The Matrix exponential, Dynamic Systems and Control

Niels Kjølstad Poulsen

Informatics and Mathematical Modelling, Building 321
Technical University of Denmark
DK-2800 Lyngby

2004-02-18 15.06

Abstract

The matrix exponential can be found in various connections in analysis and control of **dynamic systems**. In this short note we are going to list a few examples. The matrix exponential usably pops up in connection to the sampling process, whatever it is in a deterministic or a stochastic setting or it is a tool for determining a Gramian matrix.

This note is intended to be used in connection to the teaching post the course in **Stochastic Adaptive Control (02421)** given at Informatics and Mathematical Modelling **(IMM)**, The Technical University of Denmark. This work is a result of a study of the litterature.

1 Introduction

One way to give a formal definition on the matrix exponential is through its Taylor expansion

$$e^A = I + A + \frac{1}{2}A^2 + \ ... \ + \frac{1}{n!}A^n + \ ...$$

The numerical evaluation of the matrix exponential can in some situations be done by applying the definition and the Taylor expansion. Depending on the properties of A different numerical alternatives might be used.

The following Lemma can be found in e.g. [1] (page 235). Consider matrices A_{11}, A_{12} and A_{22} with adequate dimensions. Let

$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} h \right) \tag{1}$$

Then

$$F_{11} = e^{A_{11}h} \qquad F_{22} = e^{A_{22}h}$$

and

$$F_{12} = \int_0^h e^{A_{11}(h-s)} A_{12} e^{A_{22}s} \ ds$$

Since the matrices are block upper triangular, we easily get

$$F_{11} = e^{A_{11}h} \quad \text{and} \quad F_{22} = e^{A_{22}h}$$

If we differentiate (1) we get

$$\frac{d}{dt}\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix}$$

and

$$\frac{d}{dt}F_{12} = A_{11}F_{12} + A_{12}F_{22}$$

Using the solution for F_{22} and $F_{12}(0) = 0$ we have

$$F_{12} = \int_0^h e^{A_{11}(h-s)}A_{12}e^{A_{22}s}ds$$

This lemma has several applications in connection to system theory, as we will illustrate in the next sections.

2 Sampling of a deterministic system

Let a deterministic (LTI) system in continuous time be given by the state space description

$$\frac{d}{dt}x = Ax + Bu \qquad x(0) = x_0$$

It is well known that the solutions to this description is given by

$$x(t) = e^{At}x_0 + \int_0^t e^{A(t-s)}Bu(s)\ ds$$

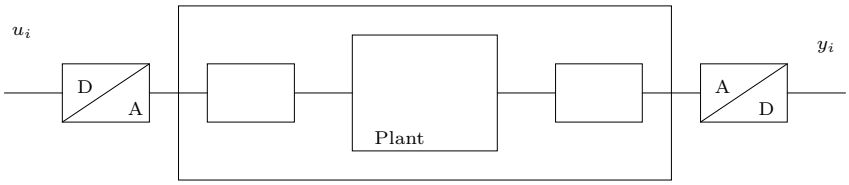

Figure 1. Sampled-data control system with the plant embedded with sensors and actuators among filters and converters.

From a computer point of view (and using a zero order hold sample and hold network) the system (or the plant) can in discrete be described by a discrete time model

$$x_{i+1} = \Phi x_i + \Gamma u_i$$

$$y_i = Cx_i$$

where

$$\Phi = e^{Ah} \qquad \Gamma = \int_0^h e^{A(h-s)}B\ ds = \int_0^h e^{A\tau}B\ d\tau$$

In the latter substitution we have used $\tau = h - s$. Notice the control action is assumed to be constant between samples (when we use a zero order hold network).

Also

$$\Gamma = A^{-1}(\Phi - I)B = (\Phi - I)A^{-1}B$$

Let

$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} h\right)$$

and using the Lemma we have

$$F_{11} = e^A h \qquad F_{12} = \int_0^h e^{A(h-s)} B I ds$$

and

$$\Phi = F_{11} \qquad \Gamma = F_{12}$$

The Matlab implementation of this algorithm is listed in Appendix C as c2d.m.

