PyMC Pandas Example

This example project shows how to fit a fixed effects Poisson model with PyMC. It uses pandas Series and DataFrame objects to store data in a classy way.

In [1]:

```
import pylab as pl
import pymc as mc
import pandas
```

1. Simulate Noisy Data

In [2]:

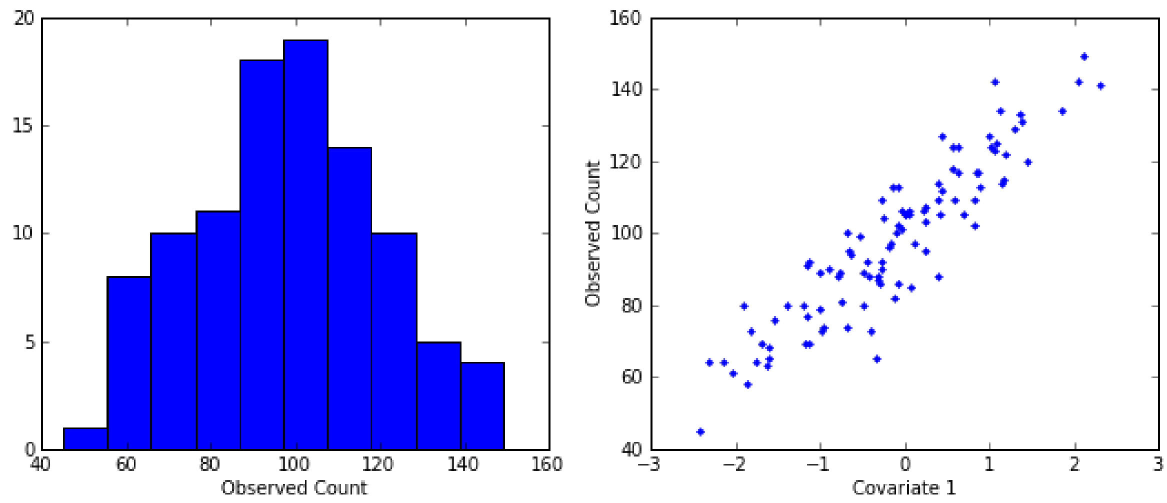
```
# simulate data with known distribution

N = 100
X = pandas.DataFrame({'constant': pl.ones(N), 'cov_1': pl.randn(N)})

beta_true = pandas.Series(dict(constant=100., cov_1=20.))
mu_true = pl.dot(X, beta_true)

Y = mc.rpoisson(mu_true)
```

In [3]:

```
# explore the data a little bit graphically

pl.figure(figsize=(11,4.25))

pl.subplot(1,2,1)
pl.hist(Y)
pl.xlabel('Observed Count')

pl.subplot(1,2,2)
pl.plot(X['cov_1'], Y, '.')
pl.xlabel('Covariate 1')
pl.ylabel('Observed Count')
```

Out[3]:

```
<matplotlib.text.Text at 0xadbbf0c>
```



2. Model data with PyMC

The following code creates a fixed effect Poisson model where the observed data stored in Y is explained by the covariate data in X, according to the formula:

$$Y_i \sim \text{Poisson}(\mu_i),$$

$$\mu_i = X_i \cdot \beta.$$

In [4]:
```
# the simplest approach doesn't work with PyMC 2.1alpha, but it does with 2.2grad
print 'pymc version:', mc.__version__

beta = mc.Uninformative('beta', value=[Y.mean(), 0.])
mu_pred = mc.Lambda('mu_pred', lambda beta=beta, X=X: pl.dot(X, beta))
Y_obs = mc.Poisson('Y_obs', mu=mu_pred, value=Y, observed=True)
```

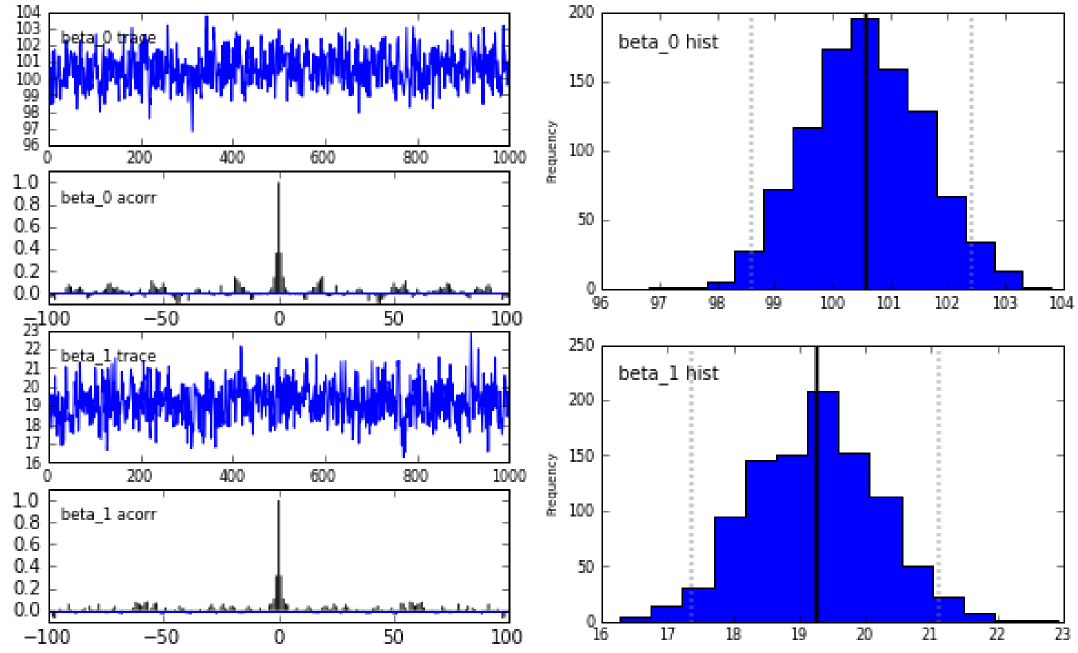
```
pymc version: 2.2grad
```

In [5]:
```
m = mc.Model([beta, mu_pred, Y_obs])
%time mc.MCMC(m).sample(10000, 5000, 5, progress_bar=False)
```

```
CPU times: user 1.65 s, sys: 0.01 s, total: 1.66 s
Wall time: 1.84 s
```

In [6]:
```
mc.Matplot.plot(beta, common_scale=False)
print '\ntrue value of beta\n', beta_true
print '\npredicted:'
print pandas.DataFrame({'mean':beta.stats()['mean'],
                        'lb':beta.stats()['95% HPD interval'][:,0],
                        'ub':beta.stats()['95% HPD interval'][:,1]},
                       columns=['mean','lb','ub'])
```

```
Plotting beta_0
Plotting beta_1

true value of beta
constant    100
cov_1       20.

predicted:
   mean   lb     ub
0  100.6  98.59  102.4
1  19.22  17.34  21.11
```



2a. TODO: Integrate PyMC and Pandas further

In [7]:

```
# making beta.value a pandas.Series would be slightly cooler than the above

@mc.stochastic
def beta(value=pandas.Series(dict(constant=Y.mean(), cov_1=0))):
    return 0.
mu_pred = mc.Lambda('mu_pred', lambda beta=beta, X=X: pl.dot(X, beta))
Y_obs = mc.Poisson('Y_obs', mu=mu_pred, value=Y, observed=True)
```

In [8]:

```
beta.value
```

Out[8]:

```
constant    98.01
cov_1       0.000
```

In [9]:

```
# unfortunately the pandas.Series becomes a numpy.array during MCMC
m = mc.Model([beta, mu_pred, Y_obs])
mc.MCMC(m).sample(10000, 5000, 5, progress_bar=False)
```

In [10]:

```
beta.value # in a pandas-centric version of PyMC, this would still be a pandas.Series
```

Out[10]:

```
array([ 100.50635223,   19.87033188])
```

In [ ]: