Democratic and Popular Republic of Algeria
Ministry of Higher Education and Scientific Research
University of M'hamed Bougara-Boumerdes

Electrical and Electronic Engineering Institute

Doctoral Thesis

Presented by:

YAICI Malika

In view to obtain a DOCTORATE degree in

Option: CONTROL

MIMO Systems Compensator Design

In front of the Examiner Board:

M^r Fares	BOUDJEMA	Professor	ENP El Harrach	President
M^r Kamel	HARICHE	Professor	UMB Boumerdes	Supervisor
M^r Djamel	BOUKHETALA	Professor	ENP El Harrach	Examiner
M^r Mohammed	TADJINE	Professor	ENP El Harrach	Examiner
M^r Djamel	BENAZZOUZ	Professor	UMB Boumerdes	Examiner

-2014/2015-

ملخص

في هذه ألأطروحة يقترح عملية تصميم لتحقيق تعيين هيكل ذآتي "eigenstructure" بإستخدام التنسيب كتل أقطاب مع معوض دينآميكي لانظمة متعددة المدخلات و متعددة المخرجات خطية و ثابة في الزمن. تم تحويل أنظمة موصوفة بمعادلات دولية إلى أنظمة التي وصفها مصفوفات الكسور, و لهذه الأخيرة, يطلق على القيم الذاتية إسم القيم الكامنة و تسمى المتجهات الذاتية الناقلات الكامنة. الطريقة المقرحة هنا تسمح تعيين مجموعة كاملة (حتى أكثر) من القيم الكامنة و الناقلات الكامنة المحصولة من هيكل ذاتي مرغوب. قدم تذكير نضرية المصفوفات متعددة الحدود, و تم تطوير طريقة لبناء كتل جذور لمصفوفات متعددة الحدود من القيم و الناقلات الكامنة. ثم تم عرض توصيف الفضاء الدولي و توصيف بمصفوفة الجزئيات. بعد ذلك تم تأسيس العلاقة بين الهيكل الذاتي و الهيكل الكامن .بإلإظافة إلى ذلك ,تم الحصول على نتيجة : طريقة سريعة و سهلة لحل مشكلة حساب القيم الذاتية لمصفوفات متعددة الحدود العادية. دراسة التحكم بحلقة مغلقة قد خضعت , و تتضمن دراسة تكوينات مختلفة و تطوير المعادلات المرتبطة بإلمعوض. و قد تم إختيار التكوين حلقة مغلقة بإلمدخلات و المخرجات لتصميم المعوض الذي يسمح وضع كتل أقطاب من القاسم المطلوب شيد الهيكل الكامن المرغوب. أخيرا تم تصميم المعوض لنظام التحكم في الطيران الهليكوبتر لتوضيح النهج المقترح.

الكلمات المفتاحية: تنسب كتل أقطاب, مراقبة الدارة المغلقة, تمثيل المصفوفة متعدد الحدود, تعين هيكل ذاتي, تعويض دنناميكى

Résumé

Dans cette thèse, un processus de conception est proposé pour réaliser le placement d'une structure propre à l'aide de placement de pôles bloc en utilisant un compensateur dynamique pour des systèmes MIMO linéaires et invariants dans le temps. Des systèmes représentés par des équations d'état sont transformés en systèmes décrits par des matrices polynomiales et pour ces derniers, les valeurs propres sont appelés valeurs latentes et les vecteurs propres sont appelés vecteurs latents. La méthode proposée ici permet de placer l'ensemble (et même plus) des valeurs latentes et vecteurs latents obtenus à partir d'une structure propre désirée. Un rappel de la théorie des matrices polynomiales a été réalisé et une méthode pour construire des racines bloc d'une matrice polynômiale à partir de valeurs et vecteurs latents a été développée. Ensuite, la représentation espace d'état et la représentation par matrices polynomiales, ont été étudiés. La relation entre structure propre et structure latente a été établi. En outre, un résultat conséquent a été obtenu: une méthode rapide et facile pour résoudre le problème du calcul de valeurs propres pour les matrices polynômiales régulières. Une étude sur la commande par boucle fermée a été effectuée, qui comprend une étude de différentes configurations et le développement des équations de compensation associés. La commande par retour des entrées et des sorties a été choisi pour concevoir le compensateur qui permet le placement de pôles par bloc d'un dénominateur désiré construit à partir d'une structure latente désirée. Enfin, pour illustrer l'approche proposée, un compensateur pour système de commande de vol d'hélicoptère a été conçu.

Mots clés : Placement de pôles bloc, Commande par boucle fermée, Représentation par matrices polynomiales, Placement de structure propre, Compensation dynamique.

Abstract

In this thesis a design process is proposed to achieve eigenstructure assignment using block poles placement with a dynamic compensator for linear invariant MIMO systems. Systems described in state space equations are transformed to systems in matrix fractions description and for such systems, eigenvalues are called latent values and eigenvectors are called latent vectors. The method proposed here allows the assignment of the whole set (and even more) of latent values and vectors obtained from a desired eigenstructure. A review of matrix polynomial theory has been achieved and a method to construct block roots of a matrix polynomial from latent values and vectors has been developed. Then the state space description and the matrix fraction description have been presented. The relationship between eigenstructure and latent structure has also been established. Additionally, a consequent result, consisting on a quick and easy method to solve the polynomial eigenvalue problem for regular matrix polynomials, has been obtained. Furthermore, a study on feedback control has been undertaken; this includes a study on different feedback configuration and the development of the associated compensator equations. The input-output feedback configuration has been chosen to design the compensator which allows the placement of block poles of a desired denominator constructed from a desired latent structure. Finally, to illustrate the proposed approach, a compensator for a helicopter flight control system has been designed.

Keywords: Block poles placement, Feedback control, Matrix fraction descritpion, Eigenstructure assignment, Dynamic compensation.

Acknowledgements

I would like to express my great gratitude and respect to Prof. Kamel Hariche, my supervisor, for his help, suggestions and constant support during this research.

Prof. Tim Clarke expressed his interest in my work and supplied me with so many documents, and tried to answer my "stupid" questions till the last minute of this research work. My stays at York have always been successful. Thanks a lot.

My sincere gratitude is due to Prof. Boudjema for his acceptance to be the president of the jury and to Prof. Boukhetala, Prof. Tadine and Prof. Benazzouz for their kind acceptance to be members of the jury.

I take this opportunity to thank my students, master and engineer, of the computer department for their incessant questioning, more particularly Athmane Bentahar, Hayet Khaled and Zakia Khaled, my co-authors in one of my contributions.

I would also like to thank on this occasion my friends of the Department of Computer: Hassina, Karima, Soraya; the staff of the University of Bejaia and Prof. Saidani Boualem; my friends of the University of Boumerdes: M. Dehmas; and my friends of the University of York: JingLing, Mungo, Lewis, Maureen and Mav.

Of course, I am grateful to my family for their patience and *love*: my mother, my brothers and sisters, my brothers in law and my sisters in law.

Finally, special thanks for: Sarah, Khelaf, Yucef, Elias, Massine, Amir, Yuran, Wilan, Luna, Ines, Imene, Sofiane and Amine.

Malika Yaici
Bejaia, Algeria

This thesis is dedicated to the memory of my father ***Vava****.*
He was the only one to regret my abandon and to encourage me to resume and finish my thesis.
I pray God to grant him with His Infinite Mercy and welcome him in His Eternal Paradise.
God bless him.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

MIMO	Multiple Inputs Multiple Outputs.
SISO	Single Input Single Output.
SSD	State Space Description.
MFD/RMFD/LMFD	Matrix Fraction Description/ Right MFD / Left MFD.
TF/TFMD	Tranfer Function/TF Matrix Description.
PEP	Polynomial Eigenvalue Problem
EA	Eigenstructure Assignment.
FBW	Fly-By-Wire.
FBL	Fly-By-Light.
AFCS	Automatic Flight Control System
Def.Stan.00-970	UK Ministry of Defence Standard 00-970.
PDME	Polynomial Diophantine Matrix Equation.
MPI	Message Passing Interface.
MPICH	MPI Chameleon.

Nomenclature

$(.)^{-1}$	Inverse
$(.)^{T}$	Transpose
$(.)^{+}$	Pseudo-inverse
$(.)^{*}$	Conjugate
$(.)^{(i)}$	i^{th} derivative
I_{ij}	Identity matrix of dimension $i \times j$
0_{ij}	Zero matrix of dimension $i \times j$
$A(\lambda)/A_R/A_L$	Matrix polynomial/Right/Left
λ_i	i^{th} eigenvalue/latent value
Λ	Spectral matrix of eigenvalues/latent values
v_i/w_i	Right/left latent vector
V/W	Modal matrix of right/left latent vectors
V_R/V_L	Right/ Left block Vandermonde matrix
R_i/L_i	Right/ Left Solvent (block Root)
σ	Spectrum of a rational or polynomial matrix
S_i	i^{th} spectral factor
A_{lb}/A_{rb}	Right/low block companion form
δ_{ij}	Dirac delta function
A	State matrix
B	Input matrix
C	Output matrix
E	Input-Output matrix
x	Vector of states
u	Vector of inputs

y	Vector of outputs
$\tilde{v}_i/\tilde{w}_i$	Right/left eigenvector
$\tilde{V}/\tilde{W}$	Modal matrix of right/left latent vectors
$\mathcal{C}n$	Controllability matrix
μ	Controllability index
T_c/T_{c1}	Controller similarity transformation matrices
$\{A_c,B_c,C_d\}$	Block Controller state-space system
$\mathcal{O}$	Observability matrix
ν	Observability index
T_o/T_{o1}	Observer similarity transformation matrices
$\{A_o,\ B_o,\ C_o\}$	Block Observer state-space system
$\{A_d,\ B_d,\ C_d\}$	Block Diagonal state-space system
$D/D_R/D_L$	Denominator (Right / Left) of MFD system
$N/N_R/N_L$	Numerator (Right /Left) of MFD system
$\bar{v}_i/\bar{w}_i$	Right/left eigenvector of transformed system
$T(s)$	Transfer function
K	Gain feedback matrix
$G(s)$	Open loop system transfer function
G_c	Compensator transfer function
D_c/N_c	Denominator and numerator of G_c
G_{c0}/G_{c1}	Compensators of input-output feedback configuration
G_p	Pre-compensator transfer function
$G_{cl}(s)$	Closed loop system transfer function
$d\tilde{v}_i/d\tilde{w}_i$	Desired right/left eigenvector
$d\tilde{V}/d\tilde{W}$	Matrix of desired right/left eigenvectors
dv_i/dw_i	Desired right/left latent vector
dV/dW	Matrix of desired right/left latent vectors
dR/dL	Desired right/left solvent
$d\Lambda$	Modal matrix of desired eigenvalues
D_c	Compensator denominator

D_f	Desired denominator
L_c, M_c	Compensator numerators
$G_p(s)$	Pre-compensator transfer function
N_p	Pre-compensator numerator

Helicopter modelling:

θ/ ϕ	Pitch/ Roll angle
q/ p	Pitch/ Roll rate
r	Yaw rate
u	Forward speed
v	Lateral speed (sideslip velocity)
w	Vertical speed (heave velocity)
$\dot{h}$	Vertical speed in inertial frame
A_1/ B_1	Lateral/ Longitudinal Cyclic Pitch inputs
θ_0/θ_t	Main/Tail rotor collective pitch input

Chapter 1

General Introduction

1.1 Introduction

In most applied mathematical research, the aim is to investigate and control a given system. A system is defined to mean a collection of objects which are related by interactions and produce various outputs in response to different inputs.

Examples of such systems are chemical plants, aircrafts, spacecraft, biological systems, or even the economic structure of a country or region. The control problems associated with these systems might be the production of some chemical product as efficiently as possible, automatic landing of aircraft, rendezvous with an artificial satellite, regulation of body functions such as heartbeat or blood pressure, and the ever-present problem of economic inflation.

To be able to control a system, we need a valid mathematical model. However practical systems are inherently complicated and highly non-linear. Thus, simplifications are made, such as the linearization of the system. Error analysis can then be employed to give information on how valid the linear mathematical model is, as an approximation to the real system.

It is desirable that systems are controlled automatically, that is, they adapt to behave in a specified manner, without direct intervention. To achieve automatic control, information describing the system and the way it changes is needed. This is provided by a feedback control system, which calculates the difference between the measured variables and the desired output responses, and attempts to change the system to compensate for this.

Ideally, we would like to be able to measure all of the variables, or states of a system in order to design a feedback. If this is the case, then we are performing state feedback. In practice, not all of the system states are available: the feedback then has to use the outputs to control the system. This is called output feedback.

1.2 Problem positioning

In modelling systems or plants, a number of descriptions can be used to represent them. Two commonly used model descriptions for linear time-invariant MIMO systems are the state space description (SSD) and the transfer function matrix description (TFMD) or matrix fraction description (MFD).

In this thesis we will consider a linear multi-variable time-invariant system described by the following state space equations:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Eu \end{cases} \tag{1.2.1}$$

Where A is an $n \times n$ state matrix, B is an $n \times m$ input matrix, C is a $p \times n$ output matrix and E is a $p \times m$ transmission matrix.

Such systems can be studied via the eigenstructure, eigenvalues and eigenvectors, of the state matrix A. The eigenvalues and eigenvectors can determine system performance and robustness far more directly and explicitly than other indicators. Hence their assignment should improve feedback system performance and robustness distinctly and effectively.

Eigenstructure assignment (EA) is the process of applying negative feedback to a linear, time-invariant system with the objective of forcing the eigenvalues and eigenvectors to become as close as possible to a desired eigenstructure.

EA, in common with other multivariable design methodologies, is inclined to use all of the available design freedom to generate a control solution. It is a natural choice for the design of any control system whose desired performance is readily represented in terms of an ideal eigenstructure. Many research works has been done on EA [1, 2, 3, 4, 5] and more specifically on flight control systems [6, 7, 8].

The same system can be described by a m-inputs p-outputs transfer function $G(s)$ in matrix fractions as follows:

$$G(s) = N_R(s)D_R^{-1}(s) \text{ or } G(s) = D_L^{-1}(s)N_L(s) \tag{1.2.2}$$

Where $D_R(s)$ ($D_L(s)$) and $N_R(s)$ ($N_L(s)$) are matrix polynomials of degree less or equal to r and of order m or p, r being the controllability or observability index. These matrix polynomials are polynomials which coefficients are matrices of right dimensions.

In recent years, MFDs have been used widely in linear control theory, example to define poles and zeros of transfer function matrices [9], or to obtain controllable and observable state space realizations of transfer functions [10]. Many authors have considered using matrix polynomials and block roots for solving some linear algebra problems or control problems such as: compensator design using numerical optimization techniques [11], block partial fraction expansion of a matrix fraction description (MFD) with single and repeated poles [12], reduction of the order of single-input single-output and multi-input multi-output robust controllers [13], feedforward compensator in two degree-of-freedom controllers [14] etc.

The design of a control system is concerned with the arrangement, or the plan, of the system structure and the selection of suitable components and parameters. Often the alteration or adjustment of a control system, in order to provide a suitable performance, is called compensation.

In redesigning a control system in order to alter the system response, an additional component is inserted within the structure of the feedback system. It is this additional component or device that equalizes or compensates for the performance deficiency.

Because the transient performance specifications are often cast in terms of closed loop pole positions, it seemed that a pole placement technique would be an appropriate design approach and be a direct method to design dynamic compensator for MIMO systems. Multivariable compensator design has been a fertile research area for many years, in particular, the problem of pole assignment which has been studied thoroughly [15, 16, 17, 18, 19, 20, 21, 22].

In our case, performance specifications are better expressed in terms of eigenvalues and eigenvectors, and the best way to assign them is to design a compensator to place block poles constructed with this desired eigenstructure.

Important remark: For SISO systems, computations involving transfer function are relatively straight-forward; However, this is not the case for MIMO systems. So in this thesis, only MIMO systems will be considered.

1.3 Objective

The initial objective of this research work was to find a method to assign a desired latent structure, thus latent values and latent vectors, for systems described in MFD, using block

poles placement.

The idea is to construct desired block poles (and eventually block zeros) from a desired set of latent values and linearly independent latent vectors.

From the block poles, we can construct a desired closed loop matrix polynomial.

Then we have to determine a compensator to assign this matrix polynomial as a denominator of the closed loop system. The block zeros can, also, be placed through the denominator of the compensator, or via a designed pre-compensator.

In literature, systems described in matrix transfer functions, to which a desired eigenstructure has been specified, do not exist. So the idea was to apply the method for systems described in SSD with a desired eigenstructure.

So a method to transform the system into a MFD has been studied and then the eigenstructure is transformed into a latent structure. A method to determine latent vectors from eigenvectors has been elaborated, and consequently a method to compute the inverse of a matrix polynomial has been proposed.

The conversion methods to transform a controllable or/and observable system in SSD gives a proper, coprime and row or column reduced MFD system. These are the conditions to solve the compensator equation or the Diophantine equation. So a study of methods to solve such equation has been undertaken and an improvement, in the form of a parallelization, has been proposed.

Finally, an application to validate the design process has to be done. After an internship in the University of York, we agreed on the Helicopter flight control. Helicopters, being inherently complex systems, required the development of control system technology before they could become useful. Without feedback control, helicopters tend to be highly unstable and difficult to fly. Many research works has been done on the helicopter flight control, and a linear state space model has been elaborated to which a desired eigenstructure has been designed. Furthermore, the system verifies the conditions of applicability of the proposed design approach.

The proposed design approach is more valuable for multi-variable systems so only MIMO systems have been treated in this thesis.

1.4 Thesis overview

To produce an accessible, readable document, the organization of this thesis is such that the chapters form self-contained units. In each case an introduction to the chapter will be found at the start, and a summary and list of references at the end. A number of appendices follow the main body of the thesis, and contain supporting material which would otherwise break the flow of the main text.

Chapter 2 introduces a review of matrix polynomials. The concept of latent values and latent vectors are defined. The most important part is on solvents, and the particular case of a complete set of solvents. Spectral factorization of matrix polynomials, block companion matrix forms are also presented. As a first contribution, a method to construct solvents from a set of latent values and latent vectors is presented, and conditions to the existence of such solvents and the complete set of solvents are given. A consequent result is the development of a method to compute the inverse of matrix polynomial, which is detailed in Appendix C.

The two main descriptions, used to represent MIMO systems, are given in chapter 3. In the SSD, some notions as eigenstructure, controllability and observability, and transformations into block companion forms are given. In the MFD, definition of poles and zeros of a system, the block partial fraction expansion of a system transfer function, and some canonical forms are given. Important properties of MFD are introduced which will be needed for the resolution of the Diophantine equation. Methods to transform SSD into MFD are given. One of the important contributions of this thesis is the establishment of the relationship between a latent structure and an eigenstructure, and it is given with a consequent result, a solution to the polynomial eigenvalue problem.

In chapter 4, the main methods of feedback control of systems described by state equations and matrix fraction description are given. We started by methods of poles placement using state feedback, and output feedback for systems described in SSD, as well as methods of eigenstructure assignment. Then, the different feedback configurations for systems described in MFD are given with the development to determine the compensator equations. Designing pre-compensators is also introduced in this chapter.

Chapter 5 details the main contribution of this thesis. The steps of the design process are detailed with an illustrative example for more clarity. The last step of this design process is

solving the Diophantine equations, and methods to achieve this are given in appendix A. An improvement of this method, by parallelizing the row-searching algorithm is given in appendix B.

As an example of compensator design application, the helicopter flight control has been chosen. The details of this example are given in chapter 6. A presentation of some notions on helicopters and helicopter flight are given, and then the design process is detailed following the steps given in the precedent chapter.

Finally, a conclusion and a summary of suggested further work can be found in Chapter 7.

Chapter Bibliography

[1] B. C. Moore, "On the flexibility offered by state feedback in multivariable systems beyond closed loop eigenvalue assignment", IEEE Trans. Autom. Control, Vol AC-21, No.5,pp. 689-692, 1976.

[2] J. Lu, H. D. Chiang and J. S. Thorp, "Eigenstructure assignment by decentralized feedback control", IEEE Trans. Autom. Control, Vol. AC-38, No.4, pp. 587-594, 1993.

[3] T. Clarke, S. J. Griffin and J. Ensor, "Output feedback eigenstructure assignment using a new reduced orthogonality condition", Int. J. Control, Vol. 76, No.4, pp. 390-402, 2002.

[4] O. Bachelier, J. Bosche and D. Mehdi, On pole placement via eigenstructure assignment approach, IEEE Trans. Autom. Control, Vol. AC-51, No.9, pp. 1554-1558, 2006.

[5] A. Pomfret and T. Clarke, "An extension to output-feedback eigenstructure assignment: structuring controllers by exploiting unused design freedom", Int. J. Control, Vol.82, No.2, pp. 207-216, 2009.

[6] A. N. Andry, E. Y. Shapiro and J. C. Chung, "Eigenstructure assignment for linear systems", IEEE Trans. Aerosp. Electron. Syst., Vol. 19, No.5, pp. 711-729, 1983.

[7] K. M. Sobel and E.J. Lallman, "Eigenstructure assignment for the control of highly augmented aircraft", J. of Guidance, Control, and Dynamics, Vol. 12, No.3, pp. 18-324, 1989.

[8] J. -F. Magni, "Multimodel eigenstructure assignment in flight-control design", Aerosp. Sci. Technol., Vol.3, pp. 141-151, 1999.

[9] C. A. Desoer and J. D. Shulman, "Zeros and poles of matrix transfer functions and their dynamical interpretation", IEEE Trans. Circuits Syst., Vol. CAS-21, pp. 3-8, 1974.

[10] W. A. Wolovich, "The determination of state space representations for linear multi-variable systems", Automatica, Vol.9, No.1, pp. 97-106, 1973.

[11] L. F. A. Pereira, J. F. Haffner and L. F. Chaves, "Compensator design in the frequency domain using optimization", 28^{th} Annual Conf. of the Industrial Electronics Society, Sevilla, Spain, Vol. 2, pp. 1463-1467, 2002.

[12] J. Leyva-Ramos, "A new method for block partial fraction expansion of matrix fraction descriptions", IEEE Trans. Autom. Cont., Vol. AC-36, No.12, pp. 1482-1485, 1991.

[13] V. R. Dehkordi and B. Boulet, "Robust controller order redeuction", Int. J. Cont., Vol. 84, No. 5, pp. 985-997, 2011.

[14] A. Leva and L. Bascetta, "On the design of the feedforward compensator in two-degree-of-freedom controllers", Mechatronics, Vol. 16, pp. 533-546, 2006.

[15] N. Munro and A. I. Vardulakis, "Pole shifting using output feedback", Int. J. Contr., Vol. 18, pp. 1267-1273, 1973.

[16] E. J. Davison and S H. Wang, "On pole assignment in linear multivariable systems using output feedback", IEEE Trans. Autom. Contr. , Vol AC-20, No.4, pp. 516-518, 1975.

[17] R. V. Patel, "Design of dynamic compensators for pole assignment", Int. J. Sys. Sci., Vol. 7, No.2, pp. 207-224, 1976.

[18] H. H. Rosenbrock and G. E. Hayton, "The general problem of pole assignment", Int. J. Contr., Vol. 27, pp. 837-852, 1978.

[19] B. D. O. Anderson and V. Kucera, "Matrix fraction construction of linear compensators", IEEE Trans. Autom. Contr., Vol.30, No.11, pp. 1112-1114, 1985.

[20] C.-L. Chen and Y. -Y. Hsu, "Pole assignment using dynamic output feedback compensators", Int J. Contr., Vol. 45, No.6, pp. 1985-1994, 1987.

[21] B. Shafai and L. H. Keel, "New pole placement algorithm: polynomial matrix approach", in Proc. of 9th American Control Conference, SanDiego, USA, Vol. 2, pp. 1684-1685, 1990.

[22] P. Stefanidis, A. P. Paplinski, and M. J. Gibbard, "Numerical operations with polynomial matrices", Lecture Notes in Control and Information Sciences, Vol. 171, Springer, Berlin, 1992.

Chapter 2

A Review on Matrix Polynomials

2.1 Introduction

Matrix polynomials, (or Polynomial matrices), also called λ-matrices, arise naturally when modelling physical systems. They play a central role in mathematics, in mechanics and robotics as hermitian quadratic matrix polynomials, in acoustics and fluid dynamics as quadratic matrix polynomials, etc. In control engineering, matrix polynomials are used as a mathematical description of the dynamics of multivariable systems. These representations are used as an alternative to state-space representation when designing controllers for linear systems.

In this chapter, a recall of the notions of matrix polynomials will be presented, such as latent values and latent vectors which, by analogy, are compared to eigenvalues and eigenvectors of a rational matrix. The concepts of solvents (also called block roots), complete set of solvents, and spectral factorization of matrix polynomials are also detailed. As a contribution, a method to construct solvents from a set of latent values and latent vectors is presented, and conditions to the existence of such solvents, and the complete set of solvents are given.

2.2 Literature review

Matrix polynomials play a central role in the mathematical description of the dynamics of multivariable systems. This fact has led to an active research effort in matrix polynomials theory [1].

A theoretical introduction based on spectral approach of matrix polynomial theory is given in [2]. The same authors proposed an algorithm based on Jordan chains for the computation of a solvent of a matrix polynomial.

The algebraic theory of matrix polynomials has been investigated by Dennis et al.[3], Gohberg et al.[4, 5, 6], and in [7, 8, 9]. Spectral factors of a lambda matrix and right (left) solvents for a right (left) matrix polynomial have been defined. The different transformations between solvents and spectral factors are mainly proposed by Shieh and Tsay [8].

Various computational algorithms [3, 7, 4, 8, 10, 11] are available for finding the solvents and spectral factors of a matrix polynomial. In [12] is proposed an algorithm for the computation of the dominant solvents of a matrix polynomial. Methods and algorithms for numerical solutions of spectral problems for one and two parameter polynomial and rational matrices are given in [13].

Some mathematical theory of λ-matrices, in particular solvents and interpolating λ-matrices are discussed in [1]. In this paper, the necessary and sufficient conditions for the existence of solvents and the corresponding generalized Lagrange interpolating λ-matrices are given.

Several methods have been developed for solving the problem of a complete set of solvents and spectral factors, without prior knowledge of the latent roots and latent vectors of a matrix polynomial. For instance, Shieh et al. [11] have derived a generalized Newton's method. Dahimene [14] proposed a generalization of the Quotient-Difference algorithm for the computation of spectral factors of a matrix polynomial. Tsai et al. [9] have obtained several algorithms for computing the complete set of solvents and spectral factors.

The relationship between solvents and spectral factors are explored by Shieh and Tsay in [8] and various transformations have been proposed. The condition for the existence and uniqueness of the complete set of solvents have been investigated by Lancaster [15], Dennis et al. [3] and Gohberg et al. [16].

We will show that right and left solvents can be constructed from latent roots and corresponding right and left latent vectors. It follows that each of the solvents contains a part of the latent structure of the matrix polynomial. A particular case of interest is the one where we can construct a set of solvents covering all the latent structure of a matrix polynomial. This set is referred to as a complete set of block roots.

2.3 Definitions

We introduce some basic concepts and notations.

2.3.1 Matrix polynomials

Definition 2.3.1. The following $m \times m$ matrix:

$$A(\lambda) = \begin{pmatrix} a_{11}(\lambda) & a_{12}(\lambda) & \cdots & a_{1m}(\lambda) \\ a_{21}(\lambda) & a_{22}(\lambda) & \cdots & a_{m2}\lambda) \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1}(\lambda) & a_{m2}(\lambda) & \cdots & a_{mm}(\lambda) \end{pmatrix} \tag{2.3.1}$$

is called a polynomial matrix, of order m, where $a_{ij}(\lambda)$ are scalar polynomials over the field of complex numbers.

From a polynomial matrix we can construct a matrix polynomial and vice-versa.

Definition 2.3.2. An mth order, rth degree matrix polynomial is given by:

$$A(\lambda) = A_r\lambda^r + A_{r-1}\lambda^{r-1} + \cdots + A_1\lambda + A_0 \tag{2.3.2}$$

Where A_i are $m \times m$ real matrices, λ a complex number, and the degree r is equal to the maximum of the degrees of the scalar polynomials of the polynomial matrix.

Definition 2.3.3. Let X be a $m \times m$ complex matrix. A right matrix polynomial is defined by:

$$A_R(X) = A_rX^r + A_{r-1}X^{r-1} + \cdots + A_1X + A_0 \tag{2.3.3}$$

And a left matrix polynomial is defined by:

$$A_L(X) = X^rA_r + X^{r-1}A_{r-1} + \cdots + XA_1 + A_0 \tag{2.3.4}$$

The matrix polynomial $A(\lambda)$ is called:

- monic if the leading matrix coefficient A_r is the identity matrix
- co-monic if the trailing matrix coefficient A_0 is also an identity matrix
- regular if the leading matrix coefficient A_r is non-singular.
- co-regular if the trailing matrix coefficient A_0 is also non-singular
- nonsingular if the determinant of $A(\lambda)$ is not identically zero.

- unimodular if the determinant of $A(\lambda)$ is a nonzero constant (independent of λ). In this case its inverse is also a matrix polynomial.

Remark 2.3.1. If the leading matrix coefficient A_r is non singular but not an identity matrix, then $A(\lambda)$ can be multiplied by A_r^{-1} to get a monic matrix polynomial.

In case A_r is singular and, if $det(A(\lambda)) \neq 0$ for any λ, thus A_0 is non-singular, then $A(\lambda)$ can be reversed to make A_r non-singular.

2.3.2 Latent values and vectors

Definition 2.3.4. The complex number λ_i is called a latent value of $A(\lambda)$ if it is a solution of the scalar polynomial equation $det(A(\lambda)) = 0$. The non-trivial vector v_i, solution of the equation $A(\lambda_i)v_i = 0$, is called a primary right latent vector associated to the latent value λ_i. Similarly the non trivial row vector w_i, solution of the equation $w_i A(\lambda_i) = 0$ is called a primary left latent vector associated with λ_i [11].

Remark 2.3.2. From the definition we can see that the latent problem is a generalization of the concept of eigenproblem for square matrices. We can consider that the classical eigen value/vector problem is finding the latent value/vector of a linear polynomial $\lambda I - A$.

We can also define the spectrum of a matrix polynomial as being the set of all its latent roots noted $\sigma[A(\lambda)]$.

Theorem 2.3.1. *The number of latent roots of a regular matrix polynomial $A(\lambda)$ in a domain enclosed by a contour Γ is equal to:*

$$\frac{1}{2\pi} \oint_{\Gamma} trace[A^{-1}(\lambda)A'(\lambda)]d\lambda \tag{2.3.5}$$

Where $A'(\lambda)$ is the first derivative of $A(\lambda)$. Each latent root being counted according to its multiplicity.

Proof: see [14]

A generalization of the latent root/vector is the Jordan chain which is defined by:

Definition 2.3.5. A set of vectors $\{x_0, x_1, ..., x_k\}$ in $\mathfrak{C}^{\Gamma}$ is called a right Jordan chain of length $k+1$ associated with the latent root λ_0 and primary latent vector x_0 if they satisfy the relation [14]:

$$\sum_{p=0}^{j} \frac{1}{p!} A^{(p)}(\lambda_0) x_{j-p} = 0, j = 0, ..., k \tag{2.3.6}$$

Where $A^{(p)}(\lambda)$ is the p^{th} derivative of $A(\lambda)$.

2.3.3 Solvents of a matrix polynomial

Definition 2.3.6. A right solvent (or a block root) R of a matrix polynomial $A(\lambda)$ is defined by:

$$A(R) = A_r R^r + A_{r-1} R^{r-1} + \cdots + A_1 R + A_0 = 0_m \tag{2.3.7}$$

And the left solvent of a matrix polynomial $A(\lambda)$ is defined by:

$$A(L) = L^r A_r + L^{r-1} A_{r-1} + \cdots + L A_1 + A_0 = 0_m \tag{2.3.8}$$

The relationship between a right and a left solvent is given by the following theorem:

Theorem 2.3.2. *If $A(\lambda)$ has n latent roots $\{\lambda_1 \lambda_2 \cdots \lambda_n\}$ and n corresponding right latent vectors $\{v_1 v_2 \cdots v_n\}$ and n corresponding left latent vectors $\{w_1 w_2 \cdots w_n\}$ then the right solvent R is related to the left solvent L by:*

$$R = VWL(VW)^{-1} \tag{2.3.9}$$

where

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix}, W = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}, V = \begin{pmatrix} v_1 & v_2 & \cdots & v_n \end{pmatrix} \tag{2.3.10}$$

Proof: see [17]

More generally, if we consider a monic matrix polynomial $A(\lambda)$, and v_i (w_i) a right (left) latent vector associated to the latent root λ_i and if all latent vectors satisfy the following:

$$A^{(p)}(\lambda) v_j|_{s=\lambda_i} = 0_m \text{ or } w_j A^{(p)}(\lambda)|_{s=\lambda_i} = 0_m \text{ for } p = 1...m_{i-1} \text{ and } j = 1..m$$

then there exist a right (left) solvent R_i (L_i) with multiplicity m_i.

In this case the right (left) solvents satisfy the following:

$$A^{(p)} R_i = 0_m \text{ or } A^{(p)}(L_i) = 0_m \text{ for } p = 0, 1, ...m_{i-1}$$

where (p) stand for the p^{th} derivative [11].

Theorem 2.3.3. *Given $A(\lambda)$ as in equation 2.3.2 then:*

- *the remainder of the division of $A(\lambda)$ on the right by binomial $(\lambda I - R)$ is equal to $A_R(R)$ (equation 2.3.3).*
- *the remainder of the division of $A(\lambda)$ on the left by binomial $(\lambda I - L)$ is equal to $A_L(L)$ (equation 2.3.4).*

Proof: see [18]

Corollary 2.3.4. *A matrix R (respectively L) is a right (respectively left) solvent of $A(\lambda)$ if and only if $(\lambda I - R)$ (respectively $(\lambda I - L)$) divides exactly $A(\lambda)$ on the right (respectively left).*

Proof: see [18]

Theorem 2.3.5. *The generalized right (left) eigenvectors of a right (left) solvent are generalized latent vectors of $A(\lambda)$.*

Proof: see [18]

In the following some important facts on solvents:

- If R is a right solvent (L a left solvent) , then there exist a λ-matrix $Q(\lambda)$ of degree $r-1$ such that $A(\lambda) = Q(\lambda)(I\lambda - R)$ $(A(\lambda) = (I\lambda - L)Q(\lambda))$ [15].
- If $A(\lambda)$ has $m * r$ latent roots then the r eigenvalues of the solvent and the $m(r-1)$ roots of $Q(\lambda)$ are latent roots of $A(\lambda)$ [3].
- Solvents of a λ-matrix do not always exist.
- W is called a week solvent if $A(W)$ is singular [3].

Remark 2.3.3. If $A(\lambda)$ has mr distinct latent roots and the set of right (left) latent vectors verify the condition that every m of them are linearly independent (Haar Condition) then there are exactly $\begin{pmatrix} mr \\ r \end{pmatrix}$ different right (left) solvents [3].

2.3.4 Block Vandermonde matrix

As for an eigenvalue system, a block Vandermonde matrix can be defined for solvents with particular properties [11].

Given a set of r right solvents of a matrix polynomial $A(\lambda)$ then the following $rm \times rm$ matrix:

$$V_R = \begin{pmatrix} I_m & I_m & \cdots & I_m \\ R_1 & R_2 & \cdots & R_r \\ \vdots & \vdots & \cdots & \vdots \\ R_1^{r-1} & R_2^{r-1} & \cdots & R_r^{r-1} \end{pmatrix} \tag{2.3.11}$$

is a "right" block Vandermonde matrix of order r.

And given a set of r left solvents of a matrix polynomial $A(\lambda)$ then the following $rm \times rm$ matrix:

$$V_L = \begin{pmatrix} I_m & L_1 & \cdots & L_r^{r-1} \\ I_m & L_2 & \cdots & L_r^{r-1} \\ \vdots & \vdots & \cdots & \vdots \\ I_m & L_r & \cdots & L_r^{r-1} \end{pmatrix} \tag{2.3.12}$$

is a "left" block Vandermonde matrix of order r.

If a right solvent R_i with multiplicity m_i exists then the corresponding general right block Vandermonde matrix will be constructed as follows [19]:

$$V_R = \begin{pmatrix} I_m & \cdots & I_m & 0_m & 0_m & \cdots & 0_m & \cdots & I_m \\ R_1 & \cdots & R_i & I_m & 0_m & \cdots & \vdots & \cdots & R_r \\ R_1^2 & \cdots & R_i^2 & 2R_i & I_m & \cdots & 0_m & \cdots & R_r^2 \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & I_m & \cdots & \vdots \\ R_1^{r-} & \cdots & R_i^{r-1} & \begin{pmatrix} r-1 \\ 1 \end{pmatrix} R_i^{r-2} & \begin{pmatrix} r-1 \\ 2 \end{pmatrix} R_i^{r-3} & \cdots & \begin{pmatrix} r-1 \\ m_i-1 \end{pmatrix} R_i^{r-m_i} & \cdots & R_r^{r-1} \end{pmatrix} \tag{2.3.13}$$

A similar left block Vandermonde matrix can be constructed with a left solvent L_i with multiplicity m_i.

2.3.5 Complete set of solvents

The nodal point is the existence of a set of m linearly independent right latent vectors associated with a set of m latent roots of $D(\lambda)$. Existence of such a set insures the existence of a right block root.

Theorem 2.3.6. *A block Vandermonde matrix as defined in 2.3.11, 2.3.12 and 2.3.13 are non-singular matrices if and only if the set of r solvents* $\{R_1 ... R_r\}$ *with multiplicities* $\{m_1, ..., m_r\}$

is a complete set.

Proof: see [3, 19]

Definition 2.3.7. If we let $\sigma[A(\lambda)]$ denote the set of all latent roots of $A(\lambda)$ and $\sigma[R_i]$ the set of eigenvalues of the right solvent R_i, then a complete set of right solvents is obtained if we can find r right solvents such that [11]:

$$\begin{cases} \bigcup_{i=1}^{r} \sigma(R_i) = \sigma(A(\lambda)) \\ \sigma(R_i) \cap \sigma(R_j) = \varnothing \end{cases} \tag{2.3.14}$$

and the block vandermonde matrix thus constructed is non-singular.

Just as for the right solvents, the existence of a left solvent depends on the existence of a set of m linearly independent left latent vectors. A complete set of left solvents (covering totally the latent structure of $A(\lambda)$) is obtained if we can find r left solvents where each solvent involves a distinct set of m latent roots of $A(\lambda)$. This in turn requires that for each such a distinct set, we can find a corresponding set of linearly left latent vectors.

The following definition summarizes that:

Definition 2.3.8. The set of left solvents of $A(\lambda)$, which satisfies the following properties, $r = \sum_{i=1}^{k} m_i$, $\sigma[A(\lambda)] = \bigcup_{i=1}^{k} \sigma[L_i]$ and a nonsingular block vandermonde matrix, is called the complete set of the left solvents of $A(\lambda)$ [11].

Remark 2.3.4. A complete set of right or left solvents will then describe completely the latent structure of $A(\lambda)$.

2.3.6 Construction of coefficients of a matrix polynomial

If a complete set of r (right or left) solvents exist then we can determine, from this set, the matrix coefficients (A_i i=0..r) of a monic matrix polynomial $A(\lambda)$ of the corresponding solvents [3].

1) For each right solvent R_j of $A(\lambda)$ we have:

$$R_j^r + A_{r-1}R_j^{r-1} + \cdots + A_1 R_j + A_0 = 0_m \; j = 1..r$$

then:

$$A_{r-1}R_j^{r-1} + \cdots + A_1R_j + A_0 = -R_j^r \; j = 1..r$$

Or

$$\begin{pmatrix} A_0 & A_1 & \cdots & A_{r-1} \end{pmatrix} \begin{pmatrix} I_m \\ R_j \\ \vdots \\ R_j^{r-1} \end{pmatrix} = -R_j^r \; j = 1..r$$

So:

$$\begin{pmatrix} A_0 & A_1 & \cdots & A_{r-1} \end{pmatrix} = -\begin{pmatrix} R_1^r & R_2^r & \cdots & R_r^r \end{pmatrix} V_R^{-1} \tag{2.3.15}$$

2) For each left solvent L_j of $A(\lambda)$ we have:

$$L_j^r + L_j^{r-1}A_{r-1} + \cdots + R_jA_1 + A_0 = 0_m \; j = 1..r$$

Then following the same development we obtain:

$$\begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_{r-1} \end{pmatrix} = -V_L^{-1} \begin{pmatrix} L_1^r \\ L_2^r \\ \vdots \\ L_r^r \end{pmatrix} \tag{2.3.16}$$

Theorem 2.3.7. *Equation 2.3.15 and equation 2.3.16 are satisfied iff the set of r left or right solvents is a complete set and V_L and V_R are respectively the left and right non-singular block Vandermonde matrices.*

Proof: See [3]

2.4 Spectral factorization of a matrix polynomial

Spectral factors and solvents are two close concepts. In the following some information on spectral factors and the link with solvents are given.

Definition 2.4.1. Let $A(\lambda)$ be a $q \times p$ matrix polynomial, then a $q \times q$ matrix polynomial $B(\lambda)$ is called a left divisor of $A(\lambda)$ if there exist a $q \times p$ matrix polynomial $X(\lambda)$ such that: $A(\lambda) = B(\lambda).X(\lambda)$. $A(\lambda)$ is called a right multiple of $B(\lambda)$.

- If the elementary divisors of $A(\lambda)$ are linear then $A(\lambda)$ has a complete set of right and left solvents [8].

- If the elementary divisors of $A(\lambda)$ are linear, then $A(\lambda)$ can be factored into the product of r linear monic matrix polynomials:

$$A(\lambda) = (\lambda I_m - S_r)(\lambda I_m - S_{r-1}) \cdots (\lambda I_m - S_1) \tag{2.4.1}$$

  Where the $m \times m$ matrices S_i are called spectral factors, and represent a complete spectral factorization of A [8].

- S_i is a right solvent of F_i where:

$$\begin{cases} F_1(\lambda) = A(\lambda) \\ F_i(\lambda) = F_{i+1}(\lambda)(\lambda I_m - S_i) \ i = 1..r \end{cases} \tag{2.4.2}$$

  and S_{n+1-i} is a left solvent of B_i where:

$$\begin{cases} B_1(\lambda) = A(\lambda) \\ B_i(\lambda) = (\lambda I_m - S_{n+1-i})B_{i+1}(\lambda) \ i = 1..r \end{cases} \tag{2.4.3}$$

- In general, S_i is a spectral factor of $A(\lambda)$, S_1 is a right solvent of $A(\lambda)$ and S_r is a left solvent of $A(\lambda)$ [8].

Definition 2.4.2. If $A(\lambda) = A_1(\lambda)A_2(\lambda)$ is a particular factorization of the monic matrix polynomial A, with $\sigma(A_1) \cap \sigma(A_2) = \varnothing$, then the monic matrix polynomials A_1 and A_2 are called spectral divisors of A [14].

2.5 Block companion forms

The best form for the analysis of matrix polynomials is the block companion form matrix. In this thesis, only two (needed) forms will be presented.

Given a monic r^{th} degree matrix polynomial as in equation 2.3.2, the associated low block companion form is:

$$A_{lb} = \begin{pmatrix} 0_m & I_m & 0_m & \cdots & 0_m \\ 0_m & 0_m & I_m & \cdots & 0_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \cdots & I_m \\ -A_0 & -A_1 & -A_2 & \cdots & -A_{r-1} \end{pmatrix} \tag{2.5.1}$$

And the associated right block companion form is:

$$A_{rb} = \begin{pmatrix} 0_m & 0_m & \cdots & 0_m & -A_0 \\ I_m & 0_m & \cdots & 0_m & -A_1 \\ 0_m & I_m & \cdots & 0_m & -A_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_m & 0_m & \cdots & I_m & -A_{r-1} \end{pmatrix} \tag{2.5.2}$$

with the following properties [3]:

- The eigenvalues of A_{lb} (A_{rb}) are latent roots of $A(\lambda)$.
- $det(A_{rb} - \lambda I) \equiv (-1)^{mr} det(A(\lambda)) \equiv det(A_{lb} - \lambda I)$
- A_{rb} is the block transpose of A_{lb}.
- If λ_i is a latent root of $A(\lambda)$ and v_i and w_i are the corresponding right and left latent vectors respectively then:
  - $\begin{pmatrix} v_i \\ \lambda_i v_i \\ \vdots \\ \lambda_i^{r-1} v_i \end{pmatrix}$ is a right eigenvector of A_{lb},
  - $\begin{pmatrix} w_i \\ \lambda_i w_i \\ \vdots \\ \lambda_i^{r-1} w_i \end{pmatrix}$ is a left eigenvector of A_{rb},
  - $\begin{pmatrix} v_i^{(r-1)} \\ \vdots \\ v_i^{(1)} \\ v_i \end{pmatrix}$ is a right eigenvector of A_{rb} where

$$\frac{A(\lambda)v_i}{(\lambda - \lambda_i)} = v_i\lambda^{r-1} + v_i^{(1)}\lambda^{r-2} + \cdots + v_i^{(r-1)}$$

  - $\begin{pmatrix} w_i^{(r-1)} \\ \vdots \\ w_i^{(1)} \\ w_i \end{pmatrix}$ is a left eigenvector of A_{lb} where

$$\frac{A(\lambda)w_i}{(\lambda - \lambda_i)} = w_i\lambda^{r-1} + w_i^{(1)}\lambda^{r-2} + \cdots + w_i^{(r-1)}$$

Remark 2.5.1. The block companion forms of matrix polynomials are very important because their forms are preferred in computations problems (latent roots and latent vectors, solvents, etc.). Software tools dealing with matrices outnumber those dealing with matrix polynomials.

2.6 Construction of right/left solvents

2.6.1 Introduction

A contribution (the first for this thesis) to the theory of matrix polynomials is given in this section and has been published in [20]. The problem of constructing solvents is important and many methods exist. In the following a method based on latent roots and latent vectors is given. For the rest of the thesis, the knowledge of the latent roots and vectors of a system is assumed, so a method based on this knowledge was necessary.

2.6.2 Theorem

The main result is summarized in the following theorem. Let $A(\lambda)$ be an r^{th} degree m^{th} order monic matrix polynomial.

Theorem 2.6.1. *If $A(\lambda)$ has m latent roots $\{\lambda_1\lambda_2\cdots\lambda_m\}$ and m corresponding right latent (column) vectors $\{v_1v_2\cdots v_m\}$ and m corresponding left latent (row) vectors $\{w_1w_2\cdots w_m\}$, both of dimension m, then a right solvent R of dimension $m \times m$ is:*

$$R = \begin{pmatrix} v_1 & \cdots & v_m \end{pmatrix} \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_m \end{pmatrix} \begin{pmatrix} v_1 & \cdots & v_m \end{pmatrix}^{-1} \tag{2.6.1}$$

And a left solvent L of dimension $m \times m$ is:

$$L = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix}^{-1} \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_m \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \tag{2.6.2}$$

2.6.3 Proof

Construction of right solvents:

Suppose that the set $\{\lambda_1, \lambda_2, \cdots, \lambda_m\}$ of m latent roots of $A(\lambda)$ has a linearly independent set of corresponding right latent vectors $\{v_1, v_2, \cdots, v_m\}$.

Let $V = (v_1, v_2, \cdots, v_m)$ be the $m \times m$ matrix whose columns are the linearly independent right latent vectors and $V^{-1} = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix}$ be its inverse.

It follows that $\sum_{i=1}^{m} v_i w_i = I_m$ and $w_i v_j = \delta_{ij}$ for $i, j = 1, 2, \cdots, m$, this is verified from $VV^{-1} = V^{-1}V = I_m$.

In the following we will show that the $m \times m$ matrix $R = V\Lambda V^{-1}$ where $\Lambda = diag\{\lambda_1, \lambda_2, \cdots, \lambda_m\}$ is a right solvent of $A(\lambda)$.

