IMPLEMENTACIÓN DE UN ALGORITMO BIOINSPIRADO PARA LA SINTONIZACIÓN DE CONTROLADORES PID EN UN RECTIFICADOR PFC BOOST EN MEDIO PUENTE

JULIÁN ANDRÉS TORRES MONTUFAR
YERSON ESTEBAN ROLDÁN TORRES

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2022

IMPLEMENTACIÓN DE UN ALGORITMO BIOINSPIRADO PARA LA SINTONIZACIÓN DE CONTROLADORES PID EN UN RECTIFICADOR PFC BOOST EN MEDIO PUENTE

JULIÁN ANDRÉS TORRES MONTUFAR
YERSON ESTEBAN ROLDÁN TORRES

Proyecto de grado presentado como requisito para optar al título de
INGENIERO ELECTRÓNICO

DIRECTOR: ING. JOSÉ GUILLERMO GUARNIZO MARÍN. PhD
CODIRECTOR: ING. JHON FREDY BAYONA NAVARRO. MSc.

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2022

Dedicatoria

A mi madre Maria Cristina por creer en mis habilidades y mi inteligencia. Por su incondicional apoyo en todo momento de la carrera. Le agradezco sus enseñanzas, su paciencia y su amor. A mis familiares por todo el apoyo brindado durante los años de estudio.

Julian Andres Torres Montufar

A mis padres Yalbleidy y Ramón por mantener en mi la motivación de continuar sin importar las caídas, por su paciencia, enseñanzas, apoyo y amor incondicional a lo largo de la vida, a mi abuela y demás allegados que de manera directa o indirecta han contribuido positivamente en mí.

Yerson Esteban Roldán Torres

Resumen

En el presente documento muestra la sintonización de parámetros en un controlador PID y un filtro precompensador en un rectificador PFC Halft Bridge Boost. Se hace una investigación para determinar un algoritmo Bioinspirado que pueda cumplir con la optimización que se necesita en este proyecto. Se escoge el algoritmo de selección clonal basado en el sistema inmune natural y se realiza la implementación en MATLAB® SIMULINK®. Las constantes del controlador PID y el filtro son representadas por las poblaciones de antígenos del algoritmo. Para la sintonización se escoge una población inicial de 200 antígenos que se mantiene en todas las iteraciones y una población memoria de 20 anticuerpos, al final de las iteraciones está población de memoria representa el conjunto de soluciones de la sintonización, con varios óptimos locales y un óptimo global, la población inicial no se genera de forma aleatoria sino un rango acotado a partir de una solución funcional en uno de los puntos de operación del rectificador. El error a minimizar es de la variable factor de potencia, es utilizada en la función de aptitud. En la búsqueda poblaciones se establecen limites en los cuales se puede mover las constantes, debido a que el sistema puede llegar a desestabilizarse. La población resultante del algoritmo logra alcanzar un factor de potencia de 0.998 con una carga resistiva de 6.6 kΩ. Se comprueba las constantes en un prototipo real de la planta y se comparan con los resultados simulados. En el prototipo también se analiza la robustez de la solución cambiando el punto de operación del convertidor.

Glosario

- ***HBB***: Half Bridge Boost.
- ***PI***: Proporcional, Integral.
- ***PID***: Proporcional, Integral, Derivativo.
- ***FOPID***: Proporcional, Integral, Derivativo de orden fraccional.
- ***PFC***: Corrección del factor de potencia.
- ***AC***: Corriente alterna.
- ***DC***: Corriente directa.
- ***CREG***: Comisión de Regulación de Energía y Gas.
- ***RMSE***: Error cuadrático medio.

Índice general

Índice de figuras

Índice de cuadros

.

Capítulo 1

Introducción

Los convertidores eléctricos son una alternativa de gran uso como fuentes de energía eléctrica, no solo en temas de eficiencia relacionando la potencia de salida respecto a la potencia de entrada, sino también la posibilidad de controlar sus variables a partir del modelo del determinado convertidor, las técnicas de control varían dependiendo la robustez que se desee y la naturaleza del convertidor. Uno de los esquemas de control más utilizados son el PID ya que presenta simplicidad en su implementación, pero no robustez reflejando errores en sus variables cuando se varia la carga. Como solución a lo dicho anteriormente se plantea la implementación de un algoritmo Bioinspirado para el cálculo de las constantes de controlador.

En el presente documento se desarrolla la implementación de un algoritmo basado en el principio de selección clonal utilizado en la respuesta adaptativa del sistema inmune, con la cual se busca sintonizar un controlador PID y un filtro precompensador en el lazo de corriente del convertidor HBB. El algoritmo tiene como propósito minimizar una función objetivo que está relacionada con los errores de los parámetros a controlar, realizando clonaciones y mutaciones cíclicamente hasta encontrar el mínimo valor de error.

La conformación del documento está dividida en capítulos y se encuentra distribuida de la siguiente manera: los capítulos uno y dos corresponden a introducción y planteamiento del problema en estos se busca mencionar la problemática y sustentar una solución, posteriormente se encuentra el estado del arte donde se indagan los antecedentes en las áreas de estudio dando como resultado la justificación del porqué la realización del trabajo, en el quinto capítulo se tendrán los objetivos a cumplir, posteriormente un marco teórico, a partir del séptimo capitulo se lleva a cabo de manera detallada el desarrollo del proyecto, los resultados obtenidos y conclusiones.

Capítulo 2

Planteamiento del problema

El uso de elementos eléctricos a nivel industrial como, transformadores, motores, variadores de velocidad según la CREG [1] provoca un bajo factor de potencia, debido a la energía almacenada en los devanados de los inductores, también conocida como energía reactiva. Un mal factor de potencia implica que la red debe entregar demasiada energía reactiva, causando perdidas en la red y desgaste en los elementos del sistema Eléctrico. Asimismo en Colombia habrán penalidades basadas en el artículo 25 de la Resolución 108 de 1997 donde se determinó el control al factor de potencia en el servicio de energía eléctrica a los suscriptores o usuarios no residenciales, y de los usuarios residenciales conectados a un voltaje superior al Nivel de Tensión 1(menor a 1kV), el cual debería ser igual o superior a 0.9 inductivo [1].

Mejorar la calidad de corriente consumida por los equipos electrónicos es un tema de gran interés, la mayoría de los dispositivos necesitan una fuente de alimentación que convierta el voltaje de AC de la red en voltaje DC, una forma común de realizar este procedimiento es con el uso de diodos rectificadores, pero está solución produce una entrada de corriente no sinusoidal que disminuye el factor de potencia [2–5] además de generar problemas con interferencia electromagnética causados por una alta distorsión armónica [3].

Con la motivación de cumplir también diferentes estándares eléctricos (IEC 61000-3-2 [6], IEEE 519 [7]) los rectificadores de factor de potencia unitario son la mejor alternativa para sobrellevar estos problemas [8].

El convertidor ac-dc Boost en medio puente o por sus siglas en ingles HBB puede ser utilizado como un rectificador de factor de potencia unitario al controlar su corriente de entrada.Este convertidor tiene tres variables de estado: la corriente en el inductor (iL),el voltaje del primer condensador (Vc1) y el voltaje del segundo (Vc2). Pero esta topología presenta dos principales problemas intrínsecos del sistema. Primero, un desbalance en el voltaje de los condensadores C1 y C2. Segundo,el modelo en variables de estado de HBB no es lineal, al haber una multiplicación de señales, entre iL y la señal de conmutación (PWM). Y otro producto entre las señales Vc y PWM [9].

Los convertidores conmutados, se encuentran conformados por una etapa de potencia de entrada, una etapa de control y una etapa de potencia de salida. La etapa de control, permite procesar la potencia de entrada de manera que se pueda obtener la potencia deseada en la salida. Generalmente en convertidores, la obtención de un controlador lineal a partir de un sistema no lineal se centra en modelos de pequeña señal, linealizando las ecuaciones de estado del convertidor en un punto de operación del sistema. Sin embargo los modelos de pequeña señal son válidos para entornos limitados al punto de funcionamiento del sistema, afectando estabilidad y respuesta temporal al arranque o al producir una perturbación en la carga del convertidor [10].

La propuesta de un nuevo sistema de control que garantice estabilización ha sido un foco de investigación por muchos años, sin embargo satisfacer un conjunto de especificaciones y requerimientos de control es otro problema importante en la industria, siendo abordado por el ajuste o sintonización del controlador desde métodos realizados por Ziegler y Nichols en 1942. Actualmente la sintonización de controladores es un tema crucial en el control inteligente [11, 12].

2.1. Pregunta Problemática

¿Qué mejoras sobre el factor de potencia de un convertidor HBB se obtendrían al utilizar un algoritmo Bioinspirado para sintonizar el controlador del lazo interno de corriente?

Capítulo 3

Antecedentes

3.1. Introducción

El uso de rectificadores no controlados como por ejemplo puentes de diodos con filtro capacitivo es un caso convencional, sin embargo presenta distorsión en la corriente de entrada y bajo factor de potencia ocasionando malas prestaciones en la calidad de energía [2–5, 8, 13].

La estrategia básica para obtener un rectificador PFC, consiste en controlar la corriente de entrada tratando que su forma de onda siga a la tensión de alimentación y de esta manera lograr un factor de potencia unitario [13]. Algunas de las técnicas de control de corriente implementadas en este circuito son 1) control de modo de corriente de frecuencia fija y 2) control de corriente de histéresis de banda fija, ambas técnicas sensan la corriente de entrada para hacer operaciones matemáticas y luego determinar los instantes de conmutación del convertidor [9]. Al ser una planta no lineal, una de las técnicas utilizadas para calcular el control de corriente es linealizar el modelo del convertidor en un punto de operación. Luego se calcula los parámetros de un controlador Lineal con técnicas de diseño clásicas.

Esta sección pretende mostrar desarrollos en Algoritmos Bio-inspirados que se han trabajado en los últimos años, usados para calcular de forma autónoma parámetros de un sistema de control, ya sea en áreas de electrónica de potencia o en otras. Con la implementación de este tipo de desarrollos se logrará una alta eficiencia del circuito PFC Boost en varios puntos de operación.

3.2. Sintonización de controladores

Existen diferentes técnicas de control entre las cuales el control de corriente promediado es una de las más utilizadas en los circuitos PFC. El control del PFC Boost en medio puente es un doble lazo de realimentación. El lazo de control externo regula el voltaje de salida y el lazo de control interno, intenta que la forma de onda de la corriente de entrada siga a la tensión de entrada. Para proveer una regulación deseada se debe ajustar o sintonizar los parámetros del controlador utilizado, ya sea un controlador PI, PID o un controlador con mejores prestaciones. Tradicionalmente estos coeficientes son obtenidos por Zieglers Nichols o mediante diseño en frecuencia haciendo uso de controladores tipo II y tipo III, pero el proceso resulta ser tedioso puesto que conlleva un consumo de tiempo significativo [14].

Para obtener coeficientes de control de manera eficiente se utilizan los métodos basados en algoritmos bioinspirados. Estos pueden ser categorizados de dos formas. La primera son los métodos basados en el descenso de gradiente, estos métodos reducen una función de costo hasta encontrar un mínimo o máximo local. El segundo método usa algoritmos heurísticos. Estos algoritmos inician con una solución inicial

aleatoria y cambian a medida que pasan las iteraciones [14].

3.3. Algoritmos Bioinspirados para la sintonización de controladores

Bastantes algoritmos modernos heurísticos han sido desarrollados para resolver problemas numéricos. Estos algoritmos pueden ser clasificados de distintas maneras de acuerdo a factores basados en población, iteración, estocástica o determinista. Los algoritmos basados en poblaciones trabajan con un conjunto de soluciones e intentan mejorarlas. El algoritmo iterativo utiliza múltiples iteraciones para acercarse a una solución deseada. Los algoritmos estocásticos utilizan una regla probabilística para mejorar una solución [15]. Otra clasificación puede realizarse por la naturaleza del fenómeno simulado por el algoritmo, esta clasificación se divide en dos: Algoritmos Evolutivos EA (Evolutionary algorithms) y algoritmos basados en inteligencia de enjambres (Swarm algorithms). El EA más popular es el algoritmo Genetico GA (Genetic Algorithm), este intenta simular el fenómeno de la evolución natural, donde cada especie busca adaptaciones beneficiosas en un ambiente de constante cambio. Como las especies evolucionan los atributos nuevos son codificados en los cromosomas de cada miembro de la especie. Esta información cambia por mutación aleatoriamente, pero la verdadera fuerza impulsora detrás del desarrollo evolutivo es la combinación y el intercambio del material cromosómico durante la reproducción [15].

Estos principios se han tratado de incorporar en rutinas de optimización desde 1960, El termino enjambre se usa en general para referirse a un conjunto de agentes que interactúan entré si [15].

A continuación se muestran algunos algoritmos ampliamente aprovechados en la sintonización de diferentes controladores como por ejemplo PI, PID, FOPID etc. En algunos de los siguientes trabajos, los sistemas utilizados no son convertidores de potencia, estos trabajos son relevantes ya que los controladores utilizados pueden implementarse en nuestro campo de estudio.

3.3.1. ABC Colonia de Abejas Artificial

El conocido algoritmo Bio-inspirado de enjambres ha sido desarrollado modelando lógicamente el comportamiento coordinado del enjambre de abejas meliferas [16].

El controlador type-1 fuzzy logic es sintonizado mediante la adopción por colonia de abejas BCO (Bee Colony). Mostrando resultados estadísticos que nos indican que está adaptación mejora la explotación y la exploración del mecanismo de búsqueda y luego proporciona un controlador sobresaliente en el controlador difuso. [17]

En [18], Zafer Bingul y Oguzhan Karahan realizan una comparación del algoritmo colonia de abejas artificial (ABC) y el algoritmo enjambre de partículas (PSO) sintonizando controladores PID y FOPID en sistemas con retardo. Los autores utilizan tres funciones de costo para encontrar los parámetros de los controladores. Encontraron que el resultado obtenido con el algoritmo ABC presenta mayor robustez a perturbaciones internas y externas, la solución es encontrada con menos iteraciones pero sus cálculos tienen mayor complejidad. Por su parte el controlador con el algoritmo PSO se halla en un menor tiempo de computo, hay una rápida convergencia. Se puede decir que el algoritmo ABC obtuvo mejores resultados pero su gasto computacional es más alto.

En el año 2012 proponen la unión de dos algoritmos de optimización, Algoritmo de selección clonal y algoritmo de enjambre de abejas artificial (ABC). Este algoritmo es utilizados para encontrar los parámetros de un controlador PID fuzzy. Este algoritmo llamado HCABC demostró que alcanzó los mejores

parámetros del controlador PID para una planta no lineal. El principio de selección clonal es basado en los autores L. Castro y J. Von Zuben. donde especifican un cambio en el proceso de mutación. Se utiliza un parámetro "ρ" para controlar el decaimiento del factor exponencial de afinidad. En el algoritmo el numero de clones no lo dejan proporcional a la afinidad presentada por el conjunto de anticuerpos en memoria, sino que es una constante, y utilizan una codificación binaria para la actualización de los nuevos anticuerpos en el vector de memoria. El algoritmo no solamente logró la optimización de las variables del controlador, sino que obtuvo una velocidad de convergencia mucho mayor que los algoritmos ABC y de selección clonal [19].

3.3.2. ACO Colonia de Hormigas

El algoritmo de la colonia de hormigas inspirado en el comportamiento natural de las hormigas, es adaptativo y se puede aplicar a varios problemas [20, 21].

La idea principal de ACO es modelar el problema como la búsqueda de una ruta mínima de costo en un gráfico: Las hormigas artificiales caminan por este gráfico, buscando buenos caminos. Cada hormiga tiene un comportamiento bastante simple de modo que solo encontrará caminos no eficientes por si solas. Se encuentran mejores caminos como resultado de la cooperación entre las hormigas de la colonia [21].

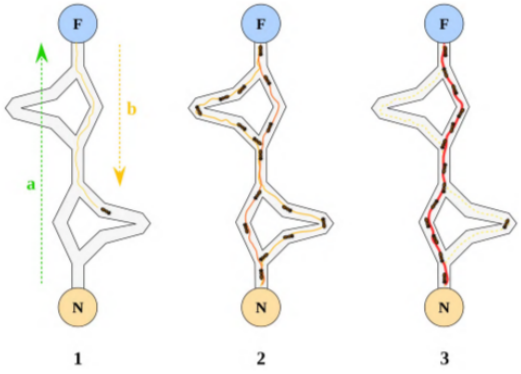

Figura 3.1: Algoritmo de colonia de hormigas [20]

La mejor solución del problema se determina cuando la colonia de hormigas se comunica entre sí como se muestra en la figura 3.1 (La comunicación entre hormigas se realiza indirectamente agregando feromonas en su medio ambiente). El camino más corto al destino lo determina una hormiga cuando comienza desde un estado inicial y lo hace a lo largo de la secuencia de estados vecinos. La regla probabilística local controla el movimiento de las hormigas que se rige por la información ambiental local, los rastros de las feromonas y los estados internos [20].

El algoritmo ACO es utilizado para el diseño y control de un sistema de levitación usando control de orden fraccional PID o FOPID en el año 2020 por Abdullah Mughees y Syed Mohsin [20]. Se muestran un recuento de trabajos ya realizados utilizando diferentes algoritmos Bioinspirados en controladores FOPID. Se obtuvo una función de transferencia del sistema de levitación de orden 24, es aplicado el criterio de estabilidad de ROUTH HURWITZ sobre la función. Inicialmente el controlador FOPID es sintonizado por medio de Zieglers Nichols asumiendo una función de transferencia aproximada. El controlador FOPID tiene los tres parámetros convencionales de un PID y además otros dos extra ("lambda", ü"). Luego fue

implementado el algoritmo ACO, y se comparan las respuestas mediante simulaciones en MATLAB®. Se encontraron resultados brillantes al usar el algoritmo ACO, la eficiencia del tiempo de establecimiento incremento en un 95.99 % en comparación a la sintonización convencional por medio de Zieglers NIchols. Los autores concluyeron que ACO mejoró la eficiencia y estabilidad del sistema.

3.3.3. Algoritmo basado en la selección clonal

En el año 2002 proponen un algoritmo computacional basado en el principio de selección clonal. Los autores realizan una comparación del algoritmo propuesto con algoritmos evolutivos. Analizan cómo se puede usar para trabajos de optimización, categorización, reconocimiento de patrones y muestran sus respectivos diagramas de flujo. Utilizan varios problemas para evaluar su rendimiento y enfatizan su utilidad en los problemas de optimización multimodal. Realizan pruebas con funciones de dos y tres dimensiones, donde el algoritmo muestra que es capaz de encontrar los máximos locales y a su vez un máximo global. En las pruebas de reconocimiento de patrones, había una población explicita inicial para empezar el algoritmo. El algoritmo se adaptó para los problemas de optimización y logró alcanzar un conjunto diverso de soluciones óptimas locales, mientras que el algoritmo evolutivo tiende a sesgar a toda la población de individuos hacia la mejor solución candidata [22].

Un motor eléctrico es una estructura compleja y con ello encontrar un modelo matemático preciso. Uno de los métodos tradicionales para el control de voltaje del motor es el controlador PID, pero presenta limitaciones, es por esta razón que Qinghua Meng y Tingting Liu proponen la sintonización del control PID mediante un algoritmo basado en el sistema inmune artificial, en el cual la función objetivo se tienen en cuenta los índices de desempeño del controlador (Tiempo de levantamiento, tiempo de establecimiento, sobrepaso y error de estado estacionario) y las constantes del controlador (K_i, K_p, K_d) son los anticuerpos. EL valor inicial para cada uno de los anticuerpos es aleatorio dentro de un rango determinado y por medio de procesos de clonación, cruce y mutación se encuentran los valores óptimos para el control. El esquema de control propuesto se encuentra en la figura 3.2.

