A Case Study of Fractional-order Control

Yousaf A. Alarfaj

MSc Electronics Engineering

A thesis submitted in partial fulfilment

for the degree of Doctor of Philosophy

Engineering Department

UK

March 2020

Abstract

This thesis concerns fractional-order (non-integer) methods for control system design. Although fractional-order calculus has a long history in mathematics and engineering, the uptake of relevant fractional-order concepts in control systems research has been relatively slow, and interest in the topic remains comparatively low—albeit with some important exceptions, as highlighted by the literature review of this thesis.

The first part of the thesis considers fractional-order methods for modelling and control in quite broad terms, before later focusing on one particular approach from the control systems literature, namely Fractional-order Generalised Predictive Control (FGPC). The FGPC approach is of particular interest here because of its relationship with the well-known, conventional control algorithm, namely Generalised Predictive Control (GPC). Both algorithms have a relatively straightforward implementation form, making them attractive to practitioners.

Hence, one contribution of the thesis is to use worked examples in MATLAB as an introduction to GPC and FGPC design methods, in part for tutorial reasons. More significantly, the thesis demonstrates how fractional-order methods are utilised to increase control design flexibility. In this regard, the thesis investigates both conventional GPC and FGPC methods using various simulation examples. The robustness of control systems is investigated via Monte Carlo simulation, with consideration of model mismatch and unmeasured disturbances. These results

are utilised to develop recommendations for how to optimise the extra design coefficients introduced in the fractional-order case.

The comparative study is extended to a laboratory example, namely the control of airflow in a 1 m by 2 m by 2 m forced ventilation environmental test chamber. To facilitate further uptake of FGPC methods in the future, the algorithms developed are prepared as a MATLAB toolbox, i.e. a collection of functions that calculate and implement the FGPC approach and subsequently measure the performance of the controller.

Declaration

I declare that this thesis is my own work, and has not been submitted in substantially the same form for the award of a higher degree elsewhere.

Yousaf Alarfaj

Date:.....

Signature:.....

Acknowledgements

This thesis is the result of a lot of hard work and effort and it would not have been possible to complete it without the help of a group of people. I would like to acknowledge and extend my heartfelt gratitude to the following people:

- I would like to express my appreciation to my mentor Prof. C. J. Taylor for his continuous support. It would have not been possible for me to conclude this research without his guidance and motivation. His time, patience, and efforts in helping me throughout the research are very much appreciated.
- Thanks to Dr Emma Wilson for her time and assistance in the laboratory during the practical testing.
- Thanks to my country Kuwait, the government of Kuwait, and my employers who granted study leave for me and gave me the chance to be here.
- A warm and special thanks to my beloved wife for her endless support and patience.
- Thanks to all my family and friends, who didn't stop believing in me, for their support and encouragement.

Contents

List of Figures

List of Tables

Abbreviations

FOC	Fractional-Order Control
FOPID	Fractional-order Proportional Integral Derivative
FGPC	Fractional-order Generalised Predictive Controller
PID	Proportional Integral Derivative
GPC	Generalised Predictive Controller
PI	Proportional Integral controller
PD	Proportional Derivative controller
MC	Monte Carlo
MPC	Model Predictive Controller
MAGLEV	Magnetic Levitation Train
CRONE	Command Robust d'Ordre Non-Entier
LTI	Linear Time-Invariant
TID	Tilted Integral Derivative
FOPDT	First-Order Plus Dead-Time

IAE	Integrated Absolute Error
PSO	Particle Swarm Optimisation
GA	Genetic Algorithm
FOMCON	Fractional-order Modelling & Controlling
FSST	Fractional-order discrete State-Space system Simulink Toolkit
TBMs	Tunnelling Boring Machines
AMESim	Adaptive Modelling Environment for Simulation
LQR	Linear Quadratic Regulator
MPHC	Model Predictive Heuristic Control
MAC	Model Algorithmic Control
DAC	Dynamic Matrix Control
DMC	Dynamic Matrix Control
CGPC	Continuous-time Generalised Predictive Control
GL	Grunwald-Letnikov
RL	Riemann-Liouville
TDC	True Digital Control
NMSS	Non-Minimum State Space
SVF	State Variable Feedback
RIV	Refined Instrumental Variable

SRIV	Simplified Refined Instrumental Variable
YIC	Young Identification Criterion

Chapter 1 Introduction

This thesis concerns fractional-order control (FOC) system design, with a focus on Fractional-order Generalised Predictive Control (FGPC). Although fractional-order calculus has a long history in mathematics and engineering, the uptake of relevant fractional-order concepts in control systems research has been relatively slow, and interest in the topic remains comparatively low—albeit with some important exceptions, as highlighted by the literature review of this thesis. In fact, one aim of the thesis is to use worked examples in MATLAB as a tutorial introduction to the topic for interested control engineers. Besides, the thesis aims to demonstrate how FOC methods can be utilised to increase the design flexibility of a well-known conventional control algorithm, namely Generalised Predictive Control (GPC). The thesis considers both conventional GPC and fractional-order GPC methods using simulation and laboratory examples. These results are used to make recommendations for how to utilise the extra design coefficients in the fractional-order case.

1.1 Motivation

There is no doubt that science is often developing and control theory is no exception. Recently, the control systems community has seen an increased interest in the field of fractional-order calculus, filling in the blanks towards developing differential equation (or difference equation in the discrete-time case) controllers in fractional-order. During the last decade, the fractional-order concept has appeared much more frequently in the control systems literature, in various

articles and conferences proceedings. This includes articles about controlling fractional-order plants using traditional integer-order controllers (e.g. Romero *et al.*, 2007, 2008), but with other articles introducing fractional-order controllers, such as Fractional-order Proportional Integral Derivative (FOPID) control (Podlubny, 1999) and FGPC (Romero *et al.*, 2010a, 2010b), building on conventional PID and GPC (Clarke *et al*. 1987a, 1987b) control respectively. An increasing number of articles present various comparisons, together with applications and tuning methods for FOC. Such developments have led to a new view of controlling plants, with additional control coefficients arising because of the fractional-order framework, i.e. additional parameters for tuning that potentially yield improved precision and robustness of the controller.

Every physical system has a desired output that can be achieved in more than one way. Take the analogy of building a car: it can be either a complex process or a simple one and both will lead to the desired output, building a car. A simple process will be buying a used car, which will be cheaper, but you will have no control over the car's specifications. A complex process involves going to a dealer and stating the exact specifications needed to the seller and waiting for the car to be built the exact way you desire. The output of both processes is the same; however, the difference is major. In the first case (the simple process), you have no control over the car's colour, the type of transmission, or how fast can it go and you may need to adjust a few things; however, your adjustments will be limited. The second case (the complex process) gives you more degrees of freedom to communicate with the seller and ask to have your exact requirements fulfilled. In the same manner, the conventional controller might get the job done, whereas the fractional-order controller offers the opportunity for a more precise result, with more degrees of freedom i.e. to adjust or tune the parameters and hence the response characteristics.

The main purpose of introducing a fractional-order version of an existing conventional controller is to achieve a better response (Chen *et al.*, 2009) (in terms of robustness, faster response, precise response, etc.) and ideally to reach peak performance for the system. To shed more light on this, the classical PID controller can be tuned utilising (at most) three tuning terms (assuming there is a solution) which are the proportional, the integral, and the derivative terms, whereas in the Fractional-order PID (FOPID) case we have five terms (see Chapter 2) that can potentially provide better performance for the system. This provides motivation to pursue research on FOC. To provide focus, however, the present thesis primarily concerns the FGPC approach, by means of a new simulation and laboratory study of both GPC and FGPC, comparing the results to illustrate the differences between the fractional-order and its counterpart, the integer-order controller.

1.2 Research Objectives

This thesis makes novel research contributions in four main areas, as follows:

- A specific, comparative simulation study of GPC and FGPC for several different scenarios, including model mismatch and the disturbance response, i.e. with the robustness of the controllers investigated using Monte Carlo (MC) simulation (Chapter 4). The aim here is to draw recommendations for how to design and apply FGPC methods to different types of plant. In addition, the methods are compared in terms of their eigenvalues or poles location on the unit circle (Chapter 5). These comparisons between GPC and FGPC aim to investigate differences between the two approaches and how fractional-order methods can benefit the design process.
- Application of FGPC to a laboratory example, namely the control of ventilation rate in an environmental test chamber. To the authors' knowledge, this represents one of the

first implementations of FGPC design to a physical system in a laboratory context (Chapter 5).

1.3 Thesis Outline

The rest of the thesis is organised as follows. Chapter 2 presents the literature review, which covers the fractional-order controller background, including the key mathematical concepts, as necessary to provide the foundation for the later topics in the thesis. Next, Chapter 3 revises the fundamentals of Model Predictive Control (MPC), especially the ubiquitous GPC algorithm, and builds on this to explain the FGPC approach. The aim is to provide a full understanding of the concept of FGPC, which is the core of later research.

Chapter 4 introduces several simulation examples and uses these to develop a systematic comparison between the conventional GPC approach and FGPC, i.e. in terms of the closed-loop responses for each plant. The criteria for comparison include, for example, the response time, overshoot, and Monte Carlo analysis for robustness. Subsequently, Chapter 5 illustrates how the two extra coefficients in the FGPC case affect the locations of the poles within the unit circle. Furthermore, a comparison is established between GPC and FGPC in terms of these eigenvalues. The penultimate chapter, Chapter 6 considers a practical laboratory application to test FGPC in comparison to GPC, using the forced ventilation chamber in the Engineering Department at Lancaster University (Taylor, 2004). The ventilation chamber is a nonlinear application, which is a good challenge for FGPC to demonstrate its effectiveness in comparison with GPC. Finally, Chapter 7 presents the conclusions, a discussion of the limitations of the research, and recommendations for future research.

1.4 Articles Arising

The following peer-reviewed conference articles have arisen as a result of the research described in this thesis:

- Y. Alarfaj, A practical example of fractional-order generalised predictive control: forced ventilation in a micro-climate test chamber, *IFAC-PapersOnLine* **52** (11), pp. 97-102, 2019 (DOI: 10.1016/j.ifacol.2019.09.124). This paper was presented at the *5th IFAC International Conference on Intelligent Control and Automation Science* (ICON), Belfast, UK. Nominated as one of the best articles at the conference, the present author received a 'Best Young Author' certificate. The article is based on Chapter 6 of this thesis.
- Y. Alarfaj and C. J. Taylor, Eigenvalue analysis and case study examples for fractional-order generalised predictive control, published by IEEE Xplore (DOI: 10.23919/IConAC.2019.8895068). Presented at the *25th International Conference on Automation and Computing* (ICAC), Lancaster, UK, 2019. The article is based on Chapters 4 and 5 of this thesis.

Chapter 2 Literature Review

The idea of fractional calculus was born in 1695 when G.W Leibniz suggested that there is a possibility for fractional-order differentiation (Caponetto *et al.*, 2010). Since then, the topic of fractional-order calculus has been drawing increasing attention, since it offers the possibility of representing the system more accurately without (or with minimal) approximation. Moreover, this approach is a suitable tool to analyse fractional dimension systems, with long-term "memory" and chaotic behaviour (Gutierrez *et al.*, 2010), and it is an advantage to model the behaviour of a process with the fractional-order as the response will include many values that have been neglected by integer-order due to approximations (Podlubny,1994).

The lack of solution methods for differential equations was the main reason for using integer order instead of the more general representation, namely the non-integer (fractional) order. Recently, there are numerous methods for the approximation of fractional-order derivative and integral calculus, which ease the handling of non-integer systems. Thus, FOC systems have become one of the hottest topics in control engineering (Ladaci and Bensafia, 2015; Malek *et al.*, 2013; Razminia *et al.*, 2013).

Some examples of recent fractional-order controllers that have achieved promising results are active FOC for the Magnetic Levitation Train (MAGLEV) suspension system (Yu *et al.*, 2015); the optimal design of a robust fractional-order flight control system (Kumar *et al.*, 2015); FOC of a hydraulic thrust system for a tunnelling boring machine (Fei *et al.*, 2013); fractional-order

human arm dynamics (Tejado *et al.*, 2013); and FOC of a Hexapod Robot (Silva *et al.*, 2004). These and other applications will be further discussed in section 2.7.

As a control engineer, building a better system is what matters. According to Monje *et al.* (2010), based on a comparison of multiple simulations, it has been shown that using the best available fractional-order controller yields better results, in terms of robustness, fast response and minimal overshoot, compared to using the best integer controller for the same system. Although fractional-order controllers will be widely accepted in the future, many reasons are holding back the replacement of integer controllers with non-integer controllers. One of these reasons is that the improvement of performance that non-integer controllers provide has not been fully characterised yet. Also, some additional functionalities that are well established for integer controllers (e.g. PID design) have not been fully developed for non-integer controllers yet (Chen *et al.*, 2004).

This chapter provides a comprehensive literature review of fractional-order controllers. The engineering literature about fractional-order methods is relatively sparse since engineers and researchers have shown great interest in the topic only recently. However, it is worth mentioning that the existing control literature tends to focus on FOPID methods rather than describing the fractional-order problem in general; thus, most of the sections within this chapter will also concern FOPID. The rest of the chapter is organised as follows. Section 2.1 discusses the history behind the topic. Section 2.2 discusses why we should consider the fractional-order of a system. Section 2.3 explains the fundamentals behind the fractional-order methods, including fractional-order calculus. Section 2.4 deals with PID tuning techniques. Section 2.5 briefly introduces the fractional-order generalised predictive controller. Section 2.6 presents some implementation and design tools for MATLAB and LabVIEW. Section 2.7 contains some real-life applications of fractional-order systems, to illustrate its effectiveness on the overall

performance of the control system. The chapter ends with the conclusions in section 2.8.

2.1 Historical background of Fractional-Order Control

The very first theoretical contributions to the field of fractional-order calculus were made by Euler and Lagrange in the 1800s, while Abel was the first to use the fractional-order calculus on an application in 1823. The first systematic studies have been done between 1900 and 1950 by Liouville, Riemann and Holmgren. The nth-order series has been defined by Liouville who has also expanded the functions in a series of exponentials. Riemann presented a different approach which involved a definite integral. After that, Grunwald and Krug unified the results of Liouville and Riemann (Oldham and Spanier, 2006; Miller and Ross, 1993).

The need for solving a major design problem of a feedback amplifier was the key step towards engineers introducing fractional-order calculus methods. Bode presented an elegant solution for design a feedback loop for the amplifier so that the performance of the closed-loop will resist the changes in the gain of the amplifier (Monje *et al.*, 2010). The solution was called "*ideal cut-off characteristic"* by Bode himself, which is known as "*Bode's ideal loop transfer function"* nowadays. The characteristic of this frequency is very useful in the robustness of the system to parameter changes or uncertainties.

The step Bode took has encouraged other engineers and curious mathematicians to adopt the concept of fractional-order and helped to motivate new contributions in FOC systems, including both theory and applications. Over the last decades of the twentieth century, there was a growth of the practical application of fractional calculus, mainly in the engineering fields of feedback control, signal processing and system theory.

Manabe (1961) introduced a new application of FOC. Oustaloup has studied the algorithms of FOC of the dynamic systems and developed a PID controller called “CRONE” (Command Robust d’Ordre Non-Entier) which means non-integer order robust control (Oustaloup, 1991). A generalisation of PID control has been presented by Podlubny (Podlubny, 1999). He was the first one to come up with the general form of $PI^{\lambda}D^{\mu}$, where the integrator and the differentiator come with the order of λ and μ, respectively. Also, Podlubny has demonstrated a comparison in terms of response between fractional-order PID against classical PID, as used to control fractional-order systems (Podlubny, 1999). In the next section, we shall understand why we need to use fractional-order controllers instead of conventional integer-order ones.

2.2 Benefits of using FOC (FOPID)

The most commonly used controller in the industry field is PID controller which is a special case of a more general form $PI^{\lambda}D^{\mu}$. The fractional-order PID controller is just an approach of the family of fractional-order controllers. This general form contains two more extra parameters (λ and μ) which, according to Faieghi and Nemati, (2011), adds more control reliability to the model. The transfer function of such a controller can be presented as:

$$G_c(s) = \frac{U(s)}{E(s)} = K_P + K_I\frac{1}{s^{\lambda}} + K_D s^{\mu}\ , (\lambda, \mu\ > 0) \qquad (2.1)$$

where $G_c(s)$ is the transfer function of the controller (the ratio of output and input of the controller, as U(s) and E(s) are the output of the controller and the error respectively.

The time-domain representation is as follows:

$$u(t) = K_p e(t) + T_i D_t^{-\lambda} e(t) + T_i D_t^{\mu} e(t). \qquad \left(D_t^{(*)} \equiv_0 D_t^{(*)}\right) \qquad (2.2)$$

It is known from basic control theory that feedback PID control systems will affect the controlled system behaviour in certain actions, which are Proportional, Derivative and Integral. The main effects on those parameters over the controlled system behaviour are (Astrom and Murray, 2008):

- Proportional parameter: Increase or decrease the response speed and decrease the steady-state error and relative stability by adjusting the value of the gain.
- Integral parameter: Decrease or increase the relative stability and eliminating the steady-state error.
- Derivative parameter: Increase or decrease sensitivity to noise and relative stability.

The fractional-order controller has 5 parameters to tune instead of 3 which provides more flexibility to the dynamic properties of the fractional-order system. According to Monje *et al.* (2005), the following parameters are noted to provide promising results when using fractional-order controllers:

i. No steady-state error.
ii. More robustness to variations in the gain of the plant.
iii. Better output disturbance rejection.
iv. More robustness to high-frequency noise

In the next section, the fundamentals of the fractional-order approach are presented.

2.3 Fundamentals of FOC and approaches for FOC

The idea of fractional-order calculus is as old as the integer (conventional) order calculus. This fact can be seen from the letter written by Leibniz to L'Hopital in 1695 (Xue *et al.*, 2006). Since

then, attention has been drawn to this topic by various mathematicians such as: Euler; Laplace; Fourier; Abel; Liouville; Riemann; and Laurent. Analytically, the fractional-order calculus is nothing but a generalisation representation of the integer order calculus:

$$D^{\alpha} = \begin{cases} \dfrac{d^{\alpha}}{dt^{\alpha}} & \alpha > 0, \\ 1 & \alpha = 0, \\ \displaystyle\int_{\alpha}^{t} (d\tau)^{-\alpha} & \alpha < 0 \end{cases} \tag{2.3}$$

with $\alpha \in \mathfrak{R}$.

- Although integer-order is sufficient to solve many engineering problems, many natural phenomena can be better described if fractional-order calculus is used. The reason for this is that fractional-order models can model higher order integer systems that FO can take into account larger periods of past behaviour and it is compact when expressing high-order dynamics (Xue *et al.*, 2006; Magin and Ovadia, 2006). According to Cafagna (2007) and Ortigueira *et al.* (2005), there are common definitions of fractional-order calculus that have been used in the literature which are mentioned below in equations (2.2-2.7). I have used the Grunwald-Letnikov definition, i.e. equation (2.4), in the later chapters of this thesis to define FGPC systems.

- Riemann-Liouville:

  Integral:

$$I^{a} f(t) = \frac{1}{\Gamma(a)} \int_{c}^{t} \frac{f(\tau)}{(t-\tau)^{1-a}} d\tau, \tag{2.4}$$

  Derivative:

$$D^a f(t) = \frac{d^m}{dt^m}\left[\frac{1}{\Gamma(m-a)}\int_0^t \frac{f(\tau)}{(t-\tau)^{a+1-m}}\,d\tau\right], \tag{2.5}$$

$$m \in \mathbb{Z}^+, m-1 < a \leq m,$$

- Grunwald-Letnikov:

  Integral:

$$D^{-a} = \lim_{h\to 0} h^a \sum_{i=0}^{(t-a)/h} \frac{\Gamma(a+i)}{i!\,\Gamma(a)} f(t-ih), \tag{2.6}$$

  Derivative:

$$D^{\alpha} f(t) = \lim_{h\to 0} h^{-\alpha} \sum_{i=0}^{\infty} (-1)^i \binom{\alpha}{i} f(t-ih) \tag{2.7}$$

- Caputo:

$$D_*^a f(t) = \frac{1}{\Gamma(m-a)}\int_0^t \frac{f^{(m)}(\tau)}{(t-\tau)^{a+1-m}}\,d\tau, \tag{2.8}$$

- Cauchy:

$$f_+^{(a)} = \int f\,(\tau)\frac{(t-\tau)^{-a-1}}{\Gamma(-\mathrm{a})}\,d\tau, \tag{2.9}$$

The notation "$\Gamma(a)$" is the generalisation of fractional function (Oldham and Spanier, 2006) which is defined as a restriction of x:

$$\Gamma(x) \equiv \int_0^{\infty} y^{x-1}e^{-y}dy, \quad x > 0 \tag{2.10}$$

Or with no restriction for x:

$$\boldsymbol{\Gamma}(x) \equiv \lim_{N\to\infty}\left[\frac{N!\,N^{x}}{x(x+1)(x+2)\cdots(x+N)}\right] \tag{2.11}$$

Depending on the designer and the application, any of those applications can be used.

As for the continuous-time dynamic system of non-integer order, it can be represented as follows (Monje *et al.*, 2010):

$$H(D^{a_0 a_1 a_2 \cdots a_m})(y_1, y_2, \cdots, y_l) = G\left(D^{\beta_0 \beta_1 \beta_2 \cdots \beta_n}\right)(u_1, u_2, \cdots, u_k), \tag{2.12}$$

As y_i, u_i are functions of time and $H(\cdot), G(\cdot)$ are the combination of laws of the non-integer derivative operator. For the single variable case of the Linear Time-Invariant (LTI), the following equations can be seen:

$$H(D^{a0a1a2\cdots an})y(t) = G\left(D^{\beta 0 \beta 1 \beta 2 \cdots \beta m}\right)u(t), \tag{2.13}$$

$$\downarrow$$

$$H(D^{a0a1a2\cdots an}) = \sum_{k=0}^{n} a_k D^{ak}; \quad G\left(D^{\beta 0 \beta 1 \beta 2 \cdots \beta m}\right) = \sum_{k=0}^{m} b_k D^{\beta m}, \tag{2.14}$$

where $a_k, b_k \in \Re$.

The system will be of *commensurate-order* if all the orders in equation (2.12) of derivation are integer multiples of a base order, a that is, $a_k, \beta_k = ka, a \epsilon \Re^{+}$, so that equation (2.12) will be:

$$\sum_{k=0}^{n} a_k D^{ak} y(t) = \sum_{k=0}^{m} b_k D^{\beta m} u(t), \tag{2.15}$$

As if $a = {}^{1}/_{q}$, $q \in \mathbb{z}^{+}$, then the system will be of *rational order.*

To sum up, for the LTI systems, it can be categorised as follows:

$$LTI\ Systems \begin{cases} Non-Integer \begin{cases} Commensurate & \begin{cases} Rational \\ Irrational \end{cases} \\ Non-Commensurate \end{cases} \\ Integer \end{cases}$$

The theory of stability states that the LTI system is stable if the characteristic polynomial's roots are negative or have negative real parts if they are complex conjugate. This means they are located in the left half of the complex plane. The stability of the same system in fraction order is different. A stable fractional system may have roots in the right half of the complex plane. Figure 2.1 will illustrate more about the stability region of fractional-order systems (Chen *et al.*, 2009).

While in the discrete-time systems, an approximation can be made for the definition of fractional-order operator defined by Grunwald–Letnikov (Monje *et al.*, 2010):

$$\begin{aligned} a_n \Delta_h^{a_n} y(t) + a_{n-1} \Delta_h^{a_{n-1}} y(t) + \cdots + a_0 \Delta_h^{a_0} y(t) \\ = b_m \Delta_h^{\beta_m} u(t) + b_{m-1} \Delta_h^{\beta_{m-1}} u(t) + \cdots + b_0 \Delta_h^{\beta_0} u(t). \end{aligned} \tag{2.16}$$

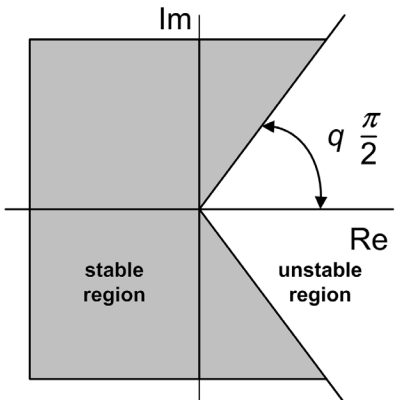


Figure 2.1: Stability region of LTI fractional-order systems with order 0<q≤1 (Monje et al., 2010).

By applying the Laplace transform to equation (2.14), the input-output representations of fractional-order systems can be found. Below is an illustration of both continuous and discrete-time transfer functions:

- The continuous-time transfer function form:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{a_n} + a_{n-1} s^{a_{n-1}} + \cdots + a_0 s^{a_0}} \tag{2.17}$$

- The discrete-time transfer function form:

$$G(z) = \frac{b_m(\omega(z^{-1}))^{\beta_m} + b_{m-1}(\omega(z^{-1}))^{\beta_{m-1}} + \cdots + b_0(\omega(z^{-1}))^{\beta_0}}{a_n(\omega(z^{-1}))^{n} + a_{n-1}(\omega(z^{-1}))^{n-1} + \cdots + a_0(\omega(z^{-1}))^{0}} \tag{2.18}$$

In the engineering industry, the focus is more on closed-loop control systems. There are four types of closed-loop control systems based on their order (integer or fraction) of its plant and its controller. The first type is to have both the plant and the controller in integer-order. The second type is to have integer order plant with the fractional-order controller. The third one is

to have a fractional-order plant with an integer-order controller. The final type is to have both the plant and the controller in fractional-order (Chen, 2006).

Some typical fractional-order controllers have been used since the early attempts of finding a solution for fractional-order systems; such as Tilted Integral Derivative (TID) and CRONE (Xue and Chen, 2002).

- TID controller is a feedback control system that is considered to be a compensator of the PID controller, where the compensator's proportional component is replaced with a tilted component with a transfer function $s^{-1/n}$. The entire compensator transfer function will more closely approximate an optimal loop transfer function which leads to improvement in the performance of feedback control, simpler tuning, better noise rejection and smaller effects on plant parameter variations on the closed-loop response compared to conventional PID (Chen *et al.*, 2009). Figure 2.2 below shows a block diagram of a system that uses the TID controller.

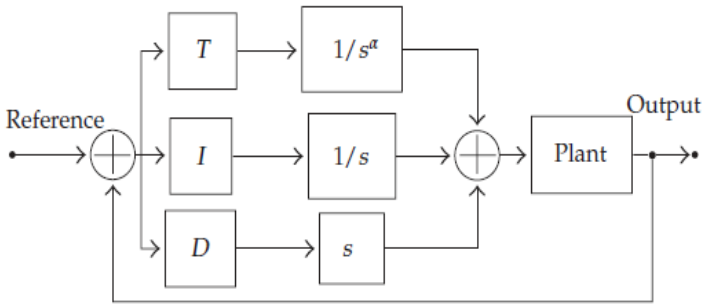


Figure 2.2: Reconstructed block diagram of a system that uses TID controller with $0 \leq \alpha \leq 1$

adopted from Xue and Chen (2002)

- CRONE control was first presented by Oustaloup who was seeking fractional robustness. The term "fractional robustness" is used to describe the iso-damping and

the vertical sliding form of frequency template in the Nicholas chart (Oustaloup, 1995). There are several real-life applications of CRONE controllers such as the flexible transmission and car suspension control (Oustaloup *et al.*, 1996). Figure 2.3 shows a block diagram of a system that uses the CRONE controller.

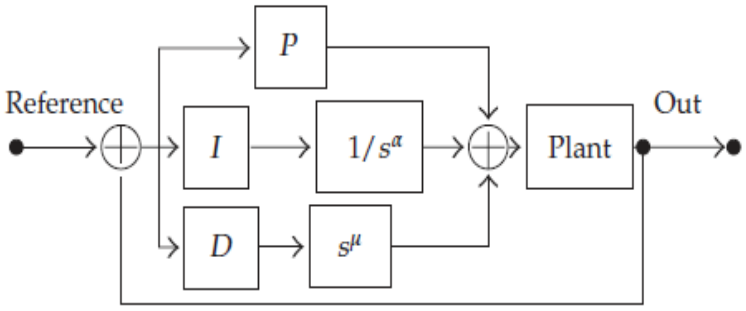


Figure 2.3: Reconstructed block diagram of a system that uses CRONE controller with $0 \leq \alpha \leq 1$ and $0 \leq \mu \leq 1$ adopted from Xue and Chen (2002)

In Moreau and Daou, (2014), a comparison between three types of controllers (integer-order PID, generalised $PI^{\lambda}D^{\mu}$, and CRONE) has been made. These controllers have been implemented to control a hydro electromechanical system and the observed output performances were promising. The experiment illustrates that the output simulations of the CRONE controller show more robustness than the other two controllers.

In the next section, I will address the different approaches for tuning FOC and as stated earlier, the literature focuses on FOPID which means that most of the approaches will be tuning FOPID.

2.4 Tuning methods and approaches of FOC

As interest in FOC has been increasing recently, researchers and engineers have put their efforts into create tuning methods that can be used to design the FOC based on various specific designs.

2.4.1 Tuning by minimization

Monje *et al.* (2004) proposed an optimisation method for tuning FOC to fulfil a certain desirable behaviour expected from the controller. There are six criteria which describe the desirable dynamics in this method:

1. No steady-state error

To achieve steady-state error cancellation, a fractional-order integrator of order $k + \lambda,\ k \in N,$ $0 < \lambda < 1$ is implemented which is as good as integer order integrator of order $k + 1$.

2. Gain cross-over frequency (ω_{cg}) specification

$$\left|G_c(j\omega_{cg})G(j\omega_{cg}\right| = 0dB \tag{2.19}$$

3. Phase margin (φ_m) specification

$$-\pi + \varphi_m = \arg\left(G_c(j\omega_{cg})G(j\omega_{cg})\right) \tag{2.20}$$

4. Gain margin (g_m) and phase crossover frequency (ω_{cp}) specifications

see Chen and Moore(2005) and Chen *et al.* (2004)

$$\left(\frac{d(Arg\left(C(j\omega)G(j\omega)\right))}{d\omega}\right)_{\omega=\omega_{cg}} = 0 \tag{2.21}$$

This condition will force the phase to be flat at ω_{cg} and constant within an interval around ω_{cg}, which immunises the system against changes, and the overshoot of the response within the interval is almost constant.

5. Robustness to variation in plant's gain

$$\left|T(j\omega) = \frac{G_c(j\omega)G(j\omega)}{1 + G_c(j\omega)G(j\omega)}\right| < AdB \forall \omega \geq \omega_t \rightarrow |T(j\omega)| = AdB \qquad (2.22)$$

where A is the desired noise attenuation for frequency $\omega \geq \omega_t \, rad/s$.

6. Robustness to noise in high frequency

$$\left|S(j\omega) = \frac{1}{1 + C(j\omega)G(j\omega)}\right|_{dB} \leq BdB, \qquad (2.23)$$

$$\forall \omega \leq \omega_s rad/s \rightarrow |S(j\omega_s)|_{dB} = BdB$$

where B is the desired value of the sensitivity function for frequency $\omega \geq \omega_s \, {rad}/{s}$.

Five of those six criteria can be achieved by the closed-loop system because the FOC has 5 parameters which can be tuned. The specifications from 2 to 6 (if achieved) can ensure a robust response of the controlled system against gain variation and noises, whereas the no steady-state error is achieved just with the introduction of integral action.

From these, constraints (2 to 6), 5 nonlinear equations with 5 unknown parameters ($K_p, \lambda, K_I, \mu, K_D$) are obtained. One of the proposed solutions for those equations is using the MATLAB function "*fmincon*" to find the optimised solution with minimum error. In Monje *et al.* (2004), the specification in criteria number 2 has been taken to be the main function to minimise, and the rest of the specifications (criteria 3-6) are used as constraints for the minimisation, which are all subject to the optimisation parameters defined within the MATLAB function "*fmincon"*.

This tuning technique has proven its effectiveness in practice; however, the major limitation of this technique is the dependence on the initial estimation of the parameters provided. If the initial estimation was not good enough then the solution may be unfeasible or lead to unstable loops. Only well-chosen initial estimations of the parameters will provide acceptable solutions.

2.4.2 Ziegler-Nichols type tuning rules

Valerio and Costa (2006) have been motivated by the drawbacks of the previous tuning technique which is the dependence on the initial estimations, and they introduced some Ziegler-Nichols-type tuning rules for FOPID. The tuning rules are only applicable for systems with an s-shape response for the step input. The simplest S-shaped response system for the step input is

$$G(s) = \frac{K}{Ts+1}e^{-sL} \tag{2.24}$$

With K=1 and several values of L and T, Valerio and Costa have employed the minimisation tuning technique on the plant and found that the parameters of FOPID obtained vary regularly. By formulating this regularity, some rules have been obtained for specific desired responses.

1. The first set of rules

The first set of rules is presented in Table 2.1 and Table 2.2. These are to be read as:

$P = -0.0048 + 0.2664L + 0.4982T + 0.0232L^2 - 0.0720T^2 - 0.0348TL$,.... and so on. They can be used if $0.1 \leq T \leq 50, L \leq 2$ and were designed as per the following specifications according to Valerio and Costa (2006):

$\omega_{cg} = 0.5rad/s, \varphi_m = \frac{2}{3}rad,\ \omega_t = 10rad/s,\ \omega_s = 0.01rad/s,\ A = -10dB, B = -20dB$

Table 2.1: Parameters FOPID for the first set of rules when 0.1≤T≤5 (Valerio and Costa, 2006)

	k_P	k_I	λ	k_D	μ
1	-0.0048	0.3254	1.5766	0.0662	0.8736
L	0.2664	0.2478	-0.2098	-0.2528	0.2746
T	0.4982	0.1429	-0.1313	0.1081	0.1489
L^2	0.0232	-0.1330	0.0713	0.0702	-0.1557
T^2	-0.0720	0.0258	0.0016	0.0328	-0.0250
LT	-0.0348	-0.0171	0.0114	0.2202	-0.0323

Table 2.2: Parameters FOPID for the first set of rules when 5 ≤ T ≤ 50 (Valerio and Costa, 2006)

	k_P	k_I	λ	k_D	μ
1	2.1187	-0.5201	1.0645	1.1421	1.2902
L	-3.5207	2.6643	-0.3268	-1.3707	-0.5371
T	-0.1563	0.3453	-0.0229	0.0357	-0.0381
L^2	1.5827	-1.0944	0.2018	0.5552	0.2208
T^2	0.0025	0.0002	0.0003	-0.0002	0.0007
LT	0.1824	-0.1054	0.0028	0.2630	-0.0014

2. The second set of rules

Table 2.3 shows the second set of rules which can be applied for $0.1 \leq T \leq 50 \; and \; L \leq 0.5$. As the range of values of L these rules cope with is reduced, only one set of parameters is needed. The rules were designed based on the following specifications:

$\omega_{cg} = 0.5 rad/s, \varphi_m = 1 \; rad, \; \omega_t = 10 rad/s, \; \omega_s = 0.01 rad/s, \; A = -20 dB, B = -20 dB$

Table 2.3: Parameters FOPID for the second set of rules (Valerio and Costa, 2006)

	k_P	k_I	λ	k_D	μ
1	-1.0574	0.6014	1.1851	0.8793	0.2778
L	24.5420	0.4025	-03464	-15.0846	-2.1522
T	0.3544	0.7921	-0.0492	-0.0771	0.0675
L^2	-46.7325	-0.4508	1.7317	28.0388	2.4387
T^2	-0.0021	0.0018	0.0006	-0.0000	-0.0013
LT	-0.3106	-1.2050	0.0380	1.6711	0.0021

2.4.3 The Padula and Visioli method

Another set of tuning rules has been presented by Padula & Visioli (2011) for FOC. The idea of this tuning technique has been conceived from a First-Order Plus Dead-Time (FOPDT) model by minimising the Integrated Absolute Error (IAE). This is achieved by applying a constraint on the maximum sensitivity.

To illustrate the concept of the technique, consider a system with a transfer function as mentioned in (2.24).

The plant dynamics can be characterised by the normalised dead time and represented as:

$$\tau = \frac{L}{L+T} \tag{2.25}$$

which shows a measure of difficulty in controlling the plant. The proposed tuning rules have been set for the normalised dead time $0.05 \leq \tau \leq 0.8$. Frankly, for $\tau < 0.05$ the dead time can be neglected and the controller design is rather trivial, whereas, for $\tau > 0.8$, the plant will be

significantly ruled by the dead time; thus dead time compensator must be employed. Padula and Visioli (2011) developed a method to model FOC by the following transfer function:

$$G_c(s) = K_P \frac{K_i s^\lambda + 1}{K_i s^\lambda} \frac{K_d s^\mu + 1}{\frac{K_d}{N} s^\mu + 1} \tag{2.26}$$

This FOPID transfer function has an additional first-order filter employed to make the controller proper and that is the major difference between this transfer function and the original FOPID transfer function. The N parameter is chosen to be $N = T^{(\mu-1)}$. The performance index is IAE which is defined as:

$$IAE = \int_0^\infty |e(t)| dt \tag{2.27}$$

Using this equation as a performance index will lead to low overshoot and faster settling time (Shinskey, 1994). According to Astrom and Hagglund (1995), maximum sensitivity is defined as:

$$M_s = max \left\{ \frac{1}{1 + G_c(s)G(s)} \right\} \tag{2.28}$$

This represents the inverse of the maximum distance of the Nyquist plot from the critical point (-1,0). The higher M_s, the less robustness against uncertainties. The tuning rules have been obtained based on the typical values of M_s=1.4 and M_s=2. If the only concern is load disturbance rejection then:

$$K_p = \frac{1}{K}(a\tau^b + c) \tag{2.29}$$

$$K_i = T\left(a\left(\frac{L}{T}\right)^b + c\right) \tag{2.30}$$

$$K_d = T\left(a\left(\frac{L}{T}\right)^b + c\right) \tag{2.31}$$

where the values of the parameters can be found in the following tables based on the value of M_s:

Table 2.4: Tuning rules for K_p,K_i,K_d,λ and μ when M_s = 1.4

	a	b	c
k_P	0.2776	-1.097	-0.1426
k_D	0.6241	0.5573	0.0442
k_I	0.4793	0.7469	-0.0239
λ		μ	
1		1.0 if $\tau < 0.1$	
		1.1 if $0.1 \leq \tau < 0.4$	
		1.2 if $0.4 \leq \tau$	

Table 2.5: Tuning rules for Kp,Ki,Kd,λ and μ when Ms = 2

	a	b	c
k_P	0.164	-1.449	-0.2108
k_D	0.6426	0.8069	0.0653
k_I	0.597	0.5568	0.0954
λ		μ	
1		1.0 if $\tau < 0.2$	
		1.1 if $0.2 \leq \tau < 0.6$	
		1.2 if $0.6 \leq \tau$	

2.4.4 Tuning using Particle Swarm Optimisation (PSO)

Particle Swarm Optimisation is widely used for tuning controllers. In Maiti *et al.* (2008), this tuning methodology has been introduced to be used in the FOPID controller. To illustrate its effectiveness in tuning FOC, Figure 2.4 below shows a simulation comparison between a conventional PID controller and a FOPID controller which have both been tuned using the PSO technique.

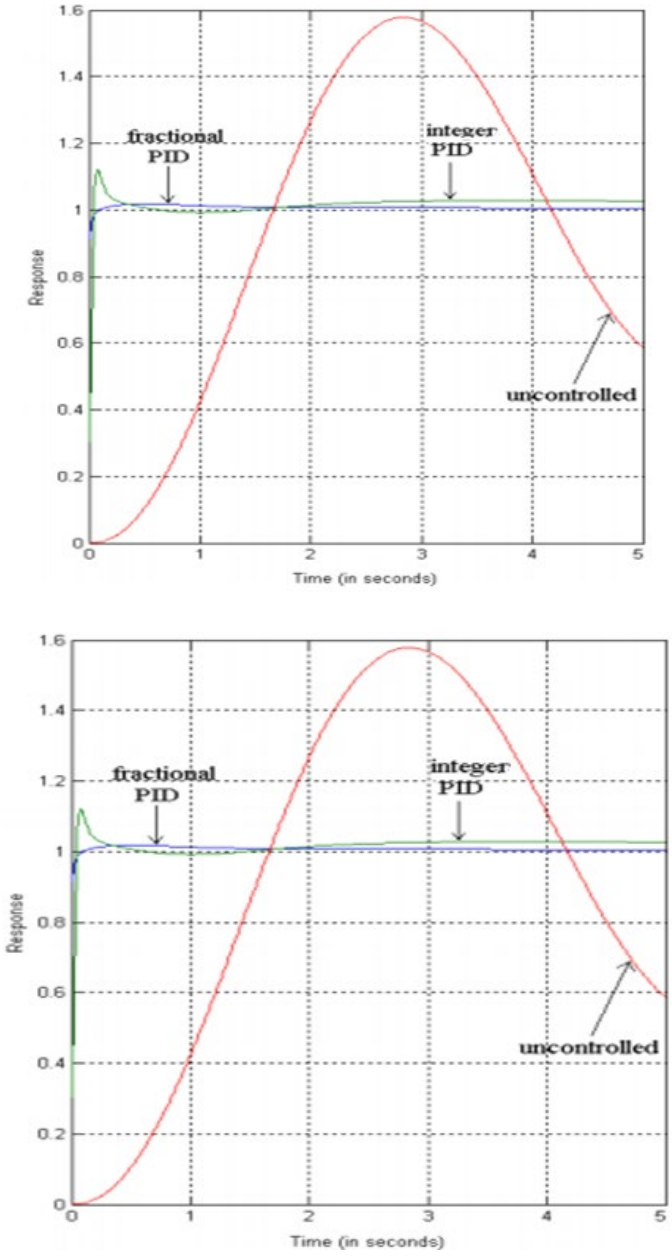


Figure 2.4: adapted from Maiti *et al.* (2008). The difference between FOPID and PID controller after using the PSO method for tuning

This simulation was the result of an illustrative example that is mentioned in Deepyaman *et al.* (2008). The example was for a fractional-order model with a transfer function as follows:

$$\frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1} \tag{2.32}$$

There were some specifications for designing a controller for this plant such as: $M_p = 10\%$ and $t_{raise} = 0.3\ sec.$

The same algorithm has been used by Bingul and Karahan (2011) to enhance the controlling of robot trajectory.

