Banesto

Computación científica aplicada a Finanzas

César Sánchez de Lucas (jcsanchd@banesto.es)

Sábado, 13 de Marzo de 2010

Introducción

La Matemática financiera conlleva de forma inherente el uso y desarrollo de técnicas numéricas para la solución de distintos problemas. Todas las técnicas matemáticas que se pueden pensar no serían nada sin el apoyo de los ordenadores modernos. De hecho, es muy fácil encontrarse problemas financieros que hace pocos años eran intratables computacionalmente y que con el paso del tiempo y la mejora del hardware han pasado a ser "triviales".

En la siguiente charla vamos a ver algunos ejemplos de problemas relacionados con la matemática financiera y la implementación de su solución numérica.

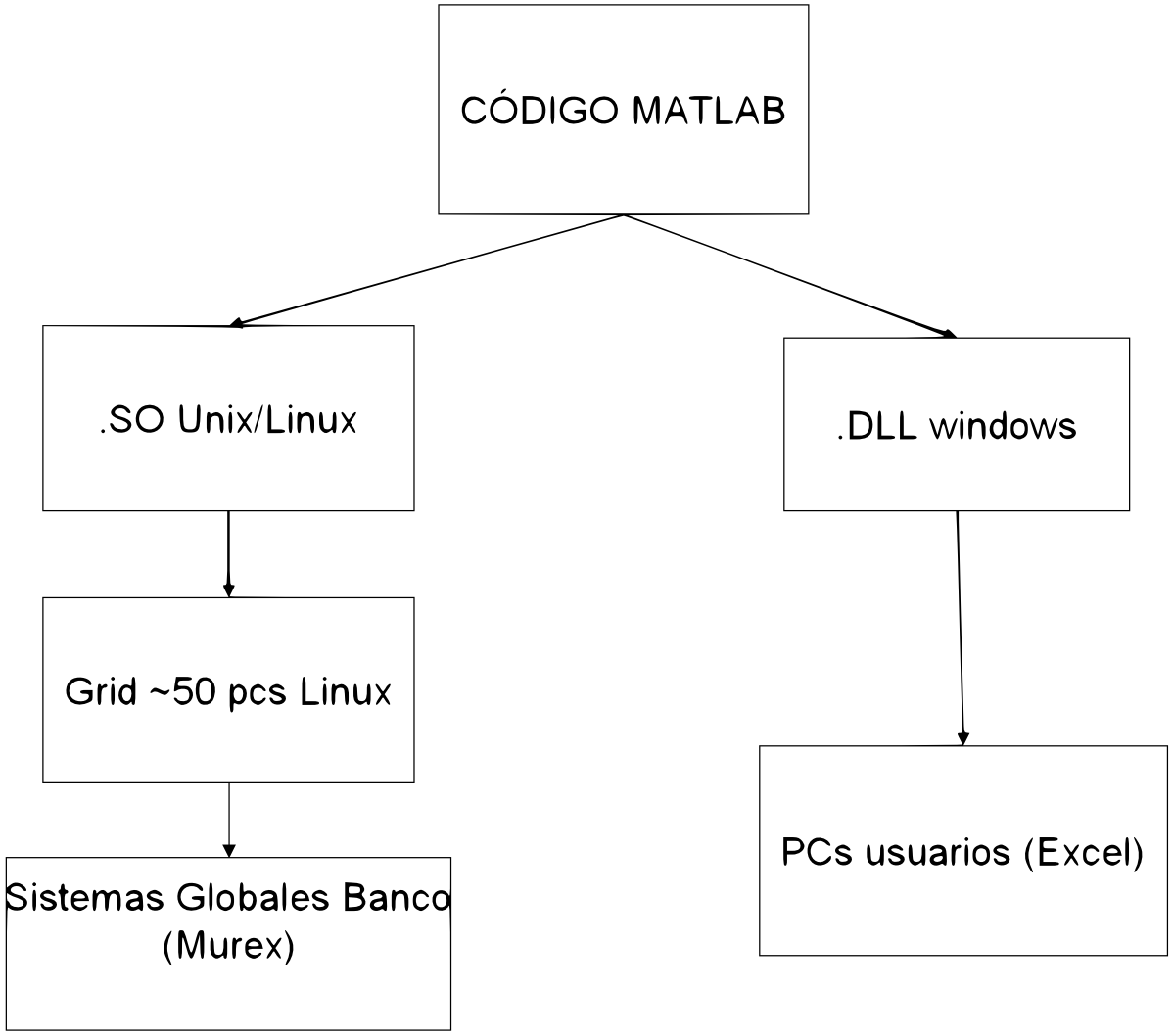
El entorno que hemos elegido para implementar esta solución es Matlab, por diversas razones. De hecho es el entorno de desarrollo que usamos en mi entorno profesional (Banesto).

Entre las razones que podemos citar para usar Matlab cabe destacar el hecho de que utilice un lenguaje de muy ***alto nivel*** (entendiendo *alto nivel* como próximo al lenguaje humano). Esto nos permite generar código muy fácilmente, que a su vez es muy fácil de mantener y de usar por el resto de desarrolladores.

Otra razón para usar Matlab es la facilidad con la que se puede integrar con el resto de software del banco. Desde las aplicaciones de escritorio de los traders (Excel) hasta los sistemas globales de gestión de la tesorería (Murex)

También (y para que no se diga) podemos mencionar algún inconveniente como sería el que se genere código en principio menos eficiente computacionalmente que el que haríamos, por ejemplo, con un programa en C. Otro inconveniente a tener en cuenta sería que se use para desarrollar un lenguaje que no es standard.

Esquema Técnico



Planteamiento y primera aproximación.

El problema fundamental en la matemática financiera es el de la valoración de lo que se ha dado en llamar ***"Derivados"***. Es decir, valorar un contrato financiero que pagará en un tiempo futuro en función del comportamiento de un determinado subyacente básico. Estos subyacentes básicos pueden ser el precio de las acciones, tipos de cambio, tipos de interés, precio del petróleo, zumo de naranja concentrado ...

El ejemplo típico de *Derivado* es la opción que me permite comprar (pero no obliga) en un tiempo futuro T una acción/subyacente a un precio fijado K, independientemente del valor de la acción en el mercado en ese momento. A este *Derivado* se le llama opción de compra o más comúnmente opción **Call**. Si denotamos por S_t el precio del subyacente en tiempo t, tendremos que el valor de la Call en tiempo T será:

$$\max(S_T - K, 0)$$

A esta función de pago se la denomina *payoff de la Call*. Nuestro objetivo será poner precio a este contrato.

Hipótesis fundamental, el subyacente sigue la siguiente ecuación de evolución estocástica:

$$\boxed{\frac{dS_t}{S_t} = \mu dt + \sigma dW_t}$$

donde dW_t es un browniano standard. Esta es la ecuación de evolución de un **browniano geométrico**. Al término μ se le conoce como **deriva** y a σ como **volatilidad**. Podemos interpretar la ecuación estocástica anterior según la siguiente aproximación discreta:

$$\frac{\Delta S_i}{S_i} = \frac{S_{i+1} - S_i}{S_i} = \mu \Delta t + \sigma \sqrt{\Delta t} \xi_i$$

