Universidad Nacional Autonóma de México

Facultad de Ciencias

Proyecto Final: Simulación de Tasas de Interés: Modelo de Vasicek y CIR

Simulación Estocástica
Semestre 2025-1

Integrantes del Equipo:
Chirino Hernández Guadalupe Abigail
León Luna Luis Diego

27 de noviembre de 2024

Índice

Simulación de Tasas de Interés: Modelo de Vasicek y CIR

Chirino Hernández Guadalupe Abigail
León Luna Luis Diego

Introducción

El modelo de Vasicek sirve para modelar curvas de tasas de interés a corto plazo, su importancia se basa en que tienen los tipos de interés en los mercados financieros debido a que el comportamiento de los tipos de interés tiene repercusión no solo en la visión inversora de los particulares y entidades privadas que acuden a los mercados financieros, sino también influyen en las políticas monetarias que desarrollan las autoridades que gobiernan en cada país para obtener financiación.

Matemáticamente hablando, el modelo de Vasicek es una aplicacion en finanzas de un modelo que creado medio siglo antes llamado: **Modelo de Ornstein - Uhlenbeck**, el cual tiene numerosas aplicaciones en muy distintos campos *no solo en finanzas*.

A su vez, el modelo de CIR es una modificación al modelo de Vasicek, en el que lo único que cambia es que agrega un término: al componente estocástico que garantiza que el proceso estocástico de las tasas nunca sea negativo. Las propiedades que surgen de esa modificación hacen que sea un gran modelo para modelar curvas de interés más generales o riesgosas

Simulación del modelo Ornstein - Uhlenbeck

Un proceso de Ornstein - Uhlenbeck esta definido por la siguiente Ecuacion Diferencial Estocastica (EDE)

$$dX_t = \alpha(\mu - X_t)dt + \sigma dW_t$$

donde W_t es un movimiento browniano estandar, tambien llamado proceso de Wiener (de ahi la notacion W_t)
A un proceso estocastico como este se le llama "*de reversion a la media*" y es, de hecho, **una cadena de Markov continua** homogenea en el tiempo; esto significa que su probabilidad de transicion depende unicamente del tiempo transcurrido y no del punto en el tiempo del que parta, es decir:

$$\mathbb{P}[\, X_{t+s} \,|\, X_t \,] = \mathbb{P}[\, X_s \,|\, X_0 \,]$$

- Su parametro μ se le conoce como parametro de reversion a la media. Esto significa que *(a largo plazo se diria en el contexto de finanzas)* conforme t crezca o mas precisamente conforme $t \to$ ínf , el proceso tendera a estabilizarse al rededor de μ
- Su parametro α se le conoce como velocidad de reversion a la media. Entre mas grande sea, mas rapido sera la convergencia del proceso entorno a μ
- Su parametro σ es un parametro relacionado con dispersion del proceso. Observacion: No es igual a la desviacion estandar del mismo, aunque si esta asociado a esta.

Es posible llegar a una expresion de la solucion exacta de un proceso de Ornstein - Uhlenbeck y conocer la distribucion exacta del proceso solucion X_t en cada punto del tiempo $t \in \mathbb{R}^+$, y esta es:

$$X_t \sim N(\mu_t, \sigma_t^2)$$

con $\mu_t = \mu + (X_0 - \mu)e^{-\alpha t}$ y $\sigma_t = \sigma \frac{1-e^{-2\alpha t}}{2\alpha}$. Entonces, podemos expresarlo asi:

$$X_t = \mu_t + \sigma_t Z_t$$

Modelo de Vasicek

Es importante hacer una aclaracion historica antes de seguir avanzando. El modelo de Ornstein - Uhlenbeck fue desarrollado en 1930 por Leonard Ornstein y George Uhlenbeck. Con el tiempo se ha descubierto que se puede aplicar en una gran variedad de campos por lo general que es el comportamiento que describe la Ecuacion Diferencial Estocastica que lo define. Esta variedad de aplicaciones, tambien proviene de que dentro de las muy distintas formas en las que se puede plantear una Ecuacion Diferencial Estocastica, esta es relativamente sencilla de resolver y de encontrar una expresion cerrada para su solucion exacta.

Una de estas aplicaciones la realizo Oldrich Vasicek en 1977 en el campo de las finanzas. Concretamente, en la modelacion de la dinamica de tasas y curvas de interes a corto plazo. En finanzas, a este modelo que matematicamente es el mismo que el de Ornstein - Uhlenbeck, se le conoce como **Modelo de Vasicek** y en este:

- el parametro μ se escribe como $r^{(equilibrio)}$ o r_e, y representa la tasa *(de interes)* a la cual el proceso tendera a estabilizarse a largo plazo en el modelo
- los otros dos parametros si mantienen la notacion

Simulación Monte Carlo de Trayectorias

Computacionalmente es imposible simular una trayectoria a tiempo t, para todo t en $\mathbb{R}^+$ *(pues son infinitos y solo podemos hacer operaciones finitas con una computadora)*. Toda simulacion de trayectorias de algun proceso estocastico involucra una discretizacion en el tiempo, esto lo controlaremos de la siguiente forma:

- Especificamos un punto en el tiempo t_f hasta el cual hayamos decidido simular la trayectoria.
- Determinamos que los pasos entre t_0 y t_f seran equidistantes y definiremos el tamaño de cada paso como $dt = t_{i+1} - t_i$ *(*que es igual para todo i)
- Asi, al introducir como argumento dt, tenemos un vector con los puntos en el tiempo $(t_0, t_1, t_2, ..., t_n)$ en los que simularemos los puntos de las trayectorias. Obs que $t_0 = 0$ y $t_n = t_f$

La simulacion de las trayectorias hace uso de la **Simulacion Monte Carlo**, pero vale la pena hacer unas observaciones para que no hayan confusiones tras esta seccion.
En la Simulacion Monte Carlo usual, buscamos realizar n simulaciones de alguna distribucion de probabilidad, obteniendo una muestra de tamaño n de la variable aleatoria que nos interesaba analizar *(y que tenia dicha distribucion)*. Queremos puntualizar esto ultimo: la muestra que obtenemos es una muestra iid (independiente e identicamente distribuida) de esa distribucion, es decir cada elemento de la muestra $\in \mathbb{R}$.

