



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE DEPARTAMENTO DE INGENIERÍA
ELÉCTRICA

APLICACIÓN DE LA TÉCNICA PSO A LA DETERMINACIÓN DE FUNCIONES DE
LYAPUNOV CUADRÁTICAS COMUNES Y A SISTEMAS ADAPTABLES BASADOS EN
MODELOS DE ERROR

TESIS PARA OPTAR AL TÍTULO DE DOCTOR EN INGENIERÍA ELÉCTRICA

RODRIGO HERNÁN ORDÓÑEZ HURTADO

PROFESOR GUÍA:
MANUEL DUARTE MERMOUD

MIEMBROS DE LA COMISIÓN:
MARCOS ORCHARD CONCHA
JOSÉ PÉREZ CORREA
DANIEL SBARBARO HOFER

Este trabajo ha sido parcialmente financiado por CONICYT-CHILE a través del proyecto “Beca Doctorado Latinoamericano 2008”

SANTIAGO DE CHILE
JULIO 2012

“APLICACIÓN DE LA TÉCNICA PSO A LA DETERMINACIÓN DE FUNCIONES DE LYAPUNOV CUADRÁTICAS COMUNES Y A SISTEMAS ADAPTABLES BASADOS EN MODELOS DE ERROR”

La presente Tesis Doctoral explora el problema de la determinación de funciones de Lyapunov cuadráticas comunes (CQLF, por su sigla en inglés), en el marco de los sistemas conmutados, y el problema de la identificación en línea y control adaptable, en el marco de los sistemas adaptables basados en modelos de error. Ambos en el área de los sistemas dinámicos lineales y no lineales, y son resueltos aquí bajo el enfoque de la optimización basada en una herramienta llamada Optimización por Enjambre de Partículas (PSO, por su sigla en inglés).

Los problemas anteriormente mencionados son de gran importancia y trascendencia en la actualidad, pues el primero entrega los elementos para la determinación de la estabilidad de sistemas lineales conmutados, y el segundo se relaciona con el control de plantas de parámetros desconocidos.

Estos dos problemas poseen soluciones parciales, tanto desde el punto de vista de la optimización como de otros enfoques. Sin embargo, las soluciones existentes poseen beneficios demostrados, pero también limitaciones marcadas, que los siguen justificando como problemas abiertos.

En cuanto al problema de la determinación de CQLFs, en la presente Tesis Doctoral se desarrollan dos nuevas metodologías: i) una metodología basada en PSO para la determinación de la no-existencia de una CQLF, y ii) una metodología basada en PSO para el cálculo de una CQLF. Ambas metodologías presentan evidentes mejoras comparativas respecto de las mejores soluciones actuales, con base en indicadores de desempeño objetivos.

En el ámbito de los sistemas adaptables, el principal producto de la presente Tesis Doctoral es una metodología basada en PSO para el diseño de leyes de ajuste paramétrico en sistemas adaptables de tiempo discreto, representados por modelos de error. Desde este punto de vista, la investigación se centra en las propiedades de estabilidad que presenta el uso de PSO en sistemas adaptables, además de estudiar las ventajas comparativas respecto de técnicas tradicionalmente usadas como gradiente y mínimos cuadrados.

Dedicatoria

A los seres que me otorgan la luz para descubrir y recorrer nuevos y prósperos caminos. Mis guías en la universidad de la vida: Dios y la Diosa.

Al ser que con amor decidió traerme al mundo, y me regaló la curiosidad como virtud para explorar el universo: Mi madre, María del Socorro.

A la misma sangre que corre por mis venas, y que hacen de mi casa un verdadero hogar: Mis hermanos, Andrés Fernando y Eduardo José.

A mis tallos, mis ramas, mis hojas, mis raíces. A toda mi gente. Mi tribu: familia, parientes, amigos y compañeros.

A quien me quiere, me cuida, me entibia, me espera y me aguanta. Y a quién yo también: Claudia Viviana.

Agradecimientos

A Dios y la Diosa, por la maravillosa leyenda personal que me han dado. A Colombia, mi tierra natal, donde inició este viaje habiéndome formado como ingeniero. A Chile, al que acogí como mi segunda patria, en donde tuve la oportunidad de cursar el Doctorado.

A la Universidad de Chile, en especial a los académicos y administrativos que participaron en mi formación como Doctor. A mi Tutor y Mentor PhD Manuel Duarte Mermoud por ser un guía de excelencia, y a todo su grupo de investigación por apoyarme en mi desarrollo profesional. A los nuevos amigos y compañeros de Chile, dentro y fuera del Doctorado, quienes hicieron de esta etapa de mi vida una gran experiencia a nivel interpersonal. Al equipo de investigación del Instituto Hamilton de la Universidad Nacional de Irlanda, en especial a PhD Robert Shorten con quien trabajo en la ampliación de los alcances de mi investigación doctoral.

A toda mi familia, de sangre y de cercanía, por estar siempre conmigo a pesar de la distancia. A mi madre María del Socorro por darnos la vida, y darnos su vida, a mí y a mis hermanos: todo nuestros logros se los debemos a ella, a su entereza, a su integridad, a su entrega, a su amor. A mis hermanos Andrés Fernando y Eduardo José, por quererme tanto a pesar de haber sido el complicado hermano mayor. A mi tío Julio Ariel por creer en mí e impulsarme como un verdadero padre. A Claudia Viviana, quien ha caminado a mi lado y nutre mi corazón del más abrigador amor de mujer. A todos los viejos y queridos amigos, míos y de mi familia, que han sobrevivido a mí y no cesan de compartirme buena energía. A mis hijitos adoptivos Pelusa, Perro, Negra y Pepe: su existencia me trajo bastante de la felicidad que necesité para seguir en pie de guerra.

Finalmente, pero no menos importante, a los promotores financieros de esta experiencia: CONICYT-CHILE a través de los proyectos “Beca Doctorado Latinoamericano 2008” y FONDECYT No. 1090208, y el Programa de Financiamiento Basal “Centro de Tecnología para la Minería” FB0809 (Advanced Mining Technology Center - AMTC).

Índice general

1. Introducción	1
2. Marco teórico	5
2.1. Introducción	5
2.2. Sistemas lineales conmutados	5
2.2.1. Consideraciones matemáticas	6
2.2.2. El problema CQLF	7
2.2.3. Metodologías de cálculo de una CQLF	8
2.2.4. Determinación de la existencia/no-existencia de una CQLF	10
2.2.5. Algunos resultados previos	11
2.3. Sistemas adaptables	13
2.3.1. Sistemas adaptables discretos basados en modelos de error	14
2.4. Optimización global	17
2.4.1. Técnicas basadas en Computación Evolucionaria	19
2.5. Particle Swarm Optimization	20
2.5.1. Esquemas de configuración	22
2.5.2. Estabilidad de PSO	25
2.5.3. GCPSO: PSO con convergencia garantizada	27
2.5.4. Estado del arte de la técnica	32
3. PSO en el Problema CQLF	34
3.1. Introducción	34
3.2. Cálculo de una CQLF usando PSO	34
3.2.1. Funciones de fitness	35
3.2.2. Representación de las partículas	37
3.2.3. Análisis de desempeño de la metodología propuesta	38
3.2.4. Análisis de convergencia	46
3.3. Determinación de la no-existencia de una CQLF usando PSO	47
3.3.1. Diseño de la metodología	47
3.3.2. Resultados experimentales	51
4. PSO en el diseño de leyes de ajuste paramétrico	63
4.1. Introducción	63
4.2. Usando PSO en tiempo real	63
4.3. Diseño de las funciones de fitness	65
4.3.1. Modelo de Error 1	66
4.3.2. Modelo de Error 2	75

4.3.3. Modelo de Error 3	77
4.4. Resultados experimentales	79
4.4.1. Modelo de Error 1	80
4.4.2. Modelo de Error 2	101
4.4.3. Modelo de Error 3.	114
5. Conclusiones y trabajo futuro	124
5.1. PSO en sistemas conmutados	124
5.1.1. Cálculo de una CQLF	124
5.1.2. No existencia de una CQLF	125
5.1.3. Trabajo futuro	126
5.2. PSO en sistemas adaptables	126
5.2.1. Trabajo futuro	127
A. Toolbox de PSO	128
A.1. Introducción	128
A.2. Instalación (Versión Matlab)	128
A.3. Código fuente	129
Bibliografía	133
Nomenclatura	144

Índice de cuadros

2.1. Configuraciones con características convergentes.	23
3.1. Comparación de la robustez para la primera solución	45
3.2. Comparación de la robustez para la mejor solución	46
3.3. Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 1 , con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	54
3.4. Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 2, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	55
3.5. Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 3, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	56
3.6. Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 4, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	57
3.7. Muestra de la aplicación de la metodología basada en PSO para Ejemplo 5, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	58
3.8. Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 6.1, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	60
3.9. Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 6.2, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.	62

Índice de figuras

2.1.	Ejemplo de diagramas de fase para la conmutación de dos sistemas estables de segundo orden (Liberzon, 2003): (a) Primer sistema, (b) segundo sistema, (c) conmutación estable, y (d) conmutación inestable.	7
2.2.	Taxonomía de los métodos de optimización global. (Se conserva la denominación original en inglés por uniformidad con la literatura científica.) (Pérez-López, 2005)	18
2.3.	Hipercubo soporte M_k del espacio de muestreo μ_k , centrado alrededor del punto $\mathbf{g}_k + \omega \mathbf{v}_k$, como es definido por el paso de actualización de la posición usando GCPSO. El punto \mathbf{x}_{k+1} es un ejemplo de un punto de muestreo por el nuevo paso de actualización de velocidad. (Van Den Bergh, 2002)	30
2.4.	Intersección $C \cap B$. (Solis and Wets, 1981; Van Den Bergh, 2002)	31
3.1.	Tiempo de cálculo promedio hasta la primera solución factible para conjuntos de matrices triangulares superiores. (a) Matrices de 2×2 , (b) matrices de 3×3 , (c) matrices de 4×4 , y (d) matrices de 5×5 . (Cada marca: promedio de 20 medidas; líneas punteadas: proyecciones debido a complejidad de obtener datos; porcentajes: tasas de éxito; sin porcentajes: tasas del 100%; condiciones iniciales para PSO: aleatorias) (Ordóñez-Hurtado and Duarte-Mermoud, 2012).	40
3.2.	Tiempo de cálculo promedio hasta la primera solución factible para otra clase de conjuntos de N matrices en $\mathfrak{R}^{5 \times 5}$. (a) Matrices que conmutan, (b) matrices diagonales, (c) matrices comúnmente diagonalizables, y (d) matrices genéricas. (Cada marca: promedio de 20 medidas; líneas punteadas: proyecciones debido a complejidad de obtener datos; porcentajes: tasas de éxito; sin porcentajes: tasas del 100%; R.I.C.: condiciones iniciales aleatorias; P.I.C.: condiciones iniciales predefinidas) (Ordóñez-Hurtado and Duarte-Mermoud, 2012).	41
3.3.	Comparación gráfica para el cálculo de una CQLF en un conjunto de tres matrices de segundo orden que comparten una CQLF (Ordóñez-Hurtado and Duarte-Mermoud, 2012): a) matrices que conmutan, b) matrices triangulares, c) matrices negativas definidas, y d) matrices genéricas.	43
3.4.	Conjunto de CQLFs $\mathcal{P} = \begin{bmatrix} 1 & P_{12} \\ P_{12} & P_{22} \end{bmatrix}$ para A_1 , A_2 y A_3 (Shorten and Narendra, 2002) para el Sección 3.3.2.	53
3.5.	Análisis de singularidad gráfico para las matrices pencil (Ecuación 3.32) con A_1, A_2 dadas para el Ejemplo 6.1, en función de $\alpha \in [0, 1]$ (Ordóñez-Hurtado and Duarte-Mermoud, 2011b).	59

3.6. Análisis de singularidad gráfico para las matrices pencil (Ecuación 3.32) con A_1, A_2 dadas para el Ejemplo 6.2, en función de $\alpha \in [0, 1]$ (Ordóñez-Hurtado and Duarte-Mermoud, 2011b).	61
4.1. Representación gráfica del proceso de optimización.	64
4.2. Evolución gráfica de la optimización del funcional J_3 en \mathbb{R}^1	74
4.3. Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo I.	81
4.4. Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo I.	82
4.5. Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo I.	82
4.6. Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo I.	83
4.7. Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo II.	84
4.8. Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para GC-PSO J3, GC-PSO J4, GC-PSO J5 y GC-PSO J6. Señal de entrada utilizada: Tipo II.	84
4.9. Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo II.	85
4.10. Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo II.	85
4.11. Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo III.	86
4.12. Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo III.	87
4.13. Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo III.	87
4.14. Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo III.	88
4.15. Resultados de simulación para la Subsubsección 4.4.1.2: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2.	89
4.16. Resultados de simulación para la Subsubsección 4.4.1.2: Comparación de salidas y error de identificación para GC-PSOJ3 y GC-PSOJ4.	90
4.17. Resultados de simulación para la Subsubsección 4.4.1.2: Estimación y error paramétrico para NG, NLS, GC-PSO J1 y GC-PSO J2.	90

4.18. Resultados de simulación para la Subsubsección 4.4.1.2: Estimación y error paramétrico para GC-PSOJ3 y GC-PSOJ4.	91
4.19. Resultados de simulación para la Subsubsección 4.4.1.3: Comparación de salidas y error de identificación para NG, NLS y GC-PSOJ2.	92
4.20. Resultados de simulación para la Subsubsección 4.4.1.3: Comparación de salidas y error de identificación para GC-PSOJ3 y GC-PSOJ4.	93
4.21. Resultados de simulación para la Subsubsección 4.4.1.3: Estimación y error paramétrico para NG, NLS y GC-PSOJ2.	93
4.22. Resultados de simulación para la Subsubsección 4.4.1.3: Estimación y error paramétrico para GC-PSOJ3 y GC-PSOJ4.	94
4.23. Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas estimada y real, y error de identificación para NG, NLS y GC-PSOJ2.	97
4.24. Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas estimada y real, y error de identificación para GC-PSOJ3 y GC-PSOJ4.	97
4.25. Resultados de simulación para la Subsubsección 4.4.1.4: Estimación de los parámetros de la planta y error paramétrico para NG, NLS y GC-PSOJ2.	98
4.26. Resultados de simulación para la Subsubsección 4.4.1.4: Estimación de los parámetros de la planta y error paramétrico para GC-PSOJ3 y GC-PSOJ4.	98
4.27. Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas real y del modelo de referencia, y error de control para NG, NLS y GC-PSOJ2.	99
4.28. Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas real y del modelo de referencia, y error de control para GC-PSOJ3 y GC-PSOJ4.	100
4.29. Resultados de simulación para la Subsubsección 4.4.1.4: Cálculo de los parámetros del controlador y su error paramétrico para NG, NLS y GC-PSOJ2.	100
4.30. Resultados de simulación para la Subsubsección 4.4.1.4: Cálculo de los parámetros del controlador y su error paramétrico para GC-PSOJ3 y GC-PSOJ4.	101
4.31. Resultados de simulación para la Subsubsección 4.4.2.1: Comparación de las salidas real y estimada, y error de identificación para NG, NLS y GC-PSOJ2.	103
4.32. Resultados de simulación para la Subsubsección 4.4.2.1: Comparación de las salidas real y estimada, y error de identificación para GC-PSOJ3 y GC-PSOJ4.	103
4.33. Resultados de simulación para la Subsubsección 4.4.2.1: Estimación de los parámetros de la planta y error paramétrico para NG, NLS y GC-PSOJ2.	104
4.34. Resultados de simulación para la Subsubsección 4.4.2.1: Estimación de los parámetros de la planta y error paramétrico para GC-PSOJ3 y GC-PSOJ4.	104
4.35. Resultados de simulación para la Subsubsección 4.4.2.1: Evolución de $\epsilon(t)$ para NG, NLS, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4.	105
4.36. Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas estimada y real, y error de identificación para NG, NLS y GC-PSOJ2.	108
4.37. Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas estimada y real, y error de identificación para GC-PSOJ3 y GC-PSOJ4.	109
4.38. Resultados de simulación para la Subsubsección 4.4.2.2: Estimación de los parámetros de la planta y error paramétrico para NG, NLS y GC-PSOJ2.	109
4.39. Resultados de simulación para la Subsubsección 4.4.2.2: Estimación de los parámetros de la planta y error paramétrico para GC-PSOJ3 y GC-PSOJ4.	110

4.40. Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas real y del modelo de referencia, y error de control para NG, NLS y GC-PSOJ2.	111
4.41. Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas real y del modelo de referencia, y error de control para GC-PSOJ3 y GC-PSOJ4.	111
4.42. Resultados de simulación para la Subsubsección 4.4.2.2: Cálculo de los parámetros del controlador y su error paramétrico para NG, NLS y GC-PSOJ2.	112
4.43. Resultados de simulación para la Subsubsección 4.4.2.2: Cálculo de los parámetros del controlador y su error paramétrico para GC-PSOJ3 y GC-PSOJ4.	113
4.44. Resultados de simulación para la Subsubsección 4.4.2.2: Evolución de $\epsilon(t)$ para NG, NLS, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4.	113
4.45. Resultados de simulación para la Subsubsección 4.4.3.1: Evolución del $e(t)$ y $\epsilon(t)$ para NG y GC-PSOJ2.	115
4.46. Resultados de simulación para la Subsubsección 4.4.3.1: Evolución del error y de para GC-PSOJ3 y GC-PSOJ4.	116
4.47. Resultados de simulación para la Subsubsección 4.4.3.1: Estimación de los parámetros y error paramétrico para NG, NLS y GC-PSOJ2.	117
4.48. Resultados de simulación para la Subsubsección 4.4.3.1: Estimación de los parámetros y error paramétrico para GC-PSOJ3 y GC-PSOJ4.	117
4.49. Resultados de simulación para la Subsubsección 4.4.3.2: Comparación salida real y salida del modelo de referencia para NG y GC-PSOJ2.	119
4.50. Resultados de simulación para la Subsubsección 4.4.3.2: Comparación salida real y salida del modelo de referencia para GC-PSOJ3 y GC-PSOJ4.	120
4.51. Resultados de simulación para la Subsubsección 4.4.3.2: Estimación de los parámetros del controlador para NG, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4.	121
4.52. Resultados de simulación para la Subsubsección 4.4.3.2: Comparación Z y \hat{Z} para NG y GC-PSOJ2.	121
4.53. Resultados de simulación para la Subsubsección 4.4.3.2: Comparación Z y \hat{Z} para GC-PSOJ3 y GC-PSOJ4.	122
4.54. Resultados de simulación para la Subsubsección 4.4.3.2: Señales para GC-PSOJ4(0.99).123	123

1. Introducción

La presente Tesis Doctoral explora el problema de la determinación de funciones de Lyapunov cuadráticas comunes (CQLFs), en el marco de los sistemas conmutados (Liberzon, 2003), y el problema de la identificación en línea y control adaptable, en el marco de los sistemas adaptables basados en modelos de error (Duarte-Mermoud and Narendra, 1996), ambos en el área de los sistemas dinámicos lineales y no lineales, bajo el enfoque de la optimización basada en una herramienta llamada Optimización por Enjambre de Partículas (PSO) (Kameyama, 2009). Estos dos problemas, que son de gran importancia y trascendencia en la actualidad, tienen soluciones parciales desde el punto de vista de la optimización u otros enfoques, con beneficios demostrados pero también con limitaciones marcadas que los siguen justificando como problemas abiertos.

El problema de la determinación de una CQLF aparece al extender la teoría de estabilidad de Lyapunov al campo de sistemas dinámicos lineales conmutados (SLSs) (Lin and Antsaklis, 2009). Los SLSs son aquellos sistemas cuya matriz de evolución conmuta en ciertos instantes de tiempo (aleatorios o predeterminados), tomando valores constantes durante ciertos intervalos. El aseguramiento de la estabilidad global uniforme de un sistema conmutado es un problema que puede resolverse al garantizar la existencia de una función de Lyapunov cuadrática, cuya derivada a lo largo de todas las posibles soluciones del sistema, y que satisface simultáneamente todas las ecuaciones de Lyapunov generadas por los subsistemas que componen el sistema conmutado. No obstante, este problema se presenta con grado de dificultad creciente en la medida que aumentan las variables que intervienen en el problema, relacionadas directamente con el orden y la cantidad de sistemas analizados. Las dos ramas principales en que se dividen las alternativas de solución existentes son: 1) la determinación de condiciones de existencia/no-existencia de una CQLF, y 2) el cálculo de una CQLF. En el ámbito de las soluciones dadas a la determinación de condiciones de existencia/no-existencia, es importante decir que éstas se inclinan hacia la derivación de condiciones analíticas. Por otro lado, las soluciones al cálculo de una CQLF se ven marcadamente inclinadas hacia la exploración numérica, ya sea por el uso de herramientas que analizan desigualdades matriciales (LMIs) o herramientas de optimización local (gradiente), entre las más importantes.

Como segunda instancia se tienen los sistemas adaptables basados en modelos de error, con sus dos ramas fundamentales: la identificación en línea y el control adaptable de sistemas dinámicos (Narendra and Annaswamy, 1989; Tao, 2003; Ioannou and Fidan, 2006). La identificación en línea se refiere normalmente a la estimación de los parámetros desconocidos de un modelo del sistema, utilizando métodos de entrada-salida en tiempo real. El control adaptable se refiere al diseño de una estrategia de control para un sistema cuyos parámetros son desconocidos; la principal idea detrás de esta metodología está en el diseño de un controlador cuyos parámetros se auto ajustan a lo largo del tiempo. Bajo esta consideración, puede verse a la identificación en línea como un componente del control adaptable, más específicamente hablando, del control adaptable indirecto. Un controlador adaptable indirecto está formado por la combinación de un estimador en

línea de los parámetros de la planta, el cual proporciona en cada instante el valor estimado de los parámetros desconocidos de la planta, y una acción de control que es deducida a partir de los parámetros estimados. En contraste, el controlador adaptable directo hace ajustes directamente sobre los parámetros del controlador sin necesidad de estimar los parámetros de la planta. Cualquiera de estos dos enfoques se fundamenta en el argumento de que la salida de un sistema provee información interna del sistema, y entonces a través de ella se puede aprender sobre el estado y la variación de los parámetros de la planta. Luego, con base en este aprendizaje se puede diseñar un controlador con ganancias adaptables mediante una ley de ajuste para lograr los objetivos de control deseados. El diseño de tal ley de ajuste es crucial para las propiedades de estabilidad del sistema adaptable resultante, por lo cual la exigencia de condiciones que garanticen la estabilidad del sistema en todo momento es uno de los aspectos primordiales en la etapa de análisis.

En la actualidad, los esfuerzos en el área de los sistemas conmutados se han concentrado en el estudio del desempeño dinámico, el análisis de estabilidad y el diseño de controladores (Lin and Antsaklis, 2009). Dado que la estabilidad es el punto crítico al final de cuentas, es necesario analizar en detalle este aspecto para poder garantizar la seguridad de los operadores y evitar daños a los equipos y demás personas que están relacionadas directamente con el proceso (sistema conmutado). Como el simple hecho de tratar con subsistemas estables no garantiza que el sistema conmutado global sea estable, aparece entonces el concepto de estabilidad de sistemas conmutados, cuyos principales avances se basan, entre otros, en el estudio de las CQLFs: la existencia de una CQLF para un conjunto de subsistemas dinámicos que componen un sistema conmutado garantiza la estabilidad de ese sistema conmutado (Lin and Antsaklis, 2009). En este orden de ideas, el hecho de abordar el problema de la determinación de una CQLF usando el enfoque de la optimización global, por medio de técnicas de computación evolucionaria, implica una innovación en cuanto al modo de resolver el problema. Adicionalmente, la originalidad de las metodologías desarrolladas, entre las cuales se encuentran la determinación de la no-existencia de una CQLF usando PSO, se enmarca también en el modo de acondicionar este problema a la técnica de computación evolucionaria escogida, lo que incluye aspectos tan importantes como: la adecuación de condiciones conocidas de existencia/no-existencia de una CQLF a la técnica PSO, la determinación de nuevas condiciones de existencia/no-existencia de una CQLF que puedan ser interpretadas como una función objetivo a optimizar con PSO, y el estudio comparativo con las mejores alternativas de solución existentes. Específicamente hablando, el hecho de asumir el problema de la definición de condiciones que garanticen la existencia/no-existencia de una CQLF, analizado desde el punto de vista de la optimización numérica, plantea una nueva forma de comprender el problema y una nueva forma de enfrentarlo, y con ello se crea una nueva línea de soluciones en el área. En lo referente a las funciones de Lyapunov, la computación evolucionaria sólo ha llegado a explorar la forma en que se puede calcular una función de Lyapunov individual para un sistema dinámico (Banks, 2002; Grosman and Lewin, 2005). En este ámbito, en la presente Tesis Doctoral se desarrollan dos nuevas metodologías: i) una metodología basada en PSO para la determinación de la no-existencia de una CQLF, y ii) una metodología basada en PSO para el cálculo de una CQLF. Ambas metodologías presentan evidentes mejoras comparativas respecto de las mejores soluciones actuales, con base en indicadores de desempeño objetivos.

En relación al control adaptable, se tiene que para su posicionamiento como una práctica aceptable de control avanzado ha sido necesaria una larga etapa de evolución constante, que hasta la actualidad se mantiene en proceso de renovación. Y es que hoy en día el hecho de tratar con técnicas de

control adaptable implica tener amplio conocimiento en áreas como el diseño de controladores para sistemas lineales e invariantes en el tiempo, teoría de sistemas lineales y no lineales, y un alto grado de conocimiento matemático. A lo largo de su evolución, el control adaptable ha resuelto diversos puntos críticos en aspectos como la definición del esquema de control, el diseño de leyes de ajuste, y las respectivas demostraciones de estabilidad y robustez que garanticen el éxito del enfoque. En particular, los desarrollos en la teoría de control, la teoría de identificación de sistemas y estimación de parámetros, y la teoría de estabilidad basada en Lyapunov, han jugado un papel muy importante en la reformulación y el rediseño del control adaptable, tanto así que sin ellas este esquema de control no tendría las bases sólidas que en la actualidad posee, y su campo de aplicación sería muy reducido. El éxito de los diversos esquemas propuestos que aseguran propiedades de estabilidad, acompañado por el creciente desarrollo tecnológico que permite la implementación de controladores cada vez más complejos, han hecho posible un considerable incremento en el interés depositado en esta técnica y en sus áreas de aplicación. Una característica importante de mencionar respecto del panorama descrito para el control adaptable, es que las técnicas usadas para la identificación de parámetros y síntesis de leyes de ajuste de los parámetros de los controladores han mostrado una marcada tendencia a basarse en el método del gradiente combinado con el método directo de Lyapunov (Narendra and Annaswamy, 1989; Tao, 2003; Ioannou and Fidan, 2006), lo cual es debido al éxito de una solución elegante con muy buenas propiedades de estabilidad. Recientes estudios sobre herramientas para la identificación paramétrica en línea han sido desarrolladas en el marco de otras áreas de investigación (Angelov et al., 2004; Trejo et al., 2006; Alizadeh et al., 2007), pero aún hoy día no son aplicadas abiertamente en el control adaptable, y tampoco se ha pensado con severidad la posibilidad de llegar a ser una práctica confiable que reemplace la tradicional técnica del gradiente descendiente. La falta de investigación en este aspecto lleva a un consecuente desconocimiento de posibles beneficios que pudieran traer herramientas como PSO y sus diversas versiones. Como resultado de esto, otro de los productos principales de esta Tesis Doctoral es una novedosa metodología basada en PSO para el diseño de leyes de ajuste paramétrico en sistemas adaptables de tiempo discreto, representados por modelos de error. Desde este punto de vista, la presente Tesis Doctoral se constituye como un estudio de las ventajas comparativas con técnicas tradicionales como gradiente y mínimos cuadrados y, sobretodo, las propiedades de estabilidad que ofrece el uso de PSO.

A continuación se presentan los objetivos con que dieron inicio el estudio presentado en esta Tesis Doctoral, los cuales están directamente relacionados con la organización de sus contenidos:

Objetivo general Desarrollar soluciones a la determinación de funciones de Lyapunov cuadráticas comunes y a los problemas de identificación en línea y control adaptable basados en modelos de error, utilizando el algoritmo PSO estándar, que posean ventajas significativas respecto de las mejores soluciones existentes en la literatura especializada, medidas a través de índices objetivos de desempeño y robustez.

Objetivos específicos

1. Diseñar algoritmos basados en PSO, que determinen la existencia/no-existencia de una CQLF para un sistema lineal conmutado variante en el tiempo.

2. Diseñar un método de cálculo de una CQLF basado en PSO, que mejore los resultados obtenidos por métodos como gradiente y resolución de desigualdades matriciales lineales, en base a criterios e indicadores objetivos como tiempos de cálculo, orden y número de matrices a analizar, y sensibilidad del método frente a variaciones paramétricas, entre los más importantes.
3. Diseñar leyes de ajuste estables para sistemas adaptables basados en modelos de error, utilizando PSO, para su utilización en identificación en línea y control adaptable, que muestren ventajas respecto del método del gradiente en base a criterios e indicadores objetivos como tiempos de convergencia, estabilidad de los transitorios, e índices de error estándar, entre los más importantes
4. Realizar un análisis comparativo entre las metodologías de solución diseñadas para el problema CQLF y basadas en PSO, y las técnicas existentes tales como las basadas en gradiente y en resolución de desigualdades matriciales lineales, respecto a criterios e indicadores descritos en el Objetivo 2.
5. Realizar un análisis comparativo entre las metodologías de solución diseñadas para el problema de identificación en línea y control adaptable basadas en PSO, y la técnica basada en gradiente, respecto a criterios e indicadores como los descritos en el Objetivo 3.

Con la anterior recapitulación de los objetivos de la Tesis, queda clara su relación con el organización del presente documento, descrita a continuación. El Capítulo 2 presenta el marco teórico y la revisión bibliográfica sobre estabilidad de sistemas conmutados, diseño de leyes de ajuste en sistemas adaptables, técnicas de optimización global, y Optimización por Enjambre de Partículas. El Capítulo 3 aborda el desarrollo de los objetivos específicos 1, 2 y 4, presentando el diseño analítico y la validación experimental de un par de metodologías relacionadas con la determinación de CQLFs usando PSO: una para el cálculo de una CQLF, y otra para la determinación de la no-existencia de una CQLF. El Capítulo 4 aborda los objetivos específicos 3 y 5, presentando el diseño analítico y la validación experimental de una metodología para el diseño de leyes de ajuste en sistemas adaptables discretos basada en PSO. Finalmente, el Capítulo 5 contiene las conclusiones generales y específicas de todos los desarrollos presentados en el Capítulo 3 y el Capítulo 4.

2. Marco teórico

2.1. Introducción

El presente capítulo contiene las bases teóricas necesarias para abordar los temas en que se desenvuelve la presente Tesis, así como también los principales y más relevantes resultados dentro de las áreas de estudio.

Consideraciones generales En adelante, denotaremos por $V(x) > 0$ y $P > 0$ a una función o una matriz positiva definida, y $V(x) \geq 0$ y $P \geq 0$ a una función o una matriz positiva semidefinida. Similarmente, la notación $V(x) < 0$ y $P < 0$ será usada para una función o una matriz negativa definida, y $V(x) \leq 0$ y $P \leq 0$ para una función o una matriz negativa semidefinida.

2.2. Sistemas lineales conmutados

Un sistema lineal conmutado (SLS por *switched linear system*) es el tipo de sistema dinámico lineal cuya matriz de evolución varía en el tiempo, dependiendo de una señal llamada regla de conmutación. Los modelos matemáticos de los SLS representan más adecuadamente la dinámica de diversos tipos de sistemas en las gran variedad de áreas de aplicación (Liberzon, 2003; Lin and Antsaklis, 2009). En la actualidad, los principales esfuerzos en el campo de los SLS están enfocados en el estudio del desempeño dinámico, diseño de controladores, y análisis de estabilidad. Este último es un aspecto crítico en lo que se refiere a seguridad de los operadores y la integridad del hardware implicado, los cuales son los principalmente afectados cuando hay fallas en el sistema. Sin embargo, a diferencia del análisis de estabilidad de sistemas lineales LTI, el análisis de estabilidad de este tipo de sistemas no solamente requiere que los subsistemas que lo conforman sean estables, puesto que una determinada regla de conmutación puede producir la inestabilidad del SLS completo a pesar de estar conformado por sistemas estables. De la misma forma, también existe la posibilidad de que un conjunto de subsistemas inestables con una regla de conmutación adecuada puedan producir un SLS estable.

Dentro de la investigación sobre el análisis de estabilidad de sistemas conmutados se destacan enfoques basados en la estabilidad según Lyapunov, adaptada al caso de SLS, lográndose líneas de análisis como el del teorema de Lyapunov inverso, funciones de Lyapunov cuadráticas conmutadas, y las funciones de Lyapunov cuadráticas comunes (CQLF por *common quadratic Lyapunov function*) (Lin and Antsaklis, 2009). De los anteriores, uno de los enfoques más ampliamente explorados ha sido el de la existencia de una CQLF, orientando los esfuerzos tanto a la determinación de condiciones de existencia/no-existencia de una CQLF (tales como Lin and Antsaklis, 2009), como

al cálculo de una CQLF dada su existencia (tales como Boyd et al., 1994; Liberzon and Tempo, 2003). En este ámbito, la determinación de condiciones de existencia/no-existencia y/o el diseño de un método de cálculo de una CQLF son conocidos como el *Problema CQLF*.

Desde el planteamiento Problema CQLF, el cálculo de una CQLF fue asumido como un subproblema a enfrentar usando desigualdades lineales de matrices (LMI por *linear matrix inequality*) y herramientas de resolución de tales desigualdades (Shorten and Narendra, 2003). A pesar de ello, y aunque existen herramientas bastante eficientes, sus algoritmos se ven fuertemente afectados por el orden y el número de matrices a utilizar (Liberzon and Tempo, 2004). Otras metodologías de cálculo desarrolladas posteriormente al uso de herramientas LMI buscaron mejorar sus limitaciones, y entre ellas la más sobresaliente es la basada en el método del gradiente (Liberzon and Tempo, 2004). Si bien en Liberzon and Tempo (2004) se desarrolla una metodología con un sólido respaldo analítico y se destacan todas las ventajas respecto de las herramientas LMI, el método propuesto deja entrever que es muy susceptible de ser mejorado respecto del tiempo de cálculo consumido, entre otros aspectos. Otras técnicas han sido desarrolladas para el cálculo de una CQLF, como por ejemplo Ibeas and de la Sen (2009), pero cabe aclarar que su área de aplicación está restringida a matrices triangulares.

Por su parte, los estudios para establecer las condiciones de existencia/no-existencia de una CQLF generalmente parten por analizar parejas de matrices, y una vez logrado esto se trata de extender los resultados al caso de N matrices. No obstante, cabe recalcar que solamente el caso de segundo orden tiene hasta la actualidad una solución analítica completa (Shorten and Narendra, 2002). Muchos otros enfoques se han usado para enfrentar el problema de determinar si una CQLF puede o no existir (por ejemplo Shorten et al. (2004, 2007); Lin and Antsaklis (2009); Laffey and Šmigoc (2009)), pero la mayoría de estos enfoque imponen fuertes restricciones sobre el número/orden de los sistemas a analizar, o el cumplimiento de determinadas propiedades especiales, por lo que queda claro que el caso general para N sistemas de orden n está lejos de tener solución completa en la actualidad.

2.2.1. Consideraciones matemáticas

Un SLS puede ser definido, como es mostrado en Shorten et al. (2007), de la siguiente forma

$$\dot{x} = A_{\zeta(t)}x(t), \quad A_{\zeta(t)} \in A := \{A_1, \dots, A_N\}, \quad (2.1)$$

donde $A_i \in \mathbb{R}^{n \times n}$, $i = 1, 2, \dots, N$ son matrices constantes y Hurwitz (valores propios en el semiplano complejo izquierdo abierto). El valor de la matriz $A_{\zeta(t)}$ depende del valor de la función $\zeta(t) : [0, \infty) \rightarrow \{1, 2, \dots, N\}$, llamada *señal de conmutación*, la cual es una función constante a tramos con un número finito de discontinuidades (*instantes de conmutación*) en todo intervalo acotado de tiempo. Por lo tanto, $A_{\zeta(t)}$ toma valores constantes del conjunto A en cada intervalo definido entre dos instantes de conmutación consecutivos (Shorten et al., 2007; Lin and Antsaklis, 2009).

Ya que el simple hecho de tratar con subsistemas estables no garantiza que el sistema conmutado global sea estable, puesto que la conmutación entre sistemas estables puede conducir a un sistema conmutado inestable para una determinada regla de conmutación (ver Figura 2.1), es necesario hacer un análisis de estabilidad para el sistema conmutado completo. Para el caso de los SLS, una

importante línea de análisis de estabilidad está relacionada con las funciones de Lyapunov comunes. Más particularmente, la existencia de una CQLF para un conjunto de sistemas dinámicos lineales que componen un sistema lineal conmutado, garantiza la estabilidad de dicho sistema conmutado (Liberzon, 2003; Lin and Antsaklis, 2009). Algunos ejemplos de aplicación del enfoque CQLF pueden ser encontrados en (Cheng and Zhang, 2006) para tiempo continuo, y (Benzaouia et al., 2010, 2011) para tiempo discreto.

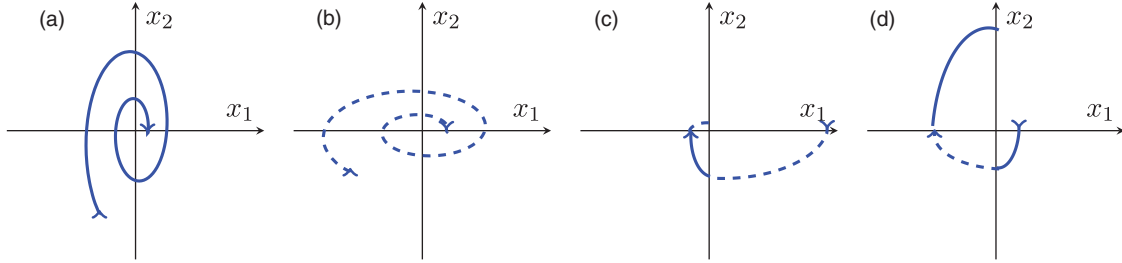


Figura 2.1.: Ejemplo de diagramas de fase para la conmutación de dos sistemas estables de segundo orden (Liberzon, 2003): (a) Primer sistema, (b) sgundo sistema, (c) conmutación estable, y (d) conmutación inestable.

2.2.2. El problema CQLF

Sea el SLS en tiempo continuo definido por (Ecuación 2.1), y sea $V(x)$ una función cuadrática de Lyapunov candidata de la forma

$$V(x) = x^T P x, P > 0, P \in \mathfrak{R}^{n \times n}, \quad (2.2)$$

cuya derivada con respecto al tiempo se requiere que sea negativa definida a lo largo de cualquier trayectoria no nula del sistema, es decir

$$\dot{V}(x) = x^T (P A_i + A_i^T P) x < 0, \forall i \in \{1, \dots, N\}, \quad (2.3)$$

o equivalentemente,

$$P A_i + A_i^T P = -Q_i < 0, \forall A_i \in A. \quad (2.4)$$

Entonces, si existe una matriz $P > 0$ satisfaciendo (Ecuación 2.2) y (Ecuación 2.4), la función $V(x)$ es una CQLF para todos los subsistemas de la forma

$$\Sigma_{A_i} : \dot{x}(t) = A_i x(t), i = 1, 2, \dots, N, \quad (2.5)$$

y su existencia garantiza la estabilidad uniforme y asintótica del SLS (Ecuación 2.1) bajo cualquier conmutación arbitraria $\varsigma(t)$ (Liberzon, 2003; Lin and Antsaklis, 2009). Algunos autores distinguen entre la existencia de P para $Q_i > 0$ ($Q \geq 0$), lo cual lleva al concepto de CQLF fuerte (débil). En adelante, en esta Tesis el término CQLF tendrá siempre el significado de CQLF fuerte. Además, y como abuso de notación, al referirse a una CQLF se hablará indistintamente de (Ecuación 2.2) o de la respectiva matriz P asociada a la forma cuadrática.

2.2.3. Metodologías de cálculo de una CQLF

Desde el punto de vista numérico, existen en la literatura diversas alternativas de cálculo de una CQLF, siendo la solución más tradicional las técnicas de resolución de LMI. El sistema LMI (Ecuación 2.4) se dice *factible* si se satisface para algún $P > 0 \forall i$, y *no factible* en caso contrario. A pesar de que existen métodos eficientes para resolver sistemas LMI desde hace un par de décadas (Boyd et al., 1994; Shorten and Narendra, 2003), el problema CQLF sigue siendo un problema abierto dado que tales métodos imponen condiciones fuertes para la búsqueda de soluciones, lo que produce que este método se torne ineficaz en la medida que aumenta la dimensionalidad del problema (orden y número de subsistemas). Sobra decir entonces que es impensable aplicar métodos de resolución de LMIs en el caso de familias infinitas de subsistemas, salvo en casos específicos como la combinación convexa de una familia finita, y es aquí donde las técnicas de malla adquieren importancia (Tempo et al., 2004). No obstante, los métodos de malla requieren un número de puntos de malla que crece exponencialmente con la dimensionalidad del problema, y no se asegura una solución pues ésta podría estar ubicada entre los puntos de la malla (siendo imposible acceder a ella).

Un ejemplo de método analítico de cálculo de una CQLF en $\mathbb{R}^{n \times n}$ es el clásico caso del que aquí se denominará Algoritmo NB (de **N**arendra-**B**alakrishnan), referido así por los resultados de estos autores presentados en Narendra and Balakrishnan (1994). Este método tiene la ventaja de garantizar una solución factible siempre y cuando se cumplan ciertas condiciones sobre los sistemas a analizar. Tales condiciones son bastante restrictivas ya que imponen que las matrices conmuten bajo la multiplicación, y una extensión al algoritmo NB impone que las matrices cumplan con cierta propiedad sobre el corchete de Lie (Zhu et al., 2007). La idea de este algoritmo está basada en el hecho de que N matrices, cumpliendo las condiciones anteriormente mencionadas, comparten una CQLF P_N , la cual puede ser calculada con la fórmula

$$P_i A_i + A_i^T P_i = -P_{i-1}, \quad i = 1, \dots, N, \quad (2.6)$$

todo esto sin importar el orden en que se tomen las matrices y cuál sea la matriz inicial $P_0 > 0$ escogida.

Otro de los métodos de cálculo analítico que puede usarse es el planteado en Cheng et al. (2006), basado en el cálculo de la bola más pequeña cubre un número finito de puntos $p_1, \dots, p_n \in \mathbb{R}^n$. En el caso del Problema CQLF, los puntos representan los sistemas a analizar, y la bola representa la CQLF a ser calculada. Aunque el método no fue desarrollado directamente para calcular una CQLF, uno de los ejemplos de aplicación permite esta posibilidad. Así, los autores proponen asimilar cada matriz como un punto en un espacio multidimensional, y entonces calculan la mínima bola que incluye a todos los puntos (matrices). Si el centro de esta bola mínima incluyente mapea a una

matriz inestable, entonces se concluye que el conjunto de matrices no comparte una CQLF. Por otra parte, si ese centro mapea a una matriz asintóticamente estable se calcula una función de Lyapunov \tilde{P} para ese centro, función que es candidata a CQLF. El proceso finaliza después de verificar una a una las ecuaciones de Lyapunov para confirmar que \tilde{P} es una CQLF. Al proceso de calcular una matriz \tilde{P} candidata a CQLF utilizando este método se le denominará en el presente trabajo el Algoritmo EB (por *Enclosing Balls*).

Sin embargo, entre las más importantes metodologías de cálculo se encuentra la de Liberzon and Tempo (2003, 2004), en la cual se plantea la minimización de funcionales apropiados usando el método de gradiente, que se basa en la tradicional fórmula

$$f(R + \Delta R) \approx f(R) + \langle \partial_R f, \Delta R \rangle \quad (2.7)$$

donde ΔR denota una pequeña perturbación con respecto a R , $\partial_R f$ denota el gradiente de f con respecto a R , y \approx denota igualdad hasta los términos de primer orden en ΔR . Ahora, dada una matriz simétrica P y una matriz arbitraria A se genera una función $v(P, A)$ de la forma

$$v(P, A) := f(PA + A^T P + Q) \quad (2.8)$$

y su derivada parcial $\partial_P v$, donde f es un funcional adecuado para el problema CQLF que se desea resolver y $Q > 0$ es una matriz arbitraria y dada. Liberzon and Tempo (2003) se proponen dos funcionales, se demuestra su continuidad y diferenciabilidad de una forma elegante, y se plantean las fórmulas para el cálculo de la CQLF en función de $v(P, A)$ y $\partial_P v$, que se resumen en el siguiente algoritmo

$$P_{k+1} = \begin{cases} [P_k - \mu_k \partial_P v(P_k, A_{h(k)})]^+, & \text{si } v(P_k, A_{h(k)}) > 0, \\ P_k, & \text{en otro caso.} \end{cases} \quad (2.9)$$

$$\mu_k := \frac{\alpha v(P_k, A_{h(k)}) + r \|\partial_P v(P_k, A_{h(k)})\|}{\|\partial_P v(P_k, A_{h(k)})\|^2}, \quad (2.10)$$

$$h(k) := (k \bmod N) + 1 = k - N \lfloor k/N \rfloor + 1, \quad (2.11)$$

donde el operador $[\cdot]^+$ denota la proyección al espacio de las matrices positivas semidefinidas, el operador $\lfloor \cdot \rfloor$ denota la aproximación al menor entero más próximo, μ_k es el paso de avance con $0 \leq \alpha \leq 2$, $r > 0$, y $\|\cdot\|$ denota la norma de Frobenius. Los resultados experimentales obtenidos son comparados con técnicas de resolución de LMI, adicionando el respectivo análisis de convergencia determinística y probabilística, convergencia que queda en función de la adecuada sintonización de los parámetros r y α que definen el paso de avance.

Otros métodos de cálculo de una CQLF han sido reportados en la literatura, pero muchos de

ellos son para casos muy restrictivos, como por ejemplo el propuesto por Cheng et al. (2000), con alta complejidad computacional debido al desarrollo de cálculos con matrices de dimensionalidad $N * n^2$, siendo N el número de las matrices de orden n a ser analizadas. También está el método propuesto por Nguyen et al. (2003) y el de Yanami and Anai (2005), con complejidad algorítmica y computacional que restringe la aplicación de la metodología solamente a sistemas de segundo orden. Finalmente es interesante mencionar el método de Paul et al. (2004), que se enfoca en cierta clase de sistemas de segundo orden, el de Chen and Gao (2007), cuya área de aplicación está limitada a matrices triangulares, y el de Ibeas and de la Sen (2009), que se aplica a un conjunto de matrices que son simultáneamente triangularizables.

2.2.4. Determinación de la existencia/no-existencia de una CQLF

Diversos estudios se han realizado en relación a la determinación de condiciones de existencia/no-existencia de una CQLF (Lin and Antsaklis, 2009). Sin embargo, esta parte del Problema CQLF sigue siendo un problema abierto sin solución completa al día de hoy. En Narendra and Balakrishnan (1994) se estudia el caso general del corchete de Lie igual a cero, y se demuestra que la conmutatividad en el producto de dos matrices Hurwitz es una condición suficiente para que éstas compartan una CQLF. El análisis ahí presentado ha dado lugar al Algoritmo NB para el cálculo de tal CQLF, el cual puede ser aplicado con éxito a (Ecuación 2.1) donde las N matrices conmutan por parejas. Más tarde, en Zhu et al. (2007), se extiende el alcance de aplicación del Algoritmo NB al caso en el que el corchete de Lie es representado por una combinación lineal de sus argumentos, con ciertas condiciones impuestas sobre los coeficientes de la combinación.

En general, el establecimiento de condiciones para la existencia/no-existencia de una CQLF usualmente comienza con el análisis de parejas de matrices, y luego se trata de extender los resultados obtenidos al caso de N matrices. Hace aproximadamente una década atrás, una prometedora conjetura relacionada con la existencia de una CQLF para N sistemas positivos de orden arbitrario fue presentada en Mason and Shorten (2003). Desafortunadamente, un trabajo reciente (Gurvits et al., 2007) muestra que la conjetura no es válida para el caso general: fue verificado que sólo es válida para el caso de N sistemas positivos de segundo orden. Los SLS formados por matrices triangulares o simultáneamente triangularizables (Mori et al., 1997; Shorten and Narendra, 1998; Ibeas and de la Sen, 2009) son ejemplos de condiciones suficientes para la existencia de una CQLF, las cuales no son restrictivas respecto del orden o el número de matrices a analizar, pero sí son muy restrictivas respecto de la estructura de los sistemas, o de la existencia de una matriz de transformación común, respectivamente.

Uno de los resultados más importantes en relación al problema CQLF es la solución general para N sistemas de segundo orden (Shorten and Narendra, 2002) (excepto por la restricción $a_{21i} \neq 0 \forall i \in 1, \dots, N$). En (Shorten and Narendra, 2002) se presentan los principales resultados al respecto. En primera instancia, se presenta una solución elegante para el caso de parejas de matrices; este primer resultado está basado en el análisis de estabilidad de combinaciones lineales convexas (CLC por sus siglas en inglés *convex linear combination*) de dos sistemas. Como segunda instancia se aborda el caso de N sistemas de segundo orden usando el Teorema de Helly. A pesar de que ambos análisis están basados en diferentes enfoques, se considera que el caso de segundo orden tiene una solución analítica completa en lo que al Problema CQLF se refiere.

Existen otros tipos de análisis que pueden ser aplicados a parejas de matrices (véase por ejemplo, Shorten and Narendra (2003); King and Nathanson (2006); Laffey and Šmigoc (2009)). En King and Nathanson (2006) y Laffey and Šmigoc (2009) se estudia el caso de parejas de matrices cuya diferencia es de rango 1, caso al cual también pertenecen las parejas de matrices que están en la forma canónica companion (Shorten and Narendra, 2003); para parejas de este tipo de matrices se deduce una simple condición algebraica que asegura la existencia de una CQLF por parejas. Existen otros casos más restrictivos, como por ejemplo cuando el orden y el número de las matrices es fijo, como lo es el trabajo de King and Shorten (2006) en donde se obtiene una condición necesaria y suficiente para la no existencia de una CQLF para parejas de sistemas de tercer orden por medio del análisis de CLCs de las matrices de evolución y sus inversas. Muchos otros enfoques han sido propuestos para solucionar el Problema CQLF (Shorten et al., 2003, 2004, 2007; Moldovan and Seetharama, 2009), pero generalmente se observa que la mayoría de esos enfoques imponen restricciones sobre el orden de los sistemas, el número de sistemas a analizar, o alguna otra propiedad especial (rango de la diferencia, conmutatividad, etc.).

En el escenario de soluciones numéricas, una solución interesante para el caso general de N sistemas de orden n es presentado en (Cheng et al., 2003). Allí se propone un método para determinar la existencia de una CQLF basado en una condición necesaria y suficiente relacionada con la positividad de una integral dada. Aunque el método se plantea para el caso general, solamente el caso de sistemas de segundo orden es abordado y demostrado en detalle. Métodos basados en optimización numérica han sido diseñados para determinar la existencia de una CQLF por medio del cálculo de la misma (véase Subsección 2.2.3). Entre los más relevantes están: un método para calcular una CQLF basado en resolución de sistemas LMI (Boyd et al., 1994) usando, por ejemplo, un toolbox de Matlab (Enfoque LMI); un método para calcular una CQLF basado en el método de gradiente (Liberzon and Tempo, 2003, 2004) (Enfoque L-T); y un método para calcular una CQLF basado en inteligencia de enjambre (Ordóñez-Hurtado and Duarte-Mermoud, 2012) (Enfoque O-D) el cual es el resultado del desarrollo de la presente Tesis Doctoral.

A pesar de que la existencia de una CQLF es una garantía de estabilidad asintótica para un SLS dado, también es importante determinar cuándo una CQLF no puede existir. En este sentido las metodologías de cálculo fallan, porque la obtención de una solución final no factible no implica que una solución factible no exista, ya que el éxito del proceso de búsqueda depende de una buena sintonización de los parámetros de configuración. Sin embargo, existen algunos métodos orientados a la determinación numérica de la no existencia de una CQLF, tales como el trabajo de Davis and Eisenbarth (2011) basado en programación lineal para la solución simultánea de sistemas polinomiales de inecuaciones en sistemas de segundo orden, y el trabajo de Ordóñez-Hurtado and Duarte-Mermoud (2011b) basado en inteligencia de enjambre para analizar combinaciones convexas de sistemas, siendo este último enfoque otro de los resultados del desarrollo de la presente Tesis Doctoral. Cabe decir que este último tipo de metodologías son complementarias a los métodos de cálculo ya que pueden ofrecer información concluyente en casos en los cuales los métodos de cálculo no logran su objetivo.

2.2.5. Algunos resultados previos

En la literatura de control, el tema de la determinación de condiciones para la existencia/no-existencia de una CQLF ha sido ampliamente estudiado, y existe actualmente una solución com-

pleta para el caso de dos matrices $\{A_1, A_2\}$ de segundo orden (Shorten and Narendra, 2002). La principal herramienta para lograr tal resultado es el análisis de estabilidad de CLCs de $A_1^{\pm 1}$ y A_2 , de la forma

$$\begin{aligned}\sigma_\alpha [A_1, A_2] &:= \text{CLC} [A_1, A_2] \\ &:= \alpha A_1 + (1 - \alpha) A_2, \quad \forall \alpha \in [0, 1],\end{aligned}$$

$$\begin{aligned}\sigma_\alpha [A_1^{-1}, A_2] &:= \text{CLC} [A_1^{-1}, A_2] \\ &:= \alpha A_1^{-1} + (1 - \alpha) A_2, \quad \forall \alpha \in [0, 1],\end{aligned}$$

donde $\sigma_\alpha [A_i, A_j]$ denota la matriz *pencil* de A_i y A_j (Shorten and Narendra, 2002), el cual es Hurwitz si todos sus valores propios están en el semiplano complejo izquierdo abierto para todo $\alpha \in [0, 1]$ (Shorten et al., 2003). Para la conveniencia del lector, a continuación se listan los principales teoremas, lemas y proposiciones relacionados con el Problema CQLF, los cuales serán usados en el desarrollo de las metodologías propuestas en esta Tesis Doctoral.

Lema 2.2.1. (Shorten and Narendra, 2002). Sean los sistemas $\Sigma_A : \dot{x} = Ax$ y $\Sigma_{A^{-1}} : \dot{x} = A^{-1}x$, donde $A \in \mathfrak{R}^{n \times n}$ es Hurwitz. Entonces, cualquier función cuadrática de Lyapunov para A también lo es para A^{-1} .

Teorema 2.2.2. (Shorten and Narendra, 2002). Sea el SLS (Ecuación 2.1) con $x \in \mathfrak{R}^2$ y $N = 2$. Entonces los siguientes enunciados son equivalentes:

1. Existe una CQLF para $A = \{A_1, A_2\}$.
2. Los pencil $\sigma_\alpha [A_1, A_2]$ y $\sigma_\alpha [A_1, A_2^{-1}]$ son Hurwitz.
3. Los productos $A_1 A_2$ y $A_1 A_2^{-1}$ no tienen valores propios reales negativos.

Proposición 2.2.3. (Liberzon, 2003). Los sistemas lineales $\dot{x} = A_1 x$ y $\dot{x} = A_2 x$, con $A_1, A_2 \in \mathfrak{R}^{2 \times 2}$, comparten una CQLF si y sólo si todas las CLC por parejas en $\{A_1, A_2, A_1^{-1}, A_2^{-1}\}$ son Hurwitz.

Lema 2.2.4. (Shorten et al., 2004). Si los sistemas LTI estables $\dot{x} = A_1 x$ y $\dot{x} = A_2 x$, con $A_1, A_2 \in \mathfrak{R}^{n \times n}$, comparten una CQLF, entonces los pencil $\sigma_\alpha [A_1, A_2]$ y $\sigma_\alpha [A_1, A_2^{-1}]$ son no singulares. Equivalentemente, los productos $A_1 A_2$ y $A_1^{-1} A_2$ no tienen valores propios reales negativos.

Lema 2.2.5. (Horn and Johnson, 1985). Sea $C, D \in \mathfrak{R}^{n \times n}$, con $C, D > 0$ y $\alpha \in \mathfrak{R}^+$. Entonces (i) $C + D > 0$, y (ii) $\alpha C > 0$.

Lema 2.2.6. (Horn and Johnson, 1985). Sea A una matriz arbitraria en $\mathfrak{R}^{n \times n}$. Entonces se sigue que (i) $A > 0 \Leftrightarrow -A < 0$, y (ii) $A > 0 \Leftrightarrow (A + A^T) > 0$.

2.3. Sistemas adaptables

Al hablar de sistemas adaptables se hace referencia a aquellos sistemas capaces de hacer ajustes a su estructura y/o parámetros a fin de satisfacer condiciones especificadas en la etapa de diseño. Desde este punto de vista, un sistema adaptable puede ser definido en general como un sistema no-lineal variante en el tiempo, descrito por una ecuación diferencial genérica

$$\dot{x}(t) = f(x(t), u(t), t)$$

en tiempo continuo, o la ecuación en diferencias genérica

$$x(t+1) = f(x(t), u(t), t)$$

en tiempo discreto. Un importante ejemplo de este tipo de sistemas es el ampliamente conocido Control Adaptable por Modelo de Referencia (MRAC por *Model Reference Adaptive Control*), el cual está orientado al ajuste en línea de los parámetros de un controlador y/o de un modelo de identificación (estimador), de tal forma que el sistema planta-controlador-estimador se comporte como un modelo de referencia previamente diseñado según especificaciones de desempeño deseadas.

Aunque puede darse el caso de tener que adaptar en línea la estructura de un modelo, en general es más común encontrarse con el problema del ajuste en línea de los parámetros de un modelo/controlador. Para ello, se han desarrollado diversas técnicas, encontrando entre las más conocidas el método de mínimos cuadrados, y el método de gradiente en su versión genérica (regla del MIT) o derivado de un riguroso análisis de estabilidad según el enfoque de Lyapunov (Ioannou and Fidan, 2006). No obstante, en la última época, y gracias al auge de las técnicas heurísticas de optimización, se ha incursionado en la aplicación de técnicas basadas en Computación Evolucionaria al área de la estimación en línea tales como los Algoritmos Genéticos y Optimización por Enjambre de Partículas.

