UNL - FICH - Departamento de Informática - Ingeniería Informática

Inteligencia Computacional

Computación Evolutiva
Algoritmos Genéticos

Temas a tratar

- Generalidades de los algoritmos de computación evolutiva.
- Diseño de la solución de un problema mediante algoritmos genéticos.
- Aplicación de la técnica de algoritmos genéticos.

Objetivos

- Aprender los fundamentos de la técnica.
- Aplicar las técnicas de computación evolutiva a problemas reales.
- Comprender la potencialidad de la técnica y sus limitaciones más importantes.
- Implementar un algoritmo genético completo.
- Comparar a los algoritmos genéticos con otras técnicas de optimización y búsqueda de soluciones.

1. Introducción

Los algoritmos genéticos (AGs) junto con las estrategias evolutivas y la programación evolutiva constituyen un nuevo enfoque a la solución de numerosos problemas de optimización y búsqueda. Esta técnica está en el contexto de una nueva ciencia denominada *computación evolutiva*. El campo de las aplicaciones de la computación evolutiva no reconoce fronteras. Es posible resolver problemas de las características más diversas.

La analogía en que se basan los AGs estriba en reconocer el mecanismo esencial del proceso evolutivo en la naturaleza. Los componentes fundamentales de éste mecanismo son los cromosomas, el material genético de un individuo biológico, que determinan sus características únicas. Los cambios en el material genético de las especies permiten el proceso de adaptación.

Las fuerzas que subyacen al proceso evolutivo son: la selección natural, la recombinación de material genético y la mutación, fenómenos que se presentan durante la reproducción de las especies. La competencia entre los individuos por los recursos naturales limitados y por la posibilidad de procreación o reproducción permite que sólo los más fuertes o más adaptados sobrevivan. Esto significa que el material genético de los mejores individuos sobrevive y se reproduce, mientras que los genes de los individuos más débiles o menos adaptados, mueren o se extinguen. Y de esta forma la naturaleza optimiza la solución al problema de la vida.

Se propone a los AGs como una técnica computacional que intenta *imitar* el proceso evolutivo de la naturaleza, para el diseño de sistemas artificiales adaptativos.

2. Estructura de un AG

Los AGs clásicos, manipulan una población de soluciones potenciales codificadas en cadenas binarias que las representan. Este conjunto de cadenas representan el material genético de una población de individuos. Los operadores artificiales de selección, cruza y mutación son aplicados para buscar los mejores individuos –mejores soluciones– a través de la simulación del proceso evolutivo natural. Cada solución potencial se asocia con un valor de *fitness*, el cual mide qué tan buena es comparada con las otras soluciones de la población. Este valor de fitness es la simulación del papel que juega el ambiente en la evolución natural darwiniana.

El paradigma de los algoritmos genéticos puede esquematizarse como sigue:

```
Inicializar(Población)
MejorFitness=Evaluar(Población)
Mientras (MejorFitness<FitnessRequerido)
  Selección=Seleccionar(Población)
  Población=CruzarYMutar(Selección)
  MejorFitness=Evaluar(Población)
FinMientras
```

Para comenzar, se inicializa la población completamente al azar. En la inicialización hay que tener en cuenta que la distribución de valores debe ser uniforme para cada rango representado por los cromosomas. A continuación se decodifica el genotipo en fenotipo de esta población inicial y evalúa el fitness para cada individuo. Es decir, se le asigna un valor numérico a su *capacidad de supervivencia*; en el espacio de soluciones de nuestro problema medimos qué tan bien resuelve el problema cada individuo.

Luego entramos en el bucle de optimización o búsqueda. Este ciclo termina cuando nuestro AG ha encontrado una solución adecuada para el problema. Es decir, deberemos valernos del fitness para el mejor individuo y de un umbral para evaluar esta condición de finalización.

Durante el proceso evolutivo artificial, se aplican varios operadores. Mediante un proceso fuertemente estocástico se genera una nueva población de individuos tomando en cuenta su fitness. Básicamente durante la selección se decide cuáles individuos serán padres de la nueva generación. La nueva población puede reemplazar completamente a la población anterior o solamente a los peores individuos, las peores soluciones.

Con los cromosomas candidatos a ser padres de la nueva población se efectúan cruzas y mutaciones. Las cruzas son intercambios de genes: el proceso consiste en intercambiar segmentos de los cromosomas de las parejas seleccionadas en forma aleatoria. Cuando un cromosoma sufre una mutación, el alelo de uno de sus genes cambia en forma aleatoria. Las mutaciones ocurren según una probabilidad de mutación, esta probabilidad es uno de los parámetros que gobierna la evolución.

Finalmente, la población nace y se decodifica el genotipo en fenotipo para evaluar su fitness. Al volver al principio del ciclo evolutivo verificamos nuevamente si nuestro mejor individuo supera los requisitos de la búsqueda, en caso contrario volvemos a repetir todo el proceso para obtener una nueva generación.

