Lecture 23

Cross Validation and Bayesian Workflow

Last time:

- Deviance, WAIC
- Model Comparison vs ensembling
- Bayesian Model Averaging: pseudo-BMA vs stacking
- Oceanic Tools: ensembling "regularizes" counterfactuals

Today:

- Cross Validation
- What Priors?
- Bayesian Workflow

Deviance

$$D(q) = -\frac{N}{2} E_p[log(q)]$$

Using empirical distribution on sample (deviance is a stochastic quantity)

$$D(q) = -2 \sum_i log(q_i),$$

Bayesian deviance

$D(q) = -\frac{N}{2} E_p[log(pp(y))]$ posterior predictive for points y on the test set or future data

replace joint posterior predictive over new points y by product of marginals (exact if using point estimate):

ELPD: $\sum_i E_p[log(pp(y_i))]$

Since we do not know the true distribution p,

replace elpd: $\sum_i E_p[log(pp(y_i))]$

by the computed "log pointwise predictive density" (lppd) **in-sample**

$$\sum_j log\left\langle p(y_j|\theta)\right\rangle = \sum_j log\left(\frac{1}{S}\sum_s p(y_j|\theta_s)\right)$$

WAIC

$$WAIC = lppd + 2p_W$$

where

$$p_W = 2\sum_i (log(E_{post}[p(y_i|\theta)] - E_{post}[log(p(y_i|\theta))])$$

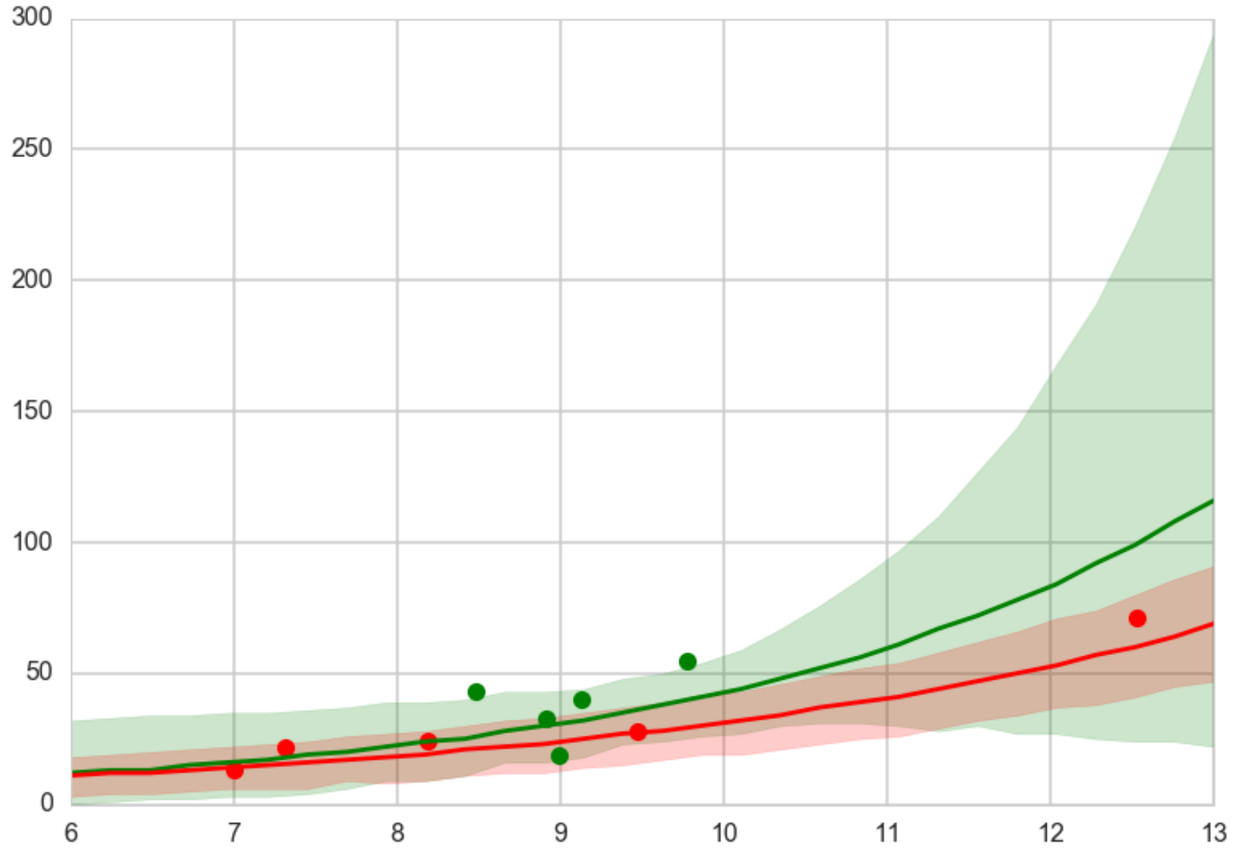
Once again this can be estimated by

$$\sum_i Var_{post}[log(p(y_i|\theta))]$$

it is tempting to use information criteria to compare models with different likelihood functions. Is a Gaussian or binomial better? Can't we just let WAIC sort it out?
Unfortunately, WAIC (or any other information criterion) cannot sort it out. The problem is that deviance is part normalizing constant. The constant affects the absolute magnitude of the deviance, but it doesn't affect fit to data.

--McElreath

Counterfactual Posterior predictive



Bayes Theorem in model space

$$p(M_k|D) \propto p(D|M_k)p(M_k)$$

where:

$$p(D|M_k) = \int d\theta_k \, p(y|\theta_k, M_k)p(\theta_k|M_k)$$

is the marginal likelihood under each model. Can use these "Bayes Factors" to compare but high sensitivity to prior.

Bayesian Model Averaging

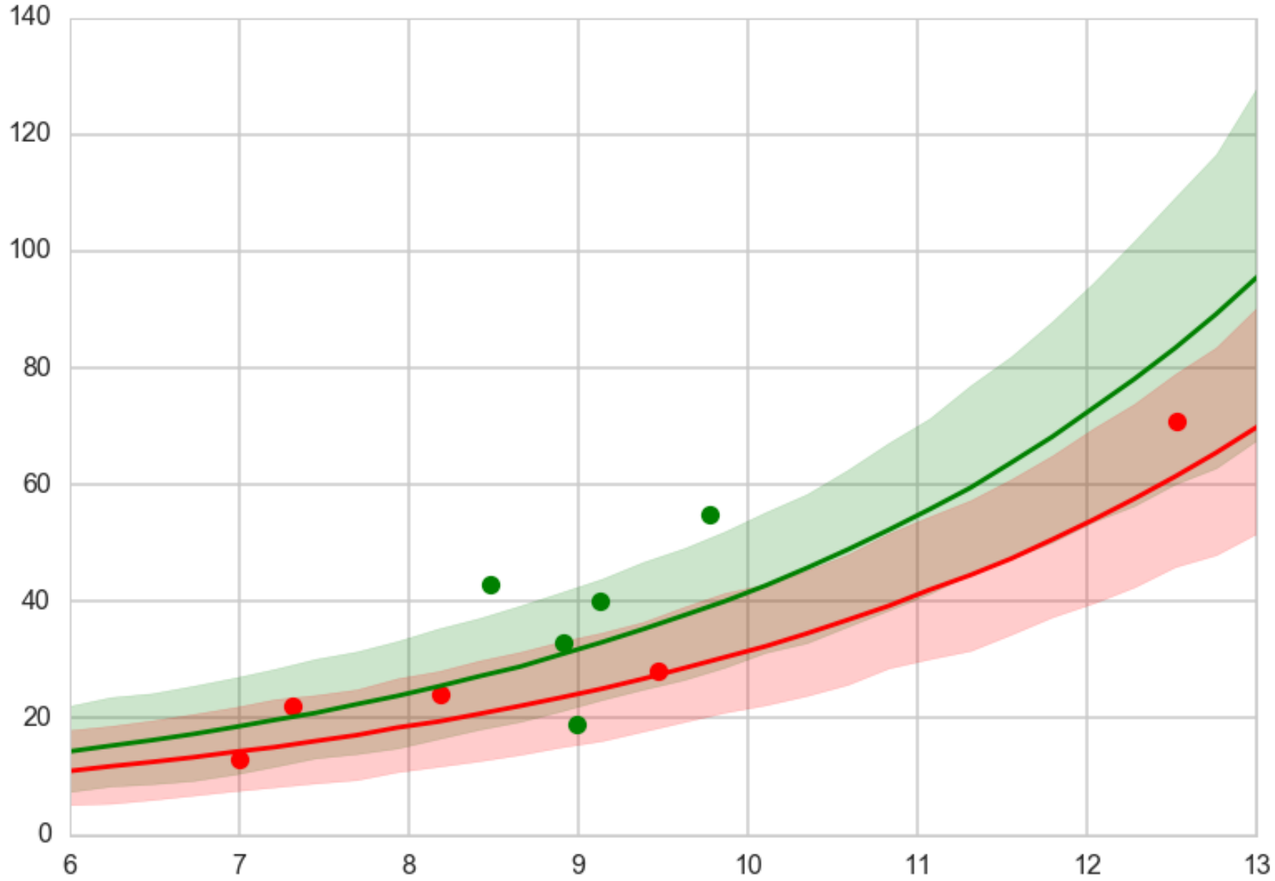
$$p_{BMA}(y^*|x^*, D) = \sum_k p(y^*|x^*, D, M_k)p(M_k|D)$$

where the averaging is with respect to weights $w_k = p(M_k|D)$, the posterior probabilities of the models M_k.

We will use the "Akaike" weights from the WAIC. This is called `pseudo-BMA`

Ensembling

- use WAIC based akaike weights for top 3
- regularizes down the green band at high population by giving more weight to the no-interaction model.



- BMA is appropriate in the M-closed case, which is when the true generating process is one of the models
- what we will use here is to estimate weights by the WAIC, following McElreath (pseudo-BMA)
- But see Yao et. al. which claims log-score stacking is better. Implemented in pymc3

$$\max_{w} \frac{1}{n} \sum_{i=1}^{n} \log \sum_{k=1}^{K} w_k p(y_i|y_{-i}, M_k), \quad s.t. \quad w_k \geq 0, \quad \sum_{k=1}^{K} w_k = 1.$$

Pseudo BMA vs stacking

name	WAIC	pWAIC	dWAIC	weight	SE	dSE	warning
m2c_nopc	79.06	4.24	0	0.87	11.06	0	1
m1c	84.09	7.05	5.04	0.07	12.19	3.77	1
m2c_onlyp	84.43	3.75	5.37	0.06	8.94	7.93	1
m2c_onlyic	141.65	8.38	62.6	0	31.7	32.84	1
m2c_onlyc	150.44	16.94	71.38	0	44.67	44.44	1

name	WAIC	pWAIC	dWAIC	weight	SE	dSE	warning
m2c_nopc	79.06	4.24	0	0.76	11.06	0	1
m1c	84.09	7.05	5.04	0	12.19	3.77	1
m2c_onlyp	84.43	3.75	5.37	0.24	8.94	7.93	1
m2c_onlyic	141.65	8.38	62.6	0	31.7	32.84	1
m2c_onlyc	150.44	16.94	71.38	0	44.67	44.44	1

...it is tempting to use information criteria to compare models with different likelihood functions. Is a Gaussian or binomial better? Can't we just let WAIC sort it out?

Unfortunately, WAIC (or any other information criterion) cannot sort it out. The problem is that deviance is part normalizing constant. The constant affects the absolute magnitude of the deviance, but it doesn't affect fit to data.

— McElreath

How to handle non-nested models?

- cross-validation
- less data to fit so biased models
- we are not talking here about cross-validation to do hyperparameter optimization
- specifically we will use Leave-One-Out-Cross-Validation (LOOCV) with importance sampling

LOOCV

- The idea here is that you fit a model on N-1 data points, and use the Nth point as a validation point. Clearly this can be done in N ways.
- the N-point and N-1 point posteriors are likely to be quite similar, and one can sample one from the other by using importance sampling.

$$E_f[h] = \frac{\sum_s w_s h_s}{\sum_s w_s} \text{ where } w_s = f_s/g_s.$$

Fit the full posterior once. Then we have

$$w_s = \frac{p(\theta_s|y_{-i})}{p(\theta_s|y)} \propto \frac{1}{p(y_i|\theta_s, y_{-i})}$$

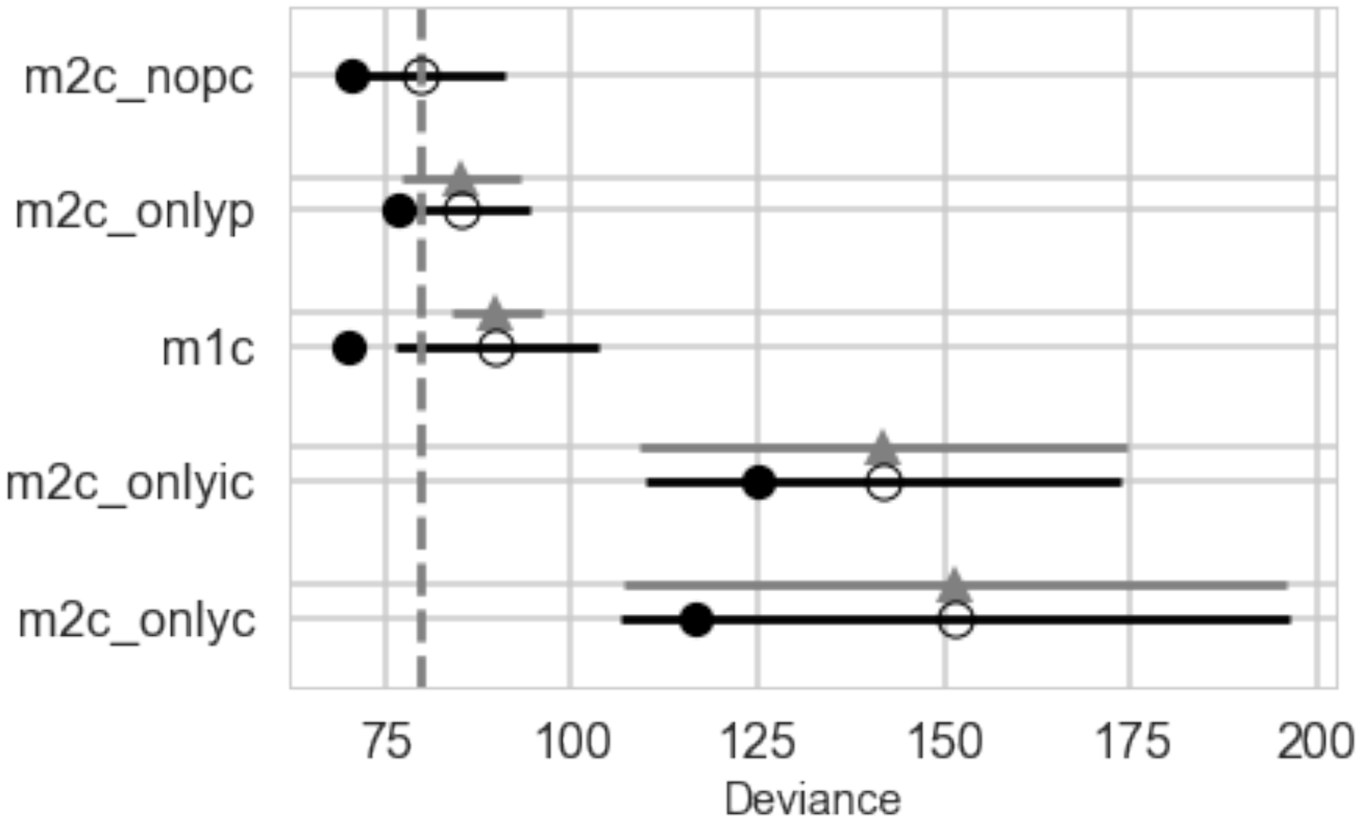
- the importance sampling weights can be unstable out in the tails.
- importance weights have a long right tail, pymc (`pm.loo`) fits a generalized pareto to the tail (largest 20% importance ratios) for each held out data point i (a MLE fit). This smooths out any large variations.

$$elpd_{loo} = \sum_i log(p(y_i|y_{-i}))$$

$$= \sum_i log\left(\frac{\sum_s w_{is} p(y_i|\theta_s)}{\sum_s w_{is}}\right)$$

over the training sample.

