~~Chaos~~
Random

LAWFUL GOOD

Many see him as a naive boyscout whipped by his own selflessness. They will not, cannot, see him for what he is, a hero.

NEUTRAL GOOD

A good tonic, altruism. Nothing helps one put problems in perspective like allegiance to a higher cause.

CHAOTIC GOOD

No rest. No mercy. No matter what.

LAWFUL NEUTRAL

I am the state.

TRUE NEUTRAL

I don't expect you to understand. It's how we should exist. How we were meant to exist.

CHAOTIC NEUTRAL

I'm not gonna split hairs and I'm not gonna fight my teammates. I mean, unless it involves Terry's clothes coming off and mud or chocolate pudding or something like that.

LAWFUL EVIL

Slaves would be tyrants, were the chance theirs.

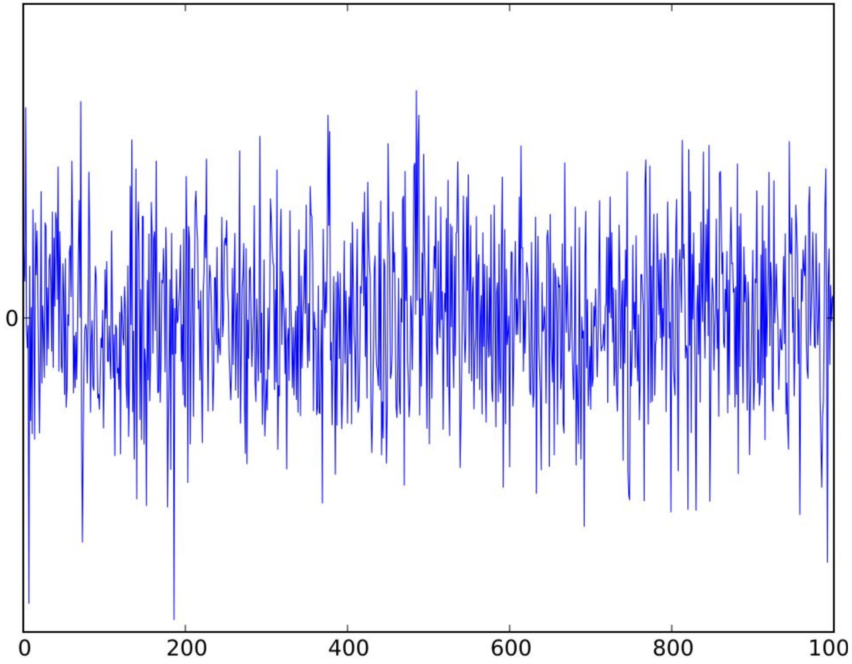
NEUTRAL EVIL

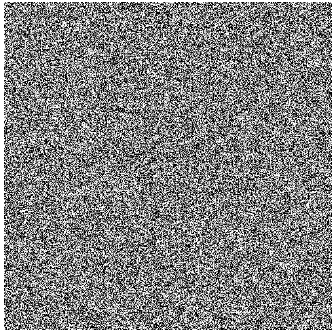
Do you really think fate turned us into gods so we could refuse these gifts?

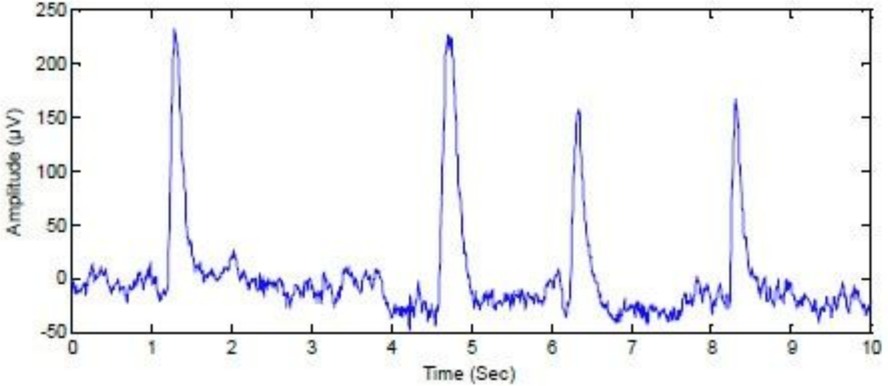
CHAOTIC EVIL

All it takes is one bad day to reduce the sanest man alive to lunacy. Thats how far the world is from where I am. Just one bad day.

