Appendix-B

MATLAB Functions

Some of the MATLAB functions that have been used in this thesis work have been given in this appendix.

1. **eig**: Eigenvalues and eigenvectors

e = eig(A) returns a column vector containing the eigenvalues of square matrix A

[V,D] = eig(A) returns diagonal matrix D of eigenvalues and matrix V whose columns are the corresponding right eigenvectors, so that A*V = V*

[V,D,W] = eig(A) also returns full matrix W whose columns are the corresponding left eigenvectors, so that W'*A = D*W'.

The eigenvalue problem is to determine the solution to the equation $Av = \lambda v$, where A is an n-by-n matrix, v is a column vector of length n, and λ is a scalar. The values of λ that satisfy the equation are the eigenvalues. The corresponding values of v that satisfy the equation are the right eigenvectors. The left eigenvectors, w, satisfy the equation $w'A = \lambda w'$.

2. **lsim**: Simulate time response of dynamic system to arbitrary inputs

lsim simulates the (time) response of continuous or discrete linear systems to arbitrary inputs. When invoked without left-hand arguments, it plots the response on the screen.

lsim(sys,u,t) produces a plot of the time response of the dynamic system model sys to the input history, t,u. The vector t specifies the time samples for the simulation (in system time units, specified in the TimeUnit property of sys), and consists of regularly spaced time samples, i.e., t = 0:dt:Tfinal

The input u is an array having as many rows as time samples (length(t)) and as many columns as system inputs. The model sys can be continuous or discrete, SISO or MIMO. lsim(sys,u,t,x0) further specifies an initial condition x0 for the system states. This syntax applies only when sys is a state-space model. x0 is a vector whose entries are the initial values of the corresponding states of sys.

3. **hsvd**: Hankel singular values of dynamic system

sv = hsvd(*sys*) computes the Hankel singular values hsv of the dynamic system sys. In state coordinates that equalize the input-to-state and state-to-output energy transfers, the Hankel singular values measure the contribution of each state to the input/output behavior. In particular, small Hankel singular values signal states that can be discarded to simplify the model. For models with unstable poles, hsvd only computes the Hankel singular values of the stable part and entries of hsv corresponding to unstable modes are set to Inf.

hsvd(___) displays a Hankel singular values plot.

4. **balred**: Model order reduction

rsys = balred(*sys*,*ORDERS*) computes a reduced-order approximation rsys of the LTI model sys. The desired order (number of states) for rsys is specified by ORDERS.

5. **schurmr**: Balanced model truncation via Schur method

schurmr returns a reduced order model GRED of G and a struct array info containing the error bound of the reduced model and Hankel singular values of the original system. The error bound is computed based on Hankel singular values of G. For a stable system Hankel singular values indicate the respective state energy of the system. Hence, reduced order can be directly determined by examining the system Hankel SV's, σ. This method guarantees an error bound on the infinity norm of the additive error $\|G - G_{red}\|_{\infty}$ for well-conditioned model reduced problems:

$$\|G - G_{red}\|_{\infty} \leq 2 \sum_{k+1}^{n} \sigma_i$$

6. **fotf**: Creates a new fractional-order transfer function object.

G = fotf(A, NA, B, NB, T)

G = fotf(BPOLY, APOLY, T)

G = fotf(H)

where G,H - FOTF objects; A, NA, B, NB - vectors containing pole polynomial coefficients and exponents and zero polynomial coefficients and exponents respectively; BPOLY, APOLY - polynomial strings; T - input/output delay (sec).

7. **fracpid**: Create a new fractional-order PID controller in parallel form.

A parallel form of the fractional-order PID controller is assumed,

i.e. $K_{FOPID}(s) = k_P + \frac{k_I}{s^{\lambda}} + k_D s^{\mu}$

[PIDCTRL, CTRLTYPE] = fracpid(KP, KI, LAMBDA, KD, MU)

CTRLTYPE will contain a string with controller type.

8. **augw**: State-space or transfer function plant augmentation for use in weighted mixed-sensitivity H_∞ and H_2 loopshaping design

P = augw(G,W1,W2,W3) computes a state-space model of an augmented LTI plant $P(s)$ with weighting functions $W_1(s)$, $W_2(s)$, and $W_3(s)$ penalizing the error signal, control signal and output signal respectively (see block diagram) so that the closed-loop transfer function matrix is the weighted mixed sensitivity.

9. **hinfsyn**: Compute H-infinity optimal controller

[K,CL,gamma]=hinfsyn(P,nmeas,ncont) computes a stabilizing H_∞-optimal controller K for the plant P. The plant has a partitioned form

$$\begin{bmatrix} z \\ y \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} w \\ u \end{bmatrix}$$

where: w represents the disturbance inputs, u represents the control inputs, z represents the error outputs to be kept small, y represents the measurement outputs provided to the controller.

nmeas and ncont are the number of signals in y and u, respectively. y and u are the last outputs and inputs of P, respectively. hinfsyn returns a controller K that stabilizes P and has the same number of states. The closed-loop system CL = lft(P,K) achieves the performance level gamma, which is the H_∞ norm of CL.

10. **norm**: Norm of matrix or vector

norm(A) returns the 2-norm of matrix A.

norm(A,p) returns the p-norm of matrix A.

norm(V) returns the 2-norm of vector V.

norm(V,P) returns the P-norm of vector V.

11. **hinfnorm**: H_∞ norm of dynamic system

hinf = hinfnorm(sys) returns the H_∞ in absolute units of the dynamic system model, sys. If sys is a stable SISO system, then the H_∞ norm is the peak gain, the largest value of the frequency response magnitude. If sys is a stable MIMO system, then the H_∞ norm is the largest singular value across frequencies. If sys is an unstable system, then the H_∞ norm is defined as Inf.