Oceanic tools LOOCV



What should you use?

1. LOOCV and WAIC are fine. The former can be used for models not having the same likelihood, the latter can be used with models having the same likelihood.

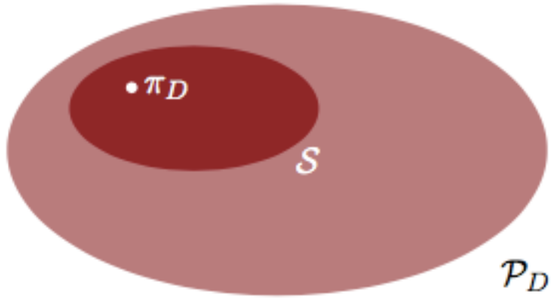
2. WAIC is fast and computationally less intensive, so for same-likelihood models (especially nested models where you are really performing feature selection), it is the first line of attack

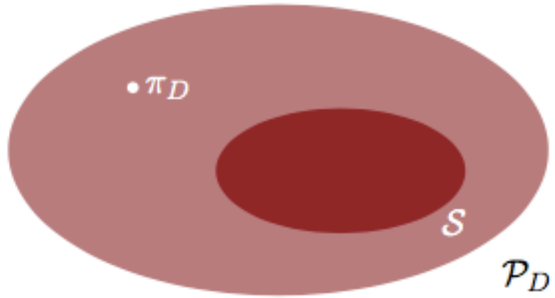
3. One does not always have to do model selection. Sometimes just do posterior predictive checks to see how the predictions are, and you might deem it fine.

4. For hierarchical models, WAIC is best for predictive performance within an existing cluster or group. Cross validation is best for new observations from new groups

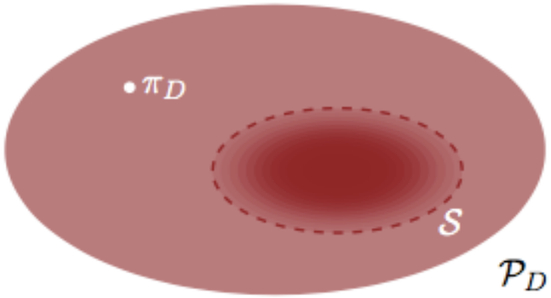
Bayesian Workflow

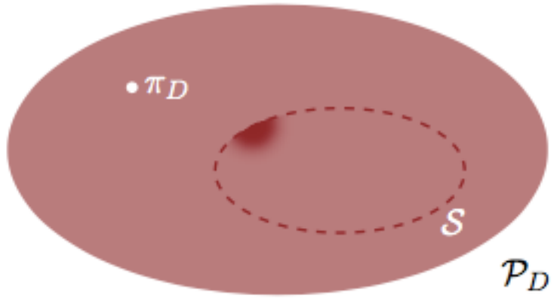
$$p(\theta|y) = \frac{p(\theta, y)}{p(y)}$$



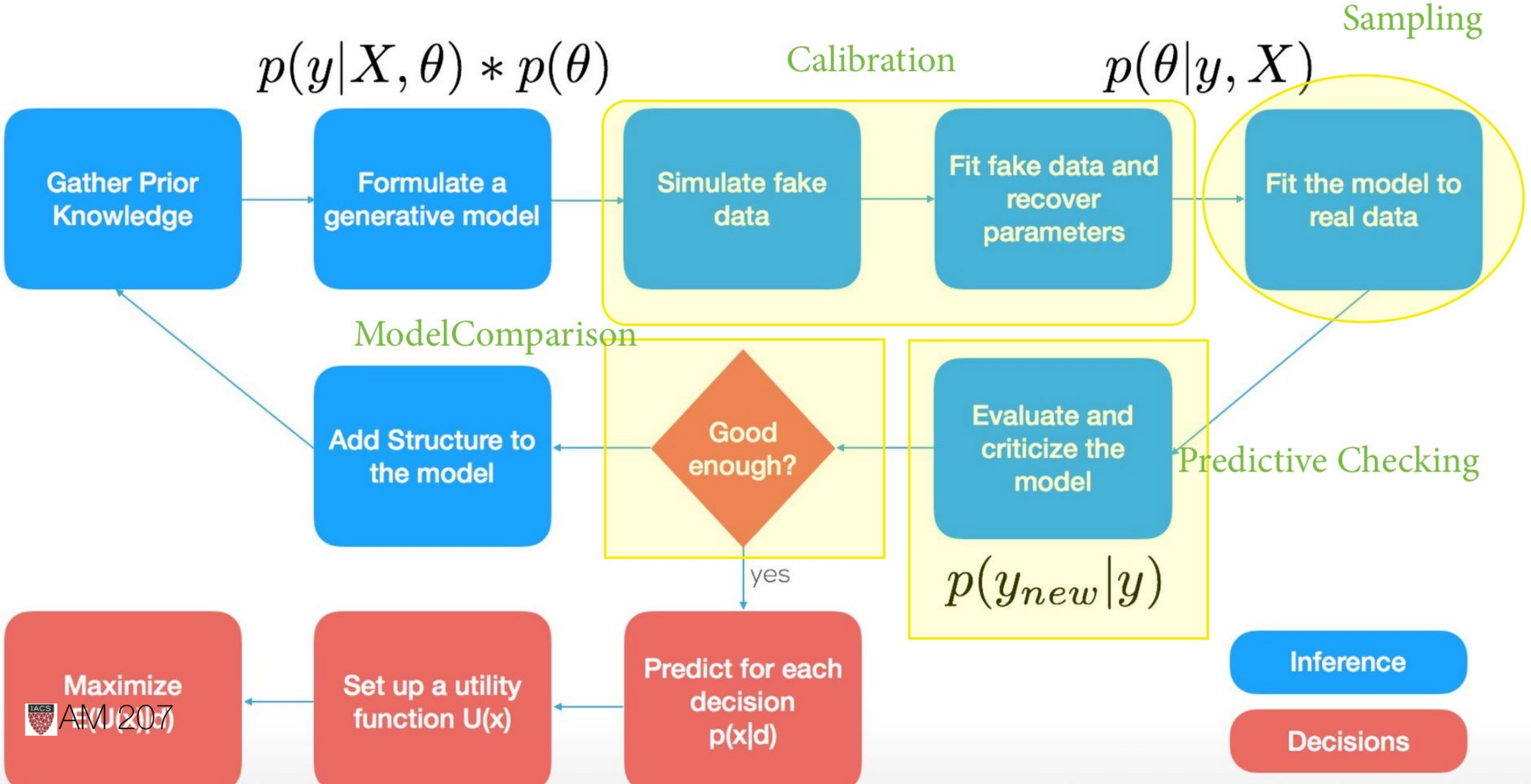


$$= \frac{p(y \mid \theta)p(\theta)}{p(y)}$$





Bayesian Workflow



Questions to answer

QA: *Domain Expertise Consistency*: Is our model consistent with our domain expertise?

QB: *Computational Faithfulness*: Are our computational tools sufficient to accurately fit the model?

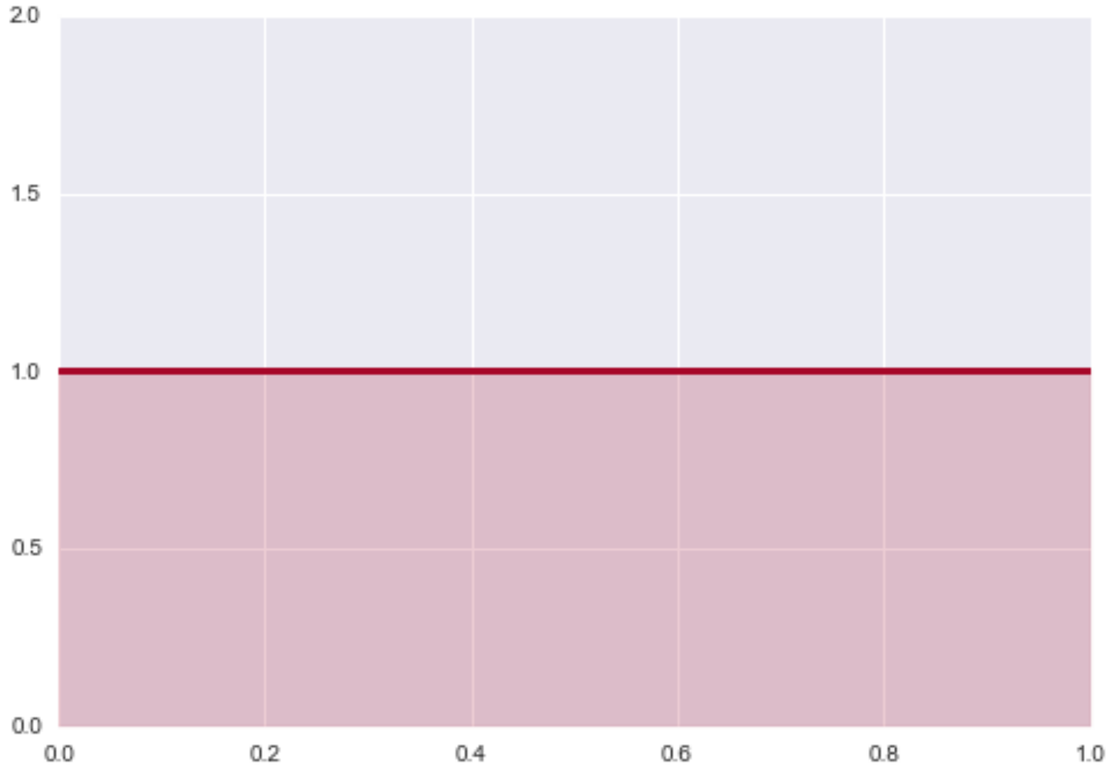
QC: *Model Sensitivity*: How do we expect our inferences to perform over the possible realizations of the measurement process?

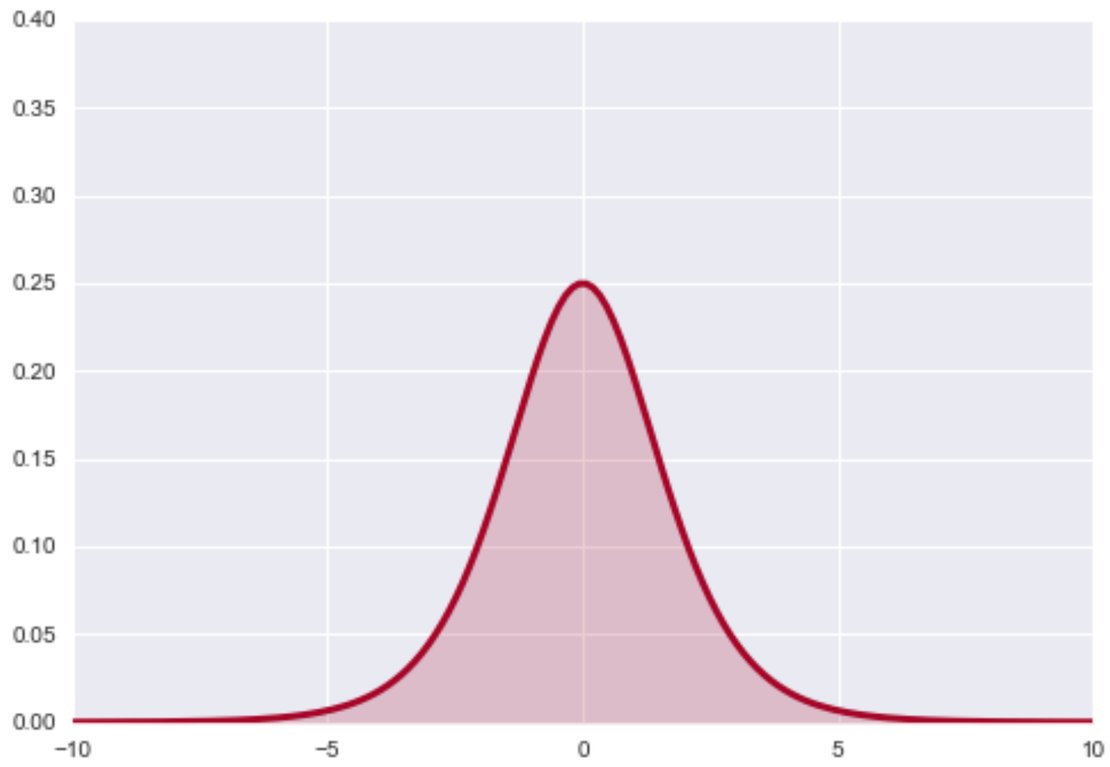
QD: *Model Adequacy*: Is our model rich enough to capture the relevant structure of the true data generating process?

