Lecture 20

Model Specification and glms

REMEMBER LECTURE 9 BAYESIAN REGRESSION

Bayesian Formulation of Regression

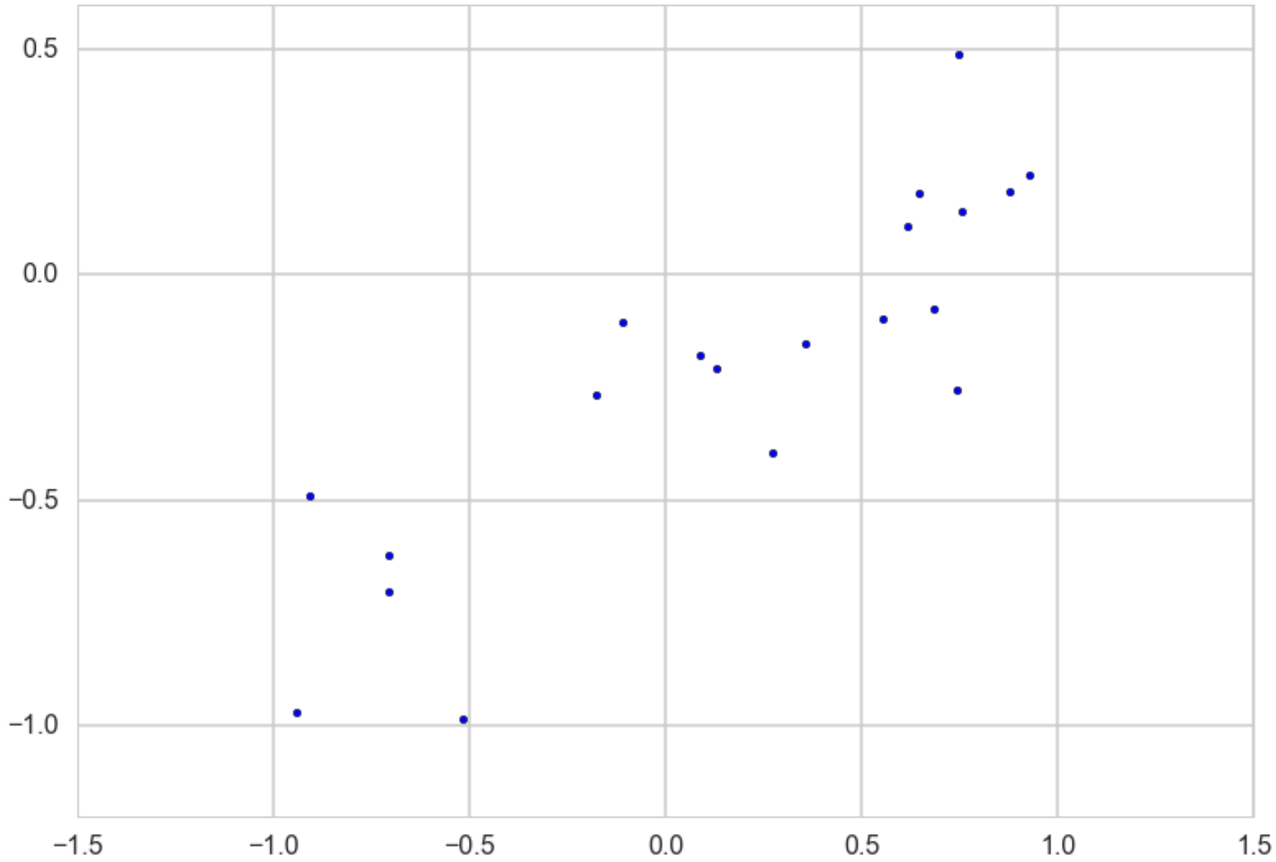
Data
$D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$

All data points are combined into a $D \times n$ matrix X.

Model:

$$y = \mathbf{x}^{\mathbf{T}}\mathbf{w} + \epsilon$$

$$\epsilon \sim N(0, \sigma_n^2)$$



Likelihood

The likelihood is, because we assume independency, the product

$$\mathcal{L} = p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{p}(\mathbf{y_i}|\mathbf{X_i}, \mathbf{w}) = \prod_{\mathbf{i=1}}^{\mathbf{n}} \frac{\mathbf{1}}{\sqrt{\mathbf{2}\pi}\sigma_\mathbf{n}} \exp\left(-\frac{(\mathbf{y_i} - \mathbf{X_i^T w})^\mathbf{2}}{\mathbf{2}\sigma_\mathbf{n}^\mathbf{2}}\right)$$

$$\propto \exp\left(-\frac{|\mathbf{y} - \mathbf{X^T w}|^\mathbf{2}}{2\sigma_n^2}\right) \propto N(X^T\mathbf{w}, \sigma_\mathbf{n}^\mathbf{2}\mathbf{I})$$

Now, suppose we have prior: $\mathbf{w} \sim \mathbf{N}(\mathbf{w_0}, \tau^\mathbf{2}\mathbf{I})$

Then, Posterior:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}) \propto p(\mathbf{y}|\mathbf{X}, \mathbf{w})\, \mathbf{p(w)}$$
$$\propto \exp\left(-\frac{1}{2\sigma_n^2}(\mathbf{y} - \mathbf{X^T w})^\mathbf{T}(\mathbf{y} - \mathbf{X^T w})\right) \exp\left(-\frac{1}{2}\mathbf{w^T \Sigma^{-1} w}\right)$$

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}) \propto \exp\left(-\frac{\mathbf{1}}{\mathbf{2}}(\mathbf{w} - \bar{\mathbf{w}})^\mathbf{T}(\frac{\mathbf{1}}{\sigma_\mathbf{n}^\mathbf{2}}\mathbf{X X^T} + \mathbf{\Sigma^{-1}})(\mathbf{w} - \bar{\mathbf{w}})\right)$$

Inverse covariance $A = \sigma_n^{-2} X X^T + \Sigma^{-1}$

where the new mean is $\bar{\mathbf{w}} = A^{-1}\Sigma^{-1}\mathbf{w_0} + \sigma_n^{-2}(A^{-1}X^T\mathbf{y})$

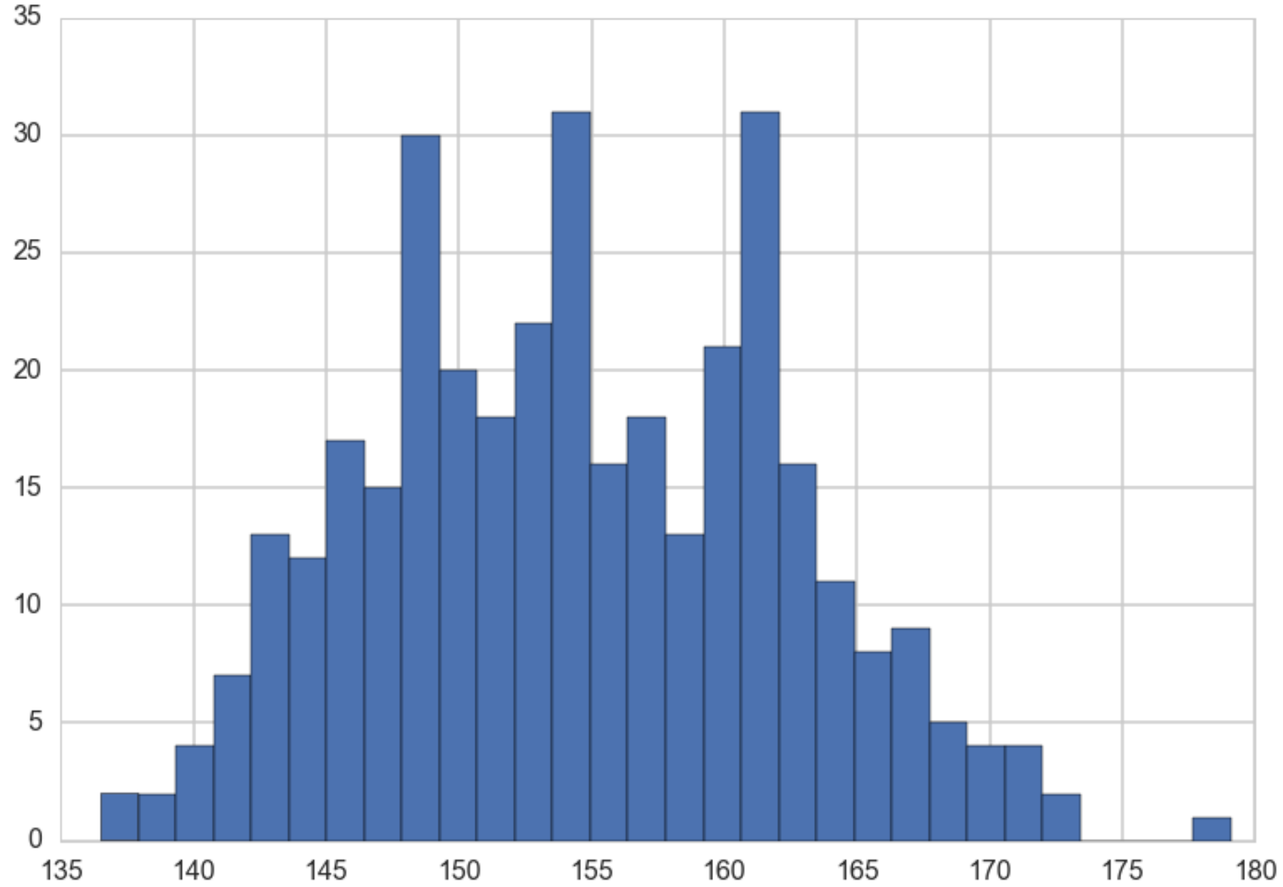
Posterior Predictive

$$p(y^*|x^*, \mathbf{x}, \mathbf{y}) =$$

$$\int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) p(\mathbf{w}|\mathbf{X}, \mathbf{y}) \mathbf{dw}$$

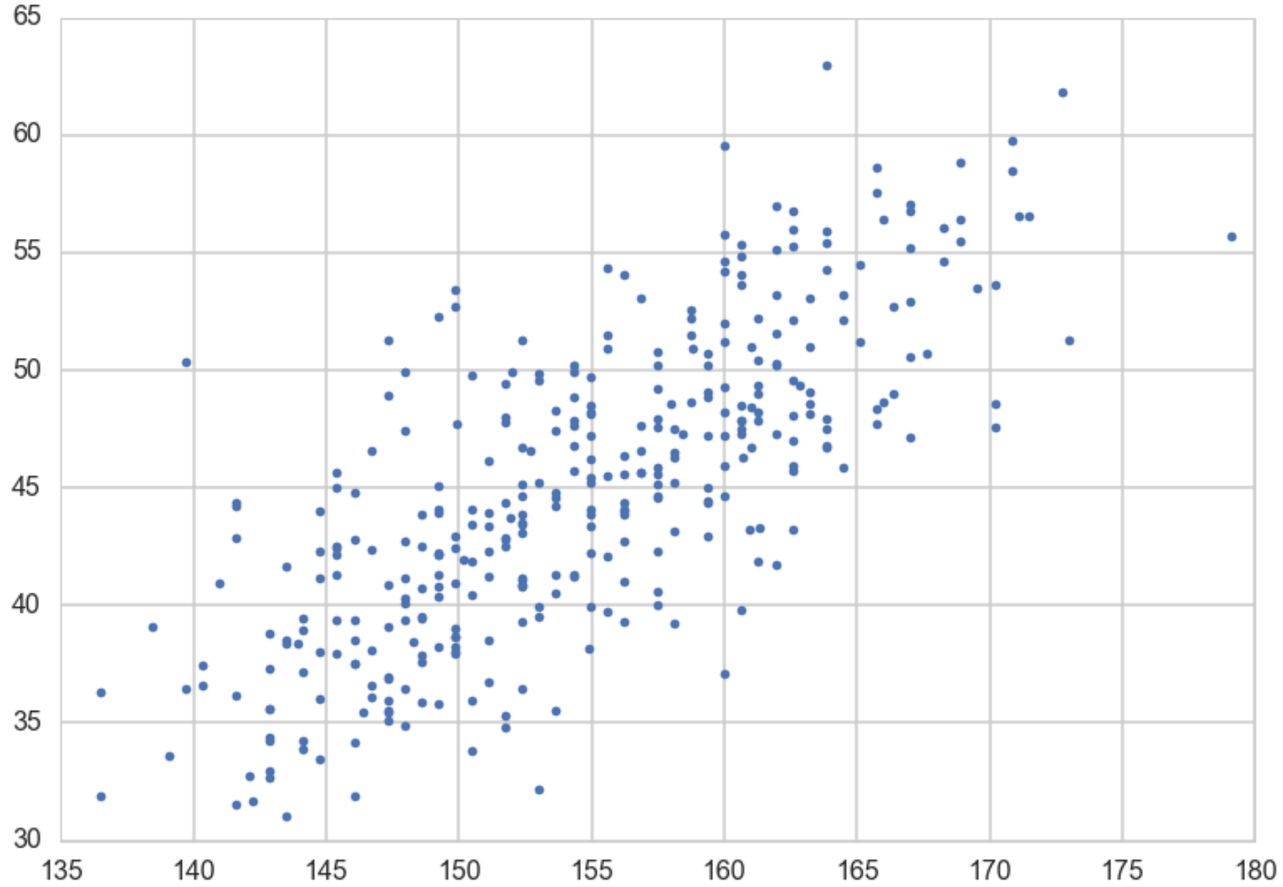
$$= \mathcal{N}\left(y|\bar{\mathbf{w}}^T x^*, \sigma_n^2 + {x^*}^T A^{-1} x^*\right)$$

Howell's data



- These are census data for the Dobe area !Kung San people
- Nancy Howell conducted detailed quantitative studies of this Kalahari foraging population in the 1960s.

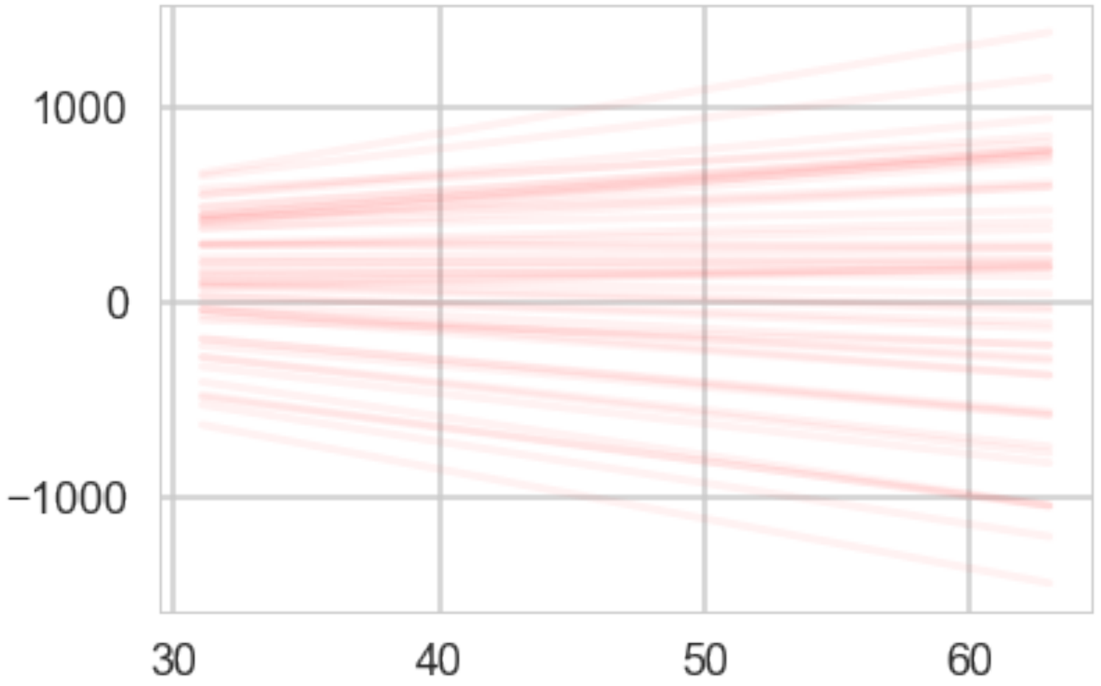
	height	weight	age	male
0	151.765	47.825606	63.0	1
1	139.700	36.485807	63.0	0
2	136.525	31.864838	65.0	0
3	156.845	53.041915	41.0	1
4	145.415	41.276872	51.0	0

