

Introducción a Bonos


Aplicaciones con R


Gabriel Cabrera G.

1 de Octubre del 2018

- 1 Precio de un Bono
- 2 Otra cosa es con funciones
- 3 Relación precio del Bono y Yield
- 4 Graficar nuestros datos
- 5 Trabajando con yields reales
- 6 Duración y Convexidad de un Bono

Información de Contacto

 gcabrerag@fen.uchile.cl

 gcabrerag.rbind.io

 @GaboC_g

 @GaboCg

 Facultad de Economía & Negocios, Universidad de Chile

Precio de un Bono

Precio de un Bono: Ejemplo de Clases

Como se vio en clases el precio de un bono se calcula como:

$$P_B = \sum_{t=1}^T \frac{C}{(1+r)^t} + \frac{ValorNominal}{(1+r)^t}$$

Donde:

- P_B : Precio del Bono
- C_t : Pago intereses o cupones
- T : Números de períodos o madurez
- r : Tasa de descuento o yield-to-maturity semi-anual

Precio de un Bono: Utilizando R

Considere el siguiente ejercicio: Calcular el precio de un bono con pago de cupón semestral, Madurez 25 años, Tasa cupón 6.5%, Yield semi-anual de 6.9% y Valor nominal de 100. Para desarrollar el ejercicio, vamos a construir por parte los componentes de nuestro bono.

- 1 Construimos tanto la tasa cupón como la *Yield semi-anual*.

```
tc <- 0.065  
y <- 0.069
```

- 2 Construimos un vector con los valores de los cupones más el principal

```
pago <- c(rep(tc*100/2,49),(100 + tc*100/2))
```

tanto `tc`, `r` y `pago` se encontraran en Values del global environment, en tipo `numeric`.

Precio de un Bono: Utilizando R

- 3 Para poder trabajar con nuestra base de datos, transformamos nuestro vector pago que está en forma `numeric` a `data frame`.

```
pago <- as.data.frame(pago)
```

Ahora existe un objeto con estructuración de datos `data frame` en nuestro global environment. Se podría haber llamado de cualquier forma a nuestro nuevo objeto pago.

Precio de un Bono: Dos formas de hacer lo mismo

Ya construido nuestro objeto pago veremos que en **R** existen muchas formas de hacer lo mismo:

- 1 Al principio de la sesión cargamos la librería tidyverse, está nos permitirá trabajar con un “megapaquete” que incluye otros paquetes en su interior (ggplot2, dplyr, magittr, entre otros). Todos los paquetes que conforman “el Tidyverse” comparten la misma visión sobre el trabajo con datos y la escritura de código. Si va a la pestaña packages y escribe dplyr verá que está activa, pero nunca la “llamamos”, esto se debe a tidyverse lo hizo por nosotros.

```
pago1 <- pago %>%
  mutate(t1 = as.numeric(index(pago)), factor_desc = 1/(1+y/2)^(t1),
         val_present = pago*factor_desc) %>%
  summarise(sum(val_present))
```


Precio de un Bono: Dos formas de hacer lo mismo

Antes de continuar hablemos un poco del símbolo `%>%`. ¿Qué es `%>%`? Este símbolo se llama pipe (tubo en inglés) y en el código lo podemos leer como “luego” o “a continuación”. El atajo del teclado para realizarlo es `control + shift + m` (Linux / Windows) o `comando + shift + m` (Mac.).

`%>%` es un operador que nos permite encadenar las acciones que queremos aplicar en un objeto (en este caso, `pago`). Con este código le estamos diciendo a R que :

- tome el objeto `pago`
- luego(`%>%`)
- genere tres variables(`mutate` de la librería `dplyr`)
- luego(`%>%`)
- que resuma(`summarize`) la siguiente información;
- la suma (`sum`) de los valores de la columna `val_present`
- construyendo un nuevo objeto `pago1`

Precio de un Bono: Dos formas de hacer lo mismo

② La otra forma es:

```
# replicamos el objeto
pago2 <- pago

pago2$t2 <- as.numeric(rownames(pago2))

# Calculamos el factor de descuento
pago2$factor_desc <- 1 / (1 + r)^(pago2$t2)
# Calculamos el valor presente
pago2$val_present <- pago$pv_factor*pago2$pago
# Calculamos el precio
sum(pago2$val_present)
```

