A Sliding Mode Control Matlab Toolbox

Christopher Edwards & Sarah K. Spurgeon

Control Systems Research
Department of Engineering
University of Leicester
Leicester LE1 7RH
U.K.

Revised 21/8/98

Report No : 98-9

Abstract

This document serves as a manual for a suite of MATLAB functions for designing sliding mode controllers for linear systems with bounded nonlinear uncertainty. The algorithms are described in detail elsewhere and in this document the emphasis is on describing the software which implements these ideas. A number of worked examples are included to demonstrate different facets of the design procedures together with diaries of relevant MATLAB sessions.

1 An Introduction to Variable Structure Control

The focus of much of the research in the area of control systems theory during the seventies and eighties has addressed the issue of *robustness* – i.e. designing controllers with the ability to maintain stability and performance in the presence of discrepancies between the plant and model. One nonlinear approach to robust controller design which emerged during this period is the Variable structure control systems methodology. Variable structure control systems evolved from the pioneering work in Russia of Emel'yanov and Barbashin in the early 1960's. The ideas did not appear outside the Soviet Union until the mid 1970's when a book by Itkis [2] and a survey paper by Utkin [4] were published in English. Variable structure systems concepts have subsequently been utilised in the design of robust regulators, model-reference systems, adaptive schemes, tracking systems and state observers. The ideas have successfully been applied to problems as diverse as automatic flight control, control of electrical motors, chemical processes, helicopter stability augmentation, space systems and robotics.

Variable structure control systems comprise a collection of different, usually quite simple, feedback control laws and a decision rule. Depending on the status of the system, a decision rule, often termed the *switching function*, determines which of the control laws is 'on-line' at any one time. Unlike, for example, a gain-scheduling methodology, the decision rule is designed to force the system states to reach, and subsequently remain on, a pre-defined surface within the state-space. The dynamical behaviour of the system when confined to the surface is described as the *ideal sliding motion*. The advantages of obtaining such a motion are twofold: firstly there is a reduction in order and secondly the sliding motion is insensitive to parameter variations implicit in the input channels. The latter property of invariance towards so-called *matched uncertainty* makes the methodology an attractive one for designing robust controllers for uncertain systems.

The following notation (which is explained in detail in [1]) will be used throughout: $\mathbb{R}$ and $\mathbb{C}$ will represent the field of real and complex numbers respectively and $\mathbb{C}_-$ will represent the open left half of the complex plane; $\mathcal{N}(\cdot)$ and $\mathcal{R}(\cdot)$ will denote the null and range spaces of a matrix respectively and $\|\cdot\|$ will represent the Euclidean norm for vectors and the induced spectral norm for matrices.

2 State-feedback Sliding Mode Control

2.1 Introduction

Consider the nth order linear time invariant system with m inputs given by

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ with $1 \leq m \leq n$. Without loss of generality it can be assumed that the input distribution matrix B has full rank. Define a switching function $s : \mathbb{R} \to \mathbb{R}^m$ to be

$$s(t) = Sx(t) \tag{2}$$

where $S \in \mathbb{R}^{m \times n}$ is of full rank and let $\mathcal{S}$ be the hyperplane defined by

$$\mathcal{S} = \{x \in \mathbb{R}^n \quad : \quad Sx = 0\} \tag{3}$$

Suppose $u(s(t), x(t))$ represents a variable structure control law where the changes in control strategy depend on the value of the switching function. It is natural to explore the possibility of choosing the control action and selecting the switching strategy so that an *ideal sliding motion* takes place on the hyperplane, i.e. there exists a time t_s such that

$$s(t) = Sx(t) = 0 \qquad \text{for all } t > t_s \tag{4}$$

Suppose at time $t = t_s$ the systems states lie on the surface $\mathcal{S}$ and an ideal sliding motion takes place. This can be expressed mathematically as $Sx(t) = 0$ and $\dot{s}(t) = S\dot{x}(t) = 0$ for all $t \geq t_s$. Substituting for $\dot{x}(t)$ from (1) gives

$$S\dot{x}(t) = SAx(t) + SBu(t) = 0 \qquad \text{for all } t \geq t_s \tag{5}$$

Suppose the matrix S is designed so that the square matrix SB is nonsingular (in practise this is easily accomplished since B is full rank and S is a free parameter). The *equivalent control*, written as u_{eq}, is defined to be the unique solution to the algebraic equation (5), namely

$$u_{eq}(t) = -(SB)^{-1} SAx(t) \tag{6}$$

This represents the control action which is required to maintain the states on the switching surface. The ideal sliding motion is then given by substituting the expression for the equivalent control into equation (1) which results in a free motion

$$\dot{x}(t) = \left(I_n - B(SB)^{-1} S\right) Ax(t) \qquad \text{for all } t \geq t_s \text{ and } Sx(t_s) = 0 \tag{7}$$

It can be seen from equation (7) that the sliding motion is a control independent free motion which depends on the choice of sliding surface, although the precise effect is not readily apparent. A convenient way to shed light on the problem is to first transform the system into a suitable canonical form. In this form the system is decomposed into two connected subsystems, one acting in $\mathcal{R}(B)$ and the other in $\mathcal{N}(S)$. Since by assumption $\text{rank}(B) = m$ there exists an orthogonal matrix $T_r \in \mathbb{R}^{n \times n}$ such that

$$T_r B = \begin{bmatrix} 0 \\ B_2 \end{bmatrix} \tag{8}$$

where $B_2 \in \mathbb{R}^{m \times m}$ and is nonsingular. Let $z = Tx$ and partition the new coordinates so that

$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{9}$$

where $z_1 \in \mathbb{R}^{n-m}$ and $z_2 \in \mathbb{R}^m$. The nominal linear system (1) can then be written as

$$\dot{z}_1(t) = A_{11} z_1(t) + A_{12} z_2(t) \tag{10}$$

$$\dot{z}_2(t) = A_{21} z_1(t) + A_{22} z_2(t) + B_2 u(t) \tag{11}$$

which is referred to as *regular form*. Equation (10) is referred to as describing the *null-space dynamics* and equation (11) as describing the *range-space dynamics*. Suppose the matrix defining the switching function (in the new coordinate system) is compatibly partitioned as

$$ST_r^{\mathrm{T}} = \begin{bmatrix} S_1 & S_2 \end{bmatrix} \tag{12}$$

where $S_1 \in \mathbb{R}^{m\times(n-m)}$ and $S_2 \in \mathbb{R}^{m\times m}$. Since $SB = S_2B_2$ it follows that a necessary and sufficient condition for the matrix SB to be nonsingular is that $\det(S_2) \neq 0$. By design assume this to be the case. During an ideal sliding motion

$$S_1z_1(t) + S_2z_2(t) = 0 \qquad \text{for all } t > t_s \tag{13}$$

and therefore formally expressing $z_2(t)$ in terms of $z_1(t)$ yields

$$z_2(t) = -Mz_1(t) \tag{14}$$

where $M = S_2^{-1}S_1$. Substituting in equation (10) gives

$$\dot{z}_1(t) = (A_{11} - A_{12}M)\, z_1(t) \tag{15}$$

and thus the problem of hyperplane design may be considered to be a state feedback problem for the system (10) where $z_2(t)$ is considered to play the role of the control action. In the context of designing a regulator, the matrix governing the sliding motion $(A_{11} - A_{12}M)$ must have stable eigenvalues. The switching surface design problem can therefore be considered to be one of choosing a state feedback matrix M to stabilise the reduced order system (A_{11}, A_{12}). Because of the special structure of the regular form, it follows that the pair (A_{11}, A_{12}) is controllable if and only if (A, B) is controllable. It can be seen from equation (15) that S_2 has no direct effect on the dynamics of the sliding motion and acts only as a scaling factor for the switching function. The choice of S_2 is therefore somewhat arbitrary. A common choice however, which stems from the so-called *hierarchical* design procedure, is to let $S_2 = \Lambda B_2^{-1}$ for some diagonal design matrix $\Lambda \in \mathbb{R}^{m\times m}$ which implies $SB = \Lambda$. By selecting M and S_2 the switching function in equation (12) it completely determined.

Several approaches have been proposed for the design of the feedback matrix M including quadratic minimisation, eigenvalue placement and eigenstructure assignment methods (see Chapter 4 in [1]). These approaches will be discussed in the following sections.

Example

A linearisation of the nonlinear equations of motion of an inverted pendulum about the equilibrium point at the origin (§5.6.3 in [1]) is given by

$$A = \begin{bmatrix} 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 1.0000 \\ 0 & -1.9333 & -1.9872 & 0.0091 \\ 0 & 36.9771 & 6.2589 & -0.1738 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0 \\ 0.3205 \\ -1.0095 \end{bmatrix} \tag{16}$$

This system is not in regular form because the input enters the last two state-space equation. The following MATLAB session performs the necessary transformation and identifies the important sub-blocks.

Session 1: regfor

```
>> [A11,A12,B2,Tr]=regfor(A,B)

A11 =
           0    0.2884    0.0916
      8.9089   -0.2870   -0.0911
      2.8286    0.8620    0.2737

A12 =
      0.9531
      0.0517
     -0.3011

B2 =
     -1.0592

Tr =
           0    1.0000         0         0
     -0.3026         0    0.9084    0.2884
      0.9531         0    0.2884    0.0916
           0         0   -0.3026    0.9531

>>
```

Notice the transformation T_r is symmetric and in fact orthogonal. This is desirable from the numerical viewpoint.

