Improved generalized predictive controllers for decentralized control

Karunakaran, Giridharan

2013

Karunakaran, G. (2013). Improved generalized predictive controllers for decentralized control. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/54966>

<https://doi.org/10.32657/10356/54966>

IMPROVED GENERALIZED PREDICTIVE CONTROLLERS FOR DECENTRALIZED CONTROL

KARUNAGARAN GIRIDHARAN

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2013

Acknowledgment

The work of the last four years has culminated into the ideas presented in this thesis. As I sit here, with a sense of completion, pondering the last four wonderful years, I realize that I have many people to thank. It was with their guidance, help and support that I sailed through the PhD process.

I thank my supervisor Prof. Cai Wenjian, who first inspired in me an interest in process control through the classes I had attended as a Masters student in NTU. And then, he nurtured this interest by offering me this PhD position. I am grateful for the freedom he gave me in the pursuit of my own crazy solutions to the PhD problem. I had a lot of pitfalls, confusions and other assortments of problems, but a few words of practical advice from him always helped to bolster my failing emotions.

I express my gratitude to this wonderful university for the financial support, facilities and opportunity to pursue my academic goals. The verdant gardens in the campus were the workshops where I would often go to pick new ideas and mould new solutions when I was stuck on a problem or two.

Two years into my PhD, my roommate and friend Srikanth M.K. introduced me to Yoga. Through the practise of Hatayoga, my health problems vanished, my mind achieved a certain balance, and my emotions settled to a steady state. It thus moulded me into the a newer and balanced version of myself, a version that had the necessary clarity of thought to conduct this PhD process in an efficient manner. *Yoga Karmasu Kaushalyam*, indeed. Thus, I express my sincere love and gratitude to the friend who helped me start yoga. I thank all the volunteers and instructors of the Nikam Guruji Yoga Kutir (NGYK) group in Singapore. I thank Mr. Atul Deshpande, the founder of NGYK, who took up his Guru's unflinching ideal to teach the ancient yogic techniques free of cost, for the benefit of humanity's health. I also thank all my teachers at the Queenstown yoga center who inspired me.

I bow down to my teachers, my parents and my friends.

Giridharan Karungaran

Contents

1	Introduction	1
1.1	Introduction - Decentralized Control	1
1.2	Introduction - GPC	2
1.3	Literature Survey - GPC Tuning Methods	5
1.3.1	Motivation - N^* tuning method	6
1.4	Literature Survey & Motivation - PCS	7
1.5	Literature Survey - Variants of GPC for Robustness	9
1.5.1	Motivation - 2GPC	11
1.5.2	Motivation - CDGPC	12
1.6	Contributions and Organization of Thesis	13
2	The N^* tuning method	14
2.1	Overview of the GPC for FOPTD Systems	15
2.1.1	FOPTD CARIMA model	15
2.1.2	Prediction Equation	16
2.1.3	Prediction Vector	17
2.1.4	Control Law	18
2.1.5	Optimal Predictor	18
2.1.6	Closed Loop Transfer Function & Nominal Characteristic Equation	19
2.1.7	Poles of the Characteristic Equation	21
2.2	Study of Pole Movements	22
2.2.1	Pole Movement, varying N with constant λ	22
2.2.1.1	Stable System:	22
2.2.1.2	Unstable System:	23
2.2.2	Finding N_l	24
2.2.2.1	Stable System:	25
2.2.2.2	Unstable System:	26
2.2.3	Finding N_l^*	26
2.2.3.1	Stable System:	27

2.2.3.2	Unstable System:	27
2.2.4	Varying λ	30
2.2.4.1	Stable System:	30
2.2.4.2	Unstable System:	30
2.2.5	Tuning rule	32
2.3	Properties of the N^* method	34
2.3.1	Performance	34
2.3.2	Nominal and Internal Stability	36
2.3.3	Robustness	38
2.3.4	Sampling Time and k selection for stable FOPTD systems	41
2.3.5	Dead-Beat and Pole Placement	41
2.3.6	Differences between using N_l and N_l^*	43
2.4	Simulations and Experiments	43
2.4.1	Example 1 - stable system	43
2.4.2	Example 2 - unstable system	46
2.4.3	Level Control in a Single Tank System	49
2.5	Chapter Conclusions and Future work	52
2.5.1	Summary of the Tuning Procedure	53
3	The Parallel Control Structure (PCS)	55
3.1	The Basics of the PCS	56
3.1.1	Prefilter Form	58
3.1.2	Transparent Online Tuning on the PCS	59
3.2	PCS Design Procedure	59
3.2.1	Design of Controllers and Asymptotic Constraints	59
3.2.2	Single Online Tuning Parameter	63
3.2.3	Bounds on the single online tuning parameters	63
3.2.3.1	From Nominal Stability (Routh-Hurwitz Stability)[34]	63
3.2.3.2	From Robust Stability considerations[34]	63
3.2.3.3	From Gain Margin Specifications	64
3.3	Transparent Online Tuning	64
3.3.1	Effect of online tuning of g_{c2} and g_{c1} on Disturbance Rejection Response	65
3.3.2	Effect of online tuning of g_{c2} and g_{c1} on Set-Point Tracking Response	66
3.3.3	Example	68
3.3.4	Greater Robustness	69
3.3.5	Summary	70
3.4	Simulations and Experiments	70

3.4.1	Example 1. Same Controller Type, Different Parameters	70
3.4.2	Example 2. Different Controllers	70
3.4.3	Example 3. Transparent Online Tuning - FOPTD system	72
3.4.3.1	Nominal Case	72
3.4.3.2	Mismatch case	73
3.4.4	Example 4. Transparent Online Tuning - Integrator Systems . . .	75
3.4.4.1	Nominal Case	75
3.4.4.2	Mismatch Case	77
3.4.5	Level Control in a Single Tank System	79
3.5	Discussion	82
3.6	Chapter Conclusion	83
4	2GPC	85
4.1	The 2GPC Controller	86
4.1.1	Designing the 2GPC	86
4.1.2	The 2 Cost Functions	88
4.1.3	Combining the Cost Functions	90
4.1.4	Combining Constraints	91
4.1.5	Constrained Optimization	93
4.2	Unconstrained 2GPC	94
4.2.1	Closed Loop Transfer Function	95
4.2.2	Two Properties	96
4.3	Transparent Online Tuning, Robustness & Stability	96
4.3.1	Effect of online tuning of GPC_1 and GPC_2 on Disturbance Rejection Response	98
4.3.2	Effect of online tuning of GPC_1 and GPC_2 on Set-Point Tracking Response	99
4.3.2.1	Numerical Example	101
4.3.3	Robustness	101
4.3.4	Stability	101
4.3.5	Summary	103
4.4	Simulation and Experiments	103
4.4.1	Transparent Tuning	103
4.4.2	Robustness	104
4.4.3	Level Control in a Single Tank System	106
4.5	Discussions	108
4.6	Chapter Conclusions	113

5	GPC with Constant Disturbance Model (CDGPC)	114
5.1	CDGPC - General Method	115
5.1.1	Prediction Equation	115
5.1.2	Prediction Vector	118
5.1.3	Control Law	121
5.1.3.1	Transfer Function form	122
5.1.3.2	Matrix Form (for implementation)	122
5.1.3.3	At Steady State	123
5.1.4	Control Structure & Closed Loop Transfer Function - General Method	123
5.1.4.1	Primary Controller	123
5.1.4.2	Closed Loop Transfer Function	124
5.1.4.3	Alternative look at the Control Law	124
5.2	CDGPC - Predictor Seperate Derivation	125
5.2.1	Prediction Equation	125
5.2.2	Prediction Vector:	125
5.2.3	Control Law: Case ($d > n_a$)	126
5.2.3.1	Transfer Function Form	126
5.2.3.2	Matrix form (for implementation)	126
5.2.4	CDGPC Prediction Equation	127
5.2.4.1	Matrix Form	127
5.2.4.2	Transfer Function Form	129
5.2.4.3	At Steady State: Case ($d > n_a$)	131
5.2.5	Control Structure & Closed Loop Transfer Function: Predictor Separate Case ($d > n_a$)	131
5.2.5.1	Primary Controller	131
5.2.5.2	Closed Loop Transfer Function	132
5.2.5.3	Alternative look at the Control Law:	133
5.3	Properties of the CDGPC	133
5.3.1	Robustness of CDGPC	133
5.3.2	Nominal and Internal Stability	135
5.3.3	CDGPC with N^* tuning for FOPTD processes	135
5.4	Simulations and Experiments	136
5.4.1	Example 1 - Delay Uncertainty	136
5.4.2	Example 2 - Worst-Case Uncertainty	140
5.4.3	Level Control in a Single Tank System	142
5.5	Chapter Conclusions	144

6	RNGA-based Decentralized GPC	146
6.1	Multivariable Loop Pairing Methods	147
6.1.1	Dynamic Relative Gain Array (DRGA)	147
6.1.1.1	Deriving DRGA from first principles	148
6.1.1.2	DRGA by First Principles	152
6.1.1.3	Properties of DRGA	153
6.1.2	Relative Gain Array (RGA)	153
6.1.2.1	First Principles Method	154
6.1.2.2	Matrix Method	156
6.1.2.3	Properties of RGA	157
6.1.3	Niederlinski Index	158
6.1.4	Relative Effective Gain Array (REGA)	158
6.1.4.1	Relative Critical Frequency Array (RCFA)	162
6.1.5	Relative Normalized Gain Array (RNGA)	162
6.1.5.1	Relative Average Residence Time Array (RARTA)	165
6.2	RNGA based Decentralized PID control	166
6.2.1	ETFs for Loop Integrity	167
6.2.2	Numerical Example	170
6.3	RNGA-based Decentralized GPC control	173
6.3.1	Proposed Procedure	173
6.4	2×2 Case Studies	177
6.4.1	Wardle & Wood	178
6.4.2	Wood and Berry	180
6.4.3	Vinante & Luyben	182
6.4.4	Industrial-Scale Polymerization Reactor	184
6.5	Chapter Conclusions and Future Work	185
i	Appendix: Example Systems	194
i.1	2×2 Case Studies	194
i.1.1	<i>System 1:</i> Wardle & Wood	194
i.1.2	<i>System 2:</i> Wood & Berry	195
i.1.3	<i>System 3:</i> Vinante & Luyben	195
i.1.4	<i>System 4:</i> Industrial-Scale Polymerization Reactor	196
ii	Appendix: Robust Stability for Discrete-time Control Loop	198
iii	Appendix: Worst-case Multiplicative Uncertainty	199
iv	Appendix: Gain Margin Phase Margin PID tuning method	201
v	Appendix: Derivation of GPC Control Law	202
v.1	General Derivation	203
v.2	Predictor Separate Derivation	204

List of Figures

1.1.1 A 2×2 MIMO Decentralized Closed Loop	2
1.2.1 GPC - implementation form	4
1.2.2 GPC - the analytical form	4
1.4.1 Conventional Control Structure (CCS)	7
1.5.1 GPC	10
1.5.2 PCS	10
2.2.1 Stable Case: varying N (from 1 to 16) with fixed $\lambda = 4.5$	23
2.2.2 Unstable Case: varying N (from 1 to 11) with fixed $\lambda = 9$	24
2.2.3 Stable Case: varying N (from 0.1 to 16) with fixed $\lambda = 4.5$	26
2.2.4 Unstable Case: varying N (from 0.1 to 16) with constant $\lambda = 9$	27
2.2.5 Stable Case: r & $1 - \sqrt{k_1 b}$ vs N	28
2.2.6 Unstable Case: r , $1 + \sqrt{k_1 b}$ & $1 - \sqrt{k_1 b}$ vs N	29
2.2.7 Stable Case: pole movement with continuous N for $\lambda = 1, 3, 5$	31
2.2.8 Unstable Case: pole movement with continuous N for $\lambda = 1, 3, 5$	31
2.2.9 Stable Case: Varying k from 0.01 to 0.9	34
2.2.10 Unstable Case: Varying k from 0.01 to 0.9	35
2.3.1	37
2.3.2 Stable Case: Performance vs Robustness while varying k from 0.1 to 0.9 .	39
2.3.3 Unstable Case: Performance vs Robustness while varying k from 0.1 to 0.9	40
2.3.4 Stable Case: Closed loop responses and Robustness for desired $c = 2, 4, 8$	42
2.3.5 Stable Case: Differences between using $N = N_l^*$ and $N = \text{ceiling}(N_l^*)$ for $k = 0.3, 0.5, 0.9$	44
2.3.6 Unstable Case: Differences between using $N = N_l^*$ and $N = \text{ceiling}(N_l^*)$ for $k = 0.3, 0.5, 0.9$	45
2.4.1 Stable Case: Comparing closed loop responses & Robustness Limits of the N^* and Shridhar-Cooper methods. (A setpoint of 1 is applied at $t = 0$ and an output disturbance of -1 is applied at $t = 15$)	47

2.4.2 Unstable Case: Comparing closed loop responses & Robustness Limits of the N^* and Shridhar-Cooper methods. (A setpoint of 1 is applied at $t = 0$ and an output disturbance of -1 is applied at $t = 30$)	48
2.4.3 The water tank testbed	49
2.4.4 Closed loop step responses of the level control experiment with sampling time $T = 0.1s$ (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	50
2.4.5 Closed loop step responses of the level control experiment with sampling time $T = 0.3s$ (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	51
3.1.1 The Parallel Control Structure	56
3.1.2 PCS Prefilter Form	59
3.2.1 A summary of the design procedure	60
3.3.1 Comparing Sensitivities of $S_r^{(g_p \neq g_m)}$ and $S_d^{(g_p \neq g_m)}$ to g_{c2}	69
3.4.1 Closed Loop Responses of the CCS with pid_A , CCS with pid_B & the PCS with pid_A & pid_B , when there is white noise in the measurement	71
3.4.2 Closed Loop Responses of the CCS with g_{c1} , CCS with g_{c2} & the PCS with g_{c1} & g_{c2} , when the disturbance is a ramp (applied at $t = 50$)	72
3.4.3 PCS closed loop responses under nominal conditions with (a) varying A_{m1} ($A_{m2} = 3$) (b) varying A_{m2} ($A_{m1} = 3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 10$)	74
3.4.4 PCS closed loop responses under 20% worst-case mismatch with (a) varying A_{m1} ($A_{m2} = 3$) (b) varying A_{m2} ($A_{m1} = 3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 15$)	75
3.4.5 PCS closed loop responses under nominal conditions with (a) varying λ_1 ($\lambda_2 = 0.3$) (b) varying λ_2 ($\lambda_1 = 0.3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 40$)	78
3.4.6 PCS closed loop responses under 20% worst-case mismatch with (a) varying λ_1 ($\lambda_2 = 0.3$) (b) varying λ_2 ($\lambda_1 = 0.3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 40$)	79
3.4.7 2PID Experimental Results: Varying the gain of g_{c1} (via A_{m1}) varies the <i>Tracking Performance</i> alone ($A_{m2} = 40$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	80
3.4.8 2PID Experimental Results: Varying the gain of g_{c2} (via A_{m2}) varies only the <i>Disturbance Rejection Performance</i> alone ($A_{m1} = 40$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	81

3.4.9 2PID Mismatch Results: Varying the gain of g_{c2} (via A_{m2}) varies <i>mainly</i> the <i>Disturbance Rejection Performance</i> ($A_{m1} = 60$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	82
3.5.1 The general Two Degree of Freedom Control Structure	83
4.1.1 The 2GPC Control Loop	87
4.3.1 Comparing Sensitivities of $S_r^{(g_p \neq g_m)}$ and $S_d^{(g_p \neq g_m)}$ to g_{c2}	102
4.4.1 Closed loop responses of the 2GPC for the Nominal case: (a)Varying k_1 ($k_2 = 0.7$) (b)Varying k_2 ($k_1 = 0.7$) (A setpoint of 1 is applied at time $t = 0$ and a disturbance of -1 is applied at $t = 10$)	104
4.4.2 Closed loop responses of the 2GPC for the Mismatch case (a)Varying k_1 ($k_2 = 0.95$) (b)Varying k_2 ($k_1 = 0.9$) (A setpoint of 1 is applied at time $t = 0$ and a disturbance of -1 is applied at $t = 15$)	105
4.4.3 Robustness Limits of the 2GPC for the two cases ($k_1 = 0.87, k_2 = 0.87$) and ($k_1 = 0.87, k_2 = 0.95$)	107
4.4.4 Closed Loop Responses of the 2GPC for different $k_1 k_2$ under mismatch settings (the delay of the plant and model differ by 2 samples) (A setpoint of 1 is applied at $t = 0$ and an output disturbance of -1 is applied at $t = 50$)	108
4.4.5 2GPC Experimental Results: Varying k_1 varies the <i>Tracking Performance</i> alone ($k_2 = 0.98$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	109
4.4.6 2GPC Experimental Results: Varying k_2 varies the <i>Disturbance Rejection Performance</i> alone ($k_1 = 0.98$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	110
4.4.7 2GPC Mismatch Results: Varying k_2 varies <i>mainly</i> the <i>Disturbance Rejection Performance</i> alone ($k_1 = 0.98$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	111
5.2.1 Unconstrained CDGPC control loop - Predictor Separate Case $d > n_a$	132
5.4.1 SPGPC, CDGPC & GPC Robustness Limits & delay multiplicative uncertainty $ \Delta^*(e^{j\omega}) $ vs ω	138
5.4.2 Closed Loop Responses of GPC, SPGPC & CDGPC for delay uncertainty (A setpoint of 1 was applied at $t = 0s$ and an output disturbance of -1 was applied at $t = 25s$)	139
5.4.3 Robustness Limits of GPC, SPGPC & CDGPC for different process delays (Robustness of SPGPC and CDGPC are independent of delay)	140

5.4.4 SPGPC, CDGPC & GPC Robustness Limits & worst-case multiplicative uncertainty $ \Delta_w^*(e^{j\omega T}) $	141
5.4.5 Closed Loop Responses of GPC, CDGPC & SPGPC for worst-case uncertainty (A setpoint of 1 was applied at $t = 0s$ and an output disturbance of -1 was applied at $t = 25s$)	142
5.4.6 Closed Loop Responses of SPGPC, CDGPC & GPC under nominal condition ($k = 0.97$, $N = \text{ceil}(N_t^*)$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	143
5.4.7 Closed Loop Responses of GPC, CDGPC & SPGPC under a delay mismatch of $1.5s$ ($k = 0.97$, $N = \text{ceil}(N_t^*)$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)	144
6.1.1 Loop $y_2 - m_2$ closed	148
6.1.2 Loop $y_2 - m_1$ closed	150
6.1.3 Loop $y_1 - m_2$ closed	151
6.1.4 Loop $y_1 - m_1$ closed	151
6.1.5 Bandwidth Frequency	160
6.1.6 Ultimate Frequency	160
6.2.1 Case $\lambda_{ij} < 1$ and $\gamma_{ij} > 1$	168
6.2.2 Case $\lambda_{ij} < 1$ and $\gamma_{ij} < 1$	169
6.2.3 Case $\lambda_{ij} > 1$ and $\gamma_{ij} < 1$	169
6.2.4 Comparison of closed loop responses of RNGA-based vs no-ETF PID design. (A step of 1 was applied at $t = 0$ to loop 1 and at $t = 150s$ to loop 2)	172
6.2.5 Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. Top: $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*	173
6.3.1 Comparison of closed loop responses of RNGA-based vs no-ETF CDGPC design. (A step of 1 was applied at $t = 0$ to loop 1 and at $t = 150s$ to loop 2)	175
6.3.2 Left: Outputs 1 and 2 Right: Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 150$ a setpoint of 1 is applied to Loop 2	177
6.4.1 Wardle & Wood - Left: Outputs 1 and 2 Right: Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 300$ a setpoint of 1 is applied to Loop 2	178

6.4.2 Wardle & Wood - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. Top: $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*	179
6.4.3 Wood & Berry - Left: Outputs 1 and 2 Right: Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 100$ a setpoint of 1 is applied to Loop 2	180
6.4.4 Wood & Berry - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. Top: $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*	181
6.4.5 Vinante & Luyben - Left: Outputs 1 and 2 Right: Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 50$ a setpoint of 1 is applied to Loop 2	182
6.4.6 Vinante & Luyben - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. Top: $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*	183
6.4.7 Industrial-Scale Polymerization Reactor - Left: Outputs 1 and 2 Right: Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 25$ a setpoint of 1 is applied to Loop 2	184
6.4.8 Polymerization Reactor - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. Top: $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*	185
v.1 Unconstrained GPC control structure - Implementation Form	204
v.2 Unconstrained GPC control loop - Predictor Seperate Structure: Case $d > n_a$	206

Summary

In general, decentralized control refers to the multivariable control of a $n \times n$ square process using n SISO (Single-Input Single-Output) control loops designed for the diagonal elements. Under consideration is an equivalent transfer function method for decentralized controller design proposed by Cai et.al.[24] known as the (Relative Normalized Gain Array) RNGA-based decentralized PID design method. The design procedure involves loop pairing using the RNGA, RGA (Relative Gain Array) and NI (Niederlinski Index) analysis, deriving the RNGA-based equivalent transfer functions (ETFs) of the n diagonal elements and using them to design PID controllers for the n loops. The objective of this work was to use the RNGA-based ETFs to design GPCs (Generalized Predictive Controllers) in place of the standard PID in order to exploit some of the inherent features of the GPC such as constraints handling. The GPC or Generalized Predictive Controller is one member of the family of long range predictive controllers (or Model Predictive Controllers) which was proposed in 1979 by Clarke et.al. to handle unstable, non minimum phase processes. It differs from other varieties of Model Predictive Controllers (MPC) in that it uses the CARIMA (Controlled Auto-Regressive Integrated Moving Average) model of the process to derive its output predictions.

Simulation studies proved that the SISO unconstrained GPC was unsuitable for direct application to a decentralized structure. Modifications were needed on two fronts: the parameter tuning and disturbance model of the GPC (for Robustness). Research in the tuning direction led to the development of two novel tuning methods: The first is the N^* tuning method which is applicable to the conventional GPC for the control of stable and unstable FOPTD (First Order Plus Time Delay) processes. The second is the 2GPC method which is an extension of the general Parallel Control structure (PCS) to predictive control. The 2GPC algorithm is offered as a variant of the conventional GPC. It consists of two GPCs working in tandem; one governs the set-point tracking response and the other controls the disturbance rejection response. However, the 2GPC is formulated as a single optimization problem integrating system constraints. The extent to which the 2GPC algorithm can perform Transparent Online Parameter Tuning (TOPT) is explained. (Transparent Online Parameter Tuning (TOPT) is the facility of the controller to allow the user to independently manipulate online the three most important loop performance attributes - *Set-point Tracking Performance*, *Disturbance Rejection Performance* and *Robustness* - with the use of three separate parameters). It is shown through derivations that the extent to which the 2GPC method can perform TOPT is maintained even under mismatch conditions. Most importantly, it is also proved that the 2GPC control loop, by utilization of its TOPT feature, can have greater *Robustness* than a conventional GPC loop.

On the disturbance model front, the structure of the GPC was first analysed. The structure of the conventional unconstrained SISO GPC can be split into a primary loop with setpoint filter and an optimal predictor. The filter transfer function of this optimal predictor is inversely proportional to the loop robustness and is also the only transfer function in the GPC loop that is a function of the process delay. For higher delays, the filter is such that the Robustness deteriorates. But it is the optimal predictor's filter that is responsible for one of the GPC's most attractive features - guaranteed internal stability even in the case of open loop unstable processes. At the loss of this feature, in order to improve disturbance rejection/robustness, there are variants of the GPC that utilize modified filters, such as the SPGPC (Smith Predictor based GPC). A new GPC is proposed in this work called the CDGPC (or GPC with Constant Disturbance Model). This effectively makes it equivalent to the popular DMC (or Dynamic Matrix Controller) with the exception that the former uses the transfer function model for predictions and can thus work for integrator systems as well. It is proved that the CDGPC has greater robustness than the conventional GPC and the SPGPC.

The CDGPC together with N^* tuning method generate the required level of robustness and precision in tuning that enables it to be applied to decentralized control. The CDGPC with N^* tuning is designed for the RNGA-based ETFs of the diagonal elements of the MIMO system. Closed loop responses of 2×2 MIMO systems were studied in simulation and were compared to the performance of RNGA-based PID controllers.

Publications Related to the Thesis

1. Giridharan Karunagaran and CaiWenjian. The parallel control structure for transparent online tuning. *Journal of Process Control*, 21(7):1072–1079, 2011.
2. Giridharan Karunagaran and CaiWenjian. A novel parameter tuning method for the unconstrained GPC for FOPTD stable and unstable processes, *submitted to Journal of Process Control*, April 2012. (*under review*)

Chapter 1

Introduction

1.1 Introduction - Decentralized Control

There are two fundamental approaches to multivariable control. The first and more straightforward method is the Centralized control scheme where a single controller takes all output measurements at once and calculates control actions on all inputs simultaneously. The second approach is the Decentralized control scheme.

In general, decentralized control refers to the multivariable control of a $n \times n$ square process using n SISO (Single-Input Single-Output) control loops designed for the diagonal elements. The challenge in decentralized control is in assessing the interaction effect of one loop on another loop in order to decide the loop pairing and then designing controllers accordingly. However, there are several obvious advantages to the Decentralized method. It allows independent design of controllers, it is easier to tune and cost effective to implement.

There are several methods for decentralized design. A few are listed here: BLT (Biggest Log Modulus Tuning) method [42, 36], Independent Design Method [65, 27], simple multiloop tuning method derived from controller synthesis method [12], Nysquist Stability analysis based tuning approaches [11, 31], Dominant Pole Placement method [81], approach based on gain and phase specifications [25], SVD based approach [55] and Sequential Loop Design approach [28, 60, 38].

But the focus of the work presented in this thesis is on methods which use Effective Transfer Functions for controller design. The Effective Transfer Function or ETF is the approximation of the transfer function between an input and output of the MIMO system when all other loops are in perfect control. An interaction analysis technique such as the DRGA (Dynamic Relative Gain Array) [71], REGA (Relative Effective Gain Array) [74, 72], RGA (Relative Gain Array) [12, 39] or RNGA (Relative Normalized Gain Array) [24] is used to generate these ETFs. These interactions measures also double as

interaction measure tools to decide the loop pairing.

One such effective transfer function method for decentralized controller design was proposed by Cai et.al.[24], known as the RNGA-based decentralized PID design method. The design procedure involves interaction analysis and loop pairing using the RNGA (Relative Normalized Gain Array), RGA and NI (Niederlinski Index) analysis, deriving the RNGA-based Effective transfer functions (ETFs) of the n diagonal elements and using them to design PID controllers for the n loops. (Further details on RNGA-based Loop Pairing and the RNGA-based decentralized PID design will be presented in Chapter(6))

Figure 1.1.1: A 2×2 MIMO Decentralized Closed Loop

The main objective of this work was to apply the Generalized Predictive Controller (GPC) for decentralized control using the RNGA-based ETF strategy. The GPC was chosen from amongst several other members of the family of Model Predictive Controllers (MPC) because it uses the transfer function model and so does not require separate observer designs which could complicate the dynamics of a decentralized implementation. Another reason is that the unconstrained GPC's control law and structure can be derived explicitly and analysed[10].

1.2 Introduction - GPC

The Generalized Predictive Controller is a popular member of the family of Model Predictive Control algorithms and was introduced by Clarke et.al. in 1987 [15, 14, 13] to handle non-minimum phase, long dead-time and open loop unstable processes[4, 6]. Its versatility has attracted both the industry and academia[10].

The GPC algorithm minimizes an index which is the sum of squares of the errors of a future output trajectory and a future setpoint signal over a prediction horizon plus the sum of squares of change in control actions over a future horizon called the control horizon.

It is essentially an open-loop optimization problem that is repeated at every sample time updated with the latest feedback information at the time of computation. The GPC uses the CARIMA (or Controlled Auto-Regressive Integrated Moving Average) model to predict the future output trajectory. The GPC also takes into account the various constraints inherent to the process such as slew rates, control levels, output constraints for computing the optimal control action at each sample instant.

The CARIMA model for any SISO plant is of the form,

$$A(z^{-1})y_p(t) = B(z^{-1})z^{-d}u(t-1) + \frac{\epsilon(t)}{\Delta} \quad (1.2.1)$$

where,

$$\begin{aligned} A(z^{-1}) &= 1 + a_1z^{-1} + \dots + a_{n_a}z^{-n_a} \\ B(z^{-1}) &= b_0 + b_1z^{-1} + \dots + b_{n_b}z^{-n_b} \end{aligned}$$

(The degrees of the polynomials are $\delta(A) = n_a$ and $\delta(B) = n_b$). d is the discrete-time delay, $\Delta = (1 - z^{-1})$ and $\epsilon(t)$ is the stochastic noise term.

The optimization problem or performance index to be minimized is,

$$\begin{aligned} \min_U \quad & J \\ \text{s.t. } & RU \leq r \end{aligned} \quad (1.2.2)$$

with,

$$J = (Y - W)^T (Y - W) + U^T (\lambda I) U$$

where, Y is the vector of future output predictions, U is the vector of future change in control actions, W is the vector of future set-point signal values, and λ is a parameter that balances the importance given to the output error vs the future change-in-control actions. At every iteration, this optimization is solved but only the first element of the optimal U is applied to the plant, in accordance with the *receding horizon principle*. R and r are the constraint matrices (all constraints can be lumped together into this form).

In the absense of constraints, the optimal solution to Eq.1.2.2 can be found explicitly. Depending on how the future output predictions are derived from the CARIMA model, the structure of the GPC can be expressed in two forms, both of which are equivalent, but one of which is extremely helpful in assessing the GPC's properties[10, 52]. Figures (1.2.1) and (1.2.2) show the closed loop structure of the unconstrained SISO GPC in the two possible ways. Appendix(v) will present the exact derivation of the control laws in both ways.

The structure in Figure(1.2.2) helps explain some properties of the GPC:

Figure 1.2.1: GPC - implementation form

Figure 1.2.2: GPC - the analytical form

- **Robustness:** The magnitude of the reciprocal of the primary loop's complementary sensitivity is called the Robustness Limit of a loop $\Delta_{lim}^*(\omega)$; the greater this is across all frequencies, the higher the robustness of the loop. Under nominal conditions, the primary loop's complementary sensitivity is (See Appendix(v) for exact derivation),

$$\frac{g_m L_2 F_r}{\Delta (1 + L_3 z^{-1}) + L_2 g_m^*}$$

where g_m is the transfer function model of the process,

$$g_m(z^{-1}) = \frac{B(z^{-1})z^{-1}}{A(z^{-1})}z^{-d}$$

and g_m^* is the nominal model of the process without the delay. Thus, the Robustness is inversely proportional to $|F_r|$. Furthermore, the filter F_r is a function of the process delay, d . And the structure of the filter is such that for a larger d , the Robustness is lower. Hence, the GPC is not as robust for large delay systems as it is with smaller delay processes.

- **Internal Stability for open loop unstable systems:** If the structure of Fig(1.2.2) was posed into a conventional controller and setpoint filter form, the

primary controller would be,

$$\frac{L_2 F_r}{\Delta(1 + L_3 z^{-1}) + g_m^* L_2 (1 - z^{-d} F_r)}$$

It can be analytically proved that the denominator of g_m^* , $A(z^{-1})$, is a factor of $L_2(1 - z^{-d} F_r)$. Therefore, even when $A(z^{-1})$ includes unstable open loop poles, they do not make to the numerator and hence, no unstable pole zero cancellation occurs[52]. Guaranteed internal stability is a very attractive feature of the GPC.

Directly applying n GPCs for the decentralized control of an $n \times n$ MIMO process failed even when the GPCs were designed and tuned for the RNGA-based ETFs. It was observed that the unconstrained SISO GPC did not have the required robustness and precise tuning to allow it to be applied as a decentralized controller. Research in the directions of GPC tuning and Robustness led to the development of two variants of GPC, namely, the 2GPC and the CDGPC and the development of a precise GPC tuning method called the N^* tuning method. The remainder of this chapter will present the motivations for the development of these techniques as well as relevant literature surveys in the areas of GPC tuning and Robustness. The 2GPC method is an extension of the Parallel Control Structure (PCS) to the predictive control framework and hence literature survey pertaining to the PCS is also provided.

1.3 Literature Survey - GPC Tuning Methods

The tuning of the GPC or the choosing of its four parameters N_1 , N_2 , N_u and λ is crucial for good performance, robustness and even for nominal stability as in the case of unstable systems. Various tuning strategies for the GPC have been exhaustively listed and studied in the work [75] and more recently in [18]. The tuning strategies are presented as explicit tuning formula or as bounds on the tuning parameters or even as guidelines that require some amount of trial and error for one or more parameters.

One of the popular tuning rules for the SISO GPC is the method by Shridhar et.al. [61] which is applicable to a broad class of open loop stable systems. The actual process is approximated to a First Order Plus Time Delay (FOPTD) model and the formula for the GPC parameters are presented as functions of the FOPTD process parameters. The main focus of the work was on the derivation of an analytical expression for the move suppression coefficient, λ . Further tuning work from the same authors for the multivariable GPC can be found in refs [62, 63]. The seminal paper by Clarke et.al.[15] includes general guidelines for parameter selection for the control of stable, unstable, and variable dead-time processes. In the work by McIntosh et.al [40, 41], three tuning strategies

were proposed, namely, the Output Horizon, Lambda Weighting and the Detuned Model Following strategies, each with one active parameter that can allow the practitioner to adjust the closed loop performance on-line.

1.3.1 Motivation - N^* tuning method

The N^* tuning method is a novel tuning rule for the Single-Input-Single-Output (SISO) unconstrained GPC, for first order stable and unstable processes with dead-time based on studies of nominal closed loop pole locations in connection with GPC tuning parameters. The tuning rule reduces the four parameters of the GPC to a single normalized parameter k that can be used to trade-off robustness and performance directly.

While there are a few GPC tuning methods for *stable* FOPTD systems (as indicated by the literature survey), the development of the N^* method also includes the GPC tuning for the unstable FOPTD process for performance and robustness. However, there do exist tuning guidelines given by Clarke and Mohtadi [13] that establish bounds on the GPC parameters for control of processes that are stabilizable and detectable (thus including the unstable FOPTD process), but these bounds are to ensure nominal closed loop stability. For the stable FOPTD process, in general, on the one hand, there are GPC tuning rules that are derived assuming a perfect model and with the objective of obtaining good performance. While on the other hand, in practice, Model Predictive Control (MPC) users are forced to detune controllers to stabilize the loop [18]. With four parameters, the task of detuning or the task of improving robustness of the GPC may not be apparent to the general user unfamiliar with the literature on the relationship between robustness and the GPC tuning parameters [4] and the relationship between performance and the parameters [37, 52, 57]. **The N^* tuning method aims to bridge this divide for the case of the FOPTD stable and unstable process, by incorporating a single normalized parameter k that can directly balance performance and robustness with good results.**

The proposed tuning rule is applicable for the GPC control of the FOPTD system,

$$g_m(s) = \frac{K}{(\tau s + 1)} e^{-Ls} \quad (1.3.1)$$

whose discrete-time version (sampled at T sec) is,

$$g_p(z) = \frac{[b_0 + b_1 z^{-1}] z^{-1}}{1 - a z^{-1}} z^{-d} \quad (1.3.2)$$

When $\tau > 0$, then $a > 1$ and the system is stable, and when $\tau < 0$, then $a > 1$ and the system is unstable. The resulting nominal closed loop characteristic polynomial is

of second order and its solutions (poles) are functions of the GPC parameters. For *any* a , b and d , the movement of the poles when the GPC parameters are varied generates patterns that are similar and the general best pole location in these patterns is identified. Then, a rigorous analytical derivation arrives at the general formula for the values of the tuning parameters needed to place the closed loop poles at the identified best location. The pole movement patterns are only slightly different for the stable and unstable system but the steps of the derivation of the tuning rule for both cases are identical. A parameter k (such that $0 < k < 1$) is incorporated into the derivation and the tuning formula are its functions. As mentioned before, this parameter k can be used to balance performance and robustness directly, with a k closer to 1 giving more robustness while a k closer to 0 giving more performance, each at the cost of the other.

1.4 Literature Survey & Motivation - PCS

Figure 1.4.1: Conventional Control Structure (CCS)

The three most important performance attributes of a control loop are - *set-point tracking performance*, *disturbance-rejection performance* and *robustness*. After implementation of a control system, due to various reasons (such as process variability and uncertainty), the control practitioner has to perform online adjustment or *online tuning* of the controller parameters, in order to achieve the desired performance level in one or more of the three attributes. The Conventional Control Structure (CCS) shown in Figure (1.4.1) with a PID controller is the most popular control scheme for the control of industrial processes [53]. However, in a Conventional Control Structure, the three loop attributes are closely knit and trying to vary one property will affect the other properties. With the aim of tackling this issue of the CCS (and other drawbacks of the PID controller), an alternate controller called the RTD-A controller [44, 45, 53] was proposed, where the three loop attributes can be directly manipulated by three controller parameters. But some of its obvious disadvantages are a combination of approximate tuning rules and stability conditions. Therefore, the development of an alternative control structure that allows for ***Transparent Online Parameter Tuning*** is significant, by which

it is meant that three parameters of the control scheme should directly and independently tune for the three loop attributes. The alternative control scheme of interest is the Parallel Control Structure (PCS), shown in Figure (3.1.1a), with the focus on its capabilities for Transparent Online Tuning.

There are many works in the literature regarding modifications to the conventional Smith Predictor [66] structure with an aim to overcome the problems inherent in it, namely, its poor disturbance rejection and robustness. To achieve this goal, a very small subset of the 'Modified' Smith Predictor literature utilizes the Parallel Control Structure (PCS) with a slight modification for plants with large time delay, as shown in Figure (3.1.1b). In particular, a well recognized feature of the PCS is employed whereby, assuming nominal conditions, the tracking response is decoupled from the disturbance rejection response [68, 69], allowing the user to design independently for each.

- The first such Modified Smith Predictor structure that utilizes the decoupling is Astrom's Smith Predictor [3]. The main focus of the work, however, was the control of processes with an integrator and long dead time, $\frac{e^{-Ls}}{s}$. Some shortcomings of this method were that, for the controller design, only integrator/dead time processes with integral constant 1 were considered and also the proposed tuning rules were not simple.
- These issues were pointed out in [80] where a more logical and systematic controller design procedure for the Modified Smith Predictor structure in Figure (3.1.1b) was presented, for the control of a general integrator/dead time process $K\frac{e^{-Ls}}{s}$. The same group later presented a similar design procedure for FOPTD (First Order Plus Time Delay) processes in [79]. Continuing in this vein, further work in [34] focused on the control of processes with time delays and double integrators.
- Other works [67, 70] provide a systematic direct synthesis design for the two controllers of the structure (in Figure (3.1.1b)), for a class of linear stable processes that are commonly encountered in the industry.
- The PCS structure reported in ref [68] was the basis for a Two Degree of Freedom (TDF) Multivariable Control[30] where along with an inverse-based decoupling controller for disturbance rejection, the feedforward and model-following controllers are utilized for tracking. This was followed by an analytical method[35] for deriving the tracking and disturbance rejection controllers based on H_2 optimal performance specification, along with stability considerations.

The capacity of the PCS for transparent online tuning has not been explored fully. Neither has any general design procedure been established that can

guide the designer towards the proper utilization of the transparent tuning feature of the PCS.

1.5 Literature Survey - Variants of GPC for Robustness

The (unconstrained) GPC structure consists of a primary control loop with an optimal predictor[52], as shown in Figure(1.5.1) (a more general version of the structure in Figure(1.2.2)). As explained in Section (1.2), the magnitude of the filter R in the predictor is inversely proportional to *Robustness*. The R of the conventional GPC (F_r) is a function of the system delay and reduces the Robustness for large delays. Since C , W and R are all functions of the GPC tuning parameters, it is not possible to modify R alone to improve the *Robustness*.

The popular alternative to this is the T -method[10, 7] which is incorporated into the CARIMA model and is considered as a prefilter or an observer; effects of prefiltering can be found in ref [13]. Although C , W and R are T -dependent but it is known that the nominal tracking transfer function is independent of T [78]. Since R is a function of T , the choice of the T polynomial affects the *Robustness* and guidelines for its selection are found in refs [16, 78, 1, 77]. However, selecting the T polynomial is not a trivial issue [10, 56] mainly because of the T -dependence of C and W . Moreover, stronger T filtering does not always provide better robustness, as pointed out with counter-examples in ref [78]. The difficulty of choosing T is overcome in ref [49] where T is used only for the predictor block (in Figure(1.2.2)) while the GPC optimization that yields C and W do not involve T .

Another prefiltering strategy is the Youla (or Q) Parameterization[1] of the GPC; the T -method is a special case of this Q -method. Effects of Prefilters (T and Q poly method) can also be found in refs [14, 56, 76].

The SGPC (Stable GPC) [33] uses H_∞ optimization based on the Youla Parameterization of the GPC to obtain an optimal Q filter. A similar approach is also used by the authors in ref [29]. In SGPC [33], the authors point out that prefiltering the GPC is an alternative to the optimal solution obtained using the H_∞ optimization. But in ref [1], it is pointed out that in a receding horizon algorithm, the H_∞ optimization can be time consuming.

Other methods have successfully obtained higher *Robustness* by replacing the default optimal predictor of the GPC with the Smith Predictor ($R = 1$) and Filtered Smith Predictor [48] (with a special design $R \neq 1$). Not only is the *Robustness* much improved but it also no longer depends on the system delay, as with the conventional GPC. The

Figure 1.5.1: GPC

Figure 1.5.2: PCS

GPC utilizing the Filtered Smith Predictor is also known as the DTC-GPC (Dead-Time Compensator GPC) [52] and has a two-step design procedure. First, the GPC tuning parameters are chosen for the desired *Nominal Tracking Performance*. Then, the filter R is designed for *Robustness* traded off with *Disturbance Rejection Performance*. Use of smith predictors instead of the GPC's optimal predictor were first reported in [47, 50]. Advantages of the SPGPC in real world applications were then reported in ref [51] (Mobile Robot Path Tracking).

An existing method [5] combines signal processing techniques for mismatch uncertainty estimation with the robust design of GPC (based on small-gain theorem). In essence, upper bounds for uncertainty for SISO linear time-invariant systems are estimated and these bounds are used with graphical based tuning guidelines for robust design of the GPC. The GPC's tuning parameters are used to shape the robustness margin. This approach of using the tuning parameters has also been explored exclusively for FOPTD systems in ref [6] and details on how the small gain theorem in combination with tuning guidelines can be used for robust design can be found in ref [4].

1.5.1 Motivation - 2GPC

The 2GPC method is proposed as a new variant of the GPC. The 2GPC algorithm is the adaptation of the Parallel Control Structure (PCS) to the predictive control framework and is posed a single Quadratic Programming (QP) problem integrating system constraints into the formulation. And it inherits all the features of the PCS, that is, it will be shown that the 2GPC can perform *Transparent Online Parameter Tuning* (TOPT) to the exact same extent. It will shown through derivations that the extent to which the 2GPC method can perform TOPT is maintained even under mismatch conditions. Importantly, it will also be shown that by utilizing the TOPT feature of the 2GPC, it is possible to achieve *Robustness* greater than what the conventional GPC can offer. A detailed derivation of the 2GPC, its constraints formulation and unconstrained closed loop transfer function will be presented in Chapter(4).

The motivation for the 2GPC can be explained by highlighting the difficulty in performing offline parameter tuning for disturbance rejection response in the conventional GPC. Hence, what follows is a discussion on offline parameter tuning issues related to the conventional GPC. After which a discussion on online parameter tuning with the GPC and its variants will follow.

1. **Offline tuning:** In the conventional GPC, the setpoint filter W that results from the unconstrained optimization, is a low-pass filter, for open loop stable and integrator systems. Naturally, for any given set of GPC tuning parameters, the disturbance rejection response will be aggressive relative to the tracking response, which may be undesirable. Hypothetically, since the conventional GPC is a Two Degree of Freedom control structure, it should be possible to satisfy both the tracking and disturbance rejection requirements simultaneously. However, the complicated relationship between the controller transfer functions C , W and R and the GPC tuning parameters renders it difficult to choose a parameter set that can do this. Thus, attempting to reduce the relative aggressiveness of the disturbance rejection by parameter tuning (for eg., by increasing suppression coefficient) will make the tracking more conservative and lead to a loss in *Tracking Performance*. (This is in contrast to the PID where it is known, for instance, that if the gain parameter is increased, both the *Tracking Performance* & the *Disturbance Rejection Performance* increase **equally**). Studies of the effect of the tuning parameters on tracking are available in contemporary literature[57, 37]. Available GPC tuning rules focus on providing the best *Tracking Performance* [75, 63, 61, 18] for First Order Plus Time Delay (FOPTD) systems. Thus, it is difficult to handle the disturbance rejection response of the GPC strictly by offline parameter tuning. The same can be said of the offline parameter tuning for *Robustness*.

2. **Online tuning:** Ofcourse, this offline parameter tuning issue can be circumvented by using a disturbance modeling method to improve the disturbance rejection while keeping the same nominal tracking response. For instance, in the the Smith Predictor based GPC (SPGPC), the error term $y_p(t-i) - y_m(t-i)$ is added to the output prediction $y_p(t+d-i)$; this provides a better disturbance rejection response and *Robustness* than the conventional GPC[48]. Also, the FSPGPC, the T or Q -methods can improve *Robustness* by suitable filter design, as mentioned before. But each of these variants of GPC can be considered in the common framework of the block diagram in Figure(1.2.2). And the closed loop transfer function of the structure can show that under mismatch conditions (practical conditions), *tuning R online* (varying a parameter in R) will necessarily affect the tracking transfer function in some manner. Therefore, R lends itself only to a static *offline* design and not to *online* parameter tuning; any post-implementation insufficiency will require a redesign. (This will be explained further in Chapter(4)). Meanwhile, because the 2GPC is essentially the same as the PCS, it is capable of online tuning.

1.5.2 Motivation - CDGPC

In this variant of the GPC, instead of using the CARIMA model for output predictions, the transfer function model is used. Output predictions at time $t+j$ will simply be the predicted output of the model plus the plant-model error at time t , thus assuming it as a **constant disturbance**; hence, the name CDGPC or GPC with Constant Disturbance into the future. Again, as with the conventional GPC, the output prediction can be derived from the transfer function model in two ways. These will be explored in Chapter (5).

The CDGPC is the equivalent to the Dynamic Matrix Control (DMC) which was introduced by Cutler and Ramaker in 1979 [17]. This is obvious because they utilize the same disturbance model. The main difference is that the former uses the transfer function model while the latter uses a step response model. Using the transfer function model is definitely more advantageous as it reduces the number of parameters to work with.

Even though it is equivalent to the DMC, the structure of the CDGPC in the framework of the structure in Figure(1.5.1) has not been explored in existing literature. Nor has its Robustness been compared with that of the conventional GPC or the SPGPC.

1.6 Contributions and Organization of Thesis

The contributions of this thesis are four in number; three related to the field of the GPC (the N^* tuning method, PCS/2GPC, CDGPC), and one in the field of decentralized control (RNGA-based GPC for Decentralized Control).

The Introductions to each of these methods, the motivations behind them and the corresponding literature survey were presented in this chapter. Therefore, the remaining chapters will directly dwelve into the mathematics of the four methods, their simulation studies, experimental results and conclusions. The remaining chapters are titled,

- Chapter 2 - The N^* tuning method
- Chapter 3 & 4 - The Parallel Control Structure (PCS) & 2GPC resp.
- Chapter 5 - GPC with a Constant Disturbance Model (CDGPC)
- Chapter 6 - RNGA-based Decentralized GPC

Each chapter will begin with the details of how it is sectioned and will end with a summary of the chapter's results and contributions.

Chapter 2

The N^* tuning method

A novel tuning rule called the N^* method is proposed for the Single-Input Single-Output (SISO) Generalized Predictive Controller (GPC) for the control of stable and unstable First Order Plus Time Delay (FOPTD) processes. The method is based on studies of the nominal closed loop poles in correlation with the GPC parameters. The movement of the poles when the GPC parameters (λ , the move suppression coefficient and N , the prediction horizon) are varied generates a similar pattern for different FOPTD systems. So the general best location for the poles can be identified. Then, an analytical derivation arrives at the formula for the value of the tuning parameters needed to place the poles at the identified location. The analytical method makes use of the fact that when the control horizon N_u is fixed at 1, the closed loop characteristic equation can be made into a simple function of N , without any summation-to- N terms. This allows a study of pole locations for real values of N , which are denoted N^* . The derived tuning rule also incorporates a single normalized parameter k using which the performance and robustness of the loop can be traded-off easily and intuitively. Simulation results are used to demonstrate robust tuning using the proposed method for both stable and unstable FOPTD systems. Experiments are also used to compare the proposed tuning method with the method of Shridhar et.al. in terms of performance, robustness and the flexibility to balance them. The Chapter is organized as follows:

- Section(2.1) provides pre-requisite information on the unconstrained GPC for the general FOPTD process.
- Section (2.2) provides the derivation of the proposed tuning rule for stable and unstable FOPTD systems.
- Section (2.3) follows this up with a more in-depth analysis of the tuning rule, including analysis of how the k parameter functions.

- Section (2.4) examines closed loop simulation studies of the GPC loop, demonstrating Robust Tuning using the proposed tuning rule and comparing the results with an existing popular tuning strategy.
- Section (2.5) concludes with a summary and discussion of key points of this chapter.

2.1 Overview of the GPC for FOPTD Systems

2.1.1 FOPTD CARIMA model

Let $y_p(t)$ be the measured process output,

$$y_p(t) = y_m(t) + n(t) \quad (2.1.1)$$

where $y_m(t)$ is the model output and $n(t)$ represents the disturbance (the difference between the measured and model output) which is modeled as [10],

$$n(t) = \frac{C(z^{-1})\epsilon(t)}{D(z^{-1})}$$

where $\epsilon(t)$ is zero mean white noise, and C and D are polynomials chosen as $C(z^{-1}) = 1$ and $D(z^{-1}) = \Delta(1 - az^{-1})$ (with $\Delta = (1 - z^{-1})$).

The model output $y_m(t)$ is,

$$y_m(t) = g_m(z)u(t)$$

where $u(t)$ is the control input to the plant and $g_m(z)$ is the discrete-time equivalent of the FOPTD system,

$$\frac{K}{(\tau s + 1)}e^{-Ls}$$

where the delay time L is a *non-integral* multiple of the sampling time T . Therefore,

$$g_m(z) = \frac{bz^{-1}}{1 - az^{-1}}z^{-d}[(1 - \alpha) + \alpha z^{-1}] \equiv \frac{[b_0 + b_1 z^{-1}]z^{-1}}{1 - az^{-1}}z^{-d} \quad (2.1.2)$$

where,

$$a = e^{-\frac{T}{\tau}} \quad (2.1.3)$$

$$b = K(1 - a) \quad (2.1.4)$$

$$b_0 = (1 - \alpha)b \quad (2.1.5)$$

$$b_1 = \alpha b \quad (2.1.6)$$

$$d = \text{floor} \left(\frac{L}{T} \right) \quad (2.1.7)$$

$$d + \varepsilon = \frac{L}{T} \quad (2.1.8)$$

$$\alpha = \frac{a(a^{-\varepsilon} - 1)}{1 - a} \quad (2.1.9)$$

Substituting for $y_m(t)$ and $n(t)$ into Eq.(2.1.1), the FOPTD CARIMA (Controlled Auto-Regressive Integrated Moving Average) model is,

$$(1 - az^{-1}) y_p(t) = (b_0 + b_1 z^{-1}) z^{-d} u(t - 1) + \frac{\epsilon(t)}{\Delta} \quad (2.1.10)$$

2.1.2 Prediction Equation

Eq.(2.1.10) can be modified to,

$$y_p(t + 1) = (1 + a) y_p(t) - a y_p(t - 1) + (b_0 + b_1 z^{-1}) \Delta u(t - d) + \epsilon(t + 1) \quad (2.1.11)$$

Taking expectations on both sides of Eq.(2.1.11) removes the stochastic noise term and gives the best possible prediction of $y_p(t + 1)$, given all the information upto time t ,

$$\hat{y}_p(t + 1|t) = (1 + a) y_p(t) - a y_p(t - 1) + (b_0 + b_1 z^{-1}) \Delta u(t - d) \quad (2.1.12)$$

where the $\hat{}$ represents prediction. Then, the prediction of the output at a time $(t + j)$ can be obtained by recursively applying Eq.(2.1.12),

$$\begin{aligned} \hat{y}_p(t + j|t) = & S_{j+1} y_p(t) - (S_{j+1} - 1) y_p(t - 1) \\ & + (b_0 + b_1 z^{-1}) z^{-d} \left[S_j \quad S_{j-1} \quad \cdots \quad S_2 \quad S_1 \right]_{1 \times j} \begin{bmatrix} \Delta u(t) \\ \Delta u(t + 1) \\ \vdots \\ \Delta u(t + j - 2) \\ \Delta u(t + j - 1) \end{bmatrix}_{j \times 1} \end{aligned} \quad (2.1.13)$$

where,

$$S_j = \frac{(1 - a^j)}{(1 - a)} \quad (2.1.14)$$

The prediction equation in Eq.(2.1.13) is again modified by first time shifting as $t \leftarrow t+d$, and applying the control horizon restriction of $N_u = 1$,

$$\hat{y}_p(t+d+j|t) = S_{j+1}\hat{y}_p(t+d|t) - (S_{j+1} - 1)\hat{y}_p(t+d-1|t) + (b_0S_j + b_1S_{j-1})\Delta u(t) + (b_1S_j)\Delta u(t-1) \quad (2.1.15)$$

2.1.3 Prediction Vector

Taking j in Eq.(2.1.15) from 1 to N (that is, taking the prediction horizon from $N_1 = d+1$ to $N_2 = d+N$), the prediction vector is,

$$\begin{aligned} \hat{Y} &= \begin{bmatrix} \hat{y}_p(t+N_1|t) \\ \hat{y}_p(t+N_1+1|t) \\ \vdots \\ \hat{y}_p(t+N_2|t) \end{bmatrix} = \begin{bmatrix} \hat{y}_p(t+d+1|t) \\ \hat{y}_p(t+d+2|t) \\ \vdots \\ \hat{y}_p(t+d+N|t) \end{bmatrix} \\ \therefore \hat{Y} &= \underbrace{\begin{bmatrix} S_2 & -(S_2-1) \\ S_3 & -(S_3-1) \\ \vdots & \vdots \\ S_{N+1} & -(S_{N+1}-1) \end{bmatrix}}_{\overline{F}} \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \end{bmatrix}_{2 \times 1} \\ &\quad + \alpha \times b \underbrace{\begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{N-1} \\ S_N \end{bmatrix}}_{\overline{G}} \Delta u(t-1) + b \underbrace{\begin{bmatrix} (1-\alpha)S_1 + \alpha S_0 \\ (1-\alpha)S_2 + \alpha S_1 \\ \vdots \\ (1-\alpha)S_{N-1} + \alpha S_{N-2} \\ (1-\alpha)S_N + \alpha S_{N-1} \end{bmatrix}}_{\overline{G}} \Delta u(t) \end{aligned} \quad (2.1.16)$$

The matrices of the Prediction Vector Eq.(2.1.16) are denoted $\overline{F}$, $\overline{G}$ and $\overline{G}$, as indicated. Using $S_j = 1 + aS_{j-1}$, $\overline{G}$ can be expressed as,

$$\overline{G} = G - \alpha bX \quad (2.1.17)$$

where, $X = \begin{bmatrix} 1 & a & a^2 & \cdots & a^{N-1} \end{bmatrix}^T_{1 \times N}$.

2.1.4 Control Law

The proposed quadratic cost function,

$$J = \left(\widehat{Y} - W \right)^T \left(\widehat{Y} - W \right) + \lambda [\Delta u(t)]^2$$

where,

$$W = \underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{N \times 1} w(t) \equiv \mathbf{1} w(t)$$

is minimized to obtain the control law,

$$\Delta u(t) = \left(\overline{G}^T \overline{G} + \lambda I \right)^{-1} \overline{G}^T \left\{ W - \overline{F} \times \begin{bmatrix} \widehat{y}_p(t+d|t) \\ \widehat{y}_p(t+d-1|t) \end{bmatrix} - \alpha G \Delta u(t-1) \right\}$$

$$\Delta u(t) = k_1 w(t) - l_2 \begin{bmatrix} \widehat{y}_p(t+d|t) \\ \widehat{y}_p(t+d-1|t) \end{bmatrix} - l_3 \Delta u(t-1) \quad (2.1.18)$$

where,

$$[k_1]_{1 \times 1} = \frac{\overline{G}^T \mathbf{1}}{\overline{G}^T \overline{G} + \lambda} \quad (2.1.19)$$

$$[l_2]_{1 \times 2} = \frac{\overline{G}^T \overline{F}}{\overline{G}^T \overline{G} + \lambda} \quad (2.1.20)$$

$$[l_3]_{1 \times 1} = \frac{\alpha \overline{G}^T G}{\overline{G}^T \overline{G} + \lambda} \quad (2.1.21)$$

(Since $N_u = 1$, $\overline{G}^T \overline{G}$ is singleton). In transfer function form, the control law is,

$$\Delta u(t) = k_1 w(t) - L_2(z^{-1}) \widehat{y}_p(t+d|t) - l_3 \Delta u(t-1) \quad (2.1.22)$$

where $L_2(z^{-1}) = l_2^0 + l_2^1 z^{-1}$.

2.1.5 Optimal Predictor

Equivalently, the prediction equation in Eq.(2.1.13) can be written as,

$$\widehat{y}_p(t+j|t) = F_j(z^{-1}) y_p(t) + (b_0 + b_1 z^{-1}) E_j(z^{-1}) z^{-d} \Delta u(t+j-1) \quad (2.1.23)$$

where, F_j and E_j are related via the diophantine equation,

$$\frac{1}{\Delta(1 - az^{-1})} = E_j + z^{-j} \frac{F_j}{\Delta(1 - az^{-1})} \implies E_j = \frac{1 - z^{-j} F_j}{\Delta(1 - az^{-1})} \quad (2.1.24)$$

where,

$$E_j(z^{-1}) = S_1 + S_2 z^{-1} + \dots + S_{j-1} z^{-(j-2)} + S_j z^{-(j-1)} \quad (2.1.25)$$

$$F_j(z^{-1}) = S_{j+1} - (S_{j+1} - 1) z^{-1} \quad (2.1.26)$$

Substituting Eq.(2.1.24) into Eq.(2.1.23),

$$\hat{y}_p(t+j|t) = F_j(z^{-1}) (y_p(t) - y_m(t)) + g_m u(t+j) \quad (2.1.27)$$

The second term of the control law is,

$$L_2(z^{-1}) \hat{y}_p(t+d|t) = l_2^0 \hat{y}_p(t+d|t) + l_2^1 \hat{y}_p(t+d-1|t)$$

Substituting for the prediction terms using Eq.(2.1.27) gives,

$$\begin{aligned} L_2(z^{-1}) \hat{y}_p(t+d|t) &= l_2^0 [F_d(y_p(t) - y_m(t)) + g_m u(t+d)] + l_2^1 [F_{d-1}(y_p(t) - y_m(t)) + g_m u(t+d)] \\ &= (l_2^0 F_d + l_2^1 F_{d-1}) (y_p(t) - y_m(t)) + (l_2^0 + l_2^1 z^{-1}) g_m^* u(t) \\ &= L_2(z^{-1}) \left[g_m^* u(t) + \frac{l_2^0 F_d + l_2^1 F_{d-1}}{L_2(z^{-1})} (y_p(t) - y_m(t)) \right] \end{aligned}$$

where $g_m^* = \frac{bz^{-1}}{1-az^{-1}}$.

$$\therefore \hat{y}_p(t+d|t) = g_m^* u(t) + F_r(z^{-1}) (y_p(t) - y_m(t)) \quad (2.1.28)$$

with the filter F_r ,

$$F_r(z^{-1}) = \left[\frac{l_2^0 F_d + l_2^1 F_{d-1}}{L_2(z^{-1})} \right]$$

where F_d and F_{d-1} can be obtained from Eq.(2.1.26).

2.1.6 Closed Loop Transfer Function & Nominal Characteristic Equation

Substituting Eq.(2.1.28) into the control law in Eq.(2.1.22),

$$\Delta u(t) = k_1 w(t) - L_2(z^{-1}) [g_m^* u(t) + F_r(z^{-1}) (y_p(t) - y_m(t))] - l_3 \Delta u(t-1)$$

$$\left[\Delta (1 + l_3 z^{-1}) + L_2 g_m^* - L_2 g_m F_r \right] u(t) = k_1 w(t) - L_2 F_r y_p(t) \quad (2.1.29)$$

Let the actual process output be governed by,

$$y_p(t) = g_p u(t) + n_o(t) \quad (2.1.30)$$

where,

$$g_p(z) = \frac{(b_{0o} + b_{1o} z^{-1}) z^{-1}}{1 - a_o z^{-1}} z^{-d_o}$$

is the unknown actual process transfer function and $n_o(t)$ represents the actual disturbance process. Substituting for $u(t)$ from Eq.(2.1.29) into Eq.(2.1.12), the closed loop transfer functions are,

$$y_p(t) = P \times T^{p \neq m} \times w(t) + (1 - T^{p \neq m}) \times n_o(t) \quad (2.1.31)$$

where,

$$T^{p \neq m} = \frac{g_p L_2 F_r}{\Delta (1 + l_3 z^{-1}) + L_2 g_m^* + L_2 F_r (g_p - g_m)} \quad (2.1.32)$$

$$P = \left(\frac{k_1}{L_2 F_r} \right) \quad (2.1.33)$$

Under Nominal conditions, the closed loop transfer functions are,

$$y_p(t) = P \times T^{p=m} \times w(t) + (1 - T^{p=m}) \times n_o(t) \quad (2.1.34)$$

where,

$$T^{p=m} = \frac{g_m L_2 F_r}{\Delta (1 + l_3 z^{-1}) + L_2 g_m^*} \quad (2.1.35)$$

The general characteristic equation is thus,

$$\Delta (1 + l_3 z^{-1}) + L_2 g_m^* + L_2 F_r (g_p - g_m) = 0$$

which under nominal conditions is,

$$\Delta (1 + l_3 z^{-1}) + L_2 g_m^* = 0 \quad (2.1.36)$$

2.1.7 Poles of the Characteristic Equation

The matrix $\bar{F}$ (Eq.(2.1.16)) is,

$$\bar{F} = \begin{bmatrix} S_2 & -(S_2 - 1) \\ S_3 & -(S_3 - 1) \\ \vdots & \vdots \\ S_{N+1} & -(S_{N+1} - 1) \end{bmatrix}$$

Using $S_j = 1 + aS_{j-1}$,

$$\begin{aligned} \bar{F} &= \begin{bmatrix} 1 + aS_1 & -aS_1 \\ 1 + aS_2 & -aS_2 \\ \vdots & \vdots \\ 1 + aS_N & -aS_N \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \end{bmatrix} + a \begin{bmatrix} S_1 & -S_1 \\ S_2 & -S_2 \\ \vdots & \vdots \\ S_N & -S_N \end{bmatrix} \\ \bar{F} &= \begin{bmatrix} \mathbf{1} & \mathbf{0} \end{bmatrix} + \frac{a}{b} \begin{bmatrix} G & -G \end{bmatrix} \end{aligned}$$

Multiplying on both sides by $\frac{\bar{G}^T}{\bar{G}^T \bar{G} + \lambda}$,

$$l_2 = \begin{bmatrix} k_1 & \mathbf{0} \end{bmatrix} + \frac{a}{b} \begin{bmatrix} \frac{\bar{G}^T G}{\bar{G}^T \bar{G} + \lambda} & -\frac{\bar{G}^T G}{\bar{G}^T \bar{G} + \lambda} \end{bmatrix} = \begin{bmatrix} k_1 & \mathbf{0} \end{bmatrix} + \left(\frac{al_3}{b\alpha} \right) \begin{bmatrix} 1 & -1 \end{bmatrix} \quad (2.1.37)$$

where k_1 is taken from Eq.(2.1.19). In transfer function form, Eq.(2.1.37) can be written as,

$$L_2(z^{-1}) = k_1 + \Delta \left(\frac{al_3}{b\alpha} \right) \quad (2.1.38)$$

Substituting $L_2(z^{-1})$ into the characteristic equation in Eq.(2.1.36),

$$\begin{aligned} \Delta (1 + l_3 z^{-1}) + \left(k_1 + \Delta \left(\frac{al_3}{b\alpha} \right) \right) g_m^* &= 0 \\ \Delta (1 + l_3 z^{-1}) (1 - az^{-1}) + \left(k_1 + \Delta \left(\frac{al_3}{b\alpha} \right) \right) b [(1 - \alpha) + \alpha z^{-1}] z^{-1} &= 0 \\ \implies r^2 z^{-2} + (-r^2 + k_1 b - 1) z^{-1} + 1 &= 0 \end{aligned} \quad (2.1.39)$$

where,

$$r = \sqrt{a \left(1 - \frac{l_3}{\alpha} \right) + \alpha k_1 b + (a - 1) l_3} \quad (2.1.40)$$

2.2 Study of Pole Movements

The development of the proposed tuning rule will be described with the help of simulations of the example systems,

$$\frac{1}{0.1738s + 1} e^{-1.03s} \quad (2.2.1)$$

$$\frac{1}{-0.1738s + 1} e^{-1.03s} \quad (2.2.2)$$

whose discrete-time equivalents, sampled at 0.05sec are,

$$\frac{0.1087 + 0.1413z^{-1}}{1 - 0.75z^{-1}} z^{-21} \quad (2.2.3)$$

$$\frac{-(0.122 + 0.2114z^{-1})}{1 - 1.3333z^{-1}} z^{-21} \quad (2.2.4)$$

2.2.1 Pole Movement, varying N with constant λ

The roots of the characteristic equation $z_{1,2}$ are clearly functions of the parameters N and λ (through k_1 and l_3). The objective is to develop a method to find good values for these two tuning parameters, based on a study of pole locations.

2.2.1.1 Stable System:

Figure(2.2.1) shows the movement of the closed loop nominal pole locations for the example stable system (Eq.(2.2.3)) when varying N from 1 to 16 with a constant $\lambda = 4.5$. The location of the pole of the system $z = a$ is also indicated. This general pattern is exhibited for any $a < 1$, b , d , α of the FOPTD system and for any given λ : the poles start out real and distinct to the right of $z = a$, become complex and then converge to real, distinct roots once more to the left of $z = a$. Let N_l denote the value of N at which the poles are complex such that at $N = N_l + 1$ the poles are real and distinct, to the left of $z = a$. Similarly, at $N = N_r$, the poles are complex such that at $N = N_r - 1$, the poles are real and distinct and to the right of $z = a$.

Figure 2.2.1: Stable Case: varying N (from 1 to 16) with fixed $\lambda = 4.5$

The step response of a second order discrete-time system with complex poles $z_{1,2} = x \pm jy$ ($y > 0$), $|z_{1,2}| < 1$ and unit gain is given by,

$$y[n] = u[n] \left\{ 1 - |z_1|^n \times \frac{\sqrt{(1-x)^2 + y^2}}{y} \times \sin \left(\angle z_1 \times n + \tan^{-1} \left(\frac{y}{1-x} \right) \right) \right\}$$

The **magnitude** of the two complex poles of a system determines its settling time; larger the magnitude, longer the settling time. Similarly, the larger the **phase** of the pole z_1 , the greater the damped frequency of oscillations.

Among all the values of N that offer complex poles, the pole locations when $N = N_l$ have the lowest magnitude and hence the fastest response. When $N > N_l$, one of the resulting real-distinct poles is always slower than the poles from $N = N_l$. Hence, given a λ , the corresponding $N = N_l$ is declared as the best choice. With regard to the phase of the poles, in the case examined in Figure(2.2.1), $N = N_l$ gives the lowest phase among the complex choices and will thus result in the lowest damped frequency. However, in general, the phase of the poles from $N = N_l$ may not always be the lowest. But the closed loop settling time (and therefore the magnitude) must take precedence and so, this is acceptable.

2.2.1.2 Unstable System:

Figure(2.2.2) shows the movement of the closed loop nominal pole locations for the example unstable system (Eq.(2.2.4)) when varying N from 1 to 11 for a constant $\lambda = 9$.

Figure 2.2.2: Unstable Case: varying N (from 1 to 11) with fixed $\lambda = 9$

The pattern exhibited is similar to that of the stable FOPTD system (in Figure(2.2.1)) but is different in that it is centered around $z = 1$ instead of $z = a$. The poles start out real and distinct to the right of $z = 1$, become complex and then converge to real, distinct roots once more to the left of $z = 1$. And this general pattern is exhibited for any $a > 1$, b , d , α of the unstable FOPTD system and for any λ . Once more, N_l be the value of N at which the poles are complex such that at $N = N_l + 1$ the poles are real and distinct, to the left of $z = 1$. Similarly, at $N = N_r$, the poles are complex such that at $N = N_r - 1$, the poles are real and distinct, to the right of $z = 1$.

Considering stability and magnitude, for a given λ , $N = N_l$ offers the best nominal closed loop pole location. It has been found with other examples (not provided here) that it is possible for $N = N_l$ to give complex poles that lie outside of the unit circle. Therefore, in the case of the unstable system, it is necessary to use $N = N_l + 1$ (which results in two distinct real roots inside the unit circle) in order to avoid instability.

2.2.2 Finding N_l

The next objective is to find N_l for a given λ , without resorting to a manual search by solving the characteristic equation for every N . For this, firstly, the poles of the characteristic equation (2.1.39) will be expressed as a direct function of N without any summation-to- N terms. This then involves converting k_1 and l_3 into direct functions of N .

The following matrix multiplications are singleton and are summations-to- N terms

that are simplified into direct functions of N and the system parameters (a, b, α) .

$$G^T \mathbf{1} = b \sum_{i=1}^N S_i = \frac{b}{(1-a)} [N - aS_N] \equiv f_2(N) \quad (2.2.5)$$

$$X^T X = \sum_{i=1}^N (a^2)^{i-1} = \frac{1-a^{2N}}{1-a^2} \equiv f_4(N) \quad (2.2.6)$$

$$G^T X = b \sum_{i=1}^N a^{i-1} S_i = \frac{b}{(1-a)} [S_N - a f_4] \equiv f_3(N) \quad (2.2.7)$$

$$G^T G = b \sum_{i=1}^N S_i^2 = \frac{b^2}{(1-a)^2} [N - 2aS_N + a^2 f_4] \equiv f_1(N) \quad (2.2.8)$$

Then,

$$\overline{G}^T \overline{G} = (G - \alpha b X)^T (G - \alpha b X) = f_1 - 2\alpha b f_3 + \alpha^2 b^2 f_4 \equiv \overline{f}_1(N) \quad (2.2.9)$$

$$\overline{G}^T \mathbf{1} = f_2 - \alpha b S_N \equiv \overline{f}_2(N) \quad (2.2.10)$$

$$\overline{G}^T G = f_1 - \alpha b f_3 \equiv \overline{f}_3(N) \quad (2.2.11)$$

Thus, k_1, l_3 from Eqs.(2.1.19) and (2.1.21) respectively can be expressed without any summation-to- N terms as,

$$k_1 = \frac{\overline{f}_2(N)}{\overline{f}_1(N) + \lambda} \quad l_3 = \frac{\alpha \overline{f}_3(N)}{\overline{f}_1(N) + \lambda} \quad (2.2.12)$$

Substituting for k_1 and l_3 into the characteristic equation Eq.(2.1.39), the closed loop poles are now direct functions of only N and λ and the system parameters a, b and α . Now, this allows the pole locations to be examined for a 'continuous' $N > 0$.

2.2.2.1 Stable System:

The nominal pole locations of the example stable system (Eq.(2.2.3)) are shown again in Figure (2.2.3), but this time with N varying from 0.1 to 16, with the same constant $\lambda = 4.5$ as before. As shown in Figure (2.2.3), at particular values of the 'continuous' N , the poles become real and identical. Let N_r^* be the value of N at which this occurs, to the right of $z = a$. Similarly, the poles become real and identical again at $N = N_l^*$, to the left of $z = a$. If N_r^* and N_l^* are known, then N_l and N_r can be easily found using,

$$N_l = \text{floor}(N_l^*) \quad (2.2.13)$$

$$N_r = \text{ceiling}(N_r^*)$$

Figure 2.2.3: Stable Case: varying N (from 0.1 to 16) with fixed $\lambda = 4.5$

2.2.2.2 Unstable System:

For the unstable system too, the value of N_l can also be found by finding N_l^* first. The pole movement pattern of the example unstable system (Eq.(2.2.4)) for a 'continuous' $N > 0$ is shown in Figure (2.2.4). As before, at $N = N_l^*$, to the left of $z = 1$, and at $N = N_r^*$, to the right of $z = 1$, the closed loop nominal poles become real, identical. Again, N_l and N_r can be easily found using Eq.(2.2.13).

2.2.3 Finding N_l^*

The objective now is to find N_l^* (in order to find N_l). This is done by forcing the discriminant of the (quadratic) characteristic equation in Eq.(2.1.39) to be 0. That is,

$$4r^2 = (-r^2 + (k_1b - 1))^2$$

which has the following 4 solutions,

$$\begin{aligned} r &= -1 + \sqrt{k_1b} \\ r &= -1 - \sqrt{k_1b} \\ r &= 1 - \sqrt{k_1b} \\ r &= 1 + \sqrt{k_1b} \end{aligned}$$

Figure 2.2.4: Unstable Case: varying N (from 0.1 to 16) with constant $\lambda = 9$

That is, both the RHS and LHS of the 4 solution-equations are functions of N and each equation is satisfied for one or more particular values of N .

2.2.3.1 Stable System:

However, examining the four solutions showed that for the real, continuous $N > 0$, only one of the 4 solution equations, $r = 1 - \sqrt{k_1 b}$ is relevant for the stable system. To demonstrate this, continuing with the example system (Eq.(2.2.3)), plots of r & $1 - \sqrt{k_1 b}$ vs N are shown in Figure (2.2.5). Thus, solving,

$$r = \sqrt{a \left(1 - \frac{l_3}{\alpha}\right) + \alpha k_1 b + (a - 1) l_3} = 1 - \sqrt{k_1 b} \quad (2.2.14)$$

for N will give the required values $N = N_r^*$ and $N = N_l^*$ for real, identical poles. But solving Eq.(2.2.14) directly is cumbersome.

2.2.3.2 Unstable System:

For the unstable system, two of the four solution equations were found to be relevant. Plots of r , $1 - \sqrt{k_1 b}$ & $1 + \sqrt{k_1 b}$ vs N are shown in Figure (2.2.5). However, from inspection of the figure, it should be clear that, to obtain the required value $N = N_l^*$, it is again enough to solve Eq.(2.2.14). It should be noted that, in the unstable case, Eq.(2.2.14) has only that one solution, $N = N_l^*$; only solving $\sqrt{a - bc} = 1 + \sqrt{k_1 b}$ will provide $N = N_r^*$.

Figure 2.2.5: Stable Case: r & $1 - \sqrt{k_1 b}$ vs N

Figure 2.2.6: Unstable Case: r , $1 + \sqrt{k_1 b}$ & $1 - \sqrt{k_1 b}$ vs N

2.2.4 Varying λ

From Eq.(2.1.39), the position of the poles when they are real and identical (for a given $\lambda = \lambda^*$ and its corresponding $N = N_l^*$) is,

$$z_{1,2}/(\lambda = \lambda^*, N = N_l^*) = \frac{-(-r^2 + k_1 b - 1)}{2}/(\lambda^* N_l^*)$$

But under the condition of Eq.(2.2.14), this becomes,

$$z_{1,2}/(\lambda^* N_l^*) = 1 - \sqrt{k_1 b}/(\lambda^* N_l^*) = r/(\lambda^* N_l^*)$$

Thus, $r/(\lambda^* N_l^*)$ denotes the distance at which the two complex poles meet, to the left of $z = a$ (in the stable case) and to the left of $z = 1$ (in the unstable case), for a given $\lambda = \lambda^*$ and its corresponding $N = N_l^*$.

2.2.4.1 Stable System:

It was found that the left-side meeting point gets closer to $z = a$ for a higher λ^* ; this is demonstrated in Figure (2.2.7). Thus, for the stable system, it is possible to write,

$$r/(\lambda^* N_l^*) = k \times a \text{ where } 0 < k < 1 \quad (2.2.15)$$

where k denotes the fraction of the distance from 0 to a where the poles meet to become real and identical, for a given $\lambda = \lambda^*$ and corresponding $N = N_l^*$. (It is to be noted that the right-side meeting point also moves closer to $z = a$ but in much shorter steps and is therefore, not visible in the figure). Ofcourse, for different $\lambda = \lambda^*$, the two roots will meet for a different $N = N_l^*$.

2.2.4.2 Unstable System:

In the unstable case, the left-side meeting point also gets closer to $z = 1$ for a higher λ^* ; this is shown in Figure(2.2.8). Thus,

$$r/(\lambda^* N_l^*) = k \times 1 \text{ where } 0 < k < 1 \quad (2.2.16)$$

where k now denotes the fraction of the distance from 0 to 1 where the poles meet to become real and identical.

For ease of use and for unifying both cases, Eqs.(2.2.15) and (2.2.16) are combined together as,

$$r/(\lambda^* N_l^*) = k \times \gamma \quad (2.2.17)$$

Figure 2.2.7: Stable Case: pole movement with continuous N for $\lambda = 1, 3, 5$ Figure 2.2.8: Unstable Case: pole movement with continuous N for $\lambda = 1, 3, 5$

where,

$$\gamma = \begin{cases} a & \text{for } a < 1 \\ 1 & \text{for } a > 1 \end{cases} \quad (2.2.18)$$

2.2.5 Tuning rule

To reiterate, for a given $\lambda = \lambda^*$, there is a corresponding $N = N_l^*$ at which the roots become real and identical at a fraction k of the distance from $z = a$ or $z = 1$. Thus, while the method started out with the objective of finding the N_l^* corresponding to a $\lambda = \lambda^*$, the introduction of k modifies the objective: Assuming that a k is specified beforehand, the aim is now to find the $\lambda = \lambda^*$ and the corresponding $N = N_l^*$ that will place the poles at the fraction k of the distance from $z = a$ or $z = 1$. This is achieved by combining Eqs.(2.2.17) and (2.2.14), as follows:

- Substituting $r = 1 - \sqrt{k_1 b}$ into Eq.(2.2.15),

$$1 - \sqrt{k_1 b} / (\lambda^* N_l^*) = k\gamma$$

Substituting for k_1 from Eq.(2.2.12),

$$\begin{aligned} 1 - \sqrt{\frac{\bar{f}_2(N_l^*) b}{\bar{f}_1(N_l^*) + \lambda^*}} &= k\gamma \\ \implies \lambda^* &= \frac{\bar{f}_2(N_l^*) b}{(1 - k\gamma)^2} - \bar{f}_1(N_l^*) \end{aligned} \quad (2.2.19)$$

This Eq.(2.2.19) casts λ^* as a function of k and N_l^* .

- Substituting for k_1 and l_3 into Eq.(2.2.14),

$$\sqrt{a \left(1 - \frac{l_3}{\alpha}\right) + \alpha k_1 b + (a - 1) l_3} = 1 - \sqrt{k_1 b}$$

and simplifying,

$$\sqrt{a (\bar{f}_1(N) - \bar{f}_3(N) + \lambda) + \alpha b \bar{f}_2(N) + \alpha (a - 1) \bar{f}_3(N)} = \sqrt{\bar{f}_1(N) + \lambda} - \sqrt{\bar{f}_2(N) b}$$

At $\lambda = \lambda^*$ and corresponding $N = N_l^*$,

$$\sqrt{a (\bar{f}_1(N_l^*) - \bar{f}_3(N_l^*) + \lambda^*) + \alpha b \bar{f}_2(N_l^*) + \alpha (a - 1) \bar{f}_3(N_l^*)} = \sqrt{\bar{f}_1(N_l^*) + \lambda^*} - \sqrt{\bar{f}_2(N_l^*) b} \quad (2.2.20)$$

Substituting for λ^* from Eq.(2.2.19) and re-arranging,

$$\frac{a}{(1-k\gamma)^2} - \frac{a\bar{f}_3}{b\bar{f}_2} + \alpha + \frac{\alpha(a-1)\bar{f}_3}{b\bar{f}_2} = \frac{k^2\gamma^2}{(1-k\gamma)^2}$$

Rearranging further,

$$\frac{\bar{f}_3(N_l^*)}{\bar{f}_2(N_l^*)} = -\frac{b}{a+\alpha(1-a)} \left[\frac{(k^2\gamma^2-a)}{(1-k\gamma)^2} - \alpha \right] \quad (2.2.21)$$

- Using Eqs.(2.2.10) & (2.2.11), $\bar{f}_3/\bar{f}_2$ can be simplified to,

$$\frac{\bar{f}_3(N)}{\bar{f}_2(N)} = \frac{(f_1 - \alpha b f_3)}{(f_2 - \alpha b S_N)} = \frac{b}{(1-a)} \left\{ 1 + \frac{-aS_N + a(a+\alpha(1-a))f_4}{N - (a+\alpha(1-a))S_N} \right\} \quad (2.2.22)$$

- Equating the RHS' of both Eq.(2.2.21) and Eq.(2.2.22) (at N_l^*), and simplifying,

$$\left\{ \frac{-aS_{N_l^*} + a(a+\alpha(1-a))f_4}{N_l^* - (a+\alpha(1-a))S_{N_l^*}} \right\} = -\frac{(a-k\gamma)^2}{(1-k\gamma)^2(a+\alpha(1-a))}$$

Using,

$$-aS_N + a(a+\alpha(1-a))f_4 = \frac{a(1-a)}{(1+a)}S_{N_l^*} \left\{ -(1+aS_{N_l^*}) + \alpha(2-(1-a)S_{N_l^*}) \right\}$$

further simplification results in,

$$\begin{aligned} \frac{S_{N_l^*} \left\{ -(1+aS_{N_l^*}) + \alpha(2-(1-a)S_{N_l^*}) \right\}}{N_l^* - (a+\alpha(1-a))S_{N_l^*}} &= M \\ \Rightarrow N_l^* &= \left\{ \frac{(Ma+1)S_{N_l^*} + aS_{N_l^*}^2}{M} \right\} + \alpha \left\{ \frac{S_{N_l^*} [(1-a)(M+S_{N_l^*}) - 2]}{M} \right\} \end{aligned} \quad (2.2.23)$$

where,

$$M = \frac{(1+a)}{a(1-a)(a+(1-a)\alpha)} \times \frac{(a-k\gamma)^2}{(1-k\gamma)^2} \quad (2.2.24)$$

Eq.(2.2.23) is the final equation that can be used to solve for N_l^* using a computational search method such as the interval bisection method.

- Once N_l^* is found, λ^* can be calculated using Eq.(2.2.19). Thus, both N_l^* and λ^* have been parametrized in terms of k . Thus, each k corresponds to a value of N_l^* and λ^* each. Figures (2.2.9) and (2.2.10) show this relation for the example systems. It is recommended to use $N = N_l^*$ and $\lambda = \lambda^*$ to calculate the transfer functions

Figure 2.2.9: Stable Case: Varying k from 0.01 to 0.9

of the unconstrained GPC. (But when a discrete N is necessary, $N = \text{ceiling}(N_l^*)$ and $\lambda = \lambda^*$ is recommended. This will be investigated further in Section (2.3.6)).

2.3 Properties of the N^* method

2.3.1 Performance

Given $N = N_l^*$ and $\lambda = \lambda^*$ corresponding to some k , the transfer functions of the control law can be simplified. Using Eq.(2.2.19),

$$k_1/(\lambda^* N_l^*) = \frac{\bar{f}_2(N_l^*)}{\bar{f}_1(N_l^*) + \lambda^*} = \frac{(1 - \gamma k)^2}{b} \quad (2.3.1)$$

$$l_3/(\lambda^* N_l^*) = \frac{\alpha \bar{f}_3(N_l^*)}{\bar{f}_1(N_l^*) + \lambda^*} = \frac{-\alpha}{a + \alpha(1 - a)} \times [(k^2 \gamma^2 - a) - \alpha(1 - k\gamma)^2] \quad (2.3.2)$$

Figure 2.2.10: Unstable Case: Varying k from 0.01 to 0.9

The characteristic polynomial at λ^* and N_l^* is,

$$\Delta(1+l_3z^{-1})+L_2g_m^*/(\lambda^*N_l^*) = \Delta(1+l_3z^{-1}) + \left(k_1 + \Delta\left(\frac{al_3}{b\alpha}\right)\right)g_m^*$$

Using Eqs.(2.3.1) and (2.3.2), this can be simplified to,

$$\therefore \Delta(1+l_3z^{-1})+L_2g_m^*/(\lambda^*N_l^*) = \frac{(1-\gamma kz^{-1})^2}{(1-az^{-1})}$$

The tracking transfer function under nominal condition (from Eq.(2.1.34)) at $\lambda = \lambda^*$ and $N = N_l^*$ is,

$$\therefore [P(z) \times T^{p=m}(z)]/(\lambda^*N_l^*) = \frac{\frac{bz^{-1}z^{-d}}{(1-az^{-1})} \times \frac{(1-\gamma k)^2}{b}}{\frac{(1-\gamma kz^{-1})^2}{(1-az^{-1})}} = \left(\frac{(1-\gamma k)^2 z^{-1}}{(1-\gamma kz^{-1})^2} \right) z^{-d} \quad (2.3.3)$$

Eq.(2.3.3) indicates that the proposed method generates nominal closed loop poles that are real & identical and at a fraction k of the distance from $z = 0$ to $z = a$ for the stable system, and at a fraction k of the distance from $z = 0$ to $z = 1$ for the unstable system. In other words, a k closer to 1 will place the dual poles closer to γ (a for the stable case and 1 for the unstable case) and result in lower performance while a k closer to 0 will place the dual poles close to $z = 0$ and will provide greater nominal closed loop performance. The location of the nominal closed loop poles for the example systems, for different $k = 0.1, 0.25, 0.5, 0.75, 0.9$ is shown in Figure (2.3.1).

2.3.2 Nominal and Internal Stability

Eq.(2.3.3) also implies guaranteed nominal stability for both the stable and unstable cases. Internal stability, in the case of the unstable system can be verified as follows: If the GPC loop is cast in the form of the conventional structure, then the primary controller of the loop is given by,

$$G_c = \frac{L_2 F_r}{\Delta(1+l_3z^{-1}) + L_2 g_m^* (1-z^{-d} F_r)}$$

It might appear at first that the unstable pole of g_m^* might make it to the numerator and thus cause unstable pole-zero cancellation. However, it can be easily proved that $(1-az^{-1})$ is a factor of the terms $L_2(1-z^{-d}F_r)$ and thus, the unstable pole of g_m^* in the denominator of G_c is nullified and internal stability is guaranteed. The fact that the GPC structure can guarantee *internal* stability for open loop unstable systems is one of its well known and important features [52]. While this may be a given, the distinction must be

(a) Stable Case: Nominal closed loop poles for $k = 0.1, 0.25, 0.5, 0.75, 0.9$ (b) Unstable Case: Nominal closed loop poles for $k = 0.1, 0.25, 0.5, 0.75, 0.9$

Figure 2.3.1

made that nominal closed loop stability is achieved only when the GPC parameters are chosen properly. The proposed tuning method does this and thus guarantees nominal stability for first order unstable processes.

2.3.3 Robustness

Next, the effect of k on robustness is examined. For the purposes of robustness analysis, the actual process $g_p(z)$ is taken as $g_p(z) = g_m(z)(1 + \Delta^*(z))$ [52], where $\Delta^*(z)$ is the multiplicative uncertainty. Then, the sufficient condition for Robust Stability is given by the *small gain theorem*[43, 4],

$$|\Delta^*(e^{j\omega T})| < \frac{1}{|T^{p=m}(e^{j\omega T})|} = \Delta_{lim}^*(\omega) \quad \forall \left(0 \leq \omega \leq \frac{\pi}{T}\right) \quad (2.3.4)$$

where $T^{p=m}$ is the primary loop's tracking transfer function and $\Delta_{lim}^*(\omega)$ is the *Robustness Limit* curve; it defines the upper limit for the multiplicative uncertainty across all frequencies. Alternatively, the greater the function $\Delta_{lim}^*(\omega)$ across all frequencies, the greater the *robustness* of the closed loop. Substituting for $T^{p=m}$ from Eq.(2.3.3) and considering the transfer function at $N = N_l^*$ and $\lambda = \lambda^*$,

$$\begin{aligned} \Delta_{lim}^*(\omega)/(\lambda^* N_l^*) &= \frac{1}{|T^{p=m}(e^{j\omega T})|_{(\lambda^* N_l^*)}} = |P(e^{j\omega T})|_{(\lambda^* N_l^*)} \times \frac{|1 - \gamma k e^{-j\omega T}|^2}{(1 - \gamma k)^2} \quad (2.3.5) \\ &= |P(e^{j\omega T})|_{(\lambda^* N_l^*)} \left(1 + \frac{4\gamma k}{(1 - \gamma k)^2} \sin^2\left(\frac{\omega T}{2}\right)\right) \quad (2.3.6) \end{aligned}$$

This indicates that, at a particular frequency ω , when $k \rightarrow 0$, $\Delta_{lim}^*(\omega) \rightarrow |P(e^{j\omega T})|_{(\lambda^* N_l^*)}$ and thus, for a $k > 0$, $\Delta_{lim}^*(\omega) > |P(e^{j\omega T})|_{(\lambda^* N_l^*)}$ (because the second term in Eq.(2.3.6) is a function of ω which is always greater than 1 in the range $(0 \leq \omega \leq \frac{\pi}{T})$). It follows that Robustness is higher at a higher k value.

It is noted that $P(z)$ is also a function of k and it can be reduced to,

$$P(z)/(\lambda^* N_l^*) = \frac{(1 - a)(a + \alpha(1 - a))}{(1 - az^{-1})(a + \alpha(1 - a)) + \Delta a^d \left[- (a + \alpha(1 - a))^2 + \frac{a - k^2 \gamma^2}{(1 - \gamma k)^2} \right]}$$

Because the expression for $|P(e^{j\omega T})|$ will be quite large, simulation studies of the stable case were used to determine that it does not detract from the relationship between robustness and k . That is, even considering $|P(e^{j\omega T})|$, increasing k continues to increase the Robustness Limit $\Delta_{lim}^*(\omega)$ across all frequencies. For the unstable case, the corresponding $|P(e^{j\omega T})|$ does distort this behavior but, at high frequencies, where model uncertainty is

Figure 2.3.2: Stable Case: Performance vs Robustness while varying k from 0.1 to 0.9

maximum, the desired trend continues with an increase in $\Delta_{lim}^*(\omega)$ following an increase in k .

