Probabilistic Programming in Python with PyMC3

John Salvatier
@johnsalvatier

About me

- Author of PyMC3 rewrite
- ex-Amazonian
- Seattle Effective Altruists founder

Probabilistic Programming

- Write program that could generate your data
- Automatic inference for unknown parameters

PyMC3

- Bayesian probabilistic programming
- Short, clear models
- Simple inference
- Rewrite of PyMC2
  - Built around theano
  - Advanced samplers support bigger models

Blackbox Machine Learning

- not good at conveying how answer was provided
- assumptions are typically implicit and opaque

image by Olivier Grisel

Black Box ML

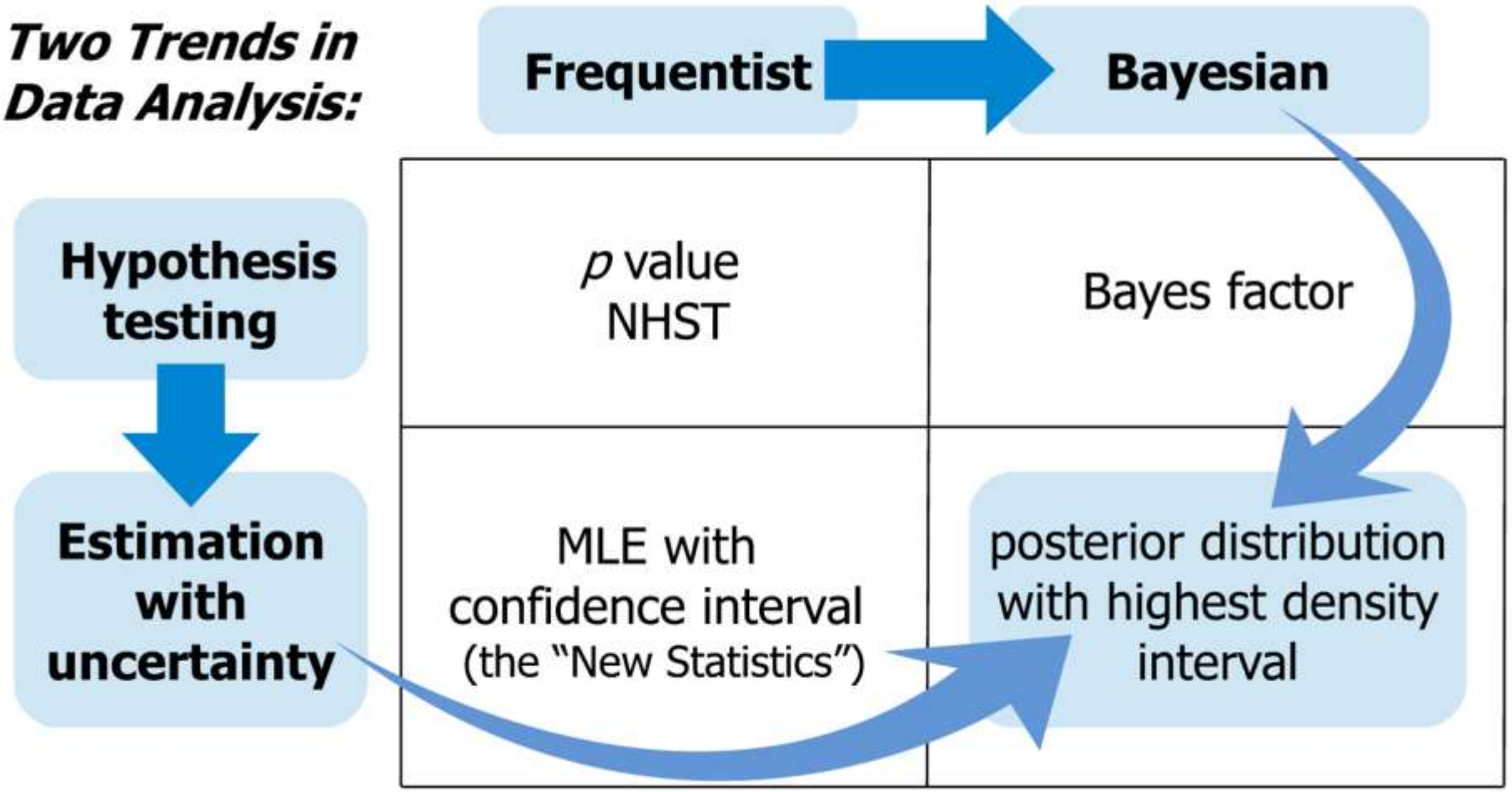
vs.

Probabilistic Programming

- Opaque inference
- Constrained models
- Uncertainty ???

- Clear inference
- Extreme flexibility
- Full uncertainty

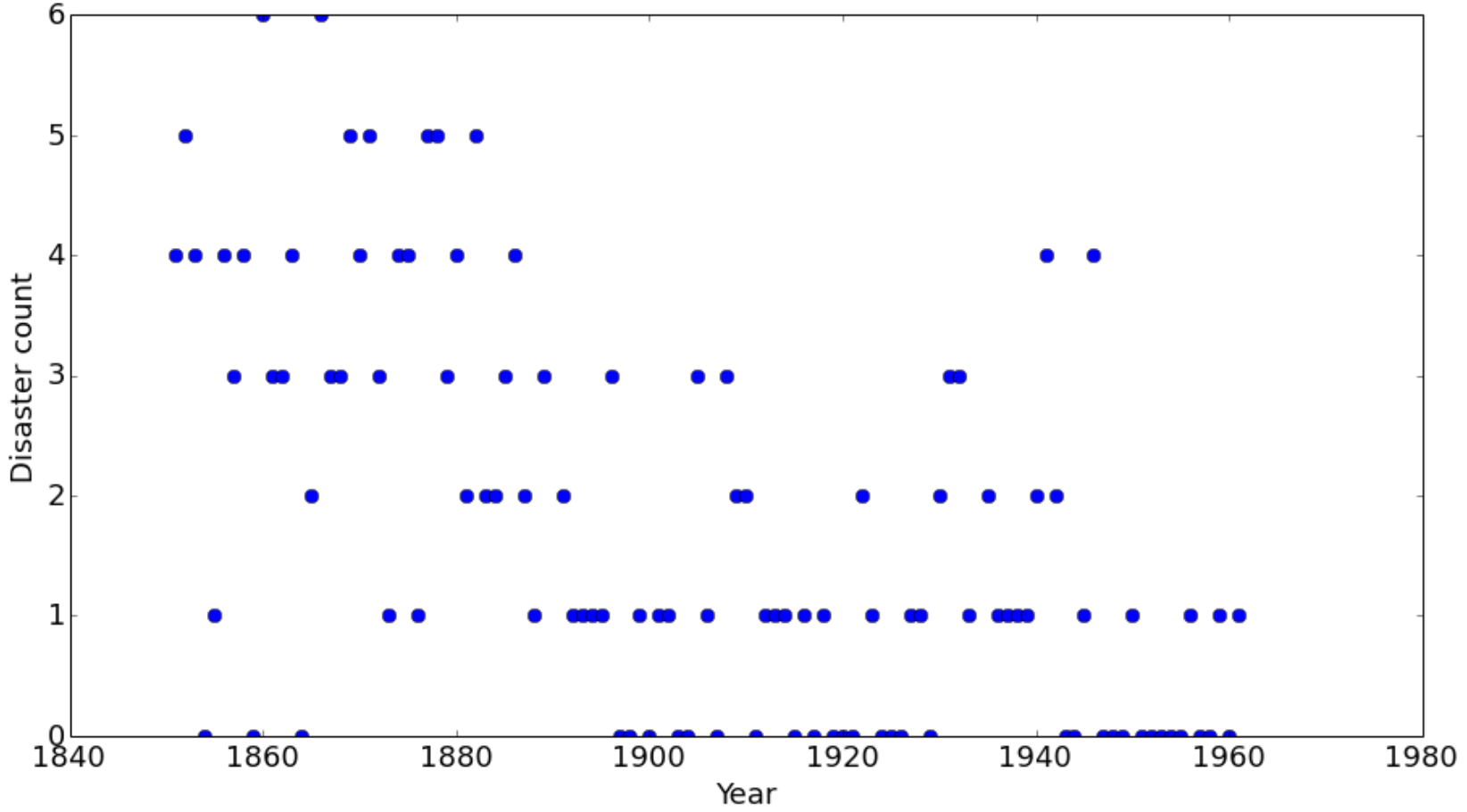

bop it!