3 Sampling of a stochastic system

Consider a (LTI) continuous time stochastic system

$$\frac{d}{dt}x = Ax + w$$

where the intensity of w is R. In discrete time, i.e. for $t = ih$ where h is the sampling period, the system can be described by

$$x_{i+1} = \Phi x_i + v_i$$

where the variance of v_i is Σ which (see e.g. [3] p. 84) is given by

$$\Sigma = \int_0^h e^{A_c s} R_c e^{A_c^T s}\, ds$$

or the solution to

$$\frac{d}{dt}\Sigma = A\Sigma + \Sigma A^T + R \qquad \Sigma(0) = 0$$

One method can be found in [3] p. 111. Let

$$\begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} A & R \\ 0 & -A^T \end{bmatrix} h\right)$$

then

$$F_{11} = e^A h = \Phi \qquad F_{22} = e^{-A^T h} = \Phi^{-1}$$

and

$$F_{12} = \int_0^h e^{A(h-s)} R_s e^{-A^T s} ds = \int_0^h e^{A(h-s)} R_s e^{A^T(h-s)} ds\; e^{-A^T h}$$

If we apply the substitution $\tau = h - s$ we get

$$\Phi = F_{11} \qquad \Sigma = F_{12} F_{22}^{-1}$$

Another method is the following. Let

$$\begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} -A & R \\ 0 & A^T \end{bmatrix} h\right)$$

then

$$F_{11} = e^{-Ah} = \Phi^{-1} \qquad F_{22} = e^{A^T h} = \Phi^T$$

and

$$F_{12} = \int_0^h e^{-A(h-s)} R e^{A^T s}\, ds = e^{-Ah} \int_0^h e^{As} R e^{A^T s}\, ds$$

and the

$$\Phi = F_{22}^T \qquad \Sigma = F_{22}^T F_{12}$$

The Matlab implementation of this algorithm is listed in Appendix C as nc2d.m.

4 Controlability

If we address the question whatever it is possible to drive the system

$$\frac{d}{dt}x = Ax + Bu \qquad x(0) = x_0$$

from any initial state to any target state in finite time, we might answer that (See e.g. [2] p. 610 or [1] p. 236) by checking the rank properties of the controlability Gramian

$$W_c = \int_0^h e^{As} B B^T e^{A^T s}\, ds$$

The controlability Gramian can also be found as the solution to the following differential equation

$$\frac{d}{dt}W_c = AW_c + W_c A^T + BB^T \qquad W_c(0) = 0$$

Define

$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} -A & BB^T \\ 0 & A^T \end{bmatrix} h \right)$$

then

$$F_{11} = e^{-Ah} = \Phi^{-1} \qquad F_{22} = e^{A^T h} = \Phi^T$$

and

$$F_{12} = \int_0^h e^{-A(h-s)} B B^T e^{A^T s}\, ds = e^{-Ah} \int_0^h e^{As} B B^T e^{A^T s}\, ds$$

and

$$\Phi = F_{22}^T \qquad W_c = F_{22}^T F_{12}$$

The Matlab implementation of this algorithm is listed in Appendix C as syswc.m.

5 Observability

The dual to the controlability problem is the oberservability problem. If we observe the output from the system

$$\frac{d}{dt}x = Ax$$

over a finite period of time, then the question is whatever we can determine any initial state value. This problem is solved (See e.g. [2] p. 615.) by checking the rank properties of the obervervability Gramian

$$W_o = \int_0^h e^{A^T s} C^T C e^{As}\ ds$$

$$\frac{d}{dt} W_o = A^T W_o + W_o A + C^T C \qquad W_o(0) = 0$$

Define

$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} -A^T & C^T C \\ 0 & A \end{bmatrix} h \right)$$

then

$$F_{11} = e^{-A^T h} = \Phi^{-T} \qquad\qquad F_{22} = e^{Ah} = \Phi$$

and

$$F_{12} = \int_0^h e^{-A^T(h-s)} C^T C e^{As}\ ds = e^{-A^T h} \int_0^h e^{A^T s} C^T C e^{As}\ ds$$

and

$$\Phi = F_{22} \qquad\qquad W_c = F_{22}^T F_{12}$$

The Matlab implementation of this algorithm is listed in Appendix C as syswo.m.

