BANCO CENTRAL DE RESERVA DEL PERÚ

Valorización de bonos estructurados

Omar Pinedo*

* Banco Central de Reserva del Perú

DT. N° 2015-016
Serie de Documentos de Trabajo
Working Paper series
Diciembre 2015

Valorización de bonos estructurados

Omar Pinedo[*]

Banco Central de Reserva del Perú

omar.pinedo@bcrp.gob.pe

Resumen

Se propone una metodología para la valorización de bonos estructurados que consiste en dos módulos principales: (1) construcción de curvas cupón cero para descontar los flujos de caja y (2) simulaciones para los flujos de caja de la opción implícita. Para (1) se aplica una metodología de B-Splines penalizados y para (2) se usa un Cuasi Monte Carlo que emplea una secuencia generalizada de Halton.

Abstract

A method to price structured bonds is proposed. It includes two key elements: (1) construction of zero coupon curves to discount cash flows and (2) simulation of cash flows for the underlying option. For (1) the study uses the penalized B-Spline method and for (2) it uses a Quasi-Monte Carlo based on a generalized Halton sequence.

Keywords: Structured Bonds, Quasi Monte Carlo, P-Splines, Yield curve estimation, Term structure of interest rates, Derivatives valuation.

[*] Estudio presentado en el Encuentro de Economistas 2013. El autor agradece a Carlos Barrera por sus acertadas recomendaciones, a Marco Vega por la paciencia y a sus jefes directos por la libertad otorgada para investigar.

1. Introducción

El problema de la valorización de bonos estructurados que no cotizan de forma continua en un mercado tiene una complejidad producida por la dificultad de obtener algún estimado del valor de un bono que no cotiza y de una opción que no siempre tiene una solución analítica.

En ambos casos la mejor opción es realizar una valorización relativa: el bono se valoriza respecto de otros bonos líquidos comparables y la opción mediante algún método equivalente al cálculo del precio del sintético que la reproduce. Para valorizar el bono se construye una curva cupón cero con la que se descuenta los flujos generados hasta el vencimiento y para la opción se realiza simulaciones del flujo de caja al día de vencimiento y se trae a valor presente el promedio de esos flujos con la curva cupón cero antes construida.

Para la construcción de curvas cupón cero hay dos metodologías paramétricas que son de uso común: las propuestas en Nelson & Siegel (1987) y Svensson (1994). Estas metodologías son muy empleadas en modelos dinámicos de equilibrio general debido a su parsimonia y tratabilidad matemática; sin embargo, su precisión no es suficiente en el marco de la valorización relativa de instrumentos financieros Choudhry (2005), Ait-Sahalia & Duarte (2003), Poletti, Laurini & Moura (2010). Esta situación obliga la revisión del campo de las metodologías no paramétricas, donde se busca aproximar la curva usando una combinación lineal de funciones polinomiales.

Existen distintas formas de especificar las funciones polinomiales que se usarán para el cálculo: splines polinomiales, splines cúbicos, polinomios de Bernstein o curvas de Bézier; sin embargo, aquí uso B-Splines, que son una generalización de las curvas de Bézier, debido a la robustez, simplicidad y eficiencia del algoritmo que realiza su cálculo numérico y las propiedades teóricas de ser capaz de reproducir,

mediante una adecuada combinación lineal, cualquier polinomio en el dominio especificado y eliminar la multicolinealidad.

Para valorizar el flujo de caja de la opción implícita el método más común es usar simulaciones Monte Carlo; sin embargo, el principal problema que conlleva el aplicar este método es la excesiva carga computacional requerida para alcanzar la convergencia del resultado final. Para lidiar con este problema se busca emplear métodos y suposiciones no vinculantes que permitan aliviar la carga computacional.

Esto se logra cambiando el método de muestreo de números aleatorios: en lugar de usar un generador de números pseudo-aleatorios empleamos uno de números cuasi-aleatorios, lo cual hace que el método que se emplea para el cálculo del precio de la opción sea un Cuasi Monte Carlo.

Por último, el cálculo de una prima por iliquidez del instrumento escapa al marco teórico de la valorización relativa, por lo cual se supone que el tenedor mantendrá su posición hasta el vencimiento. Con esta suposición, el valor razonable del instrumento coincidiría con el valor del sintético conformado por la opción y el bono.

2. Construcción de curvas cupón cero

Para la construcción de las curvas cupón cero se empleará una metodología de regresión con B-Splines penalizados basada en Eilers & Marx (1996).

2.1 B-Splines

El objetivo de los B-Splines es generar una curva suave que aproxime un conjunto de puntos que se presume representan alguna función $f(x)$. La forma de lidiar con el problema es sencilla: generar varias funciones "básicas" a lo largo del dominio especificado, que debe ser muy similar al del conjunto de puntos, y, empleando una regresión, encontrar una combinación lineal de dichas funciones que aproxime en buena medida los puntos.

Primero, se generan las funciones básicas usando el algoritmo propuesto en De Boor (1972), donde $\boldsymbol{t} = (t_j)$, el vector de nodos, es una secuencia no decreciente de al menos $d + 2$ números reales y el j-ésimo B-Spline de grado d con nodos $\boldsymbol{t}$ se define como:

$$B_{j,d,t}(x) = \frac{x - t_j}{t_{j+d} - t_j} B_{j,d-1,t}(x) + \frac{t_{j+1+d} - x}{t_{j+1+d} - t_{j+1}} B_{j+1,d-1,t}(x)$$

$$B_{j,0,t}(x) = \begin{cases} 1, & si\ t_j \leq x \leq t_{j+1} \\ 0, & de\ otro\ modo \end{cases}$$

Este algoritmo tiene dos propiedades computacionales clave: es numéricamente estable y es eficiente. Por otra parte, tiene varias propiedades teóricas que son de interés, cuyas demostraciones pueden encontrarse en (Lyche & Mørken, 2008):

1. Localidad de los nodos: el j-ésimo B-Spline de grado $d, B_{j,d}$, depende únicamente de los nodos $t_j, t_{j+1}, \dots, t_{j+d+1}$.

2. Soporte local: si $x \notin [t_j, t_{j+d+1}\rangle \rightarrow B_{j,d} = 0$. Aún si $t_j = t_{j+d+1}$, $B_{j,d} = 0$.

3. Positividad: si $x \in [t_j, t_{j+d+1}\rangle \rightarrow B_{j,d} > 0$.

4. Suavidad: si un nodo aparece m veces ($t_j = t_{j+1} = \cdots = t_{j+m-1} = z$) las derivadas de orden $0,1,2,\ldots,d-m$ de $B_{j,d}$ son continuas en z.

5. Independencia lineal: Si ningún nodo se repite al menos $d+1$ veces, la única forma de obtener una combinación lineal de B-Splines nula es cuando los coeficientes son ceros.

6. Representatividad: con una cantidad suficiente de nodos (N) se puede reproducir cualquier polinomio de grado d en el dominio $[a, b]$, donde $t_{d+1} \leq a$ y $b \leq t_{N-d}$ como una combinación lineal de B-Splines.

Luego, se usa la aproximación de variación decreciente de Schöenberg para generar la curva que aproximará el conjunto de puntos:

$$\hat{f}(x) = \sum_{i=1}^{n} \beta_i B_{i,d}(x)$$

Aquí se constata que, haciendo una suposición plausible respecto del grado absoluto de la relación funcional que obedece el conjunto de puntos, se puede realizar una aproximación adecuada en el dominio objetivo usando un algoritmo de regresión. Sin embargo, quedarían por definir el vector de nodos y la cantidad de B-Splines que se emplearán para realizar la regresión, ya que se supone que el dominio objetivo es conocido de antemano.

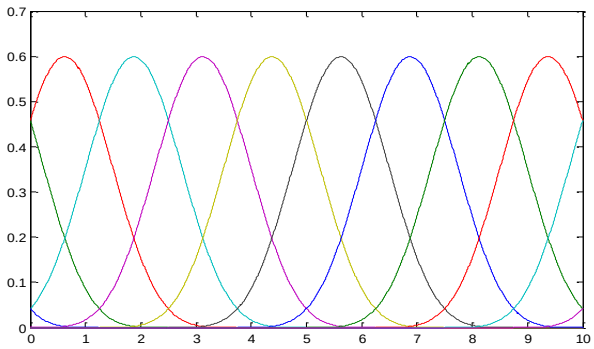

Fig. 1: Doce B-Splines de cuarto grado en el dominio $[0,10]$ generados con un vector de nodos equidistantes con nodos fantasmas hacia los extremos.

