UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA MECÁNICA

TESIS

SINTONIZACIÓN SIMULTÁNEA DE LOS PARÁMETROS DEL REGULADOR AUTOMÁTICO DE TENSIÓN Y DEL ESTABILIZADOR DE SISTEMA DE POTENCIA UTILIZANDO UNA VARIANTE DEL ALGORITMO DE OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS

Para Optar el Grado Académico de Doctor en Ciencias con Mención en Energética

Elaborado por

FRANKLIN ALFREDO CABEZAS HUERTA

Asesores

DR. YURI MOLINA RODRIGUEZ DR. JAIME LUYO KUONG

LIMA - PERÚ

2024

A todos los difusores de conocimiento del mundo.

Agradecimientos

Agradezco a todos aquellos quienes promovieron el desarrollo de este trabajo.

Índice general

Índice de Figuras V

Índice de Tablas VII

1 Introduccíon **1**

1.1 Antecedentes 1

1.2 Problemática 7

1.3 Planteamiento del problema 8

1.3.1 Preguntas 9

1.4 Objetivos de la investigación 10

1.4.1 Objetivo General 10

1.4.2 Objetivos Específicos 10

1.5 Hipótesis de la investigación 10

1.5.1 Hipótesis Específicas 10

1.6 Justificación 11

1.7 Aporte 12

1.8 Estructura del trabajo 12

2 Fundamentación teórica **14**

2.1 Estabilidad de sistemas de potencia 14

2.1.1 Estabilidad angular del rotor 18

2.2 Regulador Automático de Tensión 18

2.2.1 Modelo del Amplificador 19

2.2.2 Modelo del Excitador 20

2.2.3 Modelo del Generador 20

2.2.4 Modelo del Sensor 20

2.3 Análisis de estabilidad de sistemas de control 22

2.4 Análisis de respuesta temporal de sistemas de control 26
2.5 Estabilizador de Sistema de Potencia 31
2.6 Algoritmos de control empleados para la sintonización simultánea de los parámetros del AVR y PSS 33
2.6.1 Algoritmos de Control Óptimo basados en técnicas metaheurísticas 33
2.6.1.1 Algoritmo de Optimización por Enjambre de Partículas (PSO) . . 34
2.6.1.2 Algoritmo de Optimización por Enjambre de Partículas con Decaimiento Lineal (PSO - LD) 36
2.6.1.3 Algoritmo de Optimización por Enjambre de Partículas con Peso Inercial Oscilatorio (PSO - OIW) 37
2.6.1.4 Algoritmo de Optimización por Enjambre de Partículas con Decaimiento Exponencial Oscilatorio (PSO - OED) 38
2.7 Modelo linealizado de la máquina síncrona 39

3 Aplicaciones del método de optimización PSO clásico en los sistemas de potencia 43
3.1 Minimización de pérdidas en sistemas de potencia mediante despacho de potencia reactiva usando Optimización por Enjambre de Partículas 43
3.2 Estimación de parámetros de las líneas de transmisión usando Optimización por Enjambre de Partículas 49

4 Algoritmo de optimización propuesto 55

5 Estudio de casos y Análisis de resultados 58
5.1 Caso de estudio del Sistema Máquina Barra Infinita 59
5.2 Caso de estudio del Sistema IEEE de 9 barras 61

6 Conclusiones y Comentarios Finales 69

Bibliografía 71

A Código DPL: Sistema Máquina - Barra Infinita 74

B Código DPL: Sistema IEEE de 9 barras 90

Índice de Figuras

Figura 2.1 Diagrama de bloques de un sistema de generación eléctrica que incluye un LFC y un AVR [1]. 16
Figura 2.2 Clasificación de la estabilidad [2]. 17
Figura 2.3 Respuesta del ángulo del rotor a una perturbación transitoria [3]. 18
Figura 2.4 Componentes de un AVR [1]. 19
Figura 2.5 Diagrama de bloques del AVR [1]. 21
Figura 2.6 Diagrama de polos y ceros del sistema AVR con $K_A = 12.16$. 26
Figura 2.7 Respuesta a entrada escalón unitario en sistemas de primer orden. 27
Figura 2.8 Señal de respuesta típica de un sistema subamortiguado a una señal de prueba y sus parámetros de desempeño [4]. 29
Figura 2.9 Señal de respuesta a un escalón unitario del sistema AVR con $K_A = 10$. . . . 30
Figura 2.10 Diagrama de bloques de un generador síncrono con un AVR y PSS [5]. . . . 32
Figura 2.11 Señal de respuesta a un escalón unitario del sistema AVR con $K_A = 10$ y PSS. 32
Figura 2.12 Taxonomía de algunos de los algoritmos de optimización metaheurística [6]. 34
Figura 2.13 Factor de peso inercial del PSO - LD. 36
Figura 2.14 Factor de peso inercial del PSO - OIW. 38
Figura 2.15 Factor de peso inercial del PSO - OED. 39
Figura 2.16 Máquina sincrona conectada a una barra infinita por medio de una reactancia. 40

Figura 3.1 Flujograma del PSO aplicado a la minimización de pérdidas de potencia activa [7]. 46
Figura 3.2 Sistema IEEE de 14 barras [7]. 47
Figura 3.3 Pérdida de Potencia Activa vs. Iteración [7]. 48
Figura 3.4 Modelo PI de las líneas de transmisión [8]. 51
Figura 3.5 Identificación de parámetros usando Newton Raphson [8]. 53

Figura 4.1 Factor de peso inercial del PSO - POED. 56

Figura 4.2 Flujograma del algoritmo de optimización propuesto PSO - POED. 57

Figura 5.1 Sistema Máquina Barra Infinita. 59
Figura 5.2 Comparación de los algoritmos aplicados en el SMIB. 60
Figura 5.3 Sistema IEEE de 9 barras [9]. 61
Figura 5.4 POED para el generador G2 del sistema IEEE de 9 barras - Mejor resultado. 62
Figura 5.5 Comparación de resultados de las mejores simulaciones obtenidas al aplicar los algoritmos en el Generador 2 del Sistema de 9 barras. 64
Figura 5.6 POED para el generador G3 del sistema IEEE de 9 barras - Mejor resultado. 65
Figura 5.7 Comparación de resultados de la mejores simulaciones obtenidas al aplicar los algoritmos en el Generador 3 del Sistema de 9 barras. 66

Índice de Tablas

Tabla 3.1 Valores de las variables de control en el caso base [7]. 48

Tabla 5.1 Límites de los parámetros del AVR y PSS del Sistema Máquina Barra Infinita. 59
Tabla 5.2 Parámetros del AVR y PSS obtenidos al aplicar los diferentes algoritmos. . . 60
Tabla 5.3 Parámetros de respuesta temporal obtenidos al aplicar los diferentes algoritmos. 61
Tabla 5.4 Límites de los parámetros del AVR y PSS del Sistema IEEE de 9 barras. . . . 67
Tabla 5.5 Parámetros del AVR y PSS obtenidos al aplicar los diferentes algoritmos. . . 67
Tabla 5.6 Parámetros de la señal de tensión en el Análisis de Respuesta Temporal de ambos generadores usando los diferentes algoritmos. 68

Resumen

En esta tesis se propone un nuevo algoritmo de optimización para el control óptimo de la tensión en los terminales de los generadores síncronos de los sistemas de potencia. Este control se consigue mediante la sintonización simultánea y óptima de los parámetros del Regulador Automático de Tensión (AVR) y del Estabilizador de Sistema de Potencia (PSS), los cuales son dos de los principales controladores empleados en los sistemas de potencia. El algoritmo propuesto se prueba en el Sistema Máquina Barra Infinita (SMIB) y se aplica al Sistema de prueba IEEE de 9 barras, este algoritmo consiste en una variación de la Optimización por Enjambre de Partículas (PSO), en el que se propone una nueva forma de calcular el factor de peso inercial (ω), empleando una función coseno que decrece exponencialmente al aumentar el número de iteraciones para luego crecer y volver a decrecer periódicamente.

En el análisis de la respuesta temporal del sistema controlado se verifica que la aplicación del algoritmo propuesto permite al sistema estabilizarse bajo perturbaciones. Inicialmente, se realizaron pruebas utilizando el PSO clásico para abordar problemas típicos en los sistemas de potencia, lo cual proporcionó una base de comparación y destacó sus limitaciones, tales como el elevado número de iteraciones para conseguir la convergencia y la predisposición a entramparse en óptimos locales al abordar problemas de mayor envergadura. Posteriormente, el algoritmo propuesto fue aplicado a un problema más complejo de sintonización simultánea de los parámetros del AVR y del PSS, demostrando mejores resultados en términos de máximo sobrepaso (M_p), tiempo de establecimiento (t_s), tiempo de subida (t_r) y error en estado estacionario (E_{ss}) en comparación con otros algoritmos. Estas mejoras destacan el rendimiento superior del nuevo algoritmo de optimización en relación con otros algoritmos derivados de PSO, reafirmando su efectividad e introduciendo avances significativos cruciales para la confiabilidad y eficiencia de los sistemas de potencia, particularmente para mantener la estabilidad y el rendimiento en condiciones operativas variables.
Palabras - clave: Regulador Automático de Tensión, Estabilizador de Sistema de Potencia, Control Óptimo, Optimización por Enjambre de Partículas.

ABSTRACT

In this thesis, a new optimization algorithm is proposed for the optimal control of the voltage at the terminals of the synchronous generators of the power systems. This control is achieved by the simultaneous and optimal tuning of the parameters of the Automatic Voltage Regulator (AVR) and the Power System Stabilizer (PSS), which are two of the main controllers used in power systems. The proposed algorithm is tested in the Single Machine Infinite Bus (SMIB) system and is applied to the IEEE 9-bar test system, this algorithm consists of a variation of Particle Swarm Optimization (PSO), in which a new way to calculate the inertial weight factor (ω) is proposed, using a cosine function that decreases exponentially as the number of iterations increases, then grows and decreases again periodically.

In the analysis of the temporal response of the controlled system, it is verified that the application of the proposed algorithm allows the system to stabilize under disturbances. Initially, classical PSO was tested to address typical problems in power systems, serving as a benchmark and highlighting its limitations, such as the high number of iterations to achieve convergence and the predisposition to get stuck in local optima when addressing larger problems. Subsequently, the proposed algorithm was applied to a more complex problem of simultaneous tuning of the AVR and PSS parameters, demonstrating better results in terms of maximum overshoot (M_p), settling time (t_s), rise time (t_r) and steady-state error (E_{ss}) compared to other algorithms. These improvements highlight the superior performance of the new optimization algorithm relative to other PSO-derived algorithms, reaffirming its effectiveness and introducing significant advancements crucial for the reliability and efficiency of power systems, particularly for maintaining stability and performance under varying operational conditions.

Keywords: Automatic Power Regulator, Power System Stabilizer, Optimal Control, Particle Swarm Optimization.

Capítulo 1

Introduccíon

1.1. Antecedentes

Los controladores Regulador Automático de Tensión (AVR) y Estabilizador de Sistema de Potencia (PSS) son esenciales para la correcta operación de los sistemas de potencia, ya que permiten mantener los valores requeridos de la tensión en los terminales de los generadores síncronos aun en presencia de perturbaciones, que bajo otras circunstancias afectarían severamente la seguridad y confianza de estos sistemas. Durante los últimos años se vienen dando a conocer nuevos métodos de optimización basados en algoritmos de Inteligencia Artificial, empleados para hallar los mejores parámetros de los controladores AVR y PSS.

Rodrigues et al. propusieron un nuevo algoritmo para la sintonización simultánea de los parámetros óptimos de los controladores AVR y PSS, basado en la técnica de Optimización por Enjambre de Partículas con Decaimiento Exponencial Oscilatorio (PSO – OED). El algoritmo propuesto fue aplicado al sistema de potencia Sistema Máquina Barra Infinita (SMIB) y al sistema de potencia IEEE de 9 barras. Los resultados de aplicar este algoritmo fueron comparados con los obtenidos al aplicar los algoritmos de Optimización por Enjambre de Partículas con Decaimiento Lineal (PSO – LD) y Optimización por Enjambre de Partículas con Peso Inercial Oscilatorio (PSO - OIW). Al igual que con estos dos algoritmos, con PSO – OED los autores propusieron una nueva forma de calcular el factor de peso inercial (ω) del algoritmo PSO, para ello, con este nuevo algoritmo, emplearon una señal cosenoidal con decrecimiento exponencial. Para evaluar la eficiencia del método propuesto, los autores aplicaron a los sistemas controlados una señal de entrada de tipo escalón unitario y analizaron 4 parámetros de las respuestas temporales:

Máximo sobrepaso (M_p), error en estado estacionario (E_{ss}), tiempo de establecimiento (t_s) y tiempo de subida (t_r). Los parámetros de los controladores fueron optimizados empleando el algoritmo propuesto para minimizar las oscilaciones en los sistemas de potencia durante perturbaciones por medio de la maximización de su función objetivo, la cual tiene como variables a los 4 parámetros de las respuestas temporales y las restricciones fueron los límites superior e inferior de los parámetros del AVR y del PSS. Dado que en el sistema SMIB el generador es termoeléctrico y en el sistema IEEE de 9 barras los generadores son hidráulicos, los valores de los límites superior e inferior de los parámetros son diferentes. Los resultados obtenidos al aplicar el algoritmo PSO - OED fueron superiores a los obtenidos al aplicar los algoritmos PSO - LD y PSO - OID, esta mejora fue cuantificada mediante la desviación estándar de las simulaciones, la cual fue mucho menor al aplicar el algoritmo propuesto. Asimismo, se validó la pertinencia del uso de este algoritmo en comparación a los demás, ya que en la respuesta temporal a una señal escalón unitario del sistema controlado, se obtuvo un menor valor de M_p y se consiguieron valores adecuados del e_{ss}, t_s y t_r [9].

Manuaba et al. aplicaron un método de sintonización simultánea óptima de los parámetros de un controlador PID aplicado a un AVR y a un PSS con la finalidad de mejorar la regulación de tensión y la amortiguación de los generadores síncronos en los sistemas de potencia. Este método está basado en la combinación de las técnicas de optimización de Forraje Bacterial y Optimización por Enjambre de Partículas con coeficiente de aceleración variante en el tiempo y Evolución Diferencial (BFO - PSOTVAC - DE). Como parte de su propuesta, los autores formularon nuevas expresiones para el cálculo de los coeficientes de aceleración c_1, c_2 y del factor de peso inercial ω. Este algoritmo fue aplicado a un sistema SMIB y los resultados fueron comparados con los obtenidos al aplicar un sistema SMIB controlado con AVR sin PSS, un sistema SMIB controlado con AVR y un PSS convencional (CPSS), un sistema SMIB controlado con AVR y PSS basados en un PID, un sistema SMIB controlado con AVRs y PSSs basados en PID con parámetros de control optimizados con PSO, un sistema SMIB controlado con AVR y PSS basados en PID con parámetros de control optimizados con BFO y un sistema SMIB controlado con AVR y PSS basados en un PID con parámetros de control optimizados con BFO y PSO. Los resultados muestran que al aplicar el método propuesto, el Máximo sobrepaso (M_p) y el tiempo de establecimiento (t_s) son menores en comparación con los obtenidos al usar los otros algoritmos [10].

Usman, Mustafa y Aliyu emplearon el Algoritmo de Optimización por Enjambre de Partículas de Iteración (IPSO) para la sintonización simultánea de los parámetros óptimos del AVR y del PSS con la finalidad de minimizar las oscilaciones en los sistemas de potencia durante perturbaciones. Los autores aplicaron este algoritmo a un sistema SMIB y compararon los resultados con los obtenidos al aplicar PSO. A diferencia del algoritmo PSO, en el algoritmo IPSO se añaden nuevos parámetros para el cálculo de la velocidad y posición de las partículas en cada iteración, estos nuevos parámetros son: el mejor valor de la función *fitness* alcanzado por cualquier partícula en cada iteración (Ibest) y la constante de aceleración dinámica (c_3). En los resultados de las simulaciones, se observa que en comparación con el algoritmo PSO, el algoritmo IPSO consigue un menor valor de Máximo sobrepaso (M_p), un menor tiempo de establecimiento (t_s) y que las partículas convergen en menor cantidad de iteraciones [11].

Nirmal y Auxillia emplearon los algoritmos PSO, PSO simplificado (MOL) y PSO adaptativo (APSO), así como el método de Ziegler - Nichols para sintonizar los parámetros de un controlador PID aplicado a un AVR. Para obtener los valores óptimos de los parámetros del PID, los autores minimizaron una función objetivo, la cual es un índice de desempeño, conocido como Integral del Error Absoluto por el Tiempo (ITAE), luego compararon los valores de los parámetros obtenidos al aplicar las tres técnicas de optimización usadas para minimizar esta función, además de los parámetros obtenidos al usar el método de Ziegler - Nichols y realizaron el Análisis de la Respuesta Temporal del sistema controlado. Los resultados mostraron que el algoritmo APSO es superior a las otras técnicas de sintonización aplicadas, ya que produjo un menor sobrepaso (M_p) y menor tiempo de establecimiento (t_s). Adicionalmente, los autores realizaron un análisis de convergencia para determinar el tiempo computacional de los algoritmos de optimización PSO, MOL y APSO, hallando que el algoritmo APSO converge con mayor rapidez. Finalmente, los autores aplicaron el Análisis del Lugar de las Raíces y el Análisis de Bode para verificar la estabilidad del sistema AVR controlado, comparando cada uno de las cuatro propuestas y hallaron que con todas ellas, el sistema AVR controlado es estable [12].

Selvabala y Devaraj aplicaron el algoritmo de Evolución Diferencial (DE) para la sintonización óptima y simultánea de los parámetros del controlador de un AVR y un PSS. Esta técnica fue simulada para un sistema SMIB bajo dos condiciones: normal y con falla en la terminal del generador síncrono. Adicionalmente, los autores aplicaron Algoritmos Genéticos (GA) con la finalidad

de aplicarlo para el mismo propósito y comparar el desempeño de ambos. Como función objetivo, los autores emplearon el Índice de Amortiguamiento, esta función fue minimizada aplicando por separado ambos algoritmos y obteniendo con ellos los valores óptimos de los parámetros del AVR y del PSS. Al realizar el Análisis de Respuesta Temporal para la condición normal del generador, los resultados obtenidos mostraron que el algoritmo DE fue superior a GA, ya que los valores de M_p, e_{ss}, t_r y t_s fueron menores; para el caso de la condición con falla, el valor de Mp obtenido al aplicar DE fue mayor que el obtenido al aplicar GA, el valor de t_r obtenido fue casi similar con ambos algoritmos y los valores de t_s y e_{ss} fueron menores al emplear DE, por lo que en promedio para este caso, el algoritmo DE también fue superior. Finalmente, se observó que el algoritmo DE alcanzó la solución óptima en menor tiempo computacional que aplicando el algoritmo GA [13].

Mitra et al. presentaron el diseño de un AVR y PSS basado en el Sistema de Inferencia Difuso Neuro Adaptivo (ANFIS), simulándolo para un sistema SMIB y para un Sistema de Potencia Multi Máquinas (cuatro generadores en dos áreas). Para tal fin, usaron el modelo difuso de Sugeno, empleando un algoritmo de aprendizaje híbrido, el cual es una combinación del Estimador de Mínimos Cuadrados y el Método de la Retropropagación de Errores. Como señales de entrada del controlador propuesto, los autores emplearon la señal de error de tensión y su derivada, además emplearon siete variables lingüísticas para cada una de las variables de entrada y emplearon funciones de membresía gaussiana para definir el grado de pertenencia de las variables de entrada. Para verificar la eficiencia del método propuesto, los autores también aplicaron Lógica Difusa para el diseño del AVR y PSS y compararon los resultados del método propuesto con el método alternativo, adicionando también perturbaciones al sistema. Al comparar los resultados, se obtuvo que el método propuesto presenta mejores valores de los parámetros al efectuar el Análisis de Respuesta Temporal ante una señal escalón unitario [14].

Khezri y Bevrani usaron Lógica Difusa para sintonizar las ganancias de un AVR y de un PSS y así diseñar un controlador difuso. Las señales de entrada de este controlador fueron la desviación de la tension terminal y la diferencia de fase, mientras que las señales de salida fueron las ganancias. Además, evaluaron el desempeño del controlador propuesto, considerando también la presencia de fallas en el sistema, por lo que normalizaron ambas señales en el intervalo de [-1, 1]. Los autores eligieron cinco funciones de membresía trapezoidales para las señales de entrada y cuatro funciones de membresía triangulares para las variables de salida.

El algoritmo propuesto fue aplicado a un sistema de potencia de prueba de once barras y de dos áreas, con dos generadores en cada área. El analisis de respuesta transitoria mostró que el algoritmo propuesto permite estabilizar al sistema después de la presencia de fallas [15].

Selvabala y Devaraj aplicaron la técnica de Algoritmos Genéticos Multi Objetivo (MOGA) para la sintonía óptima y simultánea de los parámetros de un AVR y un PSS. Esta técnica fue simulada para un sistema SMIB bajo condiciones de operación normal y con falla. Para obtener los valores óptimos de los parámetros, los autores minimizaron dos funciones objetivo con restricciones: Integral del Error al cuadrado multiplicada por el Tiempo (ITSE) y el Índice de Amortiguamiento (DI). Dado que los autores abordaron la optimización de los parámetros como un problema multiobjetivo, analizaron el Frente de Pareto, obteniendo hasta 28 soluciones óptimas de Pareto. De la curva del Frente Óptimo de Pareto, los autores eligieron los puntos extremos que corresponden al valor mínimo del ITSE y al valor mínimo del DI, obteniendo así los valores óptimos de los parámetros de control correspondientes a estos puntos [16].

Spoljaric, Pavic y Alinjak usaron optimización multiobjetivo para sintonizar simultáneamente los parámetros del AVR y PSS. Para ello, emplearon dos funciones objetivo: La primera consistió en la Integral del Valor Absoluto del Error Ponderado multiplicado por el Tiempo (ITAE) de las diferencias de velocidad del rotor, de la tensión del generador y de la transferencia de potencia en las líneas de transmisión. La segunda contempló a la media de los parámetros del análisis de la respuesta temporal, como el máximo sobrepaso y el tiempo de establecimiento de las velocidades del rotor de los generadores. Para resolver el problema de optimización, los autores usaron el Método de No Preferencia y el Método de Suma Ponderada, combinados con cuatro diferentes algoritmos, dos de los cuales son nuevos algoritmos de optimización: Optimizador de Hormiga León Multiobjetivo (MOALO) y el Algoritmo de Enjambre de Salpas Multiobjetivo (MOSSA), y dos algoritmos clásicos: Optimización por Enjambre de Partículas Multiobjetivo con Relajación de Velocidad (MOVRPSO) y Recocido Simulado Multiobjetivo (MOSA). Estos métodos fueron simulados en un Sistema de Potencia de prueba de dos áreas y cuatro máquinas (TAFM), el cual fue modelado en Matlab y en el que se consideraron cuatro casos de pueba de perturbaciones: Corte y Reconexión automática de una de las líneas de transmisión del TAFM, Falla trifásica con cortocircuito a tierra, Inyección de carga adicional y Corte de carga temporal. Se definieron siete parámetros ajustables dentro de ciertos rangos para los controladores AVRs y PSSs de cada ge-

nerador. De estos siete parámetros, dos son de los AVRs y cinco son de los PSSs, que incluyen ganancias y constantes de tiempo, mientras que al resto de parámetros se le asignaron valores fijos. Los resultados obtenidos muestran que el algoritmo MOSA, operando con el Método de No Preferencia para el segundo caso de prueba, fue el que tardó más tiempo en completar la simulación de los cálculos (90.493s) y el algoritmo MOVRPSO, operando con el método de Suma Ponderada para el tercer caso de prueba, fue el que tardó menos tiempo (6.490s), lo que da cuenta de la alta no linealidad del sistema de prueba. Los resultados mostraron también que los nuevos algoritmos propuestos MOALO y MOSSA fueron más eficientes que los algoritmos clásicos MOVRPSO y MOSA, y que el Método de No Preferencia provó ser ligeramente superior que el Método de Suma Ponderada, ya que el primero obtuvo mejores resultados en 9 de 16 casos para cada uno de los cuatro algoritmos evaluados en los cuatro casos de prueba [17].

Dudgeon et al. evaluaron dos efectos contradictorios del uso simultáneo de los AVRs y PSSs en los sistemas de potencia. Los autores analizaron primero el efecto que producen las altas ganancias de los AVRs en la disminución de la estabilidad oscilatoria, así como el aumento de la estabilidad transitoria y viceversa. Además, analizaron el efecto que producen los PSSs en la reducción de la estabilidad transitoria, mientras que estos consiguen aumentar la estabilidad oscilatoria y viceversa. Para tal efecto, los autores emplearon un sistema de prueba de dos áreas y cuatro máquinas (TAFM), el cual fue modelado en el software PSS/E, y el Sistema de Prueba de Nueva Inglaterra, conformado por 10 generadores y 39 barras. Para ello, realizaron el Análisis de Respuesta Frecuencial del AVR de cada generador del TAFM mediante el Diagrama de Bode de las funciones de transferencia correspondientes, observando que a mayor ganancia del AVR, se obtuvieron valores próximos a los 20dB en los diagramas de magnitud y que además las perturbaciones ocurridas alrededor de 3.5 y 6.5 rad/s se observaban en los diagramas de magnitud y fase. Seguidamente, realizaron el Análisis de Respuesta Frecuencial del PSS, valiéndose de su función de transferencia y obteniendo el Diagrama de Bode correspondiente. Los autores observaron que mediante un ajuste de los picos de ganancia, los diagramas de Bode del PSS pueden ser usados para sintonizar los mismos PSSs y que además, al incluir un único PSS en el generador 2, consiguieron amortiguar las oscilaciones del resto de generadores. Posteriormente, realizaron el mismo procedimiento para el segundo sistema de prueba, valiéndose de las funciones de transferencia de los AVRs y PSSs y obteniendo los Diagramas de Bode correspondientes. Finalmente, los autores evaluaron la robustez de su análisis mediante la consideración de dos escenarios para el primer sistema de prueba:

Cambio del flujo de potencia interárea y Migración de modo aislado a modo conectado a una barra infinita. Para el primer escenario, se evaluó el caso nominal del flujo de potencia de 400MW desde el Área 1 al Área 2, el caso de ausencia de flujo de potencia y el caso del flujo de potencia de 400MW desde el Área 2 al Área 1. Los Diagramas de Bode obtenidos para estos casos mostraron que el PSS del generador 2 aún conseguía estabilizar al sistema, pero con un amortiguamiento reducido. Al final, la inclusión de una barra infinita en el primer sistema de prueba, mostró una variación porcentual del nivel de amortiguamiento de las oscilaciones. Los resultados de este trabajo muestran que el Diagrama de Bode es una herramienta útil para sintonizar los PSSs y para determinar la ubicación adecuada de los mismos en los sistemas de potencia, así como para la evaluación de la robustez de los sistemas de potencia ante variaciones de las condiciones de operación [18].

1.2. Problemática

Debido al aumento sostenido de la demanda eléctrica y a los estrictos requerimientos de seguridad y confiabilidad en los sistemas de potencia, cada vez es más necesario contar con sistemas de control que permitan corregir de la mejor manera y en el menor tiempo posible las desviaciones de las señales de tensión en los terminales de sus generadores síncronos, producidos por perturbaciones, sean estas de naturaleza exógena o endógena.

En los sistemas de potencia interconectados, los AVRs y PSSs instalados en cada generador, requieren una sintonización adecuada de sus parámetros para tener un buen desempeño. En el estado del arte existen varias propuestas para este fin, en algunos casos se sintonizan solo los parámetros del AVR y PSS, mientras que en otros casos se adicionan a estos, otros controladores como los PI o PID. Los parámetros de los controladores son sintonizados mediante técnicas clásicas como Ziegler-Nichols o Cohen-Coon, o mediante el uso de técnicas de Inteligencia Artificial como Redes Neuronales, Lógica Difusa, Sistema de Inferencia Difuso Neuro Adaptivo (ANFIS), etc. Otras formas de sintonización están basadas en la aplicación de técnicas de optimización, ya sean clásicas como el Método de Newton - Raphson o la Gradiente Descendiente o empleando algoritmos de Computación Evolutiva, tales como Optimización por Enjambre de Partículas, Recocido Simulado, Algoritmos Genéticos, métodos híbridos, etc., los cuales, emplean funciones multi - objetivo con restricciones que deben ser máximizadas o minimizadas según corresponda.

