Tesis de Maestría

Supernova: un algoritmo novedoso de optimización global

Eddy Mesa Delgado, Ing.

UNIVERSIDAD
NACIONAL
DE COLOMBIA
SEDE MEDELLÍN

Tesis de maestría presentada como requisito para optar al título de magister en ingeniería de sistemas

Director de Tesis

Prof. Juan David Velásquez Henao , Ph.D.

Co-director de Tesis

Prof. Gloria Patricia Jaramillo , Ph.D.

Universidad Nacional de Colombia
Facultad Nacional de Minas
Escuela de Sistemas
Programa Maestría en Ingeniería de Sistemas - Investigación de Operaciones

2010

A Dios,

A mis padres,

Y a los ángeles que me acompañan: corpóreos y etéreos.

AGRADECIMIENTOS

Agradezco de todo corazón a Dios y a todos aquellos que hicieron posible la culminación de este trabajo. Es imposible nombrarlos extensivamente a todos, pero se mencionan por su orientación y conceptos en diferentes niveles a los profesores Juan David Velásquez y Gloria Patricia Jaramillo, al Dr. Frank Brand, M Sc. Fernán Villa, M Sc. Carlos Garavito, Antonio Sánchez y Javier Cano. Por su ayuda y apoyo a mi familia y amigos que estuvieron presentes en las diferentes etapas.

La optimización se utiliza en diversas áreas: ciencias aplicadas, ingeniería, economía y muchas más. De las diversas formas de abordar el problema de optimización, las metaheurísticas han mostrado la estrecha relación existente entre los diversos procesos del entorno y la solución del problema de optimización; la analogía de procesos como la evolución provee elementos que permiten la solución rápida y eficiente de problemas complejos. En este trabajo se propone una heurística novedosa y funcional, para resolver problemas de optimización no lineal sin restricciones, basada en el fenómeno de las supernovas. Esta heurística se denominó Supernova. Inicialmente se plantearon las reglas estratégicas para la heurística. Seguidamente se implementó y probó su convergencia inicial. Luego, se validaron los resultados obtenidos comparando el desempeño de Supernova con tres métodos heurísticos (evolución diferencial adaptativa, evolución diferencial y programación evolutiva más rápida) en problemas benchmark utilizados para la prueba de métodos de optimización global. Finalmente, se aplicó Supernova al problema de estimación de parámetros para redes neuronales artificiales tipo cascada correlación. Supernova mostró ser robusta y también un desempeño superior en algunos de los problemas teóricos evaluados y para el problema de aplicación.

Palabras clave: Optimización no lineal, metaheurísticas, algoritmos bioinspirados, Supernova.

Key Words: Nonlinear optimization, metaheuristics, bioinspired algorithms, Supernova.

ÍNDICE GENERAL

Resumen VII

1. Problemática de la optimización global con metaheurísticas 1

1.1. Las Supernovas . . . 1

1.2. El problema de optimización . . . 2

1.2.1. Definición . . . 2

1.2.2. Clasificaciones . . . 3

1.2.3. Optimización global . . . 3

1.3. Métodos de optimización . . . 5

1.3.1. Optimización lineal. . . . 5

1.3.2. Optimización no lineal. . . . 5

1.4. Metaheurísticas . . . 7

1.4.1. Técnicas inspiradas en la evolución . . . 9

1.4.2. Técnicas inspiradas en los procesos físicos . . . 9

1.4.3. Técnicas inspiradas en el aprendizaje . . . 10

1.5. Dificultades del problema de optimización global sin restricciones . . . 10

1.5.1. Respecto a la optimización global . . . 10

1.5.2. Sensibilidad al punto de inicio . . . 10

1.5.3. Precisión . . . 11

1.5.4. Complejidad . . . 11

1.6. Discusión . . . 13

1.7. Delimitación del problema . . . 14

1.8. Objetivos . . . 15

1.8.1. Objetivo General . . . 15

1.8.2. Objetivos Específicos . . . 15

1.9. Organización del documento . . . 15

1.10. Divulgación de los resultados . . . 15

2. Supernova 17

2.1. Descripción del fenómeno . . . 17

2.2. Estrategia . . . 19

2.2.1. Explosión . . . 19
2.2.2. Escalamiento . . . 21
2.2.3. Cálculos de fuerzas atractivas y repulsivas . . . 21
2.2.4. Reinicialización . . . 22
2.3. El algoritmo . . . 22
2.3.1. Explosión . . . 22
2.3.2. Evaluación . . . 23
2.3.3. Cálculos . . . 23
2.3.4. Reinicio . . . 24
2.3.5. Pseudocódigo . . . 24
2.4. Similitudes y diferencias con otras metaheurísticas . . . 25
2.5. Conclusiones . . . 26

3. Implementación y parametrización **27**
3.1. ¿Qué evaluar? . . . 27
3.1.1. Implementación . . . 28
3.1.2. Parametrización . . . 29
3.2. Funciones de prueba utilizadas . . . 29
3.3. Implementación . . . 32
3.3.1. Proceso de búsqueda . . . 32
3.3.2. Estrategia de exploración y estrategia de explotación . . . 32
3.3.3. Robustez . . . 35
3.3.4. Costo computacional . . . 38
3.4. Parametrización . . . 38
3.4.1. Número de partículas . . . 38
3.4.2. Parámetros de Impulso y atracción . . . 40
3.4.3. Número de iteraciones y reinicios . . . 40
3.4.4. Fuerza de la explosión y factor de disminución . . . 41
3.4.5. Parámetros . . . 41
3.5. Conclusiones . . . 41

4. Validación **45**
4.1. Funciones de prueba . . . 45
4.2. Resultados . . . 46
4.3. Conclusiones . . . 50

5. Caso de aplicación en predicción de series de tiempo **51**
5.1. Optimización en redes cascada correlación 51
5.2. Manejo de restricciones 53
5.3. Casos de Aplicación 54
5.3.1. Parametrización del algoritmo 54
5.3.2. Caso de estudio: Pasajeros de una Aerolínea 55
5.4. Caso de aplicación: Linces canadienses 56
5.5. Conclusiones 58

6. Conclusiones **59**

I Anexos **67**

A. Algoritmos Bio-Inspirados **69**
A.1. Computación evolutiva (EC, del inglés Evolutionary computation) 69
A.1.1. Algoritmos Genéticos: (GA del inglés Genetic Algorithms): 69
A.1.2. Programación Evolutiva (EP del inglés Evolutionary Programming): 70
A.1.3. Estrategias Evolutivas (ES del inglés Evolution Strategies): 70
A.1.4. Evolución Diferencial (DE del inglés Differencial evolution): 71
A.2. Algoritmos meméticos (MA del inglés Memetic Algorithm): 71
A.3. Búsqueda tabú (TS del inglés tabu search): 71
A.4. Temple simulado (SA del inglés simulated annealing): 72
A.5. Búsqueda de dispersa (SS del inglés Scatter Search): 73
A.6. Optimización de enjambre de partículas (PSO del inglés Particle swarm optimization): 73
A.7. Optimización de colonia de hormigas (ACO del inglés Ant Colony optimization) 74
A.8. Sistema inmune artificial (AIS del inglés Artificial Immune Systems) 74
A.9. GRASP (del inglés Greedy Randomized Adaptive Search Procedures) 75
A.10.Quimiotaxisis bacteriana (BC del inglés Bacterial chemotaxis): 75
A.11. Gran explosión-Gran implosión (BB-BC del inglés Big Bang-Big Crunch) 76

ÍNDICE DE CUADROS

1.1. Número de artículos por técnicas bio-inspirada 8
1.2. Equivalencias entre la teoría de evolución y el problema de optimización 9

2.1. Equivalencias entre el modelo físico y el modelo de optimización 19

3.1. Funciones de prueba para las etapas de implementación y parametrización 30
3.2. Resultados para los diferentes puntos de inicio y funciones de prueba 36
3.3. Promedio y desviación para la traslación de la funciones en el eje z 37
3.4. Promedio y desviación para la traslación de la funciones en el eje x y y 37
3.5. Tiempo de corrida de acuerdo a las dimensiones y número de partículas para f_1, f_2 y f_3 . . . 38
3.6. Tres mejores promedios encontrados para los parámetros F y G para la funciones de prueba 41
3.7. Valor promedio encontrado para diferente número de iteraciones y reinicios 42
3.8. Parámetros λ y α recomendados para Supernova según el tamaño de la región de búsqueda 43
3.9. Parámetros recomendados para las funciones 43

4.1. Funciones unimodales 46
4.2. Funciones multimodales para muchas dimensiones 46
4.3. Funciones multimodales para pocas dimensiones 47
4.4. a_{ij} para la función trincheras f_{14} 47
4.5. a_{ij} para la función de Kowalik f_{15} 47
4.6. a_{ij}, c_i y p_{ij} para la función de Hartman f_{19} 47
4.7. a_{ij}, c_i y p_{ij} para las funciones de Shekel f_{21},f_{22},yf_{23} 48
4.8. a_{ij}, c_i y p_{ij} para la función de Hartman f_{20} 48
4.9. Resultados para los diferentes métodos de optimización utilizados 49

5.1. Parámetros utilizados para Supernova los casos de aplicación 55
5.2. Valores del SSE para diferentes modelos pronosticando la serie del primer caso 55
5.3. Porcentajes de mejora de Redes CASCOR estimadas con Supernova sobre DAN2. 56
5.4. Valores del error cuadrático medio para diferentes modelos pronosticando la serie de los linces canadiences. 58

ÍNDICE DE FIGURAS

1.1. Secuencia del fenómeno 2
1.2. Clasificación general de los problemas de optimización. 4
1.3. Métodos de optimización de acuerdo al tipo de problema. 6
1.4. Metaheurísticas de acuerdo a su inspiración. 8
1.5. Función de Griewank 11
1.6. Trayectoria de búsqueda con distintos pasos en la función de Rosenbrock 12
1.7. Complejidad para diferentes funciones 12
1.8. Aumento de la complejidad con las dimensiones en la función Rosenbrock 13

2.1. Distribución de partículas para la función esférica 20
2.2. vectores de la interacción entre partículas 22

3.1. Comportamiento asintótico. 28
3.2. Funciones de prueba. 31
3.3. Proceso de búsqueda función esférica f_1 33
3.4. Resultados para la función sin reinicios 34
3.5. Resultados solo reiniciando en el mejor punto 34
3.6. Resultados para la heurística completa 35
3.7. Dimensiones Vs. resultados para supernova para la función esférica 36
3.8. Resultados de las funciones de prueba para el número de partículas. 39
3.9. Resultado promedio de las funciones de prueba para F y G. 40
3.10. Resultado promedio del mínimo encontrado para las funciones de prueba para diferentes valores de λ y α. 43

5.1. Etapas de aprendizaje incremental red cascada correlación 52
5.2. Penalización de la función 54
5.3. Predicción con una red CASCOR optimizada con Supernova y ConRprop para la serie de pasajeros de una aerolínea. 57
5.4. Predicción con una red CASCOR para la serie de tiempo de Linces Canadienses 57

ÍNDICE DE PSEUDOCÓDIGOS

1. Pseudocódigo Supernova 25
2. Pseudocódigo computación evolutiva 69
3. Pseudocódigo Algoritmos meméticos 71
4. Pseudocódigo Búsqueda Tabú 72
5. Pseudocódigo Temple simulado 72
6. Pseudocódigo Búsqueda dispersa 73
7. Pseudocódigo Optimización de enjambre de partículas 74
8. Pseudocódigo Optimización de colonia de hormigas 74
9. Pseudocódigo GRASP 75
10. Pseudocódigo Quimotaxisis Bacteriana 76
11. Pseudocódigo Gran explosión- Gran implosión 76

1. PROBLEMÁTICA DE LA OPTIMIZACIÓN GLOBAL CON METAHEURÍSTICAS

El mundo es un lugar de recursos limitados y necesidades crecientes, la búsqueda de obtener más con menos es apremiante, y es a partir de este concepto, donde toma relevancia la optimización como disciplina. Cada día surgen nuevos problemas que deben ser resueltos de la mejor manera sin desperdiciar recursos y en el menor tiempo posible, llevando, de esta manera, a evolucionar los métodos de solución existentes e incluso crear nuevos métodos que tengan mejores características o solucionen problemas particulares más eficientemente.

En este capítulo inicialmente se describe el fenómeno de las supernovas de cuyo símil puede desarrollarse una heurística. También se presentan, de manera general, el problema de optimización y las formas como se ha abordado este problema con metaheurísticas; se analizan las dificultades y se presenta como esta idea puede llegar a ser una buena estrategia para la solución del problema de optimización. Finalmente, se delimita el trabajo de investigación y se plantean sus objetivos.

1.1. LAS SUPERNOVAS

Las supernovas, fenómenos estelares de gran luminosidad, han causado curiosidad desde la antigüedad. El estudio del fenómeno y las diversas técnicas desarrolladas por la astrofísica han permitido determinar que las supernovas son el destino final de algunas estrellas. El principio del fin para una de estas estrellas se inicia cuando su combustible interno, el hidrógeno, se comienza a agotar creando una inestabilidad interior que produce una explosión de gran magnitud (Carroll y Ostlie, 2007). Luego de la explosión, las partículas expulsadas interactúan entre ellas y con otros cuerpos cercanos su trayectoria, donde la atracción gravitatoria aumenta entre las partículas y cuerpos a medida que pierden su impulso (Alonso y Finn, 1995).

En la Figura 1.1 se muestra la secuencia del fenómeno en cuatro etapas sencillas: inicialmente hay un núcleo inestable inicial que explota y expulsa toda clase de partículas a grandes velocidades. Estas partículas interactúan entre ellas y llegan a un estado de equilibrio y orden donde las partículas y cuerpos de mayor masa son orbitados por otros de menor masa de acuerdo a la distancia entre ellos. Una heurística que emule las etapas del fenómeno de las supernovas podrían encontrar regiones importantes dentro una región de búsqueda pues, las soluciones se aglomerarían alrededor de mejores soluciones por la atracción que ejercen estas sobre otras no tan buenas, tal como sucede con las partículas expulsadas de las supernova. Ésta es la idea que fundamenta la investigación realizada en esta tesis de maestría; la pregunta de investigación planteada es: ¿Cómo platear una analogía computacional basada en el fenómeno de la supernova que permita revolver problemas de optimización?

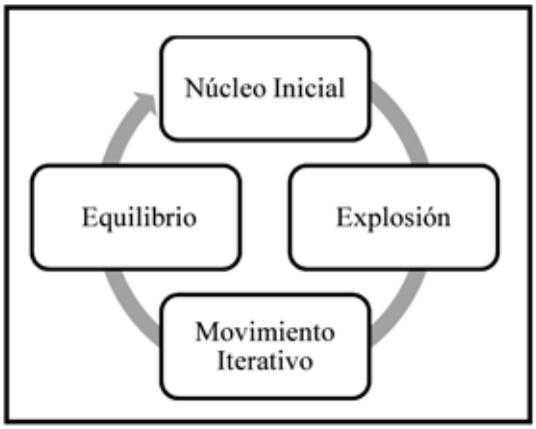


Figura 1.1: Secuencia del fenómeno

1.2. EL PROBLEMA DE OPTIMIZACIÓN

1.2.1. Definición

Las primeras ideas de optimización se remontan a los inicios del cálculo diferencial pero, sólo hasta los años 40's se unen en un área específica con el desarrollo de la programación lineal. Esta área ha tenido un rápido desarrollo posterior en lo que se refiere a métodos e implementaciones para la solución de problemas de optimización (Pierre, 1986).

La optimización se aplica al minimizar o maximizar, costos, tiempo, riesgos, ganancias, calidad, eficiencia entre otros. De estas aplicaciones básicas pueden derivarse innumerables clases de problemas diferentes entre sí. Muchos de estos problemas se han abordado con éxito, pero, aún así, hay problemas en el mundo real que no han sido solucionados por los métodos existentes; y otros, que aunque teóricamente pueden ser solucionados por los métodos existentes, el tiempo y costos necesarios para resolverlos los hace inviables desde la práctica (Pierre, 1986; Nash y Sofer, 1996; Talbi, 2009).

El planteamiento general del problema de optimización abordado en este trabajo es:

$$\text{mín } f(x) \tag{1.1}$$

Sujeta a:

$$\left.\begin{array}{c} Ax \leq B \\ Cx = D \\ LB \leq x \leq UB \\ G(x) > 0 \\ H(x) = 0 \end{array}\right\} \tag{1.2}$$

En (1.1), $f(x)$ es la función a optimizar o función objetivo donde x es un vector en $\mathbb{R}^n$. Las ecuaciones (1.2) representan las restricciones del problema, donde A y C son matrices de orden $m \times n$, UB y LB son vectores numéricos de $n \times 1$ que definen los límites superior e inferior del vector x y $G(x)$ y $H(x)$ son funciones vectoriales no lineales (Rao, 1996).

El problema de optimización descrito por (1.1) y (1.2) puede tener diferentes niveles de complejidad dependiendo del tipo de restricciones dadas (lineales, no lineales, ecuaciones, inecuaciones) y se resuelve de diferente forma dependiendo de dicha complejidad.

1.2.2. Clasificaciones

Existen diferentes tipos de problemas de optimización de acuerdo con las características mostradas en las Ecuaciones (1.1) y (1.2); en la Figura 1.2 se muestran la clasificación de cada tipo de problema de acuerdo a la definición presentada en la sección anterior. A continuación se definen las características que permiten clasificar los diferentes tipos de problema.

- El problema puede o no tener restricciones.
- Si la función o funciones objetivo y las restricciones son lineales se dice que el problema es lineal; en caso contrario es no lineal.
- Las funciones objetivo pueden ser una o varias; cuando hay más de una función objetivo se llama optimización multiobjetivo.
- Las variables pueden ser toda continuas, todas discretas o mixtas; se definen los problemas como continuos, combinatorios o mixtos.

Cada una de estas variantes generales dan paso a un tipo de problema diferente, que a su vez tienen subtipos de problemas cada vez más específicos; por ejemplo, las ecuaciones cuadráticas son un tipo específico dentro de la clasificación de optimización no lineal que presenta características particulares (Pierre, 1986; Rao, 1996).

La amplia gama de problemas descritos impiden que un único método de solución resuelva de manera eficiente; es decir, que cada método está diseñado para resolver cierto tipos de problemas específicos de manera eficiente y, en particular, un método diseñado para cierto problema específico podrá solucionarlo mejor que otros métodos más generales (Pierre, 1986; Nash y Sofer, 1996; Wolpert y Macready, 1997).

1.2.3. Optimización global

En muchas ocasiones, las funciones tienen más de un mínimo en la región de soluciones fatibles; los mínimos para una subregión de A se denominan mínimos locales, es decir, pueden encontrarse varios míni-

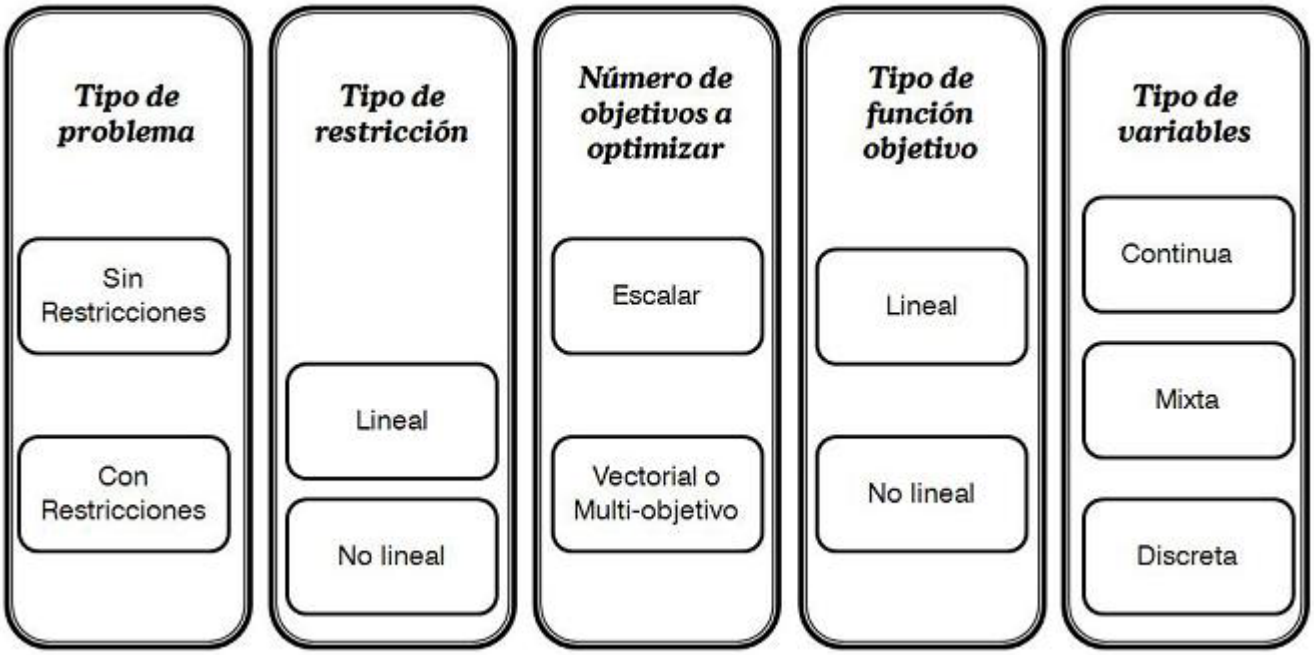


Figura 1.2: Clasificación general de los problemas de optimización.

mos para una misma región a los que el método puede llegar. El problema de optimización global se define formalmente como (Bäck, 1996) :

Dada una función $f : A \subseteq \mathbb{R}^n \to \mathbb{R}, A \neq 0$, *para* $\vec{x^*} \in A$ *el valor de* $f^* := f(\vec{x^*}) > -\infty$ *se puede llamar mínimo global, si y solo si*

$$\forall \vec{x} \in A : f(\vec{x^*}) \leq f(\vec{x}) \tag{1.3}$$

entonces, $\vec{x}$ *es un mínimo global,* f *es la función objetivo, y el conjunto A es la región factible. El problema de determinar un punto de mínimo global es llamado problema de optimización global.*

Esta definición puede extenderse al problema de optimización global sin restricciones donde $A \subseteq \mathbb{R}^n$; es importante aclarar que aunque en este caso se define la región factible como los reales, otros autores tienen una visión menos estricta al respecto y toman como un problema de optimización sin restricciones a aquellos problemas delimitados por bordes UB y LB. Aún así, las regiones acotadas; al ser continuas, siguen siendo espacios de búsqueda infinitos (Vesterstrom y Thomsen, 2004; Brest et al., 2006). En este trabajo se tomará el problema de optimización no lineal sin restricciones con el enfoque de regiones acotadas; eso es, se impone la restricción $LB \leq x \leq UB$.

1.3. MÉTODOS DE OPTIMIZACIÓN

El problema de optimización ha sido abordado desde diferentes perspectivas. En las siguientes secciones se describirán los principales enfoques desarrollados de acuerdo al tipo de problemas.

1.3.1. Optimización lineal.

También llamada programación lineal; en este caso, las todas la funciones del problema deben ser lineales. El método más ampliamente aplicado a este tipo de problemas es el Simplex; este método permite abordar problemas con varias funciones objetivo, restricciones lineales de igualdad e inecuaciones y varias variables. El método, como tal, ha evolucionado desde sus inicios incluyendo varias técnicas, lo que lo hace más eficiente (Pierre, 1986; Nash y Sofer, 1996).

1.3.2. Optimización no lineal.

La mayoría de problemas reales tienen características de no linealidad. A su vez, la no linealidad puede ser descrita de muchas maneras de acuerdo a su forma (polinómica, no polinómica); de acuerdo a su grado de no linealidad (cuadrática, cúbica, etc); de acuerdo al número de mínimos: unimodal o multimodal; de acuerdo a su geometría convexa o no convexa. Cada característica de la función enmarca un problema diferente que requiere diferentes métodos de solución y se pueden clasificar en dos grandes grupos: métodos diferenciales y no diferenciales (Nash y Sofer, 1996; Pierre, 1986; Jang et al., 1996).

Feoktistov (2006) presenta una clasificación de los métodos según el tipo de variable que describe muy bien la problemática y como se ha solucionado dando una visión más global de la relación entre los tipos de problemas de optimización y la forma de resolverlos. Esta clasificación se puede observar en la Figura 1.3: los métodos diferenciales estarían incluidos en la categoría de métodos clásicos en la rama de la optimización continua, no lineal y global; los otros métodos observados no utilizan las derivadas para resolver el problema.

Los métodos diferenciales

Históricamente el problema de optimización no lineal se solucionó con la teoría clásica de máximos y mínimos que utiliza los criterios de las derivadas para hallar los puntos estacionarios que muestran las inflexiones de una función y pueden ser: puntos máximos, mínimos o de cambio de concavidad. Este se denomina un método indirecto o analítico, pues la solución se determina por las propiedades diferenciales de la función o funciones involucradas, y no por la evaluación directa de la misma. Es un método que es de uso muy restringido a funciones no lineales derivables (Pierre, 1986).

Otros métodos utilizan la información de los puntos estacionarios encontrados por las derivadas y los mezcla con algoritmos numéricos de búsqueda directa, lo que permite mayor velocidad y abarcan mayor

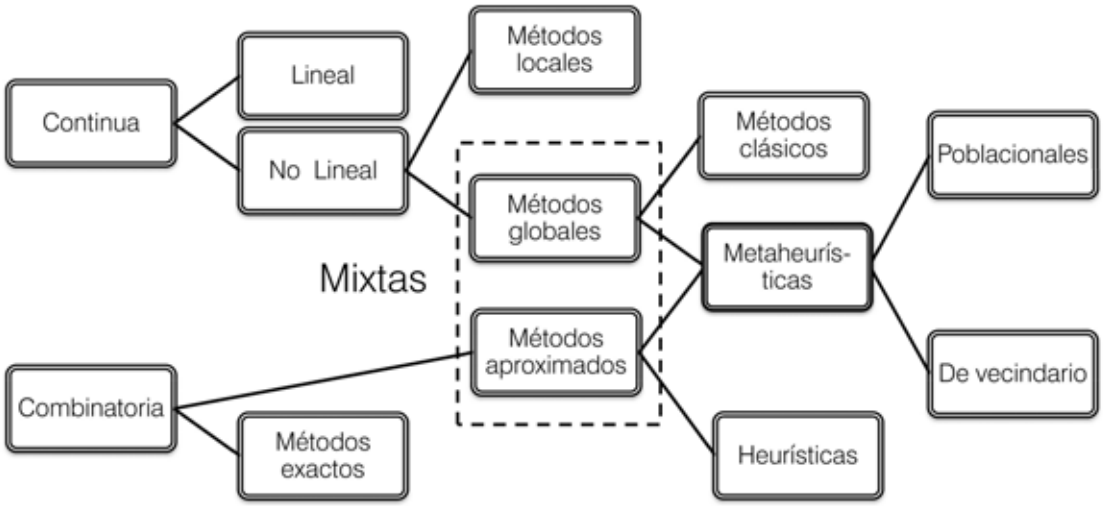


Figura 1.3: Métodos de optimización de acuerdo al tipo de problema.

cantidad de tipos de problema. Estos métodos han sido ampliamente aplicados, son rápidos y eficaces, siempre y cuando las condiciones iníciales necesarias estén dadas: deben ser funciones donde la labor analítica inicial sea posible (funciones derivables), lo que los aleja de muchos problemas del mundo real. Entre estos métodos se pueden mencionar: el método del gradiente y sus variaciones, método de Newton y sus variaciones, punto interior (Jang et al., 1996; Nash y Sofer, 1996).