En contraste, lo que esta obteniendo nuestro algoritmo es **una muestra de tamaño m independiente e identicamente distribuida de trayectorias** del proceso estocastico completo hasta tiempo t_f en los tiempos $t_0, t_1, t_2, ..., t_n$. La simulacion Monte Carlo *(en el sentido unidimensional usual)* esta corriendo sobre cada uno de los n pasos de las m trayectorias.

Discretización del Proceso

Habiendo esclarecido eso, no esta de mas escribir la expresion de la discretizacion de un proceso de Ornstein - Uhlenbeck que estamos usando para simular las trayectorias:

$$X_{t+\Delta t} \;=\; \mu_{t+\Delta t} + \sigma_{t+\Delta t}\, Z_{t+\Delta t}$$

donde son iid todas las Z: $Z \sim N(0,1)$,

$$\mu_{t+\Delta t} \;=\; \mu + (X_t - \mu)\, e^{-\theta\, \Delta t}$$

$$\sigma_{t+\Delta t} \;=\; \sigma \sqrt{\frac{1 - e^{-2\theta\, \Delta t}}{2\theta}}$$

donde recordemos que por la equidistancia de los tiempos: $\Delta t \;=\; t_{i+1} - t_i$, por lo que podemos reescribir esas expresiones hacia atras para llegar a la forma que esta en el codigo:

$$X_t \;=\; \mu_t + \sigma_t\, Z_t$$

con:

$$\mu_t \;=\; \mu + (X_{t-1} - \mu)\, e^{-\theta\, \Delta t}$$

$$\sigma_t \;=\; \sigma \sqrt{\frac{1 - e^{-2\theta\, \Delta t}}{2\theta}}$$

Justificación Teórica

Desde una perspectiva teorica, esta forma de discretizar iterativa, proviene de la propiedad ya mencionada $\mathbb{P}[\, X_{t+s} \,|\, X_t \,] = \mathbb{P}[\, X_s \,|\, X_0 \,]$ de este proceso estocastico en particular.

Esa propiedad permite que la discretizacion se pueda entender del siguiente modo: Supongamos que solo queremos simular 1 paso de las m trayectorias. Entonces, es claro que esta seria la discretizacion del proceso en 2 tiempos discretos: $(t_0, t_f) = (0, t_1)$, pues incluso esta discretizacion luciria funcionalmente como la expresion del proceso a tiempo continuo.

¿Que ocurriria para el segundo paso? Por la propiedad de homogeneidad en el tiempo de esta cadena de markov, podriamos simular el segundo paso, utilizando el paso que ya simulamos. Y es claro que ahora, el termino X_0 ya no seria la condicion inicial pues ahora estamos iniciando en X_1 de acuerdo a esa idea. Y observese que estamos simulando el paso 2.

Esta reflexion, podemos culminarla mediante inducción simple sobre n (iterativamente en el codigo), que es el numero de pasos y lo que estariamos diciendo en dicho paso es que para simular X_n hay que repetir el proceso pero tomando X_{n-1} como condicion inicial. Observese que lo demas en la expresion no afecta este argumento ya que todos los Δt son iguales *(por la equidistancia de los tiempos)*

Nuestra implementacion de las trayectorias funciona llenando una matriz de nxm, en la que cada columna es una trayectoria y cada renglon es uno de los n pasos *(o puntos)* que la conforman. El llenado es por renglones, es decir, en cada iteracion llenamos el i-esimo paso de todas las trayectorias para aprovechar la estructura vectorial de R

```r
# (mu, alpha, sigma) :  Son los parametros del modelo
#                  X0 :  Es simplemente la condicion inicial de la trayectoria en tiempo t0 = 0
#                  tf :  Controla hasta que punto en el tiempo se van a simular las trayectorias
#                   m :  Es el numero de trayectorias que simularemos simultaneamente

ornstein_uhlenbeck <- function( mu, alpha, sigma, X0, tf, dt, m) {

  tiempos <- seq(0, tf, by = dt)         # Discretizaci n del tiempo entre 0 y t_f
  n <- length(tiempos)                   # Queda definido n : el numero de pasos
  # Matriz para almacenar las trayectorias, n pasos x m trayectorias
  trayectorias <- matrix(0, nrow = n, ncol = m)
  trayectorias[1, ] <- X0                # A tiempo 0, todas inician en la condicion inicial X_0

  for (i in 2:n) {
    # Para no sobrecargar la notacion calculamos primero la media y la desviacion estandar del paso
    mu_t <- mu + (trayectorias[i-1, ] - mu) * exp(- alpha * dt)
    sigma_t <- sigma * sqrt((1 - exp(-2 * alpha * dt)) / (2 * alpha))
    # Con ellas, hacemos una simulacion montecarlo para el i-esimo paso de todas las trayectorias
    trayectorias[i, ] <- mu_t + sigma_t * rnorm(m, 0, 1)
  }
  return(list(tiempos = tiempos, trayectorias = trayectorias))
}
```

Listing 1: Función Ornstein-Uhlenbeck en R

Bootstrap

Cuando simulamos m trayectorias *(con m "grande")*, podemos realizar un re-muestreo bootstrap para estimar la distribucion empirica de las trayectorias en cada paso y asi crear intervalos de confianza tomando los percentiles al $(\frac{\alpha}{2}, (1 - \frac{\alpha}{2}))\, 100\,\%$ de confianza.

Esto puede parecer que no tiene caso cuando simulamos las trayectorias sin datos reales, es decir especificando a voluntad los parametros, horizonte temporal y condicion inicial. Esto ya que en este caso, conocemos la distribucion exacta de $X_t \quad \forall t \in \mathbb{R}^+$ y por tanto conocemos de manera exacta los cuantiles necesarios por lo que podriamos calcular los intervalos de confianza exactos en cada tiempo que los deseemos. Observese que tambien con bootstrap, podriamos obtener en el mismo algoritmo la media y mediana en cada punto del tiempo.