Una importante línea de análisis dentro de los sistemas adaptables son los denominados modelos de error, existiendo en la actualidad cuatro modelos de error en tiempo continuo (Narendra and Annaswamy, 1989; Duarte-Mermoud and Narendra, 1996), de los cuales tres tienen su paralelo en tiempo discreto (Duarte-Mermoud and Narendra, 1996; Tao, 2003). El empleo de este tipo de modelos brindan la gran ventaja de que, por un lado, se pueden agrupar familias enteras de sistemas adaptables en cada modelo de error y, por otro lado, se elimina el problema de soluciones particulares al obtenerse soluciones genéricas. Sin embargo, a pesar de la confiabilidad de las soluciones existentes (en general basadas en el método directo de Lyapunov y el método del gradiente), estas soluciones presentan deficiencias en otros aspectos no referidos a la estabilidad. Tales deficiencias están directamente relacionadas con el manejo de las respuestas transitorias en los procesos de adaptación, teniendo que incluso para el caso de sistemas de orden bajo con pocos parámetros se presentan transitorios altamente oscilatorios de gran amplitud y frecuencia, a causa de usar ganancias de adaptación altas que busquen tiempos de convergencia más cortos. Para estos casos, en pro de la consecución de transitorios más estables y cortos que conlleven entre otras cosas a un alargamiento de la vida útil de los actuadores, generalmente se recurre al uso de ganancias de adaptación muy pequeñas, pero éstas a su vez hacen que el problema de adaptación se resuelva muy lentamente.

2.3.1. Sistemas adaptables discretos basados en modelos de error

Los sistemas adaptables son concebidos, en general, a partir del diseño de identificadores y controladores adaptables para plantas con parámetros desconocidos (Narendra and Annaswamy, 1989). El análisis de este problema abarca desde plantas lineales de primer orden, hasta plantas descritas por ecuaciones diferenciales con variables de estado no medibles (no accesibles). De cualquier forma, el interés se centra en el problema de identificación a partir de la estimación de parámetros usando datos de entrada-salida, incluyendo también el problema de control, donde los parámetros del controlador son ajustados directamente para lograr que el error entre la salida de la planta y la salida de un modelo de referencia tienda a cero asintóticamente. Entonces, es necesaria la definición de leyes de ajuste basadas en algún método que permita asegurar la estabilidad del sistema completo, siendo el método directo de Lyapunov es la principal herramienta usada para la derivación de leyes de ajuste estables (Narendra and Annaswamy, 1989).

En general, en todos los casos de identificación y control adaptable de sistemas dinámicos de parámetros desconocidos, los parámetros del estimador o del controlador adaptable son ajustados de tal manera que el error de identificación (diferencia entre la salida de la planta y la salida del modelo de identificación) o el error de control (diferencia entre la salida de la planta y la salida del modelo de referencia) tienda a cero asintóticamente. Desde el punto de vista del desempeño y la estabilidad, es importante lidiar con el comportamiento de los errores de parámetros y de salida como funciones del tiempo, teniendo así que los modelos matemáticos que describen la evolución de estos errores son llamados modelos del error (Narendra and Annaswamy, 1989, Capítulo 7). Entonces, resulta que las diferentes formulaciones y las correspondientes elecciones de leyes de adaptación conllevan al uso de diferentes modelos del error. Analizando estos modelos de error es posible obtener un mejor entendimiento de una gran clase de sistemas adaptables.

La descripción general de un modelo discreto de error, análoga al caso continuo (Narendra and Annaswamy, 1989), puede ser entendida como la evolución del vector de error de salida $e(t)$ por medio de la ecuación en diferencias $\Delta e(t) = f(e, \phi, t)$ (ecuación del error), donde $\phi(t)$ es el error paramétrico en el instante t , ecuación que es obtenida directamente de las ecuaciones que describen a la planta. Entonces, la idea de la adaptación es lograr determinar una función $g(\cdot, t)$ para que la ecuación en diferencias $\Delta \phi(t) = g(e, t)$ (ley de ajuste) junto con la ecuación del error tengan $e(t) = 0$, $\phi(t) = 0$ como estado de equilibrio, lo cual asegura las propiedades de estabilidad deseadas. Ya que el error paramétrico ϕ es desconocido en cualquier problema de adaptación, la función g no puede depender explícitamente de $\phi(t)$. En términos del modelo del error, un problema de identificación o control adaptable puede ser establecido como uno de determinación de un conjunto de m ecuaciones en diferencias (la ley de adaptación) dado un conjunto de n ecuaciones en diferencias (la ecuación del error) para que la combinación de las $m + n$ ecuaciones tenga las propiedades de estabilidad deseadas. De esta forma, los modelos del error capturan las características esenciales de los procesos adaptivos.

2.3.1.1. Modelo de Error 1

El llamado Modelo del Error 1 discreto (Goodwin et al., 1980; Duarte-Mermoud and Narendra, 1996) emplea la parametrización del tipo regresión lineal, frecuentemente usada en identificación y control (Tao, 2003). Este puede ser obtenido al considerar una planta descrita por $y(t) = \theta^{*T} \omega(t)$,

donde $\omega(t) \in \mathfrak{R}^n$, $y(t) \in \mathfrak{R}$ pueden ser medidos para todo $t \geq t_0$. Por su parte, θ^* es un vector desconocido constante en \mathfrak{R}^n . Para generar la señal $\hat{y}(t) = \hat{\theta}^T(t) \omega(t)$ se utiliza el estimado $\hat{\theta}(t)$ de θ^* , y se define $\phi(t) = \hat{\theta}(t) - \theta^* \in \mathfrak{R}^n$ como el error paramétrico y $e(t) = \hat{y}(t) - y(t) \in \mathfrak{R}$ como el error de salida, para obtener una ecuación de error del tipo

$$e(t) = \phi^T(t) \omega(t). \quad (2.12)$$

La Ecuación 2.12 (en conjunto con su ley de ajuste) es llamada Modelo de Error 1, y de ella se deduce que $e(t)$ y $\omega(t)$ contienen información acerca del signo del error paramétrico $\phi(t)$.

Dos soluciones al problema del diseño de leyes adaptivas para los sistemas representados por este modelo de error son presentadas en Tao (2003). La primera solución usa el método directo de Lyapunov para derivar las leyes adaptivas basadas en Gradiente Normalizado (NG por sus siglas en inglés *Normalized Gradient*), las cuales toman la forma

$$\hat{\theta}(t+1) = \hat{\theta}(t) - \frac{\Gamma e(t) \omega(t)}{m^2(t)}, \quad \hat{\theta}(t_0) = \hat{\theta}_0, \quad \Gamma = \Gamma^T > 0, \quad (2.13)$$

$$m^2(t) = \kappa + \omega^T(t) \omega(t), \quad \kappa > 0, \quad (2.14)$$

usando comúnmente $\kappa = 1$, y $\Gamma = \text{diag}\{\gamma_1, \dots, \gamma_n\} \in \mathfrak{R}^{n \times n}$ es la matriz diagonal de ganancias de adaptación. La segunda solución usa el método de los mínimos cuadrados normalizados (NLS de sus siglas en inglés *Normalized Least-Squares*) para ajustar los parámetros de la forma

$$\hat{\theta}(t+1) = \hat{\theta}(t) - \frac{\alpha_{LS} \Gamma(t-1) e(t) \omega(t)}{m^2(t)}, \quad \hat{\theta}(t_0) = \hat{\theta}_0, \quad (2.15)$$

$$\Gamma(t) = \Gamma(t-1) - \frac{\alpha_{LS} \Gamma(t-1) \omega(t) \omega^T(t) \Gamma(t-1)}{m^2(t)}, \quad (2.16)$$

$$\Gamma(t_0 - 1) = \Gamma_0 = \Gamma_0^T > 0,$$

con $m^2(t)$ dada por (Ecuación 2.14). Aquí también se usa comúnmente $\kappa = 1$, y $\Gamma(t)$ es una matriz que actúa como una ganancia de adaptación variante en el tiempo. Las propiedades de estos algoritmos pueden encontrarse en Duarte-Mermoud and Narendra (1996); Tao (2003).

2.3.1.2. Modelo de Error 2

Sea la ecuación de error definida en Duarte-Mermoud and Narendra (1996) (con $k = 1, b = 1$) como:

$$e(t+1) = Ae(t) + \phi^T(t+1) \omega(t), \quad \phi^T(t), \omega(t), e(t), y(t) \in \mathfrak{R}^n, \quad A \in \mathfrak{R}^{n \times n}, \quad (2.17)$$

con A asintóticamente estable y $e(t)$ medible. Entonces se puede definir una nueva señal de error $\epsilon(t)$ de la forma

$$\epsilon(t+1) = e(t+1) - Ae(t), \quad (2.18)$$

de modo que la ecuación del error queda descrita ahora por

$$\epsilon(t+1) = \phi^T(t+1)\omega(t), \quad (2.19)$$

con lo cual se obtiene un modelo de error equivalente al Modelo de Error 1. Con ello se pueden usar, por ejemplo, las leyes de adaptación (Ecuación 2.15) (NG) de la forma

$$\hat{\theta}(t+1) = \hat{\theta}(t) - \frac{\alpha\Gamma\epsilon(t)\omega(t-1)}{m^2(t)}, \quad \hat{\theta}(t_0) = \hat{\theta}_0, \quad \Gamma = \Gamma^T > 0, \quad (2.20)$$

$$\Rightarrow \hat{\theta}(t+1) = \hat{\theta}(t) - \frac{\alpha\Gamma[e(t) - Ae(t-1)]\omega(t-1)}{m^2(t)}, \quad \hat{\theta}(t_0) = \hat{\theta}_0, \quad (2.21)$$

con $0 < \alpha < 2$ y $m^2(t)$ dado por (Ecuación 2.14). Un proceso similar se aplica para el uso de las leyes de ajuste NLS. Las propiedades del modelo definido por (Ecuación 3.1) o (Ecuación 3.3) y (Ecuación 2.20) o (Ecuación 2.21) se pueden encontrar en Duarte-Mermoud and Narendra (1996), el cual se conoce como Modelo de Error 2 discreto.

2.3.1.3. Modelo de Error 3

Sea el caso en que el error está definido por (Duarte-Mermoud and Narendra, 1996):

$$e(t+1) = Ae(t) + bv(t), \quad (2.22)$$

$$e_1(t) = c^T e(t) + dv(t), \quad (2.23)$$

$$v(t) = \phi^T(t)u(t) - \alpha\omega^T(t)\Gamma\omega(t)e_1(t), \quad (2.24)$$

$$\alpha > \frac{1}{2} \quad \Gamma = \Gamma^T > 0,$$

con $W(z) = d + c^T(zI - A)^{-1}b$ una función de transferencia estrictamente real positiva (SPR por *strict positive real*), de modo que $e_1(t)$ se pueda escribir como

$$e_1(t) = W(z)[v(t)].$$

Entonces, la leyes de ajuste para NG quedan dadas por:

$$\hat{\theta}(t+1) = \hat{\theta}(t) - \frac{\alpha \Gamma e_1(t) \omega(t)}{m^2(t)}, \quad \hat{\theta}(t_0) = \hat{\theta}_0, \quad \Gamma = \Gamma^T > 0, \quad (2.25)$$

con $0 < \alpha < 2$ y $m^2(t)$ dado por (Ecuación 2.14). Las propiedades del modelo definido por (Ecuación 2.22), (Ecuación 2.23), (Ecuación 2.24) y (Ecuación 2.25), se pueden encontrar en Duarte-Mermoud and Narendra (1996), el cual se conoce como Modelo de Error 3 discreto.

2.4. Optimización global

Uno de los principios fundamentales en el mundo en que vivimos es la búsqueda de un estado óptimo. Esto se refleja desde el microcosmos, con los átomos tratando de formar cadenas a fin de minimizar la energía entre sus electrones, hasta las grandes organizaciones sociales, en las cuales los procesos de evolución de las especies buscan la adaptación de los individuos a su entorno como estado óptimo. Desde que la humanidad existe siempre ha buscado la perfección en diversas áreas, queriendo obtener el máximo grado de satisfacción con la menor cantidad de esfuerzo. Un ejemplo claro de esto es la economía mundial, en donde las utilidades y las ventas deben ser maximizadas y los costos deben llevarse a su más bajo valor posible. En este orden de ideas, puede comprenderse que la optimización es una de las más antiguas ciencias que se extienden hasta la actualidad. Si bien es claro que al hablar de optimización se habla de una disciplina matemática, es necesario entonces plantear los problemas de optimización en este lenguaje, partiendo desde la misma meta que se busca. Una vez hecho esto, se escoge la técnica de optimización global idónea para solucionar el problema, de entre una amplia lista que actualmente existen.

Una taxonomía aproximada de los métodos de optimización global (Weise, 2008), presentada en la Figura 2.2, muestra una clasificación respecto del modo de operación: determinísticos y probabilísticos. Los algoritmos determinísticos tienen la ventaja de correlacionar las características de las posibles soluciones con su utilidad, y entonces explorar eficientemente el espacio de búsqueda. Sin embargo, si la relación entre una solución candidata y su bondad es bastante compleja, o el espacio de búsqueda tiene una dimensionalidad alta, estos algoritmos resultan ser ineficientes incluso en problemas pequeños. Es aquí donde los algoritmos probabilísticos revelan sus ventajas. Una de las principales virtudes de los algoritmos probabilísticos es que pueden garantizar soluciones muy cercanas a la óptima en un tiempo de ejecución relativamente corto, que es mucho mejor que obtener una solución óptima con un algoritmo determinístico que pueda necesitar, por ejemplo, 10^{100} años en ser alcanzada. Por otra parte, los algoritmos probabilísticos se apoyan en la heurística y meta heurística para ayudar a decidir cuáles posibles soluciones serán las próximas a evaluar, siendo esta una forma muy eficiente y útil de hacerlo.

La rama más importante de los algoritmos probabilísticos se basa en el método Monte Carlo (Owen, 1998), y entre ellos una clase sobresaliente es la Computación Evolucionaria (EC por *Evolutionary Computation*), que abarca todas las técnicas inspiradas en la naturaleza y en la evolución de un conjunto de múltiples soluciones candidatas llamado población.

A pesar de que la Figura 2.2 muestra la clasificación respecto del modo de operación, una clasificación más comúnmente usada está orientada a las propiedades de las técnicas de optimización,

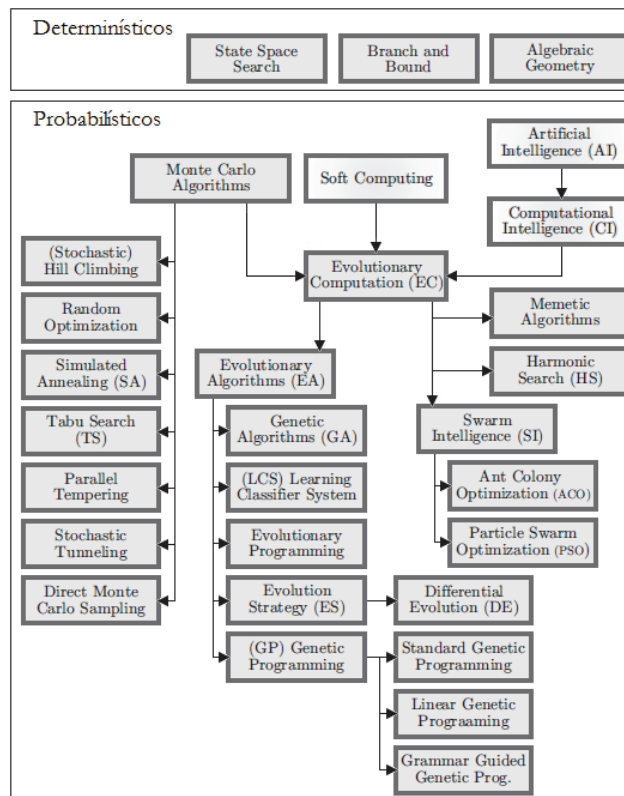


Figura 2.2.: Taxonomía de los métodos de optimización global. (Se conserva la denominación original en inglés por uniformidad con la literatura científica.) (Pérez-López, 2005)

tales como la velocidad de optimización y el número de criterios. Las categorías respecto de la velocidad de optimización suelen ser:

1. Optimización en línea, relacionada con problemas de optimización que requieren soluciones en tiempo real (localización de robots, control adaptable, control predictivo, entre otros), y
2. Optimización fuera de línea, en donde el tiempo de optimización no es tan importante y los resultados son susceptibles de ser esperados incluso días o semanas, aún para soluciones óptimas y sub-óptimas (minería de datos, identificación estructural de sistemas dinámicos, planificación de esquemas de transporte, entre otros).

Independientemente de la técnica de optimización usada, la optimización global ha visto su aplicación en áreas tan diversas como (Weise, 2008): química, bioquímica e ingeniería bioquímica, problemas de satisfacción de restricciones (CPS), toma de decisiones multi-criterio (MCDM), biología, ingeniería eléctrica, control de sistemas dinámicos, optimización estructural, economía y finanzas, estimación de parámetros, problemas matemáticos, redes y comunicación, y óptica, entre muchas otras. Al mirar el panorama de aplicación de las diversas técnicas en las áreas mencionadas, se observa una tendencia amplia y creciente de utilización de técnicas probabilísticas, especialmente las derivadas de la EC, como una alternativa eficiente respecto de las técnicas determinísticas.

2.4.1. Técnicas basadas en Computación Evolucionaria

Comúnmente este tipo de técnicas se basan en la evolución iterativa de unas soluciones candidatas iniciales, por medio de la aplicación de algoritmos perturbados por señales aleatorias y determinísticas. Dependiendo de la técnica, cada algoritmo tiene sus propios operadores, esquemas y estrategias. Dada la diversidad de técnicas dentro de la computación evolucionaria, se hará énfasis en las familias más importantes, a saber: (i) Algoritmos Evolutivos (EA por *Evolutionary Algorithm*), y (ii) Inteligencia de Enjambre (SI por *Swarm Intelligence*).

La primera y más extensa familia dentro de la EC es la de los EA, con sus representantes más destacados: los Algoritmos Genéticos (GA por *Genetic Algorithm*), la Programación Genética (GP por *Genetic Programming*), y la Evolución Diferencial (DE por *Differential Evolution*) (Storn and Price, 1995) como parte de las Estrategias Evolutivas (ES por *Evolution Strategy*). En base a los principios de selección y evolución natural, esta familia utiliza un esquema clásico de evolución basado en la presión ejercida por operadores de selección, cruce, mutación, y supervivencia. Sin embargo, este esquema de evolución ha sido objeto de mejoras y variaciones en pro del incremento del rendimiento de los métodos. La ventaja de los EA frente a otros métodos de optimización es que su característica de “caja negra” permite manejar pocos supuestos acerca de las funciones objetivo subyacentes, funciones que usualmente requieren menos profundización en la estructura del problema que en el caso de una función objetivo para un método determinístico. Mientras que para el caso básico de los EA se realiza evolución de individuos representados por cadenas binarias, DE se encarga de evolucionar vectores de números reales, y GP evoluciona programas computacionales representados por árboles. Cabe aclarar que si bien existen versiones de EA que realizan codificación real en vez de binaria, existen otras técnicas que evolucionan de mejor forma las cadenas de números reales, como es el caso de DE u otros de la familia SI.

Como segundo punto se tiene la familia SI, constituida por Optimización por Colonia de Hormigas (ACO por *Ant Colony Optimization*) y Optimización por Enjambre de Partículas (PSO por *Particle Swarm Optimization*). Esta familia utiliza la simulación del comportamiento de colectividades sociales (enjambres, bancos, bandadas, colonias, etc.) como base para la evolución de las poblaciones de soluciones candidatas (individuos). Otro aspecto importante es que se dejan de lado conceptos como selección y reproducción para incorporar otros como posición, velocidad, visibilidad, comunicación y memoria, entre otros. La técnica ACO se deriva de investigaciones sobre comportamiento de hormigas reales y las correspondientes simulaciones experimentales. Su principal aplicación está orientada a áreas en donde se buscan rutas óptimas en gráficos, lo que corresponde a la metáfora de las hormigas buscando comida en un área específica. A pesar de ello, hay estudios que muestran que la optimización vectorial de números reales se puede interpretar como la optimización de rutas en un gráfico, esto con el fin de ampliar el área de aplicación de ACO. La segunda técnica dentro de IS es PSO, en la cual se simula el comportamiento de un enjambre de partículas en un espacio multidimensional. Ya que cada partícula tiene un grado de libertad en su movimiento por tal espacio, aparecen conceptos como posición y velocidad. El vector de velocidades determina la dirección de la siguiente búsqueda, como también el tipo de búsqueda: de exploración (velocidad alta), de explotación (velocidad baja), o combinación entre exploración-explotación al incorporar elementos de Recocido Simulado (SA por *Simulated Annealing*).

En contraste con la familia EA, la familia IS tiene como elementos básicos componentes como memoria individual y global, y comunicación. En base a esto se ha hecho inclusión de elementos de

memoria global en los EA, apareciendo conceptos como elitismo (permanencia del mejor individuo de una generación a otra). Estos hechos hacen que técnicas como PSO muestren ser más eficientes, debido esencialmente al hecho de que los individuos mantienen diversidad y cooperación en las soluciones al memorizar las mejores posiciones tanto individuales como globales, y se colaboran entre ellos compartiendo información. Además es importante resaltar que, al igual que DE, PSO tiene un amplio rango de aplicación debido a que muchos de los problemas de optimización implican espacios de búsqueda en \mathcal{R}^n .

2.5. Particle Swarm Optimization

La técnica PSO (Eberhart and Kennedy, 1995a,b; Del Valle et al., 2008) es una técnica de optimización global heurística que pertenece a la categoría SI, que a su vez es una subcategoría de la EC. Esta técnica permite resolver problemas de optimización mediante el uso de cúmulos (enjambres) de partículas, los cuales simulan computacionalmente el comportamiento de grupos sociales en la naturaleza (bandadas, bancos, muchedumbres, etc.) en su proceso de búsqueda de algún beneficio común.

PSO pertenece a la clase de técnicas heurísticas modernas basadas en *poblaciones*, donde cada potencial solución es llamada una *partícula*. Estas partículas evolucionan iterativamente según estrategias, operadores y elementos relacionados con el movimiento en un espacio λ -dimensional, siendo λ es el número de incógnitas en la función a optimizar, con el objetivo de encontrar la mejor solución global posible (o al menos la mejor aproximación a ella). A diferencia de otras técnicas de EC, tal como los GA, PSO no implementa en su algoritmo básico estrategias de combinación o mutación, y siempre mantiene lo mejor de su experiencia evolutiva (local y global) (Del Valle et al., 2008). Las partículas entonces tienen la posibilidad de “volar” (moverse) en el espacio de búsqueda multidimensional sin olvidar la mejor posición encontrada, y a su vez son influenciadas por la partícula que ha encontrado la mejor posición global.

Como la idea de PSO es simular el movimiento de partículas en un espacio multidimensional, se requiere entonces de un conjunto de fórmulas para actualizar la velocidad y la posición de cada partícula dentro del espacio de búsqueda. En el algoritmo básico de PSO (Eberhart and Kennedy, 1995b), las partículas evolucionan por medio de las ecuaciones

$$v_{i,d}(k+1) = v_{i,d}(k) + r_1(k) c_1 (p_{i,d}(k) - x_{i,d}(k)) + r_2(k) c_2 (g_d(k) - x_{i,d}(k)) \quad (2.26)$$

$$x_{i,d}(k+1) = x_{i,d}(k) + v_{i,d}(k+1) \quad (2.27)$$

donde $v_{i,d}(k)$ y $x_{i,d}(k)$ representan respectivamente la velocidad y la posición del componente $d \in \{1, 2, \dots, \lambda\}$ de la partícula $i \in \{1, 2, \dots, s\}$ en la iteración $k \in \{1, \dots, iter_{\max}\}$. Las constantes c_1 y c_2 son los coeficientes de aceleración cognitiva y social, y determinan la influencia de la experiencia individual y colectiva sobre el desempeño de cada partícula. Los términos $r_1(k)$ y $r_2(k)$ son un par de números aleatorios distribuidos uniformemente en el intervalo $[0, 1]$ (es decir,

$r_1, r_2 \sim U[0, 1]$), empleados para emular el componente estocástico de cualquier enjambre social. La variable $p_{i,d}(k)$ es la d -ésima componente de la mejor posición de la partícula i , y la variable $g_d(k)$ es la d -ésima componente de la mejor posición global, ambas en la iteración k .

El algoritmo básico de PSO puede ser resumido en los siguientes pasos, en los cuales la función de fitness a ser optimizada es denotada por f :

Algoritmo 2.5.1. *Algoritmo PSO Básico*

1. Inicialización: Defina el vector de posiciones \mathbf{x}_i y el vector de velocidades \mathbf{v}_i aleatoriamente con distribución uniforme. Defina el vector de mejores posiciones individuales \mathbf{p}_i y la mejor posición global \mathbf{g} como $\{\mathbf{p}_i = \mathbf{x}_i\}_{i=1}^s$ y

$$\mathbf{g} = \operatorname{argmin} \{f(\mathbf{p}_i)\}_{i=1}^s. \quad (2.28)$$

2. Búsqueda: defina $k = k + 1$, y
 - a) escoja $r_{1,2} \sim U[0, 1]$, y actualice las velocidades de las partículas usando la Ecuación 2.26,
 - b) actualice las posiciones de las partículas usando la Ecuación 2.27,
 - c) actualice \mathbf{p}_i como sigue

$$\{\mathbf{p}_i = \operatorname{argmin}(f(\mathbf{x}_i), f(\mathbf{p}_i))\}_{i=1}^s, \quad (2.29)$$

- d) actualice \mathbf{g} usando la Ecuación 2.28.
3. Finalización: vaya al Paso 2 hasta que el criterio de término sea satisfecho.

Sin embargo, no es práctico utilizar la versión básica de PSO, ya que la estabilidad y convergencia del algoritmo están fuertemente limitadas. Por ello, poco después de su creación aparecen dos de las variantes de PSO más ampliamente conocidas y usadas por su simplicidad de implementación y complejidad computacional: 1) PSO con peso de inercia (PSOiw) (Shi and Eberhart, 1998a), y 2) PSO con factor de constricción (PSOcf) (Clerc, 1999). La versión PSOiw incorpora el parámetro w (peso de inercia) en la ecuación de velocidad (Ecuación 2.27), de forma que se obtiene la nueva ecuación para la velocidad

$$v_{i,d}(k+1) = wv_{i,d}(k) + c_1r_1(k)(p_{i,d}(k) - x_{i,d}(k)) + c_2r_2(k)(g_d(k) - x_{i,d}(k)), \quad (2.30)$$

donde $w \in [0, 1]$ sirve para limitar la velocidad de las partículas, y consecuentemente lograr convergencia a un punto de equilibrio, lo que contribuye a los análisis de estabilidad y convergencia de las partículas. De la misma forma, PSOcf incorpora al término χ , llamado factor de constricción, que funciona similar al peso de inercia, pero multiplicando a toda la Ecuación 2.26, en vez de solamente al término $v_{i,d}(k)$. Sin embargo, en Eberhart and Shi (2000) se muestra que ambas versiones del algoritmo son equivalentes.

A pesar de que PSOiw y PSOcf son las primeras y más conocidas variantes del PSO original,

muchas otras variantes y/o híbridos de PSO (Reyes-Sierra and Coello, 2006; Del Valle et al., 2008; de Oca et al., 2009; Kameyama, 2009) han surgido como el resultado de la incorporación de diversos enfoques a la técnica, cada uno de los cuales eventualmente pudiendo llegar a tener un desempeño sobresaliente en algunos espacios de búsqueda pero deficiente en otros. A causa de esto, en Bratton and Kennedy (2007) se planteó la posibilidad de establecer un estándar para la técnica mediante la definición de una línea de base en aspectos como topología, ecuaciones de actualización de velocidad, tamaño e inicialización del enjambre, y condiciones de borde, de modo de proveer un medio de comparación para futuros desarrollos y mejoras.

Si bien PSO ha mostrado ser eficaz y eficiente en diversos problemas prácticos de naturaleza muy variada Poli (2008), los beneficios de PSO no sólo son el resultado de una buena sintonización de sus parámetros, sino también de la elección de una función fitness adecuada para calificar las bondades de las potenciales soluciones. Como técnica heurística, PSO tiene la ventaja de poder usar funciones fitness más versátiles que en el caso de una técnica determinística, pudiéndose implementar sin mayor complicación funciones no diferenciables, no lineales y/o discontinuas. Además, cabe resaltar que PSO en sus diferentes variantes y/o modificaciones es una muy buena alternativa de solución, comparada con técnicas como GA (Rahmat-Samii, 2003; Habib and Alkazemi, 2005) y DE (Dong, 2009; Semnani et al., 2009), para problemas de optimización global con múltiples máximos y mínimos, discontinuidades y con soluciones determinísticas en tiempo no polinomial. Esto también se puede ver reflejado en el creciente aumento (cercanamente exponencial) de aplicaciones exitosas basadas en PSO (Poli, 2008).

2.5.1. Esquemas de configuración

Por fines prácticos relacionados con el análisis de la aplicabilidad de PSO a los problemas abordados, se decidió utilizar una de las versiones más simples cercanas a la versión estándar: PSOiw, para el cual se tienen los siguientes parámetros de configuración:

- Peso de inercia: w .
- Coeficientes de aceleración cognitiva y social: c_1 y c_2 , respectivamente.
- Tamaño de población: s .
- Máximo número de iteraciones: $iter_{max}$.
- Condiciones iniciales y criterios de término.

La dimensión de las partículas λ no se asume como parámetro de configuración, puesto que generalmente es impuesto por el problema a abordar, representando el número de incógnitas en la función de fitness. Para el resto de los parámetros existen diversos estudios sobre su adecuada definición, los cuales son presentados a continuación.

2.5.1.1. Peso de inercia y coeficientes de aceleración

El peso de inercia fue introducido por primera vez en Shi and Eberhart (1998a). En la actualidad existen diversos enfoques para la elección adecuada del peso de inercia, entre los cuales se incluyen:

1. Peso de inercia constante: (Shi and Eberhart, 1998a; Eberhart and Shi, 2000).

2. Peso de inercia variable en el tiempo:

- Variación lineal: peso de inercia linealmente decreciente (Shi and Eberhart, 1998b, 1999; Hu et al., 2009), peso de inercia linealmente creciente (Zheng et al., 2003a).
- Variación no lineal: peso de inercia estocástico (Eberhart and Shi, 2001a), peso de inercia calculado por medio de lógica difusa (Shi and Eberhart, 2001), peso de inercia basado en análisis de Lyapunov (Fan et al., 2009), y un *survey* de estrategias no lineales presentado en Hu et al. (2009) que incluye estrategia diferencial no lineal, peso de inercia en función de dos nuevos parámetros, peso de inercia dependiente de la posición de las partículas entre otros.

De entre los diversos esquemas de variación de $w(k)$ (Hu et al., 2009), uno de los esquemas más típicamente usado es el de variación linealmente decreciente respecto del avance del contador de iteraciones (Del Valle et al., 2008), definido de la siguiente manera

$$w(k) = w_{\max} - [w_{\max} - w_{\min}] \frac{k}{iter_{\max}}. \quad (2.31)$$

Entre las ventajas de usar el esquema de peso de inercia linealmente decreciente se encuentra la de ofrecer un equilibrio entre exploración y explotación. Sin embargo, se obtiene un mejor beneficio al escoger unos coeficientes de aceleración adecuados para cada esquema de variación del peso de inercia, para lo cual algunas configuraciones favorecen que $c_1 = c_2 = c$ (Jiang et al., 2007b), y en otras se prefiere usar unos coeficientes variantes en el tiempo (Rapaić and Željko Kanović, 2009). Las configuraciones más comúnmente usadas son presentadas en la Tabla 2.1.

w	c_1	c_2	Referencia
1	2	$c_2 = c_1$	Eberhart and Shi (2000)
1	2,8	1,3	Carlisle and Dozier (2001)
0,729	1,4944	$c_2 = c_1$	Eberhart and Shi (2000)
0,729	2,05	$c_2 = c_1$	Clerc and Kennedy (2002)
0,6	1,7	$c_2 = c_1$	Trelea (2003)
0,9 → 0,4	2	$c_2 = c_1$	Eberhart and Shi (2000)
0,9 → 0,4	1,4944	$c_2 = c_1$	Rapaić and Željko Kanović (2009)
0,4 → 0,9	$\lceil 1,5 + \left(\frac{Rnd}{2}\right) \rceil$	$\lceil 1,5 + \left(\frac{Rnd}{2}\right) \rceil$	Zheng et al. (2003a)
0,9 → 0,4	2,5 → 0,5	0,5 → 2,5	Rapaić and Željko Kanović (2009)
$0 \leq w < 1$	$0 \leq c_1 < 1$	$c_2 = c_1$	Jiang et al. (2007b)
$\lceil 0,5 + \left(\frac{Rnd}{2}\right) \rceil$	1,4944	$c_2 = c_1$	Eberhart and Shi (2001b,a)

Cuadro 2.1.: Configuraciones con características convergentes.

2.5.1.2. Tamaño de enjambre y número máximo de iteraciones

En relación al número de partículas a utilizar, se sabe que su adecuada elección está relacionada directamente con la complejidad del problema de optimización que se busca resolver, complejidad

que generalmente es determinada por la dimensión de las partículas o la definición del funcional de fitness. Mientras que un tamaño grande del enjambre permite una mayor exploración del espacio de búsqueda, los costos computacionales se ven directamente afectados por la mayor cantidad de cálculos a realizar. Por otra parte, un tamaño de población insuficiente generalmente conduce a una convergencia prematura.

Los primeros estudios acerca de un valor adecuado para el tamaño del enjambre hablan de entre 10 y 50 partículas para problemas sencillos, y entre 100 y 200 partículas para problemas más complejos (Carlisle and Dozier, 2001). Sin embargo, una práctica común, más acorde a determinado tipo de problema, es la de escoger un tamaño de enjambre que sea mayor o igual a dos veces la dimensión de las partículas, es decir

$$s \geq 2\lambda, \tag{2.32}$$

siendo λ la dimensión de las partículas. No obstante, en Particle Swarm Central (2007) se presenta una forma automática más acertada de escoger el tamaño s de la población a evolucionar, igualmente en dependencia de la dimensión de las partículas, con s siendo calculado de la forma

$$s = 10 + 2\sqrt{\lambda}, \tag{2.33}$$

siendo λ la dimensión de las partículas.

Existen algunos enfoques en los cuales se utiliza un tamaño de población variable. En Leong and Yen (2008) puede encontrarse una recopilación de las principales técnicas en esta categoría. Por su parte, el estudio de Leong and Yen (2008) sobre un tamaño de población dinámico arroja como conclusiones que una población creciente beneficia las capacidades de exploración y explotación, mientras que una estrategia de decrecimiento de la población sirve para controlar la dinámica del tamaño del enjambre.

Cabe aclarar que, a pesar de la existencia de esquemas para definir el valor adecuado de s , su valor óptimo depende del problema a ser abordado, al igual que el parámetro $iter_{\max}$. Por tanto, $iter_{\max}$ y s son parámetros problema-dependientes. Para el caso del parámetro $iter_{\max}$ se suele escoger un valor por defecto, valor que puede estar sujeto a cambios según el caso de estudio.

2.5.1.3. Inicializaciones y criterios de término

Usualmente se suele inicializar las posiciones de las partículas de forma aleatoria con distribución uniforme (Clerc, 2008). Sin embargo, este tipo de inicialización no es el único, ni el mejor, y usualmente sólo es adecuado para espacios de búsqueda poco complejos. Otros tipos más beneficiosos de inicialización son los propuestos en Clerc (2008), listados a continuación:

- Inicialización basada en muestreo con puntos Hammersley, más adecuada para sistemas con componentes estocásticos;
- Inicialización basada en método de teselaciones (método potencial), que favorece una distribución mucho más uniforme sobre espacios de búsqueda complejos; y
- Distribución al azar con sesgo, que usa distribución no uniforme debido a la premisa de que no todos los óptimos debieran estar distribuidos uniformemente.

Otro estudio importante acerca de como las inicializaciones pueden mejorar el desempeño de PSO es mostrado en Norouzzadeh et al. (2010). Aquí se adiciona un nuevo parámetro llamado *plow operator*, que sirve básicamente para variar aleatoriamente una dimensión de las partículas por iteración, mientras que las otras dimensiones permaneces fijas. De esta forma, se dice que este operador actúa como el arado en la agricultura, y en este caso ayuda a PSO hacer mejor búsqueda adicionando las ventajas de la búsqueda aleatoria ciega.

Respecto de los criterios de término, éstos son los parámetros de diseño más fácilmente sintonizables, y generalmente se suele utilizar, con prioridad 1, el cruce de $f(\mathbf{x}_i)$ por un umbral definido como

$$f(\mathbf{x}_i) < f_{\min},$$

y, con prioridad 2, el sobrepaso del máximo número de iteraciones definido como

$$k > iter_{\max}.$$

2.5.2. Estabilidad de PSO

El tema de la estabilidad y convergencia de PSO estándar (indistintamente PSOiw o PSOcf) es un tema ampliamente discutido, pero a la vez es analizado sesgadamente debido a la naturaleza compleja del mismo algoritmo, que incluye: un conjunto de al menos cuatro ecuaciones de actualización (en su versión estándar en Bratton and Kennedy (2007)), de las cuales la principal es una ecuación en diferencias estocástica; múltiples variables y parámetros de ajuste; actualización no lineal de algunas de sus variables; y, eventualmente, parámetros variables en el tiempo, entre otros aspectos. Es importante aclarar que demostrar la estabilidad del algoritmo no implica demostrar convergencia ni siquiera a un óptimo local, y ello complica aún más el análisis. Entonces, como un primer acercamiento a la estabilidad del algoritmo, diversos autores han optado por analizar:

- Versiones reducidas del mismo: se analizan partículas de una dimensión (Ozcan and Mohan, 1998; Clerc and Kennedy, 2002; Zheng et al., 2003b; Liu et al., 2007a; Jiang et al., 2007b);
- Versiones determinísticas: variables estocásticas reemplazadas por sus valores esperados, máximos o mínimos (Ozcan and Mohan, 1999; Clerc, 1999; Clerc and Kennedy, 2002; Trelea, 2003; Chen et al., 2003; Zheng et al., 2003b; Yasuda et al., 2003; Yasuda and Iwasaki, 2004; Liu et al., 2007a; Liang et al., 2010);
- Versiones estacionarias: ciertas variables se suponen que han llegado a un estado estacionario, o son constantes desde un principio (Ozcan and Mohan, 1999; Clerc and Kennedy, 2002; Jiang et al., 2007b); y
- Versiones linealizadas: aproximación lineal de no linealidades (Emara and Abdel Fattah, 2004).

Adicionalmente, los diferentes enfoques para analizar las distintas versiones propuestas son tan variados como:

- Resolución de ecuaciones en diferencias y/o recurrencias determinísticas: análisis de trayectorias para la solución explícita y/o análisis de autovalores (Ozcan and Mohan, 1998, 1999;

Van Den Bergh, 2002; Clerc and Kennedy, 2002; Chuan and Quanyuan, 2007; Poli et al., 2007; Liu et al., 2009; Zhao and Mao, 2009; Qingqing et al., 2009);

- Asimilación del sistema de ecuaciones como un sistema realimentado: localización de polos (Sen and Dey, 2007; Samal et al., 2007, 2008);
- Resolución de ecuaciones en diferencias estocásticas con convergencia en probabilidad: análisis del valor esperado y la varianza (Van Den Bergh, 2002; Liu et al., 2007a; Jiang et al., 2007b,a; Zhang et al., 2008; Lian et al., 2008; Rapaić and Željko Kanović, 2009; Mei et al., 2010);
- Método directo de Lyapunov: análisis con funciones candidatas de Lyapunov (Emara and Abdel Fattah, 2004; Kadirkamanathan et al., 2006; Liu et al., 2007a; Bhattacharya et al., 2008; Fan et al., 2009; Wakasa et al., 2009);
- Método basado en límites: análisis de la estabilidad de PSO por medio de límites (Sun et al., 2004; Guilan et al., 2008; Lian et al., 2008; Ren et al., 2008); y
- Desigualdades lineales de matrices: análisis por solución de LMIs (Wakasa et al., 2009).

No obstante, uno de los resultados más importantes indica que, a pesar de todas la ventajas prácticas que PSO ofrece, la convergencia al máximo global no está garantizada, ni siquiera para el caso de un mínimo local (Van Den Bergh, 2002; Jiang et al., 2007b, Sección 3.3.2).

A pesar de que los diversas vías de análisis propuestas han sido de mucha ayuda para entender el comportamiento del algoritmo respecto de su estabilidad, son sólo acercamientos y aún no existe una metodología general que permita el análisis completo del algoritmo con todas las complejidades. Sin embargo, respecto del análisis de convergencia existen resultados reportados en la literatura especializada que son de considerable ayuda. Uno de estos resultados es la Tesis Doctoral de Frans Van Den Bergh (Van Den Bergh, 2002), en la cual se hace el análisis de la estabilidad del algoritmo para una versión reducida, determinística y estacionaria y luego se hace la extensión al caso general, estocástico y variable en el tiempo. El análisis de la convergencia del algoritmo demuestra que las condiciones de estabilidad no aseguran convergencia ni siquiera local. De esta forma Van Den Bergh demuestra que PSO no es un optimizador local, y que si bien los resultados prácticos demuestran convergencia en general a buenas soluciones, esto se debe a que el mismo componente estocástico que introduce una probabilidad de convergencia prematura a cualquier estado, eventualmente permite llegar a un mínimo local. Por lo tanto, Van den Bergh propone modificaciones al algoritmo PSO estándar para hacer de éste un optimizador local o global: la adición de una ley de ajuste diferente para la velocidad de la mejor partícula, que incluye un nuevo parámetro variable cuyo ajuste demanda el uso de otras 4 ecuaciones, permitiendo la demostración de convergencia local; y la reiniciación del proceso de optimización manteniendo la mejor solución anterior conduce a la demostración de convergencia global.

A partir del trabajo seminal en Van Den Bergh (2002), otras variantes de PSO con características de convergencia global han sido derivadas, como el caso de Peer et al. (2003); Brits et al. (2003); Cui and Zeng (2004); Zhihua et al. (2004); Brits et al. (2007) y Tang and Bagchi (2010). De los anteriores, el trabajo de Van Den Bergh (2002) se tomará como referencia en el desarrollo de la presente Tesis, por lo cual será tomado en más detalle en la siguiente subsección.

2.5.3. GCPSO: PSO con convergencia garantizada

Como se muestra en Van Den Bergh (2002), el algoritmo PSO no es un optimizador local, y menos global. Debido a esto, es necesario la inclusión de una modificación al algoritmo básico para poder asegurar la convergencia, en un principio, local. A esta versión modificada de PSO se le llamará GCPSO (por sus siglas en inglés *Guaranteed Convergence PSO*) (Van Den Bergh, 2002; Van Den Bergh and Engelbrecht, 2002), y su descripción detallada se presenta a continuación. Es de aclarar que la información contenida en esta sección es tomada exactamente de Van Den Bergh (2002) a fin de que este documento sea lo más autocontenido posible para el lector. No obstante, las respectivas demostraciones deberán ser verificadas en la respectiva referencia, pues por motivos de espacio no se incluyen en este documento.

Sea τ el índice de la mejor partícula tal que

$$x_{\tau,d} = g_d.$$

De esta forma, la ecuación de actualización de la velocidad para la mejor partícula global es

$$v_{\tau,d}(k+1) = wv_{\tau,d}(k) + (g_d(k) - x_{\tau,d}(k)) + \rho(k)(1 - 2r_{2,d}(k)) \quad (2.34)$$

donde ρ es un factor de escala definido por

$$\rho(k+1) = \begin{cases} 2\rho(k) & \text{si } \#éxitos > s_c \\ 0,5\rho(k) & \text{si } \#fracasos > f_c \\ \rho(k) & \text{de otra forma} \end{cases}, \quad (2.35)$$

donde $\#éxitos$ y $\#fracasos$ denotan el número consecutivo de éxitos o fallas, respectivamente. Una falla se define como $f(g_d(k)) = f(g_d(k-1))$, y s_c y f_c son umbrales que dependerán de la función fitness. Empíricamente se ha hallado que $\rho(0) = 1$, $s_c = 15$ y $f_c = 5$ son recomendados para obtener resultados aceptables. Adicionalmente, para que la Ecuación 2.35 esté bien definida se necesitan de las siguientes reglas de actualización:

$$\#éxitos(k+1) > \#éxitos(k) \Rightarrow \#fracasos(k+1) = 0,$$

$$\#fracasos(k+1) > \#fracasos(k) \Rightarrow \#éxitos(k+1) = 0.$$

Consecuentemente, el valor de ρ se adaptará de modo de alcanzar el paso óptimo de volumen de muestreo dado el estado actual del algoritmo.

Para una detallada prueba de la convergencia local de GCPSO, a continuación se presentan los teoremas, lemas, y demás elementos útiles para dicho fin.

Proposición 2.5.2. (*Criterio de convergencia local*) (Solis and Wets, 1981; Van Den Bergh, 2002): Dada una función f de \mathbb{R}^n a \mathbb{R} , y S un subconjunto de \mathbb{R}^n . Entonces se busca un punto z en S que minimiza f en S o al menos brinda una aproximación aceptable del ínfimo de f en S .

Algoritmo 2.5.3. (*Algoritmo de optimización aleatorio básico*) (Solis and Wets, 1981; Van Den Bergh, 2002):

Paso 0: Encuentre un \mathbf{z}_0 en S y defina $k = 0$.

Paso 1: Genere un vector ξ_k en el espacio de muestreo $(\mathbb{R}^n, \mathfrak{B}, \mu_k)$.

Paso 2: Defina $\mathbf{z}_{k+1} = D(\mathbf{z}_k, \xi_k)$, elija μ_{k+1} , defina $k := k + 1$ y retorne al Paso 1.

Condición. H1 (Algoritmo de optimización) (Solis and Wets, 1981; Van Den Bergh, 2002): Si $f(D(\mathbf{z}, \xi)) \leq f(\mathbf{z})$ y si $\xi \in S$, entonces $f(D(\mathbf{z}, \xi)) \leq f(\xi)$.

Definición 2.5.4. (Ínfimo esencial) (Solis and Wets, 1981; Van Den Bergh, 2002): Se define el ínfimo esencial ψ de una función f como

$$\psi = \inf (t : v[\mathbf{z} \in S | f(\mathbf{z}) < t] > 0)$$

donde $v[A]$ es la medida de Lebesgue sobre el conjunto A .

Condición. H2 (Optimización global) (Solis and Wets, 1981; Van Den Bergh, 2002): Para cualquier subconjunto (Borel) A de S con $v[A] > 0$, se tiene que

$$\prod_{k=0}^{\infty} (1 - \mu_k[A]) = 0$$

donde $\mu_k[A]$ es la probabilidad de que A sea generada por μ_k , y $v[A]$ es la medida de Lebesgue sobre el conjunto A .

Definición 2.5.5. (Región de optimalidad) (Solis and Wets, 1981; Van Den Bergh, 2002): Se define la región de optimalidad R_ϵ como:

$$R_\epsilon = \{\mathbf{z} \in S | f(\mathbf{z}) < \psi + \epsilon\}$$

con $\epsilon > 0$.

Condición. H3 (Optimización local) (Solis and Wets, 1981; Van Den Bergh, 2002): Para cualquier $\mathbf{z}_0 \in S$, corresponde un $\gamma > 0$ y un $0 < \eta \leq 1$ tal que:

$$\mu_k[(\text{dist}(D(\mathbf{z}, \xi), R_\epsilon) < \text{dist}(\mathbf{z}, R_\epsilon) - \gamma) \text{ o } (D(\mathbf{z}, \xi) \in R_\epsilon)] \geq \eta$$

para todo k y todo \mathbf{z} en el conjunto compacto $L_0 = \{\mathbf{z} \in S | f(\mathbf{z}) \leq f(\mathbf{z}_0)\}$.

Teorema 2.5.6. (*Búsqueda global*) (Solis and Wets, 1981; Van Den Bergh, 2002): Supongamos que f es una función medible, S es un subconjunto medible de \mathbb{R}^n , y **H1** y **H2** se satisfacen. Sea $\{z_k\}_{k=0}^{+\infty}$ una secuencia generada por un algoritmo dado. Entonces

$$\lim_{k \rightarrow +\infty} P[z_k \in R_\epsilon] = 1$$

donde $P[z_k \in R_\epsilon]$ es la probabilidad de que en el paso k , el punto z_k generado por el algoritmo esté en R_ϵ .

Demostración. La demostración del Teorema 2.5.6 se puede ver en Solis and Wets (1981) o Van Den Bergh (2002). \square

Teorema 2.5.7. (*Búsqueda local*) (Solis and Wets, 1981; Van Den Bergh, 2002): Supongamos que f es una función medible, S es un subconjunto medible de \mathbb{R}^n , y **H1** y **H3** se satisfacen. Sea $\{z_k\}_{k=0}^{+\infty}$ una secuencia generada por un algoritmo dado. Entonces

$$\lim_{k \rightarrow \infty} P[\mathbf{z}_k \in R_\epsilon] = 1$$

donde $P[z_k \in R_\epsilon]$ es la probabilidad de que en el paso k , el punto \mathbf{z}_k generado por el algoritmo esté en R_ϵ .

Demostración. La demostración del Teorema 2.5.7 se puede ver en Solis and Wets (1981) o Van Den Bergh (2002). \square

Teorema 2.5.8. (Van Den Bergh, 2002): GCPSO es un algoritmo de optimización local.

Demostración. Para comenzar, se tiene GCPSO realiza la actualización de g y p_i por medio de (Ecuación 2.28) y (Ecuación 2.29), y con esto se satisface que

$$D(p_{i,d}(k), x_{i,d}(k)) = \begin{cases} p_{i,d}(k) & \text{si } f(p_{i,d}(k)) \leq f(x_{i,d}(k+1)) \\ x_{i,d}(k+1) & \text{si } f(p_{i,d}(k)) > f(x_{i,d}(k+1)) \end{cases}.$$

Consecuentemente, GCPSO cumple con **H1**. Ahora, considerando la ecuación de actualización usada por GCPSO (Ecuación 2.34), se tiene que ésta representa la acción de muestrear un punto de un hipercubo con lados 2ρ y centrado alrededor de $\mathbf{g} + w\mathbf{v}$, denotando este cubo como M_k , y μ_k denotando la medida uniforme de probabilidad definida sobre el hipercubo M_k . El estancamiento entonces se previene con $\rho > 0$ para todos los pasos de avance.

Del hecho que **H3** se satisface, y con $L_0 = \{\mathbf{z} \in S | f(\mathbf{z}) \leq f(\mathbf{z}_0)\}$, entonces \mathbf{g} siempre está en L_0 . Sin embargo, es posible que $x_{\tau,k} \notin L_0$, debido al efecto acumulativo de un creciente vector \mathbf{v} , por lo cual $\mathbf{g} + w\mathbf{v}_k \notin L_0$. Aparecen de esta forma dos escenarios: $g \in M_k$ o $g \notin M_k$. El primer caso significa que un punto arbitrariamente cercano a g puede ser muestreado, incluyendo g mismo. Ya que $g \in M_k$, esto significa que $v[M_k \cap L_0] > 0$, esto es, la intersección entre M_k y L_0 no es vacía. El segundo caso implica que ρ es tal que M_k no incluye a g . Esto sucede cuando v_τ apunta fuera de

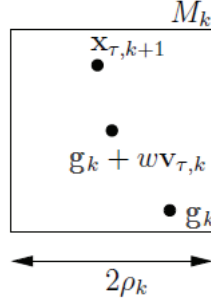


Figura 2.3.: Hipercubo soporte M_k del espacio de muestreo μ_k , centrado alrededor del punto $\mathbf{g}_k + \omega \mathbf{v}_k$, como es definido por el paso de actualización de la posición usando GCPSO. El punto \mathbf{x}_{k+1} es un ejemplo de un punto de muestreo por el nuevo paso de actualización de velocidad. (Van Den Bergh, 2002)

L_0 . Ya que g sólo se actualiza cuando se encuentra una mejor solución, y por la misma definición de L_0 , es claro que ninguno de los puntos fuera de L_0 serán seleccionados para reemplazar a g . Por otro lado, x_τ es capaz de moverse fuera de L_0 debido a la velocidad residual v_τ . Supóngase por el momento que ρ es insignificanemente pequeño. La Ecuación 2.34 muestra que, si $\omega < 1$, entonces claramente $\|\mathbf{x}_\tau(k+1) - \mathbf{g}\| < \|\mathbf{x}_\tau(k) - \mathbf{g}\|$, suponiendo que $\mathbf{x}_\tau(k) \neq \mathbf{g}$. Después de un número finito de pasos l , $\mathbf{x}_\tau(k+l)$ se encontrará en L_0 una vez más. Esto implica que $v[M_{k+l} \cap L_0] > 0$, y entonces un punto arbitrariamente cercano a g puede ser muestreado una vez más. Ambos casos implican que un nuevo punto muestreado arbitrariamente cercano a g , y por lo tanto en L_0 , puede ser generado. Cabe notar que el segundo caso ocurre sólo cuando g es cercano a los límites de L_0 . El primer caso, donde $M_k \subset L_0$, puede ser considerado el normal.

Por lo tanto GCPSO tiene un volumen no degenerado de muestreo μ_k con soporte M_k . Bajo este hecho, ahora es posible considerar la propiedad de convergencia local para GCPSO. Si se supone que S es compacto y tiene un interior no vacío, entonces L_0 también será compacto con interior no vacío. Adicionalmente, L_0 incluirá el ínfimo esencial, contenido en la región de optimalidad R_ϵ por definición. Ahora R_ϵ es compacto con interior no vacío, y por lo tanto se puede definir una bola B' centrada en \mathbf{c}' contenida en R_ϵ , tal como se muestra en la Figura 2.4.

Ahora, se encuentra el punto $\mathbf{x}' \in \operatorname{argmax}_{\mathbf{x}} \{\operatorname{dist}(\mathbf{c}', \mathbf{x}) \mid \mathbf{x} \in L_0\}$, tal como se ilustra en la Figura 2.4. Sea B el hipercubo centrado en \mathbf{c}' , con lados de longitud $2(\operatorname{dist}(\mathbf{c}', \mathbf{x}') - 0,5\rho)$. Sea C el casco convexo de \mathbf{x}' y B' . Considérese una línea tangente a B' pasando a través de \mathbf{x}' (es decir uno de los bordes de C). La línea más larga así definida es para x' , que es el punto más alejado de B' . Esto implica que el ángulo subtendido por \mathbf{x}' es el más pequeño de cualquier punto en L_0 . A su vez, esto implica que el volumen $C \cap B$ es más pequeño que el de $C' \cap B$ para cualquier otro casco convexo C' definido para cualquier punto arbitrario $\mathbf{x} \in L_0$.

Entonces, para todo \mathbf{x} en L_0

$$\mu_k[\operatorname{dist}(D(\mathbf{g}, \mathbf{x}_\tau), R_\epsilon) < \operatorname{dist}(x, R_\epsilon) - 0,5\rho] \geq \eta = \mu[C \cap B] > 0$$

donde μ_k es la distribución uniforme sobre el hipercubo centrado en \mathbf{x} , con lados de longitud 2ρ , cubo que es provisto por GCPSO.

Ya que $\mu[C \cap B] > 0$, la probabilidad de seleccionar un nuevo punto en \mathbf{x} tal que esté cercano a la

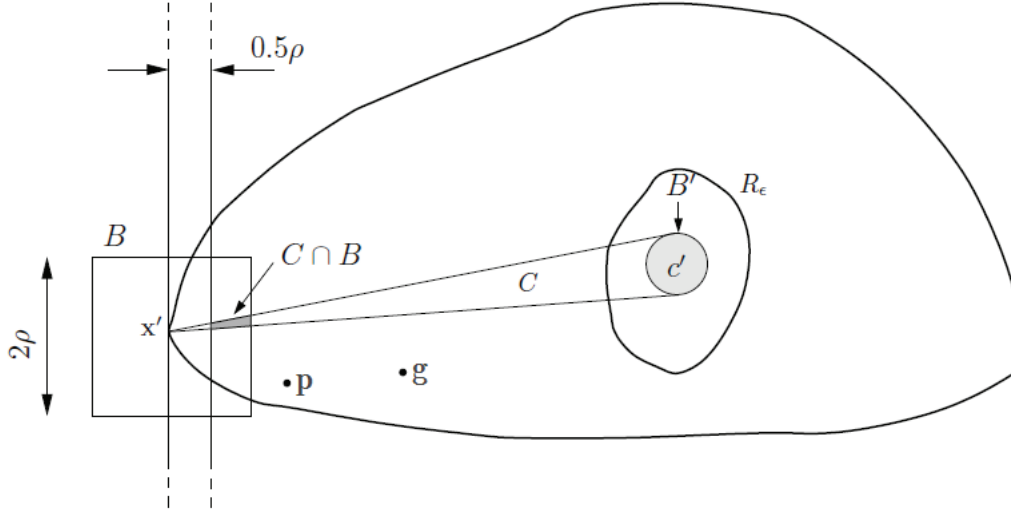


Figura 2.4.: Intersección $C \cap B$. (Solis and Wets, 1981; Van Den Bergh, 2002)

región de optimalidad R_ϵ es siempre diferente de cero. Esto es suficiente para mostrar que GCPSO cumple con **H3** porque:

1. GCPSO puede generar siempre una muestra alrededor de un punto en L_0 , suponiendo $g, p_i \in L_0$;
2. Dado un punto inicial en L_0 , GCPSO garantiza un volumen de muestreo no degenerado con probabilidad diferente de cero de muestrear un punto cercano a la región de optimalidad R_ϵ .