Otra cosa es con funciones

Otra cosa es con funciones

Ahora se contruirá una función para valorizar cualquier bono que pague cupones iguales:

```
# p: valor nominal; tc: tasa cupón; t: madurez; y: yield to maturity
precio.bono <- function(p,tc,t,y){
  # rep returns a vector with value = p * r and times = ttm -1
  pago    <- c(rep(tc*p, t - 1),p*(1 + tc))
  pago    <- as.data.frame(pago)
  pago$t  <- as.numeric(rownames(pago))
  pago$factor_desc <- 1 / (1 + y)^(pago$t)
  pago$valor_prese <- pago$factor_desc*pago$pago
  sum(pago$valor_prese)
}
```

```
precio.bono(100,0.065/2,50,0.069/2)
```

Relación precio del Bono y Yield

Relación precio del Bono y Yield: Valorización

Ahora utilizando la función `precio.bono` valorizaremos un bono con las siguientes características:

- Principal : 100
- Tasa Cupón: 5%
- Madurez: 10 años
- Yield: 4.29%

```
# Valoramos el siguiente Bono  
precio.bono(p = 100, tc = 0.05, t = 10, y = 0.0429)
```

Relación precio del Bono y Yield: Construcción yields

Se contruirá una secuencia de yields:

```
# Cosntruimos yields  
yields <- seq(0.02, 0.4, 0.01)
```

La función seq generará una secuencia. En este caso parte del 0.02 hasta el 0.4 pero con intervalos de 0.01.

```
# Convertimos yields a data frame como antes  
yields <- as.data.frame(yields)
```

Relación precio del Bono y Yield: Loop

Explicación en clases.

```
# Calculamos el precio del bono para distintas yields
for (i in 1:nrow(yields)) {
  yields$precio[i] <- precio.bono(100, 0.10, 20, yields$yields[i])
}
```


Graficar nuestros datos

Graficar nuestros datos: Con ggplot2

Una manera de visualizar datos es usar ggplot2, se recomienda que añadan por parte lo que desean en su gráfico.

```
# Graficamos con ggplot2
g1 <- ggplot(data = yields,aes(x = yields*100, y = precio)) + geom_line(size = 1.5, color = "red") +
g1 <- g1 + geom_point(size = 3, color = "red")
g1 <- g1 + ggtitle("Relación inversa:", subtitle = "Precio del Bono vs Yield")
g1 <- g1 + xlab("Yield (%)") + ylab("Precio del bono")
g1 <- g1 + geom_ribbon(aes(ymin = 0, ymax = pmax(precio,0)), fill="pink", col="red", alpha=0.5)
g1 <- g1 + theme_bw()
g1 <- g1 + theme(panel.border = element_rect(colour = "black", fill = NA, size = .5),
                  panel.grid.major = element_line(colour = "#d3d3d3"))

g1
```

```
# Guardamos gráfico
ggsave("retorno-yield.png",width = 8.5, height = 4.5, dpi = 300)
```

Graficar nuestros datos: Con plot

otra manera de visualizar datos es usar plot, esta opción es valida pero es más “arcaica” y más limitada que ggplot2.

```
# Con plot  
g2 <- plot(yields$yields*100,yields$precio,type = "l",col = "red",  
           main = "Relación inversa: Precio del Bono vs Yield",  
           xlab="Yield (%)", ylab="Precio del bono")
```

Trabajando con yields reales

Trabajando con yields reales: quantmod

quantmod es uno de las librerías más ocupadas en R para extraer datos financieros, te permite graficar, realizar análisis técnico, calcular retornos ($\Delta t(x)$), etc. Aunque las series son descargadas con estructura xts, la podemos transformar a data frame. A continuación descargaremos la yield de los bonos del tesoro de Estados Unidos a 10 años:

```
t10yr <- getSymbols(Symbols = "DGS10", src = "FRED", auto.assign = FALSE)
t10yr <- subset(t10yr["2000-01-01/2018-04-17"])
```

```
# Grafico con chartSeries de quantmod solo funciona con xts
chartSeries(t10yr)
```

```

t10yr.df <- as.data.frame(t10yr)

t10yr.df <- t10yr.df %>%
  mutate(fecha = as.Date(rownames(t10yr.df))) %>%
  na.omit()

g3 <- ggplot(data = t10yr.df, aes(x = fecha , y = DGS10)) + geom_line(size = 1, color = "green")
g3 <- g3 + ggtitle("10-Year US Treasury Yields", subtitle = "Desde 2000-01-01 hasta 2018-04-17")
g3 <- g3 + ylab("Fecha") + xlab("Yield(%)")
g3 <- g3 + theme_bw() + theme(panel.border = element_rect(colour = "black", fill = NA, size = .5),
                             panel.grid.major = element_line(colour = "#d3d3d3"))
g3