Sin embargo, cuando trabajemos con datos reales, todo esto a nivel poblacional, ya no es conocido. Asi que el bootstrap seria la mejor forma de obtener **intervalos de confianza en cada punto del tiempo en el que tengamos datos** sin hacer suposiciones fuertes sobre la distribucion del proceso.

Hay que resaltar, que el calculo mediante bootstrap de estos intervalos de confianza o incluso de algunos cuantiles de interes, la mediana o incluso la media en cada punto del tiempo *(donde haya datos)*, ya no resulta algo despreciable pues ahora nos da informacion del proceso y nos permite "validar" o "alertarnos" si el modelo no esta logrando explicar bien la dinamica interna del conunto de datos.

En particular, hablando de cuantiles: el bootstrap aqui aplicado nos puede ayudar a estimar nuestra perdida maxima esperada *a un cierto nivel de confianza* por medio del calculo del VaR_α**: Value at Risk** ya en el contexto de la valuacion de un bono o algun activo que dependa de cantidades conocidas y una tasa de interes a un cierto tiempo futuro, pues matematicamente el VaR_α es tan solo el cuantil al $(1-\alpha)100\,\%$ de confianza de la variable aleatoria de perdida, definida a partir de los posibles resultados de perdida-ganancia que pudieramos tener dada una posicion en un bono u activo financiero en general

Trayectorias con ggplot2

Ggplot2 nos exige que los datos a graficar vengan en un data frame en donde cada columna represente una y solo una variable a graficar en las diferentes aesthetics disponibles. La opcion mas natural es crear un df con una columna de tiempos (eje x), una columna con los valores de las trayectorias en cada tiempo (eje y) y una mas que identifique de que trayectoria es cada punto (colores).

Una forma de lograrlo es crear ese formato de datos largos con mutate y pivot-longer *del tidyr* o algo similar, pero se tarda demasiado por que esta haciendo varias operaciones innecesarias en el proceso.

Otro modo mucho mas rapido y sin operaciones innecesarias es usando lapply con una funcion que cree un data frame para cada trayectoria. El lapply me devuelve una lista de estos data frames. Al final, el do.call le aplica a todos los elementos de la lista *(con el argumento ".")* la funcion rbind para "pegar por renglones" , asi pega un data frame debajo del otro debajo del otro y eso ya termina siendo un data frame justo como lo necesitamos. Aqui el extracto de la parte del codigo en el que implementamos esto:

```r
datos_largos <- lapply(1:m, function(i) {
                                data.frame(
                                      Trayectoria = i,
                                      Tiempo = simulacion$tiempo,
                                      Tasa = simulacion$trayectorias[,
                                         i]   )
                  }) %>% do.call(rbind, . )
```

Listing 2: Transformación de datos a formato largo con lapply y do.call

Vamos a realizar 4 graficas sobre el mismo ejemplo con la misma semilla y mismo θ para comparar

- La primer grafica es la misma que la que generamos antes con R base
- La segunda grafica es en lugar de simular 10 dias, simula 18.5 *(2 semanas y media)* y lo hace simulando 2 pasos por dia, es decir, $dt = \frac{1}{2}$
- La tercera es como la segunda pero con $t_f = 30$ *(1 mes)* simulando 4 pasos por dia
- La cuarta es como la tercera grafica pero hasta $t_f = 45$ *(1 mes y medio)* , simulando **un paso cada hora y media**, es decir $dt = \frac{1}{16}$ y en lugar de simular 8 trayectorias, simulamos 16.

En todas las graficas hay 2 lineas de referencia: una en y $= r^{eq}$ y otra en y $= 0$