Los nodos son un concepto clave en el proceso de construcción de los B-Splines. Pueden ser interpretados como los puntos entre los cuales se interpolarán las funciones polinomiales que deseamos construir. Al momento de construir los B-Splines se debe responder a dos cuestiones respecto de los nodos: qué se hará para otorgar soporte suficiente a los extremos del dominio objetivo, donde usualmente no hay una cantidad suficiente de splines para aproximar la curva objetivo, y cómo se determinará la distancia entre un nodo y el siguiente.

Para responder a lo primero, se escoge la alternativa de los nodos fantasmas (nodos fuera del dominio objetivo), de forma que haya la misma cantidad de B-Splines a lo largo del dominio objetivo y tener continuidad en todas las derivadas de la curva, aún en los extremos. De este modo, descartamos la opción de nodos de multiplicidad en los extremos (repetir varias veces el nodo inicial y el nodo final), la cual no cumplía con brindar continuidad en las derivadas de mayor orden, y por tanto, con otorgar la misma suavidad en los extremos. Para responder a lo segundo, la convención para generar el vector de nodos es usar nodos equidistantes o nodos por cuantiles, con lo cual el vector de nodos queda determinado por la elección del número de B-Splines por emplear.

El uso de nodos equidistantes con nodos fantasma simplifica en gran medida la generación de las funciones básicas, ya que brinda una serie de relaciones entre el número de B-Splines (S), la cantidad de nodos interiores (IK), la cantidad total de nodos (TK), el tamaño del intervalo $(step)$, el grado (deg) escogido y los límites superior (ub) e inferior del dominio (lb), que son de gran utilidad al momento de implementar una librería de generación de funciones básicas:

$$IK = S - deg + 1$$

$$TK = S + deg + 1$$

$$step = (ub - lb)/(IK - 1)$$

$$Knot_i = lb + step(i - 1 - deg), \quad i \in \mathbb{N} \wedge i \in [1, TK]$$

2.2 P-Splines

Esta técnica se caracteriza por realizar una regresión que incluya una penalidad si los $\widehat{\beta_\iota}$ son muy distintos entre sí, lo cual controla cualquier sobreajuste. Con esta metodología podemos restar importancia a la elección del número de B-Splines, ya que podríamos incluir un número relativamente alto y evitar el sobre ajuste que implicaría realizar una regresión por mínimos cuadrados ordinarios sobre ese espacio. La función de costo propuesta es:

$$Q = \left(y - \hat{\beta}X\right)' W \left(y - \hat{\beta}X\right) + \lambda (D_k \hat{\beta})' (D_k \hat{\beta})$$

Donde y es la variable dependiente, X el conjunto de variables que forman el espacio sobre el que se regresiona, W es la matriz de pesos, D_k la representación matricial del operador de k -ésima diferencia Δ^k , S la cantidad de B-Splines empleados, N la cantidad de observaciones y $\hat{\beta}$ el vector de pesos estimados. Optimizando esta función de costo se obtiene el estimador:

$$\hat{\beta} = (X'WX + \lambda D_k{}' D_k)^{-1} X'Wy$$

Esta función de costo tiene dos partes bien diferenciadas: la primera es la fidelidad, es decir, el grado de ajuste de la curva, y la segunda es una medida de la rugosidad de la función. El *trade-off* entre ambos está controlado por λ, por lo cual su elección resulta crucial para la robustez del modelo. Aquí se siguen las recomendaciones dadas en Eilers & Marx (1996), según las cuales $k = 1$ y λ se escoge primero realizando validación cruzada generalizada (GCV), después usando el λ_{GCV} como insumo para el criterio de información de Akaike (AIC) y, por último, empleando λ_{AIC} en el cálculo de $\hat{\beta}$:

$$D_1 = \begin{bmatrix} -1 & 1 & 0 & & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -1 & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}_{(S-1)xS}$$

$$H = X(X'WX + \lambda D_k' D_k)^{-1} X'W \longrightarrow Hy = \hat{y}$$

$$GCV(\lambda) = \frac{(y - Hy)'(y - Hy)}{\left(N - Tr(H)\right)^2}$$

$$\hat{\sigma}_0^2 = \frac{(y - H_{GCV}y)'(y - H_{GCV}y)}{(N - 1)}$$

$$AIC(\lambda) = \frac{(y - Hy)'(y - Hy)}{\hat{\sigma}_0^2} + 2Tr(H) - Nln(2\pi\hat{\sigma}_0^2)$$

Para minimizar $GCV(\lambda)$ y $AIC(\lambda)$ las librerías de software de minimización de funciones suelen requerir como insumo un λ inicial. En Eilers & Marx (1996) se recomienda, para el caso de $GCV(\lambda)$, usar $\lambda = 0$, que es el caso de regresión sin penalización y, para el caso de $AIC(\lambda)$, λ_{GCV}. Estas mismas librerías suelen requerir también el gradiente de las funciones por optimizar (pueden hallarse los gradientes y la matemática correspondiente en el Anexo 1).

Un tema bastante interesante que surge en este tipo de regresiones es el de la dimensión efectiva de $\hat{\beta}$. En el caso de una regresión sin penalizaciones, la dimensión efectiva sería S. Pero, en el caso de P-Splines, la penalización restringe los grados de libertad de la distribución posterior de $\hat{\beta}$, por lo que $\dim(\hat{\beta}) \leq S$. Esto dificulta el cálculo de criterios de información como el de Akaike, que basan su cálculo en la dimensión efectiva de $\hat{\beta}$. En este documento se emplea la aproximación de Hastie & Tibshirani (1990), que emplea la traza de la matriz proyectiva H, que en la literatura de B-Splines es conocida como matriz de influencias, $\dim(\hat{\beta}) = Tr(H)$. Es fácil notar que $Tr(H) = S$ para el caso en que $\lambda = 0$.

Con este aparato teórico, dependiendo de cómo especifique las variables, se pueden aproximar curvas par o curvas cupón cero. Las primeras son de construcción más sencilla y directa: cada observación de la regresión representaría un bono, y_{Sx1} sería el vector de yields de los bonos, X_{NxS} sería la matriz que contiene el valor de los S B-Splines calculados para el plazo de vencimiento de cada papel y se asignan pesos a cada bono según algún indicador de liquidez.

La construcción de la curva cupón cero no es tan directa. Lo que se empleó aquí es la relación según la cual el precio de un bono debería de ser igual a la suma de sus flujos de caja descontados con un factor de descuento que disminuye cuando el flujo está más lejos en el tiempo. Entonces, primero es necesario modelar la curva de factores de descuento para, a partir de esta, obtener la curva cupón cero. Esto implica generar una matriz de flujos de caja de todos los bonos que se emplean para la regresión, por ejemplo, para tres bonos distintos:

$$CF = \begin{vmatrix} 100 & 0 & 0 & 0 & 0 \\ 0 & 2.50 & 0 & 100 & 0 \\ 0 & 0 & 1.25 & 0 & 100 \end{vmatrix}$$

Donde cada fila representa el flujo de caja de un instrumento y cada columna contiene todos los flujos de caja que se recibirán en una fecha específica. Luego, la ecuación empleada es: $CF_{NxT} \cdot \hat{\delta}_{Tx1} + \varepsilon = P_{Nx1}$, donde T es la cantidad de plazos distintos en los que se reciben todos los flujos de caja. Adicionalmente, $\hat{\delta}_{Tx1} = BS_{TxS} \cdot \hat{\beta}_{Sx1} \longrightarrow (CF_{NxT} \cdot BS_{TxS}) \cdot \hat{\beta}_{Sx1} + \varepsilon = P_{Nx1}$. Con esto se ve de forma más directa que y_{Sx1} sería el vector de precios de los bonos y que X_{NxS} sería el resultado de multiplicar la matriz de flujos de caja por la matriz que contiene los S B-Splines calculados para cada plazo distinto en que se recibe un flujo de caja.

Únicamente falta especificar la forma que tendrá la matriz de pesos. En este punto seguimos lo propuesto en Subramanian (2001), que es modelar los pesos usando tangentes hiperbólicas sobre algún indicador de liquidez. La ventaja que tiene esta forma funcional es que mientras el indicador de liquidez sea cercano a uno, los pesos no presentarán diferencias significativas; sin embargo, para los bonos ilíquidos, el peso disminuiría rápidamente.