2.4.5 The graphical tuning method

In Zheng *et al.* (2014), a very recently developed graphical tuning method is introduced when the analytical model of the plant suffers from interval uncertainties. The solution proposed is to solve the problem of robustly stabilising an interval fractional-order plant using a fractional-order PID controller.

2.4.6 Tuning based on Genetic Algorithm (GA)

Zhang and Li (2011) have presented a new approach using a genetic algorithm to tune the FOPID controller which provides promising results in comparison to conventional PID.

2.5 Fractional-order GPC

There is an expanding enthusiasm for utilising fractional calculus applied to control theory to create a generalisation of the classical controller. Romero *et al.*, 2010a & 2010b) have used fractional-order operators and applied it to GPC and its cost function to derive a generalised form of GPC which is known as Fractional-order Generalised Predictive Control "FGPC".

Romero and his team have successfully generated a fractional-order cost function that can be seen as a potential generalisation of GPC with the use of fractional-order parameters that have added two extra coefficients that will be useful for tuning. More details and in-depth analysis of this will be discussed in Chapter 3.

2.6 Implementation and design of FOC

To digest and understand the concepts and basics of FOC systems, simulations and practical demonstrations are needed.

There are different methods for designing fractional-order controllers. In Dormido *et al.* (2012), two interactive tools have been presented to design non-integral controllers. The first one deals with the time and frequency domain of Fractional-order PID controllers, which gives the benefit for the user to identify the effects of changing user-chosen parameters. Both set-point and load disturbance step responses of the control system are shown in the time domain. Besides, the effect of measurements noise will be shown. The bode diagrams of all the critical closed-loop transfer functions are plotted in the frequency domain. The second tool will give the user the liberty to automatically determine the controller's parameters by applying a loop shaping technique, namely, by mapping a point in the process Nyquist plot to a target point of the loop transfer function Nyquist plot with a predefined value of its derivative (Dormido *et al.*, 2012).

There are a handful of simulation tools for the fractional-order which have various limitations. Good examples of these tools are Fractional-order Modelling & Controlling (FOMCON) for MATLAB and Simulink which is introduced by Aleksei (2012), and Fractional-order discrete State-Space system Simulink Toolkit (FSST) for MATLAB Simulink by Sierociuk (2010). This is considered to be the first toolkit that has a flexible optimisation tool that suits fractional-

order PID control design. Due to the nature of the fractional-order controllers (5 parameters to be adjusted), using this tool will give more strategies of tuning to provide more accuracy as illustrated in Figure 2.5. This tool is eligible to be used for both fractional and integer plants.

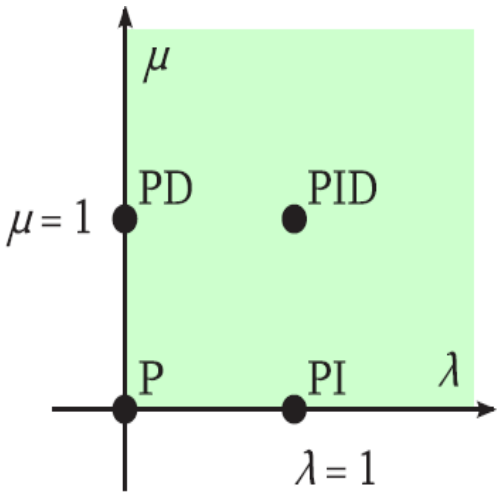


Figure 2.5: Adopted from Aleksei (2012), variety options of fractional-order PID coefficients

The second tool is FSST which is used for fractional-order discrete state-space which is a powerful toolkit that is compatible with MATLAB SIMULINK where the functions used are represented by blocks.

Other notable toolboxes used in MATLAB are: Ninteger and CRONE which are useful for designing fractional-order controllers (Aleksei, 2012).

In addition, fractional-order systems have been introduced using LabVIEW software in Jin *et al.* (2009) where the software has been used for a fast prototyping experimental setup to validate the fractional-order advantages over conventional integer-order in motion control systems.

2.7 Applications of FOC

Silva *et al.* (2004) presented a performance comparison between FOC and integer-order control for the joint leg control of a hexapod robot. This comparison is achieved by implementing joint leg actuator and transmission models that consolidate dynamical attributes to estimate how the controllers will respond to non-ideal joint actuators and transmissions. To analyse the system performance, several quantitative measures have been set on the dynamics and hip trajectory errors of the system. The analysis along with the experiment have illustrated that FOC (PD^{μ}) has a better response compared to the conventional PD controller in terms of robustness.

Another application of FOC is the electro-hydraulic controller for Tunnelling Boring Machines (TBMs) introduced by Fei *et al.* (2013). It's known that hydraulic systems have the characteristics of complexity, nonlinearity and variable loads, which make them good candidates for performing a comparison between the FOPID controller and conventional PID controller. Fei *et al.* (2013) introduced a comparison between the FOPID controller and PID controller to control pressure and flow of double shield TBM. The simulation models have been built using Adaptive Modelling Environment for Simulation (AMESim) and MATLAB/SIMULINK. Figure 2.6 shows the flow rate response for both FOPID and PID controllers.

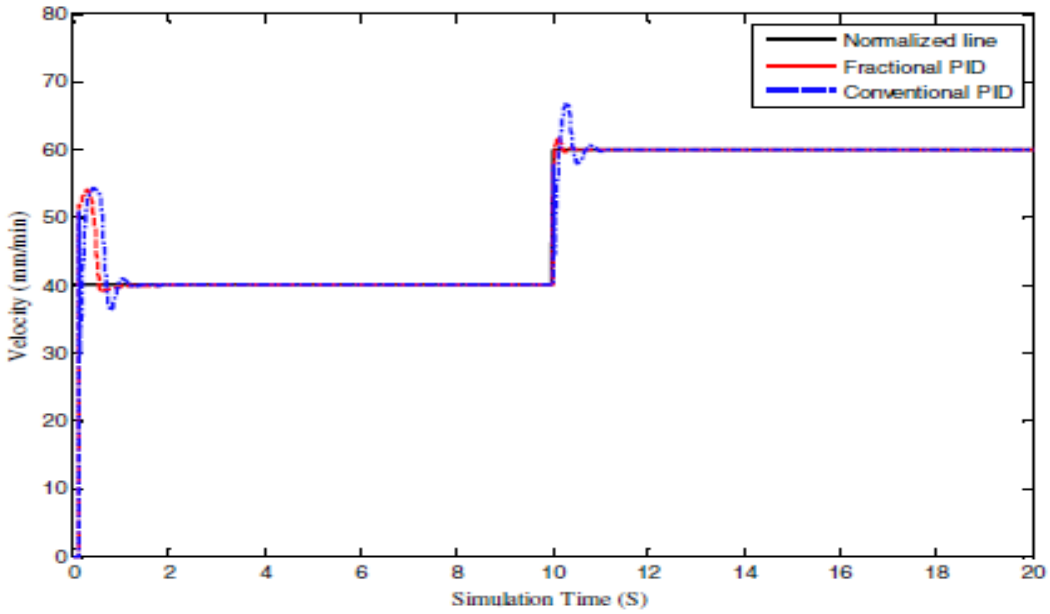


Figure 2.6: Adopted from Fei *et al.* (2013), flow rate response for FOPID and PID

After analysing the simulations, Fei *et al.* (2013) concluded that FOPID can make the controller parameters track the set values faster, steadily, and with less oscillation compared to the PID controller.

Fractional-order controllers have played a major role in energy conversion. According to Tejado *et al.* (2013), using FOC for wind turbines has improved the performance of disturbance attenuation and system robustness.

The MAGLEV train constitutes a fast-developing field around the world because of its advantages (lower noise, less maintenance cost, environment-friendly, and other features) which make it one of the best options available for urban transportation. Until now, two commercial routes and several test routes have been built around the world (Yu *et al.*, 2015). The suspension control system is one of the most important systems in the MAGLEV train, which is considered to be a nonlinear system in practice. Furthermore, the electromagnetic force produced by the constant current is inversely proportional to the square of the levitation.

Those facts will lead to an unstable system. It's a challenging task to design an ideal practicable control system which can satisfy the dynamic performance when the train is running under the actual environment. The authors (Yu *et al.*, 2015) have presented a closed-loop fractional-order PID controller and compared this with an integer order closed-loop Linear Quadratic Regulator (LQR) controller. The simulation comparison is shown in Figure 2.7 and 2.8 below. The fractional-order controller has shown better dynamic performance and robustness.

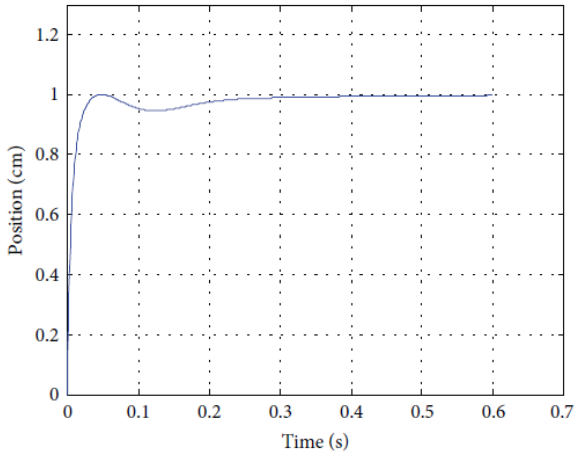


Figure 2.7: Adopted from Yu *et al.* (2015) Step response of the closed-loop system with fractional-order PID

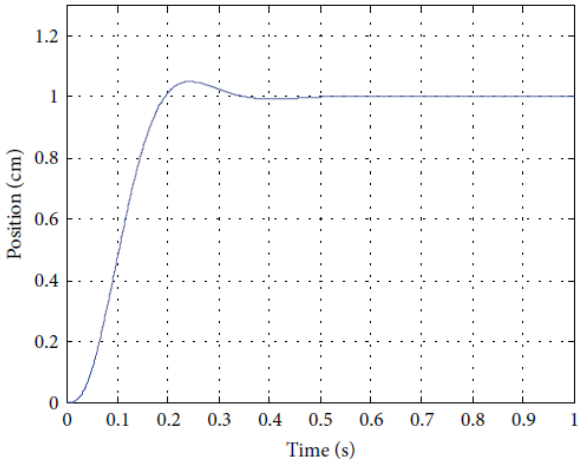


Figure 2.8: Adopted from Yu *et al.* (2015) Step response of the closed-loop system with a conventional PID controller.

Kumar *et al.* (2014) have introduced FOC on one of the benchmark application tests in the control engineering which is the inverted pendulum. The authors have validated the advantages of the fractional-order PID controller over the conventional PID controller by conducting simulations to illustrate the enhancements that fractional-order PID has achieved in the sense of robustness.

2.8 Concluding Remarks

This chapter has discussed the historical background of FOC, the features and advantages of using FOC, and has highlighted some of the design methodologies and tuning techniques available in the literature, with a focus on FOPID control. Furthermore, several Toolkits have been suggested for implementing the fractional-order controllers for design and simulation purposes. The literature about FOC is relatively small but growing fast recently with a narrow focus on fractional-order PID. Nonetheless, this literature suggests that fractional-order methods (in particular FOPID) have promising results compared to conventional PID, especially in robustness, noise rejection, and in reaching the desired output faster with minimum overshoot. These all provide good motivation for further research into fractional order methods. The next chapter will focus on the FGPC approach selected for study in this thesis. In addition, Chapter 3 will introduce the new FGPC MATLAB platform program which is created by the present author and a numerical worked example that illustrates the use of this toolbox.

Chapter 3 FGPC Fundamentals

Chapter 3 introduces the fundamentals of Fractional-order Generalised Predictive Control (FGPC), which is the method chosen as the focus of the simulation and application study in this thesis (see Chapter 1). It will illustrate the basis on which FGPC is built, the mathematical derivation of FGPC and the key differences between FGPC and conventional Generalised Predictive Control (GPC). To provide background content, the chapter first considers the Model-based Predictive Controller or Model Predictive Control (MPC) in general terms.

Sections 3.1, 3.2, and 3.3 discuss MPC, GPC, and FGPC, respectively. This is followed by a brief review of the differences between GPC and FGPC (section 3.4), the MATLAB functions and worked example (section 3.5), and concluding remarks (section 3.6).

3.1 MPC Review

The philosophy of MPC is to take advantage of the system model's current state measurements or estimates to predict the future behaviour of the system by minimising a given cost function. MPC is one of the most widely known aspects of modern control theory in both academia and industry. Thanks to its receding horizon implementation, it offers a practical compromise between optimality and fast calculation (Bitmead, 1990).

Historically, by the end of the 1970s, a growing interest in predictive control was noticed especially in the industrial field. This interest was reflected by many articles that contributed to the MPC family, for instance, Richlet and his team (1976; 1978) presented Model Predictive Heuristic Control (MPHC) which became known as Model Algorithmic Control (MAC) later on. In addition, Cutler and Ramaker (1980) introduced Dynamic Matrix Control (DAC).

3.1.1 MPC Elements

All model predictive controllers have common elements which are the basis of any MPC. These elements have different options to choose from, which constitute different algorithms (Camacho & Bordons, 2004). These elements can be summarised as follows:

- Predictive model
- Objective function
- Obtaining the control law.

The basic idea of the predictive controllers is at each instant of time "present state" (t), the future process $(t+k|t)$ is predicted within a time window defined by $k = 1,2,3,\dots,N$ using the plant model of this controller.

The notation used here $(t+k|t)$, indicates the value of the variable calculated in the future time and N is the forecasting horizon (De Keyser, 1992; Camacho & Bordones, 2004). This can be seen graphically in Figure 3.1. The structure block diagram of the MPC can be found in Figure 3.2.

This approach also defines a reference trajectory, $r(t+k|t)$. The trajectory describes the path of the process output from its current value at the point (t) towards the future point, $w(t+k|t)$, throughout the horizon prediction within the time window defined by k.

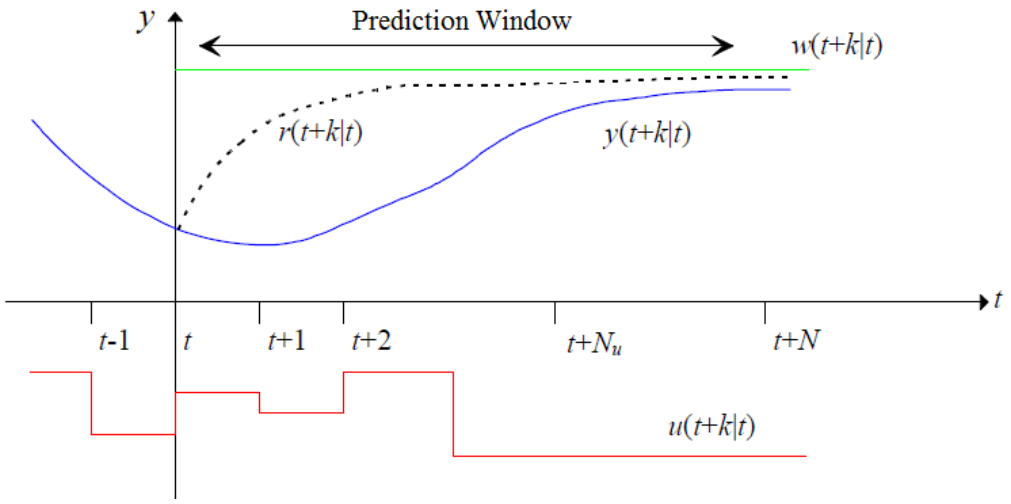


Figure 3.1: Model-based prediction methodology

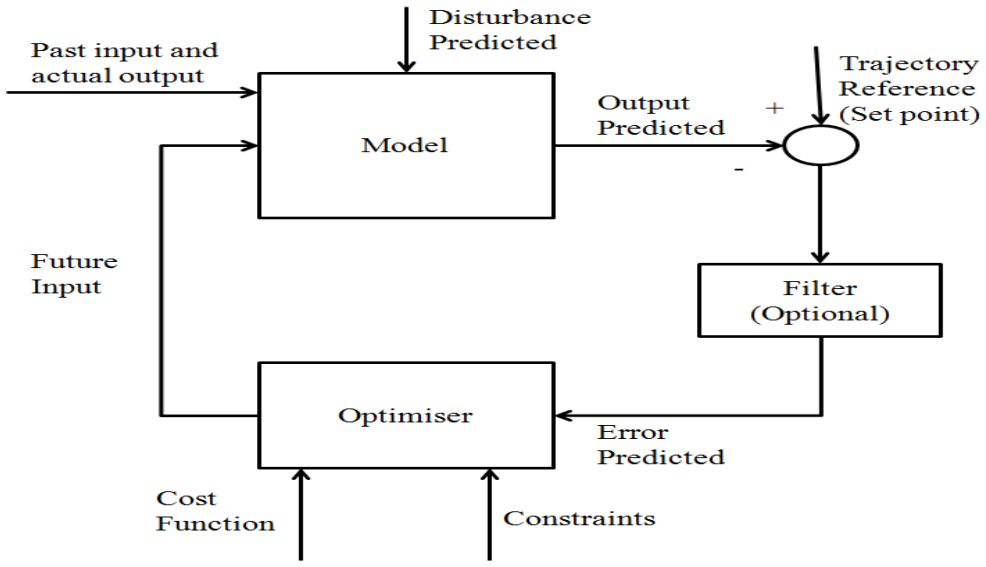


Figure 3.2: Model-Based Predictive Controller block diagram structure

Future input is calculated based on the minimisation of a determined objective function, known as the cost function, which relies on errors to predict the future.

3.1.2 MPC cost function

MPC algorithms utilise distinctive cost functions to obtain control. The basic principle is to contain the errors between the prediction of the process and the reference trajectory, on one hand, and the control, on the other hand. As we are focusing on GPC, we will consider the GPC cost function (Clarke *et al.,* 1987a):

$$J(\Delta u, t) = E\left\{\sum_{j=N_1}^{N_2} \gamma(j)[r(t+j|t) - y(t+j|t)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t+j-1|t)]^2\right\} \quad (3.1)$$

where:

- E{-} indicates the expectation operator
- $r(t+j|t)$ is the future reference trajectory
- $y(t+j|t)$ is the optimal prediction of output j steps forward, calculated using data known at time t.
- $\Delta u(t+j|t)$ is the increment of the control signal calculated with predictions made at time t, where Δ indicates $1 - z^{-1}$.
- N_1 and N_2 are the lower and upper-cost horizon respectively. These parameters are used to define the number of predictions made (N), where: $N = N_2 - N_1 + 1$.
- N_u is the control horizon. It is determined by $N_2 \leq N_u$ and it quantifies the number of degrees of freedom of the control signal. Also, this parameter influences the controller behaviour, as a larger value of N_u will result in more aggressive control which, in some cases, can destabilise the system.
- $\gamma(j)$ represents the future error weighting sequences.

- $\lambda(j)$ represents control weighting sequences.
- The notation $x(j|t)$ represents the predicted value of $x(j)$ at time instant t, where "x" represents any variable (for instance y or λ).

3.1.3 Prediction equation

The predicted output of the process is the sum of two effects, as shown in Figure 3.3 and described in the following expression:

$$y(t+k|t) = y_c(t+k|t) + y_f(t+k|t) \tag{3.2}$$

where

- y_c is the controlled response which depends on future control actions $u(t+k|t)$, which is to be determined.
- y_f is the free-response, assuming no future control actions and, therefore, the control remains at the value it has at time t (hence y_0).

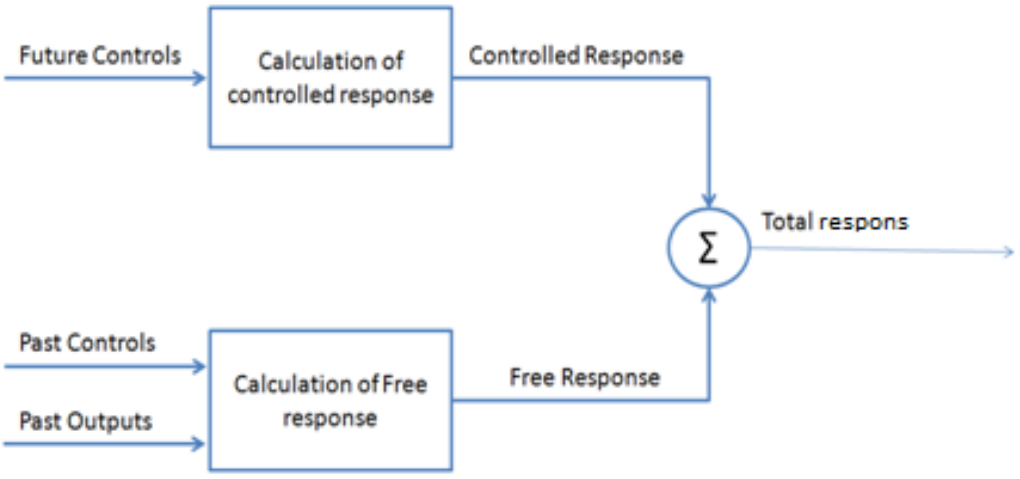


Figure 3.3: Effects that contributes to the total process response

It is essential to note that the prediction equations are formulated usually in terms of $\Delta u(t+k|t)$ due to the nature of the disturbance models used.

According to De Keyser, (1992) and Camacho, (2004), the general case for the prediction equation takes the form:

$$\bar{y}(t) = \overline{y_f}(t) + G\overline{\Delta u}(t) \tag{3.3}$$

Equation (3.3) can be extended to be:

$$\bar{y}(t+k|t) = \overline{y_f}(t+k|t) + \sum_{i=1}^{N} g_i \overline{\Delta u}(t+k-i|t) \tag{3.4}$$

where

- $\bar{y}(t)$ is the general matrix of the output $y(t)$
- G is a matrix containing the coefficients g_i of the step response process

$$G = \begin{bmatrix} g_1 & 0 & \cdots & 0 \\ g_2 & g_1 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ g_N & g_{N-1} & \cdots & g_{N-N_u+1} \end{bmatrix}$$

- g_i are the sampled output values for the step input, as shown in Figure 3.4
- $\overline{\Delta u}(t)$ is the general matrix of the control signal
- $\overline{y_f}(t)$ is the general matrix of the free response y_f (i.e y_0)

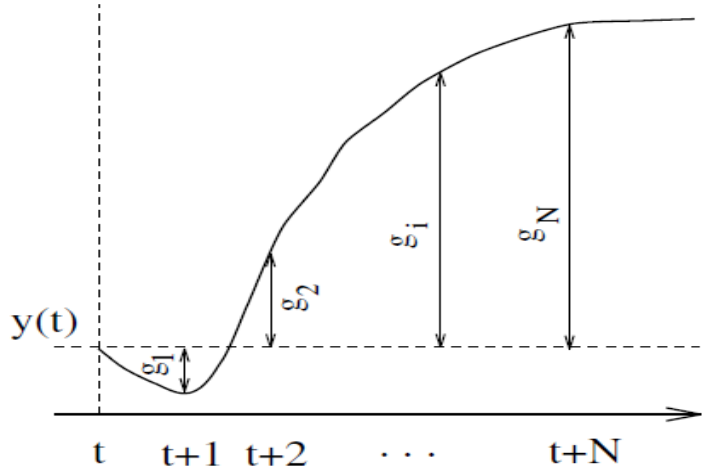


Figure 3.4: Adopted from Camacho (2004), step response

3.1.4 Obtaining the control law

To obtain the control low, we need to define the error vector first:

$$\bar{E} = [e(t+1|t), e(t+2|t), e(t+3|t), \dots, e(t+N|t)\ ]' \tag{3.5}$$

where

$$e(t+k|t) = r(t+k|t) - y(t+k|t) \tag{3.6}$$

Substituting equation (3.4) into equation (3.6):

$$e(t+k|t) = r(t+k|t) - [\overline{y_f}(t+k|t) + \sum_{i=1}^{N} g_i \overline{\Delta u}(t+k-i|t)] \tag{3.7}$$

Rearranging,

$$e(t+k|t) = \left[r(t+k|t) - \overline{y_f}(t+k|t)\right] - G\overline{\Delta u}(t) \tag{3.8}$$

As per equation (3.7) and equation (3.6),

$$\bar{E} = \overline{E_0} - G\overline{\Delta u}(t) \tag{3.9}$$

Noting that:

$$\overline{E_0} = \big[r(t+1|t) - y_f(t+1|t), r(t+2|t) - y_f(t+2|t), \dots, r(t+N|t) - y_f(t+N|t)\big] \tag{3.10}$$

And the incremental control law has the following expression:

$$\overline{\Delta u}(t) = [\Delta u(t|t), \Delta u(t+1|t), \Delta u(t+2|t), \dots, \Delta u(t+N-1|t)]' \tag{3.11}$$

Now substituting into the cost function, equation (3.1), to obtain the general form of the cost function:

$$J(\overline{\Delta u}, t) = \overline{\Delta u}'[G'\Gamma G + \Lambda]\overline{\Delta u} - 2\overline{E_0}'\Gamma G\overline{\Delta u} + \overline{E_0}'\Gamma\overline{E_0} \tag{3.12}$$

where

- Γ is a diagonal matrix containing weight factor $\gamma(j)$ in the multivariate general case.
- Λ is a diagonal matrix containing weight factor $\lambda(j)$ in the general case.

The sequence of optimal control, $\overline{\Delta u}^*$, is obtained by minimising the general cost function (i.e equation (3.11)):

$$\overline{\Delta u}^* = argmin_{\overline{\Delta u}}(J(\overline{\Delta u}, t)) \tag{3.13}$$

where

- "arg" is a mathematical function stands for "argument" which operates on a complex number.

- "min" is a mathematical function stands for "minimum" which is used to return the minimum of a function.

In this minimisation process, it is normally assumed that the control signal, $u(t)$, remains constant from time $t + N_u$.

According to De Keyser (1991), it is possible to analytically obtain the optimal control sequence if the following conditions apply:

- The cost function (J) is quadratic for the independent variables.
- The model equations are linear.
- No constrains on the control.

$$\overline{\Delta u}^* = (G'\Gamma G + \Lambda)^{-1} G'\Gamma \overline{E_0} = K\overline{E_0} \tag{3.14}$$

where $K = (G'\Gamma G + \Lambda)^{-1} G'\Gamma$.

Equation (3.14) represents the time-invariant linear control law, formed by controller matrix gain "K" and multiplied by the vector E_0. Since K is invariant over time, it can be calculated offline, and it will remain constant as long as there are no changes to the system.

3.2 Generalised Predictive Controller Review

Generalised predictive control is a member of the MPC family, which is known as GPC, and developed in the mid-1980s, within the academic field, by D.W Clarke (Clarke *et al*., 1987a,b). GPC shares many aspects with an earlier version of predictive control known as Dynamic Matrix Control (DMC) (Cutler & Ramaker, 1979). However, DMC has a completely deterministic formulation and therefore does not explicitly include any model of disturbances (Morari & Lee, 1997). On the other hand, stochastic aspects play an important role in GPC.

The vast majority of predictive control algorithms have been developed according to discrete methodology; however, there are continuous implementations of predictive control, e.g. Continuous-time Generalised Predictive Control (CGPC) (Demircioğlu & Gawthrop, 1991), which is a continuous interpretation of the discrete GPC algorithm. However, in this chapter, I will be using discrete-time only.

3.2.1 Prediction equation

The formulation of GPC is based on a stochastic model known as ARIMAX or CARIMA. The abbreviation ARIMAX is a special case of ARIMA which stands for Autoregressive Integrated Moving Average. When other time series included as input variables to ARIMA, then it is known as ARIMAX. As for CARIMA, it stands for Controlled Auto-Regressive Integrated Moving Average. When using these models, the mathematical model will be as follows:

$$A(z^{-1})y(k) = B(z^{-1})u(t) + \frac{T(z^{-1})}{\Delta}\xi(k) \tag{3.15}$$

where

- $A(z^{-1})$ is a polynomial that is defined appropriately which represents the discrete expression of the denominator of the transfer function of the plant. The general form of the polynomial can be:

$$1 + a_1 z^{-1} + \cdots + a_n z^{-n}$$

- $B(z^{-1})$ is a polynomial that is defined appropriately which represents the discrete expression of the numerator of the transfer function of the plant. The general form of the polynomial can be:

$$b_1 z^{-1} + \cdots + b_m z^{-m}$$

- $T(z^{-1})$ is a polynomial that is defined appropriately which represents the pre-filter used to improve the robustness of the system by rejecting noise and disturbance.
- Δ is the difference operator and it is defined by $1 - z^{-1}$.
- $\xi(k)$ is the uncorrelated zero-mean white noise (unmeasurable disturbance)

The predictor is obtained by solving the following Diophantine equation:

$$T(z^{-1}) = E_j(z^{-1})\Delta A(z^{-1}) + z^{-1}F_j(z^{-1}) \tag{3.16}$$

This leads to the following prediction equation:

$$y(t+j|t) = \frac{F_j}{T}y(t) + \frac{E_j B}{T}\Delta u(t+j|t) \tag{3.17}$$

The expression in equation (3.17) is a function of signals of known values at time t also, the function of values of the future control values that have to be determined. To distinguish the past and future controls, a second Diophantine equation is proposed:

$$E_j(z^{-1})B(z^{-1}) = G_j(z^{-1})T(z^{-1}) + z^{-j}\Phi_j(z^{-1}) \tag{3.18}$$

which leads to the following prediction equation:

$$y(t+j|t) = G_j\Delta u(t+j|t) + \Phi_j u^f(t) + F_j y^f(t) \tag{3.19}$$

where

- the polynomial G_j contains the first j coefficient of the step response to the plant $B(z^{-1})/\Delta A(z^{-1})$. For further details, refer to Bitmead *et al.* (1990)
- $u^f(t)$ is $\Delta u(t)$ filtered by $T(z^{-1})$
- $y^f(t)$ is $y(t)$ filtered by $T(z^{-1})$

$T(z^{-1})$ has known values at the instant time which can be represented by the following expression:

$$\begin{cases} u^f(t) = T^{-1}(z^{-1})\Delta u(t) \\ y^f(t) = T^{-1}(z^{-1})y(t) \end{cases} \tag{3.20}$$

From equation (3.19), it easily verified that the free response of the system is given by:

$$y_f(t+j|t) = \Phi_j u^f(t) + F_j y^f(t) \tag{3.21}$$

where

- $y_f(t+j|t)$ is the free-response on equation (3.19)

which supports the general expression in equation (3.2).

3.2.2 Controller Adjustment recommendation

Horizons and factors of weight play a major role in shaping the characteristics of the operation of the controller and cover a wide range of possibilities (refer to equation (3.1)). These parameters have a decisive influence on the behaviour and stability of the closed-loop controller, so their proper choice is of vital importance. Therefore, a set of standard "default" settings, which have proven to be suitable for most processes, have been recommended (Clarke *et al*., 1987b):

- N_1: If system delay "d" is known, then $N_1 \geq d$ must be taken to avoid superfluous calculations. If it is unknown or variable, then the value $N_1 = 1$ is valid, defining the model so that it can encompass the maximum delay of the process.
- N_2: If the sampling period is adequate, a value of 10 is usually sufficient.
- N_u: Must be taken equal to or greater than the number of unstable or poorly damped poles of the process. For most industrial processes, and generally for stable open-loop processes, a value of $N_u = 1$ usually produces an acceptable control action.
- λ: Values other than 0 that contribute to improving the robustness of the optimization algorithm (Clarke & Mohtadi, 1989). In general, small values, of the order of 10^{-6}, are sufficient unless the application requires greater damping of the control signal (for example, in the identification phase of the models and the tuning of the controller)
- γ: It is usually 1, although it can take different values depending on the type of application.

3.2.3 Closed-loop GPC

In the case where there are no active constraints, the control law will be linear time-invariant; therefore, a transfer function can be obtained. The closed-loop control system is shown in Figure 3.5 below, where R, S, and T are specific polynomials that define the controller, known as forward path, feedback path, and prefilter polynomial.

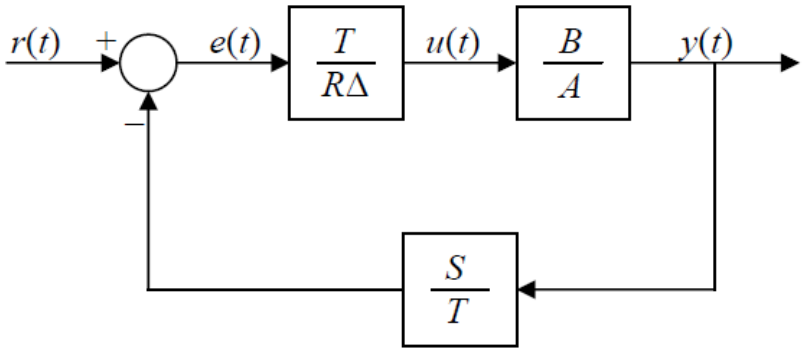


Figure 3.5: General diagram of the control loop

According to the above Figure, the closed-loop discrete transfer function is:

$$\frac{y(t)}{r(t)} = \frac{TB}{R(z^{-1})\Delta A + S(z^{-1})B} \tag{3.22}$$

And the characteristic equation of the system, obtained from equation (3.22), is:

$$R(z^{-1})\Delta A + S(z^{-1})B = 0 \tag{3.23}$$

From Figure 3.5, it is obvious to establish the following:

$$R(z^{-1})\Delta u(t) = Tr(k) - S(z^{-1})y(k) \tag{3.24}$$

To define the controller polynomials R, S and T, we need to define equation (3.24) in the same way we defined equation (3.14):

$$k_1 = [1\ 0 \dots 0]K \tag{3.25}$$

As we take the upper row (k_1) of the matrix K.

The first component of the optimal controller vector is:

$$\Delta u^*(t|t) = k_1 \bar{E}_0(1) = k_1 \left(w(1) - y_f(1) \right) \tag{3.26}$$

Substituting the free-response expression of GPC (y_f), which is given in equation (3.21):

$$\Delta u = k_1 (r - \Phi u^f - F y^f) \tag{3.27}$$

where Φ and F are matrices formed by the polynomials Φ_j and F_j respectively. Applying the definition of u^f and y^f according to equation (3.20):

$$\begin{aligned} \Delta u &= k_1 (r - \Phi T^{-1} \Delta u - F T^{-1} y) \\ &= \sum_{i=N_1}^{N_2} k_{1i} r(t+i) - \sum_{i=N_1}^{N_2} k_{1i} \frac{\Phi_i}{T} \Delta u(t) - \sum_{i=N_1}^{N_2} k_{1i} \frac{F_i}{T} y(t) \end{aligned} \tag{3.28}$$

This leads to the following equation:

$$\left(T + \sum_{i=N_1}^{N_2} k_i \Phi_i \right) \Delta u(t) = \left(T \sum_{i=N_1}^{N_2} k_i z^{-N_2+i} \right) r(t+N_2) - \left(\sum_{i=N_1}^{N_2} k_i F_i \right) y(t) \tag{3.29}$$

Comparing with equation (3.24) to obtain $R(z^{-1})$ and $S(z^{-1})$:

$$R(z^{-1}) = \frac{T + \sum_{i=N_1}^{N_2} k_{1i} \Phi_i}{\sum_{i=N_1}^{N_2} k_{1i} z^{-N_2+i}} \ , \qquad S(z^{-1}) = \frac{\sum_{i=N_1}^{N_2} k_{1i} F_i}{\sum_{i=N_1}^{N_2} k_{1i} z^{-N_2+i}} \tag{3.30}$$

It can be concluded from equation (3.15) and equation (3.24) that the closed-loop transfer function can be formed as follow:

$$(A\Delta R(z^{-1}) + BS(z^{-1}))y(t) = BTr(t + N_2) + TR(z^{-1})\xi(t) \quad (3.31)$$

3.3 Fractional-order Generalised Predictive Controller

As stated in Chapter 2, most approaches used for general fractional calculus are based on Grunwald-Letnikov (GL) and Riemann-Liouville (RL). In the world of engineering applications, usually, RL is used for algebraic manipulation and GL for numerical simulation and integration (Podlubny, 1999).

The idea of Fractional-order Generalised Predictive Controller was born from combining fractional calculus with GPC to create a generalisation of GPC. As shown in the previous sections, the GPC law is obtained from minimising the cost function found in equation (3.1), as GPC is using the CARIMA model found in equation (3.15). FGPC will follow the same path to formulate its control law.

There are two approaches to defining FGPC. In the following subsections, we will demonstrate both ways.

3.3.1 First approach of deriving FGPC

Romero and his team (Romero *et al.*, 2010b) have created FGPC by using the defined fractional integral operator. This new controller is based on the GPC predictive controller, which is used as a generalised fractional integral operator to redefine its cost function.

To get to FGPC, a series of mathematical equations shall be explained and illustrated. The following series of derivation is essential to understanding where the FGPC equation came from (Romero *at al*., 2010b).

We will give a discrete method to evaluate

$$\int_{g}^{l} D^{\alpha} f(t) dt \tag{3.32}$$

The notation g and l are chosen to be multiples of the sampling time Δt and $\alpha \in \Re$. The fractional derivative D^{α} can be evaluated using the GL definition (Ortigueira *et al.*, 2005), recalling equation (2.5)

$$D^{\alpha} f(t) = \lim_{h \to 0} h^{-\alpha} \sum_{i=0}^{\infty} (-1)^{i} \binom{\alpha}{i} f(t - ih)$$

where

- $h \in \Re^{+}$
- $\alpha \in \Re$

Substituting equation (2.5) to equation (3.32):

$$\int_{g}^{l} D^{\alpha} f(t) dt = \int_{g}^{l} \left[\Delta t^{-\alpha} \sum_{i=0}^{\infty} (-1)^{i} \binom{\alpha}{i} f(t - i\Delta t) \right] dt \tag{3.33}$$

As $h = \Delta t$, equation (3.33) can be expanded to

$$\int_g^l D^\alpha f(t)dt = \Delta t^{-\alpha}\left[\int_g^l (-1)^0 \binom{\alpha}{0} f(t)dt + \int_g^l (-1)^1 \binom{\alpha}{1} f(t-\Delta t)dt + \int_g^l (-1)^2 \binom{\alpha}{2} f(t-2\Delta t)dt + \cdots\right] \tag{3.34}$$

Using the standard notation $\omega_x \equiv (-1)^x \binom{\alpha}{x}$, equation (3.34) can be rewritten as:

$$\int_g^l D^\alpha f(t)dt = \Delta t^{-\alpha}\left[\int_g^l \omega_0 f(t)dt + \int_g^l \omega_1 f(t-\Delta t)dt + \int_g^l \omega_2 f(t-2\Delta t)dt + \cdots\right] \tag{3.35}$$

As noticed earlier, equation (3.35) is nothing but an infinite summation of definite integrals:

$$\omega_p \int_g^l f(t-p\Delta t)\,dt \tag{3.36}$$

which can be evaluated as

$$\omega_p \int_g^l f(t-p\Delta t)\,dt = \omega_p\left[F_p(l) - F_p(g)\right] \tag{3.37}$$

where F_p is a primitive function of $f(t-p\Delta t)$.

Applying the GL definition (i.e., equation (2.5)) to expand the expression

$$\omega_p \int_g^l f(t - p\Delta t)\, dt$$

$$= \omega_p \left[\left(\Delta t \sum_{i=0}^{\infty} (-1)^i \binom{-1}{i} f(l - (i+p)\Delta t) \right) \right. \tag{3.38}$$

$$\left. - \left(\Delta t \sum_{i=0}^{\infty} (-1)^i \binom{-1}{i} f(g - (i+p)\Delta t) \right) \right]$$

Now, equation (3.38) can be evaluated for each value of p, which will lead to the following general term:

$$\omega_p \int_g^l f(t - p\Delta t)\, dt$$

$$= \omega_p \Delta t [f(l - p\Delta t) - f(g - p\Delta t) + f(l - (2+p)\Delta t) \tag{3.39}$$

$$- f(g - (2+p)\Delta t) + f(l - (3+p)\Delta t) - f(g - (3+p)\Delta t)$$

$$+ \cdots]$$

Equation (3.35) can be summed as:

$$\int_g^l D^{\alpha} f(t) dt = \Delta t^{1-\alpha} . \bar{k} . \bar{f} \tag{3.40}$$

where $\Delta t^{1-\alpha}$ came from equation 3.33 as Δt has been taken as a common factor out of the summation and then multiplied by $\Delta t^{-\alpha}$ (which was there already as a result of GL definition) which led to the term $\Delta t^{1-\alpha}$.

$$\bar{k} = (\dots, \omega_{L+1} + \omega_L + \cdots + \omega_{G+2}\,, \omega_L + \omega_{L-1} + \cdots + \omega_{G+1}, \omega_{L-1} + \omega_{L-2} + \cdots + \omega_G, \dots, \omega_{L-n+1} + \omega_{L-n} + \cdots + \omega_{G-n+2}, \omega_{L-n} + \omega_{L-n-1} + \cdots + \omega_{G-n+1}, \dots, \omega_1 + \omega_0, \omega_0) \quad (3.41)$$

where; $G = {}^{g}/_{\Delta t}$, $L = {}^{l}/_{\Delta t}$, $n = L - G$

$$\bar{f} = \big(\dots, f(-\Delta t), f(0), f(\Delta t), \dots, f(g - \Delta t), f(g), \dots, f(l - \Delta t), f(l)\big) \quad (3.42)$$

Since $\bar{f}$ and $\bar{k}$ have an infinite number of terms, the integral in equation (3.40) has infinite memory. However, in practice, only a finite number of terms is needed due to the short memory principle (Ortigueira *et al.*, 2005).