Random examples







Random

Not very interesting, because ... wait for the next slide)

```
import random
```

```
random.randint(0, 10)
```

```
1
```

Random in np

Better cause can produce large random arrays

```
import numpy as np
```

```
# Uniformly distributed
# At least 1 argument - ceiling
np.random.randint(5)
```

```
2
```

```python
# From to
np.random.randint(-5, 5)
```

```
-4
```

```python
# Several values
np.random.randint(-5, 6, 10)
```

```
array([ 0,  3,  5,  5,  4, -2, -4,  2,  2,  5])
```

```python
# Uniformly distributed floats
np.random.random()
```

```
0.943104668696257
```

Different distributions

```
# Normal
np.random.randn(2, 3)

array([[-0.30303934,  0.39839465, -0.48802327],
       [ 0.79074641,  2.01141451,  0.92467618]])


# Have 2 distribution parameters
np.random.beta(1, 2, (2, 3))

array([[0.04100369, 0.45135484, 0.61265842],
       [0.89379879, 0.67421185, 0.80170757]])
```

```python
# Again 2 parameters
np.random.binomial(10, 0.3)
```

```
4
```

```python
# 1 parameter
np.random.pareto(3)
```

```
0.7120484442254971
```

```python
# Normal distribution with arbitrary mean and variance, which are defaulted to 0 and 1
np.random.normal(3, 0.1, 3)
```

```
array([3.07348849, 2.88520331, 3.16730714])
```

```python
# df parameter
np.random.chisquare(5)
```

```
3.391598378778647
```

```python
# lambda parameter
np.random.poisson(3)
```

```
5
```

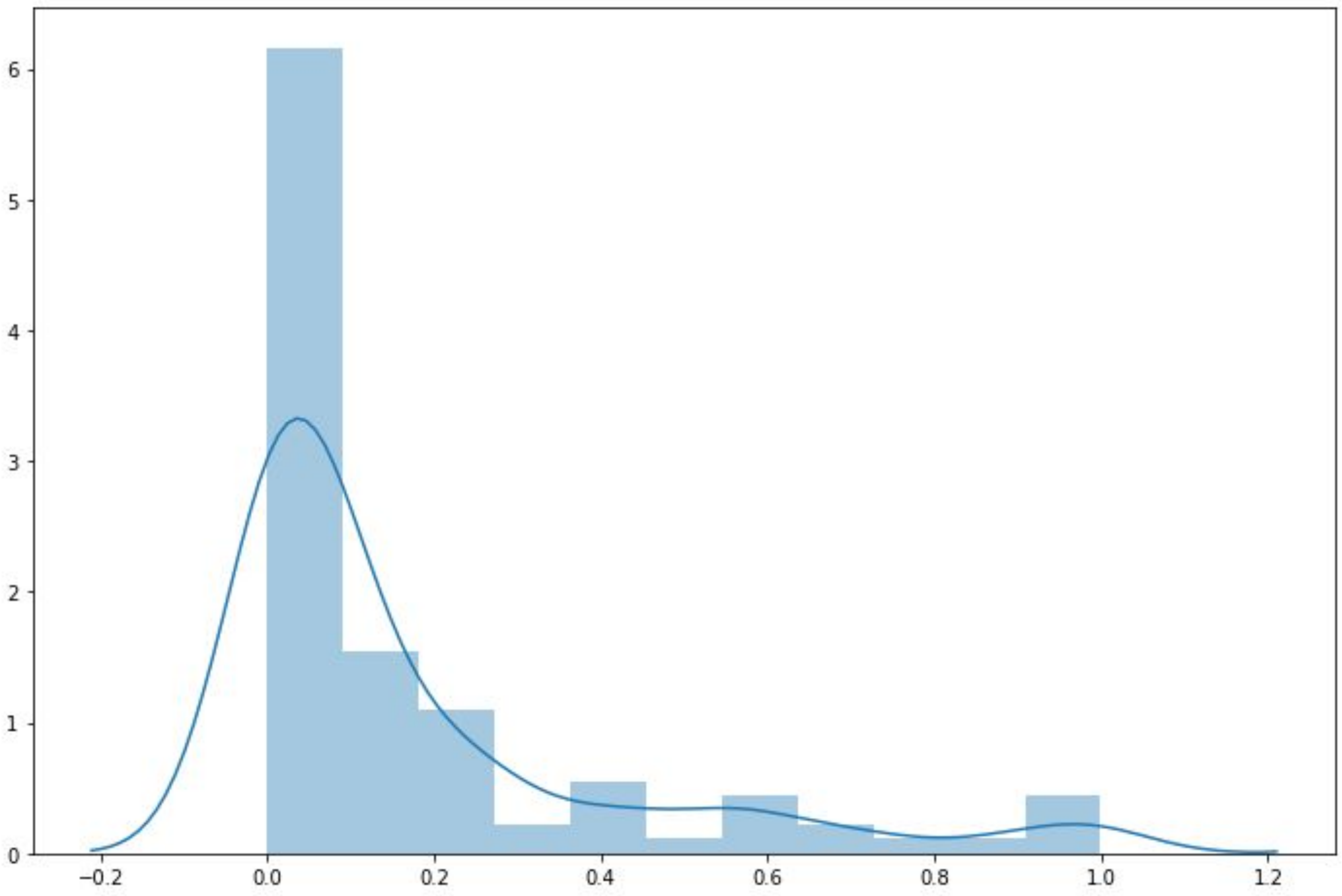
```python
# Another parameter
np.random.power(3)
```

```
0.5101895579233388
```