Otro de los principales problemas de los sistemas de potencia son las pérdidas de potencia activa en las líneas de transmisión eléctrica, las cuales son muy grandes en países con importantes regiones costeras como el Perú, y que por tanto requieren de algoritmos que las minimicen, puesto que estas pérdidas suponen un alto costo de operación de la transmisión, que conlleva a un incremento del costo de las tarifas eléctricas, afectando principalmente a la economía de los consumidores.

Similarmente, un problema adicional de relevancia en los sistemas de potencia es la ausencia de mediciones fidedignas de los parámetros de las líneas de transmisión, ya que por efectos de humedad, contaminación y desgaste que afectan a las líneas, los valores medidos distan sustancialmente de sus valores reales, lo que conlleva a problemas en el despacho económico. Además, ciertos parámetros como la conductancia de fuga son muy difíciles de medir, por lo que el empleo de técnicas de estimación supone una herramienta valiosa para superar estas dificultades.

Entre los algoritmos de Computación Evolutiva que pueden dar solución a los problemas citados, destaca la Optimización por Enjambre de Partículas (PSO), debido a la facilidad de su implementación y a su bajo costo computacional. Sin embargo, el éxito que se obtenga al usar este algoritmo va a depender de cuán bien sean determinados los valores de sus parámetros, caso contrario, se corre el riesgo de caer en óptimos locales y en consecuencia, no hallar el óptimo global.

En los últimos años, se han venido proponiendo variaciones de este algoritmo para el cálculo de algunos de sus parámetros, como el factor de peso inercial (ω), el coeficiente de aceleración individual (c_1) o el coeficiente de aceleración social (c_2), dando lugar a nuevas alternativas de solución que permiten a la comunidad científica y a los operadores de los sistemas de potencia resolver problemas específicos.

1.3. Planteamiento del problema

En los sistemas de potencia es necesario mantener el valor de la tensión terminal de los generadores dentro de márgenes aceptables, ya que de este modo se garantiza la seguridad y confianza que se espera de los mismos. Por tal motivo se incorporan AVRs y PSSs en los generadores, cuyos parámetros son sintonizados mediante la aplicación de diversos algoritmos.

Los algoritmos basados en Computación Evolutiva resultan ser muy eficientes para este propósito, ya que inicializan la búsqueda del óptimo global con varias potenciales alternativas de solución que van mejorando en cada iteración. Uno de los algoritmos más empleados de este tipo es la Optimización por Enjambre de Partículas (PSO), que al igual que otros algoritmos basados en Computación Evolutiva, opera con ecuaciones en los que ciertos parámetros son inicializados aleatoriamente, lo que conlleva a que en ciertos casos la respuesta se entrampe en óptimos locales. Para evitar este problema, se vienen proponiendo variaciones del PSO clásico, en los que se añaden nuevas ecuaciones.

En ese sentido, en esta tesis se propone una nueva variante del algoritmo PSO, llamada Optimización por Enjambre de Partículas con Decaimiento Exponencial Oscilatorio Periódico (PSO - POED), en el que el factor de peso inercial (ω) emplea una función coseno con decrecimiento exponencial con restitución periódica, con el propósito de hallar los valores óptimos de los parámetros de los controladores AVR y PSS, y en consecuencia conseguir mejores valores de los parámetros del Análisis de Respuesta Temporal que los obtenidos usando otros algoritmos variantes del PSO clásico, tales como el PSO - LD, PSO - OIW o PSO - OED, mismos que también son aplicados en esta tesis como algortimos alternativos para resolver el mismo problema.

Además, para dar solución a los problemas de minimización de pérdidas de potencia activa en los sistemas de potencia y estimación de parámetros en las líneas de transmisión, el PSO clásico es aplicado en esta tesis.

1.3.1. Preguntas

- ¿Cómo sintonizar simultáneamente los parámetros del Regulador Automático de Tensión (AVR) y del Estabilizador de Sistema de Potencia (PSS) utilizando una variante del Algoritmo de Optimización por Enjambre de Partículas (PSO)?
- ¿Cómo sintonizar alternativamente de forma simultánea los parámetros del AVR y del PSS en base al uso de otras variantes del algoritmo PSO?
- ¿Cómo minimizar óptimamente las pérdidas de potencia activa en los sistemas de potencia?
- ¿Cómo estimar óptimamente los parámetros de las líneas de transmisión?

1.4. Objetivos de la investigación

1.4.1. Objetivo General

- Desarrollar un nuevo algoritmo variante del PSO clásico que permita la sintonización simultánea de los parámetros del AVR y PSS, de modo que ofrezca una mejora tanto en la estabilidad dinámica como en la convergencia entre sus simulaciones.

1.4.2. Objetivos Específicos

Los objetivos específicos para el desarrollo de esta tesis son:

- Desarrollar y aplicar el nuevo algoritmo PSO propuesto para la sintonización óptima del AVR y del PSS para un sistema SMIB y el sistema IEEE de 9 barras;
- Desarrollar y aplicar los algoritmos variantes del PSO para la sintonización óptima del AVR y del PSS para un sistema SMIB y el sistema IEEE de 9 barras;
- Desarrollar y aplicar el algoritmo PSO clásico para la minimización de pérdidas de potencia activa en el sistema IEEE de 14 barras;
- Desarrollar y aplicar el algoritmo PSO clásico para la estimación de parámetros de 47 líneas de transmisión de alta tensión del sistema eléctrico peruano;

1.5. Hipótesis de la investigación

- La implementación de un nuevo algoritmo variante del PSO clásico permitirá sintonizar simultáneamente los parámetros del AVR y PSS, ofreciendo una mejora tanto en la estabilidad dinámica como en la convergencia entre sus simulaciones.

1.5.1. Hipótesis Específicas

Las hipótesis específicas para el desarrollo de esta tesis son:

- El desarrollo y aplicación del nuevo algoritmo PSO propuesto permitirá sintonizar óptimamente el AVR y el PSS de un sistema SMIB y del sistema IEEE de 9 barras;

- El desarrollo y aplicación de los algoritmos variantes del PSO permitirá sintonizar óptimamente el AVR y el PSS de un sistema SMIB y del sistema IEEE de 9 barras;
- El desarrollo y aplicación del algoritmo PSO clásico en el sistema IEEE de 14 barras, permitirá minimizar sus pérdidas de potencia activa;
- El desarrollo y aplicación del algoritmo PSO clásico, permitirá estimar los parámetros de 47 líneas de transmisión de alta tensión del sistema eléctrico peruano;

1.6. Justificación

Existe actualmente una mayor demanda de energía eléctrica en los sistemas de potencia que cuente con las características de seguridad y confianza, por lo que el desarrollo de algoritmos de sintonización de los parámetros de los dispositivos de control AVR y PSS que sean cada vez más eficientes es una necesidad imperante.

Las técnicas de optimización basadas en computación evolutiva permiten la sintonización óptima de estos parámetros y entre estas técnicas destaca por su buen desempeño la Optimización por Enjambre de Partículas. Por tal motivo, se justifica este trabajo de tesis, ya que en él se desarrolla y aplica un nuevo algoritmo variante del PSO clásico para la sintonización simultánea de estos parámetros.

Las pérdidas de potencia activa en los sistemas de potencia representan uno de los principales problemas del sector eléctrico en muchos países, particularmente en aquellos en los que el clima húmedo es prevalente, ya que por el efecto corona, estas pérdidas transversales pueden ser muy altas, ocasionando incrementos sustanciales en las tarifas eléctricas. El algoritmo de optimización PSO clásico permite reducir estas pérdidas de potencia mediante el despacho óptimo de la potencia reactiva y el control óptimo de las tensiones.

La imprecisión y en muchos casos, la ausencia de parámetros medidos en las líneas de transmisión, representa un problema de relevancia que amerita solución, ya que menoscaba la realización de un adecuado despacho económico. Estos problemas referidos a la determinación certera de los parámetros se debe a la contaminación y corrosión a las que están sometidas las líneas de transmisión costeras en regiones con alta humedad.

El algoritmo de optimización PSO clásico permite estimar estos parámetros mediante la minimización de una función objetivo de error.

1.7. Aporte

El desarrollo de la presente tesis tiene los siguientes aportes significativos en el campo de la optimización de controladores en los sistemas de potencia, específicamente mediante la sintonización simultánea de los parámetros del Regulador Automático de Tensión (AVR) y del Estabilizador de Sistema de Potencia (PSS):

- **Desarrollo de un Nuevo Algoritmo Basado en PSO**: Se introduce un algoritmo innovador basado en la Optimización por Enjambre de Partículas (PSO) clásico usando una variante, diseñado para la sintonización simultánea de los parámetros del AVR y PSS. Este nuevo algoritmo mejora notablemente los resultados en el Análisis de Respuesta Temporal, logrando una mayor convergencia en las simulaciones y reduciendo la distorsión en la señal de tensión terminal del generador síncrono, lo que es crucial para la estabilidad y eficiencia del sistema de potencia.

- **Comparación y Validación del Algoritmo Propuesto**: Se demuestra que el algoritmo propuesto supera a otros algoritmos alternativos también basados en PSO aplicados en sistemas de prueba SMIB e IEEE de 9 barras. Esta validación no solo muestra la superioridad del nuevo algoritmo en términos de desempeño, sino que también aporta al conocimiento en este campo, proporcionando una base sólida para futuras investigaciones y desarrollos.

- **Aplicabilidad en el Sector Eléctrico**: Los resultados obtenidos tienen una relevancia práctica considerable, ya que ofrecen a los profesionales del sector eléctrico una nueva y mejor alternativa para la sintonización simultánea de los parámetros del AVR y PSS. Este avance facilita la implementación de sistemas de control más robustos y eficientes, contribuyendo a la mejora de la estabilidad y confiabilidad de los sistemas de potencia.

1.8. Estructura del trabajo

El presente trabajo de tesis está estructurado de la siguiente manera:

En el **Capítulo 1: Introducción**, se presentan los antecedentes y se revisa el estado del arte de los métodos alternativos propuestos para la sintonización de controladores AVR y PSS, destacando tanto su eficiencia como sus limitaciones. Este capítulo proporciona una comprensión profunda de las diversas técnicas utilizadas y sus fundamentos matemáticos.

El **Capítulo 2: Fundamentación Teórica** presenta los conceptos fundamentales relacionados con la estabilidad de los sistemas de potencia, así como los principios de funcionamiento del AVR y PSS. Además, se desarrolla el modelo matemático necesario para la simulación dinámica de la máquina síncrona y los controladores asociados.

En el **Capítulo 3: Aplicaciones del Método de optimización PSO clásico en los Sistemas Eléctricos de Potencia**, se describen dos aplicaciones del algoritmo PSO clásico en el ámbito de los sistemas de potencia, ilustrando su capacidad para resolver problemas específicos.

El **Capítulo 4: Algortimo de optimización propuesto** introduce una nueva variante del algoritmo PSO clásico, detallando las modificaciones realizadas y su justificación teórica. Este capítulo también incluye el desarrollo del algoritmo y su implementación práctica.

En el **Capítulo 5: Estudio de Casos y Análisis de Resultados**, se aplica el algoritmo de optimización propuesto para determinar los parámetros del AVR y PSS en los sistemas de prueba SMIB e IEEE de 9 barras. Los resultados obtenidos se comparan con aquellos derivados de otros algoritmos alternativos basados en PSO clásico, evaluando su desempeño en términos de estabilidad dinámica y convergencia.

Finalmente, en el **Capítulo 6: Conclusiones y Comentarios Finales**, se presentan las conclusiones obtenidas a partir de la aplicación del nuevo algoritmo propuesto, destacando su eficacia y ventajas sobre los métodos tradicionales, así como las conclusiones obtenidas al aplicar el algoritmo PSO clásico en los dos problemas en los que fue aplicado. Además, se sugieren posibles direcciones para investigaciones futuras basadas en los hallazgos de esta tesis.

Capítulo 2

Fundamentación teórica

En este capítulo se realiza una revisión conceptual de la estabilidad de los sistemas de potencia, particularmente de la estabilidad angular de los rotores bajo perturbaciones. Además, se presentan los conceptos referidos a los controladores AVR y PSS.

2.1. Estabilidad de sistemas de potencia

Un sistema de potencia está conformado básicamente por generadores síncronos, líneas de transmisión y cargas. En régimen permanente, todas las máquinas síncronas tienen la misma velocidad angular y la potencia generada es igual a la suma de las potencias consumidas por las cargas más las pérdidas en la transmisión producidas por efecto Joule, efecto Skin, efecto Corona, etc.

La estabilidad de los sistemas de potencia es la capacidad que les permite permanecer en un estado de equilibrio operativo en condiciones normales de funcionamiento y de recuperar un estado de equilibrio aceptable después de haber sido sometidos a alguna perturbación. La robustez de un sistema de potencia es un indicativo de esta cualidad y para conseguirla, todos los generadores síncronos deben operar en sincronismo, es decir con tensiones y frecuencias constantes o dentro de ciertos límites, por tal motivo es escencial el uso de controladores [1].

Los generadores síncronos cuentan principalmente con dos tipos de controladores: Controlador de Potencia Activa, llamado también Controlador de Velocidad o Controlador de Frecuencia de Carga y Controlador de Potencia Reactiva, llamado también Regulador Automático de Tensión.

El Controlador de Frecuencia de Carga (LFC) recibe constantemente información del error o desviación existente entre la velocidad de rotación deseada (ω_{ref}) y la velocidad de rotación medida de las turbinas de los generadores síncronos (ω). En régimen permanente, el error es cero, ya que la demanda de potencia activa en las cargas (P_L) es igual a la oferta de potencia activa de los generadores (P_G). Sin embargo, si se presentase un incremento de la demanda de potencia activa en la cargas, ocurriría un desbalance, que causaría una disminución de la frecuencia (f), que a su vez provocaría una disminución de la velocidad de rotación de las turbinas, esta relación viene expresada por la ecuación:

$$\omega = 2 * \pi * f \tag{2.1}$$

Cuando el LFC detecta que la frecuencia disminuye, envía una señal de control a los inyectores para que entre más agua, vapor o gas según sea el tipo de turbina, de modo que esta aumente la potencia producida y la velocidad de rotación, restableciendo así su valor de frecuencia. Todo este proceso tarda aproximadamente dos minutos, debido a los componentes mecánicos que participan como actuadores.

El Regulador Automático de Tensión (AVR) recibe constantemente información del error o desviación existente entre la tensión deseada (V_{ref}) y la tensión medida en uno de los terminales de los generadores síncronos (V). En régimen permanente, el error es cero, ya que la demanda de potencia reactiva en las cargas (Q_L) es igual a la oferta de potencia reactiva de los generadores (Q_G). Sin embargo, si se presentase un incremento de la demanda de potencia reactiva en la cargas, ocurriría un desbalance, que causaría una disminución de la tensión en los terminales del generador. La relación entre la potencia reactiva y la tensión viene expresada por la ecuación:

$$Q = V * I * sin(\alpha) \tag{2.2}$$

Cuando el AVR detecta que la tensión en los terminales del generador disminuye, aumenta la cantidad de corriente (I) suministrada al devanado de campo del rotor del generador mediante un sistema de excitación, por lo que aumenta el flujo magnético y la tensión, así como la potencia reactiva (ver Figura 2.1).

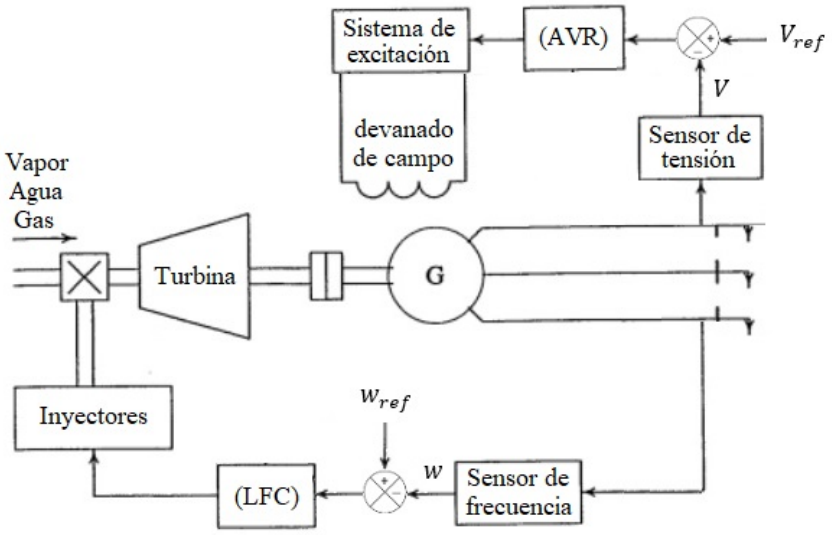


Figura 2.1: Diagrama de bloques de un sistema de generación eléctrica que incluye un LFC y un AVR [1].

Las perturbaciones pueden ser divididas en dos categorías: pequeñas y grandes. Las pequeñas perturbaciones son causadas por leves cambios en la carga o en la generación, así como por aperturas de una línea, siempre y cuando la potencia que fluye sea pequeña. Para este tipo de perturbaciones, los sistemas dinámicos pueden ser analizados usando ecuaciones lineales, su estudio se conoce como Análisis de pequeña señal. Las grandes perturbaciones son debidas a fallas de los componentes, que a su vez causan caídas repentinas en las tensiones de las barras. Para este tipo de perturbaciones, los sistemas dinámicos se analizan mediante ecuaciones diferenciales no lineales que no pueden ser linealizadas.

La estabilidad de los sistemas de potencia se divide en dos clases: Estabilidad a pequeñas perturbaciones y Estabilidad a grandes perturbaciones. Un sistema de potencia es estable a pequeñas o grandes perturbaciones si es que después de una pequeña o gran perturbación respectivamente, el sistema recupera o se aproxima a su condición de operación en estado estacionario.

Las técnicas usadas en el Análisis de Estabilidad a pequeñas perturbaciones están basadas en el Control Lineal (autovalores, autovectores, analisis modal, etc.), mientras que para el Análisis de Estabilidad a grandes perturbaciones se emplean técnicas de Control No Lineal, tales como el Método de Lyapunov.

En el periodo transitorio posterior a una perturbación, el funcionamiento del sistema se torna oscilatorio, causando cambios en las potencias circulantes en las líneas de transmisión. No obstante, si el sistema es estable estas oscilaciones serán fuertemente amortiguadas. Así, la robustez de un sistema de potencia depende en gran medida de las variables de control del sistema para amortiguar las oscilaciones electromecánicas.

En la Figura 2.2 se muestra la clasificación del problema de estabilidad con sus categorías y subcategorías.

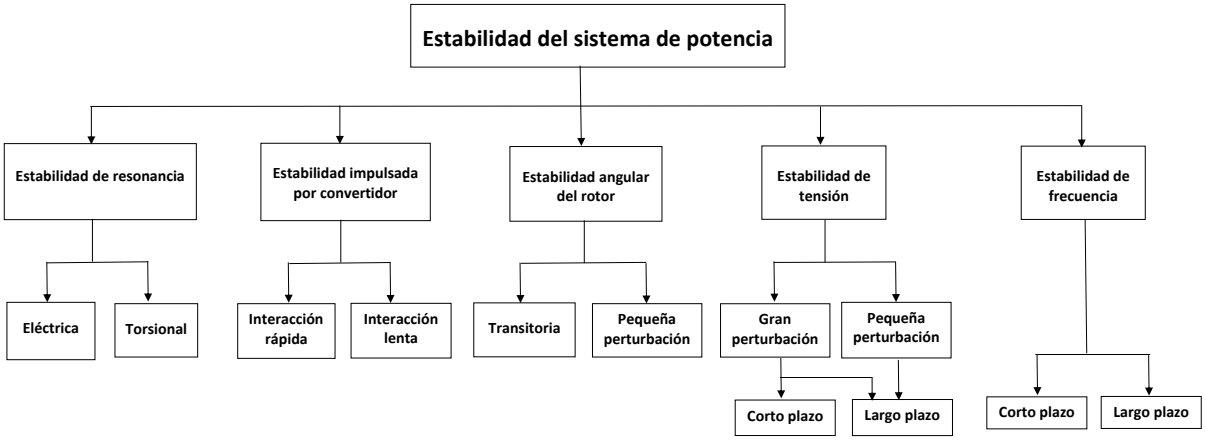


Figura 2.2: Clasificación de la estabilidad [2].

Según Kundur [3], la estabilidad de los sistemas de potencia se analiza según: La naturaleza física de la variable que pierde estabilidad (ángulo, tensión o frecuencia), el tamaño de la perturbación, el cual define el método de cálculo empleado en la previsión de inestabilidad (grandes perturbaciones o pequeñas perturbaciones) y el intervalo de tiempo requerido para evaluar la estabilidad (corto plazo o largo plazo).

2.1.1. Estabilidad angular del rotor

Es la habilidad que tienen las máquinas síncronas interconectadas de un sistema de potencia de permanecer en sincronismo luego de ser sometidas a una perturbación. Bajo condiciones de estado estacionario, existe equilibrio entre el torque mecánico de entrada y el torque eléctrico de salida, de modo que la velocidad angular permanece constante. (ver Figura 2.3).

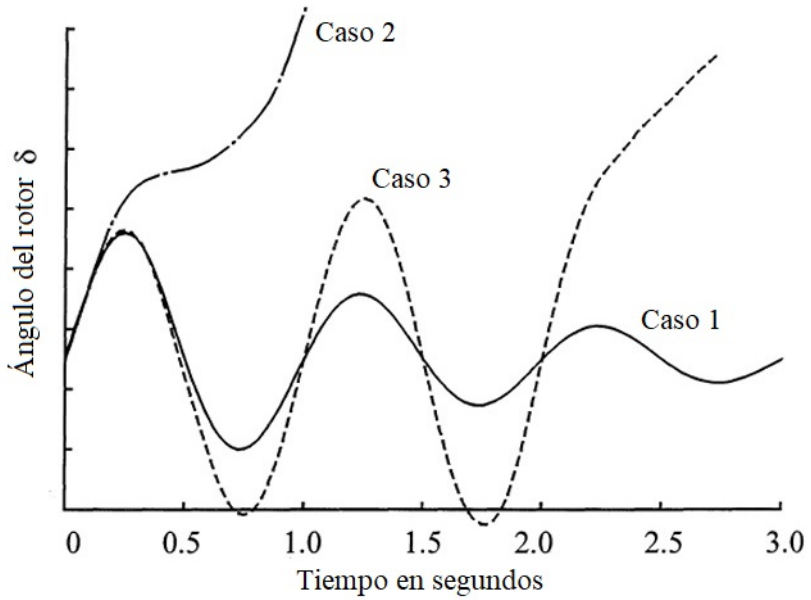


Figura 2.3: Respuesta del ángulo del rotor a una perturbación transitoria [3].

El Caso 1 representa el estado estable, el Caso 2 es un caso inestable, conocido como inestabilidad transitoria, que es causada por una gran perturbación que se presenta en las primeras oscilaciones del generador y el Caso 3 es también un caso inestable, conocido como inestabilidad para pequeñas señales, que es causado por pequeñas perturbaciones.

2.2. Regulador Automático de Tensión

La pérdida de estabilidad en los sistemas de potencia se debe a un desequilibrio entre la generación y la carga, causado por fallas en los generadores, líneas de transmisión, transformadores o incluso debido a una variación brusca en la demanda.

Una forma de evitar estos desequilibrios es equipando a los generadores síncronos con Reguladores Automáticos de Tensión (AVR), cuya función es mantener la magnitud de voltaje en el terminal de un generador síncrono a un valor específico. El diagrama esquemático de un AVR simplificado se muestra en la Figura 2.4.

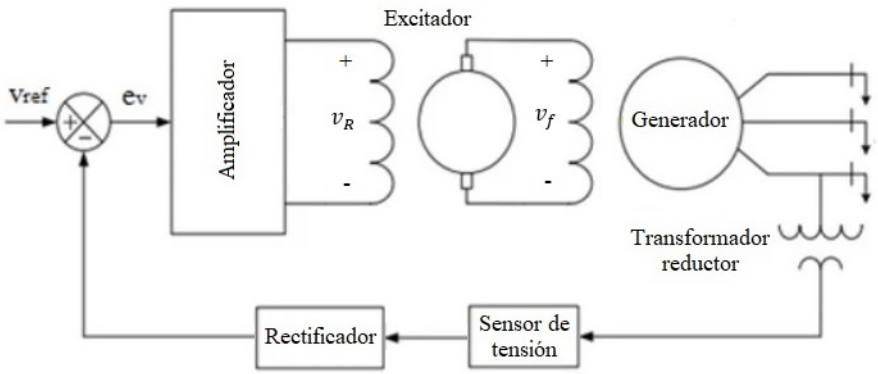


Figura 2.4: Componentes de un AVR [1].

A continuación, se explican los modelos simplificados de los componentes del AVR:

2.2.1. Modelo del Amplificador

El amplificador del sistema de excitación puede ser un amplificador magnético, un amplificador rotativo o un amplificador electrónico. El amplificador es representado por un ganancia K_A y una constante de tiempo τ_A, su función de transferencia es:

$$\frac{V_R(S)}{V_e(S)} = \frac{K_A}{1+\tau_A S} \tag{2.3}$$

Los valores típicos de K_A oscilan entre 10 a 400 y los de τ_A oscilan entre 0.02 a 0.1 segundos.

2.2.2. Modelo del Excitador

Existen muchos tipos de sistemas de excitación, de los cuales uno de los más usados son aquellos basados en SCRs. La función de transferencia de un excitador puede ser representada mediante una constante de tiempo τ_E y una ganancia K_E:

$$\frac{V_F(S)}{V_R(S)} = \frac{K_E}{1+\tau_E S} \tag{2.4}$$

2.2.3. Modelo del Generador

La fuerza electromotriz (fem) generada por la máquina síncrona es función de la curva de magnetización de la máquina y su voltaje terminal depende de la carga del generador. En el modelo linealizado, la función de transferencia que relaciona el voltage en el terminal del generador con su voltage de campo v_f puede representarse mediante una ganancia K_G y una constante de tiempo τ_G:

$$\frac{V_t(S)}{V_F(S)} = \frac{K_G}{1+\tau_G S} \tag{2.5}$$

2.2.4. Modelo del Sensor

El sensor se modela mediante una función de transferencia de primer orden:

$$\frac{V_S(S)}{V_t(S)} = \frac{K_R}{1+\tau_R S} \tag{2.6}$$

El valor de τ_R es muy pequeño y se puede asumir que su valor oscila entre 0.01 y 0.06 segundos. El uso de los modelos anteriores da como resultado el diagrama de bloques del AVR mostrado en la Figura 2.5.

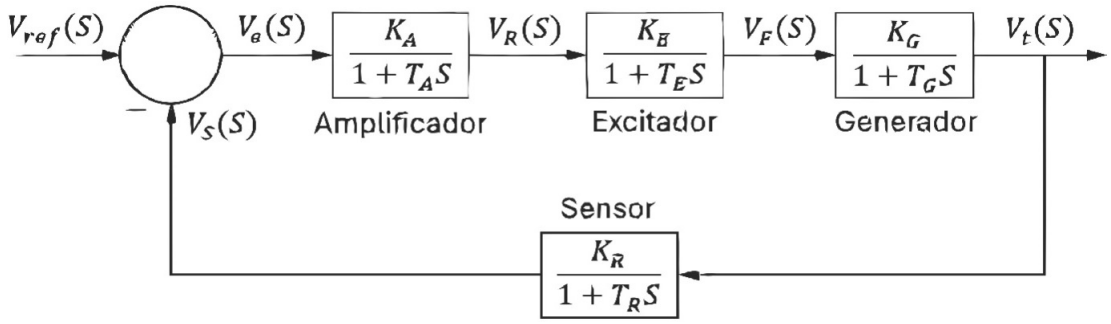


Figura 2.5: Diagrama de bloques del AVR [1].