Figures (2.3.2) and (2.3.3) show the drop in nominal tracking performance and the increase in robustness upon increasing k from 0.1 to 0.9 in four steps, for the stable and unstable system respectively. The closed loop simulations shown are those of the example systems (Eqs.(2.2.3) and (2.2.4)). Thus, k is a single parameter that controls the performance (of both disturbance rejection and tracking) traded off with robustness, and is thus useful as a simple design parameter.

Finally, Eq.(2.3.6) indicates that through P (Eq.(2.1.33)), the optimal filter F_r influences robustness. Therefore, if a variant of the GPC, such as the Smith-Predictor based GPC (SPGPC)[48] whose $F_r = 1$, is tuned using the proposed tuning method, then better Robustness Limits can be achieved. In other words, for the same k , the SPGPC will provide better robustness than the conventional GPC. (This is not applicable to the unstable case because the loop will no longer possess internal stability when the structure of the F_r filter is changed [52]).

Figure 2.3.3: Unstable Case: Performance vs Robustness while varying k from 0.1 to 0.9

2.3.4 Sampling Time and k selection for stable FOPTD systems

Let the settling time of the output of the closed loop tracking transfer function in Eq.(2.3.3) be t_s^{CL} . That is, for a unit step setpoint, t_s^{CL} is the time it takes for the output to reach 99% of the setpoint. The speed of the unit step response of the transfer function in Eq.(2.3.3) will depend on how fast the term $(ak)^{n-d}$ decays to zero, where d is the discrete delay and n represents discrete-time. Therefore,

$$(ak)^{\frac{t_s^{CL}}{T}-d} = 0.01 \quad (2.3.7)$$

where $d = \text{floor}\left(\frac{L}{T}\right)$. Similarly, let the settling time of the stable FOPTD process be t_s^g , which is the time it takes to reach 99% of steady state for a step input. Once more, the speed of the response will depend on the speed of decay of a^{n-d} .

$$(a)^{\frac{t_s^g}{T}-d} = 0.01 \quad (2.3.8)$$

Equating Eqs.(2.3.7) and (2.3.8), and rearranging,

$$k = a^{c-1} \quad (2.3.9)$$

where $c = (t_s^{CL}-dT)/(t_s^g-dT)$ is the time ratio that numerically represents how much faster the closed loop is in comparison to the original FOPTD process. Substituting for a and solving for the sampling time T ,

$$k = e^{-\frac{T}{\tau}(c-1)} \implies T = -\frac{\tau}{(c-1)} \ln(k) \quad (2.3.10)$$

Thus, a design procedure for GPC can be as follows: A designer selects a desired value for $c \geq 1$ while k is fixed at 0.8 (chosen arbitrarily). Applying Eq.(2.3.10), a sampling time value is obtained which places the two closed loop poles at a^c (Eq.(2.3.9)). Figure (2.3.4) shows the closed loop tracking response and robustness limits for three c values; obviously, a higher c produces a faster response and a lower robustness limit curve. After a c and its corresponding T are determined and the loop is implemented, online tuning for trading off performance with robustness can be performed by adjusting k up or down. Or if the sampling time T is fixed, then the k required to get a desired c can be calculated using Eq.(2.3.10).

2.3.5 Dead-Beat and Pole Placement

When $k = 0$, Eq.(2.3.3) shows that the poles get placed at the origin. Also, the recommended $\lambda = \lambda^*$ in Eq.(2.2.19) reduces to 0. Thus, the $k = 0$ setting corresponds to

Figure 2.3.4: Stable Case: Closed loop responses and Robustness for desired $c = 2, 4, 8$

Dead-Beat control[40, 15].

While the proposed parameter tuning method may appear at first to be similar to the pole placement technique for the GPC as proposed in [14, 13], they differ in many ways. The GPC pole placement technique augments the polynomial $P(z^{-1})$ to the plant model and then, in order to place the closed loop poles at the zeros of $P(z^{-1})$, dead-beat parameter tuning is applied which in turn requires a high value for N_u . The proposed tuning approach uses only $N_u = 1$ and because no augmented model is utilized, no filtering is required. Finally, with the proposed approach, when a discrete $N = N_l + 1 = \text{ceiling}(N_l^*)$ is used, the poles will not be placed exactly as $k\gamma$. Thus, it remains chiefly a parameter tuning method more than a pole placement technique.

2.3.6 Differences between using N_l and N_l^*

The original purpose of finding N_l^* was to find N_l , a positive integer. But when the GPC is operated without constraints, since its transfer functions were reduced to simple functions of N (for the FOPTD case), a discrete N is not necessary. Thus, it was possible to continue using $N = N_l^*$ to calculate the transfer functions - k_1 and L_2 of the controller and F_r of the optimal predictor. However, $N = N_l + 1 = \text{ceil}(N_l^*)$ can still be used in place of $N = N_l^*$, but the responses will be slightly more conservative because the poles will now be real, distinct poles with the slower pole being slower than the dual poles from $N = N_l^*$. The poles from $N = \text{ceiling}(N_l^*)$ will also be stable as suggested by the pole location studies in earlier sections. The difference in nominal closed loop responses and Robustness Limits when using $N = N_l^*$ and when using $N = N_l + 1 = \text{ceiling}(N_l^*)$ are shown in Figures (2.3.5) and (2.3.6), for the stable and unstable case respectively. It is concluded that the differences are minor and thus, the proposed tuning rule can be applied in either mode.

2.4 Simulations and Experiments

2.4.1 Example 1 - stable system

Robust Tuning of the GPC is demonstrated using the proposed tuning method, for an example stable system with nominal model,

$$g_m(s) = \frac{1}{2s + 1} e^{-s}$$

Figure 2.3.5: Stable Case: Differences between using $N = N_l^*$ and $N = \text{ceiling}(N_l^*)$ for $k = 0.3, 0.5, 0.9$

Figure 2.3.6: Unstable Case: Differences between using $N = N_l^*$ and $N = \text{ceiling}(N_l^*)$ for $k = 0.3, 0.5, 0.9$

sampled at $T = 0.1s$. The actual plant is taken as the 20% worst-case mismatch[43],

$$g_p(s) = \frac{1.2}{1.6s + 1} e^{-1.2s}$$

The corresponding multiplicative uncertainty $\Delta^*(e^{j\omega T})$ and the Robustness Limits $\Delta_{lim}^*(\omega)$ achieved for $k = 0.79, 0.89, 0.93, 0.96$ are shown in Figure (2.4.1); in this example, $k = 0.89$ and above satisfies Robust Stability. Also shown are the nominal closed loop responses for the different k settings. The responses and Robustness Limits are compared with those obtained with the tuning method suggested by Shridhar et.al.[61] The method allows for the control horizon N_u to be a integer value from 1 to 6. (Also, in the method[61], the value of λ is dependant on N_u). So, four settings $N_u = 1, 2, 4, 6$ are shown in Figure (2.4.1). None of the settings satisfy Robust Stability and any further increase in N_u also does not help.

With a $k = 0.79$, the proposed tuning method can match the performance obtained with the method of Shridhar et.al., as shown in Figure (2.4.1), and if required, this can be improved further with lower k values. Thus, it is concluded that the performance and robustness obtained with the proposed tuning method is superior to the method of the Sridhar et.al. Importantly, while the method of Shridhar et.al. allows a degree of freedom in the choice of N_u , it is not apparent how the user may employ it to balance robustness and performance. Meanwhile, as demonstrated, the k parameter in the proposed tuning method streamlines the procedure to trade-off performance for robustness.

2.4.2 Example 2 - unstable system

Robust Tuning for an unstable system whose model is,

$$g_m(s) = \frac{1}{-2s + 1} e^{-1s}$$

is demonstrated. For purposes of analysis, the actual plant is assumed to differ from the model by 10% in the process parameters,

$$g_p(s) = \frac{1.1}{-1.6s + 1} e^{-1.1s}$$

The resulting multiplicative uncertainty $\Delta^*(e^{j\omega T})$ was calculated and its magnitude is shown in Figure (2.4.2) along with the Robustness Limits achieved with different k values. The corresponding closed loop nominal responses are also shown in the Figure (2.4.2). For this example, a value of k between 0.9 and 0.96 was found to be a good choice.

Figure 2.4.1: Stable Case: Comparing closed loop responses & Robustness Limits of the N^* and Shridhar-Cooper methods. (A setpoint of 1 is applied at $t = 0$ and an output disturbance of -1 is applied at $t = 15$)

Figure 2.4.2: Unstable Case: Comparing closed loop responses & Robustness Limits of the N^* and Shridhar-Cooper methods. (A setpoint of 1 is applied at $t = 0$ and an output disturbance of -1 is applied at $t = 30$)

Figure 2.4.3: The water tank testbed

2.4.3 Level Control in a Single Tank System

The GPC with proposed tuning was implemented on a Single Input Single Output water tank system for level control. The water tank system, shown in Figure (2.4.3) consists of a reservoir, a DC motor ($0 - 10V$) that pumps water into the top of the tank and a level sensor for measuring the height of water in the tank ($0 - 30cm$). The inlet and outlet at the top and bottom of the tank respectively can be adjusted using manual valves; these were left half-open.

Though such a system is non-linear, it was considered to be linear in the operating region of $13 - 20cm$ of water height. A step test was conducted in this region and the system was estimated using Matlab Identification Toolbox to be the FOPTD system:

$$\frac{3.4715e^{-0.3s}}{32.613s + 1} \quad (2.4.1)$$

The GPC was implemented (with a sampling time of $T = 0.1s$) on the water tank system. The system was brought to a steady state of $13.6cm$ by applying an input voltage of $6v$. Then, the loop was closed with a deviation setpoint of 0 . At a time $t = 30s$ after data logging starts, a deviation setpoint of $3cm$ was applied and at $t = 180s$, an input disturbance of $+4v$ was applied to the closed loop. The closed loop responses obtained using the Shridhar-Cooper tuning [61] and the proposed tuning method are shown in Figure (2.4.4).

As can be seen in Figure (2.4.4), the performance obtained with the N^* method is better for tracking performance than with the Shridhar-Cooper method. The reason why k values lower than 0.97 were not studied is because the system's input voltage saturates at $10V$. If it had been possible to use lower values of k , the disturbance rejection with the

Figure 2.4.4: Closed loop step responses of the level control experiment with sampling time $T = 0.1s$ (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)

Figure 2.4.5: Closed loop step responses of the level control experiment with sampling time $T = 0.3s$ (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)

N^* method could have matched those of the Shridhar-Cooper method in Figure (2.4.4). The most important point, however, is that changing the N_u value from 2 to 4 to 6 in the Sridhar-Cooper method does not allow significant variation in performance, and therefore, the method does not lend itself to online tuning for robustness vs performance as the N^* method does.

The experiments were repeated at a different sampling time $T = 0.3s$ whose results are shown in Figure(2.4.5). For the N^* method, Eq.(2.3.10) was used to determine the k values at $T = 0.3s$ that corresponded to the closed loop performances obtained with the k values at $T = 0.1s$ (see legend in Figure(2.4.4)). That is, the Eq.(2.3.10) offers a relation between k values at different sampling times that would keep the time ratio c constant:

$$\frac{\log(k_1)}{T_1} = \frac{\log(k_2)}{T_2} \implies k_2 = (k_1)^3$$

Thus, it can be verified that the performances of the three k settings in Figures (2.4.4) and (2.4.5) are identical, verifying that the N^* method can be used effectively at whatever sampling time. Meanwhile, the differences in performances obtained with the SC method at $T = 0.1s$ and $T = 0.3s$ are drastic. It was found that sampling times as high as $T = 1s$ gave better performances. Therefore, even though the shridhar-cooper method [61] allows the use of any sampling time, it is highly reliant on the proper choice of sampling time.

2.5 Chapter Conclusions and Future work

This chapter details a novel tuning method for the unconstrained GPC, for both stable and unstable FOPTD processes. Apart from an in-depth analysis of the GPC loop for unstable FOPTD process with respect to performance and robustness, another highlight of the proposed tuning method is the inclusion of a single normalized parameter k ($0 < k < 1$) in the tuning formula that allows the practitioner to balance performance and robustness.

The derivation of the tuning rule is based on studies of the nominal closed loop poles of the GPC loop as functions of the GPC parameters. Studies of the pole locations for a fixed λ and varying N revealed a general pattern from which the best pole location was identified to be from $N = N_l$. By simplifying the expression for the closed loop poles to simple functions in λ and N , it was found that, for a given $\lambda = \lambda^*$, the poles became real and identical for the setting $N = N_l^*$, a real value between the positive integers $N = N_l$ and $N = N_l + 1$. The location of the two real, identical poles lay at a fraction k of the distance from $z = 0$ to $z = a$ (stable case) or $z = 1$ (unstable case) and a higher λ^* placed the poles' location closer to a or 1 respectively. By expressing these observations mathematically, an analytical derivation parameterized λ^* and N_l^* in terms of k . Thus, the user specifies a k and the method suggests a corresponding λ^* and N_l^* . Because the pole movement patterns for the stable and unstable systems are different, the resulting tuning formula are slightly different. A summary of the proposed tuning rule is in Section(2.5.1).

Details of how the k parameter functions to influence performance and robustness was presented, along with a discussion on nominal and internal stability using the proposed parameter tuning method. Guidelines were presented for the selection of sampling time for the stable FOPTD. Simulations were used to demonstrate Robust Tuning using the proposed tuning rule for both stable and unstable systems. For the stable system, a comparison with the popular GPC tuning strategy proposed by Shirdhar-Cooper[61] was provided and the proposed method was found to be superior in terms of performance and robustness. The GPC was implemented as a level control system and the two tuning methods were compared: It was found the N^* method provided better tracking perfor-

mance than the Shridhar-Cooper method. The Shridhar-Cooper method allows a little flexibility in the choice of N_u but varying N_u in order to effect a change in performance met with poor results while with the N^* method, the k parameter could be used easily to adjust the performance. And while the Shridhar-Cooper method had drastic performance differences for different sampling times, it was demonstrated how the N^* method could have exactly the same performance at different sampling times by virtue of a relationship between k and T .

In practice, due to process uncertainties, a tuning rule would inevitably require some detuning in order to gain some robustness and stabilize the loop. And so, the focus of the work has been on providing an easy and intuitive way to balance robustness and performance in the GPC loop for FOPTD systems using just one parameter and this has been successfully achieved.

A disadvantage of the method for the stable FOPTD is that, currently, the N^* method does not allow a $k > 1$. That is, it won't be possible to make the closed loop slower than the original FOPTD process. This feature would be advantageous in situations where maximum robustness is required and the possibility of having this feature will be explored in the future. Another future work in this method will include extension to other other types of systems such as integrator systems of the forms $\frac{K}{s}$ and $\frac{K}{s(\tau s + 1)}$, and systems with a RHP zero.

2.5.1 Summary of the Tuning Procedure

The proposed tuning method for the general stable or unstable FOPTD system is summarized:

- Given a k , the equation,

$$N_l^* = \left\{ \frac{(Ma + 1) S_{N_l^*} + a S_{N_l^*}^2}{M} \right\} + \alpha \left\{ \frac{S_{N_l^*} [(1 - a)(M + S_{N_l^*}) - 2]}{M} \right\} \quad (2.5.1)$$

is solved for N_l^* , where,

$$M = \frac{(1 + a)}{a(1 - a)(a + (1 - a)\alpha)} \times \frac{(a - k\gamma)^2}{(1 - k\gamma)^2} \quad (2.5.2)$$

and S_j is given by Eq.(2.1.14).

- N_l^* is used to calculate λ^* ,

$$\lambda^* = \frac{\bar{f}_2(N_l^*)b}{(1 - k\gamma)^2} - \bar{f}_1(N_l^*) \quad (2.5.3)$$

where $\bar{f}_1(N)$ is given by Eq.(2.2.9).

- The GPC parameters are then: $N_u = 1$, $N_1 = 1 + N$, $N_2 = d + N$, where $N = \text{ceil}(N_l^*)$ and $\lambda = \lambda^*$.
- For the stable FOPTD, a suitable sampling time T can be calculated using,

$$T = -\frac{\tau}{(c-1)} \ln(k)$$

where the designer specifies a value for c which is the time ratio that represents how much faster the closed loop is in comparison to the original FOPTD process while k is fixed at 0.8 (chosen arbitrarily). Or if the sampling time T is fixed, then the k required to get a desired c can be calculated.

Chapter 3

The Parallel Control Structure (PCS)

The Parallel Control Structure (PCS) as an alternative to the Conventional Control Structure (CCS) is proposed to perform transparent online tuning - the ability to independently manipulate the three most important performance attributes of a control loop: *Set-point Tracking Performance*, *Disturbance-Rejection Performance* and *Robustness*. Firstly, the extent to which transparent online tuning is possible in the PCS is described and it is shown through derivations that the PCS maintains its transparent tuning feature even under mismatch conditions, provided that the Robust Stability condition is satisfied. Secondly, by utilizing its transparent tuning feature the PCS can have greater robustness than a Conventional Control Structure (CCS) with no loss in tracking performance. In addition, a generalized design procedure is presented which will direct the user towards proper utilization of the PCS' transparent tuning. Simulation and Experimental results are used to demonstrate the proposed design procedure and the transparent online tuning feature of the PCS. Arguments will be presented to highlight that the PCS' features are not available in a general Two Degree of Freedom (TDF) and One Degree of Freedom (ODF) structures. This chapter is sectioned as follows:

- Section (3.1) explains the basic properties of the PCS.
- Section (3.2) sketches a general design procedure that can properly utilize the transparent online tuning feature of the PCS.
- The main aim of this chapter is to explain *how* the PCS can be utilized for transparent online tuning and to what extent, and this is handled in Section (3.3). Most importantly, it will be proved that transparent online tuning (to the extent that is possible in the PCS) operates even under mismatch conditions. Finally, it will be shown that utilizing the transparent tuning feature of the PCS, it will be possible

to have greater robustness than a Conventional Control Structure (CCS).

- In Section (3.4), using simulations, the versatility of the PCS with respect to independent design and transparent online tuning will be demonstrated.
- In order to demonstrate in practice the transparent online tuning feature of the PCS (to the extent that is possible), it was implemented for the level control of a single tank process. Details and experimental results are provided in Section (3.4.5).
- Finally, in Section (3.5) we will present arguments that will make it clear that the 2 important features of the PCS (decoupled design and transparent online tuning) are not possible in general One Degree of Freedom (ODF) and Two Degree of Freedom (TDF) control structures.
- The chapter concludes in Section (3.6) by summarizing the important features of the PCS with respect to transparent online tuning.

3.1 The Basics of the PCS

(a) General Structure

(b) Modified PCS for systems with long time delays

Figure 3.1.1: The Parallel Control Structure

The Parallel Control Structure (PCS) is shown in Figure (3.1.1a). The PCS is a Two Degree of Freedom (TDF) control structure that incorporates the nominal model of the system, g_m . There are two controllers, g_{c1} and g_{c2} whose control actions u_1 and u_2 are added together before application to the plant, g_p . Assuming nominal conditions, the operation of the PCS is as follows:

1. Firstly, in a scenario where the setpoint $r \neq 0$, but no output disturbance ($d = 0$ in Figure (3.1.1a)) is present, $y = y_m$. Therefore, the signal $e = y_m - y = 0$, which means that $u_2 = 0$ always too. Of course, this essentially boils down to open loop control of the plant g_p , with only g_{c1} and g_m forming a 'virtual' control loop. Nevertheless, since we have assumed an ideal setting where no disturbance is present (and where $g_m = g_p$), the resulting setup is sufficient for the purpose of taking the output to the setpoint.
2. In the reverse case where the setpoint $r = 0$, but a disturbance $d \neq 0$ is present, u_1 and y_m are also always 0. That is, when $r = 0$, the structure in figure (3.1.1a) reduces to just a unity feedback loop with g_{c2} as the controller, controlling g_p , operating at zero setpoint.
3. It is the superpositional operation of the two cases, (1) $r \neq 0$ $d = 0$ (2) $r = 0$ $d \neq 0$, that is the PCS.

The closed loop transfer function of the Parallel Control Structure (PCS) in Figure (3.1.1a) is given by,

$$y = \frac{g_p(1 + g_{c2}g_m)}{g_m(1 + g_{c2}g_p)} \left[\frac{g_{c1}g_m}{1 + g_{c1}g_m} \right] r + \left[\frac{1}{1 + g_{c2}g_p} \right] d \equiv S_r^{(g_p \neq g_m)} r + S_d^{(g_p \neq g_m)} d \quad (3.1.1)$$

where all the signals and systems are in the laplace domain. The analysis in this article is confined to the general form of the PCS in Figure (3.1.1a). However, all results automatically apply to the alternate form in Figure (3.1.1b) which only differs in the fact that the nominal delay term e^{-Ls} (i.e. $g_m = g_m^* e^{-Ls}$) is seperated from the closed loop of $g_{c1}g_m$. (The majority of the literature referenced in Section(1.4) utilize this alternate form).

Two important properties of the PCS are listed below.

1. Under nominal conditions when $g_p \approx g_m$, Equation (3.1.1) can be reduced to,

$$y = \left[\frac{g_{c1}g_m}{1 + g_{c1}g_m} \right] r + \left[\frac{1}{1 + g_{c2}g_m} \right] d \equiv S_r^{(g_p = g_m)} r + S_d^{(g_p = g_m)} d \quad (3.1.2)$$

In this way, *the tracking response is **decoupled** from the rejection response*. The disturbance is rejected by the action of g_{c2} and the tracking of the set-point is taken

care of by g_{c1} . Hence, g_{c1} is the ***set-point tracking controller*** while g_{c2} is the ***disturbance-rejection controller***.

2. If we set the two controllers the same, $g_{c1} = g_{c2} = g_c$, then the Equation (3.1.1) reduces to,

$$y = \left[\frac{g_p g_c}{1 + g_c g_p} \right] r + \left[\frac{1}{1 + g_c g_p} \right] d \quad (3.1.3)$$

which is the same as the closed loop transfer function of a One Degree of Freedom (ODF) Conventional Control Structure (CCS) with g_c as the controller.

3.1.1 Prefilter Form

- Rearranging the terms in Eq.(3.1.1) we can write,

$$y = \left[\frac{g_{c2} g_p}{1 + g_{c2} g_p} \right] (pr) + \left[\frac{1}{1 + g_{c2} g_p} \right] d \quad (3.1.4)$$

which is nothing but a unity feedback loop with set-point weighting $p(s)$,

$$p(s) = \frac{g_{c1} (1 + g_{c2} g_m)}{g_{c2} (1 + g_{c1} g_m)} \quad (3.1.5)$$

Thus, the PCS can be re-drawn in the form of a general Two-Degree-Of-Freedom (TDF) control structure as shown in Figure (3.1.2).

- Some salient points regarding the pre-filter $p(s)$ in Eq.(3.1.5) are:
 1. Firstly, as shown in figure (3.1.2), the prefilter p depends on the model g_m and the feedback controller g_{c2} .
 2. The prefilter can be separated into two portions,

$$p(s) = \frac{g_{c1} (1 + g_{c2} g_m)}{g_{c2} (1 + g_{c1} g_m)} = \frac{(1 + g_{c2} g_m)}{g_m g_{c2}} \frac{g_m g_{c1}}{(1 + g_{c1} g_m)} = \left[1 - S_d^{(g_p \neq g_m)} \right]^{-1} S_r^{(g_p = g_m)}$$

Then, the closed loop transfer function in (3.1.4) can be written as,

$$y = \left[1 - S_d^{(g_p = g_m)} \right] \left[1 - S_d^{(g_p \neq g_m)} \right]^{-1} S_r^{(g_p = g_m)} r + S_d^{(g_p \neq g_m)} d \quad (3.1.6)$$

Under nominal conditions, $S_d^{(g_p \neq g_m)} = S_d^{(g_p = g_m)}$ and Eq. (3.1.6) will reduce to Eq.(3.1.2).

Figure 3.1.2: PCS Prefilter Form

3.1.2 Transparent Online Tuning on the PCS

If a control scheme has the facility for Transparent Online Tuning, it is meant that three parameters of the control scheme should directly and independently tune for the three most critical loop attributes - *Set-point Tracking Performance*, *Disturbance Rejection Performance* and *Robustness*. This ideal may not be full realizable.

As such, in the PCS, only 2 online tuning parameters, one from each controller, are available; the first can be used for online tuning of setpoint tracking alone while the second can be used to independently adjust the Disturbance Rejection Performance traded-off with the loop Robustness.

The entire procedure from the design of the controllers of the PCS to the choice of the 2 online tuning parameters is discussed in detail in Section (3.2).

Whatever the extent of *Transparent Online Tuning* possible on the PCS, it holds up even under mismatch conditions; this will be proved in Section (3.3).

3.2 PCS Design Procedure

For the purpose of clarity, we shall define:

- **(Controller) Design** as the choice of controller transfer function or the process of its analytical derivation,
- **(Offline) Tuning** as the preliminary choosing of values for the controller parameters using tuning methods
- **Online Tuning** as the adjustment of controller parameters after the closed loop is implemented and operational.

3.2.1 Design of Controllers and Asymptotic Constraints

- For the design of the controllers, we assume nominal conditions as is the normal practice with standard design methods like in IMC, Smith Predictor [54, 58] etc. Hence, $g_p = g_m$ is assumed and so Eq. (3.1.2) governs the design. Since $S_r^{(g_p=g_m)}$

Figure 3.2.1: A summary of the design procedure

and $S_d^{(g_p=g_m)}$ are independent of one another in terms of the controllers, the PCS decouples the set-point tracking response from the disturbance rejection response and allows for *independent design* for each [68].

- Controller designs for the PCS for different kinds of systems are described in [80, 79, 34] where, generally, both the controllers of the PCS are separately derived by proposing a desired transfer function for $S_r(g_p=g_m)$ and $S_d(g_p=g_m)$ (Direct Synthesis Method).
 - In particular, [79] studies how the PCS can be applied to FOPTD processes.
 - [80] studies integrator processes with delay, $\frac{K}{s}e^{-Ls}$.
 - [34] studies processes of the form, $\frac{K}{s^2(\tau s+1)}e^{-Ls}$, with double integrators.
- [67] also uses the *Direct Synthesis* approach to derive controllers for the modified PCS in Figure (3.1.1b) for the control of popular FOPTD and SOPTD Industrial

processes [2],

$$\left. \begin{aligned} g_1(s) &= \frac{K}{(\tau s + 1)} e^{-Ls} \\ g_2(s) &= \frac{K}{(\tau_1 s + 1)(\tau_2 s + 1)} e^{-Ls} \quad (\tau_1 \neq \tau_2) \\ g_3(s) &= \frac{K}{(\tau s + 1)^2} e^{-Ls} \\ g_4(s) &= \frac{K}{as^2 + bs + c} e^{-Ls} \quad (b^2 - 4ac) < 0 \end{aligned} \right\} \quad (3.2.1)$$

which are then approximated to the *practical PID* form.

- **Direct Synthesis design in PCS:** If g_m is a transfer function of the forms given in (3.2.1), we can propose a desired transfer function as $\frac{1}{\lambda_1 s + 1} e^{-Ls}$ (where L is the nominal system delay), for tracking response. Hence,

$$\frac{g_m g_{c1}}{1 + g_{c1} g_m} = \frac{1}{\lambda_1 s + 1} e^{-Ls} \quad (3.2.2)$$

and the controller g_{c1} is back calculated from the above equation. Similarly, we can propose another desired transfer function $1 - \frac{1}{\lambda_2 s + 1} e^{-Ls}$ for the disturbance rejection response. Hence,

$$\frac{1}{1 + g_{c2} g_m} = 1 - \frac{1}{\lambda_2 s + 1} e^{-Ls} \quad (3.2.3)$$

Once more, g_{c2} is back calculated from the above equation. The procedures for this back calculation are well established for different systems in [58] and other textbooks.

- **FOPTD processes with PCS:** For commonplace FOPTD industrial processes, instead of a direct synthesis design, the use of standard PI controllers for both g_{c1} and g_{c2} is preferable, with offline tuning done by any standard tuning technique. Thus, for the FOPTD system $g_1(s)$ in Eq.(3.2.1), g_{c1} and g_{c2} can be chosen as PI controllers,

$$g_{c1} = k_{p1} + \frac{k_{i1}}{s} = k_{p1} \left(1 + \frac{1}{\tau_{i1} s} \right) \quad (3.2.4)$$

$$g_{c2} = k_{p2} + \frac{k_{i2}}{s} = k_{p2} \left(1 + \frac{1}{\tau_{i2} s} \right) \quad (3.2.5)$$

and then any PID tuning method (such as the Gain Margin Phase Margin method described in Appendix (iv)) may be used to decide the controller parameters. Since nominal conditions were assumed, both g_{c1} and g_{c2} will be tuned for g_m .

- **Integrator Systems with PCS:** For integrator systems of the form,

$$\frac{K}{(\tau s + 1)s} e^{-Ls}$$

simple P controllers,

$$g_{c1} = \lambda_1 \quad (3.2.6)$$

$$g_{c2} = \lambda_2 \quad (3.2.7)$$

are effective.

- **Asymptotic Constraints:** The controllers must be able to satisfy the following asymptotic tracking constraints,

$$\lim_{s \rightarrow 0} S_r^{(g_p=g_m)} = 1 \quad (3.2.8)$$

$$\lim_{s \rightarrow 0} S_d^{(g_p=g_m)} = 0 \quad (3.2.9)$$

in order that there be no steady state errors. The PID controllers readily satisfy these two constraints for processes of the forms given by Eq.(3.2.1).

- What has not been established in the literature is that by virtue of its *independent design* property, the PCS can handle differing set-point and disturbance signal specifications by having different designs for the two controllers.

1. *Different controller design :*

- (a) **Same Controller Type, Different Parameters:** In the control of a system with a single controller in a CCS loop, there may be instances where a controller parameter set A gives good set-point tracking while another parameter set B gives good disturbance rejection. That is, *the different parameter sets for the same controller reflect the differing nature of the set-point and disturbance signals*. With the PCS, we would insert parameter set A into the first controller and parameter set B into the second, and thereby, simultaneously obtain the required tracking and load rejection performances. Such a scenario is explained with simulations in Section (3.4.1).
- (b) **Different Controller Type:** It might also be that one *type* of controller might be good for tracking while another *type* might be good for disturbance rejection, and we can employ both simultaneously in the PCS. To illustrate this situation, another simulation case study is given in Section (3.4.2).

2. **Same controller design:** When there are no different set-point and disturbance signal specifications, we would simply use the same design for both controllers, in the sense that their transfer functions will have the same structure and the

same offline tuned parameter values; their parameters can still be varied online independently.

3.2.2 Single Online Tuning Parameter

We choose a *single parameter* from the two controllers, which are to be used as online tuning parameters for adjusting *Tracking* and *Disturbance Rejection Performance*. The choice of the single parameter is either,

1. ***By virtue of the controller design:*** Using the Direct Synthesis Method for controller design, automatically gives λ_1 in Eq. (3.2.2) and λ_2 in Eq. (3.2.3) as the *single online tuning parameters* of the respective controllers.
2. Or we can simply choose the gains of the two controllers k_{p1} and k_{p2} to be the single online tuning parameters, since they directly reflect performance.

3.2.3 Bounds on the single online tuning parameters

Bounds on the *single online tuning parameters* are derived using Nominal Stability considerations [34], Robust Stability considerations [34] or Gain Margin Specifications (See Section (3.4) for examples). The reason for this is that once the PCS system is implemented, the *single online tuning parameters* will be varied online to adjust the three performance attributes, and so a range over which the parameters can be safely varied must be fixed beforehand. Some of the ways to obtain bounds are given below.

3.2.3.1 From Nominal Stability (Routh-Hurwitz Stability)[34]

When the modified version of the PCS in Figure (3.1.1b) is adopted, the tracking transfer function under nominal conditions will be,

$$\bar{S}_r^{(g_p=g_m)} = \frac{g_{c1}g_m^*}{1 + g_{c1}g_m^*}e^{-Ls}$$

where L is the nominal system delay and g_m^* is the delay-free portion of g_m . As the characteristic equation is delay-free, we can employ Routh-Hurwitz stability [32] criteria to derive an exact range for the tracking controller's *single online tuning parameter*.

3.2.3.2 From Robust Stability considerations[34]

A method to obtain bounds on the *single online tuning parameter* of the disturbance rejection controller is to use Robust Stability Considerations; this method was employed

in ref[34]. The bounds were not analytically derived but were approximate bounds based on simulation tests.

If this method is opted, it is recommended to use the *worst-case* multiplicative uncertainty for FOPTD systems (given in Appendix (iii)) along with the Robust Stability condition for the PCS (presented in the Section (3.3)) to determine the upper limits on the gain of the disturbance rejection controller. Since, this employs a worst-case scenario, the bounds obtained will naturally be conservative.

3.2.3.3 From Gain Margin Specifications

From Eq.(3.1.2) is clear that there are two open loop transfer functions, $g_{c1}g_m$ and $g_{c2}g_m$, corresponding to tracking and disturbance rejection. Let A_{m1} be the gain margin for the open loop $g_{c1}g_m$. Normally, A_{m1} can be chosen to be between 2 and 5. By definition,

$$\frac{1}{A_{m1}} = |g_{c1}(j\omega) g_m(j\omega)|$$

If $g_{c1} = k_{p1}\bar{g}_{c1}$, where $\bar{g}_{c1}$ is the normalized controller and k_{p1} is its gain, then we have,

$$k_{p1} = \frac{1}{A_{m1} |\bar{g}_{c1}(j\omega) g_m(j\omega)|}$$

which gives us a range over which the controller gain can be varied. Similarly, we can get a range for k_{p2} , the gain of the second controller. (That is, $g_{c2} = k_{p2}\bar{g}_{c2}$, where $\bar{g}_{c2}$ is the normalized controller).

3.3 Transparent Online Tuning

Under nominal conditions, from Eq.(3.1.2), $S_r^{(g_p=g_m)}$ and $S_d^{(g_p=g_m)}$ are the transfer functions that govern the tracking and disturbance rejection responses respectively. Therefore,

- varying g_{c1} online (i.e. changing k_{p1} - the *single online tuning parameter* of g_{c1}) will affect only the tracking response while,
- varying g_{c2} online (i.e. changing k_{p2} - the *single online tuning parameter* of g_{c2}) will affect only the disturbance rejection response.

Hence, under ideal conditions, transparent online tuning is directly established. It will be shown in this section that even under non-ideal (mismatch) conditions, this fact approximately holds true.

3.3.1 Effect of online tuning of g_{c2} and g_{c1} on Disturbance Rejection Response

First, the effect of online tuning of g_{c2} and g_{c1} on the disturbance rejection response under general conditions is examined, for which $S_d^{(g_p \neq g_m)}$ (in Eq. (3.1.1)) is the governing transfer function.

- Straightaway, it is clear that online tuning of g_{c1} (changing k_{p1}) will not affect the disturbance rejection response at all.
- Now, in order to reveal how disturbance rejection response is affected by online tuning of g_{c2} , the Robust Stability Condition for the PCS will be employed.

Lemma 1. (*Robust Stability Condition for the PCS*): Assume that g_p is any member of the family Π of plants[43],

$$\Pi = \left\{ g_p : \frac{|g_p(j\omega) - g_m(j\omega)|}{|g_m(j\omega)|} \leq \overline{\Delta}(\omega) \right\}$$

which have the same number of RHP poles. Every member of Π satisfies,

$$g_p = g_m(1 + \Delta) \quad (3.3.1)$$

where Δ is the Multiplicative Uncertainty corresponding to g_p with the bound on the allowed Δ being,

$$|\Delta(j\omega)| \leq \overline{\Delta}(\omega) \quad \forall \omega \quad (3.3.2)$$

Also assume that g_{c2} stabilizes g_m , ie. that $1 + g_{c2}g_m$ is stable. Then, the closed loop PCS system is robustly stable if and only if the nominal disturbance rejection transfer function $S_d^{(g_p = g_m)}$ satisfies the following bound

$$\left\| \left(1 - S_d^{(g_p = g_m)}(j\omega) \right) \Delta(j\omega) \right\|_{\infty} < 1 \quad (3.3.3)$$

Proof. The closed loop equation in (3.1.1) is re-written as,

$$y = \frac{g_p(1 + g_{c2}g_m)}{1 + g_{c2}g_p} \left(\frac{g_{c1}}{1 + g_{c1}g_m} r \right) + \frac{1}{1 + g_{c2}g_p} d$$

The $\frac{g_{c1}}{1 + g_{c1}g_m}$ portion is user controlled and is always stable. Hence, the characteristic polynomial that governs stability is $1 + g_{c2}g_p$. Assuming corresponding nominal stability, that is, that $1 + g_{c2}g_m$ is stable, and that the real plant g_p is some member of the family of plants Π , then, from geometric considerations[43] on the nyquist plot of $g_{c2}g_m$, the

robust stability condition is,

$$|1 + g_{c2}(j\omega) g_m(j\omega)| > |g_p(j\omega) g_c(j\omega) - g_m(j\omega) g_c(j\omega)|$$

Substituting for g_p from Eq.(3.3.1),

$$|1 + g_{c2}(j\omega) g_m(j\omega)| > |g_m(j\omega) g_c(j\omega)| |\Delta(j\omega)|$$

$$\therefore \left\| \frac{g_{c2}(j\omega) g_m(j\omega)}{1 + g_{c2}(j\omega) g_m(j\omega)} \Delta(j\omega) \right\|_{\infty} < 1$$

$$\implies \left\| \left(1 - S_d^{(g_p=g_m)}(j\omega) \right) \Delta(j\omega) \right\|_{\infty} < 1$$

□

According to Eq.(3.3.3), under general (mismatch) conditions, k_{p2} is bounded on top by Robust Stability bounds; the lower the gain, the further from the bound it is and the better the *Robustness*. At the same time, increasing k_{p2} will increase the open loop gain $|g_{c2}(j\omega) g_m(j\omega)|$ (which will then increase the peak of $\left| \frac{g_{c2}(j\omega) g_m(j\omega)}{1 + g_{c2}(j\omega) g_m(j\omega)} \right|$) and improve the **Nominal Disturbance Rejection Performance**. In this way, k_{p2} can be adjusted for a trade-off between *Nominal Disturbance Rejection Performance* and *Robustness*; increasing it increases the *Nominal Disturbance Rejection Performance* while decreasing the loop *Robustness*.

3.3.2 Effect of online tuning of g_{c2} and g_{c1} on Set-Point Tracking Response

Now, the effect of online tuning of g_{c2} and g_{c1} on tracking response under general conditions is investigated, for which $S_r^{(g_p \neq g_m)}$ (in Eq.(3.1.1)) is the governing transfer function.

- Firstly, increasing the controller gain k_{p1} , will increase the peak of $\left| \frac{g_{c1}(j\omega) g_m(j\omega)}{1 + g_{c1}(j\omega) g_m(j\omega)} \right|$ in $S_r^{(g_p \neq g_m)}$ and will provide better *Nominal Tracking Performance* and vice versa.
- Meanwhile, with the use of the concept of sensitivity, it will be explained how the online tuning of g_{c2} does not affect the tracking response much *in comparison* to the change that occurs in the disturbance rejection response.

Definition 2. (Sensitivity): In general, the sensitivity of a quantity Y to a change in X is defined as the ratio of the *percentage change in Y* due to a given *percentage change in X* . That is,

$$Sens_X^Y \triangleq \left(\frac{\Delta Y}{\Delta X} \right) \times \left(\frac{X}{Y} \right)$$

In the limiting case, $(\Delta X \rightarrow 0)$,

$$Sens_X^Y = \frac{dY}{dX} \times \frac{X}{Y} (\equiv Sens[Y \text{ to } X]) \quad (3.3.4)$$

We are interested in the ratio of these changes dY/dX as well as in the ratio of the relative changes. Where X and Y are transfer functions, $Sens_X^Y$ refers to the effect that a change in the transfer function X has on the other transfer function Y ; $Sens_X^Y$ may itself be a transfer function[58].

Theorem 3. ($S_r^{(g_p \neq g_m)}$ is less sensitive to changes in g_{c2} than $S_d^{(g_p \neq g_m)}$): Provided that the Robust Stability Condition for the PCS (in Lemma (1)) is satisfied (which includes the condition that g_{c2} stabilizes g_m), then,

$$Sens[S_r^{(g_p \neq g_m)} \text{ to } g_{c2}] \ll Sens[S_d^{(g_p \neq g_m)} \text{ to } g_{c2}] \quad \forall \text{ low } \omega \quad (3.3.5)$$

which implies that online tuning for the disturbance rejection response (by g_{c2}) does not affect the tracking response, even under mismatch conditions.

Proof. The sensitivity of $S_r^{(g_p \neq g_m)}$ to a change in g_{c2} is ,

$$\begin{aligned} Sens[S_r^{(g_p \neq g_m)} \text{ to } g_{c2}] &= \frac{dS_r^{(g_p \neq g_m)}}{dg_{c2}} \times \frac{g_{c2}}{S_r^{(g_p \neq g_m)}} \\ &= -\frac{1}{1 + g_{c2}g_p} \times \left(\Delta \frac{g_{c2}g_m}{1 + g_{c2}g_m} \right) \\ &= -\frac{\alpha}{1 + g_{c2}g_p} \end{aligned} \quad (3.3.6)$$

where $\alpha = \Delta \frac{g_{c2}g_m}{1 + g_{c2}g_m}$. Similarly, the sensitivity of $S_d^{(g_p \neq g_m)}$ to a change in g_{c2} is,

$$\begin{aligned} Sens[S_d^{(g_p \neq g_m)} \text{ to } g_{c2}] &\triangleq \frac{dS_d^{(g_p \neq g_m)}}{dg_{c2}} \times \frac{g_{c2}}{S_d^{(g_p \neq g_m)}} \\ &= -\left(\frac{g_{c2}g_p}{1 + g_{c2}g_p} \right) \end{aligned} \quad (3.3.7)$$

The numerators of Eqs.(3.3.6) and (3.3.7) are compared.

Since α is nothing but $\left(1 - S_d^{(g_p = g_m)}(j\omega)\right) \Delta(j\omega)$, if the Robust Stability condition in Eq.(3.3.3) is satisfied, we have $|\alpha(j\omega)| < 1 \forall \omega$. Meanwhile, the numerator of Eq.(3.3.7), the open loop magnitude $|g_{c2}(j\omega)g_p(j\omega)|$ will normally be greater than 1 for low frequencies. (This is true when the open loop satisfies Eqs.(3.2.8) and (3.2.9) and therefore includes an integrator term. Even if there is no integrator term, $g_{c2}g_p = g_{c2}g_m(1 + \Delta)$

will have a greater magnitude than α at low frequencies). Therefore,

$$Sens [S_r^{(g_p \neq g_m)} \text{ to } g_{c2}] \ll Sens [S_d^{(g_p \neq g_m)} \text{ to } g_{c2}] \quad \forall \text{ low } \omega$$

Hence, online tuning for the disturbance rejection response (by adjusting g_{c2} in some way) does not affect the tracking response, even under mismatch conditions, provided that the Robust Stability Condition is satisfied. $\square$

Note that this does not mean that set-point response under nominal and mismatch conditions are approximately the same; only that tuning g_{c2} online (by adjusting k_{p2}) will not affect the tracking response much. This is demonstrated via simulations in Section (3.4).

3.3.3 Example

In order to further illustrate Theorem (3), a numerical example is used, where $g_m = \frac{1}{2s+1}e^{-s}$, $g_{c1} = \frac{\pi}{3} \left(1 + \frac{0.5}{s}\right)$ (obtained using the Gain Margin Phase Margin Method in Appendix), and g_p was taken as the 20% worst-case mismatch plant, $g_p = \frac{1.2}{1.6s+1}e^{-1.2s}$. And the multiplicative uncertainty Δ corresponding to this worst-case g_p was calculated using Eq. (3.3.1).

Figure (3.3.1) compares the Sensitivities of $S_r^{(g_p \neq g_m)}$ and $S_d^{(g_p \neq g_m)}$ to g_{c2} , and

- illustrates the fact of Eq.(3.3.5).
- Note that at certain high frequencies, Eq.(3.3.5) does not hold, as expected.

Figure 3.3.1: Comparing Sensitivities of $S_r^{(g_p \neq g_m)}$ and $S_d^{(g_p \neq g_m)}$ to g_{c2}

3.3.4 Greater Robustness

A chief consequence of Theorem (3) is that we can operate the PCS loop at a level of *Robustness* greater than that of a CCS loop, without sacrificing *Tracking Performance*.

Theorem 4. (*Superior Robustness of the PCS*): If we set $k_{p2} < k_{p1}$, then the PCS loop will have greater *Robustness* than a CCS loop while having the same *Tracking Performance* as the CCS loop.

Proof. Let initially $g_{c1} = g_{c2}$. Then, the PCS has the same *Tracking Performance* and *Robustness* as a CCS loop, as inferred from Eq.(3.1.3). Then, if k_{p2} is decreased so that $k_{p2} < k_{p1}$, by Theorem (3) (assuming that the Robust Stability Condition in Eq.(3.3.3) is satisfied), the *Tracking Performance* is unchanged while disturbance rejection response is slowed down. But, by Eq.(3.3.3), decreasing k_{p2} decreases the *Nominal Disturbance Rejection Performance* and increases the *Robustness*. $\square$

3.3.5 Summary

To summarize, with the PCS, it is not possible to achieve the ideal of *total* transparent online tuning, whereby, 3 parameters of the controller can independently tune for the three most critical loop attributes. As such, the PCS has only two parameters, one for adjusting *Set-Point Tracking Performance* and the other for trading off *Nominal Disturbance Rejection Performance* vs loop *Robustness*. Section (3.5) will discuss alone how even this is not achieved with general One Degree of Freedom (ODF) and Two Degree of Freedom (TDF) structures. The important point to note is that the PCS maintains this extent of transparent online tuning even under mismatch conditions.

