Lista 1
Macroeconomia II

Yuri Passuelo – yuripassuelo@usp.br

August 23, 2025

Resolução dos exercícios da primeira lista da disciplinas de Macroeconomia II.

Cálculo do Equity Premium no Brasil

Neste exercício, você utilizará o modelo de Mehra e Prescott (1985) para calcular o Equity Premium no Brasil. Considere os seguintes estados para o crescimento do dividendo:

$$X = \begin{bmatrix} 0.1359 \\ 0.5949 \\ 1.0539 \\ 1.5129 \\ 1.9719 \end{bmatrix}$$

E a seguinte matriz de transição associada ao crescimento do consumo:

$$M = \begin{bmatrix} 0.0300 & 0.3262 & 0.5175 & 0.1224 & 0.0040 \\ 0.0190 & 0.2679 & 0.5420 & 0.1642 & 0.0069 \\ 0.0116 & 0.2131 & 0.5505 & 0.2131 & 0.0116 \\ 0.0069 & 0.1642 & 0.5420 & 0.2679 & 0.0190 \\ 0.0040 & 0.1224 & 0.5175 & 0.3262 & 0.0300 \end{bmatrix}$$

(a) Apresente a função valor do agente representativo desta economia.

(b) Apresente a definição de um equilíbrio competitivo recursivo para esta economia.

(c) Apresente uma solução anal ıtica para os preços do *equity* e de um ativo livre de risco.

(d) Utilizando os dados dos vetores X e M, compute o *Equity Premium* para o Brasil, variando o fator de desconto β e o coeficiente de aversão relativa ao risco σ. Explique os efeitos dessas variáveis sobre o *Equity Premium*.

(e) Colete dados sobre os retornos dos ativos arriscados e livre de risco no Brasil e compare o Equity Premium dos dados com aqueles computados no item anterior. Você conclui que há um puzzle?

(f) Simule séries temporais não estacionárias para os preços nesta economia. Comente sobre o comportamento das séries simuladas.

Solution

Vamos inciar relembrando o problema básico do Equity Premium, aqui temos:

- Individuo tem opção de alocar renda entre consumo e ativos;
- Ativos disponíveis são:
    1. Arvore s que gera dividendos y com taxa de retorno x e tem preço $p(x, y)$
    2. *Bonds* B que pagam uma unidade de consumo c ao fim do período e tem preço $q(x, y)$

(a) Do problema apresentado podemos construir nosso problema recursivo:

$$V(s, B, x, y) = \max \left\{ u(c) + \beta \sum_{x} \pi(x', x) V(x', y', B') \right\}$$

Sujeito a restrição:

$$c + p(x, y)s' + q(x, y)B' \leq (p + x)s + B$$

Em resumo opções de gastos são:

1. Consumo c
2. Compra do ativo s' a preços $p(x, y)$
3. Compra do ativo B' a preços $q(x, y)$

Se quisermos simplificar alguns iténs, assumimos que tanto s quanto B pertencem ao que chamaremos de w, de forma que nosso problema recursivo será:

$$V(w, x, y) = \max \left\{ u(c) + \beta \sum_{x} V(w', x', y') \right\}$$

(b) Para essa economia um equilíbrio competitivo é definido como (i) Uma função Valor, (ii) V funções politicas g_c, g_B e g_s; e (iii) funções preço $p(x, y)$ e $q(x, y)$ tais que:

1. Dada funções preço $p(x,y)$ e $q(x,y)$ as funções valor V e politicas g_B, g_s e g_c resolvem o problema do consumidor

2. Market Clearing:

$$s' = g_s(w,x,y) = 1$$

$$B' = g_B(w,x,y) = 0$$

$$c = y$$

(c) Derivando as soluções, precisamos diferenciar nossa função valor tanto por s' quanto B', começando por s':

$$\frac{\partial V(w,x,y)}{\partial s'} = -p(x,y)u'(c) + \beta \sum_x \pi(x',x)\frac{\partial V(w',x',y')}{\partial s'} = 0$$

