Calibración de un modelo financiero con métodos metaheurísticos

Juan Ignacio Gil Gómez

Índice general

Capítulo 1

Introducción

El objetivo básico de este trabajo es estudiar la efectividad de algoritmos metaheurísticos para la optimización de funciones muy complejas sin representación analítica, que a menudo aparecen en los modelos para el cálculo de derivados financieros. Mostraremos un ejemplo de cómo este tipo de algoritmos puede ser mucho más efectivo que un algoritmo determinista.
Introduciremos algunos conceptos básicos de los modelos estocásticos utilizados para la valoración de opciones, describiendo el más básico de todos, el de Black-Scholes, donde los rendimientos del subyacente (el valor de una opción, de un futuro, un tipo de interés, el precio del oro...) sigue una distribución lognormal, para describir después al modelo que utilizaremos en este estudio, con un proceso más complejo que permite una volatilidad estocástica y la posibilidad de saltos bruscos en el valor.
Introduciremos después tres métodos de optimización utilizados para calcular los parámetros del modelo (a este proceso lo llamamos *calibración*). Dos de ellos son algoritmos metaheurísticos, basados en procesos que encontramos en la naturaleza: un algoritmo genético el algoritmo del murciélago, y el tercero es un algoritmo determinista, el algoritmo de Nelder-Mead, usado aquí para comparar los resultados con los primeros.
Describiremos después la implementación de estos algoritmos, y la función objetivo, que buscará que el modelo estocástico reproduzca lo mejor posible los valores de opciones cotizados en el mercado, para a continuación presentar los resultados y comparar la efectividad de estos algoritmos. Veremos que el algoritmo determinista no es capaz de solucionar este problema, y que el algoritmo genético presenta resultados muy superiores a los otros dos. También estudiaremos la estabilidad de las soluciones, comparando los resultados obtenidos con los datos de mercado de dos días consecutivos, y veremos que estos resultados, como era deseable, no cambian demasiado.
Para terminar estudiaremos una versión más simple de este mismo modelo, para comprobar si esta simplicidad favorece una mejor calibración.

Capítulo 2

Modelos estocásticos y valoración de opciones

1. Conceptos básicos. El modelo de Black-Scholes

En los mercados financieros existe la necesidad de disponer de modelos estocásticos que simulen la evolución de un subyacente (el precio de una acción, de un futuro, de tipos de interés o de cambio ...). Una vez que tenemos una expresión que simula adecuadamente el comportamiento de estos subyacentes, podemos utilizarla para calcular precios de instrumentos derivados (opciones, futuros, y especialmente otros productos más complicados) u otras magnitudes que nos interesen (por ejemplo, la distribución de rendimientos para cálculos de riesgos) expresándolas como funciones de los procesos estocásticos que el modelo define.
Uno de los modelos más sencillos, y todavía muy usado en el sector financiero, es el de Black-Scholes, para describir la evolución del precio S de una acción (suponemos aquí que no da dividendos), que viene dada por:

$$\frac{dS}{S} = rdt + \sigma dW$$

La expresión es la suma de dos términos: El primero es un término determinista, de deriva, que nos da la evolución del valor esperado del subyacente, y que en la práctica suele ser el tipo de interés.

$$\mathbb{E}[S(t)] = S(t_0)e^{r(t-t_0)}$$

El segundo término es estocástico,siendo σ la volatilidad, que se supone constante, y W es un movimiento browniano estándar (de media cero y varianza uno). Con esta expresión podemos calcular, por ejemplo, el valor de una opción. Una opción es un instrumento financiero que nos da el derecho de comprar (opción call, c) o vender (opción put, p) el subyacente a un precio fijo K en una determinada fecha de madurez T. Su valor viene dado por:

$$c = \mathbb{E}[\text{máx}(0, S-K)e^{-r(T-t)}]$$

$$p = \mathbb{E}[\text{máx}(0, K-S))e^{-r(T-t)}]$$

Con el proceso estocástico que definimos antes, podemos calcular estos valores esperados. El resultado es la fórmula de Black-Scholes[**Joshi**]:

$$c = N(d_1)S - N(d_2)Ke^{-r(T-t)}$$

$$p = N(-d_2)Ke^{-r(T-t)} - N(-d_1)S$$

donde $N(x)$ es la función de distribución acumulativa de la distribución normal, y

$$d_1 = \frac{log(\frac{S}{K}) + (r + \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}}$$

$$d_2 = d_1 - \sigma\sqrt{T-t}$$

Estas fórmulas son muy usadas en el sector financiero, incluso aunque se es consciente de que el modelo de Black-Scholes del que resultan no describe exactamente el comportamiento del subyacente. La razón principal es que sólo dependen de un parámetro propio del modelo: la volatilidad σ, pues el otro, r, es el tipo de interés. Esto permite definir la volatilidad implícita, que es el valor del parámetro sigma que, introducido en la fórmula de Black-Sholes, nos da el precio al que una opción se cotiza en el mercado. Es habitual es dar la cotización de las opciones como una volatilidad implícita, en vez de el precio en sí.
Modelos más complejos, como el que veremos en la sección siguiente, dependen de un mayor número de parámetros, que no necesariamente corresponderán con valores observables, lo que nos plantea el problema de cómo encontrarlos. El procedimiento habitual es similar al de encontrar la volatilidad implícita para el modelo de Black-Scholes: exigimos que nuestro modelo reproduzca los precios de los instrumentos que se cotizan en el mercado (futuros y opciones). El proceso de encontrar los valores óptimos de los parámetros, al que llamamos *calibración del modelo*, consiste en minimizar una función que suele ser una combinación lineal de los cuadrados de las diferencias entre los valores que predice el modelo y los que se negocian en el mercado.
Si nuestro modelo logra reproducir los precios que conocemos, aceptaremos su bondad para predecir precios de productos exóticos o poco líquidos, cuyos precios no se cotizan y por tanto no conocemos, o para poder utilizarlo en cálculos de riesgos.
La función que nos da los precios de las opciones puede ser muy compleja, y, en muchos modelos, no tener siquiera una expresión analítica, y tener que recurrir a métodos de Montecarlo para su cálculo. Esto hace ineficaces los métodos tradicionales de optimización. Una buena alternativa son los métodos metaheurísticos, métodos que combinan aleatoriedad y búsqueda local. Estos algoritmos están a menudo, como es el que caso de los que usaremos aquí, inspirados en procesos naturales[**Yang**].

2. Modelo a estudiar

El modelo de Black-Scholes, presentado en la sección anterior, asume la lognormalidad de los rendimientos, que no se observa en la realidad, especialmente en los productos

más inestables, presentando colas más largas que las de la distribución normal. Por ello, se han desarrollado muchos modelos más complejos que este, que intentan reflejar mejor la dinámica de los precios, incorporando reversión a la media, volatilidades variables, saltos o cambios de régimen[**Eydeland**].
El modelo que vamos a presentar aquí incorpora volatilidad estocástica (la volatilidad es otro proceso estocástico en sí misma) y saltos en el precio del subyacente, que serán más frecuentes en épocas de volatilidad alta. La dinámica de los precios viene dada por:

$$\frac{dS_t}{S_t} = \sqrt{V_t}dW_1 + (Y_t - 1)dq_t$$

$$\frac{dV_t}{V_t} = \sigma dW_2$$

donde:

- S_t Es el precio del subyacente en el tiempo t.
- V_t es el cuadrado de la volatilidad del subyacente en el tiempo t.
- W_1 y W_2 son movimientos brownianos estándar, correlacionados entre ellos:
$$dW_1 dW_2 = \rho dt$$
- q_t es un proceso de Poisson[**Ibarrola**] que rige la probabilidad de un salto. Su intensidad es proporcional a la volatilidad:
$$\lambda = \alpha\sqrt{V_t}$$
- $Y_t - 1$ es la amplitud de los saltos. Y_t sigue una distribución normal con volatilidad δ.