La función de transferencia a lazo abierto del diagrama de bloques es:

$$G(S)H(S) = \frac{K_A K_E K_G K_R}{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)(1+\tau_R S)} \tag{2.7}$$

Para hallar la función de transferencia a lazo cerrado del diagrama de bloques, se emplean las ecuaciones del Álgebra de Bloques, así:

Para tres bloques en serie $G_1(S)$, $G_2(S)$ y $G_3(S)$:

$$G_{eq}(S) = G_1(S)G_2(S)G_3(S) \tag{2.8}$$

Para dos bloques en realimentación $G(S)$ y $H(S)$:

$$G_{eq}(S) = \frac{G(S)}{1 \pm G(S)H(s)} \tag{2.9}$$

Por lo tanto, la función de transferencia a lazo cerrado que relaciona la tensión en la terminal del generador $V_t(S)$ con la tensión de referencia $V_{ref}(S)$ es:

$$G_{eq}(S) = \frac{G_1(S)G_2(S)G_3(S)}{1 + G_1(S)G_2(S)G_3(S)H(s)} \tag{2.10}$$

Reemplazando las funciones de transferencia, se obtiene:

$$\frac{V_t(S)}{V_{ref}(S)} = \frac{(\frac{K_A}{1+\tau_A S})(\frac{K_E}{1+\tau_E S})(\frac{K_G}{1+\tau_G S})}{1+(\frac{K_A}{1+\tau_A S})(\frac{K_E}{1+\tau_E S})(\frac{K_G}{1+\tau_G S})(\frac{K_R}{1+\tau_R S})} \tag{2.11}$$

Ordenando la expresión anterior, se obtiene:

$$\frac{V_t(S)}{V_{ref}(S)} = \frac{\frac{(K_A)(K_E)(K_G)}{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)}}{1+\frac{(K_A)(K_E)(K_G)(K_R)}{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)(1+\tau_R S)}} \tag{2.12}$$

Factorizando, se obtiene:

$$\frac{V_t(S)}{V_{ref}(S)} = \frac{\frac{(K_A)(K_E)(K_G)}{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)}}{\frac{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)(1+\tau_R S)+(K_A)(K_E)(K_G)(K_R)}{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)(1+\tau_R S)}} \tag{2.13}$$

Simplificando, se obtiene:

$$\frac{V_t(S)}{V_{ref}(S)} = \frac{(K_A)(K_E)(K_G)\cancel{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)}(1+\tau_R S)}{\cancel{(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)}[(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)(1+\tau_R S)+(K_A)(K_E)(K_G)(K_R)]} \tag{2.14}$$

Finalmente, la función de transferencia a lazo cerrado del AVR es:

$$\frac{V_t(S)}{V_{ref}(S)} = \frac{(K_A)(K_E)(K_G)(1+\tau_R S)}{[(1+\tau_A S)(1+\tau_E S)(1+\tau_G S)(1+\tau_R S)+(K_A)(K_E)(K_G)(K_R)]} \tag{2.15}$$

2.3. Análisis de estabilidad de sistemas de control

Para determinar si un sistema es estable se emplean técnicas que operan, ya sea en el dominio del tiempo, sea este continuo o discreto, como en el dominio de la frecuencia. Entre las técnicas de análisis de estabilidad en el dominio del tiempo continuo destacan el Criterio de Estabilidad de Routh - Hurwitz y el Método del Lugar de las Raíces, mientras que para el tiempo discreto resalta el método de Jury. Por su parte, para analizar la estabilidad de sistemas en el dominio de la frecuencia, las técnicas más empleadas son el Criterio de la Estabilidad de Nyquist, los Diagramas de Bode y las trazas de Nichols.

A continuación, se explica el Criterio de Estabilidad de Routh - Hurwitz, el cual también se usa para hallar el rango de valores que pueden tomar los parámetros de los componentes del AVR para conseguir la estabilidad. Por ejemplo, si el sistema AVR de un generador tuviera los siguientes parámetros: K_A de valor desconocido, τ_A = 0.1, K_E = 1, τ_E = 0.4, K_G = 1, τ_G = 1, K_R = 1 y τ_R = 0.05, su función de transferencia a lazo abierto sería:

$$G(S)H(S) = \frac{K_A}{(1+0{,}1S)(1+0{,}4S)(1+S)(1+0{,}05S)} \tag{2.16}$$

Desarrollando se obtiene:

$$G(S)H(S) = \frac{500K_A}{(S+10)(S+2{,}5)(S+1)(S+20)} \tag{2.17}$$

Factorizando se obtiene:

$$G(S)H(S) = \frac{500K_A}{S^4+33{,}5S^3+307{,}5S^2+775S+500} \tag{2.18}$$

La ecuación característica es expresada mediante:

$$1+G(S)H(S) = 0 \tag{2.19}$$

Así:

$$1+\frac{500K_A}{S^4+33{,}5S^3+307{,}5S^2+775S+500} = 0 \tag{2.20}$$

Por lo que el polinomio característico es:

$$S^4+33{,}5S^3+307{,}5S^2+775S+500+500K_A = 0 \tag{2.21}$$

El arreglo de Routh - Hurwitz para este polinomio es:

S^4	1	$307{,}5$	$500+500K_A$
S^3	$33{,}5$	775	0
S^2	$284{,}365$	$500+500K_A$	0
S^1	$-58{,}9K_A+716{,}1$	0	0
S^0	$500+500K_A$		

$$-\frac{1}{33{,}5}\det\begin{pmatrix}1 & 307{,}5\\ 33{,}5 & 775\end{pmatrix}=284{,}365 \tag{2.22}$$

$$-\frac{1}{33{,}5}\det\begin{pmatrix}1 & 500+500K_A\\ 33{,}5 & 0\end{pmatrix}=500+500K_A \tag{2.23}$$

$$-\frac{1}{284{,}365}\det\begin{pmatrix}33{,}5 & 775\\ 284{,}365 & 500+500K_A\end{pmatrix}=-58{,}9K_A+716{,}1 \tag{2.24}$$

$$-\frac{1}{284{,}365}\det\begin{pmatrix}33{,}5 & 0\\ 284{,}365 & 0\end{pmatrix}=0 \tag{2.25}$$

$$-\frac{1}{58{,}9K_A-716{,}1}\det\begin{pmatrix}284{,}365 & 500+500K_A\\ 58{,}9K_A-716{,}1 & 0\end{pmatrix}=500+500K_A \tag{2.26}$$

De la fila S^1 se observa que para conseguir la estabilidad del sistema:

$$-58{,}9K_A+716{,}1>0 \tag{2.27}$$

$$-58{,}9K_A>-716{,}1 \tag{2.28}$$

$$58{,}9K_A<716{,}1 \tag{2.29}$$

$$K_A < \frac{716,1}{58,9} \tag{2.30}$$

$$K_A < 12,16 \tag{2.31}$$

Similarmente, de la fila S^0 se observa que para conseguir la estabilidad del sistema:

$$500 + 500K_A > 0 \tag{2.32}$$

$$500 > -500K_A \tag{2.33}$$

$$1 > -K_A \tag{2.34}$$

$$-1 < K_A \tag{2.35}$$

De esta manera, el intervalo de K_A para hacer estable al sistema es:

$$-1 < K_A < 12,16 \tag{2.36}$$

Al reemplazar K_A=12.16 en S^1 del arreglo de Routh - Hurwitz, todos los componentes de la fila S^1 se hacen cero, por lo que S^2 es el polinomio auxiliar, así:

$$284,365S^2 + 500 + 500K_A = 0 \tag{2.37}$$

$$284,365S^2 + 500 + 500(12,6) = 0 \tag{2.38}$$

$$S = \pm j4,81 \tag{2.39}$$

Así, para K_A = 12.16 se tienen un par de polos conjugados en el eje jω, haciendo que el sistema de control sea marginalmente estable. En la Figura 2.6 se observa el Diagrama de Polos y Ceros correspondiente al sistema AVR analizado con el valor de K_A indicado, en el que se obsevan los cuatro polos correspondientes al cuarto grado del polinomio.

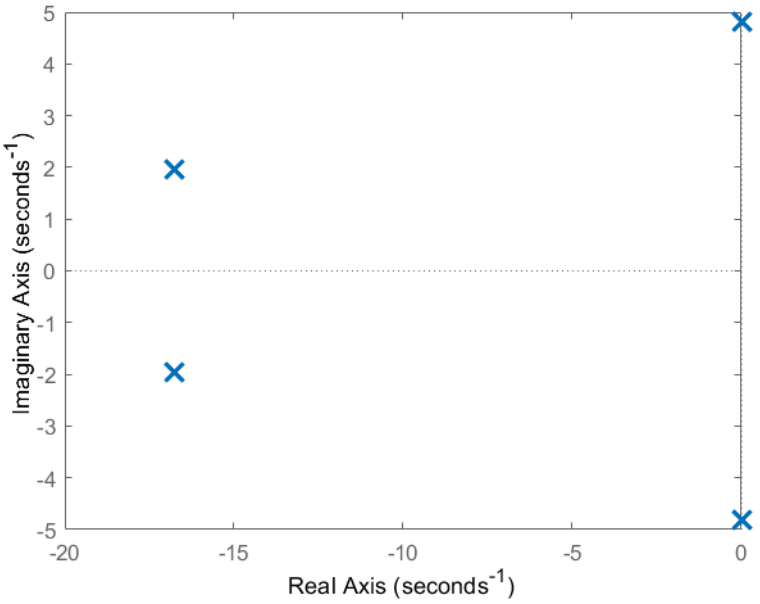


Figura 2.6: Diagrama de polos y ceros del sistema AVR con K_A = 12.16.

2.4. Análisis de respuesta temporal de sistemas de control

Este análisis permite estudiar la respuesta y(t) de los sistemas a una entrada estímulo de prueba r(t). En los sistemas controlados, permite conocer el desempeño de los algoritmos de sintonización de los controladores mediante parámetros de la señal de respuesta. Las señales de prueba que se usan regularmente son funciones escalón, rampa, parábola, impulso, etc.

La señal de respuesta a una señal de prueba puede clasificarse como:

a. Respuesta sobreamortiguada: En caso la señal no presente oscilaciones.

b. Respuesta subamortiguada: En caso la señal presente oscilaciones amortiguadas.

c. Respuesta oscilatoria: En caso el sistema oscile alrededor de un valor de referencia sin lograr estabilizarse.

Los sistemas de primer orden están gobernados por ecuaciones diferenciales de primer orden de la forma:

$$T\frac{dy(t)}{dt} + y(t) = Ku(t) \tag{2.40}$$

Cuya función de transferencia es:

$$G(S) = \frac{Y(S)}{U(S)} = \frac{K}{\tau S + 1} \tag{2.41}$$

Donde K es la ganancia o valor final y τ es la constante de tiempo, es decir es el instante de tiempo en el que la señal de respuesta llega al 0,63 % del valor final. Se considera que cuando la señal de respuesta alcanza el tiempo igual a 4τ, esta alcanza el valor final, así:

$$t_s = 4\tau \tag{2.42}$$

Donde t_s es el tiempo de establecimiento, es decir es el tiempo en el que la señal llega al 98 % del valor final. Estos sistemas tienen un polo en S = -1/T, tal como se observa en la Figura 2.7.

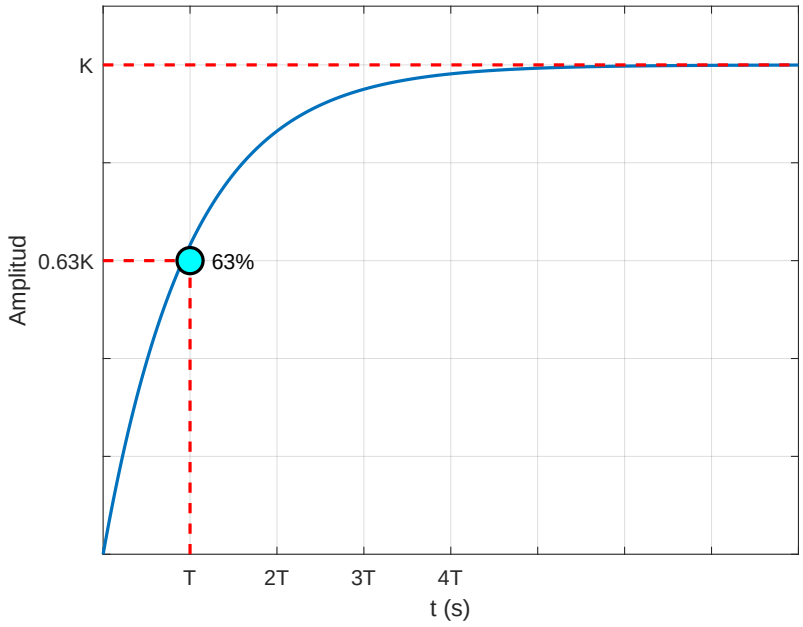


Figura 2.7: Respuesta a entrada escalón unitario en sistemas de primer orden.

Los sistemas de segundo orden tienen dos polos y están gobernados por ecuaciones diferenciales de segundo orden de la forma:

$$\frac{d^2y(t)}{dt^2} + 2\delta\omega_n\frac{dy(t)}{dt} + \omega_n^2 y(t) = K\omega_n^2 u(t) \tag{2.43}$$

Cuya función de transferencia es:

$$G(S) = \frac{Y(S)}{U(S)} = \frac{K\omega_n^2}{S^2 + 2\delta\omega_n S + \omega_n^2} \tag{2.44}$$

Donde los parámetros ω_n y δ son la frecuencia natural no amortiguada y el coeficiente de amortiguamiento respectivamente.

Mientras que en los sistemas de primer orden los polos son reales, en los sistemas de segundo orden los polos pueden ser complejos. Según el tipo de respuesta ante señales de prueba, los sistemas de segundo orden se clasifican en sistemas sub amortiguados, sistemas críticamente amortiguados, sistemas sobreamortiguados y sistemas oscilatorios.

Los sistemas sub amortiguados son aquellos en los que $1 < \delta < 1$ y tienen dos polos complejos ubicados en:

$$S_{1,2} = -\delta\omega_n \pm j\omega_n\sqrt{1-\delta^2} \tag{2.45}$$

Donde:

$$\omega_n\sqrt{1-\delta^2} = \omega_d \tag{2.46}$$

Siendo ω_d: frecuencia natural amortiguada.

Estos sistemas se caracterizan porque su respuesta a una señal de prueba presenta oscilaciones amortiguadas. A menor δ, mayor oscilación.

Los sistemas críticamente amortiguados son aquellos en los que $\delta = 1$ y tienen dos polos reales ubicados en $S_{1,2}$ = -ω_n.

Los sistemas sobreamortiguados son aquellos en los que $\delta > 1$ y tienen dos polos reales ubicados en:

$$S_1 = -\delta\omega_n + \omega_n\sqrt{\delta^2 - 1} \tag{2.47}$$

$$S_2 = -\delta\omega_n - \omega_n\sqrt{\delta^2 - 1} \tag{2.48}$$

Estos sistemas se caracterizan porque su respuesta a una señal de prueba es bien lenta y sin oscilaciones. A mayor δ, más lenta es la respuesta.

Finalmente, los sistemas oscilatorios, llamados también críticamente estables, son aquellos en los que $\delta = 0$ y tienen dos polos imaginarios ubicados en:

$$S_{1,2} = \pm j\omega_n \tag{2.49}$$

Los sistemas subamortiguados tienen un comportamiento similar al mostrado en la Figura 2.8.

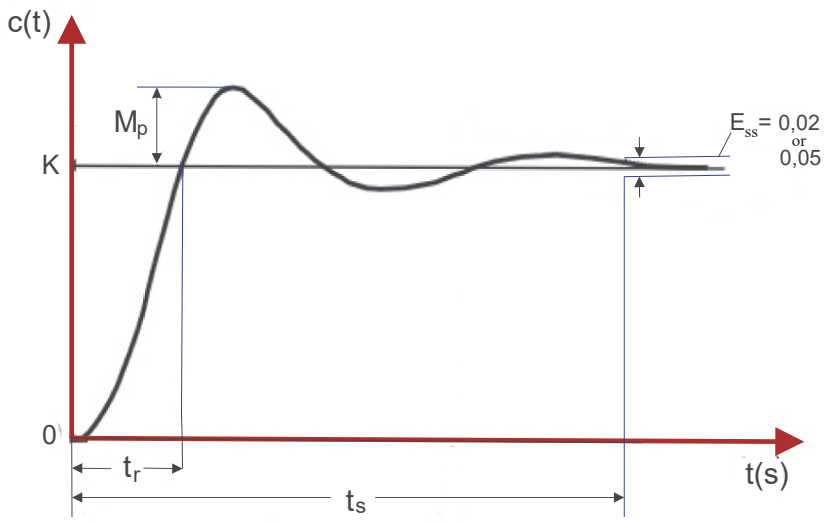


Figura 2.8: Señal de respuesta típica de un sistema subamortiguado a una señal de prueba y sus parámetros de desempeño [4].

La señal de respuesta del sistema a una señal de prueba presenta unos parámetros que definen al sistema y al desempeño de sus controladores asociados, estos son:

Máximo sobrepaso o máximo sobreimpulso (M_p): Es el valor pico máximo de la curva de respuesta medido a partir de la ganancia K.

Error en estado estacionario (E_{ss}): Es el tamaño de la señal medida desde el instante de tiempo en el que la señal ingresa y ya no sale de una banda de error comprendida entre $K \pm$Porcentaje de error admisible, cuyo valor suele ser de 2 % o de 5 %.

Tiempo de crecimiento (t_r): Tiempo transcurrido desde el surgimiento de la señal hasta que esta cruza al valor de la ganancia K.

Tiempo de establecimiento (t_s): Tiempo que transcurre desde el surgimiento de la señal hasta que ingresa y permanece dentro de una banda de error específica.

De acuerdo al Criterio de Estabilidad de Routh - Hurwitz realizado al AVR con los parámetros especificados, se obtuvo que para que sea estable, la ganancia del amplificador (K_A) debe ser menor que 12.16. Si se asigna el valor de 10 a la ganancia del amplificador y si se aplica una señal de prueba de tipo escalón unitario a este AVR, la señal de respuesta obtenida es la mostrada en la Figura 2.9.

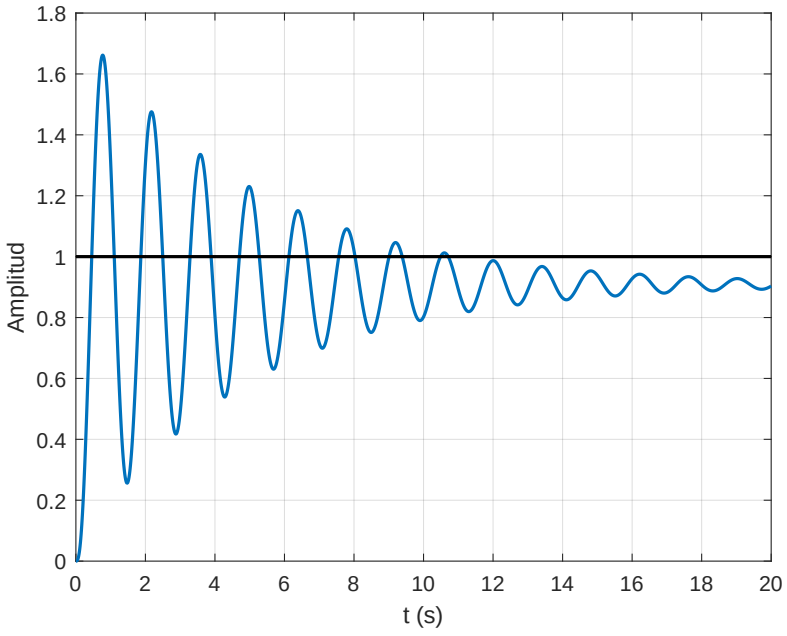


Figura 2.9: Señal de respuesta a un escalón unitario del sistema AVR con $K_A = 10$.

En la Figura 2.9 se observa que la señal es subamortiguada, tiene un valor de máximo sobrepaso muy grande (M_p=1.7) y un largo tiempo de establecimiento (t_s=20s). Se observa también que la señal es altamente oscilante, lo cual puede dañar a los demás componentes del AVR y que se estabiliza en 0.9 y no en el valor de referencia deseado de 1, por lo que el error en estado estacionario es 0.1. Para reducir este error, se debe aumentar la ganancia del amplificador, pero hacerlo causaría que el sistema se torne inestable, por tal motivo se incluye junto al AVR otro dispositivo de control llamado Estabilizador de Sistema de Potencia (PSS).

2.5. Estabilizador de Sistema de Potencia

El Estabilizador de Sistema de Potencia (PSS) permite amortiguar las oscilaciones de baja frecuencia en los sistemas de potencia, ya que amortigua las oscilaciones del rotor del generador síncrono mediante la inclusión de una señal auxiliar de estabilización, produciendo un componente de torque eléctrico en fase con la velocidad del generador. Este dispositivo está conformado por tres bloques:

1. Ganancia del estabilizador (K_{stab}): Determina la cantidad de amortiguamiento asignada por el PSS.

2. Bloque de filtrado de señal: Aplica un filtro pasa alto, donde T_W es la constante de tiempo.

3. Bloque de compensación de fase: Proporciona la característica de avance de fase adecuada para compensar el retraso de fase entre la entrada de excitación y el torque eléctrico del generador, las variables T_1 y T_2 corresponden a las constantes de tiempo.

En la Figura 2.10 se muestra el diagrama de bloques del modelo linealizado de tercer orden de un generador síncrono con un AVR y PSS. El AVR se muestra en color verde, el PSS en color rojo y el generador síncrono en color azul.

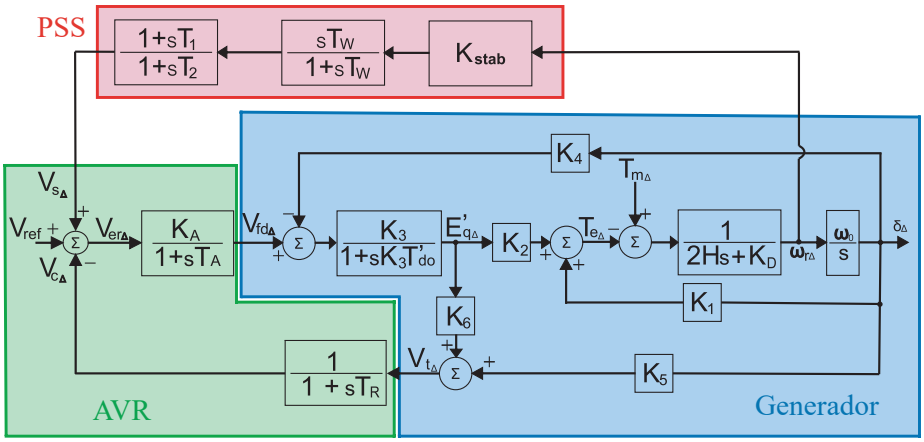


Figura 2.10: Diagrama de bloques de un generador síncrono con un AVR y PSS [5].

Al asignar valores aleatorios a los parámetros del PSS y AVR con K_A=10, se obtiene la respuesta al escalón unitario mostrada en la Figura 2.11.

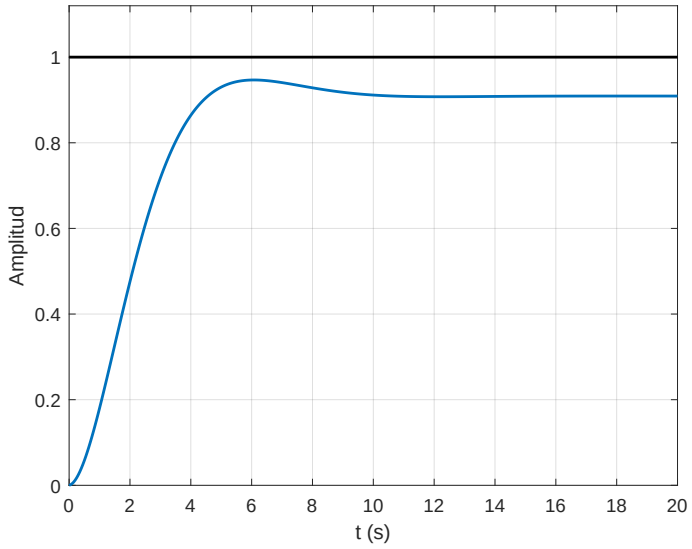


Figura 2.11: Señal de respuesta a un escalón unitario del sistema AVR con K_A = 10 y PSS.

En la señal de respuesta obtenida del voltaje en el terminal del generador, se observa que al incluir un PSS al sistema, se amortiguan en gran medida los sobreimpulsos y se reduce el tiempo de establecimiento t_s aproximadamente a 8s. Sin embargo, la señal de respuesta se sigue estabilizando por debajo de la señal de referencia, por lo que se requiere adicionar algún algoritmo de control para superar este problema.

2.6. Algoritmos de control empleados para la sintonización simultánea de los parámetros del AVR y PSS

Existen muchas maneras de abordar el problema de la sintonización simultánea de estos parámetros, algunos autores proponen incluir junto al AVR, un bloque adicional de un controlador PI o PID y aplicar en él los algoritmos propuestos, otros por su parte proponen aplicar los algoritmos de sintonización directamente en el AVR. Por otro lado, algunos autores eligen ciertos parámetros y les asignan valores iniciales fijos, luego aplican los algoritmos propuestos para sintonizar los parámetros restantes, mientras que otros deciden sintonizar todos los parámetros sin excepción.

Los algoritmos propuestos para este fin son muy variados, pero todos tienen en común corregir de la mejor manera las variaciones de la señal de tensión posterior a perturbaciones con el objetivo de mantener la tensión en el valor deseado, cuantificando su desempeño en base a los valores de los parámetros de la señal de tensión obtenida mediante el uso de técnicas de análisis temporal o frecuencial. Entre los algoritmos más aplicados se encuentran las Redes Neuronales, Lógica Difusa, Algoritmos Genéticos, Optimización por Enjambre de Partículas, etc.

2.6.1. Algoritmos de Control Óptimo basados en técnicas metaheurísticas

El Control Óptimo es una rama de la Teoría de Control enfocada en optimizar una función objetivo asociada a un problema de control. Existen muchas técnicas analíticas de Control Óptimo, tales como las de Pontriagin, Hamilton o Riccati, pero también existen técnicas de Control Óptimo inspiradas en la biología y fenómenos naturales, es el caso de las técnicas de optimización metaheurística, las cuales destacan principalmente debido a su característica de evitar caer en óptimos locales. En la Figura 2.12 se observa la taxonomía de algunos de los algoritmos de optimización metaheurística.

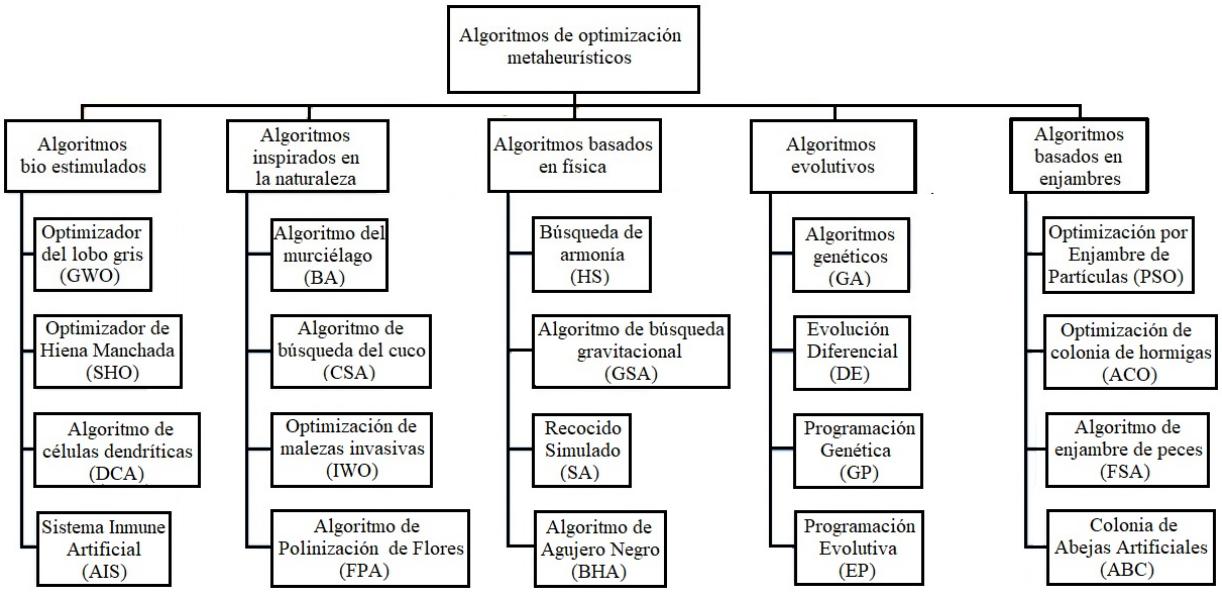


Figura 2.12: Taxonomía de algunos de los algoritmos de optimización metaheurística [6].