Usando Benveniste Sheickman para a segunda parte da equação (diferenciação da função valor) temos:

$$\frac{\partial V(w',x',y')}{\partial s'} = (p(x',y') + x')u'(c')$$

Substituindo:

$$\frac{\partial V(w,x,y)}{\partial s'} = -p(x,y)u'(c) + \beta \sum_x \pi(x',x)(p(x',y') + x')u'(c') = 0$$

Remanejando a equação temos:

$$p(x,y)u'(c) = \beta \sum_x \pi(x',x)(p(x',y') + x')u'(c')$$

Isolando $p(x,y)$:

$$p(x,y) = \beta \sum_x \pi(x',x)(p(x',y') + x')\frac{u'(c')}{u'(c)}$$

Vamos a duas importantes substituições aqui, a primeira é que como no exemplo original de Mehra e Prescott (1985), nossa utilidade aqui é:

$$u(c) = \frac{c^{1-\sigma}}{1-\sigma}$$

E temos ao mesmo tempo a etapa de Guess, aonde colocamos que os preços $p(x,y)$ são lineares em relação ao valor do dividendo do ativo s, portanto:

$$p(x,y) = p_i y$$

Desse modo temos:

$$p_i y = \beta \sum_x \pi(x',x)(p_i y + x')(\frac{c'}{c})^{1-\sigma}$$

Outro ponto importante de se relembra do inicio, em equilibrio o individuo iguala seu consumo c ao valor do dividendo recebido pela arvore, de modo que:

$$c = y$$
$$c' = y' = x'y$$

Portanto:

$$p_i y = \beta \sum_x \pi(x',x)(p_i y' + x')(\frac{y'}{y})^{-\sigma}$$

$$p_i y = \beta \sum_x \pi(x',x)(p_i x' y + x')(\frac{x'y}{y})^{-\sigma}$$

$$p_i = \beta \sum_x \pi(x',x)(p_i x' + x')x'^{-\sigma}$$

$$p_i = \beta \sum_x \pi(x',x)(p_i + 1)x'^{1-\sigma}$$

Separando o somatório temos:

$$p_i = \beta \sum_x \pi(x',x) p_i x'^{1-\sigma} + \beta \sum_x \pi(x',x) x'^{1-\sigma}$$

Fazendo uma pequena substituição em relação a índices e somatórios temos:

$$p_i = \beta \sum_{j=1}^{N} \pi(x_j,x_i) p_i x_j^{1-\sigma} + \beta \sum_{j=1}^{N} \pi(x_j,x_i) x_j^{1-\sigma}$$

Como temos ao todo N estados definidos pela nossa cadeia de Markov relacionada aos retornos dos dividendos, podemos montar um sistema de equações e colocamos na forma matricial.

$$\boldsymbol{A} = \beta \boldsymbol{\pi} \odot \bar{\boldsymbol{X}}$$

Aonde:

$$\boldsymbol{X} = \begin{bmatrix} x_1^{1-\sigma} \\ \vdots \\ x_N^{1-\sigma} \end{bmatrix}, \bar{\boldsymbol{X}} = \begin{bmatrix} ... & \boldsymbol{X^T} & ... \\ & \vdots & \\ ... & \boldsymbol{X^T} & ... \end{bmatrix}$$

Outra matriz importante é:

$$\boldsymbol{B} = \beta \boldsymbol{X^T} \boldsymbol{\pi}$$

De modo que tenhamos:

$$\boldsymbol{p} = \boldsymbol{pA} + \boldsymbol{B}$$

$$\boldsymbol{p} - \boldsymbol{pA} = \boldsymbol{B}$$

$$\boldsymbol{p(I - A) = B}$$

$$\boldsymbol{p = (I - A)^{-1} B}$$

Definido o preço de equilíbrio para o ativo s' vamos ao ativo B' começando pela sua C.P.O:

$$\frac{\partial V(w,x,y)}{\partial B'} = -q(x,y)u'(c) + \beta \sum_x \pi(x',x) \frac{\partial V(w',x',y')}{\partial B'} = 0$$

Usando Benveniste Scheikman para a diferenciação da função valor no lado direito temos:

$$\frac{\partial V(w',x',y')}{\partial B'} = u'(c')$$

Substituindo na Equação anterior temos:

$$\frac{\partial V(w,x,y)}{\partial B'} = -q(x,y)u'(c) + \beta \sum_x \pi(x',x) u'(c') = 0$$

Logo:

$$q(x,y)u'(c) = \beta \sum_x \pi(x',x) u'(c')$$

Isolando $q(x,y)$ temos:

$$q(x,y) = \beta \sum_x \pi(x',x) \frac{u'(c')}{u'(c)}$$

Aqui podemos substituir a utilidade marginal dado a parametrização da nossa função utilidade:

$$u(c) = \frac{c^{1-\sigma}}{1-\sigma}$$

Logo:

$$q(x, y) = \beta \sum_{x} \pi(x', x)(\frac{c'}{c})^{-\sigma}$$

Da mesma forma temos que em equilíbrio $c = y$ e $c' = y' = x'y$ e portanto:

$$q(x, y) = \beta \sum_{x} \pi(x', x)(\frac{x'y}{y})^{-\sigma}$$

$$q(x, y) = \beta \sum_{x} \pi(x', x)x'^{-\sigma}$$

Portanto os preços podem ser determinados dessas maneiras, para $p(x, y)$ resolvendo o sistema de equações, e para $q(x, y)$ computando o preço.

(d) Vamos ao desenho de um algoritmo em Python para que possamos computar, de acordo com o modelo visto acima.

```
import numpy as np
import matplotlib.pyplot as plt

# Lista 1 - Parametrizacao

X  = np.array([ 0.1359,
                0.5949,
                1.0539,
                1.5129,
                1.9719])

pi = np.array([ [0.0300, 0.3262, 0.5175, 0.1224, 0.0040],
                [0.0190, 0.2679, 0.5420, 0.1642, 0.0069],
                [0.0116, 0.2131, 0.5506, 0.2131, 0.0116],
                [0.0069, 0.1642, 0.5420, 0.2679, 0.0190],
                [0.0040, 0.1224, 0.5175, 0.3262, 0.0300]])
# Funcao para Calculo do Premio de Risco
def risk_pre( beta, sigma, pi=pi, X=X ):

    # Distribui  o Invariante
    pi_bar = np.array( np.matrix(pi)**1000 )

    # Tamanho de X
    l_x    = len( X )

    # Matrizes Formatadas X
    X_s1 = X**(1-sigma)
    X_s2 = X**( -sigma)
```

```python
    # Matrizes para Calculo (A e B)
    # Matriz A
    A = beta*np.full( (l_x,l_x), [X_s1] )*pi

    # Matriz B
    B = beta* ( X_s1.T @ pi )

    # Precos da arvore
    p = np.linalg.inv( np.identity( l_x ) - A  ) @ B

    # Retorno da arvore
    m_g_p = np.meshgrid( p, p )
    R  = ( (m_g_p[0] + np.ones( (l_x,l_x))) * np.full( (l_x,l_x), X) -
                                        m_g_p[1] )/ m_g_p[1]
    # Retorno esperado
    Re = ( pi.T * R ) @ np.ones( l_x )

    # Retorno esperado - Longo Prazo
    r_e = np.dot(Re, pi_bar[0] )

    # Parte do Bond:
    Q = beta * pi @ X_s2

    # Retornos Bonds Livres de Risco
    r_f = np.sum( ( 1/Q - 1) * pi_bar[0] )

    return ( r_e, r_f, r_e-r_f )

# Construindo Grids de Combina  es Distintas de \beta e \sigma
beta_grid = np.linspace( 0.1, 1, 100 )
sigm_grid = np.linspace( 0.1, 5, 100  )

# Mesh Grid
bet, sig = np.meshgrid( beta_grid, sigm_grid )

# Combinacoes
comb = np.array( [ [ risk_pre( beta = b, sigma = s) for s in sigm_grid ]
                                      for b in beta_grid ] )

# Plots
# Reorganizando Vetores
z_0 = comb[:,:,0]
z_1 = comb[:,:,1]
z_2 = comb[:,:,2]
```

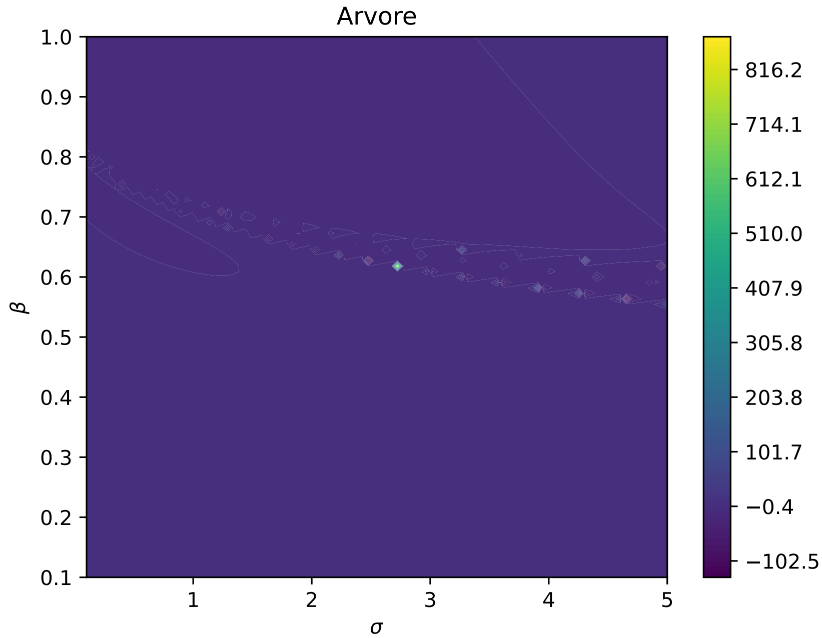

Figure 1: Heatmap Retorno Arvore

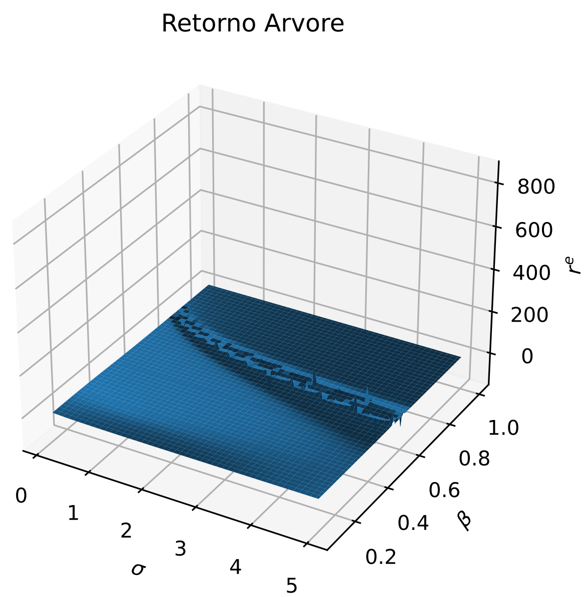

Figure 2: Superficie Retorno Arvore

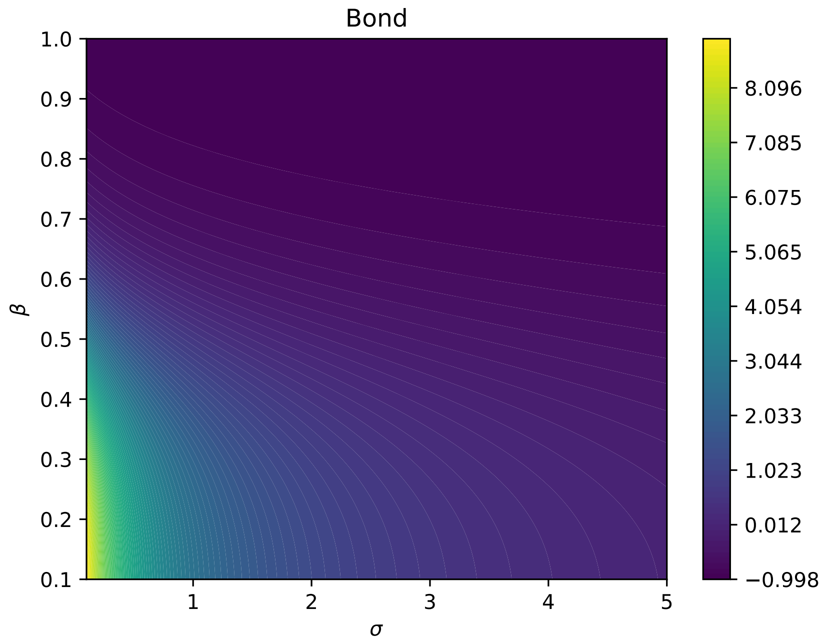

Figure 3: Heatmap Retorno Bond

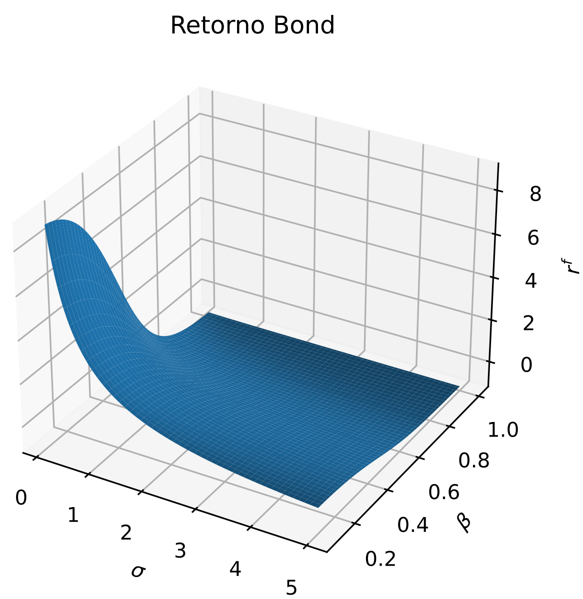

Figure 4: Superficie Retorno Bond

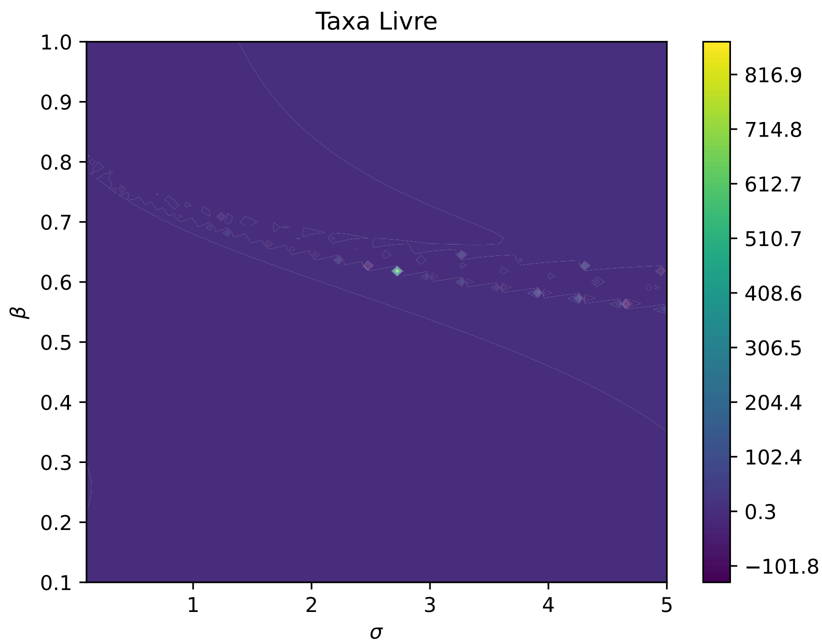

Figure 5: Heatmap Risk Premium

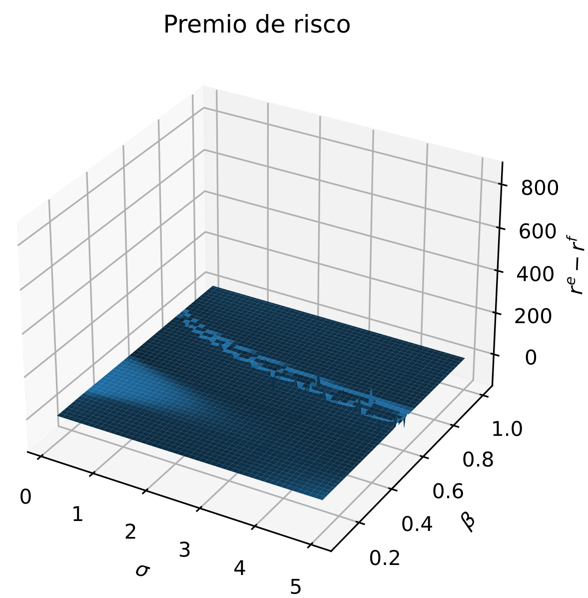

Figure 6: Superficie Risk Premium

Atenção: A matriz X[1] apresenta valores extremos demais o que pode causar retornos estranhos e imprecisos, os gráficos de superficie acima relatam os potenciais problemas, para driblar esses problemas e poder continuar resolvendo os exercicios vamos a baixo propor uma parametrização um pouco diferente:

$$X = [0.93041763, 0.97779789, 1.02517815, 1.0725584, 1.11993866]$$