3. Subyacente. El mercado de emisiones

Vamos a aplicar este modelo a los precios de las emisiones de CO_2. El Protocolo de Kyoto, firmado en 1997, establece unas cuotas de emisiones de gases de efecto invernadero que se asignan a las empresas contaminantes, y permite la creación de un mercado para estas cuotas. Así, una empresa de uno de los países firmantes que necesite contaminar más que la cuota que se le ha asignado está obligada a acudir al mercado de emisiones a comprar las cuotas que necesite, bien en forma de cuotas directas de emisión (CER) o de participación de proyectos de desarrollo que se demuestren que ahorran esa misma cantidad de emisiones (ERU). De la misma manera, una empresa que contamine menos que la cuota asignada, o que participe en estos proyectos de desarrollo, puede convertirse en vendedora de derechos de emisión, ganando dinero con su reducción de emisiones. Teóricamente, esto debería producir un fuerte incentivo a las empresas contaminantes a reducir su producción de gases de efecto invernadero.
Alrededor de estos derechos de emisión se ha desarrollado un mercado muy líquido, de los propios certificados de emisiones, y más todavía de futuros de estos derechos (es decir, el compromiso de entregar uno de estos certificados en una fecha determinada a un precio fijado de antemano). También ha aparecido todo un mercado de opciones,

donde el subyacente son estos futuros de los derechos de emisión.
El que sea un mercado de futuros implica que su valor esperado es igual al valor actual, pues de lo contrario aparecerían oportunidades de arbitraje. Por eso nuestro modelo no incorpora ningún término de deriva (términos con dt), y

$$\mathbb{E}(S_t) = S_0$$

.

Capítulo 3

Algoritmos de optimización

Intentaremos calibrar este modelo con tres algoritmos diferentes: dos de ellos metaheurísticos, un algoritmo del murciélago (un algoritmo de creación muy reciente) y un algoritmo genético (más antiguo, y cuya efectividad está mucho más establecida), y otro determinista, el de Nelder-Mead (usaremos este porque es el que utiliza por defecto Matlab, el software en que hemos programado los otros dos algoritmos, con su función *fminsearch*[**Matlab**]).

1. Algoritmo del murciélago

El algoritmo del murciélago[**Yang**] fue desarrollado por Xin-She Yang en el 2010, simulando el comportamiento de una bandada de murciélagos al cazar, y su capacidad de ecolocalización. Para esta simulación, generamos unas reglas idealizadas que siguen nuestros murciélagos virtuales:

1. Todos los murciélagos usan la ecolocalización para medir las distancias, y son capaces de reconocer la diferencia entre la presa (el mínimo de la función a optimizar) y las barreras (que serán aquí las restricciones de esta función).
2. Los murciélagos vuelan aleatoriamente con velocidad v_i en la posición x_i, y usan para la ecolocalización una frecuencia fija f_{min}, una longitud de onda variable *lambda* y un volumen A. Según la proximidad de la presa pueden ajustar la longitud de onda de los pulsos y la frecuencia de pulso $r \in [0, 1]$ con que son emitidos.
3. Asumimos que el volumen varía desde un máximo A_0 hasta un mínimo A_{min}.

Siguiendo estas reglas, podemos crear nuestros murciélagos virtuales y especificar las ecuaciones con que se mueve el murciélago i en el tiempo t (al estar orientado a una simulación por ordenador, el tiempo es, obviamente, discreto):

$$f_i = f_{min} + (f_{max} - f_{min})\beta$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_\star)f_i$$

$$x_i^t = x_i^{t-1} + v_i^t$$

donde $\beta \in [0, 1]$ es un vector aleatorio con distribución normal, y $x_\star$ es la mejor solución en el instante $t - 1$, calculada tras evaluar la función en todos los x_i. En nuestro caso, x es un espacio de seis dimensiones, una por cada parámetro.

Algorithm 1 Pseudocódigo del algoritmo del murciélago

Iniciamos la población de murciélagos $x_i (i = 1, 2 \dots n)$ y v_i
Iniciamos las frecuencias f_i y las frecuencias de pulso r_i y los volúmenes A_i
while $t <$ Máximo número de iteraciones **do**
  Generamos nuevas soluciones ajustando la frecuencia y recalculando velocidades y posiciones
  $f_i = f_{min} + (f_{max} - f_{min})\beta$
  $v_i^t = v_i^{t-1} + (x_i^t - x_\star) f_i$
  $x_i^t = x_i^{t-1} + v_i^t$
  if $rand > r_i$ **then**
    Seleccionamos una solución entre las mejores
    Generamos una solución local alrededor de la mejor solución
  end if
  Generamos una nueva solución volando aleatoriamente
  if $(rand > A_i \| f(x_i) < f(x_\star)$ **then**
    Aceptamos las nuevas soluciones
    Aumentamos r_i y disminuimos A_i
    $A_i^{t+1} = \alpha A_i^t$
    $r_i^{t+1} = r_i^0[1 - e^{-\gamma t}]$
  end if
  Ordenamos las soluciones y elegimos la mejor $x_\star$
end while

Una vez que se ha encontrado la mejor solución de entre las que hemos generado, creamos una nueva solución para cada murciélago para la parte de búsqueda local:

$$x_{new} = x_{old} + \epsilon A^t$$

donde $\epsilon \in [-1, 1]$ es un número aleatorio, mientras que $A^t =< A^t >$ es el volumen medio de todos los murciélagos en el instante t.
En cada iteración, hay que actualizar el volumen A_i y la frecuencia de pulso r_i. Cuando el murciélago se va acercando a su presa, decrece el volumen (hasta llegar a cero, cuando ha atrapado la presa y ya no emite ningún sonido) y aumenta la frecuencia de los pulsos. Los valores iniciales de A_i y de r_i se crean aleatoriamente al principio de la simulación. En cada paso, se actualizan sólo si la nueva solución de este murciélago mejora a la anterior. Simulamos A_i^t y r_i^t como:

$$A_i^{t+1} = \alpha A_i^t$$

$$r_i^t + 1 = r_i^0[1 - e^{-\gamma t}]$$

donde α y γ son constantes que habrá que elegir para que el algoritmo converja lo mejor posible.

2. Algoritmo genético

Los algoritmos genéticos son seguramente los más populares y establecidos en el campo de la metaheurística. El primer algoritmo genético fue desarrollado por John Holland como una abstracción la evolución biológica, incorporando características de esta como el entrecruzamiento y la mutación genética.
La esencia de un algoritmo genético consiste en codificar una función de optimización como un vector de bits o caracteres que actuarán como cromosomas, y la manipulación de estos vectores con operadores genéticos, y la selección de los cromosomas más aptos con el objetivo de optimizar nuestra función.
El proceso es el siguiente:

- Codificar las variables como un cromosoma.
- Definir una función o criterio de selección
- Crear un población inicial
- Simular el paso de generaciones creando nuevas generaciones mediante entrecruzamientos y mutaciones, permitiendo sólo que los más aptos se reproduzcan, reemplazando a la nueva población.
- Decodificar el individuo más apto para conseguir la solución al problema

El entrecruzamiento de dos vectores parentelas es el principal operador, que actúa con una probabilidad p_c, y consiste en intercambiar un fragmento de un individuo con el correspondiente del otro individuo. La mutación, que se da con probabilidad $p_m < p_c$, consiste en cambiar aleatoriamente puntos del vector. La selección consiste en asegurarse que sólo los individuos más aptos (los que dan un resultado más próximo a nuestro objetivo) se reproduzcan a la siguiente generación, o, al menos, que lo hagan con mucha más frecuencia. La elección de este criterio de selección es uno de los puntos más delicados en este tipo de algoritmos.

