Lecture 15

Recap and Data Augmentation to HMC

Law of Large numbers

Let $x_1, x_2, \ldots, x_n$ be a sequence of IID values from random variable X, which has finite mean μ. Let:

$$S_n = \frac{1}{n} \sum_{i=1}^{n} x_i,$$

Then:

$$S_n \to \mu \, as \, n \to \infty.$$

LOTUS: Law of the unconscious statistician

Also known as **The rule of the lazy statistician**.

Theorem:

if $Y = r(X)$,

$$E[Y] = \int r(x) dF(x)$$

Law of Large numbers (LLN)

- Expectations become sample averages. Convergence for large N.

$$E_f[g] = \int g(x) dF = \int g(x) f(x) dx$$
$$= \lim_{n \to \infty} \frac{1}{N} \sum_{x_i \sim f} g(x_i)$$

- allows for monte-carlo

Frequentist Statistics

Answers the question: **What is Data?** with

"data is a **sample** from an existing **population**"

- data is stochastic, variable
- model the sample. The model may have parameters
- find parameters for our sample. The parameters are considered fixed.

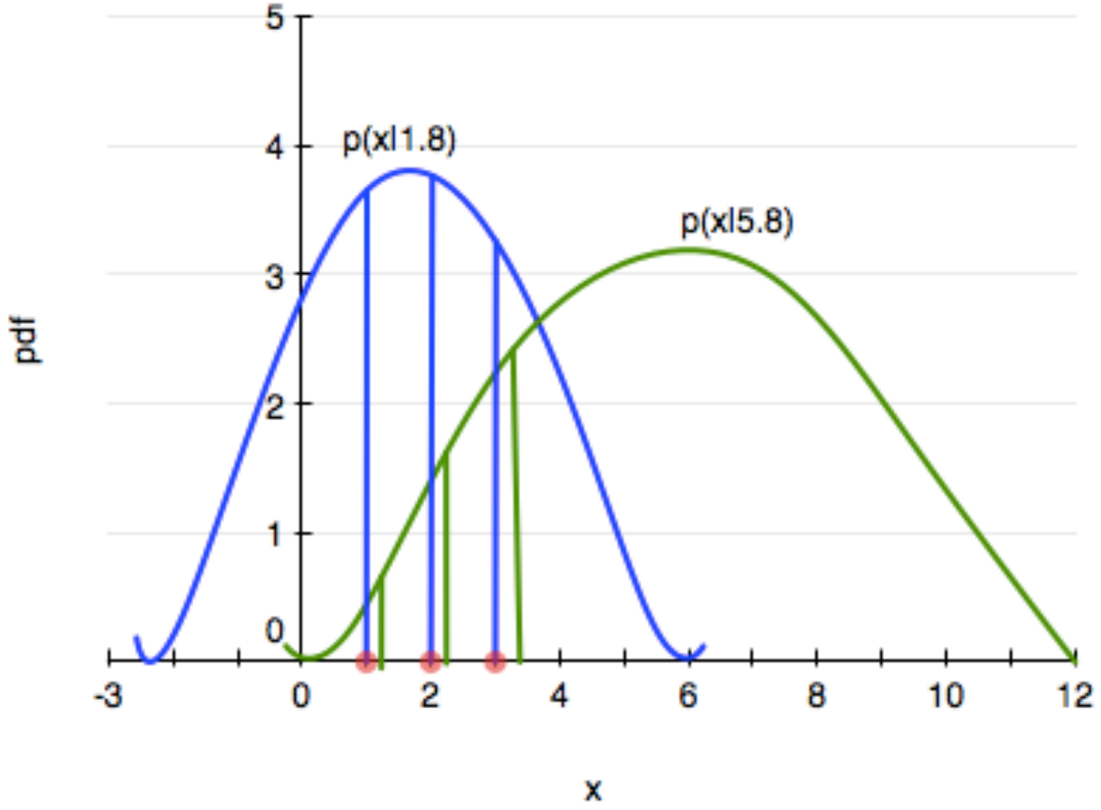
Point Estimates

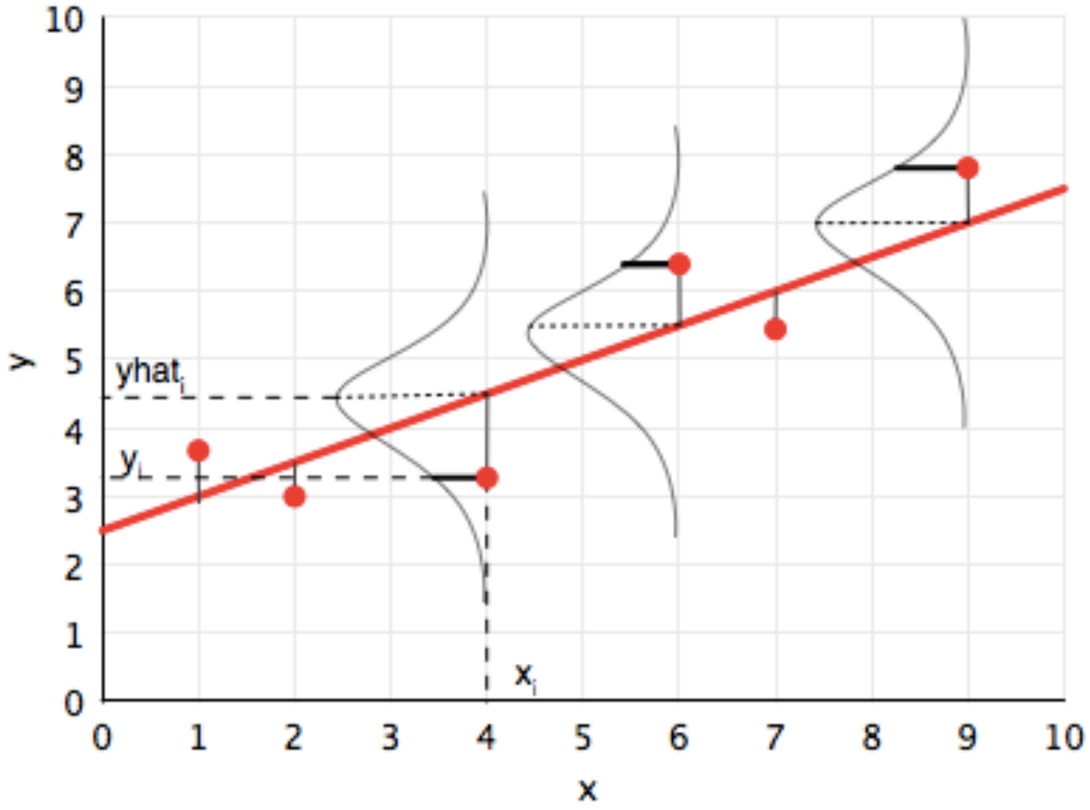
Estimate on the sample by applying an estimator F to the sample data D, so $\hat{\mu} = F(D)$.

If your model describes the true generating process for the data, then there is some true μ^*.

The best we can do is to estimate $\hat{\mu}$.

MLE


p(xl1.8)
p(xl5.8)
pdf
x
5
4
3
2
1
0
-3
0
2
4
6
8
10
12


yhat$_i$
y$_i$
x$_i$
y
x
10
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8
9
10