3. Diseño de la solución de un problema mediante AGs

Cuando queremos resolver un problema mediante AGs debemos determinar un conjunto de especificaciones clave:

Representación de los individuos: ¿cómo representamos una solución de nuestro problema mediante cromosomas?, ¿a partir de un conjunto de cromosomas dado, cómo obtenemos una solución? Deberemos determinar de qué forma se traduce el fenotipo en genotipo y viceversa.

Función de fitness: ¿cómo medimos la capacidad de supervivencia de un individuo, sus posibilidades de procrear y transferir la información de sus genes a la próxima generación? En el dominio de las soluciones debemos poder medir qué tan buena es cada solución con relación a las demás.

Mecanismo de selección: tenemos toda una población evaluada según el fitness y debemos elegir a los individuos que serán padres de la próxima generación. No es tan sencillo como elegir los M mejores. Veremos algunas formas de realizar una selección que, si bien premia a los mejores, no deja de dar la posibilidad azarosa de que uno de los peores individuos sea padre, como sucede en la naturaleza.

Operadores de variación y reproducción: los operadores básicos son las cruzas y mutaciones. Sin embargo, a pesar de que limitemos el estudio sólo a estos operadores, veremos varias formas de aplicarlos. A partir de los operadores podemos reproducir y obtener la nueva población. Durante la reproducción también tenemos que considerar algunas opciones.

4. Representación de los individuos

El primer aspecto a resolver es la codificación del problema en un alfabeto finito. Tradicionalmente se han empleado cadenas binarias, pero recientemente se están empleando esquemas más flexibles. Deberemos encontrar la forma de pasar desde el espacio del genotipo al espacio del fenotipo como muestra la Figura 1.

Para los AGs utilizamos la base canónica $\{0, 1\}$ en cadenas de longitud fija y finita para representar soluciones de nuestro problema. Lo

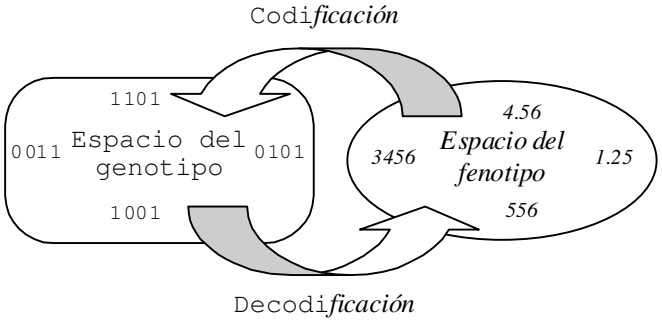

Figura 1. Representación de los individuos. Buscamos la forma de codificar soluciones en cromosomas y viceversa.

que representamos en estas cadenas depende de la aplicación. Podemos representar las conexiones, componentes y valores de éstos para un circuito electrónico; las líneas del código de un programa; los pesos y conexiones estructurales de una red neuronal y una infinidad de ejemplos más. Entre estos también encontramos a los coeficientes de un filtro ARMA o de un sistema no lineal.

Vamos a utilizar el caso más sencillo que consiste en representar una serie de coeficientes que describen algún sistema. Podemos utilizar un cromosoma para cada coeficiente. El cromosoma codificará en forma binaria el valor del coeficiente. Existen muchos métodos para codificar en forma binaria un valor real o entero, sólo veremos el caso más sencillo.

Deberemos decidir con qué resolución queremos representar a cada coeficiente. Cuando más bits tenga nuestra cadena más amplio será el espacio de búsqueda. Es decir, deberemos establecer un compromiso entre la resolución de la codificación y la cantidad de dimensiones del espacio de búsqueda.

Suponga que necesitamos codificar un coeficiente x en el rango $[a, b]$ mediante un cromosoma de n bits (genes). Debemos seguir dos pasos:

1. Aplicar un factor de escala y truncamiento para convertir al real x en un entero X perteneciente al rango $[0, 2^n - 1]$

2. Convertir el entero X en un número binario.

Para decodificar y transformar el genotipo en fenotipo aplicamos los pasos inversos:

1. Convertimos el número binario de la cadena del cromosoma en entero X.

2. Aplicar un factor de escala para convertir el entero X perteneciente al rango $[0, 2^n - 1]$ en el real x en $[a, b]$.

5. Función de fitness

Debemos obtener ahora una medida para conocer qué tan buena es la solución de cada individuo. Esta función trabaja en el dominio del problema, sobre el fenotipo de cada individuo. Por lo tanto para obtener el valor de fitness para un individuo deberemos previamente hacerlo *nacer*, y en algunos casos *crecer*, a partir de su genotipo para luego evaluar objetivamente qué tan bueno es en relación con los otros.

Hay un aspecto fundamental a resolver, el valor de fitness debe ser monotónicamente creciente con la bondad de la solución que un individuo representa. Si ésta dependiera de varios factores, de la forma en que éstos sean pesados en la función de fitness dependerá la optimización que realice el AG.

El "cómo" nace y crece un individuo es completamente dependiente de la aplicación. En el caso de que estemos optimizando la trayectoria de un móvil robot deberemos tener un robot real para cada fenotipo necesario o, como generalmente se resuelve, utilizar una buena simulación de su comportamiento. En el otro extremo, la sencillez está representada por la simple evaluación de una ecuación que es directamente una medida del fitness.

6. Selección

Existen varias formas de realizar la selección de los progenitores. Recordemos que, al igual que en la naturaleza, no se selecciona simplemente los mejores. La selección no está relacionada directamente con el fitness de un individuo sino a través de operadores probabilísticos. Veamos algunos métodos sencillos.

6.1. Rueda de ruleta

También denominado *selección basada en rangos*, consiste en asignar a cada individuo una porción de la ruleta que es proporcional a su fitness. Luego, hacemos rodar la ruleta y el favorecido es seleccionado para ser padre en la

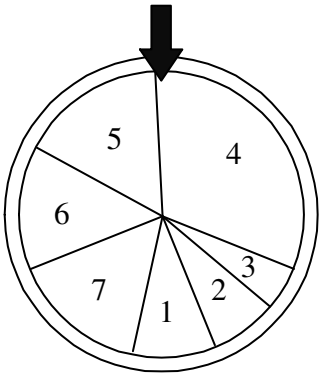

Figura 2. En este ejemplo de la rueda de ruleta el individuo número 4 es el que más probabilidades tiene de ser seleccionado. Sin embargo cualquiera puede ser padre.

próxima generación. A continuación mostramos el algoritmo del método de rueda de ruleta y en la Figura 2 un ejemplo gráfico.

$sumF =$ suma de todos los fitness
$sumR = rand(0,1)sumF$
$sumP = 0, j = 0$
`Repetir`
  $j = j + 1$
  $sumP = sumP + fitness_j$
`Hasta` $(sumP \geq sumR)$
Seleccionado$= j$

En poblaciones grandes es común que unos pocos individuos extraordinarios se encuentren sumergidos en un mar de colegas mediocres. Es por esta razón que la selección de la rueda de ruleta suele generar una convergencia lenta para el AG. Las áreas asignadas para los individuos buenos son proporcionalmente mayores pero hay tantos individuos mediocres que su área total es muy significativa. La selección no favorece suficientemente a los buenos individuos. Existe una técnica denominada *escalamiento de fitness* que ataca este problema justamente redefiniendo las distancias en fitness entre los mejores y los mediocres y peores. Sin embargo vamos a ver otras técnicas de selección en la que este problema no es tan marcado y aseguran una convergencia más rápida para el algoritmo.

6.2. Ventanas

En este método es necesario que primero ordenemos por fitness descendiente a los individuos. Hacemos N ventanas o rangos de selección $[0, q_i]$. El límite superior q_i va creciendo hasta llegar al total de la población. De cada ventana se elige al azar un individuo que queda seleccionado para ser padre.

En este método la mayor probabilidad de ser padre se asigna a los primeros individuos en la ordenación ya que están en todas las ventanas de selección. Los últimos individuos (menor fitness) sólo se encontrarán en condiciones de ser elegidos cuando se seleccione en la ventana $[0, q_N]$.

6.3. Competencias

En este caso se eligen, completamente al azar, $k > 1$ individuos; se los hace competir por fitness y queda seleccionado el ganador. Generalmente se utilizan valores de k entre 2 y 5 dependiendo del tamaño de la población. Este método es uno de los más utilizados debido a lo simple de su implementación.

7. Reproducción y operadores de variación

La reproducción es el proceso mediante el cual se obtiene la nueva población a partir de los individuos seleccionados y los operadores de variación. Existen varias alternativas para realizar la reproducción, en el caso más sencillo se obtienen *todos* los individuos de la nueva población a partir de variaciones (cruzas y mutaciones) de los progenitores.

Es posible también transferir directamente a la población nueva los padres seleccionados en la población anterior y completar los individuos faltantes mediante variaciones. En este caso interviene un parámetro denominado *brecha generacional* que determina cuántos padres son copiados directamente desde la población anterior a la nueva. Este parámetro es un número G real que está entre 0 y 1. Multiplicándolo por la cantidad total de individuos de la población permite encontrar la cantidad de padres que serán copiados.

Otra alternativa algo menos biológica pero que es utilizada con muy buenos resultados es el *elitismo*. En esta estrategia se busca el mejor individuo de la población anterior e *independientemente de la selección y variación* se lo copia exactamente en la nueva población. De esta manera nos aseguramos de no perder la mejor solución generación tras generación. Como veremos esta estrategia permite elevar el índice de mutaciones y aumentar

así la dispersión de las soluciones en el espacio de búsqueda. Vamos a ver a continuación los dos operadores de variación más utilizados.

7.1. Mutaciones

La mutación trabaja invirtiendo alelos de genes con una probabilidad p_m muy baja, por ejemplo $p_m = 0{,}001$ (ver Figura 3). Las mutaciones son típicamente realizadas con una probabilidad uniforme en toda la población y el número de mutaciones por individuo puede ser fijado de acuerdo a esta probabilidad y la cantidad de individuos. En los casos más simples se da la posibilidad de mutar sólo un alelo por individuo o se distribuye uniformemente sobre todo el cromosoma.

El papel fundamental de las mutaciones es no dejar que todos los individuos de la población se conviertan en mediocres (caigan en mínimos locales) permitiendo que constantemente se redistribuya la población sobre el espacio de búsqueda. Sin embargo una probabilidad de mutaciones demasiado elevada podría hacer que perdamos buenas soluciones impidiendo o retardando la convergencia. Cuando utilizamos elitismo nos aseguramos de no perder la mejor solución en probabilidades altas de mutación.

7.2. Cruzas

La cruza es un operador que, en el caso más sencillo, actúa sobre dos cromosomas para obtener otros dos. Existen dos tipos de cruzas: cruzas simples y cruzas múltiples. En las cruzas simples se elige un punto de cruza al azar y se intercambia el material genético correspondiente (ver Figura 4). En la cruza múltiple puede cortarse el cromosoma en más de dos partes para realizar el intercambio. En ambos casos el o los puntos de cruza son elegidos al azar. Veamos a continuación el algoritmo para la cruza simple.

$cruza = rand(1, LongCromo - 1)$
`Para` j `de` 1 `a` $cruza$
  $Hijo1_j = \text{Mutar}(Padre1_j)$
  $Hijo2_j = \text{Mutar}(Padre2_j)$
`FinPara`
`Para` j `de` $cruza$ `a` $LongCromo$
  $Hijo1_j = \text{Mutar}(Padre2_j)$
  $Hijo2_j = \text{Mutar}(Padre1_j)$
`FinPara`

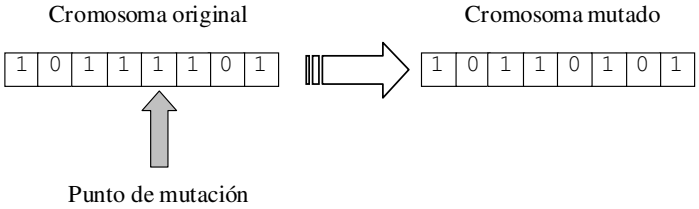

Figura 3. Mutación en un cromosoma de 8 genes.

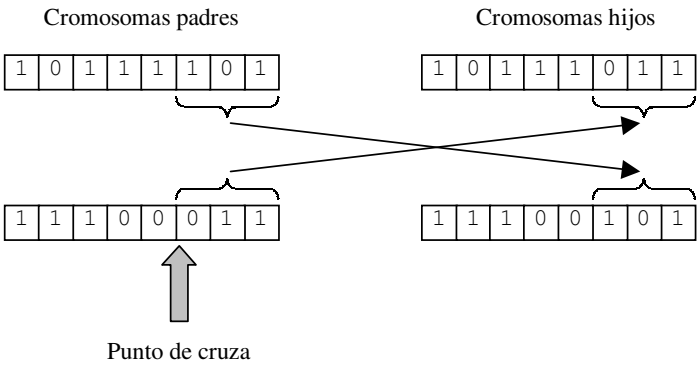

Figura 4. Cruza simple a partir de dos cromosomas de 8 genes.

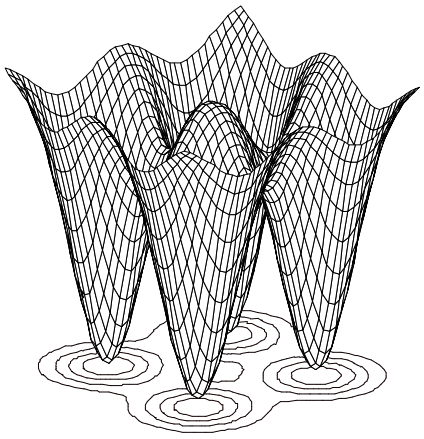

Figura 5. Curva de error o fitness invertido para un espacio de soluciones de dimensión 2. Uno de los picos llega 10^{-5} veces más abajo que los otros tres.

8. Características principales

Concebidos como una arquitectura genérica, como una técnica de búsqueda fuerte; cuando se aplican a problemas particulares los algoritmos genéticos deben sufrir modificaciones, para que presenten un buen desempeño y eficiencia. Así, a pesar de que tradicionalmente han empleado cromosomas binarios para codificar las soluciones, recientemente se está generalizando el empleo de estructuras más complejas de representación, como vectores de números reales, árboles, etc. Los operadores genéticos también sufren adecuaciones cuando se modifica la representación de las soluciones, en general, los nuevos operadores están en función del problema particular. Sin embargo, a diferencia de otras técnicas de computación evolutiva, el aspecto central de los algoritmos genéticos es la importancia que cumple el operador de cruzamiento y su relación con el operador de mutación. Mientras el operador cruza tiene una tasa o probabilidad elevada en todo algoritmo genético, el parámetro probabilidad de mutación tiene generalmente valores menos significativos.

Consideremos algunos ejemplos prácticos para hacer una comparación entre los algoritmos genéticos y los métodos de búsqueda por gradientes que vimos anteriormente.

Tomemos el ejemplo de la optimización para el sistema AR de segundo orden cuya superficie de error (fitness invertido) está dada por la suma de cuatro funciones gaussianas invertidas, como muestra la Figura 5. Agregue además la particularidad de que uno de los valles del error está solo 10^{-5} veces más cerca del cero que los otros tres. Encontrar el mínimo global

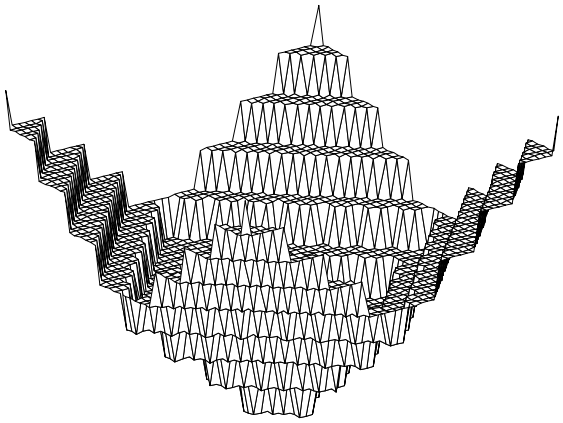

Figura 6. Curva de error o fitness invertido para un espacio de soluciones de dimensión 2. Los métodos basados en el gradiente pueden quedarse fijos en cualquiera de los escalones de la superficie.

implica encontrar el mínimo del valle que llega más abajo. Mientras un AG distribuye la población uniformemente sobre la superficie y analiza en paralelo todo el espacio de soluciones, el método de gradientes se encuentra limitado a la baja probabilidad de que, dadas las condiciones iniciales, el vector de coeficientes se encuentre en el valle apropiado. El AG busca simultáneamente es todo el espacio de soluciones, analiza *toda* la superficie antes de tomar una decisión. En contraste, el método de gradiente analiza sólo un punto de la superficie y un entorno pequeño de éste.

Otro caso donde el AG se ve favorecido es cuando la curva de error posee mesetas o escalones como se muestra en la Figura 6. ¿Qué pasa ahora con el método del gradiente del error en los escalones? Como el gradiente da cero si estamos en un escalón, el método no puede determinar la dirección que hay que seguir para caer en la superficie. Nuevamente el AG se ve favorecido por analizar simultáneamente muchos puntos uniformemente distribuidos en la superficie.

A pesar de estas grandes ventajas del método no podemos dejar de mencionar que generalmente los algoritmos genéticos requieren un tiempo de cálculo y capacidad de almacenamiento mucho mayor que otros métodos más convencionales. Este es el costo de la versatilidad y amplio espectro de aplicaciones que ofrecen los algoritmos genéticos.

Resumiremos en la siguiente lista las principales diferencias entre los algoritmos genéticos y otros métodos más tradicionales en la optimización y búsqueda:

1. Los AGs trabajan con una codificación del conjunto de parámetros de nuestro problema y no con los parámetros en sí mismos.
2. Los AGs buscan en una población de puntos en el espacio de soluciones, no en un punto en particular.
3. Los AGs utilizan la información de la función objetivo solamente y no sus derivadas o conocimiento auxiliar.
4. Los AGs utilizan reglas probabilísticas para las transiciones, no reglas determinísticas.

9. Introducción a los fundamentos matemáticos

Las características descriptas de un algoritmo genético, responden a un cuerpo teórico sólido que los soporta. Este se puede sintetizar en la *teoría de esquemas*. Un esquema es una cadena de caracteres definida sobre un alfabeto finito.

La teoría de esquemas predice el comportamiento de un esquema describiendo un subconjunto de cadenas con similaridades en ciertas posiciones. De acuerdo a esto, la solución óptima de un problema, se encuentra –debido a que el *teorema fundamental de los algoritmos genéticos* garantiza que el crecimiento de una instancia de un esquema determinado es aproximadamente proporcional al fitnesss relativo observado de tal esquema, dado que los esquemas con longitud definida pequeña, bajo orden y fitness superior al promedio de la población, son incrementados exponencialmente en la población–, a través de las generaciones, mientras que los esquemas con valores de fitness inferiores al promedio, longitud definida grande y alto orden, desaparecen.

La *hipótesis de los bloques constructores* plantea que los esquemas con alto valor de fitness (mayor que el promedio de la población), y longitud definida pequeña, donde longitud es la distancia entre la primera y la última posición fija en toda la cadena, constituyen los bloques constructores, los cuales permiten localizar las cadenas de mayor valor de fitness (las mejores soluciones o soluciones óptimas), mediante muestras de estos bloques con valor de fitness alto (selección) y también por combinación de bloques constructores (cruzamiento). La búsqueda altamente eficiente de los algoritmos genéticos, es explicada por lo que se ha denominado

paralelismo implícito, un aspecto fundamental de los mismos. Esto significa que a pesar de que un algoritmo genético procesa solamente N cromosomas o soluciones en forma directa, se ha demostrado, que implícitamente se evalúan o exploran N^3 esquemas, dado que en el caso de una representación binaria un cromosoma representa 2^d esquemas diferentes, donde d es la longitud del cromosoma.

10. Trabajos prácticos

Ejercicio 1: Implemente las estructuras de datos y algoritmos básicos para la solución de un problema mediante algoritmos genéticos. Pruebe estas rutinas buscando el mínimo global de las siguientes funciones:

- $f(x) = x^2$
  con $x \in [-5{,}12 \ldots 5{,}12]$,
- $f(x) = -x \sin(\sqrt{|x|})$
  con $x \in [-512 \ldots 512]$,
- $f(x, y) = (x^2 + y^2)^{0{,}25} \left[\sin^2 (50(x^2 + y^2)^{0{,}1}) + 1\right]$
  con $x, y \in [-100 \ldots 100]$.

Ejercicio 2: Población de escarabajos Tribolium Se ha utilizado un modelo demográfico para predecir la dinámica de una población de escarabajos de harina Tribolium bajo condiciones de laboratorio. Este modelo consiste en tres ecuaciones diferenciales acopladas, no lineales y estocásticas que bajo ciertas condiciones muestran un comportamiento caótico.

El modelo no ha sido completamente ajustado a los resultados obtenidos en experiencias de laboratorio y, si bien se conocen los rangos en los que pueden estar sus parámetros, no se conoce una buena aproximación de sus valores.

El objetivo principal de este trabajo es encontrar el conjunto de parámetros para los que el modelo se comporta de acuerdo a las mediciones experimentales realizadas en el laboratorio. Para lograr esto contamos con las mediciones experimentales, las ecuaciones en diferencia del modelo y los rangos biológicamente válidos en los que pueden encontrarse los valores de sus parámetros. Deberemos minimizar el error entre las mediciones de laboratorio y la salida del modelo mediante alguna técnica de identificación de sistemas.

Las características no lineales, el caos y las variables aleatorias presentes en el modelo dificultan la utilización de técnicas convencionales para la

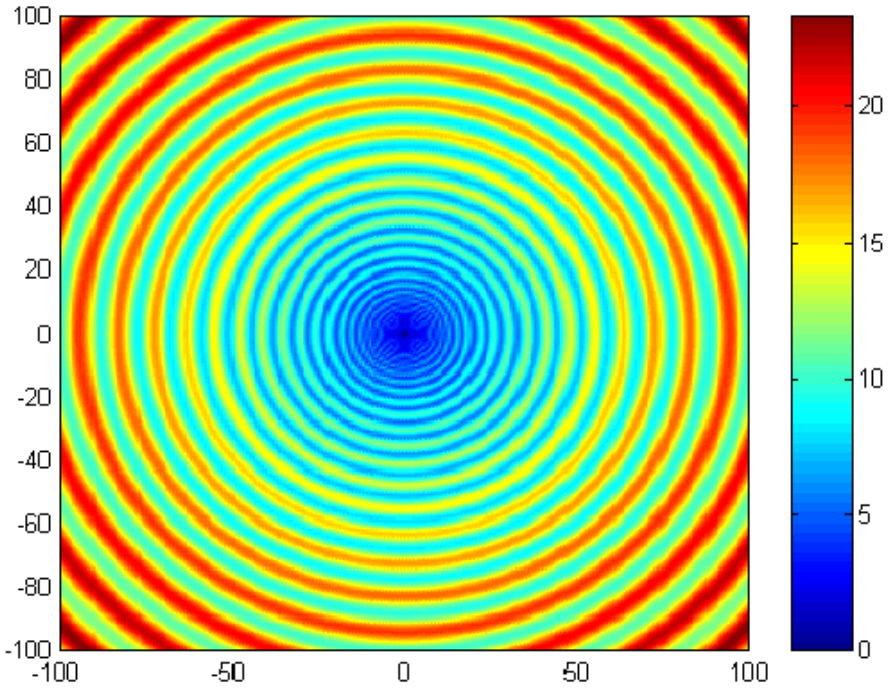

Figura 7. La función $f(x,y) = (x^2+y^2)^{0,25}\left[\sin^2\left(50(x^2+y^2)^{0,1}\right)+1\right]$ posee muchos mínimos locales y un mínimo global, que puede observarse en el centro de la gráfica.

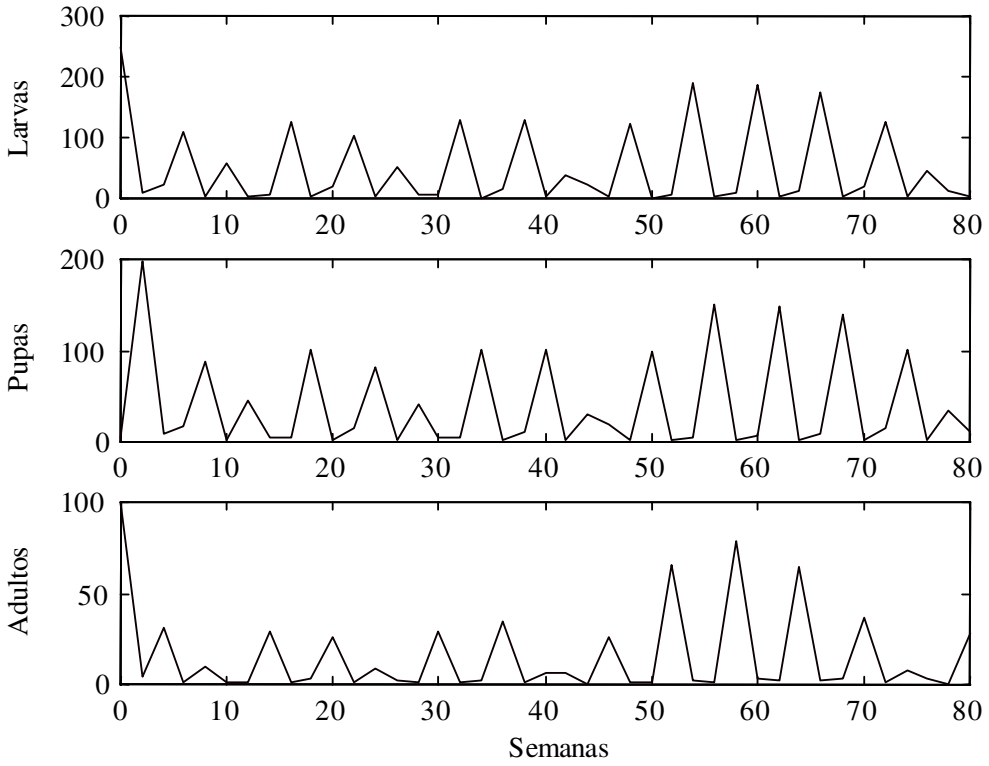

Figura 8. Evolución temporal de la población en sus tres estadios.

identificación de parámetros, y se propone en este ejercicio aplicar la técnica de algoritmos genéticos al problema de identificación de parámetros.

El modelo. El modelo describe la relación entre las poblaciones de escarabajos Tribolium de tres distintos estados de madurez. Las ecuaciones en diferencias de este modelo estocásticos son las siguientes:

$$\begin{aligned} L_{n+1} &= bA_n e^{-c_{el}L_n - c_{ea}A_n + E_{1n}} \\ P_{n+1} &= (1-\mu_l)L_n e^{E_{2n}} \\ A_{n+1} &= \left[P_n e^{-c_{pa}A_n} + (1-\mu_a)A_n\right] e^{E_{3n}} \end{aligned}$$

donde notamos como:

- n: tiempo discreto cuya unidad en el modelo es de 2 semanas.

- L_n: cantidad de larvas que sirven de alimento, para el tiempo n.

- P_n: cantidad de larvas que no sirven de alimento, pupas y adultos jóvenes, para el tiempo n.

- A_n: cantidad de adultos sexualmente maduros, para el tiempo n.

- b: número de larvas reclutadas por adulto y por unidad de tiempo en ausencia de canibalismo.

- μ_l: tasa de mortalidad de larvas en una unidad de tiempo.

- μ_a: tasa de mortalidad de adultos en una unidad de tiempo.

- $e^{-c_{el}L_n}$: probabilidad de que un huevo no sea comido en la presencia de L_n larvas, por unidad de tiempo (canibalismo).