Bajo las condiciones de optimización local, y suponiendo que GCPSO satisface **H1** y **H3**, por el Teorema 2.5.7 se tiene que la secuencia de valores $\{\mathbf{g}_k\}_{k=0}^\infty$ generada por el algoritmo GCPSO convergerá a la región de optimalidad, independientemente del estado inicial del enjambre. \square

Teorema 2.5.9. (Van Den Bergh, 2002): Sea el enfoque GCPSO/PSO con múltiple reinicialización (MPSO) definido por el algoritmo:

1. Inicializar todas las partículas en posiciones aleatorias en el espacio de búsqueda,
2. Ejecutar el algoritmo GCPSO/PSO hasta que se cumpla el criterio de reinicio,
3. Mantener la mejor solución, y regresar al Paso 1.

Entonces, MPSO es un algoritmo de optimización global.

Demostración. Debido a que la actualización de las mejores posiciones personales en GCPSO son las mismas que en PSO original, se tiene que GCPSO cumple con **H1**. Ahora, el proceso de reinicialización asigna a cada partícula una posición muestreada desde todo el espacio de búsqueda S . Ya que el soporte del espacio de muestreo, M_k , es igual al espacio de búsqueda, se tiene que $v[M_k] = v[S]$. Esto hace que se satisfaga **H2**, lo que significa que, por el 2.5.6, MPSO es un algoritmo de búsqueda global. \square

2.5.4. Estado del arte de la técnica

PSO tiene sus raíces en los estudios desarrollados por Eberhart y Kennedy (Eberhart and Kennedy, 1995a,b), siguiendo la idea de imitar los procesos naturales de colectividades como las bandadas de pájaros o cardúmenes para desarrollar una nueva herramienta de optimización estocástica global. Esta idea surge principalmente por el hecho sobresaliente de que tales colectividades tienen mucho más éxito en la búsqueda de soluciones a sus problemas (hallar comida, defenderse de ataques, etc.) que cada una de sus partes por separado. De tal forma, y basados en los resultados de estudios en áreas como la biología y la sociología, estos autores se embarcan en el proyecto de modelar matemáticamente el comportamiento de las colectividades (incluyendo aspectos como inteligencia, movimiento, comunicación y aprovechamiento de la experiencia), y aplicar los resultados a la optimización.

Diversas versiones para la implementación de la técnica PSO han salido a la luz. Por ejemplo, gracias a su éxito a nivel de optimización global, existen también esquemas que permiten su aplicación a nivel de optimización local (Eberhart and Shi, 2001a). El tipo de actualización de la posición de las partículas conlleva igualmente a diversos esquemas de implementación. Se distinguen en este caso PSO síncrono y PSO asíncrono (Carlisle and Dozier, 2001; Pérez-López, 2005). Sin embargo, a pesar de que las anteriores vertientes son las más popularmente difundidas, existen versiones y mejoras tan variadas como se les puedan ocurrir a los investigadores en pro de impulsar el rendimiento de aplicaciones específicas. Existen por su parte versiones simplificadas de PSO (Kennedy and Eberhart, 1997; Ozcan and Mohan, 1998; Liu et al., 2007a), pero más que una variante del algoritmo son desarrollos que se han hecho para facilitar un poco el trabajo de encontrar pruebas de estabilidad y poder asegurar convergencia (Ozcan and Mohan, 1998; Clerc and Kennedy, 2002; Trelea, 2003; Liu et al., 2007a; Jiang et al., 2007b). En el área de estabilidad y convergencia cabe decir que, dada la naturaleza estocástica del método, los avances son bastante limitados, y las pruebas existentes en la bibliografía especializada son pocas en la actualidad. Otro punto importante a tener en cuenta en los progresos de la técnica PSO es que recientemente se están desarrollando estudios sobre su aplicación en tiempo continuo (Fernández-Martínez et al., 2008; Fernández-Martínez and García-Gonzalo, 2008), y esto es porque inicialmente fue diseñado para tiempo discreto y su vinculación con tiempo continuo se resume a su aplicación en determinados instantes de muestreo (Coelho and Guerra, 2007). Por otro lado, se advierte que en la actualidad existe una gran variedad de versiones modernas de PSO entre las que se encuentran: adaptaciones al caso de funciones multiobjetivo (*multi-objective particle swarm optimization*, MP-SO) (Reyes-Sierra and Coello, 2006), como el caso de *vector evaluated particle swarm optimization* (VEPSO) (Parsopoulos et al., 2004); versiones basadas en cooperación, como *cooperative particle swarm optimization* (CPSO) (van den Bergh and Engelbrecht, 2004); versiones basadas en estrategias de aprendizaje, como *comprehensive learning particle swarm optimization* (CLPSO) (Liang et al., 2006) y *fully informed particle swarm* (FIPS) (Mendes et al., 2004); versiones basadas en subpoblaciones, como *particle swarm optimization with migration* (MPSO) (Jordan et al., 2008) y *quantum-behaved particle swarm optimization* (QPSO) (Sun et al., 2004); versiones adaptables de PSO, como *adaptive particle swarm optimization* (APSO) (Zhan et al., 2009) y *adaptive hierarchical particle swarm optimizer* (AHPSO) (Janson and Middendorf, 2005), entre las más relevantes; y versiones híbridas de PSO, como (de Oca et al., 2009).

PSO guarda relación con los GA, al basarse ambos en la idea de imitar procesos naturales. Sin

embargo, a diferencia de éstos, PSO tiene la ventaja inherente de poseer un funcionamiento mucho más intuitivo y simple, una implementación mucho más rápida y sencilla (Pérez-López, 2005), y arrojar mejores resultados en problemas de optimización tipo *benchmark* (van den Bergh and Engelbrecht, 2004). Debido a esto, y teniendo en cuenta que los EA resultan ser una muy buena alternativa de solución para problemas de optimización global con múltiples máximos y mínimos, discontinuidades y soluciones determinísticas en tiempo no polinomial, PSO resulta ser una muy buena opción alternativa frente a los GA (Rahmat-Samii, 2003; van den Bergh and Engelbrecht, 2004; Boeringer and Werner, 2004). Esto último también se puede ver reflejado también en el creciente aumento (cercanamente exponencial) de aplicaciones exitosas apoyadas en PSO (Poli, 2008), y en campos de aplicación tan variados como (Weise, 2008; Poli, 2008): diseño de antenas, entrenamiento de redes neuronales, biomedicina, procesamiento de señales, sistemas de potencia, control de sistemas dinámicos, servicios de telecomunicación, gestión empresarial, redes de distribución, teoría de juegos, electrónica y electromagnetismo, entretenimiento, procesamiento de imágenes y video, modelamiento, y robótica.

Finalmente, se presenta la revisión de aplicaciones de PSO en las áreas a tratar en la presente Tesis Doctoral. Para el caso específico del problema CQLF, no se encontraron aplicaciones de PSO en las bases de datos de bibliografía especializada consultadas. Respecto de la identificación en línea, se encontró el trabajo de Liu et al. (2007b) en el cual se presenta la identificación paramétrica en línea basada en PSO, aplicada al caso a motores sincrónicos de imán permanente, y en Liu et al. (2008) se presenta el estudio de su implementación en tiempo real. Por otra parte, en Su et al. (2009) se presenta el estudio de la identificación en línea basada en CPSO, cuyos resultados se aplican en Zhao et al. (2009). Sin embargo, tanto los trabajos de Liu como el de Su se orientan a la configuración de la aplicación, sin ahondar en características de los resultados obtenidos más allá de la reducción del tiempo de convergencia y el error promedio.

En cuanto a control adaptable basado en modelos de error, sólo fueron encontradas un par aplicaciones del mismo autor en control adaptable por modelo libre (MFAC por *Model-Free Adaptive Control*) (Coelho and Guerra, 2007; dos Santos-Coelho and Coelho, 2009), para regulación y sin comparaciones con otros métodos, mostrando convergencia solamente por resultados de simulación. No obstante, no se encontraron aplicaciones de PSO para el caso de MRAC. Además, dos publicaciones adicionales (Wang et al., 2005; Liu et al., 2007c) relacionan el control adaptable inverso con PSO, aplicación en la cual se utiliza PSO para optimizar fuera de línea una red neuronal que hace las veces de controlador adaptable. Por tanto, en cuanto a PSO se refiere, estos últimos dos trabajos no son más que una aplicación de PSO al entrenamiento de una red neuronal, y además en forma off-line.

3. PSO en el Problema CQLF

3.1. Introducción

La existencia de una CQLF para un SLS es garantía de su estabilidad asintótica global. A pesar de que la determinación de condiciones para la existencia/no-existencia de una CQLF ha tenido diversos avances, sobre todo en parejas de sistemas de orden n y en N sistemas de segundo orden, hoy en día sigue siendo un problema abierto para el caso de N sistemas de orden n . Es innegable entonces la importancia de poder asegurar la existencia o no-existencia de una CQLF para un determinado sistema conmutado, pero tanto o más importante es calcularla para obtener información más específica acerca del comportamiento (estabilidad) del sistema conmutado bajo análisis.

En esa dirección, en el presente capítulo se desarrolla una nueva metodología que permite el cálculo de una CQLF basada en la técnica PSO. Por otra parte, en este capítulo también se deduce una condición necesaria para la existencia de una CQLF, y con base en ella se diseña una metodología para determinar la no-existencia de una CQLF utilizando la técnica PSO y el software Matlab. Diversos análisis comparativos se presentan para mostrar las fortalezas y ventajas de la metodología propuesta, extrayendo las respectivas conclusiones.

Todos los desarrollos en el presente capítulo han sido avalados por la comunidad científica, debidamente reportados en Ordóñez-Hurtado and Duarte-Mermoud (2011b), Ordóñez-Hurtado and Duarte-Mermoud (2012) y Duarte-Mermoud et al. (2012).

3.2. Cálculo de una CQLF usando PSO

Para el desarrollo de la nueva metodología de cálculo propuesta en esta Tesis se plantea un proceso de diseño que consta de los siguientes pasos:

1. Diseñar la(s) función(es) de fitness a optimizar con PSO.
2. Definir la forma en que las partículas a evolucionar representan las potenciales soluciones factibles en el problema a resolver.
3. Definir una correcta implementación la técnica PSO para obtener un buen desempeño. Esto incluye: configuración de los parámetros de PSO, depuración de la programación para minimizar tiempos de procesamiento, e identificación de las mejores condiciones iniciales, entre otras.

A continuación se desarrolla cada uno de los puntos mencionados anteriormente, a fin de describir los pasos teóricos relacionados con el diseño de la nueva metodología de cálculo de una CQLF, para un conjunto de matrices estables $A = \{A_1, \dots, A_N\}$, basada en PSO.

3.2.1. Funciones de fitness

A partir del segundo funcional propuesto en Liberzon and Tempo (2004), el cual es aquí llamado f_g y definido como

$$f_g(R) := v(R, A_i) := \max \left\{ \text{eig} \left(RA_i + A_i^T R \right) \right\}, \quad i \in \{1, \dots, N\}, \quad (3.1)$$

cuyo gradiente es

$$\partial_R v(R, A_i) = A_i x x^T + x x^T A_i^T, \quad (3.2)$$

siendo $\text{eig}(\cdot)$ la función que entrega los valores propios de una matriz, se propone un funcional genérico $f(R)$. No obstante, este funcional aquí propuesto no se utiliza de la misma forma que f_g en Liberzon and Tempo (2004) (minimizando respecto de una matriz e iterando el proceso para otra matriz), sino que permite la minimización respecto de todas las matrices en cada iteración. Adicionalmente, se definen dos funcionales específicos $f_1(R)$ y $f_2(R)$ a partir de las diferentes representaciones que se dan para la potencial solución factible R .

Sin pérdida de generalidad, sea R una matriz simétrica en $\mathfrak{R}^{n \times n}$ candidata a solucionar simultáneamente las N ecuaciones de Lyapunov (Ecuación 2.4) generadas por N sistemas, es decir, candidata a ser una CQLF para el SLS definido por Ecuación 2.1. Se define entonces el funcional genérico

$$f(R) = \max_i \left\{ \text{eig} \left(RA_i + A_i^T R \right) \right\}, \quad \forall i = 1, \dots, N. \quad (3.3)$$

Es evidente el hecho de que R sea una matriz simétrica no implica que sea positiva definida, lo cual es requisito primordial para que R produzca una CQLF de la forma $V(x) = x^T R x$. Por este hecho, se debe restringir el espacio de búsqueda a matrices positivas definidas para mejorar la convergencia del proceso de optimización, al no tener que evaluar potenciales soluciones no factibles.

Una primera forma de restringir el espacio de búsqueda se basa en la representación $R = V^{-1} D V$, siendo V la matriz que diagonaliza a R , y D una matriz diagonal semejante a R con entradas d_1, \dots, d_n , es decir $D = \text{diag}(d_1, \dots, d_n)$. De esta forma se utiliza (Ecuación 3.3) pero se reparan las soluciones no factibles por medio de la proyección

$$\tilde{R} = V^{-1} \tilde{D} V, \quad (3.4)$$

con

$$\tilde{D} = \text{diag}(|d_1|, \dots, |d_n|), \quad (3.5)$$

donde d_1, \dots, d_n son obtenidos de D . La consideración de que R es diagonalizable es razonable, puesto que R es simétrica y toda matriz simétrica tiene esa propiedad (Horn and Johnson, 1985). Cabe aclarar que (Ecuación 3.4) es una versión modificada de la proyección de una matriz sobre el cono de matrices no-negativas definidas presentada en Liberzon and Tempo (2004), teniendo que para la matriz simétrica R se cumple $V^{-1} = V^T$, y usando \tilde{D} en vez de la matriz diagonal $D^+ = \text{diag}(\max(d_1, 0), \dots, \max(d_n, 0))$. Con esta proyección se puede definir, a partir del fitness

genérico (Ecuación 3.3), un primer fitness específico

$$f_1 \doteq f(\tilde{R}_1) \quad (3.6)$$

con

$$\tilde{R}_1 = V^{-1}\tilde{D}V, \quad (3.7)$$

y D , V obtenidas a partir de una matriz simétrica $R_1 = V^{-1}DV$ a evolucionar.

Una segunda forma de restringir el espacio de búsqueda del funcional genérico (Ecuación 3.3) se obtiene con base en la Proposición 3.2.1 enunciada a continuación, la cual es una forma equivalente alternativa y original al Problema CQLF.

Proposición 3.2.1. (*Ordóñez-Hurtado and Duarte-Mermoud, 2012*): *Sea el sistema conmutado (Ecuación 2.1) con $x \in \mathfrak{R}^n$, y A un conjunto de matrices Hurwitz en $\mathfrak{R}^{n \times n}$. Entonces, la existencia de una CQLF para A es equivalente a la existencia de una matriz triangular superior (inferior) $B \in \mathfrak{R}^{n \times n}$, tal que*

$$B^{-1}A_i^T B < 0, \quad \forall i \in \{1, 2, \dots, N\}. \quad (3.8)$$

Notar que B es una transformación de similaridad común para todo A_i , y la CQLF está dada por $P = BB^T$.

Demostración. Parte 1: Primero se demostrará la implicación $B \rightarrow$ CQLF. Supóngase que (Ecuación 3.8) se cumple. Por extensión (ver Lema 2.2.6) también se cumple que

$$B^{-1}A_i^T B + (B^{-1}A_i^T B)^T < 0, \quad \forall i \in \{1, \dots, N\},$$

y por lo tanto

$$A_i^T B B^T + B B^T A_i < 0, \quad \forall i \in \{1, \dots, N\}. \quad (3.9)$$

Definiendo $P = B B^T > 0$ se obtiene (Ecuación 2.4), y se cumplen así las condiciones para que P sea una CQLF.

Parte 2: Ahora se demostrará la implicación CQLF $\rightarrow B$. Supóngase que existe una CQLF P para A , y sin pérdida de generalidad, supóngase $P = P^T$. Dado que $P > 0$, por la factorización de Cholesky (Horn and Johnson, 1985) se tiene que $P = B B^T$ para alguna matriz triangular superior (inferior) B , y con ello se satisface (Ecuación 3.9). Finalmente, con un procedimiento inverso al presentado en la Parte 1 (multiplicando por la derecha el termino B^{-T} y usando el Lema 2.2.6) se llega a (Ecuación 3.8). \square

De esta forma, basados en la Proposición 3.2.1 se propone, a partir del fitness genérico (Ecuación 3.3), el segundo fitness específico

$$f_2 = f(\tilde{R}_2) \quad (3.10)$$

con

$$\tilde{R}_2 = R_2 R_2^T, \quad (3.11)$$

siendo R_2 una matriz triangular superior (inferior) en $\mathfrak{R}^{n \times n}$ a evolucionar.

Una vez definidas las funciones de fitness a utilizar, se procede a diseñar la forma en que las partículas representarán las diferentes matrices a evolucionar, es decir, R_1 y R_2 .

3.2.2. Representación de las partículas

Dado que se usarán dos fitness, se necesita entonces diseñar dos formas en las cuales las s partículas representarán a las matrices a evolucionar $R_{1,2}^{(i)}$ $i = 1, \dots, s$, (y consecuentemente $\tilde{R}_{1,2}^{(i)}$ por medio de (Ecuación 3.7) y (Ecuación 3.11)). Inicialmente es necesario redefinir la posición de las partículas como

$$\mathbf{x}_i = [C_j^{(i)}], \quad j = 1, 2, \dots, \lambda, \quad i = 1, 2, \dots, s. \quad (3.12)$$

Hasta este punto pareciera intuitivo relacionar cada componente j de las partículas con cada elemento de $R_1^{(i)}$ o $R_2^{(i)}$, pero esto no es muy beneficioso para el proceso de optimización debido a la naturaleza del problema CQLF: si una matriz arbitraria P es una CQLF, entonces αP también lo es para todo $\alpha > 0$ (ver ecuación (Ecuación 2.4)). Entonces, es necesario restringir nuevamente el espacio de búsqueda acotando la norma de los vectores $[C_j^{(i)}]$. Para esto es necesario definir un vector $M_i = [m_k^{(i)}]$, $k = 1, 2, \dots, n_p$, con

$$m_k^{(i)} = \begin{cases} \sin(C_1^{(i)}), & k = 1, \\ \sin(C_k^{(i)}) \prod_{h=1}^{k-1} \cos(C_h^{(i)}), & k = 2, 3, \dots, n_p - 1, \\ \prod_{h=1}^{2N-1} \cos(C_h^{(i)}), & k = n_p, \end{cases} \quad (3.13)$$

tal que n_p es el grado de libertad de $R_1^{(i)}$ o $R_2^{(i)}$ según el caso, imponiendo así que $\lambda = n_p - 1$. Ahora es necesario definir un par de funciones $g_{1,2} : \mathfrak{R}^{n_p} \rightarrow \mathfrak{R}^{n \times n}$ tal que $R_{1,2}^{(i)} = g_{1,2}(M_i)$ son las matrices que serán evolucionadas con PSO, a fin de corresponder con $f_{1,2} : \mathfrak{R}^{n \times n} \rightarrow \mathfrak{R}$. Sin embargo, tanto $R_1^{(i)}$ como $R_2^{(i)}$ tienen $n_p = \frac{n(n+1)}{2}$, dado que $R_1^{(i)}$ son matrices simétricas en $\mathfrak{R}^{n \times n}$ y $R_2^{(i)}$ son matrices triangulares superiores (inferiores) en $\mathfrak{R}^{n \times n}$. Entonces, las $R_1^{(i)}$ usadas con f_1 quedan definidas por

$$R_1^{(i)} = \begin{bmatrix} m_1^{(i)} & m_2^{(i)} & m_3^{(i)} & \cdots & m_n^{(i)} \\ m_2^{(i)} & m_{n+1}^{(i)} & m_{n+2}^{(i)} & \cdots & m_{2n-1}^{(i)} \\ m_3^{(i)} & m_{n+2}^{(i)} & m_{2n}^{(i)} & \cdots & m_{3n-3}^{(i)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m_n^{(i)} & m_{2n-1}^{(i)} & m_{3n-3}^{(i)} & \cdots & m_{n_p}^{(i)} \end{bmatrix}, \quad (3.14)$$

y sin pérdida de generalidad las $R_2^{(i)}$ usadas con f_2 quedan definidas por

$$R_2^{(i)} = \begin{bmatrix} m_1^{(i)} & m_2^{(i)} & m_3^{(i)} & \cdots & m_n^{(i)} \\ 0 & m_{n+1}^{(i)} & m_{n+2}^{(i)} & \cdots & m_{2n-1}^{(i)} \\ 0 & 0 & m_{2n}^{(i)} & \cdots & m_{3n-3}^{(i)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & m_{n_p}^{(i)} \end{bmatrix}. \quad (3.15)$$

Finalmente, dado que $\tilde{R}_{1,2}^{(i)}$, $R_{1,2}^{(i)}$, \mathbf{x}_i y M_i representan un mismo elemento, se tiene que $f_{1,2}(R_{1,2}^{(i)}) \doteq f_{1,2}(\mathbf{x}_i)$.

De esta forma, y después de haber definido la parte analítica del diseño, se pasa a la fase de implementación y análisis de desempeño.

3.2.3. Análisis de desempeño de la metodología propuesta

En esta sección se muestra la implementación de la metodología de cálculo de una CQLF basada en PSO, y los resultados de las pruebas realizadas para evaluar su desempeño. La implementación de dicha propuesta implicó, entre otros, el desarrollo de 2 programas principales en el software Matlab para definir los fitness f_1 y f_2 diseñados en la sección anterior. Como herramienta de software se utiliza el Toolbox PSO (un conjunto de archivos Matlab (.m) que implementan el algoritmo PSO estándar para la optimización de sistemas) (ver Apéndice A) desarrollado por Jagatpreet Singh (Singh, 2003). Los programas desarrollados requirieron de la definición de un archivo auxiliar en el cual se calcula la función de fitness correspondiente, y una modificación al archivo principal del Toolbox PSO (PSO.m) para definir la población inicial a conveniencia.

La configuración de PSOiw utilizada es la siguiente:

- Peso de inercia: decreciente linealmente con $w(0) = 0,9$, $w(iter_{\max}) = 0,4$,
- Coeficientes de aceleración: $c_1 = c_2 = 2$,
- Tamaño del enjambre: $s = 10 + 2\sqrt{\lambda}$,
- Dimensión de las partículas: $\lambda = \frac{n(n+1)}{2} - 1$, siendo n el orden de las matrices A_i .
- Máximo número de iteraciones: $iter_{\max} = 200$ (por defecto).
- Criterio de término: i) $f(\mathbf{x}_i) < 0$, y ii) $k > iter_{\max}$.
- Condiciones iniciales para el enjambre: aleatorias con distribución uniforme.

Como datos de prueba no se utilizan bases de datos, sino conjuntos de matrices que a *priori* se sabe que comparten una CQLF, como por ejemplo matrices que conmutan (Narendra and Balakrishnan, 1994), y matrices triangulares (Shorten and Narendra, 1998), entre otras. Finalmente, se usan condiciones iniciales para \mathbf{x}_i de tipo aleatorias (con distribución uniforme) y predefinidas (solucionando s ecuaciones de Lyapunov individuales).

A continuación se presenta un conjunto de resultados experimentales que son comparados con los resultados obtenidos con otras metodologías.

3.2.3.1. Análisis comparativo: tiempo de cálculo para la primera solución factible

En el análisis comparativo respecto del tiempo de cálculo, las metodologías NB y EB no se tienen en cuenta, debido a que son algoritmos determinísticos que emplean un tiempo fijo (iteraciones) para encontrar la solución. Además de esto, no tiene sentido usar la metodología NB en el caso general, dado que no siempre se satisfarán las condiciones para su aplicación. Por otra parte, la metodología EB, a pesar de poderse aplicar al caso general sin ningún problema, es una metodología que llega a ser analíticamente incapaz de hallar una CQLF a pesar de que ésta exista, lo que se refleja en una tasa de éxito bastante baja. Finalmente, ya que en Liberzon and Tempo (2004) se hace una comparación de gradiente respecto de la metodología LMI y se evidencian las ventajas de gradiente sobre LMI, en este documento se buscará comparar la metodología propuesta (basada en PSO) con respecto a la metodología de Liberzon y Tempo (Liberzon and Tempo, 2004) basada en la técnica del gradiente.

Como primer conjunto de pruebas comparativas, se valida cada metodología con grupos de matrices triangulares superiores (que se sabe comparten una CQLF por Shorten and Narendra (1998)) de órdenes 2, 3, 4 y 5. La Figura 3.1 muestra los resultados de estas pruebas.

De la Figura 3.1 se pueden extraer dos tipos de conclusiones: la primera proviene de la comparación de f_1 y f_2 usando PSO, y la segunda de la comparación entre la metodología basada en gradiente y la metodología basada en PSO. Respecto del primer análisis, la Figura 3.2 muestra una marcada mejora en el tiempo de cálculo de f_2 sobre f_1 , lo cual es respaldado por el tipo de espacio de búsqueda que se emplea. Mientras que f_1 hace una reparación (ver (Ecuación 3.4)) de las potenciales soluciones, proyectándolas sobre un espacio de búsqueda restringido a matrices definidas positivas, f_2 siempre consigue potenciales soluciones dentro de ese espacio, no teniendo que recurrir a ningún tipo de reparación. Otro aspecto importante de destacar es la tendencia lineal del consumo de tiempo empleado para calcular una primera solución factible respecto del número de matrices a analizar, con una pendiente más pronunciada para f_1 que para f_2 .

En cuanto a la comparación entre las metodologías, se destaca el hecho de que PSO con f_2 generalmente es mucho más eficiente que gradiente, no siendo así con f_1 , que solamente es más eficiente que gradiente a partir de un determinado número de matrices en adelante. Estos aspectos son claramente justificables, ya que gradiente muestra una tendencia aproximadamente exponencial en el consumo de tiempo hasta la primera solución factible en función del número de matrices analizadas, y con la mencionada tendencia lineal de PSO entonces llega un momento en que el tiempo consumido por gradiente supera al consumido por f_1 .

Un elemento importante de resaltar es que para cada realización de PSO se utilizó una misma configuración (mismo conjunto de parámetros), no así para el caso de gradiente, en donde la ejecución para cada nuevo número de matrices requirió de una exploración experimental para determinar los mejores valores de los parámetros r y α a emplear. Esto trae como consecuencia que para un número mayor de matrices se necesite un valor más pequeño de r y α , lo que en el fondo ralentiza el proceso de optimización y le da características exponenciales a la relación tiempo

3.2 Cálculo de una CQLF usando PSO

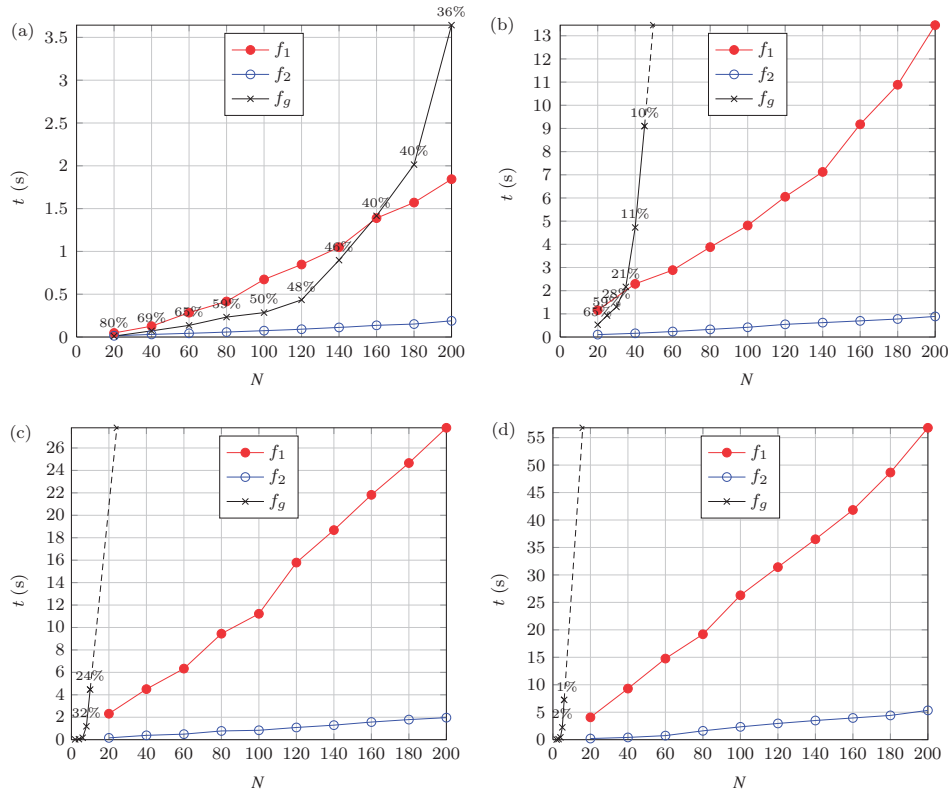


Figura 3.1.: Tiempo de cálculo promedio hasta la primera solución factible para conjuntos de matrices triangulares superiores. (a) Matrices de 2×2 , (b) matrices de 3×3 , (c) matrices de 4×4 , y (d) matrices de 5×5 . (Cada marca: promedio de 20 medidas; líneas punteadas: proyecciones debido a complejidad de obtener datos; porcentajes: tasas de éxito; sin porcentajes: tasas del 100%; condiciones iniciales para PSO: aleatorias) (Ordóñez-Hurtado and Duarte-Mermoud, 2012).

de cálculo versus número de matrices. No obstante, es claro que una misma configuración de PSO tiene una cota superior de efectividad, pero es expansible incrementando $iter_{max}$ o eventualmente s .

A pesar de que PSO muestra una ventaja marcada sobre gradiente al utilizar matrices triangulares, es interesante observar el comportamiento de la metodología propuesta sobre otro tipo de matrices (utilizando ahora solamente f_2 debido a las ventajas descritas respecto de f_1). Por ello, se ejecutan pruebas sobre otros conjuntos de matrices que se sabe a priori comparten una CQLF: matrices que conmutan (Narendra and Balakrishnan, 1994), matrices simultáneamente triangularizables (Ibeas and de la Sen, 2009), matrices negativas definidas (Shorten et al., 2007), matrices diagonales (conmutan y son negativas definidas), y un conjunto de matrices genéricas obtenidas a partir de una matriz simétrica arbitraria P positiva definida y buscando por exploración aleatoria matrices estables para las cuales P soluciona su ecuación de Lyapunov individual (Ecuación 2.4). La Figura 3.2 muestra los resultados de aplicar las dos metodologías sobre estos 4 tipos de matrices.

3.2 Cálculo de una CQLF usando PSO

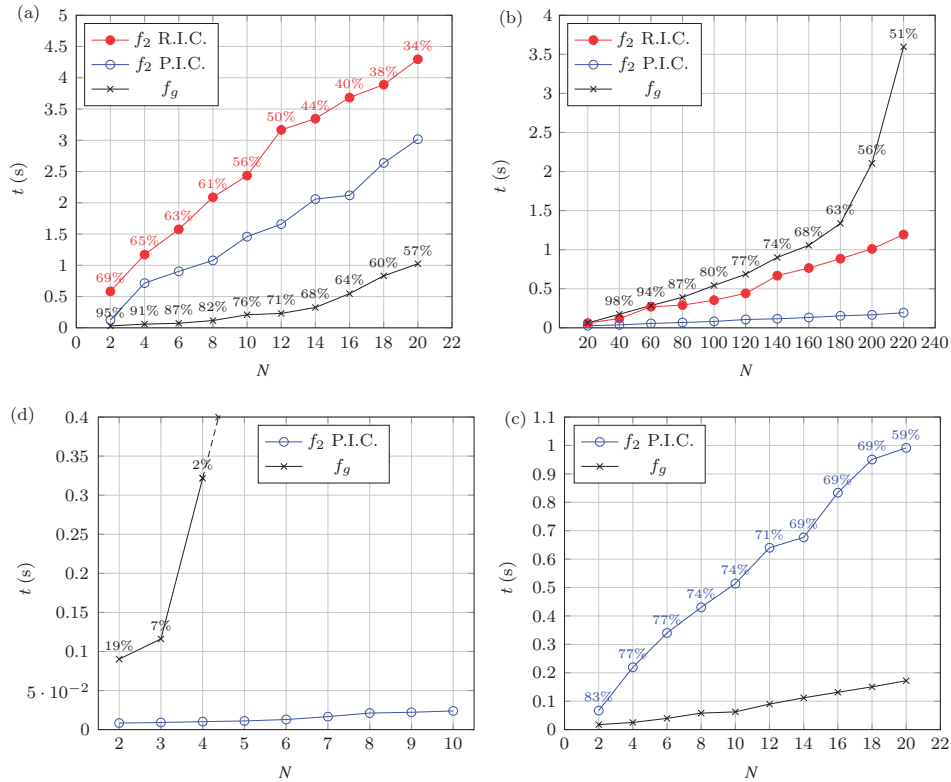


Figura 3.2.: Tiempo de cálculo promedio hasta la primera solución factible para otra clase de conjuntos de N matrices en $\mathbb{R}^{5 \times 5}$. (a) Matrices que conmutan, (b) matrices diagonales, (c) matrices comúnmente diagonalizables, y (d) matrices genéricas. (Cada marca: promedio de 20 medidas; líneas punteadas: proyecciones debido a complejidad de obtener datos; porcentajes: tasas de éxito; sin porcentajes: tasas del 100%; R.I.C.: condiciones iniciales aleatorias; P.I.C.: condiciones iniciales predefinidas) (Ordóñez-Hurtado and Duarte-Mermoud, 2012).

Al analizar la Figura 3.2, es interesante apreciar el hecho de que PSO no siempre tiene un desempeño superior a gradiente. De hecho, se encuentra que PSO tiene mejor desempeño que gradiente en el caso de matrices diagonales y triangularizables, no siendo así para el caso de matrices genéricas y matrices que conmutan.

Otro aspecto interesante de analizar es la tasa de efectividad (porcentaje de acierto al tratar de obtener soluciones factibles) de las metodologías. Si bien gradiente asegura que, dado un valor óptimo de r y α se logra obtener una CQLF en un determinado número de pasos, tales valores óptimos son desconocidos y por lo tanto cada configuración de gradiente tendrá una determinada tasa de efectividad. En las pruebas realizadas se observó que, a medida que gradiente necesita analizar un número mayor de matrices, se necesita una reconfiguración de sus parámetros para al menos mantener una misma tasa de efectividad. Sin embargo, con PSO esto generalmente no ocurre, y una misma configuración inicial permite mantener altas tasas de efectividad, aún en grandes conjuntos de matrices. A diferencia de gradiente, con PSO la convergencia no está asegurada analíticamente, pero a pesar de esto los experimentos muestran que es mucho más fácil obtener una adecuada sintonización de sus dos parámetros (a saber, el tamaño de población s y el número máximo de iteraciones $iter_{\max}$) para obtener soluciones factibles. Es importante aclarar

que si bien PSO tiene muchos más parámetros de sintonización, la elección de variables como c_1 , c_2 y $w(k)$ está respaldada por investigaciones previas (Del Valle et al., 2008) para su uso general, pero $iter_{max}$ y s son parámetros problema-dependientes.

Sin embargo, más allá de la conclusión parcial sobre la efectiva aplicabilidad de PSO al cálculo de una CQLF, es interesante analizar un aspecto mucho más importante: la robustez de las soluciones obtenidas por PSO (específicamente con f_2) respecto de las otras metodologías. Por ello, la siguiente sección se dedica a este asunto.

3.2.3.2. Análisis comparativo: Robustez de la mejor solución

La idea de este análisis es obtener una medida cuantitativa y cualitativa de la robustez para las mejores soluciones (CQLFs) obtenidas por varias metodologías. Para este tipo de análisis es necesario tener en cuenta que una comparación gráfica sólo es posible para conjuntos de matrices de segundo orden, debido al grado de libertad de las soluciones. Por tanto, la comparación para conjuntos de matrices de orden superior a 2 sólo es posible mediante la definición de un índice de desempeño. Tal índice se define aquí como el porcentaje de variación máxima que puede llegar a sufrir un elemento cualquiera de una matriz, de tal modo que una CQLF calculada previamente continúe siendo CQLF, a pesar de dicha variación.

Como experimento base se plantea calcular una CQLF para un mismo conjunto de 3 matrices, y luego evaluar la tolerancia de la mejor solución encontrada ante una posible variación paramétrica en alguno(s) de los parámetros de las matrices analizadas (matrices como las definidas en los ensayos de la Subsubsección 3.2.3.1, para las cuales se tiene el respaldo previo, analítico o experimental, de que comparten una CQLF). Las metodologías escogidas para hacer la comparación son: gradiente (f_g), LMI, Algoritmo EB, PSO (f_2), y como caso especial el Algoritmo NB, a pesar de que no es aplicable en forma general.

Caso Orden = 2: Análisis gráfico El hecho de que la matriz P asociada a una CQLF de segundo orden pueda ser supuesta, sin pérdida de generalidad, como una matriz simétrica, permite que las áreas de soluciones factibles para una matriz puedan ser representadas como un casquete elipsoide sobre una esfera de círculo unitario (también sin pérdida de generalidad). De esta forma, en la medida que dos casquetes asociados a dos sistemas (matrices) tengan intersección, se sigue que esas dos matrices comparten un área de CQLFs, y este hecho se puede extender directamente a un conjunto de N matrices.

La Figura 3.3 muestra un conjunto de resultados gráficos obtenidos al realizar la prueba anteriormente definida, para los cuales se utilizaron las siguientes matrices:

1. Matrices que conmutan

$$A \approx \begin{bmatrix} -5,3788 & 4,5974 \\ -0,7009 & -1,4067 \end{bmatrix}, B \approx \begin{bmatrix} -6,9024 & -2,4484 \\ 0,37327 & -9,0178 \end{bmatrix}, C \approx \begin{bmatrix} -0,7287 & -7,6940 \\ 1,1730 & -7,3760 \end{bmatrix},$$

2. Matrices triangulares

$$A = \begin{bmatrix} -3,74 & -10,06 \\ 0 & -13,71 \end{bmatrix}, B = \begin{bmatrix} -15,53 & 25,43 \\ 0 & -9,90 \end{bmatrix}, C = \begin{bmatrix} -5,37 & 10,98 \\ 0 & -21,21 \end{bmatrix},$$

3. Matrices negativas definidas

$$A = \begin{bmatrix} -9,26 & 5,65 \\ -6,88 & -0,073 \end{bmatrix}, B = \begin{bmatrix} -4,48 & 3,22 \\ -11,17 & -6,39 \end{bmatrix}, C = \begin{bmatrix} -10,83 & 1,61 \\ 1,05 & -11,76 \end{bmatrix},$$

4. Matrices genéricas

$$A = \begin{bmatrix} -9,83 & -5,56 \\ 4,85 & -4,21 \end{bmatrix}, B = \begin{bmatrix} -2,47 & -7,70 \\ 1,47 & -4,45 \end{bmatrix}, C = \begin{bmatrix} -8,69 & 1,16 \\ 8,84 & -5,27 \end{bmatrix}.$$

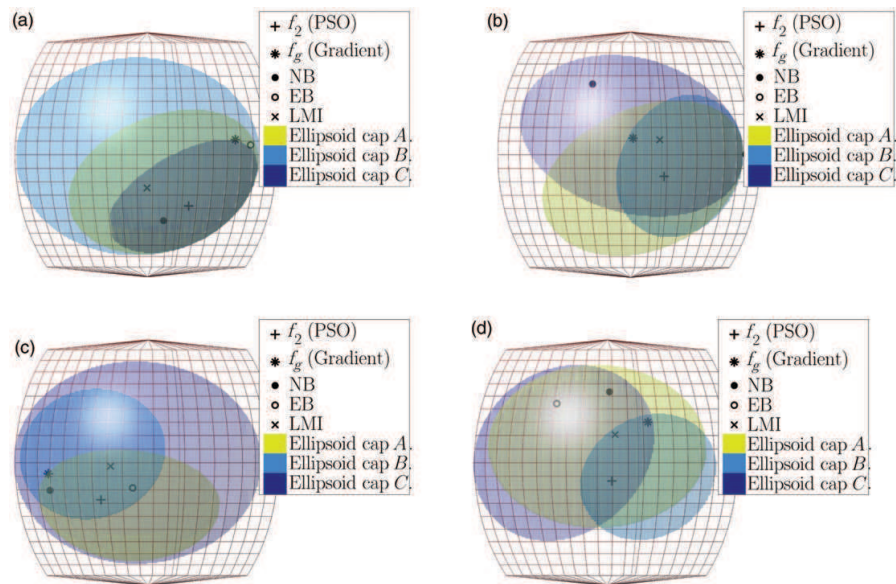


Figura 3.3.: Comparación gráfica para el cálculo de una CQLF en un conjunto de tres matrices de segundo orden que comparten una CQLF(Ordóñez-Hurtado and Duarte-Mermoud, 2012): a) matrices que conmutan, b) matrices triangulares, c) matrices negativas definidas, y d) matrices genéricas.

En la Figura 3.3, cada casquete elipsoide representa el espacio de soluciones factibles individuales a la ecuación de Lyapunov de cada sistema. Por lo tanto, de esta forma se puede verificar fácilmente, a través de la la intersección de áreas, que las matrices analizadas comparten una CQLF, y así mismo se puede delimitar el área de soluciones factibles para el problema CQLF. A pesar de

que la Figura 3.3 muestra el resultado de realizaciones específicas para un conjunto de matrices particulares, se puede hacer un acercamiento a la robustez de las soluciones obtenidas con las diferentes metodologías comparadas. De partida se advierte claramente que, en la medida que una solución esté más alejada de los bordes de la zona de existencia de CQLFs, ésta será más robusta a la variación paramétrica en las matrices analizadas, variaciones que hacen que los casquetes modifiquen su forma, orientación y ubicación. Entonces, al analizar la ubicación espacial de la mejor solución con PSO, se advierte que ésta se encuentra ubicada hacia lo que pudiera llamarse el centro de masa de la zona de CQLF (alejada de todos los bordes), a diferencia de lo que ocurre con las soluciones de las otras metodologías. Por lo tanto, las soluciones obtenidas con PSO son más robustas en comparación a las ofrecidas por las otras metodologías, respecto de variaciones paramétricas en los parámetros de las matrices analizadas (lo cual puede ser chequeado gráficamente), debido a que un pequeño cambio en alguna matriz puede reducir el área de CQLFs causando que una posible solución factible cercana a los bordes deje de serlo.

Caso Orden > 2: Análisis por índice de desempeño Para este caso de estudio se definirá un índice de desempeño para cuantificar el nivel de variación paramétrica que pueda sufrir un parámetro de las matrices a las cuales se les ha calculado una CQLF, de modo tal que la CQLF se mantenga como tal a pesar de dichas variaciones. Las dos máximas variaciones posibles, definidas como Var_+ y Var_- , son el máximo/mínimo porcentaje de cambio que puede ser sumado/restado al parámetro respecto de su magnitud original, es decir

$$Var_{\pm}(a_{i,j}, P) = \left\{ \text{Max}_{N\%} (a_{i,j} \pm N\% \text{abs}(a_{i,j})) \mid P \text{ siga siendo CQLF} \right\}. \quad (3.16)$$

Como ejemplo ilustrativo se escogió un grupo de 3 matrices estables en $\mathfrak{R}^{4 \times 4}$ con estructura genérica y escogidas aleatoriamente como

$$A = \begin{bmatrix} -1 & -3 & 2 & 0 \\ -1,8 & 0 & -0,9 & 2 \\ -1 & 6 & -5 & 1 \\ 4,8 & -3,2 & 2,9 & -5,1 \end{bmatrix}, B = \begin{bmatrix} -1,65 & -4 & 2,85 & -0,3 \\ -3 & 0,15 & -1,4 & 3,1 \\ -1,35 & 9,6 & -7,35 & 2,1 \\ 7,5 & -4,65 & 4,3 & -7,65 \end{bmatrix},$$

$$C = \begin{bmatrix} -2,2 & -6,5 & 3,8 & 0,5 \\ -4 & 0,2 & -1,9 & 4,2 \\ -1,8 & 13 & -9,8 & 3 \\ 10 & -6,2 & -5,8 & -10,2 \end{bmatrix}.$$

Usando f_g (metodología basada en gradiente) y f_2 (metodología basada en PSO), se consigue llegar a dos tipos de funciones de Lyapunov, que dependen de si la matriz P asociada es la primera solución factible (P^{first}) o la mejor solución factible (P^{best}). Las respectivas CQLFs obtenidas para el conjunto de matrices analizado fueron

$$P_G^{first} = \begin{bmatrix} 0,2919 & -0,1773 & 0,1365 & -0,1944 \\ -0,1773 & 0,6055 & -0,3181 & 0,2619 \\ 0,1365 & -0,3181 & 0,3613 & -0,2020 \\ -0,1944 & 0,2619 & -0,2020 & 0,3452 \end{bmatrix},$$

$$P_G^{best} = \begin{bmatrix} 0,3090 & -0,1976 & 0,1550 & -0,1936 \\ -0,1976 & 0,5909 & -0,3173 & 0,2601 \\ 0,1550 & -0,3173 & 0,3501 & -0,2153 \\ -0,1936 & 0,2601 & -0,2153 & 0,3430 \end{bmatrix},$$

$$P_{PSO}^{first} = \begin{bmatrix} 0,8673 & 0,3741 & 0,0819 & -0,1429 \\ 0,3741 & 0,2415 & -0,0277 & 0,1046 \\ 0,0819 & -0,0277 & 0,0706 & 0,0336 \\ -0,1429 & 0,1046 & 0,0336 & 0,0681 \end{bmatrix},$$

$$P_{PSO}^{best} = \begin{bmatrix} 0,6452 & -0,1355 & 0,1545 & -0,1854 \\ -0,1355 & 0,5835 & -0,0752 & 0,1015 \\ 0,1545 & -0,0752 & 0,1787 & 0,0915 \\ -0,1854 & 0,1015 & 0,0915 & 0,3322 \end{bmatrix}.$$

El análisis de la robustez de P_G^{first} , P_{PSO}^{first} , P_G^{best} y P_{PSO}^{best} frente a variaciones paramétricas de ciertos parámetros de A , B y C se desarrolla utilizando el índice de desempeño definido por (Ecuación 3.16), obteniéndose los resultados consignados en la Tabla 3.1 y la Tabla 3.2, donde los valores en negrita indican para cuál método se obtiene el máximo límite superior (Var_+) y el máximo límite inferior (Var_-) de variación paramétrica.

Cuadro 3.1.: Comparación de la robustez para la primera solución

	P_G^{first} (%)		P_{PSO}^{first} (%)	
	Var ₊	Var ₋	Var ₊	Var ₋
$a_{1,1}$	6	1733	2.5	31
$a_{2,1}$	55.6	4.28	4.83	27.22
$a_{1,2}$	99.7	3.33	1.9	1.2
$b_{1,1}$	0.42	1363	3.03	50.3
$b_{2,1}$	53.33	13	0.2	23.33
$b_{1,2}$	89.58	0.19	7.08	0.625
$c_{1,1}$	2.73	1318	1.68	55.09
$c_{2,1}$	55	1.23	8.25	25
$c_{1,2}$	87.69	1.07	8.2	0.41

Al analizar la Tabla 3.1 y la Tabla 3.2, se observan dos aspectos importantes. El primero de ellos es que, si se utiliza una CQLF calculada a partir de la primera solución factible (Tabla 3.1), la metodología basada en gradiente ofrece mejores resultados que la metodología basada en PSO,

Cuadro 3.2.: Comparación de la robustez para la mejor solución

	P_G^{best} (%)		P_{PSO}^{best} (%)	
	Var ₊	Var ₋	Var ₊	Var ₋
$a_{1,1}$	16.7	1935	70	1022
$a_{2,1}$	73.44	9.77	94.44	38.89
$a_{1,2}$	86.67	10.07	73.6	21.67
$b_{1,1}$	7.27	1575	51.51	799.39
$b_{2,1}$	69.33	3.27	70	25
$b_{1,2}$	83.33	2.9	63.46	12.5
$c_{1,1}$	9.11	795.45	50	792.18
$c_{2,1}$	69.2	3.98	70	22.5
$c_{1,2}$	80	3.37	60.92	11.54

en cuanto a robustez. Sin embargo, generalmente uno de los dos índices de desempeño es muy bajo al usar gradiente y por ello eventualmente PSO es mejor. Como segundo aspecto se tiene que al utilizar una CQLF calculada a partir de la mejor solución factible (Tabla 3.2), se tiene que la metodología basada en PSO permite obtener soluciones más robustas comparadas con gradiente. Cabe notar, además, que las veces en que PSO no mejora el índice respecto de gradiente es porque este último llegó a ofrecer índices muy por encima del obtenido con PSO.

3.2.4. Análisis de convergencia

Mientras que el funcional f_g (Ecuación 3.1) es convexo para cada matriz individual A_i , esto no se cumple para el caso en que todas las matrices A_i son analizadas simultáneamente. El espacio de soluciones factibles es la intersección de los espacios convexos asociados a las soluciones individuales (y por lo tanto convexo), pero el espacio de búsqueda es la unión de los espacios de búsqueda individuales, haciendo el funcional (Ecuación 3.3) no convexo al considerarse múltiples mínimos. Además, la complejidad del espacio de búsqueda se incrementa con el número de matrices a ser analizadas.

Ya que PSO tiene un probabilidad diferente de cero de quedarse estancado en un punto que podría incluso no ser mínimo local, la convergencia al espacio de soluciones factibles no puede ser asegurada (Jiang et al., 2007b). Sin embargo, esa probabilidad es muy pequeña, y con unas buenas características de explotación y exploración (obtenidas por medio de una adecuada elección de los parámetros de PSO) se pueden obtener altas tasas de éxito con tiempos de ejecución moderados.

A pesar de todo esto, existe una forma para asegurar la convergencia al espacio de soluciones factibles con probabilidad 1, pero esto es logrado en detrimento de los tiempos de cálculo demandados por la metodología. La idea es emplear una versión modificada de PSO propuesta por Van den Bergh (Van Den Bergh, 2002), llamada PSO con Convergencia Garantizada (GCPSO) (ver Subsección 2.5.3), que es usado para garantizar la convergencia local. Una vez se obtiene un mínimo local, se inicia otra búsqueda almacenando la mejor solución encontrada y reiniciando aleatoriamente el enjambre. Usando esta versión modificada de PSO se produce un incremento en los tiempos de cálculo debido a la incorporación de más ecuaciones de ajuste (con más parámetros

de sintonización de GCP SO) y el encadenamiento de búsquedas consecutivas, pero la convergencia es siempre asegurada.

3.3. Determinación de la no-existencia de una CQLF usando PSO

Para poder determinar la no-existencia de una CQLF usando PSO, es necesario el desarrollo de dos pasos primordiales:

1. Explorar la viabilidad de interpretar tal determinación como un problema de optimización, y
2. Diseñar una función de fitness a optimizar con PSO (dado que sea viable).

Los anteriores ítemes, que constituyen la metodología propuesta, son desarrollados a continuación.

3.3.1. Diseño de la metodología

Al analizar abstractamente la determinación de la no-existencia de una CQLF usando PSO se concluye que no se está buscando una solución óptima como tal, sino que se busca verificar el cruce de un umbral definido por el no cumplimiento de una condición necesaria (o necesaria y suficiente) para la existencia de una CQLF, o equivalentemente, el cumplimiento de una condición suficiente (o necesaria y suficiente) para la no-existencia de una CQLF. De este modo, al optimizar se busca probar la no-existencia de una CQLF por medio de un contra ejemplo de existencia. Esta conclusión sienta las bases para el diseño de una función de fitness adecuada, y con ello la construcción de la metodología basada en PSO para resolver el problema de la no-existencia de una CQLF.

El paso siguiente es encontrar una condición adecuada. Debido a la generalidad requerida para la solución propuesta, se requiere entonces de la evaluación de condiciones para el caso de N sistemas de orden n . Una primera condición a ser analizada es la presentada en Kamenetskiy and Pyatnitskiy (1987), la cual es la base del trabajo de King and Shorten (2006).

Teorema 3.3.1. (*King and Shorten, 2006*): Sea $A = \{A_1, \dots, A_N\}$ un conjunto de matrices Hurwitz de $n \times n$. Entonces A no tiene una CQLF si y sólo si existen matrices positivas semidefinidas X_1, \dots, X_N , no todas cero, tal que

$$\sum_{i=1}^N A_i X_i + X_i A_i^T = 0. \quad (3.17)$$

A pesar de que el Teorema 3.3.1 representa una condición necesaria y suficiente para la no-existencia de una CQLF en el caso general de N matrices de orden n , hay dos aspectos críticos desde el punto de vista de la optimización: (i) la precisión necesaria para satisfacer la igualdad (Ecuación 3.17), y (ii) la dimensionalidad de las potenciales soluciones. Respecto del primer aspecto, la precisión limitada para todos los formatos numéricos computacionales no permite asegurar

siempre que es posible encontrar experimentalmente, para todo conjunto arbitrario de matrices, un conjunto de matrices X_1, \dots, X_N a pesar de que éste último conjunto exista. En relación con el segundo aspecto, la dimensionalidad está dada por la cantidad de elementos diferentes que deben ser encontrados (cada entrada de las matrices a ser calculadas), es decir, $N * (n^2)$, o al menos $N * \frac{n(n+1)}{2}$ si las matrices X_i, \dots, X_N son simétricas, lo que implica una alta carga computacional.

Como segunda opción se analiza la condición presentada en Shorten and O’Cairbre (2001, Teorema 2.1, Corolario 2.1b), la cual se resume en el siguiente teorema.

Teorema 3.3.2. (*Shorten and O’Cairbre, 2001*): *Una condición necesaria para la existencia de una CQLF para el sistema (Ecuación 2.1) es que la matriz pencil*

$$\sum_{i=1}^N \alpha_i A_i + \beta_i A_i^{-1}, \quad \forall \alpha_i, \beta_i \geq 0, \quad \sum_{i=1}^N \alpha_i + \beta_i > 0, \quad (3.18)$$

sea Hurwitz.

El Teorema 3.3.2, que plantea la verificación de la propiedad Hurwitz para el politopo de matrices vértice $\{A_i^{\pm 1}\}_{i=1}^N$, representa una condición menos general que (Ecuación 3.17) ya que solamente es necesaria, y no necesaria y suficiente. Sin embargo, el Teorema 3.3.2 tiene un par de ventajas significativas: (i) la precisión numérica no es un aspecto crítico, porque se está buscando satisfacer una desigualdad en lugar de una igualdad (la condición Hurwitz se satisface si la parte real de todos los valores propios es menor que cero), y (ii) la dimensionalidad de las potenciales soluciones es solamente $2N$, que corresponde a los $2N$ diferentes coeficientes de cada combinación lineal a ser evaluada, lo cual es mucho menor que en el caso del Teorema 3.3.1.

Del anterior resultado, se plantea la siguiente proposición, la cual es la base de la metodología propuesta más adelante en esta Tesis.

Proposición 3.3.3. (*Duarte-Mermoud et al., 2012*): *Considérese el SLS (Ecuación 2.1) con $x \in \mathbb{R}^n$, $A_i \in \mathbb{R}^{n \times n}$ Hurwitz $\forall i \in \{1, 2, \dots, N\}$ y $\bar{A} = \{A_1^{-1}, \dots, A_N^{-1}\}$. Una condición suficiente para la no-existencia de una CQLF para (Ecuación 2.1) es que exista al menos una CLC no Hurwitz en $\{A, \bar{A}\}$.*

Demostración. Supóngase que existe una CQLF de la forma $V(x) = x^T P x$, $P > 0$ para (Ecuación 2.1), o equivalentemente en A . Del Lema 2.2.1, P también es una CQLF para \bar{A} . Considere una CLC arbitraria en $\{A, \bar{A}\}$ definida como

$$W = \text{CLC} [A_i, A_i^{-1}]_{i=1, \dots, N} = \sum_{i=1}^N (\alpha_i A_i + \alpha_{N+i} A_i^{-1}), \quad \alpha_j \geq 0, \quad \sum_{j=1}^{2N} \alpha_j = 1. \quad (3.19)$$

Entonces P es una CQLF para W , ya que se puede verificar que

$$PW + W^T P = -X, \quad (3.20)$$

con $X = \sum_{j=1}^{2N} [\alpha_j Q_j] > 0$ (Lema 2.2.5). Dado que existe P satisfaciendo (Ecuación 3.20), entonces toda W debe ser Hurwitz (Teorema 3.3.2). Por lo tanto, la existencia de una W no Hurwitz contradice la existencia de una CQLF, y con esto se completa la demostración. \square

Observación 3.3.4. La justificación de por qué la Proposición 3.3.3 es solamente una condición suficiente, está basada en la condición necesaria y suficiente para la no-existencia de una CQLF en parejas de matrices de tercer orden presentada en King and Shorten (2006, Teorema 5). Tal condición plantea la evaluación de la singularidad de $\tilde{W} = \text{CLC} [A_1^{\pm 1}, A_2^{\pm 1}, \sigma_\alpha [A_1, A_2]]$ (entre otras), y ya que $W \subset \tilde{W}$, entonces la evaluación de la no singularidad de W sólo es la verificación de una parte de la condición total. Se puede observar que la complejidad de una condición necesaria y suficiente para la no-existencia de una CQLF en parejas de sistemas incrementa con el orden de las matrices a ser analizadas, ya que en el caso de matrices de 2×2 , solamente es necesario evaluar la singularidad de $\sigma_\alpha [A_1^{\pm 1}, A_2]$. Esto muestra que para sistemas de orden mayor a 3, la condición de suficiencia estará más lejos de ser necesaria y suficiente. Sin embargo, un caso particular en el cual la Proposición 3.3.3 llega a ser una condición necesaria y suficiente para la no-existencia de una CQLF, es cuando se analiza una pareja de matrices de $n \times n$ cuya diferencia es de rango 1 King and Nathanson (2006), pero esto limita fuertemente la generalidad del método.