2.6.1.1. Algoritmo de Optimización por Enjambre de Partículas (PSO)

El Algoritmo de Optimización por Enjambre de Partículas (PSO) fue desarrollado por Kennedy y Eberhart en el año 1995, este algoritmo está inspirado en el comportamiento de seres vivos, tales como aves, peces, insectos, etc., que se desplazan en grupos con la finalidad de alzanzar algún objetivo fundamental como es el alimento o la seguridad, casi siempre logrando su objetivo en óptimas condiciones. Esto es debido a que todos los individuos (partículas) buscan alzanzar el objetivo común al mismo tiempo, mientras que consiguen comunicarse entre ellos. En cada instante, cada individuo sabe cuán buena es su propia búsqueda y también sabe cuál es el individuo que está obteniendo el mejor resultado. Con esta información, en el siguiente instante, todos los individuos redirigen su exploración hacia el individuo que está obteniendo el mejor resultado de búsqueda, siendo este proceso repetitivo.

De acuerdo con esto, si alguna partícula está realizando la mejor búsqueda y en cada instante mejora hasta encontrar el objetivo, todas las partículas del enjambre la seguirán y convergerán en el objetivo. Matemáticamente, las partículas son posibles soluciones a un problema de optimización que inicialmente son desperdigadas aleatoriamente en diferentes posiciones de una función multivariable (función objetivo) y están delimitadas a un espacio de búsqueda multidimensional.

El problema es hallar los valores de las variables de esta función que correspondan al valor máximo o mínimo (punto óptimo de la superficie n – dimensional de la función). La calidad de la posición de cada partícula i en un instante de tiempo t es evaluada al calcular la función objetivo cada vez que esta toma una nueva posición en el siguiente instante de tiempo (t+1). En este algoritmo, las partículas son inicializadas en el espacio de búsqueda con posiciones aleatorias (x_i) y velocidades aleatorias (v_i) de acuerdo con una distribución de probabilidad uniforme. El mejor valor encontrado por cada partícula es llamado P_{best} y el mejor valor de entre todos los P_{best} es llamado G_{best}.

La formulación básica del algoritmo PSO es acelerar cada partícula hacia las ubicaciones de P_{best} y G_{best}. La velocidad y posición de las partículas son actualizadas usando las ecuaciones (2.50) y (2.51):

$$v_i(t+1) = \omega.v_i(t) + c_1.r_1.[P_{best_i} - x_i(t))] + c_2.r_2.[G_{best} - x_i(t)] \tag{2.50}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2.51}$$

La ecuación 2.50 permite actualizar la velocidad de las partícula para la siguiente iteración $v_i(t+1)$, esta ecuación es una función de la velocidad previa $v_i(t)$, la mejor posición de la partícula P_{best}, la mejor posición global del conjunto de partículas G_{best} y la posición actual de la partícula $x_i(t)$.

La ecuación 2.51 actualiza la posición de la siguiente partícula del enjambre $x_i(t+1)$, la cual se calcula añadiendo la posición previamente ocupada por la partícula $x_i(t)$ más la velocidad de la partícula $v_i(t+1)$, obtenida en la ecuación 2.50.

El parámetro ω es el factor de peso inercial, c_1 y c_2 son las constantes de aceleración cognitiva y social respectivamente y tienen valores que oscilan entre 0 y 2, r_1 y r_2 son números aleatorios entre 0 y 1.

2.6.1.2. Algoritmo de Optimización por Enjambre de Partículas con Decaimiento Lineal (PSO - LD)

Este algoritmo es una variación del PSO clásico, el cual propone una nueva forma de obtener el valor de ω mediante la siguiente ecuación:

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{iter_{max}}.iter \tag{2.52}$$

Donde ω_{max} es el valor inicial del factor de peso inercial, ω_{min} es el valor final del factor de peso inercial, $iter_{max}$ es el número máximo de iteraciones e iter es la iteración actual.

Se observa que con esta variación del algoritmo PSO original, el valor de ω es alto al inicio y que va decreciendo linealmente en cada iteración hasta un valor pequeño durante la optimización. De esta manera, el PSO tendrá una mayor capacidad para realizar una búsqueda global al inicio y tendrá una mayor eficiencia de búsqueda local a medida que se aproxime al final del proceso, tal como se observa en la Figura 2.13.

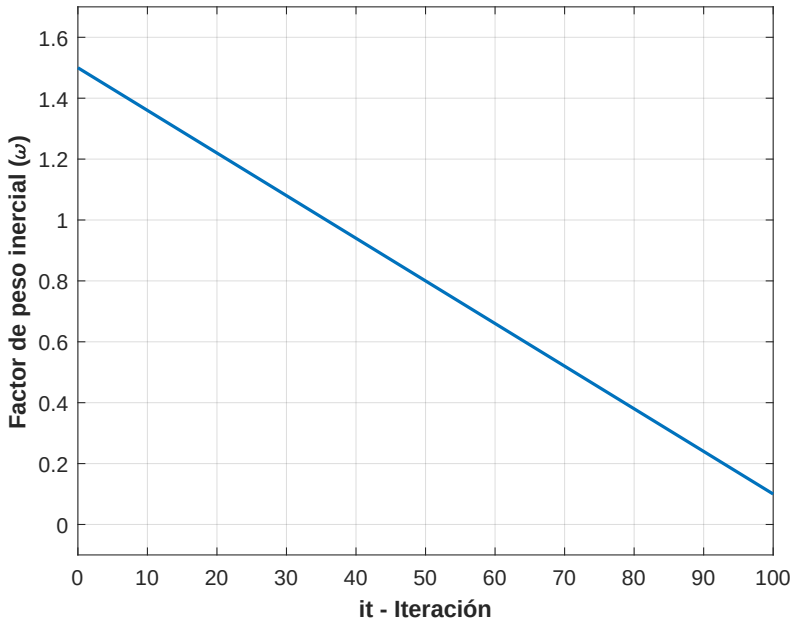


Figura 2.13: Factor de peso inercial del PSO - LD.

2.6.1.3. Algoritmo de Optimización por Enjambre de Partículas con Peso Inercial Oscilatorio (PSO - OIW)

En caso se necesite resolver problemas de mayor complejidad, el algoritmo PSO - LD podría fallar en la búsqueda de la mejor solución, esto debido a las posibles grandes cantidades de soluciones locales, acarreando así convergencias prematuras. Es por este motivo que se desarrolló este nuevo algoritmo en el que se emplea la función coseno. El factor de peso inercial (ω) del algoritmo PSO - OIW se calcula como:

$$\omega = [cos(\frac{2.\pi.iter}{ciclos}).m] + S \tag{2.53}$$

Donde iter es la iteración actual, ciclos es el número de iteraciones necesarias para completar el periodo, m es un multiplicador del cual depende el tamaño de la función ω y S es la función de desplazamiento, la cual permite desplazar a la función ω a lo largo del eje y del plano cartesiano.

Las variables m y S se obtienen mediante:

$$m = \frac{(\omega_{max} - \omega_{min})}{2} \tag{2.54}$$

$$S = (m + \omega_{min}) \tag{2.55}$$

En la Figura 2.14 se muestra el comportamiento del Factor de peso inercial del algoritmo de optimización PSO - OIW.

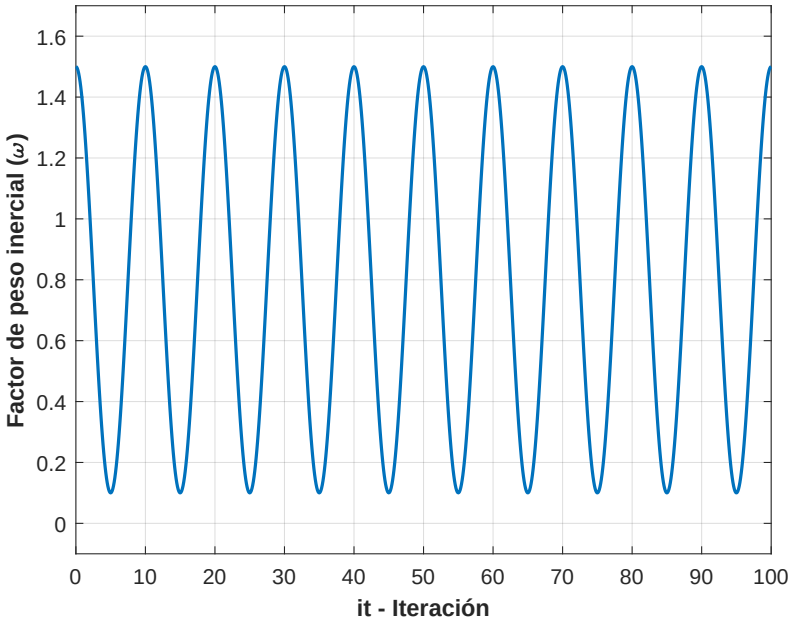


Figura 2.14: Factor de peso inercial del PSO - OIW.

2.6.1.4. Algoritmo de Optimización por Enjambre de Partículas con Decaimiento Exponencial Oscilatorio (PSO - OED)

El desarrollo de este algoritmo surgió como una idea de unir lo mejor de los algoritmos PSO - LD y PSO - OIW, usando una función coseno para asignar el comportamiento oscilatorio e implementar el decaimiento usando una función exponencial. El factor de peso inercial (ω) del algoritmo PSO - OED se calcula como:

$$\omega = [cos(\frac{2.\pi.iter}{ciclos}).m.E_d] + S \tag{2.56}$$

Donde iter es la iteración actual, ciclos es el número de iteraciones necesarias para completar el periodo, m es un multiplicador del cual depende el tamaño de la función ω, E_d es la amplitud de la señal y S es la función de desplazamiento, la cual permite desplazar a la función ω a lo largo del eje y del plano cartesiano.

Las variables m, S y E_d se calculan mediante:

$$m = \frac{(\omega_{max} - \omega_{min})}{2} \tag{2.57}$$

$$S = (m + \omega_{min}) \tag{2.58}$$

$$E_d = e^{\frac{iter}{iter_{max}}} \tag{2.59}$$

En la ecuación 2.56 se muestra la nueva expresión para el cálculo del parámetro ω con una variable adicional llamada E_d, la cual se calcula empleando la ecuación 2.59. Esta variable es la responsable del decaimiento exponencial de la función coseno. El desarrollo de este algoritmo surgió como una alternativa prometedora para hallar la solución óptima.

En la Figura 2.15 se muestra el comportamiento del Factor de peso inercial del algoritmo de optimización PSO - OED.

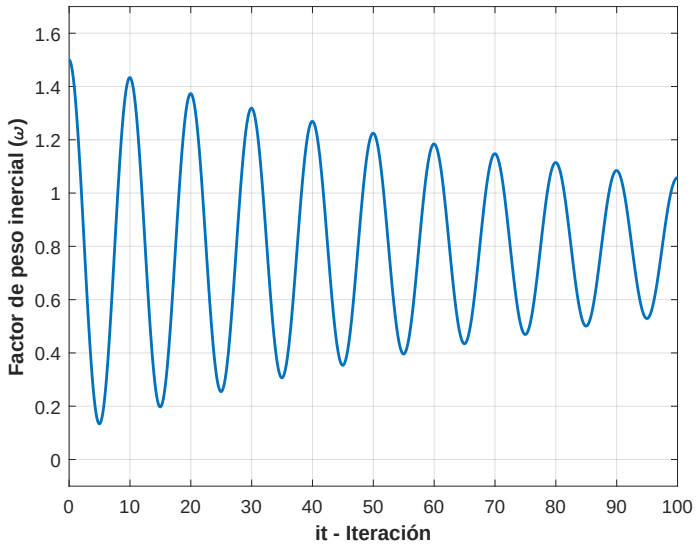


Figura 2.15: Factor de peso inercial del PSO - OED.

2.7. Modelo linealizado de la máquina síncrona

En este capítulo se presenta el modelo matemático para la simulación dinámica de la máquina síncrona en función de sus reactancias y constantes de tiempo, el cual fue desarrollado a partir de las ecuaciones de Park. Con el modelo de sexto orden presentado, se obtiene el modelo linealizado de tercer orden de un sistema eléctrico de potencia conformado por una máquina síncrona conectada a una barra infinita (SMIB) por medio de una impedancia equivalente, tal como se observa en la Figura 2.16.

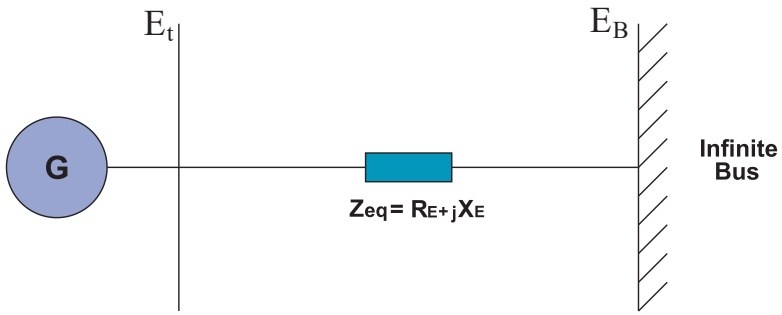


Figura 2.16: Máquina sincrona conectada a una barra infinita por medio de una reactancia.

Por su tipo de polos, las máquinas síncronas se pueden clasificar como máquinas de polos lisos (rotor cilíndrico) o máquinas de polos salientes. Las máquinas que operan en las centrales hidroeléctricas funcionan a baja velocidad y son de polos salientes, mientras que aquellas que operan en centrales termoeléctricas funcionan a alta velocidad y son de polos lisos.

El agua, vapor o gas, según sea el caso, que impacta en los álabes de la turbina conectada al rotor hace que este gire. El rotor a su vez tiene un bobinado por el cual se hace circular una corriente continua, emulando a un electroimán, produciendo así un campo magnético giratorio. El rotor con su campo magnético giratorio se encuentra en el interior de un estator, el cual a su vez tiene su propio bobinado. El campo magnético giratorio induce una corriente eléctrica alterna en el bobinado del estator.

La Transformada de Park define un nuevo grupo de variables estatóricas empleando una referencia girante con el rotor. Estas nuevas variables se obtienen mediante la proyección de las variables actuales del estator en tres ejes: el primer eje se halla alineado con el eje magnético del bobinado del campo (eje directo o eje d), el segundo está alineado con el eje magnéticamente neutro del bobinado de campo, situado a 90° eléctricos atrasado respecto al eje d, el cual es conocido como eje en cuadratura (eje q) y el tercero es un eje estacionario [19].

La Transformada de Park, conocida también como Transformada dq0, permite que las ecuaciones diferenciales que definen las tensiones en los bobinados de la máquina tengan coeficientes constantes, lo cual facilita sobremanera su modelamiento matemático. Esta transformada se describe mediante la ecuación 2.60, siendo P la matriz de transformación de Park.

$$\begin{bmatrix} i_d \\ i_q \\ i_0 \end{bmatrix} = P \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{2.60}$$

La matriz P se define como:

$$P = \frac{2}{3} \begin{bmatrix} cos\theta & cos(\theta - \frac{2\pi}{3}) & cos(\theta + \frac{2\pi}{3}) \\ -sen\theta & -sen(\theta - \frac{2\pi}{3}) & -sen(\theta + \frac{2\pi}{3}) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{2.61}$$

La Transformada Inversa se obtiene mediante P^{-1}:

$$P^{-1} = \begin{bmatrix} cos\theta & -sen\theta & 1 \\ cos(\theta - \frac{2\pi}{3}) & -sen(\theta - \frac{2\pi}{3}) & 1 \\ cos(\theta + \frac{2\pi}{3}) & -sen(\theta + \frac{2\pi}{3}) & 1 \end{bmatrix} \tag{2.62}$$

El modelo de sexto orden puede representarse mediante un modelo más simple, en el que se considera solamente el bobinado de armadura y de campo, este modelo es llamado de tercer orden y considera solo las siguientes ecuaciones:

A. Ecuación de oscilación de la máquina síncrona:

$$\frac{d^2\delta}{dt^2} = \frac{\pi f}{H}(P_m - P_e) \tag{2.63}$$

B. Ecuación para la variación del flujo en el campo principal de la máquina:

$$e_q = \frac{1}{T_{d0}}(E_{fd} - E_I) \tag{2.64}$$

El sistema SMIB se representa mediante las constantes K_1 hasta K_6 que resultan de la linealización de las ecuaciones para un punto de operación según se describe a continuación:

1. Variación del torque eléctrico para un cambio en el ángulo del rotor con flujo concatenado en el eje directo constante:

$$K_1 = \frac{T_e}{\delta} \tag{2.65}$$

2. Variación del torque eléctrico para un cambio del flujo concatenado en el eje directo con ángulo del rotor δ constante:

$$K_2 = \frac{T_e}{e_q} \tag{2.66}$$

3. Factor de impedancia:

$$K_3 = \frac{1}{[1 + K_x(x_d - x_{d0})(x_q + X_E)]} \tag{2.67}$$

4. El efecto desmagnetizador de una variación del ángulo del rotor con tensión de campo constante:

$$K_4 = \frac{1}{K_3}\frac{e_q}{\delta} \tag{2.68}$$

5. Variación de la tensión terminal para un cambio del ángulo del rotor con flujo concatenado en el eje directo constante:

$$K_5 = \frac{E_t}{\delta} \tag{2.69}$$

6. Variación de la tensión terminal con la variación de e_q para el ángulo del rotor constante:

$$K_6 = \frac{E_t}{e_q} \tag{2.70}$$

Con los valores de K obtenidos, es posible obtener los valores correspondientes para los K_i del Diagrama de Bloques del Generador Síncrono con un AVR y PSS de la Figura 2.10.

Capítulo 3

Aplicaciones del método de optimización PSO clásico en los sistemas de potencia

En el presente capítulo se detallan dos aplicaciones del algoritmo PSO clásico para la solución de problemas en los sistemas de potencia. Estas aplicaciones, que derivaron en dos publicaciones indexadas, fueron desarrolladas como parte del proceso de inmersión en el conocimiento de este algoritmo y en la detección de las posibles mejoras que se le podrían plantear, mismas que posteriormente dieron como resultado al método de optimización propuesto en esta tesis (PSO - POED).

3.1. Minimización de pérdidas en sistemas de potencia mediante despacho de potencia reactiva usando Optimización por Enjambre de Partículas

Esta aplicación consiste en el uso del algoritmo PSO clásico para reducir óptimamente las pérdidas de potencia activa en las líneas de transmisión eléctrica, lo cual se consiguió mediante el despacho óptimo de potencia reactiva y el control óptimo de tensiones. En trabajos relacionados [20] aplicaron PSO para minimizar las pérdidas de potencia reales y para mantener la estabilidad de sistemas de potencia y [21] emplearon PSO para minimizar las pérdidas de potencia activas totales en el modelo de prueba IEEE de 6 barras. En contraste, en esta aplicación, se usó el algoritmo PSO en el sistema de prueba IEEE de 14 barras, consiguiendo una reducción de pérdidas de 8.38 %. La pérdida de potencia activa convergió después de 100 iteraciones, desde 13.393MW a 12.27MW.

Las simulaciones se realizaron en Matlab y se empleó MATPOWER [22] para resolver el flujo de carga. La motivación que conllevó al desarrollo de esta aplicación fue la de buscar una solución para reducir las muy altas pérdidas de fuga en las líneas costeras del Perú, las cuales producen grandes pérdidas activas transversales y suponen un alto costo de operación de la transmisión, lo cual finalmente conlleva a un incremento del costo de las tarifas eléctricas, asumido en gran medida por los usuarios [7].

Las pérdidas de potencia activa en las líneas de transmisión dependen de las resistencias, perditancias de fuga y tensiones y, además las tensiones están cercanamente acopladas a las potencias reactivas [23]. Debido a esto, las pérdidas activas pueden ser reducidas al mínimo mediante el control óptimo de: tensiones de generadores, tensiones de condensadores síncronos, posición de taps de transformadores, potencia reactiva de compensadores shunt, etc.

Reducir óptimamente las pérdidas de potencia durante la operación de un sistema de potencia es un problema de optimización matemática donde la función objetivo y restricciones son no lineales y forman una superficie espacial no convexa con muchos óptimos locales. Los métodos clásicos y heurísticos tienen problemas en resolver este problema, ya que tienden a confundir el óptimo local con el óptimo absoluto [24].

PSO permite conseguir soluciones muy cercanas al óptimo absoluto (los valores de las variables de control o variables independientes que producen las mínimas pérdidas activas o variable dependiente), debido a que no se entrampa fácilmente en óptimos locales [25].

El problema planteado es de optimización no lineal entera mixta con restricciones, donde la función objetivo a ser minimizada es la pérdida de potencia activa del sistema, que es la sumatoria de las pérdidas activas de todas las ramas del sistema, tal como se observa a continuación:

$$P_{loss} = \sum_{k=1}^{N} [g_{ij}(V_i^2 + V_j^2 - 2V_iV_jcos\theta_{ij}) + y_{ij}(V_i^2 + V_j^2)] \tag{3.1}$$

El primer sumando de esta función objetivo representa a las pérdidas Joule o pérdidas longitudinales, mientras que el segundo sumando representa a las pérdidas Corona o pérdidas transversales.

donde N es el número de ramas.

g_{ij} es la conductancia de la rama entre la barra i y la barra j.

y_{ij} es la perditancia (conductancia de fuga) de la rama entre la barra i y la barra j.

V_i es la magnitud de voltaje de la barra i.

V_j es la magnitud de voltaje de la barra j.

θ_{ij} es la diferencia de ángulo de fase entre la barra i y la barra j.

Se deben encontrar los valores de los voltajes fasoriales en todas las barras, de tal forma que minimicen la función de pérdidas y que se respeten las siguientes restricciones físicas del problema, representadas por las siguientes restricciones de igualdad, las cuales son las ecuaciones de balance de potencia que resultan de aplicar la Segunda Ley de Kirchhoff:

$$P_{gi} - P_{di} - V_i \sum V_j (G_{ij} cos\theta_{ij} + B_{ij} sin\theta_{ij}) = 0 \tag{3.2}$$

$$Q_{gi} - Q_{di} - V_i \sum V_j (G_{ij} cos\theta_{ij} + B_{ij} sin\theta_{ij}) = 0 \tag{3.3}$$

donde:

P_{gi} es la generación de potencia real en la barra i.

P_{di} es la demanda de potencia real en la barra i.

Q_{gi} es la generación de potencia reactiva en la barra i.

Q_{di} es la demanda de potencia reactiva en la barra i.

A continuación, se muestran las restricciones de desigualdad, las cuales son los rangos de las magnitudes de voltaje, las posiciones de los taps de los transformadores y las inyecciones de potencias reactivas.

$$V_i^{min} < V_i < V_i^{max} \tag{3.4}$$

$$t_j^{min} < t_j < t_j^{max} \tag{3.5}$$

$$Q_{gi}^{min} < Q_{gi} < Q_{gi}^{max} \tag{3.6}$$

En la Figura 3.1 se muestra el diagrama de flujo correspondiente al algoritmo PSO clásico para esta aplicación.

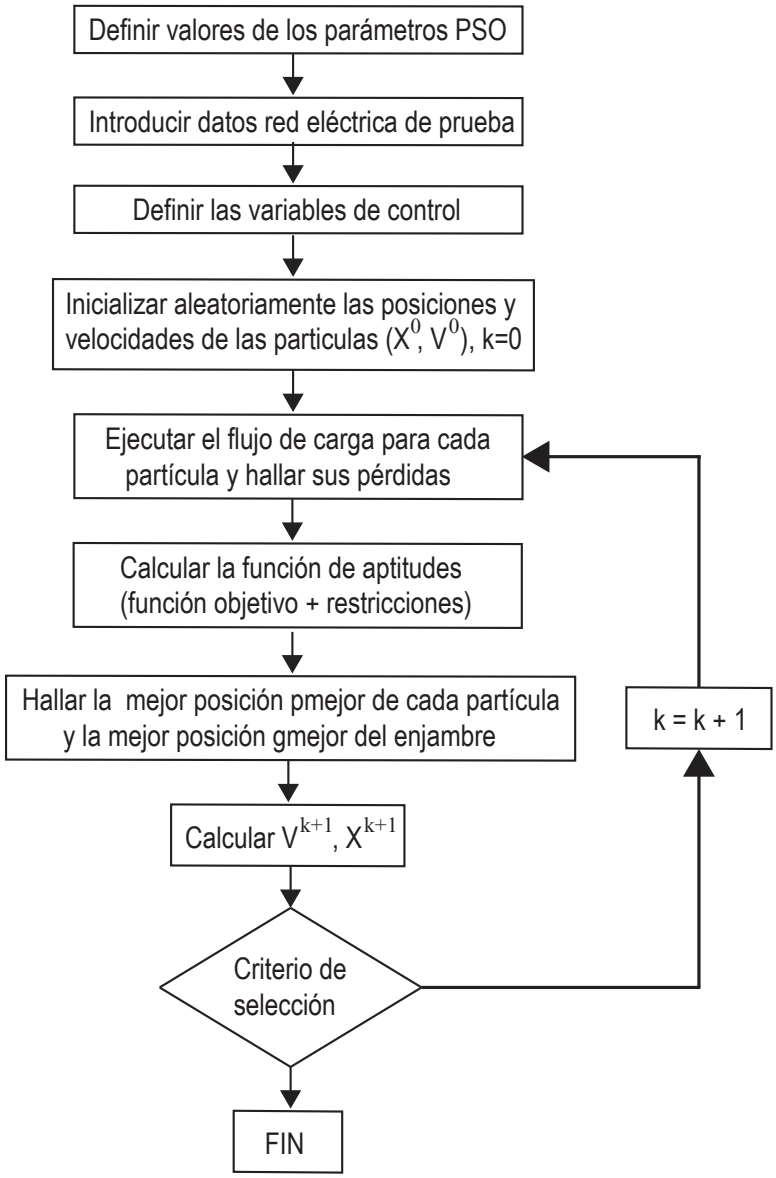


Figura 3.1: Flujograma del PSO aplicado a la minimización de pérdidas de potencia activa [7].

El algoritmo PSO clásico fue aplicado al sistema IEEE de 14 barras, el cual está conformado por: 14 barras, 30 ramas, 2 generadores (uno de los cuales está en la barra *swing* o *slack* y el otro está en la barra 2), 3 condensadores síncronos ubicados en las barras 3, 6 y 8 respectivamente, 3 transformadores con taps y un condensador de potencia reactiva *shunt* en la barra 9. En la Figura 3.2 se muestra su esquema correspondiente.

