Revista de Matemática: Teoría y Aplicaciones 2006 **13**(2) : 139–150
cimpa – ucr – ccss issn: 1409-2433

ALGORITMOS EVOLUTIVOS EN LA SOLUCIÓN DE PROBLEMAS DE ESTIMACIÓN DE PARÁMETROS

Aymée de los Ángeles Marrero Severo*
Liuva M. Pedroso Rodríguez† Jorge Barrios Ginart‡

Recibido/Received: 25/09/06 — Aceptado/Accepted: 23/10/06

Resumen

El problema general de determinar los valores de los parámetros de un modelo dinámico a partir de resultados experimentales, se conoce comúnmente como problema de estimación de parámetros y está presente en muchas de las facetas del desarrollo actual de la matemática, la computación y muchas otras ramas y aplicaciones. La estimación de parámetros de un modelo matemático, sin embargo, se puede enfocar por vías muy diversas. Este trabajo tiene como principal objetivo, proponer un enfoque para la obtención de soluciones numéricas aproximadas del problema de estimación de parámetros en modelos dinámicos no lineales descritos por sistemas de ecuaciones diferenciales ordinarias, con el uso de algoritmos evolutivos como estrategias de optimización, proponiendo algunos resultados obtenidos en ejemplos ilustrativos, con el propósito de mostrar la factibilidad de su uso para la resolución de dicho problema.

Palabras clave: Estimación de parámetros, modelos dinámicos, algoritmos evolutivos.

Abstract

The global problem to determine the values of some parameters in dynamical models knowing experimental results is frequently known as *Parameters Estimation Problem* and it appears in many areas of sciences. So is clear that there are many ways to obtain a famuly of parametersvalues that satisfy the satated conditions. This work shows some experiences in the treatment of this kind of models when the constraints are ordinary differential equations using evolutive algorithms. Our aim is to show that another ways can be usefull too to solve this problem with similar facilities and efficiency.

*Facultad de Matemática y Computación, Universidad de La Habana, Cuba. E-Mail: aymee@matcom.uh.cu
†Igual que A. Marrero.
‡Igual que A. Marrero. E-Mail: jbarrios@matcom.uh.cu

Keywords: Estimation of parameters problem, dynamical systems, genetic or evolutive algorithm.

Mathematics Subject Classification: 65K05, 65K10, 37N40, 47A10.

1 El problema de estimación de parámetros

El problema de estimación de parámetros para sistemas definidos por ecuaciones diferenciales ordinarias o problema inverso descrito por sistemas de ecuaciones diferenciales ordinarias, es ya un problema clásico y ha sido considerado por varios autores sobre todo en los últimos tiempos. Diferentes métodos y puntos de vista han sido propuestos, incluso considerando problemas con estructuras especiales y estrategias y conceptos estadísticos ([1]), ([11]) y ([15]). También han sido diseñados nuevos esquemas de integración numérica con el objetivo de estudiar y resolver los llamados problemas rígidos descritos por sistemas de ecuaciones diferenciales ordinarias ([5]), ([8]) y ([4]), los que aparecen con mucha frecuencia en modelos que describen reacciones químicas y otras áreas de las ciencias aplicadas.

Teniendo en cuenta, que el interés esencial de este trabajo es la solución numérica del problema de estimación de parámetros en modelos dinámicos definidos por sistemas de ecuaciones diferenciales ordinarias, abordado como un problema de optimización, el mismo tiene como objetivo, además de mostrar el problema y su discretización, tratar la adecuación y validación de los algoritmos evolutivos para la resolución del mismo.

1.1 Planteamiento del problema

Sea el siguiente problema continuo de optimización:

$$\begin{aligned} \min J(\mu) &= \sum_{i=1}^{s} G_i(x(\tau_i), \overline{x}^i) \qquad (1)\\ \text{s.a. } \dot{x}\,(t) &= F(x(t), \mu, t), t \in [0,T] \\ x(0) &= x^0 \\ 0 &\leq \tau_i < \tau_{i+1} \leq T, i = 1, \ldots, s-1 \end{aligned}$$

donde: $x \in \Re^n, x^0$dado, $\mu \in \Re^p, G_i : \Re^n \times \Re^n \to \Re, F : [0,T] \times \Re^n \times \Re^p \to \Re$ lo que significa que se está modelando un proceso dinámico definido por un sistema n-dimensional de ecuaciones diferenciales ordinarias, las cuales dependen de un vector p-dimensional de parámetros desconocidos μ.

