ADAPTIVE PREDICTIVE CONTROL:
ANALYSIS AND EXPERT IMPLEMENTATION

By
Abdel-Latif Elshafei

B. Sc. Electrical Engineering, Cairo University, Egypt, 1983
M. Sc. Electrical Engineering, Cairo University, Egypt, 1986

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
Doctor of Philosophy

in
THE FACULTY OF GRADUATE STUDIES
ELECTRICAL ENGINEERING

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
August 1991

© Abdel-Latif Elshafei, 1991
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Electrical Engineering
The University of British Columbia
2075 Wesbrook Place
Vancouver, Canada
V6T 1W5

Date:    Oct. 10, 1991
Acknowledgement

I praise and thank God for His help and guidance to make this research happen.

I would like to express my deep gratitude to my supervisor Prof. Guy Dumont for providing scientific guidance and financial support throughout my Ph.D. research years. Through Prof. Dumont's course on self-tuning regulators, the ideas of predictive control and expert control have been introduced in such inspiring way that I have been motivated to carry out my research on these topics.

I also thank all my instructors both at the University of British Columbia and Cairo University as they have contributed effectively in widening my scope of thinking. My thanks should also be directed to the PAPRICAN not only for providing me with financial support during the last two years but also for providing an excellent research environment through the Pulp and Paper Centre.

All the ideas and details of this thesis have been discussed, revised, or modified with my friend Ashraf Elnaggar. To Ashraf, I direct my deep and sincere gratitude.

Finally, I would like to thank and congratulate my parents, brother and wife. This work is nothing but a result of their love, support, and care. To my little son, Ibraheem, I devote this thesis.
Abstract

A generalized predictive controller has been derived based on a general state-space model. The case of a one-step control horizon has been analyzed and its equivalence to a perturbation problem has been emphasized. In the case of a small perturbation, the closed-loop poles have been calculated with high accuracy. For the case of a general perturbation, an upper bound on the permissible perturbation norm has been derived. A functional analysis approach has also been adopted to assess the closed-loop stability in the case of nonlinear systems. Both the plant-model match and plant-model mismatch cases have been analyzed. The proposed controller has proven to be so robust that an adaptive implementation based on Laguerre-filter modelling has been motivated. Both SISO and MIMO schemes have been analyzed. Using a sufficient number of Laguerre filters for modelling, the adaptive controller has been proven to be globally convergent. For low-order models, the robustness of the adaptive controller can be insured by increasing the prediction horizon. The convergence and robustness results have been extended to other predictive controllers. A comparative study has shown that the proposed controller would be superior to the other predictive controllers if the open-loop system is stable, well-damped, and of unknown order or time delay. To achieve a reliable control without deep user involvement, the adaptive version of the proposed controller has been implemented using the expert shell, G2. The resulting expert system has been used to orchestrate the operation of the controller, provide an interactive user interface, adjust the Laguerre-filter model using AI search algorithms, and evaluate the performance of the controller on-line. Based on the performance evaluation, the tuning parameters of the controller can be adjusted on-line using fuzzy-logic rules.
Table of Contents

Acknowledgement ii

Abstract iii

List of Figures vii

1 Introduction 1
   1.1 Motivation ......................................................... 1
   1.2 Literature Survey ................................................. 3
      1.2.1 Prediction ..................................................... 3
      1.2.2 Non-parameteric predictive control .......................... 4
      1.2.3 CARMA/CARIMA model based predictive control ................. 5
      1.2.4 Model reference predictive control .......................... 7
      1.2.5 State-space based predictive control ........................ 7
      1.2.6 Extensions .................................................... 9
   1.3 Review of basic predictive control algorithms .................... 9
      1.3.1 Model-sequence predictive control .......................... 9
      1.3.2 Generalized Predictive Control ............................. 12
   1.4 Thesis contribution ............................................... 14
   1.5 Thesis Outline ................................................... 16

2 Generalized Predictive Control 18
   2.1 Introduction ..................................................... 18
2.2 GPC based on a state space modelling ........................................ 19
  2.2.1 General case ................................................................. 19
  2.2.2 Case of a single-step control-horizon .................................. 28
2.3 Case of Plant-model match ...................................................... 36
  2.3.1 Analysis of open-loop stable systems ................................... 36
  2.3.2 Analysis of open-loop unstable systems ................................. 39
2.4 Case of plant-model mismatch .................................................. 50
  2.4.1 General plant-representation ............................................. 50
  2.4.2 Linear time-invariant plant-representation ............................. 56
2.5 Conclusions ............................................................................. 58

3 Adaptive Generalized Predictive Control ........................................ 59
  3.1 Introduction ............................................................................ 59
  3.2 Modelling and identification .................................................. 60
  3.3 Analysis of single-input, single-output systems ........................... 63
    3.3.1 Stability and convergence analyses in case of plant-model match 64
    3.3.2 Robustness analysis ......................................................... 67
    3.3.3 Further analyses .............................................................. 72
  3.4 Extensions to multivariable systems ......................................... 83
    3.4.1 Single-input, multi-output systems .................................... 84
    3.4.2 Multi-input, multi-output systems ..................................... 89
  3.5 Comparative study ................................................................... 95
  3.6 Conclusion ............................................................................... 99

4 Expert Adaptive GPC Based on Laguerre-Filter Modelling ................ 104
  4.1 Introduction ............................................................................ 104
  4.2 The expert shell, G2 ............................................................... 108
4.3 Implementation .......................................................... 111
  4.3.1 Overview ............................................................ 111
  4.3.2 User interface ....................................................... 113
  4.3.3 Model adjustment .................................................... 115
  4.3.4 Tuning based on fuzzy logic ........................................ 119
  4.3.5 Illustrative examples ............................................... 126
4.4 Conclusion ............................................................ 129

5 Conclusions .......................................................... 131
  5.1 The main results of the thesis ...................................... 131
  5.2 Suggestions for further research ................................... 134

Bibliography .......................................................... 136

A Building an Expert System ........................................ 146
  A.1 Definitions and characteristics .................................... 146
  A.2 Major stages of knowledge acquisition ............................ 147
  A.3 Techniques of knowledge representation ........................... 147
    A.3.1 Semantic networks ................................................. 147
    A.3.2 Frame systems ................................................ 148
    A.3.3 Object-oriented programming .................................. 148
    A.3.4 Rule-based representation ...................................... 149

B Fuzzy-Logic Definitions and Terminology ........................ 150
List of Figures

2.1 GPC $n_2 = 10, \beta = 1$ and $n_u = 1$ ........................................... 26
2.2 GPC $n_2 = 10, \beta = 1$ and $n_u = 2$ ........................................... 26
2.3 GPC $n_2 = 10, \beta = 1$ and $n_u = 5$ ........................................... 27
2.4 GPC $n_2 = 10, \beta = 10$ and $n_u = 5$ ........................................... 27
2.5 GPC $n_2 = 15, \beta = 1$ and $n_u = 2$ ........................................... 28
2.6 GPC of a first-order system .......................................................... 33
2.7 GPC of a nonminimum-phase system with one unstable pole ............ 48
2.8 Using the perturbation analysis to choose $n_2$ ............................... 50
2.9 The control system in case of exact representation ......................... 51
2.10 The approximate control system .................................................... 52

3.11 a- The open-loop response. b- The control signal. c- The closed-loop
response. ................................................................................. 68
3.12 The estimates of the Laguerre-filters' gains ..................................... 69
3.13 A comparison between the frequency responses of the true system (dotted
line) and the identified model (solid line). .................................... 70
3.14 Robustness of the Laguerre-filters based GPC ................................. 73
3.15 Control of a nonlinear system. a- The open-loop response. b- The set-
point. c- The manipulated variable. c- The closed-loop response. .......... 79
3.16 Control of an open-loop unstable system ....................................... 82
3.17 Performance of the LAG-GPC in a noisy environment .................... 84
3.18 Control of a single-input, multi-output system ............................... 88
3.19 Control of a multi-input, multi-output system ........................................... 96
3.20 Predictive control of a high order system .................................................. 98
3.21 Predictive control of a poorly damped system .......................................... 100
3.22 Predictive control of a nonminimum-phase system ................................... 101
3.23 Comparison of the robustness of the MAC, GPC and LAG-GPC. .................. 102
4.24 Expert control of an overdamped system with time delay .......................... 128
4.25 Expert control of an underdamped system .................................................. 129
1.1 Motivation

Interest in adaptive control has been steady over the last two decades from both users and researchers of control systems. In the American Control Conference 1988, a whole session was devoted to explore how a designer can pick up a suitable adaptive control algorithm. Quoting Masten and Cohen, (1988):

"The adaptive control literature contains hundreds of technical papers on many approaches to the subject. However, engineers who are not specialists in adaptive control theory often have difficulties selecting which approach to use in a given problem."

Quoting Wittenmark and Åström, (1984):

"A non-specialist who tries to get an understanding of adaptive control is confronted with contradicting information such as:

- Adaptive control works like a beauty in this particular application.
- Adaptive control is ridiculous. Look how the algorithm behaves in this particular simulation.
- You cannot possibly use adaptive control because there is no proper theory that guarantees stability and convergence of the algorithm."
Clarke, et al., (1987-a) give some guidelines for the admissibility of an adaptive control algorithm. They state that the algorithm should be applicable to:

- Nonminimum-phase plants.
- An open-loop unstable plant or plant with badly damped poles.
- A plant with variable or unknown dead time.
- A plant with unknown order.

Consequently, it is necessary in practice to trade the performance of the control algorithm against its robustness to the above requirements. This is the real motive for introducing predictive control algorithms.

In predictive control, the controller is detuned by using a control law based on an extended prediction horizon. In such a case, the process is allowed to reach its desired state after a time as long as the control designer wishes. This policy helps to overcome some difficulties like dealing with nonminimum phase plants or plants with unknown dead times. Although simulations show satisfactory results, it is true that more theoretical studies are still needed since most of the available results are only asymptotic.

It is also noted that predictive controllers contain many knobs to be tuned. Rules of thumb, 'heuristics', and expertise play an important role in getting an adaptive predictive controller working. The design problem still needs an expert to tackle it. Artificial Intelligence (AI) techniques provide a reasonable solution for the tuning problem. This is natural as the tuning of controller parameters looks like a search process in AI. It may also be argued that AI can provide a sort of supervision to look after some important implementation issues such as: signal conditioning, estimator wind-up, ringing of controllers, etc.
Chapter 1. Introduction

Assuming a single-stage prediction horizon, one-step control horizon and zero weight on the control actions, a predictive controller based on Laguerre-filter modelling is derived in Zervos, (1988). The initial success of the Laguerre-filter based predictive controller motivates us to develop a generalized predictive controller based on the same modelling technique. By using a multi-stage prediction horizon, rather than a single-stage one, the robustness of the resulting control system is enhanced. By introducing control weight, the increments of the control action are better controlled. Finally, by allowing a flexible control horizon, rather than a one-step horizon, systems which are not necessary well-damped can be controlled better.

Since Laguerre-filter models are available in state-space form, it motivates us to derive a generalized predictive controller in state-space form. The flexibility of Laguerre-filter representations motivates us to apply it not only to SISO systems but also to SIMO and MIMO systems. Theoretical properties, including stability, robustness and convergence of the adaptive control system, are analyzed.

Finally, motivated by the demand to make the proposed scheme more accessible to practitioners, the proposed scheme is implemented using expert-system and fuzzy-logic techniques.

1.2 Literature Survey

In this section predictive control techniques are surveyed. A survey of the expert-control literature is given in Chapter 4.

1.2.1 Prediction

In De Keyser and Van Cauwenberghe (1981), a self-tuning multistep predictor is proposed to predict the future output trajectory of a stochastic process over a finite horizon.
The goal of the predictor is to help the operator decide the correct control action. The global convergence of adaptive predictors is studied in Goodwin et al. (1981). It is shown there that using a priori residuals in the regressor vector and knowing an upper bound on the system order, global convergence can be achieved. In Clarke, et al.(1983), prediction regulation against offsets such as those induced by load disturbances is dealt with using an incremental model. As a by-product the estimator becomes better conditioned numerically because it works on zero mean data. In Mosca, et al. (1989), effects of multipredictor information in adaptive control are studied especially from the standpoint of structured and unstructured uncertainties.

A review of k-step-ahead predictors for SISO stochastic systems is given in Family and Dubois (1990). A comprehensive review of prediction can be found in Goodwin and Sin (1984).

1.2.2 Non-parameteric predictive control

The current interest of the process-control industry in predictive control can be traced back to a set of papers which appeared in the late 1970s. In Richalet, et al.(1978), a new algorithm called "Model Algorithmic control, (MAC)" is proposed. The new algorithm relies on three principles. First, the plant is represented by its impulse response coefficients which are to be used for long range prediction. Second, the behavior of the closed-loop system is prescribed by reference trajectories. Third, a heuristic approach is used to compute the control variables. In Cutler and Ramaker (1980), the Dynamic Matrix Control, (DMC), algorithm evolves from representing process dynamics with step-response coefficients. Both feedback and feedforward are incorporated into the DMC algorithm to compensate for unmeasured and measured disturbances. In Martin (1981), the DMC and MAC algorithms are compared and similarity conditions to dead-beat control are pointed out.
In Mehra, et al. (1979) and Rouhani and Mehra (1982), theoretical properties of MAC are discussed with emphasis on stability and robustness analysis. Of significance is the analytical explanation of the enhanced robustness of the control schemes as the reference trajectory is slowed down. An interpretation of the MAC algorithm in terms of state feedback and pole-placement is also discussed there. In Warwick and Clarke (1986), the DMC algorithm is modified as follows: a set of control inputs is calculated and made available at each time instant, the actual input applied is a weighted summation of the inputs within the set. In Garcia, et al. (1989), a number of design techniques including DMC and MAC are compared. It is pointed out there that predictive control is not inherently more or less robust than classical feedback, but, that it can be adjusted more easily for robustness.

Predictive control based on Laguerre-filter modelling is proposed in Zervos and Dumont (1986), Zervos (1988), and Dumont and Zervos (1988). The predictive controller suggested there is based on a single stage prediction, a one step control horizon and zero weighting on the control actions. Tuning the model by making an optimal choice of the Laguerre-filter pole is derived in Dumont, et al. (1991) and Fu and Dumont (1991). Chapter 3 in this thesis is devoted to the use of Laguerre-filter based models in predictive control.

1.2.3 CARMA/CARIMA model based predictive control

The idea of predictive control is first introduced in Martin-Sanchez, (1976), where a multi-variable ARMA model is used for modelling. The control signal is generated by a control block which behaves as the inverse of the process and has as its input the desired output of the process. In Peterka (1984), the main ingredients of generalized predictive control, (GPC), are presented. The internal representation of the system is based on linear finite-memory output predictors. Both CARMA-model and incremental-model
based predictors are analyzed. A quadratic-optimum control method is used to calculate the control actions. Both regulator and servo problems are covered.

In Van Cauwenberghe and De Keyser (1985), the properties of long-range predictive control are discussed. It is shown that the control action is independent of the process time-delay if the prediction horizon beyond the time delay is kept the same. It is also shown there that, for a second order system without delay, the control law can be written as a linear time-invariant feedback law and that the relation of the controller parameters to the control weighting factor and the prediction horizon is totally lost.

GPC algorithms have acquired their present popularity due to a series of papers by Mohtadi and Clarke. In Mohtadi and Clarke (1986), GPC is suggested to incorporate the "Linear Quadratic, (LQ)" and pole-placement advantages in one scheme. In Clarke, et al. (1987-a), a novel approach to GPC is presented. It is shown there that by choosing particular values for of the tuning parameters various algorithms such as GMV (Clarke and Gawthrop, 1975), EPSAC (De Keyser, 1985), Peterka's predictive controller (1984), and EHAC (Ydstie, 1984) result in. In Clarke, et al. (1987-b), asymptotic properties of the GPC are studied and the vital role of the observer polynomial is pointed out. The robustness of the GPC approach to over- and under-parameterization and to fast sampling is demonstrated by simulation. In Mohtadi (1989), the roles of an observer polynomial and a parameter-estimator prefilter are emphasized and rules of thumb for designing them are stated. The GPC algorithm and its properties are reviewed in Clarke and Mohtadi (1989).

In McIntosh, et al. (1989), it is shown that the GPC control law can be written in an equivalent linear transfer function form. Tuning strategies of the GPC are explained in McIntosh, et al.(1991). The use of only one tuning parameter to adjust the GPC performance during the commissioning stage is suggested. There are three strategies suggested to tune the GPC based on output horizon, control weight and detuned model.
Chapter 1. Introduction

In Soeterboek, et al. (1990), an attempt is made to unify predictive control techniques. Using a general model and performance index, the designer can pick up a specific predictive controller by a proper adjustment of the tuning parameters. Tuning procedures of the unified predictive controller are discussed in Soeterboek, et al. (1991).

1.2.4 Model reference predictive control

In Irving, et al. (1986), an adaptive controller inspired by pole-placement, multiple reference models and predictive control is presented. The advantages of model reference predictive control are: there is no need to solve the Diophantine equation to achieve pole-placement equivalence, the multiple reference model is not restricted to minimum phase systems, and the predictive control law is weakly sensitive to weighting factors. Robustness and stability problems associated with model reference predictive control are reported in Wertz (1987).

In Wertz, et al. (1987) and Gorez, et al. (1987), two generalizations are brought to model reference predictive control. The first one applies the same filter to input as well as output errors so that the designer can impose the dominant poles of the closed-loop system. The second generalization introduces, in the performance index, weighting factors which can be tuned according to the controlled process. In M’Saad, et al. (1988), model reference predictive control is used in a real-time experiment to control a flexible arm. Robustness is achieved by using a regularized least-squares algorithm in combination with input-output data filtering and normalization.

1.2.5 State-space based predictive control

In Lam (1982), the relationship between the k-step-ahead predictor approach and the state-space approach, based on minimizing an N-stage fixed-horizon cost function, is explored. The use of an N-stage receding-horizon cost function is proposed. Lam’s work
Chapter 1. Introduction

is based on CARMA models. A state-space formulation based on CARIMA models and minimization of multi-stage cost functions is reported in Clarke, et al.(1985). The algorithm reported there is claimed to be robust against wrong a priori assumptions made about the plant dead-time or order. In Little and Edgar (1986), an algorithm is proposed to enable the designer to constrain the system states and control actions. Little and Edgar’s algorithm provides an alternative to the use of quadratic programming for predictive control.

In Mohtadi and Clarke (1986), the relation between the GPC algorithm and the LQ problem is pointed out. In Bitmead, et al.(1989), this relation is examined in detail, with an attempt being made to analyze the GPC algorithm theoretically from an optimal control perspective. The conclusion is that the GPC algorithm does not meet the sufficient stability conditions. In Bitmead, et al.(1989-b), an explicit connection is made between the adaptive GPC algorithm and the Linear Quadratic Gaussian control with Loop Transfer Recovery, (LQG/LTR). It is shown there that the GPC algorithm embodies the elements of LQG/LTR design. It is shown also that adaptation and robustness may be mutually supportive in adaptive GPC. A detailed study of the link between GPC and LQG/LTR can be found in Bitmead, et al.(1990).

In Kwon and Byun (1989), a receding-horizon tracking controller based on state-space modelling is presented. The control law is shown to be a fixed-gain state feedback controller with a feedforward control term that provides the preview actions. For specific classes of weighting matrices and a prediction horizon greater than the system order, asymptotic stability is guaranteed. In Albertos and Ortega(1989), alternative solutions to the GPC problem are proposed. It is shown there that the prediction equation used for the GPC can be derived from the transfer function coefficients replacing the Diophantine equation recursions by the inversion of a lower triangle matrix.
1.2.6 Extensions

In Bequette (1991), a nonlinear predictive control strategy based on a nonlinear lumped parameter model of the process is suggested. Similar to linear predictive control, an optimal sequence of future control actions is calculated so that a certain performance index is minimized. Unmeasured state variables and load disturbances are estimated using a constrained approach. Model-process mismatch is handled by using an additional output term which is equivalent to the internal model approach. In Soeterboek, et al. (1991), a nonlinear predictive controller is proposed assuming the availability of a model of the process including its nonlinearity. Control of a pH process is given as an example.

In Demircioglu and Gawthrop (1991), a continuous-time version of the discrete-time adaptive GPC is presented. An important feature of the continuous GPC algorithm is that, unlike polynomial LQ and pole-placement controllers, it is not necessary to solve a Diophantine equation to compute the control law. It is also shown there that control weighting is not necessary to control nonminimum phase systems. As in the discrete-time GPC, LQ control can be considered as a subalgorithm of the continuous-time GPC.

1.3 Review of basic predictive control algorithms

In this section the MAC and the CARIMA-model based GPC are reviewed. They will be used later on for comparison purposes.

1.3.1 Model-sequence predictive control

MAC and DMC are referred to as model-sequence predictive control. In this section the basic MAC is explained in detail. Following Clarke and Zhang (1985), the MAC
algorithm proceeds by assuming a model given by,

\[ y(t) = \sum_{j=1}^{N} g_j q^{-j} u(t) + \frac{\zeta(t)}{1 - q^{-1}} \]  \hspace{1cm} (1.1)

where,

\[ y(t): \text{ the model output at time } t. \]

\[ u(t): \text{ the control signal at time } t. \]

\[ t: \text{ the discrete time index.} \]

\[ g_j: \text{ the } j^{th} \text{ impulse response coefficient.} \]

\[ \frac{\zeta(t)}{(1-q^{-1})}: \text{ disturbance model with } \zeta(t) \text{ a white noise signal.} \]

This model implicitly assumes a stable and causal system. Using this representation, it can be shown that the predicted output \( \hat{y}(t+i+1) \) is given by,

\[ \hat{y}(t+i+1) = \hat{y}(t+i) + g^T \Delta u(t+i), i = 0, ..., n_2 - 1 \]  \hspace{1cm} (1.2)

where,

\[ \hat{y}: \text{ predicted output} \]

\[ g^T = [g_1 \ g_2 \ ... \ g_{n_2}] \]

\[ \Delta u(t) = u(t) - u(t-1) \]

To obtain a prediction at time \( t \) of the output \( \hat{y}(t+i) \), it is assumed that this prediction can be separated into two components:

1. \( \hat{y}_1(t+i) \) which is the predicted response based on the assumption that future controls equal the previous control, \( u(t-1) \).

\[ \Rightarrow \hat{y}_1(t+i+1) = \hat{y}_1(t+i) + g^T \Delta u_1(t+i) \]  \hspace{1cm} (1.3)
where,
\[ \Delta u_1(t + i) = [0, ..., 0, \Delta u(t - 1), ..., \Delta u(t - n_2 + 1 + i)]^T \]

2. \( \hat{y}_2(t + i) \) which is the extra response due to future control actions.

\[ \Rightarrow \hat{y}_2(t + i + 1) = \hat{g}^T u_2(t + i) \quad (1.4) \]

where,
\[ u_2(t + i) = [u(t + i) - u(t - 1), ..., u(t) - u(t - 1), 0, ..., 0] \]

It is required to drive the output \( y(t) \) smoothly to the set-point \( w \) at the end of the prediction horizon \( n_2 \). This smooth transition can be achieved if the plant tracks a first-order model reference given by:

\[ y_{mr}(t) = y(t) \quad (1.5) \]

\[ y_{mr}(t + i) = (1 - \alpha)w + \alpha y_{mr}(t + i - 1) \quad (1.6) \]

where,
\[ y_{mr} \]: model reference output.
\[ \alpha \]: a constant of the model reference.

The error reference vector, \( e \), can be expressed as:

\[ e = y_{mr} - y_1 - G\hat{e} \quad (1.7) \]

where,
\[ \hat{e}^T = [u(t) - u(t - 1), u(t + 1) - u(t - 1), ..., u(t + n_2 - 1) - u(t - 1)]^T \]

\[ G = \begin{bmatrix} g_1 & 0 & \ldots & 0 \\
                   g_2 & g_1 & \ldots & 0 \\
                   \vdots & & \ddots & \vdots \\
                   g_{n_2} & g_{n_2-1} & \ldots & g_1 \end{bmatrix} \]
\( \delta \) should be chosen to minimize some function of \( \epsilon \). The performance index \( J \) is chosen as:

\[
J = \epsilon^T \epsilon + \beta \delta^T \delta
\]

(1.8)

where,

\( \beta \): a weighting factor to trade performance against control effort.

The control law is given by:

\[
\delta = (G^T G + \beta I)^{-1} G^T (y_{mr} - \hat{y}_1)
\]

(1.9)

In the adaptive version of the MAC, the coefficients of the impulse response are obtained using a recursive identification procedure.

### 1.3.2 Generalized Predictive Control

The GPC design uses a structured model, CARIMA model, to represent the actual plant. Studying GPC is interesting because it is an umbrella under which many predictive controllers like the Extended Horizon (EHC) of Ydstie (1984), and the Extended Predictive Self Adaptive Controller (EPSAC) of De Keyser (1985), come as special cases. The GPC makes use of the different predictive control ideas in the literature. In contrast to MAC and DMC, GPC is able to handle unstable systems.

The model used to design a GPC is given by:

\[
A(q^{-1})y(t) = B(q^{-1})u(t - 1) + C(q^{-1})\zeta(t)/\Delta
\]

(1.10)

where,

\[
\begin{align*}
A(q^{-1}) & = a_0 + a_1 q^{-1} + \cdots + a_n q^{-n} \\
B(q^{-1}) & = b_0 + b_1 q^{-1} + \cdots + b_n q^{-n} \\
\Delta & = 1 - q^{-1}
\end{align*}
\]
For simplicity $C(q^{-1})$ is taken to be 1. The $j$-step ahead predictor is given by:

\[
y(t + j|t) = G_j(q^{-1})\Delta u(t + j - 1) + F_j(q^{-1})y(t) + E_j\zeta(t + j)
\]

\[
G_j(q^{-1}) = E_j(q^{-1})B(q^{-1})
\]

$E_j$ and $F_j$ are calculated by solving the Diophantine equation given by:

\[
1 = E_j(q^{-1})A\Delta + q^{-j}F_j(q^{-1})
\]

A significant contribution of the GPC is the recursive solution of Diophantine equation, Clarke, et al. (1987-a). In Clarke, et al. (1987-b) and McIntosh (1991), two alternative solutions of the Diophantine equation are given for the case $C(q^{-1}) \neq 1$.

The control law is derived based on ignoring the future noise sequence $\zeta(t + j)$. Let $f(t + j)$ be the component of $y(t + j)$ composed of signals which are known at $t$ such that:

\[
f(t + 1) = [G_1(q^{-1}) - g_{10}]\Delta u(t) + F_1y(t)
\]

\[
f(t + 2) = q[G_2(q^{-1}) - q^{-1}g_{21} - g_{20}]\Delta u(t) + F_2y(t)
\]

\[
\vdots
\]

\[
\text{etc.}
\]

where,

\[
G_i(q^{-1}) = g_{i0} + g_{i1}q^{-1} + \cdots
\]

Then, the predicted output can be written in the form:

\[
\hat{y} = G\Delta u + f
\]

where,

\[
\hat{y} = [\hat{y}(t + 1), \hat{y}(t + 2), \cdots, \hat{y}(t + n_2)]^T
\]
\[ \Delta u = [\Delta u(t), \Delta u(t+1), \ldots, \Delta u(t+n_2-1)]^T \]
\[ f = [f(t+1), f(t+2), \ldots, f(t+n_2)]^T \]
\[ G = \begin{bmatrix} g_0 & 0 & \cdots & 0 \\ g_1 & g_0 & \cdots & 0 \\ \vdots \\ g_{n_2-1} & g_{n_2-2} & \cdots & g_{n_2-n_u} \end{bmatrix} \]

and,
\[ g_{ij} = g_j \text{ for } j = 0, 1, \ldots, \leq i. \]

\( n_2 \): prediction horizon.

\( n_u \): control horizon.

The control is chosen to minimize a cost function given by:
\[ J = (\hat{y} - y_{mr})^T(\hat{y} - y_{mr}) + \beta \Delta u^T \Delta u \] (1.15)

The control law is given by:
\[ \Delta u = (G^T G + \beta I)^{-1}G^T(y_{mr} - f) \] (1.16)

1.4 Thesis contribution

A generalized predictive controller is derived based on state-space modelling. Not only is the case of plant-model match analyzed but also the case of plant-model mismatch. Analysis is carried out for open-loop stable as well as unstable systems. Three techniques are then introduced to analyze the proposed control scheme:

1. Perturbation analysis is effectively used to monitor the closed-loop poles and direct the choice of the controller tuning parameters.
Chapter 1. Introduction

2. A new general lemma, which gives sufficient stability conditions of state-feedback control schemes, is derived and used efficiently to assess the stability of the proposed control scheme.

3. Functional analysis is used as a tool to assess the stability of nonlinear systems.

The proposed controller is successfully implemented using Laguerre-filter models. Not only are SISO schemes implemented and analyzed but also SIMO and MIMO schemes. The main characteristics of the control system are:

- The adaptive GPC based on Laguerre-filter modelling is globally convergent. This result is extended to other predictive controllers such as MAC.

- Robustness can always be guaranteed by increasing the prediction horizon or the control weight. No SPR condition is required to guarantee stability.

- Laguerre-filter based models can be modified to allow modelling and control of unstable systems.

Expert-systems and fuzzy-logic techniques are adopted to produce a flexible, easy-to-use version of the Laguerre-filter based adaptive GPC. The main features of the expert controller are:

- It has an interactive user interface.

- To choose a model, a breadth-first strategy is used to adjust the pole of the Laguerre-filter model while a depth-first strategy is used to adjust the number of filters used in the model.

- Linguistic rules are used on-line to adjust the controller tuning parameters.
1.5 Thesis Outline

In Chapter 1, the main prediction and predictive control techniques have been reviewed.

In Chapter 2, a GPC formulation based on state-space modelling is derived and analyzed. The derivation of the control law is in Section 2.2.1. Tuning techniques and an illustrative example are also given there. In section 2.2.2, the case of a one-step control horizon is analyzed. First-order systems are studied followed by high order ones. The GPC problem is analyzed as a perturbation problem. Both small and large perturbations are studied. In section 2.3, the results of section 2.2 are used to study stable as well as unstable systems. New asymptotic as well as non-asymptotic results are reported. In Section 2.4, the case of plant-model match is considered. Nonlinear systems and linear time invariant systems are studied in Section 2.4.1 and Section 2.4.2, respectively.

In Chapter 3, an adaptive GPC based on Laguerre-filter modelling is introduced. In Section 3.2, modelling and identification using the Laguerre filters are put into prospective. In Section 3.3, stability and global convergence of SISO schemes are analyzed. Links with other predictive controllers such as MAC and Zervos' controller, Zervos (1988), are shown. Control of nonlinear systems and unstable systems is also considered. In Section 3.4, extensions to multivariable systems are introduced. Both SIMO and MIMO systems are studied. Stability and convergence analyses of the MIMO GPC are also considered. In Section 3.5, the Laguerre-filter based GPC is compared with the CARIMA-model based GPC and the MAC algorithms.

In Chapter 4, the Laguerre-filter based GPC is implemented using expert-system and fuzzy-logic techniques. In section 4.1, a survey of the expert-control literature is provided. In Section 4.2, the expert shell "G2" is briefly reviewed. In section 4.3, an implementation of the adaptive Laguerre-filter based GPC is described in detail. In Section 4.3.1, an overview of the implementation is outlined. The user interface is explained in Section
4.3.2. Model adjustment using AI search techniques is detailed in Section 4.3.3. On-line tuning and monitoring using fuzzy logic is described in Section 4.3.4. Two illustrative examples are given in Section 4.3.5 to demonstrate the main features of the expert GPC. In Appendix A, the major stages of knowledge acquisition and the main techniques of knowledge representation are pointed out. In Appendix B, the basic definitions of fuzzy logic are covered.

Chapter 5 concludes this thesis and provides some suggestions for further studies.
Chapter 2

Generalized Predictive Control

2.1 Introduction

Since generalized predictive controllers (GPC) were introduced in Mohtadi and Clarke (1986), they have received considerable attention. GPCs have shown success in controlling plants with unknown dead-times, unmodelled dynamics, and nonminimum-phase responses.

The design of a GPC proceeds by assuming a set point $y_{sp}$ which is known over the prediction horizon $[t, t + n_2]$. An auxiliary signal $y_c(t + i), i \in [t, t + n_2]$ is generated to provide a smooth transition from the actual plant output, $y(t)$, to the set-point $y_{sp}(t)$. The control law which drives $y(t)$ to $y_{sp}(t)$ is derived such that some function of the prediction error is minimized over the horizon. To generate the prediction error, output predictions are required. The mechanism for generating these predictions is similar to having a state estimator or an observer, Åström and Wittenmark (1984). As hinted in Bitmead, et al.(1989-b), the predictive part of a GPC is seen as incorporating an observer associated with the state feedback solution of a receding-horizon linear quadratic problem. However, the state estimator is normally hidden since the calculations are based on input-output models. Also, it is shown in Bitmead, et al.(1989-a) that the predictive control problem has an equivalent optimal control one. The solution of the equivalent Riccati equation is monotonically increasing. Were the solution monotonically non-increasing, stability conditions could be obtained.
There are two sources of difficulty in analyzing the GPC theoretically. First, the poles of the closed-loop system depend on the tuning parameters in a complex way which can be described only in a heuristic sense. Actually, there is no explicit expression to describe the relation between the closed-loop poles and the GPC tuning parameters. Second, a GPC utilizes a finite prediction horizon with a receding policy. So, it is not possible to make use of the readily available results in optimal-control literature. Assuming an infinite prediction horizon or a zero weight on the control increments are impractical but common assumptions to investigate the closed-loop poles. In this chapter, a version of the GPC is derived based on a state-space modelling. This enables us to put the closed-loop system in a form amenable for applying the perturbation analysis. Results from Dief (1982) are applied to give approximate formulas for the closed-loop eigenvalues. Using these formulas, it is possible to study the effect of varying the prediction horizon and/or the control weighting factor on the closed-loop poles. Furthermore, the limiting eigenvalues of the closed-loop system are derived. Also, bounds on the admissible perturbation are derived to ensure closed-loop stability. As a by-product of this formulation, the GPC can be put in an explicit observer plus state feedback form.

2.2 GPC based on a state space modelling

2.2.1 General case

Let the plant be represented by a general SISO state space model in the form:

\[ \dot{x}(t+1) = Ax(t) + bu(t) \]  \hspace{1cm} (2.17)

\[ y(t) = c^T x(t) \]  \hspace{1cm} (2.18)

where,

\( x, b, \) and \( c \) are \( n \times 1 \) vectors.
Chapter 2. Generalized Predictive Control

\( A \) is an \( n \times n \) matrix.

\( n \) is the model order.

Using a \( j \)-step ahead predictor, the output is given by:

\[
\hat{y}(t + j) = c^T x(t + j) \tag{2.19}
\]

where,

\[
x(t + j) = A^j x(t) + A^{j-1} bu(t) + A^{j-2} bu(t + 1) + \ldots + bu(t + j - 1) \tag{2.20}
\]

Eq.(2.20) can be derived by recursive use of Eq.(2.17). Assuming a prediction horizon \( n_2 \), the predicted output over that horizon can be written in the form:

\[
\hat{Y} = \bar{f} + G u \tag{2.21}
\]

where,

\[
\bar{f} = [f_1 \ldots f_j \ldots f_{n_2}]^T = c^T A^j x(t)
\]

\[
G = \begin{bmatrix}
g_0 & 0 & 0 & \ldots & 0 \\
g_1 & g_0 & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
g_{n_2-1} & g_{n_2-2} & \ldots & g_1 & g_0
\end{bmatrix}
\]

\[
u = [u(t), u(t + 1), \ldots, u(t + n_2 - 1)]^T
\]

Note that \( g_i, i = 0, \ldots, n_2 - 1 \) are the Markov coefficients of the system (2.17) and (2.18) and are given by

\[
g_j = c^T A^j b
\]

The above derivation of the predicted output does not make use of the output measurement available at time \( t \). It is more useful if \( y(t) \) is used as a base and the model
is implicitly used to predict the increments of the output over the prediction horizon. This formulation insures an unbiased tracking of the reference in case of a plant-model mismatch, Mehra (1979). Eq.(2.21) leads to

\[ \hat{Y} = l + Gu \]  

(2.22)

where,

\[ l = [l_1 \ldots l_j \ldots l_{n_2}]^T \]

\[ l_j = y(t) + \xi^T(A^j - I)\xi(t) \]

Eq.(2.22) can be rewritten as

\[ \hat{Y} = l + Gu_1 + G(u - u_1) \]  

(2.23)

The vector \( u_1 \) can be, in general, any reference input trajectory defined by the designer. However in the following derivation, \( u_1 \) is chosen as the input at \((t - 1)\), i.e.

\[ u_1 = [u(t - 1) \ldots u(t - 1)]^T \]

Eq.(2.23) can be written as

\[ \hat{Y} = l + Gu_1 + S\Delta u \]  

(2.24)

where,

\[ \Delta = 1 - q^{-1} \]

\( S \) is a matrix of the step-response coefficients and is given by

\[
S = \begin{bmatrix}
s_0 & 0 & 0 & \ldots & 0 \\
s_1 & s_0 & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
s_{n_2 - 1} & s_{n_2 - 2} & \ldots & s_1 & s_0
\end{bmatrix}
\]
The step-response coefficients are related to the impulse-response coefficients by

\[ s_0 = g_0 \]

\[ s_i = g_i + s_{i-1}, \ i = 1, \ldots, n_2 - 1 \]

The vector \( \Delta u \) is given by

\[ \Delta u = [\Delta u(t), \Delta u(t+1), \ldots, \Delta u(t+n_2-1)]^T \]

The three terms in the right hand side of Eq.(2.24) can be interpreted as follows

- the first term represents the free response of the system.
- the second term represents the forced response of the system if the control signal is kept constant at its value at \( (t-1) \).
- the third term represents the future increments of the output due to the \( n_2 \) incremental changes of the control signal.

To provide a smooth transition from the output \( y(t) \) to the set point \( y_{sp} \), an auxiliary reference model is used to generate a signal \( y_c \) which eventually converges to \( y_{sp} \). That model is given by

\[ y_c(t+j) = \alpha^j y(t) + (1 - \alpha^j) y_{sp} \]  

(2.25)

where,

\( \alpha \) is a tuning parameter chosen by the designer according to the required speed of convergence of \( y(t) \) to \( y_{sp} \).

The control signal is chosen to minimize a quadratic performance index given by

\[ J = \sum_{j=n_1}^{n_2} (\hat{y}(t+j) - y_c(t+j))^2 + \sum_{j=1}^{n_u} \beta (\Delta u(t+j-1))^2 \]  

(2.26)
Chapter 2. Generalized Predictive Control

\( n_2, n_1, \) and \( n_u \) are the maximum output horizon, the minimum output horizon, and the control horizon respectively. The effects of these horizons and the rules of thumb to choose them are discussed in detail in Clarke, et al. (1987).

The control law is derived by setting \( \frac{\partial J}{\partial \Delta u} = 0 \) which leads to

\[
\Delta u = (S_1^T S_1 + \beta I)^{-1} S_1^T (y_c - I - G u_1) \tag{2.27}
\]

\( S_1 \) consists of the first \( n_u \) columns of \( S \).

Using the receding action idea, the first component in \( u \) is applied to the system and the calculations are repeated at each sample.

Comments

- From Eq. (2.23), \( \hat{y}(t + j) \) is given by

\[
\hat{y}(t + j) = e^T \hat{x}(t + j) + y(t) - e^T \hat{x}(t)
\]

\[
= e^T [A^j \hat{x}(t) + A^{j-1}bu(t) + \ldots + bu(t + j - 1) + k_f(y(t) - e^T \hat{x}(t))]
\]

\[
= e^T \hat{x}(t + j/t) 
\tag{2.28}
\]

Where \( \hat{x}(t + j/t) \) is the \( j \)-step ahead prediction of the state vector given measurements up to time \( t \), and

\[
e^T k_f = 1 
\tag{2.29}
\]

Eq. (2.28) shows that the \( j \)-step ahead predictor implicitly uses an observer.

- Choosing \( n_1 = 1, n_2 = n_u = n \) and \( \lambda = 0 \), Eq. (2.24)-(2.27) lead to

\[
u(t) = \frac{1}{s_0} [y_c(t + 1) - y(t) - e^T (A^j - I) \hat{x}(t)] 
\tag{2.30}
\]

Eq. (2.30) is equivalent to the control law which minimizes

\[
J = [\hat{y}(t + 1) - y_c(t + 1)]^2 
\tag{2.31}
\]
A similar result is shown in Rouhani, et al.(1982) for the model algorithmic control. Using this fact it will be shown below that the GPC can achieve an unbiased tracking of the set-point in the presence of unmodelled dynamics. In case of perfect plant-model match the control law uses the inverse of the model so the above tuning parameters cannot be used in controlling nonminimum-phase systems.

Assume the plant is described by

\[ Y(z) = H(z)U(z) \]  \hspace{1cm} (2.32)

and the control law is based on a model given by

\[ Y_m = \tilde{H}(z)U(z) \]  \hspace{1cm} (2.33)

The one-step predictor is given by

\[ zY(z) = zH(z)U(z) + [H(z) - \tilde{H}(z)]U(z) \]  \hspace{1cm} (2.34)

and the command signal is

\[ zY_c(z) = \alpha H(z)U(z) + (1 - \alpha)Y_{sp} \]  \hspace{1cm} (2.35)

The control law satisfies

\[ Y_c(z) = \dot{Y}(z) \]  \hspace{1cm} (2.36)

Eqs.(2.34)-(2.36) lead to

\[ U(z) = \frac{1 - \alpha}{(z - 1)\dot{H}(z) + (1 - \alpha)H(z)}Y_{sp} \]  \hspace{1cm} (2.37)

\[ Y(z) = \frac{(1 - \alpha)H(z)}{(z - 1)\dot{H}(z) + (1 - \alpha)H(z)}Y_{sp} \]  \hspace{1cm} (2.38)

Eq.(2.38) shows that \( \lim_{z \to 1} \frac{Y(z)}{Y_{sp}} = 1 \), i.e. the closed-loop system achieves perfect tracking. On the other hand if \( \dot{H}(z) = H(z) \), then Eq.(2.37) gives

\[ U(z) = \frac{1 - \alpha}{(z - \alpha)H(z)}Y_{sp} \]  \hspace{1cm} (2.39)
Chapter 2. Generalized Predictive Control

It is clear that the control law uses the inverse of the plant transfer function. Hence, the plant should be a minimum phase one.

- If the open-loop system is completely controllable and observable, choosing $\beta = 0$, $n_1 = n$, $n_2 \geq 2n - 1$, and $n_u = n$ will result in a dead-beat controller. The proof is the same as in Clarke, et al.(1987).

Illustrative example

Consider the nonminimum-phase and unstable plant, Wertz (1987), given by

$$y(t) - 2.7y(t - 1) + 1.8y(t - 2) = u(t - 1) + 5u(t - 2) + 6u(t - 3) \quad (2.40)$$

Assume the plant parameters are known. The effect of the GPC tuning parameters will be studied. Since, there are two unstable poles, $n_u$ should be at least two. This can be understood by noting that a plant of order $n$ needs $n$ different control values for dead-beat response. With a complex system these values may change sign so frequently that a short control-horizon does not allow enough degrees of freedom in the control action to stabilize the system, Clarke, et al.(1987). Indeed, choosing $n_u = 1, n_2 = 10$ and $\beta = 1$, leads to instability as shown in Fig.(2.1). By increasing $n_u$ to 2, the number of unstable modes, the closed-loop system becomes stable as shown in Fig.(2.2). Increasing $n_u$ results in a more active control signal and a faster output response with overshoot as in Fig.(2.3). Increasing $\beta$ from 1 to 10 results in a great damping of the output response as can be seen by comparing Fig.(2.3) and (2.4). Of course, increasing $\beta$ leads to a slower output response. If Figs.(2.2) and (2.5) are compared, the effect of increasing $n_2$ can be noticed. The output response as well as the control action get slower and smoother as $n_2$ increases.
Figure 2.1: GPC $n_2 = 10, \beta = 1$ and $n_u = 1$

Figure 2.2: GPC $n_2 = 10, \beta = 1$ and $n_u = 2$
Chapter 2. Generalized Predictive Control

Figure 2.3: GPC $n_2 = 10, \beta = 1$ and $n_u = 5$

Figure 2.4: GPC $n_2 = 10, \beta = 10$ and $n_u = 5$
2.2.2 Case of a single control-horizon

Consider the regulator problem, $y_{sp} = 0$. Assume $n_u = 1$ and $\alpha = 0$. This allows $(n_2 - 1)$ steps of infinite weighting on the incremental changes of the control action followed by one step of finite weight $\beta$. This choice is valid as long as the plant is stable with fairly damped poles as shown in Clarke, et al. (1987). The advantages of choosing $n_u = 1$ are

- As shown later, the resulting control law is naturally robust against unmodelled dynamics. So, it is attractive for adaptive control applications.

- It may lead to a solution of the Riccati Equation which is monotonically decreasing, Bitmead, et al. (1989-a). Whence, stability would follow.

- It is computationally efficient as no matrix inversion is required.
Chapter 2. Generalized Predictive Control

Under the above conditions, the control law given by Eq.(2.27) can be written as

$$u(t) = \epsilon[k^T \bar{x}(t) + \beta u(t - 1)] \quad (2.41)$$

where,

$$k^T = -\sum_{i=0}^{n_2-1} s_i \xi^T A^{i+1}$$

$$\epsilon = \frac{1}{\bar{s}^T \bar{s} + \beta}$$

$$\bar{s}^T = [s_0 \ s_1 \ \ldots \ s_{n_2-1}]$$

Using Eqs. (2.17) and (2.41), the closed loop system is:

$$Z(t + 1) = (\Lambda + \epsilon A_1)Z(t) \quad (2.42)$$

where,

$$Z(t) = \begin{bmatrix} \bar{x}(t) \\ u(t) \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} A & b \\ 0 & 0 \end{bmatrix}$$

$$A_1 = \begin{bmatrix} 0 & 0 \\ k^T A & k^T b + \beta \end{bmatrix}$$

Equation (2.42) shows the GPC problem can be formalized as a perturbation problem.

In case of $\beta = 0$, the perturbed system will be:

$$\bar{x}(t + 1) = (A + \epsilon \bar{A}_1)\bar{x}(t) \quad (2.43)$$

where,

$$\epsilon = \frac{1}{\bar{s}^T \bar{s}} \quad (2.44)$$
Eq. (2.44) indicates that in order to guarantee a finite value of $\epsilon$, the prediction horizon should be longer than the plant's dead time. Equation (2.45) shows that the prediction horizon should be greater than the non-minimum phase response to avoid the possibility of a feedback gain with a wrong sign.

In the GPC literature, the analysis usually assumes that either the prediction horizon is infinite or it can be approximated as an infinite one. Practically, this assumption is not always realistic. For a first-order system without a delay, analytical analysis is possible. Specifically, the case of a first-order unstable system is analyzed below. For multi-variable systems, the perturbation analysis is introduced to deal with the case of a finite prediction horizon. As shown above, the GPC problem can be dealt with as a perturbation introduced to the open-loop system. If the perturbation is small, approximate formulas are derived to calculate the closed-loop eigenvalues. On the other hand, if the perturbation is not necessarily small, bounds on the perturbation which guarantee the closed-loop stability are derived.

**Case of a first-order system**

It is stated in Clarke, et al. (1987) that the GPC can stabilize any first-order open-loop stable system for any $n_2 > 0$. A first-order unstable system is analyzed below. Let the system be given by

$$
\frac{y(t)}{u(t)} = \frac{k}{z - a}, \quad a > 1
$$

(2.46)

We have the following claim.

**Claim**: If $a > 1$, the GPC is still a stabilizing controller as long as $n_2$ is finite.
Chapter 2. Generalized Predictive Control

Proof:
Choose $\beta = 0$. Using the predictive control law in Eq.(2.41), the system’s closed-loop pole will be

$$\bar{a} = \frac{an_2(1-a^2) - a(1-a^{n_2})(1+a)^2 + a^2(1-a^{2n_2})}{n_2(1-a^2) - 2a(1+a)(1-a^{n_2}) + a^2(1-a^{2n_2})}$$  \hspace{1cm} (2.47)

By substituting,

$$a = 1 + \mu \quad , \quad \mu > 0$$  \hspace{1cm} (2.48)

in Eq.(2.47), it is possible to show after some algebraic manipulations that both the numerator and the denominator of Eq.(2.47) are negative for any $n_2 \geq 2$, ($\bar{a} = 0$ for $n_2 = 1$).

Assuming that the closed-loop system is unstable; i.e $\bar{a} \geq 1$, Eq.(2.47) leads to

$$\psi = n_2(a-1) + a(1-a^{n_2}) \geq 0$$  \hspace{1cm} (2.49)

Substituting Eq.(2.48) in Eq.(2.49) we get.

$$\psi = n_2\mu - (1+\mu) \sum_{i=1}^{n_2} \frac{n_2!}{(n_2-i)!i!} \mu^i$$
$$= -n_2\mu^2 - (1+\mu) \sum_{i=2}^{n_2} \frac{n_2!}{(n_2-i)!i!} \mu^i$$  \hspace{1cm} (2.50)

It is clear that $\psi < 0$ which forms a contradiction to the assumption $\bar{a} \geq 1$. So, we conclude that

$$\bar{a} < 1$$  \hspace{1cm} (2.51)

By induction, it is easy to show that the above inequality holds for all values $n_2 < \infty$. \hfill \Box

Note that the above analysis holds only if $a > 1$. This condition is naturally satisfied if Eq.(2.46) is obtained by sampling a continuous first-order unstable system. If $a < -1$, Theorem 2.3 guarantees that $\bar{a} \rightarrow 1$ as $n_2 \rightarrow \infty$. However, stability cannot be guaranteed
for any prediction horizon as in the case $a > 1$. Fig.(2.6) shows the typical behaviors of the GPC closed-loop pole for $|a| < 1$, $a > 1$, and $a < -1$. The three-dimensional graph in Fig.(2.6) can be divided into three regions. The upper left corner of the graph corresponds to $a < -1$. As $n_2$ changes, the closed-loop pole, $\bar{a}$, exhibits an oscillating behavior. For some values of $n_2$, $\bar{a}$ is outside the unit circle. As $n_2 \to \infty$, $\bar{a}$ converges to one. The middle region corresponds to $|a| < 1$. It is clear that $\bar{a}$ is inside the unit circle for any value $n_2$. As $n_2 \to \infty$, $\bar{a}$ converges to $a$. The lower right corner corresponds to $a > 1$. The closed-loop system is stable for any finite $n_2$ as all the curves converge to 1 from below. Typical curves from the three regions are shown in the lower graph of Fig.(2.6).

