# PROGRAMACIÓN NO LINEAL USANDO ALGORITMOS EVOLUTIVOS: CASOS DE ESTUDIO

Tutor: Prof. Eliezer Colina
Cotutor: Prof. Sebastián Medina
Autor: Br. María Victoria Marcano C

PROYECTO DE GRADO PRESENTADO ANTE LA UNIVERSIDAD DE LOS ANDES COMO REQUISITO FINAL PARA OPTAR AL TITULO DE INGENIERO DE SISTEMAS

Mérida, Venezuela
Mayo 2006

*A DIOS,*
*LA VIRGEN DEL VALLE,*
*MI PAPÁ, MI MAMÁ,*
*Y A MIS HERMANOS*

## *AGRADECIMIENTOS*

Primero que todo, quisiera agradecer a mi tutor, Profesor Eliezer Colina Morles, por su apoyo y guía en esta investigación.

Al Profesor Sebastián Medina por su apoyo y confianza.

Al Profesor Ernesto Ponsot, por brindarme su ayuda incondicional.

Al Ingeniero Andrade, por su amistad sincera y ayuda desinteresada.

# *ÍNDICE*

**3 MARCO METODOLÓGICO**

# PROGRAMACION NO LINEAL USANDO ALGORITMOS EVOLUTIVOS: CASOS DE ESTUDIO

Br. María Victoria Marcano Cabrera.

Tutor: Prof. Eliezer Colina Morles.

Proyecto de Grado — Ingeniería de Sistemas
Area: Investigación de Operaciones, páginas 98

**Resumen:**

Este trabajo contempla el desarrollo de un Algoritmo Evolutivo para la resolución de problemas de Programación Matemática, particularmente problemas de programación lineal y Programación no lineal. Para este propósito, el Algoritmo Evolutivo se fundamentará en un método de búsqueda estocástica del óptimo del problema a ser considerado, inspirado en la evolución de las potenciales soluciones del mismo en la dirección del negativo del gradiente de una expresión que incluye, tanto a la función objetivo como a las restricciones del problema ponderadas por multiplicadores de Lagrange.

La solución matemática que da respaldo conceptual a este enfoque, incluyendo sus características de convergencia, han sido previamente reportadas en la literatura [1].

El Algoritmo Evolutivo será desarrollado usando Matlab y sus características de desempeño serán ilustradas por medio de la resolución de algunos problemas sencillos de optimización no lineal.

**Palabras claves:** Algoritmos genéticos, Programación no lineal, Optimización, Gradiente.

# *INTRODUCCIÓN*

Los Algoritmos Evolutivos, son métodos estocásticos de optimización, que con bases en la evolución y genética natural, permiten encontrar la solución óptima o casi óptima a problemas donde se requiera optimizar (maximizar o minimizar) una función sujeta a ciertas restricciones.

El Algoritmo evolutivo que se desarrollará en este trabajo está basado en un enfoque de optimización fundamentado en el uso del gradiente descendente [1] y hará uso de dos operadores genéticos: selección y mutación, para encontrar la solución a problemas de tipo no lineal.

La Programación no Lineal es una herramienta matemática utilizada para resolver problemas donde se requiere optimizar (maximizar, minimizar) una función objetivo de tipo no lineal sujeta a restricciones que pueden ser o no del mismo tipo.

Matemáticamente, este tipo de problemas pueden ser formulados como:

$$Max_X \ f(X)$$
$$Donde \ f(X) = f(x_1, x_2, .....x_n)$$
$$Sujeto \ a:$$
$$X \in Q = \{X / g_i(X) \leq 0, i = 1,2,3,...., m \ \}$$

Este trabajo ha sido estructurado de la siguiente manera: El Capítulo uno incluye la formulación del problema a ser abordado, así como los alcances de los resultados a ser alcanzados en su desarrollo.

El segundo capítulo contempla una breve introducción a algunos métodos clásicos de resolución de problemas de programación no lineal; así como también aspectos relevantes sobre el diseño de Algoritmos Genéticos en general y sobre Algoritmos Evolutivos, en particular.

En el capítulo tres se hace el diseño e implantación computacional del Algoritmo Evolutivo, basado en el enfoque del gradiente descendente, que servirá de base para la búsqueda de soluciones óptimas a problemas de programación lineal y no lineal. Igualmente se incluyen las formulaciones de algunos problemas típicos y las soluciones obtenidas desde el punto de vista de métodos clásicos y desde el ángulo del Algoritmo Evolutivo implantado.

El capítulo cuatro se centra en la discusión de los resultados obtenidos y de las características de convergencia del Algoritmo Evolutivo diseñado.

El trabajo también incluye anexos sobre el procedimiento que sirvió de base para el desarrollo del Algoritmo Evolutivo implantado.

# *CAPÍTULO I.*

## 1.1 INTRODUCCIÓN

En la actualidad se emplean diversos enfoques metodológicos para la solución a problemas de ingeniería inspirados en ciertos mecanismos de la naturaleza, tales como Redes Neuronales Artificiales inspiradas en las neuronas biológicas, Algoritmos Genéticos con fundamentación en la evolución biológica y los Sistemas Lógicos Difusos que están basados en la modelación del razonamiento a través del lenguaje. En particular estos enfoques metodológicos pueden ser empleados para resolver problemas de optimización.

Los Algoritmos Genéticos constituyen una familia de modelos computacionales, motivados por la evolución. Estos algoritmos permiten obtener una solución potencial para un problema especificado en una estructura de datos denominada cromosoma y permite aplicar ciertos operadores o transformaciones sobre estas estructuras para manejar la información en forma genética. De esta manera es factible obtener la solución a problemas de búsqueda estocástica, que conducen a la solución de una variedad de problemas de optimización.

En el trabajo que se presenta a continuación se definen los aspectos más importantes de un algoritmo genético y con base en sus operadores: selección, mutación y sustitución, se procede a desarrollar e implantar computacionalmente un algoritmo evolutivo, que basado en el método del gradiente descendente, es capaz de resolver problemas de programación no lineal.

## 1.2 FORMULACIÓN DEL PROBLEMA

Se propone la construcción de un Algoritmo Evolutivo para encontrar la solución de problemas Programación Matemática, en lo particular, problemas de programación lineal y no lineal. El Algoritmo Evolutivo a ser desarrollado está inspirado en el trabajo señalado en [1], donde se requiere la codificación de los miembros de una población de potenciales soluciones candidatas en términos de números reales y la aplicación de un operador de mutación basado en gradiente descendente sobre una función objetivo sujeta a restricciones.

Haciendo uso de procedimientos sugeridos en los Algoritmos Genéticos, se realizará el diseño e implantación de un sistema programado para acometer la búsqueda de soluciones a problemas de optimización de tipo no lineal. Se considerarán cuatro problemas sencillos de programación no lineal. Se aplicará el Algoritmo Evolutivo a un problema en particular y posteriormente con solo cambiar las derivadas y restricciones se procederá a darle solución al resto de los problemas, para validar el algoritmo.

De manera más puntual, los problemas de Programación no Lineal a ser desarrollados en este trabajo, son los siguientes:

1.- Determinación de la política óptima de extracción de crudos de una empresa petrolera, que dispone de capital, equipos de infraestructura, pozos petroleros y demás elementos necesarios para llevar a cabo el proceso antes mencionado.

2.- Encontrar el triángulo de mayor área inscrito en un triángulo isósceles ABC cuya altura y vértice C son conocidos.

3.- Determinar la solución óptima a dos problemas de tipo netamente académicos, como lo son:

$$Max\ f = x_1 x_2 x_3$$
$$sujeta\ a\ x_1 + x_2 + x_3 = L \quad \text{y}$$
$$x_1, x_2, x_3 \geq 0$$

$$Max\ f = x_1^2 + x_2^2 + x_3^2$$
$$sujeta\ a: x_1 + x_2 = 6$$
$$x_1 - x_2 = 2$$
$$x_1, x_2, x_3 \geq 0$$

## 1.4 OBJETIVOS

General:

Diseñar un Algoritmo Evolutivo especial, basado en el uso del gradiente descendente [1], utilizando una herramienta adecuada (MATLAB) para encontrar la solución a problemas de Programación Matemática, en particular, problemas de Programación lineal y no lineal.

Específicos:

Comprender desde el punto de vista matemático y computacional, el trabajo que bosqueja el procedimiento para encontrar la solución a problemas de programación no lineal señalado en [1].

Implantar Computacionalmente el Algoritmo Evolutivo para resolver estrategias de optimización.

Comparar la solución óptima arrojada por el Algoritmo Evolutivo construido con la solución analítica del problema, para así validar el programa realizado.

# *CAPITULO II. MARCO TEÓRICO*

## 2.1 INTRODUCCIÓN

Para la resolución de problemas de programación no lineal existen diversas técnicas dentro de las cuales se pueden mencionar:

Técnicas Analíticas:

- Puntos interior
- Gradientes
- Lagrangiano
- Puntos de Esquina

Técnicas Empíricas:

- Diseño del Algoritmo Evolutivo para la búsqueda de la solución óptima (esta es la técnica de interés en este trabajo)

La Evolución Natural es un proceso de cambio sobre una población reproductiva, que contiene variedades de individuos con algunas características hereditables y en donde algunas variedades difieren en su aptitud.

Las especies nacen, evolucionan y desaparecen si no se adaptan, solo los mejores, los más aptos, los que mejor se adapten al medio sobreviven para perpetuar sus aptitudes.

## 2.2 ANTECEDENTES

La naturaleza encuentra brillantes soluciones a los problemas mediante la evolución continuada de las especies. Ligeras variaciones aleatorias en los genes de los descendientes, dan lugar a individuos con diferente capacidad de adaptación a su entorno. Primando la reproducción de los mejores adaptados, la especie evoluciona a formas cada vez más eficientes para sobrevivir en su entorno.

Hablamos de Computación Evolutiva siempre que disponemos de una población de individuos y ejercemos sobre ésta procesos de selección, evaluación, y recombinación de la información. Es decir, nos referimos a aquellos algoritmos que se inspiran en el proceso de evolución natural (del que todos formamos parte),son los individuos mejor evaluados o más adaptados (con mayor aptitud o fitness) los que más tienden a reproducirse es por ello que tienen mayor probabilidad de ser seleccionados y transmitir sus "genes" a nuevos individuos [4].

La Computación Evolutiva es una rama de la Computación Emergente que engloba técnicas que simulan la Evolución Natural. Con el término de Computación Evolutiva o Algoritmos Evolutivos, se engloba un conjunto de técnicas que basándose en la imitación de varios procesos naturales que intervienen en la evolución de las especies (selección natural, mutaciones, cruces) tratan de resolver complejos problemas de búsqueda, optimización, aprendizaje, predicción o clasificación [4].

La Computación Evolutiva trata de obtener aprendizaje imitando la evolución. Son los métodos que basan su funcionamiento en una metáfora de la evolución biológica.

Dentro de la Computación Evolutiva (CE) también llamada Algoritmos Evolutivos (AE) se encuentran:

- Solidificación Simulada (SS)
- Algoritmos Genéticos (AG)
- Programas de Evolución
- Estrategias Evolutivas (EE)
- Programación Genética (PG)
- Programación Evolutiva (PE)
- Clasificadores Genéticos (CG)
- Algoritmos Miméticos

Partiendo del conocimiento del funcionamiento de los Algoritmos Genéticos (probablemente la técnica más conocida de todas ellas), el resto de los algoritmos se

pueden interpretar como variaciones o mejoras de los Algoritmos Genéticos, añadiendo complejidad o modificando ciertas características del algoritmo.

Los algoritmos evolutivos y genéticos son herramientas que se han mostrado eficientes para resolver problemas de optimización de funciones complicadas con un número elevado de variables

Los Algoritmos Genéticos, pueden ser considerados como procedimientos de búsqueda aleatoria fundamentados en los principios biológicos de la selección natural y la transmisión de características hereditarias. La idea básica parte del establecimiento de una población inicial de soluciones candidatas, (a un problema planteado), que constituyen individuos que deben competir entre sí por su supervivencia. Un proceso de evaluación permitirá seleccionar los individuos más fuertes; que en comparación con los más débiles, tendrán mayor posibilidad de "procrear" nuevos individuos. Los nuevos individuos o descendientes se producen como consecuencia de procesos de recombinación y/o mutación que les confieren características genéticas similares a las de sus padres y/o características genéticas innovadoras respecto a las de sus ascendentes. De esta manera, la población inicial de individuos evoluciona, por medio de un proceso de selección donde padres e hijos compiten, en procura de "mejorar genéticamente" las poblaciones resultantes [5].

La biología y la genética han establecido que todo organismo viviente consiste en un grupo de células, y cada célula contiene el mismo tipo de cromosomas. Los cromosomas son básicamente cadenas de Ácido Desoxirribonucleico (ADN), que sirven como modelo para la creación del organismo. Cada gen codifica la síntesis de un grupo de proteínas en particular y estas proteínas determinan el color de los ojos, el del cabello, la estatura, entre otras características.

La base de los algoritmos genéticos es que dada la elevada tasa de reproducción de todos los seres orgánicos, su número tiende a crecer a ritmo exponencial, y dado a que el espacio físico no lo hace en la misma proporción, nacerán muchos más individuos de los que es posible que sobrevivan, y en consecuencia, se recurrirá a la

lucha por la existencia, ya sea con individuos de la misma especie o de especies diferentes, o simplemente con el entorno, intentando modificar sus características adversas. El resultado final es que los seres vivos tienden a perfeccionarse en relación con las circunstancias que los envuelven.

Un amplio rango de problemas de optimización contienen funciones objetivos no lineales que envuelven variables reales o continuas y variables enteras o discretas. Esta clase de problemas de optimización comprende una gran variedad de aplicaciones entre ellas la síntesis y diseño integrado de procesos; y son denotadas como problemas de programación no lineal.

La Programación no lineal es una herramienta matemática empleada para resolver problemas donde se requiere optimizar (maximizar o minimizar) una función objetivo no lineal sujeta a restricciones de tipo lineal o no lineal.

El problema de Programación no lineal puede ser descrito de la siguiente forma general:

**Función objetivo:**

$$Max_X \ f(X)$$
$$Donde \ f(X) = f(x_1, x_2, .....x_n)$$

**Sujeto a:**

$$X \in Q = \{X / g_i(X) \leq 0, i = 1,2,3,...., m \ \}$$

Existen numerosos métodos analíticos clásicos para resolver problemas de programación no lineal. El más utilizado es el método de *Branch & Boun*(B&B), el cual es un método de optimización analítico que teóricamente garantiza la búsqueda de la mejor solución de problemas lineales y no lineales convexos. El B&B resuelve el problema como si fuera de programación lineal o no lineal (variables reales), sin tomar

en cuenta que algunas variables pueden ser enteras. (Edgar, Himmelblau, Lasdon, 2001) [7].

La diferencia fundamental entre los algoritmos genéticos y los métodos tradicionales se debe principalmente a que los Algoritmos Genéticos trabajan con un código del conjunto de parámetros, no con los parámetros en sí, además de que emplean la información de una función objetivo y no se valen de derivadas u otro conocimiento auxiliar. También se valen de reglas de transición probabilísticas mas no de reglas determinísticas.

## 2.3 MÉTODOS TRADICIONALES

Si un problema está formulado de forma unívoca, y el espacio de todas las soluciones posibles es conocido, un método de búsqueda trata de encontrar la solución o soluciones que satisfagan el enunciado del problema. Dicho así parece una tautología, pero no todos los problemas son problemas de búsqueda, ni todos los espacios de búsqueda se pueden definir también de forma unívoca.