También se considera un conjunto de datos o mediciones $\left\{\overline{x}^1, \overline{x}^2, \ldots, \overline{x}^s\right\}$ del vector variable observado y de dimensión n, en diferentes instantes de tiempo $\{\tau_1, \tau_2, \ldots, \tau_s\}$ y se desea minimizar J, una suma de funciones residuales.

En trabajos anteriores, en los que se usaron técnicas deterministas ([6]) se presentó la idea de usar esquemas de integración numérica como restricciones, en lugar de las ecuaciones diferenciales, transformando el problema continuo en un problema discreto que puede resolverse de una manera más sencilla y exigiendo diferenciabilidad para cada una

de las funciones que intervienen en el modelo. Dicha transformación depende directamente del método o sistema de integración que se utilice.

En este trabajo se formuló el siguiente problema discreto en el cual, el sistema de ecuaciones diferenciales es sustituido por un esquema de paso simple:

$$\begin{aligned} \min \widetilde{J}(\mu) &= \sum_{i=1}^{s} \widetilde{G}_i(x_i, \overline{x}^i) \qquad (2) \\ x_{m+1} &= x_m + \sum_{i=1}^{v} b_i K_i, m = 0, 1, \ldots, k-1 \\ x_0 &= x^0 \end{aligned}$$

Donde las funciones K_i representan las fórmulas explícitas de Runge-Kutta y las fórmulas de integración de Rosenbrock y b_i, los coeficientes correspondientes al esquema utilizado. De manera general, las funciones K_i pueden ser tomadas de diferentes formas según los esquemas de integración numérica que se utilicen.

Considerando que la partición de integración debe contener al conjunto de tiempos de medición $\{\tau_1, \tau_2, \ldots, \tau_s\}$, de modo que a cada índice de medición $i \in \{1, 2, \ldots, s\}$ corresponde un índice de integración , se define cierta correspondencia entre los índices construyendo un conjunto que contiene los índices de integración correspondientes a las mediciones, lo que permite expresar la función objetivo $\widetilde{G}_i$ dependiendo de la función característica de dicho conjunto de índices.

En este trabajo, con el objetivo de abordar el problema que se ha descrito, se procedió haciendo un tipo de enfoque de "método directo" para la solución del problema inverso, tal y como se desarrolló en ([7]) para resolver el citado problema aproximado, pero se utilizó una estrategia no determinística, que se describe a continuación.

2 Uso de los algoritmos evolutivos

Durante los últimos años ha existido un creciente interés en las estrategias de resolución de problemas basados en los principios de la evolución y la herencia. Tales algoritmos mantienen una población de soluciones factibles, tienen un proceso de selección basado en la bondad o adaptabilidad (*fitness*) de los individuos y algunos operadores de recombinación. A este tipo de filosofía de solución pertenece la clase de las Estrategias Evolutivas, o sea, algoritmos que imitan los principios de la evolución natural.

Para un problema dado pueden ser formulados muchos programas evolutivos. Estos programas pueden ser diferentes en muchos aspectos:

- Estructuras de datos para la representación de los individuos,
- Operadores genéticos para la transformación de los individuos,
- Métodos para crear la población inicial,

- Parámetros (tamaño de la población, probabilidad de aplicación de los operadores, etc.).

El propio concepto de programa evolutivo ha evolucionado a lo largo de estos años, pasando de los originalmente llamados Algoritmos Genéticos ([9]), a los actuales Algoritmos Evolutivos. Todavía hoy día es frecuente encontrar referencias en las que se diferencia entre Algoritmo Genético y Algoritmo Evolutivo, en función de la estructura de datos utilizada para la representación de individuos y los operadores de evolución utilizados, aunque esta diferenciación tiene un alto porcentaje de subjetividad y carece de criterios estrictos.

Muchos autores plantean que una representación "natural" de las soluciones factibles de un problema dado, junto a una familia de operadores genéticos aplicables puede ser verdaderamente útil en la aproximación de soluciones de muchos problemas, y que este enfoque de modelado natural es una dirección prometedora para la solución de problemas en general.

Dada la complejidad que puede tener el problema de estimación de parámetros que en este trabajo se aborda, decidimos tratar su solución utilizando un Algoritmo Evolutivo como método de optimización.

2.1 Descripción de las estrategias evolutivas desarrolladas

Los Algoritmos Evolutivos constituyen algoritmos estocásticos cuyos métodos de búsqueda simulan el proceso evolutivo de una población dada, durante un período de tiempo prefijado, modelando fenómenos naturales tales como la herencia genética y la competencia Darwiniana por la supervivencia dentro de un hábitat determinado. La idea detrás de estos algoritmos es imitar lo que la naturaleza hace, incluyendo los cambios que añade mediante las mutaciones. (ver entre otros [14]).

El Algoritmo Evolutivo que se presenta en este trabajo siguen paso por paso el clásico descrito por los principales autores del tema, denominando *población* al conjunto de soluciones factibles del problema a resolver que procesa el algoritmo, *cromosoma o individuo* a una solución factible en la población y *gen* a cada elemento en la sucesión lineal que constituye el cromosoma.

Para distinguir entre los mejores y los peores individuos de la población usamos una función objetivo o de evaluación, que desempeña el rol de un medio ambiente, incorporando los requerimientos del problema a resolver y llamamos *fitness o adaptabilidad* del cromosoma al valor que retorna la función objetivo al ser evaluada en la solución factible representada por tal cromosoma.

Un *proceso evolutivo* sobre la población corresponde a una búsqueda multi-direccional sobre el espacio de soluciones factibles y una *iteración* del algoritmo es la generación obtenida durante el proceso evolutivo. En cada iteración las soluciones relativamente "buenas" se reproducen, mientras que las soluciones relativamente "malas" desaparecen.

En vistas de que nuestro objetivo es minimizar una suma de funciones residuales adoptamos el criterio siguiente: *La mejor solución es aquella que tiene asociado el menor valor de fitness, y la peor, la que corresponde al mayor.*

El algoritmo mantiene una población $P(t) = \{x_1^t, x_2^t, \ldots, x_m^t\}$ de m cromosomas en cada iteración t y x_i^t, $i = 1, 2, \ldots, m$, es cada individuo en ella. Una nueva población

(*iteración* $t+1$) se forma por la *selección* de los cromosomas con mejor *fitness*. Los individuos seleccionados experimentan transformaciones por la aplicación de los *operadores genéticos*, los cuales denominaremos *mutación y cruzamiento*, formando así *individuos* (soluciones factibles) nuevos.

Este Algoritmo Evolutivo, como cualquier evolutivo clásico, precisa para cada problema particular los siguientes componentes:

- Una representación genética para las soluciones factibles del problema.
- Una forma de crear una población inicial de soluciones factibles.
- Una función para evaluar el fitness de cada solución.
- Los operadores genéticos que alteren la composición de los individuos durante la reproducción.
- Los valores de los parámetros que usa el Algoritmo Genético (tamaño de la población, número de genes en un cromosoma, probabilidades de aplicación de los operadores genéticos, etc.).
- Un criterio de parada del algoritmo.

Hay muchos ejemplos de aplicaciones en los que se asegura que este algoritmo no converge si no se le incluye una *estrategia elitista*. Esta estrategia consiste en lo siguiente: *"Para cualquier iteración del algoritmo, se denota por* x^{t*} *al individuo mejor adaptado de P(t), definido tal que* $f(x^{t*}) = \min\left\{f(x_i^t) : i = 1, \ldots, m\right\}$. *Se consida la población de tamaño* $m+1$, *de manera que un individuo particular, por ejemplo el primero, siempre sea* x^{t-1*} *de la iteración anterior, y a este no se le harán mutaciones ni cruzamientos, aunque en la iteración* $t+1$ *podría ser sustituido por otro mejor adaptado".*

