Python for Quants

A Hitchhikers Guide to Snake Charming!

Subramanian Swaminathan
Fabio Marelli

November 19, 2018

Python! What is Python?

- **high-level, garbage-collected, general-purpose** programming language that is both **easy to learn and use**
- encourages **modularity of code** by supporting **modules and packages**
- it supports **object-oriented and functional programming** paradigms
- shining feature is an excellent ecosystem, be it **IDE, standard libraries and/or community**

Python! What is Python?

- **high-level, garbage-collected, general-purpose** programming language that is both **easy to learn and use**
- encourages **modularity of code** by supporting **modules and packages**
- it supports **object-oriented and functional programming** paradigms
- shining feature is an excellent ecosystem, be it **IDE, standard libraries and/or community**

Python! What is Python?

- **high-level, garbage-collected, general-purpose** programming language that is both **easy to learn and use**
- encourages **modularity of code** by supporting **modules and packages**
- it supports **object-oriented and functional programming** paradigms
- shining feature is an excellent ecosystem, be it **IDE, standard libraries and/or community**

Python! What is Python?

- **high-level, garbage-collected, general-purpose** programming language that is both **easy to learn and use**
- encourages **modularity of code** by supporting **modules and packages**
- it supports **object-oriented and functional programming** paradigms
- shining feature is an excellent ecosystem, be it **IDE, standard libraries and/or community**

Python! What is Python?

- **high-level, garbage-collected, general-purpose** programming language that is both **easy to learn and use**
- encourages **modularity of code** by supporting **modules and packages**
- it supports **object-oriented and functional programming** paradigms
- shining feature is an excellent ecosystem, be it **IDE, standard libraries and/or community**

Functions

function definition

```
1  def function_name(pos, *args, **kwds):
2    """ documentation string """
3    body
```

Three important components of a function are it's:

- arguments
- body
- environment

greetings.py

```
1  def greetings(name):
2    """ takes a string and writes a message to the console. """
3    print('Hello, {whom}. It is nice meeting
     ↪  you!'.format(whom=name))
```

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Lists

Is it true that **[ ]** is a list?

Yes,
Anything enclosed by a [and] is a list.This special data structure is a called an **empty** list.

Is it true that **[1, 2, 3, ..., 10]** is a list?

Yes,
This is a list of first 10 **integers**.

Is it true that **[[ ], [ ]]** is a list?

Yes. This is a two element list of lists. The elements of this list are empty lists.

Are these lists? ( ), { }, ( ], and [ )

No. Lists should begin and end with square brackets.

Data Structures - Strings

Is it true that ‘ ’ is a string?

Yes,
Anything enclosed by a ‘ and ’ is a string.This special data structure is a called an **empty** string.

Is it true that “ ” is a string?

Yes,
Anything enclosed by a “ and ” is also a string.

Is it true that ‘ **python rocks** ” is a string?

No. This is an ill-formed operation. You have a single quote on the left side and a double on the right.

Data Structures - Strings

Is it true that ‘ ’ is a string?

Yes,
Anything enclosed by a ‘ and ’ is a string.This special data structure is a called an **empty** string.

Is it true that “ ” is a string?

Yes,
Anything enclosed by a “ and ” is also a string.

Is it true that ‘ **python rocks** ” is a string?

No. This is an ill-formed operation. You have a single quote on the left side and a double on the right.

Data Structures - Strings

Is it true that ‘ ’ is a string?

Yes,
Anything enclosed by a ‘ and ’ is a string.This special data structure is a called an **empty** string.

Is it true that “ ” is a string?

Yes,
Anything enclosed by a “ and ” is also a string.

Is it true that ‘ **python rocks** ” is a string?

No. This is an ill-formed operation. You have a single quote on the left side and a double on the right.

Data Structures - Strings

Is it true that ‘ ’ is a string?

Yes,
Anything enclosed by a ‘ and ’ is a string.This special data structure is a called an **empty** string.

Is it true that “ ” is a string?

Yes,
Anything enclosed by a “ and ” is also a string.

Is it true that ‘ **python rocks** ” is a string?

No. This is an ill-formed operation. You have a single quote on the left side and a double on the right.

Data Structures - Strings

Is it true that ‘ ’ is a string?

Yes,
Anything enclosed by a ‘ and ’ is a string.This special data structure is a called an **empty** string.

Is it true that “ ” is a string?