Let

$$A_R(R) = R^r + A_{r-1}R^{r-1} + \cdots + A_1 R + A_0$$

hence $A_R(R)$ can be rewritten as :

$$A_R(R) = V\Lambda^r V^{-1} + A_{r-1}V\Lambda^{r-1}V^{-1} + \cdots + A_1 V\Lambda V^{-1} + A_0$$

It can easily verified that

$$V\Lambda^l V^{-1} = \sum_{i=1}^{m} \lambda^l v_i w_i \ \ l = 1, .., r$$

which gives

$$A_R(R) = \sum_{i=1}^{m} \lambda^r v_i w_i + A_{r-1} \sum_{i=1}^{m} \lambda^{r-1} v_i w_i + \cdots + A_1 \sum_{i=1}^{m} \lambda v_i w_i + A_0$$

or

$A_R(R) = [\lambda_1^r v_1 + A_{r-1}\lambda_1^{r-1} v_1 + \cdots + A_1\lambda_1 v_1]w_1 + [\lambda_2^r v_2 + A_{r-1}\lambda_2^{r-1} v_2 + \cdots + A_1\lambda_2 v_2]w_2 + \cdots + [\lambda_m^r v_m + A_{r-1}\lambda_m^{r-1} v_m + \cdots + A_1\lambda_m v_m]w_m + A_0$

Since $\{\lambda_i, v_i\}$ is a latent pair it follows that

$$\lambda_i^r v_i + A_{r-1}\lambda_i^{r-1} v_i + \cdots + A_1\lambda_i v_i + A_0 v_i = 0_m$$

or

$$\lambda_i^r v_i + A_{r-1}\lambda_i^{r-1} v_i + \cdots + A_1\lambda_i v_i = -A_0 v_i$$

leading to

$$A_R(R) = -A_0 v_1 w_1 - A_0 v_2 w_2 - \cdots - A_0 v_m w_m + A_0 = -A_0 I_m + A_0 = 0_m$$

Hence showing that a matrix R constructed from a set of m latent roots and a corresponding set of m linearly independent right latent vectors is a right solvent of the matrix polynomial $A(\lambda)$.

It can easily be verified that the right latent vector v_i of $A(\lambda)$ associated with the latent root λ_i is a right eigenvector of the right solvent R. Since λ_i is an eigenvalue of R, it follows that the right eigenstructure of R captures a part of the right latent structure of $A(\lambda)$.

Construction of left solvents:

In a similar manner, we will establish that a left solvent $L \in \Re^{m\times m}$ can be constructed from a set $\{\lambda_1, \lambda_2, \cdots, \lambda_m\}$of m latent roots and a corresponding set of m linearly independent left latent (row) vectors $\{w_1, w_2, \cdots, w_m\}$ as $L = W^{-1}\Lambda W$ where $W = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix}$ and $\Lambda = diag(\lambda_1, \lambda_2, \cdots, \lambda_m)$. Let $W^{-1} = (v_1, v_2, \cdots, v_m)$ which implies $\sum_{i=1}^{m} v_i w_i = I_m$ and $w_i v_j = \delta_{ij}$ for $i, j = 1, .., m$.

Evaluating

$$A_L(L) = L^r + L^{r-1}A_{r-1} + \cdots + LA_1 + A_0$$

leads to

$$A_L(L) = W^{-1}\Lambda^r W + W^{-1}\Lambda^{r-1}WA_{r-1} + \cdots + W^{-1}\Lambda WA_1 + A_0$$

or

$$A_L(L) = [\sum_{i=1}^{m} \lambda^r v_i w_i] + [\sum_{i=1}^{m} \lambda^{r-1} v_i w_i]A_{r-1} + \cdots + [\sum_{i=1}^{m} \lambda v_i w_i]A_1 + A_0$$

or

$A_L(L) = v_1[\lambda_1^r w_1 + \lambda_1^{r-1} w_1 A_{r-1} + \cdots + \lambda_1 w_1 A_1] + v_2[\lambda_2^r w_2 + \lambda_2^{r-1} w_2 A_{r-1} + \cdots + \lambda_2 w_2 A_1] + \cdots + v_m[\lambda_m^r w_m + \lambda_m^{r-1} w_m A_{r-1} + \cdots + \lambda_m w_m A_1] + A_0$

From $w_i A(\lambda_i) = 0$ for $i = 1..m$ it follows that:

$$\lambda_i^r w_i + \lambda_i^{r-1} w_i A_{r-1} + \cdots + \lambda_i w_i A_1 = -w_i A_0$$

then

$$A_L(L) = -v_1 w_1 A_0 - v_2 w_2 A_0 - \cdots - v_m w_m A_0 + A_0 = 0_m$$

indicating that L is indeed a left solvent of $A(\lambda)$.

Here again, we can verify that the left latent vector of $A(\lambda)$ associated with the latent root λ_i is a left eigenvector of the left block root L. Since λ_i is an eigenvalue of L, it follows that the left eigenstructure of L captures the left latent structure of $A(\lambda)$.

2.6.4 Illustrative example

Consider

$$A(\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \lambda^3 + \begin{pmatrix} 0 & 1 \\ 0 & 5 \end{pmatrix} \lambda^2 + \begin{pmatrix} -1 & 5 \\ 0 & 6 \end{pmatrix} \lambda + \begin{pmatrix} 0 & 4 \\ 0 & 0 \end{pmatrix}$$

where the latent roots are $\{0, 1, -1, -2, -3\}$ with 0 being a double root.

The right latent vectors corresponding to these latent roots are
$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ -3 \end{pmatrix} \begin{pmatrix} 1 \\ -12 \end{pmatrix} \right\}$ respectively.

Each right block root being a 2×2 matrix, we form right solvents by pairing latent roots with corresponding linearly independent right latent vectors such as:

{0, -2}, {0, -3}, {1, -2}, {1, -3}, {-1, -2}, {-1, -3} or {-2, -3}.

The right solvent, say involving the pair {1, -2}, is then:

$$R = \begin{pmatrix} 1 & 1 \\ 0 & -3 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -2 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & -3 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 1 \\ 0 & -2 \end{pmatrix}$$

It can be checked that:$A(R) = 0_2$.

We cannot have in this case a complete set of right solvents since out of the seven possible right solvents, we cannot obtain three disjoint spectra covering the whole spectrum of $A(\lambda)$.

Left latent vectors associated with the latent roots are

$\left\{ \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} -6 & 5 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} \right\}$

The left solvent involving, say the latent root {1, -2}, is then:

$$L = \begin{pmatrix} -6 & 5 \\ 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 \\ 0 & -2 \end{pmatrix} \begin{pmatrix} -6 & 5 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -2.5 \\ 0 & -2 \end{pmatrix}$$

2.7 Conclusion

Matrix polynomials are important in the control of multivariable systems described in matrix fraction. In this case the behavior of the system will be given by the latent structure of the denominator and control can be designed using the solvents. We showed that solvents or block roots of a matrix polynomial can be constructed from latent roots and latent vectors. Conditions of their existence are discussed.

The importance of latent structure and block roots (solvents) of matrix polynomials will be shown in the next chapter where the link between the latent structure of a system described by

matrix fraction description and the eigenstructure of its corresponding state space description is established.

Chapter Bibliography

[1] K. Hariche and E. D. Denman, "On Solvents and Lagrange Interpolating Lambda-matrices", App. Math. And Comp., Vol. 25, No.4 , pp. 321-332, 1988.

[2] I. Bras and T.P. de Lima, "A Spectral Approach to Polynomial Matrix Solvents", Appl. Math. Lett., Vol. 9, No.4, pp. 27-33, 1996.

[3] J.E. Dennis, J.F. Traub and R.P. Weber, "The algebraic theory of matrix polynomials", SIAM J. Num. Anal., Vol. 13, pp. 831-845, 1976.

[4] I. Gohberg, P. Lancaster and L. Rodman, Matrix polynomials, Academic Press, 1982.

[5] I. Gohberg, P. Lancaster and L. Rodman, "Spectral analysis of matrix polynomials: I. Canonical forms and divisors", Lin. Alg. App., Vol. 20, pp. 1-44, 1978.

[6] I. Gohberg, P. Lancaster and L. Rodman, "Spectral analysis of matrix polynomials: II. The resolvent form and spectral divisors", Lin. Alg. App., Vol. 21, pp. 65-88, 1978.

[7] E. D. Denman, "Matrix polynomials, roots, and spectral factors", App. Math. Comput., Vol. 3, pp. 359-368, 1977.

[8] L. S. Shieh and Y.T. Tsay, "Transformation of solvents and spectral factors of matrix polynomials and their applications", Int. J. Contr., Vol. 34, pp. 813-823, 1981.

[9] J. S. H. Tsai, L. S. Shieh and T. T. C. Shen, "Block power method for computing solvents and spectral factors of matrix polynomials", Comput. Math. Applic., Vol. 16, No. 9, pp. 683-699, 1988.

[10] L. S. Shieh and Y. T. Tsay, "Transformation of a class of multivariable control systems to block companion forms", IEEE Trans. Auto. Contr., Vol AC-27, No. 1, pp. 199-203, 1982.

[11] L. S. Shieh, F. R. Chang and B. C. Mcinnis, "The block partial fraction expansion of a matrix fraction description with repeated block poles", IEEE Trans. Auto. Contr., Vol. AC-31, No. 3, pp. 236-239, 1986.

[12] J. E. Dennis, J. F. Traub and R. P. Weber, "Algorithms for solvents of matrix polynomials", Siam J. Num. Anal., Vol. 15, No.3, pp. 523-533, 1978.

[13] V. N. Kublanovskaya, "Methods and Algorithms of Solving Spectral Problems for Polynomial and Rational Matrices", J. Math. Sci., Vol. 96, No. 3, pp. 3085-3287, 1999.

[14] A. Dahimene, "Algorithms for the factorization of matrix polynomials", Magister thesis, INELEC, Boumerdes, 1992.

[15] P. Lancaster, Lambda-Matrices and Vibrating Systems, Pergamon Press, NewYork , 1966.

[16] I. Gohberg, M.A. Kaashoek and L. Rodman, "Spectral Analysis of families of operator polynomial and generalized Vandermonde matrix. I. finite dimensional case", topics in functional analysis, Adv. in Math., Supplementary Studies, Vol.3, pp. 91-128, 1978.

[17] F. Kuo and G. S. Chen, "A recursive algorithm for coprime fractions and Diophantine equations", IEEE Trans. Autom. Contr., Vol. AC-34, No. 12, pp. 1276-1279, 1989.

[18] K. Hariche, Interpolation theory in the structural analysis of Lambda-matrices, PhD thesis, University of Houston, USA, 1987.

[19] K. Hariche and E. D. Denman, "Interpolation theory and Lambda-matrices", J. Math. Anal. Appl., No. 143, pp. 530-547, 1989.

[20] M. Yaici and K. Hariche, "On solvents of matrix polynomials", Int. J. Mod, Optim., Vol.4, No. 4 , pp. 273-277, 2014.

Chapter 3

State space description and matrix fraction description

3.1 Introduction

The analysis and synthesis of complex engineering systems always start by building up models which realistically describe their behaviour. Because of different used analytical methods, we may set up different mathematical equations to describe the same system.

The transfer function that describes only the terminal property of a system is called the input-output description of the system. The set of differential equations that describes the internal as well as the terminal behaviour is called state variable description of the system [1].

Multiple inputs multiple outputs dynamical systems can be described with many representations: State space equations, transfer function, or matrix fraction description.

A matrix fraction description (MFD) is a decomposition of a given rational transfer function matrix into two matrices $D(s)$ and $N(s)$. The entries of these matrices are polynomials of s, so called matrix polynomials [2].

In this chapter we will deal with state space description and some needed notions will be recalled as eigenstructure and block companion transformations. The Matrix fraction description will also be presented with methods to transform an SSD to an MFD. Then the relationship between latent structure and eigenstructure is established and as a consequent result a method to solve the polynomial eigen problem is proposed.

3.2 Literature review

The process of controlling a dynamic system starts with its modelling in terms of mathematical equations [3]. In particular, in the case of linear time invariant systems, we have a set of p differential equations of the type:

$$\begin{array}{l} a_{1d_1}y_1^{(d_1)}(t) + \cdots + a_{10}y_1(t) + a_1 + \cdots + a_{pd_p}y_p^{(d_p)}(t) + \cdots + a_{p0}y_p(t) + a_p = \\ b_{1f_1}u_1^{(f_1)}(t) + \cdots + b_{10}u_1(t) + b_1 + \cdots + b_{mf_m}u_m^{(f_m)}(t) + \cdots + b_{m0}u_m(t) + b_m \end{array} \tag{3.2.1}$$

where $\alpha^{(q)}(t)$ represents the q^{th} derivative of the scalar signal $\alpha(t)$ versus time.

The parameters a_{ij} and b_{kl} for $i = 1..p$, $j = 0..d_i$, $k = 1..m$ and $l = 0..f_k$ are constants. By convention, all variables on which there is a control that will be used to control the system are called the inputs $u_i(t)$, $i = 1..m$. Moreover, all the variables which can be observed on the effect of the command are called outputs $y_i(t)$, $i = 1..p$.

There are different approaches to interact with this mathematical model. With each approach, the model is shown or reformulated in a particular way in which the different properties of the system are more or less revealed. One can for example write each equation of type 3.2.1 as a set of differential equations of order 1 with the inclusion of state variables [4]. After algebraic manipulations, the system model can be written as follows:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Eu(t) \end{cases} \tag{3.2.2}$$

The vector $x(t) \in \Re^n$ is the state variable vector and thus is used to link the vector of inputs $u(t) \in \Re^m$ and the vector of outputs $y(t) \in \Re^p$. The matrices $A \in \Re^{n\times n}$, $B \in \Re^{n\times m}$, $C \in \Re^{p\times n}$ and $E \in \Re^{p\times m}$ are constants. The representation 3.2.2 is used by the state-space approach [4, 5].

In state space model, the modal decomposition of the state matrix into its eigenstructure is very useful as it defines the stability and the dynamic behavior of a linear multi-variable system.

The importance of the state space model comes from its causality (strictly causal if $E = 0$), the fact that it is convenient for controller/filter/predictor design, and it is also convenient for computation [2].

We note that by applying the Laplace transform on the representation 3.2.2 $x(t)$ can be eliminated and obtain the transfer function $T(s)$ of the system [4] assuming zero initial conditions.

$$y(s) = [C(sI - A)^{-1}B + E]u(s) = T(s)u(s)$$

Another way of rewriting the set of p equations of type (3.2.1) representing the dynamic system is the matrix form. By grouping all inputs and outputs in the corresponding vectors u and y we get:

$$D(s)y(s) = N(s)u(s) \tag{3.2.3}$$

The matrices $D(s)$ and $N(s)$ are polynomial matrices in the variable s, and of dimensions $p \times p$ and $p \times m$ respectively. These polynomial matrices can be rewritten as matrix polynomials as defined in the precedent chapter (section 2.3.1).

We recall that in the continuous case, the variable s is the Laplace operator. If the system is in discrete time, then the variable is the shift operator.

Since the transfer function of the system is unique [4], it is checked that:

$$T(s) = D^{-1}(s)N(s) = C(sI - A)^{-1}B + E \tag{3.2.4}$$

Representation 3.2.3 is used by the polynomial approach [6] or by the behavioral approach [7].

Each representation has its strengths and weaknesses, and involves a different way of looking at the same system. Therefore every approach to control systems also has its peculiarities and uses its own methods and mathematical tools. For some problems, the state space approach will be more effective, for other problems the polynomial approach or behavioral approach will be more appropriate.

Analysis and synthesis based on the matrix fraction description (MFD) of multivariable systems have received a great deal of attention for the past few decades (Callier and Desoer 1982 [8]; Chen 1984 [9]; Kailath 1980 [4]; Wolovich 1974 [10]).

If a linear system is represented with matrix polynomials in the form (3.2.3), then the locations of its zeros and poles, which determine the dynamics of the system, are contained in the eigenstructure (latent structure) of matrix polynomials $N(s)$ and $D(s)$ [4].

In the problem of decoupling linear systems for example, infinite structural indices of a suitable polynomial matrix are needed to determine if the system is decouplable [11], and also to determine the structure that could have the decoupled closed loop system [3]. Model

matching [12] and disturbance rejection [13] are other examples of control problems where the eigenstructure of polynomial matrices plays a fundamental role. The zero structure of a polynomial matrix is also instrumental to its spectral factorization which has applications in several optimal and robust control problems [14, 15].

Many papers have considered using solvents (block roots of $D(s)$) for solving some linear algebra problems or control problems such as: block partial fraction expansion of a matrix fraction description (MFD) with single and repeated poles [16, 17], cascade decomposition and realization of multivariable systems via block-pole and block-zero placement [18], state-feedback decomposition of multivariable systems via block-pole placement [19].

The link between state space representation and matrix polynomials is well established in the book of Rosenbrock [5]. Latent structure is, in some ways, analogous to eigenstructure but its use and importance has still to be explored particularly in control theory. It is known and has been verified that latent values of the MFD of a system and eigenvalues of the SSD of the same system are the same, but the link between eigenvectors and latent vectors has not yet been elaborated to my knowledge.

3.3 State space description

3.3.1 Introduction

The state space description of a system provides a complete picture of the system structure showing how all the internal variables x_i $i = 1..n$ interact with one another, how the inputs $u_k(t)$ $k = 1..m$ affect the system states $x_i(t)$, and how the outputs $y_j(t)$ $j = 1..p$ are obtained from various combinations of the state variables $x_i(t)$ and the inputs $u_k(t)$.

The state variable representation of linear systems enables a large number of interconnected simultaneous differential equations to be formed into a single matrix equation. If, for a given system, enough information is known to permit the calculation of the unforced system output for all future time, then the system's state is known [20]. The state variable approach assigns a set of independent variables to represent the system state, and considers the variation of the state with time as the system response.

A linear state model is formed by a set of first order linear differential equations with

constant coefficients and a set of linear equations.

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Eu(t) \end{cases} \tag{3.3.1}$$

Where $x(t)$ is the system state vector:$x(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{pmatrix}$ and $x_i(t)$ $i = 1..n$ are the system state variables

u(t) is the system input vector: $u(t) = \begin{pmatrix} u_1(t) \\ \vdots \\ u_m(t) \end{pmatrix}$

and $y(t)$ is the system output vector:$y(t) = \begin{pmatrix} y_1(t) \\ \vdots \\ y_p(t) \end{pmatrix}$

and (A,B,C,E) are real, constant, respectively $n \times n$, $n \times m$, $p \times n$ state system matrix, input matrix, output matrix, and direct transmission matrix.

The first part of the system of equations describes the dynamic part of the behaviour of a system and how the initial system state $x(0)$ and system input $u(t)$ will determine the new state $x(t)$. The second part describes how the system state $x(t)$ and the system input $u(t)$ will instantly determine the system output $y(t)$.

If $p = 1$ and $m = 1$ then the system is single input, single output (SISO) and if $m > 1$, $p > 1$ then it is multiple input, multiple output (MIMO).

Definition 3.3.1. The state of a system at time t_0 is the amount of information at t_0 that, together with $u_{[t_0,\infty)}$ determines uniquely the behavior of the system for all $t \geq t_0$ [9].

System analysis generally consists of two parts: quantitative and qualitative. In the quantitative study, it is dealt with the search for the exact response of the system to certain input and initial conditions. In qualitative study, the general properties of a system are sought.

For more convenience we suppose that the input-output matrix is null in the rest of this thesis.

3.3.2 Eigenstructure

Definition 3.3.2. The eigenstructure of the system given by equation 3.3.1 consists in the set of n eigenvalues and its corresponding eigenvectors of the state matrix A:

$$\begin{aligned} A\tilde{v}_i &= \lambda_i \tilde{v}_i \ , \ i = 1..n \\ \tilde{w}_i A &= \lambda_j \tilde{w}_j \ , \ j = 1..n \end{aligned} \tag{3.3.2}$$

where λ_i is the i^{th} eigenvalue of A, $\tilde{v}_i \in \Re^n \times 1$ is the associated right eigenvector and $\tilde{w}_j \in \Re^{1\times n}$ is the associated left eigenvector.

If $\Lambda = diag\{\lambda_1 \ \lambda_2 \ \cdots \ \lambda_n\}$ is the spectral matrix composed of the set of eigenvalues of A, $\tilde{V}$ the modal matrix composed of the right eigenvectors, and $\tilde{W}$ the modal matrix composed of the left eigenvectors then we have the following equations [21]:

$$\begin{cases} A\tilde{V} = \tilde{V}\Lambda \\ \tilde{W}A = \Lambda\tilde{W} \\ \tilde{W}^T = \tilde{V}^{-1} \\ \Lambda = \tilde{W}^T A\tilde{V} \end{cases} \tag{3.3.3}$$

3.3.3 Time response

We consider a standard state space system as described by equation 3.3.1. The standard result is the time response composed of a zero-input response and the zero-state response:

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau}Bu(\tau)d\tau \tag{3.3.4}$$

The vector $x(t)$ defines the trajectory of the system time response in state space. The matrix e^{At} is called the state transition matrix.

If the state vector is considered as describing a direction in a space (or a hyperspace), then the state transition matrix describes the trajectory of the state vector through the state space. The state transition matrix, although useful for determining a time response, is not particularly informative. It is not clear how changing the system matrix A will affect the time response of the resulting system.

If Λ, $\tilde{V}$ and $\tilde{W}$ are as defined previously, then the time response will be:

$$x(t) = \tilde{V}e^{\Lambda}t\tilde{W}^T x(0) + \tilde{V}\int_0^t e^{\Lambda(t-\tau)}\tilde{W}^T Bu(\tau)d\tau \tag{3.3.5}$$

Where $e^{\Lambda}t = diag\{e^{\lambda_1 t}, \cdots, e^{\lambda_n t}\}$.

So the state transition matrix has been decoupled into its n constituent modes. Theses modes are mapped into the state space by eigenvectors.

Hence the eigenvalues of A correspond directly to the system poles, if no pole-zero cancellation occurs. Secondly, the matrices $\tilde{V}$ and $\tilde{W}$ of the right and left eigenvectors respectively have a physical interpretation as well. The initial state vector $x(0)$ acts to excite the system modes via $\tilde{W}$, and the modal response manifests itself in the time-varying state vector $x(t)$ via $\tilde{V}$. Hence, for a given mode, the associated left eigenvector describes the way in which state disturbances excite the system modes, and the associated right eigenvector describes the coupling of the system modes into the state vector [20].

Zero-Input Response:

We suppose no forcing function ($u(t) = 0$). The time response will be:

$$x(t) = \tilde{V}e^{\Lambda t}\tilde{W}^T x(0) = \sum_{i=1}^{n} \tilde{v}_i e^{\lambda_i t}\tilde{w}_i^T x(0) \tag{3.3.6}$$

The value $\tilde{w}_i^T x(0)$ is a scalar quantity which magnitude is given by $\tilde{w}_i$ and direction is given by $\tilde{v}_i$. Thus if the initial condition excitation is orthogonal to one of the vectors then the related mode is not excited in the states and hence in the outputs.

So the role of the eigenstructure in the zero-input response is the following:

- Eigenvalues: determine the decay or growth rate of each natural mode $e^{\lambda_i t}$
- Right eigenvectors: fix the shape of the response of each natural mode.
- Left eigenvectors: determine the proportion of each natural mode that is present in the time response when excited by a forcing function.

For the zero state response: the value of $\tilde{W}^T B$ shows the extent to which a particular input excites certain modes.

Remark 3.3.1. These facts underpin the idea of Eigenstructure Assignment: If a control system can be designed that affects the eigenstructure of A in a predictable manner, then the system poles may be placed and the coupling between the system modes and the states may be influenced [20].

3.3.4 Controllability and Observability

Controllability and observability have an important role in both theoretical and practical aspects of modern control. Before the control system designer can apply a particular design method, it is necessary to establish to what extent the available inputs influence the system behaviour, and to what extent the available outputs indicates the system behaviour. The extent to which the inputs influence the system is defined as the controllability of the system, and the extent to which the outputs shows the system behaviour is defined as the observability of the system [1]

Controllability of linear time invariant system:

Definition 3.3.3. For the system given in 3.3.1, if there exists an input $u[0,t)$ which transfers the initial state $x(0) = x_0$ to the zero state $x(t_1) = 0$ in a finite time t_1, the state x_0 is said to be controllable. If all initial states are controllable then the system is said fully controllable [22].

If the system is controllable, i.e. there exists an input to make $x(t_1) = x_1 = 0$ at a finite time $t = t_1$ then after pre-multiplying the time response, given by equation 3.3.4, by e^{-At} the solution yield:

$$x_0 = \int_0^{t_1} e^{-A\tau} B u(\tau) d\tau \tag{3.3.7}$$

Therefore any controllable state satisfies 3.3.7 and for a fully controllable system every state $x_0 \in \Re^n$ satisfies $t_1 > 0$ and $u_{[0,t_1)}$. It is found that full controllability of a system depends on matrix A and B but independent of the output matrix C.

Theorem 3.3.1. *The n dimensional linear time invariant state equation is controllable if and only if any of the following equivalent conditions is satisfied:*

i) all rows of $[e^{-At}B]$ are linearly independent on $[0,\infty)$ over the field of complex numbers.

ii) $W(0,t_1) = \int_0^{t_1} e^{-At} B B^T e^{-A^T t} dt$ is non-singular for any $t_1 > 0$.

iii) the $n \times nm$ controllability matrix has full rank.

$$\mathcal{C}n = \begin{pmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{pmatrix} \tag{3.3.8}$$

Proof: see [9].

Observability of linear time invariant system:

Dual to controllability, observability studies the possibility of estimating the state from the output. If a dynamical system of equations is observable then all the modes of the system can be observed and measured from the outputs.

Definition 3.3.4. When using the input of the system measured from time zero to time t_1, if the initial state $x(0) = x_0$ is uniquely determined then x_0 is said to be observable when the input is assumed to be completely known. When all states are observable the system is said to be fully observable [22].

The output of the system is determined by:

$$y(t) = Ce^{At}x_0 + \int_0^t Ce^{A(t-\tau)}Bu(\tau)d\tau + Eu(t) \tag{3.3.9}$$

Theorem 3.3.2. *The n dimensional linear time invariant dynamical equation is observable iff any of the following equivalent conditions is satisfied:*

i) all columns of $[Ce^{At}]$ *are linearly independent on* $[0, \infty)$ *over the field of complex numbers.*

ii) $W(0, t_1) = \int_0^{t_1} e^{A^T t}C^T Ce^{At}dt$ *is non-singular for any* $t_1 > 0$.

iii) the $np \times n$ *observability matrix has full rank:*

$$\mathcal{O} = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix} \tag{3.3.10}$$

Proof: see [9]

Controllability and observability are often given as preconditions for forming multivariable control systems. If a system has uncontrollable modes, either these modes are important to the system response or they are not; if they are not, a reduced model may be formed that does not include these dynamics. Similarly, if unobservable modes are present then further sensors should be added to the system to allow detection of these modes, lest they become unstable under feedback.

Interestingly, it is possible for the controllability or observability test matrices to be full rank but ill-conditioned; this implies either a nearly-uncontrollable mode, requiring large control efforts to affect it, or a nearly-unobservable mode, whose effect is hard to detect in the output vector. For this reason Mehrmann and Xu [23] define the concept of 'distance to controllability' and use it as a system analysis tool [20].

3.3.5 Block companion forms

The system as described in 3.3.1 can be put in different forms (controller form, observer form, diagonal form, etc.). We are interested in block controller form and block observer form, which will be needed in the rest of the thesis. For other canonical forms, their use and their utility see [4, 2].

Consider the n-dimensional linear time-invariant multivariable dynamical equations as in 3.3.1, and we suppose E equal to the null matrix:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{3.3.11}$$

Block controllability:

Definition 3.3.5. The system (3.3.11) is block controllable iff [24]:

i) the number $n/m = \mu$ is an integer and

ii) the controllability matrix of degree μ has full rank n:

$$\mathcal{C}n_\mu = \begin{pmatrix} B & AB & \cdots & A^{\mu-1}B \end{pmatrix} \tag{3.3.12}$$

Remark 3.3.2. μ is the controllability index.

Block controller form:

If the system (3.3.11) is block controllable of index μ then it can be transformed to block controller form:

Let $x_c = T_c x$ where:

$$T_c = \begin{pmatrix} T_{c1} \\ T_{c1}A \\ \vdots \\ T_{c1}A^{\mu-1} \end{pmatrix} \text{ and } T_{c1} = \begin{pmatrix} 0_m & \cdots & 0_m & I_m \end{pmatrix} \mathcal{C}n_\mu^{-1} \tag{3.3.13}$$

In the new coordinates, the state space equations will become:

$$\begin{cases} \dot{x}_c = A_c x_c + B_c u \\ y = C_c x_c \end{cases} \text{ where } \begin{cases} A_c = T_c A T_c^{-1} \\ B_c = T_c B \\ C_c = C T_c^{-1} \end{cases}$$

or

$$A_c = \begin{pmatrix} 0_m & I_m & 0_m & \cdots & 0_m \\ 0_m & 0_m & I_m & \cdots & 0_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \cdots & I_m \\ -A_0 & -A_1 & -A_2 & \cdots & -A_{\mu-1} \end{pmatrix}; B_c = \begin{pmatrix} 0_m \\ 0_m \\ \vdots \\ 0_m \\ I_m \end{pmatrix}; C_c = \begin{pmatrix} C_0 & C_1 & \cdots & C_{\mu-1} \end{pmatrix} \tag{3.3.14}$$

Block observability:

Definition 3.3.6. The system as in equation 3.3.11 is block observable iff [24]:

i) the number $n/p = \nu$ is an integer and

ii) the observability matrix of degree ν has full rank

$$\mathcal{O}_\nu = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{\nu-1} \end{pmatrix} \tag{3.3.15}$$

Remark 3.3.3. ν is the observability index

Block Observer form:

If the system is block observable of index ν then it can be transformed to block observer form.

Let us make a change of coordinates: $x_o = T_o^{-1} x$ where

$$T_o = \begin{pmatrix} T_{o1} & AT_{o1} & \cdots & A^{\nu-1} T_{o1} \end{pmatrix} \text{ and } T_{o1} = \mathcal{O}_\nu^{-1} \begin{pmatrix} 0_p \\ \vdots \\ 0_p \\ I_p \end{pmatrix} \tag{3.3.16}$$

In the new coordinates we have:

$$\begin{cases} \dot{x}_o = A_o x_o + B_o u \\ y = C_o x_o \end{cases} \text{ where } \begin{cases} A_o = T_o^{-1} A T_o \\ B_o = T_o^{-1} B \\ C_o = C T_o \end{cases}$$

or

$$A_o = \begin{pmatrix} 0_p & 0_p & \cdots & 0_p & -A_0 \\ I_p & 0_p & \cdots & 0_p & -A_1 \\ 0_p & I_p & \cdots & 0_p & -A_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_p & 0_p & \cdots & I_p & -A_{\nu-1} \end{pmatrix}; B_o = \begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ \vdots \\ B_{\nu-1} \end{pmatrix}; C_o = \begin{pmatrix} 0_p & \cdots & 0_p & I_p \end{pmatrix} \quad (3.3.17)$$

Block diagonal form:

Once we have the block controller (observer) form, we can transform it into a block diagonal from using the similarity transformation [17]:

$$x_d = V_R^{-1} x_c \text{ or } x_d = V_L^{-1} x_o \quad (3.3.18)$$

where V_R (V_L) is a "right" (left) block Vandermonde matrix (defined in chapter 2 section 2.3.4). The conditions are, of course, block controllability and block observability.

In the new coordinates, the system will be:

$$\begin{cases} \dot{x}_d = A_d x_d + B_d u \\ y = C_d x_x \end{cases} \text{ where } \begin{cases} A_d = V_R^{-1} A_c V_R \\ B_d = V_R^{-1} B_c \\ C_d = C_c V_R \end{cases} \text{ or } \begin{cases} A_d = V_L A_o V_L^{-1} \\ B_d = V_L B_o \\ C_d = C_o V_L^{-1} \end{cases} \quad (3.3.19)$$

which gives:

$$A_d = \begin{pmatrix} R_1 & 0_m & \cdots & 0_m \\ 0_m & R_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0_m \\ 0_m & \cdots & 0_m & R_\mu \end{pmatrix}; B_d = \begin{pmatrix} B_{d1} \\ \vdots \\ B_{d\mu} \end{pmatrix}; C_d = \begin{pmatrix} I_p & \cdots & I_p \end{pmatrix} \quad (3.3.20)$$

or

$$A_d = \begin{pmatrix} L_1 & 0_p & \cdots & 0_p \\ 0_p & L_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0_p \\ 0_p & \cdots & 0_p & L_\nu \end{pmatrix}; B_d = \begin{pmatrix} I_m \\ \vdots \\ I_m \end{pmatrix}; C_d = \begin{pmatrix} C_{d1} & \cdots & C_{d\nu} \end{pmatrix} \quad (3.3.21)$$

Remark 3.3.4. .

- The set $\{R_1, \cdots, R_\mu\}$ or $\{L_1 \cdots, L_\nu\}$ is a complete set of solvents of the corresponding system described in MFD.
- As it can be seen, this is a block decoupled system. Thus it can be decoupled into μ or ν independent subsystems.
- The eigenvalues of A_d are the union of the sets of eigenvalues of each Solvent.

3.4 Matrix fraction description

3.4.1 Generation of a MFD

Matrix fraction description provides a natural generalization of the scalar rational transfer function, though in multivariable case we have to distinguish between right and left descriptions. In linear time-invariant single-input single-output system, the transfer function is a ratio of two scalar polynomials. The system modelling of physical, linear time-invariant multi-input multi-output control systems, results in high degree coupled differential equations or an r^{th} degree m^{th} order differential equation of the form:

$$X^{(r)}(t) + D_{r-1}X^{(r-1)}(t) + \cdots + D_1X^{(1)}(t) + D_0X(t) = u(t) \tag{3.4.1}$$

Where D_i are $m \times m$ real matrices, $X^{(i)}$ of dimension $m \times 1$ represents the i^{th} derivative of the vector $X(t)$ and $u(t)$ of dimension $m \times 1$ being the input vector. The output $y(t)$ of dimension $p \times 1$ is generally given by a differential equation in the form:

$$y(t) = N_{r-1}X^{(r-1)}(t) + N_{r-2}X^{(r-2)}(t) + \cdots + N_1X^{(1)}(t) + N_0X(t) \tag{3.4.2}$$

where N_i are $p \times m$ real matrices.

The Laplace transformation with zero initial conditions results in:

$$s^rX(s) + D_{r-1}s^{r-1}X(s) + \cdots + D_0X(s) = U(s)$$

and

$$Y(s) = N_{r-1}s^{r-1}X(s) + N_{r-2}s^{r-2}X(s) + \cdots + N_0X(s)$$

which yields:

$$Y(s) = [N_{r-1}s^{r-1} + N_{r-2}s^{r-2} + \cdots + N_0][I_ms^r + D_{r-1}s^{r-1} + \cdots + D_0]^{-1}U(s) \tag{3.4.3}$$

So the $p \times m$ transfer function matrix is:

$$T(s) = N_R(s)D_R^{-1}(s) \tag{3.4.4}$$

Where N_R and D_R are $p \times m$ and $m \times m$ numerator and denominator matrix polynomials defined by:

$$N_R(s) = N_{r-1}s^{r-1} + N_{r-2}s^{r-2} + \cdots + N_1s + N_0 \tag{3.4.5}$$

and

$$D_R(s) = I_m s^r + D_{r-1} s^{r-1} + \cdots + D_1 s + D_0 \tag{3.4.6}$$

The equation 3.4.4 is the right matrix fraction description (RMFD) of the MIMO system defined by equations 3.4.1 and 3.4.2 [4].

An alternative factorization of $T(s)$ is the left matrix fraction description (LMFD) defined by:

$$T(s) = D_L^{-1}(s) N_L(s) \tag{3.4.7}$$

Where D_L is a $p \times p$ denominator matrix polynomial and N_L is a $p \times m$ numerator matrix polynomial.

The latent roots of the denominator are called latent values with their corresponding latent vectors and the solvents will be called block roots (block poles for the denominator and block zeros for the numerator).

Remark 3.4.1. .

- The degree of a MFD $\{D(s),\ N(s)\}$ is defined as the degree of the determinant of $D(s)$, and the characteristic polynomial of $T(s)$ is defined as the determinant of $D(s)$ [2].
- There exists a certain duality between right and left MFD's [4].

3.4.2 Poles and Zeros of systems in MFD

It is known that the poles and zeros of a SISO system are those values of s which cause the transfer function $T(s)$ to equal ∞ or 0, respectively.

The system poles of a MIMO system are those values $\{\lambda\}$ for which $T(\lambda) = \infty$. Such a definition is possible because all the entries in the transfer function matrix share a common denominator [20].

Since the entries of the transfer function matrix do not share a common numerator, such a simple definition will not suffice for multivariable zeros. Instead, we may define any constant z such that $T(z) = 0$ as a blocking zero. The response of a system with a blocking zero z to an input $u(t)e^{zt}$ is zero for any $u(t)$ [25].

However, this definition tells us only when a value of s causes every element in $T(s)$ to become zero. It is also useful to characterize those values of s which cause any element in $T(s)$

to become zero, and hence represent a null in the response of a subset of the input-output paths. Such values are known as transmission zeros, and they may be identified as those values of z for which $rank(T(s)) < min(m, p)$ for a system with m inputs and p outputs.

3.4.3 Block partial fraction expansion

A large-scale multivariable control system described by high degree matrix fraction descriptions (MFD's) is often decomposed into parallel subsystems with low degree MFD's for less sensitivity, good reliability, and simple simulations and designs [16].

In the rest of the sections the MFD of a MIMO system will be noted $G(s)$ which can be in RMFD or LMFD form.

We consider a MIMO system $G(s)$ described in RMFD then $G(s)$ can be written as [19]:

$$G(s) = \sum_{i=1}^{r} G_i (sI_m - L_i)^{-1} \tag{3.4.8}$$

This is a block partial fraction expansion where the L_i are the block poles of $G(s)$ (left solvents of $D(s)$) and the G_i are called the residues of the block partial fraction expansion $G_i = \sum_{j=1}^{r} C_j J_i L_i^{r-j}$ for $i = 1..r$.

The $m \times m$ J_i matrices can be determined from:

$$\begin{pmatrix} J_1 & J_2 & \cdots & J_r \end{pmatrix} = \begin{pmatrix} 0_m & \cdots & 0_m & I_m \end{pmatrix} V_L^{-1} \tag{3.4.9}$$

Where V_L is the left Vandermonde matrix constructed from the full set of left solvents.

By analogy, if we consider a MIMO system $G(s)$ described in LMFD then it can be factorized as follows:

$$G(s) = \sum_{i=1}^{r} (sI_m - R_i)^{-1} H_i \tag{3.4.10}$$

Where the R_i are the block poles of $G(s)$ (right solvents of $D(s)$) and the H_i are called the residues of the block partial fraction expansion $H_i = \sum_{j=1}^{r} R_i^{r-j} J_i' B_j$ for $i = 1..r$ and the matrices K_i' are as follows:

$$\begin{pmatrix} J_1' & J_2' & \cdots & J_r' \end{pmatrix} = V_R^{-1} \begin{pmatrix} 0_p \\ \vdots \\ 0_p \\ I_p \end{pmatrix}$$

Where V_R is the right Vandermonde matrix constructed from the full set of right solvents.

In case of multiple block roots of the denominator, the following equation gives the block partial fraction expansion. We consider a matrix polynomial described in LMFD. As before we can write the block partial fraction expansion as follows:

$$G(s) = \sum_{i=1}^{k} \sum_{j=1}^{m_i} (sI_m - R_i)^{-j} H_{ij} \tag{3.4.11}$$

R_i are the block poles of $G(s)$ and H_{ij} are the matrix residues of the block partial fraction expansion, k is the number of distinct block poles with multiplicity m_i [17].

Remark 3.4.2. Conditions of existence of block roots (solvents) or repeated block roots, and a full set of solvents are given in the previous chapter (see section 2.3.5).

3.4.4 Important properties of MFD

The following properties are important for the notions developed in the rest of the thesis.

Properness:

Definition 3.4.1. A rational matrix $G(s)$ is said to be proper (strictly proper) if $G(\infty)$ is a finite constant (zero) matrix.

If $G(s)$ is described in RMFD or LMFD such that the degree of the numerator is less than the degree of the denominator, then $G(s)$ is strictly proper [4].

Remark 3.4.3. From [[9], p.91], it is stated that a transfer function obtained from a state space description of a system is proper, and if its input-output direct transmission is zero then it is strictly proper.

Row/column reduceness

Definition 3.4.2. A non singular matrix polynomial $D(s)$, as in equation 3.4.6 , is said column reduced if $deg(detD(s)) = \sum_{i=1}^{p} \gamma_{ci}$ where γ_{ci} are the max column degrees of $D(s)$, and it is said row reduced if $deg(detD(s)) = \sum_{i=1}^{p} \gamma_{ri}$ where γ_{ri} are the max row degrees of $D(s)$ [9].

Theorem 3.4.1. *Let $D(s) = D_{hc}H_c(s) + D_{lc}(s)$ where H_c=diag{higher column degrees on s of $D(s)$}, D_{hc} is a constant matrix of coefficients with higher column degrees of $D(s)$ and D_{lc} is the rest of $D(s)$. Then $D(s)$ is said column reduced iff D_{hc} is non singular.*

By duality, let $D(s) = H_r(s)D_{hr} + D_{lr}(s)$ *where:* H_r*=diag{higher row degrees on* s *of* $D(s)$*},* D_{hr} *is a constant matrix of coefficients with higher row degrees of* $D(s)$ *and* D_{lr} *is the rest of* $D(s)$*. Then* $D(s)$ *is said row-reduced iff* D_{hr} *is non singular.*

Proof: See [[9] page 601].

Remark 3.4.4. From the previous theorem, we can deduce that a monic matrixpolynomial is row and column reduced.

Coprimeness

Definition 3.4.3. A system described in RMFD (as in equation 3.4.4) has the following properties:

- $N(s)$ and $D(s)$ are right-coprime if they have only unimodular common right divisors.
- $N(s)$ and $D(s)$ are right coprime if $\begin{pmatrix} N \\ D \end{pmatrix}$ has full rank for any s.

If $N(s)$ and $D(s)$ are right coprime then $G(s)$ is said irreducible.

The same definitions can be applied for left-coprimeness for systems described in LMFD.

Irreducible MFDs

Definition 3.4.4. If the transfer function matrix $G(s) = N(s)D^{-1}(s)$ is irreducible then:

- The poles of $G(s)$ are the roots of the determinant of $D(s)$.
- If $N(s)$ is square and non-singular then its zeros are the roots of the determinant of $N(s)$.
- otherwise the zeros of the non-square $G(s)$ are the frequencies at which the rank of $N(s)$ drops below its normal rank.
- $N(s)$ and $D(s)$ are right (left) coprime (i.e. $D(s)$ and $N(s)$ have no greatest common left (right) divisor).

3.4.5 Canonical forms

Irreducible MFD with no further constraints are not unique. We have to impose extra constraints in order to single out unique MFD. to describe MIMO systems. One way of doing this is by using so called canonical MFD. model types.

Definition 3.4.5. Two MFD.'s $\{D_a(s),N_a(s)\}$ and $\{D_b(s),N_b(s)\}$ are equivalent if they give rise to the same transfer function [2].

Under the condition of non-singularity of the matrix denominator $D(s)$ and aiming some properties, MFD of a multivariable system $\{D(s),\ N(s)\}$ can be transformed into a canonical form. Table 3.1 gives a list of these forms[2]:

Table 3.1: Canonical model types

Canonical form	**Properties**
Hermite	Column proper, irreducible
Echelon	Column proper, row proper, irreducible
Guidorzi	Column proper, row proper, irreducible
Diagonal	Row proper, column proper

Remark 3.4.5. It is sometimes easier to solve a control problem if the MFD of the system is transformed into one of these canonical forms. For more informations see [2, 4, 26, 27].

3.5 Transformation between models

Transformations between the three main descriptions of MIMO (SSD, TF, MFD) systems is possible. Several methods exist to obtain MFDs from SSD and mainly TF [28, 29, 30, 2]. In [30] detailed methods to obtain MFD forms are given. In this section a method to convert SSD into MFD is given. Other methods are given in Appendix D.

3.5.1 Conversion Methods from SSD to MFD

Let a multi-variable continuous dynamic linear system described by the following state space equations:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{3.5.1}$$

with the following dimensions: $A : n \times n$, $B : n \times m$, $C : p \times n$.

To convert this SSD description to a MFD (left or right) description such that:

$$G(s) = N_R(s)D_R^{-1}(s) \text{ or } G(s) = D_L^{-1}(s)N_L(s)$$

where N_R, D_R, N_L and D_L are matrix polynomials, two conversion approaches can be used, using the transformation in a block controller form to get Right MFD or in a block observer form to get left MFD.

3.5.2 Block controller form

If the system described as in equation 3.5.1 is block controllable with index μ (as defined in section 3.3.5) the system is transformed into a block controller form as follows:

$$A_c = \begin{pmatrix} 0_m & I_m & 0_m & \cdots & 0_m \\ 0_m & 0_m & I_m & \cdots & 0_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \cdots & I_m \\ -A_0 & -A_1 & -A_2 & \cdots & -A_{\mu-1} \end{pmatrix}; B_c = \begin{pmatrix} 0_m \\ 0_m \\ \vdots \\ 0_m \\ I_m \end{pmatrix}; C_c = \begin{pmatrix} C_0 & C_1 & \cdots & C_{\mu-1} \end{pmatrix} \tag{3.5.2}$$

Then the right matrix fraction description (RMFD) of the system is given by: $G(s) = N_R(s)D_R^{-1}(s)$ Where :

$$\begin{cases} A_\mu = I_m \\ D_R(s) = \sum_{i=0}^{\mu} A_i s^i \\ N_R(s) = \sum_{i=0}^{\mu-1} C_i s^i \end{cases} \tag{3.5.3}$$

Remark 3.5.1. The obtained MFD is strictly proper, row-reduced and right coprime.

3.5.3 Block observer form

By duality, if the system described as in equation 3.5.1 is block observable with index ν (as defined in section 3.3.5) the system is transformed into a block observer form as follows:

$$A_o = \begin{pmatrix} 0_p & 0_p & \cdots & 0_p & -A_0 \\ I_p & 0_p & \cdots & 0_p & -A_1 \\ 0_p & I_p & \cdots & 0_p & -A_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_p & 0_p & \cdots & I_p & -A_{\nu-1} \end{pmatrix}; B_o = \begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ \vdots \\ B_{\nu-1} \end{pmatrix}; C_o = \begin{pmatrix} 0_p & \cdots & 0_p & I_p \end{pmatrix} \tag{3.5.4}$$

then the left matrix fraction description (LMFD) of the system is given by: $G(s) = D_L^{-1}(s)N_L(s)$

Where

$$\begin{cases} A_\nu = I_p \\ D_L(s) = \sum_{i=0}^{\nu} A_i s^i \\ N_L(s) = \sum_{i=0}^{\nu-1} B_i s^i \end{cases} \tag{3.5.5}$$

Remark 3.5.2. The obtained MFD is strictly proper, column-reduced and left coprime.

In general, it is not difficult to obtain an MFD from a SSD, but the difficulty is in obtaining a coprime least order MFD. A number of methods to derive the coprime MFD exist in the literature [31, 32], a comparative summary of which is presented in [29].