Small perturbation

For sufficiently small $\epsilon$, the formulation of Eq.(2.42) suggests that the perturbation theorem can be used to find the closed-loop poles, Elshafei, et. al. (1991). The immediate question is: how small should $\epsilon$ be? The practical experience with the GPC shows that it is sufficient to choose $n_2$ to correspond to the time which the plant's output needs to reach 90% of its steady-state value and $\beta \geq 0$. The following lemmas, proved in Deif (1982), give, to a first approximation, the eigenvalues $\tilde{\lambda}$ of the perturbed system in Eq.(2.42) as related to the eigenvalues $\lambda$ of the matrix $\Lambda$.

**Lemma 2.1** If $\lambda_i$ is a distinct eigenvalue of a semi-simple matrix$^1$ $\Lambda$ with the corresponding eigenvector $u^i$, the eigenvalue $\tilde{\lambda}_i$ of the perturbed matrix $\Lambda + \epsilon A_1$ is given for the first order approximation by:

$$\tilde{\lambda}_i \approx \lambda_i + \epsilon < u^i, A_1 u^i >$$  \hspace{1cm} (2.52)

$^1$A matrix is semi-simple if it is diagonalizable.
Typical Effect of $n_2$ on the Pole Location

Figure 2.6: GPC of a first-order system
where \( u^1, \ldots, u^n \) are the eigenvectors of \( A \) and \( v^1, \ldots, v^n \) are their reciprocal basis.

**Lemma 2.2** If \( \lambda \) is a semi-simple eigenvalue \(^2\) of multiplicity \( m \) of a matrix \( A \) with corresponding eigenvectors \( u^1, \ldots, u^m \), the eigenvalues \( \tilde{\lambda}_1, \ldots, \tilde{\lambda}_m \) of \( A + \epsilon A_1 \) are given for the first order approximation by:

\[
\tilde{\lambda}_i \approx \lambda + \epsilon \lambda_i^{(1)}, \quad i = 1, \ldots, m
\]  

(2.53)

where \( \lambda_1^{(1)}, \ldots, \lambda_m^{(1)} \) are the eigenvalues of:

\[
S = [v^1^*] A \left[ \begin{array}{c} u^1 \ldots u^m \\ \vdots \\ v^m^* \end{array} \right]
\]  

(2.54)

\( u^1, \ldots, u^n \) are the eigenvectors of \( A \) and \( v^1^*, \ldots, v^n^* \) are the conjugate transpose of their reciprocal basis.

**Lemma 2.3** If \( \lambda \) is a non-semi-simple eigenvalue \(^3\) of multiplicity \( m \) of a matrix \( A \), with the corresponding generalized eigenvectors \( u^1, \ldots, u^m \), the eigenvalues \( \tilde{\lambda}_1, \ldots, \tilde{\lambda}_m \) of \( A + \epsilon A_1 \) will lie on the circumference of a circle with center \( z \) and radius \( r \approx |\sqrt[\epsilon]{\lambda_i^{(1)}}| \), where

\[
z \approx \lambda + \frac{1}{m} \Sigma_{j=1}^{m} < v^j, \epsilon A_1 u^j >
\]  

(2.55)

\[
\lambda_i^{(1)} = \sqrt[\epsilon]{< v^m, A_1 u^1 >} e^{\frac{2 \pi i j}{m}}, \quad j = \sqrt{-1}, \quad i = 1, \ldots, m
\]  

(2.56)

The above lemmas provide a handy way to study the effect of \( n_2 \) and \( \beta \) on the closed-loop poles. The computational cost is small as the eigenvectors of \( A \) and their reciprocal bases are calculated only once and \( b^T \), the main burden in forming \( A_1 \), is required anyway in Eq.(2.41).

\(^2\)An eigenvalue is semi-simple if the dimension of the associated Jordan block is \( 1 \times 1 \).

\(^3\)An eigenvalue is non-semi-simple of multiplicity \( m \) if the dimension of the associated Jordan block is \( m \times m \).
General perturbation

It is clear that the GPC control law has a state-feedback term. The following lemma gives a sufficient stability condition which will be used to assess the stability of our control scheme.

Lemma 2.4 Let

\[ z(t + 1) = Az(t) \]  
\[ \exists P > 0 \forall Q > 0 \]  

such that

\[ A^T PA - P = -Q \]  

Then the perturbed system

\[ z(t + 1) = (A + \delta)z(t) \]  

is stable if

\[ 0 \leq ||\delta|| < -||A|| + \sqrt{||A||^2 + \frac{\lambda_{\min}(Q)}{||P||}} \]  

Proof: It is required to prove that

\[ (A + \delta)^T P(A + \delta) - P = -Q + \delta^T P\delta + A^T P\delta + \delta^T PA \]  

such that

\[ (-Q + \delta^T P\delta + A^T P\delta + \delta^T PA) \leq 0 \]  

It is possible to write

\[ v^T Qv \geq \lambda_{\min}(Q)v^Tv \]
Assume
\[ \|\delta\| \in [0, -\|A\| + \sqrt{\|A\|^2 + \frac{\lambda_{\min}(Q)}{\|P\|}}] \]  
(2.65)

Then, it is easy to show that
\[ \|P\|\|\delta\|^2 + 2\|P\|\|A\|\|\delta\| - \lambda_{\min} < 0 \]  
(2.66)

This leads to
\[ \lambda_{\min} > \|\delta^T P\delta + A^T P\delta + \delta^T P A\| \]  
(2.67)

Using Eqs. (2.64) and (2.67) leads to
\[ v^T Q v > v^T (\delta^T P\delta + A^T P\delta + \delta^T P A) v \]  
(2.68)

And the proof is complete. □

2.3 Case of Plant-model match

Eq.(2.41) is the state-space version of the GPC derived by Clarke, et al.(1987) for the case of a single-step control horizon and zero weighting on the control action. Analysis of the controller asymptotic behavior is available only for open-loop stable processes. Using the state space formulation, the controller asymptotic behavior will be studied below for both stable and unstable systems. Theorem 2.1 has been derived by Clarke, et al.[6]. However, we believe our proof is more straightforward. Theorem 2.2 presents a general case where asymptotic as well as arbitrary values of \( n_2 \) and \( \beta \) can be handled. Theorem 2.3 illustrates the asymptotic behavior of the GPC if the open-loop system is unstable

2.3.1 Analysis of open-loop stable systems
Chapter 2. Generalized Predictive Control

**Theorem 2.1** Let the system be given by Eqs. (2.17) and (2.18) and controlled using the control law in Eq. (2.41). Assume the eigenvalues of the open-loop system are \( \lambda_i, i = 1, \ldots, n \) where \( |\lambda_i| < 1 \). If \( \beta = 0 \) and \( n_2 \to \infty \) then \( \tilde{\lambda}_i \to \lambda_i \) where \( \tilde{\lambda}_i \) is the \( i \)th closed-loop eigenvalue.

**Proof:**

From Eqs. (2.43) and (2.45), the closed-loop eigenvalues are the eigenvalues of the matrix

\[
A = \frac{b \sum_{j=0}^{n_2-1} s_j \xi^T A^{j+1}}{\sum_{j=0}^{n_2-1} s_j^2}
\]

Since \( |\lambda(A)| < 1 \), it is clear that \( \lim_{n_2 \to \infty} \frac{\sum_{j=0}^{n_2-1} s_j \xi^T A^{j+1}}{\sum_{j=0}^{n_2-1} s_j^2} = 0 \). So, the proof is complete.

\[\square\]

**Theorem 2.2** Let the system be described by Eq. (2.17) and controlled by the control law in Eq. (2.41). Assume the system is open-loop stable. Then, the closed-loop system described by Eq. (2.42) can be stabilized using Eq. (2.41). The closed-loop poles, \( \tilde{\lambda}_i, i = 1, \ldots, n + 1 \), are given by either Lemma 2.1, Lemma 2.2, or Lemma 2.3, as appropriate. If \( \epsilon \to 0 \), then:

- \( \tilde{\lambda}_i \to \lambda_i \), \( i = 1, \ldots, n \).
- \( \tilde{\lambda}_{n+1} \to \epsilon \beta \).

where \( \lambda_i, i = 1, \ldots, n \) are the open-loop poles.

**Proof:** According to the form of Eq. (2.42), it is clear that Lemma 2.1 - Lemma 2.3 can be applied to find the system's eigenvalues.

Note that \( \|k^T\| \) is always finite because the system is open-loop stable. If \( \epsilon \to 0 \), then \( \|\epsilon k^T\| \to 0 \).
The closed-loop equation will be:

\[
\begin{bmatrix}
\dot{x}(t+1) \\
u(t+1)
\end{bmatrix} =
\begin{bmatrix}
A & b \\
0 & c \beta
\end{bmatrix}
\begin{bmatrix}
\dot{x}(t) \\
u(t)
\end{bmatrix}
\]

(2.69)

The result of the Theorem follows immediately.

Comments:

• In Clarke, et al. (1987), it is proved that, for stable systems, the GPC is a mean level controller if the tuning parameters are chosen such that \( \beta = 0 \), \( n_u = 1 \), and \( n_2 \to \infty \). The above theorem reaches the same conclusion without assuming \( \beta = 0 \) nor \( n_2 \to \infty \) and so is less restrictive.

• \( \tilde{\lambda}_{n+1} \) is always such that \( 0 < \tilde{\lambda}_{n+1} < 1 \). As \( \beta \to \infty \) then \( \tilde{\lambda}_{n+1} \to 1 \) which explains the well-known observation that the closed-loop response gets slower as the control weighting increases.

If the case of \( \beta = 0 \) is considered, Lemma 2.1 and Lemma 2.2 can be elaborated to give computationally easier formulas to calculate the perturbed eigenvalues.

Under the conditions of Lemma 2.1, the eigenvalues of Eq.(2.43) can be calculated using Eqs.(2.52) and (2.45) as follows:

\[
\tilde{\lambda}_i = \lambda_i - \epsilon \Sigma_{j=0}^{n_2-1} s_j b e^T A^{j+1} u^i
\]

(2.70)

\[
\tilde{\lambda}_i = \lambda_i + \epsilon_1(n_2) \zeta_i
\]

(2.71)

where,

\[
\epsilon_1(n_2) = -\epsilon \Sigma_{j=0}^{n_2-1} s_j \lambda_i^{j+1}
\]

(2.72)
Chapter 2. Generalized Predictive Control

\[ \zeta_i = \langle v^i, b \xi^T u^i \rangle \] (2.73)

Under the conditions of Lemma 2.2, the eigenvalues of Eq.(2.43) can be calculated using Eqs.(2.53) and (2.45) as follows:

\[
S = \begin{bmatrix}
    v_1^{1*} \\
    \vdots \\
    v_m^{m*}
\end{bmatrix} + (-\Sigma_{j=0}^{n_2-1} s_j b \xi^T A^{j+1}) \begin{bmatrix}
    u_1 \\
    \vdots \\
    u_m
\end{bmatrix}
\] (2.74)

\[ \tilde{\lambda}_i = \lambda + \epsilon_1(n_2) \tilde{\lambda}_i \] (2.75)

where \( \tilde{\lambda}_i \) is the \( i^{th} \) eigenvalue of:

\[
\tilde{S} = \begin{bmatrix}
    v_1^{1*} \\
    \vdots \\
    v_m^{m*}
\end{bmatrix} + b \xi^T \begin{bmatrix}
    u_1 \\
    \vdots \\
    u_m
\end{bmatrix}
\] (2.76)

and \( \epsilon_1(n_2) \) is as given in Eq.(2.44).

Theorem 2.2 is still valid with the exception that \( \tilde{\lambda}_{n+1} \) is disregarded as the order of the closed-loop system is \( n \).

2.3.2 Analysis of open-loop unstable systems

Using a GPC to control an open-loop unstable system is analyzed below. First, a heuristic approach is taken to give insight into the problem. Then, a formal analysis is given.

Consider a first-order unstable system given by

\[ x_1(t+1) = \lambda_1 x_1(t) + b_1 u(t), \quad |\lambda_1| > 1 \] (2.77)

A GPC having \( n_u = 1 \) and \( n_2 \rightarrow \infty \) is equivalent to using a constant control-signal \( u \).

The closed-loop system will not explode iff

\[ x_1(t+1) = x_1(t) \] (2.78)
giving
\[ u = \frac{1 - \lambda_1}{b_1} x_1 \] (2.79)
It is clear from Eq.(2.78) that the closed-loop pole is at 1. Now, assume a general system that has one unstable pole
\[
\begin{bmatrix}
  x_1(t + 1) \\
  x_2(t + 1) \\
  \vdots \\
  x_n(t + 1)
\end{bmatrix} =
\begin{bmatrix}
  \lambda_1 & & & \\
  & J & & \\
  & & \ddots & \\
  & & & 1
\end{bmatrix}
\begin{bmatrix}
  x_1(t) \\
  x_2(t) \\
  \vdots \\
  x_n(t)
\end{bmatrix} +
\begin{bmatrix}
  b_1 \\
  b_2 \\
  \vdots \\
  b_n
\end{bmatrix} u(t)\] (2.80)
where \( J \) is a Jordan canonical form with all diagonal elements less than 1. Using the same idea used for the first-order system given by Eq.(2.78), the closed-loop system does not explode if
\[ u = \frac{1 - \lambda_1}{b_1} x_1 \]
\[ = \frac{1 - \lambda_1}{b_1} \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \] (2.81)
Substituting Eq.(2.81) in Eq.(2.80) shows that one closed-loop pole is at 1 and the others lie at the stable open-loop poles.

To consider systems with more than one open-loop unstable pole, assume a system given by Eq.(2.77) and
\[ x_2(t + 1) = \lambda_2 x_2(t) + b_2 u(t); \quad |\lambda_2| > 1 \] (2.82)
If the closed-loop system is not to explode, it is necessary and sufficient to satisfy Eq.(2.78) and
\[ x_2(t + 1) = x_2(t) \] (2.83)
using a constant control signal $u$. Generally, this is not possible as it requires one un­
known, $u$, to satisfy two independent linear equations. Now, it is possible to proceed to
a formal study of systems that have one unstable pole.

**Lemma 2.5** Let the system be given by Eqs. (2.17) and (2.18). Assume the eigenvalues
of the open-loop system, $\lambda_i, i = 1, \ldots, n$, are distributed such that:

- $|\lambda_1| > 1$.
- $|\lambda_i| < 1 \quad i = 2, \ldots, n$.

then,

$$\lim_{n \to \infty} e_k^T = \begin{bmatrix} \frac{1 - \lambda_1}{b_1} & 0 & \cdots & 0 \end{bmatrix}$$ (2.84)

where $k^T$ and $e$ are given in Eq.(2.41) and Eq.(2.44), respectively.

**Proof:**
Assume, for simplicity, that the system is diagonalizable, then

$$A = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \quad e^T = \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix}$$

$$s_j = \sum_{k=0}^{j} e^T A^k b = \sum_{k=0}^{j} \sum_{i=1}^{n} \alpha_i \lambda_i^k$$ (2.85)

where,

$$\alpha_i = c_i b_i, \quad i = 1, \ldots, n$$ (2.86)
Eq.(2.85) leads to

\[ s_j = \beta_0 + \sum_{i=1}^{n} \beta_i \lambda_i^{j+1} \]  

(2.87)

where,

\[ \beta_0 = \sum_{i=1}^{n} \frac{\alpha_i}{1 - \lambda_i} \]  

(2.88)

\[ \beta_i = -\frac{\alpha_i}{1 - \lambda_i}, \quad i = 1, \ldots, n \]  

(2.89)

Let \( \sigma = \sum_{j=0}^{n-1} s_j^2 \), then

\[ \sigma = \sum_{j=0}^{n-1} \left[ \beta_0^2 + \sum_{i=1}^{n} \beta_i^2 \lambda_i^{2j+2} + 2 \beta_0 \sum_{i=1}^{n} \beta_i \lambda_i^{j+1} + 2 \beta_1 \sum_{i=2}^{n} \beta_i (\lambda_{i-1} \lambda_i) \lambda_{i+1} + \ldots + 2 \beta_{n-1} \lambda_n (\lambda_{n-1} \lambda_n) \right] \]  

(2.90)

The above equation leads to

\[ \sigma = \beta_0^2 n_2 + \sum_{i=1}^{n} \beta_i^2 \lambda_i^2 \frac{1 - \lambda_i^{2n_2}}{1 - \lambda_i^2} + 2 \beta_0 \sum_{i=1}^{n} \beta_i \lambda_i \frac{1 - \lambda_i^{n_2}}{1 - \lambda_i} + \]  

(2.91)

\[ 2 \beta_1 \sum_{i=2}^{n} \beta_i \lambda_1 \lambda_i \frac{1 - (\lambda_{i-1} \lambda_i)^{n_2}}{1 - \lambda_1 \lambda_i} + \ldots + 2 \beta_{n-1} \lambda_n (\lambda_{n-1} \lambda_n) \frac{1 - (\lambda_{n-1} \lambda_n)^{n_2}}{1 - \lambda_{n-1} \lambda_n} \]

Let \( \bar{s}_i = \sum_{j=0}^{n-1} s_j \lambda_i^{j+1} \), then

\[ \bar{s}_i = \sum_{j=0}^{n_2-1} \left[ \beta_0 \lambda_i^{j+1} + \sum_{k=1}^{n} \beta_k (\lambda_i \lambda_k)^{j+1} \right] \]

\[ = \beta_0 \lambda_i^{j+1} + \sum_{k=1}^{n} \beta_k \lambda_i \lambda_k \frac{1 - (\lambda_i \lambda_k)^{n_2}}{1 - \lambda_i \lambda_k} \]

(2.92)

Using Eqs.(2.91) and (2.92), it is possible to show that

\[ \gamma_i = \lim_{n_2 \to \infty} \frac{\bar{s}_i}{\sigma} = \begin{cases} \left( \frac{1 - \lambda_i}{\alpha_i} \right) & \text{if } i = 1 \\ 0 & \text{otherwise} \end{cases} \]  

(2.93)

The result of the lemma follows directly from Eq.(2.93). \( \square \)
Theorem 2.3 Let the system be described by Eq. (2.17), with the corresponding open-loop poles \( \lambda_i, \ i = 1, \ldots, n \) be controlled by the control law given in Eq. (2.41). Assume the system has one unstable pole, \( |\lambda_1| > 1 \) and \( (n - 1) \) stable poles, \( |\lambda_i| < 1, \ i = 2, \ldots, n \). Then, for a sufficiently small \( \epsilon \), the closed-loop system described by Eq. (2.42) will have its closed-loop poles, \( \tilde{\lambda}_i, \ i = 1, \ldots, n + 1 \), given by either Lemma 2.1, Lemma 2.2, or Lemma 2.3, as appropriate. Furthermore, if \( n_2 \to \infty \), then the closed-loop poles will be such that:

- \( \tilde{\lambda}_1 \to 1. \)
- \( \tilde{\lambda}_i \to \lambda_i, \quad i = 2, \ldots, n. \)
- \( \tilde{\lambda}_{n+1} \to 0. \)

Proof: According to Eq. (2.42), Lemma 2.1 - Lemma 2.3 can be applied to find the system's eigenvalues. Assume for simplicity that the system is diagonalizable. Then, it is possible to write:

\[
A = \begin{bmatrix}
\lambda_1 & 0 & \ldots & 0 \\
0 & \lambda_2 & 0 \\
\vdots & \ddots & \ddots \\
0 & \ldots & 0 & \lambda_n
\end{bmatrix}, \quad \vec{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \quad \vec{e}_T = \begin{bmatrix} c_1 & c_2 & \ldots & c_n \end{bmatrix}
\]

Using Lemma 2.5, it is possible to show that:

\[
\lim_{n_2 \to \infty} \epsilon_k^T = \begin{bmatrix}
\frac{1 - \lambda_1}{b_1} & 0 & \ldots & 0
\end{bmatrix}
\]

(2.94)

Eq. (2.94) leads to

\[
\epsilon_k^T A = \begin{bmatrix}
\frac{\lambda_1(1 - \lambda_1)}{b_1} & 0 & \ldots & 0
\end{bmatrix}
\]

(2.95)

and,

\[
\epsilon_k^T \vec{b} = 1 - \lambda_1
\]

(2.96)
Chapter 2. Generalized Predictive Control

The closed-loop poles are the solutions of:

\[ |zI - \Lambda_1| = 0 \]  \hspace{1cm} (2.97)

where,

\[
\Lambda_1 = \begin{bmatrix}
\lambda_1 & 0 & \ldots & 0 & b_1 \\
0 & \lambda_2 & 0 & b_2 \\
\vdots & \ddots & \vdots & \vdots \\
0 & \ldots & 0 & \lambda_n & b_n \\
\frac{\lambda_1(1-\lambda_1)}{b_1} & 0 & \ldots & 0 & 1 - \lambda_1
\end{bmatrix}
\]  \hspace{1cm} (2.98)

Consider the following identity, Kailath (1980):

\[
\begin{vmatrix}
M_1 & M_2 \\
M_3 & M_4
\end{vmatrix} = |M_4||M_1 - M_2M_4^{-1}M_3|, \quad |M_4| \neq 0 \]  \hspace{1cm} (2.99)

Choose,

\[
M_1 = zI - A \\
M_2 = -b \\
M_3 = \begin{bmatrix}
-\frac{\lambda_1(1-\lambda_1)}{b_1} & 0 & \ldots & 0 \\
\vdots & \ddots & \ddots & \vdots \\
-\frac{b_n\lambda_1(1-\lambda_1)}{b_1(z-(1-\lambda_1))} & \ldots & 0 & z - \lambda_n
\end{bmatrix} \\
M_4 = z - (1 - \lambda_1) \]  \hspace{1cm} (2.100)

Hence,

\[
|zI - \Lambda_1| = (z - (1 - \lambda_1)) \begin{vmatrix}
(z - \lambda_1) - \frac{\lambda_1(1-\lambda_1)}{z-(1-\lambda_1)} & 0 & \ldots & 0 \\
-\frac{b_2\lambda_1(1-\lambda_1)}{b_1(z-(1-\lambda_1))} & z - \lambda_2 & 0 \\
\vdots & \ddots & \ddots & \vdots \\
-\frac{b_n\lambda_1(1-\lambda_1)}{b_1(z-(1-\lambda_1))} & \ldots & 0 & z - \lambda_n
\end{vmatrix} = (z - 1)(z - \lambda_2)\ldots(z - \lambda_n)z \]  \hspace{1cm} (2.101)
Chapter 2. Generalized Predictive Control

Theorem 2.3 states that $\bar{\lambda}_1 = 1 + \delta$ as $n_2 \to \infty$. Whether $\delta$ is positive or negative can be assessed using either Lemma 2.1, Lemma 2.2, or Lemma 2.3. Theorem 2.3 is, in fact, a good tool to find the upper and lower bounds of $n_2$ for which the GPC remains a stabilizing controller.

Comments:

- Assume that the open-loop plant has $m$ unstable poles such that $|\lambda_i| > |\lambda_{i+1}|, i = 1, \ldots, m$. It is easy to show that $\bar{\lambda}_1 \to 1$ and $\bar{\lambda}_i \to \lambda_i, i = 2, \ldots, m$.

- Theorem 2.3 is true even if the system is non-diagonalizable as shown by the above analysis of Eq.(2.80). Example 1 demonstrates this fact.

- It is logical to restrict the analysis to systems having one unstable pole because it is well known that a single-step control horizon is not adequate to stabilize systems having more than one unstable pole.

Example 1: Consider the third-order non-diagonalizable system

\[
A = \begin{bmatrix}
\lambda_1 & 0 & 0 \\
0 & \lambda & 1 \\
0 & 0 & \lambda
\end{bmatrix}, \quad b = \begin{bmatrix}
b_1 \\
b_2 \\
b_3
\end{bmatrix}, \quad e^T = \begin{bmatrix}
c_1 & c_2 & c_3
\end{bmatrix}
\] (2.102)

where, $|\lambda_1| > 1$ and $|\lambda| < 1$.