Aun cuando un método numérico basado en una condición suficiente para la no-existencia de una CQLF (Proposición 3.3.3) parece débil, es interesante analizar primero la siguiente situación antes de juzgar que es débil. Desde el punto de vista numérico, la determinación de la existencia de una CQLF ha sido abordada por autores como Liberzon and Tempo (2003, 2004) y Ordóñez-Hurtado and Duarte-Mermoud (2012), y sin embargo ninguno de estos trabajos entrega resultados concluyentes respecto del aseguramiento de la no-existencia de una CQLF. Esto queda más claro en el caso de encontrar una solución final que es no factible (no CQLF), porque esto puede ser la consecuencia de que tal CQLF no exista, o exista pero no haya podido ser calculada debido a una inadecuada elección de los parámetros de configuración o a una precisión numérica limitada. Por lo tanto, un método como el aquí propuesto es complementario a los métodos de cálculo LMI/L-T/O-D, y constituye un importante progreso en la búsqueda de una solución completa al problema CQLF desde el punto de vista experimental.

Una consecuencia directa de la Proposición 3.3.3 es la generalización del Lema 2.2.4 al caso de N matrices en $\mathfrak{R}^{n \times n}$, tal como se presenta a continuación.

Lema 3.3.5. (Duarte-Mermoud et al., 2012): *Si el SLS (Ecuación 2.1) tiene una CQLF, entonces cada $W = \text{CLC} [A_i, A_i^{-1}]$ es no-singular. Equivalentemente, ninguno de los productos de matrices $A_j W, \forall A_j \in A$ tiene autovalores reales negativos.*

Demostración. La condición necesaria de no singularidad de cada W se deriva directamente como Corolario de la Proposición 3.3.3. Ahora, con

$$\sigma_\alpha [A_j, W] = \left[\alpha A_j + (1 - \alpha) \text{CLC} [A_i, A_i^{-1}]_{i=1, \dots, N} \right] \subseteq W,$$

y de la misma forma para $\sigma_\alpha [A_j^{-1}, W] \subseteq W, \forall A_j \in A$, usando el Lema 2.2.4 se sigue que $A_j W$ y $A_j^{-1} W$ no tienen valores propios reales negativos. Finalmente, sin pérdida de generalidad, dado que $\sigma_\alpha [A_j, W] \subseteq \sigma_\alpha [A_j^{-1}, W]$, se sigue que $A_j W$ no tiene valores propios reales negativos. \square

Observación 3.3.6. Es interesante observar que del Lema 3.3.5 para el caso de $n = 2$, la no singularidad de W asegura la existencia de una CQLF por parejas en A , y esto es una consecuencia de la Proposición 2.2.3 y el hecho de que $\sigma_\alpha [A_i, A_j] \subset W$ y $\sigma_\alpha [A_i^{-1}, A_j] \subset W$ para todo $A_i, A_j \in A$. Sin embargo, aplicando el Teorema 2.2.2, es suficiente con chequear los valores propios de $A_i A_j$ y $A_i^{-1} A_j$ para asegurar la no singularidad de las dos matrices pencil. La utilidad del Lema 3.3.5 se destaca para $n > 2$, debido a que en esta situación la condición algebraica sobre los valores propios de $A_i^{\pm 1} A_j$ sólo es necesaria para la existencia de una CQLF en A (y por lo tanto para todo A), pero no suficiente. Por lo tanto, un hecho interesante para $A_1, A_2 \in \mathfrak{R}^{2 \times 2}$ y $n = 2$, es que la no singularidad de $\sigma_\alpha [A_1^{-1}, A_2]$ implica la no singularidad de $\sigma_\alpha [A_1, A_2^{-1}]$ y viceversa, pero esto no es generalmente cierto para el caso de $n > 2$.

Para la metodología aquí propuesta definimos

$$\Lambda = [\alpha_1, \dots, \alpha_{2N}] \quad (3.21)$$

como un conjunto arbitrario de coeficientes en la CLC (Ecuación 3.19), que como tal debe cumplir

$$\sum_{j=1}^{2N} \alpha_j = 1, \alpha_j \geq 0 \forall j, \quad (3.22)$$

con N el número de matrices a analizar. Entonces, esto sugiere que podemos definir una función de fitness $f(\Lambda)$ de la siguiente manera

$$f(\Lambda) = \min [\operatorname{Re} [\operatorname{eig}(-W(\Lambda))]], \quad (3.23)$$

donde $\operatorname{eig}(\cdot)$ denota la función que calcula los valores propios de una matriz. La interpretación de la anterior función de fitness es que, si se existe una $W(\Lambda)$ que sea no-Hurwitz, es decir, una $-W(\Lambda)$ que tenga al menos un valor propio con parte real negativa, entonces no existe una CQLF para A (ver Proposición 3.3.3). De esta forma, la función de fitness sólo entrega resultados útiles al obtenerse $f(\Lambda) < 0$, es decir, al atravesar un umbral definido en cero, lo cual permite deducir la no-existencia de una CQLF mediante un contra ejemplo de existencia.

Al analizar (Ecuación 3.19), se observa que su adecuación a PSO es casi directa, al relacionar Λ con las partículas a evolucionar. Definiendo ahora

$$\mathbf{x}_i = [C_d^{(i)}], \quad d = 1, 2, \dots, \lambda, \quad i = 1, 2, \dots, s, \quad (3.24)$$

entonces se desea encontrar una función $\nu(\mathbf{x}_i) = \Lambda_i$, tal que $\nu: \mathfrak{R}^M \rightarrow \mathfrak{R}^{2N}$ relacione $\{\alpha_j^{(i)}\}$ con $\{C_d^{(i)}\}$, cumpliendo además con (Ecuación 3.22). Por otra parte, ya que $\{\alpha_j^{(i)}\}$ y $\{C_d^{(i)}\}$ representan un mismo elemento, se tiene que $f(\mathbf{x}_i) \triangleq f(\Lambda_i)$. Entonces, la función de fitness a utilizar queda definida por

$$f(\mathbf{x}_i) = \min [\operatorname{Re} [\operatorname{eig}(-W(\nu(\mathbf{x}_i)))]]. \quad (3.25)$$

Ahora, la creación de una función de fitness específica queda determinada por la forma de definir la función ν . Para la definición de ν se proponen 2 alternativas que satisfagan (Ecuación 3.22). Una primera ν se obtiene por medio de los siguientes pasos:

1. Considerar $\lambda = 2N - 1$, y cada componente de la partícula como un ángulo, esto es

$$C_d^{(i)} \in [0, 2\pi], \forall d \in \{1, \dots, 2N - 1\}. \quad (3.26)$$

2. Calcular los elementos de Λ_i de la forma:

$$\alpha_j^{(i)} = \begin{cases} \left(\sin \left(C_1^{(i)} \right) \right)^2, & j = 1. \\ \left(\sin \left(C_j^{(i)} \right) \prod_{h=1}^{j-1} \cos \left(C_h^{(i)} \right) \right)^2, & j = 2, 3, \dots, 2N - 1. \\ \left(\prod_{h=1}^{2N-1} \cos \left(C_h^{(i)} \right) \right)^2, & j = 2N. \end{cases} \quad (3.27)$$

Mediante el uso de identidades trigonométricas simples y la adecuada factorización, se comprueba que con la representación (Ecuación 3.27) se satisface (Ecuación 3.22). Al conjunto formado por las ecuaciones (Ecuación 3.24) - (Ecuación 3.27) en adelante se le referirá como *Fitness 1* (f_1).

Una segunda opción para definir ν es la siguiente:

1. Considerar $\lambda = 2N - 1$, y cada componente de la partícula como un número en $[0, 1]$, es decir

$$C_d^{(i)} \in [0, 1], \forall d \in \{1, \dots, 2N - 1\}. \quad (3.28)$$

2. Calcular $\tilde{\alpha}_j^{(i)}$ de la forma:

$$\tilde{\alpha}_j^{(i)} = \begin{cases} C_j^{(i)} & \forall j \in 1, \dots, 2N - 1. \\ 1 - \text{mod} \left(\sum_{h=1}^{2N-1} \tilde{\alpha}_h^{(i)}, 1 \right) & j = 2N. \end{cases} \quad (3.29)$$

3. Definir $K = \sum_{k=1}^{2N} \tilde{\alpha}_k^{(i)}$, y calcular los elementos de Λ_i de la forma:

$$\alpha_j^{(i)} = \frac{\tilde{\alpha}_j^{(i)}}{K}, j = 1, 2, \dots, 2N. \quad (3.30)$$

Se puede verificar que la anterior representación satisface (Ecuación 3.22). El conjunto formado por (Ecuación 3.24) - (Ecuación 3.25) y (Ecuación 3.28) - (Ecuación 3.30) en adelante será referido como *Fitness 2* (f_2).

3.3.2. Resultados experimentales

En esta sección se muestran los resultados de las pruebas realizadas para evaluar el desempeño de la metodología propuesta, usando varios conjuntos de matrices. La implementación de dicha propuesta implicó el desarrollo de 2 programas principales en el software Matlab, que se basan las

definiciones en *Fitness 1* y *Fitness 2*. Como herramienta fundamental se hace uso del Toolbox PSO para Matlab (ver Apéndice A) desarrollado por Jagatpreet Singh (Singh, 2003) que implementa el algoritmo PSO para la optimización de sistemas. Los programas desarrollados requirieron de la definición de un archivo auxiliar, en el cual se define la función de fitness correspondiente, y una modificación al archivo principal del Toolbox PSO (PSO.m) para definir la población inicial a conveniencia. A fin de comparar el desempeño de la metodología propuesta, se hacen pruebas conjuntas y comparativas con las metodologías presentadas en Shorten and Narendra (2002), Cheng et al. (2003), y con las metodologías LMI, L-T y O-D para los casos en que sea aplicable.

La configuración usada para PSOiw es la siguiente:

- Peso de inercia: decreciente linealmente con $w(0) = 0,9$, $w(iter_{\max}) = 0,4$,
- Coeficientes de aceleración: $c_1 = c_2 = 2$,
- Tamaño del enjambre: $s = 4N$,
- Dimensión de las partículas: $\lambda = 2N - 1$,
- Máximo número de iteraciones: $iter_{\max} = 20 * N$ (por defecto).
- Criterio de término: i) $f(\mathbf{x}_i) < 0$, y ii) $k > iter_{\max}$.
- Condiciones iniciales para el enjambre: i) aleatorias con distribución uniforme, y (ii) predefinidas, con partículas que permiten iniciar el proceso de optimización analizando únicamente CLCs entre parejas de elementos (con igual ponderación).
- Progreso: se presenta el progreso del proceso de optimización para cada $0,1 * iter_{\max}$ iteraciones, donde $fGbest$ es el mejor valor de fitness encontrado.

A continuación se presenta un conjunto de resultados experimentales. Como ejemplos iniciales, se parte por analizar conjuntos de matrices estables que no comparten una CQLF (demostrado analíticamente), para luego pasar a un caso sin respaldo analítico a priori de la existencia/no-existencia de una CQLF.

Ejemplo 1 Tres matrices en $\mathfrak{R}^{2 \times 2}$ (no comparten CQLF).

Consideremos el caso de las tres matrices estables A_1 , A_2 y A_3 usadas en el Ejemplo 5.4 en Shorten and Narendra (2003):

$$A_1 = \begin{bmatrix} 0 & 5 \\ -30 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 5 \\ -26 & -1 \end{bmatrix}, A_3 = \begin{bmatrix} -6 & 27 \\ -150 & -1 \end{bmatrix}.$$

Usando las metodologías LMI, L-T y O-D, no es posible obtener información concluyente (no se puede encontrar una solución factible), planteando la situación de que una CQLF no existe, o existe pero se utilizó una inadecuada sintonización de los parámetros de las metodologías. Al aplicar la metodología de Cheng (Cheng et al., 2003) se obtienen los siguientes resultados:

$$\Theta_{A_1} = (0, 0,04), \quad \Theta_{A_2} = (0, 0,0476), \quad \Theta_{A_3} = (0, 0,0081),$$

$$\Theta = \Theta_{A_1} \cap \Theta_{A_2} \cap \Theta_{A_3} = (0, 0,0081),$$

$$\int_{t \in \Theta} (V(t) - L(t)) dt = 0, \quad (3.31)$$

y ya que el valor de la integral (Ecuación 3.31) es no positivo, se demuestra con esta metodología que la terna de sistemas no comparte una CQLF. Sin embargo, al aplicar la metodología propuesta se obtienen resultados no concluyentes al tener un valor positivo como salida final, independientemente de la función de fitness o inicialización usada, como se puede ver en Tabla 3.3. Es interesante resaltar que en este ejemplo es posible asegurar analíticamente que cada pareja comparte una CQLF (Teorema 2.2.2), y que esto no es suficiente para que los tres sistemas compartan una CQLF debido a que hay una intersección vacía de los tres casquetes elipsoidales asociados (Figura 3.4). Con esto queda claro que usando la metodología propuesta no es posible obtener resultados concluyentes, ya que lo que se hace es la verificación experimental de una condición necesaria para la no-existencia de una CQLF en parejas de sistemas (Lema 3.3.5) de una forma agregada en un mismo funcional.

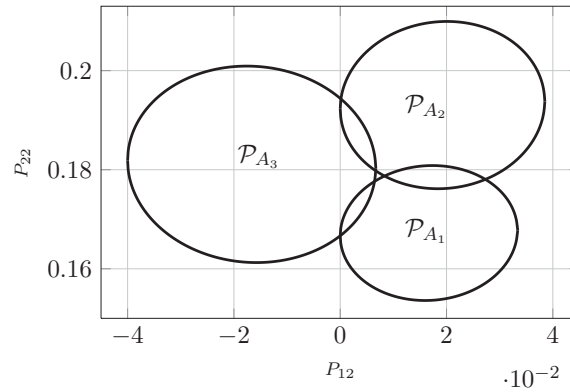


Figura 3.4.: Conjunto de CQLFs $\mathcal{P} = \begin{bmatrix} 1 & P_{12} \\ P_{12} & P_{22} \end{bmatrix}$ para A_1 , A_2 y A_3 (Shorten and Narendra, 2002) para el Sección 3.3.2.

Ejemplo 2 Cinco matrices en $\mathfrak{R}^{3 \times 3}$ (no comparten CQLF).

Ahora escogemos A_1 , A_2 , A_3 y A_4 como 4 matrices triangulares superiores, obtenidas aleatoriamente como

$$A_1 = \begin{bmatrix} -1,1764 & -2,2016 & -30,4614 \\ 0 & -28,6391 & 1,8565 \\ 0 & 0 & -0,8325 \end{bmatrix}, \quad A_2 = \begin{bmatrix} -21,2914 & 1,888 & 8,2560 \\ 0 & -11,2562 & 3,4612 \\ 0 & 0 & -6,3290 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} -1,3489 & 0,7536 & 8,8639 \\ 0 & -5,3062 & 6,0803 \\ 0 & 0 & -25,3543 \end{bmatrix}, \quad A_4 = \begin{bmatrix} -10,9267 & 5,1315 & -11,2256 \\ 0 & -10,4192 & -13,8875 \\ 0 & 0 & -0,8438 \end{bmatrix},$$

las cuales comparten una CQLF (Shorten and Narendra, 1998). Adicionalmente, escogemos una

Cuadro 3.3.: Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 1 , con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_1		Usandp f_2					
PPI Random		PPI Predefinida		PPI Random		PPI Predefinida	
Iter.	fGBest	Iter.	fGBest	Iter.	fGBest	Iter.	fGBest
20	0.000698	20	0.003882	20	0.075553	20	0.046997
40	0.000505	40	0.001485	40	0.046589	40	0.03228
60	0.000363	60	0.001169	60	0.013142	60	0.025
80	0.000360	80	0.001169	80	0.002047	80	0.012635
100	0.000278	100	0.001169	100	0.001566	100	0.006040
120	0.000169	120	0.001161	120	0.001015	120	0.003907
140	0.000132	140	0.001161	140	0.008699	140	0.003456
160	0.000111	160	0.001127	160	0.000162	160	0.002978
180	0.000111	180	0.001127	180	0.000150	180	0.001856
200	<u>0.000111</u>	200	<u>0.001127</u>	200	<u>0.000150</u>	200	<u>0.001856</u>
Autovalores finales: -0.00011 -0.00290		Autovalores finales: -0.01422 -0.00113		Autovalores finales: -0.00015 -0.02947		Autovalores finales: -0.01093 -0.00185	

matriz genérica Hurwitz A_5 , también obtenida aleatoriamente como

$$A_5 = \begin{bmatrix} -14,7578 & 6,3562 & -15,4226 \\ -9,9717 & 1,0197 & -4,2403 \\ 5,1944 & 5,5201 & -9,9615 \end{bmatrix}$$

tal que $A = \{A_i\}_{i=1,\dots,5}$ satisface las condiciones del Lema 2.2.4 sobre los valores propios de $A_i A_j$ y $A_i A_j^{-1}$ para todo $A_i, A_j \in A$. Al aplicar las metodologías LMI, L-T y O-D se obtienen resultados no concluyentes (la solución final es no factible), y ya que en este ejemplo no puede ser usado el método de Cheng (solamente desarrollado para sistemas de segundo orden), entonces queda saber si la metodología propuesta es capaz de entregar resultados concluyentes.

La Tabla 3.4 muestra los resultados de aplicar la metodología propuesta, los cuales confirman experimentalmente la no-existencia de una CQLF (se obtuvo una salida final negativa). Por búsqueda exhaustiva y métodos gráficos se encontró que $\sigma_\alpha [A_5^{\pm 1}, A_2^{\pm 1}]$, $\sigma_\alpha [A_5^{\pm 1}, A_3^{\pm 1}]$, $\sigma_\alpha [A_5^{\pm 1}, A_4^{\pm 1}]$ y $\sigma_\alpha [A_5^{\pm 1}, A_1]$ son Hurwitz, pero $\sigma_\alpha [A_5^{\pm 1}, A_1^{-1}]$ no son Hurwitz a pesar de que $A_5 A_1^{-1}$ y $A_5^{-1} A_1^{-1}$ no tienen valores propios reales negativos. Esto muestra analíticamente que el conjunto de sistemas no puede compartir una CQLF (Proposición 3.3.3).

Es interesante notar que los resultados particulares de este caso muestran que *Fitness 2* presenta convergencia más rápida comparado con *Fitness 1*. En este caso, la inicialización del enjambre juega un rol central dependiendo de cuál función de fitness es usada. Una inicialización predefinida se muestra beneficiosa para *Fitness 1*, mientras que una inicialización aleatoria muestra ser beneficiosa

Cuadro 3.4.: Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 2, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_1		Usando f_2	
PPI Random	PPI Predefinida	PPI Random	PPI Predefinida
Iter. fGBest	Iter. fGBest	Iter. fGBest	Iter. fGBest
20 0.60993	1 <u>-1.2552</u>	8 <u>-0.3206</u>	20 0.51751
40 0.60993			24 <u>-0.40567</u>
45 <u>-0.21958</u>			
Autovalores finales:	Autovalores finales:	Autovalores finales:	Autovalores finales:
-3.9829	-6.1156	0.3206	-4.6153
0.2196	1.2552	-4.7074	0.4057
-1.9150	0.0429	-4.3139	-1.4826

para *Fitness 2*.

Ejemplo 3 Cinco matrices en $\mathbb{R}^{4 \times 4}$ (no comparten CQLF).

En este caso el conjunto de matrices a analizar consiste en:

$$\begin{aligned}
 A_1 &= \begin{bmatrix} -10,68 & 20,44 & -1,34 & -4,68 \\ -15,72 & -6,74 & -7,78 & -6,97 \\ -5,47 & 12,22 & -12,89 & -1,58 \\ 2,75 & -6,74 & -4,29 & 0,74 \end{bmatrix}, \quad A_2 = \begin{bmatrix} -20,55 & -23,93 & -3,83 & 9,85 \\ 20,12 & 2,30 & -7,64 & 4,24 \\ -12,88 & 23,25 & 0,99 & -15,16 \\ -11,78 & -12,68 & -5,12 & -3,69 \end{bmatrix}, \\
 A_3 &= \begin{bmatrix} -9,05 & -0,68 & 0,35 & -3,65 \\ -22,45 & -9,43 & 6,64 & -7,88 \\ 1,63 & -11,13 & -13,10 & -7,73 \\ -8,86 & 2,51 & -4,56 & -7,59 \end{bmatrix}, \quad A_4 = \begin{bmatrix} -6,34 & -5,83 & 2,80 & -9,74 \\ 7,04 & -16,54 & -5,36 & -5,74 \\ -8,39 & 16,87 & 0,40 & 2,23 \\ 23,75 & 20,73 & -11,64 & 2,81 \end{bmatrix}, \\
 A_5 &= \begin{bmatrix} -17,92 & -7,37 & -6,62 & 2,41 \\ 0,43 & -13,17 & -0,29 & -22,42 \\ 7,09 & 13,22 & 4,98 & 6,64 \\ 4,80 & 8,95 & 6,10 & 2,47 \end{bmatrix},
 \end{aligned}$$

las cuales también fueron obtenidas de manera aleatoria.

Una vez más el método Cheng no puede ser usado aquí, y usando las metodologías LMI, L-T y O-D no es posible encontrar una CQLF para el sistema analizado. Sin embargo, la Tabla 3.5 muestra los resultados de aplicar la metodología propuesta, corroborando la no-existencia de una CQLF al obtener un valor negativo a la salida del proceso de optimización.

Se puede ver que en este caso los resultados no son sensibles a la función de fitness elegida o la inicialización empleada. Pero este hecho no es sorprendente, debido al cumplimiento de muchas condiciones analíticas para la no-existencia de una CQLF que pueden ser verificadas para este

Cuadro 3.5.: Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 3, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_1		Usando f_2	
PPI Random	PI Predefinida	PPI Random	PPI Predefinida
Iter. fGBest	Iter. fGBest	Iter. fGBest	Iter. fGBest
1 <u>-2.7342</u>	1 <u>-1.4437</u>	1 <u>-3.2387</u>	1 <u>-2.7042</u>
Autovalores finales:	Autovalores finales:	Autovalores finales:	Autovalores finales:
-9.4955	-4.3755 + 9.7904i	-4.7457 + 8.7296i	2.7042
-5.8707 + 7.5219i	-4.3755 - 9.7904i	-4.7457 - 8.7296i	-9.1585
-5.8707 - 7.5219i	-4.9084	-7.7287	-5.7044 + 6.5177i
2.7342	1.4437	3.2387	-5.7044 - 6.5177i

ejemplo, a saber:

- $A_1A_2^{-1}$, $A_2A_4^{-1}$ y $A_4^{-1}A_5$ tienen valores propios reales negativos,
- $A_2 + A_3^{-1}$ es inestable.

Con base en esta información, se sigue analíticamente (Proposición 3.3.3, Lema 3.3.5) que varias parejas del conjunto de matrices propuesto no pueden compartir una CQLF, y por tanto todo el conjunto tampoco puede compartir una.

Habiendo chequeado tres casos en los que existe una garantía analítica de la no-existencia de una CQLF, pasamos a explorar el caso de conjuntos de matrices para los cuales no hay información *a priori* acerca de la existencia de una CQLF, excepto que son matrices Hurwitz.

Ejemplo 4 Diez matrices en $\mathfrak{R}^{5 \times 5}$ (existencia de CQLF indeterminada).

A diferencia de los ejemplos previos, se escoge ahora un conjunto de 10 matrices estables en $\mathfrak{R}^{5 \times 5}$ para el cual no se sabe previamente si existe o no una CQLF. Estas matrices son generadas a partir de una matriz estable A y una perturbación estocástica del tipo $A_i = A + R_i$, $i \in \{1, 2, \dots, 10\}$, donde R_i son matrices de orden 5, cuyos elementos son números pseudo aleatorios con distribución normal, de media 0 y varianza 1. La matriz A es definida como

$$A = \begin{bmatrix} -5,6255 & -3,6453 & 4,0045 & -26,1274 & -5,2049 \\ -1,3169 & -2,5247 & 1,8567 & -17,9511 & -4,1578 \\ 3,5778 & -2,6112 & -0,7498 & -2,3677 & 5,6897 \\ 10,8137 & 6,6703 & -0,970 & -0,4835 & 11,4901 \\ -30806 & 5,2276 & -7,6581 & 14,9509 & -9,0751 \end{bmatrix},$$

matriz que también fue escogida aleatoriamente. Este ejemplo puede representar el caso donde variaciones paramétricas sobre una matriz nominal producen una familia de matrices, en la cual es desconocido si la familia comparte o no una CQLF. El uso de la metodología Cheng es de nuevo descartado, y las metodologías LMI, L-T y O-D no son capaces de encontrar una CQLF para el

conjunto de sistemas. La Tabla 3.6 muestra los resultados de aplicar la metodología propuesta al conjunto de matrices obtenidas mediante el procedimiento anteriormente explicado. Una vez más, la convergencia a un número negativo garantiza que el conjunto de matrices no comparte una CQLF. En este caso se observa que *Fitness 1* tiene una tendencia de mejor desempeño que *Fitness 2*, pero los beneficios de una inicialización predefinida o aleatoria no son muy significativos.

Cuadro 3.6.: Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 4, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_1		Usando f_2	
PPI Random	PPI Predefinida	PPI Random	PPI Predefinida
Iter. fGBest	Iter. fGBest	Iter. fGBest	Iter. fGBest
40 0.0018	1 0.0023	40 0.0736	40 0.0015
73 <u>-0.0032</u>	62 <u>-0.0066</u>	80 0.0588	80 0.0015
		120 0.0435	120 0.0015
		160 0.0206	160 0.0007
		200 0.0075	174 <u>-0.0005</u>
		225 <u>-0.0007</u>	
Autovalores finales: -0.2259 -0.1201 + 0.0573i -0.1201 - 0.0573i 0.0032 -0.0104	Autovalores finales: -0.2279 -0.1184 + 0.0631i -0.1184 - 0.0631i 0.0066 -0.0130	Autovalores finales: -0.5896 -0.2882 -0.1256 0.0007 -0.0140	Autovalores finales: -0.7579 -0.2447 -0.1144 0.0005 -0.0115

A pesar de que ambas funciones de fitness logran el objetivo de optimización, se evidencia que al usar un $iter_{max} < 100$ con el *Fitness 2* pudieran obtenerse resultados no concluyentes, dado que el valor final sería un número positivo, a pesar de que ahora se sabe que el conjunto analizado no comparte una CQLF.

Ejemplo 5 Veinte matrices en $\mathbb{R}^{6 \times 6}$ (existencia de CQLF indeterminada).

Finalmente, se utiliza un conjunto de 20 matrices Hurwitz en $\mathbb{R}^{6 \times 6}$, el cual consiste de 19 matrices triangulares superiores (que comparten una CQLF por Shorten and Narendra (1998)) y la matriz arbitraria Hurwitz en forma canónica *companion*

$$A_{20} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -0,0642 & -0,9107 & -4,5204 & -9,8177 & -10,3470 & -5,2047 \end{bmatrix}$$

Analíticamente se sabe que existe una CQLF para $\{A_i\}_{i=1}^{19}$, pero esto no es suficiente para asegurar la existencia de una CQLF para $A = \{A_i\}_{i=1}^{20}$. Al aplicar las metodologías LMI, L-T y O-D no es posible encontrar una CQLF para A, y descartando el uso del método Cheng estamos de nuevo en una situación de incertidumbre.

Cuadro 3.7.: Muestra de la aplicación de la metodología basada en PSO para Ejemplo 5, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_1		Usando f_2	
PPI Random	PPI Predefinida	PPI Random	PPI Predefinida
Iter. fGBest	Iter. fGBest	Iter. fGBest	Iter. fGBest
80 0.1882	80 0.0801	6 <u>-0.2297</u>	10 <u>-0.0005</u>
160 0.0504	160 0.0801		
240 0.0196	240 0.0552		
320 0.0181	320 0.0547		
400 0.0181	400 0.0546		
480 0.0181	480 0.0546		
560 0.0181	505 <u>-0.4997</u>		
640 0.0181			
720 0.0181			
800 <u>0.0181</u>			
Autovalores finales:	Autovalores finales:	Autovalores finales:	Autovalores finales:
-1.8715	-6.1034	0.2296	0.0005
-0.3121	-1.7480 + 1.7841i	-1.0298	-1.0048
-0.0181	-1.7480 - 1.7841i	-1.6637	-1.6142
-0.8687	0.4997	-3.2947 + 0.1456i	-3.2188 + 1.1629i
-1.1835	-0.2952 + 1.3676i	-3.2947 - 0.1456i	-3.2188 - 1.1629i
-0.8282	-0.2952 - 1.3676i	-4.8349	-4.6435

Los resultados experimentales presentados en la Tabla 3.7 muestran que el proceso de optimización usando *Fitness 1* con inicialización aleatoria, converge a un valor positivo en este caso particular. Una vez más esta información no es útil debido a que, por una parte, se trata de la verificación de una condición suficiente y, por otra parte, PSO no desarrolla una búsqueda completa, ni siquiera en sus versiones más recientes (Del Valle et al., 2008), y posteriores, ya que se trata de un problema de dimensión infinita. Sin embargo, al usar *Fitness 1* con una inicialización predefinida se tiene una convergencia a un número negativo. Por su parte, al usar *Fitness 2* se obtiene una convergencia a un valor negativo independientemente de la inicialización usada, y mucho más rápido que *Fitness 1*.

3.3.2.1. Ejemplos especiales

Los ejemplos aquí presentados son para un caso especial de la metodología propuesta, que fue desarrollado en principio para el análisis de parejas de sistemas. Este desarrollo se encuentra

reportado en Ordóñez-Hurtado and Duarte-Mermoud (2011b). Para este caso, se hace especial énfasis en el análisis de la no singularidad de las matrices *pencil*

$$\begin{aligned} M_1(\alpha) &= \sigma_\alpha [A_1, A_2], M_2(\alpha) = \sigma_\alpha [A_1^{-1}, A_2], \\ M_3(\alpha) &= \sigma_\alpha [A_1, A_2^{-1}], M_4(\alpha) = \sigma_\alpha [A_1^{-1}, A_2^{-1}], \end{aligned} \quad (3.32)$$

en vez de analizar la propiedad Hurwitz sobre Ecuación 3.19, y solamente teniendo en cuenta el funcional f_2 . Los ejemplos desarrollados para este caso se presentan a continuación.

Ejemplo 6.1 Dos matrices en $\mathbb{R}^{3 \times 3}$ (no CQLF).

Se escogen las siguientes matrices estables (aleatoriamente generadas):

$$A_1 = \begin{bmatrix} 8,5694 & 23,7247 & -39704 \\ -5,8897 & -19,3368 & -17,7087 \\ 5,56181 & -2,1832 & -18,7699 \end{bmatrix}, \quad A_2 = \begin{bmatrix} -5,3040 & -3,7383 & 0,8686 \\ -10,4602 & -7,8711 & -4,6389 \\ -5,2119 & 3,8493 & -6,5609 \end{bmatrix}.$$

Para las anteriores matrices se tiene que ninguno de los productos $A_1 A_2$, $A_1^{-1} A_2$, $A_1 A_2^{-1}$, $A_1^{-1} A_2^{-1}$ tiene valores propios reales negativos (ver 2.2.4). Por otra parte, usando tales matrices para generar las cuatro matrices *pencil* (Ecuación 3.32), se puede hacer un análisis gráfico de su singularidad, como se presenta en la Figura 3.5.

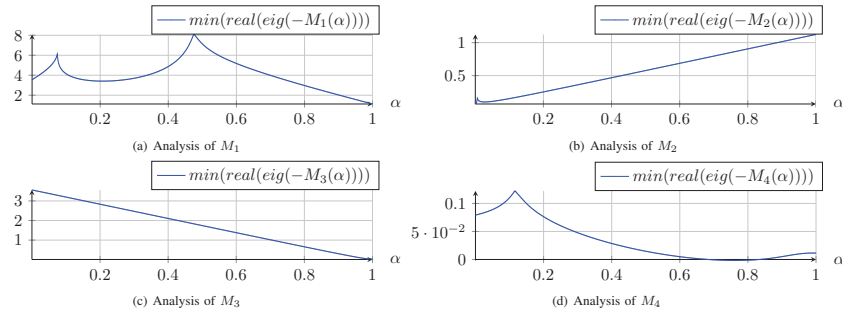


Figura 3.5.: Análisis de singularidad gráfico para las matrices *pencil* (Ecuación 3.32) con A_1, A_2 dadas para el Ejemplo 6.1, en función de $\alpha \in [0, 1]$ (Ordóñez-Hurtado and Duarte-Mermoud, 2011b).

En la Figura 3.5 se puede observar que M_4 es no-Hurwitz para $\alpha \approx [0,68, 0,83]$, a pesar de que M_1 es Hurwitz para todo α (véase Observación 3.3.6). Esto confirma que la singularidad de M_1 no implica la singularidad de M_4 , ni viceversa (igual para el caso de M_2 y M_3), excepto para el caso de sistemas de segundo orden. Por lo tanto, a pesar de que en principio se tiene que analíticamente no es posible deducir que A_1, A_2 no comparten una CQLF, el detallado análisis gráfico despeja las dudas. Ahora, aplicando la metodología propuesta, fue posible confirmar experimentalmente la no-existencia de una CQLF al obtener una salida final negativa, como se muestra en la Tabla 3.8. Para este caso particular, los resultados no son sensibles al tipo de inicialización usado.

Cuadro 3.8.: Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 6.1, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_2			
PPI Random		PPI Predefinida	
Iter.	fGBest	Iter.	fGBest
1	<u>-143.49</u>	1	<u>-72.618</u>
Autovalores finales:		Autovalores finales:	
143.4916		72.6284	
-145.3753		-70.3021	
-19.3309		-17.1643	
Tiempo consumido: 0.1237 s.		Tiempo consumido: 0.1204 s.	

Ejemplo 6.2 Dos matrices en $\mathfrak{R}^{10 \times 10}$ (no CQLF).

Para este caso, se generan aleatoriamente dos matrices Hurwitz $A_1 = [A_{1,1}, A_{1,2}]$ y $A_2 = [A_{2,1}, A_{2,2}]$ (por disposición de espacio), con

$$A_{1,1} = \begin{bmatrix} 5,09 & -0,66 & 12,90 & -6,27 & 8,70 \\ 7,99 & -10,78 & -1,22 & 6,55 & -3,03 \\ -8,31 & -5,26 & -13,77 & 9,82 & -9,67 \\ 7,31 & -11,88 & -12,81 & -3,34 & 9,62 \\ -7,70 & -5,61 & 5,53 & -15,08 & -14,67 \\ -19,44 & 6,21 & -6,47 & 5,29 & 10,42 \\ 4,44 & -9,44 & -0,45 & -6,07 & -2,53 \\ 6,21 & 18,29 & -0,59 & -10,57 & 0,40 \\ 2,79 & 7,75 & 8,06 & 1,26 & -6,30 \\ -4,26 & 4,73 & -7,53 & 14,70 & -25,50 \end{bmatrix},$$

$$A_{1,2} = \begin{bmatrix} 18,43 & -7,91 & 12,29 & -2,84 & -15,16 \\ 2,59 & 4,13 & -12,61 & -2,03 & -6,86 \\ -5,32 & -5,38 & 0,47 & 2,55 & 13,97 \\ -0,56 & -2,31 & 10,73 & 3,65 & -7,31 \\ -12,76 & -2,86 & 13,49 & -4,26 & 6,74 \\ -11,98 & -4,62 & 9,32 & 4,90 & -5,55 \\ 5,72 & -3,87 & -1,55 & 0,58 & 1,48 \\ 5,45 & 10,39 & -17,25 & 17,59 & 4,47 \\ 5,05 & 2,88 & -33,39 & -6,90 & -6,81 \\ -0,74 & -21,09 & 10,22 & 13,70 & -9,34 \end{bmatrix},$$

$$A_{2,1} = \begin{bmatrix} -19,59 & -3,57 & -1,88 & 1,71 & 15,55 \\ 10,76 & -2,89 & -6,64 & -15,90 & -30,19 \\ 3,97 & -16,90 & 4,27 & 10,33 & -21,21 \\ -6,41 & 12,71 & -3,79 & -15,86 & 7,16 \\ -1,35 & 11,86 & -0,33 & -1,94 & -17,75 \\ 2,94 & 10,42 & -0,30 & 1,26 & 4,24 \\ 7,46 & 2,83 & 8,09 & 12,83 & -5,60 \\ 10,08 & 11,95 & -4,25 & -4,36 & 14,34 \\ 16,66 & 0,27 & -4,21 & -0,77 & -10,77 \\ -2,22 & -5,57 & 0,27 & 7,65 & 6,61 \end{bmatrix},$$

$$A_{2,2} = \begin{bmatrix} 2,50 & 11,11 & -6,92 & -8,80 & 5,78 \\ -3,80 & -3,73 & -15,35 & 14,09 & 8,18 \\ 6,85 & -7,85 & -6,58 & 11,58 & -0,24 \\ 0,86 & -16,90 & -6,96 & -10,15 & 1,14 \\ 5,21 & -21,96 & -18,02 & -15,64 & -9,00 \\ -14,65 & 11,06 & 10,05 & 7,43 & -2,62 \\ -9,52 & -10,78 & 9,38 & 20,24 & -1,99 \\ 6,60 & -6,49 & -9,00 & 1,76 & 28,66 \\ -7,89 & -16,14 & -8,59 & -12,91 & -5,27 \\ -0,19 & -2,97 & -11,88 & 1,12 & -7,03 \end{bmatrix},$$

para las cuales ningún producto $A_1^{\pm 1} A_2^{\pm 1}$ contiene valores propios reales negativos (ver Lema 2.2.4). Similar al Ejemplo 6.1, se generan las cuatro matrices *pencil* (Ecuación 3.32), y su análisis gráfico de singularidad se presenta en la Figura 3.6, concluyendo que M_2 es no-Hurwitz para $\alpha \approx [0,004, 0,068]$.

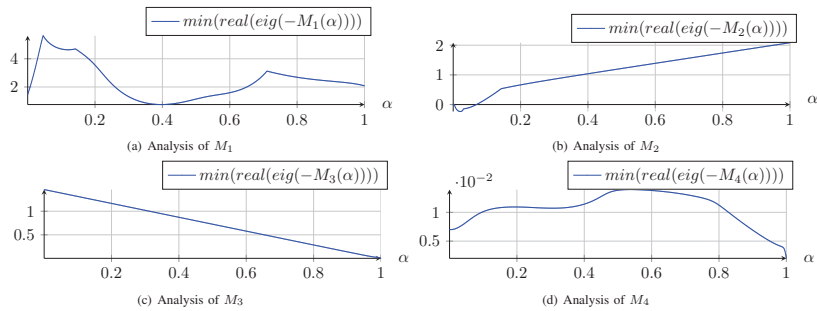


Figura 3.6.: Análisis de singularidad gráfico para las matrices pencil (Ecuación 3.32) con A_1, A_2 dadas para el Ejemplo 6.2, en función de $\alpha \in [0, 1]$ (Ordóñez-Hurtado and Duarte-Mermoud, 2011b).

Cuadro 3.9.: Muestra de la aplicación de la metodología basada en PSO para el Ejemplo 6.2, con diferente inicialización para las posiciones de las partículas (PPI). Las salidas finales están subrayadas.

Usando f_2			
PPI Random		PPI Predefined	
Iter.	fGBest	Iter.	fGBest
8	0.1209	8	0.1303
16	0.1209	16	0.1303
17	<u>-0.24051</u>	24	0.010316
		27	<u>-0.019013</u>
Autovalores finales:		Autovalores finales:	
-0.2084 + 1.0684i		-0.2136 + 1.3766i	
-0.2084 - 1.0684i		-0.2136 - 1.3766i	
-1.4787		-0.3314 + 0.9016i	
0.2485		-0.3314 - 0.9016i	
-0.2857 + 0.6877i		-1.3644	
-0.2857 - 0.6877i		-0.9724 + 0.2385i	
-0.2892 + 0.2248i		-0.9724 - 0.2385i	
-0.2892 - 0.2248i		0.0190	
-0.7642		-0.4498	
-0.9628		-0.3515	
Tiempo consumido: 0.1613 s.		Tiempo consumido: 0.1956 s.	

Con base en esta información, puede ser concluido que $A = \{A_1, A_2\}$ no comparten una CQLF. La Tabla 3.9 muestra el resultado de aplicar la metodología propuesta para este ejemplo, corroborando experimentalmente la no existencia de una CQLF, al obtener un valor negativo a la salida del proceso de optimización. Puede observarse además que los resultados son poco sensibles al tipo de inicialización utilizado.

4. PSO en el diseño de leyes de ajuste paramétrico

4.1. Introducción

Gran variedad de sistemas adaptables lineales/no-lineales en tiempo continuo/discreto pueden ser representados por modelos de error, facilitando así su análisis. La solución para un determinado modelo de error constituye una estrategia universal, que se aplica a los diversos sistemas que se puedan representar mediante ese modelo de error. En este capítulo se presenta una metodología para el ajuste de parámetros de un sistema adaptable discreto representado por modelos de error Tipo 1, 2 y 3, la cual está basada en PSO y que permite ser usada en aplicaciones en línea.

Dado que el problema de ajuste paramétrico en sistemas adaptables puede asimilarse como un problema de optimización, se aprovecha este hecho para tratar de hacer el proceso de optimización utilizando PSO en vez de gradiente o mínimos cuadrados, que son las técnicas tradicionalmente más usadas. Sin embargo, cabe aclarar que debido a la naturaleza discreta del algoritmo estándar de PSO, en esta Tesis Doctoral solamente se abarca la aplicación de PSO a sistemas adaptables de tiempo discreto.

En este orden de ideas, el diseño de la nueva metodología consiste en dos pasos principales:

1. Establecer la forma en la cual PSO será usado en tiempo real, y
2. Diseñar las funciones de fitness adecuadas.

En las siguientes secciones se abordan más detalladamente estos dos pasos.

Parte del contenido del presente capítulo ha sido avalado por la comunidad científica, debidamente reportado en Ordóñez-Hurtado and Duarte-Mermoud (2011a).

Consideraciones generales En adelante, se denotarán indistintamente las funciones $h(t) = [h_1(t), \dots, h_M(t)]$ vectorial y $h_i(t)$ escalar arbitrarias como

$$h(t) \triangleq h_t, \quad h_i(t) \triangleq h_t^{(i)}.$$

4.2. Usando PSO en tiempo real

Aunque existen aplicaciones on-line de PSO, generalmente esta técnica no es usada en tiempo real. La mayoría de sus aplicaciones están orientadas a optimización fuera de línea, por lo cual el primer

reto de la aplicación de PSO al diseño de leyes de ajuste en sistemas adaptables es solucionar este aspecto.

De forma natural puede decirse que, en el caso de sistemas dinámicos de tiempo discreto, la planta entre instantes de muestreo es invariante. Por ello, se decide estudiar la posibilidad de desarrollar procesos de optimización off-line entre instantes de muestreo consecutivos, y encadenar los procesos mediante la reutilización de la mejor solución encontrada en el proceso de optimización inmediatamente anterior. Para comprender esto en más detalle, se procede al análisis gráfico del proceso de optimización tal cual se presenta en la Figura 4.1.

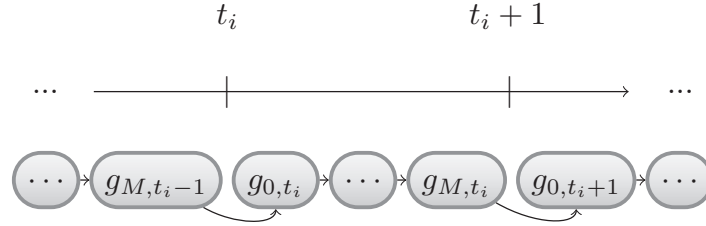


Figura 4.1.: Representación gráfica del proceso de optimización.

En la (Figura 4.1) se tiene la representación gráfica del proceso de optimización planteado para obtener una ley de ajuste usando PSO: $t_i, t_{i+1}, t_{i+2} \dots$ son instantes de muestreo arbitrarios del sistema adaptable bajo análisis, $g(k, t)$ es el óptimo entregado por PSO para una iteración k y un instante de muestreo t dados, y $M = iter_{\max}$ es el número máximo de iteraciones (criterio de término). De esta forma, queda claro que el proceso de optimización desarrollado por PSO es un encadenamiento de procesos de optimización individuales realizados en medio de instantes de muestreo consecutivos, donde la mejor solución final (remarcada en rojo) es reciclada para el nuevo proceso de optimización que inicia justo después de adquirir los nuevos datos del siguiente instante de muestreo. Según la representación anterior, la ley de ajuste para el sistema adaptable queda dada por

$$\hat{\theta}(t + 1) = \tilde{g}_M(t) = g(iter_{\max}, t). \tag{4.1}$$

Sin embargo, desde el punto de vista de optimización es importante poder asegurar que la técnica utilizada sea capaz de encontrar un óptimo en cada proceso de optimización independiente. Dado que $t_i - t_{i-1} = T \ll \infty$, se tiene entonces un tiempo limitado para cada proceso de optimización independiente. No obstante, el encadenamiento de los procesos individuales y el reciclaje de la mejor solución final individual constituyen un nuevo proceso de optimización con salida $\tilde{g}_M(t)$, tal que representa un proceso de optimización continuado al cual, en cada instante de muestreo, se le reinicia tanto la posición como la velocidad de las partículas. Bajo este panorama se tienen dos posibles soluciones:

1. Utilizar GCPSO: con un número adecuado de iteraciones, lograr la convergencia a la mejor aproximación de un óptimo local en cada proceso independiente. Esto, acompañado de una función fitness estrictamente convexa, hace que en cada intervalo $t_i - t_{i-1}$ pueda ser posible encontrar el óptimo global (en funciones estrictamente convexas, el único óptimo local es el óptimo global). Además, dado que $\tilde{g}_M(t)$ es un proceso continuado con $t \rightarrow \infty$, entonces se

satisfacen las condiciones del Teorema 2.5.8, y con ello se asegura la convergencia al óptimo cuando $t \rightarrow \infty$.

2. Utilizar PSO: a pesar de que PSO no es un optimizador local, el hecho de reiniciar cada proceso de optimización off-line conservando la mejor solución anterior, hace que se satisfagan las condiciones del Teorema 2.5.9, y por ende, esto hace que PSO sea un optimizador global. No obstante, no se puede asegurar que el óptimo pueda ser alcanzado entre cada intervalo $t_i - t_{i-1}$, pues la convergencia sólo se da para $t \rightarrow \infty$.

Sin embargo, las dos soluciones anteriores imponen ciertas limitaciones:

- El tiempo de ejecución del algoritmo PSO/GCPSO: los procesos individuales tienen un tiempo finito de ejecución impuesto por el tiempo de muestreo. Esto se puede solventar al escoger como criterio de término principal una tolerancia mínima aceptable, y como criterio de término secundario un número máximo de iteraciones. Con esto se logra alivianar la carga computacional, en situaciones en que el óptimo se alcanza antes de que el contador de iteraciones llegue a su fin.
- La configuración adecuada de los parámetros de PSO/GCPSO: una mala configuración lleva a un desempeño pobre. Para este caso las consideraciones a tener en cuenta son más del tipo práctico, basadas en resultados de convergencia hechos por otros autores (ver Subsección 2.5.1).

Es interesante resaltar el hecho que bien puede usarse el algoritmo PSO estándar para generar leyes de adaptación del tipo (Ecuación 4.1), a pesar de no ser un optimizador global. Esto se puede justificar por medio del Teorema 2.5.9, y teniendo que PSO estándar con reiniciación satisface las condiciones requeridas por dicho Teorema. Sin embargo, al utilizar PSO estándar, la convergencia local no está asegurada para cada proceso de optimización individual, y esto pudiera desmejorar los resultados prácticos, en la medida que existe la posibilidad de que haya convergencia prematura en cada proceso independiente debido a un estancamiento (no se satisface la condición **H2** (2.5.3)). Si bien lo anteriormente mencionado puede parecer una desventaja, cabe aclarar que la carga computacional exigida por GCPSO es mayor que la de PSO estándar, y entonces existe también la posibilidad de que éste último converja más rápido.

Una vez definida la forma en que PSO/GCPSO serán utilizados on-line, se pasa al diseño de las funciones de fitness adecuadas.

4.3. Diseño de las funciones de fitness

El esquema de implementación de PSO/GCPSO en línea permite utilizar potenciales funciones de fitness que no sean estrictamente convexas, puesto que al reiniciar los procesos de optimización off-line manteniendo la mejor solución lo que se hace es construir un optimizador MPSO (optimizador global). No obstante, queda claro que, al utilizar funciones de fitness estrictamente convexas se puede obtener un mejor desempeño, en la medida que hay más posibilidad de encontrar un único óptimo global entre instantes de muestreo.

No obstante, se explorarán diversas funciones fitness a fin de obtener los mejores resultados.

4.3.1. Modelo de Error 1

El método del gradiente aparece de forma natural en los sistemas adaptables, al aplicar el método directo de Lyapunov para el análisis de estabilidad de un sistema dado. Sin embargo, este método de gradiente surge a partir de la minimización del funcional

$$J = \frac{1}{2}e_t^2, \quad (4.2)$$

que para el Modelo del Error 1 (Goodwin et al., 1980; Duarte-Mermoud and Narendra, 1996) equivale a

$$J(\hat{\theta}_t) = \frac{1}{2}(\hat{\theta}_t^T \omega_t - y_t)^2 \quad (4.3)$$

La minimización de (Ecuación 4.3) se logra por medio de la siguiente ecuación iterativa

$$\hat{\theta}_{t+1}^T = \hat{\theta}_t^T - \gamma \frac{\partial J_1(\hat{\theta}_t)}{\partial \hat{\theta}_t},$$

donde $\partial J / \partial \hat{\theta}_t$ es el gradiente de (Ecuación 4.3) calculado como

$$\frac{\partial J(\hat{\theta}_t)}{\partial \hat{\theta}_{t-1}} = e_t \omega_t,$$

y γ es una ganancia positiva que representa la ganancia de adaptación. Se observa entonces como la tradicional ley de ajuste por medio de gradiente también puede ser obtenida mediante la interpretación del problema como uno de optimización.

Ahora, para realizar la minimización de (Ecuación 4.3) usando PSO en lugar de gradiente, se deben establecer ciertos elementos concernientes a la adaptación del problema a la naturaleza del algoritmo PSO. Como primer paso, se busca que en el intervalo entre dos instantes de muestreo consecutivos exista la mayor probabilidad de encontrar el óptimo global. Para ello, la idea es dar prioridad a funciones de fitness convexas, para lo cual es necesario analizar más detenidamente la Ecuación 4.3. Esto requiere del análisis de la primera y segunda derivada (Hessiano) de (Ecuación 4.3) respecto de $\hat{\theta}(t)$, calculados de la forma

$$\frac{\partial J}{\partial \hat{\theta}_t} = e_t \omega_t, \quad (4.4)$$

$$H = \frac{\partial^2 J}{\partial \hat{\theta}_t^{(i)} \partial \hat{\theta}_t^{(j)}} = \omega_t \omega_t^T \geq 0, \quad i, j = 1, 2, \dots, N. \quad (4.5)$$

El hecho de que $\omega(t) \omega^T(t)$ sea una matriz simétrica cuyos valores propios están dados por $\{0^{(1)}, \dots, 0^{(n-1)}, (\omega_t^{(1)})^2 + \dots + (\omega_t^{(n)})^2\}$, hace que (Ecuación 4.5) sea semipositiva definida para to-

do $\{\hat{\theta}(t), \omega(t)\}$, y con ello J es convexa pero no estrictamente. Sin embargo, dado que los múltiples mínimos (debido a la convexidad no estricta) se dan para $e_t \omega_t = 0$, es decir, para todo e_t ortogonal a ω_t , y dado que $e_t \in \mathfrak{R}^1$ entonces el único valor posible es $e_t = 0$. No obstante, dado que $e_t = \phi_t^T \omega_t$, el único ϕ_t que siempre hace $e_t = 0$ es $\phi_t = 0$ cuando ω_t es suficientemente rico en información. Nótese entonces la similitud de la condición sobre $\omega(t)$ con el concepto de excitación persistente, pero que en este caso parece ser menos restrictivo ya que una señal que tiende a cero puede tener la suficiente información para minimizar (Ecuación 4.3) usando PSO, pero no así para el caso de usar gradiente. Esta condición será llamada en el desarrollo de la presente metodología como *excitación instantáneamente persistente* (EIP).

Ahora, al optimizar (Ecuación 4.3) con GCPSO entre los instantes consecutivos de muestreo $\{t, t+1\}$, se asegura por el Teorema 2.5.8 que un mínimo local con $e(t) = 0$ es alcanzado. Además, cumpliendo la condición EIP para $\omega^T(t) \omega(t)$, entonces se consigue el punto de equilibrio $e(t) = 0$, $\phi(t) = 0$. Sin embargo, la velocidad de convergencia dependerá de aspectos tales como los relativos a la sintonización de los parámetros del algoritmo, y a las limitaciones físicas de la ejecución del algoritmo entre tiempos de muestreo.

A fin de formalizar lo anteriormente descrito, se plantea entonces el Teorema 4.3.1. Para ello, y con el fin de evitar confusiones por las dos escalas de tiempo manejadas, llámese Θ a la variable de optimización entre los tiempos $\{t, t+1\}$, para el cual todos los valores en t son conocidos.

Teorema 4.3.1. : Sea el Modelo de Error 1 en tiempo discreto definido por (Ecuación 2.12)

$$e_t = \hat{\theta}_t^T \omega_t - y_t, \quad \hat{\theta}_t^T, \omega_t \in \mathfrak{R}^n, \quad y_t \in \mathfrak{R}^n,$$

y sea el funcional del error cuadrático

$$J_1 = \frac{(\Theta^T \omega_t - y_t)^2}{2}. \tag{4.6}$$

siendo Θ la variable a ser optimizada todo intervalo $\{t_i, t_i+1\}$, $i = 0, 1, \dots$. Sea además el algoritmo de optimización GCPSO usado repetidamente para optimizar (Ecuación 4.6) en medio de todos los instantes de muestreo consecutivos (enfoque MPSO), con una configuración convergente de sus parámetros, y entregando como salida

$$\hat{\theta}(t+1) = \tilde{g}_M(t) = g(\text{iter}_{max}, t).$$

Entonces,

$$P \left[\lim_{t \rightarrow \infty} e(t) = 0 \right] = 1.$$

Además, si $\omega(t)$ es EIP, se consigue que

$$P \left[\lim_{t \rightarrow \infty} \hat{\theta}(t) - \theta^* = 0 \right] = 1.$$

Demostración. Al minimizar el funcional (Ecuación 4.6) con GCPSO entre todo intervalo de muestreo $\{t_i, t_{i+1}\}$, $i = 0, 1, \dots$, la probabilidad de encontrar el mínimo local (global) es diferente

de cero (Teorema (2.5.8)). Además, concatenando los procesos de optimización y manteniendo siempre la mejor solución mientras $t \rightarrow \infty$, se sigue que la probabilidad de lograr el mínimo local (global) es 1 (Teorema (2.5.9)). Dado que el mínimo de (Ecuación 4.6) se encuentra en

$$\mathbf{0} = \underbrace{(\Theta^T \omega_t - y_t)}_{\tilde{e}_t} \omega_t = \omega_t \omega_t^T \underbrace{(\Theta - \theta^*)}_{\tilde{\phi}_t} = \omega_t \omega_t^T \tilde{\phi}_t \quad (4.7)$$

entonces, es claro el mínimo se alcanza para $\tilde{e}(t)$ o $\omega(t)$ nulos. Sin embargo, dado que $\omega(t) = \mathbf{0}$ produce $\tilde{e}(t) = \tilde{\phi}_t^T(t) \omega(t) = \tilde{\phi}_t^T(t) \mathbf{0} = 0$, entonces se toma sin pérdida de generalidad el mínimo en $\tilde{e}(t) = 0$.

Existen, sin embargo, las siguientes posibles situaciones:

1. Si $\omega(t)$ es EIP en el intervalo $\{t, t + 1\}$, entonces de (Ecuación 4.7) se llega a que, entre todo instante de muestreo $[t, t + 1]$,

$$P \left[\tilde{\phi}(t) = \mathbf{0} \right]_t^{t+1} = 1 \Rightarrow P \left[\Theta = \theta^* \right]_t^{t+1} = 1,$$

y dado que

$$\tilde{e}_t = \Theta_t^T \omega_t - y_t = \tilde{\phi}_t^T \omega_t$$

entonces se sigue que

$$P \left[\tilde{e}(t) = 0 \right]_t^{t+1} = 1.$$

Ademas, si $\omega(t)$ es de EIP para todo $t \rightarrow \infty$, se sigue directamente que

$$P \left[\lim_{t \rightarrow \infty} \phi(t) = \mathbf{0} \right] = 1 \Rightarrow P \left[\lim_{t \rightarrow \infty} \hat{\theta}(t) = \theta^* \right] = 1,$$

y

$$P \left[\lim_{t \rightarrow \infty} e(t) = 0 \right] = 1.$$

Nota: Notar que un $\omega(t)$ de excitación persistente no equivale a decir que $\omega(t)$ es de EIP. Esto se ejemplifica en el caso escalar con $\omega(t) = \sin(t)$, puesto que esta señal es de excitación persistente, pero no EIP para los casos en que $\sin(t) = 0$.

2. Sin embargo, cuando $\omega(t)$ no es EIP, entonces se cumple para (Ecuación 4.7) que el mínimo

$$\tilde{e}_t \omega_t = \omega_t \omega_t^T \tilde{\phi}_t = \mathbf{0}$$

se satisface no solamente para $\Theta = \theta^*$, sino para todo Θ que hace que $\tilde{\phi}_t$ y ω_t sean ortogonales. Sin embargo, para un $\omega(t)$ no nulo esto se logra solamente para $\tilde{e}_t = 0$ y esto se consigue para un $\Theta = \bar{\Theta}$ dado (con $\bar{\Theta}$ constante no necesariamente igual a θ^*). Por lo tanto se tiene que

$$P \left[\tilde{e}(t) = 0 \right]_t^{t+1} = 1,$$

y

$$P \left[\tilde{\phi}(t) = \mathbf{C} \right]_t^{t+1} = 1.$$

Con ello, se sigue directamente que

$$P \left[\lim_{t \rightarrow \infty} e(t) = 0 \right] = 1,$$

pero no se puede asegurar que $\lim_{t \rightarrow \infty} \phi(t) = \mathbf{C}$ (es decir, convergencia paramétrica dada por $\lim_{t \rightarrow \infty} \Delta\phi(t) = 0$), ya que \mathbf{C} es diferente entre intervalos, y sobre todo porque para un eventual $\omega(t) = \mathbf{0}$ se tiene que $e(t)\omega(t) = \mathbf{0} \quad \forall \hat{\theta}(t)$.