3.5.4 Relation between latent structure and eigenstructure

From the block diagonal canonical form of a system described in SSD, as developed in section 3.3.5, we have:

$$A_d = \begin{pmatrix} R_1 & 0_m & \cdots & 0_m \\ 0_m & R_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0_m \\ 0_m & \cdots & 0_m & R_\mu \end{pmatrix} \text{ or } A_d = \begin{pmatrix} L_1 & 0_p & \cdots & 0_p \\ 0_p & L_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0_p \\ 0_p & \cdots & 0_p & L_\nu \end{pmatrix} \tag{3.5.6}$$

Let $\tilde{V}$ be the modal matrix of A and $\tilde{V}_d$ be the modal matrix of A_d. So we have:

$$\begin{cases} A = \tilde{V}^{-1}\Lambda\tilde{V} \\ A_c = T_c * \tilde{V}^{-1}\Lambda\tilde{V} * T_c^{-1} \\ A_d = V_R^{-1} * T_c\tilde{V}^{-1}\Lambda\tilde{V}T_c^{-1} * V_R \\ A_d = \tilde{V}_d^{-1}\Lambda\tilde{V}_d \end{cases} \tag{3.5.7}$$

So:

$$\tilde{V}_d = V_R^{-1} * T_c * \tilde{V}^{-1} \tag{3.5.8}$$

Knowing that A_d is a diagonal matrix of right solvents (left), let $\tilde{V}_i$ be the modal matrix of solvent R_i so we can write: $R_i = \tilde{V}_i \Lambda_i \tilde{V}_i^{-1}$ such that Λ_i is a diagonal matrix of eigenvalues of R_i then we have this result:

$$A_d = \begin{pmatrix} \tilde{V}_1\Lambda_1\tilde{V}_1^{-1} & 0_m & \cdots & 0_m \\ 0_m & \tilde{V}_2\Lambda_2\tilde{V}_2^{-1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0_m \\ 0_m & \cdots & 0_m & \tilde{V}_\mu\Lambda_\mu\tilde{V}_\mu^{-1} \end{pmatrix} \tag{3.5.9}$$

This establishes a first step to the relationship between the eigenstructure of a system described in SSD and the latent structure of its corresponding MFD.

3.6 Relation between latent vectors and eigenvectors

In this section we present an important result and contribution to the theory of MFD which is published in [33]. The relationship between eigenstructure and latent structure is the core of the proposed compensator design approach presented in chapter 5.

3.6.1 Block controller form

Let a system be described in block controller form as in equation 3.5.2. Let (λ_i,$\bar{v}_i$) be an eigenvalue and its corresponding right eigenvector of the matrix A_c, hence $A_c\bar{v}_i = \lambda_i\bar{v}_i$. Let (λ_i,v_i) be a latent pair of $D_R(\lambda)$, hence $D_R(\lambda_i)v_i = 0_m$.

Theorem 3.6.1. *The latent vector v_i is obtained from the eigenvector $\bar{v}_i$ by using the following equation:*

$$v_i = \bar{v}_{i1} \tag{3.6.1}$$

where $\bar{v}_{i1}$ is composed of the first m components of $\bar{v}_i$.

Proof. $D_R(\lambda_i)v_i = 0_m \Rightarrow [I_m\lambda_i^\mu + A_{\mu-1}\lambda_i^{\mu-1} + \cdots + A_1\lambda_i + A_0]v_i = 0_m$.

So the latent vector v_i satisfies

$$\lambda_i^\mu v_i + \lambda_i^{\mu-1}A_{\mu-1}v_i + \cdots + \lambda_i A_1 v_i + A_0 v_i = 0_m \tag{3.6.2}$$

On the other hand $A_c\bar{v}_i = \lambda_i\bar{v}_i$ leads to:

$$\begin{pmatrix} 0_m & I_m & 0_m & \cdots & 0_m \\ 0_m & 0_m & I_m & \cdots & 0_m \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \cdots & I_m \\ -A_0 & -A_1 & -A_2 & \cdots & -A_{\mu-1} \end{pmatrix} \begin{pmatrix} \bar{v}_{i1} \\ \bar{v}_{i2} \\ \vdots \\ \bar{v}_{i\mu-1} \\ \bar{v}_{i\mu} \end{pmatrix} = \lambda_i \begin{pmatrix} \bar{v}_{i1} \\ \bar{v}_{i2} \\ \vdots \\ \bar{v}_{i\mu-1} \\ \bar{v}_{i\mu} \end{pmatrix} \tag{3.6.3}$$

Where $\bar{v}_{ij}$with $j = 1..\mu$ are the block elements of the eigenvector $\bar{v}_i$ of dimension m. Hence the following set of equations may be obtained:

$$\begin{cases} \bar{v}_{i2} = \lambda_i\bar{v}_{i1} \\ \bar{v}_{i3} = \lambda_i\bar{v}_{i2} = \lambda_i^2\bar{v}_{i1} \\ \vdots \\ \bar{v}_{i\mu} = \lambda_i\bar{v}_{i\mu-1} = \lambda_i^{\mu-1}\bar{v}_{i1} \end{cases} \tag{3.6.4}$$

The last equation can be rewritten as:

$$-A_0\bar{v}_{i1} - A_1\bar{v}_{i2} - \cdots - A_{\mu-1}\bar{v}_{i\mu} = \lambda_i\bar{v}_{i\mu}$$

or

$$-A_0\bar{v}_{i1} - \lambda_iA_1\bar{v}_{i1} - \cdots - \lambda_i^{\mu-1}A_{\mu-1}\bar{v}_{i1} = \lambda_i^{\mu}\bar{v}_{i1}$$

then

$$\lambda_i^{\mu}\bar{v}_{i1} + \lambda_i^{\mu-1}A_{\mu-1}\bar{v}_{i1} + \cdots + \lambda_iA_1\bar{v}_{i1} + A_0\bar{v}_{i1} = 0_m \tag{3.6.5}$$

Comparing equations 3.6.2 and 3.6.5, we conclude that $v_i = \bar{v}_{i1}$ or

$$v_i = \begin{pmatrix} I_m & 0_m & \cdots & 0_m \end{pmatrix} \begin{pmatrix} \bar{v}_{i1} \\ \bar{v}_{i2} \\ \vdots \\ \bar{v}_{i\mu} \end{pmatrix} \tag{3.6.6}$$

That is: the latent vector v_i of $D_R(\lambda)$ is constituted from the first m components of the eigenvector $\bar{v}_i$ of A_c corresponding to the same latent root/eigenvalue λ_i. □

Conversely we can state the consequent result as a corollary:

Corollary 3.6.2. *The eigenvector $\bar{v}_i$ of A_c can be constructed from the latent vector v_i using:*

$$\bar{v}_i = \begin{pmatrix} v_i \\ \lambda_i v_i \\ \lambda_i^2 v_i \\ \vdots \\ \lambda_i^{\mu-1} v_i \end{pmatrix} \tag{3.6.7}$$

Proof: The result is obtained from equations 3.6.4 and 3.6.6.

3.6.2 Block observer form

By duality we can show the relationship between left eigenvectors and latent vectors. So for a system described in block observer form as in equation 3.5.4.

Let $(\lambda_i, \bar{w}_i)$ be an eigenvalue and its corresponding left eigenvector of the state matrix A_o. Hence $\bar{w}_i A_o = \lambda_i \bar{w}_i$. Let (λ_i, w_i) be a latent pair of $D_L(\lambda)$. Hence $w_i D_L(\lambda_i) = 0_p$.

Theorem 3.6.3. *The latent vector w_i is obtained from the eigenvector $\bar{w}_i$ by using the following equation:*

$$w_i = \bar{w}_{i1} \tag{3.6.8}$$

where $\bar{w}_{i1}$ is composed of the first p components of the eigenvector $\bar{w}_i$.

Proof. $w_i D_L(\lambda_i) = 0_p \Rightarrow w_i[I_p \lambda_i^\nu + A_{\nu-1}\lambda_i^{\nu-1} + \cdots + A_1 \lambda_i + A_0] = 0_p$

So w_i satisfies:

$$\lambda_i^\nu w_i + \lambda_i^{\nu-1} w_i A_{\nu-1} + \cdots + \lambda_i w_i A_1 + w_i A_0 = 0_p \tag{3.6.9}$$

On the other hand $\bar{w}_i A_o = \lambda_i \bar{w}_i$ leads to:

$$\begin{pmatrix} \bar{w}_{i1} \\ \bar{w}_{i2} \\ \bar{w}_{i3} \\ \vdots \\ \bar{w}_{i\nu} \end{pmatrix}^T \begin{pmatrix} 0_p & 0_p & \cdots & 0_p & -A_0 \\ I_p & 0_p & \ddots & 0_p & -A_1 \\ 0_p & I_p & \ddots & 0_p & -A_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_p & 0_p & \cdots & I_p & -A_{\nu-1} \end{pmatrix} = \lambda_i \begin{pmatrix} \bar{w}_{i1} \\ \bar{w}_{i2} \\ \bar{w}_{i3} \\ \vdots \\ \bar{w}_{i\nu} \end{pmatrix}^T \tag{3.6.10}$$

Where $\bar{w}_{ij}$ with $j = 1..\nu$ are the block elements of dimension p of the eigenvector $\bar{w}_i$. Hence the following set of equations may be obtained:

$$\begin{cases} \bar{w}_{i2} = \lambda_i \bar{w}_{i1} \\ \bar{w}_{i3} = \lambda_i \bar{w}_{i2} = \lambda_i^2 \bar{w}_{i1} \\ \vdots \\ \bar{w}_{i\nu} = \lambda_i \bar{w}_{i\nu-1} = \lambda_i^{\nu-1} \bar{w}_{i1} \end{cases} \tag{3.6.11}$$

The last equation can be rewritten as:

$$-\bar{w}_{i1}A_0 - \bar{w}_{i2}A_2 - \cdots - \bar{w}_{i\nu}A_{\nu-1} = \lambda_i \bar{w}_{i\nu}$$

or

$$-\bar{w}_{i1}A_0 - \lambda_i \bar{w}_{i1}A_1 - \cdots - \lambda_i^{\nu-1} \bar{w}_{i1}A_{\nu-1} = \lambda_i^{\nu} \bar{w}_{i1}$$

then:

$$\lambda_i^{\nu} \bar{w}_{i1} + \lambda_i^{\nu-1} \bar{w}_{i1}A_{\nu-1} + \cdots + \lambda_i \bar{w}_{i1}A_1 + \bar{w}_{i1}A_0 = 0_p \tag{3.6.12}$$

Comparing equations 3.6.9 and 3.6.12, we conclude that $w_i = \bar{w}_{i1}$ or

$$w_i = \begin{pmatrix} \bar{w}_{i1} & \bar{w}_{i2} & \cdots & \bar{w}_{i\nu} \end{pmatrix} \begin{pmatrix} I_p \\ 0_p \\ \vdots \\ 0_p \end{pmatrix} \tag{3.6.13}$$

So the latent vector w_i of $D_L(\lambda)$ is composed of the first p components of the eigenvector $\bar{w}_i$ of A_o, corresponding to the same latent root/eigenvalue λ_i. □

Conversely a consequent result is stated as a corollary.

Corollary 3.6.4. *the eigenvector $\bar{w}_i$ of A_o can be constructed from the latent vector w_i using:*

$$\bar{w}_i = \begin{pmatrix} w_i & \lambda_i w_i & \cdots & \lambda_i^{\nu-1} w_i \end{pmatrix} \tag{3.6.14}$$

Proof: The result is straight forward from equations 3.6.11 and 3.6.13.

3.6.3 General state matrix

If $(\lambda_i,\tilde{v}_i)$ is an eigenvalue and right eigenvector of the general state matrix A, and let (λ_i,v_i) be a right latent pair of the matrix $D_R(\lambda)$ of the corresponding right MFD then the following theorem states the relationship:

Theorem 3.6.5. *The latent vector v_i can be obtained from the eigenvector $\tilde{v}_i$ by using the following equation:*

$$v_i = T_{c1}\tilde{v}_i \tag{3.6.15}$$

where T_{c1} is given by equation 3.3.13.

Proof. Recalling that the block controller SSD is generated from a general SSD via the similarity transformation $A_c = T_cAT_c^{-1}$, we have $A_cT_c = T_cA$.

If $(\lambda_i,\ \tilde{v}_i)$ is a right eigenpair of the general state matrix A then $A\tilde{v}_i = \lambda_i\tilde{v}_i$. Hence $A_cT_c\tilde{v}_i = T_cA\tilde{v}_i = \lambda_iT_c\tilde{v}_i$. Thus $(\lambda_i,\ T_c\tilde{v}_i)$ is a right eigenpair of A_c and

$$\bar{v}_i = T_c\tilde{v}_i \tag{3.6.16}$$

It follows from the equation 3.3.13 that:

$$v_i = \begin{pmatrix} I_m & 0_m & \cdots & 0_m \end{pmatrix} T_c\tilde{v}_i \tag{3.6.17}$$

Since $T_c = \begin{pmatrix} T_{c1} \\ T_{c1}A \\ \vdots \\ T_{c1}A^{\mu-1} \end{pmatrix}$ we have

$$v_i = Tc_1\tilde{v}_i = \begin{pmatrix} 0_m & \cdots & 0_m & I_m \end{pmatrix} \begin{pmatrix} B & AB & \cdots & A^{\mu-1}B \end{pmatrix}^{-1} \tilde{v}_i \tag{3.6.18}$$

□

The reverse identity, determining eigenvectors from latent vectors is established in a corollary:

Corollary 3.6.6. *The eigenvector $\tilde{v}_i$ is obtained from its corresponding latent vector v_i by using the following equation:*

$$\tilde{v}_i = T_c^{-1} \begin{pmatrix} v_i \\ \lambda_i v_i \\ \vdots \\ \lambda_i^{\mu-1} v_i \end{pmatrix} \tag{3.6.19}$$

Proof: From $\bar{v}_i = T_c \tilde{v}_i$ (equation 3.6.16) we have $\tilde{v}_i = T_c^{-1} \bar{v}_i$ then using equation 3.6.7 we obtain the precedent result.

We now establish the dual results.

If $(\lambda_i, \tilde{w}_i)$ is an eigenvalue and left eigenvector of the general form of a matrix A, and let (λ_i, w_i) be a left latent pair of the matrix $D_L(\lambda)$ of the corresponding left MFD then the following theorem states the relationship:

Theorem 3.6.7. *The latent vector w_i can be obtained from the eigenvector $\tilde{w}_i$ by using the following equation:*

$$w_i = \tilde{w}_i T_{o1} \tag{3.6.20}$$

where T_{o1} is given by equation 3.3.16.

Proof. From $A_o = T_o^{-1} A T_o$ we have $A T_o = T_o A_o$.

If $(\lambda_i,\ \tilde{w}_i)$ is a left eigen-pair of the general form of a matrix A, then $\tilde{w}_i A = \lambda_i \tilde{w}_i$. Hence $\tilde{w}_i A T_o = \tilde{w}_i T_o A_o = \lambda_i \tilde{w}_i T_o$.

Thus $(\lambda_i, \tilde{w}_i T_o)$ is an eigenpair of A_o and

$$\bar{w}_i = \tilde{w}_i T_o \tag{3.6.21}$$

Thus $w_i = \tilde{w}_i T_o \begin{pmatrix} I_p \\ 0_p \\ \vdots \\ 0_p \end{pmatrix}$.

Since $T_o = \begin{pmatrix} T_{o1} & AT_{o1} & \cdots & A^{\nu-1}T_{o1} \end{pmatrix}$ we have:

$$w_i = \tilde{w}_i T_{o1} = \tilde{w}_i \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{\nu-1} \end{pmatrix}^{-1} \begin{pmatrix} 0_p \\ \vdots \\ 0_p \\ I_p \end{pmatrix} \tag{3.6.22}$$

□

Again, the reverse identity, determining eigenvectors from latent vectors, is given by the following corollary:

Corollary 3.6.8. *The eigenvector $\tilde{w}_i$ is obtained from its corresponding latent vector w_i by using the following equation:*

$$\tilde{w}_i = \begin{pmatrix} w_i & \lambda_i w_i & \cdots & \lambda_i^{\nu-1} w_i \end{pmatrix} T_o^{-1} \tag{3.6.23}$$

Proof: from $\bar{w}_i = \tilde{w}_i T_o$ (equation 3.6.21) we have $\tilde{w}_i = \bar{w}_i T_o^{-1}$ and using equation 3.6.14 we obtain the precedent result.

To summarize:

- If we have the eigenstructure of a system: $(\lambda_i, \tilde{v}_i)$ or $(\lambda_i, \tilde{w}_i)$, then we can determine the latent vectors by using:

  $v_i = \begin{pmatrix} I_m & 0_m & \cdots & 0_m \end{pmatrix} T_c \tilde{v}_i$ or $w_i = \tilde{w}_i T_o \begin{pmatrix} I_p \\ 0_p \\ \vdots \\ 0_p \end{pmatrix}$

- If we have the latent structure of a system: (λ_i, v_i) or (λ_i, w_i), then we can determine the eigenvectors by using: $\tilde{v}_i = T_c^{-1} v_i$ or $\tilde{w}_i = w_i T_o^{-1}$

3.6.4 Numerical example

Consider:

$$\begin{cases} \dot{x} = \begin{pmatrix} 0 & 1 & -1 & 1 \\ 0 & 1 & 1 & -1 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} u \\ y = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} x \end{cases}$$

1) Obtaining latent vectors from eigenvectors using the block controller form:

Using: $T_{c1} = \begin{pmatrix} -0.25 & 0.25 & 0.25 & -0.25 \\ 0.25 & 0.75 & -0.25 & -0.75 \end{pmatrix}$

We obtain the following block controller form:

$$\begin{cases} \dot{x}_c = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & -1 & 2 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} x_c + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} u \\ y = \begin{pmatrix} -1 & -1 & 0 & 2 \\ -3 & 1 & 2 & 0 \end{pmatrix} x_c \end{cases}$$

Then the corresponding RMFD is:

$$\begin{cases} D_R(\lambda) = I_2\lambda^2 + \begin{pmatrix} -2 & -1 \\ -1 & 0 \end{pmatrix}\lambda + \begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix} \\ N_R(\lambda) = \begin{pmatrix} 0 & 2 \\ 2 & 0 \end{pmatrix}\lambda + \begin{pmatrix} -1 & -1 \\ -3 & 1 \end{pmatrix} \end{cases}$$

We can check that:

- The latent roots are: {0, 1, -1, 2} with corresponding latent vectors:

$V = \left\{ \begin{pmatrix} 1 \\ -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ -2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$

- Eigenvalues of A: {0, 1, -1, 2}

- Right eigenvectors are the columns of $\tilde{V} = \begin{pmatrix} 1 & 1 & 6 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -3 & 0 \end{pmatrix}$

- The latent vectors v_i can be obtained from the eigenvectors $\tilde{v}_i$ by using $v_i = T_{c1}\tilde{v}_i$

$$v_1 = T_{c1}\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -0.25 \\ 0.25 \end{pmatrix};$$

$$v_2 = T_{c1}\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix};$$

$$v_3 = T_{c1}\begin{pmatrix} 6 \\ -2 \\ 1 \\ -3 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix};$$

$$v_4 = T_{c1}\begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

It can be verified that the computed vectors are indeed latent vectors of $D_R(\lambda)$.

2) Obtaining eigenvectors from latent vectors using block observer form:

Using $T_o = \begin{pmatrix} -0.25 & -0.75 & -0.25 & -0.75 \\ 0.25 & -0.25 & 0.75 & 0.25 \\ 0.25 & 0.75 & 0.25 & 1.75 \\ -0.25 & 0.25 & 0.25 & -0.25 \end{pmatrix}$

We obtain the following block observer form:

$$\begin{cases} \dot{x}_o = \begin{pmatrix} 0 & 0 & -2 & 0 \\ 0 & 0 & -2 & 1 \\ 1 & 0 & 0.5 & 2.5 \\ 0 & 1 & 1.5 & 1.5 \end{pmatrix} x_o + \begin{pmatrix} -4 & -2 \\ -2 & 0 \\ 0 & 2 \\ 2 & 0 \end{pmatrix} u \\ y = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x_o \end{cases}$$

Then the corresponding LMFD is:

$$\begin{cases} D_L(\lambda) = I_2\lambda^2 + \begin{pmatrix} -0.5 & -2.5 \\ -1.5 & -1.5 \end{pmatrix}\lambda + \begin{pmatrix} 2 & 0 \\ 2 & 0 \end{pmatrix} \\ N_L(\lambda) = \begin{pmatrix} 0 & 2 \\ 2 & 0 \end{pmatrix}\lambda + \begin{pmatrix} -4 & -2 \\ -2 & 0 \end{pmatrix} \end{cases}$$

We can check the following:

- For the same set of eigenvalues {0, 1, -1, 2} the left eigenvectors of the system are given by the rows of $\tilde{W} = \tilde{V}^{-1} = \begin{pmatrix} 1 & -1 & 1 & 3 \\ 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 3 & 1 \end{pmatrix}$

- The left latent vectors of the system are:$W = \left\{ \begin{pmatrix} -1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & -5 \end{pmatrix}, \begin{pmatrix} 1 & -1 \end{pmatrix}, \begin{pmatrix} -1 & -5 \end{pmatrix} \right\}$

- The left eigenvectors of A_o are obtained by: $\bar{w}_i = \begin{pmatrix} w_i & \lambda_i w_i \end{pmatrix}$ and the left eigenvectors of A are obtained by using: $\tilde{w}_i = \bar{w}_i T_o^{-1}$ to get:

$w_1 = \begin{pmatrix} -1 & 1 \end{pmatrix} \Rightarrow \bar{w}_1 = \begin{pmatrix} -1 & 1 & 0 & 0 \end{pmatrix} \Rightarrow \tilde{w}_1 = \begin{pmatrix} 1 & -1 & 1 & 3 \end{pmatrix}$

$w_2 = \begin{pmatrix} 1 & -5 \end{pmatrix} \Rightarrow \bar{w}_2 = \begin{pmatrix} 1 & -5 & 1 & -5 \end{pmatrix} \Rightarrow \tilde{w}_2 = \begin{pmatrix} 0 & 4 & -4 & -4 \end{pmatrix}$

$w_3 = \begin{pmatrix} 1 & -1 \end{pmatrix} \Rightarrow \bar{w}_3 = \begin{pmatrix} 1 & -1 & -1 & 1 \end{pmatrix} \Rightarrow \tilde{w}_3 = \begin{pmatrix} 0 & 0 & 0 & -4 \end{pmatrix}$

$w_4 = \begin{pmatrix} -1 & -5 \end{pmatrix} \Rightarrow \bar{w}_4 = \begin{pmatrix} -1 & -5 & -2 & -10 \end{pmatrix} \Rightarrow \tilde{w}_4 = \begin{pmatrix} 0 & 0 & -6 & -2 \end{pmatrix}$

It can be verified that the computed vectors are the left eigenvectors of the state matrix A.

3.7 The Polynomial Eigenvalue Problem

The Polynomial eigenvalue problem (PEP) consists of computing the eigenvalues and eigenvectors of a matrix polynomial(which are called latent roots and latent vectors in this thesis).

In [34], the authors solve for the latent roots by using matrices in controller companion form, but they do not consider the left or right latent vectors.

Here we exploit the relationships we have established between eigenstructure and latent structure to extend that work. The idea is to construct a low (right) block companion form

matrix from the matrix polynomial and then compute the eigenvalues and right left eigenvectors of these normal matrices. Then, using the relationship established between these eigenvectors and latent vectors, we can directly obtain the latent roots and right left latent vectors of the matrix polynomial.

3.7.1 Introduction

We consider an r^{th} degree order n matrix polynomial given by :

$$D(\lambda) = D_r\lambda^r + D_{r-1}\lambda^{r-1} + \cdots + D_1\lambda + D_0$$

where D_i are $n \times n$ real matrices and either D_0 or D_r is non-singular. If D_r is singular and D_0 is non-singular then $D(\lambda)$ can be rewritten such that $D_0 = I_n$ and $D(\lambda)$ will be monic.

The polynomial eigen problem is how to determine the latent values and the latent vectors of a matrix polynomial such that: $D(\lambda)v = w^T D(\lambda) = 0$ with v and w nonzero vectors.

The authors in [35] propose a method which consists of:

i) transforming D to a regular matrix using pencils matrices

ii) then using a classical method to compute the eigenvalues of this regular matrix.

They validated their method by using "backward stability".

The classical approach to investigating or numerically solving the polynomial eigenvalue problem is linearization, in which D is transformed into a matrix pencil $P(\lambda) = \lambda X + Y$ that satisfies: $E(\lambda)P(\lambda)F(\lambda) = \begin{pmatrix} D(\lambda) & 0 \\ 0 & I_{(k-1)n} \end{pmatrix}$ where E and F are unimodular real matrices.

In [36], the authors adapted this method to be applied for palindromic and even matrix polynomials, they propose pencil spaces which preserve the structure of the given matrix polynomials. As in the previous paper, the authors in [37] propose two forms of matrix pencils $P(\lambda)$ to compute classically its eigenvalues and eigenvectors.

In the preceding section we have established a relationship between the eigenstructure of a general matrix and a block controller matrix and the latent structure of the corresponding matrix polynomial. So this relationship may be used to solve the polynomial eigen problem.

To solve the PEP, in [34], matrices in companion forms (Controller) are proposed to determine the eigenvalues of a matrix polynomial but without referring to eigenvectors either right

or left. The idea here is to construct from the matrix polynomial a low block companion form or a right block companion form matrix and then compute the eigenvalues and right or left eigenvectors of these "normal" matrices. Then using the relationship established between these eigenvectors and latent vectors, we can directly obtain the latent values and vectors, either right or left, of the matrix polynomial. These results have been published in [33].

Matlab uses two matrices as a linearization to compute the eigenvalues and latent vectors of a matrix polynomial through the function `polyeig()`

3.7.2 Algorithm

Let an r^{th} degree n^{th} order monic matrix polynomial be rewritten as:

$$D(\lambda) = I_n\lambda^r + D_{r-1}\lambda^{r-1} + \cdots + D_1\lambda + D_0$$

step1: Construct the low block companion form matrix A_{lb} as in equation 2.5.1:

$$A_{lb} = \begin{pmatrix} 0_n & I_n & 0_n & \cdots & 0_n \\ 0_n & 0_n & I_n & \cdots & 0_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_n & 0_n & 0_n & \cdots & I_n \\ -D_0 & -D_1 & -D_3 & \cdots & -D_{r-1} \end{pmatrix}$$

or the right block companion form matrix A_{rb} as in Equation 2.5.2:

$$A_{rb} = \begin{pmatrix} 0_n & 0_n & \cdots & 0_n & -D_0 \\ I_n & 0_n & \cdots & 0_n & -D_1 \\ 0_n & I_n & \cdots & 0_n & -D_3 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_n & 0_n & \cdots & I_n & -D_{r-1} \end{pmatrix}$$

step2: Compute the eigenvalues and the right (left) eigenvectors of the block companion matrix. In accordance with (Edelman 1995), the eigenvalues are the latent roots of $D(\lambda)$.

step3: Using Equation 3.6.1 or Equation 3.6.8 compute the right (left) latent vectors.

3.7.3 Illustrative example

Consider $D(\lambda) = I_2\lambda^3 + D_2\lambda^2 + D_1\lambda + D_0$ where:

$$D_2 = \begin{pmatrix} 0 & 1 \\ 0 & 5 \end{pmatrix} D_1 = \begin{pmatrix} -1 & 5 \\ 0 & 6 \end{pmatrix} D_0 = \begin{pmatrix} 0 & 4 \\ 0 & 0 \end{pmatrix}$$

1) Right latent structure

The latent roots are {0, 0, 1, -1, -2, -3}.

The right latent vectors corresponding to these latent roots are respectively:

$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ -3 \end{pmatrix} \begin{pmatrix} 1 \\ -12 \end{pmatrix} \right\}$$

Remark 3.7.1. Using the MATLAB function `polyeig`(D_0,D_1,D_2,`eye(2)`),

we obtain the latent roots $= \begin{pmatrix} 0 \\ 1.0000 \\ 0.0000 \\ -1.0000 \\ -3.0000 \\ -2.0000 \end{pmatrix}$

and the latent vectors $= \begin{pmatrix} -1.0000 & 1.0000 & -1.0000 & -1.0000 & -0.0830 & 0.3162 \\ 0 & -0.0000 & -0.0000 & -0.0000 & 0.9965 & -0.9487 \end{pmatrix}$

These all have the correct directions.

2) Left latent structure

For the latent roots {0, 0, 1, -1, -2, -3}, the associated left latent vectors are:

$$\left\{ \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} -6 & 5 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} \right\}$$

Remark 3.7.2. Using the same function `polyeig`(D_0',D_1',D_2',`eye(2)`) to compute the left latent vectors we obtain the following:

Latent roots$= \begin{pmatrix} -3.0000 \\ -2.0000 \\ 0.0000 \\ 0.0000 \\ -1.0000 \\ 1.00000 \end{pmatrix}$

And latent vectors$= \begin{pmatrix} -0.0000 & -0.00000 & 0.00000 & 0.0000 & -1.0000 & 0.7682 \\ -1.0000 & 1.0000 & -1.0000 & 1.0000 & 0.0000 & -0.6402 \end{pmatrix}$

Again these all have the correct directions.

3) Latent roots and right latent vectors:

First we construct the low block companion form of this matrix polynomial:

$$A_{lb} = \begin{pmatrix} 0_2 & I_2 & 0_2 \\ 0_2 & 0_2 & I_2 \\ -D_0 & -D_1 & -D_2 \end{pmatrix}$$

Its eigenvalues and right eigenvectors are the following (computed using MATLAB by using the function `eig`(A_{lb}):

$$\text{Eigenvalues}= \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & -3.0000 \end{pmatrix}$$

$$\text{Eigenvectors}= \begin{pmatrix} 1.0000 & 0.5774 & 1.0000 & -0.5774 & -0.0690 & -0.0087 \\ 0 & -0.0000 & -0.0000 & 0.0000 & 0.2070 & 0.1045 \\ 0 & -0.5774 & -0.0000 & -0.5774 & 0.1380 & 0.0261 \\ 0 & 0 & 0 & 0 & -0.4140 & -0.3134 \\ 0 & 0.5774 & 0.0000 & -0.5774 & -0.2760 & -0.0783 \\ 0 & 0 & 0 & 0 & 0.8281 & 0.9402 \end{pmatrix}$$

The latent roots are the eigenvalues. The right latent vectors are computed using equation 3.6.1:

$$\text{Latent vectors}= \begin{pmatrix} 1.0000 & 0.5774 & 1.0000 & -0.5774 & -0.0690 & -0.0087 \\ 0 & -0.0000 & -0.0000 & 0.0000 & 0.2070 & 0.1045 \end{pmatrix}$$

It is easy to verify that the latent vector directions are correct, through appropriate scaling.

4) Latent roots and left latent vectors

The right block companion form of $D(\lambda)$ is the following:

$$A_{rb} = \begin{pmatrix} 0_2 & 0_2 & -D_0 \\ I_2 & 0_2 & -D_1 \\ 0_2 & I_2 & -D_2 \end{pmatrix}$$

The eigenvalues and left eigenvectors are (computed using MATLAB by using the function `eig`(A'_{rb}):

$$\text{Eigenvalues}= \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -3.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1.0000 \end{pmatrix}$$

$$\text{Eigenvectors}=\begin{pmatrix} 0 & -0.0000 & 0.0000 & 0.0000 & 0.4435 & -0.5774 \\ 1.000 & -1.0000 & -0.1048 & -0.2182 & -0.3696 & 0 \\ 0 & 0 & 0 & 0 & 0.4435 & 0.5774 \\ 0 & 0.0000 & 0.3145 & 0.4364 & -0.3696 & 0 \\ 0 & 0 & 0 & 0 & 0.4435 & -0.5774 \\ 0 & -0.0000 & -0.9435 & -0.8729 & -0.3696 & 0.0000 \end{pmatrix}$$

The latent roots are the eigenvalues. The left latent vectors are computed using equation 3.6.8:

$$\text{Latent vectors}=\begin{pmatrix} 0 & 1.0000 \\ -0.0000 & -1.0000 \\ 0.0000 & -0.1048 \\ 0.0000 & -0.2182 \\ 0.4435 & -0.3696 \\ -0.5774 & 0 \end{pmatrix}$$

Again, it is easy to verify that the latent vector directions are correct, through appropriate scaling.

3.8 Conclusion

The importance of both MFD and SSD in control theory is well known. The MFD provides a very natural way of expressing desired zero/pole positions, whereas the eigenstructure of the SSD is a natural way of describing a desired multivariable system time response. At the onset of this work, we privately postulated that, if we could establish the structural links between them, then we would be able to combine design methodologies and get the benefits of both descriptions.

We have achieved this very important initial result and have been able to utilize it in proposing a new algorithm for solving the polynomial eigenvalue problem for any regular matrix polynomial. The proposed algorithm is easier and requires less computing time and memory to determine the eigenvalues and eigenvectors of block controller/observer form state matrix than pencil matrix. The proposed method is similar to the one used by the Control system toolbox of Matlab, but the latter uses two matrices.

Chapter Bibliography

[1] M. G. Singh and J-P Elloy, Applied industrial control, Vol.1, Pergamon press, 1980.

[2] C. F. P. M Bekkers, Structure and Parameter identification of multivariable systems repre-

sented by matrix fraction description, M.Sc. Report, University of Technology of Eindhoven, Netherlands, 1985.

[3] J. C. Zúñiga, J. Ruiz-Lon and D. Henrion, "Algorithm for decoupling and complete pole assignement of linear multivariable systems", in Proc. of European Control Conference, Cambridge, UK, 2003.

[4] T. Kailath, Linear Systems, Prentice Hall, Englewood Cliffs, 1980.

[5] H. Rosenbrock, State-Space and Multivariable Theory, Wiley, New York, 1970.

[6] V. Kucera , Discrete Linear Control: The Polynomial Equation Approach, John Wiley and Sons, Chichester, 1979.

[7] J. W. Polderman and J. C. Willems, Introduction to Mathematical Systems Theory: a Behavioral Approach, Springer-Verlag, 1998.

[8] F.M. Callier and C.A. Desoer, Multivariable Feedback Systems, Springer-Verlag, New York, 1982.

[9] C.T. Chen, Linear System Theory and Design, Holt Rinehart and Winston, NY, 1984.

[10] W. A. Wolovich, Linear Multivariable Systems, Springer-Verlag, New York, 1974.

[11] J. Descusse and J. M. Dion, "On the structure at infinity of linear square decoupled systems", IEEE Trans. Autom. Contr., Vol. AC-27, No.4, pp. 971-974, 1982.

[12] M. Malabre and V. Kucera, "Infinite structure and exact model matching problem: a geometric approach", IEEE Trans. Autom. Contr., Vol. AC-29, No.3, pp. 266-268, 1984.

[13] S. P. Bhattacharyya, "Frequency domain conditions for disturbance rejection", IEEE Trans. Autom. Contr., Vol AC-25, No.6, pp. 1211-1213, 1980.

[14] M. Grimble and V. Kucera (Eds.), A polynomial approach to H_2 and H_∞ robust control design, Springer-Verlag, London, 1996.

[15] H. Kwakernaak and M. Šebek, "Polynomial J-spectral factorization", IEEE Trans. Autom. Contr., Vol. AC-39, No.2, pp. 315-328, 1994.

[16] L. S . Shieh, F. R. Chang and B. C. Mcinnis, "The Block Partial Fraction Expansion of a Matrix Fraction Description with Repeated Block Poles", IEEE Trans. Autom. Contr., Vol. AC-31, No.3, pp. 236-239, 1986.

[17] J. Leyva-Ramos, "A New Method for Block Partial Fraction Expansion of Matrix Fraction Descriptions", IEEE Trans. Autom. Contr., Vol. AC-36, No.12, pp. 1482-1485, 1991.

[18] L. S. Shieh, M. M. Mehio and R. E. Yates, "Cascade Decomposition and Realization of Multivariable Systems Via Block-Pole and Block-Zero Placement", IEEE Trans. Autom. Cont., Vol AC-30, No.11, pp. 1109-1112, 1985.

[19] L. S. Shieh, Y. T. Tsay and R. E. Yates, "State-Feedback Decomposition of Multivariable Systems Via Block-Pole Placement", IEEE Trans. Autom. Contr., Vol. AC-28, No.8, pp. 850-852, 1983.

[20] A. J. Pomfret, Eigenstructure Assignment for Helicopter Flight Control, PhD Thesis, Department of Electronics, University of York, UK, 2006.

[21] R. Davies, Robust Eigenstructure Assignment for Flight Control Applications, PhD thesis, University of York, UK, 1994.

[22] K. Zhou, J. C. Doyle and K. Glover, Robust and Optimal Control, Prentice Hall, 1996.

[23] V. Mehrmann and H. Xu, "Numerical methods in control', J. of Computational and Applied Mathematics, Vol. 123, pp. 371-394, 2000.

[24] L. S. Shieh and Y. T. Tsay, "Transformation of a class of multivariable control system to block companion forms", IEEE Trans. Autom. Contr., Vol AC-27, No.1, pp. 199-203, 1982.

[25] C. -C. Tsui, , Robust Control System Design - Advanced State-Space Techniques, Marcel Dekker, New York, 1996.

[26] R. P. Guidorzi and S. Beghelli, "A new input-output form for multivariable systems", IEEE Trans. Autom. Contr., Vol. AC-21, No. 5, pp. 692-696, 1976.

[27] R. P. Guidorzi, "Invariants and canonical forms for systems, structural and parametric identification", Automatica, Vol. 17, No. 1, pp. 117-133, 1981.

[28] W.A. Wolovich, "The determination of state-space representation for linear multivariable systems", Automatica, Vol.9, No.1, pp. 97-106, 1973.

[29] R.V. Patel, "Computation of matrix fraction description of linear time-invariant systems", IEEE Trans. Auto. Contr., Vol AC-26, No.1, pp. 148-161, 1981.

[30] P. Stefanidis, A. P. Paplinski, and M. J. Gibbard, Numerical Operations with Polynomial Matrices: Application to Multivariable Dynamic Compensator Design, Lecture Notes in Control and Information Sciences, Vol. 171, Springer, Berlin, 1992.

[31] S. Wang and E. J. Davison, "A minimization algorithm for the design of linear multivariable systems", IEEE Trans. Autom. Contr., Vol. AC-18, pp. 220-225, 1973.

[32] J. C. Basilio and B. Kouvaritakis, "An algorithm for coprime matrix fraction description using Sylvester matrices", Lin. Algeb. And its App., Vol. 266, pp. 107-125, 1997.

[33] M. Yaici, K. Hariche and T. Clarke, A contribution to the Polynomial Eigen Problem, Int. J. Mathemat., Comp., Physic. and Quant. Eng., Vol.8, No.10, pp. 1228-1235, 2014.

[34] A. Edelman and H. Murakami,"Polynomial Roots from Companion Matrix Eigenvalues", Math. Comp., Vol. 64, pp. 763-776, 1995.

[35] P. Van Dooren and P. Dewilde, "The Eigenstructure of an Arbitrary Polynomial Matrix: Cdmputational Aspects",Lin. Alg. App., Vol.50, pp. 545-579, 1983.

[36] D. S. Mackey, N. Mackey, C. Mehl and V. Mehrmann, "Structured Polynomial Eigenvalue Problems: Good Vibrations from good Linearizations", SIAM J. Matrix Anal. App., Vol. 28, No. 4, pp. 1029-1051, 2006.

[37] N. J. Higham, D. S. Mackey and F. Tisseur, "The Conditiong of Linearization of Matrix Polynomials", SIAM J. Matrix Anal. App., Vol. 28, No. 4, pp. 1005-1028, 2006.

Chapter 4

Feedback control

4.1 Introduction

Before the approaches to control can be considered, it is important to understand the form of the systems to which control is to be applied. Firstly, a mathematical model of a system is formed. This model is then cast into a useful form (matrix fraction description or state space description) for the purposes of applying control, and finally some function of the output is fed back to the input in order to affect the behaviour of the system.

The problem of eigenstructure assignment (EA) and poles placement has a long history going back to Rosenbrock in 1960's [1]. And it has attracted much research interest over many years; see for instance the survey paper of White [2].

Pole placement (also called Pole Assignment or Pole Allocation) is placing the poles or eigenvalues of the closed-loop system at specified locations.

Eigenstructure assignment is the process of applying negative feedback to a linear, time-invariant system with the objective of forcing the eigenvalues and eigenvectors (the eigenstructure) to become as close as possible to a desired eigenstructure.

The eigenvalues and eigenvectors can determine system performance and robustness far more directly and explicitly than other indicators. Hence their assignment should improve feedback system performance and robustness effectively.

There exists, in the literature, a fair amount of confusion between the terms 'eigenstructure assignment' and 'pole placement'. If eigenvector assignment is not considered, or if the eigenvectors are implicitly assigned to meet a goal that does not form part of the design specification, then such an algorithm constitutes pole placement and not EA.

In this chapter, a review of the control techniques of multi-variable time-invariant linear

systems to achieve pole placement and eigenstructure assignment is presented. The chapter is decomposed in two parts: Part 1 is dedicated to control methods for systems described in state space equations and Part2 is dedicated to matrix polynomials control methods. We consider MIMO systems with n states, m inputs and p outputs described in state space equations (SSD) or in transfer function as matrix fractions (MFD).

4.2 Literature review

One of the most popular and well known techniques used to assign the eigenvalues of a closed loop system to desired locations is state feedback. In the case of multivariable systems, the feedback gain matrix permitting the assignment of the desired set of poles is not unique.

The fundamental result on pole placement by state feedback in linear time-invariant controllable systems was presented in the 1960's by Wonham [3] who states that the closed loop eigenvalues of any controllable system may be arbitrarily assigned by state feedback control. Davison in 1970 generalized Wonham's result and showed that if the number of output variables p is less than the order of the system n, then it is always possible, by a constant feedback gain matrix, to assign p poles of the closed-loop system matrix [4]. Song and Ishida developed a method to assign the poles of the system, only one output and only one input in system was used to create the feedback controller [5]. Many different aspects of pole placement via feedback have been studied [6, 7]

Historically, computer tools use state space methods to calculate compensators in the frequency domain because of numerical reasons. Nowadays, a more direct approach through transfer functions and matrix polynomials seems to be more adequate [8]. A number of frequency domain philosophies have been advanced in which appealing analytic solutions are obtained for the classical asymptotic design problems: stabilization, regulation, tracking and disturbance rejection, robust design, etc.

But the pole placement problem is typically still formulated in the time domain. Interestingly, however, in the frequency domain design formulation, one may control the number as well as the placement of the poles via the use of an appropriate dynamic compensator.

In [9] an algorithmic procedure is proposed for the computation of a proper compensator

to achieve internal stability by assigning a desired closed loop denominator obtained through Euclidean divisions of matrix polynomials.

Hippe and O'Reilly [10] proposed a solution to the problem of parametric eigenvalue assignment using general dynamic output feedback compensator of order l ($0 \leq l \leq n$). They showed that $n + l$ eigenvalues are assignable by a dynamical compensator of order $l = r - 1$ where r is either the system controllablity index or the system observability index.

The famous book by Kailath [11] documents the aspect of system theory in matrix fractions thoroughly, and in [12] the numerical aspect of polynomial matrices and their use in control theory (closed loop system compensator design) has been detailed and illustrated with many design examples.

In [13] a direct pole placement algorithm is introduced for dynamical systems having a block companion state matrix. The algorithm utilizes well established properties of matrix polynomials. Pole placement is achieved by appropriately assigning coefficient matrices of the corresponding matrix polynomial then computing the state feedback allowing the placement.

A large-scale MIMO system described by state equations in general coordinates is often decomposed into small subsystems, from which the analysis and design of the MIMO system can be performed. In [14] a new block-pole placement for the state-feedback block decomposition of a class of MIMO systems is derived.

In [15] a matrix fraction description for the development of a compensator for a linear system using a state feedback law and estimator design for that system is presented and in [16] a new approach to compute the coprime MFD and the state feedback gains of MIMO systems is presented.

In state space description, the modal decomposition of the state matrix into its eigenstructure is very useful as it defines the stability and the dynamic behavior of a linear multi-variable system. In general, the speed of response is determined by the eigenvalues whereas the shape of the response is furnished by the eigenvectors. If, through feedback, we are able to assign the eigenvalues to predetermined values and we are able to align the closed loop eigenvectors along predetermined directions, we will be able to control the behaviour of a linear multivariable system in both speed of response and shape of the response, achieving design objectives such as input and output decoupling, reducing sensitivity to perturbations in dynamic structure as

well as appropriate stability criteria [17].

Eigenstructure assignment is a design methodology that facilitates control system design by synthesizing a feedback gain matrix that exactly places the closed loop eigenvalues whilst matching the closed loop eigenvectors as closely as possible to a desired set [18]. Such technique can be related to the design objectives as well as the final performance through its clear links with the time domain response [19].

Eigenstructure assignment is an excellent method for incorporating classical specifications on damping, settling time, and mode decoupling into a modern multivariable control framework [18], and has been shown to be a useful tool for flight control design [20, 21].

In the past 30 years, techniques for eigenstructure assignment have been widely investigated and applied to many problems. Eigenstructure assignment has been achieved using both state and output feedback [22].

Eigenstructure assignment is a natural choice for the design of any control system whose desired performance is readily represented in terms of an ideal eigenstructure [17].

The degrees of freedom available in eigenstructure assignment using state-feedback control are well known. In 1976, Moore [23] described the freedom available to assign eigenvectors for an arbitrary self-conjugate set of eigenvalues using state feedback. He gives both necessary and sufficient conditions for a full-state feedback matrix K to exist. Furthermore, if such a feedback matrix K exists and the input matrix B is of full rank, then K is unique.

If B is full rank, a maximum of n eigenvectors can be partially assigned with a minimum of m entries in each eigenvector arbitrarily chosen. For a particular problem, it may be desirable to consider more than the minimum number of entries in a given eigenvector. In this case, the set of basis vectors which span the allowable subspace must be determined and a best possible achievable eigenvector chosen [24].

The authors of the interesting paper [19] discussed the conditions for the number of reachable eigenvectors, and presented the different techniques to assign them, via full state feedback, output feedback and constrained output feedback.

In more recent papers, the authors of [25] proposed an iterative algorithm, based on alternating projections ideas. Given n subsets of the complex plane, the algorithm is used to find a static output feedback that places a pole for each subset. Kimura' condition ($m + p > n$) [26]

is generally considered as the best sufficient condition for a problem of EA to always have a solution, and Wang's condition ($mp > n$) is less restrictive but the solution if it exists is not easily obtained. In [27] a simple non-iterative technique is proposed, it is based on eigenstructure assignment, which places, by static output feedback, $m + p$ poles when $mp > m + p$. So if $m + p = n$ and $mp > n$ then the method assigns the whole desired spectrum.

In [28] a two-stage design process is used to formulate the gain matrix. In the first stage a subset of desired eigenvectors is assigned, then, in the second stage, a dual set is assigned if necessary and sufficient conditions are met. In [29] an algorithm is presented which exploits unused design freedom to introduce structure to the resulting gain matrix without affecting the assigned eigenstructure by output feedback. Magni, in [30], proposed a control design approach based on EA by dynamic feedback to handle simultaneously robustness against real parameter variations, and the use of structured gain including scheduled gains.

It has been shown that, in the multivariable case, there is a great deal of freedom in the choice of the feedback gain matrix to achieve a given set of closed-loop modes, and that transfer function analysis leads to a unique characterization of this feedback gain matrix in terms of the eigenvalues and the eigenvectors of the closed loop system [11].

4.3 State space description

4.3.1 Pole placement using state feedback

Consider the following state feedback system:

$$\dot{x} = Ax + Bu \tag{4.3.1}$$

$$u = Kx + z \tag{4.3.2}$$

where A is $n \times n$, B is $n \times m$, K is $m \times n$ and $rank(B) = m$. Let z be the reference input vector and K is a real constant matrix called the feedback gain matrix.

The closed loop system is generated by substituting Equation 4.3.2 into Equation 4.3.1 as follows:

$$\dot{x} = Ax + B(Kx + z) = (A + BK)x + Bz \tag{4.3.3}$$

It has been shown that if the system is controllable then the eigenvalues of $(A + BK)$ can

be arbitrarily assigned by a proper choice of K, provided that complex conjugate eigenvalues appears in pairs [31, 11].