$$S_{i+1} = S_i(1 + \mu \Delta t + \sigma \sqrt{\Delta t} \xi_i)$$

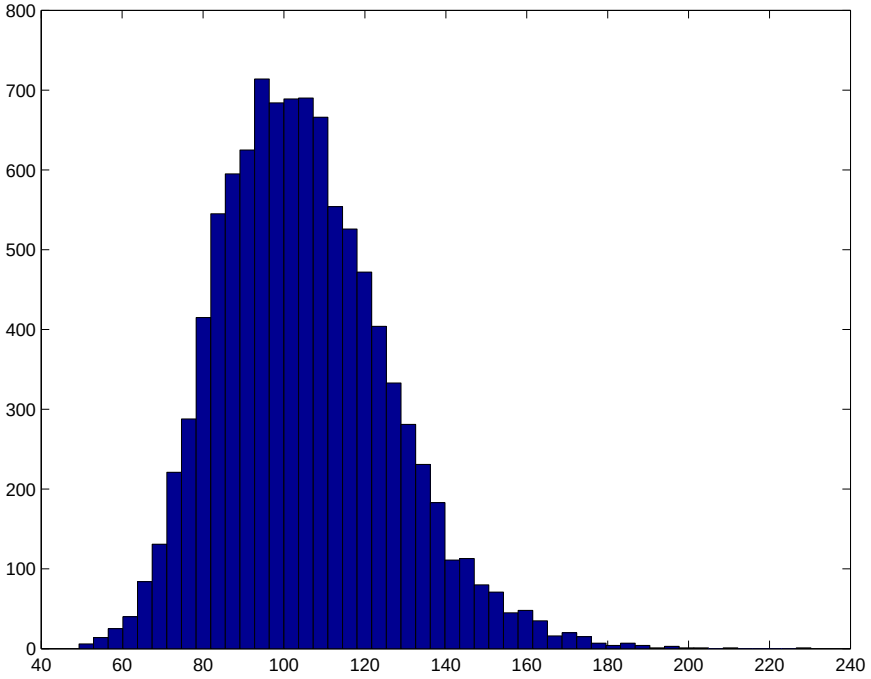
$\{\xi_i\}$ es una colección de normales standard independientes.

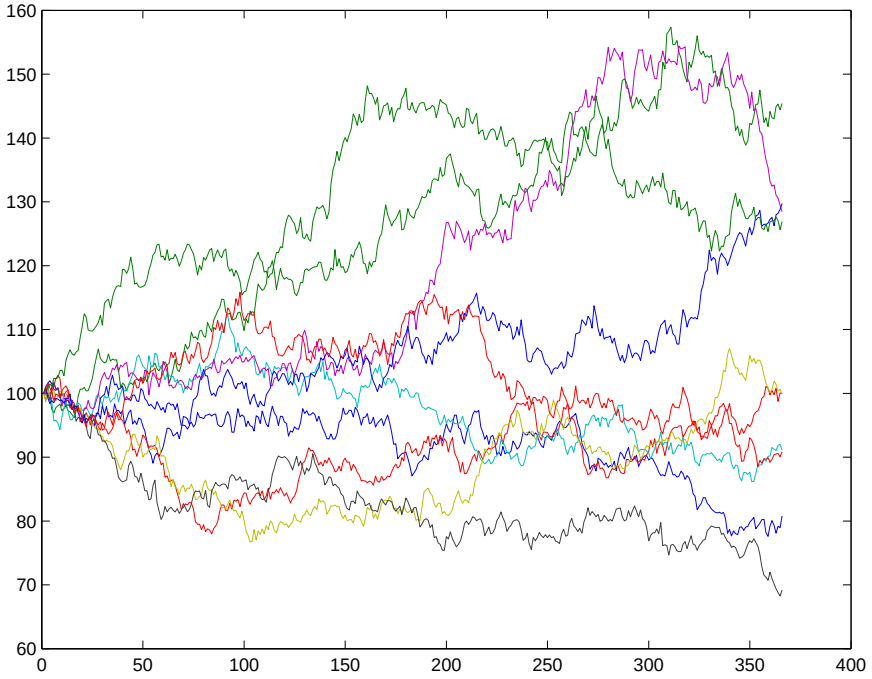
Es decir, podemos interpretar nuestra hipótesis fundamental sobre el subyacente diciendo que los **rendimientos instantáneos** del subyacente vienen dados por una **distribución normal** de media $\mu\Delta t$ y varianza $\sigma^2\Delta t$. Algo en principio, bastante razonable y que encuentra justificación a partir del análisis estadístico de las series de datos de los mercados financieros.

Queda, por supuesto, la tarea de determinar cuánto debe valer μ y σ.

Esto nos lleva al primer ejemplo que vamos a tratar que es **¿Cuál es la distribución de S en tiempo T ?**. En principio no la conocemos, pero dado que sabemos que los rendimientos tienen saltos normales podemos **"simular"** esta distribución.

```
0001 function BG = brownianogeometrico(S0,T,sigma,mu, nsaltos, nsim)
0002 %   Funcion para generar una aproximacion discreta de una
0003 %   simulacion de un browniano geometrico
0004 %   BG = brownianogeometrico(S0,T,sigma,mu, nsaltos, nsim)
0005 %
0006 %   INPUT:
0007 %
0008 %   S0:     Valor inicial del subyacente/browniano geometrico.
0009 %   T:      Tiempo hasta vencimiento(en aos)
0010 %   sigma:  Volatilidad del BG.
0011 %   mu:     Termino de deriva.
0012 %   nsaltos:Numero de saltos en la discretizacion.
0013 %   nsim:   Numero de simulaciones.
0014 %
0015 %   OUTPUT:
0016 %
0017 %   BG:     Matriz nsim x (nsaltos + 1) columnas con el resultado de la
0018 %           simulacion
0019
0020 dt = T/nsaltos;
0021
0022 N = randn([nsim nsaltos]);
0023 N = 1 + mu*dt + sigma*sqrt(dt)*N;
0024 BG = zeros(nsim, nsaltos+1);
0025
0026 BG(:,1) = S0;
0027 BG(:,2:end) = N;
0028
0029 BG = cumprod(BG,2);
```





Ejercicio

Comprobar el efecto que tiene asignar distintos valores de μ y σ

Utilización del lema de Ito.

Lo que hemos hecho hasta ahora no es más que simular una aproximación discreta de la distribución final. El siguiente paso sería calcular dicha distribución con las herramientas que nos proporciona el cálculo estocástico. Usando el lema de Ito se puede comprobar que la distribución final de S_T viene dada por:

$$S_T = S_0 \exp\left\{\left(\mu - \frac{\sigma^2}{2}\right) T + \sigma\sqrt{T}\xi\right\}$$

Donde ξ es, de nuevo, una normal standard. Es decir, el subyacente a vencimiento sigue una distribución lognormal de parámetros $\left(\mu - \frac{\sigma^2}{2}\right) T$, $\sigma\sqrt{T}$.

Ejercicio

A partir de la discretización anterior, haciendo $\Delta t \to 0$ y usando las propiedades de la distribución normal se puede calcular la distribución de S_T sin necesidad del Lema de Ito.

El código para simular ahora el subyacente a vencimiento sería:

```
0001 function escenarios = escenariosLN(S0,T,sigma,mu,nsaltos, nsim)
0002 %   Funcion para generar una simulacion exacta de un browniano geometrico
0003 %   escenarios = brownianogeometrico(S0,T,sigma,mu, nsaltos, nsim)
0004 %
0005 %   INPUT:
0006 %
0007 %   S0:         Valor inicial del subyacente/browniano geometrico.
0008 %   T:          Tiempo hasta vencimiento(en aos)
0009 %   sigma:      Volatilidad del BG.
0010 %   mu:         Termino de deriva.
0011 %   nsaltos:    Numero de saltos en la discretizacion.
0012 %   nsim:       Numero de simulaciones.
0013 %
0014 %   OUTPUT:
0015 %
0016 %   escenarios:     Matriz nsim x (nsaltos + 1) columnas con el resultado de la
0017 %           simulacion
0018
0019 N = randn([nsim nsaltos]);
0020 escenarios = zeros( nsim, nsaltos + 1 );
0021
0022 dt = T/nsaltos;
0023
0024 escenarios(:,1) = S0;
0025 escenarios(:,2:end) = exp((mu- sigma*sigma*0.5)*dt + sigma*sqrt(dt)* N);
0026
0027 escenarios = cumprod(escenarios,2);
0028
0029
```

Como comentario señalar que en esta versión de la simulación también pedimos como input ***nsaltos*** que es el número de saltos que damos hasta vencimiento, al igual que en la primera versión. Pero mientras que en la primera versión este número debía ser necesariamente alto para obtener una buena aproximación con el consiguiente coste computacional, en esta versión basta con dar un único salto para obtener una ***simulación exacta***. La razón para seguir incluyendo este input será aparente en breve.