After establishing the foundations of the mathematical derivation, FGPC cost function can be defined easily by comparing with MPC (GPC) cost function (equation 3.1). For the sake of simplicity, the expectation operator "E" and the notation "$|t$" have not been explicitly written:

$$J_{FGPC}(\Delta u, t) = \int_{N_1}^{N_2} D^{\alpha}[r(t) - y(t)]^2 dt + \int_{1}^{N_u} D^{\beta}[\Delta u(t-1)]^2 dt \quad (3.43)$$

As noticed, the notation j is not mentioned in equation (3.43) as the summation operator has been replaced with the integral operator.

Substituting equation (3.40) into equation (3.43):

$$J_{FGPC}(\Delta u, t) = (\Delta t^{1-\alpha}.\bar{e}.\bar{\gamma}.\bar{e}') + \big(\Delta t^{1-\beta}.\overline{\Delta u}.\bar{\lambda}.\overline{\Delta u}'\big) \quad (3.44)$$

where:

- $\bar{\gamma}$ represents the future error weighting sequences. The bar notation indicates it is a matrix and it is square with infinite dimension
- $\bar{\lambda}$ represents the control weighting sequences. The bar notation indicates it is a matrix and it is square with infinite dimension.
- α and β are fractional-order coefficients.

By truncating equation (3.44) to consider just the future values of e and Δu in the intervals of interest, $[N_1,\ N_2]$ and $[1, N_u]$, respectively. The truncated version, final value, of equation (3.44) can be obtained as follows:

$$J_{FGPC} = e\gamma(\alpha, \Delta t)e' + \Delta u\lambda(\beta, \Delta t)\Delta u' \tag{3.45}$$

where

- $\gamma(\alpha, \Delta t)$ and $\lambda(\beta, \Delta t)$ are defined in equations (3.46) and (3.47) respectively.

$$\gamma(\alpha, \Delta t) = \begin{pmatrix} \omega_h + \omega_{h-1} + \cdots \omega_{N_1-1} & 0 & 0 & \cdots & 0 & 0 \\ 0 & \omega_{h-1} + \omega_{h-2} + \cdots \omega_{N_1-2} & 0 & \cdots & 0 & 0 \\ 0 & 0 & \omega_{h-2} + \omega_{h-3} + \cdots \omega_{N_1-3} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \omega_1 + \omega_0 & 0 \\ 0 & 0 & \cdots & \cdots & 0 & \omega_0 \end{pmatrix} \tag{3.46}$$

and $h = N_2 - N_1$

$$\lambda(\beta, \Delta t) = \begin{pmatrix} \omega_{N_u-1} + \omega_{N_u-2} + \cdots \omega_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \omega_{N_u-2} + \omega_{N_u-3} + \cdots \omega_0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \omega_{N_u-3} + \omega_{N_u-4} + \cdots \omega_0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \omega_1 + \omega_0 & 0 \\ 0 & 0 & \cdots & \cdots & 0 & \omega_0 \end{pmatrix} \tag{3.47}$$

By defining those equations, we have identified a classical GPC formulation with weighting sequences γ and λ given by the sampling time Δt and the fractional-order of derivation α and β. The optimal control law can be derived by minimising equation (3.44) using conventional GPC techniques (Camacho & Bordones, 2004; Clarke, 1987a; 1987b; 1988; Maciejowski, 2002; Rossiter, 2003).

3.3.2 Second approach of deriving FGPC

Another possible solution for driving FGPC is using the definition of a fractional definite integral operator, which will lead to a slightly different formula for FGPC concluded in the previous equation (Romero *et al.*, 2010a; 2012). The definition of the definite fractional integral operator is as follows:

$$\int_g^l D^{1-\alpha} f(t) dt \tag{3.48}$$

Using the GL definition, equation (2.5), and applying to $D^{1-\alpha}$ to the equation

$$\int_g^l D^{1-\alpha} f(t) dt = \int_g^l \left[\Delta t^{\alpha-1} \sum_{i=0}^{\infty} (-1)^i \binom{1-\alpha}{i} f(t - i\Delta t) \right] dt \tag{3.49}$$

which can be expanded to

$$\int_g^l D^{1-\alpha} f(t)dt$$

$$= \Delta t^{\alpha-1}\left[\int_g^l (-1)^0 \binom{1-\alpha}{0} f(t)dt + \int_g^l (-1)^1 \binom{1-\alpha}{1} f(t-\Delta t)dt + \int_g^l (-1)^2 \binom{1-\alpha}{2} f(t-2\Delta t)dt + \cdots\right] \tag{3.50}$$

Using the standard notation $\omega_x \equiv (-1)^x \binom{1-\alpha}{x}$, equation (3.50) can be rewritten as

$$\int_g^l D^{1-\alpha} f(t)dt$$

$$= \Delta t^{\alpha-1}\left[\int_g^l \omega_0 f(t)dt + \int_g^l \omega_1 f(t-\Delta t)dt + \int_g^l \omega_2 f(t-2\Delta t)dt + \cdots\right] \tag{3.51}$$

Following the same steps from the first approach which yields:

$$\int_g^l D^{1-\alpha} f(t)dt = \Delta t^{\alpha}.\bar{k}.\bar{f} \tag{3.52}$$

where:

Δt^{α} appeared from equation (3.48) as Δt has been taken as a common factor out of the summation and then multiplied by $\Delta t^{\alpha-1}$ (which was there already as a result of the GL definition) which led to the term Δt^{α}.

Now, the truncated *FGPC* equation can be rewritten as

$$J_{FGPC}(\Delta u, t) = e\Gamma(\alpha, \Delta t)e^{'} + \Delta u\Lambda(\beta, \Delta t)\Delta u^{'} \tag{3.53}$$

where:

$$\Gamma(\alpha, \Delta t) = \Delta t^{\alpha} diag(\, \omega_m \; \omega_{m-1} \ldots \; \omega_1 \; \omega_0) \tag{3.54}$$

and,

- $m = N_2 - N_1$
- $\omega_l = (-1)^l \binom{-\alpha}{l}$, and $\omega_l = 0$ for $\forall l < 0$

$$\Lambda(\beta, \Delta t) = \Delta t^{\beta} diag(\omega_{N_u-1} \; \omega_{N_u-2} \ldots \omega_1 \; \omega_0) \tag{3.55}$$

and,

- $\omega_l = (-1)^l \binom{-\beta}{l}$, and $\omega_l = 0$ for all $l < 0$

3.3.3 Obtaining the control law

By minimising the cost function J_{FGPC} , the values of the control law $u(t + j|\mathrm{t})$ can be obtained. The first step is to replace the cost function with the equation of prediction (equation (3.2) with the omission on the notation "$|t$" for simplification):

$$y(t + k) = y_c(t + k) + y_f(t + k)$$

Due to the nature of the terms that form part of the function, the following notation is used: "→" for predicted future values and "←" for the past values. Thus, for FGPC we are going to present all its vectorial variables in terms of past values and predicted future values as follows:

- Control "Δu" and error "e" vectors:

$$\Delta u = \begin{bmatrix} \Delta u_{(\leftarrow)} \\ - \ - \\ \Delta u_{(\rightarrow)} \end{bmatrix} = \begin{bmatrix} \vdots \\ \Delta u(k-3) \\ \Delta u(k-2) \\ \Delta u(k-1) \\ - \ - \ - \ - \\ \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_u-1) \end{bmatrix} \tag{3.56}$$

$$e = \begin{bmatrix} e_{(\leftarrow)} \\ - \ - \\ e_{(\rightarrow)} \end{bmatrix} = \begin{bmatrix} \vdots \\ e(k-2) \\ e(k-1) \\ e(k) \\ - \ - \ - \ - \\ e(k+1) \\ e(k+2) \\ \vdots \\ e(k+N_2) \end{bmatrix} \tag{3.57}$$

- System reference "r" and output "y"

$$r = \begin{bmatrix} r_{(\leftarrow)} \\ - \ - \\ r_{(\rightarrow)} \end{bmatrix} = \begin{bmatrix} \vdots \\ r(k-2) \\ r(k-1) \\ r(k) \\ - \ - \ - \ - \\ r(k+1) \\ r(k+2) \\ \vdots \\ \Delta u(k+N_2) \end{bmatrix} \tag{3.58}$$

$$y = \begin{bmatrix} y_{(\leftarrow)} \\ - \ - \\ y_{(\rightarrow)} \end{bmatrix} = \begin{bmatrix} \vdots \\ y(k-2) \\ y(k-1) \\ y(k) \\ - \ - \ - \ - \\ y(k+1) \\ y(k+2) \\ \vdots \\ y(k+N_2) \end{bmatrix} \tag{3.59}$$

- Weight sequence "Γ and Λ"

$$\Gamma = \begin{bmatrix} \Gamma_{(\leftarrow)} & 0 \\ 0 & \Gamma_{(\rightarrow)} \end{bmatrix} = \begin{bmatrix} \ddots & & & & \\ & \gamma_{-1} & & 0 & \\ & & \gamma_0 & & \\ & & & \gamma_1 & \\ & 0 & & & \ddots \\ & & & & \gamma_{N_2} \end{bmatrix} \tag{3.60}$$

$$\Lambda = \begin{bmatrix} \Lambda_{(\leftarrow)} & 0 \\ 0 & \Lambda_{(\rightarrow)} \end{bmatrix} = \begin{bmatrix} \ddots & & & & \\ & \lambda_{-1} & & 0 & \\ & & \lambda_0 & & \\ & & & \lambda_1 & \\ & 0 & & & \ddots \\ & & & & \lambda_{N_u-1} \end{bmatrix} \tag{3.61}$$

Using this new notation, equation (3.52) can be re-written as a discrete FGPC cost function as follows:

$$\begin{aligned} J_{FGPC}(t, \Delta u_{(\rightarrow)}) &= [e'_{(\leftarrow)} \quad | \quad e'_{(\rightarrow)}] \begin{bmatrix} \Gamma_{(\leftarrow)} & 0 \\ 0 & \Gamma_{(\rightarrow)} \end{bmatrix} \begin{bmatrix} e_{(\leftarrow)} \\ - \\ e_{(\rightarrow)} \end{bmatrix} \\ &+ [\Delta u'_{(\leftarrow)} \quad | \quad \Delta u'_{(\rightarrow)}] \begin{bmatrix} \Lambda_{(\leftarrow)} & 0 \\ 0 & \Lambda_{(\rightarrow)} \end{bmatrix} \begin{bmatrix} \Delta u_{(\leftarrow)} \\ - - \\ \Delta u_{(\rightarrow)} \end{bmatrix} \\ &= \left(e'_{(\rightarrow)} \Gamma_{(\rightarrow)} e_{(\rightarrow)} + \Delta u'_{(\rightarrow)} \Lambda_{(\rightarrow)} \Delta u_{(\rightarrow)}\right) \\ &+ \left(e'_{(\leftarrow)} \Gamma_{(\leftarrow)} e_{(\leftarrow)} + \Delta u'_{(\leftarrow)} \Lambda_{(\leftarrow)} \Delta u_{(\leftarrow)}\right) \equiv J_{FGPC_{(\rightarrow)}} + J_{FGPC_{(\leftarrow)}} \end{aligned} \tag{3.62}$$

Following the same steps taken in obtaining the control low of MPC (refer to section 3.1.4) and substituting into equation (5.53):

$$\begin{aligned} J_{FGPC} = \Big[&\Delta u'_{(\rightarrow)} \left(G' \Gamma_{(\rightarrow)} G + \Lambda_{(\rightarrow)}\right) \Delta u_{(\rightarrow)} - 2 E_{0_{(\rightarrow)}} \Gamma_{(\rightarrow)} G \Delta u_{(\rightarrow)} \\ &+ E_{0_{(\rightarrow)}} \Gamma_{(\rightarrow)} E_{0_{(\rightarrow)}} \Big] + \left[e'_{(\leftarrow)} \Gamma_{(\leftarrow)} e_{(\leftarrow)} + \Delta u'_{(\leftarrow)} \Lambda_{(\leftarrow)} \Delta u_{(\leftarrow)}\right] \end{aligned} \tag{3.63}$$

With no active constraints, the optimal control law will be as follows:

$$\Delta u^*_{FGPC}(t) = argmin_{\Delta u} J_{FGPC} = (G'^{T_{(f)}} G + \Lambda_{(\rightarrow)})^{-1} G' T_{(\rightarrow)} E_{0_{(\rightarrow)}} \equiv K E_{0_{(\rightarrow)}} \qquad (3.64)$$

The same expression has been obtained for GPC (Clarke, 1987a,b); however, in FGPC, the weighting sequences (λ and γ) are defined by high-level tuning fractional-order parameters (α and β) using the previously defined equations.

Without any active constraints, the minimisation of J_{FGPC} will lead to an LTI control law, which can be calculated in advance.

3.4 Key differences between FGPC and GPC

The first difference that can be noticed on the number of predicted errors that each one of those controllers is taking into account. Considering the error function on GPC,

$$E_o \equiv \big(e_o(t+N_1), \dots, e_o(t+N_2)\big)' \qquad (3.65)$$

whereas in FGPC the error function is

$$E_o \equiv \big(e_o(t+1), \dots, e_o(t+N_2)\big)' \qquad (3.66)$$

Assuming that $N_1 > 1$, GPC is taking into account N_1 and N_2 regardless of the value of N_1, hence equation (3.65), whereas FGPC will always assume that $N_1 = 1$, hence equation (3.66). Thus, it can be easily observed that FGPC is taking into account more elements than GPC.

The second difference is regarding the way that FGPC and GPC define the weighting sequences. In GPC, the weighting sequences are defined directly from the cost function, recalling equation (3.1):

$$J_{GPC}(\Delta u, t) = \sum_{k=N_1}^{N_2} \gamma[r(t+k) - y(t+k)]^2 + \sum_{k=1}^{N_u} \lambda_i \Delta u(t+k)^2$$

On the other hand, FGPC defines the weighing sequences through α and β as shown in the previous section. The key thing here is that the weighting sequences in GPC are constant and nonnegative in most cases whereas, in FGPC, the weighting sequences are defined using α and β. Thus, they can be negative and not necessarily constant. Although having negative weights in predictive control is an unusual case, it cannot be ignored, as there are some situations where the controller gives its maximum potential in such a case (Romero *et al.*, 2010a).

The third and most important difference is that FGPC and GPC are not defining the same set of controllers. In other words, for each system, there will be three different types of controllers as stated by Romero *et al.* (2010b). Figure 3.6 demonstrates these.

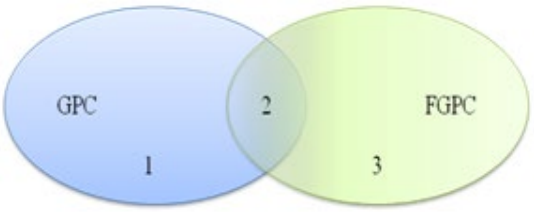


Figure 3.6: The three sets of controllers

Referring to the figure above, the system either has a type 1 controller (GPC), a type 2 controller (GPC or FGPC), or a type 3 controller (FGPC). As stated earlier in this chapter, FGPC is a more general case of GPC and this can be observed in the previous figure. The main

difference between GPC and FGPC is that FGPC is considering Λ whereas GPC doesn't. This parameter will give FGPC the privilege to calculate the controller's weightings with negative sequences which are ignored by GPC. Back to the figure, a type 1 controller is a pure GPC controller that has no negative weightings and has no (or a very small value that can be ignored) Λ and a unity Γ. A type 2 controller can be considered as GPC or FGPC, for which Λ and Γ can be calculated and still have positive values. Finally, a type 3 controller is purely FGPC with negative values of Λ and Γ. The key point that can convert FGPC to pure GPC is to find a value of α and β that can define Γ as unity and Λ is zero (or a very small fraction that can be ignored).

3.5 MATLAB implementation

To implement the FGPC formula in MATLAB, we need to understand the basis of the mathematical equations used and measure their compatibility to be applied in MATLAB. For example, to apply the standard notation $\omega_x \equiv (-1)^x \binom{\varphi}{x}$, we need to understand what $\binom{\varphi}{x}$ means and how it can be computed.

The term $\binom{\varphi}{x}$ is the binomial coefficient and it reads as " φ choose x". There are several ways to solve this mathematical term. It can be computed using one of the following methods (Gross, 2009):

- Recursive formula:

$$\binom{\varphi}{x} = \binom{\varphi-1}{x-1} + \binom{\varphi-1}{x} \tag{3.67}$$

for all integers $\varphi, x: 1 \leq x \leq \varphi - 1$

- Factorial formula:

$$\binom{\varphi}{x} = \frac{\varphi!}{x!\,(\varphi - x)!} \tag{3.68}$$

for $0 \leq x \leq \varphi$

- Newton's generalised formula:

$$\binom{\varphi}{x} = \frac{\varphi^{\underline{x}}}{x!} = \frac{\varphi(\varphi-1)(\varphi-2)\ldots(\varphi-(x-1))}{x(x-1)(x-2)\ldots 1} = \prod_{i=1}^{x} \frac{\varphi-(x-i)}{i}$$

$$= \prod_{i=1}^{x} \frac{\varphi+1-i}{i} \tag{3.69}$$

Here the symbol $\varphi^{\underline{x}}$ is expressed as a falling factorial power. This formula can be used to compute an arbitrary number φ (a real, complex, or negative number).

We have used the latter formula (i.e. equation (3.69)) as it is the most general and compatible with the characteristics of φ which can be any arbitrary number. The code has been divided into functions to compute each mathematical equation separately for error tracking and debugging. The first function created was a function “gamma” that uses Newton’s binomial formula to compute the values of the first weighting element of FGPC (i.e. $\gamma(\beta, \Delta t)$) using equations (3.46). The second function created was the function "lambda" which is created to compute the second weighting element in FGPC ($\lambda(\alpha, \Delta t)$) using equation (3.47). The third and last function was the function "FOGPC" which carries out all the numerical calculations to compute the expectation equation of FGPC (i.e. equation (3.45)). The FGPC output and control law are simulated using the "Captain" toolbox created by Taylor *et al*. (1999).

3.5.1 Worked example

To shed more light on the ideas in this chapter, a detailed worked simple example is considered in this section for tutorial purposes. We will go through the example and apply both GPC and FGPC controllers on the model and compare the responses of each controller. The comparison will illustrate the difference between the two responses. Equation (3.70) represents a model that is discretised by $\Delta t = 1$.

$$G(z^{-1}) = \frac{1 - 3z^{-1}}{1 - 0.75z^{-1}} \tag{3.70}$$

According to the recommendations stated in section 3.2.2 and (Rossiter, 2003), N_u and N_2 need to be chosen sufficiently large for this kind of plant. Therefore, we will assume the following:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- No model mismatch
- For GPC, constant values for (γ, λ), as $\gamma = 1$, $\lambda = 10^{-6}$
- The simulation ended at the final time $T_f = 25$ with a sampling time $T_s = 1$
- For FGPC, 2 different values for (α, β) which result from 2 different sets of values for γ and λ, according to equation (3.46) and equation (3.47), respectively. The choice of (α, β) is based on an initial guess and observation of the FGPC response. The first initial guess will be based on achieving the same response as GPC. The second guess will aim to get a faster response to achieve the desired output with minimum overshoot. After several simulations, the

following values of (α, β) have been chosen to achieve a similar response to GPC:

1. (α, β) = (1.5,2), referring to equation (3.46) & (3.47), respectively, and using the MATLAB code created, we obtained the following values:

$$\gamma = \begin{pmatrix} 2.5239 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3.3385 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3.1421 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.9326 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.7070 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2.4609 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.1875 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.8750 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\lambda = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

After running those parameters in the code created in MATLAB (2017a) on a laptop (using Windows 10, with an Intel(R) i5-4200u CPU processor and 8GB RAM), the following control polynomials were derived (i.e. forward path polynomial $R(z^{-1})$ and feedback $S(z^{-1})$) for both GPC and FGPC. The control polynomials of both controllers are as follows:

$$R_{GPC}(z^{-1}) = -2.78 - 6.78z^{-1} \tag{3.71}$$

$$S_{GPC}(z^{-1}) = 2.7 - 1.7z^{-1} \tag{3.72}$$

$$R_{FGPC}(z^{-1}) = -2.73 - 6.7z^{-1} \tag{3.73}$$

$$S_{FGPC}(z^{-1}) = 2.7 - 1.7z^{-1} \tag{3.74}$$

Comparing equation (3.71) to equation (3.73) and equation (3.72) to equation (3.74), GPC and FGPC have almost identical control polynomials which will yield very similar closed-loop responses.

In addition, the responses of the plant to the different controllers (GPC and FGPC) were demonstrated. In the following Figure 3.7, matching responses of both GPC and FGPC can be observed:

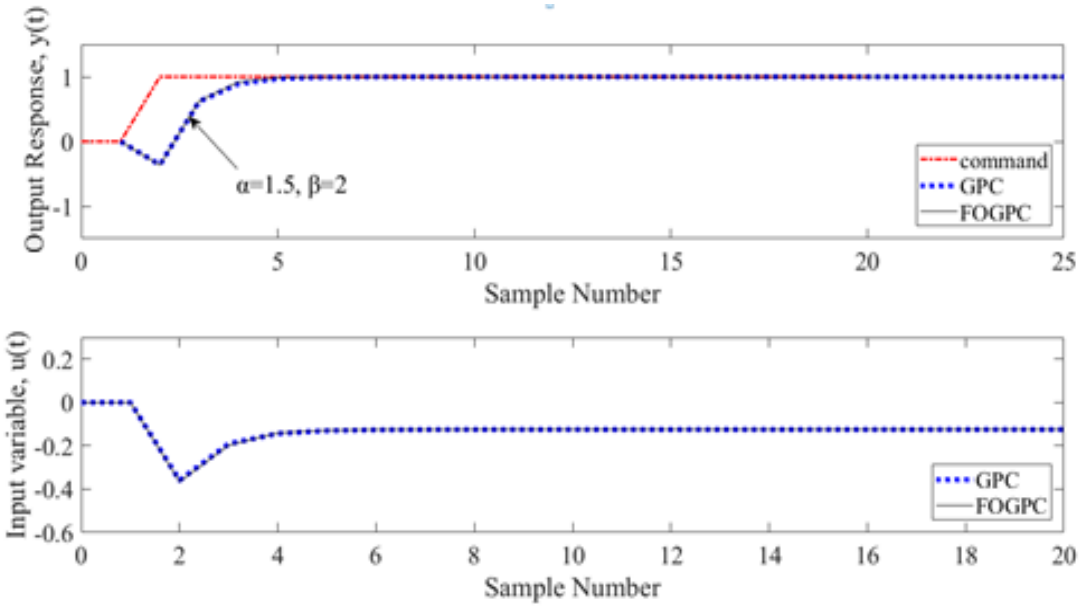


Figure 3.7: Response of both GPC and FGPC with the parameters chosen above.

We notice that the response is quite similar which indicates that this is a type 2 controller where FGPC is behaving like GPC as stated in section 3.4

The second choice of (α, β) has yielded the following corresponding values of γ and λ:

2. $(\alpha, \beta) = (0.8, 0.9)$

$$\gamma = \begin{pmatrix} -0.2525 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.7559 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.7655 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.7766 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7897 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8058 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.8265 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.8550 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\lambda = \begin{pmatrix} -0.2 & 0 \\ 0 & 1 \end{pmatrix}$$

As GPC parameters have not been changed, the control polynomials for GPC remains the same (i.e., the same equations (3.71) and (3.72)), whereas FGPC controller polynomials are changed due to the change in both α and β.

$$R_{FGPC}(z^{-1}) = -1.81 - 5.84z^{-1} \tag{3.75}$$

$$S_{FGPC}(z^{-1}) = 2.46 - 1.46z^{-1} \tag{3.76}$$

Comparing the new calculated control polynomials to GPC ones, an obvious difference can be noticed, despite the way in which they are both using the same design horizons (i.e N_1, N_2 and N_u). Figure 3.8 illustrates both GPC and FGPC responses to the new values of α and β.

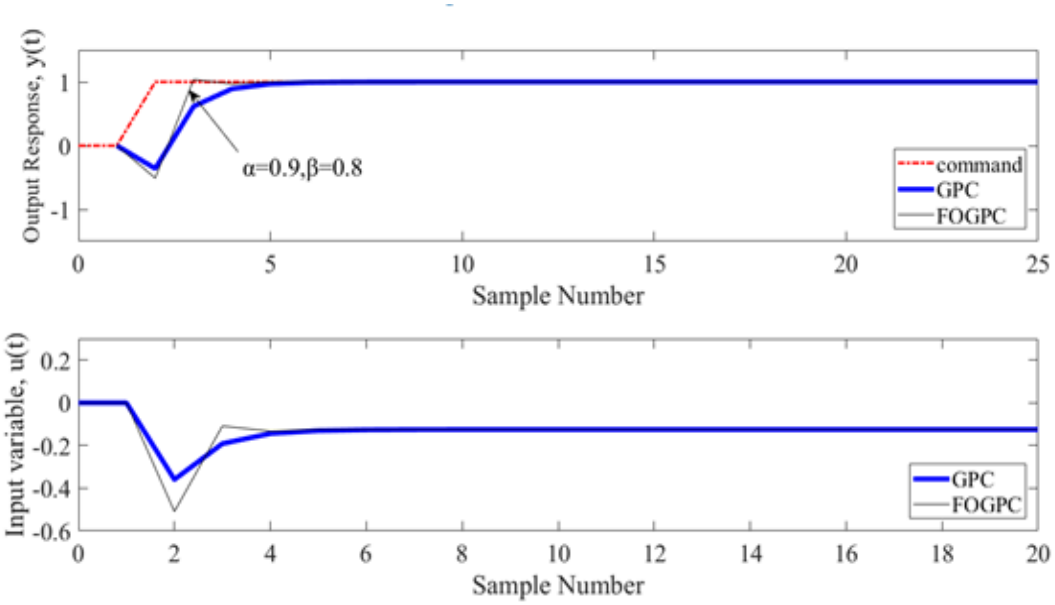


Figure 3.8: Response of both GPC and FGPC with the parameters chosen.

We notice that the response is different which indicates that this is a type 3 controller where FGPC can response in a different way (faster in this case) to GPC as stated in section 3.4

In the second plot, an interesting observation can be seen, i.e. with (α, β) changed, the response of FGPC changed. The next chapter will include further examples and detailed analysis for the responses of GPC and FGPC in a more rigorous comparison.

3.6 Concluding Remarks

The present chapter has reviewed MPC, GPC, and FGPC in terms of their respective mathematical numerical derivations and control algorithm equations. GPC is a special case of MPC, whilst FGPC is presented here as a generalised case of GPC. A worked example at the end of the chapter has briefly demonstrated the potential difference between GPC and FGPC in terms of their responses to the same model using the same horizons in their design (but with several different settings for the extra FGPC design terms). The following Chapter builds on this, to consider different models and simulation scenarios, such as the disturbance response and model mismatch.

Chapter 4 FGPC Simulation Study

Chapter 4 considers the performance of FGPC for various case study examples. The responses of FGPC are compared to conventional GPC in terms of the rise time, robustness and settling time for several different models. A trial and error approach is utilised to set the GPC tuning coefficients for each example, whilst the conventional GPC settings, such as the various forecasting horizons, are the same for both GPC and FGPC. However, in the case of FGPC, the additional tuning coefficients α and β are investigated by simulation, i.e. how can these be used to modify the closed-loop characteristics? Note that the FGPC tuning method (i.e. α and β choice) was proposed by Romero *et al.* (2011), based on a previously developed method used for tuning fractional-order $PI^{\lambda}D^{\mu}$ systems (Monje *et al.*, 2004; Chen *et al.*, 2006; Monje *et al.*, 2008). This latter methodology is based on an optimisation tool presented in the MATLAB platform. Here, the function *fmincon* (Romero *et al*., 2013) is used to solve the corresponding optimisation problem (a similar function has been used in Chapter 5 section 5.5.1).

For this chapter, simulations are performed for each case study using MATLAB 2017a installed on a laptop (Windows 10, with an Intel(R) i5-4200u CPU processor and 8GB RAM). The CAPTAIN Toolbox for MATLAB (Taylor *et al.*, 1999) is used for conventional GPC design, whilst the new functions developed by the present author (see section 3.5) are used to solve the FGPC problem. The chapter aims to demonstrate the features and advantages of FGPC over GPC design. Sections 4.1 to 4.3 consider each case study example, in turn, followed by the discussion and conclusions in sections 4.4 and 4.5 respectively.

4.1 Case study 1 (First Order model)

The key advantage of FGPC over GPC is that FGPC provides more freedom for tuning by introducing extra parameters that can set the controller in the fractional-order (i.e. α and β). The effect of α and β on FGPC is huge as the selection of these two coefficients will affect directly the values of γ and λ respectively which, in turn, will influence the response of the controller. To examine this further, we will conduct systematic simulations of FGPC with different values of α and β to observe the effect of changing their values on the controller (FGPC) behaviour. The first case study concerns a first-order model. We will use trial and error to find the best response of FGPC in terms of the fastest response and with minimum overshoot by adjusting the α and β coefficients only (the other settings are chosen *a priori*: see below). The following model will be used in this case study:

$$G(z^{-1}) = \frac{1 - 2z^{-1}}{1 - 0.9z^{-1}} \tag{4.1}$$

The tuning parameters for controlling this model are the following:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- For GPC, constant values for (γ, λ), as $\gamma = 1$, $\lambda = 10^{-6}$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$

After several simulations, we have chosen $\alpha = 0.77$ and $\beta = 1$ to be our reference values for those coefficients. This yields the following values of γ and λ, respectively:

$$\gamma = \begin{pmatrix} -0.5020 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5111 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5262 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5441 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5657 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5930 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.6292 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.6815 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.7700 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\lambda = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

Figure 4.1 below shows the responses of GPC and FGPC, showing both the system outputs and the control input variables.

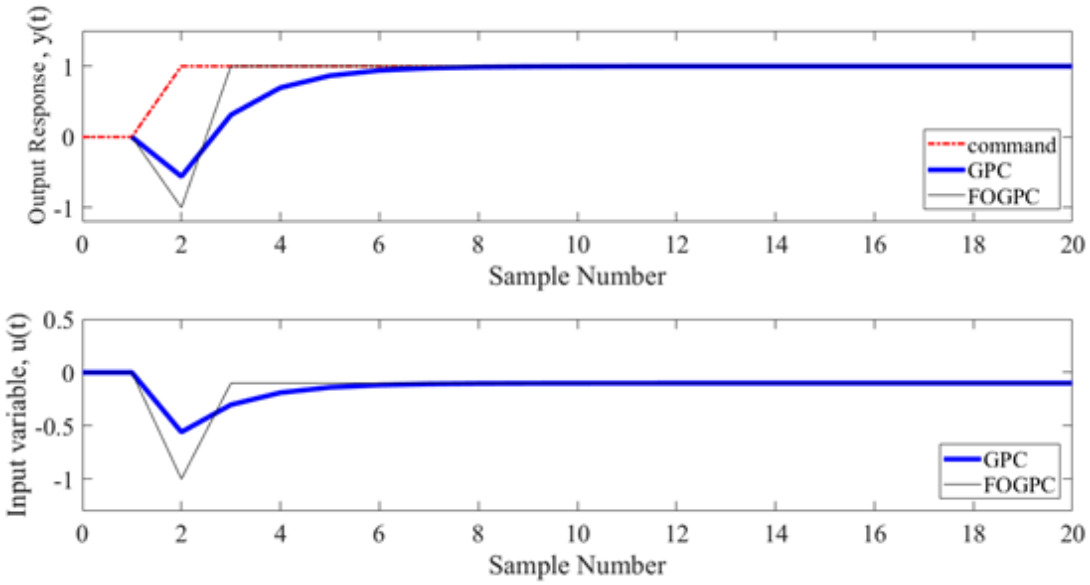


Figure 4.1: GPC and FGPC responses along with their input variables

From Figure 4.1, we can notice that FGPC (when adjusted using α and β) reaches the set point faster than GPC, with no overshoot. However, this comes at the expense of a large deviation from the setpoint at the second sample.

After setting our reference values (i.e. $\alpha = 0.77$ and $\beta = 1$), we will start analysing the effect of α and β over FGPC response to the same model by simulating FGPC with different values of α and β.

4.1.1 Changing α with fixed β

Like any tuning parameters, α and β has a limitation of tuning, as after a certain range of value they will cause an unstable response. Indeed, the range of those values can be defined by analytical methods by using mathematical equations mentioned in the previous chapter (i.e., equations 3.46 and 3.47); however, in this section and the throughout this chapter, the goal is to determine the effects of α and β by fixing one parameter and running various values to the other to observe the effect on the response. Thus, different values of α will be tried while β remains constant at 1. The critical point of α before the controller is unstable is 0.38, hence we have assigned the lowest value $\alpha = 0.5$ in Figure 4.2, with increments of 0.1 for 10 iterations, all with $\beta = 1$. Figure 4.2 illustrates the different responses of FGPC compared to the GPC response.

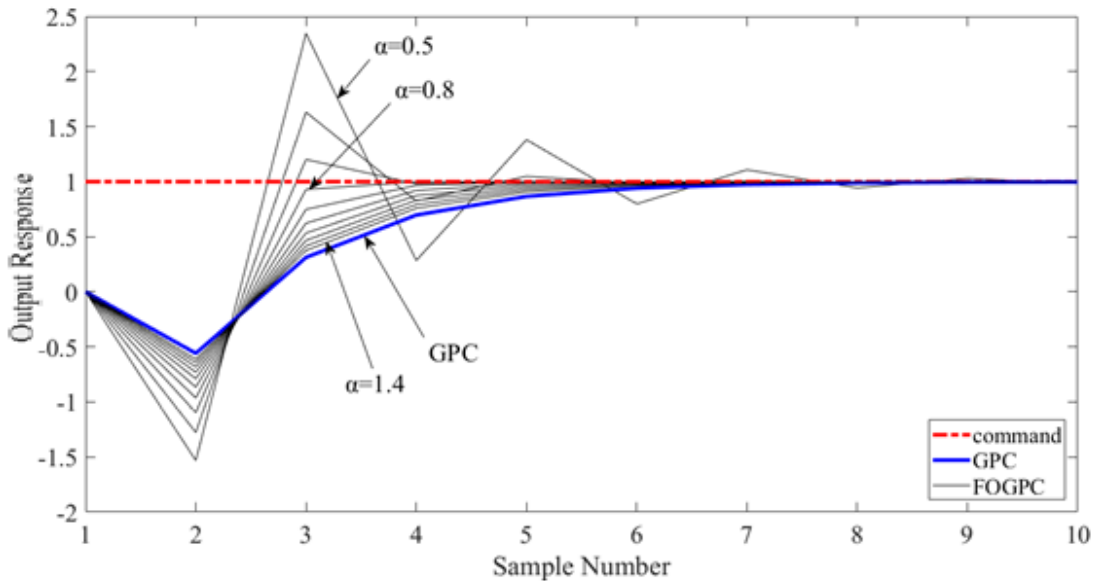


Figure 4.2: Responses of FGPC with α varied from 0.5 to 1.4 by an increment of 0.1 combined and $\beta = 1$ with GPC response

By observing Figure 4.5, we notice that by increasing α the response of FGPC tends to be slower with a smaller overshoot and longer settling time. There is a direct relationship between the value of α and the response of FGPC; thus, the value of α needs to be chosen carefully to fulfil the required specifications of the system. Table 4.1 below represents a tabular comparison.

Table 4.1 A comparison between different α values in terms of response time and overshoot percentage for case study 1.

	α=0.5	α=0.6	α=0.7	α=0.8	α=0.9	α=1	α=1.1	α=1.2	α=1.3	α=1.4
Response time [seconds]	2.73	2.81	2.94	3.86	4.48	4.96	6.32	6.52	6.78	7.09
Overshoot [% of step up]	%146	%56	%17	%0	%0	%0	%0	%0	%0	%0

4.1.2 Changing β with fixed α

In this section, we are interested in discussing the effect of β on the FGPC response to the model by increasing β from 0.5 by a fixed value of 0.3 for 10 iterations with constant $\alpha = 0.77$. The value 0.5 is set to β as the lower values will cause the response to be unstable. Figure 4.3 shows the different responses of FGPC with different β's.

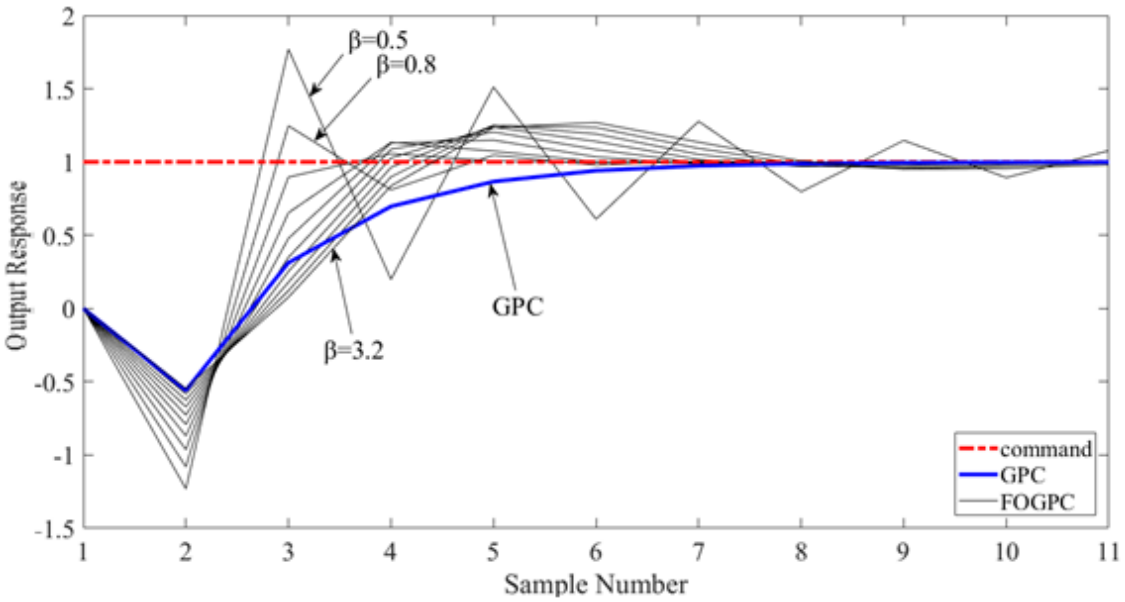


Figure 4.3: Responses of FGPC with β varied from 0.5 to 3.2 by an increment of 0.3 and $\alpha = 0.77$ with GPC response

Figure 4.3 demonstrates the effect of β on the FGPC response and, as we can see, larger β yields smaller overshoot of the FGPC response. However, the response of FGPC is tending to be slower with bigger β. Thus, both α and β play an important role in the design of FGPC. Table 4.2 shows the comparison.

Table 4.2 A comparison between different β values in terms of response time and overshoot percentage for case study 1.

	β=0.5	β=0.8	β=1.1	β=1.4	β=1.7	β=2	β=2.3	β=2.6	β=2.9	β=3.2
Response time [seconds]	2.83	2.91	2.94	3.86	3.87	3.91	4.01	4.13	4.27	4.32
Overshoot [% of step up]	%78	%23	%0.5	%7	%7	%19	%19	%20	%19	%19

4.1.3 Monte Carlo Analysis

One of the most vital considerations in practical implements of any control system is the robustness of the control system to uncertainty arising from both model parameter estimation and stochastic disturbance inputs to the system. There are many ways to handle this problem; however, with modern technology and the ability to access powerful desktop computers, MC analysis is one of the easiest, non-complicated and most appealing approaches to the problem. MC analysis is a computerised simulation that is built using repeated random samples and statistical analysis, probability distribution, to process the outcome. In other words, MC methodology is very closely related to random experiments, where the results are not known in advance (Raychaudhuri, 2008).

Here, the closed-loop system is repeatedly simulated with the model parameters for each such realisation selected randomly from the estimated joint probability distribution.