El concepto implícito detrás de asignar pesos por liquidez es la suposición de que, si un papel es más líquido, su precio de mercado (que es el observado) es más fiable y tiene menor cantidad de ruido; por lo tanto, aportará mayor cantidad de información a la regresión y se le asigna un mayor peso en la función de costo. Aquí se usa el spread $s_i = P_i^{ask} - P_i^{bid}$ para calcular como

indicador de liquidez a $\ell_i = 1 - s_i/s_{max}$. Con esto se calcula $w_i = \tanh(\ell_i)$ y la matriz W_{NxN} de pesos sería una matriz diagonal con los N w_i.

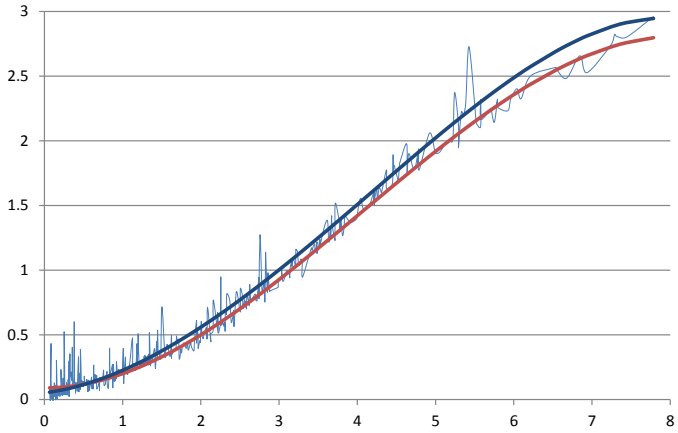

Fig. 2: Curva par (rojo) y curva cupón cero (azul) aproximadas con los bid yields (linea celeste delgada) de 593 instrumentos de deuda de agencias gubernamentales y organismos supranacionales. La diferencia entre las curvas se acentúa con el incremento del plazo, pero tienen una forma muy similar.

3. Cuasi Monte Carlo

3.1 Monte Carlo

Al valorizar opciones se afronta el problema de una integral múltiple que, por lo general, no tiene solución analítica. Para esto se recurre al método de evaluación numérica Monte Carlo:

$$I(g) = \int_{[0,1]^d} g(X)dX$$

$$Q(g) = \frac{1}{N}\sum_{l=1}^{l=N} g(X_l)$$

$$Var_{MC}(g) = \int_{[0,1]^d} [g(X) - I(g)]^2 dX$$

$$P_N = \{X_1, X_2, \dots, X_N\} = (x^1, x^2, \dots, x^d)$$

La esperanza del estimador $Q(g)$ será igual a $I(g)$ y su varianza $Var_{MC}(g)/N$ siempre que los puntos P_N (las llaves denotan vectores columna y los paréntesis vectores fila) sean muestras de una distribución uniforme en $[0,1]^d$ y con $g(x)$: $[0,1]^d \rightarrow \mathbb{R}$ de norma al cuadrado integrable.

En este caso, la función g es el flujo de caja a la fecha de madurez del instrumento, por lo que depende del valor de los activos subyacentes a dicha fecha. Se hace la suposición de que el logaritmo del precio de cada activo sigue un proceso de Wiener:

$$S_{i,T} = S_{i,0} Exp\left[T\left(\mu_i - 0.5\sigma^{i,i}\right) + (T\sigma^{i,i})^{0.5}\Phi^{-1}\left(x^i\right)\right] \rightarrow E_{t=0}(S_{i,T}) = S_{i,0}e^{\mu_i}$$

$$r_{i,t} = \ln\left({S_t}/{S_{t-1}}\right) \rightarrow \mu_i = \frac{1}{T}\sum_{t=-V}^{t=0} r_{i,t}$$

Donde Φ^{-1} es la función inversa de la distribución normal estándar acumulada y supongo que el retorno futuro medio será igual al promedio de los últimos V retornos históricos. Asimismo, se realiza la suposición de que tendrán una varianza igual al promedio ponderado del cuadrado de los retornos diarios de los últimos V periodos, donde los pesos seguirán un patrón de decaimiento exponencial:

$$\sigma_{t+1}^{i,j} = (1-\lambda) r_{i,t} r_{j,t} + \lambda \sigma_t^{i,j}$$

Así, la función g de flujo de caja de la opción termina dependiendo del conjunto P_N de números extraídos de una distribución uniforme en $[0,1]^d$.

En la práctica, la simulación Monte Carlo tradicional tiene los siguientes pasos:

1. Se obtienen números pseudo-aleatorios no correlacionados a extraídos de una distribución uniforme.
2. Se aplica la función inversa de la distribución normal estándar acumulada, obteniendo números pseudo-aleatorios que siguen una distribución normal estándar multivariada (media cero y matriz de covarianzas igual a la identidad).

3. Se aplica la descomposición de Cholesky a la matriz de covarianzas objetivo (que se calcula con los factores de decaimiento exponencial).
4. Se multiplica la matriz de Cholesky por la matriz de números pseudo-aleatorios no correlacionados y se suma la media objetivo de cada serie.
5. Se obtienen simulaciones de retornos de los activos correlacionados.

El método anterior es de aplicación directa; sin embargo, la carga computacional requerida para alcanzar la convergencia del resultado final es muy grande, ya que la desviación estándar del error de estimación es proporcional a $N^{-0.5}$, lo cual implica que, para reducirlo a la mitad, se deben cuadruplicar la cantidad de simulaciones. Para acelerar la convergencia se realizan dos modificaciones al Monte Carlo tradicional antes descrito:

1. Para la función inversa de la distribución normal estándar acumulada se usa el algoritmo de Acklam (2004), que es más rápido y brinda una mejor aproximación que el algoritmo de Abramowitz & Stegun (1964), que usa la mayoría de paquetes estadísticos y hojas de cálculo.

2. Para la generación de números "aleatorios" se emplean secuencias de números cuasi-aleatorios de baja discrepancia. Debido a que ya no se usan números pseudo-aleatorios, sino cuasi-aleatorios, el método pasa a llamarse Cuasi Monte Carlo. Se escoge la secuencia generalizada de Halton para la generación de números cuasi-aleatorios, ya que es la secuencia sobre la que hay mayor cantidad de material académico disponible.

3.1 Secuencia generalizada de Halton

De forma general, las secuencias de números cuasi-aleatorios están conformadas por puntos del hipercubo d-dimensional $[0,1)^d$ y están diseñadas para lograr un alto nivel de uniformidad con una cantidad relativamente pequeña de observaciones en la muestra. Para medir el grado de uniformidad de la secuencia P_N se emplea el concepto de discrepancia extrema D_∞^*, el cual se define formalmente como:

$$p(J, P_N) = \ card(P_N \cap J)/N - Vol(J)$$

$$D_\infty^*(P_N) = sup_{J \epsilon \ \mathcal{J}^*} |p(J, P_N)|$$

Donde $card(x)$ es el número cardinal del conjunto x, $sup(x)$ es el supremo del conjunto x, J es cualquier hipercubo contenido en $[0,1]^d$ y $\mathcal{J}^*$ es la familia de todos los hipercubos contenidos en $[0,1]^d$ que tienen una esquina en **0** y otra en cualquier punto de $[0,1]^d$. La función $p(J, P_N)$ es la diferencia entre la fracción de puntos de la secuencia P_N contenida en J y el volumen de J. La secuencia P_N tiene una distribución uniforme si y solo si $\lim_{N \to \infty} D_\infty^*(P_N) = 0$.

Esta medida de uniformidad nos permite acotar el error de estimación de la integral $I(g)$ mediante la desigualdad de Koksma-Hlawka:

$$\left| \int_{[0,1]^d} g(X) dX - \frac{1}{N} \sum_{l=1}^{l=N} g(X_l) \right| \leq V(g) D_\infty^*(P_N)$$

Donde $V(g)$ es la variación de $g(X)$ sobre $[0,1]^d$ en el sentido de Hardy-Krause (en Tuffin (1996-A) se puede encontrar más detalle sobre esta medida de variación) y es acotada. Se ve que para disminuir la cota del error es necesario disminuir $V(g)$o $D_\infty^*(P_N)$. La variabilidad es intrínseca al integrando, por lo que no podemos modificarla; sin embargo, la discrepancia depende del conjunto de puntos P_N que escogemos para realizar la integración numérica.

En este contexto, las secuencias de baja discrepancia aparecen como una posible mejora en la calidad de la estimación, ya que están diseñadas para obtener $D_\infty^*(P_N) = O(N^{-1} \log^{\alpha(d)} N)$, con lo que su error de estimación $|I(g) - Q(g)| = O(N^{-1} \log^{\alpha(d)} N)$, el cual, en muchos casos, es mejor que el $O(N^{-0.5})$ del error de estimación del Monte Carlo tradicional.