Many methods exist to solve for the gain matrix K. In [11] a method based on transforming the system $\{A,\ B\}$ into a controller form, and another direct method based on cyclic state matrix, are given.

In this thesis we choose to give the algorithm of the method which consists in solving a Lyapunov equation [31], and summarized in the following steps:

1) Choose an arbitrary $n \times n$ matrix F with no common eigenvalues with the state matrix A.

2) Choose an arbitrary $n \times n$ matrix K' such that $\{F,\ K'\}$ is observable.

3) Solve for a unique matrix T such that the Lyapunov equation is satisfied: $AT - TF = -BK'$.

4) If T is non-singular, then we have $K = K'T^{-1}$ and the closed loop state matrix $(A - BK)$ has the same eigenvalues as the matrix F. If T is singular, then choose a different matrix F to get a different matrix K' and repeat the process.

4.3.2 Pole placement using output feedback

The state feedback problem assumes that all the states are measurable. Unfortunately, this is impractical for most systems. To solve this, either an observer is designed to estimate the states, or it is usually required to use output feedback.

The equivalent output feedback system is as follows:

$$\dot{x} = Ax + Bu; y = Cx \tag{4.3.4}$$

$$u = Ky + z \tag{4.3.5}$$

where A is $n \times n$, B is $n \times m$, C is $p \times n$, K is $m \times p$, $rank(B) = m$ and $rank(C) = p$.

The closed loop system is:

$$\dot{x} = (A + BKC)x + Bz; y = Cx \tag{4.3.6}$$

So the closed loop system eigenvalues is the set of eigenvalues of the matrix $(A + BKC)$ and thus can be set by choosing the right gain matrix K.

If the system $\{A, B, C\}$ is both controllable and observable and satisfies $m + p > n$, then it is pole-assignable, i.e., for any given desired eigenvalue (λ_i) there exists K such that the eigenvalues of $(A + BKC)$ is in arbitrary neighborhood of (λ_i) [19, 32].

Many methods exist, among them the pseudo-inverse matrix method [33].

The eigenvalues of the closed loop system can be written as: $\Lambda = diag\{eig(A - BK_oC)\}$ where K_o is the output feedback gain matrix. Let K_s be the state feedback gain matrix, then $\Lambda = \{eig(A - BK_s)\}$. From these two equations we can deduce: $K_o = K_sC^+$ where C^+ is the pseudo-inverse of the output matrix C.

4.3.3 Block pole placement using state feedback

Given a MIMO system described by the state equations:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{4.3.7}$$

With n states, m inputs and p outputs.

Let the system be block controllable and $\mu = n/m$ being an integer. Let $D_f(\lambda)$ be a desired matrix polynomial, find an $m \times n$ gain matrix K such that under state feedback: $u = z - Kx$ the matrix $(A - BK)$ in the new state equation: $\dot{x} = (A - BK)x + Bz$ has the desired characteristic matrix polynomial $D_f(\lambda)$. The desired matrix polynomial $D_f(\lambda)$ is constructed from a desired set of block poles (see section 2.3.6 of chapter 2).

1) Block controller form:

Let a system described by equations 4.3.7 be block controllable and $n = \mu * m$, then it is transformed into a block controller form $\{A_c, B_c, C_c\}$ (as shown in chapter 3). The method consists in the computation of the state feedback gain matrix K_c such that the closed loop matrix $A_c - B_cK_c$ has a desired right characteristic matrix polynomial $D_f(\lambda)$ [34].

Let the state feedback control law be: $u = z_c - K_cx_c$ where z_c is the new reference input, and $K_c = \begin{pmatrix} K_{c\mu} & K_{c\mu-1} & \cdots & K_{c1} \end{pmatrix}$ is a $m \times \mu m$ block gain matrix where K_{ci} $(i = 1..\mu)$ are $m \times m$ matrices.

Then the closed loop state equations become:

$$\begin{cases} \dot{x}_c = \hat{A}_c x_c + B_c x_c \\ y = C_c x_c \end{cases} \tag{4.3.8}$$

Where the closed-loop system matrix $\hat{A}_c$ is given by:

$$\hat{A}_c = \begin{pmatrix} 0_m & I_m & 0_m & \cdots & 0_m \\ 0_m & 0_m & I_m & \cdots & 0_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \cdots & I_m \\ -\hat{A}_0 & -\hat{A}_1 & -\hat{A}_2 & \cdots & -\hat{A}_{\mu-1} \end{pmatrix}$$

where $\hat{A}_i = \hat{A}_i + K_{ci}$ for $i = 1..\mu$ hence $K_{ci} = \hat{A}_i - A_i$.

The closed loop right characteristic matrix polynomial is:

$$D_R(\lambda) = \sum_{i=0}^{\mu-1} \hat{A}_i \lambda^i ; \hat{A}_\mu = I_m \tag{4.3.9}$$

2) General form:

If a system is controllable, but n/m is not an integer, then the previous method cannot be applied. One solution is to enlarge the system by adding a set of non-dominant eigenvalues [14].

Another method is proposed by [35] which consist in using a similarity transformation to obtain another block decomposition of the system. The method is summarized in the following steps:

1) Let a system be described by state equation as in equation 4.3.7, controllable but n/m is not an integer, then we can write: $n = \mu * m + k$ with $k < m$.

   Let $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ be the set of eigenvalues of the state matrix A.

2) From this set, choose k eigenvalues and let $\tilde{v}_i$ $i = 1..k$ its corresponding right eigenvectors and let $\tilde{w}_i$ $i = 1..k$, its corresponding left eigenvectors such that the following matrix is non-singular:

$$\Phi = \begin{pmatrix} B & AB & \cdots & A^{\mu-1}B & \tilde{v}_1 & \tilde{v}_2 & \cdots & \tilde{v}_k \end{pmatrix} \tag{4.3.10}$$

3) Use the similarity transformation which transforms x to x_c: $x_c = T_c x$ where $T_c = \begin{pmatrix} T_{c1} \\ \vdots \\ T_{c\mu} \\ T_{c\mu+1} \end{pmatrix}$ and T_{ci} $(i = 1..\mu)$ are $m \times n$ matrices and $T_{c\mu+1}$ is a $k \times n$ matrix to ob-

tain a new system:

$$\left\{ \begin{array}{c} \dot{x}_c = A_c x_c + B_c u \\ y = C_c x_c \end{array} \right. \quad \text{where} \quad \left\{ \begin{array}{c} A_c = T_c A T_c^{-1} \\ B_c = T_c B \\ C_c = C T_c^{-1} \end{array} \right. \tag{4.3.11}$$

and: $A_c = \left(\begin{array}{cccc|c} 0_m & I_m & \cdots & 0_m & 0_{m\times k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_m & 0_m & \cdots & I_m & 0_{m\times k} \\ -A_0 & -A_1 & \cdots & -A_{\mu-1} & 0_{m\times k} \\ -- & -- & -- & -- & -- \\ 0_{m\times k} & 0_{m\times k} & \cdots & 0_{m\times k} & \Lambda_k \end{array} \right)$

with $\Lambda_k = diag\{\lambda_1, \lambda_2, \cdots, \lambda_k\}$

and $B_c = \left(\begin{array}{c} 0_m \\ \vdots \\ I_m \\ B_{mk} \end{array} \right)$

The result can be rewritten as:

$$\left\{ \begin{array}{c} A_c = \left(\begin{array}{cc} A_{c1} & 0_{\mu m\times k} \\ 0_{k\times \mu m} & P \end{array} \right) \\ B_c = \left(\begin{array}{c} B_{c1} \\ B_{c2} \end{array} \right) \end{array} \right. \tag{4.3.12}$$

3) Compute a state feedback gain K_{c1} to place the desired block poles of $(A_{c1} - B_{c1}K_{c1})$ at the desired $\mu * m$ desired locations using:

$$K_{c1} = \left(\begin{array}{ccc} K_\mu & \cdots & K_1 \end{array} \right) \tag{4.3.13}$$

Where $K_i = D_i - A_i$ $i = 1..\mu$ are $m \times m$ matrices, A_i are obtained from equation 4.3.12 and D_i are the matrix coefficients of the desired closed loop denominator $D_f(s)$.

4) Compute a $k \times \mu m$ matrix L satisfying the following Lyapunov equation:

$$L(A_{c1} - B_{c1}K_{c1}) - \Lambda_k L = B_{c2}K_{c1} \tag{4.3.14}$$

5) Compute a feedback gain matrix K_{c2} that places the k poles of $\Lambda_k - (B_{c2} + LB_{c1})K_{c2}$ at the k remaining desired locations.

6) Construct the general state feedback gain matrix as: $K_c = \left(\begin{array}{cc} K_{c1} + K_{c2}L & K_{c2} \end{array} \right)$ and transform it back into its original coordinates as: $K = K_c T_c$.

4.3.4 Eigenstructure assignment using state feedback

Given a controllable state space system (A, B) as in equation 4.3.1, find a real gain matrix K that assigns a desired set of self conjugate eigenvalues (λ_i) and right eigenvectors (v_i) .

Consider the following state feedback system:

$$\dot{x} = Ax + Bu \tag{4.3.15}$$

$$u = Kx + z \tag{4.3.16}$$

where A is $n \times n$, B is $n \times m$, K is $m \times n$ and $rank(B) = m$. z is the reference input vector. The closed loop system is given by:

$$\dot{x} = (A + BK)x + Bz \tag{4.3.17}$$

So the closed loop system eigenstructure is the set of eigenvalues and associated eigenvectors of the matrix $(A + BK)$.

Moore in 1976 [23] shows that for the assignment to succeed, the selected eigenvectors $\{\tilde{v}_i\}$ must be linearly independent and that $\lambda_i = \lambda_i^*$ must imply $\tilde{v}_i = \tilde{v}_i^*$.

4.3.5 EA using output feedback

Given a controllable and observable system and an output control system, the equivalent output feedback system:

$$\dot{x} = Ax + Bu; y = Cx \tag{4.3.18}$$

$$u = Ky + z \tag{4.3.19}$$

where A is $n \times n$, B is $n \times m$, C is $p \times n$, K is $m \times p$, $rank(B) = m$ and $rank(C) = p$ (full rank).

The closed loop system is:

$$\dot{x} = (A + BKC)x + Bz; y = Cx \tag{4.3.20}$$

The output feedback problem is defined as follows: Given the state space system (A, B, C) defined by equations 4.3.18, find a real gain matrix K that assigns a desired set of n self-conjugate eigenvalues (λ_i) and a total of n right or left eigenvectors ($\tilde{v}_i$) or ($\tilde{w}_i$) to the closed-loop system matrix $(A + BKC)$.

It has been shown that for a self-conjugate set of eigenvalues $\{\lambda_1,\lambda_2,\cdots,\lambda_n\}$ with a corresponding set of eigenvectors $\{\tilde{v}_1,\tilde{v}_2,\cdots,\tilde{v}_n\}$ there exist a matrix K such that [36]:

$$(A - BKC)\tilde{v}_i = \lambda_i \tilde{v}_i$$

as long as the following conditions are satisfied:

- vectors $\tilde{v}_i$ are linearly independent in $\mathbb{C}^n$.
- $\tilde{v}_i = \tilde{v}_j^*$ whenever$\lambda_i = \lambda_j^*$.
- $\tilde{v}_i$ belongs to the span of allowable eigenvector subspace.

It is therefore possible, by considering either the right or left eigenvector subspaces, to specify $max(m,p)$ eigenvalues and $min(m,p)$ elements of each corresponding eigenvector, a result found by Srinathkumar (1978)[37]. However, by imposing more stringent constraints on the eigenvector subspaces, it is possible to assign both $a \leq m$ eigenvalues and right eigenvectors, and $b \leq r$ eigenvalues and left eigenvectors (though usually not if $a = m$ and $b = p$); hence, if $m + p > n$, it is possible to assign all n eigenvalues [29].

Let Λ be the diagonal matrix of desired eigenvalues and let $\tilde{V}$ be the right eigenvector matrix and $\tilde{W}$ the left eigenvector matrix, then the gain matrix K must satisfy the following three equations:

$$\begin{cases} \tilde{W}(A - BKC) = \Lambda\tilde{W} \\ (A - BKC)\tilde{V} = \tilde{V}\Lambda \\ \tilde{W}\tilde{V} = I \end{cases} \text{ or in another form: } \begin{cases} \tilde{W}A - \Lambda\tilde{W} = \tilde{W}BKC = K'C \\ A\tilde{V} - \tilde{V}\Lambda = BK' \\ \tilde{W}\tilde{V} = I \end{cases}$$

In [38] a detailed algorithm is given to solve for the gain matrix which consist in two major steps.

4.3.6 Dynamic compensation

Dynamic compensation increases the available design freedom and can therefore achieve design goals that are not possible using static feedback alone. Dynamic compensation methods yield controllers with dynamic rather than static gains, which can result in an augmented system with improved design freedom. For a system given as in equation 4.3.7 the compensator can be described by the following state equations:

$$\begin{aligned} \dot{x}_{co} &= A_{co}x_{co} + B_{co}u \\ y &= C_{co}x_{co} + D_{co}u \end{aligned} \tag{4.3.21}$$

The closed loop system is an augmented system, and the problem becomes for which arbitrarily assignable eigenvalues of the compensator the assignment of desirable eigenvalues of the plant is achieved [30]. In general classical techniques such as washout filters and lead-lag networks for integration into the augmented system are used.

Using a dynamic compensator instead of a static feedback network increases the number of DoF available for the design of a control system. It has been seen that the distribution of this additional freedom is not simple, and its exploitation fraught with difficulties; the system order is increased, transmission zeros are added, and no additional freedom over the coupling of modes into the original system states is gained. Although the additional DoF can be exploited for other means [10], their use for EA is very limited [29].

4.4 Matrix fraction description

For systems described in transfer function where the denominator and the numerator are matrix polynomials, the following are methods to design compensators.

4.4.1 Introduction

The poles and zeros of the transfer function of a system define its dynamic response to any given input. Manipulation of the location of these poles and zeros is therefore the aim of classical feedback control system design. Feedback can take many forms. In each case the system output is taken, modified and used to augment the input, leading to a modified system response.

The lack of design freedom offered by the simple gain controllers can be overcome by choosing a more complex transfer function for the controller.

Many different configurations exist. Their application and design is the choice of the control engineer, and although tools exist for assessing the likely impact of the use of a compensator, such design decisions must ultimately come from experience. The most used feedback configurations are: unity feedback, output feedback and input-output feedback depending on where the compensator is placed in the closed loop system.

The design process finishes with the resolution of a compensator equation, also called Diophantine equation. Conditions for a solution to exist and methods to solve this equation are

detailed in Appendix A. So the main objective in this section is to find the best feedback configuration to solve this Diophantine equation.

4.4.2 Problem formulation

Figure 4.1 shows a simple form of feedback control known as unity negative feedback. The output is subtracted from the input, leaving an error signal which is fed to the system input. The overall input to the closed-loop system is now a demand for a particular output, and while the system output does not match the demand, the error signal will be nonzero. A controller is added and in the simplest case, $G_c(s)$ would be a simple gain. Only for this section, we will deal with single-input single output (SISO) systems. The effect on the system transfer function is simple to deduce.

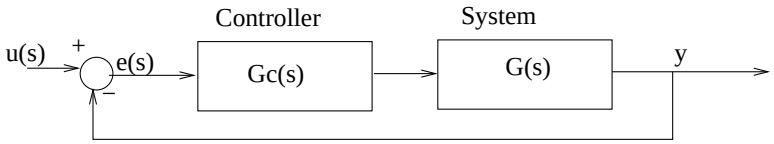


Figure 4.1: Unity negative feedback

The transfer function $G_{cl}(s)$ of the equivalent unity negative feedback closed-loop system can now be found:

$$G_{cl}(s) = \frac{y}{u} = \frac{G(s)G_c(s)}{1 + G(s)G_c(s)} \tag{4.4.1}$$

The poles and zeros of this closed-loop system are interesting to investigate. If the open loop transfer function $G(s)$ is defined as:

$$G(s) = Z(s)/P(s) \tag{4.4.2}$$

and the controller $C(s)$ as:

$$G_c(s) = Zc(s)/Pc(s) \tag{4.4.3}$$

then by substitution,

$$G_{cl}(s) = \frac{Z(s)Z_c(s)}{P(s)P_c(s) + Z(s)Z_c(s)} \tag{4.4.4}$$

Examination of equation 4.4.4 shows that any root of $Z(s)$ is a root of the numerator of the closed-loop transfer function $G_{cl}(s)$. This shows that the zeros of $G(s)$ are invariant under

feedback, though any zeros present in the controller $G_c(s)$ will manifest themselves as additional zeros in $G_{cl}(s)$.

An alternative feedback structure is shown in figure 4.2. Now the controller $G_c(s)$ is placed in the feedback path. Note that in the absence of an external input $u(s)$, the structure is identical to that of the system in figure 4.1. Consequently it would be expected that the natural response would be the same, and hence that the closed loop system poles would also be the same.

The role of the external input $u(s)$ has changed, however. In the system of figure 4.1, it acted to demand a particular output response; this is no longer the case, since the summing junction appears after the feedback signal has been modified by the controller $G_c(s)$. For the same reason, it is no longer strictly accurate to refer to the signal $e(s)$ as the error signal, though the notation will be retained for convenience.

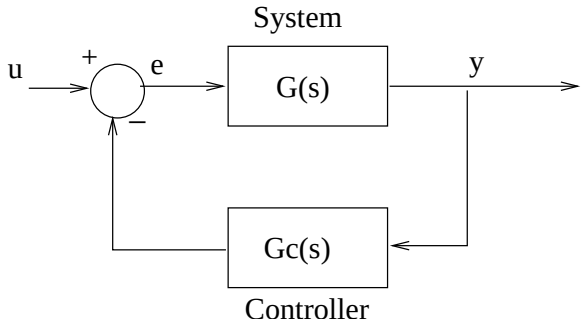


Figure 4.2: Alternative negative feedback

Analysis of figure 4.2 allows the construction of the closed loop transfer function:

$e(s) = u(s) - G_c(s)y(s)$ and

$y(s) = G(s)e(s) = G(s)u(s) - G(s)G_c(s)y(s) \Rightarrow (1 + G(s)G_c(s))y(s) = G(s)u(s)$

Then the transfer function is given by:

$$G_{cl}(s) = \frac{y(s)}{u(s)} = \frac{G(s)}{1 + G(s)G_c(s)} \tag{4.4.5}$$

Substitution of equations 4.4.2 and 4.4.3 now gives:

$$G_{cl}(s) = \frac{Z(s)P_c(s)}{P(s)P_c(s) + Z(s)Z_c(s)} \tag{4.4.6}$$

Comparison with equation 4.4.4 shows that the poles are indeed identical as predicted. The zeros of this alternative closed loop system, however, are composed not of the zeros of the plant and controller, but of the plant zeros and the controller poles.

Determining the gain, or controller transfer function, necessary to place the system poles at the required locations is not a trivial task. Even before this, however, comes the problem of determining suitable pole locations to satisfy a given set of design criteria.

For the design of feedback systems many configurations exist and the challenge is to find the best for particular multivariable systems. Let $G(s)$ a MIMO system with m inputs and p outputs be described in right or left matrix fraction description as follows:

$$G(s) = N_R(s)D_R^{-1}(s) \tag{4.4.7}$$

or

$$G(s) = D_L^{-1}(s)N_L(s) \tag{4.4.8}$$

4.4.3 Unity feedback configuration

The closed loop transfer function of a feedback system as shown in figure 4.1 is given by:

$$y = Gu = G[G_c(r-y)] \Rightarrow y + GG_cy = GG_cr \Rightarrow [I + GG_c]y = GG_cr \Rightarrow G_{cl} = [I + GG_c]^{-1}GG_c$$

$$G_{cl}(s) = [I + G(s)G_c(s)]^{-1}G(s)G_c(s) \tag{4.4.9}$$

From the matrix identity: $A(I + BC)^{-1} = (I + CB)^{-1}A$

We have :

$$G_{cl}(s) = G(s)[I + G_c(s)G(s)]^{-1}G_c(s) \tag{4.4.10}$$

Case 1:

For a $p \times m$ system $G(s)$ described by a RMFD as in equation 4.4.7 the $m \times p$ compensator $G_c(s)$ will be described by a LMFD (see figure 4.3):

$$G_c(s) = D_c^{-1}(s)N_c(s) \tag{4.4.11}$$

where $D_R(s)$, $N_R(s)$, $D_c(s)$ and $N_c(s)$ are matrix polynomials with parameters: D_{Ri}, D_{ci} are $m \times m$ real matrices, N_{Ri} are $p \times m$ real matrices, and N_{ci} are $m \times p$ real matrices.

So replacing equations 4.4.7 and 4.4.11 in 4.4.10 we have :

$$G_{cl} = N_RD_R^{-1}[I + D_c^{-1}N_cN_RD_R^{-1}]^{-1}D_c^{-1}N_c \Rightarrow G_{cl} = N_R[D_cD_R + N_cN_R]^{-1}N_c$$

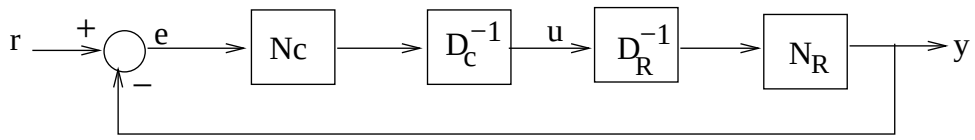


Figure 4.3: Unity feedback: Case 1

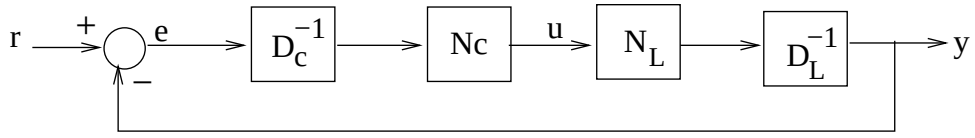


Figure 4.4: Unity feedback: Case 2

Let $D_f(s) = D_c(s)D_R(s) + N_c(s)N_R(s)$ be the "right" Diophantine equation or compensator equation and its poles are the poles of the closed loop system. So the closed loop system is given by:

$$G_{cl}(s) = N_R(s)D_f^{-1}(s)N_c(s) \tag{4.4.12}$$

To determine the compensator which will assign the desired poles (or block poles) we have to solve the Diophantine (compensator) equation.

Case 2:

For a system described in LMFD as in equation 4.4.8 the compensator will be described by a RMFD:

$$G_c(s) = N_c(s)D_c^{-1}(s) \tag{4.4.13}$$

Where, like before, D_L, D_c , N_L and N_c are matrix polynomials whose coefficients are: D_{Li} and D_{ci} $p \times p$ real matrices; N_{Li} are $p \times m$ real matrices and N_{ci} are $m \times p$ real matrices.

The closed loop transfer function of the feedback system of figure 4.4 can be rewritten as :

$$G_{cl}(s) = [I + G(s)G_c(s)]^{-1}G(s)G_c(s) \tag{4.4.14}$$

And replacing equation 4.4.8 and 4.4.13 in 4.4.14 we have:

$$G_{cl} = [I + D_L^{-1}N_L N_c D_c^{-1}]^{-1} D_L^{-1}N_L N_c D_c^{-1}$$

or:

$$G_{cl} = D_c[D_L D_c + N_L N_c]^{-1} N_L N_c D_c^{-1}$$

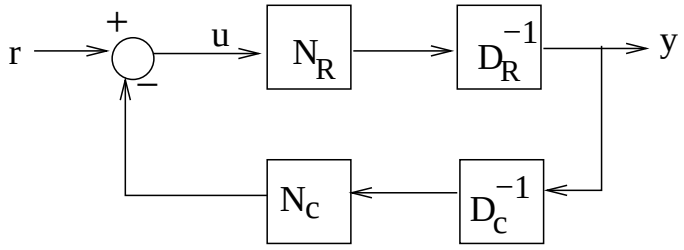


Figure 4.5: Output feedback: Case 1

Let $D_f(s) = D_L(s)D_c(s) + N_L(s)N_c(s)$ be the "left" Diophantine equation. So:

$$G_{cl} = D_c[D_f]^{-1}N_L N_c D_c^{-1} \tag{4.4.15}$$

If we replace $N_L N_c = D_f - D_L D_c$ in the precedent equation then we have:

$$G_{cl} = D_c D_f^{-1}(D_f - D_L D_c)D_c^{-1} \tag{4.4.16}$$

So the closed loop transfer function is given by:

$$G_{cl}(s) = I_q - D_c(s)D_f^{-1}(s)D_L(s) \tag{4.4.17}$$

Like before the poles of D_f are the poles of the closed loop system, and the compensator is determined by solving the Diophantine equation.

4.4.4 Output feedback configuration

In this case the controller is put on the feedback line as in figure 4.2.

Case 1:

For a system described in RMFD as in equation 4.4.7 and the compensator described in LMFD as in equation 4.4.11 the closed loop transfer function of the system of figure4.5 is given by:

$$G_{cl}(s) = G(s)[I + G_c(s)G(s)]^{-1} \tag{4.4.18}$$

with $G(s)$ and $G_c(s)$ matrix polynomial parameters: D_{Ri}, D_{ci} are $m \times m$ real matrices, N_{Ri} are $p \times m$ real matrices, and N_{ci} are $m \times p$ real matrices.

So $G_{cl} = N_R D_R^{-1}[I + D_c^{-1}N_c N_R D_R^{-1}]^{-1}$ or $G_{cl} = N_R[D_c D_R + N_c N_R]^{-1}D_c$

If we let $D_f = D_c D_R + N_c N_R$ be the right Diophantine equation then:

$$G_{cl}(s) = N_R(s)D_f^{-1}(s)D_c(s) \tag{4.4.19}$$

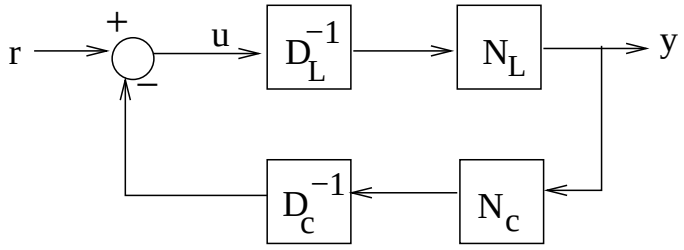


Figure 4.6: Output feedback: Case2

And the poles of D_f are the poles of the closed loop system, and as before the compensator is fully determined by solving the Diophantine equation.

Case 2:

For a system described by LMFD as in 4.4.8, the compensator is described by a RMFD as in 4.4.13, and the closed loop system of figure 4.6 will be rewritten as :

$$G_{cl}(s) = [I + G(s)G_c(s)]^{-1}G(s) \tag{4.4.20}$$

where the matrix parameters are: D_{Li}, D_{ci} are $p \times p$ real matrices, N_{Li} are $p \times m$ real matrices and N_{ci} are $m \times p$ real matrices.

So $G_{cl} = [I + D_c^{-1}N_cN_LD_L^{-1}]^{-1}N_LD_c^{-1}$ or $G_{cl} = D_c[D_LD_c + N_LN_c]^{-1}N_L$

Let $D_f(s) = D_L(s)D_c(s) + N_L(s)N_c(s)$ be the left Diophantine equation then :

$$G_{cl}(s) = N_L(s)D_f^{-1}(s)D_c(s) \tag{4.4.21}$$

The poles of the closed loop system are fully defined by the poles of D_f and the compensator can be determined.

4.4.5 Input-Output feedback configuration

Here a compensator is placed on the feedback path and another takes its inputs from the input references (figure 4.7).

Case1:

Let $G(s)$ described in RMFD as in equation 4.4.7, then the compensators are LMFD such that:

$$G_{c0}(s) = D_c^{-1}(s)L(s) \text{ and } G_{c1}(s) = D_c^{-1}(s)M(s) \tag{4.4.22}$$

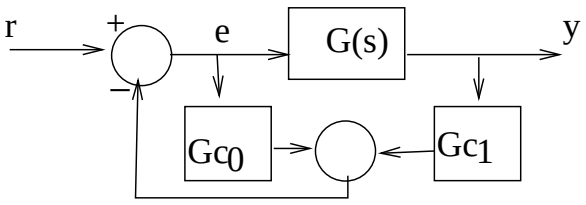


Figure 4.7: Input-output feedback configuration

From figure 4.7 we have the following developments:

$$\left.\begin{array}{c} y = G(s)e \\ e = r - (G_{c0}(s)e + G_{c1}(s)y) \end{array}\right\} \Rightarrow e + G_{c0}e = r - G_{c1}Ge \Rightarrow e[I + G_{c0} + G_{c1}G] = r$$

then $e = [I + G_{c0} + G_{c1}G]^{-1}r \Rightarrow y = G[I + G_{c0} + G_{c1}G]^{-1}r$

The closed loop transfer function will thus be given by:

$$G_{cl}(s) = G(s)[I_p + G_{c0}(s) + G_{c1}(s)G(s)]^{-1} \tag{4.4.23}$$

If we replace equation 4.4.7 and equation 4.4.22 in 4.4.23 then the closed loop system transfer function will be:

$$G_{cl} = N_R D_R^{-1}[I_p + D_c^{-1}L + D_c^{-1}MN_R D_R^{-1}]^{-1} \tag{4.4.24}$$

where: D_{Ri}, D_{ci} and L_i are $m \times m$ real matrices, N_{Ri} are $p \times m$ real matrices and M_i are $m \times p$ real matrices.

Or

$$G_{cl} = N_R[D_c D_R + LD_R + MN_R]^{-1}D_c \tag{4.4.25}$$

Let

$$D_f(s) = D_c(s)D_R(s) + L(s)D_R(s) + M(s)N_R(s) \tag{4.4.26}$$

The poles of the closed loop system are fully defined by the poles of D_f.

Let

$$E(s) = D_f(s) - D_c(s)D_R(s) = L(s)D_R(s) + M(s)N_R(s) \tag{4.4.27}$$

The second part of the equation is a "right" Diophantine equation which solution will determine the compensator numerators.

In order to determine the compensators, the compensators denominators must be the same (D_c) and may be assigned. Then the Diophantine equation can be resolved to determine $M(s)$ and $L(s)$.

So the closed loop system will be:

$$G_{cl}(s) = N_R(s)D_f^{-1}(s)D_c(s) \tag{4.4.28}$$

Case2:

Let us have the same feedback configuration as in figure 4.7. If $G(s)$ is in LMFD (equation 4.4.8) and the two compensators are in RMFD:

$$G_{c0}(s) = L(s)D_c^{-1}(s) \text{ and } G_{c1}(s) = M(s)D_c^{-1}(s) \tag{4.4.29}$$

Then the following development is obtained:

$$\left.\begin{array}{c} y = G(s)e \\ e = r - G_{c0}(s)e - G_{c1}(s)y \end{array}\right\} \Rightarrow e[I + G_{c0}] = r - G_{c1}y \Rightarrow Ge[I + G_{c0}] = G(r - G_{c1}y)$$

or $y[I + G_{c0} + GG_{c1}] = Gr$

then the closed loop transfer function will be given by:

$$G_{cl}(s) = [I + G_{c0}(s) + G(s)G_{c1}(s)]^{-1}G(s) \tag{4.4.30}$$

So if we replace equation 4.4.7 and equation 4.4.29 in equation 4.4.30 then we get:

$$G_{cl} = [I + LD_c^{-1} + D_L^{-1}N_L M D_c^{-1}]^{-1}D_L^{-1}N_L$$

where: D_{Li}, D_{ci} and L_i are $p \times p$ real matrices, N_{Li} are $p \times m$ real matrices and M_i are $m \times p$ real matrices.

Or:

$$G_{cl} = D_c[D_L D_c + D_L L + N_L M]^{-1}N_L \tag{4.4.31}$$

Let us define $D_f(s) = D_L(s)D_c(s) + D_L(s)L(s) + N_L(s)M(s)$ as the desired closed loop denominator and let

$$E(s) = D_f(s) - D_L(s)D_c(s) = D_L(s)L(s) + N_L(s)M(s) \tag{4.4.32}$$

As before, the second part of the equation is the compensator equation which can be solved if the compensator denominator is fixed arbitrarily.

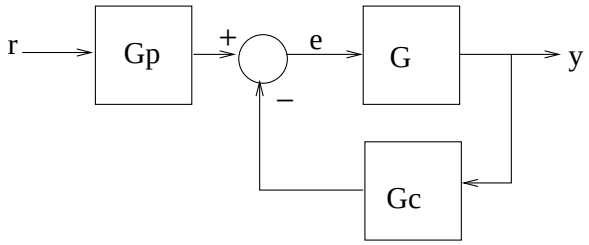


Figure 4.8: Feedback compensation with a pre-compensator

4.4.6 Using a pre-compensator

Although the problem of pole placement has received enormous attention, the problem of zeros placements has been relatively neglected. The latter, however, could be of substantial importance in some design problems, such as actuators and sensors positioning as in [39] and closed loop system transient response as in [40].

In general, zeros are placed via a pre-compensator. Pre-compensators are used in multivariable control systems to reduce (or eliminate) open loop system interactions. Classical methods for the design of pre-compensator are traditionally based on static designs. Static pre-compensators are preferred for their simplicity, but in many applications, only the more powerful dynamic pre-compensators are able to deliver the desired behaviour.

In [41] the author proposes a new method for the design of dynamic pre-compensators based on a Quadratic Programming (QP) optimization to achieve diagonal dominance for decoupling. In [42] a pre-compensator is designed to minimize the eigenvector matrix condition number and a defined measure of normality in order to apply the characteristic locus method effectively.

In figure 4.8 an example of a feedback configuration with a dynamic pre-compensator to place eventual desired zeros, where $G_p(s)$ is $p \times q$, $G(s)$ is $q \times p$ and $G_c(s)$ is $p \times q$.

Case1:

If $G(s)$ is in RMFD so $G_p(s)$ is in LMFD then the closed loop system will be as follows: $y = Ge = G[G_p r - G_c y]$ then

$$G_{cl} = GG_p[I + GG_c]^{-1} = G[I + G_c G]^{-1} G_p \tag{4.4.33}$$

If we replace equation 4.4.7 and equation 4.4.11 in equation 4.4.33 then the closed loop

transfer function will be:

$$G_{cl} = N_R D_R^{-1}[I + D_c^{-1} N_c N D_R^{-1}]^{-1} G_p$$

where: D_{Ri}, D_{ci} and K_i are $m \times m$ real matrices and N_{Ri} are $p \times m$ real matrices.

Or

$$G_{cl} = N_R[D_c D_R + N_c N_R]^{-1} D_c G_p \tag{4.4.34}$$

If $D_f(s) = D_c(s)D_R(s) + N_c(s)N_R(s)$ then the closed loop transfer function will be:

$$G_{cl}(s) = N_R(s) D_f^{-1}(s) D_c(s) G_p(s) \tag{4.4.35}$$

Case2:

If $G(s)$ is in LMFD (as in equation 4.4.8) so $G_c(s)$ is in RMFD (as in equation 4.4.13) then the closed loop system will be as follows: $y = Ge = G[G_p r - G_c y]$ then

$$G_{cl} = [I + GG_c]^{-1} GG_p \tag{4.4.36}$$

If we replace equation 4.4.8 and equation 4.4.13 in equation 4.4.36 then the closed loop system transfer function will be:

$$G_{cl} = [I + D_L^{-1} N_L N_c D_c^{-1}]^{-1} D_L^{-1} N_L G_p$$

where: D_{Li}, D_{ci} are $p \times p$ real matrices, N_{Li} are $p \times m$ real matrices and G_{pi} are $m \times m$ real matrices.

So

$$G_{cl} = D_c[D_L D_c + N_L N_c]^{-1} N_L G_p \tag{4.4.37}$$

If $D_f(s) = D_L(s)D_c(s) + N_L(s)N_c(s)$ then:

$$G_{cl}(s) = D_c(s) D_f^{-1}(s) N_L(s) G_p(s) \tag{4.4.38}$$

Remark 4.4.1. From equations 4.4.35 and 4.4.38 we conclude that the poles of D_f are the poles of the closed loop system and the zeros of the closed loop system will be constituted by the zeros of the plant (that cannot be moved), the poles of the compensator denominator, and the zeros of the pre-compensator that we can chose in order to achieve a certain goal.

4.5 Conclusion

In this chapter a review of feedback control techniques is presented for MIMO systems described in state space equations or in matrix fractions. Static and dynamic compensation and examples of feedback configurations are presented which lead to the Diophantine equation. Techniques to solve this compensator equation are detailed in appendix A with a small contribution for a better performing resolution process.

In the next chapter, the proposed design method will be presented. This method is based on systems described in MFD, so that dynamic compensators are simple to design, then a desired eigenstructure (and even more) can be assigned using block poles placement, finally desired block zeros can be placed simultaneously using a compensator and a pre-compensator.

The choice of desired eigenvalues depends highly on the performance criteria such as the rise time, settling time, overshoot, largest magnitude of the actuating signals and so forth [31]. One way to proceed is by computer simulation, or by minimizing a quadratic performance equation, and this problem is not considered in this thesis but is a natural further step in the research undertaken.

Chapter Bibliography

[1] H.H. Rosenbrock, "Distinctive problems of process control", Chem. Eng. Prog., Vol. 58, pp. 43-50, 1962.

[2] B. A. White, "Eigenstructure Assignment: A Survey", in Proceedings of the I Mech E Part I: Journal of Systems and Control Engineering, Vol. 209, pp. 1-11.1995.

[3] W. M. Wonham, "On pole assignment in multi-input controllable linear systems", IEEE Trans. Autom. Contr., Vol. AC-12, pp. 660-665, 1967.

[4] E. J. Davison, "On Pole Assignment in Linear Systems with Incomplete State Feedback", IEEE Trans. Autom. Contr., Vol. AC-15, pp. 348-351, 1970.

[5] J. B. Song and Y. Ishida, "The pole placement for linear multivariable system using the single feedback controller", 34_{th} SICE Conference, Japan, 1995.

[6] G. S. Miminis and C. C. Paige, "A direct algorithm for pole assignment of time invariant multi-input linear systems using state feedbacdk", Automatica, Vol.24, pp. 343-356, 1988.

[7] D. Baksi, K.B. Datta and G.D. Roy, "Parallel algorithm for pole assignemtn of multivariable input system", IEE-proc. Contr. Theo. App., pp. 367-372, 1994.

[8] W. C. Haas and J. K. Weinhofer, "Multivariable compensator design using polynomial matrices and interval arithmetic", in Proc. of the IEEE Int. Symposium on Computer-Aided Control System Design, Dearborn, MI, USA, 1996.

[9] A. I. G. Vardulakis and C. Kazantzidou, "Denominator assignment, invariants and canonical forms under dynamic feedback compensation in linear multivariable systems", in Proc. of the 17th Mediterranean Conference on Control and Automation, Greece, pp. 336-341, 2009.

[10] P. Hippe and J. O'Reilly, "Parametric compensator design", Int. J. Control, Vol. 45, No.4, pp. 1455-1468, 1987.

[11] T. Kailath, Linear systems, Englewood Cliffs, Prentice-Hall, NJ, USA, 1980.

[12] P. Stefanidis, A. P. Paplinski and M. J. Gibbard, Numerical operations with polynomial matrices: application to multivariable dynamic compensator design, Lectures notes in control and information sciences, Vol. 171, Springer-Verlag, Berlin, 1992.

[13] B. Shafai and L. H. Keel, "New pole placement algorithm: polynomial matrix approach", in Proc. of 9^{th} American Control Conference, SanDiego, Vol. 2, pp. 1684-1685, 1990.

[14] L. S. Shieh, Y. T. Tsay and R.E. Yates, "State-feedback decomposition of multivariable systems via block-pole placement", IEEE Trans. Autom. Contr., Vol. AC-28, No. 8, pp. 850-852, 1983.

[15] B. D. O. Anderson and V. Kucera, "Matrix fraction construction of linear compensators", IEEE Trans. Autom. Contr., Vol. AC-30, No. 11, pp. 1112-1114, 1985.

[16] J. D. Wang, Y. T. Juang, "A new approach for computing the state feedback gains of multivariable systems", IEEE Trans. Autom. Contr., Vol. AC-40, No.10, pp. 1823-1826, 1995.

[17] K. M. Sobel and E. J. Lallman, "Eigenstructure assignment for the control of highly augmented aircraft", J. of Guidance, Control, and Dynamics, Vol. 12, No.3, pp. 18-324, 1989.

[18] T. Clarke, S.J. Griffin and J. Ensor, "Output feedback eigenstructure assignment using a new reduced orthogonality condition", Int. J. Contr., Vol. 76, No.4, pp. 1-13, 2002.

[19] A. N. Andry, E. Y. Shapiro and J. C Chung, "Eigenstructure assignment for linear systems", IEEE Trans. on Aerospace and Electronic Systems, Vol. 19, No. 5, pp. 711-727, 1983.

[20] K. M. Sobel, E.Y. Shapiro and A. N. Andry, "Eigenstructure assignment", Int. J. Contr., Vol. 59, No. 1, pp. 13-37, 1994.

[21] J. Farineau, "Lateral Electric Flight Control Laws of Civil Aircraft based on Eigenstructure Assignment Technique," Paper 89-3594 in AIAA Guidance Navigation and Control Conference, Boston, MA, pp.1-15, August 1989.

[22] D. M. Littleboy, Numerical Techniques for Eigenstructure Assignment by Output Feedback in Aircraft Applications, PhD thesis, University of reading, UK, 1994.

[23] B. C. Moore, "On the flexibility offered by state feedback in multivariable systems beyond closed loop eigenvalue assignment", IEEE Trans. Automat. Contr., Vol. AC-21, No.5, pp. 689-692, 1976.

[24] C. A. Harvey and G. Stein, "Quadratic weights for asymptotic regulator properties", IEEE Trans. Automat. Contr., Vol. 23, No.3, pp. 378-387, 1978.

[25] K. Yang and R. Orsi, "Pole placement via output feedback: a methodology based on projections", in Proc. of the 16^{th} IFAC World Congress, Praha, Czech Republic, Vol.16, No.1, pp. 534-539, 2005.

[26] H. Kimura, "Pole Assignment by Gain Output Feedback", IEEE Trans. Autom. Contr., Vol. AC-20, No.8, pp. 509-515, 1975.

[27] O. Bachelier, J. Bosche and D. Mehdi, "On pole placement via eigenstructure assignment approach", IEEE Trans. Autom. Contr., Vol. AC-51, No.9, pp. 1554-1558, 2006.

[28] T. Clarke, S.J. Griffin and J. Ensor, "Output feedback eigenstructure assignment using a new reduced orthogonality condition", Int .J. Contr., Vol.76, No.4, pp. 390-402, 2003.

[29] A. J. Pomfret, Eigenstructure assignment for helicopter flight control, PhD thesis, university of York, UK, 2006.

[30] J.-F. Magni, "Multi-model eigenstructure assignment in flight-control design", Aerosp. Sci. Technol., Vol. 3, pp. 141-151, 1999.

[31] C. T. Chen, Linear system theory and design, Holt, Rinehart and Winston, New York, 1984.

[32] A. Pomfret and T. Clarke, "An extension to output-feedback eigenstructure assignment: Structuring controllers by exploiting unused design freedom", Int. J. Contr., Vol. 82, No.2, pp. 207-216, 2009.

[33] S. Güney and A. Atasoy, "An Approach to Pole Placement Method with Output Feedback", UKACC Control, Manchester, UK, 2008.

[34] F. Kessal and K. Hariche, "On the choice of closed loop block poles in multi-variable control design", in Proc. of Applications of Large scale industrial systems, Silija Finland, Vol. 1, No. 1, pp. 59-63, 2006.

[35] H. Loubar, Mutlivariable control system design via solvent placement, Magister thesis, INELEC, Boumerdes, 1998.

[36] R. Davies, Robust eigenstructure assignment for flight control applications, PhD thesis, University of York, UK, 1994.

[37] S. Srinathkumar, "Eigenvalue/Eigenvector Assignment Using Output Feedback", IEEE Trans. Autom. Contr., Vol. AC-23, No.1, pp. 79-81, 1978.

[38] C. C. Tsui, Robust Control system Design: Advanced state-space techniques, Marcel Dekker Inc., NY, 1996.

[39] P. T. Kabamba, S. M. Meerkov and E. K. Poh, "H2-Optimal zeros", IEEE Trans. Automat. Contr., Vol. AC-39, No.6, pp. 1298-1301, 1994.

[40] C. C. Hang, "The choice of controller Zeros", IEEE Contr. Syst. Mag., pp. 72-75, 1989.

[41] L. Shamgah, A. Nejati, A. Nobakhti and H. Karimi, "Design of low order dynamic pre-compensators using convex methods, in Proc. American Control Conference, USA, pp. 3680-3685, 2011.

[42] J. C. Basilio and J.A. Sahate, "A normalizing precompensator for the design of effective and reliable commutative controllers, Int. J. Contr., Vol. 73, No.14, pp. 1280-1297, 2000.

Chapter 5

Eigenstructure assignment using block poles placement

5.1 Introduction

The important identities obtained in chapter 3 open up some interesting possibilities. The link between an SSD eigenstructure and that of an MFD opens an issue for a direct way of combining traditional eigenstructure assignment objectives with polynomial methods, moving beyond traditional fixed state or output feedback gains, but enabling dynamic compensators to be incorporated into the controller structure in a very natural way.

We have developed an approach that allows us to design dynamic compensators and pre-compensators that place block poles and block zeros. This enables latent structure, hence eigenstructure, assignment possible, not only for poles but for zeros as well. The approach can therefore be used to improve the behaviour of MIMO systems and resolving control problems such as: sensitivity, robustness, decoupling, and disturbance rejection.

The use of dynamic compensation allows additional degrees of freedom in the design process, enabling a designer to achieve closer matches to a closed loop time domain specification than in using simple gain output feedback of conventional eigenstructure assignment. This contribution has been published in [1].

From chapter 4 we can conclude that transfer function analysis described in MFD is a better choice to design a feedback compensator to achieve assignment of both eigenvalues and eigenvectors of the closed loop system, and, to the author knowledge, using block poles and block zeros placement to achieve this EA has not yet been done.

In [2] desired poles have been used to construct block poles in different canonical forms (diagonal form, controller form and observer form) in order to achieve some desired specifications

(small settling time, better time response, less sensitivity etc.). In our case block poles are constructed from a desired set of latent values and latent vectors, so the form is of no-concern here.

5.2 Introduction to the contribution

The main contribution of the thesis is a new method based on MFD and matrix polynomials to achieve eigenstructure assignment by block poles placement using a dynamic compensator. For a system described in SSD, the first step consists in converting it to MFD, then converting the desired eigenstructure into a latent structure, and construct desired block poles. A desired denominator of the closed loop system is thus obtained and the compensator to achieve this placement is computed by solving a Diophantine equation.

So for an m-input, p-output, n-state system with a controllability index (or observability index) equal r, the proposed method can place r block poles of dimension $m \times m$ (or $p \times p$), thus assigning n eigenvalues and latent vectors (right or left) using a static compensator. But more important , we can determine a dynamic compensator of degree $l \geq 1$ by placing a number $r + l$ block poles, thus assigning a number $n + l * m$ (or $n + l * p$) of eigenvalues and their corresponding latent vectors, as long as the following conditions are satisfied:

(i) the system is block controllable (or block observable),

(ii) there exist in the set of $n + l * m$ (or $n + l * p$) latent vectors, r subsets of linearly independent vectors,

(iii) the conditions for the resolution of the obtained Diophantine equation are met,

(iv) and finally the computed compensator is stable.

If the computed compensator is not stable, its degree is increased by increasing the number of the desired eigenvalues and its corresponding latent vectors.

However, the issue of minimal compensator degree is less important, in that increased degree provides additional useful degrees of design freedom beyond arbitrary eigenvalue assignment [3].

The design process offers a larger degree of freedom (more than the set of original desired eigenvalues can be assigned) and the algorithm is direct and allows assigning desired eigenstructure through block poles but block zeros can also be placed simultaneously.

Compared to the previous works and to the best of the author' knowledge nobody considered using block poles placement for systems described by matrix fractions to assign an eigenstructure using dynamic compensators.

In the following an algorithm which summarizes the proposed approach to eigenvector assignment is given. All the steps will be detailed in the following sections, and assuming that the most of the theoretical notions and literature review has been done in the precedent chapters.

Let a system be described in state space equations as follows:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Eu \end{cases} \tag{5.2.1}$$

Where A is $n \times n$, B is $n \times m$, C is $p \times n$ and E is $p \times m$ (but considered null for more clarity). And we suppose that a set of desired eigenvectors and eigenvectors have already been specified to achieve a certain requirement.

Proposed Algorithm

To assign a set of desired eigenvalues and a set of corresponding eigenvectors using feedback compensation, the following steps are proposed:

Step1: Convert the SSD system into a RMFD or LMFD system, depending if it is block controllable or block observable.

Step2: Convert the desired eigenstructure into a latent structure.

Step3: Construct the desired block poles (block zeros).

Step4: Construct the desired matrix polynomial (denominator or numerator of the closed loop system).

Step5: Choose the fixed denominator of the compensation system arbitrarily or as to meet some desired specification if an input-output feedback configuration is used.

Step6: Solve the Diophantine equation to determine the numerators of the two compensators.

Remark 5.2.1. The compensation configuration chosen is the input-output feedback system which has been presented in Chapter 4, and an illustrative example will be given throughout the steps for more clarity. Systems described in matrix transfer functions or directly in matrix fractions can be treated, but for the sake of generality, the design process is applied for systems in SSD, to include the, possibly needed, conversion.

5.3 Proposed approach

In this section the first steps of the design process are given, then in the following section the design of the compensator is presented.

5.3.1 Conversion methods

The first step is to convert the system described in SSD to a system in MFD as explained in chapter 3 section 3.5. If the system in 5.2.1 verifies the conditions, then it can be transformed into a right MFD or a left MFD as follows.

1. System description:

If the system is block controllable of degree μ then the block controller form will be as follows:

$$A_c = \begin{pmatrix} 0_m & I_m & 0_m & \cdots & 0_m \\ 0_m & 0_m & I_m & \cdots & 0_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0_m & 0_m & 0_m & \cdots & I_m \\ -A_0 & -A_1 & -A_2 & \cdots & -A_{\mu-1} \end{pmatrix}; B_c = \begin{pmatrix} 0_m \\ 0_m \\ \vdots \\ 0_m \\ I_m \end{pmatrix}; C_c = \begin{pmatrix} C_0 & C_1 & \cdots & C_{\mu-1} \end{pmatrix} \tag{5.3.1}$$

And the transfer function in right MFD is given by:

$$G(s) = N_R(s) D_R^{-1}(s) \tag{5.3.2}$$

Where

$$N_R(s) = C_{\mu-1} s^{\mu-1} + C_{\mu-2} s^{\mu-2} + \cdots + C_1 s + C_0 \tag{5.3.3}$$

$$D_R(s) = I_m s^r + A_{\mu-1} s^{\mu-1} + \cdots + A_1 s + A_0 \tag{5.3.4}$$

And if the system is block observable of degree ν then the block observer form will be as follows:

$$A_o = \begin{pmatrix} 0_p & 0_p & \cdots & 0_p & -A_0 \\ I_p & 0_p & \cdots & 0_p & -A_1 \\ 0_p & I_p & \cdots & 0_p & -A_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_p & 0_p & \cdots & I_p & -A_{\nu-1} \end{pmatrix}; B_o = \begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ \vdots \\ B_{\nu-1} \end{pmatrix}; C_o = \begin{pmatrix} 0_p & \cdots & 0_p & I_p \end{pmatrix} \quad (5.3.5)$$

and the transfer function in left MFD is given by:

$$G(s) = D_L^{-1}(s) N_L(s) \quad (5.3.6)$$

Where

$$D_L(s) = I_p s^\nu + A_{\nu-1} s^{\nu-1} + \cdots + A_1 s + A_0 \quad (5.3.7)$$

$$N_L(s) = B_{\nu-1} s^{\nu-1} + B_{\nu-2} s^{\nu-2} + \cdots + B_1 s + B_0 \quad (5.3.8)$$

2. Example:

The system treated by the example and its desired eigenstructure is taken from ((Ensor 2000, pages146-147)[4], but slightly modified to have a block controllable system.

Let a 2 input 2 output system be described by state space equations as in equation 5.2.1 where:

$$A = \begin{pmatrix} 1 & 2 & -3 & 5 \\ 0 & 3 & -1 & 7 \\ 5 & 8 & 1 & -9 \\ 2 & 6 & 3 & 8 \end{pmatrix}; B = \begin{pmatrix} 1 & 0 \\ 2 & 3 \\ 9 & -2 \\ 5 & 2 \end{pmatrix}; C = \begin{pmatrix} 7 & 3 & 0 & 2 \\ 1 & -1 & 0 & 1 \end{pmatrix}$$

We can verify that the system is block controllable and block observable. For this case we chose to transform it in a matrix transfer function in RMFD using equations 5.3.3 and 5.3.4.

We obtain the following: $G(s) = N(s)D^{-1}(s)$ with $D(s) = I_2 s^2 + D_1 s + D_0$ and $N(s) = N_1 s + N_0$ where:

$$D_1 = \begin{pmatrix} -4.4369 & -2.3091 \\ -25.4220 & -8.5631 \end{pmatrix}; D_0 = \begin{pmatrix} 55.5957 & -4.6843 \\ -3.8866 & 10.1124 \end{pmatrix}$$

And

$$N_1 = \begin{pmatrix} 23 & 13 \\ 4 & -1 \end{pmatrix}; N_0 = \begin{pmatrix} -153.5351 & 120.5706 \\ 59.6745 & 24.3268 \end{pmatrix}$$

3. Eigenstructure:

The second step is to convert the desired eigenstructure into a desired latent structure using the method developed in chapter 3 section 3.6. Instead of assigning an eigenstructure, we will assign its corresponding latent structure (latent values and corresponding latent vectors).

For each eigenvector $\tilde{v}_i$ $(i = 1..n)$ its corresponding latent vector is:

$$v_i = T_{c1}\tilde{v}_i \tag{5.3.9}$$

where T_{c1} is given by equation 3.3.13 of chapter 3.

By duality, for each left eigenvector $\tilde{w}_i$ $(i = 1..n)$, we can compute its corresponding left latent vector:

$$w_i = \tilde{w}_i T_{o1} \tag{5.3.10}$$

where T_{o1} is given by equation 3.3.16 of chapter 3.

Remark 5.3.1. Only single eigenvalues (latent values) are considered. The condition for solvents to exist is the possibility to find from a set of n latent vectors, subsets of p (or m) linearly independent vectors, so multiplicity of the latent values is not a problem here.

4. Example:

The following is the desired eigenstructure [4]:

Desired eigenvalues: $d\lambda = \{-1 - 3 - 5 - 6\}$ then $\Lambda = diag\{-1, -3, -5, -6\}$

Desired eigenvectors (right): $d\tilde{V} = \begin{pmatrix} 0.707 & 0.707 & 0 & 0 \\ 0.707 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.707 \\ 0 & 0.707 & 0 & 0.707 \end{pmatrix}$ and $d\tilde{W} = dV^{-1}$.

From this desired eigenstructure a desired state matrix is constructed ($dA = d\tilde{V}\Lambda d\tilde{W}$), then a desired latent structure (to construct desired block poles) is obtained using equation 5.3.9 where T_{c1} is computed using the desired state matrix and the input matrix B.

Desired latent values: $d\lambda = \{-1 - 3 - 5 - 6\}$

And the desired latent vectors: $dV = \begin{pmatrix} -0.1713 & 0.2383 & 0.4678 & 0.0352 \\ -0.2680 & 0.2930 & 0.6578 & 0.0664 \end{pmatrix}$

5.3.2 Construction of desired block roots

The proposed process consists in assigning a desired latent structure by placing block poles or block zeros. So the third step consists in constructing a block root (a block pole or a block

zero) from a desired latent structure using the method proposed in chapter 2 section 2.6.

For a set of $m \times \mu$ (m, μ integers) desired latent roots (λ_1,...,$\lambda_{m\mu}$) and a set of corresponding desired latent vectors (left or right) (v_1,...,$v_{m\mu}$) we can construct μ block roots if there exist μ groups of m linearly independent latent vectors.

Let an i^{th} set ($i = 1..\mu$) of m linearly independent latent vectors be: (v_{i1},...,v_{im}) and its corresponding latent values:(λ_{i1},...,λ_{im}) then we can determine a right block root using the following equation:

$$R_i = \begin{pmatrix} v_{i1} & \cdots & v_{im} \end{pmatrix} \begin{pmatrix} \lambda_{i1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_{im} \end{pmatrix} \begin{pmatrix} v_{i1} & \cdots & v_{im} \end{pmatrix}^{-1} \tag{5.3.11}$$

Let a j^{th} set ($j = 1..\nu$) of p linearly independent left latent 'row' vectors be: (w_{j1},...,w_{jp}) and its corresponding latent values: (λ_{j1},...,λ_{jp}) then we can determine a left block root using the following equation:

$$L_j = \begin{pmatrix} w_{j1} \\ \vdots \\ w_{jp} \end{pmatrix}^{-1} \begin{pmatrix} \lambda_{j1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_{jp} \end{pmatrix} \begin{pmatrix} w_{j1} \\ \vdots \\ w_{jp} \end{pmatrix} \tag{5.3.12}$$

Example:

From the latent structure obtained previously, desired "right" block roots (poles) are constructed using equation 5.3.11.

With latent values: (-1 -3) and its corresponding latent vectors we get:

$dR_1 = \begin{pmatrix} 12.9285 & -11.3285 \\ 19.5842 & -16.8285 \end{pmatrix}$

And with latent values : (-5 -6) and its corresponding vectors we have:

$dR_2 = \begin{pmatrix} -0.1768 & -3.0829 \\ 9.1105 & -10.8232 \end{pmatrix}$

5.3.3 Construction of a desired matrix polynomial

From the desired block poles we can construct the denominator of the closed loop system using the method presented in chapter 2 section 2.3.6, which represents the fourth step.

From a set of μ $m \times m$ matrices R_i ($i = 1..\mu$) we can construct a right matrix polynomial of degree μ:

$$I_m s^\mu + D_{\mu-1} s^{\mu-1} + \cdots + D_1 s + D_0 \tag{5.3.13}$$

using the following equation:

$$\begin{pmatrix} D_0 & D_1 & \cdots & D_{\mu-1} \end{pmatrix} = -\begin{pmatrix} R_1^{\mu} & R_2^{\mu} & \cdots & R_{\mu}^{\mu} \end{pmatrix} V_R^{-1} \tag{5.3.14}$$

where V_R is the 'right' block Vandermonde matrix of degree μ defined in chapter 2 section 2.3.4.

From a set of ν $p \times p$ matrices L_j $(j = 1..\nu)$ we can construct a left matrix polynomial of degree ν as in equation 5.3.13 using the following equation:

$$\begin{pmatrix} D_0 \\ D_1 \\ \vdots \\ D_{\nu-1} \end{pmatrix} = -V_L^{-1} \begin{pmatrix} L_1^{\nu} \\ L_2^{\nu} \\ \vdots \\ L_{\nu}^{\nu} \end{pmatrix} \tag{5.3.15}$$

where V_L is a 'left' block Vandermonde matrix of order ν.

In the following, an important theorem which states the condition to satisfy in order to achieve the compensator design.

Theorem 5.3.1. *The set of μ, $m \times m$, matrices R_i (or ν, $p \times p$ matrices L_i) are block roots (solvents) of the matrix polynomial as defined in equation 5.3.13 if and only if the right (left) block Vandermonde matrix constructed from these matrices is non-singular.*

Proof The proof is straight forward from the fact that: A block Vandermonde matrix constructed from a full set of solvents of a particular matrix polynomial is non singular [5].

Remark 5.3.2. The singularity or non-singularity of the block Vandermonde matrix, therefore, provides an excellent check as to the reasonableness of the chosen latent vectors.

The matrix polynomial, thus obtained, will be the desired denominator of the closed loop system. If the block roots constructed represents desired block zeros of the closed loop system then the matrix polynomial thus constructed is the numerator of a pre-compensator.

Example:

With two solvents, as constructed previously, we can construct a matrix polynomial of degree 2 using equation 5.3.14:

$D_{cl} = I_2 s^2 + dD_1 s + dD_0$ where

$$dD_1 = \begin{pmatrix} -6.8232 & 11.0829 \\ -19.1105 & 21.8232 \end{pmatrix}; dD_0 = \begin{pmatrix} -74.1215 & 65.0055 \\ -101.9834 & 88.2265 \end{pmatrix}$$

5.4 Dynamic compensator design

The final step of the design process is to determine a compensation system to assign the desired denominator obtained in the previous section for a p outputs m inputs system $G(s)$.

5.4.1 Choice of the feedback configuration

In general, output feedback compensation is used for pole placement, and feedforward compensation (unity feedback) is used for better sensitivity (see chapter 4).

The input output feedback is generally used in the presence of non observable states [6]. In this chapter we preferred to detail the input-output feedback configuration, because it is general, and the denominator of the two compensators is fixed and may be chosen arbitrarily.

Remark 5.4.1. The roots of the fixed denominator of the compensator become the zeros of the closed loop system, so it can be used to place desired zeros.

5.4.2 Compensator equations

The following is a recall of the notions detailed in chapter 4 section 4.4.5. In the input-output feedback configuration a compensator is placed on the feedback path and another takes its inputs from the input references (figure 5.1).

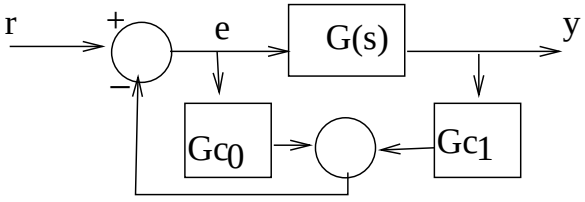


Figure 5.1: Input-Output feedback configuration

The closed loop transfer function will be given by [7]:

$$G_{cl}(s) = G(s)[I_m + G_{c0}(s) + G_{c1}(s)G(s)]^{-1} \tag{5.4.1}$$

Let $G(s)$ be described in RMFD as : $G(s) = N_R(s)D_R^{-1}(s)$ With N_R and D_R are matrix polynomials with N_{Ri} and D_{Ri} $(i = 1..\mu)$ their matrix coefficients.

Then the compensators are in LMFD such that: $G_{c0}(s) = D_c^{-1}(s)L_c(s)$ and $G_{c1}(s) = D_c^{-1}(s)M_c(s)$ where D_c, L_c, M_c are respectively $m \times m$, $m \times m$ and $m \times p$ matrix polynomials with Mc_i, Dc_i and Lc_i their matrix coefficients (figure 5.2).

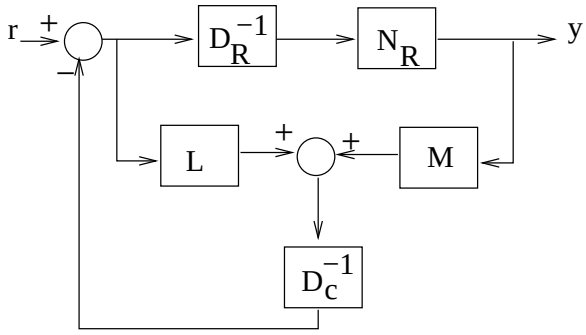


Figure 5.2: Detailed Input-Output feedback configuration

Then the closed loop system will be given by:

$$G_{cl} = N_R D_R^{-1}[I_m + D_c^{-1}L_c + D_c^{-1}M_c N_R D_R^{-1}]^{-1} \tag{5.4.2}$$

Where D_{Ri}, Dc_i and Lc_i are $m \times m$ real matrices, N_{Ri} are $p \times m$ real matrices and Mc_i are $m \times p$ real matrices.

Then:

$$G_{cl} = N_R[D_c D_R + L_c D_R + M_c N_R]^{-1} D_c \tag{5.4.3}$$

Let D_f be the closed loop desired denominator given by:

$$D_f(s) = D_c(s)D_R(s) + L_c(s)D_R(s) + M_c(s)N_R(s) \tag{5.4.4}$$

then the closed loop system transfer function is:

$$G_{cl}(s) = N_R(s)D_f^{-1}(s)D_c(s) \tag{5.4.5}$$

and its poles are fully defined by the poles of D_f .

The desired matrix polynomial (generated in the previous section) will be a desired closed loop m^{th} order μ^{th} degree denominator D_f.

Let:

$$E(s) = D_f(s) - D_c(s)D_R(s) = L_c(s)D_R(s) + M_c(s)N_R(s) \tag{5.4.6}$$

be a 'right' Diophantine equation. If D_c and D_f are known then the resolution of the Diophantine equation will determine the numerators of the two compensators.

Remark 5.4.2. The system $G(s)$ obtained by transformation verifies the conditions of solvability of the Diophantine equation (see Appendix A) thus obtained: strictly proper, column-reduced and right co-prime.

5.4.3 Degree of the compensator

Let the plant be described in RMFD, where the denominator D_R is a m^{th} order μ^{th} degree matrix polynomial, and the numerator with degree less than μ. To assign n desired latent values and corresponding latent vectors, the desired matrix polynomial (generated in the previous section) will be a desired closed loop m^{th} order n^{th} degree denominator. But equation 5.4.4 imposes that if the degree of the compensator is 1 (at least), then the degree of the closed loop denominator be at least equal to $\mu+ 1$. So the number of latent values that can be assigned with a compensator of degree l is equal to $n + (l * m)$. The least value of the degree would be in this case (μ+1).

To get a higher degree compensator, other solvents can be added with non-dominant latent values (as far as possible in the left half s-plane) in order to not interfere with the behaviour of the closed loop system. A higher degree compensator will allow higher useful degree of design freedom.

- For a desired closed loop denominator matrix polynomial D_f, the degree is determined by the number of desired block poles to assign.
- The zeros of the closed loop system will be fully defined by the zeros of the plant and the latent roots of the compensator denominator constructed from the bock zeros. So we can specify desired zeros which will be assigned to the fixed compensator m^{th} order l^{th} degree denominator D_c.

5.4.4 Example

With a closed loop denominator of degree 2 (constructed previously using only 2 solvents), only a static compensator can be determined. So to determine a dynamic first-degree compensator another solvent is added with arbitrary latent values as large as possible in order to not interfere with the behaviour of the closed loop system. If the first-degree compensator obtained after solving the Diophantine equation is not proper or stable, a second-degree will be computed

using the same method (etc.).

Adding a solvent with latent values: (-30 -31) and unity vectors we have:

$dR_3 = \begin{pmatrix} -30 & 0 \\ 0 & -31 \end{pmatrix}$

Using equation 5.3.14 we construct the desired closed loop denominator with degree 3 using 3 solvents: $D_{cl} = I_2 s^3 + dD_2 s^2 + dD_1 s + dD_0$ where:

$dD_2 = \begin{pmatrix} 23.4315 & 10.5259 \\ -19.4513 & 52.5685 \end{pmatrix}$; $dD_1 = \begin{pmatrix} -269.9112 & 388.1594 \\ -687.2149 & 755.4090 \end{pmatrix}$

and $dD_0 = \begin{pmatrix} -2185.723 & 1917.583 \\ -3110.243 & 2690.390 \end{pmatrix}$

Using the function '`polyeig`' of Matlab we obtained the following latent values and latent vectors of the desired denominator:

$$d\lambda = \begin{pmatrix} -31.0000 & -30.0000 & -1.0000 & -3.0000 & -6.0000 & -5.0000 \end{pmatrix}$$

$$dW = \begin{pmatrix} 0.0000 & -1.0000 & 0.6310 & -0.5796 & 0.4679 & -0.5386 \\ 1.0000 & 0.0000 & 0.7758 & -0.8149 & 0.8838 & -0.8426 \end{pmatrix}$$

Remark 5.4.3. We have verified that the desired solvents are the block roots of the closed loop denominator.

The input output feedback configuration needs to fix the compensator denominator. So let us assign the following block root to D_c: $Dc_0 = \begin{pmatrix} 20 & 0 \\ 0 & 2 \end{pmatrix}$

Then D_c will be equal to: $D_c = I_2 s + Dc_0$.

The resolution of the compensator equation 5.4.6 will lead to the following numerators of the compensator:

$L_c = Lc_0$ where $Lc_0 = \begin{pmatrix} 33.4830 & -42.2452 \\ -29.9267 & -41.5775 \end{pmatrix}$

$M_c = Mc_1 s + Mc_0$ where

$Mc_1 = \begin{pmatrix} 2.5961 & -21.3311 \\ 5.8498 & -24.6619 \end{pmatrix}$ and $Mc_0 = \begin{pmatrix} 26.0177 & -22.2659 \\ 19.6701 & 23.7922 \end{pmatrix}$

The two compensators are: $G_{c0}(s) = D_c^{-1}(s) L_c(s)$ and $G_{c1}(s) = D_c^{-1}(s) M_c(s)$.

5.5 Design of a pre-compensator

In an input-output configuration, the zeros of the closed loop system will be fully defined by the zeros of the plant and the poles of the compensator denominator.

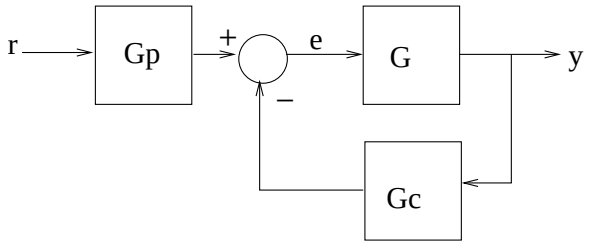


Figure 5.3: Feedback compensation with a pre-compensator

So, if desired zeros (block zeros) are needed to be placed, we can use the compensators denominator. But if these zeros are positive, we need to design a pre-compensator to place them.

In this section, it is shown how to use the proposed method to design a pre-compensator to assign eventual desired block zeros.

5.5.1 Compensator equations:

So the new feedback configuration is shown in figure 5.3 and the closed loop transfer function will be the following:

$$G_{cl}(s) = G(s)[I_p + G_{c0}(s) + G_{c1}(s)G(s)]^{-1}G_p(s) \tag{5.5.1}$$

Let the pre-compensator be described by a LMFD:

$$G_p(s) = D_c^{-1}(s)N_p(s) \tag{5.5.2}$$

where D_c is the denominator of the compensator and the other components of the closed loop system are as defined previously.

The desired zeros or block zeros will be assigned (using equations 5.3.14 or 5.3.15) to the numerator $N_p(s)$.

Then the closed loop transfer function is:

$$G_{cl} = N_R D_R^{-1}[I_p + D_c^{-1}L + D_c^{-1}MN_R D_R^{-1}]^{-1}D_c^{-1})N_p \tag{5.5.3}$$

After some simplifications we get:

$$G_{cl} = N_R[D_c D_R + LD_R + MN_R]^{-1}N_p \tag{5.5.4}$$

Or

$$G_{cl}(s) = N_R(s)D_f^{-1}(s)N_p(s) \tag{5.5.5}$$

where D_f is defined as previously (equation 5.4.4) and the compensator numerators will be determined by resolving the compensator equation as defined in equation 5.4.6.

Remark 5.5.1. If no desired zeros are present, then we have to choose arbitrarily the poles of the compensator (zeros of the system). In general the zeros are chosen in the left half s-plane (not on the imaginary axis) and far from the desired poles, to not annihilate the effect of the poles.

5.5.2 Example

To show how to assign a block zero, the zeros of the closed loop system, which would be obtained if a state feedback controller has been used, are determined. These zeros are chosen to construct a desired block zero: $dZ_0 = \begin{pmatrix} -10.5 & 0 \\ 0 & 2.5 \end{pmatrix}$ which will allow placing a positive zero 10.5 and a negative zero 2.5 through a pre-compensator.

So the numerator of the pre-compensator will be equal to: $N_p(s) = I_2 s + dZ_0$.

5.5.3 Validation

To validate the results obtained for the illustrative example, first we verified that the latent values and vectors of the closed loop denominator computed as in equation 5.4.4 are exactly the same as the latent values and vectors of the desired denominator.

Second, we used the function "`pzmap`" of Matlab on the transfer function matrix of the closed loop system (obtained from equation 5.5.5), we found that the poles are equal to the desired poles (-31,-30, -6,-5,-3 -1) and that the zeros are equal to the zeros of the plant (-3.6333 +11.5123i, -3.6333 -11.5123i), that we cannot replace, and the desired zeros (10.5, -2.5).

5.6 Conclusion

The chapter summarizes a method to assign a desired eigenstrutcure using block poles placement. In this case a transformation from a SSD to a MFD is undertaken for some MIMO systems, then an exact assignment of n (and even more) eigenvalues and its eigenvectors is always possible for these systems as long as the latent vectors composing the block pole are linearly independent. Of course these latent vectors must be specified and chosen.

An MFD can be obtained from any representation so we can also rethink the problem of eigenstructure assignment using the proposed approach described in the chapter. The conditions of the proposed design process make its application limited to a certain class of systems (right/left coprime and column/row reduced) but the similarity transformation, used here, generates directly MFD systems which verify these conditions.

The method has been applied to design a compensator for a helicopter flight control and the results are given in the next chapter.

Chapter Bibliography

[1] M. Yaici and K. Hariche, "On eigenstructure assignment using block poles placement", E. J. CON, Vol. 20, pp. 217-226, 2014.

[2] F. Kessal and K. Hariche, "On the choice of closed loop block poles in multi-variable control design", in Proc. Applications of Large scale industrial systems, Vol. 1, No.1, pp. 59-63, 2006.

[3] P. Hippe and J. O'Reilly, "Parametric Compensator Design", Int. J. Control, Vol. 45, No.4, pp. 1455-1468, 1987.

[4] J. E. Ensor, Subspace Methods for Eigenstructure Assignment, PhD Thesis, University of York, UK, 2000.

[5] J. E. Dennis, J. F. Traub and R. P. Weber, "The algebraic theory of matrix polynomials", Siam J. Numer. Anal., Vol. 13, No.6, pp. 831-845, 1976.

[6] T. Kailath, Linear Systems, Englewood Cliffs, Prentice-Hall, NJ, 1980.

[7] C.T. Chen, Linear System Theory and Design, Holt Rinehart and Winston, NY, 1984.

Chapter 6

Application: Helicopter flight control

6.1 Introduction

In the precedent chapter, a method to design a compensator is given with an illustrative example. This chapter presents a design example for attitude stabilisation of a Lynx helicopter in hover. The example is not intended as a definitive solution, but is to illustrate and validate the proposed method. Helicopters are inherently unstable, very non-linear and highly cross coupled, and thus must be augmented with feedback control to reduce the pilot workload to an acceptable level. A linear description has already been achieved and a desired behaviour (eigenstructure) has been elaborated [1]. The objective is to compute a compensator and eventually a pre-compensator to achieve the assignment of a set of desired eigenvalues and eigenvectors.

The state feedback and the output feedback control of the helicopter using eigenstructure assignment have already been done [1, 2, 3] which will allow comparison.

Although state feedback is not realistic for helicopter control law design but it does have a role to play in the development of the more practical fixed gain output feedback solution. The state feedback solution illustrates the best that one can hope to achieve with output feedback. If an acceptable state feedback solution can not be found then progression to output feedback is pointless and one should instead re-examine the synthesis technique and design objectives [1].

The open loop system, even though complex and large, satisfies the conditions of the proposed method, which justifies the choice of this system to be an application for the proposed method.

6.2 Literature review

All high performance aircraft employ feedback control to achieve the desired level of performance, handling and stability. Helicopters are inherently unstable, highly cross coupled and very non-linear. Hence feedback control is particularly important. Flight control has been thoroughly studied and many published works deal with feedback control of aircraft and a more complex case study would be the helicopter flight control. Flight control based on state feedback, or compensators for systems described in MFD has been studied but the best choice was the one based on eigenstructure assignment [1].

In [4] the questions related to the matrix fraction descriptions (MFDs) and the minimal realizations of the transfer matrix of a spinning satellite system and control system design using output feedback are considered, and a new approach using interval analysis for design of a robust compensator for jet engine is proposed in [5].

An investigation into the application of eigenstructure assignment to aircraft problems is detailed in [6] and confirmed that feedback is used to ensure stability, a satisfactory response and good decoupling in the closed loop system.

The authors of the interesting paper [7] gave the importance of the eigenstructure assignment in linear control systems with an extensive flight control example. They discussed the conditions for the number of reachable eigenvectors, and presented the different techniques to assign them, via full state feedback, output feedback and constrained output feedback.

The eigenstructure assignment flight control design methodology is extended to include dynamic compensator synthesis in [8]. Dynamic compensators are designed via eigenstructure assignment by utilizing a composite system structure and the method has been applied on the lateral dynamics of an L-1011 aircraft.

In [9] a method for computing allowable eigenvector subspaces using singular value decomposition for both real and complex eigenvalues is illustrated and used to design a stability augmentation system for the lateral motion of a light aircraft.

Magni et al., in 1999, presented a control design approach based on eigenstructure assignment by dynamic feedback for a flight control which permits the designer to handle simultaneously robustness against real parameter variations and the use of structured gain [10].

Helicopter flight control is complex and so not frequently used for illustration unless the

helicopter is the subject of the research as in [1, 3]. A new method, based on simple pole-placement considerations, for active control of vibrations in helicopter A129 of Agusta S.P.A and taking into account the specific features of the A129 rotor, has been presented in [11].

The eigenstructure necessary to achieve a good short-term attitude command response in a generic single-rotor helicopter has been elaborated by Clarke et al. [12]. It achieves appropriate mode decoupling and is consistent with the physical relationships between the state variables. This eigenstructure translates exactly into ideal transfer functions for use with a variety of control design methodologies.

6.3 Flight Control

Fly-by-wire (FBW) describes aircraft in which the mechanical links between the pilot controls and the control surfaces have been replaced by electrical connections (Optical connections may also be employed this is sometimes called Fly-by-Light (FBL)). The two major benefits of FBW are weight reduction, due to omission of the mechanical control runs, and the ability to use a full authority Automatic Flight Control System (AFCS).

Flying a helicopter without the assistance of a stability augmentation system (SAS) or automatic flight control system (AFCS) is a taxing task for any pilot. The AFCS is a complicated and challenging system to develop since it must resolve a wide range of implementations and control problems. Some of the requirements on a modern AFCS are listed below [1]:

- Reduced pilot work load through improved handling qualities.
- Be fully integrated with the navigation system, mission management computer, and engine management system and sensor units.
- Must be fault tolerant, incorporating self verification, multiple redundant systems and limited authority.
- Should provide a range of autopilot functions from simple heading, height and speed hold to more complicated functions such as auto-land and course following.

The progression of the helicopter industry towards FBW has far reaching implications for control law design, some of which are highlighted below [1]:

- Improved handling qualities with command interpretation will require more sophisticated control laws.

- Integration with subsystems and the need for more sensor data to implement load reduction and suppression of flexure modes will increase the number inputs and outputs. This will lead to larger control laws.

- The increased dependence upon the control law to meet performance goals and the growing impact of the airframe and rotor designs on the control law solution will require the control law design to be initiated earlier as a parallel task that forms an integral part of the vehicle design.

- Multi-mode AFCSs will be needed for mission tailored handling qualities this will require more control laws. Furthermore, gain scheduling will be needed to maintain performance across the flight envelope and extend it into non-linear regions. Each schedule implies an additional control law. Thus an increased number of control laws will be needed.

- The complete AFCS will use switching logic and scheduling algorithms to combine all the control laws. Efficient integration will place constraints on the control law structure.

- A multi-objective approach to control law design will be needed to achieved the array of benefits that a full authority AFCS can bring. Inevitably, objectives will have to be traded-off and the design process should make the trade-offs evident and facilitate fast design iterations.

Consideration of the above points, in particular robustness, fast design iterations, increased complexity, difficulty and size, indicates that large sophisticated, cross-coupled control laws are needed to exploit the opportunities of FBW. Only multivariable techniques can deliver this.

The problem of eigenstructure assignment is of great importance in control theory and applications because the stability and dynamic behaviour of a linear multivariable system are governed by the eigenstructure of the system [13]. In [1] many design techniques have studied and the authors concluded that eigenstructure is the best tool for helicopter flight control.

Eigenstructure assignment is the only technique that addresses the two issues important to bridging the practice-theory gap: design parameter visibility and controller structure. The

eigenstructure can not only be related to the design objectives but, through its clear links with the time domain response [7], it can also be related to the final performance.

Working with the eigenstructure has further benefits. It is a natural representation for the system dynamics, and its analysis is an established branch of aeronautical engineering [14]. Thus the eigenstructure facilitates a free flow of information between analysis and design, by allowing both control engineers and aerodynamicists to work with familiar concepts. This should aid an integrated approach to vehicle design.

Eigenstructure assignment does not inherently encompass dynamic compensation but simple extensions have been developed [15] that give complete control over the dynamic compensator order and some ability to determine its structure through the choice of input/output variables.

Eigenstructure assignment is a very flexible technique. It provides access to all the available design freedom and a range of add-ons has been developed [16] to exploit this.

6.4 On Helicopters

Due to the operating characteristics of the helicopter -its ability to take off and land vertically, and to hover for extended periods of time, as well as the aircraft's handling properties under low airspeed conditions- it has been chosen to conduct tasks that were previously not possible with other aircraft, or were time- or work-intensive to accomplish on the ground.

Today, helicopter uses include transportation of people and cargo, military uses, construction, firefighting, search and rescue, tourism, medical transport, aerial photography and observation, and reflection seismology, among others.

In order to effect control of a system so complex as a helicopter, it must first be understood from a theoretical standpoint. Once the general characteristics have been identified by examining the physics of the system, a controller can be developed which will specifically address the problems found.

A helicopter is a type of rotorcraft in which lift and thrust are supplied by rotors. This allows the helicopter to take off and land vertically, to hover, and to fly forward, backward, and laterally. These attributes allow helicopters to be used in congested or isolated areas where fixed-wing aircraft and other forms of vertical takeoff and landing aircraft cannot perform [3].

It should be noted that the vast majority of helicopters currently in service are in the same

configuration: a main rotor on top of the aircraft and a tail rotor mounted vertically at the rear.

6.4.1 Helicopter components

A Helicopter is composed, mainly, of the following components:

- Rotor system: The rotor system, or more simply rotor, is the rotating part of a helicopter that generates lift. A rotor system may be mounted horizontally, as main rotors are, providing lift vertically, or it may be mounted vertically, such as a tail rotor, to provide horizontally thrust to counteract torque from the main rotors. The rotor consists of a mast, hub and rotor blades.

- Flight controls: The pilot must be given a large amount of control over the helicopter. He must be able to control its height, its velocity in both horizontal axes, and its yaw rate at the very least. A helicopter has four flight control inputs. These are the cyclic, the collective, the anti-torque pedals, and the throttle.

- The cyclic control is usually located between the pilot's legs and is commonly called the cyclic stick or just cyclic. On most helicopters, the cyclic is similar to a joystick. The control is called the cyclic because it changes the pitch of the rotor blades cyclically. The result is to tilt the rotor disk in a particular direction, resulting in the helicopter moving in that direction.

- The collective pitch control or collective is located on the left side of the pilot's seat with a settable friction control to prevent inadvertent movement. The collective changes the pitch angle of all the main rotor blades collectively (i.e. all at the same time) and independently of their position. Therefore, if a collective input is made, all the blades change equally, and the result is the helicopter increasing or decreasing in altitude.

- The anti-torque pedals are located in the same position as the rudder pedals in a fixed-wing aircraft, and serve a similar purpose, namely to control the direction in which the nose of the aircraft is pointed.

- The throttle controls the power produced by the engine, which is connected to the rotor by a fixed ratio transmission. The purpose of the throttle is to maintain enough engine

power to keep the rotor RPM (revolutions per minute) within allowable limits so that the rotor produces enough lift for flight. A swash plate transmits the pilot commands to the main rotor blades for articulated rotors.

6.4.2 Helicopter Flight

There are three basic flight conditions for a helicopter: hover, forward flight and the transition between the two.

- Hover : Hovering is the most challenging part of flying a helicopter. This is because a helicopter generates its own gusty air while in a hover, which acts against the fuselage and flight control surfaces. The end result is constant control inputs and corrections by the pilot to keep the helicopter where it is required to be.

- Transition from hover to forward flight: As a helicopter moves from hover to forward flight it enters a state called translational lift which provides extra lift without increasing power.

- Forward flight : In forward flight a helicopter's flight controls behave more like those of a fixed-wing aircraft. Coordinating these two inputs, down collective plus aft cyclic or up collective plus forward cyclic, will result in airspeed changes while maintaining a constant altitude.

Helicopters are inherently extremely complex machines, and detailed analysis of their operation is consequently very involved. However, various simplifications can be made to the flight model, allowing a variety of compromises between complexity and accuracy.

A full description of helicopter dynamics may be found in a number of texts [17, 18, 19], and useful summaries are also available [1, 2, 3].

6.4.3 The pilot controls

In the cockpit, the pilot has a collective lever and cyclic stick. These are generally connected to a swash-plate assembly. This transfers the body-fixed controls to the rotating rotor system. It consists of both a fixed and a rotating plate. The rotating plate is connected to the blades and determines their pitch. The swash-plate is moved vertically to provide collective control and tilted to provide cyclic control.

The pilot also has two foot pedals to control the tail rotor collective (θ_t). Depressing the left or right pedal decreases or increases the tail rotor thrust and is used to balance the main rotor torque reaction or control heading and side slip at low speeds.

The pilot uses the main rotor collective to vary the collective pitch (θ_0) of the main rotor blades and thus the amount of thrust developed. The blades rotate to change the angles of pitch on all blades simultaneously and equally. This rotation is often called feathering and is accomplished through a feathering hinge.

By using the lateral and longitudinal cyclic controls (A_1 and B_1 respectively), the pilot can vary the angle of incidence of the rotor blades as a function of their rotational position about the main rotor shaft. The blades respond to the cyclically-changing angles of pitch by moving out of their plane of rotation (flapping). This provides control over the distribution of lift across the rotor disk and effectively enables the main rotor thrust to be tilted away from the vertical.

A pilot principal task while flying is to maintain a desired flight condition by holding the forces and moments about the three airframe axes at equilibrium. When equilibrium is achieved the helicopter is said to be trimmed. Common trim conditions are [3]:

- Hover: all resultant forces, moments and translational velocities are zero. To hold this trim condition the helicopter will have to adopt starboard roll attitude to balance the tail rotor thrust and often a nose-up pitch attitude to accommodate the main rotor shaft tilt and an aft center of gravity.

- Straight and level flight: again all angular and linear accelerations are zero and the forward velocity is constant. At low speeds the pilot will trim for zero side slip and accept a small roll attitude.

- Coordinated turn: the helicopter turns with a fixed bank angle while holding all linear accelerations and pitch rate at zero. Roll and yaw rate are adjusted via the tail rotor thrust such that the fuselage side force is zero. This ensures passengers and pilot are not swept from side to side as the helicopter turns.

- Steady climb: the helicopter is trimmed to a fixed climb angle and rate.

Mathematically, a trim condition is a set of constraints that, when applied to the equations of motion and flapping, enables the control angles and other parameters to be determined. For

a unique solution to a given trim condition, the number of constraints must equal the number of variables.

6.4.4 Flight equations

The mathematical description of a helicopter can ultimately be expressed in the following generic form:

$$\dot{x} = f(x, u, t) \tag{6.4.1}$$

Where $f(x, u, t)$ is a vector of functions thus $[f_1(x, u, t), \cdots, f_2(x, u, t)]^T$, x is the state vector, u is the input vector and t denotes time.

Typical definitions of the state and input vectors, for a simple standard model, are:

$$x^T \hat{=} [u, v, w, p, q, r, \phi, \theta, \psi, a_{1s}, b_{1s}] \tag{6.4.2}$$

$$u^T \hat{=} [A_1, B_1, \theta_0, \theta_t] \tag{6.4.3}$$

Where the states and inputs are as follows and some of them are shown on figure 6.1 [1]:

- u: Forward speed,
- v: Lateral speed (side-slip velocity),
- w: Vertical speed (heave velocity),
- p: Roll rate,
- q: Pitch rate,
- r: Yaw rate,
- ϕ: Roll angle,
- θ: Pitch angle,
- ψ: Yaw angle,
- a_{1s}, b_{1s}: Longitudinal and lateral blade flapping angles,
- θ_0: Collective pitch input,

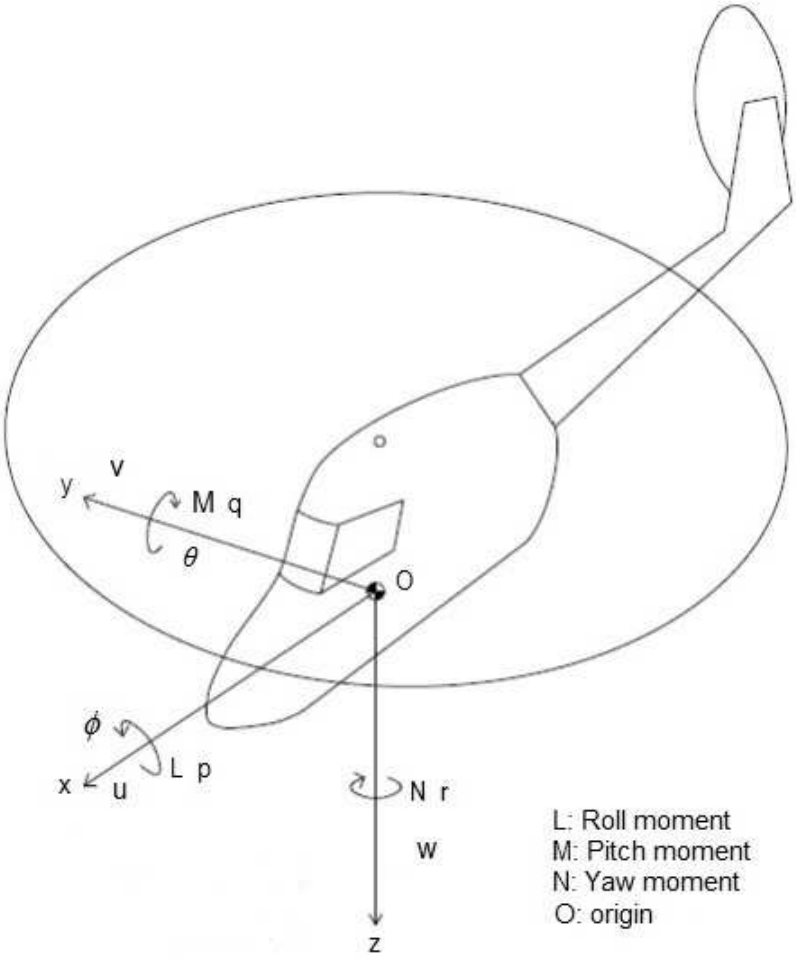


Figure 6.1: Conventional body fixed axis set for a helicopter

- θ_t: Tail rotor pitch input,
- A_1, B_1: Lateral and longitudinal Cyclic pitch inputs

6.5 Helicopter flight compensator design

In this section the steps of the design process established in the precedent chapter will be used to control the flight of a helicopter. Like previously, at each step a recall of the step will be given.

6.5.1 Helicopter open loop system

The following state, input and output matrices are the 8th order linearization of the Westland Lynx helicopter model [2]. The states are, respectively, sideslip velocity (v), roll rate (p), roll angle (ϕ), forward velocity (u), pitch rate (q), pitch angle (θ), vertical velocity (w) and yaw

rate (r), and the inputs to the system are lateral (A_1) and longitudinal (B_1) cyclic, main rotor collective (θ_0) and tail rotor collective (θ_t). The measurable outputs form a subset of the system states. p, q, r, θ and ϕ are commonly measured directly, whilst $\dot{h}$ may be estimated in place of w (h measures the height and $\dot{h}$ measures the rate of change in height). The following provides a measure for $\dot{h}$ with $\theta = 3.4°$ and $\phi = 3.3°$ for the hover trim condition [1]:

$$\dot{h} = sin(\theta)u - cos(\theta)sin(\phi)v - cos(\theta)cos(\phi)w$$

So the outputs are the following: vertical speed in inertial frame ($\dot{h}$), roll rate (p), pitch rate (q), yaw rate (r), pitch angle (θ) and roll angle (ϕ).

The Helicopter Continuous-time model with 8 states, 4 inputs and 6 outputs is as follows:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{6.5.1}$$

where A, B, C are 8×8, 8×4 and 6×8 rational matrices:

$$A = \begin{pmatrix} -0.0384 & -0.2890 & 3.2064 & 0.0494 & -0.0678 & 0.0110 & 0 & 0.0354 \\ -0.5643 & -9.7105 & 0 & 1.16778 & 4.5094 & 0 & 0.01167 & -0.0260 \\ 0 & 1 & 0 & 0 & -0.0034 & 0 & 0 & 0.0596 \\ 0.0002 & -0.0411 & 0 & -0.0337 & 0.2883 & -3.2117 & 0.0157 & 0 \\ -0.0010 & -0.7938 & 0 & 0.1580 & -1.5223 & 0 & -0.0104 & 0 \\ 0 & 0 & 0 & 0 & 0.9984 & 0 & 0 & 0.0572 \\ 0 & -0.0029 & 0.4836 & 0.0278 & 0.0147 & -0.1914 & -0.3230 & 0 \\ -0.0150 & -1.7137 & 0 & 0.02979 & 0.8642 & 0 & 0.0481 & -0.2208 \end{pmatrix}$$

$$B = \begin{pmatrix} 37.28 & 0.5602 & -1.415 & 12.89 \\ 128.3 & 1.928 & 6.723 & -0.9451 \\ 0 & 0 & 0 & 0 \\ -0.5570 & 37.50 & 17.90 & 0 \\ 0.2920 & -19.66 & -1.523 & 0 \\ 0 & 0 & 0 & 0 \\ -0.0389 & 2.618 & -299.4 & 0 \\ 23.12 & 0.3475 & 14.28 & -8.030 \end{pmatrix}$$

$$C = \begin{pmatrix} 0.057 & 0 & 0 & 0.06 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Remark 6.5.1. For more convenience we suppose the input output matrix null.

Table 6.1: Desired eigenvalue locations

λ_p	λ_v	λ_q	λ_u	λ_w	λ_r
-1.5± 1.6j	-0.004	-1.5±1.6j	-0.002	-0.33	-1.75

Table 6.2: Desired eigenvectors

$\tilde{v}_p$	$\tilde{v}_p^*$	$\tilde{v}_v$	$\tilde{v}_q$	$\tilde{v}_q^*$	$\tilde{v}_u$	$\tilde{v}_w$	$\tilde{v}_r$
$\frac{1}{(\lambda_v-\lambda_p)}$	$\frac{1}{(\lambda_v-\lambda_p^*)}$	1	0	0	0	0	0
$\frac{1}{\lambda_p^*}$	$\frac{1}{\lambda_p}$	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	$\frac{1}{(\lambda_u-\lambda_q)}$	$\frac{1}{(\lambda_u-\lambda_q^*)}$	1	0	0
0	0	0	$\frac{1}{\lambda_q^*}$	$\frac{1}{\lambda_q}$	0	0	0
0	0	0	1	1	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

6.5.2 An existing ideal eigenstructure

The formulation of an ideal eigenstructure was considered in detail by Griffin (1997) and Clarke et al. (2003) [1, 12], in order to satisfy the Def-Stan Level 1 handling qualities criteria for attitude command at hover [20]. Eigenvalues and eigenvectors were treated separately and the eigenvectors were formulated for the case where the cyclic pitch inputs control the attitude of the helicopter. Level 1 handling qualities are achievable only in hover using this type of control response.