What Priors? (QA and QC)

- we'll ask this question throughout the course
- also see https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations
- 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.

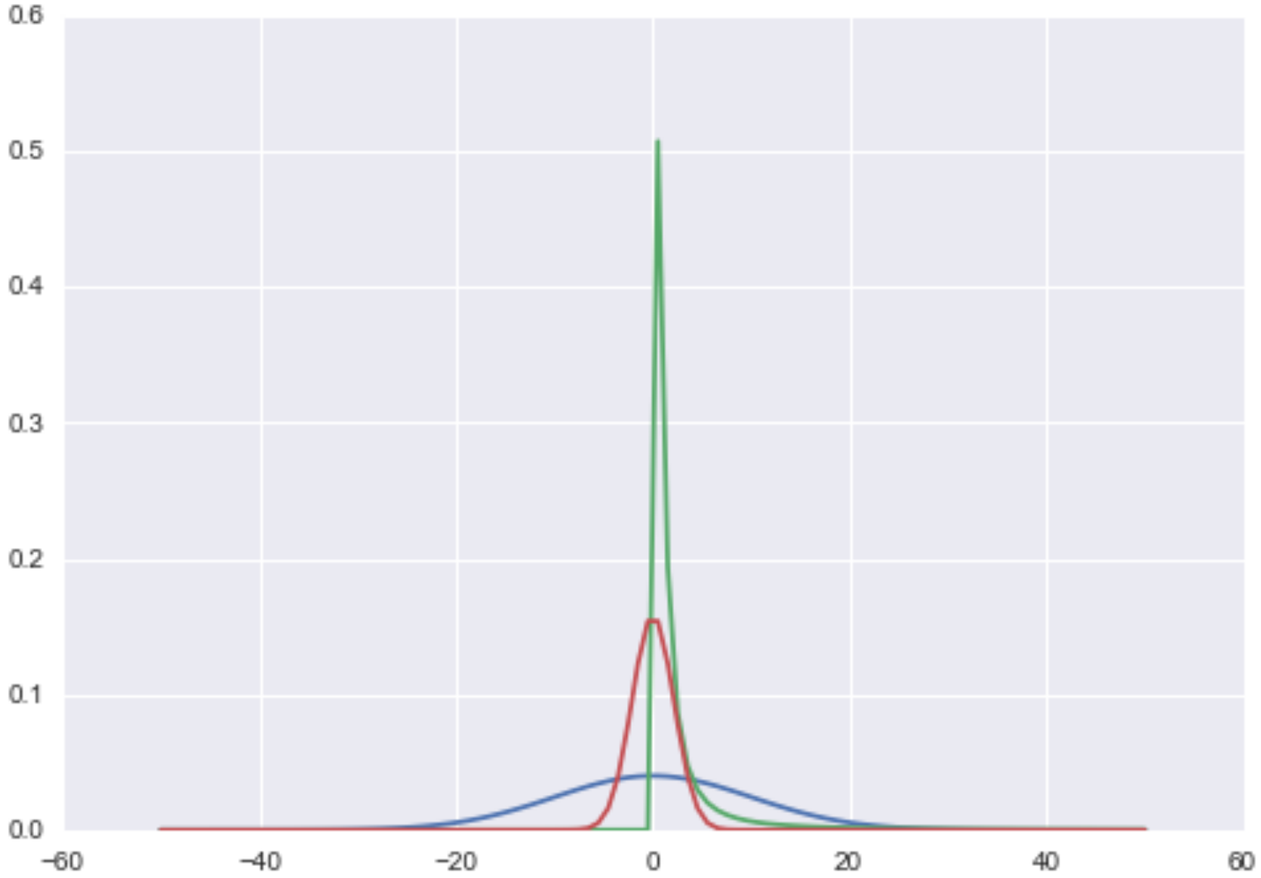
Weakly informative or regularizing priors

- these are the priors we will concern ourselves most with
- restrict parameter ranges
- help samplers
- regularizing priors may use the data "twice" as we shall see

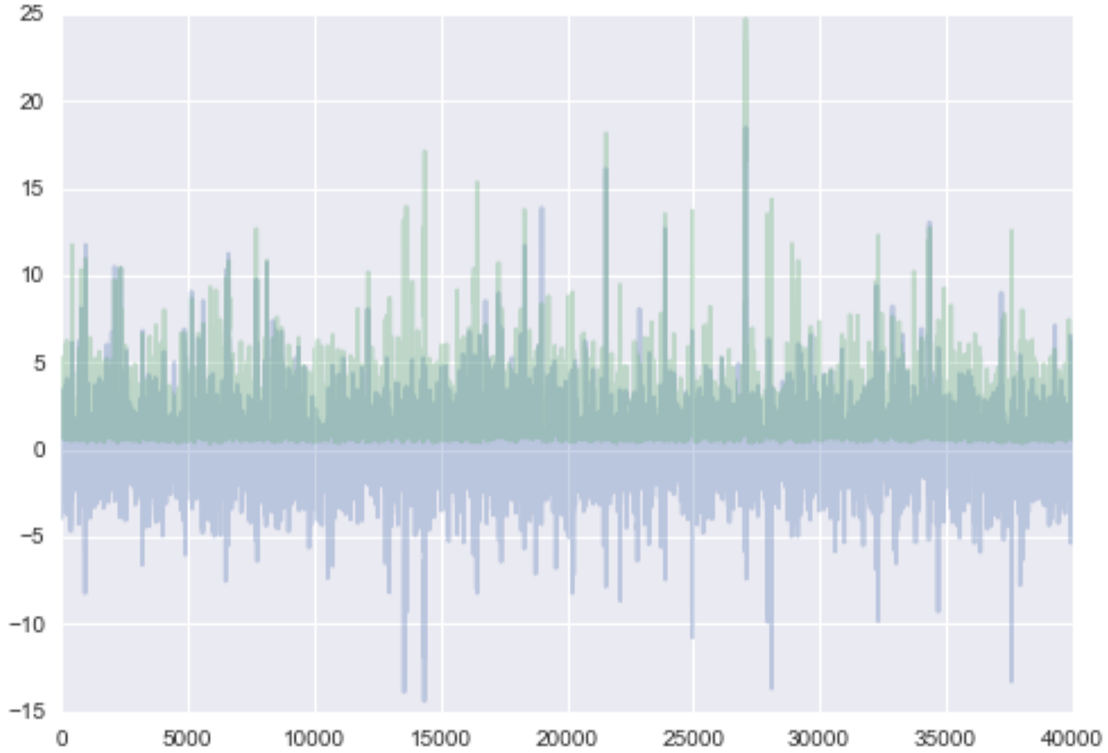
Normal model Example

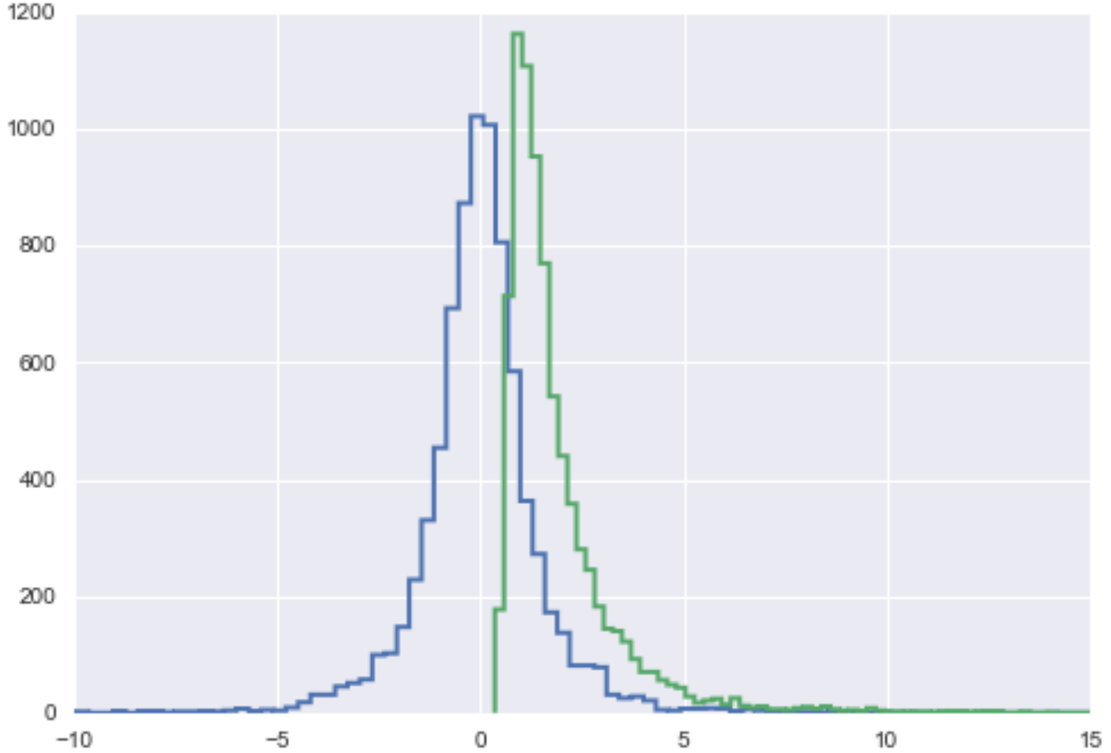
- two data points 1 and -1
- flat improper priors on $\mu, \sigma > 0$
- model drifts wildly as less data
- flat priors say extreme implausible values quite likely
- extreme drifts overwhelm chain

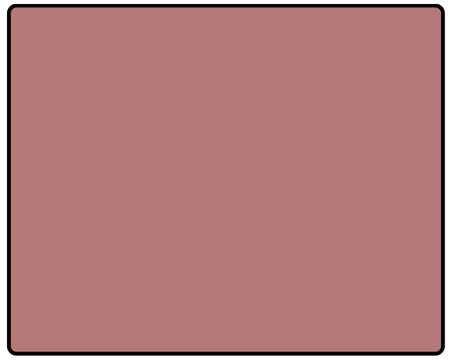
weakly regularizing priors



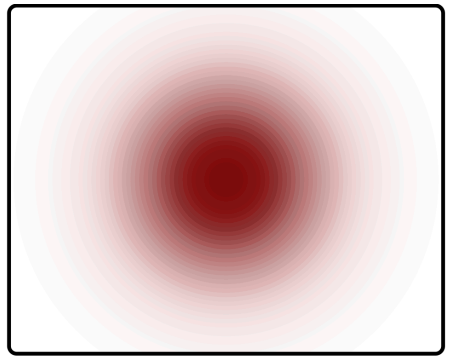
- choose $\mu \sum N(0, 10)$
- choose $\sigma \sim HalfCauchy(0, 1)$
- lets mean vary widely but not crazily
- HalfCauchy lets variance be positive and occasionally can have high value samples


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

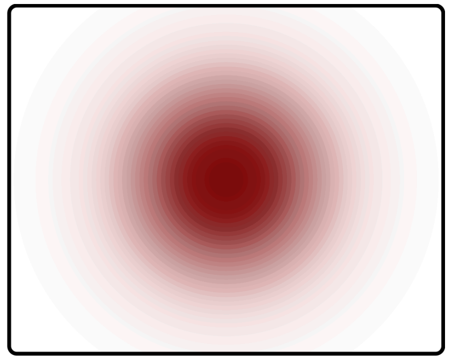

1200
1000
800
600
400
200
0
−10
−5
0
5
10
15



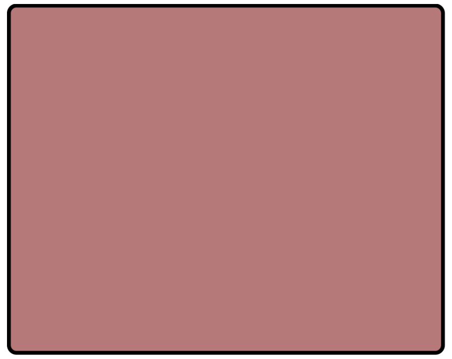
$\pi_{\mathcal{S}}(\theta)$



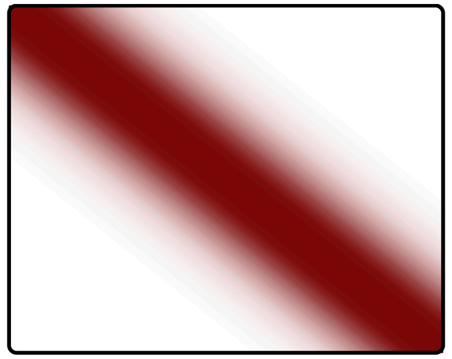
$\pi_{\mathcal{S}}(\tilde{y} \mid \theta)$



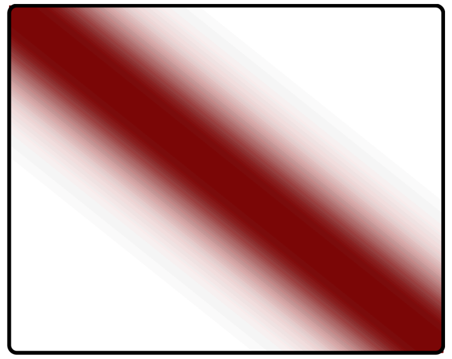
$\pi_{\mathcal{S}}(\theta \mid \tilde{y})$