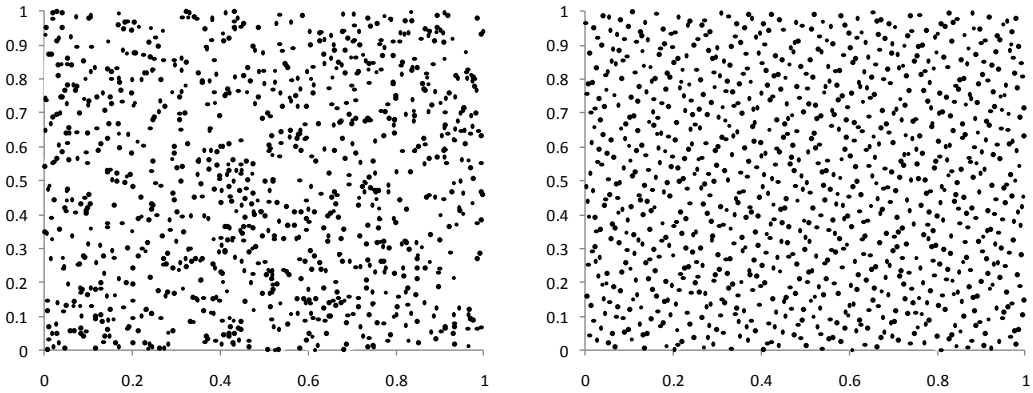

Fig. 3: comparación de la capacidad de mil puntos para llenar el dominio $[0,1]^2$ de forma uniforme usando números pseudo-aleatorios (izq) y una proyección sobre las dos primeras dimensiones de los números cuasi-aleatorios de la secuencia de Halton (der).

Existen varias secuencias de números cuasi-aleatorios, todas cumpliendo con acotar el error al menos con $O(N^{-1}\log^{\alpha(d)} N)$, donde $d/2 \leq \alpha(d) \leq d$, siendo las más conocidas las secuencias de Faure, Sobol, Halton y Niederreiter. Este documento se concentra en la secuencia de Halton por ser aquella que cuenta con mayor cantidad de material académico y empírico disponible.

La secuencia de Halton en $[0,1]^d$ se construye con el conjunto de bases $b = \{b_1, b_2, \dots, b_d\}$, todas primas entre sí (usualmente los primeros d números primos), la secuencia de enteros $E = \{1, 2, .., N\}$ y el hecho que cualquier número entero $e \geq 0$, siempre que la base $b_i \geq 2$ sea entera, puede ser escrito de la forma:

$$e = a_0 + a_1 b_i + a_2 b_i^2 + \cdots + a_q b_i^q = \sum_{p=0}^{\infty} a_p(e) b_i^p$$

Donde el dígito $a_p \in \{0, 1, \dots, b_i - 1\}$; es decir, e puede escribirse como $e_{b_i} = \overline{a_q \dots a_1 a_0}_{b_i}$ y

$$\phi_{b_i}(e) = a_0 b_i^{-1} + a_1 b_i^{-2} + \cdots + a_q b_i^{-(q+1)} = \sum_{p=0}^{\infty} a_p(e) b_i^{-(p+1)}$$

Es la función radical inversa de de e en base b_i, siendo posible escribirla como el simétrico en el punto decimal de e_{b_i}: $\phi_{b_i}(e) = \overline{0.a_0 a_1 \dots a_q}_{b_i}$ Luego, la secuencia de Halton se define como:

$$X_e = (\phi_{b_1}(e), \phi_{b_2}(e), \dots, \phi_{b_d}(e))$$

$$x_i = \{\phi_{b_i}(1), \phi_{b_i}(2), \dots, \phi_{b_i}(N)\}$$

$$H_d = \{X_1, X_2, \dots, X_N\} = (x_1, x_2, \dots, x_d)$$

Donde x_i es la secuencia de Van der Corput en base b_i. El problema principal de la secuencia de Halton es que, para más de 10 dimensiones, la discrepancia sufre un incremento drástico, producto de las correlaciones que surgen entre las secuencias de Van der Corput en bases grandes. Se han propuesto distintas soluciones tentativas a este problema y a estas secuencias modificadas se les denomina, en conjunto, secuencias generalizadas de Halton.

Entre las secuencias más discutidas tenemos la secuencia de Halton con saltos, que consiste en tomar secuencias de la forma $\phi_{b_i}(L_i e)$, donde L_i define el salto del muestreo, las secuencias aleatorizadas de Halton y las secuencias permutadas de Halton, que consisten en intercambiar los dígitos de e_{b_i} de alguna forma. En lo sucesivo nos centraremos en las secuencias permutadas de Halton con permutaciones digitales lineales de la forma:

$$\pi_{i;a_0,a_1,\dots,a_{p-1}}(a_p) = \sum_{c=0}^{c=p} f_{i,p,c}\, a_c + z_{i,p}\ (mod\ b_i)$$

$$\phi_{b_i}^{s}(e) = \sum_{p=0}^{\infty} \pi_{i;a_0,a_1,\dots,a_{p-1}}(a_p) b_i^{-(p+1)}$$

$$X_e^s = (\phi_{b_1}^s(e), \phi_{b_2}^s(e), \dots, \phi_{b_d}^s(e))$$

$$x_i = \{\phi_{b_i}^s(y_i + 1), \phi_{b_i}^s(y_i + 2), \dots, \phi_{b_i}^s(y_i + N)\}$$

$$GH_d = \{X_1^s, X_2^s, \dots, X_N^s\} = (x_1^s, x_2^s, \dots, x_d^s)$$

Donde $z_{i,p}$ es la permutación digital aleatoria propuesta en Tuffin (1996-A), y_i es el inicio aleatorio propuesto en (Wang & Hickernell, 2000), $f_{i,p,c}$ una versión general de los multiplicadores propuestos en (Atanasov, 2004).

Esta forma de presentar la construcción de la secuencia es lo más general posible y normalmente se usan sólo algunas modificaciones y no todas a la vez; sin embargo, es posible emplear varios tipos de generalizaciones a la vez. Nosotros emplearemos una secuencia generalizada de Halton permutada con los multiplicadores propuestos en Faure & Lemieux (2009), y aleatorizada con permutaciones digitales aleatorias e inicio aleatorio:

$$u \sim U[0,1] \ \wedge \ y_i \in \mathbb{Z}^+ \sim U[0,\infty)$$

$$z_i = u.b_i \ (mod\ 1)$$

$$\pi_i(a_p) = f_i a_p + z_i \ (mod\ b_i)$$

$$\phi_{b_i}^{s}(e) = \sum_{p=0}^{\infty} \pi_i(a_p) b_i^{-(p+1)}$$

$$X_e^s = (\phi_{b_1}^s(e), \phi_{b_2}^s(e), \dots, \phi_{b_d}^s(e))$$

$$x_i^s = \{\phi_{b_i}^s(y_i + 1), \phi_{b_i}^s(y_i + 2), \dots, \phi_{b_i}^s(y_i + N)\}$$

$$GH_d = \{X_1^s, X_2^s, \dots, X_N^s\} = (x_1^s, x_2^s, \dots, x_d^s)$$

Es decir, emplearemos solo un multiplicador, una permutación digital aleatoria y un inicio aleatorio para todas las observaciones de una misma dimensión, cambiándolos de dimensión en dimensión. Esta construcción nos permite aleatorizar las secuencias determinísticas que se obtienen empleando multiplicadores perturbando al mínimo la estructura de la secuencia subyacente, tal como se demuestra en Wang & Hickernell (2000) para el inicio aleatorio y en Faure (2005) para la permutación digital aleatoria.

Se incluye esta aleatorización con la finalidad de obtener un estimador de la varianza del error de estimación de la integral; sin embargo, es preciso señalar que las aleatorizaciones, por sí mismas, no son buenas soluciones al problema de correlación en altas dimensiones, ya que, por su construcción, realizan una perturbación mínima a la estructura subyacente. Es por esto que se realiza la aleatorización sobre una estructura permutada linealmente, las cuales quiebran en

mayor medida el problema de las correlaciones. Por otra parte, para estimar la integral no es necesario aleatorizar la secuencia, lo cual significa una ganancia adicional en tiempo de cálculo.

