Int. J. Appl. Engng Ed. Vol. 7, No. 5, pp. 392–397, 1991
Printed in Great Britain.

0742–0269/91 $3.00+0.00

Computer Simulation for Teaching System Stability Concepts*

MD. SAMIN WAHID
ALI M. EYDGAHI
State University of New York, Department of Electrical Engineering, New Paltz, NY 12561, U.S.A.

Determining the stability of a system, either analog or digital, is a very important subject in Control Systems courses. A computer simulation for teaching system stability concepts, called SYSTA, has therefore been developed as a teaching aid in the learning of the stability of a system. The software assists the students in understanding how the stability of a system is determined and how it is affected by the parameters in the characteristic equation. The software is menu driven and user friendly, and a manual thus seems unnecessary. The program is written in Turbo C version 2.0 and may be run on any IBM personal computers or compatibles with CGA, EGA, or VGA monitors. This program also provides support for a Microsoft mouse or compatibles.

INTRODUCTION

ALMOST ALL books on Control Systems courses talk about the stability of a system, either analog or digital. By definition, a system is said to be stable if for all bounded inputs there are corresponding bounded outputs. There are many approaches to determine the stability of a system. For analog or linear continuous-time systems (s-domain), three approaches are used in the software: (1) The Routh–Hurwitz stability criterion [1–3]. (2) Roots analysis (determining the location of the roots of the characteristic equation on the s-plane) [1–3]. (3) Graphical plot of the roots on the s-plane.

For digital or linear discrete-time systems (z-domain), however, the following three approaches are used in the software: (1) The Jury stability criterion [2, 3]. (2) The magnitude and angle of the roots of the characteristic equation on the z-plane. (3) Graphical plot of the roots on the z-plane.

From the students' point of view, it is not too hard to understand how the stability of a system is determined as long as the concepts are understood. A computer simulation program for teaching system stability concepts, SYSTA, has therefore been developed to make the students understand the processes behind the stability of the systems by manipulating the approaches listed above. The program is menu driven and highly interactive, so that even the first-time user can use it with ease. Its mouse capability makes it even more convenient. It may be run on any IBM PCs or compatibles equipped with either CGA, EGA, or VGA monitors.

* Paper accepted 20 May 1991.

SYSTEM STABILITY IN THE s-DOMAIN

The stability requirement may be defined in terms of the location of the roots or the poles of the closed-loop transfer function. The closed-loop transfer function may be written as

$$T(s) = \frac{p(s)}{q(s)}$$

$$= \frac{K\prod_{i=1}^{M}(s+z_i)}{s^N \prod_{k=1}^{Q}(s+\sigma_k)\prod_{m=1}^{R}(s^2+2\zeta_m\omega_m s+\omega_m^2)} \quad (1)$$

where $q(s)$ is the characteristic equation whose roots are the poles of the closed-loop system.

Routh–Hurwitz criterion

The first method used by this program is the Routh array and the Routh–Hurwitz criterion [1, 4]. This method gives the absolute stability of a system. The necessary condition for this method is that all of the coefficients of the characteristic equation should be non-zero and of the same sign. The Routh–Hurwitz criterion is based on ordering the coefficients of the characteristic equation

$$a_n s^n + a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \ldots + a_1 s + a_0 = 0 \quad (2)$$

into an array or schedule as follows [1, 4]:

$$\begin{array}{c|cccc}
s^n & a_n & a_{n-2} & a_{n-4} & \cdots \\
s^{n-1} & a_{n-1} & a_{n-3} & a_{n-5} & \cdots \\
s^{n-2} & b_{n-1} & b_{n-3} & b_{n-5} & \cdots \\
s^{n-3} & c_{n-1} & c_{n-3} & c_{n-5} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \cdots \\
s^0 & h_{n-1} & h_{n-3} & h_{n-5} & \cdots
\end{array}$$

where

$$b_{n-1} = \frac{(a_{n-1})(a_{n-2}) - a_n(a_{n-3})}{a_{n-1}} = \frac{-1}{a_{n-1}} \begin{vmatrix} a_n & a_{n-2} \\ a_{n-1} & a_{n-3} \end{vmatrix} \quad (3a)$$

$$b_{n-3} = \frac{-1}{a_{n-1}} \begin{vmatrix} a_n & a_{n-4} \\ a_{n-1} & a_{n-5} \end{vmatrix} \quad (3b)$$

$$c_{n-1} = \frac{-1}{b_{n-1}} \begin{vmatrix} a_{n-1} & a_{n-3} \\ b_{n-1} & b_{n-3} \end{vmatrix} \quad (3c)$$

and so on. The necessary and sufficient condition for the Routh–Hurwitz criterion is that there should be no changes in sign in the first column of the array for a stable system. The Routh–Hurwitz criterion states that the number of roots of the characteristic equation, $q(s)$, with positive real parts is equal to the number of the sign changes in the first column of the Routh array. In the software, the number of the sign changes is determined after the Routh array is completed. It shows where the sign changes are and the total number of sign changes. There are three distinct cases in the Routh–Hurwitz criterion that should be dealt with separately, requiring suitable modifications of the array calculation procedure.