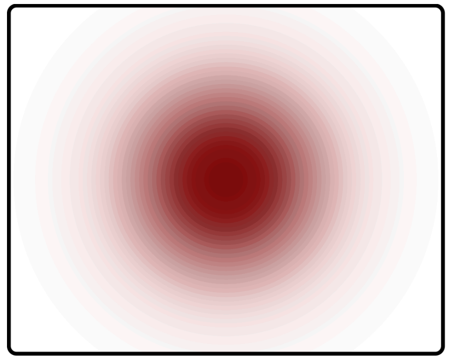
$\pi_{\mathcal{S}}(\theta)$



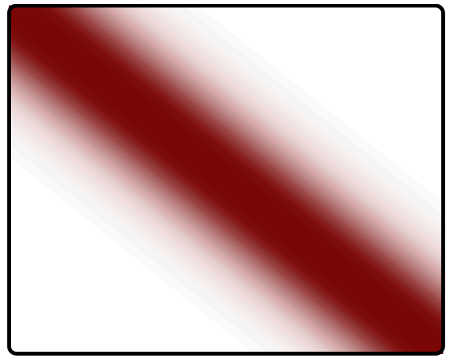
$\pi_{\mathcal{S}}(\tilde{y} \mid \theta)$



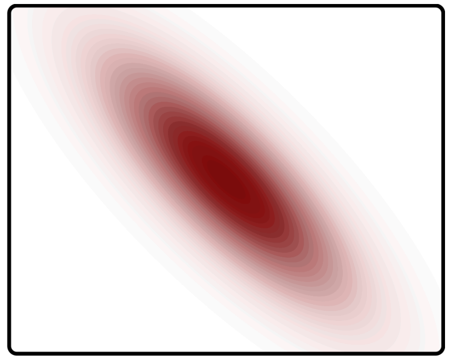
$\pi_{\mathcal{S}}(\theta \mid \tilde{y})$



$\pi_{\mathcal{S}}(\theta)$



$\pi_{\mathcal{S}}(\tilde{y} \mid \theta)$

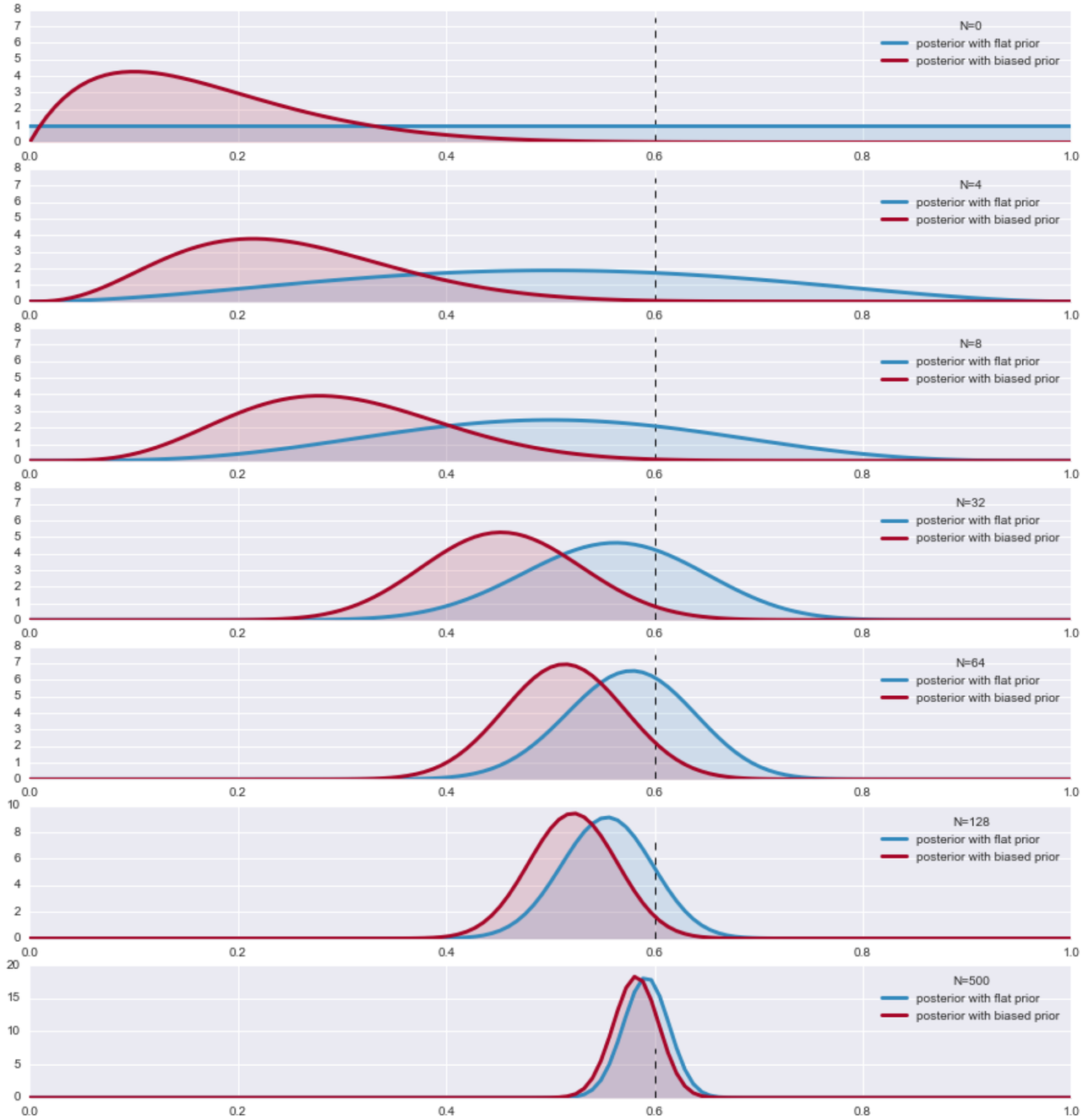


$\pi_{\mathcal{S}}(\theta \mid \tilde{y})$

Other priors

- KL Maximization non-informative prior by Bernardo
- Maximum Entropy prior when some assumptions but no more..
- Empirical bayes prior: usee data! in hierarchical models

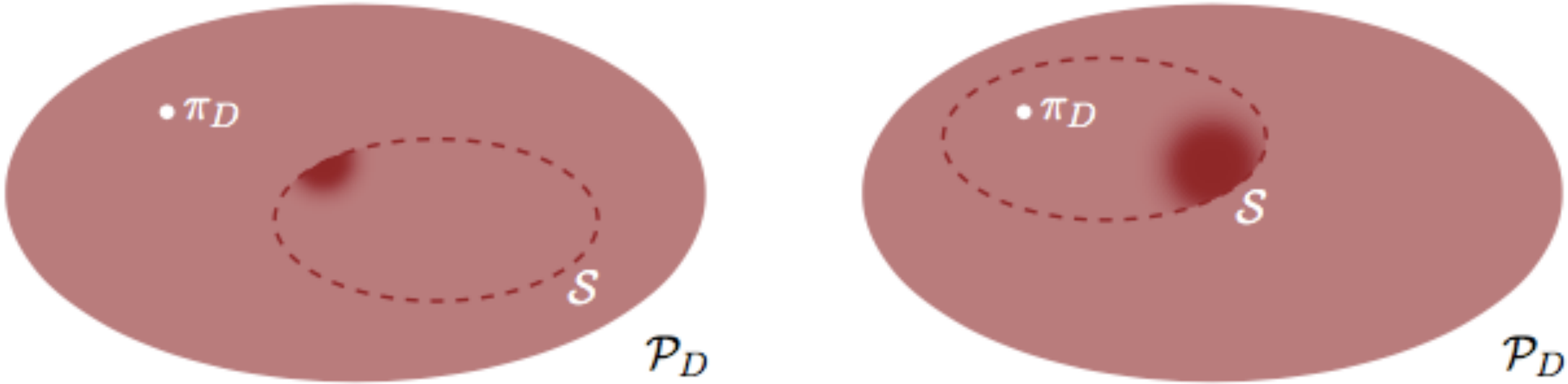
Data overwhelms prior



How to choose priors?

- mild regularization
- un-informativity
- sensible parameter space
- should correspond to scales and units of process being modeled
- we should calibrate to them

Think of the prior generatively AND predictively

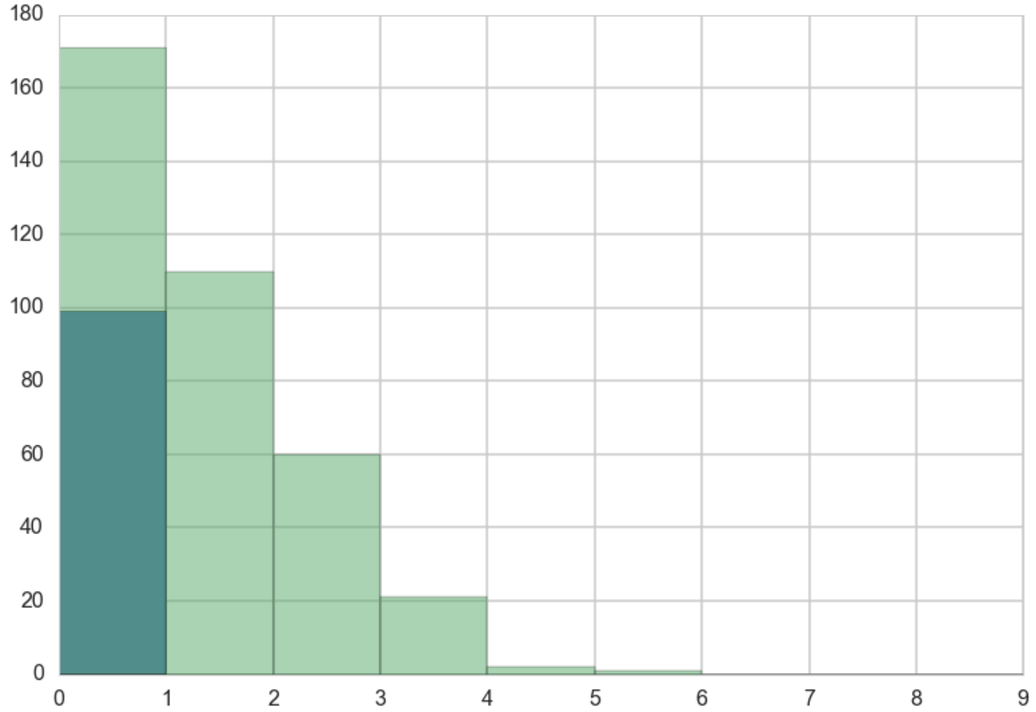


Bias can come from a prior, but do not construct a prior to allow for overfitting (draws far away from good place). Too many heavy tails can be bad.

Drunk monks writing manuscripts

- (A) Monks take a break on some days, drink, produce no manuscripts
- (B) looks the same like other unproductive days
- (B) some days are productive and produce manuscripts
- a mixture of (A) and (B)

Data



0 manuscripts can come from both driniking and slacking...

Poisson model

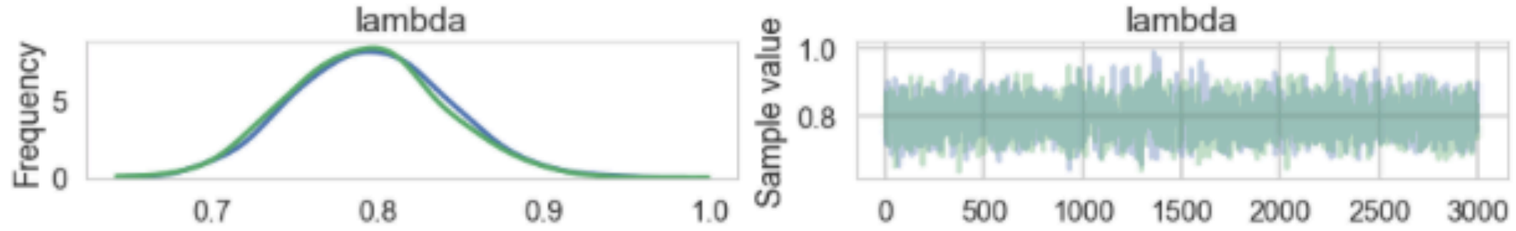
$$y \sim Poisson(\lambda)$$
$$\lambda \sim HN(100)$$

λ constrained to be positive.

```
with pm.Model() as model:
        lam=pm.HalfNormal("lambda", 100)
        like = pm.Poisson("obsv", mu=lam, observed=observed)
```

So far: just work on answer

```
Out[64]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x12456cd68>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x1263975f8>]],
               dtype=object)
```



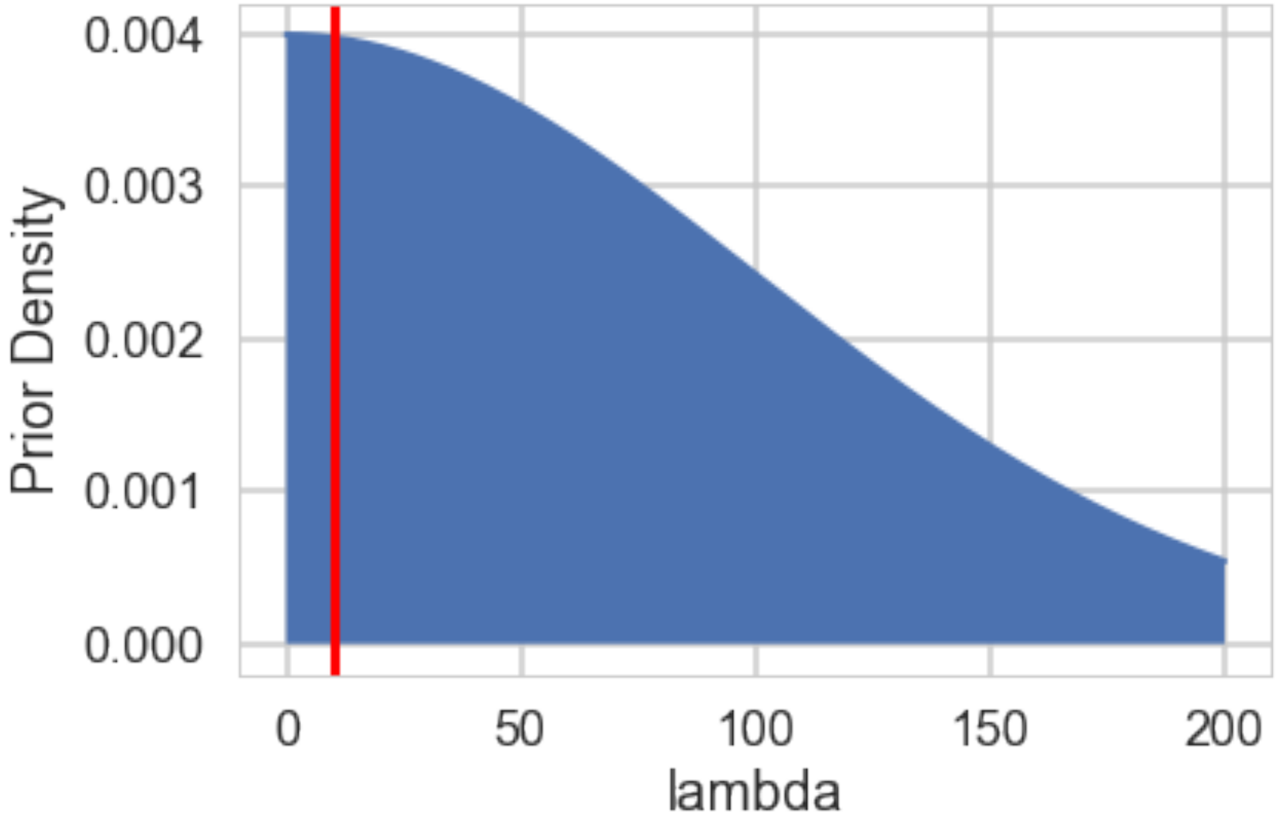

```
In [65]: pm.summary(trace0)
```

Out[65]:

	mean	sd	mc_error	hpd_2.5	hpd_97.5	n_eff	Rhat
lambda	0.794177	0.046064	0.001038	0.704709	0.884261	2167.755813	1.001699