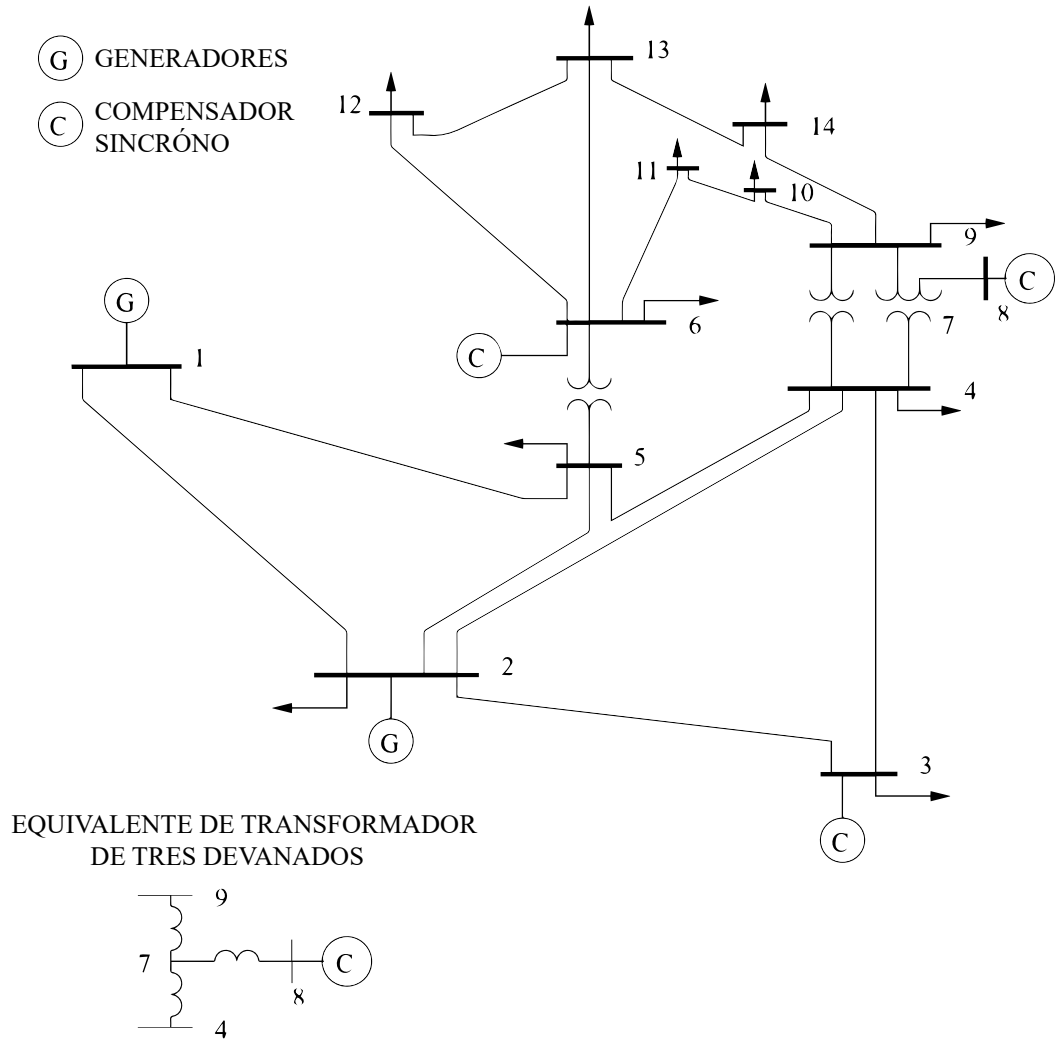


Figura 3.2: Sistema IEEE de 14 barras [7].

De acuerdo a lo descrito, este sistema tiene 9 variables de control: 2 voltajes de los generadores, 3 voltajes de los condensadores síncronos, 3 taps variables de los transformadores y una potencia reactiva del compensador *shunt*, por lo que se determinaron los valores que deben tener las variables de control para que el sistema opere produciendo la mínima cantidad de pérdidas activas.

Los rangos de magnitudes de las variables de control son:

Para voltajes: de 0.95 p.u. a 1.10 p.u.

Para posiciones de taps: de 0.975 a 1.025.

Para inyecciones de potencia reactiva de los compensadores: de 0 MVAR a 20 MVAR.

En la Tabla 3.1 se muestran los valores de las variables de control del sistema para el caso base (sin aplicar el PSO clásico).

Tabla 3.1: Valores de las variables de control en el caso base [7].

V_1	V_2	V_3	V_6	V_8	T_1	T_2	T_3	S_9
1.060	1.045	1.010	1.070	1.090	0.978	0.969	0.932	19

En la Tabla 3.1 los subíndices de las variables señalan las barras en las que se ubican, a excepción de los transformadores. En el caso base, la carga de potencia real total es de 259 MW, la carga de potencia reactiva total es 73.5 MVAR y la pérdida de potencia real es de 13.393 MW.

En la Figura 3.3 se muestra en forma gráfica el resultado del proceso de minimización de pérdidas de transmisión usando el algoritmo PSO. De acuerdo con esto, se observa que la posición de cada partícula se actualiza continuamente hasta alcanzar el número máximo de iteraciones establecido.

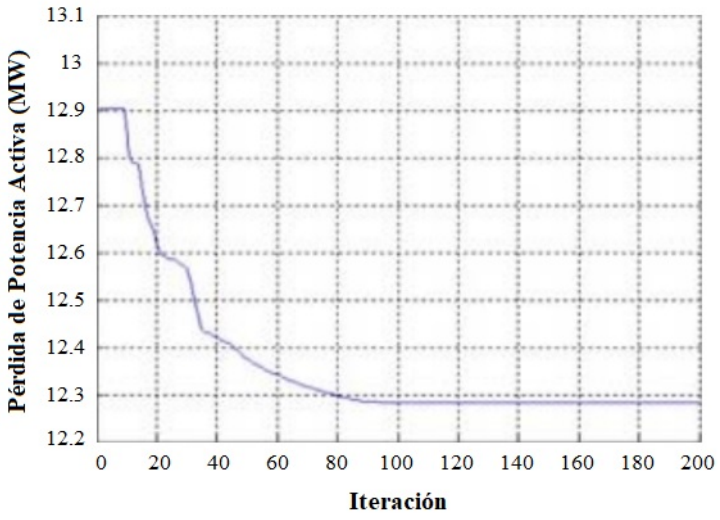


Figura 3.3: Pérdida de Potencia Activa vs. Iteración [7].

Al inicio del proceso se seleccionan al azar las posiciones de las partículas y la pérdida activa global inicial fue 12.91 MW. En cada iteración las partículas actualizan sus posiciones buscando la mejor solución (la producción de menores pérdidas). La pérdida de potencia activa converge después de 100 iteraciones a 12.27 MW, consiguiendo una disminución de pérdidas de 8.38 %.

La aplicación del PSO clásico en este problema de minimización óptima de las pérdidas de potencia activa en las líneas de transmisión, demostró cuantitativamente ser efectiva, ya que permitió reducir estas pérdidas en más de 1MW. Sin embargo, de considerarse aplicar está técnica para resolver problemas más complejos, tales como la reducción de pérdidas en el sistema elétrico nacional, planeamiento de la potencia reactiva o planeamiento de la expansión de la red de transmisión, será necesario probar métodos alternativos, como posibles variantes del algoritmo PSO, en los que factores como el peso inercial ω, varíen dinámicamente con la finalidad de comparar sus rendimientos y elegir aquel algoritmo que presente el mejor desempeño, dadas las múltiples variables de control a considerar.

3.2. Estimación de parámetros de las líneas de transmisión usando Optimización por Enjambre de Partículas

Esta aplicación consiste en el uso del algoritmo PSO clásico para estimar los parámetros físicos de las líneas de transmisión eléctrica [26], lo cual se consiguió mediante la minimización de una función objetivo que representa al error o diferencia entre las variables de operación medidas y las variables de operación calculadas. El algoritmo fue aplicado a 47 líneas de transmisión de alta tensión del sistema eléctrico peruano y, con fines comparativos, paralelamente se aplicó el Método de Optimización de Newton Raphson. Los resultados evidenciaron que las conductancias de fuga de muchas de las líneas costeras son muy altas, con valor mínimo de 0.1 y máximo de 1.96μS/km, mientras que en la sierra peruana estos valores son mínimos. Los resultados obtenidos al aplicar ambos métodos fueron similares en la mayoría de las líneas, sin embargo al usar PSO se obtuvo un menor error de estimación.

La motivación que impulsó el desarrollo de esta aplicación fue la concientización respecto al hecho de que las líneas de transmisión costeras, al estar expuestas a valores muy altos de humedad y contaminación, padecen problemas de corrosión y deterioro temprano, que menoscaba el obtener mediciones fidedignas de sus parámetros y, al proponer una alternativa de estimación de los mismos, se podría mejorar el despacho económico, produciendo importantes ahorros para la economía del país. [8].

El planeamiento, diseño, operación, mantenimiento y comercialización de los sistemas de transmisión eléctrica se sustentan en conocer con buena precisión los valores de sus variables de operación: potencias activas, potencias reactivas y tensiones fasoriales. Estas variables son calculadas y en general son funciones de los parámetros físicos de los sistemas de transmisión, tales como: resistores, reactancias inductivas, admitancias capacitivas y conductancias de fuga. Sin embargo, los verdaderos valores de los parámetros físicos de muchas líneas son desconocidos, ya que estos varían debido a condiciones medioambientales adversas o porque los valores nominales originales no fueron determinados adecuadamente por los fabricantes, entre otras razones. Estos errores pueden ser muy grandes, representando valores del orden de 25 a 30 % [27].

En el Perú y en otros países con niveles de humedad bien altos, es importante conocer los valores correctos de los parámetros físicos, ya que se ha detectado que las conductancias de fuga de las líneas costeras son muy altas, causando pérdidas transversales de más de 60kW/km, mientras que en la mayoría de países del mundo, estas no exceden los 3kW/km. De acuerdo a informes propocionados por el OSINERGMIN, en el segundo semestre del año 2016, la facturación del sector eléctrico peruano alcanzó los 4,407 millones de dólares americanos. Las principales variables involucradas en la operación de las líneas de transmisión son las siguientes:

V_1, V_2: Tensiones en las barras 1 y 2 (kV).
d_1, d_2: Ángulos de fase de las tensiones con respecto a una referencia (radianes).
d = d_1, d_2: Desplazamiento relativo de fase de las tensiones (radianes).
P_1, P_2: Potencias activas en las barras 1 y 2 (MW).
Q_1, Q_2: Potencias reactivas en las barras 1 y 2 (MVAR).

Las líneas de transmisión, debido a las características físicas del conductor empleado y por su propia estructura física, presentan efectos resistivos, inductivos, capacitivos y conductancias de fuga. Estos efectos se representan mediante los siguientes parámetros físicos:

R: Resistencia (ohms)
Y_R: Conductancia de fuga (siemens).
X_L: Reactancia inductiva (ohms).
Y_C: Admitancia capacitiva (mhos).

El valor de R cambia ligeramente con la temperatura, la degradación de la conductividad por envejecimiento, corrosión, contaminación y uso del conductor. Los valores de X_L y Y_C dependen principalmente de la geometría de la estructura de la línea. El valor de Y_R depende de la tensión de operación y del ambiente externo de la línea, por lo que varía todo el tiempo.

En la Figura 3.4 se muestra el Modelo PI de las líneas de transmisión, en el cual se hallan presentes las variables y parámetros analizados.

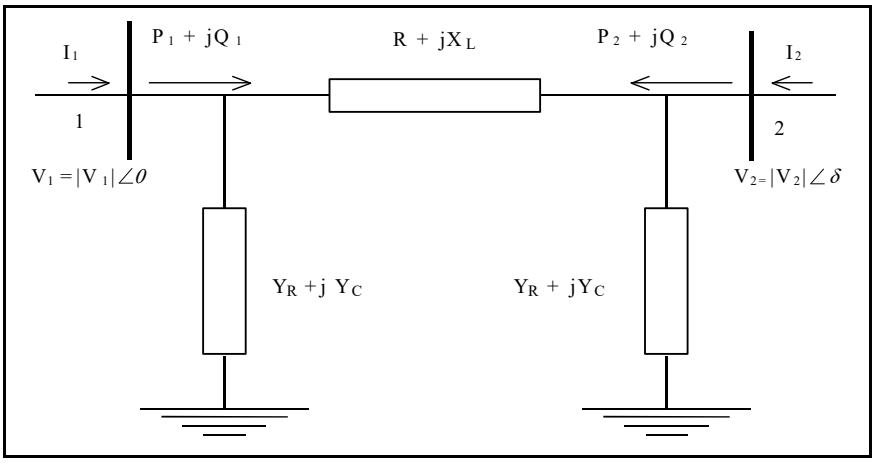


Figura 3.4: Modelo PI de las líneas de transmisión [8].

Las ecuaciones de flujo de carga del modelo PI de una línea de transmisión [28] son las siguientes:

$$P_1 = |V_1|^2 Y_R + \frac{|V_1|^2 R}{R^2 + X_L^2} - \frac{|V_1||V_2| R cos\delta}{R^2 + X_L^2} - \frac{|V_1||V_2| X_L sin\delta}{R^2 + X_L^2} \tag{3.7}$$

$$Q_1 = -|V_1|^2 Y_C + \frac{|V_1||V_2| R sin\delta}{R^2 + X_L^2} + \frac{|V_1|^2 X_L}{R^2 + X_L^2} - \frac{|V_1||V_2| X_L cos\delta}{R^2 + X_L^2} \tag{3.8}$$

$$P_2 = |V_2|^2 Y_R + \frac{|V_2|^2 R}{R^2 + X_L^2} - \frac{|V_1||V_2| R cos\delta}{R^2 + X_L^2} + \frac{|V_1||V_2| X_L sin\delta}{R^2 + X_L^2} \tag{3.9}$$

$$Q_2 = -|V_2|^2 Y_C + \frac{|V_2|^2 X_L}{R^2 + X_L^2} - \frac{|V_1||V_2| R sin\delta}{R^2 + X_L^2} - \frac{|V_1||V_2| X_L cos\delta}{R^2 + X_L^2} \tag{3.10}$$

Para el análisis del proceso de estimación se usan los siguientes símbolos:

k: Intervalos de tiempo en los que se toman las mediciones.

N: Número total de intervalos de las series temporales.

x(k): Vector de estado, x = $[V_1\ V_2\ \delta]^T$

p: Vector de parámetros físicos, p = [R Y_R X_L Y_C]^T

p^0: Vector de parámetros físicos iniciales.

$f(x(k), p, k)$: Vector de variables de operación calculadas (potencias activas, potencias reactivas y tensiones).

z(k): Vector de variables de operación medidas.

v(k): Vector de desviación resultante de la diferencia entre el vector de medición y el vector de operación.

R(k): Matriz de covarianza del vector v(k) o matriz de calidad de los medidores.

w: Vector de desviación de los parámetros físicos respecto al vector de parámetros físicos iniciales.

M: Matriz de covarianza del vector w o matriz de desconfianza de los parámetros iniciales.

En cada intervalo de medición k, la relación entre los valores medidos z(k) de las variables de operación y sus valores calculados f(x(k), p, k) se representa mediante:

$$v(k) = z(k) - f(x(k), p, k) \tag{3.11}$$

Teniendo esto en cuenta, una función natural del error cuadrático de estado y estimación de parámetros para todo el sistema, ponderando la calidad de los medidores es:

$$J(x(k)), k = 1...N; p = (\sum_{k=1}^{N} [z(k) - f(x(k), p]^T R^{-1} [z(k) - f(x(k), p]) + (p^0 - p)^T M^{-1} (p^0 - p) \tag{3.12}$$

El proceso de estimación de estado e identificación de parámetros consiste en encontrar los valores x(k), k = 1...N de los vectores de estado y de los vectores p de los parámetros físicos que minimizen J. El vector p permanece constante para todos los vectores x(k).

El método de optimización de Newton Raphson es eficiente, además de ser probado y usado por los procesos de estimación de estados de la mayoría de centros de control en el mundo [29]. Su empleo para la estimación de parámetros fue propuesto inicialmente por Debs [30] y Schweppe [31]. En la Figura 3.5 se muestra el flujograma respectivo para la implementación de la solución de la estimación de los parámetros.

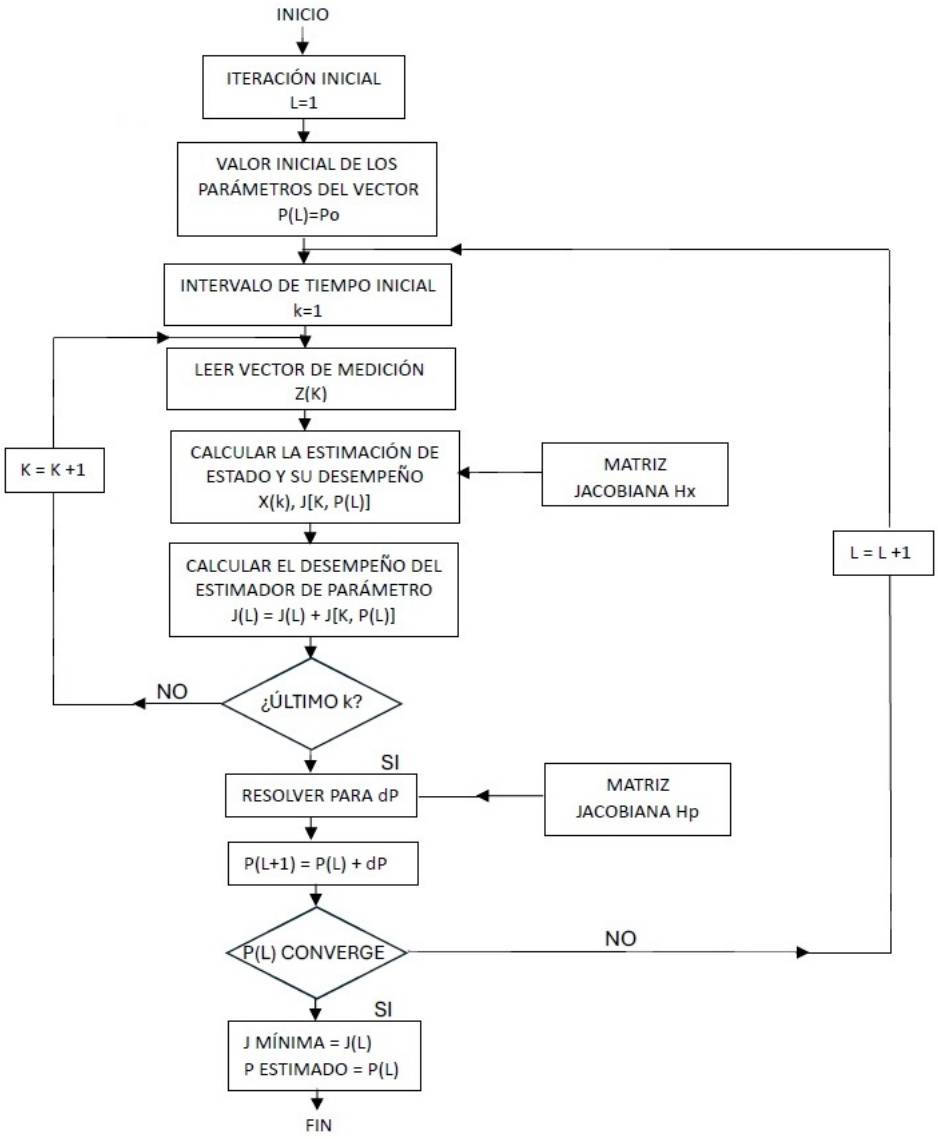


Figura 3.5: Identificación de parámetros usando Newton Raphson [8].

Para un vector de parámetros p(L) de la iteración L, se calcula la estimación de estado x[k, p(L)] de cada vector de medición z(k), obteniendo el error de estimación J[k, P(L)].

La suma de los errores de estimación de los vectores de medición en todos los intervalos es el error de estimación J(L) del parámetro P(L). Si J(L) no converge respecto a su valor anterior, se calcula la corrección óptima dP utilizando Newton - Raphson del parámetro P(L) y se realiza la corrección P(L+1) = P(L)+dP para la siguiente iteración.

La estimación de parámetros finaliza cuando J(L) converge, obteniendo el error mínimo de estimación de parámetros J_{error} y el vector de los valores de los parámetros estimados $P_{estimado}$.

$$dp = \{\sum_{k=1}^{N}[H_p^T(k)R^{-1}H_p(k)] + M^{-1}\}^{-1}\{\sum_{k=1}^{N}[H_p^T(k)R^{-1}[z(k) - y(x(k), p_L)] + M^{-1}(p^0 - p_L)\} \tag{3.13}$$

En el año 1997 el Instituto Nacional de Estándares de Medición (INMS) de Canadá realizó mediciones experimentales en algunas de las líneas de transmisión del Perú para determinar sus pérdidas transversales. Posteriormente, en el año 2002 se realizó un mantenimiento intensivo que consistió en el cepillado de las líneas y en consecuencia, las pérdidas transversales decrecieron considerablemente. Los resultados de la estimación óptima promedio de las pérdidas transversales obtenidos de las simulaciones desde los años 2002 al 2015 son consistentes con las mediciones experimentales, lo que confirma la validez del método propuesto en esta aplicación.

La aplicación del PSO clásico en este problema de estimación de parámetros de las líneas de transmisión, demostró ser efectiva, ya que permitió estimar parámetros que por aspectos medioambientales y propios de su desgaste, físicamente muestran valores alterados. Además, permitió identificar los valores de la conductancia de fuga Y_R en las líneas de transmisión, siendo este parámetro uno de los más complicados de medir. Sin embargo, de considerarse aplicar está técnica para resolver problemas más complejos, será necesario probar métodos alternativos, como posibles variantes del algoritmo PSO, en los que factores como el peso inercial ω varíen dinámicamente con la finalidad de comparar sus rendimientos y elegir aquel algoritmo que presente el mejor desempeño, dadas las múltiples variables de control a considerar.

Capítulo 4

Algoritmo de optimización propuesto

En este trabajo de tesis se propone un nuevo algoritmo de optimización metaheurístico llamado Optimización por Enjambre de Partículas con Decaimiento Exponencial Oscilatorio Periódico (PSO - POED). Este algoritmo es una variante del algoritmo PSO clásico, por lo que las ecuaciones de actualización de la velocidad y posición de las i partículas o posibles soluciones al problema de optimización, conservan su naturaleza estocástica (no determinística).

En este algoritmo, el factor de peso inercial ω opera con una función cosenoidal decreciente, que luego crece y vuelve a decrecer periódicamente, conforme aumenta el número de iteraciones.

El factor de peso inercial ω del algoritmo PSO - POED se calcula mediante:

$$\omega = [cos(\frac{2.\pi.iter}{ciclos}).M.E_d] + S \tag{4.1}$$

Donde iter es la iteración actual, ciclos es el número de iteraciones necesarias para completar el periodo, M es el periodo de la señal, E_d es la amplitud de la señal y S es la función de desplazamiento, la cual permite desplazar a la función ω a lo largo del eje y del plano cartesiano.

Las variables E_d y S se calculan mediante:

$$E_d = e^{\frac{M.E_t - iter}{\theta}} \tag{4.2}$$

$$S = (m + \omega_{min}) \tag{4.3}$$

$$E_t = int(iter/M) \tag{4.4}$$

Donde: int(iter/M) es la parte entera de iter/M, M representa el perido de la oscilación, mientras que θ es el factor que multiplica a la amplitud de dicha oscilación, obteniéndose m mediante:

$$m = \frac{(\omega_{max} - \omega_{min})}{2} \tag{4.5}$$

En la Figura 4.1 se muestra el comportamiento del Factor de peso inercial del algoritmo de optimización propuesto PSO - POED.

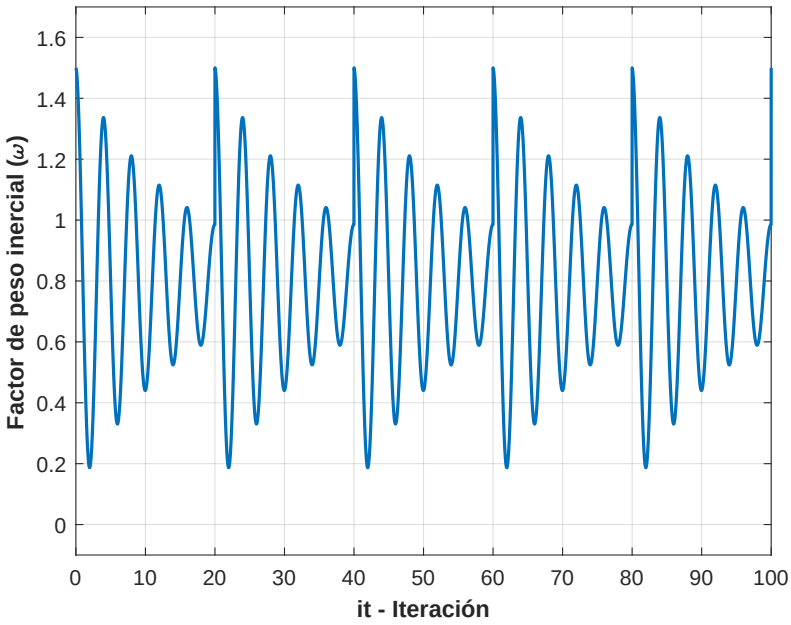


Figura 4.1: Factor de peso inercial del PSO - POED.

En la Figura 4.2 se muestra el diagrama de flujo correspondiente al algoritmo de optimización propuesto PSO - POED.

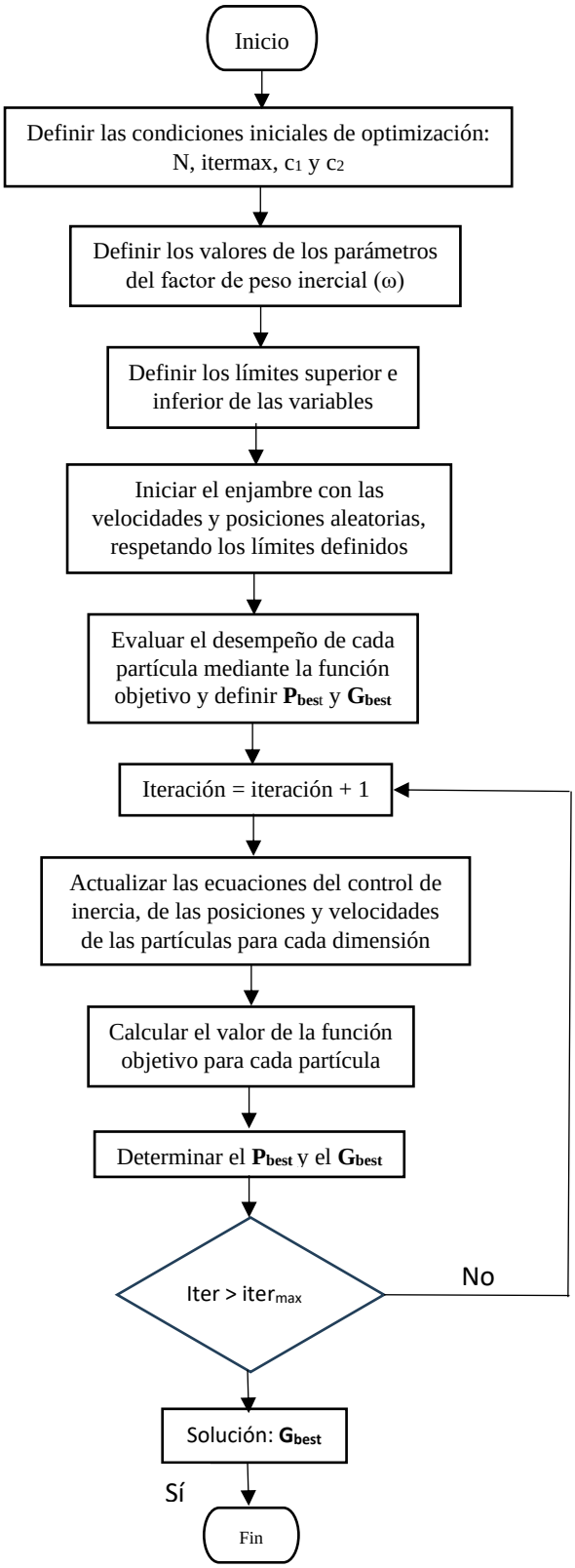


Figura 4.2: Flujograma del algoritmo de optimización propuesto PSO - POED.

Capítulo 5

Estudio de casos y Análisis de resultados

En el presente capítulo se aplica el algoritmo de optimización propuesto (PSO - POED) para sintonizar óptimamente los parámetros K_A del AVR y K_{stab}, T_1 y T_2 del PSS de los generadores síncronos de los sistemas de prueba SMIB y sistema IEEE de 9 barras, manteniendo constantes los valores de los demás parámetros, así: $T_A = 0{,}02s$, $T_R = 0{,}02s$ y $T_W = 1s$. Los resultados son comparados con los obtenidos al aplicar los algoritmos de optimización alternativos PSO - LD, PSO - OIW y PSO - OED. En todos los algoritmos se emplearon los siguientes parámetros de simulación: Número máximo de iteraciones = 100, w_{max}=1.5, w_{min}=0.1, c_1=0.5, c_2=1.5 y β=0.005. Respecto al número de partículas, para el sistema SMIB se emplearon 50 partículas, mientras que para el sistema IEEE de 9 barras se emplearon 100 partículas.

La función a objetivo (FO_i) a maximizar para cada generador G_i es la siguiente:

$$FO_i = \frac{1}{(1-e^{-}\beta)(M_p + E_{ss}) + e^{-}\beta(t_s - t_r)} \tag{5.1}$$

Por lo que la función objetivo (FO) para el número de generadores empleados (NG) es:

$$FO = \sum_{i=1}^{NG} FO_i \tag{5.2}$$

La función objetivo (FO) se hace máxima cuando los parámetros M_p, e_{ss}, t_s y t_r que están en el denominador se hacen mínimos, los cual es lo que se busca obtener en el Análisis de Respuesta Temporal.

5.1. Caso de estudio del Sistema Máquina Barra Infinita

Este sistema está conformado por un generador termoeléctrico G1, tal como se observa en la Figura 5.1.

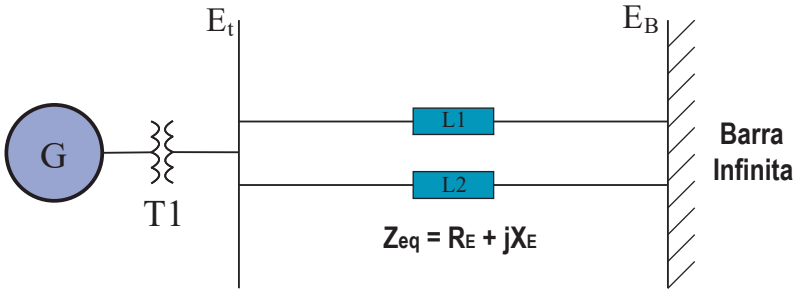


Figura 5.1: Sistema Máquina Barra Infinita.

En este sistema se produce una perturbación trifásica de corto circuito en la línea L1 al 50 %. Esta perturbación que inicia en el instante de tiempo de 1s y se despeja después de 100ms, causa la caída de tensión producida por el generador G1, la cual se intenta restablecer mediante el uso de los controladores AVR y PSS, cuyos parámetros indicados son sintonizados óptimamente. En la Tabla 5.1 se muestran los límites superior e inferior de los parámetros a ser determinados.

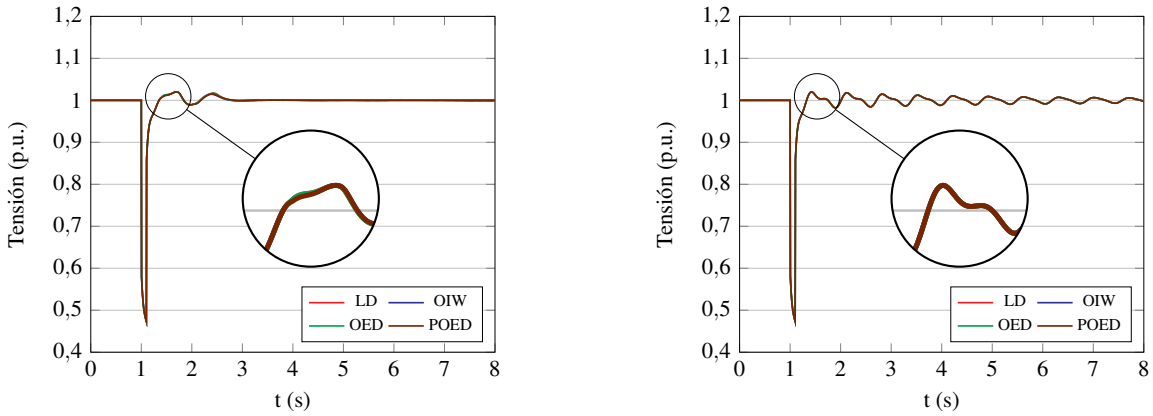
Tabla 5.1: Límites de los parámetros del AVR y PSS del Sistema Máquina Barra Infinita.

Parámetros	$K_A(p.u.)$	$K_{stab}(p.u.)$	$T_1(s)$	$T_2(s)$
Límite superior	400	2	3	0.5
Límite inferior	1	0.1	1	0.0001

Para el Sistema Máquina Barra Infinita, al usar el algoritmo propuesto PSO - POED, los mejores valores hallados de los parámetros de ω son:

Ciclos = 2, θ = 20, M = 50.

En la Figura 5.2 se observa el Análisis de Respuesta Temporal de la señal de tensión del generador G1 posterior a la perturbación de corto circuito para la mejor y peor simulación respectivamente. La señal de tensión fue corregida por acción del algoritmo propuesto PSO - POED con el que se sintonizaron los parámetros de los controladores, siendo el Set Point la tensión en los terminales del generador G1.



(a) Mejor simulación

(b) Peor simulación

Figura 5.2: Comparación de los algoritmos aplicados en el SMIB.

Los valores de la mejor y peor simulación se muestran en las Tablas 5.2 y 5.3 respectivamente, en las que se puede observar que el algoritmo PSO - POED propuesto obtiene un mejor promedio de la función objetivo y menor desviación estándar que los algoritmos alternativos.

Tabla 5.2: Parámetros del AVR y PSS obtenidos al aplicar los diferentes algoritmos.

		Solución				**Función Objetivo**			
Algoritmo		K_A(p.u)	K_{stab}(p.u)	T_1(s)	T_2(s)	Desviación Estándar	Promedio	Mejor	Peor
PSO-LD	Mejor	124.06	0.1	1	0.5	2.367	5.259	9.934	2.791
	Peor	100.47	2	3	0.0001				
PSO-OIW	Mejor	124.06	0.1	1	0.5	2.118	5.031	9.934	2.791
	Peor	104.68	1.92	1	0.0001				
PSO-OED	Mejor	124.06	0.1	1	0.5	2.220	5.220	9.934	2.791
	Peor	100.19	1.62	3	0.0001				
PSO-POED	Mejor	124.06	0.1	1	0.5	2.108	5.305	9.934	2.791
propuesto	Peor	109.36	2	1.54	0.0102				

Tabla 5.3: Parámetros de respuesta temporal obtenidos al aplicar los diferentes algoritmos.

	Mejor solución				Peor solución			
Algoritmo	t_r(s)	t_s(s)	$E_{ss}(\%)$	$M_p(\%)$	t_r(s)	t_s(s)	$E_{ss}(\%)$	$M_p(\%)$
PSO-LD	0.332	0.422	0.229	2.0	0.352	0.702	0.0000	2.0
PSO-OIW	0.332	0.422	0.229	2.0	0.352	0.702	0.0002	2.0
PSO-OED	0.332	0.422	0.229	2.0	0.352	0.702	0.0002	2.0001
PSO-POED	0.332	0.422	0.229	2.0	0.352	0.702	0.0004	2.0

5.2. Caso de estudio del Sistema IEEE de 9 barras

Este sistema está conformado por tres generadores hidráulicos síncronos: G1 (slack), G2 y G3, tal como se observa en la Figura 5.3.

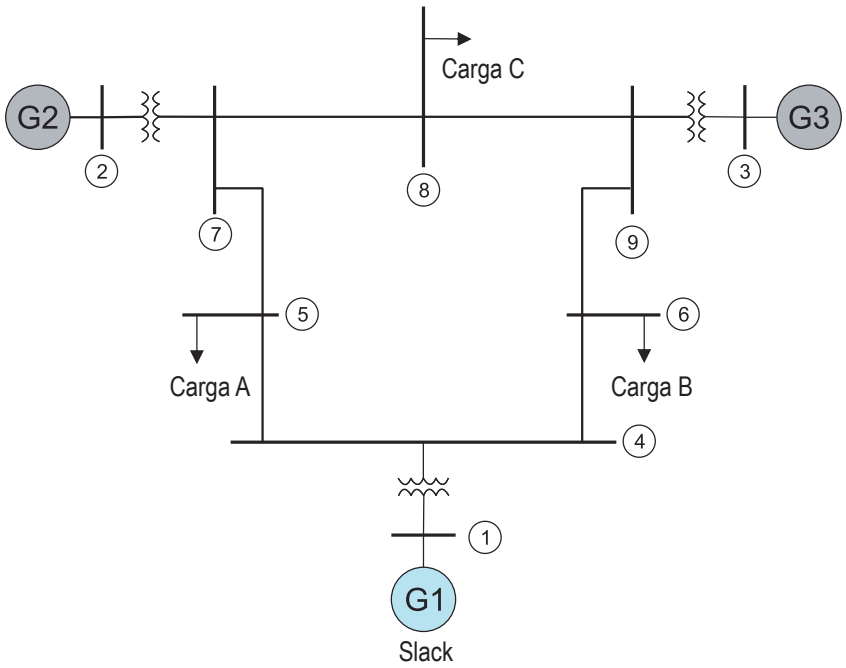


Figura 5.3: Sistema IEEE de 9 barras [9].

Los generadores G2 y G3 producen una tensión en sus terminales de 1.025 p.u. y 1.026 p.u. respectivamente. En el sistema IEEE de 9 barras ocurre una perturbación en el segundo 1 de operación, la cual es despejada luego de 100ms. Esta perturbación consiste en un corto circuito cerca a la línea 5 - 7, el cual produce un aumento de la corriente y una caída de la tensión de los generadores.

Por acción de los controladores es llevada de vuelta en 1ms a los valores de tensión producidos previos a la perturbación. Sin embargo, en esta recuperación se presentan oscilaciones, las cuales requieren ser corregidas de la mejor manera para evitar daños en el sistema de potencia, lo cual se consigue mediante una adecuada sintonización de los parámetros del AVR y PSS.

Para el sistema IEEE de 9 barras, al usar el algoritmo propuesto PSO - POED, los mejores valores hallados de los parámetros de ω son:

Ciclos = 4, θ = 10, M = 20.

En la Figura 5.4 se observa el Análisis de Respuesta Temporal de la señal de tensión del generador G2 posterior a la perturbación de corto circuito, la cual fue corregida por acción del algoritmo propuesto PSO - POED con el que se sintonizaron los parámetros de los controladores, siendo el Set Point la tensión en los terminales del generador G2.

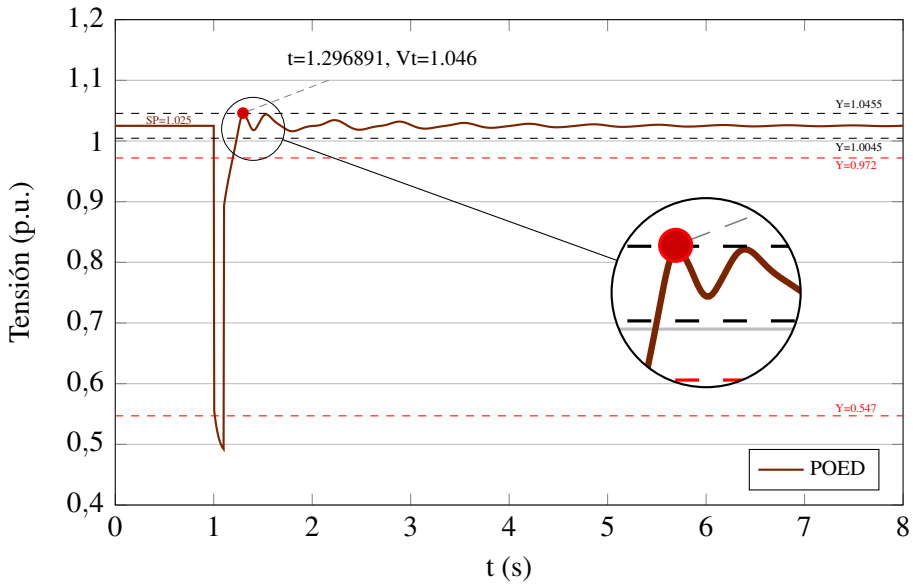


Figura 5.4: POED para el generador G2 del sistema IEEE de 9 barras - Mejor resultado.

En la Figura 5.4 se observa que el pico máximo de la señal de tensión corregida por el algoritmo propuesto PSO - POED es de 1.046 p.u., por lo tanto el máximo sobrepaso (M_p) es:

$$M_p = 1{,}046 p.u. - 1{,}025 p.u. = 0{,}0210 p.u. \tag{5.3}$$

Similarmente:

$$M_p(\%) = (\frac{1{,}046 - 1{,}025}{1{,}025}) * 100\,\% = 2{,}05\,\% \tag{5.4}$$

Además, se observa que la señal se mantiene en la banda de error desde el segundo pico, cuyo valor es 1.0362. El error de estado estacionario (E_{ss}) es la diferencia entre este valor y el Set Point (1.025), dando el valor de $0.0112\,\%$.

En la Figura 5.4 la línea roja discontinua inferior señala el $10\,\%$ del valor final, mientras que la línea roja discontinua superior señala el $90\,\%$ del valor final. Una vez identificados estos valores se deduce que el tiempo de subida (t_r) de la señal es de 0.2617s, ya que este valor es la diferencia entre el instante en que la señal cruza el Set Point y el tiempo en que aparece la señal.

La tolerancia admisible para considerar que el sistema se encuentra en régimen permanente se asigna al $2\,\%$, por lo tanto la banda que indica que la señal de respuesta permanece en este régimen está definida por los limites superior (l_s) e inferior (l_i), tal como se observa en la Figura 5.4, los cuales se obtienen según:

$$l_s = 1{,}025 + (0{,}02)(1{,}025) = 1{,}0455 \tag{5.5}$$

$$l_i = 1{,}025 - (0{,}02)(1{,}025) = 1{,}0045 \tag{5.6}$$

Conociendo estos valores, el tiempo de establecimiento (t_s) de la señal de respuesta es de 0.3017s, ya que este valor es la diferencia entre el tiempo en que la señal entra y permanece dentro de la banda de error, lo cual ocurre en el segundo pico, y el tiempo de surgimiento de la señal.

Similarmente, se aplicaron los algoritmos alternativos variantes del PSO clásico con la finalidad de evaluar su desempeño y comparar los resultados obtenidos con el algoritmo propuesto. En la Figura 5.5 se observan las señales de respuesta a la perturbación aplicando los algoritmos alternativos y el algoritmo propuesto PSO - POED.

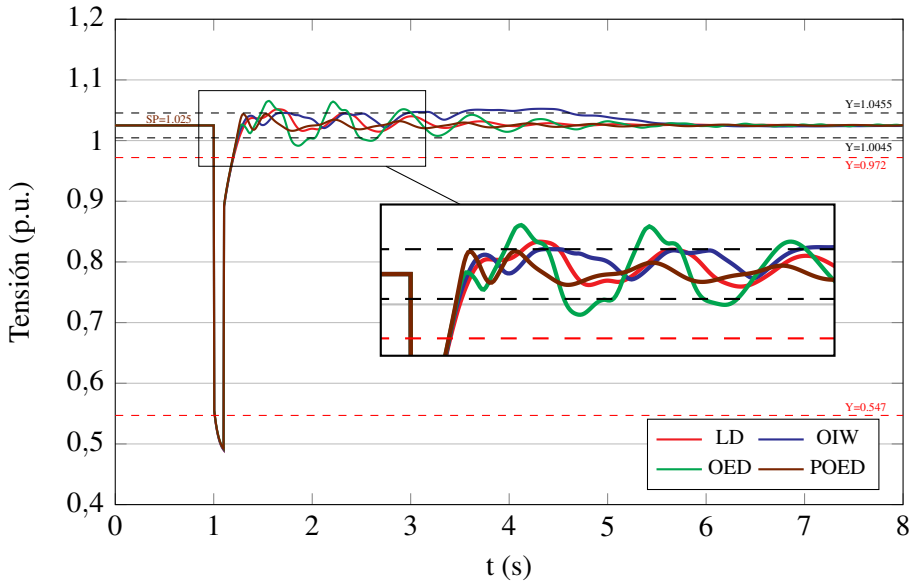


Figura 5.5: Comparación de resultados de las mejores simulaciones obtenidas al aplicar los algoritmos en el Generador 2 del Sistema de 9 barras.

En la Figura 5.5 se observa que la señal del algoritmo propuesto PSO - POED es la más rápida, ya que llega primero a la referencia, por lo que su tiempo de crecimiento (t_r) es el menor de entre los cuatro algoritmos. Así mismo, se observa que la señal del algoritmo propuesto PSO - POED presenta el menor valor de máximo sobrepaso (M_p), el menor tiempo de establecimiento (t_s) y el menor error en estado estacionario (E_{ss}).

De forma similar, se procede a efectuar este análisis para el generador G3 del Sistema IEEE de 9 barras. En la Figura 5.6 se observa el Análisis de Respuesta Temporal de la señal de tensión del generador G3 posterior a la perturbación de corto circuito, la cual fue corregida por acción del algoritmo propuesto PSO - POED con el que se sintonizaron los parámetros de sus controladores.

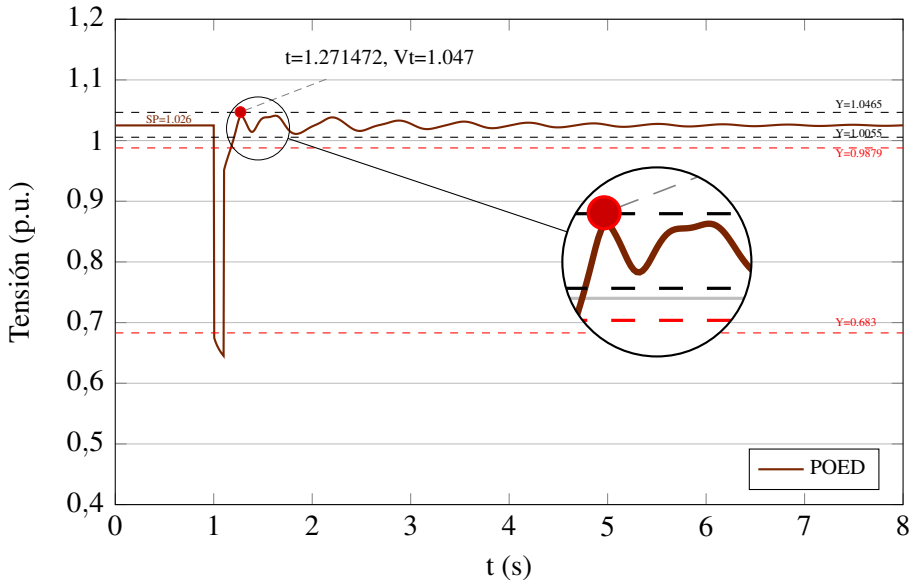


Figura 5.6: POED para el generador G3 del sistema IEEE de 9 barras - Mejor resultado.

En la Figura 5.6 se observa que el pico máximo de la señal de tensión corregida por el algoritmo propuesto PSO - POED es de 1.047 p.u., por lo tanto el máximo sobrepaso (M_p) es:

$$M_p = 1{,}047 p.u. - 1{,}026 p.u. = 0{,}0210 p.u. \tag{5.7}$$

Similarmente:

$$M_p(\%) = (\frac{1{,}047 - 1{,}026}{1{,}026}) * 100\,\% = 2{,}05\,\% \tag{5.8}$$

Adicionalmente, se observa que el error en estado estacionario (E_{ss}) es de: 0.0077 %.

Similarmente, en la Figura 5.6 la línea roja discontinua inferior señala el 10 % del valor final, mientras que la línea roja discontinua superior señala el 90 % del valor final. Identificados estos valores, se desprende que el tiempo de crecimiento (t_r) de la señal es de 0.242s.

Considerando 2 % como valor de tolerancia admisible para que el sistema se encuentra en régimen permanente, los limites superior (l_s) e inferior (l_i), se calculan según:

$$l_s = 1{,}026 + (0{,}02)(1{,}026) = 1{,}0465 \tag{5.9}$$

$$l_i = 1{,}026 - (0{,}02)(1{,}026) = 1{,}0055 \tag{5.10}$$

Conocidos estos valores, el tiempo de establecimiento (t_s) de la señal de respuesta es 0.2817s.

Similarmente, se aplicaron los algoritmos alternativos variantes del PSO clásico con la finalidad de evaluar su desempeño y comparar los resultados obtenidos con el algoritmo propuesto. En la Figura 5.7 se observan las señales de respuesta a la perturbación aplicando los algoritmos alternativos y el algoritmo propuesto PSO - POED.

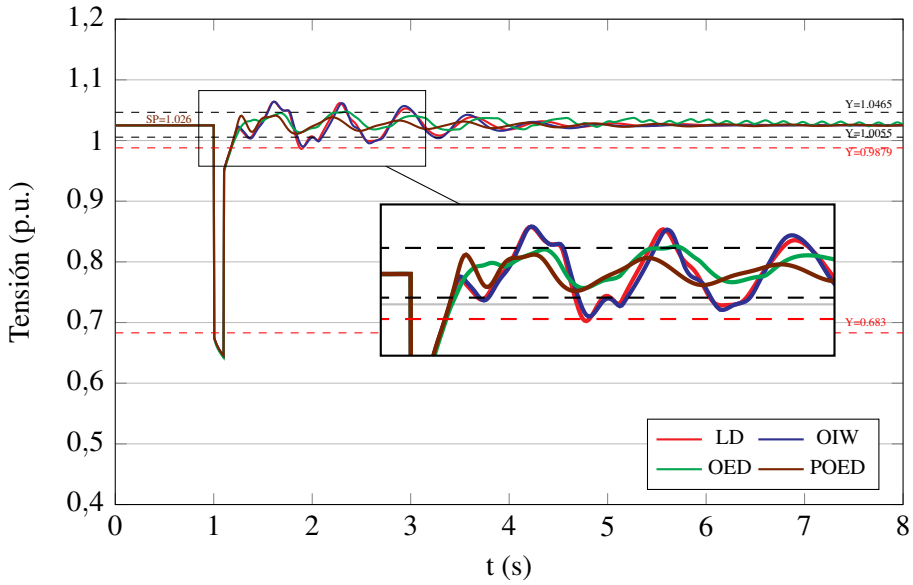


Figura 5.7: Comparación de resultados de la mejores simulaciones obtenidas al aplicar los algoritmos en el Generador 3 del Sistema de 9 barras.

En la Figura 5.7 se observa que la señal del algoritmo propuesto PSO - POED es la más rápida, ya que llega primero a la referencia, por lo que su tiempo de crecimiento (t_r) es el menor de entre los cuatro algoritmos. Así mismo, se observa que la señal del algoritmo propuesto PSO - POED presenta el menor valor de máximo sobrepaso (M_p), el menor tiempo de establecimiento (t_s) y el menor error en estado estacionario (E_{ss}).

Los parámetros de los controladores AVR y PSS se sintonizan en valores comprendidos entre márgenes establecidos. En la Tabla 5.4 se muestran estos límites para ambos controladores.

Tabla 5.4: Límites de los parámetros del AVR y PSS del Sistema IEEE de 9 barras.

Parámetros	$K_A(p.u.)$	$K_{stab}(p.u.)$	$T_1(s)$	$T_2(s)$
Límite superior	400	50	1	0.05
Límite inferior	1	0.1	0.6	0.005

En la Tabla 5.4 se observa que los límites establecidos son diferentes a los indicados en la Tabla 5.3, esto se debe a que el Sistema SMIB tiene un generador termoeléctrico, mientras que en el Sistema IEEE de 9 barras los generadores son hidráulicos.

En la Tabla 5.5 se muestran los parámetros del AVR y PSS obtenidos al aplicar el algoritmo propuesto y los algoritmos alternativos, los cuales también fueron aplicados en esta tesis para efectos de comparación. Así mismo, se muestran los resultados de convergencia para las 100 simulaciones realizadas empleando los diferentes algoritmos.

Tabla 5.5: Parámetros del AVR y PSS obtenidos al aplicar los diferentes algoritmos.

		Mejor solución				**Función Objetivo**			
Algoritmo	Generador	K_A(p.u)	K_{stab}(p.u)	T_1(s)	T_2(s)	Desviación Estándar	Promedio	Mejor	Peor
PSO-LD	G_2	37.29	1.2197	1	0.0356	0.6917	13.9374	14.1578	12.0357
	G_3	400	1	1	0.0261				
PSO-OIW	G_2	39.75	49.1465	0.0868	0.0489	0.4835	14.1358	14.5754	13.7565
	G_3	400	1	1	0.0305				
PSO-OED	G_2	400	1	0.6001	0.0050	0.1208	14.5941	14.7762	14.0978
	G_3	34.77	50	1	0.0470				
PSO-POED	G_2	400	0.1038	1	0.0271	4.1126	35.9307	39.9424	20.1955
propuesto	G_3	400	0.1457	1	0.0464				

En la Tabla 5.5 se observa que el algoritmo propuesto (PSO - POED) es superior en varios aspectos a los algoritmos alternativos (PSO - LD, PSO - OIW y PSO - OED), principalmente al observar el mejor resultado obtenido de las 100 iteraciones, así como el peor resultado y el resultado promedio, que resultan ser mayores que los obtenidos con los otros algoritmos. El valor de la desviación estándar obtenido mediante el uso del PSO - POED es mayor que los demás, ya que los resultados obtenidos son significativamente altos. De hecho, la peor solución encontrada por PSO - POED supera ampliamente a las mejores soluciones de los otros algoritmos.

En la Tabla 5.6 se muestran los valores de los parámetros de la señal de tensión de los generadores en el Análisis de Respuesta Temporal, los cuales fueron obtenidos al aplicar los diferentes algoritmos.

Tabla 5.6: Parámetros de la señal de tensión en el Análisis de Respuesta Temporal de ambos generadores usando los diferentes algoritmos.

		Tensión terminal			
		t_r(s)	t_s(s)	$E_{ss}(\%)$	$M_p(\%)$
PSO-LD	G_2	1.1417	1.2517	2.5856	4.4956
PSO-OIW	G_2	1.1517	1.2517	3.0245	4.3509
PSO-OED	G_2	1.1416	1.2417	2.5890	3.9360
PSO-POED propuesto	G_2	0.2617	0.3017	0.0112	2.0500
PSO-LD	G_3	1.1050	1.2117	2.3702	4.0054
PSO-OIW	G_3	1.1050	1.2117	2.3655	4.0567
PSO-OED	G_3	1.1050	1.2117	2.6455	3.9099
PSO-POED propuesto	G_3	0.2417	0.2817	0.0077	2.0500

En la Tabla 5.6 se observa que al aplicar el algoritmo propuesto (PSO - POED), se obtienen mejores valores de los parámetros de tiempo de subida (t_r), tiempo de establecimiento (t_s), error en estado estacionario (E_{ss}) y Máximo sobrepaso (M_p) que al aplicar los algoritmos alternativos (PSO - LD, PSO - OIW y PSO - OED), lo cual se cumple para ambos generadores síncronos.

Capítulo 6

Conclusiones y Comentarios Finales

En esta tesis se propuso una nueva variante del algoritmo PSO clásico, llamada Optimización por Enjambre de Partículas con Decaimiento Exponencial Oscilatorio Periódico (PSO - POED). Este nuevo algoritmo fue ideado con la intención de mejorar la calidad del proceso de búsqueda del punto óptimo de una función objetivo para la sintonía simultánea de los parámetros del AVR y PSS, que están asociados al control de la tensión y estabilidad de generadores síncronos, así como para mejorar la tasa de convergencia de las simulaciones.

Por medio del análisis del ángulo del rotor y de la tensión en los terminales de los generadores de los sistemas de prueba analizados (Sistema Máquina Barra - Infinita e IEEE de 9 barras), se concluye que el método propuesto presenta un mejor desempeño, tanto por la calidad de los valores determinados como por la tasa de convergencia de las simulaciones. Así, para efectos comparativos, se emplearon también otros métodos alternativos en estos sistemas de prueba, siendo estos el PSO - LD, PSO - OIW y PSO - OED. El algoritmo propuesto demostró ser superior a estos algoritmos alternativos, ya que en ambos casos de prueba, PSO - POED presentó menorores valores de máximo sobrepaso (M_p), tiempo de establecimiento (t_s), tiempo de subida (t_r) y error en estado estacionario (E_{ss}). Además, en esta tesis se incluyeron dos aplicaciones del algoritmo PSO clásico en los sistemas de potencia, cuyos resultados, si bien son satisfactorios, requieren que el PSO clásico sea perfeccionado para dar solución a problemas de mayor complejidad.

El método propuesto PSO - POED demostró ser suficientemente robusto, ya que obtuvo un buen desempeño en ambos sistemas de prueba aun con perturbaciones de corto circuito, logrando corregir en corto tiempo la caída de la señal, obteniendo los mejores valores de parámetros posibles en el Análisis de Respuesta Temporal, lo cual atribuye calidad y confianza a este algoritmo.

Para el desarrollo de la presente tesis se usaron los programas DIgSILENT PowerFactory, su entorno de programación DPL y Matlab. El programa DIgSILENT se empleó para diseñar los sistemas de prueba y para programar el algoritmo propuesto, así como los algoritmos alternativos. Por su parte, Matlab fue empleado para simular los algoritmos de optimización.

Para trabajos futuros se sugiere incluir un controlador PID sintonizable y aplicar otros algoritmos de optimización metaheurística, tales como Recocido Simulado o Forraje Bacterial. Además, se sugiere sintonizar los demás parámetros del AVR y PSS con la intención de verificar si los resultados que se obtendrían compensarían el mayor tiempo de simulación que implica sintonizar más parámetros.