2.2 Robust Eigenstructure Assignment

For the case of a scalar controlled problem, specification of the $(n-1)$ eigenvalues associated with the sliding mode will uniquely determine the matrix M of equation (14). For multi-input systems this is not the case. In such a situation the available degrees of freedom may be used to modally shape the system response by a judicious choice of eigenvector form and/or ensure that the resulting closed-loop system is maximally robust to system parameter variations. However the eigenvector corresponding to a given eigenvalue must lie in an allowable sub-space which is determined by the system matrix, the input matrix and the eigenvalue itself. To evaluate robustness, a bound upon the individual eigenvalue sensitivities c_i is given by

$$\max\{c_i\} \leq \kappa(V) = \|V\|\|V^{-1}\| \tag{17}$$

Here $\kappa(V)$ denotes the condition number of the matrix V of right eigenvectors and is a measure of the orthogonality of the eigenvectors v_i. The closer the eigenvectors of a matrix are to being orthogonal, the smaller is the associated condition number and the greater the robustness of the eigenvalue locations to changes in the elements of the matrix. In robust eigenstructure assignment the feedback matrix is obtained by assigning a set of linearly independent right eigenvectors corresponding to the

required eigenvalues such that the matrix of eigenvectors is as well-conditioned as possible (§4.2.1 in [1]). This can be achieved via the MATLAB command **place**.

Example

Here the use of robust pole placement will be demonstrated on a dc motor example from §3.6.4 in[1]. The state and input distribution matrices are given by

$$A = \begin{bmatrix} 0 & 1.0000 & 0 \\ 0 & 0 & 4.4379 \\ 0 & -12.0000 & -24.0000 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0 \\ 20 \end{bmatrix} \tag{18}$$

Notice the system is already in regular form and the sliding motion will be a second order system. In the following session the reduced order sliding motion is assigned a pair of complex poles with damping ratio $\zeta = 0.95$ and natural frequency $w_n = 2$.

Session 2: rpp

```
zeta=0.95;
wn=2.0;
>> sp=roots([1 2*zeta*wn wn*wn])

sp =
    -1.9000 + 0.6245i
    -1.9000 - 0.6245i

>> S=rpp(A,B,sp)
place: ndigits= 16

S =
    -0.9013   -0.8563   -1.0000
>>
```

This compares with the (specific) analytic approach described in §3.6.4 [1]. Notice the hyperplane matrix S is returned in the original coordinate system. The transformation to regular form and the subsequent computations take place internally within the function.

2.3 Direct Eigenstructure Assignment

In the previous subsection, the additional degrees of freedom in the pole placement problem for multi-input systems were used to minimise the condition number of the associated eigenvalues. If information is known about a desirable weighting of the system states for each mode, it is possible to choose a desired eigenvector specification. Again this will not necessarily be achievable because it may not lie within the prescribed allowable sub-space: details are given in §4.3 in [1].

A numerical example using the approach appears later in §2.6.

2.4 Quadratic Minimisation

Consider the problem of minimising the quadratic performance index

$$J = \frac{1}{2}\int_{t_s}^{\infty} x(t)^{\mathrm{T}} Q x(t)\ dt \tag{19}$$

where Q is both symmetric and positive definite and t_s is the time at which sliding motion commences. The aim is to minimise equation (19) subject to the system equation (1) under the assumption that sliding takes place. It is assumed that the state of the system at time t_s, $x(t_s)$, is a known initial condition and is such that $x(t) \to 0$ as $t \to \infty$. The matrix Q from equation (19) is transformed and partitioned compatibly with z so that

$$T_r Q T_r^{\mathrm{T}} = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^{\mathrm{T}} & Q_{22} \end{bmatrix} \tag{20}$$

and subsequently define

$$\hat{Q} = Q_{11} - Q_{12}Q_{22}^{-1}Q_{21} \tag{21}$$

and

$$v = z_2 + Q_{22}^{-1}Q_{21}z_1 \tag{22}$$

After some algebraic manipulation equation (19) may then be written in the new coordinate system as

$$J = \frac{1}{2}\int_{t_s}^{\infty} z_1^{\mathrm{T}}\hat{Q}z_1 + v^{\mathrm{T}}Q_{22}v\ dt \tag{23}$$

Recall the constraint equation may be written as

$$\dot{z}_1(t) = A_{11}z_1(t) + A_{12}z_2(t) \tag{24}$$

Eliminating the z_2 contribution from equation (24) using equation (22), the modified constraint equation becomes

$$\dot{z}_1(t) = \hat{A}z_1(t) + A_{12}v(t) \tag{25}$$

where

$$\hat{A} = A_{11} - A_{12}Q_{22}^{-1}Q_{21} \tag{26}$$

The positive definiteness of Q ensures that $Q_{22} > 0$, so that Q_{22}^{-1} exists, and also that $\hat{Q} > 0$. Furthermore, the controllability of the original (A, B) pair ensures that the pair $(\hat{A}, A_{12})$ is controllable. The problem thus becomes that of minimising the functional (23) subject to the system (25) and thus can be interpreted as a standard linear-quadratic optimal state-regulator problem. A more detailed description of this approach is given in §4.2.2 in [1].

Example

Consider once again the linearised equations of motion of an inverted pendulum given in (16). The following session designs a hyperplane for this system by minimising a quadratic cost function.

Session 3: lqcf

```
>> Q=diag([10 1 1 0.1])

Q =
    10.0000         0         0         0
          0    1.0000         0         0
          0         0    1.0000         0
          0         0         0    0.1000

>> [S,E]=lqcf(A,B,Q)

S =
     7.4040   14.9867    5.3020    2.7326

E =
    -4.3241 + 1.7852i
    -4.3241 - 1.7852i
    -3.1623
```

Notice the hyperplane matrix S is returned in the original coordinate system and all the transformations take place internally. The remainder of this section is devoted to the synthesis of a control law which will theoretically bring about ideal sliding.

2.5 State-feedback Control Laws

Of the many different multivariable sliding mode control structures which exist the one that will be considered here is essentially that of Ryan & Corless [3] and may be described as a *unit vector* approach. Consider an uncertain system of the form

$$\dot{x}(t) = Ax(t) + Bu(t) + f(t, x, u) \tag{27}$$

where the function $f : \mathbb{R} \times \times \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^m$ which represents the uncertainties or nonlinearities satisfies the so-called *matching condition*, i.e.

$$f(t, x, u) = B\xi(t, x, u) \tag{28}$$

where $\xi : \mathbb{R} \times \times \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^m$ and is unknown but satisfies

$$\|\xi(t, x, u)\| \leq k_1\|u\| + \alpha(t, x) \tag{29}$$

where $1 > k_1 \geq 0$ is a known constants and $\alpha(\cdot)$ is a known function. The proposed control law comprises two components; a linear component to stabilise the nominal linear system; and a discontinuous component. Specifically

$$u(t) = u_l(t) + u_n(t) \tag{30}$$

where the linear component is given by

$$u_l(t) = -\Lambda^{-1}\left(SA - \Phi S\right)x(t) \tag{31}$$

where $\Phi \in \mathbb{R}^{m\times m}$ is any stable design matrix and $\Lambda = SB$. The nonlinear component is defined to be

$$u_n(t) = -\rho(t,x)\Lambda^{-1}\frac{P_2 s(t)}{\|P_2 s(t)\|} \qquad \text{for } s(t) \neq 0 \tag{32}$$

where $P_2 \in \mathbb{R}^{m\times m}$ is a symmetric positive definite matrix satisfying the Lyapunov equation

$$P_2\Phi + \Phi^{\mathrm{T}}P_2 = -I \tag{33}$$

and the scalar function $\rho(t,x)$, which depends only on the magnitude of the uncertainty, is any function satisfying

$$\rho(t,x) \geq \left(k_1\|u_l\| + \alpha(t,x) + \gamma\right)/(1 - k_1\kappa(\Lambda)) \tag{34}$$

where $\gamma > 0$ is a design parameter. In this equation it is assumed that the scaling parameter has been chosen so that

$$k_1\kappa(\Lambda) < 1$$

It can easily be established that any function satisfying equation (34) also satisfies

$$\rho(t,x) \geq \|\xi(t,x,u)\| + \gamma \tag{35}$$

and therefore $\rho(t,x)$ is greater in magnitude than the matched uncertainty occurring in equation (28). It is straightforward to verify that $V(s) = s^{\mathrm{T}}P_2 s$ guarantees quadratic stability for the switching states s and in particular

$$\dot{V} \leq -s^{\mathrm{T}}s - 2\gamma\|P_2 s\| \tag{36}$$

This control law guarantees that the switching surface is reached in finite time despite the disturbance or uncertainty and once the sliding motion is attained it is completely independent of the uncertainty. The relationship between the control law in (31)–(32) and the original description of Ryan & Corless is described in §3.6.3 in [1].

The following session considers the inverted pendulum from Session 3 and constructs the components of the control law described in (31) – (32).