Yes,
Anything enclosed by a “ and ” is also a string.

Is it true that ‘ **python rocks** ” is a string?

No. This is an ill-formed operation. You have a single quote on the left side and a double on the right.

Data Structures - Strings

Is it true that ‘ ’ is a string?

Yes,
Anything enclosed by a ‘ and ’ is a string.This special data structure is a called an **empty** string.

Is it true that “ ” is a string?

Yes,
Anything enclosed by a “ and ” is also a string.

Is it true that ‘ **python rocks** ” is a string?

No. This is an ill-formed operation. You have a single quote on the left side and a double on the right.

Looping through Lists

for loop

```
1 for item in iterable:
2   do-something
```

while loop

```
1 while condition-holds:
2   do-something
```

```
1 friends = "Batman, Superman, Wonder Woman".split(',')
```

for_loop.py

```
1 for friend in friends:
2   print('Hello, {char}'.format(char=friend.strip()))
```

while_loop.py

```
1 friends_copy = friends[:]  #start by leaving out [:]
2 while friends_copy != []:
3   print('Hello,
  ↪  {friend}'.format(friend=friends_copy.pop().strip()))
```

Branching and List Comprehension

if statements

```
if test-1:
  code to run should test 1 succeed
elif test-2:
  code to run should test 2 succeed
else:
  run this should all other tests fail
```

multiway_branching.py

```
heros = [h.strip() for h in "Batman, Superman, Wonder Woman".split(',')]
for hero in heros:
  if hero == 'Batman':
    print('I am Batman.')
  elif hero == 'Superman':
    print('Up, up and away!')
  else:
    print('Sorry boys! Amazonians don\'t market', sep='')
```

Import statement

import

```
1  import module-name                          #variant 1
2  from   module-name import name [,names]     #variant 2
3  from   module-name import *                 #variant 3
4  import module-name as new-name              #variant 4
```

cipher.py

```
1  def rot(string, n):
2    def shift(c): return shift(c.lower()).upper() if c.isupper() else
     ↪  chr((((ord(c) - 97) + n) % 26) + 97)
3    return ''.join([(lambda c: shift(c) if c.isalpha() else c)(ch) for ch
     ↪  in string])
```

```
1  prompt> import cipher
2  prompt> cipher.rot('hal', 1)
3  prompt> 'ibm'
```

Consolidation

ratings.py

```
1  import random
2
3  def ratings():
4    courses = ['Python for Quants'
5      , 'Monte Carlo Methods'
6      , 'Financial Engineering for Pedestrians'
7      , 'Least Square Monte Carlo Methods...']
8
9    draw_stars = lambda n: '*' * n  #pay attention to this line
10
11   ratings = "5 4 3 2 1".split()
12   print('PRINTING COURSE RATINGS...')
13   for course in courses:
14     if course == 'Python for Quants':
15       print('| {ratings:5s} |. {course_title}'.format(ratings=
         ↪  draw_stars(5), course_title = course))
16     else:
17     print('| {ratings:5s} |. {course_title}'.format(ratings=
       ↪  draw_stars(int(random.choice(ratings))), course_title = course))
```

IO Operations

read mode

```
1 with open(fname, 'r') as f:
2   do-something
```

write mode

```
1 with open(fname, 'w') as f:
2   do-something
```

read_write.py

```
1 from cipher import rot
2
3 def readfile(fname):
4   with open(fname, 'r') as f:
5     return f.readlines()
6
7 def writelines(infile, outfile, shiftby):
8   with open(outfile, 'w') as f:
9     f.writelines(rot(''.join(readfile(infile)), shiftby))
```

```
1 prompt> #writefile(infile,outfile,shiftby)
2 prompt> writefile('cipher.py', 'secret.py', 1)
```

What is a Brownian motion?

A stochastic process $t \longmapsto W_t$ is a Brownian motion if:

- $W_0 = 0$
- W_t has independent increments
- $W_t - W_s$ is $\mathbf{N}(0, t-s)$, a gaussian random variable with mean 0 and variance $t - s$.

What is a Brownian motion?