□

Corolario 4.3.2. *Al usar PSO estándar en lugar de GCPSO, se obtienen resultados similares respecto de $e(t) \rightarrow 0$ cuando $t \rightarrow \infty$. Sin embargo, cada proceso de optimización individual tiene una probabilidad diferente de cero de quedarse estancado, en un punto que ni siquiera sea un óptimo local.*

Demostración. La interconexión de procesos de optimización individuales optimizados con PSO estándar, el cual se reinicia con cada nuevo proceso individual manteniendo la mejor solución previa, es un MPSO, es decir, un algoritmo de optimización global (Teorema 2.5.9). Por tanto, el proceso de optimización completo (no entre instantes de muestreo, sino cuando $t \rightarrow \infty$), es desarrollado con un optimizador global, y por tanto el óptimo $e(t) \rightarrow 0$ cuando $t \rightarrow \infty$ es alcanzado con probabilidad 1. Sin embargo, cada proceso individual es optimizado con PSO estándar, el cual ni siquiera es un optimizador local. □

La importancia del Teorema 4.3.1 y el Corolario 4.3.2 radica en que, independientemente de la técnica utilizada (PSO o GCPSO), e independientemente de que se cumpla o no la condición EIP, se puede asegurar que $e(t) \rightarrow 0$ cuando $t \rightarrow \infty$. Esto es primordial en problemas de control, en los cuales interesa poco los valores que tome $\phi(t)$, incluso si no hay convergencia a un valor constante.

No obstante, en los problemas de identificación sí es importante el valor final de $\phi(t)$, y el Teorema 4.3.1 presenta una debilidad respecto de este aspecto. Se tiene que la convergencia a $\phi(t) = 0$ es alcanzada mientras $\omega(t)$ sea EIP, pero este tipo de señal es muy restrictiva. Como ejemplo, téngase el caso del Modelo de Error 1 cuando $\theta^* \in \mathfrak{R}^1$: aquí, basta que $\omega(t)$ tome un valor arbitrariamente cercano a cero para que no se cumpla la condición EIP. En este caso, $\omega(t) = \sin(t)$ que es una señal de excitación persistente, no es EIP para $\sin(t) = 0$. Como consecuencia, se tiene una pérdida de una convergencia previamente lograda para $\phi(t)$ cuando $t = k\pi$, $k = 0, 1, \dots$, pero en ningún caso esto afecta la convergencia de $e(t)$.

Por tanto, es importante invertir esfuerzos en el diseño de funciones de fitness que favorezcan la convergencia paramétrica, y que sean lo más independiente de $\omega(t)$. Es por ello que a continuación se aborda el problema de la convergencia paramétrica en forma más detallada.

4.3.1.1. Convergencia paramétrica

La solución dada por el Teorema 4.3.1 para actualizar los parámetros no asegura siempre convergencia paramétrica de $\theta(t) \rightarrow \theta^*$, o a otra constante, cuando no se cumple la condición EIP. Esto finalmente es debido a que el funcional empleado (Ecuación 4.6) no es estrictamente convexo, y los múltiples mínimos ocurren para $e(t) = 0$ y valores arbitrarios de $\phi(t)$ que dependen de si $\omega(t)$ es o no EIP (incluido $\phi(t) = 0$ cuando $\omega(t)$ es EIP). Por lo tanto, es necesario asegurar convexidad estricta en los funcionales a utilizar, a fin de garantizar un único óptimo global $e(t) = 0$, $\phi(t) = 0$.

Entonces, si se quiere utilizar como base el funcional (Ecuación 4.6), se requiere de alguna de las extensiones propuestas a continuación:

1. Minimizar un historial del error cuadrático

$$J_2(\Theta) = \frac{1}{2M_1} \sum_{j=0}^{M_1-1} [\Theta^T \omega_{t-j} - y_{t-j}]^2, \quad (4.8)$$

2. Minimizar adicionalmente la variación del vector de parámetros $\Delta\Theta(t) = \Theta - \hat{\theta}(t)$

$$J_3(\Theta) = \frac{\alpha}{2} [\Theta^T \omega_t - y_t]^2 + \frac{(1-\alpha)}{2} \Delta\Theta_t^T \Delta\Theta_t, \quad (4.9)$$

con $\alpha \in [0, 1]$.

3. Minimizar una combinación de J_2 y J_3

$$J_4(\Theta) = \frac{\alpha}{2M_1} \sum_{j=0}^{M_1-1} [\Theta^T \omega_{t-j} - y_{t-j}]^2 + \frac{(1-\alpha)}{2} \Delta\Theta_t^T \Delta\Theta_t, \quad (4.10)$$

con $\alpha \in [0, 1]$.

4. Minimizar una extensión del funcional J_3 , con $\Delta\Theta(t-j) = \Theta - \hat{\theta}(t-j)$:

$$J_5(\Theta) = \frac{\alpha}{2} [\Theta^T \omega_t - y_t]^2 + \frac{(1-\alpha)}{2M_2} \sum_{j=0}^{M_2-1} \Delta\Theta_{t-j}^T \Delta\Theta_{t-j}, \quad (4.11)$$

con $\alpha \in [0, 1]$.

5. Minimizar una combinación entre J_2 y J_5

$$J_6(\Theta) = \frac{\alpha}{2M_1} \sum_{j=0}^{M_1-1} [\Theta^T \omega_{t-j} - y_{t-j}]^2 + \frac{(1-\alpha)}{2M_2} \sum_{j=0}^{M_2-1} \Delta\Theta_{t-j}^T \Delta\Theta_{t-j}, \quad (4.12)$$

con $\alpha \in [0, 1]$.

Sin embargo, es necesario evaluar primero las propiedades de los funcionales propuestos, (convexidad, convexidad estricta, etc.). El análisis de los funcionales J_2 , J_3 , J_4 , J_5 y J_6 se presenta a continuación.

Funcional J_2 El funcional (Ecuación 4.8) se puede reescribir como

$$J_2(\Theta) = \frac{1}{2M_1} \left[\left(\Theta^T \omega_t - y_t \right)^2 + \dots + \left(\Theta^T \omega_{t-(M_1-1)} - y_{t-(M_1-1)} \right)^2 \right].$$

De esta forma, es más claro que la primera derivada parcial de J_2 está dada por

$$\begin{aligned} \frac{\partial J_2(\Theta)}{\partial \Theta} &= \frac{1}{M_1} \left[\left(\Theta^T \omega_t - y_t \right) \omega_t + \dots + \left(\Theta^T \omega_{t-(M_1-1)} - y_{t-(M_1-1)} \right) \omega_{t-(M_1-1)} \right], \\ \Rightarrow \frac{\partial J_2(\Theta)}{\partial \Theta} &= \frac{1}{M_1} \sum_{j=0}^{M_1-1} \left[\left(\Theta^T \omega_{t-j} - y_{t-j} \right) \omega_{t-j} \right] \end{aligned}$$

y con ello el Hessiano de J_2 (segunda derivada parcial) es

$$H_2 = \frac{1}{M_1} \left[\omega_t \omega_t^T + \dots + \omega_{t-(M_1-1)} \omega_{t-(M_1-1)}^T \right] = \frac{1}{M_1} \sum_{j=0}^{M_1-1} \left[\omega_{t-j} \omega_{t-j}^T \right] \geq 0.$$

Igualmente que con J_1 , el funcional J_2 es convexo pero no estrictamente. Por otra parte, queda claro que la condición de excitación persistente se hace más débil en la medida que $M_1 \gg 1$, puesto que ya no se requiere que $\omega(t)$ sea EIP, sino que $\omega(t)$ no sea nulo en todo intervalo $[t - (M_1 - 1), t]$. Además, en la medida que se estimen parámetros invariantes en el tiempo, entonces el único vector Θ que minimiza J_2 en la ventana $[t - (M_1 - 1), t]$ bajo la presencia de excitación persistente es θ^* , y con ello el punto de equilibrio es $\phi(t) = 0$, $e(t) = 0$. Finalmente, para mejorar la velocidad de convergencia, el valor de M_1 debiera ser escogido como $M_1 \geq N$, evitando que en la ventana de minimización haya sobreparametrización (similar a los Índices de Akaike (Burnham and Anderson, 2004)) que produzca un conjunto de ϕ_t ortogonales al historial del error diferentes al óptimo, en vez de una única solución óptima $\phi_t = 0$.

Funcional J_3 Reescribiendo el funcional (Ecuación 4.9) como

$$J_3(\Theta) = \frac{\alpha}{2} \left(\Theta^T \omega_t - y_t \right)^2 + \frac{(1-\alpha)}{2} \left(\Theta - \hat{\theta}_t \right)^T \left(\Theta - \hat{\theta}_t \right),$$

se tiene que su primera derivada parcial respecto de Θ es

$$\frac{\partial J_3(\Theta)}{\partial \Theta} = \alpha \left(\Theta^T \omega_t - y_t \right) \omega_t^T + (1-\alpha) \left(\Theta - \hat{\theta}_t \right),$$

y su Hessiano es

$$H_3 = \alpha \omega_t \omega_t^T + (1-\alpha) I > 0.$$

Inmediatamente se verifica que este funcional es estrictamente convexo, independientemente de si $\omega(t)$ es de excitación persistente o no, dado que H_3 siempre es definido positivo gracias a la matriz I (con $\alpha > 0$). En este caso, el punto de equilibrio (único) está dado por $e(t) = 0$, $\Delta \Theta(t) = 0$, es decir $\hat{\theta}(t) = \hat{\theta}(t-1)$.

Por otra parte, dado que el mínimo de J_3 satisface

$$\begin{aligned} \frac{\partial J_3(\Theta)}{\partial \Theta} &= 0 \\ \implies -\alpha \underbrace{(\Theta^T \omega_t - y_t)}_{\tilde{e}_t} \omega_t &= (1 - \alpha) \Delta \Theta_t, \end{aligned}$$

y es estimado con GCP SO con probabilidad 1, entonces la evolución de $\Delta \Theta_t$ está sujeta a

$$P \left[\Delta \Theta_t = -\frac{\alpha}{(1 - \alpha)} \tilde{e}_t \omega_t \right] = 1.$$

De la minimización de J_3 se pueden sacar las siguientes conclusiones:

- Debido al término $\Delta \Theta_t^T \Delta \Theta_t$, se puede asegurar un mínimo local para J_3 en $\Delta \Theta_t = 0$ si $e(t) = 0$ (es decir, $J_3(\Theta) = J_3(\hat{\theta}_t)$ ya que $\Delta \Theta_t = 0 \rightarrow \Theta - \hat{\theta}_t = 0 \rightarrow \Theta = \hat{\theta}_t$). Por lo tanto, en el siguiente paso se garantiza al menos este punto, y sólo se actualiza $\Delta \Theta_t$ en la medida que $J_3(\Theta) < J_3(\hat{\theta}_t)$. De esta forma se asegura un $\|\Theta - \hat{\theta}_t\| \leq \|\hat{\theta}_t - \hat{\theta}_{t-1}\|$.
- Dado que $J_3 \doteq f(\Delta \Theta_t)$, entonces las cotas sobre la variación de $\Delta \Theta_t$ están dadas por:

$$0 \leq (1 - \alpha) \|\Delta \Theta_t\| \leq \alpha \left| (\hat{\theta}_t - \theta^*)^T \omega_t \right| = \alpha |\tilde{e}_t|. \quad (4.13)$$

- Dado que $\tilde{e}_t \in \Re$ y $\Theta, \theta^* \in \Re^N$, entonces no existe un único $\Delta \Theta_t$ ortogonal a ω_t , incluso con excitación persistente. Es más, el vector $\Theta = \theta^*$ sólo minimiza a J_3 en la medida que $\|\theta^* - \hat{\theta}_t\| = 0$. Por lo tanto, se cumple lo siguiente
 - Si $\Theta = \hat{\theta}_t$ no minimiza a \tilde{e}_{t-1} y a \tilde{e}_t , entonces $\Delta \Theta_t$ será diferente de cero de modo que se minimice $J_3(t)$, con un ϕ_t ortogonal a ω_t obtenido por un paso $\|\Delta \Theta_t\| \leq \frac{\alpha}{1 - \alpha} \|\tilde{e}_t\|$ (por la Ecuación 4.13). Ahora, dado que $\|\Theta - \hat{\theta}_t\| \leq \|\hat{\theta}_t - \hat{\theta}_{t-1}\|$, entonces la evolución de $\Delta \Theta_t^T \Delta \Theta_t$ está dada por

$$\begin{aligned} \|\Delta \Theta_t^T \Delta \Theta_t\| &\leq \left| (\hat{\theta}_{t-1} - \theta^*)^T \omega_t \right|^2, \\ \implies \|\Delta \Theta_t^T \Delta \Theta_t\| &\leq \frac{\alpha}{1 - \alpha} \left| e_t - (\hat{\theta}_t - \hat{\theta}_{t-1})^T \omega_t \right|^2, \end{aligned}$$

y teniendo que $\|\hat{\theta}_t - \hat{\theta}_{t-1}\| \leq \|\hat{\theta}_{t-1} - \hat{\theta}_{t-2}\| \dots \leq \|\hat{\theta}_0 - \theta^*\|$, la expresión anterior tiende a cero cuando $t \rightarrow \infty$ ($P[\lim_{t \rightarrow \infty} \Delta \phi_t = 0] = 1 \implies P[\lim_{t \rightarrow \infty} \hat{\theta}_t = \mathbf{C}] = 1$), dado que el mínimo de J_3 es $\tilde{e}_t = 0$ cuando $\Delta \Theta_t = 0$. De esta forma también se logra que $P[\lim_{t \rightarrow \infty} J_3 = 0] = 1$, y con ello $P[\lim_{t \rightarrow \infty} e_t = 0] = 1$, y dada la presencia de excitación persistente, se asegura además que $\mathbf{C} = \theta^*$, y con ello $P[\lim_{t \rightarrow \infty} \hat{\theta}_t = \theta^*] = 1$.

Un ejemplo de evolución de $\hat{\theta}(t) \in \Re^1$ utilizando la metodología propuesta, basada en PSO, es mostrado en la Figura 4.2. Allí se muestra que el proceso de optimización, bajo condiciones de

excitación persistente, siempre mejora el desempeño tras cada nueva muestra de datos, garantizándose que al menos la solución en el siguiente proceso individual es mejor o igual que la obtenida en el anterior proceso individual.

Funcional J_4 Reescribiendo el funcional (Ecuación 4.10) como

$$J_4(\Theta) = \frac{\alpha}{2M_1} \left[(\Theta^T \omega_t - y_t)^2 + \dots + (\Theta_t^T \omega_{t-(M_1-1)} - y_{t-(M_1-1)})^2 \right] + \frac{(1-\alpha)}{2} (\Theta - \hat{\theta}_t)^T (\Theta - \hat{\theta}_t),$$

su derivada parcial queda definida por

$$\frac{\partial J_4(\Theta)}{\partial \Theta} = \frac{\alpha}{M_1} \sum_{j=0}^{M_1-1} [(\Theta^T \omega_{t-j} - y_{t-j}) \omega_{t-j}] + (1-\alpha) (\Theta - \hat{\theta}_t),$$

y su Hessiano por

$$H_4 = \frac{\alpha}{M_1} \sum_{j=0}^{M_1-1} [\omega_{t-j} \omega_{t-j}^T] + (1-\alpha) I > 0.$$

Nuevamente el funcional resulta ser estrictamente convexo, y con las propiedades heredadas de J_2 y J_3 , se tiene que el punto de equilibrio es $e(t) = 0$, $\Delta\phi(t) = 0$, y además $\phi(t) = 0$ bajo la presencia de excitación persistente. En la medida en que M_1 sea más grande (y preferiblemente $M_1 \geq N$), habrá menos sobreparametrización respecto del ajuste del error en la ventana de minimización. Esto permite, por un lado, que se reduzca el tamaño del conjunto de ϕ_t ortogonales al historial del error, y por otro lado, que aumente la velocidad de convergencia dado que $\Delta\Theta_t^T \Delta\Theta_t$ tiende a cero a una mayor tasa, acotada ahora por $\frac{\alpha}{1-\alpha} (|\tilde{e}_t|^2 + |\tilde{e}_{t-1}|^2 + \dots + |\tilde{e}_{t-M}|^2)$.

Funcional J_5 Reescribiendo el funcional (Ecuación 4.11) como

$$J_5(\Theta) = \frac{\alpha}{2} (\Theta^T \omega_t - y_t)^2 + \frac{(1-\alpha)}{2M_2} [\Delta\Theta_t^T \Delta\Theta_t + \dots + \Delta\Theta_{t-j}^T \Delta\Theta_{t-j}],$$

su primera derivada parcial respecto de Θ es

$$\begin{aligned} \frac{\partial J_5(\Theta)}{\partial \Theta} &= \alpha (\Theta^T \omega_t - y_t) \omega_t + \frac{(1-\alpha)}{M_2} [(\Theta - \hat{\theta}_{t-1}) + \dots + (\Theta - \hat{\theta}_{t-M_2})] \\ &= \alpha (\Theta^T \omega_t - y_t) + \frac{(1-\alpha)}{M_2} \sum_{j=0}^{M_2-1} \Delta\Theta_{t-j}, \end{aligned}$$

y su Hessiano es

$$H_5 = \alpha \omega_t \omega_t^T + (1-\alpha) I > 0.$$

Este funcional tiene como posible punto de equilibrio $e(t) = 0$, $\hat{\theta}(t) = \text{constante}$ (es decir, $\Delta\phi_t = 0$), pero esto es altamente dependiente de factores como α , M_2 y las condiciones iniciales de la estimación paramétrica. En general, la convexidad del funcional asegura el acotamiento de $e(t)$ y de $\hat{\theta}(t)$, pero la respuesta transitoria de las estimaciones paramétricas ofrecida no son favorables en comparación a los casos anteriores. Esto es debido a que se están tomando como referencia valores anteriores de $\hat{\theta}_t$, que no necesariamente han sido bien estimados, introduciendo de esta forma cierta especie de “ruido” en las nuevas estimaciones paramétricas. Por otra parte, la velocidad de convergencia, la convergencia misma, y la calidad del transitorio empeoran en la medida en que M_2 se hace cada vez más grande.

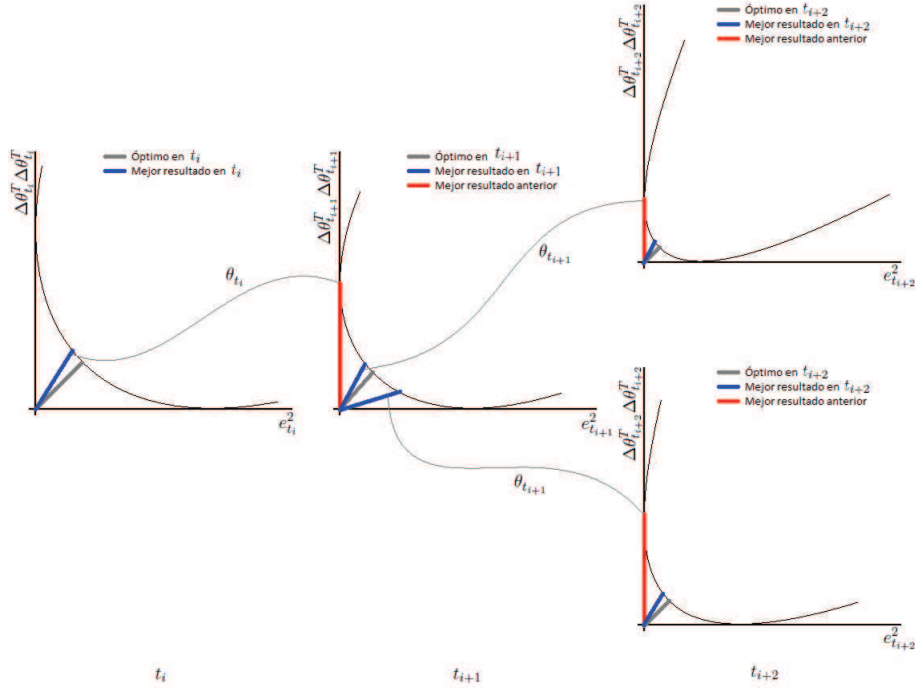


Figura 4.2.: Evolución gráfica de la optimización del funcional J_3 en \mathbb{R}^1 .

Funcional J_6 Reescribiendo el funcional (Ecuación 4.12) como

$$J_6(\Theta) = \frac{\alpha}{2M_1} \left[(\Theta^T \omega_t - y_t)^2 + \dots + (\Theta^T \omega_{t-(M_1-1)} - y_{t-(M_1-1)})^2 \right] + \frac{(1-\alpha)}{2M_2} \left[\Delta\Theta_t^T \Delta\Theta_t + \dots + \Delta\Theta_{t-j}^T \Delta\Theta_{t-j} \right],$$

su derivada parcial queda definida por

$$\frac{\partial J_6(\Theta)}{\partial \Theta} = \frac{\alpha}{M_1} \sum_{j=0}^{M_1-1} \left[(\Theta^T \omega_{t-j} - y_{t-j}) \omega_{t-j} \right] + \frac{(1-\alpha)}{M_2} \left[(\Theta - \hat{\theta}_{t-1}) + \dots + (\Theta - \hat{\theta}_{t-M_2}) \right],$$

y su Hessiano por

$$H_6 = \frac{\alpha}{M_1} \sum_{j=0}^{M_1-1} [\omega_{t-j} \omega_{t-j}^T] + (1 - \alpha) I > 0.$$

Este funcional también tiene como posible punto de equilibrio $e(t) = 0$, $\Delta\Theta(t) = 0$, y también es altamente dependiente de factores como α , M_2 y las condiciones iniciales de la estimación paramétrica. Al igual que el funcional J_5 , se acota $e(t)$ y $\hat{\theta}(t)$, pero se obtiene una calidad desmejorada de los transitorios en la estimación paramétrica, en comparación con los demás funcionales analizados (e incluso $e(t)$ acotado pero sin convergencia a cero).

4.3.1.2. Relación entre el factor de ponderación y la ganancia de adaptación

Debido a que (Ecuación 4.9) es estrictamente convexo, se tiene que el mínimo se encuentra en $\frac{\partial J_3(\Theta)}{\partial \Theta} = 0$, es decir en

$$\Theta = \hat{\theta}_t - \frac{\alpha}{(1 - \alpha)} \tilde{e}_t \omega_t,$$

lo cual coincide con la ley de ajuste de parámetros hecha con gradiente, para

$$\frac{\alpha}{1 - \alpha} = \gamma. \tag{4.14}$$

Este resultado permite evaluar comparativamente las leyes de adaptación usando gradiente con ganancia de adaptación γ constante y PSO con factor de ponderación α .

4.3.2. Modelo de Error 2

Para el caso del Modelo de Error 2, se analiza inicialmente su versión tradicional, y luego se pasa a analizar un caso particular.

4.3.2.1. Modelo de Error 2: caso tradicional

Sea el Modelo de Error 2 (Duarte-Mermoud and Narendra, 1996) definido por (Ecuación 2.17). Entonces, redefiniendo $\epsilon_{t+1} = e_{t+1} - Ae_t$ se obtiene (Ecuación 2.19) de la forma

$$\epsilon_{t+1} = \phi_{t+1}^T \omega_t,$$

llegando así a un modelo del error equivalente al Modelo del Error 1. Definiendo un funcional similar a (Ecuación 4.6) de la forma

$$J = \frac{1}{2} \epsilon_t^T \epsilon_t \tag{4.15}$$

con gradiente

$$\frac{\partial J}{\partial \hat{\theta}_t} = \epsilon_t^T \omega_{t-1} \quad (4.16)$$

y Hessiano

$$H = \frac{\partial^2 J}{\partial \hat{\theta}_t^{(i)} \partial \hat{\theta}_t^{(i)}} = \frac{\partial}{\partial \hat{\theta}_t^{(i)}} (\omega \epsilon_t) = \frac{\partial}{\partial \hat{\theta}_t^{(j)}} (\omega_{t-1}^T (\phi_t \omega_{t-1})), \quad \forall i, j = 1, 2, \dots, N$$

$$H = \frac{\partial}{\partial \hat{\theta}^{(j)}} (\omega \phi_t \omega_{t-1}), \quad \forall i, j = 1, 2, \dots, N$$

$$H = [\omega_{t-1}] [\omega_{t-1}]^T \geq 0, \quad (4.17)$$

se obtienen los mismos resultados para J_1 en la Subsección 4.3.1 con similares propiedades. Con un proceso de adecuación similar, pueden ser obtenidos funcionales equivalentes a J_1 - J_6 de la Subsubsección 4.3.1.1. Sin embargo, dado que las propiedades están directamente relacionadas con ϵ_t y no con e_t , entonces es necesario hacer un análisis adicional. Ya que para (Ecuación 4.15) se tiene la propiedad

$$P \left[\lim_{t \rightarrow \infty} \epsilon(t) = 0 \right] = 1,$$

y dado que $\epsilon(t) = e(t) - Ae(t-1)$, entonces se sigue que la recurrencia

$$e_t - Ae_{t-1} = 0$$

$$\implies e_t = Ae_{t-1}$$

es estable con punto de equilibrio $e(t) = 0$, dado que A es asintóticamente estable. Por lo tanto, como consecuencia directa se logra que

$$P \left[\lim_{t \rightarrow \infty} e(t) = 0 \right] = 1,$$

$$P \left[\lim_{t \rightarrow \infty} \Delta \phi(t) = 0 \right] = 1.$$

y en presencia de excitación persistente se tiene

$$P \left[\lim_{t \rightarrow \infty} \phi(t) = 0 \right] = 1.$$

4.3.2.2. Modelo del Error 2: caso particular

Sea el caso particular del Modelo de Error 2 (Duarte-Mermoud and Narendra, 1996)(con $k = 1$) definido por:

$$e_t = Ae_{t-1} + b\phi_t^T \omega_{t-1}, \quad \phi_t^T, \omega_{t-1}, e_t, y_t \in \mathfrak{R}^n, \quad A \in \mathfrak{R}^{n \times n}.$$

Redefiniendo $\epsilon_t = e_t - Ae_{t-1}$, y considerando el funcional cuadrático

$$J = \frac{1}{2} \epsilon_t^T \epsilon_t, \tag{4.18}$$

se desea hacer un análisis similar al de los casos anteriores. Entonces se sigue que

$$\frac{\partial J}{\partial \hat{\theta}_t} = b\epsilon_t^T \omega_{t-1} \tag{4.19}$$

y

$$H = \frac{\partial^2 J}{\partial \hat{\theta}_t^{(i)} \partial \hat{\theta}_t^{(j)}} = \frac{\partial}{\partial \hat{\theta}_t^{(j)}} (b\omega_{t-1}^T \tilde{\epsilon}_t) = \frac{\partial}{\partial \hat{\theta}_t^{(j)}} (b\omega_{t-1}^T (b\tilde{\phi}_t \omega_{t-1})), \quad \forall i, j = 1, 2, \dots, N$$

$$H = \frac{\partial}{\partial \hat{\theta}_t^{(j)}} (b\omega_{t-1}^T b\tilde{\phi}_t \omega_{t-1}), \quad \forall i, j = 1, 2, \dots, N$$

$$H = [b\omega_{t-1}] [b\omega_{t-1}]^T \geq 0. \tag{4.20}$$

Este funcional es similar al funcional J_1 , con la misma propiedad de convexidad estricta dependiente de si ω_{t-1} es de excitación persistente o no. Por lo tanto, las mismas modificaciones exitosas hechas al mismo (funcionales J_2 , J_3 y J_4) pueden ser aplicadas a este caso también.

4.3.3. Modelo de Error 3

Sea el caso del Modelo de Error 3 (Duarte-Mermoud and Narendra, 1996) definido como:

$$e(t+1) = Ae(t) + bv(t)$$

$$e_1(t) = c^T e(t) + dv(t)$$

$$v(t) = \phi^T(t) u(t) - \alpha u^T(t) \Gamma u(t) e_1(t); \quad \alpha > \frac{1}{2} \quad \Gamma = \Gamma^T > 0,$$

con $W(z) = d + c^T (zI - A)^{-1} b$ una función de transferencia SPR, de tal modo que $e_1(t)$ puede ser reescrito como

$$e_1(t) = W(z) [v(t)].$$

Entonces, la leyes de ajuste quedan dadas por

$$\Delta\phi(t) \triangleq \phi(t+1) - \phi(t) = -\alpha\Gamma e_1(t) u(t), \quad 0 < \alpha < 2, \quad \Gamma = \Gamma^T.$$

Aquí no se puede usar directamente la metodología propuesta, y esto es debido a que en un funcional de la forma

$$J = \frac{1}{2} e_1^2(t),$$

se tiene que

$$J = \frac{1}{2} \left[W(z) \left[\phi^T(t) \omega(t) \right] - \alpha W(z) \left[\omega^T(t) \Gamma \omega(t) e_1(t) \right] \right]^2,$$

$$J = \frac{1}{2} \left[W(z) \left[\left(\hat{\theta}(t) - \theta^* \right)^T \omega(t) \right] - \alpha W(z) \left[\omega^T(t) \Gamma \omega(t) e_1(t) \right] \right]^2,$$

y el vector de parámetros a estimar no aparece explícitamente para el instante de muestreo actual, sino filtrado a través de $W(z)$. Por lo tanto, una posibilidad es reescribir el error de la forma

$$Z(t) = e_1(t) = W(z) \left[\left(\hat{\theta}^T(t) - \theta^{*T} \right) \omega(t) - \alpha \omega^T(t) \Gamma \omega(t) e_1(t) \right]$$

$$\Rightarrow Z(t) = -\theta^{*T} W(z) [\omega(t)] + W(z) \left[\hat{\theta}^T(t) \omega(t) - \alpha \omega^T(t) \Gamma \omega(t) e_1(t) \right].$$

Con esto, y a partir del estimado de $Z(t)$ dado por

$$\hat{Z}(t) = -\hat{\theta}^T W(z) [\omega(t)] + W(z) \left[\hat{\theta}^T(t) \omega(t) - \alpha \omega^T(t) \Gamma \omega(t) e_1(t) \right],$$

podemos llegar al modelo paramétrico estático (SPM por *static parametric model*) (Ioannou and Fidan, 2006) de la forma

$$\epsilon(t) = Z(t) - \hat{Z}(t) = \phi^T(t) \tilde{u}(t), \tag{4.21}$$

$$e_1(t) = W(z) [v(t)]$$

$$Z(t) = e_1(t), \quad \tilde{u}(t) = W(z) [\omega(t)],$$

que no es otra cosa que el equivalente al Modelo de Error 1, y entonces se pueden usar las leyes de ajuste propuestas para ese modelo a fin de obtener las mismas propiedades de convergencia para $\epsilon(t)$. No obstante, es necesario obtener tales características para $e_1(t)$ en vez de $\epsilon(t)$, para lo cual se hace el siguiente análisis: ya que ϵ tiende a cero, entonces necesariamente $\phi^T(t)$ tiende a una constante tal que $\phi^T(t) \tilde{u}(t) = 0$ (si $\tilde{u}(t)$ es de excitación persistente, esa constante es cero, es decir $\hat{\theta}(t) = \theta^*$). Ahora, dado que $\phi^T(t) \tilde{u}(t) = 0$, entonces de la ecuación de $e_1(t)$ se tiene

$$e_{1,t} = \underbrace{\phi_t^T \tilde{u}_t}_0 + \hat{\theta}_t^T W(z) [\omega_t] - W(z) \left[\hat{\theta}_t^T \omega_t \right] - W(z) \left[\alpha \omega_t^T \Gamma \omega_t e_{1,t} \right],$$

y para el estado estacionario de $\hat{\theta}_t$

$$e_{1t} = -\alpha W(z) \left[\omega_t^T \Gamma \omega_t e_{1,t} \right],$$

$$\Rightarrow W^{-1}(z) [e_{1,t}] = -\underbrace{\alpha \omega_t^T \Gamma \omega_t}_{>0} e_{1,t},$$

y dado que $W(z)$ es SPR, entonces tiene solución en $e_1 \rightarrow 0$ cuando $t \rightarrow \infty$, independientemente de las condiciones iniciales.

4.4. Resultados experimentales

A fin de hacer un análisis experimental del desempeño obtenido con la metodología propuesta, en la presente sección se presentan un conjunto de ejemplos de aplicación para los cuales se utiliza PSO y GCPSO optimizando los funcionales J_1, J_2, J_3, J_4, J_5 y J_6 con base en el ToolBox PSO de J. Singh (Singh, 2003)(ver Apéndice A) para implementar los algoritmos PSO/GCPSO, en comparación con el método NG y el método NLS.

Para las pruebas desarrolladas se tendrán las siguientes consideraciones:

- Configuración para NG: $\Gamma = \gamma I$ con $\gamma \in \{0,5, 1,5\}$, $\kappa = 1$.
- Configuración para NLS: $\Gamma_0 = \alpha_{LS} I$ con $\alpha_{LS} \in \{0,5, 1,5\}$, $\kappa = 1$.
- Configuración para PSOiw: peso de inercia linealmente decreciente con $w(0) = 0,9$ y $w(iter_{\max}) = 0,4$, $c_1 = c_2 = 1,49445$, $iter_{\max} = 20 * \lambda$, $f_{\min} = 1 \times 10^{-10}$, $s = 2 * \lambda$, $M_1 = \{1, 6\}$, $M_2 \in \{1, 2\}$, $\alpha \in [0, 1]$ (Ver Subsección 2.5.1).
- Configuración para GCPSO: $\rho_0 = 1$, $s_c = 15$, $f_c = 5$.
- Configuración de Simulink: Tiempo de muestreo $0,1[s]$, Solver *Discrete* .

Adicionalmente, se denotará como GC-PSO a las metodologías basadas en PSO o GCPSO cuando se hable indistintamente de ambas, teniendo los siguientes esquemas de configuración específica:

- GC-PSOJ1: $M_1 = 1$, $M_2 = 1$, $\alpha = 1$.
- GC-PSOJ2: $M_1 = 6$, $M_2 = 1$, $\alpha = 1$.
- GC-PSOJ3: $M_1 = 1$, $M_2 = 1$, $\alpha = 0.\bar{3}$, $\alpha = 0,6$.
- GC-PSOJ4: $M_1 = 6$, $M_2 = 1$, $\alpha = 0.\bar{3}$, $\alpha = 0,6$.
- GC-PSOJ5: $M_1 = 1$, $M_2 = 2$, $\alpha = 0.\bar{3}$, $\alpha = 0,6$.
- GC-PSOJ6: $M_1 = 6$, $M_2 = 2$, $\alpha = 0.\bar{3}$, $\alpha = 0,6$.

La escogencia de los valores propuestos para α vienen de la equivalencia (Ecuación 4.14), teniendo que $\alpha = 0.\bar{3}$ es equivalente a $\gamma = 0,5$, y $\alpha = 0,6$ es equivalente a $\gamma = 1,5$.

Para el caso de las configuraciones GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6, el correspondiente valor de α estará puesto a continuación de la sigla en medio de paréntesis. De esta forma, por ejemplo, GC-PSOJ4(0.6) hace referencia a GC-PSOJ4 con $\alpha = 0,6$.

Los principales aspectos a tener en cuenta en la comparación son, en principio, la magnitud del error de estado estable y los tiempos de convergencia (cuantitativos), y la turbulencia del transitorio (cualitativo), pero no necesariamente los únicos.

4.4.1. Modelo de Error 1

A continuación se presentan los resultados obtenidos de las pruebas hechas para el Modelo de Error 1.

4.4.1.1. Excitación persistente versus EIP

Se decide utilizar el esquema de identificación más simple para evaluar la respuesta de los funcionales para señales de Excitación Persistente (EP) y señales EIP.

Sea el problema de identificación estático con un solo parámetro, cuya planta a identificar tiene la forma

$$x(t) = \theta_1 u_1(t-1),$$

para la cual el modelo de identificación es escogido como

$$\hat{x}(t) = \hat{\theta}_1(t) u_1(t-1).$$

Definiendo el error como $e(t) = \hat{x}(t) - x(t)$, se obtiene el Modelo de Error 1 (Ecuación 2.12), donde

$$\phi^T(t) = \tilde{\theta}(t) \in \mathfrak{R}^1, \quad \omega(t) = u_1(t-1) \in \mathfrak{R}^1,$$

$$\tilde{\theta}_1(t) = \hat{\theta}_1(t) - \theta_1.$$

La configuración usada en las pruebas es la siguiente:

- Señales de entrada: Tipo I) $u_1(t) = 1$ (EP y EIP), Tipo II) $u_1(t) = e^{-t}$ (no EP, y sólo EIP para $u_1(t) \gg 0$), y Tipo III) $u_1(t)$ es un tren de pulsos de amplitud 1, período 1,4[s] y ancho de pulso 50% (EP, y sólo EIP para $u_1(t) \neq 0$).
- Parámetros de la planta: $\theta_1 = 2$.
- Configuración de PSO: dado que hay 1 incógnita, entonces $\lambda = 1$, y con ello $iter_{\max} = 20$, $s = 2$.
- Condiciones iniciales: $[x(0), \hat{x}(0)] = [10, 0]$, $\hat{\theta}(0) = 0$.
- Tiempo de simulación: 3[s].

Los experimentos realizados con las entradas Tipo I, II y III son presentados en la colección de imágenes desde la Figura 4.3 hasta la Figura 4.14.

Los resultados al utilizar la entrada Tipo I son presentados en las Figura 4.3, Figura 4.4, Figura 4.5 y Figura 4.6. Para este tipo de entrada (que es tanto EP como EIP), se tiene que, a excepción

4.4 Resultados experimentales

de los casos GC-PSOJ5 y GC-PSOJ6, las metodologías basadas en PSO cumplen el objetivo de conseguir $e(t) \rightarrow 0$ para $t \rightarrow \infty$ satisfactoriamente, independientemente de la comparación con las otras metodologías. No obstante, para los casos GC-PSOJ5 y GC-PSOJ6 se pudo observar que la evolución de $e(t)$ es altamente dependiente de las condiciones iniciales de la estimación paramétrica (en cuanto más lejos esté la condición inicial del valor real del parámetro, más lejos se estabiliza), del valor de M_2 , y del valor de α . Sin embargo, este comportamiento estaba previsto desde el análisis de los funcionales (ver Subsubsección 4.3.1.1).

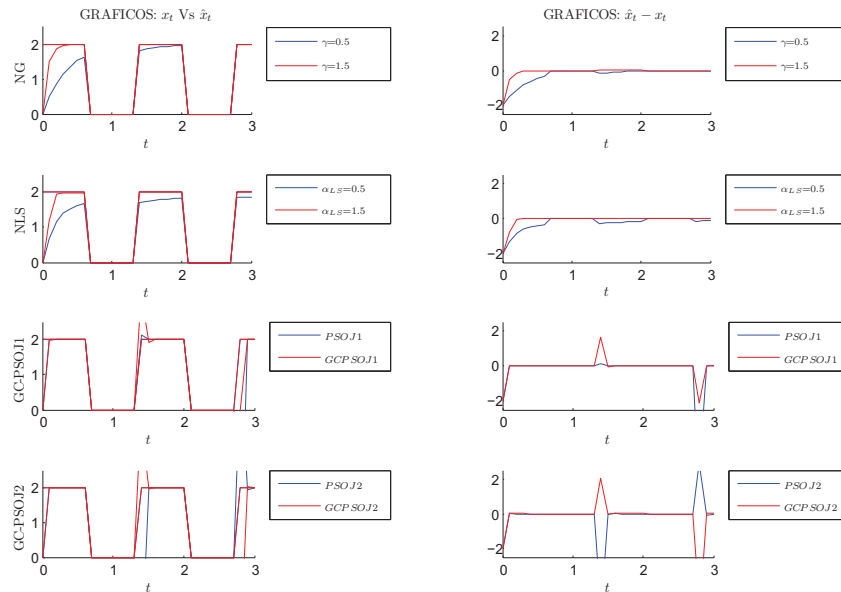


Figura 4.3.: Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo I.

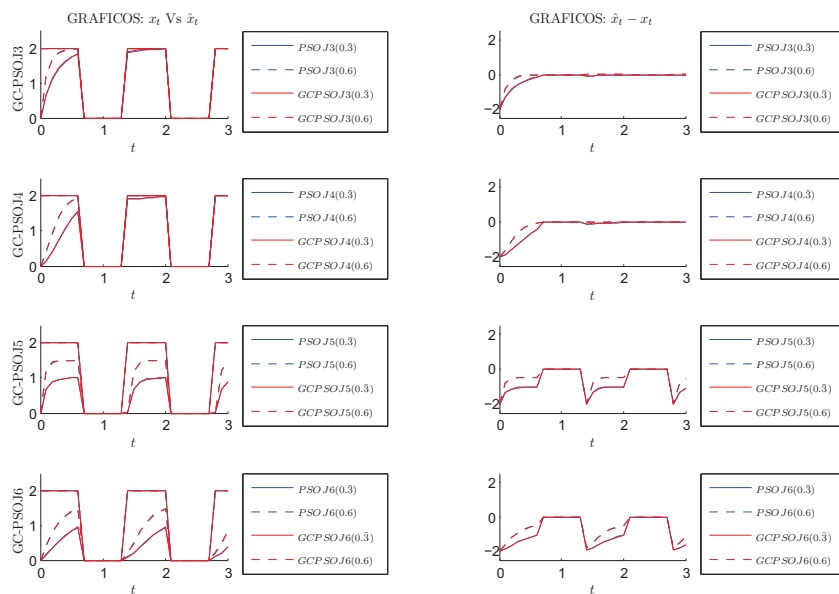


Figura 4.4.: Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo I.

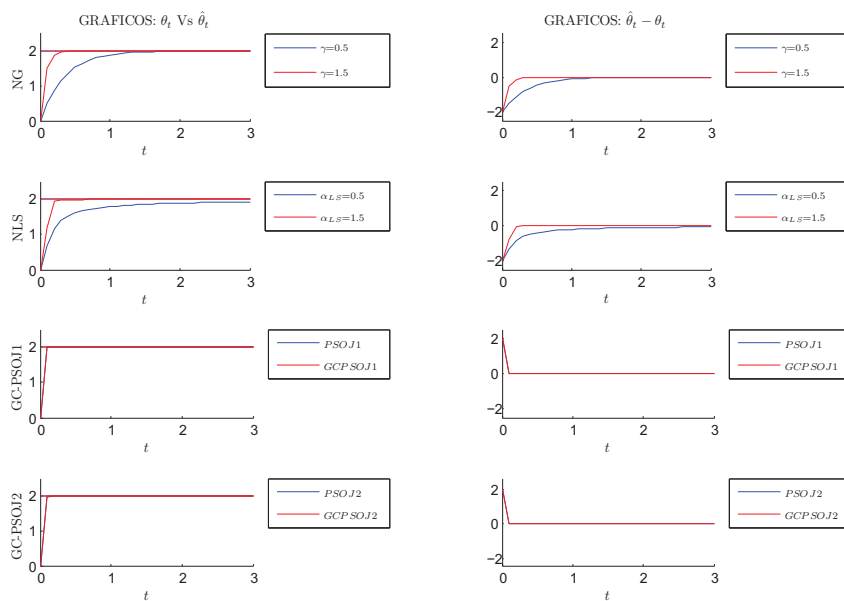


Figura 4.5.: Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo I.

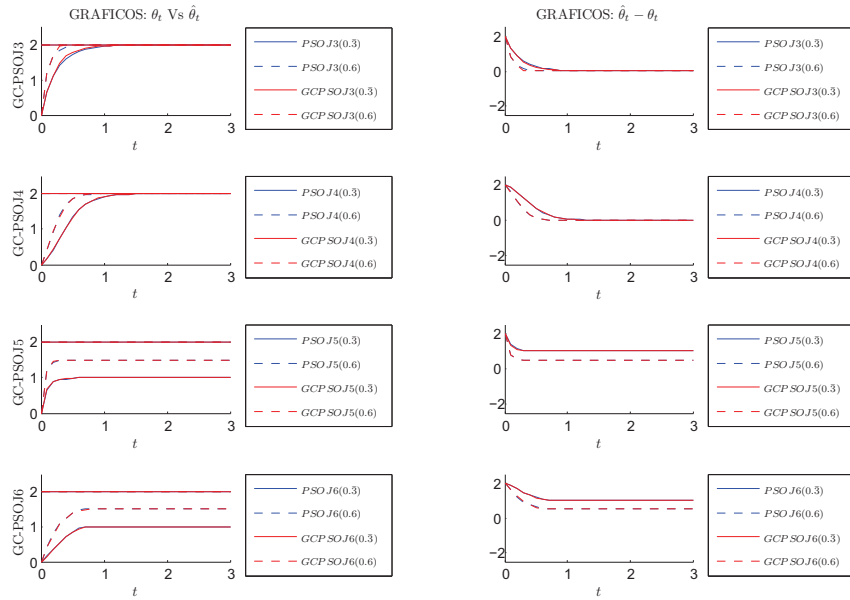


Figura 4.6.: Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo I.

Comparativamente hablando, las respuestas de GC-PSOJ1 y GC-PSOJ2 son las mejores obtenidas respecto de todas las metodologías bajo análisis, tanto para el error de identificación como para el error paramétrico. No obstante, la eficiencia de GC-PSOJ3 y GC-PSOJ4 puede ser mejorada escogiendo adecuadamente α .

Para el caso de utilizar la entrada Tipo II, se obtuvieron los resultados presentados en las Figura 4.7, Figura 4.8, Figura 4.9 y Figura 4.10. Este tipo de entrada no es EP, y mientras $u(t) \gg 0$ es EIP. Las conclusiones para este tipo de entrada son similares al caso anterior: GC-PSOJ1 y GC-PSOJ2 tienen el mejor desempeño global, GC-PSOJ3 y GC-PSOJ4 tienen buen desempeño comparativo con NG y mejor que NLS para ciertos casos (mejorable aún más con valores más adecuados de α), y pobre desempeño para GC-PSOJ5 y GC-PSOJ6 (ni siquiera se minimiza el error de identificación). No obstante el buen desempeño de GC-PSOJ1 y GC-PSOJ2, se tiene que cuando no se cumple $u(t) \gg 0$ (aproximadamente a partir de $u(t) = 10e^{-10}$, según los experimentos) la estimación paramétrica se pierde, a pesar de que el error de identificación se mantiene en $e(t) = 0$; esto es una consecuencia directa del no cumplimiento de la condición EIP. Sin embargo, para el caso de GC-PSOJ3 y GC-PSOJ4 el desempeño no se ve afectado por el no cumplimiento de la condición EIP, teniendo que GC-PSOJ4, con las propiedades heredadas de GC-PSOJ2 (minimizar un historial del error), aprovecha de mejor forma el transitorio para la estimación paramétrica, logrando con esto convergencia paramétrica en casos en que NG o NLS no pudieran lograrla.

4.4 Resultados experimentales

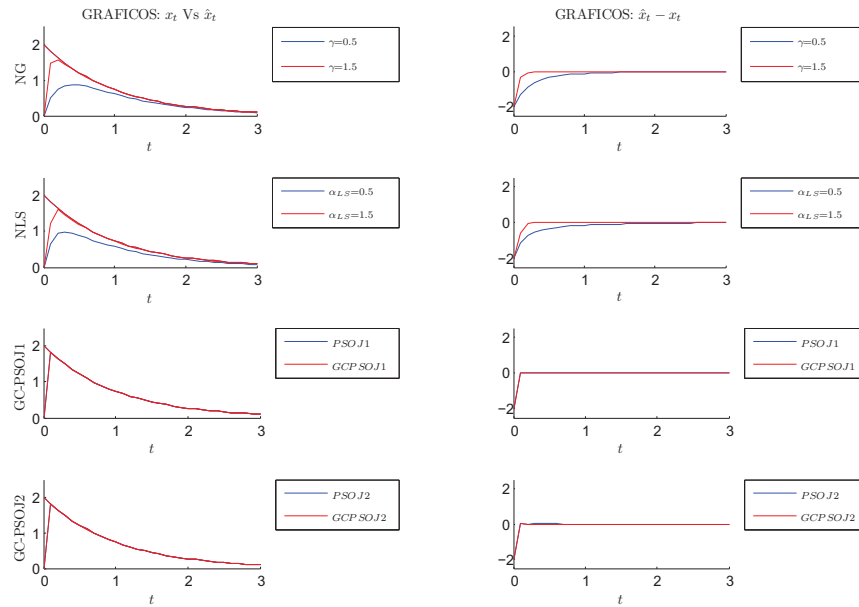


Figura 4.7.: Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo II.

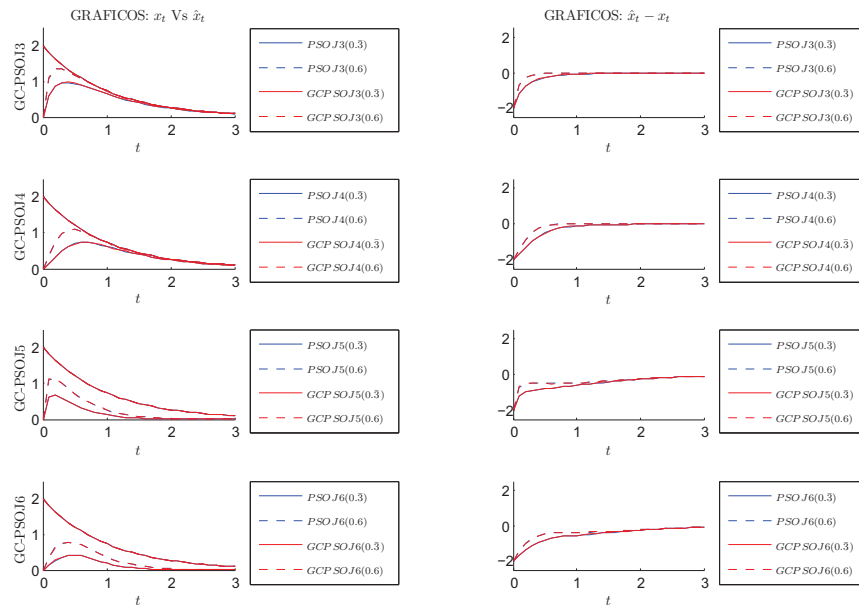


Figura 4.8.: Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para GC-PSO J3, GC-PSO J4, GC-PSO J5 y GC-PSO J6. Señal de entrada utilizada: Tipo II.

4.4 Resultados experimentales

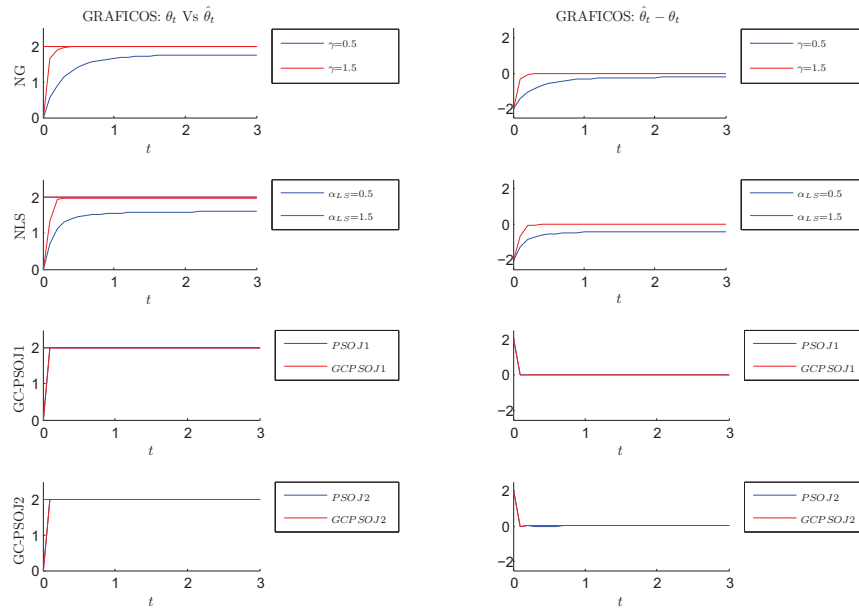


Figura 4.9.: Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo II.

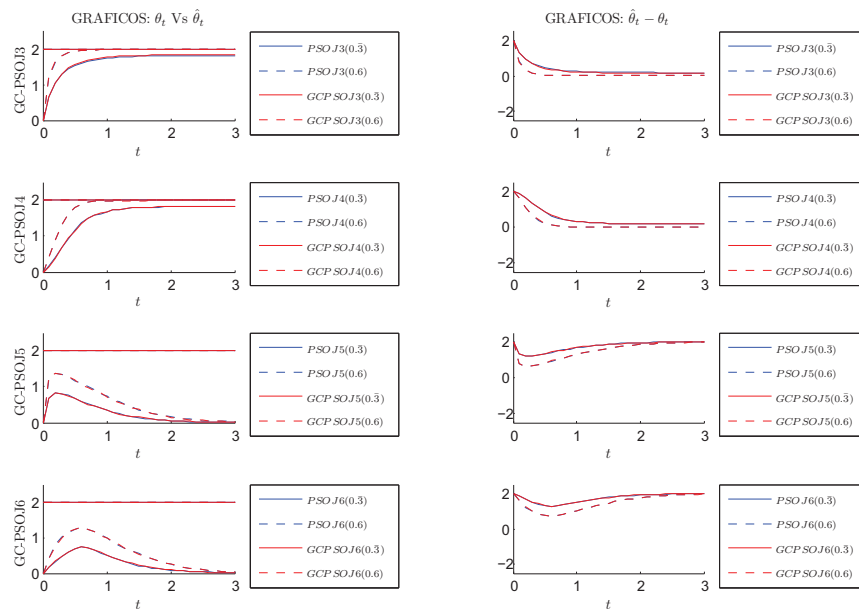


Figura 4.10.: Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo II.

Finalmente, cuando se utiliza la entrada Tipo III, se obtienen los resultados mostrados en las Figura 4.11, Figura 4.12, Figura 4.13 y Figura 4.14. Este tipo de entrada es de EP, y EIP para todo t tal que $u(t) \neq 0$.

Para este tipo de señal es posible verificar mejor la principal desventaja de GC-PSOJ1 y GC-PSOJ2: una vez obtenida convergencia paramétrica, ésta se puede perder si la señal de entrada pierde la propiedad EIP. Sin embargo, al mismo tiempo se pueden verificar los beneficios de GC-PSOJ3 y GC-PSOJ4: la convergencia paramétrica no se ve desmejorada cuando la entrada no es de EIP, y se mantiene acotada por la minimización del término $\Delta\Theta_t^T \Delta\Theta_t$.

Una vez analizadas todas la respuestas de las metodologías basadas en PSO respecto de las propiedades EP y EIP del vector de información $\omega(t)$, se decide solamente utilizar GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4, en los ejemplos subsiguientes, y eventualmente GC-PSOJ1 en algún caso caso de interés.

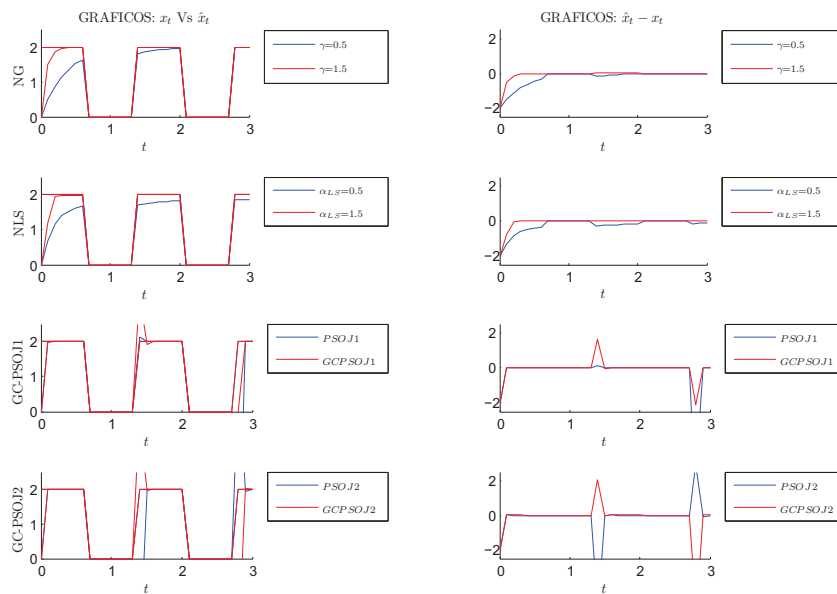


Figura 4.11.: Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo III.

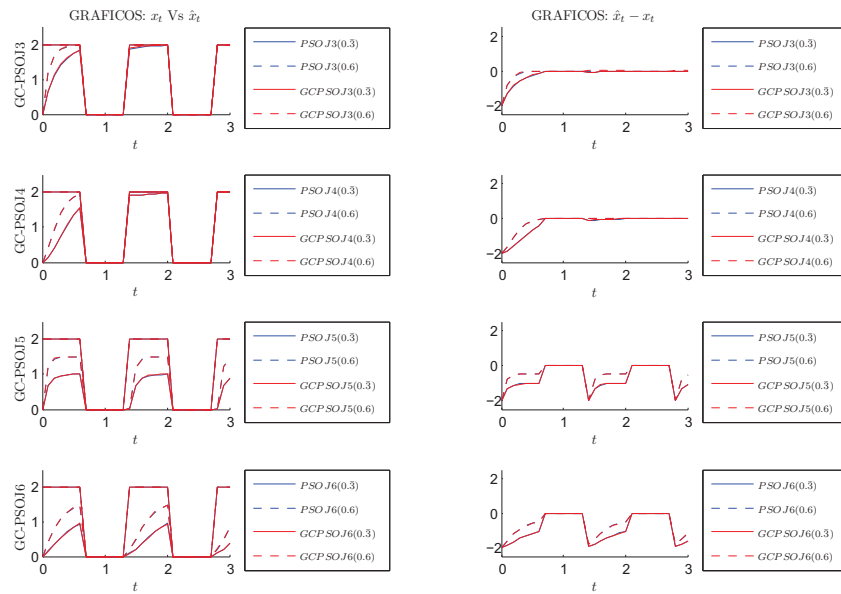


Figura 4.12.: Resultados de simulación para la Subsubsección 4.4.1.1: Comparación de salidas y error de identificación para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo III.