Algorithm 2 Pseudocódigo del algoritmo genético

Función objetivo $f(x), x = (x_1, \ldots, x_n)^T$
Codificar la solución como cromosomas (vectores)
Definir la función de selección $F(f(x)$
Generar la población inicial
Definir la función de selección $F(f(x)$
Definir los valores de las probabilidades de entrecruzamiento p_c y mutación p_m
while $t <$ Máximo número de iteraciones **do**
  Generamos nuevas soluciones por entrecruzamiento y mutación
  if $p_c > rand$ **then**
    Entrecruzamiento
  end if
  if $p_c > rand$ **then**
    Mutación
  end if
  Aceptamos la nueva solución si mejora el valor de la función de selección
  Elegimos los mejores para criar una nueva generación
end while
Decodificar los resultados

3. Algoritmo determinista

A efectos de comparar los resultados con los de los métodos metaheurísticos, hemos utilizado el algoritmo de Nelder-Mead[**Nelder-Mead**], que Matlab utiliza por defecto en su función *fminsearch*. El método utiliza el concepto de un simplex, que es un politopo de N+1 vértices en N dimensiones, y, iteración tras iteración, aplica varios operadores sobre él (reflexión, contracción y expansión).

Algorithm 3 Pseudocódigo del algoritmo de Nelder-Mead

```
Función objetivo f(x), x = (x_1, ..., x_n)^T
Generar el simplex inicial
while t < Máximo número de iteraciones do
  Ordenar los puntos
  Reflexión
  if Mejora al mejor de los puntos then
    Expansión
    if Mejora al reflejado then
      Cambiamos por el expandido
    else
      Cambiamos por el reflejado
    end if
  else if Mejora a alguno bueno then
    Cambiamos por el reflejado
  else
    Contracción
    if Mejora al reflejado o al peor then
      Cambio por el contraído
    else
      Encogimiento
    end if
  end if
end while
```

Capítulo 4

Implementación

Implementar la calibración de nuestro modelo estocástico es un tema delicado, con muchas decisiones a tomar. La primera de todas, es, obviamente, con qué calibrar. A diferencia de otros activos de energía, existe un mercado muy líquido de opciones de emisiones de CO_2, con lo que usaremos las cotizaciones de opciones de estas opciones y exigiremos a nuestro modelo que las reproduzca. Otro camino posible, de no existir este mercado de opciones, habría sido tomar la serie histórica de las cotizaciones de las emisiones, y exigir que la distribución producida por nuestro modelo tenga los mismos modelos que esta. Sin embargo, los resultados de esta aproximación serían menos fiables, pues no incluirían las expectativas del mercado sobre la evolución futura del subyacente.

1. Función a minimizar

Vimos que el valor de la opción viene dado por:

$$c = \mathbb{E}[\text{máx}(0, S - K)e^{-r(T-t)}]$$

$$p = \mathbb{E}[\text{máx}(0, K - S))e^{-r(T-t)}]$$

Con nuestro modelo no existe una expresión cerrada que nos de estos valores, por lo que tenemos que recurrir a un método de Montecarlo[**Eydeland**]. Simularemos una cantidad suficientemente elevada de caminos que puede seguir la evolución de nuestro subyacente, y al final de cada uno de ellos calcularemos el valor de la opción para ese camino concreto. El valor esperado, que aceptaremos como un buena aproximación del valor de la opción, será la media de los valores de la opción a lo largo de todos estos caminos.
Recordemos que este modelo contiene cuatro variable aleatorias, dos de ellas correlacionadas, que tendremos que calcular en cada paso, lo que hace los cálculos muy pesados:

$$\frac{dS_t}{S_t} = \sqrt{V_t}dW_1 + (Y_t - 1)dq_t$$

$$\frac{dV_t}{V_t} = \sigma dW_2$$

$$dW_1 dW_2 = \rho dt$$

Queremos que los parámetros del modelo reproduzcan lo mejor posible los valores de las opciones que se cotizan en el mercado (llamaremos $P(t)$ a ese conjunto de precios). La opción más obvia es usando mínimos cuadrados. Nuestra función será:

$$f(V(0), \sigma, \rho, \kappa, \delta, \alpha; S(0), P(0), r(0)) = \sum_n w_n(valor_n - P_n)^2$$

Algorithm 4 Pseudocódigo de método de Montecarlo para el cálculo del valor de una opción

while $n <$ Máximo número de caminos **do**
  Damos a S y V sus valores iniciales
  while $t <$ Máximo número de pasos **do**
    Generamos el valor actual de la covarianza de $(S_n(t), V_n(t))$, como
    $L = chol\Big(\begin{pmatrix}\sqrt{V_n(t)} & 0\\ 0 & \sigma\end{pmatrix}\begin{pmatrix}1 & \rho\\ \rho & 1\end{pmatrix}\begin{pmatrix}\sqrt{V_n(t)} & 0\\ 0 & \sigma\end{pmatrix}\Big)$
    (donde chol representa la transformación de Cholesky de la matriz)
    Generamos números aleatorios con distribución normal y la covarianza deseada
    $W = Lrandn$
    Generamos el resto de los valores necesarios para calcular la evolución en este paso
    $\lambda_n(t) = \alpha V_n(t)$
    $Y_n(t) = e^{\kappa + \delta randn}$
    $q_n(t) = poissrnd(\lambda_n(t))\Delta t$
    Con estos factores podemos calcular la evolución de los subyacentes en este paso
    $S_n(t) = S_n(t-1)(1 + \sqrt{V_n(t)}W(1) + (Y_n(t) - 1)q_n(t)$
    $V_n(t) = V_n(t-1)(1 + \sigma W(2)$
  end while
  Nos quedemos con el valor final del subyacente $S_n(T)$
end while
Calculamos el valor de la opción como el promedio de los valores individuales en cada camino
$valor = \frac{1}{n}\sum_n[\text{máx}(callput(S_n(T) - K), 0)e^{-rT}]$
donde $callput = 1$ *para una opción call y* $callput = -1$ *para una put*

Donde w_n es el peso de la opción n. Como función de pesos, seguiremos una práctica habitual en el sector financiero, que es utilizar la vega de Black-Scholes de las opciones. La vega[1], υ, es la derivada del precio que nos da la fórmula de Black-Sholes repecto a la volatilidad implícita[**Joshi**]:

$$\upsilon = \sqrt{\frac{T}{2\pi}} S(0) e^{-d_1^2/2 - rT}$$

con

$$d_1 = \frac{log(\frac{S}{K}) + (r + \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}}$$

