Simulating Data with rvpy

Tim Book

3/26/2019

What is it?

`rvpy` is a wrapper for `scipy.stats` that has two goals:

- To be a fun "sandbox" for beginner data scientists (aka statisticians) to learn the algebra behind random variables.
- To be a useful tool for advanced users to easily simulate data for a variety of tasks.

Getting started

It's readily available on PyPI and requires at least Python version 3.6. Install with:

```
pip install rvpy
```

Source can be found at:

https://www.github.com/timbook/rvpy

Usage: Algebra

```
>>> import rvpy
>>> X = rvpy.Normal(3, 5); X
Normal(mu=3, sigma=5)
```

Usage: Algebra

```
>>> import rvpy
>>> X = rvpy.Normal(3, 5); X
Normal(mu=3, sigma=5)
```

```
>>> X - 3
Normal(mu=0, sigma=5)
```

Usage: Algebra

```
>>> Z = (X - 3) / 5; Z
Normal(mu=0.0, sigma=1.0)
```

Usage: Algebra

```
>>> Z = (X - 3) / 5; Z
Normal(mu=0.0, sigma=1.0)
```

```
>>> Z**2
ChiSq(df=1)
```

Usage: Algebra

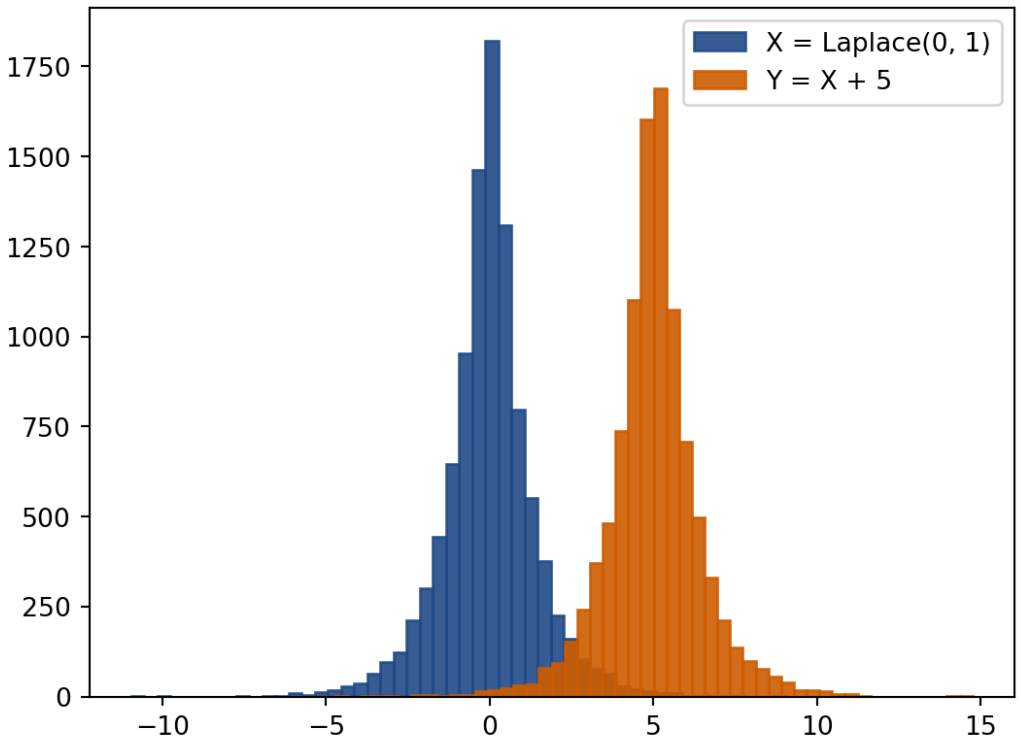
```
>>> Z = (X - 3) / 5; Z
Normal(mu=0.0, sigma=1.0)
```

```
>>> Z**2
ChiSq(df=1)
```

```
>>> (Z**2).sample(2, 3)
array([[5.60860865e-01, 1.33413424e-01, 5.89250764e-01],
       [1.37286266e-05, 1.87912991e-02, 4.03546592e-01]])
```

Usage: Teaching

```
X = rvpy.Laplace(0, 1)
Y = X + 5
```



What I've Got So Far

I've already included all the "named" distributions, and a few of their offshoots:

- Normal
- Standard Normal
- LogNormal
- Cont. Uniform
- Gamma
- Exponential
- χ^2
- Beta
- T
- F
- Cauchy
- Standard Cauchy
- Laplace
- Weibull
- Rayleigh
- Pareto
- Logistic
- LogLogistic
- Gompertz
- Gumbel
- Bernoulli
- Binomial
- Poisson
- Discrete Uniform
- Neg. Binomial
- Geometric
- Hypergeometric
- Degenerate

Desires

These are things I'd like to add, if there is any desire:

- **Multivariate Distributions**
  - Multivariate normal, Multinomial, Dirichlet, etc.
- **Conditional Distributions**
  - For example, for $X, Y \sim \mathcal{N}$, then $X|Y \sim \mathcal{N}$
  - Possible implementations:
    - `X.given(mu=Y)`
    - `X | {'mu': Y}`
    - `X = rvpy.Normal(mu, Y)`

Bottleneck to True Enlightenment

Right now, rvpy does not support arbitrary transformations. Only “named” ones.

```
X = rvpy.Binomial(5, 0.3)
X + 3 # Error!
```

I want to support some kind of tree-based system for storing operations for sampling.

Bottleneck to True Enlightenment

For example, if

```
A = rvpy.Normal(3, 5)
B = rvpy.Exponential(7)
C = rvpy.Weibull(2, 3)
n = 100
```

I want to replace

```
X_sample = A.sample(n) + B.sample(n) + C.sample(n)
```

with

```
X = A + B + C
X_sample = X.sample(n)
```