1º Trabalho de Modelagem Matemática em Finanças II

Luiz R. S. de Souza

23 de agosto de 2018

Setup

Primeiramente, carregamos os pacotes necessários.

```
library(purrr)
library(dplyr)
library(cubature)
library(ggplot2)
```

Primeira Questão

Precisamos calcular a primeira, segunda e terceira variações de $x \mapsto x^2$ e $x \mapsto \sin x$ no intervalo $[0, 1]$. Nesse caso, podemos calcular a primeira variação diretamente usando a fórmula $var_{1,[a,b]}(f) = \int_a^b |f'(x)| dx$ para funções diferenciáveis, obtendo $var_{1,[0,1]}(x \mapsto x^2) = 1$ e $var_{1,[0,1]}(x \mapsto sin(x)) = sin(1)$. Podemos testar isso numericamente. Para isso, desenvolvi as funções abaixo.

```
# calcula a n_ésima variabilidade de um conjunto de valores
nth_var = function(fs, order) {
  sum(abs(diff(fs))^order)
}

# S = vetor com número de passos
# N = variação máxima (calcula de 1 até N)
# no intervalo [a, b]
var_table = function(f, S = 1000*2^(1:10), N = 3, a = 0, b = 1) {
  cbind(data.frame(Steps = S), # coluna com o número de passos
    map_dfr(S, function(s) {
      map(1:N, nth_var, fs = f(seq(a, b, length.out = s))) %>%
        set_names(paste0("Var", 1:N)) %>%
        as.data.frame
    })
  )
}
```

Obtemos:

```
var_table(sin)
```

```
##      Steps     Var1         Var2         Var3
## 1     2000 0.841471 3.638441e-04 1.608766e-07
## 2     4000 0.841471 1.818766e-04 4.019905e-08
## 3     8000 0.841471 9.092691e-05 1.004725e-08
## 4    16000 0.841471 4.546061e-05 2.511498e-09
## 5    32000 0.841471 2.272960e-05 6.278354e-10
## 6    64000 0.841471 1.136462e-05 1.569539e-10
## 7   128000 0.841471 5.682266e-06 3.923787e-11
## 8   256000 0.841471 2.841122e-06 9.809391e-12
## 9   512000 0.841471 1.420558e-06 2.452338e-12
```

```
## 10 1024000 0.841471 7.102784e-07 6.130834e-13
```

```
var_table(function(x) x^2)
```

```
##      Steps Var1         Var2         Var3
## 1     2000    1 6.670001e-04 5.005003e-07
## 2     4000    1 3.334167e-04 1.250625e-07
## 3     8000    1 1.666875e-04 3.125781e-08
## 4    16000    1 8.333854e-05 7.813477e-09
## 5    32000    1 4.166797e-05 1.953247e-09
## 6    64000    1 2.083366e-05 4.882965e-10
## 7   128000    1 1.041675e-05 1.220722e-10
## 8   256000    1 5.208354e-06 3.051782e-11
## 9   512000    1 2.604172e-06 7.629424e-12
## 10 1024000    1 1.302085e-06 1.907352e-12
```

Verifica-se que a primeira variação converge rapidamente para o valor exato, enquanto as segundas e terceiras variações parecem convergir linearmente/quadraticamente para 0.

Para o movimento browniano, o que podemos fazer é estimar a esperança da variabilidade.

```
# gera um caminho browniano avaliado em uma sequência ts de pontos
brownian = function(ts) {
  c(0, cumsum(rnorm(length(ts) - 1, sd = sqrt(diff(ts)))))
}

nTrials = 100
(map(1:nTrials, ~ as.matrix(var_table(brownian))) %>% reduce(`+`)) / nTrials
```

```
##         Steps      Var1      Var2        Var3
##  [1,]    2000  35.73575 1.0026715 0.035811766
##  [2,]    4000  50.49795 1.0006660 0.025245535
##  [3,]    8000  71.30989 0.9989026 0.017812959
##  [4,]   16000 100.81202 0.9980664 0.012580397
##  [5,]   32000 142.62662 0.9991415 0.008913208
##  [6,]   64000 201.86365 1.0001652 0.006311208
##  [7,]  128000 285.33360 0.9992133 0.004455472
##  [8,]  256000 403.74658 1.0001947 0.003154520
##  [9,]  512000 570.96401 1.0002078 0.002230892
## [10,] 1024000 807.45010 1.0001114 0.001577241
```

Daí se vê que o valor esperado da primeira variação é infinito, o da segunda variação é finito $= 1$, e a terceira é 0.

Segunda Questão

Determine a probabilidade de $B(1) \in [1, 2]$ e $B(2) \in [-3, -2]$ por Monte Carlo.

```
N = 1000000

bAt1 = rnorm(N, sd = 1)
bAt4 = bAt1 + rnorm(N, sd = sqrt(3))
sum(as.integer(between(bAt1, 1, 2) & between(bAt4, -3, -2))) / N
```

```
## [1] 0.002835
```

Terceira Questão

Escreva a integral do Item 2 e a calcule numericamente.

$$\int_1^2 \int_{-3}^{-2} \frac{1}{2\sqrt{3}\pi} \exp \frac{-x_1^2}{2} \exp \frac{-(x_2 - x_1)^2}{6} dx_2 dx_1$$

```
# usando a dnorm() densidade da normal já built-in na linguagem
density = function(x) {
  dnorm(x[1], sd = 1) * dnorm(x[2] - x[1], sd = sqrt(3))
}

adaptIntegrate(density, lowerLimit = c(1, -3), upperLimit = c(2, -2))
```

```
## $integral
## [1] 0.002796764
##
## $error
## [1] 2.586796e-08
##
## $functionEvaluations
## [1] 85
##
## $returnCode
## [1] 0
```

Quarta Questão

Determine por Monte Carlo a probabilidade de $B(t) < \sqrt{t} + 0.5$ para todo t.

