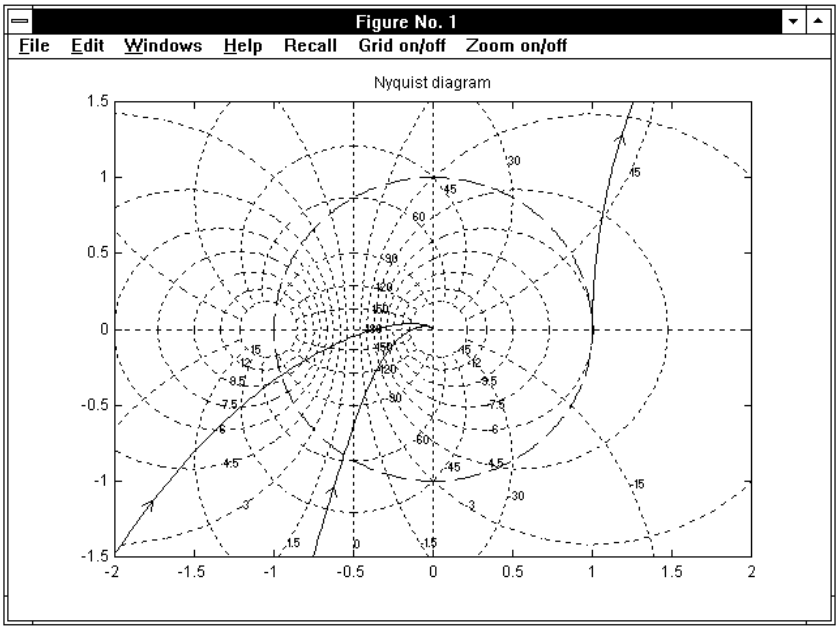

Figure 3.19. Nyquist diagrams with M and N loci added.

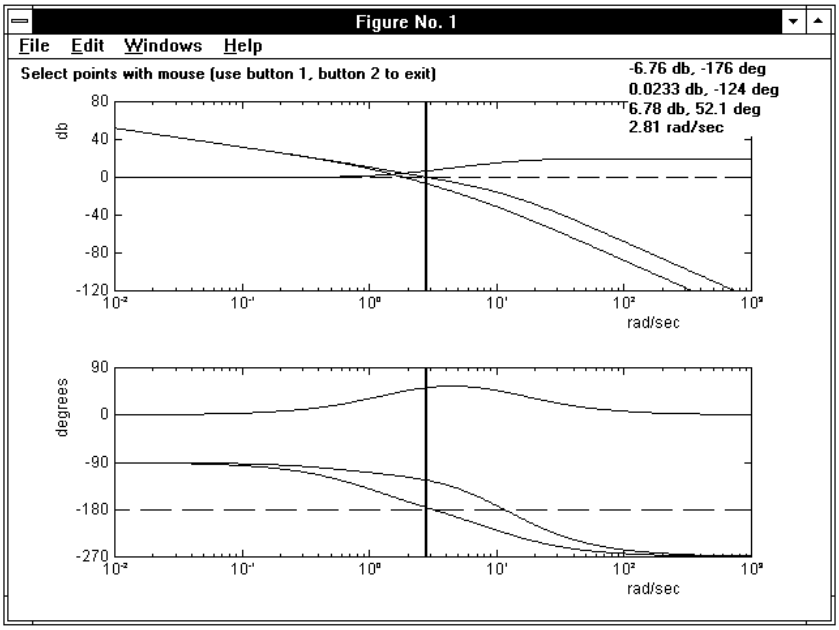

Figure 3.20. Information with mouse (option 6) in Bode diagrams.

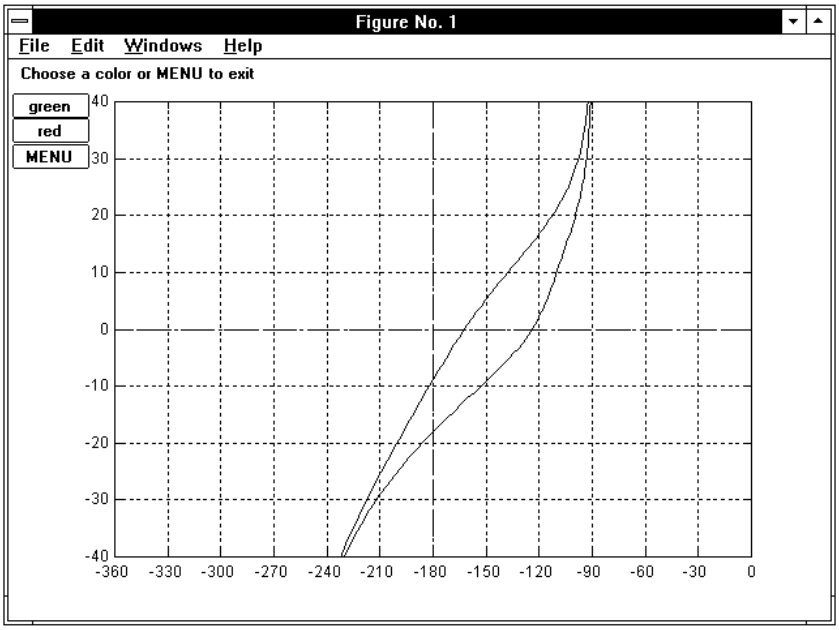

Figure 3.21. Information with mouse (option 6) in Nichols diagrams - step 1.

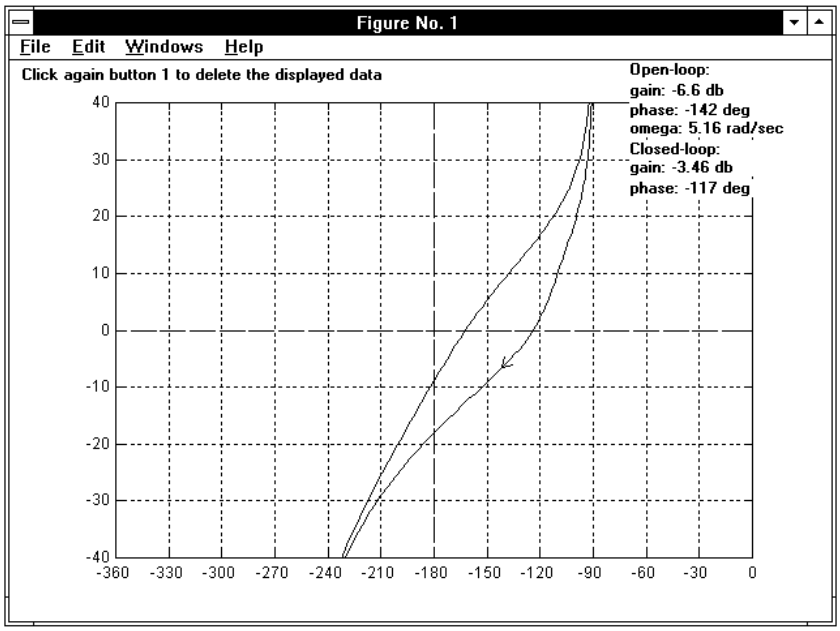

Figure 3.22. Information with mouse (option 6) in Nichols diagrams - step 2.

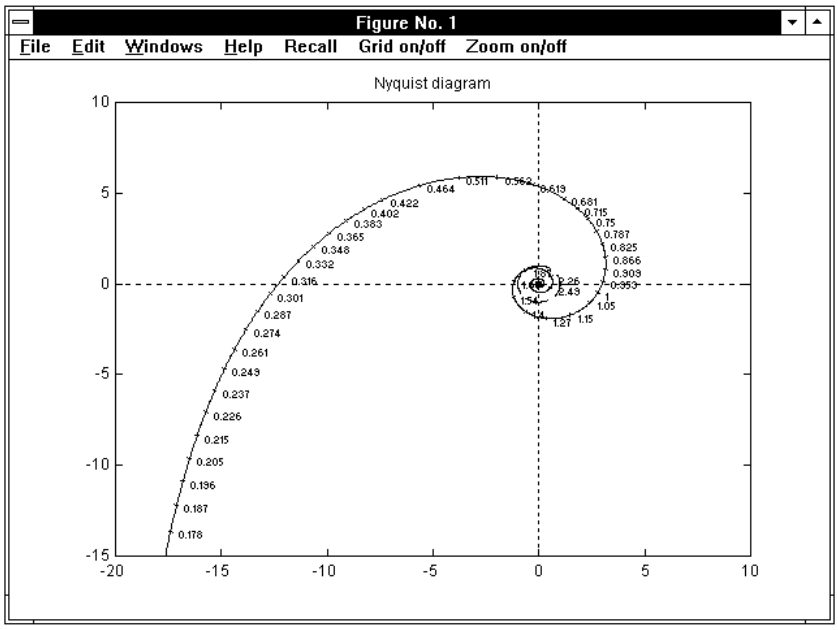

Figure 3.23. The Nyquist diagram of a system with a finite delay.

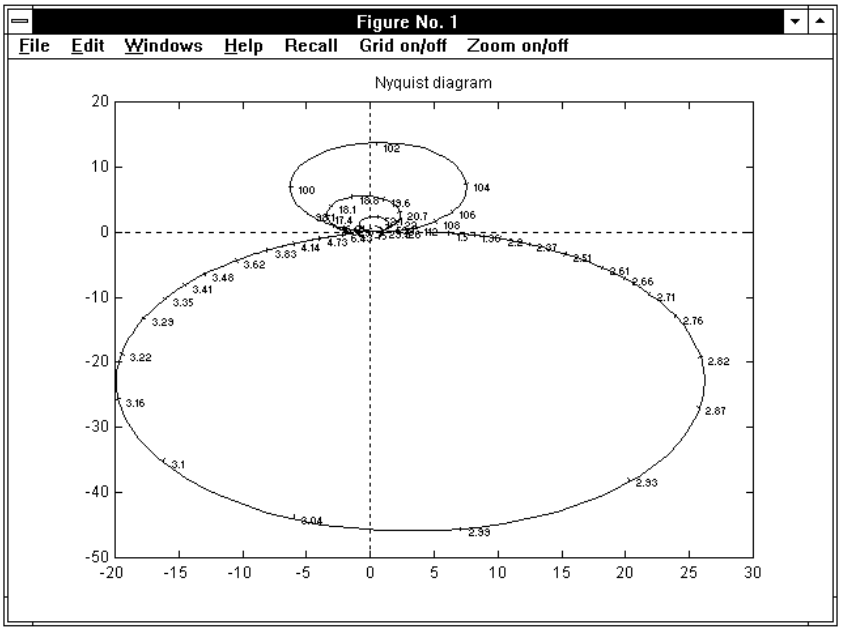

Figure 3.24. The Nyquist diagram of a system with multiple resonances.

3.8 Gpmarg

The command

`> gpmarg,gi[,1]` *(enter)*

displays the gain and phase margins of the continuous-time transfer function $gi(s)$ or the discrete-time transfer function $gi(z)$. Option [1] provides the generalized stability margins, with interactive request of the reference phase and gain values.

3.8.1 Recall

Let us consider the continuous-time transfer function

$$G(s) = \frac{N(s)}{D(s)} := \frac{b_m\, s^m + b_{m-1}\, s^{m-1} \ldots + b_2\, s^2 + b_1\, s + b_0}{s^n + a_{n-1}\, s^{n-1} + \ldots + a_2\, s^2 + a_1\, s + a_0} \ ,$$

and denote by $R_N(\omega)$ and $I_N(\omega)$ the real and imaginary part of $N(j\omega)$, and by $R_D(\omega)$ and $I_D(\omega)$ the real and imaginary part of $D(j\omega)$. They are defined in terms of the coefficients as

$$\begin{aligned} R_N(\omega) &= b_0 - b_2\,\omega^2 + b_4\,\omega^4 - b_6\,\omega^6 + \ldots \ , \\ I_N(\omega) &= b_1\,\omega - b_3\,\omega^3 + b_5\,\omega^5 - b_7\,\omega^7 + \ldots \ , \\ R_D(\omega) &= a_0 - a_2\,\omega^2 + a_4\,\omega^4 - a_6\,\omega^6 + \ldots \ , \\ I_D(\omega) &= a_1\,\omega - a_3\,\omega^3 + a_5\,\omega^5 - a_7\,\omega^7 + \ldots \ . \end{aligned}$$

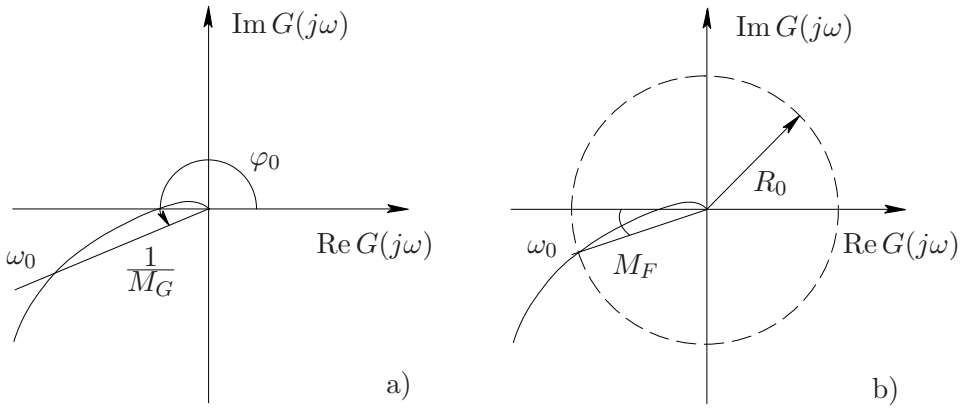

Figure 3.25. Derivation of the generalized gain and phase margins.

Generalized gain margin

Let us refer to Fig. 3.25,a: given an angle φ_0, we call the value $M_G := 1/G(j\omega_0)$, where ω_0 is the smallest angular frequency such that $\arg G(j\omega_0) = \varphi_0$, the *generalized gain margin* of $G(s)$. The standard gain margin usually referred to in the control system design corresponds to $\varphi_0 = \pi$.

The frequency ω_0 can be determined as a root of a polynomial equation as follows. Let us define

$$\alpha(\omega) := \arg N(j\omega) \ , \quad \text{so that} \quad \tan\alpha(\omega) = \frac{I_N(\omega)}{R_N(\omega)} \ ,$$
$$\beta(\omega) := \arg D(j\omega) \ , \quad \text{so that} \quad \tan\beta(\omega) = \frac{I_D(\omega)}{R_D(\omega)} \ .$$

By substitution, from

$$\tan(\arg G(j\omega)) = \tan(\alpha(\omega) - \beta(\omega)) = \frac{\tan\alpha(\omega)\ \tan\beta(\omega)}{1 + \tan\alpha(\omega)\ \tan\beta(\omega)} = \tan\varphi_0$$

we derive the polynomial equation

$$\sin\varphi_0\,(R_N(\omega)\,R_D(\omega) + I_N(\omega)\,I_D(\omega)) - \cos\varphi_0\,(I_N(\omega)\,R_D(\omega) - I_D(\omega)\,R_N(\omega)) = 0 \ ,$$

whose roots include ω_0. We select as ω_0 the least real nonzero root. Then, the generalized gain margin is derived as

$$M_G = \Big|\frac{N(j\omega_0)}{D(j\omega_0)}\Big|^{-1} \ .$$

Generalized phase margin

Let us now refer to Fig. 3.25,b: given a real positive number R_0, we call the value $M_P := \pi + \arg G(j\omega_0)$, where ω_0 is the smallest angular frequency such that $|G(j\omega_0)| = R_0$, *generalized phase margin* of $G(s)$. It is worth noting that $\arg G(j\omega)$ is assumed to be a continuous function at π. The standard phase margin usually referred to in the control system design corresponds to $R_0 = 1$.

The frequency ω_0 can be determined as a root of a polynomial equation as follows. From the identity

$$R_0^2 = \frac{|N(j\omega)|^2}{|D(j\omega)|^2}$$

we derive the polynomial equation

$$R_N^2(\omega) + I_N^2(\omega) - R_0^2\,(R_D^2(\omega) + I_D^2(\omega)) = 0 \ ,$$

whose roots include ω_0. We select as ω_0 the smallest real nonzero root. Then, the generalized phase margin is derived as

$$M_P = \pi + \arg \frac{N(j\omega_0)}{D(j\omega_0)} \ .$$

The above procedure to derive the generalized gain and phase margins can easily be extended to the discrete-time case. Let us recall that the frequency response corresponding to $G(z)$ is defined as $G(e^{j\omega T})$, but it can be computed as $G_w(j\omega)$, where $G_w(s)$ denotes the w-plane equivalent of $G(z)$ (see the *Recall* section of application *wplane*). Thus, the above method can be applied without any change if the transfer function $G_w(s)$ is considered in place of $G(z)$. However, the frequencies derived with the margins are points of the imaginary axis of the w-plane, while the frequency response is usually defined in terms of phase ωT along the unit circle of the z-plane. Let ω_0' be the angular frequency determined in the w-plane. From

$$z = e^{j\omega_0 T} = \frac{1 + \frac{j\omega_0' T}{2}}{1 - \frac{j\omega_0' T}{2}}$$

the corresponding ω_0 to be displayed is computed as

$$\omega_0 = \frac{1}{T} \arg \left(\frac{1 + \frac{j\omega_0' T}{2}}{1 - \frac{j\omega_0' T}{2}} \right) .$$

3.8.2 Operation and Examples

Let us consider the transfer function

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ .$$

On entering "gpmarg,gi" the following appears:

```
gain margin of transfer function gi(s): 2.75 (8.787 db)
(phase = -180 degrees) at frequency: 3.162 rad/sec

phase margin of transfer function gi(s): 17.7 degrees
(gain = 1) at frequency: 1.861 rad/sec}
```

while, on entering "gpmarg,gi,1", we obtain:

```
enter the reference phase for gain margin (default -180) : -150

enter the reference gain for phase margin (default 1)    : 2

gain margin of transfer function gi(s): 0.5416 (-5.326 db)
(phase = -150 degrees) at frequency: 1.306 rad/sec

phase margin of transfer function gi(s): 31.71 degrees
(gain = 2) at frequency: 1.244 rad/sec
```

where the values -150 and 2 have been entered by the user with keyboard.

As another example, let us consider

$$gj(s) = \frac{1}{(s+1)\,(s+2)}\ .$$

Since the Nyquist diagram of $gj(s)$ is completely contained in the unit circle and does not intersect the negative real axis, using the standard call "gpmarg,gj" we obtain:

```
gain margin non-computable

phase margin non-computable
```

while, with "gpmarg,gj,1" the following appears:

```
enter the reference phase for gain margin (default -180) : -100

enter the reference gain for phase margin (default 1)    : .2

gain margin of transfer function gj(s): 5.188 (14.3 db)
(phase = -100 degrees) at frequency: 1.703 rad/sec

phase margin of transfer function gj(s): 81.72 degrees
(gain = 0.2) at frequency: 1.649 rad/sec
```

with -100 and .2 typed by the user during the interactive session.

If function $gj(s)$ is multiplied by 10, thus obtaining

$$gk(s) = 10\, gj(s) = \frac{10}{(s+1)\,(s+2)} \ ,$$

with the standard references we have:

```
gain margin non-computable

phase margin of transfer function gk(s): 55.86 degrees
(gain = 1) at frequency: 2.759 rad/sec
```

since the negative real axis is still not intersected, but the unit circle is.

3.9 Invtr

The command

`> invtr,gi` *(enter)*

displays the expression in finite terms of the inverse $\mathcal{L}$-transform (impulse response) $gi(t)$ of the continuous-time transfer function $gi(s)$ or of the inverse $\mathcal{Z}$-transform (impulse response) $gi(k)$ of the discrete-time transfer function $gi(z)$.

3.9.1 Recall

In both cases the inverse transform is obtained through partial fraction expansion. Referring first to the continuous-time case, let n be the degree of the denominator and $m \leq n$ that of the numerator of $gi(s)$. It is well-known that

$$gi(s) = K_0 + \frac{P(s)}{Q(s)} = K_0 + \frac{P(s)}{(s-p_1)^{r_1}\,(s-p_2)^{r_2}\ldots(s-p_h)^{r_h}} = K_0 + \sum_{i=1}^{h}\sum_{\ell=1}^{r_i} \frac{K_{i\ell}}{(s-p_i)^{r_i-\ell+1}} \ ,$$

where constant K_0 is present only if $m=n$. In this case it is derived by first dividing the numerator by the denominator, thus obtaining as $P(s)/Q(s)$ a strictly proper fraction. In the above expression $p_1, \ldots p_h$ denote the distinct roots of $Q(s)$, i.e., the poles of $gi(s)$, and $r_1, \ldots, r_h$ their multiplicities: clearly $\sum_{i=1}^{h} r_i = n$. Constants $K_{i\ell}$ are defined by

$$K_{i\ell} = \frac{1}{(\ell-1)!}\,\frac{d^{\,\ell-1}}{ds^{\,\ell-1}}(s-p_i)^{r_i}\,\frac{P(s)}{Q(s)}\bigg|_{s=p_i} \qquad (i=1,\ldots,h;\ell=1,\ldots,r_i) \ .$$

The inverse $\mathcal{L}$-transform is obtained by inverting, term by term, the partial fraction expansion. We obtain

$$gi(t) = K_0\,\delta(t) + \sum_{i=1}^{h}\sum_{\ell=1}^{r_i} \frac{K_{i\ell}}{(r_i-\ell)!}\,t^{r_i-\ell}\,e^{p_i t} \ ,$$

where $\delta(t)$ denotes the Dirac impulse at $t=0$.

When p_i is complex, the conjugate complex $\bar{p}_i$ is also present with the same multiplicity r_i and with conjugate complex coefficients $\bar{K}_{i\ell}$. Let $p_i = \sigma_i + j\omega_i$, so that $e^{p_i t} = e^{\sigma_i t}\,(\cos\omega_i t + j\sin\omega_i t)$, and $K_{i\ell} = u_{i\ell} + j\,v_{i\ell} = M_{i\ell}\,(\cos\varphi_{i\ell} + j\sin\varphi_{i\ell})$; then

$$\begin{aligned}\frac{t^{r_i-\ell}\,(K_{i\ell}\,e^{p_i t} + \bar{K}_{i\ell}\,e^{\bar{p}_i t})}{(r_i-\ell)!} &= \frac{2\,t^{r_i-\ell}\,e^{\sigma_i t}}{(r_i-\ell)!}\,(u_{i\ell}\cos\omega_i t - v_{i\ell}\sin\omega_i t)\\ &= \frac{2\,M_{i\ell}\,t^{r_i-\ell}\,e^{\sigma_i t}}{(r_i-\ell)!}\,(\cos\omega_i t + \varphi_{i\ell})\\ &= \frac{2\,M_{i\ell}\,t^{r_i-\ell}\,e^{\sigma_i t}}{(r_i-\ell)!}\,(\sin\omega_i t + \varphi_{i\ell} + \frac{\pi}{2})\ .\end{aligned}$$

The first of the above expressions is called *Cartesian form*, the second *polar form in terms of cosine* and the third *polar form in terms of sine*. In the polar forms it is customary to assume the phase angle added to $\cos\omega_i t$ or $\sin\omega_i t$ in the default interval $[-\pi/2, \pi/2)$; this is obtained by shifting the angle by $h\,\pi$, with h integer, and multiplying the coefficient by $(-1)^{|h|}$.

In the discrete-time case some different manipulations are necessary. In fact, a possible pole at the origin with multiplicity r_0 is considered apart from the other ones in the partial fraction expansion, since the corresponding inverse $\mathcal{Z}$-transform cannot be expressed as a particular case of that of nonzero poles as in the continuous-time case. In this case we adopt the following contrivance: function $gi(z)$ is divided by z before expansion and multiplied by z after expansion, thus obtaining factor z in all terms, except, of course, those corresponding to the possible pole at the origin. We obtain

$$\begin{aligned}gi(z) = \frac{P(z)}{Q(z)} &= \frac{P(z)}{z^{r_0}\,(z-p_1)^{r_1}\,(z-p_2)^{r_2}\ldots(z-p_h)^{r_h}}\\ &= \sum_{\ell=1}^{r_0+1}\frac{K_{0\ell}}{z^{r_0-\ell+1}} + \sum_{i=1}^{h}\sum_{\ell=1}^{r_i}\frac{K_{i\ell}\,z}{(z-p_i)^{r_i-\ell+1}}\ ,\end{aligned}$$

where in this case h denotes the number of nonzero poles, so that $r_0 + \sum_{i=1}^{h} r_i = n$, and $P(z)/Q(z)$ is no longer required to be strictly proper.

The constants $K_{i\ell}$ are defined by

$$K_{i\ell} = \frac{1}{(\ell-1)!}\,\frac{d^{\,\ell-1}}{dz^{\,\ell-1}}(z-p_i)^{r_i}\,\frac{P(z)}{z\,Q(z)}\bigg|_{z=p_i} \qquad (i=0,\ldots,h;\ell=1,\ldots,\rho_i)\ ,$$

with $p_0=0,\ \rho_0=r_0+1,\ \rho_i=r_i\ \ (i=1,\ldots,h)$. Then, the inverse $\mathcal{Z}$-transform is obtained like the inverse $\mathcal{L}$-transform in the continuous-time case, by inverting term by term the partial fraction expansion. We get

$$gi(k) = \sum_{\ell=1}^{r_0+1} K_{0\ell}\,\delta(k-\ell+1) + \sum_{i=1}^{h}\sum_{\ell=1}^{r_i} \frac{K_{i\ell}}{{p_i}^{r_1-\ell}} \binom{k}{r_i-\ell} {p_i}^k\ ,$$

where $\delta(k-i)$ denotes the unitary impulse at $k=i$ and the binomial coefficient is defined as

$$\binom{k}{n} := \begin{cases} \dfrac{k(k-1)\ldots(k-n+1)}{n\,!} & \text{for } n>0\ , \\ 1 & \text{for } n=0\ . \end{cases}$$

Also in this case a complex conjugate pair of poles p_i, $\bar{p}_i$ have complex conjugate coefficients $K_{i\ell}$, $\bar{K}_{i\ell}$ in the expansion. Let $K_{i\ell}=M_{i\ell}\,e^{j\varphi_{i\ell}}=M_{i\ell}\,(\cos\varphi_{i\ell}+j\sin\varphi_{i\ell})$ as before, and $p_i=N_i\,e^{j\vartheta_i}=N_i\,(\cos\vartheta_i+j\sin\vartheta_i)$. Define the new parameters $K'_{i\ell}$, $u'_{i\ell}$, $v'_{i\ell}$, $M'_{i\ell}$ and $\varphi'_{i\ell}$ through

$$K'_{i\ell} = u'_{i\ell} + j\,v'_{i\ell} = M'_{i\ell}\,e^{j\varphi'_{i\ell}} = \frac{M_{i\ell}\,e^{j\varphi_{i\ell}}}{{N_i}^{r_i-\ell}\,e^{j(r_i-\ell)\vartheta_i}} = \frac{M_{i\ell}}{{N_i}^{r_i-\ell}}\,e^{j(\varphi_{i\ell}-(r_i-\ell)\vartheta_i)}\ ;$$

they enables every term of the inverse $\mathcal{Z}$-transform to be expressed in the three forms

$$\begin{aligned} \binom{k}{r_i-\ell}\,(K'_{i\ell}\,{p_i}^k + \bar{K}'_{i\ell}\,{\bar{p}_i}^k) &= 2\binom{k}{r_i-\ell}\,{N_i}^k\,(u'_{i\ell}\cos k\vartheta_i - v'_{i\ell}\sin k\vartheta_i) \\ &= 2\binom{k}{r_i-\ell}\,M'_{i\ell}\,{N_i}^k\,(\cos k\vartheta_i + \varphi'_{i\ell}) \\ &= 2\binom{k}{r_i-\ell}\,M'_{i\ell}\,{N_i}^k\,(\sin k\vartheta_i + \varphi'_{i\ell} + \frac{\pi}{2})\ , \end{aligned}$$

that are again called *Cartesian form*, *polar form in terms of cosine* and *polar form in terms of sine*, respectively.

3.9.2 Operation and Examples

We will show how the result is displayed by means of some examples. Consider the transfer functions

$$
\begin{aligned}
gi(s) &= \frac{1350\,(s+4)}{s\,(s+2)\,(s+3)^3}\ , \\
gj(z) &= \frac{1.56\,(z+0.31)\,(z-0.45)\,(z+1.47)}{(z-1)\,(z-0.67)\,(z-0.55)^3}\ , \\
gk(s) &= \frac{50\,(s+4)}{(s+2)\,(s^2+0.2\,s+1)^3}\ , \\
gw(z) &= \frac{1000\,(z+0.31)\,(z-0.45)\,(z+1.47)}{(z-0.67)\,(z^2+1.92\,z+0.96)^3}\ .
\end{aligned}
$$

If the transfer function considered has all the poles real the result is directly displayed in the Command Window. For $gi(s)$ we have:

```
Inverse Laplace Transform of gi(s) :
------------------------------------
gi(t) =   100
        - 1350*exp(-2 t)
        + (1250 + 1050 t + 225 t^2)*exp(-3 t)
```

while for $gj(z)$ the result displayed is:

```
Inverse Z Transform of gj(z) :
------------------------------
gj(k) =   (1789 + 323.3 k + 15.08 k^2)*(0.55)^k
        - 1884*(0.67)^k
        + 92.32
        + 2.87*delta(k)
```

If the transfer function considered has at least one pair of conjugate complex poles the following menu is displayed in the Command Window:

```
1 - complex modes in Cartesian form
2 - complex modes in polar form - function sine
3 - complex modes in polar form - function cosine

your choice :
```

Referring to $gk(s)$ the result for the three choices is displayed, respectively, as:

```
Inverse Laplace Transform of gk(s) :
-----------------------------------
gk(t) = 1.027*exp(-2 t)
        + [41.63 sin(0.995 t) - 1.027 cos(0.995 t)
        + (- 5.134 sin(0.995 t) - 39.47 cos(0.995 t)) t
        + (- 11.59 sin(0.995 t) + 2.745 cos(0.995 t)) t^2]
        *exp(-0.1 t)

Inverse Laplace Transform of gk(s) :
-----------------------------------
gk(t) =   1.027*exp(-2 t)
        + [41.64 sin(0.995 t - 0.02467)
          - 39.8 t sin(0.995 t + 1.441)
          - 11.91 t^2 sin(0.995 t - 0.2326)]*exp(-0.1 t)

Inverse Laplace Transform of gk(s) :
-----------------------------------
gk(t) =   1.027*exp(-2 t)
        + [- 41.64 cos(0.995 t + 1.546)
          - 39.8 t cos(0.995 t - 0.1293)
          + 11.91 t\ee 2 cos(0.995 t + 1.338)]*exp(-0.1 t)
```

while for $gw(z)$ the three choices of the menu give:

```
Inverse Z Transform of gw(z) :
------------------------------
gw(k) =   35.17*(0.67)^k
        + [3.786e+005 sin(2.94 k) - 381.1 cos(2.94 k)
          + (- 1.672e+004 sin(2.94 k) + 7.684e+004 cos(2.94 k)) k
          + (- 4488 sin(2.94 k) - 3481 cos(2.94 k)) k^2]*(0.979)^k
        + 345.9*delta(k)
```

```
Inverse Z Transform of gw(z) :
------------------------------
gw(k) =   35.17*(0.67)^k
        + [3.786e+005 sin(2.94 k - 0.001007)
          - 7.863e+004 k sin(2.94 k - 1.357)
          - 5680 k^2 sin(2.94 k + 0.6597)]*(0.9798)^k
        + 345.9*delta(k)

Inverse Z Transform of gw(z) :
------------------------------
gw(k) =   35.17*(0.67)^k
        + [- 3.786e+005 cos(2.94 k + 1.57)
          + 7.863e+005 k cos(2.94 k + 0.2142)
          - 5680 k^2 cos(2.94 k - 0.9111)]*(0.9798)^k
        + 345.9*delta(k)
```

The following further examples refer to multiple poles at the origin, both in the continuous and in the discrete-time case. Let us consider the following transfer functions

$$gc(s) = \frac{(s+1)(s+10)}{s^8} ,$$
$$gd(z) = \frac{(z-0.2)(z-0.8)}{z^8} .$$

For $gc(s)$ the corresponding information displayed is:

```
Inverse Laplace Transform of gc(s) :
------------------------------------
gc(t) =   (0.008333 t^5 + 0.01528 t^6 + 0.001984 t^7)
```

while for $gd(z)$ we obtain:

```
Inverse Z Transform of gd(z) :
------------------------------
gd(k) = + 1*delta(k-6) - 1*delta(k-7) + 0.16*delta(k-8)
```

thus pointing out that a multiple pole at the origin in discrete-time systems produces a time delay.

3.10 Lagc

The command

`> lagc,gi,gj` *(enter)*

provides trial-and-error design of a lag compensator $gj(s)$ for the plant $gi(s)$ by using the Bode diagrams. See the application *regnich* for design with the Nichols diagram.

3.10.1 Recall

The lag compensator improves the phase margin of a system by shifting the crossover frequency of the Bode gain diagram towards the left by means of a suitable gain reduction. Use of the lag compensator is restricted to type 0 or 1 systems, where acceptable phase margins are obtainable with a gain reduction.

With respect to simple gain reduction, the lag compensator has the advantage of preserving the dc gain, but, on the other hand, it also gives a certain small phase lag, that reduces the phase margin. We recall that the corresponding transfer function is

$$G_j(s) = \frac{1 + \alpha\,\tau\,s}{1 + \tau\,s} \ ,$$

with a significant phase lag in the angular frequency range $1/\tau \leq \omega \leq 1/(\alpha\tau)$, where the asymptotic Bode gain diagram has a -20 db/decade slope. The maximum phase lag, given by

$$\varphi_0 = -\arcsin\frac{1-\alpha}{1+\alpha} \ ,$$

hence depending only on α, occurs at the mid-band frequency

$$\omega_0 = \frac{1}{\tau\sqrt{\alpha}} \ .$$

Since phase lag makes stability worse, in the design the crossover frequency of the overall system must be located close to the right extreme of the above frequency range, where it is small, and the corresponding phase margin reduction is compensated by assuming a conservative value for α.

The program uses the following design procedure:

1. the phase margin φ_m and the corresponding angular frequency ω_m of the controlled system are computed and displayed;

2. the required phase margin φ_d is entered by the user, and the value of the gain reduction α_0 which gives this phase margin and the corresponding angular frequency ω_0 are computed and displayed;

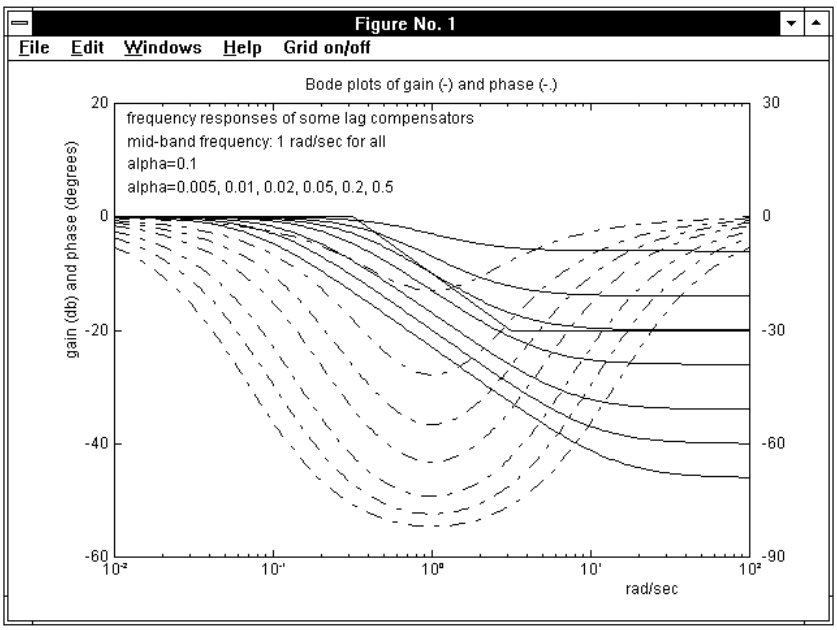

Figure 3.26. The Bode diagrams of some lag compensators.

3. the value $\alpha = \alpha_0/2$ is chosen for the first-trial lag compensator;

4. τ is varied between the values $100/(\omega_0\,\alpha_0)$ and $1/(10\,\omega_0\,\alpha_0)$ by 400 uniformly spaced steps and the value corresponding to the overall system phase margin closest to φ_d is selected and displayed; the step responses of the overall system and the compensator output are also plotted;

5. if these responses are unsatisfactory, the design procedure can be repeated from step 3 on with a different value of α, interactively specified by the user in the range from $\alpha_0/10$ (rounded to the lower bound of the corresponding decade) to α_0.

Let us point out that in the above-outlined synthesis procedure the value of the phase margin that is entered at the beginning not only influences the value of α automatically selected for the first trial, but is really imposed at step 4. It follows that it is necessary to enter *lagc* again with a different phase margin if the result remains unsatisfactory after several trials with different values of α, contrary to *leadc*, where the result is independent of the phase margin initially entered, since the maximum obtainable phase margin is selected at each trial.

3.10.2 Operation and Examples

Let us consider the transfer function

$$gi(s) = \frac{10000}{(s+1)\,(s+2)\,(s+10)} \ .$$

The command "lagc,gi,gj" first displays a medium-size figure with a block diagram showing the connection referred to, and the message:

```
**** press return to continue
```

When the return key is pressed, we have the request:

```
information on design of lag compensator ? (1) :
```

Entering 1 provides a short description of the procedure used to derive the lag compensator, that is not repeated here since it is contained in the previous *Recall* section. This optional information also generates the full-size figure with the Bode diagrams shown in Fig. 3.26.

If the request is skipped by simply pressing the return key, the Bode diagrams of $gi(s)$ are plotted with the reference lines pointing out the gain margin (blue) and phase margin (red). The numerical values of gain and phase margins are also displayed inside the diagram frame. On pressing the return key, the Command Window is recovered and the following display appears:

```
phase margin without compensator: -56.04 degrees
                    at frequency: 20.78 rad/sec

enter the required phase margin : 60
```

Let us enter the phase margin 60 for the first set of trials. First, the following lines appear in the display, then the previous Bode diagrams are shown again, with a dotted cyan line added pointing out the gain reduction providing the desired phase margin (Fig. 3.27). This serves to set the value of alpha for the first trial:

```
maximum alpha: 0.006485
minimum alpha: 0.0001

the selected alpha: 0.003242

**** press any key to continue
```

On pressing a key, the step-by-step construction to derive the compensator begins. First, the uncompensated system Bode plots are shown again in green. When the return key is pressed, the gain diagrams referring to the compensator corner frequencies search interval are shown in yellow, then, on pressing return again, the plots of the derived compensator are shown in cyan and eventually, on pressing return again, the plots of the corrected overall systems are shown in magenta, with a vertical magenta dotted line pointing out the phase margin obtained. At this point, the screen appears as in Fig. 3.28.

When the return key is pressed to exit from the Bode diagrams so obtained, the corrected system step response is shown to evaluate the transient behavior with the compensator derived. The output of the compensator is also plotted subsequently, to show the transient behavior of the manipulated variable. When the return key is pressed again, the following display appears on the Command Window:

```
reference color: g ; transfer function of the plant: gi

400 step search for tau in the selected interval:
required phase margin found at step: 218

phase margin without compensator: -56.05
phase margin with compensator   : 59.88

the compensator derived:
alpha = 0.003242 , tau = 424.2 sec

you may change the values of alpha; tau will be
automatically set to obtain the required phase margin

enter alpha (min 0.0001, max 0.006485), return to exit :
```

We assume that the obtained step responses are not satisfactory and decide to proceed with a second trial. Decreasing the value of alpha usually produces reduction of the maximum overshoot but increase in the settling time. We enter .001 as the new value of alpha. The construction is repeated and the corresponding Bode diagrams shown. The step responses of the subsequent trials are all shown together in different colors according to the sequence: *g, r, c, y, m, b, w* or *k* (the maximum number of trials is seven), so that effects of the parameter changes are easily noted. The step responses of the closed-loop system at the output of the plant and of the compensator, corresponding to the two trials performed so far in the example on hand, are shown in Figs. 3.29 and 3.30, with plots in green and red, respectively. Note the pop-up menus **Recall**, **Time axis** and **Grid on/off**.

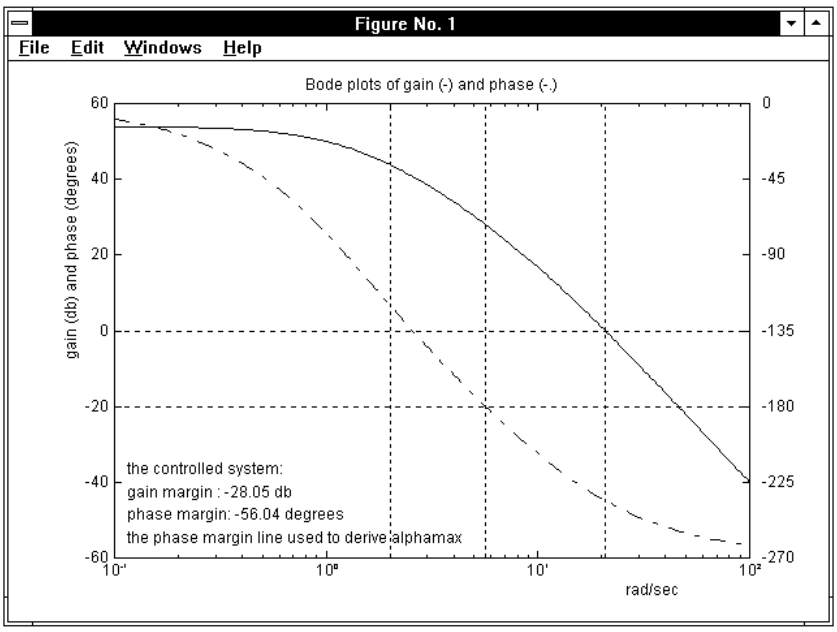

Figure 3.27. The Bode diagrams of the controlled system.

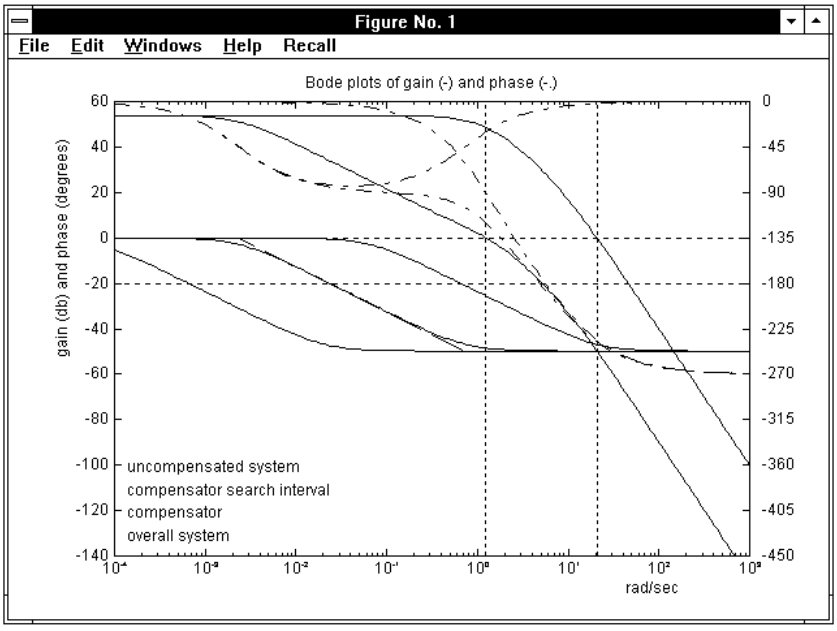

Figure 3.28. The Bode diagrams the end of the design procedure.

On pressing return, the result obtained is displayed in the Command Window as:

```
reference color: r ; transfer function of the plant: gi

400 step search for tau in the selected interval:
required phase margin found at step: 256

phase margin without compensator: -56.05
phase margin with compensator   : 60.09

the compensator derived:
alpha = 0.001 , tau = 690 sec

you may change the values of alpha; tau will be
automatically set to obtain the required phase margin

enter alpha (min 0.0001, max 0.006485), return to exit :
```

Since the maximum overshoot of the step response is satisfactory in this case, we decide to exit. The following appears:

```
select function by entering color :
```

On entering r (the color of the second-trial step response plots) the corresponding transfer function $gj(s)$ is saved in the hard disk and displayed as:

```
THE COMPENSATOR OBTAINED :

alpha = 0.001 , tau = 690 sec

     0.001 (s + 1.449)
gj = -----------------
     (s + 0.001449)
```

NOTES:

- As mentioned in the previous *Recall* section, during the trial-and-error design procedure the phase margin originally introduced has not been changed. If the results obtained after some trials are far from being satisfactory, it is necessary to exit and enter the program again to introduce a new phase margin.

- If *lagc* is applied to a system of type 2 or more, the following message appears:

```
**** lag correction applies only to type 0 or 1 systems
```

and the program is quitted.

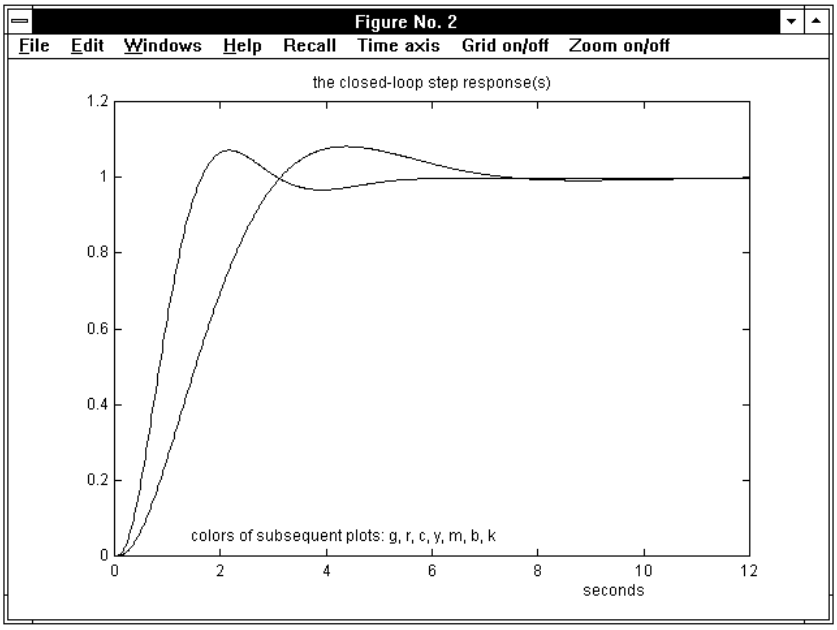

Figure 3.29. The closed-loop step responses.

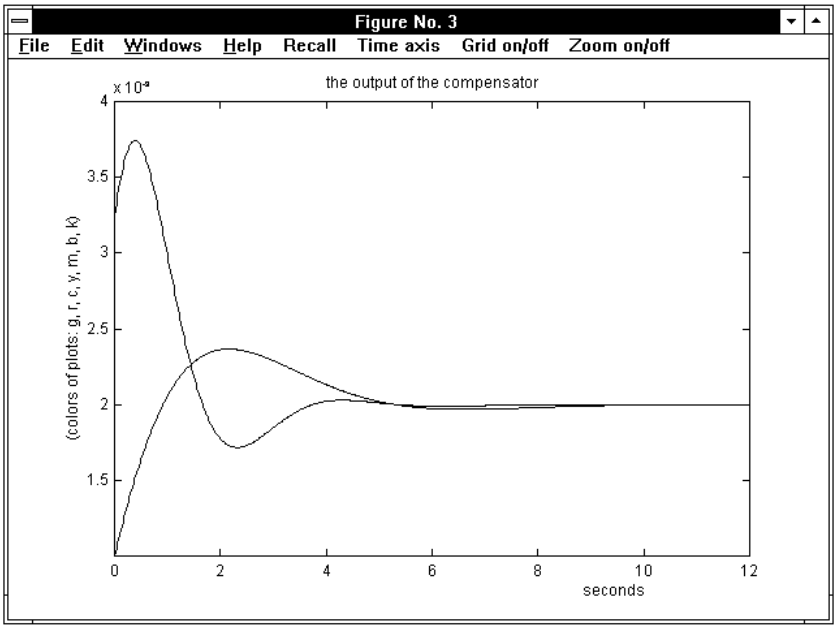

Figure 3.30. The compensator outputs.

3.11 Leadc

The command

```
> leadc,gi,gj   (enter)
```

provides trial-and-error design of a lead compensator $gj(s)$ for the plant $gi(s)$ by using the Bode diagrams. See the application *regnich* for design with the Nichols diagram.

3.11.1 Recall

The lead compensator directly improves the phase margin of a system, since its frequency response provides a phase lead in a suitable frequency range. We recall that the corresponding transfer function is

$$G_j(s) = \frac{1 + \tau\, s}{1 + \alpha\, \tau\, s} \ .$$

The phase lead is significant in the angular frequency range $1/\tau \leq \omega \leq 1/(\alpha\tau)$, where the asymptotic Bode gain diagram has a +20 db/decade slope. The maximum phase lead, given by

$$\varphi_0 = \arcsin \frac{1 - \alpha}{1 + \alpha} \ ,$$

hence depending only on α, occurs at the mid-band frequency

$$\omega_0 = \frac{1}{\tau \sqrt{\alpha}} \ .$$

The value of φ_0 is 90 degrees at the limit for α approaching zero. Since the value of α must be finite, it is advisable to set a minimum value for it in the design (.005, corresponding to about 82 degrees of maximum phase lead); in the interactive synthesis procedure smaller values are rejected.

Let us note that a single lead compensator cannot stabilize, with a good phase margin, a system having large negative phase margin, just because the obtainable phase lead is less than 90 degrees. If the phase margin of the controlled system is negative, a warning message is displayed, and the possibility of running the *leadc* program two or more times in order to achieve a multiple lead compensator is pointed out as a hint for the designer.

The program uses the following design procedure:

1. the phase margin φ_m and the corresponding angular frequency ω_m of the controlled system are computed and displayed;

2. the required phase margin φ_d is entered by the user, and the value α_0 corresponding to mid-band phase lead equal to $\varphi_d - \varphi_m$ is computed and displayed;

3. the value $\alpha = \alpha_0/2$ is chosen for the first-trial lead compensator;

4. τ is varied between the values $\sqrt{\alpha}/\omega_m$ and $1/(\omega_m \sqrt{\alpha})$ by 100 uniformly spaced steps and the value corresponding to the overall system maximum phase margin is selected and displayed; the step responses of the overall system and the compensator output are also plotted;

5. if these responses are unsatisfactory, the design procedure can be repeated from step 3 on with a different value of α, interactively specified by the user in the range from 0.005 to $2\alpha_0$.

Let us point out that in the above-outlined synthesis procedure the value of the phase margin that is entered at the beginning only influences the value of α automatically selected for the first trial. It does not influence the final result if more than one trial is done, since for each trial the phase margin is maximized. Then, if the result remains unsatisfactory after several trials with different values of α, it is not necessary to enter *leadc* again with a different phase margin, contrary to *lagc* and *pidc*, where the results largely depend on the phase margin initially entered, since this is strictly imposed and obtained at every trial.

3.11.2 Operation and Examples

Let us consider the transfer function

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ .$$

The command "leadc,gi,gj" displays a medium-size figure with the system block diagram informing about the connection referred to, and the message:

```
**** press return to continue
```

When the return key is pressed, we have the request:

```
information on design of lead compensator ? (1) :
```

Entering 1 provides a short description of the procedure used to derive the lead compensator, that is not repeated here since it is contained in the previous *Recall* section. This optional information also generates the full-size figure with the Bode diagrams shown in Fig. 3.31.

If the request is skipped by simply pressing the return key, the Bode diagrams of $gi(s)$ are displayed with the reference lines pointing out the gain margin (blue) and phase margin (red). The numerical values of gain and phase margins are also displayed inside the diagram frame. On pressing the return key again, the Command Window is recovered and the following display appears:

```
phase margin without compensator: 17.7 degrees
                    at frequency: 1.862 rad/sec

enter the required phase margin :
```

Let us enter the phase margin 60. The following lines are added in the display:

```
necessary phase lead: 42.3 degrees
the selected alpha  : 0.09772

**** press any key to continue
```

When a key is pressed, the step-by-step construction to derive the compensator begins. First, the Bode plots of the uncompensated system are shown again in green. When the return key is pressed, the gain diagrams referring to the compensator corner frequencies search interval are shown in yellow, then, on pressing return again, the plots of the derived compensator are shown in cyan and eventually, on pressing return again, the plots of the corrected overall systems are shown in magenta, with a vertical magenta dotted line pointing out the phase margin obtained. The corresponding screen layout is shown in Fig. 3.32.

When the return key is pressed to exit from the Bode diagrams so obtained, the step response of the corrected system is shown to evaluate the closed-loop behavior with the compensator derived. The output of the compensator is also plotted subsequently, to show the peak value of the manipulated variable due to the derivative action.

When the return key is pressed again, the following display appears on the Command Window:

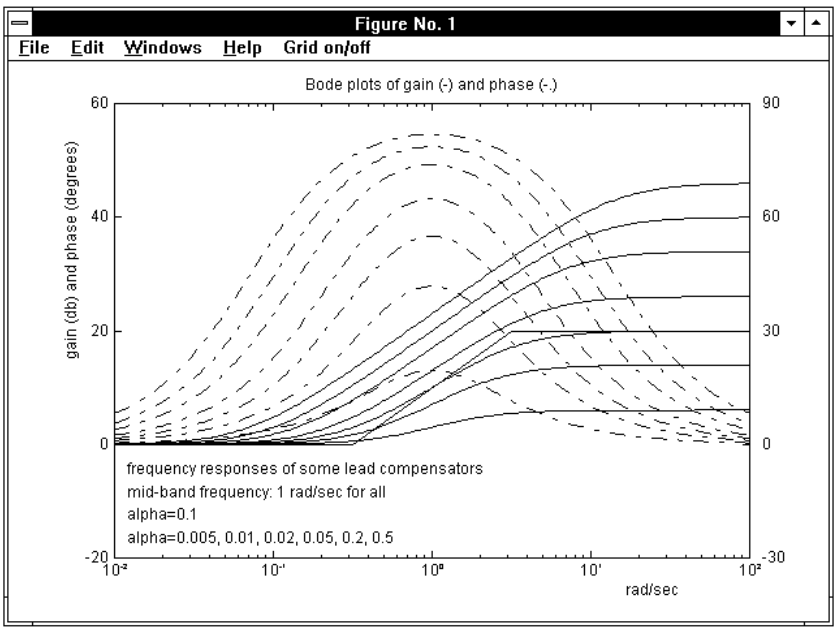

Figure 3.31. The Bode diagrams of some lead compensators.

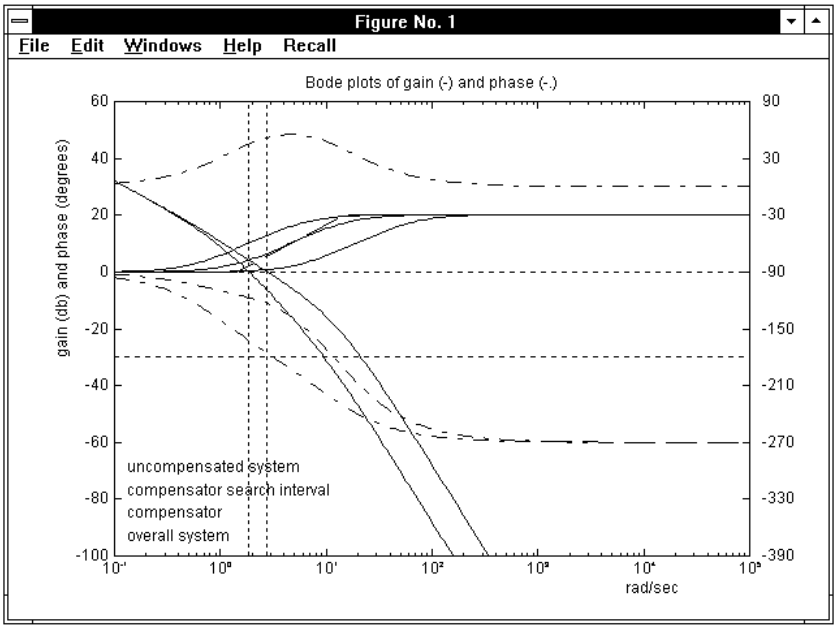

Figure 3.32. The Bode diagrams the end of the design procedure.

```
reference color: g ; transfer function of the plant : gi

100 step search for tau in the selected interval:
maximum phase margin found at step: 46

phase margin without compensator: 17.68
phase margin with compensator   : 56.13

the compensator derived:
alpha = 0.09772 , tau = 0.69 sec

you may change the values of alpha; tau will be
automatically set to obtain the maximum phase margin

enter alpha (min .005, max 0.3909), return to exit :
```

We assume that the step responses obtained are not satisfactory and decide to proceed with a second trial. Decreasing the value of alpha produces a higher phase margin, hence a reduction of the maximum overshoot in the step response. We enter .06 as the new value of alpha. The construction is repeated and the corresponding Bode diagrams shown. The step responses of the subsequent trials are all shown together in different colors according to the sequence: *g, r, c, y, m, b, w* or *k* (the maximum number of trials is seven), so that the effects of the parameter changes are easily noted. By pressing the return key, the Command Window is selected again and the result obtained is displayed as:

```
reference color: r ; transfer function of the plant : gi

100 step search for tau in the selected interval:
maximum phase margin found at step: 44

phase margin without compensator: 17.68
phase margin with compensator   : 60.31

the compensator derived:
alpha = 0.06 , tau = 0.7642 sec

you may change the values of alpha; tau will be
automatically set to obtain the maximum phase margin

enter alpha (min .005, max 0.3909), return to exit :
```

The step responses of the closed-loop system at the output of the plant and of the compensator, corresponding to the two trials performed so far in the example on hand, are shown in Figs. 3.33 and 3.34. In the top control bar above these figures the pop-up menus **Recall**, **Time axis** and **Grid on/off** make it possible to see the parameters of the regulators derived, to change the time axis, and to see or remove the grid.

Since the maximum overshoot and the phase margin are satisfactory in this case, we decide to exit. The following appears:

```
select function by entering color :
```

On entering r (the color of the second trial step response plots), we obtain

```
THE COMPENSATOR OBTAINED :

alpha = 0.06 , tau = 0.7642 sec

     16.67 (s + 1.308)
gj = -----------------
     (s + 21.81)}
```

Of course, the transfer function $gj(s)$ is also saved in the hard disk.

NOTE:

- As mentioned in the previous *Recall* section, a single lead compensator may not be sufficient when the phase margin of the uncompensated system is far from the desired value. If this is negative, the following warning message and related option for possible exit are displayed before the reference phase margin for the design is requested:

```
warning: negative phase margin - this case probably requires
a lag or two or more lead compensators; in the latter case
you may run this program two or more times

do you want to continue ? (1/0) : 1

enter the required phase margin :
```

and the above design procedure is enabled. If a negative phase margin is still obtained at the end of the first trial, the corresponding step responses are not shown.

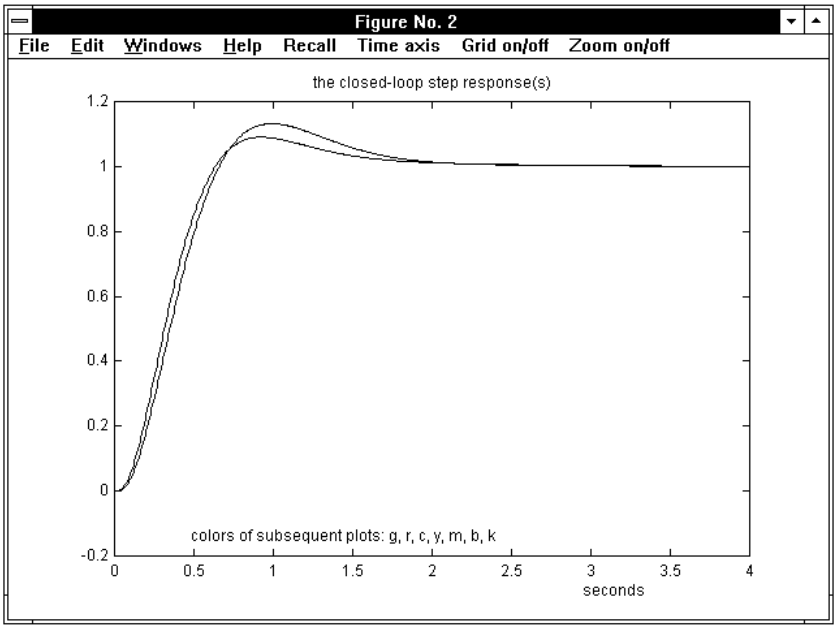

Figure 3.33. The closed-loop step responses.

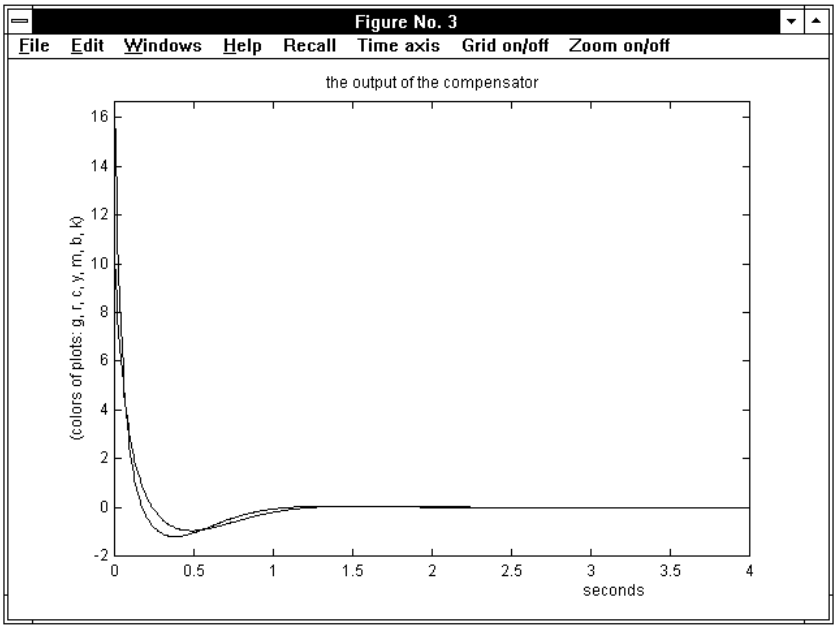

Figure 3.34. The compensator outputs.

3.12 Makeleg

The command

`> makeleg` *(enter)*

creates a legend in the last figure generated by a TFI application. In Matlab 4, where the legend may be covered by the grid or other spurious lines, particularly after shifting with the mouse, command "makeleg" can also be used to clean a legend already present in the figure.

3.12.1 Operation

When a legend in figures is not permanently set by using the corresponding option provided by the "startint" command, it is possible to obtain it with "makeleg", that operates when the application is quitted and the figure generated is reduced to small size and located in the upper-right corner of the screen. The figure is selected, enlarged at full-screen size and the legend is created on it. The figure is reduced to small again when the return key is pressed.

If the currently selected figure (e.g., figure no. 1) is not the last figure generated (e.g., figure no. 3), the message:

```
**** error: you must select figure 3
```

is displayed and the legend is not created, while if the selected figure is the last one generated, but by an application that does not support legend, the following appears:

```
**** legend is not available in this case
```

The legend facility is available both in Matlab 4 and in Matlab 5, but has a slightly different appearance and behavior.

3.13 Nlsim

The command

```
> nlsim,gi,gj,gk,gw  (enter)
```

performs the simulation of a loop where $gi(s)$ is the transfer function of a linear block between the summing junction and an algebraic nonlinear element NL, $gj(s)$ that of a linear block between NL and the output, $gk(s)$ that of the feedback connection, while $gw(s)$ is the $\mathcal{L}$-transform of the reference signal. Usually $gi(s)$ is the transfer function of the regulator and $gj(s)$ that of the plant. Any positive real number (typically 1) can be entered in place of $gi(s)$ and/or $gk(s)$.

3.13.1 Recall

The block diagram of the feedback system considered is shown in Fig. 3.35. The *exosystem* is modelled by $G_w(s)$, i.e., the reference y_e is obtained as the Dirac impulse response or the inverse $\mathcal{L}$-transform of the ratio of polynomilas $G_w(s)$, while $G_i(s)$ is the transfer function of the *regulator*, $G_j(s)$ that of the *plant*, and $G_k(s)$ that of the *feedback connection*.

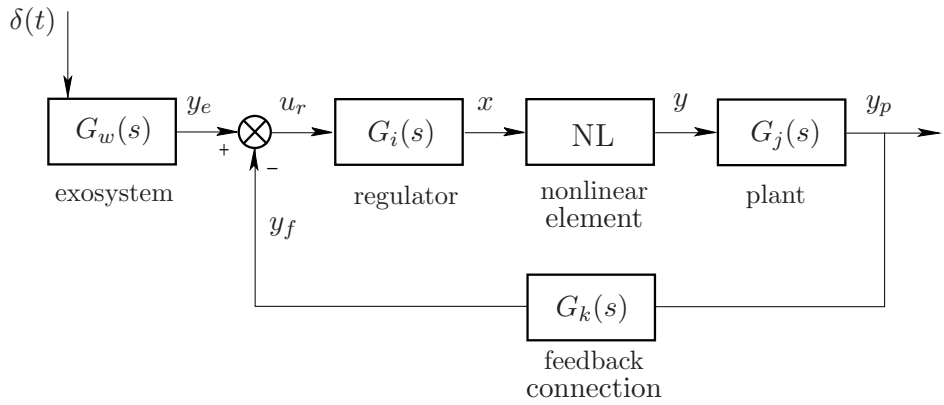

Figure 3.35. The feedback system considered.

The *nonlinear element NL* can be interactively selected among the following standard types:

1 - saturation
2 - dead zone
3 - saturation with dead zone
4 - any linearly interpolated nonlinearity

5 - ideal relay
6 - relay with dead zone
7 - relay with hysteresis
8 - backlash

Every type, after selection, is defined by entering the values of some parameters, that are requested in interactive mode. See the *Recall* section of application *descrf* for more information on the above nonlinearities.

The overall system is shown in Fig. 38. Its linear part can be described with the state-space model

$$\begin{aligned} &\dot{\eta}(t) = A\,\eta(t) + b\,y(t)\ , \quad \eta(0) = \eta_0\ , \\ &x(t) = c\,\eta(t)\ , \end{aligned} \tag{3.5}$$

where η_0 denotes the initial state (due to the Dirac impulse), $x(t)$ and $y(t)$ the input and output of *NL*, that are an output and the input of the linear part.

A, b and c are built with the matrices corresponding to the state-space descriptions of the four linear blocks in Fig. 3.35, that for the plant, the feedback connection, the regulator and the exosystem are, respectively

$$\begin{aligned} &\dot{x}_p(t) = A_p\,x_p(t) + b_p\,u_p(t)\,, && \dot{x}_f(t) = A_f\,x_f(t) + b_f\,u_f(t)\,, \\ &y_p(t) = c_p\,x_p(t)\,; && y_f(t) = c_f\,x_f(t) + d_f\,u_f(t)\,; \\ \\ &\dot{x}_r(t) = A_r\,x_r(t) + b_r\,u_r(t)\,, && \dot{x}_e(t) = A_e\,x_e(t) + b_e\,u_e(t)\,, \\ &y_r(t) = c_r\,x_r(t) + d_r\,u_r(t)\,; && y_e(t) = c_e\,x_e(t)\,; \end{aligned}$$

with, clearly, $u_p(t) = y(t)$. The connection equation corresponding to the summing junction is

$$\begin{aligned} u_r(t) &= y_e(t) - y_f(t) \\ &= c_e\,x_e(t) - c_f\,x_f(t) - d_f\,u_f(t) = c_e\,x_e(t) - c_f\,x_f(t) - d_f\,c_p\,x_p(t)\ , \end{aligned}$$

so that, by substitution, the output $x(t) = y_r(t)$ is obtained as

$$x(t) = c_r\,x_r(t) + d_r\,u_r(t) = c_r\,x_r(t) - d_r\,c_e\,x_e(t) - d_r\,c_f\,x_f(t) - d_r\,d_f\,c_p\,x_p(t)\ . \tag{3.6}$$

Taking into account other straightforward connection equations, we obtain the set of differential equations

$$\begin{aligned}
\dot{x}_p(t) &= A_p\, x_p(t) + b_p\, y(t)\,,\\
\dot{x}_f(t) &= A_f\, x_f(t) + b_f\, c_p\, x_p(t)\,,\\
\dot{x}_r(t) &= A_r\, x_r(t) + b_r\, c_e\, x_e(t) - b_r\, c_f\, x_f(t) - b_r\, d_f\, c_p\, x_p(t)\,,\\
\dot{x}_e(t) &= A_e\, x_e(t) + b_e\, \delta(t)\,,
\end{aligned}$$

that, together with (3.6), completely describe the linear part of the system. It follows that the matrices appearing in (3.5) are defined in terms of those of the connected linear subsystems as

$$\eta = \begin{bmatrix} x_p \\ x_f \\ x_r \\ x_e \end{bmatrix}\,,\ A = \begin{bmatrix} A_p & O & O & O \\ b_f\, c_p & A_f & O & O \\ -b_r\, d_f\, c_p & -b_r\, c_f & A_r & b_r\, c_e \\ O & O & O & A_e \end{bmatrix}\,,\ b = \begin{bmatrix} b_p \\ o \\ o \\ o \end{bmatrix}\,,\ \eta_0 = \begin{bmatrix} o \\ o \\ o \\ b_e \end{bmatrix}\,,$$

$$c = \begin{bmatrix} -d_r\, d_f\, c_p & -d_r\, c_f & c_r & d_r\, c_e \end{bmatrix}\,.$$

Assuming that NL is described by a generic algebraic function $y(t) = f(x(t))$, the solution of the overall system can be obtained by step-by-step solution of the corresponding discrete-time system

$$\begin{aligned}
\eta(k+1) &= A_d\, \eta(k) + b_d\, y(k)\,, \quad \eta(0) = \eta_0\,,\\
x(k) &= c\, \eta(k)\,,\\
y(k) &= f(x(k))\,.
\end{aligned}$$

Let Δt be the assumed time discretization step. Matrix A_d and row vector b_d are derived from

$$e^{\hat{A}\,\Delta t} = \begin{bmatrix} A_d & b_d \\ o & 1 \end{bmatrix}\,,\ \text{with}\ \ \hat{A} := \begin{bmatrix} A & b \\ o & 0 \end{bmatrix}\,.$$

The other signals appearing in the plots, i.e., the reference (the output of the exosystem) and the output of the plant, can be derived as linear functions of the state variables of the above nonlinear discrete-time system as

$$\begin{aligned}
y_e(k) &= \begin{bmatrix} o & o & o & c_e \end{bmatrix}\, \eta(k)\,,\\
y_p(k) &= \begin{bmatrix} c_p & o & o & o \end{bmatrix}\, \eta(k)\,.
\end{aligned}$$

3.13.2 Operation and Examples

Let

$$gi(s) = \frac{10\,(s + 1.413)}{s + 14.13}\ , \qquad gj(s) = \frac{40}{s\,(s+1)\,(s+10)}$$

be the transfer functions of the regulator and the plant shown in Fig. 3.35, and

$$step(s) = \frac{5}{s}$$

the $\mathcal{L}$-transform of the reference, i.e. the transfer function of the exosystem. The command "nlsim,gi,gj,1,step" displays a medium-size figure with a block diagram giving information on the connection referred to, and the message:

```
**** press return to continue
```

to be displayed in the Command Window. When the return key is pressed, the following appears:

```
Choose the nonlinear element :

 1 - saturation
 2 - dead zone
 3 - saturation with dead zone
 4 - any linearly interpolated nonlinearity
 5 - ideal relay
 6 - relay with dead zone
 7 - relay with hysteresis
 8 - backlash

enter your choice (press return to exit) :
```

If, for instance, option 5 (ideal relay) is entered, the corresponding parameters are requested and communicated as follows:

```
enter the x-axis discontinuity point (default 0) :

enter the saturation levels [Y1 Y2] : [-1 1]
```

At this point a medium-size figure with the input-output characteristic of the nonlinear element is shown, while the message:

```
**** press return to continue}
```

is displayed again in the Command Window.

When the return key is pressed, we obtain for a while the message:

```
**** computing ... please wait
```

and the simulation result is shown in the current graphic window. It appears as a time response plot with the reference input to the summing junction, the output of the plant, the input of the nonlinear element and the output of the nonlinear element (see Fig. 3.36).