Session 4: contl

```
>> Phi=-6;
>> [L,P,Lam]=contl(A,B,S,Phi)

L =
     41.9427  170.6191   43.2255   29.2265

P =
      0.0833

Lam =
     -0.9441

>>
```

Thus far only regulation problems have been considered. In practise tracking problems are often encountered whereby (usually) the output of the system is required to follow a pre-defined reference signal. One approach is to use a model-following sliding mode control law. In this situation a state error system is formed between the plant states and the ideal model states and a sliding mode (regulation-like) control law is designed to force the error system to zero (§4.4.1 in [1]). In the following section an integral action based method is considered for output tracking.

2.6 Output Tracking with Integral Action

Consider the development of a tracking control law for the nominal linear system

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{37}$$
$$y(t) = Cx(t) \tag{38}$$

which is assumed to be square. In addition, for convenience, assume the matrix pair (A, B) is in regular form. The control law described here utilises an *integral action* methodology. Consider the introduction of additional states $x_r \in \mathbb{R}^p$ satisfying

$$\dot{x}_r(t) = r(t) - y(t) \tag{39}$$

where the differentiable signal $r(t)$ satisfies

$$\dot{r}(t) = \Gamma\left(r(t) - R\right) \tag{40}$$

with $\Gamma \in \mathbb{R}^{p\times p}$ a stable design matrix and R a constant demand vector. Augment the states with the integral action states and define

$$\tilde{x} = \begin{bmatrix} x_r \\ x \end{bmatrix} \tag{41}$$

The associated system and input distribution matrices for the *augmented system* are

$$\tilde{A} = \begin{bmatrix} 0 & -C \\ 0 & A \end{bmatrix} \qquad \text{and} \qquad \tilde{B} = \begin{bmatrix} 0 \\ B \end{bmatrix} \tag{42}$$

and assuming the pair (A, B) is in regular form, the pair $(\tilde{A}, \tilde{B})$ is in regular form. The proposed controller seeks to induce a sliding motion on the surface

$$\mathcal{S} = \{\tilde{x} \in \mathbb{R}^{n+p} \;\; : \;\; S\tilde{x} = S_r r\} \tag{43}$$

where $S \in \mathbb{R}^{m\times(n+p)}$ and $S_r \in \mathbb{R}^{p\times p}$ are design parameters which govern the reduced order motion. Partition the hyperplane system matrix as

$$S = \begin{array}{c} \overset{n}{\longleftrightarrow} \quad \overset{m}{\longleftrightarrow} \\ \begin{bmatrix} S_1 & S_2 \end{bmatrix} \end{array} \tag{44}$$

and the system matrix

$$\tilde{A} = \begin{array}{c} \overset{n}{\longleftrightarrow} \quad \overset{m}{\longleftrightarrow} \\ \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \begin{array}{l} \updownarrow n \\ \updownarrow m \end{array} \end{array} \tag{45}$$

and assume $\Lambda = S\tilde{B}$ is nonsingular. If a controller exists which induces an ideal sliding motion on $\mathcal{S}$ and the augmented states are partitioned as

$$\tilde{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{46}$$

where $x_1 \in \mathbb{R}^n$ and $x_2 \in \mathbb{R}^m$, then the ideal sliding motion is given by

$$\dot{x}_1(t) = (\tilde{A}_{11} - \tilde{A}_{12}M)x_1(t) + \left(\tilde{A}_{12}S_2^{-1}S_r + B_r\right) r(t) \tag{47}$$

where $M = S_2^{-1}S_1$ and $B_r = \begin{bmatrix} I_p & 0_{n\times p} \end{bmatrix}^{\mathrm{T}}$. In order for the hyperplane design methods described earlier to be valid, it is necessary for the matrix pair $(\tilde{A}_{11}, \tilde{A}_{12})$ to be completely controllable. Necessary conditions on the original system are that (A, B, C) is completely controllable and has no invariant zeros at the origin (§4.4.2 in [1]).

The development that follows mirrors the approach in §2.5 where Φ is any stable design matrix. The overall control law is then given by

$$u = u_l(\tilde{x}, r) + u_n(\tilde{x}, r) \tag{48}$$

where the discontinuous vector

$$u_n(s, r) = \begin{cases} -\rho_c(u_L, y)\Lambda^{-1}\frac{P_2(S\tilde{x} - S_r r)}{\|P_2(S\tilde{x} - S_r r)\|} & \text{if } S\tilde{x} \neq S_r r \\ 0 & \text{otherwise} \end{cases} \tag{49}$$

where P_2 is a symmetric positive definite matrix satisfying

$$P_2\Phi + \Phi^{\mathrm{T}}P_2 = -I \tag{50}$$

The positive scalar function which multiplies the unit vector component can be obtained from arguments similar to those in in §2.5. It follows that, in terms of the original coordinates

$$u_n(\tilde{x}, r) = L\tilde{x} + L_r r + L_{\dot{r}}\dot{r} \tag{51}$$

with gains defined as

$$L = -\Lambda^{-1}(S\tilde{A} - \Phi S) \tag{52}$$

$$L_r = -\Lambda^{-1}\left(\Phi S_r + S_1 B_r\right) \tag{53}$$

$$L_{\dot{r}} = \Lambda^{-1}S_r \tag{54}$$

The parameter S_r can take any value and does not affect the stability of the closed loop system. One common choice is to let $S_r = 0$ for simplicity. Another option, which has been found to give good results with practical applications, is to choose S_r so that at steady state the integral action states are zero in the absence of any uncertainty (§7.3.4 in [1]).

This session repeats the design aircraft design study in §4.5 in [1] which uses integral action and a hyperplane calculated using direct eigenstructure assignment. The system triple given below represents the longitudinal motion of of an aircraft at

0.77 Mach and the outputs correspond to flight path angle and pitch angle

$$A = \begin{bmatrix} 0 & 0 & 1.7400 & 0.0800 & 0.5900 \\ 0 & -1.9900 & -13.4100 & -18.9500 & -3.6000 \\ 0 & 1.0000 & -1.7400 & -0.0800 & -0.5900 \\ 0 & 0 & 0 & -20.0000 & 0 \\ 0 & 0 & 0 & 0 & -20.0000 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 20 & 0 \\ 0 & 20 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{55}$$

First the augmented system represented by (Aa, Ba) is constructed. The command *eigbuil* is then used to form the matrix which contains the information identifying the eigenvector components which are to be specified and the associated desired values. For more details of the design and closed-loop simulations see §4.5 in [1].

Session 5: intac, eigbiul and dea

```
>> [Aa,Ba]=intac(A,B,C)

Aa =
     0         0   -1.0000         0   -1.0000         0         0
     0         0   -1.0000         0         0         0         0
     0         0         0         0    1.7400    0.0800    0.5900
     0         0         0   -1.9900  -13.4100  -18.9500   -3.6000
     0         0         0    1.0000   -1.7400   -0.0800   -0.5900
     0         0         0         0         0  -20.0000         0
     0         0         0         0         0         0  -20.0000

Ba =
     0     0
     0     0
     0     0
     0     0
     0     0
    20     0
     0    20

>> nocomp=1;
>> specent=eigbuil(7,2,nocomp)

Eigenvector 1 (Real Part)
Enter position of specified entry [0 to terminate]
Followed by the value of specified entry

Enter position 3
Enter associated value 0

Enter position 4
Enter associated value 1

Enter position 0
--------------------------------------------------
```

```
Eigenvector 1 (Imaginary part)
Enter position of specified entry [0 to terminate]
Followed by the value of specified entry

Enter position 3
Enter associated value 0

Enter position 5
Enter associated value 1

Enter position 0
--------------------------------------------------

Eigenvector 2
Enter position of specified entry [0 to terminate]
Followed by the value of specified entry

Enter position 3
Enter associated value 1

Enter position 4
Enter associated value 0

Enter position 5
Enter associated value 0

Enter position 0
--------------------------------------------------

Eigenvector 3
Enter position of specified entry [0 to terminate]
Followed by the value of specified entry

Enter position 3
Enter associated value 0

Enter position 4
Enter associated value 1

Enter position 0
--------------------------------------------------

Eigenvector 4
Enter position of specified entry [0 to terminate]
Followed by the value of specified entry

Enter position 3
Enter associated value 1

Enter position 4
Enter associated value 0
```

```
Enter position 5
Enter associated value 0

Enter position 0
--------------------------------------------------


specent =
      NaN   NaN   NaN   NaN   NaN
      NaN   NaN   NaN   NaN   NaN
        0     0     1     0     1
        1   NaN     0     1     0
      NaN     1     0   NaN     0
      NaN   NaN   NaN   NaN   NaN
      NaN   NaN   NaN   NaN   NaN

>> lambda=[-5.6 4.2 -1 -0.4 -0.7]

lambda =
   -5.6000    4.2000   -1.0000   -0.4000   -0.7000

>> [S,V]=dea(Aa,Ba,specent,lambda,1)

S =
    0.0075    0.2640   -0.6566   -0.0037   -0.6707   -0.0229   -0.2209
    0.5218   -0.3661   -0.1946   -0.2559   -0.1042    0.5488    0.4222


V =
   -0.1918    0.0347    0.3333   -6.2500    0.7215
    0.0000    0.0000    0.6667    0.0000    1.0750
   -0.0000    0.0000    0.6667    0.0000    0.7525
    1.0000   -9.5000   -0.3333    1.0000   -0.3535
   -0.9286    1.0000   -0.3333   -2.5000   -0.2475
   -1.8252   -2.2364    0.2886    0.2921    0.2362
    2.9860   -2.6459   -0.1860    7.3333   -0.1950

>>
```

The vector V represents the achieved eigenstructure. Notice when only two elements have been specified in an eigenvector the requirements have been achieved exactly. This is not the case for the two eigenvectors in which three elements were specified. This is to be expected as a maximum of m entries can be specified exactly.