6 Sampled-data control

Consider the problem of controlling a continuous time LTI system

$$\frac{d}{dt} x(t) = Ax(t) + Bu(t) \qquad x(0) = \underline{x}(0) \tag{2}$$

such that the (standard continuous time) objective function

$$J = \frac{1}{2} x^T(T) P x(T) + \frac{1}{2T} \int_0^T x^T(t) Q x(t) + u^T(t) R u(t)\ dt$$

is minimized. The control actions are assumed to be constant between samples, i.e.

$$u(t) = u_i \qquad \text{for} \qquad ih < t \leq ih + h$$

where h is the length of the (constant) sampling period. We assume for the sake of simplicity that the horizon is a multiple of the sampling period, i.e. $T = Nh$. The Bellman equation becomes in this situation

$$V_i(x_i) = \min_{u_i} \left[\frac{1}{h} \int_{ih}^{ih+h} \frac{1}{2} x^T(t) Q x(t)\ dt + \frac{1}{2} u_i^T R u_i + V_{i+1}(x_{i+1}) \right] \tag{3}$$

$$V_N(x_N) = \frac{1}{2} x_N^T P x_N$$

where the Bellman function, $V_i(x_i)$, is the optimal cost to go. We will investigate the following candidate function

$$V_i(x_i) = \frac{1}{2} x_i^T S_i x_i$$

which obviously is satisfied for $i = N$. By notation $x_i = x(ih)$ and $x_N = x(T)$. Let for short $s = t - ih \leq h$. The solution to (2) is well known and is

$$\begin{aligned} x(t) &= e^{As}x_i + \int_0^h e^{A(s-\tau)}B\ d\tau\ u_i \\ &= \Phi_s x_i + \Gamma_s u_i \end{aligned}$$

where

$$\Phi_s = e^{As} \qquad \Gamma_s = \int_0^h e^{A(s-\tau)}B\ d\tau = \int_0^h e^{A\tau}B\ d\tau$$

If we furthermore introduces the integrals

$$Q_1 = \frac{1}{h}\int_0^h \Phi_s^T Q \Phi_s\ ds \quad Q_{12} = \frac{1}{h}\int_0^h \Phi_s^T Q \Gamma_s\ ds \quad Q_2 = \frac{1}{h}\int_0^h \Gamma_s^T Q \Gamma_s\ ds$$

then the inner part of the minimization in (3) can be written as

$$I = \frac{1}{2}\begin{bmatrix} x_i^T & u_i^T \end{bmatrix}\begin{bmatrix} Q_1 & Q_{12} \\ Q_{12}^T & Q_2 \end{bmatrix}\begin{bmatrix} x_i \\ u_i \end{bmatrix} + \frac{1}{2}u_i^T R u_i + \frac{1}{2}\begin{bmatrix} x_i^T & u_i^T \end{bmatrix}\begin{bmatrix} \Phi_h^T S_{i+1}\Phi_h & \Phi_h^T S_{i+1}\Gamma_h \\ \Gamma_h^T S_{i+1}\Phi_h & \Gamma_h^T S_{i+1}\Gamma_h \end{bmatrix}\begin{bmatrix} x_i \\ u_i \end{bmatrix}$$

or as

$$I = \frac{1}{2}\begin{bmatrix} x_i^T & u_i^T \end{bmatrix}\begin{bmatrix} Q_1 + \Phi_h^T S_{i+1}\Phi_h & Q_{12} + \Phi_h^T S_{i+1}\Gamma_h \\ Q_{12}^T + \Gamma_h^T S_{i+1}\Phi_h & R + Q_2 + \Gamma_h^T S_{i+1}\Gamma_h \end{bmatrix}\begin{bmatrix} x_i \\ u_i \end{bmatrix}$$