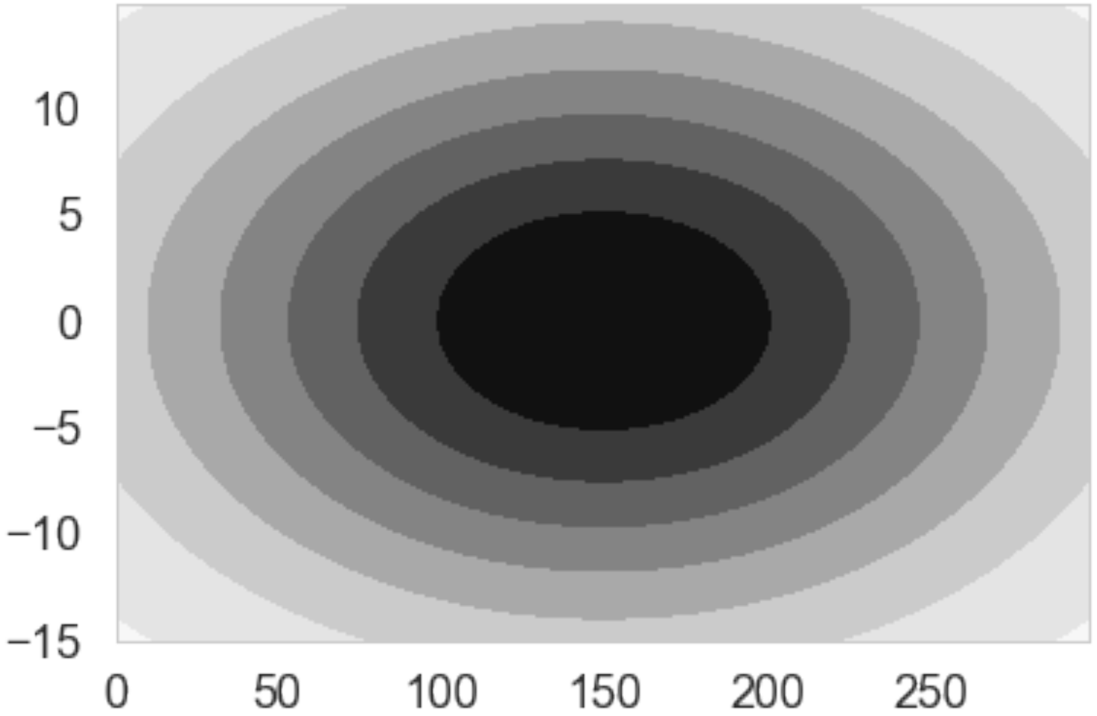


Regression against predictor, weight

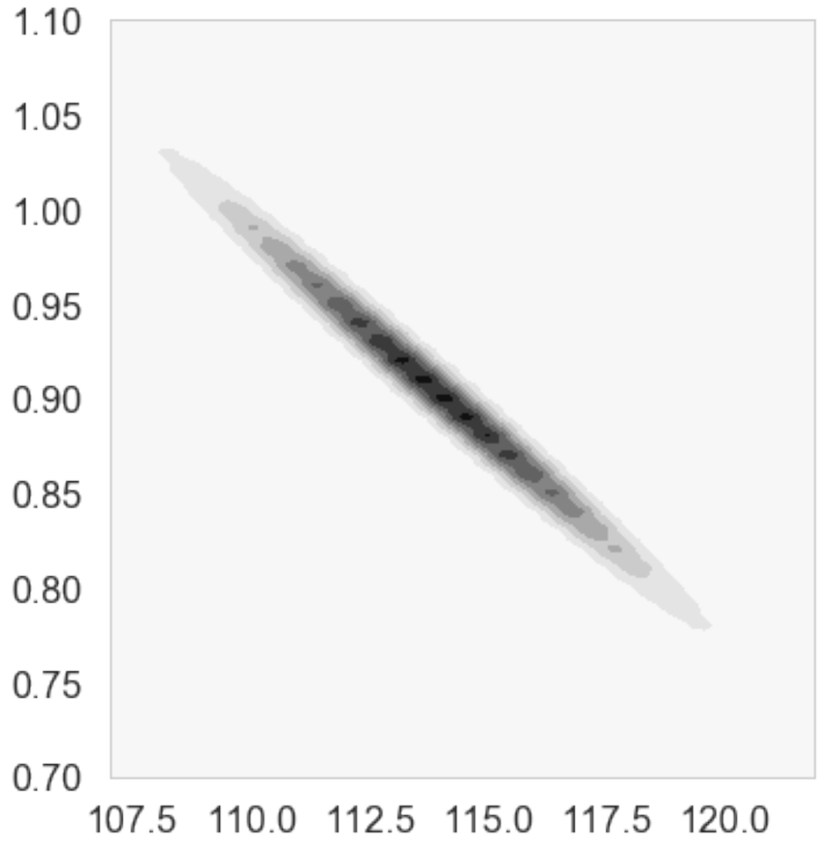
$$h \sim N(\mu, \sigma)$$
$$\mu = intercept + slope \times weight$$
$$intercept \sim N(150, 100)$$
$$slope \sim N(0, 10)$$
$$\sigma = std.dev$$

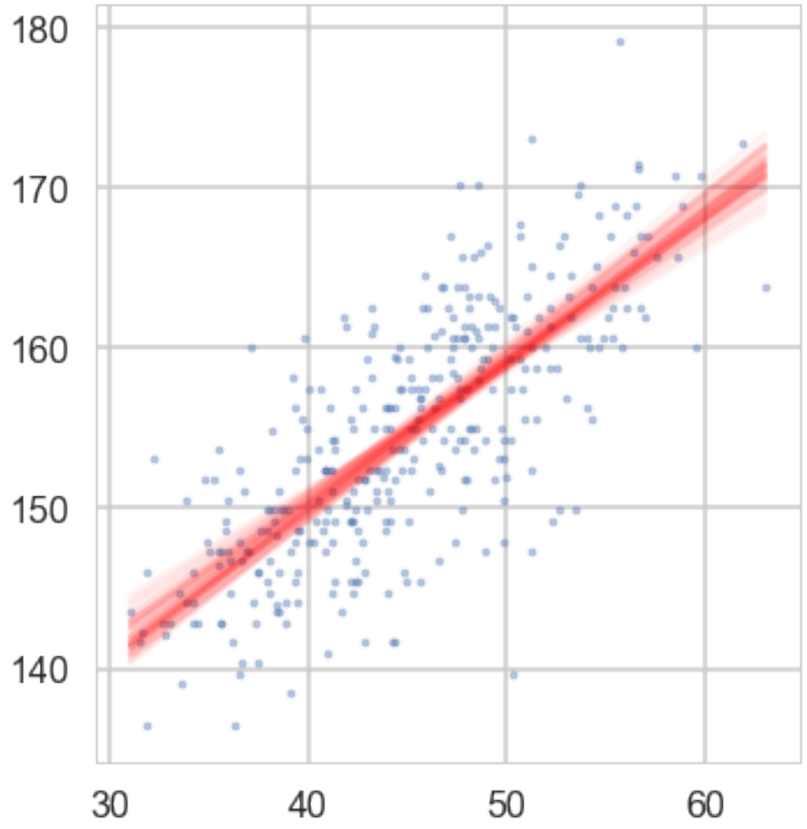
Priors





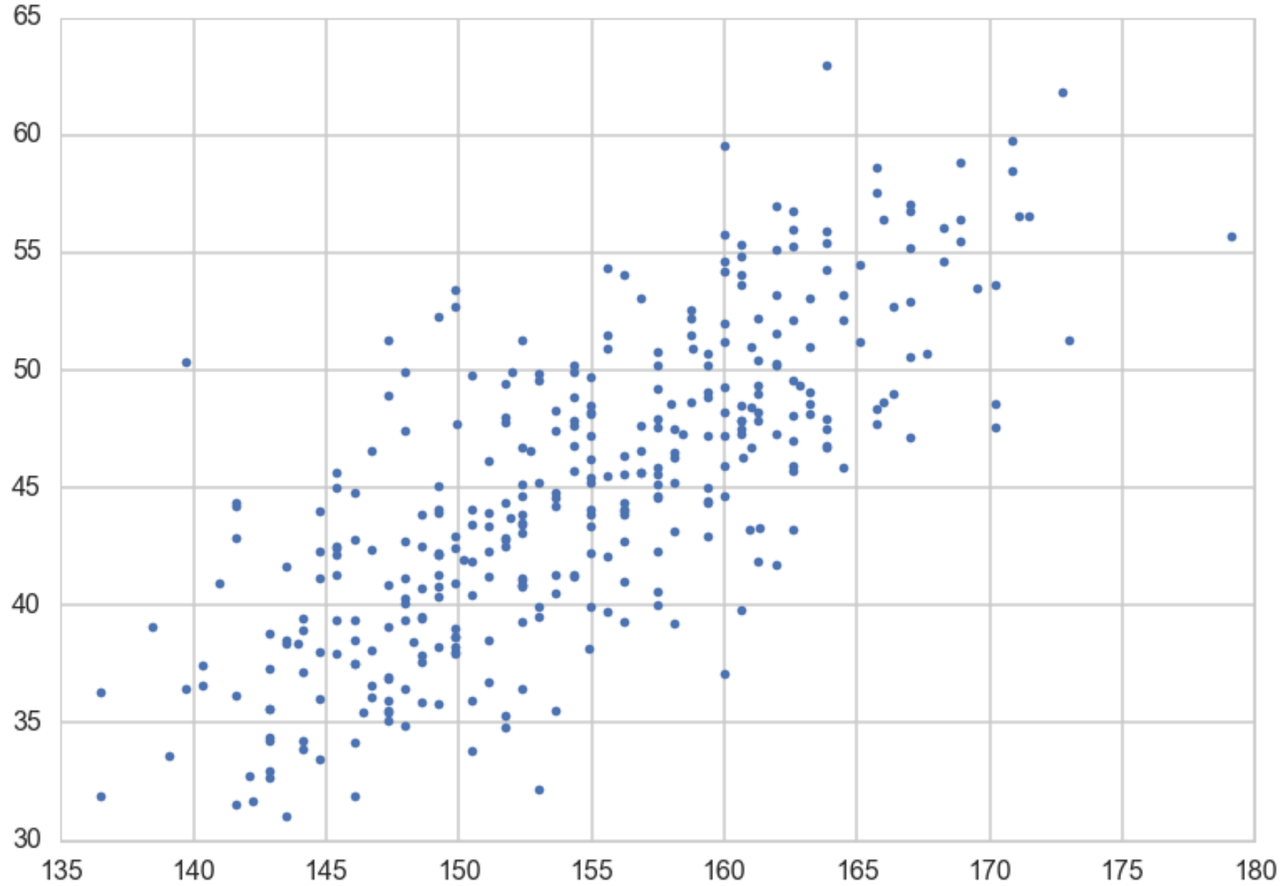
Posteriors


1.10
1.05
1.00
0.95
0.90
0.85
0.80
0.75
0.70
107.5
110.0
112.5
115.0
117.5
120.0


180
170
160
150
140
30
40
50
60

DO IT BY SAMPLING

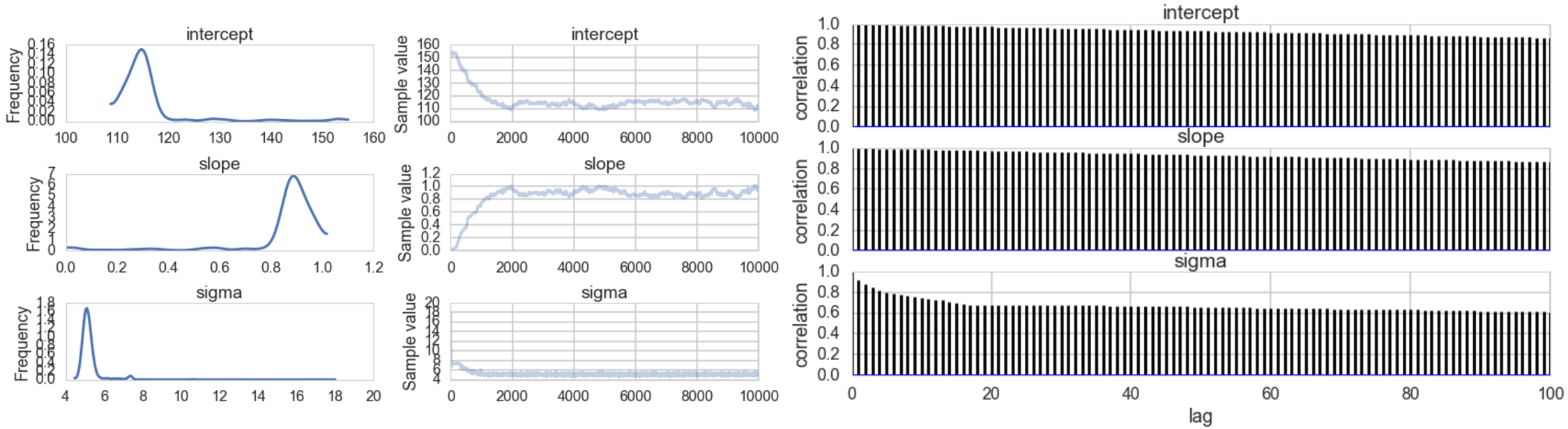
Regression, adding a predictor, weight



$$h \sim N(\mu, \sigma)$$
$$\mu = intercept + slope \times weight$$
$$intercept \sim N(150, 100)$$
$$slope \sim N(0, 10)$$
$$\sigma \sim Unif(0, 50)$$

```
with pm.Model() as hm2:
    intercept = pm.Normal('intercept', mu=150, sd=100)
    slope = pm.Normal('slope', mu=0, sd=10)
    sigma = pm.Uniform('sigma', lower=0, upper=50)
    # below is a deterministic
    mu = intercept + slope * df2.weight
    height = pm.Normal('height', mu=mu, sd=sigma, observed=df2.height)
    stepper=pm.Metropolis()
    tracehm2 = pm.sample(10000, step=stepper)
```

Traces are awful

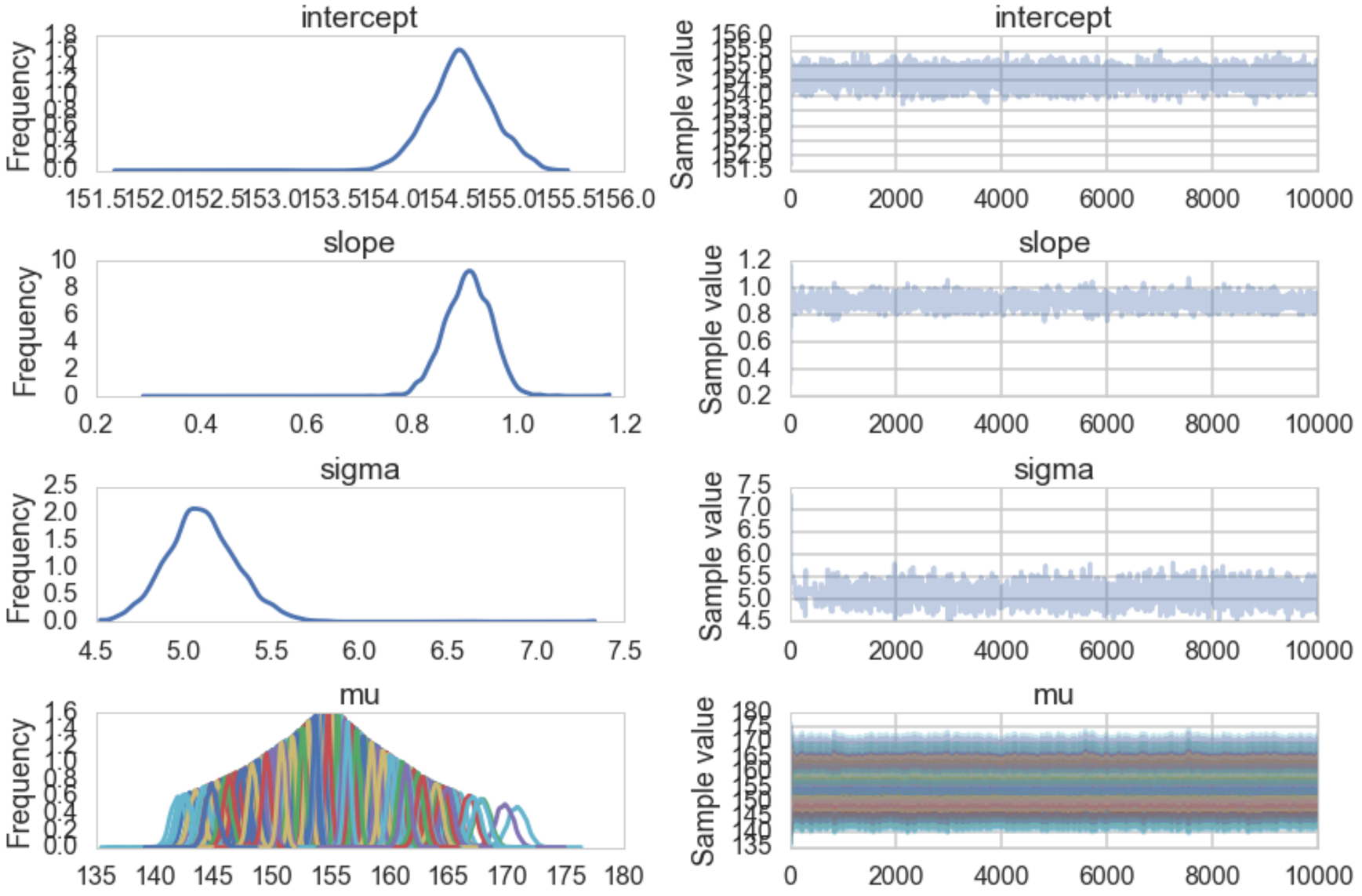


The slope and intercept are very highly correlated: -0.99!