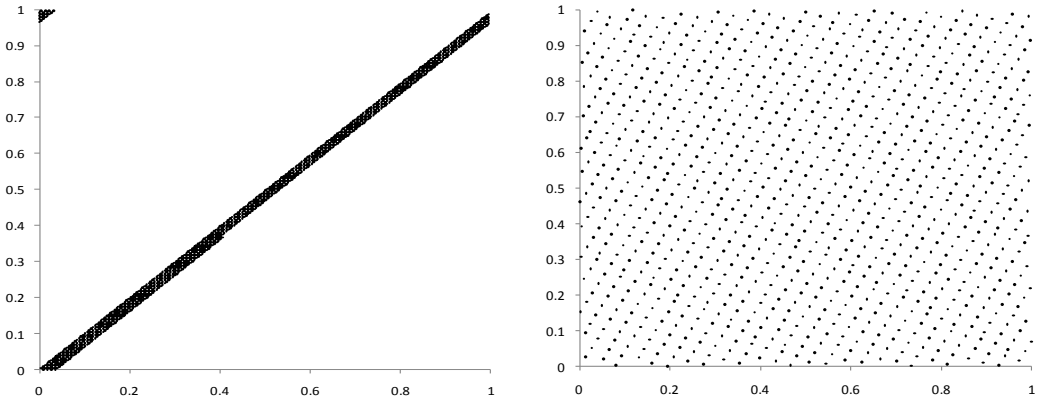

Fig. 4: comparación de la capacidad de mil puntos para llenar el dominio $[0,1]^2$ de forma uniforme de las proyecciones sobre las dimensiones 49 y 59 de la secuencia de Halton (izq) y la secuencia de generalizada de Halton con multiplicadores de Faure-Lemieux.

Otro punto importante es el hecho de que en Wang & Hickernell (2000) se emplea la transformada de Von Neumann-Kakutani de forma recursiva para el cálculo de la secuencia de Halton. Esto es posible gracias a que, tal como se demuestra en Pagès (1992), las secuencias de Van der Corput en base b_i son equivalentes al sistema ergódico generado por la transformada de Von Neumann-Kakutani en base b_i alrededor de 0. Esta forma de cálculo evita la necesidad de acotar y_i, pero hace muy impráctica la implementación de las permutaciones digitales, por lo cual, en nuestro algoritmo usamos 10^8 como una cota superior razonable para y_i.

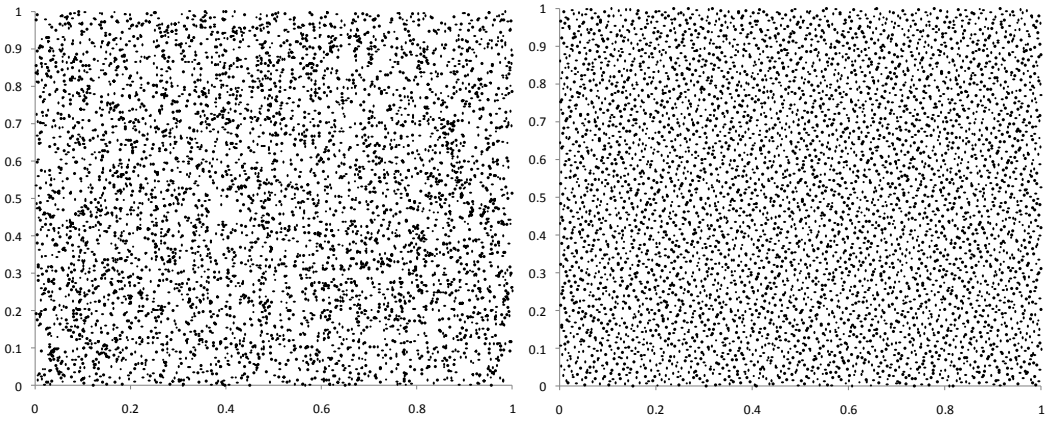

Fig. 5: comparación de la capacidad de cinco mil puntos para llenar el dominio $[0,1]^2$ de forma uniforme usando números pseudo-aleatorios (izq) y una proyección sobre las dos primeras dimensiones de los números cuasi-aleatorios de la secuencia de Halton (der).

3.3 Descomposición de Cholesky

Es una técnica matemática empleada para descomponer una matriz definida positiva en el producto de una matriz triangular inferior y su traspuesta: $\mathrm{L} = \mathrm{A} * \mathrm{A}^T$. La técnica es relevante para nuestra aplicación debido a que nos permite correlacionar los números aleatorios que se usan para la simulación.

Suponiendo que E_{NxM} es la matriz de números aleatorios no correlacionados, donde N es la cantidad de simulaciones y M es la cantidad de activos simulados, entonces definimos $Cov(E) = E^T * E = I_M$. Ahora suponemos que Σ_{MxM} es la matriz de covarianzas entre los activos simulados y hallamos su descomposición de Cholesky: $\Sigma = \mathrm{Z} * \mathrm{Z}^{\mathrm{T}}$. Por último, calculamos: $Cov(E * Z^T) = (E * Z^T)^T * (E * Z^T) = Z * (E^T * E) * Z^T = Z * Z^T = \Sigma$. Por tanto, la matriz $A = E * Z^T$ contiene números aleatorios con una matriz de covarianzas idéntica a la matriz objetivo Σ; es decir, los números aleatorios correlacionados que buscamos.

4. Breve comparación metodológica

A pesar de encontrarse fuera del enfoque de presentación metodológica, se consideró necesaria la inclusión de ejemplos que sustenten nuestras tesis de mejoras en la eficiencia de la valorización de activos de renta fija y de instrumentos derivados.

En el caso de la valorización de activos de renta fija se comparó la metodología de Nelson & Siegel (1987) con la metodología de P-Splines presentada en este documento. Construimos curvas cupón cero únicamente con papeles de gobierno en dólares americanos, en euros, y en dólares australianos. El intervalo de análisis es del 30/06/2014 al 18/03/2015, con 154 instrumentos en dólares americanos, 25 instrumentos en euros, y 13 instrumentos en dólares australianos, todos valorizados

diariamente. Principalmente analizamos el error cuadrático medio y el error absoluto medio, y acompañamos el análisis con un gráfico del error de valorización.

USD	Nelson-Siegel	P-Splines
Abs(e)	0.0444	0.0609
e^2	0.0220	0.0078

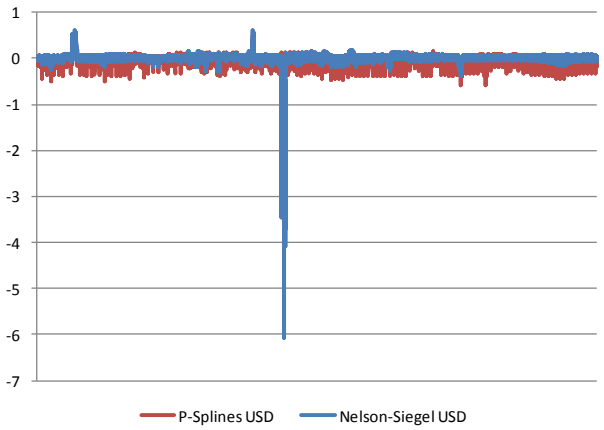

Fig. 6: Error de valorización diaria de 154 bonos del tesoro americano en USD desde el 30/6/2014 hasta el 18/03/2015, se compara la metodología de P-Splines con la de Nelson-Siegel.

EUR	Nelson-Siegel	P-Splines
Abs(e)	0.3150	0.2740
e^2	0.2582	0.1463

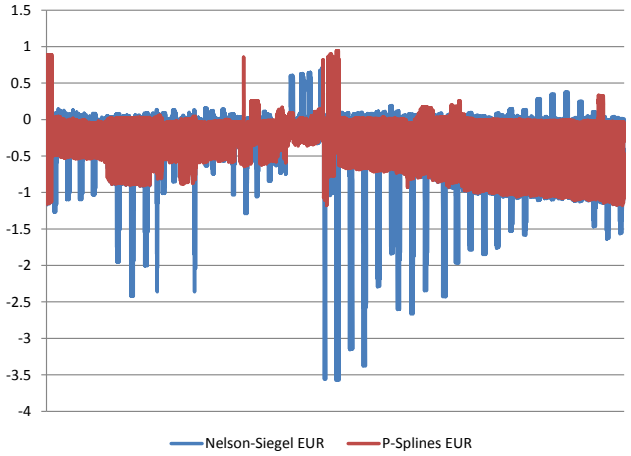

Fig. 7: Error de valorización diaria de 25 bonos del gobierno alemán en EUR desde el 30/6/2014 hasta el 18/03/2015, se compara la metodología de P-Splines con la de Nelson-Siegel.