3.4 Simulations and Experiments

3.4.1 Example 1. Same Controller Type, Different Parameters

The nominal model of the system to be controlled with a PID loop is $g_m(s) = \frac{1}{s^2+2s+1}e^{-s}$. Using the Gain Margin & Phase Margin (GM-PM) tuning formula (given in the Appendix iv) with g_m yields the parameter set $A : [K_p \ K_i \ K_d] = [1.0472 \ 0.5263 \ 0.5263]$ for the single PID controller. But if there is noise in the loop, it would not be recommended to use the differential term of the controller which would result in actuator chattering. So, an alternative would be to approximate the model, using the half-rule [64], to $\tilde{g}_m = \frac{1}{1.5s+1}e^{-1.5s}$. Then, using $\tilde{g}_m$ with the GM-PM tuning rule, we get another PID parameter set $B : [K_p \ K_i \ K_d] = [0.5263 \ 0.3491 \ 0]$, where the differential term is zero.

However, it can be verified that parameter set A gives considerably better tracking performance than parameter set B . But obviously because of noise in the loop, design B should be opted, in the case of a CCS loop. In the PCS, one would simply insert parameter set A into the first PID controller g_{c1} and parameter set B into the second PID controller g_{c2} , and thereby, simultaneously obtain excellent tracking performance, while having no differential action on the noise; the simulation in Figure (3.4.1) demonstrates this.

3.4.2 Example 2. Different Controllers

This example will illustrate how, due to different nature of set-point and disturbance signals, we may have conflicting controller designs and how this can be overcome with the PCS. The model of the system to be controlled with a PID loop is $g_m(s) = \frac{1}{2s+1}e^{-s}$. The disturbances encountered are ramp signals, while the setpoint signals are regular step signals.

If the fact that disturbances are ramp signals is ignored, then a regular PI controller tuned using the GM-PM tuning method $g_{c1}(s) = \frac{\pi}{4} \left(1 + \frac{1}{2s}\right)$, would satisfy the tracking requirements. But if disturbances are to be rejected, an extra double integrator term must be included with the regular PI controller as, $g_{c2}(s) = g_{c1}(s) + \frac{0.04}{s^2}$ which will be necessary to asymptotically reject the ramp load.

If g_{c1} is used in a CCS loop, the ramp disturbance cannot be rejected, while if g_{c2} is used in a CCS loop, tracking would be bad (because of the double integration). It would be necessary to find a compromise between the two, spending time trying to balance the tracking vs load rejection trade-off. Using g_{c1} and g_{c2} in a PCS, provides the best *Tracking Performance*, while simultaneously incorporating the double integration required to reject ramp disturbances. The corresponding simulations are shown in Figure (3.4.2).

Figure 3.4.1: Closed Loop Responses of the CCS with pid_A , CCS with pid_B & the PCS with pid_A & pid_B , when there is white noise in the measurement

Figure 3.4.2: Closed Loop Responses of the CCS with g_{c1} , CCS with g_{c2} & the PCS with g_{c1} & g_{c2} , when the disturbance is a ramp (applied at $t = 50$)

3.4.3 Example 3. Transparent Online Tuning - FOPTD system

3.4.3.1 Nominal Case

In order to validate Theorem(7), the experiment was repeated by introducing model mismatch on purpose. The model used for controller tuning was taken as,

$$\frac{1.7357}{40.7662s + 1}e^{-0.25s}$$

A simulation example of the PCS control of an example FOPTD (First Order Plus Time Delay) system,

$$g_p(s) = g_m(s) = \frac{1}{2s + 1}e^{-s} \quad (3.4.1)$$

is presented here. The two controllers of the PCS are designed as PI controllers,

$$g_{c1} = K_{p1} + \frac{K_{i1}}{s} \quad \text{and} \quad g_{c2} = K_{p2} + \frac{K_{i2}}{s}$$

where, by use of the GM-PM tuning formula ,

$$\begin{bmatrix} K_{p1} \\ K_{i1} \end{bmatrix} = \frac{\pi}{2A_{m1} \times 1 \times 1} \begin{bmatrix} 2 \\ 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} K_{p2} \\ K_{i2} \end{bmatrix} = \frac{\pi}{2A_{m2} \times 1 \times 1} \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

where A_{m1} and A_{m2} are the gain margins confined to vary from 2 to 4; this bounds the gains of the controllers and hence A_{m1} and A_{m2} are chosen to be the *single online tuning parameters*.

Although both g_{c1} and g_{c2} are tuned for g_m , the controllers are different if we set $A_{m1} \neq A_{m2}$. If we set $A_{m1} = A_{m2}$, then $g_{c1} = g_{c2}$ and as mentioned in Section (3.1), the two-PI PCS control loop would behave as a single PI control loop. Figure (3.4.3) shows the responses of the two-PI PCS control loop and demonstrates what happens when A_{m1} and A_{m2} are varied over their respective ranges; varying A_{m1} affects only the *Tracking Performance* while varying A_{m2} affects only the *Disturbance Rejection Performance*.

3.4.3.2 Mismatch case

The responses of the previous example were simulated in the presence of a 20% worst case plant-model mismatch [43],

$$g_m(s) = \frac{1}{2s+1}e^{-s} \neq g_p(s) = \frac{1.2}{1.6s+1}e^{-1.2s} \quad (3.4.2)$$

Figures (3.4.4) show the responses of the two-PI PCS control loop, when A_{m1} and A_{m2} are respectively varied. Obviously, varying A_{m1} does not affect the disturbance rejection response, but varying A_{m2} affects both the disturbance rejection and set-point tracking responses, but the effect on the latter is minimal, as explained in Section (3.3). It should be noted that the range of A_{m2} was shifted from 2-4 to 3-5 in order to have a greater robustness.

In conclusion, under nominal conditions, A_{m1} and A_{m2} respectively govern only the set-point tracking response and only the disturbance rejection response. Under mismatch conditions, varying A_{m1} affects only the set-point tracking response but varying A_{m2} will mainly affect the disturbance rejection. A_{m2} also balances a trade-off between *Nominal Disturbance Rejection Performance* and *Robustness*. That is, larger the value of A_{m2} (smaller the gain of g_{c2}) poorer the *Disturbance Rejection Performance*, but better the *Robustness* to mismatch and vice versa.

Figure 3.4.3: PCS closed loop responses under nominal conditions with (a) varying A_{m1} ($A_{m2} = 3$) (b) varying A_{m2} ($A_{m1} = 3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 10$)

Figure 3.4.4: PCS closed loop responses under 20% worst-case mismatch with (a) varying A_{m1} ($A_{m2} = 3$) (b) varying A_{m2} ($A_{m1} = 3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 15$)

3.4.4 Example 4. Transparent Online Tuning - Integrator Systems

3.4.4.1 Nominal Case

A simulation example of the PCS control of an integrator system,

$$g_p(s) = g_m(s) = \frac{1}{s(2s+1)}e^{-s} \quad (3.4.3)$$

is presented. The two controllers of the PCS are designed as P controllers,

$$g_{c1} = \lambda_1 \quad \text{and} \quad g_{c2} = \lambda_2$$

which satisfy the asymptotic tracking constraints in Eqs. (2.3.8) & (2.3.7) and where, by nominal stability considerations,

$$0 < \lambda_1, \lambda_2 < 1$$

Figures (3.4.5) show the responses of the two-P PCS control loop and demonstrates what happens when λ_1 and λ_2 are varied over their respective ranges; varying λ_1 affects only the *Tracking Performance* while varying λ_2 affects only the *Disturbance Rejection Performance*. But λ_1 and λ_2 can only be varied over a certain range:

Nominal Stability Considerations:

For a system of the form $g_m = \frac{K}{s(\tau s + 1)}e^{-Ls}$, we can choose a controller $g_c = \lambda$. The Closed loop transfer functions are of the form,

$$y_p(s) = \frac{K\lambda}{s(\tau s + 1) + K\lambda e^{-Ls}}e^{-Ls}r + \frac{s(\tau s + 1)}{s(\tau s + 1) + K\lambda e^{-Ls}}d$$

Hence, the characteristic equation is,

$$s(\tau s + 1) + K\lambda e^{-Ls} = 0$$

Using the Taylor's approximation for the delay term,

$$\begin{aligned} s(\tau s + 1) + K\lambda(1 - Ls) &= 0 \\ \tau s^2 + (1 - K\lambda L)s + K\lambda &= 0 \end{aligned}$$

The roots of this equation are,

$$s = \frac{-(1 - K\lambda L) \pm \sqrt{(1 - K\lambda L)^2 - 4K\lambda\tau}}{2\tau}$$

Case $b^2 - 4ac < 0$: In this case, the condition for stability would be,

$$\begin{aligned} \text{Re}\{s\} &< 0 \\ \Rightarrow -\frac{1 - K\lambda L}{2\tau} &< 0 \\ \lambda &< \frac{1}{KL} \end{aligned}$$

Case $b^2 - 4ac > 0$: In this case, the condition for stability would be,

$$\begin{aligned} \frac{s}{- (1 - K\lambda L) \pm \sqrt{(1 - K\lambda L)^2 - 4K\lambda\tau}} &< 0 \\ \frac{- (1 - K\lambda L) \pm \sqrt{(1 - K\lambda L)^2 - 4K\lambda\tau}}{2\tau} &< 0 \\ - (1 - K\lambda L) &< \pm \sqrt{(1 - K\lambda L)^2 - 4K\lambda\tau} \\ \implies \lambda &> 0 \end{aligned}$$

Thus, the condition for stability is,

$$0 < \lambda < \frac{1}{KL}$$

3.4.4.2 Mismatch Case

The responses of the previous example were simulated in the presence of a 20% worst case plant-model mismatch [43],

$$g_m(s) = \frac{1}{s(2s+1)}e^{-s} \neq g_p(s) = \frac{1.2}{s(1.6s+1)}e^{-1.2s} \quad (3.4.4)$$

Figure (3.4.6) shows the responses of the two-PI PCS control loop, when λ_1 and λ_2 are respectively varied. Obviously, varying λ_1 does not affect the disturbance rejection response, but varying λ_2 affects mainly the disturbance rejection response.

In conclusion, under nominal conditions, λ_1 and λ_2 respectively govern only the set-point tracking response and only the disturbance rejection response. Under mismatch conditions, varying λ_1 affects only the set-point tracking response but varying λ_2 will mainly affect the disturbance rejection response. λ_2 also balances a trade-off between *Nominal Disturbance Rejection Performance* and *Robustness*.

Figure 3.4.5: PCS closed loop responses under nominal conditions with (a) varying λ_1 ($\lambda_2 = 0.3$) (b) varying λ_2 ($\lambda_1 = 0.3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 40$)

Figure 3.4.6: PCS closed loop responses under 20% worst-case mismatch with (a) varying λ_1 ($\lambda_2 = 0.3$) (b) varying λ_2 ($\lambda_1 = 0.3$) (setpoint of 1 is applied at $t = 0$ and output disturbance of -1 is injected at time $t = 40$)

3.4.5 Level Control in a Single Tank System

The PCS with PID was implemented for level control of a single tank system, shown in Figure (2.4.3) and the details regarding the system and its identification have already been discussed in Section(2.4.3).

The two controllers of the PCS were designed as PI controllers using the GM-PM tuning formula (See Appendix(iv)). However, the controllers were further detuned (using $A_{m1} = A_{m2} = 40$) in order that control actions would not saturate at the 10V limit of motor.

$$g_{c1} = g_{c2} = \left(1.2297 + \frac{0.0377}{s} \right)$$

The sampling time of the control loop was 0.005s.

The system was first brought to a steady state of 13.6cm by applying an input voltage of 6v. Then, the loop was closed with a deviation setpoint of 0. At a time $t = 30s$ after

data logging starts, a deviation setpoint of 3cm was applied and at $t = 180\text{s}$, an input disturbance of $+4\text{v}$ was applied to the closed loop. The closed loop responses obtained when varying A_{m1} up by 50% and down by 50% are shown in Figure (3.4.7); larger the A_m value the smaller the gain (See Appendix(iv)). Similarly, the closed loop responses obtained when varying A_{m2} are shown in Figure (3.4.8). The experimental results in Figures (3.4.7) and (3.4.8) demonstrate that the transparent online tuning is viable in practice with the use of the PCS; the disturbance rejection response and the tracking response can be adjusted independently and effectively.

Figure 3.4.7: 2PID Experimental Results: Varying the gain of g_{c1} (via A_{m1}) varies the *Tracking Performance* alone ($A_{m2} = 40$) (A setpoint of 3cm was applied at $t = 30\text{s}$ and an input disturbance of $+4\text{v}$ was applied at $t = 180\text{s}$)

Figure 3.4.8: 2PID Experimental Results: Varying the gain of g_{c2} (via A_{m2}) varies only the *Disturbance Rejection Performance* alone ($A_{m1} = 40$) (A setpoint of 3cm was applied at $t = 30\text{s}$ and an input disturbance of $+4\text{v}$ was applied at $t = 180\text{s}$)

In order to validate Theorem(3), the experiment was repeated by introducing model mismatch on purpose. The model used for controller tuning was taken as,

$$\frac{1.7357}{40.7662s + 1} e^{-0.25s}$$

so that the actual process (Eq.(2.4.1)) was the 20% worst-case mismatch. The closed loop responses with $A_{m1} = 60$ and varying A_{m2} are shown in Figure (3.4.9). It can be observed that the variation in g_{c2} doesn't affect the tracking response much. (However, this does not mean that the responses are exactly the same as when conditions were nominal in Figure (3.4.8)).

Figure 3.4.9: 2PID Mismatch Results: Varying the gain of g_{c2} (via A_{m2}) varies *mainly* the *Disturbance Rejection Performance* ($A_{m1} = 60$) (A setpoint of 3cm was applied at $t = 30$ s and an input disturbance of +4v was applied at $t = 180$ s)

3.5 Discussion

Here we highlight the differences between the three control structures - a general One Degree of Freedom (ODF) structure (same as the CCS shown in Figure (1.4.1)), a general Two Degree of Freedom (TDF) structure (shown in Figure (3.5.1)) and the PCS, with respect to *design* and *transparent online tuning*.

- The closed loop transfer function of a One Degree Of Freedom (ODF) Control loop is,

$$y = \frac{g_c g_p}{1 + g_c g_p} r + \frac{1}{1 + g_c g_p} d \quad (3.5.1)$$

It is obvious that we cannot design separately for tracking and disturbance rejection; a trade-off between tracking and disturbance rejection requirements must be made at the design stage. And when attempting online tuning, increasing the controller

gain will increase the *Nominal Performance* (of both tracking and disturbance rejection) at the cost of *Robustness* and vice versa; moreover, we cannot tune for the two responses independently.

- The closed loop transfer function of a general Two Degree Of Freedom (TDF) control loop is of the form,

$$y = \frac{g_p c_2 c_1}{1 + g_p c_2 c_3} r + \frac{1}{1 + g_p c_2 c_3} d \quad (3.5.2)$$

where c_1 , c_2 and c_3 are controller transfer functions. We may design c_2 and c_3 for disturbance rejection first, then select a c_1 so that we also obtain the required tracking response; hence, we can design for the two responses independently, but sequentially. In comparison, design is independent and separate in the PCS.

Online tuning of c_1 will affect the *Tracking Performance* only, while online tuning of either c_2 or c_3 online will affect performance of both responses and also the *Robustness*. In comparison, in the PCS, by virtue of Eq.(3.3.5), online tuning of g_{c2} affects only *Disturbance Rejection Performance* traded off with the loop *Robustness*.

Figure 3.5.1: The general Two Degree of Freedom Control Structure

3.6 Chapter Conclusion

The Parallel Control Structure (PCS) is proposed as an alternative to the Conventional Control Structure (CCS) to perform *transparent online tuning* - the ability of a control scheme to independently manipulate the three important attributes of the control loop (*Tracking Performance*, *Disturbance Rejection Performance* and *Robustness*).

A simple design procedure was sketched to guide the user towards proper utilization of the transparent tuning feature of the PCS; including the design of the two controllers and choice of *single online tuning parameters* and their bounds.

In conclusion, three special features unique to the PCS have been brought to light, alongside the already existing *independent design* feature:

1. **(Partial) Transparent Online Tuning:**

- (a) **Under Nominal Conditions**, online tuning of g_{c1} (changing k_{p1}) will let us adjust the *Tracking Performance* alone, while online tuning of g_{c2} (changing k_{p2}) will let us adjust the *Disturbance Rejection Performance* alone.
- (b) **Under Mismatch Conditions**, online tuning of g_{c1} (changing k_{p1}) will let us adjust the *Tracking Performance* alone, while online tuning of g_{c2} (changing k_{p2}) will let us trade-off *Nominal Disturbance Rejection Performance against Robustness* (Robust Stability).

2. *Online tuning for the disturbance rejection response (by changing g_{c2}) will not affect the tracking response even under mismatch conditions, provided the robust stability condition is satisfied.*

- 3. If we set $k_{p1} > k_{p2}$ then we can have greater robustness while having the same *Tracking Performance* as a CCS loop.

Arguments were presented to highlight that PCS is better than the general ODF and TDF structures in terms of its *independent design* and *transparent online tuning* features. Experimental results from the implementation of the PCS for level control in a single tank process were presented to demonstrate the transparent online tuning feature of the PCS.

Chapter 4

2GPC

The 2GPC algorithm is proposed as a variant of the conventional GPC. It consists of two GPCs working in tandem; one governs the set-point tracking response and the other controls the disturbance rejection response. However, the 2GPC is formulated as a single optimization problem integrating system constraints. The extent to which the 2GPC algorithm can perform Transparent Online Parameter Tuning (TOPT) is explained. (Transparent Online Parameter Tuning (TOPT) is the facility of the controller to allow the user to independently manipulate online the three most important loop performance attributes - *Set-point Tracking Performance*, *Disturbance Rejection Performance* and *Robustness* - with the use of three separate parameters). It is shown through derivations that the extent to which the 2GPC method can perform TOPT is maintained even under mismatch conditions. It is also proved that the 2GPC control loop, by utilization of its TOPT feature, can have greater *Robustness* than a conventional GPC loop. Simulation examples are used to demonstrate the TOPT and greater *Robustness* features of the 2GPC method. Experimental results are used to prove that the 2GPC and TOPT are viable in practise. Apart from presenting the 2GPC algorithm and its features, another objective of this work is to highlight the difficulty of offline parameter tuning for disturbance rejection in the conventional GPC algorithm and to show that the 2GPC provides an easy solution. A final objective is to contrast the 2GPC algorithm with already existing popular variants of GPC in a detailed manner.

The Chapter is sectioned as follows:

- Sections (4.1) and (4.2) provides the details of the derivation of the 2GPC algorithm including its cost function, constraints formulation, unconstrained explicit control law and unconstrained closed loop transfer function & its properties.
- Section (4.3) will detail the two features of the 2GPC, namely, Transparent Online Parameter Tuning (TOPT) and greater *Robustness* than a conventional GPC.

- Simulations with First Order Plus Time Delay (FOPTD) systems will be used to demonstrate these features in Section (4.4).
- In Section (3.4.5), experimental results from the 2GPC level control of a water tank system will be presented to prove the viability of the 2GPC and its features in practise.
- Apart from presenting the 2GPC and its particulars, a second objective is to contrast it with the existing variants of GPC and this discussion is done in Section (4.5). Another objective was to highlight the limitations of the conventional GPC with respect to offline parameter tuning and explain that they are overcome by utilization of the features of the 2GPC; this discussion is also done in Section (4.5).
- Finally, Section (4.6) will conclude with a summary of its contributions and a note on future directions.

4.1 The 2GPC Controller

4.1.1 Designing the 2GPC

Let $y_p(t)$ be the *real* output of the plant while $y_m(t)$ be the *expected* or model output of the plant, so that,

$$y_p(t) = g_p(z^{-1}) u(t) \quad (4.1.1)$$

$$y_m(t) = g_m(z^{-1}) u(t) \quad (4.1.2)$$

where,

$$g_p = \frac{B_0(z^{-1}) z^{-1}}{A_0(z^{-1})} z^{-d_0}$$

$$g_m = \frac{B(z^{-1}) z^{-1}}{A(z^{-1})} z^{-d} \quad (4.1.3)$$

Therefore,

$$A_0 y_p(t) = B_0 z^{-d_0} u(t-1) \quad (4.1.4)$$

$$A y_m(t) = B z^{-d} u(t-1) \quad (4.1.5)$$

Figure 4.1.1: The 2GPC Control Loop

Let, $u(t) = u_1(t) + u_2(t)$ such that $u(t)$ is applied to the actual plant g_p but only $u_1(t)$ is applied to the model $g_m(t)$. So,

$$A_0 y_p(t) = B_0 z^{-d_0} [u_1(t-1) + u_2(t-1)] \quad (4.1.6)$$

$$A y_m(t) = B z^{-d} u_1(t-1) \quad (4.1.7)$$

Subtracting Eq.(4.1.7) from Eq.(4.1.6),

$$A_0 y_p(t) - A y_m(t) = B_0 z^{-d_0} u_2(t-1) + [B_0 z^{-d_0} - B z^{-d}] u_1(t-1)$$

Assuming that the plant polynomials A_0 , B_0 and delay d_0 are approximately the same as the model polynomials A , B and delay d but defining $e(t) = y_p(t) - y_m(t)$ as the effect of external unmodelled disturbances,

$$A e(t) = B z^{-d} u_2(t-1) \quad (4.1.8)$$

A GPC is designed for this system with $e(t)$ as output and $u_2(t)$ as input in order to drive $e(t)$ to 0. When $e(t) = 0$, $y_p = y_m$, that is, the control system simply follows y_m . In order to make y_p follow the setpoint w , y_m should be at w and so, a separate GPC is designed for Eq.(4.1.7) to take y_m to w . The overall closed loop control system structure is shown in Figure (4.1.1).

At first, the virtual closed loop of y_m and GPC_1 might seem unnecessary but it is needed for two reasons: (1) the objective of this newly designed 2GPC controller is to make it very intuitive for the user to perform online tuning of the closed loop as compared to the online tuning of the regular GPC (with respect to tracking performance, disturbance rejection performance and robustness), and (2) the formulation of the overall controller into the well known quadratic optimization with constraints form is possible when this virtual loop is employed.

4.1.2 The 2 Cost Functions

For simplicity, the cost functions of GPC_1 and GPC_2 are taken to be of the same type. Furthermore, both costs are taken to be quadratic. These 2 assumptions will be required in order to be able to combine the independent costs into a single cost function. Future set-points are considered unknown and error weighting in the cost function is not considered.

The four parameters associated with the optimization formulation of GPC_1 are N_{a1} , N_{b1} , N_{u1} and λ_1 . The Cost Function of GPC_1 is,

$$J_1 = (Y_m - W)^T (Y_m - W) + U_1^T (\lambda_1 I) U_1 \quad (4.1.9)$$

where,

$$U_1 = \underbrace{\begin{bmatrix} \Delta u_1(t) \\ \Delta u_1(t+1) \\ \vdots \\ \Delta u_1(t+N_{u1}-1) \end{bmatrix}}_{N_{u1} \times 1}$$

$$W = \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{(N_{b1}-N_{a1}+1) \times 1} \quad w(t) = \mathbf{1}w(t)$$

and,

$$Y_m = \underbrace{\begin{bmatrix} y_m(t+d+N_{a1}) \\ y_m(t+d+N_{a1}+1) \\ \vdots \\ y_m(t+d+N_{b1}) \end{bmatrix}}_{(N_{b1}-N_{a1}+1) \times 1}$$

where the future terms in Y_m can be obtained using Eq.(4.1.5) after first multiplying it with Δ on both sides (for integral action).

Y_m can be written as,

$$Y_m = \underbrace{[G_1]}_{(N_{b1}-N_{a1}+1) \times N_{u1}} U_1 + \underbrace{[P_1]}_{(N_{b1}-N_{a1}+1) \times 1} \quad (4.1.10)$$

where,

$$P_1 = F_1 \underbrace{\begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix}}_{(n_a+1) \times 1} + I_1 \underbrace{\begin{bmatrix} \Delta u_1(t-1) \\ \Delta u_1(t-2) \\ \vdots \\ \Delta u_1(t-(n_b+d)) \end{bmatrix}}_{(n_b+d) \times 1} \quad (4.1.11)$$

Similarly, the four parameters associated with the optimization formulation of GPC_2 be N_{a2} , N_{b2} , N_{u2} and λ_2 . The Cost Function of GPC_2 is,

$$J_2 = E^T E + U_2^T (\lambda_2 I) U_2 \quad (4.1.12)$$

where,

$$U_2 = \underbrace{\begin{bmatrix} \Delta u_2(t) \\ \Delta u_2(t+1) \\ \vdots \\ \Delta u_2(t+N_{u2}-1) \end{bmatrix}}_{N_{u2} \times 1}$$

and,

$$E = \underbrace{\begin{bmatrix} e(t+d+N_{a2}) \\ e(t+d+N_{a2}+1) \\ \vdots \\ e(t+d+N_{b2}) \end{bmatrix}}_{(N_{b2}-N_{a2}+1) \times 1}$$

where the expectations of the future terms in E can be obtained using Eq.(4.1.8), after first multiplying it with Δ on both sides.

E can be written as,

$$E = \underbrace{[G_2]}_{(N_{b2}-N_{a2}+1) \times N_{u2}} U_2 + \underbrace{[P_2]}_{(N_{b2}-N_{a2}+1) \times 1} \quad (4.1.13)$$

where,

$$P_2 = F_2 \underbrace{\begin{bmatrix} e(t) \\ e(t-1) \\ \vdots \\ e(t-n_a) \end{bmatrix}}_{(n_a+1) \times 1} + I_2 \underbrace{\begin{bmatrix} \Delta u_2(t-1) \\ \Delta u_2(t-2) \\ \vdots \\ \Delta u_2(t-(n_b+d)) \end{bmatrix}}_{(n_b+d) \times 1} \quad (4.1.14)$$

4.1.3 Combining the Cost Functions

In order to integrate constraints into the 2GPC optimization, the separate quadratic costs of GPC_1 and GPC_2 are combined into a single optimization formulation. Since the cost function of GPC_1 (4.1.9) is only a function of U_1 and the cost function of GPC_2 (4.1.12) is only a function of U_2 , the overall optimization is posed as the sum of two separate optimizations.

Thus, the sum of Eqs.(4.1.9) & (4.1.12) is,

$$\bar{J} = J_1 + J_2 = \left[\begin{bmatrix} Y_m \\ E \end{bmatrix} - \begin{bmatrix} W \\ 0 \end{bmatrix} \right]^T \left[\begin{bmatrix} Y_m \\ E \end{bmatrix} - \begin{bmatrix} W \\ 0 \end{bmatrix} \right] + \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}^T \begin{bmatrix} \lambda_1 I & \mathbf{0} \\ \mathbf{0} & \lambda_2 I \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$$

Letting,

$$\bar{Y} = \underbrace{\begin{bmatrix} Y_{m(N_{b1}-N_{a1}+1) \times 1} \\ E_{(N_{b2}-N_{a2}+1) \times 1} \end{bmatrix}}_{(N_{b1}+N_{b2}-N_{a1}-N_{a2}+2) \times 1}$$

$$\bar{W} = \underbrace{\begin{bmatrix} W_{(N_{b1}-N_{a1}+1) \times 1} \\ \mathbf{0}_{(N_{b2}-N_{a2}+1) \times 1} \end{bmatrix}}_{(N_{b1}+N_{b2}-N_{a1}-N_{a2}+2) \times 1}$$

$$\bar{U} = \underbrace{\begin{bmatrix} U_1 \\ U_2 \end{bmatrix}}_{(N_{u1}+N_{u2}) \times 1}$$

and,

$$\lambda = \underbrace{\begin{bmatrix} \lambda_1 I & \mathbf{0} \\ \mathbf{0} & \lambda_2 I \end{bmatrix}}_{(N_{u1}+N_{u2}) \times (N_{u1}+N_{u2})}$$

Therefore, the overall cost function can be written as,

$$\bar{J} = (\bar{Y} - \bar{W})^T (\bar{Y} - \bar{W}) + \bar{U}^T \lambda \bar{U} \quad (4.1.15)$$

where $\bar{Y}$ can be expanded using Eqs.(4.1.10) & (4.1.13) as,

$$\begin{aligned} \bar{Y} = \begin{bmatrix} Y \\ E \end{bmatrix} &= \begin{bmatrix} G_1 U_1 + P_1 \\ G_2 U_2 + P_2 \end{bmatrix} = \underbrace{\begin{bmatrix} G_1 & \mathbf{0} \\ \mathbf{0} & G_2 \end{bmatrix}}_G \bar{U} + \underbrace{\begin{bmatrix} P_1 \\ P_2 \end{bmatrix}}_P \\ &\equiv G \bar{U} + P \end{aligned} \quad (4.1.16)$$

Further, using Eqs.(4.1.11) & (4.1.14), P can be written as,

$$\therefore P = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \underbrace{\begin{bmatrix} F_1 & \mathbf{0} \\ \mathbf{0} & F_2 \end{bmatrix}}_F \begin{bmatrix} y_m(t) \\ \vdots \\ y_m(t - n_a) \\ e(t) \\ \vdots \\ e(t - n_a) \end{bmatrix} + \underbrace{\begin{bmatrix} I_1 & \mathbf{0} \\ \mathbf{0} & I_2 \end{bmatrix}}_I \begin{bmatrix} \Delta u_1(t-1) \\ \vdots \\ \Delta u_1(t - (n_b + d)) \\ \Delta u_2(t-1) \\ \vdots \\ \Delta u_2(t - (n_b + d)) \end{bmatrix} \quad (4.1.17)$$

4.1.4 Combining Constraints

Normally, constraints associated with a system are specified in terms of $\Delta u(t+i)$, $i = 1 \cdots N_u$ (or are reduced to functions of $\Delta u(t+i)$). In the case of the 2GPC algorithm, $\Delta u(t+i)$ is the *total* change in control action, $\Delta u(t+i) = \Delta u_1(t+i) + \Delta u_2(t+i)$. Thus, for the 2GPC algorithm, all the constraints (slew rate, control levels, output constraints) can only be written in the form,

$$RU_{sum} \leq r \quad (4.1.18)$$

where,

$$U_{sum} = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t + \bar{N}_u - 1) \end{bmatrix}_{\bar{N}_u \times 1}$$

with,

$$\bar{N}_u = \max(N_{u1}, N_{u2}) \quad (4.1.19)$$

But the combined cost function $\bar{J}$ in Eq.(4.1.15) is a function of $\bar{U} = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$ and not U_{sum} . Hence, the task is to convert U_{sum} to $\bar{U}$. For instance, when $N_{u1} > N_{u2}$ ($\therefore \bar{N}_u = N_{u1}$),

$$U_{sum} = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t + \bar{N}_u - 1) \end{bmatrix} = \begin{bmatrix} \Delta u_1(t) \\ \vdots \\ \Delta u_1(t + N_{u1} - 1) \end{bmatrix}_{\bar{N}_u \times 1} + \begin{bmatrix} \Delta u_2(t) \\ \vdots \\ \Delta u_2(t + N_{u2} - 1) \\ \dots\dots\dots \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{\bar{N}_u \times 1} \quad (4.1.20)$$

Re-arranging it, after first multiplying it with Δ on both sides (for integral action).

$$U_{sum} = \begin{bmatrix} I_{\bar{N}_u \times \bar{N}_u} & I_{\bar{N}_u \times \bar{N}_u} \end{bmatrix} \begin{bmatrix} \Delta u_1(t) \\ \vdots \\ \Delta u_1(t + N_{u1} - 1) \\ \dots\dots\dots \\ \Delta u_2(t) \\ \vdots \\ \Delta u_2(t + N_{u2} - 1) \\ \dots\dots\dots \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{2\bar{N}_u \times 1}$$

Getting rid of the zeros at the end,

$$U_{sum} = \begin{bmatrix} I_{\bar{N}_u \times \bar{N}_u} & I_{\bar{N}_u \times N_{u2}} \end{bmatrix} \underbrace{\begin{bmatrix} \Delta u_1(t) \\ \vdots \\ \Delta u_1(t + N_{u1} - 1) \\ \dots\dots\dots \\ \Delta u_2(t) \\ \vdots \\ \Delta u_2(t + N_{u2} - 1) \end{bmatrix}}_{(N_{u1} + N_{u2}) \times 1}$$

which is compacted as,

$$U_{sum} = H\bar{U} \quad (4.1.21)$$

where $H = \begin{bmatrix} I_{\bar{N}_u \times \bar{N}_u} & I_{\bar{N}_u \times N_{u2}} \end{bmatrix}$ in this case. The same manipulation can be done for the case when $N_{u1} < N_{u2}$; it is direct in the case when $N_{u1} = N_{u2}$. In general, the matrix H is,

$$H = \begin{bmatrix} I_{\bar{N}_u \times N_{u1}} & I_{\bar{N}_u \times N_{u2}} \end{bmatrix} \quad (4.1.22)$$

Substituting U_{sum} in Eq.(4.1.21) into the constraints in Eq.(4.1.18),

$$RH \times \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} \leq r \implies RH \times \bar{U} \leq r \quad (4.1.23)$$

Thus, the constraints on U_{sum} have been brought down to constraints on $\bar{U}$.

4.1.5 Constrained Optimization

Now that the constraints are in terms of $\bar{U}$, together with the cost function in Eq.(4.1.15), we have a single QP problem:

$$\begin{aligned} \min_{\bar{U}} \quad & \bar{J} \\ \text{s.t.} \quad & RH\bar{U} \leq r \end{aligned}$$

At every iteration this optimization problem will be solved to obtain the optimal U . But, in accordance with the *Receding Horizon Principle*, only $\Delta u_1(t) + \Delta u_2(t)$ must be applied to the plant. Thus, the change in control action to be applied to the plant is,

$$\Delta u(t) = \Delta u_1(t) + \Delta u_2(t) = \left[\underbrace{1 \ 0 \ \dots \ 0}_{1 \times N_{u1}} \underbrace{1 \ 0 \ \dots \ 0}_{1 \times N_{u2}} \right] \times \bar{U} \quad (4.1.24)$$

In the following lemma, the special case when the parameters of the GPCs are made the same will be explored.

Lemma 5. *When the parameters of both GPC_1 and GPC_2 are set the same; that is, $N_{a1} = N_{a2}$, $N_{b1} = N_{b2}$, $N_{u1} = N_{u2}$ and $\lambda_1 = \lambda_2$, then the 2GPC loop will reduce to a conventional GPC loop.*

Proof. If the parameters of both GPC_1 and GPC_2 are made the same, $N_{a1} = N_{a2} = N_a$, $N_{b1} = N_{b2} = N_b$, $N_{u1} = N_{u2} = N_u$ and $\lambda_1 = \lambda_2 = \lambda$, their individual cost functions in Eqs.(4.1.9) & (4.1.12) can be written as,

$$\begin{aligned} J_1 &= (Y_m - W)^T (Y_m - W) + U_1^T (\lambda I) U_1 \\ J_2 &= (Y - Y_m)^T (Y - Y_m) + U_2^T (\lambda I) U_2 \end{aligned}$$

where because the parameters are the same, E in J_2 was expanded as $Y - Y_m$, where Y is the vector of output predictions,

$$Y = \underbrace{\begin{bmatrix} \hat{y}_p(t + d + N_a | t) \\ \hat{y}_p(t + d + N_a + 1 | t) \\ \vdots \\ \hat{y}_p(t + d + N_b | t) \end{bmatrix}}_{(N_2 - N_1 + 1) \times 1}$$

which is employed in the conventional GPC (See Appendix(v)). Also, U_1 and U_2 are both of the same dimensions, $N_u \times 1$. Adding J_1 and J_2 gives the overall cost function $\bar{J}$ of the 2GPC,

$$\bar{J} = J_1 + J_2 = J$$

where J is just the cost function of the conventional GPC.

As for constraints of the 2GPC in Eq.(4.1.23), when the parameters are the same, the product $H \times \bar{U} = U$ where U is the vector of future control actions of the conventional GPC, (See Appendix(v))

$$U = \underbrace{\begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-1) \end{bmatrix}}_{N_u \times 1}$$

Thus, the constraints of the 2GPC revert to the form of the conventional GPC constraints, $R \times U \leq r$. $\square$

4.2 Unconstrained 2GPC

In the unconstrained case, either $\min_{U_1} J_1$ and $\min_{U_2} J_2$ are separately solved and the resulting individual control actions $\Delta u_1(t)$ and $\Delta u_2(t)$ are added or the total cost J in Eq.(4.1.15) is directly minimized to directly obtain the total control action $\Delta u(t)$; both options yield the same result. Here, the latter method is employed to examine the control laws of GPC_1 and GPC_2 separately and then the total control law is studied.

The control laws of the unconstrained GPC_1 and GPC_2 are,

$$\Delta u_1(t) = c_1 w(t) - c_2 \begin{bmatrix} y_m(t) \\ \vdots \\ y_m(t-n_a) \end{bmatrix} - c_3 \begin{bmatrix} \Delta u_1(t-1) \\ \vdots \\ \Delta u_1(t-(n_b+d)) \end{bmatrix} \quad (4.2.1)$$

$$\Delta u_2(t) = d_1 \times 0 - d_2 \begin{bmatrix} e(t) \\ \vdots \\ e(t-n_a) \end{bmatrix} - d_3 \begin{bmatrix} \Delta u_2(t-1) \\ \vdots \\ \Delta u_2(t-(n_b+d)) \end{bmatrix} \quad (4.2.2)$$

It is noted that the d_1 coefficient in Eq.(4.2.2) is inconsequential, since the “set-point” to GPC_2 is 0.

Eqs.(4.2.1) & (4.2.2) are added to obtain the total control action is $\Delta u(t) = \Delta u_1(t) +$

$\Delta u_2(t)$, while substituting for $e(t)$ from its definition,

$$\begin{aligned} \Delta u(t) = & c_1 w(t) - (c_2 - d_2) \begin{bmatrix} y_m(t) \\ \vdots \\ y_m(t - n_a) \end{bmatrix} - d_2 \begin{bmatrix} y_p(t) \\ \vdots \\ y_p(t - n_a) \end{bmatrix} \\ & - d_3 \begin{bmatrix} \Delta u_1(t-1) \\ \vdots \\ \Delta u_1(t - (n_b + d)) \end{bmatrix} - d_3 \begin{bmatrix} \Delta u_2(t-1) \\ \vdots \\ \Delta u_2(t - (n_b + d)) \end{bmatrix} \end{aligned} \quad (4.2.3)$$

Eq.(4.2.3) is the unconstrained control law of the 2GPC algorithm.

4.2.1 Closed Loop Transfer Function

For the purpose of studying the closed loop transfer function, the plant output is taken as $y_p(t) = g_p(z^{-1})u(t) + d_o(t)$ where $d_o(t)$ is an external disturbance. The transfer function expression for $u(t)$ is then substituted into it:

$$y_p(t) = \frac{[(1 + D_3 z^{-1})\Delta + D_2 g_m]}{[(1 + D_3 z^{-1})\Delta + D_2 g_p]} \times \frac{c_1 g_p}{[(1 + C_3 z^{-1})\Delta + C_2 g_m]} \times r(t) + \frac{(1 + D_3 z^{-1})\Delta}{[(1 + D_3 z^{-1})\Delta + D_2 g_p]} d_o(t)$$

which is further rearranged to,

$$y_p(t) = S_r^{(g_p \neq g_m)}(z^{-1}) r(t) + S_d^{(g_p \neq g_m)}(z^{-1}) d_o(t) \quad (4.2.4)$$

where $S_r^{(g_p \neq g_m)}$ is the tracking transfer function and $S_d^{(g_p \neq g_m)}$ is the disturbance rejection transfer function,

$$S_r^{(g_p \neq g_m)} = \left(\frac{g_{c1}}{g_{c2}} \right) \times \frac{[1 + g_{c2} g_m]}{[1 + g_{c1} g_m]} \times \frac{g_{c2} g_p}{[1 + g_{c2} g_p]} \times \left(\frac{c_1}{C_2} \right) \quad (4.2.5)$$

$$S_d^{(g_p \neq g_m)} = \frac{1}{[1 + g_{c2} g_p]} \quad (4.2.6)$$

with,

$$g_{c1} = \frac{C_2}{(1 + C_3 z^{-1})\Delta} \quad (4.2.7)$$

$$g_{c2} = \frac{D_2}{(1 + D_3 z^{-1})\Delta} \quad (4.2.8)$$

where g_{c1} and g_{c2} are the primary controllers of the two GPC loops respectively. It should be noted that $C_2(z^{-1})$ and $C_3(z^{-1})$ are the transfer function forms of the vectors c_2 and c_3 respectively. Likewise, $D_2(z^{-1})$ and $D_3(z^{-1})$ are the transfer function versions of d_2

and d_3 respectively. (In some places, the (z^{-1}) has been omitted for convenience.)

4.2.2 Two Properties

The 2GPC algorithm has two special properties:

1. Under nominal conditions, $g_p = g_m$, the closed loop transfer function in Eq.(4.2.4) reduces to,

$$y_p(t) = S_r^{(g_p=g_m)}(z^{-1}) r(t) + S_d^{(g_p=g_m)}(z^{-1}) d_o(t) \quad (4.2.9)$$

with,

$$S_r^{(g_p=g_m)}(z^{-1}) = \frac{g_{c1}g_m}{[1 + g_{c1}g_m]} \times \left(\frac{c_1}{C_2} \right) \quad (4.2.10)$$

$$S_d^{(g_p=g_m)}(z^{-1}) = \frac{1}{1 + g_{c2}g_m} \quad (4.2.11)$$

In this way, *the tracking response is **decoupled** from the rejection response*. The disturbance is rejected by the action of g_{c2} and the tracking of the set-point is taken care of by g_{c1} . Hence, GPC_1 is the ***set-point tracking controller*** while GPC_2 is the ***disturbance-rejection controller***.

2. If we set the parameters of the 2GPC the same, that is, $N_{a1} = N_{a2}$, $N_{b1} = N_{b2}$, $N_{u1} = N_{u2}$ and $\lambda_1 = \lambda_2$, then the controller coefficients of both GPC_1 and GPC_2 will be the same; in Eqs.(4.2.1) & (4.2.2), $d_1 = c_1$, $d_2 = c_2$ and $d_3 = c_3$. Therefore, $g_{c1} = g_{c2}$ and Eq.(4.2.4) reduces to,

$$y_p(t) = \frac{g_{c1}g_p}{[1 + g_{c1}g_p]} \times \left(\frac{c_1}{C_2} \right) r(t) + \frac{1}{[1 + g_{c1}g_p]} d_o(t) \quad (4.2.12)$$

which is the same as the closed loop transfer function of a conventional *GPC*. This is a restatement of Lemma (5) for the unconstrained case.

4.3 Transparent Online Tuning, Robustness & Stability

Transparent Online Tuning is the facility whereby the controller has three parameters that can independently tune for the three most critical loop attributes - *Set-point Tracking Performance*, *Disturbance Rejection Performance* and *Robustness*. This ideal may not be fully realizable but this section explores the extent to which the 2GPC algorithm allows for ***Transparent Online Tuning***.

For this, firstly, a single tuning parameter from each controller is chosen to function as an online tuning parameter. Preferably, the chosen parameters should directly affect the performance of their respective controllers (preferably by reducing or increasing controller gain). For simple FOPTD (stable or unstable) processes, a good choice would be the control suppression co-efficients λ_1 and λ_2 from GPC_1 and GPC_2 respectively. While the 2GPC method can be applied to different kinds of systems and can employ any GPC tuning technique, **the following results are restricted to FOPTD systems (stable or unstable) and the N^* tuning technique.**

The newly proposed N^* GPC tuning method is applicable to FOPTD processes of the form,

$$\frac{K}{\tau s + 1} e^{-Ls}$$

whose discrete-time version with sampling time T is,

$$\frac{bz^{-1}}{1 - az^{-1}} z^{-d} [(1 - \alpha) + \alpha z^{-1}] \quad (4.3.1)$$

where,

$$\begin{aligned} a &= e^{-\frac{T}{\tau}} \\ b &= K(1 - a) \\ d &= \text{floor}\left(\frac{L}{T}\right) \\ \varepsilon &= \frac{L}{T} - d \\ \alpha &= \frac{a(a^{-\varepsilon} - 1)}{1 - a} \end{aligned}$$

The N^* tuning method reduces all the four tuning parameters of a GPC to just one normalized parameter k . A value of k closer to 0 will offer faster closed loop response while a k closer to 1 will offer slower performance but greater robustness. So, in the 2GPC case, k_1 of GPC_1 and k_2 of GPC_2 will be used as the single online tuning parameters.

The capability of the 2GPC method for Transparent Online Tuning can be easily examined for nominal conditions. From Eq.(4.2.9), $S_r^{(g_p=g_m)}$ and $S_d^{(g_p=g_m)}$ are the transfer functions that govern the tracking and disturbance rejection responses respectively. So, it is only straightforward that,

- varying GPC_1 online (i.e. changing k_1 - the *single online tuning parameter* of GPC_1) will affect only the tracking response while,
- varying GPC_2 online (i.e. changing k_2 - the *single online tuning parameter* of GPC_2) will affect only the disturbance rejection response.