The associated control law which includes feed-forward terms for both the reference and its derivative can be obtained as shown below. The hyperplane is first scaled so that $\Lambda = I_2$ and the two range-space poles are selected to be at -20.

Session 6: contlia

```
>> S=inv(S*Ba)*S
S =
```

```
    0.0531    0.0137   -0.1436   -0.0260   -0.1373    0.0500         0
   -0.0072   -0.0612    0.1635    0.0035    0.1661         0    0.0500

>> [L,Lr,Lrdot,Sr,Lam,P]=contlia(A,B,C,S,-20*eye(2))

L =
   -1.0616   -0.2743    2.9379    0.6060    2.4605   -0.4927   -0.0900
    0.1440    1.2236   -3.3390   -0.2296   -3.2769    0.0671    0.0142

Lr =
   -2.9499    0.0120
    0.4000    2.9391

Lrdot =
   -0.1448    0.0013
    0.0196    0.1439

Sr =
   -0.1448    0.0013
    0.0196    0.1439

Lam =
    1.0000         0
         0    1.0000

P =
    0.0250         0
         0    0.0250
>>
```

The following session reconstructs the case study from Chapter 9 in [1] and designs a state feedback output tracking controller utilising integral action for the furnace system linearisation

$$A = \begin{bmatrix} -0.0186 & -0.0065 & 0.0190 & 0.0129 \\ 0.0026 & -0.1354 & 0.0310 & 0.0040 \\ -0.0972 & 0.0695 & -0.1273 & 0.0530 \\ -0.0193 & -0.0155 & -0.1121 & -0.4934 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & -0.0960 \\ 0.4969 & 0.0453 \end{bmatrix}$$

$$C = \begin{bmatrix} 0.6707 & -0.1085 & -0.0286 & 0.0086 \\ -0.2750 & -0.1933 & -0.2175 & 0.0060 \end{bmatrix} \tag{56}$$

The robust pole placement method has been chosen for the design of the switching function. Once the matrix S has been calculated the 'feedback' gain matrix M is computed (this is easily accomplished in this case since the system is in regular form) and the system matrix governing the reduced order sliding motion $(A_{11} - A_{12}M)$ is formed. The condition number of the associated eigenvectors is also computed purely for pedagogical purposes to verify that a robust design has been achieved.

Session 7: intac and rpp

```
>> sp=[-0.0849 -0.0118 -0.0357+0.0220*j -0.0357-0.0220*j];
>> [Aa,Ba]=intac(A,B,C);
>> S=rpp(Aa,Ba,sp)
place: ndigits= 17

S =
       0.1659    0.0407   -8.4771   -4.2487   -0.0000   -1.0000
       0.0389    0.0414   -1.5359   -3.0467    1.0000   -0.0000

>> M=inv(S(:,5:6))*S(:,1:4)

M =
       0.0389    0.0414   -1.5359   -3.0467
      -0.1659   -0.0407    8.4771    4.2487

>> A11=Aa(1:4,1:4)

A11 =
       0         0   -0.6707    0.1085
       0         0    0.2750    0.1933
       0         0   -0.0186   -0.0065
       0         0    0.0026   -0.1354

>> A12=Aa(1:4,5:6)

A12 =
       0.0286   -0.0086
       0.2175   -0.0060
       0.0190    0.0129
       0.0310    0.0040

>> eig(A11-A12*M)

ans =
      -0.0849
      -0.0118
      -0.0357 + 0.0220i
      -0.0357 - 0.0220i

>> [V,D]=eig(A11-A12*M);
>> cond(V)

ans =
      52.4555

>>
```

The condition number of the eigenvectors is reasonably small indicating that the design is relatively robust with respect to unmatched uncertainty.

Up to this point it has been assumed that all the states are available for use in the control law. This hypothesis will be dropped in the remaining sections.

3 Output Based Hyperplane Design

Consider the linear system in (1) and suppose that only the measured outputs

$$y(t) = Cx(t) \tag{57}$$

where $C \in \mathbb{R}^{p \times n}$ are available. One strategy is to restrict the class of sliding surfaces and control laws to those which require only output information. Recent work has isolated the class of systems for which this approach is applicable and a design framework has been proposed for synthesising appropriate sliding surfaces and control laws (§5.4 in [1]). Here the case when there are more outputs than inputs is considered since, in the square case, no design freedom exists in terms of selecting the dynamics of the sliding motion. Two assumptions will be made:

A1) the Markov parameter CB is full rank

A2) any invariant zeros of (A, B, C) are in the $\mathbb{C}_-$

These assumptions will be central to the output feedback based sliding mode control and observer methods used here. The following lemma provides a canonical form for the system triple (A, B, C) which will be used in the subsequent analysis:

Lemma 1 *Let (A, B, C) be a linear system with $p > m$ and $rank(CB) = m$. Then a change of coordinates exists so that the system triple with respect to the new coordinates has the following structure:*

a) *the system matrix can be written as*

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad \text{where } A_{11} \in \mathbb{R}^{(n-m)\times(n-m)} \tag{58}$$

and the sub-block A_{11} when partitioned has the structure

$$A_{11} = \left[\begin{array}{cc|c} A^o_{11} & A^o_{12} & A^m_{12} \\ 0 & A^o_{22} & \\ \hline 0 & A^o_{21} & A^m_{22} \end{array}\right] \tag{59}$$

where $A^o_{11} \in \mathbb{R}^{r\times r}$, $A^o_{22} \in \mathbb{R}^{(n-p-r)\times(n-p-r)}$ and $A^o_{21} \in \mathbb{R}^{(p-m)\times(n-p-r)}$ for some $r \geq 0$ and the pair (A^o_{22}, A^o_{21}) is completely observable. Furthermore the invariant zeros of the system are the r eigenvalues of A^o_{11}.

b) *the input distribution matrix has the form*

$$B = \begin{bmatrix} 0 \\ B_2 \end{bmatrix} \quad \text{where } B_2 \in \mathbb{R}^{m\times m} \text{ and is nonsingular} \tag{60}$$

c) *the output distribution matrix has the form*

$$C \quad = \quad \left[\begin{array}{cc} 0 & T \end{array} \right] \qquad \textit{where } T \in \mathbb{R}^{p \times p} \textit{ and is orthogonal} \qquad (61)$$

■

Example

The following session considers the inverted pendulum example from (16) and assumes that only the first three states are measured. This session considers a change of coordinates which puts the system triple into the output-feedback canonical form.

Session 8: outfor

```
>> [Ac,Bc,Cc,Tc,r]=outfor(A,B,C)

Ac =
   -0.1452         0   30.8881   -0.4572
         0         0         0    1.0000
    1.0000         0         0    3.1496
   -0.0091         0    1.9333   -2.0158

Bc =
   -0.0000
         0
         0
   -0.3205

Cc =
     0    -1     0     0
     0     0     1     0
     0     0     0    -1

Tc =
         0         0    3.1496    1.0000
   -1.0000         0         0         0
         0    1.0000         0         0
         0         0   -1.0000         0

r =
     0

>>
```

Notice $r = 0$ which indicates that the system (A, B, C) does not have any invariant zeros.