Black-Scholes

Hasta ahora lo único que hemos utilizado son principios del cálculo estocástico que son de alguna forma independientes de la Matemática Financiera.

El siguiente paso es fundamental, y, de hecho, le valió el premio Nobel a Scholes y Merton (Fischer Black había fallecido con anterioridad). Bajo ciertas hipótesis entre las que cabe destacar la ***Ausencia de Oportunidades de Arbitraje*** el valor de un contrato que paga un cierto *payoff(S,K,....)* en un tiempo futuro T viene dado por:

$$\boxed{V(0) = e^{-rT} E[payoff(S, K, ...)]}$$

donde r es el tipo libre de riesgo. El término e^{-rT} representa el descuento a día de hoy de un cierto pago futuro. El valorar un producto como una Esperanza de un posible pago futuro coincide con la intuición que tenemos todos a partir de la teoría de juegos. Pero ojo porque la razón es mucho más profunda y de hecho esta esperanza se calcula suponiendo que $\boxed{\mu = r}$. Esto es consideramos que S evoluciona según:

$$\frac{dS_t}{S_t} = rdt + \sigma dW_t$$

y que por tanto, para la valoración correcta de cualquier derivado no interviene nada que tenga que ver con posibles rentabilidades medias pasadas o estimaciones de rentabilidades futuras (como poco es chocante ¿no?).

En concreto el valor de la Call vendrá dado por:

$$V(0) = e^{-rT} E[(S_T - K)^+]$$

Esta esperanza tiene fórmula explícita:

$$\text{Valor Call} : C(S, K, T, \sigma, r) = SN(d_1) - Ke^{-rT} N(d_2)$$
$$\text{Valor Put} : P(S, K, T, \sigma, r) = Ke^{-rT} N(-d_2) - SN(-d_1)$$

Donde:

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

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

Que son las fórmulas de Black-Scholes. *Valor Put* corresponde al valor de una opción Put de venta (derecho de vender una acción a un precio fijado hoy K), su payoff viene dado por $(K - S_T)^+$.

Lo que haremos en el siguiente código es valorar una opción Call, una Put por Montecarlo y compararlo con el precio obtenido de la fórmula explícita.

```
0001 S0 = 100;
0002 T = 1;
0003 sigma = 0.2;
0004 mu = 0.05;
0005 r = 0.05;
0006 K = 90;
0007
0008 nsaltos  = 1;
0009 nsim = 1000;
0010 tic
0011 escenarios = escenariosLN(S0,T,sigma,mu, nsaltos, nsim);
0012 toc
0013 tic
0014 escenarios2 = escenariosLN2(S0,T,sigma,mu, nsaltos, nsim);
0015 toc
0016
0017 escenarios = escenarios(:,end);
0018
0019 payoffCall = max(escenarios - K, 0);
0020 payoffPut =  max(K - escenarios, 0);
0021
0022 valorCall = mean(payoffCall)*exp(-r*T);
0023 valorPut  = mean(payoffPut)*exp(-r*T);
0024
0025 %Paridad Call Put (valorCall + dinero = valorPut + S0)
0026
0027 v0 = valorCall + K *exp(-r*T);
0028 v1 = valorPut + S0;
0029 if v0 == v1
0030     disp('Paridad Call Put se cumple');
0031 else
0032     disp('Hay algo mal');
```

```
0033 end
0034
0035
0036 d1 = (log(S0/K) + (r + sigma*sigma*0.5)*T)/(sigma*sqrt(T));
0037 d2 = d1 - sigma*sqrt(T);
0038
0039 valorExactoCall = S0 * normcdf(d1) - K *exp(-r*T) * normcdf(d2);
0040 valorExactoPut  = -S0 * normcdf(-d1) + K *exp(-r*T) * normcdf(-d2);
0041
0042 %Paridad Call Put (valorCall + dinero = valorPut + S0)
0043
0044 v0 = valorExactoCall + K *exp(-r*T);
0045 v1 = valorExactoPut + S0;
0046 if v0 == v1
0047     disp('Paridad Call Put se cumple');
0048 else
0049     disp('Hay algo mal');
0050 end
```