The desired eigenvector sets may be considered in small decoupled subsystems since one requirement was that coupling between the responses in different axes should be minimal. Clarke et al. (2003) derive the ideal eigenvector set by using a transfer function approach [12]. In this way kinematic constraints are introduced early on, ensuring the correct integral relationship between q and θ, for example. Combining these kinematic constraints with the mathematical constraint of orthogonality of the left and right eigenvector sets leads to the result given by tables 6.1 and 6.2.

6.5.3 Conversion from SSD to MFD

The helicopter model is block controllable (i.e. controllable), the number of states is a multiple of the number of inputs, $n/m = \mu$ with μ, the controllability index, equal to 2, and the controllability matrix of degree 2 is of full rank (i.e. rank=n).

Remark 6.5.2. The system is not block observable.

The block controller form of the SSD is obtained first then the right MFD form is obtained. The results obtained (using Matlab) with a precision of 4 digits are the following:

So the RMFD form of the helicopter model with 4 inputs and 6 outputs is given by :

$$G(s) = D^{-1}(s)N(s) \tag{6.5.2}$$

where

$$D(s) = I_4 s^2 + D_1 s + D_0 \tag{6.5.3}$$

with the following matrix coefficients:

$$D_1 = \begin{pmatrix} 9.3784 & 0.6955 & 0.3870 & -0.0062 \\ -1.0635 & 3.4029 & 0.4206 & 0.0380 \\ -18.3850 & -13.4559 & -1.4574 & -0.1090 \\ -144.1628 & -48.7963 & -11.1223 & 0.5248 \end{pmatrix}$$

$$D_0 = \begin{pmatrix} -5.6566 & -2.0288 & -0.5544 & 0.0394 \\ 0.5609 & -1.3436 & -0.0131 & -0.0366 \\ -5.8876 & -4.3651 & -0.5733 & -0.0363 \\ -23.5351 & 2.7859 & -0.8893 & 0.2704 \end{pmatrix}$$

And

$$N(s) = N_1 s + N_0 \tag{6.5.4}$$

with the following matrix coefficients:

$$N_1 = \begin{pmatrix} 2.1304 & -0.3361 & 300.3933 & 0.7347 \\ 128.3000 & 1.9280 & 6.7230 & -0.9451 \\ 0.2920 & -19.6600 & -1.5230 & 0 \\ 23.1200 & 0.3475 & 14.2800 & -8.0300 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$N_0 = \begin{pmatrix} -5610.395 & -4077.754 & -542.8364 & -32.4198 \\ -52.9843 & -12.4292 & -3.1425 & 0.1616 \\ -50.7662 & -11.9089 & -3.0110 & 0.1548 \\ 886.1016 & 207.8647 & 52.5552 & -2.7021 \\ 1.6140 & -19.6087 & -0.7037 & -0.4593 \\ 129.6770 & 2.0156 & 7.5793 & -1.4237 \end{pmatrix}$$

6.5.4 Desired latent structure

The desired eigenstructure for the helicopter model [2] is given by:

$$d\Lambda = diag\{\begin{array}{cccc} -1.5-1.6j & -1.5+1.6j & -0.004 & -1.5-1.6j \\ & -1.5+1.6j & -0.002 & -0.33 \quad -1.75 \end{array}\}$$

$$d\tilde{V} = \begin{pmatrix} 0.31+0.33i & 0.31-0.33i & 1 & 0 & 0 & 0 & 0 & 0 \\ -0.31+0.33i & -0.31-0.33i & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.31+0.33i & 0.31-0.33i & 1 & 0 & 0 \\ 0 & 0 & 0 & -0.31+0.33i & -0.31-0.33i & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.5.5}$$

This desired eigenstructure has to be converted into a desired latent structure.

From this desired eigenstructure a desired state matrix is constructed ($dA = d\tilde{V}d\Lambda d\tilde{V}^{-1}$), then a desired latent structure is obtained using equation 6.5.6 where T_{c1} is computed using the desired state matrix and the input matrix B.

$$v_i = T_{c1}\tilde{v}_i \tag{6.5.6}$$

Where

$$T_{c1} = \begin{pmatrix} 0_4 & I_4 \end{pmatrix} \begin{pmatrix} B & dAB \end{pmatrix} \tag{6.5.7}$$

Then the desired latent values are:

{-0.004, -0.002, -1.5+1.6j, -1.5-1.6j, -1.5+1.6j, -1.5-1.6j, -0.33, -1.75}

And the corresponding desired latent vectors are:

$$dV = \begin{pmatrix} 0.0003 & -0.0083 & 0.0014 & 0.0014 \\ 0 & -0.0159 & 0 & 0 \\ 0 & 0.2033 & 0 & 0 \\ 0.0435 & 0.2847 & 0.0087+0.0050i & 0.0087-0.0050i \\ -0.0062-0.0068i & -0.0062+0.0068i & -0.0004 & 0.0005 \\ -0.0210-0.0129i & -0.0210+0.0129i & -0.0008 & 0 \\ 0.1551+0.1657i & 0.1551-0.1657i & 0.0102 & 0 \\ 0.2172+0.2321i & 0.2172-0.2321i & 0.0169 & 0.0729 \end{pmatrix}$$

Remark 6.5.3. The desired latent values are equal to the eigenvalues.

6.5.5 Desired block poles

With only two block roots we can get only a static compensator.

To get a dynamic compensator the desired system is augmented, so to the previous desired eigenvalues we added 4 new eigenvalues with a high magnitude (to lower its effect on the system response): {-10, -11, -12, -13} and four unity latent vectors chosen such that three solvents can be constructed from the desired set using equation 6.5.8:

$$R_i = dV_i d\Lambda_i dW_i \tag{6.5.8}$$

Where Λ_{di} is a diagonal matrix of four latent values, dV_i, and dW_i is the matrix of corresponding linearly independent right latent vectors and its dual matrix of left latent vectors.

- From the set of complex latent values {-1.5000 + 1.6000i, -1.5000 - 1.6000i, -1.5000 +,1.6000i, -1.5000 - 1.6000i} and its corresponding latent vectors:
$$dR_1 = \begin{pmatrix} -4.4203 & -2.3122 & -1.0309 & 0.4790 \\ 0.0432 & -7.3664 & -0.6689 & 0.0071 \\ -0.8311 & 55.9568 & 4.3320 & 0 \\ -24.7284 & 78.0120 & 3.0376 & 1.4546 \end{pmatrix}$$
- From the set {-0.0040, -0.0020, -10, -11} and its corresponding latent vectors:
$$dR_2 = \begin{pmatrix} -10.0000 & 0 & -0.5121 & 0.0736 \\ 0 & -11.0000 & -0.8603 & 0.0012 \\ 0 & 0 & -0.0020 & 0 \\ 0 & 0 & 0.0028 & -0.0040 \end{pmatrix}$$
- From the set {-0.3300, -1.7500, -12, -13} and its latent vectors:
$$dR_3 = \begin{pmatrix} -1.7606 & 0.7143 & 0 & 0 \\ -0.0212 & -0.3194 & 0 & 0 \\ 2.2375 & -150.6449 & -12 & 0 \\ 1543.016 & -1040.608 & 0 & -13 \end{pmatrix}$$

6.5.6 Desired closed loop denominator

From the 3 desired block roots we can construct the denominator of the closed loop system using equation 6.5.9:

$$\begin{pmatrix} dD_0 & dD_1 & dD_2 \end{pmatrix} = -\begin{pmatrix} dR_1^3 & dR_2^3 & dR_3^3 \end{pmatrix} V_R^{-1} \tag{6.5.9}$$

where V_R is the 'right' block Vandermonde matrix of degree 3.

The following desired denominator of the closed loop system is thus obtained:

$$D_f(s) = I_4 s^3 + dD_2 s^2 + dD_1 s + dD_0 \tag{6.5.10}$$

where

$$dD_2 = \begin{pmatrix} 13.0735 & 2.2190 & 0.2644 & -0.0081 \\ -0.0764 & 18.6579 & 0.5241 & -0.0001 \\ 1.0918 & -65.0579 & 7.7115 & -0.0001 \\ 20.8985 & -98.7786 & -8.7255 & 14.6431 \end{pmatrix}$$

$$dD_1 = \begin{pmatrix} 35.7827 & 25.2594 & 3.4496 & -0.1338 \\ -0.7912 & 90.6892 & 6.7505 & -0.0021 \\ 11.3864 & -740.4158 & -53.2180 & -0.0014 \\ 254.9963 & -1101.375 & -103.8139 & 21.1080 \end{pmatrix}$$

$$dD_0 = \begin{pmatrix} 50.4734 & 9.3514 & 3.3160 & -0.3722 \\ -0.2721 & 70.9814 & 5.5382 & -0.0058 \\ 4.6804 & -272.5657 & -21.0743 & -0.0046 \\ 460.1159 & -162.9178 & 10.6996 & -3.2796 \end{pmatrix}$$

Remark 6.5.4. .

- The block Vandermonde matrix V_R has been verified non-singular so the desired latent vectors satisfy the design process conditions.
- D_f has been verified by generating its latent values and latent vectors which match the desired latent structure.
- The design process in this case (deriving MFD models from SSD models of the plant and the desired one) gives always a column reduced D (open loop system denominator), right (left) coprime D and N (open loop system numerator) and column and row reduced D_f (desired closed loop system denominator). So all the conditions, for the existence of a solution to the Diophantine equation, are satisfied.

6.5.7 Input-Output feedback compensation

The final step of the design process is to choose the compensator feedback configuration, which in our case will be the input-output feedback.

In this configuration, we recall that the common denominator of the two compensators D_c is fixed and solving the compensator equation will determine the two numerators.

D_c is assigned arbitrarily with arbitrary poles to get a stable compensator as follows:

$$D_c(s) = I_4 s + Dc_0 \tag{6.5.11}$$

where:

$$Dc_0 = \begin{pmatrix} 0.0049 & 0 & 0 & 0 \\ 0 & 0.0049 & 0 & 0 \\ 0 & 0 & 0.0022 & 0 \\ 0 & 0 & 0 & 0.0022 \end{pmatrix}$$

Here we chose to assign the zeros {-0.0049, -0.0022} of the closed loop system obtained by Griffin using state feedback [1] as the poles of the compensator. Its poles will become, anyway, the zeros of our closed loop system.

Then the Diophantine equation to solve is the following:

$$E(s) = D_f(s) - D_c(s)D(s) = L_c(s)D(s) + M_c(s)N(s) \tag{6.5.12}$$

and we obtain L_c (degree 0):

$$Lc = \begin{pmatrix} 14.1767 & 0.4138 & 1034.261 & 2.5045 \\ 301.4866 & -7.1840 & 22001.38 & 52.6916 \\ -1449.037 & 57.8403 & -107353.5 & -257.1875 \\ -2320.307 & 130.5896 & -177906.8 & -411.8570 \end{pmatrix}$$

And a first degree M_c:

$$M_c(s) = Mc_1 s + Mc_0 \tag{6.5.13}$$

Where

$$Mc_0 = \begin{pmatrix} -1.1137 & -0.2249 & -0.7990 & -0.0420 & -0.0613 & 0.1181 \\ -23.6730 & -11.7747 & 1.5520 & -0.7480 & -3.2043 & -1.5971 \\ 115.5086 & 57.4007 & 5.3439 & 3.5875 & 0.5300 & 14.2470 \\ 191.4321 & 105.9956 & -12.3045 & 3.3359 & -43.9868 & 48.0997 \end{pmatrix}$$

and

$$Mc_1 = \begin{pmatrix} -3.4429 & -0.0246 & 0 & 0 & 0 & 0 \\ -73.2164 & -1.1264 & 0 & 0 & 0 & 0 \\ 357.2835 & 5.5132 & 0 & 0 & 0 & 0 \\ 592.0405 & 9.5406 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The two compensators are of course stable.

6.6 Validation

6.6.1 Handling qualities specification

When designing any controller for any plant it is vitally important to have a detailed specification of the desired final performance of the system. In the case of a helicopter, however, these specifications are hard to characterize. The final performance of the helicopter in a mission role is dependent on both the machine and its pilot, and consequently the pilot's feel for the aircraft is vital.

The short- and long-term modes that form the responses of the helicopter to the pilot inputs are well defined in [1, 21, 20]. They are defined in terms of time-domain criteria, which are more intuitive than the frequency-domain criteria; qualitative descriptions of handling qualities tend to be described in the time domain, using words such as speed, overshoot and sensitivity. The derivation of an ideal eigenstructure in [1] used these criteria directly.

The short term stability criteria defined in [20] are summarized in template form in Figure 6.2. They are specified in terms of the helicopter's response to a one-second pulse at 10% maximum control deflection.

These parameters are defined with, time referenced from the point at which the peak response occurs, and magnitude is measured relative to the peak response. All the parameters including the peak response are defined and justified below [1]:

1. The peak response is the maximum value reached by the time response. The peak response is a measure of control responsiveness that must be greater than a minimum value for adequate responsiveness and less than some maximum to avoid over-sensitivity.

2. The initial delay; is the value of the response (Y_1) at a specified time (T_1=0.5sec) from the initiation of the control input and it must be greater than a lower limit (30%). This parameter prevents the control response from exhibiting excessive sluggishness due to delays and lags.

3. To avoid over-sensitivity the value of Y_1 must be less than an upper limit (70%-80%). Furthermore the response build up should possess no obtrusive hesitation.

4. To achieve the desired stability the transient response must decay rapidly. This is verified using the following parameters:

   a) The time taken (T_{30}) to decay to less the 30% of the peak value must be less than an upper limit (1 sec for level 1).

   b) The minimum value of the first trough (X_1) must be less than an upper limit (15% for level 1). The limit is expressed as a percentage of the peak value.

   c) The maximum of the second peak (X_2) must be less than an upper limit (10%for level 1).

5. The accuracy with which the helicopter returns to the original datum after the pulse input, is specified as a percentage of the peak value ($X_F \approx 10\%$) and must be achieved before $3s \leq T_F \leq 5s$ seconds.

6. Experience has shown that pilots find a small amount of overshoot desirable. For level 1 handling qualities it is therefore necessary to pass through the original datum. This

point (T_{01}) must fall between an upper and lower bound ($1s < T_{01} < 2s$), and the second crossing point (T_{02}) must be greater than a lower limit $T_{02} > 2s$. For level 2 compliance this overshoot criterion is replaced with a simpler one. This states that the time (T_{10}) at which the response reaches 10% of the peak value must lie between an upper and lower limit. To ensure the overshoot is discernible, the parameter X_1 requires a lower bound in addition to the upper bound. A value of ≈1% was used in practice.

So the parameters T_1 and Y_1 confine the initial response delay, while T_F and X_F do the same for the final settling time. T_{30} and T_{10} help define the shape of the response decay after the initial peak, and T_{01}, T_{02}, X_1 and X_2 put bounds on the damping of the response [3].

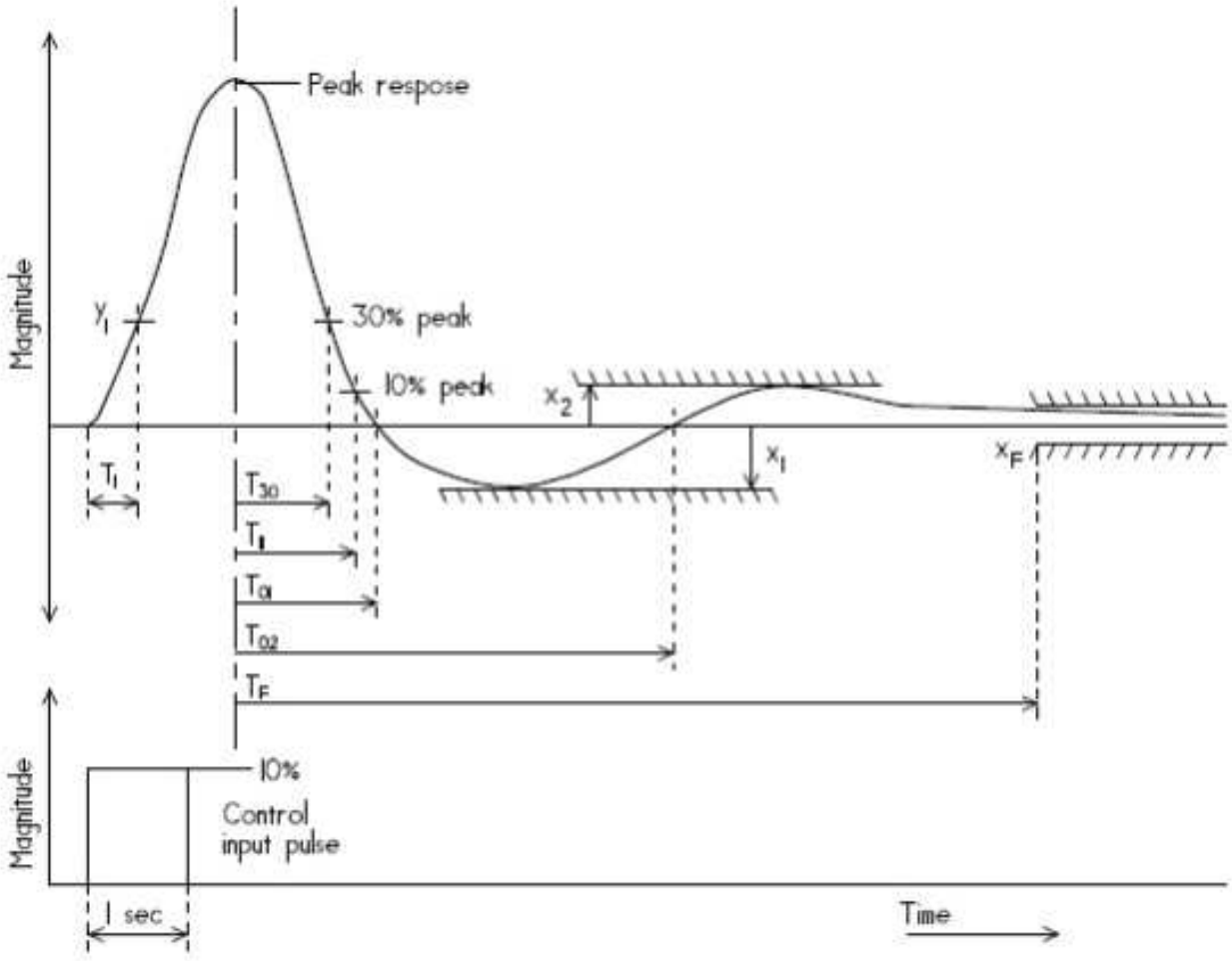


Figure 6.2: A typical transient response

The template shown on figure 6.2 will be used to evaluate the proposed method results.

6.6.2 Results validation

To validate our results, first the poles of the closed loop system have been verified to be in the neighborhood of the desired poles, then a verification of some responses to the desired shapes of a helicopter flight are obtained. The shape of the response and some parameters are verified using the previous template, finally some responses are compared directly to the responses obtained from a state feedback and an output feedback law performed in [1].

1) Closed loop poles:

The closed loop transfer function is:

$$G_{cl}(s) = N(s)D_f^{-1}(s)D_c(s) \tag{6.6.1}$$

Where

$$D_f(s) = D_c(s)D(s) + L_c(s)D(s) + M_c(s)N(s) \tag{6.6.2}$$

The latent values and latent vectors of the closed loop denominator D_f computed, as in equation 6.6.2, have been verified using the function "`polyeig`" of Matlab:

```
[dv,de]=polyeig(dD0,dD1,dD2,dD3);
```

and the results are:

$$de = \begin{pmatrix} -13.0000 & -10.0000 & -12.0000 & -11.0000 & -1.7500 & -0.3300 & -0.0040 & -0.0020 \\ -1.5000+1.6000i & -1.5000-1.6000i & -1.5000+1.6000i & -1.5000-1.6000i \end{pmatrix}$$

$$dv = \begin{pmatrix} 0.0000 & -1.0000 & 0.0000 & 0.0000 & 0.0074 & -0.0200 & 0.0074 & 0.0237 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0001 & -0.0402 & 0.0001 & 0.0453 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.5145 & 0.0000 & -0.5805 \\ -1.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.8563 & 1.0000 & -0.8127 \\ 0.1659+0.1002i & 0.1659-0.1002i & -0.0097-0.0278i & -0.0097+0.0278i \\ 0.0336+0.0097i & 0.0336-0.0097i & -0.0314-0.0571i & -0.0314+0.0571i \\ -0.2574-0.1497i & -0.2574+0.1497i & 0.1403+0.5825i & 0.1403-0.5825i \\ 0.2851+0.8895i & 0.2851-0.8895i & 0.1837+0.7760i & 0.1837-0.7760i \end{pmatrix}$$

From these latent values, if we compute the block roots, we verified that they are equal to the desired ones. For example the first block root ddR_1 computed as follows:

$$dE = \begin{pmatrix} -1.5000+1.6000i & 0 & 0 & 0 \\ 0 & -1.5000-1.6000i & 0 & 0 \\ 0 & 0 & -1.5000+1.6000i & 0 \\ 0 & 0 & 0 & -1.5000-1.6000i \end{pmatrix};$$

$$dV = \begin{pmatrix} 0.1659 + 0.1002i & 0.1659 - 0.1002i & -0.0097 - 0.0278i & -0.0097 + 0.0278i \\ 0.0336 + 0.0097i & 0.0336 - 0.0097i & -0.0314 - 0.0571i & -0.0314 + 0.0571i \\ -0.2574 - 0.1497i & -0.2574 + 0.1497i & 0.1403 + 0.5825i & 0.1403 - 0.5825i \\ 0.2851 + 0.8895i & 0.2851 - 0.8895i & 0.1837 + 0.7760i & 0.1837 - 0.7760i \end{pmatrix};$$

$ddR1 = real(dV * dE * dV^{-1})$ to get

$$ddR1 = \begin{pmatrix} -4.4203 & -2.3122 & -1.0309 & 0.4790 \\ 0.0432 & -7.3664 & -0.6689 & 0.0071 \\ -0.8311 & 55.9568 & 4.3320 & 0 \\ -24.7284 & 78.0120 & 3.0376 & 1.4546 \end{pmatrix}$$

is equal to the desired first block root dR_1 and the same thing for the two other block roots.

Moreover, the poles of the closed loop transfer function, computed as in equation 6.6.1, have been verified, using the function "`pzmap`" of Matlab, to be equal to the desired latent values.

2) Verification of some responses

The following figures shows the response of the proposed approach closed loop system as desired for a helicopter flight.

- Pitch angle (θ) and roll angle (ϕ) responses to the lateral (blue continuous line) and longitudinal (green dashed line) one second pulse input signal are shown in figures 6.3 and 6.4. The roll angle should respond to the lateral stick input but not the longitudinal stick input and the inverse for the pitch angle. The first figure is rather good but not the second figure is not and the reason is the cross coupling of the state variables.

- Figure 6.5 shows the roll rate (p) (in blue continuous line) and roll angle (ϕ) (in green dashed line) to the lateral stick input (A1), and figure 6.6 shows the pitch rate (q) (in blue continuous line) and the pitch angle (θ) (in green dashed line) responses to the longitudinal stick input. The shapes of the responses of the roll rate to the roll angle should correspond and the same thing for the pitch angle and the pitch rate.

- Figure 6.7 shows the yaw rate (r) response to the tail rotor collective (θ_t) pulse input signal and figure 6.8 shows the heave velocity ($\dot{h}$) response to the main rotor collective (θ_0) step input signal. The responses exhibit the desired first order characteristic.

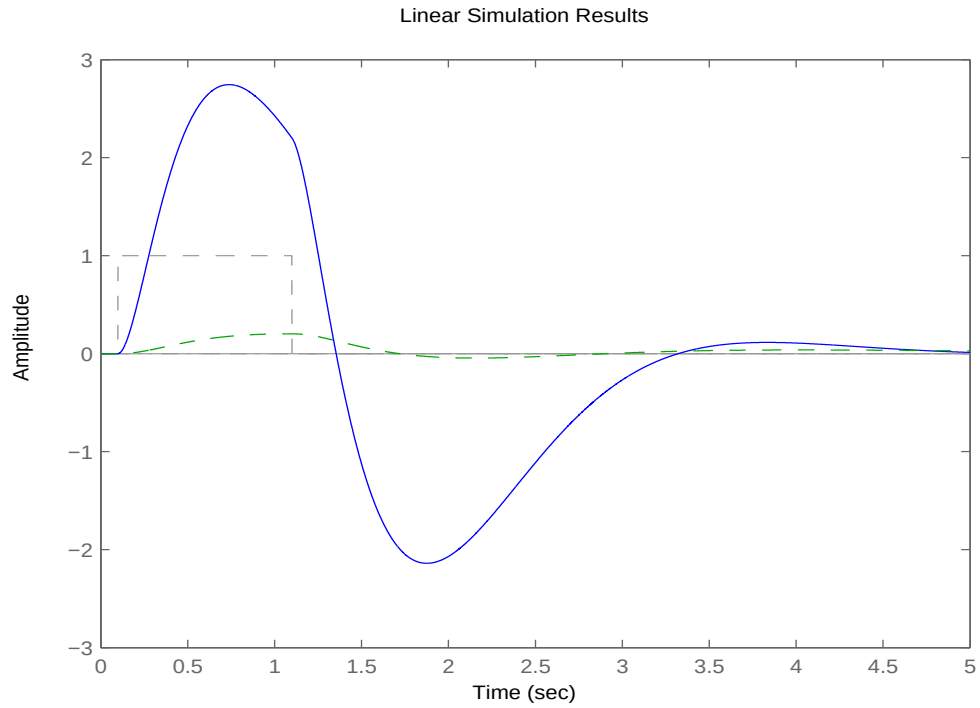


Figure 6.3: Roll angle (ϕ) response to the lateral (A1) and longitudinal (B1) stick input

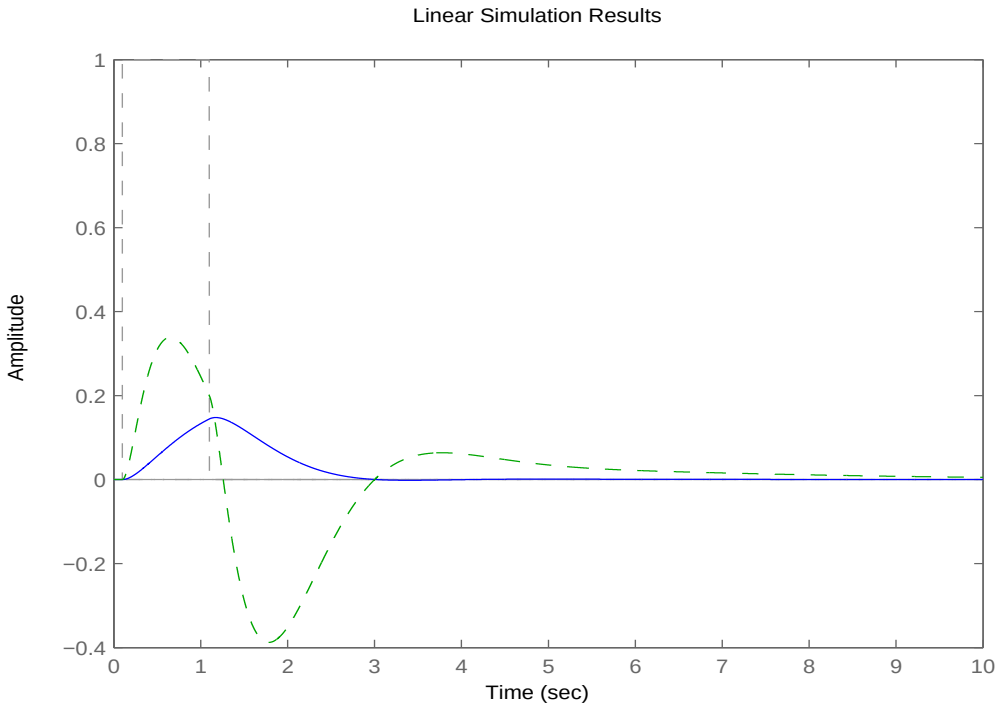


Figure 6.4: Pitch angle (θ) response to the lateral (A1) and longitudinal (B1) stick input

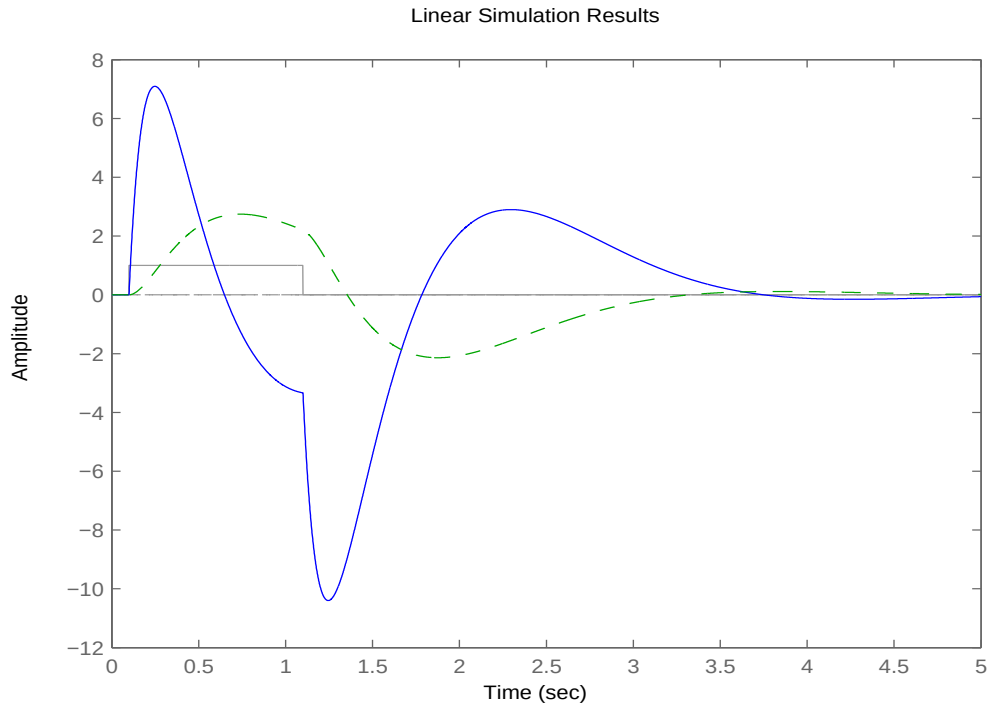


Figure 6.5: Roll rate (p) and roll angle (ϕ) responses to the A1 input

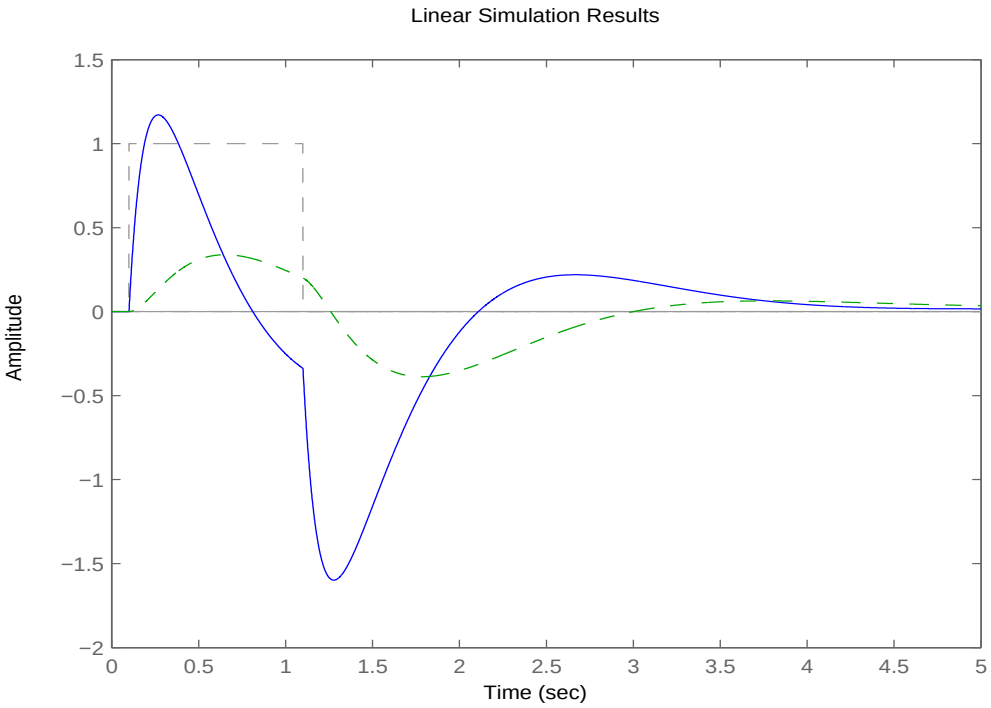


Figure 6.6: Pitch rate (q) and pitch angle (θ) responses to the B1 input

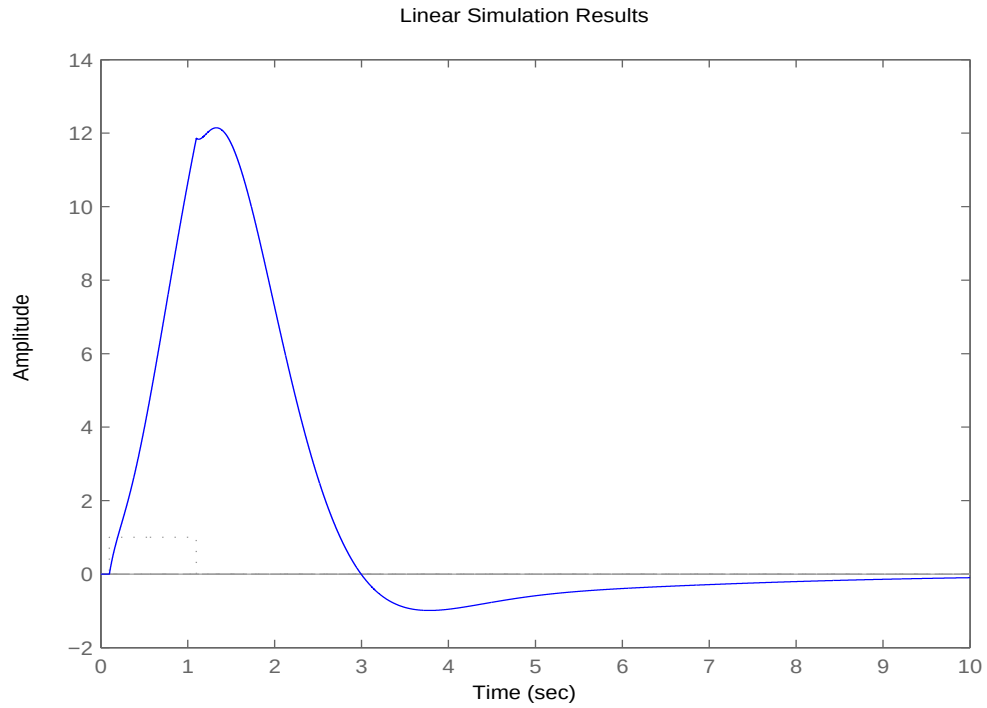


Figure 6.7: Yaw rate (r) response to a tail rotor collective θ_t impulse input

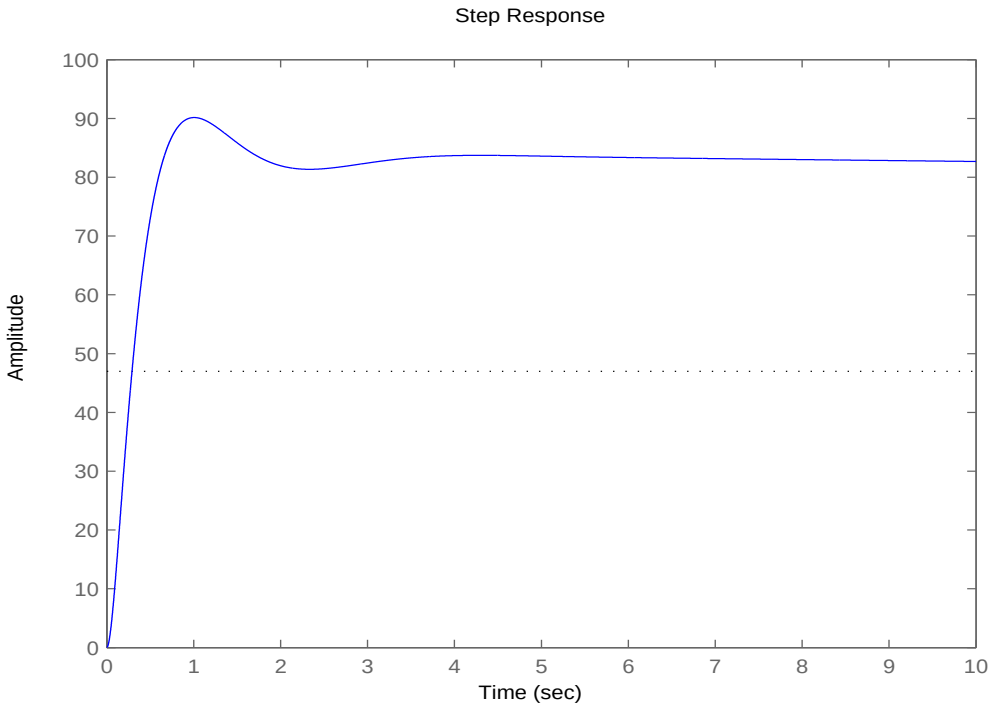


Figure 6.8: Heave velocity ($\dot{h}$) response to a main rotor collective θ_0 step input

3) Comparison with template:

The responses of the closed loop system to a pulse of 1 second are compared to the template given previously (figure 6.2). The following figure shows the roll angle response (ϕ) to a lateral stick 1 second pulse input (A_1).

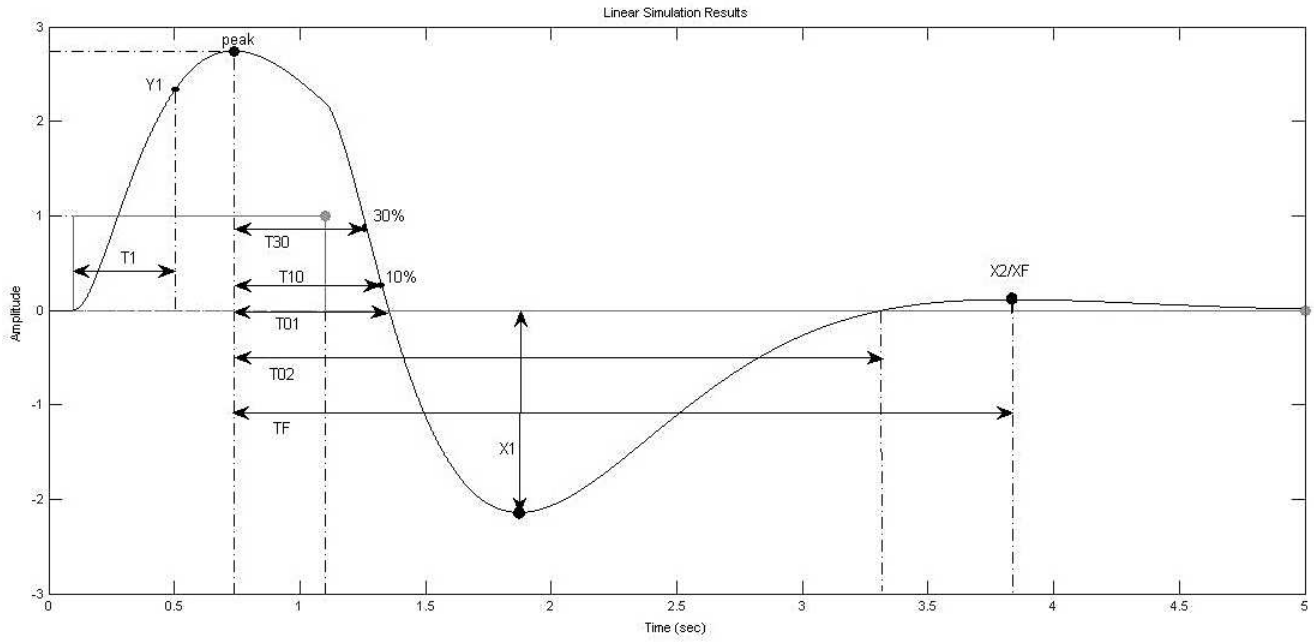


Figure 6.9: Roll angle (ϕ) response to a lateral stick input (A_1)

From figure 6.9, the following results have been obtained:

- Peak response=2.75 at t=0.738s.
- at T1=0.5s the response Y1≈2.32
- T30≈0.53s is under the limit of 1s (T30<1s).
- T10≈0.6s, T01≈0.62s, and T02≈2.57s.
- X1=-2.14.
- X2/XF≈0.114 for TF=3.89s is less than the limit of 10% of peak=1.33.

$Y1$ is over 30% of peak=0.83 which will avoid sluggishness but it is over the upper limit 80% of peak=2.2, i.e. the responses will be over-sensitive. The T02 measure has also limits ($T02 > 2s$) which is rather a good result, but the measures T01 and T10 are not ($1s < T01 < 2s$, $1.5s < T10 < 3s$) because the peak of the response is not as high as it should be. The measure

X1 is over the lower limit (X1>15%) but X2 and XF are under the limit of 10% of peak for TF which is within the limits ($3s < TF < 5s$). So finally, the overall measures are rather good.

4) Comparison with state feedback and output feedback responses:

The responses of the closed loop system to a pulse of 1 second are compared to the responses obtained from a state feedback closed loop system generated by [1] with the following gain matrix:

$$K = \begin{pmatrix} -0.0013 & 0.0004 & 0 & 0.0525 & -0.0361 & -0.0002 & -0.0383 & -0.0034 \\ 0.0008 & -0.0001 & -0.0398 & 0.0743 & 0.0058 & -0.0015 & 0.2450 & \\ 0 & 0 & 0 & 0 & 0.0007 & -0.0002 & 0.0062 & -0.0052 \\ 0.0001 & 0.0011 & 0.0004 & -0.0576 & 0.0154 & 0.1911 & -0.0807 & 0.0103 \end{pmatrix}$$

The responses are also compared to an output feedback closed loop system generated by Griffin [1] with the following gain matrix:

$$K = \begin{pmatrix} 0.0084 & 0.0534 & -0.0336 & -0.0715 & -0.0384 & -0.0031 \\ 0.0004 & -0.0403 & 0.0732 & 0.0564 & -0.0012 & 0.2435 \\ -0.0001 & 0.0004 & 0.0001 & 0.0007 & 0.0059 & 0.0027 \\ 0.0221 & -0.0554 & 0.0072 & -0.0008 & -0.0930 & 0.0714 \end{pmatrix}$$

Where the eigenvectors (r, w) are assigned as input vectors in stage1 and the rest of the eigenvectors are assigned as right eigenvectors in stage2.

For the following figures, the proposed method closed loop system responses are in (blue) continuous lines and the state feedback closed loop system responses are in (green) dashed lines, and the output feedback closed loop system responses are in (red) dotted lines, for the rest of figures the responses are separated for more clearness.

In general state feedback law responses represent the optimal response we could obtain.

- Roll angle responses: Figure 6.10(b) shows a good result (the output signals should approach zero), and figure 6.10(a) shows that the amplitude of the proposed approach is not satisfying. In the rest of figures the output signals should also approach zero which is obtained by the proposed approach responses.

- Pitch angle responses:figures 6.11(a) shows a good response of the proposed approach, but figure 6.11(b) shows a response with too low amplitude.

- Yaw rate responses: Figure 6.12 shows the yaw rate responses. The response to the θ_t seems very good, but it is not the case for the responses to the other three inputs.

- Pitch rate responses: The overall responses are rather good unless in figure 6.13(b) where the amplitude of the proposed approach response is not satisfying as in the figure 6.11(b).

- Roll rate responses: Like in figure 6.10, figure 6.14(b) shows a good result (the output signals should approach zero), and figure 6.14(a) shows that the amplitude of the proposed approach is not satisfying. In the rest of figures the output signals should also approach zero which is obtained by the proposed approach responses.

- Heave velocity responses in figure 6.15: even though this output has not been measured in the state feedback closed loop system in [1], but it has been included for comparison. The overall responses seem better than the output feedback responses.

6.7 Conclusion

Using the parameters defined on the template we can conclude that the proposed method result has better parameters: a better peak angle, the initial response delay is more confined, and presents a better damping and shaping of the response.

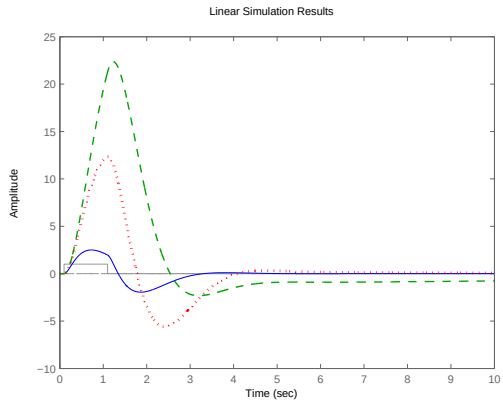
From the shapes of the overall figures, the results are rather good, compared to a state feedback law, which is supposed the most performing and were not far from meeting the level 1 handling qualities criteria.

It was shown that the input-output feedback configuration was able to retain the performance of a state-feedback solution while using only measurable information.

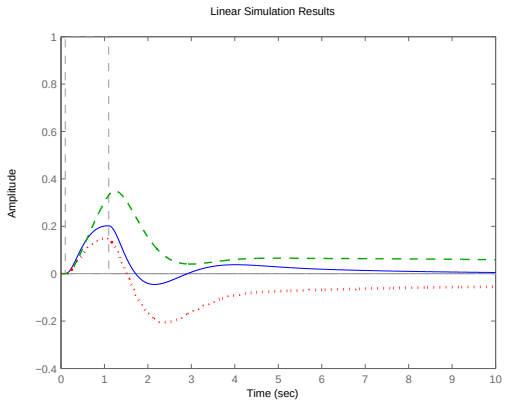
The main objective is to validate the proposed design method by designing a compensator for the helicopter flight control system and, from the results, it has been seen that it is a suitable mechanism for the design of compensators for helicopters. A state feedback solution, provided for reference, has been matched in performance by the input-output feedback solution.

The Helicopter control system was a good design example because the open loop system verifies the resolution conditions of the Diophantine equation and conversion method, and the choice of input-output feedback allows us to design a stable compensation system and a pre-compensator to assign, simultaneously, an eigenstructure for the denominator and zeros for the numerator of the closed loop system. If desired eigenvectors for the block zeros of the closed loop system were known, then an eigenstructure would have been assigned for the numerator as well.

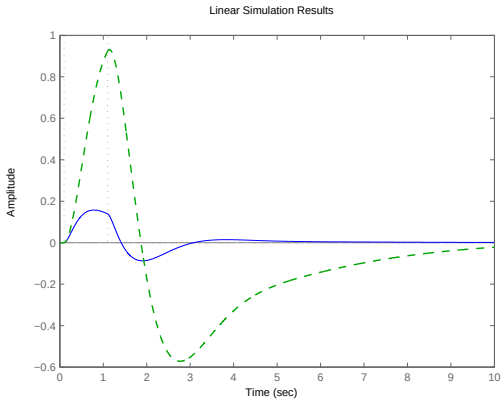
Specification of eigenvalues and eigenvectors for both poles and zeros of the helicopter system described in matrix fraction description will be a natural future step for this thesis.