Pero si resulta que efectivamente, un problema se puede efectuar como problema de búsqueda, en muchos casos se puede reducir a un problema de hallar un máximo o mínimo, o sea, un óptimo. Sólo va a funcionar esto en el caso de que se pueda calcular la bondad de una solución: la solución del problema será aquella, o aquellas, que optimicen una función de bondad, ajuste, evaluación o fitness; en muchos casos, por lo tanto, un problema de búsqueda se puede reducirá un problema de optimización (maximización o minimización).

A continuación serán presentados algunos métodos analíticos clásicos de búsqueda, empleados para encontrar la solución óptima a problemas de Programación matemática, en particular, problemas de programación no lineal.

### 2.3.1 Método de Lagrange:

Para hacer uso de este método es necesario conocer la función de interés u objetivo ( $f(\bar{x})$, con $\bar{x} = x_1, x_2, x_3, \ldots\ldots x_n$; donde $n$ es igual al nº de variables ) y las

restricciones ( $g_m(\bar{x})$ ,donde m es igual al nº de restricciones que posee el problema) que conformarán la función lagrangiana $L(\bar{x},\bar{\lambda})$.

El problema de programación no lineal debe tener la siguiente estructura:

$$
\begin{aligned}
&opt\ (\max o \min)\ \ f(\bar{x})\\
&Sujeto\ a:\ g_1(\bar{x}) = \bar{0}\\
&\qquad\quad g_2(\bar{x}) = \bar{0}\\
&\qquad\quad g_m(\bar{x}) = \bar{0}
\end{aligned}
$$

La función de interés $f(\bar{x})$ debe ser de tipo no lineal, pero las restricciones pueden o no ser de este tipo. La función Lagrangiana es aquella que le asignará el valor de aptitud a cada individuo y de ella dependerá la participación de este en la próxima generación. Dicha función se expresa como:

$$L(\bar{x},\bar{\lambda}) = f(\bar{x}) + \lambda_1 g_1(\bar{x}) + \lambda_2 g_2(\bar{x}) + .....\lambda_m g_m(\bar{x})$$

Donde $f(\bar{x})$ es la función objetivo, $\lambda$ son los multiplicadores de Lagrange (por cada restricción tendremos un multiplicador), $g_m(\bar{x})$ es la restricción.

Para encontrar la solución al problema de optimización planteado, es necesario encontrar el gradiente de la función lagrangiana e igualarlo a cero, $\nabla L(\bar{x},\bar{\lambda}) = 0$.

Debe realizarse el estudio del Hessiano, como condición suficiente, para verificar que la solución arrojada es la requerida (máxima o mínima).

El Hessiano para un problema que posee dos variables y una restricción, se escribe:

$$H = \begin{bmatrix} \frac{\partial^2 L}{\partial x_1^2} & \frac{\partial^2 L}{\partial x_1 \partial x_2} & \frac{\partial^2 L}{\partial x_1 \partial \lambda_1} \\ \frac{\partial^2 L}{\partial x_2 \partial x_1} & \frac{\partial^2 L}{\partial x_2^2} & \frac{\partial^2 L}{\partial x_2 \partial \lambda_1} \\ \frac{\partial^2 L}{\partial \lambda_1 \partial x_1} & \frac{\partial^2 L}{\partial \lambda_1 \partial x_2} & \frac{\partial^2 L}{\partial \lambda_1^2} \end{bmatrix}$$

Dependiendo del signo que posean todas las derivadas parciales que forman la matriz Hessiana, se tendrá la certeza de que el punto encontrado es el óptimo, es decir, si el Hessiano es:

Definido negativo, estamos en presencia de un máximo.

Definido positivo, estamos en presencia de un mínimo.

Ni definido positivo, ni definido negativo, sin información, punto de silla.

Cero: sin información.

### 2.3.2 Método de Hancock:

Debemos aplicar este método cuando al calcular el Hessiano, no encontramos información suficiente, para concluir si estamos en presencia de un valor máximo o mínimo.

Condiciones necesarias de Hancock:

Que el determinante de la siguiente matriz sea cero, así se calcula z, que es la raíz característica de H y determina si el punto óptimo encontrado es un máximo o mínimo:

$$\begin{vmatrix} L_{11}-z & L_{12} & \ldots\ldots\ldots & L_{1n} & g_{11}\ldots\ldots & g_{m1} \\ L_{21} & L_{22}-z & \ldots\ldots\ldots & L_{2n} & g_{12}\ldots\ldots & g_{m2} \\ \vdots & & & & \vdots & \\ L_{m1} & L_{m2} & \ldots\ldots\ldots & L_{nn}-z & g_{1n}\ldots\ldots & g_{mm} \\ g_{11} & \ldots\ldots & \ldots\ldots & g_{1n} & 0\ldots\ldots & 0 \\ & & & & & \\ g_{m1} & \ldots\ldots & \ldots\ldots & g_{mn} & 0\ldots\ldots & 0 \end{vmatrix} = 0$$

Donde: $L_{ij} = \frac{\partial^2 L}{\partial x_i \partial x_j}; g_{ij} = \frac{\partial g_i}{\partial x_j}$

El polinomio en z resultante de resolver el determinante, especificado anteriormente debe ser de orden n-m (n = nº de variables, m = nº de restricciones)

Si $z_{(k)} > 0$, entonces estoy en presencia de un mínimo.

Si $z_{(k)} < 0$, entonces estoy en presencia de un máximo.

Si $z_{(k')} < 0 \ y \ z_{(k'')} > 0$, entonces $\bar{x}*$ no es un punto extremo.

$k'$ y $k'' = k$, es decir número de raíces.

**2.3.3 Método de Khun-Tucker:**

Para poder hacer uso de este método debemos tomar en cuenta si la función objetivo requiere maximizar o minimizar, si las restricciones son de mayor igual o de menor igual y el valor de los multiplicadores de lagrange, de esta manera se garantiza que el valor encontrado es el óptimo,es decir:
Si la función objetivo, requiere:

- Minimizar y posee restricciones de menor igual, los multiplicadores deben resultar mayores o iguales a cero.

- Maximizar, con restricciones de menor igual, los multiplicadores deben ser menores o iguales a cero.

- Minimizar, con restriccioines de mayor igual, los multiplicadores deben ser menores o iguales a cero.

- Maximizar, con restricciones de mayor igual, los multiplicadores deben resultar mayores o iguales a cero.

En resúmen:

| Si la $f$ | s.a restricción | los $\lambda$ deben ser |
|---|---|---|
| *Min* | $\leq$ | $\geq 0$ |
| *Max* | $\leq$ | $\leq 0$ |
| *Min* | $\geq$ | $\leq 0$ |
| *Max* | $\geq$ | $\geq 0$ |

El problema de Programación no Lineal debe poseer la siguiente estructura.

$$Min\ f(\bar{x})$$
$$sujeta\ a: g_1(\bar{x}) \leq 0$$
$$g_2(\bar{x}) \leq 0$$
$$g_m(\bar{x}) \leq 0$$

Para hallar la solución a problemas de programación matemática, en particular problemas de Programación no lineal, utilizando el método de Khun-Tucker debemos encontrar la derivada parcial de la función lagrangiana con respecto a cada una de las variables e igualarla a cero: $\frac{\partial L}{\partial x_i} = 0$, así mismo, cada restricción acompañada por su respectivo multiplicador deberá ser igualado a cero: $\lambda_j g_j = 0$ .Estas serán las nuevas restricciones que posee el problema.

Por ejemplo, si se quiere:

$$Min\ f = x_1^2 + x_2^2 + x_3^2 + 20(x_1 - 50) + 20(x_1 + x_2 - 100)$$
$$s.a\ x_1 \geq 50$$
$$x_1 + x_2 \geq 100$$
$$x_1 + x_2 + x_3 \geq 150$$

Aplico Condiciones de Khun-Tucker.

$$g_1 = (x_1 - 50)$$
$$g_2 = (x_1 + x_2 - 100)$$
$$g_3 = (x_1 + x_2 + x_3 - 150)$$

$$\frac{\partial f}{\partial x_1} = 2x_1 + 20 + 20 + \lambda_1 + \lambda_2 + \lambda_3 = 0$$

$$\frac{\partial f}{\partial x_2} = 2x_2 + 20 + \lambda_2 + \lambda_3 = 0$$

$$\frac{\partial f}{\partial x_3} = 2x_3 + \lambda_3 = 0$$

$$\lambda_1 (x_1 - 50) = 0$$

$$\lambda_2 (x_1 + x_2 - 100) = 0$$

$$\lambda_3 (x_1 + x_2 + x_3 - 150) = 0$$

$$\lambda_1 = \lambda_2 = \lambda_3 = 0, \text{ encuentro } x_1^*, x_2^*, x_3^*$$

$$\lambda_1 = 0, \lambda_2 = 0, \lambda_3 \neq 0; \text{ 4 ecuaciones, 4 incognitas}$$

$$\lambda_1 = 0, \lambda_2 \neq 0, \lambda_3 = 0$$

$$\lambda_1 \neq 0, \lambda_2 = 0, \lambda_3 = 0$$

$$\lambda_1 = 0, \lambda_2 \neq 0, \lambda_3 \neq 0$$

Ahora estamos en presencia de un sistema determinado, ya que posee 6 ecuaciones con 6 incognitas, seguidamente debemos encontrar las posibles soluciones y tomar la óptima a los fines del programa.

Para abordar los problemas desde el punto de vista empírico, es necesario conocer algunos términos o conceptos de gran importancia, que nos permitirán comprender de manera más sencilla las estrategias, métodos y operadores utilizados en el desarrollo del Algoritmo Evolutivo, los cuales se exponen a continuación.

## 2.4 COMPUTACIÓN EVOLUTIVA

Es un enfoque alternativo que permite abordar problemas complejos de búsqueda y aprendizaje a través de modelos computacionales de procesos evolutivos.

Parte de un hecho observado en la naturaleza: los organismos vivos poseen una destreza consumada en la resolución de los problemas que se les presentan, y obtienen sus habilidades, casi sin proponérselo, a través del mecanismo de la evolución natural**.**

La evolución se produce, en casi todos los organismos, como consecuencia de dos procesos primarios: la selección natural y la reproducción sexuall (cruce) [4].

## 2.5 ALGORITMOS EVOLUTIVOS

Término empleado para describir sistemas de resolución de problemas de optimización o búsqueda basados en el computador; que emplean modelos computacionales de algún conocido mecanismo de evolución como elemento clave en su diseño e implementación [4].

### 2.5.1.- Características

Métodos de generación de soluciones que partiendo de un conjunto de soluciones iniciales, emplean un conjunto de operadores de búsqueda para refinar ésta.

El refinamiento, es el mecanismo biológico de exploración dentro de la población de soluciones haciendo uso de operadores genéticos.

### 2.5.2.- Clasificación

- Estrategias Evolutivas:

Técnica desarrollada por Rechenberg y Schwefel, extendida por Herdy, Kursawe, y otros, fue diseñada inicialmente con la meta de resolver problemas de optimización discretos y continuos, principalmente experimentales y considerados difíciles.

Utiliza recombinación o cruce y la operación de selección, ya sea determinística o probabilística, elimina las peores soluciones de la población y no genera copia de aquellos individuos con una aptitud por debajo de la aptitud promedio [4].

- Programación Evolutiva:

Técnica introducida por Fogel, extendida por Burgin,Atmar y otros, inicialmente fue diseñada como un intento de crear inteligencia artificial.

El procedimiento es muy similar a las estrategias evolutivas con la diferencia de que no emplea la recombinación, de tal forma que son denominadas en conjunto algoritmos evolutivos como una manera de diferenciarlas de los algoritmos genéticos [4].

- Algoritmos Genéticos:

Modelan el proceso de evolución como una sucesión de frecuentes cambios en los genes, con soluciones análogas a cromosomas. El espacio de soluciones posibles es explorado aplicando transformaciones a éstas soluciones candidatas tal y como se observa en los organismos vivientes: cruce, selección, mutación [4].

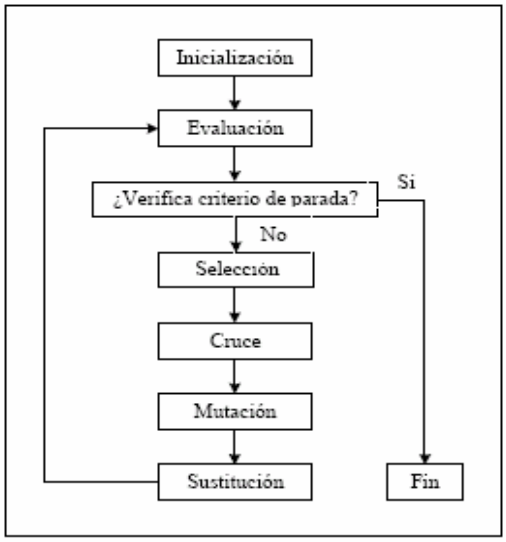
Constituyen el paradigma más completo de la computación evolutiva, resumen de modo natural todas las ideas fundamentales de dicho enfoque.

La idea básica en la utilización de un algoritmo genético para resolver problemas de optimización es representar las variables a optimizar en un cromosoma que será

evaluado para medir la calidad de esa solución. Se tiene una población de cromosomas que se someten a los operadores de selección, cruce, mutación y sustitución para obtener una solución óptima. [8]

En la inicialización se genera una población de posibles soluciones escogidas al azar, a través de una combinación lineal de los valores máximos y mínimos, padres que luego serán evaluados para ver si cumplen las restricciones, en caso de que no cumplan serán penalizados. Posteriormente se someten a los procesos de cruce, selección, mutación y sustitución hasta que se cumpla el criterio de parada. En la evaluación se determina la función objetivo de cada individuo de la población, penalizando a los individuos que no cumplan con las restricciones. [8]

## 2.6 ESQUEMA DEL PROCEDIMIENTO GENERAL DE UN A.G




## 2.7 OPERADORES GENÉTICOS

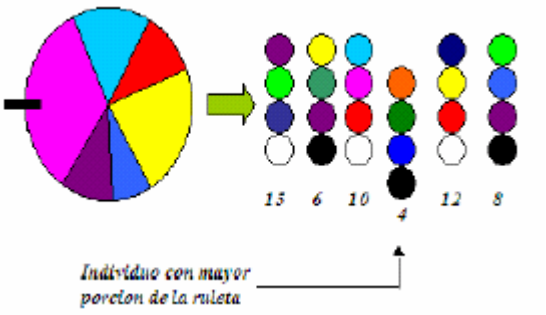
1. Selección
2. Elitismo
3. Cruce
4. Mutación
5. Sustitución

### 2.7.1 Selección.

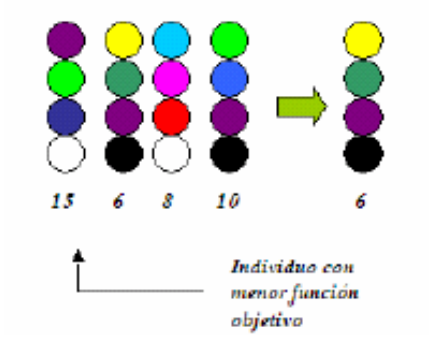
Es un proceso en el cual las cadenas con valores mejores tienen mayores probabilidades de contribuir con uno o más sucesores en la siguiente generación. [10]

La función objetivo es lo que se quiere maximizar o minimizar, también se conoce con el nombre de función de adaptación o "fitness". Por lo que al obtener un mejor valor (mínimo o máximo según sea el caso) para la función objetivo, éste individuo tendrá mayor posibilidad de contribuir con uno o más descendientes en la siguiente generación. En términos biológicos "El individuo más apto tiene mayor probabilidad de sobrevivir y reproducirse". La selección se puede dar de dos formas:

a) Selección tipo ruleta. Para el caso de la ruleta de nuevo a cada individuo se le determina el "fitness", que debe estar entre 0 y 1. A cada miembro se le asigna una porción de una ruleta inversamente proporcional al "fitness" del individuo, es decir, mientras mejor sea su función objetivo mayor será la porción de la ruleta que le corresponde. La ruleta se hace girar N veces, donde N es el número de individuos que faltan para llegar al número de seleccionados deseado. Esto se traduce en que el algoritmo le da un valor a cada función objetivo, inversamente proporcional al mismo, y la suma de todos debe dar 1, luego el algoritmo llama un valor al azar entre 0 y 1 y el que tenga la mayor porción en la ruleta será seleccionado [10].



b) Selección tipo torneo. Se escoge un grupo de individuos aleatoriamente dentro de la población y se escoge al que tenga mejor función objetivo. Los individuos son luego devueltos a la población original y pueden ser seleccionados de nuevo. Se repite la selección hasta obtener el número de individuos seleccionados que se desee [10]




### 2.7.2 Elitismo.

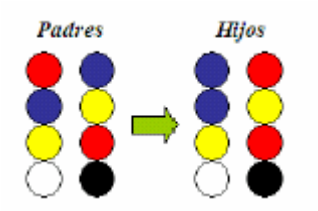
Es una idea introducida por Kenneth de Jong (1975). [10]

Es un método que se combina con los métodos de selección, ya que ayuda a los algoritmos genéticos a alcanzar un mejor desempeño. El elitismo copia el mejor o los mejores cromosomas de la población en la nueva población antes del apareamiento. Este método puede incrementar el desempeño del algoritmo genético, ya que evita que se pierda la mejor solución encontrada.

### 2.7.3 Cruce.

El cruce, consiste en la combinación de los caracteres de dos individuos para generar uno nuevo. La población nueva intercambia material cromosómico y sus descendientes forman la siguiente generación. Es el más importante de los operadores ya que permite el intercambio de información genética entre los individuos. Introduce nuevas combinaciones de genes gracias al intercambio de material entre dos cadenas. El operador de cruce puede llevarse a cabo en varias formas, cruce simple, cruce aritmético, heurística, multipunto, etc. [10].

a) Cruce Simple. El cruce simple se lleva a cabo en dos pasos. Primero los miembros de las cadenas se aparean al azar. Luego, cada par de cadenas es sujeta a los efectos del operador cruce, donde se realiza un intercambio del contenido genético, es decir, el valor de una de las variables es intercambiado [10].




b) Cruce Aritmético. En el cruce aritmético el nuevo individuo tiene carga genética de ambos padres. Es equivalente a realizar una combinación lineal entre las variables del par de vectores o cromosomas de la población a ser cruzados, dando como resultado un nuevo individuo con contribución de ambos padres. Cuando se trabaja con variables binarias se escoge al azar 0 ó 1 [10].

$zi = r.\ xi + (1-r).yi$ (4)

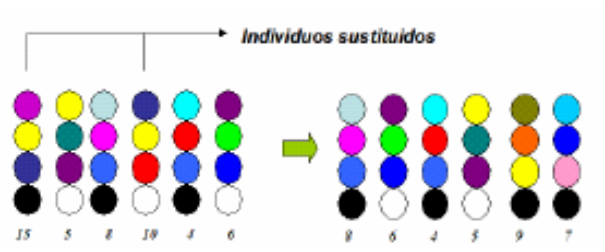

### 2.7.4 Mutación.

Es importante garantizar la diversidad de las soluciones para evitar conversión prematura. Modifica aleatoriamente un gen de un número de individuos llamados al

azar. Es un parámetro que garantiza que la probabilidad de búsqueda dentro de un sub espacio particular dentro del espacio problema nunca sea cero. Introduce un grado de aleatoriedad y evita de esta manera que se generen óptimos locales [10].


### 2.7.5 Sustitución.

Es un operador que simplemente indica que los individuos más aptos son los que sobreviven y que los menos aptos son sustituidos por nuevos individuos (hijos) con mejor función objetivo. Se debe fijar el número de individuos de la población que serán sustituidos por nuevas generaciones. Si el porcentaje de individuos a ser sustituidos es bajo se va a conservar siempre la misma población ocasionando la homogenización de esta. Si el porcentaje a ser sustituidos es alto se garantiza que la población se esté renovando continuamente, y que los peores individuos sean sustituidos. En cada iteración se genera una nueva población. Con esto los individuos con el peor fitness de la generación anterior son sustituidos por los descendientes con mejor fitness [10].



## 2.8 ENFOQUE DEL GRADIENTE DESCENDENTE [1]

Método Empírico:

Un problema multivariable de Programación no lineal, puede ser expresado en la siguiente forma canónica:

$$Max_X \ f(X)$$
$$Donde \ f(X) = f(x_1, x_2, .....x_n)$$

Sujeto a las siguientes restricciones:

$$X \in Q = \{X / g_i(X) \leq 0, i = 1,2,3,...., m \ \}$$

Con bases en el método del gradiente descendente y en el Algoritmo Evolutivo especial planteado en [1] y diseñado para resolver problemas de Programación matemática, específicamente problemas de Programación no Lineal en forma óptima (casi-óptima) con poco esfuerzo computacional; lo que se busca es encontrar una solución óptima, en el peor de los casos casi-óptima arrojada por el Algoritmo Evolutivo y compararla con la solución encontrada utilizando el lagrangiano o cualquier método tradicional, para así verificar la eficiencia del programa implantado computacionalmente.

Si tenemos una función objetivo, sujeta a ciertas restricciones, debemos hacer que el problema se convierta en una función no restringida, es decir introducimos las restricciones en la función objetivo, utilizando el multiplicador de Lagrange, de esta manera obtenemos la función de penalización, la cual decidirá en su debido momento cual individuo es apto para formar parte de la nueva generación. Si el multiplicador de Lagrange tiende a cero o a infinito, el punto de iteración tiende al óptimo o casi óptimo.

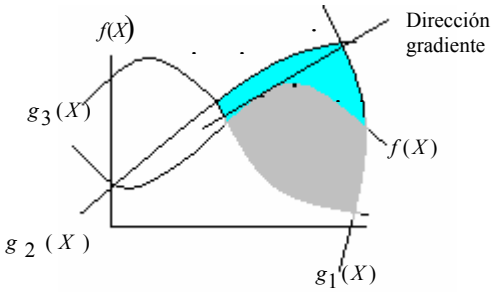
En este enfoque se propone un algoritmo evolutivo especial combinado con una función de penalización para resolver problemas de NLP, el cual funcionará así:

primero producimos una población inicial de manera aleatoria con NP individuos, cada individuo es seleccionado para reproducir hijos por mutación a lo largo de la dirección del gradiente pesado. Si $x \notin Q$ y tiene un valor de aptitud malo, tendrá un efecto en la función objetivo malo, por lo que este individuo debe ser eliminado, por otra parte si tenemos un individuo $x \in Q$ cuyo valor en la función de penalización es bajo, también será eliminado. Solo los individuos $x \in Q$ con un alto valor en la función objetivo, son los que van a procrear por mutación las nuevas generaciones, para así encontrar el óptimo o casi óptimo.

Para un individuo $x$, si $x \in Q$ y nos movemos en la dirección del gradiente $\nabla f(x)$, la función objetivo mejorará; si $x \notin Q$ entonces ésta no pertenece al dominio factible.

Para resolver este tipo de problemas utilizando el método del gradiente descendente, creamos un artificio con la finalidad de que se satisfagan las restricciones $g_i(x) \leq 0$ para i=1,2,.....,m ; esto es, llamamos $I^+ = \{i / g_i(x) > 0, x \in E_n \}$, a la región infactible (aquella región cuyos puntos están fuera del dominio factible o no satisfacen las restricciones ); entonces para un $i \in I^+$, si x se mueve a lo largo de la dirección negativa del gradiente pesado $\nabla g_i(x)$, se satisfacerá que $g_i(x) \leq 0$.

Gráficamente:




Tenemos una función $f(X)$, sujeta a tres restricciones $g_1(X), g_2(X)$ y $g_3(X)$, el área encerrada por las restricciones (área gris y azul), corresponde al dominio factible; todos los puntos que se encuentran fuera de esta, conforman la región infactible $I^+ = \{i / g_i(x) > 0, x \in E_n \}$, si nos movemos en la dirección negativa del gradiente encontraremos el valor óptimo o casi óptimo a problemas de optimización.

Basado en esto la dirección del gradiente pesado denotado por $d(X)$ viene dada por:

$d(X) = \nabla f(X) - \sum \omega_i \nabla g_i(X)$ donde $\omega_i$ es el peso de la dirección del gradiente, calculado como:

$$\omega_i = \begin{cases} 0 & \text{si } g_i(X) \le 0 \\ \delta_i & \text{si } g_i(X) > 0 \end{cases}$$

Así $\delta_i = \dfrac{1}{g_{\max} - g_i(X) + \delta}$ donde $g_{\max}(X) = \max\{g_i(X), i = 1,2,.....m\}$ y $\delta$ es un número positivo muy pequeño.

Entonces $x_i^{(k+1)}$ es generado de $x_j^{(k)}$ por mutación a lo largo de la dirección del gradiente ponderado $d(X)$ de loa siguiente manera:

$x_i^{(k+1)} = x_i^{(k)} + \beta^{(k)} d(x_i^{(k)})$ donde $\beta^{(k)}$ es un número de Erlang generado aleatoriamente

La función objetivo es calculada como:

$$f(X) = \begin{cases} f(X) & g_{\max}(X) \le 0 \\ \dfrac{f(X)}{M + g_{\max}(X)} & g_{\max}(X) > 0 \end{cases}$$

**Análisis de Convergencia:**

Para probar que el Algoritmo Evolutivo especial, planteado en [1] converge; a continuación se definen los siguientes casos:

**Caso 1:**

$x^*$ ($x^* = x_1, x_2, x_3, .....x_n$) es el valor encontrado al aplicar cualquier método analítico de optimización, este se llamará óptimo si se cumple que:

La dirección del gradiente evaluada en el punto (individuo) $x^*$ es igual al gradiente de la función de penalización ($d(x^*) = \nabla f(x^*)$), y esta a su vez es igual a cero $\nabla f(x^*)$=0, se satisfacerá que las restricciones $g_i(X) \leq 0, \text{i} = 1,2,....., \text{m}$, por lo tanto el valor encontrado es el óptimo.

**Caso 2:**

$x^*$ es igual al valor encontrado.

Si $\nabla f(x^*) - \sum_{i=1}^{m} \lambda_i{}^* \nabla g_i(x^*) = 0$, los $\lambda_i \neq 0$ ($\lambda_i \geq 0$) y la $d(x^*) \neq 0$; implica que $x^*$ está en el borde de $\text{E}_\text{n}$, por lo que se puede concluir que $x^*$ coincide con el óptimo o está en una vecindad del óptimo.

# *CAPÍTULO III. MARCO METODOLÓGICO*

## 3.1 INTRODUCCIÓN

En este Capítulo serán desarrollados de manera analítica, los diferentes problemas de tipo no lineal, que haciendo uso de diversas técnicas clásicas, como lo son: método de Lagrange, Khun-Tucker, Hancock, etc, arrojan la solución óptima que en su debido momento será comparada con la solución arrojada por el Algoritmo Evolutivo implantado computacionalmente en MATLAB para así verificar la eficiencia del mismo, además se realizará el diseño e implantación computacional de este.

MATLAB es un entorno de computación y desarrollo de aplicaciones totalmente integrado orientado para llevar a cabo proyectos en donde se encuentren implicados elevados cálculos matemáticos y la visualización gráfica de los mismos. Este software integra análisis numérico, cálculo matricial, proceso de señal y visualización gráfica en un entorno completo donde los problemas y sus soluciones son expresados del mismo modo en que se escribirían tradicionalmente, sin necesidad de hacer uso de la programación tradicional.

Dentro de los usos tópicos de Matlab se encuentra: Cálculo numérico; Desarrollo de Algoritmos; Modelado, simulación y desarrollo de prototipos; Análisis y visualización de datos; Construcción de gráficas; Desarrollo de aplicaciones en distintas áreas científicas y tecnológicas, etc.

La herramienta antes mencionada versión 7.0, será utilizada para obtener la solución óptima a problemas de programación no lineal, basados en Algoritmos Evolutivos.

## 3.2 ALGORITMO O PASOS PARA LA CONSTRUCCIÓN DEL A.E

Para resolver los problemas de programación no lineal antes citados, de una manera más rápida y eficiente aplicando el Algoritmo Evolutivo sugerido en el trabajo [1], es necesario seguir los siguientes pasos:

0.- Especificar la estructura del cromosoma.

1.- Generar la población inicial de forma aleatoria.

2.- Especificar la función de evaluación y la función de aptitud.

3.- Evaluar aptitud de los individuos.

4.- Aplicar operadores (mutación).

5.- Hacer selección.

6.- Chequear el tamaño de la población.

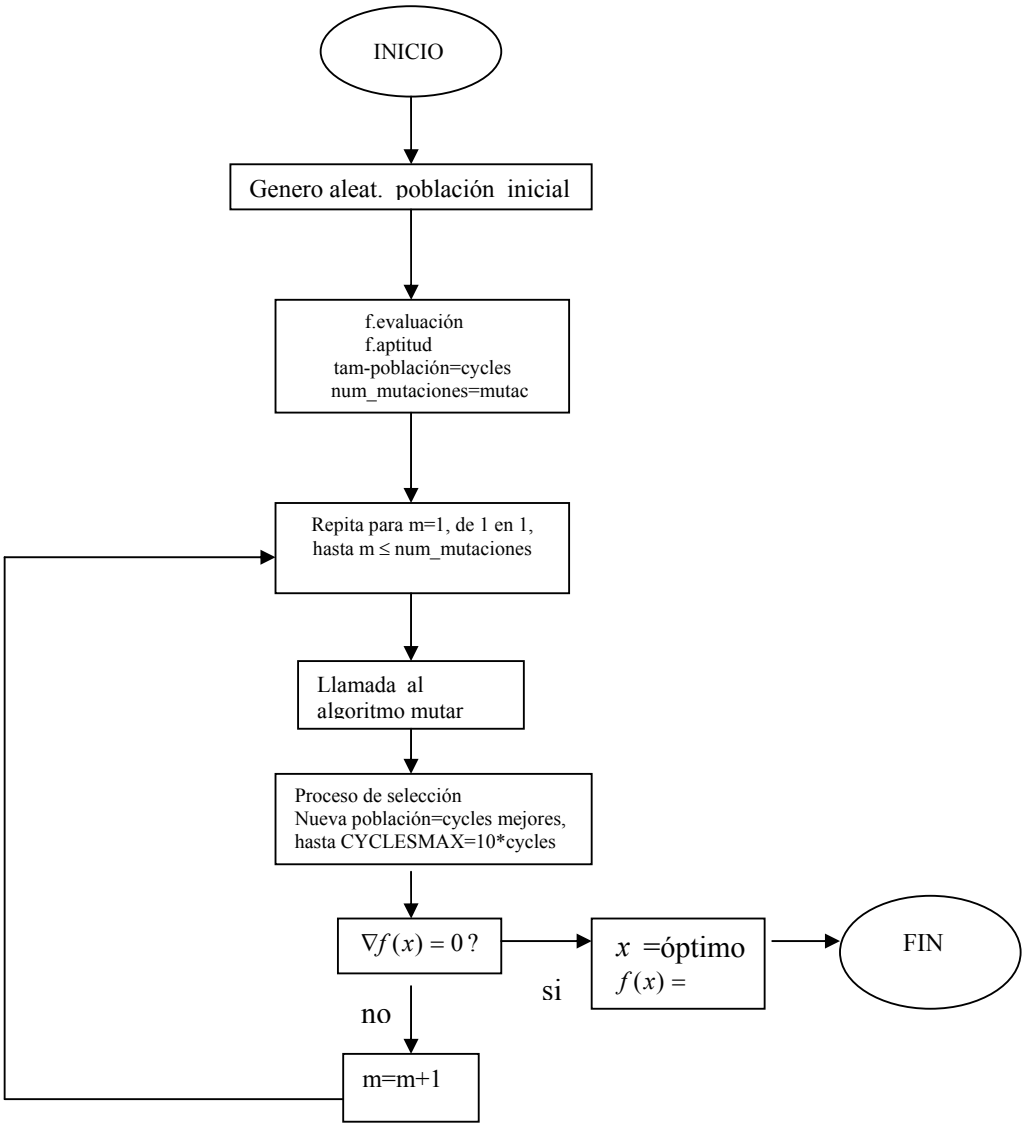
7.- Establecer criterio de parada.

## 3.3 DIAGRAMA DE FLUJO

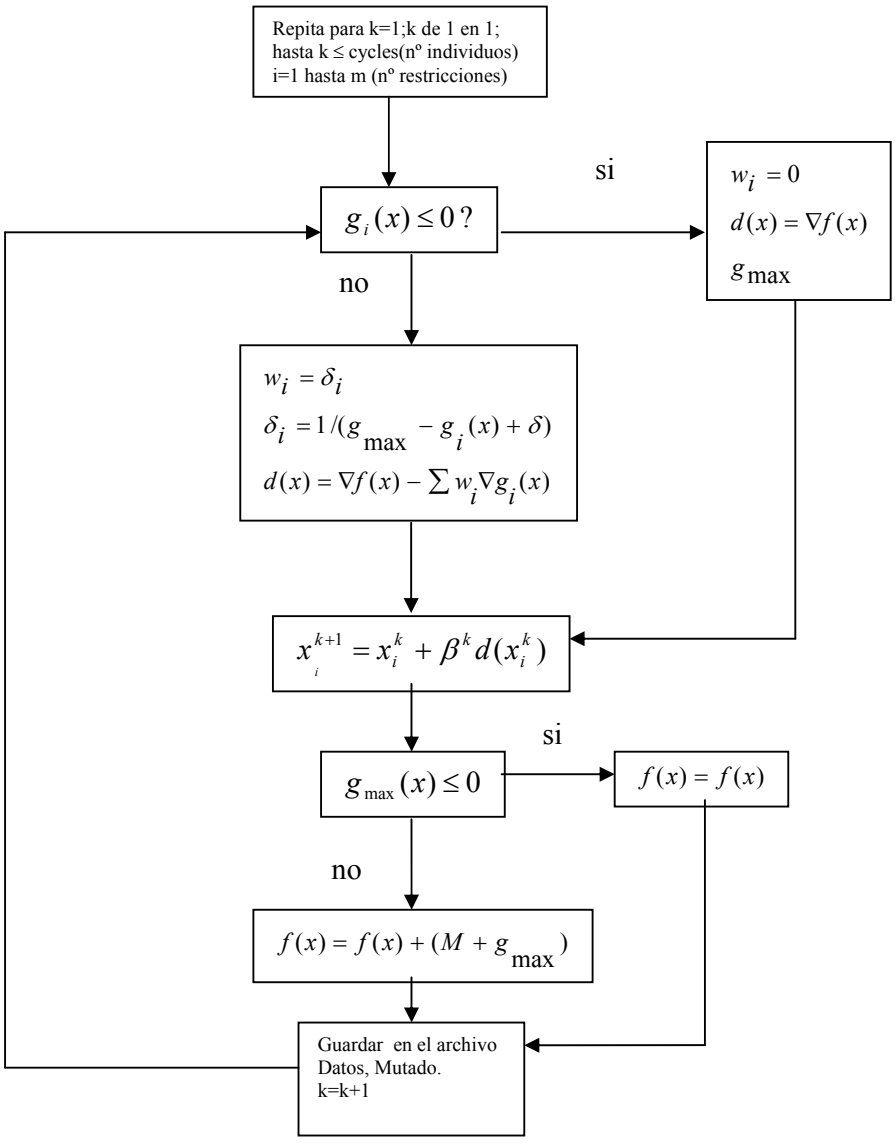
A Continuación se presentará el Diagrama de Flujo, diseñado con bases en Enfoque propuesto en [1] y en el uso de los operadores genético selección y mutación.

### 3.3.1 DIAGRAMA PRINCIPAL

El siguiente diagrama muestra el funcionamiento del Algoritmo Evolutivo, planteado en [1], las estructuras utilizadas, los parámetros requeridos por este y los criterios de parada necesarios para llevar a cabo el proceso.



### 3.3.2 Algoritmo Mutar



## 3.4 DISEÑO DEL ALGORITMO EVOLUTIVO

### 3.4.1 Población de individuos:

Una colección de individuos con características genéticas comunes constituye una población. Las características genéticas de los individuos pueden ser clasificadas en dos tipos: fenotípicas y genotípicas.

El fenotipo de un individuo corresponde a su valor, en el dominio donde es definida una función objetivo, asociado a las variables de decisión del problema. Por ejemplo, hacer referencia al valor de la presión o la temperatura en un reactor, o al peso o volumen de un cuerpo es referirse al fenotipo de esas variables.

El genotipo, corresponde a una representación del fenotipo, que es manipulada por el Algoritmo Evolutivo; así, el fenotipo es codificado en un genotipo y puede ser representado en cualquier estructura de datos apropiada para el problema a ser resuelto, como matrices, árboles o incluso lista de reglas.

Haciendo referencia al problema citado en la sección 3.5.1, la población inicial es generada aleatoriamente y almacenada en una matriz que posee k filas y 6 columnas, correspondientes al número de individuos, a las variables que intervienen en dicho problema y a al valor de penalización respectivamente.

La matriz A que almacena la población inicial tiene la siguiente estructura:

| | | | | | | |
|---|---|---|---|---|---|---|
| Individuo 1 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p_1$ |
| Individuo 2 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p_2$ |
| Individuo 3 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p_3$ |
| | ….. | …… | …… | …… | …… | ….. |
| | …… | ….. | …… | …… | …… | ….. |
| Individuo k → | …… | …… | …… | …… | …… | …… |

### 3.4.2 Función Objetivo y Función de Aptitud:

En el planteamiento de problemas de optimización o búsqueda es necesario definir funciones objetivos, que permitan evaluar a los individuos de una población asignándoles determinado costo para diferenciarlos entre sí.

La aptitud de cada individuo se deriva de su costo, tomando en cuenta toda la población.

El costo asignado por la función objetivo a un individuo es intrínseco al problema y así, no puede ser cambiado a voluntad y asume sus valores dentro de cualquier conjunto ordenado. Por otro lado, el mapeo entre costo y aptitud está relacionado con el proceso de selección de individuos; de modo que constituye una parte ajustable de la estrategia de optimización y corresponde a un mapa monótono sobre el conjunto de los reales positivos.

Los dos tipos de estrategias comúnmente usadas para la asignación de aptitudes son las siguientes:

- El escalamiento: donde un valor positivo de aptitud es asignado por medio de una función, (posiblemente no lineal), de los valores de costo, que toma en cuenta el otorgamiento de una ventaja controlada a los mejores individuos de la población. Este enfoque es el más tradicional.
- El ranqueo: donde la información de escala es descartada ya que la asignación de aptitud a cada individuo se realiza de acuerdo a la posición o rango del mismo dentro de la población.

En el enfoque de escalamiento se pueden presentar los siguientes inconvenientes:

Alteración significativa de la aptitud relativa asignada a cada individuo. Esto se debe a que el cálculo de la aptitud bruta se realiza por medio de una función monótona del costo, desviada en cierta cantidad y luego linealmente escalada.

Utilizando la estrategia definida anteriormente como el escalamiento, en el problema de interés, el valor de la aptitud de cada individuo, es calculado a través de la función de penalización, formada por la función Objetivo y las restricciones, estas ultimas acompañadas de sus respectivos multiplicadores de Lagrange, es decir:

$$h(X,\underline{\lambda}) = f(X) + \sum_{i=1}^{m} \lambda_i g_i(x)$$

Para el problema citado en la sección 3.5.1, las funciones: objetivo y de penalización (aptitud) se escriben como:

Función Objetivo

$$Max\ B(x_1,x_2,x_3) = \left[(28-x_1)x_1 - x_1^{\,2}\right] + (1.04)^{-1}\left[(30-x_2)x_2 - 1.5x_2^{\,2}\right] + \\ + (1.04)^{-2}\left[(33-x_3)x_3 - 2x_3^{\,2}\right] - 50$$

Función de Penalización $h(X,\underline{\lambda})$:

$$Max\ h(x_1,x_2,x_3,\lambda_1,\lambda_2) = \left[(28-x_1)x_1 - x_1^{\,2}\right] + (1.04)^{-1}\left[(30-x_2)x_2 - 1.5x_2^{\,2}\right] + \\ + (1.04)^{-2}\left[(33-x_3)x_3 - 2x_3^{\,2}\right] - 50 + \lambda_1\left[x_1 + x_2 + x_3 - 17\right] + \\ + \lambda_2\left[x_1^{\,2} + 1.5x_2^{\,2} + 2x_3^{\,2} - 150\right]$$

### 3.4.3 Proceso de Selección

La escogencia de individuos, tomando en cuenta sus valores de aptitud, para que participen en la producción de descendientes, constituye el proceso de selección.

Los métodos para seleccionar los individuos son:

Selección por ruleta

Selección por torneo

En el Algoritmo Evolutivo implantado el proceso de selección se realizó por torneo de la siguiente manera: Si el valor de la aptitud para el individuo "k" perteneciente a la población inicial es positiva y si al aplicar operadores de mutación, los nuevos individuos poseen un valor de aptitud también positivo, estos serán los posibles candidatos para formar la nueva población y solo aquellos que posean el mejor valor de aptitud serán los seleccionados, debido a que debemos mantener cierto tamaño de población para no incurrir en un crecimiento descontrolado de la misma.

En síntesis del valor arrojado al evaluar a cada individuo en la función $h(X, \underline{\lambda})$, dependerá su participación en la nueva población, es decir en la próxima mutación.

### 3.4.4 Operadores de cruce y mutación:

La repetición del proceso de selección sobre individuos de una misma población trae como consecuencia la repetición del mejor individuo que ésta posee.

Para ocasionar la ocurrencia de mejoras en la población es necesaria la introducción de operadores genéticos, aplicados entre los pasos de selección.

Existen dos tipos esenciales de operadores genéticos capaces de producir modificaciones genotípicas en los individuos de una población. Estos operadores son:

- Cruce, que causa un intercambio de la información genética de individuos tomados en pares o en grupos. De esta manera, las características genéticas de los individuos "padres" puede ser heredadas por los descendientes "hijos".
  En Algoritmo Evolutivo de interés no existe recombinación o cruce.

- Mutación: este operador conlleva al cambio de genotipos en cada individuo. En las mutaciones solo una pequeña parte del cromosoma es cambiado. La mutación de un bit es comúnmente implantada cambiando algún bit del cromosoma aleatoriamente, según se ilustra en la figura, de esta manera ocurre el proceso de mutación en un Algoritmo genético:

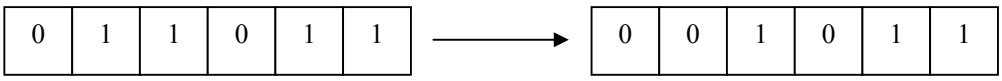



Figura: Mutación de un bit

En el caso del Algoritmo Evolutivo que se está implantando el proceso de mutación ocurre en términos reales, moviéndonos en la dirección del gradiente descendente ponderado y cuya función u operador de interés es:

$$x_i^{(k+1)} = x_i^{(k)} + \beta^{(k)} d(x_i^{(k)})$$

### 3.4.5 Reinserción o sustitución:

Existen varias maneras para reinsertar un descendiente, (hijo), dentro de la población. En el esquema de reemplazo generacional, que es tomado de algunas especies naturales, toda la población de padres es incondicionalmente reemplazada por sus hijos. Una ligera variación de este esquema es la de permitir que algunos hijos puedan competir con algunos padres.

En el esquema incremental, un número pequeño de hijos, (uno o dos), producidos por cruce y mutación son evaluados y posiblemente reinsertados en la población en reemplazo de sus padres, o sustituyéndolos por los padres más viejos o los menos aptos de la población o simplemente cambiándolos por padres seleccionados aleatoriamente de la población.

La reinserción puede ocurrir en forma incondicional, condicionada a que los hijos sean más aptos que los individuos a ser reemplazados, dependiendo de las aptitudes de hijos y padres.

La reinserción tiene el mismo efecto que la selección cuando se niega a algunos individuos la posibilidad de continuar reproduciéndose.

### 3.5 PROBLEMAS

Los cuatro problemas que se presentan a continuación, fueron desarrollados de manera analítica y empírica, con la finalidad de obtener una solución óptima, ésta nos asegurará que el Algoritmo Evolutivo implantado, el cual hace uso de la técnica del gradiente descendente, arroja los resultados esperados.

**3.5.1.-** Una compañía petrolífera acaba de adquirir por 50 millones (unidades monetarias) una concesión para explotar un campo petrolífero con unas reservas de 17 millones de barriles. La empresa tiene que determinar cuantos barriles de petróleo va a extraer del pozo en los tres próximos años (periodo de la concesión para maximizar los beneficios) [2].

Conoce que si extrae $x_1$ millones de barriles durante el primer año, puede vender cada uno de ellos por (28 - $x_1$) unidades monetarias, siendo el coste de extracción de $x_1^2$ millones (unidades monetarias). Durante el segundo año, si extrae $x_2$ millones de barriles, el precio de venta será de (30 - $x_2$) unidades monetarias, por barril y los costes de extracción serán de 1.5 $x_2^2$. En el tercer año si se extraen $x_3$ millones de barriles, cada uno de ellos se puede vender a (33 - $x_3$) unidades monetarias el barril, siendo en este caso los costes de extracción de $2x_3^2$.

La empresa, por otra parte, conoce que a lo largo de los tres años puede extraer como máximo el volumen de las reservas conocidas, es decir, un total de 17 millones de barriles y gastar, con independencia del año, 150 millones (unidades monetarias) en las tareas de extracción.

a) Supuesto que el tipo de interés anual sea del 4 por ciento, determinar la política óptima de extracción de los próximos tres años.

Tomando en cuenta que las cantidades 28,30,33,1.5 y 2 están expresadas en millones de unidades monetarias; así mismo las variables $x_1, x_2, x_3$ son internamente multiplicadas por un elemento que transforma los millones de barriles en unidades monetarias.

Planteamiento

$$Max\ B(x_1, x_2, x_3) = \left[(28 - x_1)x_1 - x_1^{\,2}\right] + (1.04)^{-1}\left[(30 - x_2)x_2 - 1.5x_2^{\,2}\right] + \\ + (1.04)^{-2}\left[(33 - x_3)x_3 - 2x_3^{\,2}\right] - 50$$

$$s.a\begin{cases} x_1 + x_2 + x_3 \le 17 \\ x_1^{\,2} + 1.5x_2^{\,2} + 2x_3^{\,2} \le 150 \\ x_1 \ge 0; x_2 \ge 0; x_3 \ge 0 \end{cases}$$

**La Solución analítica**, a dicho problema de optimización usando condiciones de Khun-Tucker, viene dada de la siguiente manera:

La Función Lagrangiana es:

$$Max\ L(x_1, x_2, x_3, \lambda_1, \lambda_2) = \left[(28 - x_1)x_1 - x_1^{\,2}\right] + (1.04)^{-1}\left[(30 - x_2)x_2 - 1.5x_2^{\,2}\right] + \\ + (1.04)^{-2}\left[(33 - x_3)x_3 - 2x_3^{\,2}\right] - 50 + \lambda_1\left[x_1 + x_2 + x_3 - 17\right] + \\ + \lambda_2\left[x_1^{\,2} + 1.5x_2^{\,2} + 2x_3^{\,2} - 150\right]$$

Las condiciones de Khun Tucker son:

Con relación a $x_1$:

$$\frac{\partial L}{\partial x_1} = 28 - 4x_1 + \lambda_1 + 2\lambda_2 x_1 \le 0$$
$$x_1 \frac{\partial L}{\partial x_1} = x_1\left[28 - 4x_1 + \lambda_1 + 2\lambda_2 x_1\right] = 0$$
$$x_1 \ge 0$$

Con relación a $x_2$:

$$\frac{\partial L}{\partial x_2} = (1.04)^{-1}(30 - 5x_2) + \lambda_1 + 3\lambda_2 x_2 \le 0$$
$$x_2 \frac{\partial L}{\partial x_2} = x_2\left[(1.04)^{-1}(30 - 5x_2) + \lambda_1 + 3\lambda_2 x_2\right] = 0$$
$$x_2 \ge 0$$

Con relación a $x_3$:

$$\frac{\partial L}{\partial x_3} = (1.04)^{-2}(33 - 6x_3) + \lambda_1 + 4\lambda_2 x_3 \leq 0$$
$$x_3 \frac{\partial L}{\partial x_3} = x_3\left[(1.04)^{-2}(33 - 6x_3) + \lambda_1 + 4\lambda_2 x_3\right] = 0$$
$$x_3 \geq 0$$

Con relación a $\lambda_1$:

$$\frac{\partial L}{\partial \lambda_1} = x_1 + x_2 + x_3 - 17 \leq 0$$
$$\lambda_1 \frac{\partial L}{\partial \lambda_1} = \lambda_1\left[x_1 + x_2 + x_3 - 17\right] = 0$$
$$\lambda_1 \geq 0$$

Con relación a $\lambda_2$:

$$\frac{\partial L}{\partial \lambda_2} = x_1^{\,2} + 1.5x_2^{\,2} + 2x_3^{\,2} - 150 \leq 0$$
$$\lambda_2 \frac{\partial L}{\partial \lambda_2} = \lambda_2\left[x_1^{\,2} + 1.5x_2^{\,2} + 2x_3^{\,2} - 150\right] = 0$$
$$\lambda_2 \geq 0$$

La matriz Hessiana será:

$$H = \begin{pmatrix} -4 & 0 & 0 \\ 0 & -(1.04)^{-1}5 & 0 \\ 0 & 0 & -(1.04)^{-2}6 \end{pmatrix}$$

Al resolver el problema aplicando el método de Khun-Tucker los valores óptimos encontrados fueron:

x1*= 6.412

x2*=5.511

x3*=5.076

f*=216.679

**En la implantación** del Algoritmo Evolutivo realizada para este problema es necesario saber que:

La población inicial es generada de manera aleatoria y almacenada en una matriz llamada "A", con el número de individuos "cycles" (tamaño de la población) que el usuario determina, de igual manera el número de mutaciones "mutac" debe ser especificado.

Los individuos que han sido mutados, se almacenarán en una matriz de igual tamaño que "A" denominada "mutado", Además todas las poblaciones pertenecientes a la población "A" generadas en cada mutación serán almacenadas en un archivo denominado Datos.rtf, de igual manera se guardarán los individuos resultantes de la mutación en un archivo de nombre mutados.rtf.

Si se cumple que $g_i(x) \leq 0$ para i=1,2,….., m; donde m es el número de restricciones que posee el problema, la dirección del gradiente se calcula como: $d(X) = \nabla f(X)$ ya que el peso $w_i = 0$ para i=1,2.

$g_1(x) = x_1 + x_2 + x_3 - 17$

$g_2(x) = x_1^2 + 1.5x_2^2 + 2x_3^2 - 150$

$$\nabla f(X) = \begin{bmatrix} 28 - 4x_1 + \lambda_1 + 2\lambda_2 x_1 \\ (1.04)^{-1}(30 - 5x_2) + \lambda_1 + 3\lambda_2 x_2 \\ (1.04)^{-2}(33 - 6x_3) + \lambda_1 + 4\lambda_2 x_3 \\ x_1 + x_2 + x_3 - 17 \\ x_1^2 + 1.5x_2^2 + 2x_3^2 - 150 \end{bmatrix}$$

En caso de que $g_i(x) > 0$ la dirección del gradiente viene dada por:

$$d(X) = \nabla f(X) - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

$$d(X) = \begin{bmatrix} 28 - 4x_1 + \lambda_1 + 2\lambda_2 x_1 \\ (1.04)^{-1}(30 - 5x_2) + \lambda_1 + 3\lambda_2 x_2 \\ (1.04)^{-2}(33 - 6x_3) + \lambda_1 + 4\lambda_2 x_3 \\ x_1 + x_2 + x_3 - 17 \\ x_1^2 + 1.5x_2^2 + 2x_3^2 - 150 \end{bmatrix} - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

Por lo que la dirección del gradiente ponderado para cada una de las variables pertenecientes a cada individuo se calcula como:

$$d(x_1) = 28 - 4x_1 + \lambda_1 + 2\lambda_2 x_1 - w_1(1) - w_2(2x_1)$$
$$d(x_2) = (1.04)^{-1}(30 - 5x_2) + \lambda_1 + 3\lambda_2 x_2 - w_1(1) - w_2(3x_2)$$
$$d(x_3) = (1.04)^{-2}(33 - 6x_3) + \lambda_1 + 4\lambda_2 x_3 - w_1(1) - w_2(4x_3)$$
$$d(\lambda_1) = x_1 + x_2 + x_3 - 17$$
$$d(\lambda_2) = x_1^2 + 1.5x_2^2 + 2x_3^2 - 150$$

Encuentro el valor máximo $g_{\max}$ entre $g_1(x)$ y $g_2(x)$, para cada individuo de la población inicial y mutada; si este valor es menor o igual a cero $f(x) = f(x)$, este es el valor de aptitud del individuo, por otro lado si el valor de $g_{\max}$ es mayor que cero, este individuo es penalizado y su valor de aptitud viene dado por $f(x) = \dfrac{f(x)}{M + g_{\max}}$

Luego de la evaluación de los individuos de la matriz inicial "A" en la función de penalización; sus valores de aptitud serán almacenados en la última columna de la matriz "A", de igual manera ocurre con los individuos mutados en la matriz denominada "mutado"

Seguidamente a la última columna de "A" y de "mutado" donde se encuentran almacenados los valores de penalización para cada individuo, se le aplicará un método de ordenamiento conocido: método de la burbuja, para tomar de allí los mejores individuos; en este caso aquellos con valor de aptitud positivo, para que formen parte de la nueva matriz "A"(de igual o diferente tamaño a la matriz "A" inicial) y generen por medio de mutación nuevos individuos para así encontrar el punto óptimo o casi óptimo.

Como el comportamiento del crecimiento de la población ocurre de manera exponencial, se colocó un límite denominado CYCLESMAX, el cual es 10 veces el tamaño especificado como cycles al inicio del programa, si el algoritmo en algún momento sobrepasa el CYCLESMAX se tomarán aquellos individuos con mejor valor de aptitud para generar nuevo individuos, es decir el tamaño final de la población es especificado por esta variable la cual el usuario especifica.

Estructuras de manera gráfica:

Matriz "A"

| $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p$ |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |



Matriz "mutado"

| $x_{1m}$ | $x_{2m}$ | $x_{3m}$ | $\lambda_{1m}$ | $\lambda_{2m}$ | $p_m$ |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |

Seleccionando los individuos con mejor valor de aptitud, obtengo la nueva población:

| $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p$ |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

CYCLESMAX →

Matriz "A"

Haciendo uso de CYCLESMAX, se tiene un control en el crecimiento de la población y los individuos más aptos de cada generación serán los que intervendrán en el proceso.

A continuación serán mostrados de manera computacional, el Algoritmo Evolutivo correspondiente a este problema, parte de la población inicial y final generada al momento de la ejecución del Algoritmo Evolutivo, la gráfica que representa el valor aptitud vs. el valor óptimo de cada una de las variables.

De igual manera se mostrarán los parámetros especificados anteriormente para cada uno de los problemas de optimización considerados en este trabajo.

#### 3.5.1.1 Algoritmo Evolutivo

Esto es solo parte de la población inicial generada de manera aleatoria para el programa de la empresa petrolera

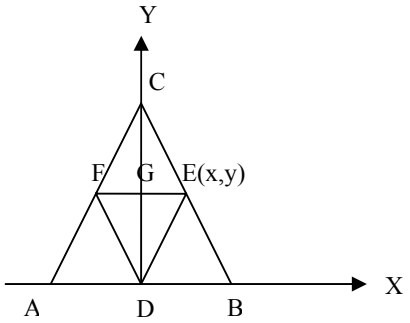
| | | | | | |
|---|---|---|---|---|---|
| 2.3060 | 0.9502 | 4.3877 | 1.2607 | 2.3164 | 0.0000 |
| 1.9248 | 1.1998 | 0.4599 | 1.9238 | 2.6970 | 0.0000 |
| 0.3046 | 1.9411 | 3.3822 | 1.9641 | 2.2324 | 0.0000 |
| 1.0234 | 2.7891 | 2.1736 | 0.0758 | 2.2231 | 0.0000 |
| 1.0726 | 4.2246 | 3.4410 | 1.7486 | 1.9071 | 0.0000 |
| 0.7715 | 2.3684 | 4.3166 | 1.6981 | 3.1171 | 0.0000 |
| 4.5307 | 3.3396 | 3.1924 | 0.1711 | 4.3109 | 0.0000 |
| 2.8731 | 0.5286 | 4.8552 | 2.2073 | 1.5144 | 0.0000 |
| 4.6294 | 2.7882 | 4.6687 | 1.6960 | 2.5733 | 0.0000 |
| 0.7220 | 2.5470 | 4.4132 | 1.2112 | 0.4248 | 0.0000 |
| 3.2331 | 4.7594 | 4.4065 | 4.9278 | 3.8009 | 0.0000 |
| 4.5645 | 3.5081 | 0.9384 | 0.2064 | 3.5094 | 0.0000 |
| 3.5407 | 0.9247 | 4.8933 | 4.1718 | 3.9597 | 0.0000 |
| 4.9564 | 4.2169 | 2.3454 | 0.9972 | 3.5198 | 0.0000 |
| 4.6047 | 2.2034 | 2.1712 | 0.9349 | 0.1067 | 0.0000 |
| 0.1970 | 2.1742 | 4.6659 | 0.3201 | 1.0943 | 0.0000 |
| 2.4019 | 4.8363 | 2.4604 | 4.3674 | 3.0949 | 0.0000 |
| 4.2238 | 3.9041 | 1.4522 | 0.3960 | 4.3725 | 0.0000 |
| 2.8287 | 2.7292 | 1.7537 | 2.0419 | 0.7462 | 0.0000 |
| 2.8504 | 0.3754 | 0.4981 | 3.8252 | 4.7625 | 0.0000 |
| 0.8857 | 3.0433 | 1.0246 | 4.5215 | 4.0770 | 0.0000 |
| 3.1578 | 3.9948 | 4.8457 | 4.9454 | 0.3100 | 0.0000 |
| 3.7616 | 3.1499 | 4.3826 | 0.4002 | 3.2661 | 0.0000 |
| 4.6676 | 1.4839 | 1.3795 | 4.5034 | 2.2178 | 0.0000 |
| 2.7599 | 0.2375 | 1.6068 | 4.7697 | 2.8005 | 0.0000 |
| 3.8462 | 0.4685 | 0.3510 | 0.7686 | 4.2442 | 0.0000 |
| 0.9482 | 3.0470 | 4.1181 | 3.0652 | 2.2458 | 0.0000 |
| 3.2494 | 0.9183 | 2.7172 | 3.0262 | 4.5373 | 0.0000 |
| 0.2970 | 2.3696 | 0.8480 | 2.3709 | 2.7581 | 0.0000 |
| 1.4371 | 1.1311 | 2.7687 | 3.2832 | 1.2383 | 0.0000 |
| 3.4605 | 1.2460 | 2.5445 | 1.8019 | 2.7163 | 0.0000 |
| 3.0691 | 1.4770 | 1.4353 | 3.5891 | 3.4479 | 0.0000 |
| 0.2969 | 0.3631 | 1.6122 | 1.6643 | 0.2909 | 0.0000 |
| 4.2514 | 0.9316 | 1.5521 | 4.3703 | 2.0614 | 0.0000 |
| 3.1589 | 2.7519 | 0.3752 | 4.8116 | 4.4613 | 0.0000 |
| 0.2991 | 4.8194 | 2.4583 | 4.4603 | 1.3158 | 0.0000 |
| 4.9725 | 1.7791 | 3.4519 | 2.7136 | 2.9806 | 0.0000 |
| 2.8260 | 4.9752 | 1.6324 | 1.3920 | 0.2554 | 0.0000 |
| 4.9539 | 3.9239 | 1.7292 | 3.0257 | 2.5417 | 0.0000 |
| 2.7288 | 3.9753 | 4.4168 | 4.4416 | 0.9864 | 0.0000 |
| 2.4290 | 3.0532 | 4.2089 | 0.3953 | 2.3053 | 0.0000 |
| 2.8027 | 4.2591 | 1.4384 | 4.8907 | 0.6672 | 0.0000 |
| 2.8005 | 2.5267 | 2.1722 | 1.3796 | 3.5941 | 0.0000 |
| 0.4107 | 4.7332 | 0.5689 | 3.1364 | 4.6929 | 0.0000 |
| 4.4336 | 2.2595 | 0.0705 | 0.9447 | 1.7399 | 0.0000 |
| 0.0086 | 2.4614 | 3.3753 | 0.9488 | 0.4816 | 0.0000 |
| 0.0015 | 4.0773 | 4.9231 | 4.3332 | 3.6646 | 0.0000 |
| 2.7924 | 0.9140 | 1.0485 | 0.3508 | 4.9156 | 0.0000 |
| 0.7780 | 1.7041 | 3.6220 | 3.5264 | 4.7129 | 0.0000 |
| 2.4671 | 2.7067 | 4.7254 | 1.5431 | 0.5146 | 0.0000 |
| 4.1635 | 1.5704 | 0.5861 | 2.9105 | 3.1883 | 0.0000 |
| 0.2788 | 2.5163 | 3.9894 | 0.4714 | 0.8459 | 0.0000 |
| 3.6530 | 1.9508 | 3.0640 | 2.9984 | 4.8309 | 0.0000 |
| 2.4044 | 0.2353 | 4.7950 | 4.5467 | 2.9846 | 0.0000 |
| 3.2987 | 3.7155 | 1.6797 | 3.1043 | 2.8571 | 0.0000 |
| 0.3255 | 1.3925 | 2.9115 | 3.9815 | 3.4285 | 0.0000 |

| | | | | | |
|---|---|---|---|---|---|
| 0.1418 | 2.4499 | 1.7941 | 1.6231 | 0.8283 | 0.0000 |
| 2.1090 | 0.7280 | 0.2932 | 4.7484 | 2.3308 | 0.0000 |
| 1.7822 | 2.0202 | 3.6446 | 1.2519 | 0.5780 | 0.0000 |
| 4.0326 | 0.5207 | 4.7294 | 2.5129 | 3.0647 | 0.0000 |
| 0.8385 | 2.3307 | 2.1016 | 1.7213 | 3.4479 | 0.0000 |
| 4.8551 | 4.7435 | 4.9693 | 4.0602 | 1.3129 | 0.0000 |
| 4.4129 | 4.6869 | 3.1708 | 1.9825 | 0.4558 | 0.0000 |
| 2.4336 | 1.5088 | 4.1542 | 3.6804 | 2.0835 | 0.0000 |
| 2.8703 | 3.1796 | 1.5158 | 1.9235 | 4.0869 | 0.0000 |
| 3.8582 | 0.5620 | 1.8546 | 0.2227 | 0.2494 | 0.0000 |
| 3.9353 | 2.7786 | 2.2258 | 3.7664 | 4.0634 | 0.0000 |
| 4.0288 | 2.6227 | 3.3909 | 2.0831 | 0.8499 | 0.0000 |
| 1.5890 | 1.2712 | 0.2457 | 2.5818 | 4.2339 | 0.0000 |
| 3.4847 | 1.9525 | 4.3695 | 1.4736 | 3.3374 | 0.0000 |
| 1.4564 | 3.7146 | 1.3070 | 4.5537 | 4.3001 | 0.0000 |
| 1.3837 | 3.8454 | 3.2316 | 4.3285 | 0.6614 | 0.0000 |
| 1.0086 | 0.1455 | 1.8496 | 0.7891 | 2.0826 | 0.0000 |
| 4.5365 | 3.1066 | 2.6613 | 4.0659 | 4.2473 | 0.0000 |
| 3.4414 | 3.4870 | 2.5249 | 2.3972 | 3.5816 | 0.0000 |
| 1.2077 | 0.0330 | 1.3309 | 2.6305 | 4.4494 | 0.0000 |
| 2.2796 | 1.6044 | 2.9209 | 0.5764 | 1.9233 | 0.0000 |
| 2.7891 | 3.1568 | 0.9047 | 0.3668 | 2.6121 | 0.0000 |
| 2.9692 | 2.9127 | 2.9085 | 2.5004 | 2.3710 | 0.0000 |
| 1.2120 | 1.7203 | 1.1450 | 2.1962 | 3.2320 | 0.0000 |
| 1.2247 | 1.6809 | 1.9964 | 2.9960 | 0.1938 | 0.0000 |
| 3.5118 | 4.4670 | 0.9580 | 4.4208 | 3.2859 | 0.0000 |
| 0.8393 | 1.5781 | 3.8778 | 0.1348 | 0.0412 | 0.0000 |
| 2.3002 | 2.0872 | 1.5864 | 2.3134 | 0.0652 | 0.0000 |
| 4.3969 | 3.6407 | 1.8538 | 2.0591 | 0.4544 | 0.0000 |
| 1.1287 | 3.8374 | 0.2113 | 1.7664 | 4.5189 | 0.0000 |
| 0.7453 | 2.9214 | 3.3832 | 1.7627 | 0.1945 | 0.0000 |
| 0.2181 | 4.8980 | 2.7487 | 2.3318 | 0.0821 | 0.0000 |
| 4.1822 | 3.0744 | 1.0575 | 1.9857 | 0.0115 | 0.0000 |
| 0.0171 | 3.0877 | 2.0473 | 2.6426 | 1.5504 | 0.0000 |
| 2.7425 | 1.3648 | 0.1554 | 1.8706 | 0.0358 | 0.0000 |
| 2.5139 | 1.7475 | 2.5698 | 4.8859 | 0.0377 | 0.0000 |
| 1.7065 | 1.0418 | 0.2668 | 3.3179 | 2.9966 | 0.0000 |
| 3.4129 | 4.3351 | 4.1920 | 4.6096 | 1.8548 | 0.0000 |
| 1.4361 | 4.6247 | 4.7729 | 2.7231 | 3.1132 | 0.0000 |
| 4.9904 | 0.1136 | 1.6589 | 4.1002 | 0.5489 | 0.0000 |
| 2.3698 | 3.8285 | 3.5824 | 0.5916 | 2.2196 | 0.0000 |
| 3.9687 | 2.9767 | 4.2577 | 2.8837 | 0.5536 | 0.0000 |
| 4.9494 | 3.2496 | 1.2385 | 3.7518 | 3.5307 | 0.0000 |
| 3.5414 | 3.6998 | 1.6565 | 0.9924 | 4.2818 | 0.0000 |
| 1.0214 | 3.4953 | 3.3855 | 2.5818 | 0.6332 | 0.0000 |
| 0.1287 | 0.9355 | 4.8876 | 4.9108 | 4.7581 | 0.0000 |
| 1.3399 | 1.8756 | 2.5026 | 4.0328 | 1.3409 | 0.0000 |
| 1.5475 | 0.0386 | 3.2142 | 3.9952 | 4.0231 | 0.0000 |
| 4.8322 | 3.8789 | 4.0125 | 3.5098 | 2.3907 | 0.0000 |
| 2.6793 | 3.4190 | 2.5907 | 3.8862 | 4.0232 | 0.0000 |
| 0.6720 | 2.0187 | 2.4371 | 0.6747 | 1.9911 | 0.0000 |
| 2.8960 | 4.8699 | 2.8110 | 3.0352 | 0.3107 | 0.0000 |
| 4.5053 | 4.6140 | 2.3046 | 2.2080 | 2.3132 | 0.0000 |
| 4.7395 | 4.8345 | 0.1430 | 3.7901 | 0.3347 | 0.0000 |
| 4.5171 | 2.8820 | 3.8018 | 1.9614 | 3.8436 | 0.0000 |
| 4.8682 | 3.0630 | 0.5599 | 4.6693 | 1.8107 | 0.0000 |
| 0.3879 | 0.6358 | 2.1446 | 1.4456 | 3.7930 | 0.0000 |
| 3.1579 | 2.7794 | 0.3115 | 0.5198 | 3.5509 | 0.0000 |
| 1.9607 | 4.7149 | 4.2750 | 1.6657 | 2.3787 | 0.0000 |

Después de realizar la mutaciones especificadas por el usuario, el programa implantado para el problema de la empresa petrolera, arroja los siguientes resultados:

```
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
6.0365      5.7157      5.2072      0.8760      0.0037      216.0792
```

**3.5.2** Encontrar el $\Delta DEF$ de área máxima inscrito en el $\Delta ABC$ conocido e isósceles ( $\overline{AC} = \overline{BC}$ ) [17].




**Solución Analítica:**

Considere el sistema de coordenadas con origen en D y $\overline{AB} \subseteq$ eje X., $\overline{CD} \subseteq$ eje Y; el punto E se mueve en $\overline{BC}$; cuando coincide con C el área del triángulo inscrito es cero. E entre $\overline{BC}$ corresponde a un triángulo inscrito de área máxima. Sea f(x,y) el área del $\Delta DEF$ entonces, se pide:

$$Max\ \mathrm{f(x,y)} = 2x * \frac{y}{2} = xy$$

$$sujeta\ a: \frac{x}{c/2} = \frac{h-y}{h}$$

$$x, y \geq 0$$

Por sustitución se obtiene:

Con $c/2 = \overline{DB}$

$h = \overline{CD}$

Sustituyendo x en f(x,y) se tiene: $f(y) = \frac{c}{2}\frac{(h-y)}{h}y$

$$f'(y) = \frac{c}{2h}(h-2y) = 0$$

$$f''(y) = \frac{-c}{h} < 0$$

Como $f'' < 0$ estamos en presencia de un máximo, así los valores óptimos son:

$$y^* = \frac{h}{2}$$

$$x^* = \frac{c}{4}$$

$$f^* = \frac{ch}{8}$$

Utilizando el método de Lagrange:

$$Max\ \mathrm{f(x,y)} = 2x\frac{y}{2} = xy$$

$$sujeta\ a: \frac{x}{c/2} = \frac{h-y}{h}$$

La función Lagrangiana viene dada por:

$$L(x, y, \lambda_1) = xy + \lambda_1\left(\frac{x}{c/2} - \left(\frac{h-y}{h}\right)\right)$$

$$\nabla L(x, y, \lambda_1) = \begin{bmatrix} \frac{\partial L}{\partial x} \\ \frac{\partial L}{\partial y} \\ \frac{\partial L}{\partial \lambda_1} \end{bmatrix} = \begin{bmatrix} y + \frac{2\lambda_1}{c} \\ x + \frac{\lambda_1}{h} \\ \frac{2x}{c} - 1 + \frac{y}{h} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

El Hessiano es calculado como:

$$H = \begin{pmatrix} 0 & 1 & 2/c \\ 1 & 0 & 1/h \\ 2/c & 1/h & 0 \end{pmatrix}$$

Criterio para buscar el óptimo o condiciones suficientes para un óptimo.

Si todos los autovalores de HL son:

-Positivos, estamos en presencia de un mínimo.

-Negativos, estamos en presencia de un máximo.

- Positivos y negativos, punto de silla.

- Cero, sin información.

Como necesitamos un máximo y los autovalores arrojan un mínimo, debe aplicarse el criterio de Hancock:

$$\text{Det}(H)=\begin{bmatrix} 0-z & 1 & 2/c & 0-z & 1 \\ 1 & 0-z & 1/h & 1 & 0-z \\ 2/c & 1/h & 0 & 2/c & 1/h \end{bmatrix}$$

$z = \dfrac{-4}{\dfrac{4h^2 + c^2}{ch^2}} < 0$, este valor es negativo ya que c y h son trazas de trigonometría y son valores positivos.

Por lo tanto los valores óptimos son:

$$y^* = \frac{h}{2}$$

$$x^* = \frac{c}{4}$$

$$f^* = \frac{ch}{8}$$

**Para Encontrar la solución empírica** a este problema es necesario definir:

cycles: nº individuos que posee la población inicial.

mutac: nº de mutaciones que realizará el algoritmo.

La población inicial es generada de manera aleatoria y almacenada en una matriz denominada “A” que posee la siguiente estructura:

| | | | | |
|---|---|---|---|---|
| Individuo 1 → | $x$ | $y$ | $\lambda_1$ | $p_1$ |
| Individuo 2 → | $x$ | $y$ | $\lambda_1$ | $p_2$ |
| Individuo 3 → | $x$ | $y$ | $\lambda_1$ | $p_3$ |
| | ……. | …… | …… | ….. |
| | …… | ……. | …… | ….. |
| Individuo k → | …… | …… | …… | …… |

$$g_1(x) = \frac{x}{c/2} - \left(\frac{h-y}{h}\right)$$

La función Lagrangiana viene dada por:

$$L(x, y, \lambda_1) = xy + \lambda_1\left(\frac{x}{c/2} - \left(\frac{h-y}{h}\right)\right)$$

Si se cumple que $g_1(x) \leq 0$ , $d(X) = \nabla L(X)$

$$\nabla L(X) = \begin{bmatrix} y + \dfrac{2\lambda_1}{c} \\ x + \dfrac{\lambda_1}{h} \\ \dfrac{2x}{c} - 1 + \dfrac{y}{h} \end{bmatrix}$$ donde ( $X$ ) es un vector que posee 3 componentes $(x, y, \lambda_1)$

En caso de que $g_i(x) > 0$ la dirección del gradiente viene dada por:

$$d(X) = \nabla L(X) - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

$$d(X) = \begin{bmatrix} y + \frac{2\lambda_1}{c} \\ x + \frac{\lambda_1}{h} \\ \frac{2x}{c} - 1 + \frac{y}{h} \end{bmatrix} - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

Por lo que la dirección del gradiente ponderado para cada una de las variables pertenecientes a cada individuo se calcula como:

$$d(x_1) = y + \frac{2\lambda_1}{c} - w_1(2/c)$$
$$d(x_2) = x + \frac{\lambda_1}{h} - w_1(1/h)$$
$$d(\lambda_1) = \frac{2x}{c} - 1 + \frac{y}{h}$$

Encuentro el valor máximo $g_{\max}$ (en este caso como hay una sola restricción $g_{\max} = g_1(x)$) para cada individuo de la población inicial y mutada; si este valor es menor o igual a cero $f(x) = f(x)$, este es el valor de aptitud del individuo, por otro lado si el valor de $g_{\max}$ es mayor que cero, este individuo es penalizado y su valor de aptitud viene dado por $f(x) = \frac{f(x)}{M + g_{\max}}$

A este problema se le aplica la Metodología explicada anteriormente, con diferencias en el número de variables, función de penalización, restricciones, valor del gradiente y por lo tanto dirección del mismo.

### 3.5.2.1 Algoritmo Evolutivo

Población inicial correspondiente al problema del triángulo, citado en la sección 3.2.2, para un c=6 y h=8

| | | | |
|---|---|---|---|
| 0.6052 | 3.5827 | 0.9393 | 0.0000 |
| 2.2538 | 2.5566 | 2.4532 | 0.0000 |
| 3.5794 | 3.8820 | 2.0464 | 0.0000 |
| 4.4642 | 2.4467 | 2.3176 | 0.0000 |
| 1.3655 | 0.9295 | 3.0547 | 0.0000 |
| 1.2738 | 3.5032 | 0.3558 | 0.0000 |
| 4.3280 | 4.9135 | 1.5714 | 0.0000 |
| 1.1618 | 4.0332 | 3.0419 | 0.0000 |
| 4.0244 | 3.5178 | 0.8751 | 0.0000 |
| 4.5420 | 2.4248 | 3.1051 | 0.0000 |
| 1.1595 | 0.5731 | 1.2298 | 0.0000 |
| 1.1966 | 3.3243 | 2.9368 | 0.0000 |
| 0.2488 | 1.8269 | 2.5303 | 0.0000 |
| 0.3919 | 0.7002 | 2.3239 | 0.0000 |
| 3.2041 | 2.8339 | 2.7071 | 0.0000 |
| 0.9544 | 4.1150 | 4.7116 | 0.0000 |
| 4.2193 | 3.3697 | 1.7088 | 0.0000 |
| 0.8695 | 4.9972 | 2.0090 | 0.0000 |
| 0.8540 | 4.8082 | 1.5384 | 0.0000 |
| 4.9715 | 0.2943 | 2.0578 | 0.0000 |
| 2.1990 | 1.8016 | 1.4297 | 0.0000 |
| 1.7002 | 2.7426 | 1.9706 | 0.0000 |
| 1.5711 | 1.3088 | 2.5151 | 0.0000 |
| 1.8254 | 2.9867 | 3.6099 | 0.0000 |
| 1.9662 | 0.2464 | 1.5310 | 0.0000 |
| 2.9576 | 2.8553 | 0.5608 | 0.0000 |
| 0.5987 | 3.5043 | 2.2164 | 0.0000 |
| 0.1906 | 4.8114 | 2.3338 | 0.0000 |
| 2.2930 | 3.7526 | 0.0733 | 0.0000 |
| 4.3493 | 3.7000 | 3.3203 | 0.0000 |
| 4.6712 | 2.1594 | 3.6203 | 0.0000 |
| 1.3222 | 3.1713 | 1.4082 | 0.0000 |
| 0.8015 | 4.0151 | 1.3091 | 0.0000 |
| 4.3643 | 0.4194 | 3.5424 | 0.0000 |
| 1.1894 | 4.7273 | 3.9193 | 0.0000 |
| 3.2292 | 4.5797 | 4.9308 | 0.0000 |
| 4.8344 | 3.0099 | 2.3667 | 0.0000 |
| 3.3247 | 1.2678 | 4.5141 | 0.0000 |
| 4.3519 | 4.3673 | 2.2553 | 0.0000 |
| 0.0496 | 2.5670 | 4.0226 | 0.0000 |
| 0.6850 | 3.6633 | 4.1443 | 0.0000 |
| 4.0938 | 2.1111 | 0.8314 | 0.0000 |
| 2.1508 | 4.8068 | 1.9695 | 0.0000 |
| 4.4516 | 0.3603 | 2.6038 | 0.0000 |
| 3.6745 | 2.7670 | 3.5906 | 0.0000 |
| 3.4366 | 1.4599 | 2.8459 | 0.0000 |
| 1.7306 | 4.2898 | 2.3040 | 0.0000 |
| 0.8302 | 1.6788 | 2.2265 | 0.0000 |
| 0.7781 | 3.4010 | 0.4387 | 0.0000 |
| 0.9556 | 0.2672 | 2.2174 | 0.0000 |
| 2.1123 | 1.7833 | 1.8315 | 0.0000 |
| 4.2799 | 2.4915 | 1.5127 | 0.0000 |
| 2.4512 | 2.1722 | 4.2592 | 0.0000 |
| 4.0797 | 2.8123 | 3.7974 | 0.0000 |
| 2.3038 | 3.0831 | 4.7488 | 0.0000 |
| 2.2868 | 0.5667 | 2.7897 | 0.0000 |

| | | | |
|---|---|---|---|
| 2.2534 | 4.4913 | 0.0712 | 0.0000 |
| 2.0611 | 3.7728 | 2.9809 | 0.0000 |
| 4.5080 | 3.9556 | 4.0810 | 0.0000 |
| 0.0279 | 4.0748 | 4.8855 | 0.0000 |
| 1.4870 | 3.3500 | 1.1095 | 0.0000 |
| 0.2458 | 1.0044 | 3.5184 | 0.0000 |
| 3.4659 | 1.3654 | 2.6103 | 0.0000 |
| 3.2505 | 3.1312 | 4.6645 | 0.0000 |
| 4.9149 | 2.6843 | 3.5668 | 0.0000 |
| 2.7634 | 0.2975 | 1.1402 | 0.0000 |
| 2.0004 | 0.4448 | 2.2482 | 0.0000 |
| 0.9939 | 1.3565 | 0.8610 | 0.0000 |
| 3.1260 | 2.0454 | 4.8441 | 0.0000 |
| 3.6668 | 2.3702 | 1.7786 | 0.0000 |
| 1.8794 | 4.5449 | 0.2452 | 0.0000 |
| 0.0494 | 2.9812 | 3.7767 | 0.0000 |
| 2.0993 | 1.6448 | 4.4741 | 0.0000 |
| 3.7683 | 2.3910 | 1.4307 | 0.0000 |
| 3.9694 | 2.9859 | 1.2560 | 0.0000 |
| 4.5998 | 0.8072 | 4.6637 | 0.0000 |
| 4.2236 | 4.1474 | 0.6549 | 0.0000 |
| 1.8388 | 4.7806 | 4.7041 | 0.0000 |
| 3.1040 | 2.9777 | 3.5093 | 0.0000 |
| 3.6564 | 0.1437 | 4.2384 | 0.0000 |
| 0.9695 | 4.0606 | 1.0464 | 0.0000 |
| 4.5241 | 3.0506 | 2.2755 | 0.0000 |
| 2.8460 | 3.5075 | 0.4054 | 0.0000 |
| 3.1589 | 0.4610 | 4.2556 | 0.0000 |
| 1.1721 | 2.1244 | 2.8102 | 0.0000 |
| 2.7439 | 1.8779 | 1.5965 | 0.0000 |
| 4.6579 | 0.8308 | 1.8745 | 0.0000 |
| 1.6760 | 4.1658 | 4.3390 | 0.0000 |
| 3.2777 | 4.1932 | 1.8609 | 0.0000 |
| 1.9595 | 2.2581 | 0.3685 | 0.0000 |
| 3.1366 | 4.7830 | 0.9992 | 0.0000 |
| 3.4954 | 0.7358 | 0.2475 | 0.0000 |
| 1.9859 | 4.3497 | 2.8335 | 0.0000 |
| 2.0681 | 3.8472 | 0.6096 | 0.0000 |
| 3.2761 | 2.2208 | 2.6106 | 0.0000 |
| 4.1879 | 3.1031 | 0.5853 | 0.0000 |
| 1.8580 | 4.7584 | 3.8496 | 0.0000 |
| 2.1263 | 3.2000 | 1.8753 | 0.0000 |
| 2.9733 | 1.2366 | 4.1169 | 0.0000 |
| 2.8287 | 1.7635 | 0.2332 | 0.0000 |

Resultados arrojados por el programa triángulo:

| | | | |
|---|---|---|---|
| 1.5028 | 3.9925 | 0.7069 | 5.9999 |

**3.5.3** Consideremos ahora el siguiente problema, donde se quiere optimizar una función de tipo no lineal, sujeta a una restricción. Este problema es expresado como: [17]

$$\begin{aligned} &Max\ f = x_1 x_2 x_3 \\ &sujeta\ a\ x_1 + x_2 + x_3 = L \quad (0) \\ &x_1, x_2, x_3 \geq 0 \end{aligned}$$

Despejando a $x_3$ y sustituyendo, se obtiene:

$$f = x_1 x_2 (L - x_1 - x_2) = L x_1 x_2 - x_1^2 x_2 - x_1 x_2^2$$

$$\frac{\partial f}{\partial x_1} = L x_2 - 2 x_1 x_2 - x_2^2 = 0 \qquad (1)$$

$$\frac{\partial f}{x_2} = L x_1 - x_1^2 - 2 x_1 x_2 = 0 \qquad (2)$$

De (1) $x_2 = 0, L - 2x_1 - x_2 = 0$ (3)

De (2) $x_1 = 0, L - 2x_2 - x_1 = 0$ (4)

Sustituyendo (3) en (4) se obtiene:

$$L - 2(L - 2x_1) - x_1 = 0$$

$$L - 2L + 4x_1 - x_1 = 0$$

$$-L + 3x_1 = 0$$

$$x_1^* = \frac{L}{3}$$

$$x_2^* = L - 2\frac{L}{3} = \frac{L}{3}$$

Sustituyendo en (0);

$x_3^* = \frac{L}{3}$ por lo tanto $f^* = \frac{L^3}{27}$

Utilizando el método de Lagrange, la función lagrangiana se escribe como:

$$h(x_1, x_2, x_3, \lambda_1) = x_1 x_2 x_3 + \lambda_1 (x_1 + x_2 + x_3 - L)$$

$$\nabla h(x_1, x_2, x_3, \lambda_1) = \begin{bmatrix} \frac{\partial h}{\partial x_1} \\ \frac{\partial h}{\partial x_2} \\ \frac{\partial h}{\partial x_3} \\ \frac{\partial h}{\partial \lambda_1} \end{bmatrix} = \begin{bmatrix} x_2 x_3 + \lambda_1 \\ x_1 x_3 + \lambda_1 \\ x_1 x_2 + \lambda_1 \\ x_1 + x_2 + x_3 - L \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

El Hessiano se escribe como:

$$H = \begin{pmatrix} -\frac{2L}{3} & -\frac{L}{3} \\ -\frac{L}{3} & -\frac{2L}{3} \end{pmatrix}$$

Det (H)= $\frac{4L^2}{9} - \frac{L^2}{9} = \frac{3L^2}{9} > 0$

Como H es definida negativa, entonces la solución $x_1^* = x_2^* = x_3^* = \frac{L}{3}$ corresponde a un máximo.

**Para implantar computacionalmente** el Algoritmo Evolutivo se requiere especificar:

cycles: nº individuos que posee la población inicial.
mutac: nº de mutaciones que realizará el algoritmo.

La población inicial es generada de manera aleatoria y almacenada en una matriz denominada “A” que posee la siguiente estructura:

| | | | | | |
|---|---|---|---|---|---|
| Individuo 1 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $p_1$ |
| Individuo 2 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $p_2$ |
| Individuo 3 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $p_3$ |
| | ……. | …… | …… | …… | ….. |
| | …… | ……. | ……. | …… | ….. |
| Individuo k → | …… | …… | …… | …… | …… |

$$g_1(x) = x_1 + x_2 + x_3 - L$$

La función Lagrangiana viene dada por:

$$f(x_1, x_2, x_3, \lambda_1) = x_1 x_2 x_3 + \lambda_1 (x_1 + x_2 + x_3 - L)$$

Si se cumple que $g_1(x) \leq 0$ , $d(X) = \nabla f(X)$

$$\nabla f(X) = \begin{bmatrix} x_2 x_3 + \lambda_1 \\ x_1 x_3 + \lambda_1 \\ x_1 x_2 + \lambda_1 \\ x_1 + x_2 + x_3 - L \end{bmatrix}$$

Donde ( $X$ ) es un vector que posee 4 componentes $(x_1, x_2, x_3, \lambda_1)$

En caso de que $g_i(x) > 0$ la dirección del gradiente viene dada por:

$$d(X) = \nabla f(X) - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

$$d(X) = \begin{bmatrix} x_2 x_3 + \lambda_1 \\ x_1 x_3 + \lambda_1 \\ x_1 x_2 + \lambda_1 \\ x_1 + x_2 + x_3 - L \end{bmatrix} - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

Por lo que la dirección del gradiente ponderado para cada una de las variables pertenecientes a cada individuo se calcula como:

$$d(x_1) = x_2 x_3 + \lambda_1 - w_1(1)$$
$$d(x_2) = x_1 x_3 + \lambda_1 - w_1(1)$$
$$d(x_3) = x_1 x_2 + \lambda_1 - w_1(1)$$
$$d(\lambda_1) = x_1 + x_2 + x_3 - L$$

A este problema se le aplica la Metodología explicada anteriormente, con diferencias en el número de variables, función de penalización, restricciones, valor del gradiente y por lo tanto dirección del mismo.

#### 3.5.3.1 Algoritmo Evolutivo

Parte de la población inicial del problema 3.2.3, generada de manera aleatoria

| | | | | |
|---|---|---|---|---|
| 4.7506 | 2.1990 | 1.4297 | 4.4760 | 0.0000 |
| 1.1557 | 1.7002 | 1.9706 | 4.7119 | 0.0000 |
| 3.0342 | 1.5711 | 2.5151 | 1.6754 | 0.0000 |
| 2.4299 | 1.8254 | 3.6099 | 2.1868 | 0.0000 |
| 4.4565 | 1.9662 | 1.5310 | 2.3558 | 0.0000 |
| 3.8105 | 2.9576 | 0.5608 | 0.7466 | 0.0000 |
| 2.2823 | 0.5987 | 2.2164 | 0.6793 | 0.0000 |
| 0.0925 | 0.1906 | 2.3338 | 2.6625 | 0.0000 |
| 4.1070 | 2.2930 | 0.0733 | 3.6289 | 0.0000 |
| 2.2235 | 4.3493 | 3.3203 | 1.9935 | 0.0000 |
| 3.0772 | 4.6712 | 3.6203 | 1.7921 | 0.0000 |
| 3.9597 | 1.3222 | 1.4082 | 1.4264 | 0.0000 |
| 4.6091 | 0.8015 | 1.3091 | 4.3432 | 0.0000 |
| 3.6910 | 4.3643 | 3.5424 | 3.1321 | 0.0000 |
| 0.8813 | 1.1894 | 3.9193 | 1.2059 | 0.0000 |
| 2.0285 | 3.2292 | 4.9308 | 4.8904 | 0.0000 |
| 4.6773 | 4.8344 | 2.3667 | 3.2025 | 0.0000 |
| 4.5845 | 3.3247 | 4.5141 | 1.1492 | 0.0000 |
| 2.0514 | 4.3519 | 2.2553 | 3.4067 | 0.0000 |
| 4.4682 | 0.0496 | 4.0226 | 3.3291 | 0.0000 |
| 0.2895 | 0.6850 | 4.1443 | 0.6736 | 0.0000 |
| 1.7643 | 4.0938 | 0.8314 | 0.1125 | 0.0000 |
| 4.0658 | 2.1508 | 1.9695 | 1.3110 | 0.0000 |
| 0.0493 | 4.4516 | 2.6038 | 0.5826 | 0.0000 |
| 0.6945 | 3.6745 | 3.5906 | 0.3466 | 0.0000 |
| 1.0138 | 3.4366 | 2.8459 | 4.2647 | 0.0000 |
| 0.9936 | 1.7306 | 2.3040 | 0.9017 | 0.0000 |
| 3.0190 | 0.8302 | 2.2265 | 0.1621 | 0.0000 |
| 1.3609 | 0.7781 | 0.4387 | 3.6696 | 0.0000 |
| 0.9941 | 0.9556 | 2.2174 | 2.6826 | 0.0000 |
| 0.0764 | 2.1123 | 1.8315 | 1.3801 | 0.0000 |
| 3.7339 | 4.2799 | 1.5127 | 1.8423 | 0.0000 |
| 2.2255 | 2.4512 | 4.2592 | 0.0644 | 0.0000 |
| 4.6591 | 4.0797 | 3.7974 | 4.4460 | 0.0000 |
| 2.3300 | 2.3038 | 4.7488 | 4.3301 | 0.0000 |
| 2.0932 | 2.2868 | 2.7897 | 1.2712 | 0.0000 |
| 4.2311 | 2.2534 | 0.0712 | 2.8474 | 0.0000 |
| 2.6258 | 2.0611 | 2.9809 | 0.7963 | 0.0000 |
| 1.0132 | 4.5080 | 4.0810 | 2.9718 | 0.0000 |
| 3.3607 | 0.0279 | 4.8855 | 1.6555 | 0.0000 |
| 4.1906 | 1.4870 | 1.1095 | 3.2931 | 0.0000 |
| 0.0982 | 0.2458 | 3.5184 | 4.3182 | 0.0000 |
| 3.4064 | 3.4659 | 2.6103 | 2.8381 | 0.0000 |
| 1.8974 | 3.2505 | 4.6645 | 4.9024 | 0.0000 |
| 4.1590 | 4.9149 | 3.5668 | 3.9592 | 0.0000 |
| 2.5141 | 2.7634 | 1.1402 | 0.7630 | 0.0000 |
| 3.5474 | 2.0004 | 2.2482 | 4.1651 | 0.0000 |
| 2.1445 | 0.9939 | 0.8610 | 0.9593 | 0.0000 |
| 1.5231 | 3.1260 | 4.8441 | 3.1949 | 0.0000 |
| 0.9483 | 3.6668 | 1.7786 | 3.3450 | 0.0000 |
| 0.9672 | 1.8794 | 0.2452 | 3.8604 | 0.0000 |
| 3.4111 | 0.0494 | 3.7767 | 1.8991 | 0.0000 |
| 1.5138 | 2.0993 | 4.4741 | 2.2079 | 0.0000 |
| 2.7084 | 3.7683 | 1.4307 | 2.4153 | 0.0000 |
| 0.7544 | 3.9694 | 1.2560 | 3.0405 | 0.0000 |
| 3.4895 | 4.5998 | 4.6637 | 0.8800 | 0.0000 |

Resultado de la corrida realizada al problema 3.2.3

```
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
4.8695      4.9484      5.1774      1.7488      124.7457
```

**3.5.4** Consideremos el problema, donde se requiere maximizar una función de tipo no lineal, sujeta a dos restricciones de tipo lineal. Este problema se escribe como: [17]

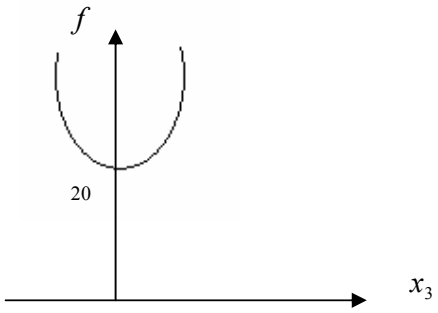
$$\begin{aligned} &Max\ f = x_1^2 + x_2^2 + x_3^2 \\ &sujeta\ a: x_1 + x_2 = 6 \\ &\qquad\quad x_1 - x_2 = 2 \\ &\qquad\quad x_1, x_2, x_3 \geq 0 \end{aligned}$$

**Por Sustitución** (analíticamente) $x_1 = 4; x_2 = 2$

$f(x_1, x_2, x_3) = 16 + 4 + x_3^2$

$\frac{\partial f}{\partial x_3} = 2x_3 = 0; \quad x_3 = 0 \qquad$ y $\qquad \frac{\partial^2 f}{\partial x_3^2} = 2 > 0$

Como el valor de la segunda derivada de la función es positiva, estoy en presencia de un mínimo, el valor óptimo para este caso sería $x_1 = 4; x_2 = 2; x_3 = 0; f = 20$, pero como estamos buscando el punto máximo, éste debe ser investigado en la función.




$x_3 = \infty$ ó $x_3 = -\infty$

Utilizando el método de Lagrange:

$L = x_1^2 + x_2^2 + x_3^2 + \lambda_1(x_1 + x_2 - 6) + \lambda_2(x_1 - x_2 - 2)$

$$\nabla L(x_1, x_2, x_3, \lambda_1, \lambda_2) = \begin{bmatrix} 2x_1 + \lambda_1 + \lambda_2 \\ 2x_2 + \lambda_1 - \lambda_2 \\ 2x_3 \\ x_1 + x_2 - 6 \\ x_1 - x_2 - 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

El Hessiano viene dado por:

$$H = \begin{bmatrix} 2 & 0 & 0 & 1 & 1 \\ 0 & 2 & 0 & 1 & -1 \\ 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \end{bmatrix}$$

Aplicando el criterio de Hancock:

$$\text{Det}(H) = \begin{bmatrix} 2-z & 0 & 0 & 1 & 1 \\ 0 & 2-z & 0 & 1 & -1 \\ 0 & 0 & 2-z & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \end{bmatrix} \quad z = 2$$

Por lo que se determina que el valor óptimo para este problema es:

$$x_1 = 4; x_2 = 2; x_3 = \pm\infty; f = \pm\infty$$

**Empíricamente:**

cycles: nº individuos que posee la población inicial.

mutac: nº de mutaciones que realizará el algoritmo.

La población inicial es generada de manera aleatoria y almacenada en una matriz denominada “A” que posee la siguiente estructura:

| | | | | | | |
|---|---|---|---|---|---|---|
| Individuo 1 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p_1$ |
| Individuo 2 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p_2$ |
| Individuo 3 → | $x_1$ | $x_2$ | $x_3$ | $\lambda_1$ | $\lambda_2$ | $p_3$ |
| | ……. | …… | …… | …… | …… | ….. |
| | …… | ……. | ……. | …… | …… | ….. |
| Individuo k → | …… | …… | …… | …… | …… | …… |

$$g_1(x) = x_1 + x_2 - 6$$
$$g_2(x) = x_1 - x_2 - 2$$

La función Lagrangiana viene dada por:

$$L = x_1^2 + x_2^2 + x_3^2 + \lambda_1(x_1 + x_2 - 6) + \lambda_2(x_1 - x_2 - 2)$$

Si se cumple que $g_1(x) \leq 0$ , $d(X) = \nabla L(X)$

$$\nabla L(X) = \begin{bmatrix} 2x_1 + \lambda_1 + \lambda_2 \\ 2x_2 + \lambda_1 - \lambda_2 \\ 2x_3 \\ x_1 + x_2 - 6 \\ x_1 - x_2 - 2 \end{bmatrix}$$

Donde ( $X$ ) es un vector que posee 5 componentes $(x_1, x_2, x_3, \lambda_1, \lambda_2)$

En caso de que $g_i(x) > 0$ la dirección del gradiente viene dada por:

$$d(X) = \nabla L(X) - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

$$d(X) = \begin{bmatrix} 2x_1 + \lambda_1 + \lambda_2 \\ 2x_2 + \lambda_1 - \lambda_2 \\ 2x_3 \\ x_1 + x_2 - 6 \\ x_1 - x_2 - 2 \end{bmatrix} - \sum_{i=1}^{m} w_i \nabla g_i(x)$$

Por lo que la dirección del gradiente ponderado para cada una de las variables pertenecientes a cada individuo se calcula como:

$$d(x_1) = 2x_1 + \lambda_1 + \lambda_2 - w_1(1) - w_2(1)$$
$$d(x_2) = 2x_2 + \lambda_1 - \lambda_2 - w_1(1) - w_2(-1)$$
$$d(x_3) = 2x_3$$
$$d(\lambda_1) = x_1 + x_2 - 6$$
$$d(\lambda_2) = x_1 - x_2 - 2$$

#### 3.5.4.1 Algoritmo Evolutivo

Parte de la población inicial generada de manera aleatoria para el problema 3.2.4.

| | | | | | |
|---|---|---|---|---|---|
| 3.0053 | 1.7963 | 2.5460 | 3.5841 | 2.9075 | 0.0000 |
| 0.0212 | 3.9382 | 4.8375 | 0.5052 | 1.9366 | 0.0000 |
| 3.9956 | 3.6214 | 4.3750 | 0.0526 | 1.9715 | 0.0000 |
| 3.4024 | 0.4445 | 4.6692 | 0.5014 | 0.4883 | 0.0000 |
| 4.7664 | 4.9954 | 4.8742 | 3.1337 | 3.0098 | 0.0000 |
| 1.8107 | 4.3662 | 0.9925 | 1.1715 | 1.4625 | 0.0000 |
| 0.0293 | 2.9758 | 2.5581 | 2.2012 | 1.0201 | 0.0000 |
| 2.6307 | 3.0170 | 0.3867 | 3.5859 | 0.9221 | 0.0000 |
| 2.3135 | 2.9306 | 0.1970 | 4.8493 | 4.5977 | 0.0000 |
| 2.1683 | 4.5937 | 3.0863 | 1.5975 | 4.2097 | 0.0000 |
| 0.8134 | 1.4718 | 0.8298 | 1.3733 | 4.9100 | 0.0000 |
| 3.6419 | 3.9522 | 1.3122 | 0.8980 | 3.6910 | 0.0000 |
| 3.3576 | 3.2095 | 4.2422 | 3.5294 | 0.5017 | 0.0000 |
| 1.6548 | 2.4156 | 0.4417 | 3.2118 | 1.1691 | 0.0000 |
| 0.4356 | 3.8176 | 3.8697 | 4.3976 | 1.9929 | 0.0000 |
| 4.0240 | 3.5887 | 0.9901 | 3.0107 | 2.8484 | 0.0000 |
| 3.4217 | 0.6019 | 1.9821 | 2.9311 | 1.2148 | 0.0000 |
| 3.3897 | 2.3707 | 0.5613 | 4.4993 | 0.1155 | 0.0000 |
| 3.4582 | 2.1585 | 1.0801 | 1.1181 | 4.6523 | 0.0000 |
| 4.0458 | 0.7905 | 0.6076 | 0.0081 | 0.4794 | 0.0000 |
| 4.5312 | 3.5544 | 3.9161 | 3.5191 | 4.9081 | 0.0000 |
| 2.2780 | 3.8208 | 2.7000 | 2.6249 | 0.9966 | 0.0000 |
| 3.2052 | 0.0537 | 4.8871 | 2.9983 | 0.9597 | 0.0000 |
| 0.6934 | 0.8908 | 2.7415 | 2.9171 | 4.1709 | 0.0000 |
| 3.7963 | 4.3974 | 2.2090 | 2.6875 | 2.3226 | 0.0000 |
| 0.2616 | 4.5285 | 3.4295 | 3.2496 | 3.0562 | 0.0000 |
| 3.5020 | 0.0457 | 1.8247 | 4.4338 | 3.9211 | 0.0000 |
| 0.7764 | 1.9139 | 2.2981 | 0.1257 | 3.5292 | 0.0000 |
| 2.6363 | 2.1216 | 3.5145 | 4.9541 | 3.9355 | 0.0000 |
| 4.7976 | 4.7310 | 0.9475 | 4.7460 | 4.7520 | 0.0000 |
| 4.8139 | 2.2923 | 3.2800 | 0.3589 | 1.9058 | 0.0000 |
| 0.2876 | 4.6707 | 0.4401 | 0.6528 | 0.7423 | 0.0000 |
| 1.3353 | 1.8001 | 2.0294 | 4.7331 | 3.1358 | 0.0000 |
| 1.5327 | 0.4543 | 0.4916 | 1.7605 | 0.2418 | 0.0000 |
| 0.0942 | 0.3741 | 4.2150 | 4.5354 | 0.0859 | 0.0000 |
| 2.7000 | 2.7510 | 2.2722 | 0.4076 | 3.2375 | 0.0000 |
| 4.7675 | 4.5267 | 3.1807 | 1.0139 | 3.4846 | 0.0000 |
| 3.2335 | 2.6427 | 2.2627 | 1.6930 | 1.5004 | 0.0000 |
| 3.4000 | 1.5859 | 2.8500 | 3.8673 | 0.0646 | 0.0000 |
| 1.6305 | 2.6976 | 2.6850 | 0.7890 | 3.0994 | 0.0000 |
| 1.2933 | 0.0285 | 3.3969 | 4.0710 | 4.6741 | 0.0000 |
| 3.1597 | 2.8935 | 4.3745 | 1.2237 | 2.4974 | 0.0000 |
| 0.0288 | 2.0090 | 1.9622 | 0.9047 | 3.1817 | 0.0000 |
| 2.1778 | 1.7819 | 3.4443 | 1.9012 | 3.0264 | 0.0000 |
| 3.8202 | 1.1910 | 0.7449 | 2.8771 | 2.5670 | 0.0000 |
| 4.2147 | 2.8448 | 2.5057 | 1.9210 | 1.8873 | 0.0000 |
| 0.9641 | 2.5233 | 4.3223 | 3.2073 | 4.1044 | 0.0000 |
| 4.8966 | 2.6230 | 3.2748 | 0.8232 | 0.3603 | 0.0000 |
| 0.9583 | 3.2968 | 0.4236 | 1.6427 | 4.7411 | 0.0000 |
| 0.4395 | 2.6979 | 4.8295 | 4.9807 | 0.0792 | 0.0000 |
| 1.8858 | 0.9695 | 2.2512 | 3.6057 | 1.6306 | 0.0000 |
| 1.7239 | 0.2506 | 0.7035 | 4.6195 | 2.2109 | 0.0000 |
| 1.2970 | 0.5161 | 0.7326 | 3.7926 | 4.3384 | 0.0000 |
| 2.4888 | 4.3085 | 1.9949 | 2.6082 | 2.2931 | 0.0000 |
| 4.9789 | 0.4641 | 0.9268 | 1.1517 | 0.1319 | 0.0000 |
| 2.9158 | 0.5920 | 4.1289 | 4.7305 | 0.1924 | 0.0000 |

```
Resultado:

3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
3.7266      2.1003      8.9727      2.3595      3.7414      97.0008
```

# *CAPITULO IV*

# *CONCLUSIONES*

El trabajo realizado fundamenta su enfoque en el uso de un algoritmo evolutivo para resolver problemas de programación matemática; particularmente problemas de Programación no lineal. En este sentido, cada potencial solución al problema de optimización es vista como un individuo, dentro de la población, que es sometido a operadores genéticos del tipo selección y mutación, hasta lograr una convergencia a la solución del problema de optimización considerado.

El proceso de selección ha sido realizado de la manera siguiente: Partiendo de una población inicial, que fue generada en forma aleatoria, se hizo uso del operador mutación para crear nuevos individuos, produciendo como consecuencia una duplicación del tamaño de la población inicial. La selección conllevó a tomar aquellos individuos que poseían un valor de aptitud positivo, para luego continuar con la aplicación del operador mutación, logrando así que el tamaño de la población fuese creciendo hasta alcanzar un límite, especificado como “CYCLESMAX” , equivalente a diez veces el tamaño de la población inicial.

Los resultados arrojados por el Algoritmo Evolutivo para el problema especificado en la sección 3.5.1 fueron los siguientes:

$x_1 = 6.0365,\ x_2 = 5.7157,\ x_3 = 5.2072,\ \lambda_1 = 0.8760,\ \lambda_2 = 0.0037, f = 216.0792$

mientras que la solución encontrada en forma analítica fue:

$x_1 = 6.412, x_2 = 5.511,\ x_3 = 5.076,\ f = 216.6$, por lo que es posible concluir que se logró un resultado en una vecindad del óptimo.

Con respecto a la aplicación del algoritmo evolutivo a los problemas especificados en la sección 3.5.2, 3.5.3, 3.5.4, los resultados fueron bastantes cercanos a lo óptimos obtenidos en forma analítica.

Para el problema 3.5.2, con c=6 y h=8, los valores óptimos calculados analíticamente son: $x = \frac{c}{4} = \frac{6}{4} = 1.5$, $y = \frac{h}{2} = \frac{8}{4} = 2$, $f = \frac{ch}{8} = \frac{6(8)}{8} = 6$, mientras que el resultado arrojado por el Algoritmo Evolutivo fue:

$$x = \frac{c}{4} = \frac{6}{4} = 1.5028, \quad y = \frac{h}{2} = \frac{8}{4} = 3.9925, \quad f = \frac{ch}{8} = \frac{6(8)}{8} = 5.9999$$

Para el problema 3.2.3, con L=15 , los valores óptimos calculados de manera analítica vienen dados por: $x_1 = x_2 = x_3 = \frac{L}{3} = 5$ , mientras que la construcción Algoritmo Evolutivo para este problema arroja los siguientes resultados: $x_1 = 4.8695,\ x_2 = 4.9484,\ x_3 = 5.1774,\ f = 124.7457$

Para el problema 3.5.4, no existe un valor máximo definido, ya que la gráfica de la función objetivo, está representada por una parábola cuyo punto de inflexión es un mínimo, y el valor para la función objetivo, tiende a infinito a medida que la variable $x_3$ toma valores entre $\pm\infty$; mientras que los valores para las variables $x_1$ y $x_2$ se mantienen en 4 y 2 respectivamente.

# *REFERENCIAS BIBLIOGRÁFICAS*

[1]Tang, J. and Wang, D., 1997, An interactive approach based on GA for a type of Quadratic Programming Problems with fuzzy objetive and resourses, Computers & Operations Research, 24(5), pp413-422.

[2]http:/www.uv.es/~sala/pnl01.PDF

[3]http:/departamentos.unican.es/macc/ personal/profesores/castillo/Libro/Chap3

[4]http://www.umanizales.edu.co/congreso/ponencias/computacionevolutiva.pdf

[5]http://tf.usb.ve/miqusb/volii/no2/luna.pdf

[6]http://ccc.inaoep.mx/~emorales/Cursos/Busqueda04/node98.html

[7] http://es.wikipedia.org/wiki/Algoritmos_gen%C3%A9ticos

[8] http://the-geek.org/docs/algen/ algoritmos geneticos

[9] http://www.depi.itch.edu.mx/apacheco/expo/html/ai14/ga.html - 47k

[10] http://tf.usb.ve/miqusb/voliii/no1/ camargo-mendez-orta%20(resumen).pdf

[11]http://www2.ing.puc.cl/~dmery/vision/matlab61pro.pdf

[12]http://www14.uniovi.es/ia/Genetico-TSP/AGs.htm

[13]http://supercriticos.univalle.edu.co/Reprints%20PDF/Patricia.PDF

[14]http://www.ua.es/cuantica/docencia/qc_av/qc_av_b/node39.html

[15] http://cursos.itam.mx/lomeli/maf/apuntes/convex.pdf

[16] Rao, S.S: Optimization, Theory and Aplications
John Wiley,1979.

[17] Ponsot Ernesto: Programación no lineal, Apuntes.

[18] Matlab Help, versión 7.0