Finalmente se propone la siguiente estructura para el algoritmo:

$T \leftarrow 0$
Generar $P(0)$ (Población inicial),
Generar x^{0*} y ubicarlo de primero en $P(0)$,

Para $t = 1, 2, \ldots, Last\_Gener - 1$
1. Hacer cruces entre individuos de $P(t)$, excepto el primero
2. Hacer mutaciones en $P(t)$, excepto el primero,
3. Evaluar f en $P(t)$,
4. Calcular x^{t*},
5. Ubicar x^{t*} como primer individuo de $P(t+1)$,
6. Seleccionar de $P(t)$ los m individuos restantes para completar la población $P(t+1)$.

Nótese que x^{0*} pudiera no ser el individuo con mejor desempeño en $P(0)$.

2.2 Aplicación al problema de estimación de parámetros

Para la representación cromosómica de una solución factible μ de un problema se usa un arreglo de números reales. Luego, un cromosoma será un vector de $\Re^p$, donde p es la dimensión del vector de parámetros μ. Se denota por $[a_i, b_i]$ el intervalo donde μ_i toma valores. Sin pérdida de generalidad se ha asumido que los parámetros están acotados porque en la mayoría de las aplicaciones se conoce un rango donde dichos parámetros toman valores. Se considera una población de tamaño m. Sean $p_c \in (0,1)$ y $p_m \in (0,1)$ las probabilidades de cruzamiento y mutación de la población respectivamente. Se ha supuesto además, que la población evoluciona en el tiempo hasta obtener un número suficientemente grande de generaciones ($Last\_Gener$). Entonces, el mejor cromosoma en la última generación representa una solución óptima.

La población inicial se crea aleatoriamente con una distribución uniforme en el conjunto $[a_1, b_1]\times[a_2, b_2] \times \ldots \times [a_p, b_p]$ donde los parámetros están definidos. En caso que se conozca alguna información adicional sobre los parámetros, ésta se tomará en cuenta para crear la población.

Para medir el *fitness* de un cromosoma se precisa:

1) Resolver la ecuación en diferencias del problema.

2) Evaluar la función objetivo en la solución numérica obtenida. Este valor se toma como *fitness* del cromosoma.

Para el proceso de selección se han utilizado diferentes estrategias clásicas como *Ruleta*, donde a cada cromosoma $x_i, i = 1, 2, \ldots, m$ se asocia una porción c_i de círculo proporcional a su *fitness*, de forma que la suma de todas las partes sea 1. Cada parte c_i del círculo puede identificarse con la probabilidad de que el individuo $x_i, i = 1, 2, \ldots, m$ sea seleccionado. Es evidente que los individuos con mayor *fitness* tienen mayor probabilidad de selección. Sin embargo, esta estrategia solo es válida para maximización de funciones. Este trabajo ofrece una modificación de la estrategia anterior, adaptada a nuestro objetivo, minimización de una suma de funciones residuales.

Se construye una ruleta con ranuras de tamaño inversamente proporcional a los *fitness* usados de la siguiente manera:

- Calcular el valor del *fitness* para cada cromosoma $x_i, i = 1, 2, \ldots, m$, denotado por $f(x_i)$.
- Encontrar el *fitness* total de la población: $F = \sum_{i=1}^{m} f(x_i)$
- Calcular la probabilidad de selección p_i para cada cromosoma $x_i, i = 1, 2, \ldots, m$:

$$p_i = \frac{1}{m-1}(1 - \frac{f(x_i)}{F})$$

- Calcular una probabilidad acumulativa q_i para cada cromosoma $x_i, i = 1, 2, \ldots, m$:

$$q_i = \sum_{j=1}^{i} p_j$$

Durante el proceso de selección la ruleta se hace girar m veces. Cada vez se selecciona solo un cromosoma para la nueva población de la siguiente forma:

- Generar un número aleatorio r con distribución uniforme en el intervalo $[0, 1]$.
- Si $r < q_1$, seleccionar el cromosoma x_1; en caso contrario, seleccionar el cromosoma $x_i, i = 2, \ldots, m$ de modo que $q_{i-1} < r \leq q_i$.