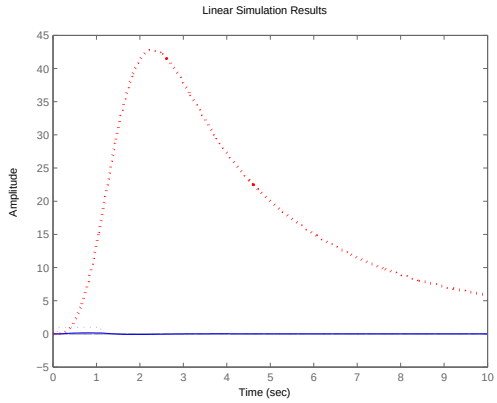
(a) Response to A1 input



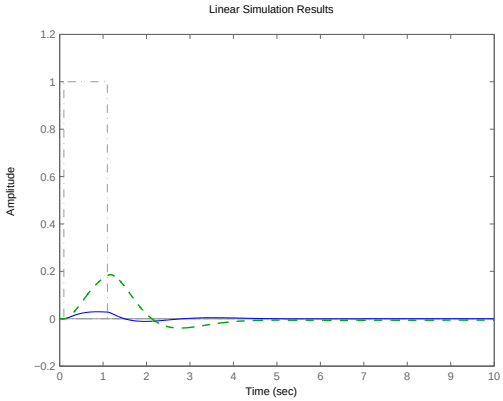
(b) Response to B1 input



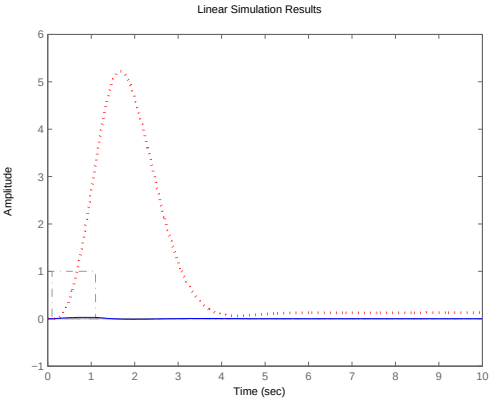
(c) compared to state feedback response to θ_0 input



(d) compared to output feedback response to θ_0 input

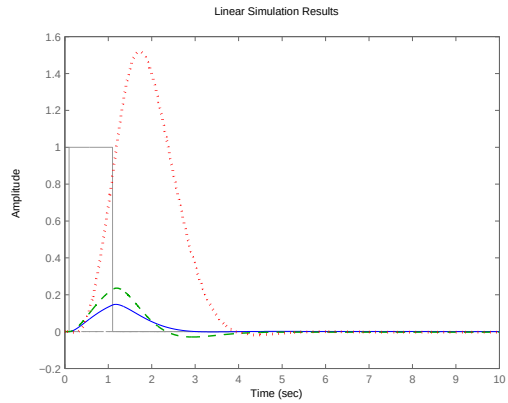


(e) compared to state feedback response to θ_t input

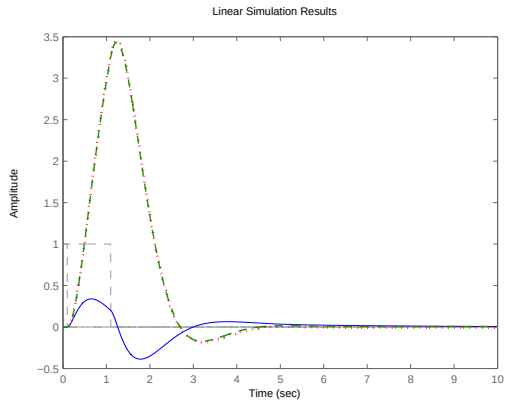


(f) compared to output feedback response to θ_t input

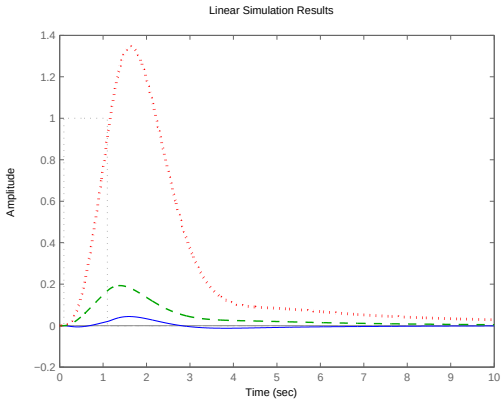
Figure 6.10: Roll angle (ϕ) responses



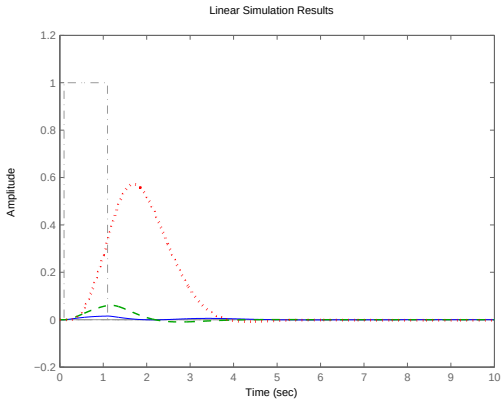
(a) Response to A1 input



(b) Response to B1 input

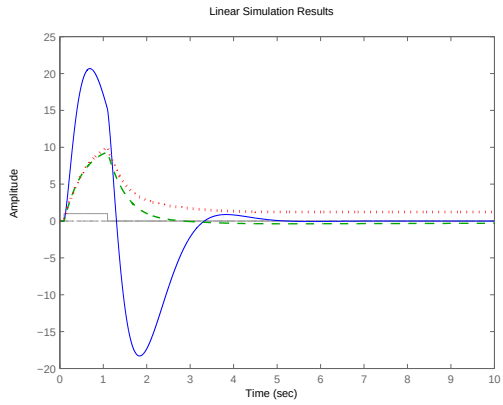


(c) Response to θ_0 input

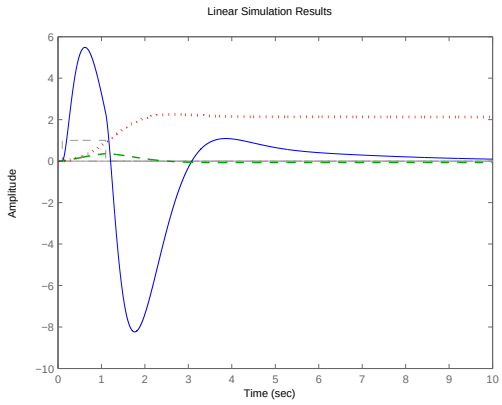


(d) Response to θ_t input

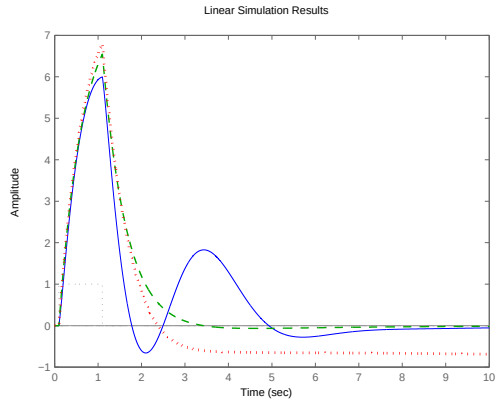
Figure 6.11: Pitch angle (θ) responses



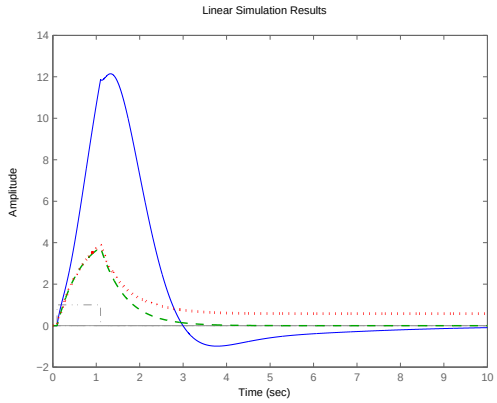
(a) Response to A1 input



(b) Response to B1 input

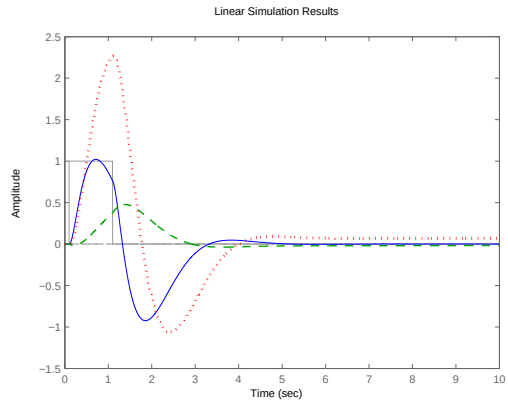


(c) Response to θ_0 input

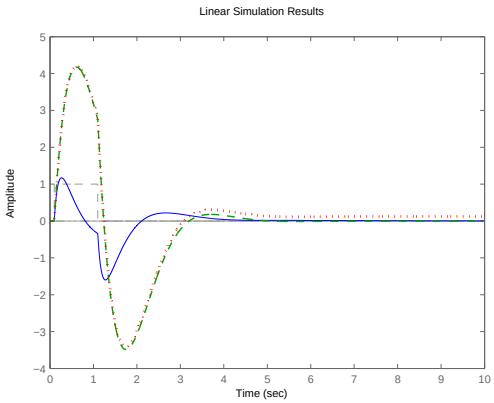


(d) Response to θ_t input

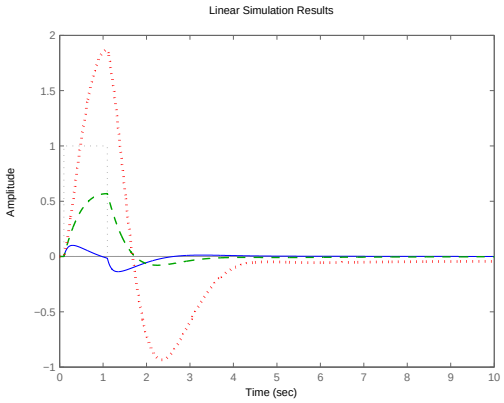
Figure 6.12: Yaw rate (r) responses



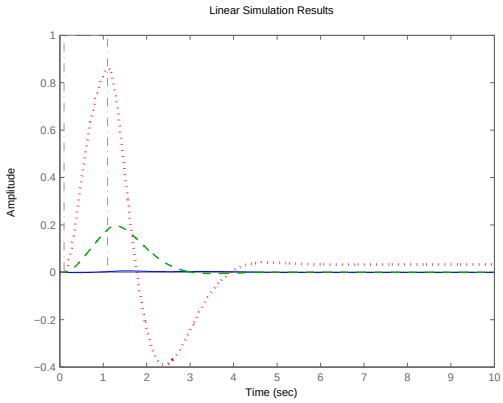
(a) Response to A1 input



(b) Response to B1 input

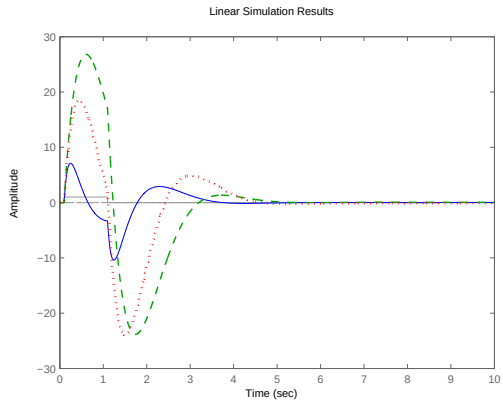


(c) Response to θ_0 input

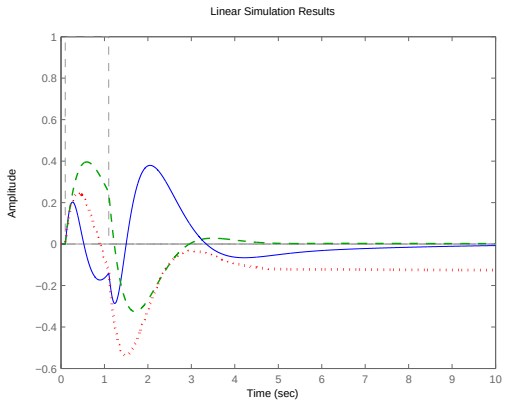


(d) Response to θ_t input

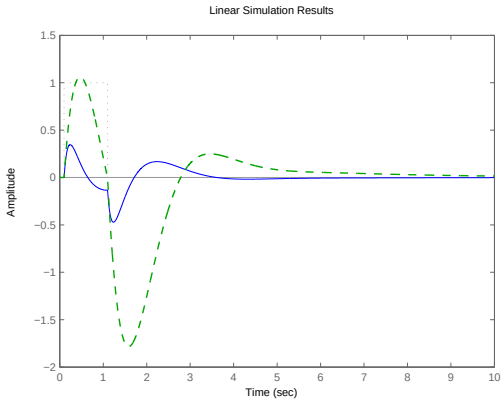
Figure 6.13: Pitch rate (q) responses



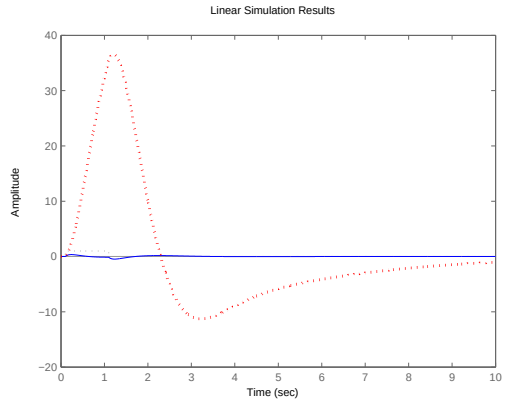
(a) Response to A1 input



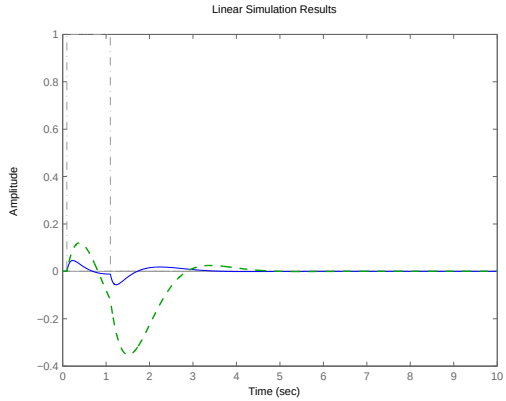
(b) Response to B1 input



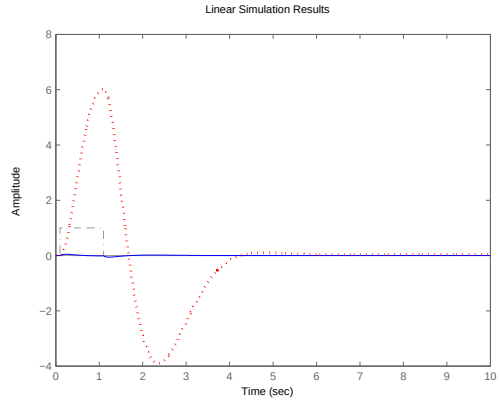
(c) compared to state feedback response to θ_0 input



(d) compared to output feedback response to θ_0 input

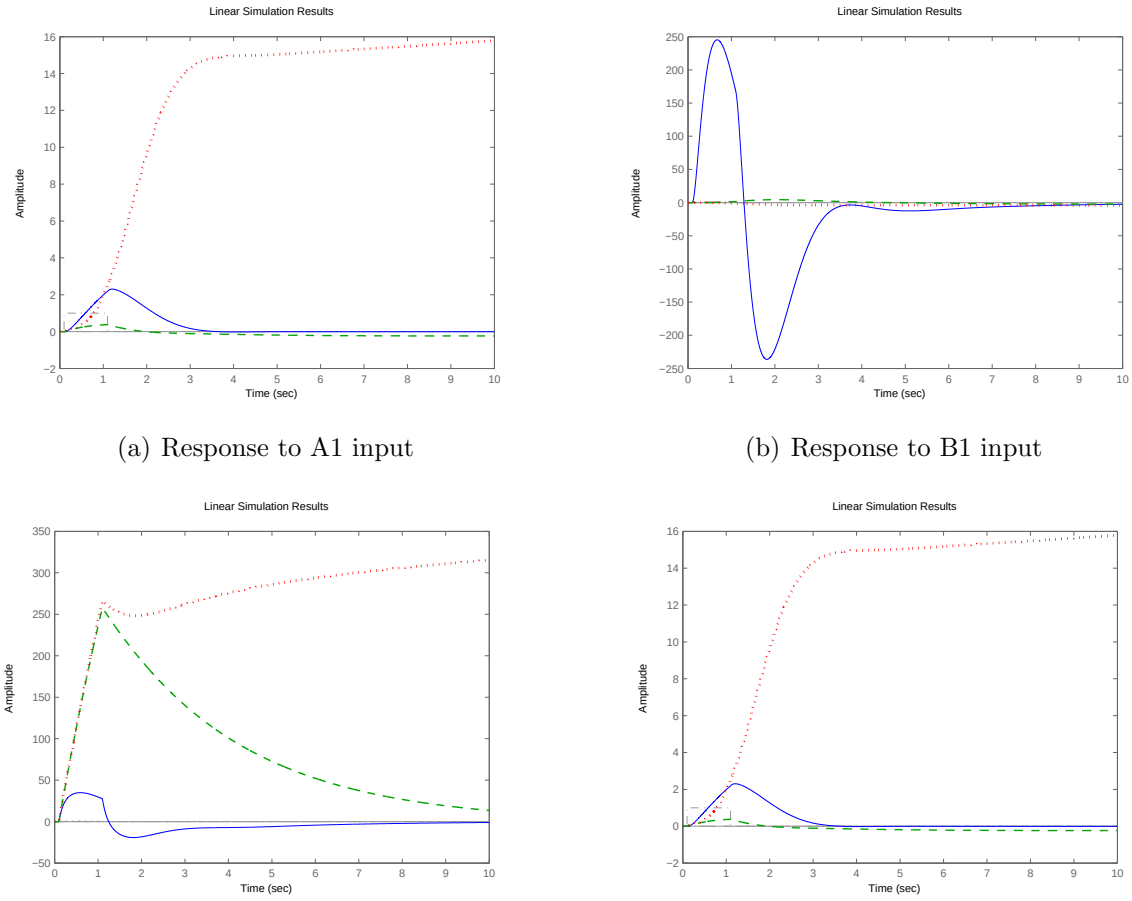


(e) compared to state feedback response to θ_t input



(f) compared to output feedback response to θ_t input

Figure 6.14: Roll rate (p) responses



(a) Response to A1 input

(b) Response to B1 input

(c) Response to θ_0 input

(d) Response to θ_t input

Figure 6.15: Heave velocity ($\dot{h}$) responses

Chapter Bibliography

[1] S. J. Griffin, Helicopter Control Law Design Using Eigenstructure Assignment. DPhil Thesis, Department of Electronics, University of York, 1997.

[2] J. E. Ensor, Subspace Methods for Eigenstructure Assignment, PhD Thesis, University of York, UK, 2000.

[3] A. J. Pomfret, Eigenstructure assignment for helicopter flight control, PhD thesis, university of York, UK, 2006.

[4] A. Iyer and S. N. Singh, "MFDs Of Spinning Satellite and Attitude Control Using Gyrotorquers", IEEE Trans. Aero. Electr. Syst., Vol. 25, No. 5, pp. 611-620, 1989.

[5] P. J. Deore and B. M. Patre, "Design of Robust Compensator for Jet Engine: An Interval

Analysis Approach", in Proc. IEEE Conference on Control Applications, Toronto, Canada, August 28-31, pp. 376-381, 2005.

[6] D. M. Littleboy, Numerical Techniques for Eigenstructure Assignment by Output Feedback in Aircraft Applications, PhD thesis, University of reading, UK, 1994.

[7] A. N. Andry, E. Y. Shapiro, and J. C. Chung. "Eigenstructure assignment for linear systems", IEEE Trans. on Aerospace and Electronic Systems, Vol. 19, No.5, pp. 711-727, 1983.

[8] K. Sobel and E. Y. Shapiro, "Dynamic Output Feedback Flight Control Law Using Eigenstructure Assignment", in Proc. the 5th American Control Conference, Vol. 1, Seattle, WA, pp. 605-610, 1986.

[9] S. K. Mudge and R. J. Patton, "Analysis of the technique of robust eigenstructure assignment with application to aircraft control", IEE PROCEEDINGS, Vol. 135, Pt. D, No. 4, 1988.

[10] J. -F. Magni, "Multimodel eigenstructure assignment in flight-control design Aerospace Science and Technology", No. 3, pp. 141-151, 1999.

[11] S. Bittanti and L. Moiraghi, "Active Control of Vibrations in Helicopters via Pole Assignment Techniques", IEEE Trans. Contr. Syst. Techn., Vol. 2, No. 4, pp. 343-351, 1994.

[12] T. Clarke, J. Ensor and S. J. Griffin, "Desirable eigenstructure for good short-term helicopter handling qualities: the attitude command response case", J. Aerospace Engineering, Vol. 217, pp. 43-55, 2003.

[13] J. Lu, H. D. Chiang and J. S. Thorp, "Eigenstructure assignment by decentralized feedback control", IEEE Trans. Autom. Contr., Vol. AC-38, No.4, pp. 587-594, 1993.

[14] R. W. Prouty. Helicopter Performance Stability and Control, Robert E. Krieger Publishing Company, 1990.

[15] H. Kimura, "Pole assignment by gain output feedback", IEEE Trans. Autom. Contr., Vol. AC-20, pp. 509-516, August 1975.

[16] K. M. Sobel, E. Y. Shapiro and A. N. Andry, "Eigenstructure assignment", Int. J. of Control, Vol. 59, No.1, pp. 13-37, 1994.

[17] A. R. S. Bramwell, Helicopter Dynamics, Edward Arnold Edt., London, 1976.

[18] J. Seddon, Basic Helicopter Aerodynamics, BSP Professional Books, Oxford, 1990.

[19] G. D. Padfield, Helicopter Flight Dynamics, Blackwell Science Ltd, Oxford, 1996.

[20] B. Pitkin, Military Defence Specification DEF-STAN 00-970, British Ministry of Defence, 1989.

[21] T. Clarke and P. Taylor, Helicopter design manual, Technical report, Department of Electronics, University of York, UK, 1999.

Chapter 7

General Conclusion

The aim of this thesis is to design a compensator for systems described in MFD or SSD, by placing block poles, thus performing an eigenstructure assignment. The author asserts that this aim has been met, although further work must be undertaken to improve the design process.

In this chapter remarks on the achieved results and some future works are presented. After a summary of the contents of this thesis, a section on the contributions is given, then the comments and problems encountered while achieving this research work is presented in the third section. Finally future research works which are deduced from the problems are given in the fourth section.

7.1 Introduction

The main contribution presented in this thesis is a design process to assign an eigenstructure for a system described in state space equations by using block poles placement constructed from a desired latent structure, itself obtained from the desired eigenstructure.

The same design process can be applied for systems described in matrix transfer functions, as long as the obtained system in MFD is proper, and can, of course, be applied directly on systems described in matrix fractions.

The steps of the proposed method are as follows:

- A system described by state space equations (SSD), to which a desired eigenstructure has been designed, can be transformed to a system in right or left matrix fraction description (MFD), and the desired eigenstructure can be transformed to a desired latent structure. The condition is (block) controllability or (block) observability of the system. If the system is both block controllable and block observable, then either transformation is

chosen depending on the number of inputs or the number of outputs, to get the smallest size MFD.

- Block poles (solvents) and eventually block zeros can be constructed from this desired latent structure. Conditions of existence of such solvents have been established.
- These block poles are used to construct the desired closed loop denominator. The condition here is the non-singularity of the block Vandermonde matrix composed of block poles which is guaranteed if the set of solvents is a complete set.
- A feedback configuration is to be chosen such that the obtained compensator is proper, stable with least degree. Using the input-output feedback configuration gives more design freedom, in fixing the compensator denominator (so the compensator is stable) and using this denominator to place the eventual desired block zeros.
- The placement of the denominator is done by a dynamic compensator determined by solving the corresponding Diophantine equation (for unity, output or input-output feedback configuration). Conditions of the solvability of the equation are properness, coprimeness and row or column reduceness. Generally, the transformation of the system generates directly a suitable MFD system!
- We know that zeros of the plant cannot be replaced, but we can modify the behavior of a system by designing zeros and add them to the closed loop system. In the case of unity and output feedback configuration, a pre-compensator is added to the closed loop system to place the desired block zeros.

In order to obtain these results, a review on matrix polynomials has been given, and two methods to construct solvents from a set of latent values and latent vectors, and to obtain the inverse of a matrix polynomial have been developed. Then the two needed system descriptions, SSD and MFD, have been presented. As a result the relationship between eigenvectors and latent vectors has been established and as a consequence, a method to solve the polynomial eigenvalue problem has been obtained. The feedback control approaches for systems either in SSD or MFD have been detailed, then the proposed design approach has been presented. As an application, an input-output feedback compensator, to assign a desired eigenstructure for a helicopter flight control, has been designed.

7.2 Contributions

To the best of the author's knowledge, the list below describes the novel contributions of this work:

1. A method to construct right and left solvents of a matrix polynomial from a set of latent values and latent vectors has been developed. The conditions have been summarized in a theorem.

2. The inverse of a matrix polynomial is given as a particular case of the block partial fraction expansion of the related rational matrix.

3. The relationship between the eigenstructure of the state matrix of a system described in state space equations and the latent structure of the denominator of a matrix fraction description of the same system has been established by two theorems and two corollaries.

4. As a consequence of this relationship, a method to determine the latent values (eigenvalues) of a matrix polynomial has been developed.

5. A method to design a compensator for systems described in MFD by assigning a set of desired latent values and latent vectors has been developed.

6. The same method can be used for systems described in SSD, by starting with a conversion of the SSD into a MFD and the conversion of a desired set of eigenvalues and eigenvectors (eigenstructure) into a desired latent structure, thus achieving an eigenstrucure assignment for systems described in matrix polynomials.

7. Finally, a matrix fraction description for a helicopter flight system has been developed and a compensation system, by assigning a desired eigenstructure, has been obtained.

7.3 Comments and problems

In the following some comments on the results obtained during this research work and the questions generated but not yet answered.

- The input-Output feedback configuration allows placing block zeros using the fixed compensator denominator so for some applications, this configuration will be preferred.

- The degree of compensator will depend on the degree of the desired closed loop denominator. The latter depends on the number of desired block roots (poles or zeros) to place. If the solution of the Diophantine equation gives a non-proper, or a non-stable compensator (for other configurations than the input-output) the degree of the compensator may be increased, by increasing the degree of the closed loop denominator! The latter can be increased by placing arbitrary block roots

- If the set of desired latent values and latent vectors is not suitable (linearly independent vectors), then other arbitrary latent values with corresponding latent vectors may be chosen so that compensation is possible.

- The arbitrary roots may be chosen to improve the performance of the closed loop system behavior (robustness and sensitivity) or simply to not interfere on it.

- We can only place (additional) zeros but not replace some zeros by other desired ones.

- Some closed loop configurations allow zeros placement (Input-Output). The poles of the compensator or the pre-compensator become the zeros of the closed loop system.

- If a latent root is complex it must be composed with its conjugate and their corresponding complex latent vectors. Block roots are real.

- If the system is both block controllable and block observable, we will chose either RMFD or LMFD depending on the smallest value between the two dimensions number of inputs or number of outputs.

- The difficulty of the proposed method (if adequate latent roots and latent vectors exist) is in the inversion of the block Vandermonde matrix, and in solving the Diophantine equation. For the first, a recursive algorithm has been developed, and a future paper on block Vandermonde matrices (including this algorithm and others) is under realization. For the second, the conversion method used in this approach results in systems satisfying the conditions of existence of a solution, furthermore a parallelization of the Gaussian elimination algorithm, used to search for linearly dependent vectors in a matrix, has been developed and it is presented in Appendix B.

- The conversion method used in the proposed design process guaranteed that the Diophantine equation will have a solution. Many works have concerned the resolution of the Diophantine equation to get solutions which are proper. But, we found that the resolution of the Diophantine equation does not always give solutions which are stable, i.e., the poles of the solution (the denominator of the compensator in our case) are not in the open left half s-plane. It is obvious that the stability of the compensator is required. One solution to this problem is to determine compensators with higher degrees, which means increasing the degree of the desired closed loop denominator by adding arbitrary latent values (block poles) to be assigned with the desired ones.

- It is stated that, unless the dynamic compensator is required and explicitly defined by the closed loop system specification, such structures do not currently represent a flexible way of increasing the design freedom available for EA, but we showed that even for a system described in SSD, we can design a stable dynamic compensator and achieve the assignment of the whole set of desired eigenvalues and eigenvectors easily and efficiently.

7.4 Future works

Assigning latent structure opens a wide area of research; all the results obtained through eigenstructure assignment may be applied in the frequency domain to systems described in either representation (SSD, MFD, TF). The results obtained during this research work generated many questions and problems for which solutions are to be explored:

- If the number of eigenvalues to assign is not suitable (to make compensation), or we need to increase the degree of the closed loop denominator (to have stable compensator), the idea is to add block roots to assign and thus increase the design freedom to achieve performance specifications (sensitivity, robustness, etc.)

- Now that zeros (block zeros) may be determined from latent roots and latent vectors their effect on a closed loop system may be better studied.

- Many research works have been done on the specification of eigenvalues and eigenvectors for multivariable systems to achieve a certain desired behaviour or performance of the system, but (to my knowledge) nothing has been done for latent structure specifications.

- Robustness and sensitivity of the closed loop system can be studied and explored through either: latent values/eigenvalues, latent values and vectors of the block poles, or block poles themselves.

- In State space description, we have seen that eigenvectors are used to shape the system responses, so what will be the role of latent vectors for systems in matrix fraction description?

- Many feedback configurations exist, and the idea is to choose the right configuration for a particular closed loop system. The choice criteria have to be explored.

- Explore the problem of the resolution of the Diophantine equation and determine the conditions for which a solution is stable.

- Because of the diversity of fields of application of the first degree Diophantine equation, a future task will be the parallelization of the totality of its resolution.

- Dynamic compensation has been shown to offer additional design freedom to the eigenstructure assignment problem, but it is clear from the literature that methods for exploiting this freedom are not well developed. Therefore, a significant area of further study would be to develop a meaningful description of the additional benefits available to the control system designer via the specification of compensator eigenvectors as well as eigenvalues.

Finally, parallelization may be a good option to improve the efficiency and performances of resolution methods in many fields, more precisely in control theory and systems engineering.

Appendix A

Diophantine equation and its solutions

The polynomial Diophantine matrix equation (PDME) plays a very important role in the analysis and design of control systems such as compensators design [1], multivariable adaptive feedback systems [2], predictive control, robust controllers design [3], etc.

In this thesis, we showed that to place block poles using feedback for systems in matrix fraction description, the design method leads to the resolution of a Diophantine equation, called also compensator equation, to determine the compensator's numerator and denominator.

Because of the great use of PDME, finding solutions to this kind of equations has received much attention in the past several decades. Generally speaking, all the methods can be classified into three main categories: the state-space related approaches [4, 5], the Taylor series treatment [6, 7], and methods involving coefficient matching [1, 8, 9].

In [10] a global parametrization of the solutions to the PDME has been given.

Lai, in [7], presented a procedure involved in solving a set of linear equations which will result in a unique solution if the solution exists, and in [4], based on the state-space concepts, Fang proposed a simple approach to find all solutions of the PDME and all solutions are expressed in explicit formula forms.

In the category of methods based on coefficient matching, Chen's method [1] uses a numerical QR algorithm to look for a nonsingular matrix as a solution, and the method in [11] sets up equations in a different way such that an analytical method can be used to guarantee the non-singularity of the square system thus formed.

In [12] the problem is firstly transformed into solving a class of Stein equations, one of whose coefficient matrices are nilpotent, thus obtaining explicit solutions in a finite summation form, and an exhaustive literature review on methods to solve the PDME is also given.

A.1 Introduction

Diophantine equations are the best example of matrix problems, their resolution is not an easy task since a direct resolution method is difficult to find. Several methods of resolution exist, and most of them seek to determine linear dependent rows in a resultant matrix [7].

A.1.1 Definition

Diophantine equation is of the form $P(x, y, z, ...) = 0$ where P is a polynomial with integer coefficients (or rational). We look for the roots in N (natural integers) or Q (rational numbers). A set of p-tuples (a_1,a_2, ..., a_p) is called Diophantians if the Diophantine equation $P(x, y, z, ...) = 0$ with coefficients (a_1,a_2, ..., a_p) has at least one solution.

Some classical examples of Diophantine equations are: the study of the general shape of the Pythagoreans triplets, Bezout theorem, the general solution of the integer equation $ax+by = c$, the famous theorem of Fermat, Pell's equation, etc. [13].

A.1.2 Fields of application

There are various fields which use the Diophantine equation such as biology, chemistry, control, computer science, mathematics and others. The following are examples:

- Computer: In 2007, an ID-based digital signature scheme based upon the difficulty of elliptic curve discrete logarithm problem (ECDLP) has been proposed. However, a security flaw has been found and a feasible attack is proposed by [14]. This attack enables an attacker to easily obtain the signer's secret key without facing the difficulty of ECDLP by using the technique of solving the linear Diophantine equation.

- Systems Engineering: An application of simple continuous-time robust regulators of PI and PID type is designed through general solutions of Diophantine equations, in the ring of proper and stable rational functions, to control temperature in a laboratory model of a hot-air tunnel, whose mathematical model is supposed to include parametric uncertainty [15].

- Biology: A method for the inverse design of small ligands has been developed in [16]. A key step in this method involves computing the Hilbert basis of a system of linear

Diophantine equations.

- Control theory: Feedback compensation in systems described by matrix polynomials lead, in general, to solving a first degree Diophantine equation (see Chapter 4).

A.1.3 Resolution of the second degree Diophantine equation

Such an equation is in the form:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0 \tag{A.1.1}$$

where a, b, c, d, e and f are integers, and at least one of the numbers a, b, c is different from zero.

The solution of such equation is a task much more complicated than the resolution of the linear system of equations. A complete solution exists and is due to Joseph-Louis Lagrange (1769) [13].

The following techniques have been proposed:

- Modular Arithmetics: It can be employed to show that a solution of a given Diophantine equation does not exist. Specifically, if the equation in question is proven to be never true for a certain integer m, then it is proven that the equation is false. However, this technique cannot be employed to prove that the solutions exist [13].

- Induction: When some solutions were found, induction can be employed to find a family of solutions. The techniques such as the infinite descent can also prove that the solution of a particular equation does not exist, or that the solution apart from particular family does not exist [13].

A.2 Polynomial Diophantine matrix equation

In this thesis we are interested in solving the polynomial Diophantine matrix equation.

A.2.1 Definition

Let a p-inputs q-outputs system be described in RMFD or LMFD as follows:

$$G(s) = N(s)D^{-1}(s) \tag{A.2.1}$$

or

$$G(s) = D^{-1}(s)N(s) \tag{A.2.2}$$

The famous Diophantine equation is stated as follows:

Definition A.2.1. Given matrix polynomials $H(s)$, $D(s)$ and $N(s)$, we need to determine $X(s)$ and $Y(s)$, both matrix polynomials with convenient dimensions, by solving the following equation:

$$\text{"right"} : H(s) = X(s)D(s) + Y(s)N(s) \tag{A.2.3}$$

$$\text{"left"} : H(s) = D(s)X(s) + N(s)Y(s) \tag{A.2.4}$$

such that $X(s)$ and $Y(s)$ have a minimum degree.

Remark A.2.1. In this thesis, H is the desired closed loop system denominator (D_f), D and N are the plant transfer function, and X and Y are the numerator and denominator of a unity or output feedback compensator, or the two numerators of an input-output feedback compensators (see chapters 4 and 5).

A.2.2 Existence of a solution

We start with some recalls.

Definition A.2.2. A rational matrix $G(s)$ is said to be proper (strictly proper) if $G(\infty)$ is a finite constant (zero) matrix.

Definition A.2.3. A non singular $p \times p$ polynomial matrix $M(s)$ is said column reduced if $deg(detM(s)) = \sum_{i=1}^{p} \delta_{ci}M(s)$ and it is said row reduced if $deg(detM(s)) = \sum_{i=1}^{p} \delta_{ri}M(s)$ where $\delta_{ci}M(s)$ and $\delta_{ri}M(s)$ are respectively the max column degrees and the max row degrees of $M(s)$.

Definition A.2.4. Let a polynomial matrix $M(s) = M_{hc}H_c(s) + M_{lc}(s)$ where H_c=diag{higher column degrees of $M(s)$}, M_{hc} is a constant matrix of coefficients with higher column degrees of $M(s)$, and M_{lc} is the rest of $M(s)$. Then $M(s)$ is said column-reduced iff M_{hc} is nonsingular.

And, by duality, let $M(s) = H_r(s)M_{hr} + M_{lr}(s)$ where: H_r=diag{higher row degrees of $M(s)$}, M_{hr} is a constant matrix of coefficients with higher row degrees of $M(s)$ and M_{lr} is the rest of $M(s)$. Then $M(s)$ is said row-reduced iff M_{hr} is non singular.

Theorem A.2.1. *Let $N(s)$ and $D(s)$ be $q \times p$ and $p \times p$ polynomial matrices and let $D(s)$ be column reduced, then the rational function $N(s)D^{-1}(s)$ is proper (strictly proper) iff $\delta_{ci}(N(s)) \leqslant \delta_{ci}(D(s))$ ($\delta_{ci}(N(s)) < \delta_{ci}(D(s))$) for $i = 1..p$.*

Proof: see [1]

A similar theorem exist for the row-reducedness [1].

The following theorem states the condition for the existence of a solution to the Diophantine equation:

Theorem A.2.2. *Let $G(s) = N(s)D^{-1}(s)$ being a proper system, k_i $i = 1..p$ being the column degrees of $D(s)$ and γ the row index of $G(s)$. If for $l \geq \gamma - 1$ $\forall H(s)$ with column degrees $\leq l + k_i$ $\exists X$ and Y of row degrees $\leq l$ such that (X, Y) verifies equation A.2.3 iff D and N are right co-prime and $D(s)$ is column reduced.*

Proof: see [1].

By analogy, Let $G(s) = D^{-1}(s)N(s)$ being a proper system, k_i $i = 1..q$ being the row degrees of $D(s)$ and μ be the column index of $G(s)$. If for $l \geq \mu - 1 \; \forall \; H(s)$ with row degrees $\leq l + k_i$ then $\exists X$ and Y of column degrees $\leq l$ such that (X,Y) verifies equation A.2.4 iff D and N are left co-prime and $D(s)$ is row reduced.

Remark A.2.2. It has been stated that if a solution exists then it is unique [7].

A.3 Resolution methods

Many authors proposed methods to solve the Diophantine equation. Two simple methods (Chen's method [1] and Lai's method [7]) have been adopted for this research work and will be presented.

A.3.1 Chen's Method

The following solving method is detailed and proved in [1].

Let

$$\begin{array}{l} D(s) = \sum_{i=0}^{n} D_i s^i \\ N(s) = \sum_{i=0}^{n} N_i s^i \end{array} \tag{A.3.1}$$

$$\begin{array}{l} X(s) = \sum_{i=0}^{m} X_i s^i \\ Y(s) = \sum_{i=0}^{m} Y_i s^i \end{array} \tag{A.3.2}$$

And

$$H(s) = \sum_{i=0}^{n+m} H_i s^i \tag{A.3.3}$$

where n is the highest column degree of $D(s)$ (or its degree)

So equation A.2.3 will become:

$$\begin{pmatrix} X_0Y_0 & X_1Y_1 & \cdots & X_mY_m \end{pmatrix} S_m = \begin{pmatrix} H_0 & H_1 & \cdots & H_{n+m} \end{pmatrix} \tag{A.3.4}$$

Where

$$S_m = \begin{pmatrix} D_0 & D_1 & \cdots & D_n & 0 & 0 & \cdots & 0 \\ N_0 & N_1 & \cdots & N_n & 0 & 0 & \cdots & 0 \\ 0 & D_0 & \cdots & D_{n-1} & D_n & 0 & \cdots & 0 \\ 0 & N_0 & \cdots & N_{n-1} & N_n & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & \ddots & \ddots & & & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & D_0 & D_1 & \cdots & D_n \\ 0 & 0 & \cdots & 0 & N_0 & N_1 & \cdots & N_n \end{pmatrix} \tag{A.3.5}$$

where 0 stands for a null matrix of right dimensions and S_m is a matrix which consists of $m+1$ blocks of p "D" rows and q "N" rows. If ND^{-1} is proper then each "D" row is linearly independent from the previous "D" rows. But the N rows are not. Let n be the degree of $G(s)$ and γ the row index of $G(s)$, it is also the observability index of any irreducible realization of $G(s)$, then the following theorem states the condition for equation A.3.4 to have a solution.

Theorem A.3.1. *Let a (strictly) proper rational matrix $G(s) = N(s)D^{-1}(s)$ with $deg(D) = n$ and let γ the row index of $G(s)$. For any $H(s)$ of degree $n+m$ there exist a (strictly) proper solution $X^{-1}(s)Y(s)$ with $deg(X) = m$ for equation A.2.3 iff ($m > \gamma$) $m \geq \gamma$ and D and N are right co-prime and $D(s)$ is column- reduced.*

Proof: see [1]

The same development can be done for equation A.2.4. If we replace equations A.3.1, A.3.2 and A.3.3 into equation A.2.4 we obtain:

$$T_m \begin{pmatrix} X_0 \\ Y_0 \\ X_1 \\ Y_1 \\ \vdots \\ X_m \\ Y_m \end{pmatrix} = \begin{pmatrix} H_0 \\ H_1 \\ \vdots \\ H_{n+m} \end{pmatrix} \tag{A.3.6}$$

where

$$T_m = \begin{pmatrix} D_0 & N_0 & 0 & 0 & \vdots & & \vdots & 0 & 0 \\ D_1 & N_1 & D_0 & N_0 & \vdots & & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & 0 & 0 \\ D_n & N_n & D_{n-1} & N_{n-1} & \vdots & \ddots & \vdots & 0 & 0 \\ 0 & 0 & D_n & N_n & \vdots & \ddots & \vdots & D_0 & N_0 \\ 0 & 0 & 0 & 0 & \vdots & & \vdots & D_1 & N_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \vdots & & \vdots & D_n & N_n \end{pmatrix} \tag{A.3.7}$$

Here 0 stands for a null matrix of right dimensions.

Matrix T_m has $m+1$ block columns formed from the matrix coefficients of $D(s)$ and $N(s)$. This matrix is searched for linearly dependent columns from left to right. Let γ be the column index of $G(s)$ or the controllability index of any realization of $G(s)$, then a similar theorem states the condition for equation A.3.6 to have a solution, and if a solution exists then it is unique.

Theorem A.3.2. *Let a (strictly) proper rational matrix $G(s) = D^{-1}(s)N(s)$ with $deg(D) = n$ and let γ the column index of $G(s)$. For any $H(s)$ of degree $n+m$ there exist a (strictly) proper*

solution $Y(s)X^{-1}(s)$ *with* $deg(X) = m$ *for equation A.2.4 iff* $(m > \gamma\ )\ m \geq \gamma$ *and* D *and* N *are left co-prime and* $D(s)$ *is row- reduced.*

Proof: see [1]

A.3.2 Lai's Method

The author in [7] proposed an improvement of the previous method, and showed the uniqueness of a solution if it exists.

Equation A.2.3 may be written in the following form:

$$\begin{pmatrix} X(s) & Y(s) & I \end{pmatrix} \begin{pmatrix} D(s) \\ N(s) \\ -H(s) \end{pmatrix} = 0 \tag{A.3.8}$$

where $D(s)$ and $N(s)$ are as defined in equation A.3.1 and $H(s) = \sum_{i=0}^{l} H_i s^i$, n is the highest degree of s in $D(s)$ and $N(s)$ and $l \geq n$ the highest degree in $H(s)$. All the elements of D, N and H are real constant matrices not necessarily all non-zero.

Given $D(s)$, $N(s)$ and $H(s)$, if a solution exists, then a matrix $\begin{pmatrix} X(s) & Y(s) & I \end{pmatrix}$ can always be found to satisfy equation A.3.8 such that the highest degree of $X(s)$ and $Y(s)$ is $m = l - n$. In this case, equation A.3.8 is transformed into a system of algebraic equations with $X(s)$ and $Y(s)$ as defined in equation A.3.2.

The resulting system of equations is of the form:

$$\begin{pmatrix} X_0 & Y_0 & I & X_1 & Y_1 & \cdots & X_m & Y_m \end{pmatrix} Z_m = 0 \tag{A.3.9}$$

Where 0 stands for the null matrix of right dimensions and

$$Z_m = \begin{pmatrix} D_0 & D_1 & \cdots & D_n & 0 & 0 & \cdots & 0 \\ N_0 & N_1 & \cdots & N_n & 0 & 0 & \cdots & 0 \\ -H_0 & -H_1 & \cdots & \cdots & -H_l & 0 & \cdots & 0 \\ 0 & D_0 & \cdots & D_{n-1} & D_n & 0 & \cdots & 0 \\ 0 & N_0 & \cdots & N_{n-1} & N_n & 0 & \cdots & 0 \\ \vdots & & \ddots & & & \ddots & & \vdots \\ \vdots & & & \ddots & & & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & D_0 & D_1 & \cdots & D_n \\ 0 & 0 & \cdots & 0 & N_0 & N_1 & \cdots & N_n \end{pmatrix}$$

The solution of equation A.3.9 consists in determining the null space of the resultant matrix Z_m.

A.4 Illustrative example

The example is from ([1], page 484).

Let a system be described by this RMFD:

$$G(s) = N(s)D^{-1}(s) = \begin{pmatrix} s^2+1 & s \\ 0 & s^2+s+1 \end{pmatrix} \begin{pmatrix} s^2-1 & 0 \\ 0 & s^2-1 \end{pmatrix}^{-1}$$

And let the desired matrix $H(s)$ be equal to:

$$H(s) = \begin{pmatrix} (s+1)^4 & 0 \\ 0 & (s+1)^2(s^2+s+1) \end{pmatrix}$$

Then we can rewrite $N(s)$ and $D(s)$ as follows:

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

$$D(s) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} s^2 + \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$$

And $H(s)$ as follows:

$$H(s) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} s^4 + \begin{pmatrix} 4 & 0 \\ 0 & 3 \end{pmatrix} s^3 + \begin{pmatrix} 6 & 0 \\ 0 & 4 \end{pmatrix} s^2 + \begin{pmatrix} 4 & 0 \\ 0 & 3 \end{pmatrix} s + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

The degree of the solution in this case is 2.

Then we can rewrite $X(s)$ and $Y(s)$ as follows:

$$\begin{cases} X(s) = X_2 s^2 + X_1 s + X_0 \\ Y(s) = Y_2 s^2 + Y_1 s + Y_0 \end{cases}$$

Then equation A.3.4 becomes:

$$\begin{pmatrix} X_0 Y_0 & X_1 Y_1 & X_2 Y_2 \end{pmatrix} S_2 = \begin{pmatrix} 1 & 0 & 4 & 0 & 6 & 0 & 4 & 0 & 1 & 0 \\ 0 & 1 & 0 & 3 & 0 & 4 & 0 & 3 & 0 & 1 \end{pmatrix} \tag{A.4.1}$$

Where the 12×10 matrix S_2 is given by:

$$S_2 = \begin{pmatrix} -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Then using the row-searching algorithm to find the linearly independent rows of S_2 and solving equation A.4.1 (using Matlab for example) we obtain the following results:

$$X(s) = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} s^2 + \begin{pmatrix} 0 & -1.3333 \\ 0 & -1 \end{pmatrix} s + \begin{pmatrix} -3 & 1.3333 \\ 0 & -1 \end{pmatrix}$$

$$Y(s) = \begin{pmatrix} -4 & 1.3333 \\ 0 & -2 \end{pmatrix} s + \begin{pmatrix} -4 & 1.3333 \\ 0 & -2 \end{pmatrix}$$

Or in a better form:

$$X(s) = \begin{pmatrix} s^2+3 & \frac{4(s-1)}{3} \\ 0 & s^2+s+1 \end{pmatrix} \text{ and } Y(s) = \begin{pmatrix} 4s+4 & \frac{-4(s+1)}{3} \\ 0 & 2(s+1) \end{pmatrix}$$

A.5 Conclusion

Many methods exist to search for linear independent vectors but the row-searching algorithm is the most used for its feasibility and efficiency. That is why we chose it to be parallelized and the results are presented in the next appendix.