As shown in Brogan [7]

\[
A^j = \begin{bmatrix}
\lambda_1^j & 0 & 0 \\
0 & \lambda^j & j\lambda^{j-1} \\
0 & 0 & \lambda^j
\end{bmatrix}
\] (2.103)
The step-response coefficients are

\[ s_j = \sum_{i=0}^{j} c^T A^i b \]

\[ = \beta_0 + \beta_1 \lambda_j + \beta_2 \lambda^{2j} + \beta_3 j \lambda^j \]  

(2.104)

where,

\[ \beta_0 = \frac{c_1 b_1}{1 - \lambda} + \frac{c_2 b_2 + c_3 b_3}{1 - \lambda} - \frac{c_2 b_3}{(1 - \lambda)^2} \]  

(2.105)

\[ \beta_1 = \frac{c_1 b_1 \lambda}{1 - \lambda} \]  

(2.106)

\[ \beta_2 = \frac{(c_2 b_2 + c_3 b_3) \lambda}{1 - \lambda} - \frac{c_2 b_3}{(1 - \lambda)^2} \]  

(2.107)

\[ \beta_3 = -\frac{c_2 b_3}{1 + \lambda} \]  

(2.108)

Note that

\[ \sum_{j=0}^{n_2-1} j \lambda^j = \sum_{j=0}^{n_2-1} \lambda \frac{d}{d \lambda} \lambda^j = \lambda \frac{d}{d \lambda} \frac{1 - \lambda^{n_2}}{1 - \lambda} \]  

(2.109)

\[ \sum_{j=0}^{n_2-1} j^2 \lambda^{2j} = \sum_{j=0}^{n_2-1} \frac{d^2}{d^2 \lambda^2} (\lambda^2)^{j+2} - \frac{d}{d \lambda} \lambda^{3j} - 2 \lambda^{2j} \]

\[ = \frac{d^2}{d^2 \lambda^2} \lambda^4 \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} - \frac{d}{d \lambda} \frac{1 - \lambda^{3n_2}}{1 - \lambda^3} - 2 \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} \]  

(2.110)

Hence,

\[ \sigma = \sum_{j=0}^{n_2-1} s_j^2 \]

\[ = n_2 \beta_0^2 + \beta_1^2 \frac{1 - \lambda^{n_2}}{1 - \lambda^2} + \beta_2^2 \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} + 2 \beta_0 \beta_1 \frac{1 - \lambda^{n_2}}{1 - \lambda^2} + \beta_0 \beta_2 \frac{1 - \lambda^{n_2}}{1 - \lambda^2} + \]

\[ 2 \beta_0 \beta_3 \lambda \frac{d}{d \lambda} \frac{1 - \lambda^{n_2}}{1 - \lambda^2} + 2 \beta_1 \beta_2 \frac{1 - (\lambda_1 \lambda)^{n_2}}{1 - \lambda^2} + 2 \beta_2 \beta_3 \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} + \]

\[ 2 \beta_1 \beta_3 \lambda \frac{d}{d \lambda} \frac{1 - (\lambda_1 \lambda)^{n_2}}{1 - \lambda \lambda_1} + \beta_3^2 \frac{d^2}{d^2 \lambda^2} \lambda \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} - \]
\[
\frac{d}{d\lambda} \left[ \frac{1 - \lambda^{3n_2}}{1 - \lambda^2} - 2 \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} \right]
\]

\[
\sigma_1 = \sum_{j=0}^{n_2-1} s_j \lambda^{j+1}
\]
\[
= \beta_0 \lambda_1 \frac{1 - \lambda^{n_2}}{1 - \lambda_1} + \beta_1 \lambda_1 \frac{1 - \lambda^{2n_2}}{1 - \lambda_1^2} + \beta_2 \lambda_1 \frac{1 - (\lambda_1 \lambda)^{n_2}}{1 - \lambda_1 \lambda} + \beta_3 \lambda_1 \lambda \frac{d}{d\lambda} \frac{1 - (\lambda_1 \lambda)^{n_2}}{1 - \lambda_1 \lambda} \tag{2.111}
\]

\[
\sigma_2 = \sum_{j=0}^{n_2-1} s_j \lambda^{j+1}
\]
\[
= \beta_0 \lambda_1 \frac{1 - \lambda^{n_2}}{1 - \lambda} + \beta_1 \lambda \frac{1 - (\lambda_1 \lambda)^{n_2}}{1 - \lambda_1 \lambda} + \beta_2 \lambda \frac{1 - \lambda^{2n_2}}{1 - \lambda} + \beta_3 \lambda^2 \frac{d}{d\lambda} \frac{1 - \lambda^{2n_2}}{1 - \lambda} \tag{2.112}
\]

\[
\sigma_3 = \sum_{j=0}^{n_2-1} s_j j \lambda^j
\]
\[
= \beta_0 \lambda \frac{d}{d\lambda} \frac{1 - \lambda^{n_2}}{1 - \lambda} + \beta_1 \lambda \frac{d}{d\lambda} \frac{1 - (\lambda_1 \lambda)^{n_2}}{1 - \lambda_1 \lambda} + \beta_2 \lambda^2 \frac{d}{d\lambda^2} \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} + \beta_3 \frac{d^2}{d\lambda^2} \lambda \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} - \frac{d}{d\lambda} \frac{1 - \lambda^{3n_2}}{1 - \lambda^3} - \frac{1 - \lambda^{2n_2}}{1 - \lambda^2} \tag{2.113}
\]

Note that the highest exponent value of \(\lambda_1\) in Eq.(2.111) is \(2n_2\) while it is always less than \(2n_2\) in Eqs.(2.113)- (2.114). It is then possible to show that

\[
\lim_{n_2 \to \infty} \frac{\sigma_i}{\sigma} = \begin{cases} 
\frac{1 - \lambda_1}{c_{1b_1}} & i = 1 \\
0 & i = 2, 3
\end{cases} \tag{2.115}
\]

Using Eq.(2.43), the closed loop poles are the eigenvalues of

\[
\tilde{A} = \begin{bmatrix}
1 & 0 & 0 \\
(1 - \lambda_1)\frac{b_2}{b_1} & \lambda & 1 \\
(1 - \lambda_1)\frac{b_3}{b_1} & 0 & \lambda
\end{bmatrix} \tag{2.116}
\]

It is clear that \(\tilde{A}\) has its eigenvalues at 1, \(\lambda\), \(\lambda\). □

It was conjectured, Clarke, et al.(1987), that the GPC would stabilize an unstable system if the control horizon was chosen at least equal to the number of the system's
unstable poles. This conjecture together with the above result concerning first-order systems would seem to motivate the following claim.

Claim: Given a SISO system with one unstable pole, it is always possible to find a prediction horizon $n_2$ such that the control law given by Eq.(2.41) results in a stable closed-loop system. However, as shown by the following counter-example, this claim is not true.

Counter example: Consider the following system

$$\frac{y}{u} = \frac{2.5z - 2.75}{z^2 - 2.5z + 1}$$

(2.117)

The above plant is non-minimum phase and has open-loop poles at 0.5 and 2. Fig.(2.7) shows that the control law given by Eq.(2.41) cannot stabilize the system whatever the value of $n_2$ is.

Comments
• The source of the instability in the above example is the unstable pole. So, it would be sufficient to track the locus of that pole alone as the prediction horizon varies.

• The closed-loop system can be described by Eq. (2.43) If $\epsilon$ is small enough, the perturbation analysis can be used to calculate the closed-loop eigenvalues efficiently as shown below. If perturbation analysis is used, the calculation of each eigenvalue will be separate from and independent of the calculation of the other eigenvalues.

The following example demonstrates how the perturbation analysis can be used to monitor a particular eigenvalue and pick up the prediction horizon which stabilizes the system.

Example 2: Consider a plant given by

$$\frac{y}{u} = \frac{20z^2 + 4.7z - 3.15}{z^3 + 2.4z^2 - 1.75z + 0.15}$$

(2.118)

The open-loop poles are at -3.0, 0.1, 0.5. The pole at -3 is monitored as the prediction horizon, $n_2$, varies. Fig.(2.8) shows that there are only few choices of $n_2$ which result in a closed loop-pole inside the unit circle. The solid line represents the pole calculated using Lemma 1, while the dotted line represents the exact values. The following observations are worth noting. First, the perturbation analysis is always successful in predicting whether the pole is inside or outside the unit circle. Second, the accuracy of the calculations is acceptable even for small prediction horizons where $\epsilon$ is not necessarily small. Third, the perturbation calculations give conservative values of the closed-loop pole.
2.4 Case of plant-model mismatch

2.4.1 General plant-representation

Let the plant be represented by a non-anticipative dynamic operator that maps the input time-function into the output time-function. The plant is controlled using the control law in Eq.(2.41). If we consider a regulator problem and choose $n_u = 1$ and $\beta = 0$, the control law will be:

$$u(t) = \epsilon[m(\hat{y}(t) - y(t)) + k^T \varepsilon(t)]$$  \hspace{1cm} (2.119)

where,

$$m = \sum_{i=0}^{n_2-1} s_i$$

$\epsilon$ and $k^T$ are given in Eq.(2.41)
Chapter 2. Generalized Predictive Control

Figure 2.9: The control system in case of exact representation

Fig.(2.9) shows the complete system representation, while Fig.(2.10) is an approximation which assumes that the input to the plant is $e_k^T x(t)$. This approximation imposes a severe situation on the controller as it deprives it of the correction term, $em(y(t) - \dot{y}(t))$. On the other hand, Fig.(2.10) enables us to put the control law in a clear observer plus state-feedback form and makes it possible to apply directly the results in Safonov (1980).

The investigation of the stability conditions for the system in Fig.(2.10) is tackled in three steps. First, the stability conditions in case of a direct state feedback are stated. This will be followed by studying the stability conditions for the observer, assuming disconnected state feedback. Finally, the stability of the overall system is concluded using the Separation of Estimation and Control theorem, Safonov (1980). Before going any further, the following definitions are stated. For more details, the reader is referred to Safonov's work.
Definition 1: Graph \((G) \equiv \{ (\omega, \sigma) \in \Omega \times \Sigma \mid \sigma = G\omega \}\). \(\Omega\) and \(\Sigma\) are normed vector-spaces.

Definition 2: \(\zeta(t), \gamma(t) \equiv \frac{1}{\tau} \sum_{t=0}^{T} \zeta(t) \gamma(t)\).

Definition 3: Sector \((F) \equiv \{ (\omega, \sigma) \in \Omega \times \Sigma \mid F(\omega, \sigma, \tau) \leq 0 \forall \tau \in T \}\) where, \(F(\omega, \sigma, \tau) \equiv < F_{11} \sigma + F_{12} \omega, F_{21} \sigma + F_{22} \omega >\). For convenience, we write

\[
F \equiv \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix}
\]

The stability of the system in Fig.(2.10) will be established by the following lemmas.
Lemma 2.6 Let the plant be described by:

\[ x(t + 1) = Ax(t) + bN_i u(t) \]  \hspace{1cm} (2.120)

where \( N_i \) is a general operator which represents the mismatch between the plant and the linear model used to design the controller.

Let the control law,

\[ u(t) = e_k^T x(t) \]  \hspace{1cm} (2.121)

be designed based on a model given by:

\[ x(t + 1) = Ax(t) + bu(t) \]  \hspace{1cm} (2.122)

\[ y(t) = c^T x(t) \]  \hspace{1cm} (2.123)

Assume that \( \exists P_0 > 0 \forall Q_0 \geq 0 \) such that

\[ P_0 = A^T P_0 A + Q_0 \]  \hspace{1cm} (2.124)

Then, as \( n_2 \rightarrow \infty \), \( \text{Graph}(-N_i \circ e_k^T) \) is strictly inside the sector

\[
\begin{bmatrix}
P_{\frac{1}{2}} b & P_{\frac{1}{2}} A - P_{\frac{1}{2}} \\
P_{\frac{1}{2}} b & P_{\frac{1}{2}} A + P_{\frac{1}{2}}
\end{bmatrix}
\]

and the system is closed-loop finite-gain stable, where \( P \) is the unique solution of:

\[ P = (A - e b k^T)^T P (A - e b k^T) + c e^T c \]  \hspace{1cm} (2.125)
Proof:

\[
Graph(-N_i \circ \epsilon_k) = \{(x, u)|u = -N_i \circ \epsilon_k^T x(t)\} \tag{2.126}
\]

Consider,

\[
\Phi = \{(x, u)| < P^k \frac{1}{2} b u + (P^k A - P^k) x, P^k \frac{1}{2} b u + (P^k A + P^k) x > \} \tag{2.127}
\]

Using Definition 2 and Eq.(2.126), Eq.(2.127) can be rewritten as

\[
\Phi = \{(x(t), u(t)) \mid \frac{1}{T} \sum_{t=0}^{T} x(t)^T [(A - b N_i \epsilon_k^T)^T P (A - b N_i \epsilon_k) - P] x(t) < 0\} \tag{2.128}
\]

According to Eq.(2.124) and noting that \( \epsilon \to 0 \) as \( n_2 \to \infty \) then Eq.(2.128) leads to:

\[
\Phi = \{(x(t), u(t)) \mid \frac{1}{T} \sum_{t=0}^{T} x(t)(A^T PA - P)x(t) < 0\} \tag{2.129}
\]
i.e \( Graph(-N_i \circ \epsilon_k^T) \) is strictly inside the sector

\[
\begin{bmatrix}
P^k b & P^k \frac{1}{2} A - P^k \\
P^k b & P^k \frac{1}{2} A + P^k
\end{bmatrix}
\]

Stability follows directly from the sector properties and Lemma (4.1) in Safonov (1980). Eq.(2.125) is based on the analogy between the GPC problem and the equivalent optimal control problem where they both minimize:

\[
J = \sum_{i=0}^{n_2-1} y(t+i+1)^2
= \sum_{i=0}^{n_2-1} \epsilon(t+i+1) \epsilon^T x(t+i+1) + \beta(i) \Delta u^2(t+i) \tag{2.130}
\]

where \( \beta(0) = 0 \) and \( \beta(i) = \infty \) for \( i = 1, \ldots, n_2 - 1 \). □
Lemma 2.7 Let the plant be described by:

\[ x(t+1) = Ax(t) + bu(t) \]  \hspace{1cm} (2.131)

\[ y(t) = N_o c^T x(t) \]  \hspace{1cm} (2.132)

where \( N_o \) is a general operator which represents the mismatch between the plant and the linear model used to implement the observer

\[ \hat{x}(t+1) = A\hat{x}(t) + bu(t) + b\epsilon^N(y(t) - \hat{y}(t)) \]  \hspace{1cm} (2.133)

\[ \hat{y}(t) = c^T \hat{x}(t) \]  \hspace{1cm} (2.134)

Then, as \( n_2 \to \infty \), \( \text{Graph}(-em\Theta c^T N_o) \) is strictly inside the sector

\[
\begin{bmatrix}
    P^{-\frac{1}{2}} & P^{-\frac{1}{2}}A - P^{-\frac{1}{2}} \\
    P^{-\frac{1}{2}}A - P^{-\frac{1}{2}} & P^{-\frac{1}{2}}A + P^{-\frac{1}{2}}
\end{bmatrix}
\]

and the observer is non-divergent, where \( P \) is the unique solution of

\[ P = (A - em\Theta c^T)^T P (A - em\Theta c^T) + Q \]  \hspace{1cm} (2.135)

\[ Q > 0 \]

Proof: The proof is similar to that in Lemma 2.6.

It follows from Lemmas 2.6 and 2.7 that \( \epsilon_k^T \) is bounded and \( \hat{x}(t) \) is a non-divergent estimate of \( x(t) \). Consequently, using the Separation of Estimation and Control theorem, the closed-loop system, shown in Fig.(2.10), is finite gain stable. This shows that by increasing the prediction horizon the robustness of the closed-loop system increases as long as the open-loop plant is stable, Elshafei, et al.(1991).
2.4.2 Linear time-invariant plant-representation

The plant is assumed to be represented by

\[ x(t + 1) = A_{11}x(t) + A_{12}z(t) + b_1 u(t) \]  \hspace{1cm} (2.136)

\[ z(t + 1) = A_{21}x(t) + A_{22}z(t) + b_2 u(t) \]  \hspace{1cm} (2.137)

\[ y(t) = c_1^T x(t) + c_2^T z(t) \]  \hspace{1cm} (2.138)

where,

\( x(t) \) represents the modelled dynamics.

\( z(t) \) represents the unmodelled dynamics.

\( A_{21} = 0 \) as the plant's representation is assumed to be in the Jordan canonical form.

Let the control law be based on the following model

\[ \hat{x}(t + 1) = A_{11}x(t) + b_1 u(t) \]  \hspace{1cm} (2.139)

\[ \hat{y}(t) = c_1^T \hat{x}(t) \]  \hspace{1cm} (2.140)

Using Eqs.(2.119), (2.138) and (2.140), the control law is

\[ u(t) = -cm(c_1^T z(t) + c_2^T \hat{z}(t)) + c(m c_1^T + k^T) \hat{z}(t) \]  \hspace{1cm} (2.141)

Substituting for \( u(t) \) in Eqs.(2.136), (2.137), and (2.139), we get

\[ \begin{bmatrix} x(t + 1) \\ \hat{x}(t + 1) \\ z(t + 1) \end{bmatrix} = \begin{bmatrix} A_{11} & 0 & A_{12} \\ 0 & A_{11} & 0 \\ 0 & 0 & A_{22} \end{bmatrix} \begin{bmatrix} x(t) \\ \hat{x}(t) \\ z(t) \end{bmatrix} + \]
Chapter 2. Generalized Predictive Control

\[
\begin{bmatrix}
-mb_1c_1^T & b_1(mc_1^T + k^T) & -mb_1c_2^T \\
-mb_1c_1^T & b_1(mc_1^T + k^T) & -mb_1c_2^T \\
-mb_2c_2^T & b_2(mc_2^T + k^T) & -mb_2c_2^T
\end{bmatrix}
\begin{bmatrix}
x(t) \\
\dot{x}(t) \\
z(t)
\end{bmatrix}
\]

(2.142)

Comments:

- The formulation of Eq.(2.142) shows that the problem can be dealt with using the perturbation analysis.
- \(\epsilon \to 0\) as \(n_2 \to \infty\), i.e the control law turns to be a mean level controller.
- Using the same approach used in section (2.4.1), the GPC can be interpreted as an observer plus state-feedback scheme.

The above analysis leads to the following theorem.

**Theorem 2.4** Let the plant be described by Eqs.(2.136) - (2.138) and the control law, Eq.(2.141), be based on the model given by Eqs.(2.139) - (2.140). Assume the following conditions are satisfied:

- The plant is open-loop stable.
- \(\beta = 0\).
- \(\epsilon\) is sufficiently small.

Then, the closed-loop system described by Eq.(2.142) will have its closed-loop poles given by either Lemma 2.1, Lemma 2.2, or Lemma 2.3 as appropriate. Furthermore, if \(n_2 \to \infty\), the closed-loop poles will be at the open loop poles of the plant, Eqs.(2.136)-(2.138), and its approximate model, Eq.(2.139).

**Proof:** The proof is immediate from Eq.(2.142) and the subsequent comments. □
2.5 Conclusions

A generalized predictive controller has been derived based on state-space modelling. The case of one-step control horizon has been analyzed in detail. First, it has been shown that the GPC can stabilize any first-order system. Then, it has been shown that the GPC problem is equivalent to a perturbation problem. For the case of small perturbations, approximate formulas have been used to calculate the closed-loop poles. This approach has enabled us to relate the prediction horizon to the closed-loop poles so that specific poles would have been calculated without solving the whole eigenvalue problem. Consequently, prediction horizons and control-weighting factors which insure closed-loop stability have been picked up correctly. For the case of general perturbations, an upper bound on the perturbation norm has been derived to insure closed-loop stability. The GPC has always been able to satisfy the robustness bound by proper tuning of the prediction-horizon and control-weighting factor. In the next chapter, this robustness bound will be used to study the stability of the adaptive Laguerre-filter based GPC.

Both the plant-model match and plant-model mismatch cases have been studied in this chapter. The main conclusion has been that the GPC can achieve arbitrary robustness by increasing the prediction horizon. This result has been the motivation to implement an adaptive Laguerre-filter based GPC as shown in Chapter 3.
Chapter 3

Adaptive Generalized Predictive Control

3.1 Introduction

From the modelling point of view, it is possible to classify predictive control schemes into two main categories. The first uses a structured model representation, e.g. ARMAX or CARIMA models. The second approach uses an unstructured model representation, e.g. an impulse-response based model or an orthonormal-series based model.

In De Keyser, et al.(1981), an ARMAX model is used to derive the extended prediction self adaptive controller, (EPSAC). The EPSAC needs $n$-self tuning $k$-step-ahead predictors in parallel, where $n$ is the model order. Hence, the Diophantine equation has to be solved for each predictor. In Ydstie (1984), an ARMA model is used to derive the extended horizon predictive controller, (EHC), where the receding horizon idea is introduced. An ARMA model is also used in Irving, et al.(1986) to derive a generalized predictive controller, (GPC), using input and output models. Difficulties in tuning GPC while preserving stability are shown in Wertz, et al.(1987). A CARIMA model is used to derive another version of the GPC in Clarke, et al.(1987) where an elegant recursive solution of the Diophantine is introduced. Still, the theoretical properties of the adaptive GPC need further study.

In model algorithmic control, (MAC), the impulse response coefficients are used to model actual plants, Richalet, et al.(1978). The impulse-response coefficients are used to derive the step-response coefficients which are used in the dynamic matrix, (DMC),
Cutler (1980). The use of the impulse-response coefficients is based on the assumption that the impulse response of the true plant converges to zero. This assumption means that the MAC and the DMC are applicable only to stable systems. Furthermore, if the open-loop system is poorly damped, the number of coefficients required to represent the system effectively is large.

A new predictive controller which uses an unstructured model is proposed in Zervos and Dumont (1988). The new controller uses Laguerre functions for modelling. There are some advantages in using Laguerre-functions based models. First, any stable system can be modelled without the need for accurate information on the true plant order and time delay. Second, the Laguerre functions can represent signals which exhibit long time-delays because of their similarity to Pade approximants. Third, in the presence of unmodelled dynamics and colored noise, contrary to the ARMAX model, a Laguerre-function based model results in an unbiased estimate of the nominal plant when the input is white, Dumont, et al. (1991). Fourth, Laguerre functions can accurately model nonlinear plants, Lee (1968).

In Zervos (1988) the theoretical properties of a SISO single-step ahead predictive controller are studied. This controller has a one-step control horizon and zero weight on the control actions. In this chapter a generalized predictive controller based on Laguerre-function modelling is derived. Allowing multi-step predictions and a non-zero weight on the changes of the control signal help avoid excessive control actions, too active control signal and sluggish output response. Both SISO and MIMO systems are analyzed.

3.2 Modelling and identification

The following definitions establish the mathematical bases for the Laguerre-filter based models which are the main topic in this chapter.
Chapter 3. Adaptive Generalized Predictive Control

Definition 3.1 The signal \( y(t) \) is said to be in the Lebesgue space \( L_2[0, \infty) \) iff

\[
\int_0^\infty y^2(t)dt < \infty
\]

(3.143)

Definition 3.2 The sequence \( f_i(t), i = 1, 2, \ldots, n \) is orthonormal iff

\[
\int_0^\infty f_i(t)f_j(t)dt = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}
\]

(3.144)

Definition 3.3 The orthonormal set \( \{f_i(t)\} \) with \( f_i(t) \in L_2[a, b] \) is called complete or closed if either of the following statements is true:

1. There exists no function \( x(t) \in L_2[a, b] \) such that

\[
\int_a^b x(t)f_i(t)dt = 0, \quad i = 1, 2, \ldots
\]

(3.145)

2. For any piecewise continuous function \( y(t) \in L_2[a, b] \) and \( \epsilon > 0 \), however small, there exists an integer \( N \) and a polynomial

\[
\sum_{i=1}^{N} c_i f_i(t)
\]

(3.146)

such that

\[
\int_a^b |y(t) - \sum_{i=1}^{N} c_i f_i(t)|^2dt < \epsilon
\]

(3.147)

The study in most of this chapter is concerned with signals in the Lebesgue space, \( L_2[0, \infty) \). The Laguerre functions, a complete orthonormal set, are used for signal modelling. In the time domain, the Laguerre functions are described by

\[
l_i(t) = \sqrt{2p} \frac{e^{pt}}{(i - 1)!} \frac{d^{i-1}}{dt^{i-1}}[t^{i-1}e^{-pt}]
\]

(3.148)

where \( i \) is the order of the function and \( p \) is the time-scale. Since \( \{l_i(t)\} \) forms a complete orthonormal set, the classical Riesz-Fisher theorem is applicable, Zemanian (1968).
Theorem 3.1 Let \( \{l_i(t)\} \) be a complete orthonormal set as specified above, and let \( \{c_i\} \) be a sequence of real numbers such that \( \sum_{i=1}^{\infty} |c_i|^2 \) converges. Then, there exists a unique \( y(t) \in L_2[0, \infty) \) such that

\[
\begin{align*}
    c_i &= \int_{0}^{\infty} y(t)l_i(t)dt \\
    y(t) &= \sum_{i=1}^{\infty} c_i l_i(t)
\end{align*}
\] (3.149) (3.150)

The above theorem means that the impulse response of any stable system can be represented with arbitrary accuracy by a series of Laguerre functions.

In the Laplace transform domain, the Laguerre filters are given by

\[
L_i(s) = \sqrt{2p} \frac{(s - p)^{i-1}}{(s + p)^i}
\] (3.151)

The Laguerre filters can be implemented by a simple ladder network which has the following SISO discrete-time state-space representation, Zervos(1988).

\[
\begin{align*}
    \dot{\mathbf{x}}(t+1) &= A\mathbf{x}(t) + b\mathbf{u}(t) \\
    y(t) &= \mathbf{c}^T \mathbf{x}(t)
\end{align*}
\] (3.152) (3.153)

where

\[
A = \begin{bmatrix}
    \tau_1 & 0 & \ldots & 0 \\
    -\frac{\tau_2}{T_s} & \tau_1 & \ldots & 0 \\
    \vdots & \ddots & \ddots & \vdots \\
    -\frac{(1)^{N-1} \tau_{N-2}(\tau_1 \tau_2 + \tau_3)}{T_s^{N-1}} & \ldots & -\frac{\tau_{N-2} \tau_3}{T_s} & \tau_1
\end{bmatrix}
\] (3.154)

\[
\mathbf{b}^T = \begin{bmatrix}
    \tau_4 & -\frac{\tau_2}{T_s} \tau_4 & \ldots & -\frac{\tau_{N-2}}{T_s} \tau_4
\end{bmatrix}
\] (3.155)

\[
\mathbf{c}^T = \begin{bmatrix}
    c_1 & c_2 & \ldots & c_N
\end{bmatrix}
\] (3.156)

The constants \( \tau_1, \tau_2, \tau_3, \) and \( \tau_4 \) are given in terms of the sampling period \( T_s \) and the Laguerre-functions' time-scale \( p \) as

\[
\tau_1 = e^{-pT_s}
\] (3.157)
\[ \tau_2 = T_s + \frac{2}{p}(e^{-pT_s} - 1) \]  
\[ \tau_3 = -T_s e^{pT_s} - \frac{2}{p}(e^{pT_s} - 1) \]  
\[ \tau_4 = \sqrt{2p \left(1 - e^{pT_s}\right)} \]  

The Laguerre spectrum gains, \( c_i, i = 1, \ldots, N \) are identified on-line using recursive estimation.

