A POLYNOMIAL CONTROL SYSTEMS PACKAGE

Neil Munro, *Fellow, IEEE*

Abstract--A new integrated software package called Polynomial Control Systems, which is fully compatible with Mathematica's Control System Professional, is described. The package, which provides several additional model manipulation facilities and system analysis tools, and two frequency-domain multivariable system design methodologies, is also fully compatible with a related package dealing with descriptor systems..

Index terms--CAD, polynomial systems, model manipulation, system analysis, frequency-domain design.

I. INTRODUCTION

Mathematica [1], like other similar computing environments, in addition to its kernel system, is enhanced by various domain-specific packages, such as the "Control System Professional" [2]. This latter package provides the classical single-input single-output control system analysis and design tools, the established Kalman tests for controllability and observability, the controllability and observability Gramians, minimal realisation algorithms, pole assignment, optimal control system design, system interconnection facilities, system simulation, and many other tools.

In recent years, the author has been developing a new polynomial control systems package using Mathematica to provide CAD tools to support the teaching of multivariable control systems theory, and also for use in research and in solving real industrial control problems. In the following, the facilities implemented for linear system model manipulation, linear system analysis, and linear multivariable system design in the frequency domain will be presented. All of the facilities to be considered are fully compatible with Mathematica's Control System Professional (CSP), and are also fully compatible with a related package dealing with descriptor systems [3].

II. MODEL MANIPULATION

The author is Head of the Control Systems Centre, University of Manchester Institute of Science and Technology (UMIST), Manchester M60 1QD, UK. Email: n.munro @umist.ac.uk

The Linear Models manipulation facilities implemented provide the various transformations needed to automatically manipulate linear system models between any of the following standard forms; state space, transfer-function (matrix), left or right matrix-fraction form, and Rosenbrock's [4,5] system matrix in state-space or polynomial form. The data structure for state space objects is the form used by the H_∞ robust control system design method. Using the transformations to left or right matrix-fraction forms, now widely used in the H_∞ design method, the resulting system models can also be readily reduced to least-order, or coprime, form.

Using the facilities currently available in Mathematica's CSP, a user can enter a system description either as a TransferFunction, StateSpace, or ZeroPoleGain object, as shown below:

tf = TransferFunction[s, {{1/(s+1), 2/(s+3)}}]
ss = StateSpace[{{0, 1},{-3, -4}},{{0},{1}},{{3, 1},{2, 2}}]

These can then be manipulated into the alternative model forms provided in CSP by simply applying the appropriate wrapper around the form concerned; e.g.

ss = StateSpace[tf]
tf =TransferFunction[s, ss]
zpg = ZeroPoleGain[tf]

The resulting model data held in these objects can then be inspected by using the CSP command //ReviewForm. However, these model formats have now been improved so that entering the state space object **ss** or transfer function object **tf** automatically results in the more compact and pleasing output formats, shown below

$$\mathrm{ss} = \left(\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right) \Rightarrow \left(\begin{array}{cc|c} 0 & 1 & 0 \\ -3 & -4 & 1 \\ \hline 3 & 1 & 0 \\ 2 & 2 & 0 \end{array}\right)^{S} \qquad \mathrm{tf} = \begin{pmatrix} \frac{1}{s+1} \\ \frac{2}{s+3} \end{pmatrix}^{T} \qquad (1)$$

These new model formats, which are fully interactively editable, have been supplemented by Rosenbrock's system matrix description for systems in state space or polynomial form, and matrix-fraction descriptions; e.g.

rmf = RightMatrixFraction[tf]

results in

$$\begin{pmatrix} s+3 \\ 2(s+1) \end{pmatrix}\left((s+1)(s+3)\right)^{-1}|^{R} \tag{2}$$

This, or any other of the now extended model forms, can equally be transformed to a left matrix-fraction object by applying the appropriate wrapper to the current object form; e.g.

lmf = LeftMatrixFraction[rmf]

which results in

$$\begin{pmatrix} s+1 & 0 \\ 0 & s+3 \end{pmatrix}^{-1}\begin{pmatrix} 1 \\ 2 \end{pmatrix}|^{L} \tag{3}$$

Any of the model objects considered above can equally be manipulated into a system matrix object, by using the command

SystemMatrix[tf, TargetForm→RightFraction]

which results in

$$\left(\begin{array}{c|c} T(s) & U(s) \\ \hline -V(s) & W(s) \end{array}\right) \Rightarrow \left(\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & (s+1)(s+3) & 1 \\ \hline 0 & -s-3 & 0 \\ 0 & -2(s+1) & 0 \end{array}\right)^{M} \tag{4}$$

Here, the option **TargetForm** can be used to force the resulting system matrix object to be created from a right matrix-fraction form as in (2), or from a left matrix-fraction form as in (3), by setting **TargetForm→LeftFraction**.

III. SYSTEM ANALYSIS

The linear System Analysis facilities implemented provide a range of well-established analysis tools. In addition to the established state-space model controllability and observability tests, provided in the CSP facility, the user can now also request the controllability or observability of a system described by a polynomial system matrix model, by simply entering **Controllable[ps]** or **Observable[ps],** where **ps** is the name of the SystemMatrix object concerned; e.g. entering the command

Controllable[ps]

generates the response **True**, where **ps** is the system matrix object defined by (4) above.

The functions **SmithForm** and **McMillanForm** have been provided to determine the Smith form of a polynomial matrix and the McMillan form of a rational polynomial matrix. The algorithm used for the Smith form is that proposed by Kailath [6]. For example, the McMillan form of the TransferFunction object corresponding to the two-input two-output StateSpace object

ss =

$$\left(\begin{array}{cccc|cc} 1.38 & -0.2077 & 6.715 & -5.676 & 0 & 0 \\ -0.5814 & -4.29 & 0 & 0.675 & 5.679 & 0 \\ 1.067 & 4.273 & -6.654 & 5.893 & 1.136 & -3.146 \\ 0.048 & 4.273 & 1.343 & -2.104 & 1.136 & 0 \\ \hline 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right)^{S} \tag{5}$$

is determined as

McMillanForm[TransferFunction[s, ss]]

$$\begin{pmatrix} \dfrac{1}{(s-1.991)(s-0.0635)(s+5.056)(s+8.666)} & 0 \\ 0 & (s+1.192)(s+5.039) \end{pmatrix} \tag{6}$$

Several related functions are also provided; namely, **InvariantZeros,** which uses the Smith form to calculate the invariant zeros of a system, **McMillanDegree**, which uses the McMillan form to determine the finite poles of a multivariable system, and can be used to determine the order of any minimal state space realisation of the transfer function matrix concerned, and **TransmissionZeros** which can be used to determine the existence of any right-half plane “transmission zeros”. The position of any rhp-zeros in the diagonal terms of the McMillan form indicates the point in any loop-closure procedure at which the non-minimum phase effect will manifest itself. A further minimal realisation algorithm (Patel and Munro, [5]), based on Rosenbrock’s decoupling zeros theory, has been implemented that carries out a minimum of numerical operations, compared with most other algorithms, in determining the desired minimal-order state space model.

The functions **LeftCoprime** and **RightCoprime** are provided to determine whether two polynomial matrices are coprime, or not, and return True or False. An algorithm to determine a coprime factorisation of a given transfer function matrix model has also been implemented. This can be used with any initial Left or RightMatrixFraction object that may not be in least-order form to detect and remove any Left or Right Matrix Common Factor in the resulting numerator and denominator matrices. Suppose that a system description has been entered as a SystemMatrix object in polynomial form; e.g.

ps = SystemMatrix[s, t, u, v, w]

$$\left(\begin{array}{cccc|c} s^2(s+1) & s^3+s^2-1 & 1-s^2(s+1) & 0 & 0 \\ s(s+2) & s^2+3s+2 & -s(s+2) & 0 & 1 \\ s(s+2) & s^2+3s+1 & 1-s(s+2) & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & -1 & 0 \end{array}\right)^{M} \tag{7}$$

Then, by invoking **LeftGCDDecomposition[t, u, s]**, the user can determine the Matrix Left Greatest Common Divisor, L(s), of the matrices T(s) and U(s), if any, yielding here

$$L(s) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & \mathbf{s^2(s+1)} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{U_r(s)} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{8}$$

$$T_r(s) = \begin{pmatrix} s^2(s+1) & s^3+s^2-1 & -(s^3+s^2-1) & 0 \\ s(s+2) & s^2+3s+2 & -s(s+2) & 0 \\ -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{9}$$

where $U_r(s)$ and $T_r(s)$ are the coprime forms of U(s) and T(s), respectively.

Of course, the Smith form of the system matrix **ps**, defined in (7), can be obtained using the command

SmithForm[ps]

yielding

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & s^2(s+1) & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{10}$$

which would equally have revealed that the system was not least order, and had either, or both, input decoupling zeros and output decoupling zeros.

The functions **InputDecouplingZeros** and **OutputDecouplingZeros** are provided to detect the presence of such input or output decoupling zeros in a system matrix model, in polynomial or state-space form, and **RemoveInputDecouplingZeros**, and **RemoveOutputDecouplingZeros** can be used to remove these. For example, using the system matrix defined by (7) above,

InputDecouplingZeros[ps]

would yield {-1, 0, 0} (11)

and **OutputDecouplingZeros[ps]**

would yield {-2, -1, 0, 0} (12)

However, if the input decoupling zeros were first removed from **ps**, yielding the reduced-order system

$$\left(\begin{array}{cccc|c} s^2(s+1) & s^3+s^2-1 & -s^3-s^2+1 & 0 & 0 \\ s(s+2) & s^2+3s+2 & -s(s+2) & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & -1 & 0 \end{array}\right)^{M} \tag{13}$$

then applying the command

OutputDecouplingZeros[ps]

to the system matrix **ps** defined by (13) would yield

{-2} (14)

as expected. The function **LeastOrderSystem** uses these latter functions to determine a least-order, or lowest-order, form of a system described by a system matrix in polynomial form; e.g.

LeastOrderSystem[ps]

where **ps** is the system matrix defined earlier by (7) yields

$$\left(\begin{array}{cccc|c} s^2(s+1) & \frac{1}{4}(-s^3-s^2-4) & 0 & 0 & 0 \\ s(s+2) & \frac{1}{4}(-s^2+2s+8) & 1 & 0 & 1 \\ -1 & \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & -1 & 0 \end{array}\right)^{M} \tag{15}$$

IV. MULTIVARIABLE SYSTEM DESIGN

Here, two well-established frequency domain design techniques for multivariable systems have been provided. The functions **Direct** and **InverseNyquistArrayPlot** implement Rosenbrock's Direct and Inverse Nyquist Array design methods [4,5].

In addition, various other recent interaction measures, such as functions to determine Bristol's Relative Gain Array [7] **RGA** and the **RGANumberPlot**, and functions to plot the behaviour of the Perron-Frobenius eigenvalue and the elements of the Perron-Frobenius eigenvector of the system with frequency, for scaling of the system inputs [8], are provided.

For example, applying the relative gain array function **RGA[tf]** to the transfer function object **tf** defined as

$$\begin{pmatrix} \frac{s+4}{(s+1)(s+5)} & \frac{1}{5s+1} \\ \frac{s+1}{s^2+10s+100} & \frac{2}{2s+1} \end{pmatrix}^{T} \tag{16}$$

yields

$$\begin{pmatrix} 1.00629 & -0.00628931 \\ -0.00628931 & 1.00629 \end{pmatrix} \qquad (17)$$

which clearly shows that the current input-output pairing is appropriate for further design considerations.

The direct Nyquist array for the transfer function object **tf** defined by (16), with Gershgorin disks for row diagonal dominance superimposed on the diagonal elements, can be generated over a range of frequencies ω going from 0 to 20 rad/sec, by the function

DirectNyquistArrayPlot[tf, {0, 20},
GershgorinDisks→Row]

The resulting graphical output is shown in Figure 1, which shows that the system is not row diagonal dominant. The option **GershgorinDisks** can equally be set to the value **Column**, if the column diagonal dominance of the system is to be assessed.

Applying the function **PFEigenvaluePlot[tf]** produces the output shown in Figure 2. As the PF eigenvalue is less than 2 over the bandwidth of interest, this indicates that a dynamic diagonal input scaling compensator exists that will make the system row diagonal dominant over this bandwidth. This compensator must be a good approximation to the magnitude behaviour of the elements of the PF right eigenvector over this bandwidth. The function **PFScalingCompensator**, has been implemented to determine the constant or dynamic forms of an appropriate Perron-Frobenius based diagonal scaling compensator. A suitable scaling compensator is determined using this function, as shown below

$$\begin{pmatrix} 0.396114(s+0.360572)(s+6.65634)(s+6.658) & 0 \\ 0 & 1 \end{pmatrix}^{T} \qquad (18)$$

and the goodness of the resulting approximation is shown in Figure 3. The resulting Direct Nyquist Array, with row 2 scaled up by a factor of 6, as shown in Figure 4, is clearly row diagonal dominant over the frequency range of interest.

Hawkins's **PseudoDiagonalisation** method [9] for determining a wholly real compensator that achieves good diagonal dominance in a system at a given frequency is also implemented. A graphical function **DominanceRatiosPlot,** that can be used to determine the row, or column, dominance ratios of a given system, has also been implemented. These ratios are defined for a m × m matrix in terms of column dominance, as

$$dr_j^c = \sum_{\substack{i=1 \\ i\neq j}}^{m} |\, q_{ij}(j\omega)\,| \,/\, |\, q_{ii}(j\omega)\,| \qquad (19)$$

for column j. The dominance ratios achieved with the 3rd-order scaling compensator are shown in Figure 5. With the perfect Perron-Frobenius scaling compensator, the same dominance ratios would be achieved in each row of the direct Nyquist array at the same frequency. However, here you can see that the dominance ratios are the same at low and intermediate frequencies, but deviate at the higher frequencies.

The Characteristic Locus based design method [10] has also been implemented with functions **CharacteristicValuesPlot** to display the behaviour of the characteristic loci, and **CharacteristicVectorsPlot** to display the misalignment between the system's characteristic vectors and the Euclidean basis set with frequency, where the latter is used as a measure of interaction in a system. The **Align** algorithm [11] is provided to determine a high frequency constant compensator used with this method to improve this alignment, and also the wholly real matrices L and R of a commutative controller

$$K(s) = L.\mathrm{diag}\{k_i(s)\}.R \qquad (20)$$

that can be used to shape the mid-frequency behaviour of the characteristic loci.

To illustrate these tools, consider the 4-input 4-output transfer function model of a reheat furnace [12], given by

$$G(s)=\begin{bmatrix} \frac{1.0}{1+4s} & \frac{0.7}{1+5s} & \frac{0.3}{1+5s} & \frac{0.2}{1+5s} \\ \frac{0.6}{1+5s} & \frac{1.0}{1+4s} & \frac{0.4}{1+5s} & \frac{0.35}{1+5s} \\ \frac{0.35}{1+5s} & \frac{0.4}{1+5s} & \frac{1.0}{1+4s} & \frac{0.6}{1+5s} \\ \frac{0.2}{1+5s} & \frac{0.3}{1+5s} & \frac{0.7}{1+5s} & \frac{1.0}{1+4s} \end{bmatrix} \qquad (21)$$

The four characteristic loci for this system, shown in Figure 6, are badly out of balance, and the characteristic locus design method attempts to equalise these magnitude characteristics. The corresponding direct Nyquist plot of element $G_{1,1}(j\omega)$ is shown in Figure 7, with Gershgorin row-dominance circles. Since all four diagonal elements of G(s) are identical, and the transfer-function matrix has quasi-symmetry, the resulting dominance characteristics are similar for each of the other three diagonal elements, which are clearly not open-loop diagonal dominant.

Using the **Align** function, a high-frequency compensator

$$\mathbf{K_h} = \begin{bmatrix} 11.1734 & -6.0307 & -0.7699 & 0.0282 \\ -4.9052 & 11.7083 & -1.9666 & 1.0831 \\ -1.7370 & -1.6062 & 11.8075 & -5.0156 \\ 0.3620 & -0.9472 & -6.0307 & 11.1317 \end{bmatrix} \qquad (22)$$

determined at 2 radians/second, yields the more balanced characteristic loci shown in Figure 8 for $Q(s) = G(s)K_h$, and the corresponding direct Nyquist plot for $Q_{1,1}(s)$, with superimposed Gershgorin row dominance circles shown in Figure 9, and similarly for the other diagonal terms. Since Q(s) is now diagonal dominant with minimal transient interaction, and is open-loop stable, the resulting closed-loop system will be stable with large forward-path gains, or proportional-plus-integral action controllers, which can be implemented to reduce the remaining steady-state interaction and achieve good steady-state performance.

V. ACKNOWLEDGEMENTS

The help of Dr Igor Bakshee, of Wolfram Research Inc., in integrating the Linear Models, System Analysis, and System Design facilities into the Control System Professional is gratefully acknowledged.

VI. REFERENCES

[1] S. Wolfram, *Mathematica, A System for Doing Mathematics*, Second Edn., Addison-Wesley, 1991.

[2] Wolfram Research Inc., *The Control System Professional*, Illinois, 1997.

[3] N.P. Karampetakis and A.I. Vardulakis, A Descriptor Systems Package for Mathematica, 11th Mediterranean Conference on Control and Automation MED'03, Rhodes, 2003.

[4] H.H. Rosenbrock, *State Space and Multivariable Theory*, Wiley, 1970.

[5] R.V. Patel and N. Munro, *Multivariable System Theory and Design*, Pergamon Press, UK, 1982.

[6] T. Kailath, *Linear Systems*, Prentice-Hall, 1980.

[7] E.H. Bristol, On a new measure of interaction for multivariable process control, IEEE Transactions on Automatic Control, AC-11, pp.133-4, 1966.

[8] A.I. Mees, Achieving Diagonal Dominance, Systems & Control Letters, Vol. 1, No. 3, 1981.

[9] D.J. Hawkins, Pseudodiagonalisation and the Inverse Nyquist Array Method, Proc. IEE, vol. 119, pp.337-342, 1972.

[10] A.G.J. MacFarlane, *Complex Variable Methods for Linear Multivariable Feedback Systems*, Taylor & Francis Ltd. (London), 1980.

[11] B. Kouvaritakis, *Characteristic Locus Methods for Multivariable Feedback System Design*, PhD Thesis, UMIST, Manchester, UK, 1974.

[12] R. Luntz and N. Munro, Applications of the Inverse Nyquist Array Design Method, Ch. 6 in *Modern Approaches to Control System Design*, (Ed. N. Munro), Peter Perigrinus Ltd., 1979.

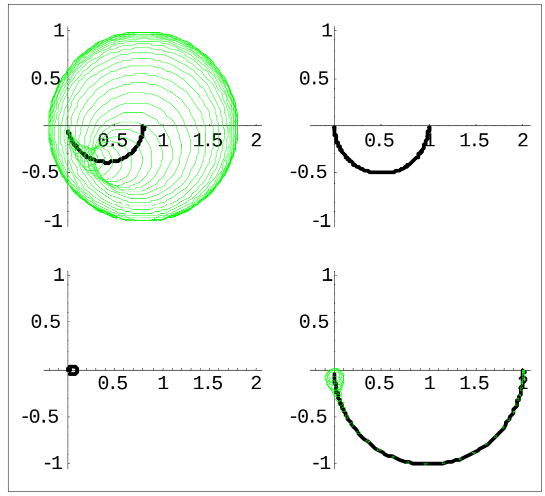

Figure 1: Direct Nyquist Array

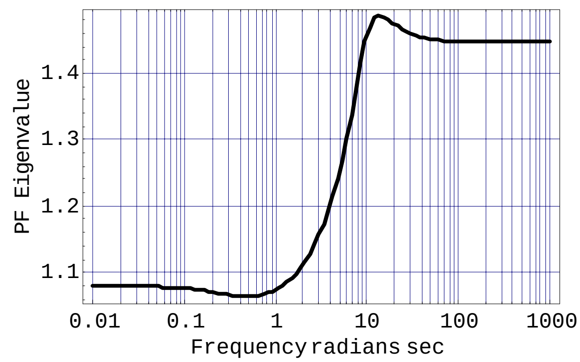

Figure 2: Plot of Perron-Frobenius eigenvalue.

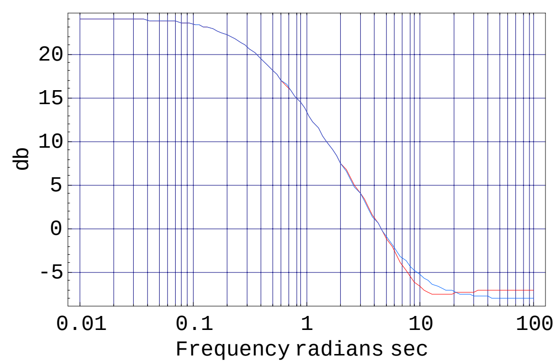

Figure 3: Fit to the second element of the PF eigenvector.