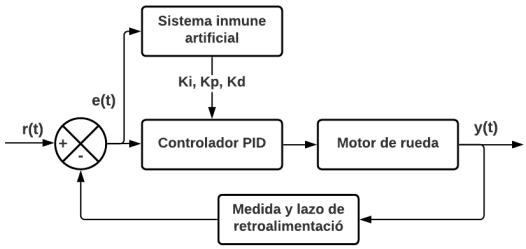

Figura 3.2: Diagrama de flujo del algoritmo PID inmune [23].

Los resultados son presentados en la figura 3.3 donde se evidencia disminución en tiempo de levantamiento, sobrepaso máximo y tiempo de establecimiento [23].

3.4. Algoritmos Bioinspirados para sintonización de controladores en convertidores

3.4.1. ANN Redes neuronales artificiales

Uno de los trabajos que presentan el uso de algoritmos Bioinspirados para convertidores conmutados es el de G. Tsakyridis, N. Xiros, M. Scharringhausen, L. Witte, ellos realizan este desarrollo como solución

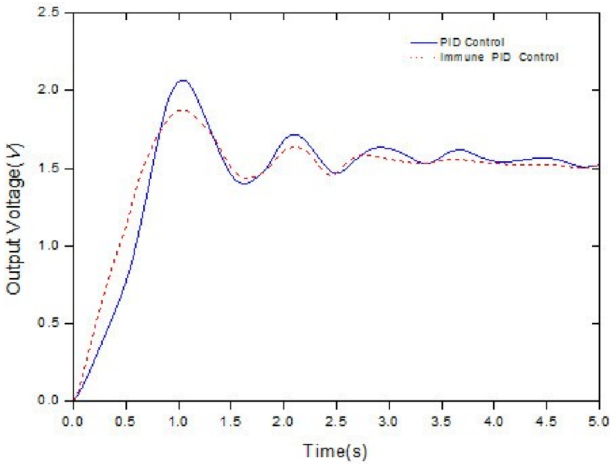

Figura 3.3: Comparación entre el control PID y el control PID inmune [23].

para la generación de energía en impulsores marítimos, estos son alimentados por pilas de combustible y necesitan mantener un voltaje de salida estable en un convertidor Boost que presenta una naturaleza no lineal el cual presenta cambios en su estado transitorio cuando se cambia el punto de operación de ahí el uso de redes neuronales artificiales (ANN) para controlar la respuesta. La red que se utilizó fue una "Back propagation" con tres capas, la capa de entrada con 3 neuronas, la capa intermedia con 5 neuronas y la capa de salida con una neurona, el entrenamiento de la misma se realizó a partir de 512 datos de voltaje de entrada, voltaje de salida, carga externa y ciclo útil. Lo anterior se realiza mediante el uso de la herramienta de redes neuronales de MATLAB®. Al comparar los resultados con los obtenidos mediante los controladores convencionales se encontró una mejora en tiempo de establecimiento y sobrepaso máximo en la salida de voltaje [24].

J. Shaik y V. Ganesh presentan el uso de una red neuronal artificial sintonizada por el método luciérnaga Levenberg-Marquardt (ANN-FLM) para el arranque suave o el restablecimiento de la energía después de un apagón del sistema de un convertidor de fuente de voltaje (VSC): corriente continua de alto voltaje (HVDC). Dicho algoritmo sigue 3 reglas principales basadas en el comportamiento de las luciérnagas, (i) las funciones objetivo evalúan el brillo de las luciérnagas, (iii) El brillo de las luciérnagas es proporcional a la distancia entre sí, (iii) las luciérnagas son unisex, por ende, se atraen entre sí comúnmente. Con la posterior implementación y puesta a prueba del sistema se concluye que es una solución económicamente viable para la recuperación del sistema eléctrico ya que reduce las sobretensiones en el proceso de restauración y se realiza con un menor tiempo de respuesta, lo anterior se sustenta a partir de la comparación con un controlador PI convencional [25].

Los autores J.P. Romero, D. Paez, B. Noriega, J. Guarnizo y J. Bayona realizan otra publicación sobre la sintonización de un controlador para convertidores eléctricos [26] en este trabajo el objetivo es sintonizar el controlador de voltaje para un convertidor DC-DC Forward el cual es una topología derivada de un Buck pero con un transformador adicional para la salida de voltaje. La sintonización del controlador PID se realiza mediante el uso de redes neuronales artificiales, esta red neuronal consta de 3 capas, la de entrada con las variables de voltaje de referencia, voltaje de salida, corriente en el inductor, error entre lo esperado y el resultado y el ciclo útil, siguiente una capa oculta y en la capa de salida el ciclo útil, como resultado se tiene un voltaje que se estabiliza a variaciones de carga.

En otro documento los autores realizan un control de corriente para un convertidor de voltaje Buck. Para ello utilizan un control neuronal general inverso. También utilizan un controlador de módulos neuronales en paralelo con el fin de mejorar el rendimiento del control. Luego de realizar la simulación de la

red neuronal junto con el circuito, lo autores concluyen que este tipo de controlador no es adecuado para el control de corriente en convertidores Buck. También comentan que al aumentar el número de neuronas en la capa oculta no mejoran los resultados. Ante esto entrenan los controladores neuronales en paralelo en diferentes puntos de operación y se activa el correspondiente controlador dependiendo de la corriente y el voltaje [27].

3.4.2. ACO Colonia de Hormigas

Suri Babu y K. Dinesh en el año 2016 [21] sintonizan un controlador FOPID en un sistema regulador de voltaje automático AVR ("Automatic Voltage Regulator"). El controlador FOPID es modelado utilizando algunas aproximaciones, y se obtiene una función de transferencia de cuarto orden. El diseño del controlador es obtenido y dos algoritmos Bioinspirados ("Genetic Algorithm y Ant Colony Algorithm") son implementados. Son enseñados los parámetros utilizados para inicializar cada algoritmo y posteriormente simular en MATLAB®. Las gráficas presentadas en los documentos muestran que ACO obtuvo el mejor tiempo de respuesta y el algoritmo genético logró el menor sobrepico. Dentro de las conclusiones del trabajo se puede mencionar que los algoritmos realizaron una búsqueda de los parámetros en el controlador FOPID para el sistema AVR practico. Además el controlador FOPID propuesto tiene características de estabilidad y rendimiento más robustas que el controlador PID aplicado al sistema AVR.

3.4.3. Algoritmo de Enjambre

Enjambre de particulas es un algoritmo evolutivo relativamente nuevo que se puede utilizar para encontrar soluciones con buen desempeño en problemas numéricos y cualitativos. Fue propuesto en el año 1995 por Eberhart y Kennedy y está inspirado en el comportamiento de bandadas de pájaros y cardumenes de peces. Este proceso obtuvo bastante atención debido a su calculo simple y características de convergencia rápida [28].

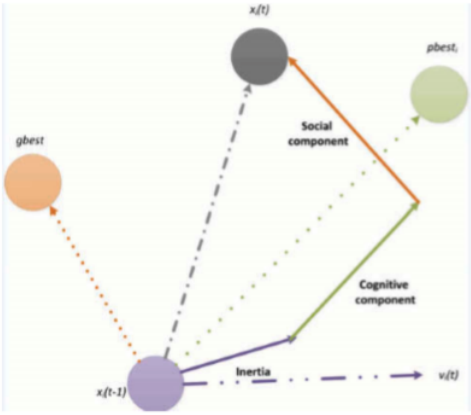

Figura 3.4: Proceso de Enjambre [28]

La trayectoria final de una partícula se presenta en la figura 3.4. En cada iteración, las partículas son atraídas hacia la mejor solución. Y el proceso termina cuando el error de entrenamiento es menor a un valor predeterminado o cuando se cumple un numero especifico de iteraciones [28].

En el año 2021 Jesus Aguila, Cristian Chiñas, Carlos Vargas, Elias Hurtado y Edith Garcia [29] en el Journal "Advances in Science, Technology and Engineering Systems Journal", tres algoritmos Bioinspirados fueron aprovechados para obtener los parámetros de un controlador PID en un convertidor DC-DC

Boost. Primero se obtiene la representación del sistema en variables de estado. Los parámetros mediante los algoritmos son encontrados. Para esto es usado un vector X que contiene cada una de las constantes en el controlador PID (Kp,Ki,Kd). Para minimizar la función de error e(t) fue utilizando RMES (Root Mean Square Error"), y es definida como la función objetivo del algoritmo. Es presentado el diagrama de flujo que encuentra los mejores valores de las constantes. Se expone una tabla con los parámetros de inicialización en cada algoritmo. Las curvas de convergencia son comparadas, para luego implementar los sistemas con los parámetros encontrados. Luego en la simulación del convertidor con su respectivo controlador calculado, se añaden perturbaciones para observar el rendimiento de cada control dependiendo del algoritmo. Por ultimo al realizar la comparación de las respuestas, se observa que los tres algoritmos obtuvieron resultados satisfactorios.

3.4.4. ABC Colonia de Abejas Artificial

En octubre del 2021 utilizan un convertidor Buck DC-DC y regulan su voltaje de salida mediante un controlador PID. Sintonizan el controlador haciendo uso de los algoritmos de optimización enjambre de partículas y colonia de abejas. Los resultados mostraron que los dos algoritmos lograron encontrar parámetros que regulan el voltaje de salida del convertidor Buck sin embargo el PID del algoritmo de colonia de abejas mostró una respuesta más rápida con menor error de estado estacionario [30].

3.4.5. AG Algoritmo genético

En el año 2021 los autores utilizaron un convertidor Buck-Boost controlando el voltaje de salida mediante controladores PI y PID. Utilizaron los algoritmos de Lobos grises, algoritmo Genetico, algoritmo de enjambre de partículas, Optmización Ant-Lion, Algoritmo de Optimización Ballena para sintonizar los controladores PI y PID. Utilizaron dos funciones en la optimización: Error integral cuadrático y error integral absoluto. El algoritmo que mejor desempeño obtuvo fue el algoritmo genético y demostraron que es la mejor técnica de optimización para un PID en convertidores Buck-Boost, mostrando un valor significativamente más bajo del tiempo de amortiguación de oscilaciones [31].

3.4.6. GWO Algoritmo Lobos Grises

El algoritmo bio-inspirado Lobo Gris (GWO) por sus siglas en ingles "Gray wolf optimizationïntroducido por Mirjalili en el año 2014 [32] imita la jerarquía de liderazgo y mecanismo de caza de los lobos en la naturaleza [33].

En [29], en el año 2021 se realiza una comparación entre el rendimiento de métodos de optimización Particle Swarm, Genetic algorithm y Grey Wolf para sintonizar el controlador PID de un convertidor Dc-Dc Boost. Este convertidor es modelado por ecuaciones de espacio de estados, y el proceso relacionado con el controlador PID es realizada utilizando SIMULINK® de MATLAB®. El rendimiento del algoritmo es evaluado usando el error cuadrático medio. Los resultados muestran que el algoritmo GWO es una solución factible para el problema de sintonización del controlador para convertidores de potencia ya que su desempeño general es mejor que el obtenido por PSO y GA.

El controlador sintonizado GWO-PID tuvo el mejor rendimiento con un RMSE aproximadamente 50,89 por ciento más bajo que PSO y 44,70 por ciento más bajo que GA. Las diferencias entre el RMSE obtenido utilizando los tres algoritmos da una idea de la mayor susceptibilidad de PSO y GA a quedar atrapados en soluciones óptimas locales, mientras que el valor de RMSE más bajo obtenido utilizando el algoritmo GWO indica que este algoritmo elude de mejor forma un mayor número de soluciones óptimas locales y así encontrar una solución superior que los algoritmos PSO y GA para esta aplicación.

3.4.7. PSO Algoritmo de enjambre de partículas

También en el año 2021 Utilizan algoritmos de optimización para encontrar los mejores parámetros de un controlador adaptativo gausiano en un convertidor Buck DC-DC. Este controlador es un problema multimodal donde varias soluciones distintas pueden lograr rendimientos similares y los algoritmos pueden comportarse de manera diferente durante el proceso en muchas ocasiones no se tiene una solución algebraica. Utilizan el algoritmo enjambre de partículas, algoritmo Colonia de abejas y el algoritmo de optimización ballena. El algoritmo presentó mejores resultados fue PSO con una convergencia mucho más rápida [34].

En el articulo presentado por J.P. Romero, D. Paez, B. Noriega, J. Guarnizo y J. Bayona [35] realizan la sintonización de un controlador PID en el lazo de corriente de un controlador Boost para la corrección de factor de potencia, el algoritmo Bioinspirado usado fue el Particle Swarm Optimization (PSO) y posteriormente realizan ajustes para tener un Multi-PSO con el fin de realizar búsquedas múltiples y lograr una tasa de éxito mayor. El mínimo de error lo encuentran en el enjambre numero 7 y con este un factor de potencia de 0.996 a 100w, mejorando el desempeño del controlador existente el cual daba 0.991, estos resultados mostrados son basados en simulaciones por medio del software SIMULINK® de MATLAB ®.

3.4.8. AG Algoritmo genético

Un mecanismo de búsqueda basado en el principio de selección natural el cual se conoce como algoritmo genético (AG) fue utilizado para eliminar ordenes de armónicos de la forma de onda de un convertidor AC/AC por medio de la modulación por ancho de pulso (PWM), se propone un algoritmo híbrido con la unión del AG y un algoritmo de búsqueda convencional, esto con el fin de contrarrestar la dificultad que presenta el GA que ocupa la mayor parte del tiempo compitiendo entre colinas y no en la búsqueda optima en una de ellas. En la primera fase se hace la implementación del AG donde se inicializa un vector con una distribución uniforme entre 0 y 1, posteriormente por medio de operadores aritméticos de cruce y mutación se genera una nueva descendencia hasta lograr que las mejores soluciones no cambien entre dos intervalos de generación, la segunda fase utiliza la optimización de búsqueda directa partiendo del vector solución obtenido en la fase uno. Con el fin de demostrar le eficacia del método se realizan pruebas con diferentes puntos operativos y como resultado se encuentra que el método realizado es superior a las técnicas usadas convencionalmente dado en que algunos casos pueden fallar en la convergencia [36].

3.4.9. Algoritmo basado en el comportamiento de los saltamontes

Las fuentes de energía renovable presentan un inconveniente en su generación debido a su naturaleza intermitente, en el documento de K. Aseem y S. Kumar se propone un sistema de adquisición de energía a partir de un generador eólico y un panel solar, posteriormente un seguimiento del punto de máxima de potencia (MPPT), en continuación se coloca un convertido Buck- Boost multi entrada, el tipo de controlador a utilizar en este convertidor es proporcional integral derivativo de orden fraccional (FOPID) sintonizado mediante un algoritmo de optimización basado en el comportamiento de los saltamontes (GOA) utilizando en una primera fase un algoritmo k-medias con el fin de eliminar la posibilidad de que la solución de quede en un óptimo local (eliminan el comienzo aleatorio). El algoritmo de optimización se realiza mediante la fuerza de repulsión, la fuerza de atracción y la zona de confort, cuando los saltamontes se encuentran en un rango de distancias de 0 a 2.079 se disuaden evadiendo la colisión, en el rango de 2.079 a 4 la fuerza de cohesión aumenta para mantener el enjambre y cuando se encuentra a 2.079 se denomina zona de confort en la cual no existe ni atracción ni repulsión. EL sistema se implementa y se valida con los rendimientos obtenidos mediante otros algoritmos de optimización y se encuentra una eficiencia del 98.5 % un valor

superior al que se alcanzó con los demás sistemas [37].

3.4.10. Algoritmo Bacterial Foraging

En [18] los autores sintonizan el controlador de un convertidor Buck-Boost usando de optimización Bacterial Foraging. En este trabajo se establecen limites para las constantes kp, ki y kd de un controlador PID. Minimizan una función de costo en 50 iteraciones. Simulan diferentes perturbaciones y encuentran que el control hallado es robusto en el sentido que rechaza las perturbaciones internas y externas, además provee una excelente respuesta dinámica en todos los puntos de operación. Con bajos sobre picos, y tiempos de levantamiento rápidos.

3.5. Algoritmos Bioinspirados para sintonización de controladores en convertidores con corrección de factor de potencia

3.5.1. ANN Redes neuronales artificiales

La corrección del factor de potencia también ha sido objeto de estudio, es el caso del articulo presentado por A. Benyamina, S. Moulahoum, I. Colak y R. Bayindir, dichos autores presentan un diseño de corrección de factor de potencia en un convertidor Boost usando para la sintonización del control por medio de la combinación de lógica difusa con redes neuronales artificiales (control difuso neuronal), la estructura general del control se dividen en 4 pasos, (i) Entrenar la red a partir de los datos, (ii) Dichos datos de entrenamiento se transforman en un conjunto difuso por medio de una función miembro y a partir del mismo se crea el condicional “if then else” de acuerdo a la entrada difusa ponderada, (iii) por medio del uso de un algoritmo de entrenamiento basado en minimización de error se actualizan los parámetros, (iv) Defusificación: conversión de conjunto de datos difusos en números y se guarda el modelo dado. El sistema fue colocado a prueba con variación de carga de 40 a 60 Ω consiguiendo un factor de potencia de 0.998 [38].

3.5.2. ABC Colonia de Abejas Artificial

En el año 2016 es sintonizado un controlador PID para un PFC utilizando BC ("Bee colony") [39]. Realizaron las validaciones del controlador obtenido mediante SIMULINK® de MATLAB®, donde se obtuvo un menor tiempo de establecimiento al usar el algoritmo BC. En este articulo se comprobó que el controlador BC-PID tuvo mejor desempeño en comparación al controlador PSO-PID para el convertidor AC-DC utilizado.

3.5.3. SGA Algoritmo genético simple

En el año 2019 Basándose en el algoritmo SGA es sintonizado el controlador para un Rectificador trifásico activo [40]. El objetivo de utilizar este algoritmo es encontrar una solución óptima global. El modelado del convertidor es realizado por medio del análisis "DQ reference frame". Se exponen tres problemas en particular de un rectificador PFC HBB. Son presentadas tres funciones de ajuste, ”fitness functions” comúnmente usadas en la teoría de control, integral de error absoluto IAE (” Integral of absolute error”) e integral de error cuadrado ISE (” Integral of square error”). Estas funciones dependen de valores iniciales, afectando características de estado estable. El factor de énfasis "x"mejora la respuesta dinámica del sistema.La relación de los parámetros del algoritmo genético es explicada. Las simulaciones son realizadas mediante el software MATLAB ®. Obtienen varios valores del constantes en un controlador PI, variando el factor de énfasis "x". Por ultimo concluyen que el factor de énfasis "x"para evaluar y optimizar los

parámetros de PI proporcionó más flexibilidad y grado de libertad en la función objetivo en comparación con la función convencional [40].

3.5.4. DE Algoritmo diferencial evolutivo

Tres métodos heurísticos son utilizados(Genetic algorithm, Swarm algorithm y diferenttial evolutionary) en el año 2017 por Tulay, İskender y Erdem [14], para la sintonización de un controlador PI en un rectificador Boost de topología clásica. Una función de costo es planteada en función del Valor de Distorsión armónica total THD "Total harmonic Distortion", corriente de entrada y errores de voltaje de salida. Luego es minimizada utilizando los métodos heurísticos mencionados. Dentro de la función de costo además se utiliza la media del error absoluto MAE ("Mean Absolute Error"). MAE fue calculado con el comando de MATLAB® "mae". Con el método diferential evolutionary (DE) se encuentran resultados satisfactorios obteniendo los coeficientes del PI y valores de Voltaje de salida Vout = 600 V, Factor de Potencia PF = 1, y THD = 0.02326. Son presentados los limites de las constantes del controlador(KP,KI) que presentan mejores resultados para cada uno de los algoritmos utilizados. Luego de emplear los algoritmos se muestra la gráfica de su función de costo en cada iteración, en una sola imagen, donde se observa la mejor respuesta, obtenida por el algoritmo "DE". Se evalúan los parámetros encontrados por cada algoritmo por medio de simulaciones en SIMULINK® de MATLAB®, donde se encuentra menor tiempo de estabilización con "DE". A modo de conclusión los autores mencionan que todos los algoritmos utilizados son aceptables, pero "DE"presenta rendimiento superior, asegurando el factor de potencia más alto y THD mínimo. Dentro de las conclusiones del trabajo se puede mencionar que estos métodos heurísticos son utilizados fácilmente a cualquier tipo de control de doble lazo.