La razón de esta elección es que nos interesa calibrar mejor las opciones que sean más sensibles a un cambio en la volatilidad, que son aquellas donde $S(0)$ está más próximo al strike K (en la terminología financiera, aquellas que están *at the money*), pues una mala calibración nos daría valores muy erróneos, mientras que en aquellas con menor vega, el valor que calcularíamos no distaría tanto del real.
Nos queda todavía un punto: las restricciones de la función. En nuestras seis variables a encontrar, tenemos las siguientes restricciones:

$$V(0) \geqslant 0 \qquad \sigma \geqslant 0$$

$$-1 \leqslant \rho \leqslant 1 \qquad \kappa \geqslant 0$$

$$\delta \geqslant 0 \qquad \alpha \geqslant 0$$

Sin embargo, aunque teóricamente sea posible, permitir que las variables varíen hasta el infinito nos podría dar resultados que, aunque matemáticamente sean correctos y produzcan un valor mínimo de la función, no sean realistas, y provoquen una dinámica con mucha mayor variabilidad que la que realmente se observa. Además, tener un dominio de búsqueda demasiado amplio haría mucho más lento e ineficiente el proceso de optimización. Nunca, por ejemplo, se observan en el mercado volatilidades por encima del 100 %, ni tampoco podemos admitir valores altos de α, que llevarían a la inexistencia de saltos (para eso mejor eliminarlos del modelo),ni de κ o δ, que llevarían a que el valor del subyacente pudiera multiplicarse o dividirse por dos o tres en un sólo día. Por eso, estableceremos artificialmente límites a las variables, para evitar estos resultados poco realistas:

[1] A las derivadas del precio de la opción respecto a los diferentes parámetros del modelo se les llama *griegas*, ya que están representadas por letras del alfabeto griego. Así, δ es la primera derivada respecto al valor del subyacente, y γ la segunda. Sin embargo, para la derivada respecto de la volatilidad, se eligió el término *vega*, que no es una letra griega, y para la que nunca se ha tenido claro qué símbolo utilizar. Algunos prefieres utilizar el término κ para esta derivada, pero la propuesta no ha terminado de cuajar.

$$0 \leqslant V(0) \leqslant 2 \qquad 0 \leqslant \sigma \leqslant 2$$

$$-1 \leqslant \rho \leqslant 1 \qquad 0 \leqslant \kappa \leqslant 2$$

$$0 \leqslant \delta \leqslant 2 \qquad 0 \leqslant \alpha \leqslant 2$$

Para incluir estas restricciones en la optimización usaremos el método de la función de penalización[**Ramos**], añadiendo una función que aumente en un valor muy alto el resultado de la función cuando se incumpla alguna de las restricciones. Nuestra función a minimizar quedará entonces como:

$$f(x; S(0), P(0), r(0)) = \sum_n w_n(valor_n - P_n)^2 + 100000 * \|max(0, x - x_{max}) + max(0, x_{min} - x)\|$$

Donde x es el vector formado por la variables que buscamos:

$$x = (V(0), \sigma, \rho, \kappa, \delta, \alpha)$$

El método de Montecarlo que calcula el valor de los opciones necesita dos parámetros: el número de caminos a generar, y el número de pasos en cada camino. Decidimos tomar siempre el mismo valor para ambos, con un parámetro que llamaremos *calidad*.

2. Algoritmo del murciélago

Para el algoritmo del murciélago hicimos varias pruebas para buscar los valores más adecuados de sus parámetros, buscando a la vez que el algoritmo fuera capaz de explorar el espacio de soluciones con suficiente libertad, pero no tanto como para no ser capaz de converger a la solución, o tardar un tiempo intolerable en los cálculos. Finalmente, decidimos tomar:

$$n = 20 \qquad A0 = 0.25$$

$$r0 = 0.1 \qquad \alpha = 0.9$$

$$\gamma = 0.9$$

3. Algoritmo genético

En este algoritmo hay más decisiones que tomar, pues hemos de codificar nuestra solución en un cromosoma. Decidimos usar como cromosoma un vector Cr de treinta caracteres, cada uno de los cuales es un dígito del 0 al 9. Una solución x se convierte en un cromosoma de la siguiente manera:

$$x(1) = 2 * (0.1 * Cr(1) + 0.01 * Cr(2) + 0.001 * Cr(3) + 0.0001 * Cr(4) + 0.00001 * Cr(5))$$

$$x(2) = 2 * (0.1 * Cr(6) + 0.01 * Cr(7) + 0.001 * Cr(8) + 0.0001 * Cr(9) + 0.00001 * Cr(10))$$

$$x(3) = -1 + 2 * (0.1 * Cr(11) + 0.01 * Cr(12) + 0.001 * Cr(13) + 0.0001 * Cr(14) + 0.00001 * Cr(15))$$

$$x(4) = 2 * (0.1 * Cr(16) + 0.01 * Cr(17) + 0.001 * Cr(18) + 0.0001 * Cr(19) + 0.00001 * Cr(20))$$

$$x(5) = 2 * (0.1 * Cr(21) + 0.01 * Cr(22) + 0.001 * Cr(23) + 0.0001 * Cr(24) + 0.00001 * Cr(25))$$

$$x(6) = 2 * (0.1 * Cr(26) + 0.01 * Cr(27) + 0.001 * Cr(28) + 0.0001 * Cr(29) + 0.00001 * Cr(30))]$$

Tomaremos $n = 20$, $p_c = 0.2$ y $p_m = 0.1$. En cada generación sólo se reproducirán los 10 mejores, creando dos vástagos cada uno.
La implementación de los operadores de mutación y entrecruzamiento es muy simple:

- Mutación Recorremos uno por uno los cromosomas (dígitos del vector). En cada uno de ellos calculamos un número aleatorio con distribución uniforme en $[0, 1]$. Si este número es menor que p_m, mutamos el cromosoma seleccionando aleatoriamente un nuevo dígito.
- Entrecruzamiento Calculamos un número aleatorio con distribución uniforme en $[0, 1]$. Si es menor que p_c, elegimos al azar un dígito de origen y uno de fin, y otro miembro de la población, e intercambiamos entre ellos los cromosomas entre el dígito de origen y el de fin.

Capítulo 5

Calibración. Comparación de resultados

1. Resultados

Procederemos ahora a calibrar nuestros modelo usando los tres algoritmos descritos contra los datos del mercado. Usaremos las cotizaciones de opciones del 30 de junio del 2011 de las opciones sobre emisiones de CO_2 con entrega a final del 2013 (el 11 de diciembre). Haremos varias calibraciones, usando los tres algoritmos. Para cada una de ellas generaremos un punto de partida aleatorio en el dominio de soluciones posibles. En cada calibración subimos el parámetro de calidad del Montecarlo que calcula el valor de las opciones (este parámetro lo usamos para representar a la vez el número de pasos y caminos del Montecarlo).
En lugar de un número máximo de iteraciones, lo que forzamos es un tiempo máximo de cálculo. Así podemos, por un lado, comparar en igualdad de condiciones los tres algoritmos, y controlar además los tiempos de ejecución, que son muy largos, y a efectos prácticos, es nuestra principal restricción[1]
Repetiremos después el mismo cálculo con las cotizaciones del 1 de septiembre de 2011 de las opciones de emisiones de entrega al final del 2012.

[1]Todos los cálculos se han llevado a cabo en un MacBook con procesador Core 2 Duo de 2 Gz, y 2Gb de RAM, con MacOSX 10.7.1 y Matlab R2010a. En un ordenador más potente se podrían reducir muchísimo los tiempos de ejecución.