Ejercicio

Comprobar la paridad Call-Put

El código de la función escenariosLN2 es:

```
0001 function escenarios = escenariosLN2(S0,T,sigma,mu,nsaltos, nsim)
0002 %   Funcion para generar una simulacion exacta de un browniano geometrico
0003 %   escenarios = brownianogeometrico(S0,T,sigma,mu, nsaltos, nsim)
0004 %
0005 %   INPUT:
0006 %
0007 %   S0:         Valor inicial del subyacente/browniano geometrico.
0008 %   T:          Tiempo hasta vencimiento(en aos)
0009 %   sigma:      Volatilidad del BG.
0010 %   mu:         Termino de deriva.
0011 %   nsaltos:    Numero de saltos en la discretizacion.
0012 %   nsim:       Numero de simulaciones.
0013 %
0014 %   OUTPUT:
0015 %
0016 %   escenarios:     Matriz nsim x (nsaltos + 1) columnas con el resultado de la
0017 %           simulacion
0018
0019 N = randn([1 nsim]);
0020 escenarios = zeros(  nsaltos + 1 , nsim );
0021
0022 dt = T/nsaltos;
0023
0024 escenarios(1,:) = S0;
0025 escenarios(2:end,:) = exp((mu- sigma*sigma*0.5)*dt + sigma*sqrt(dt)* N);
0026
0027 escenarios = cumprod(escenarios,1);
```

¿Cuál es la diferencia?

Valoración de productos con dependencia del camino

Si nuestro problema fuese valorar únicamente derivados *europeos* (aquellos cuyo valor depende del nivel de subyacente a vencimiento) no estaríamos hablando de valoraciones por simulación, bastaría con integrar la función de pago a vencimiento contra la densidad adecuada. Pero ¿qué pasa cuando tenemos un producto que paga a vencimiento según la siguiente fórmula?:

$$Payoff = \left(\frac{1}{n} \sum_{i=1}^{n} S_i - K \right)^+$$

Donde S_i es el subyacente observado en tiempo $t_i \in [0, T] \forall i$. A este contrato se le conoce como *Call Asiática* y es un ejemplo de derivado cuyo payoff depende del camino que haya seguido el subyacente.

Para esta *Call Asiática* no existe fórmula de valoración explícita y no nos queda otro remedio por tanto que simular los posibles caminos del subyacente para calcular el pago en función de dicho camino.

```
0001 S0 = 100;
0002 T = 1;
0003 sigma = 0.2;
0004 mu = 0.05;
0005 r = 0.05;
0006 K = 90;
0007
0008 nsaltos  = 12;
0009 nsim = 10000;
0010 escenarios = escenariosLN(S0,T,sigma,mu, nsaltos, nsim);
0011
0012 mediaAsiatica = sum(escenarios(2:end),2)/nsaltos;
0013 payoffCall = max(mediaAsiatica - K, 0);
0014
0015 valorAsiatica = mean(payoffCall)*exp(-r*T);
```

Valoración de productos multisubyacente

La cosa se complica un poco si nos planteamos valorar un producto multisubyacente como haremos en el siguiente ejemplo.

Queremos poner precio a una *Call Asiática sobre una cesta de subyacentes*. La fórmula del payoff vendrá dada por:

$$Payoff = \left(\frac{1}{n} \sum_{i=1}^{n} \bar{S}_i - K \right)^+$$

donde $\bar{S}_i = \frac{1}{2}(S_i^1 + S_i^2)$ es la media de 2 activos vistos en tiempo t_i.