AUD	Nelson-Siegel	P-Splines
Abs(e)	0.2682	0.2379
e^2	0.2888	0.1695

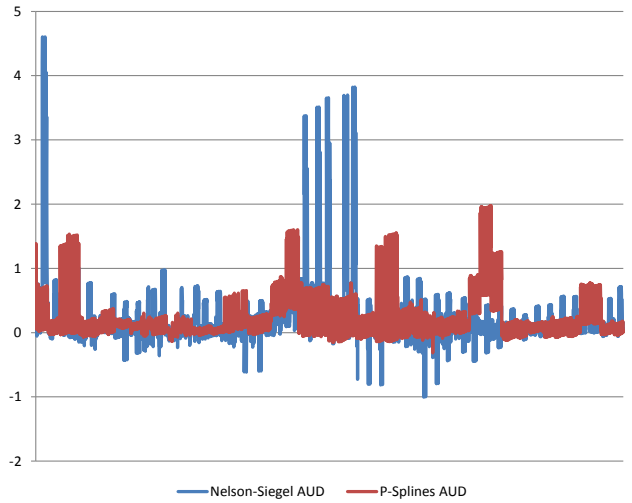

Fig. 8: Error de valorización diaria de 13 bonos del gobierno australiano en AUD desde el 30/6/2014 hasta el 18/03/2015, se compara la metodología de P-Splines con la de Nelson-Siegel.

En todos los casos, la metodología de regresión ponderada con P-Splines obtiene resultados muy superiores en error cuadrático medio. En el caso del error absoluto medio la metodología Nelson-Siegel supera a la de P-Splines únicamente para la curva de bonos del tesoro norteamericano, aunque ambas metodologías obtienen un error absoluto medio ínfimo. En el caso de los bonos del tesoro norteamericano, la metodología Nelson-Siegel genera errores bastante elevados para fechas en las que hubo deformaciones pronunciadas en la curva, lo cual eleva mucho su error cuadrático medio. Por otra parte, la razón por la que el error absoluto medio de Nelson-Siegel es menor en este caso es porque en esa metodología no se emplean ponderaciones por liquidez al momento de realizar la estimación de la curva, lo cual nos indica que, tal vez, para mercados tan eficientes como el de bonos del tesoro norteamericano, podríamos no emplear ponderaciones por liquidez, o usar alguna función que nos provea ponderaciones más homogéneas. Por último, es importante recalcar que los grupos de instrumentos con los que se construyeron las curvas no coinciden con los grupos de instrumentos que se valorizan con estas curvas al momento de comparar ambas metodologías. Esto evita cualquier tipo de sesgo de selección en la muestra.

En el caso de la valorización del derivado financiero, se comparó el método Monte Carlo con el método Cuasi Monte Carlo aquí expuesto. Los derivados valorizados son una opción call y una opción sobre una canasta de monedas. El primer caso es bastante sencillo e inclusive cuenta con solución matemática cerrada. El segundo caso es bastante más complicado y presenta un problema multidimensional que amerita el uso de una integración numérica. En ambos casos queda demostrada la superioridad de la metodología Cuasi Monte Carlo para problemas de dimensionalidad relativamente baja (menor a 20). Podemos notar la mayor precisión de la metodología en el gráfico ilustrativo: el Cuasi Monte Carlo converge mucho más rápidamente hacia el valor asintótico final del derivado y, además, es menos ruidoso en su convergencia hacia este valor. Esto significa que con una menor cantidad de simulaciones podemos obtener un resultado más aproximado y más confiable que con la metodología Monte Carlo.

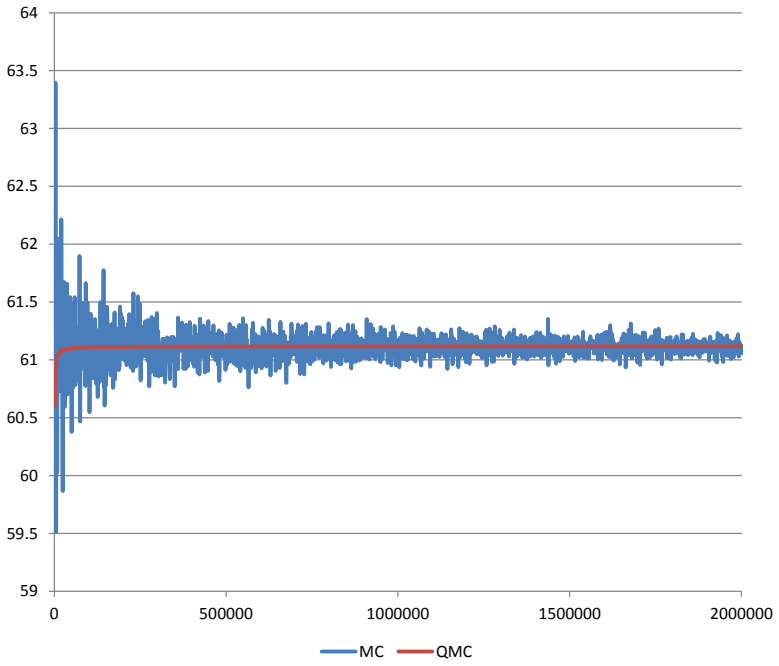

Fig. 9: Valorización de una opción call (eje vertical) empleando distinta cantidad de simulaciones (eje horizontal). Se emplearon hasta dos millones de simulaciones con saltos de mil en mil.

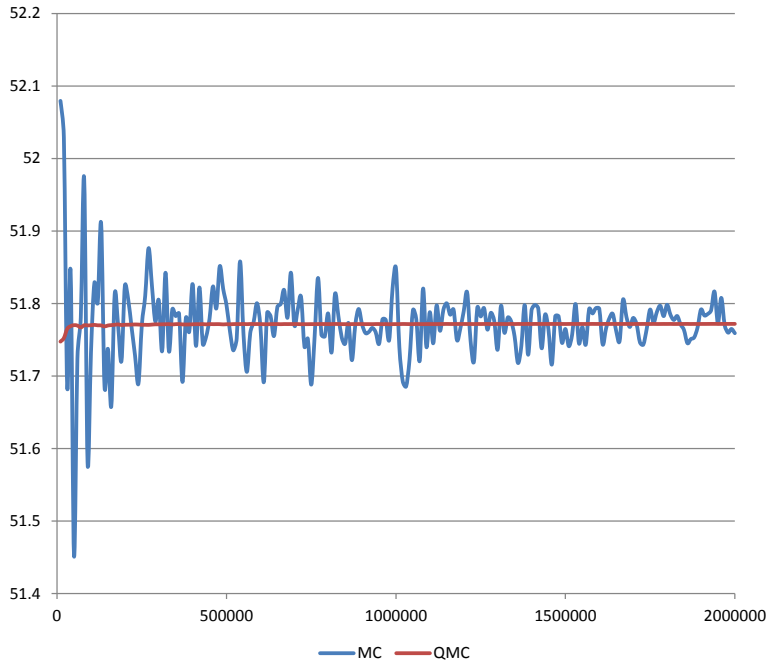

Fig. 10: Valorización de una opción sobre una canasta de monedas (eje vertical) empleando distinta cantidad de simulaciones (eje horizontal). Se emplearon hasta dos millones de simulaciones con saltos de diez mil en diez mil.

5. Conclusiones y recomendaciones

El objetivo del trabajo es presentar una metodología para la obtención de un valor razonable para bonos estructurados que no coticen continuamente. El enfoque que se adopta es el de presentar las propiedades teóricas indispensables para entender los métodos empleados, explicar adecuadamente qué se hace en cada uno y cómo realizar una adecuada implementación. Luego, lo inmediato es construir librerías en cualquier lenguaje de programación. Se recomienda Matlab, sobre todo para las librerías de P-Splines, donde hay un uso intensivo del álgebra matricial.

En mercados ilíquidos de bonos es mejor usar métodos de regresión por cuantiles Dutta, Basu & Vaidyanathan (2005); sin embargo, no parece posible en el estado actual del conocimiento, ya que, por la naturaleza recursiva del algoritmo de regresión por cuantiles, el cálculo de la dimensión efectiva del vector de parámetros se vuelve intratable. Es por ello que se usa una regresión de mínimos cuadrados,

donde hay una aproximación razonable de la dimensión efectiva del vector de parámetros en Eilers & Marx (1996).

Otro método que se puede aplicar para el cálculo de la matriz de covarianzas es el GARCH; sin embargo, este método tiene el problema de que, para la predicción conjunta de las covarianzas de varios activos, la cantidad de parámetros crece desproporcionadamente si no se realizan suposiciones muy restrictivas sobre las correlaciones condicionales entre los activos y, además, aumenta en gran medida la carga computacional, lo cual hace ineficiente su aplicación a bonos estructurados que tengan como activo subyacente una canasta de instrumentos.