Bibliografía

[1] H. Saadat, *Power System Analysis*. McGraw - Hill, 1999.

[2] N. Hatziargyriou, J. Milanovic, C. Rahmann, V. Ajjarapu, C. Canizares, I. Erlich, D. Hill, I. Hiskens, I. Kamwa, B. Pal, P. Pourbeik, J. Sanchez-Gasca, A. Stankovic, T. Custem, V. Vittal, and C. Vournas, "Definition and classification of power system stability - revisited & extended," in *IEEE Transactions on Power Systems*, vol. 36, pp. 3271–3281, 2021.

[3] K. Prabha, *Power System Stability and Control*. McGraw - Hill, 1994.

[4] M. A. Golato, "Análisis de respuestas transitorias - sistemas de segundo orden." urlhttps://catedras.facet.unt.edu.ar/sistemasdecontrol, 2015.

[5] F. W. Rodrigues, *Projeto Simultâneo do Regulador Automático de Tensão e Estabilizador de Sistema de Potência utilizando Otimização por Enxame de Partículas Modificado*. PhD thesis, Universidade Federal de Paraiba, 2019.

[6] A. Kumar and S. Bawa, "A comparative review of meta - heuristic approaches to optimize the sla violation costs for dynamic execution of cloud services," *Soft Computing*, vol. 24, 2020.

[7] F. Cabezas-Soldevilla and F. Cabezas-Huerta, "Minimization of losses in power systems by reactive power dispatch using particle swarm optimization," in *54th International Universities Power Engineering Conference (UPEC)*, 2019.

[8] F. Cabezas-Soldevilla and F. Cabezas-Huerta, "Estimation of transmission lines parameters using particle swarm optimization," in *IEEE PES Transmission & Distribution Conference and Exhibition - Latin America (T&D-LA)*, 2018.

[9] F. Rodrigues, Y. Molina, C. Silva, and Z. Ñaupari, "Simultaneous tuning of the avr and pss parameters using particle swarm optimization with oscillating exponential decay," *Electrical Power and Energy Systems*, vol. 133, 2021.

[10] I. Manuaba, A. P. Muhammad Abdillah, and M. Purnomo, "Coordinated tuning of pid - based pss and avr using bacterial foraging - psotvac - de algorithm," *Control and Intelligent Systems*, vol. 43, no. 3, pp. 125–133, 2015.

[11] J. Usman, M. W. Mustafa, and G. Aliyu, "Design of avr and pss for power system stability based on iteration particle swarm optimization," *International Journal of Engineering and Innovative Technology*, vol. 2, no. 6, pp. 307–314, 2012.

[12] J. F. Nirmal and D. J. Auxillia, "Adaptive pso based tuning of pid controller for an automatic voltage regulator system," in *2013 International Conference on Circuits, Power and Computing Technologies [ICCPCT - 2013]*, vol. 1, pp. 661–666, 2013.

[13] B. Selvabala and D. Devaraj, "Coordinated tuning of avr - pss using differential evolution algorithm," in *2010 Conference Proceedings IPEC*, vol. 1, pp. 439–444, 2010.

[14] P. Mitra, S. P. Chowdhury, S. Chowdhury, K. Pal, and P. A. Crossley, "Intelligent avr and pss with adaptive hybrid learning algorithm," in *2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century*, 2008.

[15] R. Khezri and H. Bevrani, "Fuzzy - based coordinated control design for avr and pss in multi-machine power systems," in *2013 Iranian Conference on Fuzzy Systems (IFSC)*, 2013.

[16] B. Selvabala and D. Devaraj, "Coordinated design of avr - pss using multi objective genetic algorithm," in *International Conference on Swarm, Evolutionary, and Memetic Computing*, vol. 1, pp. 481–493, 2010.

[17] I. P. T Spoljaric and T. Alinjak, "Performance comparison of no - preference and weighted sum objective methods in multi - objective optimization of avr - pss tuning in multi - machine power system," in *Tehnički vjesnik*, vol. 29, pp. 1931–1940, 2022.

[18] G. Dudgeon, W. Leithead, A. Dysko, J. O'Reilly, and J. McDonald, "The effective role of avr and pss in power systems: Frequency response analysis," in *IEEE Transactions on Power Systems*, vol. 22, pp. 1986–1994, 2007.

[19] J. N. Rodrigues-Da-Silva-Júnior, "Sintonía Ótima de regulador automático de tensão e estabilizador de sistema de potência utilizando algoritmo de otimização por enxame de partículas," Master's thesis, Universidade Federal do Ceará, 2012.

[20] P. A. A. Ananthi and P. Kumar, "Reactive power optimization based on pso and considering voltage security in a practical power system," in *International Conference on Emerging Trends in Science, Engineering and Technology.*, 2012.

[21] J. Abugri and M. Karam, "Particle swarm optimization for the minimization of power losses in distribution networks," in *12th International Conference on Information Technology - New Generations*, 2015.

[22] R. Zimmerman, "Particle swarm optimization," in *IEEE Transactions on Power Systems*, vol. 26, pp. 12–19, 2010.

[23] C. Taylor, *Power System Voltage Stability*. McGraw - Hill, 1994.

[24] Z. Zhu, *Optimization of Power System Operation*. WILEY - IEEE Press, 2015.

[25] J. Kennedy and E. Russell, "Particle swarm optimization," in *Proceedings of IEEE International Conference of Neural Networks*, vol. 1, pp. 1942–1948, 1995.

[26] B. Akbal, "Pso and csa to estimate of parameter in power line," in *Proceedings of the International Conference on ELECTRONICS, COMPUTERS and ARTIFICIAL INTELLIGENCE - ECAI*, 2013.

[27] G. Kusic, *Computer Aided Power Systems Analysis*. Prentice Hall, 1986.

[28] J. Grainger and W. Stevenson, *Power Generation, Operation and Control*. Mc Graw Hill, 1995.

[29] A. Wood, *Power Generation, Operation and Control*. WILEY, 2013.

[30] A. Debs, "Estimation of steady state power system model parameters," in *IEEE Transactions on Power Apparatus and Systems*, pp. 1260–1268, 1974.

[31] F. Schweppe, "Static state estimation in electric power systems," in *Proceedings of the IEEE*, vol. 62, pp. 972–982, 1974.

Apéndice A

Código DPL: Sistema Máquina - Barra Infinita

```
!Programa inicial
int i,iter,k_veces,g_indice,g_best_indice;

!En el generador G1 se sintonizan 4 parámetros (1 del AVR y 3 del PSS): 4 parámetros en total
double G1_Ka,G1_K,G1_T1,G1_T2,
V_G1_Ka,V_G1_K,V_G1_T1,V_G1_T2,
local_best,global_best,
g_G1_Ka,g_G1_K,g_G1_T1,g_G1_T2,
pG1_Ka,pG1_K,pG1_T1,pG1_T2,
w,r1,r2,FO,FO_G1,
maxKa,minKa,maxK,minK,maxT1,minT1,maxT2,minT2,
Start,Stop,Time,
g_Mp_G1,g_Ess_G1,g_ts_G1,g_tr_G1,
VEL,rdn;

int error,n,m,nW;
double dato;
string strname;

ResetCalculation();
```

```
Start = GetTime(1); ! Obtener hora actual
ClearOutput();

!Calcular los valores de W
MODEL_W:wmax=wmax;
MODEL_W:wmin=wmin;
MODEL_W:IterMax=MaxIter;
MODEL_W:Mod=ModW;
MODEL_W:ciclos=ciclos;
MODEL_W:M=M;
MODEL_W:theta=theta;
MODEL_W.Execute();
!————————————
g_best_indice=-1000;
k_veces=1;

MatSOL.Init(1,1);
while (k_veces<=Veces){
printf('\n Inicio Simulación %.0f',k_veces);
printf('————————————-\n');
global_best=-10000;
maxKa=400;
minKa=1;
maxK=50;
minK=0.1;
maxT1=1;
minT1=0.6;
maxT2=0.05;
minT2=0.005;
VEL =0.50;
```

```
!Inicializando Vectores y Matrices
Mat_FO.Init(1,1);
Mat_AVRG1_Ka.Init(1,1);
Mat_PSSG1_K.Init(1,1);
Mat_PSSG1_T1.Init(1,1);
Mat_PSSG1_T2.Init(1,1);

MatV_AVRG1_Ka.Init(1,1);
MatV_PSSG1_K.Init(1,1);
MatV_PSSG1_T1.Init(1,1);
MatV_PSSG1_T2.Init(1,1);
Vec_pG1_Ka.Init(1);
Vec_pG1_K.Init(1);
Vec_pG1_T1.Init(1);
Vec_pG1_T2.Init(1);

!Generando los valores iniciales y velocidad inicial de la población
i=1;
while(i<=nPop){

! Valores iniciales
G1_Ka=Random(minKa,maxKa);
Mat_AVRG1_Ka.Set(i,1,G1_Ka);
G1_K=Random(minK,maxK);
Mat_PSSG1_K.Set(i,1,G1_K);
G1_T1=Random(minT1,maxT1);
Mat_PSSG1_T1.Set(i,1,G1_T1);
G1_T2=Random(minT2,maxT2);
Mat_PSSG1_T2.Set(i,1,G1_T2);

! Velocidades Iniciales
V_G1_Ka=Random(-VEL*maxKa,VEL*maxKa);
```

```
MatV_AVRG1_Ka.Set(i,1,V_G1_Ka);
V_G1_K=Random(-VEL*maxK,VEL*maxK);
MatV_PSSG1_K.Set(i,1,V_G1_K);
V_G1_T1=Random(-VEL*maxT1,VEL*maxT1);
MatV_PSSG1_T1.Set(i,1,V_G1_T1);
V_G1_T2=Random(-VEL*maxT2,VEL*maxT2);
MatV_PSSG1_T2.Set(i,1,V_G1_T2);

!Calculo de FO
Teste_FO_PSO:AVR_G1_Ka=G1_Ka;
Teste_FO_PSO:PSS_G1_K=G1_K;
Teste_FO_PSO:PSS_G1_T1=G1_T1;
Teste_FO_PSO:PSS_G1_T2=G1_T2;
Teste_FO_PSO.Execute();

FO=Teste_FO_PSO:FO;
FO_G1=Teste_FO_PSO:FO_G1;

Mat_FO.Set(i,1,FO);

! Determinados los parámetros del Best Local
pG1_Ka=G1_Ka;
Vec_pG1_Ka.Set(i,pG1_Ka);
pG1_K=G1_K;
Vec_pG1_K.Set(i,pG1_K);
pG1_T1=G1_T1;
Vec_pG1_T1.Set(i,pG1_T1);
pG1_T2=G1_T2;
Vec_pG1_T2.Set(i,pG1_T2);
local_best = Mat_FO.Get(i,1);
Vec_local_best.Set(i,local_best) ;
if (local_best >global_best){
```

```
global_best = local_best;
g_G1_Ka = pG1_Ka;
g_G1_K = pG1_K;
g_G1_T1 = pG1_T1;
g_G1_T2 = pG1_T2;
printf('FO global = %f , FO_G1= %f –>C.Inicial –>Individuo = %d',FO,FO_G1,i);

!Se guardan los parametros Mp, Ess, ts, tr
g_Mp_G1 = Teste_FO_PSO:over_G1;
g_Ess_G1 = Teste_FO_PSO:erro_G1;
g_ts_G1 = Teste_FO_PSO:ta_G1;
g_tr_G1 = Teste_FO_PSO:ts_09_G1;
}
i+=1;
}
!————————————————————————
!————————- LOOP Principal ————————————-
iter = 0;
w=VEC_W.Get(iter+1);
printf('Valor de w= %f para iter= %d',w,iter);
while (iter<= MaxIter) {
for (i = 1; i<= nPop; i = i+1){

! Actualizando las velocidades
G1_Ka = Mat_AVRG1_Ka.Get(i,iter+1);
G1_K = Mat_PSSG1_K.Get(i,iter+1);
G1_T1 = Mat_PSSG1_T1.Get(i,iter+1);
G1_T2 = Mat_PSSG1_T2.Get(i,iter+1);
V_G1_Ka = MatV_AVRG1_Ka.Get(i,iter+1);
V_G1_K = MatV_PSSG1_K.Get(i,iter+1);
V_G1_T1 = MatV_PSSG1_T1.Get(i,iter+1);
V_G1_T2 = MatV_PSSG1_T2.Get(i,iter+1);
```

```
pG1_Ka = Vec_pG1_Ka.Get(i);
pG1_K = Vec_pG1_K.Get(i);
pG1_T1 = Vec_pG1_T1.Get(i);
pG1_T2 = Vec_pG1_T2.Get(i);

!——— Velocidad del parámetro G1_Ka————-
r1 = Random(0,1) ;
r2 = Random(0,1) ;
V_G1_Ka = w*V_G1_Ka + c1*r1*(pG1_Ka - G1_Ka) + c2*r2*(g_G1_Ka - G1_Ka);
if(V_G1_Ka >= VEL*maxKa){
V_G1_Ka = VEL*maxKa;
}
if(V_G1_Ka <= -VEL*maxKa){
V_G1_Ka = -VEL*maxKa;
}
MatV_AVRG1_Ka.Set(i,iter+2,V_G1_Ka);

!———- Velocidad del parámetro G1_K———-
r1 = Random(0,1) ;
r2 = Random(0,1) ;
V_G1_K = w*V_G1_K + c1*r1*(pG1_K - G1_K) + c2*r2*(g_G1_K - G1_K);
if(V_G1_K >= VEL*maxK){
V_G1_K = VEL*maxK;
}
if(V_G1_K <= -VEL*maxK){
V_G1_K = -VEL*maxK;
}
MatV_PSSG1_K.Set(i,iter+2,V_G1_K);

!——— Velocidad del parámetro G1_T1————-
r1 = Random(0,1) ;
r2 = Random(0,1) ;
```

```
V_G1_T1 = w*V_G1_T1 + c1*r1*(pG1_T1 - G1_T1) + c2*r2*(g_G1_T1 - G1_T1);
if(V_G1_T1 >= VEL*maxT1){
V_G1_T1 = VEL*maxT1;
}
if(V_G1_T1 <= -VEL*maxT1){
V_G1_T1 = -VEL*maxT1;
}
MatV_PSSG1_T1.Set(i,iter+2,V_G1_T1);

!——- Velocidad del parámetro G1_T2——
r1 = Random(0,1) ;
r2 = Random(0,1) ;
V_G1_T2 = w*V_G1_T2 + c1*r1*(pG1_T2 - G1_T2) + c2*r2*(g_G1_T2 - G1_T2);
if(V_G1_T2 >= VEL*maxT2){
V_G1_T2 = VEL*maxT2;
}
if(V_G1_T2 <= -VEL*maxT2){
V_G1_T2 = -VEL*maxT2;
}
MatV_PSSG1_T2.Set(i,iter+2,V_G1_T2);

!—— Actualizando las posiciones de los parámetros ——
G1_Ka = G1_Ka + V_G1_Ka;
if(G1_Ka>maxKa){
G1_Ka=maxKa;
} if(G1_Ka<minKa){
G1_Ka=minKa;
}
Mat_AVRG1_Ka.Set(i,iter+2,G1_Ka);
G1_K = G1_K + V_G1_K;
if(G1_K>maxK){
G1_K=maxK;
```

```
}
if(G1_K<minK){
G1_K=minK;
}
Mat_PSSG1_K.Set(i,iter+2,G1_K);
G1_T1 = G1_T1 + V_G1_T1;
if(G1_T1>maxT1){
G1_T1=maxT1;
}
if(G1_T1<minT1){
G1_T1=minT1;
}
Mat_PSSG1_T1.Set(i,iter+2,G1_T1);
G1_T2 = G1_T2 + V_G1_T2;
if(G1_T2>maxT2){
G1_T2=maxT2;
}
if(G1_T2<minT2){
G1_T2=minT2;
}
Mat_PSSG1_T2.Set(i,iter+2,G1_T2);

!—— Calculando la Función Objetivo de cada indivíduo ——
!Calculo de FO
Teste_FO_PSO:AVR_G1_Ka=G1_Ka;
Teste_FO_PSO:PSS_G1_K=G1_K;
Teste_FO_PSO:PSS_G1_T1=G1_T1;
Teste_FO_PSO:PSS_G1_T2=G1_T2;
Teste_FO_PSO.Execute();
FO=Teste_FO_PSO:FO;
FO_G1=Teste_FO_PSO:FO_G1;
```

```
Mat_FO.Set(i,iter+2,FO);

! Criterio para Best Local
local_best = Vec_local_best.Get(i);
if (FO>=local_best){
pG1_Ka=G1_Ka;
Vec_pG1_Ka.Set(i,pG1_Ka);
pG1_K=G1_K;
Vec_pG1_K.Set(i,pG1_K);
pG1_T1=G1_T1;
Vec_pG1_T1.Set(i,pG1_T1);
pG1_T2=G1_T2;
Vec_pG1_T2.Set(i,pG1_T2);
Vec_local_best.Set(i,FO);
}
if (FO >global_best){
global_best = FO;
g_G1_Ka = pG1_Ka;
g_G1_K = pG1_K;
g_G1_T1 = pG1_T1;
g_G1_T2 = pG1_T2;

!Se guardan los parámetros Mp, Ess, ts, tr
g_Mp_G1 = Teste_FO_PSO:over_G1;
g_Ess_G1 = Teste_FO_PSO:erro_G1;
g_ts_G1 = Teste_FO_PSO:ta_G1;
g_tr_G1 = Teste_FO_PSO:ts_09_G1;
printf('FO global = %f , FO_G1= %f –>iter= %d –>Individuo = %d',FO,FO_G1,iter,i);
}
}
```

```
!——— Actualizando la constante de inercia ———-
if (iter+1<= MaxIter) {
w=VEC_W.Get(iter+2);
printf('Valor de w=%f para iter=%d',w,iter+1);
}
iter += 1;
}
printf('\n Gráfica para FO =%f',global_best);
printf('\n Solucion');
printf('==========================================');
printf('G1:Ka=%f G1:K=%f G1:T1=%f G1:T2=%f –>FO=%f', g_G1_Ka,g_G1_K,g_G1_T1,g_G1_T2,gl
printf('Mp_G1=%f Ess_G1=%f ts_G1=%f tr_G1=%f', g_Mp_G1,g_Ess_G1,g_ts_G1,g_tr_G1);
printf('==========================================');
MatSOL.Set(k_veces,1,global_best);
MatSOL.Set(k_veces,2,g_G1_Ka);
MatSOL.Set(k_veces,3,g_G1_K);
MatSOL.Set(k_veces,4,g_G1_T1);
MatSOL.Set(k_veces,5,g_G1_T2);
MatSOL.Set(k_veces,6,g_Mp_G1);
MatSOL.Set(k_veces,7,g_Ess_G1);
MatSOL.Set(k_veces,8,g_ts_G1);
MatSOL.Set(k_veces,9,g_tr_G1);
if(global_best>=g_best_indice){
g_best_indice=global_best;
g_indice=k_veces;
}
k_veces+=1;
}

!Correr una simulación para la mejor opción
printf('\n ======SOLUCIÓN GENERAL======');
printf('Indice del mejor valor =%f',g_indice);
```

```
printf('Se actualiza los valores para la mejor solución');
Teste_FO_PSO:AVR_G1_Ka=MatSOL.Get(g_indice,2);
Teste_FO_PSO:PSS_G1_K=MatSOL.Get(g_indice,3);
Teste_FO_PSO:PSS_G1_T1=MatSOL.Get(g_indice,4);
Teste_FO_PSO:PSS_G1_T2=MatSOL.Get(g_indice,5);
Teste_FO_PSO.Execute();
Stop = GetTime(1);
Time = (Stop - Start)/60; !min
printf(' %s %f %s','Run Time:', Time, '[min]');

!———-Pasar los datos a Excel———-
printf('\n ==================== GUARDANDO EXCEL====================');
error =xlStart(); !arrancar Excel
if(error){
Error('No se puede abrir la aplicación Excel');
exit();
}

! Lectura del archivo
error = xlOpenWorkbook(Archivo);
if(error){
Error('No se puede abrir el archivo Excel');
xlTerminate();
exit();
}
strname=xlGetWorksheetName(1);
printf('Hoja - %s',strname);

! Activar hoja
error = xlActivateWorksheet(1);
if(error){
Error('No se puede abrir el archivo Excel');
```

```
xlTerminate();
exit();
}
xlSetValue(7,4,ModW);
xlSetValue(7,6,MaxIter);
xlSetValue(7,7,nPop);
xlSetValue(11,4,wmax);
xlSetValue(11,5,wmin);
xlSetValue(11,6,c1);
xlSetValue(11,7,c2);
nW=ModW;
if(nW=1.or.nW=2.or.nW=3){
xlSetValue(16,4,ciclos);
}
if(nW=3.or.nW=5){
xlSetValue(16,6,M);
}
if(nW=3){
xlSetValue(16,5,theta);
}
n=1;
while(n<=Veces){
m=1;
while(m<=9){
dato=MatSOL.Get(n,m);
xlSetValue(2+m,9+n,dato);
m+=1;
}
n+=1;
}
strname=xlGetWorksheetName(2);
printf('Hoja - %s',strname);
```

```
! Activar hoja
error = xlActivateWorksheet(2);
if(error){
Error('No se puede abrir el archivo Excel');
xlTerminate();
exit();
}
n=0;
while(n<=MaxIter){
dato=VEC_W.Get(n+1);
xlSetValue(3,4+n,dato);
n+=1;
}

! Guardar
error = xlSaveWorkbookAs(Archivo);
if(error){
Error('No se puede guardar el archivo Excel');
exit();
}
xlTerminate();

!'Modelo del Factor de Peso Inercial (ω)'
double m,s,Ed,Et,b,Rd,niter;

!PSO with Periodic Oscillating Exponential Decay-POED
if(Mod=3){
while(niter<=IterMax){
m=(wmax-wmin)/2;
s=(m+wmin);
Et=trunc(niter/M);
Ed=exp((M*Et-niter)/theta);
```

```
w=(cos(2*pi()*niter/ciclos)*m*Ed)+s;
Vec_W.Set(niter+1,w);

!Función objetivo
object ldf, Inc, sim;
int N,j,i;
double row_G1, max_G1,reg_G1,
N_1, N_11, N_2, N_22, N_3, N_33,
reg_09_G1,d_09_G1,
t_evento,t_ini,t_step,
vf_098_G1,vf_102_G1, du_G1,k,
No_G1,To,pre_1,pre_2,pre_3,pfalla_G1;

!Limpiar la ventana de salida
ResetCalculation();
EchoOff();

!==== Generación aleatoria de las partículas iniciales. ====
! Alterando la Ganancia Ka del G2 y generando la velocidad inicial de Ka del G2
AVR_G1:Ka=G1_Ka;!AVR_G1_Ka;
PSS_G1:K = G1_K;!PSS_G1_K;
PSS_G1:T1 = G1_T1;!PSS_G1_T1;
PSS_G1:T2 = G1_T2;!PSS_G1_T2;

!====== Cargando Condiciones Iniciales y Pertubación ======
! Ejecuta el flujo de carga y la Simulación
Inc = GetCaseObject('ComInc');
Inc:p_resvar=Resultado;
sim = GetCaseObject('ComSim');
sim.Execute();
EchoOff();
```

```
! Cargar el archivo de resultados Resultado en la memoria
LoadResData(Resultado);

! Calcula el número de líneas del Resultado 3
N = ResNval(Resultado,0);

! Obtiene el índice de la columna para la tensión 's: ut' para el objeto seleccionado
!=========TENSIÓN DE REFERENCIA DE LOS GENERADORES========
Resultado.GetValue(No_G1, 0,0);
Resultado.GetValue(To, 0,-1);

!===========ESTABILIDAD ANTES DE FALLA=============
! Calculando la fila apartir del tiempo del evento.
t_evento= EventFalla:time; !Tiempo que ocurre el evento
t_ini = Inc:tstart;!Tiempo de inicio de la simulación
t_step = Inc:dtgrd; !Paso de la simulacion

!=========== CALCULA EL OVERSHOOT ==========
row_G1=Resultado.FindMaxInColumn(0,max_G1);
over_G1 = (max_G1 - No_G1)*100;

!======== CALCULAR EL ERROR DE RÉGIMEN PERMANENTE =========
Resultado.GetValue(N_1, N-1,0);
Resultado.GetValue(N_2, N-2,0);
Resultado.GetValue(N_3, N-3,0);
reg_G1 = (N_1 + N_2 + N_3)/3 ;
erro_G1 = abs((No_G1 - reg_G1)*100) ;

!=========== CALCULAR EL TIEMPO DE SUBIDA ==========
reg_09_G1 = No_G1;
```

```
! Calculando el tiempo de subida para G2
j=trunc(abs(t_ini-t_evento)/t_step)+1;
while ( j <N ) {
Resultado.GetValue(d_09_G1, j,0);
if (d_09_G1 >= reg_09_G1) {
Resultado.GetValue(ts_09_G1, j,-1);
ts_09_G1=ts_09_G1-t_evento;
break;
}
j = j + 1;
}

!============ CALCULAR EL TIEMPO DE ACOMODACIÓN ===========
vf_098_G1 = 0.98 * No_G1;
vf_102_G1 = 1.02 * No_G1;
j=trunc(abs(t_ini-t_evento)/t_step)+1;
i=N-1;
while(i>=j)
Resultado.GetValue(du_G1, i, 0);
if( du_G1>=vf_102_G1 .or. du_G1<=vf_098_G1 ){
Resultado.GetValue(ta_G1, i, -1);
ta_G1=ta_G1-t_evento;
break;
}
i-=1;
}

!======== CALCULAR LA FUNCIÓN OBJETIVO ========
FO_G1 = 100*(1/((1 - exp(-betha))*(over_G1 + erro_G1) + (exp(-betha))*(ta_G1 + ts_09_G1)));
FO = FO_G1 ;
```