A stochastic process $t \longmapsto W_t$ is a Brownian motion if:

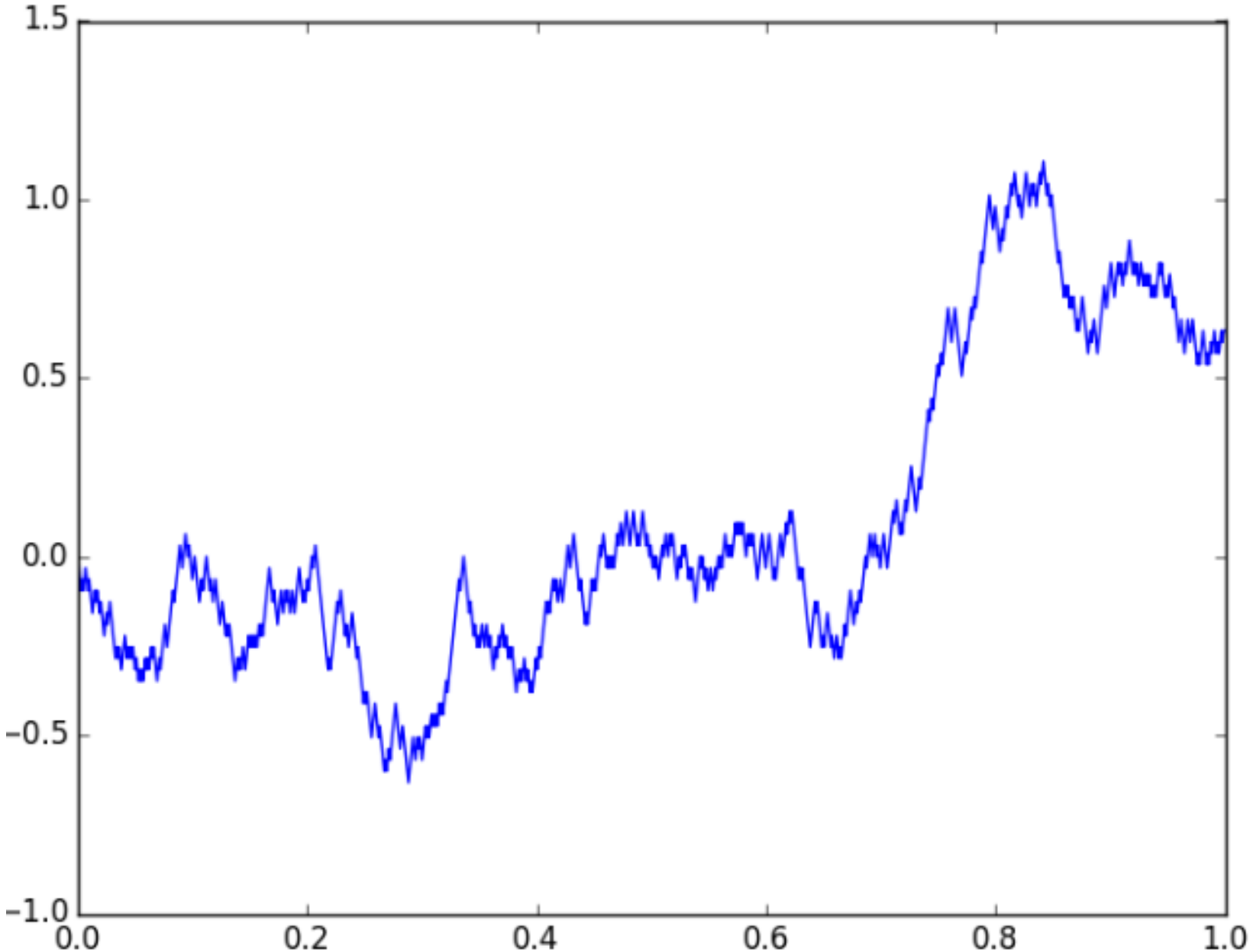
- $W_0 = 0$
- W_t has independent increments
- $W_t - W_s$ is $\mathbf{N}(0, t-s)$, a gaussian random variable with mean 0 and variance $t - s$.

What is a Brownian motion?

A stochastic process $t \longmapsto W_t$ is a Brownian motion if:

- $W_0 = 0$
- W_t has independent increments
- $W_t - W_s$ is $\mathbf{N}(0, t-s)$, a gaussian random variable with mean 0 and variance $t - s$.

What is a Brownian motion?



How to generate one Brownian motion?