$$M = \begin{bmatrix} 0.083 & 0.494 & 0.384 & 0.038 & 0 \\ 0.03 & 0.353 & 0.517 & 0.098 & 0.002 \\ 0.009 & 0.206 & 0.57 & 0.206 & 0.009 \\ 0.002 & 0.098 & 0.517 & 0.353 & 0.03 \\ 0 & 0.038 & 0.384 & 0.494 & 0.083 \end{bmatrix}$$

Aqui temos Vetores discretizados e suas respectivas matrizes Markovianas usando os dados de despesas de consumo das famílias dessazonalizadas, a discretização feita por meio do método de Tauchen com 5 pontos e um λ de 7.

Com os resultados corrigidos acima temos uma melhor visão de como cada variável β e σ influência sobre o retorno livre de risco:

- Quanto maior σ ou seja a propensão ao risco maior o premio de risco;
- Quanto menor β ou seja o fator de impaciência, maior o premio de risco;

(e) Para a economia Simulada com os parâmetros de:

- $\beta = 0.99$
- $\sigma = 2.0$

Variáveis	Modelo
r^e	4.20%
r^f	5.84%
$r^e - r^f$	-1.64%

(f) Vamos agora simular séries para os preços nessa economia, para isso vamos usar a seguinte função para simular uma cadeia de markov:

```
def simulate_markov(M, T, x0=2):
    # P( x' = Proximo | x = Atual )
    states = np.arange(M.shape[0])
    s = np.empty(T, dtype=int)
    s[0] = x0
    for t in range(1, T):
        s[t] = np.random.choice(states, p=M[s[t-1], :])
    return s
```

[1] Aqui apresentamos valores discretizados da variação da série de consumo no Brasil

E vamos setar os preços como:

```
T = 2000
s = simulate_markov(pi, T, x0=2)

# Simulando Cadeia de Markov

# Simula dividendos Y_t (levels) ---
Y = np.empty(T)
Y[0] = 1.0   # n vel inicial
for t in range(1, T):
    Y[t] = X[s[t]] * Y[t-1]  # Y_{t} = x_t * Y_{t-1}

# Pre os P_t ---
P = p[s] * Y   # p[s] seleciona p_{x_t}

# An lises b sicas
logP = np.log(P)
logY = np.log(Y)
rets = np.diff(logP)
```

Assim nossa simulação fica com a seguinte cara:

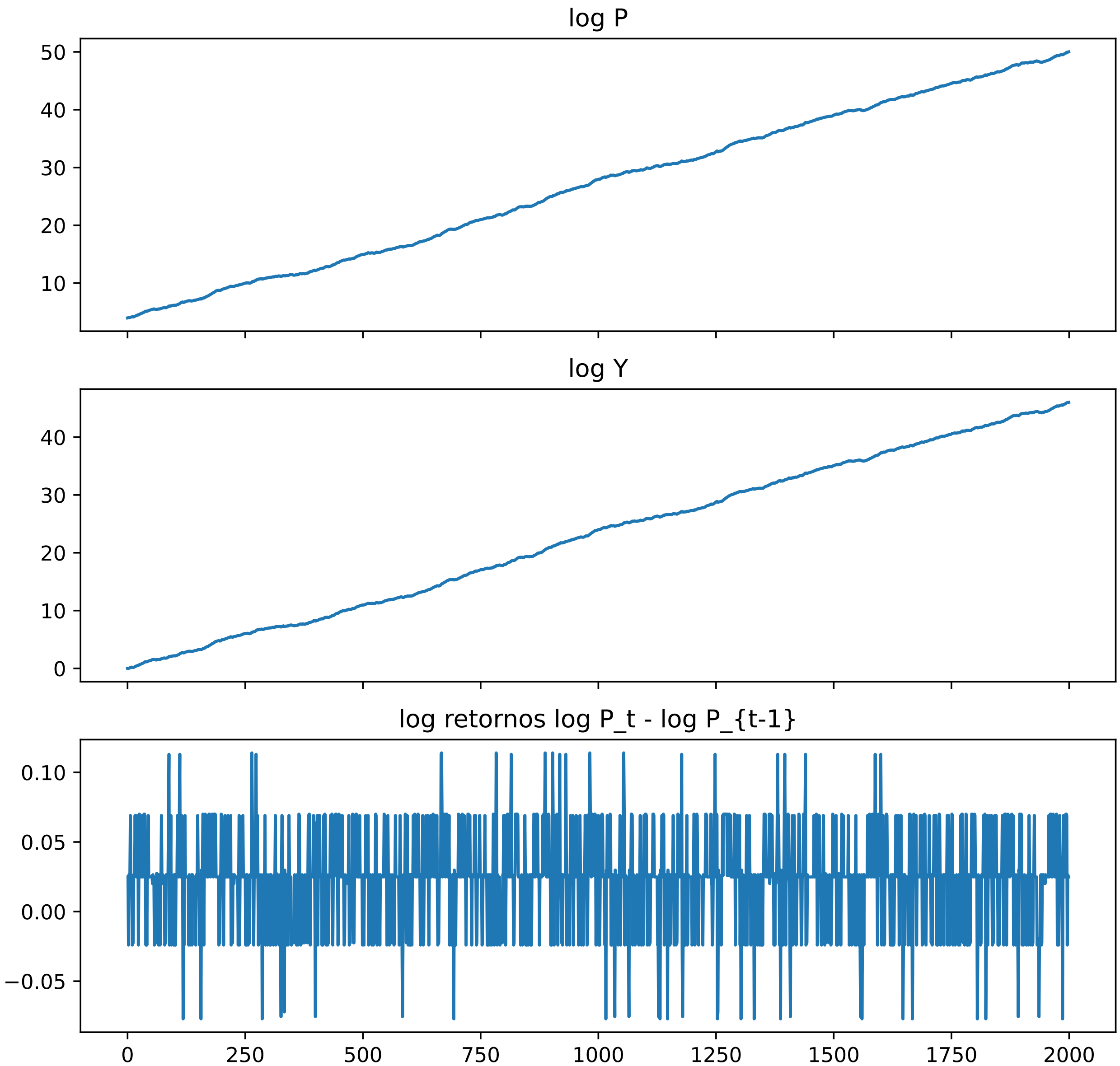

Figure 7: Simulações de Preços e Dividendos

As escalas estão em log, pois a discretização que usamos mostra que de 5 pontos nossos retornos são positivos em três, o que mostra uma tendencia de crescimento quase que exponencia e por isso temos esse movimento forte de crescimento tanto dos preços quanto dos dividendos.