Under the premise that only output information is available, the switching function must be of the form

$$s(t) = FCx(t) \qquad (62)$$

where $F \in \mathbb{R}^{m\times p}$. Suppose a controller exists which induces a stable sliding motion on the surface

$$\mathcal{S} = \{x \in \mathbb{R}^n \quad : \quad FCx = 0\} \tag{63}$$

For a unique equivalent control[1] to exist the matrix $FCB \in \mathbb{R}^{m\times m}$ must have full rank: this implies that $\text{rank}(CB) = m$ since $\text{rank}(F) \leq m$. Therefore in all the analysis which follows it can be assumed without loss of generality that the system is already in the canonical form of Lemma 1. Let

$$\begin{array}{cc} \overset{p-m}{\leftrightarrow} & \overset{m}{\leftrightarrow} \\ \left[\begin{array}{cc} F_1 & F_2 \end{array}\right] & = FT \end{array} \tag{64}$$

then the matrix which defines the switching function can be written as

$$FC = \left[\begin{array}{cc} F_1C_1 & F_2 \end{array}\right]$$

where

$$C_1 \quad = \quad \left[\begin{array}{cc} 0_{(p-m)\times(n-p)} & I_{(p-m)} \end{array}\right] \tag{65}$$

In this way $FCB = F_2B_2$ and in particular the square matrix F_2 is nonsingular. The canonical form of Lemma 1 can be viewed as a special case of the 'regular form' normally used in sliding mode controller design and therefore arguing as in §2.1 the reduced order sliding motion is governed by a free motion with system matrix

$$A_{11}^s \quad = \quad A_{11} - A_{12}F_2^{-1}F_1C_1 \tag{66}$$

Define $K = F_2^{-1}F_1$ then the problem of designing a suitable hyperplane is equivalent to an output feedback problem for the system (A_{11}, A_{12}, C_1). It is well established that invariant zeros play an important part in the sliding motion and, in particular, any invariant zeros of (A, B, C) must appear in the sliding mode dynamics when only output information is used. Intuitively, if the original system has invariant zeros then by Lemma 1 the eigenvalues of A_{11}^o must appear in the spectrum of A_{11}^s and so the poles of $A_{11} - A_{12}KC_1$ cannot be assigned arbitrarily by choice of K. To this end partition the matrices A_{12} and A_{12}^m as

$$A_{12} \quad = \quad \left[\begin{array}{c} A_{121} \\ A_{122} \end{array}\right] \qquad \text{and} \qquad A_{12}^m \quad = \quad \left[\begin{array}{c} A_{121}^m \\ A_{122}^m \end{array}\right] \tag{67}$$

where $A_{122} \in \mathbb{R}^{(n-m-r)\times m}$ and $A_{122}^m \in \mathbb{R}^{(n-p-r)\times(p-m)}$ and form a new sub-system represented by the triple $(\tilde{A}_{11}, A_{122}, \tilde{C}_1)$ where

$$\tilde{A}_{11} \quad = \quad \left[\begin{array}{cc} A_{22}^o & A_{122}^m \\ A_{21}^o & A_{22}^m \end{array}\right] \qquad \text{and} \qquad \tilde{C}_1 \quad = \quad \left[\begin{array}{cc} 0_{(p-m)\times(n-p-r)} & I_{(p-m)} \end{array}\right] \tag{68}$$

As a result it can be shown that the spectrum of A_{11}^s contains the invariant zeros of (A, B, C) and in particular

$$\lambda(A_{11} - A_{12}KC_1) = \lambda(A_{11}^o) \cup \lambda(\tilde{A}_{11} - A_{122}K\tilde{C}_1) \tag{69}$$

[1] For comprehensive details see Chapter 2 in Utkin [5].

It follows directly that for a stable sliding motion the invariant zeros of the system (A, B, C) must lie in the open left half plane and the triple $(\tilde{A}_{11}, A_{122}, \tilde{C}_1)$ must be stabilisable with respect to output feedback.[2] It can be shown that if the pair (A, B) is completely controllable then the pair $(\tilde{A}_{11}, A_{122})$ is completely controllable; the pair $(\tilde{A}_{11}, \tilde{C}_1)$ is completely observable by construction and thus if the triple $(\tilde{A}_{11}, A_{122}, \tilde{C}_1)$ satisfies the Kimura–Davison conditions, output feedback pole placement methods can be used to place the poles appropriately.

It is well established in the static output feedback literature that if the Kimura-Davison conditions are not satisfied by the nominal triple, the system can be augmented by an appropriately dimensioned compensator. Specifically let

$$\dot{x}_c(t) = Hx_c(t) + Dy(t) \tag{70}$$

where the matrices $H \in \mathbb{R}^{q\times q}$ and $D \in \mathbb{R}^{q\times p}$ are to be determined. Define a new hyperplane in the augmented state space, formed from the plant and compensator state spaces, as

$$\mathcal{S}_c = \{(x, x_c) \in \mathbb{R}^{n+q} \ : \ F_c x_c + FCx = 0\} \tag{71}$$

where $F_c \in \mathbb{R}^{m\times q}$ and $F \in \mathbb{R}^{m\times p}$. Define $D_1 \in \mathbb{R}^{q\times(p-m)}$ and $D_2 \in \mathbb{R}^{q\times m}$ as

$$\begin{bmatrix} D_1 & D_2 \end{bmatrix} = DT \tag{72}$$

If the the input distribution matrix is partitioned conformably as

$$A_{122} = \begin{bmatrix} A_{1221} \\ A_{1222} \end{bmatrix} \begin{matrix} \updownarrow n-p-r \\ \updownarrow p-m \end{matrix} \tag{73}$$

then it can be shown (§5.6 in [1]) that a parametrisation of the compensator is

$$H = A^o_{22} + L^o A^o_{21} \tag{74}$$
$$D_1 = A^m_{122} + L^o A^m_{22} - (A^o_{22} + L^o A^o_{21})L^o \tag{75}$$
$$D_2 = A_{1221} + L^o A_{1222} \tag{76}$$

where $L^o \in \mathbb{R}^{(n-p-r)\times(p-m)}$ is any gain matrix so that $A^o_{22} + L^o A^o_{21}$ is stable. Let $\mathcal{K}$ be any state feedback matrix for the controllable pair $(\tilde{A}_{11}, A_{122})$ so that $\tilde{A}_{11} - A_{122}\mathcal{K}$ is stable and partition the state feedback matrix so that

$$\begin{bmatrix} \overset{n-r-p}{\longleftrightarrow} & \overset{p-m}{\longleftrightarrow} \\ \mathcal{K}_1 & \mathcal{K}_2 \end{bmatrix} = \mathcal{K} \tag{77}$$

and define

$$K = \mathcal{K}_2 - \mathcal{K}_1 L^o \tag{78}$$
$$K_c = \mathcal{K}_1 \tag{79}$$

If $r > 0$ define a new dynamical system by

$$\dot{z}_r(t) = A^o_{11} z_r(t) + A^o_{12} x_c(t) + (A^m_{121} - A^o_{12}L^o)x_{12}(t) + A_{121}x_2(t) \tag{80}$$

[2] The phrase '*stabilisable with respect to output feedback*' will indicate that for the linear system (A, B, C) there exists a fixed gain K such that the matrix $A - BKC$ is stable.

and augment (80) with (70) to form a new compensator

$$\dot{\hat{x}}_c(t) = \hat{H}\hat{x}_c(t) + \hat{D}y(t) \tag{81}$$

where

$$\hat{H} = \begin{bmatrix} A^o_{11} & A^o_{12} \\ 0 & H \end{bmatrix} \quad \text{and} \quad \hat{D} = \begin{bmatrix} (A^m_{121} - A^o_{12}L^o) & A_{121} \\ D_1 & D_2 \end{bmatrix} T^{\mathrm{T}}$$

If

$$\hat{x} = \begin{bmatrix} \hat{x}_c \\ T^{\mathrm{T}}y \end{bmatrix} \tag{82}$$

then the sliding surface $\mathcal{S}_c$ can be written as $\{\hat{x} \in \mathbb{R}^n \ : \ S\hat{x} = 0\}$ where

$$S = F_2 \begin{bmatrix} 0_{m\times r} & K_c & K & I_m \end{bmatrix} \tag{83}$$

A control law to induce a sliding motion on the sliding surface $\mathcal{S}_c$ is given by

$$u(t) = L\hat{x}(t) + u_n(t) \tag{84}$$

where L is defined in §5.6 of [1] and $u_n(t)$ is the scaled unit-vector law in (32).

Example

The following session calculates a compensator based output dependant sliding mode controller. The canonical form from Session 8 underpins the synthesis and the specified poles obtained from Session 3 are used to specify part of the sliding mode dynamics.

Session 9: comrobs

```
>> [Hhat,Dhat,S,L,P]=comrobs(A,B,C)

Enter 1 pole for the error dynamics -10

Enter 3 pole for sliding motion [-4.3241+1.7852*j -4.3241-1.7852*j -3.1623]

Enter 1 pole for the range space dynamics -6

place: ndigits= 15
place: ndigits= 17

Hhat =
   -10

Dhat =
         0  -67.6603   31.4960

S =
    0.8269   -2.2405   12.6843    1.0000

L =
   29.2263  -41.9425  458.6392   48.8259
```

```
P =
    0.0833
>>
```

This example is consider in greater detail in §5.6.3 in [1].

4 Sliding Mode Observers

Despite fruitful research and development activity in the area of variable structure control theory, relatively few authors have considered the application of the underlying principles to the problem of observer design. Consider the nominal linear system subject to a class of uncertainty described by

$$\dot{x}(t) = Ax(t) + Bu(t) + D\xi(t, y, u) \tag{85}$$
$$y(t) = Cx(t) \tag{86}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{n\times q}$ with $q \leq p < n$ and the matrices B, C and D are of full rank. The function $\xi(t, y, u)$ is assumed to be unknown but bounded by a known function. It is further assumed that the states of the system are unknown and only the signals $u(t)$ and $y(t)$ are available.

The objective is to synthesise an observer to generate a state estimate $\hat{x}(t)$ such that the error

$$e(t) = \hat{x}(t) - x(t) \tag{87}$$

tends to zero despite the presence of the uncertainty. Furthermore the intention is to induce a sliding motion on the surface in the error space

$$\mathcal{S}_o = \{e \in \mathbb{R}^n : Ce = 0\} \tag{88}$$

The particular observer structure that will be considered can be written in the form

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) - G_l Ce(t) + G_n \nu \tag{89}$$

where $G_l, G_n \in \mathbb{R}^{n\times p}$ are appropriate gain matrices and ν represents a discontinuous switched component to induce a sliding motion on $\mathcal{S}_o$.