Regression traces

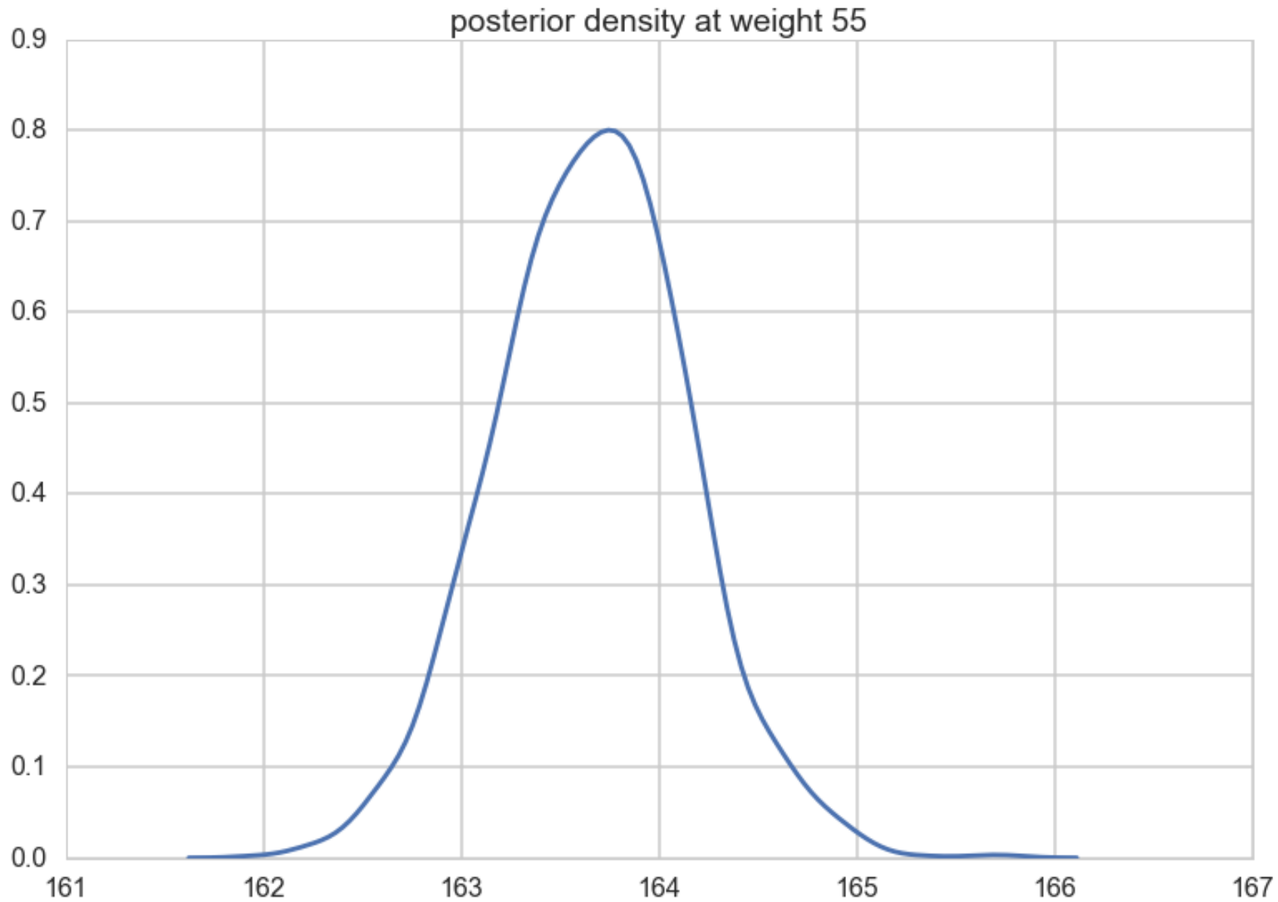
- symptom of shared information and identifiability
- fix by centering. intercept then gives response when predictor=mean.

```
with pm.Model() as hm2c:
    intercept = pm.Normal('intercept', mu=150, sd=100)
    slope = pm.Normal('slope', mu=0, sd=10)
    sigma = pm.Uniform('sigma', lower=0, upper=50)
    mu = pm.Deterministic('mu', intercept + slope * (df2.weight -df2.weight.mean()))
    height = pm.Normal('height', mu=mu, sd=sigma, observed=df2.height)
    stepper=pm.Metropolis()
    tracehm2c = pm.sample(10000, step=stepper)
```

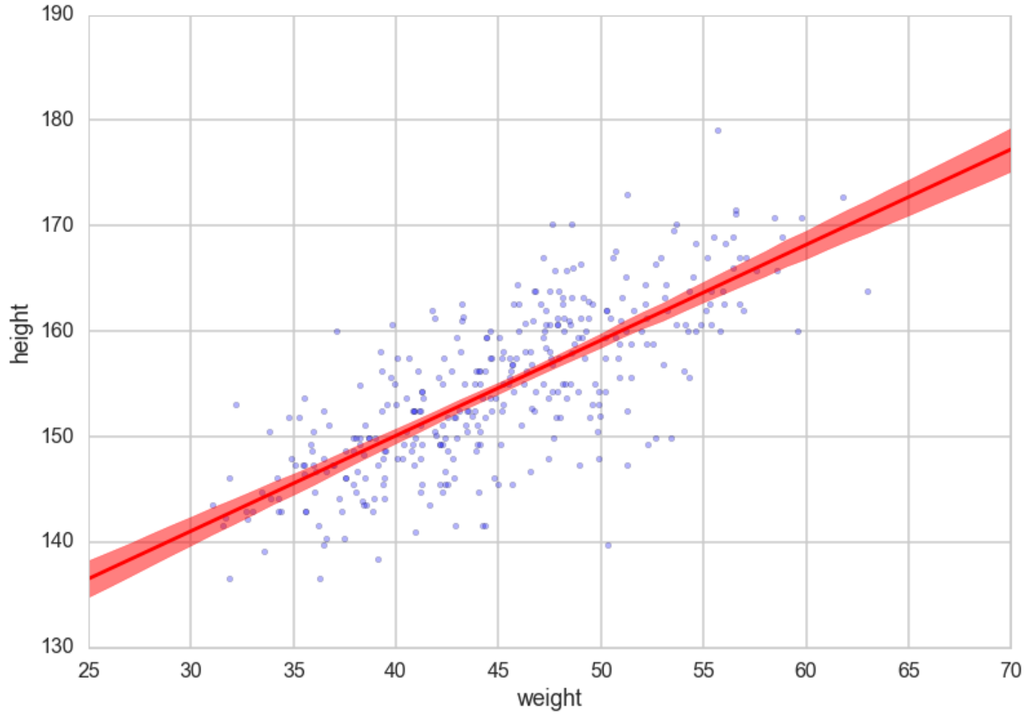

intercept
Frequency
151.5152.0152.5153.0153.5154.0154.5155.0155.5156.0
intercept
Sample value
0
2000
4000
6000
8000
10000
slope
Frequency
0
2
4
6
8
10
0.2
0.4
0.6
0.8
1.0
1.2
slope
Sample value
0.2
0.4
0.6
0.8
1.0
1.2
sigma
Frequency
0.0
0.5
1.0
1.5
2.0
2.5
4.5
5.0
5.5
6.0
6.5
7.0
7.5
sigma
Sample value
4.5
5.0
5.5
6.0
6.5
7.0
7.5
mu
Frequency
135
140
145
150
155
160
165
170
175
180
mu
Sample value
0
2000
4000
6000
8000
10000

Posteriors

```
meanweight = df2.weight.mean()
weightgrid = np.arange(25, 71)
mu_pred = np.zeros((len(weightgrid), len(tr2c)))
for i, w in enumerate(weightgrid):
    mu_pred[i] = tr2c['intercept'] + tr2c['slope'] * (w - meanweight)

mu_mean = mu_pred.mean(axis=1)
mu_hpd = pm.hpd(mu_pred.T)
```



Posteriors on a grid



Why so tight?

Posterior predictive

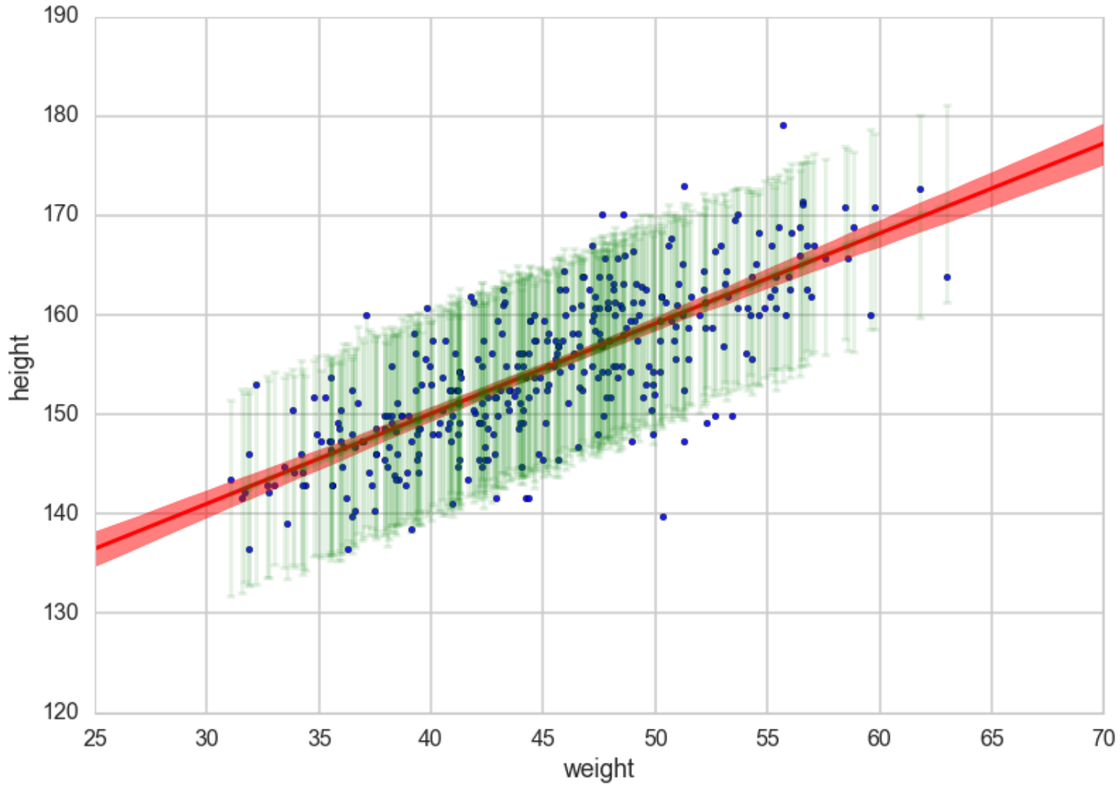
At data:

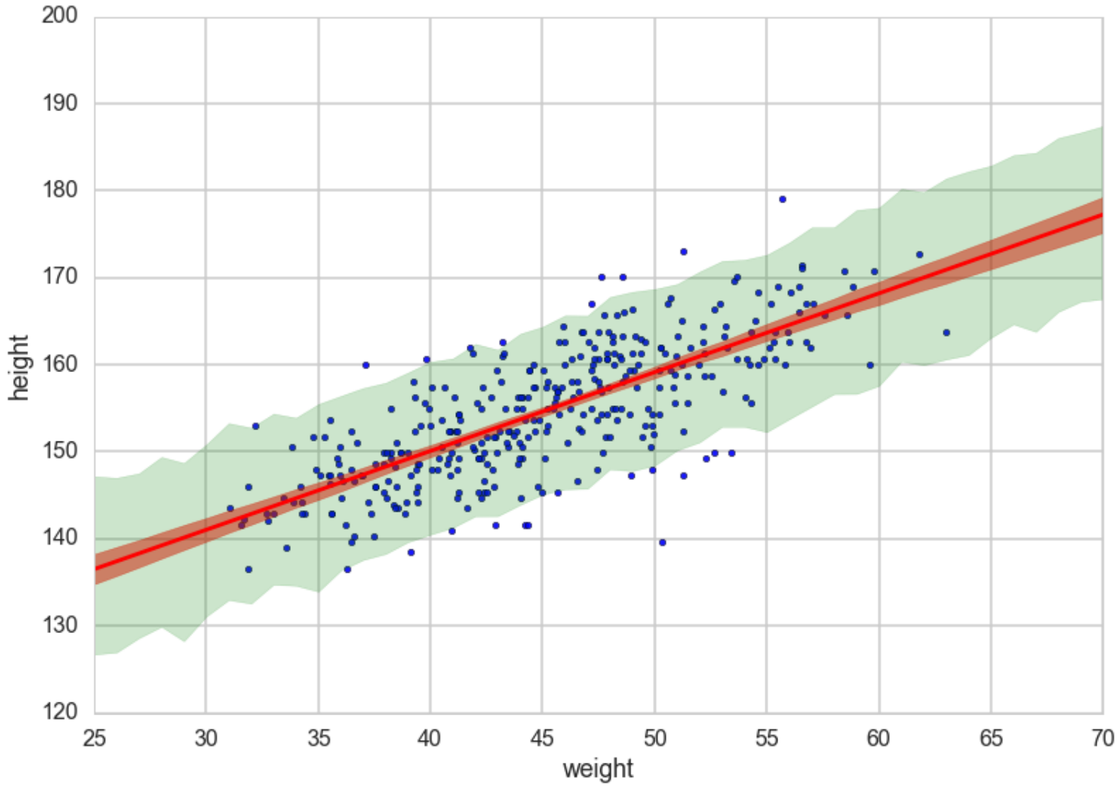
```
postpred = pm.sample_ppc(tr2c, 1000, hm2c)
100%|██████████| 1000/1000 [00:19<00:00, 57.56it/s]    | 1/1000 [00:00<08:17,  2.01it/s]
```

On a full grid:

```
n_ppredsamps=1000
weightgrid = np.arange(25, 71)
meanweight = df2.weight.mean()
ppc_samples=np.zeros((len(weightgrid), n_ppredsamps))

for j in range(n_ppredsamps):
    k=np.random.randint(len(tr2c))#samples with replacement
    musamps = tr2c['intercept'][k] + tr2c['slope'][k] * (weightgrid - meanweight)
    sigmasamp = tr2c['sigma'][k]
    ppc_samples[:,j] = np.random.normal(musamps, sigmasamp)
```

Predictives at data and on grid





glms: MAXENT and LINK

- MAXENT: use all the information we have about the constraints on an outcome variable to choose a likelihood, typically in the exponential family, that is a maxent distribution.
- LINK: $f(p_i) = \alpha + \beta x_i$ where p_i is the parameter at the ith data point.
- common links we use are the *logit* link and the *log* link.

MAXENT

- gaussian likelihood for linear regression maxent choice
- poor choice for constraints such as the outcome being counts, or being only positive.
- use all the information we have about the constraints on an outcome variable to choose a likelihood, typically in the exponential family, that is a maxent distribution.

LINK

$f(p_i) = \alpha + \beta x_i$ where p_i is the parameter at the ith data point.

Bioassay: f is the logit, and the parameter p_i is the probability in the ith experiment, so that we have

$$logit(p_i) = \alpha + \beta x_i,$$

And where the likelihood used is $Binom(n_i, p_i)$.

Poisson GLM

$$y_i \sim Poisson(\lambda_i)$$
$$log(\lambda_i) = \alpha + \beta x_i$$

λ_i is rate, μ_i is counts, τ_i is exposure.

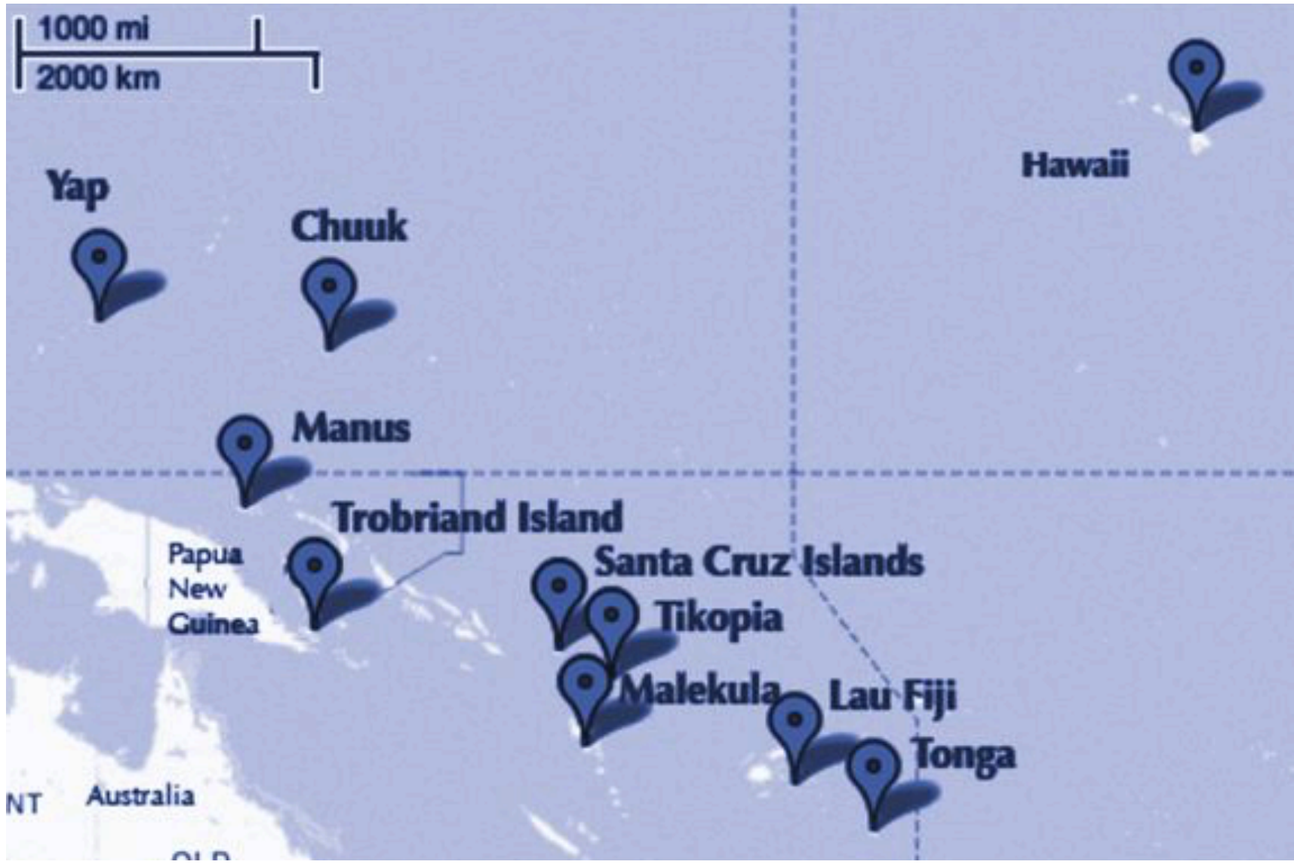
μ_i or λ_i constrained to be positive.

For most GLMs, the common links we use are the *logit* link, already used by you in the bioassay Binomial GLM to model the space of probabilities, and the *log* link which you will use here to enforce positiveness on a parameter in poisson regression.