We need:

- a time horizon, for example 1 year;
- a number of time steps, for example 10;
- a number of simulations, for example 100;
- a discretization of the Brownian motion:

$$W_{t+\Delta t} = W_t + \sigma \cdot \sqrt{\Delta t} \cdot Z,$$

where Z is $\mathbf{N}(0,1)$ and $\sigma > 0$ is a volatility factor.

How to generate one Brownian motion?

We need:

- a time horizon, for example 1 year;
- a number of time steps, for example 10;
- a number of simulations, for example 100;
- a discretization of the Brownian motion:

$$W_{t+\Delta t} = W_t + \sigma \cdot \sqrt{\Delta t} \cdot Z,$$

where Z is $\mathbf{N}(0,1)$ and $\sigma > 0$ is a volatility factor.

How to generate one Brownian motion?

We need:

- a time horizon, for example 1 year;
- a number of time steps, for example 10;
- a number of simulations, for example 100;
- a discretization of the Brownian motion:

$$W_{t+\Delta t} = W_t + \sigma \cdot \sqrt{\Delta t} \cdot Z,$$

where Z is $\mathbf{N}(0,1)$ and $\sigma > 0$ is a volatility factor.

How to generate one Brownian motion?

We need:

- a time horizon, for example 1 year;
- a number of time steps, for example 10;
- a number of simulations, for example 100;
- a discretization of the Brownian motion:

$$W_{t+\Delta t} = W_t + \sigma \cdot \sqrt{\Delta t} \cdot Z,$$

where Z is $\mathbf{N}(0,1)$ and $\sigma > 0$ is a volatility factor.

How to generate one Brownian motion?

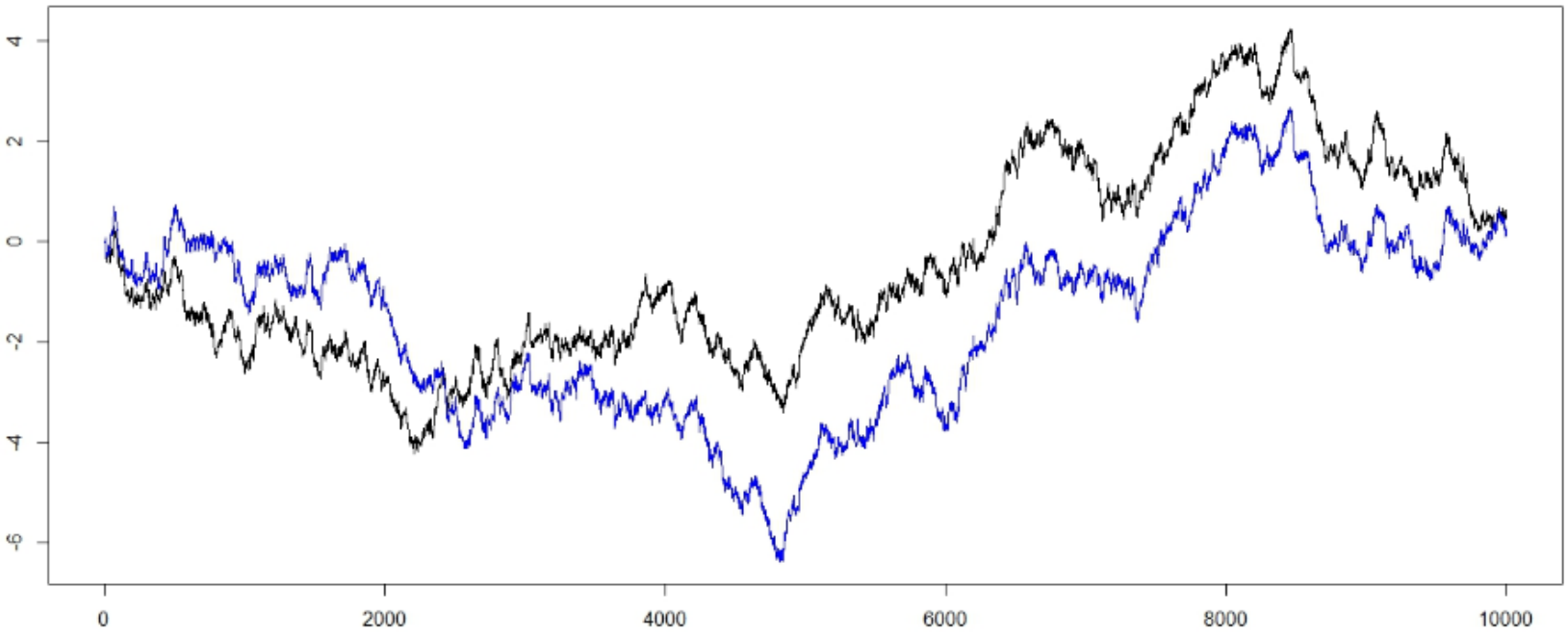
We need:

- a time horizon, for example 1 year;
- a number of time steps, for example 10;
- a number of simulations, for example 100;
- a discretization of the Brownian motion:

$$W_{t+\Delta t} = W_t + \sigma \cdot \sqrt{\Delta t} \cdot Z,$$

where Z is $\mathbf{N}(0, 1)$ and $\sigma > 0$ is a volatility factor.

Two correlated Brownian motions - 1



Two correlated Brownian motions - 1

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$

$$W^2_{t+\Delta t} = W^2_t + \sigma_2 \cdot \sqrt{\Delta t} \cdot Z_2$$

where

$$Z_2 = \rho \cdot Z_1 + \sqrt{1-\rho^2} \cdot X_1$$

and Z_1, X_1 are independent $\mathbf{N}(0,1)$, $\rho \in [-1,1]$.

Two correlated Brownian motions - 1

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$

$$W^2_{t+\Delta t} = W^2_t + \sigma_2 \cdot \sqrt{\Delta t} \cdot Z_2$$

where

$$Z_2 = \rho \cdot Z_1 + \sqrt{1-\rho^2} \cdot X_1$$

and Z_1, X_1 are independent $\mathbf{N}(0,1)$, $\rho \in [-1,1]$.

Two correlated Brownian motions - 1

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$

$$W^2_{t+\Delta t} = W^2_t + \sigma_2 \cdot \sqrt{\Delta t} \cdot Z_2$$

where

$$Z_2 = \rho \cdot Z_1 + \sqrt{1-\rho^2} \cdot X_1$$

and Z_1, X_1 are independent $\mathbf{N}(0,1)$, $\rho \in [-1,1]$.

Two correlated Brownian motions - 1

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$

$$W^2_{t+\Delta t} = W^2_t + \sigma_2 \cdot \sqrt{\Delta t} \cdot Z_2$$

where

$$Z_2 = \rho \cdot Z_1 + \sqrt{1-\rho^2} \cdot X_1$$

and Z_1, X_1 are independent $\mathbf{N}(0,1)$, $\rho \in [-1,1]$.

Two correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix}}_{C} \cdot \begin{bmatrix} Z_1 \\ X_1 \end{bmatrix}$$

Observe:

$$C \cdot C^T = \underbrace{\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}}_{Cov}$$

$$= \begin{bmatrix} Var(Z_1) & Cov(Z_1, Z_2) \\ Cov(Z_1, Z_2) & Var(Z_2) \end{bmatrix}$$

Cov is the Variance-Covariance matrix.

Two correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix}}_{C} \cdot \begin{bmatrix} Z_1 \\ X_1 \end{bmatrix}$$

Observe:

$$C \cdot C^T = \underbrace{\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}}_{Cov}$$

$$= \begin{bmatrix} Var(Z_1) & Cov(Z_1, Z_2) \\ Cov(Z_1, Z_2) & Var(Z_2) \end{bmatrix}$$

Cov is the Variance-Covariance matrix.

Two correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix}}_{C} \cdot \begin{bmatrix} Z_1 \\ X_1 \end{bmatrix}$$

Observe:

$$C \cdot C^T = \underbrace{\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}}_{Cov}$$

$$= \begin{bmatrix} Var(Z_1) & Cov(Z_1, Z_2) \\ Cov(Z_1, Z_2) & Var(Z_2) \end{bmatrix}$$

Cov is the Variance-Covariance matrix.

Two correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix}}_{C} \cdot \begin{bmatrix} Z_1 \\ X_1 \end{bmatrix}$$

Observe:

$$C \cdot C^T = \underbrace{\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}}_{Cov}$$

$$= \begin{bmatrix} Var(Z_1) & Cov(Z_1, Z_2) \\ Cov(Z_1, Z_2) & Var(Z_2) \end{bmatrix}$$

Cov is the Variance-Covariance matrix.

Many correlated Brownian motions - 1

We have $Z_1, Z_2, \ldots, Z_N$ all correlated to each other.