That means that the control is given by:

$$\boxed{u_i = -\left[R + Q_2 + \Gamma_h^T S_{i+1}\Gamma_h\right]^{-1}\left[Q_{12}^T + \Gamma_h^T S_{i+1}\Phi_h\right]x_i}$$

where S_i is given by the recursion

$$S_i = Q_1 + \Phi_h^T S_{i+1}\Phi_h - \left[Q_{12} + \Phi_h^T S_{i+1}\Gamma_h\right]\left[R + Q_2 + \Gamma_h^T S_{i+1}\Gamma_h\right] - 1\left[Q_{12}^T + \Gamma_h^T S_{i+1}\Phi_h\right]$$

$$S_N = P$$

This ensures that the candidate function satisfy the Bellman equation. Notice, the solution to this problem coincide with the solution to a discrete time problem, just with transformed weight matrices.

We will now use the matrix exponential for determine the these weight matrices. Let

$$\Sigma = \begin{bmatrix} Q_1 & Q_{12} \\ Q_{12}^T & Q_2 \end{bmatrix}$$

For determining the matrices, define the square matrix

$$\mathbf{A} = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix}$$

Then by the Lemma

$$e^{\mathbf{A}s} = \begin{bmatrix} e^{As} & \int_0^s e^{A(s-t)}dt \\ 0 & I \end{bmatrix} = \begin{bmatrix} e^{As} & \int_0^s e^{At}dt \\ 0 & I \end{bmatrix} = \begin{bmatrix} \Phi_s & \Gamma_s \\ 0 & I \end{bmatrix}$$

If we furthermore define the matrix

$$\mathbf{Q}_c = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix}$$

it is straight forward to check that

$$\Sigma = \int_0^h e^{\mathbf{A}^T s} \mathbf{Q}_c e^{\mathbf{A} s} \, ds = \int_0^h \begin{bmatrix} \Phi_s^T & 0 \\ \Gamma_s^T & I \end{bmatrix} \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Phi_s & \Gamma_s \\ 0 & I \end{bmatrix} \, ds$$

If we compute the matrix:

$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} -\mathbf{A}^T & \mathbf{Q}_c \\ 0 & \mathbf{A} \end{bmatrix} h \right)$$

then

$$F_{11} = e^{-\mathbf{A}^T h} \qquad F_{22} = e^{\mathbf{A} h}$$

and

$$F_{12} = \int_0^h e^{-\mathbf{A}^T (h-s)} \mathbf{Q}_c e^{\mathbf{A} s} \, ds = e^{-\mathbf{A}^T h} \int_0^h e^{\mathbf{A}^T s} \mathbf{Q}_c e^{\mathbf{A} s} \, ds$$

and finally

$$\boxed{\Sigma = F_{22}^T F_{12}}$$

and

$$F_{22} = \begin{bmatrix} \Phi_h & \Gamma_h \\ 0 & I \end{bmatrix}$$

The Matlab implementation of this algorithm is listed in Appendix C as conc2d.m.

References

[1] T. Chen and B. Francis. *Optimal Sampled-Data Control Systems*. Communications and Control Engineering Series. Springer-Verlag New York Inc., 1995.

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

[3] T. Söderström. *Discrete-Time Stochastic Systems: Estimation and Control*. Prentice Hall, 1994.

A Sampled-data control

The problem in section 6 can be treated in a more general framework (see e.g. [1] page 238). Here the problem will be solved in the same setting as in section 6.