Case 1: no element in the first column is zero. No changes should be made since the array is already completed. The software therefore determines the stability of the system by calculating the number of the sign changes in the first column of the array. It also gives some comments about this case.

Case 2: zeros in the first column while some other elements of the row containing the zero are non-zero. The zero in the first column should be substituted by a very small positive number, ε, that will approach zero after the array is completed. In the program, the value of ε is 1.0×10^{-5}. When the program encounters this case, the construction of the Routh array is paused. It shows some comments at the bottom of the array and waits for the user's response to continue with the construction of the array by using the value of ε as mentioned above. After the array is completed, the number of sign changes is calculated and some comments are given. (See the example in Fig. 1.)

Case 3: zero in the first column, and other elements of the row containing the zero are also zero. The row containing the zeros should be substituted by the coefficients of the derivative of the auxiliary equation. The coefficients of the auxiliary equation, $A(s)$, are the elements of the row preceding the row of zeros. Once the program encounters this case, it halts the construction of the Routh array and waits for the user's response to continue with the construction of the array. Some comments are also given at the bottom of the array. After the array is completed, the program computes the auxiliary equation, $A(s)$, and the number of the sign changes. Some explanations about this case are also given.

Roots analysis

The stability of a linear system can also be determined by the location of the roots or the poles on the s-plane. This method gives the relative stability of the system.

There are many approaches to finding the roots of a polynomial, including synthetic division [1, 5], Newton–Raphson [5], and Bairstow's method [6]. Our software uses Bairstow's method because this can calculate complex roots from real coefficients. The locations of the real part of the roots indicate

EDIT | ROUTH–HURWITZ | ROOTS | GRAPH | DOMAIN | QUIT

Click the mouse button to continue . . .

— — — — — — — — — — ROUTH–HURWITZ CRITERIA — — — — — —

s ^5	1.00	2.00	11.00	
s ^4	2.00	4.00	10.00	
s ^3	ε	6.00	0.00	
s ^2	−1.200E + 06	10.00	0.00	← sign change #1
s ^1	6.00	0.00	0.00	← sign change #2
s ^0	10.00	0.00	0.00	

So, the zero in row s ^0 is substituted by ε, a very small positive number which will approach zero after the array is completed. Here, ε has a value of 1.0E − 5.

The number of right-hand poles on the s-plane is determined by the number of sign changes of the entries in the first column. There are 2 poles at the right-hand side of the s-plane. So, the system is UNSTABLE.

Fig. 1 An example of a Routh–Hurwitz window.

[EDIT] [ROUTH–HURWITZ] [ROOTS] [GRAPH] [DOMAIN] [QUIT]

Click the mouse button to continue . . .

— — — — — — — — ROOTS ANALYSIS — — — — — — — — — — — —

right-hand root ⟶ s(1) = 0.8950 − j 1.4561
right-hand root ⟶ s(2) = 0.8950 + j 1.4561
left-hand root ⟶ s(3) = −1.2407 − j 1.0375
left-hand root ⟶ s(4) = −1.2407 + j 1.0375
left-hand root ⟶ s(5) = −1.3087 + j 0.0000

A system is unstable if there is a root with positive real part. There are 2 roots with positive real part on the s-plane. So, the system is UNSTABLE.

Fig. 2. An example of the roots analysis option.

how stable the system is. The three conditions for the roots analysis are:

1. If one or more roots are located at the positive half of the s-plane, the system is *unstable*.
2. If there are no roots with positive real part, and one or more roots are located along the imaginary axis, the system is *critically stable*.
3. If real parts of all roots are negative, the system is *stable*.

The software calculates the roots of the characteristic equation and determines the number of roots with positive real part. If the roots with positive real part exist, the program will tell the user that the system is unstable and give the reasons for this. (See Fig. 2 for an example of the roots analysis.)