Hence, under ideal conditions, **Transparent Online Tuning** is directly established. The remainder of the section will prove that this fact approximately holds true even under non-ideal (mismatch) conditions, and will also explain how k_2 also ties into *Robustness*.

4.3.1 Effect of online tuning of GPC_1 and GPC_2 on Disturbance Rejection Response

First, the effect of online tuning of GPC_1 and GPC_2 on the disturbance rejection response under general conditions is examined, for which $S_d^{(g_p \neq g_m)}$ (in Eq.(4.2.6)) is the governing transfer function. It is obvious that online tuning of GPC_1 (changing k_1) will not affect the disturbance rejection response at all. So, how the disturbance rejection response is affected by online tuning of GPC_2 is examined:

Assuming nominal stability (that is, $1 + g_c g_m$ is stable), the condition for the robust stability of the 2GPC loop is,

$$\left\| \left(1 - S_d^{(g_p = g_m)} (e^{j\omega T}) \right) \Delta^* (e^{j\omega T}) \right\|_{\infty} < 1 \quad (4.3.2)$$

$$\text{or} \quad \left\| \frac{g_{c2} (e^{j\omega T}) g_m (e^{j\omega T})}{1 + g_{c2} (e^{j\omega T}) g_m (e^{j\omega T})} \Delta^* (e^{j\omega T}) \right\|_{\infty} < 1 \quad (4.3.3)$$

where $\Delta^*(z)$ is the *Multiplicative Uncertainty* corresponding to $g_p(z)$,

$$g_p(z) = g_m(z) (1 + \Delta^*(z)) \quad (4.3.4)$$

with the bound on the allowed $\Delta^*(z)$ being,

$$|\Delta^*(e^{j\omega T})| \leq \overline{\Delta}^*(\omega) \quad (4.3.5)$$

The robust stability condition can also be written as,

$$\text{or} \quad |\Delta^*(e^{j\omega T})| < \frac{1}{\left| 1 - S_d^{(g_p = g_m)} (e^{j\omega T}) \right|} = \Delta_{lim}^*(\omega) \quad \forall \omega \in \left[0 \quad \frac{\pi}{T} \right] \quad (4.3.6)$$

where $\Delta_{lim}^*(\omega)$ is the Robustness Limit of the control of the control loop; the Robustness Limit should be larger than the magnitude of the multiplicative uncertainty at all frequencies for the Robust Stability condition to be satisfied.

With the N^* tuning method, at $N = N_l^*$ and $\lambda = \lambda^*$, the transfer function $1 -$

$S_d^{(g_p=g_m)}(e^{j\omega T})$ will reduce to,

$$1 - S_d^{(g_p=g_m)} = \frac{g_{c2}g_m}{1 + g_{c2}g_m} = \frac{(1 - \gamma k_2)^2 z^{-1} z^{-d}}{(1 - \gamma k_2 z^{-1})^2} \times \left(\frac{D_2}{d_1} \right)$$

where $\gamma = a$ if $a < 1$ and $\gamma = 1$ if $a > 1$. From this it is clear that a value of k_2 close to 0 will produce fast poles, resulting in a higher **Nominal Disturbance Rejection Performance**. The magnitude of $1 - S_d^{(g_p=g_m)}$ (at $N = N_l^*$ and $\lambda = \lambda^*$) is,

$$\left| 1 - S_d^{(g_p=g_m)} \right| = \frac{(1 - \gamma k_2)^2}{(1 - \gamma k_2 e^{-j\omega T})^2} \times \left| \frac{D_2}{d_1} \right| = \frac{(1 - \gamma k_2)^2}{(1 - \gamma k_2)^2 + 4\gamma k_2 \sin^2\left(\frac{\omega T}{2}\right)} \times \left| \frac{D_2}{d_1} \right|$$

When $k_2 \rightarrow 0$, at a fixed frequency ω , the first term is 1 and when $k_2 > 0$ the value of the first term is < 1 . Because the robustness limit of the loop, $\Delta_{lim}^*(\omega)$, is inversely proportional to $\left| 1 - S_d^{(g_p=g_m)} \right|$, this means that the Robustness of the loop is higher for a higher k_2 .

It is noted that although the factor D_2/d_1 is also a function of k_2 , it does not detract from the relationship between k_2 and Robustness or Nominal Disturbance Rejection Performance (See Section (2.3.3)). In this way, the online tuning of GPC_2 by adjusting k_2 can be used to balance the trade-off between *Nominal Disturbance Rejection Performance* and *Robustness*.

4.3.2 Effect of online tuning of GPC_1 and GPC_2 on Set-Point Tracking Response

The effect of online tuning of GPC_1 and GPC_2 on tracking response under general conditions is investigated, for which $S_r^{(g_p \neq g_m)}$ is the governing transfer function. $S_r^{(g_p \neq g_m)}$ in Eq.(4.2.4) can be written as,

$$S_r^{(g_p \neq g_m)} = \left[\left(\frac{c_1}{C_2} \right) \frac{g_{c2}g_m}{1 + g_{c1}g_m} \right] \times \left[\frac{1 + g_{c2}g_m}{g_{c2}g_m} \frac{g_{c2}g_p}{1 + g_{c2}g_p} \right]$$

which when the N^* tuning method is employed (at $N = N_l^*$ and $\lambda = \lambda^*$) is,

$$S_r^{(g_p \neq g_m)} = \frac{(1 - \gamma k_1)^2 z^{-1} z^{-d}}{(1 - \gamma k_1 z^{-1})^2} \times \left[\frac{1 + g_{c2}g_m}{g_{c2}g_m} \frac{g_{c2}g_p}{1 + g_{c2}g_p} \right]$$

It is obvious that a value of k_2 closer to 0 will result in faster poles which will improve the performance of the overall tracking transfer function $S_r^{(g_p \neq g_m)}$. But more importantly, with the concept of sensitivity, it will be shown that the online tuning of GPC_2 (by adjusting k_2) does not affect the tracking response much *in comparison* to the change

that occurs in the disturbance rejection response.

Definition 6. (Sensitivity): In general, the sensitivity of a quantity Y to a change in X is defined as the ratio of the *percentage change in Y* due to a given *percentage change in X* . That is,

$$Sens_X^Y \triangleq \left(\frac{\Delta Y}{\Delta X} \right) \times \left(\frac{X}{Y} \right)$$

In the limiting case, $(\Delta X \rightarrow 0)$,

$$Sens_X^Y = \frac{dY}{dX} \times \frac{X}{Y} (\equiv Sens[Y \text{ to } X]) \quad (4.3.7)$$

We are interested in the ratio of these changes dY/dX as well as in the ratio of the relative changes. Where X and Y are transfer functions, $Sens_X^Y$ refers to the effect that a change in the transfer function X has on the other transfer function Y ; $Sens_X^Y$ may itself be a transfer function[58].

Theorem 7. ($S_r^{(g_p \neq g_m)}$ is less sensitive to changes in g_{c2} than $S_d^{(g_p \neq g_m)}$): Provided that the Robust Stability Condition for the PCS is satisfied, then,

$$Sens[S_r^{(g_p \neq g_m)} \text{ to } g_{c2}] \ll Sens[S_d^{(g_p \neq g_m)} \text{ to } g_{c2}] \quad \forall \text{ low } \omega \quad (4.3.8)$$

which implies that online tuning for the disturbance rejection response (by adjusting g_{c2} in some way) does not affect the tracking response, even under mismatch conditions.

Proof. The sensitivity of $S_r^{(g_p \neq g_m)}$ to a change in g_{c2} is ,

$$\begin{aligned} Sens[S_r^{(g_p \neq g_m)} \text{ to } g_{c2}] &\triangleq \frac{dS_r^{(g_p \neq g_m)}}{d(g_{c2})} \times \frac{g_{c2}}{S_r^{(g_p \neq g_m)}} \\ &= -\frac{\alpha^*}{1 + g_{c2}g_p} \end{aligned} \quad (4.3.9)$$

where $\alpha^* = \Delta^* \frac{g_{c2}g_m}{1 + g_{c2}g_m}$. Similarly, the sensitivity of $S_d^{(g_p \neq g_m)}$ to a change in g_{c2} is,

$$\begin{aligned} Sens[S_d^{(g_p \neq g_m)} \text{ to } g_{c2}] &\triangleq \frac{dS_d^{(g_p \neq g_m)}}{dg_{c2}} \times \frac{g_{c2}}{S_d^{(g_p \neq g_m)}} \\ &= -\left(\frac{g_{c2}g_p}{1 + g_{c2}g_p} \right) \end{aligned} \quad (4.3.10)$$

The numerators of Eqs.(4.3.9) and (4.3.10) are compared.

Since α^* is nothing but $\left(1 - S_d^{(g_p = g_m)}(e^{j\omega T})\right) \Delta^*(e^{j\omega T})$, if the Robust Stability condition in Eq.(4.3.3) is satisfied, we have $|\alpha^*(j\omega)| < 1 \quad \forall 0 < \omega \leq \frac{\pi}{T}$. (This is true when the open loop includes an integrator term which it does because of the structure

of g_{c2} (Eq.(4.2.8))). Meanwhile, the numerator of Eq.(4.3.10), the open loop magnitude $|g_{c2}(e^{j\omega T})g_p(e^{j\omega T})|$ will normally be greater than 1 for low frequencies. Therefore,

$$Sens[S_r^{(g_p \neq g_m)} \text{ to } g_{c2}] \ll Sens[S_d^{(g_p \neq g_m)} \text{ to } g_{c2}] \quad \forall \text{ low } \omega \quad (4.3.11)$$

Hence, online tuning for the disturbance rejection response (by adjusting g_{c2} in some way) does not affect the tracking response much, even under mismatch conditions, provided that the Robust Stability condition is satisfied. $\square$

This does not imply that set-point response under nominal and mismatch conditions are approximately the same; only that tuning GPC_2 online (by adjusting k_2) will not affect the tracking response much in comparison to the disturbance rejection response.

4.3.2.1 Numerical Example

To examine Eq.(4.3.8), a numerical example is used in which $g_m(s) = \frac{1}{2s+1}e^{-s}$, and g_p was taken as the 20% worst-case mismatch plant, $g_p(s) = \frac{1.2}{1.6s+1}e^{-1.2s}$. g_{c2} was evaluated with the settings $T = 0.5s$, $N_{a2} = 1$, $N_{b2} = 40$, $N_{u2} = 1$ and $\lambda_2 = 10$. The multiplicative uncertainty $\Delta^*(z)$ corresponding to this worst-case $g_p(z)$ was calculated using Eq.(4.3.4).

Figure(4.3.1) compares the Sensitivities of $S_r^{(g_p \neq g_m)}$ and $S_d^{(g_p \neq g_m)}$ to g_{c2} , and illustrates the fact of Eq.(4.3.8). Note that at high frequencies, Eq.(4.3.8) does not hold. Theorem(3) will be further investigated in Section(4.4.1) with closed loop simulations.

4.3.3 Robustness

A chief consequence of Theorem (3) is that we can operate the 2GPC loop at a level of *Robustness* greater than that of a conventional GPC loop, without sacrificing *Tracking Performance*. Let initially $g_{c1} = g_{c2}$, which can be done by keeping all the parameters of GPC_1 and GPC_2 the same. Then, the 2GPC has the same *Tracking Performance* and *Robustness* as a Single GPC loop with GPC_1 as the controller, as inferred from Eq.(4.2.12). At this point, if k_2 is increased, then by Theorem (3) (assuming that the Robust Stability Condition in Eq.(4.3.3) is satisfied), the *Tracking Performance* is relatively unchanged while the disturbance rejection response is slowed down. But, by Eq.(4.3.11), increasing k_2 also increases the *Robustness*. Hence, we gain *Robustness* with no significant loss in *Tracking Performance*.

4.3.4 Stability

For nominal stability, Eq.4.2.9 suggests that the two characteristic polynomials, $1 + g_{c1}g_m$ and $1 + g_{c2}g_m$ should be stable. This depends to a large extent on the tuning parameters,

Figure 4.3.1: Comparing Sensitivities of $S_r^{(g_p \neq g_m)}$ and $S_d^{(g_p \neq g_m)}$ to g_{c2}

especially in the case of unstable systems. If for FOPTD (stable or unstable) processes, the N^* tuning method is used for both GPC_1 and GPC_2 , then nominal stability is guaranteed.

Internal stability is also guaranteed even for unstable processes. This is due to the structure of the GPC itself which ensures that no unstable pole-zero cancellation occurs in the open loop [52].

4.3.5 Summary

To summarize, with the 2GPC algorithm, it is not possible to achieve the ideal of *total* transparent online tuning, whereby, 3 parameters of the controller can independently tune for the three most critical loop attributes. As such, there are only two parameters k_1 and k_2 , one for adjusting *Nominal Set-Point Tracking Performance* and the other for trading off *Nominal Disturbance Rejection Performance* vs loop *Robustness*. The important point is that the 2GPC maintains this extent of **Transparent Online Tuning** even under mismatch conditions.

4.4 Simulation and Experiments

4.4.1 Transparent Tuning

The simple example system $g_m(z) = \frac{0.04877}{(1-0.9512z^{-1})}z^{-10}$ (which is the discrete-time equivalent of $\frac{1}{2s+1}e^{-s}$ with zero-order hold, sampled at 0.1 secs) is used to demonstrate the responses of the 2GPC control loop, under nominal conditions. The parameters of GPC_1 and GPC_2 were chosen following the N^* tuning method.

Figure(4.4.1) shows the simulation results of the 2GPC controller when the set-point signal $r = 1$ is applied at $t = 0$ and an output step disturbance $d = -1$ is injected at time $t = 10$. The figures also demonstrate what happens when k_1 and k_2 are varied. Obviously, under nominal conditions, changing k_1 changes only GPC_1 which affects only the set-point tracking response. Similarly, varying k_2 affects only the GPC_2 controller and thus only the disturbance rejection response.

The same simulations were conducted in the presence of a 20% worst case model-mismatch[43]

$$g_m(z) = \frac{0.04877}{(1 - 0.9512z^{-1})}z^{-10} \neq g_p(z) = \frac{0.0727}{(1 - 0.9394z^{-1})}z^{-12} \quad (4.4.1)$$

keeping the same controller parameter values. ($g_p(z)$ is the discrete-time version of $\frac{1.2}{1.6s+1}e^{-1.2s}$ with zero-order hold, sampled at 0.1 secs). Figure(4.4.2) shows the responses

when varying k_1 and k_2 respectively. Varying k_1 affects only the set-point tracking response while varying k_2 affects mainly the disturbance rejection response as implied by Theorem (3). k_2 also balances a trade-off between *Disturbance Rejection Performance* and loop *Robustness*.

Futhermore, in the Figures (4.4.1) & (4.4.2) , the responses with $k_1 = k_2$ are also the responses of a single GPC loop (when all the parameters of GPC_1 and GPC_2 are the same, in accordance with Eq.(4.2.12), the 2GPC loop behaves as a conventional GPC loop).

Figure 4.4.1: Closed loop responses of the 2GPC for the Nominal case: (a)Varying k_1 ($k_2 = 0.7$) (b)Varying k_2 ($k_1 = 0.7$) (A setpoint of 1 is applied at time $t = 0$ and a disturbance of -1 is applied at $t = 10$)

4.4.2 Robustness

Here, it will be shown that the Transparent Online Tuning feature of the 2GPC can be utilized to obtain greater *Robustness* than the standard GPC algorithm, without losing

Figure 4.4.2: Closed loop responses of the 2GPC for the Mismatch case (a)Varying k_1 ($k_2 = 0.95$) (b)Varying k_2 ($k_1 = 0.9$) (A setpoint of 1 is applied at time $t = 0$ and a disturbance of -1 is applied at $t = 15$)

Tracking Performance.

The example system considered is,

$$g_p(z) = \frac{0.095z^{-1}}{1 - 0.905z^{-1}}z^{-22}$$

It is known that for higher delay systems, even a 10% error in the estimated delay can cause instability of the GPC loop[52]. So, the estimated delay is assumed to be off by $\Delta d = 2$ units, so that,

$$g_m(z) = \frac{0.095z^{-1}}{1 - 0.905z^{-1}}z^{-20}$$

Therefore, using Eq.(4.3.4),

$$\Delta^*(e^{j\omega T}) = e^{-(\Delta d)j\omega} - 1 \quad (4.4.2)$$

The condition for Robust Stability given in Eq.(4.3.6) is,

$$|\Delta^*(e^{j\omega T})| < \frac{1}{|1 - S_d^{(g_p=g_m)}(e^{j\omega T})|} = \Delta_{lim}^*(\omega) \quad \forall 0 < \omega < \frac{\pi}{T}$$

where $\Delta_{lim}^*(\omega)$ represents the *robustness limit*.

A 2GPC is designed for $g_m(z)$ with parameters $k_1 = k_2 = 0.87$; since the parameters of both GPC_1 and GPC_2 are the same, this setting is equivalent to a single GPC loop with $k = 0.87$. Then, another setting with $k_1 = 0.87$ and $k_2 = 0.95$ is examined. The Δ_{lim}^* of both settings and the multiplicative uncertainty $|\Delta^*(e^{j\omega T})|$ corresponding to the error in delay estimation are shown in Figure(4.4.3). While both settings satisfy the Robust Stability condition, the second setting with $k_2 = 0.95$ offers greater robustness. Thus, by increasing (online) the value of k_2 , a higher *Robustness* than a GPC loop can be obtained.

But the k parameter of the single GPC could also have been increased to obtain the same higher *Robustness*. The difference is that while increasing k of the GPC loop from 0.87 to 0.95 will drastically slow down both the tracking and the disturbance rejection responses, doing the same on the 2GPC will not affect the *Tracking Performance* much *in comparison* to the drop in *Disturbance Rejection Performance*. In other words, if k of the conventional GPC is used to tune online for *Robustness*, it would result in the drop in both *Performances* - tracking and disturbance rejection whereas tuning k_2 online only trades off *Robustness* with *Disturbance Rejection Performance*, even under mismatch conditions.

Figure (4.4.4) examines the closed loop responses for three settings $k_1 = k_2 = 0.87$, $k_1 = k_2 = 0.95$ and $k_1 = 0.87$, $k_2 = 0.95$. It is clear that the responses for the setting $k_1 = k_2 = 0.87$ (which are the responses of the single GPC with $k = 0.87$) are undesirable and that the setting also has poor *Robustness*. Meanwhile, the settings $k_1 = k_2 = 0.95$ and $k_1 = 0.87$, $k_2 = 0.97$ have the same disturbance rejection response but the latter setting has better *Tracking Performance*, while both have the same *Robustness*.

4.4.3 Level Control in a Single Tank System

The PCS with 2GPC was implemented for level control of a single tank system, shown in Figure (2.4.3) and the details regarding the system and its identification have already been discussed in Section (2.4.3).

The (unconstrained) 2GPC control loop was implemented with a sampling time of $T = 0.1s$ and was designed with the N^* tuning settings $k_1 = 0.98$ and $k_2 = 0.98$. The system was first brought to a steady state of $13.6cm$ by applying an input voltage of $6v$. Then, the loop was closed with a deviation setpoint of 0. At a time $t = 30s$ after

Figure 4.4.3: Robustness Limits of the 2GPC for the two cases ($k_1 = 0.87, k_2 = 0.87$) and ($k_1 = 0.87, k_2 = 0.95$)

data logging starts, a deviation setpoint of $3cm$ was applied and at $t = 180s$, an input disturbance of $+4v$ was applied to the closed loop.

The closed loop responses obtained when varying k_1 are shown in figure (4.4.5). Similarly, the closed loop responses obtained when varying k_2 are shown in Figure (4.4.6). The experimental results in Figures (4.4.5) and (4.4.6) demonstrate that the transparent online tuning is viable in practice with the use of the PCS with GPC; the disturbance rejection response and the tracking response can be adjusted independently and effectively.

In order to validate Theorem(7), the experiment was repeated by introducing model mismatch on purpose. The model used for controller tuning was taken as,

$$\frac{1.7357}{40.7662s + 1} e^{-0.25s}$$

so that the actual process (Eq.(2.4.1)) was the 20% worst-case mismatch. The closed loop responses with $A_{m1} = 60$ and varying A_{m2} are shown in Figure(4.4.7). It can be observed

Figure 4.4.4: Closed Loop Responses of the 2GPC for different k_1 k_2 under mismatch settings (the delay of the plant and model differ by 2 samples) (A setpoint of 1 is applied at $t = 0$ and an output disturbance of -1 is applied at $t = 50$)

that the variation in g_{c2} doesn't affect the tracking response much. (However, this does not mean that the responses are exactly the same as when conditions were nominal in Figure (4.4.6)).

4.5 Discussions

The variants of GPC mentioned in Section(1.5) can be considered in the common framework of the structure Figure(1.5.1). The closed loop transfer function of the structure is,

$$y_p(t) = \frac{g_p CR}{1 + Cg_m^* + CR(g_p - g_m)} \times \frac{W}{R} \times r(t) + \frac{1 + Cg_m^* - CRg_m}{1 + Cg_m^* + CR(g_p - g_m)} \times d_o(t) \quad (4.5.1)$$

Figure 4.4.5: 2GPC Experimental Results: Varying k_1 varies the *Tracking Performance* alone ($k_2 = 0.98$) (A setpoint of 3cm was applied at $t = 30s$ and an input disturbance of +4v was applied at $t = 180s$)

Its obvious that the filter R does not appear in the *nominal* tracking transfer function (ie. when $g_p = g_m$). The *Robustness Limit* for this structure is,

$$\Delta_{lim}^*(\omega) = \left| \frac{1 + Cg_m^*}{g_m C} \right| \times \frac{1}{|R|}$$

At the same time, the nominal transfer function between u and d_o is,

$$\left| \frac{u}{d_o} \right| = \left| \frac{CR}{1 + Cg_m^*} \right|$$

which must be close to 1 for all frequencies inside the closed loop bandwidth, for good *Disturbance Rejection Performance*. But,

$$\left| \frac{u}{d_o} \right| = \frac{1}{\Delta_{lim}^*(\omega) |g_m|}$$

Figure 4.4.6: 2GPC Experimental Results: Varying k_2 varies the *Disturbance Rejection Performance* alone ($k_1 = 0.98$) (A setpoint of 3cm was applied at $t = 30s$ and an input disturbance of +4v was applied at $t = 180s$)

which indicates that *Robustness* and *Disturbance Rejection Performance* are inversely related. Thus, R is used to trade-off *Robustness* with *Disturbance Rejection Performance* by directly designing R as is the case with DTC-GPC or indirectly via a good choice of the T polynomial. But R remains in the tracking transfer function when the general (non-ideal) closed loop transfer function is considered. Thus, in practical situations, if R is used for online adjustment of *Robustness* vs *Disturbance Rejection Performance*, there is the possibility that it could affect the *Tracking Performance* adversely. In other words, it cannot be proved that the sensitivity of the tracking transfer function to R is far less when compared to the sensitivity of the disturbance rejection transfer function to R and therefore, *Transparent Online Parameter Tuning* is not possible. On the other hand, with the 2GPC method, Theorem(3) shows that g_{c2} (which is analogous to R) can be used for online tuning without much effect upon the tracking transfer function.

There is also a stark difference in the design philosophy of the 2GPC method and

Figure 4.4.7: 2GPC Mismatch Results: Varying k_2 varies mainly the *Disturbance Rejection Performance* alone ($k_1 = 0.98$) (A setpoint of 3cm was applied at $t = 30$ s and an input disturbance of +4v was applied at $t = 180$ s)

that of DTC-GPC and T polynomial methods:

- The latter methods consist of a two step design: the user must first choose GPC parameters for desired *Nominal Tracking Performance*. Any of the popular parameter tuning rules[75] may be employed. In the second step, the user must design R (or indirectly, T) to improve *Robustness* at the cost of *Disturbance Rejection Performance*. This is similar to IMC design[48].
- Meanwhile, with the 2GPC approach, the user first chooses the *same* GPC parameters (for both GPC_1 and GPC_2) for desired *Nominal Tracking Performance*. Then, the user is free to tweak the *single online tuning parameter* k_2 of GPC_2 (online or offline in simulation) to improve robustness knowing that this will not affect the tracking response much (theorem 5) in comparison to the disturbance rejection response. In contrast, changing online any parameter in R in Eq.(4.5.1) will change

the tracking response in some unknown manner.

Firstly, as can be seen, the former method is a static offline design approach while the 2GPC method has a flexible online parameter tuning approach. When compared with one-shot design approach of the DTC-GPC and T polynomial methods, the 2GPC's *Transparent Online Parameter Tuning* approach may be more convenient and simpler for the user when there is a need for repeated online parameter tuning during initial implementation stages and during maintenance stages to overcome process variability. It is stressed that the objective of the 2GPC method is to provide the practitioner this flexibility and not to provide a method that offers better robustness or performance than the existing variants. Thus, while it was necessary to place the 2GPC in the context of existing GPC variants, considering that their objectives and design methodology are different, they cannot be compared, only contrasted.

Other differences include the fact that with the DTC-GPC, the filter R can be designed in detail to improve the *Robustness Limit* at particular frequencies. This is not possible with the 2GPC where increasing k_2 increases Robustness in general, as demonstrated by the example in Section (4.4.2). Another disadvantage of the 2GPC would be that by setting the parameters of the both GPC_1 and GPC_2 the same, as prescribed for the first step of its design, the control horizons are set as $N_{u1} = N_{u2} = N_u$ which means that the 2GPC optimization is now twice the size of the regular QP problem of the conventional GPC. However, simulation studies of FOPTD systems (not given in this paper) have shown that not much performance change occurs when the maximum allowed size of the optimization variable is distributed between N_{u1} & N_{u2} . For eg., if the platform for implementation can handle $N_u = 5$ then this limit can be distributed into $N_{u1} = 2$ & $N_{u2} = 3$, without much difference as compared to using the prescribed $N_{u1} = 5$ & $N_{u2} = 5$.

As mentioned in Section (4.2.2), one of the properties of the 2GPC is that, by Eq.(4.2.9), under nominal conditions, the tracking and disturbance rejection responses are perfectly decoupled. Initially setting the parameters of GPC_1 and GPC_2 the same will make the 2GPC function exactly as the conventional GPC, with the disturbance rejection relatively aggressive compared to the tracking. Then, the parameters of GPC_2 can be adjusted offline to obtain an acceptable disturbance rejection response, without affecting the nominal tracking response. As highlighted in Section (1.5), this is difficult to do with the conventional GPC. In this way, the limitations of the GPC in the context of offline parameter tuning are overcome.

4.6 Chapter Conclusions

The 2GPC algorithm was developed as a single QP optimization problem integrating system constraints, as with the conventional GPC algorithm. Though the (constrained) 2GPC controller is a single entity from a single optimization problem, the (unconstrained) 2GPC can be viewed structurally as two controllers; GPC_1 controls the tracking response while GPC_2 controls the disturbance rejection response. The explicit control law and closed loop transfer function of the 2GPC were examined. It was suggested that the (offline) tuning of 2GPC be performed using standard techniques[75, 61], initially setting the parameters of GPC_1 and GPC_2 the same. The practitioner is then free to use the control weighting parameters of GPC_1 and GPC_2 to perform *Transparent Online Parameter Tuning* (to an extent). k_1 lets the user vary *Tracking Performance* only while k_2 can be used to trade-off *Disturbance Rejection Performance* vs *Robustness*. Importantly, it was proved using derivations that this level of *Transparent Online Parameter Tuning* is maintained even under mismatch conditions, provided the Robust Stability condition of the 2GPC loop is satisfied. Finally, it was also proved that utilizing the *Transparent Online Parameter Tuning* feature of the 2GPC controller, it is possible to achieve *Robustness* greater than that of a conventional GPC loop, without sacrificing *Tracking Performance*. Simulations studies were provided to demonstrate the *Transparent Online Parameter Tuning* and greater *Robustness* features of the 2GPC. Experimental results from the 2GPC level control of single water tank system were provided to show that 2GPC and *Transparent Online Parameter Tuning* were viable in practise.

The 2GPC method was contrasted with existing variants of GPC in terms of design methodology and objective. It was explained that the existing GPC variants do not have *Transparent Online Parameter Tuning* feature. Also, the disadvantages of the 2GPC in comparison to the existing GPC variants were highlighted. Finally, it was explained that offline parameter tuning for adjusting the disturbance rejection response could be handled much better than with the conventional GPC.

Potential future research directions for the 2GPC method include an extension to state space based predictive control, more detailed study of applicability of 2GPC to unstable, integrator and RHP zero systems and to the multivariable case.

Chapter 5

GPC with Constant Disturbance Model (CDGPC)

In this variant of the GPC, instead of using the CARIMA model for output predictions, the transfer function model will be used. Output predictions at time $t + j$ will simply be the predicted output of the model plus the plant-model error at time t . While this constant disturbance model is used in other variants of predictive control such as Dynamic Matrix Control (DMC) and Model Predictive Control (MPC) (which uses state space model), the CDGPC (Constant Disturbance GPC) differs from them in a few ways. Firstly, unlike the DMC which uses a step response model, the CDGPC uses a transfer function model and can thus be used for the control of integrator systems. Otherwise, for stable processes, the tracking response, disturbance rejection response and the robustness of both the CDGPC and the DMC will be identical. The MPC also uses a constant disturbance model, and while under nominal conditions, the MPC and the CDGPC will have the same tracking response, depending on how the what state observer is used with the MPC and how it is tuned, the disturbance rejection and the robustness of the MPC will differ from that of the CDGPC.

Finally, the CDGPC has not yet been explored in the predictor filter framework of the GPC as proposed by Normey Rico [52], shown in Figure (1.5.1). In this chapter, this will be explored and the structure of the predictor filter of the CDGPC will be explored. Once this is established, the CDGPC will be compared with the GPC and its other variant, the SPGPC (Smith Predictor GPC). The CDGPC will be shown to offer the greatest robustness, followed by the SPGPC and then the GPC. Moreover, it will be shown that the Robustness of the GPC depends upon the process delay; robustness is lower for a higher delay. Meanwhile, the robustness of the SPGPC and the CDGPC are independent of the delay. In essence, the CDGPC takes the best features of the DMC and the GPC.

The chapter is sectioned as follows:

- Section (5.1) details the derivation of the CDGPC control law by the conventional or general method. The control law and closed loop transfer function are also derived
- Section (5.2) details the derivation by the alternate method. The structure of the loop by this method helps analyze the CDGPC and understand its properties.
- Section (5.3.1) demonstrates that the CDGPC has superior Robustness as compared to the conventional GPC and the SPGPC (Smith Predictor GPC).

5.1 CDGPC - General Method

Let the nominal model of the plant be,

$$A(z^{-1})y_m(t) = B(z^{-1})z^{-d}u(t-1) \quad (5.1.1)$$

where,

$$\begin{aligned} A(z^{-1}) &= 1 + a_1z^{-1} + \dots + a_{n_a}z^{-n_a} \\ B(z^{-1}) &= b_0 + b_1z^{-1} + \dots + b_{n_b}z^{-n_b} \end{aligned}$$

The degrees of the polynomials are $\delta(A) = n_a$ and $\delta(B) = n_b$. And d is the discrete-time delay. Let the actual behaviour of the real plant be governed by the equation,

$$A(z^{-1})y_p(t) = B(z^{-1})z^{-d}u(t-1) + A(z^{-1})e(t) \quad (5.1.2)$$

where $e(t)$ is representative of uncertainty, noise and output disturbances.

Subtracting Eq.(5.1.1) from Eq.(5.1.2) gives,

$$e(t) = y_p(t) - y_m(t) \quad (5.1.3)$$

5.1.1 Prediction Equation

- First $y_m(t+j)$ is derived. Multiplying Eq.(5.1.1) by Δ (this will be required for integrative action in the controller) and time shifting by j ,

$$\Delta A(z^{-1})y_m(t+j) = B(z^{-1})z^{-d}\Delta u(t+j-1)$$

Let $\tilde{A}(z^{-1}) = \Delta A(z^{-1})$ and $\tilde{B}(z^{-1}) = B(z^{-1})z^{-d}$ (so $\delta(\tilde{A}) = n_a + 1$ and $\delta(\tilde{B}) = d + n_b$). So,

$$\tilde{A}(z^{-1})y_m(t+j) = \tilde{B}(z^{-1})\Delta u(t+j-1) \quad (5.1.4)$$

- Consider the Diophantine Equation,

$$1 = \tilde{A}(z^{-1})E_j(z^{-1}) + z^{-j}F_j(z^{-1}) \quad (5.1.5)$$

$$\text{or} \quad 1 - z^{-j}F_j(z^{-1}) = E_j(z^{-1})\tilde{A}(z^{-1}) \quad (5.1.6)$$

where $\delta(E_j) = j - 1$ and $\delta(F_j) = n_a$.

- Let $F_j(z^{-1})$ be expressed as,

$$F_j(z^{-1}) = f_{j,0} + f_{j,1}z^{-1} + \cdots + f_{j,n_a}z^{-n_a} \quad (5.1.7)$$

- Eq.(5.1.4) is multiplied with $E_j(z^{-1})$ and then $E_j\tilde{A}$ is substituted using Eq.(5.1.6),

$$\begin{aligned} [1 - z^{-j}F_j(z^{-1})] y_m(t+j) &= E_j(z^{-1})\tilde{B}(z^{-1})\Delta u(t+j-1) \\ y_m(t+j) &= E_j(z^{-1})\tilde{B}(z^{-1})\Delta u(t+j-1) + F_j(z^{-1})y_m(t) \end{aligned} \quad (5.1.8)$$

It is to be noted that $\delta(E_j\tilde{B}) = j - 1 + n_b + d$.

- The polynomial $E_j(z^{-1})\tilde{B}(z^{-1})$ is to be split into past and future components w.r.t its multiplying signal $\Delta u(t+j-1)$. So, the polynomial $E_j(z^{-1})\tilde{B}(z^{-1})$ of degree $\delta(E_j\tilde{B}) = j - 1 + n_b + d$ is expressed as,

$$\begin{aligned} E_j(z^{-1})\tilde{B}(z^{-1}) &= g_{j,0} + g_{j,1}z^{-1} + \cdots + g_{j,j-1}z^{-(j-1)} + g_{j,j}z^{-j} + \\ &\quad \cdots + g_{j,n_b+d+j-1}z^{-(n_b+d+j-1)} \end{aligned} \quad (5.1.9)$$

$$\begin{aligned} &= (g_{j,0} + g_{j,1}z^{-1} + \cdots + g_{j,j-1}z^{-(j-1)}) \\ &\quad + z^{-j} (g_{j,j} + \cdots + g_{j,n_b+d+j-1}z^{-(n_b+d-1)}) \end{aligned} \quad (5.1.10)$$

$$\therefore E_j(z^{-1})\tilde{B}(z^{-1}) \equiv G_j(z^{-1}) + z^{-j}I_j(z^{-1}) \quad (5.1.11)$$

where $\delta(G_j) = j - 1$ and $\delta(I_j) = n_b + d - 1$. **Attention Point:** If $n_b = d = 0$, then $\delta(I_j) = -1$ which simply means that $I_j(z^{-1}) = 0$, the zero polynomial.

- $G_j(z^{-1})$ and $I_j(z^{-1})$ are expressed as,

$$G_j(z^{-1}) = g_{j,0} + g_{j,1}z^{-1} + \cdots + g_{j,d-1}z^{-(d-1)} + g_{j,d}z^{-d} + \cdots + g_{j,j-1}z^{-(j-1)} \quad (5.1.12)$$

$$I_j(z^{-1}) = i_{j,0} + i_{j,1}z^{-1} + \cdots + i_{j,n_b+d-1}z^{-(n_b+d-1)} \quad (5.1.13)$$

Later, when deriving the **Prediction Vector**, j will be chosen such that $j \geq$

$d + 1$. Therefore, $\delta(G_j) = (j - 1) \geq d \forall j$ and $g_{j,0} = g_{j,1} = \dots = g_{j,d-1} = 0$.

$$\therefore G_j(z^{-1}) = z^{-d} (g_{j,d} + \dots + g_{j,j-1} z^{-(j-d-1)}) \equiv z^{-d} X_j(z^{-1}) \quad (5.1.14)$$

where $\delta(X_j) = j - d - 1$. X_j is expressed separately as,

$$X_j(z^{-1}) = x_{j,0} + x_{j,1} z^{-1} + x_{j,2} z^{-2} + \dots + x_{j,j-d-1} z^{-(j-d-1)}$$

Again, because j is such that $j \geq d + 1$, we substitute j with $j + d$ in X_j to get,

$$X_{j+d}(z^{-1}) = x_{j+d,0} + x_{j+d,1} z^{-1} + x_{j+d,2} z^{-2} + \dots + x_{j+d,j-1} z^{-(j-1)} \quad (5.1.15)$$

- Eq.(5.1.11) is substituted into Eq.(5.1.8),

$$y_m(t+j) = \underbrace{G_j(z^{-1}) \Delta u(t+j-1)}_{future} + \underbrace{I_j(z^{-1}) \Delta u(t-1) + F_j(z^{-1}) y_m(t)}_{P_j(z^{-1})-past} \quad (5.1.16)$$

where,

$$P_j(z^{-1}) = I_j(z^{-1}) \Delta u(t-1) + F_j(z^{-1}) y_p(t) \quad (5.1.17)$$

is the **Estimated Free Response Portion** of the prediction. (If no input is supplied, the estimate $\hat{y}_p(t+j|t)$ would simply be $P_j(z^{-1})$; hence it is called 'Free Response').

- The prediction $\hat{y}_p(t+j|j)$ is yet to be found. From Eq. (5.1.3), it is clear that,

$$\begin{aligned} y_p(t) &= y_m(t) + e(t) \\ y_p(t+j) &= y_m(t+j) + e(t+j) \end{aligned}$$

Taking Expectations on both sides,

$$\hat{y}_p(t+j|t) = y_m(t+j) + \hat{e}(t+j|t)$$

- Since $\hat{e}(t+j|t)$ is not known, we assume **constant disturbance**,

$$\hat{e}(t+j|t) = e(t) = y_p(t) - y_m(t)$$

Therefore,

$$\hat{y}_p(t+j|t) = y_m(t+j) + e(t)$$

Finally, substituting for $y_m(t+j)$ from Eq. (5.1.16) gives,

$$\hat{y}_p(t+j|t) = G_j(z^{-1}) \Delta u(t+j-1) + I_j(z^{-1}) \Delta u(t-1) + F_j(z^{-1}) y_m(t) + e(t) \quad (5.1.18)$$

- Therefore $\hat{y}_p(t+d+j|t)$ is given by,

$$\begin{aligned} \hat{y}_p(t+d+j|t) &= G_{j+d}(z^{-1}) \Delta u(t+j+d-1) + I_{j+d}(z^{-1}) \Delta u(t-1) \\ &\quad + F_{j+d}(z^{-1}) y_m(t) + e(t) \end{aligned} \quad (5.1.19)$$

where using Eq.(5.1.14) and Eq.(5.1.15),

$$\hat{y}_p(t+d+j|t) = X_{j+d}(z^{-1}) \Delta u(t+j-1) + I_{j+d}(z^{-1}) \Delta u(t-1) + F_{j+d}(z^{-1}) y_m(t) + e(t) \quad (5.1.20)$$

This is the final prediction equation is the one that will be used in the prediction vector.

5.1.2 Prediction Vector

- The output prediction vector required for GPC optimization is,

$$\hat{Y} = \begin{bmatrix} \hat{y}_p(t+d+N_1|t) \\ \hat{y}_p(t+d+N_1+1|t) \\ \vdots \\ \hat{y}_p(t+d+N_2|t) \end{bmatrix} = \begin{bmatrix} \hat{y}_p(t+M_1|t) \\ \hat{y}_p(t+M_1+1|t) \\ \vdots \\ \hat{y}_p(t+M_2|t) \end{bmatrix} \quad (5.1.21)$$

where $M_1 = N_1 + d$ and $M_2 = N_2 + d$.

- Let $\hat{U}$ be the vector of *change in control actions* that can be applied to the plant,

$$\hat{U} = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-1) \end{bmatrix} \quad (5.1.22)$$

$\hat{U}$ will later become the *optimization variable*.

- Substituting for the individual elements using Eq.(5.1.20), we get,

$$\hat{Y} = \begin{bmatrix} X_{N_1+d}(z^{-1})\Delta u(t+N_1-1) \\ X_{N_1+d+1}(z^{-1})\Delta u(t+N_1) \\ \vdots \\ X_{N_2+d}(z^{-1})\Delta u(t+N_2-1) \end{bmatrix} + \underbrace{\begin{bmatrix} I_{M_1}(z^{-1}) \\ I_{M_1+1}(z^{-1}) \\ \vdots \\ I_{M_2}(z^{-1}) \end{bmatrix} \Delta u(t-1) + \begin{bmatrix} F_{M_1}(z^{-1}) \\ F_{M_1+1}(z^{-1}) \\ \vdots \\ F_{M_2}(z^{-1}) \end{bmatrix} y_m(t) + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} e(t)}_P$$

where P is the **Estimated Free Response Vector** in vector polynomial form and where $\delta(X_{N_1+d}) = N_1 - 1$, $\delta(X_{N_1+d+1}) = N_1$, $\dots$ $\delta(X_{N_2+d}) = N_2 - 1$.

- Substituting for the individual polynomials using Eqs. (5.1.7) and (5.1.11), the prediction vector in matrix form is,

$$\hat{Y} = \underbrace{\begin{bmatrix} x_{M_1,N_1-1} & x_{M_1,N_1-2} & \cdots & x_{M_1,1} & x_{M_1,0} & 0 & \cdots & 0 \\ x_{M_1+1,N_1} & x_{M_1+1,N_1-1} & \cdots & x_{M_1+1,2} & x_{M_1+1,1} & x_{M_1+1,0} & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{M_2,N_2-1} & x_{M_2,N_2-2} & \cdots & \cdots & \cdots & \cdots & \cdots & x_{M_2,0} \end{bmatrix}}_{(N_2-N_1+1) \times N_2} \underbrace{\begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_1-1) \\ \vdots \\ \Delta u(t+N_u-1) \\ \cdots \\ \Delta u(t+N_u) \\ \vdots \\ \Delta u(t+N_2-2) \\ \Delta u(t+N_2-1) \end{bmatrix}}_{N_2 \times 1}$$

$$\begin{aligned}
& + \underbrace{\begin{bmatrix} i_{M_1,0} & \cdots & i_{M_1,n_b+d-1} \\ \vdots & \ddots & \vdots \\ i_{M_2,0} & \cdots & i_{M_2,n_b+d-1} \end{bmatrix}}_{(N_2-N_1+1) \times (n_b+d)} \underbrace{\begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix}}_{(n_b+d) \times 1} \\
& + \underbrace{\begin{bmatrix} f_{M_1,0} & \cdots & f_{M_1,n_a} \\ \vdots & \ddots & \vdots \\ f_{M_2,0} & \cdots & f_{M_2,n_a} \end{bmatrix}}_{(N_2-N_1+1) \times (n_a+1)} \underbrace{\begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix}}_{(n_a+1) \times 1} + \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{(N_2-N_1+1) \times 1} e(t) \\
\therefore \hat{Y} = G & \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_1-1) \\ \vdots \\ \Delta u(t+N_u-1) \\ \cdots \cdots \cdots \\ \Delta u(t+N_u) \\ \vdots \\ \Delta u(t+N_2-2) \\ \Delta u(t+N_2-1) \end{bmatrix} + I \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} + F \begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix} + \mathbf{1}e(t)
\end{aligned} \tag{5.1.23}$$

- Since only the first N_u *change in control action* are allowed (that is, $\Delta u(t+N_u) = \cdots = \Delta u(t+N_2-1) = 0$), the G matrix is truncated as $G \leftarrow G(N_1 : N_2, 1 : N_u)$ so that the prediction vector is now,

$$\therefore \hat{Y} = G \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-1) \end{bmatrix} + I \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} + F \begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix} + \mathbf{1}e(t)$$

which is of the form,

$$\hat{Y} = G\hat{U} + P \tag{5.1.24}$$

where P is the **Free Response Vector** in matrix form,

$$\begin{aligned}
 P &= I \underbrace{\begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix}}_{\text{matrix form}} + F \underbrace{\begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix}}_{\text{matrix form}} + \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{\text{matrix form}} e(t) \quad (5.1.25) \\
 &= \underbrace{\begin{bmatrix} I_{M_1}(z^{-1}) \\ I_{M_1+1}(z^{-1}) \\ \vdots \\ I_{M_2}(z^{-1}) \end{bmatrix} \Delta u(t-1) + \begin{bmatrix} F_{M_1}(z^{-1}) \\ F_{M_1+1}(z^{-1}) \\ \vdots \\ F_{M_2}(z^{-1}) \end{bmatrix} y_m(t) + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{\text{polynomial form}} e(t) \quad (5.1.26)
 \end{aligned}$$

5.1.3 Control Law

The explicit control law after minimization is,

$$\Delta u(t) = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \hat{U} = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} K [W - P] \quad (5.1.27)$$

where $K = \left[(G^T G + \lambda I)^{-1} G^T \right]_{N_u \times (N_2 - N_1 + 1)}$, λ is the control weighting and W is the vector of future set-points,

$$W = \underbrace{\begin{bmatrix} w(t+N_1) \\ \vdots \\ w(t+N_2-1) \\ w(t+N_2) \end{bmatrix}}_{(N_2+N_1-1) \times 1} \quad (5.1.28)$$

and in the case where the future setpoints are unknown, they are simply taken as the value of the set-point signal at the current time, t . That is,

$$W = \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{(N_2+N_1-1) \times 1} w(t) \equiv \mathbf{1} w(t) \quad (5.1.29)$$

5.1.3.1 Transfer Function form

Substituting the **polynomial form** of the Free Response Vector P from Eq.(5.1.25), $\Delta u(t)$ in Eq.(5.1.27) is,

$$\Delta u(t) = \begin{bmatrix} 1 & \cdots & 0 \end{bmatrix} K \left\{ \mathbf{1} \times w(t) - \begin{bmatrix} I_{M_1}(z^{-1}) \\ I_{M_1+1}(z^{-1}) \\ \vdots \\ I_{M_2}(z^{-1}) \end{bmatrix} \Delta u(t-1) - \begin{bmatrix} F_{M_1}(z^{-1}) \\ F_{M_1+1}(z^{-1}) \\ \vdots \\ F_{M_2}(z^{-1}) \end{bmatrix} y_m(t) - \mathbf{1} \times e(t) \right\}$$

Let,

$$\begin{bmatrix} 1 & \cdots & 0 \end{bmatrix} K = \begin{bmatrix} c_{N_1} & c_{N_1+1} & \cdots & c_{N_2} \end{bmatrix} \quad (5.1.30)$$

Therefore,

$$\Delta u(t) = k_1 w(t) - K_2(z^{-1}) y_m(t) - K_3(z^{-1}) \Delta u(t-1) - k_4 e(t) \quad (5.1.31)$$

where,

$$k_1 = \left(\sum_{j=N_1}^{N_2} c_j \right) \quad (5.1.32)$$

$$K_2(z^{-1}) = \left(\sum_{i=N_1}^{N_2} c_j F_{j+d}(z^{-1}) \right) \quad (5.1.33)$$

$$K_3(z^{-1}) = \left(\sum_{i=N_1}^{N_2} c_j I_{j+d}(z^{-1}) \right) \quad (5.1.34)$$

$$k_4 = \left(\sum_{j=N_1}^{N_2} c_j \right) = k_1 \quad (5.1.35)$$

5.1.3.2 Matrix Form (for implementation)

Substituting the **matrix form** of the Free Response Vector P from Eq.(5.1.25), $\Delta u(t)$ in Eq.(5.1.27) is,

$$\Delta u(t) = k_1 w(t) - k_2 \begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix} - k_3 \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} - k_4 e(t) \quad (5.1.36)$$

where,

$$[k_1]_{1 \times 1} = [e]_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} [\mathbf{1}]_{(N_2 - N_1 + 1) \times 1} \quad (5.1.37)$$

$$[k_2]_{1 \times (n_a + 1)} = [e]_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} [F]_{(N_2 - N_1 + 1) \times (n_a + 1)} \quad (5.1.38)$$

$$[k_3]_{1 \times (n_b + d)} = [e]_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} [I]_{(N_2 - N_1 + 1) \times (n_b + d)} \quad (5.1.39)$$

$$[k_4]_{1 \times 1} = [e]_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} [\mathbf{1}]_{(N_2 - N_1 + 1) \times 1} = k_1 \quad (5.1.40)$$

Eq.(5.1.36) is the final unconstrained GPC control law for a general SISO system. Eqs.(5.1.31) and (5.1.36) are equivalent. It should be noted that $K_2(z^{-1})$ and $K_3(z^{-1})$ in Eq.(5.1.31) are the transfer function versions of k_2 and k_3 respectively in Eq.(5.1.36).

5.1.3.3 At Steady State

After the control loop reaches steady state, $\Delta u(t) = \Delta u(t-1) = \Delta u(t-2) = \dots \Delta u(t-(n_b+d)) = 0$, and $y_p(t) = w(t)$. And so, Eq.(5.1.36) will become,

$$0 = k_1 w(t) - k_2 \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{(n_a+1) \times 1} y_{mss} - k_4 (w(t) - y_{mss})$$

$$k_1 = k_2 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{(n_a+1) \times 1} = \text{sum}(k_2) \quad (5.1.41)$$

which is an useful result.

5.1.4 Control Structure & Closed Loop Transfer Function - General Method

5.1.4.1 Primary Controller

The Control Law Eq.(5.1.31) in the transfer function form is,

$$\Delta u(t) = k_1 w(t) - K_2(z^{-1}) y_m(t) - K_3(z^{-1}) z^{-1} \Delta u(t) - k_4 e(t)$$

(For ease of reading, the $'(z^{-1})'$ may be skipped). Solving for $u(t)$ gives,

$$\implies [1 + K_3(z^{-1})z^{-1}] \Delta u(t) = k_1 w(t) - K_2 y_m(t) - k_4 e(t) \quad (5.1.42)$$

But $y_m(t) = g_m(z^{-1})u(t)$ and from Eq.(5.1.3) $e(t) = y_p(t) - y_m(t)$. Therefore,

$$\begin{aligned} [1 + K_3 z^{-1}] \Delta u(t) &= k_1 w(t) - K_2 y_m(t) - k_1 (y_p(t) - y_m(t)) \\ &= k_1 w(t) - [K_2 - k_1] g_m u(t) - k_1 y_p(t) \\ [\Delta (1 + K_3 z^{-1}) + (K_2 - k_1) g_m] u(t) &= k_1 w(t) - k_1 y_p(t) \\ \therefore u(t) &= C_{cd} (w(t) - y_p(t)) \end{aligned} \quad (5.1.43)$$

where C_{cd} is the primary controller,

$$C_{cd} = \frac{k_1}{\Delta (1 + K_3 z^{-1}) + (K_2 - k_1) g_m} \quad (5.1.44)$$

Eq.(5.1.43) brings out the fact that the CDGPC is a One Degree of Freedom controller.

5.1.4.2 Closed Loop Transfer Function

$$y_p(t) = g_p(z^{-1}) u(t) + d_o(t) \quad (5.1.45)$$

Substituting for u from Eq.(5.2.25) into Eq.(5.1.45) and simplifying, we can get,

$$y_p(t) = T_{cd} \times w(t) + S_{cd} \times d_o(t) \quad (5.1.46)$$

where S_{cd} is the sensitivity & T_{cd} is the Complementary-Sensitivity of the 1° control loop.

$$T_{cd} = \frac{C_{cd}(z^{-1}) g_p}{1 + C_{cd}(z^{-1}) g_p} = \frac{k_1 g_p}{(1 + K_3 z^{-1}) \Delta + K_2 g_m + k_1 (g_p - g_m)} \quad (5.1.47)$$

$$S_{cd} = \frac{1}{1 + C_{cd}(z^{-1}) g_p} = 1 - T_{cd} \quad (5.1.48)$$

Eq.(5.1.46) is the Closed Loop Transfer Function of a GPC control loop.

5.1.4.3 Alternative look at the Control Law

The control law in transfer function form is,

$$\Delta u(t) = k_1 w(t) - K_2 y_m(t) - K_3 z^{-1} \Delta u(t) - k_1 e(t)$$

Substituting for $y_p(t)$ using Eq.(5.1.45) gives,

$$\Delta u(t) = k_1 w(t) - K_2 y_m(t) - K_3 z^{-1} \Delta u(t) - k_1 [g_p u(t) + d_o(t) - y_m(t)]$$

$$[\Delta(1 + K_3 z^{-1}) + K_2 g_m + k_1(g_p - g_m)] u(t) = k_1(w(t) - d_o(t)) \quad (5.1.49)$$

Eq.(5.1.49) shows that the control law behaves the same for set-point $w(t)$ and for external disturbances $d_o(t)$, and is thus One Degree of Freedom.

5.2 CDGPC - Predictor Seperate Derivation

Most of the derivation by this method is similar to that of the general method from the previous section. Hence, some details are omitted for the sake of brevity and to avoid repetition. Also, the derivation presented below is for the case where $(d > n_a)$. The other case $(d \leq n_a)$ is only slightly different and is omitted.

5.2.1 Prediction Equation

$$\hat{y}_p(t + d + j|t) = \underbrace{X_{j+d}(z^{-1}) \Delta u(t + j - 1)}_{\text{future}} + \underbrace{\bar{I}_j(z^{-1}) \Delta u(t - 1) + F_j(z^{-1}) \hat{y}_p(t + d|t)}_{\bar{P}_j(z^{-1})-\text{past}} \quad (5.2.1)$$

where,

$$\bar{P}_j(z^{-1}) = \bar{I}_j(z^{-1}) \Delta u(t - 1) + F_j(z^{-1}) \hat{y}_p(t + d|t) \quad (5.2.2)$$

is the ***Estimated Free Response Portion*** of the prediction.

5.2.2 Prediction Vector:

$$\hat{Y} = G \begin{bmatrix} \Delta u(t) \\ \Delta u(t + 1) \\ \vdots \\ \Delta u(t + N_u - 1) \end{bmatrix} + \bar{I} \begin{bmatrix} \Delta u(t - 1) \\ \Delta u(t - 2) \\ \vdots \\ \Delta u(t - n_b) \end{bmatrix} + \bar{F} \begin{bmatrix} \hat{y}_p(t + d|t) \\ \hat{y}_p(t + d - 1|t) \\ \vdots \\ \hat{y}_p(t + d - n_a|t) \end{bmatrix} \quad (5.2.3)$$

which is of the form,

$$\hat{Y} = G\hat{U} + \bar{P} \quad (5.2.4)$$

where $\hat{U}$ is the vector of *change in future control actions* and $\bar{P}$ is the **Free Response Vector** ,

$$\begin{aligned} \bar{P} &= \underbrace{\begin{bmatrix} \bar{I}_{N_1}(z^{-1}) \\ \bar{I}_{N_1+1}(z^{-1}) \\ \vdots \\ \bar{I}_{N_2}(z^{-1}) \end{bmatrix} \Delta u(t-1) + \begin{bmatrix} F_{N_1}(z^{-1}) \\ F_{N_1+1}(z^{-1}) \\ \vdots \\ F_{N_2}(z^{-1}) \end{bmatrix} \hat{y}_p(t+d|t)}_{\text{polynomial form}} \\ &= \underbrace{\bar{I} \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-n_b) \end{bmatrix}}_{\text{matrix form}} + \underbrace{\bar{F} \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix}}_{\text{matrix form}} \end{aligned} \quad (5.2.5)$$

5.2.3 Control Law: Case ($d > n_a$)

5.2.3.1 Transfer Function Form

$$\Delta u(t) = k_1 w(t) - L_2(z^{-1}) \hat{y}_p(t+d|t) - L_3(z^{-1}) \Delta u(t-1) \quad (5.2.6)$$

where,

$$k_1 = \left(\sum_{j=N_1}^{N_2} c_j \right) \quad (5.2.7)$$

$$L_2(z^{-1}) = \left(\sum_{i=N_1}^{N_2} c_i F_i(z^{-1}) \right) \quad (5.2.8)$$

$$L_3(z^{-1}) = \left(\sum_{i=N_1}^{N_2} c_i \bar{I}_i(z^{-1}) \right) \quad (5.2.9)$$

5.2.3.2 Matrix form (for implementation)

$$\Delta u(t) = k_1 w(t) - l_2 \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix} - l_3 \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-n_b) \end{bmatrix} \quad (5.2.10)$$

where,

$$[k_1]_{1 \times 1} = e_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} \mathbf{1}_{(N_2 - N_1 + 1) \times 1} \quad (5.2.11)$$

$$[l_2]_{1 \times (n_a + 1)} = e_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} [\overline{F}]_{(N_2 - N_1 + 1) \times (n_a + 1)} \quad (5.2.12)$$

$$[l_3]_{1 \times n_b} = e_{1 \times N_u} [K]_{N_u \times (N_2 - N_1 + 1)} [\overline{I}]_{(N_2 - N_1 + 1) \times n_b} \quad (5.2.13)$$

5.2.4 CDGPC Prediction Equation

The aim of this section is to show how the prediction terms of the control laws in Eqs.(5.2.6) and (5.2.10) are obtained.

5.2.4.1 Matrix Form

- In the case where $d > n_a$ the second term of the control law in Eq.(5.2.10) has only future terms, $\hat{y}_p(t + d - i)$ where $i = 0, 1, \dots, n_a$.
- These future terms can be obtained as follows:

$$\begin{aligned} y_m(t) &= g_m(z^{-1}) u(t) \\ y_m(t) &= \frac{B(z^{-1}) z^{-1}}{A(z^{-1})} z^{-d} u(t) \\ A(z^{-1}) y_m(t) &= \tilde{B}(z^{-1}) u(t - 1) \end{aligned}$$

Multiplying by Δ on both sides, and $t \rightarrow t + j$,

$$\tilde{A}(z^{-1}) y_m(t + j) = \tilde{B}(z^{-1}) \Delta u(t + j - 1)$$

Using the diophantine equation: $E_j \tilde{A} = 1 - z^{-j} F_j$, we get,

$$y_m(t + j) = E_j(z^{-1}) \tilde{B}(z^{-1}) \Delta u(t + j - 1) + F_j(z^{-1}) y_m(t)$$

Putting $j = d - i$ ($i = 0, 1, 2, \dots, n_a$),

$$y_m(t + d - i) = E_{d-i}(z^{-1}) B(z^{-1}) \Delta u(t - i - 1) + F_{d-i}(z^{-1}) y_m(t)$$

Each prediction is added to with a correction term, that is, the estimate of the future output is taken as the model future estimate + the correction term

$$\hat{y}_p(t + d - i|t) = y_m(t + d - i|t) + (y_p(t) - y_m(t)) \quad \forall i = 0, 1, 2, \dots, n_a \quad (5.2.14)$$

so that,

$$\hat{y}_p(t+d-i|t) = E_{d-i}(z^{-1})B(z^{-1})\Delta u(t-i-1) + F_{d-i}(z^{-1})y_m(t) + [y_p(t) - y_m(t)] \quad (5.2.15)$$

- Stacking up all the required future terms into a vector, we get,

$$\begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix} = \begin{bmatrix} E_d B \Delta u(t-1) \\ E_{d-1} B \Delta u(t-2) \\ \vdots \\ E_{d-n_a} B \Delta u(t-n_a-1) \end{bmatrix} + \begin{bmatrix} F_d(z^{-1}) \\ F_{d-1}(z^{-1}) \\ \vdots \\ F_{d-n_a}(z^{-1}) \end{bmatrix} y_m(t) + \begin{bmatrix} y_p(t) - y_m(t) \\ y_p(t) - y_m(t) \\ \vdots \\ y_p(t) - y_m(t) \end{bmatrix}$$

But $\delta(E_{d-n_a}B) = (n_b + d - n_a - 1)$, and $\delta(F_j) = n_a \forall j$. So, we can write this equation in matrix form as,

$$\begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix} = \underbrace{\begin{bmatrix} J \\ \vdots \\ J \end{bmatrix}}_{(n_a+1) \times (n_b+d)} \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} \quad (5.2.16)$$

$$+ \underbrace{\begin{bmatrix} M \\ \vdots \\ M \end{bmatrix}}_{(n_a+1) \times (n_a+1)} \begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} (y_p(t) - y_m(t))$$

- Eq.(5.2.16) is inserted into the Predictor Separate control law in Eq.(5.2.10), to get,

$$\Delta u(t) = k_1 w(t) - l_2 J \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} - l_2 M \begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix}$$

$$- l_3 \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-n_b) \end{bmatrix} - l_2 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} (y_p(t) - y_m(t))$$

$$\begin{aligned}
&= k_1 w(t) - \left\{ l_2 J + \begin{bmatrix} l_3 & \mathbf{0} \end{bmatrix} \right\} \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} \\
&\quad - l_2 M \begin{bmatrix} y_m(t) \\ y_m(t-1) \\ \vdots \\ y_m(t-n_a) \end{bmatrix} - l_2 \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} (y_p(t) - y_m(t))
\end{aligned}$$

It can be proved numerically that,

$$k_3 = \left\{ l_2 J + \begin{bmatrix} l_3 & \mathbf{0} \end{bmatrix} \right\}_{1 \times (n_b+d)} \quad (5.2.17)$$

$$\& \quad k_2 = l_2 M \quad (5.2.18)$$

where k_2 and k_3 are the coefficients of the GM CDGPC control law in Eq.(5.1.36).

Thus, given the coefficients of the Predictor Seperate control law, l_2 and l_3 , we can calculate k_2 and k_3 knowing only J and M .

5.2.4.2 Transfer Function Form

The second term of the control law in Eq.(5.2.10) is $L_2(z^{-1})\hat{y}_p(t+d|t)$, where $L_2(z^{-1})$ can be written as,

$$L_2(z^{-1}) = l_2 \times \begin{bmatrix} 1 \\ z^{-1} \\ z^{-2} \\ \vdots \\ z^{-n_a} \end{bmatrix}$$

Let l_2 be $l_2 = \begin{bmatrix} l_2^0 & l_2^1 & \cdots & l_2^{n_a-1} & l_2^{n_a} \end{bmatrix}_{1 \times (n_a+1)}$. Therefore,

$$L_2(z^{-1}) = l_2^0 + l_2^1 z^{-1} + \cdots + l_2^{n_a-1} z^{-(n_a-1)} + l_2^{n_a} z^{-n_a} = \sum_{i=0}^{n_a} (l_2^i z^{-i})$$

And therefore,

$$L_2(z^{-1})\hat{y}_p(t+d|t) = l_2^0 \hat{y}_p(t+d|t) + l_2^1 \hat{y}_p(t+d-1|t) + \cdots + l_2^{n_a} \hat{y}_p(t+d-n_a|t)$$

Substituting for each of the prediction terms using Eq.(5.2.15), we get,

$$\begin{aligned}
L_2(z^{-1}) \hat{y}_p(t+d|t) &= l_2^0 E_d B \Delta u(t-1) + l_2^0 F_d y_m(t) + l_2^0 [y_p(t) - y_m(t)] \\
&\quad l_2^1 E_{d-1} B \Delta u(t-2) + l_2^1 F_{d-1} y_m(t) + l_2^1 [y_p(t) - y_m(t)] \cdots + \\
&\quad l_2^{n_a} E_{d-n_a} B \Delta u(t-n_a-1) + l_2^{n_a} F_{d-n_a} y_m(t) + l_2^{n_a} [y_p(t) - y_m(t)] \\
L_2(z^{-1}) \hat{y}_p(t+d|t) &= B \left[\sum_{i=0}^{n_a} l_2^i E_{d-i} z^{-i} \right] \Delta u(t-1) + \left[\sum_{i=0}^{n_a} l_2^i F_{d-i} \right] y_m(t) + L_2(1) [y_p(t) - y_m(t)]
\end{aligned}$$

But from the diophantine equation, we have,

$$E_j = \left(\frac{1 - z^{-j} F_j}{\tilde{A}} \right)$$

Therefore,

$$\begin{aligned}
L_2(z^{-1}) \hat{y}_p(t+d|t) &= B \left[\sum_{i=0}^{n_a} l_2^i z^{-i} \left(\frac{1 - z^{-(d-i)} F_{d-i}}{\tilde{A}} \right) \right] \Delta u(t-1) \\
&\quad + \left[\sum_{i=0}^{n_a} l_2^i F_{d-i} \right] y_m(t) + L_2(1) [y_p(t) - y_m(t)] \\
&= \left(\frac{B}{A} z^{-1} \right) \left[\sum_{i=0}^{n_a} (l_2^i z^{-i}) - z^{-d} \sum_{i=0}^{n_a} (l_2^i F_{d-i}) \right] u(t) \\
&\quad + \left[\sum_{i=0}^{n_a} (l_2^i F_{d-i}) \right] y_m(t) + L_2(1) [y_p(t) - y_m(t)] \\
&= g_m^* u(t) \sum_{i=0}^{n_a} (l_2^i z^{-i}) - g_m \sum_{i=0}^{n_a} (l_2^i F_{d-i}) u(t) \\
&\quad + \sum_{i=0}^{n_a} (l_2^i F_{d-i}) y_m(t) + L_2(1) [y_p(t) - y_m(t)] \\
&= g_m^* u(t) L_2(z^{-1}) + \sum_{i=0}^{n_a} (l_2^i F_{d-i}) [y_m(t) - y_m(t)] + L_2(z^{-1}) [y_p(t) - y_m(t)] \\
&= L_2(z^{-1}) g_m^* u(t) + L_2(1) [y_p(t) - y_m(t)]
\end{aligned}$$

where $g_m^* = \frac{B}{A} z^{-1}$. Therefore,

$$\hat{y}_p(t+d|t) = g_m^* u(t) + \frac{L_2(1)}{L_2(z^{-1})} [y_p(t) - y_m(t)] \quad (5.2.19)$$

which is simply the Predictor of the CDGPC.

5.2.4.3 At Steady State: Case ($d > n_a$)

After the control loop reaches steady state, $\Delta u(t) = \Delta u(t-1) = \Delta u(t-2) = \dots \Delta u(t-(n_b+d)) = 0$, and $y_p(t) = y_p(t-1) = \dots = y_p(t-n_a) = w(t)$. And so, at steady state, the matrix form of the control law for the case ($d > n_a$), Eq.(5.2.10) gives,

$$0 = k_1 w(t) - l_2 \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix}_{ss} \quad (5.2.20)$$

At steady state, the prediction vector for the case ($d > n_a$) in Eq.(5.2.16) becomes,

$$\begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix}_{ss} = M \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} y_{mss} + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{(n_a+1) \times 1} w(t) - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{(n_a+1) \times 1} y_{mss}$$

But the matrix M can be numerically shown to be a matrix with one eigenvector as $\begin{bmatrix} 1 & \dots & 1 \end{bmatrix}$ and corresponding eigenvalue 1.

$$\therefore \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix}_{ss} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{(n_a+1) \times 1} w(t) \quad (5.2.21)$$

Putting Eq.(5.2.21) into Eq.(5.2.20),

$$k_1 = \text{sum}(l_2) \quad (5.2.22)$$

5.2.5 Control Structure & Closed Loop Transfer Function: Predictor Separate Case ($d > n_a$)

5.2.5.1 Primary Controller

The Control Law for case ($d > n_a$) (Eq.(5.2.6)) in the transfer function form is,

$$\Delta u(t) = k_1 w(t) - L_2(z^{-1})\hat{y}_p(t+d|t) - L_3(z^{-1})z^{-1}\Delta u(t)$$

Rearranging its terms,

$$\Rightarrow \Delta u(t) = \frac{k_1}{[1 + L_3 z^{-1}]} w(t) - \frac{L_2}{[1 + L_3 z^{-1}]} \hat{y}_p(t + d|t) \quad (5.2.23)$$

The overall control loop is depicted using Eq.(5.2.23) and Eq.(5.2.19), as shown in Figure(2.3.1).

Figure 5.2.1: Unconstrained CDGPC control loop - Predictor Separate Case $d > n_a$

We can proceed still further: Substituting Eq.(5.2.19) into Eq.(5.2.23), we get the overall control law in transfer function form,

$$\begin{aligned} [1 + L_3 z^{-1}] \Delta u(t) &= k_1 w(t) - L_2 \left[g_m^* u(t) + \frac{L_2(1)}{L_2(z^{-1})} (y_p(t) - y_m(t)) \right] \\ [\Delta(1 + L_3 z^{-1}) + L_2 g_m^* - L_2(1) g_m] u(t) &= k_1 w(t) - L_2(1) y_p(t) \end{aligned}$$

But from Eq.(5.1.35), we have $L_2(1) = k_1$,

$$\begin{aligned} [\Delta(1 + L_3 z^{-1}) + L_2 g_m^* - k_1 g_m] u(t) &= k_1 \{w(t) - y_p(t)\} \\ \equiv u(t) &= C_{cd}(z^{-1}) \{w(t) - y_p(t)\} \end{aligned} \quad (5.2.24)$$

where,

$$C_{cd}(z^{-1}) = \frac{k_1}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* - k_1 g_m} \quad (5.2.25)$$

The controller C_{cd} in Eq.(5.2.25) can be proved to be equivalent to the controller C_{cd} derived using the GMethod in Eq.(5.1.44).

5.2.5.2 Closed Loop Transfer Function

$$y_p(t) = g_p(z^{-1}) u(t) + d_o(t) \quad (5.2.26)$$

Substituting for u from Eq.(5.2.24) into Eq.(5.2.26) and simplifying,

$$y_p(t) = T_{cd} \times w(t) + S_{cd} \times d_o(t) \quad (5.2.27)$$

where S_{cd} is the sensitivity & T_{cd} is the Complementary-Sensitivity of the control loop.

$$T_{cd} = \frac{C_{cd}(z^{-1})g_p}{1 + C_{cd}(z^{-1})g_p} = \frac{k_1g_p}{\Delta(1 + L_3z^{-1}) + L_2g_m^* + k_1(g_p - g_m)} \quad (5.2.28)$$

$$S_{cd} = \frac{1}{1 + C_{cd}(z^{-1})g_p} = 1 - T_{cd} \quad (5.2.29)$$

Eq.(5.2.27) is the Closed Loop Transfer Function of the CDGPC control loop.

5.2.5.3 Alternative look at the Control Law:

The control law in the transfer function form is,

$$[\Delta(1 + L_3z^{-1}) + L_2g_m^* - k_1g_m] u(t) = k_1(w(t) - y_p(t))$$

Substituting for $y_p(t)$ using Eq.(5.2.26) gives,

$$[\Delta(1 + L_3z^{-1}) + L_2g_m^* + L_2(g_p - g_m)] u(t) = k_1(w(t) - d_o(t)) \quad (5.2.30)$$

Eq.(5.2.30) shows that the control law behaves the same for set-point and for external disturbances $d_o(t)$, and is thus One Degree of Freedom.

5.3 Properties of the CDGPC

5.3.1 Robustness of CDGPC

In this section, the Robustness of the conventional GPC, SPGPC and the CDGPC will be compared. The nominal primary loop tracking transfer functions and the Robustness Limits are given below:

The **primary loop's** tracking transfer function of the GPC by Predictor Separate method is,

$$T = \frac{g_p L_2 F_r}{\Delta(1 + L_3z^{-1}) + L_2g_m^* + L_2F_r(g_p - g_m)}$$

whose nominal version is thus,

$$T^{nom} = \frac{L_2F_r g_m}{\Delta(1 + L_3z^{-1}) + L_2g_m^*}$$

The corresponding **Robustness Limit** is,

$$[\Delta_{lim}^*(\omega)]_{gpc} = \left| \frac{\Delta(1 + L_3 z^{-1}) + L_2 g_m^*}{L_2 g_m} \right| \times \frac{1}{|F_r|} \quad (5.3.1)$$

For details, see Appendix(v).

T_{sp} is the tracking transfer function of the SPGPC,

$$T_{sp} = \frac{g_p L_2}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* + L_2(g_p - g_m)}$$

whose nominal version is,

$$T_{sp}^{nom} = \frac{g_p L_2}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^*}$$

The Robustness limit of the SPGPC is,

$$[\Delta_{lim}^*(\omega)]_{spgpc} = \left| \frac{\Delta(1 + L_3 z^{-1}) + L_2 g_m^*}{L_2 g_m} \right| \times 1 \quad (5.3.2)$$

For details on the SPGPC, see Section(1.5) of Chapter(1).

T_{cd} is the tracking transfer function of the CDGPC,

$$T_{cd} = \frac{k_1 g_p}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* + k_1(g_p - g_m)}$$

whose nominal version is,

$$T_{cd}^{nom} = \frac{k_1 g_m}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^*}$$

The Robustness limit of the CDGPC is,

$$[\Delta_{lim}^*(\omega)]_{cdgpc} = \left| \frac{\Delta(1 + L_3 z^{-1}) + L_2 g_m^*}{L_2 g_m} \right| \times \left| \frac{L_2}{k_1} \right| \quad (5.3.3)$$

Some important observations follow:

1. The Robustness Limits of the GPC, the SPGPC and CDGPC are identical except for the multiplying terms $\frac{1}{|F_r|}$, 1 and $\left| \frac{L_2}{k_1} \right|$ respectively. However, the nominal tracking transfer functions of the GPC, the SPGPC and the CDGPC are the same.
2. Since only F_r is the term dependent on the d , and k_1 & L_2 are independent of d , the Robustness Limits of the SPGPC and CDGPC are independent of the delay d .

It will be shown using simulations and experiments with FOPTD processes that the term

$\left| \frac{L_2}{k_1} \right|$ offers greater robustness than the terms 1 (SPGPC) and $\frac{1}{|F_r|}$ (GPC). It will also be shown using examples that with increasing process delay the robustness of the GPC decreases till it reaches a limit at a very large delay.

5.3.2 Nominal and Internal Stability

With the conventional GPC, internal stability is guaranteed even for unstable processes. This is due to the structure of the GPC itself which ensures that no unstable pole-zero cancellation occurs in the open loop.[52] This is possible because of the unique structure of the optimal predictor filter, F_r , which prevents unstable pole-zero cancellation in the open loop. This means that any variant of the GPC that modifies the filter structure cannot guarantee nominal stability for the control of open loop unstable process. Thus, both the CDGPC and the SPGPC cannot be utilized for the control of unstable systems.

5.3.3 CDGPC with N^* tuning for FOPTD processes

For the stable FOPTD process, the control law of the conventional GPC is, (Eq.(2.1.22))

$$\Delta u(t) = k_1 w(t) - L_2 (z^{-1}) \hat{y}_p(t + d|t) - l_3 \Delta u(t - 1)$$

where the prediction is,

$$\hat{y}_p(t + d|t) = g_m^* u(t) + F_r(z^{-1}) (y_p(t) - y_m(t))$$

where F_r is the predictor-filter of the conventional GPC. The control law of the CDGPC for stable FOPTD processes is also the same except that the prediction is,

$$\hat{y}_p(t + d|t) = g_m^* u(t) + \frac{k_1}{L_2(z^{-1})} (y_p(t) - y_m(t))$$

The same applies to the SPGPC with a predictor filter of just 1. For all the three GPC variants the nominal characteristic equation is identical,

$$\Delta (1 + l_3 z^{-1}) + L_2 g_m^* = 0$$

This means that the N^* tuning method can be used to tune the CDGPC and the SPGPC.

The nominal tracking transfer function of the CDGPC loop at $N = N_l^*$ and $\lambda = \lambda^*$ will be,

$$\therefore T_{cd}(z) / (\lambda^* N_l^*) = \left(\frac{(1 - ak)^2 z^{-1}}{(1 - akz^{-1})^2} \right) z^{-d}$$

and the robustness of the CDGPC loop will be,

$$\Delta_{lim}^*(\omega)/(\lambda^* N_l^*) = \frac{1}{|T^{p=m}(e^{j\omega T})|_{(\lambda^* N_l^*)}} = \left(1 + \frac{4\gamma k}{(1 - \gamma k)^2} \sin^2\left(\frac{\omega T}{2}\right)\right)$$

Thus, when for a k value closed to 1, the robustness will be high while for a k value close to 0, the performance will be high. For a detailed explanation, see Chapter 2, Section (2.3). It should be noted that the robustness limit of the conventional GPC at $N = N_l^*$ and $\lambda = \lambda^*$ in Eq.(2.3.6) had to contend with the effect of the prefilter as well. For the CDGPC, the prefilter is just 1 and the mechanism of trading off performance with robustness is without hindrance.

The following section will deal with simulations and experimental results of the GPC, SPGPC and CDGPC for stable FOPTD processes utilizing the N^* tuning.

5.4 Simulations and Experiments

5.4.1 Example 1 - Delay Uncertainty

Figure (5.4.1) shows the Robustness Limits of the GPC, SPGPC and CDGPC loops for the control of the system whose nominal model is,

$$g_m = \frac{0.09516z^{-1}}{1 - 0.9048z^{-1}}z^{-20}$$

which is the discrete-time version of $g_m(s) = \frac{1}{2s+1}e^{-4s}$, sampled at $T = 0.2$. The GPC, SPGPC & CDGPC were tuned using the N^* method with $k = 0.8$ ($N_1 = d + 1$, $N_2 = d + \text{ceil}(N_l^*) = d + 12$, $\lambda = \lambda^* = 3.853$, $N_u = 1$)

1. The model is assumed to have an erroneous delay estimate of $\Delta d = 2$, that is, the plant $g_p(z) = g_m(z) \times z^{-(\Delta d)}$. Therefore, the multiplicative uncertainty Δ^* corresponding to g_p is,

$$\Delta^*(e^{j\omega T}) = \frac{g_p(z)}{g_m(z)} - 1 = (e^{-j2\omega T} - 1)$$

the magnitude of which is also shown in Figure (5.4.1). It can thus be established that *(keeping the tuning the same) the CDGPC offers the highest robustness while GPC provides the lowest, for this large delay system.* Moreover, for the given example, the GPC loop is NOT robustly stable.

2. The Figure (5.4.2) shows the closed loop responses under nominal conditions and

also in the presense of delay uncertainty, that is, the plant g_p is,

$$g_p(s) = \frac{1}{2s+1} e^{-(L+dL)s}$$

where $dL = +0.4$ for $L = 4$. As can be seen, the GPC loop is unstable while the SPGPC and CDGPC loops are stable, for the same delay uncertainty, with the same tuning parameters. It should also be noted the nominal disturbance rejection response is most aggressive in the least robust GPC, and less aggressive with the SPGPC, and identical to the tracking in the CDGPC.

3. The Figure (5.4.3) shows the Robustnes limits of the conventional GPC at different process delays, $L = 1, 2, 4, 8$. As mentioned before, the Robustness of the GPC varies with the delay; higher the delay value, lower the robustness. However, the robustness does not vary much for any increments at large delay values; that is, the difference in robustness between $L = 1$ and $L = 2$ is drastic compared to the difference in robusntess between $L = 4$ and $L = 8$. The robustness of the SPGPC and the CDGPC which are delay independent are also shown in Figure (5.4.3) for comparison. It is concluded that the robustness of the GPC is always the lowest and will be the same as that of the SPGPC only for a delay of 0.

Figure 5.4.1: SPGPC, CDGPC & GPC Robustness Limits & delay multiplicative uncertainty $|\Delta^*(e^{j\omega})|$ vs ω

Figure 5.4.2: Closed Loop Responses of GPC, SPGPC & CDGPC for delay uncertainty (A setpoint of 1 was applied at $t = 0s$ and an output disturbance of -1 was applied at $t = 25s$)

Figure 5.4.3: Robustness Limits of GPC, SPGPC & CDGPC for different process delays (Robustness of SPGPC and CDGPC are independent of delay)

5.4.2 Example 2 - Worst-Case Uncertainty

In this example, the same nominal model, sampling time and tuning parameters are used (The GPC, SPGPC, CDGPC were tuned using the N^* method with $k = 0.8$ ($N_1 = d + 1$, $N_2 = d + \text{ceil}(N_1^*) = d + 12$, $\lambda = \lambda^* = 3.853$, $N_u = 1$)).

1. In an alternate scenario, if we assume a 20% worst-case parametric mismatch, that is, that the plant is $g_{pw}(s) = \frac{1.2}{1.6s+1}e^{-4.8s}$, then the multiplicative uncertainty corresponding to g_{pw} is:

$$\Delta_w^*(e^{j\omega T}) = \frac{g_{pw}(z)}{g_m(z)} - 1$$

Its magnitude, $|\Delta_w^*(e^{j\omega T})|$, is shown in Figure (5.4.4) alongside the Robustness Limits. Obviously, neither the GPC nor the SPGPC with the given tuning parameters is robustly stable given a 20% parametric uncertainty. But the CDGPC is *just* robustly stable for 20% worst-case parametric mismatch.

2. The Figure (5.4.5) shows the nominal and the 20% worst-case mismatch closed loop responses of the GPC, CDGPC and the SPGPC. Only the CDGPC loop is stable under 20% worst-case parametric uncertainty, for the given tuning.

Figure 5.4.4: SPGPC, CDGPC & GPC Robustness Limits & worst-case multiplicative uncertainty $|\Delta_w^*(e^{j\omega T})|$

Figure 5.4.5: Closed Loop Responses of GPC, CDGPC & SPGPC for worst-case uncertainty (A setpoint of 1 was applied at $t = 0s$ and an output disturbance of -1 was applied at $t = 25s$)

5.4.3 Level Control in a Single Tank System

The CDGPC, SPGPC and the conventional GPC with N^* -tuning were implemented for level control of a single tank system, shown in Figure (2.4.3). The details regarding the system and its identification have already been discussed in Section (2.4.3). For this experiment, however, the system was modified to have an extra delay of $2.7s$ so that the overall process was,

$$\frac{3.4715e^{-3s}}{32.613s + 1}$$

This was necessary in order to compare the robustness of the different GPC types because, as mentioned before, for a very small delay the robustness of the conventional GPC is almost the same as that of the SPGPC. The three types of GPC were implemented (with a sampling time of $T = 0.1s$) on the water tank system. The system was brought to a steady state of $13.6cm$ by applying an input voltage of $6V$. Then, the loop was closed

with a deviation setpoint of 0. At a time $t = 30s$ after data logging starts, a deviation setpoint of $3cm$ was applied and at $t = 180s$, an input disturbance of $+4V$ was applied to the closed loop. The closed loop responses of all three GPC types are shown in Figure (5.4.6). It can be seen that all three types have the exact same tracking response but the disturbance rejection of the GPC is the most aggressive while CDGPC is the most conservative.

Figure 5.4.6: Closed Loop Responses of SPGPC, CDGPC & GPC under nominal condition ($k = 0.97$, $N = \text{ceil}(N_t^*)$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)

In order to further validate that the CDGPC offered the highest robustness, the experiment was repeated by introducing model mismatch. That is, the model used for controller design and tuning was taken as,

$$\frac{3.4715e^{-1.5s}}{32.613s + 1}$$

where the delay is wrong by 50%. The closed loop responses from this experiment are

shown in Figure (5.4.7). As can be observed from the tracking responses, the CDGPC offers the most robustness. The chief disadvantage of the CDGPC is that the disturbance rejection is slow.

Figure 5.4.7: Closed Loop Responses of GPC, CDGPC & SPGPC under a delay mismatch of $1.5s$ ($k = 0.97$, $N = \text{ceil}(N_l^*)$) (A setpoint of $3cm$ was applied at $t = 30s$ and an input disturbance of $+4v$ was applied at $t = 180s$)

5.5 Chapter Conclusions

The main contribution of this chapter is the study of the CDGPC's structure and its robustness in comparison to the existing GPC variants, namely, the conventional GPC and the SPGPC. These aspects have not been explored or reported in existing literature.

- Of the three variants of GPC, namely, the conventional GPC, the Smith Predictor GPC (SPGPC) and the Constant Disturbance GPC (CDGPC), the CDGPC has the greatest Robustness. The Robustness of the CDGPC and the SPGPC do not

depend on the process delay while the Robustness of the GPC does; the higher the delay the lower the robustness.

- The CDGPC has the same structure as the GPC and the SPGPC. The difference is only in the structure of the predictor's filter which in the case of the CDGPC was found to be $\frac{k_1}{L_2(z^{-1})}$. The chief disadvantage of the CDGPC is that because the CDGPC's predictor filter is different from that of the conventional GPC, the CDGPC cannot offer internal stability for the control of open loop unstable systems. This is also true of the SPGPC.
- Because of the way that the structure of the predictor filter in the CDGPC is, the overall closed loop has only One-Degree-of-Freedom.
- As mentioned in Chapter (1), the CDGPC is the equivalent to the Dynamic Matrix Control (DMC) which was introduced by Cutler and Ramaker in 1979 [17]. This is obvious because they utilize the same disturbance model. The main difference is that the former uses the transfer function model while the latter uses a step response model. Using the transfer function model is definitely more advantageous as it reduces the number of parameters to work with.

Chapter 6

RNGA-based Decentralized GPC

Decentralized Control schemes remain dominant in the industry for two reasons. It requires fewer controllers and therefore fewer parameters to tune and is simpler to understand. The second reason is that loop failure tolerance can be designed into the system from the start. In decentralized control, RGA[12, 39] (Relative Gain Array) based loop pairing and PID tuning are popular because of the simple calculations involved. RGA based loop pairing rules are usually used in combination with the Niederlinski index[46]. The RGA-NI combination is a necessary and sufficient condition for 2×2 processes but only a necessary condition of 3×3 and higher dimensional systems [19, 26]. In order to overcome the limitations of the RGA method which only uses steady state information, the dynamic version called the DRGA using transfer function models was developed. Xiong et.al. proposed the REGA (Relative Effective Gain Array)[74, 72] as a method that works similar to the RGA method but incorporates dynamic information also. In particular, it uses the critical frequency but this introduced complications as different methods of selecting the critical frequency led to conflicting loop pairing suggestions for the same MIMO process. Finally, the RNGA (Relative Normalized Gain Array)[24] was proposed as an alternative to the REGA. RNGA uses average residence time information in its calculations and hence it is very easy to use. However, without the assumption of perfect control of all loops other than the one under consideration, the DRGA method is controller dependent.

The RNGA-decentralized method has been proposed for stable FOPTD and SOPTD processes with PID as the controller. However, in this work, the RNGA-based loop pairing and controller design rules will be used in conjunction with the GPC (in particular, the CDGPC with N^* tuning) instead of the PID. And because the N^* method is only applicable to FOPTD processes, the work in this chapter will be restricted to only these. MIMO processes that involve elements which are not FOPTD will be simplified to FOPTD using linear least squares fitting of step response data. This is certainly *not* a disadvantage

because from a practical point of view, higher order models are usually simplified to lower order models for the sake of convenience[24]. Many works in the literature[61, 8] can attest to the convenience and practicality of the simple FOPTD model.

The chapter is sectioned as follows:

- Section (6.1) will introduce some of the existing methods of interaction analysis which help in deciding the most suitable loop pairing. These include the DRGA (Dynamic Relative Gain Array)[71], RGA (Relative Gain Array)[12, 39], NI (Neiderlinski Index), REGA (Relative Effective Gain Array)[74, 72] and the RNGA (Relative Normalized Gain Array)[24].
- Section (6.2) will introduce the seminal work of Cai et.al., the RNGA-based decentralized PID design method[24]. The procedure involves loop pairing by RNGA, RGA and NI analysis, generating the RNGA-based Effective Transfer Functions (ETFs) for the diagonal elements of the MIMO process and finally, designing PID controllers for those ETFs.
- Section (6.3) will present the novel RNGA-based decentralized GPC. The CDGPC in combination with the N^* tuning method will be shown to be successful in a decentralized setting. The RNGA-based PID and the RNGA-based GPC methods will be compared.

6.1 Multivariable Loop Pairing Methods

6.1.1 Dynamic Relative Gain Array (DRGA)

Consider a 2x2 Process $G(s)$

$$G(s) = \begin{bmatrix} g_{11}(s) & g_{12}(s) \\ g_{21}(s) & g_{22}(s) \end{bmatrix}$$

$$\begin{bmatrix} y_1(s) \\ y_2(s) \end{bmatrix} = \begin{bmatrix} g_{11}(s) & g_{12}(s) \\ g_{21}(s) & g_{22}(s) \end{bmatrix} \begin{bmatrix} m_1(s) \\ m_2(s) \end{bmatrix} \quad (6.1.1)$$

$\lambda_{ij}(s)$ is defined as the **Dynamic Relative Gain** between the output $y_i(s)$ and input $m_j(s)$,

$$\lambda_{ij}(s) = \frac{\left(\frac{\partial y_i}{\partial m_j} \right)_{\text{all loops open}}}{\left(\frac{\partial y_i}{\partial m_j} \right)_{\text{all loops closed except loop } y_i-m_j}} \quad (6.1.2)$$

where the numerator is the gain between $y_i(s)$ and $m_j(s)$ when all the loops are open and the denominator is the gain between $y_i(s)$ and $m_j(s)$ when all the loops are closed

except the loop $y_i(s) - m(s)$.

For the 2x2 process $G(s)$ in Eq.(6.1.1), the different combinations of input and output will yield a *Dynamic Relative Gain Array* of the form,

$$DRGA = \Lambda(s) = \begin{bmatrix} \lambda_{11}(s) & \lambda_{12}(s) \\ \lambda_{21}(s) & \lambda_{22}(s) \end{bmatrix}$$

6.1.1.1 Deriving DRGA from first principles

From Eq.(6.1.1),

$$y_1(s) = g_{11}(s)m_1(s) + g_{12}(s)m_2(s) \quad (6.1.3)$$

$$y_2(s) = g_{21}(s)m_1(s) + g_{22}(s)m_2(s) \quad (6.1.4)$$

Deriving $\lambda_{11}(s)$: To derive $\lambda_{11}(s)$, from Eq.(6.1.2),

$$\lambda_{11}(s) = \frac{\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{all loops open}}}{\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{loop } m_2 \text{ closed}}} \quad (6.1.5)$$

Differentiating Eq.(6.1.3) will give the numerator,

$$\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{all loops open}} = g_{11}(s)$$

To derive the denominator, loop $y_2 - m_2$ must be closed, as shown in Figure (6.1.1).