Still needs a posterior predictive check to answer QD

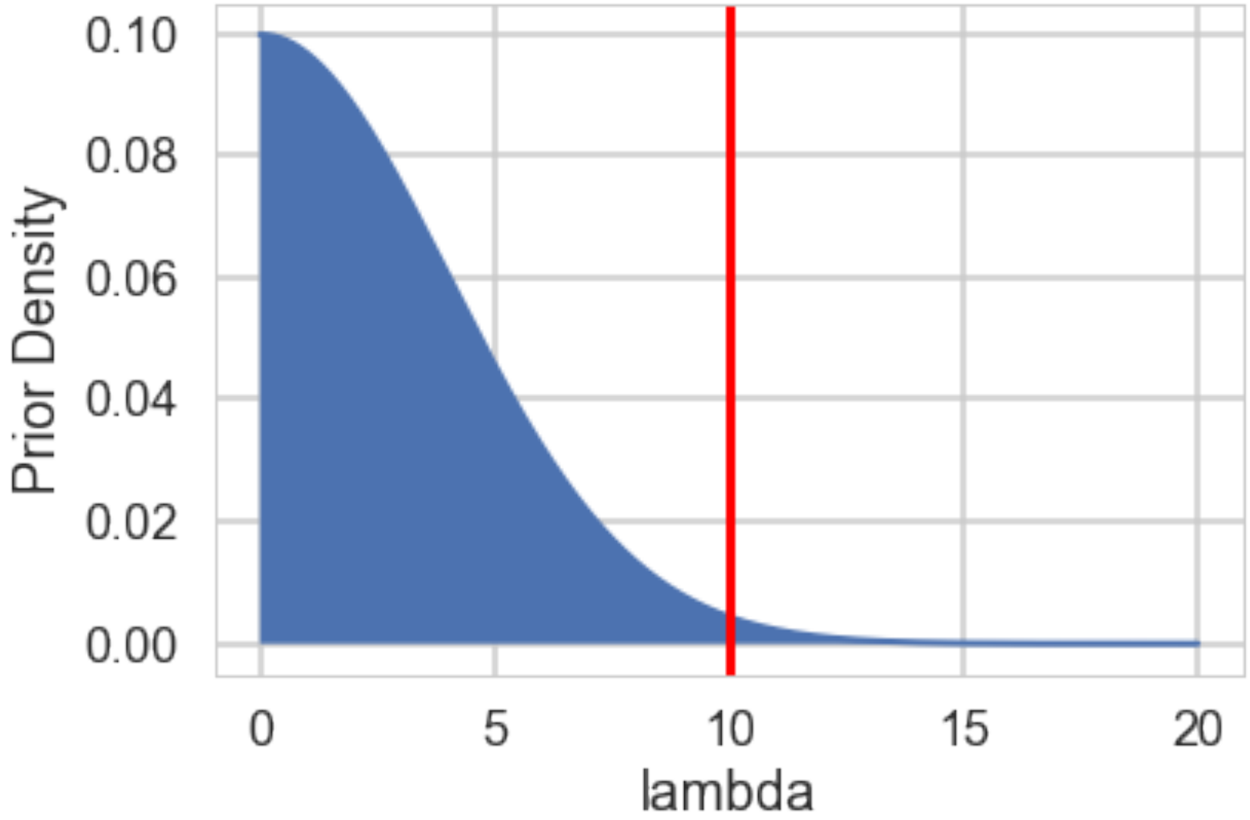
QA: Our prior so far:



Drunk Monks: QA, prior selection

- specify $\lambda \sim HalfN(0, 4)$ instead of the crazy $HalfN(0, 100)$ we had earlier
- domain knowledge: *A survey of Abbey Heads has told us, that the most a monk could produce, ever, was 10 manuscripts in a day.*
- $max(\lambda + 3\sqrt{\lambda}) < 10$, `5+3*np.sqrt(5)=11.7`
- `halfnorm.ppf(0.99, loc=0,scale=4)=10.3`

QA: Our new prior



QB, QC: Model Calibration

Think about the **Bayesian Joint distribution**.

$$p(\theta, y) = p(y \mid \theta)p(\theta)$$

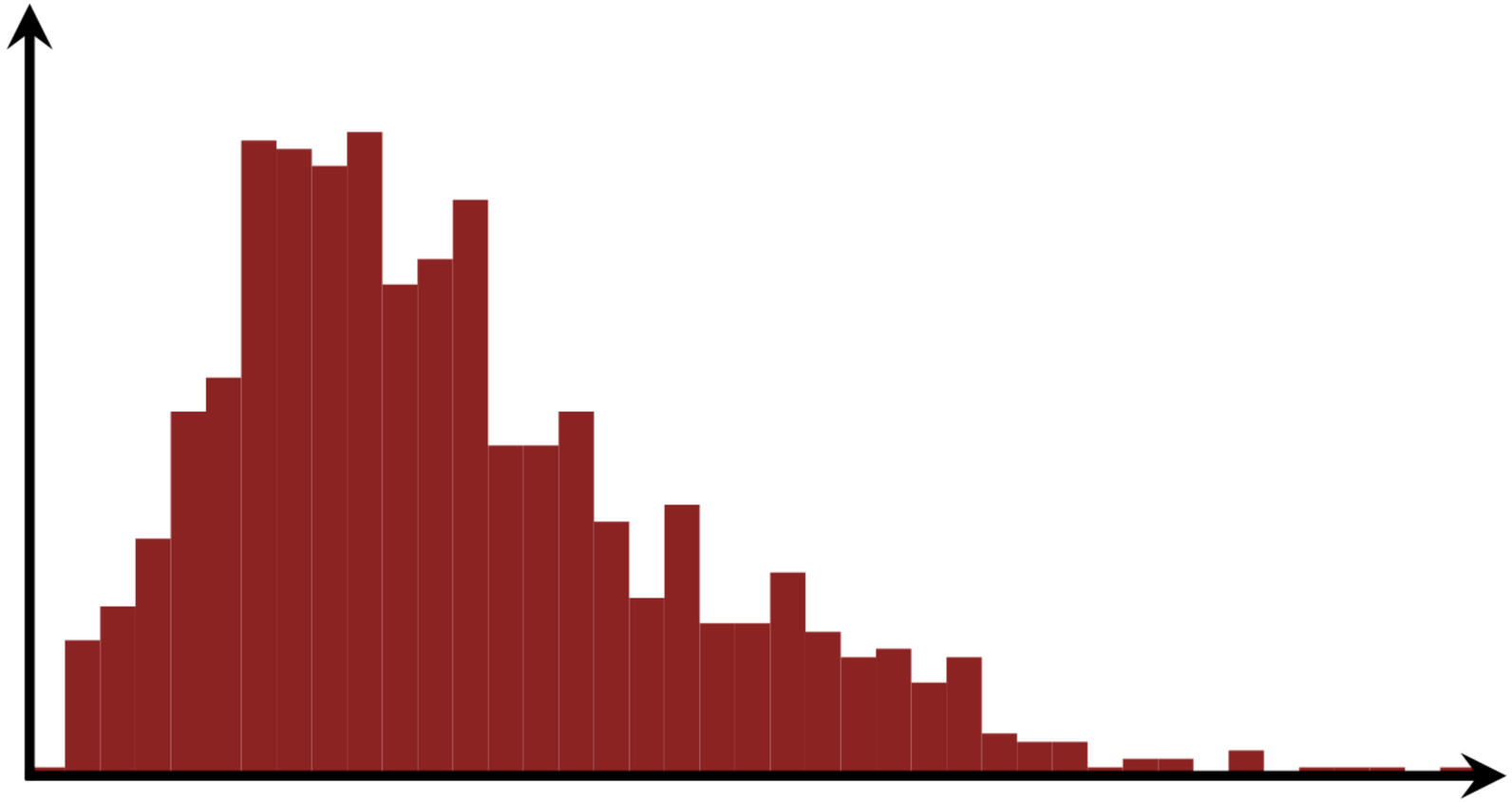
The prior predictive:

$$p(y) = \int d\theta \, p(\theta, y) = \int d\theta \, p(y \mid \theta)p(\theta)$$

Generate Artificial data sets

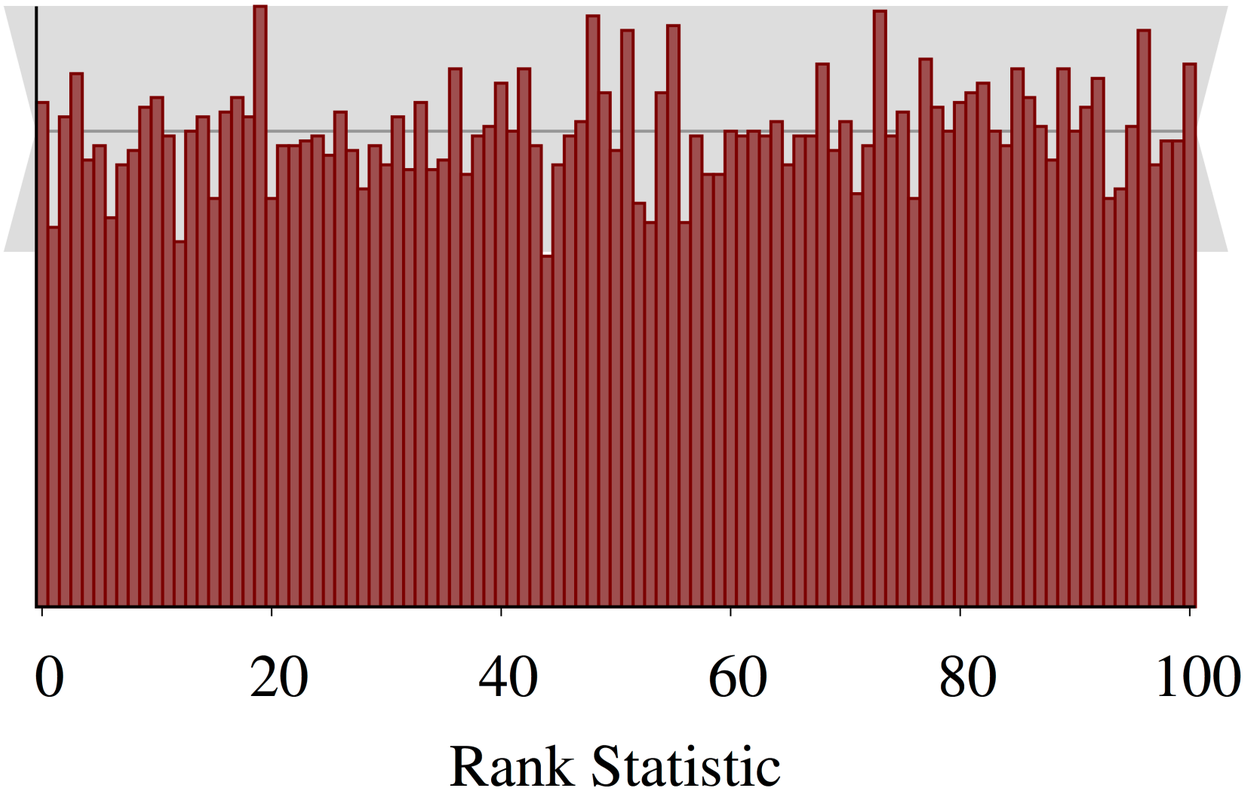
- from fixed params, but even better, from priors
- $\tilde{\theta} \sim p(\theta)$
- $\tilde{y} \sim p(y \mid \tilde{\theta})$
- callibrate inferences or decisions by analysing this data
- $U(a) = \int d\tilde{\theta} d\tilde{y} p(\tilde{y}, \tilde{\theta}) U(a(\tilde{y}), \tilde{\theta})$

$\pi_{U(A,\mathcal{S})}(U)$



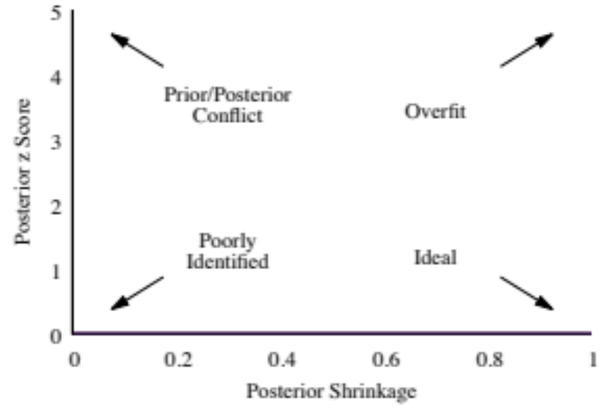
$U(a(y), \theta)$

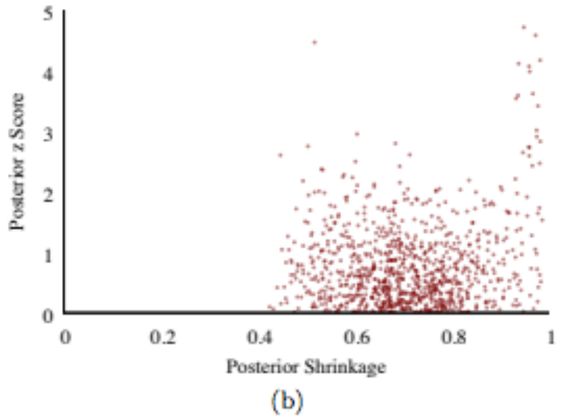
QB: General Metrics: Simulation Based Callibration