SYSTEM STABILITY IN THE z-DOMAIN

The stability of a linear discrete system is also determined from the locations of the closed-loop poles on the z-plane. Consider the following transfer function for a closed-loop linear discrete system:

$$\frac{C(z)}{R(z)} = \frac{G(z)}{1 + GH(z)} \tag{4}$$

The closed-loop poles of the system are therefore defined as the roots of the characteristic equation, $P(z)$. $P(z)$ is derived from the following equation:

$$P(z) = 1 + GH(z) = 0 \tag{5}$$

Jury stability criterion

The first method used by SYSTA to determine the stability of a linear discrete-time system is the Jury stability table and the Jury stability criterion [2, 3]. This method gives the absolute stability of a system, and may be applied to a polynomial equation with real or complex coefficients. In applying the Jury stability test to a given characteristic equation $P(z) = 0$, we construct a table whose elements are based on the coefficients of $P(z)$. The general form of the characteristic equation in a linear discrete-time system is as follows:

$$P(z) = a_0 z^n + a_1 z^{n-1} + \ldots + a_{n-1} z + a_n \tag{6}$$

Table 1 shows the general form of the Jury table. Note that the first row consists of the coefficients of the characteristic equation $P(z)$, and the second row consists of the coefficients of $P(z)$ in reverse order. The elements of rows 3 to $(2n-3)$ are given by the following determinants:

$$b_k = \begin{vmatrix} a_n & a_{n-1-k} \\ a_0 & a_{k+1} \end{vmatrix}, \; k = 0, 1, 2, \ldots, n-1 \tag{7a}$$

$$c_k = \begin{vmatrix} b_{n-1} & b_{n-2-k} \\ b_0 & b_{k+1} \end{vmatrix}, \; k = 0, 1, 2, \ldots, n-2 \tag{7b}$$

$$\vdots$$

$$q_k = \begin{vmatrix} p_3 & p_{2-k} \\ p_0 & p_{k+1} \end{vmatrix}, \; k = 0, 1, 2 \tag{7c}$$

Table 1

Row	z^0	z^1	...	z^{n-2}	z^{n-1}	z^n
1	a_n	a_{n-1}	...	a_2	a_1	a_0
2	a_0	a_1	...	a_{n-2}	a_{n-1}	a_n
3	b_{n-1}	b_{n-2}	...	b_1	b_0	
4	b_0	b_1	...	b_{n-2}	b_{n-1}	
5	c_{n-2}	c_{n-3}	...	c_1	c_0	
6	c_0	c_1	...	c_{n-3}	c_{n-2}	
⋮	⋮					
$2n-5$	p_3	p_2	p_1	p_0		
$2n-4$	p_0	p_1	p_2	p_3		
$2n-3$	q_2	q_1	q_0			

Note also that the elements in the even-numbered rows are simply the reverse of the elements in the preceding odd-numbered rows. Once the Jury table has been completed, four conditions are tested to determine the stability of the system. The conditions are:

$$|a_n| < a_0 \qquad (8)$$

$$P(z)|_{z=1} > 0 \qquad (9)$$

$$P(z)|_{z=-1} \begin{cases} > 0 \text{ for } n \text{ even} \\ < 0 \text{ for } n \text{ odd} \end{cases} \qquad (10)$$

$$\begin{aligned} |b_{n-1}| &> |b_0| \\ |c_{n-2}| &> |c_0| \\ &\vdots \\ |q_2| &> |q_0| \end{aligned} \qquad (11)$$

Based on the preceding tests, we can conclude that:

1. If all conditions are satisfied, the system is *stable*.
2. If conditions (8) and (11) are satisfied, and $P(z)|_{z=1} = 0.0$ and/or $P(z)|_{z=-1} = 0.0$, the system is *critically stable*.
3. If one or more conditions are not satisfied, the system is *unstable*.

Once the software has completed the Jury table, it runs the tests to determine the stability of the system. Based on the tests, the software determines how many conditions are not satisfied for a stable system. It also points to the unsatisfied tests. (See Fig. 3 for an example of Jury stability criterion.)

Magnitude and angle

The second method used by SYSTA to determine the stability of a linear discrete-time system is determining the location of the roots of the discrete-time system characteristic equation, $P(z)$ [3]. This method gives the relative stability of a system. The necessary and sufficient condition for the stability of a linear discrete-time system is that the roots of the characteristic equation should lie inside the unit circle on the z-plane [2]. There are three conditions for determining the stability of a linear discrete system on the z-plane:

1. If one or more roots are located outside the unit circle (magnitude > 1.0), the system is *unstable*.
2. If there are no roots located outside the unit circle, and one or more roots are located on the unit circle (magnitude = 1.0) the system is *critically stable*. However, from the physical point of view, such a system is considered as unstable because a very small change in the physical constants may create instability on the system.
3. If all roots are located inside the unit circle (magnitude < 1.0), the system is *stable*.