Oceanic Tools

From Mcelreath:

The island societies of Oceania provide a natural experiment in technological evolution. Different historical island populations possessed tool kits of different size. These kits include fish hooks, axes, boats, hand plows, and many other types of tools. A number of theories predict that larger populations will both develop and sustain more complex tool kits. So the natural variation in population size induced by natural variation in island size in Oceania provides a natural experiment to test these ideas. It's also suggested that contact rates among populations effectively increase population size, as it's relevant to technological evolution. So variation in contact rates among Oceanic societies is also relevant. (McElreath 313)



Model M1

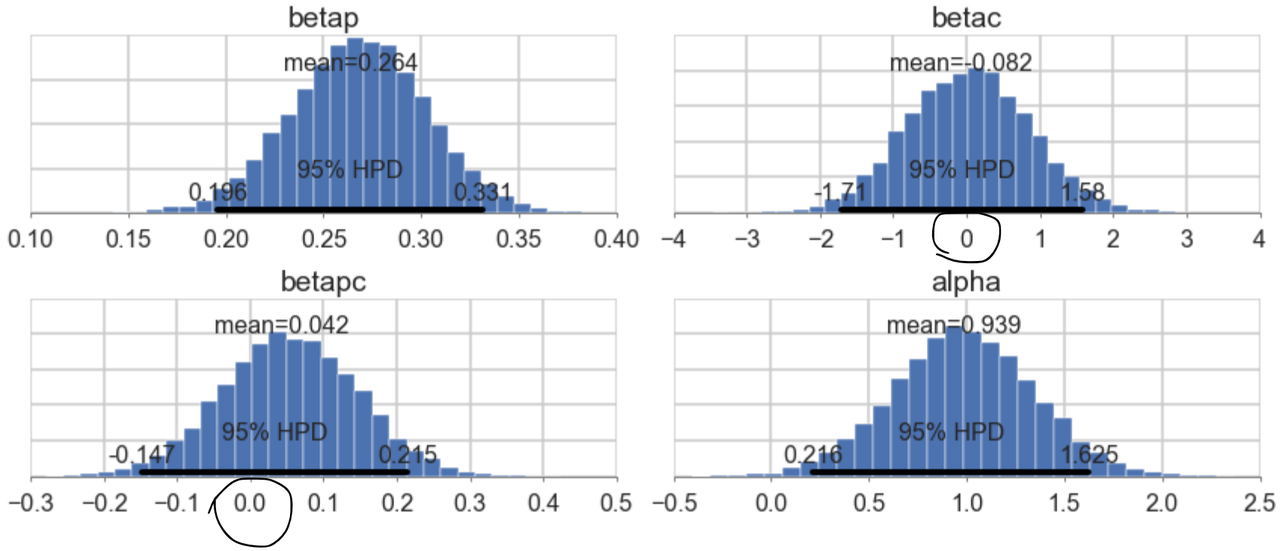
	culture	population	contact	total_tools	mean_TU	logpop	clevel
0	Malekula	1100	low	13	3.2	7.003065	0
1	Tikopia	1500	low	22	4.7	7.313220	0
2	Santa Cruz	3600	low	24	4.0	8.188689	0
3	Yap	4791	high	43	5.0	8.474494	1
4	Lau Fiji	7400	high	33	5.0	8.909235	1
5	Trobriand	8000	high	19	4.0	8.987197	1
6	Chuuk	9200	high	40	3.8	9.126959	1
7	Manus	13000	low	28	6.6	9.472705	0
8	Tonga	17500	high	55	5.4	9.769956	1
9	Hawaii	275000	low	71	6.6	12.524526	0

$$
\begin{aligned}
T_i &\sim Poisson(\lambda_i) \\
log(\lambda_i) &= \alpha + \beta_P log(P_i) + \beta_C C_i + \beta_{PC} C_i log(P_i) \\
\alpha &\sim N(0, 100) \\
\beta_P &\sim N(0, 1) \\
\beta_C &\sim N(0, 1) \\
\beta_{PC} &\sim N(0, 1)
\end{aligned}
$$

```
with pm.Model() as m1:
    betap = pm.Normal("betap", 0, 1)
    betac = pm.Normal("betac", 0, 1)
    betapc = pm.Normal("betapc", 0, 1)
    alpha = pm.Normal("alpha", 0, 100)
    loglam = alpha + betap*df.logpop +
        betac*df.clevel + betapc*df.clevel*df.logpop
    y = pm.Poisson("ntools", mu=t.exp(loglam), observed=df.total_tools)
```

```
with m1:
    trace=pm.sample(10000, njobs=2)
Average ELBO = -55.784:
100%|██████████| 200000/200000 [00:15<00:00, 13019.16it/s]   12683.03it/s]
100%|██████████| 10000/10000 [01:59<00:00, 83.80it/s]
```

Posteriors for M1



- traces and autocorrelations look good
- The posterior for β_p tightly constrained, and as expected from theory, shows a positive effect.
- The posteriors for β_c and β_{pc} both overlap 0 substantially, and seem comparatively poorly constrained.
- no substantial effect of contact rate, directly or through the interaction?

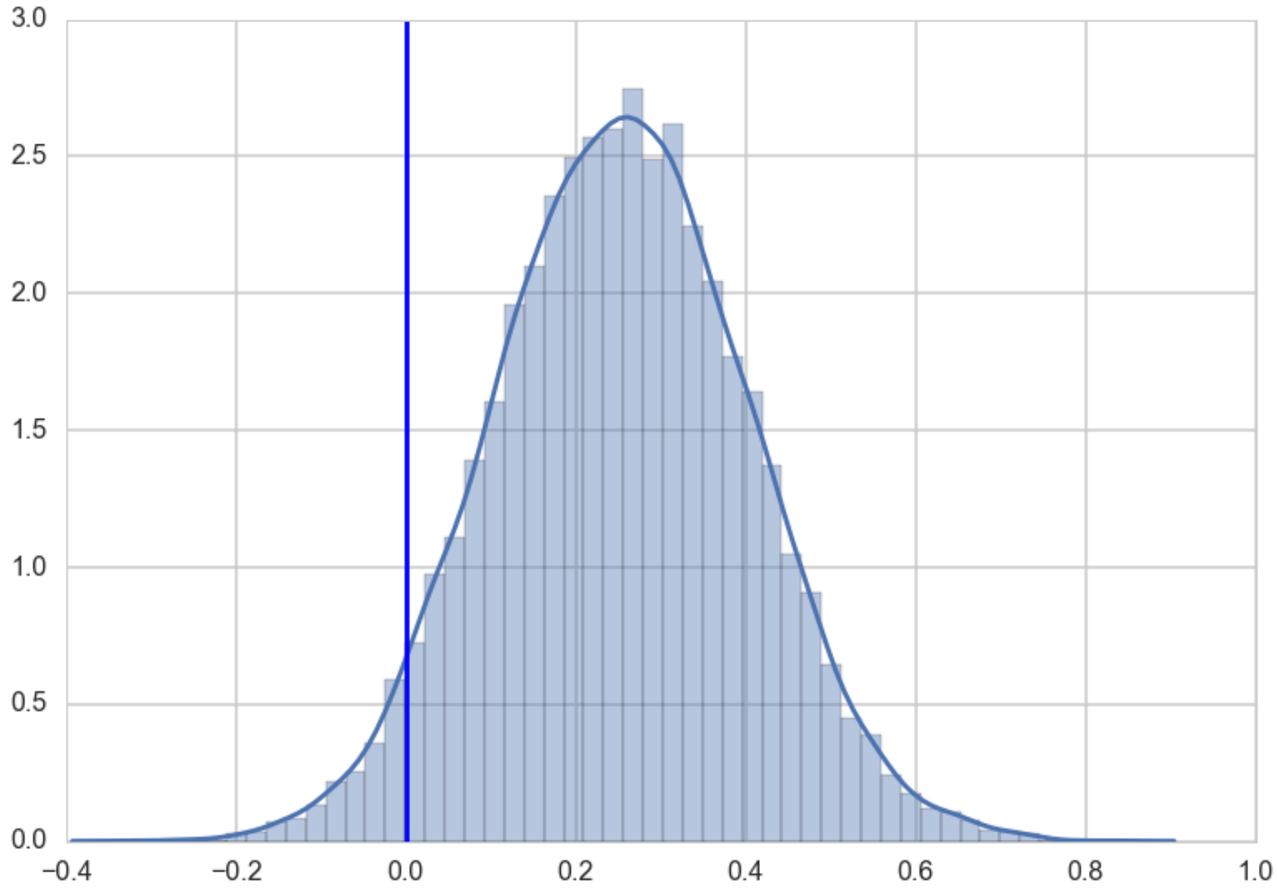
```
Since 0 \in CI of beta_c and beta_pc posterior draws,
beta_c and beta_pc **seem** to be NOT significant...
```

You would be wrong: counterfactual predictions

λ traces for high-contact and low contact, log(population) of 8.

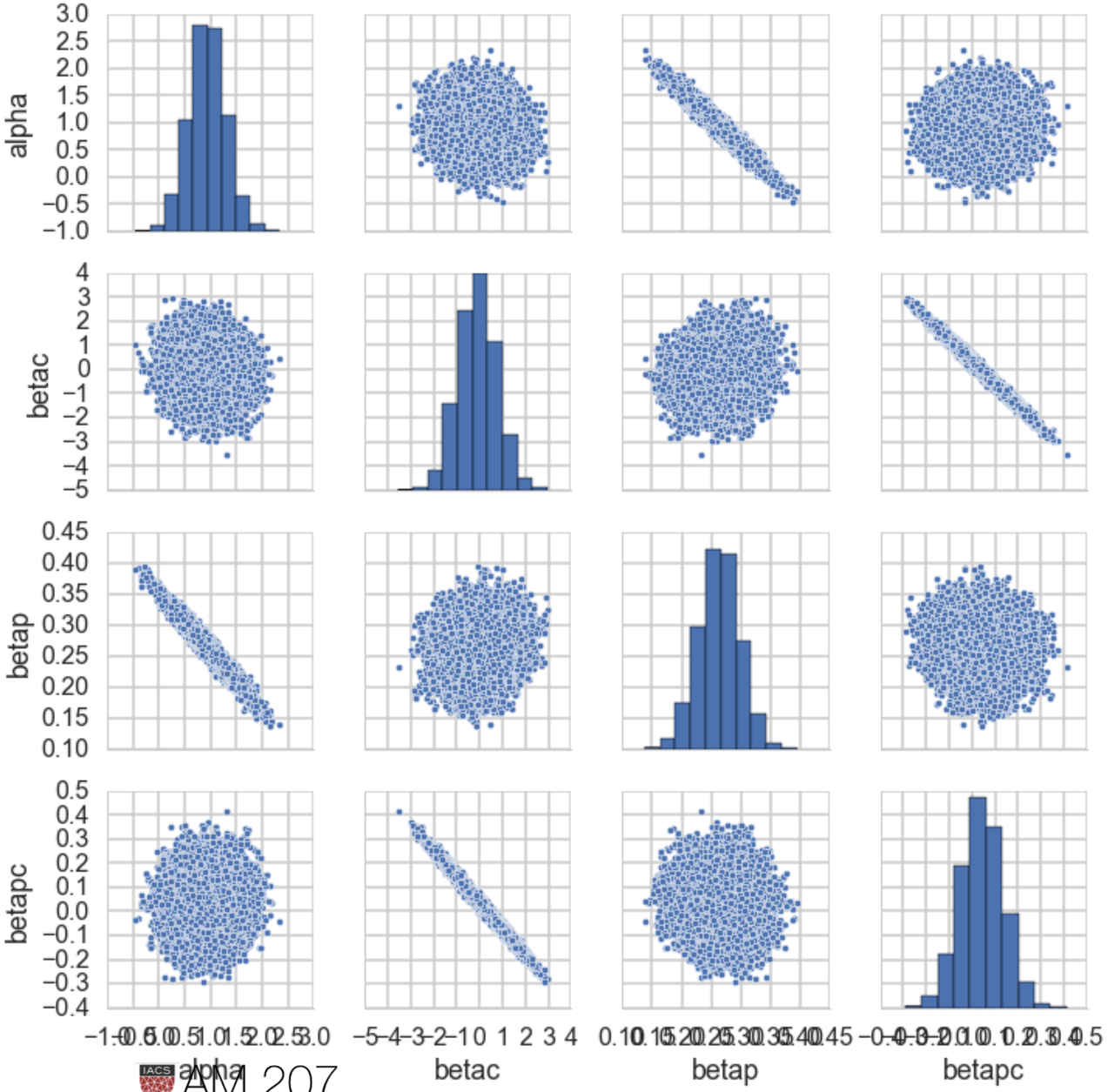
```
lamlow = lambda logpop: trace['alpha']+trace['betap']*logpop
lamhigh = lambda logpop: trace['alpha']+(trace['betap'] +
    trace['betapc'])*logpop + trace['betac']
sns.distplot(lamhigh(8) - lamlow(8));
```

A new kind of model checking.



lamlow: log(lambda)'s for contact = low
lamhigh: log(lambda)'s for contact = high

****Interpretation**: If contact is insignificant, then the distribution of (lamhigh - lamlow) would be centered around 0, which contradicts with the plot on the right**



What happened?

- very strong negative correlations between α and β_p
- very strong negative correlations between β_c and β_{pc}.
- The latter is the cause for the 0-overlaps.
- When β_c is high, β_{pc} must be low, and vice-versa. Look at the joint uncertainty of the correlated variables rather than just marginals

=> Statistical significance from whether 0 is in the posterior CI's is not credible if the features are correlated.

Fix by centering

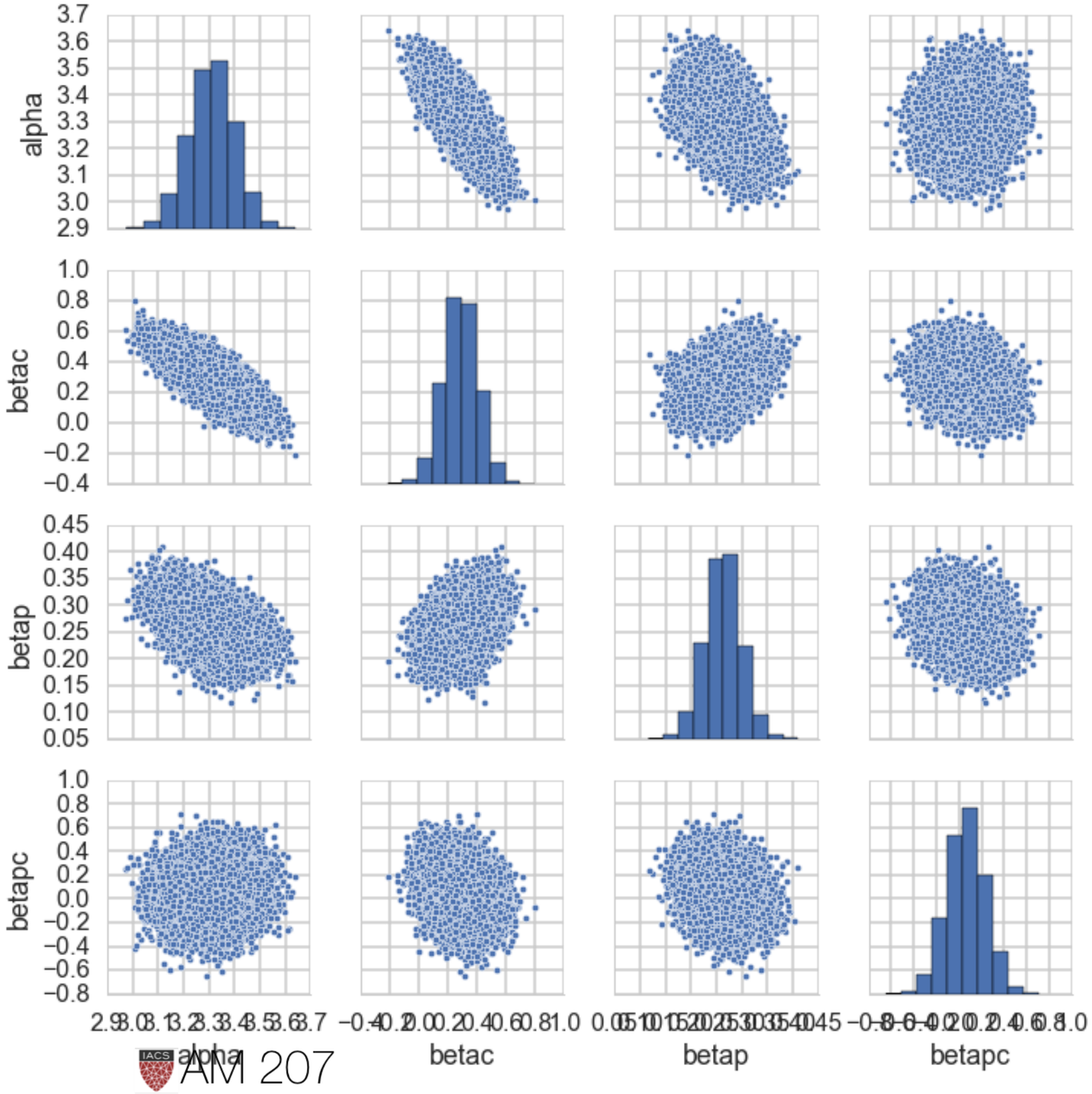
- you would have seen the problem in n_{eff}:

```
{'alpha': 8110.0, 'betac': 4600.0, 'betap': 8016.0, 'betapc': 4597.0}
```

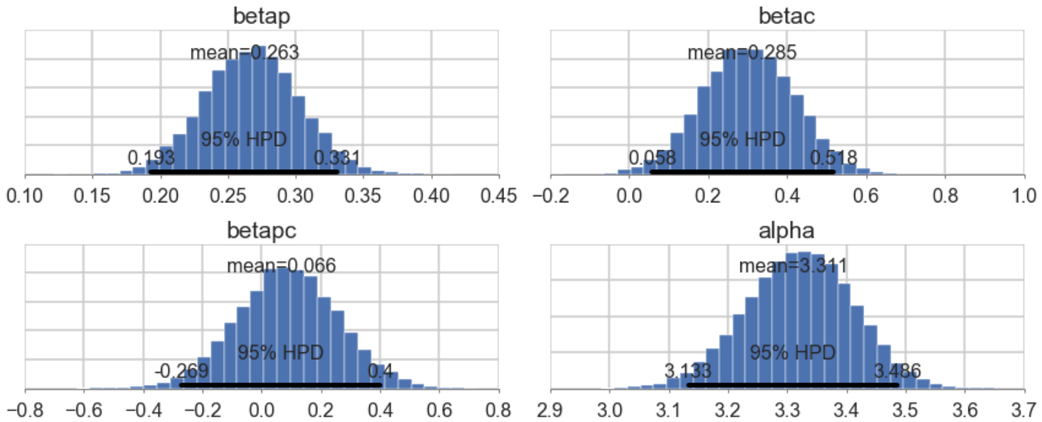
```
with pm.Model() as m1c:
    betap = pm.Normal("betap", 0, 1)
    betac = pm.Normal("betac", 0, 1)
    betapc = pm.Normal("betapc", 0, 1)
    alpha = pm.Normal("alpha", 0, 100)
    loglam = alpha + betap*df.logpop_c + betac*df.clevel + betapc*df.clevel*df.logpop_c
    y = pm.Poisson("ntools", mu=t.exp(loglam), observed=df.total_tools)
```

if logpop is 0 mean (centered),
the 2 terms would not be correlated

```
{'alpha': 7978.0, 'betac': 7898.0, 'betap': 13621.0, 'betapc': 17703.0}
```



- better constrained, less correlated, sampling faster and better
- clear effect of contact, effect of interaction not clear yet
- will use model comparison next time for this!