CUADRO 1. Resultados de la calibración. Opciones 2013. Datos del 20/6/2011

$T_{max} = 1000s$, Calidad= 40
$x_0 = (0.44329, 1.8608, 0.58476, 1.1201, 1.4005, 0.77727)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	708	(0.45351 , 1.88140 , 0.61470 , 1.07650 , 1.40340 , 0.78012)
Murciélago	2.65	(0.64110 , 1.61700 , 0.73316 , 0.12527 , 0.05847 , 0.57581)
A. genético	0.0660	(1.31720 , 0.13442 , -0.40518 , 0.74604 , 0.28190 , 0.06320)

$T_{max} = 2000s$, Calidad= 80
$x_0 = (1.71530, 0.45670, 0.01401, 7.53710, 1.2720, 0.57134)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	1680	(0.14674 , 0.49388 , -0.23427 , 1.26830 , 1.01450 , 1.58670)
Murciélago	0.511	(0.06439 , 0.50527 , 0.22292 , 0.01224 , 0.22849 , 1.62950)
A. genético	0.0428	(1.49070 , 1.14450 , 0.50528 , 0.14832 , 1.14770 , 0.04842)

$T_{max} = 3000s$, Calidad= 120
$x_0 = (1.71530, 0.45670, 0.01401, 7.53710, 1.2720, 0.57134)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	466	(1.71930 , 0.46847 , 0.01415 , 1.00640 , 1.25770 , 0.59446)
Murciélago	0.511	(1.81400 , 0.74691 , 0.76763 , 0.03007 , 0.30711 , 0.32008)
A. genético	0.0916	(1.68260 , 1.7211 , 0.99232 , 0.22010 , 0.21822 , 0.27106)

$T_{max} = 4000s$, Calidad= 160
$x_0 = (0.62182, 0.28625, -0.67189, 1.85350, 0.87647, 1.91280)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	12.9	(0.76066 , 0.38939 , -0.97369 , 0.02678 , 0.78962 , 1.45720)
Murciélago	11.5	(0.00068 , 0.29682 , -0.88818 , 1.74450 , 0.66548 , 1.95390)
A. genético	0.0374	(1.62160 , 1.87640 , 0.93546 , 0.15332 , 0.47726 , 0.11304)

$T_{max} = 5000s$, Calidad= 200
$x_0 = (0.69251, 0.01612, -0.64684, 0.77737, 0.06197, 0.41523)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	10.8	(0.68353 , 0.01675 , -0.64597 , 0.74047 , 0.06198 , 0.43983)
Murciélago	5.75	(0.88411 , 0.00640 , -0.66658 , 0.03656 , 0.05437 , 0.15331)
A. genético	0.473	(1.89160 , 1.79610 , -0.84666 , 0.23720 , 0.10938 , 0.43530)

$T_{max} = 6000s$, Calidad= 240
$x_0 = (0.38826, 1.65410, -0.48422, 1.18830, 0.03959, 0.43345)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	10.0	(0.40852 , 1.90820 , -0.47975 , 0.88588 , 0.04339 , 0.42498)
Murciélago	7.26	(0.62410 , 1.86390 , -0.37964 , 0.83488 , 0.00070 , 0.09749)
A. genético	0.437	(1.98830 , 1.60810 , -0.99216 , 0.15012 , 0.14326 , 0.46748)

CUADRO 2. Resultados de la calibración. Opciones 2012. Datos del 1/9/2011

$T_{max} = 1000s$, Calidad= 30
$x_0 = (0.55700, 1.09380, 0.91501, 1.92980, 0.31523, 1.94120)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	123	(0.59852 , 1.14660 , 0.99660 , 1.83950 , 0.32678 , 1.60740)
Murciélago	0.414	(0.00044 , 1.29160 , 0.87381 , 1.93760 , 0.31181 , 1.98010)
A. genético	0.00176	(1.55700 , 1.92780 , 0.90300 , 0.12282 , 0.33554 , 1.85100)

$T_{max} = 2000s$, Calidad= 60
$x_0 = (0.63338, 1.03220, 0.70596, 1.76200, 1.28090, 0.77751)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	1680	(0.14674 , 0.49388 , 0.23427 , 1.26830 , 1.01450 , 1.58670)
Murciélago	0.511	(0.06439 , 0.50527, 0.22292 , 0.01224 , 0.22849 , 1.62950)
A. genético	0.0428	(1.49070 , 1.14450 , 0.50528 , 0.14832 , 1.14770 , 0.04842)

$T_{max} = 3000s$, Calidad= 90
$x_0 = (0.93776, 1.83100, 0.90572, 0.44987, 0.24524, 0.58197)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	0.109	(0.95300 , 1.84890 , 0.94665 , 0.45299 , 0.25538 , 0.57755)
Murciélago	0.00786	(1.95390 , 1.77660 , 0.93622 , 0.16214 , 0.21999 , 0.07789)
A. genético	0.00979	(1.98950 , 1.81000 , 0.99566 , 0.44960 , 0.48036 , 0.45090)

CUADRO 3. Valores medios de $F(x_\star)$

Método	$\mu[\mathbf{F}(\mathbf{x}_\star)]$	$\sigma[\mathbf{F}(\mathbf{x}_\star)]$
Nelder-Mead	522.23	703.12
Murciélago	3.23	4.05
Algoritmo genético	0.13	0.03

De estos datos se pueden extraer varias conclusiones:

- El algoritmo de Nelder-Mead no es adecuado para este problema. Con demasiada facilidad se queda atascado en un mínimo local y rara vez llega a hacer un buen ajuste. Ambos algoritmos metaheurísticos sí parecen funcionar bien, aunque el del murciélago también falla algunas veces. El algoritmo genético se muestra siempre superior, y es el que elegiríamos para una aplicación real de esta calibración.
- Nuestro modelo presenta varios inconvenientes. Valores diferentes de los parámetros llevan a muy buenos ajustes de las cotizaciones de opciones. Quizás el modelo sea inestable de por sí, aunque también cabe la posibilidad de que este problema tenga solución. Un camino a seguir podría ser combinar esta calibración de las opciones con un estudio de los datos históricos de las cotizaciones de las emisiones. El estudio de los momentos, o un ajuste por máxima verosimilitud podrían llevarnos a acotar más el dominio de búsqueda, permitiéndonos quizás a eliminar resultados no realistas. Hay, por ejemplo, varias soluciones con un muy buen ajuste que corresponden a valores de $V(0)$ casi nulos, esto llevaría a que toda la variación del subyacente se produjera mediante saltos, que no consideramos creíble.
- Para valores muy altos de la calidad el algoritmo necesita tiempos demasiado largos para funcionar. En la segunda calibración, con las opciones del 2012, aumentamos la proporción tiempo/calidad, y los resultados son mejores.
- Podemos calcular un valor medio de x usando una media ponderada con la inversa del error como pesos. Más interesante aún será la media de los cuadrados de las diferencias de estos valores de x para cada método respecto de la media ponderada total calculada antes, que nos dirán mucho sobre la bondad de cada método, ya que nos gustaría que, además de ajustar bien el valor de las opciones, el resultado fuera constante de un proceso de simulación a otro. Este valor medio total sólo nos sirve para conocer la dispersión de las soluciones, no cómo un resultado óptimo. La media de dos valores donde la función sea mínima no tiene por qué ser un mínimo (bien puede ser incluso un máximo de la función).
  Los valores de estas medias ponderadas totales, son, para las opciones del 2013:

$$\bar{x} = (1.5126, 1.2716, 0.45866, 0.26521, 0.56135, 0.16949)$$

  y para las del 2012:

$$\bar{x} = (1.654, 1.8623, 0.90707, 0.17823, 0.35725, 1.3442)$$

CUADRO 4. Valores medios ponderados de $x_\star$. Opciones 2013. Datos del 20/6/2011