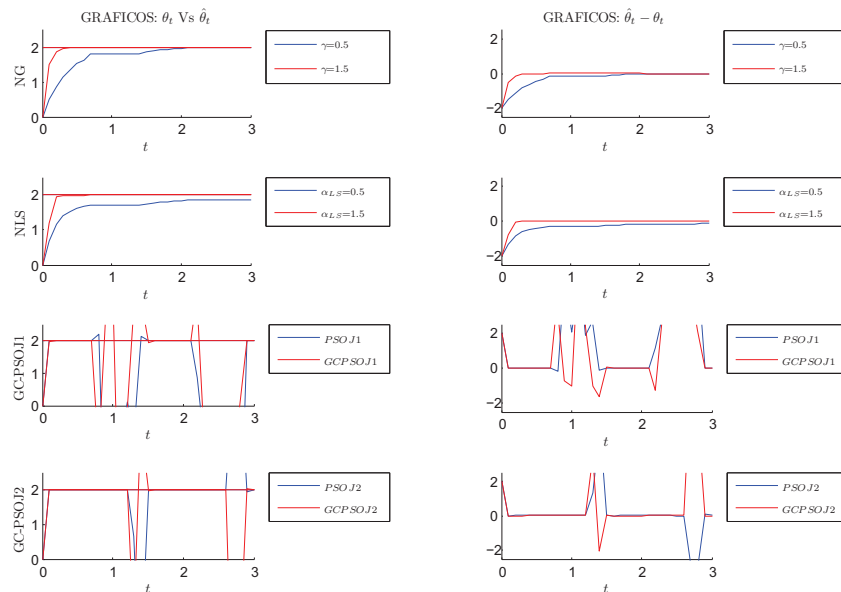


Figura 4.13.: Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para NG, NLS, GC-PSOJ1 y GC-PSOJ2. Señal de entrada utilizada: Tipo III.

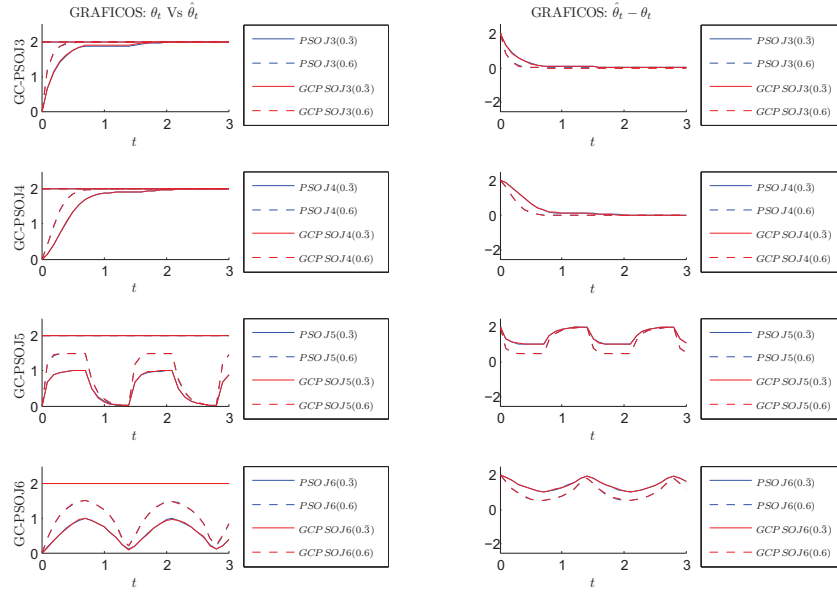


Figura 4.14.: Resultados de simulación para la Subsubsección 4.4.1.1: Estimación y error paramétrico para GC-PSOJ3, GC-PSOJ4, GC-PSOJ5 y GC-PSOJ6. Señal de entrada utilizada: Tipo III.

4.4.1.2. Identificación: Caso estático con retardo simple.

Sea el problema de identificación, cuya planta a identificar tiene la forma

$$x(t) = b_1 u_1(t-1) + b_2 u_2(t-1),$$

para la cual el modelo de identificación es escogido como

$$\hat{x}(t) = \hat{b}_1(t) u_1(t-1) + \hat{b}_2(t) u_2(t-1).$$

Definiendo el error como $e(t) = \hat{x}(t) - x(t)$ se obtiene el Modelo de Error 1 (Ecuación 2.12), donde

$$\phi^T(t) = [\tilde{b}_1(t), \tilde{b}_2(t)] \in \mathbb{R}^2, \quad \omega(t) = [u_1(t-1), u_2(t-1)]^T \in \mathbb{R}^2,$$

$$\tilde{b}_1(t) = \hat{b}_1(t) - b_1, \quad \tilde{b}_2(t) = \hat{b}_2(t) - b_2.$$

La configuración usada en las pruebas es la siguiente:

- Señal de entrada: se tienen como señales de entrada $u_1(t) = 10 + 3\text{sen}(2t)$, y $u_2(t)$ es un tren de pulsos de amplitud 1, período 5[s] y ancho de pulso 50%.
- Parámetros de la planta: $b_1 = 0,6$, $b_2 = 0,2$.
- Configuración de PSO: dado que hay 2 incógnitas, entonces $\lambda = 2$, y con ello $iter_{\max} = 40$, $s = 4$.

- Tiempo de simulación: 20[s].
- Condiciones iniciales: $[x(0), \hat{x}(0)] = [0,8, 0]$, $\hat{\theta}(0) = [0, 0]$.

Los resultados de las simulaciones realizadas para este ejemplo se muestran desde la Figura 4.15 hasta la Figura 4.18.

De las Figura 4.15, Figura 4.16, Figura 4.17 y Figura 4.18 se puede observar que, bajo las configuraciones tenidas en cuenta para las simulaciones, GC-PSO se encuentra bastante bien posicionado entre las metodologías escogidas para hacer la comparación, respecto del error de salida obtenido.

En relación al error paramétrico, GC-PSOJ1 presenta el comportamiento predicho en la Subsección 4.3.1 y no converge a cero, o a otro valor constante diferente de cero. Sin embargo, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4 convergen dentro de la ventana de simulación. Adicionalmente, se nota que todas las estimaciones convergentes con GC-PSO se obtienen transitorios menos agitados que cuando se utiliza NG o NLS con $\alpha = 1,5$, y que GC-PSOJ2(0.6) y GC-PSOJ4(0.6) tienen la más rápida convergencia del error paramétrico a cero sobre todas las demás metodologías.

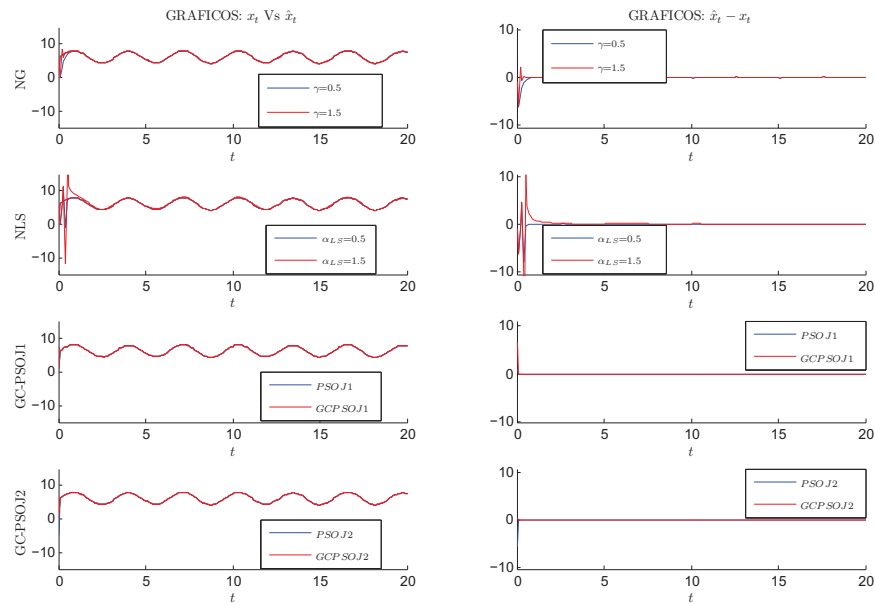


Figura 4.15.: Resultados de simulación para la Subsubsección 4.4.1.2: Comparación de salidas y error de identificación para NG, NLS, GC-PSOJ1 y GC-PSOJ2.

4.4 Resultados experimentales

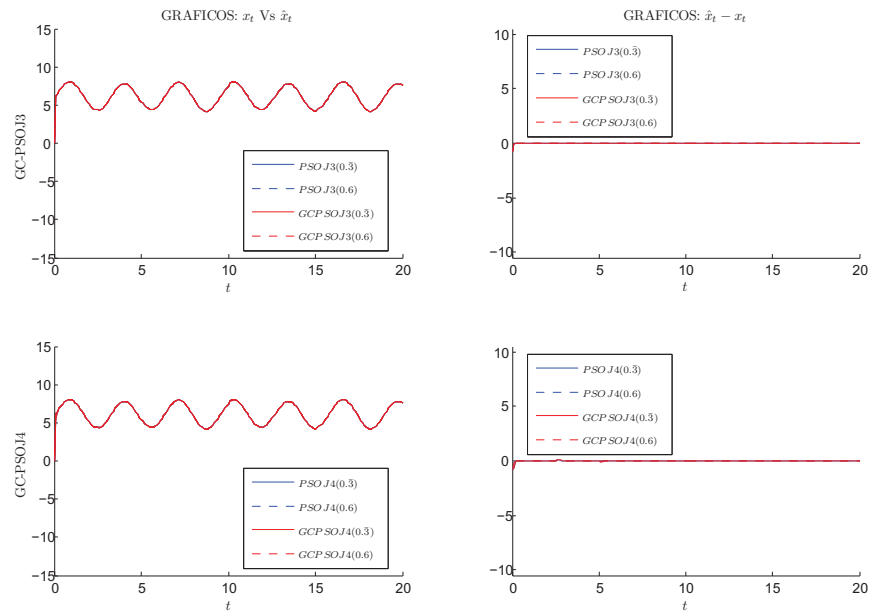


Figura 4.16.: Resultados de simulación para la Subsubsección 4.4.1.2: Comparación de salidas y error de identificación para GC-PSOJ3 y GC-PSOJ4.

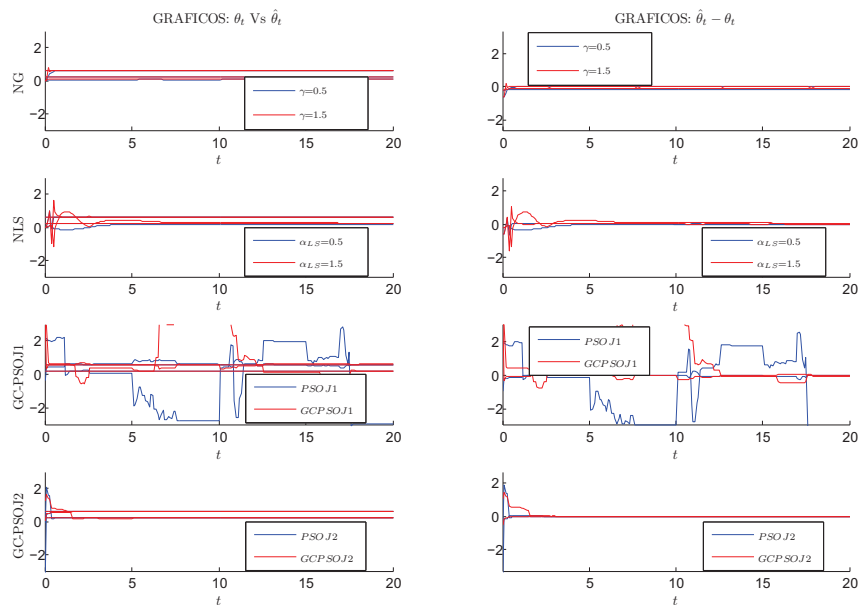


Figura 4.17.: Resultados de simulación para la Subsubsección 4.4.1.2: Estimación y error paramétrico para NG, NLS, GC-PSO J1 y GC-PSO J2.

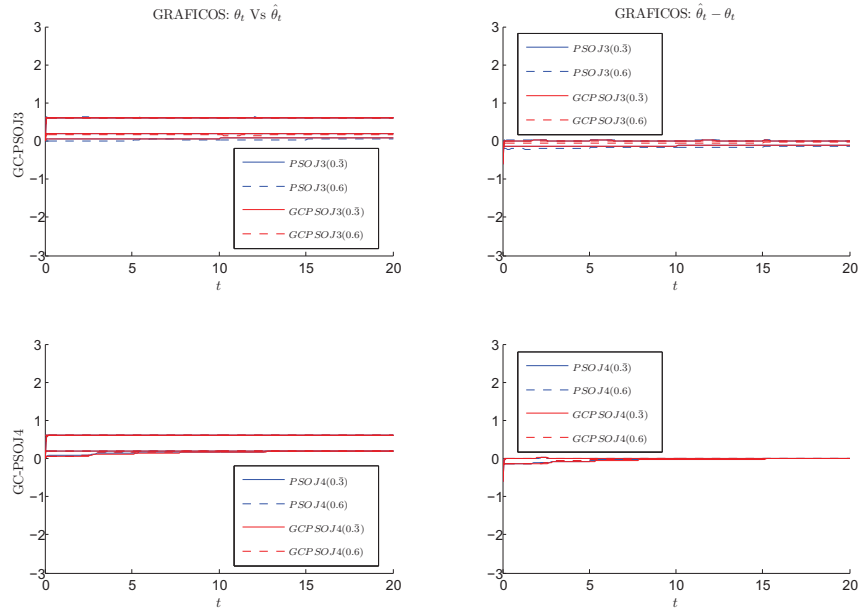


Figura 4.18.: Resultados de simulación para la Subsubsección 4.4.1.2: Estimación y error paramétrico para GC-PSOJ3 y GC-PSOJ4.

De acuerdo a las anteriores observaciones, se concluye que la metodología basada en GC-PSO constituye una interesante y útil herramienta para enfrentar problemas de control adaptable representados por el Modelo de Error 1, ya que el principal objetivo ahí es la convergencia del error de control a cero en vez de la convergencia del error paramétrico. Además, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4 son también adecuados para problemas de identificación, ya que se consigue convergencia paramétrica.

4.4.1.3. Identificación: Caso dinámico.

Sea un problema de identificación con la planta a identificar del tipo

$$x(t) = a_1x(t-1) + a_2x(t-2) + b_1u_1(t-1) + b_2u_1(t-2),$$

y el modelo de identificación es una copia de la planta, pero usando las estimaciones de los parámetros óptimos de la forma

$$\hat{x}(t) = \hat{a}_1(t)x(t-1) + \hat{a}_2(t)x(t-2) + \hat{b}_1(t)u_1(t-1) + \hat{b}_2(t)u_1(t-2).$$

La idea de utilizar $x(t-1)$ y $x(t-2)$ en el modelo de identificación en lugar de $\hat{x}(t-1)$ y $\hat{x}(t-2)$ es precisamente obtener un Modelo del Error Tipo 1, además de aprovechar tales entradas que están disponibles y así mejorar el proceso de identificación. Entonces, al definir $e(t) = \hat{x}(t) - x(t)$ se obtiene (Ecuación 2.12), donde

$$\phi^T(t) = [\tilde{a}_1(t), \tilde{a}_2(t), \tilde{b}_1(t), \tilde{b}_2(t)] \in \mathfrak{R}^4,$$

$$\omega(t) = [x(t-1), x(t-2), u_1(t-1), u_1(t-2)]^T \in \mathbb{R}^4,$$

$$\tilde{a}_1(t) = \hat{a}_1(t) - a_1, \quad \tilde{a}_2(t) = \hat{a}_2(t) - a_2,$$

$$\tilde{b}_1(t) = \hat{b}_1(t) - b_1, \quad \tilde{b}_2(t) = \hat{b}_2(t) - b_2.$$

La configuración utilizada en las pruebas es la siguiente:

- Señal de entrada: tiene la forma $u_1(t) = u_{1,1}(t) + u_{1,2}(t)$, donde $u_{1,1} = 3 + 0,5\text{sen}(0,3t) + 0,5\text{sen}(1,7t) + \text{sen}(t)$, y $u_{1,2}(t)$ es un tren de pulsos de amplitud 1, período 3[s] y ancho de pulso 50%.
- Parámetros de la planta: $a_1 = 0,5$, $a_2 = 0,3$, $b_1 = 0,1$, $b_2 = 0,2$.
- Configuración de PSO: dado que hay 4 incógnitas, entonces $\lambda = 4$ y con ello $iter_{\max} = 80$, $s = 8$.
- Tiempo de simulación: 1000[s].
- Condiciones iniciales: $[x(0), \hat{x}(0)] = [0,8, 0]$, $\hat{\theta}(0) = [0, 0]$.

Los resultados de las simulaciones realizadas para este ejemplo se muestran en las Figura 4.19, Figura 4.20, Figura 4.21 y Figura 4.22, entre los cuales se omite el uso de GC-PSOJ1, debido a que la variación paramétrica es comparativamente muy grande, a pesar de que ofrece la más rápida convergencia del error de identificación.

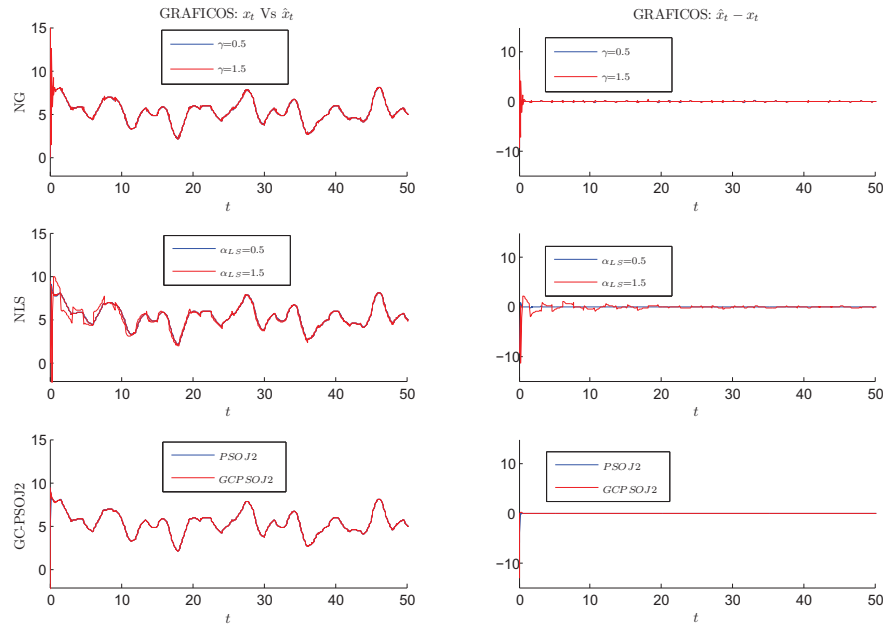


Figura 4.19.: Resultados de simulación para la Subsubsección 4.4.1.3: Comparación de salidas y error de identificación para NG, NLS y GC-PSOJ2.

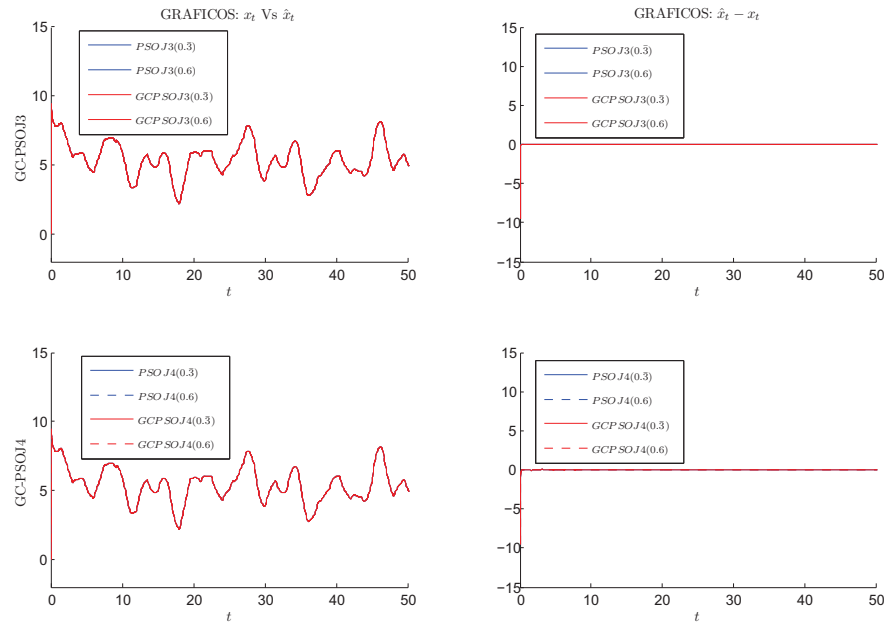


Figura 4.20.: Resultados de simulación para la Subsubsección 4.4.1.3: Comparación de salidas y error de identificación para GC-PSOJ3 y GC-PSOJ4.

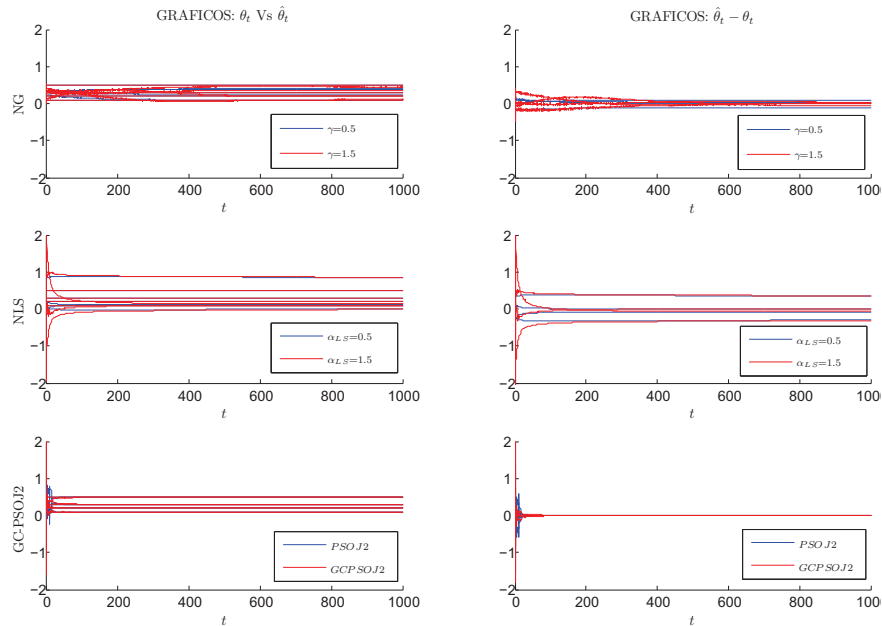


Figura 4.21.: Resultados de simulación para la Subsubsección 4.4.1.3: Estimación y error paramétrico para NG, NLS y GC-PSOJ2.

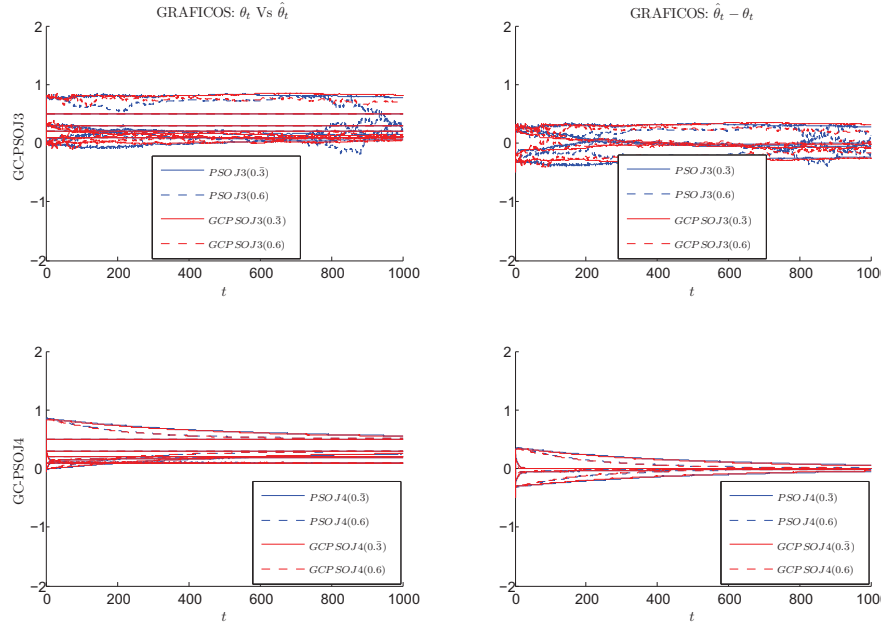


Figura 4.22.: Resultados de simulación para la Subsubsección 4.4.1.3: Estimación y error paramétrico para GC-PSOJ3 y GC-PSOJ4.

Al analizar las Figura 4.19 y Figura 4.20, se observa que la tasa de convergencia a cero del error de identificación es más rápida al usar la metodología basada en PSO. Respecto de la convergencia paramétrica, GC-PSOJ2 presenta la más rápida convergencia con pocas variaciones. Sin embargo, otras simulaciones mostraron que en el caso de que el número de parámetros a ser estimados ($N = 4$) es más grande que la ventana de minimización (por ejemplo $M_1 = 3$), la calidad del transitorio presenta variaciones bastante grandes; esta situación es similar al caso de penalización por sobreparametrización al usar un Criterio de Información de Akaike (Burnham and Anderson, 2004), pero aquí es beneficioso favorecer el incremento de los datos a usar (incrementar M_1) en vez de reducir el número de parámetros, ya que N es fijo. Por otra parte, GC-PSOJ3(0.6) y GC-PSOJ4(0.6) presentan velocidad lenta en la convergencia paramétrica, pero en todo caso siendo GC-PSOJ4 más rápida que NG o NLS, excepto para NG con $\gamma = 0,5$, $\alpha_{LS} = 0,5$.

Por lo tanto, al igual que en Subsubsección 4.4.1.2, queda claro que cualquier configuración de la metodología basada en PSO puede ser usada en problemas de control adaptable, mientras que GC-PSOJ2 y GC-PSOJ4 son también adecuados para problemas de identificación cuando el problema puede ser representado por el Modelo del Error 1.

4.4.1.4. Control: MRAC Indirecto.

En este caso se analiza un problema de control MRAC Indirecto escalar (Ioannou and Fidan, 2006). Para ello se utiliza la planta

$$x(t) = ax(t-1) + bu_1(t-1),$$

y el modelo de referencia

$$x_m(t) = a_m x_m(t-1) + b_m r(t-1),$$

con una señal de control del tipo

$$u(t) = -Kx(t) + Lr(t).$$

Con esto, el error de control viene dado por

$$e_c(t) = x(t) - x_m(t),$$

cuyo modelo de evolución es

$$\begin{aligned} e_c(t) &= ax(t-1) - b(Kx(t-1) + Lr(t-1)) - a_m x_m(t-1) - b_m r(t-1), \\ \Rightarrow e_c(t) &= a_m e(t-1) + (a - a_m - bK)x(t-1) + (bL - b_m)r(t-1). \end{aligned}$$

Entonces, escogiendo

$$K = \frac{a - a_m}{b}, \quad L = \frac{b_m}{b},$$

se consigue

$$e_c(t) = a_m e(t-1),$$

y con $|a_m| < 1$ se asegura la convergencia del error de control a 0 (Ioannou and Fidan, 2006). Sin embargo, dado que en este tipo de problemas generalmente el valor de a y b es desconocido o inexacto, es necesario hacer una identificación de la planta. Por lo tanto, se propone un modelo de identificación de la forma

$$\hat{x}(t) = \hat{a}(t-1)x(t-1) + \hat{b}(t-1)u_1(t-1),$$

y definiendo el error de identificación como

$$e_i(t) = \hat{x}(t) - x(t),$$

se llega entonces al Modelo de Error 1 (Ecuación 2.12) de la forma

$$e_i(t) = \phi^T(t)\omega(t-1),$$

con

$$\phi^T(t) = [\tilde{a}(t), \tilde{b}(t)], \quad \omega^T(t) = [x(t), u_1(t)],$$

$$\tilde{a}(t) = \hat{a}(t) - a, \quad \tilde{b}(t) = \hat{b}(t) - b.$$

De esta forma, los parámetros estimados del controlador pueden ser obtenidos por medio de

$$\hat{K}(t) = \frac{\hat{a}(t) - a_m}{\hat{b}(t)}, \quad \hat{L}(t-1) = \frac{b_m}{\hat{b}},$$

lo cual impone una restricción sobre $\hat{b}(t)$ del tipo $|\hat{b}(t)| > b_o$, para evitar que tenga como valor 0.

Para las pruebas realizadas se utiliza la siguiente configuración:

- Señal de entrada: $r(t) = \sin(2t) + \sin(3t)$,
- Parámetros de la planta: $a = 0,5$, $b = -2$.
- Parámetros del modelo de referencia: $a_m = -0,2$, $b_m = 0,5$.
- Umbrales: $b_0 = 0,2$.
- Configuración de PSO: dado que hay 2 incógnitas, entonces $\lambda = 2$, y con ello $iter_{\max} = 40$, $s = 4$.
- Tiempo de simulación 20[s].
- Condiciones iniciales: $[x(0), \hat{x}(0)] = [10, 0]$, $\hat{\theta}(0) = [0, 0, 2]$.

Los resultados de los experimentos se presentan a continuación, desde la Figura 4.23 hasta la Figura 4.30. Allí, debido a que con NLS se obtienen resultados inestables para $\alpha_{LS} = 1,5$, este valor fue reducido a $\alpha_{LS} = 1$.

Para la etapa de estimación (Figura 4.23, Figura 4.24, Figura 4.25 y Figura 4.26), los resultados son similares a la subsección anterior: un mejor desempeño para las metodologías basadas en PSO respecto del tiempo de convergencia del error de salida y error paramétrico a cero. Además, debido a que sólo se deben estimar 2 parámetros, la convergencia paramétrica presenta muchas menos oscilaciones, resultando que tanto GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4 siempre son mejores que NG o NLS.

4.4 Resultados experimentales

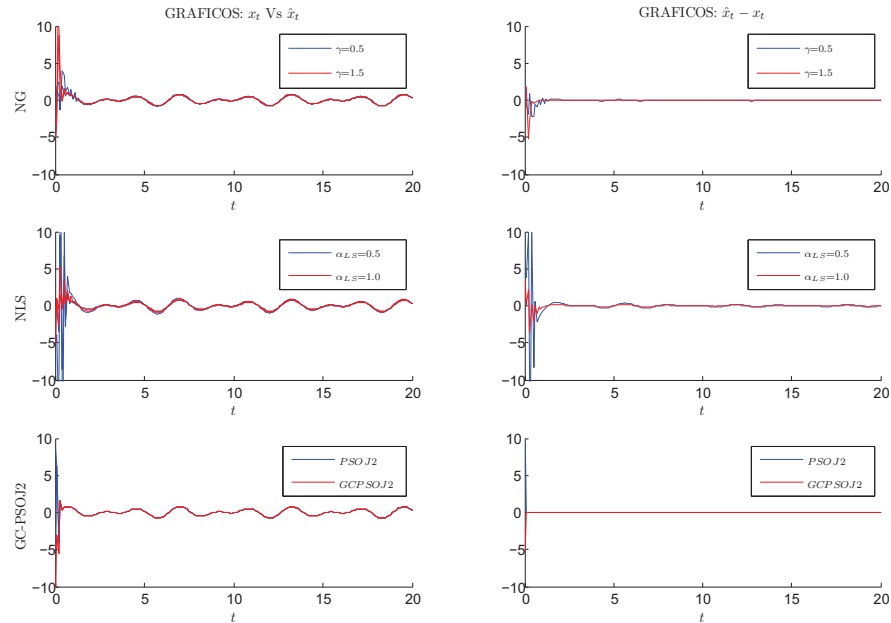


Figura 4.23.: Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas estimada y real, y error de identificación para NG, NLS y GC-PSOJ2.

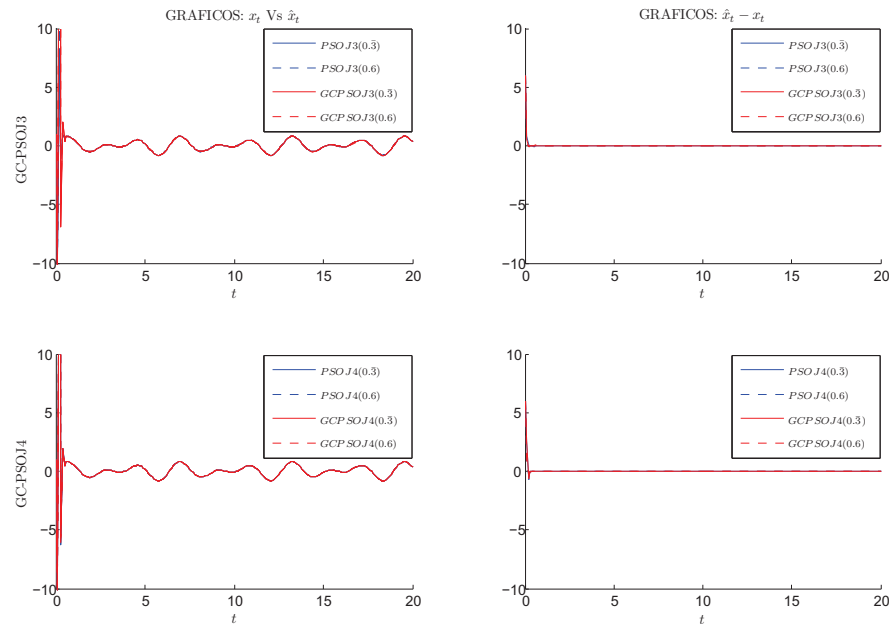


Figura 4.24.: Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas estimada y real, y error de identificación para GC-PSOJ3 y GC-PSOJ4.

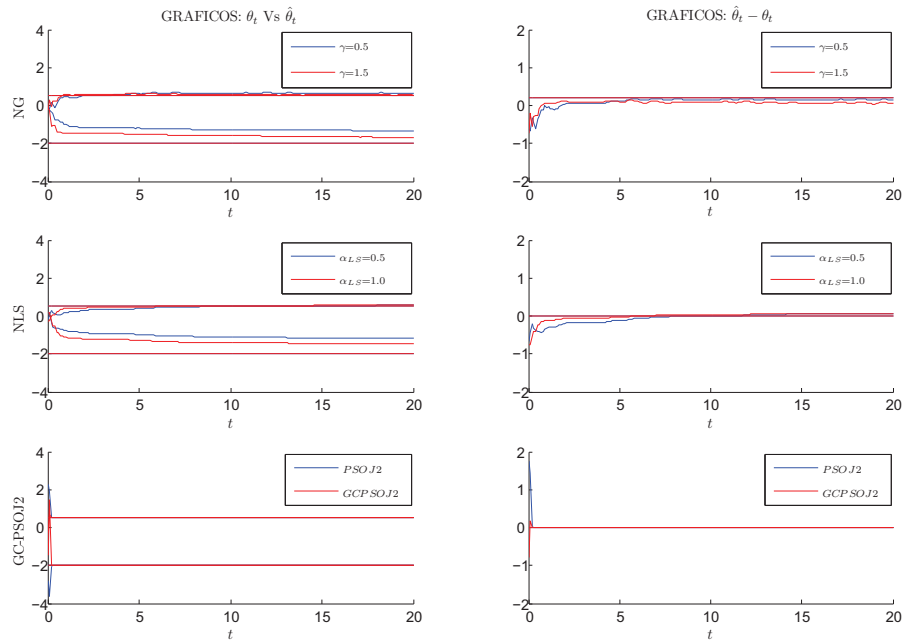


Figura 4.25.: Resultados de simulación para la Subsubsección 4.4.1.4: Estimación de los parámetros de la planta y error paramétrico para NG, NLS y GC-PSOJ2.

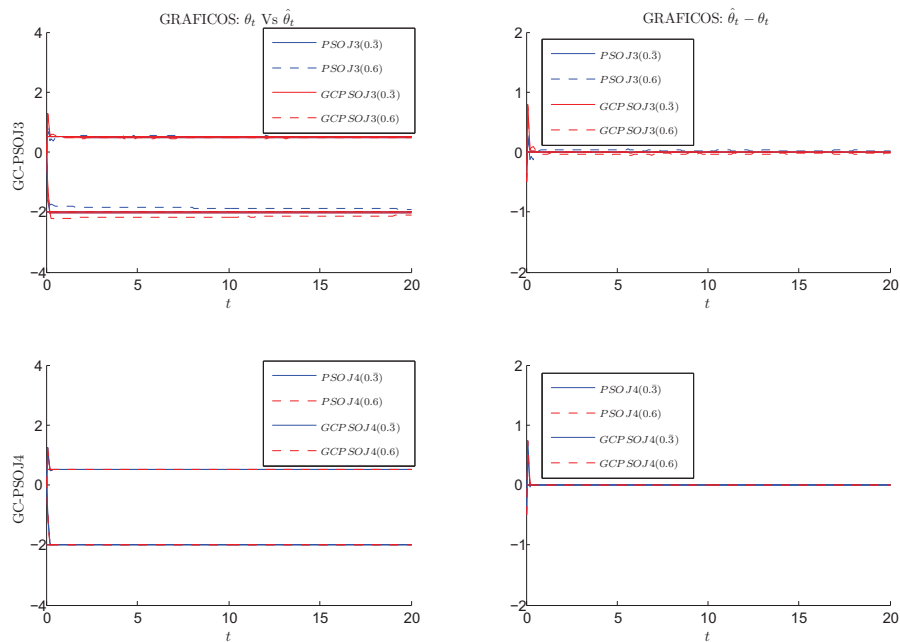


Figura 4.26.: Resultados de simulación para la Subsubsección 4.4.1.4: Estimación de los parámetros de la planta y error paramétrico para GC-PSOJ3 y GC-PSOJ4.

Para la etapa de control (Figura 4.27, Figura 4.28, Figura 4.29 y Figura 4.30) se tiene que, gracias a las bondades obtenidas en la etapa de identificación, las metodologías basadas en PSO siguen presentando mejor desempeño comparativo. Es de importancia aclarar que, aunque con PSO se encuentran algunos picos en el error de control de amplitud considerable, estos son menores en amplitud y cantidad que al usar NG o NLS.

En cuanto a la convergencia paramétrica, para NG y NLS no se da en la ventana de simulación, mostrando además características de convergencia lenta. En el caso de GC-PSO, aunque GC-PSOJ3 apenas alcanza a converger en los 20[s], todas las metodologías presentan convergencia rápida, teniendo para GC-PSOJ2 y GC-PSOJ4 las mejores características.

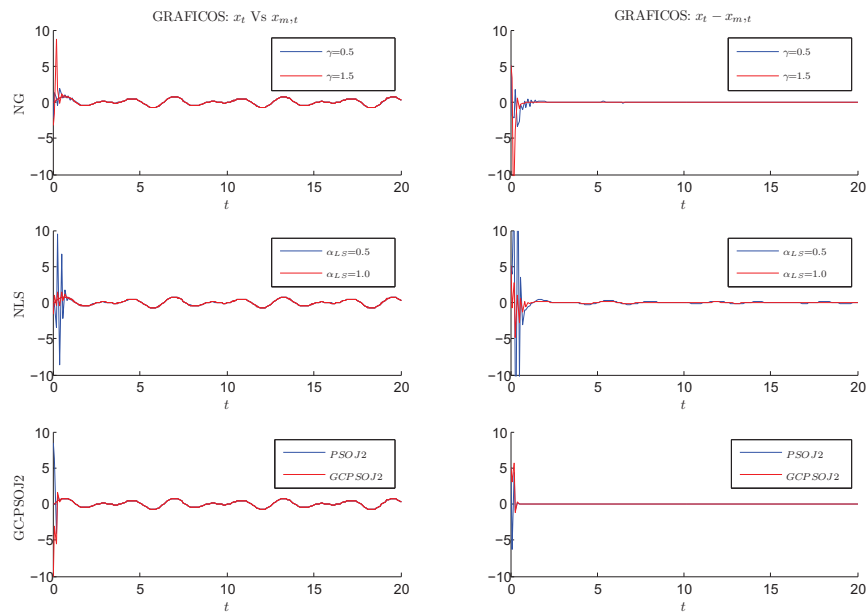


Figura 4.27.: Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas real y del modelo de referencia, y error de control para NG, NLS y GC-PSOJ2.

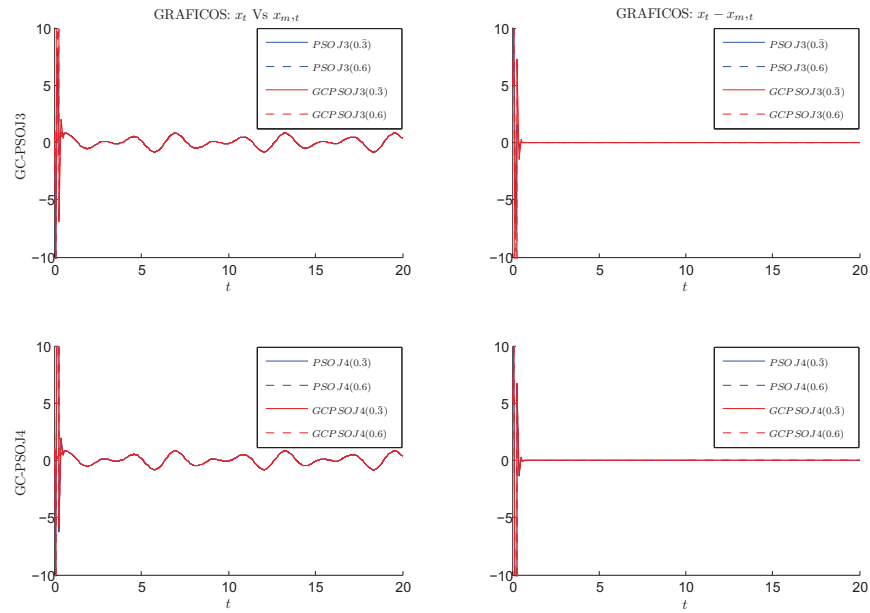


Figura 4.28.: Resultados de simulación para la Subsubsección 4.4.1.4: Comparación de salidas real y del modelo de referencia, y error de control para GC-PSOJ3 y GC-PSOJ4.

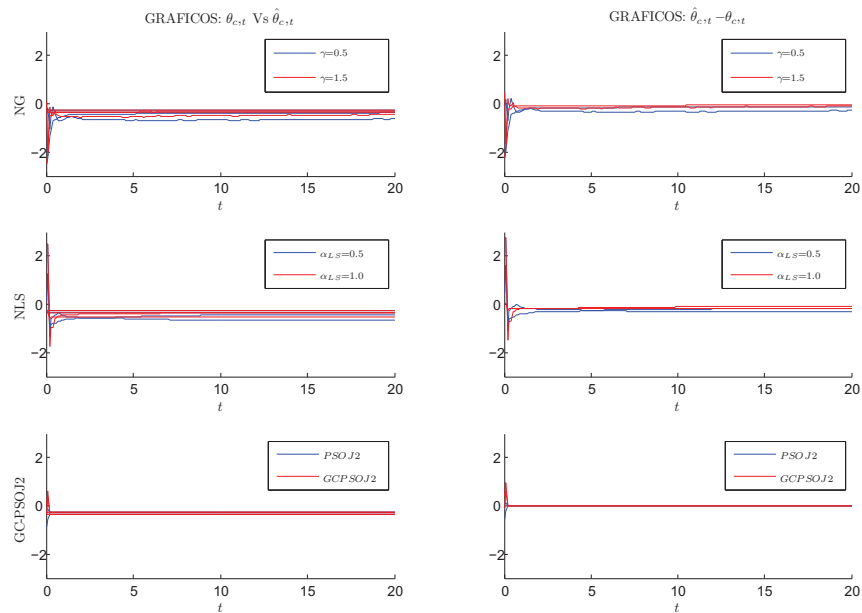


Figura 4.29.: Resultados de simulación para la Subsubsección 4.4.1.4: Cálculo de los parámetros del controlador y su error paramétrico para NG, NLS y GC-PSOJ2.

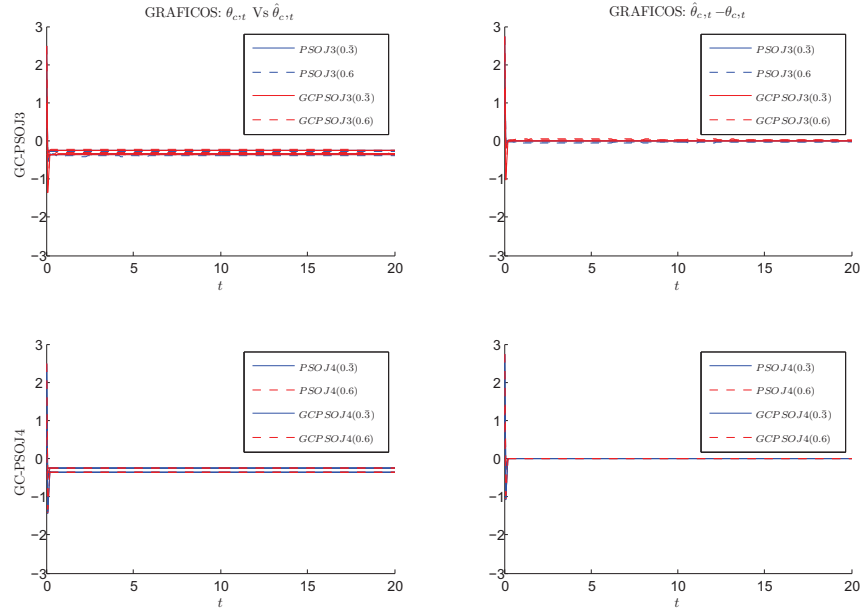


Figura 4.30.: Resultados de simulación para la Subsubsección 4.4.1.4: Cálculo de los parámetros del controlador y su error paramétrico para GC-PSOJ3 y GC-PSOJ4.

4.4.2. Modelo de Error 2

Ahora tienen lugar las pruebas realizadas para el caso del Modelo de Error 2.

4.4.2.1. Identificación: caso escalar.

Sea un problema de identificación con la planta a identificar del tipo

$$x(t) = ax(t-1) + bu(t),$$

para el cuál se utiliza el modelo de identificación

$$\hat{x}(t) = a_m \hat{x}(t-1) + (\hat{a}(t-1) - a_m) x(t-1) + \hat{b}(t-1) u(t-1).$$

A partir de lo anterior, definiendo $e(t) = \hat{x}(t) - x(t)$, se llega al Modelo de Error 2 (Ecuación 2.17) de la forma

$$e(t) = a_m e(t-1) + \phi^T(t-1) \omega(t-1),$$

con el cual se obtiene un equivalente al Modelo de Error 1 (Ecuación 2.12) de la forma

$$\epsilon(t) = e(t) - a_m e(t-1) = \phi^T(t-1) \omega(t-1),$$

con

$$\phi^T(t-1) = [\tilde{a}(t-1), \tilde{b}(t-1)], \quad \omega^T(t-1) = [x(t-1), u(t-1)],$$

$$\tilde{a}(t-1) = \hat{a}(t-1) - a, \quad \tilde{b}(t-1) = \hat{b}(t-1) - b.$$

Para las pruebas realizadas se utiliza la siguiente configuración:

- Señal de entrada: tiene la forma $u_1(t) = u_{1,1}(t) + u_{1,2}(t)$, donde $u_{1,1} = 3 + 0,5\text{sen}(0,3t) + 0,5\text{sen}(1,7t) + \text{sen}(t)$, y $u_{1,2}(t)$ es un tren de pulsos de amplitud 1, período 3[s] y ancho de pulso 50%.
- Parámetros de la planta: $a_1 = 0,5$, $b_1 = 0,1$.
- Configuración de PSO: dado que hay 2 incógnitas, entonces $\lambda = 2$, y con ello $iter_{\max} = 40$, $s = 4$.
- Condiciones iniciales: $[x(0), \hat{x}(0)] = [0,8, 0]$, $\hat{\theta}(0) = [0, 0]$.
- Parámetros del modelo de identificación: $a_m = -0,8$.
- Tiempo de simulación: 1000[s].

Los resultados de los experimentos se presentan a continuación, desde la Figura 4.31 hasta la Figura 4.35. Nuevamente, debido a que con NLS se obtienen resultados inestables con $\alpha_{LS} = 1,5$, este valor fue reducido a $\alpha_{LS} = 1$.

Desde el punto de vista del error de identificación, la metodología NG tiene ciertas similitudes con la metodología GC-PSOJ2; no obstante, GC-PSOJ2 presenta menos oscilaciones. Por otra parte, a pesar de que los máximos picos en el error de identificación obtenidos para NLS son menores que para NG y GC-PSOJ2, la convergencia a cero se ve desmejorada por una serie de oscilaciones constantes que aparecen periódicamente. Para el caso de GC-PSOJ3 y GC-PSOJ4, se obtiene que estas metodologías ofrecen la mejor convergencia del error de identificación, en cuanto a velocidad de convergencia a cero, y amplitud y número oscilaciones en el transitorio.

Desde el punto de vista del error paramétrico, la situación se vuelve aún más favorable para las metodologías basadas en PSO. Aquí, la convergencia paramétrica se da muy tempranamente para GC-PSOJ2 y GC-PSOJ4, seguidas de NG, y aunque un poco más lenta para GC-PSOJ3 también se alcanza la convergencia en la ventana de simulación. El caso de NLS ofrece buenos resultados para $\alpha_{LS} = 0,5$, mostrándose mejor que NG, pero no así para $\alpha_{LS} = 1$ que ni siquiera consigue convergencia en la ventana simulada.

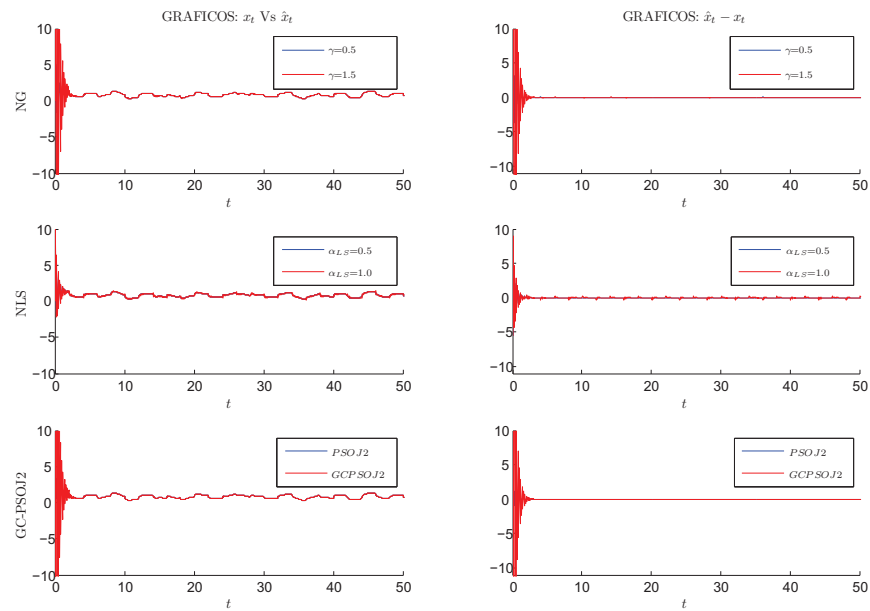


Figura 4.31.: Resultados de simulación para la Subsubsección 4.4.2.1: Comparación de las salidas real y estimada, y error de identificación para NG, NLS y GC-PSOJ2.

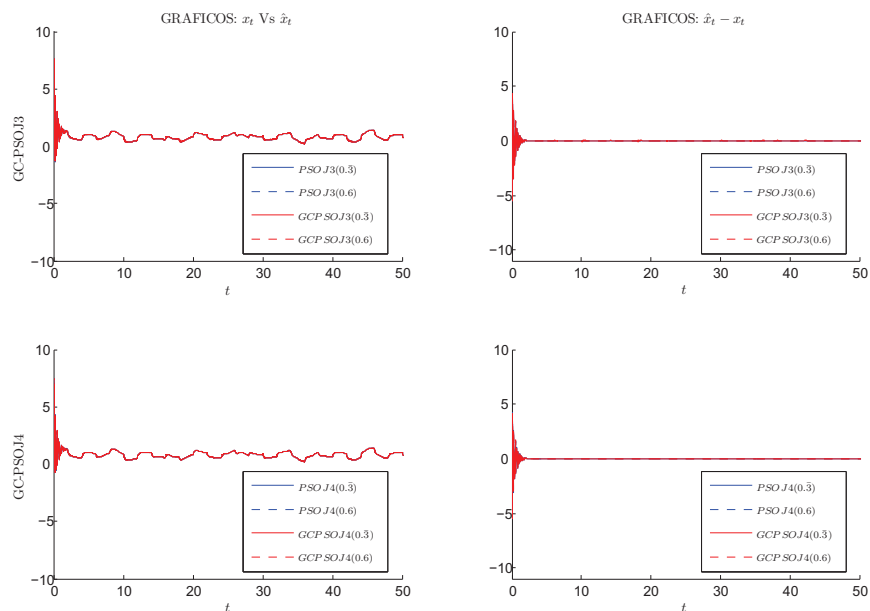


Figura 4.32.: Resultados de simulación para la Subsubsección 4.4.2.1: Comparación de las salidas real y estimada, y error de identificación para GC-PSOJ3 y GC-PSOJ4.

4.4 Resultados experimentales

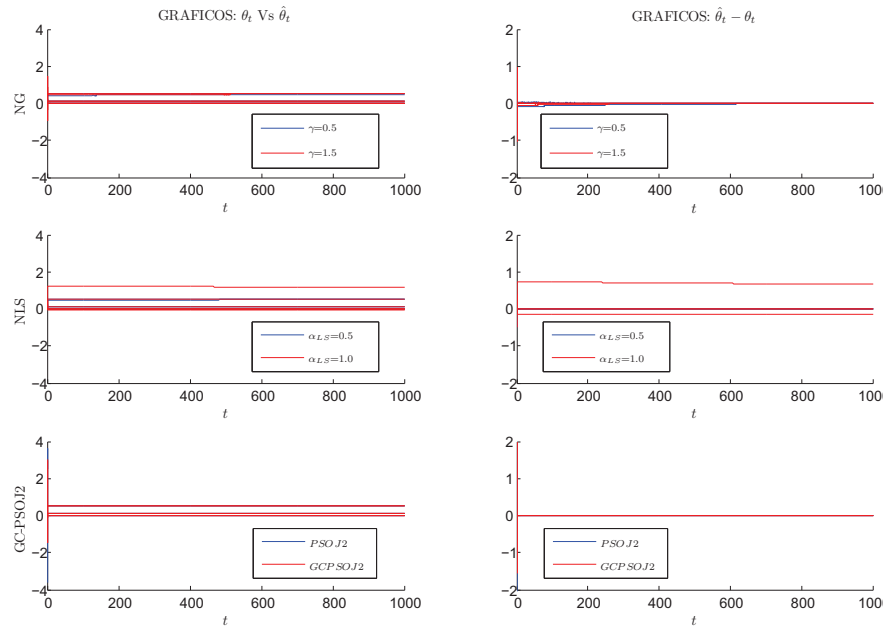


Figura 4.33.: Resultados de simulación para la Subsubsección 4.4.2.1: Estimación de los parámetros de la planta y error paramétrico para NG, NLS y GC-PSOJ2.

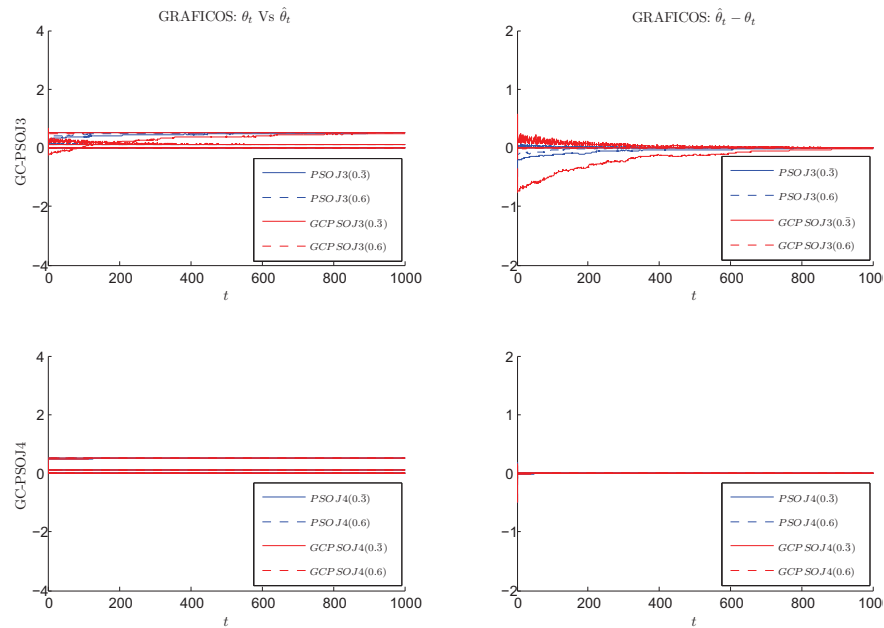


Figura 4.34.: Resultados de simulación para la Subsubsección 4.4.2.1: Estimación de los parámetros de la planta y error paramétrico para GC-PSOJ3 y GC-PSOJ4.