Los métodos no diferenciales

Estos métodos ignoran la información de las derivadas, lo que les permite abordar una gran gama de problemas; a pesar de no ser tan eficientes en términos de velocidad para problemas de fácil solución analítica (es decir, fácilmente derivables, baja no-linealidad y continuos) pueden solucionar problemas de diversa índole, con varias dimensiones, alta complejidad, para variables continuas, discretas y mixtas, mostrando gran potencial para problemas reales (Jang et al., 1996; Vesterstrom y Thomsen, 2004). Dentro de estos métodos se pueden destacar los métodos directos determinísticos y aleatorios entre los cuales se encuentran los métodos de Rosenbrock, Nelder-mead, Powell, búsqueda aleatoria repetitiva, entre otros (Himmelblau, 1972; Nash y Sofer, 1996).

En la Figura 1.3, este tipo de métodos correspondería a las heurísticas y metaheurísticas utilizadas en la solución de problemas no lineales y con cualquier tipo de variables; estos métodos son directos, heurísticos y pueden ser determinísticos o estocásticos. Dentro de estos métodos hay un subconjunto importante: las metaheurísticas (de Castro, 2007).

Las metaheurísticas no tienen un fundamento matemático robusto, son intuitivas y llegan a una solución

mediante ciertas reglas planteadas empíricamente; se inspiran en el entorno, y generalmente llegan a una buena solución, en términos de cercanía al óptimo global del problema; pero, no tienen la precisión de los métodos diferenciales. Las primeras ideas al respecto se encuentran en la literatura en los años 60's cuando Holland propone los Algoritmos Genéticos (Goldberg, 1989) y Fogel la Programación Evolutiva (Bäck et al., 1997). Aunque inicialmente no pudieron ser muy desarrollados por la tecnología computacional de la época, a partir de los 80's se han mejorado y desarrollado una serie de métodos que se presentaran más adelante (Rao, 1996). Este trabajo se enfoca en los métodos no diferenciales, particularmente las metaheurísticas.

1.4. METAHEURÍSTICAS

Las metaheurísticas son técnicas de inspiración biológica y física, que por medio de heurísticas emulan los procesos de optimización que se pueden observar en la naturaleza, como por ejemplo: la evolución. Las metaheurísticas no tienen una base matemática estricta y optimizan las funciones buscando directamente en el universo de soluciones con diversas estrategias; por sus características generalmente proveen una respuesta en menor tiempo y de mejor calidad a problemas de alta complejidad mostrando así gran potencial en aplicaciones del mundo real (Dorigo y Stützle, 2003; Michalewicz y Schmidt, 2002).

Las metaheurísticas abordan problemas que por sus condiciones no son resueltos satisfactoriamente por los métodos derivativos y por su estructura no necesitan la información diferencial. Pero, aunque resuelven problemas más variados, su naturaleza iterativa y al no tener la ayuda inicial de las derivadas, los hacen más lentos, susceptibles al punto inicial de optimización y hacen que los parámetros del algoritmo cobren gran importancia en el funcionamiento y resultado del mismo (Rao, 1996; Feoktistov, 2006).

En la Tabla 1.1[1] se presenta un conteo de publicaciones de las técnicas más populares en aplicaciones, híbridos, mejoras metodológicas, comparaciones y pruebas. Se tienen en cuenta artículos, conferencias, libros, capítulos de libros de las principales publicaciones del área. Se aprecia el aumento del uso de los métodos metaheurísticos y la mayor aceptación de algunos métodos, como Algoritmos Genéticos, Temple Simulado o PSO, mientras que métodos como Programación Evolutiva y Estrategias Evolutivas, a pesar de ser métodos más antiguos, no son tan frecuentemente aplicados. De igual manera, las metaheurísticas consideradas tanto en la Tabla como en las secciones siguientes se limitan a las que tienen más acogida según la literatura.

Las metaheurísticas toman su inspiración de procesos que mejoran sus condiciones o cuyo objeto o comportamiento emergente muestre una notable mejoría en alguna característica, ya sea en la calidad final o en el refinamiento de ciertos parámetros del proceso; por ejemplo, el temple de un material, es un proceso que mejora la calidad del material por el control de la temperatura (de Castro, 2007; Jang et al., 1996). Teniendo en mente los procesos, se intenta simularlos por medio de heurísticas emulando el comportamiento de mejoría

[1] Fuente: Base de datos SCOPUS

Tabla 1.1: Número de artículos por técnicas bio-inspirada

Metahurística	Número de Artículos (metodología y aplicaciones)			
	antes del 2001	2001-2005	2006-2010	Total
Algoritmos Genéticos	7285	14723	13850	35858
Temple simulado	3490	3287	3388	10165
PSO	7	742	5262	6011
Colonia de hormigas	47	699	2223	2969
Búsqueda Tabú	465	692	793	1950
Evolución diferencial	93	382	1362	1837
Algoritmos Inmunológicos	65	420	1180	1665
Programación evolutiva	252	382	321	955
Estrategias evolutivas	168	342	620	930
GRASP	235	298	392	925
Algoritmos Meméticos	5	141	396	542
Búsqueda dispersa	21	93	170	284
Hooke and Jeeves Pattern search	113	34	55	202
Quimiotaxis bacteriana	N.A.	1	22	23
Big Ban Big Crunch	N.A.	N.A.	6	6

(optimización) que presenta el proceso de donde nace la idea; así, estos métodos han logrado soluciones aceptables a problemas de gran dificultad en un tiempo y a unos costos razonables (Bäck et al., 1997).

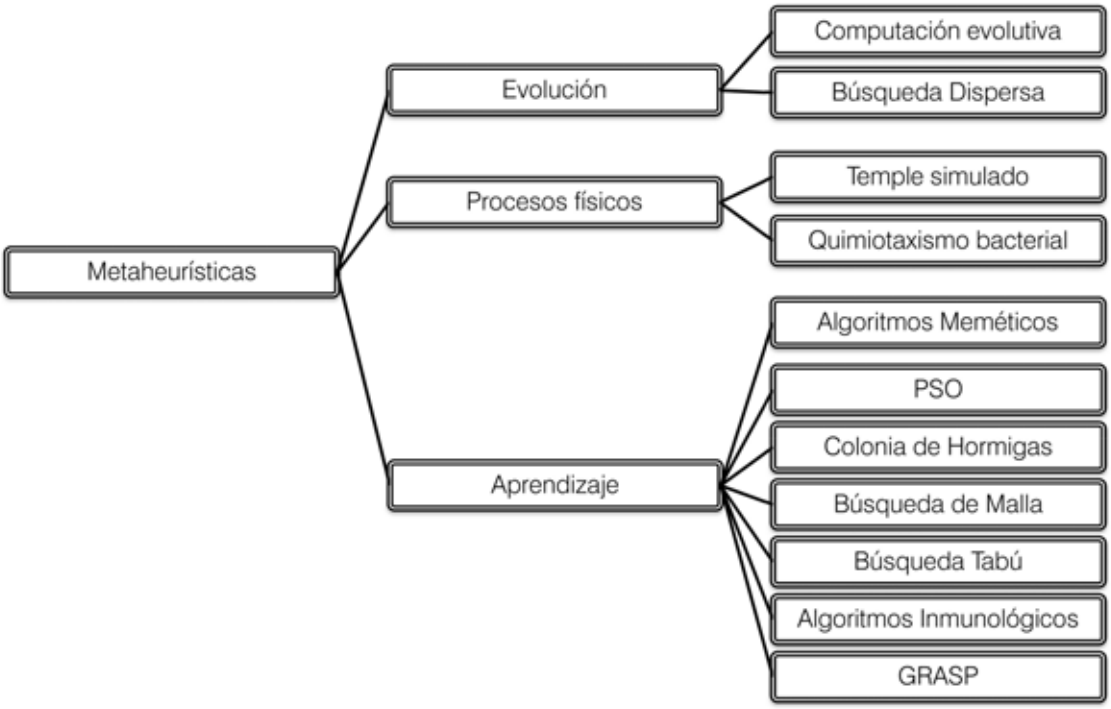


Figura 1.4: Metaheurísticas de acuerdo a su inspiración.

Las metaheurísticas tienen diversas fuentes de inspiración, pero se pueden identificar básicamente tres

Tabla 1.2: Equivalencias entre la teoría de evolución y el problema de optimización

Evolución	Modelo de Optimización
Individuo(Población) $\Longrightarrow$	Solución $x \in$ (conjunto $\aleph$ de soluciones posibles)
Adaptación $\Longrightarrow$	Valor de la función objetivo $S(f(x))$
Generación $\Longrightarrow$	Iteración del ciclo
Recombinación$\Longrightarrow$	Combinación de individuos para obtener nuevas soluciones
Mutación $\Longrightarrow$	Cambios aleatorios que dan nuevas características al individuo
Selección $\Longrightarrow$	Toma los mejores individuos

grandes grupos: evolución, procesos físicos y aprendizaje, dentro de los cuales se pueden enmarcar los diferentes métodos que han sido desarrollados, ya sea dentro de una de las líneas estrictamente o en una combinación de ellas. En la Figura 1.4 se observan las técnicas más representativas de las diferentes categorías y se detallan brevemente en el Anexo A.

1.4.1. Técnicas inspiradas en la evolución

Basado en los trabajos de Darwin, la evolución se describe como el mejoramiento de una población por medio de la reproducción de los seres más aptos. Más tarde, Lamark en su trabajo sobre evolución, le da un peso mayor a las mutaciones de los individuos y describe como pequeños cambios en los individuos logran que una población se adapte mejor a su medio o a un cambio en él. En la Tabla 1.2 se observa un paralelo entre la teoría de evolución y el problema de optimización, el mismo paralelo se presenta en cada uno de los métodos de optimización bio-inspirados; la población representa un conjunto de soluciones diferentes y el medio sería la función de adaptación que describe cuales son los mejores individuos de acuerdo al criterio del medio; en algunos casos se toma la misma función objetivo o algún escalamiento de la misma; en este enfoque, los métodos presentan una búsqueda elitista, tienen una alta tasa de convergencia y exploran especialmente a nivel local. Se han desarrollado estrategias para evitar la convergencia prematura y la ingerencia de los parámetros creando parámetros adaptativos para algunos tipos de problema (Coello, 2005).

1.4.2. Técnicas inspiradas en los procesos físicos

Estas ténicas se inspiran en fenómenos físicos en los que algún parámetro es mejorado, por ejemplo: en la quimiotaxis bacteriana, que es el proceso por el cual una bacteria direcciona sus movimiento de acuerdo la concentración química del medio y de esta forma encuentra su alimento (de Castro, 2007; Muller et al., 2000). Al igual que en el caso anterior, se tiene el fenómeno y su aproximación al modelo de optimización. Aunque este es el grupo que presenta más diferencias entre los procesos de donde toman su inspiración, tienen como común denominador que el proceso físico descrito mejora sus condiciones por el control o la búsqueda de una variable específica. En el temple simulado, por ejemplo, se tiene como variable de control

de mejoramiento a la temperatura. Las técnicas enmarcadas en este enfoque tienen una forma de búsqueda metódica de más grande a más pequeño que se va moviendo de acuerdo a la mejor solución encontrada y su convergencia depende especialmente del paso de la búsqueda, los parámetros del algoritmo y el punto de inicio de la misma. Son heurísticas de búsqueda en el vecindario.

1.4.3. Técnicas inspiradas en el aprendizaje

El aprendizaje, aunque tiene en común con la evolución el principio de transformación para mejorar, tiene su fundamento principal en la memoria, es decir, las decisiones pasadas tienen inferencia en las decisiones futuras del individuo (solución) y por lo tanto *aprende* como hacer mejor las cosas. Se inspira en diversos fenómenos y comportamientos emergentes, como el comportamiento gregario de las hormigas, donde el aprendizaje individual comunicado logra un mejor comportamiento global; o sistemas de aprendizaje como el sistema inmune que genera anticuerpos luego de vencer a cierta bacteria o virus, mejorando su comportamiento ante el mismo estímulo en el futuro (de Castro, 2007; Talbi, 2009). Esta línea de inspiración mejora algunas falencias de los grupos anteriores por implementación de los mecanismos de aprendizaje, logrando una mayor exploración del espacio de búsqueda en algunos casos. Aún así, su exactitud depende mayormente de la calibración de los parámetros del algoritmo y de la cantidad de dimensiones del problema.

1.5. DIFICULTADES DEL PROBLEMA DE OPTIMIZACIÓN GLOBAL SIN RESTRICCIONES

1.5.1. Respecto a la optimización global

Para una región factible A pueden existir múltiples máximos y mínimos; un ejemplo es la función multimodal de Griewank, mostrada en la Figura 1.5, que aumenta su número de máximos y mínimos locales exponencialmente al aumentar las dimensiones y está acotada a la región $[-600, 600]^n$. Aunque sólo tiene un solo mínimo global en la región factible, por su configuración se puede llegar a un mínimo local sin lograr una convergencia al mínimo global. Uno de los principales problemas de la optimización global sin restricciones es que para muchas funciones desconocidas no se encuentra el mínimo global, sino un mínimo local (Törn y Zilinskas, 1989).

1.5.2. Sensibilidad al punto de inicio

Los métodos de búsqueda directa necesitan un punto de inicio; dependiendo del método, este punto o puntos están dentro o fuera de la región factible. Dependiendo del lugar de inicio, en la mitad, cerca del borde, fuera de la región A, se pueden llegar a soluciones muy diferentes. De ahí la importancia del punto de inicio en el funcionamiento y respuesta (Nguyen et al., 2007; Yao et al., 1996).

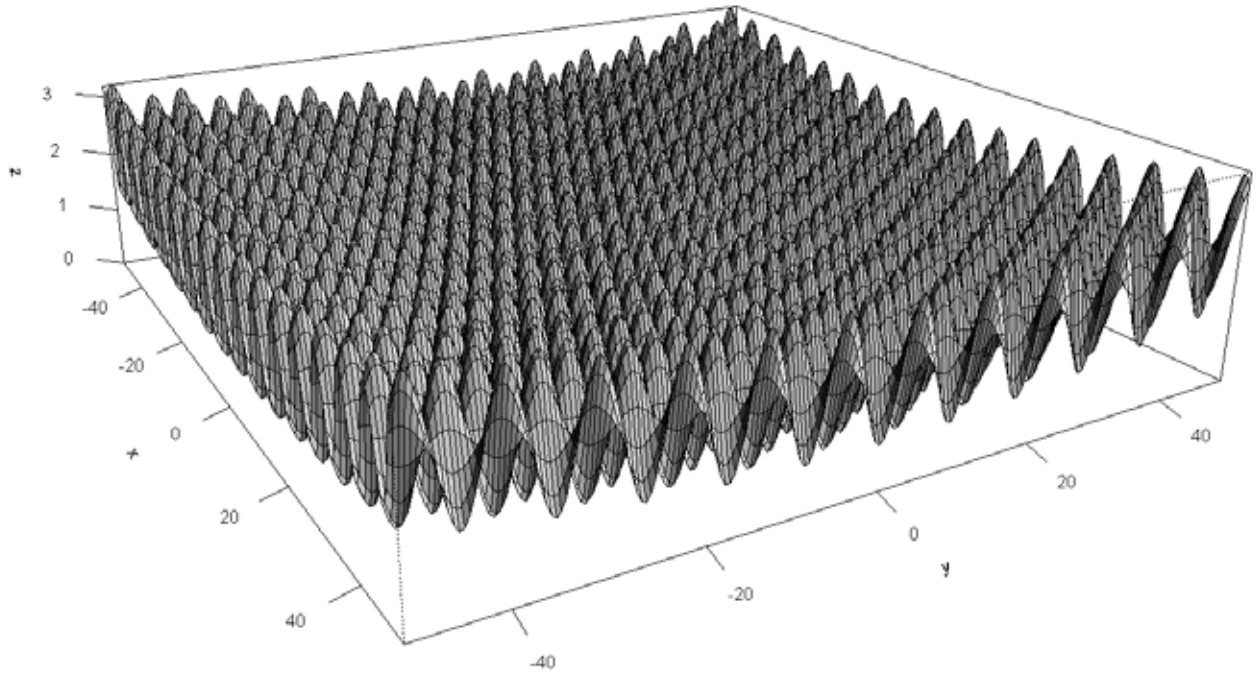


Figura 1.5: Función de Griewank

1.5.3. Precisión

Con un método numérico se esperan respuestas solamente aproximadas; ¿Cuáles son las aproximaciones aceptables? Cada problema tiene una respuesta diferente a este interrogante (Törn y Zilinskas, 1989). En la Figura 1.6 cada una de las partes (a,b,c) muestran un paso de búsqueda diferente para un mismo número de iteraciones; en el caso *a* el paso es demasiado pequeño para lograr una exploración de la región; en el caso *b* se mejora la exploración pero, aún así no es suficiente para llegar al punto de mínima, mientras que en la parte *c* de la Figura, el paso es tan grande que salta de lado a lado sin llegar al mínimo, lo que ejemplifica el problema de la precisión y el costo. Entre mas precisión se exija mas costosa resultará la solución. También es importante recordar que las funciones no son iguales y, además, tienen espacios de búsqueda diferentes lo que impide que una forma de búsqueda eficiente para un problema específico lo sea para otra función. Por esta misma razón tampoco, se puede predecir el comportamiento exacto de una heurística antes ser aplicada en otro tipo específico de problema (Pierre, 1986; Törn y Zilinskas, 1989).

1.5.4. Complejidad

La complejidad del problema de optimización se refiere a la dificultad para ser resuelto. Clásicamente se describen como problemas duros o altamente complejos los problemas no polinómicos. Las dificultades para

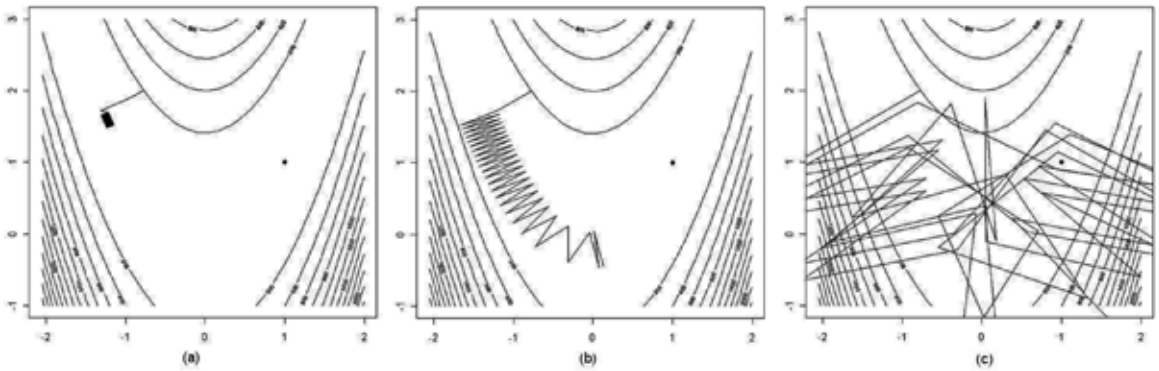


Figura 1.6: Trayectoria de búsqueda con distintos pasos en la función de Rosenbrock

solucionar el problema pueden estar en: la función objetivo, el número de dimensiones, las restricciones, el tipo de variables. Particularmente, el problema de optimización continua y sin restricciones se dificulta en dos áreas básicas: el tipo de función y las dimensiones. Las funciones miden su complejidad en ciclos de

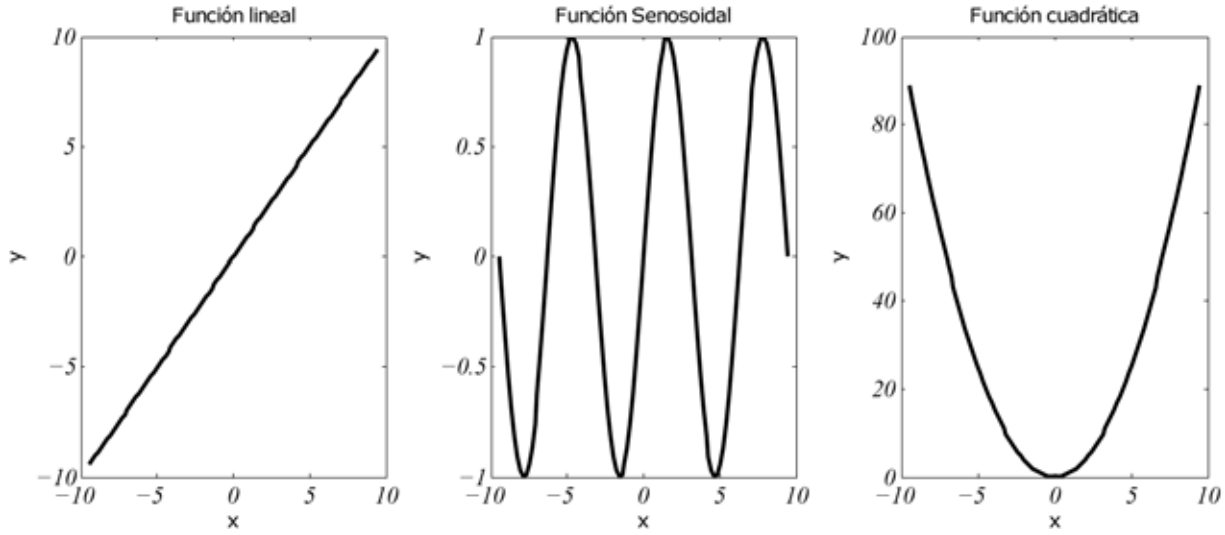


Figura 1.7: Complejidad para diferentes funciones

retroalimentación, funciones internas, funciones por partes, etc. En la Figura 1.7 se muestran algunos casos de aumento de complejidad para una función de una dimensión. El caso más simple para una función es $y = x$. Si se utiliza la función seno, $y = \sin x$, se observa el aumento de máximos y mínimos que hacen más difícil el problema de la búsqueda. Si se tiene un ciclo retroalimentado, $y = x^2$, como en la función cuadrática, aunque no hay aumento de mínimos, la geometría del vértice hace muchos métodos heurísticos fallen, pues se saltan fácilmente el mínimo.

Respecto a las dimensiones, estas hacen más intrincada la búsqueda pues, se necesita más precisión en

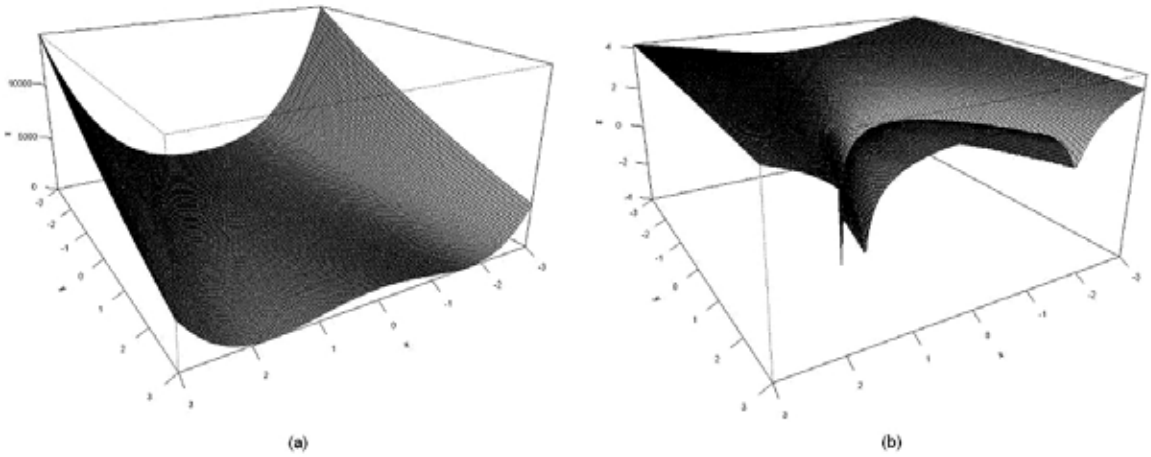


Figura 1.8: Aumento de la complejidad con las dimensiones en la función Rosenbrock

cada variable para obtener un resultado igualmente bueno. Muchos de los problemas benchmark utilizados en las pruebas de optimización global aumentan su dificultad exponencialmente a medida que aumentan las dimensiones. En la Figura 1.8 (*a*) se puede observar la función de Rosenbrock en dos dimensiones y en la parte (*b*) una idea del efecto de las dimensiones sobre la función, se observa el aumento de la profundidad en una zona casi imperceptible en dos dimensiones y el mínimo se vuelve una aguja haciendo más difícil la tarea de búsqueda pues este mínimo puede ser fácilmente saltado.

1.6. DISCUSIÓN

En las secciones anteriores se describe el problema de optimización y las formas como se ha abordado, las dificultades encontradas y un fenómeno que, simulado, podría ser una estrategia de búsqueda exitosa en el problema de optimización; aún así, es importante puntualizar y enfatizar algunos aspectos:

- Los procesos que inspiran las estrategias de búsqueda, descritas en las secciones anteriores, tienen un aspecto común: el mejoramiento en algún aspecto del proceso del cual toman la idea, por ejemplo: el sistema inmune de los vertebrados, que inspira los algoritmos inmunológicos, procesa los antígenos creando una inmunidad haciéndose cada vez más fuerte, es decir, mejora su capacidad de respuesta ante los mismos estímulos externos (Bäck, 1996; Kennedy y Eberhart, 1995; Muller et al., 2000; Resende y Ribeiro, 2002).

- Las heurísticas deben ser implementadas antes de determinar su funcionamiento, en términos de calidad y costo computacional; sin importar lo clara que parezca la idea, es imposible determinar su desempeño para un problema específico antes de probarla (Wolpert y Macready, 1997; Talbi, 2009; Törn y Zilinskas, 1989).

- Las heurísticas toman la inspiración de diferentes procesos, como: evolución, temple, aprendizaje, etc, pero una misma idea puede producir varios enfoques y reglas de búsqueda diferentes, como en el caso de la evolución, que da inspiración a cuatro métodos de solución diferentes: algoritmos genéticos, evolución diferencial, estrategias evolutivas y programación evolutiva, todas heurísticas diferentes entre sí, pero con muchas aplicaciones referenciadas (Ver Tabla 1.1). Por lo tanto, un mismo fenómeno puede desarrollarse de diferentes maneras siendo todas válidas y aplicables (Bäck, 1996).

- Ninguna heurística es totalmente superior a otra en todos los problemas; algunas heurísticas funcionan mejor en problemas combinatorios mientras otras lo hacen en problemas continuos; unas son más precisas, otras más veloces y así sucesivamente. No hay un límite específico respecto al error, coste computacional, globalidad en el desempeño de las heurísticas; es decir, siempre puede existir una mejor manera de lograr una buena respuesta (Wolpert y Macready, 1997; Törn y Zilinskas, 1989).

- Desarrollar una heurística es un proceso de depuración. La forma de desarrollo, prueba y los diferentes enfoques posibles de una metaheurística impiden que se conozca de antemano la ruta y el desarrollo que puede tener una idea inicial ni su potencial futuro antes de ser implementada y desarrollada. Por ejemplo: Kirkpatrick et al. (1983) propuso un método inicial, Temple simulado, que ha sido desarrollado y mejorado en el tiempo, utilizado en híbridos con otras metaheurísticas y renovado en métodos como: Umbral de aceptación (*Threshold acceptance*) y Temple cuántico (*Quantum annealing*) (Winker, 2001; Das y Chakrabarti, 2005), obteniendo mejores respuestas con mayor eficiencia computacional.

Dado que la supernovas son un fenómeno que va desde el caos de una explosión al orden de las partículas de acuerdo a diferentes variables del proceso; es decir, mejora, puede existir una forma de simular el proceso por medio de una heurística que sea una estrategia de optimización exitosa. Pero al mismo tiempo que no es posible determinar el rumbo y el desempeño de la misma antes de implementarla, probarla y validarla.

Una vez implementada y validada una heurística puede ser aplicada a diversos problemas del mundo real, como por ejemplo: la optimización de los parámetros de diferentes modelos de redes neuronales.

1.7. DELIMITACIÓN DEL PROBLEMA

Esta tesis se centra en el desarrollo de una nueva heurística inspirada en el fenómeno de las supernovas para resolver el problema de optimización de funciones continuas no lineales sin restricción. Se busca encontrar una heurística que por sus características converja al mínimo global de una función no lineal de alta complejidad, y que a su vez, logre un mejor funcionamiento para un grupo de funciones de prueba. También se aplicará al problema práctico de las redes neuronales artificiales. Dado lo amplio del campo de

la optimización se limitarán los alcances a la optimización continua, sin restricciones, estática, con una sola función objetivo. También, dada la cantidad de métodos existentes, se limitará su comparación a métodos de optimización global heurísticos de características similares que tengan resultados para el conjunto de prueba en la literatura.

1.8. OBJETIVOS

1.8.1. Objetivo General

Diseñar un algoritmo de optimización global basado en el fenómeno de las supernovas y realizar su implementación para compararlo con otras técnicas convencionales y analizar su comportamiento.

1.8.2. Objetivos Específicos

1. Diseñar una heurística basada en el fenómeno de las supernovas para la optimización de funciones.
2. Implementar un prototipo operativo basado en la heurística diseñada en el objetivo anterior con el fin de evaluar sus bondades y falencias al optimizar funciones no lineales de prueba.
3. Analizar el comportamiento de la metaheurística implementada al optimizar veintitrés funciones continuas que hayan sido utilizadas en la literatura y evaluar su desempeño.
4. Aplicar el algoritmo al problema de redes neuronales artificiales y comparar su desempeño con algunos algoritmos típicos utilizados para esta aplicación.

1.9. ORGANIZACIÓN DEL DOCUMENTO

El Capítulo 2 presenta la descripción de la heurística y su enfoque respecto al fenómeno de donde toma su inspiración. En el Capítulo 3 se presentan los resultados de las fases de implementación y parametrización. En el capítulo 4 se valida el desempeño de la heurística Supernova al compararlo con los resultados de otras heurísticas similares descritos en la literatura. En el capítulo 5 se presenta una aplicación del método el problema de optimización de la redes neuronales para modelos cascada correlación.

1.10. DIVULGACIÓN DE LOS RESULTADOS

En el proceso del trabajo de investigación, se han publicado los siguientes artículos:

- Mesa, E.; Velásquez, J. D.; Jaramillo, G. P. 2009. Optimización global basada en una teoría de la formación del universo. En: Tendencias en Ingeniería de Software e Inteligencia Artificial, Volumen 3. Editado por: C.M. Zapata, G. Giraldo y J. D. Velásquez. Universidad Nacional de Colombia.

- Mesa, E.; Velásquez, J. D.; Jaramillo, G. P. 2009. Algoritmos Bioinspirados. En: Tendencias en Ingeniería de Software e Inteligencia Artificial, Volumen 3. Editado por: C.M. Zapata, G. Giraldo y J. D. Velásquez. Universidad Nacional de Colombia.
- Mesa, E.; Velásquez, J. D. 2009. Optimización global basada en una teoría de la formación del universo. 4to Congreso Colombiano de Computación. Abril, 2009.

2. SUPERNOVA

Las heurísticas se mencionan como forma de solución a problemas complejos de optimización desde los años 40's. Por definición, son reglas simples que al aplicarse a un problema determinado obtienen una solución de buena calidad a bajo costo, aunque sin garantizar que ésta sea óptima. Las metaheurísticas son procesos que combinan de manera inteligente las heurísticas logrando soluciones de alta calidad, pero igualmente sencillas (Voß, 2001).

Las metaheurísticas, generalmente, se inspiran en procesos del entorno. La analogía de un proceso que se ordene, o mejore su desempeño puede proveer información valiosa que permita resolver problemas de optimización de gran complejidad. Desarrollar el conjunto de reglas a partir del proceso es un problema difícil, ya que, la heurística como tal debe ser lo más simple posible, conservando los lineamientos adecuados que logren optimizar el problema. Además, debe poder implementarse computacionalmente.

El fenómeno de las supernovas es un proceso que desde su inicio caótico llega a un equilibro mas ordenado, es decir, mejora su orden; en este capítulo se describirá el enfoque propuesto para el desarrollo de una heurística a partir del fenómeno de las supernovas. En la sección 2.1 se describe el fenómeno. En la sección 2.2. se presenta la estratégia de búsqueda que se utilizará en la heurística. En la sección 2.3. se propone un algoritmo para implementar. En la sección 2.4. se hace un paralelo con las diferentes heurísticas. Y en la sección 2.5. se concluye el trabajo realizado en el capítulo.

2.1. DESCRIPCIÓN DEL FENÓMENO

Cuando algunas estrellas agotan su combustible interno inicial (hidrógeno) llegan a su fin como supernovas (Carroll y Ostlie, 2007). No es del interés presentar una tratado extensivo sobre las supernovas, sólo se presenta su funcionamiento general y la idea de como este fenómeno puede ser una estrategia de optimización.

Inicialmente, las estrellas queman su combustible, y este, a su vez, se convierte en elementos de mayor masa: del hidrógeno al helio, luego carbono, y así sucesivamente hasta llegar a un núcleo de hierro, ordenados por capas como una cebolla. Las altas temperaturas y presiones internas descomponen los átomos de hierro haciendo que el núcleo sea inestable y colapse en cualquier momento, dando lugar a una supernova. Las ondas de choque generadas internamente por este colapso provocan una expulsión de fotones, neutrones y material particulado (Carroll y Ostlie, 2007). Las partículas expulsadas interactúan entre ellas y con otros cuerpos cercanos a su trayectorias, donde las partículas y cuerpos de mayor masa atraen a las que tienen menos masa a medida que pierden su impulso (Alonso y Finn, 1995).

Una breve síntesis del fenómeno sería: una explosión de partículas que se organizan en el tiempo de acuerdo a su velocidad y masa siguiendo las leyes de la conservación de la energía. Simplificando el fenómeno se podrían delinear algunas reglas que logren encontrar buenas respuestas a problemas de optimización. La explosión inicial se puede emular como aleatoriedad y la interacción de partículas expulsadas restringirse a dos interacciones: una que expande y otra que atrae las partículas. Aunque es un fenómeno sumamente complejo, por simplicidad, se tomaran sólo dos de las fuerzas presentes en el fenómeno, el impulso y la gravitación; la interacción entre ambas fuerzas llena las expectativas de expansión y tracción mencionadas (Lieddle, 2003).

Los valores de masa y velocidad adaptados a la optimización son bastante pequeños en relación a los del fenómeno real, por lo tanto, las ecuaciones son reemplazadas por su versión clásica, las cuales se adaptan mejor a la idea de optimización. Las constantes no se tomarán en su magnitud real pues sólo entorpecerían el funcionamiento del algoritmo, dado que las magnitudes de las variables no son comparables; más bien, se tomarán como parámetros para controlar el proceso de búsqueda. Con la síntesis del proceso planteado logra un sistema de ecuaciones simple que conserva el comportamiento de expansión y atracción observados en el fenómeno de las supernovas.

La conservación de momentum o impulso p hará que la partícula intente seguir con la dirección y velocidad que traía, de acuerdo a:

$$p = m\vec{v} \tag{2.1}$$

donde m representa la masa y $\vec{v}$ la velocidad de la partícula (Grupen, 2005). La masa y la velocidad son proporcionales al impulso, es decir, a mayor masa o velocidad se presentará un mayor impulso.

La energía potencial gravitatoria F de un grupo de k partículas clásicamente está descrita por:

$$F = m \sum_{i=1}^{k} \frac{GM_i}{r_i^2} \vec{u} \tag{2.2}$$

donde m representa la masa de la partícula para la cual se calcula la energía potencial gravitatoria; M_i es la masa de la otra partícula, G es la constante de gravitación universal, r_i representa la distancia entre ambas y $\vec{u}$ es un vector unitario que da la dirección entre ambas partículas. La interacción gravitacional intentará atraer las partículas de menor masa hacia las de mayor masa de acuerdo a sus distancias variando la dirección y velocidad inicial dada por la explosión. La masa es directamente proporcional a la fuerza, por lo tanto, entre mayor masa tenga una partícula mayor será su atracción hacia otras, mientras que la distancia es inversamente proporcional, es decir, las partículas mas lejanas tendrán menos inferencia en la dirección del movimiento que las más cercanas (Alonso y Finn, 1995).

En términos de optimización, si se define la masa como el valor de función objetivo evaluada en la solución actual, donde una mayor masa es una mejor respuesta respecto a otra de menor masa, sería deseable

tener una interacción donde las partículas con mayor masa atrajeran a otras partículas de menor masa como en el fenómeno de la supernova. Las ecuaciones planteadas, a pesar de tratarse de una simplificación del fenómeno original, llegan a un equilibro entre la fuerza atrayente (gravedad) y la fuerza detractora (impulso) mostrando el mismo comportamiento del fenómeno. En la Figura 2.1, las estrellas representan las soluciones encontradas para la función delimitada que tiene su mínimo en $(0, 0)$. Esta Figura muestra la estrategia de búsqueda deseada para la heurística; en la parte (a) se observan los puntos aleatorios uniformemente distribuidos en el espacio de búsqueda. En la parte (b) se muestra una pequeña expansión inicial y luego en la parte (c) se muestra la trayectoria de atracción esperada. Luego de llegar al equilibrio se toma el mejor punto, identificado por el cuadrado en la parte (d) de la Figura, y a partir de este punto, se explotan nuevos puntos con un radio de búsqueda más reducido (elipse punteada). Este proceso se describirá con más detalle posteriormente.

En la Tabla 2.1 se establece el paralelo entre algunas partes del fenómeno físico y su aplicación en el modelo de optimización. Se debe tener en cuenta que para una minimización la masa sería el inverso de la función objetivo, por lo tanto, es importante evitar que la función objetivo sea cero. La velocidad se toma aleatoria, y el potencial gravitacional y la energía cinética son las formas de interacción entre partículas.

Tabla 2.1: Equivalencias entre el modelo físico y el modelo de optimización

Modelo Físico	**Modelo de Optimización**
Partícula $\Longrightarrow$	Solución $x \in$ conjunto $\aleph$ de soluciones posibles
Masa $\Longrightarrow$	Valor escalado de la función objetivo $\Phi(S(x))$
Velocidad $\Longrightarrow$	Paso entre solución y solución
Potencial gravitacional $\Longrightarrow$	movimiento de la partícula hacia la mejor solución
Impulso $\Longrightarrow$	movimiento de la partícula en su trayectoria original

2.2. ESTRATEGIA

La heurística planteada sigue la secuencia tomada de las fases más generales de la supernova: el núcleo inicial, la explosión, el movimiento iterativo hasta llegar al equilibrio. Aunque no se han desarrollado heurísticas específicas sobre este fenómeno, se puede encontrar similitudes con otras; la fase de movimiento iterativo se asemeja al comportamiento de la optimización por nube de partículas, y los reinicios, estrategia de precisión tiene similitud al algoritmo de Temple simulado y Big-Bang Big-crunch, ver Anexo A. La combinación de ambas estrategias se describe a continuación.

2.2.1. Explosión

A partir del núcleo inicial, hay una explosión de material particulado. En términos de optimización, esta explosión se emulará como aleatoriedad. Se iniciará de dos formas diferentes. Se denominará Tipo I cuando

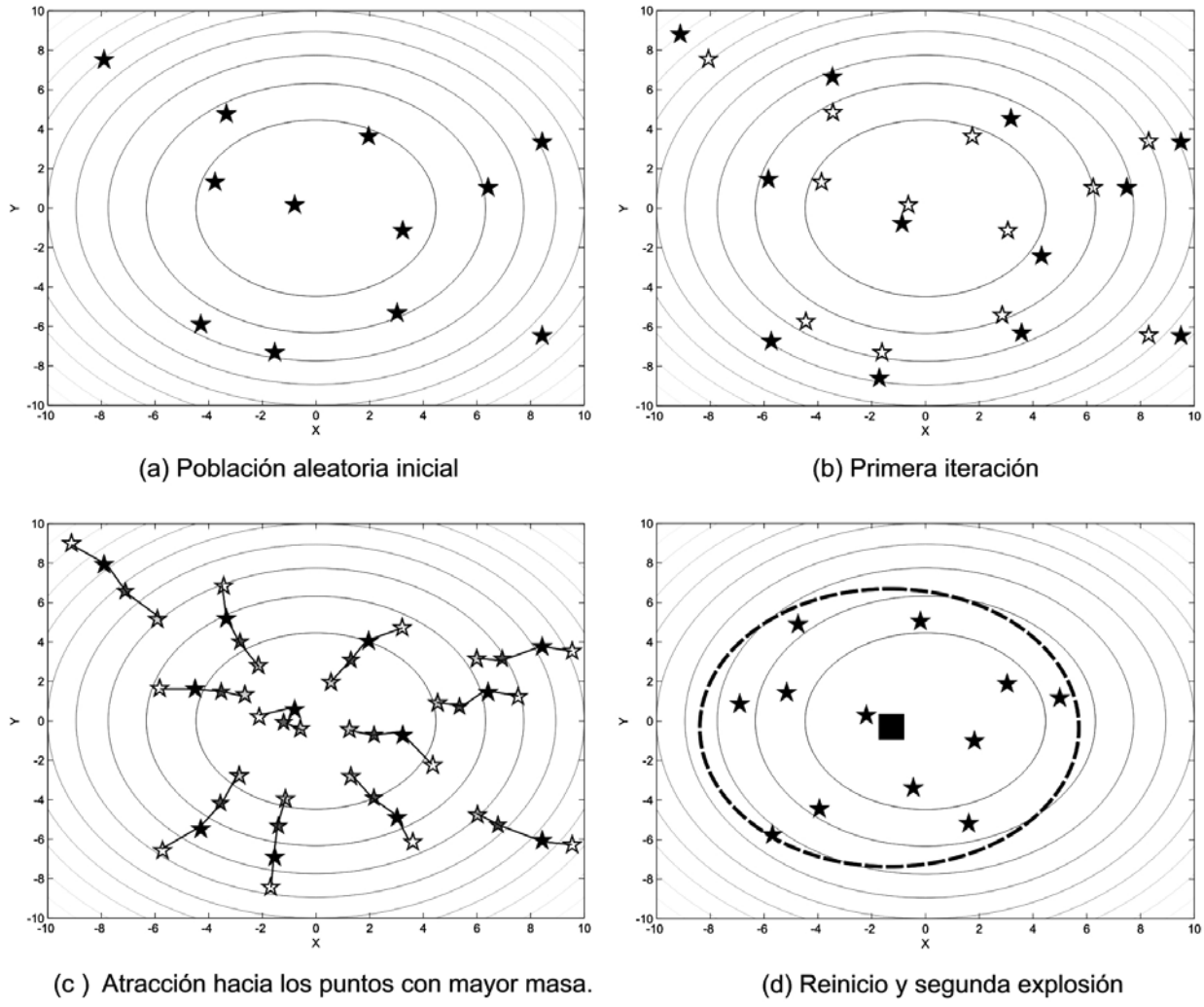


Figura 2.1: Distribución de partículas para la función esférica

desde un punto inicial x_0 se generan k soluciones aleatorias como se observa en la parte (d) de la Figura 2.1. En algunos casos no hay un punto x_0 disponible, sólo se tiene una región A acotada por los vectores UB y LB; para estos casos, se generaran k soluciones uniformemente distribuidas sobre la región A como se muestra en la parte (a) de la Figura 2.1; este inicio se denominará Tipo II.

2.2.2. Escalamiento

Cada una de las soluciones generadas en la explosión se evalúa respecto a la función objetivo f, este resultado es un número perteneciente a los reales. Al analizar detenidamente la ecuación (2.2) se puede observar que las variables son: masa y distancia; ambas magnitudes positivas. El uso de números negativos para la masa podrían generar falsos movimientos en la interacción y no llegar a una buena solución. En el caso de la minimización, al usar el inverso de la función como masa, estaría indefinido para el cero. Por tanto, para el correcto funcionamiento de la heurística es necesario garantizar que el valor de la función objetivo no sea negativo ni cero.

El problema de la no negatividad ha sido resuelto por otras heurísticas, como algoritmos genéticos, en donde se resolvió utilizando escalamientos para la función objetivo. Se han propuesto varios tipos de escalamiento: estático, dinámico, logarítmico, lineal, exponencial (Bäck, 1996). En estre trabajo se usa un escalamiento dinámico lineal modificado que garantiza que la función no tome valores negativos ni el valor de cero.

2.2.3. Cálculos de fuerzas atractivas y repulsivas

Una vez se tienen las soluciones con sus respectivos valores de masa se inicia la interacción entre ellas. Se halla la dirección de los vectores unitarios de trayectoria para cada partícula, en el caso de inicio Tipo I, desde el punto de inicio x_0 hacia un punto cualquiera; y para el caso de inicio Tipo II, desde el mejor punto encontrado hacia los otros puntos aleatorios. Una vez se tienen los vectores se hallan las distancias r_i entre los puntos. En este punto se calculan la dirección y la distancia de acuerdo al impulso y la atracción que ejercen las otras partículas.

En la Figura 2.2 representa la interacción entre partículas; r_{CA}, r_{BC} y r_{AB} son las distancias, los vectores $\vec{d}$ son los vectores del impulso, y los vectores $\vec{u}$ son los vectores correspondientes a la atracción gravitatoria. Los nuevos puntos se hallan con la sumatoria de los vectores de impulso y atracción gravitatoria hallados, y comienza la siguiente iteración que consiste en volver a calcular las fuerzas del impulso y la gravedad. Este proceso se repetirá hasta llegar a un número de iteraciones determinado.

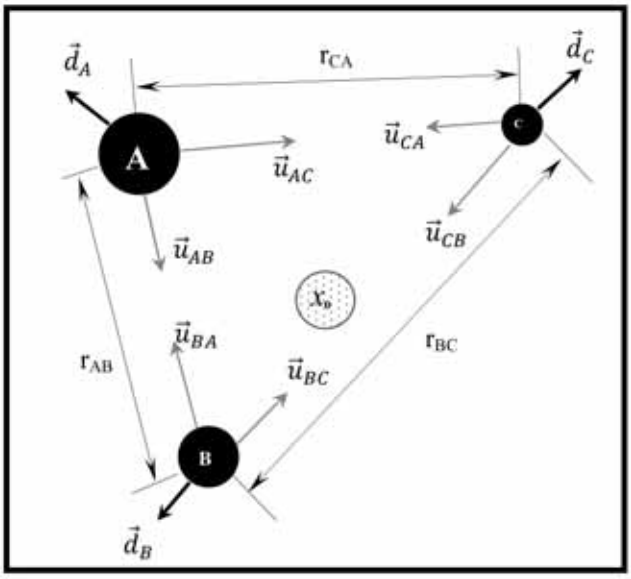


Figura 2.2: vectores de la interacción entre partículas

2.2.4. Reinicialización

La estrategia descrita hasta el momento genera una buena exploración, pero para mejorar la calidad de la respuesta se necesita refinar la búsqueda; con este objetivo se toma el mejor punto encontrado en las iteraciones de las fuerzas atractivas y repulsivas y se explota nuevamente, pero con un radio de búsqueda más pequeño buscando la explotación de un área específica, como se muestra en la Figura 2.1 (d).

2.3. EL ALGORITMO

En esta sección se realiza la descripcion más formal de la heuristica. Inicialmente se detallan las partes de la heurística y luego se presenta el pseudocódigo y los parámetros del algoritmo.

2.3.1. Explosión

Para la explosión Tipo I se genera el conjunto $S(0)$ como se muestra en la ecuación (2.3). Sobre una solución inicial x_0 se generan k puntos aleatorios, una fuerza de explosión λ y una velocidad para cada partícula v, que son parámetros independientes; $\vec{w}$ es un vector unitario aleatorio que prescribe la dirección de cada partícula i.

$$S(0) = \{\vec{x_0} + \lambda v \vec{w_1}, ..., \vec{x_0} + \lambda v \vec{w_k}\} \tag{2.3}$$

Para la explosión Tipo II el conjunto inicial $S(0)$ se genera al distribuir uniformemente los k puntos sobre una región acotada A, donde cada $\vec{s}$ es una solución aleatoria del espacio de búsqueda A; este proceso que

se describe en varias metaheurísticas poblacionales (Feoktistov, 2006; Bäck et al., 1997; Erol y Eksin, 2006).

$$S(0) = \{\vec{s_1}, ..., \vec{s_k}\} \tag{2.4}$$

2.3.2. Evaluación

Una vez generado el conjunto $S(0)$ se evalúa la función objetivo escalada $\Phi(S(0))$ para cada partícula i. En este caso se evalúa el conjunto $S(0)$ con la función objetivo, $f(S(0))$, y se encuentra el valor mínimo m_{min} del conjunto y se escala de acuerdo a:

$$\Phi(S(0)) = f(S(0)) - (m_{min} - 1) \tag{2.5}$$

2.3.3. Cálculos

Las partículas del conjunto $S(t)$, siendo t una iteración cualquiera, se vectorizan. La distancia se normaliza por medio de un promedio ponderado, donde los puntos mas cercanos tendrán un valor mayor, mientras los valores mas lejanos tendrán valores cercanos a cero, conservando el comportamiento de la distancia en (2.2), donde las partículas ejercen mas atracción a los puntos cercanos, mientras tienen una menor incidencia en el comportamiento de las partículas más lejanas. El movimiento de cada partícula se halla según:

$$S'(t) = \underbrace{\frac{F}{\Phi(S(t))}\vec{d}}_{a} + \underbrace{\frac{G}{\Phi(S(t))}\sum_{i=1}^{k}\frac{1-\frac{r_i}{\sum_{j=1}^{k} r_j}}{\Phi(S_i(t))}\vec{u}}_{b} \tag{2.6}$$

donde la parte a corresponde a la energía cinética y la parte b al potencial gravitorio. F y G son parámetros constantes, r es la distancia entre un punto i y el que se está evaluando. $\vec{d}$ y $\vec{u}$ son vectores unitarios de dirección que en el primer caso señalan la dirección que traía la partícula y en el segundo caso la dirección de cualquier solución hacia otra solución i cualquiera.

En la Figura 2.2 se muestra para un sistema de tres partículas las diferentes relaciones entre ellas, desde una partícula inicial x_0, se hallan 3 partículas aleatorias $S(0) = \{A, B, C\}$; para cada partícula se aplica (2.7). La función $\Phi(S(t))$ corresponde a la masa Φ de cada partícula que pertenece al conjunto $S(0)$ para la partícula A mostrada en la Figura, la Ecuación 2.7 correspondería a

$$S'_A(1) = \underbrace{\frac{F}{\Phi_A}\vec{d_A}}_{a} + \underbrace{\frac{G}{\Phi_A}\left(\frac{1-\frac{r_{AB}}{r_{AB}+r_{AC}}}{\Phi_B}\vec{u}_{AB} + \frac{1-\frac{r_{AC}}{r_{AB}+r_{AC}}}{\Phi_C}\vec{u}_{AC}\right)}_{b} \tag{2.7}$$

2.3.4. Reinicio

Luego de tener un punto de mínima encontrado de acuerdo a los parámetros del algoritmo; sobre este mismo punto se realiza el reinicio. Se explota nuevamente, con la explosión tipo I desde el punto x_j definido en la Ecuación 2.8.

$$\vec{x_j} = min(M(t)) \tag{2.8}$$

Esta explosión tiene un λ (ver ecuación 2.3)que disminuye con un factor α 2.8.

$$\lambda_{R+1} = \frac{\lambda_R}{\alpha} \tag{2.9}$$

es decir, la explosión tiene menos fuerza en cada reinicio, explotando así, la mejor zona encontrada en el reinicio anterior. Este proceso se repite hasta llegar al criterio de parada o el número de reinicios planteado.

2.3.5. Pseudocódigo

En el Pseudocódigo 1 se muestra la secuencia general seguida por el algoritmo Supernova en el proceso de optimización; se inicializa con una explosión de k elementos como se describió anteriormente. En este caso, puede utilizar ambos métodos de inicio indistintamente, tipo I y tipo II. Luego se evalúa la función de masa Φ y se inicializa la matriz de mejores elementos M. Posteriormente, se inicia la etapa de cálculo descrita en la Sección 2.3.3, se evalúa nuevamente y se actualizan las matrices M y Old, la matriz Old es la matriz donde se guarda las soluciones anteriores y permite la vectorización de los puntos. Cuando se llega al número de iteraciones, T, definido se vuelve a explotar sobre el mejor punto encontrado y se repite el proceso hasta llegar al número de reinicios estipulados RR. La matriz M conserva los mejores puntos encontrados por cada partícula.

El algoritmo requiere como datos de entrada: la función a optimizar, un punto inicial o los bordes específicos de la región de búsqueda de acuerdo al método de inicialización del algoritmo, dimensiones y los parámetros.

Los parámetros a elegir son:

- k, T, RR: son el número total de partículas que interactúan, el número de iteraciones máximas y el número de reinicios máximos respectivamente.

- F y G: son constantes que representan los pesos que tienen el impulso y la fuerza gravitatoria respectivamente.

- λ y α: La fuerza de explosión y su tasa de disminución en el tiempo, tienen que ver con el tamaño del área a explorar y la precisión que se requiera.

Pseudocódigo 1 Pseudocódigo Supernova

$t \Leftarrow 0$; $S \Leftarrow 0$; $M \Leftarrow 0$; $R \Leftarrow 0$
mientras $R \leq$ RR || criterio de parada **hacer**
  Explotar: $S(t)$ // Ecuación 2.3 o 2.4
  Evaluar: $\Phi(S(t))$ //Ecuación 2.5
  $M \Leftarrow S(t)$
  mientras $t \leq$ T || criterio de parada **hacer**
    Calcular: $(S(t+1)) \Leftarrow (S(t)) + S'(t))$ // Ecuación 2.7
    Evaluar: $\Phi(S(t+1))$
    $M_k = \arg\min(M_k|S_x(t+1))$ //guarda el mínimo de cada partícula
    $Old \Leftarrow S(t-1)$
    $t \Leftarrow t+1$
  fin mientras
  λ_{R+1} // Ecuación 2.9
  $R \Leftarrow R+1$

fin mientras

- El método inicial: Como se describe en la sección anterior hay dos estrategias para generar la población inicial la primera correspondiente al parámetros $metini = 1$ inicia la población desde un único punto utilizando una fuerza de explosión y direcciones aleatorias de diferente longitud. El segundo método correspondiente a $metini = 2$ inicializa k partículas uniformemente distribuidas sobre el espacio de búsqueda definido. Se utiliza el mas conveniente de acuerdo al problema dado.

2.4. SIMILITUDES Y DIFERENCIAS CON OTRAS METAHEURÍSTICAS

Supernova es una metaheurística poblacional en la que las partículas interactúan entre ellas como estrategia de exploración y reinicia en un radio mas pequeño como estrategia de explotación. Con las metaheurísticas descritas en el anexo A, Supernova comparte algunas similitudes y diferencias.

Con la heurística Big-bang Big crunch (BBBC), Supernova comparte la inspiración. La teoría cosmológica del Big-bang Big crunch y las supernovas tiene el mismo tipo de mejoramiento, van del caos al orden logrando un equilibrio final. Pero el enfoque es totalmente diferente, BBBC es un método que discretiza el espacio de solución, no tiene interacción entre partículas, se halla un centro de masa total que es tomado como el centro de la siguiente explosión, aunque explota sobre el centro de masa, esta explosión es sobre una región, no desde un punto. No se plantean estrategias que permita el trabajo en funciones con mínimos negativos.

En estrategias de búsqueda tiene similitud con el PSO de la manera que aborda el proceso de búsqueda pues tiene interacción particular, difiere en la estrategia de explotación planteada por Supernova, que reduce la búsqueda rápidamente a las áreas potenciales descubiertas. Y la estrategia de reinicio se parece mucho, al igual que en BBBC, al temple simulado con la búsqueda de vecindario, donde se explora aleatoriamente

un vecindario y el centro del vecindario se mueve al mejor punto encontrado del vecindario anterior y en el tiempo se va reduciendo el radio de búsqueda, esta idea se observa en la estrategia de reinicios propuesta por Supernova.

2.5. CONCLUSIONES

La heurística propuesta conserva los lineamientos generales del proceso de las supernovas, la explosión, el movimiento repulsivo y la atracción de los elementos más másicos. La analogía con el movimiento de las partículas se mantiene casi íntegramente, aunque claramente simplificado.

Se proponen cuatro procesos básicos: la explosión, que es la generación de partículas con velocidad y trayectoria aleatorias; la interacción entre partículas, que es el cálculo de las nuevas trayectorias y el movimiento de cada partícula; la evaluación, que determina la masa de las partícula y los reinicios, que es la estrategia de explotación. Los procesos planteados son simples y se pueden implementar computacionalmente.

En el siguiente capítulo se implementará el algoritmo planteado y se estudiarán las diferentes características de la heurística, evaluando su adecuación y funcionamiento. También se estudiará el comportamiento de la misma con sus diferentes parámetros.

3. IMPLEMENTACIÓN Y PARAMETRIZACIÓN

Los métodos metaheurísticos se utilizan comúnmente en la optimización, aunque han sido cuestionados por ser métodos directos tipo caja negra que en su mayoría no tienen una prueba matemática estricta, a excepción de algunas definiciones y generalizaciones (Granville et al., 1994; Torczon, 1997); aún así, las heurísticas han resuelto problemas difíciles, tanto teóricos como reales (Törn y Zilinskas, 1989).

La postulación de un nuevo algoritmo heurístico exige que se demuestre su capacidad para encontrar la solución global del problema, o al menos, soluciones suficientemente próximas; esto se logra directamente mediante la demostración matemática de la convergencia del algoritmo a un punto de óptima local. O mediante pruebas empíricas, demostrando tal convergencia por medio de experimentos computacionales en los que se optimizan funciones conocidas; en este último caso, se analiza el comportamiento del algoritmo para criterios dados como la calidad de la respuesta encontrada, el esfuerzo computacional y robustez (entendida en el sentido de ser capaz de encontrar soluciones suficientemente buenas para un número grande de problemas) (Jang et al., 1996; Himmelblau, 1972).

Toda heurística novedosa o aportes metodológicos a las heurísticas existentes presentan algunas pruebas iniciales de funcionamiento; la recopilación y análisis de estas pruebas muestran que el desarrollo de una heurística se puede dividir en tres fases principales: la implementación, la parametrización y la validación (Muller et al., 2000; Kennedy y Eberhart, 1995; Erol y Eksin, 2006; Dorigo y Stützle, 2003; Resende y Ribeiro, 2002; Talbi, 2009; Kirkpatrick et al., 1983). En este caso se presentan los resultados de las fases de implementación y parametrización.

En la Sección 3.1 se presenta los criterios que se van a evaluar en la heurística. En la Sección 3.2 se describen las funciones que se utilizan para probar el funcionamiento preliminar de la heurísticas. En la Sección 3.3 se muestran las pruebas de la fase de implementación. En la Sección 3.4. se presentan los resultados de la parametrización. Finalmente, en la Sección 3.5 se concluye el trabajo realizado.

3.1. ¿QUÉ EVALUAR?

Cada fase de desarrollo en una heurística tiene su objetivo y por ende, pruebas específicas que evalúan dicho objetivo. Seguidamente se describen los aspectos a evaluar para las etapas de implementación y parametrización.

3.1.1. Implementación

La fase de implementación tiene como objetivo principal desarrollar un prototipo funcional del concepto, las supernovas, que permita ajustar la heurística y conocer la mayor cantidad de características posibles: ventajas, desventajas y otros aspectos claves en la implementación.

Las pruebas en la etapa de implementación tienen dos fines específicos: primero, evaluar la implementación como tal, garantizando que no hayan errores lógicos; y segundo, analizar la convergencia del algoritmo y detectar las falencias derivadas de aspectos no considerados en el diseño preliminar de la heurística.

Las heurísticas al ser métodos tipo caja negra, no tienen un comportamiento analítico determinado; por eso es deseable tener una idea clara del funcionamiento de la idea ya implementada. Una vez se comprueba que el funcionamiento de la heurística es lógicamente correcto y sigue las estrategias de la manera esperada, se hacen las pruebas de convergencia inciales.

En las heurísticas, como proceso iterativo, la convergencia es un comportamiento asintótico hacia el mínimo global. Las gráficas del mínimo encontrado contra las iteraciones o llamadas a la función objetivo, idealmente debe tener este comportamiento (ver la Figura 3.1), pues, esto garantiza que en un número de iteraciones el algoritmo convergerá a una respuesta específica. Dependiendo de características como la pendiente y el comportamiento de esta curva para un problema dado permite la implementación de criterios de parada que acorten el proceso, logrando una buena respuesta en menor tiempo (Winker, 2001; Fogel, 1994).

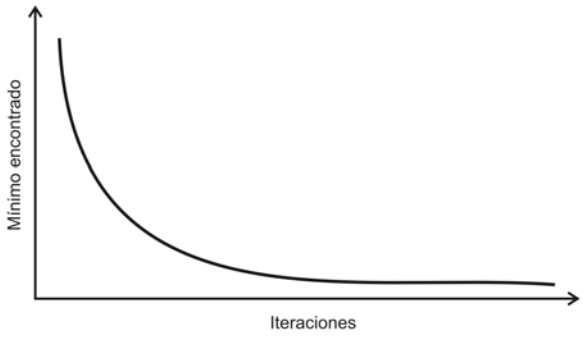


Figura 3.1: Comportamiento asintótico.

El comportamiento y las pruebas de convergencia se basan en el uso de gráficas de resultados contra número de llamadas a la función objetivo o contra otros parámetros escogidos para observar la evolución del mínimo encontrado. Para alcanzar el objetivo de las pruebas en esta fase del problema se plantean las siguientes pruebas:

- Mostrar el funcionamiento interno dibujando la trayectoria de búsqueda de la heurística; esta prueba

debería mostrar un comportamiento parecido a la Figura 2.1.

- La convergencia inicial se muestra en gráficas del valor mínimo encontrado contra las iteraciones, tanto para la heurística completa como para sus partes individuales, constatando el comportamiento asintótico esperado.
- La prueba de robustez inicial se conforma de diferentes pruebas, como el valor mínimo encontrado contra el número de dimensiones de la función, los óptimos para diferentes puntos de inicio, y los mínimos para las funciones trasladadas. Estas pruebas proveen información sobre la consistencia de la heurística para diferentes problemas.
- Finalmente, se obtiene una idea del costo computacional necesario tomando el tiempo requerido para correr la heurística con diferentes parámetros.

3.1.2. Parametrización

Luego de los resultados preliminares se inicia el proceso de parametrización. Este proceso es de suma importancia para una heurística pues el valor de los parámetros está directamente relacionado con el proceso de búsqueda y la solución encontrada, pues, para ciertos valores la heurística tiene un comportamiento más eficiente que para otros valores. La fase de desarrollo busca encontrar valores, expresar parámetros en función de otros o ecuaciones que utilicen los datos propios del problema que permitan la optimización eficiente de los problemas propuestos y la variación respecto a las diferentes funciones propuestas de acuerdo a sus características (Törn y Zilinskas, 1989).

Con la información de las pruebas iniciales, de acuerdo a la sensibilidad mostrada y el funcionamiento observado para las funciones, se pueden determinar iteraciones y rangos para cada grupo de valores de los parámetros iniciales. Con los rangos dados para los diferentes parámetros se construyen gráficas que permitan observar el comportamiento de cada parámetro o para el rango de valores permitiendo analizar la estabilidad del parámetro en el algoritmo (Kennedy y Eberhart, 1995). También es importante comparar los diferentes valores del parámetro y el funcionamiento de los mismos con el número de iteraciones.

3.2. FUNCIONES DE PRUEBA UTILIZADAS

En la etapa de implementación inicial se analiza el comportamiento del algoritmo en funciones unimodales de muy baja complejidad que se puedan rotar y trasladar fácilmente; de manera que, se pueda observar claramente el comportamiento de la heurística (Muller et al., 2000; Jang et al., 1996). Mientras que, en la fase de parametrización se requiere que el algoritmo sea probado en funciones diferentes como (Bäck y Schwefel, 1993; Feoktistov, 2006):

- Funciones continuas unimodales.

- Funciones continuas multimodales.

- Funciones discretas

Se recomienda tomar en esta fase al menos una de las funciones de los tipos mencionados; se puede utilizar algún test ya establecido, como el de De Jong que se ha utilizando ampliamente en la literatura. Lo conforman 5 funciones: 2 unimodales, una discontinua unimodal y dos multimodales. Las diferentes características de las funciones proveen la información necesaria para dar, al menos, rangos de los valores que pueden tomar los parámetros para el funcionamiento óptimo de una heurística para los diferentes tipos de problemas generales (Kennedy y Eberhart, 1995; Jong, 1975). También se han propuesto otros conjuntos de funciones con diferentes características (Schwefel, 1993; Yao et al., 1996; Törn y Zilinskas, 1989; Vesterstrom y Thomsen, 2004).

Para las pruebas se utilizaron 6 funciones ampliamente utilizadas, con diversas configuraciones que tienen el balance entre complejidad y diversidad necesario para estudiar la heurística. Las dimensiones, la región de búsqueda y el mínimo se pueden observar en la Tabla 3.1 y sus gráficas en la Figura 3.2; de estas funciones se tomaron inicialmente 3 para la fase de implementación y 5 en la fase de parametrización.

Tabla 3.1: Funciones de prueba para las etapas de implementación y parametrización

Función de prueba	Dim(D)	Región	mín
$f_1(x) = \sum_{i=1}^{D} x_i^2$	2	$[-500, 500]^D$	0
$f_2(x) = \sum_{i=1}^{D} 100(x_i^2 - x_{i+1})^2 + (1 - x_i^2)^2$	2	$[-5{,}12, 5{,}12]^D$	0
$f_3(x) = -20 \cdot exp(-0{,}2\sqrt{\frac{1}{D} \cdot \sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D} \cdot \sum_{i=1}^{D} \cos(2\pi x_i)) + 20 + exp(1)$	2	$[-30, 30]^D$	0
$f_4(x) = \sum_{i=1}^{D} (\lfloor x_i + 0{,}5 \rfloor)^2$	2	$[-5{,}12, 5{,}12]^D$	0
$f_5(x) = \sum_{i=1}^{D} i(x_i^4) + random[0, 1)$	2	$[-1{,}28, 1{,}28]^D$	0
$\frac{1}{f_6(x)} = \frac{1}{500} \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}$ $a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & -16 & \dots & 32 & 32 & 32 \end{pmatrix}$	2	$[-65{,}536, 65{,}356]^D$	≈ 1

Para evaluar el desempeño en la implementación se tomaron tres funciones de prueba f_1, f_2 y f_3 utilizadas clásicamente en optimización global; inicialmente se toman en dos dimensiones. Las tres funciones escogidas presentan diferentes características, como regiones planas y muchos mínimos locales. Las funciones f_1 y f_2 conocidas como la función esférica y Rosenbrock respectivamente, son funciones unimodales convexas para dos dimensiones. La función f_3 (función de Ackley) es una función multimodal, no separable y aumenta sus mínimos locales al aumentar el número de dimensiones (Brest et al., 2006).

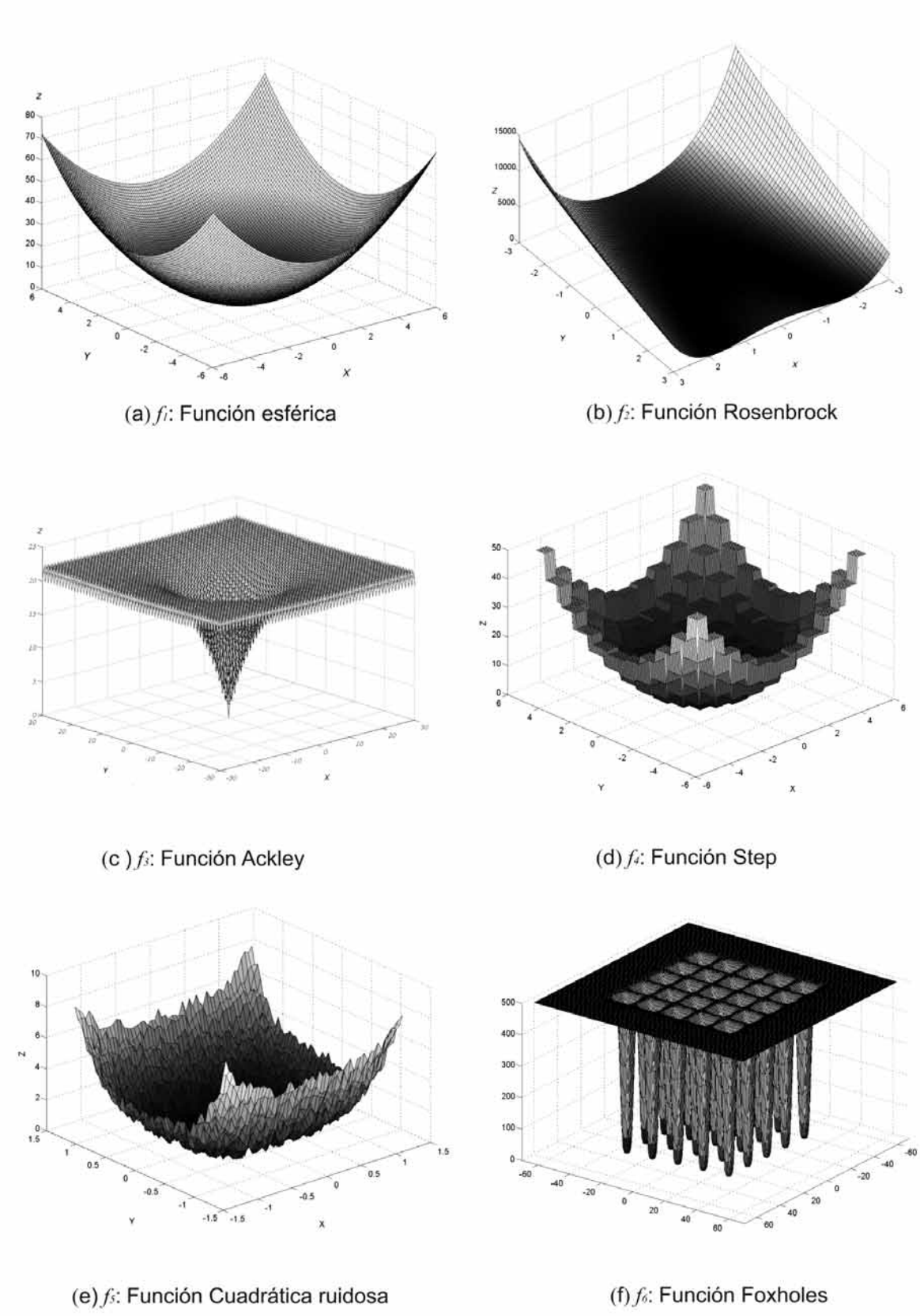


Figura 3.2: Funciones de prueba.

En la fase de parametrización se necesita observar el comportamiento en un número más amplio de casos, así que se usa el test De Jong que consta de 5 funciones f_1, f_2, f_4, f_5 y f_6. La función f_4 es conocida como la función paso, es discontinua, convexa con un único mínimo. La función f_5 corresponde a la función cuadrática con ruido de Gauss que es convexa, unimodal ruidosa y la f_6, conocida como la función de trinchera, propuesta por Shekel, es una función continua, no convexa, bidimensional, no cuadrática (Jong, 1975).

3.3. IMPLEMENTACIÓN

La heurística descrita en el capítulo anterior se implementó comprobando que la forma de búsqueda emule el proceso del cual recibe su inspiración; también se describe el comportamiento individual de cada uno de los procesos de búsqueda de la heurística para las funciones de prueba observando su comportamiento individual y en conjunto, y los resultados para diferentes disposiciones de las funciones de prueba.

3.3.1. Proceso de búsqueda

Se implementó la heurística completa y se probó para la función esférica f_1. En la Figura 3.3 se observa el proceso de búsqueda, en la parte (*a*) se muestra una población inicial de 5 individuos aleatorios uniformemente distribuidos en el espacio de búsqueda; en la parte (*b*) se observan las trayectoria que siguen cada partícula en 15 iteraciones: tras la explosión hay una pequeña expansión inicial y luego un redireccionamiento hacia las mejores partículas. Luego en la parte (*c*), se observa el mejor punto encontrado, y sobre este punto se reinicia el algoritmo explotando 5 individuos aleatorios dispersos sobre el una área mas pequeña como se muestra en la parte (*d*) de la figura.

El comportamiento observado en la trayectoria de búsqueda de la implementación corresponde a la forma de búsqueda esperada de la heurística (Ver Figura 2.1), inicialmente hay una expansión como se puede observar en la parte (*b*) de la Figura y luego una atracción hacia los mejores puntos encontrados; en este trayecto, hay una exploración de otras partes de la región.

3.3.2. Estrategia de exploración y estrategia de explotación

Una heurística de optimización global debe complementar una estrategia de exploración con otra de explotación para lograr buenas respuestas cercanas al mínimo. En el caso de Supernova, se divide en dos partes: la heurística interna que presenta los cálculos de la inercia y las fuerzas gravitacionales, y esta sería la estrategia de exploración a nivel macro. Y al reiniciar el algoritmo con menos fuerza se logra un refinamiento y explotación de un área específica más pequeña (Törn y Zilinskas, 1989).

Inicialmente se probó la heurística interna sin reiniciar, para las funciones de prueba f_1, f_2 y f_3. El comportamiento obtenido para 25 corridas independientes se muestra en la Figura 3.4. El proceso de cálculos

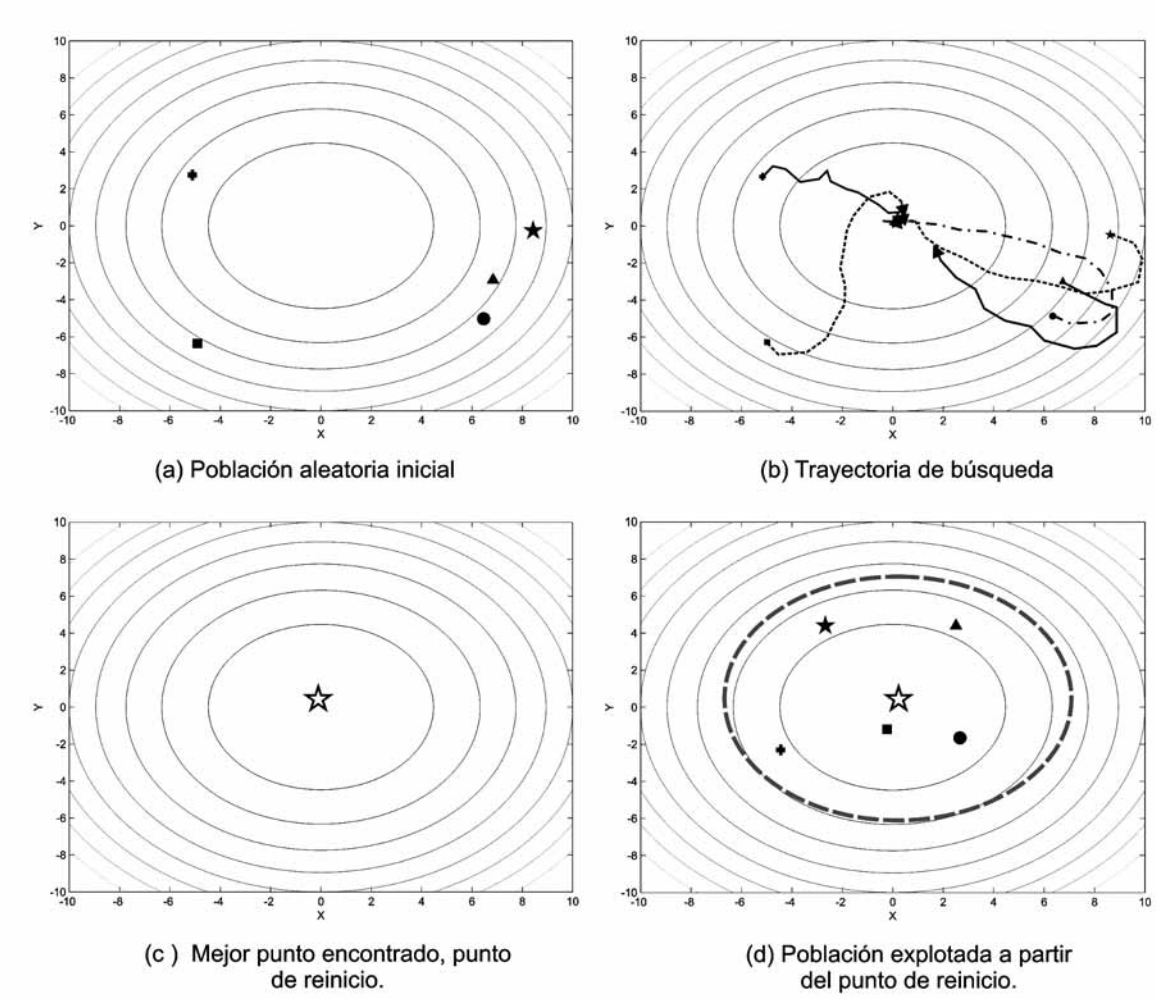


Figura 3.3: Proceso de búsqueda función esférica f_1

internos presenta un comportamiento bastante estable para la función f_1 y asintótico, mientras que para las funciones f_2 y f_3 tiene un comportamiento más inestable y llega a respuestas menos precisas.

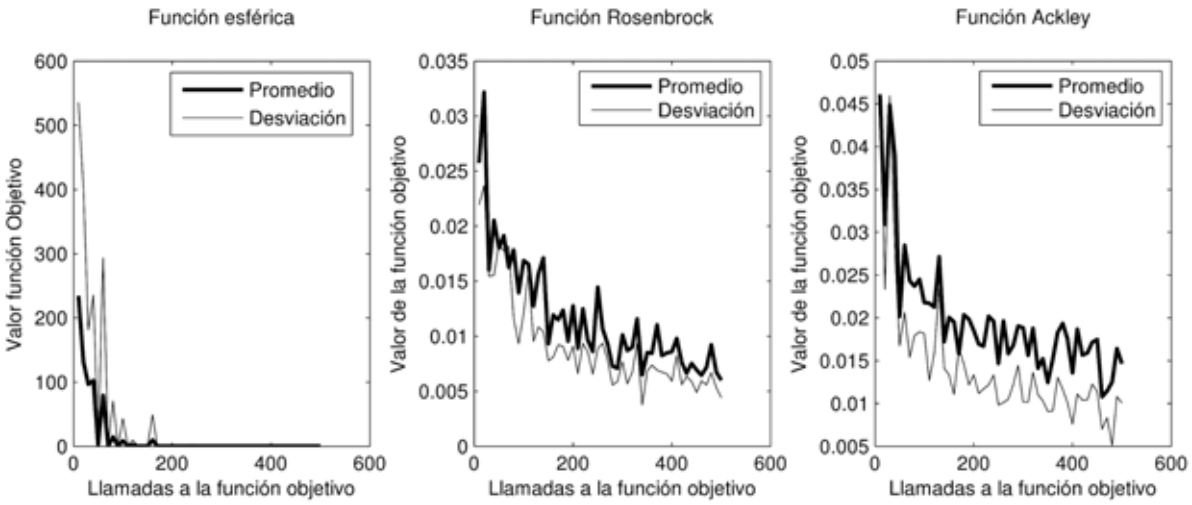


Figura 3.4: Resultados para la función sin reinicios, para 25 corridas independientes.

Luego se probó la estrategia de los reinicios por si sola, obteniendo los resultados de la Figura 3.5. Para cada función se ve un comportamiento que baja y sube sin importar el número de llamadas a la función objetivo, es decir que no se puede garantizar la convergencia usando solamente la estrategia de los reinicios. Para los resultados de la función esférica se aprecia un mejoramiento, pero luego de las 100 llamadas a la función objetivo, por si sola esta estrategia no indica comportamiento convergente .

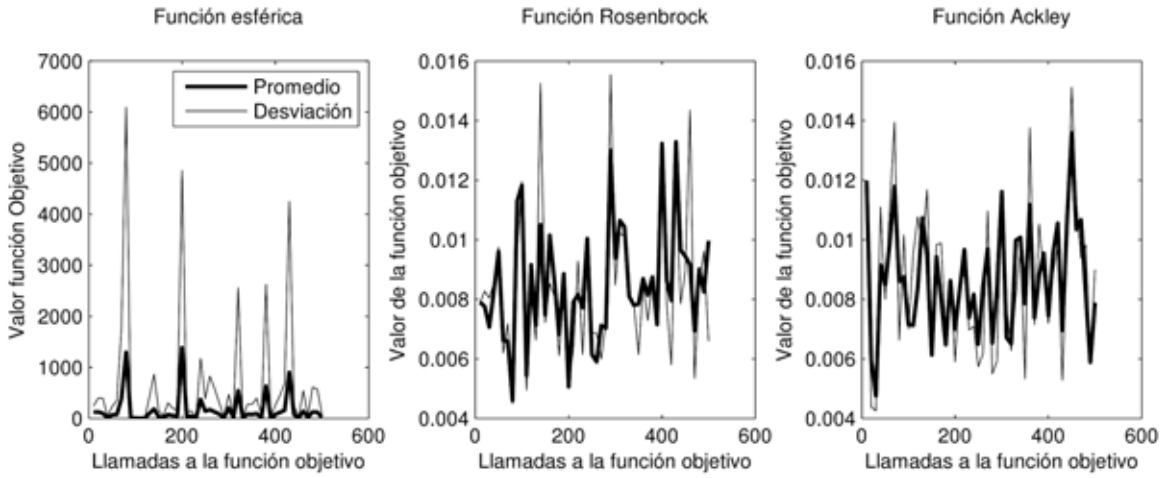


Figura 3.5: Resultados solo reiniciando en el mejor punto, 25 corridas independientes.

Finalmente, se aplicó la heurística completa y se tomaron datos de su comportamiento, los cuales se muestran en la Figura 3.6. En este caso se aprecia para las funciones f_1 y f_3 unas respuestas más cercanas al

mínimo, del orden de 1×10^{-18} para f_1 y 1×10^{-8} para f_3, también el comportamiento para estas funciones es más estable que en los casos anteriores y presenta una clara tendencia asintótica. La función f_2, presenta una disminución en el mínimo pero su comportamiento es tan estable como en los casos anteriores y el mínimo encontrado difiere mucho más que los otros casos del mínimo real.

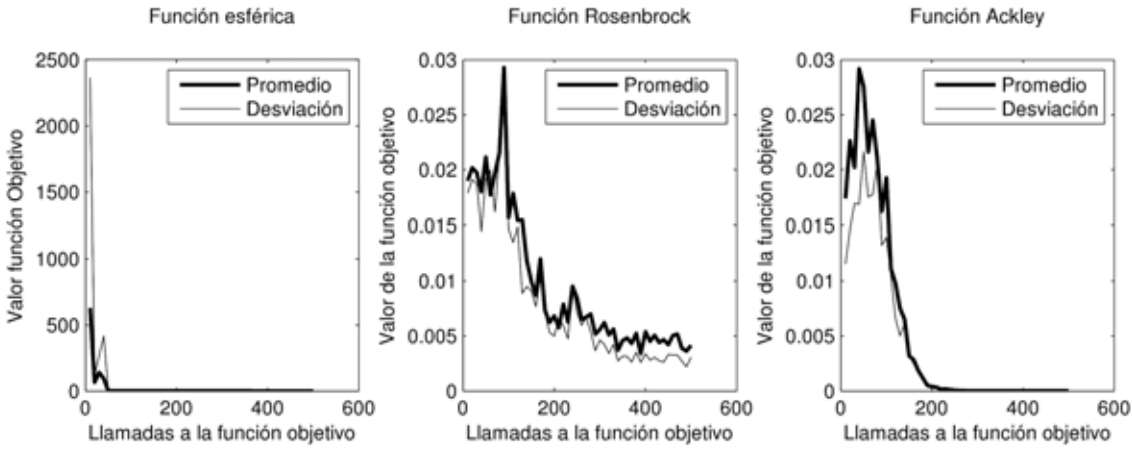


Figura 3.6: Resultados para la heurística completa, 25 corridas independientes.

3.3.3. Robustez

La robustez en una heurística está definida por su capacidad de obtener una respuesta con cierto grado de calidad en diferentes rangos de problemas (Himmelblau, 1972). Para probar la robustez de la heurística se observará su comportamiento al aumentar las dimensiones del problema, lo que aumenta su dificultad. Se harán pruebas para diferentes puntos de inicio utilizando la explosión inicial desde un punto dado x_0 y se modificarán las funciones de prueba rotando y trasladando la función, modificando las características. Y mostrando el funcionamiento para problemas de inicio no centrados, y con mínimos en puntos no enteros.

Estudio dimensional de las funciones

Al aumentar el número de dimensiones de un problema aumentan los retos para el método de optimización utilizado; aunque en la función f_1 el número de mínimos no aumenta, la precisión requerida para encontrar buenas respuestas es mucho mayor. En la función f_2, el punto de mínima global se va constituyendo en la forma de una aguja, además al ir aumentando la dimensionalidad se convierte en una función multimodal (Shang y Qiu, 2006). La función f_3 aumenta sus mínimos locales y profundiza la rugosidad de la función exponencialmente a medida que se aumentan las dimensiones, por lo tanto podría esperarse un crecimiento en el valor encontrado de acuerdo a la función y los parámetros utilizados (Brest et al., 2006; Yao et al., 1996).

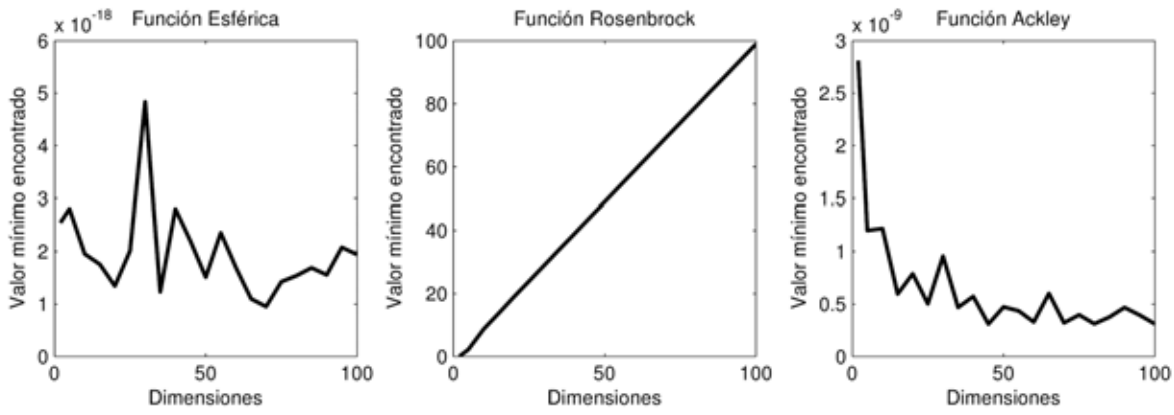


Figura 3.7: Dimensiones Vs. resultados para supernova para las funciones de prueba f_1, f_2 y f_3, promedio de 30 corridas independientes.

En la Figura 3.7 se observa el comportamiento del mínimo encontrado contra la dimensión, las variaciones para las funciones f_1 y f_3 son del orden de 1×10^{-17} para f_1 o 1×10^{-7} para f_3, mientras que para la función f_2 es del orden de 1×10^2. Se observa que el comportamiento es más estable para la función f_3 mientras que en la función f_2 las repuestas encontradas son de más baja calidad, aunque esto es lo esperado por al aumentar de las dimensiones en una función no separable (Vesterstrom y Thomsen, 2004; Yao et al., 1996).

Incidencia del punto de inicio

Al utilizar un único punto de inicio x_0, es importante observar su incidencia en el funcionamiento del algoritmo; se probaron cuatro puntos de inicio distribuidos en diferentes partes de cada región, y así, determinar la incidencia del punto de inicio en el funcionamiento del algoritmo (Muller et al., 2000).

Tabla 3.2: Resultados para los diferentes puntos de inicio y funciones de prueba

Función esférica (f_1)			Función Rosenbrock (f_2)			Función Ackley (f_3)		
x_0	$\overline{f*}$	σ	x_0	$\overline{f*}$	σ	x_0	$\overline{f*}$	σ
[2,-3]	$1{,}25 \times 10^{-18}$	$2{,}29 \times 10^{-18}$	[2,-3]	0,0022	0,0023	[2,-3]	$0{,}27 \times 10^{-8}$	$0{,}31 \times 10^{-8}$
[-500,500]	$1{,}28 \times 10^{-18}$	$1{,}85 \times 10^{-18}$	[-30,30]	0,0024	0,0027	[-32,32]	$0{,}31 \times 10^{-8}$	$0{,}30 \times 10^{-8}$
[200,350]	$3{,}32 \times 10^{-18}$	$5{,}33 \times 10^{-18}$	[12,23]	0,0016	0,0018	[12,23]	$0{,}29 \times 10^{-8}$	$0{,}29 \times 10^{-8}$
[-500,-500]	$1{,}28 \times 10^{-18}$	$2{,}22 \times 10^{-18}$	[-30,-30]	0,0018	0,0021	[-32,-32]	$0{,}30 \times 10^{-8}$	$0{,}25 \times 10^{-8}$

En la Tabla 3.2 se muestran los resultados de las pruebas. Para 4 puntos de inicio x_0 se da el valor promedio del mínimo encontrado en las diferentes corridas $\overline{f*}$ y la desviación estándar σ. La diferencia entre los promedios encontrados es similar a las desviaciones estándar para cada función, lo que sugiere un

comportamiento estable ante el cambio de punto de inicio.

Funciones modificadas

Es importante analizar otros comportamientos como: funciones con mínimos negativos, no enteros y mínimos no simétricos. Para probar la robustez del algoritmo en estos aspectos se traslada y rotan las funciones de prueba de modo que tengan mínimos negativos, no enteros y/o regiones no simétricas. Inicialmente se trasladan las funciones en el eje z de acuerdo a $z' = z + \Delta z$ donde Δz es un número cualquiera, negativo y no entero $\Delta z = -3{,}845$. Esta traslación modifica los mínimos de las funciones f_1, f_2 y f_3 originalmente en 0 por un mínimo en -3.845, permitiendo observar el comportamiento del algoritmo en funciones con mínimos negativos y no enteros. En la Tabla 3.3 se observan los datos del promedio y la desviación para 50 corridas independientes; los resultados muestran una buena respuesta del algoritmo para funciones con mínimos negativos y no enteros.

Tabla 3.3: Promedio y desviación para la traslación de la funciones en el eje z

Función	$\overline{f*}$	σ
Función esférica	-3.8450	0
Función Rosenbrock	-3.8427	0.0026
Función Ackley	-3.850	0

Las funciones f_1 y f_3 son simétricas, tienen sus mínimos en el centro de la región de búsqueda, y aunque la función f_2 no es totalmente simétrica, su mínimo esta cerca del centro de la región. Cambiando la disposición de los mínimos se tiene problemas nuevos con mayor dificultad por las disposiciones. Para las funciones de pruebas f_1, f_2 y f_3 se trasladarán dos ejes coordenados, de acuerdo a: $x_i' = x_i + \Delta x_i$. De esta manera los mínimos de las regiones cerca de los bordes en diferentes disposiciones, los resultados y detalles se observan en la Tabla 3.4. Se nota una disminución en la calidad del mínimo encontrado para las funciones, lo que sugiere que hay mayor dificultad para Supernova cuando las funciones tienen los mínimos cercanos a los bordes.

Tabla 3.4: Promedio y desviación para la traslación de la funciones en el eje x y y

Función	Δx_1	Δx_2	$\overline{f*}$	σ
Función esférica	450.698	250.578	$0{,}2098 \times 10^{-3}$	0.6008
Función Rosenbrock	-15.897	14.652	0.0017	0.0017
Función Ackley	-23.478	-28.328	0.0188	0.0120

Tabla 3.5: Tiempo de corrida de acuerdo a las dimensiones y número de partículas para f_1, f_2 y f_3

Iteraciones	Reinicios	Número de partículas	Dimensiones	Tiempo de corrida[s]
100	10	10	2	0.5
100	10	100	2	1.614
100	50	10	2	1.79
10	50	100	30	2.63
100	50	100	2	6.42
100	50	10	30	8.72
100	50	100	30	60.176

3.3.4. Costo computacional

El algoritmo se corrió en un procesador Pentium D 925, a 3,0 GHz con FSB de 800 MHz, 2GB de RAM. Para estas especificaciones se muestran en la Tabla 3.5 los resultados del tiempo en segundos para CPU para diferentes dimensiones y partículas de las funciones de prueba f_1, f_2 y f_3.

Como se observa en la Tabla, los factores críticos de aumento de costo computacional son las dimensiones, en su orden sigue el número de partículas y finalmente el número de iteraciones.

3.4. PARAMETRIZACIÓN

En esta parte del proceso se analiza más profundamente los diferentes parámetros. Aunque con las pruebas anteriores se presentan unos rangos, en esta sección se estudiará más a fondo para las funciones del test de De Jong, correspondiente a las funciones f_1, f_2,f_4,f_5 y f_6 de la Tabla 3.1. Se evalúan los diferentes parámetros para cada una de las funciones reportando los de mayor éxito. Es importante resaltar que estas funciones son bidimensionales para el conjunto de funciones escogido.

El algoritmo utiliza siete parámetros independientes y son: las constantes de impulso y gravitación F y G, el número de partículas k, el número de iteraciones T, el número de reinicios RR, la fuerza de explosión λ y la tasa de disminución de esta variables α.

El número de iteraciones y reinicio tienen un criterio de parada de acuerdo a la convergencia del algoritmo, luego de x número de iteraciones sin mejorar la respuesta termina el proceso. Para las pruebas, este criterio se ha tomado como el 10 % de las iteraciones dadas.

3.4.1. Número de partículas

El número de partículas es un factor importante en la estabilidad del algoritmo. Se hicieron pruebas de la heurística interna con el número de partículas; los resultados se muestran en la Figura 3.8. Se observa que

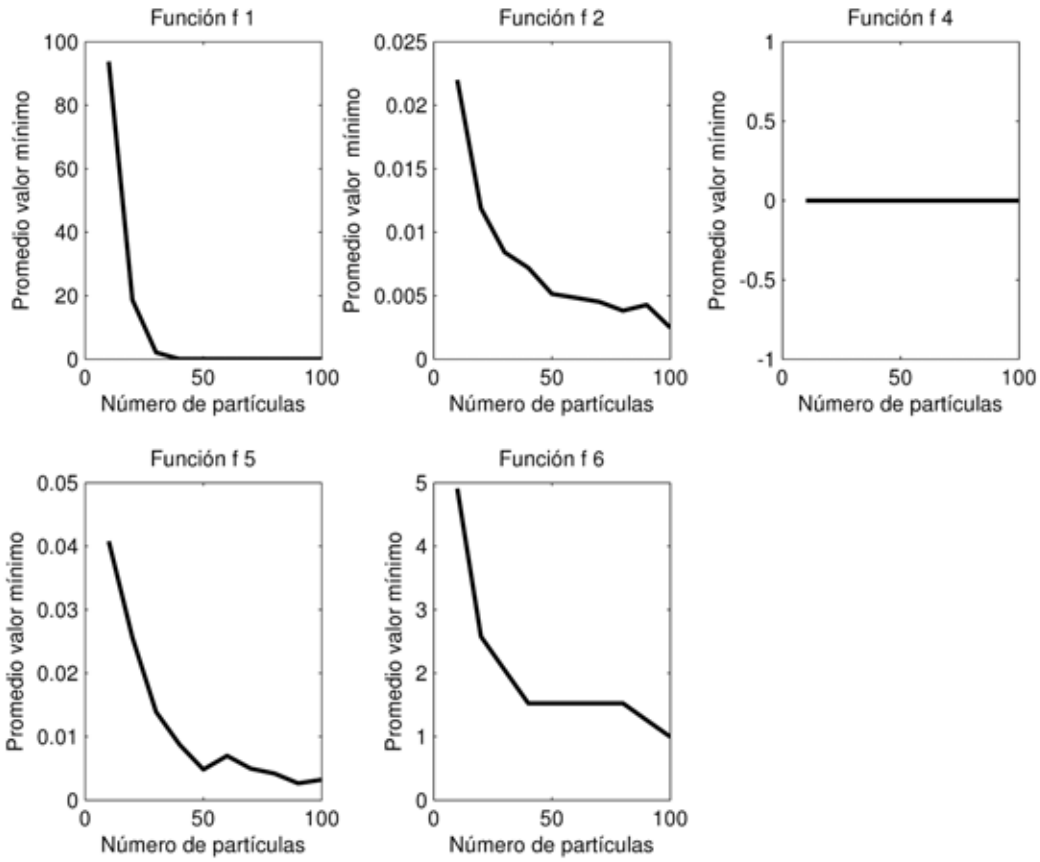


Figura 3.8: Resultados de las funciones de prueba para el número de partículas.

para las cinco funciones analizadas los promedios del mínimo encontrado se estabiliza para tamaños entre 80 y 100, rango de valores recomendado.

3.4.2. Parámetros de Impulso y atracción

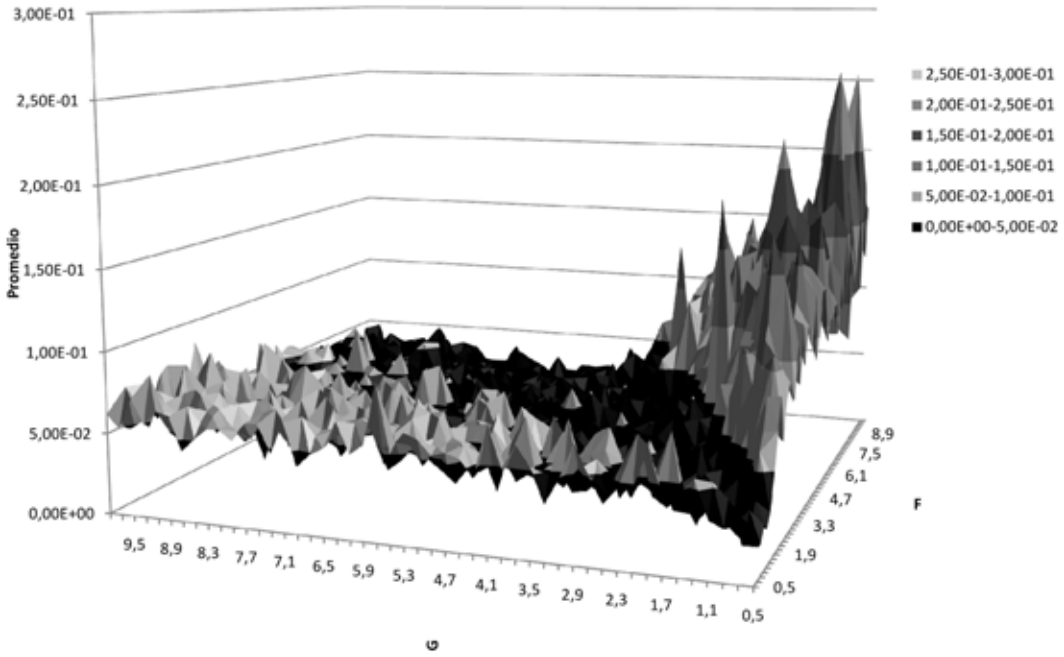


Figura 3.9: Resultado promedio de las funciones de prueba para F y G.

Las funciones de prueba se corrieron para diferentes valores de F (impulso) y G (atracción) en el rango $[0{,}5, 10]$. En la Figura 3.9 se observa el promedio encontrado para los diferentes valores en la función f_2. Las otras funciones muestran un comportamiento similar; se observa que la heurística tiene un mejor comportamiento para valores de F y G cercanos entre sí, como se observa en la zona oscura de la gráfica, se recomienda que esta diferencia sea menor de 2. En la Tabla 3.6 se especifican las tres parejas de F y G que mostraron el mejor promedio para cada función en el rango estudiado. Para cada función se observa un buen comportamiento de la pareja F=1 y G=1, por lo tanto se recomienda que para funciones nuevas se tome estos valores. En el caso de la función f_4 es poco sensible a los valores de F y G, con los valores de los otros parámetros logra alcanzar el mínimo sin importar los valores que tomen F y G

3.4.3. Número de iteraciones y reinicios

El número de iteraciones y reinicios controla la extensión de la búsqueda: un número pequeño de iteraciones y reinicios significa una búsqueda rápida. Es deseable tener el número más pequeño para el cual el algoritmo se estabilice y encuentre una respuesta dentro de los límites deseados. En la Tabla 3.7 se observan

Tabla 3.6: Tres mejores promedios encontrados para los parámetros F y G para la funciones de prueba

Función	F	G	F	G	F	G
f_1	3	3	1	1	4	1
f_2	2	1	1.5	1.5	9	4.5
f_4	0 para todos los parámetros					
f_5	1	1	2	2.5	1	6.5
f_6	1	1	4	6	2	3.5

los resultados de las pruebas. Para la función f_1 se observa que el aumento de reinicios aumenta la precisión del algoritmo. La función f_2 no cambia mucho en el rango estipulado, para la función f_4 siempre se llega al mínimo de la función. Para la función f_5 no se varía la precisión encontrada, especialmente por la variable aleatoria de la función, y f_6 al contrario de la primera función se estabiliza al aumentar las iteraciones y permanece casi invariable para los reinicios.

3.4.4. Fuerza de la explosión y factor de disminución

La fuerza de explosión λ y el factor de disminución α son los que determinan el circulo de búsqueda del reinicio (ver Figura 3.3 parte *d*). A medida que se va reiniciando, va disminuyendo su radio en un factor α. En la Figura 3.10 se observan los resultados de las pruebas para las diferentes funciones.

Los parámetros λ y α controlan la cantidad de espacio explorado por lo tanto si es una función desconocida es importante colocar valores acordes a la región objetivo, los valores mas grandes de λ deben ir acompañados de valores grandes de α para garantizar una buena calidad de respuesta. En la Tabla 3.8 se muestran los valores recomendados para una función desconocida de acuerdo al tamaño de su región de búsqueda.

3.4.5. Parámetros

Supernova tiene siete parámetros básicos que se han estudiado en las secciones anteriores. En la Tabla 3.9 se resumen los parámetros encontrados que de acuerdo al promedio y su desviación estándar que muestran el mejor comportamiento del algoritmo para cada una de las funciones.

3.5. CONCLUSIONES

La heurística probó su comportamiento asintótico en las funciones propuestas, lo que representa una prueba empírica de convergencia. La prueba de los procesos independientes de la heurística mostró que completa la metaheurística, tiene un mejor comportamiento en su totalidad que cualquiera de sus procesos

Tabla 3.7: Valor promedio encontrado para diferente número de iteraciones y reinicios

f	reinicios	Iteraciones					
		100	200	300	400	500	600
f_1	10	$3{,}27 \times 10^{-5}$	$3{,}68 \times 10^{-5}$	$2{,}94 \times 10^{-5}$	$3{,}02 \times 10^{-5}$	$2{,}36 \times 10^{-5}$	$2{,}33 \times 10^{-5}$
	20	$7{,}65 \times 10^{-8}$	$8{,}15 \times 10^{-8}$	$7{,}87 \times 10^{-8}$	$6{,}67 \times 10^{-8}$	$7{,}53 \times 10^{-8}$	$8{,}53 \times 10^{-8}$
	30	$1{,}89 \times 10^{-11}$	$1{,}73 \times 10^{-11}$	$1{,}96 \times 10^{-11}$	$2{,}01 \times 10^{-11}$	$2{,}41 \times 10^{-11}$	$1{,}96 \times 10^{-11}$
	40	$0{,}43 \times 10^{-14}$	$0{,}70 \times 10^{-14}$	$0{,}34 \times 10^{-14}$	$0{,}25 \times 10^{-14}$	$0{,}56 \times 10^{-14}$	$0{,}71 \times 10^{-14}$
	50	$1{,}65 \times 10^{-18}$	$1{,}51 \times 10^{-18}$	$0{,}65 \times 10^{-18}$	$0{,}99 \times 10^{-18}$	$2{,}38 \times 10^{-18}$	$1{,}26 \times 10^{-18}$
	60	$0{,}83 \times 10^{-21}$	$0{,}94 \times 10^{-21}$	$0{,}77 \times 10^{-21}$	$0{,}64 \times 10^{-21}$	$0{,}49 \times 10^{-21}$	$0{,}32 \times 10^{-21}$
	70	$1{,}68 \times 10^{-25}$	$1{,}05 \times 10^{-25}$	$1{,}77 \times 10^{-25}$	$2{,}35 \times 10^{-25}$	$1{,}20 \times 10^{-25}$	$1{,}53 \times 10^{-25}$
	80	$8{,}57 \times 10^{-29}$	$4{,}50 \times 10^{-29}$	$2{,}47 \times 10^{-29}$	$2{,}58 \times 10^{-29}$	$5{,}68 \times 10^{-29}$	$4{,}87 \times 10^{-29}$
	90	$1{,}40 \times 10^{-32}$	$1{,}40 \times 10^{-32}$	$0{,}97 \times 10^{-32}$	$1{,}20 \times 10^{-32}$	$1{,}32 \times 10^{-32}$	$0{,}65 \times 10^{-32}$
	100	$5{,}20 \times 10^{-36}$	$2{,}26 \times 10^{-36}$	$6{,}84 \times 10^{-36}$	$4{,}19 \times 10^{-36}$	$6{,}21 \times 10^{-36}$	$4{,}62 \times 10^{-36}$
f_2	10	$2{,}5 \times 10^{-3}$	$1{,}2 \times 10^{-3}$	$1{,}1 \times 10^{-3}$	$1{,}0 \times 10^{-3}$	$1{,}2 \times 10^{-3}$	$1{,}1 \times 10^{-3}$
	20	$2{,}2 \times 10^{-3}$	$0{,}7 \times 10^{-3}$	$1{,}0 \times 10^{-3}$	$0{,}5 \times 10^{-3}$	$0{,}6 \times 10^{-3}$	$0{,}9 \times 10^{-3}$
	30	$3{,}2 \times 10^{-3}$	$2{,}0 \times 10^{-3}$	$1{,}1 \times 10^{-3}$	$1{,}2 \times 10^{-3}$	$0{,}8 \times 10^{-3}$	$0{,}7 \times 10^{-3}$
	40	$2{,}3 \times 10^{-3}$	$1{,}0 \times 10^{-3}$	$1{,}5 \times 10^{-3}$	$0{,}7 \times 10^{-3}$	$0{,}6 \times 10^{-3}$	$0{,}6 \times 10^{-3}$
	50	$1{,}9 \times 10^{-3}$	$1{,}1 \times 10^{-3}$	$0{,}7 \times 10^{-3}$	$0{,}6 \times 10^{-3}$	$0{,}9 \times 10^{-3}$	$0{,}8 \times 10^{-3}$
	60	$2{,}5 \times 10^{-3}$	$0{,}8 \times 10^{-3}$	$0{,}8 \times 10^{-3}$	$0{,}6 \times 10^{-3}$	$0{,}9 \times 10^{-3}$	$0{,}8 \times 10^{-3}$
	70	$1{,}7 \times 10^{-3}$	$0{,}6 \times 10^{-3}$	$1{,}0 \times 10^{-3}$	$9{,}65 \times 10^{-4}$	$1{,}02 \times 10^{-3}$	$4{,}02 \times 10^{-4}$
	80	$1{,}72 \times 10^{-3}$	$1{,}06 \times 10^{-3}$	$8{,}13 \times 10^{-4}$	$6{,}26 \times 10^{-4}$	$6{,}39 \times 10^{-4}$	$5{,}41 \times 10^{-4}$
	90	$1{,}13 \times 10^{-3}$	$1{,}26 \times 10^{-3}$	$8{,}2 \times 10^{-4}$	$6{,}56 \times 10^{-4}$	$8{,}2 \times 10^{-4}$	$6{,}99 \times 10^{-4}$
	100	$1{,}24 \times 10^{-3}$	$1{,}02 \times 10^{-3}$	$7{,}8 \times 10^{-4}$	$6{,}18 \times 10^{-4}$	$8{,}56 \times 10^{-4}$	$8{,}15 \times 10^{-4}$
f_4	10	0	0	0	0	0	0
	20	0	0	0	0	0	0
	30	0	0	0	0	0	0
f_5	10	$1{,}79 \times 10^{-3}$	$1{,}96 \times 10^{-3}$	$1{,}73 \times 10^{-3}$	$1{,}81 \times 10^{-3}$	$2{,}1 \times 10^{-3}$	$1{,}77 \times 10^{-3}$
	20	$2{,}3 \times 10^{-3}$	$2{,}5 \times 10^{-3}$	$2{,}3 \times 10^{-3}$	$1{,}7 \times 10^{-3}$	$1{,}4 \times 10^{-3}$	$1{,}51 \times 10^{-3}$
	30	$2{,}53 \times 10^{-3}$	$1{,}57 \times 10^{-3}$	$2{,}4 \times 10^{-3}$	$2{,}4 \times 10^{-3}$	$1{,}32 \times 10^{-3}$	$1{,}66 \times 10^{-3}$
	40	$1{,}9 \times 10^{-3}$	$1{,}99 \times 10^{-3}$	$2{,}79 \times 10^{-3}$	$1{,}25 \times 10^{-3}$	$2{,}1 \times 10^{-3}$	$1{,}3 \times 10^{-3}$
	50	$2{,}95 \times 10^{-3}$	$2{,}21 \times 10^{-3}$	$2{,}12 \times 10^{-3}$	$1{,}94 \times 10^{-3}$	$1{,}73 \times 10^{-3}$	$1{,}57 \times 10^{-3}$
f_6	10	1,52	0.9980	0.9980	0.9980	0.9980	0.9980
	20	0.9980	0.9980	1.26	0.9980	0.9980	0.9980
	30	1.26	0.9980	0.9980	0.9980	0.9980	0.9980
	40	1.78	1.26	1.26	0.9980	0.9980	0.9980
	50	1.26	1.52	0.9980	0.9980	0.9980	0.9980
	60	1.26	0.9980	0.9980	0.9980	0.9980	0.9980
	70	2.025	1.26	0.9980	0.9980	0.9980	0.9980
	80	1.26	1.26	0.9980	0.9980	0.9980	0.9980
	90	1.26	1.26	0.9980	0.9980	0.9980	0.9980
	100	2.052	1.26	0.9980	0.9980	0.9980	0.9980

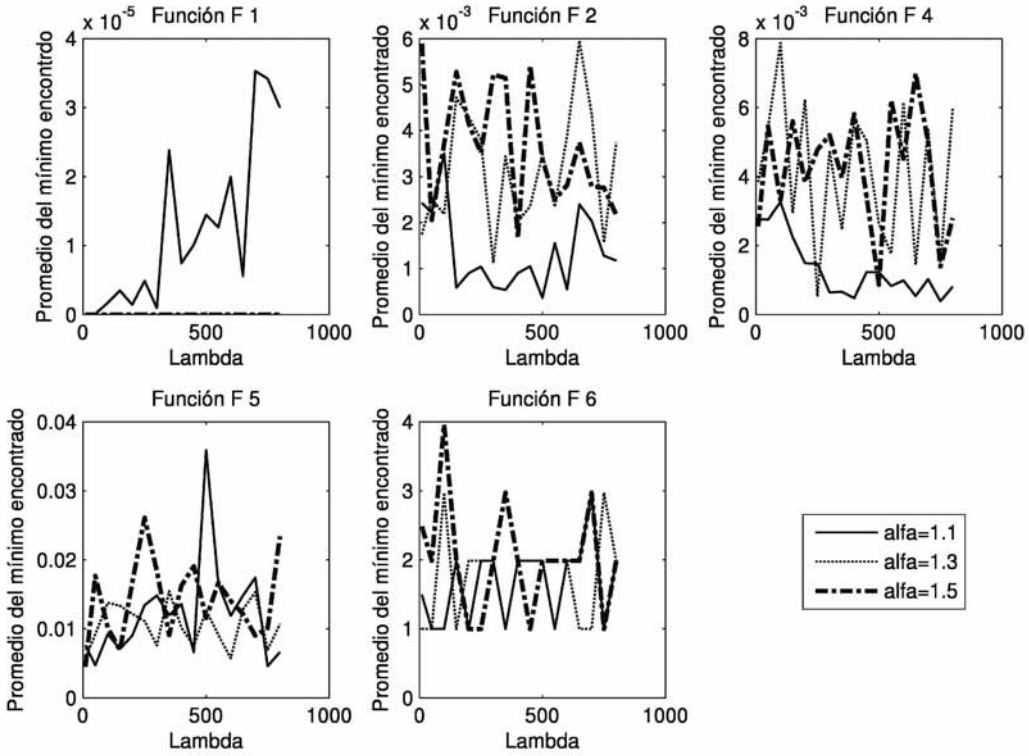


Figura 3.10: Resultado promedio del mínimo encontrado para las funciones de prueba para diferentes valores de λ y α.

Tabla 3.8: Parámetros λ y α recomendados para Supernova según el tamaño de la región de búsqueda

Tamaño de la región	λ	α
>100 unidades	200	1.5
entre 50 y 100 unidades	150	1.2
entre 10 y 50 unidades	50	1.1
<10 unidades	20	1.1

Tabla 3.9: Parámetros recomendados para las funciones

Función	k	F	G	iter	reinicios	λ	α
f_1	90	3	3	200	100	150	1.5
f_2	90	1.5	1.5	400	100	500	1.1
f_4	80	1	1	100	10	400	1.1
f_5	90	1	1	100	10	400	1.1
f_6	90	1	1	400	10	100	1.1

internos. Se mostró un comportamiento satisfactorio para diferente número de dimensiones y en mínimos negativos y no enteros. Al trasladar los mínimos a zonas cercanas a los límites de la región, hay una pequeña disminución de la calidad de la respuesta, pero aún así son respuestas satisfactorias en este punto del proceso. En otra palabras, la heurística mostró un comportamiento robusto.

Al probar los diferentes parámetros para el test propuesto por De Jong, se encontró que los parámetros de impulso y gravedad (F y G) tienen una alta correlación entre sí, y es recomendable que la diferencia entre ambos sea menor a 2. Que el número de partículas al aumentar mejora la calidad de las respuestas y el funcionamiento óptimo, dependiendo de la función está entre 80 y 100, en algunos casos desde 50 funciona bien. El aumento de reinicios e iteraciones mejora la calidad de la respuesta, aunque esta suele estabilizarse luego de un número determinado para cada función. En algunas funciones el número de iteraciones es mas importante y en otros es preferible aumentar los reinicios. Los parámetros λ y α también presentan una estrecha relación entre sí y son los más sensibles del algoritmo.

Con la información de las fases de implementación y parametrización, se procederá a validar la heurística en comparación a otras de la misma naturaleza.

4. VALIDACIÓN

Comprobar que una metaheurística converge y que es robusta no basta para hacerla válida, es importante comparar su comportamiento con otras heurísticas y corroborar que los resultados sean similares ya sea en calidad de la respuesta, robustez o esfuerzo realizado.

En la fase de validación, la heurística no solo se compara sino también se puede determinar en que tipo de problemas puede funcionar con éxito por lo tanto se desea el mayor numero de funciones de prueba que difieran entre así. La evaluación se puede comparar con una o más heurísticas reportadas en la literatura (Feoktistov, 2006; Kennedy y Eberhart, 1995; Muller et al., 2000; Erol y Eksin, 2006; Bäck y Schwefel, 1993). En este capítulo se compararán dos aspectos: calidad de la respuesta al comparar el resultado obtenido por la heurística para las funciones de prueba propuesta; y el esfuerzo al comprar el número de llamadas a la función objetivo necesarias para ese resultado.

En la Sección 4.1 se describen las funciones de pruebas utilizadas. En la Sección 4.2. se reportan los resultados obtenidos. Finalmente en la Sección 4.3 se concluye sobre los resultados hallados.

4.1. FUNCIONES DE PRUEBA

En el proceso de validación se escogieron las veintitres funciones propuestas por Yao et al. (1996) que constan de 5 funciones unimodales, una discontinua y la última una función ruidosa (ver Tabla 4.1). Estas funciones presentan un solo mínimo en dos dimensiones, y para probar la robustez del algoritmo se extendieron a 30 dimensiones. Las funciones f_1, f_5, f_6 y f_7 se utilizan mucho pues están en el conjunto propuesto Jong (1975), aunque en este caso están extendidas a 30 dimensiones. Las otras funciones están agrupadas en la serie de problemas propuesta por Schwefel (1993).

Las funciones f_8 a f_{13}, descritas en la Tabla 4.2, son funciones multimodales y multidimensionales. Dentro de esta serie de funciones se encuentran varias funciones tipo canasta de huevos (f_8 a f_{11}) y dos funciones penalizadas (f_{12} y f_{13}). Estas funciones permiten ver el comportamiento en problemas de diversa índole, puntos de mínima simétricamente distribuidos en la región de búsquedas, en los bordes y la sensibilidad del algoritmo a las penalizaciones.

Las funciones f_{14} a la f_{23} son funciones multimodales de pocas dimensiones y pocos mínimos, que tienen diferente tipos de configuraciones, y están descritas en la Tabla 4.3. Las matrices de datos necesarias se muestran en las Tablas 4.4, 4.5, 4.6, 4.8 y 4.7.

Tabla 4.1: Funciones unimodales

Función de prueba	Dimensiones(D)	Región	mínimo
$f_1(x) = \sum_{i=1}^{D} x_i^2$	30	$[-500, 500]^D$	0
$f_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	$[-10, 10]^D$	0
$f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]^D$	0
$f_4(x) = \text{máx}_i\{\lvert x_i \rvert, 1 \leq i \leq 30\}$	30	$[-100, 100]^D$	0
$f_5(x) = \sum_{i=1}^{D-1} [100(x_{i+1})]x_i^2$	30	$[-100, 100]^D$	0
$f_6(x) = \sum_{i=1}^{D} (\lfloor x_i + 0{,}5 \rfloor)^2$	30	$[-100, 100]^D$	0
$f_7(x) = \sum_{i=1}^{D} i(x_i^4) + random[0, 1)$	30	$[-1{,}28, 1{,}28]^D$	0

Tabla 4.2: Funciones multimodales para muchas dimensiones

Función de prueba	D	Región	mínimo
$f_8(x) = \sum_{i=1}^{D} [x_i \sin(\sqrt{\lvert x_i \rvert})]$	30	$[-500, 500]^D$	-12569,5
$f_9(x) = \sum_{i=1}^{D-1} [x_i^2 - 10\cos(2\pi x_i + 10)]$	30	$[-5{,}12, 5{,}12]^D$	0
$f_{10}(x) = -20 \cdot exp\left(-0{,}2\sqrt{\frac{1}{D} \cdot \sum_{i=1}^{D} x_i^2}\right) - exp\left(\frac{1}{D} \cdot \sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + exp(1)$	30	$[-32, 32]^D$	0
$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-600, 600]^D$	0
$f_{12}(x) = \frac{\pi}{D}\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 \times [1 + \sin^2(3\pi y_{i+1})]$ $+(y_D - 1)^2\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	30	$[-50, 50]^D$	0
$f_{13}(x) = 0{,}1\{10\sin^2(\pi 3x_1) + \sum_{i=1}^{D-1} (y_i - 1)^2[1 + 10\sin^2(\pi y_i + 1)]$ $+(y_D - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ para las funciones f_{12} y f_{13} $u(x_i, a, k, m) = \left\{ \begin{array}{ll} k(x_i - a)^m, & x_i > a \\ 0, & -a \leq x_i \leq a \\ k(-x_i - a)^m, & x_i < -a \end{array} \right\} y_i = 1 + \frac{1}{4}(x_i + 1)$	30	$[-50, 50]^D$	0

4.2. RESULTADOS

El conjunto de funciones es utilizado en (Yao et al., 1996; Vesterstrom y Thomsen, 2004; Brest et al., 2006) para analizar el desempeño de diferentes metaheurísticas. Se presentan los promedios de los mínimos encontrados para las corridas independientes y la desviación estándar. También se da el número de llamadas a la función objetivo (*F.O.*). Las pruebas utilizan poblaciones de 100 individuos, partículas para supernova , y los promedios se da para 50 corridas independientes; se tomaron estos mismos parámetros para las evaluaciones de esta sección.

Tabla 4.3: Funciones multimodales para pocas dimensiones

Función de prueba	D	Región	mínimo
$f_{14}(x) = [\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{D}(x_i - a_{i,j})^6}]$	2	$[-65{,}536, 65{,}536]^D$	≈ 1
$f_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_1^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	$[-5,5]^D$	$\approx 3{,}075 \times 10^{-4}$
$f_{16}(x) = 4x_1^2 - 2{,}1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5,5]^D$	1.0316285
$f_{17}(x) = \left(x_2 - \frac{5{,}1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5,10][0,15]$	0.398
$f_{18}(x) = \left[1 + (x_1+x_2+1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right]$ $\times\left[30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 30x_1x_2 + 27x_2^2\right)\right]$	2	$[-2,2]^D$	3
$f_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right]$	3	$[0,1]^D$	-3.86
$f_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right]$	6	$[-100,100]^D$	-3.32
$f_{21}(x) = \sum_{i=1}^{5}\left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$	4	$[0,10]^D$	-10
$f_{22}(x) = \sum_{i=1}^{7}\left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$	4	$[0,10]^D$	-10
$f_{23}(x) = \sum_{i=1}^{9}\left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$	4	$[0,10]^D$	-10

Tabla 4.4: a_{ij} para la función trincheras f_{14}

$$a_{ij} = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \dots & 32 & 32 & 32 \end{pmatrix}$$

Tabla 4.5: a_{ij} para la función de Kowalik f_{15}

i	a_i	b_i^{-1}
1	0.1957	0.25
2	0.1947	0.5
3	0.1735	1
4	0.1600	2
5	0.0844	4
6	0.0627	6
7	0.0456	8
8	0.0342	10
9	0.0323	12
10	0.0235	14
11	0.0246	16

Tabla 4.6: a_{ij}, c_i y p_{ij} para la función de Hartman f_{19}

i	$a_{ij}, j = 1,2,3$			c_i	$p_{ij}, j = 1,2,3$		
1	3	10	30	1	0.3689	0.1170	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.7470
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.03815	0.5743	0.8828

Tabla 4.7: a_{ij}, c_i y p_{ij} para las funciones de Shekel f_{21},f_{22},yf_{23}

i	a_{ij}, $j = 1,2,3,4$				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

Tabla 4.8: a_{ij}, c_i y p_{ij} para la función de Hartman f_{20}

i	a_{ij}, $j = 1,...,6$						c_i	p_{ij}, $j = 1,...,6$					
1	3	3	17	3.5	1.7	8	1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10	17	1.2	8	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3	3.5	1.7	3	17	8	3	0.2348	0.1415	0.3522	0.2883	0.3047	0.6050
4	17	8	0.05	3.2	0.1	14	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

El desempeño de Supernova se va a comprar con dos métodos: Evolución diferencial en dos de sus versiones y Programación evolutiva más rápida. Las versiones utilizadas son las que mejores datos reportan. Los algoritmos utilizados para la comparación, son métodos que se han desarrollado en el tiempo, tienen muchas aplicaciones prácticas y aportes metodológicos lo que fortalece la robustez de un método.

En la Tabla 4.9 se muestran los resultados para los problemas benchmark propuestos para cuatro métodos de optimización heurísticos poblacionales: Supernova, Evolución Diferencial Autoadaptativa (ADE), Evolución Diferencial Normal (DE), y Programación Evolutiva Más Rápida (FEP). Los datos reportados en las columnas correspondientes a ADE y DE son los publicados por Brest et al. (2006) y los datos reportados en la columna de FEP son los publicados por Yao et al. (1996). Para las veintitrés funciones propuestas se encontró que:

- Para las funciones f_1,f_2, y f_{10} se mejora la calidad de la respuesta en un 42.7 %, 98.9 % y 39 % respectivamente en relación a la mejor respuesta encontrada.

- Para las funciones f_1, f_2, f_3, f_6, f_7, f_9, f_{10}, f_{11} y f_{15} se disminuyen el número de llamadas a la función objetivo, conservando una calidad de respuesta similar o mejor.

- Las funciones f_4,f_{19} y f_{20} son similares con la mejor respuesta reportada en la Tabla con alguna disminución en el número de llamadas a la función objetivo.

- Las funciones f_{14},f_{16}y f_{18} sólo son similares con el resultado de FEP.

Tabla 4.9: Resultados para los diferentes métodos de optimización utilizados

f	Supernova			$F.O$	ADE		DE		FEP	
	$F.O$	$\overline{f*}$	σ		$\overline{f*}$	σ	$\overline{f*}$	σ	$\overline{f*}$	σ
f_1	**972**	$\mathbf{6{,}3 \times 10^{-29}}$	$1{,}0 \times 10^{-28}$	1500	$1{,}1 \times 10^{-28}$	$1{,}0 \times 10^{-28}$	$8{,}2 \times 10^{-14}$	$5{,}9 \times 10^{-14}$	$5{,}7 \times 10^{-4}$	$1{,}3 \times 10^{-4}$
f_2	**1812**	$\mathbf{1{,}1 \times 10^{-26}}$	$1{,}1 \times 10^{-26}$	2000	$1{,}0 \times 10^{-24}$	$9{,}7 \times 10^{-24}$	$1{,}5 \times 10^{-9}$	$9{,}9 \times 10^{-10}$	$8{,}1 \times 10^{-3}$	$7{,}7 \times 10^{-4}$
f_3	**852**	$1{,}0 \times 10^{-13}$	$8{,}8 \times 10^{-14}$	5000	$\mathbf{3{,}1 \times 10^{-14}}$	$5{,}9 \times 10^{-14}$	$6{,}8 \times 10^{-11}$	$7{,}4 \times 10^{-11}$	$1{,}6 \times 10^{-2}$	$1{,}4 \times 10^{-2}$
f_4	**1212**	$6{,}1 \times 10^{-35}$	$1{,}7 \times 10^{-34}$	5000	**0**	0	0	0	0.3	0.5
f_5	510	28.89	$2{,}5 \times 10^{-2}$	20000	**0**	0	0	0	5.06	5.87
f_6	**677**	**0**	0	1500	0	0	0	0	0	0
f_7	**552**	$3{,}2 \times 10^{-3}$	$1{,}9 \times 10^{-3}$	3000	$\mathbf{3{,}15 \times 10^{-3}}$	$7{,}5 \times 10^{-4}$	$4{,}63 \times 10^{-3}$	$1{,}2 \times 10^{-3}$	0.1415	0.3522
f_8	612	$-7{,}4 \times 10^{15}$	$4{,}28 \times 10^{15}$	9000	-12569.5	$7{,}0 \times 10^{-12}$	-11080.1	574.7	-12554.5	52.6
f_9	**612**	**0**	0	5000	0	0	69.2	38.8	$4{,}6 \times 10^{-2}$	$1{,}2 \times 10^{-2}$
f_{10}	**972**	$\mathbf{4{,}7 \times 10^{-15}}$	$2{,}3 \times 10^{-15}$	1500	$7{,}7 \times 10^{-15}$	$1{,}4 \times 10^{-15}$	$9{,}7 \times 10^{-8}$	$4{,}2 \times 10^{-8}$	$1{,}8 \times 10^{-2}$	$2{,}1 \times 10^{-3}$
f_{11}	**612**	**0**	0	2000	0	0	0	0	$1{,}6 \times 10^{-2}$	$2{,}2 \times 10^{-2}$
f_{12}	100	$2{,}9 \times 10^{-2}$	$2{,}5 \times 10^{-2}$	**1500**	$\mathbf{6{,}6 \times 10^{-30}}$	$7{,}9 \times 10^{-30}$	$7{,}9 \times 10^{-15}$	$8{,}0 \times 10^{-15}$	$9{,}2 \times 10^{-6}$	$3{,}6 \times 10^{-6}$
f_{13}	250	1.09	0.55	**1500**	$\mathbf{5{,}0 \times 10^{-29}}$	$3{,}9 \times 10^{-29}$	$5{,}1 \times 10^{-14}$	$4{,}8 \times 10^{-14}$	$1{,}6 \times 10^{-4}$	$7{,}3 \times 10^{-5}$
f_{14}	100	1.39	1.2	**100**	**0.998004**	$2{,}6 \times 10^{-16}$	0.998004	$3{,}3 \times 10^{-16}$	1.22	0.56
f_{15}	**404**	$\mathbf{4{,}4 \times 10^{-4}}$	$4{,}9 \times 10^{-5}$	4000	$4{,}4 \times 10^{-4}$	$2{,}7 \times 10^{-4}$	$4{,}5 \times 10^{-4}$	$3{,}3 \times 10^{-4}$	$5{,}0 \times 10^{-4}$	$3{,}2 \times 10^{-4}$
f_{16}	204	-1.0315	$1{,}36 \times 10^{-4}$	**100**	**-1.03163**	$9{,}7 \times 10^{-12}$	-1.03163	$3{,}1 \times 10^{-13}$	-1.03	$4{,}9 \times 10^{-7}$
f_{17}	204	0.39884	$1{,}2 \times 10^{-3}$	**100**	**0.397887**	$2{,}3 \times 10^{-8}$	0.397887	$9{,}9 \times 10^{-9}$	0.398	$1{,}5 \times 10^{-7}$
f_{18}	204	3.0024	$2{,}9 \times 10^{-3}$	**100**	**3**	$1{,}7 \times 10^{-15}$	3	$2{,}0 \times 10^{-15}$	3.02	0.11
f_{19}	**50**	**-3.862**	$7{,}4 \times 10^{-5}$	100	N. A.	N. A.	N. A.	N. A.	-3.86	$1{,}4 \times 10^{-5}$
f_{20}	**60**	**-3.26**	$3{,}68 \times 10^{-2}$	200	N. A.	N. A.	N. A.	N. A.	-3.27	$5{,}9 \times 10^{-2}$
f_{21}	510	$-8{,}73 \times 10^{-1}$	$2{,}17 \times 10^{-1}$	**100**	**-10.1532**	$2{,}2 \times 10^{-6}$	-10.1532	$2{,}5 \times 10^{-6}$	-5.52	1.59
f_{22}	510	$-8{,}8 \times 10^{-1}$	$2{,}5 \times 10^{-1}$	**100**	**-10.4029**	$4{,}9 \times 10^{-7}$	-10.4029	$3{,}9 \times 10^{-7}$	-5.52	2.12
f_{23}	510	-1.08	$3{,}5 \times 10^{-1}$	**100**	**-10.5364**	$5{,}8 \times 10^{-6}$	-10.5364	$1{,}9 \times 10^{-7}$	-6.57	3.14

- En el caso de las f_{19} y f_{20} no hay resultados con las mismas características para ADE y DE por lo tanto solo se compara con FEP; se observa una disminución al número de llamadas a la función objetivo y un promedio similar.

- Las funciones f_5,f_{12},f_{13},f_{17}, f_{21}, f_{22}, f_{23} no dan resultados satisfactorios con Supernova.

El grupo de funciones escogido presenta diversas características que dan ideas generales sobre el funcionamiento de la heurística. Este grupo de funciones logra encontrar algunas fortalezas de la heurística. Las funciones f_1, f_7, f_9, f_{10} y f_{11} presentan en su forma gráfica un declive que llega al mínimo, para este tipo de funciones, Supernova presenta un mejor desempeño ya sea en calidad, o en disminución del número de llamadas a la función objetivo. En funciones de regiones mas planas como f_5, f_{14}, f_{17}, f_{21}, f_{22} y f_{23} no presenta un buen desempeño.

La función f_8 se propuso originalmente como una función unidimensional y el mínimo local reportado

es en $x^* = 4{,}445 \times 10^{16}$ para un valor $f^* = -4{,}445 \times 10^{16}$, estos valores están por fuera de la región factible tradicional $[-500, 500]^D$ (Schwefel, 1993). En la función extendida a 30 dimensiones, el mínimo global reportado para la región factible es de -12569; Supernova llega a un mejor mínimo fuera de la región factible pues hasta el momento no se ha implementado ningún tipo de manejo de restricciones. En un segundo experimento, se implementó un sencillo sistema de penalización para la función donde se castigaban los y se obtuvo un mejor resultado -11168.

4.3. CONCLUSIONES

Supernova presenta mejora, ya sea en la calidad de respuesta o en disminución del esfuerzo para el 40 % de las funciones aplicadas. En los resultados reportados se observa que Supernova funciona mejor que los métodos comparable en funciones que tienen algún tipo de inclinación para llegar a su mínimo global, y presenta fallas en las funciones que tienen sectores planos.

En el siguiente capítulo se propone la aplicación de la heurística al problema de la estimación de parámetros para redes neuronales.

5. CASO DE APLICACIÓN EN PREDICCIÓN DE SERIES DE TIEMPO

El interés por la predicción de series de tiempo ha aumentado gradualmente a través del tiempo en diversas áreas de la ciencia, finanzas e ingeniería (Palit y Popovic, 2005), lo cual ha motivado el desarrollo de diversas técnicas para su predicción. Entre ellas, se encuentran las redes neuronales artificiales, las cuales tienen ventajas sobre otros tipos de modelos por la facilidad de capturar las relaciones no lineales entre los datos (Zhang et al., 1998). Villa et al. (2008) mostraron que las redes neuronales tipo cascada correlación permiten encontrar modelos con mejor capacidad de predicción que los perceptrones multicapa. Sin embargo, la estimación de los parámetros de las redes cascada correlación es un problema complejo dados los múltiples mínimos y la cantidad de parámetros a estimar.

El objetivo de este capítulo es comparar el desempeño de Supernova en la estimación de parámetros para una red cascada correlación con las formas clásicas de estimación de parámetros como ConRprop.

En la sección 5.1 Se describe el problema de optimización. En la sección 5.2 se describe el manejo de restricciones planteado para desarrollar la aplicación. En la sección 5.3 se describen los casos de aplicación utilizados. Finalmente en la sección 5.4 se concluye el trabajo realizado en el capítulo.

5.1. OPTIMIZACIÓN EN REDES CASCADA CORRELACIÓN

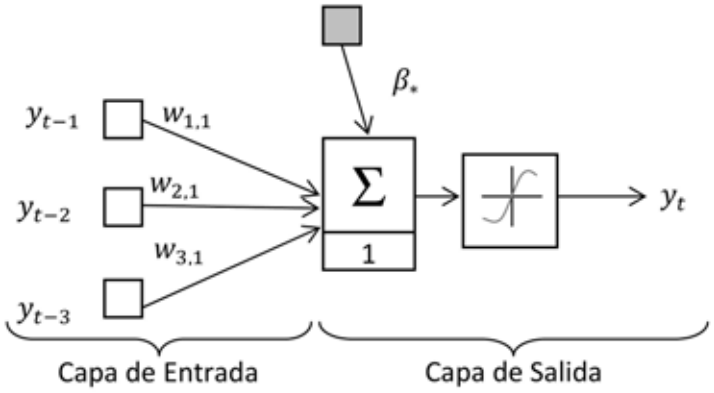
Las redes cascada correlación (CasCor) son un tipo de red neuronal artificial diseñada con aprendizaje constructivo. Iniciando la red desde la manera mas simple, una capa de entrada y una de salida como se muestra en la Figura 5.1 (*a*). Se van agregando neuronas en la capa oculta llegando a un modelo multicapa, como se observa en la Figura 5.1 (*b*); para la siguiente configuración los pesos sinápticos precedentes ($w_{p,h}$, α, etc) son congelados y estiman los nuevos pesos. Se adicionan tantas neuronas como sean necesarias para llegar a resultados satifactorios (Villa et al., 2008).

En términos de optimización, el problema se puede plantear como la minimización de la función de error:

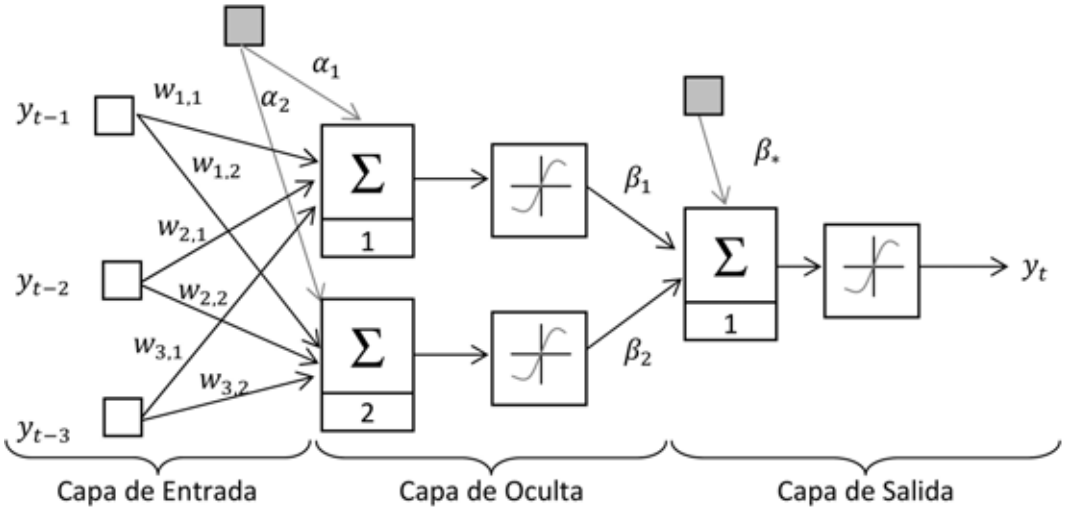
$$SSE = \sum_{t=P+1}^{T} [y_t - \hat{y}_t]^2 \tag{5.1}$$

donde P corresponde a la cantidad de rezagos considerados en el modelo, T corresponde a los periodos de tiempo, y_t a la observación real en el periodo t y $\hat{y}_t$ corresponde al valor de y estimado por la red neuronal en el mismo periodo:

$$\hat{y}_t = f(\Omega, y, \vec{x}) \tag{5.2}$$



(a). Etapa inicial red cascada correlación.



(b). Etapa posterior red cascada correlación

Figura 5.1: Etapas de aprendizaje incremental red cascada correlación

En (5.2), $\Omega = [w_{p,h}, \alpha_h, \beta_h, \beta_*]$ (para $p = 1, \ldots, P$ y $h = 1, \ldots, H$) . P es la cantidad de rezagos y H el número de neuronas en la capa oculta. β_* es un escalar. El vector $\vec{x}$ representa las variables explicativas, que en este caso son: $y_{t-1}, \ldots, y_{t-P}$. Formalmente $\hat{f}$ correspondería a un modelo estadístico no paramétrico de regresión no lineal y se define como:

$$f(\Omega, y, \vec{x}) = \beta_* + \sum_{h=1}^{H} \beta_h \times g\left[\frac{1}{2\sigma_y^{-1}} \times \left(\alpha_h + \sum_{p=1}^{P} w_{p,h} \times y_{t-p}\right) + \varepsilon_t\right] \tag{5.3}$$

donde $g(\cdot)$ es la función de activación de las neuronas de la capa oculta y ε_t se asume como una variable aleatoria con distribución normal con media 0 de varianza desconocida.

Desde el punto de vista de optimización P, H, y y $\vec{x}$ son conocidos y fijos y se busca minimizar la función de error(SSE) cuyos parámetros son las componentes de Ω. Dentro de los parámetros a estimar es importante notar que para el caso de la red cascada correlación se tienen algunas restricciones $-1 \leq w_{p,h} \leq 1$ y $-1 \leq \alpha_h \leq 1$.

El problema de la estimación de parámetros en una red neuronal tiene varias dificultades. En primer lugar, las superficies de error son intrincadas y altamente no lineales (Fine, 1999); además, en este caso particular, el número de variables a estimar crece haciendo que el problema tenga cada vez más dimensiones (Yao et al., 1996). Adicionalmente, hay un mayor grado de complejidad por las restricciones dadas (Michalewicz y Schmidt, 2002).

5.2. MANEJO DE RESTRICCIONES

Una de las formas mas populares de manejo de restricciones es la penalización. La penalización consiste en aplicar un castigo (en el caso de la minimización, aumentar el valor de la función objetivo) a las soluciones que no cumplen con las restricciones dadas, y así convertir un problema con restricciones en uno sin restricciones. De las clases de penalización existentes (estáticas, dinámicas, adaptativas, entre otras) se utilizó una penalización estática propuesta por Hoffmeister y Sprave para restricciones de igualdad o desigualdad (Coello Coello, 2002; Villa et al., 2009).

En la Figura 5.2 se observa el efecto de la penalización propuesta sobre la función; los trazos oscuros marcan los valores que tomaría la función para diferentes valores de x; como se puede observar en la región factible, estipulada entre LB y UB, se toma la función objetivo, mientras que en la región no factible se toma el valor del castigo propuesto. La función objetivo transformada $F(x)$ es la función por partes:

$$F(x) = \begin{cases} f(x) & x \in \Omega \\ M_c + d_c & x \notin \Omega \end{cases} \tag{5.4}$$

donde M_c es el valor máximo encontrado para $f(x)$ dentro de la región factible y d_c es la medida de penali-

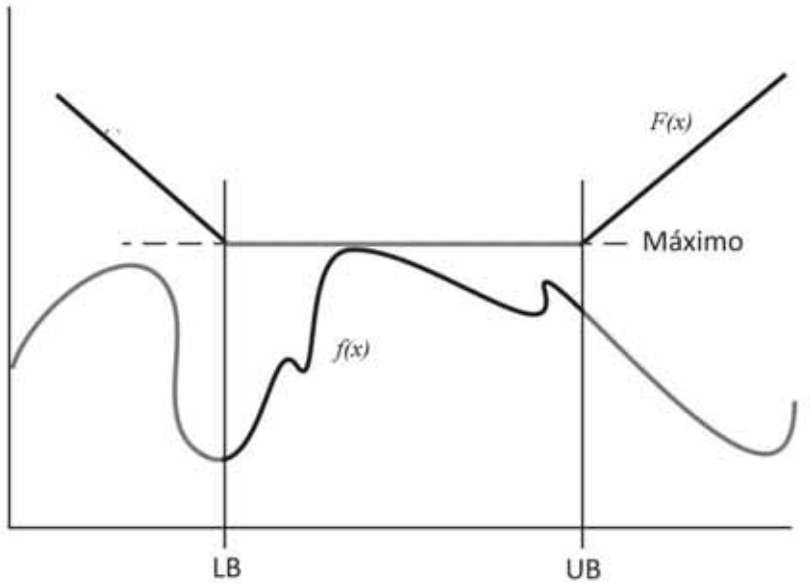


Figura 5.2: Penalización de la función

zación

$$d_c = \max\{H_1 \times |R_1|, \ldots, H_m \times |R_m|\} \tag{5.5}$$

donde m es el número de restricciones. H es una matriz de coeficientes del orden ($m \times$# de dimensiones) que toma valores de 0 cuando la restricción se cumple y 1 cuando no se cumple. Y R es el valor de restricción en ese punto. Así pues, para cualquier matriz $H \neq 0$ la función tomará el valor de $M_c + d_c$. Una vez implementado el sistema de penalización el problema planteado inicialmente con restricciones es un problema no lineal sin restricciones que puede ser resuelto con Supernova.

5.3. CASOS DE APLICACIÓN

Las diferentes versiones de redes neuronales se han optimizado clásicamente con el algoritmo de resilient back-propagation (rprop); sin embargo, para el problema de las redes cascada correlación se aplicó una versión penalizada de este mismo algoritmo llamada ConRprop (Villa et al., 2009). A continuación se presentan dos casos de estudio y los resultados encontrados en la literatura para estos casos. Los detalles del algoritmo CONRprop pueden ser consultados en (Villa et al., 2009).

5.3.1. Parametrización del algoritmo

Inicialmente se tomó un modelo de red cascada correlación y se varió los parámetros hasta el funcionamiento adecuado del algoritmo. En la Tabla 5.1 se observan los parámetros utilizados por Supernova para optimizar los casos de aplicación, el método de reinicio es tipo I.

Tabla 5.1: Parámetros utilizados para Supernova los casos de aplicación

Caso	k	F	G	iter	reinicios	λ	α
Pasajeros de una aerolínea	60	1	1	50	35	3	1.1
Linces canadienses	60	1	1	50	35	3	1.1

Tabla 5.2: Valores del SSE para diferentes modelos pronosticando la serie del primer caso

Modelo	Rezagos	H	SSE Entrenamiento			SSE Validación		
			ANN	Supernova	ConRprop	ANN	Supernova	ConRprop
1	1 - 13	2	0.73	**0.14**	0.28	0.71	**0.01**	0.09
2	1, 12	2	2.30	**0.32**	0.35	0.34	**0.02**	0.03
3	1, 2, 12	2	2.17	**0.31**	0.39	0.29	**0.02**	0.03
4	1, 2, 12, 13	2	0.99	**0.18**	0.65	0.52	**0.01**	0.05
5	1, 12, 13	4	0.84	**0.16**	0.55	0.62	**0.01**	0.03
6	1, 2, 3, 4	4	7.74	**1.10**	1.15	1.03	**0.12**	0.40

5.3.2. Caso de estudio: Pasajeros de una Aerolínea

Esta serie de tiempo contiene el registro del número total de pasajeros transportados por mes por una aerolínea, desde enero de 1949 hasta diciembre de 1960. Cada uno de los modelos presentados en la Tabla 5.2 fue estudiada por Faraway y Chatfield (1998) y Ghiassi et al. (2005). Para cada modelo, los datos de la serie se transformaron utilizando la función logaritmo natural (base - e); se usaron los primeros 120 datos para entrenamiento y los 12 últimos para validación, tal como fue realizado por Faraway y Chatfield (1998). Los parámetros utilizados para supernova se muestran en la Tabla 5.1.

Faraway y Chatfield (1998) reportaron para cada uno de los modelos de la Tabla 5.2, la sumatoria del error cuadrático (SSE) para las muestras de entrenamiento y validación, al predecir la serie con redes neuronales artificiales (ANN). Mientras que en la Tabla 5.3, se resumen los resultados reportados por Ghiassi et al. (2005). Para cada uno de los modelos de la Tabla 5.2, se calcula el estadístico de ajuste para las muestras en entrenamiento y de validación, al pronosticar la serie con redes CASCOR optimizadas con Supernova y ConRprop. Se puede apreciar que todos los errores obtenidos CASCOR optimizadas con Supernova, tanto en entrenamiento como en validación, son menores que los obtenidos con ConRprop, que los reportados por Faraway y Chatfield (1998) (Tabla 5.2) y por Ghiassi et al. (2005) (Tabla 5.3).

Para esta caso de estudio, los resultados del SSE se comprara con ANN y con ConRprop, los resultados se muestras en la Tabla 5.2. Se observa una reducción consistente para todos los modelos. Los mejores resultados para la fase de entrenamiento corresponden a loas modelos 1,4 y 5, consistentemente para los modelos 1 y 4 se nota una también una notable mejoría para la validación; pero, es el modelo 6 donde se

Tabla 5.3: Porcentajes de mejora de Redes CASCOR estimadas con Supernova sobre DAN2.

Modelo	Rezagos	SSE Entrenamiento			SSE Validación		
		DAN2	Supernova	% Mejora	DAN2	Supernova	% Mejora
1	1 - 13	0.17	**0.14**	17.65	0.23	**0.01**	95,65
2	1, 12	0.85	**0.32**	62.35	0.26	**0.02**	92,31
3	1, 2, 12	0.44	**0.31**	29.55	0.19	**0.02**	89,47
4	1, 2, 12, 13	0.30	**0.18**	40.00	0.29	**0.01**	96,55
6	1, 12, 13	0.24	**0.16**	33.33	0.22	**0.01**	95,45

presenta la mayor reducción del SSE para la validación.

En la Tabla 5.3, se presenta el porcentaje de mejoramiento al comparar supernova con DANN2 donde se observa que al igual que en la comparación anterior la mejora es consistente para todos los modelos. El modelo 4 presenta las mayores reducciones del error tanto en la fase de entrenamiento como en la validación. En este caso supernova presenta unos errores de validación mucho menores con un rango de % de mejora entre 89.47 % y 96.55 %.

En la Figura 5.3 se presentan los valores reales de la serie de tiempo y los pronosticados con el modelo 1 de CASCOR estimado con supernova y RPROP de la Tabla 5.2, en ésta gráfica se observa que la red CASCOR estimada con supernova se ajusta un poco mejor a la serie que la misma red estimada con ConRprop.

5.4. CASO DE APLICACIÓN: LINCES CANADIENSES

En esta serie se encuentra registrada la cantidad de linces capturados anualmente, desde 1821 hasta 1934, en los alrededores del río Mackenzie ubicado en el distrito de Northem, Canadá. Esta serie fue estudiada por (Cambell y Walker, 1977; Rao y Gabr, 1984; Zhang, 2003) . Los datos de la serie se transformaron utilizando la función logaritmo base 10; de sus 114 datos se tomaron los 100 primeros para entrenamiento y los últimos 14 para validación tal como se ha realizado en estudios pasados. Los parámetros utilizados para supernova se muestran en la Tabla 5.1.

En la Figura 5.4 se presentan los valores reales de la serie de tiempo y los pronosticados con el modelo 1 de CASCOR estimado con supernova y RPROP de la Tabla 5.4, en ésta gráfica se observa que la red CASCOR estimada con supernova se ajusta mejor a la serie que la misma red estimada con ConRprop. En la Tabla 5.4 presenta los datos obtenidos para los diferentes modelos de la serie de linces canadienses, en este caso, supernova no es superior en todos los modelos, pero presenta mejores resultados en la sección de entrenamiento para los modelos 1-4 y 6. La mejora mas sustancial la presenta para el modelo 4. En la parte del error de validación, presenta mejores comportamientos para los modelos 1,3,4 y 6 presentando la mayor disminución del error para el modelo 4.

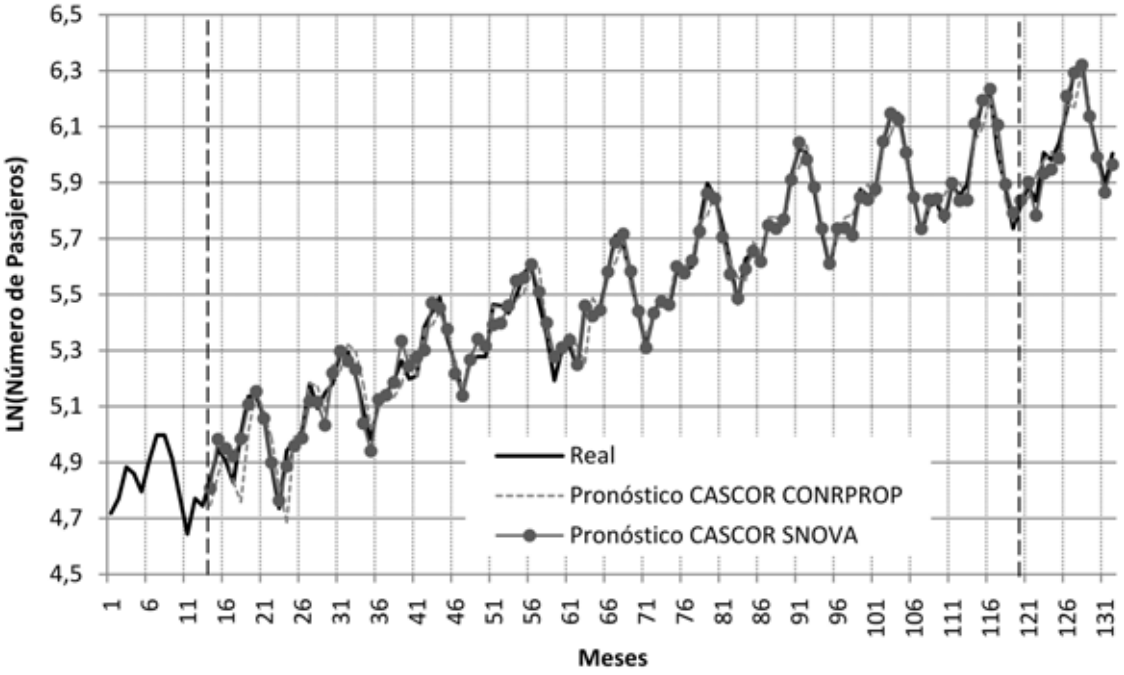


Figura 5.3: Predicción con una red CASCOR optimizada con Supernova y ConRprop para la serie de pasajeros de una aerolínea.

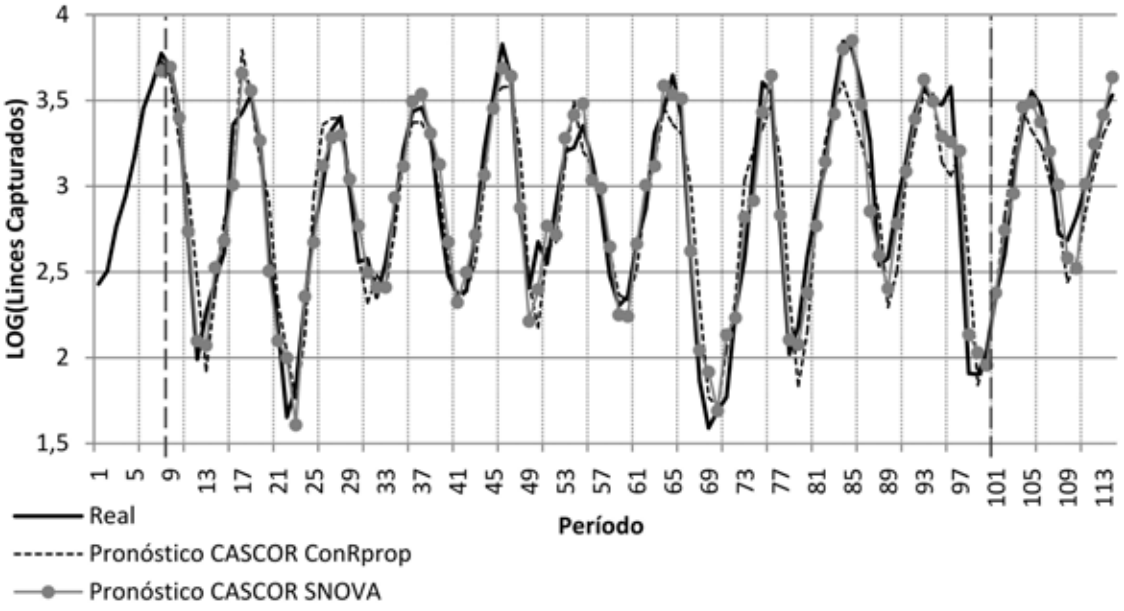


Figura 5.4: Predicción con una red CASCOR para la serie de tiempo de Linces Canadienses

Tabla 5.4: Valores del error cuadrático medio para diferentes modelos pronosticando la serie de los linces canadiences.

Modelo	Rezagos	SSE Entrenamiento		SSE Validación	
		Supernova	ConRprop	Supernova	ConRprop
1	1 - 7	**0.0371**	0.0503	**0.0135**	0.0210
2	1,2,3,8,9,10	**0.0403**	0.0492	0.0451	**0.0224**
3	1 - 9	**0.0122**	0.0124	**0.0551**	0.0553
4	1 - 6	**0.0383**	0.0536	**0.0096**	0.0270
5	1,2,3,4,8,9,10	0.0675	**0.0468**	0.0316	**0.0215**
6	1 - 8	**0.0362**	0.0483	**0.0175**	0.0242

5.5. CONCLUSIONES

En este capítulo se presentó la aplicación de Supernova al problema de estimación de parámetros en una red cascada correlación. Inicialmente se implementa con éxito el manejo de restricciones, ampliando la gama de los problemas a los que se puede aplicar Supernova.

Comparándola con ConRprop, Supernova muestra un mejor ajuste para la mayoría de modelos propuesto para los casos de aplicación. Además de lograr un mejor ajuste, Supernova tiene una ventaja adicional, los parámetros del algoritmo no se variaron para los diferentes modelos, lo que indica que un mismo conjunto de parámetros para el algoritmo proveen una buena respuesta en dos casos diferentes.

Con la aplicación al problema de la estimación de parámetros de las redes neuronales, concluye el reporte del trabajo de investigación propuesto. En el próximo capítulo se presentan las conclusiones generales de la investigación.

6. CONCLUSIONES

En el capítulo uno se presenta la inquietud básica sobre la que se desarrollada este trabajo ¿ Cómo plantear una analogía computacional basada en el fenómeno de la supernova que permita revolver problemas de optimización? En el desarrollo de la investigación se observa que aún, teniendo una idea inicial clara es difícil llegar a una metaheurística robusta y totalmente aplicable.

En el fenómeno de la supernova se observó el cambio desde su inicio caótico, producido por una explosión, hasta llegar a un equilibrio dinámico; es decir, mejora el orden. La condición de mejoramiento en un proceso es necesaria para realizar una analogía entre un proceso físico y la optimización. La emulación de un proceso del entorno provee en términos de optimización, estrategias valiosas para resolver problemas complejos.

Un proceso físico, como las supernovas, puede proveer diferentes reglas que, combinadas, recrean el fenómeno en sí. En este caso se enfocó inicialmente en la interacción de las partículas, en como cambia su trayectoria de acuerdo, a la atracción gravitatoria y el impulso inicial de la explosión, entre otros factores. La combinación de estas fuerzas provee una excelente exploración de una región, pero en términos de calidad de respuesta se queda corta. Entonces se plantearon los reinicios como una estrategia de explotación, que intensifica la búsqueda en la región de interés provista por la estrategia inicial. La combinación de estas dos estrategias logra respuestas de mejor calidad.

Las metaheurísticas como métodos tipo caja negra y planteados empíricamente no tienen una prueba matemática estricta, por lo tanto, se hacen pruebas empíricas que muestran el comportamiento y la calidad de las respuestas obtenidas. Supernova se evaluó en la fase de implementación de primer prototipo. Inicialmente se observó el proceso de búsqueda que estuvo acorde a lo planteado en el diseño de la heurística, mostrando que el algoritmo planteado cumplía con la analogía del proceso propuesta. Seguidamente se probó su funcionamiento sobre funciones sencillas obteniendo un comportamiento asintótico para las funciones de prueba, comportamiento que denota convergencia en para cualquier método numérico iterativo. También se observó que este comportamiento es consistente para varias funciones, varias dimensiones, diferentes disposiciones del mínimo, mínimos negativos y soluciones y mínimos no enteros.

Las heurísticas plantean caminos de búsqueda. Para cada problema, este camino debe adaptarse a las necesidades particulares de la región, para esto se utilizan los parámetros. Supernova tiene siete parámetros dos controlan el proceso de interacción de las partículas (F y G). Otros dos controlan la fuerza de los reinicios (λ y α). La extensión de la búsqueda está dado por los parámetros de iteraciones y reinicios máximos (T y RR). Y finalmente, el número de partículas (k). Para cada parámetro se evaluó un rango que mostró potencial

en las pruebas iniciales y se determinaron los valores para que la heurística funcionara de la mejor forma para diferentes funciones. Se observó que los parámetros que exponen mas sensibilidad al cambio son λ, α, T y RR. Se recomienda el uso de los siguientes valores de parámetros para una función desconocida:

$$Parámetros \begin{cases} F = 1 \\ G = 1 \\ k = [60, 100] \\ T = 100 \\ RR = 100 \\ \lambda = 200 \ y \ \alpha = 1{,}5 \end{cases} \tag{6.1}$$

λ y α, varian de acuerdo a la región de búsqueda.

Tener evidencias de la convergencia y un buen análisis del comportamiento de una heurística no basta para demostrar que es una heurística competitiva en términos de calidad y esfuerzo. Supernova se comparó con Evolución diferencia autoadaptativa, evolución diferencial y Programación evolutiva mas rápida, todos métodos maduros con aportes metodológicos que mejoran ampliamente su comportamiento inicial. Los métodos mencionados son métodos poblacionales que pueden ser comparados para un mismo número de partículas o individuos.

Supernova supera los datos reportados para los métodos con los que se compara en el 40 % de las funciones escogidas, ya sea con una mejor calidad de respuesta o en una disminución del esfuerzo. El comportamiento es similar en un 30 % de las funciones en términos de calidad de respuestas. Presenta su mayor fortaleza es la rápida y extensa exploración de la región de búsqueda.

La etapa de validación no solo provee datos absolutos del comportamiento, también se logra prefilar el método, mostrando fortalezas y debilidades. Supernova muestra su mejor comportamiento en funciones donde la región del mínimo global presenta un declive hacia el mínimo.

Las metaheurísticas tiene un gran potencial para problemas del mundo real. Supernova se aplicó a la estimación de parámetros para la red cascada correlación, un tipo de red neuronal utilizado para la predicción de series de tiempo. La estimación de parámetros para una red neuronal es un problema multimodal de muchas variables, en la que la mayoría de los métodos tiene problemas porque la solución encontrada generalmente es un mínimo local y depende del punto de inicio, en este caso estas redes tienen también restricciones en sus parámetro; Por lo tanto, este problema es un reto para los métodos de optimización. Se compara el desempeño de Supernova contra ConRprop, versión de Rprop con adecuación al manejo de restricciones. Supernova logró un mejor ajuste tanto en la fase de entrenamiento como en la etapa de validación en el 87,5 % de modelos planteados en los casos de aplicación.

Las restricciones aumentan la dificultad del problema de optimización , ya que, hace que el área factible sea mas compleja. Una forma sencilla de manejo de restricciones es la penalización que convierte un pro-

blema con restricciones en un con restricciones permitiendo que se le apliquen métodos para problemas sin restricciones. Para poder aplicar a Supernova en el problema de aplicación se implementó el mismo manejo de restricciones por penalización descrito por ConRprop, Con esta implementación se pudo aplicar Supernova al problema de las redes cascada correlación.

En definitiva, el trabajo concluye que es posible plantear una metaheurística a partir del fenómeno de las supernovas, que es una heurística competitiva en términos de calidad de respuesta y esfuerzo computacional; y aplicable a problemas de alta complejidad como la estimación de parámetros en las redes neuronales.

A partir de la respuesta dada al interrogante propuesto surgen varias preguntas, que quedarán abiertas para futuros trabajos de investigación.

- ¿ Es posible mejorar la eficiencia del algoritmo en términos de tiempo?
- ¿Los parámetros de la heurística Supernova podrían autoadaptarse?
- ¿Las estrategias de búsqueda de Supernova pueden reducirse a procesos mas pequeños?
- ¿Puede mejorarse el funcionamiento y robustez de la heurística hibridizándola?
- ¿Se puede mejorar el funcionamiento aplicando otras estrategias como criterios de parada dinámicos o distribuciones aleatorias diferentes a la normal?

BIBLIOGRAFÍA

Alonso, M. y Finn, E. (1995), *Física*, Addison-Wesley.

Bäck, T. y Schwefel, H.-P. (1993), 'An overview of evolutionary algorithms for parameter optimization', *Evolutionary Computation* **1**(1), 1–23.

Bäck, T. (1996), *Evolutionary Algorithms in Theory and Practice*, Oxford.

Bäck, T., Fogel, D. y Michalewicz, Z. (1997), *Evolutionary Computation 1*, IOP Press.

Beyer, H. y Schwefel, H. (2002), 'Evolution strategies', *Natural computing* **1**(1), 3–52.

Brest, J., Greiner, S., Bošković, B., Mernik, M. y Zumer, V. (2006), 'Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems', *IEEE Transactions on Evolutionary Computation* **10**(6), 897–913.

Cambell, M. J. y Walker, A. M. (1977), 'A survey of statistical work on the mackenzie river series of annual canadian lynx trappings for the years 1821-1934 and a new analysis', *Journal of the Royal Statistical Society* **140**(4), 411–431.

Carroll, B. y Ostlie, D. (2007), *An Introduction to Modern Astrophysic*, Pearson.

Coello, C. (2005), An introduction to evolutionary algorithms and their applications., *in* 'Proceeding of Internacional Symposium and school of advance distributed systems Vol. 3563', Springer, Guadalajara, pp. 425–442.

Coello Coello, C. A. (2002), 'Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art', *Computer Methods in Applied Mechanics and Engineering* **191**(11-12), 1245–1287.

Corne, D., Dorigo, M. y Glover, F. (1999), *New Ideas in Optimization*, McGraw Hill.

Crespo, E., Martí, R. y Pacheco, J. (2007), *Procedimientos Metaheurísticos En Economía Y Empresa*, Tirant Lo Blanch.

Das, A. y Chakrabarti, B. (2005), *Quantum Annealing and Other Optimization Methods*, Vol. 679 of *Lecture Notes in Physics*, Springer-Berlin, Berlin.

de Castro, L. N. (2007), 'Fundamentals of natural computing: an overview', *Physics of Life Reviews* **4**(1), 1–36.

de Castro, L. N. y Timmis, J. I. (2003), 'Artificial immune systems as a novel soft computing paradigm', *Soft Computing* (7), 526–544.

Dorigo, M. y Blum, C. (2005), 'Ant colony optimization theory: A survey', *Theoretical Computer Science* **344**(2-3), 243–278.

Dorigo, M. y Stützle, T. (2003), *Handbook of Metaheuristics*, Springer, chapter The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances.

Eiben, A. y Smith, J. E. (2003), *Introduction to Evolutionary Computing*, Springer, Verlag.

Erol, O. K. y Eksin, I. (2006), 'A new optimization method: Big bang-big crunch', *Advances in Engineering Software* **37**(2), 106–111.

Faraway, J. y Chatfield, C. (1998), 'Time series forecasting with neural networks: A comparative study using the airline data', *Applied Statistics* **47**(2), 231–250.

Feoktistov, V. (2006), *Differential Evolution*, Springer.

Fine, T. L. (1999), *Feedforward neural network methodology*, Springer.

Fogel, D. (1994), 'Asymptotic convergence properties of genetic algorithms and evolutionary programming: analysis and experiments', *Cybernet Systems* **25**, 389–407.

Ghiassi, M., Saidane, H. y Zimbra, D. (2005), 'A dynamic artificial neural network model for forecasting time series events', *International Journal of Forecasting* **21**(2), 341–362.

Glover, J. F., Kelly, J. y Laguna, M. (1995), 'Genetic algorithms and tabu search: hybrids for optimization', *computer Operation Research* **22**(1), 111–134.

Goldberg, D. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.

Granville, V., Krivánek, M. y Rasson, J. (1994), 'Simulated annealing: A proof of convergence', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(6), 652–656.

Grupen, C. (2005), *Astrophysic Particles*, Springer.

Himmelblau, D. (1972), *Applied nonlinear optimization*, McGraw Hill.

Hirsch, M., Meneses, C., Pardalos, P. y Resende, M. (2006), 'Global optimization by continuous grasp', *Optimization letters* **1**(2), 201–212.

Jang, J., Sun, C. y Mizutan, E. (1996), *Neuro-fuzzy and soft computing*, Prentice Hall.

Jong, K. D. (1975), Analysis of the behavior of a class of genetic adaptative systems, PhD thesis, The University of Michigan.

Kennedy, J. y Eberhart, R. (1995), Particle swarm optimization, *in* 'IEEE International Conference on Neural Networks, 1995', Vol. 11, pp. 1942–1948.

Kirkpatrick, S., Gelatt, C. D. y Vecchi, M. P. (1983), 'Optimization by simulated annealing', *Science, New Series* **220**(4598), 671–680.

Lieddle, A. (2003), *Introduction to Modern Cosmology*, Wiley.

Martí, R. y Laguna, M. (2003), 'Scatter search: Basic design and advanced strategies', *Revista Iberoamericana de Inteligencia Artificial* (19), 123–130.

Michalewicz, Z. y Schmidt, M. (2002), *Evolutionary Optimization*, Kluwer Academic Press, chapter Evolutionary Algorithms and constrained optimization.

Moscato, P. y Cotta, C. (2003), 'A introduction to memetic algoritms', *Revista Iberoamericana de Inteligencia Artificial* **19**, 131–148.

Muller, S., Marchetto, J., Airaghi, S. y Kournoutsakos, P. (2000), Optimization algorithms based on a model of bacterial chemotaxis, *in* 'proceeding of 6th Int. Conf. Simulation of Adaptive Behavior: From Animals to Animats, SAB 2000', Springer-Verlay, Paris, pp. 375–384.

Musilek, P., Lau, A., Reformat, M. y Wyard-scott, L. (2006), 'Immune programming', *Information Sciences* **176**, 972–1002.

Nash, S. y Sofer, A. (1996), *Linear and Nonlinear Programming*, McGraw hill.

Nguyen, X. H., Nguyen, Q. U. y McKay, R. I. (2007), Initialising PSO with randomized low-discrepancy sequences, *in* 'GECCO', pp. 173–181.

Palit, A. y Popovic, D. (2005), *Computational Intelligence in Time Series Forecasting*, Springer.

Pierre, D. A. (1986), *Optimizatión theory with applications*, Dover.

Porto, W. (2000), *Evolutionary Computation*, Institute of physics publising, chapter Evolutionary programming.

Rao, S. (1996), *Engineering Optimization, Theory and practice*, Wiley.

Rao, T. S. y Gabr, M. M. (1984), *An introduction to bispectral analysis and bilinear time series models*, number 24 *in* 'Lecture notes in statistics', Springer, New York, NY.

Resende, M. y Ribeiro, C. (2002), Greedy randomized adaptive search procedures, Technical report TD-53RSJY, AT&T Labs.

Santos, L. D. y Rodrigues, A. (1999), 'Algoritmos evolutivos em identificação e controle de processos: uma visão integrada e perspectivas', *SBA Controle & Automação* **10**(1), 13–30.

Schwefel, H. (1993), *Evolution and Optimun Seeking*, John Wiley and sons Inc.

Shang, Y.-W. y Qiu, Y.-H. (2006), 'A note on the extended rosenbrock function', *Evolutionary Computation* **14**(1), 119–126.

Talbi, E. G. (2009), *Metaheuristics: from design to implementation*, Wiley.

Torczon, V. (1997), 'On the convergence of pattern search algorithms', *SIAM Journal on Optimization* **7**(1), 1–25.

Törn, A. y Zilinskas, A. (1989), *Global optimization*, Vol. 350 of *Lecture Notes in Computer Science*, Springer-Verlag.

Vesterstrom, J. y Thomsen, R. (2004), A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, *in* 'Proceedings of congress of evolutionary computacion. Vol 2.', IEEE Press, NY, Portland, pp. 1980–1987.

Villa, F. A., Velásquez, J. D. y Souza, R. C. (2008), 'Una aproximación a la regularización de redes cascada-correlación para la predicción de series de tiempo', *Investigação Operacional* **28**, 151–161.

Villa, F., Velásquez, J. y Jaramillo, P. (2009), 'Conrprop: un algoritmo para la optimización de funciones no lineales con restricciones', *Revista Facultad de Ingeniería Universidad de Antioquia* (50), 179–185.

Voß, S. (2001), Meta-heuristics: The state of the art, *in* 'ECAI '00: Proceedings of the Workshop on Local Search for Planning and Scheduling-Revised Papers', Springer-Verlag, London, UK, pp. 1–23.

Winker, P. (2001), *Optimization Heuristic in Econometrics*, Wiley.

Wolpert, D. H. y Macready, W. G. (1997), 'No free lunch theorems for optimization', *IEEE Transactions on evolutionary computation* **1**(1), 67–82.

Wu, Q. y Cao, Y. (1997), Stochastic optimization of control parameters in genetic algorithms, *in* 'Proceedings of congress IEEE international conference on evolutionary computation', IEEE, Indianapolis, pp. 77–80.

Yao, X., Liu, Y. y Lin, G. (1996), 'Evolutionary programming made faster', *IEEE Transactions on Evolutionary Computation* **3**, 82–102.

Yiqing, L., Xigang, Y. y Yongjian, L. (2007), 'An improved pso algorithm for solving non-convex nlp/minlp problems with equality constraints', *Computer and Chemical Engineering* **31**, 153–162.

Zhang, G., Eddy Patuwo, B. y Y. Hu, M. (1998), 'Forecasting with artificial neural networks: The state of the art', *International Journal of Forecasting* **14**(1), 35–62.

Zhang, G. P. (2003), 'Time series forecasting using a hybrid arima and neural network model', *Neurocomputing* **50**, 159–175.

Parte I

Anexos

A. ALGORITMOS BIO-INSPIRADOS

En este apéndice se hace una breve descripción de las técnicas observadas en la Figura 1.4

A.1. COMPUTACIÓN EVOLUTIVA (EC, DEL INGLÉS EVOLUTIONARY COMPUTATION)

Es una rama de los métodos metaheurísticos que toma su inspiración biológica en los estudios de Leclerc, Darwin y Lamark que describen la forma en la que las diferentes especies se adaptan y evolucionan. De los métodos que subyacen de estos trabajos clásicamente se encuentra en tres paradigmas: Programación Evolutiva, Algoritmos Genéticos y Estrategias Evolutivas. Dentro de la computación evolutiva también se puede enmarcar la técnica de Evolución Diferencial (Bäck et al., 1997; Vesterstrom y Thomsen, 2004).

Utilizando operadores evolutivos como: el cruce y la mutación los algoritmos evolutivos logran converger a mejores soluciones, estos paradigmas difieren entre sí en la forma de aplicación e implementación, pero básicamente siguen la misma secuencia que se muestra en el Pseudocódigo 2.

Pseudocódigo 2 Pseudocódigo computación evolutiva

$t \Leftarrow 0$
Generar población inicial $G(t)$
repetir
  Evaluar $\Phi(G(t))$
  Seleccionar $G'(t) \subseteq G(t)$
  Aplicar operadores evolutivos a $G'(t)$
  $G(t+1) \Leftarrow G'(t)$
  $t \Leftarrow t+1$
hasta criterio de parada

Cada paradigma tiene diferentes operadores y parámetros de control que son sumamente importantes para el correcto funcionamiento de los algoritmos inspirados en la evolución, muchas de las mejoras asociadas a estas técnicas se refieren a la autoadaptación de parámetros (Wu y Cao, 1997; Beyer y Schwefel, 2002).

A.1.1. Algoritmos Genéticos: (GA del inglés Genetic Algorithms):

Propuestos inicialmente por John Holland en la década de los 60's, tienen su inspiración en el proceso de selección natural en el mejoramiento de una población basado en la reproducción sexual y la mutación. En la técnica como tal hay un pre-proceso a la secuencia básica mostrada en el Pseudocódigo 2, pues el universo de soluciones debe traducirse a cadenas binarias llamadas cromosomas. Una población $G(t)$ de n cromosomas es generada aleatoriamente, se evalúa de acuerdo a la función de ajuste Φ, se selecciona los mejores individuos para perpetuar su herencia por medio del cruce mientras que los peores individuos mueren o son mutados de acuerdo a los parámetros del algoritmo. Este proceso se repite hasta que los valores de ajuste son prácticamente iguales o se complete un número de generaciones(iteraciones) esperado (Coello, 2005).

Los parámetros del algoritmo que varía son: criterios de parada, el tamaño de población y los argumentos de los operadores de cruce y mutación (Coello, 2005).

Los GA han sido aplicados en varias áreas como: optimización de problemas combinatorios y numéricos, aprendizaje de máquina, bases de datos, predicción, reconocimiento de patrones, controladores, entre otros (Goldberg, 1989).

A.1.2. Programación Evolutiva (EP del inglés Evolutionary Programming):

Propuesto por Fogel en los 60's. Recibe su inspiración del comportamiento inteligente que permite predecir un entorno y responder a él de acuerdo a un objetivo trazado (Pierre, 1986). Fogel lo presentó como la evolución de un autómata por medio de los operadores de mutación, recombinación y selección (Vesterstrom y Thomsen, 2004). En la EP no se utiliza la traducción binaria de las respuestas como en los GA.

Al igual que los otros paradigmas de la EP sigue el Pseudocódigo 2 sólo que en este caso, la mutación es el operador utilizado y la población $G'(t)$ se selecciona no solo de la población anterior sino también de un conjunto de soluciones posibles Q diferente a la población de padres seleccionados. Los parámetros de control son: el tamaño de población, criterio de parada, argumentos de mutación, tamaño de la población Q y los argumentos de generación de esta población(Bäck, 1996).

Dentro de las aplicaciones encontradas en la literatura están: la optimización de redes neuronales, automatización y control, optimización de comportamiento de entidades autónomas entre otros (Porto, 2000).

A.1.3. Estrategias Evolutivas (ES del inglés Evolution Strategies):

Propuesto por Rechember y Schewefel en la década de los 60's. Combina el proceso de selección natural con una incipiente adaptación, similar al planteado por Lamark. Trabajan con vectores reales (Bäck et al., 1997).

Inicialmente el tamaño de la población de padres se denota con μ y λ la población de descendientes dado en el nombre del algoritmo, la versión inicial en los 60's tenía 1 padre y 1 descendiente. La idea fue evolucionando hasta llegar a $(\mu/\rho + \lambda)$-ES, donde indica indica la recombinación en caso de ser 1 se habla de clonación, y ρ indica las generaciones antes de volver a evaluar. Otros parámetros son los argumentos de selección mutación y recombinación; estos parámetros fueron los primeros en ser dinámicos permitiendo un mejor funcionamiento del algoritmo para los años 90's (Beyer y Schwefel, 2002).

El algoritmo sigue los lineamientos de la secuencia observada en el Pseudocódigo 2, los procesos de selección y recombinación no siempre tienen el mismo número de población dado que no siempre los descendientes son el mismo número que los padres. La recombinación es un proceso más elaborado que en los GA ya que combina cruce y mutación (Eiben y Smith, 2003).

Dentro de las aplicaciones descritas en la literatura se encuentras problemas combinatorios, optimización de rutas, entre otros (Santos y Rodrigues, 1999). Junto a los GA es uno de los algoritmos que tiene mas mejoras e hibridizaciones en la literatura.

A.1.4. Evolución Diferencial (DE del inglés Differencial evolution):

Pertenecen a la familia de la computación evolutiva y es de las técnicas más nuevas. Se presentó en 1995 por Storn and Prince y sigue la secuencia normal de un algoritmo evolutivo como el Pseudocódigo 2, y difiere respecto a sus predecesores en la forma de conseguir la herencia, pues en los operadores normales de recombinación o mutación, presenta una vectorización de los elementos escogidos aleatoriamente (Vesterstrom y Thomsen, 2004). Es un método diseñado específicamente para optimización global.

Los parámetros utilizados son: el tamaño de población, argumentos de mutación y recombinación y criterio de parada. Una de sus características apetecibles son los pocos parámetros que utiliza en relación a los otros del paradigma. A pesar de su reciente creación respecto a los otros algoritmos presentes en el paradigma evolutivo, tiene muchas aplicaciones en problemas del mundo real por su extensión a problemas con restricciones, manejo de variables de tipo real, discreto y mixto, trabajo que fortalece el método en sí (Feoktistov, 2006).

A.2. ALGORITMOS MEMÉTICOS (MA DEL INGLÉS MEMETIC ALGORITHM):

En la literatura aparece a partir de los 90's en un artículo propuesto por Norman. Es un algoritmo inspirado en los principios de evolución de Darwin y los principios de Dawkins con la noción de meme, unidad de información teórica cultural, para el aprendizaje individual. En otras palabras, busca la sinergia de la evolución y el aprendizaje individual (Moscato y Cotta, 2003).

Pseudocódigo 3 Pseudocódigo Algoritmos meméticos

```
t ⇐ 0
repetir
  Evaluar Φ(G(t))
  Evolucionar G'(t) = (ρG(t))
  Seleccionar P(t) ⊆ G'(t)
  Aprendizaje P'(t)
  Seleccionar G(t + 1) ⊆ P'(t) ∪ G'(t)
  t ⇐ t + 1
hasta criterio de parada
```

El proceso básico se puede ver en el Pseudocódigo 3: se genera una población inicial, esta se evalúa, evoluciona como en los algoritmos evolutivos, luego se seleccionan agentes (solución individual) y para cada uno de ellos se correrán funciones de desempeño y aprendizaje lamarkiano y de Baldwian (Nguyen et al., 2007). Los parámetros de control son, además de los utilizados en GA, un operador llamado intensidad de aprendizaje. Se ha aplicado especialmente a problemas combinatorios y planeación. Dentro de ellos se destacan la planificación de tareas y algunas aplicaciones en Inteligencia Artificial (IA).

A.3. BÚSQUEDA TABÚ (TS DEL INGLÉS TABU SEARCH):

Inspirada en la memoria, hace una búsqueda inteligente evitando las soluciones prohibidas y trata de buscar exhaustivamente, de manera determinista, tratando de escapar de los mínimos locales. Fue introducida y estructurada como Búsqueda Tabú por Glover y Laguna en 1986 (Glover et al., 1995).

Pseudocódigo 4 Pseudocódigo Búsqueda Tabú

$t \Leftarrow 0$
Inicializar $T(t) \Leftarrow \{\}$
Punto Inicial $G(t)$;
Evaluar $\Phi(G(t))$
repetir
  Generar $H(t)$
  Evaluar $\Phi(H(t))$
  Seleccionar $G'(t) \in H(t) \wedge G'(t) \notin T(t)$
  Actualizar $T(t+1); G(t+1) = G'(t)$
  $t \Leftarrow t+1$
hasta criterio de parada

En el Pseudocódigo 4 se muestra la secuencia del algoritmo: inicia con una solución inicial $G(t)$ y esta pasa a la más cercana de su vecindario $H(t)$. Estas soluciones se guardan en la memoria denominada, lista tabú $T(t)$, para evitar volver sobre ellas. Los parámetros de control de este algoritmo son: el tamaño de la lista tabú, si ésta es penalizada o no y los criterios de parada. Es utilizado especialmente para problemas de planificación, transporte, grafos (Glover et al., 1995; Crespo et al., 2007).

A.4. TEMPLE SIMULADO (SA DEL INGLÉS SIMULATED ANNEALING):

Inspirado en el proceso de temple de los metales donde al controlar la temperatura se da una mejor cristalización de la microestructura del material. Propuesto inicialmente por Krikpatrick, Gelatt y Vecchi en los años 80's.

El proceso básico se muestra en el Pseudocódigo 5: a partir de una solución inicial $G(t)$ se evalúa la función objetivo, se selecciona un paso que provea un nuevo x', se calcula nuevamente la función objetivo, de acuerdo a la función de aceptación se determina el nuevo valor de $G(t)$, el nuevo $G(t)$ se selecciona de acuerdo a una función de probabilidad que es función de la temperatura y el valor de la función objetivo para el punto. Luego se reduce la temperatura de acuerdo a una función inicialmente dada para el algoritmo y se repite el proceso hasta cumplir el número de iteraciones esperadas o el criterio de parada dado (Rao, 1996). Los parámetros principales son: criterio de parada, iteraciones máximas, función de Temperatura y la forma de seleccionar la nueva generación. Las aplicaciones típicas son para problemas combinatoriales, como problemas de planeación, redes neuronales entre otras (Kirkpatrick et al., 1983).

Pseudocódigo 5 Pseudocódigo Temple simulado

$t \Leftarrow 0; T \Leftarrow T_0$
Punto Inicial $G(t) = x_0$;
Evaluar $\Phi(G(t))$
repetir
  Generar $G(t)$
  Evaluar $\Phi(G(t))$
  Seleccionar $G'(t)$ de acuerdo a la función de probabilidad establecida
  Reducir $T(t+1) = \eta T(t)$
  $t \Leftarrow t+1$
hasta criterio de parada

A.5. BÚSQUEDA DE DISPERSA (SS DEL INGLÉS SCATTER SEARCH):

Propuesto por Glover en los años 70's , dado un conjunto de soluciones, esta heurística, combina las soluciones iniciales sistemáticamente produciendo nuevas soluciones por medio de combinaciones lineales; las nueva soluciones son los centroides entre las soluciones anteriores.

En el Pseudocódigo 6 se muestra la secuencia, para un conjunto inicial de soluciones $G(t)$ se selecciona un subconjunto $P(t)$ de las mejores soluciones; a partir de este conjunto P se generan k subconjuntos y se combinan, evalúan y cambian; luego se vuelve a seleccionar un conjunto para la siguiente iteración hasta llegar a un número de iteraciones o un criterio de parada específico (Martí y Laguna, 2003). Se ha aplicado especialmente en problemas combinatorios, entrenamiento de redes neuronales, planeamiento de tareas, problemas de transporte (Corne et al., 1999). Es de los métodos con mas híbridos o que su estrategia es utilizada por otros métodos.

Pseudocódigo 6 Pseudocódigo Búsqueda dispersa

```
t ⇐ 0
Generar G(t)0;
repetir
  Evaluar Φ(G(t))
  Evaluar P(T) ⊆ (G(t)) conjunto de referencia
  Generar Q_K(T) ⊆ P(T)
  para i = 1 hasta k hacer
    Combinar Q_i(t)
    Evaluar Φ(Q_i(t))
    i ⇐ i + 1
  fin para
  Seleccionar G(t + 1)
  t ⇐ t + 1
hasta criterio de parada
```

A.6. OPTIMIZACIÓN DE ENJAMBRE DE PARTÍCULAS (PSO DEL INGLÉS PARTICLE SWARM OPTIMIZATION):

Se inspira en el comportamiento gregario presente en la naturaleza, donde se proveen dos tipos de conocimientos, el particular y el colectivo. Busca el patrón emergente de las decisiones particulares de los individuos. Fue propuesta por Kennedy y Eberhart (1995).

Básicamente funciona, como se muestra en el Pseudocódigo 7: se tiene una población de individuos $G(t)$ que tienen un punto inicial x y una función de trayectoria v y cada uno tiene una experiencia mejor p de manera que cada individuo oscila alrededor de P de acuerdo a su velocidad, y cada individuo sigue al que tiene el mejor ajuste , es decir al líder (Yiqing et al., 2007). Tiene como parámetros de control, el tamaño del paso, el tamaño del enjambre y la inferencia del lider respecto a otros individuos. A pesar de su reciente aparición ha tenido muy buena acogida, en el entrenamiento de redes neuronales, ajuste de elementos finitos, entre muchas otras aplicaciones (Talbi, 2009).

Pseudocódigo 7 Pseudocódigo Optimización de enjambre de partículas

$t \Leftarrow 0$;*ibest* $\Leftarrow 0$;*best* $\Leftarrow 0$
Generar $G(t) = \{\vec{a_1}, ..., \vec{a_k}\}$;
Evaluar $\Phi(G(t))$
Seleccionar *ibest* $\Leftarrow min(\Phi(G(t)))$
repetir
  Calcular $(G'(t)) \Leftarrow (G(t)) + v(t)$
  Evaluar $\Phi(G'(t)$
  Actualizar *best* de cada partícula
  Seleccionar *ibest* $\Leftarrow min(\Phi(G(t)))$
  $t \Leftarrow t + 1$
hasta criterio de parada

A.7. OPTIMIZACIÓN DE COLONIA DE HORMIGAS (ACO DEL INGLÉS ANT COLONY OPTIMIZATION)

Se inspira en la comunicación indirecta de una colonia de agentes simples, hormigas, emulando el funcionamiento de las feromonas. A medida que se pasa por un mismo camino estas envían un mensaje a toda la colonia, donde las mejores soluciones son marcadas y recordadas en la memoria colectiva, como los caminos de las hormigas en la naturaleza. Fue propuesto por Dorigo a principios de los 90's (Dorigo y Blum, 2005).

El proceso básico se muestra en el Pseudocódigo 8: se constituye una población de m hormigas, estas se mueven a través de estados adyacentes creando caminos, estos movimientos se hace por decisiones individuales dadas por políticas de decisión individuales del algoritmo y usan los rastros de feromonas T. Aparte de esto hay dos procesos importantes; la evaporación de feromonas que buscan salir de las soluciones óptimas locales y acciones emergentes que no puede hacer un individuo, como aplicar mas feromonas a cierto camino; estas dos estrategias se implementan al actualizar en cada iteración. Se aplica principalmente a problemas de nodos, como al problema del agente viajero, planificación de tareas entre otras (Dorigo y Stützle, 2003).

Pseudocódigo 8 Pseudocódigo Optimización de colonia de hormigas

$t \Leftarrow 0; T \Leftarrow T_0$
Punto Inicial x; inician m partículas
repetir
  Construcción de las respuestas iniciales $(G(t))$
  Actualizar feromonas individuales τ
  evaluar $\Phi(G(t))$
  Actualizar feromonas Globales $T(t)$
  $t \Leftarrow t + 1$
hasta criterio de parada

A.8. SISTEMA INMUNE ARTIFICIAL (AIS DEL INGLÉS ARTIFICIAL IMMUNE SYSTEMS)

Inspirados en el complejo sistema inmune de los vertebrados que emulan el sistema de aprendizaje y memoria de los antígenos. Propuesto por De castro y Timmis en los 90's , nace inicialmente de algoritmos

clonales, propuestos también por de Castro y Timmis (2003).

El algoritmo sigue la secuencia básica de un algoritmo evolutivo que se muestra en el Pseudocódigo 2, pero en este caso las soluciones son anticuerpos. Se inicia con la generación de una población de anticuerpos y se evalúa la afinidad de cada uno respecto a una función de evaluación (Antígeno). En este paso, se pueden reemplazan, clonar o hipermutar los anticuerpos hasta llegar a un número de individuos deseados, completando así, la generación, Esta generación completa se vuelve a evaluar con el antígeno y se repite el proceso hasta llegar al criterio de parada (Musilek et al., 2006). A partir de este paradigma nacen otros algoritmos como: la programación inmunológica, algoritmos virales, entre otros.

A.9. GRASP (DEL INGLÉS GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURES)

Propuesto por Feo y Resende en los 90's toma su inspiración en el aprendizaje; busca mejorar su heurística en cada iteración por medio de la exploración con el constructor de soluciones codicioso y la explotación por medio de búsqueda local (Resende y Ribeiro, 2002).

La secuencia básica se muestra en el Pseudocódigo 9: Tiene una fase inicial de construcción donde se genera la población y se genera la lista de más aptos y se toma uno. A partir de este individuo se busca localmente hasta encontrar el mejor, luego se compara con el mejor hallado y si es mejor se actualiza esta lista, se vuelve a repetir el proceso hasta el criterio de parada (Hirsch et al., 2006). Se aprovechan las bondades de esta exploración local hibridizando el método con algoritmos de búsqueda más globales.

Pseudocódigo 9 Pseudocódigo GRASP

$t \Leftarrow 0; f* \Leftarrow \infty$; $best \Leftarrow 0$
Solución inicial $G(t)$;
repetir
  para $i = 1$ hasta k **hacer**
    hallar un $(G'(t))$ con el constructor codicioso
    hallar un $(G''(t)) < (G'(t))$ con un método de búsqueda local
    $i \Leftarrow i + 1$
  fin para
  Actualizar *best*
  $t \Leftarrow t + 1$
hasta criterio de parada

A.10. QUIMIOTAXISIS BACTERIANA (BC DEL INGLÉS BACTERIAL CHEMOTAXIS):

Propuesto por Berg y Brown en 1972 se inspira en el proceso de quimiotaxstimo bacterial; describe la forma en que las bacterias se mueven de acuerdo al posicionamiento de la comida en el ambiente en el cual se encuentra (Muller et al., 2000).

El proceso básico se muestra en el Pseudocódigo 10: el proceso básicamente es la trayectoria de una bacteria, que escoge un ángulo θ y una duración τ de acuerdo a unas funciones probabilísticas dadas. Este algoritmo ha sido modificado permitiendo la optimización de los parámetros por otros métodos (Muller et al., 2000).

Pseudocódigo 10 Pseudocódigo Quimotaxisis Bacteriana

$t \Leftarrow 0$; calcular parámetros iniciales
Solución inicial $G(t)$;
Evaluar $\Phi G(t)$;
repetir
  calcular $\theta(t), \tau, yl$
  Evaluar $\Phi(G(t) + l)$
  Seleccionar
  si $\Phi(G(t) + l) < \Phi(G(t))$ **entonces**
    $G(t + 1) \Leftarrow G(t) + l$
  sino
    $G(t + 1) \Leftarrow G(t)$
  fin si
  $t \Leftarrow t + 1$
hasta criterio de parada

A.11. GRAN EXPLOSIÓN-GRAN IMPLOSIÓN (BB-BC DEL INGLÉS BIG BANG-BIG CRUNCH)

Este método fue propuesto por Erol and Eksin en 2006 donde se proponen una heurística inspirada en la teoría del big bang, en el Pseudocódigo 11 se muestra como funciona; inicialmente una explosión aleatoria de k soluciones sobre una malla de números enteros e implota a un único punto que corresponde al centro de masa de las partículas aleatorias iniciales, iterando en la explosión e implosión disminuyendo la distancia de la explosión. En la literatura se presentan pocas aplicaciones, en control y vigas dado lo reciente del método (Erol y Eksin, 2006).

Pseudocódigo 11 Pseudocódigo Gran explosión- Gran implosión

$t \Leftarrow 0$;
Discretizar f;
repetir
  Generar soluciones iniciales $G(t) = \{\vec{a_1}, ..., \vec{a_k}\}\vec{a} \in \mathbb{Z}$;
  Evaluar $\Phi G(t)$;
  calcular centro de masa de $G(t)$
  Reducir fuerza $\gamma \Leftarrow \alpha\gamma$
  $t \Leftarrow t + 1$
hasta criterio de parada