La dinámica vendrá dada por:

$$\frac{dS_t^1}{S_t^1} = rdt + \sigma^1 dW_t^1$$
$$\frac{dS_t^2}{S_t^2} = rdt + \sigma^2 dW_t^2$$

dW_t^1 dW_t^2 son dos brownianos con **correlación** ρ. Lo podemos interpretar pensando que las normales que generan ambas dinámicas tienen correlación ρ. De forma análoga a como se hace en el caso de un único subyacente, el lema de Ito nos da las siguientes soluciones para este sistema de ecuaciones diferenciales estocásticas:

$$S_T^1 = S_0^1 \exp\left\{\left(r - \frac{\sigma^2}{2}\right)T + \sigma^1\sqrt{T}\xi^1\right\}$$
$$S_T^2 = S_0^2 \exp\left\{\left(r - \frac{\sigma^2}{2}\right)T + \sigma^2\sqrt{T}\xi^2\right\}$$

con ξ^1 y ξ^2 normales standard con correlacion ρ.

El problema ahora consistirá en ser capaces simular normales con correlación, ya que un ordenador lo que genera de forma "natural" son variables independientes.

Para solucionar este problema utilizamos una propiedad "estupenda" de la distribución normal que es:

- Sean X_i variables aleatorias normales independientes.
- Sea R una matriz de correlación.
- Sea M una *raíz cuadrada* de R, es decir, $R = M^t M$
- $Y_i = X_i M^t$ son variables aleatorias normales con correlación R.

Generar variables aleatorias con correlación para distribuciones distintas de la normal, es un problema que no es en absoluto trivial.

Típicamente se escoge como raíz cuadrada la descomposición de choleski: de todas las raices cuadradas, aquella que es triangular superior.

```
0001 function escenarios = escenariosLN2suby(S0, T,sigma, rho, mu,nsaltos, nsim)
0002 %   Funcion para generar una simulacion exacta de un browniano geometrico
0003 %   escenarios = brownianogeometrico(S0,T,sigma,mu, nsaltos, nsim)
0004 %
0005 %   INPUT:
0006 %
0007 %   S0:         Vector con los Valores inicial de los subyacentes.
0008 %   T:          Tiempo hasta vencimiento(en aos)
0009 %   sigma:      Vector con las Volatilidades.
0010 %   mu:         Terminos de deriva.
0011 %   nsaltos:    Numero de saltos.
0012 %   nsim:       Numero de simulaciones.
0013 %
0014 %   OUTPUT:
0015 %
0016 %   escenarios:     Matriz nsim x nsubyacentes x (nsaltos + 1) columnas
0017 %                    con el resultado de la simulacion
0018 %
0019
0020 nsuby = length(S0);
0021 X = randn([nsim nsuby nsaltos]);
0022
0023 N = X*0;
0024
0025 dt = T/nsaltos;
0026
0027 ch = chol(rho);
0028 sigmaMatriz = diag(sigma.*sqrt(dt));
0029
0030 for i = 1:nsaltos
0031     N(:,:,i) = X(:,:,i)*ch*sigmaMatriz;
0032 end
```

```
0033
0034 dt = T/nsaltos;
0035
0036 escenarios = zeros([nsim nsuby nsaltos+1]);
0037 escenarios(:,:,1) = repmat(S0, nsim,1);
0038
0039 derivaln = repmat((mu- sigma.*sigma.*0.5)*dt,[nsim,1,nsaltos]);
0040
0041 escenarios(:,:,2:end) = exp(derivaln + N);
0042 escenarios = cumprod(escenarios,3);
0043
0044
```

Algunos problemas abiertos de la Computación Científica en Finanzas

La siguiente es una lista no exhaustiva que nos puede servir para hacernos una idea de problemas abiertos con utilidad práctica en finanzas. Hay muchos más, seguramente más interesantes, pero estos los podemos entender con lo que hemos visto hasta ahora:

- Integración rápida de gaussianas multivariantes.
- Integración rápida de opciones asiáticas.
- Integración rápida de opciones Range Accrual.
- Generación de distribuciones multivariantes no gaussianas con correlación.

Lo bueno que tienen, los tres primeros al menos, es que con el paso del tiempo y aunque no los solucionemos, dejarán de ser un problema.

Bibliografía

- Options, Futures and other Derivatives
(John C. Hull)

- Derivatives
(Paul Wilmott)

- Financial Calculus: An Introduction to Derivative Pricing
(Martin Baxter, Andrew Rennie)

- Monte Carlo Methods in Financial Engineering.
(Paul Glasserman)