En lugar del generador de números cuasi-aleatorios se puede emplear el muestreo mediante hipercubo latino, hipercubo latino ortogonalizado o el hipercubo latino casi ortogonalizado; sin embargo, estas metodologías no cuentan con la misma cantidad de material académico que los generadores de números cuasi-aleatorios lo que, aunado a que las ventajas que pueda tener frente a estos últimos no son claras, favorece el uso de secuencias generalizadas de Halton.

Por último, dado que una documentación exhaustiva de resultados comparativos entre metodologías escapa del alcance de este documento, se remite a documentos donde se llevan a cabo comparaciones de cada metodología por separado en las notas bibliográficas.

6. Bibliografía

Abramowitz, M. & Stegun, I., eds. - Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. New York: *Dover Publications* **(1964)**.

Acklam, P. J. – An algorithm for computing the inverse normal cumulative distribution function. http://home.online.no/~pjacklam/notes/invnorm/ **(2004)**. Fecha de acceso: 10 de Octubre del 2013.

Ait-Sahalia, Y. & Duarte, J. – Nonparametric option pricing under shape restrictions. *Journal of Econometrics* **116.1-2 (2003)**, 9-47.

Atanassov, E.I. – On the discrepancy of the Halton sequences. *Mathematica Balkanica, New Series* **18.1-2** (2004), 15-32.

Bollaerts, K., Eilers, P. & van Mechelen, I. – Simple and multiple P-Splines regression with shape constraints. *The British Journal of Mathematical and Statistical Psychology* **59.2 (2006)**, 451-469.

Choudhry, M. – The Market Yield Curve and Fitting the Term Structure of Interest Rates. *The Handbook of Fixed Income Securities* **C-41 (2005)**, 939-965. McGraw-Hill.

Dahl, L. O. & Benth, F. E. – Valuation of Asian Basket Options with Quasi-Monte Carlo Techniques and Singular Value Decomposition. *Centre for Mathematical Physics and Stochastics internal publication* **7 (2001)**.

De Boor, C. – On Calculating with B-Splines. *Journal of Approximation Theory* **6.1 (1972)**, 50-62.

De Rainville, F.M., Gagné, C., Teytaud, O. & Laurendeau, D. – Evolutionary optimization of low-discrepancy sequences. *ACM Transactions on Modeling and Computer Simulation* **22.2 (2012), 9:**1-26.

Dutta, G., Basu, S. & Vaidyanathan, K. – Term Structure Estimation in Illiquid Government Bond Markets: An Empirical Analysis for India. *Journal of Emerging Market Finance* **4.1 (2005)**, 63-80.

Dwyer, G.P. & Williams, K.B. – Portable Random Number Generators. *Journal of Economic Dynamics and Control* **27.4 (2003)**, 645-650.

Eilers, P. & Marx, B. – Flexible Smoothing with B-Splines and Penalties. *Statistical Science* **11.2 (1996),** 89-121.

Faure, H. – Discrepancy and diaphony of digital (0,1)-sequences in prime base. *Acta Arithmetica* **117.2 (2005)**, 125-148.

Faure, H. & Lemieux, C. – Generalized Halton Sequences in 2008: A Comparative Study. *ACM Transactions on Modeling and Computer Simulation* **19.4 (2009), 15:**1-30.

Hastie, T.J. & Tibshirani, R.J. – General Additive Models. London: *Chapman and Hall* **(1990)**.

He, X. & Ng, P. – COBS: Qualitatively Constrained Smoothing via Linear Programming. *Computational Statistics* **14.3 (1999)**, 315-338.

Joslin, S., Singleton, K. & Zhu, H. – A New Perspective on Gaussian Dynamic Term Structure Models. *Review of Financial Studies* **24.3 (2011)**, 926-970.

Jullion, A. & Lambert, P. – Robust specification of the roughness penalty prior distribution in spatially adaptive Bayesian P-Splines models. *Computational Statistics and Data Analysis* **51.5 (2007)**, 2542-2558.

Koenker, R. & Ng, P. – Inequality Constrained Quantile Regression. *The Indian Journal of Statistics* **67.2 (2005)**, 418-440.

Lee, T. – Smoothing Parameter Selection for Smoothing Splines: A Simulation Study. *Computational Statistics and Data Analysis* **42 (2003)**, 139-148.

Lemieux, C. & L'Ecuyer, P. – On the use of Quasi-Monte Carlo methods in computational finance. *Lecture Notes in Computational Science* **2073 (2001)**, 607-616.

Lyche, T. & Schumaker, L. – Local Spline Approximation Methods. *Journal of Approximation Theory* **15 (1975)**, 294-325.

Lyche, T. & Mørken, K. - Spline Methods Draft. *University of Oslo – Department of Informatics internal publication* **(2008)**.

Micula, G. & Micula, S. – The Handbook of Splines. Netherlands: *Kluwer Academic Publishers* **(1999)**.

Nelson, C.R. & Siegel, A.F. – Parsimonious modeling of yield curves. *Journal of Business* **60.3 (1987)**, 473-489.

Ökten, G., Shah, M. & Goncharov, Y. – Random and Deterministic Digit Permutations of the Halton Sequence. *9th International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing* **(2012)**, 589-602.

Pagès, G. – Van der Corput sequences, Kakutani transforms and one-dimensional numerical integration. *Journal of Computational and Applied Mathematics* **44.1 (1992)**, 21-39.

Pellizzari, P. – Efficient Monte Carlo Pricing of Portfolio Options. *Rendiconti per gli Studi Economici Quantitativi* **(2001)**, 108–123.

Poletti Laurini, M. & Moura, M. – Constrained Smoothing B-Splines for the Term Structure of Interest Rates. Insurance*: Mathematics and Economics* **46.2 (2010)**, 339-350.

Price, C.J. & Price, C.P. – Recycling primes in Halton sequences: an optimization perspective. *Advance Modeling and Optimization* **14.1 (2012)**, 17-29.

Shang, J. & Cavanaugh, J. – Bootstrap variants of the Akaike information criterion for mixed model selection. *Computational Statistics and Data Analysis* **52.4 (2008)**, 2004-2021.

Shen, J. & Wang, X. – Estimation of Monotone Functions via P-Splines: A Constrained Dynamical Optimization Approach. *SIAM Journal on Control and Optimization* **49.2 (2011)**, 646-671.

Subramanian, K. V. – Term Structure Estimation in Illiquid Markets. *The Journal of Fixed Income* **11.1 (2001)**, 77-86.

Superintendencia de Banca, Seguros y AFP – Instrumentos de Deuda: Manual Metodológico y de Procedimientos. "http://www.sbs.gob.pe/repositorioaps/0/1/jer/manual_metod/MANUAL_VECTOR_DE_PRECIOS_OCTUBRE_2010.pdf" **(2010)**. Fecha de acceso: 10 de Octubre del 2013.

Svensson, L.E.O. – Estimating and interpreting forward interest rates: Sweden 1992-1994. *NBER Working Papers* **4871 (1994)**, National Bureau of Economic Research.

Tuffin, B. (1996-A) – Improvement of Halton sequences distribution. *IRISA, Publication interne* **n°998 (1996)**.

Tuffin, B. (1996-B) – On the use of low discrepancy sequences in Monte Carlo methods. *Monte Carlo Methods and Applications* **2.4 (1996)**, 295-320.

Tuffin, B. – A new permutation choice in Halton sequences. *Lecture Notes in Statistics* **127 (1998)**, 427-435. New York: *Springer*.

Tuffin, B. – Randomization of Quasi-Monte Carlo methods for error estimation: survey and normal approximation. *Monte Carlo Methods and Applications* **10.3-4 (2004)**, 617-628.

Wang, X. & Hickernell, F.J. – Randomized Halton Sequences. *Mathematical and Computer Modelling* **32.7-8 (2000)**, 887-899.