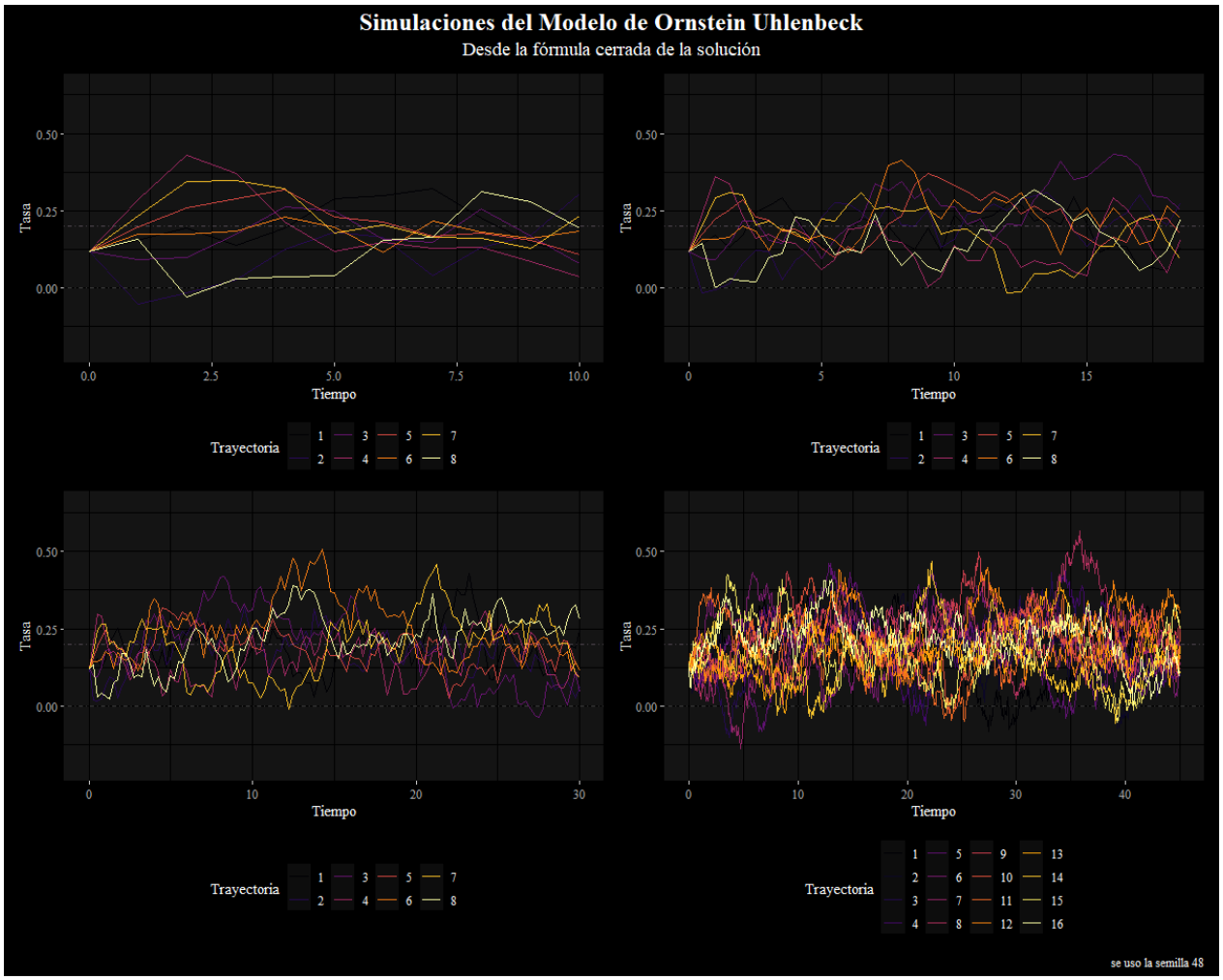

Figura 1: Simulaciones del Modelo de Ornstein Uhlenbeck desde la fórmula cerrada de la solución.

O alternativamente si estamos interesados en un analisis muy a corto plazo, podemos ver como se incrementa el nivel de detalle para un horizonte temporal mucho mas corto y esta vez fijo de 7 dias. De nuevo, con la misma semilla y mismo θ

- La primer grafica simula 8 trayectorias a 7 dias, con 2 pasos por dia, es decir cada medio dia avanza un paso
- La segunda grafica es en lugar de simular un paso cada medio dia, simular uno cada 6 horas, es decir $dt = \frac{1}{4}$
- La tercera es como la segunda pero simulando un paso **cada media hora**, es decir $dt = \frac{1}{48}$
- La cuarta es como la tercera grafica pero en lugar de 8 trayectorias, simulamos 16 y **dan un paso cada 15 minutos**, es decir $dt = \frac{1}{48}$

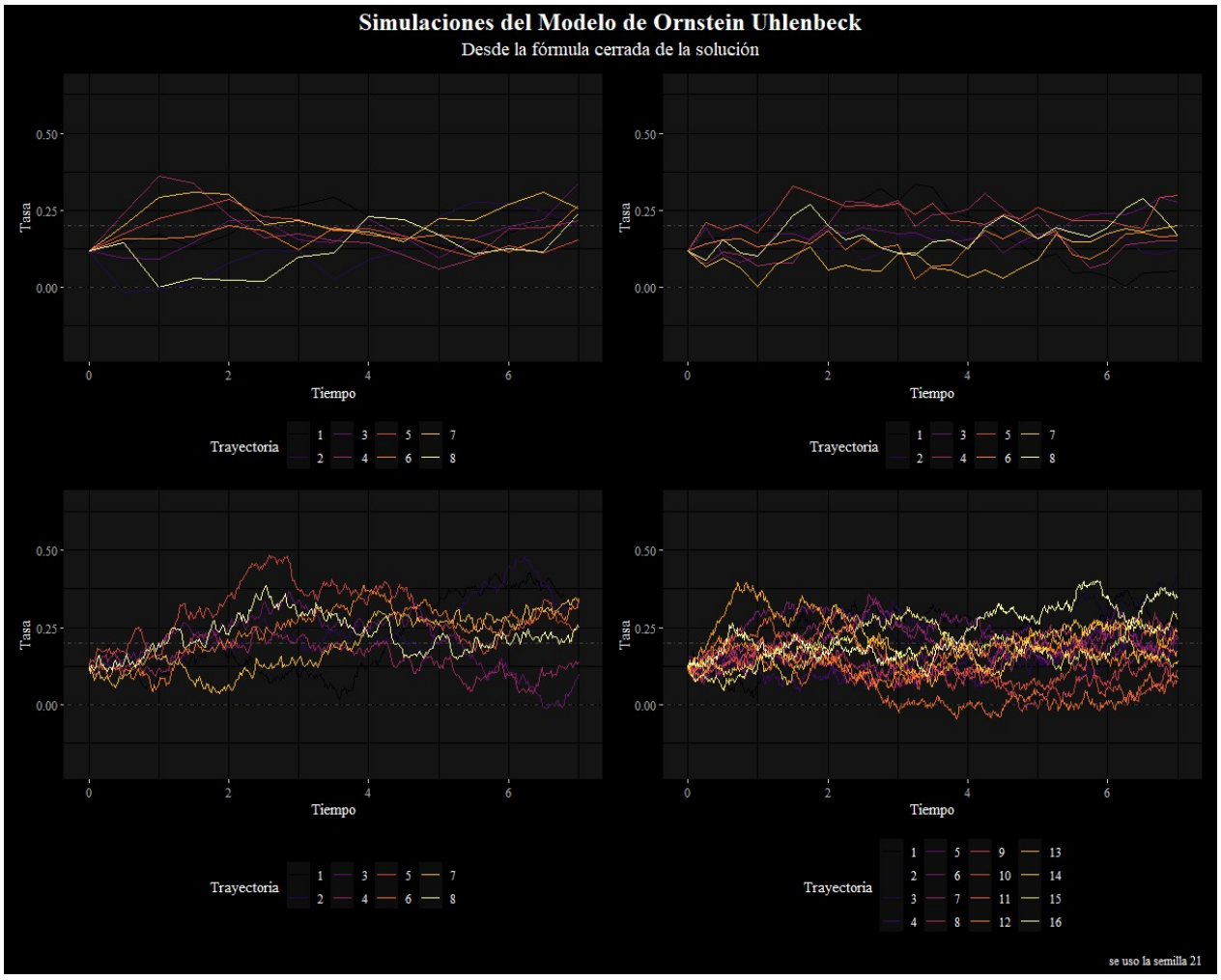

Figura 2: Simulaciones del Modelo de Ornstein Uhlenbeck desde la fórmula cerrada de la solución.