Consider the following:

$$G(z^{-1}) = \frac{B(z^{-1})z^{-1}}{A(z^{-1})} \tag{4.2}$$

where:

$$B(z^{-1}) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_n z^{-n}$$
$$A(z^{-1}) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n} \quad (4.3)$$

Then, according to equation (4.1):

$$b_0 = 1, \qquad b_1 = -2,$$

$$a_0 = 1, \qquad a_1 = -0.9$$

As the MC analysis uses a set of random samples, we will limit the samples to be between the following sets:

$$b_1 = \{1.5, 2.5\} \quad , \quad a_1 = \{0.4, 1.4\} \qquad 4.1$$

By limiting the range of random numbers between those intervals we have created a normal distribution around b_1 and a_1 as shown in Figure 4.4 and Figure 4.5

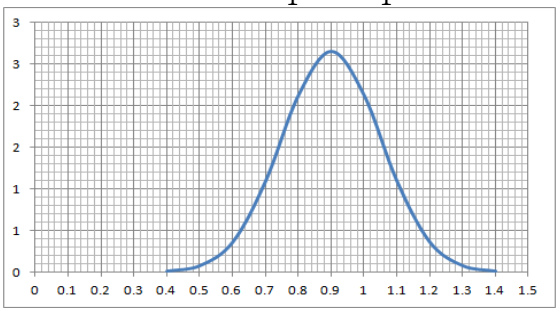


Figure 4.4: a normal distribution of a_1 around 0.9, its actual value

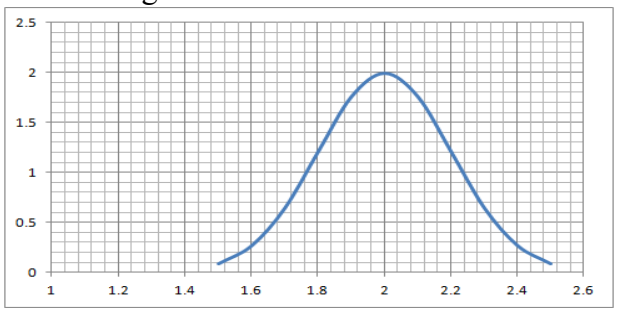


Figure 4.5: a normal distribution of b_1 around 2, its actual value

Now after defining the range of the random samples, MC analysis can be performed for the GPC and FGPC controllers that are applied in the example in the previous section as follows:

- Input forecasting horizon $N_u = 2$

- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- GPC weighting parameter $\lambda = 10^{-6}$

For this example, we have chosen $\alpha = 0.77$ and $\beta = 1$ for FGPC.

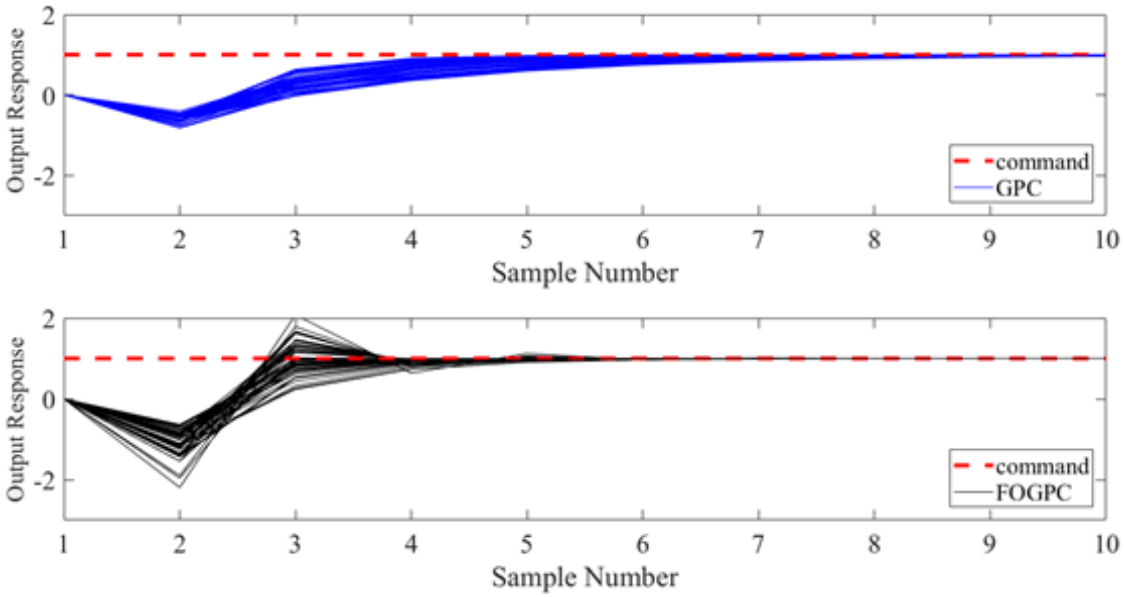


Figure 4.6: The top plot is GPC ($N_u = 2, N_1 = 1, N_2 = 10, \lambda = 10^{-6}, \gamma = 1$). The bottom plot is FGPC ($N_u = 2, N_1 = 1, N_2 = 10, \alpha = 0.77, \beta = 1$)

Within the context of the assumptions made above, FGPC provides a generalisation of the GPC cost function weights, which ultimately determine the numerical values of the control gains. Hence, the value of the approach appears dependent on whether the extra design flexibility provided by FGPC can be utilised to meet control objectives that are not achievable using standard GPC; and whether FGPC provides a straightforward to tune control algorithm – for

example, that the use of _ and _ in this way provides a meaningful or convenient approach to solving practical control problems. More discussion and illustration will be presented in the discussion section of this chapter.

4.2 Case study 2 (Higher-order model)

In this section, a higher-order model is simulated to test the effect of the α and β in the FGPC. The goal is not to optimise the response, the goal is to illustrate how these parameters can be utilised by studying their behaviour. Thus, a trial and error methodology is adopted to assign the values of α and β. The initial guess of the values will be $\alpha = \beta = 0.5$ and then the values will be increased or decreased systematically (i.e., increased or decreased by 0.1) and the results will be observed and compared. Consider the following high order plant:

$$G(z^{-1}) = \frac{1-3z^{-1} + 5z^{-2}+0.3z^{-3}}{1 - 0.6z^{-1} - z^{-2} + 1.5z^{-3}} \tag{4.5}$$

The model in equation 4.5 will be controlled by the following set of parameters which have been chosen as a reference for both GPC and FGPC based on Rossiter's (2003) recommendations:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- No model mismatch
- For GPC, a constant values for (γ, λ), as $\gamma = 1, \lambda = 10^{-6}$

- For FGPC, (α, β) as $\alpha = \beta = 0.5$, which result in the following values for γ and λ, according to equation 3.46 and equation 3.47 respectively

$$\gamma = \begin{pmatrix} -0.8145 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1964 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.2095 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2256 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.2461 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.2734 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.3125 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.3750 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\lambda = \begin{pmatrix} -0.5 & 0 \\ 0 & 1 \end{pmatrix}$$

- The simulation ended at the final time $T_f = 40$ with a sampling time $T_s = 1$

The simulation result can be seen in Figure 4.7 below:

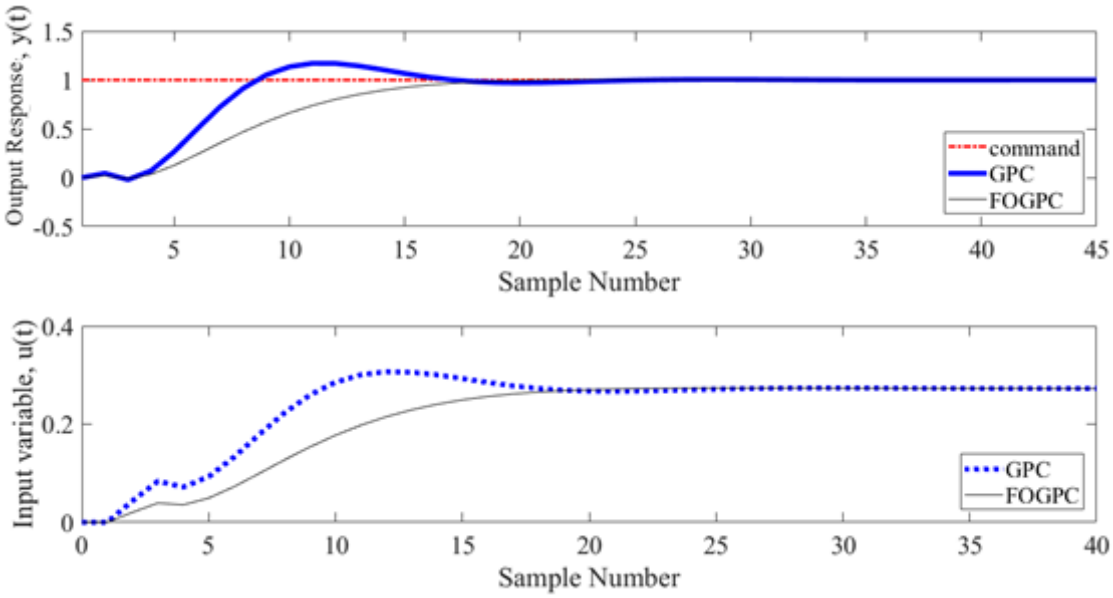


Figure 4.7: GPC and FGPC responses combined with a subplot of the input variable for both controllers

In Figure 4.7 we notice that the GPC response is faster. However, GPC has overshoot reached to 1.16 and has not reached the set point asymptotically till sample number 26. On the other hand, FGPC has a slower rise time at sample number 17 and its response has no overshoot compared to GPC. In addition, FGPC response reached the set point asymptotically at sample number 18. These values of α and β can be used for certain types of application that consider that overshoot is not acceptable and may affect its performance.

In the next sub-section, the effect of α and β will be tested individually by changing the value of one parameter while the other one remains constant.

4.2.1 Changing α with fixed β

In this section, we will be doing some adjustments on the coefficients α and β by increasing α by a fixed value of 0.1 for 15 iterations with fixed $\beta = 0.5$. For this example, the initial value of α has been set to 0.15. The purpose of assigning α to 0.15 that, when we tried to assign α to a lower value, the of FGPC was unstable and it makes no sense to include such value in the study. The responses are combined and illustrated in Figure 4.8

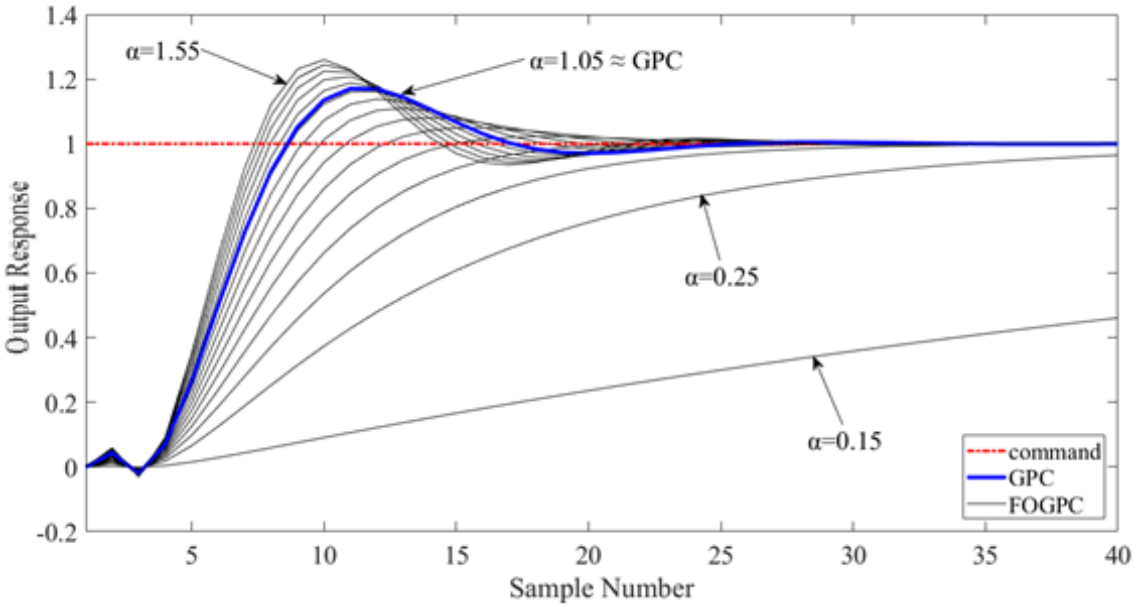


Figure 4.8: This plot illustrates the influence of α with fixed β over the response of FGPC

Clearly, from the observation of Figure 4.8, when α increases, FGPC tends to have a faster response with faster rising time. However, the bigger α gets, the larger the overshoot will be and there is a longer settling time. This result proves that adjusting α will give a different response based on the application specifications and requirements. On the other hand, smaller values of α will lead to a slower response of the FGPC controller, with longer rise time, and eventually, the FGPC will have an unstable response (i.e., for $\alpha < 0.135$). In addition, it is notable from Figure 4.8 above that, when $\alpha = 1.05$, the response of FGPC was almost the same as GPC. This result supports the fact that FGPC is a more general case of GPC and it can be tuned to have the same responses as GPC with the same model. Table 4.3 below summarises the findings.

Table 4.3 A comparison between different α values in terms of response time and overshoot percentage for case study 2

	α= 0.35	α= 0.45	α= 0.55	α= 0.75	α= 0.95	α= 1.05	α= 1.15	α= 1.25	α= 1.45	α= 1.55
Response time [seconds]	33.7	27.31	16.4	11.7	9.83	9.61	9.54	9.27	7.78	7.16
Overshoot [% of step up]	0%	0%	0.7%	9%	13%	16%	17%	19%	23%	29%

4.2.2 Changing β with fixed α

In this section, we will be investigating the effect of β on the response of FGPC by setting $\beta = 0$ and increasing it by a fixed value of 1 for 15 iterations (as increasing β with a small fraction will not have that much of effect in this model) with a fixed $\alpha = 0.45$. The responses are illustrated in the following Figure 4.9.

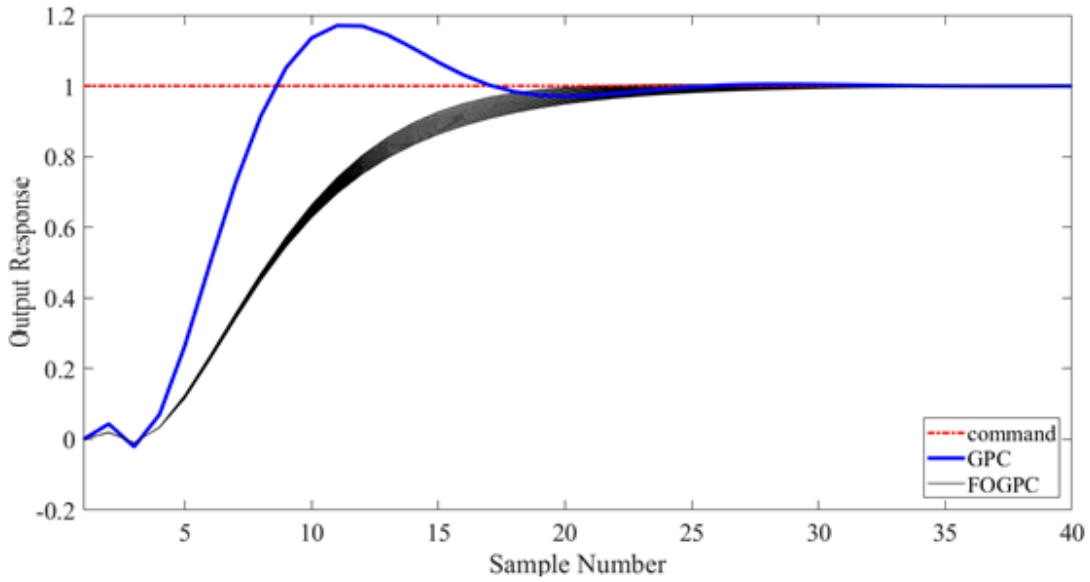


Figure 4.9: This plot illustrates the combined response of FGPC resulted from increasing β with fixed α

From Figure 4.9, we can observe that increasing β has not affected the response much; however, we can still see that increasing β will reduce the rising time and the FGPC controller will reach the desired point slower. Compared to the effect we observed by adjusting α, adjusting β has made no significant (minimal) impact on the FGPC response. Thus, a comparison table would make no sense.

4.2.3 Changing α and β together

In this section, we will try to find the best response of FGPC to achieve the fastest response with the minimum overshoot by adjusting both α and β coefficients only with the conventional GPC tuning parameters remaining constant (i.e., $N_{1,}N_2, N_u$). The methodology will be based on trial and error to achieve a better behaviour of FGPC in terms of the fastest responses with minimum overshoot. After several simulations, we have achieved a fast response with a reasonably low overshoot by adjusting $\alpha = 1.7$ and $\beta = 300$. Figure 4.10 below illustrates the response:

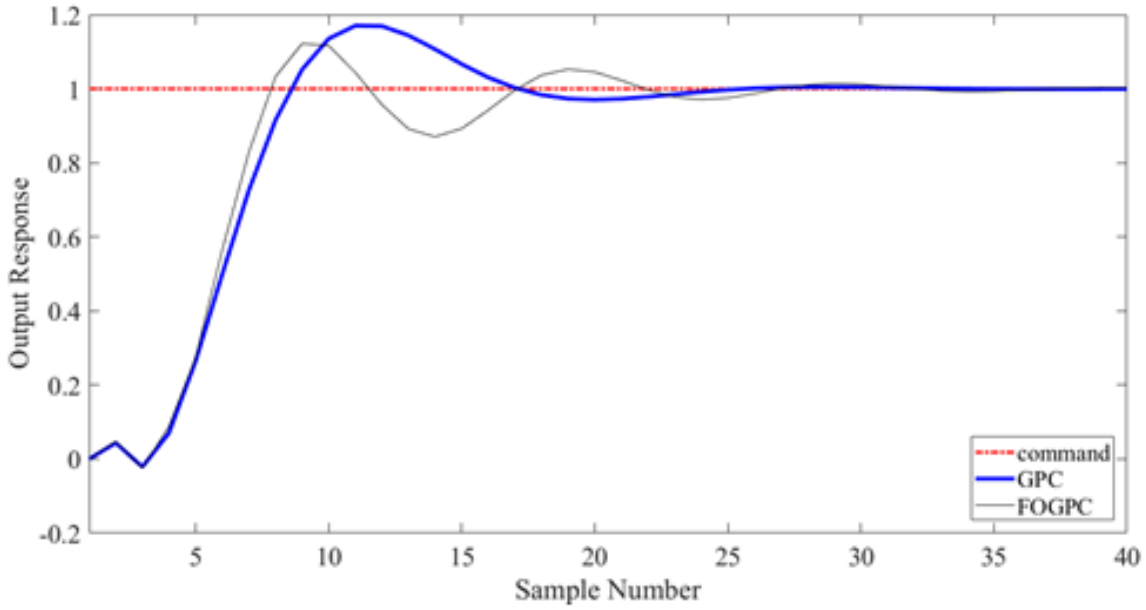


Figure 4.10: The best response of FGPC ($\beta = 300$ and $\alpha = 1.7$) in terms of the fastest responses with minimum overshoot

Figure 4.10 shows that the rise time for FGPC is faster than GPC and the overshoot of FGPC is lower than GPC. On the negative side, FGPC seems to oscillate more that GPC before it settles. The values of α and β can be optimised to achieve a better result of FGPC using the *fmincon* function in Matlab (a similar function has been used "*fminsearch"* in the next chapter 5.6.1).

4.3 Case study 3 (Marginally stable plant)

Consider the following marginally stable plant:

$$G(z^{-1}) = \frac{-z^{-2} + 2z^{-3}}{1 - 1.7z^{-1} + z^{-2}} \tag{4.6}$$

In this case study, we will compare the response of GPC and FGPC on the marginally stable plant (4.6). For that the following parameters have been set:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- No model mismatch
- For GPC, a constant values for (γ, λ), as $\gamma = 1, \lambda = 10^{-6}$
- For FGPC, We have chosen α and β randomly and assign initial values to them as $\alpha = 0.05, \beta = 1$. The basis that we have chosen α and β on were to have a reasonable response of FGPC with no specifications at all, which result in the following values for γ and λ, according to equation (3.46) and equation (3.47), respectively

$$\gamma = \begin{pmatrix} -0.9936 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0071 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0081 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0093 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0111 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0137 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0179 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0262 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0500 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\lambda = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

- The simulation ended at the final sample $T_f = 25$ with a sampling time $T_s = 1$

The response of both controllers along with their input variables can be found in the following Figure:

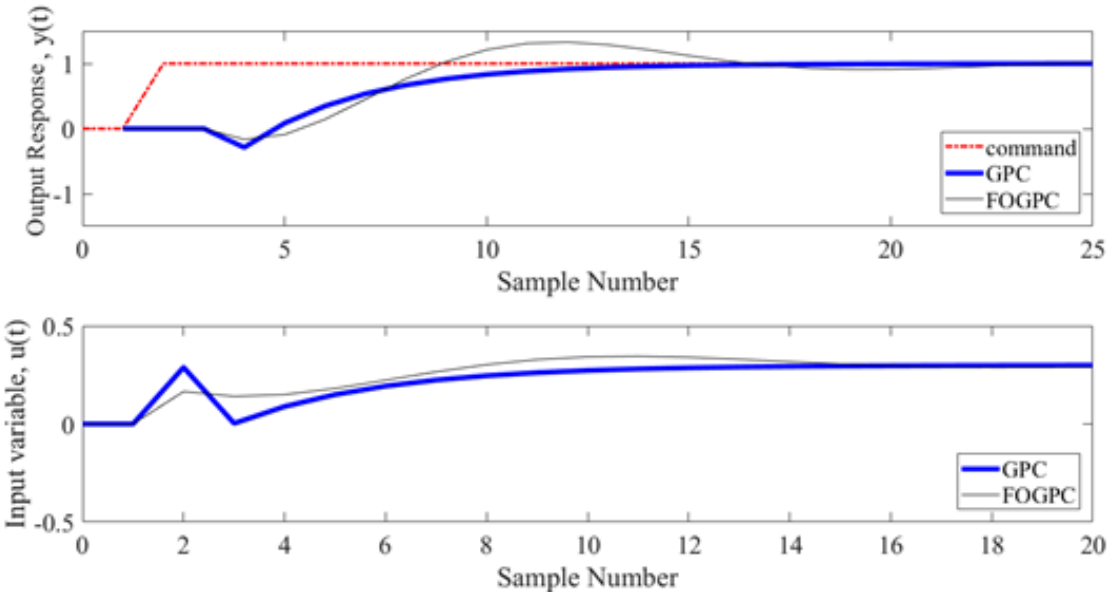


Figure 4.11: GPC and FGPC response for the plant in case study 3 plotted with the step input

The plot shown in Figure 4.11 illustrates that FGPC can have a different response by adjusting α and β with other design criteria remains constant (i.e., N_1, N_2, N_u). Although α and β of FGPC have been chosen arbitrarily for illustrative purposes (with a reasonably steady response), we can still notice that FGPC has a faster rising time than GPC. On the other hand, GPC has no overshoot, whereas FGPC has an overshoot.

Like in the previous sections, we will analyse the impact of α and β on the response of FGPC for this particular model with other criteria remains constant (i.e., N_1, N_2, N_u).

4.3.1 Changing α with fixed β

As demonstrated in the previous sections, α has an impact on the response of FGPC. To illustrate this response we will be using different values of α with a fixed value of β. The value

of α will start at 0.05 and will increase systematically by a fixed value of 0.01 for 20 iterations. The responses were combined in one plot. Figure 4.12 shows the responses:

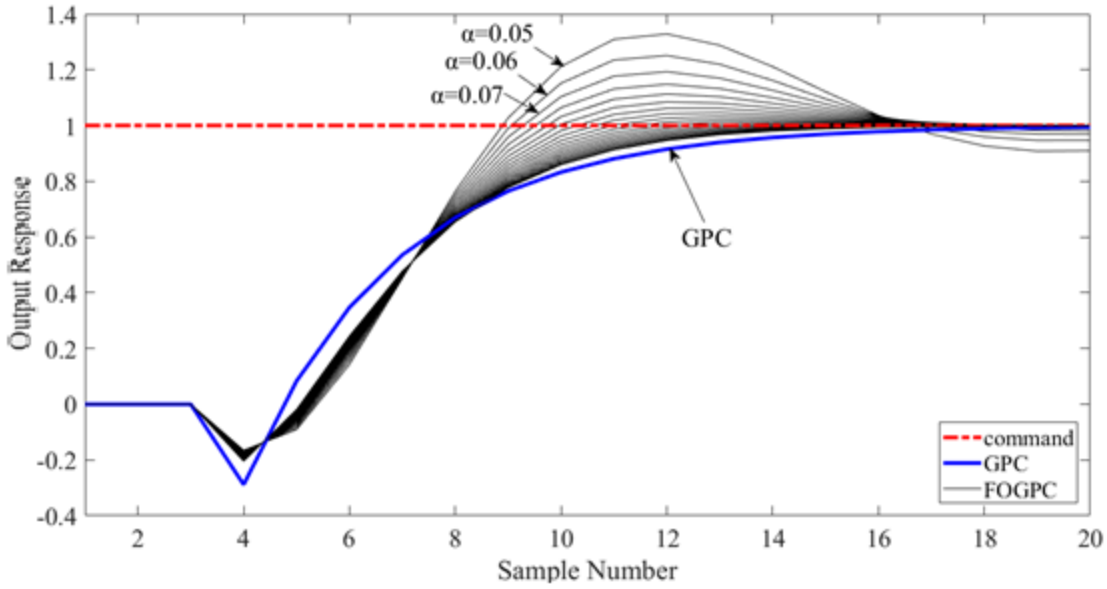


Figure 4.12 shows the effect of α on the response of FGPC with fixed β

It's noticeable that when α increases FGPC is tending to have less overshoot. When $\alpha = 0.15$ the overshoot was almost zero and the rising time was found to be faster than GPC. In this model, we have chosen to increase α by a very small value (i.e., 0.01) in contrast to the previous case studies when we increased α by 0.1 and the reason for that is because this model is marginally stable and α has a close response on it. In addition, we observe that as α increases, its effect is reduced as the response of FGPC tends to be very close. Table 4.4 below summarises the results.

Table 4.4 A comparison between different α values in terms of response time and overshoot percentage.

	α= 0.05	α= 0.07	α= 0.09	α= 0.11	α= 0.13	α= 0.15	α= 0.17	α= 0.19	α= 0.21	α= 0.25
Response time [seconds]	33.7	27.31	16.4	11.7	9.83	9.61	9.54	9.27	7.78	7.16
Overshoot [% of step up]	0%	0%	0.7%	9%	13%	16%	17%	19%	23%	29%

4.3.2 Changing β with fixed α

Now, we will try to investigate the effect of β by increasing it's value systematically while the value of α remains constant at 0.05. The initial value of β will be 0.5 and the ratio of increment will be 0.3 for 20 iterations. Figure 4.13 shows the combined plots of FGPC responses with different βs.

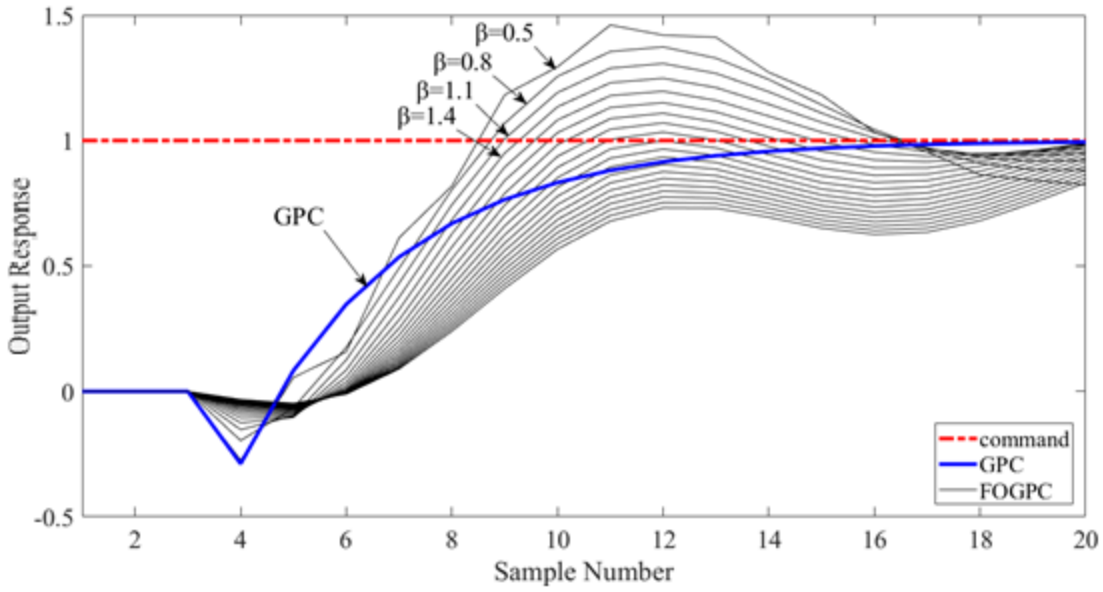


Figure 4.13: Illustration of the effect of changing the value of β on FGPC response

From the Figure above we can observe that β has an impact on changing the response of FGPC. The interesting outcome of this result is that FGPC has the same shape of response from $\beta = 1.7$ (i.e the fifth iteration) through to $\beta = 6.5$ (i.e., the twentieth iteration), whereas FGPC has different responses for the first 4 iterations of β. This indicates that, after a certain value of β, FGPC tends to take a certain shape of the response. In addition, we notice that, when β is bigger, the overshoot decreases and the rise time decreases as well. Obviously, β has a big impact on this model unlike the other models in the previous study cases. Table 4.5 below summarises the findings.

Table 4.5 A comparison between different β values in terms of response time and overshoot percentage for case study 3.

	β=0.5	β=0.8	β=1.1	β=1.4	β=1.7	β=2	β=2.3	β=2.6	β=2.9	β=3.2
Response time [seconds]	2.83	2.91	2.94	3.86	3.87	3.91	4.01	4.13	4.27	4.32
Overshoot [% of step up]	78%	23%	0.5%	7%	7%	19 %	19%	20%	19%	19%

4.3.3 Changing α and β

In this section, we will use the facts found in the previous sections (4.3.2 and 4.3.3) to optimise both α and β in an attempt to find the best response of FGPC compared to GPC based on the speed of response (rising time) with minimum overshoot, keeping in mind that the design parameters for both GPC and FGPC will be the same (i.e N_1, N_2, N_u) and remain constant ($N_1 =$

$1, N_2 = 10, N_u = 2$). The optimisation will be based on trial and error for both α and β. We will try several simulations with different values of α and β till we find the fastest response with minimum overshoot. Figure 4.14 shows the best match of those criteria:

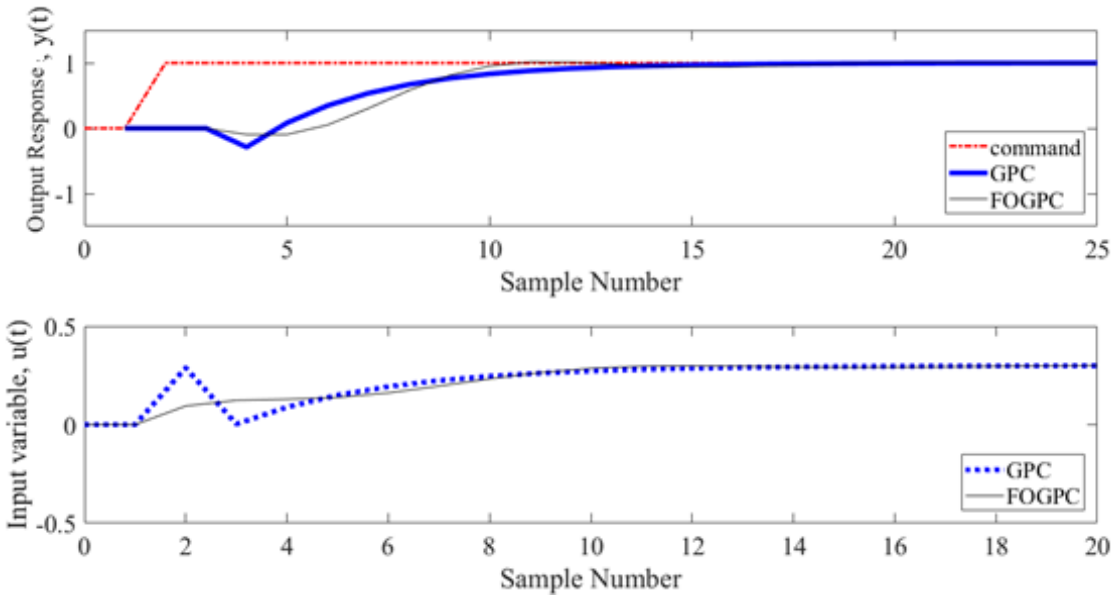


Figure 4.14 shows GPC and FGPC responses combined along with their input variables

After several simulations, we have identified α and β to be 0.077 and 2.25 respectively. Although the responses seem to be very close, FGPC managed with those values of α and β to have slightly faster responses with no overshoot. On the other hand, the input variable of FGPC seems to be much smoother than GPC.

4.3.4 Monte Carlo analysis

To test the robustness of the controller, we will perform MC analysis for both controllers by assigning model mismatch for both of them. As mentioned in section 4.1.3, we will vary the model parameters in an attempt to create a model mismatch then we will try to control the

model using both GPC and FGPC. This process will be done for 50 iterations which means 50 different models to control and that will allow us to create an envelope for both GPC and FGPC and then we will comment on the results based on the observation. The designing parameters will remain the same as stated in the previous sections and for FGPC we will use the optimised values of α and β we found in the previous section (i.e $\alpha = 0.077$ and $\beta = 2.25$). Figure 4.15 shows the resulting plots:

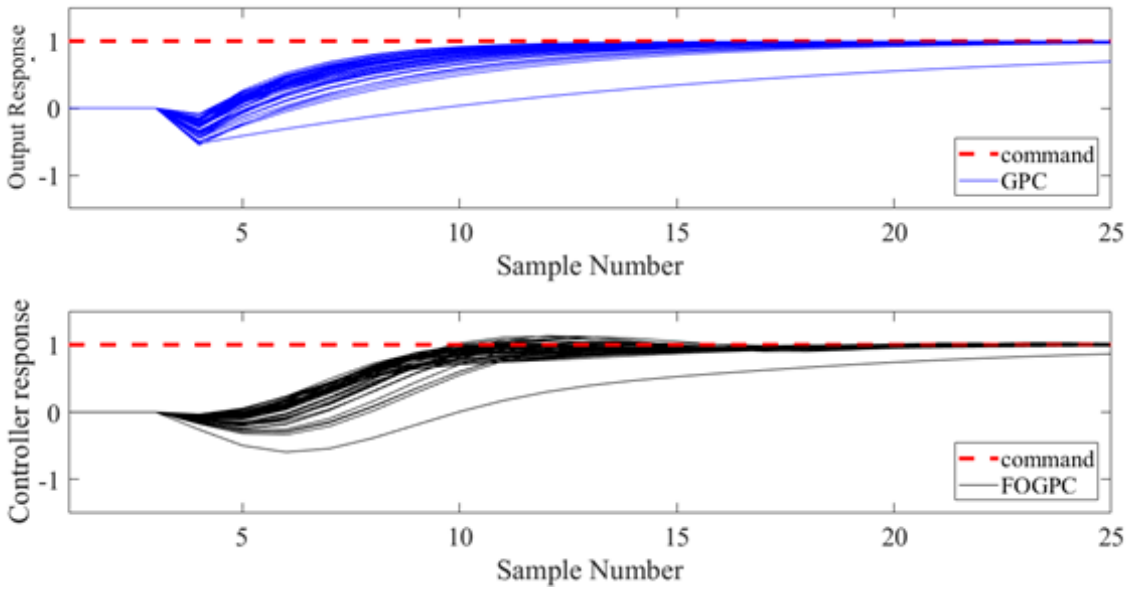


Figure 4.15 Illustration of MC analysis for both GPC and FGPC using the same designing parameters (i.e., N_1, N_2, N_u)

From the observation of the two plots, FGPC has an interesting response which illustrates the power of the extra tuning parameters. One particular result summarises the outcome of MC analysis of the controllers, which is the response in the very bottom of both controllers. We can see that FGPC has a faster and smoother response to that case whereas GPC is trying to catch up. In contrast to conventional GPC, some of the FGPC closed-loop responses overshoot the setpoint; however, what matters here is the overall response to model mismatch.

4.4 Discussion

The previous sections of this chapter show the utility of the FGPC approach, including some promising results that may lead to a new understanding of fractional-order controllers and their usage among both the academic and industrial sectors. In the first case study, we have used a simple first-order model to demonstrate the different responses of FGPC over GPC. This case study confirms the importance of selecting appropriate α and β values and shows how they affect the design of the FGPC controller and hence the closed-loop response. Since the α and β coefficients do not exist in GPC, they provide a benefit for FGPC. In some cases, the ability to change or tune the forecasting horizons (i.e. N_1, N_2, N_u) is quite limited; hence α and β will provide more 'space' in which to tune the controller to meet the system requirements without adjusting these horizons. This result has been confirmed by the following two case studies. The interesting fact here is that the impact of α and β on the closed-loop response and robustness properties of the controller differ according to the model. For example, different values of β with fixed α had a much more significant impact on the closed-loop response in case study 3 compared to case study 2. As discussed in Chapter 3, α and β are used to calculate γ and λ using equations (3.46) and (3.47) respectively. As a result, when we assign $\alpha = \beta = 1$ as a special case, this will not necessarily lead to the equivalence between FGPC and GPC, since we are tuning α and β, not γ and λ. In practical applications, GPC is usually based on $\gamma = 1$ and λ is set to be as minimal as possible (and in some designs is assumed to be zero). In addition, for FGPC, γ and λ are defined as matrices not scalar as is (most typically) the case for GPC.

In the following figures, we have optimised α and β in such a way to ensure an FGPC response approximately equal to the GPC response, for each case study example.

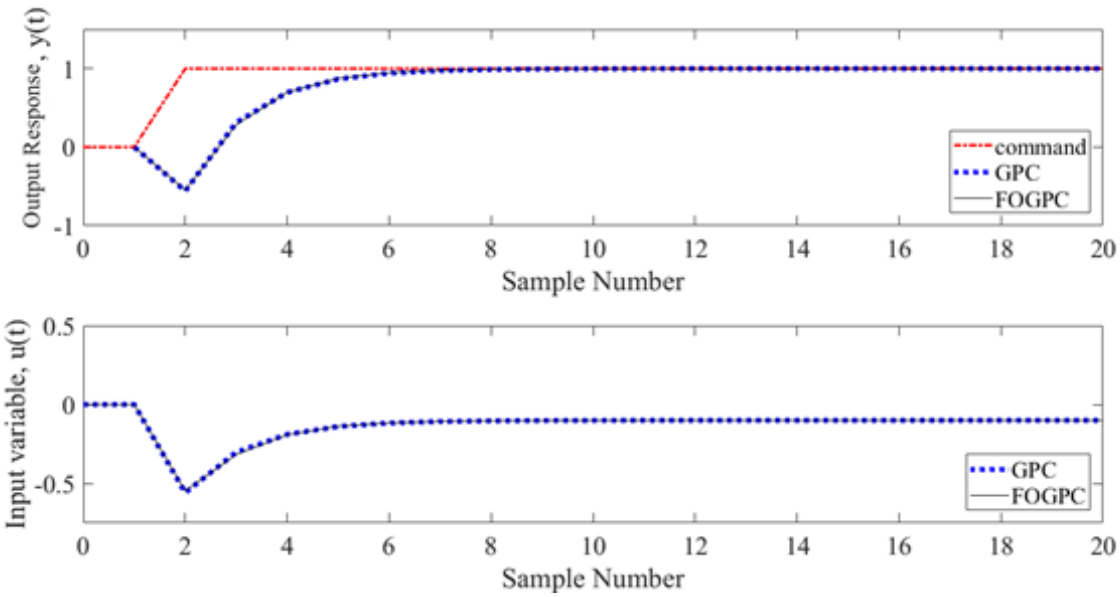


Figure 4.16: Case study 1 where $\alpha = 1.6$ and $\beta = 1.5$

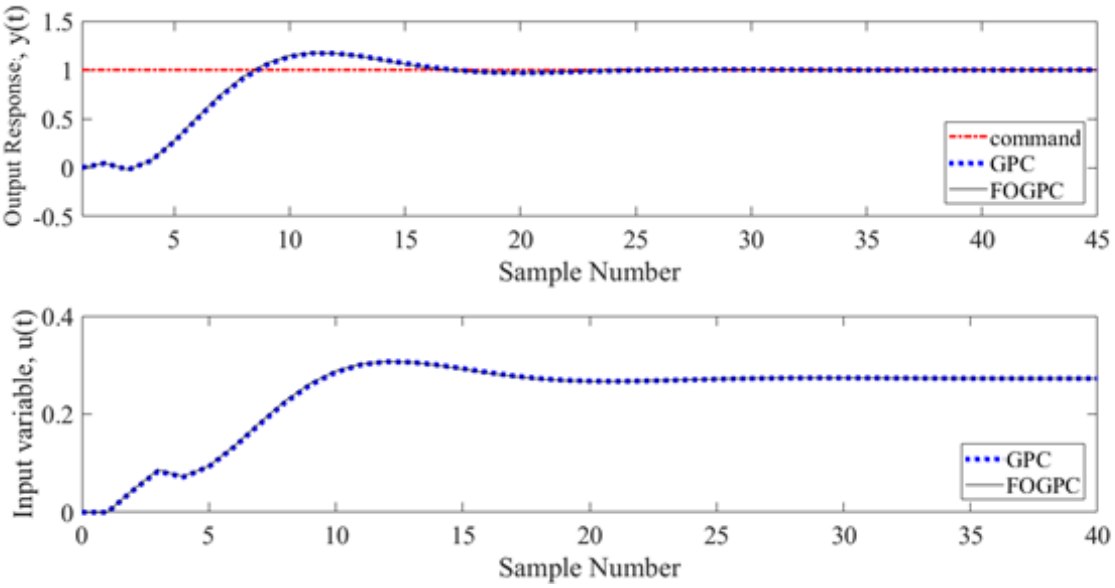


Figure 4.17 Case study 2 where $\alpha = 1.1$ and $\beta = 0.45$

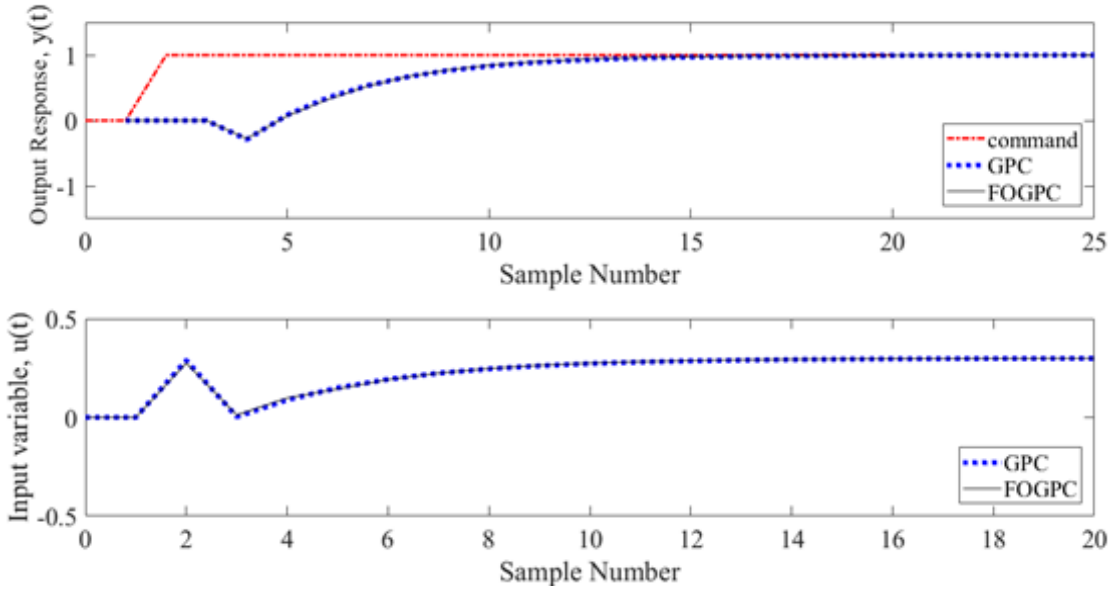


Figure 4.18 Case study 3 where $\alpha = 0.8$ and $\beta = 0.5$

The methodology used to find the matched responses of GPC and FGPC in Figures 4.16 to 4.18 is trial and error, based on the experience obtained from the simulation study concerning the effect of α and β on each model. Note that the values of α and β differ for each case study, supporting the conclusion that α and β affect the response in a manner that depends on the model itself, in addition to the various control settings.