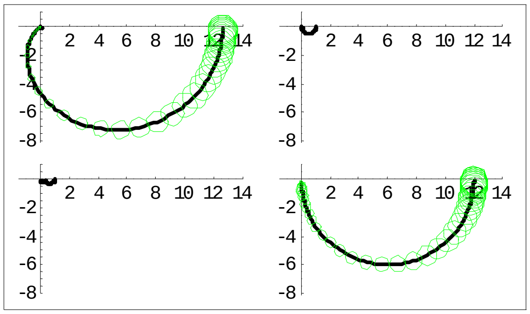

Figure 4: Direct Nyquist Array of scaled system

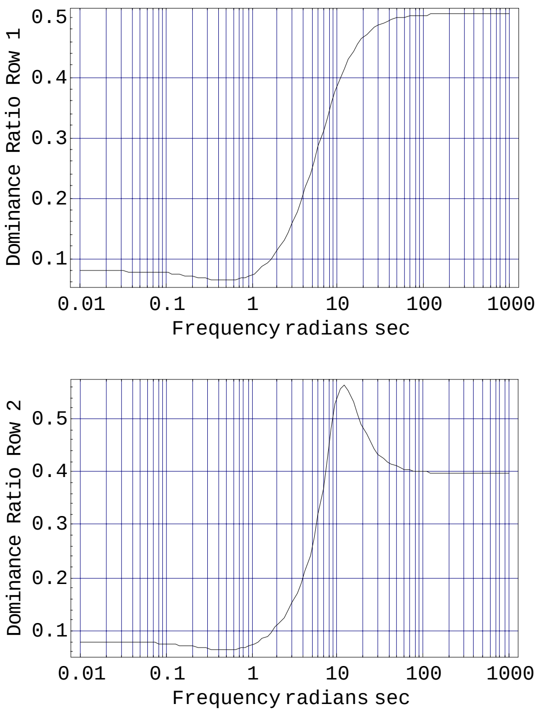

Figure 5: Dominance ratios for scaled system.

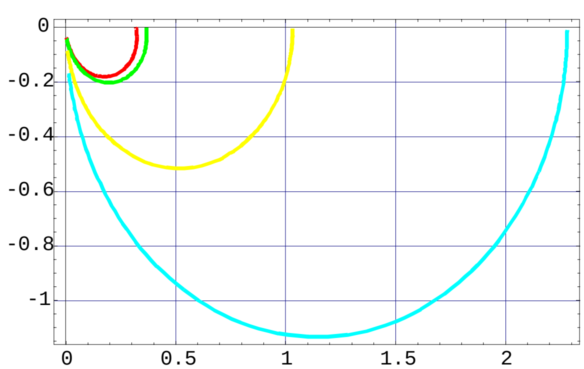

Figure 6: Characteristic Loci for the Reheat Furnace.

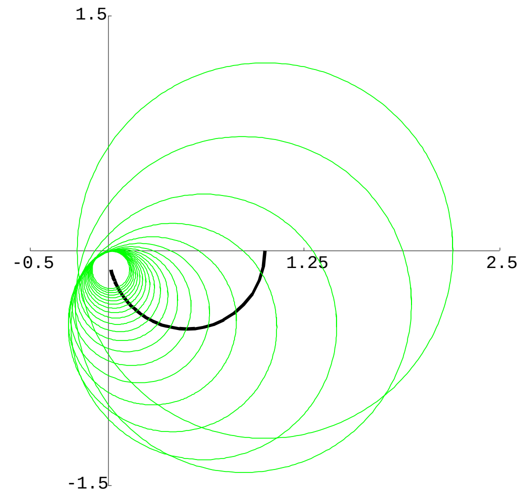

Figure 7: Nyquist plot of G_{11} for the Reheat Furnace.

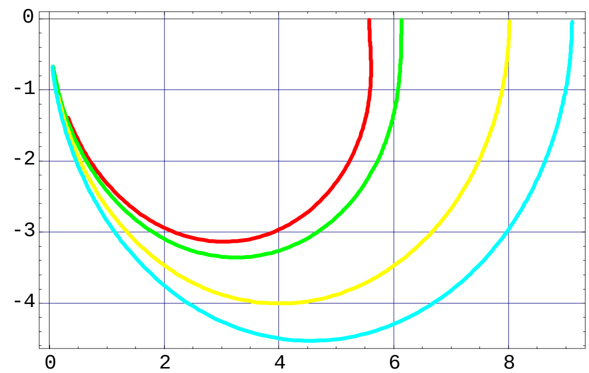

Figure 8: Characteristic loci for $G(s)K_h$.

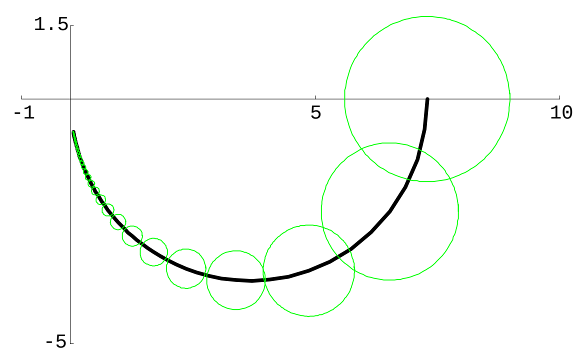

Figure 9: Nyqist plot of $Q_{11}(s)$.