On pressing the return key again, the following menu appears:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - plot reference input and output of the plant
4 - plot input and output of the nonlinear element
5 - plot everything
6 - plot again with no changes
7 - change colors
```

Besides standard change of axes and grid, this menu enables restriction of the number of plotted functions and change of colors.

Fig. 3.37 refers to another example, where the regulator and the plant are the same as before, while the reference signal is the sine function with $\mathcal{L}$-transform

$$sine(s) = \frac{0.5}{(s^2 + 0.01)} \ ,$$

and the option 6 is entered for the nonlinear element (relay with dead zone). In this case, the corresponding interactive request is:

```
enter the x discontinuity points [X1 X2] : [-.1 .1]

enter the three output levels [Y1 Y2 Y3] : [-1 0 1]

**** press return to continue
```

while the subsequent part of the operation is the same as in the previous case.

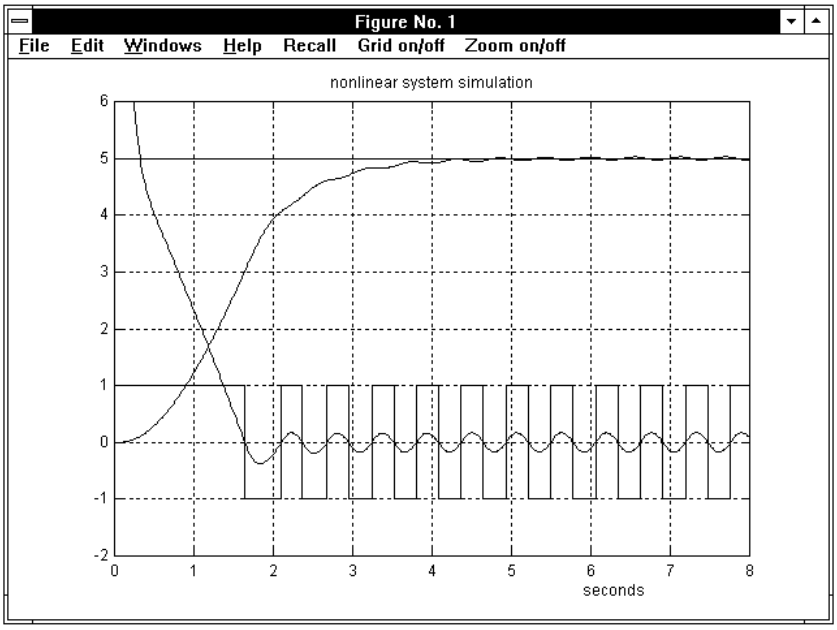

Figure 3.36. The step response of a system with an ideal relay.

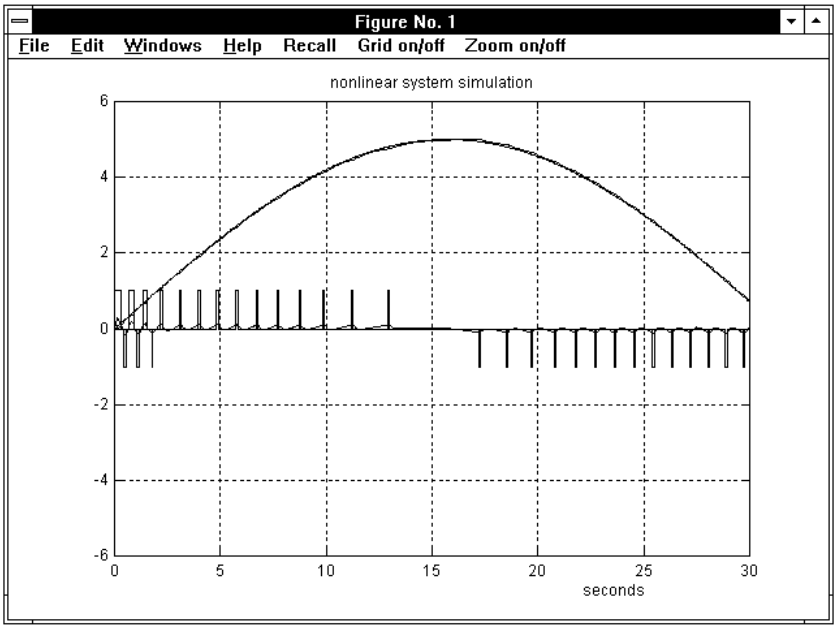

Figure 3.37. The sine response in the case of a relay with dead zone.

3.14 Perftra

The command

```
> perftra,gi,gj,gk,gw   (enter)
```

provides a complete design procedure, based on preview and preaction, for a perfect tracking feedforward compensating device for digital feedback control systems; $gi(z)$ is the transfer function of the regulator (given), $gj(z)$ that of the plant (given), while $gk(z)$ and $gw(z)$ denote those of the two units that form the compensator (to be determined). The first of these units is a pure delay, representing the necessary preview time of the reference to be tracked, while the second one is a feedforward unit to the input of the plant.

3.14.1 Recall

Fig. 3.38 represents the compensator structure considered. The purpose of *perftra* is to derive the delay $G_k(z)$ and the feedforward compensator $G_w(z)$ that realize almost perfect tracking at the output y of a delayed reference r by feeding a suitable *preaction* signal u_c at the input of the plant.

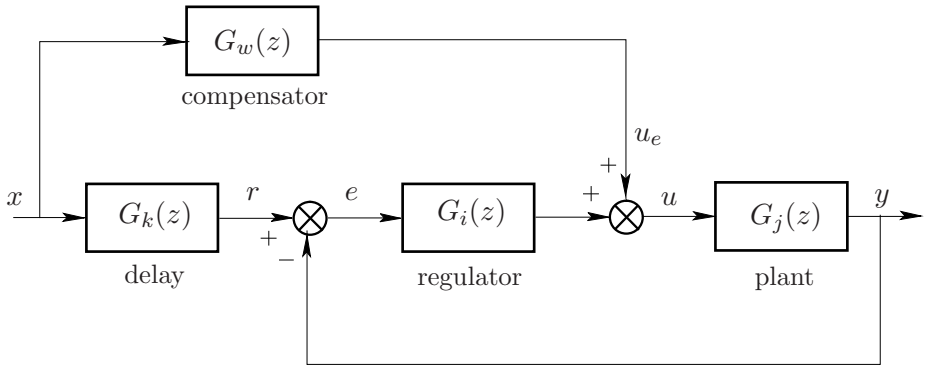

Figure 3.38. The connection of the feedforward compensating device.

Let the transfer function of the plant be

$$G_j(z) = \frac{P(z)}{z^k\, Q(z)} \ ,$$

where $P(z)$ is assumed to have the same degree as $Q(z)$. This is obtained by adding a suitable number of zeros and poles at the origin and including the added poles in the factor z^k. The value of k is called the *plant relative degree*.

Should the plant be invertible, i.e., having all the zeros with absolute value strictly less than one, perfect tracking could be obtained by simply assuming, in the block diagram in Fig. 3.38,

$$G_k(z) = \frac{1}{z^k} \ , \quad G_w(z) = \frac{Q(z)}{P(z)} \ .$$

Hence, in this particular case, the preview and preaction times are of as many samples as the plant relative degree is, while the compensator transfer function simply concides with the inverse of the zero-relative-degree factor $P(z)/Q(z)$ of the plant transfer function.

Unfortunately, in most cases the plant is not invertible. Unstable zeros are very likely to appear in the discrete-time transfer function of the plant, due to the sampling process with zero or first order hold equivalence, even if the corresponding continuous-time model is minimum-phase. The design steps of the delay and compensator when the plant has some zeros with absolute value greater than one can be better understood by making reference to the block diagram in Fig. 3.39. The only basic assumption introduced is that the plant has no zeros with absolute value strictly equal to one.

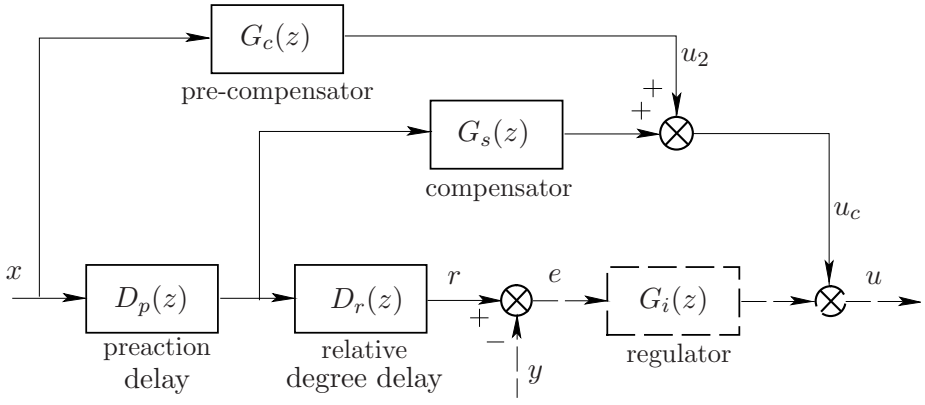

Figure 3.39. A more detailed block diagram of the compensator.

Let $P(z)$ be monic (this assumption does not affect generality, but simply implies that $Q(z)$ generally is not), so that $P(z)$ can be univocally factorized as $P(z) = P^+(z)\,P^-(z)$, with $P^-(z)$ and $P^+(z)$ having all the roots with absolute value less and greater than one, respectively.

Consider the decomposition

$$\frac{Q(z)}{P(z)} = \frac{A(z)}{P^-(z)} + \frac{B(z)}{P^+(z)} \ , \tag{3.7}$$

that can be obtained by applying the partial fraction expansion to the strictly proper ratio of polynomial $Q(z)/(z\,P(z))$, then by multiplying both members of the equality so obtained by z, and by suitably partitioning the terms on the right. Note that $B(z)$ always has a zero root, i.e. a polynomial $B_1(z)$ exists such that $B(z) = z\,B_1(z)$, because the factor z introduced at the second step of the above construction cannot cancel with a corresponding factor in the denominator (all the zero roots of $P(z)$ join to $P^-(z)$ by assumption).

Referring to Fig. 3.39, the relative-degree delay and the compensator are defined as

$$D_r(z) = \frac{1}{z^k} \ , \quad G_s(z) = \frac{A(z)}{P^-(z)} \ , \tag{3.8}$$

while the action of the second term on the right of (3.7) is substituted by an equivalent action obtained with the preaction delay and the pre-compensator.

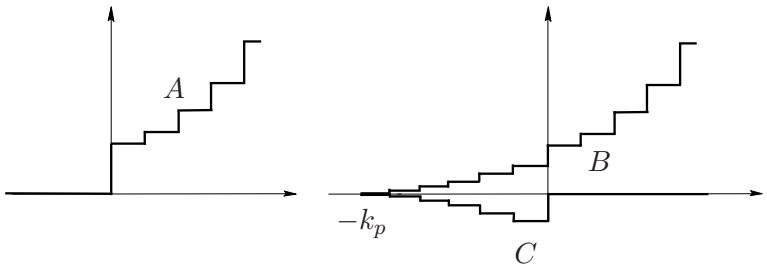

Figure 3.40. How the preaction signal is obtained.

In fact, the inverse $\mathcal{Z}$-transform of $B(z)/P^+(z)$, being strictly unstable, appears as shown in Fig. 3.40, diagram A. It is the signal that, together with the inverse $\mathcal{Z}$-transform of $A(z)/P^-(z)$, should be applied at the input of the plant in advance by the plant relative degree to obtain perfect tracking of a unit impulse at $k=0$. Of course, this is not possible because of saturation.

Note that, since

$$\frac{B(z)}{P^+(z)}\,\frac{P(z)}{z^k\,Q(z)} = \frac{z\,B_1(z)\,P^-(z)}{z^k\,Q(z)}\ , \tag{3.9}$$

lack of application of this signal is equivalent to imposing an initial state on the plant, given by the opposite of the term on the right of (3.9). In fact, this is a linear combination of the modes of the plant. Imagine extending backwards in time the signal A, thus obtaining that denoted by B in Fig. 3.40, that begins at a negative instant of time $-k_p$. Note that A is the sum of B and C, that is defined as the opposite of B for negative time. Lack of application of B is also equivalent to imposing an initial state on the plant, but very small if k_p is large enough, hence resulting in a negligible transient in the overall feedback system, that is stable by assumption.

Hence, signal C is the preaction to be applied for almost perfect tracking of a unit impulse. Of course, the preaction for a generic input signal is its convolution with C. When a suitable value of k_p has been selected, the preaction delay and the pre-compensator are defined as

$$D_p(z) = \frac{1}{z^{k_p}}\ , \quad G_u(z) := \frac{g(k_p)\,z^{k_p-1} + \ldots + g(2)\,z + g(1)}{z^{k_p-1}}\ , \tag{3.10}$$

where $g(k)$ is the inverse $\mathcal{Z}$-transform of the polynomial ratio obtained by replacing z with z^{-1} in $B(z)/P^+(z)$, that concides with signal C in Fig. 3.40, considered in the reverse time direction. Relations (3.8) and (3.10) define all the transfer functions of the feedforward units shown in Fig. 3.39. The transfer functions $G_d(z)$ and $G_c(z)$ of the equivalent blocks in Fig. 3.38 are provided by the straightforward manipulations

$$\begin{aligned} G_k(z) &= D_p(z)\,D_r(z)\ , \\ G_w(z) &= G_u(z) + D_p(z)\,G_s(z)\ . \end{aligned}$$

An estimate of the error reduction provided by preaction can be obtained as follows. Let ρ be the absolute value of the root of $P^+(z)$ closest to the unit circle. The error reduction factor is approximately given by the relation

$$m(k_p) = \rho^{-k_p}\ , \tag{3.11}$$

that can be used in design to select a first-trial value of k_p.

How to preserve robustness of asymptotic tracking

It is well-known that tracking a step or a ramp is asymptotically perfect if the transfer function of the feedback loop has a pole equal to one or a double pole equal to one, and that asymptotic perfect tracking is maintained under parameter variations provided stability is preserved.

In some cases, particularly in motion control, one of the poles equal to one that ensure asymptotically robust tracking belongs to the transfer function of the plant, and steps or ramps are asymptotically tracked if the regulator is type 0 or type 1, respectively. To preserve this property, it is necessary for the correcting signal u_c generated by the compensators to tend to zero when the reference input is a step, or to a constant when it is a ramp. This can be obtained with a simple contrivance: feeding the feedforward compensator and pre-compensator with the first difference of the reference input, whose $\mathcal{Z}$-transform is

$$\Delta R(z) = \frac{z-1}{z}\, R(z) \ ,$$

and, at the same time, multiplying both members of (3.7) by $z/(z-1)$ in the compensator design, so that the unit root of $Q(z)$ is cancelled. Of course, this method can be extended, and differences of order greater than one can be used for correction if the unit pole of the plant is multiple.

Using a single feedforward unit

Let us assume that the transfer function $G_i(z)$ of the regulator is minimum-phase and consider the block diagram in Fig. 3.41. The transfer function $G_f(z)$ is computed as

$$G_f(z) = G_k(z) + G_w(z)\, G_i^{-1}(z) \ . \tag{3.12}$$

Should the relative degree of the transfer function obtained be negative, some poles at the origin could be added. This simply increases the preaction time.

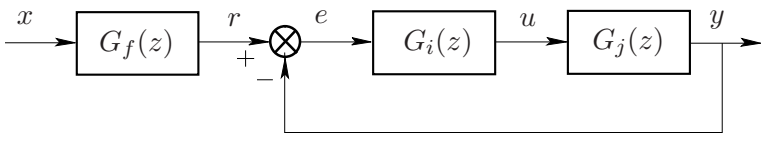

Figure 3.41. A single feedforward unit.

3.14.2 Operation and Examples

Let us consider the nonminimum-phase continuous-time system

$$g(s) = \frac{-3\,(s-2)\,(s+4)}{(s+1)\,(s+2)\,(s+8)}\ ,$$

whose zero-order hold equivalent with sampling time $T = 0.2$ sec is

$$gj(z) = \frac{-.2452\,(z-.454)\,(z-1.534)}{(z-.2019)\,(z-.6703)\,(z-.8187)}\ ,$$

and the type 2 regulator

$$gi(z) = \frac{(z-.9)^2 + .05^2}{(z-1)^2}\ .$$

The command "perftra,gi,gj,gk,gw" displays a medium-size figure with a block diagram showing the connection referred to, and the message:

```
**** press return to continue
```

When the return key is pressed, we have the request:

```
information on design of perfect tracking compensators ? (1) :
```

Entering 1 provides a short description of the procedure used to derive the preaction feedforward compensator, that is not repeated here since it is reported in the previous *Recall* section. If the request is skipped by simply pressing the return key, the transfer functions of the plant and regulator are displayed in the Command Window, followed by information on the possibility of using a single feedforward unit and on the order of the difference used at the compensator input to avoid steady state error. Pressing the return key again produces the screen layout shown in Fig. 3.42, with a medium-size figure showing a logarithmic diagram corresponding to relation (3.11), i.e., giving an estimate of error reduction versus preaction time.

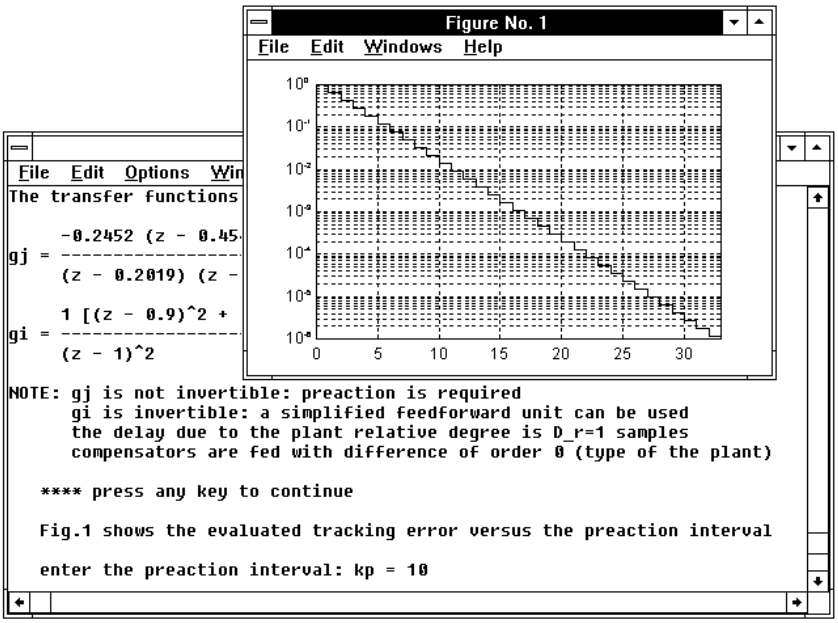

Figure 3.42. The interaction for the choice of the preaction time.

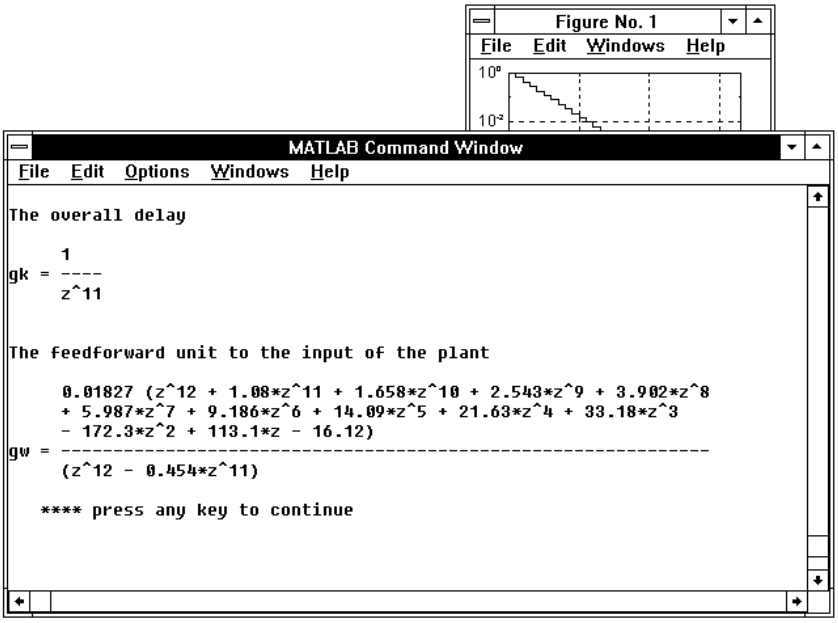

Figure 3.43. The screen layout with the final result.

In the meanwhile the interactive request:

```
enter the preaction interval: kp =
```

is displayed in the Command Window. Let us enter the value 10, corresponding to an error reduction of about two decades. The transfer functions $D_r(z)$, $D_p(z)$, $G_s(z)$ and $G_u(z)$ of the blocks shown in Fig. 3.39 are immediately computed and displayed. On pressing the return key, the transfer functions $G_k(z)$ and $G_w(z)$ of the block diagram in Fig. 3.38 are computed, displayed in the Command Window as shown in Fig. 3.43, and saved in the hard disk with the names *gk* and *gw* specified in the call list.

Since in this case the transfer function of the regulator is invertible, we obtain the request:

```
do you want to use a single feedforward unit ? (1) :
```

and, on entering 1, we have:

```
enter a name for the feedforward transfer function :
```

Let us enter the name *gf*. The transfer function is computed according to (3.12), displayed in the Command Window and saved in the hard disk.

The efficiency of the feedforward perfect tracking compensation can be checked with the system response to a standard input function. In fact, we have the request:

```
do you want to see the system behavior ? (1) :
```

and, on entering 1, a profile is shown, and on pressing the return key, we obtain the further request:

```
do you want to change the parameter of the profile ? (1) :
```

making it possible to enable an optional interactive session to adapt the profile to the case in hand. After this, the menu shown on the next page is displayed. By entering 1 and 10 we obtain the plots shown in Fig. 3.44 (reference input and output of the plant without any correction), by entering 11 and 7 those in Fig. 3.45 (output of the plant with the correction and the feedforward signal injected at the input of the plant), while entering 12 and 13 produces the plots in Fig. 3.46 (tracking error without and with correction). The tracking error is reduced by the ratio 50:1.

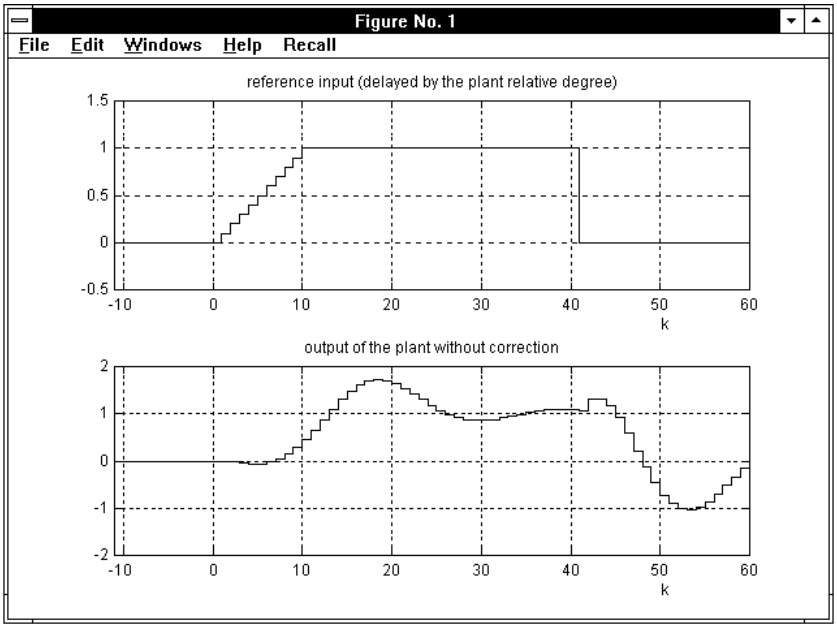

Figure 3.44. Checking the system behavior without and with the compensator.

The final menu for plots appears as:

```
 THE AVAILABLE PLOTS :

 1 - reference input (delayed by the plant relative degree)
 2 - preaction for an impulse of the selected difference
 3 - postaction for an impulse of the selected difference
 4 - correction for an impulse of the selected difference
 5 - preaction for the whole reference
 6 - postaction for the whole reference
 7 - correction for the whole reference
 8 - input of the plant without correction
 9 - input of the plant with correction
10 - output of the plant without correction
11 - output of the plant with correction
12 - tracking error without correction
13 - tracking error with correction
14 - define a new graphic window

enter your choice (press return to exit) :
```

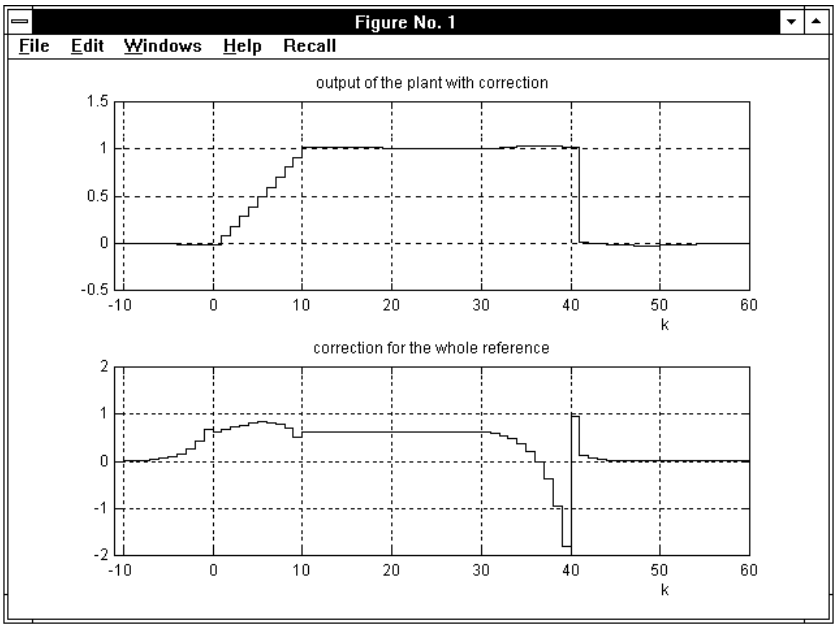

Figure 3.45. Checking the system behavior without and with the compensator.

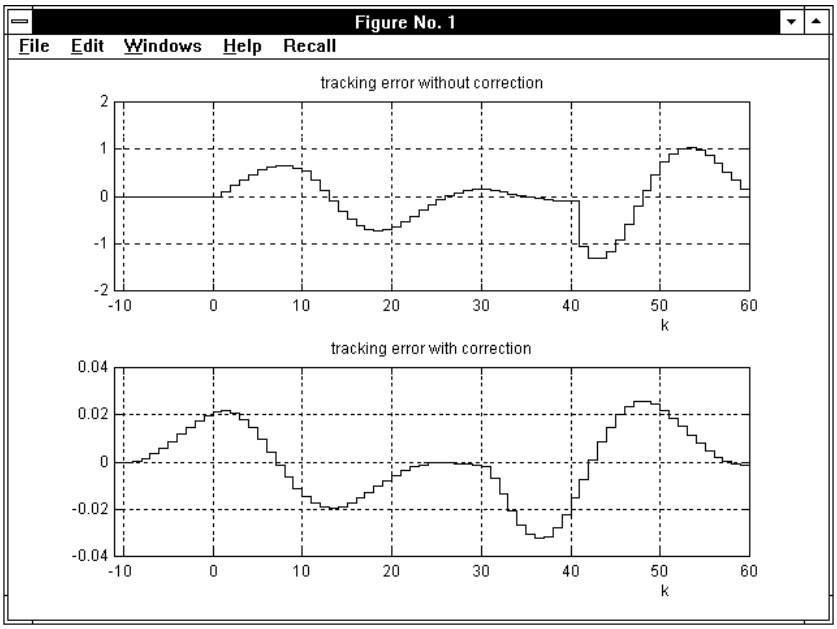

Figure 3.46. Checking the system behavior without and with the compensator.

3.15 Pidc

The command

`> pidc,gi,gj` *(enter)*

provides trial-and-error design of a PID (proportional-integral-derivative) regulator $gj(s)$ for the plant $gi(s)$ by using the Bode diagrams. See the application *pidnich* for design with the Nichols diagram.

3.15.1 Recall

The transfer function of a standard PID regulator is

$$G_j(s) = K_c \left(1 + T_d\, s + \frac{1}{T_i\, s}\right) = K_c\, \frac{T_i\, T_d\, s^2 + T_i\, s + 1}{T_i\, s} \ ,$$

where K_c is the *gain* or *proportional sensitivity*, T_d the *time constant of the derivative action* and T_i the *time constant of the integral action*. Significant parameters for the design are

$$\omega_0 = \frac{1}{\sqrt{T_i\, T_d}} \quad \text{and} \quad \rho = \frac{T_i}{T_d} \ ,$$

that are called the *mid-band frequency* and the *time constant ratio*.

Very often the derivative action is implemented in the feedback path only, to avoid peaks in the closed-loop step response. This is equivalent to splitting the regulator into two units, i.e., a *PI feedforward unit* with transfer function

$$G_{jd}(s) = K_c \left(1 + \frac{1}{T_i\, s}\right) = K_c\, \frac{T_i\, s + 1}{T_i\, s} \ ,$$

and a *feedback unit* with transfer function

$$G_{jf}(s) = \frac{T_i\, T_d\, s^2 + T_i\, s + 1}{T_i\, s + 1} \ ,$$

whose product gives the original PID transfer function.

In general, A PID regulator is not advisable if the control system is type 1, thus having a built-in integral action. In this case, after an interactive *ok* from the user, the design is automatically referred to a *PD regulator* and the above transfer functions are replaced by

$$G_j(s) = K_c\,(1 + T_d\, s)\,, \qquad G_{jd}(s) = K_c\,, \qquad G_{jf}(s) = 1 + T_d\, s \ .$$

The program *pidc* can also be used for the design of PI regulators. In fact, dividing by s the plant transfer function $gi(s)$ before entering produces the design of a PD regulator with transfer function $K_c\,(1+T_d\,s)$. It is easily seen that this is equivalent to a PI regulator for the original plant $gi(s)$ defined by $K_c' = K_c\,T_d$ and $T_i' = T_d$.

The program uses the following design procedure:

1. the phase margin φ_m and the corresponding angular frequency ω_m of the controlled system are computed and displayed;

2. the required phase margin φ_d is entered by the user, and the maximum angular frequency corresponding to phase $-180+\varphi_d$ degrees in the frequency response of the controlled system is derived;

3. this is assumed as the mid-band frequency of the regulator, while in the first trial the value of $\rho = T_i/T_d$ is set to 4, hence both T_d and T_i are determined, and the regulator transfer function determined apart from K_c (also in case of a PD regulator);

4. the desired phase margin φ_d for the overall system is obtained with a suitable choice of K_c; the step responses of the overall system and the regulator output are also plotted;

5. if these responses are unsatisfactory, the design procedure can be repeated from step 3 with different values of T_i and T_d (only T_d if the controlled system is type 1), interactively specified by the user.

Let us point out that in the above-outlined synthesis procedure the value of the phase margin that is entered at the beginning not only influences the values of T_d and T_i automatically selected for the first trial, but is really imposed at step 4. It follows that it is necessary to enter *pidc* again with a different phase margin if the result remains unsatisfactory after several trials with different values of these time constants, unlike *leadc*, where the result is independent of the phase margin initially entered, since the maximum obtainable phase margin is selected at each trial.

Use of the program *pidc* is restricted to type 0 or 1 systems, since systems of type 2 or more are uncommon in industrial applications of adjustable-parameters regulators. If this restriction is not taken into account we obtain the message:

```
**** program pidc applies only to type 0 or 1 systems
```

and the program is quitted.

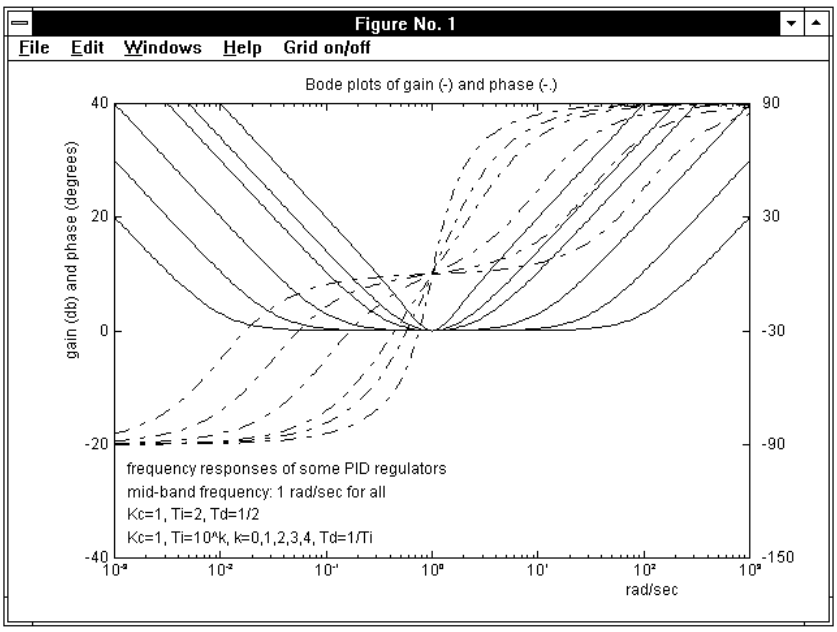

Figure 3.47. The Bode diagrams of some PID regulators.

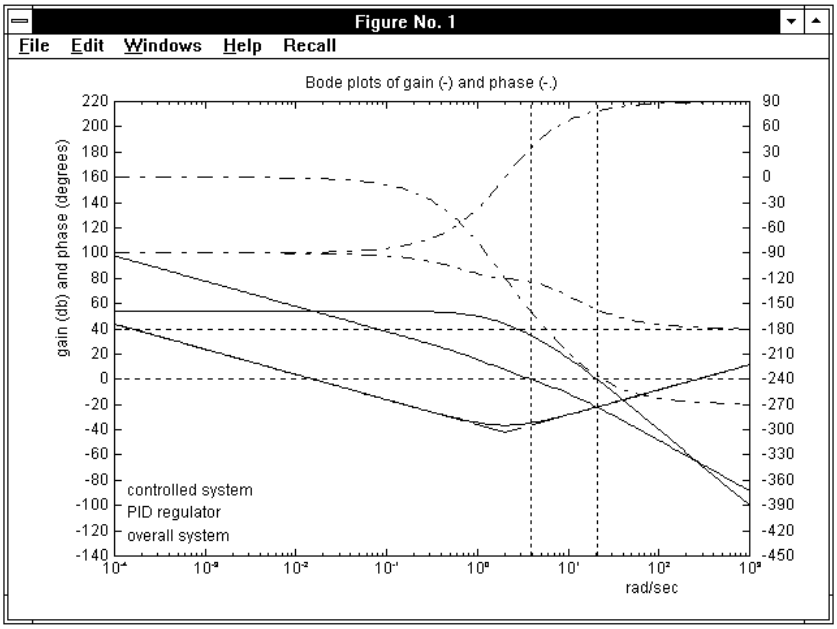

Figure 3.48. The Bode diagrams at the end of the design procedure.

3.15.2 Operation and Examples

Let us consider the transfer function

$$gi(s) = \frac{10000}{(s+1)(s+2)(s+10)} \ .$$

The command "pidc,gi,gj" displays a medium-size block diagram giving information about the connection referred to, and the message:

```
**** press return to continue
```

When the return key is pressed, we have the request:

```
information on design of PID regulators ? (1) :
```

Entering 1 provides a short description of the procedure used for the PID regulator design, that is not reported here since it is contained in the above *Recall* section. This optional information also generates the full-size figure with the Bode diagrams shown in Fig. 3.47.

If the request is skipped by simply pressing the return key, the Bode diagrams of $gi(s)$ are shown with the reference lines pointing out the gain margin (blue) and phase margin (red). The numerical values of gain and phase margins are also displayed inside the diagram frame. On pressing return the Command Window is recovered and the following display appears:

```
phase margin without regulator : -56.04 degrees
                  at frequency : 20.78 rad/sec

enter the required phase margin :
```

Let us enter the value 55 for the first set of trials. First, the Bode plots of the uncompensated system are shown again in green. When the return key is pressed, the Bode plots of the regulator derived are shown in cyan and eventually, on pressing return again, the plots of the corrected overall systems are shown in magenta, with a vertical magenta dotted line, pointing out the phase margin obtained. At this point the screen appears as in Fig. 3.48. When the return key is pressed to exit the Bode diagrams so obtained, the corrected system step response is shown to evaluate the transient behavior with the regulator derived. The output of the regulator is also plotted subsequently to show the transient behavior of the manipulated variable.

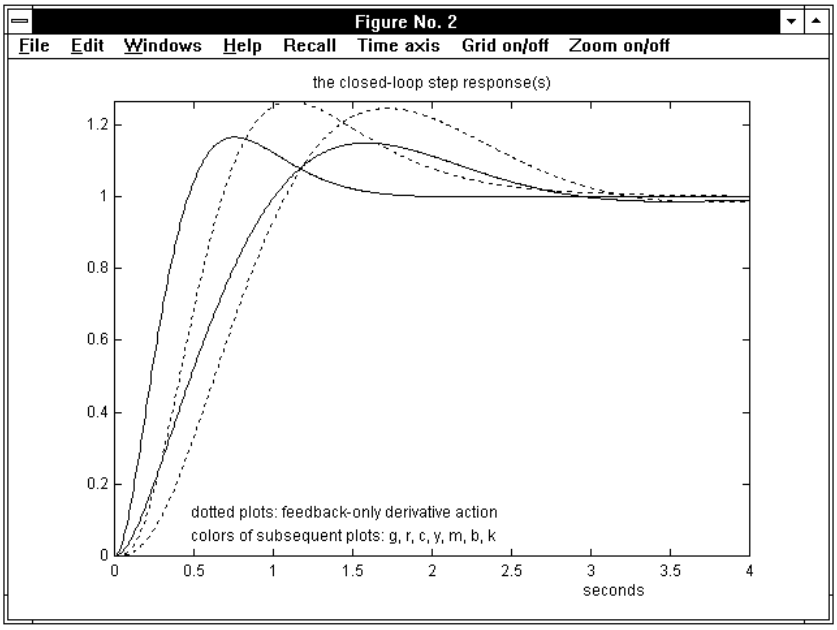

Figure 3.49. The closed-loop step responses.

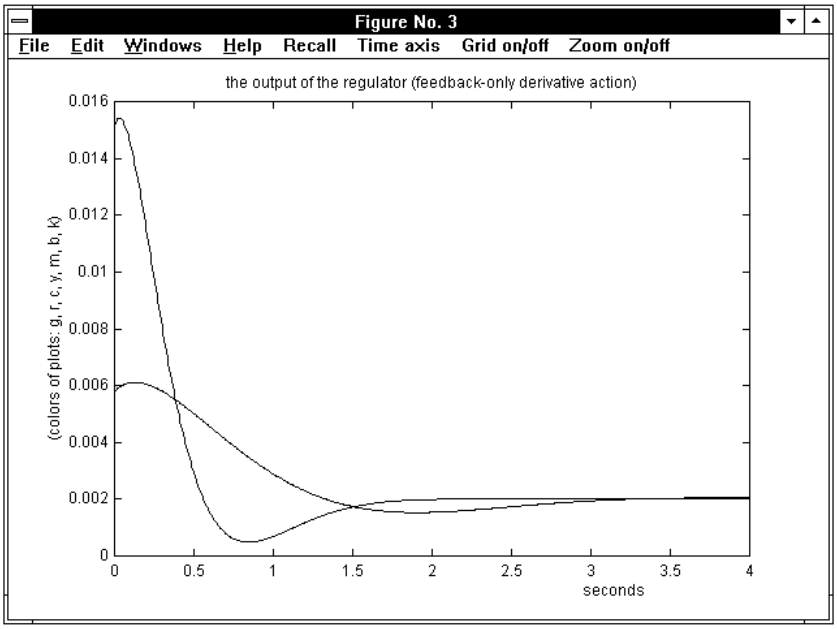

Figure 3.50. The regulator outputs.

When the return key is pressed again, we obtain, in the Command Window:

```
reference color: g ; transfer function of the plant: gi

the regulator derived:

Kc = 0.01519
Ti = 0.9909
Td = 0.2477

do you want to change the regulator ? (1/0) :
```

We assume that the step responses obtained are not satisfactory and decide to proceed with a second trial, by entering 1. As a result of this choice we obtain:

```
you may change the values of Ti and/or Td
Kc will be automatically set to obtain the required phase margin

enter a new Ti or press return to confirm
Ti = 1
enter a new Td or press return to confirm
Td = .2
```

The construction is repeated and the corresponding Bode diagrams shown. The step responses of the subsequent trials are all shown together in different colors according to the sequence: *g, r, c, y, m, b, w* or *k* (the maximum number of trials is seven), so that the effects of the parameter changes are easily noted. The step responses for the two trials performed so far are shown in Figs. 3.49 and 3.50. When the Command Window is recovered again with the return key, we obtain:

```
reference color: r ; transfer function of the plant: gi

the regulator derived:

Kc = 0.005797
Ti = 1
Td = 0.2

do you want to change the regulator ? (1/0) :
```

Since the maximum overshoot of the step response is satisfactory in this case, we decide to exit. On entering 0, we obtain:

```
select function by entering color :
```

On entering *r* (the color of the second trial step response plots), we obtain:

```
THE REGULATOR OBTAINED :

Kc = 0.005797 , Ti = 1 , Td = 0.2

     0.001159 (s^2 + 5*s + 5)
gj = ------------------------
     s

feedback-only derivative action ? (1) :
```

The above request allows factorization of the regulator transfer function into a feedforward part without the derivative action and a feedback part in order to avoid peaks due to abrupt changes of the reference input. On entering 1, the following appears:

```
enter a name for the feedforward transfer function : gf

enter a name for the feedback transfer function    : gb
```

Then, a block diagram with the regulator split is shown in medium size and the following final result is displayed:

```
     0.005797 (s + 1)
gf = ----------------
     s

     0.2 (s + 1.382) (s + 3.618)
gb = ---------------------------
     (s + 1)
```

NOTES:

- As mentioned in the previous *Recall* section, during the trial-and-error design procedure the phase margin originally introduced has not been changed. If the results obtained with this phase margin are far from being satisfactory, the overall procedure must be repeated with a different value of the phase margin.

- If *pidc* is applied to a system of type 1, the following message appears:

```
NOTE: the plant is type 1; a PD regulator will be derived
      do you want a PID regulator anyway ? (1) :
```

3.16 Pidd

The command

`> pidd,gi,gj` *(enter)*

provides trial-and-error design of a digital PID (proportional-integral-derivative) regulator $gj(z)$ for the plant $gi(z)$ by using the Bode diagrams.

3.16.1 Recall

The transfer function of a standard digital PID regulator is

$$G_j(z) = K_c \left(1 + \frac{1}{T_i}\,\frac{T\,z}{z-1} + T_d\,\frac{z-1}{T\,z}\right) = K_c\,\frac{T_i\,T_d\,(z-1)^2 + T\,z\,[T_i\,(z-1) + T\,z]}{T_i\,T\,z\,(z-1)}\ ,$$

where K_c is the *gain* or *proportional sensitivity*, T_d the *time constant of the derivative action* and T_i the *time constant of the integral action*, while T denotes the sampling time.

Very often the derivative action is implemented in the feedback path only, to avoid peaks in the closed-loop step response. This is equivalent to splitting the regulator into two units, i.e., a *PI feedforward unit* with transfer function

$$G_{jd}(z) = K_c \left(1 + \frac{1}{T_i}\,\frac{T\,z}{z-1}\right) = K_c\,\frac{T_i\,(z-1) + T\,z}{T_i\,(z-1)}\ ,$$

and a *feedback unit* with transfer function

$$G_{jf}(z) = \frac{T_i\,T_d\,(z-1)^2 + T\,z\,[T_i\,(z-1) + T\,z]}{T\,z\,[T_i\,(z-1) + T\,z]}\ ,$$

whose product gives the original PID transfer function.

In general, A PID regulator is not advisable if the control system is type 1, thus having a built-in integral action. In this case, after an interactive *ok* from the user, the design is automatically referred to a *PD regulator* and the above transfer functions are replaced by

$$G_j(s) = K_c \left(1 + T_d\,\frac{z-1}{T\,z}\right), \qquad G_{jd}(s) = K_c\,, \qquad G_{jf}(s) = 1 + T_d\,\frac{z-1}{T\,z}\ .$$

The program *pidd* can also be used for the design of PI regulators. In fact, multiplying the plant transfer function $gi(z)$ by $Tz/(z-1)$ before entering produces the design of a PD regulator with transfer function $K_c\,[1+T_d\,(z-1)/(Tz)]$. It is easily seen that this is equivalent to a PI regulator for the original plant $gi(z)$ defined by $K'_c = K_c\,T_d$ and $T'_i = T_d$.

The design procedure is exactly the same as that in the program *pidc* (see the corresponding *Recall* section) with both the plant and regulator transformed according the the w-plane equivalence (see the *Recall* section of application *wplane*). The w-plane transfer function of the regulator is easily derived. In fact, from

$$z = \frac{2+wT}{2-wT}$$

it follows that

$$\frac{T\,z}{z-1} = \frac{2+wT}{2\,w}\ ,$$

hence, in the general PID case,

$$\begin{aligned} G_j(w) &= K_c\left(1+\frac{1}{T_i}\,\frac{2+wT}{2\,w}+T_d\,\frac{2\,w}{2+wT}\right) \\ &= K_c\,\frac{(4\,T_i\,T_d+2\,T_i\,T+T^2)\,w^2+4\,(T_i+T)\,w+4}{2\,T_i\,w\,(wT+2)}\ , \end{aligned}$$

while, in the PD case,

$$G_j(w) = K_c\left(1+T_d\,\frac{2\,w}{2+wT}\right) = K_c\,\frac{(2\,T_d+T)\,w+2}{T\,w+2}\ .$$

In the same way as *pidc*, the program *pidd* is restricted to type 0 or 1 systems, since systems of type 2 or more are uncommon in industrial applications of adjustable parameters regulators. When this restriction is not taken into account, the following message appears:

```
**** program pidd applies only to type 0 or 1 systems
```

and the program is quitted.

3.16.2 Operation and Examples

Let us consider the discrete-time transfer function

$$gi(z) = \frac{7.37\,(z + 0.1313)\,(z + 2.099)}{(z - 0.1353)\,(z - 0.6703)\,(z - 0.8187)}\ ,$$

that corresponds to the continuous-time transfer function $gi(s)$ used in the *Operation and Examples* section of the program *pidc* according to the zero-hold equivalence and sampling time $T = .2$ sec. The command "pidd,gi,gj" displays a medium-size block diagram giving information about the connection referred to, and the message:

```
**** press return to continue
```

When the return key is pressed, we have the request:

```
information on design of digital PID regulators ? (1) :
```

Entering 1 provides a short description of the procedure used for the digital PID regulator design, that is not reported here since it is contained in the above *Recall* section.

If the request is skipped by simply pressing the return key, a Bode diagram of $gi(z)$ is displayed with the reference lines pointing out the gain margin (blue) and phase margin (red). The numerical values of the gain and phase margins are also displayed inside the diagram frame. On pressing return the Command Window is recovered and the following display appears:

```
phase margin without regulator : NaN degrees
                  at frequency : NaN rad/sec

enter the required phase margin :
```

The phase margin is non-computable in this case, since the Bode diagram of gain does not intersect the 0 db line. Nevertheless, let us enter the value 60 for the phase margin of the first set of trials.

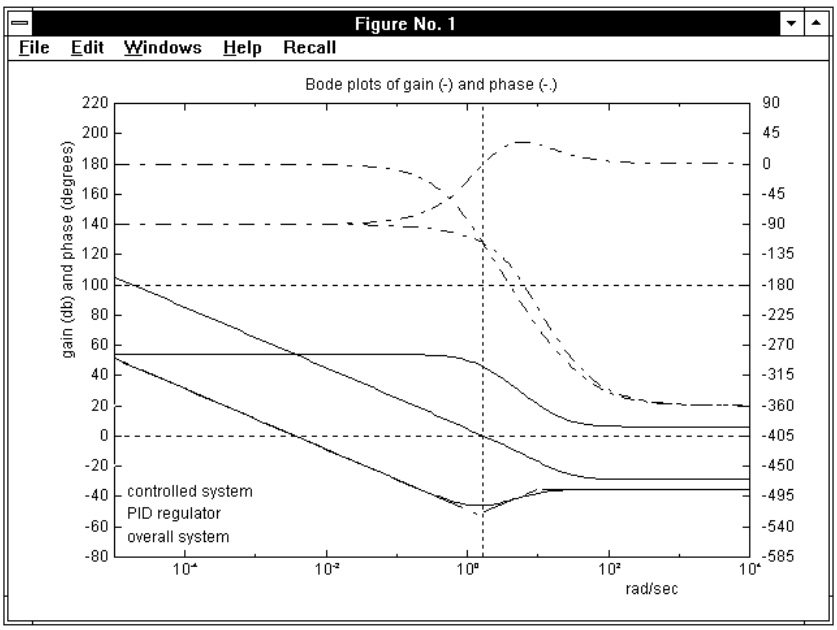

Figure 3.51. The Bode diagrams at the end of the design procedure.

First, the Bode plots of the uncompensated system are shown again in green. When the return key is pressed, the Bode plots of the regulator derived are shown in cyan and eventually, on pressing return again, the plots of the corrected overall systems are shown in magenta, with a vertical magenta dotted line, pointing out the phase margin obtained. At this point the screen appears as in Fig. 3.51.

When the return key is pressed to exit the Bode diagrams so obtained, the corrected system step response is shown to evaluate the transient behavior with the regulator derived.

The output of the regulator is also plotted subsequently to show the transient behavior of the manipulated variable.

When the return key is pressed again, the following display appears on the Command Window:

```
reference color: g ; transfer function of the plant: gi

the regulator derived:

Kc = 0.00436
Ti = 1.205
Td = 0.3013

do you want to change the regulator ? (1/0) :
```

We assume that the step responses obtained are not satisfactory and decide to proceed with a second trial, by entering 1. As a result of this choice we obtain:

```
you may change the values of Ti and/or Td
Kc will be automatically set to obtain the required phase margin

enter a new Ti or press return to confirm
Ti = 1
enter a new Td or press return to confirm
Td = .2
```

The construction is repeated and the corresponding Bode diagrams shown. The step responses of the subsequent trials are all shown together in different colors according to the sequence: *g, r, c, y, m, b, w* or *k* (the maximum number of trials is seven), so that the effects of the parameter changes are easily noted. The step responses for the two trials performed so far are shown in Figs. 3.52 and 3.53. When the Command Window is recovered again with the return key, we obtain:

```
reference color: r ; transfer function of the plant: gi

the regulator derived:

Kc = 0.002575
Ti = 1
Td = 0.2

do you want to change the regulator ? (1/0) :
```

Since the maximum overshoot of the step response is satisfactory in this case, we decide to exit. The following appears:

```
select function by entering color :
```

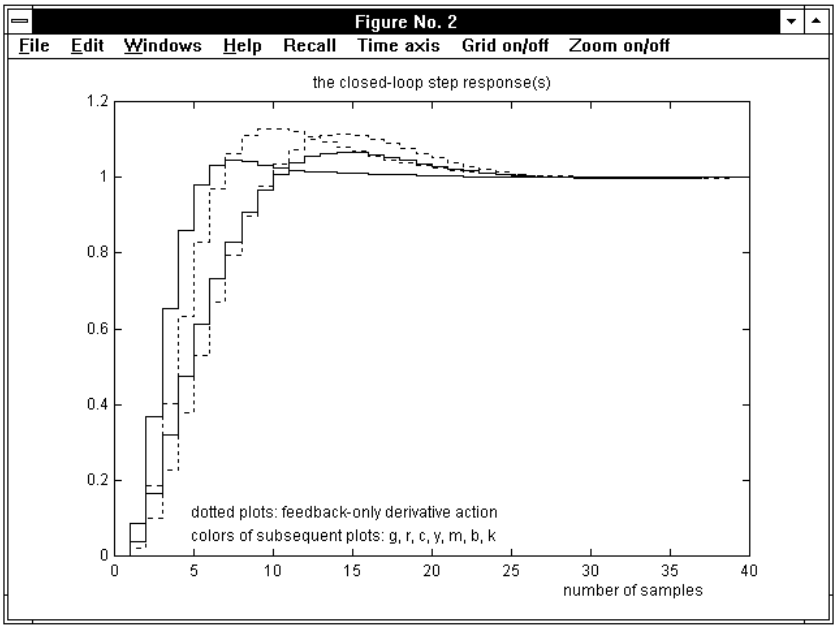

Figure 3.52. The closed-loop step responses.

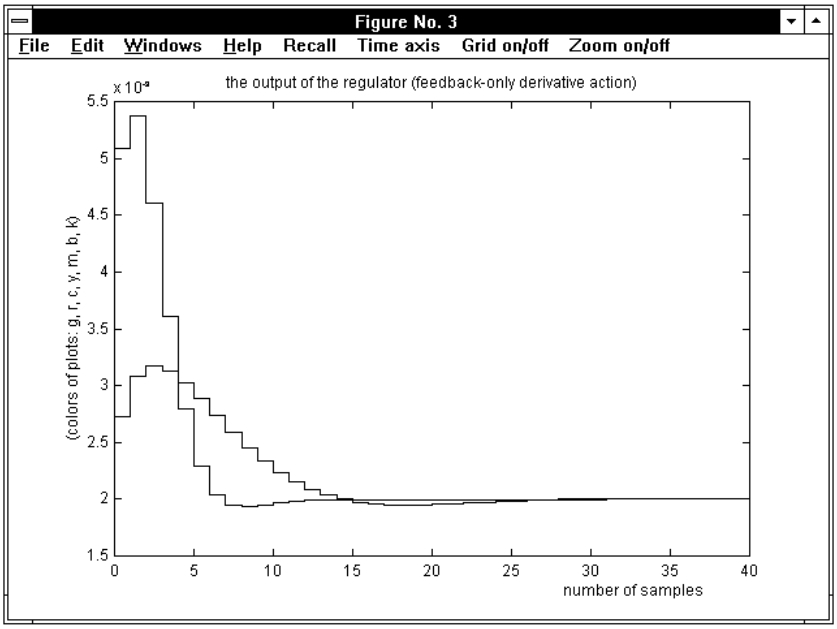

Figure 3.53. The regulator outputs.

On entering *r* (the color of the second trial step response plots), we obtain:

```
THE REGULATOR OBTAINED :

Kc = 0.002275 , Ti = 1 , Td = 0.2

     0.005004 (z - 0.5802) (z - 0.7805)
gj = ----------------------------------
     z (z - 1)

feedback-only derivative action ? (1) :
```

The above request allows factorization of the regulator transfer function into a feedforward part without the derivative action and a feedback part in order to avoid peaks due to abrupt changes of the reference input. On entering 1, the following appears:

```
enter a name for the feedforward transfer function : gf

enter a name for the feedback transfer function    : gb
```

Then, a block diagram with the regulator split is shown in medium size and the following final result is displayed:

```
     0.00273 (z - 0.8333)
gf = --------------------
     (z - 1)

     1.8333 (z - 0.5802) (z - 0.7835)
gb = --------------------------------
     z (z - 0.8333)
```

NOTES:

- During the trial-and-error design procedure the phase margin originally introduced has not been changed. If the results obtained with this phase margin are far from being satisfactory, the overall procedure must be repeated with a different value of the phase margin.

- If *pidd* is applied to a system of type 1, the following message appears:

```
NOTE: the plant is type 1; a PD regulator will be derived
      do you want a PID regulator anyway ? (1) :
```

3.17 Pidnich

The command

`> pidnich,gi,gj` *(enter)*

provides a complete design environment based on the Nichols diagram for PD, PI or PID regulators; $gi(s)$ is the transfer function of the plant (given), and $gj(s)$ that of the regulator (to be determined). See the applications *pidc* and *pidd* for design with the Bode diagrams.

3.17.1 Recall

Let $G_i(s)$ be the transfer function of the plant and $G_j(s)$ that of a regulator to be designed. This can be one of the standard types

$$G_{j,1}(s) = K_c\,(1 + T_d\,s)\,,\quad G_{j,2}(s) = K_c\left(1 + \frac{1}{T_i\,s}\right),\quad G_{j,3} = K_c\left(1 + \frac{1}{T_i\,s} + T_d\,s\right),$$

that are called *PD*, *PI* and *PID* respectively.

The design is performed on the Nichols plot of the frequency response function $G_i(j\omega)$ of the plant. The influence of one of the above regulators on the Nichols diagram is outlined in Fig. 3.54, where n_1 denotes the plot without the regulator and n_2 the plot with the regulator.

The effect of a PD regulator is represented in Fig. 3.54,a. Selection of both a point A (point FROM) on n_1 and a point B (point TO) on the right with respect to A defines a regulator such that n_2 passes through B, and A and B correspond to the same angular frequency on n_1 and n_2 respectively. The vector from A to B is constrained to belong to a suitable domain $\mathcal{D}$, shown in dashed lines in the figure, that is simply a vertical strip having a width of $\pi/2$ radians. Clearly, the figure refers to the domain where the selection of A is possible, provided B has been selected first. A dashed line divides $\mathcal{D}$ into two parts, the upper one corresponding to values of K_c less than 1, and the lower one to values greater than 1.

The selection layout for the PI case is shown in Fig. 3.54,b: however, in this case B is on the left with respect to A; the domain $\mathcal{D}$ is again a $\pi/2$ radians wide vertical strip, but located at the right of the point B, that is usually selected first.

The PID case is shown in Fig. 3.54,c: the strip is π radians wide, with the first-selected point B located on the vertical line on the middle.

To show how the choice of A and B traduces in the regulator parameters, and where the restriction on the corresponding vector has its origin, the *inversion formulae* of the PD, PI and PID regulators are briefly presented.

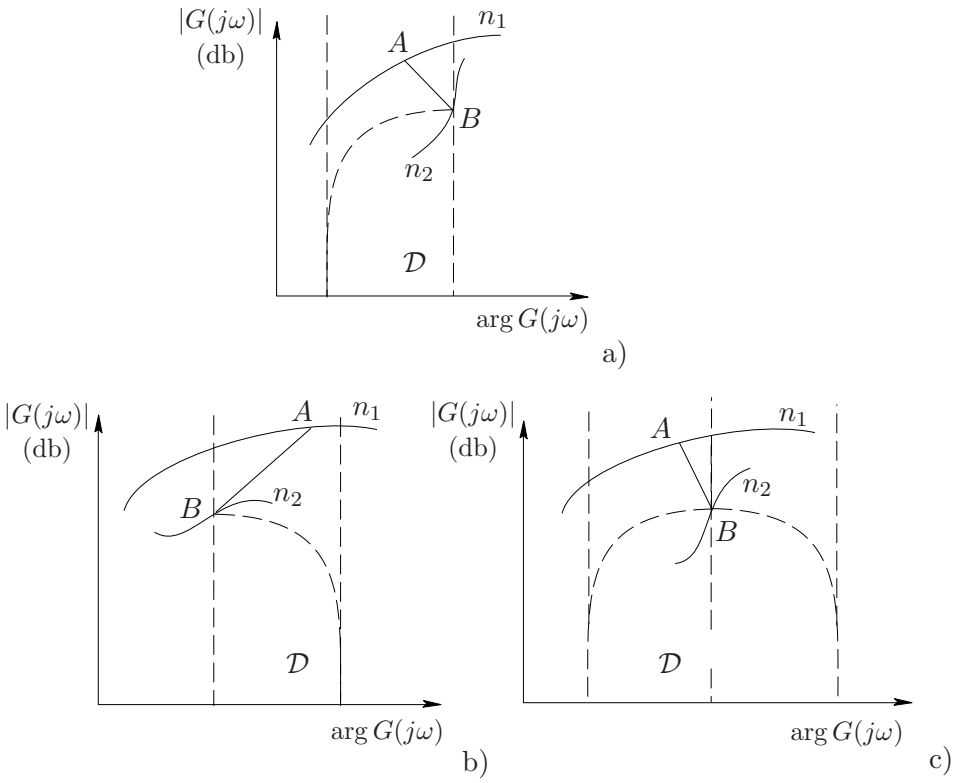

Figure 3.54. The Nichols diagrams for PD, PI, and PID regulators.

Let us define $\Delta y := y_B - y_A$, $\Delta x := x_B - x_A$ and derive M and φ as

$$M = 10^{\Delta y/20} , \qquad \varphi = \Delta x .$$

The inversion formulae for the PD regulator are obtained from the equality

$$M\, e^{j\varphi} = M(\cos\varphi + j\, \sin\varphi) = K_c(1 + j\omega T_d) ,$$

as

$$K_c = M cos\varphi , \qquad \omega T_d = \tan\varphi \ : \tag{3.13}$$

T_d is simply obtained by dividing ωT_d, provided by second relation in (3.13), by the angular frequency ω_0 corresponding to A on n_1.

Likewise, in the PI regulator case, from

$$M\, e^{j\varphi} = K_c\Big(1 + \frac{1}{j\omega T_i}\Big) = K_c \frac{1 + j\omega T_i}{j\omega T_i}$$

we derive the inversion formulae as

$$K_c = M\, \cos\varphi\ , \qquad \omega T_i = -\frac{1}{\tan\varphi}\ : \tag{3.14}$$

T_i is derived from the second relation in (3.14) as in the previous case.

In the PID regulator case we have

$$M\, e^{j\varphi} = K_c\, \frac{-T_i T_d\, \omega^2 + j\omega T_i + 1}{j\omega T_i} = K_c \left(1 - j\, \frac{\frac{T_d}{T_i}(T_i\omega)^2 - 1}{\omega T_i}\right)\ , \tag{3.15}$$

from which we derive

$$K_c \;=\; M\, \cos\varphi \tag{3.16}$$

$$\omega T_i \;=\; \frac{1}{2}\, \frac{T_i}{T_d} \left(\tan\varphi + \sqrt{(\tan\varphi)^2 + 4\frac{T_d}{T_i}}\right)\ . \tag{3.17}$$

Relation (3.17) follows as the positive solution of the second order equation

$$-\frac{T_d}{T_i}(T_i\omega)^2 - \tan\varphi\,(T_i\omega) + 1 = 0\ ,$$

that is easily derived from (3.15). The inversion formulae are (3.17) and (3.17). However, in this case it is also necessary to assign the ratio $\rho := T_i/T_d$, thus having a further degree of freedom in the design. The standard value is $\rho = 4$ (resulting in real coincident zeros), but different values, approximately in the range from 1 to 100, may be used to correct the frequency and/or time response.

3.17.2 Operation and Examples

Let us consider the transfer function

$$gi(s) = \frac{10000}{(s+1)\,(s+2)\,(s+10)\,(s+30)} \ . \tag{3.18}$$

After the command "pidnich,gi,gj", first a medium-size block diagram is shown to inform about the connection referred to and the message:

```
**** press return to continue
```

is displayed. When the return key is pressed, we have the request:

```
information on the design method ? (1) :
```

Entering 1 produces a short description of the procedure used to derive a regulator with the inversion formulae, that is not reported here since it is contained in the above *Recall* section.

If the request is skipped by simply pressing the return key, the Nichols diagram of $gi(j\omega)$ is plotted with graduation versus ω. The reference lines for the gain and phase margins are also shown, while the values of these margins are displayed at the bottom left, as shown in Fig. 3.55. In the figure, the inscription **Regulator No. 1** appears in green. In fact, the program makes it possible to design up to six different regulators, each with plots and messages in a different color, according to the standard sequence: *green*, *red*, *cyan*, *yellow*, *magenta*, and *blue*. The Nichols plot and the messages referring to the plant are shown in black when the figure background is white or in white when it is black.

In the command bar of the figure a **Change axes** menu is provided to adjust the automatic selection, with the items: **y top**, **y bottom** (plus 20 db, minus 20 db), **x left**, **x right** (plus 30 degrees, minus 30 degrees), **standard** (setting the axes as $[-360, 0, -40, 40]$), **M and N loci on/off**. The **Grid on/off** command is also provided. The **standard** option has been used in Fig. 3.55.

At the top left of the figure a pushbutton menu shows the three options for the design, **PD**, **PI** and **PID**, and, initially disabled, the items **Continue**, **Another regulator**, **Step response**, **Bode diagrams**. A further pushbutton **Exit** is provided, at the bottom left corner of the figure, to quit the program.

The small rectangular frames under the Nichols plot are used to see the step responses, both at the output of the plant and regulator, for the two last choices of the point FROM.

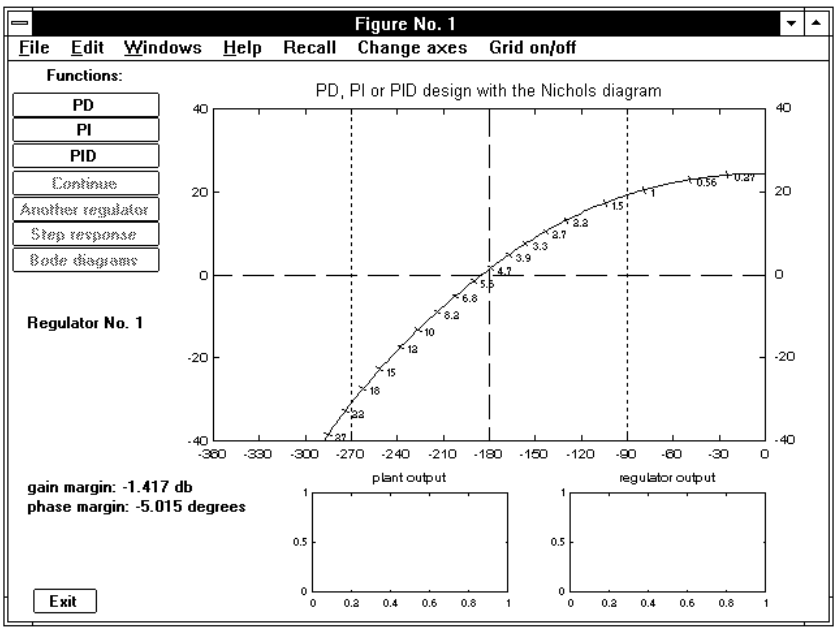

Figure 3.55. The initial figure with the Nichols diagram.

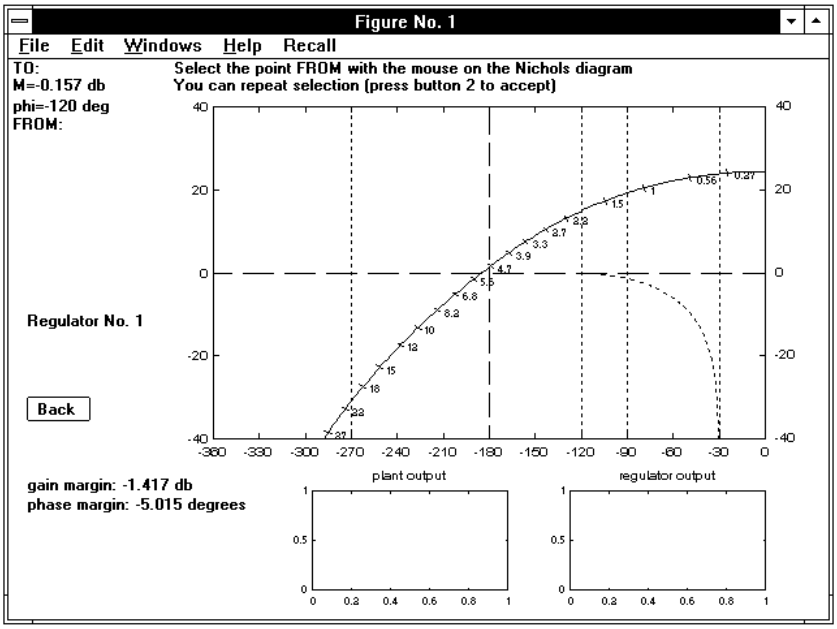

Figure 3.56. The diagram after selection of the point TO.

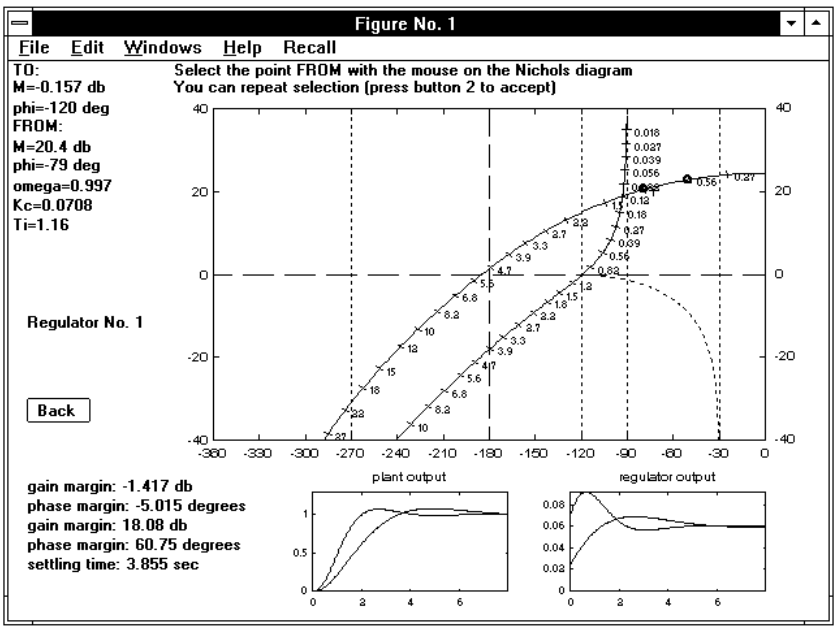

Figure 3.57. The diagram during selection of the point FROM.

Let us decide to design a PI regulator. When clicking the mouse on **PI**, the above menu disappears and the following message appears at the top of the figure:

PI : select the point TO with the mouse (button 1)
You can repeat selection (press button 2 to accept) (a)

At every selection of the point TO with button 1 the corresponding coordinates are displayed at the top left of the figure and the domain $\mathcal{D}$ where it is possible to choose the point FROM is shown. When, possibly after some trials, the point TO is accepted by pressing button 2 of the mouse, message (a) is replaced with:

Select the point FROM with the mouse on the Nichols diagram
You can repeat selection (press button 2 to accept) (b)

At this point the screen appears as shown in Fig. 3.56. The emergency pushbutton **Back** on the left enables the current design to be interrupted and the initial setting to be recovered, thus making it possible to choose a different type of regulator.

At every selection of the point FROM with button 1, the screen appears as shown in Fig. 3.57.

At the top left, the coordinates of the point selected and the corresponding value of the angular frequency, as well as the parameters of the corresponding regulator, are shown under the message referring to the point TO already present. If the point FROM does not belong to the domain $\mathcal{D}$, the message:

SELECTION NOT ACCEPTABLE

appears in red. It can be cancelled by clicking the mouse on the figure.

The Nichols diagram of the system with the regulator appears with complete graduation versus ω. Note that it passes through the point TO as desired. The point FROM is pointed out with a small circle on the original Nichols diagram. The new values of the gain and phase margins are also shown at the bottom left of the figure under those of the uncorrected system. The settling time, computed as $3/\sigma_m$, where σ_m is the minimum absolute value of the real parts of the closed-loop poles, is also shown. It is possible to repeat the selection of the point FROM in the zone allowed, in order to search for the most convenient one. For instance, if the point TO has been selected to impose the phase margin (as in the example in hand), a convenient value for the gain margin or for the settling time may be sought. For the designer's convenience, the step responses referring to the last two choices are shown in the small rectangular frames under the Nichols diagram: the corresponding points FROM and responses are contemporarily shown on the Nichols plot (with small circles) and in rectangular frames, in orange (the previous one) and in the color corresponding to the current regulator (the last one).

If the diagram and the corresponding margins are satisfactory, on pressing button 2 all the messages disappear and the pushbutton menu is shown again with the choices **Continue**, **Another regulator**, **Step response**, **Bode diagrams** enabled. The pushbutton **Exit** is also restored.

We provide here a brief description of the options.

Continue. This option makes it possible to continue the design of a regulator by selecting different points FROM, after checking response(s).

Another regulator. With this option, it is possible to repeat the synthesis procedure from the beginning without entering *pidnich* again, thus enabling comparison of different solutions in terms of the step and/or frequency responses. As previously recalled, the subsequent regulators designed are distinguished by color.

Step response. This option makes it possible to check the effectiveness of the regulator(s) by means of the step response of the overall closed-loop system. When it is entered, the menu is disabled and two new pushbuttons appear with the further choices **Plant output** and **Regulator output**. The first provides the step response of the overall system, while the second makes it possible to see the output of the regulator, that coincides with the input of the plant, in order to check the amount of the transient and orient the design towards avoiding input saturation if necessary. Figs.3.58 and 3.59 show the screen layouts in the two cases, with both a PI and a PID regulator designed for the plant defined by (3.18). The changes in the interactive session for the PID regulator are briefly described below. In the PD and PID cases the response when the derivative action is feedback-only is also shown with a dotted plot in the first figure and as the regular one in the second one: in fact, derivative action in the feedforward path would cause a meaningless Dirac impulse at the output of the regulator in these cases. See the *Recall* section of application *pidc* for more information on feedback-only derivative action. If several regulators have been derived, the corresponding plots are all shown together in different colors.

In the command bar of the figure, the facilities **Recall**, **Time axis** and **Grid on/off** provide information on all the regulators designed, change of time axis, if necessary, and grid. The figure is created with a new number. The pushbutton **Menu** provides recovering the previous figure with the Nichols diagrams and the main pushbutton menu.

Bode diagrams. This option allows the Bode diagrams of the control loop to be checked. When it is entered, the menu is disabled and two new pushbuttons appear with the further choices **Open-loop**, **Closed-loop** and **Sensitivity fnct**. If more than one regulator have been derived, all the Bode diagrams are shown in different colors as in the step response case. In the command bar of the figure, the facilities **Recall** and **Grid on/off** provide information and grid. This figure is also created with a new number. The pushbutton **Menu** is used as before. Fig. 3.60 shows the open-loop Bode diagrams corresponding to the PI and PID considered in the step response case.

Exit. This pushbutton provides exit from *pidnich*. When clicked on, the message:

PRESS RETURN TO EXIT

appears in orange at the bottom left of the screen.

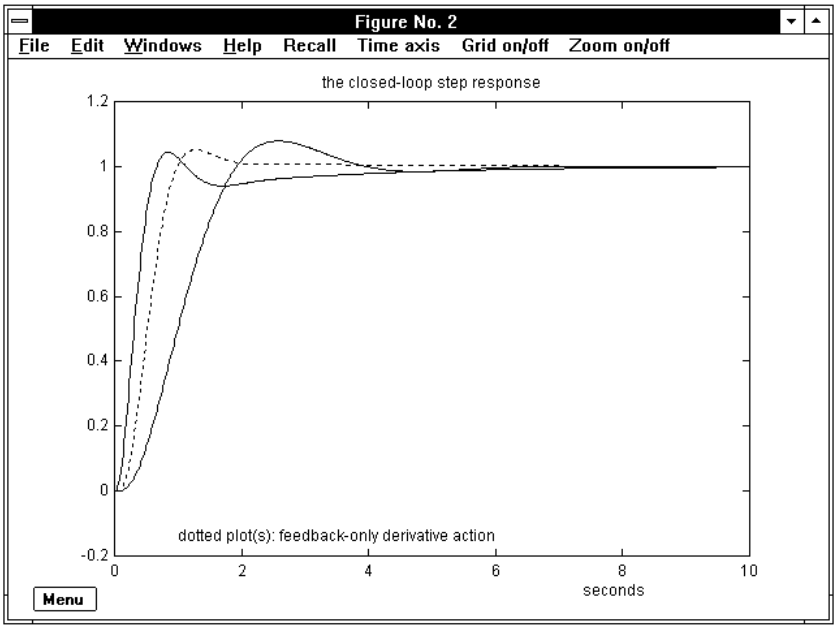

Figure 3.58. The closed-loop step responses.

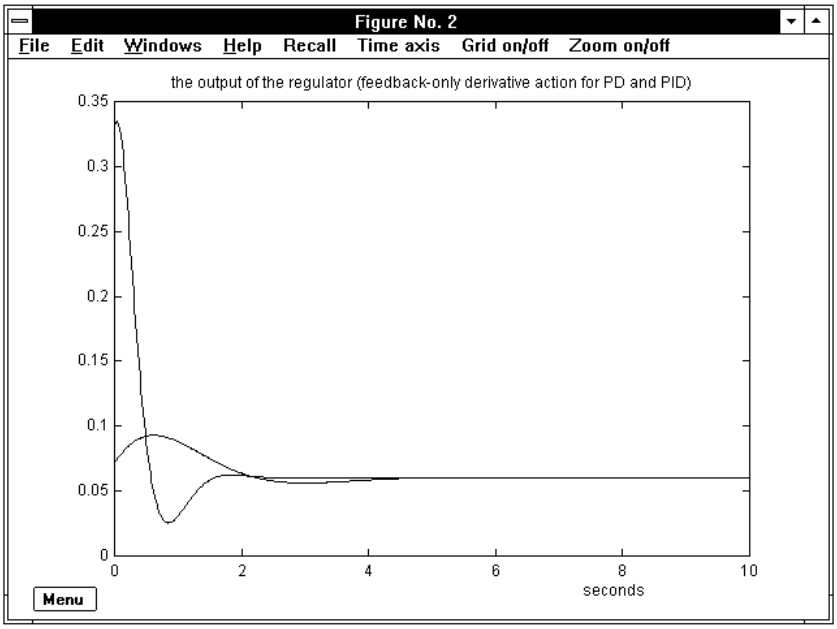

Figure 3.59. The regulator outputs.

On pressing the return key, the Command Window is shown again. If only one regulator has been derived, we obtain the display:

```
THE REGULATOR OBTAINED :

Type PI : Kc = 0.07085, Ti = 1.165 sec

     0.07085 (s + 0.8584)
gj = --------------------
     s
```

and the transfer function *gj* is saved in the hard disk.

If during the *pidnich* session several regulators have been derived and compared, the standard choice by color is requested on exiting the program. To make this choice easier, the figures with the step and frequency responses are preserved and may be easily recalled with the mouse.

In the PD and PID case the request:

```
feedback-only derivative action ? (1) :
```

appears in the Command Window, making it possible to split the regulator into a feedforward and a feedback section, whose transfer function names are interactively defined like in application *pidc*.

What is different in the PID case

When a PID regulator is being designed, in place of message (b), the following appears:

You can select the Td/Ti ratio - default is 4

and a small window with a pop-up menu is temporarily available in the figure for the selection (the possible values are 1, 2, 4, 10, 20, 40, 100). The choice is disabled when selection is done, and the selected value is permanently shown in the same location. Then, the procedure of the previously described PI case applies without any change, starting from message (b). Fig. 64 shows the screen layout after the first choice of the point FROM during the design session of a PID regulator for the plant defined in (3.18). Note that the step responses for the feedback-only derivative action case are also shown, with dotted lines, in the small rectangular frames.

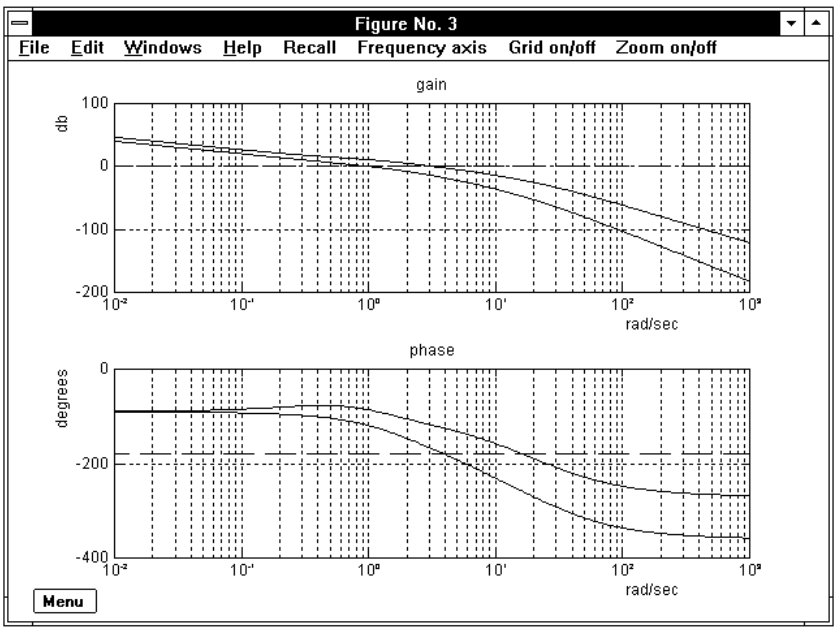

Figure 3.60. The Bode diagrams.

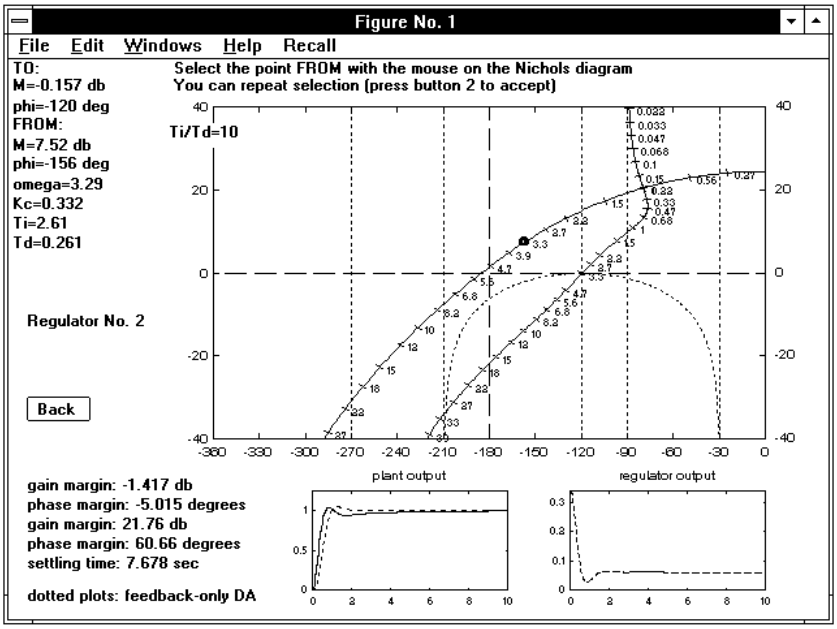

Figure 3.61. Selection of the point FROM in the PID case.

3.18 Regdph

The command

`> regdph,gi,gj[,gk,gw]` *(enter)*

provides a regulator by complete closed-loop pole assignment using the Diophantine equation. The meaning of the call list is the following: $gi(s)$ or $gi(z)$ is the transfer function of the plant, $gj(s)$ or $gj(z)$ that of the assigned part of the regulator, $gk(s)$ or $gk(z)$ a transfer function with the poles to be assigned (zeros are uninfluential), and $gw(s)$ or $gw(z)$ (output) the transfer function of the regulator derived. It is necessary to assign a sufficient number of poles to guarantee causality of the regulator: the two-argument call provides this number. See the application *regrootl* for trial-and-error mouse-oriented design also with the Diophantine equation.

3.18.1 Recall

The Diophantine equation is a convenient means for assignment of the closed-loop poles of a SISO feedback system. It also makes it possible to insert in the regulator any open-loop zero-pole map, typically a simple or multiple pole at the origin or cancellations with the plant.

The direct solution of the Diophantine equation

Let $A(s)$, $B(s)$ and $C(s)$ be any triple of polynomials. The equation

$$A(s)\,X(s) + B(s)\,Y(s) = C(s) \tag{3.19}$$

is a *Diophantine equation* in the unknown polynomials $X(s)$ and $Y(s)$. It is well known that:

1. equation (3.19), if solvable, admits an infinite number of solutions;
2. equation (3.19) is solvable if and only if the greatest common divisor of $A(s)$ and $B(s)$ is a divisor of $C(s)$. Hence, assuming that $A(s)$ and $B(s)$ are coprime does not affect generality.

A particular solution of the Diophantine equation, here called the *direct solution*, is derived through an equivalent set of algebraic linear equations. Let n, m and ℓ be the degrees of $A(s)$, $B(s)$ and $C(s)$ respectively, and suppose that $A(s)$ and $B(s)$ are coprime. The degrees k and h of the unknown polynomials $X(s)$ and $Y(s)$ provided by the direct solution satisfy the relation

$$k \begin{cases} = \ell - n & \text{if } \ell \geq m + n \\ \leq m - 1 & \text{if } \ell < m + n \end{cases}, \quad h \leq n - 1\ . \tag{3.20}$$

The direct solution of equation (3.19) is derived as follows. Let $(a_n, \ldots, a_0)$, $(b_m, \ldots, b_0)$ and $(c_\ell, \ldots, c_0)$ be the coefficients of the given polynomials, and $(x_k, \ldots, x_0)$, $(y_h, \ldots, y_0)$ those of the unknown ones. To simplify notation, we refer to the particular case $n=3$, $m=2$ and $\ell=8$ and derive a pair of unknown polynomials with $k=5$, $h=2$, according to (3.20). In this case the equivalent set of algebraic linear equations that provides the unknowns through simple matrix inversion, is

$$\begin{bmatrix} a_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_2 & a_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_1 & a_2 & a_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_0 & a_1 & a_2 & a_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_0 & a_1 & a_2 & a_3 & 0 & b_2 & 0 & 0 \\ 0 & 0 & a_0 & a_1 & a_2 & a_3 & b_1 & b_2 & 0 \\ 0 & 0 & 0 & a_0 & a_1 & a_2 & b_0 & b_1 & b_2 \\ 0 & 0 & 0 & 0 & a_0 & a_1 & 0 & b_0 & b_1 \\ 0 & 0 & 0 & 0 & 0 & a_0 & 0 & 0 & b_0 \end{bmatrix} \begin{bmatrix} x_5 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \\ x_0 \\ y_2 \\ y_1 \\ y_0 \end{bmatrix} = \begin{bmatrix} c_8 \\ c_7 \\ c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix} . \tag{3.21}$$

If $\ell \geq m+n$, as in this case, the square matrix on the left is $(\ell+1)\times(\ell+1)$, with the coefficients of $A(s)$ repeated in the first $k+1$ columns and progressively shifted downward to form a band submatrix as shown, and the remaining $h+1$ columns with those of $B(s)$ as a down-justified band submatrix. The column vector on the right simply reports the coefficients of $C(s)$.

If $\ell < m+n$, the coefficient matrix on the left is $(m+n)\times(m+n)$, built in the same way as before, while the column vector on the right, whose length is also $m+n$, has some leading zeros if the degree of $C(s)$ is less than $m+n-1$.

In any case the coefficient matrix is a *Sylvester matrix*, nonsingular if and only if $A(s)$ and $B(s)$ are coprime. It clearly follows from (3.21) that the degrees k and h specified in (3.20) with $\leq$ are equalities when $C(s)$ is generic. Let us also note that in the case $\ell \geq m+n$ an increase in ℓ produces a corresponding increase in k only, so that exchanging $A(s)$ with $B(s)$ produces two different solutions.

A very compact Matlab routine for solving the Diophantine equation is *diopha.m*, available in the TFI environment.

The closed-loop pole assignment with the Diophantine equation

Let us consider the feedback system shown in Fig. 3.62, where the transfer function of the plant (given) and that of the regulator (to be determined) are specified as

$$G_i(s) = \frac{P_i(s)}{Q_i(s)}, \qquad G_r(s) = \frac{P_r(s)}{Q_r(s)} = \frac{P'_r(s)\,P_j(s)}{Q'_r(s)\,Q_j(s)} \ . \tag{3.22}$$

We shall denote by m_i and n_i the degrees of $P_i(s)$ and $Q_i(s)$, that are assumed to be coprime, m_r and n_r those of $P_r(s)$ and $Q_r(s)$. For the sake of generality, the transfer function of the regulator is factorized as shown in (refff43), where $P'_r(s)$ and $Q'_r(s)$ are the unknowns, while $P_j(s)$ and $Q_j(s)$, with degrees m_j and n_j respectively, are given factors to be included in the regulator transfer function. A typical use of this possibility is the imposition of a simple or multiple pole at the origin to null the asymptotic tracking error of a step, a ramp or a parabola according to the internal model principle, or of some zeros in the regulator equal to poles of the plant in order to obtain a pole-zero cancellation. We assume $m_i < n_i$.

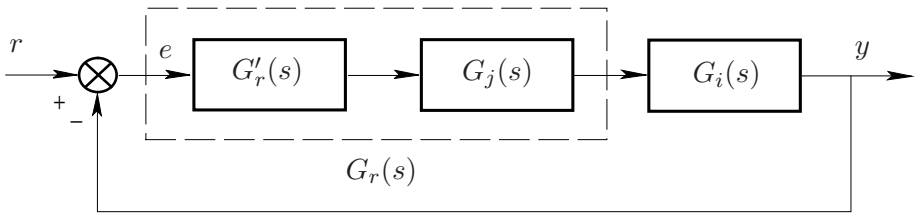

Figure 3.62. The feedback system considered.

Let us first consider the case in which the pairs $P_i(s)$, $Q_j(s)$ and $Q_i(s)$, $P_j(s)$ are coprime. It will be shown below that this assumption can easily be removed if the common roots are stable, and reduction of the minimum number of assignable poles is also obtained. Given any monic polynomial $Q_\ell(s)$ with degree $\ell \geq \ell_0 := 2n_i+m_j+n_j-1$, a regulator $P_r(s)/Q_r(s)$ with $m_r \leq n_r$ exists such that the poles of the feedback system coincide with the roots of $Q_\ell(s)$.

In fact, let us consider the Diophantine equation

$$Q'_r(s)\,(Q_j(s)\,Q_i(s)) + P'_r(s)\,(P_j(s)\,P_i(s)) = Q_\ell(s)\ , \tag{3.23}$$

where $Q'_r(s)$, $P'_r(s)$, with degrees k and h respectively, are the unknown polynomials. The expression on the left is the characteristic polynomial of the feedback system. Equation (3.23) is solvable, since the given polynomials $Q_j(s)\,Q_i(s)$ and $P_j(s)\,P_i(s)$, with degrees n_i+n_j and m_i+m_j respectively, are coprime by assumption. If $\ell=\ell_0$, from (3.20) it follows that k is univocally determined; in fact, the first case in (3.20) holds, since the assumption $m_i<n_i$ implies $\ell_0=2n_i+m_j+n_j-1\geq m_i+m_j+n_i+n_j$. The causality relation $m_r\leq n_r$, with $n_r=k+n_j$ and $m_r=h+m_j$, is satisfied, since $k=\ell_0-n_i-n_j$, $h\leq n_i+n_j-1$, hence $n_r=\ell_0-n_i=n_i+m_j+n_j-1$, $m_r\leq n_i+m_j+n_j-1$.

On the other hand, if $\ell>\ell_0$, the equality $n_r=\ell-n_i$ still being valid, n_r is increased by $\ell-\ell_0$, while m_r still satisfies the inequality $m_r\leq n_i+m_j+n_j-1$. This ensures causality to an even greater extent.

When the pairs $P_i(s)$, $Q_j(s)$ and $Q_i(s)$, $P_j(s)$ have common roots, these, if stable, can be cancelled before solving (3.23). It is well known that regulator and plant cannot have unstable cancellable factors since these would imply unstable poles of the overall feedback system. Since the cancellations of the poles and zeros of the plant reduce the number of the closed-loop poles to be assigned, it may be convenient to delete some or all of the strictly stable poles and zeros of the plant before solving (3.23), and then to insert them as zeros and poles of the obtained regulator: let n_c and m_c be the numbers of the deleted poles and zeros of the plant: it is easily shown that the minimum number of assignable poles compatible with the regulator causality condition is, in this case, $\ell'_0=\ell_0-n_c-m_c$.

In fact, removal of a pole of the plant produces a reduction by 2 of ℓ_0, but the number of the assignable poles is reduced only by 1, since the relative degree of the regulator obtained is one instead of zero, thus making it possible to insert a zero after solution.

On the other hand, removal of a zero of the plant does not affect ℓ_0, but the number of the assignable poles can be reduced by 1, since a regulator with relative degree -1 is permitted by the insertion of a pole after the design.

The program *regdph* automatically provides cancellations and recovering of the common roots of $P_i(s)$, $Q_j(s)$ and $Q_i(s)$, $P_j(s)$.

3.18.2 Operation and Examples

Let

$$gi(s) = \frac{-20\,(s+5)}{(s+1)\,(s-2)\,(s+8)}\ , \quad gj(s) = \frac{1}{s^2}\ ,$$

be the transfer functions of the plant and of the given part of the regulator, respectively. The two-argument call "regdph,gi,gj" simply produces the message:

```
the minimum number of poles to assign for a causal regulator is 7
```

In fact, $\ell_0 = 2 \cdot 3 + 2 - 1 = 7$ in this case. Assume

$$gk(s) = \frac{1}{(s+4)\,(s+6)\,(s+10)\,((s+2)^2+2^2)\,((s+4)^2+3^2)}\ .$$

After the command "regdph,gi,gj,gk,gw", a medium-size block diagram is shown to inform about the connection referred to, and:

```
**** press return to continue
```

is displayed. When the return key is pressed, the computed regulator transfer function

$$gw(s) = \frac{-8.215\,(s+1.855)\,(s+8.099)\,(s^2+1.844\,s+3.888)}{(s^4+25\,s^3+99.69\,s^2)}$$

is displayed and saved in file *gw.mat*. With:

`> gw=` *enter*

the factorized form of the same transfer function is displayed as

$$gw(s) = \frac{-8.215\,(s+1.855)\,(s+8.099)\,[(s+0.9219)\,s^2+1.743^2]}{s^2\,(s+4.979)\,(s+20.02)}\ .$$

Referring to the same plant, let us insert as zeros and poles in the fixed part of the regulator all the stable poles and zeros of the plant in order to obtain the maximum number of cancellations, i.e., define

$$gi(s) = \frac{-20\,(s+5)}{(s+1)\,(s-2)\,(s+8)}\ , \quad gj(s) = \frac{(s+1)\,(s+8)}{s^2\,(s+5)}\ .$$

According to these cancellations, the minimum number of assignable poles for the regulator to be causal is $\ell_0' = \ell_0 - n_c - m_c = 7-2-1 = 4$, so that we can assume

$$gk(s) = \frac{1}{(s+4)\,(s+6)\,((s+2)^2+2^2)}\ .$$

In this case the following additional message is displayed: v2

```
                                                  1 (s + 1) (s + 8)
regulator factors cancellable with plant = -----------------
                                                  (s + 5)

   **** press any key to continue
```

and the regulator

$$gw(s) = \frac{-5.2\,(s+1)\,(s+8)\,(s^2+1.692\,s+1.846)}{s^2\,(s+5)\,(s+16)}$$

is obtained by means of the above-described automatic zero-pole cancellation and recovering before and after design.

Let us briefly recall the messages displayed in some particular cases. Referring to the previous example, let us add a closed-loop pole, by assuming

$$gk(s) = \frac{1}{(s+4)\,(s+6)\,(s+10)\,((s+2)^2+2^2)}\ .$$

In this case, the message:

```
strictly proper regulator: the assigned poles may be reduced by 1
```

is displayed. Nevertheless, a feasible regulator is computed as

$$gw(s) = \frac{-71.2\,(s+1)\,(s+8)\,(s^2+1.371\,s+1.348)}{s^2\,(s+5)\,(s^2+26\,s+264)}\ .$$

If, on the other hand, we remove a pole and define

$$gk(s) = \frac{1}{(s+4)\,((s+2)^2+2^2)}\ ,$$

we obtain the message:

```
**** warning: non-proper regulator
```

and the computed regulator is

$$gw(s) = \frac{-0.5\,(s+1)\,(s+8)\,(s^2+2.4\,s+3.2)}{s^2\,(s+5)}\ .$$

If a further pole is removed, as in

$$gk(s) = \frac{1}{(s+4)\,(s+6)}\ ,$$

the denominator computed with the Diophantine equation is zero, hence not admissible, and the following messages appear:

```
**** warning: non-proper regulator
**** solution impossible: the computed denominator is zero !
```

On the other hand, if the number of the assigned poles is not less than ℓ_0 (or ℓ_0' when cancellations are suitably introduced), the regulator design is performed without any problem.

3.19 Regnich

The command

`> regnich,gi,gj` *(enter)*

provides a complete design environment based on the Nichols diagram for regulators whose transfer function consists of a gain coefficient and one or several zero-pole pairs of the lead or lag type; $gi(s)$ or $gi(z)$ is the transfer function of the plant (given), and $gj(s)$ or $gj(z)$ that of the regulator (to be determined). See the applications *lagc* and *leadc* for design with the Bode diagrams.

3.19.1 Recall

Let $G_i(s)$ be the transfer function of the given plant and $G_j(s)$ that of the feedback regulator to be derived. This is obtained by using one or more of the elementary factors

$$G_{j,1}(s) = K_c\,, \quad G_{j,2}(s) = \frac{1+\tau\,s}{1+\alpha\,\tau\,s}\,, \quad G_{j,3}(s) = \frac{1+\alpha\,\tau\,s}{1+\tau\,s}\,, \tag{3.24}$$

that will be referred to as *gain*, *lead compensator*, and *lag compensator*. The final regulator may consist of a gain factor and one or several lead and lag compensator sections.

The design is performed on the Nichols plot of the frequency response function $G_i(j\omega)$ of the plant (or of the partially corrected plant if some elementary factors have already been determined). The influence of the above elementary factors on the Nichols diagram is outlined in Fig. 3.63, where n_1 and n_2 denote the plot before and after the correction, respectively.

The effect of a gain section, shown in Fig. 3.63,a, is simply a vertical shift, so that, if both a point A (point FROM) on n_1 and a point B (point TO) on the vertical straight line through A are selected, the corresponding value of K_c in db is represented by the vector from A to B. In fact, this vector represents a real number, positive or negative according to whether A is below or above B.

The lead compensator case is shown in Fig. 3.63,b: selection of both a point A (point FROM) on n_1 and a point B (point TO) above and to the right of A defines a compensator such that n_2 passes through B, and A, B correspond to the same angular frequency on n_1 and n_2 respectively. In this case the vector from A to B is constrained to belong to a suitable domain $\mathcal{D}$, shown in dashed lines in the figure. Clearly, if B is selected first, the choice of A is constrained to belong to the symmetric domain $\mathcal{D}_1$, also shown in the figure.

The lag compensator case, shown in Fig. 3.63,c, is similar, but B (point TO) must be below and to the left of A (point FROM). In this case the choice of the vector from A to B is also constrained to belong to a domain $\mathcal{D}$, and, if B is the first point selected, the symmetric domain $\mathcal{D}_1$ must be considered when selecting A.

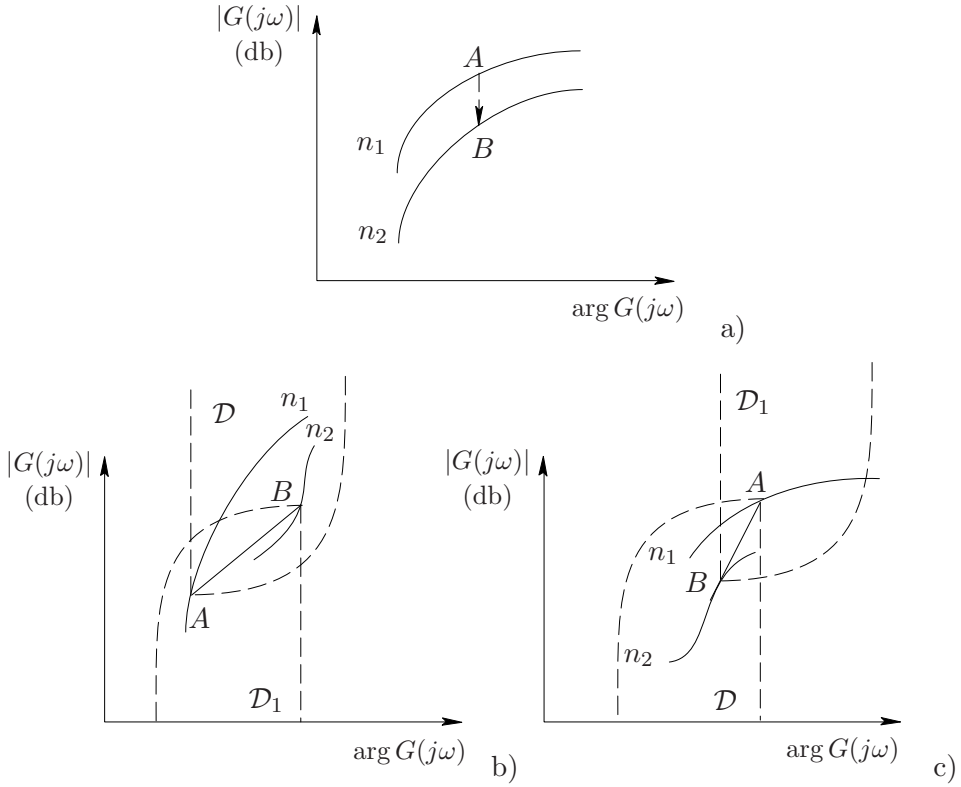

Figure 3.63. The Nichols diagrams for gain, lead and lag compensators.

The point B is usually selected on the zero db line in order to impose the phase margin. Nevertheless, different choices are possible, according to the designer's experience.

To show how the choice of A and B traduces in a factor of the lead or lag compensator type and where the restriction on the corresponding vector has its origin, the *inversion formulae* of the lead/lag compensators are presented here. These formulae make it possible to avoid and possibly forget the customary trial-and-error design methods.

Let us consider the function $F : (\alpha, (\omega\tau)) \rightarrow (M, \varphi)$ defined by

$$M\, e^{j\varphi} = \frac{1 + j\,(\omega\tau)}{1 + j\,\alpha\,(\omega\tau)}\ , \tag{3.25}$$

whose domain is $\alpha \in [0, 1)$, $(\omega\tau) \in (0, \infty)$, while the corresponding co-domain will be shown to be $M \in (1, \infty)$, $\varphi \in (0, \arccos(1/M)]$. Its inverse function $F^{-1} : (M, \varphi) \rightarrow (\alpha, (\omega\tau))$ is expressed by

$$\alpha = \frac{M\,\cos\varphi - 1}{M\,(M - \cos\varphi)}\ , \quad \omega\tau = \sqrt{\frac{M^2 - 1}{1 - M^2\alpha^2}} = \frac{M - \cos\varphi}{\sin\varphi}\ . \tag{3.26}$$

Relations (3.26) are called the *inversion formulae of the lead compensator*. Note that the expression on the right of (3.25) is the frequency response of $G_{j,2}(s)$ in (45). Referring to Fig. 66,b, let us denote by x_A, y_A and x_B, y_B the coordinates (in db and radians) of A and B respectively, and define $\Delta y := y_B - y_A$, $\Delta x := x_B - x_A$, so that M and φ are expressed by

$$M = 10^{\Delta y/20}\ , \qquad \varphi = \Delta x\ . \tag{3.27}$$

Let ω_0 be the angular frequency corresponding to A on n_1. Relations (3.26) directly define the parameters α and $\tau := (\omega\tau)/\omega_0$ of the lead compensator that transforms the Nichols plot n_1 into a Nichols plot n_2 passing through B at frequency ω_0. The domain $\mathcal{D}$ is simply the co-domain previously defined referred to a coordinate system with origin in A, while $\mathcal{D}_1$ is the symmetric domain, referred to a coordinate system with origin in B.

Since $G_{j,3}(s)$ is the reciprocal of $G_{j,2}(s)$, the inversion formulae of the lag compensator can be derived by simply replacing M with $1/M$ and φ with $-\varphi$ in (3.26). However, it is not necessary to introduce new formulae for the lag compensator, since the same effect can be obtained by simply changing the signs of Δx and Δy. Hence, referring to Fig. 66,c, the lag compensator is derived by defining $\Delta y := y_A - y_B$, $\Delta x := x_A - x_B$, and using (3.27) and (3.26) to derive α and $\omega\tau$. Of course, also in this case τ is obtained by division of $\omega\tau$ by ω_0, the angular frequency corresponding to A on n_1.

The proof of the inversion formulae

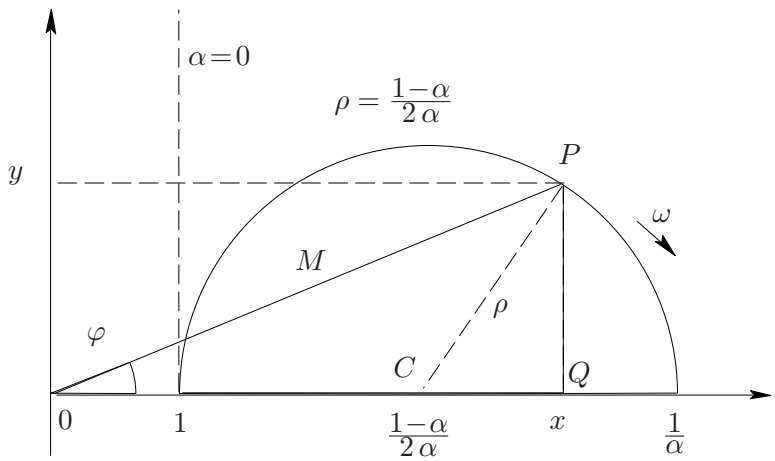

Figure 3.64. Derivation of the inversion formulae of the lead compensator.

Referring to (3.25), let us denote by $x=M\cos\varphi$ and $y=M\sin\varphi$ the real and the imaginary part of $M\,e^{j\varphi}$. It is easily shown that they describe the semi-circumference shown in Fig. 67 as ω varies from 0 to ∞. In fact, by shifting the origin of coordinates to C we obtain

$$\frac{1+j\omega\tau}{1+j\alpha\omega\tau}-\frac{1+\alpha}{2\alpha}=\frac{2\alpha\,(1+j\omega\tau)-(1+\alpha)\,(1+j\alpha\omega\tau)}{2\alpha\,(1+j\alpha\omega\tau)}=\frac{\alpha-1}{2\alpha}\;\frac{1-j\alpha\omega\tau}{1+j\alpha\omega\tau}\;:$$

since α is less than one, the last term on the right is a vector with constant absolute value $\rho=(1-\alpha)/(2\alpha)$ and phase angle ranging from 0 to π as ω varies from 0 to ∞, that clearly describes a semi-circumference. Referring to the right-angled triangle CPQ, we have

$$\Big(x-\frac{1+\alpha}{2\alpha}\Big)^2+y^2=\Big(\frac{1-\alpha}{2\alpha}\Big)^2\;,\quad\text{hence}\quad(2\alpha x-1-\alpha)^2+4\alpha^2y^2=(1-\alpha)^2\;,$$

that, by expansion and division by 4α, yields $\alpha x^2+\alpha y^2-\alpha x=x-1$. This provides the first inversion formula by simple substitution of x and y in terms of M and φ:

$$\alpha=\frac{x-1}{x^2+y^2-x}=\frac{M\,\cos\varphi-1}{M\,(M-\cos\varphi)}\;. \tag{3.28}$$

By squaring the absolute values of both members of (3.25) we obtain

$$M^2\,(1+\alpha^2\,(\omega\tau)^2) = 1+(\omega\tau)^2\,, \quad \text{hence} \quad \omega\tau = \sqrt{\frac{M^2-1}{1-M^2\alpha^2}}\,, \tag{3.29}$$

that is the first expression of the second inversion formula. On the other hand, from (3.28) we have

$$\begin{aligned} 1-M^2\alpha^2 &= 1-\frac{(M\,\cos\varphi-1)^2}{(M-\cos\varphi)^2} \\ &= \frac{M^2-2M\cos\varphi+\cos^2\varphi-M^2\cos^2\varphi-1+2M\cos\varphi}{(M-\cos\varphi)^2} \\ &= \frac{(M^2-1)\,(1-\cos^2\varphi)}{(M-\cos\varphi)^2} = \frac{(M^2-1)\,\sin^2\varphi}{(M-\cos\varphi)^2}\,, \end{aligned}$$

that, substituted in (3.29), yields the second expression, i.e.

$$\omega\tau = \frac{M-\cos\varphi}{\sin\varphi}\,.$$

Let us now derive an expression for the domain $\mathcal{D}$ to which the point TO is constrained to belong. Still referring to Fig. 67, note that the point P must belong to the infinite right-angled domain bounded below by the x-axis and on the left by the vertical line with unit abscissa. The maximum phase angle φ_M obtainable for a given M is such that $M\,\cos\varphi_M = 1$, hence $\varphi_M = \arccos(1/M)$. Note that φ_M is only obtainable with $\alpha = 0$, a limit case often excluded in standard compensator design.

If α is allowed to be zero, the domain $\mathcal{D}$ shown in Fig. 3.63,b is closed on the right. By taking into account relations (3.27), it can be expressed by

$$\mathcal{D} = \{(\Delta x, \Delta y)\,:\,\Delta y \in [0,\infty),\,\Delta x \in (0, \arccos 10^{-\Delta y/20}]\,\}\,,$$

while its symmetric domain is

$$\mathcal{D}_1 = \{(\Delta x, \Delta y)\,:\,\Delta y \in (-\infty,0],\,\Delta x \in [-\arccos 10^{\Delta y/20}, 0)\,\}\,.$$

These domains are simply exchanged with each other in the lag compensator case shown in Fig. 3.63,c.

3.19.2 Operation

Let us consider the transfer function

$$gi(s) = \frac{10000}{(s+1)\,(s+2)\,(s+10)\,(s+30)}\ , \tag{3.30}$$

that will also be considered in the next *Examples* section. After the command "regnich,gi,gj", first a medium-size block diagram is shown to inform about the connection referred to and:

```
**** press return to continue
```

is displayed. When the return key is pressed, we obtain the request:

```
information on the design method ? (1) :
```

Entering 1 produces a short description of the procedure used to derive a compensator with the inversion formulae, already presented in the previous *Recall* section. This optional information also generates the full-size figure with the Nichols diagrams of some lead compensators shown in Fig. 3.65.

If the request is skipped by simply pressing the return key, the Nichols diagram of $gi(s)$ is plotted with a complete angular frequency graduation. This is shown in Fig. 3.66: the reference lines for the gain and phase margins are also plotted, and the values of these margins are displayed at the bottom left of the figure. In the figure, the inscription **Regulator No. 1** appears in green. In fact, the program makes it possible to design up to six different regulators, each with plots and messages in a different color, according to the standard sequence *green*, *red*, *cyan*, *yellow*, *magenta*, and *blue*. The Nichols plot and the messages referring to the plant are shown in black when the background is white or in white when it is black.

In the command bar of the figure a **Change axes** menu is provided to adjust the automatic selection, with the items: **y top**, **y bottom** (plus 20 db, minus 20 db), **x left**, **x right** (plus 30 degrees, minus 30 degrees), **standard**, **M and N loci on/off**. The **Grid on/off** command is also provided. The axes in Fig. 3.66 have been adjusted by using the **standard** and **y top** options.

At the top left of the figure a pushbutton menu shows the three options for the design, **Lead**, **Lag** and **Gain**, and, initially disabled, the items **Continue**, **Another section**, **Another regulator**, **Step response**, **Bode diagrams**. A further pushbutton **Exit** is provided, at the bottom left corner of the figure, to quit the program.

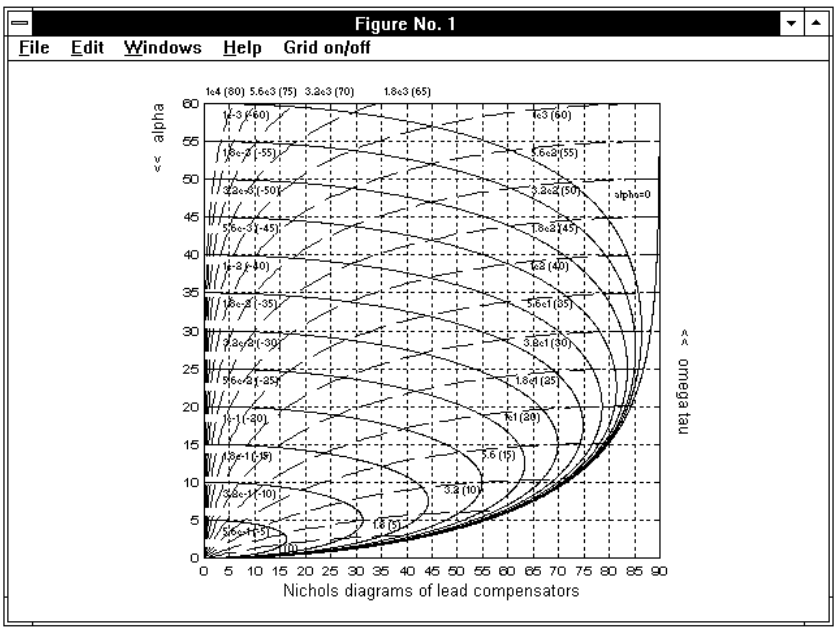

Figure 3.65. A family of Nichols diagrams of the lead compensator.

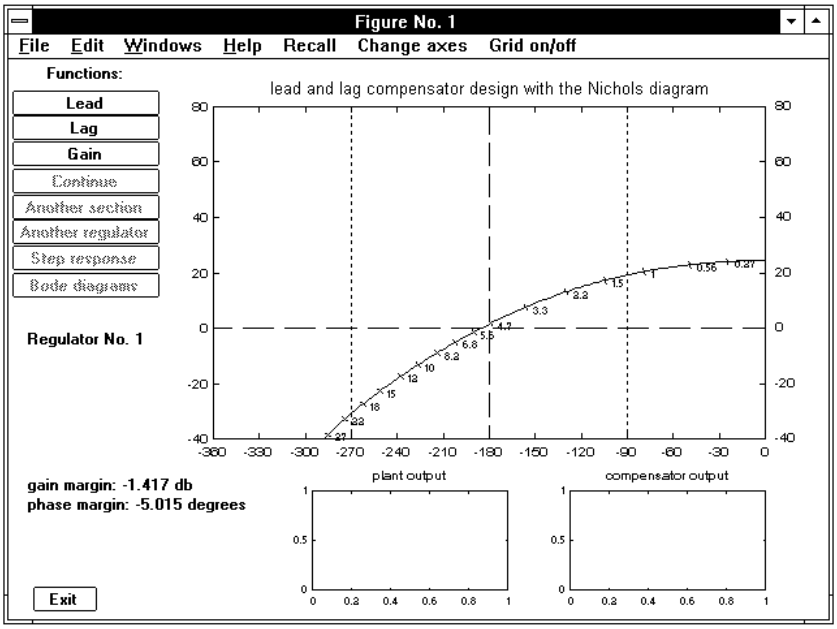

Figure 3.66. The initial figure with the Nichols diagram.

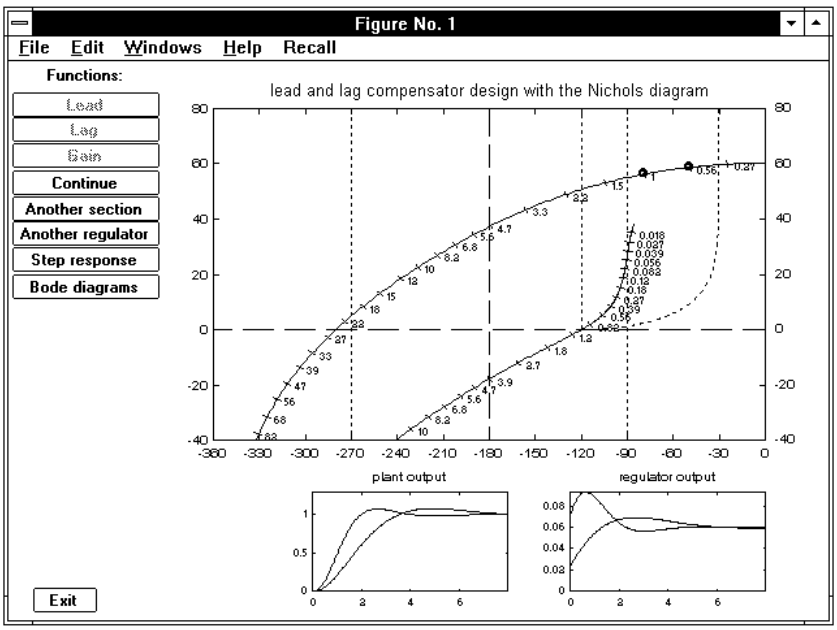

Figure 3.67. The figure layout at the end of a design session.

The small rectangles under the Nichols plot are used to see the step responses, both at the output of the plant and regulator, for the last two selections during a design session.

Clicking with the mouse on one of the three enabled pushbuttons starts a design session for one of the compensators with the transfer functions (3.24). The design procedure is not considered now, since it will be fully described in the next *Examples* section. When the design of a single compensator section is complete, the main pushbutton menu is recovered, with only the options **Continue**, **Another section**, **Another regulator**, **Step response**, and **Bode diagrams** enabled. The corresponding screen layout is shown in Fig. 3.67.

Let us provide here a brief description of these options:

Continue. This option makes it possible to continue the design of a regulator by selecting different points TO in the **Gain** case, FROM in the other cases, after checking response(s).

Another section. This option makes it possible to add another elementary factor to the regulator being designed. The consequent layout of the screen appears as at the beginning, but with the Nichols diagram modified to include the sections of the regulator already designed.

Another regulator. With this option, it is possible to repeat the synthesis procedure from the beginning without entering *regnich* again, thus enabling comparison of different solutions in terms of step and/or frequency responses. As previously recalled, the subsequent regulators designed are distinguished by color.

Step response. This option enables to check effectiveness of the regulator by means of the step response of the overall closed-loop system, both at the output of the plant and regulator. The operation and features of this command are the same as in application *pidnich*, and are not repeated here.

Bode diagrams. This option produces a display with the Bode diagrams of the overall system, open-loop or closed-loop. Since it is the same as in application *pidnich*, it is not described here.

Exit. This pushbutton provides exit from *regnich*. When clicked on, the message:

PRESS RETURN TO EXIT

appears in orange at the bottom left corner of the screen. On pressing return, the Command Window is recovered and the transfer function $gj(s)$ or $gj(z)$ displayed and saved in the hard disk. If more than one regulator have been derived, the standard selection by color is requested in interactive mode.

NOTE:

- It is clear from the above operation notes that application *regnich* also works in the discrete-time case. If the transfer function of the plant is discrete-time, i.e., a $gi(z)$ instead of a $gi(s)$ is specified on entering, the design with the Nichols diagram is automatically referred to its w-plane equivalent, and every regulator section derived is converted back from w to z. The step responses and Bode diagrams are accordingly modified.

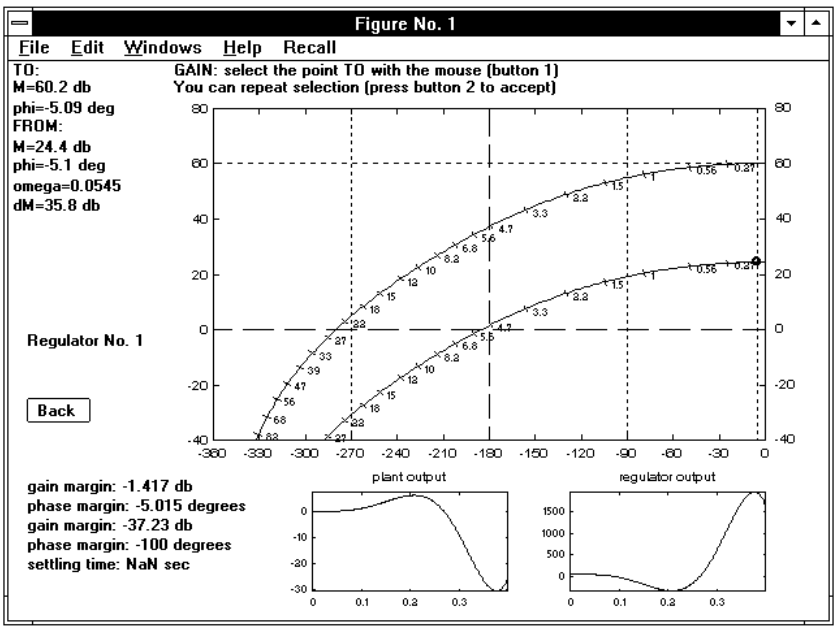

Figure 3.68. Selection of the gain.

3.19.3 Examples

Let us refer again to the transfer function (3.30). We start our design with the screen appearing as in Fig. 3.66. The system is (slightly) unstable, so that, if its dc gain is acceptable in connection with possible specifications on static behavior, it may be compensated with one or two sections of the lead type. If not, increase of gain and a lag compensator should be used. We decide to proceed in this latter way, and to adjust the gain first, to improve the steady-state error. When clicking with the mouse on **Gain**, the above menu disappears, the following message appears at the top of the figure:

GAIN: select the point TO with the mouse (button 1)
You can repeat selection (press button 2 to accept) (a)

and a white cross is displayed to enable selection. Fig. 3.68 shows the screen layout after the first selection, the aim of which was to impose 1000 (60 db) as the open-loop dc gain. A vertical green dotted line (that in this case is on the right of the figure) points out the FROM and TO point locations.

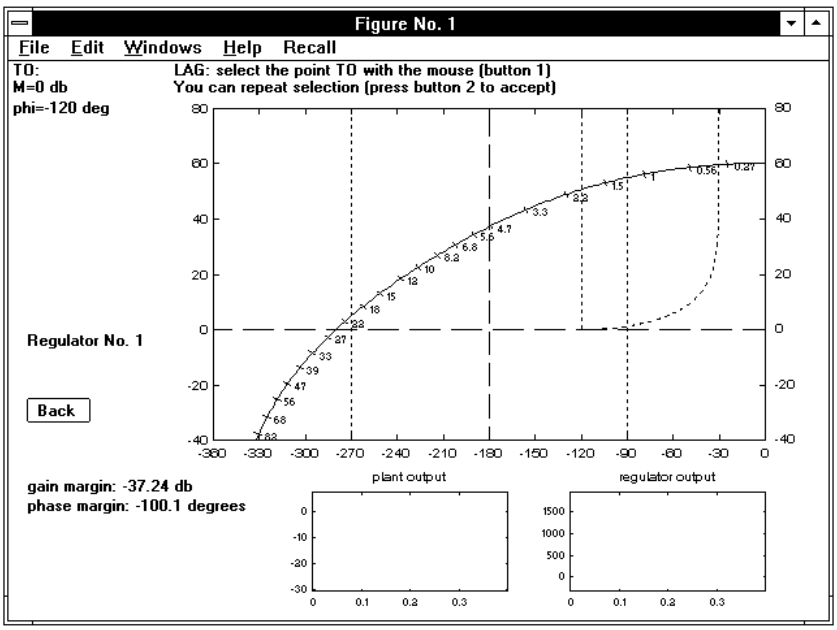

Figure 3.69. Design of a lag compensator: choice of the point TO.

The point FROM is denoted by a small circle, and some information about the imposed change of gain, the new stability margins and settling time is displayed. The step responses at the output of the plant and regulator are also shown in the small rectangles, clearly pointing out that the overall system is unstable after the first choice, and that the regulator requires a stabilizing section.

When button 2 of the mouse is pressed, the main pushbutton menu is recovered, with the options **Continue**, **Another section**, **Another regulator**, **Step response**, and **Bode diagrams** enabled. On clicking on **Another section**, we obtain again the screen layout shown in Fig. 69, but with the Nichols diagram shifted upwards according to the change of gain previously introduced and the information about the gain and phase margin accordingly modified. If we choose **Lag** for the regulator next section, the main menu disappears again and message (a) is replaced by:

LAG: select the point TO with the mouse (button 1)
You can repeat selection (press button 2 to accept) (b)

After one or more choices of the point TO, the screen appears as in Fig. 3.69.

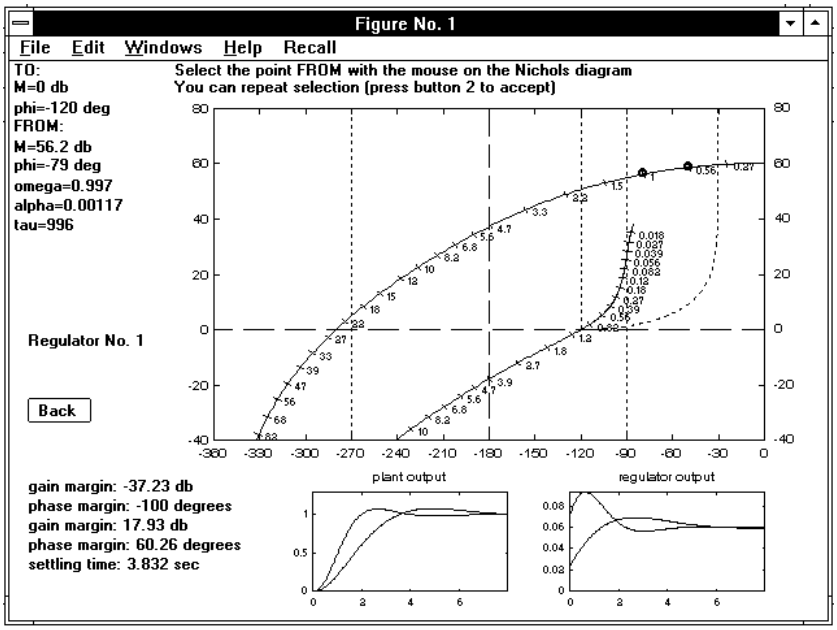

Figure 3.70. Design of a lag compensator: choice of the point FROM.

The pushbutton **Back** produces recovery of the previous layout with the three choices enabled, thus making it possible to reset the current design session. When button 2 of the mouse is pressed, selection of the point FROM is requested. This selection can also be repeated, possibly taking into account the displayed values of the stability margins, settling time and the regulator parameters. The step responses corresponding to the last two choices are shown in the two small rectangular windows under the Nichols diagram. At this point the screen appears as in Fig. 3.70. On pressing button 2 to confirm the last selection we obtain the layout in Fig. 3.67, from which it is possible to exit the program by clicking on **Exit**.

This disables the pushbutton menu and produces, at the bottom left of the screen, the request to press the return key to exit. When return is pressed, the Command Window appears with the derived transfer function, displayed as:

```
   THE REGULATOR OBTAINED :

      0.07208 (s + 0.8601)
gj = --------------------
      (s + 0.001002)
```

3.20 Regrootl

The command

```
> regrootl,gi,gj   (enter)
```

provides a design environment for regulators based on the root locus, both by open-loop pole-zero-gain assignment or closed-loop pole assignment with the Diophantine equation. In the above call string, $gi(s)$ or $gi(z)$ is the transfer function of the plant (given) and $gj(s)$ or $gj(z)$ that of the regulator (to be determined).

3.20.1 Recall

There are, in the TFI environment, six applications for regulator and compensator synthesis using the frequency domain approach, where the performance of the closed-loop system is mainly measured in terms of gain and phase margin, resonance peak and resonance frequency. They are *lagc*, *leadc*, *pidc*, *pidd*, *pidnich*, and *regnich*, all providing high-speed interactive operation (four of them with the keyboard and two with the mouse), hence particularly suited for trial-and-error design.

However, when the controlled system is unstable or nonminimum-phase, the standard frequency domain design methods are no longer applicable, and a better insight on the features of the feedback loop and possibility of intervention on its dynamic performance is obtained by considering the closed-loop zero-pole map.

It is well known that design through zero-pole allocation is based on the root locus rules, that enable relating the effect of adding and locating open-loop zeros and poles (in the regulator) to the overall closed-loop system performance. Besides addition of zeros and poles, program *regrootl* makes it possible to immediately draw the root locus for every change in the zero-pole map, and to select on it the most convenient value of the gain, to change the zero-pole map if it appears susceptible of improvement, and to check the step and frequency responses. Thus, it is a powerful tool for very fast on-line adjustment of the regulator parameters.

A further feature of *regrootl* is the optional use of the Diophantine equation for complete closed-loop pole assignment. In this case, after a sufficient number of closed-loop poles has been introduced with the mouse, the regulator is computed by the program. Nevertheless, it may also be adjusted as before by modifying its zero-pole map and/or gain if necessary. See the *Recall* section of application *regdph* for information on the use of the Diophantine equation for closed-loop pole allocation.

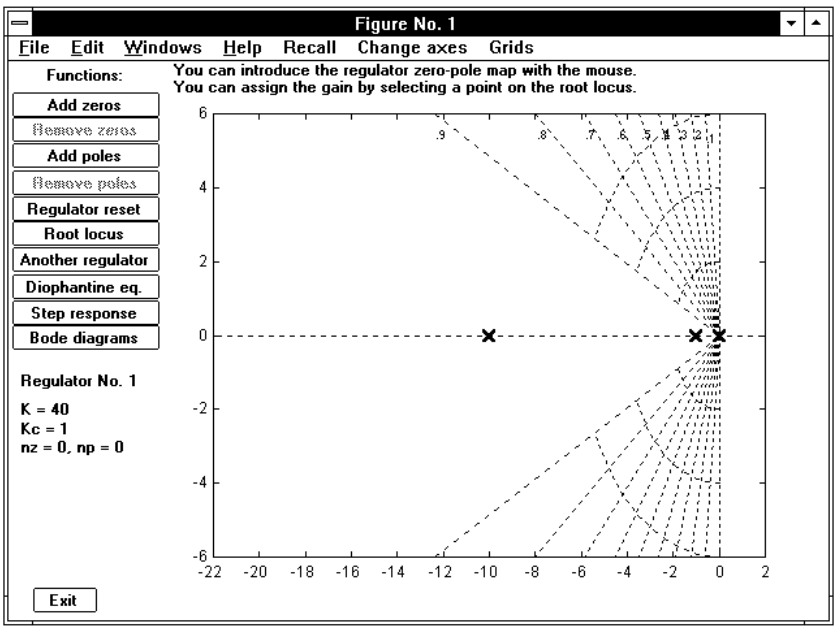

Figure 3.71. The initial layout with the poles and zeros of the plant.

3.20.2 Operation and Examples

Let us consider the plant transfer function

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ . \tag{3.31}$$

After the command "regrootl,gi,gj", first a medium-size block diagram is shown to inform about the connection referred to, and:

```
**** press return to continue
```

is displayed. When the return key is pressed, we obtain the screen layout shown in Fig. 3.71, with the zero-pole map of transfer function (3.31) shown in black when the background is white or in white when it is black. In the figure, the inscription **Regulator No. 1** appears in green. In fact, the program makes it possible to design up to six different regulators, each with plots and messages in a different color, according to the standard sequence *green*, *red*, *cyan*, *yellow*, *magenta*, and *blue*.

The data **K = 40**, **Kc = 1**, and **nz = 0, np = 0**, appearing under the above inscription, are the loop gain (referred to the zero-pole-gain form of the overall transfer function), the regulator gain, and the number of zeros and poles of the regulator. They are continuously updated in the course of the design procedure.

In the command bar at the top of the figure a **Change axes** menu is provided to adjust the automatic axes selection, with the items: **x left**, **x right**, **y up and down** (enlarge by 1 tick, enlarge by 2, reduce by 1 tick, reduce by 2), **zoom in**, **zoom out**, and a **Grids** menu with **Grid on/off**, **Constant damping loci (on/off)**. The axes in Fig. 3.71 have been adjusted by using the **x left** option of the respective menu and the constant damping loci have been inserted with the **Grids** facility.

At the top left of the figure a pushbutton menu shows the options for the design, **Add zeros**, **Remove zeros**, **Add poles**, **Remove poles**, **Regulator reset**, **Root locus**, **Another regulator**, **Diophantine eq.**, **Step response**, and **Bode diagrams**. A further pushbutton, **Exit**, is provided at the bottom left corner to quit the program. Since the regulator is initially lacking in zeros and poles, in Fig. 3.71 the **Remove zeros** and **Remove poles** options appear disabled.

The following message is displayed at the top of the figure:

You can introduce the regulator zero-pole map with the mouse.
You can assign the gain by selecting a point on the root locus. (a)

Let us briefly describe the pushbutton menu operation.

Add zeros. This choice enables selection with the mouse of a point in the figure and produces replacement of message (a) with:

Select a location for a new real zero or a new complex zero pair.
You can repeat selection with the mouse. Press button 2 to accept. (b)

Selection with the mouse of a point above the real axis defines a complex zero pair, while selection (slightly) below defines a real one. In both cases the selected zero locations are shown in green (the color of Regulator No. 1) with standard symbols in the figure, while their numerical value(s) are displayed, also in green, at the left of the figure. Selection is performed and possibly modified by clicking button 1 and eventually confirmed by button 2. Introduction of more zeros is obtained by repeating the whole procedure. A pushbutton **Back**, located at the bottom left corner of the figure, makes it possible to null the *Add zeros* session.

Remove zeros. This choice enables removal of a real zero or a complex zero pair by selection with the mouse and clicking button 1.

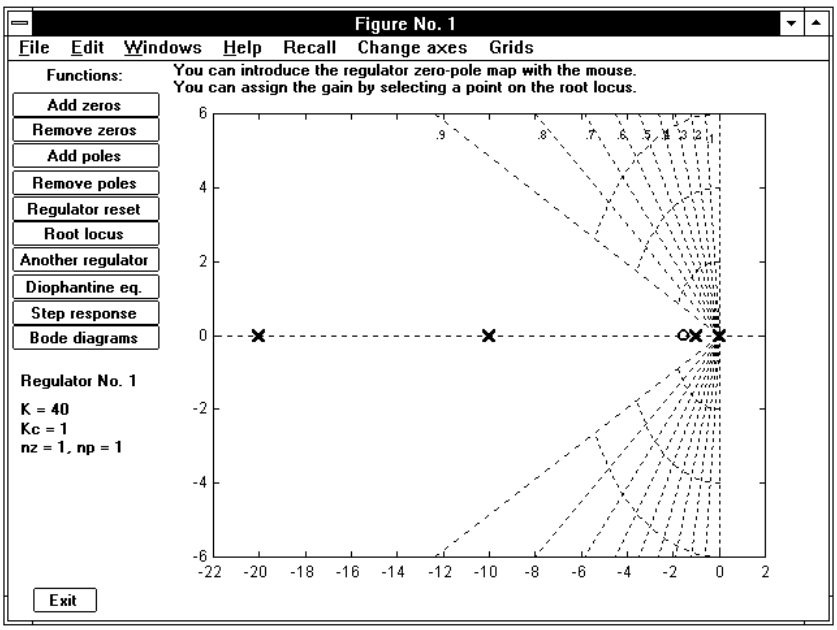

Figure 3.72. The figure layout during the design of a lead compensator.

Add poles. Same as above for zeros, with message and symbols accordingly modified.

Remove poles. Same as above for zeros.

Regulator reset. All the zeros and poles of the current regulator are deleted and its gain is set equal to one, to start its design again from the beginning.

Root locus. This pushbutton enables the main and most sophisticated feature of program *regrootl*. When clicked on, the menu is disabled and two new pushbuttons appear on the screen, with the options **Positive** and **Negative**. The first enables the root locus to be drawn for the current values of the zero-pole maps and gains of the plant and regulator, while the second changes the sign of the regulator gain before drawing and may be used in particular cases. Before the root locus plot is drawn, a small window with a pop-up menu is available in the figure for selection of the step size (the possible values are .01, .02, .05, .1, .2, .5, 1). This option is quite useful to impose the speed of drawing, thus making it possible to glance at the appearance of all the locus branches as the loop gain increases.

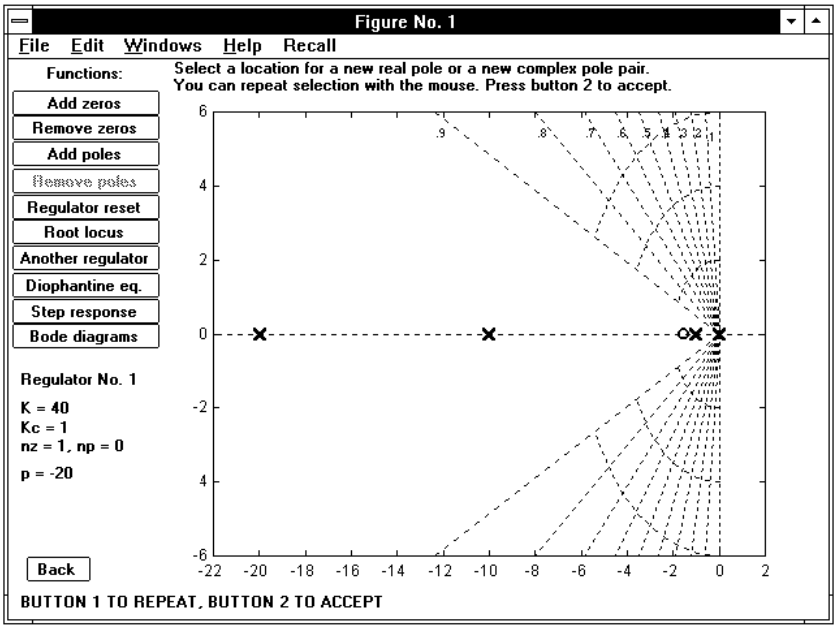

Figure 3.73. The figure after pole and zero selection.

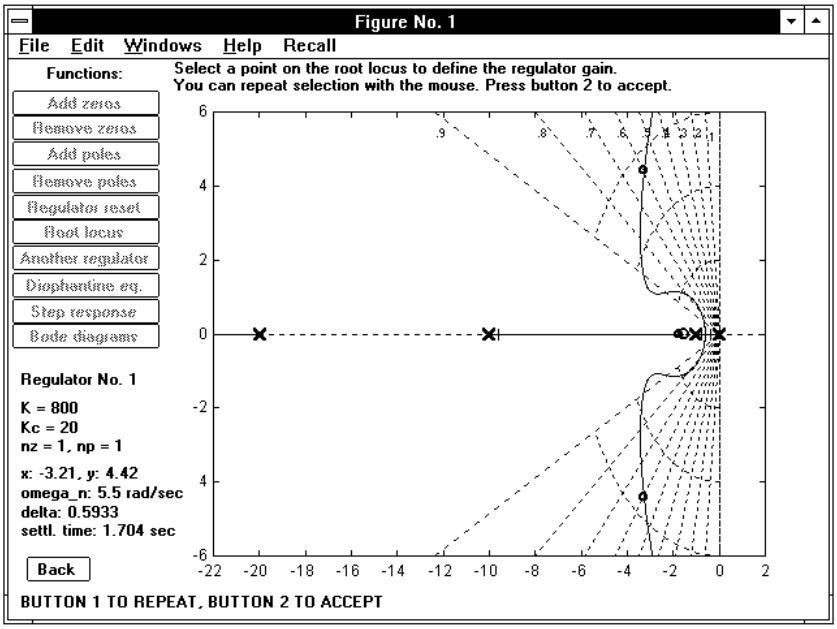

Figure 3.74. Selection of gain on the root locus.

For instance, when the Diophantine equation is used for closed-loop pole assignment, if the root locus passes very slowly through the dominant poles, we may conclude that the regulator being designed is quite robust, at least versus gain changes.

It is possible to null the **Root locus** session by clicking on the **Back** pushbutton available in the figure: this is particularly useful to change the sign of the regulator gain or when some adjustment of the zero-pole map is advisable.

The menu options considered so far are used in design by zero-pole-gain assignment. Referring to the plant (3.31), let us briefly present an illustrative example, concerning a lead regulator. Suppose we introduce a zero and a pole with values -1.5 and -20 respectively. When the pole is being located and possibly adjusted with the mouse, the screen appears as in Fig. 3.72. On exiting the pole assignment session by clicking button 2, the standard layout with all the options enabled, shown in Fig. 3.73, is obtained.

The next step is selection of the gain on the root locus, that is provided by clicking on the corresponding menu option. The typical screen layout with a root locus plotted is shown in Fig. 3.74, with the following message displayed at the top of the figure:

Select a point on the root locus to define the regulator gain.
You can repeat selection with the mouse. Press button 2 to accept. (c)

Fig. 3.74 shows the screen layout during the choice of the gain, possibly repeated, with button 1 of the mouse. The selected root locations are pointed out with small circles on the locus, and the corresponding values of the natural frequency, damping factor and settling time are displayed at the bottom left of the figure. Settling time is defined as $3/\sigma_m$, where σ_m is the minimum absolute value of the real parts of the closed-loop poles or of their transforms according to $s = (1/T)\ln z$ in the discrete-time case. Clicking button 2 provides exiting the gain assignment session, thus re-obtaining the screen layout in Fig. 3.73, with the displayed values of the overall and regulator gain updated.

Let us now continue the description of the pushbutton menu.

Another regulator. This option enables another regulator complete synthesis procedure from the beginning without entering *regrootl* again. The subsequent regulators designed are distinguished by color, in order to easily compare their time and frequency responses.

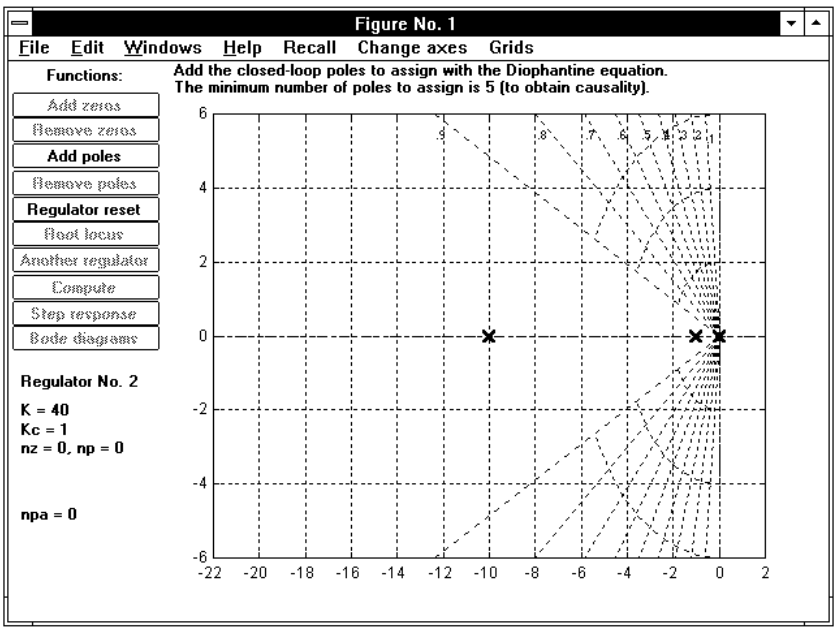

Figure 3.75. The figure before a Diophantine equation design session.

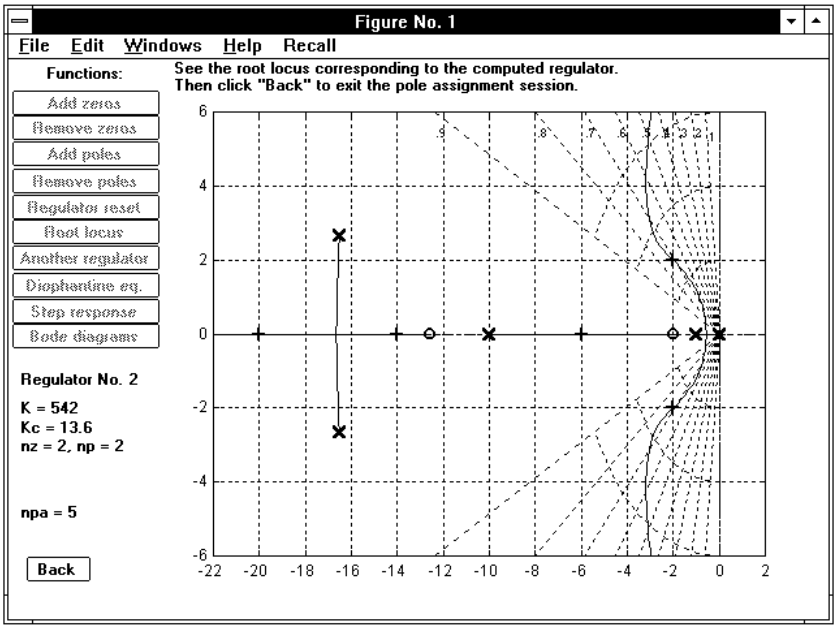

Figure 3.76. Checking the root locus after a pole assignment session.

Diophantine eq. This option can be used both at the beginning or when some regulator zeros and/or poles have already been introduced with the mouse. It enables continuation of a design session by direct computation of the regulator with complete closed-loop pole assignment.

Let us describe its operation with an example. Still referring to the plant (3.31), we continue our design session, after the lead compensator has been derived, by clicking on **Another regulator**. The choice **Diophantine eq.** produces the typical screen layout shown in Fig. 3.75, with the message:

Add the closed-loop poles to assign with the Diophantine equation
The minimum number of poles to assign is 5 (to obtain causality) (d)

at the top of the figure, and with only the options **Add poles** and **Regulator reset** enabled in the main menu. Of course, the minimum number of poles specified in the message depends on the particular case in hand. Note that the option **Diophantine eq.** has been replaced with **Compute**, initially disabled. This is enabled when some poles have been introduced and produces the regulator computation. Causality is not strictly required for computation. We introduce the poles $-2 \pm j2$, -6, -14 and -20, that are marked with small crosses in the current regulator color (red in this case), while their number N is displayed as **npa = N** under the other messages in the figure. Finally, we enable the computation by clicking on the corresponding pushbutton. If the number of poles introduced is not sufficient for the Diophantine equation to have a nontrivial solution, we obtain meanwhile the message:

SOLUTION IMPOSSIBLE: ADD OTHER POLES

in red, in the figure. It is deleted by clicking the mouse. When the regulator has been computed with the Diophantine equation, its zeros and poles are displayed and the option **Add poles** is disabled, while **Root locus** is enabled again, to check if the locus passes through the assigned poles. If it is chosen, the screen appears as in Fig. 79, with only the pushbutton **Back** enabled. This makes it possible to recover the main menu with all the options available.

Step response. This option makes it possible to check the effectiveness of the regulator(s) by means of the step response of the overall closed-loop system, both at the output of the plant and regulator. The operation and features of this command are the same as in application *pidnich* and are not repeated here. Its use in the case of our example, after the second regulator has been derived with the Diophantine equation, produces the step response plots (output of the plant) shown in Fig. 3.77.

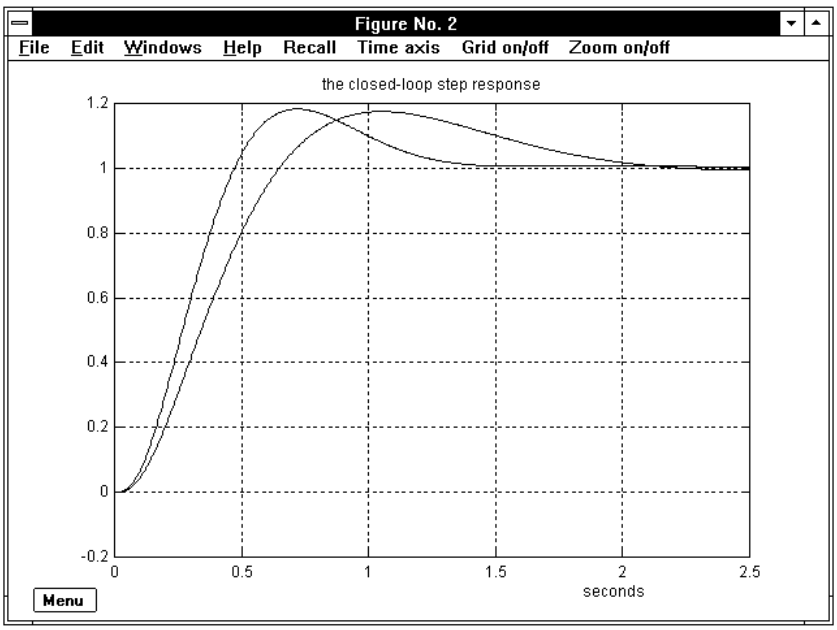

Figure 3.77. Checking the step response of the regulators obtained.

Bode diagrams. This option produces a figure with the Bode diagrams, open-loop or closed-loop, of all the previously computed regulators. Since it is the same as in application *pidnich*, it is not described here.

Exit. This pushbutton provides exit from *regrootl*. When clicked on, the message

PRESS RETURN TO EXIT

appears in orange at the bottom left of the screen. On pressing return, the Command Window is recovered and the transfer function $gj(s)$ or $gj(z)$ displayed and saved in the hard disk. If more than one regulator have been derived, the standard selection by color is requested in interactive mode.

NOTE:

- The application *regrootl* also works in the discrete-time case. If the transfer function of the plant is discrete-time, i.e., a $gi(z)$ instead of a $gi(s)$ is specified on entering, the design environment is accordingly modified. For instance, the constant damping grid is different.

3.21 Robpar

The command

`> robpar,gj` *(enter)*

defines the parametric form $gj\_p(s)$ of the transfer function $gj(s)$. This parametric form is permanently saved in the file *gj_ p.mat*.

`> robpar,gi,gj_p,h,gw` *(enter)*

performs a robustness analysis of the control loop with regulator $gi(s)$ (possibly a real constant), plant $gj\_p(s)$ (in parametric form), feedback connection $h(s)$ (possibly a real constant). The analysis concerns the closed-loop pole locations versus the parametric changes. The output $gw(s)$ (optional) is the transfer function of the worst-case plant.

3.21.1 Recall

Let us consider the feedback system shown in Fig. 3.78. The symbol p denotes a finite set of parameters, each subject to vary at random in a given *uncertainty interval*.

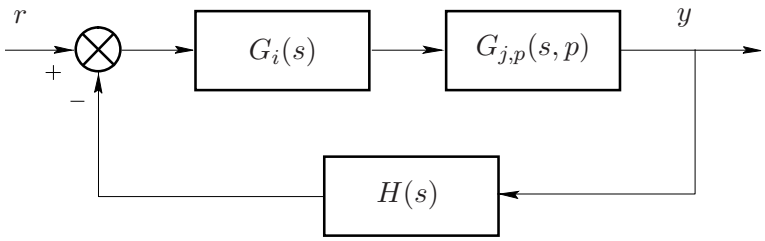

Figure 3.78. The parameter-dependent feedback system considered.

The transfer functions in parametric form considered in *robpar* are introduced by referring to any previously defined "reference" transfer function and substituting the numerical values of its coefficients with Matlab-interpretable expressions of a certain number of parameters (up to seven), each characterized by a nominal value and the extreme values of a uncertainty interval, and possibly repeated in the expressions. Let us consider the following simple example. For instance the transfer function

$$G_j(s) = \frac{40}{s\,(s+1)\,(s+10)} \tag{3.32}$$

can be used as a reference to define the corresponding parametric transfer function as

$$G_{j,p}(s,p) = \frac{40 + 20\,\sin\alpha}{s\,(s+\beta)\,(s+\gamma)}\ , \tag{3.33}$$

whose parameters have the nominal values and ranges of uncertainty specified as

$$\alpha\ :\ \text{nominal value } \pi,\ \text{range } [0, 2\pi]\ , \tag{3.34}$$
$$\beta\ :\ \text{nominal value } 1,\ \text{range } [0.1, 2]\ , \tag{3.35}$$
$$\gamma\ :\ \text{nominal value } 10,\ \text{range } [8, 16]\ . \tag{3.36}$$

Program *robpar* provides the following graphic information.

1. the closed-loop pole layout corresponding to the extreme values of each parameter in its own variation interval, in all the possible combinations, that are 2^n for n parameters;

2. the closed-loop pole scattering due to random, independent changes of all the parameters in their variation intervals;

3. the root contours of the closed-loop poles corresponding to the variation of a single parameter in its interval and passing through a point of the previous contour, selected with the mouse.

On exiting, some information on the worst-case plant is provided. The worst-case plant is defined as the plant whose closed-loop dominant (least-damped) poles have the minimum damping coefficient. The value $\bar{\delta}$ of this coefficient and the corresponding values $\bar{p}$ of the parameters are displayed. In the four-argument-call case the worst-case transfer function $G_w(s) = G(s, \bar{p})$ is also displayed and saved in the hard disk.

NOTES:

- Although the worst case plant usually corresponds to extreme values of the parameters, this is not the rule, since the plant transfer function coefficients can be arbitrary nonlinear functions of the parameters. This happens, in particular, in transfer function (3.33), where the gain assumes the same value at the extremes of the variation interval of α.

- Entering *robpar* with $gi = 1$ provides the worst-case plant in feedback connection without any compensator. This plant may be the first reference for trial-and-error robust compensator design.

3.21.2 Operation and Examples

One-argument call

Let us first consider the call with only one argument, with the aim at defining a parametric transfer function. We refer to the transfer function defined in (3.33) and (3.36), and assume that the reference transfer function $gj(s)$ defined in (3.32) has already been saved in the hard disk. Entering "robpar,gj" produces, in the Command Window, the display:

```
     40
gj = ------------------
     s (s + 1) (s + 10)

   do you want to define the parametric tf gj\so p ? (1/0) :
```

Entering 1 starts the following interactive section:

```
   LIST OF ALL THE PARAMETERS :

   parameter (return to end the list) : alpha

   nominal value : pi

   range of variation [min max] : [0 2*pi]

   parameter (return to end the list) : beta

   nominal value : 1

   range of variation [min max] : [.1 2]

   parameter (return to end the list) : gamma

   nominal value : 10

   range of variation [min max] : [8 16]

   parameter (return to end the list) :
```

On pressing return to end the introduction of parameters, the interactive session continues as follows:

```
NUMERATOR :

gain = 40
parametric expression : 40+20*sin(alpha)

DENOMINATOR :

coefficient: 1
parametric expression : beta
coefficient: 10
parametric expression : gamma
```

At this point the parametric transfer function has been completely defined. It is saved in file *gj_p.mat* and displayed as:

```
NOMINAL NUMERIC FORM :

     40
gj = ------------------
     s (s + 1) (s + 10)

PARAMETRIC FORM :

       40+20*sin(alpha)
gj_p = ------------------------
       s (s + beta) (s + gamma)

alpha: nominal value 3.142, range [0, 6.283]
 beta: nominal value 1    , range [0.1, 2]
gamma: nominal value 10   , range [8, 16]
```

The same display appears when the parametric transfer function is recalled from TFI by entering:

```
> gj_p  (enter)
```

If the parametric transfer function *gj_p* has already been defined, entering "rob-par,gj_p" produces the above display, followed by the request:

```
1 - define a new parametric form of gj_p
2 - change the parametric expessions of the coefficients
3 - change the parameter ranges

input your choice (return to exit) :
```

that enables its change, both complete or partial, in interactive mode.

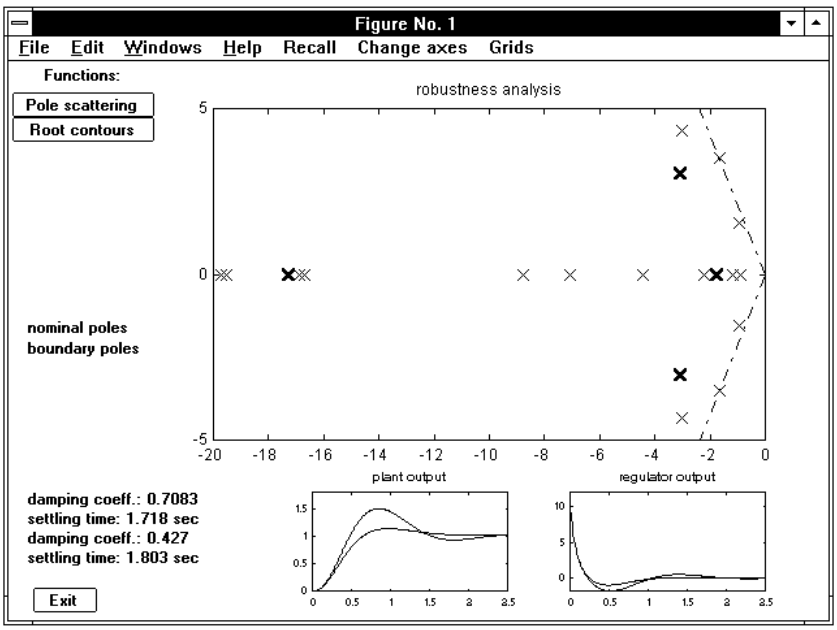

Figure 3.79. The initial layout of the graphic window.

Three-argument or four-argument call

Let us now consider the call with three arguments (to get only graphic output) or with four arguments (to get the worst-case transfer function of the plant also displayed and saved).

Let

$$gi(s) = \frac{10\,(s + 1.413)}{s + 14.13}$$

be the transfer function of a regulator for the nominal plant (3.32). Our aim is to check whether this regulator behaves satisfactorily also in the presence of any parameter variation in its uncertainty interval.

Entering "robpar,gi,gj_p,1,gw" produces a medium-size block diagram to be shown for information about the connection referred to, and the display of:

```
**** press return to continue
```

On pressing the return key, the nominal numeric form and the parametric form of $gj\_p(s)$ are shown in the Command Window, followed by:

```
**** press any key continue
```

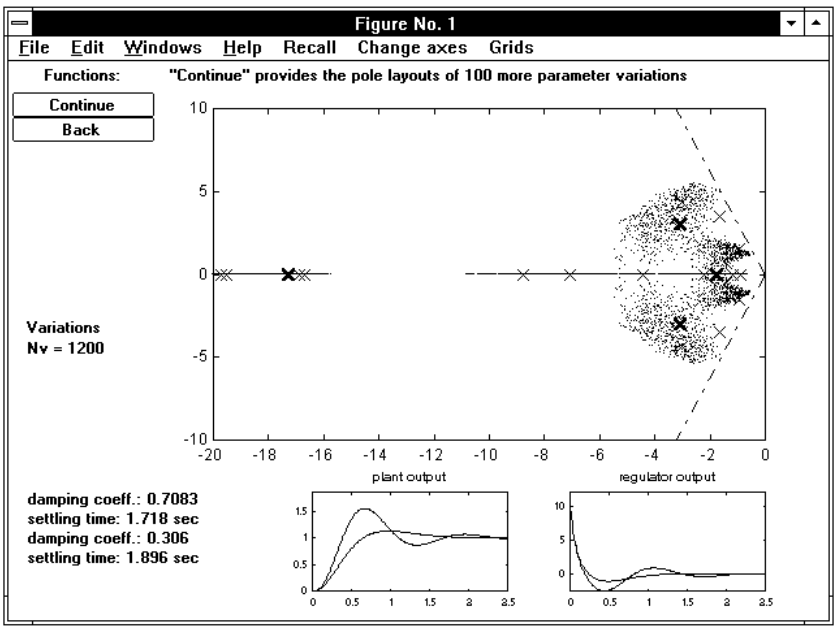

Figure 3.80. The "pole scattering" session.

On pressing a key, the screen layout in Fig. 3.79, showing the pole locations corresponding to the nominal (in green) and the extreme (in red) values of the parameters, is obtained, while in the small rectangular frames the nominal and worst-case step responses, both at the output of the plant and of the regulator, are shown in the same colors. The corresponding values of the damping coefficient and settling time are displayed on the left, while the worst-case constant damping loci (two half-lines through the origin) are also shown in orange.

In the command bar at the top of the figure a **Change axes** menu is provided to adjust the automatic selection, with the items: **x left**, **x right**, **y up and down** (enlarge by 1 tick, enlarge by 2, reduce by 1 tick, reduce by 2), **zoom in**, **zoom out**, and a **Grids** menu with **Grid on/off**, **Constant damping loci on/off**.

The pushbutton menu on the left of the figure enables two different refinements of the search for the worst-case parameter configuration, **Pole scattering** and **Root contours**. The **End** pushbutton, that provides exit from the program, is also visible at the bottom left corner of the figure. Let us briefly describe these options.

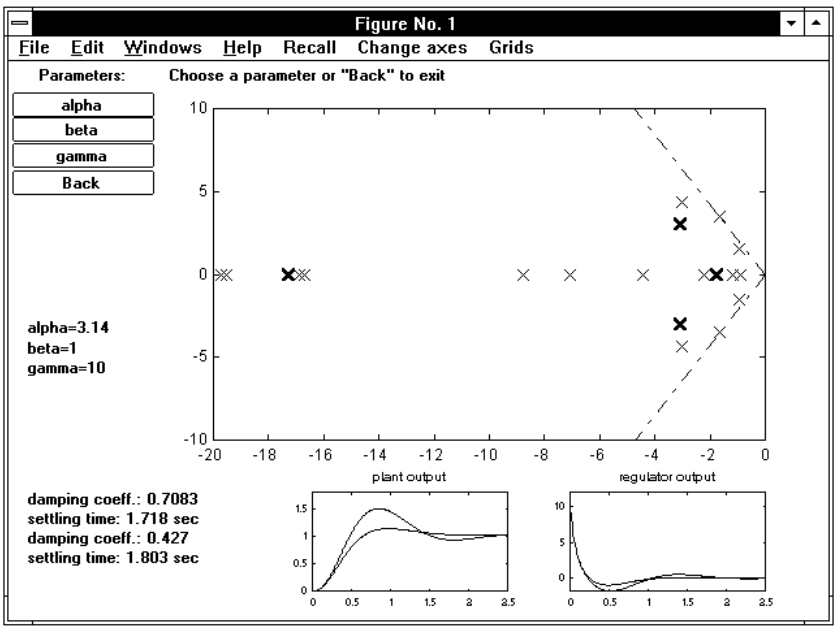

Figure 3.81. The beginning of the "root contours" session.

Pole scattering. With this option, the pole scattering due to 100 random parameter variations, where every parameter varies at random in its uncertainty interval, is computed and shown. The minimum damping coefficient case is selected and the corresponding values of the damping coefficient and settling time are displayed on the left of the figure. The correponding half-lines through the origin and the step responses are accordingly modified. A new pushbutton menu is visible, with the choices **Continue**, that provides addition of 100 more random coefficient variations, and **Back**, that enables exit to the previous menu. Fig. 3.80 shows the screen layout in the case of the example on hand, after 1,200 random parameter variations. Note that the damping coefficient is reduced with respect to that shown in Fig. 3.79, since the worst case does not correspond to extreme parameter values in this case.

Root contours. This option provides the screen layout of Fig. 3.81 with a push-button menu with all the parameter names (**alpha**, **beta** and **gamma** in this case) and **Back**. The message:

Choose a parameter or "Back" to exit (a)

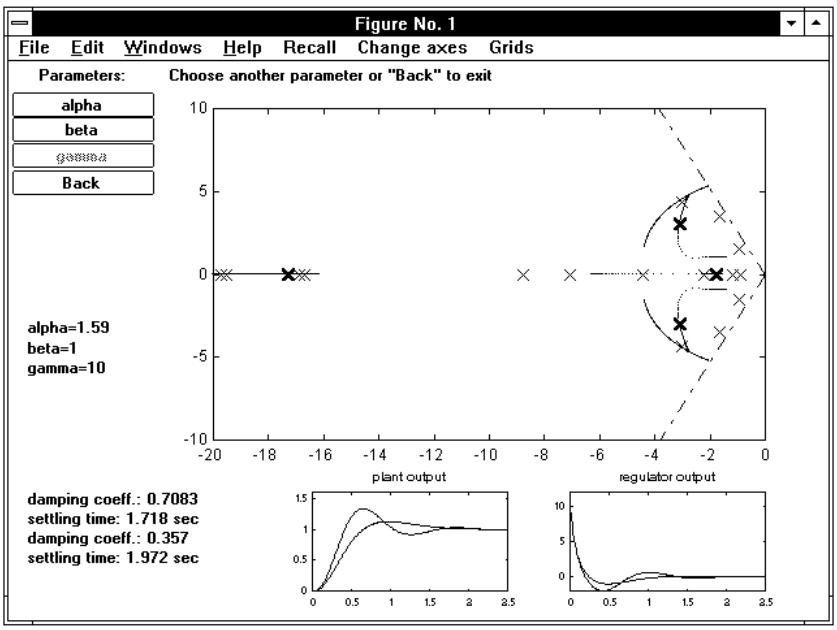

Figure 3.82. Continuing the "root contours" session.

is displayed at the top of the figure, while the nominal values of the parameters are displayed on the left. They are each distinguished with a color of the standard sequence: *green*, *red*, *cyan*, *yellow*, *magenta*, *blue*, and *black* when the background is white or *white* when it is black. Suppose we click on **alpha**: the root contour corresponding to 100 equally-spaced variations of the selected parameter in its uncertainty range is drawn in the figure, the pushbutton menu is shown again with **alpha** disabled, and the message:

Choose another parameter or "Back" to exit (b)

shown in place of (a). The worst-case step responses, the displayed values of the damping coefficient and settling time, and the corresponding constant damping loci are modified if the value of $\bar{\delta}$ is less than the previous one. When another parameter, for instance **gamma**, is selected, message (b) is replaced with:

Select a point on the green plot (button 2 to recover the previous menu) (c)

Selection of a point on the green plot defines a new value of the parameter alpha, immediately visible on the left of the figure.

Then, the root contour of gamma, passing through this point, is drawn in cyan. It is possible to repeat the selection, thus defining new values of alpha and adding new root contours. When button 2 is clicked, the previous menu is recovered, but with pushbutton **gamma** disabled and the previous contours cancelled, except for the last one, and with message (b) shown again. At this point the screen appears as shown in Fig. 3.82. Selection of a new parameter produces a new value of gamma to be defined and a new contour to be drawn, relative to the selected parameter and in the corresponding color. This procedure can be repeated. Selection of **Exit** provides recovering of the original layout of Fig. 3.79, but with the worst-case step responses, the displayed values of the damping coefficient and settling time, and the constant damping loci through the origin modified if a smaller value of $\bar{\delta}$ has been found during the root contours session.

End. This pushbutton provides exit from *robpar*. When clicked on, the message:

PRESS RETURN TO EXIT

appears in orange at the bottom left corner of the screen. On pressing return, the Command Window is recovered with the display:

```
THE LEAST-DAMPED CASE :

(delta = 0.2843)

alpha = 1.587
beta = 0.1
gamma = 8
```

and in the four-argument call case the worst-case plant transfer function is also displayed as:

```
     60
gw = --------------------
     s (s + 0.1) (s + 8)
```

and saved in the hard disk.

3.22 Rootl

The command

```
> rootl,gj   (enter)
```

plots the root locus of $1+K\,gi(s)=0$ or $1+K\,gi(z)=0$ for $K\in[0,\infty]$.

3.22.1 Recall

Refer first to the continuous-time system $gi(s)=P(s)/Q(s)$. Let m and n be the degrees of $P(s)$ and $Q(s)$ respectively and assume $m\leq n$. The roots of the polynomial equation $Q(s)+K\,P(s)=0$ as long as K varies from 0 to ∞ describe a set of n lines, called the *branches* of the root locus. Every branch starts from a pole p_i [a root of $Q(s)$] and ends into a zero z_i [a root of $P(s)$] or goes to infinity, so that the root locus has $n-m$ branches going to infinity; their tangents at infinity, called *asymptotes*, all come from a single point of the real axis having abscissa

$$\sigma_a=\frac{1}{n-m}\left(\sum_{i=1}^{n}p_i-\sum_{i=1}^{m}z_i\right)\,,$$

and, if $gi(0+)>0$ and the number of zeros and poles with strictly positive real parts is even, form with the real axis the angles

$$\vartheta_{a,\nu}=\frac{(2\,\nu+1)\,\pi}{n-m}\quad(\nu=0,\,1,\,\ldots\,,\,n-m-1)\,,$$

while, if $gi(0+)>0$ and the number of zeros and poles with strictly positive real parts is odd, the angles are

$$\vartheta_{a,\nu}=\frac{2\,\nu\,\pi}{n-m}\quad(\nu=0,\,1,\,\ldots\,,\,n-m-1)\,.$$

These two sets of angles commute each other in case of positive feedback, i.e., if $gi(0+)<0$.

Note that the asymptotes divide the complex plane into equal sectors.

Program *rootl* applies also when $n<m$. This may happen when K denotes a parameter different from the loop gain. In this case the locus of $1/gi(s)$ versus $1/K$ is automatically considered, so the plot appears to be drawn counterwise.

A *branching point* is a point of the locus corresponding to a multiple root of the above equation, characterized by ingoing branches (in number equal to multiplicity) and an equal number of interposed outgoing branches. The branching points are obtained by solving the polynomial equation

$$P(s)\,Q'(s) - Q(s)\,P'(s) = 0 \ ,$$

where $P'(s)$ and $Q'(s)$ denote the derivatives of $P(s)$ and $Q(s)$ respectively. The corresponding values of K (both negative and positive) are obtained by substitution in the root locus equation.

The root locus of a discrete-time system $gi(z) = P(z)/Q(z)$ is likewise defined, and the rules for plotting and properties are the same. Nevertheless it has a different shape, due to the presence of a greater number of zeros, which causes a greater number of branches to remain in a finite zone of the complex plane.

Constant damping loci. It is convenient to plot constant damping lines with the root locus. Let us recall that the damping factor of a pair of complex conjugate poles $\sigma \pm j\omega$ is

$$\delta := \frac{-\sigma}{\sqrt{\sigma^2+\omega^2}} = \sin\beta \ ,$$

where β denotes the angle between the imaginary axis and the segment from the origin to $\sigma+j\omega$, positive counterclockwise. In the continuous-time case the constant damping locus corresponding to a given value of δ is the pair of rays from the origin defined by

$$s = -|\omega| \tan\beta \pm j\omega = -|\omega| \frac{\delta}{\sqrt{1-\delta^2}} \pm j\omega \ , \quad -\infty < \omega < \infty \ ,$$

while in the discrete-time case it is obtained by transforming the previous one in the correspondence $z = e^{Ts}$ and with ω restricted to the primary strip, i.e., $-\Omega/2 \le \omega \le \Omega/2$, where Ω denotes the sampling angular frequency. We obtain the pair of arcs of logarithmic spiral defined by

$$z = e^{Ts} = e^{(-|\omega| \tan\beta + j\omega)T} = e^{-\frac{2\pi|\omega|}{\Omega}\tan\beta}\, e^{j\frac{2\pi\omega}{\Omega}} \ , \quad -\Omega/2 \le \omega \le \Omega/2 \ ,$$

or

$$z = \ e^{-|\varphi| \tan\beta}\, e^{j\varphi} \ , \quad -\pi \le \varphi \le \pi \ .$$

3.22.2 Operation

On entering the command "rootl,gi", first a block diagram is shown to clarify the meaning of symbols and, when the return key is pressed, the following is displayed in the Command Window:

```
ROOT LOCUS :
x open loop poles (that are closed-loop poles for K=0)
o open loop zeros (that are closed-loop poles for K=Inf)
+ closed loop poles for K=1

the locus is drawn by steps for increasing values of K
or for decreasing values of K if the system is not causal

choose color of plot: k=black, g=green,
b=blue, r=red, y=yellow, m=magenta, c=cyan, default is green :
```

and when an admissible color has been selected, the current graphic window is cleared, shown at full size, and, after some delay for computations, the root locus diagram is shown while being plotted, thus giving insight on its behavior versus K. If the transfer function introduced is non-causal, the following message appears:

```
**** warning: gi is non-causal
     the root locus will be drawn in the reverse direction
```

When the locus has been drawn in the current graphic window, on pressing the return key the main menu appears in the Command Window as:

```
MENU :

 1 - change the reference axes and plot the last function again
 2 - grid on
 3 - information on branching points and asymptotes
 4 - plot another function in different color
 5 - recover the figure
 6 - information on a point of the locus with mouse
 7 - grid with constant damping loci (on/off)
 8 - show asymptotes in the plot
 9 - display the pole layout for a particular value of K
10 - change the step size and plot the last function again

 enter your choice (press return to exit) :
```

We now provide a brief description of the items of the above menu.

Change the reference axes and plot the last function again. This option allows selection of ranges and number of divisions for both axes. The default value for range is the previous one while the number of divisions, if not specified, is automatically determined. A typical interactive selection of new reference axes in the continuous-time case is the following:

```
the x-axis interval is [-5 4];
press return to maintain or enter new values: [xm xM] = [-2 2]
number of divisions for x-axis: 9;
press return to plot again with the previous scale
or type a new value: ndx =

the y-axis interval is [-3 3];
press return to maintain or enter new values: [ym yM] = [-1.5 1.5]
number of divisions for y-axis: 6;
press return to plot again with automatic scaling
or type a new value: ndy =
```

When the new reference axes have been defined, the figure is drawn again with the root locus of the last function.

Grid on. Draws the figure again with a grid referred to the axes divisions. In subsequent menus option 2 appears as:

```
2 - grid off
```

and allows recovering of figure without grid. The two types of option 2 toggle.

Information on branching points and asymptotes. This option provides information on the most important parameters characterizing the root locus, which are displayed as:

```
BRANCHING POINTS :
value: -0.4869 +0*j ;          gain: 0.05942

ASYMPTOTES :
center: -3.667;   angles: 60
                          180
                          300

**** press any key to continue
```

and, when any key is pressed, the main menu is shown again.

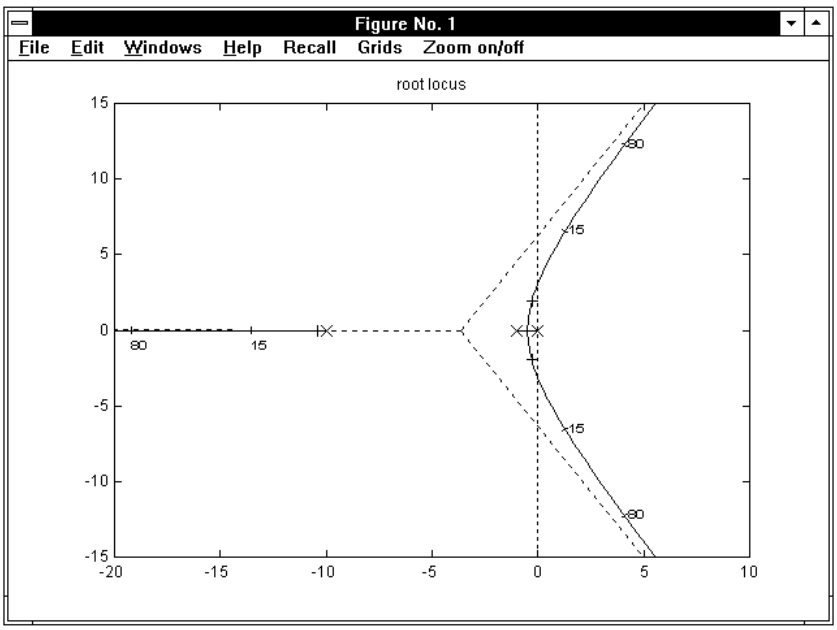

Figure 3.83. A root locus with graduation and asymptotes added.

Plot another function in different color. This option makes it possible to plot several graphs, referring to different transfer functions, in the same figure in different colors. The corresponding interactive request is:

```
enter transfer function : g2

choose color of plot: k=black, g=green,
b=blue, r=red, y=yellow, m=magenta, c=cyan, default is green : r
```

Continuous and discrete-time root loci cannot be plotted in the same figure, since the corresponding complex planes are different. This is avoided by a check in the program that produces the message:

```
**** error: continuous & discrete time in the same plot
```

pressing the return key after this message recovers the main menu.

Recover the figure. This option produces recovering of the current figure from the Command Window by using the keyboard instead of the mouse.

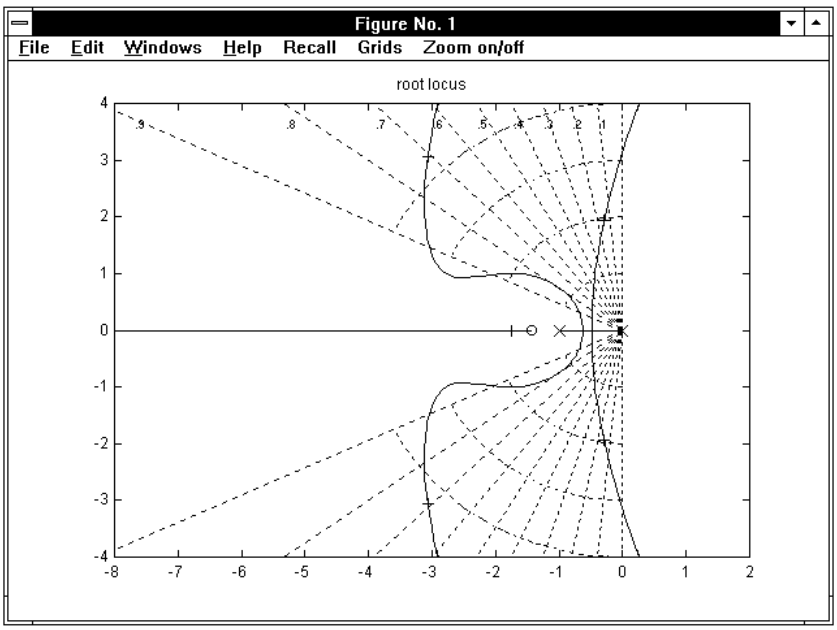

Figure 3.84. Root loci with constant damping loci added.

Information on a point of the locus with mouse. This option first produces in the Command Window the following request:

```
**** press return to enable selection
```

When the return key is pressed, the current figure is accessed from the Command Window and the mouse is activated for selection of a particular point of the graph. If more than one loci have been drawn, the figure appears as shown in Fig. 3.85, with the message:

Choose a color or MENU to exit (a)

in the top left corner, and a push-button menu that allows selection of a particular plot by color or exiting the information-with-mouse session (thus recovering the Command Window with the main menu), while, if only a root locus is present in the figure, the message (a) is replaced by:

Choose the color or MENU to exit (b)

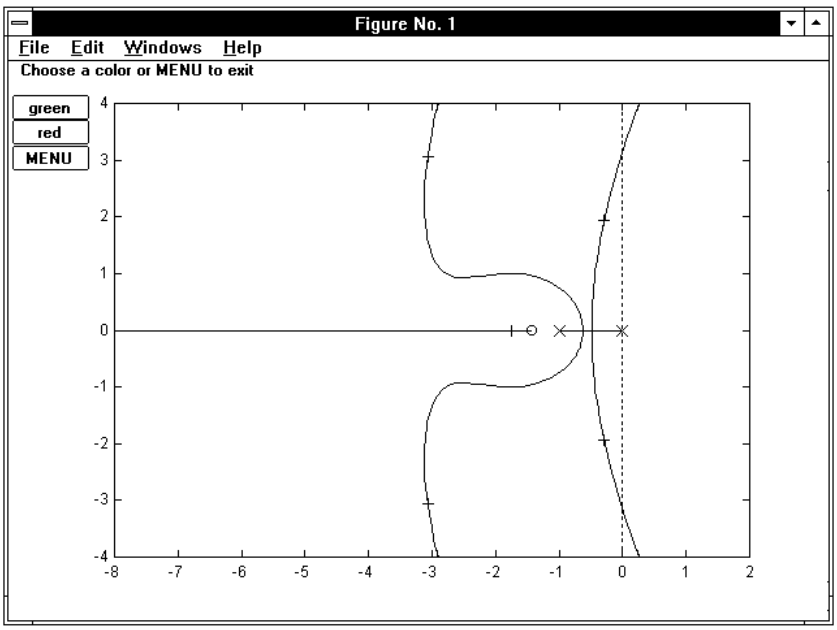

Figure 3.85. Root loci during the "information with mouse" session.

In the particular case in Fig. 3.85 there are two root loci, one green and the other red. When a color, for instance red, is selected with the mouse on the push-button menu, message (a) is replaced by:

Select a point with mouse (button 1) on the red plot (button 2 to change plot or exit) (c)

and, after selection has been performed, a small arrow is drawn on the corresponding plot in the same color in the direction of increasing gain to point out location of the point selected, as shown in Fig. 89. Furthermore, information on the coordinates of the selected point, the corresponding value of gain, and natural frequency and damping factor appears in the top right corner of the figure, while the message:

Click again button 1 to delete the displayed data (d)

is shown in the top left corner. When button 1 is clicked, message (c) is displayed: thus, selection of a point on the red plot with button 1 or return to the push-button menu with button 2 are enabled again.

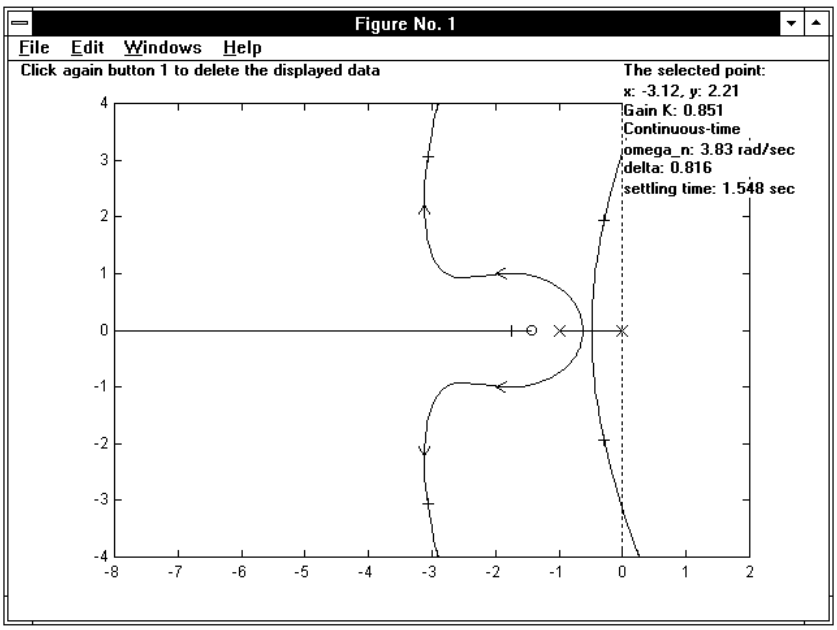

Figure 3.86. Root loci during the "information with mouse" session.

When **MENU** is selected to exit, the message:

PRESS RETURN TO RECOVER THE MAIN MENU

is displayed in the bottom left corner. Pressing the return key causes return to the Command Window with the request

```
maintain the arrows ? (1) :
```

that is useful when the figure has to be printed, since arrows provide information on the root locus behavior, thus completing the figure. Then, the main menu is shown again.

Grid with constant damping loci (on/off). When this option is selected, the root locus is shown again with some constant damping factor lines. These turn out to be very useful to approximately evaluate (referring only to the dominant poles) how the value of K affects the maximum overshoot in the step response and the resonance peak in the frequency response of the closed loop system. The values of the damping factor range from 0 to 0.9 with step 0.1, both for continuous and discrete-time systems. Some constant natural frequency loci are also plotted.

Show asymptotes in the plot. With this option the locus is shown again with the asymptotes added in the same color. If more than one loci are plotted in the same figure, selection by color is requested with

```
select function by entering color :
```

Display the pole layout for a particular value of K. If more than one loci are present, this option first produces the above request of selection by color, then:

```
type a K for closed loop poles (return to skip) : 5
```

When a value for K is entered, the following display appears:

```
POLES                    ABS. VALUE      DELTA
-11.62                   +11.62          +1
+0.3103  +4.137*j        +4.149          -0.07479
+0.3103  -4.137*j        +4.149          -0.07479

the poles locations will be pointed out with ticks
do you also want the value of K to be shown ? (1) :
```

If the return key is pressed, the figure is drawn again with the poles locations shown with ticks on the selected locus, while if 1 is entered, the value of K is also displayed as shown in Fig. 3.83. In the latter case the font size is also interactively requested. On going back to the Command Window by the return key, the request for a value of K, referring to the particular plot already selected by color, is repeated, and pressing the return key again without any new value typed causes the main menu to be recovered.

Change the step size and plot again the last function. Selection of the step size is automatically performed in the program for the best compromise between accuracy and computational time. However, in some cases the shape of the plot may be irregular. In these cases by using this option the last locus in the figure can be drawn again with smaller step size. A smaller step size can also be used to obtain slow drawing, thus achieving insight on the proceeding of the locus as K increases.

It produces the following interactive request:

```
the actual step size is: 1
press return to replot with the same step
or enter a new value : .4
```

and the last locus is plotted again in the already selected color more slowly and with a more regular shape.

3.22.3 Examples

Fig. 3.83 shows the root locus of the transfer function

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ ,$$

with the asymptotes added (option 8 of the main menu) and with some points graduated versus K (option 9). The imaginary axis, that represents the boundary of stability domain, is also shown with a dotted line, that is plotted when the locus is complete. Suppose that system $gi(s)$ is cascaded with the lead compensator

$$gc(s) = \frac{10\,(s+1.413)}{s+14.13}\ ,$$

and denote by $gh(s) := gc(s)\,gi(s)$ the overall transfer function. In Fig. 3.84 the root loci of $gi(s)$ and $gh(s)$ are plotted together (option 4) with constant damping loci (option 7) and the action of the compensator on the dominant pole locations is clarified. Fig. 3.85 has been obtained from Fig. 3.84 by deleting the constant damping loci (option 7) and enabling selection of points by mouse (option 6), while Fig. 3.86 shows the screen layout during selection.

As an example for the discrete-time case, we consider the transfer function

$$gj(z) = \frac{0.03279\,(z+0.1442)\,(z+2.342)}{(z-0.1352)\,(z-0.8187)\,(z-1)}\ ,$$

obtained from $gi(s)$ by conversion from continuous to discrete according to the zero-hold equivalence and sampling time $T = 0.2$ sec. Fig. 3.87 shows the corresponding root locus. The unit circle is shown instead of the imaginary axis when the plot (that is also drawn by steps) is complete. Fig. 3.88 shows again the root locus of $gj(s)$ with different axes (option 1) and the constant damping factor lines added (option 7).

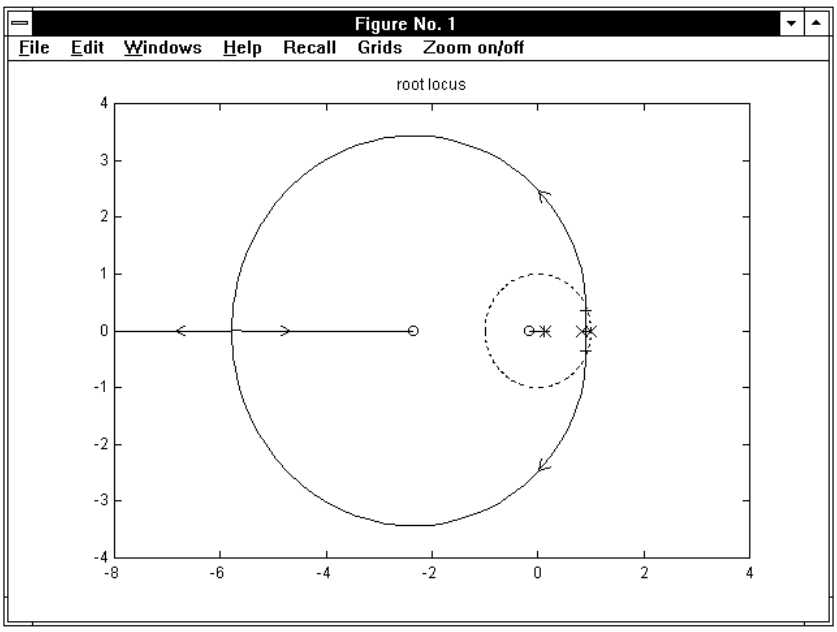

Figure 3.87. A root locus in the discrete-time case.

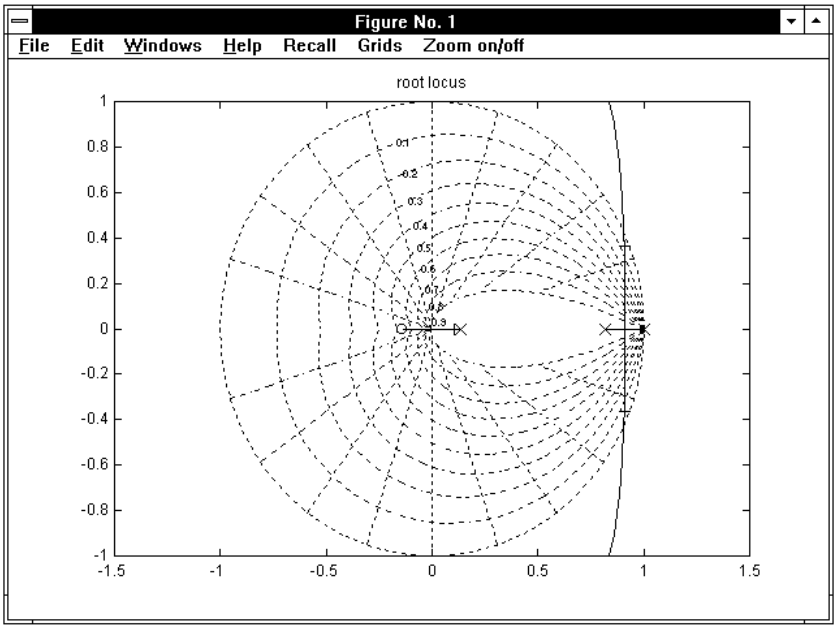

Figure 3.88. A discrete-time root locus with constant damping loci added.

3.23 Routh

The command

`> routh,gi` *(enter)*

computes and displays the stability intervals of system $gi(s)$ or $gi(z)$ in unit feedback connection in terms of the loop gain K.

3.23.1 Recall

Let us consider a linear dynamic system with the characteristic equation

$$a_n\, s^n + a_{n-1}\, s^{n-1} + \ldots + a_0 = 0\ . \tag{3.37}$$

It is well known that in order that the system be strictly stable the real parts of all the roots of the above equation must be strictly negative. The Routh criterion gives information about stability through a straightforward test on the coefficients of the characteristic equation, thus bypassing its complete solution. Although this use may be considered out-of-date by now, due to availability and wide circulation of computational routines that provide easy and fast solution of polynomial equations, the Routh criterion is still adopted in control theory to determine the stability intervals as functions of parameters on which the coefficients of the characteristic equation depend.

For the sake of simplicity we shall first consider the case where the coefficients are constant. Furthermore we shall assume that a_n is positive and a_0 is nonzero without any influence on generality, since the member on the left of (3.37) may be multiplied by -1 without affecting the roots and divided by a power of s, thus directly eliminating one or more roots equal to zero.

Let us build the *Routh table*:

n	a_n		a_{n-2}		a_{n-4}	$\ldots$
	$\downarrow$	$\nearrow$	$\downarrow$	$\nearrow$	$\downarrow$	
$n-1$	a_{n-1}		a_{n-3}		a_{n-5}	$\ldots$
$n-2$	γ_{n-2}		γ_{n-4}		$\ldots$	
$\ldots$	$\ldots$		$\ldots$			

The first two rows of the table report the polynomial coefficients arranged as shown, beginning from the highest power.

The elements of the third row are defined by

$$\gamma_{n-2} = \frac{a_{n-1}\, a_{n-2} - a_n\, a_{n-3}}{a_{n-1}} \ , \quad \gamma_{n-4} = \frac{a_{n-1}\, a_{n-4} - a_n\, a_{n-5}}{a_{n-1}} \ , \dots \ , \tag{3.38}$$

while those of the subsequent rows are likewise defined as functions of the two preceding rows. The rows of the table are non-increasing in length from top downwards and are marked on the left with the numbers n, $n-1$, ... : the last row, marked with 0, contains a single element.

The Routh criterion is stated in the following terms: to every sign change of the elements in the first column of the table corresponds a root with positive real part, while to every sign permanency corresponds a root with negative real part.

It is not strictly necessary to perform the divisions shown in (3.38) while building a new row, provided the signs of all the elements of the row are changed if the first element of the upper row is negative. When one or more elements at the beginning of a row are zero or a row is completely zero it is not possible to continue the table according to the above rule. However, it is possible to derive complete information on the signs of the real parts of all the roots by the following methods.

1. When the first h elements of a row are zero, sum to it the same row shifted by h positions on the left and multiplied by -1^h, then continue the table.

2. When a row is completely zero, it is always marked on the left with an odd number, for instance $2m+1$. In this case, build the *auxiliary equation*

$$\gamma_{2m}\, s^{2m} + \gamma_{2m-2}\, s^{2m-2} + \ldots + \gamma_0 = 0 \ ,$$

whose roots are those of the original equation not yet considered in the Routh table, derive the polynomial on the left of the auxiliary equation with respect to s, insert the coefficients of the obtained polynomial in place of the zero row considered, and continue the table. However, after these operations the interpretation of the Routh array is different: to every sign change in the first column of the table corresponds a root with positive real part and to every sign permanency corresponds a root with negative *or zero* real part. Since the auxiliary equation is lacking in odd powers of s (so that its roots are symmetric with respect to the origin by pairs), information on the sign of the real parts of all the roots of the original equation is still complete.

The above holds for polynomials whose coefficient values are all known real numbers. However, using the Routh criterion to know only whether a completely given system is stable or not is rather obsolete since there are a lot of computer programs that efficiently calculate all the roots of high-order polynomials in the complex field. For instance, in the Matlab environment the command *roots(p)* displays the roots of the polynomial whose coefficients, real or complex, are given in vector p, while in the TFI environment $gi =$ produces display of transfer function $gi(s)$ or $gi(z)$ in a factorized form where the real and imaginary parts of all zeros and poles explicitly appear. On the other hand, the program *routh* computes the stability intervals of a polynomial whose coefficients linearly depend on a parameter K. Consider the polynomial equation $1+K\,gi(s)=0$, that can be written in expanded form as

$$a_n\,s^n + a_{n-1}\,s^{n-1} + \ldots + a_0 + K\,(b_m\,s^m + b_{m-1}\,s^{m-1} + \ldots + b_0) = 0\ , \qquad (3.39)$$

where the first and the second polynomial on the left are the denominator and the numerator of $gi(s)$, respectively. Clearly a certain number of the coefficients of the polynomial equation are linear functions of K of the type $\alpha K+\beta$. The computation of the stability intervals in K is performed as follows:

1. the complete Routh table is obtained by using (3.38) without dividing by a_{n-1} on the right, thus obtaining an array of polynomials in K;

2. the roots of all the polynomials in the first column of the table are determined, and all the real ones are ordered by increasing values;

3. the bounds of every stability interval belong to this set since the sign of a polynomial in the first column is changing; hence, computing the roots in (3.39) for an intermediate value of K of the sequence determined at step 2 reveals whether the corresponding interval is stable or not;

4. contiguous stability intervals are grouped together, and form the stability intervals displayed.

The values of m and n in (3.39) are not restricted by the causality assumption ($m \leq n$). When the system under consideration is discrete-time, the transfer function $gi(z)$ is converted to its w-plane equivalent before applying the Routh criterion (see the *Recall* section of application *wplane*).

3.23.2 Operation and Examples

Let us consider the transfer function

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ .$$

Entering "routh,gi" first produces a medium-size figure with the block diagram of the closed loop system considered, and in the Command Window the display:

```
**** press return to continue
```

Upon pressing the return key, the stability interval in K appears as follows:

```
characteristic equation:   1 + K gi(s) = 0 ;

stable for K in the range: 0 to 2.75
```

As an example for the discrete-time case, let us refer to

$$gj(z) = \frac{0.3279\,(z+0.1442)\,(z+2.342)}{(z-0.1353)\,(z-0.8187)\,(z-1)}\ .$$

The stability interval in K is displayed as:

```
characteristic equation:   1 + K gj(z) = 0 ;

stable for K in the range: 0 to 1.34
```

When the stability intervals are more than one, they are all listed in the display. For instance, for the transfer function

$$gw(s) = \frac{40\,(s+40)\,(s+50)}{s\,(s+1)\,(s+10)}$$

we obtain the following result:

```
characteristic equation:   1 + K gw(s) = 0 ;

stable for K in the range: 0 to 0.002778
stable for K in the range: 0.275 to infinity
```

3.24 Samptime

The command

`> samptime[,T]` *(enter)*

defines or changes the current value of the sampling time used in conversions from continuous to discrete time and in the w-plane equivalences.

`> samptime,gi` *(enter)*

displays the sampling time of the discrete-time transfer function $gi(z)$.

3.24.1 Recall

Let us recall that the TFI environment always has a current sampling time, that must be defined when it is entered from Matlab the first time and permanently saved in file *time#.mat*, located in the TFI work directory.

The sampling time is used in applications *convert* and *wplane*. It is also stored with any discrete-time transfer function when it is saved in the hard disk. When a transfer function with a sampling time different from that of the TFI environment is called in an application, a warning message is displayed. If, for instance, the current sampling time is 0.2 sec and the transfer function $gi(z)$ was saved under a sampling time of 0.1 sec, the message:

```
**** warning: the sampling time of gi(z) is 0.1 sec

  **** press any key to continue
```

appears in the Command Window. Nevertheless, $gi(z)$ can be used in the current environment. In applications *fresp* and *tresp* (with continuous time axis) the frequency and step responses of discrete-time transfer functions with different sampling times can be consistently plotted together. To convert a discrete-time transfer function $gi(z)$ having a different sampling time to the current sampling time it is sufficient to enter:

`> gi=gi` *(enter)*

3.24.2 Operation

If *samptime* is called without any argument, we obtain the following display:

```
the current sampling time is 0.2 sec
enter a new value (default is "no  change") : .1

the current sampling time is 0.1 sec
```

while, if a numeric argument is specified, for instance as:

```
> samptime,.1   (enter)
```

the display appears as follows:

```
the current sampling time is 0.1 sec
```

The application *samptime* can also be used to display the sampling time of a discrete-time transfer function. In fact, entering:

```
> samptime,gi   (enter)
```

produces:

```
the sampling time of gi(z) is 0.1 sec
```

while, if the transfer function is continuous-time, the message:

```
transfer function gi(s) is continuous-time
```

is obtained.

3.25 Select

The command

`> select,gi,gj` *(enter)*

builds a new transfer function $gj(s)$ or $gj(z)$ by interactive selection of factors from $gi(s)$ or $gi(z)$.

3.25.1 Recall

Let us consider the transfer function in factorized form

$$gi(s) = \frac{400\,(s+1.453)}{s\,(s+1)^2(s+10)\,(s+14.53)}\ .$$

Program *select* makes it possible to select any factor from the numerator and denominator of $gi(s)$ and to set arbitrarily the dc gain. The result is saved as the new transfer function $gj(s)$. The selection is interactively performed by entering the numbers, in progressive order, of the factors to be included in the output. If a factor is raised to a power, it is counted as many times as the power is. It is possible to maintain the dc gain of the original transfer function, to maintain that of the selection made or to interactively specify a new dc gain for the selection. The dc gain is herein defined as the value of the transfer function $g(s)$ for $s=0$ (leaving out of consideration the possible poles at $s=0$) or the value of the transfer function $g(z)$ for $z=1$ (leaving out of consideration the possible poles at $z=1$).

Usefulness of *select* is related to computations where it is necessary to have perfect cancellation of factors, like in some analytical design methods. For instance, selection of factors 2 from numerator and 1,4 from denominator of the above $gi(s)$, while preserving the dc gain of selection, results in

$$gj(s) = \frac{(s+1.453)}{s\,(s+10)}\ .$$

Since the coefficients are generally rounded while displayed, in the quotient $gi(s)/gj(s)$ the selected factors certainly cancel, since they are equal to all the stored digits. On the contrary, if a transfer function having some factor apparently equal to those of $gi(s)$ is entered with the keyboard, cancellation may not occur.

3.25.2 Operation and Examples

In the above case, entering "select gi,gj" produces:

```
     400 (s + 1.453)
gi = --------------------------------
     s (s + 1)^2 (s + 10) (s + 14.53)

   numerator:
   select factors 1,2,..; * for all; return to skip : 2

   denominator:
   select factors 1,2,..; * for all; return to skip : 1,4

   the dc gain of the original transfer function (gi) is: 4
   the dc gain of the selected transfer function (gj) is: 0.1453

   1 - maintain the dc gain of the original transfer function
   2 - maintain the dc gain of the selected transfer function
   3 - specify a new dc gain

   your choice : 2

     1 (s + 1.453)
gj = -------------
     s (s + 10)
```

Let us check out cancellation of factors. Entering:

```
 > gi/gj   (enter)
```

produces:

```
ans=gi/gj
.........
--------------------
.....................