Capítulo 4

Justificación

Un corrector de factor de potencia ideal PFC debería emular una resistencia en el lado de la fuente mientras entrega un voltaje de salida suficientemente regulado [3]. Tener un factor de potencia cercano a la unidad quiere decir que las perdidas debidas a la potencia reactiva se reducirán. Por otro lado estos circuitos PFC deben ser confiables debido a que estarán conectados a diferentes dispositivos electrónicos, tanto industriales y no industriales, ejemplo electrodomésticos, y una falla en la fuente de alimentación (PFC) puede repercutir también en estos dispositivos conectados.

Al cambiar las típicas fuentes rectificadoras de puentes de diodos y utilizar fuentes de alimentación basadas en circuitos PFC, ya no se inyectarán armónicos a la red eléctrica publica. Esta es una ventaja para las empresas prestadoras del servicio eléctrico, debido a que reducirán gastos en sobredimensionamiento de los elementos eléctricos (conductores, transformadores,etc), menor riesgo a sobrecalentamiento de los transformadores y conductores, menor riesgo a degradación del aislamiento de los transformadores. Por otro el hecho de tener un circuito con un factor de potencia unitario hace que la potencia media sea igual a la potencia aparente, esto significa que el circuito no consuma más potencia aparente de la que realmente necesita.

Al conectar una carga en el circuito, es decir al cambiar el punto de operación, el PFC se puede volver inestable o generar altos niveles corriente por un tiempo muy pequeño, ocasionando daños permanentes en los elementos semiconductores presentes en el convertidor. El controlador PID u otro diferente es el encargado de sobrellevar estos cambios, y reducir picos de corriente, en un punto de operación especifico.

Entre las técnicas de inteligencia artificial los algoritmos evolutivos han sido ampliamente usados en los métodos de sintonización que requieren para encontrar los parámetros del PID más adecuados. Su ventaja es que encuentran soluciones cercanas a la global en espacios no convexos y discontinuos y no necesitan características específicas del problema [11].

Usualmente en convertidores conmutados al diseñar un PID con técnicas clásicas como lugar geométrico de las raíces, Bode, cancelación de polos entre otros, Sus constantes son obtenidas a partir de la función de transferencia del sistema calculada en un punto de operación, por ende los parámetros de diseño como el factor de potencia en este caso, serán cumplidos correctamente, solamente en dicho punto.

Al implementar un algoritmo bio-inspirado para la sintonización automática del PID digital en el convertidor, se tendrá un PFC con un factor de potencia cercano a la unidad en cualquier punto de operación. El sistema tendrá la capacidad de calcular automáticamente las constantes del controlador en cuanto la carga o punto de operación sea cambiada.

Actualmente existen diferentes configuraciones de convertidores conmutados Boost que son utilizados

como PFC,pero es un hecho que al disminuir el número de dispositivos se reduce la caída total de voltaje de conducción del dispositivo semiconductor, provocando una mejora en la eficiencia general del sistema [41]. Otra topología ampliamente usada, PFC Boost en medio puente o HBB [41], presenta solo un diodo en conducción en cualquier tiempo de conducción, por tal motivo, se escoge HBB en el presente documento para la evaluación del mismo.

Capítulo 5

Impacto social

La mayor parte de los procesos en la industria necesitan algún tipo de movimiento generado por motores eléctricos. Hay disponibles motores eléctricos de 1 hp a 100,000 hp para satisfacer cualquier necesidad industrial. Se encuentran disponibles motores y generadores asíncronos y síncronos para adaptarse a funciones especializadas. Los ejemplos de aplicaciones incluyen ventiladores industriales, sopladores, bombas, máquinas herramienta, herramientas eléctricas, turbinas, compresores, alternadores, barcos, laminadores, fábricas de papel, motores y otras aplicaciones especiales. Aquí se haya la principal aplicación donde se requiere mejorar el factor de potencia que reducirá gastos innecesarios en la factura de energía en la industria. Estos motores son utilizados desde pequeñas hasta grandes empresas.

Al mejorar las fuentes de alimentación con circuitos PFC, la empresa prestadora del servicio de red eléctrica se verá beneficiada debido a las siguientes ventajas: Primero, aumento en el factor de potencia, que implica mayor potencia disponible de la linea de AC. Las facturas eléctricas están basadas en la potencia media, pero las potencia que puede entregar un transformador es la potencia aparente.Al haber un factor de potencia unitario implica que la empresa entregue la misma potencia que se cobra en la factura eléctrica. Segundo, disminución de inyección de armónicos a la tensión de linea que interfieren con otros equipos electrónicos conectados a la misma linea y finalmente mayor estabilidad en la tensión de linea.Se reducirá el riesgo de sobrecalentamiento de transformadores y conductores debido a armónicos.

El otro gran beneficiado del desarrollo de este proyecto es el usuario de la red eléctrica. Ya sea en la industria o no, el usuario tendrá la confianza de conectar sus equipos electrónicos a la red y saber que no presentarán mal funcionamiento o algún daño debido a armónicos, incluyendo electrónicos susceptibles a la interferencia electromagnética (EMI).

Capítulo 6

Objetivos

6.1. Objetivo General

Incrementar el Factor de Potencia en el Half Bridge Boost a un valor mayor a 0.992, cuando él opere a diferentes niveles de carga, mediante el uso de técnicas Bio-inspiradas para la sintonización offline de un controlador PID y filtro pre-compensador.

6.2. Objetivos específicos

- Realizar una revisión del estado del arte sobre algoritmos Bioinspirados para la sintonización de controladores, enfocados a rectificadores con corrección de factor de potencia.
- Seleccionar un algoritmo Bio-inspirado con el propósito de sintonizar los parámetros de un controlador PID en el lazo interno de corriente, así como las constantes de un filtro pre-compensador utilizando un modelo de circuito promedio para incrementar el factor de potencia del Half Bridge Boost.
- Verificar en simulación el algoritmo de sintonización junto al Half Bridge Boost con el fin de realizar pruebas preliminares y ajustes necesarios.
- Implementar las constantes de los controladores obtenidos previamente en un prototipo del Half Bridge Boost construido en laboratorio, para realizar validación.

Capítulo 7

Marco teórico

7.1. Convertidor PFC HBB

Las gran mayoría de sistemas de distribución funcionan con voltaje AC, por lo que es necesario el uso de convertidores de potencia que transforman el voltaje a DC y de está forma ser consumido por aparatos electrónicos. Esto se logra mediante convertidores electrónicos y dispositivos con semiconductores (Diodos, Mosfets, IGBts) en conjunto con diodos [42]. Una de las topologías más utilizadas es la topología Half Bridge Boost o elevador en medio puente (Figura 7.1) debido a que está topología tiene solamente un semiconductor activo por tiempo de conmutación, reduciendo perdidas de potencia. Sin embargo requiere de un condensador con una capacitancia alta para obtener valores de rizado bajos.

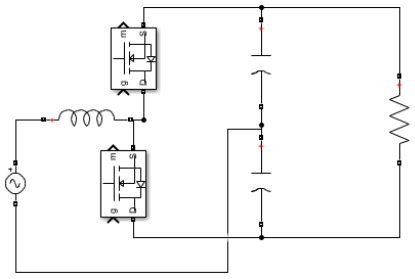

Figura 7.1: Esquema Circuito PFC HBB [43].

Existen otras topologías, que son utilizadas para este tipo de rectificadores. Pero tienen mayor cantidad de semiconductores activos en sus estados de conmutación. Algunas de estas topologías se encuentran en la figura 7.2.

7.1.1. Modelo del convertidor HBB

Para obtener el modelo del convertidor, se plantean las ecuaciones de estado en cada tiempo de conmutación.
$t1$:

$$vL = Vg + v1 \tag{7.1}$$

$$ic1 = -iL - \frac{v1}{R} - \frac{v2}{R} \tag{7.2}$$

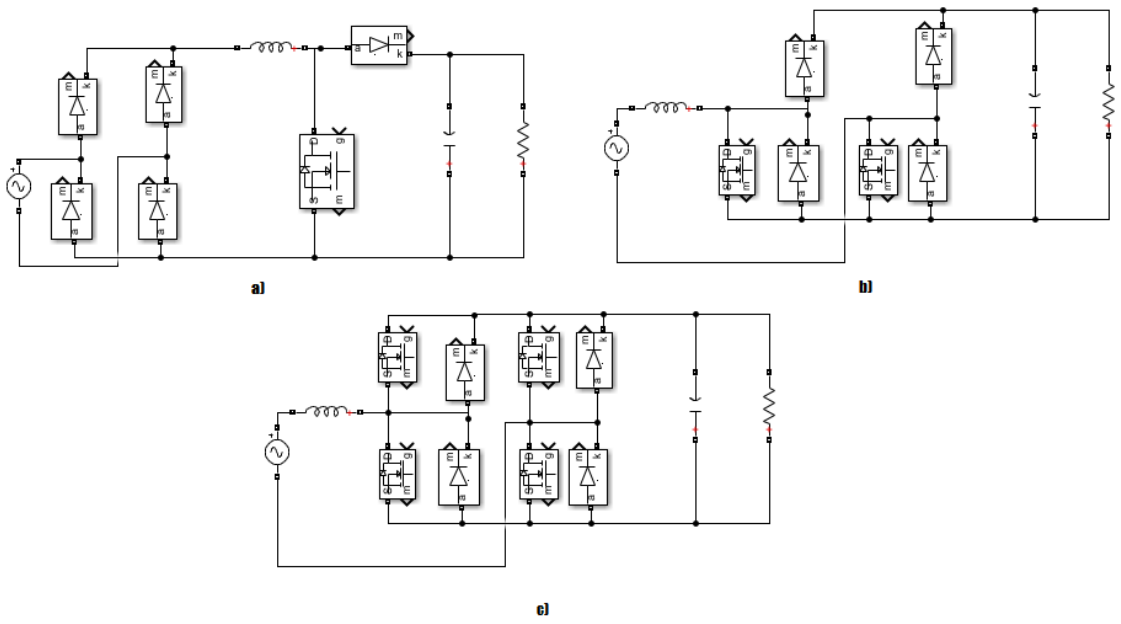

Figura 7.2: Numero de semiconductores activos por tiempo en las topologías Boost a) 3 b) y c) 2 [41].

$$ic2 = -\frac{v1}{R} - \frac{v2}{R} \tag{7.3}$$

$t2$:

$$vL = Vg - v2 \tag{7.4}$$

$$ic1 = -\frac{v1}{R} - \frac{v2}{R} \tag{7.5}$$

$$ic2 = iL - \frac{v1}{R} - \frac{v2}{R} \tag{7.6}$$

Se promedian las ecuaciones de estado y obtenemos:

$$\frac{Ld_{iL}}{dt} = Vg + dv1 - (1-d)v2 \tag{7.7}$$

$$\frac{C1d_{V1}}{dt} = -diL - \frac{V1}{R} - \frac{V2}{R} \tag{7.8}$$

$$\frac{C2d_{V2}}{dt} = (1-d)iL - \frac{V1}{R} - \frac{V2}{R} \tag{7.9}$$

Para simplificar el sistema, se hace un cambio de variable de v1 y v2 a vs y vd. Que representan el voltaje de suma y de diferencia entre los capacitores.

$$\frac{Ld_{iL}}{dt} = Vs\frac{2d-1}{2} + Vd\frac{1}{2} + Vg \tag{7.10}$$

$$\frac{Cd_{Vs}}{dt} = -(2d-1)iL - \frac{2Vs}{R} \tag{7.11}$$

$$\frac{Cd_{Vd}}{dt} = -iL \tag{7.12}$$

Componente/Variable	Valor
Fs	50 kHz
C1	100 uF
C2	100 uF
L	5 mH
R	2 kohms
Vg	120 Vp
Ip	1 A
Vs	450 Vdc
Vd	0 Vdc
d	0.5 Vdc

Cuadro 7.1: Valores de los componentes.

Al linealizar en el punto de operación con los valores del cuadro 7.1, se obtiene la función de transferencia continua (7.13) y en tiempo discreto con una frecuencia de muestreo de $20\mu s$ (7.14).

$$Gid = \frac{90000s^2 + 90000s}{s^3 + 220s^2 + 1{,}002e6 + 1e7} \tag{7.13}$$

$$Gidd = \frac{1{,}796z^2 - 3{,}52z + 1{,}796}{z^3 - 2{,}995^2 + 2{,}991z - 0{,}9956} \tag{7.14}$$

7.2. Factor de potencia y sus componentes

En las lineas de energía eléctrica coexisten 3 tipos diferentes de potencia y varios indicadores de eficiencia o rendimiento como lo es el factor de potencia.

- **Potencia activa:**
  Es representada por la letra P y se refiere a la capacidad de transformar energía en trabajo útil lo cual hace alusión a la potencia que es consumida. Su unidad de medida son los vatios (W) y su consumo a través del tiempo es la energía activa (Wh)

- **Potencia reactiva:**

  Su unidad son los voltiamperios reactivos (VAR) y está representada por la letra Q, a diferencia de la anterior esta no produce trabajo útil y es generada por elementos inductivos y capacitivos para la creación de campos magnéticos y eléctricos en los componentes mencionados.

- **Potencia aparente:**

  Representa la suma vectorial entre las dos potencias anteriormente mencionadas, su unidad son los voltiamperios y su representación es la letra S.

$$S = \sqrt{P^2 + R^2} \tag{7.15}$$

A partir de los anteriores conceptos se puede definir el factor de potencia (FP) como el indicador de eficiencia con la que un componente eléctrico absorbe potencia real, esta medida es adimencional e idealmente se busca un valor de 1, lo que conlleva a un valor de potencia reactiva igual a 0. [44]

$$FP = \frac{P}{S} \tag{7.16}$$

7.3. Control PID

Las tres funcionalidades del control PID ofrece la más simple y eficiente solución a muchos problemas de control del mundo real. Desde su invención en el año 1910, para el desarrollo del piloto automático de la nave Elmer Sperry y los métodos de sintonización por Ziegler Nichols en 1942 la popularidad del PID ha crecido de forma exponencial. [43]

El controlador PID puede ser considerado como una variación de una red de adelanto-atraso con un polo en el origen. A continuación se muestra la forma general de un controlador PID en tiempo continuo.Donde K_p es la ganancia proporcional, K_i es la ganancia integral y K_d es la ganancia derivativa y u(t) es la señal de control.

$$u(t) = Kp * e(t) + Ki \int_0^T e(t) \cdot dt + kd \frac{de(t)}{dt} \tag{7.17}$$

Al aproximar el termino integral mediante la sumatoria trapezoidaly el termino derivativo mediante y al reemplazar $ki = Kp/TiyKd = Td * Kc$

$$u(k) = Kp[e(k) + \sum_{h=0}^{k}[e(h) + e(h-1)] + \frac{Td}{Ts}[e(k) - e(k-1)]] \tag{7.18}$$

La salida del controlador en el instante $k-1$ luego de simplificar es:

$$u(k) = u(k-1) + q_0 e(k) + q_1 e(k-1) + q_2 e(k-2) \tag{7.19}$$

donde

$$q_0 = Kp(1 + \frac{Ts}{2Ti} + \frac{Td}{Ts}) \tag{7.20}$$

$$q_1 = -Kp(1 + \frac{Ts}{2Ti} + \frac{Td}{Ts}) \tag{7.21}$$

$$q_2 = \frac{KpTd}{Ts} \tag{7.22}$$

Obtenemos las ecuación del controlador PID discreto

$$Cz = \frac{q_0 + q_1 z^{-1} + q_2 z^{-2}}{1 - z^{-1}} \tag{7.23}$$

7.4. Sintonización de controladores PID

La parte proporcional del controlador PID provee acción de control proporcional a la señal de error, la parte integral reduce el error en estado estable mediante la compensación de baja frecuencia con integrador. La parte derivativa mejora la respuesta transitoria del sistema en lazo cerrado con un diferenciador.

Uno de los métodos de sintonización más famosos es el método de ajuste de Ziegler-Nichols. Este método comienza poniendo a cero las ganancias integrales y diferenciales y luego elevando la ganancia proporcional hasta que el sistema es inestable. Luego, el método retrocede la ganancia proporcional en una cantidad predeterminada y establece las ganancias integral y diferencial en función de la frecuencia de oscilación. Las ganancias P, I y D se establecen de acuerdo con el cuadro 7.2 [12].

Respuesta	t levantamiento	Sobre Pico	t Establecimiento	Error	Estabilidad
Aumentando Kp	Disminuye	Aumenta	Aumenta Poco	Disminuye	Degrada
Aumentando Ki	Disminuye Poco	Aumenta	Aumenta	Gran Disminución	Degrada
Aumentando Kd	Disminuye Poco	Disminuye	Disminuye	Cambio bajo	Mejora

Cuadro 7.2: Sintonización PID

7.5. Filtro precompensador

En la subsección 7.1.1 donde se muestra el modelo del convertidor se puede observar que la función de transferencia del lazo de corriente (7.13) presenta un cero en el origen lo cual provoca la eliminación del efecto integrador en la implementación de un contro PID, lo anterior implica que no se puede tener un error de estado estacionario igual a 0, de esta manera la corriente de referencia seria:

$$i_{ref} = I_p sin(wt + \phi), \tag{7.24}$$

donde ϕ hace referencia al ángulo de desplazamiento producido por el error de estado estacionario, por consiguiente el valor de desplazamiento del filtro deberá ser $-\phi$. Con el fin de conseguir ese desplazamiento se hace necesario el uso de una red de atraso, su función de transferencia en tiempo continuo:

$$C(s) = \frac{K}{s + \alpha}, \tag{7.25}$$

y su equivalente en el dominio discreto:

$$C(Z) = \frac{KT_s}{Z + \alpha T_s - 1}, \tag{7.26}$$

donde T_s hace referencia al tiempo de muestreo. [45]

7.6. Inteligencia artificial

Es la rama de la ciencia computacional que se encarga de temas relacionados al aprendizaje y lógica por medio de herramientas que simulan los procesos innatos por el cerebro humano, esto es realizado mediante algoritmos para la toma de decisiones. [46] Las áreas de estudio de esta rama son la robótica, la visión artificial, optimización, representaciones del conocimiento, aprendizaje autónomo, resolución de problemas y procesamiento de lenguaje natural [47].

7.6.1. Algoritmos Bioinspirados

Son métodos de optimización y búsqueda, basados en la teoría evolutiva y los mecanismos de selección natural. Existen comportamientos comunes en la mayoría de bioinspirados los cuales son, el uso de una población inicial y un posterior crecimiento de la misma con cambios comúnmente aleatorios que culminan en una evaluación por medio de funciones o esquemas.

Los operadores genéticos estándar son 3, la *selección* la cual tiene como objetivo determinar los padres de las siguientes generaciones, el *cruce* es la combinación de material genético entre los padres para la creación de nuevas generaciones, los cuales son escogidos a partir del valor de aptitud en la función objetivo y la *mutación* es la encargada de realizar cambios en las nuevas generaciones con el fin de rastrear nuevos espacios. La representación gráfica del comportamiento estándar se presenta en la figura 7.3 [48].