Otro operador utilizado fue el *Torneo*, poniendo a competir (en la evaluación de su *fitness*) dos individuos de la población aleatoriamente y seleccionando el mejor. Este operador garantiza dos copias del mejor individuo y la desaparición del peor.

Una vez realizada la selección, se aplican los operadores de mutación y cruzamiento a los individuos de la nueva población. Otros parámetros del algoritmo son la *probabilidad de cruzamiento* p_C, que representa el número esperado de cromosomas a cruzarse, mp_c y la *probabilidad de mutación* p_m, que establece el gen que será mutado en cada cromosoma.

La herramienta computacional que se propone permite el uso de diferentes operadores de cruzamiento como *Cruce por un punto, Cruce promedio y Cruce lineal* y da la posibilidad de realizar *Mutación Aleatoria,* en la que para la posición del gen que va a mutar en cada cromosoma, se genera un número aleatorio u con distribución uniforme en el intervalo de definición del gen y se reemplaza el valor del gen μ_j por u.

3 El toolbox GAPEST

El algoritmo que este trabajo propone para la resolución numérica del problema fue implementado sobre *Matlab, Versión 6.0.0.88 Release 12.*

La resolución numérica de los sistemas de ecuaciones diferenciales ordinarias, fue hecha con la ayuda de los códigos *ode*45, Mark W. Reichelt and Lawrence F. Shampine, 6-14-94, Copyright 1984-2000 The MathWorks, Inc. Revision: 5.68, Date: 2000/08/01 20:57:29; y *ode*23*s*, Mark W. Reichelt and Lawrence F. Shampine, 3-22-94, Copyright 1984-2000 The MathWorks, Inc. Revision: 1.69, Date: 2000/06/02 00:11:24. El código *ode*45 es una implementación de las fórmulas de Dormand-Prince de órdenes 5-4, y el código *ode*23*s*, una implementación de las Nuevas Fórmulas Modificadas de Rosenbrock de órdenes 2-3.

En todos los experimentos:

- La población inicial se creó aleatoriamente con una distribución uniforme en el conjunto de definición de los parámetros.
- Se utilizaron los operadores genéticos descritos, combinados de todas las formas posibles.
- El criterio de parada utilizado fue el número máximo de generaciones deseado por el usuario.
- Se utilizó función objetivo mínimo-cuadrado con pesos, para penalizar las variables deseadas, cuando se requiera.

3.1 Resultados experimentales

Para analizar los resultados del algoritmo propuesto, se consideraron numerosos ejemplos, aunque en esta sección sólo se muestran los resultados obtenidos con algunos de los más ilustrativos.

Ejemplo 1

$$\begin{aligned} x_1' &= \frac{1}{5}(2\mu_1 + 3\mu_2)x_1 + \frac{6}{5}(\mu_1 - \mu_2)x_2, x_1(0) = 1 \\ x_2' &= \frac{1}{5}(\mu_1 - \mu_2)x_1 + \frac{1}{5}(3\mu_1 + 2\mu_2)x_2, x_2(0) = 0 \end{aligned}$$

donde μ_1 y μ_2 son los parámetros del modelo.

Se conoce la solución analítica y el valor del óptimo $\mu_1 = 0.0017$ y $\mu_2 = -100$.

Parámetros del Algoritmo Evolutivo:

Tamaño de la población	Número máximo de generaciones
80	500
Probabilidad de cruzamiento	Probabilidad de mutación
0.5	0.005

Tipo de Selección: *Torneo*. Tipo de cruzamiento: *Cruce Lineal*.

Tipo de mutación: *Mutación Aleatoria*.

Función objetivo:

$$J = \frac{|x - \overline{x}|}{|x|}.$$

Mejores valores estimados de los parámetros:

μ_1	μ_2	$fitness$
0.00170048	−99.86164194	0.00013

Las figuras 1(a) y 1(b) muestran la evolución de los parámetros μ_1 y μ_2, observándose la buena adaptación del algoritmo y las figuras 2(a) y 2(b), una superposición de las variables de estado, estimadas (representadas con líneas discontinuas) y los datos (línea continua), contra el tiempo, mostrándose la buena aproximación a los datos.

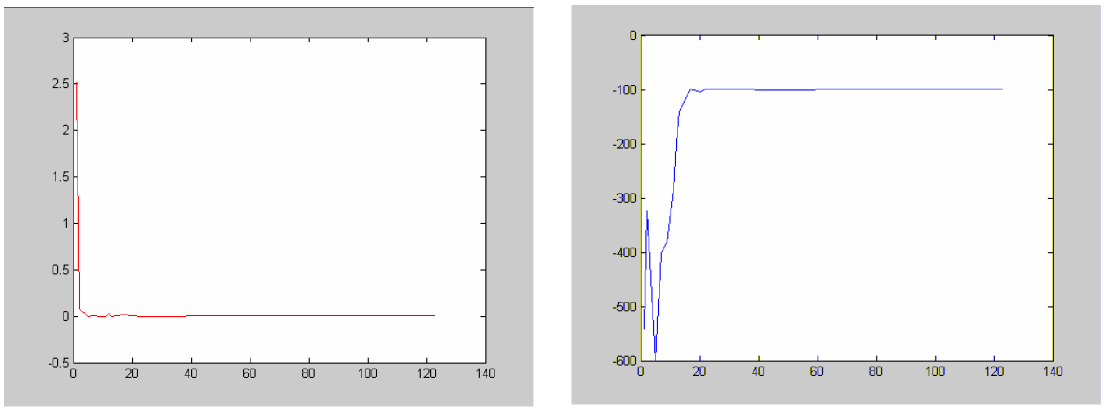

Figura 1: (a) Gráfica de evolución de μ_1. (b) Gráfica de evolución de μ_2.

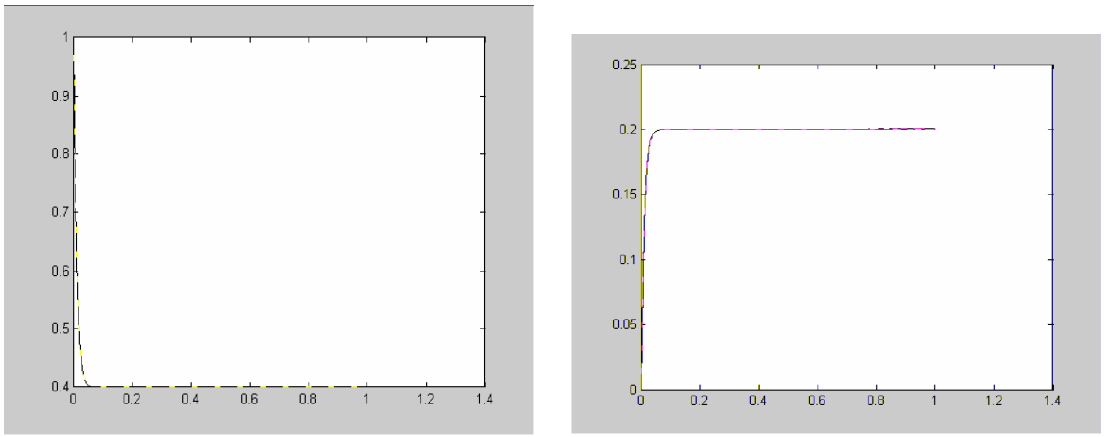

Figura 2: (a) Gráfica de x_1 y tiempo. (b) Gráfica de x_2 y tiempo.

Ejemplo 2

$$\begin{aligned} x_1' &= \mu_1 x_1 + x_2, x_1(0) = 1 \\ x_2' &= \mu_2 x_2, x_2(0) = \frac{999}{10} \end{aligned}$$

donde μ_1 y μ_2 son los parámetros del modelo.

Se conoce la solución analítica y el valor del óptimo $\mu_1 = -100$ y $\mu_2 = -0.1$.