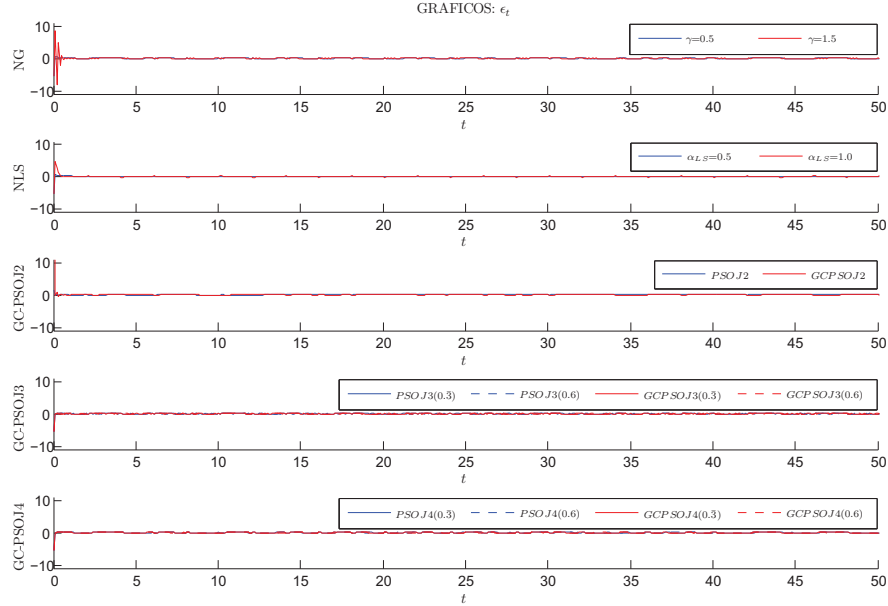


Figura 4.35.: Resultados de simulación para la Subsubsección 4.4.2.1: Evolución de $\epsilon(t)$ para NG, NLS, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4.

Finalmente, y no menos importante, la Figura 4.35 muestra la evolución de $\epsilon(t)$, que es la variable directa de minimización al utilizar las metodologías basadas en PSO. Es importante recordar de que, si bien con GC-PSO no se minimiza directamente el error de identificación (pues se utiliza para la minimización un Modelo de Error 1 equivalente cuya variable de optimización es $\epsilon(t)$), esto es conseguido como consecuencia de la buena minimización de $\epsilon(t)$. No obstante, aquí se verifican una vez más las bondades de la metodología propuesta, que en todos los casos logra un mejor desempeño en la minimización de $\epsilon(t)$ que NG y NLS.

4.4.2.2. MRAC Combinado

El enfoque MRAC Combinado (Duarte and Narendra, 1989; Duarte and Ponce, 1997) es similar al caso de MRAC Indirecto, sólo que se añade una etapa de optimización para mejorar la estimación de los parámetros del controlador, partiendo de la premisa de que existen mediciones que contienen información aprovechable al respecto ($r(t)$ es dada y medible).

Sea entonces un problema de control MRAC Combinado, para lo cual se utiliza una planta del tipo

$$x(t) = ax(t-1) + bu_1(t-1),$$

y el modelo de referencia

$$x_m(t) = a_mx_m(t-1) + b_mr(t-1),$$

con una señal de control del tipo

$$u(t) = -K(t)x(t) + L(t)r(t).$$

Con esto, el error de control viene dado por

$$e_c(t) = x(t) - x_m(t),$$

cuyo modelo de evolución (Modelo de Error de tipo 2) es

$$e_c(t) = ax(t-1) - b(K(t-1)x(t-1) + L(t-1)r(t-1)) - a_mx_m(t-1) - b_mr(t-1),$$

$$e_c(t) = a_me_c(t-1) + (a - a_m - bK(t-1))x(t-1) + (bL(t-1) - b_m)r(t-1),$$

es decir,

$$e_c(t) = a_me_c(t-1) + \underbrace{(a - a_m - bK(t-1))}_{K_1(t-1)}x(t-1) + \underbrace{(bL(t-1) - b_m)}_{L_1(t-1)}r(t-1),$$

que representa

$$e_c(t) = a_me_c(t-1) + \theta_c^T(t)\omega_c(t-1),$$

con

$$\theta_c^T(t) = [K_1(t), L_1(t)], \quad \omega_c(t) = [x(t), r(t)].$$

Proponiendo un estimador del error de control de la forma

$$\hat{e}_c(t) = a_m\hat{e}_c(t-1) + \hat{\theta}_c^T(t)\omega_c(t-1),$$

y definiendo $\tilde{e}_c(t) = \hat{e}_c(t) - e_c(t)$ se llega al Modelo de Error 2 con

$$\tilde{e}_c(t) = a_m\tilde{e}_c(t-1) + \phi_c^T(t)\omega_c(t-1),$$

$$\phi_c^T(t) = \hat{\theta}_c^T(t) - \theta_c^T(t),$$

equivalente al Modelo de Error 1

$$\epsilon(t) = \tilde{e}_c(t) - a_m\tilde{e}_c(t-1) = \phi_c^T(t)\omega_c(t-1).$$

Con esto, teniendo que

$$\hat{K}_1(t) = a - a_m - bK(t), \quad \hat{L}_1(t) = bL(t) - b_m,$$

entonces se sigue que

$$K(t) = \frac{a - a_m - \hat{K}_1(t)}{b}, \quad L(t) = \frac{\hat{L}_1(t) + b_m}{b},$$

y con esto

$$e_c(t) = a_m e_c(t-1),$$

haciendo que $e_c(t) \rightarrow 0$ cuando $t \rightarrow \infty$ siempre que $|a_m| < 1$ (ver Duarte and Narendra (1989); Duarte and Ponce (1997)).

Sin embargo, como no se tienen los valores reales de a y b , entonces se propone un modelo de identificación de la forma

$$\hat{x}(t) = \hat{a}(t-1)x(t-1) + \hat{b}(t-1)u_1(t-1),$$

y definiendo el error de identificación como

$$e_i(t) = \hat{x}(t) - x(t),$$

se llega entonces al modelo de error

$$e_i(t) = \phi_i^T(t)\omega_i(t-1),$$

con

$$\phi^T(t) = [\tilde{a}(t), \tilde{b}(t)], \quad \omega^T(t) = [x(t), u_1(t)],$$

$$\tilde{a}(t) = \hat{a}(t) - a, \quad \tilde{b}(t) = \hat{b}(t) - b.$$

Hecho esto, ahora es posible estimar los parámetros del controlador de la forma:

$$\hat{K}(t) = \frac{\hat{a} - a_m - \hat{K}_1(t)}{\hat{b}}, \quad \hat{L}(t) = \frac{\hat{L}_1(t) + b_m}{\hat{b}}, \quad (4.22)$$

lo que obliga otra vez a imponer la restricción $|\hat{b}(t)| > b_0$.

Para las pruebas realizadas se utilizan la siguiente configuración:

- Señal de entrada: tiene la forma $u_1(t) = u_{1,1}(t) + u_{1,2}(t)$, donde $u_{1,1} = 3 + 0,5\text{sen}(0,3t) + 0,5\text{sen}(1,7t) + \text{sen}(t)$, y $u_{1,2}(t)$ es un tren de pulsos de amplitud 1, período 3[s] y ancho de pulso 50%.
- Parámetros de la planta: $a = 0,5$, $b = 0,3$.
- Parámetros del modelo de referencia: $a_m = -0,2$, $b_m = 0,5$.
- Umbrales: $b_0 = 0,2$.
- Configuración de PSO: dado que hay 2 incógnitas en cada proceso de optimización, entonces cada proceso tiene $\lambda = 2$, y con ello $iter_{\max} = 40$, $s = 4$.
- Tiempo de simulación: 50[s].
- Condiciones iniciales: $[x(0), \hat{x}(0)] = [0,8, 0]$, $[\hat{a}(t), \hat{b}(t)] = [0, 0,2]$, $[\hat{K}_1(t), \hat{L}_1(t)] = [0, 0]$.

Los resultados de los experimentos se presentan a continuación, desde la Figura 4.36 hasta la Figura 4.44. Una vez más, debido a que con NLS se obtienen resultados inestables con $\alpha_{LS} = 1,5$, este valor fue reducido a $\alpha_{LS} = 1$.

La optimización en la etapa de identificación es similar que en la Subsubsección 4.4.1.4 (identificación con Modelo de Error 1), con las importantes bondades obtenidas con PSO mostradas en las Figura 4.36, Figura 4.37, Figura 4.38 y Figura 4.39. No obstante, la optimización en la etapa de control está basada en un Modelo de Error 2 con equivalente a Modelo de Error 1, para lo cual es necesario abordar más detalladamente si existen o no beneficios respecto del enfoque MRAC Indirecto.

De la etapa de identificación se tiene que el error de identificación es minimizado mucho más rápidamente al utilizar GC-PSO en cualquiera de sus configuraciones específicas (al rededor de 0,3[s]) que cualquiera de las otras metodologías evaluadas (0,5[s] para NG en el mejor caso, 10[s] para NLS en el mejor caso aunque con oscilaciones).

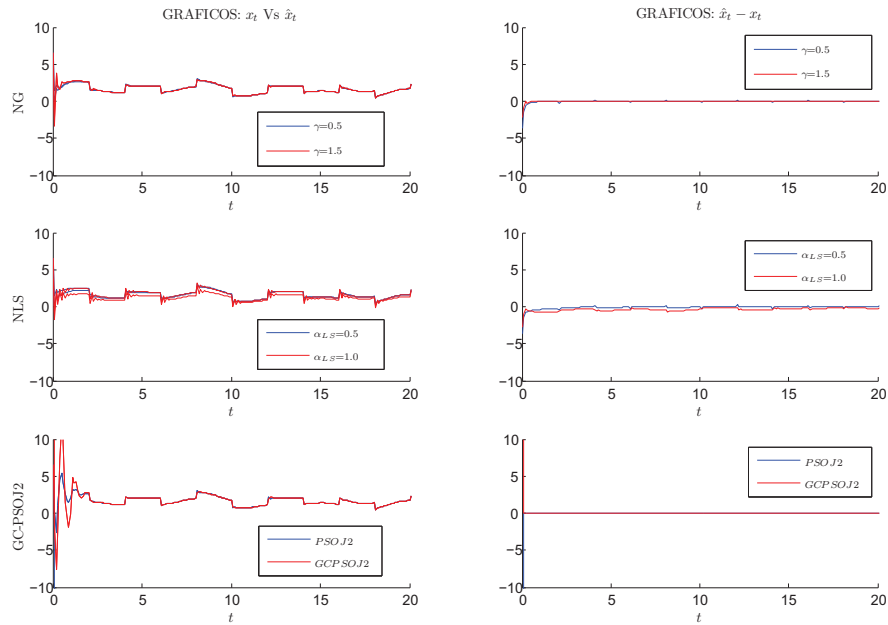


Figura 4.36.: Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas estimada y real, y error de identificación para NG, NLS y GC-PSOJ2.

4.4 Resultados experimentales

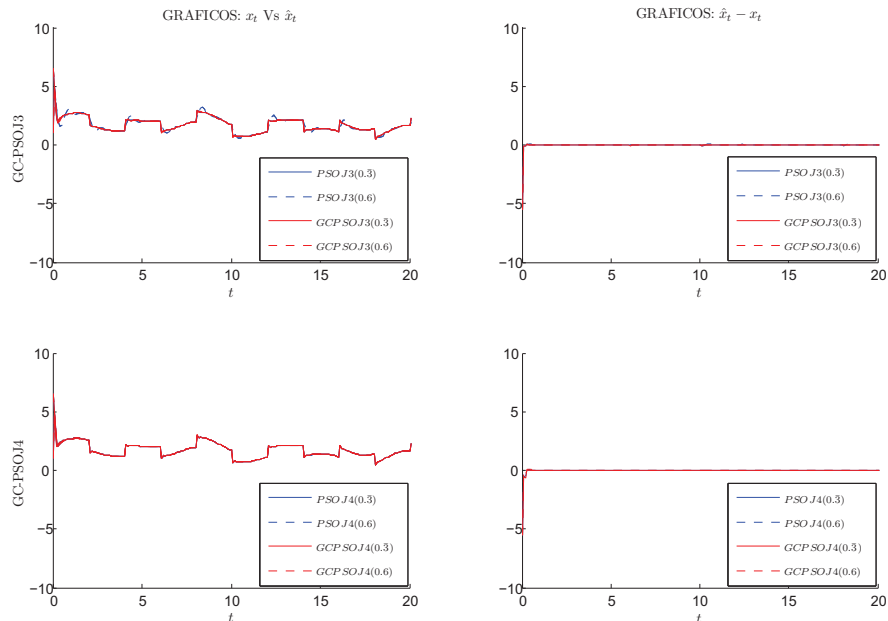


Figura 4.37.: Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas estimada y real, y error de identificación para GC-PSOJ3 y GC-PSOJ4.

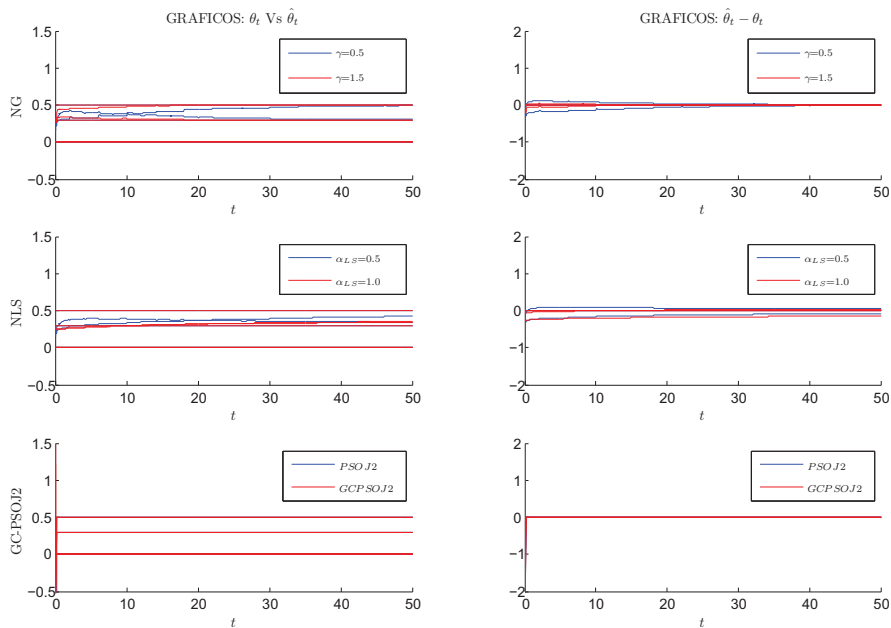


Figura 4.38.: Resultados de simulación para la Subsubsección 4.4.2.2: Estimación de los parámetros de la planta y error paramétrico para NG, NLS y GC-PSOJ2.

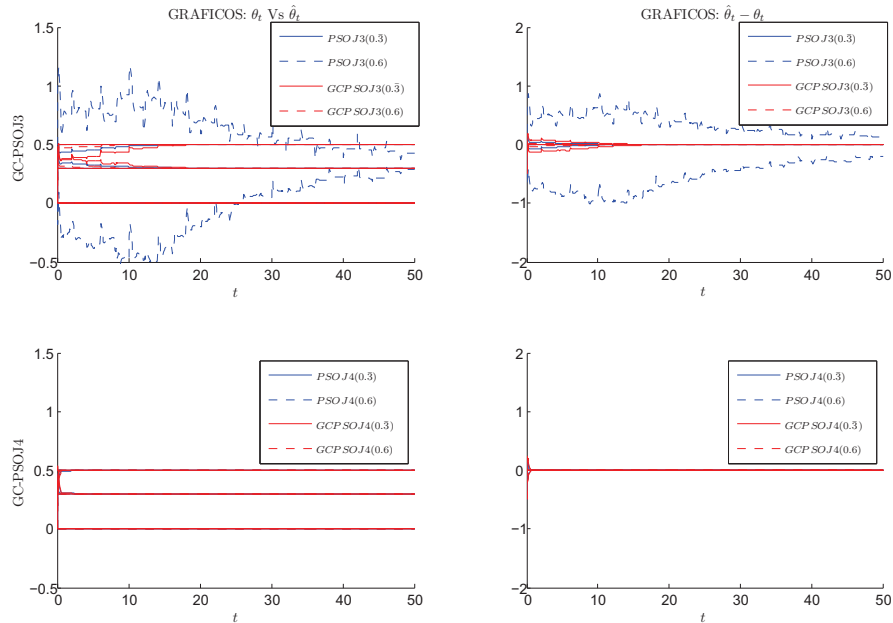


Figura 4.39.: Resultados de simulación para la Subsubsección 4.4.2.2: Estimación de los parámetros de la planta y error paramétrico para GC-PSOJ3 y GC-PSOJ4.

En cuanto al error paramétrico, éste es llevado a cero de forma casi de forma inmediata para GC-PSOJ2 y GC-PSOJ4 (alrededor de 0,2[s] y 0,4[s] respectivamente), y mucho más lenta para GC-PSOJ3(0.3) (alrededor de 9[s]), pero que en todo caso son más rápidas que para NG (10[s] en el mejor caso) y NLS (no se da convergencia del error paramétrico en la ventana de simulación). Por tanto, dadas las características descritas, la etapa de identificación es llevada a cabo con un excelente desempeño (difícilmente mejorable) por parte de GC-PSO.

Ahora, pasamos a hacer el análisis de la etapa de control. Debido a la naturaleza enfoque MRAC Combinado, los parámetros del controlador son dependientes de dos procesos de optimización: el primero, directamente de la minimización del error de control, y el segundo, indirectamente de la minimización del error de identificación. Por lo tanto, el mal desempeño en cualquiera de los procesos de optimización puede verse reflejado de manera más fuerte en el desempeño general. Así, el proceso de control se puede ver muy deteriorado por las fallas en la etapa de identificación, pero no viceversa. Es más, las oscilaciones producidas en el transitorio de la etapa de control, producidas por un seguimiento aún no logrado, son beneficiosas para la identificación porque introducen excitación persistente a la planta, mejorando así la velocidad de convergencia y características del transitorio del error de identificación (puesto que $x(t)$ hace parte del vector de información $\omega(t)$).

4.4 Resultados experimentales

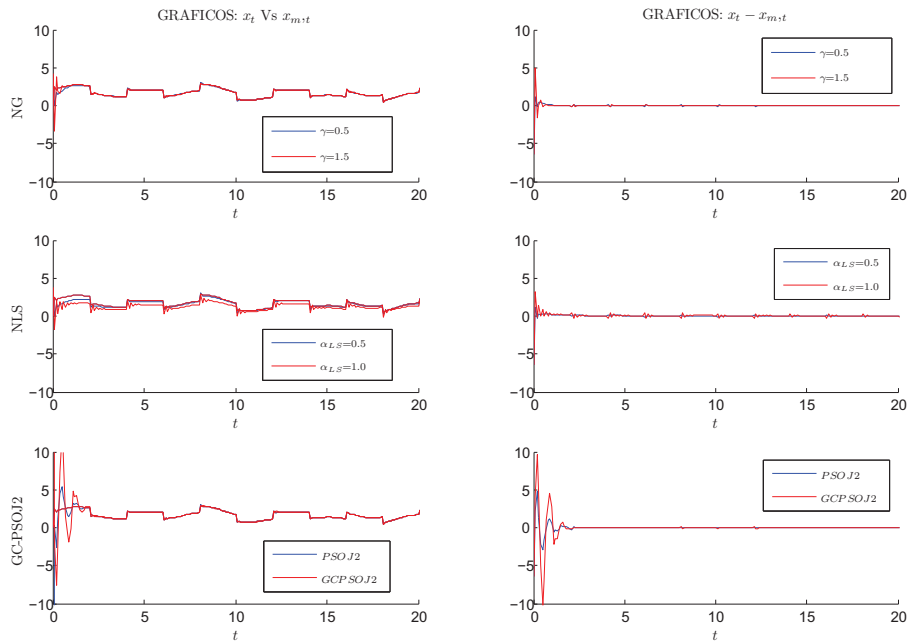


Figura 4.40.: Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas real y del modelo de referencia, y error de control para NG, NLS y GC-PSOJ2.

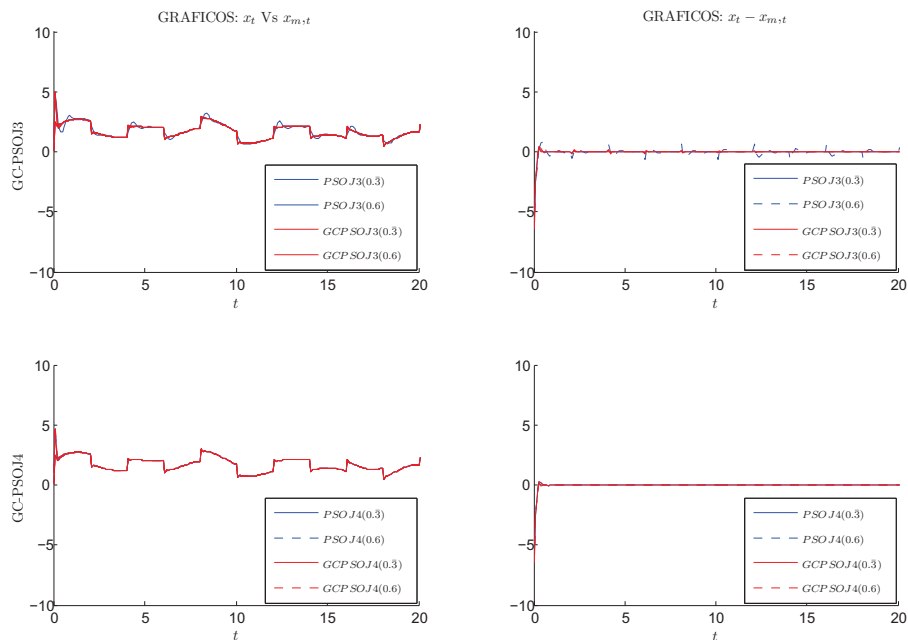


Figura 4.41.: Resultados de simulación para la Subsubsección 4.4.2.2: Comparación de salidas real y del modelo de referencia, y error de control para GC-PSOJ3 y GC-PSOJ4.

Al analizar las Figura 4.40 y Figura 4.41, se observa lo anteriormente descrito. Para el caso de GC-PSOJ2, las oscilaciones iniciales en el seguimiento (entre 0[s] y 2[s]) influyen directa y positivamente en la buena identificación; no obstante, si las oscilaciones son de gran amplitud, en el transitorio de la identificación se verá reflejado esto obteniendo un pico inicial de amplitud considerable en el error de identificación. Para el caso de GC-PSOJ3 y GC-PSOJ4 se obtiene un buen seguimiento aún más temprano, con alrededor de 0,8[s] para GC-PSOJ3(0,3) y GCPSOJ3(0,6) (PSOJ3(0,6) utiliza alrededor de 3[s]), y 0,6[s] para GC-PSOJ4. Por otra parte, debido a la interdependencia de dos procesos de optimización (para identificación y control respectivamente) se producen ciertos picos de muy pequeña amplitud para GC-PSOJ3(0,6), precisamente porque un valor de $\alpha = 0,6$ le da mayor prioridad a la minimización del error de salida que al error paramétrico, y el error paramétrico de la etapa de control es directamente dependiente del error paramétrico de la etapa de identificación (ver Ecuación 4.22). No obstante, respecto de la velocidad de convergencia u ocurrencia de picos después de la convergencia, solamente PSOJ3(0.6) no mejora el desempeño respecto de NG o NLS.

Respecto de NG se tiene una convergencia rápida, pero no es una convergencia sostenida debido a que periódicamente aparecen algunos picos (aunque de amplitud decreciente). En cuanto a NLS, el desempeño es mucho más desmejorado respecto de NG, puesto que los picos periódicos también aparecen pero con amplitud mucho menos decreciente, y teniendo además un transitorio más largo y bastante oscilatorio.

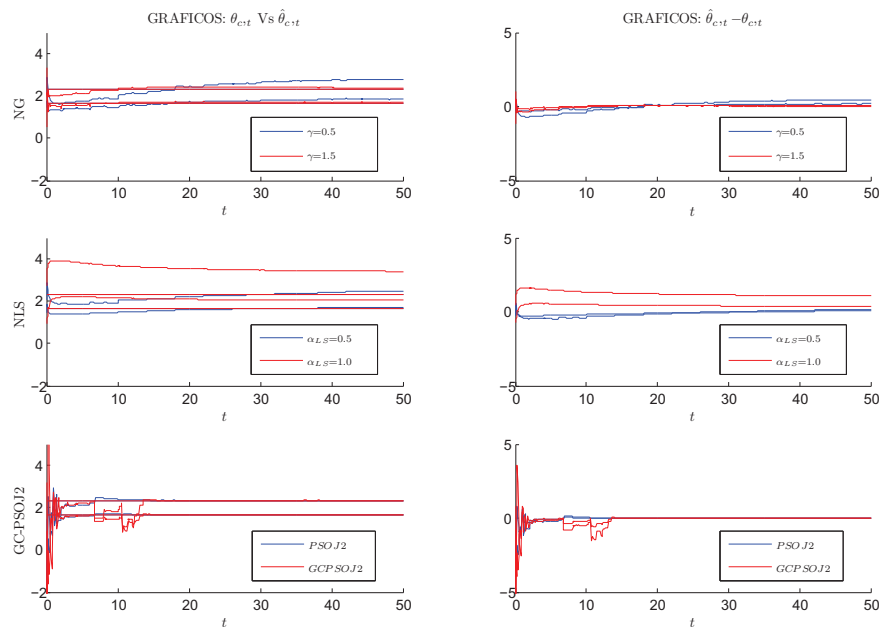


Figura 4.42.: Resultados de simulación para la Subsubsección 4.4.2.2: Cálculo de los parámetros del controlador y su error paramétrico para NG, NLS y GC-PSOJ2.

4.4 Resultados experimentales

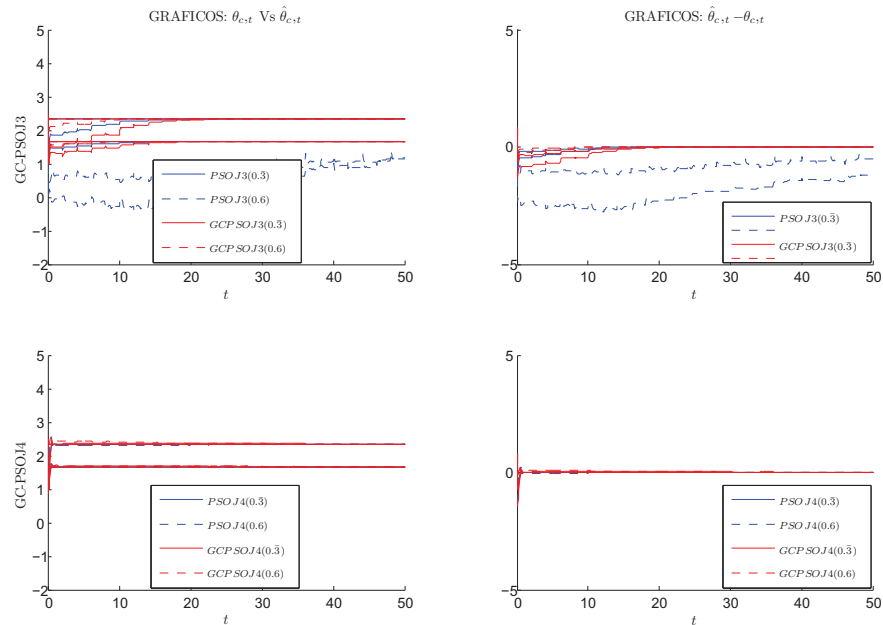


Figura 4.43.: Resultados de simulación para la Subsubsección 4.4.2.2: Cálculo de los parámetros del controlador y su error paramétrico para GC-PSOJ3 y GC-PSOJ4.

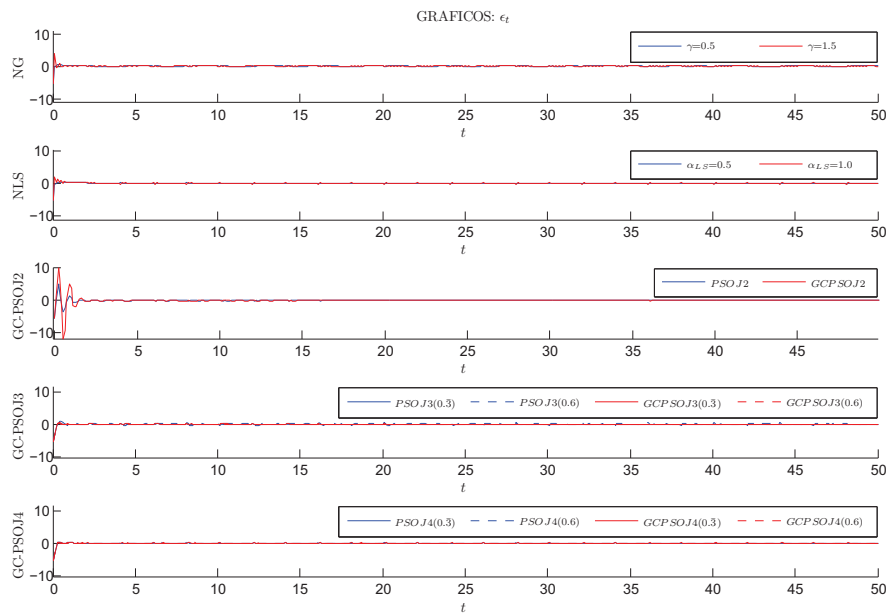


Figura 4.44.: Resultados de simulación para la Subsubsección 4.4.2.2: Evolución de $\epsilon(t)$ para NG, NLS, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4.

Respecto del error paramétrico (Figura 4.42 y Figura 4.43), se hace claro que el desempeño de las

metodologías basadas en PSO son mucho más eficaces y eficientes que NG y NLS, puesto que (a excepción de PSOJ3(0.6)) todas muestran una convergencia a 0 a dentro de la ventana de simulación, con casos altamente eficientes como GC-PSOJ2 y GC-PSOJ4(0.3) que logran convergencia paramétrica en alrededor de 15[s]. Sin embargo, el caso PSOJ6(0.6) aunque presenta convergencia lenta, se muestra que en todo caso es mucho más rápida que NG o NLS, puesto que aunque ninguna de estas tres metodologías presenta convergencia en la ventana de simulación, PSOJ3(0.6) muestra una pronta convergencia después de los 50[s].

Finalmente, al analizar la minimización de $\epsilon(t)$ (Figura 4.44) se aprecia que el mejor desempeño es obtenido para GC-PSOJ4 y GC-PSOJ3(0.3), con lo cual se muestra que la metodología propuesta presenta amplias ventajas y bondades en todos los ámbitos a ser comparados con NG y NLS.

4.4.3. Modelo de Error 3.

Para terminar las pruebas por simulación, se termina con un par de ejemplos para el caso del Modelo de Error 3, en los cuáles solamente se hace la comparación entre GC-PSO y NG.

4.4.3.1. Problema genérico

Sea una representación genérica del Modelo de Error 3 (Subsubsección 2.3.1.3) con

$$\theta^{*T} = [\theta_1^*, \theta_2^*, \theta_3^*, \theta_4^*], \quad \hat{\theta}^T(t) = [\hat{\theta}_1(t), \hat{\theta}_2(t), \hat{\theta}_3(t), \hat{\theta}_4(t)],$$

$$\phi^T(t) = \hat{\theta}^T(t) - \theta^* = [\tilde{\theta}_1(t), \tilde{\theta}_2(t), \tilde{\theta}_3(t), \tilde{\theta}_4(t)],$$

siendo además $W(z) = d + c^T(zI - A)^{-1}b$ una función de transferencia SPR con con

$$A = \begin{bmatrix} -0,5 & 1 \\ -0,4 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0,1 \\ -0,028 \end{bmatrix}, \quad C = [1 \quad 0], \quad D = 0,2,$$

tal que

$$e_1(t) = W(z)[v(t)],$$

y $e_1(t)$ es el error de salida (medible) que se desea minimizar. Entonces, definiendo $Z(t) \doteq e_1(t)$, $\hat{Z}(t) \doteq \hat{e}_1(t)$ y $\epsilon(t) = Z(t) - \hat{Z}(t)$, y utilizando una transformación como la presentada en Subsección 4.3.3, se obtiene para $\epsilon(t)$ un equivalente al Modelo de Error 1 de la forma

$$\epsilon(t) = \phi^T(t) \tilde{u}(t),$$

con

$$\tilde{u}(t) = W(z)[\omega(t)].$$

Para las pruebas realizadas se utilizan la siguiente configuración:

4.4 Resultados experimentales

- Señal de entrada: tiene la forma $u_1(t) = u_{1,1}(t) + u_{1,2}(t)$, donde $u_{1,1} = 3 + 0,5\text{sen}(0,3t) + 0,5\text{sen}(1,7t) + \text{sen}(t)$, y $u_{1,2}(t)$ es un tren de pulsos de amplitud 1, período 3[s] y ancho de pulso 50%.
- Parámetros óptimos: $\theta_1^* = 0,5$, $\theta_2^* = 0,3$, $\theta_3^* = 0,1$, $\theta_4^* = 0,2$.
- Configuración de PSO: dado que hay 4 incógnitas, entonces $\lambda = 4$, y con ello $iter_{\max} = 80$, $s = 8$.
- Tiempo de simulación: 500[s].
- Condiciones iniciales: $[x(0)] = [0,8, 0]$, $\hat{\theta}(0) = [0, 0, 0, 0]$, $\hat{Z}(0) = 0$.

Los resultados de los experimentos se presentan a continuación, desde la Figura 4.45 hasta la Figura 4.48.

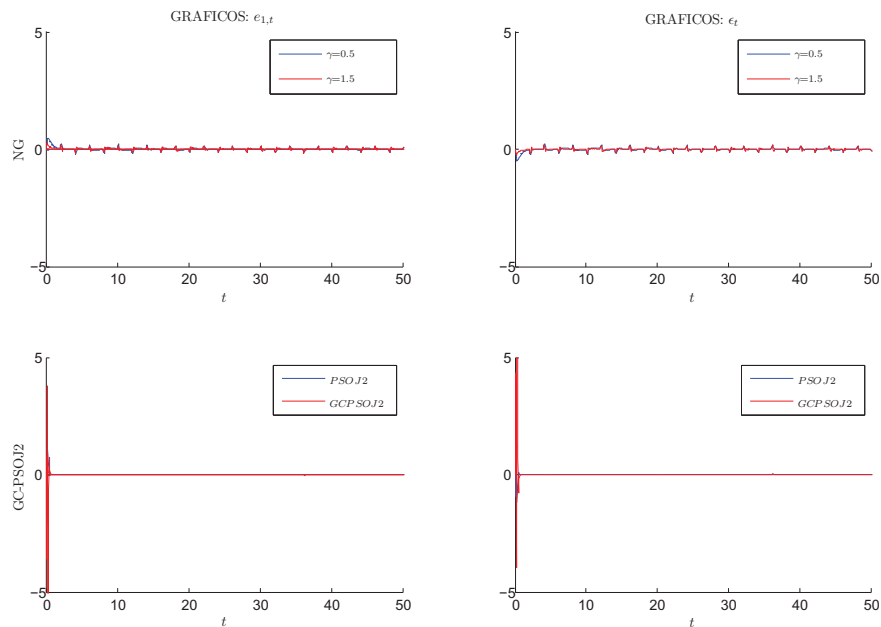


Figura 4.45.: Resultados de simulación para la Subsubsección 4.4.3.1: Evolución del $e(t)$ y $\epsilon(t)$ para NG y GC-PSOJ2.

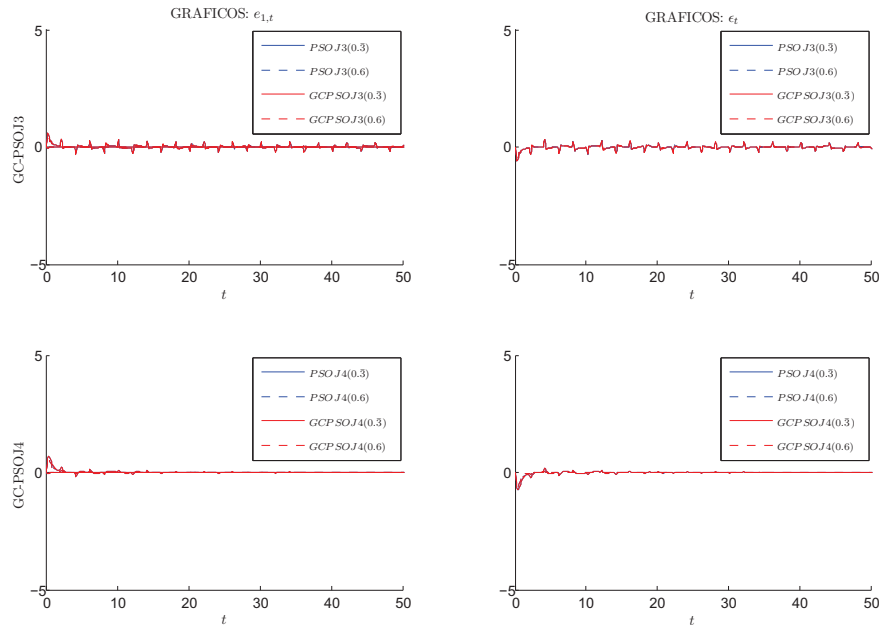


Figura 4.46.: Resultados de simulación para la Subsubsección 4.4.3.1: Evolución del error y de para GC-PSOJ3 y GC-PSOJ4.

Un análisis de la evolución de $e_1(t)$ y de $\epsilon(t)$ (Figura 4.45 y Figura 4.46) da a conocer que, a excepción de GC-PSOJ3, las metodologías basada en PSO logran minimizar de mejor forma ambas señales. En el caso de GC-PSOJ2, a pesar de que se obtiene un pico de amplitud considerable al inicio de la evolución, la convergencia a cero se logra muy prontamente (cerca de 1[s]), y se mantiene sin oscilaciones. Para GC-PSO se tiene que el transitorio es de menor amplitud para los máximos picos, aunque consumiendo un poco más de tiempo para estabilizarse en cero y mantener la convergencia (alrededor de 3[s]). Los casos NG y GC-PSOJ3 comparten similitudes respecto de oscilaciones de pequeña amplitud después de la convergencia a cero, pero queda claro que tales oscilaciones son menores para NG que para PSO bajo las condiciones evaluadas.

Respecto de la convergencia paramétrica (Figura 4.47 y Figura 4.48), se tiene que solamente para GC-PSOJ2 se da dentro de la ventana de simulación y además con buenas características como pronta velocidad de convergencia (cerca de 60[s]) y calidad del transitorio. El peor desempeño lo presenta NG, cuyas oscilaciones en el transitorio son bastante marcadas en amplitud y frecuencia, teniendo además que no se alcanza convergencia a cero. En cuanto a GP-PSOJ3, su desempeño es similar a NG, aunque con un transitorio un poco menos accidentado. Para el caso de GC-PSOJ4, si bien se tiene un transitorio más rápido y menos variable que GC-PSOJ3, se observan características de convergencia prematura a un valor que no es el vector de parámetros óptimo, pero al mismo tiempo una tendencia lenta a converger a tales parámetros después de ello.

4.4 Resultados experimentales

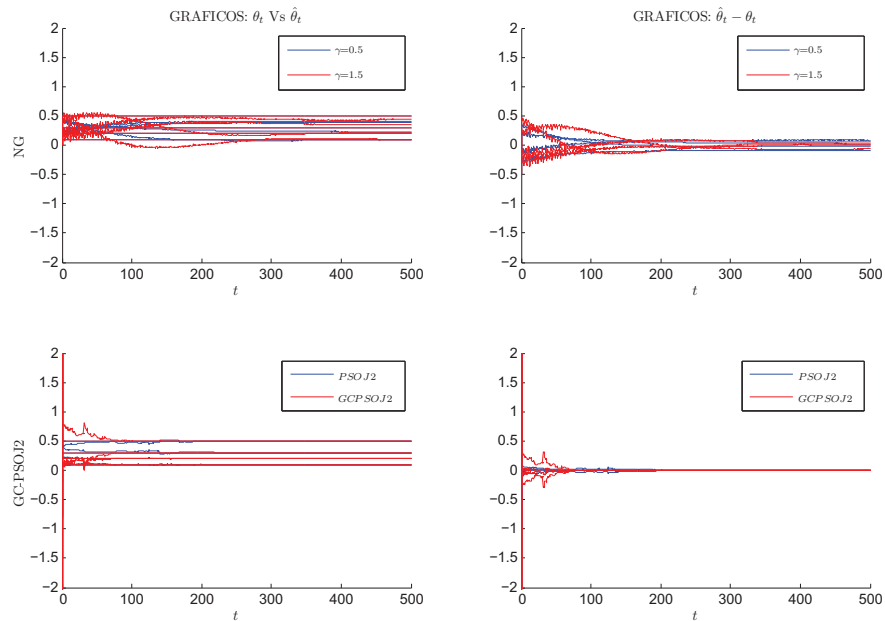


Figura 4.47.: Resultados de simulación para la Subsubsección 4.4.3.1: Estimación de los parámetros y error paramétrico para NG, NLS y GC-PSOJ2.

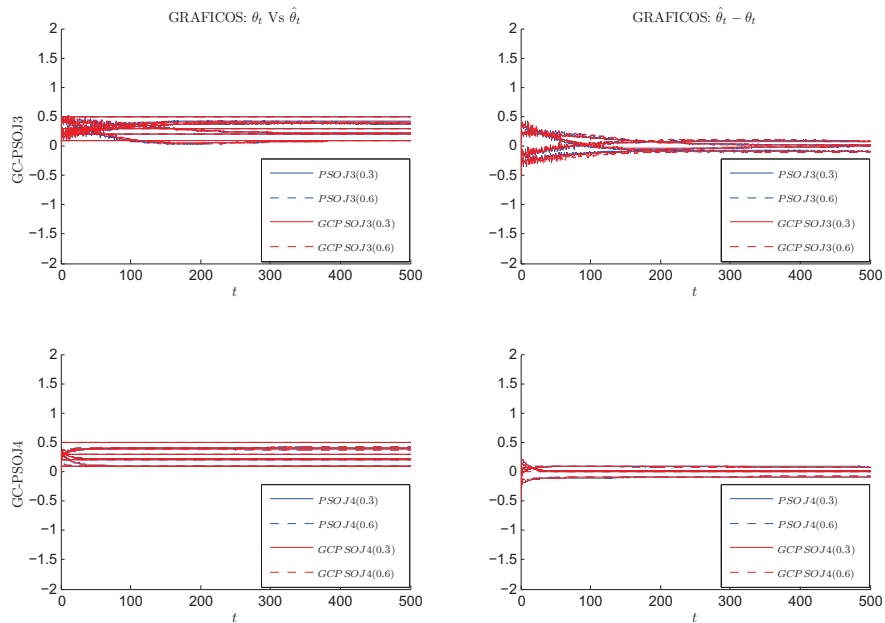


Figura 4.48.: Resultados de simulación para la Subsubsección 4.4.3.1: Estimación de los parámetros y error paramétrico para GC-PSOJ3 y GC-PSOJ4.

4.4.3.2. MRAC Directo: caso general.

Finalmente, para terminar las pruebas por simulación, se plantea un problema de control MRAC Directo con enfoque general (Ioannou and Fidan, 2006).

Sea la planta a utilizar definida por

$$G_p(z) = \frac{Y_p(z)}{U_p(z)} = 3 \frac{z + 0,5}{z^2 + z + 1},$$

y el modelo de referencia dado por

$$W_m(z) = k_m \frac{Z_m(z)}{R_m(z)} = \frac{1}{z + 0,5}.$$

A partir de lo anterior se sigue que:

- Dado $n = 2$, entonces el orden de Λ_0 es $n_0 = n - 1$, y con ello se puede escoger $\Lambda_0 = z + 0,1$.
- Con $c_0^* = \frac{1}{3}$, y a partir de $\beta_0 \in [0, c_0^*] = [0, 0,3]$ se escoge $\beta_0 = 0,2$.

Respecto de la señal de control, ésta queda definida por

$$u(t) = \theta^T(t) \omega(t),$$

con

$$\theta^T(t) = [\theta_1^T(t), \theta_2^T(t), \theta_3(t), c_0(t)],$$

$$\omega(t) = [\omega_1(t), \omega_2(t), y_p(t), r(t)],$$

y

$$\omega_1(t) = \frac{\alpha(z)}{\Lambda(z)} u_p, \quad \omega_2(t) = \frac{\alpha(z)}{\Lambda(z)} y_p,$$

$$\alpha(z) = [z, 1],$$

$$\Lambda(z) = \Lambda_0(z) Z_m(z) = z + 0,1.$$

Con todo esto, el modelo de error obtenido es de la forma (ver Subsección 4.3.3):

$$\epsilon(t) = \frac{Z(t) - \theta(t) \bar{\omega}(t)}{m_s^2(t)}$$

con

$$\omega(t) = \left[W(z) \left[\frac{\alpha^T(z)}{\Lambda(z)} \omega_1(t) \right], W(z) \left[\frac{\alpha^T(z)}{\Lambda(z)} \omega_2(t) \right], W(z) [y_p(t)], y_p(t) \right],$$

que es equivalente al modelo SPM (Ecuación 4.21)(Ioannou and Fidan, 2006, Pag. 261), es decir, un Modelo de Error 1 equivalente.

4.4 Resultados experimentales

Para las pruebas se escoge además:

- Señal de entrada: tiene la forma $r(t) = r_{1,1}(t) + r_{1,2}(t)$, donde $r_{1,1} = 3 + 0,5\text{sen}(0,3t) + 0,5\text{sen}(1,7t) + \text{sen}(t)$, y $r_{1,2}(t)$ es un tren de pulsos de amplitud 1, período 3[s] y ancho de pulso 50%.
- Configuración de PSO: dado que hay 4 incógnitas, entonces $\lambda = 4$, y con ello $iter_{\text{max}} = 80$, $s = 8$.
- Tiempo de simulación: 100[s].
- Condiciones iniciales: $\hat{\theta}(0) = [0, 0, 0, 0, 2]$, $[y_p(0), y_m(0)] = [0, 0]$.

Los resultados de los experimentos se presentan a continuación, desde la Figura 4.49 hasta la Figura 4.53.

La comparación respecto del error de control es presentada en la Figura 4.49 y Figura 4.50. A diferencia de los casos anteriores, para el caso del Modelo de Error 3 llevado a un equivalente de Modelo de Error 1 se presenta cierto desmejoramiento del desempeño general, y respecto de NG. Si bien para GC-PSOJ2 había mostrado buen desempeño en los otros modelos de error, aquí se muestra con un transitorio bastante turbulento a pesar de ser relativamente corto (2[s] para GCPSO y 5[s] para PSO).

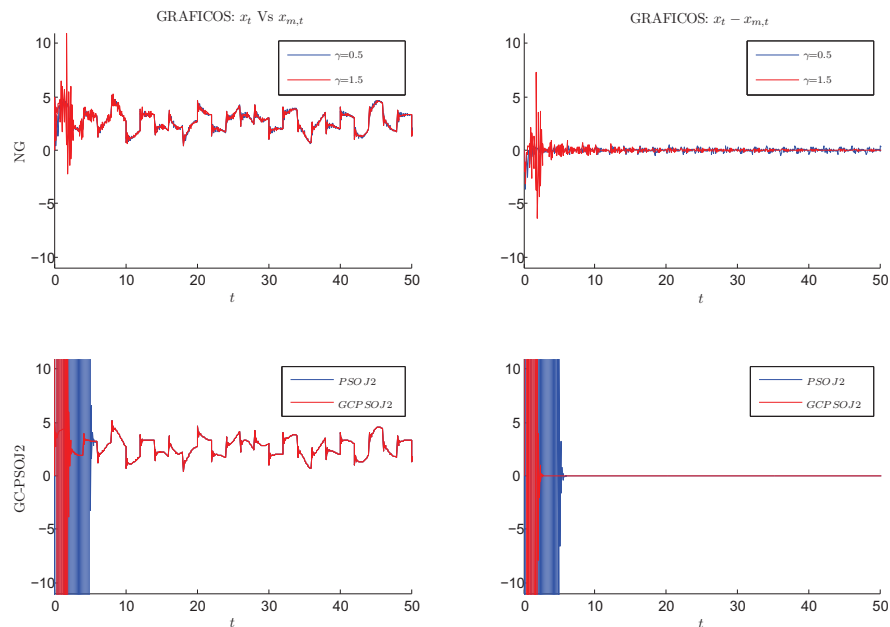


Figura 4.49.: Resultados de simulación para la Subsubsección 4.4.3.2: Comparación salida real y salida del modelo de referencia para NG y GC-PSOJ2.

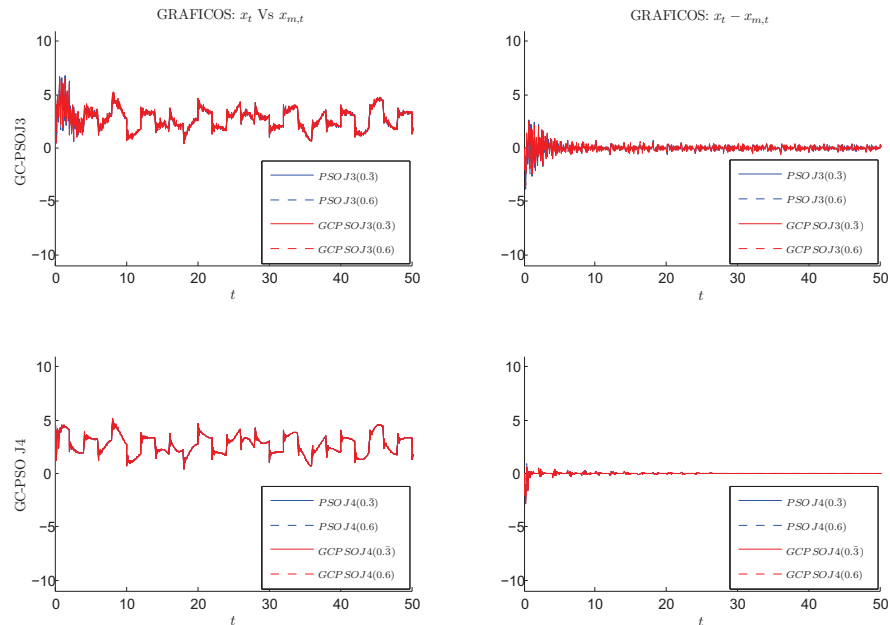


Figura 4.50.: Resultados de simulación para la Subsubsección 4.4.3.2: Comparación salida real y salida del modelo de referencia para GC-PSOJ3 y GC-PSOJ4.

El desempeño de GC-PSOJ2 puede ser explicado por varios factores. Uno de ellos es que la optimización para el error de control no está soportada en una etapa de identificación de la planta (enfoque directo), y dado que el vector de información $\omega(t)$ depende directamente de las señales $y_p(t)$ y $u(t)$, entonces este transitorio turbulento se presenta como una necesidad cubierta por el mismo sistema en busca de que $\omega(t)$ tenga mayor grado de excitación persistente. Ahora, ya que se trata de la etapa de control, entonces un mal desempeño inicial puede verse amplificado hasta que exista la información suficiente para hacer una adecuada estimación de los parámetros del controlador. Sin embargo, a pesar de este desfavorable transitorio, el error de control logra llevarse a cero y mantenerse ahí.

En cuanto a GC-PSOJ3, se tiene un desempeño similar a NG, pero éste último presenta una evolución con oscilaciones de menor amplitud. El único factor que GC-PSOJ3 mejora respecto de NG es que los máximos picos son menores, pero sólo para el caso de NG con $\gamma = 1,5$. Por su parte, GC-PSOJ4 es la metodología que presenta el mejor desempeño global: el transitorio dura apenas 2[s] y las oscilaciones que se presentan (de baja amplitud) no aparecen más allá del instante $t = 25[s]$.

Respecto de la evolución de la estimación paramétrica (Figura 4.51), se tiene que la metodología que ofrece una mayor estabilidad es GC-PSOJ4, ya que el transitorio de GC-PSOJ2 es muy turbulento en principio a pesar de su rápida convergencia. Para GC-PSOJ3 y NG se observan similitudes en sus transitorios respecto de características como convergencia prematura y posterior convergencia lenta.

4.4 Resultados experimentales

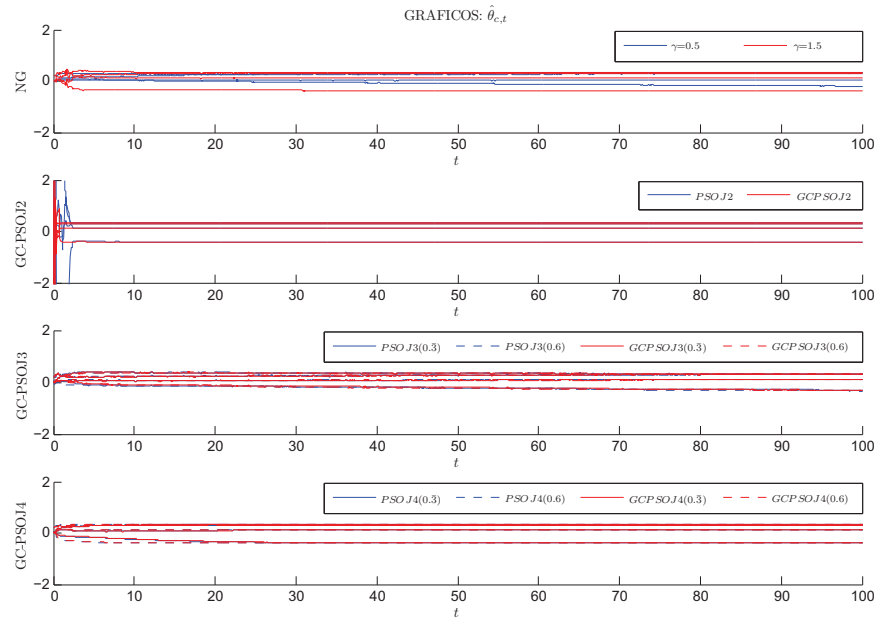


Figura 4.51.: Resultados de simulación para la Subsubsección 4.4.3.2: Estimación de los parámetros del controlador para NG, GC-PSOJ2, GC-PSOJ3 y GC-PSOJ4.

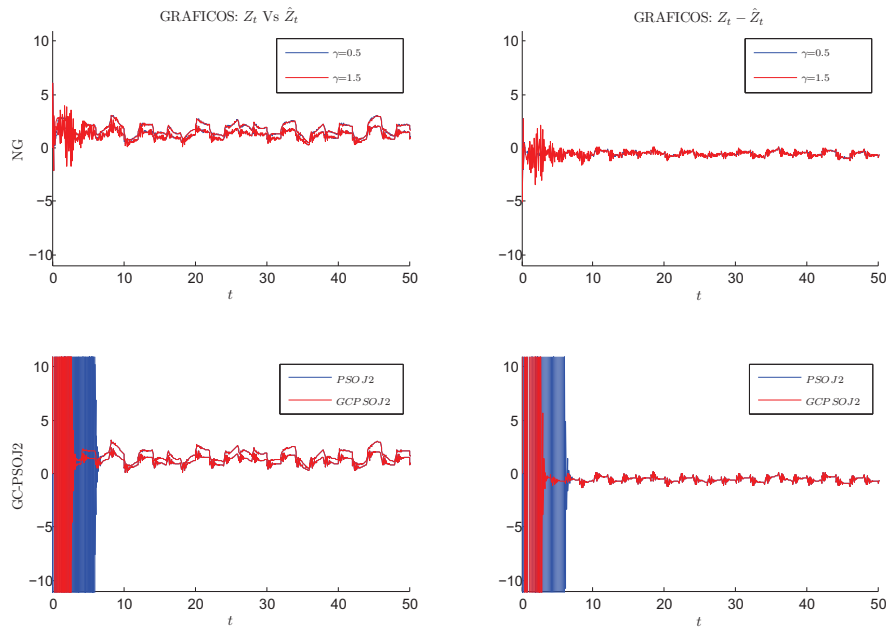


Figura 4.52.: Resultados de simulación para la Subsubsección 4.4.3.2: Comparación Z y \hat{Z} para NG y GC-PSOJ2.

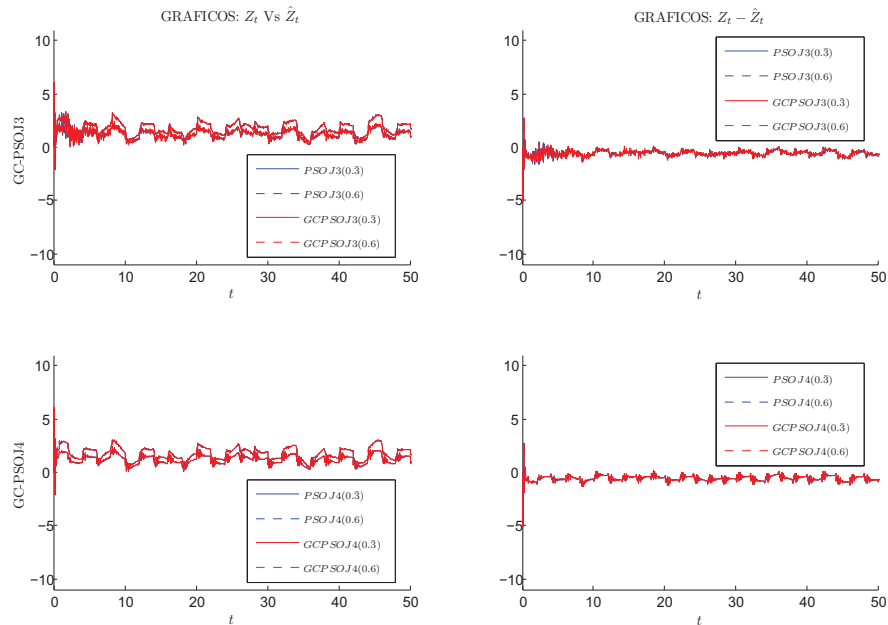


Figura 4.53.: Resultados de simulación para la Subsubsección 4.4.3.2: Comparación Z y \hat{Z} para GC-PSOJ3 y GC-PSOJ4.

A pesar de las conclusiones anteriores, es importante analizar la evolución de $Z(t) \doteq e(t)$ y $\hat{Z}(t) \doteq \hat{e}_1(t)$, y su diferencia $\epsilon(t) = Z(t) - \hat{Z}(t)$, ya que ésta es la variable directa de optimización (la minimización de $e_1(t)$ es una consecuencia de minimizar $\epsilon(t)$). La Figura 4.52 y Figura 4.53 permiten concluir que, efectivamente, GC-PSOJ4 es la metodología que logra de mejor manera minimizar $\epsilon(t)$ dentro de la ventana de simulación, mostrando además que, a pesar de que el error de control para GC-PSOJ4 llega a cero, no es necesario que esto mismo ocurra con $\epsilon(t)$, aunque sí debe estar acotado y con pequeña magnitud.