Consider the dynamical system given in (85) and (86) and assume that

- $\text{rank}(CD) = q$
- the invariant zeros of (A, D, C) lie in the $\mathbb{C}_-$.

It can be shown (§6.3.1 in [1]) that under these assumptions there exists a linear change of coordinates $x \mapsto Tx$ such that in the new coordinate system

$$\dot{x}_1(t) = \mathcal{A}_{11}x_1(t) + \mathcal{A}_{12}x_2(t) + B_1u(t) \tag{90}$$
$$\dot{x}_2(t) = \mathcal{A}_{21}x_1(t) + \mathcal{A}_{22}x_2(t) + B_2u(t) + D_2\xi(t, y, u) \tag{91}$$
$$y(t) = x_2(t) \tag{92}$$

where $x_1 \in \mathbb{R}^{(n-p)}$, $x_2 \in \mathbb{R}^p$ and the matrix $\mathcal{A}_{11}$ has *stable* eigenvalues. The freedom of choice associated with A_{11} depends on the number of invariant zeros of (A, B, C). The coordinate system above will be used as a platform for the design of a sliding mode observer. Consider a dynamical system of the form

$$\dot{\hat{x}}_1(t) = \mathcal{A}_{11}\hat{x}_1(t) + \mathcal{A}_{12}\hat{x}_2(t) + B_1 u(t) - \mathcal{A}_{12} e_y(t) \quad (93)$$

$$\dot{\hat{x}}_2(t) = \mathcal{A}_{21}\hat{x}_1(t) + \mathcal{A}_{22}\hat{x}_2(t) + B_2 u(t) - (\mathcal{A}_{22} - \mathcal{A}^s_{22}) e_y(t) + \|D_2\|\nu \quad (94)$$

$$\hat{y}(t) = \hat{x}_2(t) \quad (95)$$

where $\mathcal{A}^s_{22}$ is a stable design matrix and $e_y = \hat{y} - y$. The discontinuous vector ν is defined by

$$\nu = \begin{cases} -\rho(t,y,u)\frac{P_2 e_y}{\|P_2 e_y\|} & \text{if } e_y \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (96)$$

where $P_2 \in \mathbb{R}^{p \times p}$ is a Lyapunov matrix for $\mathcal{A}^s_{22}$ satisfying

$$P_2 A^s_{22} + (A^s_{22})^{\mathrm{T}} P_2 = -I \quad (97)$$

and the scalar function $\rho(t, y, u)$ is chosen so that

$$\|\xi(t, y, u)\| < \rho(t, y, u) \quad (98)$$

If the state estimation errors are defined as $e_1 = \hat{x}_1 - x_1$ and $e_2 = \hat{x}_2 - x_2$ then it is straightforward to show

$$\dot{e}_1(t) = \mathcal{A}_{11} e_1(t) \quad (99)$$

$$\dot{e}_y(t) = \mathcal{A}_{21} e_1(t) + \mathcal{A}^s_{22} e_y(t) + \|D_2\|\nu - D_2\xi(t, y, u) \quad (100)$$

Furthermore the nonlinear error system in (99) - (100) is quadratically stable and a sliding motion takes place on the surface $\mathcal{S}_o$ defined in (88). The dynamical system in (93) - (95) may thus be regarded as an observer for the system in (85)–(86). It follows that if the linear gain

$$G_l = T^{-1} \begin{bmatrix} \mathcal{A}_{12} \\ \mathcal{A}_{22} - \mathcal{A}^s_{22} \end{bmatrix} \quad (101)$$

and the nonlinear gain

$$G_n = \|D_2\| T^{-1} \begin{bmatrix} 0 \\ I_p \end{bmatrix} \quad (102)$$

then the observer given in (93) - (95) can be written in terms of the original coordinates in the form of (89).

Example

The following session places the aircraft system

$$A = \begin{bmatrix} 0 & 0 & 1.0000 & 0 & 0 \\ 0 & -0.1540 & -0.0042 & 1.5400 & 0 \\ 0 & 0.2490 & -1.0000 & -5.2000 & 0 \\ 0.0386 & -0.9960 & -0.0003 & -0.1170 & 0 \\ 0 & 0.5000 & 0 & 0 & -0.5000 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ -0.7440 & -0.0320 \\ 0.3370 & -1.1200 \\ 0.0200 & 0 \\ 0 & 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

from §7.5.1 in [1] in the canonical form for observer design under the assumption that the matrix which distributes the uncertainty is $D = B$.

Session 10: obsfor

```
>> [Ac,Bc,Cc,Tc]=obsfor(A,B,C)

Enter 1 desired stable sliding mode pole(s) -3
place: ndigits= 15

Ac =
    -3.0000    3.2541    0.0082  -12.4182   -0.1637
     0.1089   -0.6417   -0.0046    2.0019   -0.0000
    -0.1761    0.2292   -0.9994   -5.9468         0
     0.7043   -0.9167   -0.0026    2.8702    0.0386
          0         0    1.0000    0.0000         0

Bc =
    -0.0000    0.0000
    -0.7440   -0.0320
     0.3370   -1.1200
     0.0200   -0.0000
          0         0

Cc =
          0    1.0000    0.0000         0         0
          0         0    1.0000    0.0000         0
          0         0   -0.0000    1.0000         0
          0         0         0         0    1.0000

Tc =
          0   -0.1126    0.0032   -4.2415   -1.3017
          0    1.0000    0.0000         0   -1.0000
          0         0    1.0000         0         0
          0   -0.0000    0.0000    1.0000    0.0000
     1.0000         0         0         0         0
>>
```

Notice in this example because no invariant zeros are present the dynamics associated with the upper left element (which govern the sliding motion) can be placed arbitrarily.

The next session uses the same example and computes the observer gains from equations (101) and (102).

Session 11: disobs

```
>> [Gl,Gn,P]=disobs(A,B,C)

Enter 1 desired stable sliding mode pole(s) -3
place: ndigits= 15

Enter 4 output estimation error pole(s) [-4 -4.25 -4.5 -5]

Gl =
         0    1.0000    0.0000    5.0000
    3.5399    0.0051  -11.4944    0.0000
    0.2292    3.2506   -5.9468         0
   -0.9167   -0.0026    7.3702    0.0386
    0.1816    0.0097  -13.4963    0.0000


Gn =
         0         0         0    1.1922
    1.0973    0.0027   -3.5756         0
         0    1.1922         0         0
    0.0000   -0.0000    1.1922         0
   -0.0949    0.0027   -3.5756         0


P =
    0.1250         0         0         0
         0    0.1176         0         0
         0         0    0.1111         0
         0         0         0    0.1000

>> eig(A-Gl*C)

ans =
   -5.0000
   -3.0000
   -4.5000
   -4.2500
   -4.0000

>>
```

Another observer, introduced by Walcott & Żak [6], considers the special case when the uncertainty is matched i.e. when $D = B$. They propose the observer structure given by

$$\dot{z}(t) = Az(t) + Bu(t) - GCe(t) + B\nu_o \tag{103}$$

where z represents an estimate of the true states x, and $e = z - x$ is the state estimation error. The output error feedback gain matrix G is chosen so that the closed loop matrix $A_0 = A - GC$ is stable and has a Lyapunov matrix P satisfying

both

$$PA_0 + A_0^{\mathrm{T}} P = -Q \tag{104}$$

for some positive definite design matrix Q and the structural constraint

$$PB = C^{\mathrm{T}} F^{\mathrm{T}} \tag{105}$$

for some nonsingular matrix $F \in \mathbb{R}^{m \times p}$. The discontinuous vector ν_o is given by

$$\nu_o = \begin{cases} -\rho_o(t,y,u)\frac{FCe}{\|FCe\|} & \text{if } Ce \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{106}$$

where $\rho_o(t,y,u)$ is a scalar function which bounds the uncertainty. The observer given in (103) may be viewed as a Luenberger observer with an additional nonlinear term. It can be shown (§6.4 in [1]) that assumptions A1 and A2 relating to the triple (A,B,C) are necessary and sufficient conditions for the existence of such an observer which is insensitive to matched uncertainty and induces a sliding motion on

$$\mathcal{S}_w = \{\ e \in \mathbb{R}^n \quad : \quad FCe = 0\ \} \tag{107}$$

The original formulation of Walcott & Żak required the use of symbolic manipulation to synthesise the matrices G and P which completely define the observer. More recently an analytic solution has been proposed based on the canonical form described in (90)–(92). An appropriate choice of G is that given in (101) and an appropriate choice of

$$F = (P_2 B_2)^{\mathrm{T}} \tag{108}$$

This session considers the same example as Session 11 but this time computes a Walcott–Żak observer:

Session 12: wzobs

```
>> [G,F]=wzobs(A,B,C)

Enter 1 desired stable sliding mode pole(s) -3
place: ndigits= 15

Enter 4 desired output estimation error pole(s) [-4 -4.425 -4.5 -5]

G =
         0    1.0000    0.0000    5.0000
    3.5399    0.0055  -11.4944    0.0000
    0.2292    3.4256   -5.9468         0
   -0.9167   -0.0026    7.3702    0.0386
    0.1816    0.0101  -13.4963    0.0000


F =
   -0.0930    0.0381    0.0022         0
   -0.0040   -0.1266   -0.0000         0

>>
```

In the special case of a square system an even more explicit solution can be obtained. This does not require the attainment of the canonical form (90) - (92) explicitly and in certain circumstances produces an observer with better numerical properties.