Distribution illustration

```
import matplotlib.pyplot as plt
import seaborn as sns


plt.figure(figsize=(12, 8))
sns.distplot(np.random.power(0.3, 100))
```


0
1
2
3
4
5
6
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2

Collections functions

Really handy when you need them

```
numbers = [0, 1, 2, 4, 5, 6, 8, 10, 11]

# Change the order of the elements
np.random.shuffle(numbers)

numbers

[2, 0, 5, 6, 8, 11, 10, 1, 4]
```

```python
numbers
```

```
[2, 0, 5, 6, 8, 11, 10, 1, 4]
```

```python
# Don't change original array, return copy
np.random.permutation(numbers)
```

```
array([10,  8, 11,  0,  5,  4,  6,  2,  1])
```

```python
numbers
```

```
[2, 0, 5, 6, 8, 11, 10, 1, 4]
```

```python
# Sample from the collection
np.random.choice(numbers)
```

```
6
```

```python
# Many elements
np.random.choice(numbers, 5)
```

```
array([ 6,  5,  6,  8, 11])
```

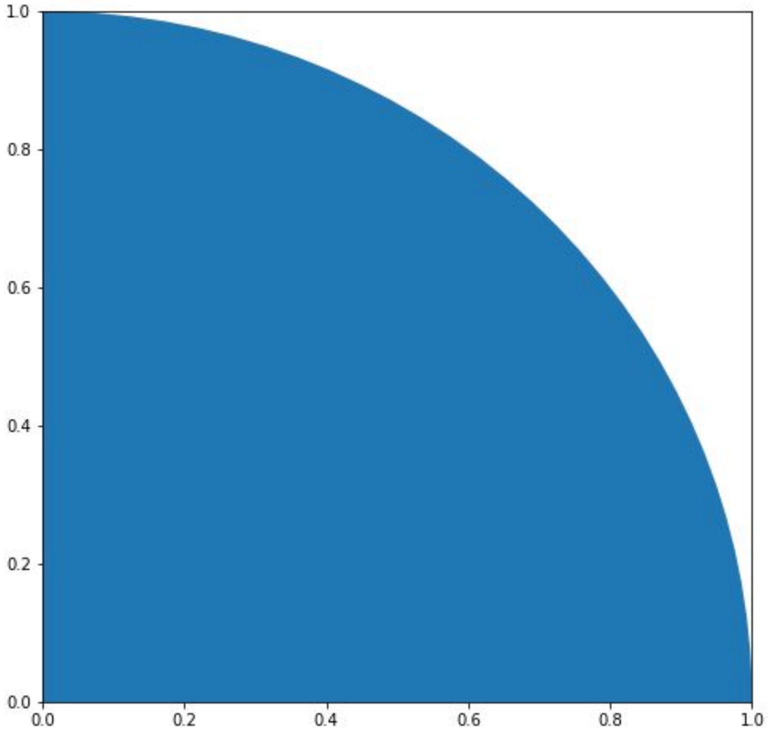
```python
# Use each element once
np.random.choice(numbers, 5, replace=False)
```

```
array([ 2,  0, 10,  1,  8])
```

```python
# Use custom probability of rolling for each element
np.random.choice(numbers,
                 5,
                 replace=False,
                 p=[0.1316098 , 0.07381153,
           0.03022185, 0.19347025, 0.14968923,
           0.12112077, 0.17881221,
           0.10049474, 0.02076963])

array([10,  8,  1,  5,  4])
```

Random simulation at the π calculation service

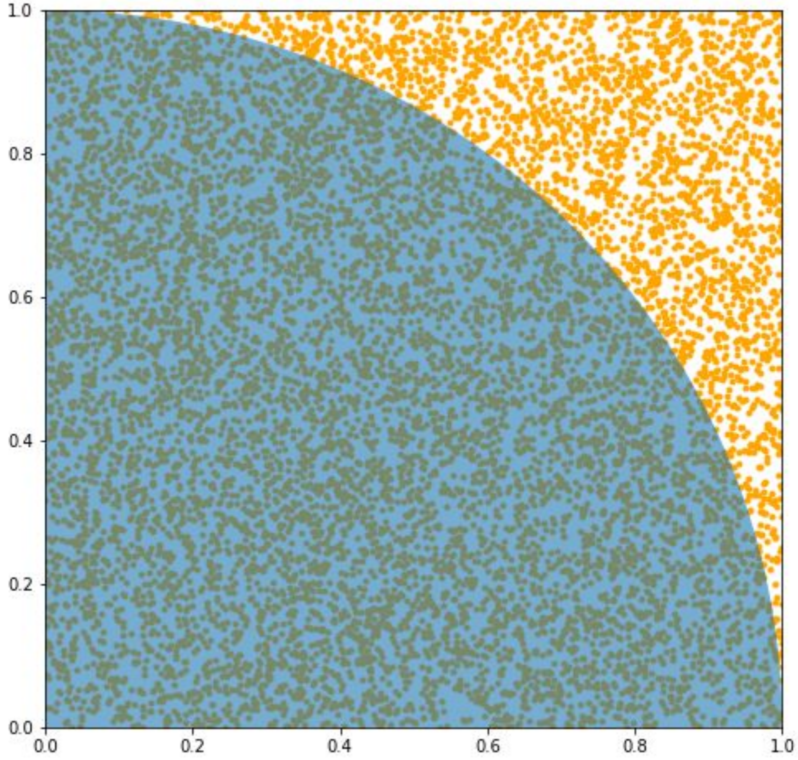


$$S = \frac{\pi \cdot r^2}{4}$$

$$\pi = \frac{4 \cdot S}{r^2}$$

$$S = \frac{points\_in\_circle}{all\_points}$$

Calculations

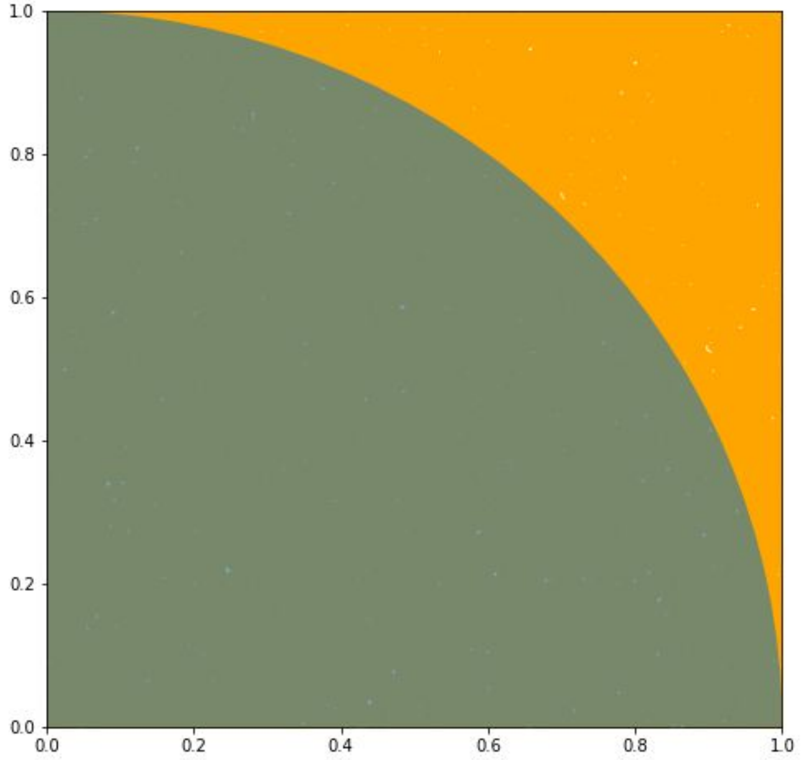


r = 1
number of all points = 10000
number of points inside circle = 7840

S = 0.784

$$\pi = \frac{4 \cdot 0.784}{1^2}$$
$$\pi = 3.136$$

Higher accuracy



r = 1
number of all points = 100000
number of points inside circle = 78532

S = 0.78532

$$\pi = \frac{4 \cdot 0.78532}{1^2}$$
$$\pi = 3.14128$$