### 3.3 Analysis of single-input, single-output systems

An adaptive version of the control law derived in section (2.2.1) is applied to the system described by Eqs.(3.152)-(3.153). Assume a one-step control horizon, Eq.(2.27) can be written as

\[ (\hat{s}_1^T \hat{s}_1 + \beta)\Delta u(t) = \hat{s}_1^T(y_c - l - \hat{s}_1 u(t - 1)) \]  

where

\[ \hat{s}_1^T = \begin{bmatrix} \hat{s}_0, \hat{s}_1, \ldots, \hat{s}_{n_2-1} \end{bmatrix} \]  
\[ \hat{s}_i = \sum_{j=0}^{i-1} \hat{x}^T A^j \hat{b} \]  
\[ \hat{s}_1^T l = \sum_{i=0}^{n_2-1} \hat{s}_i [y(t) - \hat{y}(t)] + \sum_{i=0}^{n_2-1} \hat{s}_i \hat{x}^T A^i \xi(t) \]  

Hence,

\[ u(t) = \hat{\xi}\{\hat{s}_1^T y_c + \hat{m}[\hat{y}(t) - y(t)] - 1 \hat{x}^T \xi(t) + \beta u(t - 1)\} \]  

where

\[ \hat{\xi} = \frac{1}{\hat{s}_1^T \hat{s}_1 + \beta} \]
\[
\hat{n} = \sum_{i=0}^{n_2-1} \hat{s}_i \quad (3.167)
\]
\[
\hat{k}^T = \sum_{i=0}^{n_2-1} \hat{s}_i \hat{e}_i^T A_i+1 \quad (3.168)
\]

Using Eqs. (3.152) and (3.165), the closed-loop system is
\[
\begin{bmatrix}
\hat{z}(t+1) \\
\hat{u}(t+1)
\end{bmatrix} =
\begin{bmatrix}
A & b \\
-\hat{e}_k A & -\hat{e}_k^T b + \hat{e} \beta
\end{bmatrix}
\begin{bmatrix}
\hat{z}(t) \\
\hat{u}(t)
\end{bmatrix} +
\begin{bmatrix}
0 \\
\hat{e}
\end{bmatrix}
\begin{bmatrix}
\hat{m}(\hat{y}(t+1) - y(t+1)) + \hat{s}_i^T y_e
\end{bmatrix} \quad (3.169)
\]

### 3.3.1 Stability and convergence analyses in case of plant-model match

The following theorem, Payne (1987), is essential to prove the stability of our adaptive scheme.

**Theorem 3.2** Consider the time-varying difference equation
\[
\hat{z}(t+1) = F(t)\hat{z}(t) + \psi(t) \quad (3.170)
\]
where \(\hat{z}(t)\) and \(\psi(t)\) are real vectors of finite dimension. Suppose that the sequence of matrices \(\{F(t)\}\) and \(\hat{z}(0) = z_0\) are bounded and that the free system
\[
\hat{z}(t+1) = F(t)\hat{z}(t), \quad t \geq 0 \quad (3.171)
\]
is exponentially stable. Furthermore, suppose that there exist sequences of non-negative numbers \(\{\gamma(t)\}\) and \(\{\delta(t)\}\) and an integer \(N \geq 0\) such that
\[
||\psi(t)|| \leq \gamma(t) \sum_{i=0}^{N} ||\hat{z}(t-i)|| + \delta(t) \quad (3.172)
\]
Under these conditions, if \(\{\gamma(t)\}\) converges to zero and \(\{\delta(t)\}\) is bounded, then \(\{\hat{z}(t)\}\) and \(\{\psi(t)\}\) are bounded. If in addition, \(\{\delta(t)\}\) converges to zero, then \(\{\hat{z}\}\) and \(\{\psi(t)\}\) also converge to zero.
Using Theorem 3.2, the stability of the closed-loop system described by Eq.(3.169) is concluded.

**Theorem 3.3** Assume the plant is described by Eq.(3.152) and

\[ y(t) = \xi^T a z(t) \]  

where,

\[ z(t) = \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \]  

Provided that a least-squares algorithm is used to find \( \xi^T a \) such that \( \dim(\xi^T a) = \dim(\xi^T a) \), then the closed-loop system described by Eq.(3.169) is stable, i.e. \( \{ y(t) \} \) and \( \{ u(t) \} \) are bounded for all \( t \).

**Proof**: Using the recursive-least squares estimator, the parameters’ estimates have the following properties, Goodwin and Sin (1984)

1. \( \{ \hat{\xi}_a(t) \} \) is bounded
2. \( \lim_{t \to \infty} |\hat{\xi}_a(t) - \hat{\xi}_a(t - 1)| = 0 \)
3. There exist nonnegative sequences \( \{ \zeta(t) \} \) and \( \{ \psi(t) \} \) that converge to zero such that

\[ |\hat{y}(t) - y(t)| \leq \zeta(t)||z(t)|| + \psi(t) \]  

Consider the closed-loop system given by Eq.(3.169). Choose

\[ F(t) = \begin{bmatrix} A & \hat{b} \\ -\hat{e}_k^T & -\hat{e}_k^T \hat{b} + \hat{e}_p \end{bmatrix} \]

\[ \psi(t) = \begin{bmatrix} \hat{Q} \\ \hat{e} \end{bmatrix} [\hat{m}(\hat{y}(t + 1) - y(t + 1)) + \hat{s}^T u_c] \]
Chapter 3. Adaptive Generalized Predictive Control

Theorem (2.2) shows that there always exist $n_2$ and $\beta$ such that the free system

$$z(t + 1) = F(t)z(t)$$

is stable. Using Eqs. (3.152) and (3.175) leads to

$$||\psi(t)|| \leq \dot{\epsilon}||\hat{m}||\xi||z(t + 1)|| + \psi(t) + ||s_1^T y_e||$$

$$\leq ||\Gamma^{-1}(t)||\{\gamma(t)||z(t)|| + \delta(t)\}$$

where

$$\Gamma(t) = I - \left[ \begin{array}{c} 0 \\ \dot{\epsilon} \end{array} \right] \dot{\hat{m}}(z_1^T - c_1^T)$$

$$\gamma(t) = \dot{\epsilon}||\hat{m}||\xi(t)||F(t)||$$

$$\delta(t) = \dot{\epsilon}||\hat{m}||\psi(t) + ||s_1^T y_e||$$

and $I$ is the identity matrix. The above properties of the recursive least-squares estimator clearly mean that $\gamma(t)$ converges to zero and $\delta(t)$ is bounded. Theorem 3.2 leads to the conclusion that the system described by Eq.(3.169) is stable. □

**Example**  Consider the 8th-order system given by

$$G(s) = \frac{1}{(s + 1)^8}$$

The system is sampled using a 1.0 sec. sampling period. The open-loop response, Fig.(3.11), is highly overdamped with a very slow initial response. There is an implicit dead time of roughly 4 secs. The model uses 8 Laguerre filters. The filters' gains are estimated using RLS with an initial covariance matrix $P_0 = 100 \times I$ where $I$ is the identity matrix. Fig.(3.12) shows the convergence of the Laguerre gains which agrees well with the assumptions of Theorem 3.3. In Fig.(3.13), the frequency response of the identified model (solid line) is compared to that of the true system (dotted line). Up to
the Nyquist frequency, i.e. $\pi$ rad/sec., the model achieves an almost perfect match with the plant. The GPC is tuned such that $n_2 = 8$, $n_u = 1$, and $\beta = 0.5$. The closed-loop response is shown in Fig. (3.11). It is clear that the system achieves a perfect tracking of the reference signal. The rise time of the closed-loop response is faster than the open-loop. The overshoot is acceptable.

### 3.3.2 Robustness analysis

As mentioned before, any stable plant can be represented by an infinite Laguerre series. However, a finite but large truncated Laguerre series can model a stable plant with any specified accuracy. In practice, it is desirable to use a low order model. So, the robustness problem should be studied. The following theorem addresses this issue.

**Theorem 3.4** Let the plant be represented by a large truncated Laguerre series.

\[
\begin{bmatrix}
x_1(t+1) \\
x_2(t+1)
\end{bmatrix} = \begin{bmatrix} A_1 & 0 \\ A_{21} & A_2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u(t) \quad (3.181)
\]

\[
y(t) = c^T \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad (3.182)
\]

Assume that the above plant is modelled using a low order model given by

\[
x_1(t+1) = A_1 x_1(t) + b_1 u(t) \quad (3.183)
\]

\[
\hat{y} = \begin{bmatrix} c_1^T : 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_2(t) \end{bmatrix} \quad (3.184)
\]

Assume the system is controlled by the control law given by Eq. (3.165), using $\hat{c}_1^T$. Then, the closed-loop system is stable if

\[
\epsilon \| \hat{c}^T - c^T \| < \frac{1 - \|A\|}{\|m\| \|b\|} \quad (3.185)
\]
Figure 3.11: a- The open-loop response. b- The control signal. c- The closed-loop response.
Figure 3.12: The estimates of the Laguerre-filters' gains
Figure 3.13: A comparison between the frequency responses of the true system (dotted line) and the identified model (solid line).
where

\[
\Lambda = \begin{bmatrix}
A_1 - \epsilon b_1 k^T & 0 \\
A_{21} - \epsilon b_2 k^T & A_2
\end{bmatrix}
\]

\[
b = \begin{bmatrix}
\hat{b}_1^T \\
\hat{b}_2^T
\end{bmatrix}^T
\]

\[
\hat{\xi}^T = \begin{bmatrix}
\hat{\xi}_1^T & 0 & \ldots & 0
\end{bmatrix}
\]

\(\hat{\xi}_1^T\) is obtained using RLS

**Proof**: Assume for simplicity that \(\beta = 0\) and \(y_c = 0\). let

\[
X(t) = \begin{bmatrix}
x_1^T \\
x_2^T
\end{bmatrix}^T
\]

Using Eqs.(3.165) and (3.184), the closed-loop system can be described by

\[
X(t+1) = \Lambda X(t) + \epsilon m b [\hat{y}(t) - y(t)]
\]

Using Eqs.(3.184), (3.186) and (3.187), it is easy to show that

\[
\|X(t+1)\| < \|\Lambda\| + \epsilon \|m\| \|\hat{b}\| \|\xi^T - \xi^T\| \|X(t)\|
\]

Eq.(3.188) shows that if the condition given by Eq.(3.185) is satisfied, then the closed-loop system given by Eq.(3.187) is stable. \(\Box\)

**Example**: Consider the following system, (Rohrs et al. 1982),

\[
G(s) = \frac{2}{s + 1} \frac{229}{s^2 + 30s + 229}
\]

To demonstrate the robustness of the Laguerre filters based GPC, assume that only one filter is used to model the above system. The time scale of the Laguerre filter is chosen as 1.0sec. The reference signal has the form

\[y_c = \sin(\omega t)\]
As a start, assume that $w = 1.0 \text{rad./sec.}$ Choose the sampling interval, $T_s = 0.1 \text{sec.}$ The controller is tuned such that $\beta = 0$ and $n_2 = 13$. Fig(3.14-a) shows that the output of the system is stable and tracks the reference. To study the effect of fast sampling, $T_s$ is reduced to 0.01 sec. If the tuning parameters are kept unchanged, the system becomes unstable. As theorem 3.4 predicts, by increasing $n_2$ to 40 and keeping everything else the same, the closed-loop system regains its stability, see Fig.(3.14-b).

It is also interesting to check the performance of the closed-loop system at $w = 16.1 \text{rad./sec.}$ At this frequency a first order model cannot match the actual system, Åström (1983). It is also shown in Zervos (1988) that the minimum number of Laguerre filters required to model the system at this frequency is 2. Here, the number of filters is kept equal to 1. By choosing $T_s = 0.03$, $\beta = 0.19$ and $n_2 = 13$, Fig. (3.14-c) shows that the closed-loop system is stable. Because of the poor representation of the system, the output tracking of the reference signal is poor. However, this example demonstrates that the GPC robustness can be improved be decreasing $\epsilon$. This is achieved by increasing $n_2$ or $\beta$.

3.3.3 Further analyses

1. Extension to DMC: Assume that the plant can be represented by a moving average model or a truncated series of the impulse-response coefficients (as in Dynamic Matrix Control, Cutler (1980)). Then

$$y(t) = b_1 u(t - 1) + b_2 u(t - 2) + \ldots + b_n u(t - n)$$

(3.189)
Figure 3.14: Robustness of the Laguerre-filters based GPC
A state-space representation similar to Eqs.(3.152) and (3.153) can be derived where

\[ A = \begin{bmatrix} 0 & \ldots & 0 & 0 \\ \vdots \\ I & 0 \\ 0 \end{bmatrix}, \]

\[ b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}^T, \]

\[ \xi^T = \begin{bmatrix} b_1 & b_2 & \ldots & b_n \end{bmatrix} \]

So, an adaptive system which uses the model given by Eq.(3.189), a recursive least-squares algorithm to find \( \xi^T \), and the GPC given by Eq.(3.152) is stable. The proof is the same as in Theorem 3.2.

2. More on robustness: Choosing \( \beta = 0 \) and a single-step prediction horizon of length \( d \), Eq.(3.165) becomes

\[
\begin{align*}
    u(t) &= \frac{1}{\hat{s}^2_{d-1}} \{s_{d-1} y_c + \hat{s}_{d-1} [\hat{y}(t) - y(t)] - \hat{s}_{d-1} \xi^T A^d \xi(t) \} \\
    &= \frac{1}{\hat{s}^2_{d-1}} \{y_c - y(t) - \xi^T (A^d - I) \xi(t) \}
\end{align*}
\]

(3.190)

In Zervos (1988), Eq.(3.190) is derived by forcing the constraint

\[ \hat{y}(t + d) = y_c \]

(3.191)

The above derivation is the state-space form of the single-step predictive controller which is derived based on an ARMA model in Goodwin and Sin (1984). It is clear that \( d \) should be greater than the dead-time and the non-minimum phase response of the plant to insure the stability of the closed-loop system.

To show the effect of plant-model mismatch, Dumont (1991), let the system be given by Eqs.(3.152)-(3.153) and the model given by

\[ y_m(t) = \xi_m^T \xi(t) \]

(3.192)
where,
\[ \mathbf{c}_m^T = \begin{bmatrix} c_{m1} & \ldots & c_{mN} & 0 & \ldots & 0 \end{bmatrix} \]
\[ \text{dim}(\mathbf{c}_m^T) = \text{dim}(\mathbf{c}^T) \]

The control law, Eq.(3.190), can be rewritten as

\[ u(t) = -k_c \mathbf{x}(t) + N_c y_c \]  
(3.193)

\[ k_c = \frac{1}{s_{d-1}}[(\mathbf{c}^T - \mathbf{c}_m^T) + \mathbf{c}_m^T A^d] \]  
(3.194)

\[ N_c = \frac{1}{s_{d-1}} \]  
(3.195)

\[ s_{d-1} = \sum_{i=0}^{d-1} \mathbf{c}_m A^i b \]  
(3.196)

If \( d \to \infty \), then

\[ u(t) = -\frac{1}{s_{d-1}}(\mathbf{c}^T - \mathbf{c}_m^T)\mathbf{x}(t) + N_c y_c \]  
(3.197)

It is easy to show that

\[ y - y_m = \Delta G u(t) \]
[\( \mathbf{c}^T - \mathbf{c}_m^T)(I - Aq^{-1})^{-1}bq^{-1}u(t) \)]  
(3.198)

The closed-loop system described by Eqs.(3.197) and (3.198) is stable if \( \Delta G \) is strictly positive real (SPR). The above discussion shows a fundamental difference between the control law which is based on a single-step prediction horizon, Eq.(3.193) and that which is based on a multi-step prediction horizon, Eq(3.165). The former may lead to instability if the prediction horizon increases while the latter becomes more robust as the prediction horizon increases.
3. Nonlinear systems: Consider the discrete-time nonlinear dynamic system

\[ x(t + 1) = A.x(t) + B.u(t) \]  
\[ y(t) = C.x(t) \]

where \( A, B \) and \( C \) are nonanticipative, differential, dynamic, nonlinear operators.

Assume that the nominal plant is represented by Eq.(3.152) and controlled using

\[ u(t) = - \sum_{i=0}^{n-1} s_i x^T A^i x(t) = -k^T x(t) \]

The stability of the closed-loop system is assessed by the following lemma, Savonof (1980).

**Lemma 3.1** Let the constant matrices \( P \in \mathbb{R}^{n \times n} \) and \( S \in \mathbb{R}^{n \times n} \) be symmetric, positive definite solutions of the discrete Lyapunov equation

\[ P = (A - bk^T)^T P (A - bk^T) + S \]

If uniformly for all \((x(t), u(t))\)

\[ \text{Graph}(P^{0.5}[A - bk^T + \Delta A + \Delta B(-k^T)]) \]

is strictly inside \((0, P^{0.5})\), then, the system described by Eq.(3.199) with state feedback given by Eq.(3.201) is closed-loop stable.

The above lemma gives a sufficient stability condition for the actual nonlinear state-feedback system. However, \( k^T \) is chosen such that the linearized model is stable, not the actual plant. It has been shown that there always exists a prediction horizon \( n_2 \) such that the control law given by Eq.(3.201) stabilizes the model given
by Eq. (3.152). So, Eq. (3.202) can always be satisfied. On the other hand, Lemma (3.1) is satisfied if the deviation of the model from the actual plant, \((\Delta A, \Delta B)\) is sufficiently small. Since the Laguerre functions form a complete orthonormal set, \((\Delta A, \Delta B)\) can be made arbitrarily small so that the stability of the closed-loop system is insured.

**Example**: Consider the problem of controlling the temperature of an exothermic catalytic reaction. Assuming a constant flow, the system equations are

\[
\begin{align*}
\frac{dC_A}{dt} &= \frac{F_0}{V_f}(C_{A0} - C_A) - kC_A \\
\frac{dT}{dt} &= \frac{K_2}{K_1}(T_0 - T) + \frac{V_f kC_A}{K_1} \Delta H - \frac{Q}{k_1} \\
K_1 &= V_s c_{ps} \rho_f c_{pf} + \rho_f c_{pf} V_f \\
K_2 &= F_0 \rho_f c_{pf} \\
k &= 5 \times 10^8 e^{-\frac{13000}{T}} \\
-\Delta H &= -35000 - 200 \times T
\end{align*}
\]

where,

- \(F_0\) : volumetric flow rate = \(1 \times 10^{-4} m^3/\text{sec}\).
- \(V\) : total volume = 1.0 \(m^3\).
- \(V_s\) : volume of catalyst = 0.7 \(m^3\).
- \(V_f\) : volume of fluid = 0.3 \(m^3\).
- \(c_{pf}\) : heat capacity of the fluid = 1500 \(J/kg^\circ K\).
- \(\rho_f\) : density of fluid = 10 \(kg/m^3\).
- \(c_{ps}\) : heat capacity of catalyst = 6000 \(J/kg^\circ K\).
- \(\rho_s\) : density of catalyst = 2500 \(kg/m^3\).
- \(C_{A0}\) : initial concentration = 10 \(moles/m^3\).
Chapter 3. Adaptive Generalized Predictive Control

$T_0$: initial temperature $= 400^\circ K$.

The above system is nonlinear due to the term $kC_A$. At low temperatures there is practically no reaction, e.g. at $400^\circ K, k = 3.8 \times 10^{-6}$, while at $500^\circ K, k = 2.5 \times 10^{-3}$. The controlled variable, $y(t)$, is chosen to be the temperature, $T$, and the manipulated variable, $u_c(t)$, is $Q$. Figure (3.15-a) shows the open-loop response to the series of step changes shown in Fig. (3.15-b). The open-loop response suggests that the plant can be modelled using Eq.(3.152) and

$$\Delta y(t) = e^T x(t)$$

(3.209)

The j-step-ahead predictor is

$$y(t + j) = y(t) + \sum_{i=1}^{j} \Delta y(t + i)$$

$$= y(t) + \sum_{i=1}^{j} e^T x(t + i)$$

(3.210)

Having a set of output predictions, a performance index similar to Eq.(2.26) is minimized. The resulting control law is

$$u(t) = (s^T s + \beta I)^{-1} s^T (y_c - f)$$

(3.211)

where,

$$f = \begin{bmatrix} y(t) \\ y(t) \\ \vdots \\ y(t) \end{bmatrix} + \begin{bmatrix} e^T A \\ \sum_{i=1}^{2} e^T A_i \\ \vdots \\ \sum_{i=1}^{d} e^T A_i \end{bmatrix}$$

(3.212)

The control signal and the output are shown in Figs.(3.15-c) and (3.15-d) respectively.
Figure 3.15: Control of a nonlinear system. a- The open-loop response. b- The set-point. c- The manipulated variable. c- The closed-loop response.
4. Extension to unstable systems: Use of the Laguerre filters is limited to $L_2$-stable systems. It is possible to model unstable systems as follows

$$\hat{y}(t + 1) = \sum_{i=1}^{n} \tilde{a}_{i,1} y(t - i) + \xi^T \tilde{x}(t + 1) \tag{3.213}$$

The above model can be interpreted as follows

- The term $\sum_{i=1}^{n} \tilde{a}_{1,i} y(t - i)$ can be used to model the unstable dynamics or the dominant dynamics of a plant.
- The term $\xi^T \tilde{x}(t)$ can be used to model the stable dynamics, the time-delay as well as the high order dynamics.

Using Eq. (3.213), a j-step ahead prediction of the output gives

$$\hat{y}(t + j) = \sum_{i=1}^{n} \tilde{a}_{i,j} y(t - i + 1) + \sum_{i=1}^{j-1} \tilde{a}_{1,j-i} \xi^T \tilde{x}(t + i) + \xi^T \tilde{x}(t + j) \tag{3.214}$$

where

$$\tilde{a}_{1,j} = \tilde{a}_{1,j-1} \tilde{a}_{1,1} + \tilde{a}_{2,j-1}$$

$$\vdots$$

$$\tilde{a}_{n-1,j} = \tilde{a}_{1,j-1} \tilde{a}_{n-1,1} + \tilde{a}_{n,j-1}$$

$$\tilde{a}_{n,j} = \tilde{a}_{1,j-1} \tilde{a}_{n,1} \tag{3.215}$$

**Example** Consider the unstable system given by

$$G(z) = \frac{0.014z^2 + 0.04z + 0.007}{(z - 1.28)(z^2 - 0.737z + 0.165)}$$

The future output is predicted based on the following model

$$\hat{y}(t + 1) = ay(t) + \xi^T \tilde{x}(t + 1)$$

$$\tilde{x}(t + 1) = A\tilde{x}(t) + bu(t)$$
The model is chosen to have 4 Laguerre filters. The GPC is tuned such that $n_2 = 15$, $n_u = 2$, and $\beta = 1$. Fig.(3.16) shows that the system is successfully stabilized with a zero steady-state error.

5. Stochastic systems : Consider a stochastic system given by

$$A(q^{-1})y(t) = B(q^{-1})u(t) + C(q^{-1})\zeta(t)$$  \hspace{1cm} (3.216)$$

where $\zeta(t)$ is a white-noise signal. The control law given by Eq.(3.165) is valid in case of $C(q^{-1}) = 0$, i.e. a deterministic system, or $C(q^{-1}) = 1$, i.e. a white noise input. In case of colored noise, a filtered model is recommended, Lam (1989). The optimal choice of the filter polynomial, $T(q^{-1})$, is $T(q^{-1}) = C(q^{-1})$, Åström and Wittenmark, (1984). In the GPC literature, e.g Clarke, et al.(1987), $T(q^{-1})$ is one of the design parameters. Design guidelines for choosing $T(q^{-1})$ are given in Mohtadi, (1989). The example which is given below shows the performance of the LAG-GPC in a noisy environment.

Example Consider the stochastic process

$$y(t) = 0.9y(t - 1) + 3u(t - 2) + \zeta(t) - 0.3\zeta(t - 1)$$  \hspace{1cm} (3.217)$$

where $\zeta(t)$ is the noise signal with zero mean and 1.0 standard deviation. The reference signal is set to 1.0 for 100 samples, then, it is set to zero. The Laguerre-filter model is implemented using 6 filters and 1.0 second time scale. The controller is tuned such that $n_2 = 6$, $n_u = 1$, and $\beta = 0.1$. The controller performance based on a deterministic design, i.e. using $T(q^{-1}) = 1$, is compared with that based on an optimal filter design, i.e $T(q^{-1}) = 1 - 0.3q^{-1}$. Note that the filtered input, $u_f$, and output, $y_f$, measurements are used to estimate the gains of the Laguerre-filters,
Figure 3.16: Control of an open-loop unstable system
where

\[ y_f(t) = \frac{y(t)}{T(q^{-1})} \]  \hspace{1cm} (3.218)

\[ u_f(t) = \frac{u(t)}{T(q^{-1})} \]  \hspace{1cm} (3.219)

The filtered model is used to derive the control signal, \( u_f(t) \), which is fed to Eq.(3.219) to calculate the actual control signal, \( u(t) \). In the table given below, these two designs are compared with the open-loop performance. The figures given in the table are based on the last 256 samples in Fig. (3.17). Note that the output variance achieved using a minimum-variance controller is 1.36. The main conclusion is that a deterministic design does not drive the system to instability and it gives a reasonable output variance.