The software computes the magnitude and angle of the roots and justifies the stability of the system by checking the magnitude of the roots. Based on

[EDIT] [JURY-STABILITY] [MAGNITUDE] [GRAPH] [DOMAIN] [QUIT]

Click the mouse button to continue . . .

— — — — — — — — — — — — — — JURY STABILITY — — — — — — — — — — — — — —

Row	z^0	z^1	z^2	z^3	z^4
1	1.000	2.000	11.000	18.000	18.000
2	18.000	18.000	11.000	2.000	1.000
3	18.000	187.000	322.000	323.000	
4	323.000	322.000	187.000	18.000	
5	5.460E + 04	1.006E + 05	1.040E + 05		
6	1.040E + 05	1.006E + 05	5.460E + 04		

STABILITY TESTS:

1. |18| = 18 < 1 ? ← not satisfied
2. P(1) = 50.00 > 0 ? ← satisfied
3. P(−1) = 10.00 > 0 ?; n in even ← satisfied
4. |323| > |18| ? ← satisfied
5. |104005| > |54605| ? ← satisfied

1 condition is not satisfied. So, the system is UNSTABLE.

Fig. 3. An example of a Jury stability window.

the magnitude, the program gives some comments about the stability of the system. From the GRAPH option, the software displays the locations of the roots with the unit circle on the z-plane. (Figure 4 gives an example of this type of plot.)

COMPUTER SIMULATION SOFTWARE

Upon executing the SYSTA, the title of the program appears on the screen along with the menu to select the domain that the user would like to work on. The default is the s-domain. The menu items may be selected by using the cursor keys or a mouse if one is available. If a mouse is used, click the left mouse button when a menu item has been highlighted. After selecting the domain, the main menu appears on the screen.

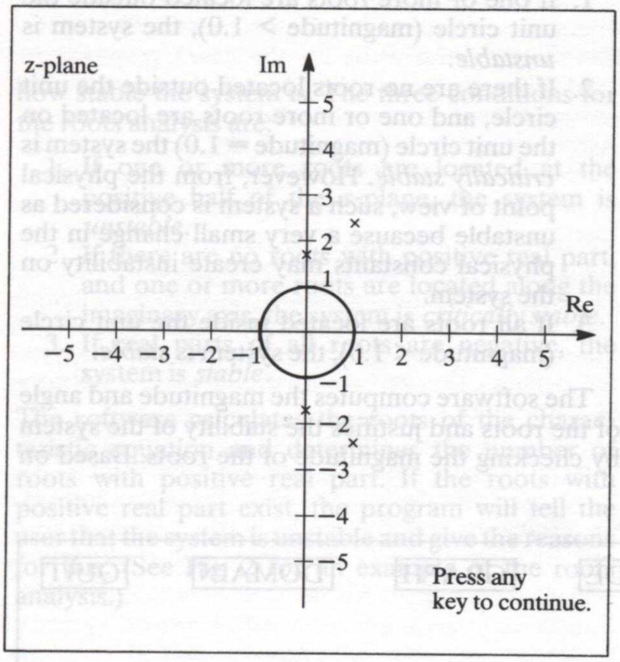

Fig. 4. An example of the GRAPH option.

The organization of the software is explained by the flowchart shown in Fig. 5. The EDIT function, which should be executed first, asks the user to input the coefficient of the characteristic equation. In typing in the coefficients, the user should press the ENTER key to confirm an entry. If all procedures have been followed, the characteristic equation will appear at the bottom of the window, and the EDIT window disappears once a key is pressed.

In the s-domain, the user can work on the ROUTH–HURWITZ stability criterion and the ROOTS ANALYSIS. In the Routh–Hurwitz analysis, the program simulates the Routh array on the screen and gives some comments on the stability of the system (based on the characteristic equation entered by the user). It explains the Routh–Hurwitz criterion concepts in an interactive way with the use of pop-up windows. In the roots analysis, the program analyzes the roots and figures out the number of roots with positive real part in the characteristic equation. A system is unstable if there is a root with positive real part. The GRAPH option displays the location of the roots on the s-plane graphically.

The JURY STABILITY test and the MAGNITUDE AND ANGLE analysis are available in the z-domain. In the Jury stability analysis, the program simulates the Jury table on the screen and analyzes the stability of the system by implementing the Jury tests. Magnitude and angle analysis computes the magnitude and angle of the roots of the characteristic equation. The system is unstable if there is a root that has a magnitude greater than 1.0. The GRAPH option gives a graphical display of the location of the roots on the z-plane.

The user can change the domain that he or she is working on by selecting the option DOMAIN from the main menu. QUIT returns the user to the DOS prompt.

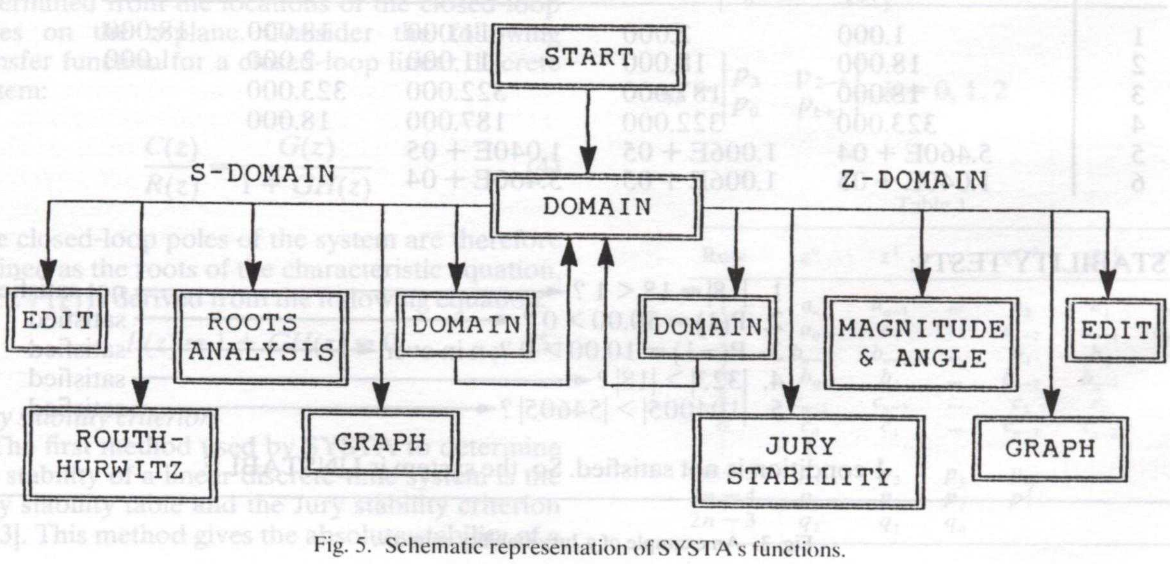

Fig. 5. Schematic representation of SYSTA's functions.

CONCLUSIONS

Many computer programs can analyze the stability of a system. However, it is hard to find one that is intended for use as an interactive teaching aid as well as a design tool. Furthermore, most of the software currently available does not interact with the user very much: the emphasis is merely on the solutions. Unlike other software, SYSTA interacts with the user very well through the use of pop-up windows and menus. It is user friendly and accurate, which makes it highly suitable for use either as a teaching aid or as a analysis tool.

REFERENCES

1. R. C. Dorf, *Modern Control Systems*, 5th edn, Addison-Wesley, New York (1989).
2. E. I. Jury, *Theory and Application of the z-Transform Method*. Robert E. Krieger, Florida (1986).
3. K. Ogata, *Discrete-Time Control System*, Prentice Hall, Englewood Cliffs, NJ (1987).
4. B. C. Kuo, *Automatic Control Systems*, 6th edn, Prentice Hall, Englewood Cliffs, NJ (1990).
5. W. H. Press and P. F. Brian, *Numerical Recipes in C*, Cambridge University Press, Cambridge (1989).
6. J. L. Melsa and K. J. Stephen, *Computer Programs for Computational Assistance in the Study of Linear Control theory*, McGraw-Hill, New York (1973).

Samin Wahid was born in Johore, Malaysia in June 1968. He has just received the B.S. degree in Electrical Engineering with a concentration in Control and Systems from the State University of New York at New Paltz.

His research interests include the applications of artificial intelligence, robotics, and software engineering. He is currently working on a GUI-based computer simulation for modeling of Control Systems.

Ali M. Eydgahi received his B.S., MS., and Ph.D. degrees in Electrical Engineering. Since 1986 he has been with the State University of New York, where he is currently an Associate Professor teaching courses in computer aided design, control theory and design, systems analysis, and robotics. His research interests are in the areas of multidimensional systems, computer-aided design and simulations for Engineering Education, computer algebra applications, digital signal processing.

He was the recipient of an ASEE Dow Outstanding Young Faculty Award in 1990. He is a senior member of the IEEE and SME, and a member of ASEE and Sigma Xi.