COSY_PAK V 0.9 MANUAL OF FUNCTIONS

by

C-K. Chen and N. Sreenath

Systems Engineering Department
Case School of Engineering
Case Western Reserve University
Cleveland OH 44106-7070

sree@mishna.cwru.edu

Introduction

All transfer functions used in **COSY_PAK** are scalar unless specified. Some **COSY_PAK** functions return graphic or boolean output. Most other functions return alpha-numeric output in the form of a list. The desired output should be extracted from the returned value if more than one variable is returned. An example is given below.

Example :

In[11] : pout ={{k,l},{m,n,j}};

In[12] : pout[[2]]
Out[12] : {m,n,j}

In[13] : pout[[2,1]]
Out[13] : m

Presently we support only single input single output systems (SISO). Future versions of **COSY_PAK** would incorporate multi-input multi-output systems (MIMO). The general algorithms for these functions can be found in any standard control engineering textbook such as Ogata [1991] (see README file).

See **README** file for more information on **COSY_PAK.** The functions in this manual are listed according to the **COSY_Notes** notebook chapter.

Symbol Definition

gopts: *Mathematica* graphics options.
s: Laplace variable.
t: time variable
Transf(s): Transfer function.
y(t): Function of time.
&&: AND operation, **||**: OR operation.

Chapter 1. Introduction to Control Systems Analysis

ChekAnal[Transf, s, r, w,showder]: Checks the analyticity of the transfer function Transf(s) at the point **s=r+jw** using **Cauchy-Reimann** conditions. If '**showder**=1' (optional) then the derivatives in the computation are shown**.**

DiracDelta[t]: a distribution that is 0 for t != 0 and satisfies **Integrate[DiracDelta[t], {t, -Infinity, Infinity}]** = 1. **DiracDelta[t1, t2, ...]** is a distribution that is 0 for t1 != 0 || t2 != 0 || ... and satisfies **Integrate[DiracDelta[t1, t2, ...], {t1, -Infinity, Infinity}, {t2, -Infinity, Infinity}, ...]** =1. (*Mathematica's* Function)

DSolve[eqn, y[t], t]: solves a differential equation for the functions **y[t]**, with independent variable **t**. **DSolve[{eqn1, eqn2, ...}, {y1[t1, ...], ...}, {t1, ...}]** solves a list of differential equations. (*Mathematica's* Function)

InverseLaplaceTransform[expr, s, t, opts]: gives **a** function of **t**, the Laplace transform of which is **expr**, a function of **s**. (*Mathematica's* Function)

LaplaceTransform[expr, t, s, opts]: gives a function of **s**, which is the Laplace transform of **expr**, a function of **t**. It is defined by **LaplaceTransform[expr, t, s] =Integrate[Exp[-s t] expr, {t, 0, Infinity}]**. (*Mathematica's* Function)

PoleZeros[Transf, s]: Computes finite poles and zeros of the transfer function **Transf .** Returns **{list of poles, list of zeros}.**

UnitStep[t]: a function that is 1 for t > 0 and 0 for t < 0. **UnitStep[t1, t2, ...]** is 1 for (t1 > 0) && (t2 > 0) && ... and 0 for (t1 < 0) || (t2 < 0) || ... . (*Mathematica's* Function)

Chapter 2. Mathematical Modeling of Dynamic Systems

Linearize[f , zvars , zpoint , vvars , vpoint]: Gives the linearization of vector function **f[zvars, vvars] = {f_1[zvars, vvars], ... ,f_n[zvars, vvars]}** around the operating point **zpoint** and **vpoint**. The length of the state variables **zvars = [x_1,...,x_n]** must be equal to the length of the operating point **zpoint** = **[x_{10},...,x_{n0}]** and the length of the control variables **vvars** = **[v_1,...,v_r]** must be equal to the length of the operating point **vpoint** = **[v_{10},...,v_{r0}].** Returns the linearized system matrices **A** and **B** as **{A,B}**.

Ode2SS[lhscoeff, rhscoeff]: Converts linear ordinary differential equation (ODE) to *state space eqn*. The ODE is in the format:

$$y^{(n)} + a_1 y^{(n-1)} + ... + a_{n-1} y^{(1)} + a_n y = b_0 u^{(n)} + b_1 u^{(n-1)} + ... b_{n-1} u^{(1)} + b_n u$$

where **lhscoeff** = [1, a_1,...,a_n], **rhscoeff** = [b_0,b_1,...,b_n]. If a_i or b_i don't exist, use a_i=0 or b_i=0. The output are Matrix A and Vector B in the state equation: dx/dt = A x + B u. Returns the state space matrices **A** and **B** as **{A,B}**.

SS2Transf[A,B,C,s]: This function transforms the state space representation of system **(A, B, C)** to its transfer function representation. Returns the transfer function **transf** as a function of the Laplace variable 's'.