Another point of interest is the MC simulation analysis concerning the robustness of the GPC and FGPC designs. The results might suggest that FGPC is more robust than conventional GPC, at least for the three case study problems under study here. However, it is not possible to generalise from these examples, and further research is required (see Chapter 7).

4.5 Concluding Remarks

This chapter has investigated GPC and FGPC design via a simulation study for three case study models. Evidence has been provided that α and β play an important role in the design process of FGPC. In this chapter, the selection of α and β was based on trial an error. However, it can be done automatically using specific MATLAB functions (please refer to the next chapters for further illustration). These coefficients can be utilised to help meet the control system requirements, either instead of or in addition to the use of the forecasting horizons and other parameters. The MC analysis has provided encouraging results that suggest FGPC could be more robust than GPC in some scenarios, but further research including practical experiments are required to support this result. In this regard, the following Chapter 5 addresses the practical implementation of FGPC design.

Chapter 5 Closed-Loop Eigenvalues

In this chapter, we will set another comparison between GPC and FGPC but from a different perspective. We will examine the eigenvalues of the GPC and FGPC closed-loop system. The aim is to determine how FGPC changes the eigenvalues of the system and how this will affect the time response. The following sections 5.1 and 5.2 provide a brief introduction to eigenvalues and illustrate their importance for applications in different sectors. Sections 5.3 to 5.5 consider three case study examples. Finally, the discussion and conclusions are presented in sections 5.7 and 5.8.

5.1 Introduction

Eigenvalues are a set of scalars associated with equations in a linear system. The eigenvector is the vector corresponding to the eigenvalue (Hoffman and Kunze, 1971). By computing eigenvalues and eigenvectors, we are one step closer towards understanding the linear transformation of a system. The importance of eigenvalues and eigenvectors in physics and engineering arises in common applications, such as stability analysis, the physics of rotating bodies, and small oscillations and vibrating systems. Some applications of eigenvalues and eigenvectors are:

1. Communication systems

   Claude Shannon has used the eigenvalues in his famous "information theory" to determine the limit of the information that can be transmitted through a communication medium (theoretically), such as a telephone line or through the air. Shannon used a water-filling algorithm (this algorithm is known in the communication community and is used for equalisation strategies on communication channels) on the communication channel's eigenvalues which represent the gains of the fundamental modes of the channel (Shannon, 1993).

2. Designing bridges

   Eigenvalues are used in structural stability analysis design to analyse the vibrations on the bridge to prevent it from collapsing. This objective can be achieved by determining the eigenvalues of the framework and by applying several tests in both full loads and freeload on the proposed design (Masur, 1984).

3. Electrical engineering (Kalman Filter)

   Using higher order of Kalman filter has its drawbacks. For instance, there will be a lack of insight into the nature of observability of the system. Determining the eigenvalues and the eigenvectors of the error covariance matrix will provide useful information that could assist in providing a proper vision of the nature of the observability of the system (Ham and Brown, 1983)

4. Control systems

   Eigenvalues play a major role in control theory as they represent the system poles of the transfer function in LTI systems. Determining the system's poles is essential to

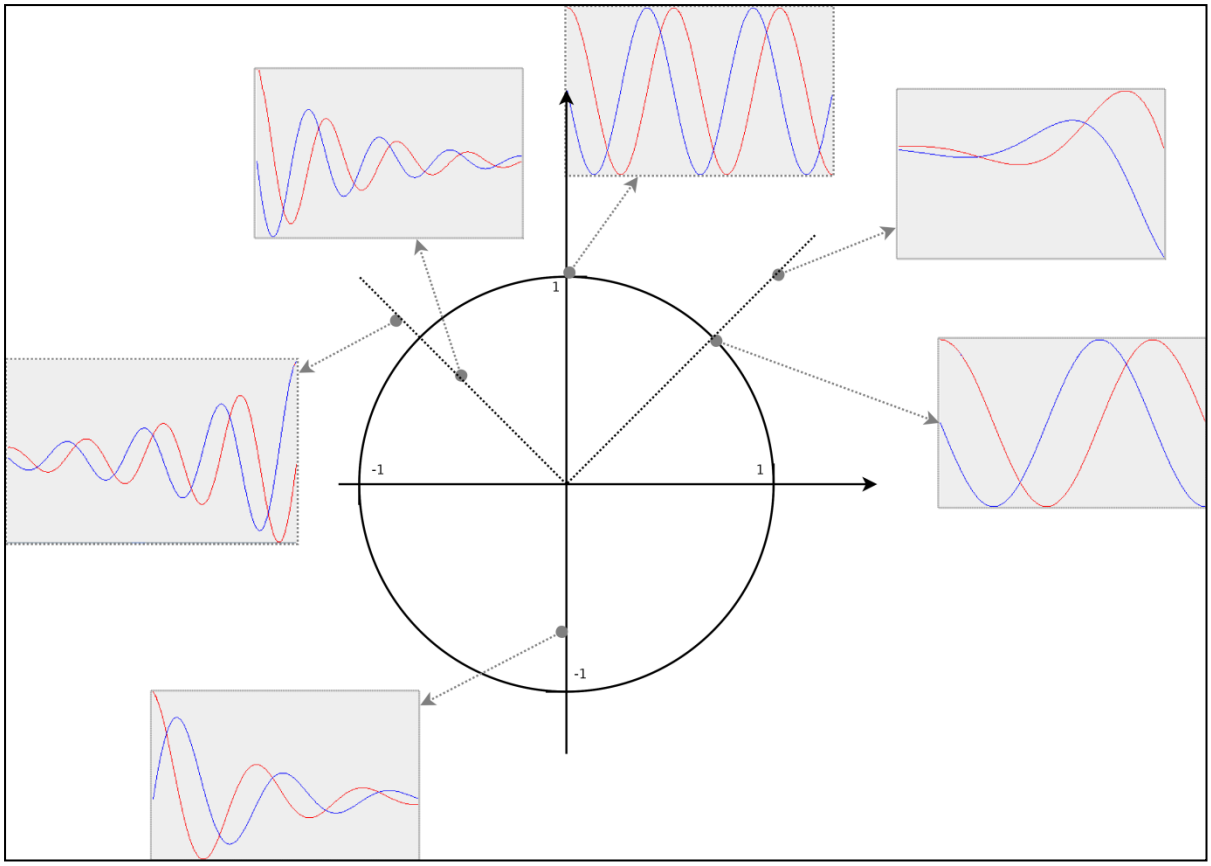
measure the system stability; thus, it is significant to understand and find the eigenvalues of a control system.

5.2 Eigenvalue significance in control theory

As stated in the introduction to this chapter, the eigenvalues are essential in designing control systems.

In this chapter, we have determined the differences between GPC and FGPC in terms of the poles' locations which subsequently affect the response of the system to the controller. This has been achieved by running a Monte Carlo analysis using the design of different parameters, such as: N_2, N_u and the fractional-order coefficients α and β. In Chapter 4, I studied the effect of α and β on the system by manipulating their values, which had an impact on the output. For the sake of consistency and to link the ideas of the thesis together, we have employed the same examples in the present chapter.

In discrete systems, as we are using discrete systems throughout the thesis, the locations of the poles in the z-plane (eigenvalues) shape the response of the control system. For example, if any poles are located outside the unit circle, the system will be considered unstable (Franklin *et al.*, 2008). Even if all the poles of the system are within the unit circle, the location of the pole will determine the shape of the response e.g. the damping. As is well-known, the z-plane can be divided into 4 quarters, upper right, upper left, lower right, and lower left. Each quarter has a different response if the pole is located in them. To shed more light on the topic, Figure 5.1 below illustrates the response of a sine wave in different pole locations.



5.1: Sine responses for different poles locations within and outside the unit circle

The following case studies will provide us with an in-depth understanding of the influence of poles locations on the controller responses.

5.3 Case Study 1 (Simple Model)

Recalling the following simple system from the earlier chapter (equation (4.1))

$$G(z^{-1}) = \frac{1 - 2z^{-1}}{1 - 0.9z^{-1}} z^{-1}$$

Assuming the parameters set for controlling this model are the following:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.77$
- The fractional-order coefficient $\beta = 1$

We have chosen these values for the control parameters based on the Chapter 4 simulations as these values showed (based on trial and error technique) the best response for FGPC compared to GPC in terms of fast response and settling time. Thus, we were comparing the locations of the poles of FGPC and GPC based on the best response of FGPC.

5.3.1 Observing the effect of the output forecasting horizon (N_2)

Based on trial and error, we have varied the value of the output forecasting horizon N_2 to start from 4 and increased by a fixed value of 1 for 100 iterations. These values were chosen in order to achieve a stable response for both controllers; for instance, if N_2= 3, FGPC response will have huge overshoot. The other control's parameters remain constant as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.77$
- The fractional-order coefficient $\beta = 1$

N_2 cannot be less than 3 as the matrix dimensions will not match. Figure 5.1 shows the response of both GPC and FGPC to the model, whereas Figure 5.2 shows the simulated result of the poles' locations for both GPC and FGPC.

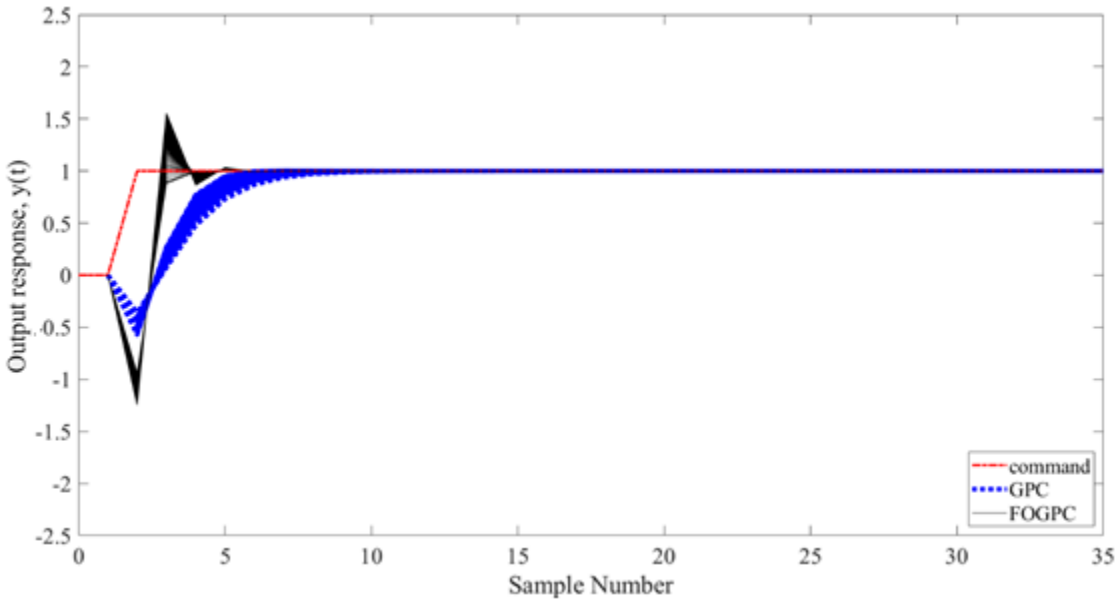


Figure 5.2: FGPC and GPC responses with N_2 varying from 3 to 103 by a fixed value of 1

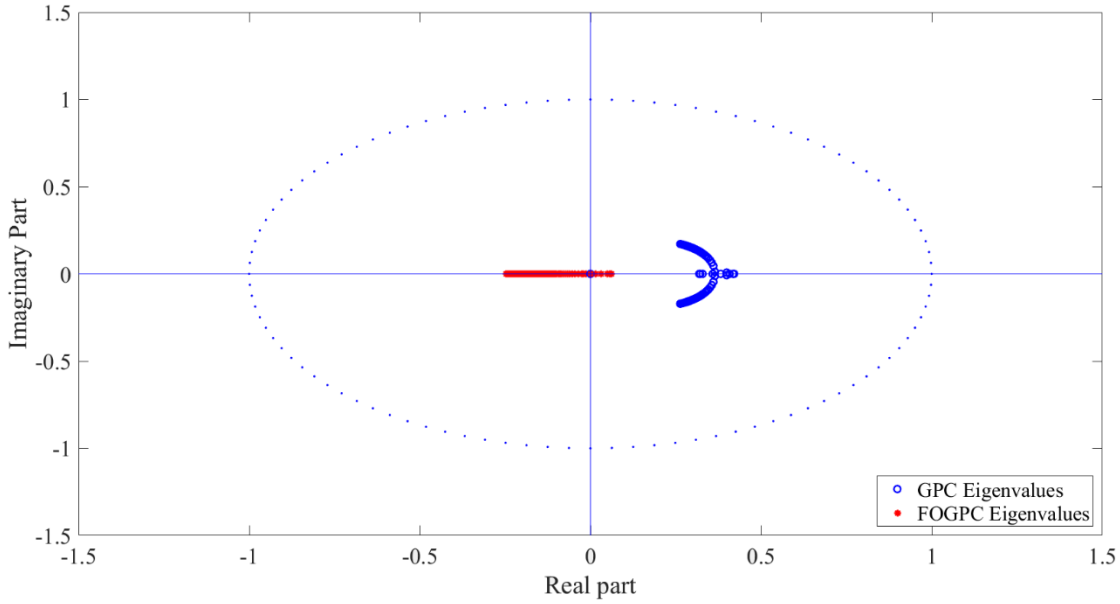


Figure 5.3: Poles locations comparison between FGPC and GPC with N_2 varying from 4 to

104

As the figure indicates, FGPC's poles have no imaginary parts. Another observation from the figure is that FGPC has most of its poles on the left-hand side of the unit circle and much closer to the origin, which subsequently affects the response to the model.

5.3.2 Observing the effect of input forecasting horizon (N_u)

In this section, we have observed the effect of the input forecast horizon N_u on the pole's locations of both GPC and FGPC. N_u has been assigned to 1 initially, and to be increased by a fixed value of 1 for 10 iterations, while the other design parameters remain constant as follows:

- Input forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.77$
- The fractional-order coefficient $\beta = 1$

By definition, we can't assign N_u to be greater than N_2 and it can't be assigned to zero as well. Figure 5.3 shows the responses of both GPC and FGPC to varying N_u and Figure 5.4 shows the simulated result of the poles' locations for both GPC and FGPC.

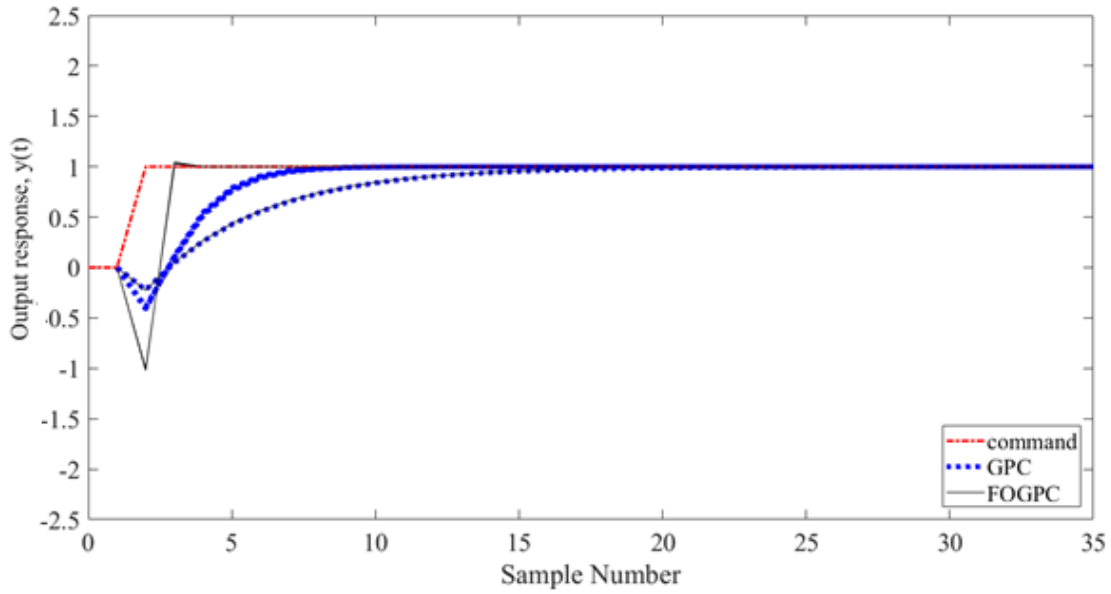


Figure 5.4: FGPC and GPC responses with N_u varying from 1 to 10 by a fixed value of 1

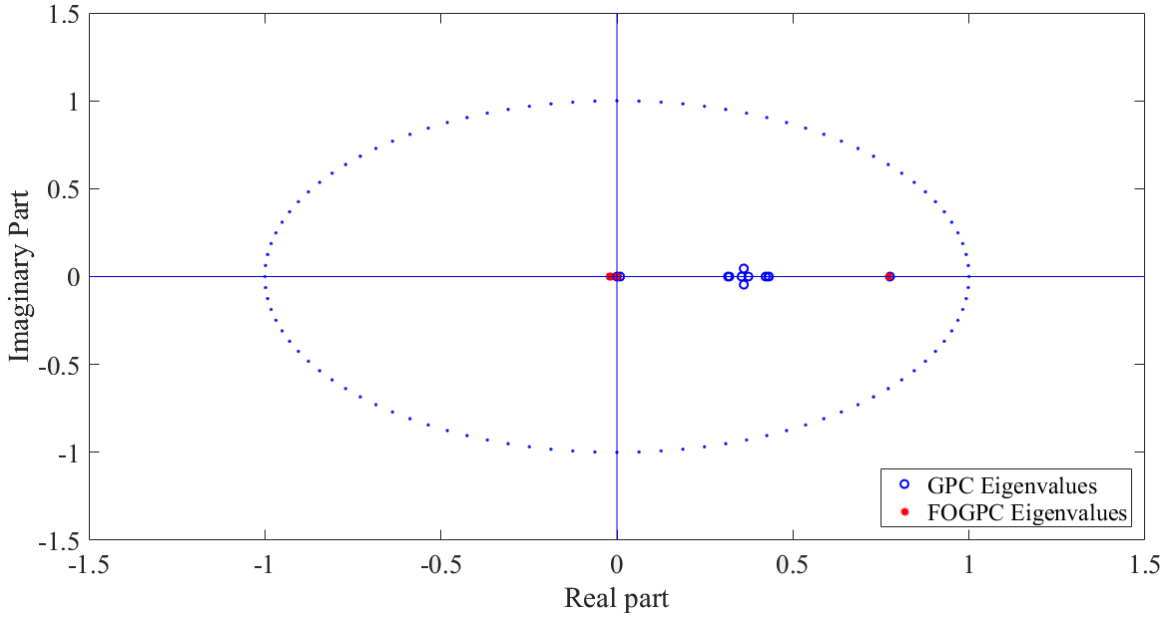


Figure 5.5 Poles locations comparison between FGPC and GPC with N_u varying from 1 to 10

As Figure 5.5 indicates, all of the GPC's poles are on the right-hand side. Some of the GPC's poles have imaginary parts. In addition, all FGPC's poles have no imaginary part.

5.3.3 Observing the effect of the fractional-order weighting (α)

In this sub-section, we have studied the effect of α on the pole locations. We have assigned an initial value for $\alpha = 0.5$ and increased it by a fixed value of 0.01 for 100 iterations. The other parameters were assigned as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\beta = 1$

Figure 5.5 illustrates the response of both GPC and FGPC while Figure 5.6 illustrates the pole locations corresponding to the responses.

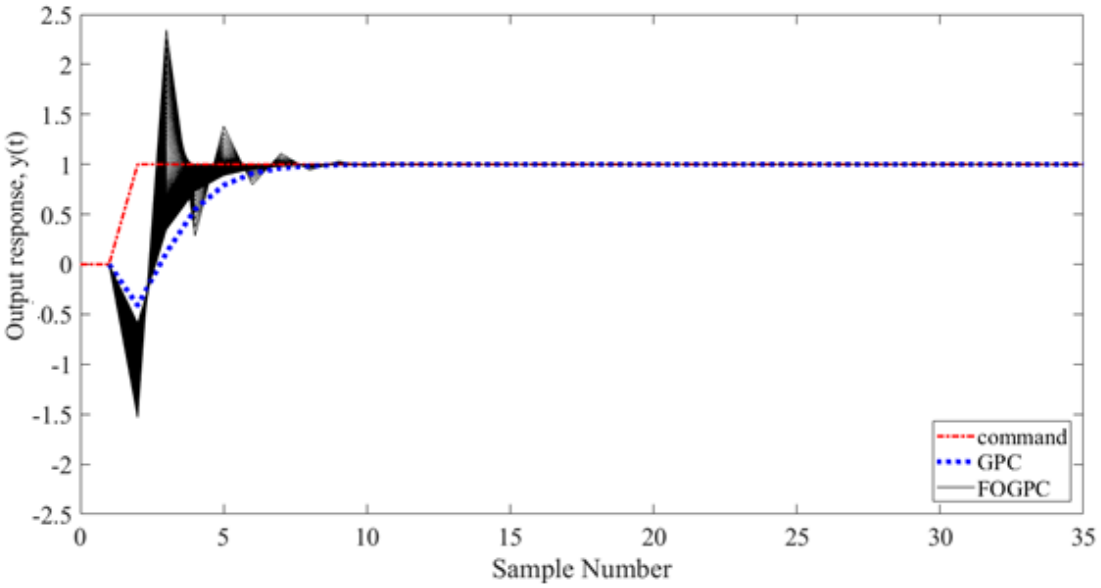


Figure 5.6: FGPC and GPC responses to the model with α varying from 0.5 to 1.5 with a fixed value of 0.01

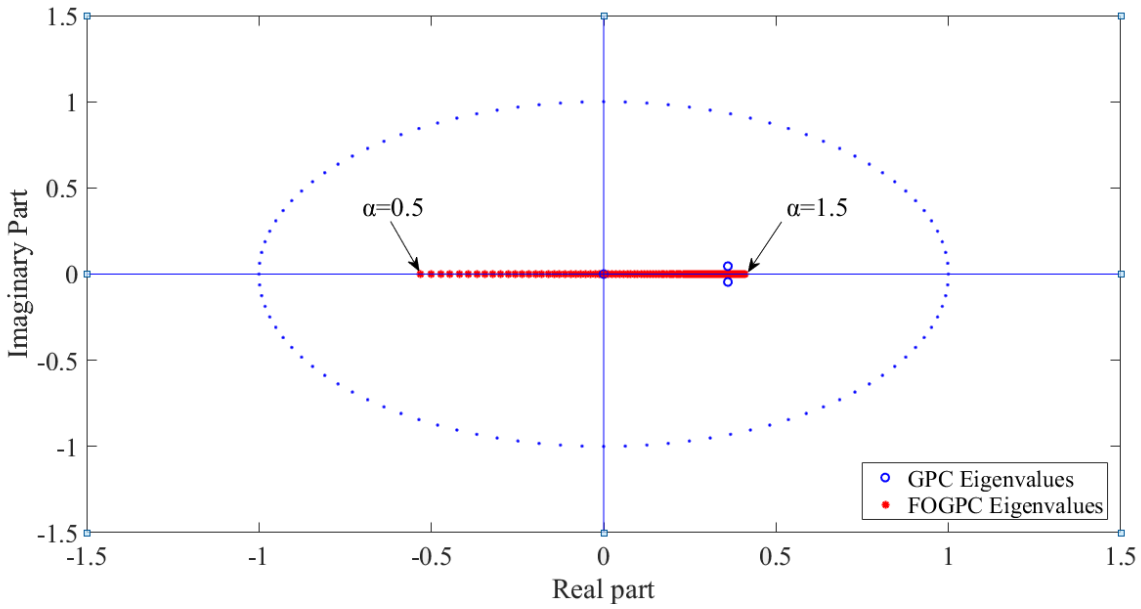


Figure 5.7: Poles locations for both GPC and FGPC with α varying from 0.5 to 1.5 with a fixed value of 0.01

The Figures above show that α has a direct effect on the pole locations which subsequently affect the response of FGPC on the model. Lower values of α yield a pole on the left-hand side and, as α increases, the pole location moves toward the right-hand side. In addition, we noticed that the change of α has no effect of introducing imaginary parts (at least in this case).

5.3.4 Observing the effect of the fractional-order weighting (β)

As we have observed in the previous sub-section, α has a direct effect on the pole location as well as the response to the model. This sub-section is dedicated to inspecting the effect of β on the response to the model and the location of the poles. We have assigned $\beta = 1$ and increased it by a fixed value of 0.1 for 50 iterations. The other design parameters were left constant as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.77$

Figures 5.7 and 5.8 shows the response of the model to the varying β and the poles' locations corresponding to that response respectively.

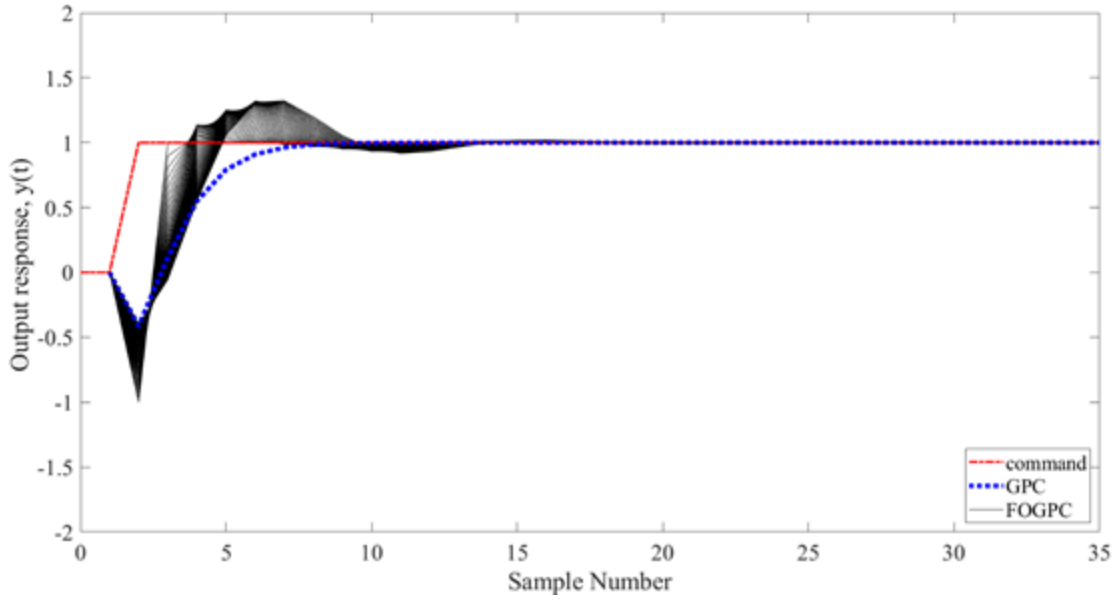


Figure 5.8: FGPC and GPC responses to the model with β varying from 1 to 6 with a fixed value of 0.1

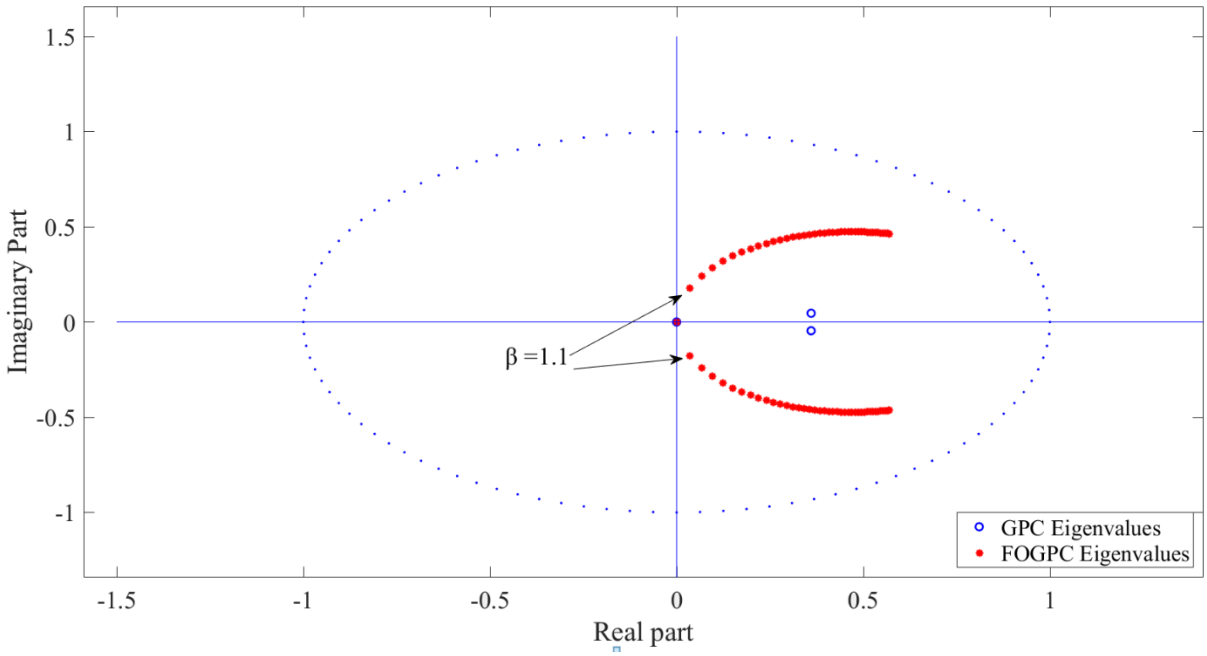


Figure 5.9: Poles locations for both GPC and FGPC with β varying from 1 to 6 with a fixed value of 0.1

Figures 5.7 and 5.8 suggest β affects both the response to the model and the pole locations corresponding to the response. The initial value of β has a pole located in the very far left-hand side. As $\beta = 1$, the poles are located in the origin; however, when β is set more than 1 (i.e., 1.1), the poles have started to develop an imaginary part.

5.4 Case study 2 (Higher-order model)

In this section, we have used a higher-order plant to observe the effect of design parameters (i.e., N_2, N_u and fractional-order coefficients α and β) on the pole locations and their corresponding responses. Recalling the plant used in the earlier chapter, equation (4.5):

$$G(z^{-1}) = \frac{1-3z^{-1} + 5z^{-2}+0.3z^{-3}}{1 - 0.6z^{-1} - z^{-2} + 1.5z^{-3}}$$

with the following design parameters:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- No model mismatch
- For GPC, a constant values for (γ, λ), as $\gamma = 1$, $\lambda = 10^{-6}$
- For FGPC coefficients, We have chosen α and β based on trial and error to find the best response for FGPC to this specific model in terms of fastest response with minimum settling time; thus, we have assigned an initial value as $\alpha = 1.9$, $\beta = 0.5$.

We have chosen those parameters of FGPC coefficients based on Chapter 4, which shows the optimal response for FGPC.

5.4.1 Observing the effect of the output forecasting horizon (N_2)

As we did in the previous case study, we have varied the value of the output forecasting horizon N_2 to start from 4 and increased by a fixed value of 1 for 5 iterations while the other control parameters remain constant as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 80$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 1.9$

- The fractional-order coefficient $\beta = 0.5$

N_2 can't be less than 4 as the matrix dimensions will not match. Figure 5.9 shows the responses of both GPC and FGPC to the model. Figure 5.10 shows the simulated result of the locations of the corresponding poles for both GPC and FGPC.

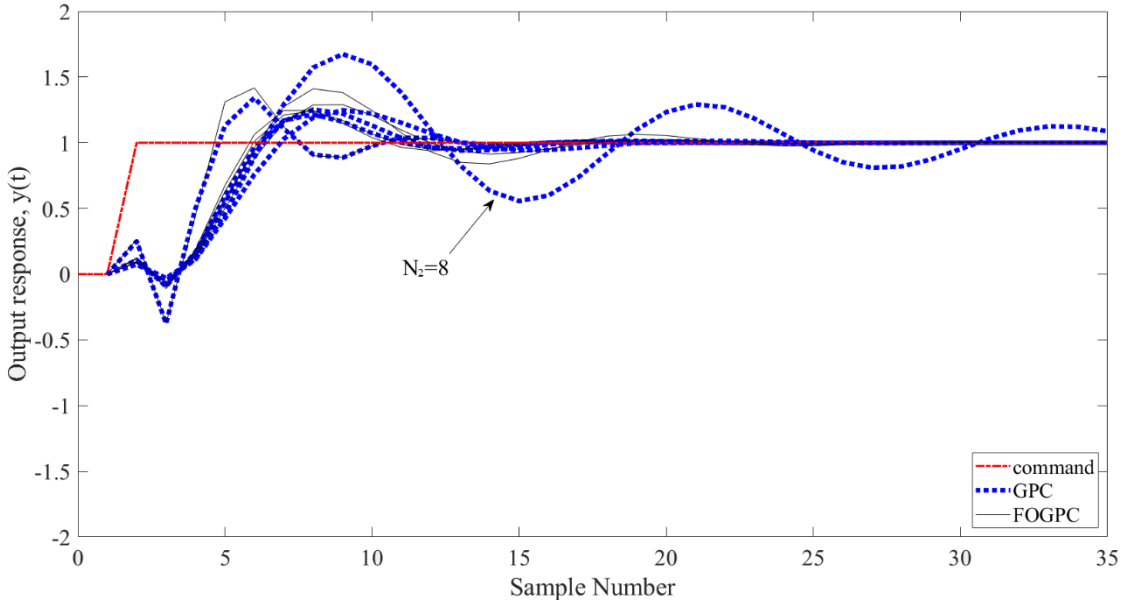


Figure 5.10: FGPC and GPC responses with N_2 varying from 4 to 8 by a fixed value of 1

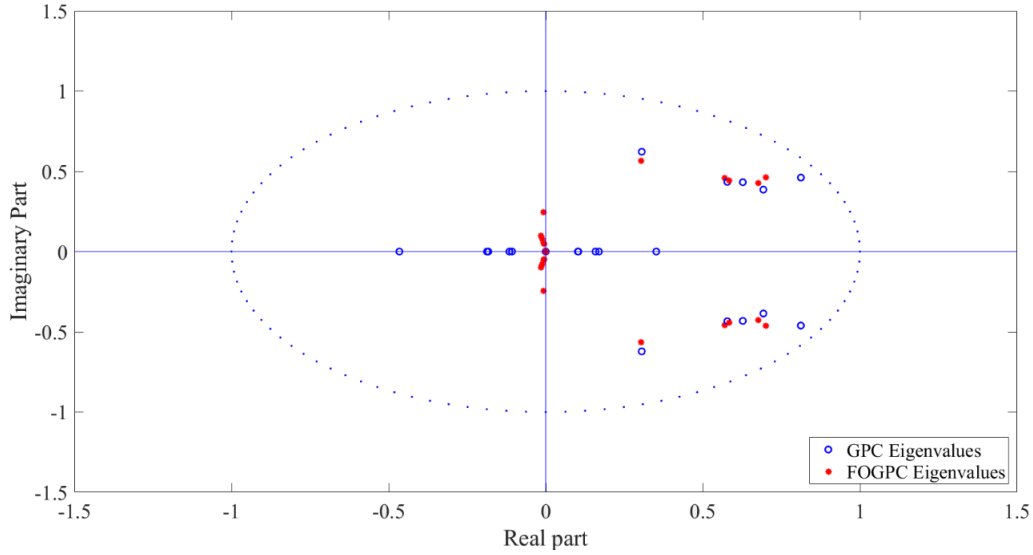


Figure 5.11: Poles locations comparison between FGPC and GPC with N_2 varying from 4 to 8

By observing Figure 5.10, we notice that when N_2= 8, FGPC showed a better response than GPC in terms of no overshoot and faster steady state time. Besides, we have noticed that most of FGPC's poles have imaginary parts, in contrast to GPC which shows the effect of the fractional-order coefficients α and β.

5.4.2 Observing the effect of the input forecasting horizon (N_u)

In this section, we have observed the effect of the input forecast horizon N_u on the pole's locations of both GPC and FGPC. N_u has been assigned to 2 initially, and to be increased by a fixed value of 1 for 8 iterations, while the other design parameters remain constant as follows:

- Input forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 1.9$
- The fractional-order coefficient $\beta = 0.5$

By definition, we can't assign N_u to be greater than N_2 and it can't be assigned to zero as well. Figure 5.11 shows the responses of both GPC and FGPC to the varying N_u and Figure 5.12 shows the simulated result of the poles' locations for both GPC and FGPC.