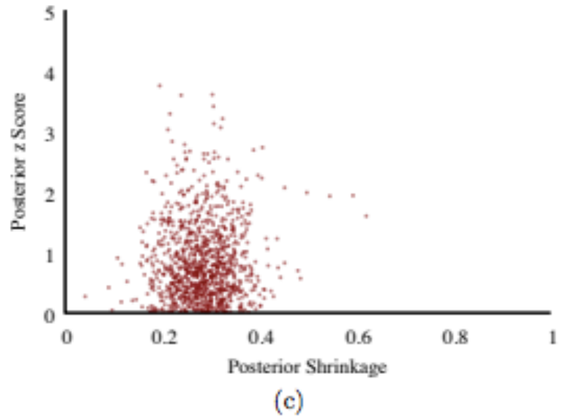


- see Cook et al
- take each $\tilde{y}$
- get a $\theta \mid \tilde{y}$ posterior
- find the rank of $\tilde{\theta}$ in "its" posterior
- a histogram of ranks should be uniform- this tests our sampling software

QC: General metrics: Sensitivity of posterior to range allowed by prior





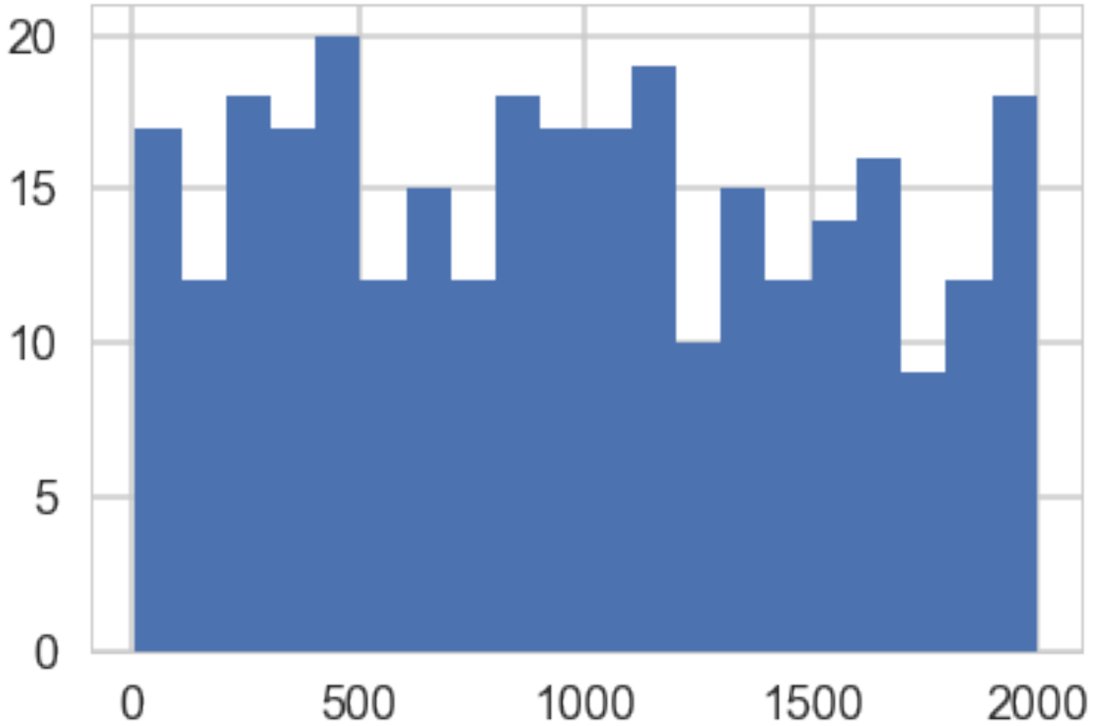


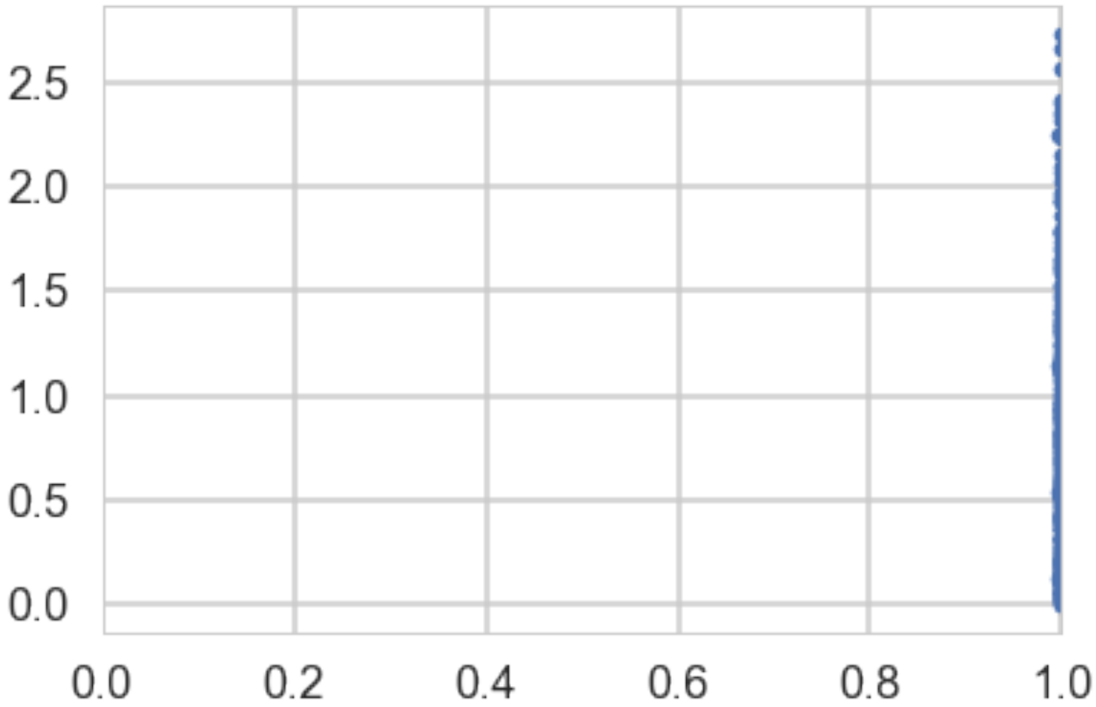
$$z_n = \left| \frac{\mu_n(\theta_n|\tilde{y}) - \tilde{\theta}_n}{\sigma_n(\theta_n|\tilde{y})} \right|$$

$$s_n = 1 - \frac{\sigma_n(\theta_n|\tilde{y})^2}{\tau_n(\tilde{y})^2}$$

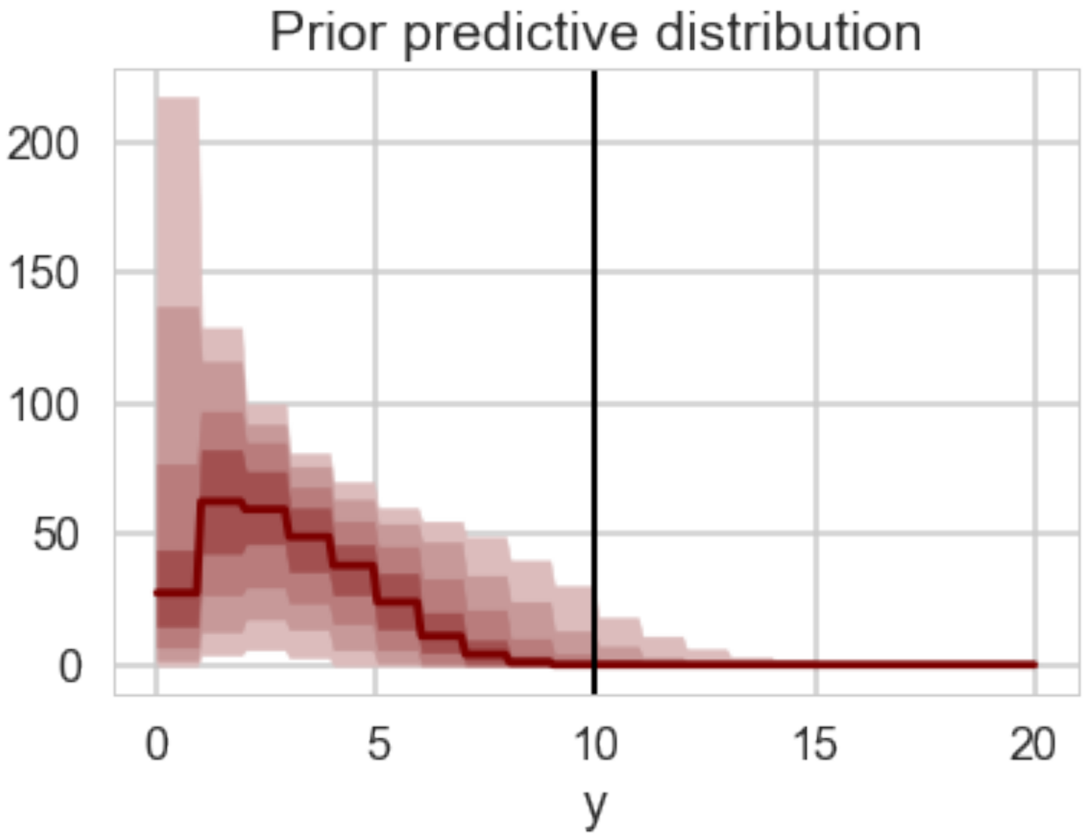
where μ and σ are generated-posterior quantities and τ is a prior one, and n indexes the parameters

Drunk Monks pre-obs





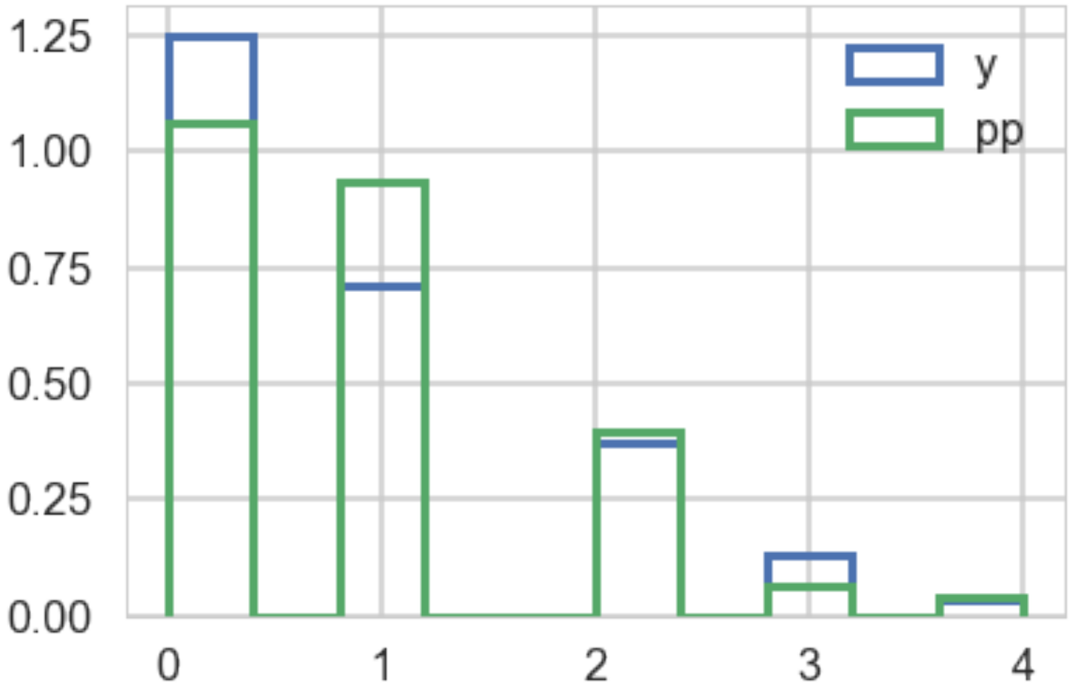
Model0: Prior Predictives

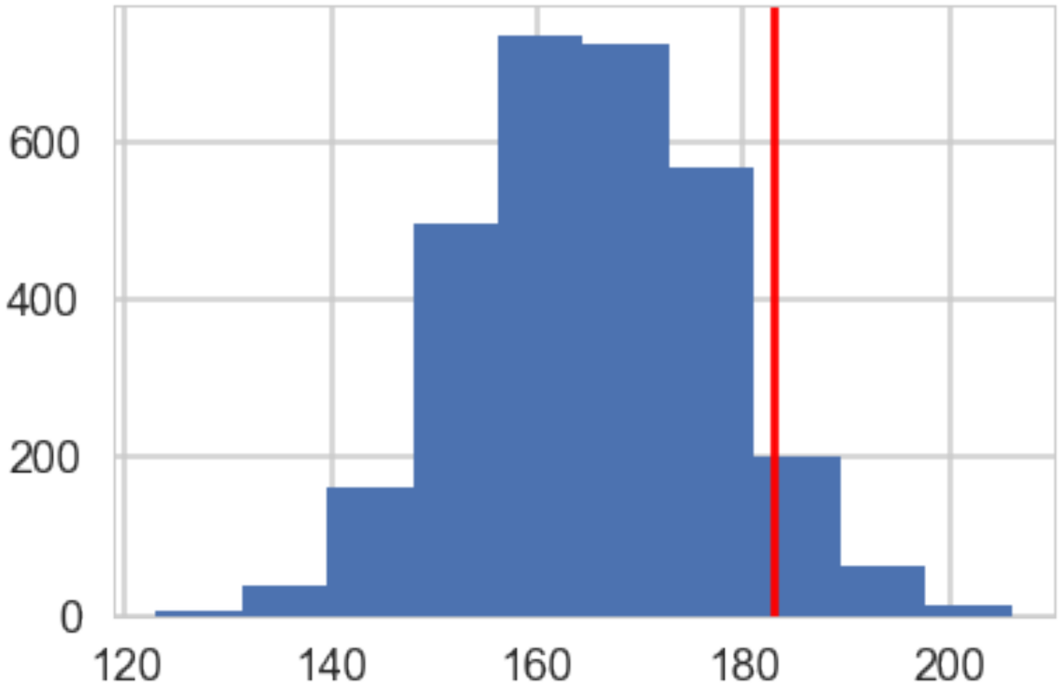


Then move to the REAL DATA posterior

- now we do posterior predictive checks
- the prior checks have specified possible data distributions that can be generated
- the posterior predictive ought to be a subset of these. If not our model is mis-specified
- this may seem strange as we didnt think priors are data generating
- they are not but are defined with respect to the likelihood