6. Anexo 1: Gradientes

$$\frac{dH}{d\lambda} = -X(X'WX + \lambda {D_k}'D_k)^{-1}({D_k}'D_k)(X'WX + \lambda {D_k}'D_k)^{-1}X'W$$

$$\frac{dGCV}{d\lambda} = -\frac{2(y - Hy)'\frac{dH}{d\lambda}y}{\left(N - Tr(H)\right)^2} + \frac{2(y - Hy)'(y - Hy)Tr(\frac{dH}{d\lambda})}{\left(N - Tr(H)\right)^3}$$

$$\frac{dAIC}{d\lambda} = -\frac{2(y - Hy)'\frac{dH}{d\lambda}y}{\widehat{\sigma}_0^2} + 2Tr(\frac{dH}{d\lambda})$$

Anexo 2: Códigos para Matlab

Esta función genera el n-ésimo número cuasi-aleatorio de la serie correspondiente a la base b.

```
function h=GeneralizedHalton(n, b, f)
% Based on Faure,H. & Lemieux,C. (2008)
% b is the base of the dimension
% f is the corresponding multiplier for the dimension
% rds must be 1 for Random Digital Shifts, 0 otherwise.

n0 = n;
GH = 0;
l = 1 / b;
        while (n0 > 0);
            % Reduce b-th order by 1
            n1 = floor(n0 / b);
            % Get a_r(n)
            ar = n0 - n1 * b;
            % Scrambling
            q = f * ar;
            s = q - floor(q / b) * b;
            % Accumulate Sum(a_r(n)/b^r)
            GH = GH + l * s;
            l = l / b;
            n0 = n1;
        end
h=GH;
end
```

Esta función genera una matriz de números cuasi-aleatorios. Incorpora los primeros cincuenta números primos y sus respectivos multiplicadores de Faure-Lemieux. Esto

significa que la dimensionalidad máxima en la que esta función es útil es cincuenta. Si se desea incorporar más dimensiones, se puede revisar Faure & Lemieux (2009), donde se presentan los primeros trescientos multiplicadores.

```
function M=GHGen(N,Dimensions)
% rds must be 1 for Random Digital Shifts, 0 otherwise.
% b is the base of the dimension
% f is the corresponding multiplier for the dimension

M=zeros(N,Dimensions);
b=[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59,
61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131,
137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199,
211, 223, 227, 229]';
f=[1, 1, 3, 3, 4, 9, 7, 5, 9, 18, 18, 8, 13, 31, 9, 19, 36, 33, 21,
44, 43, 61, 60, 56, 26, 71, 32, 77, 26, 95, 92, 47, 29, 61, 57, 69,
115, 63, 92, 31, 104, 126, 50, 80, 55, 152, 114, 80, 83, 97]';

    for n=1:N;
        for d=1:Dimensions;
        M(n,d)=GeneralizedHalton(n, b(d), f(d));
        end
    end
end
```

Esta función calcula el valor de un B-Spline en un punto t dado del dominio.

```
function bas=basicspline(t, Knots, Degree, NCurve)
x=0;
y=0;
    if (Degree == 0);
     if (t > Knots(NCurve,1) & t <= Knots(NCurve+1,1));
     bas = 1;
     else
     bas = 0;
     end
    else
     x = (t - Knots(NCurve,1)) / (Knots((NCurve+Degree),1)-
Knots(NCurve,1));
     y = (Knots((NCurve+Degree+1),1)-t) /
(Knots((NCurve+Degree+1),1)-Knots((NCurve+1),1));
     bas = x*basicspline(t, Knots, Degree-1, NCurve)+y*basicspline(t,
Knots, Degree-1, NCurve+1);
    end
 end
```

Aquí generamos toda la matriz de B-Splines para los puntos especificados en el vector grid.

```
function BSM=BSplinesMat(Grid, Degree, NumSplines, LowerBound,
UpperBound)
% Uniformly distributed knots B-Splines
% Type must be 1 for Multiple Knots at Lower Bound and Upper Bound.
```

```
% Type default is Phantom Knots

%% Generating the Knots
InteriorKnots=NumSplines-Degree+1;
NKnots=NumSplines+Degree+1;
Step=(UpperBound-LowerBound)/(InteriorKnots-1);
Knots=zeros(NKnots,1);

    % Generating Phantom Knots by default
    for i=1:(NKnots);
    Knots(i,1)=LowerBound+Step*(i-1-Degree);
    end

%% Generating the B-Splines Matrix for regression
lg=size(Grid,1);
BSM=zeros(lg,NumSplines);

    for c=1:NumSplines;
     for r=1:lg;
     BSM(r,c)=basicspline(Grid(r,1),Knots,Degree,c);
     end
    end
end
```

Con la siguiente función generamos las curvas cupón cero.

```
function [Beta
Lambda]=PSplinesRegZCC(t,CashFlowMatrix,BidPrice,AskPrice,SpreadMax,D
egree,NumSplines, LowerBound, UpperBound)
% t as a column vector that contains the time of every cashflow in
CashFlowMatrix
% CashFlowMatrix as a matrix that contains every cashflow in a (time
x securities) fashion
% phantom nodes by default

%% Inputs
    %BS=BSplinesMat(X(:,1),4,8,0,7);
    BS=BSplinesMat(t, Degree, NumSplines, LowerBound, UpperBound);

    % Update to tanh weights
        %W1=(1./((abs(BidYield-
AskYield)+1).^2)).*(1./(1+Coupon).^0.5).*(Outstanding.^0.5);
    Spread=abs(AskPrice-BidPrice);
    W1=tanh(1-Spread/SpreadMax);
    W2=diag(W1);

%% P-Splines
    % Falta analizar el caso con restricciones (de igualdad y
desigualdad) de forma adecuada
    Q=BSplinesMat(0,Degree,NumSplines,LowerBound,UpperBound)';
    E=CashFlowMatrix'*BS;
    k=size(E,2);
    m=size(E,1);
        % First Order Difference Matrix
```

```
        D=zeros(k-1,k);
        for f=1:k-1;
        D(f,f)=-1;
        D(f,f+1)=1;
        end
    % fminunc options
        options = optimset('GradObj','on');
    % General Cross Validation's lambda
    lambdaGCV=fminunc(@(lambda)(PSGCV(BidPrice, E, D, W2,
lambda)),0,options);
        % Estimate for variance using GCV's lambda
        H=E*inv(E'*W2*E+lambdaGCV*(D'*D))*E'*W2;
        sigma2=(BidPrice-H*BidPrice)'*(BidPrice-H*BidPrice)/(m-1);
    % Akaike Information Criterion's lambda
    lambdaAIC=fminunc(@(lambda)(PSAkaike(BidPrice, E, D, W2,
lambda,sigma2)),lambdaGCV,options);

    % Equality Constrained Regression's Beta
    BetaPS=inv(E'*W2*E+lambdaAIC*(D'*D))*E'*W2*BidPrice;
    BetaPSEC=BetaPS-inv(E'*W2*E)*Q*inv(Q'*inv(E'*W2*E)*Q)*(Q'*BetaPS-
1)

    %% Plot
    % plot(X(:,1),X(:,2),X(:,1),BS*BetaPS)

%% Output
    % Upgrade for Inequality Constrained Regression -> dim(H)?
Beta=BetaPSEC;
Lambda=lambdaAIC;
end
```

Esta función calcula la validación cruzada generalizada.

```
function [GCV g]=PSGCV(y,X,D,W,lambda)

% Vector length of y
m=size(y,1);

% Influence matrix
H=X*inv(X'*W*X+lambda*(D'*D))*X'*W;
% d(H)/d(lambda)
dH=-
X*inv(X'*W*X+lambda*(D'*D))*(D'*D)*inv(X'*W*X+lambda*(D'*D))*X'*W;

% Generalized Cross Validation
GCV=(y-H*y)'*(y-H*y)/(m-trace(H))^2;
% GCV's gradient wrt lambda
g=2*(y-H*y)'*(y-H*y)*trace(dH)/(m-trace(H))^3-2*(y-H*y)'*dH*y/(m-
trace(H))^2;
end
```

Esta función calcula el criterio de información de Akaike.

```
function [AIC g]=PSAkaike(y,X,D,W,lambda,sigma2)
```

```
% Vector length of y
m=size(y,1);

% Influence matrix
H=X*inv(X'*W*X+lambda*(D'*D))*X'*W;
% d(H)/d(lambda)
dH=-
X*inv(X'*W*X+lambda*(D'*D))*(D'*D)*inv(X'*W*X+lambda*(D'*D))*X'*W;

% Akaike Information Criterion
AIC=(y-H*y)'*(y-H*y)/sigma2+2*trace(H)-m*log(sigma2)-m*log(2*pi);
% AIC's gradient wrt lambda
g=-2*(y-H*y)'*(dH*y)/sigma2+2*trace(dH);
end
```

Con esta función calculamos la curva para cualquier punto dentro del dominio objetivo, una vez que ya obtuvimos los betas de la regresión.

```
function PS=PSplinesGen(t, beta,Degree,NumSplines, LowerBound,
UpperBound)
% t as double
% beta as column vector
PS=BSplinesMat(t,Degree,NumSplines, LowerBound, UpperBound)*beta
end
```