Nelder-Mead

$\mu[\mathbf{x}_\star]$	(1.45900 , 0.14234 , 0.51943 , 0.45276)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.201 , 0.34342 , 0.43647 , 0.12776)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.15107

Murciélago

$\mu[\mathbf{x}_\star]$	(0.88642 , 0.71316 , 0.41987 , 0.08610 , 0.24258 , 0.90462)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(1.05450 , 0.45686 , 0.17046 , 0.10354 , 0.11253 , 0.97146)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{6}$	0.25368

Algoritmo genético

$\mu[\mathbf{x}_\star]$	(1.55210 , 1.30560 , 0.46474 , 0.27450 , 0.58085 , 0.12483)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.02536 , 0.40702 , 0.35938 , 0.05227 , 0.14307 , 0.01321)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{6}$	0.09411

CUADRO 5. Valores medios ponderados de $x_\star$ .Opciones 2012. Datos del 1/9/2011

Nelder-Mead

$\mu[\mathbf{x}_\star]$	(0.95263 , 1.8482 , 0.94662 , 0.45427 , 0.25549 , 0.57853)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.49212 , 0.00075 , 0.00166 , 0.07794 , 0.01040 , 0.58732)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{6}$	0.12838

Murciélago

$\mu[\mathbf{x}_\star]$	(1.88990 , 1.74880 , 0.92447 , 0.19255 , 0.22180 , 0.13592)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.17476 , 0.04046 , 0.00780 , 0.05756 , 0.01850 , 1.55900)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{6}$	0.26174

Algoritmo genético

$\mu[\mathbf{x}_\star]$	(1.61850 , 1.88400 , 0.90324 , 0.17187 , 0.38430 , 1.58370)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.02520 , 0.02131 , 0.00662 , 0.01340 , 0.02371 , 0.38462)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{6}$	0.06451

Vemos en las tablas 4 y 5 que los resultados que nos da el algoritmo genético son mucho menos dispersos que los que dan los otros dos, y que en esta medida el algoritmo del murciélago, aunque los errores sean mucho menores que los del Nelder-Mead, no produce resultados menos dispersos que el determinista.

2. Estabilidad de la calibración

Otro punto que nos interesa estudiar es la estabilidad del proceso. En una aplicación real tendríamos que calibrar todos los días con los valores del subyacente y las opciones de cada día, y nuestro punto de partida no sería, como hemos hecho hasta ahora, un valor aleatorio, sino los resultados de la calibración del día anterior. Hemos reproducido aquí este proceso, con los valores de las opciones del 2012 del 1 el 2 de septiembre de 2011. Primero calibramos partiendo de un valor aleatorio las opciones del 1 de septiembre, para después calibrar las del 2 de septiembre partiendo del resultado obtenido el día anterior.

CUADRO 6. Resultados de la calibración. Opciones 2012. Datos consecutivos del 1/9/2011 y el 2/9/2011

$\mathbf{T_{max}}$(s) = 1000, calidad=30
x0 = (0.36695, 0.86623, 0.94778, 0.14727, 1.18360, 0.51523)

Nelder-Mead	
$F(x_\star, 1/9)$	0.127
$x_\star, 1/9$	(0.37448 , 0.86553 , 0.90216 , 0.15244 , 1.2347 , 0.52285)
$F(x_\star, 2/9)$	0.0810
$x_\star, 2/9$	(0.38489 , 0.86731 , 0.90407 , 0.15276 , 1.3118 , 0.5239)
$\|\Delta x_\star\|$	0.07785
Murciélago	
$F(x_\star, 1/9)$	0.0308
$x_\star, 1/9$	(1.0615 , 1.70190 , -0.99868 , 1.77870 , 0.03328 , 0.01925)
$F(x_\star, 2/9)$	0.00723
$x_\star, 2/9$	(0.88644 , 1.66300 , 0.88711 , 0.19841 , 0.29786 , 0.47925)
$\|\Delta x_\star\|$	0.10399
Algoritmo genético	
$F(x_\star, 1/9)$	0.00928
$x_\star, 1/9$	(1.50720 , 0.62502 , 0.00578 , 0.05328 , 0.51682 , 0.50122)
$F(x_\star, 2/9)$	0.00728
$x_\star, 2/9$	(1.5073 , 0.62502 , 0.02578 , 0.11328 , 0.51682 , 0.50120)
$\|\Delta x_\star\|$	0.06325

$\mathbf{T_{max}}$(s) = 2000, calidad=60
x0 = (0.72788, 1.61530, 0.14609, 1.73230, 0.59110, 0.53906)

Nelder-Mead	
$F(x_\star, 1/9)$	2.68
$x_\star, 1/9$	(0.73532 , 1.62230 , 0.15422 , 1.72800 , 0.62404 , 0.53475)
$F(x_\star, 2/9)$	1.07
$x_\star, 2/9$	(0.73763 , 1.68280 , 0.15843 , 1.57830 , 0.61563 , 0.56273)
$\|\Delta x_\star\|$	0.16416
Murciélago	
$F(x_\star, 1/9)$	0.0308
$x_\star, 1/9$	(1.06150 , 1.70190 , -0.99868 , 1.77870 , 0.03328 , 0.01925)
$F(x_\star, 2/9)$	0.00561
$x_\star, 2/9$	(1.19720 , 1.59550 , -0.99845 , 1.73030 , 0.03325 , 0.01938)
$\|\Delta x_\star\|$	0.17910
Algoritmo genético	
$F(x_\star, 1/9)$	0.00884
$x_\star, 1/9$	(1.74870 , 1.53270 , 0.64574 , 0.46308 , 0.57308 , 0.02698)
$F(x_\star, 2/9)$	0.00856
$x_\star, 2/9$	(1.81990 , 1.69270 , 0.78808 , 0.33348 , 0.13512 , 0.38558)
$\|\Delta x_\star\|$	0.16072

CUADRO 7. Resultados de la calibración. Opciones 2012. Datos consecutivos del 1/9/2011 y el 2/9/2011 (continuación)

$\mathbf{T_{max}}$(s) = 3000, calidad=90
x0 = (1.74870, 1.52670, 0.64775, 1.26250, 0.51369, 1.42700)

Nelder-Mead	
$F(x_\star, 1/9)$	11.7
$x_\star, 1/9$	(1.69070 , 1.61030 , 0.67507 , 1.23570 , 0.54182 , 1.35540)
$F(x_\star, 2/9)$	4.30
$x_\star, 2/9$	(1.7065 , 1.6799 , 0.69456 , 1.0709 , 0.55383 , 1.39940)
$\|\Delta x_\star\|$	0.18631
Murciélago	
$F(x_\star, 1/9)$	0.149
$x_\star, 1/9$	(1.8167 , 1.22560 , 0.08699 , 0.02004 , 0.98593 , 0.41528)
$F(x_\star, 2/9)$	0.0544
$x_\star, 2/9$	(1.97090 , 1.99110 , 0.38127 , 0.00313 , 0.45797 , 0.90686)
$\|\Delta x_\star\|$	1.10320
Algoritmo genético	
$F(x_\star, 1/9)$	0.00561
$x_\star, 1/9$	(1.84590 , 1.69210 , 0.62948 , 0.33350 , 0.13512 , 0.38558)
$F(x_\star, 2/9)$	0.00727
$x_\star, 2/9$	(1.94870 , 1.73240 , 0.76474 , 1.46510 , 0.57288 , 0.02656)
$\|\Delta x_\star\|$	1.2773