Drunk Monks, post-obs





pp check shows need for 0 inflation, so do that, rinse+repeat

Zero Inflated Poisson Mixture model

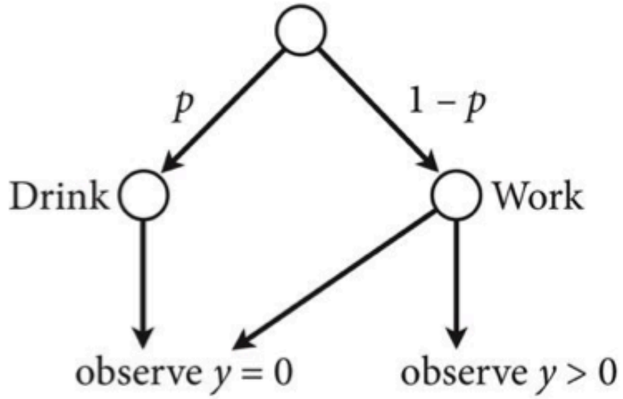
(A) : p

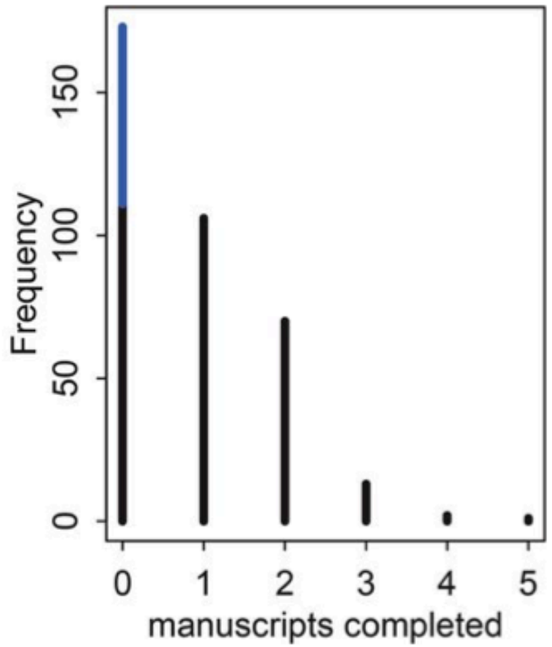
(B) : $(1-p)e^{-\lambda} + (1-p)\frac{\lambda^y e^{-\lambda}}{y!}$

Can also split this as

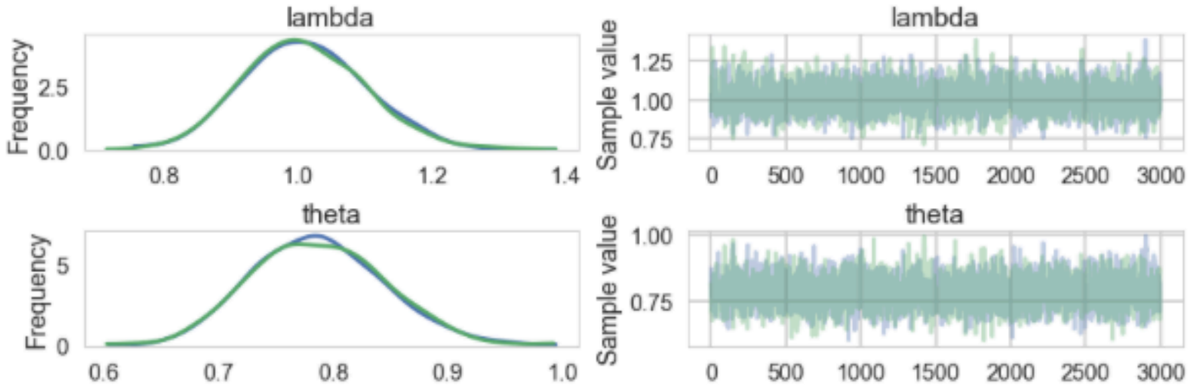
$$\mathcal{L}(y=0) = p + (1-p)e^{-\lambda}$$

$$\mathcal{L}(y\neq 0) = (1-p)\frac{\lambda^y e^{-\lambda}}{y!}$$





Fit the model



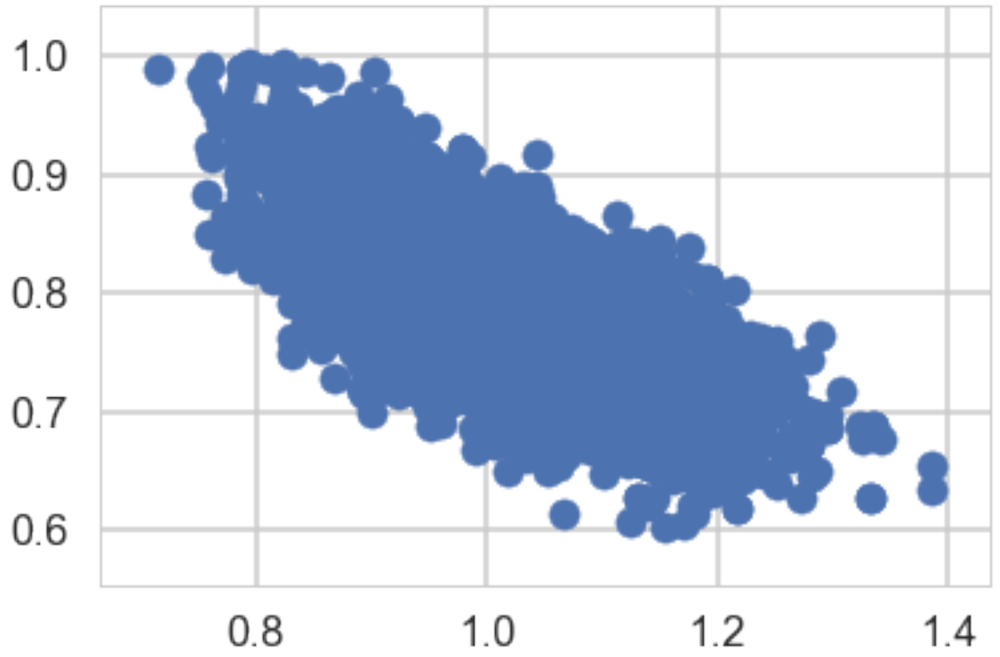

```
pm.summary(trace2)
```

	mean	sd	mc_error	hpd_2.5	hpd_97.5	n_eff	Rhat
lambda	1.010743	0.090270	0.001997	0.841846	1.188071	1877.185581	0.999891
theta	0.787184	0.058392	0.001319	0.673310	0.897211	1842.570275	0.999837

```
with pm.Model() as model2:
    lam=pm.HalfNormal("lambda", sd)
    theta=pm.Beta("theta", 1,1)
    like = pm.ZeroInflatedPoisson("obsv", theta=lam,
            psi=theta, shape = shp,
                observed=observed)
```

QC: Non-identifiability

- at low λ (productivity) with respect to high θ (chances of being drunk)
- a QC deal which WILL show up in QB



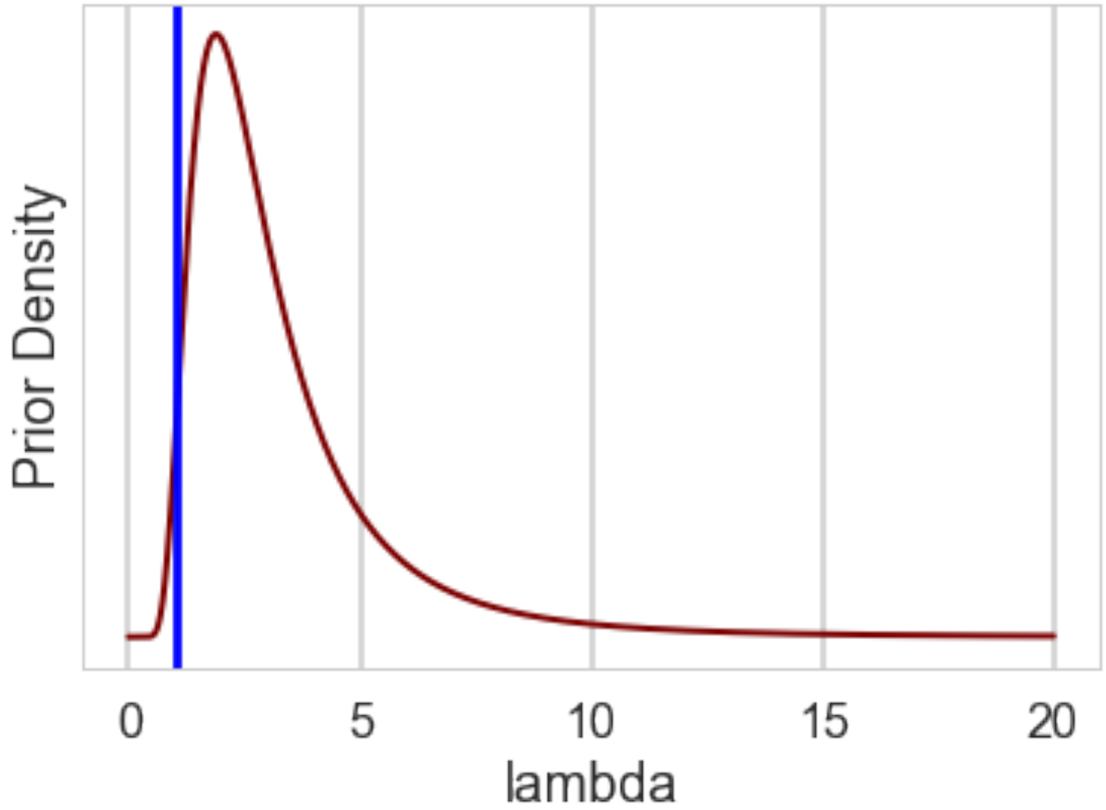

```
There were 13 divergences after tuning. Increase `target_accept` or repar
ameterize.
There were 17 divergences after tuning. Increase `target_accept` or repar
ameterize.
The number of effective samples is smaller than 25% for some parameters.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:08<00:00, 911.35draws/s
]
There was 1 divergence after tuning. Increase `target_accept` or reparame
terize.
The number of effective samples is smaller than 25% for some parameters.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:07<00:00, 1076.51draws/
s]
There were 12 divergences after tuning. Increase `target_accept` or repar
ameterize.
There were 7 divergences after tuning. Increase `target_accept` or repara
meterize.
The number of effective samples is smaller than 25% for some parameters.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
```

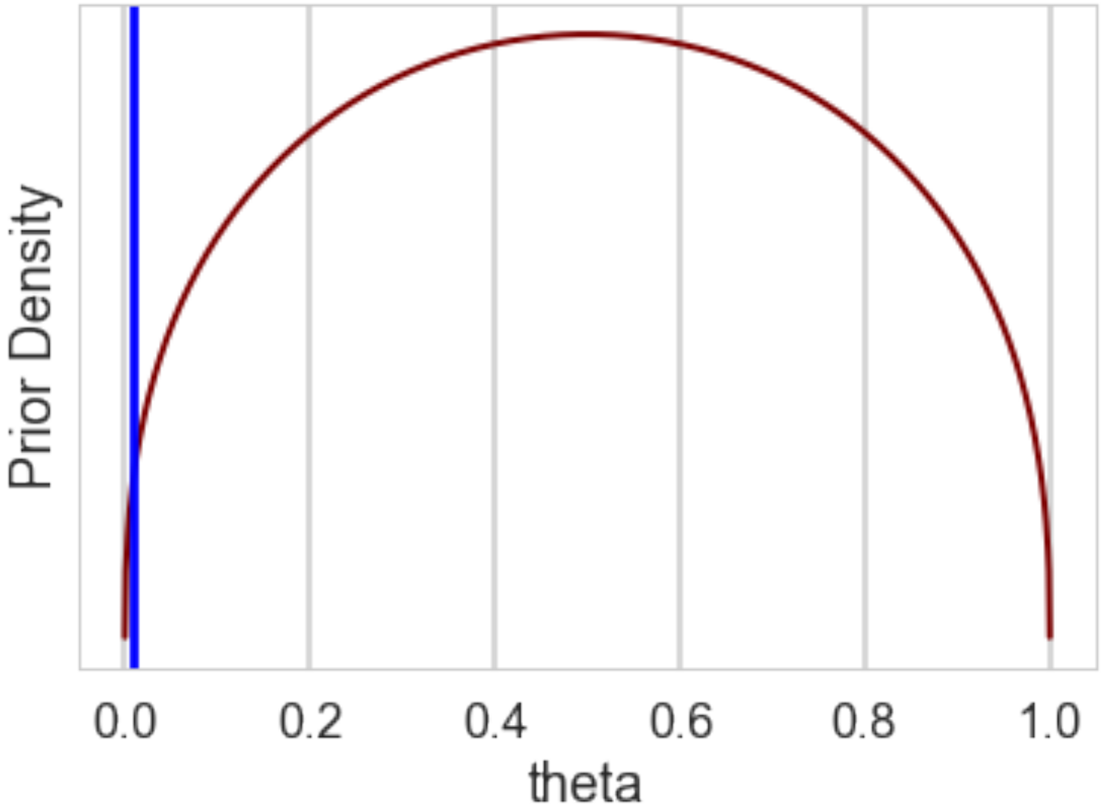
- really we should be doing the entire process we went through for model 0 again. - But making multiple prior-predictive-data fits is expensive unless you have access to massively parallel hardware
- so our belief on non-identidiability (QC) showing up in poor sampling (QB) sometimes serves as enough evidence against this 0-inflated model, we wont run theorugh the entire callibration process for QB and QC
- sometimes we might even do QD before QB and QC when we suspectmodel troubles.