Los Parámetros del Algoritmo Evolutivo coinciden con los del ejemplo anterior, asi como los operadores, excepto la función objetivo, que este caso fue

$$J = |x - \overline{x}|$$

Mejores valores estimados de los parámetros:

μ_1	μ_2	$fitness$
-100.0000000	-0.0999996	0.0000009

Al igual que en el ejemplo 1, las figuras 3(a) y 3(b), muestran la evolución de los parámetros μ_1 y μ_2, observándose la buena adaptación del algoritmo y las figuras 4(a) y 4(b), una superposición de las variables de estado, estimadas (representadas con líneas discontinuas) y los datos (línea continua), contra el tiempo, mostrándose la buena aproximación a los datos.

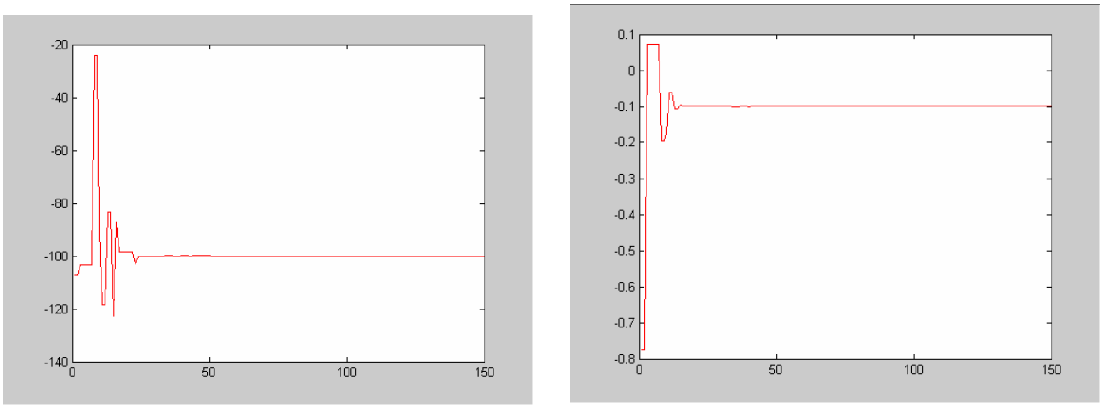

Figura 3: (a) Gráfica de evolución de μ_1. (b) Gráfica de evolución de μ_2.

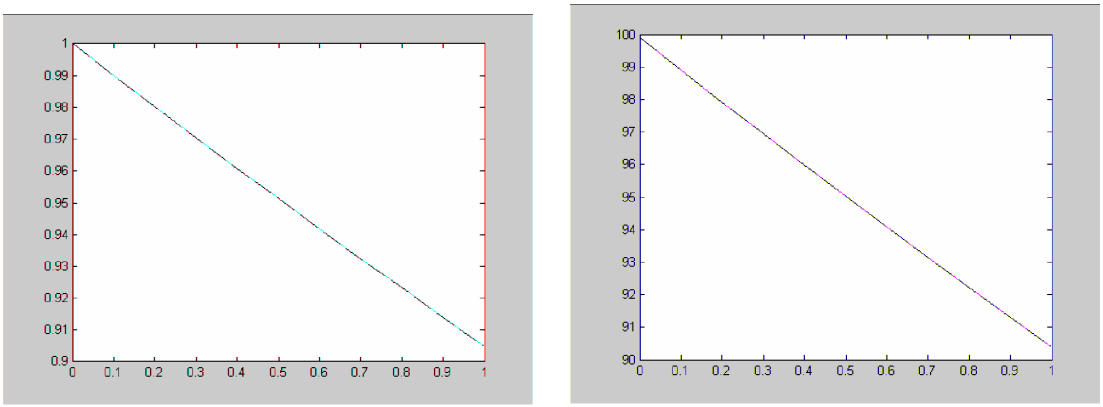

Figura 4: (a) Gráfica de x_1 y tiempo. (b) Gráfica de x_2 y tiempo.

4 Conclusiones y trabajos perspectivos

- Por su característica heurística, los algoritmos evolutivos brindan enormes posibilidades en el tratamiento de este tipo de problemas, particularmente cuando surgen

de modelar fenómenos de las ciencias aplicadas, en las que muchas veces los métodos tradicionales no se comportan eficientemente, esencialmente por la no convergencia al óptimo.

- En los resultados numéricos que se muestran en los ejemplos seleccionados, se da una clara idea del "buen" comportamiento de la estrategia evolutiva, ya que para el ejemplo 1, las estimaciones de los parámetros que muestra la tabla se obtuvieron en la generación 93, aunque un error del orden de 10-4 se obtuvo desde la generación 56, lo que demuestra la rápida convergencia de nuestro algoritmo a una solución aceptada. De forma similar, en el ejemplo 2, las estimaciones tabuladas se obtuvieron en la generación 69, pero un error aceptable, del orden de 10-7 aparece desde la generación 54. Otras simulaciones con juegos diferentes de operadores no reportaron estimaciones mejores de los parámetros.

- Sería interesante hacer un estudio más profundo de los resultados obtenidos, a partir de diferentes valores para los parámetros del algoritmo evolutivo (tamaño de la población, número máximo de generaciones, probabilidades de cruzamiento y mutación de la población), que permita arribar a conclusiones certeras sobre cuáles son los valores más apropiados de los parámetro del algoritmo que deben tomarse para obtener determinada calidad en las estimaciones.

- Comparar los resultados obtenidos utilizando los algoritmos evolutivos para los ejemplos tipo generados, con los obtenidos usando estrategias deterministas de optimización, o métodos clásicos, como los quasinewtons.

Referencias

[1] Bard, J. (1974) *Non linear Parameter Estimation*. Academic Press Inc.

[2] Butcher, J.C.(1987) *The Numerical Analysis of Ordinary Differential Equations*. John Wiley, New York.

[3] Dennis, J.E.; Schnabel, R.B. (1983) *Numerical Methods for Unconstrained Optimization and Non Linear Equations*. Prentice-Hall Series in Computational Mathematics, New Jersey.

[4] Enright, W.H.(1976) "Second derivate multi-step methods for Stiff Ordinary Differential Equations", *Numerical Analysis* **2**(2).

[5] Gear, W.C.(1971) *Numerical Initial Value Problems in ODE*. Prentice-Hall, New Jersey.

[6] Gómez, J.A.; Marrero, A. (2000) "Computing gradients of inverse problems in ODE models", *Revista de Investigación Operativa, ALIO* **9**(1,2,3): 179–206.

[7] Gómez, J.A & Marrero, A. (2000) "Convergence of discrete approximations of inverse problems in ODE models", *Revista Investigación Operativa, ALIO* **9**(1,2,3): 207–224.

[8] Hairer, E.; Warnner, G.; Nørsett, S.P. (1996) *Solving Ordinary Differential Equations I, II.* Springer-Verlag, New York.

[9] Holland, J. (1975) *Adaptation in Natural and Artificial Systems.* Univ. of Michigan Press, Ann Arbor.

[10] Iserles, A. (1996) "A First Course in the Numerical Analysis of Differential Equations", D.G. Crighton, Univ. of Cambridge.

[11] Jacquez, J.; Greif, P. (1985) "Numerical parameter identifiability and estimability and optimal sampling design", *Math. Biosciences* **77**.

[12] Luenberger, D.E. (1984) *Linear and non linear Programming*, Second Edition. Addison-Wesley, Massachusetts.

[13] Marrero, A. (2000) *Un Enfoque para la Solución Numérica del Problema de Estimación de Parámetros en Modelos Definidos por Sistemas de Ecuaciones Diferenciales Ordinarias.* Tesis Doctoral, Fac. de Mat. Comput, Universidad de La Habana.

[14] Michalewicz, Z.(1992) *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, Berlin Heidelberg.

[15] Mitsui, T. (1980) "The initial value adjusting method for problems of the least square type of ODE", *Publ. RIMS, University of Kyoto*, **16**.

[16] Nørsett, S.P.; Wolfbrandt, A. (1979) "Order conditions for Rosenbrock types methods", *Numer.Math* **32**.

[17] Shampine, L.F.; Reichelt, M.W. (1997) "The Matlab ODE suite", *SIAM J. Sci. Comput*, **18**(1).

[18] Zedan, H. (1989) "A variable order/variable –stepsize Rosenbrock-type algorithm for solving stiff systems of ODE's", *Tech. Report YCS114, Dept. Comp. Sci.*, Univ. of York, England.