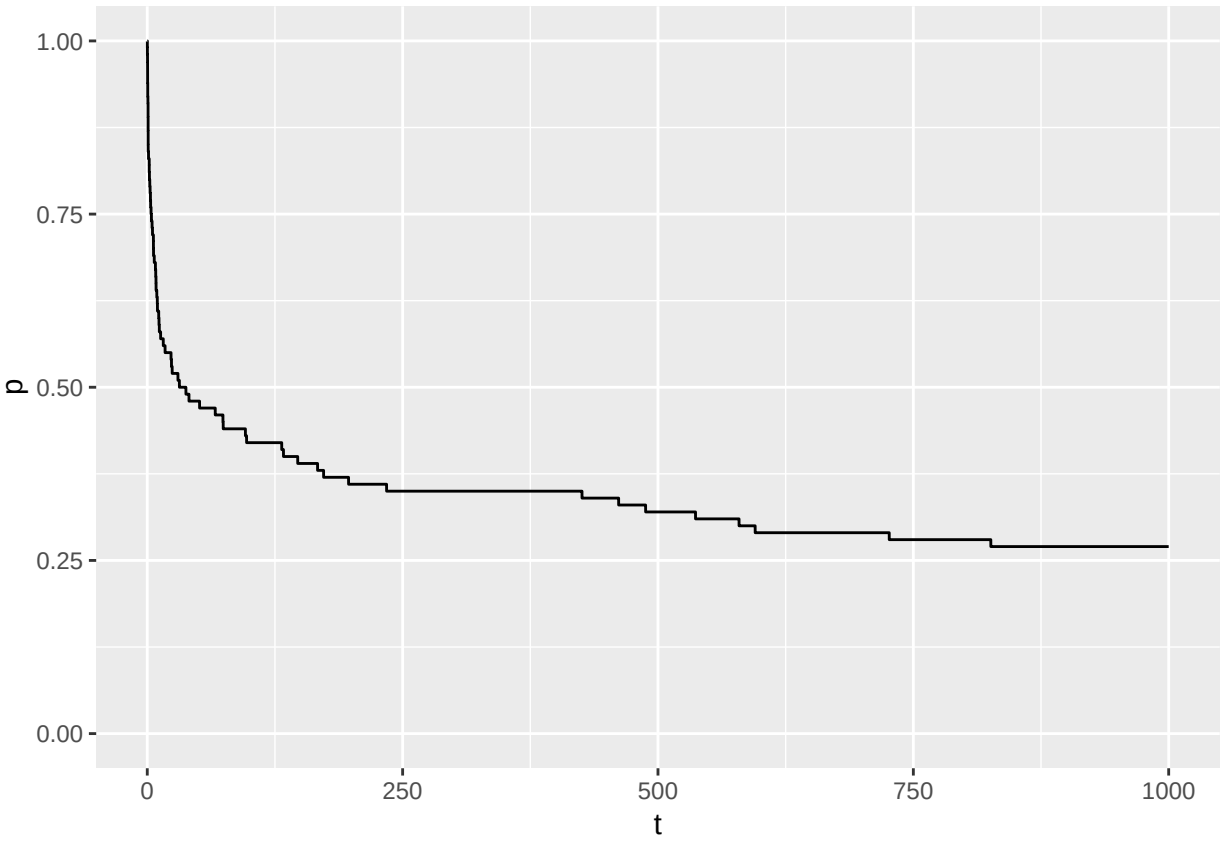
Primeiro faço uma simulação menor, guardando toda a trajetória dos movimentos brownianos, para analisar o comportamento dessa probabilidade à medida que T cresce. O gráfico sugere um decaimento convergente, mas não fica claro se é para 0 ou para alguma constante positiva.

```
ts = seq(0, 1000, by = 0.01)

probUpperBound = function(ts, nTrials) {
  # gera n_trial vetores de 0 ou 1, cada um deles do tamanho de um run browniano,
  # indicando se o upper bound é valido até aquele instante
  nSuccesses = map(1:nTrials, function(...) {
    as.integer(brownian(ts) <= sqrt(ts) + .5) %>% cumprod
  }) %>% c(recursive = TRUE) %>%
    matrix(nrow = nTrials, byrow = TRUE) %>% # transforma numa matriz e soma as colunas
    colSums()
  nSuccesses / nTrials
}

p = probUpperBound(ts, 100)

ggplot(data.frame(t = ts, p = p), aes(x = t, y = p)) + geom_path() + ylim(0, 1)
```



Para conseguir simular para T indo a infinito, faço uma pequena modificação na função de modo a não computar as probabilidades para todos os valores de T, mas apenas para o T final. Também aumento os incrementos do movimento browniano.

```
probUpperBound2 = function(ts, nTrials) {
  # gera n_trial vetores de 0 ou 1, cada um deles do tamanho de um run browniano,
  # indicando se o upper bound é valido até aquele instante
  nSuccesses = map_int(1:nTrials, function(...) {
    as.integer(all(brownian(ts) <= sqrt(ts) + .5))
  }) %>% sum
  nSuccesses / nTrials
}

10^(3:7) %>%
  set_names(.) %>%
  map_dbl(~ probUpperBound2(seq(0, ., by = 1), nTrials = 100))
```

```
##  1000 10000 1e+05 1e+06 1e+07
##  0.31  0.19  0.10  0.06  0.04
```

Os resultados sugerem que a probabilidade de um run nunca ultrapassar a barreira de $\sqrt{t} + .5$ vai para 0 à medida que t vai a infinito.