<table>
<thead>
<tr>
<th></th>
<th>Average</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>open-loop</td>
<td>0.080</td>
<td>2.23</td>
</tr>
<tr>
<td>Optimal filter design</td>
<td>0.004</td>
<td>1.69</td>
</tr>
<tr>
<td>Deterministic design</td>
<td>0.043</td>
<td>1.74</td>
</tr>
</tbody>
</table>

Statistics of the process output.

3.4 Extensions to multivariable systems

A GPC based on a Laguerre-functions model has been analyzed in section 3.3 for SISO plants. In this section, the extensions to SIMO and MIMO systems are considered. If a Laguerre-function based model is used to derive a multivariable GPC, there will be no need to define the delay matrix of the system. This is
considered a major advantage as the delay matrix for a MIMO system has no unique structure.

3.4.1 Single-Input, Multi-Output systems

Let the system be

\[
\begin{bmatrix}
\tilde{x}_1(t+1) \\
\tilde{x}_2(t+1) \\
\vdots \\
\tilde{x}_m(t+1)
\end{bmatrix} =
\begin{bmatrix}
A_1 & & \\
& A_2 & \\
& & \ddots \\
& & & A_m
\end{bmatrix}
\begin{bmatrix}
\tilde{x}_1(t) \\
\tilde{x}_2(t) \\
\vdots \\
\tilde{x}_m(t)
\end{bmatrix} +
\begin{bmatrix}
b_1 \\
b_2 \\
\vdots \\
b_m
\end{bmatrix} u(t)
\]

\[
= AX(t) + bu(t)
\]  

(3.219)
The above representation means that we have \( m \) Laguerre blocks where the \( i \)th block has \( N_i \) filters. The states of each block are fed to a summing network to yield the \( m \) outputs. Consider a one-step control horizon. Assume that the prediction horizon of the \( i \)th output, \( y_i(t) \), is \( d_i \). Using the same approach used in section 3.3, it is easy to show that

\[
y_i(t + j) = y_i - \hat{y}_i(t) + \sum_{k=1}^{m} c_{i1k}^T (A_{i1}^{j-1} + A_{i2}^{j-2} + \ldots + I)b_k + \sigma_{ij}u(t-1) + \sigma_{ij}\Delta u(t), \quad i = 1, 2, \ldots m
\]  

(3.222)

where

\[
\sigma_{ij} = \sum_{k=1}^{m} c_{iik}^T (A_{ik}^{j-1} + A_{ik}^{j-2} + \ldots + I)b_k
\]  

(3.223)

Hence,

\[
Y_i = y_i - \hat{y}_i + K_i X(t) + \sigma_i u(t-1) + \sigma_i \Delta u(t)
\]  

(3.224)

where

\[
Y_i = \begin{bmatrix} \hat{y}_i(t+1) & \hat{y}_i(t+2) & \ldots & \hat{y}_i(t+d_i) \end{bmatrix}^T
\]

\[
y_i = \begin{bmatrix} y_i(t) & y_i(t) & \ldots & y_i(t) \end{bmatrix}^T
\]

\[
\hat{y}_i = \begin{bmatrix} \hat{y}_i(t) & \hat{y}_i(t) & \ldots & \hat{y}_i(t) \end{bmatrix}^T
\]

\[
X(t) = \begin{bmatrix} x_1^T(t) & x_2^T(t) & \ldots & x_m^T(t) \end{bmatrix}^T
\]

\[
\sigma_i = \begin{bmatrix} \sigma_{i1} & \sigma_{i2} & \ldots & \sigma_{id_i} \end{bmatrix}^T
\]
The performance index is chosen to be

\[ J = \frac{1}{2} \{ \sum_{i=1}^{m} [Y_{ci} - \hat{Y}_i]^T [Y_{ci} - \hat{Y}_i] + \beta \Delta^2 u(t) \} \]  

(3.226)

where \( Y_{ci} \) is the vector which contains the future values of the \( i \)th command signal. The control law is chosen such that

\[ \frac{dJ}{d\Delta u(t)} = 0 \]

(3.227)

Eq.(3.227) leads to

\[ u(t) = \epsilon [\beta u(t-1) + \alpha - k^T \hat{X}(t)] \]

(3.228)

where

\[
\begin{align*}
\epsilon &= (\beta + \sum_{i=1}^{m} \sigma_i^T \sigma_i)^{-1} \quad (3.229) \\
\alpha &= \sum_{i=1}^{m} \sigma_i^T (Y_{ci} - \hat{Y}_i + \hat{Y}_i) \quad (3.230) \\
k^T &= \sum_{i=1}^{m} \sigma_i^T K_i \quad (3.231)
\end{align*}
\]

Using Eqs.(3.220) and (3.228), it is possible to show that the closed-loop system is

\[
\begin{bmatrix}
X(t) \\ u(t+1)
\end{bmatrix} =
\begin{bmatrix}
A & b \\ -\epsilon k^T A & -\epsilon k^T b + \epsilon \beta
\end{bmatrix}
\begin{bmatrix}
X(t) \\ u(t)
\end{bmatrix}
+ \begin{bmatrix} 0 \\ \epsilon \end{bmatrix} \alpha
\]

(3.232)

Assume that \( P \) is the positive definite solution of the Lyapunov equation

\[ \Gamma^T P \Gamma - P = -Q \]

(3.233)
where

\[ Q > 0 \]  \tag{3.234} \\
\[ \Gamma = \begin{bmatrix} A & b \\ 0 & 0 \end{bmatrix} \]  \tag{3.235} \\

Let \( \lambda_{\text{min}}(Q) \) be the minimum eigenvalue of \( Q \) and

\[ \Delta = \begin{bmatrix} 0 & 0 \\ -\epsilon k^T A & -\epsilon k^T b + \epsilon \beta \end{bmatrix} \]

Using lemma(2.4), the closed-loop system given by Eq.(3.232) is stable provided that

\[ 0 \leq ||\Delta|| < -||\Gamma|| + \sqrt{||\Gamma|| + \frac{\lambda_{\text{min}}(Q)}{||P||}} \]  \tag{3.236} \\

It is clear that \( \epsilon \) can be made arbitrarily small by appropriate choices of \( d_i, i = 1, 2, \ldots, m \) and \( \beta \). This insures that the condition given in Eq.(3.236) is satisfied. Consequently, the closed-loop system given by Eq.(3.232) is stable.

**Example** Let the open-loop plant be

\[ x(t + 1) = \begin{bmatrix} -1 & 0.1 \\ 0.1 & -0.5 \end{bmatrix} x(t) + \begin{bmatrix} 0.25 \\ 0.11 \end{bmatrix} u(t) \]

\[ y(t) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x(t) \]

The model uses two Laguerre blocks. Each block has two filters. The time scale of the first block is 1 and that of the second block is 0.5. The sampling period is 0.1 sec. The GPC is tuned such that \( d_1 = 2, d_2 = 3 \) and \( \beta = 0.8 \). The initial estimates of the Laguerre gains are zeros. The initial covariance matrix is \( P_0 = 100 \times I \) where \( I \) is the identity matrix. The closed-loop response is shown in Fig.(3.18). It is well known that one input cannot drive two outputs to track two arbitrary command signals. This explains the behavior of the system in the intervals \( t \in [30, 40] \) and \( t \in [90, 100] \).
Figure 3.18: Control of a single-input, multi-output system
3.4.2 Multi-input, multi-output systems

Assume a MIMO system that can be represented by $m$ Laguerre blocks. For simplicity, assume that the system has $m$ inputs and $m$ outputs. The state-space model is

$$
\begin{bmatrix}
    x_1(t+1) \\
    x_2(t+1) \\
    \vdots \\
    x_m(t+1)
\end{bmatrix} =
\begin{bmatrix}
    A_1 & & & \\
    & A_2 & & \\
    & & \ddots & \\
    & & & A_m
\end{bmatrix}
\begin{bmatrix}
    x_1 \\
    x_2 \\
    \vdots \\
    x_m
\end{bmatrix} +
\begin{bmatrix}
    b_1 \\
    b_2 \\
    \vdots \\
    b_m
\end{bmatrix}
\begin{bmatrix}
    u_1(t) \\
    u_2(t) \\
    \vdots \\
    u_m(t)
\end{bmatrix}
\tag{3.237}
$$

The output vector is given by Eq.(3.221). Assume a one-step control horizon. The $j$-step ahead prediction of the output is

$$
\hat{y}_i(t+j) = \sum_{k=1}^{m} C_{ik}^T x_k(t+j)
= y_i(t) - \hat{y}_i + \sum_{k=1}^{m} C_{ik}^T A_k^j x_k(t) + \sigma_{kij} u_k(t-1) + \sigma_{kij} \Delta u_k(t)
\tag{3.238}
$$

where

$$
\sigma_{kij} = C_{ik}^T (A_k^{j-1} + A_k^{j-2} + \ldots + I) b_k
\tag{3.239}
$$

The predictions of $y_i$, $i = 1, 2, \ldots, m$, over a prediction horizon that extends from $d_{i1}$ to $d_{i2}$ are given by

$$
Y_i = \begin{bmatrix}
    \hat{y}_i(t+d_{i1}) & \hat{y}_i(t+d_{i1+1}) & \ldots & \hat{y}_i(t+d_{i2})
\end{bmatrix}^T
= y_i - \hat{y}_i + K_i X(t) + \sigma_i u(t-1) + \sigma_i \Delta u(t)
\tag{3.240}
$$

where

$$
\begin{align*}
    y_i &= \begin{bmatrix}
        y_i(t) & y_i(t) & \ldots & y_i(t)
    \end{bmatrix}^T \\
    X(t) &= \begin{bmatrix}
        x_1^T(t) & x_2^T(t) & \ldots & x_m^T(t)
    \end{bmatrix}^T
\end{align*}
$$
Chapter 3. Adaptive Generalized Predictive Control

\[ u(t - 1) = \begin{bmatrix} u_1(t - 1) & u_2(t - 1) & \ldots & u_m(t - 1) \end{bmatrix}^T, \]

\[ \Delta u(t) = \begin{bmatrix} \Delta u_1(t) & \Delta u_2(t) & \ldots & \Delta u_m(t) \end{bmatrix}^T \]

\[ K_i = \begin{bmatrix} a_{i1}^T a_{i1} & a_{i1}^T a_{i2} & \ldots & a_{i1}^T a_{im} \\ a_{i2}^T a_{i1} & a_{i2}^T a_{i2} & \ldots & a_{i2}^T a_{im} \\ \vdots & \vdots & \ddots & \vdots \\ a_{im}^T a_{i1} & a_{im}^T a_{i2} & \ldots & a_{im}^T a_{im} \end{bmatrix} \]

\[ \sigma_i = \begin{bmatrix} \sigma_{i1d_i1} & \sigma_{i1d_i1} & \ldots & \sigma_{i1d_i1} \\ \sigma_{i1d_{i1+1}} & \sigma_{i1d_{i1+1}} & \ldots & \sigma_{i1d_{i1+1}} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{i1d_{i2}} & \sigma_{i1d_{i2}} & \ldots & \sigma_{i1d_{i2}} \end{bmatrix} \]

(3.241)

The performance index is chosen to be

\[ J = \frac{1}{2} \left\{ \left[ \sum_{i=1}^{m} (Y_{ci} - Y_i)^T (Y_{ci} - Y_i) \right] + \Delta u^T(t) \Phi \Delta u(t) \right\} \]

(3.242)

where \( \Phi \) is a diagonal matrix of dimension \( m \times m \) with the \( i \)th element \( \phi_i \geq 0, i = 1, \ldots, m \), and \( Y_{ci} \) is the vector which contains the future values of the \( i \)th command signal. The control law is chosen such that

\[ \frac{dJ}{d\Delta u(t)} = 0 \]

(3.243)

Eq.(3.243) leads to

\[ u(t) = (\sum_{i=1}^{m} \sigma_i^T \sigma_i + \Phi)^{-1} \left\{ \Phi u(t - 1) + \sum_{i=1}^{m} \sigma_i^T (Y_{ci} + \hat{y}_i - y_i) - \sum_{i=1}^{m} \sigma_i^T K_i X(t) \right\} \]

(3.244)

The stability of the MIMO non-adaptive GPC is assessed in the following theorem.

**Theorem 3.5** Let the system be described by Eqs. (3.237) and (3.221) and controlled by the control law given in Eq.(3.244). Assume that the system is output controllable and
that \( y_{ci}, i = 1, \ldots, m \), are constant. Then, there always exist minimum and maximum prediction horizons, \( d_{i1} \) and \( d_{i2} \) and a weighting matrix \( \Phi \) such that the control system is stable and

\[
\lim_{t \to \infty} y_i(t) = y_{ci}, \quad i = 1, 2, \ldots, m \tag{3.245}
\]

**Proof:** In case of plant-model match

\[
y_i(t) = \hat{y}_i(t) \tag{3.246}
\]

So, Eq.(3.244) can be written as

\[
u(t) = \Psi \{ \Phi u(t - 1) - K X(t) + \sum_{i=1}^{m} \sigma_i^T Y_{ci} \} \tag{3.247}
\]

where

\[
\Psi = (\sum_{i=1}^{m} \sigma_i^T \sigma_i + \Phi)^{-1} \tag{3.248}
\]

\[
K = \sum_{i=1}^{m} \sigma_i^T K_i \tag{3.249}
\]

To study the closed-loop stability, consider the free system given by

\[
\begin{bmatrix}
X(t+1) \\
u(t+1)
\end{bmatrix} =
\begin{bmatrix}
A & B \\
-\Psi KA & -\Psi KB + \Phi \Psi
\end{bmatrix}
\begin{bmatrix}
X(t) \\
u(t)
\end{bmatrix} \tag{3.250}
\]

Note that \( A_i, i = 1, 2, \ldots, m \), are stable matrices, i.e. \( \lim_{k \to \infty} ||A_i^k|| = 0 \). So,

\[
\left| \begin{bmatrix}
0 & 0 \\
-\Psi KA & -\Psi KB + \Phi \Psi
\end{bmatrix}
\right|
\]

can be made arbitrarily small by choosing \( d_{i1}, i = 1, 2, \ldots, m \), big enough and \( ||\Phi|| \) small enough. This means that Lemma (2.4) can be satisfied and the closed-loop system is stable.
Let

$$\lim_{t \to \infty} u(t) = u$$

(3.251)

then

$$\lim_{t \to \infty} \sigma_i^T[K_i X(t) - \hat{y}(t)] = F_i u$$

(3.252)

where

$$F_i = \Gamma_i A$$

$$\Gamma_i = 
\begin{bmatrix}
\sum_{j=d_{i1}}^{d_{i2}} \sigma_{1ij} e_i^T(A_i^j - I) & \cdots & \sum_{j=d_{i1}}^{d_{i2}} \sigma_{1ij} e_i^T(A_i^j - I) & \cdots & \sum_{j=d_{i1}}^{d_{i2}} \sigma_{1ij} e_m^T(A_m^j - I) \\
\vdots & \ddots & \vdots & & \vdots \\
\sum_{j=d_{i1}}^{d_{i2}} \sigma_{mij} e_i^T(A_i^j - I) & \cdots & \sum_{j=d_{i1}}^{d_{i2}} \sigma_{mij} e_i^T(A_i^j - I) & \cdots & \sum_{j=d_{i1}}^{d_{i2}} \sigma_{mij} e_m^T(A_m^j - I)
\end{bmatrix}$$

$$\Lambda = 
\begin{bmatrix}
(I - A_1)^{-1} b_1 \\
(I - A_2)^{-1} b_1 \\
\vdots \\
(I - A_m)^{-1} b_m
\end{bmatrix}$$

Let $f_{kl}$ be the element in the $k$th row and the $l$th column of $F_i$, then

$$f_{kl} = \sum_{j=d_{i1}}^{d_{i2}} \sigma_{kij} e_i^T(A_i^j - I)(I - A_l)^{-1} b_l$$

$$= - \sum_{j=d_{i1}}^{d_{i2}} \sigma_{kij} e_i^T(A_i^{j-1} + \ldots + I)(I - A_l)(I - A_l)^{-1} b_l$$

$$= - \sum_{j=d_{i1}}^{d_{i2}} \sigma_{kij} \sigma_{lij}$$

(3.253)

So, we conclude that

$$F_i = -\sigma_i^T \sigma_i$$

(3.254)
Chapter 3. Adaptive Generalized Predictive Control

Taking the limit of Eq.(3.244) as $t \to \infty$ and using Eqs.(3.252) and (3.254), it is easy to show that

$$\lim_{t \to \infty} y_i(t) = y_{ci}$$

(3.255)

Note that

$$\lim_{t \to \infty} u(t) = u$$

$$= [C(I - A)^{-1}B]^{-1}Y_c$$

(3.256)

Eq.(3.256) shows that $[C(I - A)^{-1}B]^{-1}$ should exist. This corresponds to the output controllability condition as stated by Goodwin and Sin (1984).

It is straightforward to design an explicit adaptive control scheme based on the above formulation. The recursive least-squares estimator is used to identify the parameter vector $\xi_i^T$ associated with the ith output $y_i$.

$$\hat{\xi}_i(t) = \hat{\xi}_i(t - 1) + \frac{P(t - 1)X(t)}{1 + X^T(t)P(t - 1)X(t)}[y_i(t) - \hat{\xi}_i^T(t - 1)X(t)]$$

(3.257)

$$P(t) = P(t - 1) - \frac{P(t - 1)X(t)X^T(t)P(t - 1)}{1 + X^T(t)P(t - 1)X(t)}$$

(3.258)

where

$$\hat{\xi}_i^T = \begin{bmatrix} \hat{\xi}_{i1}^T & \hat{\xi}_{i2}^T & \ldots & \hat{\xi}_{im}^T \end{bmatrix} \quad i = 1, 2, \ldots, m$$

This means that the adaptive scheme uses $m$ estimators running in parallel. The control law is then computed every sample. The stability of the adaptive scheme is given by the following theorem.

**Theorem 3.6** Assume the plant is described by Eq.(3.237) and

$$y_i(t) = \hat{\xi}_i^Tz_i, \quad i = 1, \ldots, m$$

(3.259)
where

\[ Z(t) = \begin{bmatrix} X^T(t) & u^T(t) \end{bmatrix}^T \]  \hspace{1cm} (3.260)

Provided that the least-squares algorithm is used to estimate \( e_a^T_i \), \( i = 1, \ldots, m \), such that \( \dim e_a^T_i = \dim e_a^T \), then the closed-loop system described by Eqs. (3.237) and (3.244) is stable, i.e. \( \{y(t)\} \) and \( \{u(t)\} \) are bounded for all \( t \).

**Proof:** The proof is similar to that of Theorem 3.2 δ.

**Comments**

- If the number of inputs exceeds the number of outputs, the outputs can be controlled to track arbitrary command signals as long as the system is output controllable.

- If the number of inputs is less than the number of outputs, then, in general, the outputs cannot be controlled to track arbitrary command signals, Goodwin and Sin (1984).

**Example** The control of the headbox of a paper machine is a bench-mark problem to examine a controller in a multi-input, multi-output environment. The controlled variables are the stock level, \( y_1(t) \), and the total head pressure, \( y_2(t) \). The controlling variables are the stock volume flow rate, \( u_1(t) \), and the air mass flow rate, \( u_2(t) \). The purpose of the control is to change the turbulent flow in the approaching piping system to a sheet flow out of the headbox. The linearized state-space model, Zervos and Dumont (1988), is

\[
\mathbf{z}(t + 1) = \begin{bmatrix} -0.0115 & -0.1411 \\ -0.0373 & -0.527 \end{bmatrix} \mathbf{z}(t) + \begin{bmatrix} 0.1 & 0 \\ 0.324 & 0.2 \end{bmatrix} \mathbf{u}(t)
\]
\[ y(t) = \begin{bmatrix} 1 & 0 \\ 1 & 12.2412 \end{bmatrix} \bar{z}(t) \]

where

\[ \bar{z}(t) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \]

\( x_1(t) \) is the stock level.

\( x_2(t) \) is the air pad density.

The above system is modelled using two Laguerre blocks. Each block has 8 filters. The first block has a time scale \( p_1 = 0.06 \) and the second block has a time scale \( p_2 = 0.6 \). The system is sampled using a sample interval of 1.0 sec. The control law, Eq.(3.244) is applied such that \( d_{11} = 1, \ d_{12} = 4, \ d_{21} = 1, \ d_{22} = 4, \ \beta_1 = 1 \) and \( \beta_2 = 1 \). The closed-loop response is shown in Fig.(3.19).

### 3.5 Comparative study

To compare the performance of the Laguerre-filter based GPC (LAG-GPC), the CARIMA-model based GPC (GPC), and the model algorithmic control (MAC), three sets of simulations are presented below. Table 1 shows the tuning parameters for each example set. Note that \( n_p \) represents the number of parameters to be estimated.

<table>
<thead>
<tr>
<th>Example</th>
<th>GPC</th>
<th>MAC</th>
<th>LAG-GPC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n_p )</td>
<td>( \beta )</td>
<td>( n_2 )</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>0.5</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 1.1 Numerical values of the tuning parameters used in the comparative study.
Figure 3.19: Control of a multi-input, multi-output system
Chapter 3. Adaptive Generalized Predictive Control

Example 1: A high-order system.
Assume the open-loop system is given by

\[ G(s) = \frac{1}{(s + 1)^8} \]  \hspace{1cm} (3.261)

The system is sampled using a sample interval of 1.0 sec. The open-loop system is overdamped with a very slow initial response. Each controller uses an 8-parameter model. The CARIMA model has two poles and an extended numerator polynomial of 6 parameters. Figure (3.20) shows the output responses and the control signals for the different controllers. It is noted that the LAG-GPC has the least overshoot, the fastest settling time, and the smoothest control signal. The reason for the superior performance of the LAG-GPC is the ability of the Laguerre filters to model overdamped systems. By choosing the Laguerre pole optimally, the rate of decay of the Laguerre spectrum can be enhanced, Fu and Dumont (1991). This is an essential difference between Laguerre-filter models and model-sequence models. In the later case, the rate of convergence of the impulse-response coefficients cannot be controlled and depends on the controlled system. The CARIMA-model based GPC should have been the best, had the model order been equal to that of the actual system.

Example 2: A poorly damped system.
Assume the open-loop system is given by

\[ G(s) = \frac{1}{s^2 + 4\zeta s + 4} \]  \hspace{1cm} (3.262)

Assuming the plant order is known, a 5-parameter model suffices to model the system. Again, because of the poor convergence properties of the impulse-response coefficients, the MAC fails to stabilize the system in case of \( \zeta = 0 \). If \( \zeta = 0.1 \), the MAC gives a poor
Figure 3.20: Predictive control of a high order system
but stable closed-loop system as shown in Fig.(3.21). Both the GPC and the LAG-GPC manage to stabilize the system with $\zeta = 0$. The performance of the GPC is better than the LAG-GPC as the Laguerre-filter model theoretically requires an infinite number of filters to represent an oscillatory system. The superiority of the LAG-GPC over the MAC is evident.

**Example 3**: Nonminimum-phase and time-varying systems.
Consider a nonminimum-phase system described by the discrete equation

$$y(k) = 0.7y(k - 1) + u(k - 1) + 2u(k - 2)$$  \hspace{1cm} (3.263)

This system is used to demonstrate the robustness of a pole-placement self-tuner in Clarke, (1984) and to demonstrate the robustness of a predictive controller in Zervos, (1988). Figure (3.22) shows that all the predictive controllers in our study are able to drive the output signal to track the set-point. Suppose the plant changes to

$$y(k) = 1.7y(k - 1) - 0.72y(k - 2) + 0.1u(k - 1) + 0.2u(k - 2)$$  \hspace{1cm} (3.264)

The new plant dynamics include the the first order system given by Eq.(3.263). The tuning parameters are kept the same to check the robustness of each controller. Figure (3.23) shows that the MAC destabilizes the system while the LAG-GPC and the GPC stabilize the system.

### 3.6 Conclusion

An adaptive GPC based on Laguerre-filter modelling, (LAG-GPC), has been derived. Analyses have started with SISO systems. It has been shown that stability and convergence conditions would have been satisfied by proper tuning of the controller. The results have been extended to DMC and MAC. In case of plant-model mismatch, the
Figure 3.21: Predictive control of a poorly damped system
Figure 3.22: Predictive control of a nonminimum-phase system.
Chapter 3. Adaptive Generalized Predictive Control

Figure 3.23: Comparison of the robustness of the MAC, GPC and LAG-GPC.
robustness conditions would have been met by increasing the prediction horizon. It has been shown that a LAG-GPC based on a multi-step prediction horizon is more robust than a LAG-GPC based on a single-step prediction horizon. The LAG-GPC has been modified to control unstable systems, nonlinear systems, and stochastic systems. The LAG-GPC implementation has been extended to SIMO and MIMO systems. It has been shown that the stability and convergence properties of the SISO schemes have been carried over to MIMO systems. Finally, the LAG-GPC has been compared with the MAC and the CARIMA-model based GPC. For well damped systems and in the presence of plant-model mismatch, the LAG-GPC has been the best. The LAG-GPC has always been superior to the MAC. For a poorly damped plant, the CARIMA-model based GPC has outperformed the LAG-GPC and the MAC.
Chapter 4

Expert Adaptive GPC Based on Laguerre-Filter Modelling

4.1 Introduction

The term "Expert Control" was introduced in Åström, et al. (1986). It means that an Expert System is used to emulate the role of a human expert in control operation. The motive for using an artificial intelligence approach is that there are some aspects in control design which are not naturally amenable to numerical representation or which can be more efficiently represented by heuristics or rules of thumb. Expert control is implemented by adding, to an existing feedback loop, an expert system. The expert system consists of heuristics and theoretical control knowledge concerning tuning, adaptation, monitoring, and diagnosis.