by John Kruschke

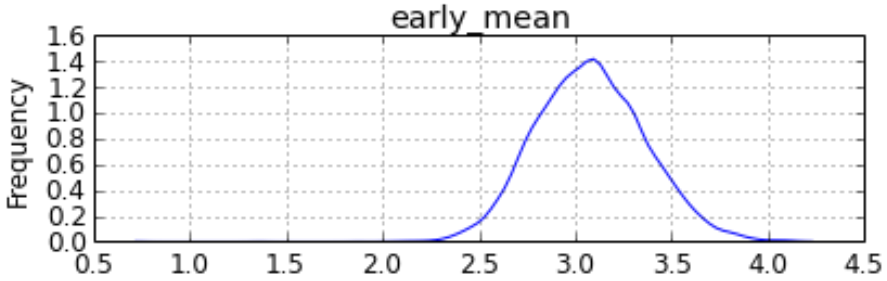
Disaster Model - Switchpoint Analysis

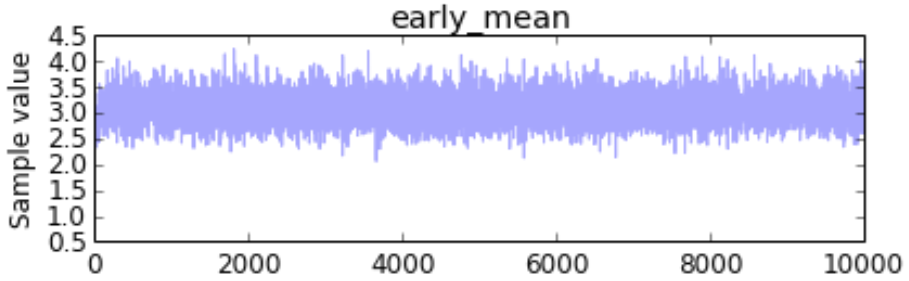
- UK Coal mining disasters 1851 - 1962

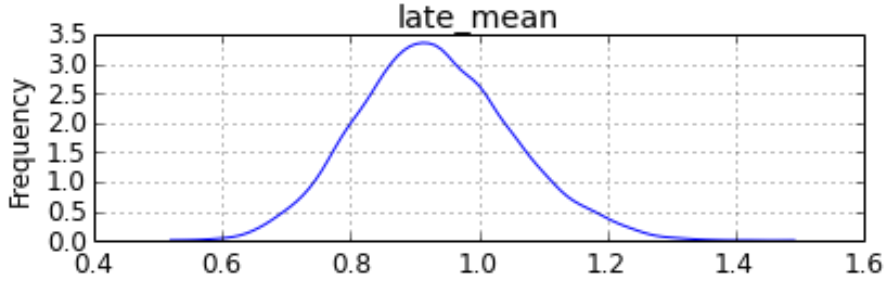


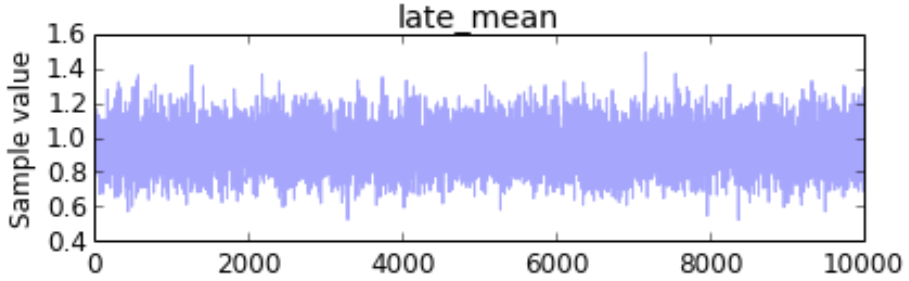
```
from pymc3 import *

year = np.arange(len(disaster_data))

with Model() as disaster_model:

    switchpoint = DiscreteUniform('switchpoint', lower=0, upper=len(year))

    early_mean = Exponential('early_mean')
    late_mean = Exponential('late_mean')

    rate = switch(switchpoint >= year, early_mean, late_mean)

    disasters = Poisson('disasters', rate, observed=disaster_data)
```

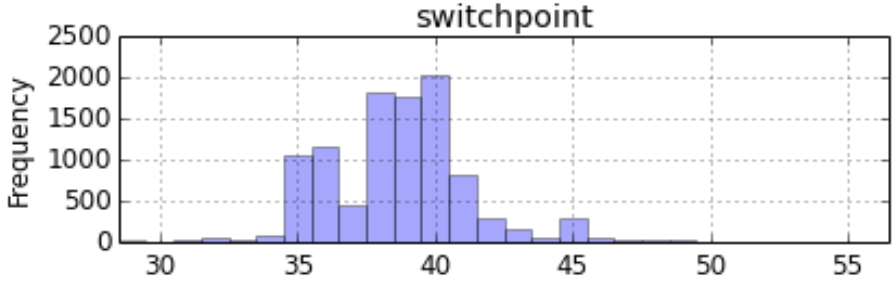
```
with disaster_model:
    trace = sample(10000, step=[Metropolis(), NUTS()])
    traceplot(trace, ['early_mean', 'late_mean', 'switchpoint'])
```