Chapter 3. Transient Response Analysis

Response[transf, Input, s, {TimeVar, StartTime, EndTime}, gopts]: Plots the output of transfer function '**transf**' with Laplace input signal '**Input**'. The Laplace variable is '**s**'. The output graph uses the variable '**TimeVar**' and starts at '**StartTime**' and ends at '**EndTime**'. Returns the output variable as a function of time '**t**'.

SecOrder[zeta, wn, t]: Gives the unit step response value at time instant **t** for a standard second order system $\omega_n^2/(s^2 + 2\zeta\ \omega_n\ s + \omega_n^2)$ with the damping ratio ζ=**zeta** and natural frequency ω_n =**wn**. Returns instantaneous value of the step response output variable at time instant '**t**'.

Chapter 4. Steady-State Response Analysis

Routh[Charpoly, s, z]: Gives the Routh's table for application of Routh's stability criterion. The parameter **Charpoly** is the characteristic polynomial with variable **s**. The parameter **z** define the symbol to replace the zero value of first column terms if any. The characteristic polynomial is the denominator of the transfer function. Returns Routh's table as an array.

Chapter 5. Root-Locus Analysis

RootLocus[Transf, s, {k,kmin,kmax}, gopts]: Plots the root-locus plot of the transfer function **Transf(s)** with the Parameter **k** varying from **k=kmin** to **k=kmax**. Returns graphics.

Chapter 6. Frequency-Response Analysis

MagPlot[Transf, s, {w,wmin,wmax}, gopts]: Plots the magnitude part of Bode plot of transfer function **Transf(s)** in decibel(dB), from frequency **w=wmin** to **w=wmax**, both > 0 and in radians per second. Returns graphics.

MagvsPhase[Transf, s, {w,wmin,wmax}, gopts]: Plots the magnitude vs. phase plot of transfer function **Transf(s)** with **s=jw**. The frequency **w** varies from **w=wmin** to **w=wmax** both > 0 and in radians per second. Returns graphics.

NyquistPlot[Transf, s, {w,wmin,wmax}, gopts]: Plots the Nyquist plot of the transfer function **Transf(s)**. The plot is composed of three parts. The first part corresponds to **s=jw** with **w=-wmin** to **w=-wmax**. The second part corresponds to **s=jw**, **w** varying from **w=+wmin** to **w=+wmax**. Encirclement information of (-1+j0) is provided by the third part which corresponds to s with theta=-pi/2 to pi/2. **wmin** and **wmax** should be > 0 and radians per second. Returns graphics.

PhasePlot[Transf, s, {w,wmin,wmax}, gopts]: Plots the phase part of Bode plot of transfer function **Transf(s)** in degree, from frequency **w=wmin** to **w=wmax**; **w** is in radian per second.Returns graphics.

Polar[Transf, s, {w,wmin,wmax}, gopts]: Plots the polar plot of the transfer function **Transf(s)** with **s=jw** , **w** varying from **w=wmin** to **w=wmax**, both > 0 and in radians per second. Returns graphics.

Chapter 7. State Space Analysis Methods

Controllable[A, B]: Returns a logic (boolean) value representing the complete state controllability of system **A, B**.

ExpAt[A]: Returns exponential matrix of square matrix **A**.

Observable[A, C]: Returns a logic (boolean) value representing the complete state observability of system **A, C**.

OutControllable[A, B, C, D]: Returns a logic (boolean) value representing the output controllability of system **A, B, C, D**.

ObsPolePlace[A, C, newpoles]: Determines state observer gain matrix using Ackermann's formula.

PolePlaceGain[A, B, newpoles]: Returns gain matirix **K** to place poles of system **A - BK** at locations specified by **newpoles**. Uses Ackermann's formula.

SysResponse[A, B, C, x0, input, s, {t, tmin, tmax}, gopts]: Graphs system output y as a function of time for the system with matrices **A**, **B**, **C**, at initial state **x0** from time **tmin** to **tmax**. Returns the time domain solutions for state x and output y.

Miscellaneous Linear Algebra Functions

Note : Some of the functions that we give here have equivalents in

***Mathematica* 2.0 and higher.**

matrixpower[A, n]: Returns the matrix A^n, where n is a positive integer. This function is used by the COSY_PAK functions controllable, observable, placepolegain, and obspoleplace functions.

rank[A]: Returns the rank of matrix **A**. The function rank returns the integer value corresponding to the number of linearly independent rows in the matrix A. The rank function is used by the functions observable, controllable, and outcont.

sspace[a,b]: This function is equivalent to the other **COSY_PAK** function **ODE2SS**. The function **sspace** returns the single input, single output state space form of the ordinary differential equation such as y''' + a2 y'' + a3 y' + a4 y = b1 u' + b2 u with the coefficients of y and its derivatives given by the list a and the coefficients of u and its derivatives given by the list b. The list a must start with a 1 and the list b should be padded with leading zeroes to make it the same length as a. The A, B, C,and D matrices of the equations

dx/dt = Ax + Bu

y = Cy + Du

are returned as the global variables AOUT, BOUT, COUT, and DOUT, representing a system with scalar input u and output y.

tpose[A]: Returns the transpose of matrix **A**.