$\mathbf{T_{max}}$(s) = 4000, calidad=120
x0 = (0.37036, 1.48900, 0.78520, 0.21695, 1.08650, 1.04000)

Nelder-Mead	
$F(x_\star, 1/9)$	0.469
$x_\star, 1/9$	(0.38352 , 1.47550 , 0.79349 , 0.21475 , 1.05830 , 1.06050)
$F(x_\star, 2/9)$	0.303
$x_\star, 2/9$	(0.38276 , 1.54740 , 0.82940 , 0.21583 , 1.06090 , 1.06060)
$\|\Delta x_\star\|$	0.08042
Murciélago	
$F(x_\star, 1/9)$	0.367
$x_\star, 1/9$	(0.14389 , 0.19542 , 0.79609 , 0.28256 , 0.20613 , 0.0094641)
$F(x_\star, 2/9)$	0.0544
$x_\star, 2/9$	(0.64918 , 1.9638 , 0.63026 , 0.37861 , 0.34299 , 0.02900)
$\|\Delta x_\star\|$	1.85430
Algoritmo genético	
$F(x_\star, 1/9)$	0.00950
$x_\star, 1/9$	(1.95500 , 1.57270 , -0.99612 , 0.64428 , 0.68272 , 0.06098)
$F(x_\star, 2/9)$	0.00820
$x_\star, 2/9$	(1.95500 , 1.57220 , -0.98614 , 0.62428 , 0.68270 , 0.06016)
$\|\Delta x_\star\|$	0.02237

CUADRO 8. Resultados de la calibración. Opciones 2012. Datos consecutivos del 1/9/2011 y el 2/9/2011 (continuación)

$\mathbf{T_{max}}$(s) = 5000, calidad=150
x0 = (0.80060, 1.10190, 0.77311, 0.21548, 0.31587, 0.41439)

Nelder-Mead	
$F(x_\star, 1/9)$	0.266
$x_\star, 1/9$	(0.82366 , 1.15500 , 0.78522 , 0.21461 , 0.33096 , 0.42039)
$F(x_\star, 2/9)$	0.205
$x_\star, 2/9$	(0.83297 , 1.21160 , 0.79854 , 0.21428 , 0.34709 , 0.41032)
$\|\Delta x_\star\|$	0.06188
Murciélago	
$F(x_\star, 1/9)$	0.0478
$x_\star, 1/9$	(1.85440 , 1.78090 , 0.74899 , 0.09845 , 0.01963 , 0.09432)
$F(x_\star, 2/9)$	0.0324
$x_\star, 2/9$	(1.88910 , 1.98310 , 0.87090 , 0.07056 , 0.05639 , 0.10071)
$\|\Delta x_\star\|$	0.24315
Algoritmo genético	
$F(x_\star, 1/9)$	0.00293
$x_\star, 1/9$	(1.96260 , 1.95180 , -0.94924 , 0.01844 , 0.58836 , 0.33298)
$F(x_\star, 2/9)$	0.00812
$x_\star, 2/9$	(1.96260 , 1.95180 , -0.94924 , 0.01804 , 0.58836 , 0.33298)
$\|\Delta x_\star\|$	0.0004

De nuevo nos enfrentamos con los mismos problemas que antes: la incapacidad del método de Nelder-Mead para calibrar correctamente el producto, y el que haya varios puntos que den muy buenos valores de las opciones. El aspecto positivo es que no se producen grandes saltos en los resultados con el cambio de día, y las variaciones son pequeñas, como cabe de esperar en un modelo que se comporte bien.
Surge aquí un problema adicional, debido a la estructura discreta del algoritmo genético, que podría resolverse usando algoritmos evolutivos en su lugar. Al actuar los operadores genéticos gen a gen, el resultado del segundo día tiende a dejar siempre varias de las variables inalteradas. Intuitivamente, sin embargo, todas deberían registrar pequeños cambios, efecto que sí conseguimos con los otros algoritmos. De nuevo, creemos que con más potencia de cálculo, y pudiendo dejar pasar un número mucho mayor de generaciones, este problema sería más leve o desaparecería.
De todas las variables, ρ es la que parece presentar mayor variabilidad, teniendo valores muy próximos a -1 y otros muy cercanos a 1, por eso en el siguiente capítulo la dejaremos fija, con valor 0, y haremos los cálculos sin ella.

Capítulo 6

Un modelo más simple

A fin de simplificar el modelo, vamos a eliminar la correlación entre S y V, con lo cual se convierten en dos variables independientes. Además, vamos a eliminar del proceso de calibración $V(0)$. Le otorgaremos un valor igual a la volatilidad implícita de Black-Scholes de las opciones *at the money*[1] Esta hipótesis es relativamente exigente, pues la variable V no es la misma que la volatilidad implícita, pero aceptamos que si σ es lo bastante alta, V alcanzará pronto los valores adecuados para reproducir el precio de las opciones. El vector con las variables a optimizar ya sólo tendrá cuatro componentes:

$$x = (\sigma, \kappa, \delta, \alpha)$$

De nuevo, haremos una serie de calibraciones usando los tres algoritmos, partiendo de valores iniciales elegidos aleatoriamente, para los dos juegos de cotizaciones que estamos usando.

[1] Opciones cuyo strike es igual al valor actual del subyacente.

CUADRO 1. Resultados de la calibración. Opciones 2013. Datos del 20/6/2011

$T_{max} = 1000s$, Calidad= 30
$x_0 = (1.62720, 0.63162, 0.62260, 0.68995)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	34.4	(1.58740 , 0.66611 , 0.63598 , 0.71367)
Murciélago	7.09	(0.19879 , 1.18150 , 1.91230 , 0.01061)
A. genético	0.775	(1.40320 , 0.57614 , 1.88840 , 0.01002)

$T_{max} = 2000s$, Calidad= 60
$x_0 = (1.42760, 0.45999, 1.93610, 0.32129)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	203	(1.38660 , 0.46453 , 1.87040 , 0.32886)
Murciélago	6.92	(1.45680 , 0.06678 , 0.17807 , 0.53416)
A. genético	1.16	(1.76520 , 1.97290 , 0.99850 , 0.01352)

$T_{max} = 3000s$, Calidad= 90
$x_0 = (0.72766, 0.22622, 0.27651, 0.60822)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	11.5	(0.79086 , 0.20994 , 0.29267 , 0.57580)
Murciélago	7.28	(0.77360 , 0.31355 , 0.40006 , 0.00742)
A. genético	2.32	(1.95450 , 0.02460 , 0.66744 , 0.16512)

$T_{max} = 4000s$, Calidad= 120
$x_0 = (0.56594, 0.42481, 0.46039, 1.43410)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	5.97	(1.78630 , 0.00675 , 0.57737 , 0.34711)
Murciélago	7.37	(0.09828 , 0.00303 , 0.17669 , 0.31594)
A. genético	3.96	(1.94600 , 0.00218 , 0.07718 , 0.49088)

$T_{max} = 5000s$, Calidad= 150
$x_0 = (1.48700, 1.54290, 1.83430, 1.99760)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	10^{25}	(1.63200 , 1.66650 , 1.60720 , 2.01300)
Murciélago	$1.3 \cdot 10^5$	(-1.23360 , 0.40053 , 0.94495 , 2.01160)
A. genético	4.13	(1.95370 , 0.00834 , 1.44750 , 0.01856)