Formas Alternativas de Simulación

Aunque proseguiremos el desarrollo de este proyecto usando la implementacion ya mostrada para simular trayectorias, presentamos dos metodos alternativos de hacerlo que son menos precisos ya que no parten de la solucion exacta de la EDE que define al proceso, sino de la expresion como tal de la EDE sin resolver.

Estos son el metodo de Euler - Maruyama y el metodo de Milstein. En general, su importancia radica en que si la EDE que define al proceso estocastico que queremos simular, fuese demasiado complicada de resolver o bien ni si quiera fuese posible encontrar una expresion cerrada de su solucion exacta, estos metodos nos dan una manera de simular una aproximacion del proceso que es la solucion exacta y desconocida de la EDE, lo cual ocurre con frecuencia.

Lease el documento adjunto sobre los Metodos Euler - Maruyama y Milstein para simular

aproximaciones de la solucion de una EDE

En el caso especifico del modelo de Vasicek, el metodo de Milstein resulta el mismo que el de Euler - Maruyama ya que el termino que mejora la aproximacion involucra a $\frac{\partial g(X_t)}{\partial X_t}$, es decir, la derivada del termino de difusion del proceso con respecto al proceso mismo y como en el modelo de Vasicek, el termion de difussion es la constante σ, esa derivada es 0. Sin embargo en otros modelos como el de CIR (Cox-Ingersoll-Ross), este termino es muy relevante para mejorar la precision de las trayectorias. Y ya que no se puede llegar a una forma cerrada de su solucion exacta, este es el mejor camino para poder simular trayectorias y, de hecho, en la practica se usa el modelo a traves de realizar muchas simulaciones y hacer inferencia sobre lo que se busque analizar en ellas

A continuacion mostramos las trayectorias con la misma semilla, parametros, horizonte temporal y tamaño de paso de la primer grafica, solo que ahora simuladas desde el metodo de euler - maruyama con el fin de poder comparar ambos metodos

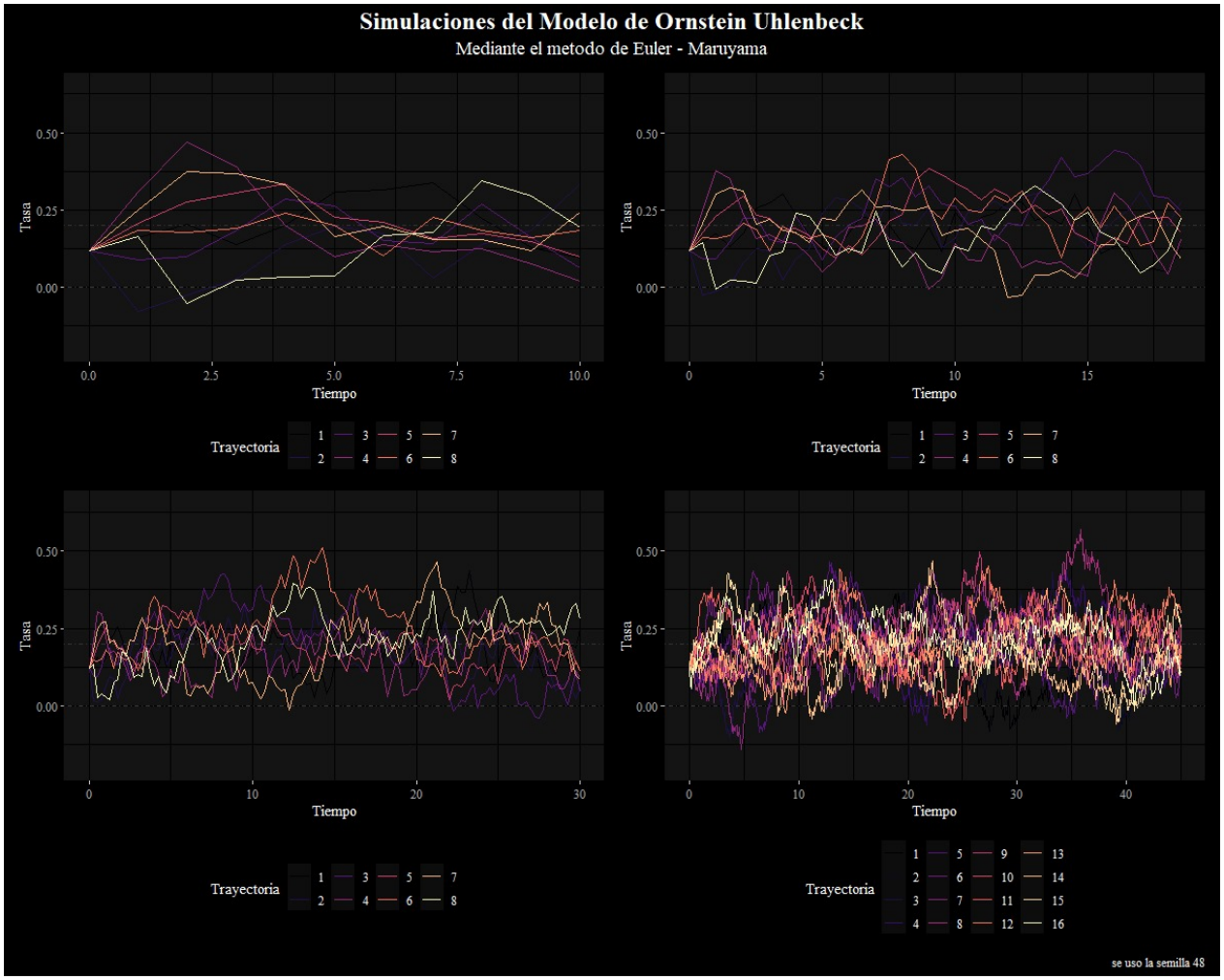

Figura 3: Simulaciones del Modelo de Ornstein Uhlenbeck desde la fórmula cerrada de la solución.