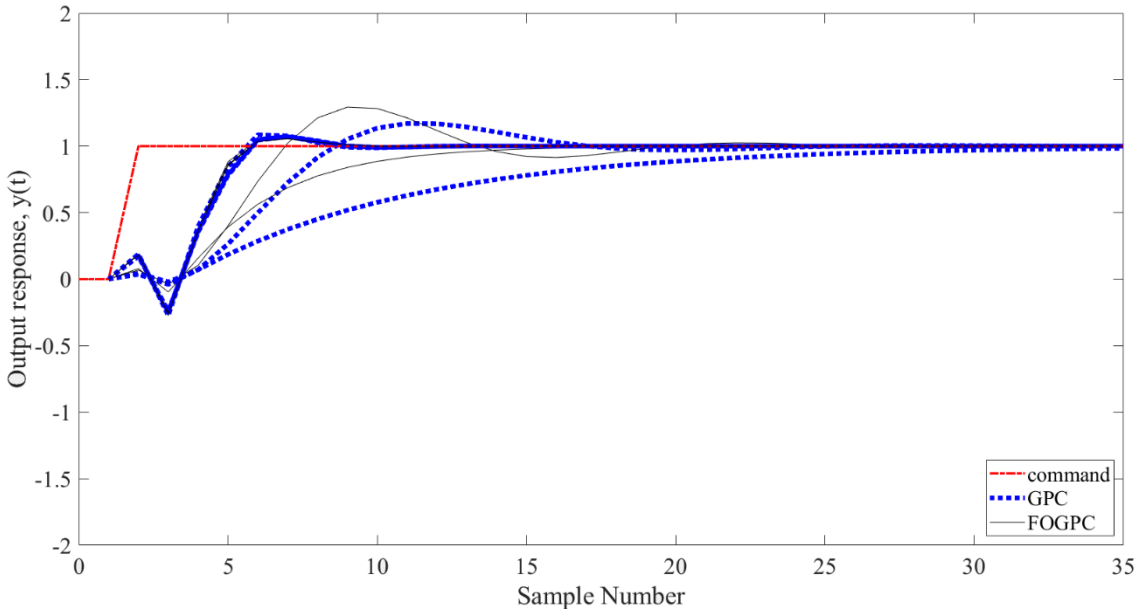


Figure 5.12: FGPC and GPC responses with N_u varying from 2 to 9 by a fixed value of 1

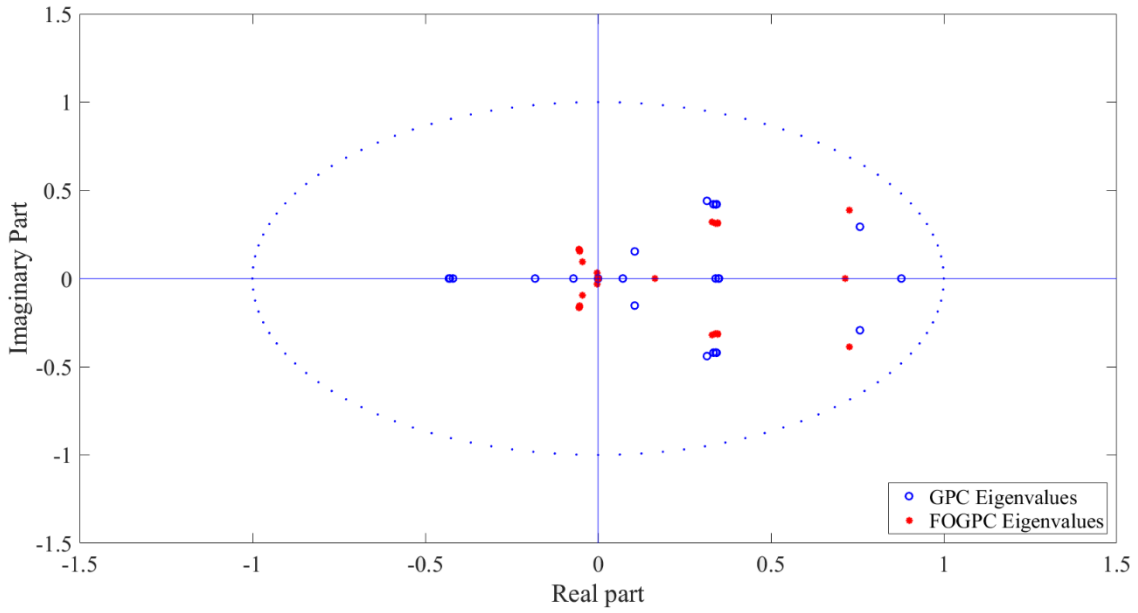


Figure 5.13: Poles locations comparison between FGPC and GPC with N_u varying from 2 to 9

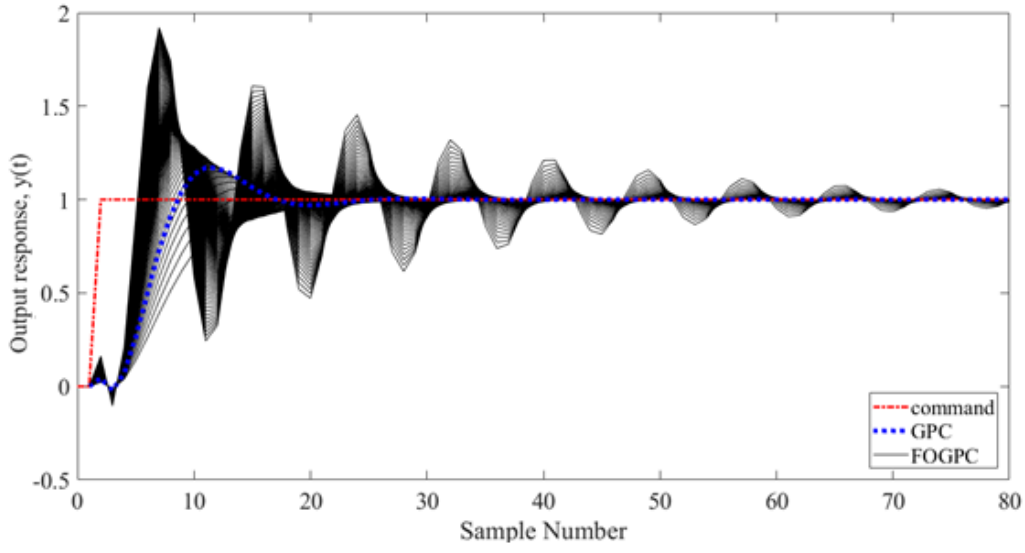
By observing the findings in Figure 5.11 and 5.12, we have found that FGPC was more robust than GPC even with the marginally stable poles. In addition, most of the FGPC's poles were found to have imaginary parts, whereas most of the GPC ones didn't have imaginary parts.

5.4.3 Observing the effect of the fractional-order weighting (α)

In this sub-section, we have studied the effect of α on the pole locations. To do so, we have assigned an initial value for $\alpha = 0.5$ and increased it by a fixed value of 0.1 for 100 iterations Whereas, the other parameters were assigned as the following:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 80$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\beta = 0.5$

Figure 5.13 illustrates the response of both GPC and FGPC while Figure 5.14 illustrates the poles' locations corresponding to the responses.



5.14: FGPC and GPC responses to the model with α varying from 0.5 to 10.5 with a fixed value of 0.1

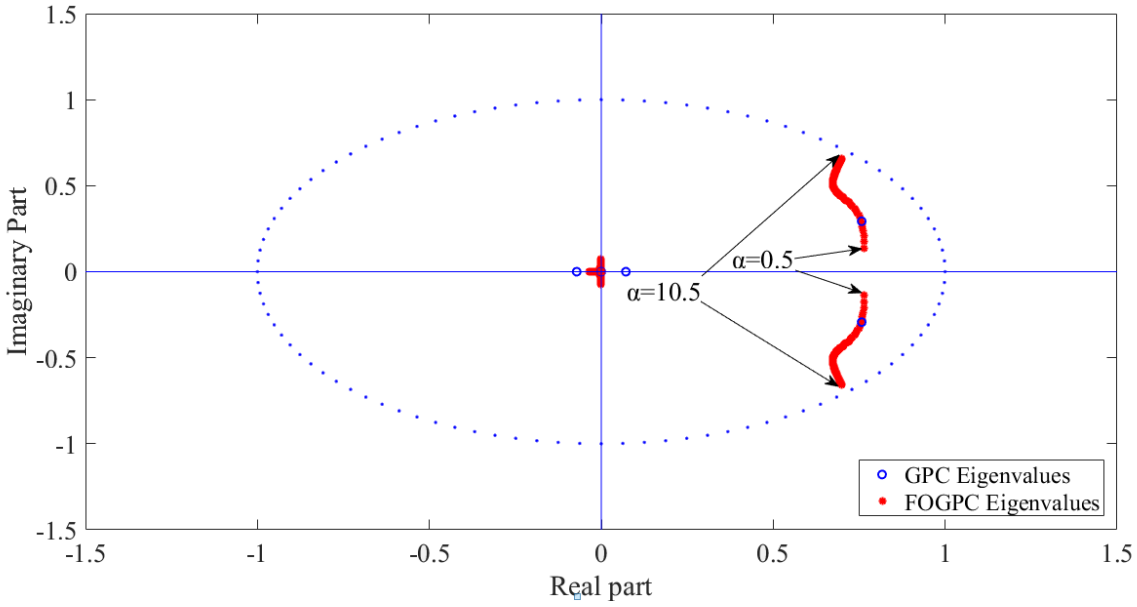


Figure 5.15: Poles locations for both GPC and FGPC with α varying from 0.5 to 10.5 with a fixed value of 0.1

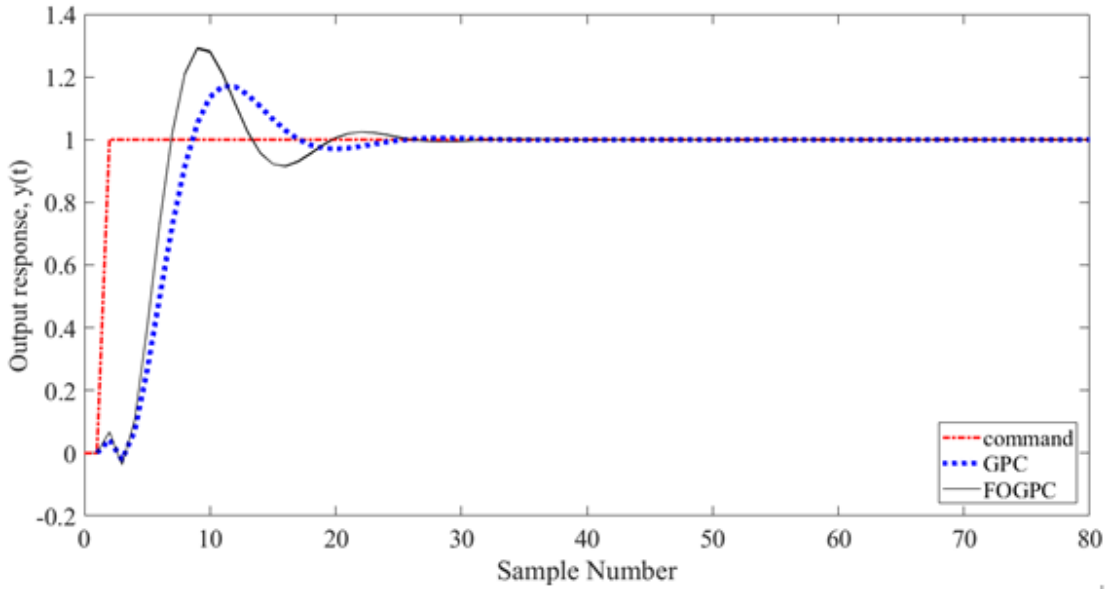
Based on the observation on Figures 5.13 and 5.14, we have found that when the value of α is getting larger, the poles tend to move further towards the border of the unit circle, and eventually will be located outside the unit circle, which causes the instability of the controller. Also, we have noticed that changing the value of α will increase the value of the imaginary part.

5.4.4 Observing the effect of the fractional-order weighting (β)

As we have concluded in the previous sub-section, α has a direct effect on the pole locations as well as the response to the model. This sub-section considers the effect of β on the response to the model and the pole locations. We have assigned $\beta = 0$ and increased it by a fixed value of 0.1 for 100 iterations. The other design parameters were left constant as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 80$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 1.9$

Figures 5.15 and 5.16 shows the response of the model to the varying β and the pole locations corresponding to that response respectively.



5.16 FGPC and GPC responses to the model with β varying from 0 to 10 with a fixed value of 0.1

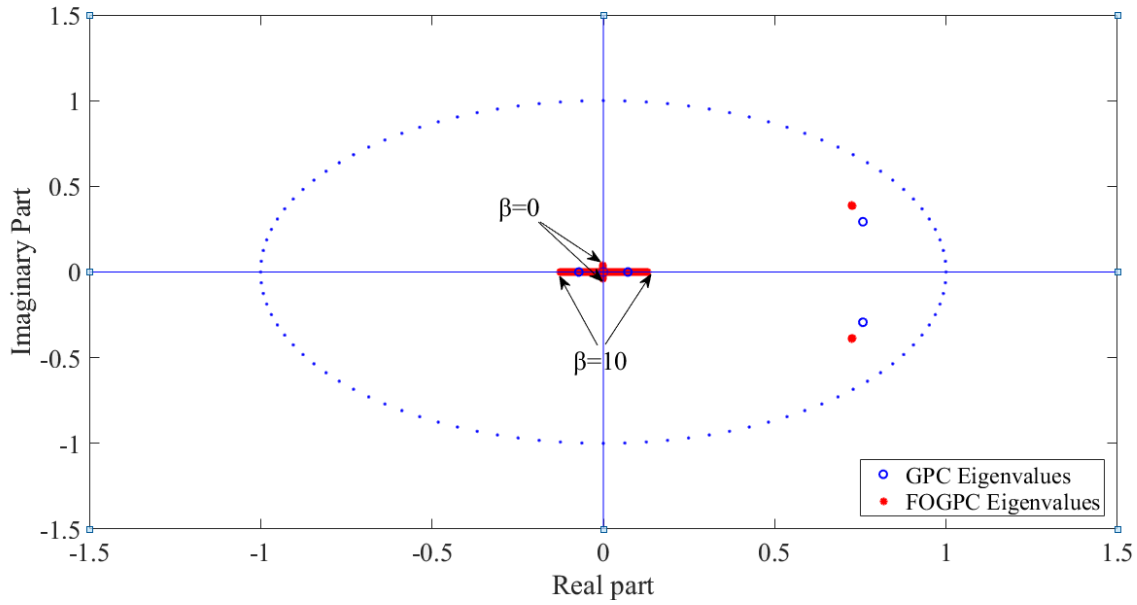


Figure 5.17 Poles locations for both GPC and FGPC with β varying from 0 to 10 with a fixed value of 0.1

Figures 5.15 and 5.16 illustrate that changing the value of β has minimal (almost none) effect on the response. However, we have noticed that the poles with no imaginary part have started to move away from the origin and towards the border of the unit circle and eventually outside the unit circle with a higher value of β.

5.5 Case study 3 (Marginally stable plant)

Recalling the plant from the earlier chapter (equation (4.6)):

$$G(z^{-1}) = \frac{-z^{-2} + 2z^{-3}}{1 - 1.7z^{-1} + z^{-2}}$$

With the following design parameters:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- No model mismatch
- For GPC, constant values for (γ, λ), as $\gamma = 1$, $\lambda = 0$
- For FGPC coefficients, we have chosen α and β based on trial and error to find the best response for FGPC to this specific model in terms of fastest response with minimum overshoot and settling time; thus, we have assigned an initial value as $\alpha = 0.08$, $\beta = 2.1$.

Similar to the previous sections, we have studied the effect of various design parameters on the response to the model and their corresponding pole locations.

5.5.1 Observing the effect of the output forecasting N_2

We have examined the effect of the output forecasting horizon N_2 by varying the value of it from 6, and increased by a fixed value of 1 for 6 iterations while the other control parameters remain constant as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 50$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.05$
- The fractional-order coefficient $\beta = 2.1$

N_2 can't be less than 3 as the matrix dimensions will not match and we have assigned an initial value of 6 (since when we assigned it to 5, both controllers were unstable). Figure 5.17 shows the responses of both GPC and FGPC to the model. Figure 5.18 shows the simulated result of the corresponding poles' locations for both GPC and FGPC.

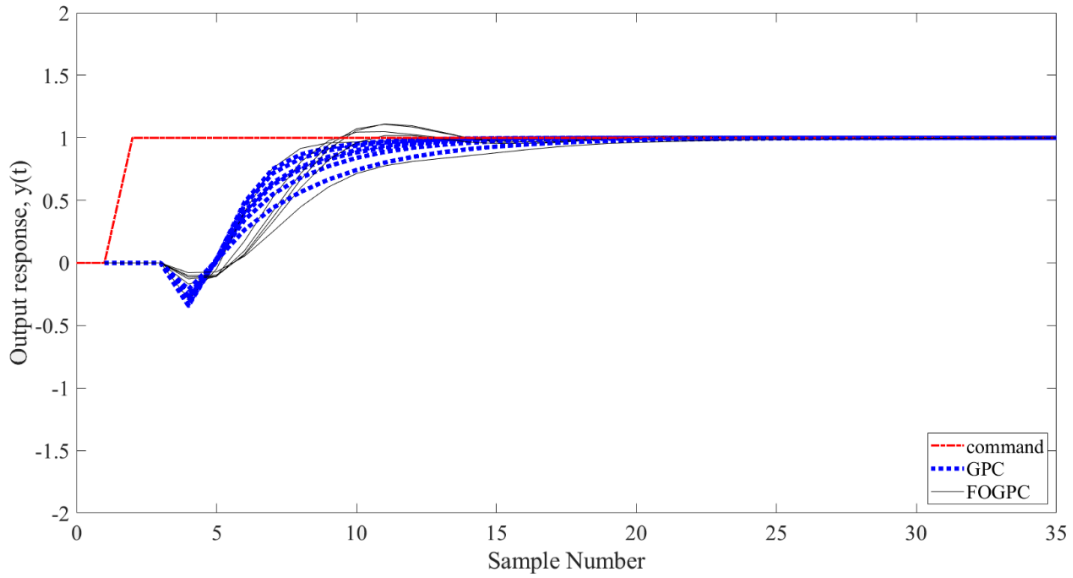


Figure 5.18: FGPC and GPC responses with N_2 varying from 6 to 11 by a fixed value of 1

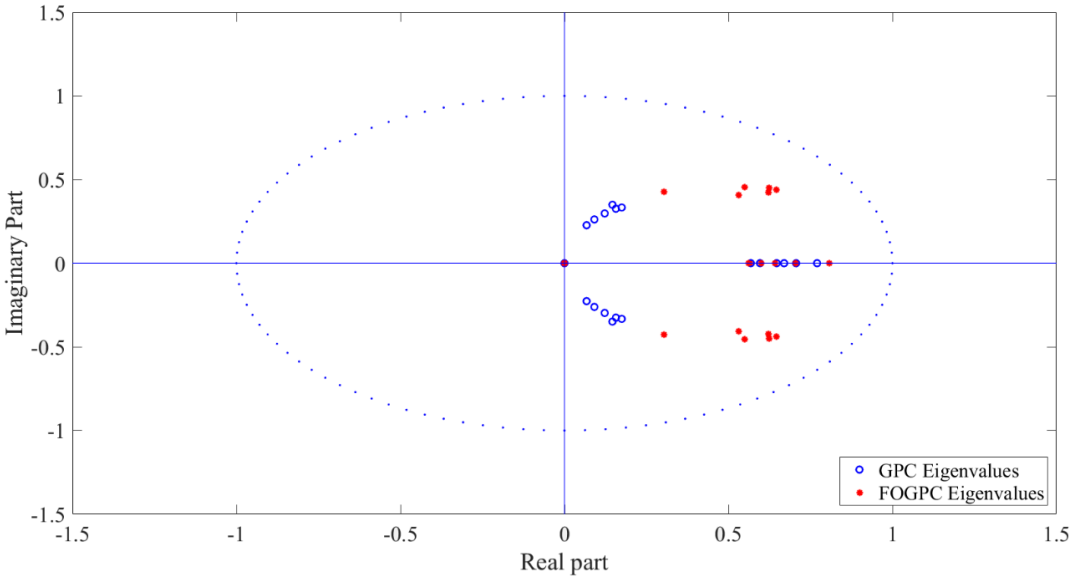


Figure 5.19: Poles locations comparison between FGPC and GPC with N_2 varying from 6 to 11

As Figure 5.17 above indicated, FGPC's poles are spreading more on the positive real part. In addition, FGPC has faster responses associated with these poles than GPC, in terms of reaching the desired set point.

5.5.2 Observing the effect of the input forecasting horizon (N_u)

We have studied the effect of the input forecast horizon N_u on the pole locations of both GPC and FGPC by assigning N_u to 2 initially, and to then increasing it by a fixed value of 1 for 10 iterations, while the other design parameters remain constant as follows:

- Input forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 50$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.05$
- The fractional-order coefficient $\beta = 2.1$

By definition, we can't assign N_u to be greater than N_2 and it can't be assigned to zero as well. Figure 5.19 shows the responses of both GPC and FGPC to the varying N_u and Figure 5.20 shows the simulated result of the poles' locations for both GPC and FGPC.

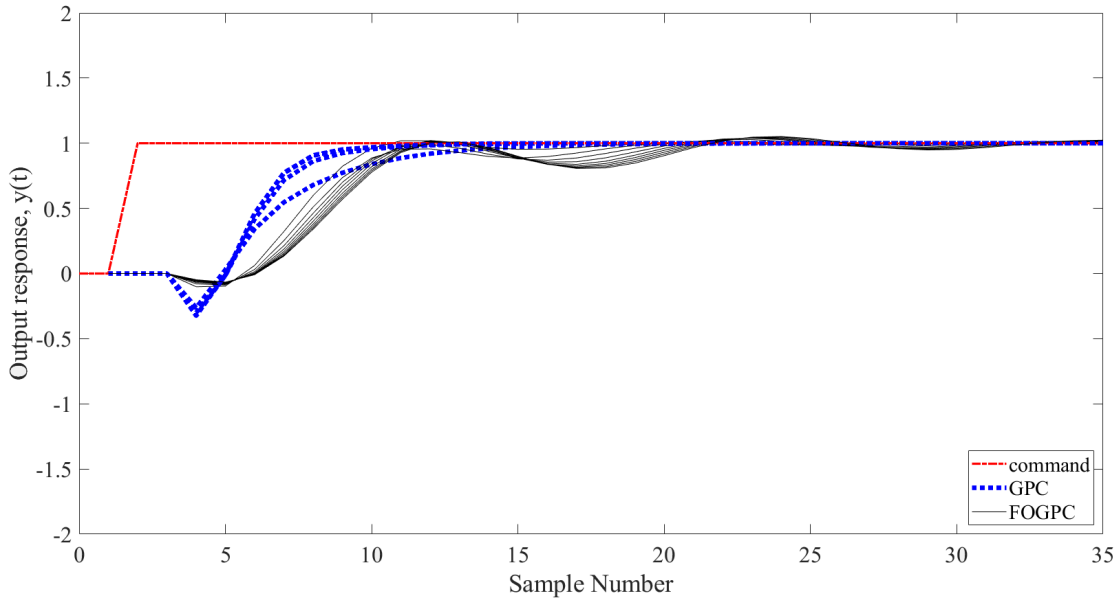


Figure 5.20: FGPC and GPC responses with N_u varying from 2 to 10 by a fixed value of 1

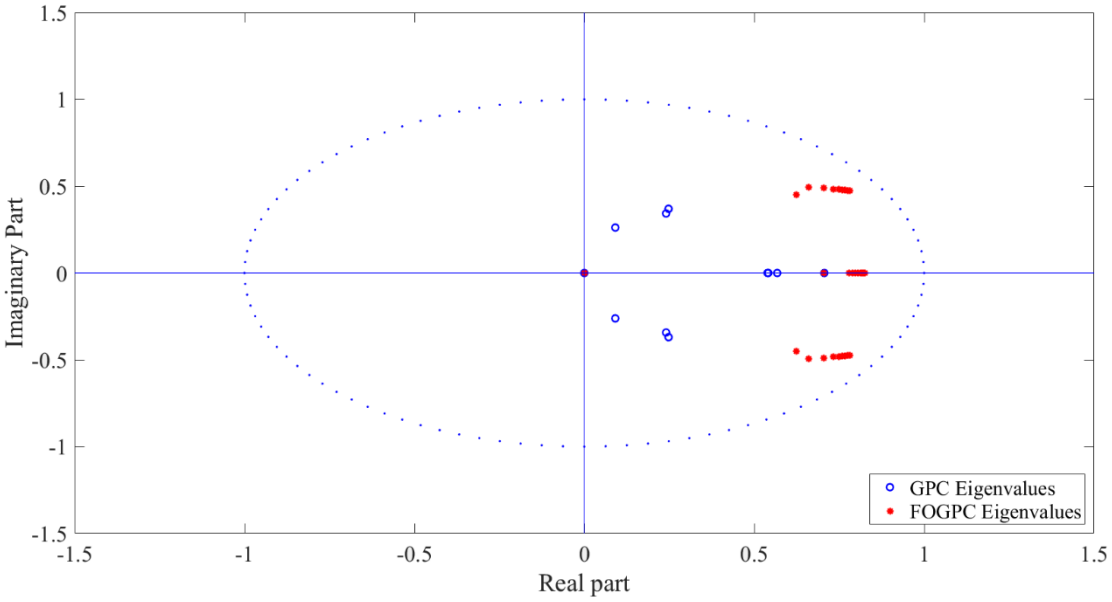


Figure 5.21: Poles locations comparison between FGPC and GPC with N_u varying from 2 to 10

We observed from the above figures that FGPC's poles are almost on the edge of the unity circle, which makes the controller marginally stable. This has been reflected on the FGPC's responses associated with those poles.

5.5.3 Observing the effect of the fractional-order weighting (α)

In this sub-section, we have studied the effect of α on the model response and its corresponding pole locations. To do so, we have assigned an initial value for $\alpha = 0$ and increased it by a fixed value of 0.01 for 100 iterations, whereas the other parameters were assigned asfollows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 50$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\beta = 2.1$

Figure 5.21 illustrates the response of both GPC and FGPC while Figure 5.22 illustrates the poles' locations corresponding to the responses.

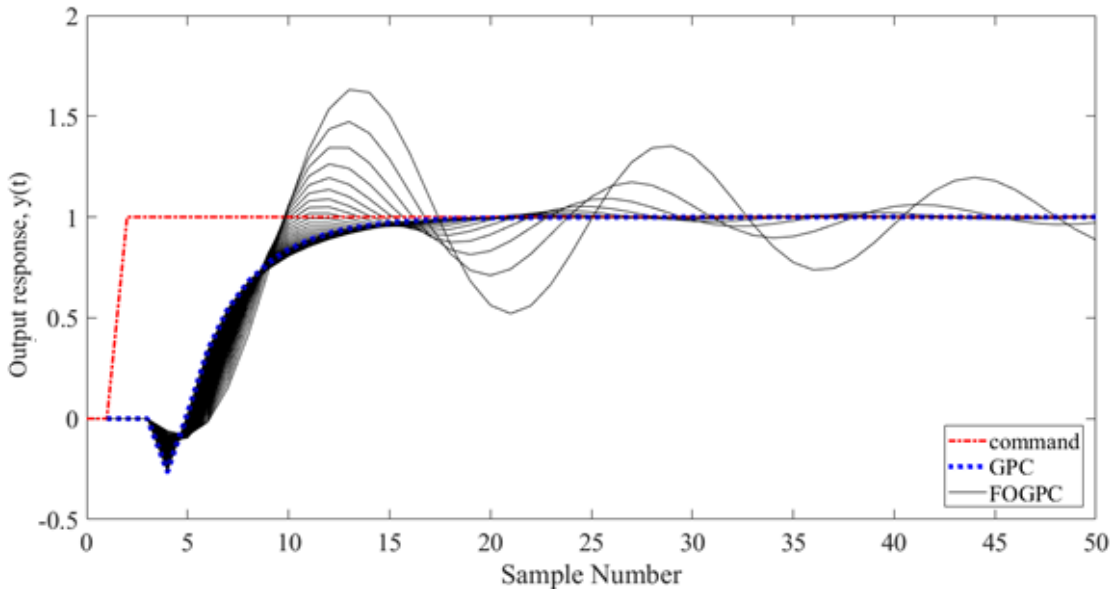


Figure 5.22: FGPC and GPC responses to the model with α varying from 0.5 to 10.5 with a fixed value of 0.1

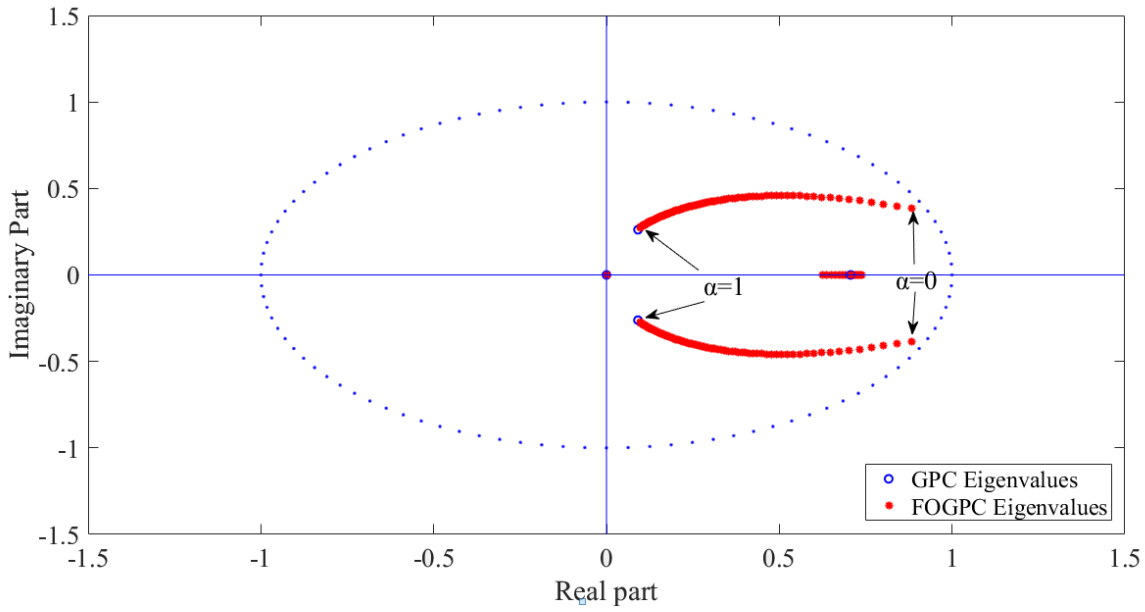


Figure 5.23: Pole locations for both GPC and FGPC with α varying from 0.5 to 10.5 with a fixed value of 0.1

When we assigned α to 0, we found a pole near the border of the unit circle and as α increased, the location of the poles shifted toward the origin in trajectory path. We have observed that when $\alpha = 1$, FGPC's poles have (more or less) the same locations of GPC poles which reflected on the response of FGPC to the model to act as GPC response.

5.5.4 Observing the effect of the fractional-order weighting (β)

As we concluded in the previous sub-section, α has a direct effect on the pole location as well as the response to the model. This sub-section concerns the effect of β on the response to the model and the pole locations. We have assigned $\beta = 0.5$ and increased it by a fixed value of 0.1 for 100 iterations. The other design parameters were left constant as follows:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 80$ with a sampling time $T_s = 1$
- The fractional-order coefficient $\alpha = 0.05$

Figures 5.23 and 5.24 show the response of the model to the varying β and the poles' locations corresponding to that response, respectively.

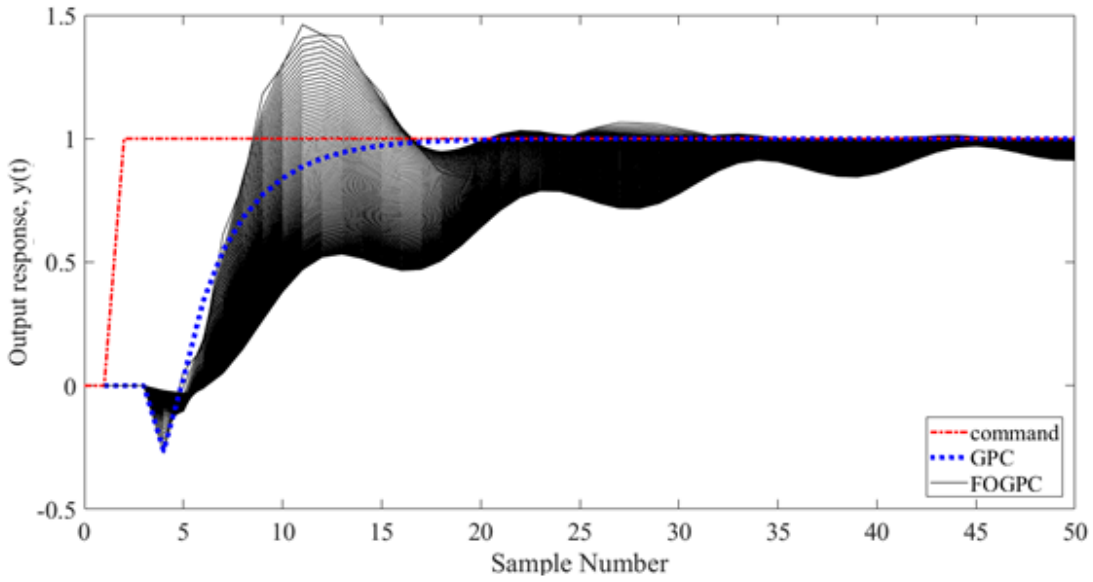


Figure 5.24: FGPC and GPC responses to the model with β varying from 0 to 10 with a fixed value of 0.1

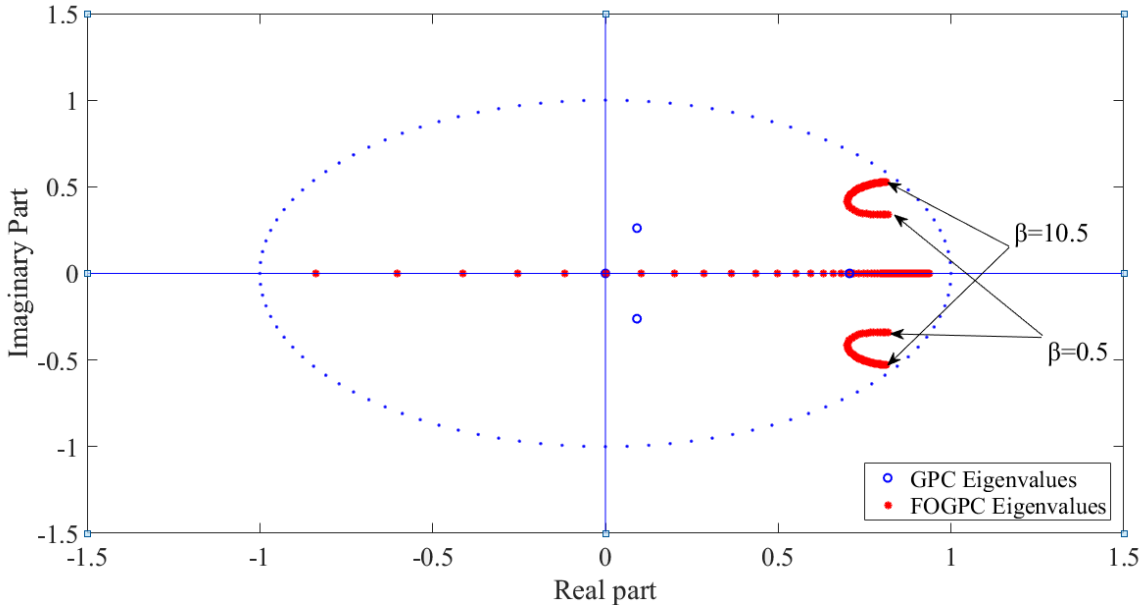


Figure 5.25: Poles locations for both GPC and FGPC with β varying from 0 to 10 with a fixed value of 0.1

Figures 5.23 and 5.24 summarised the effect of β on the response of the model along with the corresponding poles. As seen in Figure 5.24, the poles are tending to form half a circle as β increases. Clearly, from Figure 5.23, β affects the response of the plant as it was increasing.

5.6 Optimisation of α and β for a specific FGPC response

As observed in the case studies in the previous sections, the fractional-order coefficients α and β are playing a major role on the poles' locations which directly influence the response of FGPC controller to the system. Thus, we have designed a MATLAB script that can adjust the fractional-order coefficients through the use of a built-in function in the MATLAB platform known as "*fminsearch*". This function uses the Nelder-Mead method (please refer to Mathews and Fink (2004) for more details) to directly search for the minimum of an unconstrained multivariable function.

The designed script is used to fulfil a specific requirement by the user of the system. For instance, if the user demands the fastest response regardless anything else, then the script can be adjusted as per the user requirements and the function will find the optimised value of α and β for the system to achieve this goal. In this section, we will test the fastest response of FGPC by applying the script to several case studies. The case studies used are the same models that have been used throughout the thesis specifically in Chapter 4 and Chapter 5.

5.6.1 Case study 1 (Simple model)

Recalling equation (4.1)

$$G(z^{-1}) = \frac{1 - 2z^{-2}}{1 - 0.9z^{-1}}$$

Assuming the parameters set for controlling this model are the following:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- The simulation ended at the final time $T_f = 20$ with a sampling time $T_s = 1$

Those parameters have been chosen to be matched with the Chapter 4 example. The newly designed function which uses "*fminsearch*" in MATLAB has chosen the following values for α and β respectively:

- $\alpha = 0.7$
- $\beta = 0.75$

These values should correspond to the fastest response of FGPC regardless of anything else. Figure 5.26 below illustrates the findings

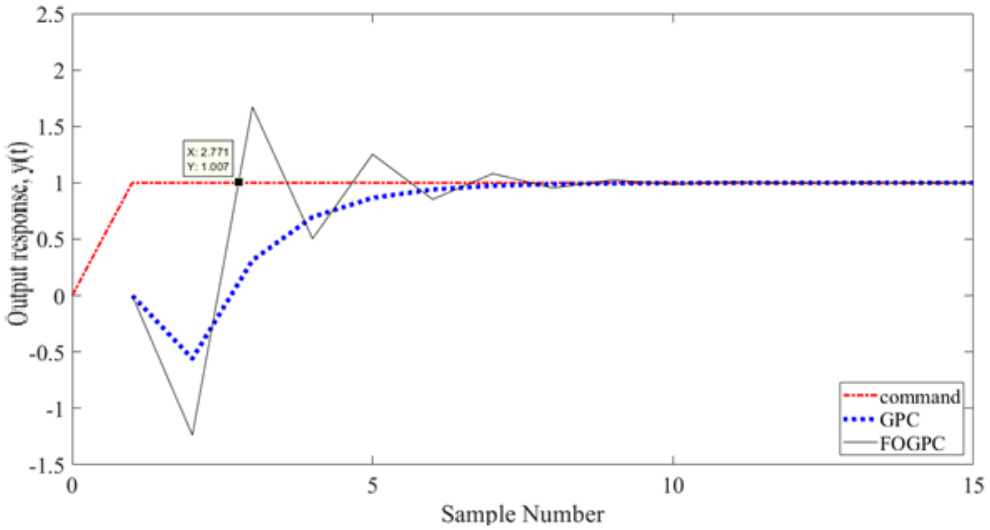


Figure 5.26: shows the fastest response of FGPC which has been achieved by changing α and β only.

We notice that the FGPC response here has achieved the desired point at 2.771 samples, whereas in comparison to the response we had from Chapter 4 which has been designed using trial and error for the values of α and β (0.77 and 1, respectively) the FGPC response has reached the desired point at 3 samples.

5.6.2 Case study 2 (Higher-order model)

Recalling equation (4.5)

$$(z^{-1}) = \frac{1-3z^{-1} + 5z^{-2}+0.3z^{-3}}{1 - 0.6z^{-1} - z^{-2} + 1.5z^{-3}}$$

The following parameters have been used to control this model:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- No model mismatch
- For GPC, constant values for (γ, λ), as $\gamma = 1$, $\lambda = 10^{-6}$

The fractional-order controller coefficients α and β have been calculated using the optimisation function designed.

- $\alpha = 5.67$
- $\beta = 0.26$

By testing the response of FGPC on this model using the provided values of α and β, we have found the fastest response of FGPC which is shown in Figure 5.27 below.

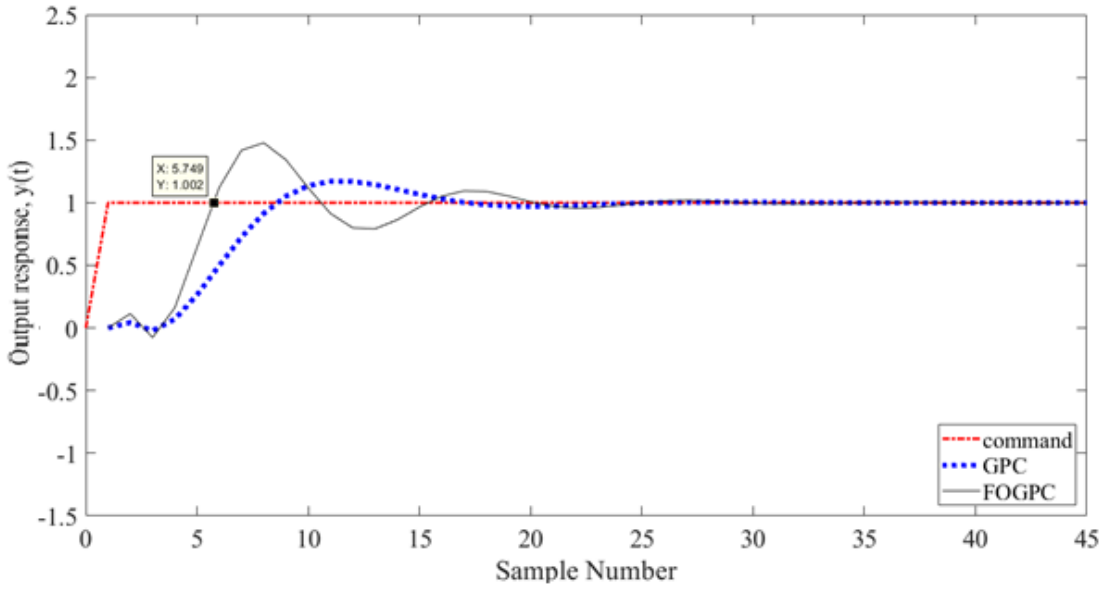


Figure 5.27: shows the FGPC fastest response using values of α and β determined by the optimisation function designed

As illustrated in the Figure, FGPC has responded to the desired input at 5.749 samples, whereas in Chapter 4 the response to the desired input was at 16.12 samples.

5.6.3 Case study 3 (Marginally stable)

In this case study, we will compare the response of GPC and FGPC on the same marginally stable plant as used in Chapter 4 (equation (4.6)).

$$G(z^{-1}) = \frac{-z^{-2} + 2z^{-3}}{1 - 1.7z^{-1} + z^{-2}}$$

The following parameters have been set:

- Input forecasting horizon $N_u = 2$
- Output forecasting horizon $N_1 = 1$

- Output forecasting horizon $N_2 = 10$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- For GPC, a constant values for (γ, λ), as $\gamma = 1, \lambda = 0$

The values of α and β have been determined by the optimisation function to be found as:

- $\alpha = 0.05$
- $\beta = 0.47$

Figure 5.28 below illustrates the response of FGPC at those points of α and β.