This final session consider the furnace system from (56) and computes a Walcott–Żak observer as in §9.1 in [1]. The observer is computed in two ways, first using the square system approach and secondly using the generic method. The numerical properties of the two resulting error systems are compared.

Session 13: sqobs

```
>> [G,F]=sqobs(A,B,C)

Enter 2 real stable poles for the output error system [-0.5 -0.5]

G =
    1.7201   -0.3132
    0.6269   -0.2251
    8.1321   -2.7820
    0.4512    0.4562

F =
    0.0043    0.0030
    0.0031    0.0212

>> [V,D]=eig(A-G*C);
>> cond(V)

ans =

   22.2895

>> [G,F]=wzobs(A,B,C)

The system has 2 stable invariant zero(s)

Enter 2 desired output estimation error pole(s) [-0.5 -0.5]

G =
    1.7201   -0.3132
    0.6269   -0.2251
    8.1321   -2.7820
    0.4512    0.4562


F =
    0.0043    0.0030
    0.0031    0.0212

>> [V,D]=eig(A-G*C);
>> cond(V)
```

```
ans =

   62.6291

>>
```

It can be seen that the square system approach has in this case furnished eigenvectors with better conditioning.

5 Summary

This document has described a collection of MATLAB mfiles which have been developed to design sliding mode controllers for linear systems with bounded nonlinear uncertainty. The algorithms are described in detail elsewhere and indeed this document is best viewed as an addendum to [1]. Here the emphasis is on describing the software and providing examples of its use.

References

[1] C. Edwards and S.K. Spurgeon. *Sliding Mode Control: Theory and Applications.* Taylor & Francis, 1998.

[2] U. Itkis. *Control Systems of Variable Structure.* Wiley, New York, 1976.

[3] E.P. Ryan and M. Corless. Ultimate boundedness and asymptotic stability of a class of uncertain dynamical systems via continuous and discontinuous control. *IMA Journal of Mathematical Control and Information*, 1:223–242, 1984.

[4] V.I. Utkin. Variable structure systems with sliding modes. *IEEE Transactions on Automatic Control*, 22:212–222, 1977.

[5] V.I. Utkin. *Sliding Modes in Control Optimization.* Springer-Verlag, Berlin, 1992.

[6] B.L. Walcott and S.H. Żak. Combined observer–controller synthesis for uncertain dynamical systems with applications. *IEEE Transactions on Systems, Man and Cybernetics*, 18:88–104, 1988.

6 Appendix: Toolbox Commands

comrobs

Purpose

To generate a compensator based output sliding mode controller

Syntax

$[Hhat, Dhat, S, L, P, Lam, T]$=comrobs$(A, B, C)$

Description

Generates a compensator and the associated components of the sliding mode control law for the triple (A, B, C). In order for this to be possible conditions A1 and A2 need to be satisfied. These are checked during the execution. If n, m, p and r are the number of states, inputs, outputs and invariant zeros respectively, then, provided the approach is applicable, the user is prompted to supply three vectors:

- an $n - p - r$ dimensional vector which represents the poles of the estimation error system. (In the notation of §3, the specified poles are assigned to the matrix H from (74) via pole placement.)
- an $n - m - r$ dimensional vector which constitute the poles of the sliding motion. (In the notation of §3, the gain matrix $\mathcal{K}$ is computed so that the eigenvalues of $(\tilde{A}_{11} - A_{112}\mathcal{K})$ are the specified values.)
- an m dimensional vector which will be used to assign the poles which govern the range space dynamics. (In the notation of §3, the matrix Φ will be built from the specified poles.)

The matrices $Hhat$, $Dhat$ which are returned are the compensators' state and input distribution matrices respectively; S is the matrix which defines the switching function, L is the linear feedback component in the control law; P is a Lyapunov matrix which scales the switching function and Lam represents the gain which pre-multiplies the unit vector component in the nonlinear control. The orthogonal matrix T is obtained from the canonical form for output feedback design from §3 and is used to scale the outputs.

Algorithm

First the canonical form for output feedback design is obtained and the components that constitute the triple $(\tilde{A}_{11}, A_{122}, \tilde{C}_1)$ are identified. A reduced order linear observer structure is then obtained in conjunction with a state feedback gain for the pair $(\tilde{A}_{11}, A_{122})$. This effectively parameterises both the compensator matrices and the switching function. The approach is described in detail in §5.4 in [1].

contl

Purpose

This assembles the components of the state feedback unit vector control law

Syntax

$[L, P, Lam]$=contl(A, B, S, Phi)

Description

The pair (A, B) constitutes the state and input distribution matrices of the system and S represents a previously designed matrix which defines the switching function. If the system has m inputs, then Phi is a stable $m \times m$ matrix which will govern the linear part of the range space dynamics. The function returns a matrix L which constitutes the linear state feedback component of the control law, the matrix Lyapunov matrix P which scales the hyperplane and Lam which is the gain which pre-multiplies the unit vector component in the nonlinear control law.

Algorithm

The control law is the unit vector control structure of Ryan & Corless [3]. Details of the components are given in (31)–(32). The justification of the formula appears in §3.6.3 in [1].

contlia

Purpose

This assembles the components of the state feedback unit vector control law in the case when integral action is employed for tracking purposes

Syntax

$[L, Lr, Lrdot, Sr, Lam, P]$=contlia$(A, B, C, S, Phi, Sr)$
$[L, Lr, Lrdot, Sr, Lam, P]$=contlia$(A, B, C, S, Phi)$

Description

The triple (A, B, C) constitutes the state, input and output distribution matrices of the system. The matrix S represents a previously designed matrix which defines the augmented state component of the switching function and the parameter S_r represents the reference dependant component of the switching function. If the system has m inputs then Phi is a stable $m \times m$ matrix which will govern the linear part of the range space dynamics. The function returns the controller parameters in the case where integral action is employed to provide tracking and all the states are available for use in the control law. L represents the linear augmented state feedback gain and Lr and $Ldot$ represent feed-forward gains for the reference and derivative of the reference respectively. The symmetric positive definite matrix P is the Lyapunov matrix which scales the hyperplane and Lam is the gain associated with the unit vector component.
If only five right hand arguments are supplied the reference dependant component of the switching function S_r is calculated so that, in the nominal case, the steady state value of the integral action states tend to zero.

Algorithm

The control law is the unit vector control structure described in (48)–(54). Details of the construction, particularly with regard to the choice of S_r, is given in §7.3.4 in [1].

See also

dea, **lqcf**, **rpp**

dea

Purpose

Hyperplane design using so-called direct eigenstructure assignment – i.e. specifying both the sliding mode poles and elements of the associated eigenvectors

Syntax

S=dea($A, B, specent, lambda$)
$[S, V]$=dea($A, B, specent, lambda$)
S=dea($A, B, specent, lambda, nocomp$)
$[S, V]$=dea($A, B, specent, lambda, nocomp$)

Description

The pair (A, B) constitutes the state and input distribution matrices of the system. When four right hand arguments are supplied, the vector *lambda* represents the $n-m$ specified sliding mode eigenvalues which must be real (n and m represent the number of states and inputs respectively). The $n \times (n-m)$ matrix *specent* indicates the position and value of the desired eigenvector entries. Elements of the eigenvector which can take arbitrary values are indicated by NaN's. Thus the first column of *specent* is the desired eigenvector associated with the first element in *lambda* etc. The function returns the matrix S which defines the switching function. If two left hand arguments are provided an $n \times (n-m)$ matrix V is returned which represents the attained eigenvectors. If complex eigenvalues are desired, five right hand arguments must be supplied and the scalar *nocomp* defines the number of pairs of complex conjugate poles. Again the vector *lambda* defines the desired sliding mode poles; the first $2i-1$ $(i = 1, nocomp)$ entries contain the real parts of the complex conjugate poles and the first $2i$ $(i = 1, nocomp)$ contain the imaginary parts. Likewise the first $2i-1$ $(i = 1, nocomp)$ columns of *specent* are associated with the real parts of the complex eigenvectors and the first $2i$ $(i = 1, nocomp)$ are associated with the imaginary parts.

Algorithm

The algorithm is that described in Appendix B.2 in [1]. For implementation purposes, the variable *specpos* in Appendix B.2 is obtained from *specent* by interrogating the user about the specified elements using the MATLAB command **isnan**.