CUADRO 2. Resultados de la calibración. Opciones 2012. Datos del 1/9/2011

$T_{max} = 1000s$, Calidad= 30
$x_0 = (1.80630, 1.41120, 1.80210, 0.14343)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	0.486	(0.09998 , 0.42991 , 0.69433 , 1.25900)
Murciélago	0.398	(1.08640 , 0.18585 , 1.31400 , 0.17592)
A. genético	0.03076	(1.76500 , 1.24500 , 1.38440 , 0.01914)

$T_{max} = 2000s$, Calidad= 60
$x_0 = (0.09410, 0.41427, 1.06600, 0.42508)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	4.13	(1.86740 , 1.45480 , 1.72250 , 0.14457)
Murciélago	0.353	(0.01240 , 0.00615 , 0.05307 , 1.09160)
A. genético	0.137	(1.95000 , 0.01356 , 1.89030 , 0.02120)

$T_{max} = 3000s$, Calidad= 90
$x_0 = (1.5530, 0.43210, 0.71724, 1.33370)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	1.63	(1.47580 , 0.44448 , 1.08960 , 0.44539)
Murciélago	0.368	(1.27060 , 0.19606 , 0.69153 , 0.00271)
A. genético	0.260	(1.98900 , 0.01288 , 0.02062 , 0.01112)

$T_{max} = 4000s$, Calidad= 120
$x_0 = (0.17717, 1.69890, 1.22210, 0.44463)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	392	(0.18167 , 1.59390 , 1.25940 , 0.45728)
Murciélago	0.373	(0.16542 , 1.45080 , 0.43549 , 0.00159)
A. genético	0.300	(1.97240 , 0.00884 , 1.86360 , 0.01848)

$T_{max} = 5000s$, Calidad= 150
$x_0 = (0.70553, 0.61432, 0.01148, 1.84310)$

Método	$\mathbf{F}(\mathbf{x}_\star)$	$\mathbf{x}_\star$
Nelder-Mead	0.362	(0.93664 , 0.12070 , 0.02135 , 0.01562)
Murciélago	0.368	(0.93613 , 0.00120 , 0.00796 , 1.75790)
A. genético	0.356	(1.95950 , 0.00552 , 0.01960 , 0.77086)

CUADRO 3. Valores medios de $F(x_\star)$

Método	$\mu[F(x_\star)]$	$\sigma[F(x_\star)]$
Nelder-Mead	$5.6 \cdot 10^{24}$	$1.8 \cdot 10^{25}$
Murciélago	13000	41100
Algoritmo genético	1.34	1.57

CUADRO 4. Valores medios de $F(x_\star)$ (sin el resultado extremo)

Método	$\mu[F(x_\star)]$	$\sigma[F(x_\star)]$
Nelder-Mead	116	169
Murciélago	3.39	3.58
Algoritmo genético	1.03	1.31

Los valores de estas medias ponderadas totales, son, para las opciones del 2013:

$$\bar{x} = (1.50770, 0.69184, 1.14110, 0.12171)$$

y para las del 2012:

$$\bar{x} = (1.51890, 0.68818, 1.07720, 0.20758)$$

CUADRO 5. Valores medios ponderados de $x_\star$. Opciones 2013. Datos del 20/6/2011

Nelder-Mead	
$\mu[\mathbf{x}_\star]$	(1.45900 , 0.14234 , 0.51943 , 0.45276)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.20100 , 0.34342 , 0.43647 , 0.12776)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.15107

Murciélago	
$\mu[\mathbf{x}_\star]$	(0.64113 , 0.39329 , 0.66998 , 0.21945)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(1.04840 , 0.31205 , 0.75106 , 0.05954)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.33204

Algoritmo genético	
$\mu[\mathbf{x}_\star]$	(1.66960 , 0.79757 , 1.28500 , 0.07292)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.08253 , 0.60272 , 0.37175 , 0.02081)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.17831

CUADRO 6. Valores medios ponderados de $x_\star$. Opciones 2012. Datos del 1/9/2011

Nelder-Mead	
$\mu[\mathbf{x}_\star]$	(0.73107 , 0.32533 , 0.45393 , 0.51848)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.90074 , 0.21261 , 0.61305 , 0.42491)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.29716

Murciélago	
$\mu[\mathbf{x}_\star]$	(0.68408 , 0.36506 , 0.48460 , 0.61803)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.95778 , 0.40496 , 0.57381 , 0.66670)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.34050

Algoritmo genético	
$\mu[\mathbf{x}_\star]$	(1.83420 , 0.81684 , 1.30840 , 0.06124)
$\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]$	(0.10855 , 0.36150 , 0.34911 , 0.05156)
$\|\mu[(\mathbf{x}_\star - \bar{\mathbf{x}})^{\mathbf{2}}]\|/\mathbf{4}$	0.12918

En estos nuevos cálculos vemos que:

- Los errores son algo mayores que en el caso anterior. Esto era de esperar: al tener sólo cuatro variables que ajustas, el modelo tiene menos flexibilidad, y no es posible ajustarse tanto a los valores de las opciones.
- El resultado que esperábamos, una menor dispersión de los resultados, no se da. Si observamos las tablas 5 y 6 las dispersiones son del mismo orden, incluso mayores. Este nuevo modelo, por tanto, no nos aporta lo que esperábamos de él.
- Si obtenemos, sin embargo, un resultado muy interesante de cara al objetivo de este trabajo, que es la comparación de los algoritmos de optimización: el último caso de la calibración de las opciones del 2013 (tabla 1). La inicialización al azar del punto de partida nos da un punto extraordinariamente malo, y el algoritmo de Nelder-Mead y el del murciélago fracasan al intentar minimizar la función objetivo. En el del murciélago llegamos a obtener un resultado fuera del rango pese a nuestra función de penalización. El algoritmo genético, sin embargo, muestra una vez más su potencia calibrando también aquí correctamente los datos.

Capítulo 7

Conclusiones

Los resultados de la aplicación de métodos metaheurísticos para la calibración de modelos de cálculo de opciones son positivos. Incluso con un modelo tan inestable como este, muestran una gran potencia para buscar por todo el dominio y encontrar rápidamente el resultado óptimo, mostrándose en estas condiciones muy superiores al algoritmo determinista, especialmente el algoritmo genético, que consigue en todos los casos encontrar un mínimo de la función superando a los otros dos. Sería necesario un estudio experimental más amplio, incluyendo un análisis de la influencia de los parámetros de los algoritmos de optimización en los resultados de la calibración.

Bibliografía

[Yang] Xin-She Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, 2010.

[Eydeland] Alexander Eydeland y Krzystof Wolyniec, *Energy and Power Risk Management. New Developments in Modeling, Pricing, and Hedging*, Wiley Finance, 2003.

[Ibarrola] Ricardo Vélez Ibarrola, *Cálculo de Probabilidades 2*, Ediciones Académicas. S.A, 2004

[Matlab] Documentación de Matlab

[Nelder-Mead] John A. Nelder; R. Mead. *A simplex method for function minimization*, Computer Journal 7: 308–313, 1965

[Ramos] Eduardo Ramos Méndez. *Modelos y Métodos de Investigación Operativa. Programación no Lineal*, UNED, 2009

[Joshi] Mark Joshi. *The Concepts and Practice of Mathematical Finance*, Cambridge University Press, 2004