

# ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL



FACULTAD DE CIENCIAS NATURALES Y MATEMÁTICAS

DEPARTAMENTO DE MATEMÁTICAS

EXAMEN COMPLEXIVO

PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

“MAGÍSTER EN CONTROL DE OPERACIONES Y GESTIÓN LOGÍSTICA”

TEMA

“DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO GENÉTICO Y LA DE UN MODELO MATEMÁTICO, PARA EL PROBLEMA DE LA PLANIFICACIÓN DE LA PRODUCCIÓN Y SIMULACIÓN DE UN PROCESO PARA LA TOMA DE PEDIDO A UNA BODEGA.”

AUTOR:

ING. ROBERTH ARTURO TINOCO ROMERO

Guayaquil- Ecuador

AÑO

2016

## DEDICATORIA Y AGRADECIMIENTO

*A Dios, por haberme permitido llegar hasta este punto y haberme fortalecido en los momentos difíciles para lograr mis objetivos, además de su infinita bondad y amor.*

*A mi esposa Yesenia, por haber sido mi apoyo en todo momento, por sus consejos, sus valores, por la motivación constante que me ha permitido ser una persona de bien, pero más que nada, por su amor.*

*A mis hijos Sebastián y Esteban, por la motivación que por sí solo representan y por sacrificar el tiempo que delegué a la obtención de este nuevo título académico en mi vida.*

*A mis padres Arturo y Marina, por ser ejemplos de perseverancia y constancia que los caracterizan y que me ha infundado siempre, por el valor mostrado para salir adelante y por su amor.*

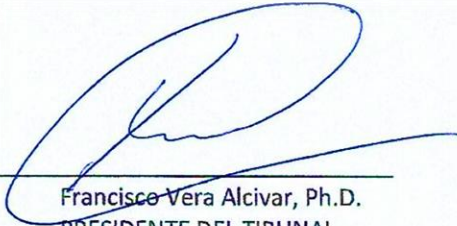
## DECLARACIÓN EXPRESA

*“La responsabilidad por los hechos y doctrinas expuestas en este Proyecto de Examen Complexivo, así como el Patrimonio Intelectual del mismo, me corresponden exclusivamente; el patrimonio intelectual del mismo, corresponde exclusivamente a la Facultad de Ciencias Naturales y Matemáticas, Departamento de Matemáticas de la Escuela Superior Politécnica del Litoral”.*

A handwritten signature in dark ink, appearing to read 'Roberth Tinoco R.', with a horizontal line drawn across the bottom of the signature.


*Ing. Roberth Tinoco R.*

## TRIBUNAL DE GRADUACIÓN



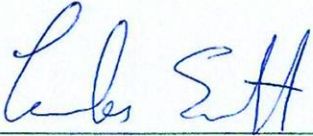
---

Francisco Vera Alcivar, Ph.D.  
PRESIDENTE DEL TRIBUNAL



---

M.Sc. Carlos Martin Barreiro.  
DIRECTOR DEL EXAMEN COMPLEXIVO



---

M.Sc. Carlos Anibal Suarez H.  
EVALUADOR

# TABLA DE CONTENIDO

	<b>Pág.</b>
ÍNDICE DE TABLAS	I
ÍNDICE DE FIGURAS	I

## CAPITULO I

### 1. GENERALIDADES

1.1 Antecedentes.....	1
1.2 Planteamiento de los Problemas.....	3
1.2.1 Problema de la Planificación de la Producción.....	3
1.2.2 Simulación para la toma de pedido a una bodega.....	4
1.4 Objetivo General.....	5
1.5 Objetivos Específicos.....	6

## CAPITULO II

### 2. PRINCIPIOS BÁSICOS DE LOS PROBLEMAS.

2.1 Problema de la Planificación de la Producción .....	7
2.2 Generación de Variables Aleatorias a partir de una Distribución Empírica continua para Simulación.....	9

## **CAPITULO III**

### **3. DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO GENÉTICO Y LA DE UN MODELO MATEMÁTICO, PARA EL PROBLEMA DE LA PLANIFICACIÓN DE LA PRODUCCIÓN**

3.1	Formulación del Modelo de Programación Lineal.....	11
3.2	Diseño y aplicación del Modelo Matemático al problema de Planificación de la Producción .....	14
3.2.1	Codificación del problema de optimización en GAMS.....	14
3.2.2	Resultados obtenidos.....	18
3.3	Diseño e Implementación del Algoritmo Genético para el problema de la Planificación de la Producción.....	20
3.3.1	Conceptos Básicos del Algoritmo Genético.....	20
3.3.2	Algoritmo Genético aplicado al Problema de la planificación de la Producción.....	21
3.3.3	Resultados Obtenidos.....	25

## **CAPITULO IV**

### **4. SIMULACIÓN DE TOMA DE PEDIDOS A UNA BODEGA A PARTIR DE UNA DISTRIBUCIÓN EMPÍRICA CONTINUA**

4.1.	Construcción de la Función de Distribución Continua Empírica ...	27
4.2.	Formulación en MATHEMATICA para la Simulación .....	29
4.2.1	Resultados Obtenidos.....	30

## **CAPITULO V**

### **5. CONCLUSIONES Y RECOMENDACIONES**

5.1 Conclusiones 31

5.2 Recomendaciones 32

**BIBLIOGRAFIA** 33

**ANEXOS** 34

## INDICES DE TABLAS

	<b>Pág.</b>
Tabla 1.1: Demanda, costos de producción e inventario por trimestre	1
Tabla 1.2: Frecuencia por intervalos de número de unidades	2
Tabla 3.1: Resultado por defectos modelo en GAMS	19
Tabla 3.2: Resultado del modelo en GAMS en archivo HTML.	19
Tabla 4.1: Resultado del AG utilizando una Población de 18 e iteraciones Máximas de 16	25
Tabla 4.2: Resultados del AG con diversas parametrizaciones	26

## INDICES DE GRÁFICAS.

	<b>Pág.</b>
Gráfica 2.1: Demanda en función del tiempo	7
Gráfica 3.1: Cabecera del programan en GAMS.	14
Gráfica 3.2: Declaración de parámetros en GAMS.	15
Gráfica 3.3: Declaración de Variables en GAMS.	16
Gráfica 3.4: Declaración de ecuaciones en GAMS.	16
Gráfica 3.5: Declaración de modelos en GAMS.	17
Gráfica 3.6: Declaración de llamada de solver en GAMS	17



# **CAPÍTULO I**

## **1. GENERALIDADES**

### **1.1 Antecedentes**

A partir de las primeras décadas del siglo pasado, la planificación de la producción se ha desarrollado rápidamente llegando en la actualidad, a ser reconocida como una de las claves para el correcto funcionamiento de las operaciones productivas y de la empresa en su conjunto.

Como resultado del enfoque en reducir los costos de los recursos necesarios para satisfacer la demanda en cierto plazo, surge el concepto de planeación de operaciones y ventas, cuyo proceso considera manejar la agregación de lado de la oferta por familias de productos y del lado de la demanda por grupo de cliente, permitiendo gestionar más fácilmente los programas de producción de productos individuales y sus correspondientes pedidos de los clientes<sup>[1]</sup>.

Al respecto, números autores han propuesto diversos modelos de optimización para resolver el problema de planificación de producción, desde una perspectiva clásica con la participación del uso de técnica de programación matemática, hasta procedimientos heurísticas y técnicas de búsqueda.

Asimismo, de forma complementaria a lo expuesto, el hombre través de la historia ha experimentado diversos métodos y procedimientos con el propósito de lograr en forma efectiva entender los continuos cambios y avances en la logística y de los sistemas productivos, para la realización de mejoras y la de toma de decisiones oportunas.

Por esta razón, desde la aparición de la computadora, se buscaron formas para aprovechar, el gran potencial que ella presentaba, entre las cuales destaca la simulación, como una excelente herramienta de apoyo para la evaluación y el análisis de los sistemas nuevos y de los ya existentes, permitiendo anticiparse al proceso real, validarlo y obtener su mejor configuración.

En el presente trabajo, se abarcará problemas relacionados tanto a la planificación de producción como al de una simulación, cuyas soluciones se solventarán en herramientas y/o lenguajes de programación disponibles.

## **1.2 Planteamiento de los Problemas.**

### **1.2.1 Problema de la Planificación de la Producción.**

Una fábrica ecuatoriana ha dividido la producción anual de uno de sus productos principales en 4 trimestres. En cada periodo, la empresa tiene una limitante de producción para dicho producto de 150 unidades, debido a la capacidad de su planta y a los recursos de los que dispone.

Considerando la tabla 1, con información de demanda, costos de producción y costos de inventario, e iniciando con una existencia de inventario de 15 unidades del producto, se requiere elaborar un modelo matemático para determinar cuánto se debe producir del producto y cuánto se debe almacenar en inventario por cada trimestre de forma tal que los costos totales sean mínimos, teniendo presente que se debe satisfacer la demanda de cada trimestre.

Periodo	Demanda (unidades)	Costos de Producción (\$/unidad)	Costos de Inventario (\$/unidad)
1	130	6	2
2	80	4	1
3	125	8	2.5
4	195	9	3

Tabla 1.1: Demanda, costos de producción e inventario por trimestre

Una vez determinada la solución óptima del problema, se deberá diseñar e implementar una Metaheurística para resolver el mismo problema usando el lenguaje y/o herramienta de preferencia, para finalmente realizar un análisis comparativo. Para tales propósitos, se diseñará e implementará un algoritmo genético usando el software MATHEMATICA.

## 1.2.2 Simulación para la toma de pedido a una bodega.

Durante una semana de producción una empresa mide el número de unidades de una materia prima en particular, que son solicitadas a una bodega de insumos con propósitos de fabricación. La siguiente tabla muestra los datos resumidos de los 50 pedidos realizados en dicha semana de estudio:

Número de Unidades	Frecuencia (# Pedidos)
[50,60)	3
[60,70)	7
[70,80)	18
[80,90)	12
[90,100)	8
[100,110)	2

Tabla 1.2: Frecuencia por intervalos de número de unidades

Considerando esta información, se debe construir una distribución continua empírica con el afán de realizar una simulación de diez pedidos, utilizando para tales fines el software MATHEMATICA.

### **1.3 Objetivo General**

Para el problema de la Planificación de la Producción, se requiere diseñar e implementar tanto un modelo de programación matemática como una técnica heurística, para resolver los siguientes problemas:

- Determinar el número máximo a producir por trimestre, respetando las limitaciones de capacidad y demanda.
- Desarrollar un proceso de formulación y solución simple, para lo cual se deberá codificar el problema de optimización empleando la herramienta de programación GAMS y MATHEMATICA para la implementación del algoritmo genético.
- Elaborar un algoritmo lo suficientemente general para resolver problemas semejantes.

Para el problema de la Simulación de Pedidos, se requiere:

- Construir una función de distribución continua empírica.
- Desarrollar una solución simple en MATHEMATICA, en base a la distribución continua empírica para la generación de variables aleatorias.

## **1.4 Objetivos Específicos**

Los objetivos específicos que persigue en el presente proyecto son:

- Definir una programación de la producción trimestral que satisfaga la demanda y restricciones del caso, incurriendo en costos totales mínimos.
- Simulación de  $n$  pedidos considerando una distribución empírica continua para a la generación de variables aleatorias.

## **CAPÍTULO II**

### **2. DEFINICIONES DE LOS PROBLEMAS:**

#### **2.1 Problema de la Planificación de la Producción.**

Un productor fabrica una pieza, cuya demanda varía en el tiempo, de acuerdo con el gráfico de la Figura 2.1, considerando la premisa de que se deberá atender siempre la demanda trimestral, ante lo cual existe dos posibilidades:

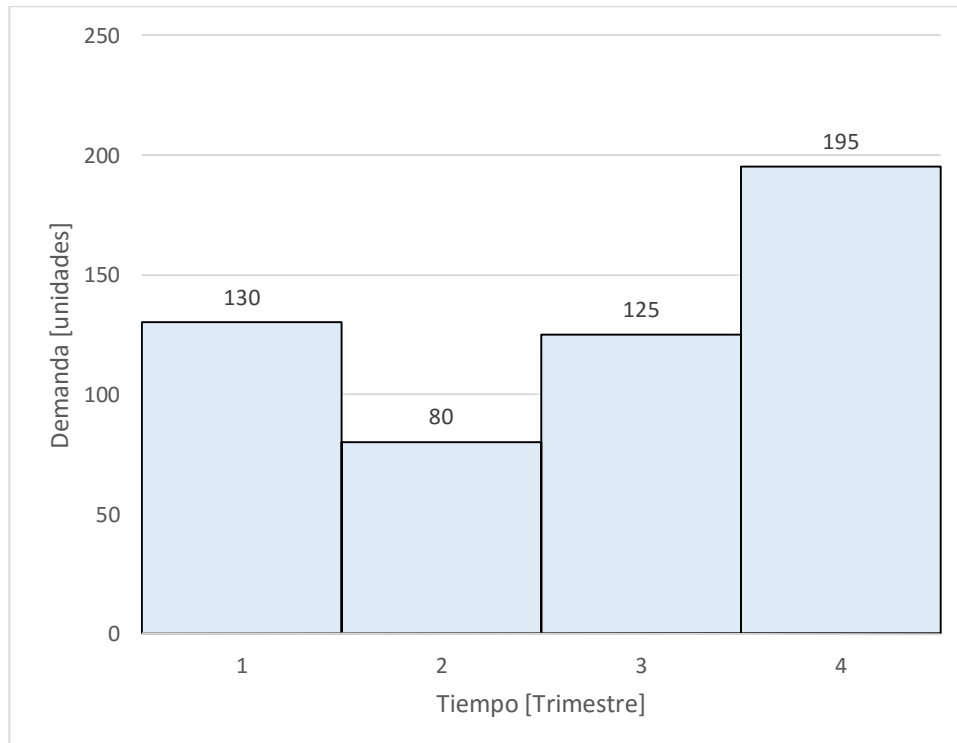


Gráfico 2.1: Demanda en función del tiempo

## **1. Producción variable.**

El fabricante puede producir cada trimestre el número exacto de unidades que le solicitan. Sin embargo, como una producción que varía es costosa de mantener, por los costes de horarios más largos en los trimestres de producción alta y los costes asociados al paro del personal y la maquinaria en los trimestres de producción baja; este tipo de producción no es eficiente.

## **2. Producción constante.**

El fabricante que debe atender una demanda que cambia con el tiempo puede producir por encima de dicho nivel en periodos de baja demanda y almacenar la sobreproducción para los periodos de demanda mayor. Así, la producción puede mantenerse constante, compensando la demanda alta con la sobreproducción de periodos pasados. Sin embargo, debido a los costes de almacenamiento, tal opción puede no ser deseable si se requiere costes altos de almacenamiento durante varios meses.

Por lo expuesto, los problemas de esta naturaleza ilustran las dificultades que surgen cuando objetivos contrarios están presentes en un sistema dado, por tal razón nuestro objetivo es llevar a cabo una planificación de la producción que minimice los costos totales después de considerar las variaciones en la producción y los almacenes, satisfaciendo la demanda a la vez <sup>[2]</sup>.



---

## **2.2 Generación de Variables Aleatorias a partir de una Distribución Empírica continua para Simulación.**

Si el modelador ha sido incapaz de identificar una distribución teórica que proporcione un buen modelo a los datos de entrada, entonces podría ser necesario usar una distribución empírica de datos. Esto es conocido como función de la distribución empírica, y tiene sentido para aquellos procesos donde la entrada es conocida al tomar un número finito de valores.

Por otro lado, si los datos se han extraído de un proceso con valores de entrada continuo, entonces la posibilidad de interpolar es posible entre los puntos de los datos observados <sup>[3]</sup>.

A continuación, se describe un método de definir y generar datos a partir de una distribución empírica continua

Para pequeñas muestras (tamaño  $n$ ):

- Ordene los datos desde los pequeños hasta los grandes.

$$x_{(1)} \leq x_{(2)} \leq \dots \dots \dots \leq x_{(n)}$$

- Obtener la función de distribución asignando la probabilidad  $1/n$  a cada intervalo  $x_{(i-1)} \leq x \leq x_{(i)}$ , es decir:

$$F(x) = \begin{cases} 0 & x < X_{(1)} \\ \frac{i-1}{n-1} + \frac{x - X_{(i)}}{(n-1)(X_{(i+1)} - X_{(i)})} & X_{(i)} \leq x < X_{(i+1)} \quad i = 1, \dots, n-1 \\ 1 & x \geq X_{(n)} \end{cases}$$

Como resultado de la transformada inversa de la función de distribución, se obtiene:

$$X = \hat{F}^{-1}(r) = x_{(i-1)} + a_i \left( r - \frac{(i-1)}{n} \right); \quad \frac{i-1}{n} < r \leq \frac{i}{n}$$

Donde 
$$a_i = \frac{x_{(i)} - x_{(i-1)}}{1/n}$$

Una versión computarizada del procedimiento descrito, será más ineficiente mientras se incremente el número de intervalos. Razón por la cual una versión sistemática computarizada es conocida como un esquema de generación de búsqueda, debido a que dado un valor de  $r$  – variable aleatoria-, el programa debe buscar el intervalo correspondiente para su evaluación final.

## **CAPÍTULO III**

### **3. DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO GENÉTICO Y LA DE UN MODELO MATEMÁTICO, PARA EL PROBLEMA DE LA PLANIFICACIÓN DE LA PRODUCCIÓN.**

#### **3.1 Formulación del Modelo de Programación Lineal.**

Remitiéndonos a que la Programación Matemática es una potente técnica de modelado, usada para la toma de decisiones, se debe cubrir las siguientes etapas que intervienen en el problema de la Planificación de la Producción, considerando los siguientes datos <sup>[2]</sup>:

- n: el número de trimestres a considerar.
- s<sub>0</sub>: la cantidad almacenada disponible al principio del periodo considerado.
- d<sub>t</sub>: el número de unidades (demanda) que se solicita en el trimestre t.
- x<sub>max</sub>: la capacidad máxima de producción.
- c<sub>i</sub>: el coste de almacenamiento en el trimestre t.
- c<sub>p</sub>: el coste de producción en el trimestre t.

1. La **primera etapa** consiste en identificar las posibles decisiones que pueden tomarse; esto lleva a identificar las *variables de decisión*. Normalmente estas variables son de carácter cuantitativo [4].

Para nuestro caso, tenemos:

**Variables de Decisión:**

$X_t$  : Número de unidades producidas en el trimestre t.

$S_t$  : Número de unidades almacenadas en el trimestre t.

2. La **segunda etapa** supone determinar qué decisiones resultan factibles; esto conduce a un conjunto de restricciones que se determinan teniendo presente la naturaleza del problema en cuestión [4].

**Restricciones.** Como la demanda  $d_t$  en el trimestre t debe coincidir con el cambio en el almacenamiento,  $s_{t-1} - s_t$ , más la producción  $x_t$  en el trimestre t; la capacidad de producción no puede excederse; y la demanda  $d_t$ , almacenamiento  $s_t$ , y producción  $x_t$  deben ser no negativas; se tienen las siguientes restricciones:

$$s_{t-1} + x_t - d_t = s_t; \quad t = 1, \dots, n$$

$$x_t \leq x_{max}; \quad t = 1, \dots, n$$

$$x_t, s_t \geq 0$$

3. En la **tercera etapa**, se calcula el coste/beneficio asociado a cada decisión factible; esto supone determinar una función objetivo que asigna, a cada conjunto de variables de decisión, un valor de coste/beneficio total [4].

***Función a optimizar.*** Una posibilidad en el problema de la planificación de la producción consiste en minimizar el costo total, es decir el costo de producir y el de almacenamiento, esto es:

$$\text{Min } Z = \sum_{t=1}^n (c_p x_t + c_i s_t)$$

---

## 3.2 Diseño y aplicación del Modelo Matemático al problema de Planificación de la Producción.

Para el modelado, análisis y resolución del problema de optimización planteado, usaremos la herramienta GAMS, cuyo lenguaje de programación coincide prácticamente con la descripción matemática del problema detallada anteriormente. Por tanto, el código a emplear usará la misma nomenclatura y su código fuente se encuentra detallado en el Anexo 1.

### 3.2.1 Codificación del problema de optimización en GAMS.

A continuación, se resume cada una de las partes esenciales que cubre el programa.:

**Cabecera del programa:** Se detalla el comentario descriptivo del programa, título, y autor, información <sup>[5]</sup>.

```
$Title Programación de la producción
$ontext
Codificación del Modelo matemático para determinar cuánto se debe producir del
producto y cuánto se debe almacenar en inventario por cada trimestre de forma »
tal
que los costos totales sean mínimos
Autor: Roberth Tinoco
$offtext
```

Gráfica 3.1: Cabecera del programan en GAMS.

---

**Opciones de Programación:** Opciones generales disponibles en GAMS, se activa únicamente la de criterio relativo de terminación<sup>[5]</sup>.

```
option  
optcr=0.00001;
```

**Declaración de conjuntos y parámetros numéricos:** Las variables, los parámetros y las ecuaciones descritos mediante subíndices, las cuales especifican los diversos conjuntos a tener en cuenta y los elementos que a ellos pertenecen<sup>[5]</sup>.

```
sets  
t periodo /t0*t4/
```

**parameter**

```
d(t) demanda de producto durante el trimestret t  
/t1 130, t2 80, t3 125, t4 195 /
```

```
cp(t) costo de produccion durante el trimestre t  
/t1 6, t2 4, t3 8, t4 9/
```

```
ci(t) costo de almacenamiento durante el trimestre t  
/t1 2, t2 1, t3 2.5, t4 3/
```

```
scalar xmax /150/;
```

Gráfica 3.2: Declaración de parámetros en GAMS.

**Declaración de Variables:** Aquí definiremos que variables vamos a emplear y de qué tipo es cada variable. Podremos también fijar los límites de las variables que queramos y especificar valores iniciales a considerar en los pases del optimizador [5].

**positive variables**

```
x(t)           Número de unidades a producir en el trimestre t,  
s(t)           Número de unidades a almacenar en el trimestre t ;  
s.fx('t0')=15;
```

Gráfica 3.3: Declaración de Variables en GAMS.

**Declaración de ecuaciones:** Parte donde se ha definido las ecuaciones del modelo [5].

**equations**

```
Costos          costos totales,  
produccion      capacidad productiva,  
inventario      cantidad almacenada;
```

```
Costos..          z =e= sum[t$(ord(t) gt 1), (x(t)*CP(t))+(s(t)*CI»  
(t))];  
produccion(t)$(ord(t) gt 1).. x(t) =l= m ;  
inventario(t)$(ord(t) gt 1).. s(t) =e= s(t-1) + x(t) - D(t);
```

Gráfica 3.4: Declaración de ecuaciones en GAMS.



**Declaración de modelo.** Consideramos dos modelos dentro del mismo programa GAMS para fines de presentar escenarios probables<sup>[5]</sup>.

```
model production1 /all/ ;  
model production2 /all/ ;
```

Gráfica 3.2: Declaración de modelos en GAMS.

**Lanzamiento del programa de optimización.** Se lanzará el programa que resuelve el caso una sola vez y paralelamente al otro modelo que hará varias llamadas al solver. Adicionalmente se incluirá el análisis de sensibilidad<sup>[5]</sup>.

```
production1.DICTFILE = 4;  
production1.OPTFILE = 1;  
  
solve production1 using LP minimizing Z  
solve production2 using LP minimizing Z
```

Gráfica 3. 3: Declaración de llamada de solver en GAMS

**Informes de resultados.** En este caso podemos optaremos por tener las instrucciones con la finalidad de preparar un informe en un archivo html. Este detalle se incluirá en el Anexo 1.

### 3.2.2 Resultados obtenidos.

Como parte de la codificación empleada para la generación de informes, a continuación, se ilustra los resultados obtenidos que tiene GAMS por defecto y el construido para generación en HTML.

```

              S O L V E      S U M M A R Y

MODEL  production1          OBJECTIVE  z
TYPE   LP                   DIRECTION MINIMIZE
SOLVER CPLEX                FROM LINE 61

**** SOLVER STATUS      1 Normal Completion
**** MODEL STATUS      1 Optimal
**** OBJECTIVE VALUE           3622.5000

RESOURCE USAGE, LIMIT      0.296      1000.000
ITERATION COUNT, LIMIT    0      2000000000

ILOG CPLEX      Aug 14, 2009 23.2.1 WIN 12361.12582 VIS x86/MS Windows
Cplex 12.1.0, GAMS Link 34
```

```
LP status(1): optimal
Optimal solution found.
Objective :      3622.500000
```

VARIABLE NAME	LOWER	CURRENT	UPPER
z	-INF	1	+INF
x(t1)	5	6	+INF
x(t2)	-INF	4	7
x(t3)	6.5	8	9
x(t4)	-INF	9	10.5
s(t0)	-INF	0	+INF
s(t1)	1	2	+INF
s(t2)	0	1	4
s(t3)	1	2.5	+INF
s(t4)	-10.5	3	+INF

Tabla 3.1: Resultados estándares del GAMS

## PROGRAMA DE PRODUCCION

Fecha:01/30/16

Hora:01:30:20

Elaborado por: Roberth Tinoco

Costo Total: \$ 3622.50

Stock Inicial: 15.00

TRIMESTRE	t1	t2	t3	t4
<b>Demanda</b>	130	80	125	195
<b>Producción</b>	115	150	100	150
<b>Stock</b>	0	70	45	0

## COSTOS VS CAPACIDAD DE PRODUCCION

CAPACIDAD	130.00	140.00	150.00	160.00	170.00	180.00	190.00
<b>Costo Total \$</b>	3722.50	3667.50	3622.50	3577.50	3532.50	3487.50	3442.50

Estrategia Make-to-order \$ : 3405.00

Tabla 3.2: Resultado del modelo en GAMS en archivo HTML

---

### **3.3 Diseño e Implementación del Algoritmo Genético para el problema de la Planificación de la Producción.**

A diferencia de los lenguajes de programación de propósito general tales como C, C++, Java, FORTRAN, Visual Basic, las herramientas con entornos de cálculo numérico o simbólico tales como hoja electrónica, MATLAB, MATHEMATICA, etc, tienen facilidad de uso y la de presentar resultados. Como Desventajas se incluye que no inducen una buena práctica de programación, y posee dificultad en la validación, mantenimiento del modelo<sup>[6]</sup>.

No obstante para fines de desarrollar el algoritmo en un lenguaje lo más simple, se usará la herramienta Mathematica.

#### **3.3.1 Conceptos Básicos del Algoritmo Genético.**

El algoritmo genético (AG) imita el proceso de evolución biológica de sobrevivencia del más apto". Cada solución factible de un problema se considera como un cromosoma codificado por un conjunto de genes. Los códigos genéticos más comunes son el binario y el numérico.

El conjunto de N soluciones factibles se conoce como población con N cromosomas y la aptitud de un cromosoma se mide en términos de una función objetivo apropiada. Un cromosoma más apto da un mejor valor a la función objetivo.

La idea general del AG es seleccionar dos padres a partir de una población. Los genes de los dos padres se cruzan entonces y (posiblemente) mutan para producir dos hijos. La descendencia reemplaza a los dos cromosomas más débiles (menos aptos) en la población, y el proceso de seleccionar nuevos padres se repite<sup>[7]</sup>.

La implementación real del AG requiere detalles adicionales del problema-específico. Asimismo, las reglas para seleccionar padres y crear hijos pueden variar, las cuales se detallará en el transcurso del presente capítulo.

### **3.3.2 Algoritmo Genético aplicado al Problema de la planificación de la Producción.**

Como hemos analizado, el problema de la planificación de la producción se enmarca dentro de un problema PL, razón por la cual una forma cómoda de representar las variables es utilizar una codificación numérica. Los cromosomas de la población inicial, pueden generarse al azar dentro de una región probablemente factible. En caso de que algún miembro de la población incurra como solución infactible, se deberá generar una nueva alternativa hasta cambiar tal condición.

La selección de los padres se determina uno al azar y el otro del menos costo, para posteriormente crear los dos hijos a partir del cruce en múltiples puntos. En caso de incurrir en una solución no factible, se deberá mutar hasta cambiar tal condición.

---

En la nueva población con la descendencia, se garantiza la inclusión de los más fuertes (más aptos) y forzar a dejar un grupo minoritario de los menos aptos para dar propiedad de salida a un entrapamiento de algún óptimo local. Posteriormente el proceso de seleccionar nuevos padres se repite en un número determinado de veces.

A continuación, se describe los pasos algorítmicos para la aplicación del AG, a sabiendas de que la única variable independiente, es las unidades a producir, para ello se utilizará las siguientes definiciones:

- $N$ : Tamaño de la población
- $S_{min}$ : Stock de Seguridad.
- $S_0$ : Inventario al inicio del periodo  $t$ .
- $X_{up}$ : Capacidad máxima de producción.
- $X_{lo}$ : Corrida mínima de producción.
- $N_{Max}$ : Número máximo de iteraciones
- $P_{mut}$ : Probabilidad de mutación.
- $X_{t_0}$  ( $X_1, X_2 \dots, X_N$ ): Solución factible encontrada durante la búsqueda de la Población Inicial, cuya matriz representa el número de unidades a producir en el periodo.
- $PbX_{t_0}$  ( $X_{t_1}, X_{t_2} \dots, X_{t_n}$ ): Población inicial de posibles soluciones.
- $\forall S \in t, S \geq S_{Min}$  Factibilidad asociada con la solución  $X_t$ . Asegurando que las unidades a almacenar serán mayor e igual al stock mínimo.
- Padre  $PbX_t$  ( $PbX_{t_1}, PbX_{t_2}$ ): Correspondiente a la solución con mejor valor objetivo, y otra establecida al azar.
- Hijos  $X_{t_0}$  ( $HijoX_{t_1}, HijoX_{t_2}$ ): La descendencia obtenida después del cruce para posterior validación de factibilidad.

- Hijos $X_t$  (Hijo $X_{t1}$ , Hijo $X_{t2}$ ): Descendencia factible, para los cual se debió mutar las veces necesarias en caso de una existencia previa de no factibilidad.
- Pbxti ( $X_{t1}$ ,  $X_{t2}$  ...,  $X_{tn}$ ): Nueva población después de reemplazar las soluciones con menor valor objetivo

Teniendo como pseudocódigo lo siguiente:

**Etap 1:** Generar una población aleatoria  $X_t$  de  $N$  cromosomas factibles.

1. Iteración **For**  $j = 1$  a  $N$
2.     **Determinar**  $X_t$  soluciones probables generadas al azar ( $X_{lo} \geq X_t \leq X_{up}$ )
3.     **For**  $i = 1$  a  $t$  (periodos)
4.         **Determinar** la factibilidad  $X_i$  asociada con el cromosoma  $i$
5.     **If**  $\forall S \in t, S \geq S_{Min}$  **then**
6.          $(X_i, S_i, Z_i) \rightarrow (X_{to}, S_{to}, Z_{to}) \in Pb$
7.     **Else:** Repetir 2

**Etap 2:** Para cada cromosoma  $X_i$  en la población seleccionada, evalúe su aptitud asociada. Registre  $Z^*$  como la mejor solución disponible hasta ahora.

8.     **If**  $z_j > z^*$  **then** establecer,  $z^* = z_j$ ,  $S^*=S_j$  y  $X^* = X_i$

**Etap 3:** Seleccione dos cromosomas padres de la población  $X$ .

9.     **Set**  $i^*$  como Padre 1 ( $PbX_{t1}$ ) y seleccionar al azar el Padre 2 ( $PbX_{t2}$ ) de entre  $\{1, 2, \dots, t\} - \{i^*\}$

---

**Etapa 4:** Cruce los genes padres para crear dos hijos y mute si no son factibles hasta lograr factibilidad.

10. **Crear** los Hijos 1 ( $HijoXt1$ ) y 2 ( $HijoXt2$ ) de los padres 1 y 2 utilizando el cruce.
11. **Determinar** la factibilidad  $HijoX_i$  asociada con el cromosoma  $i$
12. **If**  $\forall S \in t, S \geq S_{Min}$  **then**
13. **(HijoXt1, HijoXt2)  $\in$  HijosXt.**
14. **Else: Mutar** los genes de los hijos con probabilidad  $P_{mut}$  y Repita 11

**Etapa 5:** Actualice la población, reemplazando los peores padres con los dos hijos.

15. **Set**  $Xt \in Pb$
16. **For**  $r=1$  a 2
17. **Set**  $Xtr \in Pb$  como peor miembro de la Población
18. **Reemplazar**  $Xtr$  por **HijosXt.**

**Etapa 6:** Repita le proceso de escoger los padres y la de descendencia, hasta llegar a una condición de terminación.

19. Repetir paso 8
20. **If** el número de iteraciones =  $N_{Max}$  **then** deténgase y escoja  $Z_{min}$

Para un mayor detalle del código de programación empleado en Mathematica, pueden remitirse al Anexo 2.



### 3.3.3 Resultados Obtenidos.

Como parte de la codificación empleada para la generación de resultados, a continuación, se ilustra los resultados obtenidos considerando inicialmente la ejecución del algoritmo con una Población de 8 soluciones posibles y de 16 iteraciones máximas como condición de terminación, con una permutación del 50%.

Población Inicial							
{135, 131, 142, 143}	{131, 147, 146, 150}	{130, 121, 126, 144}	{144, 150, 130, 147}	{119, 148, 141, 129}	{128, 149, 118, 136}	{142, 133, 149, 100}	{139, 146, 146, 145}
{15, 20, 71, 88, 36}	{15, 16, 83, 104, 59}	{15, 15, 56, 57, 6}	{15, 29, 99, 104, 56}	{15, 4, 72, 88, 22}	{15, 13, 82, 75, 16}	{15, 27, 80, 104, 9}	{15, 24, 90, 111, 61}
4196.	4444.	3814.5	4412.	3961.	3875.5	3897.	4489.5

NIt	Producción (unidades)	Inventario (unidades)	Costos Total (\$)
0	{130, 121, 126, 144}	{15, 15, 56, 57, 6}	3814.5
1	{130, 121, 126, 144}	{15, 15, 56, 57, 6}	3814.5
2	{118, 136, 126, 144}	{15, 3, 59, 60, 9}	3798.
3	{130, 132, 118, 136}	{15, 15, 67, 60, 1}	3726.
4	{130, 132, 118, 136}	{15, 15, 67, 60, 1}	3726.
5	{130, 132, 118, 136}	{15, 15, 67, 60, 1}	3726.
6	{130, 132, 118, 136}	{15, 15, 67, 60, 1}	3726.
7	{130, 132, 118, 136}	{15, 15, 67, 60, 1}	3726.
8	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
9	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
10	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
11	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
12	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
13	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
14	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
15	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5
16	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5

Poblacion	N.Iteraciones	ProbPermut	Producción (unidades)	Inventario (unidades)	Costos Total (\$)
8	16	50	{125, 136, 118, 136}	{15, 10, 66, 59, 0}	3695.5

Tabla 4.1: Resultado del AG utilizando una Población de 18 e iteraciones 16

No obstante, para fines de análisis y revisión a continuación se resume los resultados para diferentes valores en lo que, respecto al tamaño de la Población, número máximo de iteraciones como condición de terminación y la probabilidad de la permutación. Asimismo la incorporación de un cronometro para establecer el performance respectivo

Poblacion	N.Iteraciones	ProbPermut	Producción (unidades)	Inventario (unidades)	Costos Total (\$)	Perfomance (segundos)
4	8	50	{129, 145, 97, 145}	{15, 14, 79, 51, 1}	3672.5	0.015625
4	12	50	{128, 148, 91, 148}	{15, 13, 81, 47, 0}	3644.5	0.03125
4	16	50	{127, 139, 139, 123}	{15, 12, 71, 85, 13}	3883.5	0.015625
4	32	50	{117, 140, 117, 142}	{15, 2, 62, 54, 1}	3680.	0.125
8	4	50	{127, 143, 103, 143}	{15, 12, 75, 53, 1}	3679.5	0.015625
8	8	50	{123, 140, 150, 104}	{15, 8, 68, 93, 2}	3756.5	0.015625
8	16	50	{137, 137, 101, 144}	{15, 22, 79, 55, 4}	3746.5	0.03125
8	32	50	{123, 135, 123, 134}	{15, 8, 63, 61, 0}	3699.5	0.03125
12	8	50	{119, 143, 119, 134}	{15, 4, 67, 61, 0}	3671.5	0.
12	12	50	{132, 140, 136, 108}	{15, 17, 77, 88, 1}	3746.	0.03125
12	36	50	{117, 143, 123, 134}	{15, 2, 65, 63, 2}	3696.5	0.046875
16	12	50	{127, 131, 108, 149}	{15, 12, 63, 46, 0}	3693.	0.
16	24	50	{120, 140, 116, 140}	{15, 5, 65, 56, 1}	3686.	0.046875
24	12	50	{125, 146, 121, 123}	{15, 10, 76, 72, 0}	3685.	0.03125
24	24	50	{131, 141, 119, 124}	{15, 16, 77, 71, 0}	3704.5	0.0625
64	12	50	{144, 147, 97, 127}	{15, 29, 96, 68, 0}	3695.	0.046875
64	36	50	{129, 148, 105, 134}	{15, 14, 82, 62, 1}	3680.	0.125
64	64	50	{117, 147, 123, 128}	{15, 2, 69, 67, 0}	3666.5	0.1875
64	128	50	{123, 140, 113, 139}	{15, 8, 68, 56, 0}	3677.	0.4375
128	64	50	{121, 135, 127, 133}	{15, 6, 61, 63, 1}	3712.5	0.375
128	128	50	{117, 147, 117, 134}	{15, 2, 69, 61, 0}	3657.5	0.765625
252	64	50	{125, 149, 96, 145}	{15, 10, 79, 50, 0}	3643.	1.03125
252	128	50	{134, 150, 89, 142}	{15, 19, 89, 53, 0}	3653.5	2.26563
252	252	50	{146, 148, 75, 146}	{15, 31, 99, 49, 0}	3665.5	4.57813
512	128	50	{121, 147, 101, 147}	{15, 6, 73, 49, 1}	3655.5	7.54688
512	256	50	{121, 149, 96, 149}	{15, 6, 75, 46, 0}	3633.	14.625
512	512	50	{120, 150, 124, 121}	{15, 5, 75, 74, 0}	3671.	29.7344

Tabla 4.2: Resultados del AG con diversas parametrizaciones.

El valor más bajo obtenido con la implementación del algoritmo es de 3633.0, el mismo que está por encima del 0.3% del valor optimo (3622.5) aproximadamente. No obstante, podemos percatarnos que el “esfuerzo computacional” con el incremento de la población y/o el número de iteraciones, no necesariamente garantiza un mejor valor y si se lo hace, no supera el 1% con el mejor valor obtenido con el algoritmo.

## CAPÍTULO IV

### 4. SIMULACIÓN DE TOMA DE PEDIDOS A UNA BODEGA A PARTIR DE UNA DISTRIBUCIÓN EMPÍRICA CONTINUA.

#### 4.1 Construcción de la Función de Distribución Continua Empírica.

Resulta aconsejable el empleo, en primer lugar, de las distribuciones teóricas convencionales y si ninguna de ellas describe adecuadamente el comportamiento del proceso, entonces deberemos, necesariamente, recurrir a distribuciones empíricas, como es la de nuestro caso.

Se procede a establecer la función de distribución según la definición realizada, obteniendo:

<b>i</b>	<b>Número de Unidades</b>	<b>Frecuencia (Número de Pedidos)</b>	<b>Frecuencia Relativa</b>	<b>Frecuencia Acumulativa, <math>c_i</math></b>
1	$50 < X \leq 60$	3	0,06	0,06
2	$60 < X \leq 70$	7	0,14	0,20
3	$70 < X \leq 80$	18	0,36	0,56
4	$80 < X \leq 90$	12	0,24	0,80
5	$90 < X \leq 100$	8	0,16	0,96
6	$100 < X \leq 110$	2	0,04	1,00

$$F(x) = \begin{cases} 0; & x < X_{(1)} \\ 0.06 \frac{(x - 50)}{10}; & 50 < x \leq 60 \\ 0,06 + 0.14 \frac{(x - 60)}{10}; & 60 < x \leq 70 \\ 0,20 + 0.36 \frac{(x - 70)}{10}; & 70 < x \leq 80 \\ 0,56 + 0.24 \frac{(x - 80)}{10}; & 80 < x \leq 90 \\ 0.80 + 0.16 \frac{(x - 90)}{10}; & 90 < x \leq 100 \\ 0.96 + 0.04 \frac{(x - 100)}{10}; & 100 < x \leq 110 \\ 0; & x < X_{(n)} \end{cases}$$

Aplicando la transformada inversa, obtenemos la siguiente expresión:

$$X = \hat{F}^{-1}(r) = \begin{cases} 166.67r + 50; & 0 \leq r \leq 0.06 \\ 71.43(r - 0.06) + 60; & 0.06 < r \leq 0.2 \\ 27.78(r - 0.2) + 70; & 0.2 < r \leq 0.56 \\ 41.67(r - 0.56) + 80; & 0.56 < r \leq 0.8 \\ 62.5(r - 0.8) + 90; & 0.8 < r \leq 0.96 \\ 250(r - 0.96) + 100; & 0.96 < r \leq 1. \end{cases}$$

## 4.2 Formulación en MATHEMATICA para la Simulación.

Realizado el análisis matemático, procedemos a describir la última función en el lenguaje MATHEMATICA, de la siguiente forma;

```
f[r_] :=  
  
Piecewise[{{166.67*r + 50, Inequality[0.0, LessEqual, r, LessEqual, 0.06]},  
[función a trozos [desigualdad [menor o igual [menor o igual  
{71.43*(r - 0.06) + 60, Inequality[0.06, Less, r, LessEqual, 0.2]},  
[desigualdad [menor [menor o igual  
{27.78*(r - 0.2) + 70, Inequality[0.2, Less, r, LessEqual, 0.56]},  
[desigualdad [menor [menor o igual  
{41.67*(r - 0.56) + 80, Inequality[0.56, Less, r, LessEqual, 0.8]},  
[desigualdad [menor [menor o igual  
{62.5*(r - 0.8) + 90, Inequality[0.8, Less, r, LessEqual, 0.96]},  
[desigualdad [menor [menor o igual  
{250*(r - 0.96) + 100, Inequality[0.96, Less, r, LessEqual, 1.]}]};
```

Para fines de poder evaluar la función antes expuesta, y obtención de los resultados en forma de lista procedemos con una longitud igual al número de pedidos deseado para la simulación, describimos el siguiente código.

```
NPed = 10;  
SimPed[n_] := Table[{i, f[Random[]]}, {i, n}];  
[tabla [aleatorio  
S = Round[SimPed[NPed]];  
[entero más próximo
```

### 4.2.1 Resultados obtenidos.

Como parte de la codificación empleada, a continuación, se ilustra el resultado de la simulación de diez pedidos descrita en la matriz S.

```
Text@Grid[Prepend[S, {"# Simulacion", "N.Pedido"}], Background -> {None, {Lighter[Yellow, .9],
[texto [rejilla [añade al principio [fondo de imagen [ninguno [más claro [amarillo
{White, Lighter[Blend[{Blue, Green}], .8]}}], Dividers -> {{Darker[Gray, .6],
[blanco [más claro [mezcla c- [azul [verde [separadores [más osc- [gris
{Lighter[Gray, .5]}, Darker[Gray, .6]}, {Darker[Gray, .6], Darker[Gray, .6], {False},
[más claro [gris [más osc- [gris [más osc- [gris [más osc- [gris [falso
Darker[Gray, .6]}], Alignment -> {{Center, Center, {Center}}}, ItemSize -> {{8, 7}},
[más osc- [gris [alineamiento [centro [centro [centro [tamaño de ítem
Frame -> Darker[Gray, .6], ItemStyle -> 14, Spacings -> {Automatic, .8}]
[marco [más osc- [gris [estilo de ítem [espaciados [automático
```

Out[41]=

# Simulacion	N.Pedido
1	78
2	62
3	77
4	87
5	69
6	109
7	98
8	60
9	99
10	56

## **CAPÍTULO V**

### **5 CONCLUSIONES Y RECOMENDACIONES.**

#### **5.1 Conclusiones**

Luego de analizado y revisado los resultados obtenidos, podemos inferir en las siguientes conclusiones:

- Empleando la codificación óptima del modelo matemático empleando GAMS para el problema de Planificación de la producción, se obtiene un costo óptimo de \$ 3622.50, mientras que, con el empleo de la heurística genética, las soluciones son aproximadas a este valor. En el mejor de los casos superior al 0.3% de costo óptimo.
- La heurística está diseñada para encontrar buenas soluciones, y en el mejor de los casos se próxima al valor óptimo, siempre y cuando se incremente el tamaño de la población o en su defecto también con se aumente el número de iteraciones máximas, aunque en este último punto las probabilidades de mejora son menores. La afectación del performance, no se justifica con lograr un mejor valor de aproximadamente el 1%.
- El modelo empleado en GAMS, permite analizar diversos escenarios, uno de ellos es el beneficio que acarrea aumenta capacidad de producción, sin incidencia en gran medida de incurrir a capacidad no aprovechada.

- Empleando la función de distribución empírica continua, para la generación de variables, se puede fácilmente realizar una simulación que ilustra el proceso de la toma de pedidos a través de la utilización de MATHEMATICA.

## **5.2 Recomendaciones**

Es recomendable las siguientes consideraciones:

- Por el tamaño y complejidad del problema, usar un lenguaje algebraico de modelado como GAMS, es lo más apropiado, debido a que permiten cambiar sin dificultad las dimensiones del modelo. Desde el punto de vista del modelador permiten la detección de errores de consistencia en la definición y verificación del modelo, mientras que desde el punto de vista del usuario simplifican su mantenimiento.
- Incurrir a lenguajes o entornos de cálculo numérico o simbólico como los es MATHEMATICA, presta facilidades de uso y de presentar resultados. Sin embargo, no inducen una buena práctica de programación, debido a la dificultad en la validación y mantenimiento del modelo, a pesar de nuestros esfuerzos de simplificar el modelo y convertirlo en una estándar para problemas similares.
- Se puede mejorar la codificación empleada en MATHEMATICA, para que incluya desde el establecimiento de la función de distribución continua empírica a partir de los intervalos y frecuencias relativas.



## **BIBLIOGRAFIA**

- [1] Chase, Jacobs, Aquilano. Administración de las Producción y Operaciones, 10ª Edición. McGraw-Hill, México D.F. Febrero del 2007. Páginas 574-579.
- [2] Castillo Enrique, Conejo Antonio, Pedregal Pablo, García Ricardo y Alguacil Natalia. Formulación y Resolución de Modelos de Programación Matemática en Ingeniería y Ciencia., Primera Edición, España. Febrero del 2002, Páginas 6-10.
- [3] Jerry B, Carson J. Nelson L., Nicol D. Discrete Event System Simulation. Fourth edition Pearson Prentice Hill 2004. Páginas. 273 – 278.
- [4] Tomado de los apuntes de la Materia: “Programación Matemática.”, IV Promoción Maestría en Control de Operaciones y Gestión Logística, Fernando Sandoya PhD. 2012.
- [5] Marín Alberdi José Ignacio. Introducción al lenguaje GAMS. Año 2002. Páginas 5-13.
- [6] Ramón A. Sanchez P. Ferrer J. Barquín J. Linares P. Modelos Matemáticos de Optimización. Universidad Pontificia Comillas. Septiembre 2010. Páginas 17
- [7] Taha H. Investigación de Operaciones. Pearson Novena Edición. Año 2012. Páginas 371-390.

## **ANEXOS**

## CÓDIFICACION OPTIMA DEL MODELO MATEMATICO EN GAMS, PARA EL PROBLEMA DE PLANIFICACION DE LA PRODUCCION.

```

1 $Title Programación de la producción
2
3 $ontext
4
5 Codificación del Modelo matemático para determinar cuánto se debe producir del
6 producto y cuánto se debe almacenar en inventario por cada trimestre de forma tal
7 que los costos totales sean mínimos
8
9 Autor: Roberth Tinoco
10
11 $offtext
12
13
14 option
15 optcr=0.00001;
16
17 sets
18 t periodo /t0*t4/
19
20 parameter
21
22 d(t) demanda de producto durante el trimestret t
23 /t1 130, t2 80, t3 125, t4 195 /
24
25 cp(t) costo de produccion durante el trimestre t
26
27 /t1 6, t2 4, t3 8, t4 9/
28
29 ci(t) costo de almacenamiento durante el trimestre t
30
31 /t1 2, t2 1, t3 2.5, t4 3/
32
33 scalar xmax /150/;
34
35 option limrow=0;
36 option limcol=0
37
38 variables
39 z;
40
41 positive variables
42 x(t) Número de unidades a producir en el trimestre t,
43 s(t) Número de unidades a almacenar en el trimestre t ;
44 s.fx('t0')=15;
45
46 equations
47 Costos costos totales,
48 produccion capacidad productiva,
49 inventario cantidad almacenada;
50
51
52 Costos.. z =e= sum[t$(ord(t) gt 1), (x(t)*CP(t))+(s(t)*CI(t))];
53 produccion[t]$(ord(t) gt 1).. x(t) =I= m ;
54 inventario[t]$(ord(t) gt 1).. s(t) =e= s(t-1) + x(t) - D(t);
55
56
57 model production1 /all/ ;
58 model production2 /all/ ;
59
60 production1.DICTFILE = 4;
61 production1.OPTFILE = 1;
62
63 solve production1 using LP minimizing Z
64 solve production2 using LP minimizing Z
65
66 display x.l, s.l;
67
68 file html /'Programacion.html'/;
69 put html;
70
71 put '<H1> PROGRAMA DE PRODUCCION </H1>';
72 put "Fecha:" system.date/
73 put '<p>';
74 put "Hora:" system.time/
75 put '<p>';
76 put 'Elaborado por: Roberth Tinoco';
77 put '<p>';
78 put 'Costo Total: $', z.l/
79 put '<p>';
80
81 put 'Stock Inicial: ', s.l['t0']/
82 put '<p>';
83 put '<p>';
84
85 put '<table border="1" cellpadding="8" cellspacing="2">';
86 put '<tr><th>';
87 put 'TRIMESTRE';
88 loop(t$(ord(t) gt 1), put '<th bgcolor=orange>,t.t1</th>');
89 put '</th></tr>';
90 put '<tr><th>';
91 put 'Demanda';
92 loop(t$(ord(t) gt 1), put '<th>,d(t):0:0,</th>');
93 put '</th></tr>';
94 put '<tr><th>';
95 put 'Producción';
96 loop(t$(ord(t) gt 1), put '<th>,x.l(t):0:0,</th>');
97 put '</th></tr>';
98 put '<tr><th>';
99 put 'Stock';
100 loop(t$(ord(t) gt 1), put '<th>,s.l(t):0:0,</th>');
101 put '</th></tr>';
102
103 put '</table>';
104
105
106 set k /1*7/;
107 parameter i(k)
108 /1 130
109 2 140
110 3 150
111 4 160
112 5 170
113 6 180
114 7 190/;
115 parameter salida(*,k)
116 option solprint = off;
117 loop (k, xmax = i(k);
118 SOLVE production2 using LP minimizing Z ;
119 salida("capacidad de produccion",k)=xmax;
120 salida("costos ",k)=z.l) ;
121
122
123 put html;
124
125 put '<H2> COSTOS VS CAPACIDAD DE PRODUCCION </H2>';
126
127 put '<p>';
128
129 put '<table border="1" cellpadding="8" cellspacing="2">';
130 put '<tr><th>';
131 put 'CAPACIDAD';
132 loop(k, put '<th bgcolor=orange>,i(k)</th>');
133 put '</th></tr>';
134 put '<tr><th>';
135 put 'Costo Total $';
136 loop(k, put '<th>,salida("costos ",k)</th>');
137 put '</th></tr>';
138
139 put '</table>';
140
141 put '<p>';
142 put '<H4> Estrategia Make-to-order $ :';
143 put sum[t, ((d(t)*CP(t))-(s.l['t0']*CP('t1')))] ;
144 put '</H4>';
145
146
147
148
149 putclose;
150 execute =shellexecute Programacion.html';
151

```

---

## **CODIGO FUENTE DEL ALGORITMO GENETICO EMPLEADO EN MATHEMATICA PARA SOLUCION DEL PROBLEMA DE PLANIFICACION.**

### **Algoritmo Genético**

#### **Problema de la Planificación de la Producción.**

Una fábrica ecuatoriana ha dividido la producción anual de uno de sus productos principales en 4 trimestres ( $t$ ). En cada periodo, la empresa tiene una limitante de producción para dicho producto de 150 unidades ( $x_{up}$ ), debido a la capacidad de su planta y a los recursos de los que dispone. Se tiene un inventario inicial de 15 unidades del producto ( $S_0$ ).

Considere la siguiente tabla con información de demanda ( $dt$ ), costos de producción ( $cp$ ) y costos de inventario ( $ci$ ):

Datos:

- N: Tamaño de la población
- Smin: Stock de Seguridad.
- S<sub>0</sub>: Inventario al inicio del periodo  $t$ .
- X<sub>up</sub>: Capacidad máxima de producción.
- X<sub>lo</sub>: Corrida mínima de producción.
- NMax: Número máximo de iteraciones
- Pmut: Probabilidad de mutación.

456]:=

```
dt = {130, 80, 125, 195};  
cp = {6, 4, 8, 9};  
ci = {2, 1, 2.5, 3};  
Smin = 0;  
S0 = 15;  
n = 4;  
xup = 150;  
xlo = 0;  
CostoTotalPbzto = 0;  
NMax = 12;  
Pmutac = 50;
```

**Etapas I: Generar una población aleatoria  $X_t$  de  $N$  cromosomas factibles.**

1. Iteración For  $j = 1$  a  $N$
2.     Determinar  $X_t$  soluciones probables generadas al azar ( $X_{lo} \geq X_t \leq X_{up}$ )
3.     For  $i = 1$  a  $t$  (periodos)
4.     Determinar la factibilidad  $X_i$  asociada con el cromosoma  $i$
5.     If  $\forall S \in t, S \geq S_{Min}$  then
6.      $(X_i, S_i, Z_i) \rightarrow (X_{to}, S_{to}, Z_{to}) \in Pb$
7.     Else: Repetir 2

486];=

```
Pbxto = Pbsto = Pbzto = XSZti = List[] ;  
                                     |lista
```

```
For[k = 1, k ≤ n, k++,  
  |para cada  
    xto = Table[0, {4}] ;  
        |tabla  
    sto = Table[0, {5}] ;  
        |tabla  
    sto[[1]] = So;
```

```
While[True,  
  |mientras [verdadero  
    xtoi = List[] ;  
        |lista  
    stoi = Table[0, 5] ;  
        |tabla  
    stoi[[1]] = So;
```

```
For[j = 1, j ≤ 4, j++,  
  |para cada
```

```
    xtoi = Append[xtoi, RandomInteger[{xlo, xup}]];
           [añade           [entero aleatorio]
    stoi[[j + 1]] = xtoi[[j]] + stoi[[j]] - dt[[j]];

];

If[AllTrue[stoi, # >= Smin &] && stoi[[5]] >= Smin, Break[]];
[si [todos verdaderos]                               [finaliza iteraciór]

];
For[i = 1, i <= 4, i++,
[para cada
    xto[[i]] = xtoi[[i]];
    sto[[i + 1]] = sto[[i]] + xto[[i]] - dt[[i]];

];

Pbxto = Append[Pbxto, xto];
           [añade
Pbsto = Append[Pbsto, sto];
           [añade

For[m = 1, m <= 4, m++,
[para cada
    CostoTotalPbzto = CostoTotalPbzto +
        cp[[m]] * Pbxto[[k]][[m]] + 0 +
        ci[[m]] * Pbsto[[k]][[m + 1]];
];

Pbzto = Append[Pbzto, CostoTotalPbzto];
           [añade
CostoTotalPbzto = 0;
];
```

Etapa 2 : Para cada cromosoma  $X_i$  en la población seleccionada, evalúe su aptitud asociada. Registre  $Z^*$  como la mejor solución disponible hasta ahora

8. If  $z_j > z^*$  then establecer,  $z^* = z_j$ ,  $S^* = S_j$  y  $X^* = X_i$

```
489]:= Pbzt = Sort[Pbzto, Less];
           |ordena      |menor
Pbxt = Pbst = List[];
           |lista
For[pi = 1, pi ≤ n, pi++,
  |para cada

  Pbztp = Position[Pbzto, Pbzt[[pi]]];
           |posición
  Pbxtp = Extract[Pbxto, Pbztp];
           |extrae
  Pbstp = Extract[Pbsto, Pbztp];
           |extrae
  Pbxt = Join[Pbxt, Pbxtp];
           |junta
  Pbst = Join[Pbst, Pbstp];
           |junta
];

iter = 0;
XSZti = List[];
           |lista
XSZti =
  Append[XSZti, {iter, Pbxt[[1]], Pbst[[1]], Pbzt[[1]]}];
           |añade
```

Etapa 3: Seleccione dos cromosomas padres de la población X.

Etapa 4: Cruce los genes padres para crear dos hijos y mute si no son factibles hasta lograr factibilidad.

Etapa 5: Actualice la población, reemplazando los peores padres con los dos hijos.

Etapa 6: Repita le proceso de escoger los padres y la de descendencia, hasta llegar a una condición de terminación

9. Set  $i^*$  como Padre 1 (PbXt1) y seleccionar al azar el Padre 2 (PbXt2) de entre  $\{1, 2, \dots, t\} - \{i^*\}$

10. Crear los Hijos 1 (HijoXt1) y 2 (HijoXt2) de los padres 1 y 2 utilizando el cruce.

11. Determinar la factibilidad HijoXti asociada con el cromosoma i

12. If  $\forall S \in t, S \geq S_{Min}$  then

13. (HijoXt1, HijoXt2)  $\in$  HijosXt.

14. Else: Mutar los genes de los hijos con probabilidad Pmut y

Repita 11

15. Set Xt  $\in$  Pb

16. For r=1 a 2

17. Set Xtr  $\in$  Pb como peor miembro de la Población

18. Reemplazar Xtr por HijosXt.

19. Repetir paso 8

20. If el número de iteraciones = NMax then deténgase y escoja Zmin

496]:=

```
For[ww = 1, ww ≤ NMax, ww++,
```

```
  |para cada
```

```
  mz1 = Position[Pbzt, Pbzt[[1]]];
```

```
  |posición
```

```
  mz2 = Position[Pbzt, Pbzt[[RandomInteger[{2, n}]]]];
```

```
  |posición
```

```
  |entero aleatorio
```

```
  PadrePbxt = Join[Extract[Pbxt, mz1], Extract[Pbxt, mz2]];
```

```
  |junto |extrae
```

```
  |extrae
```



```
PadrePbst = Join[Extract[Pbst, mz1], Extract[Pbst, mz2]];
           junta extrae           extrae

PadrePbzt = Join[Extract[Pbzt, mz1], Extract[Pbzt, mz2]];
           junta extrae           extrae

PadreXt1 = PadrePbxt[[1]];
PadreXt2 = PadrePbxt[[2]];
HijoXt1 = ReplacePart[PadreXt1,
                    sustituye una parte
                    {1 -> PadreXt2[[3]], 2 -> PadreXt2[[4]]}];
HijoXt2 = ReplacePart[PadreXt2,
                    sustituye una parte
                    {3 -> PadreXt1[[1]], 4 -> PadreXt1[[2]]}];

HijosXto = Append[{HijoXt1}, HijoXt2];
           añade

HijosXt = HijosSt = HijosZt = List[];
           lista

CostoTotalHijosZt = 0;

For[kk = 1, kk ≤ 2, kk++,
  para cada
  HXt = Table[0, {4}];
      tabla
  HSt = Table[0, {5}];
      tabla
  HSt[[1]] = So;

  HSti = Table[0, 5];
      tabla
  HSti[[1]] = So;
  HXti = {};
```

```
While[True,
  [mien· ·verdadero

  For[ji = 1, ji ≤ 4, ji++,
    [para cada

      tt1 = HijosXto[[kk]][[ji]];
      HXti = Append[HXti, tt1];
        [añade

      HSti[[ji + 1]] = HXti[[ji]] + HSti[[ji]] - dt[[ji]]

    ];

  If[AllTrue[HSti, # ≥ Smin &] && HSti[[5]] ≥ Smin, Break[],
    [si [todos verdaderos] [finaliza iterac

      While[True,
        [mien· ·verdadero

          HXti = {};
          HStii = Table[0, 5];
            [tabla

          HStii[[1]] = So;
          rr = Table[RandomInteger[{0, 100}], 4];
            [tabla [entero aleatorio

          For[l = 1, l ≤ 4, l++,
            [para cada

              tt2 = If[rr[[1]] ≤ Pmutac, RandomInteger[{xlo, xup}],
                [si [entero aleatorio

                HijosXto[[kk]][[1]]];
                  HXti = Append[HXti, tt2];
                    [añade

              HStii[[l + 1]] = HXti[[l]] + HStii[[l]] - dt[[l]];

            ];

    ];
```

```
        If[AllTrue[HStii, # >= 0 &] && HStii[[5]] >= 0,
        |si |todos verdaderos
        Break[]]
        |finaliza iteración
    ];

    If[AllTrue[HStii, # >= Smin &] && HStii[[5]] >= Smin,
    |si |todos verdaderos
    Break[]]
    |finaliza iteración
];

For[ii = 1, ii <= 4, ii++,
|para cada
    HXt[[ii]] = HXti[[ii]];
    HSt[[ii + 1]] = HSt[[ii]] + HXt[[ii]] - dt[[ii]];
];

HijosXt = Append[HijosXt, HXt];
        |añade
HijosSt = Append[HijosSt, HSt];
        |añade

For[mm = 1, mm <= 4, mm++,
|para cada
    CostoTotalHijosZt = CostoTotalHijosZt +
        cp[[mm]] * HijosXt[[kk]][[mm]] +
        ci[[mm]] * HijosSt[[kk]][[mm + 1]];
];

HijosZt = Append[HijosZt, CostoTotalHijosZt];
        |añade
```

```
CostoTotalHijosZt = 0;
];

PbxtHijosXt = PbstHijosSt = PbztHijosZt = List[];
|lista

Pbxti = Pbsti = Pbxtt = Pbstt = List[];
|lista

(*Los peores padres son reemplazados por los hijos *)

PbxtHijosXt = ReplacePart[Pbxt,
|sustituye una parte
{n - 1 → HijosXt[[1]], n → HijosXt[[2]]}];
PbstHijosSt = ReplacePart[Pbst,
|sustituye una parte
{n - 1 → HijosSt[[1]], n → HijosSt[[2]]}];
PbztHijosZt = ReplacePart[Pbzt,
|sustituye una parte
{n - 1 → HijosZt[[1]], n → HijosZt[[2]]}];

Pbzti = Join[TakeSmallest[PbztHijosZt, n]];
|junta |toma los menores

For[ti = 1, ti ≤ n, ti++,
|para cada

PPbzti = Position[PbztHijosZt, Pbzti[[ti]]];
|posición

Pbxtt = Extract[PbxtHijosXt, PPbzti];
|extraer
```

12 | AG PProducción RTinoco 1.nb

```
    Pbstt = Extract[PbstHijosSt, PPbzt];
    Pbxti = Join[Pbxti, Pbxtt];
    Pbsti = Join[Pbsti, Pbstt];

];

Pbxt = Pbxti;
Pbst = Pbsti;
Pbzt = Pbzt;
PadrePbxt = PadrePbst = PadrePbzt = List[];

PadreXt1 = PadreXt2 = List[];

HijoXt1 = HijoXt2 = List[];

HijoXt2 = HijosXto = List[];

iter = iter + 1;

XSZti = Append[XSZti, {iter, Pbxti[[1]], Pbsti[[1]],
    Pbzt[[1]]}];

];
```

---

---

## Resultados obtenidos

```
497]:= Summarize = {}
Summarize = Append[Summarize,
  |añade
  {n, NMax, Pmutac, Pbxti[[1]], Pbsti[[1]], Pbzti[[1]]}];

Text@
|texto
Grid[Prepend[{Pbxti, Pbsti, Pbzti}, {"Población Inicial"}],
|rejilla |añade al principio
Background →
|fondo de imagen
{None, {Lighter[Yellow, .9],
|ning... |más clarc |amarillo
  {White, Lighter[Blend[{Blue, Green}], .8]}}},
|blanco |más clarc |mezcl... |azul |verde
Dividers →
|separadores
{{Darker[Gray, .6], {Lighter[Gray, .5]}, Darker[Gray, .6]},
|más o... |gris |más clarc |gris |más o... |gris
  {Darker[Gray, .6], Darker[Gray, .6], {False},
|más o... |gris |más o... |gris |falso
  Darker[Gray, .6]}}, Alignment → {{Left, Right, {Left}}},
|más o... |gris |alineamiento |izqu... |derecha |izquierda
ItemSize → {{10, 10, 10, 10}}, Frame → Darker[Gray, .6],
|tamaño de ítem |marco |más o... |gris
ItemStyle → 14, Spacings → {Automatic, .8}
|estilo de ítem |espaciados |automático
```

```
Text@
|texto
```

```

Grid[Prepend[XSZti, {"Nit", "Producción (unidades)",
rejilla añade al principio
    "Inventario (unidades)", "Costos Total ($)"}],
Background →
fondo de imagen
{None, {Lighter[Yellow, .9],
ning... más clarc amarillo
    {White, Lighter[Blend[{Blue, Green}], .8]}}},
blanco más clarc mezcl... azul verde
Dividers →
separadores
{{Darker[Gray, .6], {Lighter[Gray, .5]}, Darker[Gray, .6]},
más o... gris más clarc gris más o... gris
{Darker[Gray, .6], Darker[Gray, .6], {False},
más o... gris más o... gris falso
    Darker[Gray, .6]}}, Alignment → {{Left, Right, {Left}}},
más o... gris alineamiento izqu... derecha izquierda
ItemSize → {{2, 11, 10, 8}}, Frame → Darker[Gray, .6],
tamaño de ítem marco más o... gris
ItemStyle → 14, Spacings → {Automatic, .8}
estilo de ítem espaciados automático

Text@
texto
Grid[Prepend[Summarize,
rejilla añade al principio
    {"Poblacion", "N.Iteraciones", "ProbPermut",
    "Producción (unidades)", "Inventario (unidades)",
    "Costos Total ($)"}],
Background →
fondo de imagen
{None, {Lighter[Yellow, .9],
ning... más clarc amarillo
    {White, Lighter[Blend[{Blue, Green}], .8]}}},
blanco más clarc mezcl... azul verde
Dividers →
separadores
{{Darker[Gray, .6], {Lighter[Gray, .5]}, Darker[Gray, .6]},
más o... gris más clarc gris más o... gris
{Darker[Gray, .6], Darker[Gray, .6], {False},
más o... gris más o... gris falso
    Darker[Gray, .6]}}, Alignment → {{Left, Right, {Left}}},
más o... gris alineamiento izqu... derecha izquierda
ItemSize → {{5, 6, 6, 12, 10, 9}}, Frame → Darker[Gray, .6],
tamaño de ítem marco más o... gris
ItemStyle → 14, Spacings → {Automatic, .8}
estilo de ítem espaciados automático

```