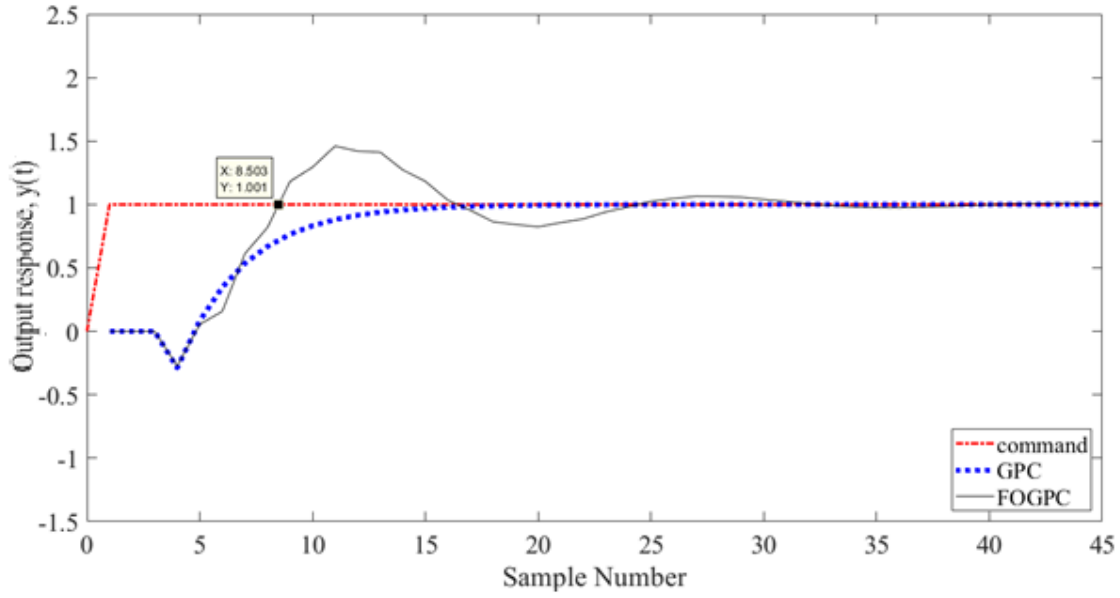


Figure 5.28: shows the response of FGPC at $\alpha = 0.08$ and $\beta = 0.54$

The response to the model was at 8.503 samples, whereas the response of FGPC for the same model in Chapter 4 was at 9 samples.

5.7 Discussion

In the previous sections of this chapter, we have studied the effect of various design parameters (N_2, N_u, α, β) on both conventional GPC and Fractional-order FGPC in terms of the pole locations. It is worth mentioning that the pole locations of GPC and FGPC of the same plant with the same parameters will never be the same as long as the fractional-order coefficient (α and β) values have not been assigned to force the FGPC response to behave like the GPC response. We have started with a simple case study (the same example used in Chapter 4) where all the design parameters have played a crucial role in terms of the response to the model or the corresponding pole locations. In this example, we have noticed that both FGPC parameters have affected the pole locations, with each parameter affecting particular poles in a particular way. This fact has been confirmed in the following case studies, as α and β have a major effect not only on the response to the model but on the pole locations as well (as would be expected, since the two are linked). In case 3 particularly, when we varied β, the responses look almost the same; however, the pole locations have changed. From these facts, we conclude that the FGPC poles are located differently to the GPC ones and that α and β are the key factors behind these pole locations.

In addition, we have introduced a new optimisation function that uses the built-in MATLAB function "*fminsearch*". This optimisation function is designed to determine the values of α and β for a specific design. For instance, we have used this function to determine the values of α and β to obtain the fastest possible response in FGPC by changing these parameters only (α and β) without changing the other design parameters (i.e N_1, N_2, N_u...etc).

It is important to stress that, as demonstrated in this chapter, the FGPC approach can yield closed-loop poles with negative real components. For completeness, these are included in the simulation results shown. For example, they are plotted on Figures 5.3, 5.7, 5.11, 5.13, 5.15,

and 5.25. The closed-loop responses associated with these poles are commented on in the text above. However, it should be reiterated here that, whilst the existence of FGPC poles on the negative real axis in the z-plane may be mathematically justified, there are practical ramifications of this and such locations should always be avoided in a practical realisation of the approach. For example, such closed-loop poles are typically associated with a "bang-bang" closed loop response, where the output jumps between extreme values from sample to sample. Clearly such a response is undesirable in practice and the associated FGPC design should not be implemented. In these cases, the FGPC settings (e.g., forecasting horizons) should be adjusted to obtain closed loop poles inside the unit circle on the right hand side of the complex z-plane, as also shown in the examples above.

5.8 Concluding Remarks

In this chapter, we have presented three case studies for which we have investigated the pole locations. All the studies have shown that the design parameters (i.e., N_2, N_u, β and α) have affected the model response and the corresponding pole locations for each response. The focus in this chapter has been on the fractional-order coefficients α and β. The key conclusion is that the fractional-order coefficients α and β play a major role in the pole locations, which subsequently affect the time response. Based on this fact, α and β may be used in pole placement or auto-tuning for FGPC. However, further research is required into these topics.

The next chapter will present a real-life application that compares the performance of FGPC and GPC. This comparison will be an extension of this chapter which compared the performance of GPC and FGPC in simulation basis.

Chapter 6 Laboratory Application

In the previous chapters, we have compared the simulated responses of GPC and FGPC; the next step for any control system is to be tested and evaluated for a real-life application, for which unpredicted, uncalculated and nonlinear effects may appear. On this basis, we have applied the fractional-order generalised predicted controller developed in previous chapters to a laboratory example. In this chapter, FGPC is evaluated using the forced ventilation chamber located in the Engineering building at Lancaster University. The aim is to experiment with the various control settings and observe the behaviour of FGPC in a real-life application, focusing on the effect of the fractional-order parameters λ and γ by adjusting the coefficients β and α, respectively. Section 6.1 introduces the laboratory equipment, section 6.2 describes the methodology, and section 6.3 presents the results. Finally, sections 6.4 and 6.5 present the discussion and conclusions, respectively.

6.1 Introduction to the ventilation chamber

It is well known that the ventilated airspace in agricultural buildings and the man-made environment are poorly mixed. As a result of imperfect mixing, stratification of environmental variables will occur, such as humidity, temperature, dust, gas, and air velocity, which all affect the surrounding micro-environment of the animals or plants. Thus, engineers and

mathematicians should bear in mind these factors when designing control systems. This provides come of the motivation for research using the forced ventilation test chamber (for further details refer to Taylor (2004)). In the following sub-sections, we will describe the hardware and software setup of the chamber.

6.1.1 Hardware setup

The chamber is a 2 m^2 by 2 m^2 cost-effective box with 320 Watts inlet and outlet fan which are powered by 0-240 volts AC supply (Taylor, 2004), as shown in Figure 6.1. In this thesis, the outlet and inlet fans are called the control fan and disturbance fan, respectively. The space between the inlet and outlet fans is filled with sensors to analyse the airflow of the chamber which is of interest in this experiment. It is worth mentioning that the hardware framework for the chamber has been recently updated (Tsitsimpelis and Taylor, 2015).

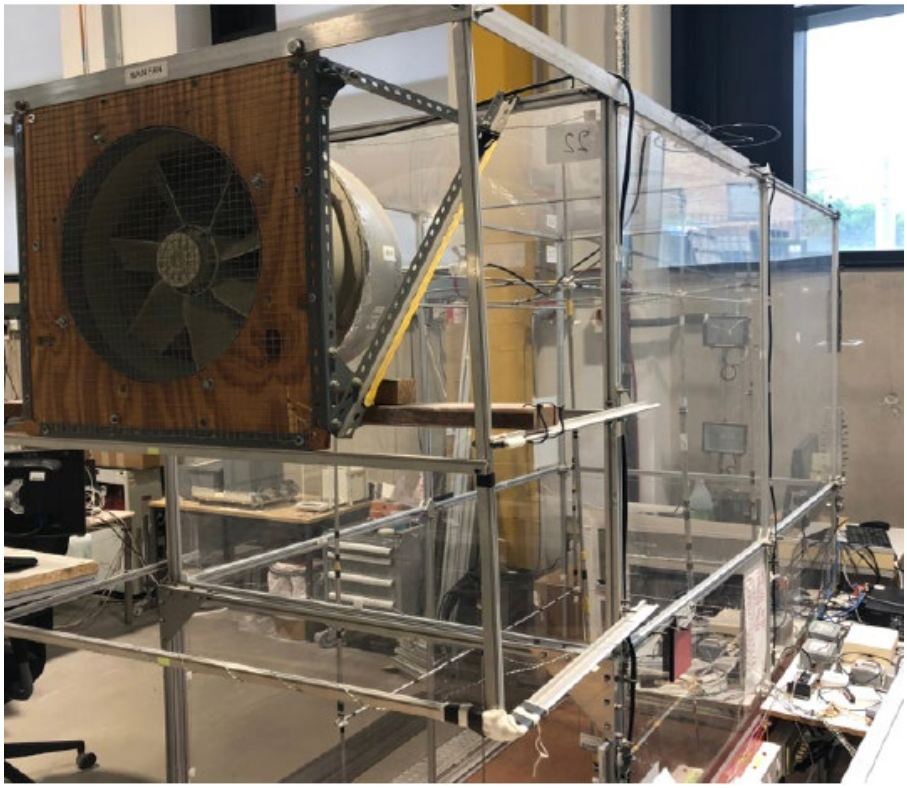


Figure 6.1: An overview of the ventilation chamber.

In Figure 6.1, the fan with wooden frame is the outlet (control) fan and the inlet (disturbance) fan is installed in the lower part of the chamber's box.

6.1.2 Software used

The ventilation chamber is equipped with a Personal Computer (PC) workstation that establishes a communication channel between the chamber and the user through a software package which is Laboratory Virtual Instrument Engineering Workbench or is known as LabVIEW. This software package is a graphical-based platform which means that it depends on building blocks rather than writing code as most of the engineering design software. However, LabVIEW allows direct communication with MATLAB software and translates its codes to interact with LabVIEW's blocks. Thus, we have used MATLAB to design and set up our controller's parameters and then transmitted it to LabVIEW which in turn sends it to the chamber and records the data from the chamber and stores it in a file that is readable by MATLAB and with a certain function, can be translated into a plot that represents the controller's response.

6.2 Methodology

As stated in the previous section, the author has designed a script to set a suitable platform for reading the data from the assigned sensors inside the confinement of the chamber (Figure 6.2) using a homogeneous blend of MATLAB code and LabVIEW building blocks. The script will record the readings from the sensors for data extraction. This step is done to extract the best models to form the most suitable transfer function. It is important to note that the results extracted from the chamber are non-linear due to the relationship between the voltage and the

airspeed, which raises a problematic issue with the controllers being used, as the controllers to be tested are linear controllers and this type of linearity in the system cannot be avoided. Thus, we have chosen the most linear region in the power curve plot that shows the relation between the applied voltage to the control fan and the steady-state ventilation rate as shown in Figure 6.2 below.

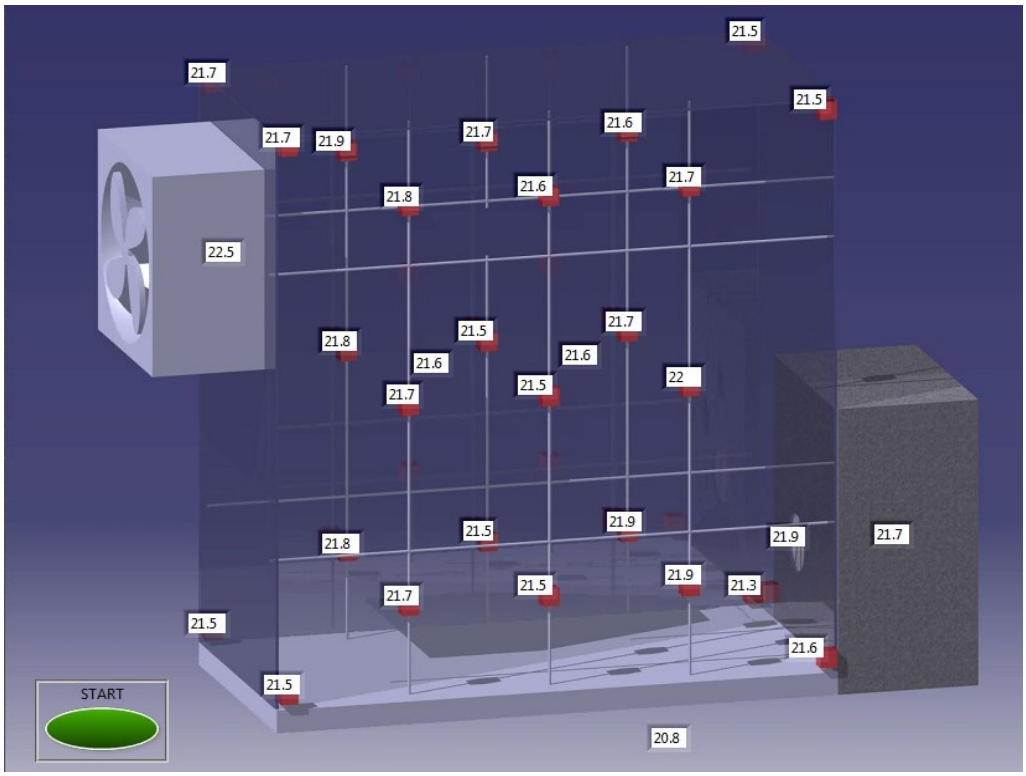


Figure 6.2: A schematic interactive diagram of the chamber that shows the temperature readings being collected from the thermal sensors

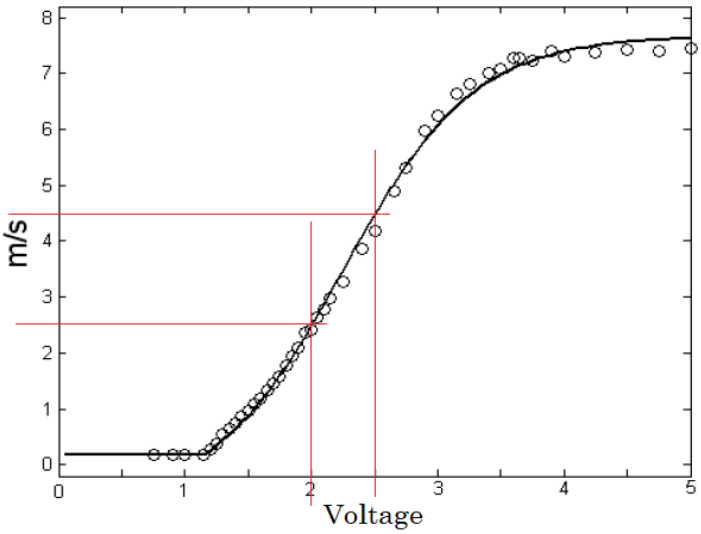


Figure 6.3: Power curve

Figure 6.3 shows the relationship between applied voltage to the control fan (Voltage) and the steady-state ventilation rate (m/s). The intersected lines indicate the selected area of operation.

As illustrated in Figure 6.3, the chosen region in this study was between the voltages (2, 2.5) which corresponds to ventilation rates approximately (2.6, 4.5) m/s. The sample rate used is one sample per second which result in Figure 6.4.

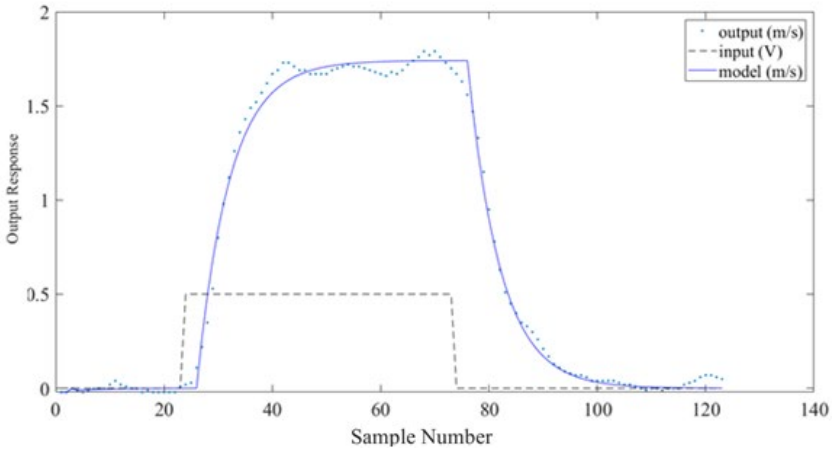


Figure 6.4: The voltage input to the ventilation chamber alongside the measured and the estimated response for the transfer function

To estimate the model to be used from the ventilation chamber, an algorithmic MATLAB toolbox (i.e., "Captain toolbox") has been used. The toolbox is using the Instrumental Variable approach. In particular, the toolbox utilises the recursive and en-block Refined Instrumental Variable (RIV) and Simplified Refined Instrumental Variable (SRIV) algorithms, as well as more conventional least-squares approaches.

Based on a user-specified model structure, the toolbox provides an estimation of transfer functions. However, for a given physical system, an appropriate model structure first needs to be identified, i.e., the most appropriate values for the time delay and the orders of the numerator and denominator polynomials. The two main statistical measures utilised here are the coefficient of determination (R^2), based on the response error, which is a simple measure of model fit (where unity indicates perfect fit) and the more sophisticated Young Identification Criterion (YIC) (Young, 2011), which provides a combined measure of fit and parametric efficiency with large negative values indicating a model which explains the output data well, without over-parameterisation.

Using the "Captain" toolbox (Taylor *et al*., 2018), we have obtained the best 20 models from the estimated transfer functions, in terms of the denominator "m", numerator "n", time delay, YIC and R^2.

Table 6.1: 20 Best matching models

m	n	Time delay	YIC	R^2
1	1	1	-8.447	0.605598
1	1	2	-8.430	0.607550
1	1	0	-8.428	0.601088
1	1	3	-8.367	0.607219
2	1	0	-7.535	0.73171
2	1	1	-7.127	0.702681
2	1	2	-6.374	0.657145
2	1	3	-5.548	0.606432
2	2	2	-5.527	0.776114
2	2	3	-5.467	0.720439
2	3	0	-4.882	0.722538
2	2	1	-4.277	0.762398
2	3	1	-4.103	0.660324
2	2	0	-3.514	0.729221
2	3	2	-3.268	0.577366
1	3	3	-2.368	0.608509
1	3	2	-2.334	0.613291
1	3	1	-2.208	0.617896
1	3	0	-2.061	0.620596
1	2	0	-1.757	0.613720

We have found that [1,1,3] (i.e., [m,n,time delay]) is a good candidate for developing control systems. The following transfer function is to be used in the design of GPC and FGPC for the sake of comparison between the two responses in an attempt to validate the hypothesis stating that FGPC has better controllability in terms of various specifications such as robustness, rising, and settling time, etc.

$$G(z^{-1}) = \frac{0.5325z^{-1}}{1 - 0.847z^{-3}} \tag{6.1}$$

Now, using the equations and MATLAB toolbox discussed earlier in Chapter 3, the author designed a GPC controller with the following parameters to be tested in the ventilation chamber using this transfer function in 6.1, keeping in mind that those parameters have been chosen based on trial and error within the chamber:

- Input forecasting horizon $N_u = 5$
- Output forecasting horizon $N_1 = 1$
- Output forecasting horizon $N_2 = 20$
- Pre-filter (noise polynomial) $T(z^{-1}) = 1$
- For GPC, a constant values for (γ, λ), as $\gamma = 1$, $\lambda = 10$

Similar to the previous chapter, the same parameters will be used for designing FGPC except for (γ, λ) values, as their values will be derived depending on the values of (β, α), respectively. Figure 6.4 shows screenshots of one of the examples used in the experiment to observe the responses of the controllers.

Looking at Figure 6.5, it can be seen that the control platform is built using LabVIEW blocks which is connected to the ventilation fan. For instance; “Main fan outlet”, “Main fan inlet”, and a function that calls the MATLAB code from the simulation. These blocks represent the functions used in FGPC/GPC (Figure 3.5). This indicates that LabVIEW programming is very straightforward and user-friendly.

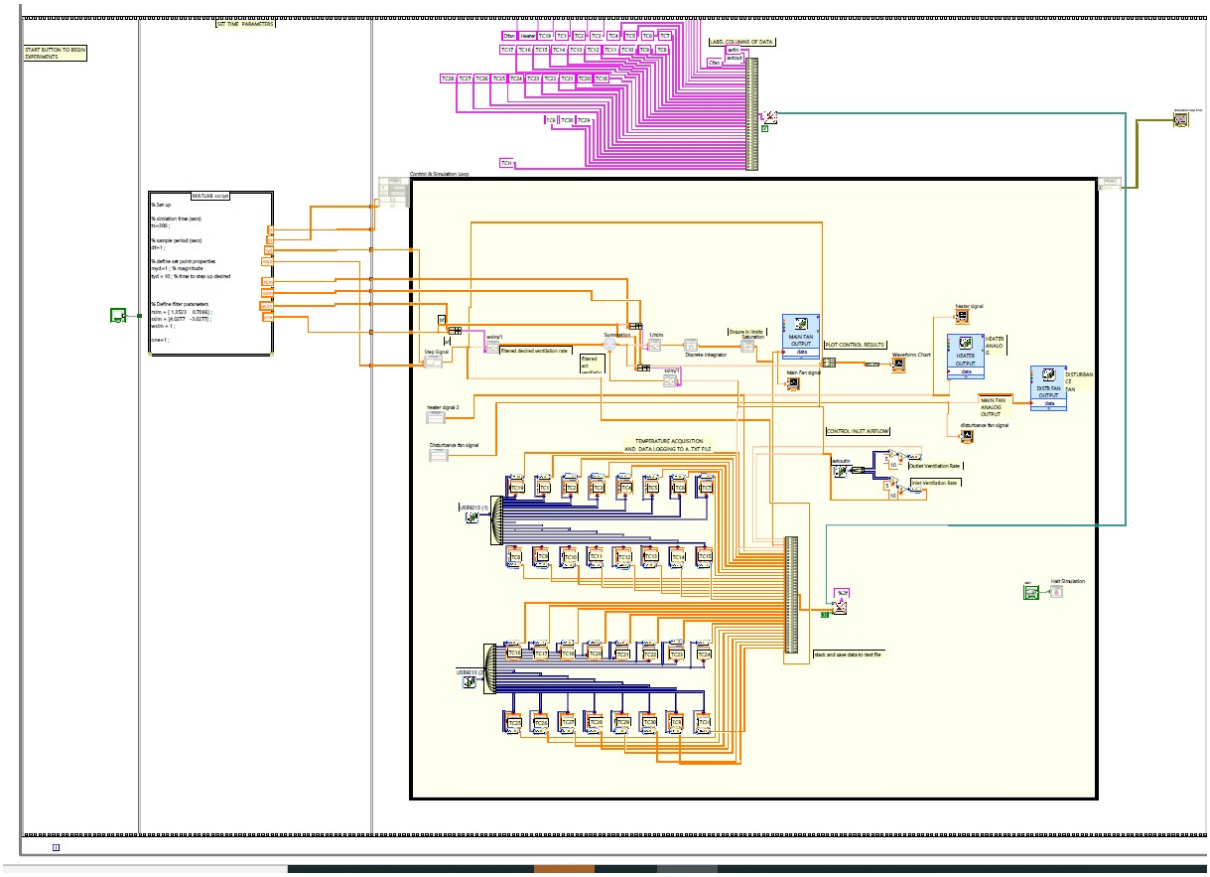


Figure 6.5: Example of the LabVIEW block diagram interface

The mechanism of testing the controllers will be based on observing the responses of the controller without any disturbance by switching off the inlet (disturbance) fan. The effect of α and β will be witnessed by manipulating their values and observing the effects on the FGPC response, then comparing it to the GPC response.

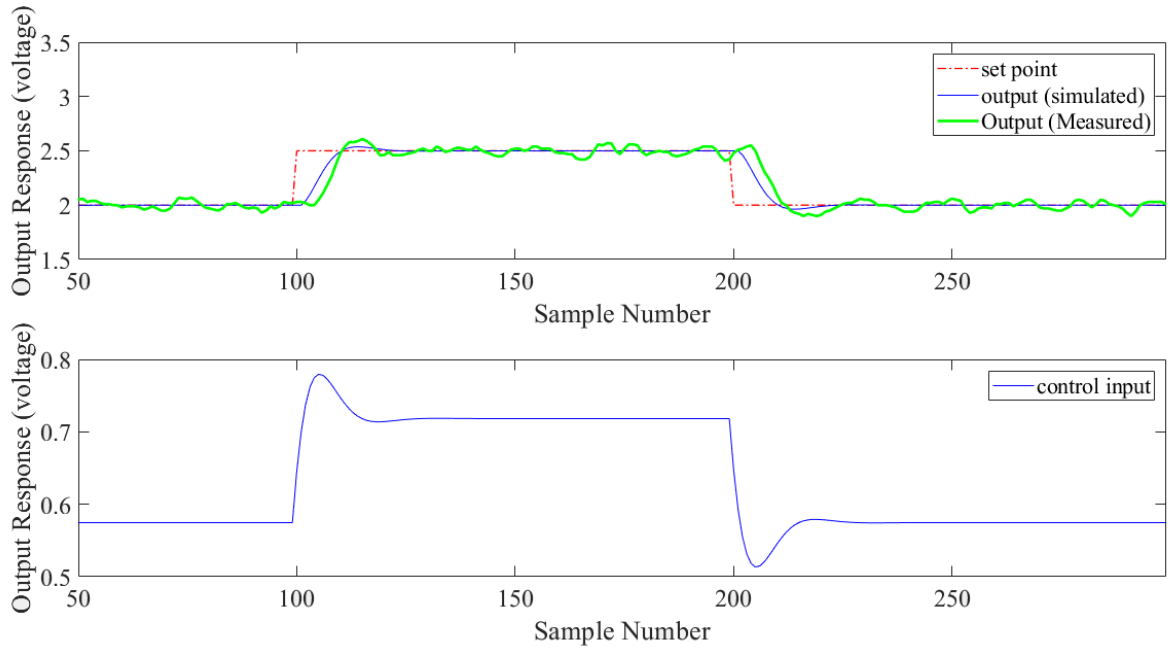
After testing both controllers in the chamber, the author will extract the data collected and plot a graph using MATLAB scripts: these call the data files recorded which will represent the response of the controller on the ventilation chamber.

6.3 Results

In this section, we will illustrate the response of each controller (i.e., GPC and FGPC) in separate sub-sections and show the effect of changing α and β on the response of FGPC.

6.3.1 GPC vs. FGPC response

Starting with the response of GPC using the parameters given in the previous section, GPC measured and simulated responses are illustrated in Figure 6.6.



6.6: GPC Simulated and Measured responses from the model extracted from the ventilation chamber, along with the simulated control input.

As seen in Figure 6.6, the real-life application has a different response than the simulated response. Indeed, the non-linearity of the chamber is the essential reason for the difference in

responses. Furthermore, in the simulated response, everything is ideal and there are no environmental or external factors influencing the results.

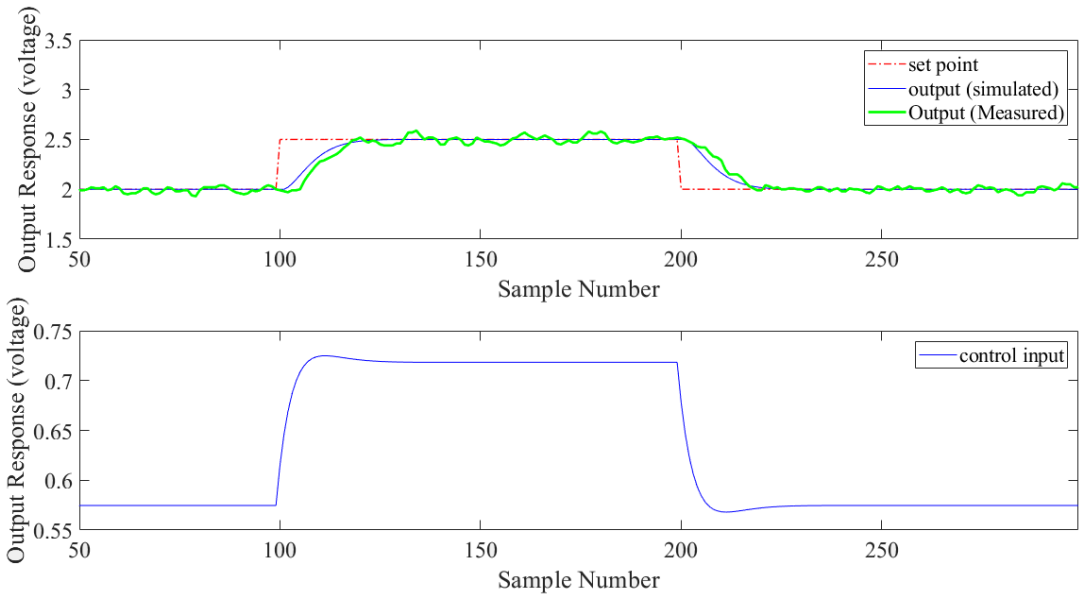
To see the response of FGPC, we need to choose a value for each of the two extra design parameters in Fractional-order GPC (i.e. α and β). With the manipulation of the fractional-order weighting coefficients (α, β), FGPC had a shorter settling time and smaller overshoot response. To investigate the effect of each one of these parameters, the author has followed the same principles as in Chapter 4, which is keeping one coefficient constant and changing the other just to observe the effect of it on the FGPC response. The following two sub-sections will be a chain of cases in which the author investigates the effect of each coefficient separately.

As concluded in the previous chapter, changing the fractional-order weighting coefficients will affect the response of FGPC. The response will be affected depending on the model; in some cases, changing α will have a huge impact on the response and the design engineer can easily manipulate it to achieve the required design specifications (i.e. fast response, smaller overshoot etc.). The same rule applies to the other coefficient β. By contrast, in some other models, the effect of one of these coefficients (or both of them) is limited and the designer will have to look at modifying other parameters. We need to study the effect of those parameters (α and β) on the response and choose the most suitable values i.e., to yield the best response we can get while (in this illustrative case) keeping the forecasting horizon parameters constant (i.e $N_1 = 1, N_2 = 20, N_u = 5$).

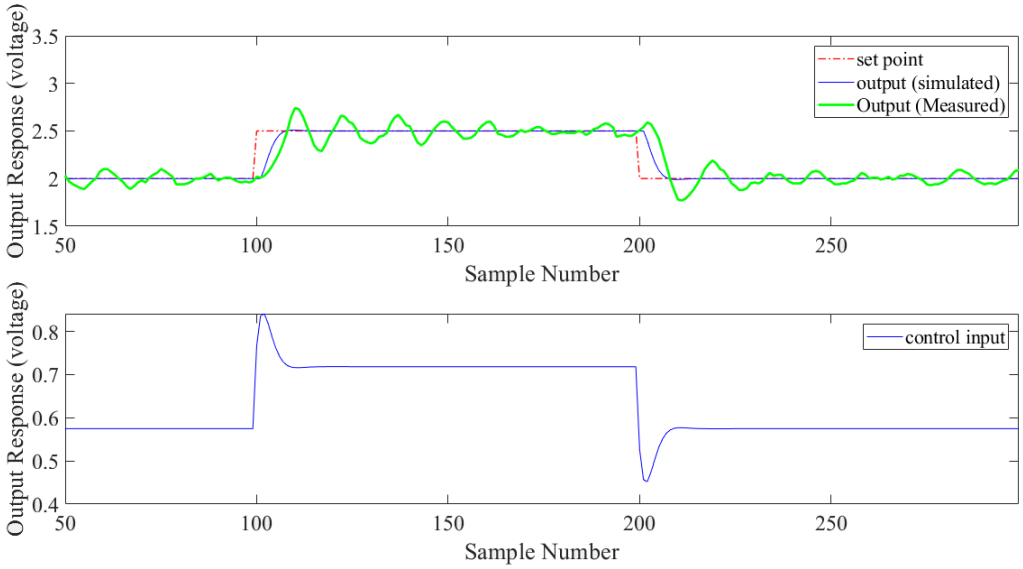
6.3.2 α affects on FGPC response

Following the same approach used in the previous chapters to investigate the effect of α on the FGPC response, we will keep all the parameters constant including β while changing the

value of α. The values of α are chosen to start at 0.1 and increase with a fixed value of, whereas the value of β is set to 1.5 and remains constant. Figure 6.8 and Figure 6.9 illustrates the response of FGPC when α is set to 0.1 and 0.7 respectfully.

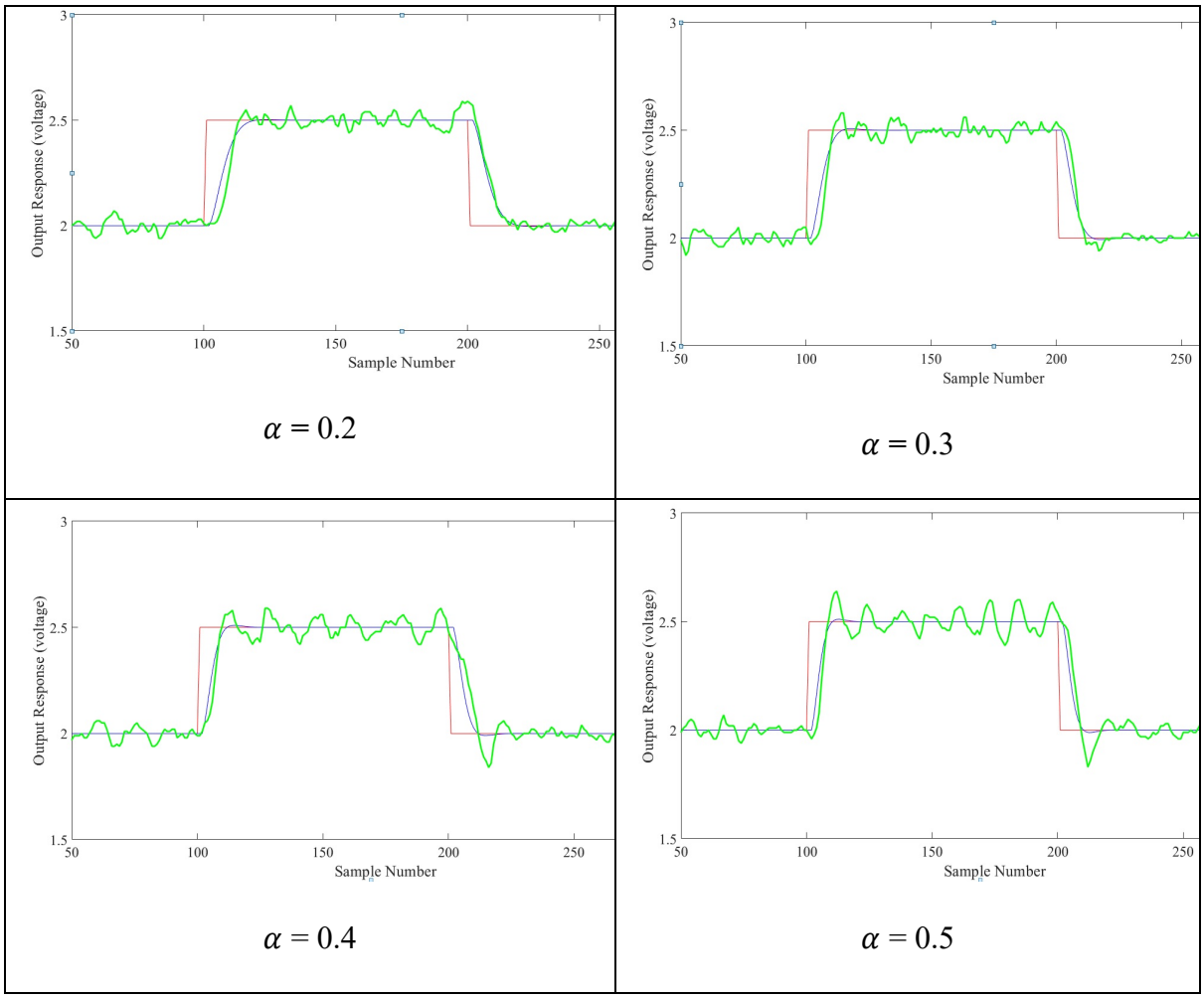


6.7: FGCP response for both simulated and measured from the ventilation chamber when α =0.1 and β = 1.5



6.8: FGCP response for both simulated and measured from the ventilation chamber when α =0.7 and β = 1.5

To shed more light on the effect of α on the FGPC response, we have observed the response for more values of α which are shown in Figure 6.9. Bear in mind that these values have been arbitrarily chosen just for observation and studying the behaviour of FGPC.



6.9: A comparison of FGBC responses with different α values while keeping β value at 1.5

According to the comparison we have made for the different responses, we can see that, with the value of α changing, the response of FGPC changes. Thus, we need to see the effect of β on the response before we can optimise their values. The next sub-section will study the effect of β on the behaviour of the FGPC controller by trying different values of β while the other

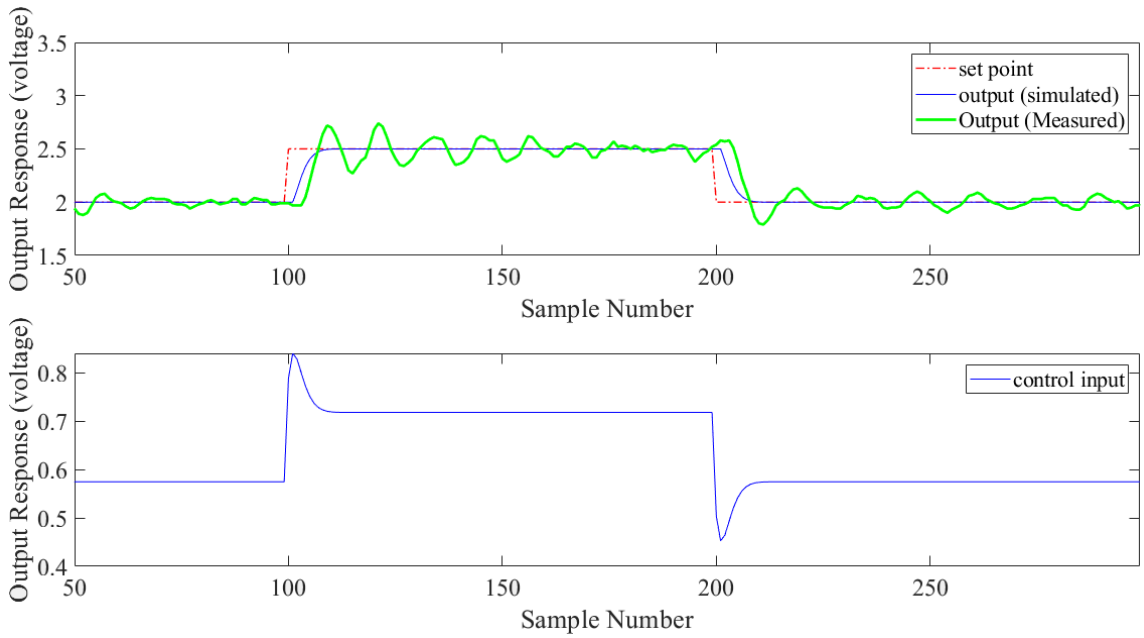
design parameters (including α) remain constant. Table 6.1 below summarises the comparison between different values of α.

Table 6.2: Comparison between different α

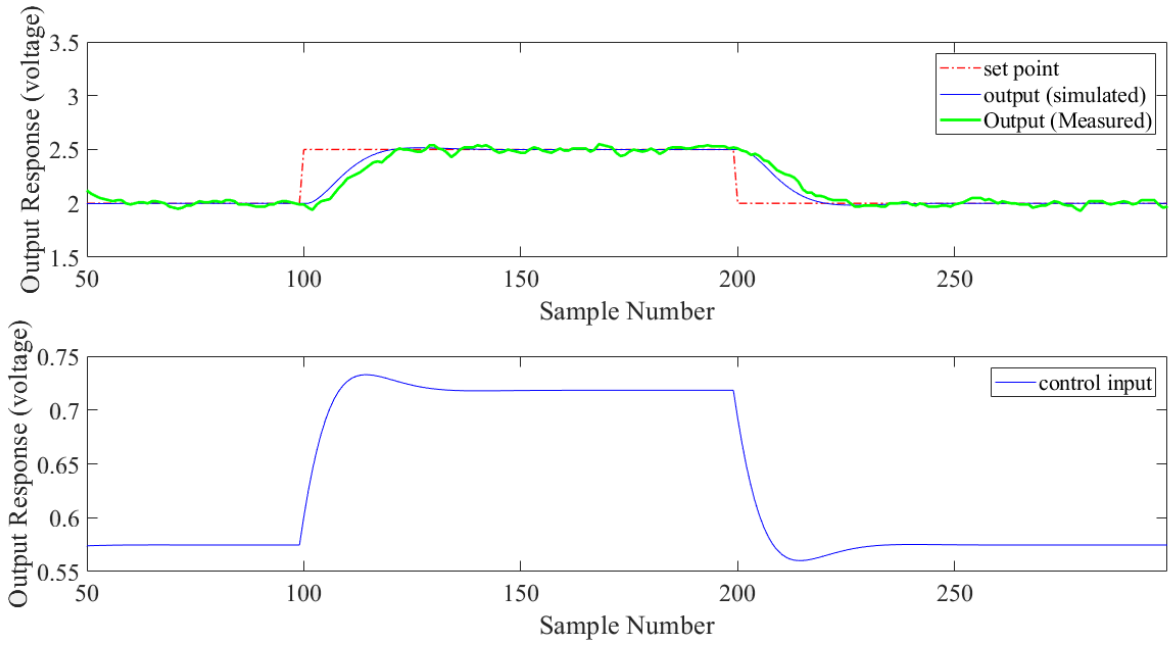
	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.7$
Response time [seconds]	18	14	11	10	8	8
Overshoot [% of step up]	0.8%	2%	3.2%	3.2%	5.6%	9.6%
Rise time [seconds]	5	4	4	2	4	1

6.3.3 β affects on FGPC response

Similar to the previous sub-section, we will keep all the design parameters constant apart from β which will be changing to study its effect on FGPC response. In this case, α has been set to 0.5 and all the horizons have been remaining the same. The following two figures illustrate two different values of β (i.e., 1.2 and 3.2, respectively) along with their input signals.

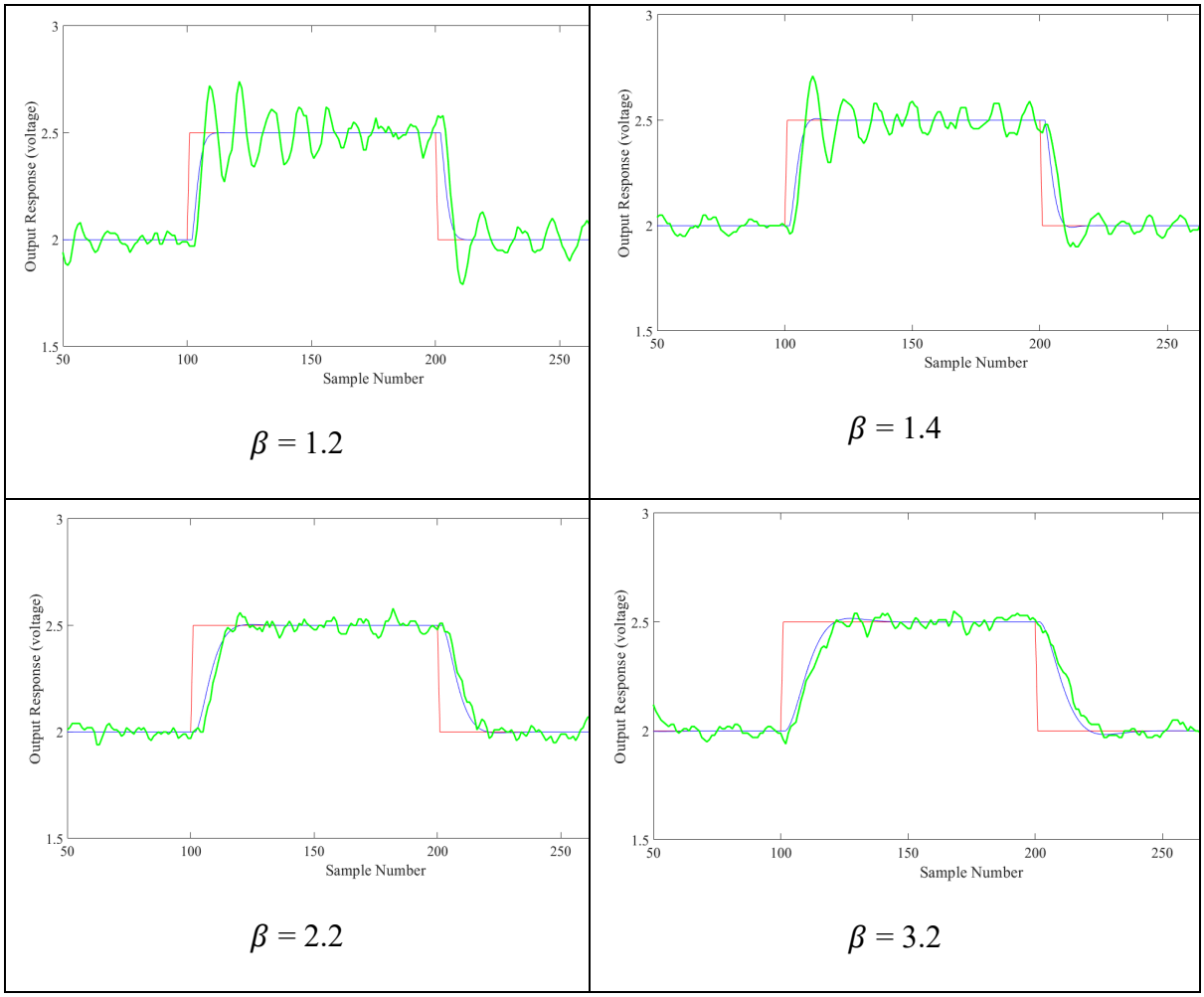


6.10: FGCP response for both simulated and measured from the ventilation chamber when α =0.5 and β = 1.2



6.11: FGCP response for both simulated and measured from the ventilation chamber when α =0.5 and β = 3.2

To shed more light on the effect of β on FGPC response, we have observed the response for more values of β which are shown in the combined Figure below. Bear in mind that those values have been chosen arbitrarily just for observation and studying the behaviour of FGPC concerning the changes in β values.



6.12: A comparison of FGBC responses with different β values while keeping α value at 0.5

Based on the responses above, we have observed that higher values of β will yield more stable responses. However, we need to be careful when choosing the values to compromise between

β and α to obtain the best response of FGPC compared to GPC, as shown in Figure 6.6. Table 6.2 below summarises the comparisons between different values of β.

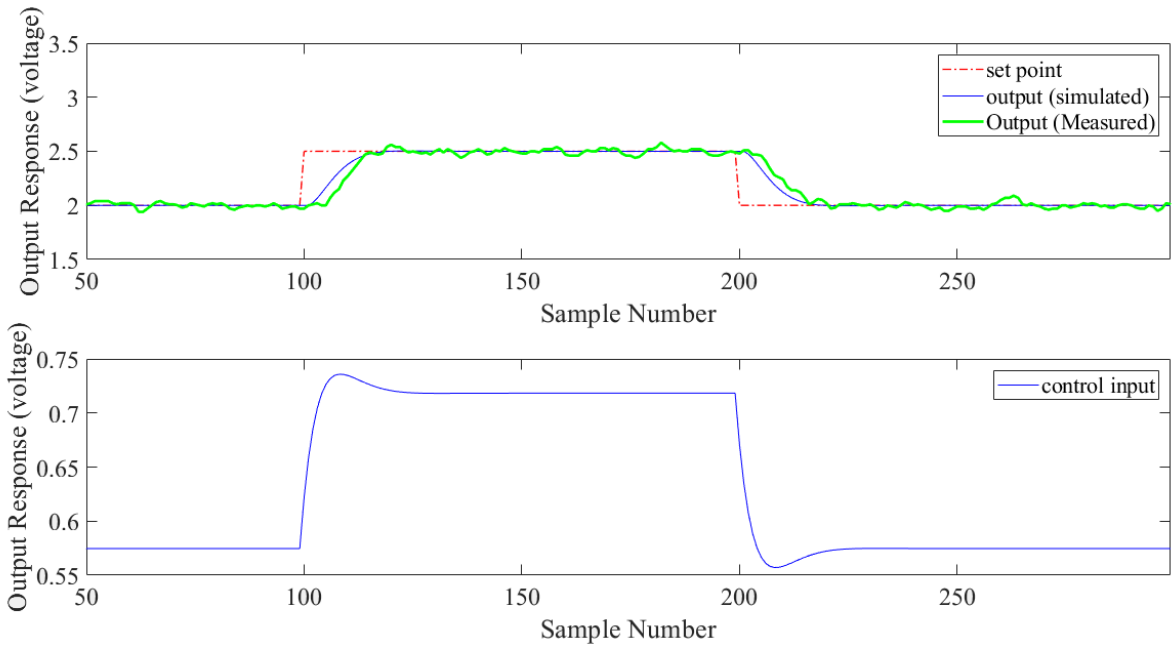
Table 6.3: Comparison between different β

	$\beta = 1$	$\beta = 1.2$	$\beta = 1.3$	$\beta = 1.4$	$\beta = 2.2$	$\beta = 3.2$
Response time [seconds]	8	7	8	9	12	21
Overshoot [% of step up]	9.1%	8.8%	8.8%	9%	4.4%	0.1%
Rise time [seconds]	4	4	4	4	3	4

The next sub-section will show the optimised values of α and β to obtain a faster and better-damped response of FGPC based on a trial and error method.

6.3.4 FGPC response with optimised α and β

In the previous sub-sections, we have observed the effect of α and β on the FGPC response and based on that observation we will choose optimised values for both α and β. After several trials, Figure 6.7 shows the response of FGPC to the same model and under the same horizon parameters ($N_1 = 1, N_2 = 20, N_u = 5$), and with α =0.14 and β =2.32. The values of α and β have been set based on studying the behaviour of FGPC concerning α and β changing. The author has chosen the most suitable response (less overshoot and more stable) on that basis.



6.13: The Simulated and Measured responses of FGPC using the same horizon parameters of GPC with manipulation of the weighting values (α, β); (0.14,2.32)

We conclude that FGPC has a much closer response to the simulated version of the model than the GPC response. In addition, FGPC has fewer oscillations and more smoothly follows the setpoint change than the GPC response.

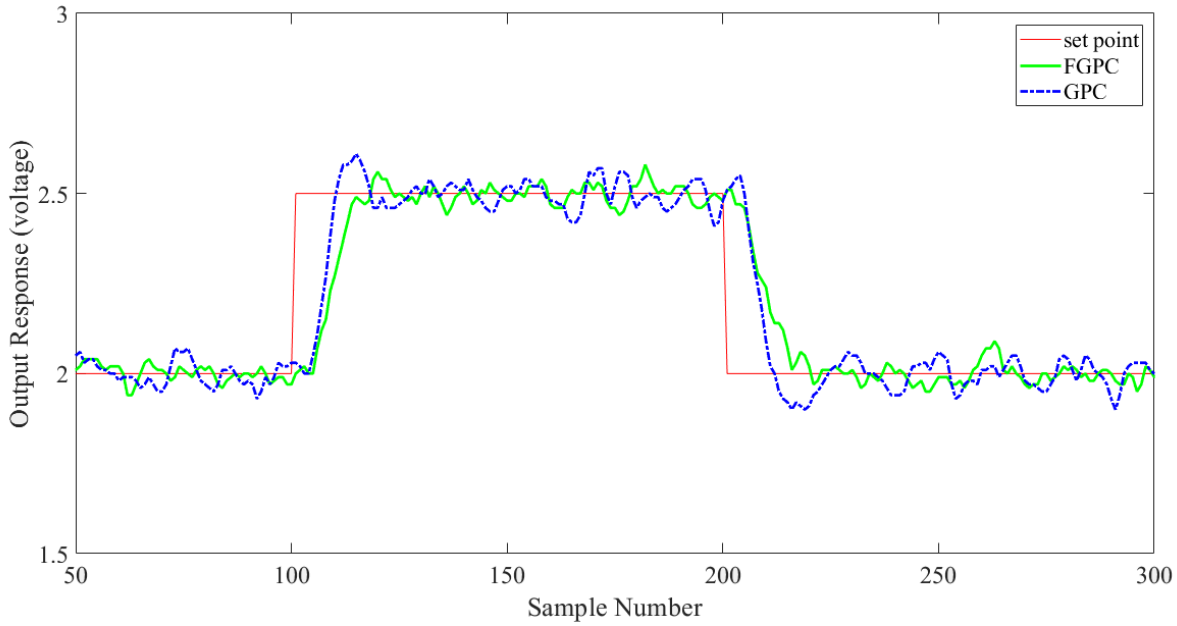
6.4 Discussion

As it has been mentioned on numerous occasions throughout the thesis, fractional-order GPC potentially is a generalised form of the conventional GPC. This section will illustrate how FGPC has extra degrees of freedom for tuning than GPC by establishing a comparison between FGPC and GPC based on their responses to the ventilation chamber.

Before starting the comparison between the two controllers, it is worth recalling that FGPC has two extra parameters (from the fractional-order part) than GPC. Those parameters are known as α and β. It has been demonstrated in section 6.3 that the effect of α and β on the behaviour and how the response of FGPC to the ventilation chamber is changed accordingly.

The extra parameters will add more complexity to the controller design; however, the extra effort is worthwhile for some control systems, i.e., to obtain better performance. The extra parameters will provide more 'space' for tuning the controller. In addition, FGPC can be created and designed exactly as GPC with the addition of the fractional-order part (for further details, please refer to Chapter 3 of the thesis).

Based on the results obtained in the previous section, we have considered some parameters to create a better-defined comparison between GPC and FGPC, according to their responses to the ventilation chamber. These parameters are defined in Figure 6.14, and Table 6.4 below.



6.14: illustrates response for both GPC and FGPC for comparison purposes.

Table 6.4: Comparison of the system responses for both GPC and FGPC

	GPC	FGPC
Response time [seconds]	11	14
Overshoot [% of step up]	4.4%	0%
Rise time [seconds]	4	4

FGPC has less oscillation response than GPC. Note that FGPC uses the same parameters as GPC (N_1, N_2 and N_u) in addition to the fractional-order parameters α and β to obtain this response. Hence, changing the values of α and β for the FGPC response will change the response accordingly to fit the design requirements.

Table 6.5: Comparison summary of GPC and FGPC

	GPC	FGPC
Advantages	• Relatively simple compared to FGPC. • Has many tuning techniques. • Well researched in the literature.	• Has 2 extra parameters which provide more flexibility for designing and tuning. • Can be tuned easily to fit any design specifications using the fractional-order parameters. • Can control fractional-order systems precisely.
Disadvantages	• Has fewer tuning parameters compared to FGPC. • Tuning has its limits and can't fit fractional-order plants without being approximated. • Fractional-order systems need to be approximated before they can be controlled	• Complex design compared to GPC. • Limited tuning techniques so far. • Has limited research in the literature.

To summarise this discussion, we have created a table of comparison that states the advantages and disadvantages of each controller (GPC and FGPC) based on the responses obtained in the present chapter.

6.5 Concluding Remarks

This chapter aimed to establish a comparison between GPC and FGPC for a practical example. The focus has been on the control of the ventilation rate in a forced ventilation test chamber. These results represent one of the first practical implementations of the FGPC approach for a laboratory example. In addition to the standard tuning terms for GPC, the FGPC approach involves α and β matrices that provide additional design freedom and potentially improved closed-loop performance. For the ventilation rate case study example under consideration in this chapter, the results demonstrate how to optimise (by trial and error experimentation) values for both α and β, to achieve a better performance of FGPC in terms of minimum overshoot and settling time.

FGPC has shown encouraging performance for this example and the comparison. However, choosing to control a system with fractional-order methods mainly depends on the output expectations for the system being controlled, as the design process includes some additional complications (compared to the GPC design) due to the fractional-order part.

In the next chapter, we will return to the simulation study, but with a new focus on the closed-loop eigenvalues, i.e., to investigate the role of the fractional-order parameters in determining the pole locations.

Chapter 7 Conclusions & Future Research

The main motivation behind the research in this thesis is the growing interest in the literature on fractional-order controllers. The research scope covered Fractional-order Generalised Predictive Control (FGPC), with a particular goal to implement this approach for a practical application. With such an example, the author hopes that researchers in both academia and industry will be motivated to investigate fractional-order methods. Section 7.1 of the present chapter briefly summarises the research outputs, while Section 7.2 provides a further discussion on the key results and suggestions for further research.

7.1 Summary

Chapter 1 stated the motivation of starting this research, the aims and objectives of the thesis that we are looking forward to achieving, alongside the articles arising based on some chapters of the thesis. While Chapter 2 presented the literature review, which covered most of the background on fractional-order controllers, including some of the key mathematical concepts (i.e., the various definitions of the fractional-order), with a specific focus on FPID. In addition, the chapter included some of the tuning techniques that have been used in the litreture to optimise the fractional-order controller, in order to get the maximum benefit of the fractional-order parameters. Chapter 3 revised the fundamentals of Model Predictive Control, especially the ubiquitous GPC algorithm, to establish the foundations needed to derive the FGPC

approach. Furthermore, the last sections of this chapter included a detailed explanation to demonstrate how to apply FGPC using MATLAB. Finally, the chapter is concluded with an illustrative example that compares GPC and FGPC output to provide a complete picture of applying FGPC using MATLAB. These preliminary chapters aimed to provide a full understanding of the concept of FOC in general, but with a focus on FGPC, which is the core of the research.

Hence, alluding to the research objectives stated in Chapter 1, the main novel contributions of the thesis followed in Chapters 4 through to Chapter 6. Chapter 4 investigated the effectiveness of the two extra parameters that exist because of the fractional-order GPC (i.e., α and β). This was achieved by applying both GPC and FGPC to several plants with various degrees of complexity. The simulation results were compared and the differences highlighted in the thesis. This chapter highlighted the behaviour of FGPC resulting from different values of the fractional order coefficient. We have noticed that even though when the horizons (i.e., N_1, N_2 and N_u) are kept constant in tuning both GPC and FGPC, FGPC could get a different response by manipulating these fractional coefficients. Hence extra degrees of freedom. Design recommendations for FGPC were outlined at the end of Chapter 4, together with a summary of the benefits of using FGPC over GPC. Subsequently, Chapter 5 brought another kind of comparison between GPC and FGPC, namely the eigenvalues of the closed-loop system (i.e., pole locations within the unit circle). The numerical results in this chapter showed that the new tuning coefficients α and β can indeed be used to potentially be on the left hand side of the complex z–plane (with the closed-loop eigenvalues) and still develop a stable response compared to the more constrained GPC approach. However, there are certain limits of manipulating α and β. Thus, we have used the trial and error approach to explore these limits and try to maintain a reasonable output for illustrative purposes though the research. This

conclusion led us to believe that, even with fractional-order controllers, there are some limitations. Chapter 6 implemented the ideas from Chapter 4 on a real-life application. The application chosen was an experimental ventilation chamber that is located within the engineering workshop at Lancaster University. The chapter focused on implementing both GPC and FGPC on the chamber, to validate the simulation results achieved in Chapter 4. We have observed that FGPC produced promising results in controlling the chamber compared to GPC in terms of fast response and lower overshoots. The approach used in choosing fractiona-order coefficients α and β was again trial and error.

7.2 Suggestions for Further Research

In general terms, model-based design (e.g., pole placement, GPC, FGPC) provides a quantitative method for determining the gains of the control system, based on the 'desired' performance of the closed-loop system. For the pole assignment method, performance relates to the poles of the closed-loop system; for GPC and FGPC it is the minimisation of the relevant predictive control cost function. As explained in this thesis, the cost function for FGPC includes two scalar hyper-parameters, α and β.

Of course, various different model-based design approaches can all produce the same outcome if the design criteria are 'correctly' set (Wilson *et al.* 2019). For example, a conventional pole assignment can be utilised to set the closed-loop poles corresponding to the poles that minimise the GPC or FGPC cost (once these are known). Hence, it can be argued that the choice of α and β is somewhat arbitrary. Indeed, Romero *et al.* (2010, 2012, 2013) who originally developed FGPC methods, provide no academic justification for why a fractional-order cost

function should be used nor any physically-based (engineering) reasons for how to select α and β, with trial and error via simulation being the implicit suggestion, as used in the present thesis.

Hence, within the context of the assumptions made above, FGPC provides a generalisation of the GPC cost function weights, which ultimately determines the numerical values of the control gains. As a result, the value of the FGPC approach appears dependent on whether the extra design flexibility provided by FGPC can be utilised to meet control objectives that are not achievable using standard GPC; and whether FGPC provides a straightforward to tune control algorithm – for example, that the use of α and β in this way provides a meaningful or convenient approach to solve practical control problems.

The first of these issues were considered by the simulation study in Chapters 4 and 6. The numerical results show that the new tuning coefficients α and β can indeed be used to provide more design flexibility than the conventional GPC approach. However, whether or not the increased range of eigenvalues, and hence potential time responses and other closed-loop characteristics, could facilitate a better control algorithm (e.g., for a given practical application and set of control objectives) requires further research.

The second issue was considered in Chapter 6, regarding the control of airflow in a forced ventilation chamber. In this case, a straightforward trial and error FGPC tuning yields a satisfactory response for this laboratory system. However, it would be true to say that the same applies to conventional GPC design, pole assignment, and various other model-based approaches. Hence, future research should consider the relative robustness and performance of the FGPC algorithm in comparison to GPC for additional laboratory scenarios and other examples. Indeed, there is a wide scope for research into other applications and the relationship

between the FGPC approach of the present thesis and other recent research into predictive control using fractional-order concepts. For example, Zou *et al.* (2016) apply fractional-order predictive functional control to industrial processes, whilst Shi *et al.* (2018) apply fuzzy generalised predictive control to a fractional-order nonlinear hydro-turbine regulating system. In addition, testing fractional-order models with fractional-order controllers and comparing its response to the conventional controllers is one of the areas that we are keen to explore.

Works Cited

Caponetto R., Dongola G.,Fortuna L. & Petráš I. (2010). Fractional Order Systems. In *Fractional Order Systems: Modeling and Control Applications* (p. 1 - 32). Singapore: World Scientific publishing Co. Pte. Ltd.

Abu-Ayyad.M & Dubay.R. (2010). Improving the Performance of Generalized Predictive Control for Nonlinear Processes. *Ind. Eng. Chem. Res*, 4809–4816.

Aleksei T., Eduard P. & Juri B. (2012). A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications. *the 31st chinese control conferance.* Hefei, China.

Astrom & Hagglund. (1995). *PID Controllers: Theory, Design and Tuning* (2nd ed.). New York: Research Triangle Park.

Astrom K. J.& Murray R.M. (2008). *Feedback Systems: An Introduction for Scientists and Engineers.* Princeton University Press.

Baleanu D. & Muslih S. I. (2008). Nonconservative systems within fractional generalized derivatives. *Journal of Vibration and Control, 14*(9-10), 1301–1311.

Barbosa R.S, Tenreiro J.A. & Ferreira I.M. (2003). A fractional calculus perspective of PID tuning. *the ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference.* Chicago, USA.

Bingul Z. & Karahan O. (2011). Tuning of Fractional PID Controllers Using PSO Algorithm for Robot Trajectory Control. *International Conference on Mechatronics.* Istanbul, Turkey: IEEE.

Bitmead R., Gevers M. & Wertz V. (1990). *Adaptive Optimal Control: The Thinking man's GPC.* New York: Prentice Hall.

Bittanti S., Laub A. J. & Willems J. C. (1991). *The Riccati Equation.* Berlin: Springer-Verlag.

Cafagna, D. (2007). Past and present—fractional calculus: a mathematical tool from the past for present engineers. *IEEE Industrial Electronics Magazine, 1*(2), 35-40.

Camacho & Bordones E. F. (2004). *Model Predictive Control.* Sevilla: Springer.

Camacho E.F. & Botdons C. (2007). Generalized Predictive control. In *Model Predictive control* (p. 48). London: Springer-Verlag.

Chen Y. Q. & Moore K. L. (2005). Relay Feedback Tuning of Robust PID Controllers With Iso-Damping Property. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B*, 23-31.

Chen Y. Q. & Xue D. (2002). A Comparative Introduction of Four Fractional Order Controllers. *the 4th World Congress on Intelligent Control and Automation.* Shanghai, China.

Chen Y. Q., Hu C. H.& Moore K. L. (2004). Relay Feedback Tuning of Robust PID Controllers. *42nd IEEE Conference on Decision and Control.* Maui, Hawaii, USA.

Chen Y. Q., Hu C. H., Vinagre B. M. & Monje C. A. (2004). Robust PIα Controller Tuning Rule with Iso-Damping Property. . *American Control Conferemce.* USA.

Chen Y., Dou H., Vinagre B., & Monje C. (2006). A robust tuning method for fractional order PI controllers. *in Proceedingsof the 2nd IFACWorkshop on Fractional Differentiation andIts Applications.* Porto, Portugal.

Chen Y., Petr I. & Xue D. (2009). Fractional Order Control - A Tutorial. St. Louis, MO, USA: American Control Conference.

Chen, Y. (2006). Ubiquitous Fractional Order Controls? Porto, Portugal: The Second IFAC Symposium on Fractional.

Chotai A., Young P.C. & Behzadi M.A. (1991). Self-adaptive design of a non-linear temperature control system. *special issue on self-tuning control. IEE Proc.*, 41-49.

Clarke D.W. & Mohtadi C. (1989). Properties of Generalized Predictive Control. *Automatica, 25*, 859 – 875.

Clarke D.W., Mohtadi C. & TUFFS P. S. (1987b). Generalized Predictive Control Part II. Extensions and Interpretations. *Automatica*, 149-160.

Clarke D.W., Mohtadi C. & TUFFS P.S. (1987a). Generalized Predictive Control-Part I. The Basic Algorithm. *Automatica*, 137-148.

Crisalle O.D., Seborg D.E. & Mellichamp D.A. (1989). Theoretical analysis of long-range predictive controllers. *ACC*, (p. 570-576.). Pittsburgh PA.

Cutler. C and Ramaker. D. (1980). Dynamic matrix control - a computer control algorithm. *Automatic Control Conference Proc.* San Francisco,CA.

De Keyser & Robin. (1992). *CONTROL SYSTEMS, ROBOTICS, AND AUTOMATION Vol. XI (Model Based Predictive Control for Linear Systems).* EOLSS.

Demircioğlu H. & Gawthrop P. J. (1991). Continuous–time Generalized Predictive Control (CGPC). *Automatica*, 55 – 74.

Demircioglu H.& Karasu E. (2000). Generalized predictive control. A practical application and comparison of discrete- and continuous-time versions. *IEEE Control Systems*, 36- 47.

Dormido S., Pisoni E. & Visioli A. (2012). INTERACTIVE TOOLS FOR DESIGNING FRACTIONAL-ORDER PID CONTROLLERS. *International Journal of Innovative Computing, Information and Control, 8*(7), 4579–4590.

Faieghi M.R.& Nemati A. (2011). On Fractional-Order PID Design. In P. T. Michalowski (A cura di), *Applications of MATLAB in Science and Engineering* (p. 273-292). InTech. Tratto il giorno 2015

Fei L., Wang J.,Zhang L., Ge Y.& Li K. (2013). Fractional-Order PID Control of Hydraulic Thrust System for Tunneling Boring Machine. In *Intelligent Robotics and Applications: 6th International Conference ICIRA 2013, Part 2* (p. 470–480). Busan, S.Korea: Springer-Verlag Berlin Heidelberg.

Franklin.G.F ,Powell.J.G & Emami-Naeini.A. (2008). *Feedback Control of Dynamic Systems (sixth edition).* San Francisco, USA: Pearson.

Gross, J. (2009). Binomial Coefficients. In *COMBINATORIAL MATHEMATICS.* Columbia: Columbia University.

Gutierrez R. E., Rosario J. M. & Machado J. T. (2010). Fractional Order Calculus: Basic Concepts and Engineering Applications. *Mathematical Problems in Engineering*, 19.

Hoffman .K & Kunze .R. (1971). *LINEAR ALGEBRA .* New Jersey: PRENTICE-HALL, INC.

Jin Y., Luo Y., Wang C. & Chen Y. Q. (2009). LabVIEW based experimental validation of fractional order motion controllers. *Control and Decision Conference.* China: IEEE.

K. Shi, B. Wang, and H. Chen. (2018). "Fuzzy generalised predictive control for a fractional-order nonlinear hydro-turbine regulating system. *IET Renewable Power Generation*, 1708–1713.

Krid.M, Benamar.F & Lenain.R. (2017). A new explicit dynamic path tracking controller using generalized predictive control. *International Journal of Control, Automation and Systems*, 303–314.

Kumar P., Narayan S., Raheja J. (2015). Optimal Design of Robust Fractional Order PID for the Flight Control System. *International Journal of Computer Applications, 128*(14).

Kumar S., Mishra & Chandra D. (2014). Stabilization and Tracking Control of Inverted Pendulum Using Fractional Controller. *Journal of Engineering, 2014*.

Ladaci S. & Bensafia Y. (2015). Indirect fractional order pole assignment based adaptive control. *Engineering Science and Technology, an International Journal*.

Lambert, E. P. (1987). *Process Control Applications Of Long Rang Prediction.* London: St. John's College, Oxford.

Lee M. & Morari J.H. (1997). Model predictive control: past, present and future. *PSE'97–ESCAPE–7 symposium.* Trondheim, Norway.

M.Vidyasagar. (2002). *Nonlinear systems analysis.* (2. Edition, A cura di) New Jersey: Prentice-Hall.

Maciejowski, J. (2002). *Predictive Control with constrains.* Hall, Harlow, UK.

Magin R. L. & Ovadia M. (2006). Modeling the cardiac tissue electrode interface using fractional calculus. *the 2nd IFAC Workshop on Fractional Differentiation and Its Applications.*

Maiti D., Biswas S., & Konar A. (2008). Design of a Fractional Order PID Controller Using, 2nd National Conference on Recent Trends in Information Systems.

Malek H., Luo Y. & Chen Y. (2013). Identification and tuning fractional order proportional integral controllers for time delayed systems with a fractional pole. *Mechatronics, 23*, 746–754.

Manabe, S. (1961). The non-integer integral and its application to control systems. *Japanese Institute of Electrical Engineers Journal, 6*((3-4)), 83–87.

Mandelbrot, B. (1982). The Fractal Geometry of Nature. San Francisco: CA: Freeman.

Masur, E. (1984). Optimal structural design under multiple eigenvalue constraints. *International Journal of Solids and Structures*, 211-231.

Mathews J.H & Fink.K.K . (2004). *Numerical Methods Using Matlab, 4th Edition.* New Jersey: Prentice-Hall Inc.

Matignon, D. (1998). Generalized Fractional Differential and Difference Equations: Stability Properties and Modelling Issues. *Mathematical Theory of Networks and Systems Symposium.* Padova.

Melício R., Mendes V. M. F. & Catalão J. P. S. . (2010). Fractional-order control and simulation of wind energy systems with PMSG/full-power converter topology. *Energy Conversion and Management, 51*(6), 320 - 325.

Merrikh-Bayat F., Mirebrahimi N. & Khalil M.i. (2015). Discrete-Time Fractional-Order PID Controller: Definition, Tuning, Digital Realization and Some Applications. *International Journal of Control, Automation, and Systems*, 81-90.

Miller K. S. & Ross B. (1993). *An Introduction to the Fractional Calculus and Fractional Differential Equations.* New York: John Wiley and Sons.

Monje C. A. , Vinagre B. M., ChenY. Q., Feliu V., Lanusse P. & Sabatier J. (2004). PROPOSALS FOR FRACTIONAL PIλDμ TUNING. *Fractional Differentiation and its applications.* Bordeaux.

Monje C. A., Calder´on A. J., Vinagre B.M., Chen Y. & Feliu V. (2004). On fractional PIλ controllers: some tuning rules for robustness to plant uncertainties. *Nonlinear Dynamics*, 369–381.

Monje C. A., Vinagre B. M., Feliu V & Chen Y. Q. (2008). Tuning and Auto-Tuning of Fractional Order Controllers for. *Control Engineering Practice, 16*(7), 798-812.

Monje C. A., Vinagre B. M., Chen Y. Q., Feliu V., Lanusse P.& Sabatier J. (2005). fractional derivatives and their applications. In *Systems analysis, implementation and simulation, systems identification and control.* Augsburg, Germany.

Monje C.A., Chen Y.Q., Vinagre B.M., Xue D. & Feliu V. (2010). *Fractional-order Systems and controls.* London: Springer.

Monje C.A., Chen Y.Q., Vinagre B.M., Xue D.& Feliu-Batlle. (2010). *Fractional-order Systems and Controls (Fundamentals and Applications).* london: Springer.

Moreau X. & Daou R. Z. A. (2014). Comparison between integer order and fractional order controller, 17th IEEE Mediterranean Electrotechnical Conference. Beirut, Lebanon,.

Moreau, DAOU R. A. & Xavier. (2014). Comparison between integer order and fractional order controllers. Beirut, Lebanon: 17th IEEE Mediterranean Electrotechnical Conference.

Neçaibia A., Ladaci S. & Mekhilef S. (2015). ABS Braking Control Enhancement Via Fractional Order Extremum Seeking Method. *Journal of Automation & Systems Engineering*, 23-36.

Oldham K. B.& Spanier J. (2006). *The Fractional Calculus. Theory and Applications of Differentiation and Integration of Arbitrary Order.* New York: Dover.

Ortigueira M. D., Machado J. A. T. & Da Costa J. S. (2005). Which differintegration? *IEE Proceedings:Vision, Image and Signal Processing, 152*(6), 846–850,.

Oustaloup A., Moreau X.& Nouillant M. (1996). The CRONE suspension. *Control Engineering Practice, 4*(8), 1101–1108.

Oustaloup, A. (1991). *La Commade CRONE: Commande Robuste d'Ordre Non Entier.* Paris: Hermes.

Oustaloup, A. (1995). *La D´erivation non Enti`ere.* Paris: Hermes.

Padula & Visioli. (2011). Tuning rules for optimal PID and fractional-order PID controllers. *Journal of Process Control, 21*(1), 69–81.

Podlubny, I. (1994). *The laplace transform method for linear differential equations of the fractional order.* Slovakia: Slovak Academy of Sciences, Institute of Experimental Physics,.

Podlubny, I. (1999). Fractional-order systems and PI^lamda D^delta controllers. *IEEE Transactions, 44*(1), 208–214.

Poinot T, & Trigeassou J. (2003). A method for modelling and simulation of fractional systems. *Signal Processing, 83*(11), 2319–2333.

Q. Zou, Q. Jin, and R. Zhang. (2016). Design of fractional order predictive functional control for fractional industrial processes. *Chemometrics and Intelligent Laboratory Systems*, 34–41.

Ramaker C. R. & Cutler B.L. (1980). Dynamic Matrix Control- A Computer Control. *In Automatic Control Conference.* San Francisco.

Ramaker C.R. & Cutler B.L. (1979). Dynamic Matrix Control – A Computer Control Algorithm. *AIChE 86th National Meeting.* Houston, Texas.

Raychaudhuri, S. (2008). INTRODUCTION TO MONTE CARLO SIMULATION. *Proceedings of the 2008 Winter Simulation Conference.* Miami, FL, USA.

Razminia A.& Torres D. F. M. (2013). Control of a novel chaotic fractional order system using a state feedback technique. *Mechatronics, 23*, 755–763.

Richalet J., Rault A., Testud J.L. & Papon J. (1976). Algorithmic Control of Industrial Processes. *In 4th IFAC Symposium on Identification and System Parameter Estimation.* USSR.

Richalet J., Rault A., Testud J.L.& Papon J. (1978). Model Predictive Heuristic Control: Application to Industrial Processes. *Automatica*, (p. 413 - 428).

Richalet. J, Rault. A, Testud. J. L, and Papon. J. (1978). “Model predictive heuristic control: Applications to industrial processes. *Automatica*, 413–428.

Robinson B.D. and Clarke D.W. . (1991). Effects of a Prefilter in Generalized Predictive Control. *Proceedings IEE, Part D, Vol. 138,*, 2-8.

Romero M., de Madrid A. P., Mañoso ., Milanés V., & Vinagre B.M. (2013, January 22). Fractional-Order Generalized Predictive Control: Application for Low-Speed Control of Gasoline-Propelled Cars. *Hindawi Publishing Corporation: Mathematical Problems in Engineering, 2013*, p. 10.

Romero M., de Madrid A., Mañoso C. & Vinagre B. (2012). A Survey of Fractional−Order Generalized Predictive Control. *51st IEEE Conference on Decision and Control.* Maui, Hawaii: IEEE.

Romero M., de Madrid A., Mañoso C. and Hernández R. (2007). Application of Generalized Predictive Control to a Fractional Order Plant. *International Design Engineering Technical Conferences and Computers and Information in Engineering* (p. 1285-1292). Las Vegas, Nevada, USA: ASME.

Romero M., de Madrid A., Mañoso C. and Vinagre B. (2010a). *11th Int. Conf. Control, Automation, Robotics and Vision.* Singapore.

Romero M., Pablo A., Hierro C., & Berlinches R. (2010b). Generalized Predictive Control of Arbitrary Real Order. In Z. G. D. BALEANU (A cura di), *New Trends in Nanotechnology and Fractional Calculus Applications* (p. 411-418). London: Springer.

Romero M., Tejado I., de Madrid A. & Vinagre B. (2011). Tuning predictive controllers with optimization: application to GPC and FGPC. *World World*, (p. 10824–10829). Milan, Italy.

Romero M., Vinagre B. M. & de Madrid A. (2008). GPC Control of a Fractional–Order Plant: Improving Stability and Robustness. *Proceedings of the 17th World Congress The International Federation of Automatic Control.* Seoul, Korea: IFAC.

Rossiter, M. (2003). *Model-based predictive control. A practical approach.* CRC, Boc Raton.

Shah P. & Agashe S. (2016). Review of fractional PID controller. *Mechatronics*, 29–41.

Shannon, C. (1993). *Claude Elwood Shannon Collected Papers.* (S. N. A., A cura di) New Jersey: Wiley-Blackwell.

Shinskey, F. (1994). *Feedback Controllers for the Process Industries.* New York: McGraw-Hill.

Sierociuk D. & Dzielinski A. (2006). FRACTIONAL KALMAN FILTER ALGORITHM FOR THE STATES, PARAMETERS AND ORDER OF FRACTIONAL SYSTEM ESTIMATION. *International JourJournal of Applied Mathematics and Computer Science, 16*(1), 129-140.

Sierociuk, D. (2010). *Fractional Order Discrete States-Space System Simulink Toolkit User Guide.* Warsaw, Poland.

SILVA M. F.,MACHADO J. A. T. & LOPES A. M. . (2004). Fractional Order Control of a Hexapod Robot. *Nonlinear Dynamics, 38*, 417–433.

Silva M. F.,Machado J. A. T. & Lopes A. M. (2004). Fractional Order Control of a Hexapod Robot. *Nonlinear Dynamics, 38*, 417–433.

Sondhi S., & Hote Y. (2012). "Fractional order controller and its applications: A review. *AsiaMIC.* Phuket, Thailand.

Spanier K. B. & Oldham J. (2006.). The Fractional Calculus: Theory and Applications of Differentiation and Integration to Arbitrary Orde. Dover, New York.

Sundaravadivu K., Jeyakumar V. & Saravanan K. . (2011). Design of Fractional Order PI Controller for Liquid level Control of Spherical tank Modeled as Fractional Order System . *IEEE International Conference on Control System, Computing and Engineering.* Penang, Malaysia: IEEE.

Taylor C. J., Tych W., Young P. C. & Pedregal D. J. (1999). CAPTIN MATLAB toolbox.

Taylor C.J. (2004). Environmental test chamber for the support of learning and teaching in intelligent control. *International Journal of Electrical Engineering Education, 41(4)*, 375–387.

Taylor, C.J., Chotai, A. and Burnham, K.J. (2011). Controllable forms for stabilising pole assignment design of generalised bilinear systems. *Electronics Letters*, 437–439.

Taylor, C.J., Young, P.C., Tych, W., Wilson, E.D. (2018). New developments in the CAPTAIN Toolbox for Matlab with case study examples. *18th IFAC Symposium on System Identification (SYSID)* (p. 694–699). Stockholm, Sweden: Elsevier, IFAC–PapersOnLine.

Tejado I., Valério D., Pires P. & Martins J. (2013). Fractional order human arm dynamics with variability analyses. *Mechatronics, 23*, 805–812.

The MathWorks, I. (2017). *MATLAB user's guide.* Tratto da MATLAB: https://www.mathworks.com/help/pdf_doc/optim/optim_tb.pdf

Tosti E. & Boni R. (2004). Electrical events during gamete maturation and fertilization in animals and humans. *Hum Reprod Update, 10*, 53–65.

Tsitsimpelis, I., & Taylor, C. J. . (2015). Partitioning of indoor airspace for multi-zone thermal modelling using hierarchical cluster analysis. *European Control Conference (ECC)* (p. 410-415). IEEE.

Valerio & Costa. (2006). Tuning of fractional PID controllers with Ziegler–Nichols type rules. *Signal Processing, 86*, 2771–2784.

Verma S., Yadav S. & Nagar S. (2016). Fractional Order PI Controller Design for Non-Monotonic Phase Systems. *International Federation of Automatic Control* (p. 236-240). Paris: Elsevier.

Vinagre B.M., Monje C.A. & Calder´on A.J. . (2004). The fractional integrator as a reference function. *the First IFAC Workshop on Fractional Differentiation and Its Applications.* Bordeaux, France.

Wilson, E.D., Clairon, Q., Henderson, R. and Taylor, C.J. (2019). Robustness evaluation and robust design for proportional–integral–plus control. *International Journal of Control*, 2939–2951.

Xue D. & Chen Y. (2002). A comparative introduction of four fractional order controllers. *Proceedings of the 4th World Congress on Intelligent Control and Automation, 4*, 3228–3235.

Xue D., Zhao C., & Chen Y. (2006). A modified approximation method of fractional order system. In Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA ’06)., (p. 1043–1048).

Young, P. C. (2011). *Recursive Estimation and Time-Series Analysis* (second edition ed.). Lancaster: Springer.

Yu P., Li J. & Li J. (2015). The Active Fractional Order Control for Maglev Suspension System. *Mathematical Problems in Engineering, 2015*.

Zhang Y. & Li J. (2011). Fractional-order PID Controller Tuning Based on Genetic Algorithm. IEEE.

Zheng S., Tang X. & Song B. (2014). A graphical tuning method of fractional order proportional integralderivative controllers for interval fractional order plant. *Journal of Process Control, 24*, 1691-1709.