Hierarchicals with NUTS

Normal-Normal Hierarchical Model

J independent experiments, experiment j estimating the parameter θ_j from n_j independent normally distributed data points, y_{ij}, each with known error variance σ^2; that is,

$$y_{ij}|\theta_j \sim N(\theta_j, \sigma^2),\ i = 1, \ldots, n_j; j = 1, \ldots, J.$$

Gelman 8-schools problem: estimated coaching effects $\bar{y}_j$ to improve SAT scores for school j, with sampling variances, σ_j^2.

Sample mean of each group j

$\bar{y}_j = \frac{1}{n_j}\sum_{i=1}^{n_j} y_{ij}$ with sampling variance

$\sigma_j^2 = \sigma^2/n_j.$

Likelihood for θ_j using suff-stats, $\bar{y}_j$:

$$\bar{y}_j|\theta_j \sim N(\theta_j, \sigma_j^2).$$

Notation flexible in allowing a separate variance σ_j^2 for the mean of each group j.

Appropriate when the variances differ for reasons other than number of data pts.

School	Estimated treatment effect, y_j	Standard error of effect estimate, σ_j
A	28	15
B	8	10
C	−3	16
D	7	11
E	−1	9
F	1	11
G	18	10
H	12	18

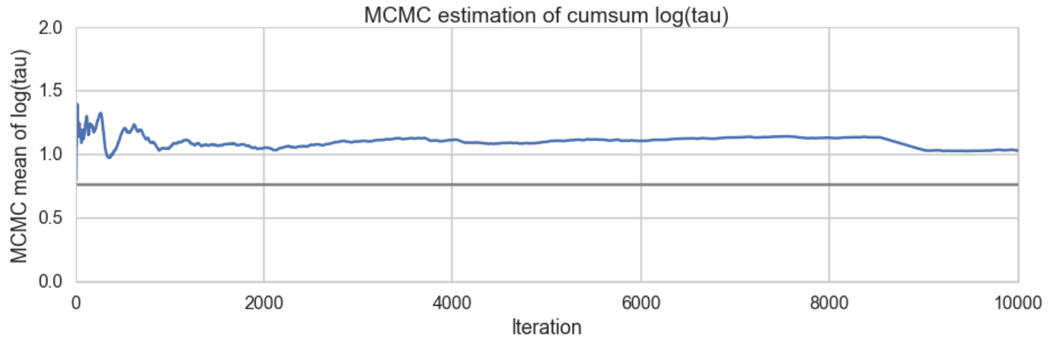
Centered Hierarchical Model

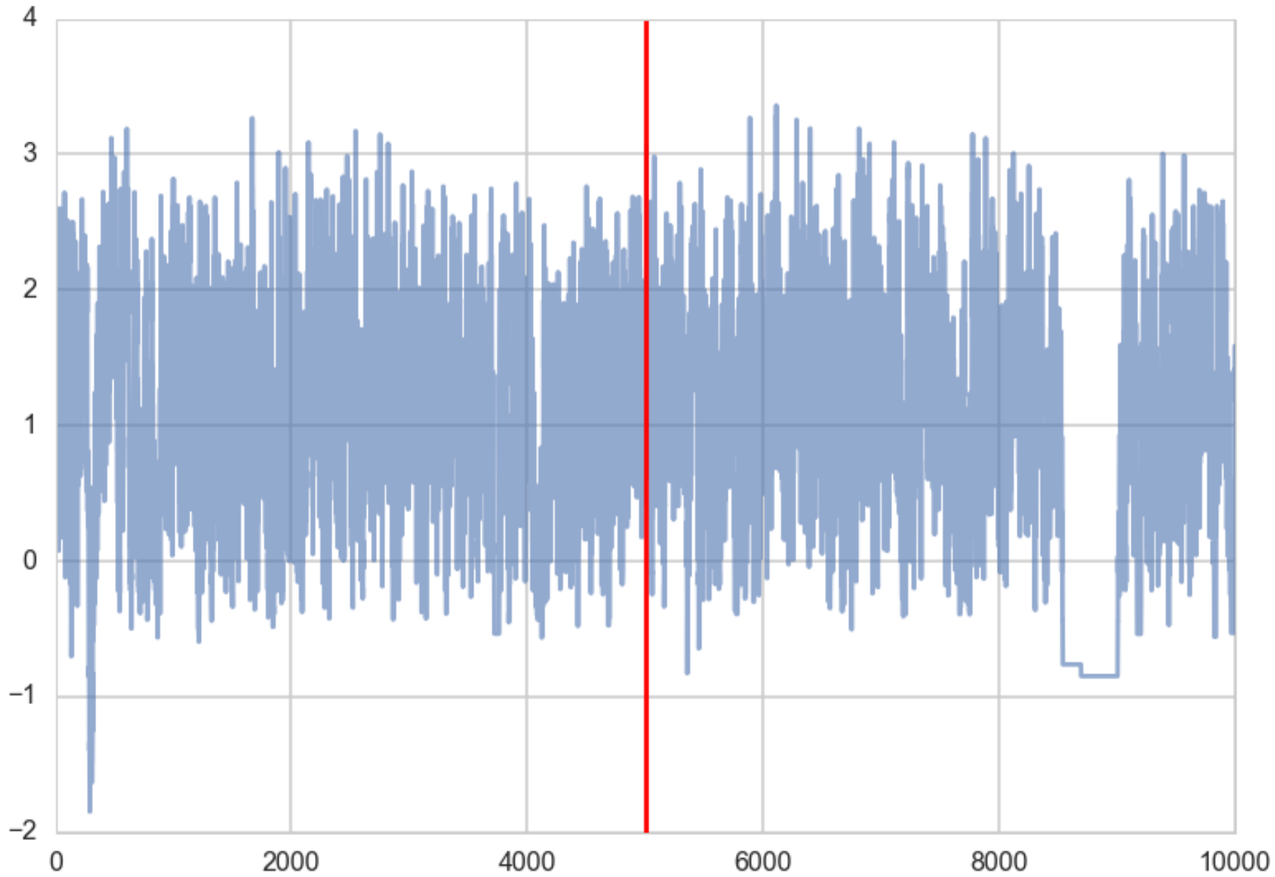
$$\mu \sim \mathcal{N}(0, 5)$$
$$\tau \sim \text{Half-Cauchy}(0, 5)$$
$$\theta_j \sim \mathcal{N}(\mu, \tau)$$
$$\bar{y}_j \sim \mathcal{N}(\theta_j, \sigma_j)$$

```
with pm.Model() as schools1:

    mu = pm.Normal('mu', 0, sd=5)
    tau = pm.HalfCauchy('tau', beta=5)
    theta = pm.Normal('theta', mu=mu, sd=tau, shape=J)
    obs = pm.Normal('obs', mu=theta, sd=sigma, observed=y)

with schools1:
    trace1 = pm.sample(5000, init=None, njobs=2, tune=500)
```

Small n_{eff}:

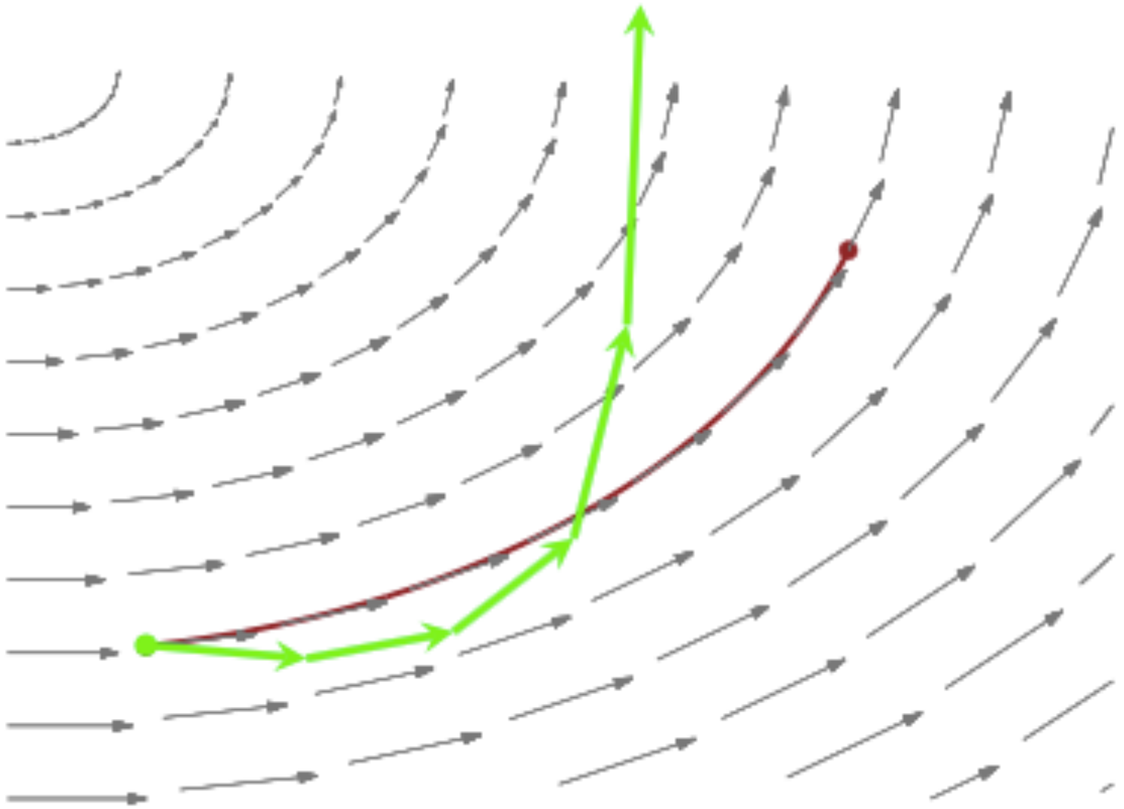
```
{'mu': 101.0,
  'tau': 273.0,
  'tau_log_': 77.0,
  'theta': array([ 169.,  199.,  236.,  193.,  211.,  231.,  139.,  204.])})
```





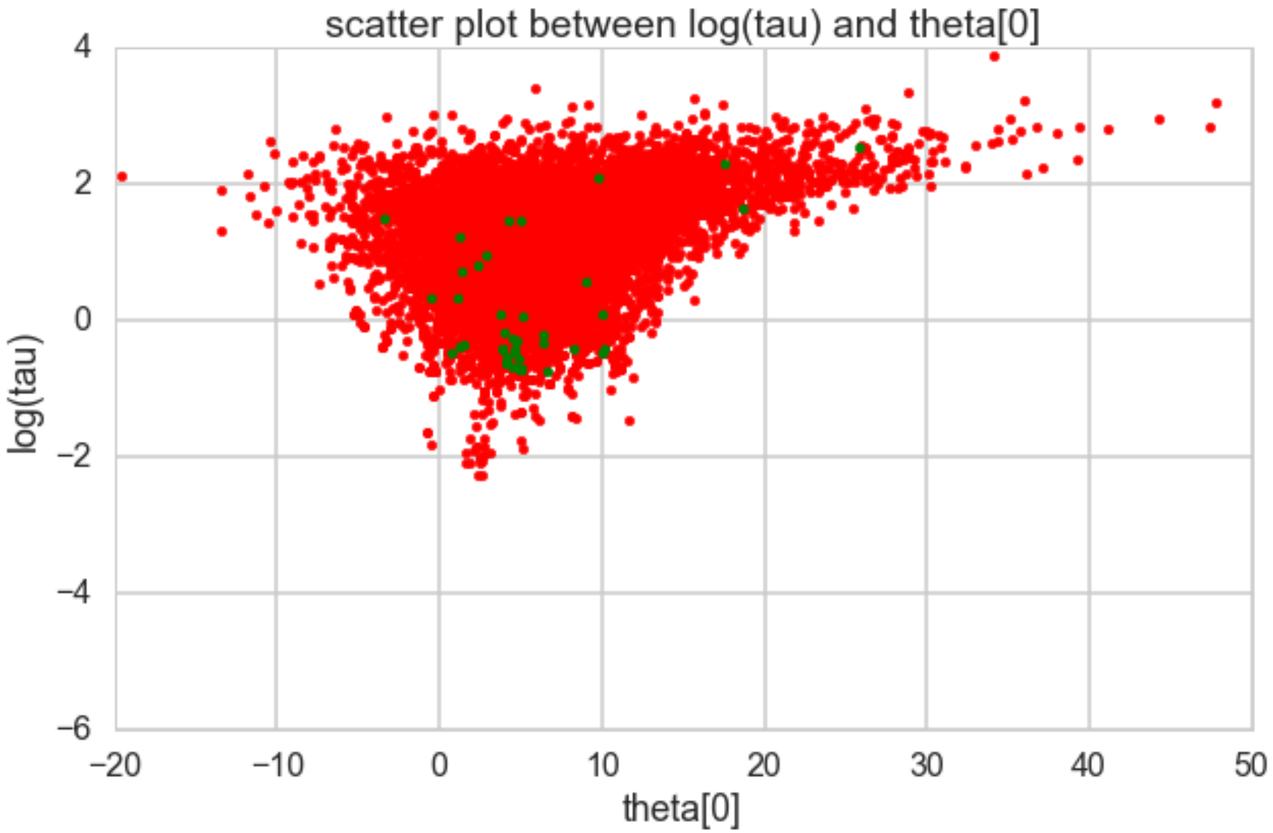
- stickys are actually trying to drive down value of trace
- we are in a region of high curvature

High Curvature Issues

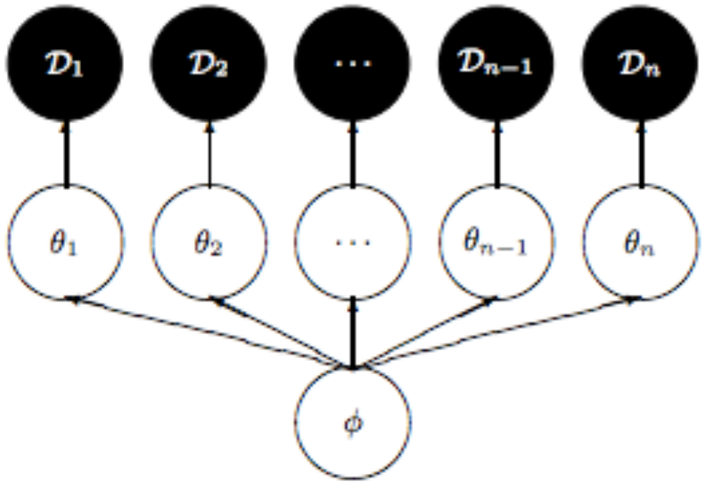


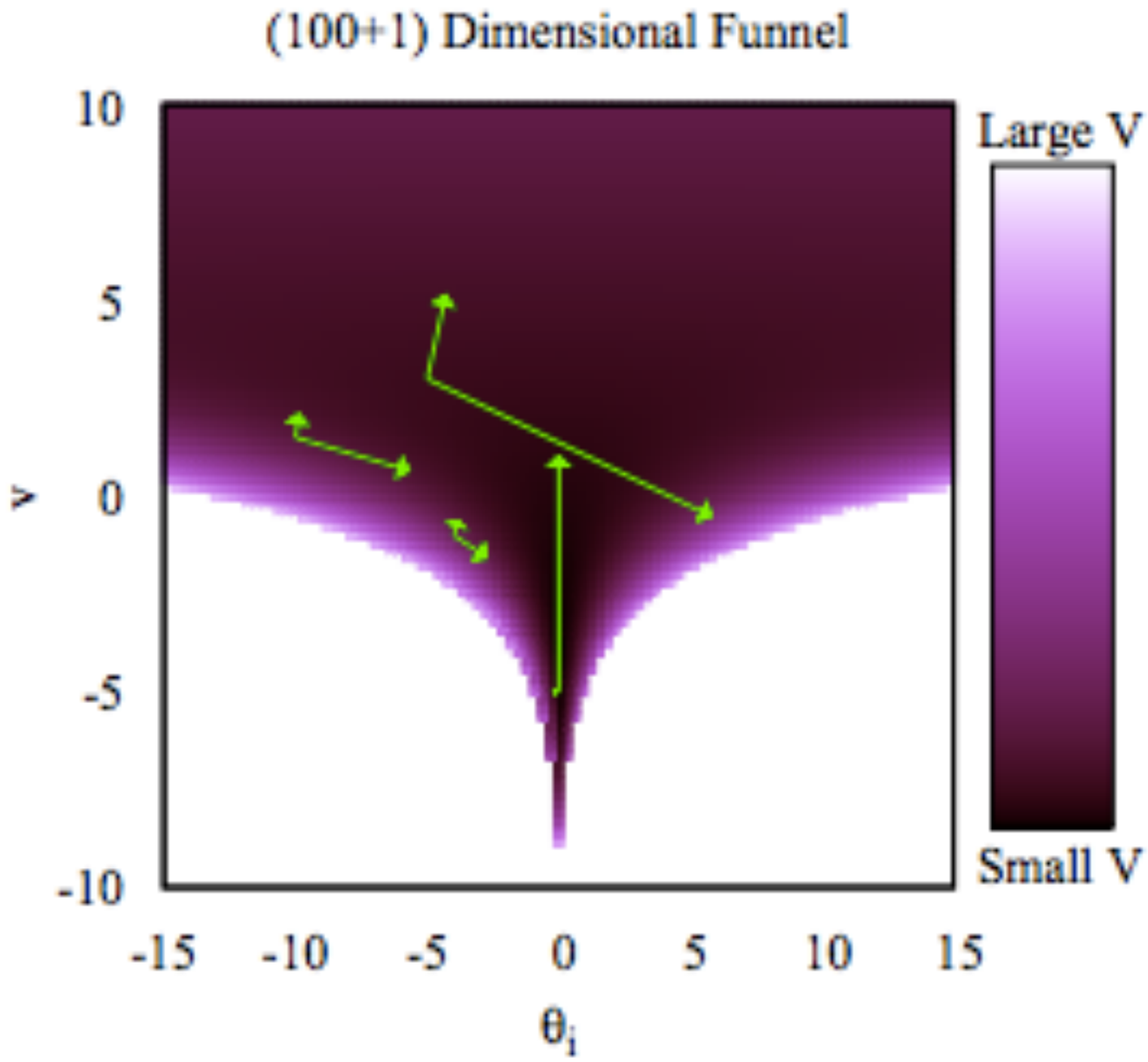
High Curvature Issues

- symplectic integration diverges: good diagnostic. False positives from heuristic.
- sampler needs to have real small steps to not diverge, but then becomes sticky
- regions of high curvature often have high energy differences, causing trouble for microcanonical jump transitions.