Estimación de Parámetros del Modelos

El objetivo es poder utilizar este modelo para simular trayectorias futuras para observaciones de datos reales y poder usar el modelo para valuar activos como bonos a corto plazo. Al trabajar con datos reales, surgen inmediatamente 2 retos:

1. Determinar como vamos a estimar los parametros del modelo
2. Asegurarnos de que podemos confiar en nuestra estimacion de parametros

El segundo punto es muy importante, ya que como los parametros verdaderos seran desconocidos, si la estimacion es mala y proseguimos el analisis asumiendola como buena, toda conclusion sera potencialmente errada.

Hacemos enfasis en ello, porque la expresion de la verosimilitud del modelo es altamente compleja y el camino usual de:

- Derivar la log-verosimilitud con respecto a todos los parametros
- Evaluar cada derivada a 0
- Resolver el sistema de ecuaciones resultante

No lleva a nada en este caso o requiere un analisis extenso de dicho sistema de ecuaciones resultante y una buena cantidad de tiempo para invertir. En su lugar, utilizaremos **simulacion estocastica** para maximizar la log-verosimilitud tratandola como simplemente una funcion objetivo a maximizar en el espacio parametral de nuestros 3 parametros: $(\mu, \alpha, \sigma) \in \mathbb{R}^3$. Mas concretamente, el espacio parametral matematicamente es:

$$\mathbb{R} \times \mathbb{R}^+ \times \mathbb{R}^+$$

aunque financieramente, tendria sentido que fuese $(\mathbb{R}^+)^3$. En cualquier caso, no es un espacio parametral sencillo y se vera que la expresion de la log verosimilitud tampoco lo es.
El metodo de Simulacion Estocastica que utilizaremos sera **Simulated Annealing**

Log Verosimilitud

En una primera instancia, la primer idea acerca de como plantear la log verosimilitud pudiese ser partir de la expresion $r_t \sim N(\mu_t, \sigma_t^2)$ y comenzar escribiendo la log verosimilitud para una muestra de la distribucion normal y luego sustituir las expresiones μ_t y $sigma_t^2$. Sin embargo esto no puede hacerse asi directamente ya que estariamos violando una hipotesis importante en ese planteamiento que es que **la muestra debe ser iid** y en este caso la muestra no es independiente, ya que, en terminos simples: el proceso $X_{t+\Delta t}$ (al tiempo t) tiene una dependencia obvia de X_t.

Para manejar correctamente esta dependencia, usamos la estructura de la muestra de datos reales entorno a la cual buscamos definir la log verosimilitud. Recordemos que cada entrada

i de los datos representa el proceso a tiempo t_i y que entre cualquier tiempo y el siguiente existe la misma distancia de paso: Δt. Por ello aprovechamos esa estructura de los datos y la propiedad de homogeneidad en el tiempo del proceso para definir la log verosimilitud partiendo de la idea de la misma expresion que usamos para la simulacion de las trayectorias, que era:

$$X_t \;=\; \mu_t + \sigma_t \, Z_t$$

A partir de ahora, usaremos la parametrizacion del modelo de Ornstein Uhlenbeck en el campo de las finanzas, que como ya habiamos mencionado toma el nombre de **Modelo de Vasicek**. Asi, ahora escribimos r_e en lugar de μ, quedando la expresion de μ_t ahora escrita asi:

$$\mu_t \;=\; \mu + (X_{t-1} - r_e)\, e^{-\alpha \, \Delta t}$$

La expresion de σ_t no cambia:

$$\sigma_t \;=\; \sigma \sqrt{\frac{1 - e^{-2\alpha \, \Delta t}}{2\alpha}}$$

Asi, podemos calcular correctamente la funcion de verosimilitud $L(\theta)$

$$L(\alpha, r_e, \sigma) \;=\; L(\theta) \;=\; f(x_0, x_1, ..., x_n \,; \theta) \;=\; f(x_0; \theta) \prod_{t=1}^{n} \frac{1}{\sqrt{2\pi \, \sigma_t^2}} exp \left(\, - \frac{(x_t - \mu_t)^2}{2\sigma_t^2} \, \right)$$

en donde μ_t y $sigma_t^2$ son las expresiones justo antes desarrolladas. Como el objetivo es maximizar, podemos de una vez despreciar la constante $f(x_0; \theta)$ .Observese que podria ahi sustituirse estas expresiones en la expresion completa de $L(\theta)$ pero realmente no ganamos nada, mas que terminar con una expresion larguisima y abrumadora de la verosimilitud, que despues habra que seguir manipulando y arrastrando
Ahora, aplicando logaritmo, la log verosimilitud $\mathcal{L}(\theta)$ es:

$$\mathcal{L}(\alpha, r_e, \sigma) \;=\; \mathcal{L}(\theta) \;=\; log(L(\alpha, r_e, \sigma)\,) = \sum_{t=1}^{n} log \left(\, \frac{1}{\sqrt{2\pi \, \sigma_t^2}} exp \left(\, - \frac{(x_t - \mu_t)^2}{2\sigma_t^2} \, \right) \right) =$$

$$\sum_{t=1}^{n} log \left(\, \frac{1}{\sqrt{2\pi}} \, \right) + log \left(\, \frac{1}{\sqrt{\sigma_t^2}} \, \right) + log \left(\, exp \left(\, - \frac{(x_t - \mu_t)^2}{2\sigma_t^2} \, \right) \right) =$$

$$\sum_{t=1}^{n} log \left(\, (2\pi)^{-\frac{1}{2}} \, \right) + log \left(\, (\sigma_t^2)^{-\frac{1}{2}} \, \right) - \left(\, \frac{(x_t - \mu_t)^2}{2\sigma_t^2} \, \right)$$