In this chapter, an adaptive GPC based on Laguerre-filter modelling is implemented using expert-system and fuzzy-logic techniques. The expert system is meant to provide the user interface, model adjustment, on-line monitoring of the control system, and control commands. Fuzzy logic is used to implement the on-line tuning rules of the controller. This section reviews the main expert controllers reported in the literature. Section 4.2 gives a general overview of the expert shell G2, Gensym (1989). G2 is used to implement the expert system in this application. The actual implementation of the expert controller is explained in Section 4.3. Appendix A reviews the main characteristics of an expert system, knowledge acquisition procedures, and knowledge representation techniques. The material of this review is based on Årzen (1986) and Hayes-Roth (1983).
The review is not claimed to be comprehensive but gives enough background as a basis for our implementation. Appendix B provides basic definitions and terminologies of fuzzy logic.

There are two main approaches to implement expert-system techniques in control systems. One approach concentrates on the development of good control heuristics. These heuristics are often expressed in terms of "if - then" rules which are implemented with standard programming techniques. The other approach is to include expert-system techniques in the implementation. Several studies concerning the implementation of expert control systems are reported in the literature. Some examples are given below. However, it should be mentioned that a complete implementation is still missing.

An example of a system focused on control heuristics but implemented with conventional techniques is EXACT which is a self-tuning controller based on pattern identification of transients in the control error caused by load disturbances or set-point changes, Bristol, (1977). Heuristics and theoretical knowledge are used to adjust the controller parameters to achieve acceptable damping and overshoot.

An example of a system which is focused on control heuristics and implemented using expert system techniques is reported in Åström, et al. (1986). Most of the work reported in this area concerns smart auto-tuning of fixed, typically PID, controllers and supervision of adaptive controllers. In the auto-tuning approach, the step response analysis, (Ziegler and Nichols 1943), and the relay feedback method (Åström and Hägglund, 1984), are the most common. OPS4, a rule-based expert system is used for expert control by Åström, et al.(1986). The task of the expert system is to orchestrate the application of different numerical algorithms to the controlled plant. The expert system is used for the auto-tuning of a PID controller using a relay controller in the tuning mode and the tuned PID controller afterwards. OPS4 is not ideal for expert control. The system has no capability of backward chaining or reasoning with uncertainty. The most important
drawback regarding OPS4 is that it is not designed for real time operation.

In Sallé and Åström (1991), another PID auto-tuner is proposed. It combines two approaches: analysis of process transient responses and estimation of the process critical point through a relay feedback experiment. The system selects among a PI, a PID or a PI regulator coupled with a Smith predictor. However, the processes are assumed to be stable, to have a globally monotonic step response and to give stable oscillations under relay feedback.

In Årzen (1989), relay auto-tuning of PID controllers is implemented using the object-oriented system, "FLAVORS", and the forward chaining production system, "YAPS". However, as reported in Årzen (1989) and (1986), the limitations of this implementation are: the process is assumed to have a finite positive steady-state gain, the tuning technique is based on a relay experiment so it cannot deal with all types of processes, and it is sensitive to load disturbances.

An expert system for real-time control is reported in Moore, (1986). PICON which stands for "Process Intelligent CONtrol" is a system meant for real-time applications, Moore (1987). A space-control expert system is described by Leinweber, (1987) as a prototype using PICON. The difficulties in using an expert system such as PICON are: knowledge requirements are unstructured and may be broad in scope, the knowledge base is highly specific to the individual plants, and the number of rules may exceed the practical limitations.

In Doraiswami and Jiang (1989), a real-time expert controller is implemented on PC-AT microcomputer and is interfaced to an analogue simulation of a hydraulic turbine generator. The turbine is controlled using a digitally tunable analogue PI controller. Under normal conditions, the gains of the controller are obtained using conventional techniques. Under contingencies, the knowledge-based controller should be able to detect, classify and correct system failures.
In Haest, et al. (1989), system identification is carried out using an expert system called "ESPION". ESPION is written using OPS85 rule-based language and is used for identification of linear MISO systems in ARMAX forms. Provided with a set of data, the system will organize a search through the set of candidate structures and end up with a "best" model according to a "quality index" set by the system developers.

In Lebow and Blankenship (1987), a microcomputer based expert controller is suggested. The control algorithm, a PID controller, resides in a separate microprocessor and the expert system resides in another. The controller is tuned using fuzzy logic, Zadeh (1973). Fuzzy logic provides a method for assigning a measure of how good a response is. Each specific criterion, e.g. the output overshoot, the settling time, etc., has its own membership function. A score is defined for each response as the sum of all the criteria's membership values. The objective of the expert system is to optimize this score.

Fuzzy logic can be used to implement linguistically expressed control policies, King and Mamdani (1977). The calculations of the control algorithm are composed of the four stages. First, the error and its rate of change are calculated. Second, the error and its rate of change are converted to fuzzy variables. Third, fuzzy-logic decision rules are applied to get a fuzzy control variable. Fourth, a deterministic control action is calculated. In Pocky and Mamdani (1979), a self organizing controller which is based on fuzzy logic is presented. The algorithm has a performance measure to evaluate the controller decisions. If a poor performance results in, a credit assignment procedure is used to translate the output deviations into corrections of the inputs that have been applied and caused this poor performance. A learning mechanism makes use of this performance measure and credit assignment to upgrade the system control rules.
4.2 The expert shell, G2

G2 is an expert shell designed for real time process control. The key element of G2 is an item. An item is a piece of knowledge in an application. Workspaces, objects, functions, rules, and displays are examples of items. By creating instances of the different items, a knowledge base can be built. Associated with each item is a table of attributes which contains knowledge about that item. For example, it is possible to define an item called Controller as an object and create GPC as an instance of Controller. The attributes of GPC can be control-horizon, prediction-horizon, and control-weight. All items in G2 are arranged in an item hierarchy. Each item in the hierarchy can inherit attributes from its superior and any rules that apply to an item apply to all items below it. For example, it is possible to design CARIMA-GPC, Laguerre-GPC, and Dynamic-Matrix-Control as subclasses of the object Controller. The new items inherit the attributes of the item Controller. A brief description of the items used in building our expert system is given below.

A workspace is like a shell which accommodates the knowledge base. It can contain rules, objects, displays, etc. A subworkspace is a workspace that is associated with an item. A subworkspace can hold items that in turn have their own subworkspaces and so on. This allows a hierarchical organization of knowledge. If an item is given the capability "activatable-subworkspace", the associated workspace can be activated or deactivated on-line. This capability will be used later on to enhance the efficiency of our expert system.

An object is a representation of some part of an application. Each class of objects is defined by an object definition. The object definition is an abstraction of the object. Actual application objects are instances. Each object has a table of attributes. Whereas the definition of an attribute specifies the allowable type of data, each instance of the
class has values of the given type. The types of data provided with G2 are quantitative, symbolic, logical and text. There are many object classes provided with G2. The one that is used in our application is the variable-or-parameter class which has Variables and Parameters as subclasses. A variable is an object that has values that change and can expire. A variable may have no current value at times. It can trigger data seeking. A parameter is an object that has a changing value that never expires. It always has a value and never causes data seeking.

A function is a predefined named sequence of operations that are not performed until the function is called. G2 has many built-in functions, e.g. max(x, y, z), cos(x), etc. G2 has also a foreign function interface which allows C and Fortran functions to be called within G2. To use a foreign function, two steps must be carried out. First, the C or Fortran object file that contains the foreign function is loaded within G2. Second, the foreign function is declared within G2 knowledge base. The declaration of the foreign function indicates the function language, whether its value is real or integer, the function name, and the arguments to be passed from G2 to the function.

A rule establishes the way that G2 responds to various conditions in the application. A rule has two parts: an antecedent which specifies the conditions and a consequent which specifies the actions. There are five types of rules provided with G2:

- if rules: An if rule is fired (executed), if the inference engine can evaluate the antecedent of that rule. The if rules can be invoked through backward and forward chaining. An if rule can be scanned periodically which makes them suitable for real time applications, e.g. monitoring the closed-loop stability, the output error, etc.

- when rules: A when rule is similar to an if rule except that G2 does not invoked a when rule through forward or backward chaining. A when rule can be invoke periodically.
• *initially* rules: These rules are invoked when the knowledge is started or when an object which has the capability *activatable subworkspace* is activated.

• *whenever* rules: A *whenever* rule is fired if special kinds of events occur, e.g. emergency situations.

• *unconditional* rules: This kind of rules is carried out unconditionally each time the rule is invoked.

A display is a screen item that shows the value of a variable or parameter. There are four types of displays: readout tables, graphs, meters, and dials. In our expert system, readout tables are used to display the values of a parameter or a variable, while, graphs are used to show a plot of the values of a parameter or a variable over time.

The on-line communication between the expert system within the G2 environment and the outside world is carried out using the "G2 Standard Interface", GSI, Gensym (1991). GSI is made up of three basic components:

• GSI base process: The base process is a G2 running program with the GSI option enabled.

• GSI extension process: The extension process is composed of the calling code, the GSI extension code, the application bridge code, and the external application.

• ICP communication link: This is a communication link between the base process and the extension process.

To build a full application that uses GSI, the developer must develop two application-specific pieces of the system in addition to developing the G2 knowledge base:

• The configuration specification which configures the knowledge base to communicate with the external application.
The application bridge code which the GSI extension uses to interact with the external application. Our expert system uses the bridge code developed in Zhou, et al. (1991).

4.3 Implementation

4.3.1 Overview

The hardware platform of the proposed expert system is a VAX-station 3100 running VMS and DEC windows. The expert shell G2 is used for implementing the expert system. G2 can exchange information with the real world outside its environment using foreign functions, which can be in C or Fortran, or using the the standard interface GSI. Both the expert shell and the numerical algorithms run on the same VAX-station. The system starts by initiating a dialogue with the user to extract the available information. Then, a search algorithm is executed to adjust the parameters of the Laguerre-filter model. After adjusting the model, the adaptive LAG-GPC, equipped with fuzzy-logic tuning rules, is applied to the process. The implementation details are explained below.

To achieve an efficient reasoning and fast decision making, the knowledge base is spread over a number of subworkspaces. These subworkspaces are activated or deactivated according to the current state of the expert system. A master workspace called INITIAL-RULES orchestrates the switching to the different subworkspaces. INITIAL-RULES has 6 subworkspaces that can be activated. The functions of these subworkspaces are described below.

1. The subworkspace INI-RULE-1 has 7 rules to extract information concerning the system settling time.

2. The subworkspace INI-RULE-2 has 8 rule to extract information concerning the
dead time.

3. The subworkspace \textbf{INI-RULE-3} has 6 rule to extract information concerning the rise time.

4. The subworkspace \textbf{MODELLING} has 9 rules and 10 instances of a quantitative parameter called \textbf{LAGUERRE}. The function of this part of the knowledge base is to choose the number of Laguerre filters used for modelling and the Laguerre time constant. The mechanism of adjusting the model will be explained in details later in this section.

5. The subworkspace \textbf{TUNING} has 3 rules designed to initially tune the GPC.

6. The subworkspace \textbf{GPC} has 9 rules and 2 subworkspaces which have 9 more rules. This part of the knowledge base is responsible for monitoring and tuning the controller on line to insure closed-loop stability and achieve acceptable closed-loop settling time and overshoot.

In addition to \textbf{INITIAL-RULES} and its associated subworkspaces, there are 5 workspaces as explained below.

1. The workspace \textbf{SENSOR-DEFINITIONS} contains three object definitions. Two of them define the sensors which will send/get integer and floating numbers to/from the external application. The third object definition defines the GSI interface. This object has one instance called \textbf{INTERFACE-1} which defines to G2 the GSI connection configuration, the attributes which identify a sensor, and a remote process initialization which identifies the sensors' names and types.

2. The workspace \textbf{FUNCTIONS} contains 10 function declarations; 5 of them are in Fortran and the rest are in C. These functions are responsible for extracting the
system dead-time, settling time and rise time from step-response data if available or carrying out a step-response experiment if allowable. These functions also store the system information and the controller tuning parameters in external files. These external files are accessible to G2 as well as the numerical algorithms outside the G2 environment. These functions contain also 72 fuzzy rules that are used on line to improve the controller performance.

3. The workspace VARIABLE-AND-PARAMETER-DEFINITION contains icon definitions for quantitative variables, qualitative parameters, and text variables. Instances of these classes are used throughout the knowledge base to represent information about the system.

4. The workspace LAGUERRE-PARAMETERS contains 14 quantitative parameters which are used to get information about the plant as well as the controller, e.g. the number of Laguerre filters, the plant dead time, the prediction horizon, etc.

5. The workspace SIGNAL-DISPLAY contains graphs to display the output and the control signals as well as readout tables to display the settling time and the maximum overshoot.

4.3.2 User interface

A user-interface facility is an essential feature of any expert system. It brings the computer close to emulating a human expert who tries to help the user solving his/her problem. The user interface developed for this application is meant to achieve the following objectives:

1. To show the user how to use the system.
2. To explain to the user in simple terms the expressions used by the experts in the control field.

3. To ask the user questions to extract information to initialize the controller.

4. To give logical interpretation of the decisions taken by the system.

5. To inform the user on the current tasks carried out by the system.

It is explained below how the developed interface achieves the above objectives.

When the user starts the knowledge base, a "welcome" screen appears followed by initial dialogue screens. Each screen is self explanatory and instructs the user on how to deal with it. For example, the "welcome" screen tells the user that the system is meant to control a stable well-damped process and asks him/her to click on a certain button to proceed. Each initial-dialogue screen instructs the user on the expected answer, the format of that answer, and where to type it. The top right corner of the computer screen has a message that tells the user how to start, stop, restart, or reset the knowledge base.

During the initial dialogue stage, the user is asked some questions, e.g. Do you know how long it takes the process step-response to reach steady state?, Does the open-loop response have a dead time?, etc. Each question has an "explain" option which the user can use to define terminology used in the control field, e.g. steady-state response, dead time, rise time, etc. According to the user's answers to these questions, he/she is prompted to give numerical values to some values to some variables, e.g. dead time, or step-response data are used to get the required information.

During the model-adjustment stage, a screen appears that shows the execution of the different rules. So, the user can follow the search process. Using "readout" tables, the user gets an explanation why a certain model parameter has changed. The membership function of the controller tuning rules are typed and the tuning actions based on those
membership functions are indicated in a window.

A message board that appears at the top left corner of the computer screen is used to inform the user that a certain task is being carried out, e.g. the model tuning is on, or has been completed, e.g. the model has been tuned. It is also used to inform the user about the output status, e.g. steady state has been reached.

4.3.3 Model adjustment

Search techniques

A search problem is characterized by an initial-state and a goal-state description. To go from the initial state to the goal state, the search process passes through several intermediate states. This search procedure proceeds using operators. A single operator transforms the state into another which is hopefully closer to a goal state. The problem may require finding the best solution or just any satisfactory solution. The objective is to search until a solution is found or the algorithm is satisfied that no solution exists, Charniak and McDermott (1987).

A search algorithm may contain: an operator-ordering function to order the applicable operators at each state from best to worst and a state-evaluation function to give the estimated distance of each state from the nearest goal state. If the search is based on rules of thumb, it is called a heuristic search, Nilsson (1980).

There are two basic approaches of systematic search: depth-first approach and breadth-first approach. Depth-first search corresponds to "last-in / first-out" strategy. This means that if two states $S_1$ and $S_2$ are produced by applying operators to a state $S$, then every state reachable from $S_1$ will be examined before any state reachable from $S_2$. If there is an infinite number of states reachable from $S_1$ then $S_2$ will never be examined. In this case, some kind of depth cut-off should be imposed. On the other
hand, breadth-first search corresponds to a "first-in / first-out" strategy. This means that the search algorithm explores all states that are \( n \) operator applications from the initial state before any that \( n + 1 \) away.

The search for a set of parameters that characterize a Laguerre-filter based model can be described in the above context. The initial state represents the initial parameters chosen for the model. The goal state represents the final parameters which are actually implemented. The operators are the decisions taken by the designer to change one or some of the model parameters. The state-evaluation function is the model prediction-error. The search algorithm used in this application is explained in detail below.

**Search implementation**

The knowledge source for choosing the Laguerre-model parameters is a workspace called **Modelling**. **Modelling** has 10 instances, called \( P, POPT, P1, \ldots, P8 \), of an object definition called **LAGUERRE**. The definition of **LAGUERRE** is

<table>
<thead>
<tr>
<th>Notes</th>
<th>ok</th>
</tr>
</thead>
<tbody>
<tr>
<td>User restrictions</td>
<td>none</td>
</tr>
<tr>
<td>Superior class</td>
<td>object</td>
</tr>
<tr>
<td>Attributes specific to class</td>
<td>Laguerre-pole is given by a quantitative variable</td>
</tr>
<tr>
<td></td>
<td>lag-model-error is given by a quantitative variable</td>
</tr>
<tr>
<td>Capabilities and restrictions</td>
<td>none</td>
</tr>
</tbody>
</table>

An example of an instance definition is
Chapter 4. Expert Adaptive GPC Based on Laguerre-Filter Modelling

Each one of the attributes: \textit{Laguerre-pole} and \textit{lag-model-error} has a subtable associated with it as follows.

<table>
<thead>
<tr>
<th>Notes</th>
<th>ok</th>
</tr>
</thead>
<tbody>
<tr>
<td>User restrictions</td>
<td>none</td>
</tr>
<tr>
<td>Name</td>
<td>P1</td>
</tr>
<tr>
<td>Laguerre-pole</td>
<td>***</td>
</tr>
<tr>
<td>Laguerre-model-error</td>
<td>***</td>
</tr>
</tbody>
</table>

Note that the formula attribute is different in each subtable of the instances, \( P_1, \ldots, P_8 \). This attribute defines the search space around \( P \). It is possible to extend the search space by creating more instances of \texttt{LAGUERRE}. The instances \( P \) and \( POPT \) have no formula in the \textit{formula} attributes of their subtables.
Chapter 4. Expert Adaptive GPC Based on Laguerre-Filter Modelling

<table>
<thead>
<tr>
<th>a quantitative variable, the lag-model-error of P1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options</td>
</tr>
<tr>
<td>Notes</td>
</tr>
<tr>
<td>Validity interval</td>
</tr>
<tr>
<td>Formula</td>
</tr>
<tr>
<td>Data server</td>
</tr>
<tr>
<td>Default update interval</td>
</tr>
</tbody>
</table>

Note that \(\text{model1}\) is a foreign function that G2 calls. The arguments passed to \(\text{model1}\) are the number of Laguerre filters, the Laguerre pole, and the sampling interval. \(\text{Model1}\) has a built-in RLS estimator and returns the model error corresponding to the suggested parameters. There are similar calls associated with the instances, P2, …, P8. In the subtable of POPT, the formula attribute has a call for a G2 built-in function that picks up one instance out of the instances, P1, …, P8, that has the minimum model error.

Given the sampling interval, the search algorithm starts with the initial number of filters and Laguerre pole. These values are calculated based on the user information and the experiments carried out during the initial dialogue stage. The goal state is represented by the number of filters and Laguerre pole which result in an error model less than a threshold value. The operators that are applied to a state to obtain a new one are the formulas in the formula attribute of the subtable of laguerre-pole. The formulas in the formula attribute of the subtable of lag-model-error are the state-evaluation functions that give the estimated distance of each state from the goal state. Fixing the model order and using the pole of the Laguerre filters as the only tuning parameter may fail to yield the required model error. Another operator that can be applied is to increase the number of filters. Because of the theoretical properties of the Laguerre filters, it is certain that the whole search problem will end up finding a goal state. The search for
the Laguerre-filter pole is carried out in a breadth first fashion while the adjustment of the number of filters adopts a depth first strategy.

The search algorithm can be summarized as follows

1. Initially conclude that the *Laguerre-pole* of $P = \text{the initial value supplied from the initial dialogue stage and calculate the lag-model-error.}$

2. Generate new states in the search space by applying operators to get the *Laguerre-pole* of $P_1, \ldots, P_8.$

3. Apply the state-evaluation function to get the *lag-model-error* corresponding to each state in step 2.

4. Choose $P_{OPT}$ to correspond to the nearest state to the goal state.

5. If the *lag-model-error* of $P_{OPT} \leq \text{the lag-model-error of } P$ then copy $P_{OPT}$ to $P.$

6. If the standard deviation of the *Laguerre-pole* of $P \leq 10^{-2}$ for 10 trials and the *lag-model-error* of $P > 5 \times 10^{-5}$ then increase the number of filters by one. Go to step 2.

7. If the *lag-model-error* of $P > 5 \times 10^{-5}$ then go to step 2.

8. If the standard deviation of the *Laguerre-pole* of $P \leq 10^{-2}$ for 5 trials and the *lag-model-error* of $P \leq 5 \times 10^{-5}$ then conclude that the model has been tuned and end the model adjustment algorithm.

4.3.4 Tuning based on fuzzy logic

There are four steps to design a fuzzy logic tuner.
1- Defining the input and output spaces

A linguistic variable in the antecedent of a fuzzy rule forms a fuzzy input space with a certain universe of discourse, while that in the consequent of the rule forms a fuzzy output space. In this implementation, there are two fuzzy input spaces associated with the linguistic variables; **Overshoot** and **Settling-time** and two fuzzy output spaces associated with the linguistic variables; **Output-horizon** and **Control-weight**.

2- Fuzzy partition

Each linguistic variable is associated with a term set. Each term in the term set is defined on the same universe of discourse. A fuzzy partition determines how many terms should exist in a term set. In this application the term sets are defined as

\[
T(x) = \{\text{very -- acceptable, acceptable, more -- or -- less -- acceptable,}
\]

\[
\text{more -- or -- less -- unacceptable, unacceptable, very -- unacceptable}\}
\]

\[
T(y) = \{\text{big -- increase, medium -- increase, small -- increase,}
\]

\[
\text{no -- change, small -- decrease, medium -- decrease, big -- decrease}\}
\]

\(x\) can be the input linguistic variable **Overshoot** or **Settling-time**. \(y\) can be the output linguistic variable **Prediction-horizon** or **Control-weight**. For example, the universes of discourses of **Overshoot**, **Settling-time**, **Prediction-horizon**, and **Control-weight** are defined, respectively, as follows :

\[
U_1 = \{7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17\}
\]

The elements of this discourse represent the percentage maximum overshoot.

\[
U_2 = \{0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5\}
\]

The elements of this discourse represent the settling time relative to the open-loop one.
$U_3 = \{-8, -4, -2, 0, 2, 4, 10\}$

The elements in this discourse represent the possible change in the output prediction horizon.

$U_4 = \{0.1, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 2.0, 5.0, 10.0\}$

The elements in this discourse represent the factors by which the current value of the control weighting factor is multiplied.

3- Defining the membership functions

There are two methods used for defining fuzzy sets; numerical and functional. In the numerical definition, a grade of membership function of a fuzzy set is represented as a vector of numbers whose dimension depends on the number of elements in the corresponding universe of discourse. In the functional definition, the membership function of a fuzzy set is expressed analytically.

The functional definition is chosen to define the fuzzy term “acceptable” both for Overshoot and Settling-time because it can adapt easily to changes in the normalization of the corresponding universes. The form of the membership function used is

$$
\mu(x_i) = \begin{cases} 
1 & \frac{x_i}{x_n} < x_f \\
\frac{x_i}{x_n} \geq x_f & \end{cases}
$$

(4.265)

$x_n$ normalizes $x_i$ so that it can be mapped to the universe of discourse. $x_f$ defines the edge between the values which have full membership and those which have partial membership. $k$ defines the crossover point in the membership function.

Consequently, the membership of the fuzzy terms very acceptable, more or less acceptable, unacceptable, more or less unacceptable, and very unacceptable are $\mu^2$, $\sqrt{\mu}$, $1 - \mu$, $\sqrt{1 - \mu}$, and $(1 - \mu)^2$, respectively.
The membership function of the fuzzy term "acceptable" in case of the fuzzy variable Overshoot has $k = 100$, $x_f = 0.02$, and $x_n = 3$, while that of the fuzzy variable Settling-time has $k = 25$, $x_f = 0.7$, and the open-loop settling time as $x_n$.

The numerical definition is chosen to define the sets of the fuzzy terms associated with the fuzzy variables; Prediction-horizon and Control-weight as shown in tables 1-2. The symbols BD, MD, SD, NC, SI, MI, and BI stand for big-decrease, medium-decrease, small-decrease, no-change, small-increase, medium-increase, and big-increase.

<table>
<thead>
<tr>
<th>$u$</th>
<th>BI</th>
<th>MI</th>
<th>SI</th>
<th>NC</th>
<th>SD</th>
<th>MD</th>
<th>BD</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.0</td>
<td>0.5</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>0.1</td>
<td>0.5</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>-2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.5</td>
<td>0.1</td>
</tr>
<tr>
<td>-4</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
</tr>
<tr>
<td>-8</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Table 1 The fuzzy sets used with the linguistic variable "Prediction-horizon".