Figure 6.1.1: Loop $y_2 - m_2$ closed

This means that in Eq.(6.1.5), $y_2(s)$ is controlled with $m_2(s)$ and any increase in $m_1(s)$ will be countered by the control of the manipulated variable $m_2(s)$, so as to keep $y_2(s)$ always at 0. So putting $y_2(s) = 0$ in Eq.(6.1.4), the value that $m_2(s)$ will take can

be obtained.

$$\begin{aligned} 0 &= g_{21}(s)m_1(s) + g_{22}(s)m_2(s) \\ m_2(s) &= -\frac{g_{21}(s)}{g_{22}(s)}m_1(s) \end{aligned}$$

Using this in Eq.(6.1.3) gives,

$$\begin{aligned} y_1(s) &= g_{11}(s)m_1(s) + g_{12}(s) - \frac{g_{21}(s)}{g_{22}(s)}m_1(s) \\ &= g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right) m_1(s) \end{aligned}$$

Differentiating this will give the denominator to Eq.(6.1.5) as,

$$\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{loop } m_2 \text{ closed}} = g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$$

Therefore,

$$\lambda_{11}(s) = \frac{1}{\left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)}$$

Deriving $\lambda_{12}(s)$: Similarly, to derive $\lambda_{12}(s)$, from Eq.(6.1.2) we can write,

$$\lambda_{12}(s) = \frac{\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{all loops open}}}{\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{loop } m_1 \text{ closed}}} \quad (6.1.6)$$

Differentiating Eq.(6.1.3) will give us the numerator,

$$\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{all loops open}} = g_{12}(s)$$

Again, for the denominator, loop $y_2 - m_1$ is closed, as shown in Figure (6.1.2).

Figure 6.1.2: Loop $y_2 - m_1$ closed

This means that in Eq.(6.1.5), $y_2(s)$ is controlled with $m_1(s)$ and any increase in $m_2(s)$ will be countered by the control of the manipulated variable $m_1(s)$, so as to keep $y_2(s)$ always at 0. So putting $y_2(s) = 0$ in Eq.(6.1.4), we can get the value $m_1(s)$ will take.

$$\begin{aligned} 0 &= g_{21}(s)m_1(s) + g_{22}(s)m_2(s) \\ m_1(s) &= -\frac{g_{22}(s)}{g_{21}(s)}m_2(s) \end{aligned}$$

Using this in Eq.(6.1.3) gives,

$$y_1(s) = g_{12}(s) \left(1 - \frac{g_{11}(s)g_{22}(s)}{g_{12}(s)g_{21}(s)} \right) m_2(s)$$

Differentiating this will give us the denominator to Eq.(6.1.6) as

$$\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{loop } m_1 \text{ closed}} = g_{12}(s) \cdot \left(1 - \frac{g_{11}(s)g_{22}(s)}{g_{12}(s)g_{21}(s)} \right)$$

Therefore,

$$\lambda_{12}(s) = \frac{1}{\left(1 - \frac{g_{11}(s)g_{22}(s)}{g_{12}(s)g_{21}(s)} \right)}$$

Deriving $\lambda_{21}(s)$ and $\lambda_{22}(s)$: Following similar procedures, we can devise experiments where we can close loop $y_1 - m_2$ to obtain $\lambda_{21}(s)$ {as shown in Figure (6.1.3)} and close $y_1 - m_1$ {as shown in Figure (6.1.4)} to obtain $\lambda_{22}(s)$ as,

Figure 6.1.3: Loop $y_1 - m_2$ closed Figure 6.1.4: Loop $y_1 - m_1$ closed

$$\lambda_{21}(s) = \lambda_{12}(s) = \frac{1}{\left(1 - \frac{g_{11}(s)g_{22}(s)}{g_{12}(s)g_{21}(s)}\right)}$$

$$\lambda_{22}(s) = \lambda_{11}(s) = \frac{1}{\left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)}$$

Taking

$$\xi(s) = \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}$$

we can write,

$$\lambda_{11}(s) = \lambda_{22}(s) = \frac{1}{(1 - \xi(s))}$$

$$\lambda_{21}(s) = \lambda_{12}(s) = \frac{-\xi(s)}{(1 - \xi(s))}$$

Therefore,

$$DRGA = \Lambda(s) = \begin{bmatrix} \lambda_{11}(s) & \lambda_{12}(s) \\ \lambda_{21}(s) & \lambda_{22}(s) \end{bmatrix} = \begin{bmatrix} \frac{1}{(1-\xi(s))} & \frac{-\xi(s)}{(1-\xi(s))} \\ \frac{-\xi(s)}{(1-\xi(s))} & \frac{1}{(1-\xi(s))} \end{bmatrix}$$

If we define, $\lambda(s) = \frac{1}{(1-\xi(s))}$, then we can rewrite the DRGA as,

$$\Lambda(s) = \begin{bmatrix} \lambda(s) & 1 - \lambda(s) \\ 1 - \lambda(s) & \lambda(s) \end{bmatrix}$$

where $\lambda(s)$ is called the **Relative Dynamic Gain Parameter**.

6.1.1.2 DRGA by First Principles

The DRGA can also be found out by the Matrix operation,

$$\Lambda(s) = G(s) \otimes G(s)^{-T}$$

where $\otimes$ is the Hadamard or Schur Product representing the element by element multiplication of the matrices $G(s)$ and $G(s)^{-T}$.

$$(DRGA) \Lambda(s) = G \otimes G^{-T}$$

$$\begin{aligned} \Lambda(s) &= \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \otimes \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-T} \\ &= \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \otimes \frac{1}{|G|} \begin{bmatrix} g_{22} & -g_{21} \\ -g_{12} & g_{11} \end{bmatrix} \\ &= \frac{1}{|G|} \begin{bmatrix} g_{11}g_{22} & -g_{12}g_{21} \\ -g_{21}g_{12} & g_{22}g_{11} \end{bmatrix} \\ &= \frac{1}{g_{11}g_{22} - g_{12}g_{21}} \begin{bmatrix} g_{11}g_{22} & -g_{12}g_{21} \\ -g_{21}g_{12} & g_{22}g_{11} \end{bmatrix} \end{aligned}$$

$$\Lambda = \frac{1}{1 - \frac{g_{12}g_{21}}{g_{11}g_{22}}} \begin{bmatrix} 1 & \frac{-g_{12}g_{21}}{g_{11}g_{22}} \\ \frac{-g_{12}g_{21}}{g_{11}g_{22}} & 1 \end{bmatrix}$$

$$\text{Taking } \frac{g_{12}g_{21}}{g_{11}g_{22}} = \xi(s)$$

$$\Lambda(s) = \frac{1}{1 - \xi(s)} \cdot \begin{bmatrix} 1 & -\xi(s) \\ -\xi(s) & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{1-\xi(s)} & \frac{-\xi(s)}{1-\xi(s)} \\ \frac{-\xi(s)}{1-\xi(s)} & \frac{1}{1-\xi(s)} \end{bmatrix}$$

6.1.1.3 Properties of DRGA

Some points of significance regarding DRGA are:

1. The elements of the DRGA across any row, or down any column, sum to 1 for $s \in (0, \infty)$
2. The value of $\lambda_{ij}(s)$ is a measure of the dynamic interaction expected in the i^{th} loop if its output y_i is paired with m_j .
3. If $g_{ij}^*(s)$ represents loop i dynamic gain when all other loops are closed, and $g_{ij}(s)$ is the normal, open-loop gain, then by the definition of *Dynamic Relative Gain* in Eq.(6.1.3),

$$g_{ij}^*(s) = \frac{1}{\lambda_{ij}(s)} g_{ij}(s)$$

$\frac{1}{\lambda_{ij}(s)}$ tells us by what factor the open-loop gain between $y_i(s)$ and $m_j(s)$ will be altered if other loops are closed.

For example, in the 2×2 case, $g_{11}^*(s)$ will be the transfer function between output $y_1(s)$ and input $m_1(s)$ when loop 2 is closed, that is, when $y_2(s)$ controlled with $m_2(s)$.

$$g_{11}^*(s) = \frac{1}{\lambda_{11}(s)} g_{11}(s) = g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$$

Similarly, $g_{12}^*(s)$ will be the transfer function between output $y_1(s)$ and input $m_2(s)$ when loop 2 is closed, that is, when $y_2(s)$ controlled with $m_1(s)$.

$$g_{12}^*(s) = \frac{1}{\lambda_{12}(s)} g_{12}(s) = g_{12}(s) \left(1 - \frac{g_{11}(s)g_{22}(s)}{g_{12}(s)g_{21}(s)} \right)$$

The same goes for $g_{21}^*(s) = \frac{1}{\lambda_{21}(s)} g_{21}(s)$ {transfer function between output $y_2(s)$ and input $m_1(s)$ when loop 1 is closed ($y_1(s)$ controlled with $m_2(s)$)} and $g_{22}^*(s) = \frac{1}{\lambda_{22}(s)} g_{22}(s)$ {transfer function between output $y_2(s)$ and input $m_2(s)$ when loop 1 is closed ($y_1(s)$ controlled with $m_1(s)$)}.

6.1.2 Relative Gain Array (RGA)

RGA is the Steady State version of DRGA. The RGA can be derived in the same ways as the DRGA by the *first principles method*, by using the Steady State Model of the same

system.

$$y_1 = k_{11}m_1 + k_{12}m_2 \quad (6.1.7)$$

$$y_2 = k_{21}m_1 + k_{22}m_2 \quad (6.1.8)$$

and using the steady state version of the definition of *Dynamic Relative Gain* from Eq.(6.1.2),

$$\lambda_{ij} = \frac{\left(\frac{\partial y_i}{\partial m_j} \right)_{\text{all loops open}}}{\left(\frac{\partial y_i}{\partial m_j} \right)_{\text{all loops closed except loop } m_j}} \quad (6.1.9)$$

6.1.2.1 First Principles Method

To derive λ_{11} , from Eq.(6.1.9) we can write,

$$\lambda_{11} = \frac{\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{all loops open}}}{\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{loop } m_2 \text{ closed}}} \quad (6.1.10)$$

Differentiating Eq.(6.1.7) will give us the numerator,

$$\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{all loops open}} = k_{11}$$

As for the denominator, we must first take into consideration that loop $y_2 - m_2$ is closed {see Figure (6.1.1)}. *This means that any increase in $m_1(s)$ will be countered by the control of the manipulated variable $m_2(s)$, so as to keep $y_2(s)$ always at 0.* ($y_2(s)$ controlled with $m_2(s)$). So at steady state $y_2 = 0$ and putting this in Eq.(6.1.8), we can get the value m_2 will take.

$$\begin{aligned} 0 &= k_{21}m_1 + k_{22}m_2 \\ m_2 &= -\frac{k_{21}}{k_{22}}m_1 \end{aligned}$$

Using this in Eq.(6.1.7) gives,

$$\begin{aligned} y_1 &= k_{11}m_1 + k_{12} \times -\frac{k_{21}}{k_{22}}m_1 \\ &= k_{11} \left(1 - \frac{k_{12}k_{21}}{k_{11}k_{22}} \right) m_1 \end{aligned}$$

Differentiating this will give us the denominator to Eq.(6.1.10) as

$$\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{loop } m_2 \text{ closed}} = k_{11} \left(1 - \frac{k_{12}k_{21}}{k_{11}k_{22}} \right)$$

Therefore,

$$\lambda_{11} = \frac{1}{\left(1 - \frac{k_{12}k_{21}}{k_{11}k_{22}} \right)}$$

To derive λ_{12} , from Eq.(6.1.9) we can write,

$$\lambda_{12} = \frac{\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{all loops open}}}{\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{loop } m_2 \text{ closed}}} \quad (6.1.11)$$

Differentiating Eq.(6.1.7) will give us the numerator,

$$\left(\frac{\partial y_1}{\partial m_2} \right)_{\text{all loops open}} = k_{12}$$

Again, for the denominator, we must first take into consideration that loop $y_2 - m_1$ is closed {see Figure (6.1.2)}. *This means that any increase in $m_2(s)$ will be countered by the control of the manipulated variable $m_1(s)$, so as to keep $y_2(s)$ always at 0.* ($y_2(s)$ controlled with $m_1(s)$). So at steady state $y_2 = 0$ and putting this in Eq.(6.1.8), we can get the value m_1 will take.

$$\begin{aligned} 0 &= k_{21}m_1 + k_{22}m_2 \\ m_1 &= -\frac{k_{22}}{k_{21}}m_2 \end{aligned}$$

Using this in Eq.(6.1.7) gives,

$$\begin{aligned} y_1 &= k_{11} \times -\frac{k_{22}}{k_{21}}m_2 + k_{12}m_2 \\ &= k_{12} \left(1 - \frac{k_{11}k_{22}}{k_{12}k_{21}} \right) m_2 \end{aligned}$$

Differentiating this will give us the denominator to Eq.(6.1.11) as

$$\left(\frac{\partial y_1}{\partial m_1} \right)_{\text{loop } m_2 \text{ closed}} = k_{12} \left(1 - \frac{k_{11}k_{22}}{k_{12}k_{21}} \right)$$

Therefore,

$$\lambda_{12} = \frac{1}{\left(1 - \frac{k_{11}k_{22}}{k_{12}k_{21}}\right)}$$

Following similar procedures, {see Figures (6.1.3) and (6.1.4)} we can obtain λ_{21} and λ_{22} as

$$\lambda_{21} = \lambda_{12} = \frac{1}{\left(1 - \frac{k_{11}k_{22}}{k_{12}k_{21}}\right)}$$

$$\lambda_{22} = \lambda_{11} = \frac{1}{\left(1 - \frac{k_{12}k_{21}}{k_{11}k_{22}}\right)}$$

Taking

$$\xi = \frac{k_{12}k_{21}}{k_{11}k_{22}}$$

we can write,

$$\lambda_{11} = \lambda_{22} = \frac{1}{(1 - \xi)}$$

$$\lambda_{21} = \lambda_{12} = \frac{-\xi}{(1 - \xi)}$$

Therefore,

$$RGA = \Lambda(s) = \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix} = \begin{bmatrix} \frac{1}{(1-\xi)} & \frac{-\xi}{(1-\xi)} \\ \frac{-\xi}{(1-\xi)} & \frac{1}{(1-\xi)} \end{bmatrix}$$

By taking $\lambda = \frac{1}{1-\xi}$, we can rewrite the RGA as

$$\Lambda = \begin{bmatrix} \lambda & 1 - \lambda \\ 1 - \lambda & \lambda \end{bmatrix}$$

and λ is usually called the ***Relative Gain Parameter***.

6.1.2.2 Matrix Method

Here, the Matrix Method is used to obtain the RGA. Consider the Steady State Gain Matrix of a 2×2 system $G(s)$ given by,

$$\lim_{s \rightarrow 0} G(s) = K = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}$$

The RGA or *Relative Gain Array* is obtained from K by matrix manipulation as

$$\Lambda(RGA) = K \otimes K^{-T}$$

$$\begin{aligned}
\Lambda &= \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \otimes \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}^{-T} \\
&= \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \otimes \frac{1}{|K|} \begin{bmatrix} k_{22} & -k_{21} \\ -k_{12} & k_{11} \end{bmatrix} \\
&= \frac{1}{|K|} \begin{bmatrix} k_{11}k_{22} & -k_{12}k_{21} \\ -k_{21}k_{12} & k_{22}k_{11} \end{bmatrix} \\
&= \frac{1}{k_{11}k_{22} - k_{12}k_{21}} \begin{bmatrix} k_{11}k_{22} & -k_{12}k_{21} \\ -k_{21}k_{12} & k_{22}k_{11} \end{bmatrix}
\end{aligned}$$

$$\text{Therefore } \Lambda = \frac{1}{1 - \frac{k_{12}k_{21}}{k_{11}k_{22}}} \begin{bmatrix} 1 & \frac{-k_{12}k_{21}}{k_{11}k_{22}} \\ \frac{-k_{12}k_{21}}{k_{11}k_{22}} & 1 \end{bmatrix}$$

$$\text{Taking } \frac{k_{12}k_{21}}{k_{11}k_{22}} = \xi$$

$$\Lambda = \frac{1}{1 - \xi} \cdot \begin{bmatrix} 1 & -\xi \\ -\xi & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{1-\xi} & \frac{-\xi}{1-\xi} \\ \frac{-\xi}{1-\xi} & \frac{1}{1-\xi} \end{bmatrix}$$

6.1.2.3 Properties of RGA

1. The individual elements of the RGA are represented as

$$\Lambda = \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix}$$

where

$$\begin{aligned}
\lambda_{11} &= \lambda_{22} = \frac{1}{1 - \xi} = \lambda \\
\text{and } \lambda_{12} &= \lambda_{21} = \frac{-\xi}{1 - \xi} = 1 - \lambda
\end{aligned}$$

Obviously, sum of the elements of the RGA across any row, or down any column equals 1.

2. Each element λ_{ij} is the Ratio of the Steady State Gain between output y_i and input m_j when all loops are open to the Steady State Gain when all loops except loop i are closed. That is,

$$\lambda_{ij} = \frac{k_{ij}}{\hat{k}_{ij}}$$

Hence λ_{ij} is called the **Relative Steady State Gain** or just **Relative Gain**. The value of λ_{ij} is a measure of the Steady State interaction expected in the i^{th} loop if its output y_i is paired with m_j .

3. If $\hat{k}_{ij}$ represents loop i Steady State Gain when all other loops are closed, and k_{ij} is the normal, open-loop steady state gain, then by the definition of Relative Gain,

$$\hat{k}_{ij} = \frac{1}{\lambda_{ij}} k_{ij}$$

$\frac{1}{\lambda_{ij}}$ tells us by what factor the open-loop steady state gain between y_i and m_j will be altered if other loops are closed. $\hat{k}_{ij}$ can also be called the **Effective Steady State Gain** when all other loops are closed.

For the 2x2 system, $\hat{k}_{11}$ will be the Steady State Gain between y_1 and m_1 when loop 2 is closed and so we can write

$$\hat{k}_{11} = \frac{k_{11}}{\lambda_{11}} = k_{11} (1 - \xi) = k_{11} \left(1 - \frac{k_{12}k_{21}}{k_{11}k_{22}} \right)$$

Similarly, $\hat{k}_{12}$ will be the Steady State Gain between y_1 and m_2 when loop 2 is closed and so we can write

$$\hat{k}_{12} = \frac{k_{12}}{\lambda_{12}} = k_{12} \left(\frac{1 - \xi}{-\xi} \right) = k_{12} \left(1 - \frac{k_{11}k_{22}}{k_{12}k_{21}} \right)$$

The same goes for $\hat{k}_{22} = \frac{k_{22}}{\lambda_{22}}$ {Steady State Gain between y_2 and m_2 when loop 1 is closed} and for $\hat{k}_{21} = \frac{k_{21}}{\lambda_{21}}$ {Steady State Gain between y_2 and m_1 when loop 2 is closed}.

6.1.3 Niederlinski Index

The Niederlinski Index is a measure that checks whether a particular loop pairing will be unstable in closed loop.

The $n \times n$ MIMO process, whose pairing has been decided as $y_1 - u_1, y_2 - u_2, \dots, y_n - u_n$, resulting in the transfer function $Y(s) = G(s)U(s)$ with $g_{ij}(s)$ rational and open loop stable, will be unstable in decentralized control for all possible controller parameters if,

$$NI = \frac{\det G(0)}{\prod_{i=1}^n g_{ii}(0)} < 0$$

6.1.4 Relative Effective Gain Array (REGA)

The RGA gives a measure of the Steady State Interaction and is thus used for Loop Pairing Decision. The REGA or the **Relative Effective Gain Array** builds upon the

RGA by incorporating dynamic information also into the interaction measure and thus helps us make a better loop pairing decision.

The first step is to extract or measure the steady state + dynamic information from the individual transfer functions of the MIMO process transfer function. Two factors in the open-loop transfer functions that are used to derive a steady state + dynamic interaction measure are:

1. **Steady State Gain:** The Steady State Gain of the transfer function reflects the effect of the manipulated variable m_j on the controlled variable y_i .
2. **Response Speed or Critical Frequency:** Response Speed is accountable for the sensitivity of the controlled variable y_i to manipulated variable m_j and thus the ability to reject interactions from other loops. Further, since Response Speed is proportional to the bandwidth or the Ultimate Frequency (Critical Frequency) in the frequency domain, we can use the Critical Frequency to give a measure for the sensitivity of the controlled variable y_i to the manipulated variable m_j .

Let the Response Characteristics of the MIMO process transfer function element $g_{ij}(s)$ be,

$$g_{ij}(j\omega) = k_{ij}\bar{g}_{ij}(j\omega)$$

where $k_{ij} = g_{ij}(j0)$ is the Steady State Gain and $\bar{g}_{ij}(j\omega)$ is the **Normalized Transfer Function** of $g_{ij}(j\omega)$.

In order to use both Steady State information and Response Speed information for interaction measure and loop pairing, the **energy transmission ratio** (also called the **effective gain**) e_{ij} for the transfer function $g_{ij}(j\omega)$ is defined as,

$$e_{ij} = k_{ij} \int_0^{\omega_{c,ij}} |\bar{g}_{ij}(j\omega)| d\omega \quad (6.1.12)$$

where k_{ij} is the Steady State Gain and $\omega_{c,ij}$ is the **Critical Frequency** of the transfer function and $|\bar{g}_{ij}(j\omega)|$ is the magnitude of the normalized transfer function. e_{ij} is called energy transmission ratio because the $|\bar{g}_{ij}(j\omega)|$ represents the magnitude of the normalized transfer function at various frequencies and we take the area under the magnitude-frequency curve till the critical frequency. The critical frequency can be defined in two ways:

1. $\omega_{c,ij} = \omega_{B,ij}$, where $\omega_{B,ij}$ is the *bandwidth* of the normalized transfer function $\bar{g}_{ij}(j\omega)$ and is determined by the frequency where the magnitude plot of frequency response

is reduced by $0.707g_{ij}(0)$, that is

$$|\bar{g}_{ij}(j\omega_{B,ij})| = 0.707 |g_{ij}(0)|$$

This is shown in Figure (6.1.5).

Figure 6.1.5: Bandwidth Frequency

2. $\omega_{c,ij} = \omega_{u,ij}$ where $\omega_{u,ij}$ is the *ultimate frequency* of normalized the transfer function $\bar{g}_{ij}(j\omega)$ and is determined by the frequency where the phase plot of the frequency response crossover equals $-\pi$, that is

$$\arg [g_{ij}(j\omega_{u,ij})] = -\pi$$

This is shown in Figure (6.1.6).

Figure 6.1.6: Ultimate Frequency

Further, the phase cross over information $\omega_{u,ij}$ is recommended for calculation of

e_{ij} because it is more closely linked to control system performance.

To simplify calculations, the integration in Eq. (6.1.12) is approximated by a rectangular area,

$$e_{ij} \approx k_{ij}\omega_{u,ij}$$

Putting together the energy transmission ratios of the individual elements of the MIMO system, the *Energy Transmission Ratio Array* (also called *Effective Gain Array*) is,

$$\begin{aligned} \mathbf{E} = \begin{bmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{bmatrix} &= \begin{bmatrix} k_{11}\omega_{u,11} & k_{12}\omega_{u,12} \\ k_{21}\omega_{u,21} & k_{22}\omega_{u,22} \end{bmatrix} \\ &= \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \otimes \begin{bmatrix} \omega_{u,11} & \omega_{u,12} \\ \omega_{u,21} & \omega_{u,22} \end{bmatrix} \end{aligned}$$

$$\text{Thus} \quad \mathbf{E} = \mathbf{K} \otimes \mathbf{\Omega} \quad (6.1.13)$$

where $\mathbf{K}$ is the Steady State Gain Matrix of the System and $\mathbf{\Omega}$ is the **Critical Frequency Array**.

Then, the **Relative Effective Gain** ϕ_{ij} between the output $y_i(s)$ and the input $m_j(s)$ is defined as the ratio of two energy transmission ratios,

$$\phi_{ij} = \frac{e_{ij}}{\hat{e}_{ij}} \quad (6.1.14)$$

where e_{ij} is the normal energy energy transmission ratio for $g_{ij}(s)$ (between output $y_i(s)$ and input $m_j(s)$) and $\hat{e}_{ij}$ is the *effective* energy transmission ratio between $y_i(s)$ and $m_j(s)$ when all loops are closed except loop i .

Again, for the 2x2 system, the matrix of relative effective gains is the **Relative Effective Gain Array**,

$$\text{REGA} \quad \Phi = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix}$$

Similar to how $\mathbf{K} \otimes \mathbf{K}^{-T}$ gives the RGA Λ , the REGA can now be obtained as

$$\text{REGA} \quad \Phi = \mathbf{E} \otimes \mathbf{E}^{-T}$$

where $\mathbf{E}$ is the *Energy Transmission Ratio Array* from 6.1.13.

6.1.4.1 Relative Critical Frequency Array (RCFA)

Elaborating on Eq. (6.1.14), the Relative Effective Gain can be written as,

$$\phi_{ij} = \frac{e_{ij}}{\hat{e}_{ij}} = \frac{k_{ij}\omega_{u,ij}}{\hat{e}_{ij}}$$

If $\hat{e}_{ij}$ is written as $\hat{e}_{ij} = \hat{k}_{ij}\hat{\omega}_{u,ij}$, where $\hat{k}_{ij}$ is the *Effective Steady State Gain* between y_i and m_j when all loops are closed except loop i , and $\hat{\omega}_{u,ij}$ is the **Effective Critical Frequency** of the transfer function $g_{ij}(s)$ (between output $y_i(s)$ and input $m_j(s)$) when all loops are closed except loop i , then,

$$\phi_{ij} = \frac{k_{ij}\omega_{u,ij}}{\hat{k}_{ij}\hat{\omega}_{u,ij}} = \frac{k_{ij}}{\hat{k}_{ij}} \frac{\omega_{u,ij}}{\hat{\omega}_{u,ij}} = \lambda_{ij}\gamma_{ij}$$

where $\frac{k_{ij}}{\hat{k}_{ij}}$ is replaced by λ_{ij} the corresponding *Relative Steady State Gain* from the RGA and,

$$\gamma_{ij} = \frac{\omega_{u,ij}}{\hat{\omega}_{u,ij}} = \frac{\phi_{ij}}{\lambda_{ij}}$$

is defined as the **Relative Critical Frequency**.

So for the 2x2 system, putting together the different *Relative Critical Frequencies*,

$$\Gamma = \begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix} = \begin{bmatrix} \frac{\phi_{11}}{\lambda_{11}} & \frac{\phi_{12}}{\lambda_{12}} \\ \frac{\phi_{21}}{\lambda_{21}} & \frac{\phi_{22}}{\lambda_{22}} \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} \odot \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix}$$

$$\text{Therefore } \Gamma = \Phi \odot \Lambda$$

where Γ is called the **RCFA or Relative Critical Frequency Array**, Φ is the REGA, Λ is the RGA and $\odot$ represents element by element division.

The elements of the RCFA and the RGA together are used to obtain REGA-based Effective Transfer Functions which are used in controller design.

6.1.5 Relative Normalized Gain Array (RNGA)

Since either the Frequency Bandwidth or the Ultimate frequency can be used in the derivation of the REGA, the REGA solution is not unique; also the two solutions can suggest different loop pairing. Further, the process input can cover the whole frequency domain and so the evaluation of overall process dynamics is required than just at particular frequency points.

The **RNGA** or the **Relative Normalized Gain Array** was developed after the REGA. Like the REGA, this RNGA provides an interaction measure that incorporates both Steady State and Dynamic information but unlike the REGA that uses frequency

domain information for evaluating dynamic properties, the RNGA uses the **Integral Error** (IE) Criterion.

Consider the 2x2 process,

$$G(s) = \begin{bmatrix} g_{11}(s) & g_{12}(s) \\ g_{21}(s) & g_{22}(s) \end{bmatrix}$$

where all the elements $g_{ij}(s)$ are of the FOPTD (First Order Plus Time Delay) form,

$$g_{ij}(s) = \frac{k_{ij}e^{-L_{ij}s}}{\tau_{ij}s + 1}$$

Let $y(t)$ be the open-loop Step Response to the transfer function $g_{ij}(s)$, and let $\bar{y}(t) = \frac{y(t)}{k_{ij}}$ be the normalized process output,

$$\bar{y}(t) = \left(1 - e^{-\frac{(t-L_{ij})}{\tau_{ij}}}\right) u(t - L_{ij}) \quad (6.1.15)$$

To measure the dynamic properties of the transfer function $g_{ij}(s)$ the **NIE** or the **Normalized Integrated Error** (IE of the normalized process output) is defined as,

$$\sigma_{ij} = \int_0^{\infty} (u(t) - \bar{y}(t)) dt$$

where $u(t)$ is the unit step input and $\bar{y}(t)$ is the normalized process output. Substituting $\bar{y}(t)$ from Eq. (6.1.15) into the above equation for NIE gives,

$$\begin{aligned} \sigma_{ij} &= \int_0^{\infty} \left(u(t) - \left(1 - e^{-\frac{(t-L_{ij})}{\tau_{ij}}}\right) u(t - L_{ij}) \right) dt \\ &= \int_0^{L_{ij}} u(t) dt + \int_{L_{ij}}^{\infty} e^{-\frac{(t-L_{ij})}{\tau_{ij}}} dt \\ &= L_{ij} + \frac{\left[e^{-\frac{(t-L_{ij})}{\tau_{ij}}} \right]_{L_{ij}}^{\infty}}{\frac{-1}{\tau_{ij}}} \\ \sigma_{ij} &= L_{ij} + \tau_{ij} \end{aligned}$$

which turns out to be the **Average Residence Time** of the normalized transfer function

$\bar{g}_{ij}(s) = \frac{g_{ij}(s)}{k_{ij}}$. The Average Residence Time of a process is the time it takes for the process response to reach 63.2% of the steady state value, from the instant the step input is applied. The smaller the value of σ_{ij} , faster will be the response and thus, better the dynamics of the system. Thus, σ_{ij} effectively captures the process dynamics of $\bar{g}_{ij}(s)$.

So there are two parameters for each transfer function that measure its steady state + dynamic properties:

1. **Steady State Gain:** The Steady State Gain of the transfer function reflects the effect of the manipulated variable m_j on the controlled variable y_i .
2. **Average Residence Time:** The Average Residence Time (of the normalized transfer function) gives a measure of the response speed of the controlled variable y_i to the manipulated variable m_j .

In order to use both the parameters for interaction measure and loop pairing, the **Normalized Gain** for a transfer function $g_{ij}(s)$ is defined as,

$$k_{N,ij} = \frac{k_{ij}}{\sigma_{ij}}$$

This indicates that for a transfer function with a large value of $k_{N,ij}$, when output y_i is paired with m_j , the combined effect of both the Steady State Gain and the Response Speed will be large. For example, if $k_{N,12} > k_{N,11}$ the dynamics of the pairing $y_1 - m_2$ will be better than the pairing $y_1 - m_1$, so it would be wiser to pair y_1 with m_2 . So loop pairing with large $k_{N,ij}$ should be preferred.

For the 2x2 system, different combinations of input and output gives rise to the **Normalized Gain Array**,

$$\begin{aligned} K_N &= \begin{bmatrix} k_{N,11} & k_{N,12} \\ k_{N,21} & k_{N,22} \end{bmatrix} \\ &= \begin{bmatrix} \frac{k_{11}}{\sigma_{11}} & \frac{k_{12}}{\sigma_{12}} \\ \frac{k_{21}}{\sigma_{21}} & \frac{k_{22}}{\sigma_{22}} \end{bmatrix} \\ &= \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \odot \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} \end{aligned}$$

$$\text{Thus} \quad K_N = K \odot T \quad (6.1.16)$$

where K is the Steady State Gain Matrix of the System and T is the **Average Residence Time Array**.

Then, the **Relative Normalized Gain** between output y_i and input m_j is defined as the ratio of two Normalized Gains,

$$\phi_{ij} = \frac{k_{N,ij}}{k_{N,ij}^*} \quad (6.1.17)$$

where $k_{N,ij}$ is the usual *Normalized Gain* of $g_{ij}(s)$ (between output $y_i(s)$ and input $m_j(s)$) and $k_{N,ij}^*$ is the *Normalized Gain* between output $y_i(s)$ and input $m_j(s)$ when all loops are closed except loop i .

Again for the 2x2 system, for different combinations of input and output, the **Relative Normalized Gain Array** is,

$$RNGA \quad \Phi = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix}$$

Similar to how $K \otimes K^{-T}$ was the RGA,

$$RNGA \quad \Phi = K_N \otimes K_N^{-T}$$

where K_N is the *Normalized Gain Array* from 6.1.16.

6.1.5.1 Relative Average Residence Time Array (RARTA)

Elaborating on Eq.(6.1.17), the *Relative Normalized Gain* can be written as,

$$\phi_{ij} = \frac{k_{N,ij}}{k_{N,ij}^*} = \frac{\frac{k_{ij}}{\sigma_{ij}}}{\frac{k_{ij}^*}{\sigma_{ij}^*}}$$

If $k_{N,ij}^*$ is written as $k_{N,ij}^* = \frac{k_{ij}^*}{\sigma_{ij}^*}$, where k_{ij}^* is the Effective Steady State Gain between y_i and m_j when all loops are closed except loop i , and $\hat{\sigma}_{ij}$ is the **Effective Average Residence Time** of the transfer function $g_{ij}(s)$ (between output $y_i(s)$ and input $m_j(s)$) when all loops are closed except loop i , then,

$$\phi_{ij} = \frac{\frac{k_{ij}}{\sigma_{ij}}}{\frac{k_{ij}^*}{\sigma_{ij}^*}} = \frac{k_{ij}}{k_{ij}^*} \frac{\sigma_{ij}^*}{\sigma_{ij}} = \lambda_{ij} \gamma_{ij}$$

where $\frac{k_{ij}}{k_{ij}^*}$ is replaced by λ_{ij} the corresponding Relative Steady State Gain from the RGA and

$$\gamma_{ij} = \frac{\sigma_{ij}^*}{\sigma_{ij}} = \frac{\phi_{ij}}{\lambda_{ij}}$$

is defined as the **Relative Average Residence Time**.

So for the 2x2 system, putting the different Relative Average Residence Times together,

$$\begin{aligned}\Gamma = \begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix} &= \begin{bmatrix} \frac{\phi_{11}}{\lambda_{11}} & \frac{\phi_{12}}{\lambda_{12}} \\ \frac{\phi_{21}}{\lambda_{21}} & \frac{\phi_{22}}{\lambda_{22}} \end{bmatrix} \\ &= \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} \odot \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix}\end{aligned}$$

$$\text{Therefore } \Gamma = \Phi \odot \Lambda$$

where Γ is called the **RARTA or Relative Average Residence Time Array**, Φ is the RNGA, Λ is the RGA and $\odot$ represents element by element division. The elements of the RARTA and the RGA together are used to obtain Effective Transfer Functions.

6.2 RNGA based Decentralized PID control

This decentralized control design method was first introduced in ref[24] (a) Steady State and transient information are used in calculations that help determine the best possible loop pairing which will give the best possible diagonally dominant combination (This was presented fully in Section (6.1.5)) (b) Then, ETFs or effective transfer functions are generated which attempt to approximate the DRGA result $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ (See Section (6.1.1)) with an FOPTD transfer function $g_{11}^*(s)$ and controllers are then designed for these FOPTD approximations, with some modifications that account for loop integrity which will be explained in the following section.

The DRGA, under the assumption of perfect control of all other loops, gives a simplified expression (for a 2×2 process) for the transfer function between input u_1 and output y_1 as,

$$g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$$

Generally, this transfer function **cannot** be simplified analytically to a simple FOPTD transfer function. However, in some cases, such as when the combined process delay of $g_{12}g_{21}$ is similar to that of the combined delay of $g_{11}g_{22}$, it can be seen that the multiplying term $1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}$ will be a rational transfer function. This in series with $g_{11}(s)$ can be considered as a higher order transfer function with delay which may be approximated as an FOPTD transfer function by reduction methods [64]. This will be especially true when the individual transfer function elements of the MIMO process are all FOPTD transfer functions. And it is only this class of MIMO processes that the RNGA-based decentralized method considers. The validity of approximating the DRGA result (when the individual transfer functions are FOPTD) with an FOPTD process will

be demonstrated with nyquist plots in numerical examples that follow.

However, this may not hold true for all 2×2 MIMO systems. Because, with respect to loop pairing, the RNGA method only gives the most diagonally dominant pairing possible but it cannot detect whether even this best pairing is or is not diagonally dominant *enough* for decentralized control[59]. In other words, even the best RNGA suggested pairing may not possess diagonal dominance and the DRGA result will not match the RNGA approximation for such cases. Cases that do not have diagonal dominance will require a sequential design approach or a compensator and such cases are not under consideration in this work.

6.2.1 ETFs for Loop Integrity

The Effective Transfer Function (ETF) by RNGA of $g_{ij}(s)$ is given by,

$$g_{ij}^*(s) = k_{ij}^* \times \frac{1}{\tau_{ij}^* s + 1} \times e^{-L_{ij}^* s}$$

But designing/tuning loop i 's PID controller $g_{ci}(s)$ for the corresponding diagonal element $g_{ii}^*(s)$ is not advisable because loop integrity must be taken into consideration. The controller $g_{ci}(s)$ should be designed for a modified RNGA-based ETF, $\hat{g}_{ii}(s)$ such that,

$$\hat{g}_{ij}(s) = \hat{k}_{ij} \times \frac{1}{\hat{\tau}_{ij} s + 1} \times e^{-\hat{L}_{ij} s}$$

where,

If $\lambda_{ij} < 1$	$\hat{k}_{ij} = \frac{k_{ij}}{\lambda_{ij}}$	elseif $\lambda_{ij} \geq 1$	$\hat{k}_{ij} = k_{ij}$
If $\gamma_{ij} > 1$	$\hat{\tau}_{ij} = \gamma_{ij} \cdot \tau_{ij}$ and $\hat{L}_{ij} = \gamma_{ij} \cdot L_{ij}$	elseif $\gamma_{ij} \leq 1$	$\hat{\tau}_{ij} = \tau_{ij}$ and $\hat{L}_{ij} = L_{ij}$

The derivation of this rule set can be found in refs[21]. Here, three relevant cases from the four possibilities are illustrated.

- **Case** $\lambda_{ij} < 1$ and $\gamma_{ij} > 1$: In this case, $\hat{k}_{ij} = \frac{k_{ij}}{\lambda_{ij}}$ and also $\hat{\tau}_{ij} = \gamma_{ij} \cdot \tau_{ij}$ and $\hat{L}_{ij} = \gamma_{ij} \cdot L_{ij}$. Therefore, $\hat{g}_{ij}(s) = g_{ij}^*(s)$. Figure(6.2.1) shows $g_{c1}g_{11}$, $g_{c1}g_{11}^*$, $g_{c2}g_{11}^*$ and $g_{c3}g_{11}^*$ where g_{c1} is designed for g_{11} itself, g_{c2} for g_{11}^* and g_{c3} for $\hat{g}_{11}$. $g_{c1}g_{11}$ is the open-loop of the first loop when the other loop of a 2×2 system is open and $g_{c1}g_{11}^*$ is what it becomes when the other loop is closed. So, the controller is redesigned for g_{11}^* as g_{c2} and also as g_{c3} for $\hat{g}_{11}$. It can be seen that $g_{c2}g_{11}^* = g_{c3}g_{11}^*$ nullifies the effect of closing the loops in $g_{c1}g_{11}^*$.

- **Case** $\lambda_{ij} < 1$ and $\gamma_{ij} < 1$: In this case, $\hat{k}_{ij} = \frac{k_{ij}}{\lambda_{ij}}$ but $\hat{\tau}_{ij} = \tau_{ij}$ and $\hat{L}_{ij} = L_{ij}$. Figure(6.2.2) shows the same open loop transfer functions again. Once more, the effect of closing the other loop ($g_{c1}g_{11}^*$) is best nullified by $g_{c3}g_{11}^*$.
- **Case** $\lambda_{ij} > 1$ and $\gamma_{ij} < 1$: In this case, $\hat{k}_{ij} = k_{ij}$ and also $\hat{\tau}_{ij} = \tau_{ij}$ and $\hat{L}_{ij} = L_{ij}$. Figure(6.2.2) shows the same open loop transfer functions for this case. The closing of the other loop ($g_{c1}g_{11}^*$) reduces the open loop gain from the designed original $g_{c1}g_{11}$. When the controller is redesigned as g_{c2} for g_{11}^* , the open loop gain is returned to the level of $g_{c1}g_{11}$. But in order to maintain loop integrity, it is better to allow the drop in gain when the other loop is closed. Hence, the rule set is such that in this case, $\hat{g}_{11} = g_{11}$.

Figure 6.2.1: **Case** $\lambda_{ij} < 1$ and $\gamma_{ij} > 1$

Figure 6.2.2: **Case $\lambda_{ij} < 1$ and $\gamma_{ij} < 1$** Figure 6.2.3: **Case $\lambda_{ij} > 1$ and $\gamma_{ij} < 1$**

6.2.2 Numerical Example

The PID control design procedure by the RNGA method is illustrated with the help of a numerical example. The example system is:

$$G(s) = \begin{bmatrix} g_{11}(s) & g_{12}(s) \\ g_{21}(s) & g_{22}(s) \end{bmatrix} = \begin{bmatrix} \frac{12.8}{16.7s+1}e^{-4s} & \frac{-18.9}{21s+1}e^{-9s} \\ \frac{6.6}{10.9s+1}e^{-5s} & \frac{19.4}{14.4s+1}e^{-5s} \end{bmatrix}$$

The Steady State Gain matrix is,

$$K = \begin{bmatrix} 12.8 & -18.9 \\ 6.6 & 19.4 \end{bmatrix}$$

The RGA matrix is computed as $\Lambda = K \otimes K^{-T}$,

$$\Lambda = \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix} = \begin{bmatrix} 0.6656 & 0.3344 \\ 0.3344 & 0.6656 \end{bmatrix}$$

The largest +ve value terms are in the diagonal positions, forcing us to opt for the **1-1/2-2 pairing**. The NI is calculated to be $1.5023 > 0$, which agrees with the chosen pairing.

The pairing by RNGA method also agrees with the pairing decision by RGA: The Normalized Gain Matrix is,

$$K_N = \begin{bmatrix} \frac{12.8}{20.7} & \frac{-18.9}{30} \\ \frac{6.6}{15.9} & \frac{19.4}{19.4} \end{bmatrix} = \begin{bmatrix} 0.6184 & -0.63 \\ 0.4151 & 1 \end{bmatrix}$$

The *Average Residence Time Array* is given by

$$\mathbf{T} = \begin{bmatrix} (16.7 + 4) & (21 + 9) \\ (10.9 + 5) & (14.4 + 5) \end{bmatrix} = \begin{bmatrix} 20.7 & 30 \\ 15.9 & 19.4 \end{bmatrix}$$

The *Normalized Gain Array* is given by

$$\mathbf{K}_N = \mathbf{K} \odot \mathbf{T} = \begin{bmatrix} \frac{12.8}{20.7} & \frac{-18.9}{30} \\ \frac{6.6}{15.9} & \frac{19.4}{19.4} \end{bmatrix} = \begin{bmatrix} 0.6184 & -0.63 \\ 0.4151 & 1 \end{bmatrix}$$

Therefore, the **RNGA** or the *Relative Normalized Gain Array* is given by

$$\Phi = \mathbf{K}_N \otimes \mathbf{K}_N^{-T} = \begin{bmatrix} 0.7028 & 0.2972 \\ 0.2972 & 0.7028 \end{bmatrix}$$

Since the diagonal elements are +ve large, the only possible **pairing is 1-1/2-2**. The

RARTA or the *Relative Average Residence time Array* is,

$$\begin{aligned}\Gamma &= \Phi \odot \Lambda = \begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix} \\ &= \begin{bmatrix} 0.7028 & 0.2972 \\ 0.2972 & 0.7028 \end{bmatrix} \odot \begin{bmatrix} 0.6656 & 0.3344 \\ 0.3344 & 0.6656 \end{bmatrix} = \begin{bmatrix} 1.0558 & 0.8889 \\ 0.8889 & 1.0558 \end{bmatrix}\end{aligned}$$

Now, the controllers $g_{c1}(s)$ and $g_{c2}(s)$ are designed for the Effective Transfer functions (with integrity rules) of $g_{11}(s)$ and $g_{22}(s)$, namely $\hat{g}_{11}(s)$ and $\hat{g}_{22}(s)$. The ETF of $g_{ij}(s)$ is given by,

$$\hat{g}_{ij}(s) = \frac{\hat{k}_{ij}}{\hat{\tau}_{ij}s + 1} e^{-\hat{L}_{ij}s}$$

In the 2x2 process in question, $\lambda_{11}, \lambda_{22} < 1$ and $\gamma_{11}, \gamma_{22} > 1$. So,

$$\hat{g}_{11}(s) = g_{11}^*(s) \quad \text{and} \quad \hat{g}_{22}(s) = g_{22}^*(s)$$

In order to verify that the FOPTD ETFs by RNGA match the DRGA results closely, the step responses and nyquists plots are compared in Figure (6.2.5).

Thus, the controllers $g_{c1}(s)$ and $g_{c2}(s)$ are designed for $g_{11}^*(s)$ and $g_{22}^*(s)$, using the Gain Margin - Phase Margin Method. The parameters of the PI controller used to control a system given by $\frac{b}{as+1} \cdot e^{-Ls}$ are,

$$\begin{bmatrix} K_p \\ K_i \end{bmatrix} = \frac{\pi}{2A_m L b} \cdot \begin{bmatrix} a \\ 1 \end{bmatrix}$$

Thus, the controllers $g_{c1}(s)$ and $g_{c2}(s)$ are calculated to be: (A_m was taken as 3)

$$\begin{aligned}g_{c1}(s) &= 0.1137 + \frac{0.006447}{s} \\ g_{c2}(s) &= 0.05174 + \frac{0.003403}{s}\end{aligned}$$

A unit step is applied to input 1 at $t = 0$ and to input 2 at $t = 150$. Figures (6.2.4) show the response of Loop 1 (output 1) and Loop 2 (output 2) respectively, to these inputs. The ISE (Integral Square Error) of each of the output signals is also shown in the legend.