Appendix Bibliography

[1] C. T. Chen, Linear System Theory and Design, Holt Rinehart and Winston, NY, 1984.

[2] K. S. Narendra and A. M. Annaswamy, Stable Adaptive Systems. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[3] G. C. Goodwin, K. Sin and K. Saluja, "Stochastic adaptive control and prediction-The general delay-colored noise case", IEEE Trans. on Automat. Contr., Vol. AC-25, Issue. 5, pp 946-950, 1980.

[4] C. H. Fang, "A simple approach to solving Diophantine equation", IEEE Trans. Automat. Contr., vol. AC-37, pp. 152-155, 1992.

[5] M. Yamada, P. C. Zue, and Y. Funahashi, "On solving Diophantine equations by real matrix manipulation", IEEE Trans. Automat. Contr., vol. AC-40, pp. 118-122, 1995.

[6] F. Feinstein and Y. Bar-ness, "The solution of the matrix polynomial equation A(s)X(s) + B(s)Y (s) = C(s)", IEEE Trans. Automat. Contr., vol. AC-29, pp. 75-77, 1984.

[7] Y. S. Lai, "An algorithm for solving the matrix polynomial equation A(s)X(s) + B(s)Y (s) = C(s)," IEEE Trans. Circuit Syst., vol. 36, pp. 1087-1089, 1989.

[8] B. Buchberger, "Groebner bases: An algorithmic method in polynomial ideal theory", N. K. Bose (Ed.), Multidimensional Systems Theory, Reidel, Dordrecht, pp. 184-232, 1985.

[9] Z. Lin and L. Xu (Eds.), Special Issue on Applications of Groebner bases multidimensional systems and signal processing," J. Multidim. Syst. Signal Processing, Vol. 12, pp. 215-381, 2001.

[10] W. A.Wolovich and P. J. Antsaklis, "The canonical Diophantine equations with applications", SIAM J. on Contr. Optimiz., Vol. 22, No. 5, pp. 777-787, 1984.

[11] K. S. Yeung and T. M. Chen, "Solving Matrix Diophantine Equations by Inverting a Square Nonsingular System of Equations", IEEE Trans. Circ. Syst., Vol. 51, No. 9, pp. 488-495, 2004.

[12] H. Kong, B. Zhou and M.-R. Zhang, "A Stein Equation Approach for Solutions to the Diophantine Equations", In 2010 Chinese Control and Decision Conference proceedings, pp. pp. 3024-3028, 2010

[13] N. P. Smart, "The Algorithmic Resolution of Diophantine Equations", Hewlet-Packard Laboratories, Bristol; 1998.

[14] J. H. Yang, and C. C. Chang, "Cryptanalysis of ID-based digital signature scheme on elliptic curve cryptosystem", In Proceedings of the international Conference on Intelligent Systems Design and Applications, November 26-28, Taiwan, pp. 3-5, 2008.

[15] R. Matusu, R. Prokop and M. Dlapa, "Robust control of temperature in hot-air tunnel", In proceedings of the Mediterranean Conference on Control and Automation Congress Centre, France, pp. 576-581, 2008.

[16] S. Martin, W.M. Brown, J.L. Faulon, D. Weis, V. Visco and J. Kenneke, "Inverse design of large molecules using linear diophantine equations", Computational Systems Bioinformatics Conference Workshops, Stanford, USA, pp. 11-14, 2005.

Appendix B

Row searching algorithm

Looking for linearly independent vectors in a set of vectors is an important step in solving the Diophantine equation and, in determining the controllability and observability indices. The row searching algorithm is the most used algorithm for looking for linearly dependent rows of a matrix. The principle is to be found in [1].

The following result have been communicated and published in [2].

B.1 Introduction

Parallel programming techniques, which are not very well used in the world of scientific computation, start to be democratized. All the great control software applications must be reconsidered in terms of parallel (and distributed) tasks in order to exploit the performance of the new-coming processors as much as possible.

Searching for linear dependent vectors (in a matrix) is one of the problems which require parallelism; it is a fundamental aspect in linear algebra. In this kind of problem one has often to deal with large systems of matrices, which requires much time and memory capacity, in addition to the fact that the methods of resolution are in general iterative.

The present study aims to parallelize the row searching algorithm to make an improvement in the resolution of the Diophantine equation then its implementation (of the algorithm) using Message Passing Interface [3, 4], because standard MPI provides a parallel programming environment and a synchronization between the various tasks using communications by message transfer.

The sequential algorithm and its two proposed parallel forms are implemented and evaluated using algorithmic complexity.

B.2 Description of the Row-Searching algorithm

The row-searching algorithm, also called Gaussian elimination, consists in searching for linearly independent rows of a $n \times n$ matrix A from top to bottom. So initially the first row is verified if not zero, then it checks if the second row is linearly independent of the first one. At the k^{th} step we check if the k^{th} row of matrix A is linearly independent of its (k-1) preceding ones. If a row is linearly dependent on the rows which precede it, it must be eliminated according to this consideration, moreover the coefficients of the linear combination of the dependent row are determined [1].

This algorithm has as input the square matrix A whose treatment to be carried out consists of two distinct but not independent phases.

Phase I

In this phase matrix A is evaluated so as to detect and eliminate the linear dependencies between its various rows. To be done, we need a vector K of dimension equal to the number of rows of matrix A and construct a matrix F. Phase I is carried out in a recursive way for all the rows of A.

step1: First we need to compute the pivot. The pivot is often selected according to a defined criterion; if not, a computing formula will be necessary.

step2: The vector K_i is initialized to the elementary vector e_i (e_i is the vector whose i^{th} element is 1 and the rest 0);

step3: The vector K_i is determined according to the pivot and column of A of the associated iteration as follows:

$$\begin{cases} \text{for } j = 1 + 1 \text{ to } n \\ \quad K_i(j) = \frac{-A_i(i,j)}{pivot_i} \\ \text{endfor} \end{cases}$$

At iteration i, the first step consists in choosing the pivot. According to the value of pivot two cases arise: If the pivot is null then the calculation of the new vector K_i is impossible and this means that row i is dependent on all the linearly independent lines preceding it in A. Else, at step two, the vector K_i is initialized then will be calculated at step three.

At the i^{th} iteration the computed vector K_i is allotted to i^{th} column of a matrix F, which is initialized at the beginning of phase I to the matrix identity. After each calculation of

K_i, the product $K_i * A_{i-1} = A_i$ is calculated, where each iteration depends on the result of its preceding one.

The result of Phase I is a matrix F containing all the vectors K_i (concatenated) and the matrix resulting from the last iteration applied to A, where the rows linearly dependent on their preceding ones are null.

Phase II

The second phase consists in determining the coefficients of combination between each dependent row identified in Phase I and its preceding ones using the matrix $F = [f_{ij}]$, this treatment gives as output a matrix $B = [b_{ij}]$ whose diagonal is all 1's. For example to determine the j^{th} row of B:

$$\begin{cases} B(j,j) = 1 \\ \text{for } k = j-1 \text{ to } 1 \\ \quad B(j,k) = \sum_{p=k+1}^{j} b_{jp} * f_{pk} \\ \text{endfor} \end{cases}$$

The resolution steps are summarized in algorithm 1.

Algorithm 1 Sequential Algorithm

```
for i=1 to n do
  choose a pivot;
  if (pivot≠0) then
    compute elements of K_i;
    Put K_i in F(i,*);
    prod(K_i, A_{i-1});
  else
    i^th row is linearly dependent;
  end if
end for
for i=1 to n do
  if Lin_Dep(i) = 0 then
    compute elements of B(i,*)
  end if
end for
```

B.3 Parallelization

B.3.1 Working frame

The working environment used is MPI (Message Passing Interface). It gives the opportunity for parallel application programmer to develop portable applications while exploiting the per-

formances of the parallel machines as much as possible [3, 4].

This library offers routines to programmers to parallelize a sequential application which will be executed on various processes and even on various machines (distributed). The number of processes must be fixed at the beginning (at the start of the environment), that is why the parallelization given here concerns a fixed number of processes (that we can generalize).

For the data distribution the following functions of MPI are used:

- `MPI_Scatter`: To distribute data on several processes.
- `MPI_Gather`: To allow a master process to gather results from its slave processes.
- `MPI_Bcast`: To perform a complete diffusion (of all data on all processes) and synchronize the slave processes.

B.3.2 Parallelization approaches

A first parallelization approach is the reduction of the matrix structure of K into a vector structure, implying some changes in the method of calculation of Phase I only, and a second approach will represent a real data decomposition parallelization.

Approach 1

This solution comes from the fact that the vision of the data structure in which must be represented K changed into a vector. This approach allows to implement a parallel row-searching by giving more flexibility to computations, the different matrices A_i $i = 1..n$, are not seen any more like fixed entities but as structures to be easily divided between various processes for treatment, The computation of the matrix A_i in function of K and A_{i-1} is not done any more by one single process, but by several.

The changing of the data structure of K produced several changes in the implementations of the parallel row-searching algorithm.

- The ordinary matrix product was replaced by a matrix-vector product which implies a new computation formula of elements of A_i:

$$\begin{cases} \text{for } j = 0 \text{ to } n \\ \quad \text{for } k = 0 \text{ to } n \\ \qquad A_i(j,k) = K_i(j) * A_{i-1}(i,k) + A_{i-1}(j,k) \\ \quad \text{endfor} \\ \text{endfor} \end{cases}$$

- The distribution of data between the various processes contributing to calculation becomes possible.

The function `MPI_Scatter` is used first, then it is followed by `MPI_Gather`, which respectively carries out the distribution and the collect of the data of the process master (PID=0) towards the others and inversely.

`MPI_Scatter` is used to decompose the matrices A_i between the different processes, by allotting to each one a portion of A_i. It implies that each process receives a certain number of entire rows from the process master (included).

After each treatment, each process will have to return the same recomputed rows to the master for the recovery and the reconstruction of the new matrix A_i by using `MPI_Gather`.

The distribution of data on several processes is illustrated in figure B.1.

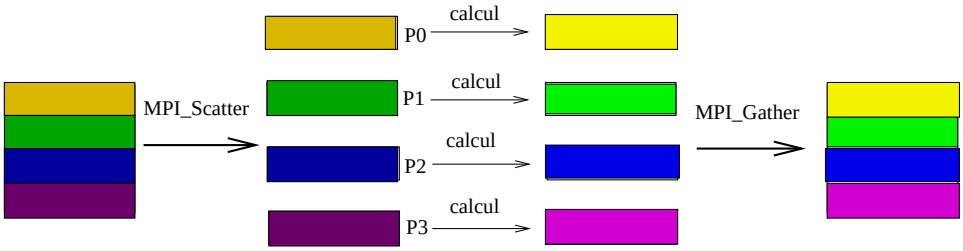


Figure B.1: Data distribution of approach 1

At each i^{th} iteration: the process P_0 only makes the selection of the pivot and initializes the vector K_i, then all the processes (P_0,P_1,P_2,P_3) executes the diffusion routine `MPI_Bcast()` whose contents are K_i, the function of distributing data `MPI_Scatter()` whose contents is A_{i-1} and a parallel execution of the procedure *Prod()* to multiply the local vector K_i of each process and the received portion of the matrix A_{i-1} which in the figure 1 is represented by "*calcul*". Phase I is finished with the shared execution of the procedure of collection of the calculated data. Then the process P_0 only deals with calculations necessary to the completion of phase II.

The resolution method is summarized in algorithm 2.

Approach 2

This second approach goes further in the division of data between the computing processes belonging to a communicator. A communicator defines a space of communication between only the processes belonging to it. The communicator by default, `MPI_Comm_world`, encompasses all the processes and thus defines a communication space for all of them.

Algorithm 2 Parallel Algorithm of approach1

```
Partition A to determine part;
for i=0 to n do
  if PID=0 then
    choose pivot;
  end if
  if pivot≠ 0 then
    if PID=0 then
      compute elements of K_i;
      Put K_i in F(i,*);
    end if
    MPI_Bcast(K_i, n, MPI_INT, 0, MPI_COMM_World);
    MPI_Scatter(A_{i-1}, m*part, MPI_INT, 0, MPI_COMM_WORLD);
    Prod(K_i, AA));
    MPI_Gather(A_i, m*part, MPI_INT, AA, m*part, MPI_INT, 0, MPI_COMM_WORLD);
  else
    Row i is linearly dependent;
  end if
end for
for i=0 to n do
  if PID=0 then
    if Lin_Dep(i) = 0 then
      Compute elements of B(i,*)
    end if
  end if
end for
```

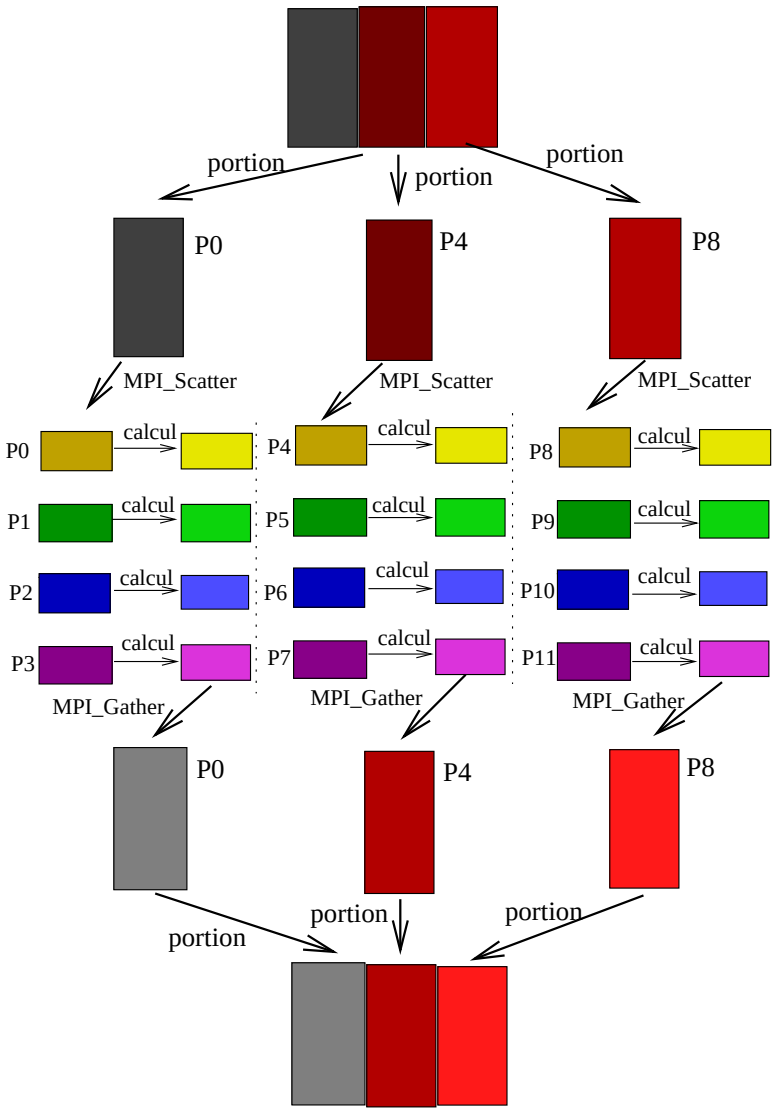


Figure B.2: Data distribution of approach 2

A first distribution of portions of whole columns, between a certain number of processes belonging to communicators which are two by two distinct, is carried out.

After this decomposition of the initial matrix, inside each of the communicator the processes will apply an `MPI_Scatter` similar to that of approach 1 followed by an `MPI_Gather` also, as illustrated in figure B.2.

The total space is decomposed in three communicators (respectively Com_0, Com_1, Com_2) containing each four processes where a local process master is designed (respectively P_0, P_4, P_8). Each one of them will carry out the procedure of data distribution *Portion()*, which makes a distribution of columns.

Inside each communicator the same algorithm of approach 1 is performed: Sharing of K_i and the portion obtained by the execution of the procedure *Portion()* to which is applied

`MPI_Scatter`(), so as to have a block per process to carry out *Prod()* in parallel, then collecting data local to each communicator by `MPI_Gather`() for the reconstruction of the matrix resulting from the iteration by a re-execution of *Prod()*, until the treatment of all the iterations of phase I. Finally the process P_0, only, does the calculations of phase II.

Algorithm 3 summarizes the precedent steps.

Algorithm 3 Parallel Algorithm of Approach2

```
for i=0 to n do
  if PID=0 then
    Choose pivot;
  end if
  if pivot≠ 0 then
    if PID=0 then
      Compute elements of K_i;
      Put K_i in F(i,*);
    end if
    MPI_Bcast(K_i, n, MPI_INT, 0, MPI_COMM_WORLD);
    for id=0 to p do
      process=id MOD SQRT(p);
      if process=0 then
        Portion(A_i), racine=id;
      end if
      MPI_Scatter(A_{i-1},n * m/SQRT(p), MPI_INT, AA, n * m/SQRT(p), MPI_INT, racine,
      Comm_id);
      Prod(K_i, AA);
      MPI_Gather(A_i, n * m/SQRT(p),MPI_INT, AA, n*m/SQRT(p), MPI_INT, racine,
      Comm_id);
      if process=0 then
        portion(A_i);
      end if
    end for
  else
    Row i is linearly dependent;
  end if
end for
for i=0 to n do
  if PID=0 then
    if Lin_Dep(i) = 0 then
      Compute elements of B(i,*);
    end if
  end if
end for
```

Table B.1: Algorithmic Complexities

Sequential Algorithm	$\theta(n^2 \times m)$
Parallel Algorithm 1	$\theta(n^2 \times m/p)$
Parallel Algorithm 2	$\theta(n^2 \times m/p^{3/2})$

B.4 Evaluation

To evaluate the parallel algorithms, a distributed implementation is realized on a two computers running a Debian GNU/Linux and an MPI implementation: MPICH [5]. Of course the sequential row-searching algorithm and the two parallel algorithms (Approaches 1 and 2) are implemented, and the following evaluation concerns a number of p processors.

The algorithmic complexity of the sequential row-searching algorithm and the two proposed parallelization approaches will be approximated using the notation θ in function of the entry matrix $A(n, m)$, the vector $K(n)$, the coefficients matrix $B(n, m)$, the matrix of the K_i's (i indicates iteration) $F(n, n)$ and the number of used processors p.

The approximate algorithmic complexities of the various implemented approaches are summarized in table B.1.

B.5 Conclusion

The obtained parallel algorithms show better performances compared to the sequential one. If a cluster of computers (or a grid computer system) is used, the computations will be divided by the number of processors constituting the cluster. But increasing this number may increase the quantity of exchanged messages (but not their size).

It will be interesting to find out the right number of processors to meet optimal performances. Also, with regard to the distribution of data per blocks, the case of the identical blocs or null blocs must be taken into account to improve the performances of the third approach.

Appendix Bibliography

[1] C.T. Chen, Linear System Theory and Design, Holt Rinehart and Winston, New York, 1984.

[2] M. Yaici, H. Khaled, Z. Khaled, and A. Bentahar, A parallelization of the row-searching al-

gorithm, 9th International Conference on Mathematical Problems in Engineering, Aerospace and Sciences, AIP Conf. Proc. Vol:1493, pp 1108-1115, 2012

[3] Message Passing Interface Forum, "MPI: a Message-Passing Interface Standard", 1994. Available from http://www.mpi-forum.org/docs/docs.html

[4] Message Passing Interface Forum, "MPI-2: Extensions to the Message-Passing Interface", 1997. Available from http://www.mpi-forum.org/docs/docs.html

[5] W. Gropp, E. Lusk and A. Skjellum, Using MPI, second ed. The MIT Press, Cambridge, 1999.

Appendix C

A method to compute the inverse of a matrix polynomial

The inverse of a matrix polynomial is crucial in control theory where multi-variable systems are described by matrix fractions. From the block partial fraction expansion of a system described in matrix fraction description, the inverse of a matrix polynomial is obtained.

The following results are published in [1].

C.1 Introduction

Computing the inverse of a matrix polynomial is very important in all the fields dealing with matrix polynomials and has been treated by many researchers.

In [2] an algorithm, which requires the matrix polynomial to be column-proper, is proposed. Zhang, in [3], addressed the case of column reduced matrix polynomials to obtain an irreducible inverse. Schuster and Hippe, in [4], computed matrix polynomial inverses by interpolation, but the efficiency of the proposed algorithm is very dependent on the interpolating points which are to be chosen. An appropriate Sylvester resultant matrix has been used by Stefanidis et al., in [5], to generate a real matrix to compute its inverse. In [6] an algorithm to obtain a minimal state-space representation is given then Leverrier's algorithm is used to determine the transfer function representation of the inverse.

Many authors have considered using block roots for solving some linear algebra problems or control problems such as: block partial fraction expansion of a matrix fraction description (MFD) with single and repeated poles [7, 8], cascade decomposition and realization of multi-variable systems via block-pole and block-zero placement [9], state feedback decomposition of multivariable systems via block pole placement [10] etc.

C.2 Block partial fraction expansion

The block partial fraction expansion is obtained for systems described in left matrix fraction descitpion (MFD) and in right MFD.

C.2.1 Left MFD

We consider the transfer function $H(s)$ of an m inputs and p outputs r^{th} degree system described by the following left matrix fraction description as :

$$H(s) = D^{-1}(s)N(s) = [s^r + D_{r-1}s^{r-1} + \cdots + D_1 s + D_0]^{-1}[N_{r-1}s^{r-1} + \cdots + N_1 s + N_0] \quad \text{(C.2.1)}$$

and $\{L_1, L_2, \cdots, L_r\}$ a complete set of left solvents of $D(s)$.

It can be verified that the state space description of this system in block observer form is:

$$\begin{cases} \dot{x}_o = A_o x_o + B_o u \\ y = C_o x_o \end{cases} \quad \text{(C.2.2)}$$

where

$$A_o = \begin{pmatrix} 0 & 0 & \cdots & 0 & -D_0 \\ I & 0 & \cdots & 0 & -D_1 \\ 0 & I & \cdots & 0 & -D_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I & -D_{r-1} \end{pmatrix}, B_o = \begin{pmatrix} N_0 \\ N_1 \\ \vdots \\ N_1 \\ N_0 \end{pmatrix}, C_o = \begin{pmatrix} 0 & \cdots & 0 & I \end{pmatrix} \quad \text{(C.2.3)}$$

satisfies

$$H(s) = C_o(sI - A_o)^{-1}B_o \quad \text{(C.2.4)}$$

Using the similarity transformation $x_d = V_L x_o$, where V_L the left Vandermonde matrix assumed non-singular, we obtain:

$$\begin{cases} \dot{x}_d = A_d x_d + B_d u \\ y = C_d x_d \end{cases} \quad \text{(C.2.5)}$$

where $\begin{cases} A_d = V_L A_o V_L^{-1} \\ B_d = V_L B_o \\ C_d = C_o V_L^{-1} \end{cases}$

with $A_d = \begin{pmatrix} L_1 & & (0) \\ & \ddots & \\ (0) & & L_r \end{pmatrix}, B_d = \begin{pmatrix} B_{d1} \\ \vdots \\ B_{dr} \end{pmatrix} = \begin{pmatrix} N_0 + L_1 N_1 + \cdots + L_1^{r-1} N_{r-1} \\ \vdots \\ N_0 + L_r N_1 + \cdots + L_r^{r-1} N_{r-1} \end{pmatrix}$

and $C_d = \begin{pmatrix} C_{d1} & \cdots & C_{dr} \end{pmatrix} = \begin{pmatrix} 0 & \cdots & 0 & I \end{pmatrix} V_L^{-1}$

C_d is the last block row of V_L^{-1}.

Then $H(s)$ will be:

$$H(s) = C_d(sI - A_d)^{-1}B_d = \sum_{i=1}^{r} C_{di}(sI - L_i)^{-1}B_{di} \tag{C.2.6}$$

Hence a block partial fraction expansion of $H(s)$ about a complete set of left solvents is obtained.

C.2.2 Right MFD

We consider the transfer function of the same system described in Right MFD as follows:

$$H(s) = N(s)D^{-1}(s) = [N_{r-1}s^{r-1} + \cdots + N_1 s + N_0][s^r + D_{r-1}s^{r-1} + \cdots + D_1 s + D_0]^{-1} \tag{C.2.7}$$

with $\{R_1, R_2, \cdots, R_r\}$ a complete set of right solvents of $D(s)$.

The corresponding state space description in block controller form as follows:

$$\begin{cases} \dot{x}_c = A_c x_c + B_c u \\ y = C_c x_c \end{cases} \tag{C.2.8}$$

with $A_c = \begin{pmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & I \\ -D_0 & -D_1 & -D_2 & \cdots & -D_r \end{pmatrix}, B_d = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ I \end{pmatrix}$

and $N_d = \begin{pmatrix} N_0 & N_1 & \cdots & N_{r-1} \end{pmatrix}$

Satisfies

$$H(s) = C_c(sI - A_c)^{-1}B_c \tag{C.2.9}$$

Using the similarity transformation $x_c = V_R x_d$, with V_R is the right Vandermonde matrix assumed non-singular, we obtain

$$\begin{cases} \dot{x}_d = A_d x_d + B_d u \\ y = C_d x_d \end{cases} \tag{C.2.10}$$

where $\begin{cases} A_d = V_R^{-1} A_c V_R \\ B_d = V_R^{-1} B_c \\ C_d = C_c V_R \end{cases}$ with $A_d = \begin{pmatrix} R_1 & & (0) \\ & \ddots & \\ (0) & & R_r \end{pmatrix}$,

$C_d = \begin{pmatrix} C_{d1} & \cdots & C_{dr} \end{pmatrix} = \begin{pmatrix} N_0 + N_1 R_1 + \cdots + N_{r-1} R_1^{r-1} \\ \vdots \\ N_0 + N_1 R_r + \cdots + N_{r-1} R_r^{r-1} \end{pmatrix}$

and $B_d = \begin{pmatrix} B_{d1} \\ \vdots \\ B_{dr} \end{pmatrix} = V_R^{-1} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ I \end{pmatrix}$ which is the last block column of V_R^{-1}.

It follows that :

$$H(s) = C_d(sI - A_d)^{-1}B_d = \sum_{i=1}^{r} C_{di}(sI - R_i)^{-1}B_{di} \tag{C.2.11}$$

leading to a block partial fraction expansion of $H(s)$ about a complete set of right solvents.

C.2.3 Expansion of the inverse of $D(s)$

From the block partial fraction expansion of the left MFD (equation C.2.6), a particular case of interest is obtained if $N_{r-1} = \cdots = N_1 = 0$ and $N_0 = I$, in which case we obtain the decomposition of $D^{-1}(s)$.

Since $B_d = \begin{pmatrix} I \\ \vdots \\ I \end{pmatrix}$ and C_d is the last block row of V_L^{-1}, hence equation C.2.6 becomes:

$$D^{-1}(s) = \sum_{i=1}^{r} C_{di}(sI - L_i)^{-1} \tag{C.2.12}$$

Like previously, from the block partial fraction expansion of the right MFD, given by equation C.2.11, and if $N_{r-1} = \cdots = N_1 = 0$ and $N_0 = I$, the decomposition of $D^{-1}(s)$ follows.

Since $C_d = \begin{pmatrix} I & \cdots & I \end{pmatrix}$ and B_d is the last block column of V_R^{-1}, it follows that equation C.2.11 becomes:

$$D^{-1}(s) = \sum_{i=1}^{r} (sI - R_i)^{-1}B_{di} \tag{C.2.13}$$

The following remark states the conditions to obtain the expansion of the inverse of a matrix polynomial.

Remark C.2.1. The inverse of a matrix polynomial $D(s)$ is obtained using equations C.2.12 or C.2.13 iff $D(s)$ presents a complete set of right or left solvents. The reason is that the condition for the previous development to be possible is the fact that the right (left) block Vandermonde matrix is non-singular, and this is verified only if a complete set of solvents of $D(s)$ exist.

C.3 Illustrative example

Let $D(s) = s^2 + \begin{pmatrix} -1 & 1 \\ 0 & -1 \end{pmatrix} s + \begin{pmatrix} 0 & -1 \\ 0 & -2 \end{pmatrix}$

with the following left solvents: $L_1 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, L_2 = \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}$

Hence the left vandermonde matrix: $V_L = \begin{pmatrix} I_2 & L_1 \\ I_2 & L_2 \end{pmatrix}$

and its inverse is: $V_L^{-1} = \begin{pmatrix} 0 & 1/3 & 1 & -1/3 \\ 0 & 2/3 & 0 & 1/3 \\ 1 & -1/3 & -1 & 1/3 \\ 0 & -1/3 & 0 & 1/3 \end{pmatrix}$

It follows that: $D^{-1}(s) = C_{d1}(sI_2 - L_1)^{-1} + Cd_2(sI_2 - L_2)^{-1}$

or $D^{-1}(s) = \begin{pmatrix} 1 & -1/3 \\ 0 & -1/3 \end{pmatrix} \begin{pmatrix} s-1 & 0 \\ 0 & s+1 \end{pmatrix}^{-1} + \begin{pmatrix} -1 & 1/3 \\ 0 & 1/3 \end{pmatrix} \begin{pmatrix} s & -1 \\ 0 & s-2 \end{pmatrix}^{-1}$

then

$$D^{-1}(s) = \begin{pmatrix} 1/s(s-1) & -1/s(s+1)(s-2) \\ 0 & 1/(s+1)(s-2) \end{pmatrix}$$

C.4 Conclusion

The block partial fraction expansion of a matrix transfer function expressed as a matrix fraction description is obtained. From this expansion, the inverse of a matrix polynomial is determined and it involves the knowledge of a complete set of solvents and the computation of the inverse of a block Vandermonde matrix.

Appendix Bibliography

[1] M. Yaici and K. Hariche, "On Solvents of matrix polynomials", Int. J. Mod. Optim., Vol. 4, No.4, 273-277, 2014.

[2] Y. Inouye, "An algortihm for inverting polynomial matrices", Int. J. Control, Vol. 30, pp. 989-999, 1979.

[3] S. Zhang, "Inversion of polynomial matrices", Int. J. Control, Vol. 46, pp. 33-37, 1987.

[4] A. Schuster and P. Hippe, "Inversion of polynomial matrices by interpolation", IEEE Trans. Autom. Control, Vol. AC-37, pp. 363-365, 1992.

[5] P. Stefanidis, A.P. Paplinski and M.J. Gibbard, Numerical Operations with Polynomial Matrices, Lecture Notes in Control and Information Sciences, Vol. 171, Springer, Berlin, 1992.

[6] J.C. Basilio, "Inversion of polynomial matrices via state-space", Lin. Algebra Appl., vol. 357, pp. 259-271, 2002.

[7] L.S. Shieh, F.R. Chang and B.C. Mcinnis, "The Block Partial Fraction Expansion of a Matrix Fraction Description with Repeated Block Poles", IEEE Trans. Auto. Cont., Vol. AC-31, No. 3, pp. 236-239, 1986.

[8] J. Leyva-Ramos, "A New Method for Block Partial Fraction Expansion of Matrix Fraction Descriptions", IEEE Trans. Auto. Cont., Vol. AC-36, No. 12, pp. 1482-1485, 1991.

[9] L.S. Shieh, M.M. Mehio and R.E. Yates, "Cascade Decomposition and Realization of Multivariable Systems Via Block-Pole and Block-Zero Placement", IEEE Trans. Auto. Cont., Vol. AC-30, No.11, pp. 1109-1112, 1985.

[10] L.S. Shieh, Y.T. Tsay and R.E. Yates, "State-Feedback Decomposition of Multivariable Systems Via Block-Pole Placement", IEEE Trans. Auto. Cont., Vol. AC-28, No.8, pp. 850-852, 1983.

Appendix D

Conversion methods

In chapter 3, a conversion method to transform a system in State Space Description (SSD) into a system in Matrix fraction description (MFD), right or left, is given and used in the design process. The condition was that the system must be block controllable (or block observable), which means that the number of states n must be a multiple of the number of inputs m (or the number of outputs p), i.e. $n = \mu * m$ (or $n = \nu * p$) where μ is the controllability index (or ν the observability index).

If the dimensions of the system do not verify the previous condition then, according to Shieh et al. [1], the system can be enlarged by adding a set of non-dominant stable eigenvalues on the diagonal entries of the state matrix so that the condition is satisfied.

Conversion methods to transform a Transfer function (TF) description into an SSD description and from an MFD to a Transfer function description exist (see [2], [3], etc.)

In the following another more general method of conversion, proposed by Stefanidis et al. in [2], is presented.

D.1 Introduction

Several methods are available for obtaining MFDs [4, 5, 6, 7, 8]. In general it is not difficult to construct a rational function matrix as a product of a polynomial matrix and the inverse of another polynomial matrix. The difficulty arises when one is interested in a co-prime (least order) MFD. In engineering applications, the underlying system can be assumed finite and it is often important to obtain a least-order model, particularly if the model is to be used to design a compensator.

An approach used by Rosenbrock [3] requires elementary row-operations on the matrix

$[G(s)-I]$ which involves multiplication and division by polynomials. The method is numerically unstable and is difficult to implement on a computer. In [4, 5, 6, 7, 9] the method of determining a proper (causal) rational function matrix searches for a minimal polynomial basis for the right or left null space of a polynomial matrix. In [2, 5, 6] numerically robust computational methods have been developed, and these methods are attractive for computer software development.

D.2 Preliminaries

We start by defining the following needed terms.

D.2.1 Column and Row structure

A $n\times n$ polynomial matrix $L(\lambda)$ of order m can be decomposed in two $n\times n$ polynomial matrices

$$L(\lambda) = P_c(\lambda) + E_c(\lambda) \tag{D.2.1}$$

such that $\delta_{cj}(|P_c|) > \delta_{cj}(E_c)$ where δ_{cj} stands for degree of the j^{th} column.

P_c is called the column structure of $L(\lambda)$ and E_c is called the column residue.

Similarly, the $n \times n$ polynomial matrix $L(\lambda)$ can be decomposed in 2 $n \times n$ polynomial matrices:

$$L(\lambda) = P_r(\lambda) + E_r(\lambda) \tag{D.2.2}$$

such that $\delta_{rj}(|P_r|) > \delta_{rj}(E_r)$ where δ_{rj} stands for the degree of the j^{th} row.

P_r is called the row structure of $L(\lambda)$ and E_r is called the row residue.

In general we write:

$$P_c(\lambda) = \Gamma_c.diag\{\lambda^{\nu_1}\lambda^{\nu_2}, \cdots, \lambda^{\nu_n}\} \text{ and } P_r(\lambda) = \Gamma_r.diag\{\lambda^{\mu_1}\lambda^{\mu_2}\cdots\lambda^{\mu_n}\} \tag{D.2.3}$$

Where Γ_c and Γ_r are rational $n\times n$ matrices of the highest column and row degree coefficients, and $diag\{\lambda\}$ is a diagonal matrix of highest degrees on λ of $L(\lambda)$.

Example:

Let $L(\lambda) = \begin{pmatrix} 5\lambda+1 & \lambda^2+3\lambda+2 & 4\lambda+5 \\ 3\lambda+1 & 2\lambda+1 & \lambda^3+\lambda^2+2 \\ \lambda+7 & 3 & 3 \end{pmatrix}$

The column decomposition of $L(\lambda)$ is:

$$L(\lambda) = \begin{pmatrix} 5 & 1 & 0 \\ 3 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda^2 & 0 \\ 0 & 0 & \lambda^3 \end{pmatrix} + \begin{pmatrix} 1 & 3\lambda+2 & 4\lambda+5 \\ 1 & 2\lambda+1 & \lambda^2+2 \\ 7 & 3 & 3 \end{pmatrix}$$

And the row decomposition of $L(\lambda)$ is:

$$L(\lambda) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \lambda^2 & 0 & 0 \\ 0 & \lambda^3 & 0 \\ 0 & 0 & \lambda \end{pmatrix} + \begin{pmatrix} 5\lambda+1 & 3\lambda+2 & 4\lambda+5 \\ 3\lambda+1 & 2\lambda+1 & \lambda^2+2 \\ 7 & 3 & 3 \end{pmatrix}$$

Here $\Gamma_c = \begin{pmatrix} 5 & 1 & 0 \\ 3 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$ and $\Gamma_r = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

Remark D.2.1. If Γ_c (Γ_r) is non-singular then $L(\lambda)$ is column (row) proper.

D.2.2 Skeleton block matrix

Let an unknown $n \times n$ non-singular $L(\lambda)$ of order m and degree r for which its column structure P_c is specified but not its residue E_c. The information contained in P_c is sufficient to construct the row block-matrix description Ls called the skeleton row block-matrix of L.

Ls is defined as having the following properties:

i) The degree of Ls is the degree of column structure matrix: $\delta(Ls) = \delta(P_c) = r$. Thus the skeleton row block-matrix can be written as: $Ls = \begin{pmatrix} Ls_0 & \cdots & Ls_r \end{pmatrix}$.

ii) The dimension of Ls is: $n \times n(r+1)$.

iii) The columns of Ls are composed of columns associated to the column structure matrix P_c and columns associated to the column residue matrix E_c:

   a) Columns associated to P_c: These columns are called the active columns of Ls (which coefficients are known) the others are called inactive (null columns).

   b) Columns in E_c: these are the columns indicated by "?". The number of the inactive columns is $nr - m$ and the number of the columns in E_c is equal to the order m but if $L(\lambda)$ is regular then the number of inactive columns will be m.

Example1:

Let $P_c = \begin{pmatrix} 8 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} diag\{\lambda^2\lambda^3\lambda\}$

Then $L(\lambda)$ can be written as:

$$L(\lambda) = \begin{pmatrix} 8\lambda^2 & 0*\lambda^2+? & 0*\lambda^2+? \\ 0*\lambda^3+? & 2*\lambda^3+? & 0*\lambda^3+? \\ 0*\lambda+? & 0*\lambda+? & \lambda+? \end{pmatrix}$$

And its Skeleton row block-matrix:

$$Ls = \left(\begin{array}{ccc:ccc:ccc} 0 & 0 & 0 & 8 & 0 & 0 & ? & ? & ? \\ 0 & 2 & 0 & ? & ? & ? & ? & ? & ? \\ 0 & 0 & 0 & 0 & 0 & 1 & ? & ? & ? \end{array}\right)$$

Example2:

Let $P_r = \begin{pmatrix} 8 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} diag\{\lambda^2\lambda^3\lambda\}$

Then $L(\lambda)$ can be written as: $L(\lambda) = \begin{pmatrix} 8\lambda^2+? & 0*\lambda^2+? & 0*\lambda^2+? \\ 0*\lambda^3+? & 2\lambda^3+? & 0*\lambda^3+? \\ 0*\lambda+? & 0*\lambda+? & \lambda+? \end{pmatrix}$

And the transpose of its Skeleton column block-matrix:

$$Ls^T = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \\ 8 & 0 & 0 \\ ? & ? & ? \\ 0 & 0 & 0 \\ ? & ? & ? \\ ? & ? & ? \\ 0 & 0 & 1 \\ ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{pmatrix}$$

D.2.3 Resultant of a polynomial matrix

Let a $n_r \times n_c$ polynomial matrix $L(\lambda)$ of degree r such that $L = \begin{pmatrix} L_0 & \cdots & L_r \end{pmatrix}$.

Definition D.2.1. The resultant of order h of L is denoted $< L >_h$ and is defined by:

$$< L >_h \doteq \sum_{i=0}^{r} R_i \otimes L_i \tag{D.2.4}$$

Where $\otimes$ denotes the Kronecker product and $R_i \doteq \begin{pmatrix} 0_{((h+1)\times i)} & I_{h+1} & 0_{(h+1)\times(m-i)} \end{pmatrix}$

Example:

Let $L = \begin{pmatrix} L_0 & L_1 & L_2 \end{pmatrix}$

Then $< L >_2 = \begin{pmatrix} L_0 & L_1 & L_2 & 0 & 0 \\ 0 & L_0 & L_1 & L_2 & 0 \\ 0 & 0 & L_0 & L_1 & L_2 \end{pmatrix}$

where 0 stands for zero matrices of right dimensions.

D.3 Conversion algorithms

We consider a MIMO system described in state space equation as follows:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Eu \end{cases} \tag{D.3.1}$$

Where A is an $n \times n$ state matrix, B is $n \times m$ input matrix, C is an $p \times n$ ouput matrix and E is an $p \times m$ transmission matrix.

D.3.1 Conversion to a Right MFD

Using the previous notions of skeleton block matrix, the following algorithm converts a system described in SSD to a right MFD: $N(s)D^{-1}(s)$.

We suppose the system controllable.

The following steps summarize the algorithm:

1. Determine the controllability indices of the state space system, let r be its highest index.
2. Construct the row structure matrix of the denominator D of degree r.
3. Construct the (unknown) skeleton block transpose column block-matrix Ds^T of D.
4. Solve for residue rows of the skeleton block matrix by solving the equation:

$$\mathcal{AB} * Ds^T = 0 \tag{D.3.2}$$

where:

$$\mathcal{AB} = \begin{pmatrix} A^rB & \cdots & AB & B \end{pmatrix} \tag{D.3.3}$$

5. Determine the block numerator:

$$N^T = \begin{pmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ CA^{r-1}B & \cdots & CAB & CB \end{pmatrix} * \begin{pmatrix} D_0 \\ D_1 \\ \vdots \\ D_r \end{pmatrix} \tag{D.3.4}$$

where 0 are null matrices of dimension $p \times m$

6. If the transmission matrix E of the SSD is not a zero matrix then the numerator is modified to:

$$\begin{pmatrix} 0_{p\times m} \\ N^T \end{pmatrix} + < E >^T_{\delta(D)} .D^T \tag{D.3.5}$$

Where $< E >$ is the block resultant matrix of E of order the degree $\delta(D)$.

D.3.2 Conversion to a Left MFD

A similar algorithm can be used to transform the system in SSD to a left MFD: $D^{-1}(s)N(s)$ and we suppose the system observable.

The algorithm is given by the following steps:

1. Determine the observability indices of the state space system, let r be the highest index.

2. Determine the column structure of the denominator D of degree r.

3. Construct the (unknown) skeleton row block-matrix Ds^T of D.

4. Solve for residue rows of the skeleton block matrix by solving the equation:

$$D * \mathcal{CA} = 0 \tag{D.3.6}$$

Where

$$\mathcal{CA} = \begin{pmatrix} CA^r \\ \vdots \\ CA \\ C \end{pmatrix} \tag{D.3.7}$$

5. Determine the block numerator:

$$N^T = \begin{pmatrix} D_0 & \cdots & D_r \end{pmatrix} \begin{pmatrix} CB & CAB & \cdots & CA^{r-1}B \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & CB & CAB \\ 0 & \cdots & 0 & CB \end{pmatrix} \tag{D.3.8}$$

6. If the transmission matrix E of the SSD is not a zero matrix then the numerator is modified as previously.

D.3.3 Illustrative example

Let the following be an SSD of a controllable and observable system with n=4 states, m=2 inputs and p=3 outputs (from [2] page 73).

$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Eu \end{cases}$ Where $A = \begin{pmatrix} -1 & 0 & 3 & -1 \\ 4 & -2 & 1 & 2 \\ 3 & 0 & -3 & 4 \\ -4 & -2 & -3 & -4 \end{pmatrix}$, $B = \begin{pmatrix} 2 & 1 \\ 0 & 2 \\ 1 & -1 \\ -1 & 0 \end{pmatrix}$,

$C = \begin{pmatrix} -1 & 2 & 1 & -2 \\ 0 & -1 & -2 & 0 \\ 1 & 1 & 0 & -2 \end{pmatrix}$ and $E = [0_{3\times 2}]$

Following the steps of the first algorithm to get a RMFD $N(s)D^{-1}(s)$ we have:

1. The controllability indices of the SSD system is: $\mu_1 = 2$ and $\mu_2 = 2$ (computed using the row-searching algorithm for example).

2. The row structure matrix of $D(s)$ is: $P_r(s) = diag\{s^{\mu_1}, s^{\mu_2}\} * I_2$ or equivalently: $P_r(s) = \begin{pmatrix} s^2 & 0 \\ 0 & s^2 \end{pmatrix}$

3. The skeleton column block-matrix of $D(s)$ is: $Ds = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ e_1 & e_1 \\ e_2 & e_2 \\ e_3 & e_3 \\ e_4 & e_4 \end{pmatrix}$

   There are m=2 active structure rows and n=4 residue rows noted here e_i to distinguish them.

4. Solve the following equation for residue rows: $\begin{pmatrix} A^2B & AB & B \end{pmatrix} Ds = 0$

   or : $\begin{pmatrix} 2 & -4 & 2 & 1 \\ 7 & -1 & 0 & 2 \\ -1 & 6 & 1 & -1 \\ -7 & -5 & -1 & 0 \end{pmatrix} \begin{pmatrix} e_4 & e_4 \\ e_3 & e_3 \\ e_2 & e_2 \\ e_1 & e_1 \end{pmatrix} = \begin{pmatrix} 2 & 27 \\ -21 & -18 \\ -19 & -50 \\ 9 & 20 \end{pmatrix}$

   To obtain D per blocs: $D^T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -3.81 & -7.13 \\ 6.64 & 13.81 \\ 2.51 & 0.83 \\ 27.18 & 40.86 \end{pmatrix}$

5. Then the numerator is obtained as follows:

$$N^T = \begin{pmatrix} CB & 0 \\ CAB & CB \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -3.81 & -7.13 \\ 6.64 & 13.81 \end{pmatrix} = \begin{pmatrix} -1 & 2 \\ -2 & 0 \\ 4 & 3 \\ 34.46 & 38.50 \\ 2.63 & 3.26 \\ 27.65 & 17.92 \end{pmatrix}$$

6. $E = 0$ then no need to modify the obtained numerator.

So the RMFD form of the system is:

$$N(s)D^{-1}(s) = \begin{pmatrix} -s + 34.46 & 2s + 38.5 \\ -2s + 2.63 & 3.26 \\ 4s + 27.65 & 3s + 17.92 \end{pmatrix} \begin{pmatrix} s^2 - 3.81s + 2.51 & -7.13s + 0.83 \\ 6.64s + 27.18 & s^2 + 13.81s + 40.86 \end{pmatrix}^{-1}$$

D.4 Conclusion

The given conversion method is efficient and more general but more complex than the method adopted in the design process. The method is well conditioned due to the use in step 2 of a numerically robust routine to compute the inverse of a real matrix.

Appendix Bibliography

[1] L. S. Shieh, Y. T. Tsay and R. F. Yates, "State feedback decomposition of multivaraiblesystems via block pole placament", IEEE Trans. Autom. Contr. Vol. AC-28, pp. 850-852, 1983.

[2] P. Stefanidis, A. P. Paplinski and M. J. Gibbard, Numerical Operations with Polynomial Matrices: Application to Multivariable Dynamic Compensator Design, Spring-verlag, Berlin, 1992.

[3] H. H. Rosenbrock, State-space and multivariable theory, Wiley-interscience, NY, 1970.

[4] G. D. Forney Jr., "Minimal basis of rational vector spaces with applications to multivariable linear systems", SIAM J. Cont., Vol. 13, No. 3, pp. 493-520, 1975.

[5] L. V. Foster, "A practical solution to the minimal design problem", IEEE Trans. Auto. Cont., Vol. AC-24, pp. 449-454, 1979.

[6] S. Kung, T. Kailath and M. Morf, "Fast and stable algorithms for minimal design problems", in Proc. 4th IFAC Int. Symp. Multivariable technol. Syst., pp. 97-104, 1977.

[7] M. K. Sain, " A free modular algorithm for minimal design of linear multivariable systems", in Proc of 6th IFAC Congr. PartIB, Section9.1, pp. 1-7, Boston, 1975.

[8] W. A. Wolovich and P. L. Falb, " On the structure of multivariable systems", SIAM. J. Contr., Vol. 7, pp. 437-451, 1969.

[9] S. H. Wang and E.J. Davison, "A minimization algorithm for the design of multivariable systems", IEEE Trans. Automat. Contr., Vol. AC-18, pp. 31-38, 1973.