Note that $u \in U_3$.
Chapter 4. Expert Adaptive GPC Based on Laguerre-Filter Modelling

<table>
<thead>
<tr>
<th>u</th>
<th>BI</th>
<th>MI</th>
<th>SI</th>
<th>NC</th>
<th>SD</th>
<th>MD</th>
<th>BD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.2</td>
<td>1.0</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.5</td>
<td>0.7</td>
</tr>
<tr>
<td>0.6</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.9</td>
<td>0.4</td>
</tr>
<tr>
<td>0.7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
<td>0.7</td>
<td>0.2</td>
</tr>
<tr>
<td>0.8</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.8</td>
<td>0.3</td>
<td>0.1</td>
</tr>
<tr>
<td>0.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.9</td>
<td>0.1</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.1</td>
<td>0.0</td>
<td>0.1</td>
<td>0.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.2</td>
<td>0.0</td>
<td>0.2</td>
<td>0.8</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.3</td>
<td>0.1</td>
<td>0.4</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.4</td>
<td>0.2</td>
<td>0.7</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.5</td>
<td>0.3</td>
<td>0.9</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2.0</td>
<td>0.6</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>5.0</td>
<td>0.8</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>10.0</td>
<td>1.0</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 2 The fuzzy sets used with the linguistic variable “Control-weight”.

Note that \( u \in U_4 \).

4- The rule base

The fuzzy-logic tuner is characterized by a set of linguistic statements based on expert knowledge. The expert knowledge is usually in the form of "IF-THEN" rules. The collection of these fuzzy-logic rules forms the rule base. The rules depend on the linguistic variables chosen in the first step. This choice is based on engineering judgement. The derivation is heuristic and relies on the qualitative knowledge gained through this research. Tables 3-4 show the linguistic rules which will be used on-line to adjust the
output prediction horizon and the control weighting factor. The symbols ST and OS stand for the linguistic variables; **Settling-time** and **Overshoot**, respectively. While the symbols; MLAC, AC, VAC, MLUAC, UAC, and VUAC stand for the fuzzy terms; more-or-less acceptable, acceptable, very-acceptable, more-or-less-unacceptable, unacceptable, and very-unacceptable, respectively. Note that the tuning rules reflect the fact the open-loop plant is assumed stable and well-damped.

<table>
<thead>
<tr>
<th>ST/OS</th>
<th>MLAC</th>
<th>AC</th>
<th>VAC</th>
<th>MLUAC</th>
<th>UAC</th>
<th>VUAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLAC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>SI</td>
<td>SI</td>
<td>MI</td>
</tr>
<tr>
<td>AC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>MI</td>
<td>MI</td>
<td>BI</td>
</tr>
<tr>
<td>VAC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>MI</td>
<td>MI</td>
<td>BI</td>
</tr>
<tr>
<td>MLUAC</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
<td>SI</td>
<td>SI</td>
<td>SI</td>
</tr>
<tr>
<td>UAC</td>
<td>MD</td>
<td>MD</td>
<td>MD</td>
<td>SI</td>
<td>SI</td>
<td>SI</td>
</tr>
<tr>
<td>VUAC</td>
<td>BD</td>
<td>BD</td>
<td>BD</td>
<td>SI</td>
<td>SI</td>
<td>SI</td>
</tr>
</tbody>
</table>

Table 3 The fuzzy-logic rules used to adjust the output prediction horizon.

<table>
<thead>
<tr>
<th>ST/OS</th>
<th>MLAC</th>
<th>AC</th>
<th>VAC</th>
<th>MLUAC</th>
<th>UAC</th>
<th>VUAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLAC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>SI</td>
<td>MI</td>
<td>MI</td>
</tr>
<tr>
<td>AC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>SI</td>
<td>BI</td>
<td>BI</td>
</tr>
<tr>
<td>VAC</td>
<td>NC</td>
<td>NC</td>
<td>NC</td>
<td>MI</td>
<td>BI</td>
<td>BI</td>
</tr>
<tr>
<td>MLUAC</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
</tr>
<tr>
<td>UAC</td>
<td>MD</td>
<td>BD</td>
<td>BD</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
</tr>
<tr>
<td>VUAC</td>
<td>MD</td>
<td>BD</td>
<td>BD</td>
<td>SD</td>
<td>SD</td>
<td>SD</td>
</tr>
</tbody>
</table>

Table 4 The fuzzy-logic rules used to adjust the control weighting factor.

Consider the fuzzy-logic rules in Table 3, the linguistic form of these rules is

\[ R_1: \text{IF } \text{Overshoot is acceptable AND Settling-time is acceptable THEN Output-horizon is no-change} \]
$R_{36}$: also, IF Overshoot is very-unacceptable AND Settling-time is very-acceptable THEN Output-horizon is big-increase

So, given that Overshoot is $A$ and Settling-time is $B$, Output-horizon is deduced to be $C$ as

$$C = \bigcup_{i=1}^{36} (A, B) \circ R_i$$

(4.266)

By interpreting $\circ$ as the sup-min operator and the connective "also" as the union operator, it is possible to show that Eq.(4.266) is equivalent to, Lee (1990),:

$$C = (A, B) \circ \bigcup_{i=1}^{36} R_i$$

(4.267)

Consequently, the rules associated with each output variable can be summed up in one matrix of three dimensions. This is carried out off-line. Tables 3-4 result in two matrices which are used to infer the tuning action on-line. The inference mechanism in this case becomes efficient time-wise.

The rules in the subworkspace GPC monitor the closed-loop system. If the tuned controller results in an unstable closed-loop system, the control signal is set to zero and the output prediction horizon is increased by 5. After the system states settle to zero, the controller is switched on again using the new parameters. On the other hand, if the closed-loop system is stable, the expert system fires some rules to calculate the percentage maximum overshoot and the settling time. The expert system judges that steady state has been reached if the output error is less than $\epsilon_e$ and the standard deviation of the output error is less than $\sigma_e$ for a time $t$, where $\epsilon_e$, $\sigma_e$ and $t$ are dependent on the controlled plant.

Once the settling time and the percentage maximum overshoot are calculated, a fuzzification operator is applied to convert each of them from a crisp value into a fuzzy
singleton within the corresponding universes of discourse. Then, the fuzzy tuning rules are applied. The expert-system rules are designed such that the controller is tuned first using the output prediction horizon. The change in the output horizon is constrained such that the controller can look behind the system dead-time and nonminimum phase response. The control weight is tuned if the output-horizon tuning rules fail or if finer tuning is needed. The result of applying the fuzzy logic tuning rules is a fuzzy set specifying a possibility distribution of tuning actions. To obtain a crisp (non-fuzzy) tuning action, defuzzification is required. Following Mamedani (1974), the max criterion method is adopted. The max criterion method produces the point at which the possibility distribution of the tuning action reaches a maximum value.

4.3.5 Illustrative examples

Example 1

Let the plant be given by

\[ \frac{y}{u} = \frac{0.16z^{-3}}{1 - 1.4z^{-1} + 0.48z^{-2}} \quad (4.268) \]

The above plant is overdamped with a time delay. It is assumed that no information, but a rough estimate of the open-loop settling time, is available. The open-loop settling time is used to set the initial value of the Laguerre-filter pole and to compare the closed-loop settling time with the open-loop one. The details of the search algorithm, which is used to tune the model, are given below. Assuming a three-filter model, the first state in the search space is given by the following table.

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_i )</td>
<td>0.133</td>
<td>0.15</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.08</td>
<td>0.075</td>
</tr>
<tr>
<td>( e_i )</td>
<td>2.64 \times 10^{-3}</td>
<td>7.90 \times 10^{-4}</td>
<td>8.09 \times 10^{-5}</td>
<td>1.111 \times 10^{-4}</td>
<td>1.115 \times 10^{-4}</td>
<td>1.115 \times 10^{-4}</td>
<td>4.32 \times 10^{-2}</td>
<td>5.00 \times 10^{-2}</td>
</tr>
</tbody>
</table>

The algorithm picks \( P_3 \) as the model pole, \( P \). A new state is generated as follows
Chapter 4. Expert Adaptive GPC Based on Laguerre-Filter Modelling

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_i$</td>
<td>0.267</td>
<td>0.30</td>
<td>0.40</td>
<td>0.60</td>
<td>0.80</td>
<td>1.00</td>
<td>0.16</td>
<td>0.15</td>
</tr>
<tr>
<td>$e_i$</td>
<td>1.095 x 10^{-4}</td>
<td>1.111 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>3.66 x 10^{-4}</td>
<td>7.9 x 10^{-4}</td>
<td></td>
</tr>
</tbody>
</table>

The algorithm picks $P_1$. However, since the model error, $e_1$, corresponding to $P_1$ is greater than that corresponding to $P$, the model pole remains unchanged. The search algorithm continues using a three-filter model. After 5 iterations it is clear that $P$ can not be adjusted to yield an acceptable model error. The algorithm decides to increase the number of filters by one. A new state is introduced as follows

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_i$</td>
<td>0.267</td>
<td>0.30</td>
<td>0.40</td>
<td>0.60</td>
<td>0.80</td>
<td>1.00</td>
<td>0.16</td>
<td>0.15</td>
</tr>
<tr>
<td>$e_i$</td>
<td>1.068 x 10^{-4}</td>
<td>1.103 x 10^{-4}</td>
<td>1.114 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.39 x 10^{-4}</td>
<td>5.97 x 10^{-4}</td>
<td></td>
</tr>
</tbody>
</table>

Again, using 4 filters, the search algorithm fails to produce an acceptable model error.

The same conclusion is true for the cases of 5-filter and 6-filter models. Eventually, a 7-filter model produces an acceptable result as shown below.

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_i$</td>
<td>0.267</td>
<td>0.30</td>
<td>0.40</td>
<td>0.60</td>
<td>0.80</td>
<td>1.00</td>
<td>0.16</td>
<td>0.15</td>
</tr>
<tr>
<td>$e_i$</td>
<td>4.0 x 10^{-5}</td>
<td>6.87 x 10^{-5}</td>
<td>1.107 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>5.19 x 10^{-3}</td>
<td>6.14 x 10^{-3}</td>
<td></td>
</tr>
</tbody>
</table>

The algorithm chooses $P_1$ as the new $P$. A new state is generated to insure that the search space in the neighborhood of $P$ does not have a better tuning pole $P_i$.

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_i$</td>
<td>0.356</td>
<td>0.40</td>
<td>0.53</td>
<td>0.80</td>
<td>1.067</td>
<td>1.33</td>
<td>0.213</td>
<td>0.20</td>
</tr>
<tr>
<td>$e_i$</td>
<td>1.06 x 10^{-4}</td>
<td>1.1 x 10^{-4}</td>
<td>1.11 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.115 x 10^{-4}</td>
<td>1.05 x 10^{-3}</td>
<td>1.73 x 10^{-3}</td>
<td></td>
</tr>
</tbody>
</table>

Figure (4.24), shows the performance of the adaptive GPC based on Laguerre-filter modelling. The controller is initially tuned such that $n_2 = 6$ and $\beta = 0.1$. The settling time of the closed-loop system is 60% of that of open-loop system. The maximum overshoot is less than 1%. Consequently, the fuzzy-logic rules do not change the initial settings.
Chapter 4. Expert Adaptive GPC Based on Laguerre-Filter Modelling

Figure 4.24: Expert control of an overdamped system with time delay

Example 2

Let the open-loop system be given by

\[ y(t) = y(t - 1) - 0.26y(t - 2) + u(t - 1) \]  \hspace{1cm} (4.271)

The open-loop system is underdamped. The expert system is provided with a rough estimate of the open-loop settling time. A search algorithm is implemented as shown in the last example. Consequently, the model is tuned such that the number of filters is 3 and the Laguerre-filter pole is at 1.47. The fuzzy-logic rules are used to tune the controller on-line. The tuning actions are shown in the table below.

<table>
<thead>
<tr>
<th>( t )</th>
<th>( 0 \leq t &lt; 90 )</th>
<th>( 90 \leq t &lt; 170 )</th>
<th>( 170 \leq t &lt; 250 )</th>
<th>( 250 \leq t &lt; 400 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n_2 )</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>( \beta )</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.08</td>
</tr>
</tbody>
</table>
Figure 4.25: Expert control of an underdamped system

Figure (4.25) shows the closed-loop response as well as the control signal. It is clear that the settling time and the rise time of the closed-loop system are successively improved. The final settling time is 4.0 seconds and the maximum overshoot is 6%.

4.4 Conclusion

The adaptive LAG-GPC has been implemented using fuzzy-logic and expert-system techniques. G2, an expert shell, has been used as a platform for the real-time implementation. The main features of the implementation have been:

- A user interface that allows a system-user interaction. The user has been given the capability to provide information and ask for explanations.

- A combination of depth-first and breadth-first techniques which has been used to tune the Laguerre-filter models.
• Linguistic rules that have been implemented on-line to monitor the closed-loop performance and tune the controller accordingly.
Chapter 5

Conclusions

5.1 The main results of the thesis

Motivated by the flexibility of the Laguerre-filter based models and the feasibility of the generalized predictive control algorithms, a GPC based on state space models is derived in Chapter 2. The control law has feedback and feedforward terms. Typical effect of the controller tuning parameters are shown by an illustrative example in Section 2.2.1. The case of a one-step control horizon is analyzed in details in Section 2.2.2. The conclusions of that analysis are:

- The proposed GPC can stabilize any sampled first-order continuous system.

- The GPC problem can be studied as a perturbation problem. For the case of a small perturbation, the closed-loop poles can be calculated as perturbed eigenvalues of the open-loop system. In case of a general perturbation, an upper bound on the perturbation norm is derived to insure the closed-loop stability.

- Assuming an open-loop stable system, the case of plant-model match is studied in Theorem 2.2. It is shown that there always exists a prediction horizon and a control-weighting factor such that the GPC scheme is stable. Asymptotically, as the perturbation becomes very small, \( n \) closed-loop poles approach the corresponding open-loop ones and one closed-loop pole approaches zero or one.
• Systems which have one unstable pole are studied in Theorem 2.3. As the prediction horizon approaches infinity, one closed-loop pole approaches unity, \((n - 1)\) closed-loop poles approach the corresponding open-loop stable poles, and one closed-loop pole approaches zero. If the open-loop system has \(m\) unstable poles such that \(|\lambda_i| > |\lambda_{i+1}|, i = 1, \ldots, m\), then the closed-loop poles \(\tilde{\lambda}_i, i = 1, \ldots, m\), are such that \(\tilde{\lambda}_1 \to 1\), and \(\tilde{\lambda}_i \to \lambda_i, i = 2, \ldots, m\).

• There is no guarantee that the GPC with one-step control horizon can stabilize a system which has one open-loop unstable pole.

• The plant-model mismatch case is studied in Section 2.4.1 using a functional analysis approach. Stability analysis is tackled in three steps. First, assuming direct state feedback, it is shown that stability conditions can be satisfied by increasing the prediction horizon. Second, assuming a disconnected feedback, it is shown that stability conditions of the state observer, embodied in the GPC scheme, can always be satisfied by increasing the prediction horizon. Third, the stability of the overall scheme is concluded.

• The robustness of the linear time-invariant GPC schemes in the presence of plant-model mismatch is studied in Section 2.4.2. The results give a solid motive to use Laguerre-filter models in the implementation. The resulting controller is called LAG-GPC.

Adaptive LAG-GPC is implemented in Chapter 3. The main conclusions are

• Assuming plant-model match, the adaptive LAG-GPC scheme is convergent and globally stable as shown in Theorem 3.3.

• Assuming a plant-model mismatch, the robustness condition of the adaptive LAG-GPC is insured by increasing the prediction horizon as shown in Theorem 3.4.
• The conclusions drawn for the adaptive LAG-GPC are equally valid for other predictive controllers such as the MAC.

• In contrast to the case of zero control weighting and single-stage prediction horizon, the adaptive LAG-GPC does not require the plant-model mismatch to satisfy an SPR condition.

• Laguerre-filter models are extended to control unstable systems.

• The implementation simplicity of the adaptive LAG-GPC carry over to SIMO and MIMO schemes. It is shown in Theorem 3.5 that there always exist prediction horizons such that the MIMO LAG-GPC is stable and tracks constant set-points with zero steady state error.

• It is shown in Theorem 3.6 that the MIMO adaptive LAG-GPC is convergent and globally stable.

• Comparing the adaptive LAG-GPC, MAC, and CARIMA-model based GPC, simulations show that the adaptive LAG-GPC is superior if the plant is well-damped, unmodelled dynamics are present and the dead time is unknown. For poorly damped systems, the CARIMA-model based GPC comes first and the LAG-GPC comes second. The MAC fails to control systems with zero damping using a reasonable model order.

The adaptive LAG-GPC is augmented with fuzzy-logic and expert-system techniques in Chapter 4. The main features of that implementation are

• A flexible, interactive user interface is implemented. Through that interface, the user understands how to use the system, understand the terminologies, gives information about the plant, and asks for interpretation of some decisions taken by the
expert system.

- The Laguerre-model parameters are tuned using AI search techniques. A breadth-first strategy is implemented to adjust the pole of a Laguerre-filter model, while, a depth-first strategy is implemented to adjust the number of filters used in the model.

- Linguistic rules are implemented using fuzzy-logic techniques to emulate a human expert while tuning the LAG-GPC.

5.2 Suggestions for further research

Throughout this thesis, an attempt has been made to understand and analyze one of the promising adaptive control techniques, namely, generalized predictive control. The goal has been to produce an adaptive controller that performs satisfactorily in practice and still has a sound theoretical basis. It has been our belief that the final implementation of the controller should be using expert system techniques.

Along the above lines, further research can be carried out. The following suggestions give some guidelines:

- The expert-system implementation can be extended to include other predictive controllers, e.g. MAC, DMC, and GPC.

- Meta rules should be implemented to give the user better understanding of the inference mechanism.

- The model adjustment and on-line tuning techniques should be modified to handle stochastic systems.
• The expert system should be applied to a pilot plant to find out its weaknesses and strengths. The final goal is to apply it to an industrial plant.
Bibliography


Bitmead, R., M. Gevers, and V. Wertz, (1989). Adaptation and Robustness in


Bibliography


Australia.


Bibliography


Zervos, C. and G. Dumont, (1988a). Deterministic Adaptive Control Based on


Appendix A

Building an Expert System

A.1 Definitions and characteristics

An expert system can be defined as a program that solves problems within a specific, limited domain that normally would require a human expert. This definition is wide and vague since some programs satisfy the definition but are not considered expert systems, e.g. a FFT program. An expert system can be better defined by its characteristics.

An important characteristic of expert systems is their capability to solve problems within limited application domains where conventional techniques are inadequate. This could be because the problem lacks a clear analytical algorithmic solution or the existing algorithm is computationally intractable. Another characteristic is that the domain knowledge is represented explicitly in an identifiable, separate part of the program. The inference engine runs the program by operating a set of rules on the knowledge base. This is in contrast to conventional programming where the domain knowledge is expressed as program statements. This transparent feature of the expert system makes it understandable to both the user and the developer. The explicit knowledge representation provides the bases for modularity. The knowledge base is built incrementally and can be expanded to more complex systems. Another aspect of expert systems is their ability to reason with uncertainty and explain their reasoning to the user.
A.2 Major stages of knowledge acquisition

Knowledge acquisition is the transfer and transformation of the problem solving expertise from some knowledge source to a program. There are five major stages to accomplish this task.

1. Identification stage: This involves identifying the sources of information, the problem characteristic, resources, and goals.

2. Conceptualizing stage: In this stage the concepts and relations mentioned during the identification stage are made explicit.

3. Formalization stage: The key concepts and relations are mapped into formal representations that can be utilized in the expert system.

4. Implementation stage: The domain knowledge, made explicit during the formalization stage, specifies the contents of the data structures, the inference rules and the control strategies. The ultimate product of this stage is a prototype expert system.

5. Testing stage: The prototype expert system is evaluated by trying it out on several examples. Weaknesses of the knowledge base and the inference structure are figured out. The expert system is revised to improve its performance.

A.3 Techniques of knowledge representation

A.3.1 Semantic networks

Semantic networks represent knowledge as a network of nodes. A node could represent the concept of objects, events, ideas, etc. Associative links represent the relations among
the nodes. Semantic networks represent the combination of a superclass-subclass hierarchy and the description of properties (attribute-value pairs). Superclass-subclass can be thought of as generalization versus specialization. Another aspect of the network formalism is the instance relation that associates a particular piece of knowledge with a class of which it is a member.

### A.3.2 Frame systems

The idea of frame systems, a variation of the semantic networks, was introduced in Minsky (1975). A frame system consists of three main blocks: frames, slots, and facets. A frame is the equivalent to a node in a semantic network, i.e., it represents the concept of objects, events, etc. A slot describes the properties or attributes of a certain frame. Facets describe the different slots, e.g., the value of the slot, the type of the slot value, the default value of the slot, etc. Frames can describe classes or individual instances. Frames can be organized in a superclass-subclass fashion. An important characteristic of semantic networks and Frame systems is that the subclasses can inherit the attributes of their superclasses besides their own.

### A.3.3 Object-oriented programming

The basic entity of object-oriented programming is the object which has a local state and a behavior. Objects are asked to perform operations by sending appropriate messages to them. Objects have associated with the procedures called "Methods" that respond to the messages. Message passing supports data abstraction and generic algorithms. A protocol, i.e., a set of messages, is defined which specifies the external behavior of the object. Objects are divided into classes and instances of classes. The classes build up a superclass-subclass hierarchy with inheritance. An overview of object-oriented programming is given in Stefik and Bobrow (1986).
A.3.4 Rule-based representation

The main parts of a rule based expert system are: database, rulebase and inference engine. The database is used to represent facts about the application domain. Data structures can be in the form of lists, as in OPS4 (Forgy 1979), object-oriented data, as in EMYCIN (Vanmodele 1981), or simple collections of variables that can take different values. The rulebase contains the rules of the system. A typical rule looks like "if < antecedent > then < consequent >". The inference engine applies the rules to the database according to some strategy. The most common strategies are forward chaining and backward chaining.
Appendix B

Fuzzy-Logic Definitions and Terminology

According to Zadeh (1984), fuzzy logic is a kind of logic which uses graded or qualified statements rather than ones that are strictly true or false. Fuzzy logic is a concept which is meant to bring the reasoning used by computers close to that used by people. This approach has three main distinct features, Zadeh (1973):

1. The use of linguistic variables: A linguistic variable is a variable whose values are sentences in a natural or an artificial language.

2. The characterization of the relations between variables by fuzzy conditional statements. Fuzzy conditional statements are expressions of the form “IF A THEN B” where A and B have fuzzy meaning, e.g. “IF $x$ is small THEN $y$ is large”

3. Characterization of complex relations by fuzzy algorithms. A fuzzy algorithm is an ordered sequence of instructions which may contain fuzzy assignments and conditional statements, e.g. “$x$ is very small” and “IF $x$ is very small THEN $y$ is large”.

In what follows, the definitions and properties of fuzzy sets, which are used in chapter 4, are summarized.

Definition B.1 The universe of discourse $U$ is a collection of objects denoted generically by $\{u\}$ where $u$ is a generic element in $U$. 

150
Definition B.2 A fuzzy subset $A$ of a universe of discourse $U$ is characterized by a membership function $\mu_A : U \rightarrow [0, 1]$. So, $\mu_A(u)$ represents the grade of membership of $u$ in $A$.

Definition B.3 The support of a fuzzy set $A$ is the crisp set of all $u \in U$ such that $\mu_A(u) > 0$. The element $u$ at which $\mu_A(u) = 0.5$ is called the crossover point. A fuzzy set whose support is a single point in $U$ with $\mu_A(u) = 1.0$ is referred to as a fuzzy singleton.

Definition B.4 A linguistic variable is characterized by a quintuple $(x, T(x), U, G, M)$ in which $x$ is the name of the variable; $T(x)$ is the term set of $x$, that is, the set of names of linguistic values of $x$ with each value being a fuzzy number defined on $U$; $G$ is a syntactic rule for generating the names of values of $x$; and $M$ is a semantic rule for associating with each value its meaning.

Definition B.5 Let $A$ and $B$ be two fuzzy sets in $U$ with membership functions $\mu_A$ and $\mu_B$, respectively. The set theoretic operations of union, intersection and complement for fuzzy set are defined as

- The membership function $\mu_A \cup B$ is pointwise defined for all $u \in U$ by
  \[ \mu_A \cup B(u) = \max\{ \mu_A(u), \mu_B(u) \} \]  
  (B.270)

- The membership function $\mu_A \cap B$ is pointwise defined for all $u \in U$ by
  \[ \mu_A \cap B(u) = \min\{ \mu_A(u), \mu_B(u) \} \]  
  (B.271)

- The membership function $\mu_A$ of the complement of a fuzzy set $A$ is pointwise defined for all $u \in U$ by
  \[ \mu_A(u) = 1 - \mu_A(u) \]  
  (B.272)
Appendix B. Fuzzy-Logic Definitions and Terminology

Definition B.6 If $A_1$, $A_2$, ..., $A_N$ are fuzzy sets in $U_1$, $U_2$, ..., $U_N$, respectively, the Cartesian product of $A_1, A_2, ..., A_N$ is a fuzzy set in the product space $U_1 \times U_2 \times \ldots \times U_N$ with membership function

$$\mu_{A_1 \times \ldots \times A_N}(u_1, \ldots, u_N) = \min\{\mu_{A_1}(u_1), \ldots, \mu_{A_N}(u_N)\} \quad (B.273)$$

A fuzzy relation $R$ from a set $U_1$ to a set $U_2$ is a fuzzy subset of the Cartesian product $U_1 \times U_2$.

Definition B.7 If $R$ and $S$ are fuzzy relations in $U \times V$ and $V \times W$, respectively, the composition of $R$ and $S$ is a fuzzy relation denoted by $R \circ S$ and is defined by

$$R \circ S = \{(u, w), \sup_v(\mu_R(u, v) \ast \mu_S(v, w))\}, u \in U, v \in V, w \in W\} \quad (B.274)$$

where $\ast$ is the minimum operator.