In [ ]:
```
# part 1
import pymc as pm; import numpy as np
n,p=100,10; X,y=np.zeros((n,p)),np.ones((n,1))

with pm.Model() as MLR:
    betas = pm.MvNormal('betas', mu=np.zeros((p,1)), cov=np.eye(p), s
    sigma = pm.TruncatedNormal('sigma', mu=1, sigma=1, lower=0) # hal
    y = pm.Normal('y', mu=pm.math.dot(X, betas), sigma=sigma, observe

with MLR:
    idata = pm.sample()
```

Homework 5: Part II

Answer the following with respect to $p(\boldsymbol{\beta}|\boldsymbol{\Sigma}, \mathbf{X}, \mathbf{y})$ on the previous slide

1. Rewrite $p(\boldsymbol{\beta}|\boldsymbol{\Sigma}, \mathbf{X}, \mathbf{y})$ in terms of σ^2 (no longer using Σ) if $\Sigma = \sigma^2 I$
2. What is $E[\boldsymbol{\beta}|\boldsymbol{\Sigma}, \mathbf{X}, \mathbf{y}]$?
3. What ***hyperparameters*** values (legal or illegal) would make $E[\boldsymbol{\beta}|\boldsymbol{\Sigma}, \mathbf{X}, \mathbf{y}] = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$?
4. What ***hyperparameters*** values (legal or illegal) would make $E[\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}|\boldsymbol{\Sigma}, \mathbf{X}, \mathbf{y}] = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$?
5. What is $\mathrm{Var}[\boldsymbol{\beta}|\boldsymbol{\Sigma}, \mathbf{X}, \mathbf{y}]$?

In [ ]:
```
# part 3
import pymc as pm
import numpy as np
from scipy import stats

p = 10  # Dimensionality
Psi = np.eye(p)  # Scale matrix
a_cov = stats.invwishart(df=p+2, scale=Psi).rvs(1)  # Inverse-Wishart

n = 1000  # Number of data points
y = stats.multivariate_normal(mean=np.zeros(p), cov=a_cov).rvs(size=r
```

```python
# Using PyMC3 to define the model
with pm.Model() as MNV_LKJ:
    # Cholesky factor of the covariance matrix
    packed_L = pm.LKJCholeskyCov("packed_L", n=p, eta=2.0,
                                 sd_dist=pm.Exponential.dist(1.0, sha
    L = pm.expand_packed_triangular(p, packed_L)
    Sigma = pm.Deterministic('Sigma', L.dot(L.T))

    # Define the prior for the mean vector
    mu = pm.MvNormal('mu', mu=np.zeros(p), cov=np.eye(p)*1e-6, shape=

    # The observed data likelihood
    y_obs = pm.MvNormal('y_obs', mu=mu, chol=L, observed=y)

# Sampling
with MNV_LKJ:
    trace = pm.sample(100)

# The trace object now contains the samples for the posterior distrib
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (4 chains in 4 jobs)
NUTS: [packed_L, mu]
```

23.66% [1893/8000 03:14<10:26

Sampling 4 chains, 0 divergences]

In [ ]: