Coursework 3

1 Main part

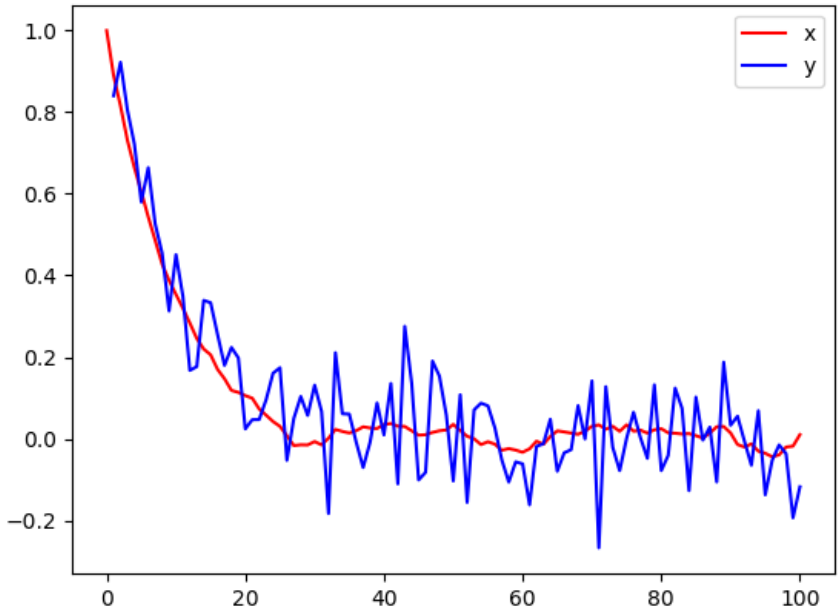
1.1 Q1

1. Here is a scatter plot of the Markov process from the coursework.

1.2 Q2

1. Simulate a Gaussian time-series corrupted by noise In the real world, x could model the exponential decay of some quantity (for example amount of radioactive matter) and y could be a measurement done of x. Notably, the error $\Delta = \|y - x\|$ is not proportional to x because the standard deviation σ_y is independent of x.

Coursework 3



2. Develop a stochastic volatility model and simulate The stochastic volatility model from [Lop] was implemented. Instead of the random variable x being the volatility (i.e. standard deviation of returns y), here it is the logarithm of the volatility. This allows it to be negative and thus a simple normal distribution may be employed from which x is sampled.

$$x_t|x_{t-1} \sim \mathcal{N}(x_t; 1 + 0.98 \cdot (x_{t-1} - 1), (0.1)^2),$$
$$y_t|x_t \sim \mathcal{N}(y_t; 0, (e^{x_t})^2)$$

Here are examples of data generated with the model. Instead of the logarithm of the volatility, x, the plot shows the volatility itself, i.e. e^x.