Figure 6.2.4: Comparison of closed loop responses of RNGA-based vs no-ETF PID design. (A step of 1 was applied at $t = 0$ to loop 1 and at $t = 150s$ to loop 2)

Figure 6.2.5: Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. **Top:** $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* **Bottom:** $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*

6.3 RNGA-based Decentralized GPC control

6.3.1 Proposed Procedure

The procedure for the design of the RNGA based decentralized GPC is as follows.

1. Loop pairing is decided using RGA, RNGA and NI analysis, as before.
2. The controller used for each loop will be the CDGPC. This is because of it offers the highest Robustness compared to the conventional GPC and the SPGPC. Most importantly, the robustness of the CDGPC does not vary with process delay as does the conventional GPC. This has been discussed in Chapter (5) in Section (5.3.1).

However, for verification, simulation studies have been conducted with the GPC and SPGPC but failed to produce stable responses. In particular, the unsuccessful combinations studied were:

- (a) conventional GPC with Shridhar-Cooper tuning[61].
 - (b) conventional GPC with N^* tuning.
 - (c) SPGPC with Shridhar-Cooper tuning.
3. The CDGPC for each loop will be tuned using the new N^* method because it was shown in Chapter (2) that it provides better performance and robustness than the tuning using the existing method of Shridhar-Cooper. Furthermore, the fact that a single parameter k can be used to trade-off performance and robustness will be very useful in the application of the CDGPC to decentralized control.
 4. Each loop is assigned a CDGPC which is designed for the RNGA based Effective Transfer Function with Integrity Rules which were presented in Section (6.2). That is, the CDGPC of loop i is designed for (the discrete-time version of) $\hat{g}_{ii}(s)$,

$$\hat{g}_{ii}(z) = \frac{\hat{b}_{ii} [(1 - \hat{\alpha}_{ii}) + \hat{\alpha}_{ii} z^{-1}] z^{-1}}{1 - \hat{a}_{ii} z^{-1}} z^{-\hat{d}_{ii}}$$

Continuing with the example 2×2 system from Section (6.2.2), Figure (6.3.1) shows closed loop responses for the cases when the CDGPC is designed for g_{ii} and $\hat{g}_{ii}$. It is very clear that using the RNGA-based ETF with Integrity Rules ($\hat{g}_{ii}$) results in much better performance (ISE values are given in brackets in the figure legend.). All other parameters including the sampling time for each loop ($T_{11} = T_{22} = 0.5$) and the k value for each loop ($k_1 = k_2 = 0.7$) were the same for both cases.

Figure 6.3.1: Comparison of closed loop responses of RNGA-based vs no-ETF CDGPC design. (A step of 1 was applied at $t = 0$ to loop 1 and at $t = 150s$ to loop 2)

5. The N^* tuning technique reduces the four parameters of the CDGPC of Loop i to just one normalized parameter k_i . The correct selection of the value for k_i and the sampling time T_i for loop i is very crucial. The procedure for this will be described below:

- (a) **Sampling Time:** The sampling time for the loop i , T_i , was chosen with the help of the sampling time formula of N^* method, in Section (2.3.4). The relation of Eq.(2.3.10) is reproduced below

$$T = -\frac{\tau}{(c-1)} \ln(k)$$

where k is fixed at 0.8 and τ is the time constant of the FOPTD process in question. c is the ratio of settling time of the FOPTD process (minus the delay) to the settling time of the closed loop (minus the delay) and reflects

how much faster the closed loop is when compared to the original process. Translating this idea to the present context of decentralized control, where the sampling time T_i of loop i should be chosen considering the process delay of $\hat{g}_{ii}$ as well, the formula is adapted as,

$$T_i = -\frac{(\hat{\tau}_{ii} + \hat{L}_{ii})}{c - 1} \ln(k)$$

Here, $\hat{\tau}_{ii} + \hat{L}_{ii}$ is nothing but the effective average residence time $\hat{\sigma}_{ii}$ of $\hat{g}_{ii}$. Finally, at this design step, c and k are chosen to be 10 and 0.8 for all loops; these numbers were determined from simulation studies.

- (b) **Tuning Parameter:** Once the controllers are designed in this way with a suitable sampling time for each loop, the k_i parameter can now be adapted for each loop as follows. Consider a 2×2 decentralized control structure with 2 loops. Initially, it is assumed that for every loop $k_{ii} = 0.5$. Now, if for instance, $\hat{\tau}_{11} + \hat{L}_{11} < \hat{\tau}_{22} + \hat{L}_{22}$, then k_2 can be reduced below 0.5 in order to make both loops operate at around the same speeds which would greatly reduce interactions caused by one loop being fast and other loop being slow. Hence, the speed of the loop with the slower process can be increased using the following formula,

$$k_{ii} \leftarrow k_{ii} \times \frac{\hat{\sigma}_{min}}{\hat{\tau}_{ii} + \hat{L}_{ii}}$$

where $\hat{\sigma}_{min} = \min(\hat{\tau}_{ii} + \hat{L}_{ii}) \forall i$. With this formula, the loop having the minimum effective average residence time will continue to have $k_{ii} = 0.5$ while all other loops will have $k_{ii} < 0.5$, thereby speeding up.

Continuing with the example 2×2 system from Section (6.2.2), with the steps above, the RNGA-based GPC scheme is compared with the original RNGA-based PID scheme in Figure (6.3.2). The ISE value is provided next to the legend entries.

Figure 6.3.2: **Left:** Outputs 1 and 2 **Right:** Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 150$ a setpoint of 1 is applied to Loop 2

6.4 2×2 Case Studies

In the following sections, simulation results from four 2×2 Case study systems [9, 20, 21, 22, 23, 24, 73, 72, 74] are provided. Details of each of the systems, the approximations of the second order elements using Half-Rule, details of the loop pairing using RNGA method and other details are given in Appendix (i).

The closed loop responses with RNGA-based PID control were compared with the closed loop responses from RNGA-based GPC control. The index used to evaluate the performance of each loop is the ISE, which is provided in the legend of the Figures. Also, the comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation are provided.

6.4.1 Wardle & Wood

Figure 6.4.1: Wardle & Wood - **Left:** Outputs 1 and 2 **Right:** Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 300$ a setpoint of 1 is applied to Loop 2

Figure 6.4.2: Wardle & Wood - Comparisons of step responses and nyquist plots of the DRGA ETF and the RGA approximation. **Top:** $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$ vs g_{11}^*
Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$ vs g_{22}^*

6.4.2 Wood and Berry

Figure 6.4.3: Wood & Berry - **Left:** Outputs 1 and 2 **Right:** Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 100$ a setpoint of 1 is applied to Loop 2

Figure 6.4.4: Wood & Berry - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. **Top:** $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$ vs g_{11}^* **Bottom:** $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$ vs g_{22}^*

6.4.3 Vinante & Luyben

Figure 6.4.5: Vinante & Luyben - **Left:** Outputs 1 and 2 **Right:** Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 50$ a setpoint of 1 is applied to Loop 2

Figure 6.4.6: Vinante & Luyben - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. **Top:** $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$ vs g_{11}^*
Bottom: $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)} \right)$ vs g_{22}^*

6.4.4 Industrial-Scale Polymerization Reactor

Figure 6.4.7: Industrial-Scale Polymerization Reactor - **Left:** Outputs 1 and 2 **Right:** Control Actions 1 and 2. A setpoint of 1 is applied at $t = 0$ to loop 1 and then at $t = 25$ a setpoint of 1 is applied to Loop 2

Figure 6.4.8: Polymerization Reactor - Comparisons of step responses and nyquist plots of the DRGA ETF and the RNGA approximation. **Top:** $g_{11}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{11}^* **Bottom:** $g_{22}(s) \left(1 - \frac{g_{12}(s)g_{21}(s)}{g_{11}(s)g_{22}(s)}\right)$ vs g_{22}^*

6.5 Chapter Conclusions and Future Work

The conclusion that can be arrived at is that the combination of CDGPC and N^* tuning approach allows the implementation of GPC in a decentralized structure where the conventional GPC design with conventional tuning fails. Loop pairing decision is done using RGA, RGA and NI analysis. RNGA based Effective transfer functions are developed for each diagonal element of the process, with considerations for loop integrity. A CDGPC with N^* tuning is designed for the ETF of each diagonal element. Closed loop simulations of four 2×2 Case Study systems were presented and compared with the performance of the RNGA-based PID designs. It was found that the performance was improved with the RNGA-based GPC controllers.

However, some of the limitations of the RNGA method are inherited. The RNGA-based loop pairing and RNGA-based controller design is applicable only to MIMO systems with FOTPD transfer functions. With respect to loop pairing, the RNGA method only gives the most diagonally dominant pairing possible but it cannot detect whether even this pairing is or is not diagonally dominant *enough* for decentralized control[59]. In other words, even the best RNGA suggested pairing may not possess diagonal dominance and the DRGA result will not match the RNGA approximation for such cases. Cases that do not have diagonal dominance will require a sequential design approach or a compensator and such cases were not under consideration.

Because the CDGPC loop does not offer internal stability for the control of unstable process (see Section (5.3.2)) like the conventional GPC does, the proposed GPC design method cannot be applied to MIMO systems with open loop unstable elements. However, the 2GPC maintains the structure of the conventional GPC and does offer internal stability (see Section (4.3.4)). So, a study of the 2GPC controller in a decentralized setting is an attractive way forward.

The objective of this work was to identify problems with the GPC that would interfere with it being used in a decentralized setting (such as a robustness that is a function of process delay) and then modify or improve it to work **specifically** with the RNGA-based loop-pairing and controller-design method. While this objective has been attained, the idea of using the newly proposed GPC variants and tuning methods with sequential loop closing methods is a promising one and could be explored in the future.

Bibliography

- [1] P. Ansay, M. Gevers, and V. Wertz. Enhancing the robustness of gpc via a simple choice of the youla parameter. *European Journal of Control*, 4(1):64–70, 1998.
- [2] K. J. Astrom and T. Hagglund. *PID Controllers: Theory, Design, and Tuning*. Instrument Society of America, Research Triangle Park, North Carolina, 2 edition, 1995.
- [3] K. J. Astrom, C. C. Hang, and B. C. Lim. New smith predictor for controlling a process with an integrator and long dead-time. *IEEE Transactions on Automatic Control*, 39:343–345, 1994.
- [4] P. Banerjee and S. L. Shah. Tuning guidelines for robust generalized predictive control. In *Proceedings of the 1992 IEEE Conference on Decision and Control*, pages 3233–3234 vol.4, 1992 1992.
- [5] P. Banerjee and S. L. Shah. The role of signal-processing methods in the robust design of predictive control. *Automatica*, 31(5):681–695, 1995.
- [6] Pranob Banerjee and Sirish L. Shah. Robust stability of gpc as applied to a first order model with delay. In *Proceedings of the American Control Conference*, volume 4, pages 2884–2889.
- [7] Robert P. Bithmead, V. Wertz, and Micheal Gerers. *Adaptive Optimal Control: The thinking man's G.P.C.* Prentice Hall, 1991.
- [8] C. Bordons and E. F. Camacho. A generalized predictive controller for a wide class of industrial processes. *IEEE Transactions on Control Systems Technology*, 6(3): 372–387, 1998.
- [9] W. J. Cai, W. Ni, M. J. He, and C. Y. Ni. Normalized decoupling - a new approach for mimo process control system design. *Industrial and Engineering Chemistry Research*, 47(19):7347–7356, 2008. ISSN 08885885.

- [10] E.F. Camacho and C. Bordons. *Model Predictive Control*. Springer, second edition, 2004.
- [11] D. Chen and D. E. Seborg. Design of decentralized pi control systems based on nyquist stability analysis. *Journal of Process Control*, 13(1):27–39, 2003.
- [12] I. L. Chien, H. P. Huang, and J. C. Yang. A simple multiloop tuning method for pid controllers with no proportional kick. *Industrial and Engineering Chemistry Research*, 38(4):1456–1468, 1999.
- [13] D. W. Clarke and C. Mohtadi. Properties of generalized predictive control. *Automatica*, 25(6):859–875, 1989.
- [14] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control-part ii extensions and interpretations. *Automatica*, 23(2):149–160, 1987.
- [15] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control-part i. the basic algorithm. *Automatica*, 23(2):137–148, 1987.
- [16] D.W. Clarke. *Advances in Model-Based Predictive Control*. Oxford University Press, Oxford, 1994.
- [17] C.R. Cutler and B.L. Ramaker. Dynamic matrix control - a computer control algorithm. *AIChE National Meeting*, 1979.
- [18] Jorge L. Garriga and Masoud Soroush. Model predictive control tuning methods: A review. *Industrial & Engineering Chemistry Research*, 2010.
- [19] Pierre Grosdidier, Manfred Morari, and Bradley R. Holt. Closed-loop properties from steady-state gain information. *Industrial & Engineering Chemistry Fundamentals*, 24(2):221–235, 1985.
- [20] M. J. He and W. J. Cai. New criterion for control-loop configuration of multivariable processes. *Industrial and Engineering Chemistry Research*, 43(22):7057–7064, 2004. Cited By (since 1996): 8 Export Date: 22 February 2010 Source: Scopus.
- [21] M. J. He, W. J. Cai, and S. Y. Li. Evaluation of the decentralized closed-loop integrity for multivariable control systems. *Industrial and Engineering Chemistry Research*, 44(10):3567–3574, 2005. Cited By (since 1996): 4 Export Date: 22 February 2010 Source: Scopus.
- [22] M. J. He, W. J. Cai, B. F. Wu, and M. He. Simple decentralized pid controller design method based on dynamic relative interaction analysis. *Industrial and Engineering Chemistry Research*, 44(22):8334–8344, 2005. ISSN 08885885.

- [23] M. J. He, W. J. Cai, and B. F. Wu. Design of decentralized imc-pid controller based on dri analysis. *AIChE Journal*, 52(11):3852–3863, 2006. ISSN 00011541.
- [24] M. J. He, W. J. Cai, W. Ni, and L. H. Xie. Rnga based control system configuration for multivariable processes. *Journal of Process Control*, 19(6):1036–1042, 2009. ISSN 09591524.
- [25] W. K. Ho, T. H. Lee, and O. P. Gan. Tuning of multiloop proportional-integral-derivative controllers based on gain and phase margin specifications. *Industrial and Engineering Chemistry Research*, 36(6):2231–2238, 1997. Cited By (since 1996): 26 Export Date: 14 August 2012 Source: Scopus.
- [26] M. Hovd and S. Skogestad. Simple frequency-dependent tools for control system analysis, structure selection and design. *Automatica*, 28(5):989–996, 1992. Cited By (since 1996): 73 Export Date: 23 February 2010 Source: Scopus.
- [27] M. Hovd and S. Skogestad. Improved independent design of robust decentralized controllers. *Journal of Process Control*, 3(1):43–51, 1993. ISSN 09591524.
- [28] M. Hovd and S. Skogestad. Sequential design of decentralized controllers. *Automatica*, 30(10):1601–1607, 1994. ISSN 00051098.
- [29] K. Hrissagis, O. D. Crisalle, and M. Sznaier. Robust design of unconstrained predictive controllers. In *American Control Conference, 1995. Proceedings of the*, volume 5, pages 3681–3687 vol.5, 21–23 Jun 1995 1995.
- [30] H. P. Huang and F. Y. Lin. Decoupling multivariable control with two degrees of freedom. *Industrial and Engineering Chemistry Research*, 45(9):3161–3173, 2006.
- [31] N. Jensen, D. G. Fisher, and S. L. Shah. Interaction analysis in multivariable control systems. *AIChE Journal*, 32(6):959–970, 1986.
- [32] Ogata Katsuhiko. *Modern Control Engineering*. Prentice Hall PTR, 2001. 516039.
- [33] B. Kouvaritakis, J. A. Rossiter, and A. O. T. Chang. Stable generalised predictive control. an algorithm with guaranteed stability. *IEE Proceedings D: Control Theory and Applications*, 139(4):349–363, 1992.
- [34] T. Liu, X. He, D. Y. Gu, and W. D. Zhang. Analytical decoupling control design for dynamic plants with time delay and double integrators. *Control Theory and Applications, IEE Proceedings -*, 151:745–753, 2004.

- [35] T. Liu, W. Zhang, and F. Gao. Analytical two-degrees-of-freedom (2-dof) decoupling control scheme for multiple-input - multiple-output (mimo) processes with time delays. *Industrial and Engineering Chemistry Research*, 46(20):6546–6557, 2007.
- [36] W. L. Luyben. Simple method for tuning siso controllers in multivariable systems. *Industrial & Engineering Chemistry Process Design and Development*, 25(3):654–660, 1986. ISSN 01964305.
- [37] J.M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 2002.
- [38] V. M. Maniar, S. L. Shah, D. G. Fisher, and R. K. Mutha. Multivariable constrained adaptive gpc: Theory and experimental evaluation. *International Journal of Adaptive Control and Signal Processing*, 11(4):343–365, 1997.
- [39] T. J. McAvoy. Connection between relative gain and control loop stability and design. *AIChE Journal*, 27(4):613–619, 1981. ISSN 00011541.
- [40] A. R. McIntosh, S. L. Shah, and D. G. Fisher. Selection of tuning parameters for adaptive generalized predictive control. In *American Control Conference, 1989*, pages 1846–1851, 21-23 June 1989 1989.
- [41] Andrew R. McIntosh, Sirish L. Shah, and D. Grant Fisher. Analysis and tuning of adaptive gpc. *Canadian Journal of Chemical Engineering*, 69(1):97–100, 1991.
- [42] T. J. Monica, C. C. Yu, and W. L. Luyben. Improved multiloop single-input/single-output (siso) controllers for multivariable processes. *Industrial and Engineering Chemistry Research*, 27(6):969–973, 1988.
- [43] M. Morari and E. Zafirov. *Robust Process Control*. Prentice Hall, Englewood Cliffs, NJ, 1989.
- [44] K. Mukati and B. Ogunnaike. Stability analysis and tuning strategies for a novel next generation regulatory controller. volume 5 of *Proceedings of the 2004 American Control Conference (AAC)*, pages 4034–4039, Boston, MA, 2004.
- [45] K. Mukati, M. Rasch, and B. A. Ogunnaike. An alternative structure for next generation regulatory controllers. part ii: Stability analysis, tuning rules and experimental validation. *Journal of Process Control*, 19(2):272–287, 2009.
- [46] A. Niederlinski. A heuristic approach to the design of linear multivariable interacting subsystems. *Automatica*, 7:691–701, 1971.
- [47] J. E. Normey-Rico and E. F. Camacho. A smith predictor based generalized predictive controller. Technical Report GAR 02-96, University of Seville, 1996.

- [48] J. E. Normey-Rico and E. F. Camacho. Robustness effects of a prefilter in a smith predictor-based generalised predictive controller. *IEE Proceedings: Control Theory and Applications*, 146(2):179–185, 1999.
- [49] J. E. Normey-Rico and E. F. Camacho. Robust design of gpc for processes with time delay. *International Journal of Robust and Nonlinear Control*, 10(13):1105–1127, 2000.
- [50] J. E. Normey-Rico, E. F. Camacho, and J. Gomez Ortega. Robustness and prediction in generalized predictive controller. In *XII Brazilian Control Conference*, pages 157–162, Uberlandia, Brazil, 1998.
- [51] J. E. Normey-Rico, J. Gomez Ortega, and E. F. Camacho. Smith predictor based generalized predictive controller for mobile robot path tracking. In *III IFAC symposium on Intelligent Autonomous Vehicles*, pages 471–476, Madrid, Spain, 1998.
- [52] J.E. Normey-Rico and E.F. Camacho. *Control of Dead-time Processes*. Springer, London, 2007.
- [53] B. A. Ogunnaike and K. Mukati. An alternative structure for next generation regulatory controllers: Part i: Basic theory for design, development and implementation. *Journal of Process Control*, 16(5):499–509, 2006.
- [54] B.A. Ogunnaike and W.H. Ray. *Process Dynamics Modeling and Control*. Oxford University Press, New York, 1994.
- [55] G. Ray, A. N. Prasad, and T. K. Bhattacharyya. Design of decentralized robust load-frequency controller based on svd method. *Computers and Electrical Engineering*, 25(6):477–492, 1999. Cited By (since 1996): 2 Export Date: 14 August 2012 Source: Scopus.
- [56] B. D. Robinson and D. W. Clarke. Robustness effects of a prefilter in generalised predictive control. *Control Theory and Applications, IEE Proceedings D*, 138(1):2–8, 1991.
- [57] J.A. Rossiter. *Model-Based Predictive Control: A Practical Approach*. CRC Press, 2003.
- [58] D.E. Seborg, T.F. Edgar, and D.A. Mellichamp. *Process Dynamics and Control*. John Wiley & Sons, 2nd edition, 2003.
- [59] Y. Shen, W. J. Cai, and S. Li. Multivariable process control: Decentralized, decoupling, or sparse? *Industrial and Engineering Chemistry Research*, 49(2):761–771, 2010.

- [60] S. J. Shiu and S. H. Hwang. Sequential design method for multivariable decoupling and multiloop pid controllers. *Industrial and Engineering Chemistry Research*, 37(1):107–119, 1998.
- [61] R. Shridhar and D. J. Cooper. A tuning strategy for unconstrained siso model predictive control. *Industrial and Engineering Chemistry Research*, 36(3):729–746, 1997.
- [62] R. Shridhar and D. J. Cooper. A novel tuning strategy for multivariable model predictive control. *ISA Transactions*, 36(4):273–280, 1997. Cited By (since 1996): 16 Export Date: 17 September 2009 Source: Scopus.
- [63] R. Shridhar and D. J. Cooper. A tuning strategy for unconstrained multivariable model predictive control. *Industrial and Engineering Chemistry Research*, 37(10):4003–4016, 1998.
- [64] S. Skogestad. Simple analytic rules for model reduction and pid controller tuning. *Journal of Process Control*, 13(4):291–309, 2003.
- [65] S. Skogestad and M. Morari. Robust performance of decentralized control systems by independent designs. *Automatica*, 25(1):119–125, 1989. ISSN 00051098.
- [66] O. J. M. Smith. Closer control of loops with dead time. *Chem. Eng. Prog.*, 53:217–219, 1957.
- [67] W. Tang, D. Meng, X. Wang, K. Zhang, and M. Wang. Controller design of a two-degree-of-freedom smith predictor scheme for improved performance. In *IEEE International Conference on Information and Automation, ICIA 2009*, pages 242–247, 2009.
- [68] Y. C. Tian and F. Gao. Double-controller scheme for separating load rejection from set-point tracking. *Chemical Engineering Research and Design*, 76:445–450, 1998.
- [69] Y. C. Tian and F. Gao. Control of integrator processes with dominant time delay. *Industrial and Engineering Chemistry Research*, 38(8):2979–2983, 1999.
- [70] T. Wei, S. Songjiao, and W. Mengxiao. A novel smith predictor with double-controller structure. In *Proceedings of the American Control Conference*, volume 6, pages 5074–5079, 2002.
- [71] Mark F. Witcher and Thomas J. McAvoy. Interacting control systems: Steady state and dynamic measurement of interaction. *ISA Transactions*, 16(3):35–41, 1977. Cited By (since 1996): 57 Export Date: 16 August 2012 Source: Scopus.

- [72] Q. Xiong and W. J. Cai. Effective transfer function method for decentralized control system design of multi-input multi-output processes. *Journal of Process Control*, 16: 773–784, 2006.
- [73] Q. Xiong, W. J. Cai, M. J. He, and M. He. Decentralized control system design for multivariable processes - a novel method based on effective relative gain array. *Industrial and Engineering Chemistry Research*, 45:2769–2776, 2006.
- [74] Q. Xiong, W. J. Cai, and M. J. He. Equivalent transfer function method for pi/pid controller design of mimo processes. *Journal of Process Control*, 17:665–673, 2007.
- [75] K. Yamuna Rani and H. Unbehauen. Study of predictive controller tuning methods. *Automatica*, 33(12):2243–2248, 1997.
- [76] T. Yoon and D. W. Clarke. Prefiltering in receding-horizon predictive control. Technical Report OUEL/93, Oxford University, 1993.
- [77] T.W. Yoon and D. W. Clarke. Towards robust adaptive predictive control. In *Advances in Model-Based Predictive Control*, pages 402–414. Oxford University Press, Oxford, U.K., 1994.
- [78] T.W. Yoon and D.W. Clarke. Observer design in receding-horizon predictive control. *International Journal of Control*, 61(1):171–191, 1995.
- [79] W. Zhang, Y. Sun, and X. Xu. Two degree-of-freedom smith predictor for processes with time delay. *Automatica*, 34:1279–1282, 1998.
- [80] W. D. Zhang and Y. X. Sun. Modified smith predictor for controlling integrator/time delay processes. *Industrial and Engineering Chemistry Research*, 35:2769–2772, 1996.
- [81] Yu Zhang, Qing-Guo Wang, and K. J. Astrom. Dominant pole placement for multi-loop control systems. volume 3, pages 1965–1969, 2000. Cited By (since 1996): 2 Export Date: 14 August 2012 Source: Scopus.

APPENDICES

i Appendix: Example Systems

i.1 2×2 Case Studies

i.1.1 *System 1: Wardle & Wood*

$$G_1 = \begin{bmatrix} \frac{0.126}{60s+1}e^{-6s} & \frac{-0.101}{(48s+1)(45s+1)}e^{-12s} \\ \frac{0.094}{38s+1}e^{-8s} & \frac{-0.12}{35s+1}e^{-8s} \end{bmatrix}$$

(After Loop Pairing 1 – 1/2 – 2)		
	Model g_{ii}	Half-Rule
g_{11}	$\frac{0.126}{60s+1}e^{-6s}$	-
g_{12}	$\frac{-0.101}{(48s+1)(45s+1)}e^{-12s}$	$\frac{-0.101}{70.5s+1}e^{-34.5s}$
g_{21}	$\frac{0.094}{38s+1}e^{-8s}$	-
g_{22}	$\frac{-0.12}{35s+1}e^{-8s}$	-

RG	$\begin{bmatrix} 2.688 & -1.688 \\ -1.688 & 2.688 \end{bmatrix}$
RNG	$\begin{bmatrix} 1.584 & -0.5844 \\ -0.5844 & 1.584 \end{bmatrix}$
NI	0.3721

Diagonal Elements	ETF g_{ii}^*	ETF for Integrity $\hat{g}_{ii}$
$g_{11} = \frac{0.126}{60s+1}e^{-6s}$	$g_{11}^* = \frac{0.0469}{35.4s+1}e^{-3.54s}$	$\hat{g}_{11} = \frac{0.126}{60s+1}e^{-6s} = g_{11}$
$g_{22} = \frac{-0.12}{35s+1}e^{-8s}$	$g_{22}^* = \frac{-0.0557}{13.2s+1}e^{-2.77s}$	$\hat{g}_{22} = \frac{-0.12}{35s+1}e^{-8s} = g_{22}$

i.1.2 *System 2: Wood & Berry*

$$G_2 = \begin{bmatrix} \frac{12.8}{16.7s+1}e^{-s} & \frac{-18.9}{21s+1}e^{-3s} \\ \frac{6.6}{10.9s+1}e^{-7s} & \frac{-19.4}{14.4s+1}e^{-3s} \end{bmatrix}$$

(After Loop Pairing 1 – 1/2 – 2)		
	Model	Half-Rule
g_{11}	$\frac{12.8}{16.7s+1}e^{-s}$	-
g_{12}	$\frac{-18.9}{21s+1}e^{-3s}$	-
g_{21}	$\frac{6.6}{10.9s+1}e^{-7s}$	-
g_{22}	$\frac{-19.4}{14.4s+1}e^{-3s}$	-

RGA	$\begin{bmatrix} 2.009 & -1.009 \\ -1.009 & 2.009 \end{bmatrix}$
RNGA	$\begin{bmatrix} 1.563 & -0.5628 \\ -0.5628 & 1.563 \end{bmatrix}$
NI	0.4977

Diagonal Elements	ETF g_{ii}^*	ETF for Integrity $\hat{g}_{ii}$
$g_{11} = \frac{12.8}{16.7s+1}e^{-s}$	$g_{11}^* = \frac{6.37}{13s+1}e^{-0.778s}$	$\hat{g}_{11} = \frac{12.8}{16.7s+1}e^{-1s} = g_{11}$
$g_{22} = \frac{-19.4}{14.4s+1}e^{-3s}$	$g_{22}^* = \frac{-6.54}{6.08s+1}e^{-3.9s}$	$\hat{g}_{22} = \frac{-19.4}{14.4s+1}e^{-3s} = g_{22}$

i.1.3 *System 3: Vinante & Luyben*

$$G_3 = \begin{bmatrix} \frac{-2.2}{7s+1}e^{-s} & \frac{1.3}{7s+1}e^{-0.3s} \\ \frac{-2.8}{9.5s+1}e^{-1.8s} & \frac{4.3}{9.2s+1}e^{-0.35s} \end{bmatrix}$$

(After Loop Pairing 1 – 1/2 – 2)		
	Model	Half-Rule
g_{11}	$\frac{-2.2}{7s+1}e^{-s}$	-
g_{12}	$\frac{1.3}{7s+1}e^{-0.3s}$	-
g_{21}	$\frac{-2.8}{9.5s+1}e^{-1.8s}$	-
g_{22}	$\frac{4.3}{9.2s+1}e^{-0.35s}$	-

RGA	$\begin{bmatrix} 1.625 & -0.6254 \\ -0.6254 & 1.625 \end{bmatrix}$
RNGA	$\begin{bmatrix} 1.554 & -0.5537 \\ -0.5537 & 1.554 \end{bmatrix}$
NI	0.6152

Diagonal Elements	ETF g_{ii}^*	ETF for Integrity $\hat{g}_{ii}$
$g_{11} = \frac{-2.2}{7s+1}e^{-s}$	$g_{11}^* = \frac{-1.35}{6.69s+1}e^{-0.956s}$	$\hat{g}_{11} = \frac{-2.2}{7s+1}e^{-1s} = g_{11}$
$g_{22} = \frac{4.3}{9.2s+1}e^{-0.35s}$	$g_{22}^* = \frac{4.48}{8.41s+1}e^{-1.59s}$	$\hat{g}_{22} = \frac{4.3}{9.2s+1}e^{-0.35s} = g_{22}$

i.1.4 System 4: Industrial-Scale Polymerization Reactor

$$G_4 = \begin{bmatrix} \frac{22.89}{4.572s+1}e^{-0.2s} & \frac{-11.64}{1.807s+1}e^{-0.4s} \\ \frac{4.689}{2.174s+1}e^{-0.2s} & \frac{5.80}{1.801s+1}e^{-0.4s} \end{bmatrix}$$

(After Loop Pairing 1 – 1/2 – 2)		
	Model	Half-Rule
g_{11}	$\frac{22.89}{4.572s+1}e^{-0.2s}$	-
g_{12}	$\frac{-11.64}{1.807s+1}e^{-0.4s}$	-
g_{21}	$\frac{4.689}{2.174s+1}e^{-0.2s}$	-
g_{22}	$\frac{5.80}{1.801s+1}e^{-0.4s}$	-

RGA	$\begin{bmatrix} 0.7087 & 0.2913 \\ 0.2913 & 0.7087 \end{bmatrix}$
RNGA	$\begin{bmatrix} 0.5482 & 0.4518 \\ 0.4518 & 0.5482 \end{bmatrix}$
NI	1.411

Diagonal Elements	ETF g_{ii}^*	ETF for Integrity $\hat{g}_{ii}$
$g_{11} = \frac{22.89}{4.572s+1}e^{-0.2s}$	$g_{11}^* = \frac{32.3}{3.54s+1}e^{-0.155s}$	$\hat{g}_{11} = \frac{32.3}{4.57s+1}e^{-0.2s} \neq g_{11}$
$g_{22} = \frac{5.80}{1.801s+1}e^{-0.4s}$	$g_{22}^* = \frac{8.18}{1.39s+1}e^{-0.309s}$	$\hat{g}_{22} = \frac{8.18}{1.8s+1}e^{-0.4s} \neq g_{22}$

ii Appendix: Robust Stability for Discrete-time Control Loop

Assume that $g_p(z)$ is any member of the family Π^* of plants[43],

$$\Pi^* = \left\{ g_p(z) : \frac{|g_p(e^{j\omega T}) - g_m(e^{j\omega T})|}{|g_m(e^{j\omega T})|} \leq \overline{\Delta}^*(\omega) \right\}$$

which have the same number of RHP poles that do not become unobservable for the sampling. Every member of Π^* satisfies,

$$g_p(z) = g_m(z) (1 + \Delta^*(z)) \quad (\text{ii.1})$$

where $\Delta^*(z)$ is the *Multiplicative Uncertainty* corresponding to $g_p(z)$ with the bound on the allowed $\Delta^*(z)$ being,

$$|\Delta^*(e^{j\omega T})| \leq \overline{\Delta}^*(\omega) \quad (\text{ii.2})$$

Also assume that the nominal closed loop is stable. Then, the discrete-time closed loop system is *robustly* stable if and only if the nominal primary loop's tracking transfer function S_t (tracking transfer function without a setpoint filter) satisfies the following bound,

$$\|(S_t(e^{j\omega T})) \Delta^*(e^{j\omega T})\|_\infty < 1 \quad (\text{ii.3})$$

or

$$|(S_t(e^{j\omega T})) \Delta^*(e^{j\omega T})| < 1 \quad \forall \omega \in \left[0 \quad \frac{\pi}{T} \right]$$

The condition for Robust Stability given in Eq.(ii.3) can be rearranged as,

$$|\Delta^*(e^{j\omega T})| < \frac{1}{|S_t(e^{j\omega T})|} = \Delta_{lim}^*(\omega) \quad \forall \left(0 \leq \omega \leq \frac{\pi}{T} \right) \quad (\text{ii.4})$$

where $\Delta_{lim}^*(\omega)$ represents the *Robustness Limit*; the multiplicative uncertainty should be below this for all frequencies.

iii Appendix: Worst-case Multiplicative Uncertainty

The nominal FOPTD (First Order Plus Time Delay) model of a system is,

$$g_m = \frac{K}{\tau s + 1} e^{-Ls}$$

The real plant is some unknown member of the family of plants Π ,

$$\Pi = \left\{ g_p : \frac{|g_p(j\omega) - g_m(j\omega)|}{|g_m(j\omega)|} \leq \overline{\Delta}(\omega) \right\} \quad (\text{iii.1})$$

where g_p denotes a single member of Π . Every member of Π satisfies,

$$g_p = g_m(1 + \Delta) \quad (\text{iii.2})$$

where Δ is the *Multiplicative Uncertainty* corresponding to g_p with the bound on the allowed Δ being,

$$|\Delta(j\omega)| \leq \overline{\Delta}(\omega) \quad \forall \omega \quad (\text{iii.3})$$

where $\overline{\Delta}(\omega)$ is the *Multiplicative Uncertainty Bound*. Δ is also assumed to be stable. At a particular frequency ω , $\Pi(\omega)$ is a disc-shaped region on the nyquist plot of $g_m(s)$, with $g_m(j\omega)$ as the center and radius $|g_m(j\omega)\overline{\Delta}(\omega)|$.

Although the uncertainty is thus expressed as *Unstructured Uncertainty*, we assume that Π approximates (as best as possible) the uncertainty region resulting from a 100 $\lambda\%$ uncertainty in the parameters of g_m where ($\lambda < 1$). Also, for a given 100 $\lambda\%$ parametric (or structured) uncertainty, the worst case plant g_{pw} [43] is known to be ,

$$g_{pw} = \frac{K(1 + \lambda)}{\tau(1 - \lambda)s + 1} e^{-L(1+\lambda)s} \quad (\text{iii.4})$$

Then, since g_{pw} is a member of Π , the Multiplicative Uncertainty Δ_w corresponding to the 100 $\lambda\%$ worst-case plant g_{pw} is,

$$g_{pw} = g_m(1 + \Delta_w)$$

$$\begin{aligned} \therefore \Delta_w &= \frac{g_{pw}}{g_m} - 1 \\ &= \left[(1 + \lambda) e^{-L\lambda s} \frac{(\tau s + 1)}{\tau(1 - \lambda)s + 1} \right] - 1 \end{aligned} \quad (\text{iii.5})$$

where Δ_w is referred to as the *worst-case multiplicative uncertainty*. Δ_w in conjunction with the Robust Stability Condition in Equation(3.3.3) can be used to determine the upper limit on the controller gain for robust stability.

iv Appendix: Gain Margin Phase Margin PID tuning method

Conventionally, to control a continuous-time system $g_p(s)$ whose nominal model is,

$$g_m(s) = \left(\frac{K}{\bar{\tau}s^2 + \tau s + 1} \right) e^{-Ls}$$

a continuous-time PID controller,

$$g_c(s) = \left(K_p + \frac{K_i}{s} + K_d s \right) \quad (\text{iv.1})$$

is designed and implemented in a unity feedback configuration. In the *Gain Margin Phase Margin method*, the tuning of the PID controller is decided as,

$$\begin{bmatrix} K_p \\ K_i \\ K_d \end{bmatrix} = \frac{\pi}{2A_m L K} \begin{bmatrix} \tau \\ 1 \\ \bar{\tau} \end{bmatrix} \quad (\text{iv.2})$$

where A_m is the desired open loop gain margin (*in amplitude ratio*), typically between 2 and 5, which the user can set.

v Appendix: Derivation of GPC Control Law

This section will present the derivation of the control law of the unconstrained SISO GPC in two ways - a general method and a predictor separate method.

The GPC algorithm generates a control action by typically minimizing a quadratic cost of the form,

$$J = (Y - W)^T (Y - W) + U^T (\lambda I) U$$

where,

$$Y = \underbrace{\begin{bmatrix} \hat{y}_p(t + d + N_a | t) \\ \hat{y}_p(t + d + N_a + 1 | t) \\ \vdots \\ \hat{y}_p(t + d + N_b | t) \end{bmatrix}}_{(N_b - N_a + 1) \times 1}$$

$$W = \underbrace{\begin{bmatrix} w(t + d + N_1) \\ \vdots \\ w(t + d + N_2 - 1) \\ w(t + d + N_2) \end{bmatrix}}_{(N_b - N_a + 1) \times 1} = \underbrace{\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}}_{(N_b - N_a + 1) \times 1} w(t) \equiv \mathbf{1} w(t)$$

$$U = \underbrace{\begin{bmatrix} \Delta u(t) \\ \Delta u(t + 1) \\ \vdots \\ \Delta u(t + N_u - 1) \end{bmatrix}}_{N_u \times 1}$$

where the prediction $\hat{y}_p(t + j + d | t)$ is obtained from the CARIMA model of the plant in Eq.(1.2.1), $w(t)$ is the set-point signal, $u(t)$ is the control action and λ is the control action weighting. N_1 and N_2 describe the prediction horizon from $t + d + N_1$ to $t + d + N_2$ while N_u denotes the control horizon.

The prediction $\hat{y}_p(t + j + d | t)$ can be derived from the CARIMA model in two ways depending on which the structure of GPC can be studied in two ways. The derivation of $\hat{y}_p(t + j + d | t)$ by the general method can be found in refCamacho and Bordons [10] and the derivation with the optimal predictor as a separate block can be found in refNormey-Rico and Camacho [52].

v.1 General Derivation

The Prediction Vector Y can be expressed in terms of the vector of future control actions U as,

$$Y = GU + P$$

where,

$$P = I \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix} + F \begin{bmatrix} y_p(t) \\ y_p(t-1) \\ \vdots \\ y_p(t-n_a) \end{bmatrix}$$

If the unconstrained optimization $\min_U J$ is solved analytically, and U 's first element $\Delta u(t)$ is extracted, an explicit control law is obtained,

$$\Delta u(t) = e_1 (G^T G + \lambda I)^{-1} G^T [W - P]$$

where e_1 is $\begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}_{1 \times N_u}$.

$$\Delta u(t) = \underbrace{[k_1]}_{1 \times 1} w(t) - \underbrace{[k_2]}_{(n_a+1) \times 1} \begin{bmatrix} y_p(t) \\ y_p(t-1) \\ \vdots \\ y_p(t-n_a) \end{bmatrix} - \underbrace{[k_3]}_{(n_b+d) \times 1} \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-(n_b+d)) \end{bmatrix}$$

This is also compactly expressed in transfer function form as,

$$\Delta u(t) = k_1 w(t) - K_2 (z^{-1}) y_p(t) - K_3 (z^{-1}) \Delta u(t-1)$$

where K_2 and K_3 are the transfer function versions of k_2 and k_3 respectively. This can be rearranged to get,

$$u(t) = g_c (z^{-1}) \{p(z^{-1}) w(t) - y_p(t)\}$$

where $C(z^{-1})$ is the equivalent or primary controller and $p(z^{-1})$ is the setpoint filter,

$$\begin{aligned} C(z^{-1}) &= \frac{K_2}{[1 + K_3 z^{-1}] \Delta} \\ p(z^{-1}) &= \frac{k_1}{K_2} \end{aligned}$$

Figure v.1: Unconstrained GPC control structure - Implementation Form

The structure of the single GPC loop is shown in Figure(v.1). The closed loop transfer function of the unconstrained single GPC loop will be of the form,

$$y_p(t) = p \times T \times w(t) + S \times d_o(t)$$

where,

$$T = \frac{Cg_p}{1 + Cg_p} = \frac{K_2g_p}{(1 + K_3z^{-1})\Delta + K_2g_p}$$

$$S = \frac{1}{1 + Cg_p} = (1 - T)$$

v.2 Predictor Seperate Derivation

The prediction vector in this case is

$$Y = GU + \bar{P}$$

where $\bar{P}$ is the **Free Response Vector** in matrix form,

$$\bar{P} = \bar{I} \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-n_b) \end{bmatrix} + \bar{F} \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix}$$

(It is worth noting that the following applies to the case when $d > n_a$. When $d \leq n_a$, a slightly different derivation will be needed.)

The explicit control law in this case is,

$$\Delta u(t) = e_1 (G^T G + \lambda I)^{-1} G^T [W - \bar{P}]$$

and when the $\overline{P}$ vector is substituted into this,

$$\Delta u(t) = k_1 w(t) - l_2 \begin{bmatrix} \hat{y}_p(t+d|t) \\ \hat{y}_p(t+d-1|t) \\ \vdots \\ \hat{y}_p(t+d-n_a|t) \end{bmatrix} - l_3 \begin{bmatrix} \Delta u(t-1) \\ \Delta u(t-2) \\ \vdots \\ \Delta u(t-n_b) \end{bmatrix}$$

The Control Law in the transfer function form is,

$$\Delta u(t) = k_1 w(t) - L_2(z^{-1})\hat{y}_p(t+d|t) - L_3(z^{-1})z^{-1}\Delta u(t)$$

Rearranging its terms,

$$\Rightarrow \Delta u(t) = \frac{k_1}{[1 + L_3 z^{-1}]} w(t) - \frac{L_2}{[1 + L_3 z^{-1}]} \hat{y}_p(t+d|t) \quad (\text{v.1})$$

The signal $\hat{y}_p(t+d|t)$ can be derived analytically to,

$$\hat{y}_p(t+d|t) = \{g_m^* u(t) + F_r(z^{-1})[y_p(t) - y_m(t)]\} \quad (\text{v.2})$$

where,

$$F_r(z^{-1}) = \left[\frac{\sum_{i=0}^{n_a} (l_2^i F_{d-i})}{L_2(z^{-1})} \right]$$

Eq.(v.2) is the transfer function form of the **optimal predictor** and F_r is called the optimal predictor filter. The F_{d-i} term in the summation can be obtained by recursively applying the diophantine equation Camacho and Bordons [10],

$$1 = \tilde{A}(z^{-1}) E_j(z^{-1}) + z^{-j} F_j(z^{-1})$$

where $\tilde{A} = \Delta A$. F_j is the reminder of j divisions of 1 by $\tilde{A}$. (**Note:** F_j is a polynomial and is not to be confused with the predictor filter which is a transfer function denoted by F_r).

The overall control loop is depicted using Eq.(v.1) and Eq.(v.2), as shown in Figure(v.2).

Figure v.2: Unconstrained GPC control loop - Predictor Separate Structure: Case $d > n_a$

The output of the plant g_p is $y_p(t)$,

$$y_p(t) = g_p(z^{-1})u(t) + d_o(t)$$

where d_o is the disturbance. Substituting for u and simplifying,

$$y_p(t) = p \times T \times w(t) + S \times d_o(t)$$

where S is the sensitivity & T is the complementary-sensitivity of the 1° control loop, and p is the prefilter.

$$T = \frac{C(z^{-1})g_p}{1 + C(z^{-1})g_p} = \frac{g_p L_2 F_r}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* + L_2 F_r (g_p - g_m)}$$

$$S = \frac{1}{1 + C(z^{-1})g_p} = \frac{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* (1 - z^{-d} F_r)}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* + L_2 F_r (g_p - g_m)}$$

where,

$$C(z^{-1}) = \frac{L_2 F_r}{\Delta(1 + L_3 z^{-1}) + L_2 g_m^* (1 - z^{-d} F_r)}$$

$$\& \quad p(z^{-1}) = \left[\frac{k_1}{L_2 F_r} \right]$$

It should be noted that the prefilter p and the primary loop controller C derived by either method are numerically equivalent.