How do we generate N correlated random variables?

$\Rightarrow$ Cholesky decomposition

$$C \cdot C^T = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1N} \\ \rho_{12} & 1 & & \vdots \\ \vdots & \vdots & & \vdots \\ \rho_{1N} & \rho_{2N} & \cdots & 1 \end{bmatrix}$$

where C is a lower triangular matrix.

Many correlated Brownian motions - 1

We have $Z_1, Z_2, \ldots, Z_N$ all correlated to each other.

How do we generate N correlated random variables?

$\Rightarrow$ Cholesky decomposition

$$C \cdot C^T = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1N} \\ \rho_{12} & 1 & & \vdots \\ \vdots & \vdots & & \vdots \\ \rho_{1N} & \rho_{2N} & \cdots & 1 \end{bmatrix}$$

where C is a lower triangular matrix.

Many correlated Brownian motions - 1

We have $Z_1, Z_2, \ldots, Z_N$ all correlated to each other.

How do we generate N correlated random variables?

$\Rightarrow$ Cholesky decomposition

$$C \cdot C^T = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1N} \\ \rho_{12} & 1 & & \vdots \\ \vdots & \vdots & & \vdots \\ \rho_{1N} & \rho_{2N} & \cdots & 1 \end{bmatrix}$$

where C is a lower triangular matrix.

Many correlated Brownian motions - 1

We have $Z_1, Z_2, \ldots, Z_N$ all correlated to each other.

How do we generate N correlated random variables?

$\Rightarrow$ Cholesky decomposition

$$C \cdot C^T = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1N} \\ \rho_{12} & 1 & & \vdots \\ \vdots & \vdots & & \vdots \\ \rho_{1N} & \rho_{2N} & \cdots & 1 \end{bmatrix}$$

where C is a lower triangular matrix.

Many correlated Brownian motions - 1

We have $Z_1, Z_2, \ldots, Z_N$ all correlated to each other.

How do we generate N correlated random variables?

$\Rightarrow$ Cholesky decomposition

$$C \cdot C^T = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1N} \\ \rho_{12} & 1 & & \vdots \\ \vdots & \vdots & & \vdots \\ \rho_{1N} & \rho_{2N} & \cdots & 1 \end{bmatrix}$$

where C is a lower triangular matrix.

Many correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_N \end{bmatrix} = C \cdot \begin{bmatrix} Z_1 \\ X_1 \\ \vdots \\ X_{N-1} \end{bmatrix}$$

and $Z_1, X_1, \ldots, X_{N-1}$ are independent $\mathbf{N}(0,1)$.

Eventually:

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$

$$\vdots$$

$$W^N_{t+\Delta t} = W^N_t + \sigma_N \cdot \sqrt{\Delta t} \cdot Z_N$$

Many correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_N \end{bmatrix} = C \cdot \begin{bmatrix} Z_1 \\ X_1 \\ \vdots \\ X_{N-1} \end{bmatrix}$$

and $Z_1, X_1, \ldots, X_{N-1}$ are independent $\mathbf{N}(0,1)$.

Eventually:

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$

$$\vdots$$

$$W^N_{t+\Delta t} = W^N_t + \sigma_N \cdot \sqrt{\Delta t} \cdot Z_N$$

Many correlated Brownian motions - 2

In matrix notation:

$$\begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_N \end{bmatrix} = C \cdot \begin{bmatrix} Z_1 \\ X_1 \\ \vdots \\ X_{N-1} \end{bmatrix}$$

and $Z_1, X_1, \ldots, X_{N-1}$ are independent $\mathbf{N}(0,1)$.

Eventually:

$$W^1_{t+\Delta t} = W^1_t + \sigma_1 \cdot \sqrt{\Delta t} \cdot Z_1$$
$$\vdots$$
$$W^N_{t+\Delta t} = W^N_t + \sigma_N \cdot \sqrt{\Delta t} \cdot Z_N$$

References

Next steps...

- **Think Python: How to Think Like a Computer Scientist** by *Allen B. Downey* B
- **Python Crash Course: A Hands-on, Project-Based Introduction to Programming** by *Eric Matthes* B
- **Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython** by *Wes McKinney* I
- **Python Cookbook** by *David Beazley and Brian K. Jones* A
- **Learning Python** by *Mark Lutz* A