See also

eigbuil

disobs

Purpose

Calculates the gains of a sliding mode observer which is insensitive to a particular class of uncertainty

Syntax

$[Gl, Gn, P2]$=disobs(A, D, C)

Description

The pair (A, C) represents the state and output distribution matrix of the system and D represents a distribution matrix which describes the channels in which the uncertainty occurs. It is assumed that if n and p represent the number of states and outputs respectively, then D is $n \times q$ where $q \leq p$. The function returns the linear gain Gl and non-linear gain Gn along with the Lyapunov matrix $P2$ which constitutes a discontinuous observer. For the theory to be valid both C and D should be full rank, the Markov parameter CD must be full rank and any invariant zeros of triple (A, D, C) must lie in the $\mathbb{C}_-$. During the computations a prompt is given for a vector of dimension $n - p - r$ where r the number of invariant zeros of (A, D, C). This vector specifies the desired stable poles of the sliding motion. In addition the user is prompted for p stable real poles which form part of the state estimation error dynamics.

Algorithm

The function puts the given system in the canonical form for observer design described in (90)–(92). The gains are constructed from equations (101) and (102) and from solving the Lyapunov equation (97). For details see §6.3 in [1].

See also

sqobs, **wzobs**

eigbuil

Purpose

Forms the matrix which determines the specified components within each eigenvector for use in the design of a hyperplane by direct eigenstructure assignment

Syntax

specent=eigbuil(n, m)
specent=eigbuil($n, m, nocomp$)

Description

The scalar n is the size of the state space vector and m is the number of inputs. If three right hand arguments are presented then *nocomp* is the number of complex pairs of eigenvalues that make up the desired poles for the sliding motion. The function returns a matrix of size $n \times (n - m)$ whose columns indicate the position and value of the specified elements in the associated eigenvector. The user is prompted for the position and values of the specified entries eigenvector by eigenvector. The remaining entries in the appropriate columns of *specent* are filled automatically by NaN's.

See also

dea

intac

Purpose

Calculates the augmented state and input distribution matrices for designing a control law incorporating integral action

Syntax

$[Aa, Ba]$=intac(A, B, C)

Description

For the system represented by the triple (A, B, C) the augmented state and input distribution matrices Aa and Ba as in (42). If this approach is not applicable because of the presence of any invariant zeros of (A, B, C) at the origin, empty matrices are returned and an appropriate error message is displayed.

See also

contlia

lqcf

Purpose

Hyperplane design using a quadratic cost function

Syntax

S=lqcf(A, B, Q)
$[S, E]$=lqcf(A, B, Q)

Description

For a given system represented by the state and input distribution matrices (A, B) the function returns a matrix S which defines the switching function by minimising the cost function

$$J = \int_{t_s}^{\infty} x^{\mathrm{T}} Q x \, dt$$

where Q is a symmetric positive definite matrix of the same order as the state space and t_s is the time at which the sliding motion begins. In the coordinates of the regular form, and thinking of the hyperplane design problem as a state feedback problem for a certain subsystem, the quadratic minimisation problem becomes a conventional well-posed LQR problem which can be solved using the standard control system toolbox routines.

Algorithm

Regular form is first established and a further series of transformations establishes the cost function (23) and the reduced order system matrix (25). In this form the problem becomes a standard Linear Quadratic Regulator problem which is solved using the routines from the Control System Toolbox. Details are given in §4.4.2 in [1].

obsfor

Purpose

Performs the necessary state transformations to put a system triple into the canonical form for observer design

Syntax

$[Ac, Dc, Cc, Tc]$=obsfor(A, D, C)

Description

Here the matrices A and C represent the state and output distribution matrices respectively and D is a matrix which describes the channels in which the uncertainty acts. The function returns the nominal triple (A, D, C) in the canonical form used in the construction of the observer gains given in (90)–(92). In order for this to be possible CD must be full rank and any invariant zeros of the triple (A, D, C) must be in the $\mathbb{C}_-$. Checks are made during the execution of the function to verify these conditions. If the number of columns of D is less than the number of rows of C then some flexibility exists in choosing the eigenvalues of the sliding motion. In this case the user is prompted to supply an appropriate number of stable poles. The state transformation matrix to induce this form is given in Tc.

Algorithm

The canonical form for output feedback sliding mode control is first established. A further transformation using all the available degrees of freedom is made to achieve the canonical form in (90)–(92). For further details see §6.3.1 in [1].

outfor

Purpose

Performs the necessary state transformation to put a system triple into the canonical form for output feedback design

Syntax

$[Af, Bf, Cf, Tcan, r]$=outfor(A, B, C)

Description

For the system represented by the triple (A, B, C) the function computes, where possible, the canonical form for output feedback design. This requires that assumptions A1 and A2 are valid. If these are found not to hold then the function returns empty matrices. The variable r represents the number of invariant zeros of the system (A, B, C). The matrix $Tcan$ represents the coordinate transformation necessary to bring about the canonical form.

Algorithm

Details of the state-space transformations are given in §5.4 in [1].

See also

comrob

regfor

Purpose

Transforms a given state and input distribution matrix into regular form

Syntax

$[A11, A12, B2, Tr]$=regfor(A, B)

Description

For the system with state and input distribution matrices A and B the function returns the sub-matrices and the orthogonal transformation matrix that brings about the regular form described in (10)–(11).

Algorithm

A QR factorisation is first obtained for the input distribution matrix - the required transformation is then obtained by reordering the rows. The approach is described in more detail in §4.2 in [1].

rpp

Purpose

Hyperplane design using pole placement with robust eigenstructure assignment

Syntax

$S = \text{rpp}(A, B, sp)$

Description

For the system represented by the state and input distribution matrices A and B the function calculates the matrix S which defines the switching function using robust eigenstructure assignment. If n and m represent the number of states and inputs respectively then sp is a vector of order $n - m$ containing the desired eigenvalues. Any complex eigenvalues in the vector sp must appear in consecutive complex conjugate pairs and the eigenvalues cannot be placed with multiplicity greater than the number of inputs.

Algorithm

Regular form is first achieved then the modified pole placement command **vplace** is utilised (which relies on the Control System Toolbox routine **place**). For details see Appendix B.1 in [1].

See also

dea

sqobs

Purpose

Calculates the gain matrices for a Walcott–Żak observer in the special case of a square system

Syntax

$[G, F]$=sqobs(A, B, C)

Description

For the system represented by the triple (A, B, C) this function produces a Walcott–Żak observer for the special case of square systems i.e. when the number of inputs equals the number of outputs. Necessary and sufficient conditions for the observer to exist are that assumptions A1 and A2 are satisfied. The arguments returned are the linear gain matrix G along with the matrix F which scales the switching function. During the execution the user is prompted for a vector of order p containing stable real poles (where p is the number of outputs). These are assigned to form a subset of the poles of the state estimation error system.

Algorithm

The design is made from regular form rather than from the observer canonical form. The explicit solution is described in §7.3.1 in [1].

See also

wzobs, **disobs**

vplace

Purpose

Performs pole placement with robust eigenstructure assignment

Syntax

K=vplace(A, B, sp)

Description

For the system represented by the state and input distribution matrix A and B this function returns a state feedback matrix K which assigns to the closed loop the poles contained in the vector sp. This command is a modification of the control system tool box command pole placement routine **place** and performs an initial transformation to remove any redundant columns of the input distribution matrix.

Algorithm

An initial QR factorisation of the input distribution matrix is made and any redundant columns are removed before invoking the **place** command from the Control System Toolbox (see Appendix B.1 in [1] for details).

wzobs

Purpose

Computes the gains for a Walcott–Żak observer

Syntax

$[G, F]$=wzobs(A, B, C)

Description

For the system represented by the triple (A, B, C), this function calculates the gains for a Walcott–Żak observer. Necessary and sufficient conditions for the observer to exist are that assumptions A1 and A2 are satisfied. The arguments returned are the linear gain matrix G along with the matrix F which scales the switching function. During the computations a prompt is given for a vector of dimension $n-p-r$ where n the number of states, p is the number of outputs and r the number of invariant zeros of (A, B, C). This vector allows the user to specify the assignable poles in the top left sub-block of the observer canonical form. The user is prompted for a vector of order p containing stable real poles. These poles are assigned to form a subset of the poles of the state estimation error system.

Algorithm

The function puts the given system in the canonical form for observer design described in (90)–(92). The gains are constructed from equations (101) and (108). For details see §6.4 in [1].

See also

sqobs

comrobs	generates a compensator based output sliding mode controller
contl	assembles the components of the state feedback unit vector control law
contlia	assembles the components of the state feedback unit vector control law in the case when integral action is employed for tracking purposes
dea	hyperplane design using direct eigenstructure assignment
disobs	calculates the gains of a sliding mode observer which is insensitive to a particular class of uncertainty
eigbuil	form the matrix which determines the specified components within each eigenvector for use in the design of a hyperplane by direct eigenstructure assignment
intac	calculates the augmented state and input distribution matrices for designing a control law incorporating integral action
lqcf	hyperplane design using a quadratic cost function
output	performs the necessary state transformations to put a system triple into the canonical form for observer design
obsfor	performs the necessary state transformation to put a system triple into the canonical form for output feedback design
regfor	transforms a given state and input distribution matrix into regular form
rpp	hyperplane design using pole placement with robust eigenstructure assignment
sqobs	calculate the gain matrices for a Walcott–Zak observer in the special case of a square system
vplace	pole placement with robust eigenstructure assignment
wzobs	computes the gains for a Walcott–Zak observer

Table 1: Summary of MATLAB commands