      400
ans = ----------------------
      (s + 1)^2 (s + 14.53)
```

Let $gk(s)$ be a transfer function entered with the keyboard, by typing:

```
 > gk=(s+1.453)/(s*(s+10))   (enter)
```

```
gk=(s+1.453)/(s*(s+10))
.......................
--------------------
.....................

     1 (s + 1.453)
gk = -------------
     s (s + 10)
```

We can check that in this case cancellation is not obtained, since the coefficient 1.453 is not the real stored value, but its rounding. In fact, introduction of:

```
 > gi/gk   (enter)
```

yields:

```
ans=gi/gk
.........
--------------------
...................

      400 (s + 1.453)
ans = ---------------------------------
      (s + 1)^2 (s + 1.453) (s + 14.53)
```

Program *select* can also be profitably used to set the dc gain of a transfer function $gi(s)$ or $gi(z)$ to any desired value. To obtain this result, simply type

```
 > select,gi,gi   (enter)
```

then, select all the factors of the numerator and denominator by entering `*`, and use option 3 for the gain.

3.26 Startint

The command

```
> startint  (enter)
```

makes it possible to change some TFI environment settings.

3.26.1 Recall

The TFI environment settings accessible through *startint* are:

1 - the work directory in the TFI environment;
2 - possible reduction of the matlabpath in the TFI environment;
3 - selection of black or white background in figures;
4 - selection of permanent storing of figure locations;
5 - activation of legend in figures.

Change of settings may be advisable for the following reasons.

1 - The work directory of TFI is the directory where all the transfer function files are saved. In the first run of TFI it is assumed by default to be the same as the Matlab work directory, but it can easily be changed to avoid unnecessary mixing of different environments. Of course, the new directory must exist (if not, a message is displayed and the work directory is not changed). For instance, it is possible to use *c:\matlab\work* for Matlab (specified as a property of the Matlab icon in Windows) and *c:\matlab\workint* for TFI. The Matlab work directory is restored when TFI is quitted.

2 - At present it is not necessary to reduce the matlabpath when entering the TFI environment. This option may be used if future toolboxes contain some files with the same names as those of the *intp* directory. In this case path reduction and location of *intp* after other directories in the matlabpath eliminate conflict. The names of the toolboxes strictly necessary in the TFI environment, specified in the file *startint.m*, are the standard ones plus *intp*, i.e., for the Matlab 4 version, \local, \general, \ops, \lang, \elmat, \elfun, \specfun, \matfun, \datafun, \polyfun, \sparfun, \plotxy, \plotxyz, \graphics, \strfun, \intp, and, for the Matlab 5 version, \general, \ops, \lang, \elmat, \elfun, \specfun, \matfun, \datafun, \polyfun, \sparfun, \graph2d, \graph3d, \specgraph, \graphics, \uitools, \strfun, \iofun, \timefun, \datatypes, \local, \intp. These names can be changed if necessary by editing *startint.m*.

3 - The background color of the figures can be set to black, whose advantages are less radiation and very clean colors in ink-jet prints, or to white, that is more similar to other Windows applications, like Word.

4 - When a high-resolution monitor is used, the figures are often located with the mouse to achieve overall view while running applications that open many figures. The figure locations can be permanently stored for subsequent TFI sessions by using this option, thus avoiding repetition of adjustment with mouse.

5 - The utility *legend* available in Matlab specifies, in the figures, the names of the transfer functions referring to the colors of the corresponding plots. It generates a small window that can be moved by the mouse to the most convenient location. Since the current version of *legend* is quite time-consuming, particularly in the Matlab 4 version, its use is left as an option that can be switched on/off with *startint*.

The above settings are permanently saved in the file *set_tfi#.mat* or *set_tfi5.mat*, located in the work directories of both Matlab and TFI.

3.26.2 Operation

Let us reproduce here, as a typical use of *startint*, the interactive session that specifies the work directory of TFI, that should be done when TFI is called the first time after installation. Entering

```
> startint   (enter)
```

produces the following display:

```
TFI was called from and will exit to C:\MATLAB\WORK
at present the TFI work directory is C:\MATLAB\WORK
to change the TFI directory enter its path
(default no change) : C:\MATLAB\WORKINT

you can reduce the matlabpath to avoid possible conflict of names
with new toolboxes; the matlabpath IS NOT REDUCED
do you want to reduce the matlabpath ? (1/0, default "no change") :

the figure background IS BLACK
black or white figure background ? (1/0, default "no change") : 0

the figure locations ARE NOT STORED for next TFI sessions
do you want to store them ? (1/0, default "no change") :

legend in figures IS NOT ACTIVATED (legend is quite time-consuming)
do you want legend in figures ? (1/0, default "no change") :
```

When some TFI settings have been changed with *startint*, it is advisable to exit and enter TFI again, to be sure that the new settings are stored in both the Matlab and TFI work directories.

3.27 Tfeval

The command

`> tfeval,gi` *(enter)*

computes and displays the value assumed by the transfer function $gi(s)$ or $gi(z)$ for any interactively given value of s or z.

3.27.1 Operation

Let us consider the transfer functions

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ ,$$
$$gj(z) = \frac{0.3279\,(z+0.1442)\,(z+2.342)}{(z-0.1353)\,(z-0.8187)\,(z-1)}\ .$$

Entering "tfeval,gi" produces interactive request of the value of the variable s (any complex number) and display of the corresponding value of $gi(s)$ as follows:

```
enter a value for s (return to exit) : 4

value of gi(4) : 0.1429
abs.value: 0.1429; angle: 0 degrees (0 radians)

enter a value for s (return to exit) : 2*j

value of gi(2i) : -0.8462-0.2308i
abs.value: 0.8771; angle: -164.7 degrees (-2.875 radians)

enter a value for s (return to exit) :
```

Likewise, in the discrete-time case, "tfeval,gj" produces:

```
enter a value for z (return to exit) : 4

value of gj(4) : 0.02337
abs.value: 0.02337; angle: 0 degrees (0 radians)

enter a value for z (return to exit) : 2*j

value of gj(2i) : -0.02006+0.005877i
abs.value: 0.0209; angle: 163.7 degrees (2.857 radians)

enter a value for z (return to exit) :
```

TFI provides a quicker access to the application *tfeval* by simply entering the value of the argument in round brackets after the transfer function name. In this case the display in the Command Window appears as follows:

```
> gi(4)  (enter)

  value of gi(4) : 0.1429
  abs.value: 0.1429; angle: 0 degrees (0 radians)

> gi(2*j)  (enter)

  value of gi(2i) : -0.8462-0.2308i
  abs.value: 0.8771; angle: -164.7 degrees (-2.875 radians)

> gj(4)  (enter)

  value of gj(4) : 0.02337
  abs.value: 0.02337; angle: 0 degrees (0 radians)

> gj(2*j)  (enter)

  value of gj(2i) : -0.02006+0.005877i
  abs.value: 0.0209; angle: 163.7 degrees (2.857 radians)
```

3.28 Tresp

The command

`> tresp,gi` *(enter)*

plots the step or impulse response of the system with transfer function $gi(s)$ or $gi(z)$, with choice between open loop or unit feedback.

3.28.1 Recall

The impulse response is obtained by computing the inverse $\mathcal{L}$-transform of $gi(s)$ [$gi(s)/(1+gi(s))$ in the closed loop case] or the inverse $\mathcal{Z}$-transform of $gi(z)$ [$gi(z)/(1+gi(z))$ in the closed loop case]. See the *Recall* section of application *invtr* for details. The step response is obtained by computing the inverse transform to the above transfer functions multiplied by $1/s$ in the continuous-time case or $z/(1-z)$ in the discrete-time case. Plots are obtained with linear interpolation of 600 samples in the continuous-time case and with stairstep graph of a maximum of 400 samples in the discrete-time case. The first plot is drawn with automatic scaling.

3.28.2 Operation

The input menu that appears in the Command Window after "tresp,gi" is:

```
1 - step response, open-loop
2 - step response, closed-loop
3 - impulse response, open-loop
4 - impulse response, closed-loop

enter your choice (default is 1) :
```

After the choice is performed, you must select the color of the plot according to the following request:

```
choose color of plot: k=black, g=green,
b=blue, r=red, y=yellow, m=magenta, c=cyan, default is green :
```

After an admissible color has been selected, the current graphic window is cleared, shown at full-screen size, and, after some delay for computations, the selected type of step reponse diagram is shown.

To go back to the Command Window and see the main menu, press return.

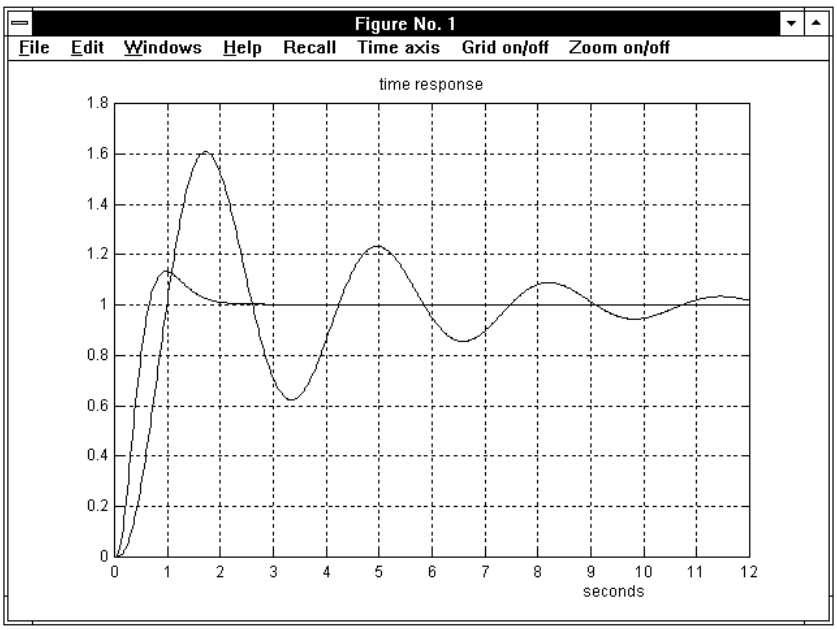

Figure 3.89. Two step responses in the continuous-time case.

Fig. 3.89 shows a typical screen layout in the continuous-time case. The main menu for continuous-time or mixed continuous and discrete-time systems is:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - information (on step response only)
4 - plot another function in different color
5 - recover the figure
6 - information on plot(s) with mouse

enter your choice (return to exit) :
```

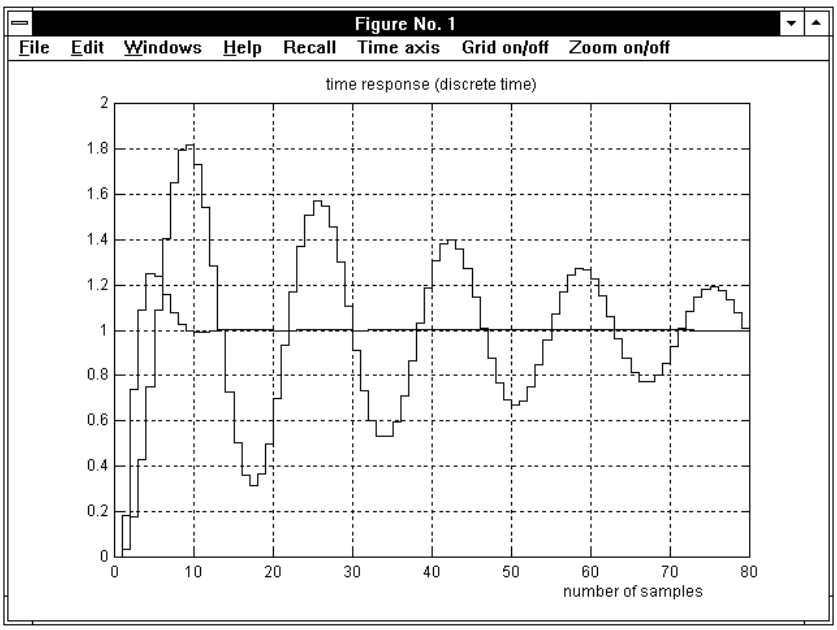

Figure 3.90. Two step responses in the discrete-time case.

Fig. 3.90 shows a typical screen layout in the discrete-time case. The main menu for discrete-time systems is:

```
MENU :

1 - change the reference axes and plot again
2 - grid on
3 - information (on step response only)
4 - plot another function in different color
5 - recover the figure
6 - information on plot(s) with mouse
7 - change from discrete to continuous time or vice versa

enter your choice (return to exit) :
```

We now provide a brief description of the items of the above menus.

Change the reference axes and plot again. This option allows selection of ranges and number of divisions for both axes. The default value for range is the previous one while the number of divisions, if not specified, is automatically determined. A typical interactive selection of new reference axes in the continuous-time case is the following:

```
the time interval is [0 25];
press return to maintain or enter new values: [t1 t2] = [0 10]
number of divisions for x-axis: 5;
press return to plot again with automatic scaling
or enter a new value: ndx =

the y-axis interval is [0 1.8];
press return to maintain or enter new values: [ym yM] =
number of divisions for y-axis: 9;
press return to plot again with the previous scale
or enter a new value: ndy =
```

while in the discrete-time case it appears as:

```
the interval of samples is [0 80];
press return to maintain or enter new values: [k1 k2] = [0 60]
number of divisions for x-axis: 8;
press return to plot again with automatic scaling
or enter a new value: ndx =

the y-axis interval is [0 2];
press return to maintain or enter new values: [ym yM] =
number of divisions for y-axis: 10;
press return to plot again with the previous scale
or enter a new value: ndy =
```

When new reference axes have been defined, the figure is drawn again.

Grid on. Draws the figure again with a grid referred to the axes divisions. In subsequent menus option 2 appears as:

```
2 - grid off
```

and allows recovering of figure without grid. The two types of option 2 toggle. A mouse-oriented **Grid on/off** facility is also provided in the command bar at the top of the figure.

Information (on step response only). This option provides information on the most important parameters of step responses that have been plotted; information appears in the Command Window. A typical example is:

```
STEP RESPONSE :
maximum overshoot: 13.31 percent at time: 0.9701 sec
delay time (to 50 percent): 0.3466 sec
rise time (from 10 to 90 percent): 0.4131 sec
settling time (to plus/minus 5 percent): 1.507 sec

STEADY-STATE ERRORS (CLOSED LOOP ONLY) :
step input steady-state error: 0
ramp input steady-state error: 0.25

press any key to continue
```

while in the discrete-time case we have:

```
STEP RESPONSE :
maximum overshoot: 81.74 percent at time: 9 samples (1.8 sec)
delay time (to 50 percent): 4 samples (0.8 sec)
rise time (from 10 to 90 percent): 3 samples (0.6 sec)
settling time non-computable: please enlarge the time scale

STEADY-STATE ERRORS (CLOSED LOOP ONLY) :
step input steady-state error: 0
ramp input steady-state error: 0.25

press any key to continue
```

Note that in the second case time is measured both in number of samples and seconds. If several plots have been drawn in the same figure in different colors (by using option 4 of the main menu), before display of information we have the following request:

```
select function by entering color :
```

that allows selection by color of the transfer function, particularly easy when a legend is available in the figure. If a legend is not present, the **Recall** facility in the command bar of the figure may be helpful.

If the function addressed is not a step response (it has been created with options 3 or 4 of the input menu) the following message is displayed:

```
**** error: not a step response

  press any key to continue
```

and, when a key is pressed, the main menu is displayed again.

Plot another function in different color. This option makes it possible to plot in the same figure several graphs in different colors, referring to different transfer functions. A typical use is for comparing time responses for different compensators. The corresponding interactive request appears as follows:

```
enter transfer function : g2
```

When a new transfer function is entered, the input menu is displayed again, to allow choice among different types of responses also for the new function. Then, we obtain the standard request:

```
choose color of plot: k=black, g=green,
b=blue, r=red, y=yellow, m=magenta, c=cyan, default is green :
```

The same function can be entered again with a different choice in a different color. The added plot is referred to the previous time axis (while the magnitude axis is automatically adapted if necessary), so that it is often necessary to use option 1 for the best compromise in the time axis selection. When working with discrete time axis and a time-continuous function is added, the following error message is displayed:

```
**** error: conflict between continuous & discrete-time
           hint: use option 7 to convert the time axis

  press any key to continue
```

On pressing a key, the main menu is recovered and option 7 can be used to change the current axis type.

Recover the figure. This option produces recovering of the current figure from the Command Window by using the keyboard instead of the mouse.

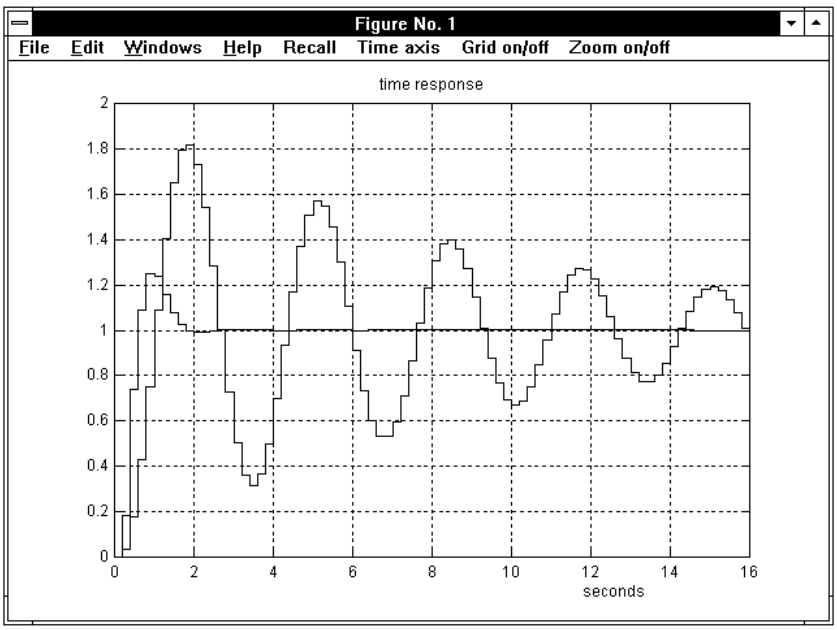

Figure 3.91. The discrete-time step responses with the time scale changed.

Information on plot(s) with mouse. This option first produces in the Command Window the following request:

```
**** press return to enable selection
```

When the return key is pressed, the current figure is accessed, the mouse is activated and the message:

Select a point with mouse (use button 1, button 2 to exit) (a)

is displayed in the top left corner of the figure. The aim of selection is to define a value of the time. In fact, when a point is selected, a vertical bar passing through it appears and the values of the time response plots in the figure corresponding to the intersections with the vertical bar are displayed, each in the color of the plot, in the top right corner, as shown in Fig. 3.92 The value of time is also displayed in orange. If both continuous and discrete-time plots are mixed in the figure, both the value of time and the number of samples are displayed. Selection with button 1 can be repeated, thus changing the abscissa of the vertical bar and, consequently, the displayed values.

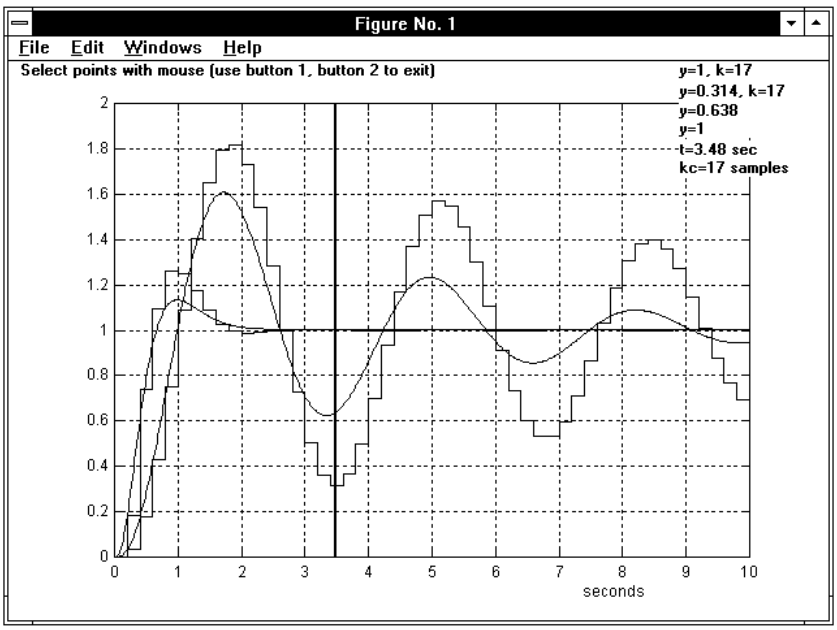

Figure 3.92. The "information with mouse" session.

When button 2 is pressed, the message:

PRESS RETURN TO RECOVER THE MAIN MENU

is displayed in the bottom left corner. Pressing the return key causes return to the Command Window with the main menu.

Change from discrete to continuous time or vice versa. This option allows switching from discrete time k to continuous time t with $t=k/T$ if all the plotted functions are discrete-time. In this case the reverse passage is also allowed. On the other hand, if some of the plotted functions are continuous-time (so that the time axis is restricted to be continuous), the option is not displayed. When the time axis is continuous, it is possible to plot together several discrete-time response functions created with different sampling times, possibly all different from the current environmental sampling time. In this case the values of **k** and **kc** appearing in the top right corner in Fig. 3.92 may be different.

3.28.3 Examples

Let us consider the unit-feedback control loop consisting of a lead compensator with transfer function $gc(s)$ and a plant with transfer function $gi(s)$, defined by:

$$gc(s) = \frac{10\,(s + 1.413)}{s + 14.13}\ , \quad gi(s) = \frac{40}{s\,(s + 1)\,(s + 10)}\ . \tag{3.40}$$

Let $gh(s) := gc(s)\,gi(s)$ be the open-loop transfer function. Fig. 3.89 shows the closed-loop step responses of $gi(s)$ and $gh(s)$; note that the compensator provides a significant improvement of the dynamic behavior, as can also be pointed out in quantitative terms by using option 3 of the main menu. In fact, the maximum overshoot from 60.8 per cent reduces to 13.3 per cent, while the settling time from 10.05 sec reduces to 1.50 sec.

Similarly as an example for the discrete-time case, we consider a discrete compensator with transfer function $gd(z)$ connected to the plant $gj(z)$ obtained from $gi(s)$ by conversion from continuous to discrete according to the zero-hold equivalence and sampling time $T = 0.2$ sec (see the TFI application *convert*). The transfer functions referred to are:

$$gd(z) = \frac{5.6\,(z - 0.72)}{z + 0.52}\ , \quad gj(z) = \frac{0.03279\,(z + 0.1442)\,(z + 2.342)}{(z - 0.1352)\,(z - 0.8187)\,(z - 1)}\ . \tag{3.41}$$

Let $gk := gd(z)\,gj(z)$. As in the continuous-time case, we plot the closed-loop step responses of $gj(z)$ and $gk(z)$ to point out the effect of the compensator. Fig. 3.90 shows these responses with respect to the discrete time axis, while Fig. 3.91, obtained with the option 7 of the main menu, shows the same plots referred to continuous time. Improvement in quantitative terms can be checked by using option 3: the maximum overshoot is reduced from 81.7 per cent to 25.8 per cent, the setting time is reduced from 134 samples (26.8 sec) to 8 samples (1.6 sec).

Fig. 3.92 shows all the above plots in one figure during an information-with-mouse session (option 6). In this case the time axis is continuous, since the transfer functions considered are of mixed types.

3.29 Wplane

The command

`> wplane,gi,gj` *(enter)*

converts a discrete-time transfer function $gi(z)$ to the w-plane function $gj(w)$, that is displayed and saved in the work directory as $gj(s)$, or a continuous-time transfer function $gi(s)$, considered as the w-plane function $gi(w)$, to $gj(z)$, that is displayed and saved in the work directory.

3.29.1 Recall

The direct and inverse w-plane conversions are defined by:

$$w = \frac{2}{T}\,\frac{z-1}{z+1} \quad \text{and} \quad z = \frac{1+\frac{wT}{2}}{1-\frac{wT}{2}}\ .$$

The basic properties of the w-plane transformation are:

1. the unit circle of the z plane is transformed in the left half w plane;
2. the w-plane equivalent transfer functions $gi(z)$ and $gj(w)$ have equal steady-state error constants.

Property 1 is proved as follows: let $w{=}\mu{+}j\nu$ and $z{=}u{+}jv$; we have

$$\begin{aligned} w \;=\; \mu + j\nu = \frac{2}{T}\,\frac{u+jv-1}{u+jv+1} &= \frac{2}{T}\,\frac{(u-1+jv)\,(u+1-jv)}{(u+1)^2+v^2} \\ = \frac{2}{T}\,\frac{(u^2+v^2-1)+j\,(-2v)}{(u+1)^2+v^2} &= \frac{2}{T}\,\frac{(|z|^2-1)+j\,(-2v)}{(u+1)^2+v^2}\ ; \end{aligned}$$

hence the real part of w is positive if and only if $|z|>1$. To prove property 2, recall that the first finite (neither zero nor infinite) error constant for discrete and continuous-time systems is respectively defined by

$$\lim_{z\to 1}\frac{1}{T^h}\,(z-1)^h\,gi(z) \quad \text{and} \quad \lim_{w\to 0} w^h\,gj(w)\ ,$$

where h denotes the order of the unit pole in the first case and that of the pole at the origin in the second case. We have

$$gi(z) = \frac{\overline{gi}(z)}{(z-1)^h}\ , \quad gj(w) = \frac{\overline{gj}(w)}{(w\,T)^h}\Big(1-\frac{w\,T}{2}\Big)^h\ ,$$

where $\overline{gi}(z)$ and $\overline{gj}(w)$ have respectively no unit and no zero poles and correspond with each other in the w-plane equivalence, so that $\overline{gi}(1){=}\overline{gj}(0)$.

3.29.2 Operation and Examples

The typical use of the w-plane conversion is to design the discrete-time feedback compensators and regulators by using the continuous-time techniques.

Let us refer to the *Examples* section of application *tresp* and consider the transfer function $gj(z)$ in (3.41), that is the discrete-time zero-order hold equivalent of $gi(s)$ in (3.40). Its w-plane transform $gw(w)$ is obtained with the command "wplane,gj,gw", that produces:

```
   transformation from discrete to continuous according to the
   direct wplane equivalence; the assumed sampling time is 0.2 sec

     0.00912 (s - 10) (s + 13.37) (s - 24.9)
gw = ---------------------------------------
     s (s + 0.9967) (s + 7.616)
```

The transfer function $gw(s)$ (that may more properly be considered a function of w instead of s) is used to derive a lead compensator $gcw(s)$, by using the application *leadc*, that works only in the continuous-time case. Let

$$gcw(s) = \frac{20.07\,(s + 1.628)}{(s + 31.67)}$$

be the derived compensator. By means of the reverse w-plane transformation we obtain a discrete-time compensator $gd(z)$ such that the open-loop transfer function $gd(z)\,gj(z)$ has the same stability margins as $gcw(s)\,gw(s)$. The corresponding command is "wplane,gcw,gd", that produces:

```
   transformation from continuous to discrete according to the
   inverse wplane equivalence; the assumed sampling time is 0.2 sec

     5.6 (z - 0.72)
gd = --------------
     (z + 0.52)
```

Let $gk(z) := gd(z)\,gj(z)$. Figs. 3.90 and 3.91 of application *tresp* show (and make it possible to compare) the step response of the non-corrected feedback system $gj(z)$ and that of the system $gk(z)$, corrected by using application *wplane* exactly as described.

3.30 Zpplots

The command

```
> zpplots,gi,gj,gk,gw   (enter)
```

plots the zero-pole maps of up to four transfer functions in different colors in the same figure.

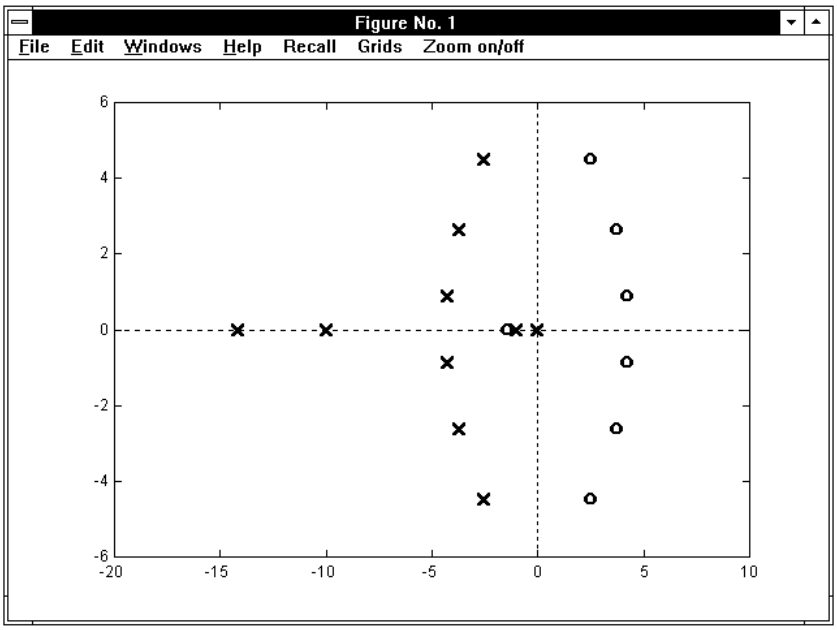

Figure 3.93. Some pole-zero layouts in the complex plane.

3.30.1 Operation

Let us consider the transfer functions

$$gi(s) = \frac{40}{s\,(s+1)\,(s+10)}\ ,$$
$$gj(s) = \frac{10\,(s+1.413)}{(s+14.13)}\ ,$$
$$gk(s) = \frac{s^6 - 21\,s^5 + 210\,s^4 - 1260\,s^3 + 4725\,s^2 - 10395\,s + 10395}{s^6 + 21\,s^5 + 210\,s^4 + 1260\,s^3 + 4725\,s^2 + 10395\,s + 10395}\ .$$

The command "zpplots,gi,gj,gk" provides the zero-pole maps of the above transfer functions, plotted together in the current graphic window as shown in Fig. 3.93.

The zero-pole maps are shown in the following color sequence: *green*, *red*, *cyan* and *yellow*, and can easily be distinguished by using the **Recall** facility available in the command bar of the figure, where the **Grids** facility is also present, making it possible to obtain the regular grid and/or the constant damping loci.

Continuous-time and discrete-time transfer functions cannot be mixed in the same call. In such a case, the following message appears in the Command Window:

```
**** error: mixed continuous and discrete time
```

TFI provides a quicker access to function *zpplots* by simply entering the transfer function name followed by a semicolon. Of course, this mode is restricted to a single transfer function map. Hence:

```
> gi;  (enter)
```

produces the zero-pole map of function $gi(s)$ or $gi(z)$. In this case the current graphic window is not overwritten, since a new figure is created and cancelled when the return key is pressed.

Bibliography

[1] **The MathWorks, Inc.,** *The Student Edition of Matlab*, Version 5 User's Guide, Prentice Hall, Upper Saddle River, NJ 07458, 1997.

[2] **D' Azzo and Houpis, C. H.,** *Linear Control Systems Analysis and Design: Conventional and Modern*, 4th Edition, McGraw-Hill, New York, 1995.

[3] **Dorf, R. C. and Bishhop, R. H.,** *Modern Control Systems*, 8th Edition, Addison-Wesley, Reading, Mass., 1998.

[4] **Franklin, G. F., Powell, J. D. and Emami-Naeini, A.,** *Feedback Control of Dynamic Systems*, 3rd Edition, Addison-Wesley, Reading, Mass., 1994.

[5] **Franklin, G. F., Powell, J. D. and Workman, M. L.,** *Digital Control of Dynamic Systems,* 3rd Edition, Addison-Wesley, Reading, Mass., 1998.

[6] **Kuo, B. C.,** *Automatic Control Systems*, 2nd Edition, Prentice-Hall, Englewood Cliffs, N.J., 1995.

[7] **Kuo, B. C.,** *Digital Control Systems*, Saunders College Publishing, Ft. Worth, 1992.

[8] **Ogata, K.,** *Modern Control Engineering*, 3rd Edition, Prentice-Hall, Englewood Cliffs, N.J., 1997.

[9] **Ogata, K.,** *Discrete-Time Control Systems*, 2nd Edition, Prentice-Hall, Englewood Cliffs, N.J., 1995.

[10] **Schultz, D. G., Melsa, J. L. and Rohrs, C. E.,** *Linear Control Systems*, McGraw-Hill, New York, 1993.

[11] **Marro, G.,** *Controlli Automatici*, 4th Edition, Zanichelli, Bologna, Italy, 1997.

[12] **Marro, G.,** *Complementi di Controlli Automatici: controlli digitali e approccio nello spazio degli stati*, Zanichelli, Bologna, Italy, 1994.

Index