```

```

# Guardados gráfico
ggsave("treasury-yields.png", width = 8.5, height = 4.5, dpi = 300)

```

Duración y Convexidad de un Bono

Duración y Convexidad de un Bono: Extracción de la yield

La función subset se explica sola, permite extraer una parte de tu base según un criterio.

```
# Extraemos un valor en específico
t10yr_yield <- t10yr.df %>%
  subset(fecha == "2017-03-03")

t10yr_yield <- as.numeric(t10yr_yield$DGS10)*0.01
```


Duración y Convexidad de un Bono: Duración

Existen dos Duraciones, la de Macaulay y modificada (o de Hicks), las que miden sensibilidad del precio ante cambios de la yield.

Macaulay:

$$\text{Duración de Macaulay} = \left[\frac{1+y}{y} - \frac{1+y+[n \cdot (c-y)]}{[c \cdot ((1+y)^n - 1)] + y} \right]$$

Modificada:

$$\text{Duración Modificada} = \text{Duración de Macaulay} / (1 + y)$$

Aprox. Duración Modificada:

$$\text{Aprox. Dur. Mod.} = \frac{MV_- - MV_+}{2 \cdot \Delta y \cdot MV_0}$$

Duración y Convexidad de un Bono: Duración Macaulay

Duración Macaulay:

```
# duracion de Macaulay
macaulay <- function(y,n,c,t,T){
  mac <- (1 + y)/y - (1+y+(n*(c-y)))/(c*((1+y)^n - 1) + y)
  print(mac)
}
```

```
macaulay <- macaulay(t10yr_yield,10,0.03)
macaulay
```

Duración y Convexidad de un Bono: Duración Modificada

Duración Modificada:

```
# duración modificada  
modificada <- macaulay/(1+t10yr_yield)  
modificada
```

Duración y Convexidad de un Bono: Aproximación Duración Modificada

Aproximación Duración Modificada:

```
# Para la aproximación de la duración modificada  
precio.arriba <- precio.bono(p = 100, tc = 0.03, t = 10, y = t10yr_yield + 0.01)  
precio        <- precio.bono(p = 100, tc = 0.03, t = 10, y = t10yr_yield)  
precio.abajo  <- precio.bono(p = 100, tc = 0.03, t = 10, y = t10yr_yield - 0.01)
```

```
# Calculo de aproximación duración modificada  
aprox.dur.mod <- (precio.abajo - precio.arriba)/(2 * precio * 0.01)  
aprox.dur.mod
```

Duración y Convexidad de un Bono: Duración con librería

```
# Con librerías  
p_load("derivmks")  
  
# Duración moficada  
duration(precio, 3, 10, 100, 1, modified = TRUE)  
  
# Duración Macaulay  
duration(precio, 3, 10, 100, 1, modified = FALSE)
```

Duración y Convexidad de un Bono: Convexidad

Cuando las tasa de interés (yield) varían en demasiados puntos base, deja de ser la duración (cualquier tipo) una buena medida de sensibilidad y se recurre a la convexidad.

$$\text{Aproximación Convexidad: } \frac{MV_- + MV_+ - 2 * MV_0}{MV_0 * \Delta y^2}$$

```
# Calculamos medida de convexidad
```

```
convexidad <- (precio.arriba + precio.abajo - 2 * precio)/(precio * (0.01)^2)
convexidad
```

Convexidad con librería:

```
convexity(precio, 3, 10, 100, 1)
```

Introducción a Bonos

Aplicaciones con R

Gabriel Cabrera G.

1 de Octubre del 2018