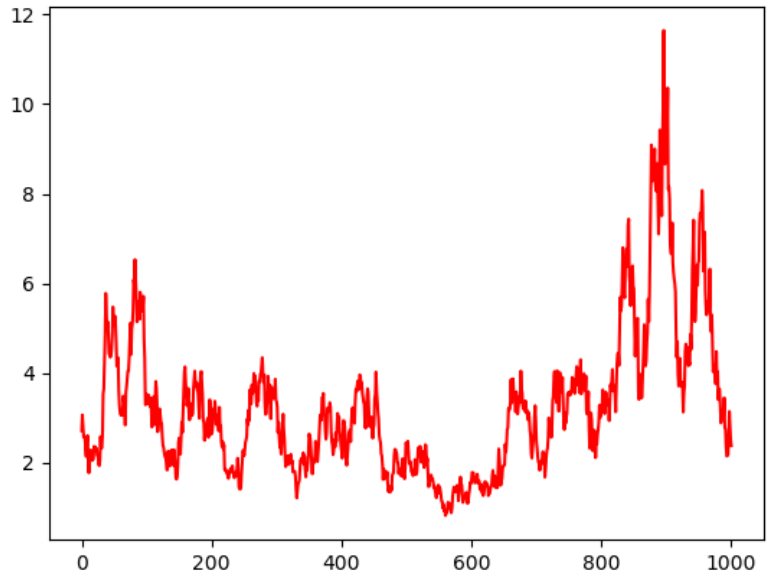


Figure 1: The volatility e^x

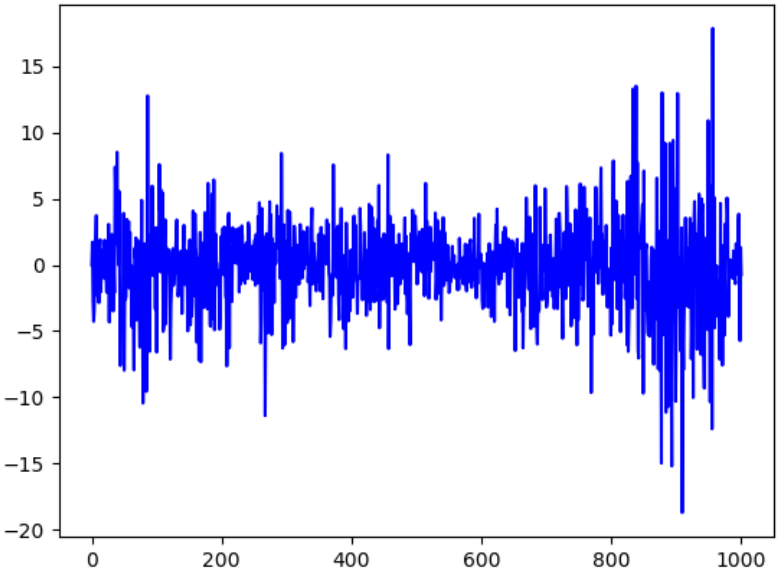


Figure 2: The returns y

2 Appendix with code

2.1 Code for Q1

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

A1 = np.matrix([[0.4, -0.3733],[0.06, 0.6]])
A2 = np.matrix([[-0.8, -0.1867],[0.1371, 0.8]])

b1 = np.array([[0.3533,0.0]]).T
b2 = np.array([[1.1,0.1]]).T

def f(i,x):
    if i == 1:
        return A1 @ x + b1
    elif i == 2:
        return A2 @ x + b2

x = [np.array([[0,0]]).T]
for n in range(1,10001):
    i_n = np.random.choice([1, 2], p=[0.2993, 0.7007])
    x.append(f(i_n,x[-1]))

x = np.array(x)

plt.scatter(x[20:,0], x[20:,1], s=0.1, color=[0.8, 0, 0])
plt.gca().spines['top'].set_visible (False)
plt.gca().spines['right'].set_visible (False)
plt.gca().spines['bottom'].set_visible (False)
plt.gca().spines['left'].set_visible (False)
plt.gca().set_xticks ([])
plt.gca().set_yticks ([])
plt.gca().set_xlim(0, 1.05)
plt.gca().set_ylim(0, 1)
plt.show()
```

2.2 Code for Q2

1. Simulate a Gaussian time-series corrupted by noise

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

a = 0.9
sigma_x = 0.01
sigma_y = 0.1

x = [1]
y = []
for t in range(1,101):
```

Coursework 3

```
    x.append(np.random.normal(a*x[-1], sigma_x))
    y.append(np.random.normal(x[-1], sigma_y))

plt.plot(range(101), x, color="red", label="x")
plt.plot(range(1,101), y, color="blue", label="y")
plt.legend()
plt.show()
```

2. Develop a stochastic volatility model and simulate The stochastic volatility model from [Lop] was implemented. Instead of the random variable x being the volatility (i.e. standard deviation of returns y), here it is the logarithm of the returns. This allows it to be negative and thus a simple normal distribution may be employed from which x is sampled.

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

# Logarithm of volatility is initially 1
x = [1]
# Observed returns start at 0
y = [0]

for n in range(1,1001):
    x.append(np.random.normal(1+0.98*(x[-1]-1), 0.1))
    y.append(np.random.normal(0, np.exp(x[-1])))

plt.plot(range(1001), np.exp(x), color="red", label="x")
plt.show()
plt.plot(range(1001), y, color="blue", label="y")
plt.show()
```

Bibliography

[Lop] Hedibert Freitas Lopes. *Econometria Avancada Lecture 9*. URL: http://hedibert.org/wp-content/uploads/2015/04/EconometriaAvancada-aula9.pdf (visited on 14/12/2022).