- $e^{-c_{ea}A_n}$ probabilidad de que un huevo no sea comido en la presencia de A_n adultos, por unidad de tiempo (canibalismo).

- $e^{-c_{pa}A_n}$ probabilidad de supervivencia de pupas en presencia de A_n adultos, por unidad de tiempo.

- E_{Nn} variables de ruido aleatorio de una distribución normal multivariable acoplada con un vector de valor medio nulo y una matriz de varianza-covarianza $\mathbf{\Sigma}$.

Las nolinealidades exponenciales presentes en el modelo representan el canibalismo sobre los huevos que realizan larvas y adultos y sobre las larvas que realizan pupas y adultos. Las variables aleatorias representan desviaciones impredecibles de las observaciones realizadas sobre la estructura determinística ($\mathbf{\Sigma} = \mathbf{0}$) del modelo, provocadas por cambios ambientales entre otras causas.

Datos experimentales. A partir de una población inicial de 250 larvas pequeñas, 5 pupas y 100 adultos, se realizaron los recuentos de la cantidad de escarabajos en cada etapa durante 80 semanas con una periodicidad de 2 semanas. Los resultados se guardaron en un archivo de texto que contiene 3 columnas y 40 filas. En cada fila se encuentran los recuentos correspondientes a un período de 2 semanas que según las columnas corresponden a L_n, P_n y A_n.

De acuerdo a las condiciones de laboratorio en la que evolucionó la población podemos estimar cierto rango para los valores de los parámetros.

Además con relación al papel que desempeñan en el modelo sabemos que todos deben ser mayores que cero. Todas estas consideraciones nos permitirán reducir el espacio de búsqueda:

$$\begin{aligned} b &< 10^1 \\ \mu_l &< 10^0 \\ \mu_a &< 10^0 \\ c_{el} &< 10^{-1} \\ c_{ea} &< 10^{-1} \\ c_{pa} &< 10^0 \end{aligned}$$

En la Figura 8 vemos tres gráficas que muestran la evolución de la población bajo condiciones de laboratorio en un período de 80 semanas.

Acerca de la implementación. Se recomienda utilizar las siguientes pautas para la implementación:

- Codificación binaria de 16 bits de resolución ajustada a rango de variación de cada parámetro.
- Cruzas simples.
- Bajas probabilidades de mutación ($< 0{,}1$ no elitista, $< 0{,}4$ elitista).
- Una estructura determinística del modelo para simplificar este trabajo práctico.
- No menos de 100 individuos por generación.

Ejercicio 3: Cada integrante del grupo debe buscar 3 problemas reales para ser solucionados mediante computación evolutiva. Estos problemas se presentarán a los docentes a cargo y se seleccionará uno por cada integrante del grupo, de acuerdo a la pertinencia y complejidad del problema. A partir de los algoritmos implementados grupalmente en el ejercicio anterior, cada integrante del grupo debe resolver su problema y exponerlo oralmente.

11. Bibliografía

- S. Sumathi, T. Hamsapriya, P. Surekha, Evolutionary Intelligence: An Introduction to Theory and Applications with Matlab, Springer, 2008.

- C. A. Coello Coello, G. B. Lamont, A. A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems Series: Genetic and Evolutionary Computation, (2da Edición), Springer, 2007.

- R. L. Haupt, S. E. Haupt, Practical Genetic Algorithms, John Wiley & Sons, 2004.

- Bäck T., Hammel U. and Schewfel H-F., "*Evolutionary Computation: Comments on History and Current State*," IEEE Trans. on Evolutionary Computation, vol. 1, no. 1, apr. 1997.

- Constantino R. F., Desharnais R. A., Cushing J. M. And Dennis B., "*Chaotic Dynamics in an Insect Population*," Science, vol. 275, pp. 389-391, jan. 1997.

- Goldberg D. E., ***Genetic Algorithms in Search, Optimization & Machine Learning***, Addison Wesley, chap. 1, 3, 4, mar. 1997.

- Maniezzo V., "*Genetic Evolution of the Topology and Weight Distribution of Neural Networks*," IEEE Trans. on Neural Networks, vol. 5, no. 1, jan. 1994.

- Michalewicz Z., *Genetic Algorithms + Data Structures = Evolution Ptrograms*, Springer-Verlag, 1992.

- Montana D. J., "*Neural Networks Weight Selection Using Genetic Algorithms*," chap. 5 in *Itelligent Hybrid Systems*, J. Wiley & Sons, 1995.

- Morikawa H., "*Adaptative Estimation of Time-Varying Model Order in the ARMA Speech Analisis*," IEEE Trans. Acoust. Speech, Signal Processing, vol. 38, pp. 1073-1083, jul. 1990.

- Park L. J., Park C. H., Park C. and Lee T., "*Application of genetic algorithms to parameter estimation of bioprocesses*," Medical & Biological Engineering & Computing, jan. 1997.