Hierarchical Models have high curvature





- characteristic funnel, also there in MH and gibbs
- reflects high correlation between levels in tree
- divergences occur in neck

Step size effect

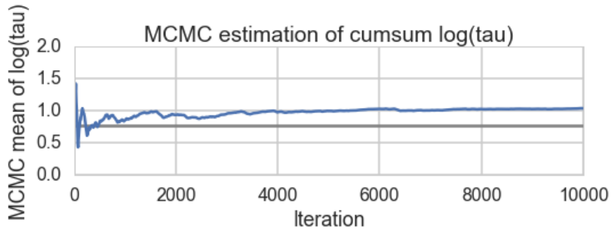
Symplectic Integrator: a numerical integration scheme for Hamiltonian systems

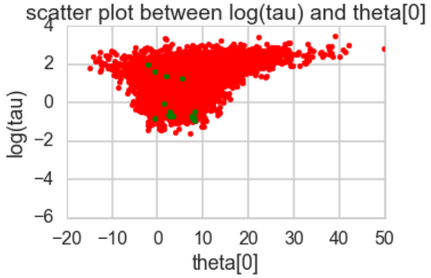
- lower step size ϵ better for symplectic integrators, especially in high curvature regions
- this allows for geometric ergodicity: we go everywhere.
- too small ϵ: return of the random walk.

Changing step size

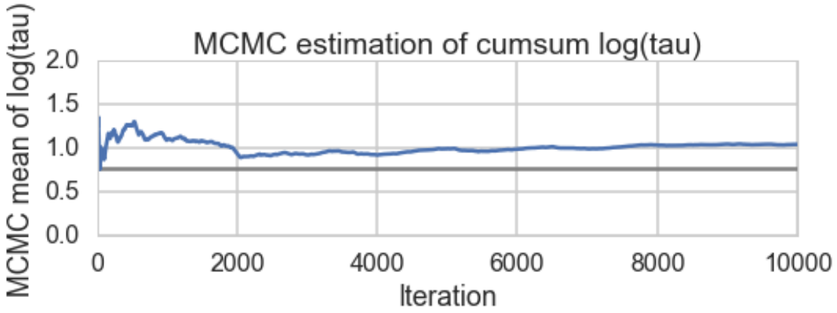
```
with schools1:
    step = pm.NUTS(target_accept=.85)
    trace1_85 = pm.sample(5000, step=step, init=None, njobs=2, tune=1000)
```

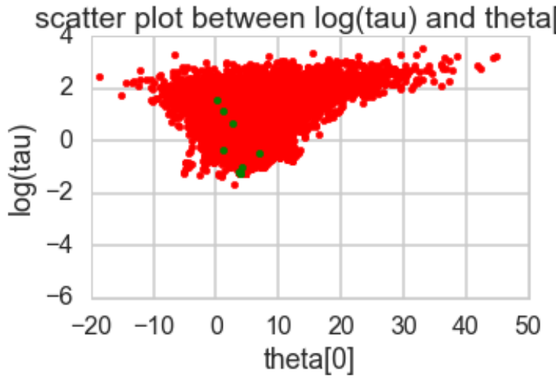
```
85: Acceptance 0.804601458758 Step Size 0.203087336483 Divergence 39
90: Acceptance 0.873340820433 Step Size 0.159223726996 Divergence 18
95: Acceptance 0.923346597897 Step Size 0.126824682121 Divergence 9
99: Acceptance 0.990173791609 Step Size 0.0164237997757 Divergence 5
```

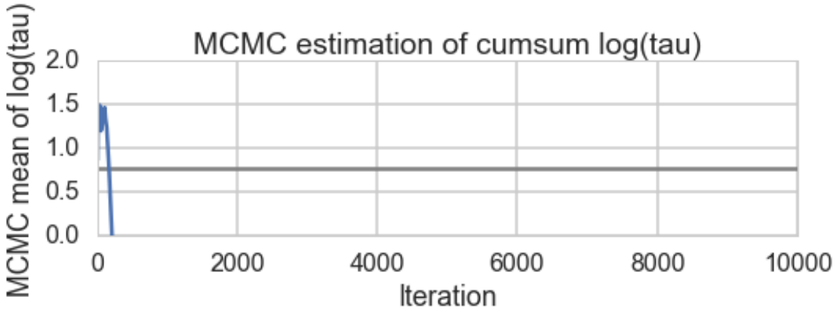


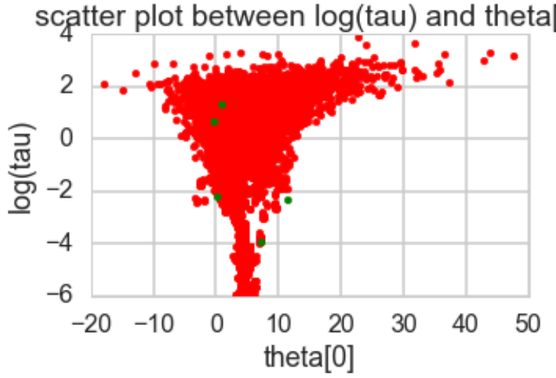


divergences persist. Too curved!



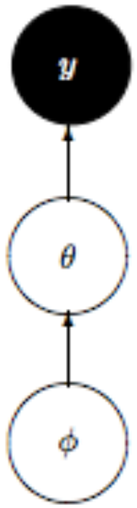


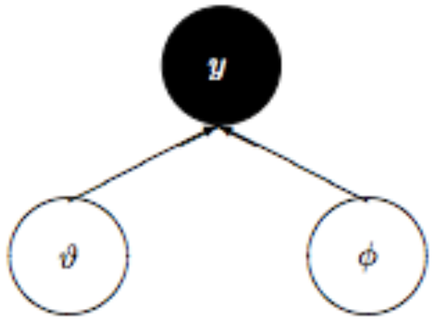




Non-centered model

- could change kinetic energy (riemannian HMC) to make mass matrix dependent upon position
- simpler: reparametrize to reduce levels in hierarchy

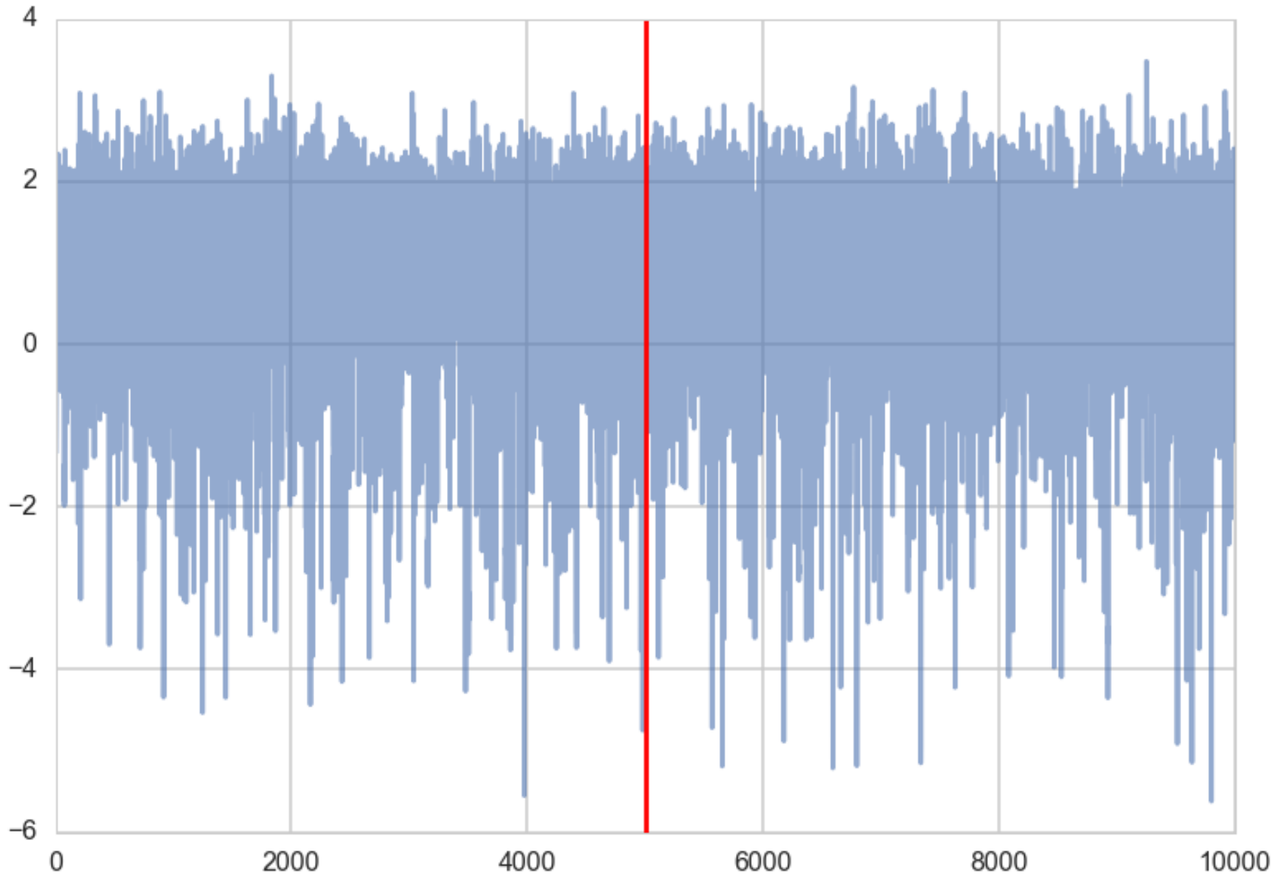


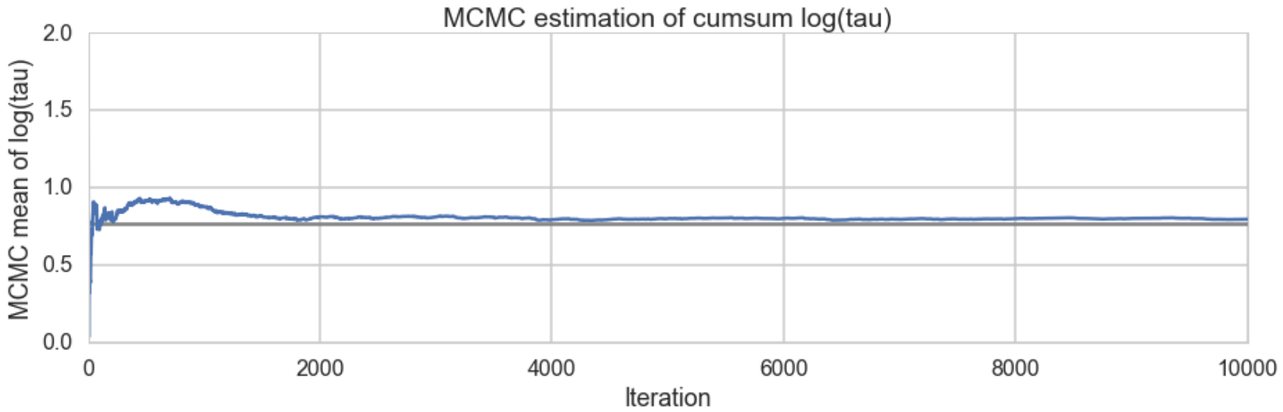


$$\mu \sim \mathcal{N}(0, 5)$$
$$\tau \sim \text{Half-Cauchy}(0, 5)$$
$$\nu_j \sim \mathcal{N}(0, 1)$$
$$\theta_j = \mu + \tau\nu_j$$
$$\bar{y}_j \sim \mathcal{N}(\theta_j, \sigma_j)$$

Factor dependency of θ on $\phi = \mu, \tau$ into a deterministic transformation between the layers, leaving the actively sampled variables uncorrelated.

```
with pm.Model() as schools2:
    mu = pm.Normal('mu', mu=0, sd=5)
    tau = pm.HalfCauchy('tau', beta=5)
    nu = pm.Normal('nu', mu=0, sd=1, shape=J)
    theta = pm.Deterministic('theta', mu + tau * nu)
    obs = pm.Normal('obs', mu=theta, sd=sigma, observed=y)
    trace2 = pm.sample(5000, init=None, njobs=2, tune=500)
```





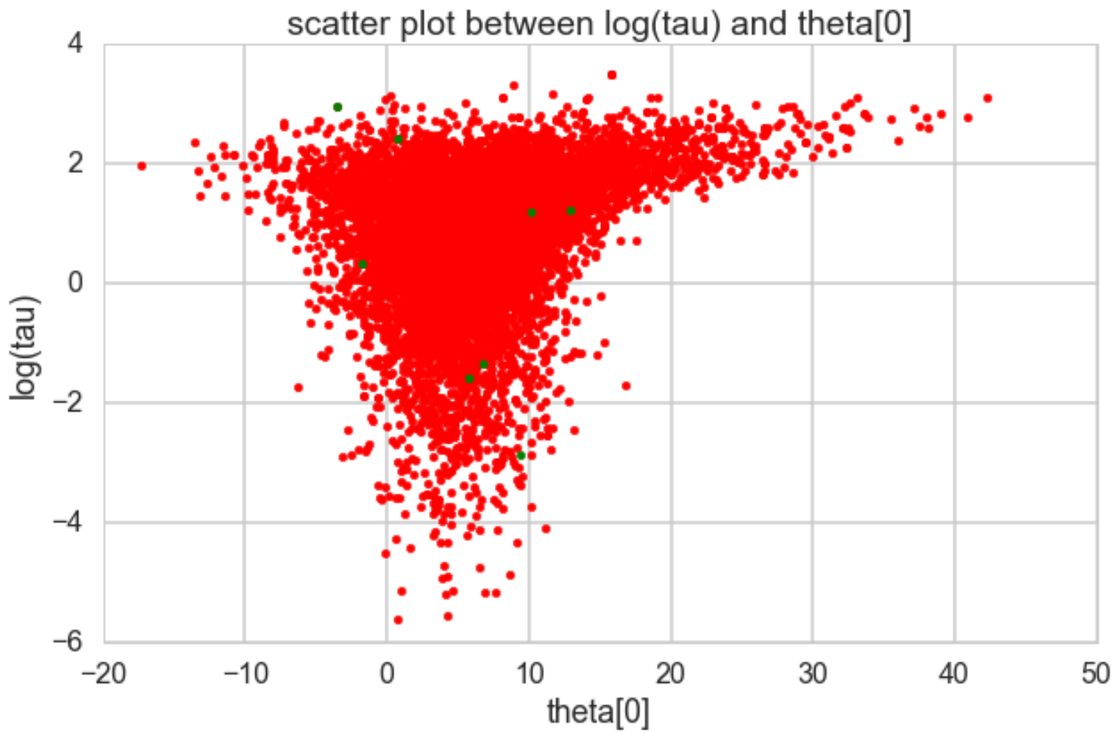
n_{eff}:

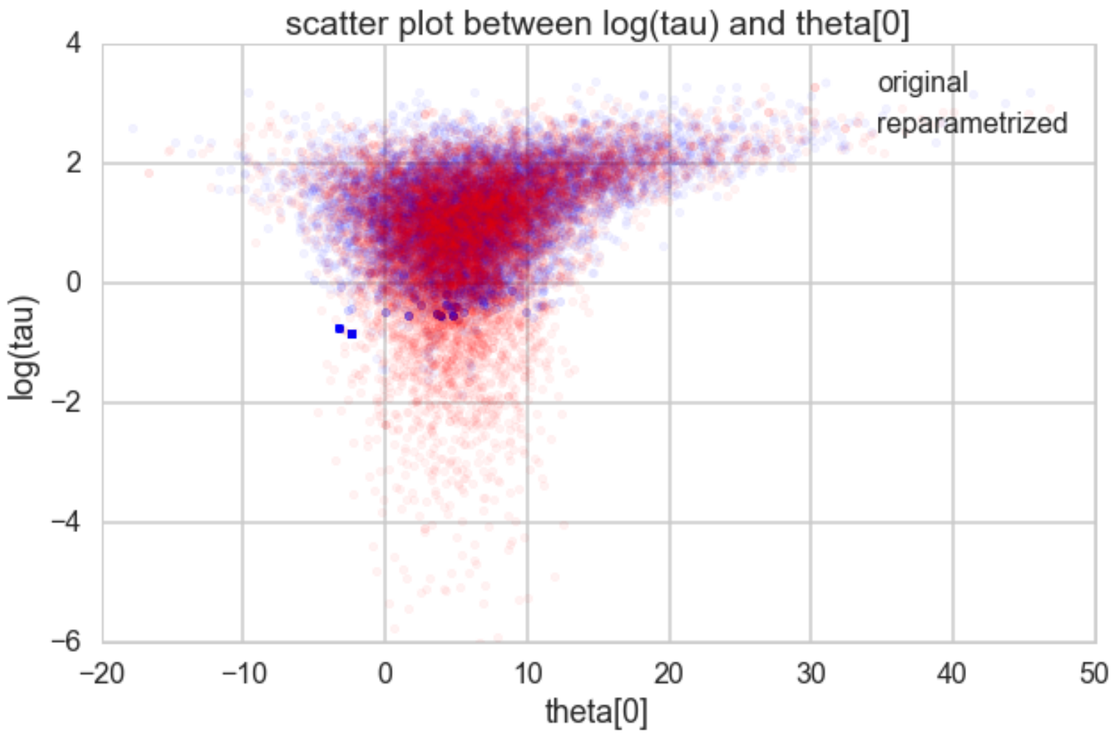
```
{'mu': 10000.0,
  'nu': array([ 10000.,  10000.,  10000.,  10000.,  10000.,  10000.,  10000.,
         10000.]),
  'tau': 6880.0,
  'tau_log_': 5193.0,
  'theta': array([  9624.,  10000.,  10000.,  10000.,  10000.,  10000.,  10000.,
          9829.])}
```

```
divergent = trace2['diverging']
print('Number of Divergent %d' % divergent.nonzero()[0].size)
divperc = divergent.nonzero()[0].size/len(trace2)
print('Percentage of Divergent %.5f' % divperc)
```

```
Number of Divergent 8
Percentage of Divergent 0.00160
```

Divergences and true length of funnel

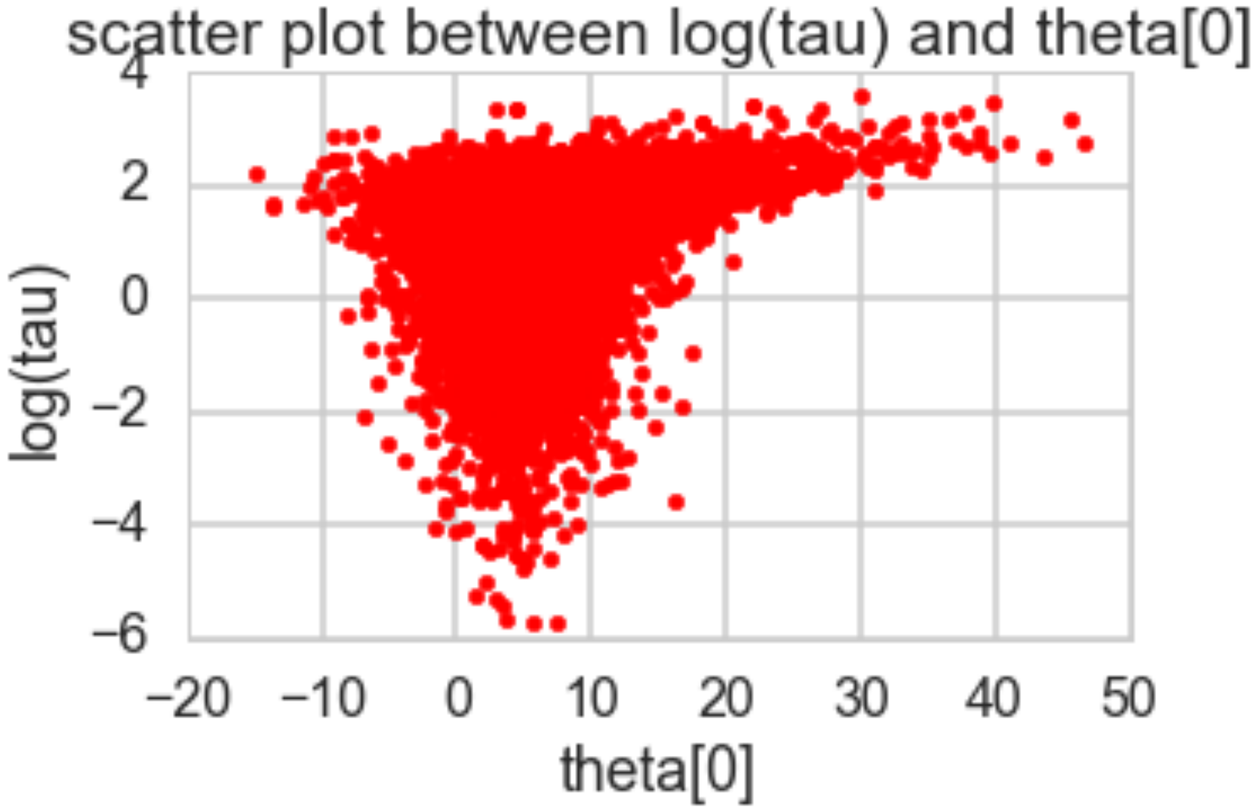


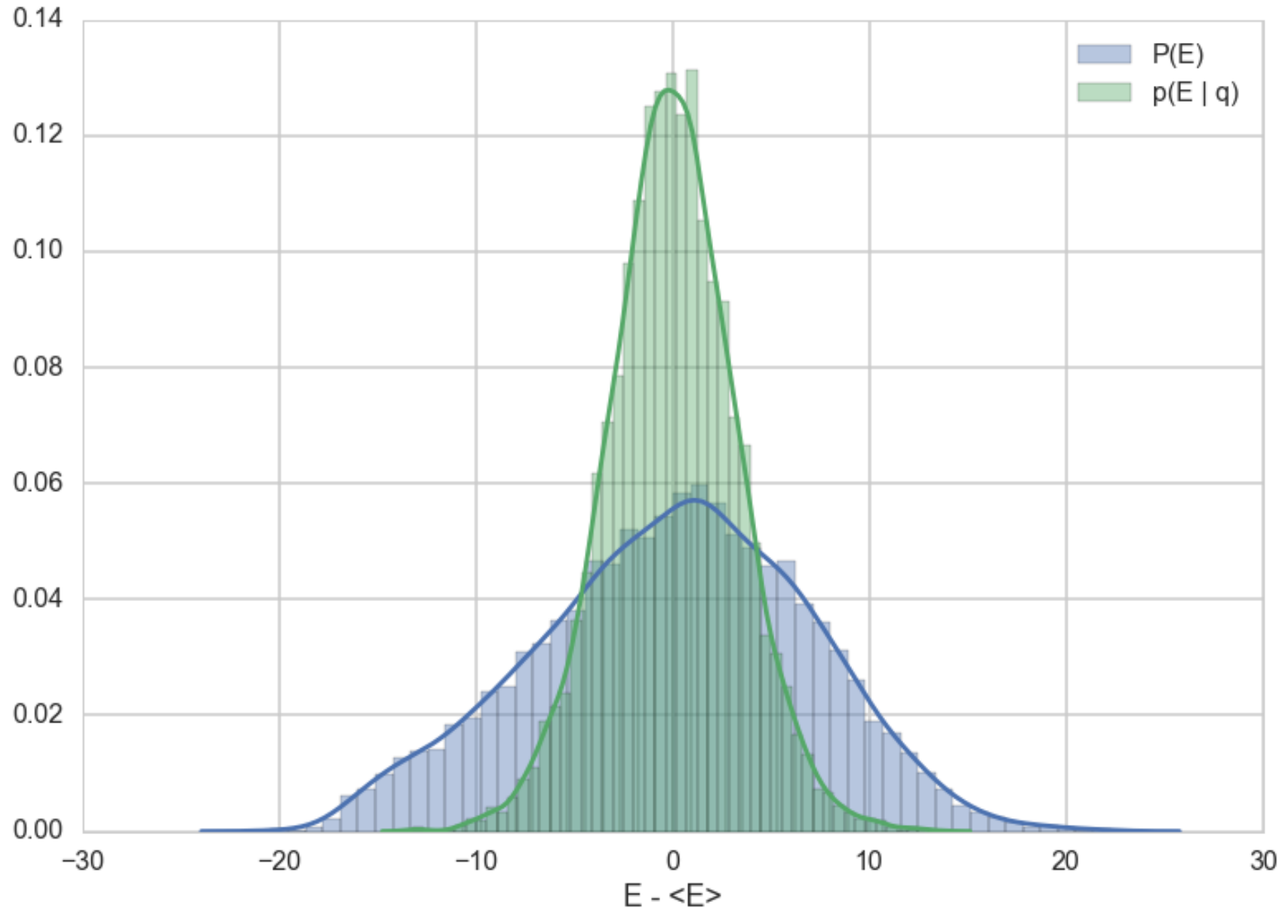


- Divergences infrequent, and all over. Mostly false positives.
- Lowering step sizes should make them go away

```
with schools2:
    step = pm.NUTS(target_accept=.95)
    trace2_95 = pm.sample(5000, step=step, init=None, njobs=2, tune=1000)
```

- lower curvature ensures geometric ergodicity deep in our funnel
- see Betancourt for big discussion





Momentum resampling Efficiency

- match transition $p(E|q)$ to marginal $p(E)$

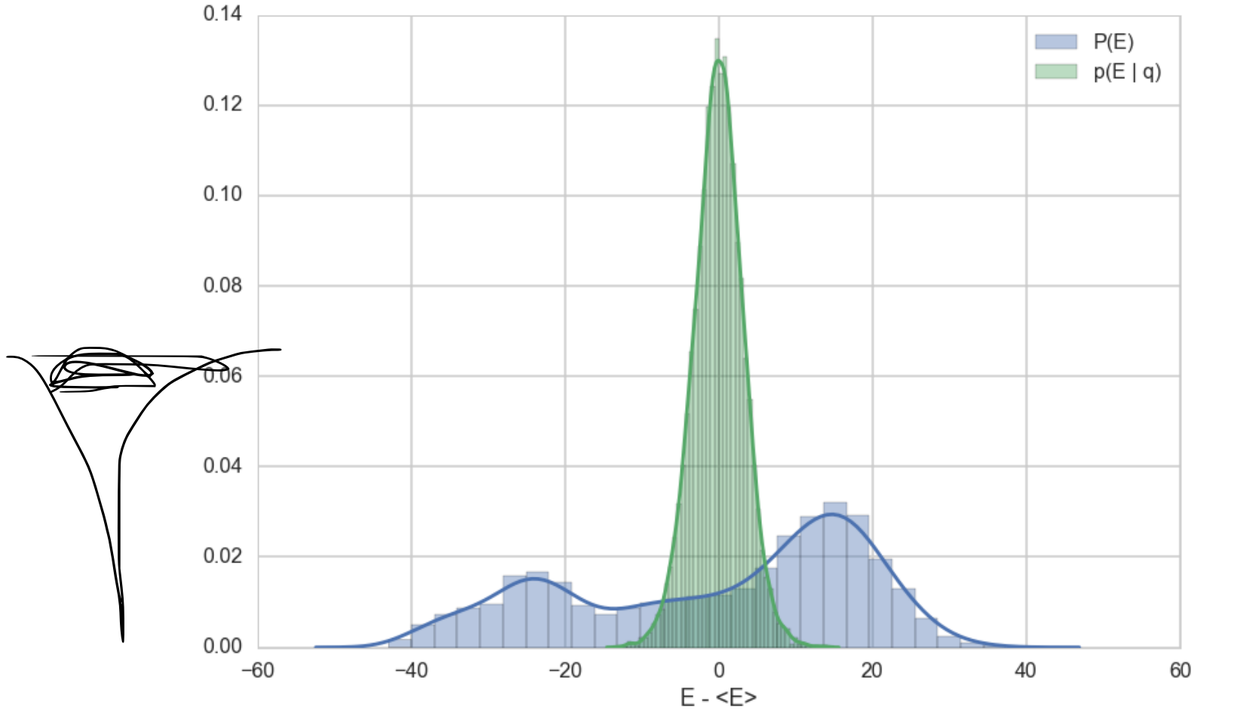
```
def resample_plot(t):
    sns.distplot(t['energy']-t['energy'].mean(), label="P(E)")
    sns.distplot(np.diff(t['energy']), label = "p(E | q)")
    plt.legend();
    plt.xlabel("E - <E>")
```

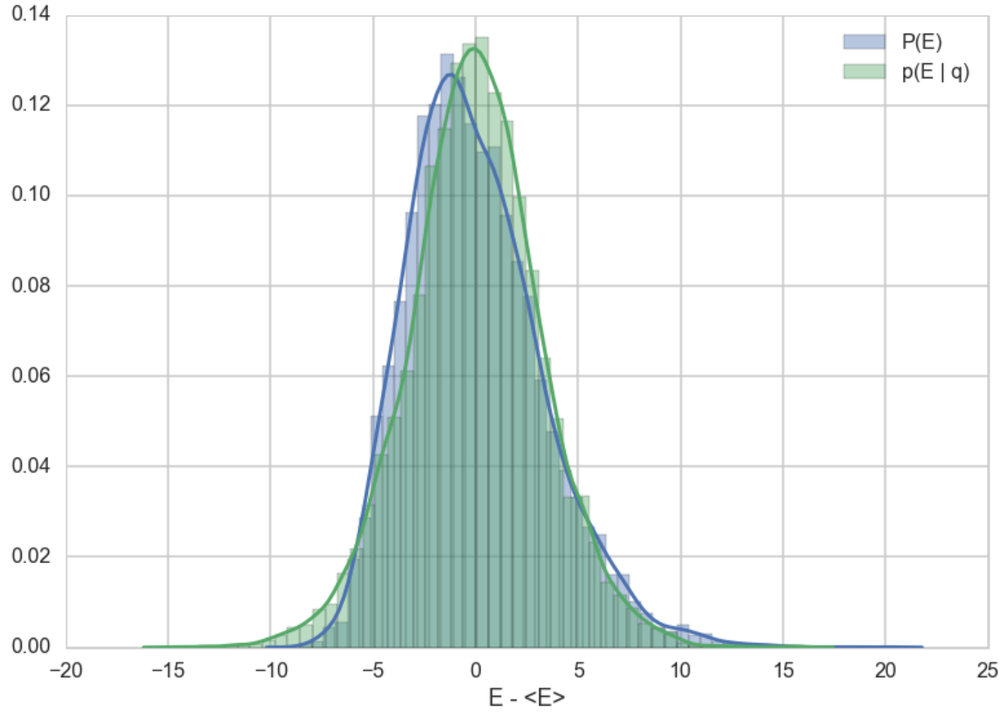
- if marginal has bigger tails we are in trouble
- indicative here of big energy changes in high-curvature regions not possible to boost to.

In a high curvature region:

centered, small step size vs Non-centered

If epsilon (step size) is large for the HMC sampler, then the sampler would not get the correct E's (energy levels; our target distribution)





large epsilon:
the sampler get stuck

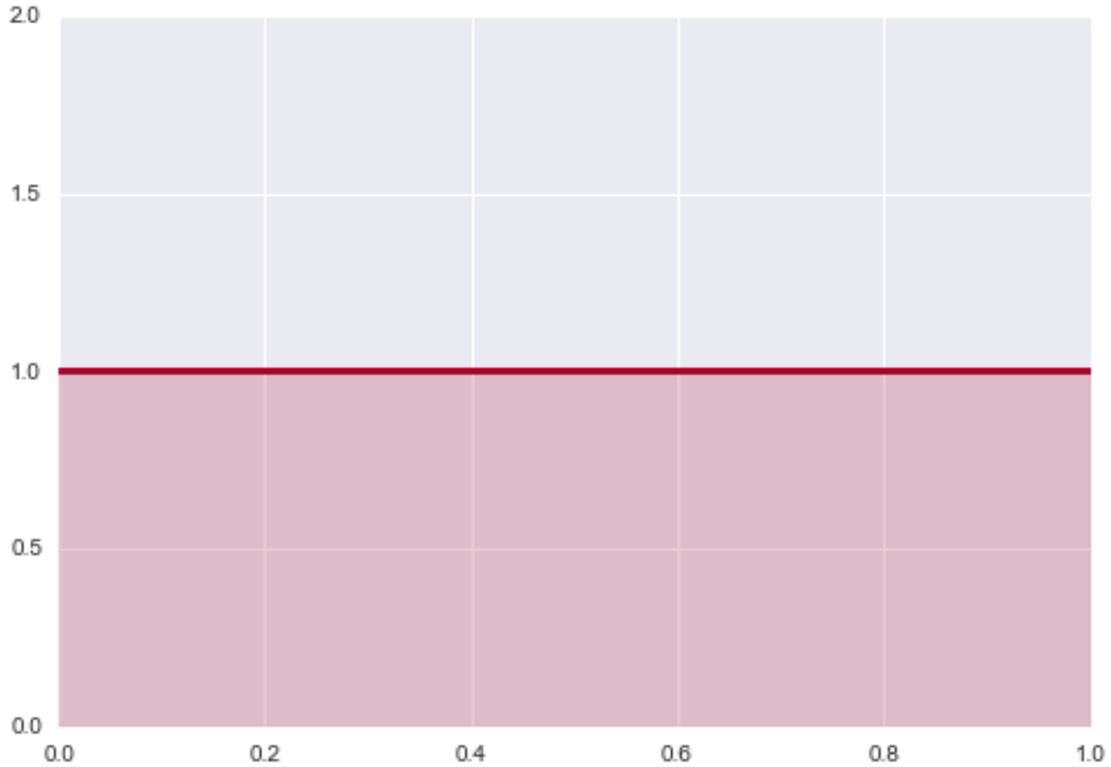
small epsilon:
more efficient exploration

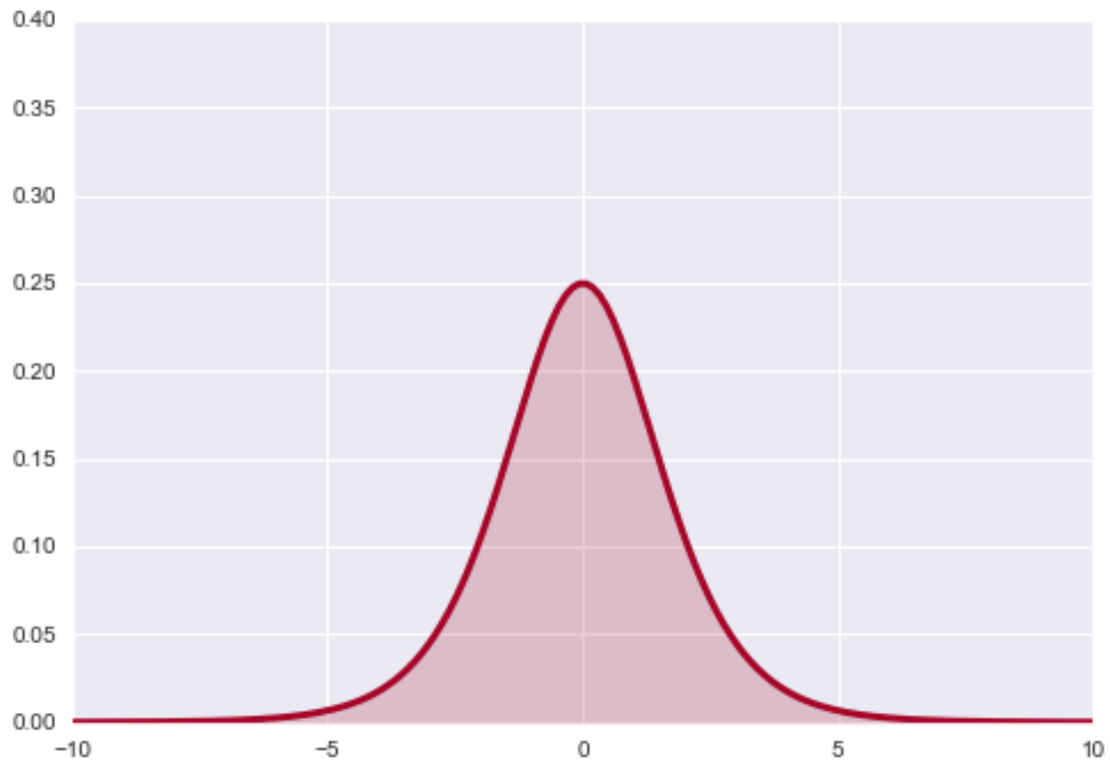
On left, centered, your sampler is not exploring, so make sure what you are diagnosing. On right, nice match!

What Priors?

- see https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations
- basic idea: choose something reasonable, and then spread it out some

Uninformative priors on location





- used transform $psi = log(\frac{\theta}{1-\theta})$ and then $dP_\psi = dP_\theta$. Shape comes in through jacobian.
- despite transformation change, flat priors still used for location priors
- may even be improper, ie integrate to ∞ as long as posterior integral is finite
- e.g. flat prior on mean in normal-normal model **with strong likelihood**.

Jeffreys prior

noninformative prior on scale variables $p_J(\theta) \propto \mathbf{I}(\theta)^{1/2}$

where

$$\mathbf{I}(\theta) = det(-E\left[\frac{d^2 \log p(X|\theta)}{d\theta_i \theta_j}\right])$$

is the Fisher Information, and expectation is with respect to the likelihood.

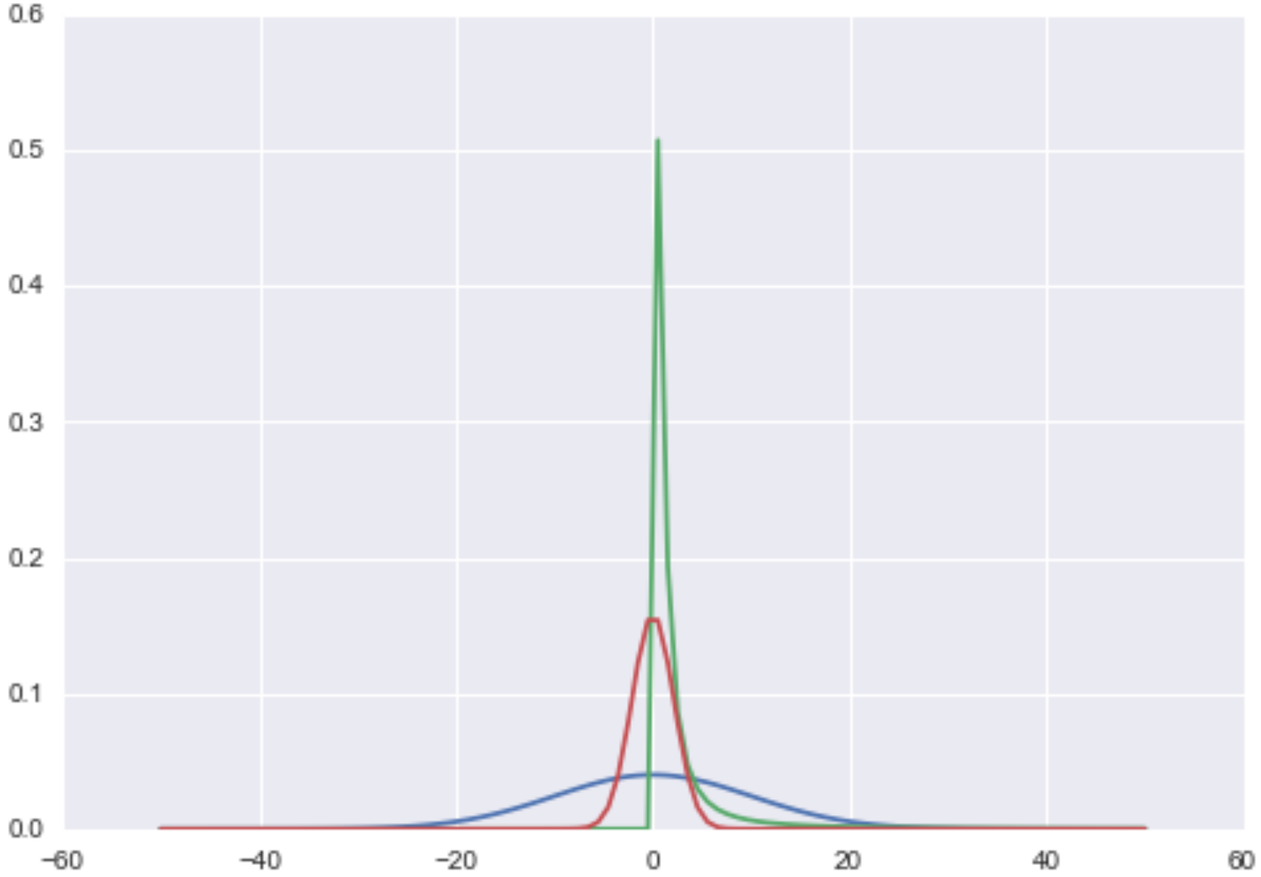
Weakly informative or regularizing priors

- these are the priors we will concern ourselves most with
- **restrict parameter ranges**
- **help samplers**
- regularizing priors may use the data "twice": hierarchical models. But guard against overfitting is being up in the hierarchy

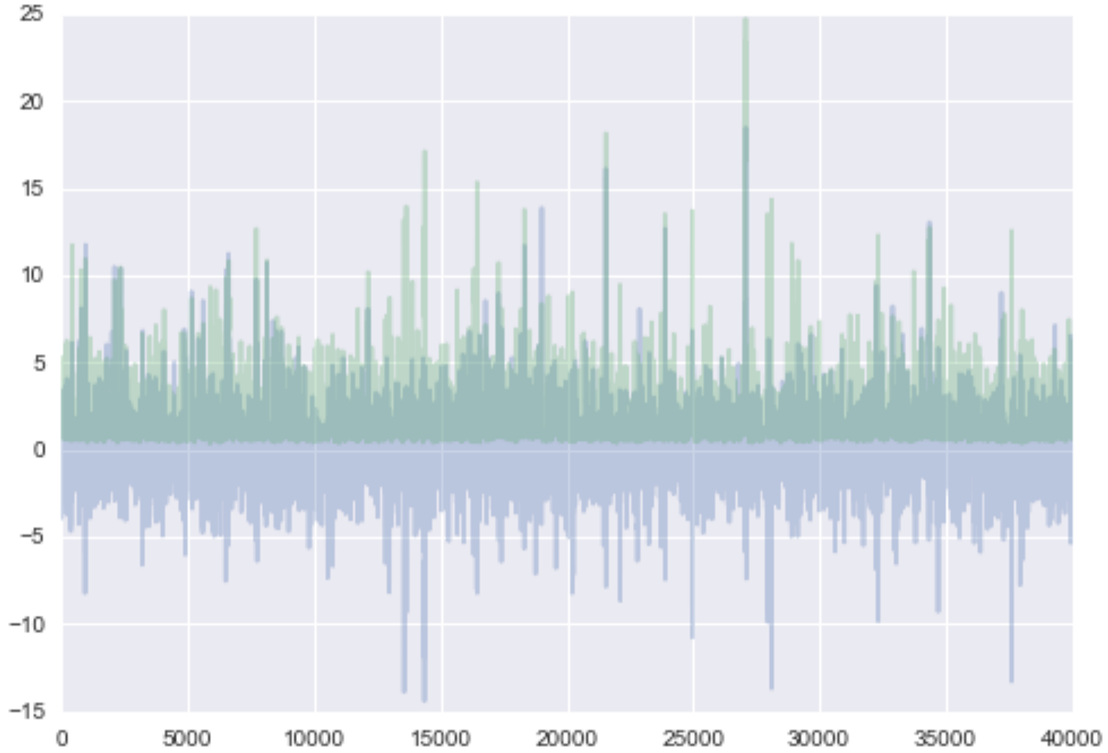
Normal model Example

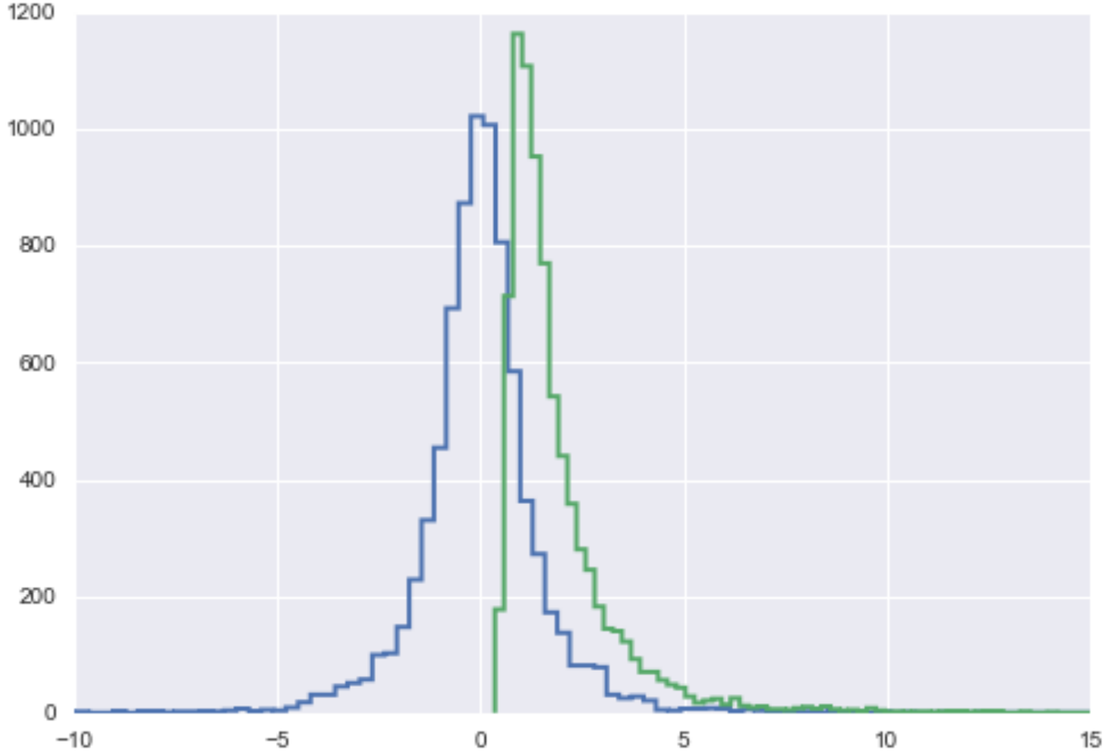
- two data points 1 and -1
- flat improper priors on $\mu, \sigma > 0$
- model drifts wildly as less data
- flat priors say extreme implausible values quite likely
- extreme drifts overwhelm chain

weakly regularizing priors



- choose $\mu \sim N(0, 10)$
- choose $\sigma \sim HalfCauchy(0, 1)$
- lets mean vary widely but not crazily
- HalfCauchy lets variance be positive and occasionally can have high value samples

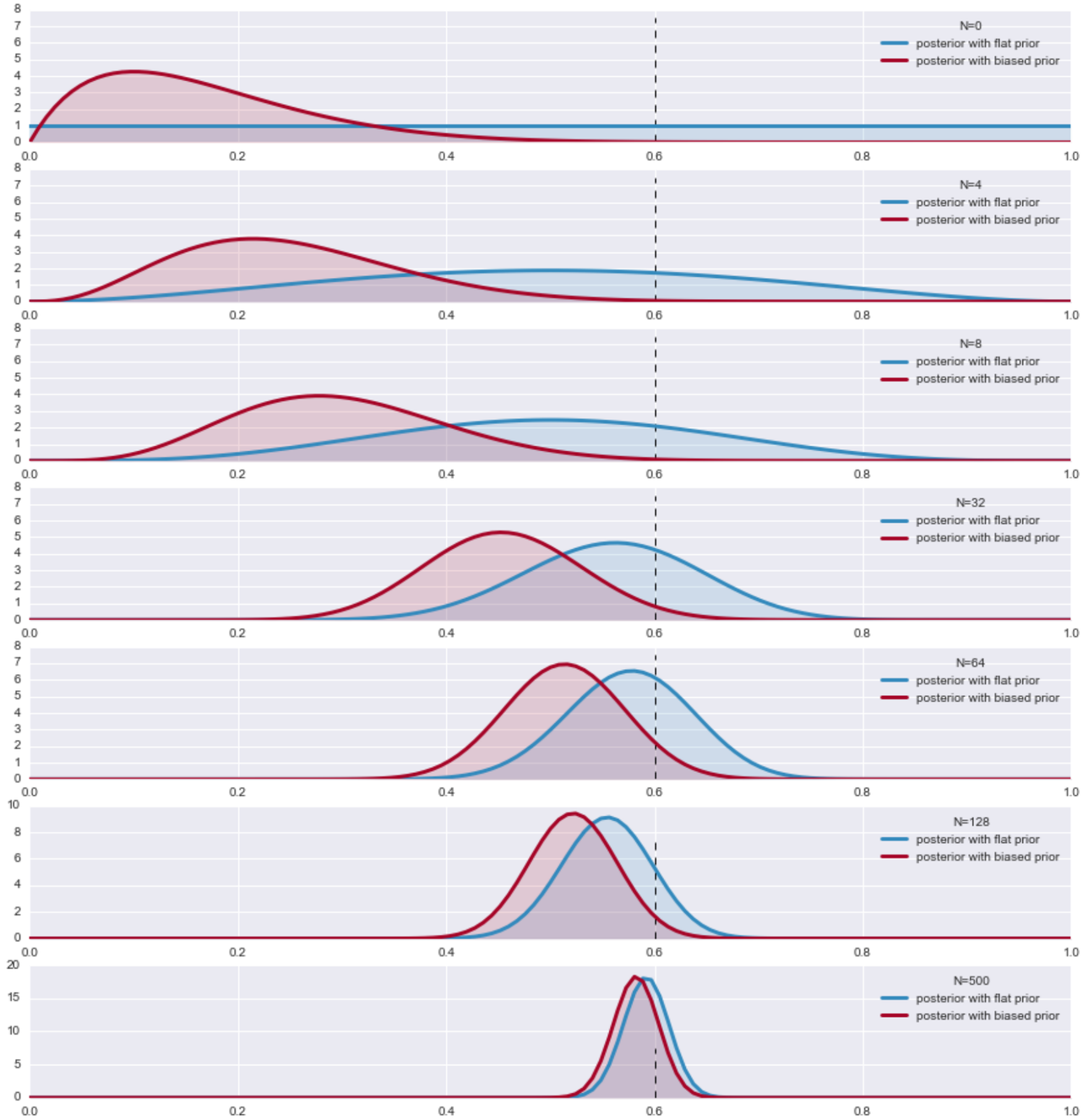

25
20
15
10
5
0
−5
−10
−15
0
5000
10000
15000
20000
25000
30000
35000
40000


1200
1000
800
600
400
200
0
−10
−5
0
5
10
15

Other priors

- KL Maximization non-informative prior by Bernardo
- Maximum Entropy prior when some assumptions but no more..
- Empirical bayes prior: uses data! in hierarchical models
- hyperparameter prior in hierarchcal models does not, we simply fit the whole model

Data overwhelms prior



Still:

- in the Bayesian Method we wish the prior to be weakly informative
- weakly means we ARE communicating some info to the prior from the data
- its mainly to ensure sensible values and help the sampler along
- and it provides some regulafrization in low data situations due to the graphical structure of our model
- we should do prior sensitivity and range analysis in our bayesian workflow

Levels of Bayes

Method	Definition
Maximum Likelihood	$\hat{\theta} = argmax_\theta p(D\|\theta)$
MAP estimation	$\hat{\theta} = argmax_\theta p(D\|\theta)p(\theta\|\eta)$
ML-2 (Empirical Bayes)	$\hat{\eta} = argmax_\eta \int d\theta\, p(D\|\theta)p(\theta\|\eta) = argmax_\eta p(D\|\eta)$
MAP-2	$\hat{\eta} = argmax_\eta \int d\theta\, p(D\|\theta)p(\theta\|\eta)p(\eta) = argmax_\eta p(D\|\eta)p(\eta)$
Full Bayes	$p(\theta, \eta\|D) \propto p(D\|\theta)p(\theta\|\eta)p(\eta)$