What to do?

- we go back to the Abbey head
- who tells us that even poorly productive monks (writing tweets bout the special counsel) will produce atleast one manuscript in a day
- this is new domain knowledge (QA), and will help fix the non-identifiability: 0 manuscripts are likely from drunkenness.
- additionally we'll assume that its very unlikely to have a high probability or very low probability that a monk is drunk: monks are neither drunk very rarely or too much
- still with lack of knowledge bust assume as uninformative a prior as reasonable on drunkenness

QA-new priors





Model 3

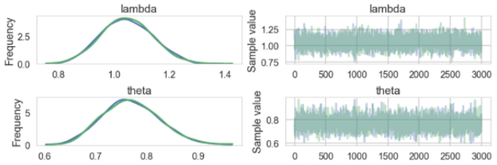
```
with pm.Model() as model:
    lam = pm.InverseGamma("lambda",alpha=alpha,beta=beta)
    theta=pm.Beta("theta", curve, curve)
    like = pm.ZeroInflatedPoisson("obsv", theta=lam,
        psi=theta,
        shape = shp, observed=observed)
```

```
model3 = model_0ipoisson2(alpha, beta, curve, y.shape[0], observed=y)
with model3:
    trace3 = pm.sample(3000, tune=1000)
```

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:06<00:00, 1263.90draws/
s]
```

```
pm.summary(trace3)
```

	mean	sd	mc_error	hpd_2.5	hpd_97.5	n_eff	Rhat
lambda	1.047114	0.091194	0.001884	0.865752	1.219998	2129.650694	0.999833
theta	0.769899	0.056432	0.001331	0.661380	0.879076	1779.443335	1.000031



- cursory prior-predictive fits on prior edges look much better
- now carry out the entire workflow fixing sampler issues and running all checks

```
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:05<00:00, 1541.88draws/
s]
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:04<00:00, 1659.39draws/
s]
There was 1 divergence after tuning. Increase `target_accept` or reparame
terize.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:05<00:00, 1522.45draws/
s]
There was 1 divergence after tuning. Increase `target_accept` or reparame
terize.
There was 1 divergence after tuning. Increase `target_accept` or reparame
terize.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:05<00:00, 1413.76draws/
s]
There was 1 divergence after tuning. Increase `target_accept` or reparame
terize.
Auto-assigning NUTS sampler...
Initializing NUTS using jitter+adapt_diag...
Multiprocess sampling (2 chains in 2 jobs)
NUTS: [theta, lambda]
Sampling 2 chains: 100%|██████████| 8000/8000 [00:05<00:00, 1468.52draws/
s]
```

The Workflow (from Betancourt, and Savage)

Prior to Observation

1. Define Data and interesting statistics
2. Build Model
3. Analyze the joint, and its data marginal (prior predictive) and its summary statistics
4. fit posteriors to simulated data to calibrate
    - check sampler diagnostics, and correlate with simulated data
    - use rank statistics to evaluate prior-posterior consistency
    - check posterior behaviors and behaviors of decisions

Posterior to Observation

1. Fit the Observed Data and Evaluate the fit
    - check sampler diagnostics, poor performance means generative model not consistent with actual data
2. Analyze the Posterior Predictive Distribution
    - do posterior predictive checks, now comparing actual data with posterior-predictive simulations
    - consider expanding the model (REPEAT THE WHOLE WORKFLOW, or SALIENT parts)
3. Do model comparison (if needed)
    - usually within a nested model, but you might want to apply a different modeling scheme, in which case use loo
    - you might want to ensemble instead

Make Sure

Final Model has

all parts of workflow done

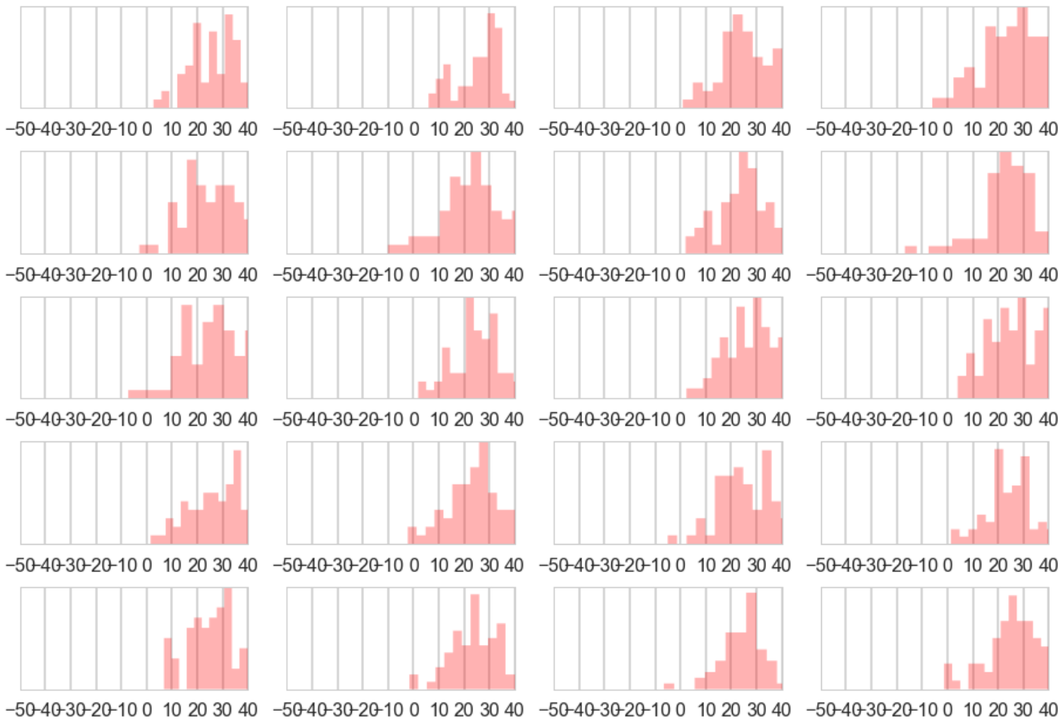
Notes on formal posterior-predictive checking

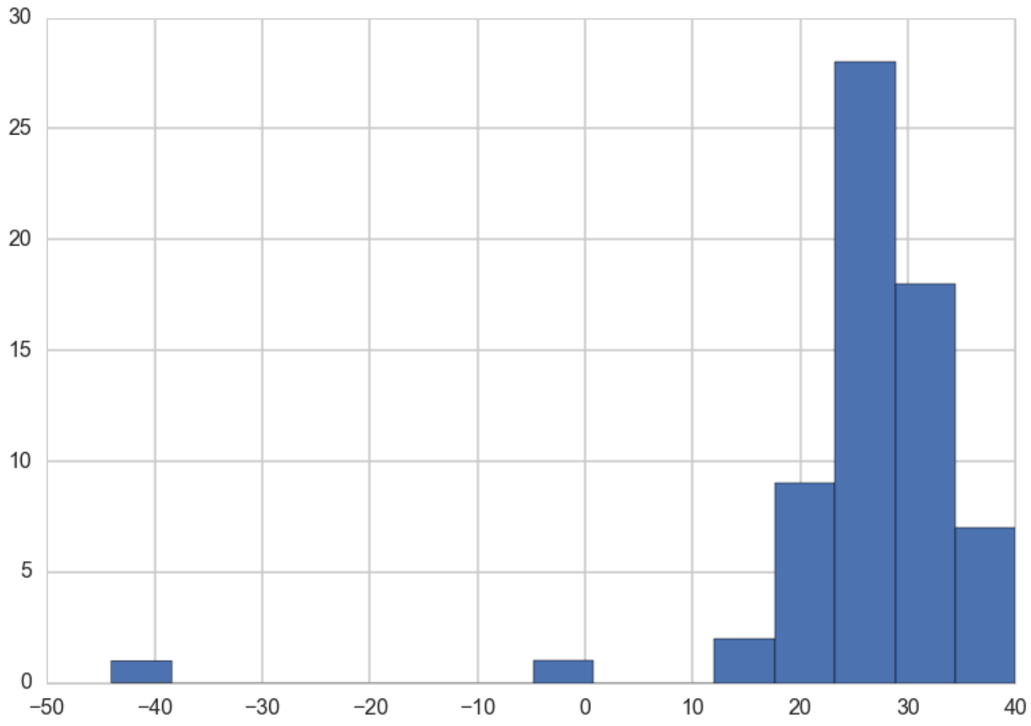
$$p(\{y^*\}) = \int p(\{y^*\}|\theta)p(\theta|\mathcal{D})d\theta$$, observed data: $\mathcal{D} = \{y\}$

Replicated Data: $\{y_r\}$: data seen tomorrow if experiment replicated with same model and value of θ producing todays data $\{y\}$.

$\{y_r\}$ comes from posterior predictive, and if there are covariates $\{x^*\}$, then $\{y_r\}$ is calculated at those covariates only (`sample_ppc`).

Visual Checking





Do these even look similar??

Discrepancy

Gelman: *A test quantity, or discrepancy measure, $T(\{y\}, \theta)$, is a scalar summary of parameters and data that is used as a standard when comparing data to predictive simulations.*

The classical p-value for the test statistic $T(\{y\})$ is given by

$$p_C = P(T(\{y_r\}) \geq T(\{y\})|\theta)$$

where probability is over distrib of $\{y_r\}$ with θ fixed (bootstrap).

Bayesian p-values

$$p_B = Pr(T(\{y_r\}, \theta) \geq T(\{y\}, \theta)|\{y\}),$$

probability over the posterior and posterior predictive (that is, the joint distribution, $p(\theta, \{y_r\}|\{y\})$).

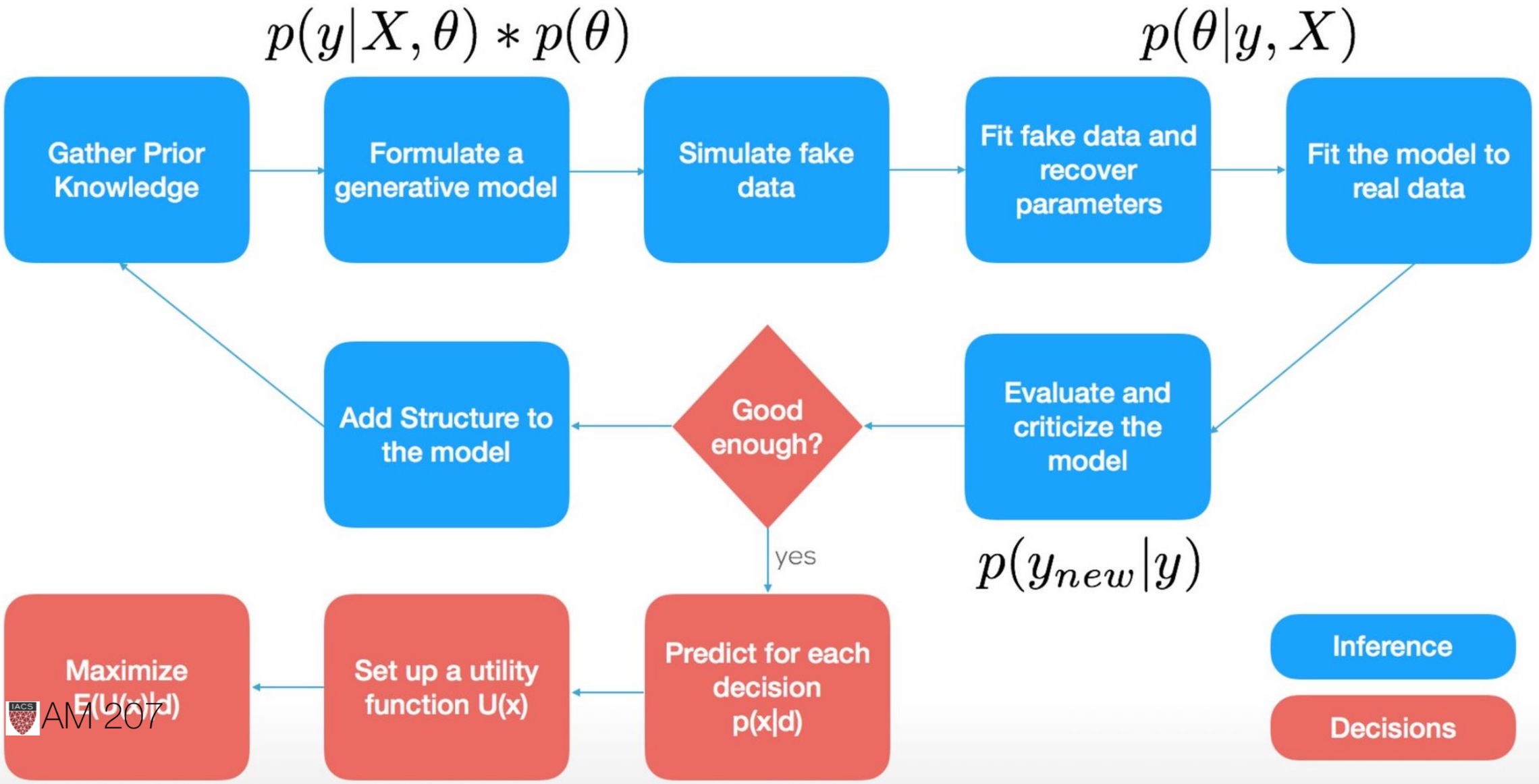
$$p_B = \int d\theta \, d\{y_r\} \, I(T(\{y_r\}, \theta) \geq T(\{y\}, \theta)) \, p(\{y_r\}|\theta) p(\theta|\{y\})$$

using $p(\{y_r\}|\theta, \{y\}) = p(\{y_r\}|\theta)$.

Appropriate usage: the scientific Method

Gelman: *Finding an extreme p-value and thus 'rejecting' a model is never the end of an analysis; the departures of the test quantity in question from its posterior predictive distribution will often suggest improvements of the model or places to check the data, as in the speed of light example. Moreover, even when the current model seems appropriate for drawing inferences (in that no unusual deviations between the model and the data are found), the next scientific step will often be a more rigorous experiment incorporating additional factors, thereby providing better data.*

Bayesian Workflow



All Models are good, but some models are useful