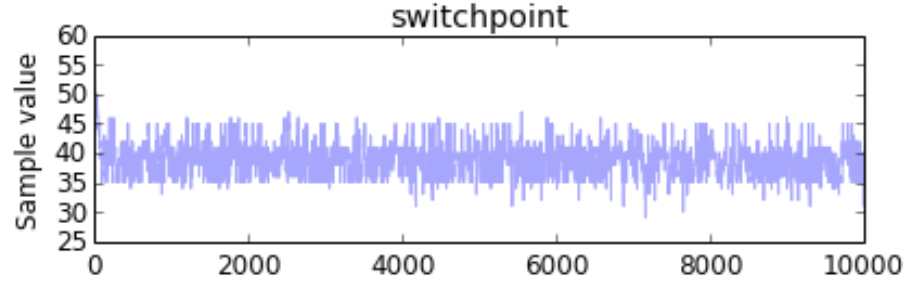






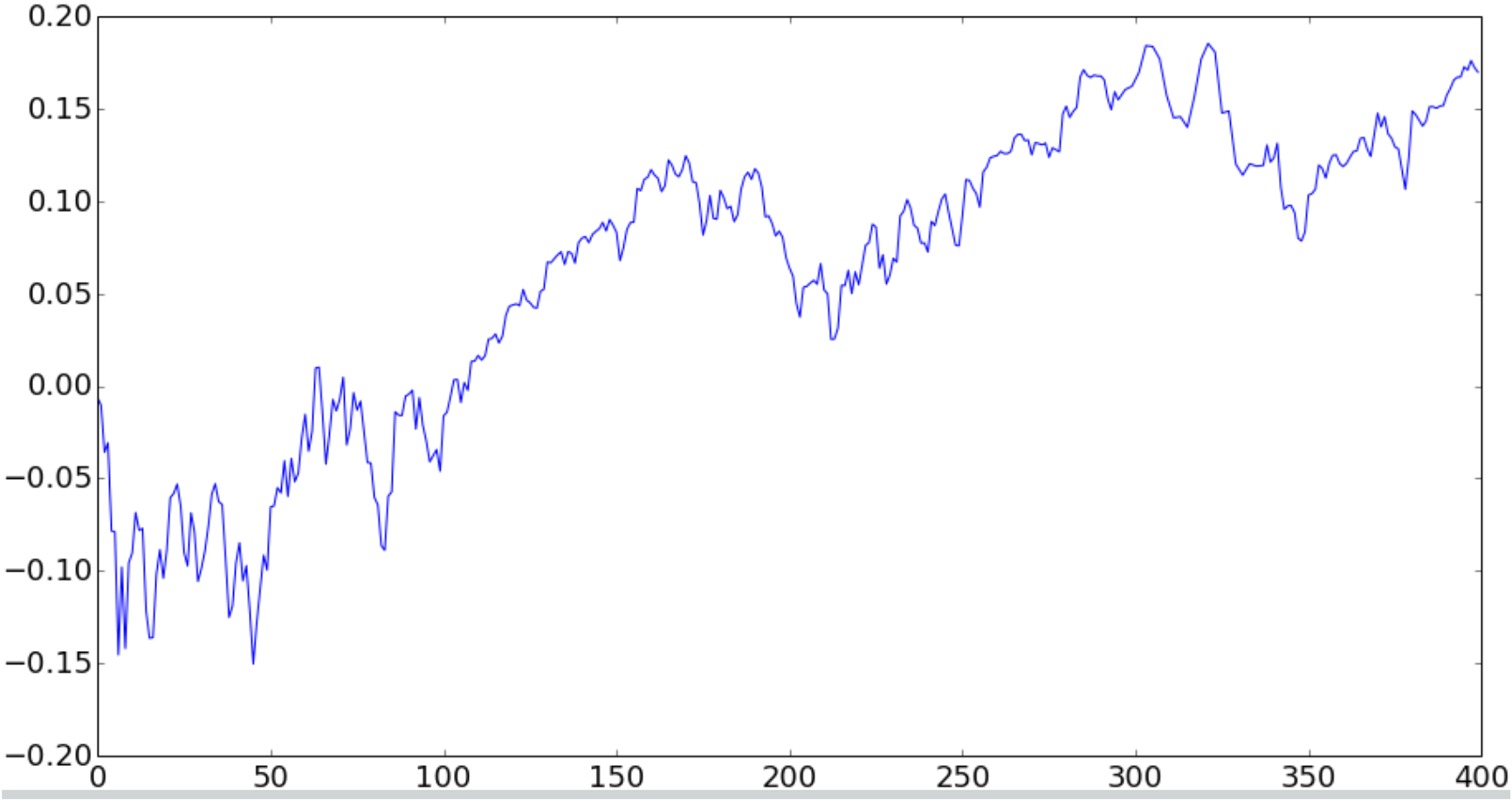




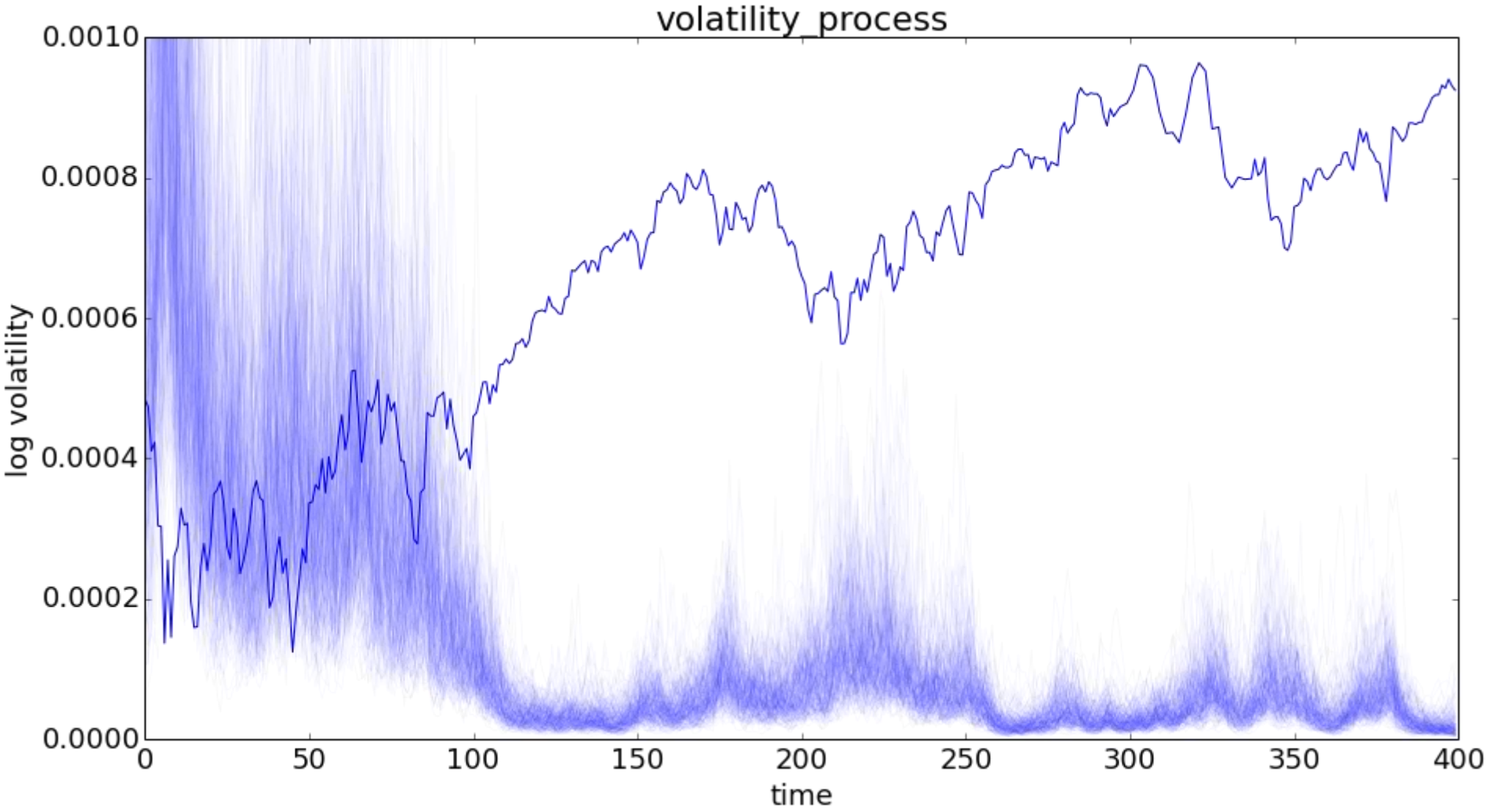


Stochastic Volatility Model

- 1 year S&P500 Data



```
from pymc3 import Exponential, T, exp, Deterministic
from pymc3.distributions.timeseries import GaussianRandomWalk

with Model() as sp500_model:

    nu = Exponential('nu', 1./10, testval=.1)

    sigma = Exponential('sigma', 1./.02, testval=.1)

    s = GaussianRandomWalk('s', sigma**-2, shape=n)

    volatility_process = Deterministic('volatility_process', exp(-2*s))

    r = T('r', nu, lam=1/volatility_process, observed=returns)
```


volatility_process
log volatility
time
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
50
100
150
200
250
300
350
400

PyMC3

- Powerful model specification syntax
  - NumPy like broadcasting and functions
- Full bayesian inference
- State of the art methods
  - Handle thousands of estimated parameters

Other Features

- Custom distributions and operators
- Handle missing values easily with Pandas nans
- Generalized Linear Models
  - glm('y ~ x1 + x2', df)
- Variational inference coming soon

Further Resources

- Twitter: @johnsalvatier
- Tutorial: http://bit.ly/1OuFqzb
- github: https://github.com/pymc-devs/pymc3

Chris Fonnesbeck - Vanderbilt
Thomas Wiecki - Quantopian