Consider the problem of controlling a continuous time LTI system

$$\begin{aligned} \frac{d}{dt}x(t) &= Ax(t) + Bu(t) \qquad x(0) = \underline{x}(0) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{4}$$

such that the objective function

$$J = \frac{1}{2}x^T(T)Px(T) + \frac{1}{2}\int_0^T \|y(t)\|^2 \, dt \tag{5}$$

is minimized. Notice the traditional weights is embedded in the output matrices, C and D. The alternative formulation

$$J = \frac{1}{2}x^T(T)Px(T) + \frac{1}{2T}\int_0^T \begin{bmatrix} x^T(t) & u^T(t) \end{bmatrix} \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} dt$$

is easily obtained from the methods described in B. The control actions are, as in section 6, assumed to be constant between samples, i.e.

$$u(t) = u_i \qquad \text{for} \qquad ih < t \leq ih + h$$

It is quite easy to check that the cost function in (5) is equivalent with the discrete time problem of controlling the system

$$x_{i+1} = \Phi x_i + \Gamma u_i \qquad x_0 = \underline{x}_0$$

such that the cost function

$$J = \frac{1}{2}x_N^T P x_N + \frac{1}{2}\sum_{i=0}^{N-1} \begin{bmatrix} x_i^T & u_i^T \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix}$$

is minimized. Here:

$$Q_{11} = \int_0^h e^{A^T s} C^T C e^{As} \, ds$$

$$Q_{12} = \int_0^h e^{A^T t} C^T \left[D + C\int_0^t e^{As} B \, ds \right] dt$$

$$Q_{22} = \int_0^h \left[D + C\int_0^t e^{As} B \, ds \right]^T \left[D + C\int_0^t e^{As} B \, ds \right] dt$$

Let

$$\Sigma = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \qquad \text{and} \qquad \mathbf{Q}_c = \begin{bmatrix} C^T \\ D^T \end{bmatrix} \begin{bmatrix} C & D \end{bmatrix} = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix}$$

Define the square matrix

$$\mathbf{A} = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix}$$

Then by the Lemma

$$e^{\mathbf{A}t} = \begin{bmatrix} e^{At} & \int_0^h e^{A(t-s)} ds \\ 0 & I \end{bmatrix} = \begin{bmatrix} e^{At} & \int_0^t e^{As} ds \\ 0 & I \end{bmatrix}$$

It is straight forward to check that

$$\Sigma = \int_0^h e^{\mathbf{A}^T s} \begin{bmatrix} C^T \\ D^T \end{bmatrix} \begin{bmatrix} C & D \end{bmatrix} e^{\mathbf{A}s}\, ds = \int_0^h e^{\mathbf{A}^T s} \mathbf{Q}_c e^{\mathbf{A}s}\, ds$$

Compute the matrix

$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} = exp\left(\begin{bmatrix} -\mathbf{A}^T & \mathbf{Q} \\ 0 & \mathbf{A} \end{bmatrix} h \right)$$

Then

$$\boxed{\Sigma = F_{22}^T F_{12}}$$

and

$$F_{22} = \begin{bmatrix} \Phi & \Gamma \\ 0 & I \end{bmatrix}$$

The Matlab implementation of this algorithm is listed in Appendix C as smplq.m.

B Manipulation of cost functions

Connection between output point of view and cost functions. Consider

$$z = Cx + Du = \begin{pmatrix} C & D \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix}$$

Then

$$J = \|z\|^2 = \begin{pmatrix} x^t & u^T \end{pmatrix} \begin{bmatrix} C^T \\ D^T \end{bmatrix} \begin{bmatrix} C & D \end{bmatrix} \begin{pmatrix} x \\ u \end{pmatrix} = \begin{pmatrix} x^t & u^T \end{pmatrix} Q \begin{pmatrix} x \\ u \end{pmatrix}$$

where

$$Q = \begin{pmatrix} C^T C & C^T D \\ D^T C & D^T D \end{pmatrix}$$

On the other hand, given Q we can easily find C and D. Assuming Q to be positive definite we can perform a cholesky factorization, i.e. find H such that