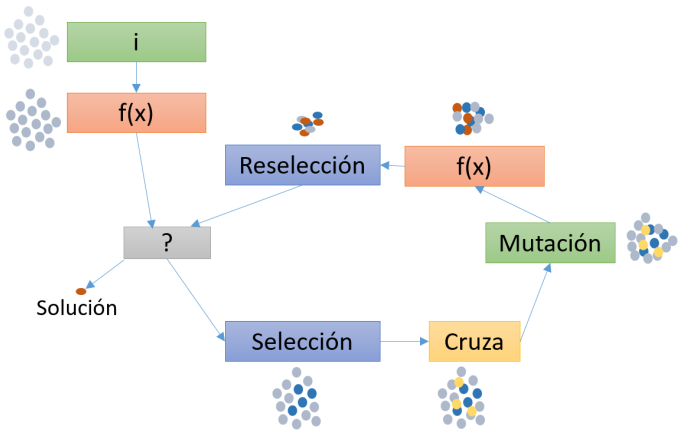

Figura 7.3: Algoritmo Bioinspirado estándar [48].

7.6.2. Selección clonal del sistema inmune

El proceso evolutivo en los seres humanos no solo generó cambios morfológicos de manera externa sino también de forma interna como lo es la inmunología, dado que a lo largo de la existencia se ha llevado a cabo una coevolución con patógenos desencadenando en el desarrollo de células especializadas que han sido las responsables de la supervivencia. Estos tipos de células son los linfocitos de tipo T y de tipo B, las cuales mediante un proceso de recombinación aleatoria (recombinación somática) originan una gran cantidad de nuevas células T y B, cada una de ellas con una disposición de reconocimiento única.

Aunque el funcionamiento del sistema inmune se centra en este tipo de células no son las únicas que intervienen en los procesos realizados por el sistema, también lo conforman otro tipo de células y moléculas, es a partir de esto que se puede definir someramente el sistema inmune como la interacción coordinada de un conjunto de moléculas y células que tiene como propósito diferenciar lo propio de lo extraño y de igual manera lo que resulta ofensivo o inofensivo, suministrando mecanismos para un rápido reconocimiento y eliminación de patógenos. Aun cuando la supervivencia se debe al sistema inmune este no es infalible y ocasionalmente puede presentar fallos a nivel del individuo, generando enfermedades, autoinmunes cuando el perjuicio acontece en la tolerancia central dando comienzo a una autorreactividad, alergias, si contrarresta un antígeno extraño, pero no inofensivo y neoplasias cuando no reconoce las células propias que han tenido transformaciones.

El sistema inmune se divide en dos grupos dependiendo de propiedades como, el tiempo de respuesta, aparición de recombinación genética, uso o no de memoria, especificidad de reconocimiento, los tipos de receptores, entre otras. Estos dos grupos se conocen como respuesta innata y respuesta adaptativa.

La respuesta innata incorpora barreras químicas y barreras físicas propias del sistema inmune. Las células que lo componen son los monocitos y macrófagos, granulocitos (mastocitos, basófilos, neutrófilos y eosinófilos), las células citotóxicas naturales, las células dendríticas y de Langerhans. Los componentes moleculares son las citocinas y las proteínas de fase aguda, este tipo de moléculas y células se encuentran en todos los animales a diferencia de la respuesta adaptativa que solo están compuestas de linfocitos T y B. La respuesta innata es la primera barrera de defensa es por ello que su accionar se encuentra en las mucosas o la dermis vigilando la entrada de patógenos, esto por medio de los mastocitos, macrófagos y células dendritas. [49] Esta se produce como respuesta a moléculas liberadas por células dañadas y estructuras moleculares pertenecientes a microbios [50].

En este artículo los autores proponen el algoritmo de selección Clonal para el reconocimiento de entorno

en un robot móvil. El robot debe cumplir con la función de reconocer objetos por medio de su forma geométrica para luego transportar dichos objetos a un lugar específico dependiendo de la forma: El robot debe tener la capacidad de reconocer objetos nuevos y diferenciarlos de los objetos conocidos. Simulan los objetos geométricos con formas específicas y también sin una forma definida para analizar el resultado de la clasificación. Los autores concluyen que los anticuerpos entrenados clasificaron correctamente los objetos encontrados y mostraron robustez en caso de ruido en la clasificación. El ruido fue modelado como deformación superficial [51].

7.6.3. Respuesta adaptativa

Lo que caracteriza a este tipo de respuesta es su especificidad, se genera posteriormente del contacto con el agente infeccioso y produce una memoria antigénica, esto conlleva a que a medida que el contacto con el agente es mayor, su eficacia para controlarlo irá en aumento. Este tipo de respuesta inicia una vez las células fagocíticas que hacen parte de la respuesta innata presentan a los linfocitos tipo T los antígenos microbianos, estos linfocitos iniciarán la respuesta adaptativa en colaboración con los linfocitos tipo B para la producción de anticuerpos con los antígenos del patógeno.

Principales características de la respuesta adaptativa:

- Son desarrolladas a partir de la previa exposición con el agente infeccioso.
- Logran diferenciar distintos microorganismos y su ejecución se basa por medio de células receptoras de antígeno especifico.
- Esta respuesta es más lenta que la respuesta innata ya que requiere una clonación de células específicas de acuerdo al antígeno.
- Tiene memoria esto deriva en el reconocimiento específico para un control más rápido del patógeno.

La respuesta adaptativa se clasifica dependiendo de su mecanismo de acción en respuesta humoral y respuesta celular.

7.6.4. Principio selección clonal

En la respuesta inmune se pueden encontrar de dos tipos, primaria o secundaria, la primaria sucede cuando el individuo es expuesto por primera vez al patógeno, [50] justo después de los linfocitos tipo T reconozcan que el patógeno no hace parte del organismo, se escogen los anticuerpos que presenten la mayor afinidad con el patógeno y estos serán sometidos a una multiplicación por medio clonación, básicamente una proliferación de copias las cuales posteriormente pasan por un proceso de mutación (llamada hipermutación) (Figura 7.4), siendo esta aleatoria e inversamente proporcional a la afinidad, incrementando así la posibilidad de similitud con el antígeno llevando a su neutralización y posterior eliminación. Cuando el patógeno es eliminado se presenta un exceso de células siendo eliminadas en un proceso llamado autorregulación, sin embargo, algunas de ellas se conservan como células de memoria. La respuesta secundaria sucede cuando el patógeno reaparecen el en individuo en dicho caso el sistema hará uso de las células memoria consiguiendo una respuesta eficiente en un lapso de tiempo corto ya que estas presentan una elevada afinidad [52, 53].

Su uso frecuentemente es para solucionar dos tipos de problemas, optimización y reconocimiento de patrones para ambos el fundamento teórico esta basado en los siguientes principios que hacen parte de la teoría de selección clonal:

1. Uso de memoria

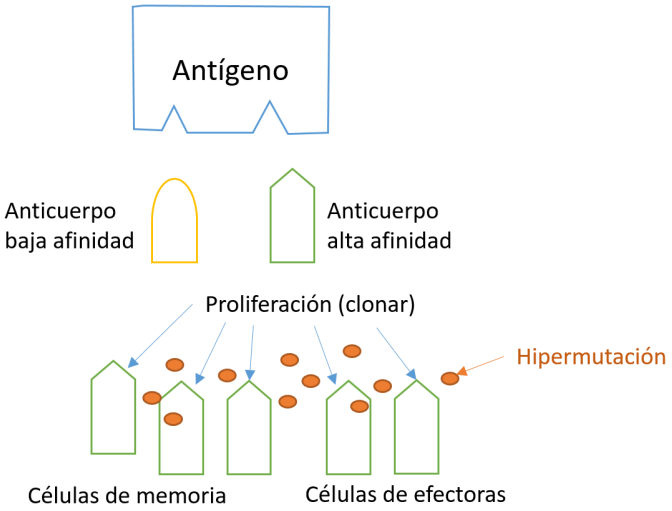

Figura 7.4: Selección clonal [52].

2. Clonación de los anticuerpos que presenten mayor afinidad
3. Eliminación de los anticuerpos con afinidad menor
4. Mutación
5. Mantenimiento de la diversidad por medio de generación de nuevos clones

Respecto al algoritmo estas bases teóricas son implementadas mediante operaciones matemáticas y los datos representaran a los antígenos y anticuerpos que se modifican con el fin de alcanzar una solución óptima.

Reconocimiento de patrones: Su funcionamiento parte de información obtenida de objetos con el fin de establecer relaciones entre ellos, una vez adquirida la información y abstraído las características se procede a la toma de decisiones y para ello el uso del algoritmo de selección clonal.

Explicación del diagrama mostrado en la figura 7.5:

1. Población inicial de antígenos y anticuerpos
2. Cálculo de la afinidad de los anticuerpos y dichas afinidades con guardadas en un vector f.
3. Selección de los elementos de mayor afinidad.
4. A parir de los elementos anteriormente seleccionados se realiza la clonación en función de su afinidad, los elementos con mayor afinidad tendrán más clones que los de menos afinidad.
5. Las nuevas generaciones sufren mutaciones inversamente proporcionales a su afinidad, entre mayor sea la afinidad menor es la mutación.
6. Se calcula la afinidad para los nuevos valores y se guardan en f*
7. Se seleccionan los elementos con mayor afinidad que serán candidatos para entrar al conjunto de memoria.
8. Los anticuerpos de baja afinidad serán remplazados por los nuevos candidatos.

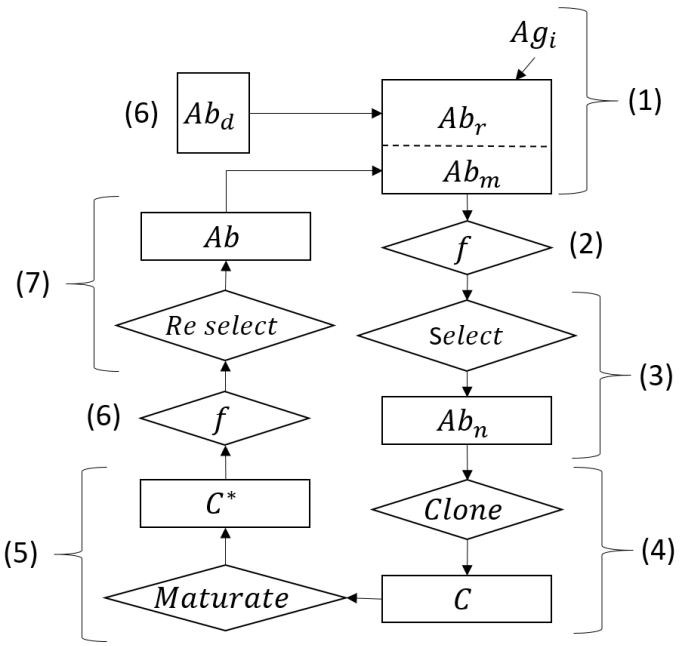

Figura 7.5: Selección clonal en el reconocimiento de patrones [53].

Optimización: El objetivo de este tipo de implementaciones es encontrar la mejor solución para una función objetivo y gracias a su funcionamiento puede realizar búsquedas minuciosas dado que se ajusta los elementos de manera local y con los nuevos elementos se explora nuevos espacios de búsqueda. A diferencia del anterior, este no necesita una población de antígenos sino una función objetivo y la evaluación de no será por la afinidad entre el antígeno y el anticuerpo sino por el valor de obtenido en la función objetivo.

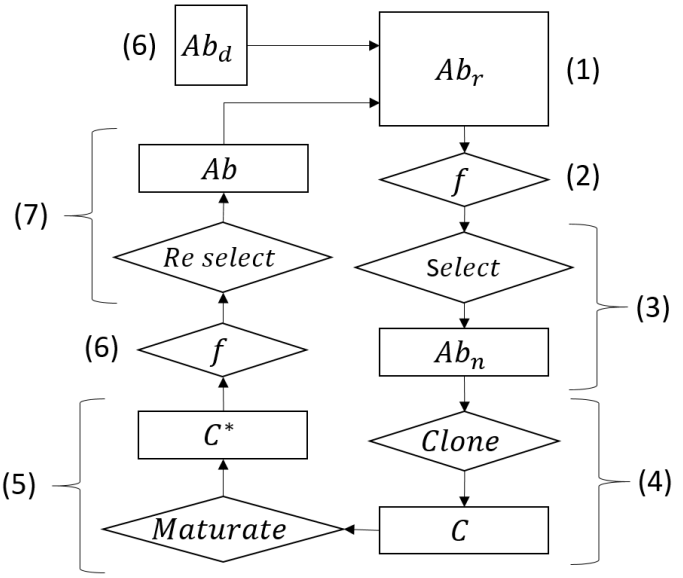

Figura 7.6: Selección clonal en Optimización [53].

Explicación del diagrama mostrado en la figura 7.6:

1. Población inicial anticuerpos
2. Se calcula el valor en la función objetivo de todos los anticuerpos y dichos valores son guardadas en

un vector f.

3. Selección de los elementos de mayor afinidad.
4. A parir de los elementos anteriormente seleccionados se realiza la clonación en función de su afinidad, los elementos con mayor afinidad tendrán mas clones que los de menos afinidad.
5. Las nuevas generaciones sufren mutaciones inversamente proporcionales a su afinidad, entre mayor sea la afinidad menor es la mutación.
6. Se calcula la afinidad para los nuevos valores y se guardan en f*
7. Se seleccionan los elementos con mayor afinidad que serán candidatos para entrar al conjunto de memoria.
8. Los anticuerpos de baja afinidad serán remplazados por los nuevos candidatos.

La información referente al algoritmo de selección clonal es tomada de [53]

Capítulo 8

Metodología

La planificación del desarrollo se llevó a cabo de la manera que lo expone el diagrama de flujo de la figura 8.1, el cual representa secuencialmente cada una de las actividades previstas en el transcurso de la realización del proyecto con miras en el cumplimiento de los objetivos planteados, posterior al diagrama se dará una breve descripción de las tareas a realizar en cada actividad.

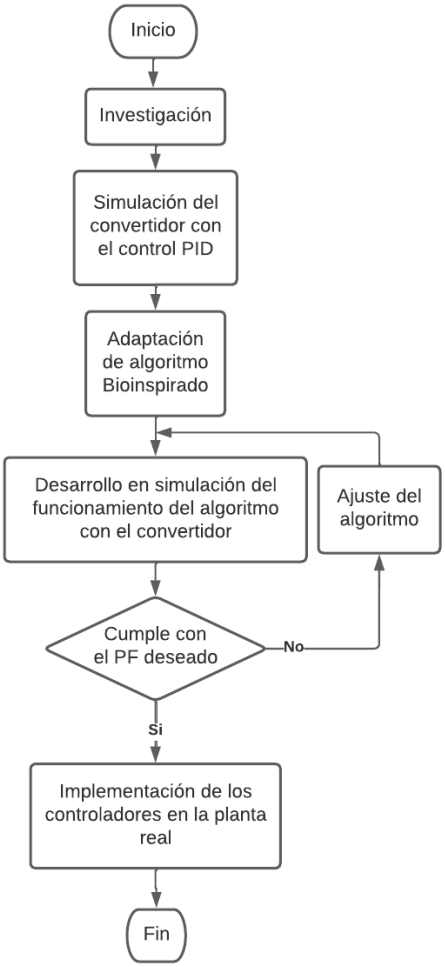

Figura 8.1: Diagrama de flujo de las actividades a realizar.

8.0.1. Investigación

Se realizo un estudio a profundidad de los algoritmos que han sido utilizados para la sintonización de controladores PID y cuales de ellos lograron mejorar considerablemente los valores de la respuesta del control y con ello las prestaciones del prototipo, también se indagará sobre los que ya se utilizaron en convertidores eléctricos y cuales en convertidores eléctricos con corrección de factor de potencia, esto con el fin de determinar un algoritmo con antecedentes de buen desempeño y poco uso en este tipo de sistemas para llevar a cabo el desarrollo del proyecto.

8.0.2. Simulación del convertidor con el control PID

A partir del trabajo realizado por el docente Jhon Bayona, Antonio Bolívar y Nelson Díaz [45] los cuales presentan en su articulo la construcción de los controladores de corriente y voltaje para el convertidor HBB, se realizó la simulación con los datos allí expuestos con el objetivo de obtener un modelo que emule el comportamiento del prototipo de la planta real y con ello posteriormente añadirle el algoritmo Bioinspirado para la sintonización de los controladores del lazo de corriente.

8.0.3. Adaptación del algoritmo Bioinspirado

Con la decisión basada en la investigación sobre cual Bioinspirado se utilizó, se continuo con escoger los parámetros del algoritmo tales como parámetros de la función objetivo, numero de épocas, numero de poblaciones, clones, cruces, mutaciones y demás consideraciones pertinentes para su funcionamiento, también es necesario realizar pruebas para determinar comportamientos de la planta como tiempo de establecimiento y variación de los parámetros una vez se encuentren en estado estable, por ultimo se determinó cual seria el funcionamiento en conjunto de ambas sintonizaciones (PID, Filtro precompensador) ya que puede ser de manera simultanea entre ambas o iniciar una posteriormente de la otra.

8.0.4. Desarrollo en simulación del funcionamiento del algoritmo con el convertidor

Teniendo los parámetros iniciales del algoritmo definidos se continuó al desarrollo en simulación, en el cual se evaluó el tipo de variables a utilizar para tener una persistencia en ellas a medida que se ejecute cada población, el bloque en el cuales de codificó el algoritmo, la forma de visualizar de manera consolidada los datos que va dejando cada una de las poblaciones y los bloques de adquisición de medidas en las variables de la función objetivo.

8.0.5. Ajustes del algoritmo

Una vez implementado el algoritmo con los parámetros escogidos inicialmente en la simulación junto con el modelo del convertidor se inicio la pruebas para ir determinando los valores obtenidos y en caso de no ser los esperados ir realizando ajustes en los parámetros, este proceso de tomar datos y posteriormente hacer ajustes se llevará a cabo hasta que se consigan los valores esperados.

8.0.6. Implementación de las constantes obtenidas en la planta real y comparación con la simulación

Ya teniendo ajustados los parámetros del algoritmo y encontrados los valores esperados en la simulación que cumplan con el objetivo, se continuó con la implementación en el prototipo de la planta el cual ya

se encontraba construida, en este punto del desarrollo se llevó a cabo la sintonización del controlador por medio de una herramienta que sintonice automáticamente un control a partir de una función de transferencia que describe el modelo de la planta, esto con el objetivo de implementar las constantes encontradas por dicha sintonización y comparar el factor de potencia obtenido por las constantes sintonización del Bioinspirado, por ultimo se probaron los resultados obtenidos por el Bioinspirado en puntos de operación diferentes al sintonizado y con ello concluir el trabajo.

Capítulo 9

Resultados

En el presente capitulo de lleva a cabo la ejecución de lo planteado en el desarrollo metodológico el cual consta de 4 subcapítulos, el primero hace referencia a la investigación y como resultado de esta se determina el algoritmo a utilizar para la sintonización, posteriormente se adapta el algoritmo al nuestro problema de sintonización,luego se presentan pruebas, ajustes realizados para lograr el objetivo en cuanto a factor de potencia y por ultimo, implementación de las constantes en el control de la planta real.

9.1. Investigación

Para la investigación se buscan artículos actualizados en las bases de datos que contienen elementos de investigación como IEEE®, SCOPUS®, SCIENCEDIRECT® y SCIVAL®, los documentos consultados están repartidos en los siguientes temas: Algoritmos Bioinspirados para la sintonización de controladores, Algoritmos Bioinspirados para sintonización de controladores en convertidores, Algoritmos Bioinspirados para sintonización de controladores en convertidores con corrección de factor de potencia.

En al capitulo 3 para cada uno de los apartados se realizó la búsqueda de artículos que den valor a la investigación, para poder seleccionar un algoritmo que pueda cumplir con los objetivos.

En el estado del arte se hizo un sondeo de los algoritmos con mayor desarrollo, funciones de aptitud utilizadas, parámetros, ventajas y desventajas de cada uno, aplicaciones hasta el momento, y sus resultados. Una de las razones fundamentales para seleccionar el algoritmo es que no haya demasiados desarrollos en convertidores ya sea con o sin corrección de factor de potencia y tenga un buen desempeño en la sintonización de controladores.

Con la investigación realizada de algoritmos utilizados para la sintonización de controladores en diferentes convertidores 3.3, se encontraron 3 algoritmos relevantes, el primero de ellos basado en colonias de abejas, este mismo algoritmo es utilizado en la sintonización de controladores en convertidores 3.4 y en convertidores con corrección de factor de potencia 3.5 y su desempeño fue mejor en comparación con el algoritmo PSO, el segundo es el algoritmo basado en colonias de hormigas, el cual también es utilizado en la sintonización de controladores en convertidores presentando mejor desempeño frente al algoritmo genético y el tercer algoritmo es basado en la selección clonal del sistema inmune, es utilizado en una planta con comportamiento no lineal y no se encontraron usos en convertidores. También presenta rápida convergencia en encontrar una solución global. Este algoritmo es comúnmente utilizado en problemas de optimización multimodal dando buenos resultados en problemas con varios máximos locales y globales.

9.2. Selección y adaptación del algoritmo

A partir de la investigación realizada sobre algoritmos Bioinspirados, se encuentra que el algoritmo basado en el sistema inmune ha sido utilizado para la sintonización de controladores y por medio de este se pudo mejorar la respuesta del sistema, otro resultado arrojado de la investigación es la inexistencia del uso de este algoritmo para la sintonización de convertidores ya sea con o sin corrección de factor de potencia, debido a esto se opta por la utilización del algoritmo basado en el principio de selección clonal del sistema inmune. El diagrama de flujo del algoritmo se muestra en la figura 9.1.

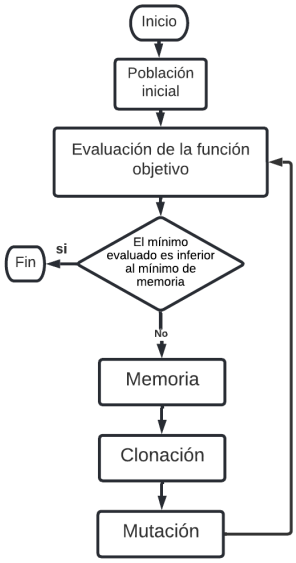

Figura 9.1: Diagrama de flujo a implementar del algoritmo basado en la selección clonal.

Para la población inicial se establecen 200 poblaciones las cuales se generar de diversas formas, las utilizadas en este proyecto fueron dos, la primera, a partir de mutar aleatoriamente los valores de las contantes presentes en los controladores que funcionen para un determinado punto de operación y la segunda, de forma aleatoria dentro de un rango acotado para cada tipo de constantes, los resultados de cada una de las formas son mostrados en la sección 9.3.

Cada una de las poblaciones iniciales se evalúan en el modelo simulado de la planta y con el resultado de factor de potencia dado por cada población se calcula el error por medio de la función de aptitud mostrada en 9.1.

$$e = 1 - pf \tag{9.1}$$

Con el objetivo de determinar el lapso de tiempo adecuado en el que cada una de las poblaciones alcanza la su valor de estado estacionario, el algoritmo pasa a la siguiente generación, se realiza un análisis a partir de la gráfica de factor de potencia en simulación, los datos para dicho análisis son tomados de [45]. A partir de la gráfica arrojada del factor de potencia (Figura 9.2) se infiere que la planta tiene un valor de estabilización cercano a los 0.2 segundos y una vez se estabiliza, la variación máxima que presenta es de 7.765e-5 mostrada en la figura 9.3. Con estos resultados se desarrolla la función encargada de determinar el cambio de población, esta función tomará el valor de factor de potencia con una frecuencia de 0.04 segundos y evaluará que la variación entre las ultimas 5 muestras no sea mayor a 1e-3, contemplando la posibilidad de que alguna de las poblaciones tenga una respuesta inestable, se cambiará de población

pasadas 10 muestras lo que equivales a un tiempo transcurrido de 0.4 segundos.

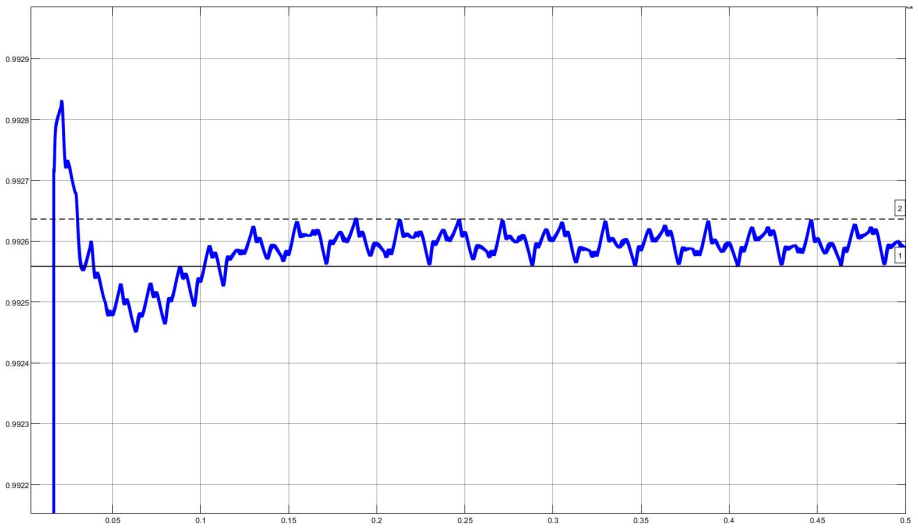

Figura 9.2: factor de potencia.

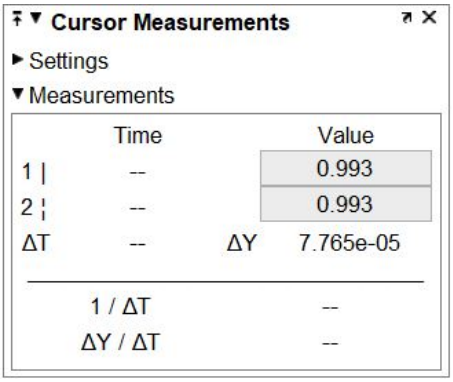

Figura 9.3: Variación del factor de potencia en estado estable.

Cuando se tengan los valores de la función de aptitud de cada una de las poblaciones se obtiene la población que obtuvo mayor factor de potencia de las 200 poblaciones y se reemplaza por la población en memoria con menor factor de potencia, si no disminuye el valor de error o es igual se dará por terminado el algoritmo y la planta se queda con la población que arrojo un valor más cercano al esperado, de lo contrario se toman las 20 mejores poblaciones, se guardan en memoria con su respectivo valor de aptitud para una posterior clonación y mutación.

De cada una de las 20 poblaciones guardadas en memoria se generan 10 clones con el fin de mantener una población total de 200, la mutación que sufre cada una de las poblaciones está dada por el porcentaje de error que presento su población padre y la iteración del algoritmo, con el objetivo que las poblaciones clones con mayor afinidad tengan un porcentaje de mutación menor y a su vez, a medida que se generen nuevas iteraciones dicho porcentaje de mutación disminuya. Una vez se tengan mutadas las 200 poblaciones se comienza una nueva iteración pasando a la siguiente generación.

Estas especificaciones del algoritmo son usadas tanto para la sintonización del controlador PID como en la sintonización del filtro precompensador, con la diferencia que la población de la sintonización del

controlador tiene las constantes Kp, Ki, Kd y un 4to parámetro N propio del bloque *Discrete PID Controller*, para la sintonización del filtro son K y α.

Cada sintonización no se realiza en simultaneo, ya que de ser así demandaría un mayor tiempo de simulación dado que para cada una de las poblaciones que genere la sintonización del controlador PID se debería sintonizar el valor de las constantes del filtro, dicho lo anterior el algoritmo inicia la sintonización del controlador PID y luego sintoniza el filtro precompensador.

9.2.1. Consumo de memoria del algoritmo

Para la medición del consumo de memoria se hace uso del comando *whos* de MATLAB® que indica el tipo, el numero de bytes y la dimensión de la variable analizada. Este procedimiento se realizó para los datos guardados en memoria: COnstantes de PID y filtro, resultado de la función de aptitud y demas variables, de forma general todos los tipos de constantes son de tipo *double* y un consumo de 8 bytes, con este dato se realizan los cálculos para ambas sintonizaciones, los resultados se exponen en el cuadro 9.1.

Sintonización del controlador PID		
Variable	***Dimensión***	***Bytes***
Población actual	1x4	32
Poblaciones	200x4	6400
Memoria	20x5	800
Sintonización del filtro precompensador		
Población actual	1x2	16
Poblaciones	200x2	3200
Memoria	20x3	480
Variables compartidas		
Función de actitud de las poblaciones	200x1	1600
Factor de potencia actual	1x1	8
Total		12532

Cuadro 9.1: Consumo de memoria de las sintonizaciones.

9.3. Simulación y ajustes

La simulación se realiza en el software simulink de MATLAB®, se utilizan los bloques de la librería SIMSCAPE ELECTRICAL para modelar el convertidor (figura 9.4). Los controladores PID discretos y el precompensador son implementados mediante los bloques *Discrete PID Controller* y *Discrete Transfer Fcn* y se utiliza medidores de voltaje y corriente para sensar estas variables (figura 9.5). El desarrollo del algoritmo se lleva a cabo en el bloque *MATLAB Function* teniendo como entrada la variable a mejorar, en este caso el factor de potencia (figura 9.6). Para el algoritmo se necesita bloques *Data store mamory* para cada variable que necesite almacenar su valor. Se debe declarar que las variables en el script se inicializan de forma global. Para la visualización de los factores de potencia, poblaciones, iteraciones y demás variables que maneja el algoritmo se hace por medio *View diagnostics* que genera una ventana auxiliar con los datos y estos se pueden exportar en una archivo .txt.

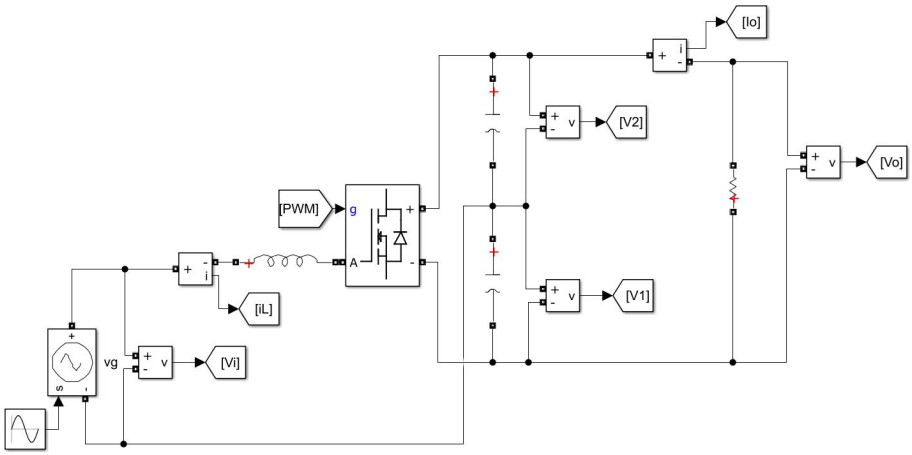

Figura 9.4: Modelo en simulación del convertidor.

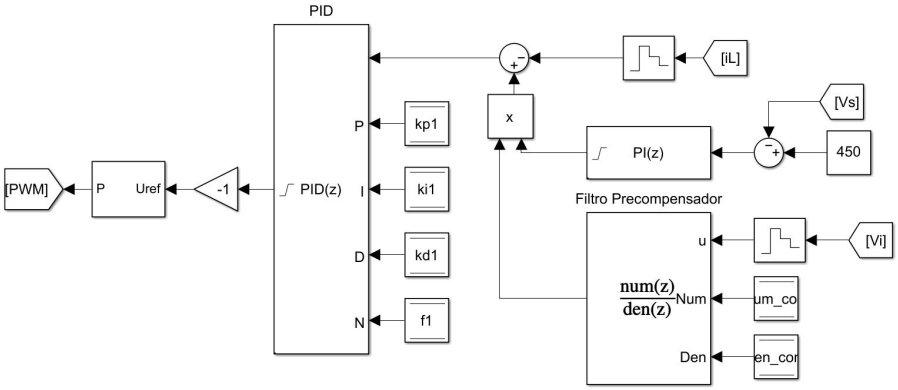

Figura 9.5: Lazos de control del convertidor.

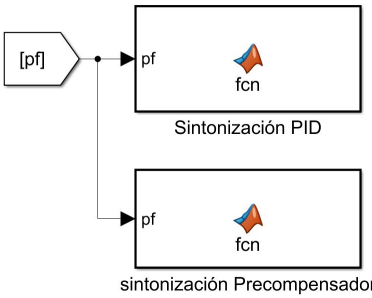

Figura 9.6: Bloques para el desarrollo del algoritmo.

9.3.1. Sintonización PID

Se busca sintonizar el algoritmo para una carga de 6.6 kΩ con un voltaje de entrada de 120 Vrms, la población inicial se genera a partir de las constantes encontradas en [45], dichos valores son $kp = 0{,}295, ki = 1281, kd = 7{,}4e-6, N = 59{,}56e3$, las 200 poblaciones se generan tomando cada uno de estos valores y sumándoles una valor aleatorio entre menos y mas el valor de cada constante tal como se muestra en 9.2.

$$k_n = k + rand(-k, k) \tag{9.2}$$

En la primera iteración se tiene valores de error en un rango de 0,437 y 0,023, correspondientes a factor de potencia de 0.563 y 0.977. Con el fin de cuantificar la dispersión entre el conjunto de valores de error, se realiza un calculo de desviación estándar por medio del uso de comandos de MATLAB®, dando como resultado un valor de 0,0812. Los resultados de aptitud obtenidos en la primera iteración se muestran en la figura 9.7.

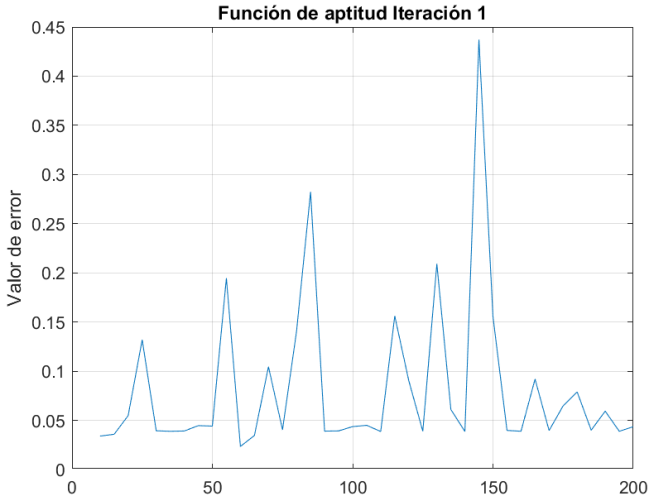

Figura 9.7: Valores de error arrojados por la primera iteración primera prueba.

Los mejores 20 valores obtenidos en la primera iteración fueron clonados y mutados para una segunda iteración, arrojando valores de error entre 0,042 y 0,038 lo que corresponde en factor de potencia a 0.958 y 0.962, con una desviación estándar de 0,0015. La totalidad de los valores es mostrada en la figura 9.8, debido a que el menor valor de error no es inferior al obtenido en la iteración el algoritmo se dio por terminado.

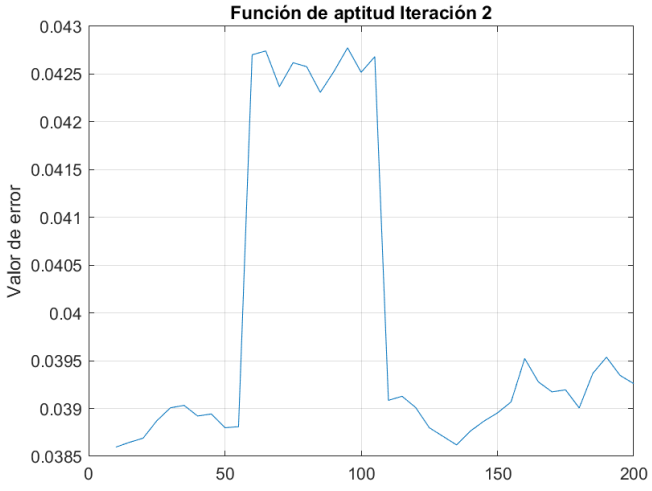

Figura 9.8: Valores de error arrojados por la segunda iteración primera prueba.

Para la segunda prueba se realiza un cambio respecto a cómo se generan las poblaciones iniciales, para este caso será de manera aleatoria en un rango de 0 hasta un máximo de 100 veces el valor de cada constante, dicho lo anterior estos serían los rangos para cada una de las constantes, kp = [0 a 29.5], ki =

[0 a 128.1e3], kd = [0 a 7.6e-4] y N = [0 a 59.56e5].

Los datos de error obtenidos de la primera iteración son mostrados en la figura 9.9 con un rango de error entre 0,891 y 0,143 en valores de factor de potencia serían 0.109 y 0.857, con desviación estándar de 0,1094, posteriormente en la iteración dos da como resultado los valores mostrados en la figura 9.10, con valor máximo y mínimo de 0,956 y 0,0089 que llevándolos a valores de factor de potencia equivalen a 0.044 y 0.991, desviación estándar de 0,2805, el valor mínimo de error es menor al encontrado en la iteración 1, debido a esto el algoritmo realiza una nueva iteración y como resultado se observan los valores de la figura 9.11, con un rango de valor entre 0,0191 y 0,01, lo que en términos de factor de potencia serían valores de 0.989 y 0.99, desviación estándar de 0,00285, dado que el mínimo valor de error encontrado en la iteración 3 no es inferior al encontrado en la iteración 2 se da por terminada la sintonización.

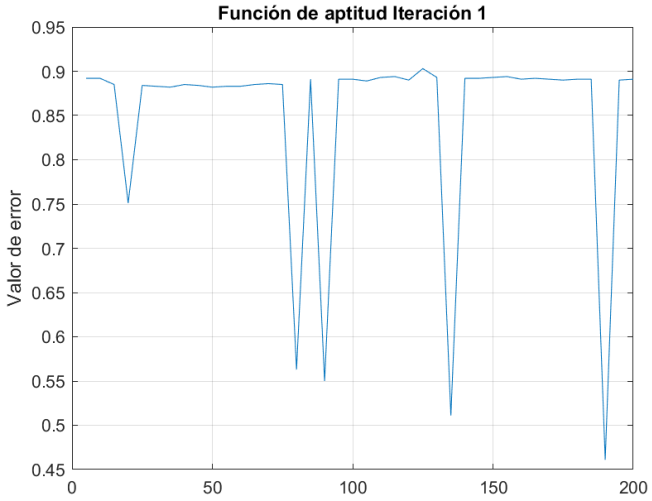

Figura 9.9: Valores de error arrojados por la primera iteración segunda prueba.

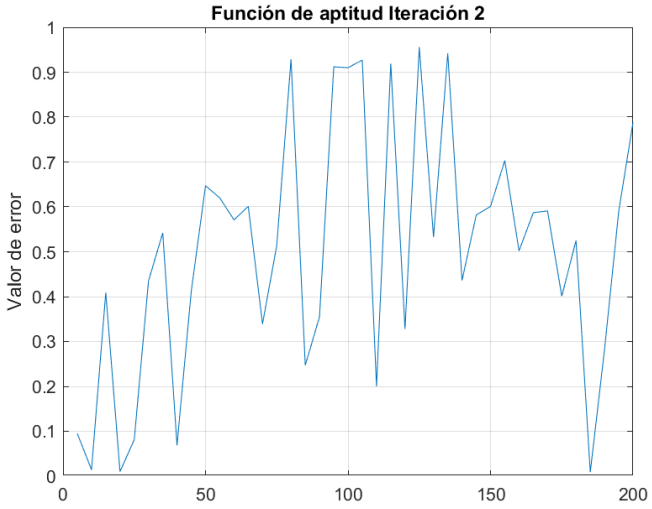

Figura 9.10: Valores de error arrojados por la segunda iteración segunda prueba.

Los resultados máximos y mínimos de factor de potencia y la desviación por iteración de las pruebas realizadas son reflejados en las tabla 9.2.

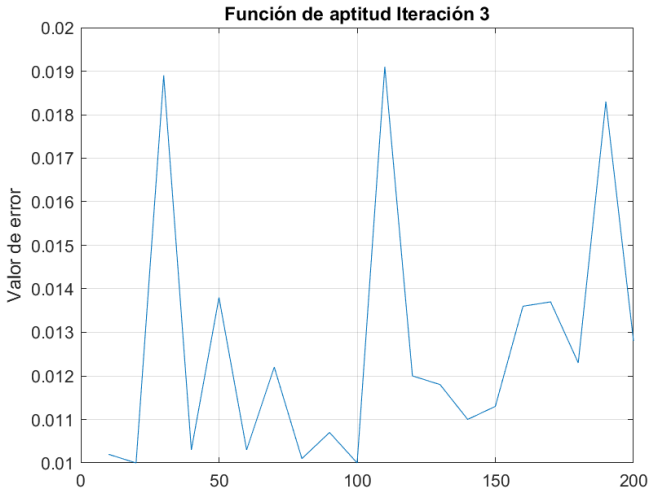

Figura 9.11: Valores de error arrojados por la tercera iteración segunda prueba.

	Iteración 1			Iteración 2			Iteración 3		
	Mínimo	Máximo	Desviación	Mínimo	Máximo	Desviación	Mínimo	Máximo	Desviación
Prueba 1	0.563	0.977	0.0812	0.958	0.962	0.0015			
Prueba 2	0.109	0.857	0.1094	0.044	0.991	0.2805	0.989	0.99	0.00285

Cuadro 9.2: Datos comparativos entre pruebas

Valores máximos de corriente

En cada una de las poblaciones generadas por el algoritmo en las pruebas realizadas se tomo el valor de corriente RMS y en el cuadro 9.3 se muestra el más elevado de todo el conjunto de poblaciones, con ello se puede deducir que en la prueba 1 se encontraron soluciones con un alto factor de potencia pero con una corriente de entrada mucho mayor.

	Máximo de corriente RMS (A)
Prueba 1	42.4
Prueba 2	0.8

Cuadro 9.3: Valores máximos de corriente RMS

9.3.2. Sintonización Filtro Precompensador

Para la sintonización del filtro precompensador se busca sintonizar dos constantes, debido a la función de transferencia del filtro (7.26) y el análisis mostrado en [45] el termino $\alpha T_s - 1$ puede tomar un valor entre -1 y 0, debido a esto se inicializan las poblaciones para el termino $\alpha T_s - 1$ de manera aleatoria con datos en este rango, para el termino del numerador kT_s se inicializan las poblaciones con un rango de 10 veces el valor de los datos encontrados en [45] lo que equivale a [0 a 0.401].

Al finalizar la primera iteración se tienen valores mostrados en la figura 9.12 con errores en un rango de 0,577 y 0,00134, en términos de factor de potencia equivale a 0.423 y 0.9986, con una desviación estándar de 0,077, de esta primera iteración se toman los 20 mejores valores para ser clonados posteriormente mutados y con ello empezar una nueva iteración la cual da como resultado lo mostrado en la figura 9.13, el rango de errores se establece en 0.169 y 0,0015, pasados a valores de factor de potencia son 0.831 y 0.9984, con una desviación estándar de 0,0277. La sintonización se da por terminada debido a que el valor mínimo de la segunda iteración no es inferior al encontrado en la primera iteración.

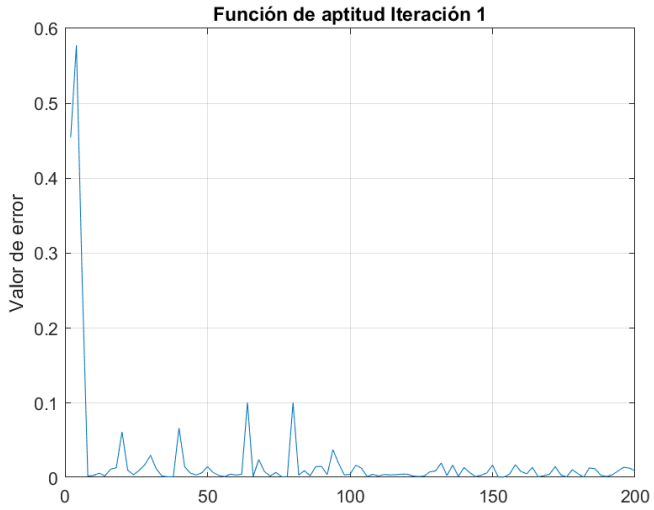

Figura 9.12: Valores de error arrojados de la primera iteración para la sintonización del filtro precompensador.

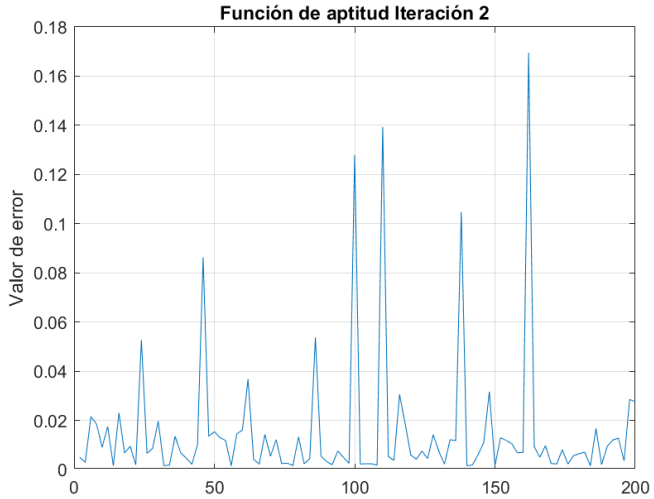

Figura 9.13: Valores de error arrojados de la segunda iteración para la sintonización del filtro precompensador.

Como resultado de la sintonización del controlador PID y el filtro precompensador se alcanza un factor de potencia de 0.998, en el cuadro 9.4 se enuncian los valores de constantes encontradas por el algoritmo tanto en la sintonización del controlador PID como en la sintonización del filtro precompensador y en la figura 9.14 se muestra la gráfica de voltaje y corriente de entrada donde se puede apreciar que se encuentran en fase, cabe resaltar que para efectos de visualización los valores de corriente se multiplican por una constante debido a que las magnitudes de ambas variables son distintas.

9.4. Implementación

El prototipo de convertidor mostrada en la figura 9.15 ya se encontraba previamente construido [45], esta compuesto por una inductancia de referencia EA77625 con núcleo de ferrita, condensadores electrolíticos de 250v, transistores de STMicroelectronics® referencia STP13NK60ZPF y como driver de disparo el

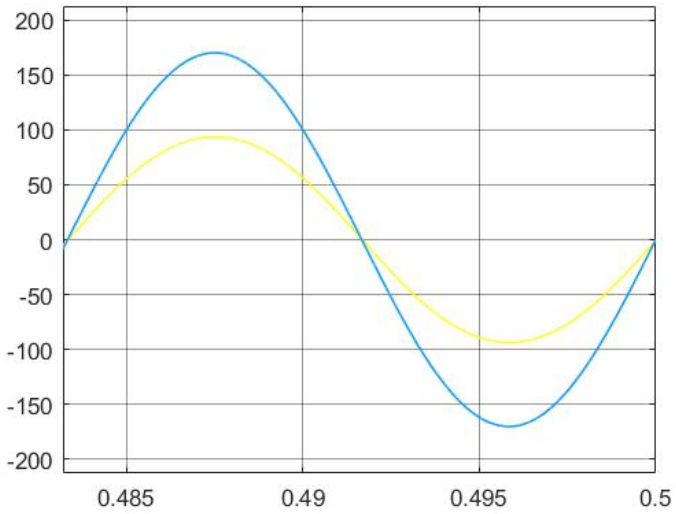

Figura 9.14: Voltaje y corriente de entrada posterior a la sintonización del controlador PID y el filtro precompensador.

Controlador PID	
k_p	0.4133
k_i	1388
k_d	5.3e-6
N	35146
Filtro precompensador	
k	283.5e-4
α	-965.4e-3

Cuadro 9.4: Constantes encontradas por el algoritmo para el controlador PID y filtro precompensador

integrado HCPL-3120. Para el control se utiliza un tarjeta DSP de Texas Instruments® TMDX28069USB. La medición de voltaje se hace por medio de amplificadores operacionales en modo diferencial y para la corriente el sensor LEM HAS50-S mediante efecto Hall [45].

La carga de la figura 9.16 es elaborada de tal manera que sea posible variar entre diferentes magnitudes, se cuenta con 12 paralelos de resistencias cada una de ella de 44 kΩ a 10w y estas se pueden ir uniendo por medio del accionar de un interruptor, dichas resistencias se encuentra cubiertas en los lados superior e inferior por dos laminas metálicas con el fin de disipar la calor.

Con el objetivo de comparar los resultados se llevaran a cabo dos pruebas, ambas para una carga de 6.6 kΩ, en la primera se llevará a cabo la sintonización del controlador PID por medio de la herramienta *PID Tuner* de SIMULINK ® y en el segundo se implementará las constantes encontradas por medio de la sintonización con el algoritmo Bioinspirado.

El elemento de medición utilizado es analizador energía de la marca *AEMC INSTRUMENTS ®* referencia 3945-B, los datos de resolución en corriente y en voltaje son mostradas en el cuadro 9.5 [54]. Debido al bajo rango de corrientes que se manejan en el prototipo de la planta y la alta resolución del instrumento de medida se opta por envolver 10 veces el cable en el núcleo de la sonda amperimétrica con el objetivo de aumentar por 10 el valor de corriente en la visualización del analizador.

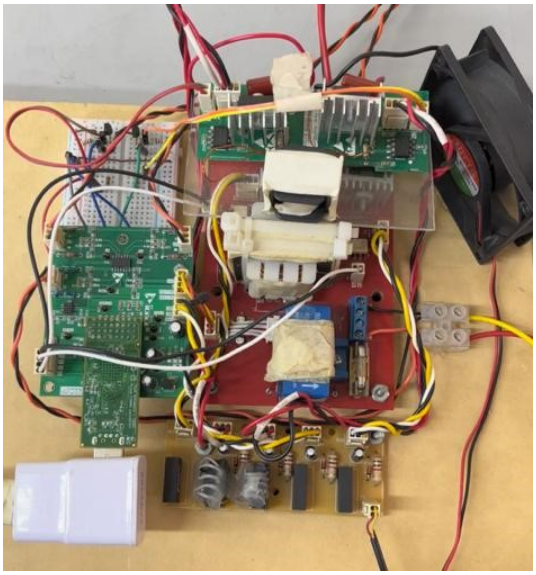

Figura 9.15: Insumo prototipo de planta.

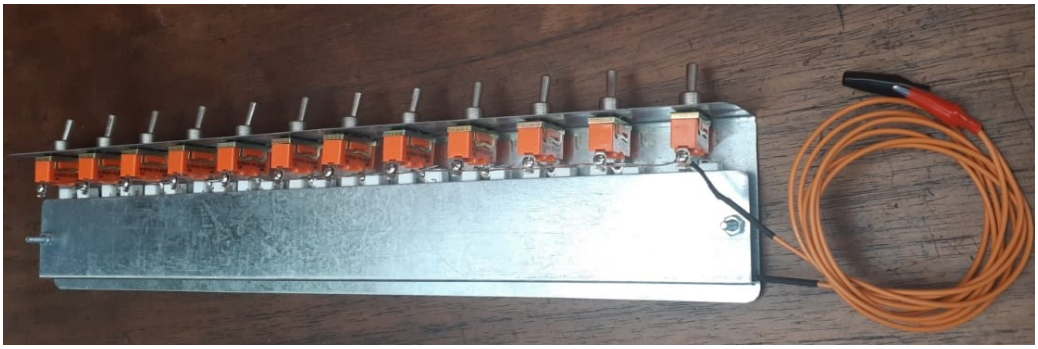

Figura 9.16: Carga elaborada para diferentes valores.

AEMC INSTRUMENTS ® provee un Software llamado PowerPad ® el cual tiene la función de visualizar las medidas observadas en el analizador en tiempo real y descargar las magnitudes en un archivo .xlsx y con ello hacer el tratamiento debido a los datos de corrientes para visualizarlos en la amplitud real.

Función	**Resolución**
Voltaje	0,1V
Corriente	0,1A si I<1000V

Cuadro 9.5: Resolución para corriente y voltaje del analizador.

La implementación en la DSP de cada una de las acciones (Proporcional, Integral, Derivativa) del control PID, se da de la misma manera que se muestra en el bloque usado en simulación *Discrete PID Controller* de SIMULINK®, donde P es la acción del proporcional 9.3, I la acción integral 9.4 y D la acción derivativa 9.5, el valor de T_s es el mismo empleado para la discretización de la función de transferencia 20 μs.

$$P = k_p \tag{9.3}$$

$$I = k_i \frac{T_s}{2} \tag{9.4}$$

$$D = k_d \frac{N}{1 + NT_s} \tag{9.5}$$

9.4.1. Constantes a partir de la herramienta PID Tuner

El proceso que se lleva a cabo tiene como primer paso encontrar la función de transferencia en tiempo discreto con los valores del nuevo punto de operación, posteriormente se realizará la sintonización por medio del PID Tuner, estas constantes sintonizadas se simulan en el modelo de la planta real y como ultimo paso se implementan en el prototipo.

La función de transferencia en tiempo continuo que describe la malla de corriente es 9.6 [45].

$$G(s) = \frac{\frac{V_s}{L}s^2 + \frac{2v_s}{RLC}s}{s^3 + K_2 s^2 + K_1 s + \frac{1}{RLC^2}} \tag{9.6}$$

$$K_1 = \frac{2I_p^2 \omega^2 L}{V_s^2 C} + \frac{1}{2LC} + \frac{2(r_L + r_{DS})}{RLC} \tag{9.7}$$

$$K_2 = \frac{r_L + r_{DS}}{L} + \frac{2}{RC} \tag{9.8}$$

Reemplazando los valores de el cuadro 9.6 para una carga de 6.6 kΩ se obtienen la función de transferencia 9.9.

$$G(s) = \frac{90000s^2 + 3{,}158e^5 s}{s^3 + 213{,}5s^2 + 1{,}001e^6 s + 3{,}509e^6} \tag{9.9}$$

Se usa el método de retención de orden cero (ZOH) para discretizar la funcion $G(s)$, tambien se multiplica por z^-1 debido a que el ciclo útil se carga en el siguiente periodo de conmutación y como resultado se obtiene la función de transferencia 9.10.

$$G(z) = \frac{0{,}89805(z^2 - 2z + 1)}{z^4 - 2{,}995z^3 + 2{,}996z^2 - 0{,}9958z} \tag{9.10}$$

Por medio del PID Tuner se realiza la sintonización automática de las contantes Kp, Ki, Kd y N para la función de transferencia $G(z)$, como resultado de la sintonización de las constantes se obtienen los valores mostrados en el cuadro 9.7.

Con el valor de estas constantes se realiza la simulación de la planta, el factor de potencia obtenido es de 0.962 y la gráfica comparativa de la corriente de entrada y el voltaje de entrada se muestra en la figura 9.17.

Las constantes del controlador del cuadro 9.7 son cargadas en la DSP y como resultado se la medición se tiene la figura 9.18 con la salvedad que la amplitud se encuentra multiplicada por 10, los datos de esta medición fueron exportados y graficados mediante el software MATLAB® con los datos reales de magnitud en corriente esto es mostrado en la figura 9.19. El factor de potencia obtenido de la sintonización es de 0.968.

$C_1 = C_2$	$100uF$
L	$5mH$
r_L	0,2 $Ohms$
r_{DS}	0,85 $Ohms$
R	5 $kOhms$
V_{gRMS}	120 V
I_p	0,8 A
ω	376,9 rad/s
f	50 kHz
T_s	20 μs
T_m	20 μs
V_s	450 Vdc
V_d	0 Vdc

Cuadro 9.6: Valores de los componentes y variables para una carga de 6.6 kΩ.

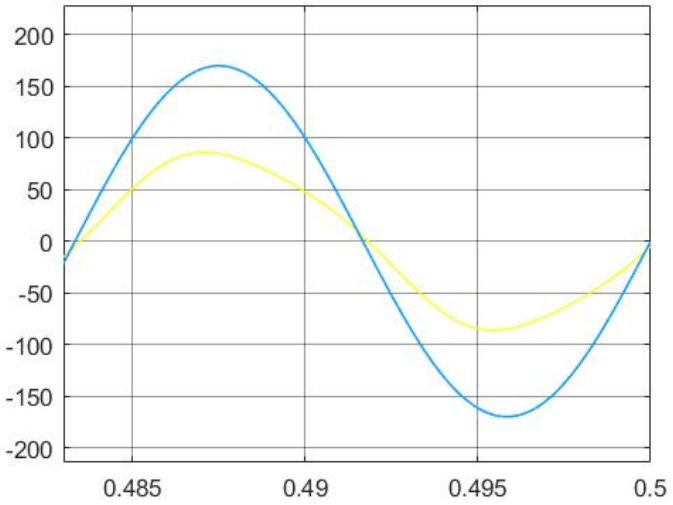

Figura 9.17: Voltaje y corriente de entrada simulados con las constantes encontradas mediante el PID Tuner.

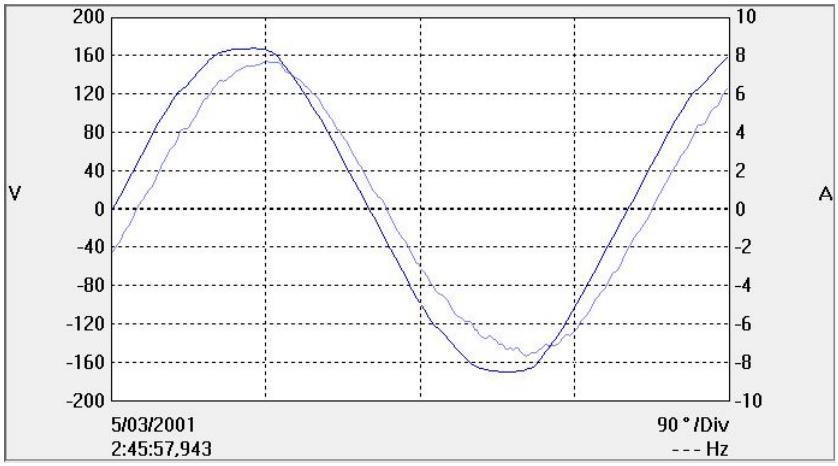

Figura 9.18: Resultados del control sintonizado mediante PID Tuner observados en el analizador.

k_p	0.488
k_i	4178
k_d	897e-9
N	39398

Cuadro 9.7: Constantes encontradas por medio del PID Tuner.

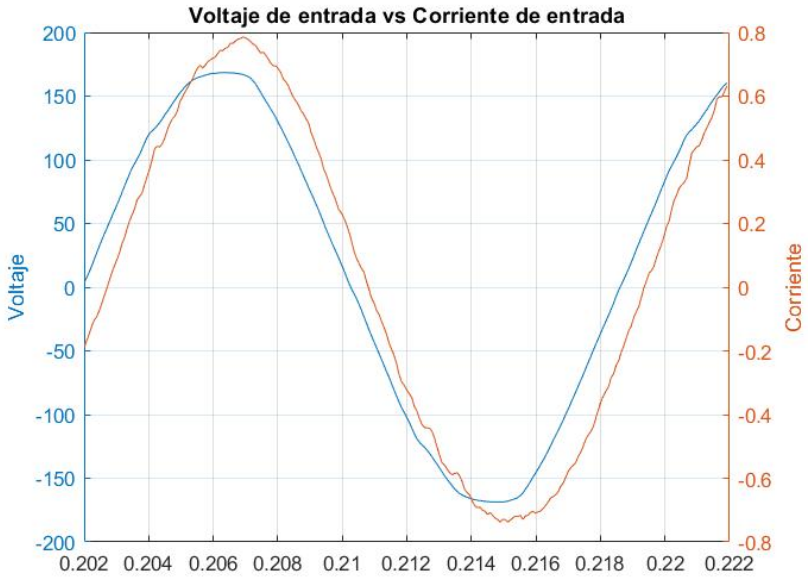

Figura 9.19: Resultados del control sintonizado mediante PID Tuner con valor de corriente real.

9.4.2. Constantes base parte la sintonización del algoritmo

La sintonización del algoritmo se realizó a partir de las constantes encontradas en [45], dichos valores son $kp = 0{,}295, ki = 1281, kd = 7{,}4e-6, N = 59{,}56e3$, estas constantes son implementadas en el prototipo de planta real y se toma el valor de factor de potencia para 11 valores de carga que van desde 2.6 kΩ hasta 40.1 kΩ corresponden a variación en potencia desde los $105w$ hasta $34.1w$, los valores de factor de potencia y potencia obtenidos para cada valor de resistencia se presentan en el cuadro 9.8.

9.4.3. Constantes a partir de sintonización con el algoritmo Bioinspirado

Las contantes del cuadro 9.4 las cuales fueron encontradas por el algoritmo basado en la teoría de selección clonal tanto en la sintonización del controlador PID como del filtro precompensador, serán implementadas en el prototipo de la planta con el fin de validar un factor de potencia igual o cercano al obtenido en simulación de 0.998, posteriormente de realizarán variaciones en la carga para determinar que factor de potencia se tiene a diferentes puntos de operación con las mismas contantes.

Como resultado de la implementación de las constantes para el controlador PID y el filtro precompensador encontradas por el algoritmo se presenta la figura 9.20 donde se observan las gráficas de voltaje y corriente de entrada tomadas directamente desde el analizador con un factor de potencia de 0.998 siendo el mismo valor obtenido en la simulación y un voltaje de salida de 440 v.

Los datos de las magnitudes de cada una de las señales fueron exportados, los de corriente fueron

Resistencia	***Factor de potencia***	***Potencia (w)***
2.6kΩ	0.989	105.6
2.9kΩ	0.99	98.2
3.3kΩ	0.992	90.3
3.8kΩ	0.995	80.7
4.5kΩ	0.997	74.3
5.3kΩ	0.998	64.8
6.6kΩ	0.998	56.8
8.2kΩ	0.997	52.1
10.9kΩ	0.995	44.5
19.3kΩ	0.988	39.0
40.1kΩ	0.967	34.1
Promedio	0.9914	

Cuadro 9.8: Valores de factor de potencia y potencia para cada una de las cargas con las constantes iniciales.

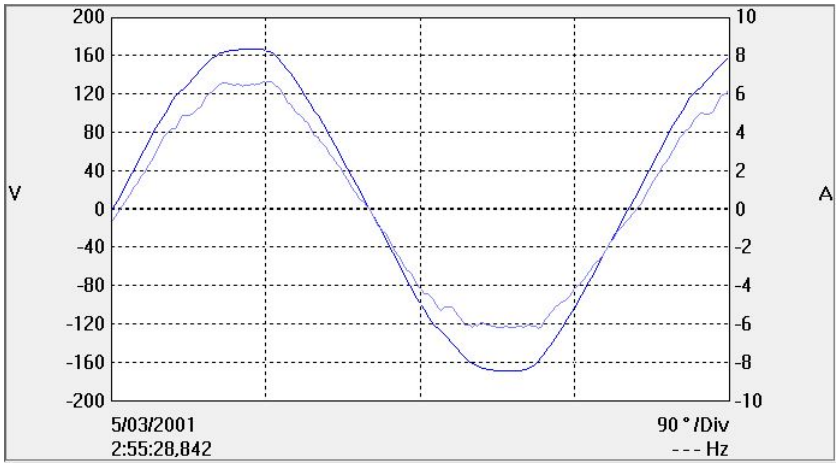

Figura 9.20: Resultados del control sintonizado mediante el algoritmo observados en el analizador.

multiplicados por 0.1 para obtener los valores reales de las magnitudes y posteriormente fueron graficados mediante el software MATLAB® y como resultado se tiene la imagen de la figura 9.21.

En la figura 9.22 se muestra las señales de corriente y voltaje de entrada para cada uno de los 10 puntos de operación diferente al sintonizado por el algoritmo y en el cuadro 9.9 se presentan los resultados de factor de potencia y potencia para cada una de las cargas, en donde se encuentra un valor de factor de potencia de 0.999 para uno de los puntos de operación y un promedio de factor de potencia superior a las pruebas realizadas con las constantes base de las cuales parte el algoritmo.

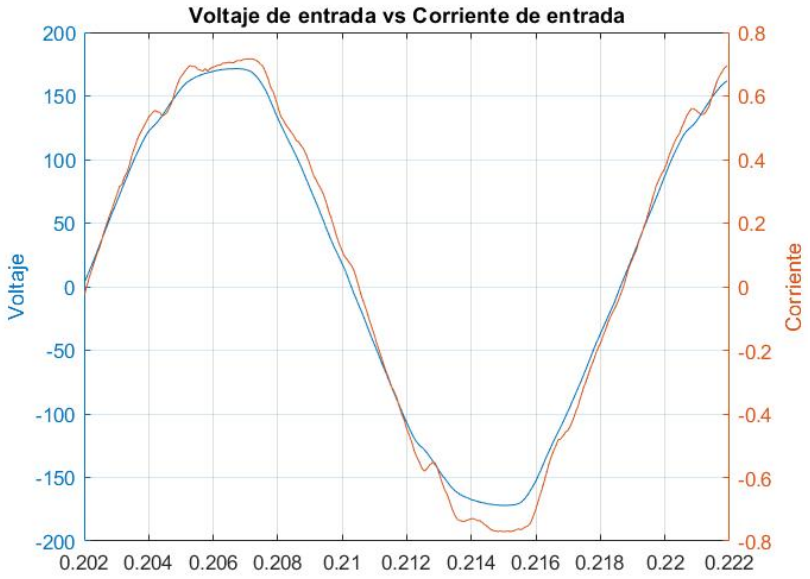

Figura 9.21: Resultados del control sintonizado mediante el algoritmo con valor de corriente real.

Resistencia	***Factor de potencia***	***Potencia (w)***
2.6kΩ	0.987	105.6
2.9kΩ	0.988	98.2
3.3kΩ	0.991	90.3
3.8kΩ	0.993	80.7
4.5kΩ	0.994	74.3
5.3kΩ	0.997	64.8
8.2kΩ	0.999	52.1
10.9kΩ	0.998	44.5
19.3kΩ	0.993	39.0
40.1kΩ	0.978	34.1
Promedio	0.9923	

Cuadro 9.9: Valores de factor de potencia y potencia para cada una de las cargas con algoritmo se selección clonal.

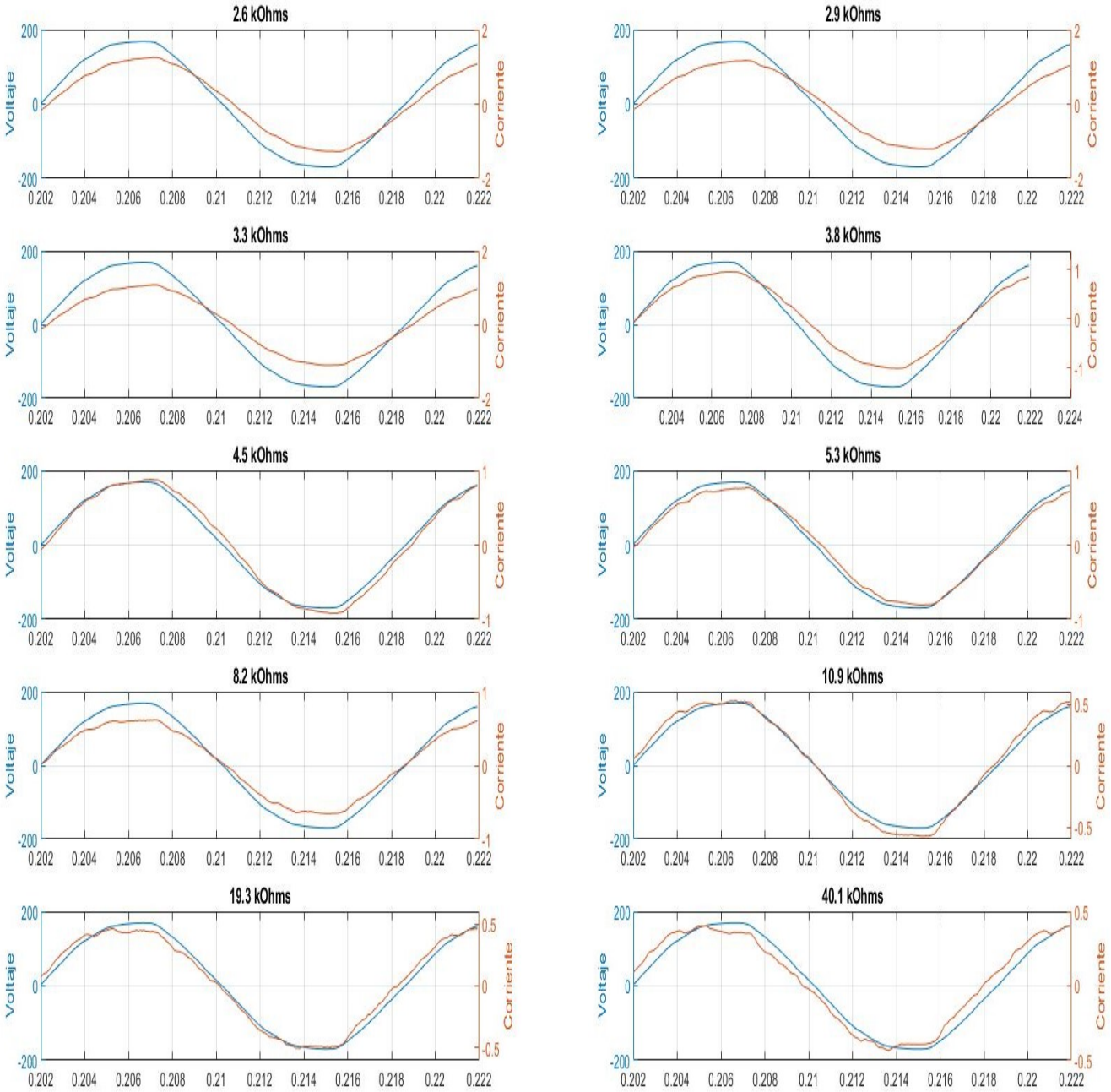

Figura 9.22: Voltaje y corriente en los puntos de operación diferentes al sintonizado.

9.5. Discusión

En los resultados arrojados en la primera prueba se obtuvo un factor de potencia de 0.977 en su primera iteración. En las 200 poblaciones de la segunda iteración no se encontró ningún valor que superé el factor de potencia de la primera iteración, dando un valor de factor de potencia igual a 0.962, por lo tanto el algoritmo dio por terminada la exploración dejando como solución la población encontrada en la primera iteración. Este resultado demuestra que limitarse a una zona tan pequeña para la exploración de nuevas poblaciones hace que el algoritmo se estanque en un mínimo local. En la segunda prueba el porcentaje de mutación se incrementa 100 veces, por lo tanto la exploración de las constantes es mucho mayor. En este caso el algoritmo alcanza un valor de 0.857, 0.991 y 0.99 correspondiente a la primera, segunda y tercera iteración. Como reflejan los resultados ampliar la exploración en las constantes de las poblaciones afecta la rapidez de convergencia, sin embargo el algoritmo logró encontrar soluciones globales con mejor factor de potencia. Como resultado resultado de estas pruebas se puede deducir que para las dos primeras iteraciones el valor de desviación de la prueba 2 es superior al de la prueba 1, explorando así un mayor número de espacios y arrojando resultados con valores de error tanto cercanos al error máximo como valores cercanos al valor esperado, caso contrario con la prueba 1 que mantiene una desviación inferior pero sus valores no alcanzan los mínimos encontrados de la prueba 2, ya respecto a la iteración 3 de la prueba 2 se puede notar que los datos son muy similares entre sí lo cual indica que ha encontrado un mínimo global.

Del cuadro 9.3 se observa que el algoritmo en la prueba 2 encontró mejores valores de factor de potencia, pero entre sus poblaciones hay conjuntos de constantes que incrementan la corriente, lo cual es un punto a tener en cuenta en una implementación online. Por ultimo al sintonizar el filtro precompensador se logra subir el factor de potencia de 0.991 a 0.998. Siendo este el máximo valor obtenido en simulación, en está sintonización se limita el rango en el cual se mueven las constantes ya que hay puntos de operación donde el algoritmo logra desestabilizar la planta. En los resultados experimentales con el prototipo real se prueba el controlador obtenido por el algoritmo, el controlador sintonizado con la función de Matlab PID Tunner y las constantes iniciales del documento [45]. Al comparar las graficas de 9.19 y 9.21 se puede observar que la corriente de entrada presenta desfase en comparación con el voltaje de entrada cuando se utilizan las constantes de PID Tunner. Por otro lado la sintonización con el algoritmo basado en sistemas inmunes artificiales tiene un desfase mucho menor sin embargo presenta una pequeña distorsión en la señal.

En los cuadros 9.8 y 9.9 se utilizan 10 puntos de operación diferentes cambiando la carga desde 2.6kΩ hasta 40.1kΩ. Se obtiene obtiene el factor de potencia en cada punto de operación utilizando las constantes iniciales y las constantes encontradas en la sintonización. Se obtiene el promedio de los factores de potencia donde notamos mayor promedio en los factores de potencia de las constantes sintonizadas. Otro factor a resaltar es que las constantes iniciales no logran llegar a un factor de potencia de 0.999 y por el contrario las constantes sintonizadas si logran este valor. El factor de potencia del controlador sintonizado mediante el algoritmo bioinspirado es superior al planteado en los objetivos para un rango de resistencia de 3.8kΩ hasta 19.3kΩ lo que equivale a 15.5kΩ, mientras que el controlador base mantiene un factor de potencia superior a 0.992 entre 3.3kΩ y 10.9kΩ, eso indica un rango resistivo de 7.6kΩ.

Capítulo 10

Conclusiones

- Se logró mediante el uso del algoritmo basado en la teoría de selección clonal sintonizar los controladores del lazo interno de corriente, con ello obtener una factor de potencia de 0.998 en un convertidor AC/DC topología Half Bridge Boost y mantener un factor superior al propuesto en los objetivos en variaciones de potencia de $39.0w$ a $80.7w$ en el prototipo real de planta.
- Un factor clave del algoritmo fue la población inicial, para este caso en particular un inicio aleatorio no era conveniente debido al numero de constantes a sintonizar y la diferencia de rangos entre ellas, esto haría que el algoritmo tarde mas iteraciones y se deja en evidencia por medio de las pruebas realizadas que inicializar la población desde un único valor tampoco es conveniente porque hay un estancamiento en mínimos locales. Debido a lo anterior, la forma encontrada con la que se obtuvo mejores resultados en un numero de iteraciones inferior es definiendo rangos para cada una de las constantes. Otro de los factores relevantes es la configuración multimodal porque permite el uso de varios datos en memoria para una mayor exploración de espacios, disminuyendo la posibilidad de estancamiento en mínimos locales.
- El tiempo requerido para el cambio de población a evaluar tiene una relación directa con la variable de lectura, en este caso el factor de potencia. Al ser una variable que se obtiene en estado estacionario, se necesita esperar un tiempo aproximadamente de 0.1 a 0.4 segundos dependiendo de la respuesta de cada población. Si no se espera el tiempo suficiente se puede llegar a leer incorrectamente el valor de factor de potencia y con ello guardar en memoria poblaciones con valores de función de aptitud erróneos.
- El gasto computacional en términos de memoria es elevado para el uso de un algoritmo de sintonización automática, esto puede ser un obstáculo a la hora de implementarse en un prototipo de la planta real ya que demandaría una memoria disponible para el algoritmo superior a 12532 Bytes.
- La inicialización de poblaciones en rangos amplios hace que se tenga una mayor exploración de espacios, esto permitiendo tener poblaciones con errores mínimos en la función de actitud, pero también generar poblaciones con valores de error muy altos que alteran variables del sistema como la corriente, siendo esta magnitud un punto a controlar en una posible implementación real ya que ocasionaría daños en los componentes.

Capítulo 11

Trabajos Futuros

Aunque los resultados obtenidos respecto a factor de potencia por el algoritmo basado en la selección clonal son superiores a los encontrados por los métodos convencionales, sería enriquecedor el uso de otros algoritmos con el fin de evaluar si es posible un menor consumo de memoria, menor numero de poblaciones por iteración, menores iteraciones y menores variaciones en corriente.

Otro tema de investigación sería el uso de un algoritmo de sintonización automática para el lazo de externo de voltaje, debió a que si se cambian parámetros del lazo interno de corriente se ve afectado el modelo externo haciendo que el control diseñado en comienzo para el voltaje no corrija el error de forma esperada.

Adicionalmente sería interesante realizar la sintonización del PID y filtro precompensador de manera online con el fin de adaptar los controladores a perturbaciones en tiempo real. Implementar el algoritmo en el prototipo de la planta podrá reducir los tiempos en que se obtienen las constantes. Al realizar está implementación se deberá tener en cuenta

Bibliografía

[1] C. de Regulación de Energía y Gas, "Resolución: Gestión de flujo de potencia reactiva 018," 2005.

[2] D.W.Hart, *RECTIFICADORES DE ONDA COMPLETA Y MEDIA ONDA: Conversión CA-CC,pp.115-169.* second edition ed., 2001.

[3] A. Uan-Zo-li, F. Lee, and R. Burgos, "Modeling, analysis and control design of single-stage voltage source pfc converter," vol. 3, pp. 1684–1691, 2005.

[4] L. Rossetto, G. Spiazz, and P. Tenti, "Control techniques for power factor correction converters," *Paper, University of Padova, Italy*, pp. 1–9, 1994.

[5] A. Martins and A. Cardoso, "Input current distortion and output voltage regulation of the boost pfc converter operating with different control methods," *International Conference on Renewable Energies and Power Quality*, vol. 1, no. 1, pp. 328–333, 2012.

[6] C. electronica Internacional, "Iec 61000-3-2," *Comisión Electrónica Internacional*, 2005.

[7] R. W. Erickson and D. Maksimovic, *Fundamentals of Power Electronics.* second edition ed., 2004.

[8] J. Bayona, H. Chamorro, A. Sanchez, J. Aguillon, and D. Rubio, "Linear control of a power factor correction rectifier in half-bridge configuration," *IEEE CACIDI 2016 - IEEE Conference on Computer Sciences*, pp. 1–6, Buenos Aires, Argentina, 2016.

[9] R. Ghosh and G. Narayanan, "A simple analog controller for single-phase half-bridge rectifier," *IEEE Transactions on Power Electronics*, vol. 22, no. 1, pp. 186–198, 2007.

[10] M. Fernandez, "Modelos no lineales y control en modo deslizante de convertidores de estructura resonante," *Trabajo de grado, Departamento Ingeniería electrónica, Universidad Politecnica de Catalunya*, 1998.

[11] M. Rodriguez, A. Villarreal and O. Serrano, "Asynchronous bio-inspired tuning for the dc motor speed controller with simultaneous identification and predictive strategies," *IEEE Congress on Evolutionary Computation, CEC 2020 - Conference Proceedings*, pp. 1–8, 2020.

[12] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 64, pp. 759–768, 1942.

[13] E. Forero, C. Torres and D. Tibaduiza, "Consideraciones de diseño de un control doble lazo de un convertidor boost para la corrección del factor de potencia," *Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2014)*, pp. 1–9, Guayaquil, Ecuador, 2014.

[14] G. Tulay, I. İskender, and H. Erdem, "Optimal tuning of a boost pfc converter pi controller using heuristic optimization methods," *International Transactions on Electrical Energy Systems*, pp. 1–10, 2017.

[15] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of Global Optimization*, vol. 39, pp. 459–471, 2007.

[16] A. K. Tanuka Bhattacharjee, Sriparna Saha and A. K. Nagar, "Static video summarization using artificial bee colony optimization," *SSCI 2018*, vol. 8, pp. 777–784, 2018.

[17] O. Castillo and L. Amador, "A new fuzzy bee colony optimization with dynamic adaptation of parameters using interval type-2 fuzzy logic for tuning fuzzy controllers," *Soft Computing*, p. 571–594, 2018.

[18] Z. Bingul and O. Karahan, "Comparison of pid and fopid controllers tuned by pso and abc algorithms for unstable and integrating systems with time delay," *Optim Control Appl Meth*, vol. 39, pp. 1431–1450, 2018.

[19] J. Tien and L. Tzuu-Hseng, "Hybrid clonal selection algorithm and the artificial bee colony algorithm for a variable pid-like fuzzy controller design," *International Conference on Fuzzy Theory and its applications*, pp. 87–94, 2012.

[20] A. MUGHEES and S. MOHSIN, "Design and control of magnetic levitation system by optimizing fractional order pid controller using ant colony optimization algorithm," *IEEE Acces*, vol. 8, pp. 116704–116723, 2020.

[21] F. A. S. babu and S. B. Chiranjeevi, "Implementation of fractional order pid controller for an avr system using ga and aco optimization techniques," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 456–461, 2016.

[22] L. de Castro and F. Von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.

[23] M. Qinghua and L. Tingting, "Study on immune pid control method of an in-wheel motor used in an electric car," pp. 9554–9559, 2017.

[24] G. Tsakyridis, N. Xiros, and M. Scharringhausen, "Design and control of a dc boost converter for fuel-cell-powered marine vehicles.," *Journal of Marine Science and Application*, p. 246–265, 2020.

[25] J. Shaik and V. Ganesh, "A power system restoration method using voltage source converter–high-voltage direct current technology, aided by time-series neural network with firefly algorithm," *Soft Computing*, p. 9495–9506, 2020.

[26] J. Romero, D. Paez, B. Noriega, J. Guarnizo, and J. Bayona, "Design and simulation of a voltage control based on neural networks," *2021 IEEE 5th Colombian Conference on Automatic Control (CCAC)*, pp. 133–138, Ibague, Colombia, 2021.

[27] J. Guarnizo, J. Guacaneme, and C. Trujillo, "General inverse neural current control for buck converter," *Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics*, p. 117–122, 2008, doi. 10.1007/978-1-4020-8737-021.

[28] E. Tapoglou, I. Trichakis, Z. Dokou, I. Nikolos, and G. Karatzas, "Groundwater-level forecasting under climate change scenarios using an artificial neural network trained with particle swarm optimization," *Hydrological Sciences Journal*, vol. 59:6, pp. 1225–1239, 2014.

[29] C. C. J. Aguila and C. Vargas, "Particle swarm optimization, genetic algorithm and grey wolf optimizer algorithms performance comparative for a dc-dc boost converter pid controller," *Advances in Science, Technology and Engineering Systems Journal*, vol. 6, pp. 619–625, 2021.

[30] M. Ahmed, I. Mohammed, and A. Younis, "Design and implementation of pso/abc tunned pid controller for buck converters," *Periodicals of Engineering and Natural Sciences*, vol. 9, pp. 641–656, 2021.

[31] S. B. Panduranga Vittal and A. Keshri, "Comparative study of pi, pid controller for buck-boost converter tuned by bio-inspired optimization techniques," *IEEE International Conference on Distributed Comput*, pp. 219–224, 2021.

[32] S. M. S. Mirjalili and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw*, vol. 69, pp. 46–61, 2014.

[33] R. Guha and Benerjee, "Load frequency control of interconnected power system using grey wolf optimization," *Swarm and Evolutionary Computation*, vol. 27, pp. 97–115, 2016.

[34] E. Pereira, P. Bassetto, L. Biuk, M. Itaborahy, A. Converti, M. dos Santos, and H. Valadares, "Swarm-inspired algorithms to optimize a nonlinear gaussian adaptive pid controller," *Energies*, vol. 14, pp. 1–20, 2021.

[35] J. Romero, D. Paez, B. Noriega, J. Guarnizo, and J. Bayona, "Bio-inspired pso technique applied to pid sintonization for powerfactor correction in a boost converter," *2021 IEEE 5th Colombian Conference on Automatic Control (CCAC)*, pp. 139–144, Ibague, Colombia, 2021.

[36] D. M.S.A. and R. M.V.C., "A hybrid genetic algorithm for selective harmonic elimination pwm ac/ac converter control," *Electrical Engineering 89*, p. 285–291, 2007.

[37] K. Aseem and S. Kumar, "Hybrid k-means grasshopper optimization algorithm based fopid controller with feed forward dc–dc converter for solar-wind generating system," *Journal of Ambient Intelligence and Humanized Computing*, p. 2439–2462, 2022.

[38] I. C. Abderrahmen B, Samir M and R. B, "Design and real time implementation of adaptive neural-fuzzy inference system controller based unity single hase power factor converter," *Electric Power Systems Research*, vol. 152, pp. 357–366, 2021.

[39] K. Umamaheswaria and V. Venkatachalamb, "Optimal pf ccorrector of single stage power converter using bc tuned pid controller," *Journal of Intelligent Fuzzy Systems*, vol. 30, pp. 3155–3166, 2016.

[40] M. Ali, L. Wang, and G. Chen, "Control system optimization of three-phase active rectifier based on evolutionary algorithm," *The 4th International Conference on Power and Renewable Energy*, pp. 212–216, 2019.

[41] R. O. Ramesh Srinivasan, "A unity power factor converter using half-bridge boost topology," *IEEE Transactions on Power Electronics*, vol. 13, pp. 1–2, 1998.

[42] A. Pereira, J. Vieira, and L. de Freitas, "A lossless switching forward converter with unity power factor operation," vol. 1, pp. 329–334, 1995.

[43] G. C. Kiam Heong Ang and Y. Li, "Pid control system analysis, design, and technology," *IEEE Transactions on Control Systems Technology*, vol. 4, pp. 559–576, 2005.

[44] J. Navarrete, "Diseño y evaluación de un rectificador en medio puente con factor de potencia unitario," *Trabajo de grado, Ingeniería electrónica, Universidad Santo Tomas*, 2014.

[45] N. D. F Bolivar and J. Bayona, "Diseño e implementación de un controlador digital tipo pid con pre-compensación para un boost pfc de medio puente," *Revista UIS Ingenierías*, vol. 19, pp. 179–192, 2020.

[46] M. M. J.F. Avila and V. Quesada, "La inteligencia artificial y sus aplicaciones en medicina i: introducción antecedentes a la ia y robótica," *Atención Primaria*, vol. 52, pp. 778–784, 2020.

[47] R. Pino, A. Goméz, and N. Martínez, *Introducción a la inteligencia artificial: sistemas expertos, redes neuronales artificiales y computación evolutiva.* 2001.

[48] J. Herrera, *Sistema Inmune Artificial con Población Reducida para Optimización Numérica.* PhD thesis, Instituto politécnico nacional, Centro de Investigación en computación , México D.F, 2011.

[49] M. S. J. Monserrat, A. Gomez and A. Prieto, "Introducción al sistema inmune. componentes celulares del sistema inmune innato," *Medicine - Programa de Formación Médica Continuada Acreditado*, vol. 12, pp. 1369–1378, 2017.

[50] H. Barcenilla, A. Prieto, M. Monserrat, D. Díaz, E. Reyes, and M. Álvarez, "Respuesta inmune adaptativa o antígeno específica," *Medicine - Programa de Formación Médica Continuada Acreditado*, vol. 10, pp. 1868–1879, 2009.

[51] J. Guarnizo and L. Nino, "Clonal selection algorithm applied to object recognition in mobile robots," *15th International Conference on Machine Learning and Data Mining, MLDM 2019*, vol. 1, pp. 50–62, 2019.

[52] N. Cruz, *Sistema inmune artificial para solucionar problemas de optimización.* PhD thesis, ingeniería eléctrica sección de computación, centro de investigación y de estudios avanzados del instituto politécnico nacional, México D.F, 2004.

[53] L. Carrasco, "Implementación del algoritmo de scan-matching basado en clonalg," *Trabajo de grado, Ingeniería electrónica industrial y automática, Universidad Carlos III de Madrid*, 2015.

[54] I. d. A. I. Chauvin Arnoux®, *Power Quality Analyzer Model 3945-B.* Chauvin Arnoux®, Inc. d.b.a. AEMC® Instruments, 15 Faraday Drive Dover, USA.

ANEXOS

11.1. Algoritmo sintonización controlador PID

```
global cont
global pf1
global poblacion
global kp1
global ki1
global kd1
global f1
global po_mutada
global iteracion_alg_inm
global funcion_aptitud
global memory_pid
global contSample
global memorySamples
global boolInitPopulation

pert_1=zeros(1,4);
if iteracion_alg_inm < 8

    if boolInitPopulation == true
        [boolInitPopulation, po_mutada] =...
            initPopulation(boolInitPopulation)
    else
        if contSample == 6
            contSample=1;
        end
        memorySamples = ingestSample(contSample, pf, memorySamples)
        contSample = contSample + 1;
        cont = cont + 1;
        contSample
        cont
    end

    if completeSamples(memorySamples) == true
        if evaluateSamples(memorySamples, cont) == true...
                && poblacion == 201 %%Cambio de iteracion
            cont = uint32(0);
            index=zeros(20,1);
            best=zeros(20,4);
```

```
    for i = 1:20
        [M, I] = min(funcion_aptitud);
        if I==1
            funcion_aptitud(I)=1000;
            [M, I] = min(funcion_aptitud);
        end
        index(i,:) = M;
        best(i,:) = po_mutada(I-1,:);
        funcion_aptitud(I)=1000;
    end
    if evaluateMemory(index, memory_pid([1:20],1))==true
            %%Terminación del algoritmo
        po_mutada(1,:) = memory_pid(1, [2 3 4 5])
        kp1 = po_mutada(1, 1);
        ki1 = po_mutada(1, 2);
        kd1 = po_mutada(1, 3);
        f1 = po_mutada(1, 4);
        iteracion_alg_inm = double(8);
        poblacion = uint8(1);
        funcion_aptitud=zeros(200,1);
        pf1=zeros(200,1);
        po_mutada(1, :)
    else
        rng('shuffle');
        memory_pid = replaceMemory(index, memory_pid, best);
        index = memory_pid([1:20],1);
        best = memory_pid([1:20],[2 3 4 5]);
        for i = 1:20
            for j = (10*i-9):(10*i)
                for k = 1:4
                    r=-best(i,k)*index(i)/iteracion_alg_inm...
                        +(best(i,k)*index(i)/iteracion_alg_inm...
                        + best(i,k)*index(i)/iteracion_alg_inm)*rand(1,1);
                    pert_1(k)=best(i,k)+r;
                end
                po_mutada(j,:)=pert_1';
            end
        end
        funcion_aptitud=zeros(200,1);
        pf1=zeros(200,1);
        index
        best
        poblacion = uint8(1);
        iteracion_alg_inm = iteracion_alg_inm + 1;
    end
elseif evaluateSamples(memorySamples, cont) == true
        %%Cambio de poblacion
    kp1 = po_mutada(poblacion,1);
    ki1 = po_mutada(poblacion,2);
    kd1 = po_mutada(poblacion,3);
```

```
                f1 = po_mutada(poblacion,4);
                error_fp = 1 - pf;
                pf1(poblacion) = pf;
                funcion_aptitud(poblacion) = error_fp;
                poblacion = poblacion +1;
                pf1
                funcion_aptitud
                poblacion
                po_mutada
                iteracion_alg_inm
                contSample = double(1)
                cont = uint32(0);
                memorySamples = zeros(5,1)
            end
        end
    end

function boolEvaluateMemory = evaluateMemory(index, memory_pid)
    if index(1)>=memory_pid(1)
        boolEvaluateMemory = true
        return
    end
    boolEvaluateMemory = false
end

function boolEvaluateSample =  evaluateSamples(memorySamples, cont)
    if cont==10
        boolEvaluateSample = true
        return
    end
    for i= 1:4
        if abs(memorySamples(i)-memorySamples(i+1)) > 1e-3
            boolEvaluateSample = false
            return
        end
    end
    boolEvaluateSample = true
end

function boolCompleteSamples = completeSamples(memorySamples)
    for i = 1:5
        if memorySamples(i) == 0
            boolCompleteSamples=false
            return
        end
    end
    boolCompleteSamples=true
end

function [bool, population] = initPopulation(boolInitPopulation)
```

```
    rng('shuffle');
    kp=(29.5)*rand(200,1);
    ki=(128.1e3)*rand(200,1);
    kd=(7.6e-4)*rand(200,1);
    f=(59.56e5)*rand(200,1);
    population = [kp ki kd f];
    bool=~boolInitPopulation;
end

function samples = ingestSample(contSample, sample, memorySamples)
    memorySamples(contSample)=sample
    samples=memorySamples
end

function memory = replaceMemory(index, memoryPid, best)
    for i=1:20
        if index(i) < memoryPid(i,1)
            memoryPid(i,:)=[index(i) best(i,:)];
        end
    end
    memory = memoryPid;
end
```

11.2. Algoritmo sintonización filtro precompensador

```
    global cont
    global pf1
    global poblacion
    global Num_com
    global den_com
    global po_mutada_com
    global funcion_aptitud
    global iteracion_alg_inm
    global memory_pre
    global contSamplePre
    global memorySamplesPre
    global boolInitPopulationPre

    pert_1_com=zeros(1,2);
    if iteracion_alg_inm > 7 && iteracion_alg_inm < 13

        if boolInitPopulationPre == true
            [boolInitPopulationPre, po_mutada_com] = ...
                initPopulation(boolInitPopulationPre)
        else
            if contSamplePre == 6
                contSamplePre=1;
            end
            memorySamplesPre =....
```

```
        ingestSample(contSamplePre, pf, memorySamplesPre)
    contSamplePre = contSamplePre + 1;
    cont = cont + 1;
    memorySamplesPre
    cont
end

if completeSamples(memorySamplesPre) == true
    if evaluateSamples(memorySamplesPre, cont) == true...
            && poblacion == 201 %%Cambio de iteracion
        cont = uint32(0);
        index=zeros(20,1);
        best=zeros(20,2);
        for i = 1:20
            [M, I] = min(funcion_aptitud);
            if I==1
                funcion_aptitud(I)=1000;
                [M, I] = min(funcion_aptitud);
            end
            index(i,:) = M;
            best(i,:) = po_mutada_com(I-1,:);
            funcion_aptitud(I)=1000;
        end
        if evaluateMemory(index, memory_pre([1:20],1))==true
            %%Terminación del algoritmo
            [M, I] = min(funcion_aptitud);
            if I==1
                funcion_aptitud(I)=1000;
                [M, I] = min(funcion_aptitud);
            end
            po_mutada_com(1,:)=...
                bestInMemory([M po_mutada_com(I-1, :)], memory_pre)
            Num_com = po_mutada_com(1,1);
            den_com =[1 po_mutada_com(1,2)]';
            iteracion_alg_inm = double(14);
            poblacion = uint8(1);
            funcion_aptitud=zeros(200,1);
            pf1=zeros(200,1);
            po_mutada_com(1,:)
        else
            rng('shuffle');
            memory_pre = replaceMemory(index, memory_pre, best);
            index = memory_pre([1:20],1);
            best = memory_pre([1:20],[2 3]);
            for i = 1:20
                for j = (10*i-9):(10*i)
                    if j == (10*i-9)
                        pert_1_com = best(i,:);
                    else
                        for k = 1:2
```

```
                                    r=-best(i,k)*index(i)/iteracion_alg_inm...
                                        +(best(i,k)*index(i)/iteracion_alg_inm...
                                        + best(i,k)*index(i)/iteracion_alg_inm)*rand(1,1);
                                    pert_1_com(k)=best(i,k)+r;
                                end
                            end
                            po_mutada_com(j,:)=pert_1_com';
                        end
                    end
                    funcion_aptitud=zeros(200,1);
                    pf1=zeros(200,1);
                    index
                    best
                    poblacion = uint8(1);
                    iteracion_alg_inm = iteracion_alg_inm + 1;
                end
            elseif evaluateSamples(memorySamplesPre, cont) == true
                    %%Cambio de poblacion
                Num_com = po_mutada_com(poblacion,1);
                den_com = [1 po_mutada_com(poblacion,2)]';
                error_fp = 1 - pf;
                pf1(poblacion) = pf;
                funcion_aptitud(poblacion) = error_fp;
                poblacion = poblacion +1;
                pf1
                funcion_aptitud
                poblacion
                po_mutada_com
                iteracion_alg_inm
                contSamplePre = double(1)
                cont = uint32(0);
                memorySamplesPre = zeros(5,1)
            end
        end
    end

function boolEvaluateSample =  evaluateSamples(memorySamples, cont)
    if cont==10
        boolEvaluateSample = true
        return
    end
    for i= 1:2
        if abs(memorySamples(i)-memorySamples(i+1)) > 1e-3
            boolEvaluateSample = false
            return
        end
    end
    boolEvaluateSample = true
end
```

```
function boolEvaluateMemory = evaluateMemory(index, memory_pre)
    if index(1)>=memory_pre(1)
        boolEvaluateMemory = true
        return
    end
    boolEvaluateMemory = false
end

function boolCompleteSamples = completeSamples(memorySamples)
    for i = 1:5
        if memorySamples(i) == 0
            boolCompleteSamples=false
            return
        end
    end
    boolCompleteSamples=true
end
function [bool, population] = initPopulation(boolInitPopulation)
    rng('shuffle');
    num=(0.401)*rand(200,1);
    dem=-0.9999 + (0.9999)*rand(200,1);
    population=[num dem];
    bool=~boolInitPopulation;
end

function samples = ingestSample(contSample, sample, memorySamples)
    memorySamples(contSample)=sample
    samples=memorySamples
end

function memory = replaceMemory(index, memoryPre, best)
    for i=1:20
        if index(i) < memoryPre(i,1)
            memoryPre(i,:)=[index(i) best(i,:)];
        end
    end
    memory = memoryPre;
end

function best = bestInMemory(bestIteration, memoryPre)
    if bestIteration(1,1) > memoryPre(1,1)
        best = bestIteration(1, [2 3]);
        return
    end
    best = memoryPre(1, [2 3]);
end
```