Por lo tanto, la expresion de la log verosimilitud con la que trabajaremos es:

$$\mathcal{L}(\alpha, r_e, \sigma) \;=\; \mathcal{L}(\theta) \;=\; -\frac{n}{2} \, log(2\pi) \sum_{t=1}^{n} -log(\sigma_t) - \left(\, \frac{x_t - \mu_t}{2\sigma_t} \, \right)^2$$

Y la implementacion es:

```r
# datos debe ser un vector
log_verosimilitud <- function(theta, datos, dt){

  alpha <- theta[1]
  r_e   <- theta[2]
  sigma <- theta[3]
  r <- datos

  # alpha y sigma deben ser positivos (no pueden ni siquiera ser 0 o
     logL(theta) se indetermina)
  if (alpha <= 0 || sigma <= 0)    return(-Inf)
  n <- length(datos) - 1      # porque en cada iteracion usaremos el
     indice i y i+1
  logL <- - n/2 * log(2*pi)     # inicializo en los terminos que no
     dependen del indice t

  for(t in 2:n){
    mu_t <- r_e + ( r[t-1] - r_e ) * exp(-alpha*dt)
    sigma_t <- sigma * sqrt( (1-exp(-2*alpha*dt)) / (2*alpha) )
    logL <- logL - log(sigma_t) - 1/2 * ( (r[t]-mu_t) / sigma_t )^2
                # voy acumulando
  }
  return(logL)
}
```

Listing 3: Función de log-verosimilitud en R

Simulated Annealing

Ya con una expresion de la log verosimilitud, no es muy dificil darse cuenta de que el sistema de ecuaciones de las derivadas igualadas a 0 va a terminar siendo algo no lineal en el que el estimador maximo verosimil de cada uno de los parametos quedara en terminos no lineales (y dificiles o imposibles de despejar) de los otros parametros. Con solo una excepcion: El EMV de: r_e que queda en terminos de α. Esto lo retomaremos mas adelante, por ahora planteamos el algoritmo de **Simulated Annealing** de forma directa para maximizar la log verosimilitud:

```r
simulated_annealing_vasicek <- function( theta_0, datos, dt, freno){

  # Inicializo una variable que va a ir conservando solo los mejores
     conjuntos de parametros
  # para theta. Recordemos que theta = ( alpha, r_e, sigma )
  mejor_theta <- theta_0
  mejor_logL <- log_verosimilitud(theta_0,datos,dt)
  # parametros para modificar la escala de las distribuciones de
     propuesta de cada parametro de theta
  c1 <- 0.1
  #c2 <- 0.01
  c2 <- 0.004
  c3 <- 0.005
  # Creo una matriz para guardar los parametros y la log
     verosimilitud
  recorrido <- matrix(0,freno,4)
  colnames(recorrido) <- c("alpha","r_e","sigma","logL")
  recorrido[1,] <- c(mejor_theta,mejor_logL)

  # Comienza el Simulated Annealing
  for(i in 2:freno){
    #Temp <- 100/log(2*i)
    Temp <- 10/i
    # Simulo de theta propuesta de las distribuciones de propuesta
    alpha <- rnorm(1,mejor_theta[1],c1)
    r_e   <- rnorm(1,mejor_theta[2],c2)
    sigma <- rnorm(1,mejor_theta[3],c3)
    theta_prop <- c(alpha,r_e,sigma)
    logL_prop <- log_verosimilitud(theta_prop,datos,dt)
    # Calculamos la probabilidad rho de aceptar el nuevo valor
    rho <- min(  exp( (logL_prop - mejor_logL)/Temp ), 1  )
    if( runif(1) < rho ) {
        mejor_theta <- theta_prop
        mejor_logL <- logL_prop
    }
    recorrido[i,] <- c(mejor_theta,mejor_logL)
  }
  return(recorrido)
}
```

Listing 4: Simulated Annealing para el modelo de Vasicek

Ahora, probemos nuestro Simulated Annealing para ir evaluando ¿que tan bueno es? o ¿que tan malo es?
Tomemos la siguiente trayectoria simulada con: $t_f = 61,\ dt = 1,\ r_0 = 0.12$, es decir supongamos que tenemos datos que asumimos como provenientes de una realizacion del proceso estocastico de vasicek (o de Ornstein - Uhlenbeck) de 2 meses con observaciones diarias, tales que el primer dato es $r_0 = 0.12$.

Los datos de este experimento fueron simulados con los parametros: $\theta = (\alpha, r_e, \sigma) = (0.30, 0.20, 0.08)$. Y se ven asi:

Figura 4: Simulaciones del Modelo de Ornstein Uhlenbeck desde la fórmula cerrada de la solución.

Tras realizar 4000 iteraciones del Simulated Annealing, los parametros que estimo el algoritmo son:

```
alpha    r_e    sigma    logL
0.3244   0.2051 0.0828   72.6407
```