Para terminar, es interesante analizar el caso de GC-PSOJ2 y GC-PSOJ4 con un poco más de detalle. Inicialmente, hay que tener en cuenta que GC-PSOJ2 es un caso especial de GC-PSOJ4 con $\alpha = 1$. Ahora, analizando las bondades de ambas metodologías, y con el fin de minimizar sus desventajas, se deduce que una buena configuración de la metodología propuesta es usar GC-PSOJ4 lo más cercano posible a ser GC-PSOJ2, es decir, un valor de α muy cercano a 1. Lo anterior se justifica desde el punto de vista de que GC-PSOJ2 proporciona una alta velocidad de convergencia del error de salida, y GC-PSOJ4 puede añadir un suavizamiento al turbulento transitorio de GC-PSOJ2. Para evaluar esta propuesta, se decide hacer una última prueba para GC-PSOJ4 con $\alpha = 0,99$ (equivalente a $\gamma = 98,99$ en NG). Los resultados se presentan en la Figura 4.54, mostrando concluyentemente que los beneficios predichos son claramente obtenidos.

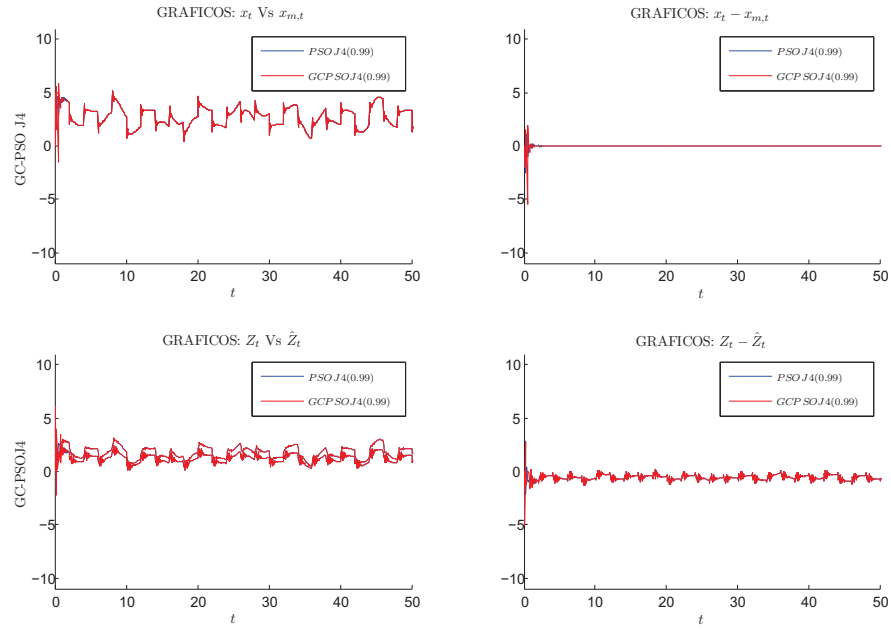


Figura 4.54.: Resultados de simulación para la Subsubsección 4.4.3.2: Señales para GC-PSOJ4(0.99).

5. Conclusiones y trabajo futuro

5.1. PSO en sistemas conmutados

5.1.1. Cálculo de una CQLF

En la presente Tesis Doctoral se desarrolló una metodología para el cálculo de una CQLF inspirada en la metodología de Liberzon and Tempo (2004), pero usando como herramienta de optimización la técnica PSO en lugar de gradiente. Una de las principales conclusiones es que se verificó con éxito que PSO es una técnica aplicable al cálculo una CQLF, con ventajas comparativas respecto de otras técnicas incluidas LMI, gradiente, algoritmo NB, y algoritmo EB. Las ventajas obtenidas están relacionadas con el tiempo de cálculo de una CQLF para cierto tipo de sistemas, y con la robustez de la CQLF calculada con PSO frente a variaciones paramétricas (medida a través de índices de desempeño cualitativos y cuantitativos).

Si bien se mostró que el tiempo de cálculo puede mejorarse en grandes proporciones para ciertos casos de estudio, también se analizó la robustez de la mejor solución que fueran capaz de entregar las metodologías bajo ciertas configuraciones, independientemente del tiempo que fuera necesario para obtenerlas. A la luz de este análisis fue posible llegar a la conclusión de que PSO muestra una tendencia a calcular CQLFs que son más robustos (respecto de las variaciones paramétricas en los sistemas analizados) que la metodología basada en gradiente, tanto cualitativa como cuantitativamente, lo que la convierte en una metodología que ofrece más confiabilidad a la hora de asegurar la estabilidad de un SLS por medio del enfoque CQLF.

En cuanto a limitaciones, se encontraron dos puntos de interés que afectan la efectividad y eficiencia de la metodología propuesta: i) la naturaleza de las matrices analizadas, y ii) la dimensionalidad del problema. Para la primera limitación se tiene que bajo un tipo de estructuras tales como triangulares y diagonales, y bajo ciertas propiedades como triangularización y diagonalización simultánea, PSO muestra ventajas notables sobre gradiente, no siendo así para el caso de matrices que conmutan, matrices definidas negativas o matrices genéricas. Sin embargo, es importante notar que justamente en los casos en que gradiente no es eficiente PSO sí lo es, por lo cual se hablaría de dos metodologías complementarias más que simplemente excluyentes. Respecto de la dimensionalidad del problema, se tiene que el aumento del orden de los sistemas es la variable que más afecta a la metodología basada en PSO, haciendo que los tiempos de cálculo se eleven debido, entre otros aspectos, a que el tamaño de la población depende directamente de esa variable. Por su parte, el aumento en el número de sistemas a analizar no mostró tener efectos tan pronunciados como en el caso de gradiente. Un ejemplo ilustrativo de esto es el caso de matrices triangulares, en donde la relación entre el número de sistemas analizados y el tiempo tiempo de cálculo es lineal con pendiente baja al usar PSO, pero aproximadamente exponencial al usar gradiente. No obstante,

independientemente del tiempo de cálculo, PSO logra soluciones factibles que mostraron ser más robustas como ya se explicó anteriormente.

A pesar de que sólo se mostraron dos alternativas de funciones de fitness a optimizar con PSO, también podrían explorarse otras funciones de fitness y/o representaciones de matrices a fin de mejorar los tiempos de convergencia. Fue encontrado además que el proceso de búsqueda desarrollado por PSO es altamente influenciado por la inicialización de las poblaciones, encontrándose que es beneficioso por ejemplo incluir una partícula con posición inicial representando la matriz identidad (que es CQLF para matrices Hurwitz definidas negativas), o un determinado número de soluciones a las ecuaciones de Lyapunov individuales. En cuanto a la metodología basada en gradiente, también podrían ser adicionadas algunas mejoras, como por ejemplo hacer que los parámetros α y r sean variantes en el tiempo para potenciar la convergencia. Sin embargo, estas mejoras quedan fuera del alcance de esta tesis.

Para finalizar, se concluye que la metodología propuesta es una nueva alternativa de solución al problema CQLF con ventajas y fortalezas ampliamente marcadas respecto de las metodologías más comúnmente usadas en la actualidad.

5.1.2. No existencia de una CQLF

En la presente Tesis Doctoral también se desarrolló una metodología para la determinación de no-existencia de una CQLF basada en PSO. La metodología diseñada fue puesta a prueba para casos de diversa dimensionalidad (orden y número de sistemas), mostrando en general un buen desempeño que parece disminuir en la medida que aumenta la dimensionalidad del problema, en especial en lo relacionado al incremento en el orden de las matrices a analizar, lo que afecta la velocidad de convergencia y eventualmente la convergencia misma.

Cuando la salida del proceso de optimización es un número no positivo, el método es capaz de asegurar con certeza 1 que el conjunto de matrices bajo análisis no comparte una CQLF. Sin embargo, ya que se trata de la verificación experimental de una condición suficiente para la no-existencia de una CQLF, cuando se tiene como salida del proceso de optimización un número positivo la metodología propuesta no es capaz de garantizar concluyentemente si existe una CQLF para el conjunto de matrices bajo análisis. No obstante, la metodología propuesta constituye un importante progreso en el proceso de dar una completa solución numérica para el problema CQLF, especialmente considerando que la complejidad del problema se incrementa para sistemas de orden superior, en los cuales la solución propuesta mostró ser capaz de ofrecer información concluyente donde otros enfoques no tuvieron éxito. PSOiw exhibió buenos resultados en el ámbito de este estudio particular, tales como la habilidad de encontrar soluciones factibles con rápida convergencia (en dependencia de la función de fitness utilizada). Por su parte, puede resultar benéfico desde el punto de vista técnico emplear una versión de PSO más reciente (por ejemplo Del Valle et al. (2008); Kameyama (2009), entre muchas otras), y desde el punto de vista analítico podría beneficiar el uso de otro esquema de variación del peso de inercia, o emplear otro tipo de inicialización del enjambre a fin de potenciar la convergencia. Ya que las ventajas de los tipos de inicializaciones usadas no fue tan claro, (mostrando ser altamente dependientes de los sistemas analizados y/o la función de fitness empleada), sería interesante explorar los potenciales beneficios de usar los tipos de inicialización presentados en Clerc (2008).

Dada la naturaleza de la metodología propuesta, las comparaciones con técnicas tradicionales como LMI o gradiente fueron desarrolladas en el contexto de metodología complementaria. Por otro lado, GA y DE podrían ser directamente utilizados como herramienta de optimización en vez de PSO, pero esto va más allá del alcance de esta Tesis. Finalmente, se debe notar que otras condiciones necesarias y/o suficientes para la existencia/no-existencia de una CQLF podrían ser analizadas a fin de extender los resultados obtenidos en este estudio, al emplear el método propuesto pero considerando las particularidades de las nuevas condiciones.

5.1.3. Trabajo futuro

Respecto del trabajo futuro, directamente previsible y realizable en el corto plazo, se encuentran los siguientes temas:

- Diseño de una metodología de cálculo de CQLFs para sistemas de tiempo discreto.
- Diseño de una metodología para la determinación de la no-existencia de una CQLF para sistemas de tiempo discreto.

Respecto del trabajo futuro en el largo plazo, a ser realizados con más detenimiento, se encuentran:

- Evaluación de otras condiciones de existencia/no-existencia de una CQLF, partiendo de la base del presente estudio.
- Estudio comparativo respecto de otras técnicas de optimización global como DE y GA, tanto práctico como analítico.

5.2. PSO en sistemas adaptables

Otro de los productos de la presente Tesis Doctoral es una metodología de ajuste de parámetros para sistemas adaptables discretos representados por los modelos de error, basada en la técnica de optimización heurística PSO. El desempeño de la metodología propuesta se comparó con el de las dos técnicas tradicionales de ajuste más ampliamente usadas: NG y NLS.

Los resultados de simulación muestran la efectiva aplicabilidad de PSO al problema de la estimación paramétrica en línea desde el enfoque de los modelos de error, y permiten identificar situaciones ventajosas en las que PSO tiene mejor desempeño que NG y NLS a partir del análisis de aspectos tanto cualitativos como cuantitativos. Los problemas analizados en los cuales PSO mostró una evidente ventaja comparativa respecto de NG y NLS incluyen:

- Identificación paramétrica,
- Control Adaptable Indirecto,
- Control Adaptable Combinado,
- Control Adaptable Directo.

No obstante, el campo de aplicación de la metodología desarrollada va mucho más allá de estos problemas específicos, puesto que la metodología puede ser aplicada a cualquier sistema discreto que pueda ser llevado a una representación por modelos discretos de error tipo 1, 2 y 3.

En cuanto a los aspectos técnicos de la metodología propuesta, se hicieron pruebas que intentaron dilucidar las posibles ventajas respecto de la utilización de las técnicas tradicionalmente usadas. No obstante, a pesar de que las pruebas de validación mostraron marcadas ventajas en el desempeño de la metodología propuesta, aún hacen falta estudios más detallados sobre la elección óptima de los parámetros de diseño, teniendo entre los más relevantes el factor de ponderación α y la ventana de minimización del historial del error M_2 , seguidos de el tamaño de población s y el número máximo de iteraciones $iter_{\max}$.

Las conclusiones específicas pueden verse resumidas de la siguiente manera:

- GC-PSOJ1: Presenta las mejores ventajas respecto de la minimización del error de salida. No obstante, no garantiza convergencia paramétrica, y eventualmente ni siquiera convergencia de $\phi(t)$ a un valor constante diferente de cero. Por tanto, apenas es recomendable su uso para casos donde se requiera minimización del error de salida pero no se requiera estimación paramétrica.
- GC-PSOJ2: Presenta muy buen desempeño en muchos de los problemas analizados, tanto para el caso de identificación como para el caso de control. Sin embargo, su utilización siempre debe tener la precaución de escoger un adecuado valor de M_2 para evitar comportamientos no deseados en una eventual ausencia de EP y/o EIP.
- GC-PSOJ3: Bajo las consideraciones tenidas en cuenta, presenta ciertas virtudes en los casos de identificación y control. Sin embargo, se observaron características de convergencia lenta del error paramétrico, que bien pueden ser producto de una inadecuada elección de α . Posiblemente para otros valores de α se obtengan mejores resultados.
- GC-PSOJ4: Es la metodología que mejor desempeño global presentó para los casos de identificación y control, incluso en problemas de gran complejidad como lo son MRAC Directo o MRAC Combinado. Las características obtenidas para la evolución del error de identificación y error de control usando esta metodología muestran que se mejora en gran magnitud el desempeño obtenido con técnicas tradicionales por excelencia tales como NG o NLS.

5.2.1. Trabajo futuro

Los resultados obtenidos en la presente Tesis Doctoral pueden verse continuados a corto plazo respecto de los siguientes aspectos:

- Estudio sobre la configuración óptima de los parámetros de diseño α , M_2 , s e $iter_{\max}$.
- Pruebas prácticas en procesos reales, corriendo PSO en tiempo real (en PC y/o embebido en hardware).
- Redacción de propuesta de publicación en revista ISI relacionada con la metodología de diseño de leyes de ajuste (que puede incluir los dos puntos anteriores).

En cuanto al trabajo futuro a realizarse en largo plazo, se puede enumerar:

- Diseño de una metodología para el ajuste paramétrico en sistemas adaptables representados por modelos de error en tiempo continuo.
- Estudio comparativo respecto de otras técnicas de optimización global como DE y GA, tanto práctico como analítico.

A. Toolbox de PSO

A.1. Introducción

El Toolbox PSO (Singh, 2003) es una colección de archivos Matlab (.m) que se pueden utilizar para implementar el algoritmo PSO para optimizar un sistema dado.

PSO fue introducido por Russel Eberhart (ingeniero eléctrico) y James Kennedy (psicólogo social) en 1995 (ambos asociados al IUPUI en ese momento) (Eberhart and Kennedy, 1995a). PSO pertenece a las categorías de inteligencia de enjambre y algoritmos evolutivos para la optimización. Se inspira en el comportamiento social de las aves, que fue estudiado por Craig Reynolds (biólogo) en los 80's y principios de los 90's. De ello se deriva una fórmula para la representación del comportamiento de las aves. Esta fue utilizada más adelante en simulaciones por ordenador sobre “aves virtuales”, conocidas como *Boids*. Eberhart y Kennedy reconocieron la idoneidad de esta técnica para la optimización, y entonces sobrevino el desarrollo del optimizador por enjambre de partículas *PSO*.

La representación del problema de optimización para PSO es similar a los métodos de codificación utilizados en GA. En lugar de los genes, las variables se denominan dimensiones, que crean un hiperespacio multidimensional. Las “partículas” vuelan en este hiperespacio, y así tratan de encontrar los mínimos/máximos globales, con su movimiento gobernado por ecuaciones matemáticas simples. Éstas ecuaciones son las que implementa el Toolbox PSO, más específicamente las que describen las versiones PSOiw (Shi and Eberhart, 1998a) y PSOcf (Clerc, 1999).

A.2. Instalación (Versión Matlab)

Para instalar la versión de Matlab del Toolbox PSO, primero es necesario descargar los archivos zip desde <http://sourceforge.net/projects/psotoolbox/files/> y descomprimirlos en una carpeta el sistema. A continuación, se inicia Matlab y se agrega la carpeta a la ruta de Matlab.

A.3. Código fuente

```

%PSO >> function for the PSO ALGORITHM
%
% USAGES:  1.) [fxmin, xmin, Swarm, history] = PSO(psoOptions);
%          2.) [fxmin, xmin, Swarm, history] = PSO;
%          3.) fxmin = PSO(psoOptions);
%          3.) PSO
%          etc.
%
% Arguments:  PSOOptions--> A Matlab structure containing all PSO related options. (see also: get_psoOptions)
% Return values: [fxmin, xmin, Swarm, history]
%               |_____|_____|_____|_____|
%               |_____|_____|_____|_____|_The history of the algorithm. Depicts how the function
%               |_____|_____|_____|_____|_value of GBest changes over the run.
%               |_____|_____|_____|_____|_The final Swarm. (A matrix containing co-ordinates of all particles)
%               |_____|_____|_____|_____|_The co-ordinates of Best (ever) particle found during the PSO's run.
%               |_____|_____|_____|_____|_The objective value of Best (λxmin) particle.
%
% History      : Author      : JAG (Jagatpreet Singh)
%               Created on   : 05022003 (Friday, 2nd May, 2003)
%               Comments    : The basic PSO algorithm.
%               Modified on  : 0710003 (Thursday, 10th July, 2003)
%               Comments    : It uses PSOOptions structure now. More organized.
%
%               see also: get_psoOptions
function [fxmin, xmin, Swarm, history] = PSO(psoOptions)

%Globals
global PSOFlags;
global PSOVars;
global PSOParameters;
global notifications;
global PSOOptions;

upbnd = 600; % Upper bound for init. of the swarm
lwbnd = 300; % Lower bound for init. of the swarm
GM = 0; % Global minimum (used in the stopping criterion)
ErrGoal = 1e-10; % Desired accuracy
%

%Initializations
if nargin == 0
    PSOOptions = get_psoOptions;
end

%For Displaying
if PSOOptions.Flags.ShowViz
    %Use the specified axes if using GUI or create a new global if called from command window
    global vizAxes;
    vizAxes = plot(0,0, 'o');
    axis([-1000 1000 -1000 1000 -1000 1000]); %Initially set to a cube of this size
    axis square;
    grid off;
    set(vizAxes, 'EraseMode', 'xor', 'MarkerSize', 8); %Set it to show particles.
    pause(1);
end
%End Display initialization

% Initializing variables
success = 0; % Success Flag
iter = 0; % Iterations' Counter
fevals = 0; % Function evaluations' counter

% Using params---
% Determine the value of weight change
w_start = PSOOptions.SParams.w_start; %Initial inertia weight's value
w_end = PSOOptions.SParams.w_end; %Final inertia weight

%Weight change step. Defines total number of iterations for which weight is changed.
w_varyfor = floor(PSOOptions.SParams.w_varyfor*PSOOptions.Vars.Iterations);

```

A.3 Código fuente

```
w_now = w_start;
inertdec = (w_start-w_end)/w_varyfor; %Inertia weight's change per iteration

% Initialize Swarm and velocity
SwarmSize = psoOptions.Vars.SwarmSize;
Swarm = (ones(SwarmSize,1)*(psoOptions.Obj.ub-psoOptions.Obj.lb)).*rand(SwarmSize,psoOptions.Vars.Dim)...
+ ones(SwarmSize,1) * psoOptions.Obj.lb;
VStep = rand(SwarmSize, psoOptions.Vars.Dim);

f2eval = psoOptions.Obj.f2eval; %The objective function to optimize.

%Find initial function values.
fSwarm = feval(f2eval, Swarm);
fevals = fevals + SwarmSize;

% Initializing the Best positions matrix and
% the corresponding function values
PBest = Swarm;
fPBest = fSwarm;

% Finding best particle in initial population
[fGBest, g] = min(fSwarm);
lastbpf = fGBest;
Best = Swarm(g,:); %Used to keep track of the Best particle ever
fBest = fGBest;
history = [0, fGBest];

if psoOptions.Flags.Neighbor
% Define social neighborhoods for all the particles
for i = 1:SwarmSize
    lo = mod(i-psoOptions.SParams.Nhood,i+psoOptions.SParams.Nhood, SwarmSize);
    nhood(i,:) = [lo];
end
nhood(find(nhood==0)) = SwarmSize; %Replace zeros with the index of last particle.
end

if psoOptions.Disp.Interval & (rem(iter, psoOptions.Disp.Interval) == 0)
disp(sprintf('Iterations\t\tfGBest\t\t\tfevals'));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                               THE PSO LOOP                               %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
while( (success == 0) & (iter <= psoOptions.Vars.Iterations) )
    iter = iter+1;

    % Update the value of the inertia weight w
    if (iter<=w_varyfor) & (iter > 1)
        w_now = w_now - inertdec; %Change inertia weight
    end

    %%%%%%%%%%% added on Apr.7 %%%%%%%%%%%
    flag = -1;
    for k = 1 : SwarmSize
        if((fSwarm(k) == fPBest(k)) & (fSwarm(k) == fGBest))
            flag = k;
            break;
        end
    end
    if(flag~-1) [fSwarm,Swarm] = generate(fSwarm,Swarm,flag); end
    %%%%%%%%%%%

    % The PLAIN PSO %

    % Set GBest
    %A = GBest. repmat(X, m, n) repeats the matrix X in m rows by n columns.
    A = repmat(Swarm(g,:), SwarmSize, 1);

    B = A; %B will be nBest (best neighbor) matrix

    % use neighborhood model
    % circular neighborhood is used
    if psoOptions.Flags.Neighbor
        for i = 1:SwarmSize
            [fNBest(i), nb(i)] = min(fSwarm( find(nhood(i)) ));
            B(i, :) = Swarm(nb(i), :);
        end
    end
end
```

```

% Generate Random Numbers
R1 = rand(SwarmSize, psoOptions.Vars.Dim);
R2 = rand(SwarmSize, psoOptions.Vars.Dim);

% Calculate Velocity
if ~psoOptions.Flags.Neighbor %Normal
    VStep = w_now*VStep+psoOptions.SParams.c1*R1.*(PBest-Swarm)+psoOptions.SParams.c2*R2.*(A-Swarm);
else %with neighborhood
    R3 = rand(SwarmSize, psoOptions.Vars.Dim); %random nos for neighborhood
    VStep = w_now*VStep+psoOptions.SParams.c1*R1.*(PBest-Swarm)+psoOptions.SParams.c2*R2.*(A-Swarm)...
        + psoOptions.SParams.c3*R3.*(B-Swarm);
end

% Apply Vmax Operator for v > Vmax
changeRows = VStep > psoOptions.SParams.Vmax;
VStep(find(changeRows)) = psoOptions.SParams.Vmax;
% Apply Vmax Operator for v < -Vmax
changeRows = VStep < -psoOptions.SParams.Vmax;
VStep(find(changeRows)) = -psoOptions.SParams.Vmax;

% ::UPDATE POSITIONS OF PARTICLES::
Swarm = Swarm + psoOptions.SParams.Chi * VStep; % Evaluate new Swarm

%%% added on Apr.8 %%%
for k = 1 : psoOptions.Vars.Dim
    changeRows = Swarm(:,k) > psoOptions.Obj.ub(k);
    Swarm(find(changeRows),k) = psoOptions.Obj.ub(k);
    changeRows = Swarm(:,k) < psoOptions.Obj.lb(k);
    Swarm(find(changeRows),k) = psoOptions.Obj.lb(k);
end
%%

fSwarm = feval(f2eval, Swarm);
fevals = fevals + SwarmSize;

% Updating the best position for each particle
changeRows = fSwarm < fPBest;
fPBest(find(changeRows)) = fSwarm(find(changeRows));
PBest(find(changeRows), :) = Swarm(find(changeRows), :);

lastbpart = PBest(g, :);
% Updating index g
[fGBest, g] = min(fPBest);

%update Best. Only if fitness has improved.
if fGBest < lastbpf
    [fBest, b] = min(fPBest);
    Best = PBest(b,:);
end

%%OUTPUT%%
if psoOptions.Save.Interval & (rem(iter, psoOptions.Save.Interval) == 0)
    history((size(history,1)+1), :) = [iter, fBest];
end

if psoOptions.Disp.Interval & (rem(iter, psoOptions.Disp.Interval) == 0)
    disp(sprintf('%4d\t\t%.5g\t\t\t%.5d', iter, fGBest, fevals));
end

if psoOptions.Flags.Showviz
    [fworst, worst] = max(fGBest);
    % DrawSwarm(Swarm, SwarmSize, iter, psoOptions.Vars.Dim, Swarm(g,:), vizAxes);
    % ViewClusters(iter, psoOptions.Vars.Dim, Swarm(g,:), vizAxes);
end

%%TERMINATION%%
if abs(fGBest-psoOptions.Obj.GM) <= psoOptions.Vars.ErrGoal %GBest
    success = 1;
elseif abs(fBest-psoOptions.Obj.GM)<=psoOptions.Vars.ErrGoal %Best
    success = 1
else
    lastbpf = fGBest; %To be used to find Best
end

end
%%%
%%
%% END OF PSO LOOP
%%
%%

```

```
[fxmin, b] = min(fPBest);
xmin = PBest(b, :);

history = history(:,1);
%Comment below line to Return Swarm. Uncomment to return previous best positions.
% Swarm = PBest; %Return PBest

function [fSwarm,Swarm] = generate(fSwarm,Swarm,num)
global psoOptions;
f2eval = psoOptions.Obj.f2eval;
x = rand(1,psoOptions.Vars.Dim).*(psoOptions.Obj.ub - psoOptions.Obj.lb) + psoOptions.Obj.lb;
y = feval(f2eval, x);
Swarm(num,:) = x;
fSwarm(num) = y;
return ...
```

Bibliografía

- Alizadeh, T., Salahshoor, K., Jafari, M., Alizadeh, A., Gholami, M., September 25-28 2007. On-line identification of hybrid systems using an adaptive growing and pruning rbf neural network. In: IEEE Conference on Emerging Technologies and Factory Automation 2007. pp. 257–264.
- Angelov, P., Xydeas, C., Filev, D., July 25-29 2004. On-line identification of mimo evolving takagi-sugeno fuzzy models. In: IEEE International Conference on Fuzzy Systems 2004. Vol. 1. pp. 55–60.
- Banks, C., 2002. Searching for lyapunov functions using genetic programming. Technical report, Virginia Polytechnic Institute and State University, Blacksburg, VA 24060: Virginia.
- Benzaouia, A., Akhrif, O., Saydy, L., 2010. Stabilisation and control synthesis of switching systems subject to actuator saturation. *International Journal of Systems Science* 41 (4), 397–409.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207720903045791>
- Benzaouia, A., Hmamed, A., Tadeo, F., Hajjaji, A. E., 2011. Stabilisation of discrete 2d time switching systems by state feedback control. *International Journal of Systems Science* 42 (3), 479–487.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207720903576522>
- Bhattacharya, S., Konar, A., Nagar, A., September 2008. A lyapunov-based extension to pso dynamics for continuous function optimization. In: Second UKSIM European Symposium on Computer Modeling and Simulation(EMS '08). pp. 28–33.
- Boeringer, D., Werner, D., March 2004. Particle swarm optimization versus genetic algorithms for phased array synthesis. *IEEE Transactions on Antennas and Propagation* 52 (3), 771–779.
- Boyd, S., El-Ghaoui, L., Feron, E., Balakrishnan, V., 1994. *Linear Matrix Inequality in System and Control Theory*. SIAM, Philadelphia, PA.
- Bratton, D., Kennedy, J., April 1-5 2007. Defining a standard for particle swarm optimization. In: *Swarm Intelligence Symposium, 2007. SIS 2007*. IEEE. Honolulu, HI, pp. 120–127.
- Brits, R., Engelbrecht, A., Van Den Bergh, F., April 2003. Scalability of niche pso. In: *Proceedings of the 2003 IEEE Swarm Intelligence Symposium (SIS 03)*. pp. 228–234.
- Brits, R., Engelbrecht, A., Van Den Bergh, F., 2007. Locating multiple optima using particle swarm optimization. *Applied Mathematics and Computation* 189 (2), 1859–1883.
URL <http://www.scopus.com/inward/record.url?eid=2-s2.0-34249068297&partnerID=40&md5=ad52201497d2de07aff681343f9b51f8>
- Burnham, K. P., Anderson, D. R., 2004. Multimodel inference: Understanding aic and bic in model selection. *Sociological Methods & Research* 33 (2), 261–304.
URL <http://smr.sagepub.com/content/33/2/261.abstract>

- Carlisle, A., Dozier, G., 2001. An off-the-shelf pso. In: in Proceedings of the Workshop on Particle Swarm Optimization, 2001. Vol. 1. Indianapolis, USA, pp. 1–6.
- Chen, J., Pan, F., Cai, T., Tu, X., 2003. Stability analysis of particle swarm optimization without lipschitz constraint. *Journal of Control Theory and Applications* 1, 86–90, 10.1007/s11768-003-0014-2.
URL <http://dx.doi.org/10.1007/s11768-003-0014-2>
- Chen, Y.-R., Dye, C.-Y., 2012. Application of particle swarm optimisation for solving deteriorating inventory model with fluctuating demand and controllable deterioration rate.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207721.2011.652224>
- Chen, Z., Gao, Y., 2007. The computation of a common quadratic lyapunov function for linear control system. *Journal of Information and Computing Science* 2, 299–304.
- Cheng, D., Guo, L., Huang, J., May 2003. On quadratic lyapunov functions. *IEEE Transactions on Automatic Control* 48 (5), 885–890.
- Cheng, D., Hu, X., Martin, C., 2006. On the smallest enclosing balls. *Communications in Information and Systems* 6, 137–160.
- Cheng, D., Martin, C., Xiang, J., 2000. An algorithm for common quadratic lyapunov function. In: Proceedings of the 3rd World Congress on Intelligent Control and Automation, 2000. Vol. 4. pp. 2965–2969.
- Cheng, D., Zhang, L., 2006. Adaptive control of linear markov jump systems. *International Journal of Systems Science* 37 (7), 477–483.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207720600752608>
- Chuan, L., Quanyuan, F., August 2007. The standard particle swarm optimization algorithm convergence analysis and parameter selection. In: Third International Conference on Natural Computation (ICNC 2007). Vol. 3. pp. 823–826.
- Clerc, M., 1999. The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99. Washington DC, pp. 1951–1957.
- Clerc, M., 2008. Initialisations for particle swarm optimisation. Tech. rep., Online technical report at <http://clerc.maurice.free.fr/pso/>.
- Clerc, M., Kennedy, J., 2002. The particle swarm-explosion, stability, and convergence in a multi-dimensional complex space. *IEEE Transactions on Evolutionary Computation* 6 (1), 58–73.
- Coelho, L. d. S., Guerra, F., 2007. Applying particle swarm optimization to adaptive controller. In: Saad, A., Dahal, K., Sarfraz, M., Roy, R. (Eds.), *Soft Computing in Industrial Applications*. Vol. 39 of *Advances in Soft Computing*. Springer Berlin / Heidelberg, pp. 82–91.
URL http://dx.doi.org/10.1007/978-3-540-70706-6_8
- Cui, Z., Zeng, J., 2004. A guaranteed global convergence particle swarm optimizer. In: Tsumoto, S., Slowinski, R., Komorowski, J., Grzymala-Busse, J. (Eds.), *Rough Sets and Current Trends in Computing*. Vol. 3066 of *Lecture Notes in Computer Science*. Springer Berlin - Heidelberg, pp. 762–767.
URL http://dx.doi.org/10.1007/978-3-540-25929-9_96

- Davis, J., Eisenbarth, G., March 2011. The positivstellensatz and nonexistence of common quadratic lyapunov functions. In: IEEE 43rd Southeastern Symposium on System Theory (SSST). pp. 55–58.
- de Oca, M., Sttzle, T., Birattari, M., Dorigo, M., October 2009. Frankenstein's pso: A composite particle swarm optimization algorithm. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION 13 (5), 1120–1132.
- Del Valle, Y., Venayagamoorthy, G., Mohagheghi, S., Hernandez, J., Harley, R., 2008. Particle swarm optimisation: Basic concepts, variants and applications in power systems. IEEE Transactions on Evolutionary Computation 12, 171–195.
- Dong, R., August 14-16 2009. Differential evolution versus particle swarm optimization for pid controller design. In: Fifth International Conference on Natural Computation 2009 (ICNC '09). Vol. 3. Tianjian, China.
- dos Santos-Coelho, L., Coelho, A. A. R., 2009. Model-free adaptive control optimization using a chaotic particle swarm approach. Chaos, Solitons & Fractals 41 (4), 2001–2009.
URL <http://www.sciencedirect.com/science/article/pii/S0960077908003627>
- Duarte, M., Narendra, K., oct 1989. Combined direct and indirect approach to adaptive control. IEEE Transactions on Automatic Control 34 (10), 1071–1075.
- Duarte, M. A., Ponce, R. F., 1997. Discrete-time combined model reference adaptive control. Int. J. Adapt. Control Signal Process., 11, 501–517.
- Duarte-Mermoud, M., Narendra, K., 1996. Error models with parameter constraints. International Journal of Control 64 (6), 1089–1111.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207179608921676>
- Duarte-Mermoud, M., Ordóñez-Hurtado, R., Zagalak, P., 2012. A method for determining the non-existence of a common quadratic lyapunov function for switched linear systems based on particle swarm optimisation, international Journal of Systems Science.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207721.2012.687787>
- Eberhart, R., Kennedy, J., 1995a. A new optimiser using particle swarm theory. In: In Proceedings of the Sixth International Symposium on Micromachine and Human Science (MHS). Nagoya, Japan, pp. 39–43.
- Eberhart, R., Kennedy, J., 1995b. Particle swarm optimization. In: In Proceedings of IEEE International Conference on Neural Networks (ICNN). Vol. 4. Piscataway, NJ, pp. 1942–1948.
- Eberhart, R., Shi, Y., 2000. Comparing inertia weights and constriction factors in particle swarm optimisation. In: In Proceedings of the 2000 Congress on Evolutionary Computation-CEC00. Vol. 1. La Jolla, CA, USA, pp. 84–88.
- Eberhart, R., Shi, Y., May 2001a. Particle swarm optimization: Developments, applications and resources. In: in Proc. IEEE Congr. Evol. Comput. 2001. Vol. 1. pp. 81–86.
- Eberhart, R., Shi, Y., 2001b. Tracking and optimizing dynamic systems with particle swarms. In: Proc. IEEE Congr. Evol. Comput. 2001. pp. 94–100.
- Emara, H., Abdel Fattah, H., June 30 - July 2 2004. Continuous swarm optimization technique with stability analysis. In: Proceedings of the 2004 American Control Conference. Vol. 3. pp. 2811–2817.

- Fan, W., Cui, Z., Ceng, J., August 2009. Inertia weight selection strategy based on lyapunov stability analysis. In: Ninth International Conference on Hybrid Intelligent Systems(HIS '09). Vol. 3. pp. 504–509.
- Fernández-Martínez, J., García-Gonzalo, E., January 2008. The generalized pso: A new door to pso evolution. *J. Artif. Evol. App.* 2008, 5:1–5:15.
URL <http://dx.doi.org/10.1155/2008/861275>
- Fernández-Martínez, J., García-Gonzalo, E., Fernández-Alvarez, J. P., 2008. Theoretical analysis of particle swarm trajectories through a mechanical analogy. *International Journal of Computational Intelligence Research* 4 (2), 93–104.
- Goodwin, G. C., Ramadge, P. J., Caines, P. E., June 1980. Discrete-time multivariable adaptive control. *IEEE Transactions on Automatic Control* AC-25 (3), 449–456.
- Grosman, B., Lewin, D., July 4-8 2005. Automatic generation of lyapunov functions using genetic programming. In: 16th IFAC World Congress. Prague|.
- Guilan, L., Hai, Z., Chunhe, S., November 2008. Convergence analysis of a dynamic discrete pso algorithm. In: First International Conference on Intelligent Networks and Intelligent Systems (ICINIS '08). pp. 89–92.
- Gurvits, L., Shorten, R., Mason, O., June 2007. On the stability of switched positive linear systems. *Automatic Control, IEEE Transactions on* 52 (6), 1099 –1103.
- Habib, S., Al-kazemi, B., September 2-5 2005. Comparative study between the internal behavior of ga and pso through problem-specific distance functions. In: The 2005 IEEE Congress on Evolutionary Computation. Vol. 3. pp. 2190–2195.
- Horn, R., Johnson, C., 1985. *Matrix Analysis*. Cambridge University Press, Cambridge (Cambridgeshire), NY.
- Hsu, C., Shyr, W., Kuo, K., 2010. Optimizing multiple interference cancellations of linear phase array based on particle swarm optimization. *Journal of Information Hiding and Multimedia Signal Processing* 1 (4), 292–300.
- Hu, J.-Z., Xu, J., qiao Wang, J., Xu, T., September 20-22 2009. Research on particle swarm optimization with dynamic inertia weight. In: International Conference on Management and Service Science, 2009. MASS '09. Wuhan, China, pp. 1–4.
- Ibeas, A., de la Sen, M., 2009. Exponential stability of simultaneously triangularizable switched systems with explicit calculation of a common lyapunov function. *Applied Mathematics Letters* 22 (10), 1549–1555.
URL <http://www.sciencedirect.com/science/article/pii/S0893965909001633>
- Ioannou, P., Fidan, B., 2006. *Adaptive Control Tutorial*. Society for Industrial and Applied Mathematics (SIAM) Series. Advanced in Design and Control, Philadelphia, USA.
- Janson, S., Middendorf, M., December 2005. A hierarchical particle swarm optimizer and its adaptive variant. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 35 (6), 1272–1282.
- Jiang, M., Luo, Y., Yang, S., April 2007a. Stagnation analysis in particle swarm optimization. In: *IEEE Swarm Intelligence Symposium (SIS 2007)*. pp. 92–99.

- Jiang, M., Luo, Y., Yang, S., 2007b. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. *Information Processing Letters* 102 (1), 8–16. URL <http://www.sciencedirect.com/science/article/pii/S0020019006003103>
- Jordan, J., Helwig, S., Wanka, R., July 2008. Social interaction in particle swarm optimization, the ranked fips, and adaptive multi-swarms. In: in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008)*. Atlanta, Georgia, USA, pp. 49–56.
- Kadirkamanathan, V., Selvarajah, K., Fleming, P., June 2006. Stability analysis of the particle dynamics in particle swarm optimizer. *IEEE Transactions on Evolutionary Computation* 10 (3), 245–255.
- Kamenetskiy, V., Pyatnitskiy, Y., 1987. An iterative method of lyapunov function construction for differential inclusions. *Systems & Control Letters* 8 (5), 445–451. URL <http://www.sciencedirect.com/science/article/pii/0167691187900855>
- Kameyama, K., 2009. Particle swarm optimization - a survey. *IEICE Transactions on Information and Systems* E92-D (7), 1354–1361.
- Kennedy, J., Eberhart, R., October 1997. A discrete binary version of the particle swarm algorithm. In: *IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation*. Vol. 5. pp. 4104–4108.
- King, C., Nathanson, M., 2006. On the existence of a common quadratic lyapunov function for a rank one difference. *Linear Algebra and its Applications* 419 (2-3), 400–416. URL <http://www.sciencedirect.com/science/article/pii/S0024379506002473>
- King, C., Shorten, R., 2006. Singularity conditions for the non-existence of a common quadratic lyapunov function for pairs of third order linear time invariant dynamic systems. *Linear Algebra and its Applications* 413 (1), 24–35. URL <http://www.sciencedirect.com/science/article/pii/S002437950500385X>
- Laffey, T. J., Šmigoc, H., 2009. Common lyapunov solutions for two matrices whose difference has rank one. *Linear Algebra and its Applications* 431 (1-2), 228–240. URL <http://www.sciencedirect.com/science/article/pii/S0024379509001116>
- Leong, W.-F., Yen, G., October 2008. Pso-based multiobjective optimization with dynamic population size and adaptive local archives. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 38 (5), 1270–1293.
- Lian, Z., Zhu, F., Guan, Z., Shao, X., June 25-27 2008. The analysis of particle swarm optimization algorithm's convergence. In: *7th World Congress on Intelligent Control and Automation (WCICA 2008)*. pp. 623–626.
- Liang, J., Qin, A., Suganthan, P., Baskar, S., June 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation* 10 (3), 281–295.
- Liang, S., Song, S., Kong, L., Cheng, J., January 2010. An improved particle swarm optimization algorithm and its convergence analysis. In: *Second International Conference on Computer Modeling and Simulation (ICCMS '10)*. Vol. 2. pp. 138–141.
- Liberzon, D., 2003. *Switching in Systems and Control*. Foundations & Applications Series. Systems & Control, Boston, MA.

- Liberzon, D., Tempo, R., December 2003. Gradient algorithms for finding common lyapunov functions. In: Proceedings of the 42nd IEEE Conference on Decision and Control. Vol. 5. pp. 4782–4787.
- Liberzon, D., Tempo, R., June 2004. Common lyapunov functions and gradient algorithms. *IEEE Transactions on Automatic Control* 49 (6), 990–994.
- Lin, H., Antsaklis, P., February 2009. Stability and stabilizability of switched linear systems: A survey of recent results. *IEEE Transactions on Automatic Control* 54 (2), 308–322.
- Liu, H., Abraham, A., Snasel, V., December 2009. Convergence analysis of swarm algorithm. In: World Congress on Nature Biologically Inspired Computing (NaBIC 2009). pp. 1714–1719.
- Liu, J., Liu, H., Shen, W., 2007a. Stability analysis of particle swarm optimization. In: Huang, D.-S., Heutte, L., Loog, M. (Eds.), *Advanced Intelligent Computing Theories and Applications, With Aspects of Artificial Intelligence*. Vol. 4682 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 781–790.
URL http://dx.doi.org/10.1007/978-3-540-74205-0_82
- Liu, L., Cartes, D., Liu, W., July 2007b. Particle swarm optimization based parameter identification applied to pmsm. In: American Control Conference, 2007. ACC '07. pp. 2955–2960.
- Liu, L., Liu, W., Cartes, D., Zhang, N., June 2008. Real time implementation of particle swarm optimization based model parameter identification and an application example. In: IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). pp. 3480–3485.
- Liu, W., Duan, Y., Shao, K., Wang, K., August 2007c. Chaotic synchronization of adaptive inverse control based on hybrid particle swarm optimization algorithm. In: International Conference on Mechatronics and Automation, 2007 (ICMA 2007). pp. 2409–2413.
- Mason, O., Shorten, R., June 2003. A conjecture on the existence of common quadratic lyapunov functions for positive linear systems. In: Proceedings of the 2003 American Control Conference. Vol. 5. Denver, CO, pp. 4469–4470.
- Mei, C., Liu, G., Xiao, X., May 2010. Improved particle swarm optimization algorithm and its global convergence analysis. In: Chinese Control and Decision Conference (CCDC 2010). pp. 1662–1667.
- Mendes, R., Kennedy, J., Neves, J., June 2004. The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation* 8 (3), 204–210.
- Moldovan, M., Seetharama, M., 2009. *Nonlinear Analysis and Variational Problems*. Vol. 35. Springer, Ch. Chapter 25 - On Common Linear/Quadratic Lyapunov Functions for Switched Linear Systems, pp. 415–429.
- Mori, Y., Mori, T., Kuroe, Y., December 1997. A solution to the common lyapunov function problem for continuous-time systems. In: Proceedings of the 36th IEEE Conference on Decision and Control. Vol. 4. San Diego, CA, pp. 3530–3531.
- Narendra, K., Annaswamy, A., 1989. *Stable Adaptive Systems*. Prentice Hall, NJ.
- Narendra, K., Balakrishnan, J., December 1994. A common lyapunov function for stable lti systems with commuting a-matrices. *IEEE Transactions on Automatic Control* 39 (12), 2469–2471.

- Nguyen, T., Mori, Y., Mori, T., Kuroe, Y., 2003. Qe approach to common lyapunov function problem. *Journal of Japan Society for Symbolic and Algebraic Computation* 10, 52–62.
- Norouzzadeh, M., Ahmadzadeh, M., Palhang, M., July 9-11 2010. Plowing pso: A novel approach to effectively initializing particle swarm optimization. In: 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), 2010. Vol. 1. pp. 705–709.
- Ordóñez-Hurtado, R., Duarte-Mermoud, M., December 19-21 2011a. Design of adaptive laws for discrete-time systems based on particle swarm optimization. In: 9th IEEE International Conference on Control and Automation (ICCA). Santiago, Chile, pp. 883–888.
- Ordóñez-Hurtado, R., Duarte-Mermoud, M., August 29 - September 1 2011b. A methodology for determining the non-existence of common quadratic lyapunov functions for pairs of stable systems. In: Fifth International Conference on Genetic and Evolutionary Computing (ICGEC). Kinmen - Taiwan, Xiamen - China, pp. 127–130.
- Ordóñez-Hurtado, R., Duarte-Mermoud, M., 2012. Finding common quadratic lyapunov functions for switched linear systems using particle swarm optimisation. *International Journal of Control* 85 (1), 12–25.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207179.2011.637133>
- Owen, A. B., 1998. Monte carlo extension of quasi-monte carlo. In: Proceedings of the 30th conference on Winter simulation (WSC '98). IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 571–578.
URL <http://dl.acm.org/citation.cfm?id=293172.293278>
- Ozcan, E., Mohan, C., 1998. Analysis of a simple particle swarm optimization system. *Intelligent Engineering Systems Through Artificial Neural Networks* 8, 253–258.
- Ozcan, E., Mohan, C., 1999. Particle swarm optimization: Surfing the waves. In: Proceedings of the IEEE Congress on Evolutionary Computation. Washington, DC, pp. 1939–1944.
- Parsopoulos, K. E., Tasoulis, D. K., Vrahatis, M. N., 2004. Multiobjective optimization using parallel vector evaluated particle swarm optimization. In: In Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA 2004). ACTA Press, pp. 823–828.
- Particle Swarm Central, 2007. Standard pso 2007 (spso-2007). Tech. rep., Particle Swarm Central, Programs Section. [Online]. <http://www.particleswarm.info>.
- Paul, A., Akar, M., Safonov, M., Mitra, U., June 30 - July 2 2004. Necessary and sufficient conditions for stability of a class of second order switched systems. In: Proceedings of the 2004 American Control Conference. Vol. 5. pp. 4561–4562.
- Peer, E., Van Den Bergh, F., Engelbrecht, A., April 2003. Using neighbourhoods with the guaranteed convergence pso. In: Proceedings of the 2003 IEEE Swarm Intelligence Symposium (SIS 03). pp. 235–242.
- Pérez-López, J., 2005. Contribución a los métodos de optimización basados en procesos naturales y su aplicación a la medida de antenas en campo próximo. Ph.D. thesis, Universidad de Cantabria, Departamento de Ingeniería de Comunicaciones, España.
- Poli, R., 2008. Analysis of the publications on the applications of particle swarm optimisation. *Journal of Artificial Evolution and Applications* 2008, 1–10.

- Poli, R., Brattonx, D., Blackwell, T., Kennedy, J., September 2007. Theoretical derivation, analysis and empirical evaluation of a simpler particle swarm optimiser. In: IEEE Congress on Evolutionary Computation (CEC 2007). pp. 1955–1962.
- Qingqing, Z., Xingshi, H., Na, S., December 2009. Convergence analysis and parameter select on pso. In: Second International Symposium on Information Science and Engineering (ISISE 2009). pp. 144–147.
- Rahmat-Samii, Y., October 1-3 2003. Genetic algorithm (ga) and particle swarm optimization (pso) in engineering electromagnetics. In: 17th International Conference on Applied Electromagnetics and Communications, 2003 (ICECom 2003). Dubrovnik, Croatia, pp. 1–5.
- Rapaić, M. R., Željko Kanović, 2009. Time-varying pso - convergence analysis, convergence-related parameterization and new parameter adjustment schemes. *Information Processing Letters* 109 (11), 548–552.
URL <http://www.sciencedirect.com/science/article/pii/S0020019009000350>
- Ratnaweera, A., Halgamuge, S., Watson, H. C., June 2004. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Trans. Evol. Comput.* 8 (3), 240–255.
- Ren, Z., Wang, J., Zhang, H., September 2008. A new particle swarm optimization algorithm and its convergence analysis. In: Second International Conference on Genetic and Evolutionary Computing (WGEC '08). pp. 319–323.
- Reyes-Sierra, M., Coello, C., 2006. Multi-objective particle swarm optimisers: A survey of the state-of-the-art. *International Journal of Computational Intelligence Research* 2 (3), 287–308.
- Samal, N., Konar, A., Das, S., Abraham, A., September 2007. A closed loop stability analysis and parameter selection of the particle swarm optimization dynamics for faster convergence. In: IEEE Congress on Evolutionary Computation (CEC 2007). pp. 1769–1776.
- Samal, N., Konar, A., Nagar, A., September 2008. Stability analysis and parameter selection of a particle swarm optimizer in a dynamic environment. In: Second UKSIM European Symposium on Computer Modeling and Simulation (EMS '08). pp. 21–27.
- Sanchez-Alvarez, M. S., 2009. Modelación y control de un sistema piloto de calentamiento de fluidos por inducción magnética. Master's thesis, Universidad de Chile, Santiago, Chile.
- Semnani, A., Kamyab, M., Rekanos, I., Octubre 2009. Reconstruction of one-dimensional dielectric scatterers using differential evolution and particle swarm optimization. *IEEE Geoscience and Remote Sensing Letters* 6 (4), 671–675.
- Sen, D., Dey, S., December 2007. A closed loop stability analysis for the particle swarm optimization dynamics. In: International Conference on Information and Communication Technology in Electrical Sciences (ICTES 2007). pp. 69–74.
- Shi, Y., Eberhart, R., May 4-9 1998a. A modified particle swarm optimizer. In: IEEE International Conference on Evolutionary Computation. Anchorage, AK, pp. 69–73.
- Shi, Y., Eberhart, R., May 1998b. Parameter selection in particle swarm optimization. In: in Proc. 7th Int. Conf. Evol. Program. (EP), 1998,. pp. 591–600.

- Shi, Y., Eberhart, R., July 1999. Empirical study of particle swarm optimization. In: in Proc. IEEE Congr. Evol. Comput., 1999. Vol. 3. pp. 1945–1950.
- Shi, Y., Eberhart, R., 2001. Fuzzy adaptive particle swarm optimization. In: Proceedings of the 2001 Congress on Evolutionary Computation, 2001. Vol. 1. pp. 101–106.
- Shorten, R., Narendra, K., December 16-18 1998. On the stability and existence of common lyapunov functions for stable linear switching systems. In: Proceedings of the 37th IEEE Conference on Decision and Control, 1998. Vol. 4. Tampa, FL, pp. 3723–3724.
- Shorten, R., Narendra, K., 2002. Necessary and sufficient conditions for the existence of a common quadratic lyapunov function for a finite number of stable second order linear time-invariant systems. *International Journal of Adaptive Control Signal Process* 16, 709–728.
- Shorten, R., Narendra, K., April 2003. On common quadratic lyapunov functions for pairs of stable lti systems whose system matrices are in companion form. *IEEE Transactions on Automatic Control* 48 (4), 618–621.
- Shorten, R., Narendra, K., Mason, O., January 2003. A result on common quadratic lyapunov functions. *IEEE Transactions on Automatic Control* 48 (1), 110–113.
- Shorten, R., O’Cairbre, F., 2001. A proof of global attractivity for a class of switching systems using a non-lyapunov approach. *IMA Journal of Mathematical Control and Information* 18, 341–353.
- Shorten, R., Wirth, F., Mason, O., Wulff, K., King, C., 2007. Stability criteria for switched and hybrid systems. *SIAM Review* 49, 545–592.
- Shorten, R. N., Mason, O., O’Cairbre, F., Curran, P., 2004. A unifying framework for the siso circle criterion and other quadratic stability criteria. *International Journal of Control* 77 (1), 1–8.
URL <http://www.tandfonline.com/doi/abs/10.1080/00207170310001633321>
- Singh, J., 2003. The pso toolbox. Tech. rep., Source Forge. Available: <http://sourceforge.net/projects/psotoolbox>.
- Solis, F. J., Wets, R. J.-B., February 1981. Minimization by random search techniques. *Mathematics of Operations Research* 6 (1), 19–30.
- Storn, R., Price, K., 1995. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Tech. rep., International Computer Science Institute, [Online]. Available: <ftp://ftp.icsi.berkeley.edu/pub/techreports/1995/tr-95-012.pdf>.
- Su, X., Zhao, J., Sun, J., November 7-8 2009. Online system identification based on quantum-behaved particle swarm optimization algorithm. In: *International Conference on Web Information Systems and Mining (WISM)*. pp. 475–479.
- Sun, C., Zeng, J., Pan, J., 2011. An improved vector particle swarm optimization for constrained optimization problems. *Information Sciences* 181 (6), 1153–1163.
- Sun, J., Feng, B., Xu, W., June 2004. Particle swarm optimization with particles having quantum behavior. In: *Congress on Evolutionary Computation (CEC2004)*. Vol. 1. pp. 325–331.
- Tang, Z., Bagchi, K. K., November 2010. Globally convergent particle swarm optimization via branch-and-bound. *Computer and Information Science* 3 (4), 60–71.
- Tao, G., 2003. *Adaptive Control Design and Analysis*. John Wiley & Sons Inc.

- Tempo, R., Calafiore, G., Dabbene, F., 2004. Randomized Algorithms for Analysis and Control of Uncertain Systems. Springer-Verlag, London, UK.
- Trejo, J. R., Yu, W., Li, X., September 6-8 2006. Support vector machine for nonlinear system online identification. In: 3rd International Conference on Electrical and Electronics Engineering. pp. 1–4.
- Trelea, I. C., 2003. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters* 85 (6), 317–325.
URL <http://www.sciencedirect.com/science/article/pii/S0020019002004477>
- Van Den Bergh, F., 2002. An analysis of particle swarm optimisers. Phd thesis, University of Pretoria, Department of Computer Science, Pretoria, South Africa.
- Van Den Bergh, F., Engelbrecht, A., October 2002. A new locally convergent particle swarm optimiser. In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 3. p. 6 p.
- van den Bergh, F., Engelbrecht, A., June 2004. A cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation* 8 (3), 225–239.
- von Borries Segovia, M. A., Marzo 2012. Estudio y simulación de sistemas adaptables fraccionarios. Mejoramiento de operaciones de biolixiviación de minerales de cobre y electro-obtención en plantas a gran altura mediante calentamiento de soluciones por inducción magnética, Proyecto FONDEF, Santiago, Chile.
- Wakasa, Y., Tanaka, K., Akashi, T., June 2009. Stability and l2 gain analysis for the particle swarm optimization algorithm. In: *American Control Conference (ACC '09)*. pp. 1748–1753.
- Wang, Y., Wang, K., Qu, J., Yang, Y., July 29 - August 1 2005. Adaptive inverse control based on particle swarm optimization algorithm. In: *IEEE International Conference on Mechatronics and Automation, 2005*. Vol. 4. pp. 2169–2172.
- Weise, T., 2008. *Global Optimization Algorithms - Theory and Application*, 2nd Edition. [Online] Available: <http://www.it-weise.de/>.
- Yanami, H., Anai, H., 2005. Development of synrac. In: *International Conference on Computational Science*. Vol. 3. pp. 602–610.
- Yasuda, K., Ide, A., Iwasaki, N., October 2003. Adaptive particle swarm optimization. In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 2. pp. 1554–1559.
- Yasuda, K., Iwasaki, N., October 2004. Adaptive particle swarm optimization using velocity information of swarm. In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 4. pp. 3475–3481.
- Zhan, Z.-H., Zhang, J., Li, Y., Chung, H.-H., December 2009. Adaptive particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 39 (6), 1362–1381.
- Zhang, W., Li, H., Zhang, Z., Wang, H., October 2008. The selection of acceleration factors for improving stability of particle swarm optimization. In: *Fourth International Conference on Natural Computation (ICNC 08)*. Vol. 1. pp. 376–380.
- Zhao, H., Mao, B., December 2009. An investigation on particle trajectories of pso. In: *International Conference on Information Engineering and Computer Science (ICIECS 2009)*. pp. 1–3.

- Zhao, J., Sun, J., Xu, W., December 2009. Application of online system identification based on improved quantum-behaved particle swarm optimization. In: Second International Symposium on Computational Intelligence and Design, 2009. ISCID '09. Vol. 2. pp. 186–189.
- Zheng, Y.-L., Ma, L.-H., Zhang, L.-Y., Qian, J.-X., December 8-12 2003a. Empirical study of particle swarm optimizer with an increasing inertia weight. In: The 2003 Congress on Evolutionary Computation, 2003 (CEC '03). Vol. 1. pp. 221–226.
- Zheng, Y.-L., Ma, L.-H., Zhang, L.-Y., Qian, J.-X., November 2003b. On the convergence analysis and parameter selection in particle swarm optimization. In: International Conference on Machine Learning and Cybernetics. Vol. 3. pp. 1802–1807.
- Zhihua, C., Jianchao, Z., Xingjuan, C., June 2004. A new stochastic particle swarm optimizer. In: Congress on Evolutionary Computation (CEC2004). Vol. 1. pp. 316–319.
- Zhu, Y., Cheng, D., Qin, H., 2007. Constructing common quadratic lyapunov functions for a class of stable matrices. *Acta Automatica Sinica* 33, 202–204.

Nomenclatura

CLC	convex linear combination
CQLF	common quadratic Lyapunov function
EA	Evolutionary Algorithm
EB	Enclosing Balls
EC	Evolutionary Computation
EIP	excitación instantáneamente persistente
EP	Excitación Persistente
GC-PSO	GCPSO o PSO indistintamente
L-T	Liberzon-Tempo
LMI	linear matrix inequality
MFAC	Model-Free Adaptive Control
MRAC	Model Reference Adaptive Control
NB	Narendra-Balakhrisnan
NG	Normalized Gradient
NLS	Normalized Least-Squares
O-D	Ordóñez-Duarte
PSOcf	PSO constriction factor
PSOiw	PSO inertia weighted
SI	Swarm Intelligence
SPM	static parametric model
SPR	strict positive real