Apéndice B

Código DPL: Sistema IEEE de 9 barras

```
!Programa inicial
int i, iter, k_veces, g_indice, g_best_indice;

!Por cada generador (G2 y G3) se sintonizan 4 parámetros (1 del AVR y 3 del PSS): 8 en total
double G2_Ka,G3_Ka,G2_K,G3_K,G2_T1,G3_T1,G2_T2,G3_T2,
V_G2_Ka,V_G3_Ka,V_G2_K,V_G3_K,V_G2_T1,V_G3_T1,V_G2_T2,V_G3_T2,
local_best,global_best,
g_G2_Ka,g_G3_Ka,g_G2_K,g_G3_K,g_G2_T1,g_G3_T1,g_G2_T2,g_G3_T2,
pG2_Ka,pG3_Ka,pG2_K,pG3_K,pG2_T1,pG3_T1,pG2_T2,pG3_T2,
w,r1,r2,FO,FO_G2,FO_G3,
maxKa,minKa,maxK,minK,maxT1,minT1,maxT2,minT2,
Start,Stop,Time,
g_Mp_G2,g_Mp_G3,g_Ess_G2,g_Ess_G3,g_ts_G2,g_ts_G3,g_tr_G2,g_tr_G3,
VEL,rdn;

int error, n ,m, nW;
double dato;
string strname;

ResetCalculation();
Start = GetTime(1); ! Obtener hora actual
ClearOutput();
```

```
!Calcular los valores de W
MODEL_W:wmax=wmax;
MODEL_W:wmin=wmin;
MODEL_W:IterMax=MaxIter;
MODEL_W:Mod=ModW;
MODEL_W:ciclos=ciclos;
MODEL_W:M=M;
MODEL_W:theta=theta;
MODEL_W.Execute();

!————————————
g_best_indice=-1000;
k_veces=1;

MatSOL.Init(1,1);
while (k_veces<=Veces){
printf('\n Inicio Simulación %.0f',k_veces);
printf('————————————-\n');
global_best=-10000;
maxKa=400;
minKa=1;
maxK=50;
minK=0.1;
maxT1=1;
minT1=0.6;
maxT2=0.05;
minT2=0.005;
VEL =0.50;

!Inicializando Vectores y Matrices
Mat_FO.Init(1,1);
Mat_AVRG2_Ka.Init(1,1);
```

```
Mat_AVRG3_Ka.Init(1,1);
Mat_PSSG2_K.Init(1,1);
Mat_PSSG3_K.Init(1,1);
Mat_PSSG2_T1.Init(1,1);
Mat_PSSG3_T1.Init(1,1);
Mat_PSSG2_T2.Init(1,1);
Mat_PSSG3_T2.Init(1,1);

MatV_AVRG2_Ka.Init(1,1);
MatV_AVRG3_Ka.Init(1,1);
MatV_PSSG2_K.Init(1,1);
MatV_PSSG3_K.Init(1,1);
MatV_PSSG2_T1.Init(1,1);
MatV_PSSG3_T1.Init(1,1);
MatV_PSSG2_T2.Init(1,1);
MatV_PSSG3_T2.Init(1,1);

Vec_pG2_Ka.Init(1);
Vec_pG3_Ka.Init(1);
Vec_pG2_K.Init(1);
Vec_pG3_K.Init(1);
Vec_pG2_T1.Init(1);
Vec_pG3_T1.Init(1);
Vec_pG2_T2.Init(1);
Vec_pG3_T2.Init(1);

!Generando los valores iniciales y velocidad inicial de la población
i=1;
while(i<=nPop){

!Valores iniciales
G2_Ka=Random(minKa,maxKa);
```

```
Mat_AVRG2_Ka.Set(i,1,G2_Ka);

G3_Ka=Random(minKa,maxKa);
Mat_AVRG3_Ka.Set(i,1,G3_Ka);

G2_K=Random(minK,maxK);
Mat_PSSG2_K.Set(i,1,G2_K);

G3_K=Random(minK,maxK);
Mat_PSSG3_K.Set(i,1,G3_K);

G2_T1=Random(minT1,maxT1);
Mat_PSSG2_T1.Set(i,1,G2_T1);

G3_T1=Random(minT1,maxT1);
Mat_PSSG3_T1.Set(i,1,G3_T1);

G2_T2=Random(minT2,maxT2);
Mat_PSSG2_T2.Set(i,1,G2_T2);

G3_T2=Random(minT2,maxT2);
Mat_PSSG3_T2.Set(i,1,G3_T2);

!Velocidades Iniciales
V_G2_Ka=Random(-VEL*maxKa,VEL*maxKa);
MatV_AVRG2_Ka.Set(i,1,V_G2_Ka);

V_G3_Ka=Random(-VEL*maxKa,VEL*maxKa);
MatV_AVRG3_Ka.Set(i,1,V_G3_Ka);

V_G2_K=Random(-VEL*maxK,VEL*maxK);
MatV_PSSG2_K.Set(i,1,V_G2_K);
```

```
V_G3_K=Random(-VEL*maxK,VEL*maxK);
MatV_PSSG3_K.Set(i,1,V_G3_K);

V_G2_T1=Random(-VEL*maxT1,VEL*maxT1);
MatV_PSSG2_T1.Set(i,1,V_G2_T1);

V_G3_T1=Random(-VEL*maxT1,VEL*maxT1);
MatV_PSSG3_T1.Set(i,1,V_G3_T1);

V_G2_T2=Random(-VEL*maxT2,VEL*maxT2);
MatV_PSSG2_T2.Set(i,1,V_G2_T2);

V_G3_T2=Random(-VEL*maxT2,VEL*maxT2);
MatV_PSSG3_T2.Set(i,1,V_G3_T2);

!Calculo de FO
Teste_FO_PSO:AVR_G2_Ka=G2_Ka;
Teste_FO_PSO:AVR_G3_Ka=G3_Ka;
Teste_FO_PSO:PSS_G2_K=G2_K;
Teste_FO_PSO:PSS_G3_K=G3_K;
Teste_FO_PSO:PSS_G2_T1=G2_T1;
Teste_FO_PSO:PSS_G3_T1=G3_T1;
Teste_FO_PSO:PSS_G2_T2=G2_T2;
Teste_FO_PSO:PSS_G3_T2=G3_T2;
Teste_FO_PSO.Execute();

FO=Teste_FO_PSO:FO;
FO_G2=Teste_FO_PSO:FO_G2;
FO_G3=Teste_FO_PSO:FO_G3;
```

```
Mat_FO.Set(i,1,FO);

!Determinados los parámetros del Best Local
pG2_Ka=G2_Ka;
Vec_pG2_Ka.Set(i,pG2_Ka);

pG3_Ka=G3_Ka;
Vec_pG3_Ka.Set(i,pG3_Ka);

pG2_K=G2_K;
Vec_pG2_K.Set(i,pG2_K);

pG3_K=G3_K;
Vec_pG3_K.Set(i,pG3_K);

pG2_T1=G2_T1;
Vec_pG2_T1.Set(i,pG2_T1);

pG3_T1=G3_T1;
Vec_pG3_T1.Set(i,pG3_T1);

pG2_T2=G2_T2;
Vec_pG2_T2.Set(i,pG2_T2);

pG3_T2=G3_T2;
Vec_pG3_T2.Set(i,pG3_T2);

local_best = Mat_FO.Get(i,1);
Vec_local_best.Set(i,local_best);
```

```
if (local_best >= global_best)
global_best = local_best;

g_G2_Ka = pG2_Ka;
g_G3_Ka = pG3_Ka;

g_G2_K = pG2_K;
g_G3_K = pG3_K;

g_G2_T1 = pG2_T1;
g_G3_T1 = pG3_T1;

g_G2_T2 = pG2_T2;
g_G3_T2 = pG3_T2;

printf('FO global = %f, FO_G2= %f FO_G3= %f –>C. Inicial –>Individuo = %d',FO,FO_G2,FO_G3,i);

!Se guardan los parámetros Mp, Ess, ts, tr
g_Mp_G2 = Teste_FO_PSO:over_G2;
g_Mp_G3 = Teste_FO_PSO:over_G3;
g_Ess_G2 = Teste_FO_PSO:erro_G2;
g_Ess_G3 = Teste_FO_PSO:erro_G3;
g_ts_G2 = Teste_FO_PSO:ta_G2;
g_ts_G3 = Teste_FO_PSO:ta_G3;
g_tr_G2 = Teste_FO_PSO:ts_09_G2;
g_tr_G3 = Teste_FO_PSO:ts_09_G3;
}
i+=1;
}
```

```
!————————————————————————
!———————— LOOP Principal ————————————-
iter = 0;
w=VEC_W.Get(iter+1);
printf('Valor de w=%f para iter=%d',w,iter);

while (iter<= MaxIter) {
for (i = 1; i<= nPop; i = i+1){

!Actualizando las velocidades
G2_Ka = Mat_AVRG2_Ka.Get(i,iter+1);
G3_Ka = Mat_AVRG3_Ka.Get(i,iter+1);
G2_K = Mat_PSSG2_K.Get(i,iter+1);
G3_K = Mat_PSSG3_K.Get(i,iter+1);
G2_T1 = Mat_PSSG2_T1.Get(i,iter+1);
G3_T1 = Mat_PSSG3_T1.Get(i,iter+1);
G2_T2 = Mat_PSSG2_T2.Get(i,iter+1);
G3_T2 = Mat_PSSG3_T2.Get(i,iter+1);

V_G2_Ka = MatV_AVRG2_Ka.Get(i,iter+1);
V_G3_Ka = MatV_AVRG3_Ka.Get(i,iter+1);
V_G2_K = MatV_PSSG2_K.Get(i,iter+1);
V_G3_K = MatV_PSSG3_K.Get(i,iter+1);
V_G2_T1 = MatV_PSSG2_T1.Get(i,iter+1);
V_G3_T1 = MatV_PSSG3_T1.Get(i,iter+1);
V_G2_T2 = MatV_PSSG2_T2.Get(i,iter+1);
V_G3_T2 = MatV_PSSG3_T2.Get(i,iter+1);

pG2_Ka = Vec_pG2_Ka.Get(i);
pG3_Ka = Vec_pG3_Ka.Get(i);
pG2_K = Vec_pG2_K.Get(i);
pG3_K = Vec_pG3_K.Get(i);
```

```
pG2_T1 = Vec_pG2_T1.Get(i);
pG3_T1 = Vec_pG3_T1.Get(i);
pG2_T2 = Vec_pG2_T2.Get(i);
pG3_T2 = Vec_pG3_T2.Get(i);

!———— Velocidad del parámetro G2_Ka————-
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G2_Ka = w*V_G2_Ka + c1*r1*(pG2_Ka - G2_Ka) + c2*r2*(g_G2_Ka - G2_Ka);

if(V_G2_Ka >= VEL*maxKa){
V_G2_Ka = VEL*maxKa;
}
if(V_G2_Ka <= -VEL*maxKa){
V_G2_Ka = -VEL*maxKa;
}
MatV_AVRG2_Ka.Set(i,iter+2,V_G2_Ka);

!————- Velocidad del parámetro G3_Ka————
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G3_Ka = w*V_G3_Ka + c1*r1*(pG3_Ka - G3_Ka) + c2*r2*(g_G3_Ka - G3_Ka);

if(V_G3_Ka >= VEL*maxKa){
V_G3_Ka = VEL*maxKa;
} if(V_G3_Ka <= -VEL*maxKa){
V_G3_Ka = -VEL*maxKa;
}
MatV_AVRG3_Ka.Set(i,iter+2,V_G3_Ka);
```

```
!————— Velocidad del parámetro G2_K————————
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G2_K = w*V_G2_K + c1*r1*(pG2_K - G2_K) + c2*r2*(g_G2_K - G2_K);

if(V_G2_K >= VEL*maxK){
V_G2_K = VEL*maxK;
}
if(V_G2_K <= -VEL*maxK){
V_G2_K = -VEL*maxK;
}
MatV_PSSG2_K.Set(i,iter+2,V_G2_K);

!————— Velocidad del parámetro G3_K————————
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G3_K = w*V_G3_K + c1*r1*(pG3_K - G3_K) + c2*r2*(g_G3_K - G3_K);

if(V_G3_K >= VEL*maxK){
V_G3_K = VEL*maxK;
}
if(V_G3_K <= -VEL*maxK){
V_G3_K = -VEL*maxK;
}
MatV_PSSG3_K.Set(i,iter+2,V_G3_K);

!————————— Velocidad del parámetro G2_T1————————-
r1 = Random(0,1) ;
r2 = Random(0,1) ;
```

```
V_G2_T1 = w*V_G2_T1 + c1*r1*(pG2_T1 - G2_T1) + c2*r2*(g_G2_T1 - G2_T1);

if(V_G2_T1 >= VEL*maxT1){
V_G2_T1 = VEL*maxT1;
}
if(V_G2_T1 <= -VEL*maxT1){
V_G2_T1 = -VEL*maxT1;
}
MatV_PSSG2_T1.Set(i,iter+2,V_G2_T1);

!————— Velocidad del parámetro G3_T1————————
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G3_T1 = w*V_G3_T1 + c1*r1*(pG3_T1 - G3_T1) + c2*r2*(g_G3_T1 - G3_T1);

if(V_G3_T1 >= VEL*maxT1){
V_G3_T1 = VEL*maxT1;
}
if(V_G3_T1 <= -VEL*maxT1){
V_G3_T1 = -VEL*maxT1;
}
MatV_PSSG3_T1.Set(i,iter+2,V_G3_T1);
```

```
!———— Velocidad del parámetro G2_T2————
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G2_T2 = w*V_G2_T2 + c1*r1*(pG2_T2 - G2_T2) + c2*r2*(g_G2_T2 - G2_T2);

if(V_G2_T2 >= VEL*maxT2){
V_G2_T2 = VEL*maxT2;
}
if(V_G2_T2 <= -VEL*maxT2){
V_G2_T2 = -VEL*maxT2;
}
MatV_PSSG2_T2.Set(i,iter+2,V_G2_T2);

!————- Velocidad del parámetro G3_T2————
r1 = Random(0,1) ;
r2 = Random(0,1) ;

V_G3_T2 = w*V_G3_T2 + c1*r1*(pG3_T2 - G3_T2) + c2*r2*(g_G3_T2 - G3_T2);

if(V_G3_T2 >= VEL*maxT2){
V_G3_T2 = VEL*maxT2;
}
if(V_G3_T2 <= -VEL*maxT2){
V_G3_T2 = -VEL*maxT2;
}
MatV_PSSG3_T2.Set(i,iter+2,V_G3_T2);
```

```
!—- Actualizando las posiciones de los parámetros —-
G2_Ka = G2_Ka + V_G2_Ka;

if(G2_Ka>maxKa){
G2_Ka=maxKa;
}

if(G2_Ka<minKa){
G2_Ka=minKa;
}
Mat_AVRG2_Ka.Set(i,iter+2,G2_Ka);
G3_Ka = G3_Ka + V_G3_Ka;

if(G3_Ka>maxKa){
G3_Ka=maxKa;
}

if(G3_Ka<minKa){
G3_Ka=minKa;
}
Mat_AVRG3_Ka.Set(i,iter+2,G3_Ka);
G2_K = G2_K + V_G2_K;

if(G2_K>maxK){ G2_K=maxK; }

if(G2_K<minK){ G2_K=minK; } Mat_PSSG2_K.Set(i,iter+2,G2_K);
G3_K = G3_K + V_G3_K;

if(G3_K>maxK){
G3_K=maxK;
}
```

```
if(G3_K<minK){
G3_K=minK;
}
Mat_PSSG3_K.Set(i,iter+2,G3_K);
G2_T1 = G2_T1 + V_G2_T1;

if(G2_T1>maxT1){
G2_T1=maxT1;
}

if(G2_T1<minT1){
G2_T1=minT1;
}
Mat_PSSG2_T1.Set(i,iter+2,G2_T1);
G3_T1 = G3_T1 + V_G3_T1;

if(G3_T1>maxT1){
G3_T1=maxT1;
}

if(G3_T1<minT1){
G3_T1=minT1;
}
Mat_PSSG3_T1.Set(i,iter+2,G3_T1);
G2_T2 = G2_T2 + V_G2_T2;

if(G2_T2>maxT2){
G2_T2=maxT2;
}
```

```
if(G2_T2<minT2){
G2_T2=minT2;
}
Mat_PSSG2_T2.Set(i,iter+2,G2_T2);
G3_T2 = G3_T2 + V_G3_T2;

if(G3_T2>maxT2){
G3_T2=maxT2;
}

if(G3_T2<minT2){
G3_T2=minT2;
}
Mat_PSSG3_T2.Set(i,iter+2,G3_T2);

!—— Calculando la Funcion Objetivo de cada indivíduo ——-

!Calculo de FO
Teste_FO_PSO:AVR_G2_Ka=G2_Ka;
Teste_FO_PSO:AVR_G3_Ka=G3_Ka;
Teste_FO_PSO:PSS_G2_K=G2_K;
Teste_FO_PSO:PSS_G3_K=G3_K;
Teste_FO_PSO:PSS_G2_T1=G2_T1;
Teste_FO_PSO:PSS_G3_T1=G3_T1;
Teste_FO_PSO:PSS_G2_T2=G2_T2;
Teste_FO_PSO:PSS_G3_T2=G3_T2;
Teste_FO_PSO.Execute();

FO=Teste_FO_PSO:FO;
FO_G2=Teste_FO_PSO:FO_G2;
FO_G3=Teste_FO_PSO:FO_G3;
Mat_FO.Set(i,iter+2,FO);
```

```
! Critério para Best Local
local_best = Vec_local_best.Get(i);
if (FO>=local_best){

pG2_Ka=G2_Ka;
Vec_pG2_Ka.Set(i,pG2_Ka);

pG3_Ka=G3_Ka;
Vec_pG3_Ka.Set(i,pG3_Ka);

pG2_K=G2_K;
Vec_pG2_K.Set(i,pG2_K);

pG3_K=G3_K;
Vec_pG3_K.Set(i,pG3_K);

pG2_T1=G2_T1;
Vec_pG2_T1.Set(i,pG2_T1);

pG3_T1=G3_T1;
Vec_pG3_T1.Set(i,pG3_T1);

pG2_T2=G2_T2;
Vec_pG2_T2.Set(i,pG2_T2);

pG3_T2=G3_T2;
Vec_pG3_T2.Set(i,pG3_T2);

Vec_local_best.Set(i,FO);
}
```

```
if (FO >= global_best){
global_best = FO;
g_G2_Ka = pG2_Ka;
g_G3_Ka = pG3_Ka;
g_G2_K = pG2_K;
g_G3_K = pG3_K;
g_G2_T1 = pG2_T1;
g_G3_T1 = pG3_T1;
g_G2_T2 = pG2_T2;
g_G3_T2 = pG3_T2;

!Se guardan los parámetros Mp, Ess, ts, tr
g_Mp_G2 = Teste_FO_PSO:over_G2;
g_Mp_G3 = Teste_FO_PSO:over_G3;
g_Ess_G2 = Teste_FO_PSO:erro_G2;
g_Ess_G3 = Teste_FO_PSO:erro_G3;
g_ts_G2 = Teste_FO_PSO:ta_G2;
g_ts_G3 = Teste_FO_PSO:ta_G3;
g_tr_G2 = Teste_FO_PSO:ts_09_G2;
g_tr_G3 = Teste_FO_PSO:ts_09_G3;

printf('FO global = %f , FO_G2= %f FO_G3= %f –>iter= %d –>Individuo = %d',FO,FO_G2,FO_G3,iter,i);
}
}

!———— Actualizando la constante de inercia ————-
if (iter+1<= MaxIter) {
w=VEC_W.Get(iter+2);
printf('Valor de w= %f para iter= %d',w,iter+1);
}
```

```
iter += 1;
}

printf('\n Gráfica para FO =%f',global_best);
printf('\n Solucion');
printf('=====================================================================');
printf('G2:Ka=%f G3:Ka=%f G2:K=%f G3:K=%f G2:T1=%f G3_T1=%f G2:T2=%f
G3:T2=%f –>FO=%f',
g_G2_Ka,g_G3_Ka,g_G2_K,g_G3_K,g_G2_T1,g_G3_T1,g_G2_T2,g_G3_T2,global_best);
printf('Mp_G2=%f Mp_G3=%f Ess_G2=%f Ess_G3=%f ts_G2=%f ts_G3=%f tr_G2=%f
tr_G3=%f', g_Mp_G2,g_Mp_G3,g_Ess_G2,g_Ess_G3,g_ts_G2,g_ts_G3,g_tr_G2,g_tr_G3);
printf('=====================================================================');

MatSOL.Set(k_veces,1,global_best);
MatSOL.Set(k_veces,2,g_G2_Ka);
MatSOL.Set(k_veces,3,g_G3_Ka);
MatSOL.Set(k_veces,4,g_G2_K);
MatSOL.Set(k_veces,5,g_G3_K);
MatSOL.Set(k_veces,6,g_G2_T1);
MatSOL.Set(k_veces,7,g_G3_T1);
MatSOL.Set(k_veces,8,g_G2_T2);
MatSOL.Set(k_veces,9,g_G3_T2);
MatSOL.Set(k_veces,10,g_Mp_G2);
MatSOL.Set(k_veces,11,g_Mp_G3);
MatSOL.Set(k_veces,12,g_Ess_G2);
MatSOL.Set(k_veces,13,g_Ess_G3);
MatSOL.Set(k_veces,14,g_ts_G2);
MatSOL.Set(k_veces,15,g_ts_G3);
MatSOL.Set(k_veces,16,g_tr_G2);
MatSOL.Set(k_veces,17,g_tr_G3);
```

```
if(global_best>=g_best_indice){
g_best_indice=global_best;
g_indice=k_veces;
}
k_veces+=1;
}

!Correr una simulación para la mejor opción.
printf('\n=====SOLUCION GENERAL====');
printf('Índice del mejor valor =
printf('Se actualiza los valores para la mejor solución');
Teste_FO_PSO:AVR_G2_Ka=MatSOL.Get(g_indice,2);
Teste_FO_PSO:AVR_G3_Ka=MatSOL.Get(g_indice,3);
Teste_FO_PSO:PSS_G2_K=MatSOL.Get(g_indice,4);
Teste_FO_PSO:PSS_G3_K=MatSOL.Get(g_indice,5);
Teste_FO_PSO:PSS_G2_T1=MatSOL.Get(g_indice,6);
Teste_FO_PSO:PSS_G3_T1=MatSOL.Get(g_indice,7);
Teste_FO_PSO:PSS_G2_T2=MatSOL.Get(g_indice,8);
Teste_FO_PSO:PSS_G3_T2=MatSOL.Get(g_indice,9);
Teste_FO_PSO.Execute();

Stop = GetTime(1);
Time = (Stop - Start)/60; !min
printf(' %s %f %s','Run Time:', Time, '[min]');

!————————Pasar los datos a Excel————————
printf('\n==================== GUARDANDO EXCEL====================');
error =xlStart(); !arrancar Excel
if(error){
Error('No se puede abrir la aplicación Excel');
exit();
}
```

```
! Lectura del archivo
error = xlOpenWorkbook(Archivo);
if(error){
Error('No se puede abrir el archivo Excel');
xlTerminate();
exit();
}
strname=xlGetWorksheetName(1);
printf('Hoja - %s',strname);

! Activar hoja
error = xlActivateWorksheet(1);
if(error){
Error('No se puede abrir el archivo Excel');
xlTerminate();
exit();
}

xlSetValue(7,4,ModW);
xlSetValue(7,6,MaxIter);
xlSetValue(7,7,nPop);
xlSetValue(11,4,wmax);
xlSetValue(11,5,wmin);
xlSetValue(11,6,c1);
xlSetValue(11,7,c2);

if(nW=1.or.nW=2.or.nW=3){
xlSetValue(16,4,ciclos);
}

if(nW=3.or.nW=5){
xlSetValue(16,6,M);
```

```
}

if(nW=3){
xlSetValue(16,5,theta);
}

n=1;
while(n<=Veces){
m=1;
while(m<=17){
dato=MatSOL.Get(n,m);
xlSetValue(2+m,9+n,dato);
m+=1;
}
n+=1;
}

strname=xlGetWorksheetName(2);
printf('Hoja - %s',strname);

! Activar hoja
error = xlActivateWorksheet(2);
if(error){
Error('No se puede abrir el archivo Excel');
xlTerminate();
exit();
}
```

```
n=1;
while(n<=Veces){
dato=VEC_W.Get(n);
xlSetValue(3,3+n,dato);
n+=1;
}

! Guardar
error = xlSaveWorkbookAs(Archivo);
if(error){
Error('No se puede guardar el archivo Excel');
exit();
}
xlTerminate();

'Modelo del Factor de Peso Inercial (ω)'
double m,s,Ed,Et,b,Rd,niter;
Vec_W.Init(1);
niter=0;
!PSO with Periodic Oscillating Exponential Decay-POED
if(Mod=3){
while(niter<=IterMax){
m=(wmax-wmin)/2;
s=(m+wmin);
Et=trunc(niter/M);
Ed=exp((M*Et-niter)/theta);
w=(cos(2*pi()*niter/ciclos)*m*Ed)+s;
Vec_W.Set(niter+1,w);
niter+=1;
}
}
```

```
!Función objetivo
object ldf, Inc, sim;
int N,j,i;
double row_G2,row_G3, max_G2, max_G3,reg_G2,reg_G3,
N_1, N_11, N_2, N_22, N_3, N_33,
reg_09_G2,reg_09_G3,d_09_G2,d_09_G3,
t_evento,t_ini,t_step,
vf_098_G2,vf_102_G2, vf_098_G3, vf_102_G3, du_G2, du_G3,k,
No_G2,No_G3,To,pre_1,pre_2,pre_3,pfalla_G2,pfalla_G3;

!Limpiar la ventana de salida
ResetCalculation();
EchoOff();

!===== Generación aleatoria de las partículas iniciales. =====
!Alterando la Ganancia Ka del G2 y generando la velocidad inicial de Ka del G2
AVR_G2:Ka=AVR_G2_Ka;
AVR_G3:Ka=AVR_G3_Ka;
PSS_G2:K = PSS_G2_K;
PSS_G2:T1 = PSS_G2_T1;
PSS_G2:T2 = PSS_G2_T2;
PSS_G3:K = PSS_G3_K;
PSS_G3:T1 = PSS_G3_T1;
PSS_G3:T2 = PSS_G3_T2;

!======== Cargando Cond. Iniciales y Pertubación =========
!Ejecuta el flujo de carga y la Simulación
Inc = GetCaseObject('ComInc');
Inc:p_resvar=Resultado;
sim = GetCaseObject('ComSim');
sim.Execute();
```

```
EchoOff();

! Cargar el archivo de resultados Resultado en la memoria
LoadResData(Resultado);

! Calcula el número de líneas del Resultado 3
N = ResNval(Resultado, 0);

! Obtiene el índice de la columna para la tensión 's: ut' para el objeto seleccionado
!=======TENSIÓN DE REFERENCIA DE LOS GENERADORES=======
Resultado.GetValue(No_G2, 0, 0);
Resultado.GetValue(No_G3, 0, 2);
Resultado.GetValue(To, 0, -1);

!==============ESTABILIDAD ANTES DE FALLA==============
! Calculando la fila a partir del tiempo del evento.
t_evento= EventFalla:time; !Tiempo que ocurre el evento
t_ini = Inc:tstart;!Tiempo de inicio de la simulación
t_step = Inc:dtgrd; !Paso de la simulacion

!============= CALCULA EL OVERSHOOT =============
row_G2=Resultado.FindMaxInColumn(0,max_G2);
row_G3=Resultado.FindMaxInColumn(2,max_G3);
over_G2 = (max_G2 - No_G2)*100;
over_G3 = (max_G3 - No_G3)*100;

!========= CALCULAR EL ERROR DE RÉGIMEN PERMANENTE ========
Resultado.GetValue(N_1, N-1,0);
Resultado.GetValue(N_2, N-2,0);
Resultado.GetValue(N_3, N-3,0);
Resultado.GetValue(N_11, N-1,2);
Resultado.GetValue(N_22, N-2,2);
```

```
Resultado.GetValue(N_33, N-3,2);
reg_G2 = (N_1 + N_2 + N_3)/3 ;
erro_G2 = abs((No_G2 - reg_G2)*100) ;
reg_G3 = (N_11 + N_22 + N_33)/3 ;
erro_G3 = abs((No_G3 - reg_G3)*100) ;

!================ CALCULAR EL TEMPO DE SUBIDA ============
reg_09_G2 = No_G2;
reg_09_G3 = No_G3;

! Calculando el tiempo de subida para G2
j=trunc(abs(t_ini-t_evento)/t_step)+1;
while ( j <N ) {
Resultado.GetValue(d_09_G2, j,0);
if (d_09_G2 >= reg_09_G2) {
Resultado.GetValue(ts_09_G2, j,-1);
ts_09_G2=ts_09_G2-t_evento;
break;
}
j = j + 1;
}

! Calculando el tiempo de subida para G3
j=trunc(abs(t_ini-t_evento)/t_step)+1;
while ( j <N ) {
Resultado.GetValue(d_09_G3, j,2);
if (d_09_G3 >= reg_09_G3) {
Resultado.GetValue(ts_09_G3, j,-1);
ts_09_G3=ts_09_G3-t_evento;
break;
}
j = j + 1;
```

```
}

!============= CALCULAR EL TIEMPO DE ACOMODACIÓN ============
vf_098_G2 = 0.98 * No_G2;
vf_102_G2 = 1.02 * No_G2;
vf_098_G3 = 0.98 * No_G3;
vf_102_G3 = 1.02 * No_G3;

! Calculando el tiempo de acomodación para G2
j=trunc(abs(t_ini-t_evento)/t_step)+1;
i=N-1;
while(i>=j){
Resultado.GetValue(du_G2, i, 0);
if( du_G2>=vf_102_G2 .or. du_G2<=vf_098_G2 ){
Resultado.GetValue(ta_G2, i, -1);
ta_G2=ta_G2-t_evento;
break;
}
i-=1;
}
i=N-1;
while(i>=j){
Resultado.GetValue(du_G3, i, 2);
if( du_G3>vf_102_G3 .or. du_G3<vf_098_G3 ){
Resultado.GetValue(ta_G3, i, -1);
ta_G3=ta_G3-t_evento;
break;
}
i-=1;
}
```

```
!============= CALCULAR LA FUNCIÓN OBJETIVO ===============
FO_G2 = 100*(1/((1 - exp(-betha))*(over_G2 + erro_G2) + (exp(-betha))*(ta_G2 + ts_09_G2)));
FO_G3 = 100*(1/((1 - exp(-betha))*(over_G3 + erro_G3) + (exp(-betha))*(ta_G3 + ts_09_G3)));
FO = FO_G2 + FO_G3 ;
```