Listing 5: Parámetros estimados

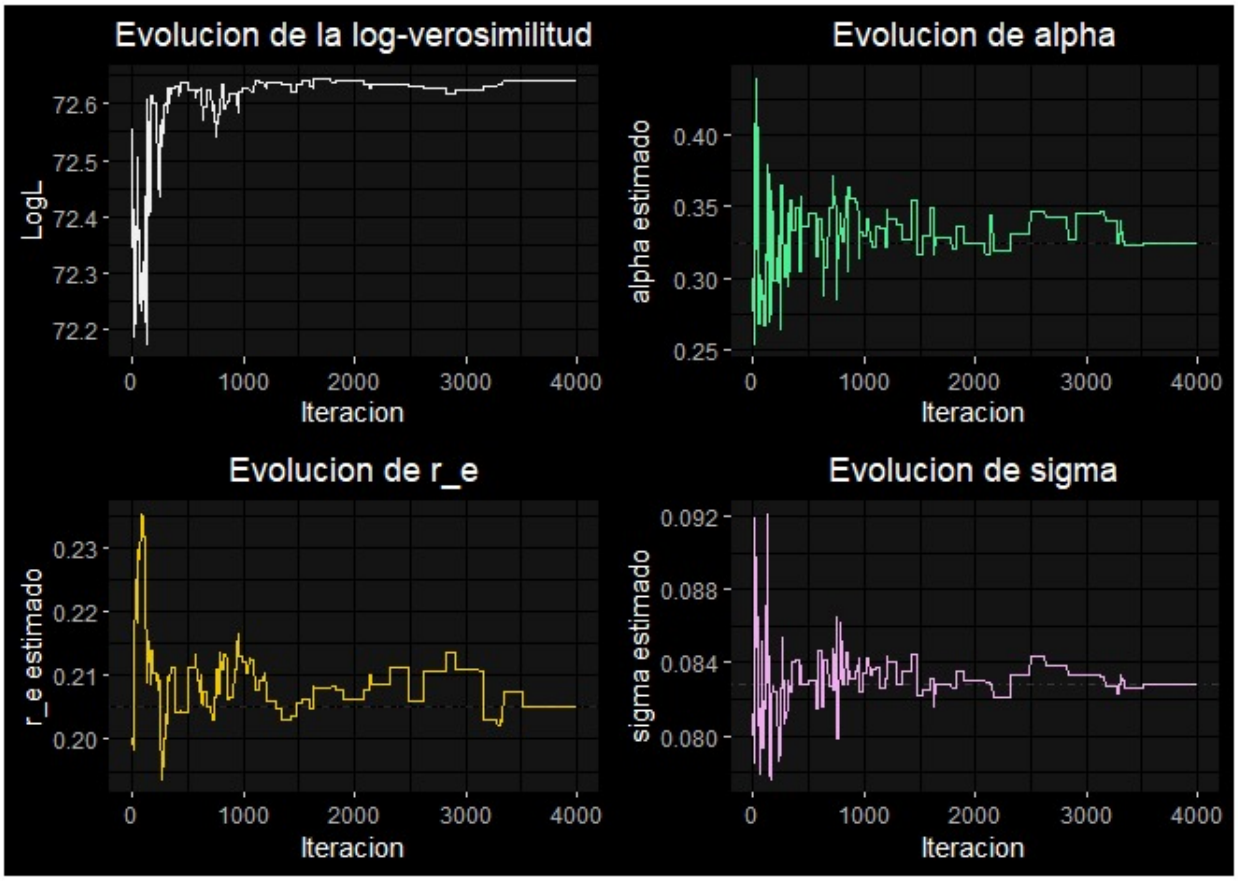

Figura 5: Simulaciones del Modelo de Ornstein Uhlenbeck desde la fórmula cerrada de la solución.

Complementariamente, en el html adjunto se prueba el algoritmo para muchas mas combinaciones de parametros, horizontes temporales y tamaños de paso

Estimador Máximo Verosímil para r_e

Si derivamos la log verosimilitud con respecto al parametro r_e e igualamos a cero, igualando a cero podemos llegar a una expresion del Estimador Maximo Verosimil de r_e solo en terminos de la muestra y del parametro α. Esto es conveniente porque α es, de hecho, el parametro mas sensible y dificil de estimar hasta el momento. Por ello, planteamos una modificacion al algoritmo de simulated annealing presentado, que aproveche esa relacion.

Una mejora al Simulated Annealing

A continuacion se implementan las 2 ideas anteriores y evaluamos como mejoro o no el algoritmo de Simulated Annealing probandolo de nuevo con muchas combinaciones de parametros en datos simulados

Uso del Modelo con Datos Reales

p r o x i m a m e n t e

Modelo de CIR

p r o x i m a m e n t e

Bibliografía

1. "¿Cómo puede considerarse el proceso de Ornstein-Uhlenbeck como el análogo continuo del proceso AR(1)?" Disponible en: https://math.stackexchange.com/questions/345773/how-the-ornstein-uhlenbeck-process-can-be-considered-as-the-continuous-time-

2. Wikipedia: "Proceso de Ornstein-Uhlenbeck". Disponible en: https://en.wikipedia.org/wiki/Ornstein%E2%80%93Uhlenbeck_process.

3. QuantStart: "Ornstein-Uhlenbeck Simulation with Python". Disponible en: https://www.quantstart.com/articles/ornstein-uhlenbeck-simulation-with-python/.

4. UExternado: "Artículo sobre Ornstein-Uhlenbeck". Disponible en: https://revistas.uexternado.edu.co/index.php/odeon/article/view/5337/6701.

5. UGent: "Publicación sobre procesos estocásticos". Disponible en: https://biblio.ugent.be/publication/8616757/file/8616758.

6. Physical Review E: "Analysis of the Discrete Ornstein-Uhlenbeck Process". Disponible en: https://journals.aps.org/pre/abstract/10.1103/PhysRevE.54.2084.

7. Redalyc: "Artículo sobre procesos estocásticos". Disponible en: https://www.redalyc.org/journal/3607/360748513003/html/.

8. Wikipedia: "Proceso de Wiener". Disponible en: https://es.wikipedia.org/wiki/Proceso_de_Wiener.

9. Wikipedia: "Cadenas de Markov". Disponible en: https://en.wikipedia.org/wiki/Markov_chain.

10. Cambridge University Press: "Discrete Ornstein-Uhlenbeck Process Analysis". Disponible en: https://www.cambridge.org/core/services/aop-cambridge-core/content/view/85ABC2817F241CCFB548B3AA134575AB/S002190020012011Xa.pdf/analysis_of_the_discrete_ornsteinuhlenbeck_process_caused_by_the_tick_size_effect.pdf.

11. Springer: "Estimation of Parameters for OU Processes". Disponible en: https://link.springer.com/article/10.1007/s10985-004-4775-9.