$$Q = H^T H$$

Then (using a matlab notation)

$$C = H(:, 1:n) \qquad D = H(:, n+1:end)$$

If Q is not positive definite (but still symmetric) we have to user SVD instead. Then

$$Q = USU^T$$

where U is an upper triangular matrix and S is a diagonal matrix. In this case

$$H = \sqrt{S}U^T$$

C Codes

The codes listed below is the Matlab implementations of the algorithms described in the previous chapters. The listning occur in their stripped version, i.e. without any significant input checking or options.

```
function [Ad,Bd] = c2d(A,B,h)
%Usage [Ad,Bd] = c2d(A,B,h)
%
%Find the discret time model
%
% x(t+1)=Ad*x(t)+Bd*u(t)
%
%when the continuous time model
%
% dot(x)=A*x+B*u
%
%is sampled with h as sampling period and the input is
%constant between samples.

[n,m]=size(B);
F=expm([[A B]*h; zeros(m,n+m)]);
Ad=F(1:n,1:n);
Bd=F(1:n,n+1:end);
```

```
function (S,Ad)=nc2d(A,R,h)
% Usage: (S,Ad)=nc2d(A,R,h)

n=length(A);
F=exp([-A R; zeros(n,n) A']*h);
F12=F(1:n,n+1:end);
F22=F(n+1:end,n+1:end);
Ad=F22';
S=Ad*F12;
```

```
function [Wc,Ad]=syscwc(A,B,h)

n=length(A);
F=exp([-A B*B'; zeros(n,n) A']*h);
F12=F(1:n,n+1:end);
F22=G(n+1:end,n+1:end);
Ad=F22';
Wc=Ad*F12;
```

```
function [Wo,Ad]=syscwo(C,A,h)
```

```
n=length(A);
F=exp([-A' C'*C; zeros(n,n) A]*h);
F12=F(1:n,n+1:end);
F22=G(n+1:end,n+1:end);
Ad=F22;
Wc=Ad'*F12;
```

```
function [Q1,Q2,Q12,Ad,Bd]=conc2d(A,B,Q,R,h)
% Usage: [Q1,Q2,Q12,Ad,Bd]=conc2d(A,B,Q,R,h)

[n,m]=size(B);
Qc=[Q zeros(n,m); zeros(m,n) R*h];
Ac=[A B; zeros(n,n+m)];
F=exp([-A' Qc; zeros(n.n) A]*h);
F22=F(n+m+1:end,n+m+1:end);
F12=F(1:n+m,n+m+1:end);
Q=F22'*F12/h;
Q1=Q(1:n,1:n);
Q2=Q(n+1:end,n+1:end);
Q12=Q(1:n,n+1:end);
Ad=F22(1:n,1:n);
Bd=F22(1:n,n+1:end);
```

```
function [Ad,Bd,Cd,Dd]=smplq(A,B,C,D,h)
% Usage [Ad,Bd,Cd,Dd]=smplq(A,B,C,D,h)
%       sysd=smplq(sysc,h)

if nargin==5,
 typ=1;
elseif nargin==2,
 typ=2,
 [A,B,C,D]=ssdata(sysc);
else
 disp('Wrong argument list');
 return 1
end

[n,m]=size(B);
Qc=[C';D']*[C D];
Ac=[A B; zeros(n,n+m)];
F=exp([-A' Qc; zeros(n.n) A]*h);
F22=F(n+m+1:end,n+m+1:end);
F12=F(1:n+m,n+m+1:end);
Q=F22'*F12;
Ad=F22(1:n,1:n);
Bd=F22(1:n,n+1:end);
[U,S]=svd(Q);
H=sqrt(S)*U';
Cd=H(:,1:n);
```

```
Dd=H(:,n+1:end);

if typ==2,
 Ad=ss(Ad,Bd,Cd,Dd,h),
end
```