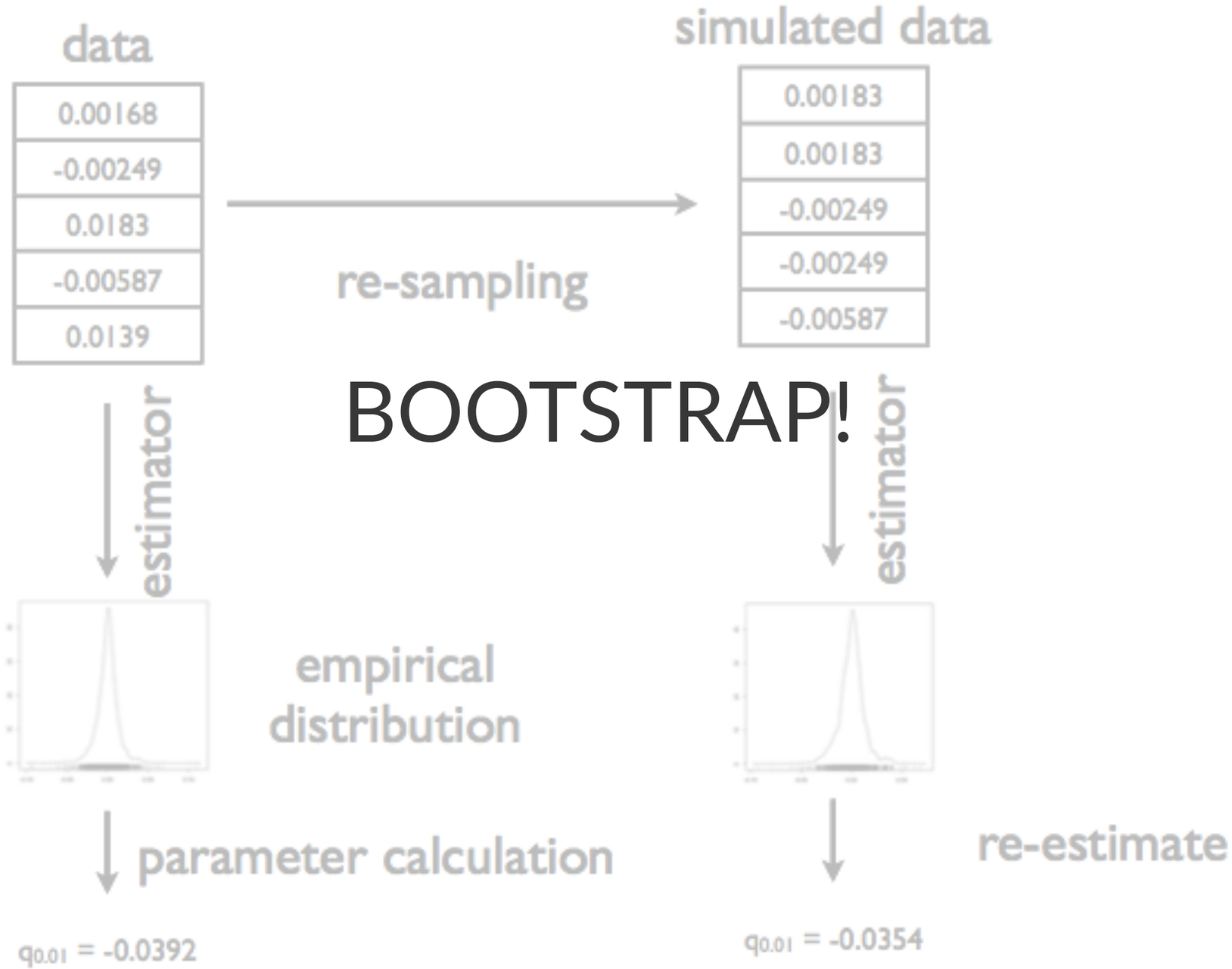
Sampling distribution

Now, imagine that God gives you some M data sets **drawn** from the population, and you can now find μ on each such dataset.

So, we'd have M estimates.

As we let $M \to \infty$, the distribution induced on $\hat{\mu}$ is the empirical **sampling distribution of the estimator**.

But we dont have M samples. What to do?


data
0.00168
-0.00249
0.0183
-0.00587
0.0139
simulated data
0.00183
0.00183
-0.00249
-0.00249
-0.00587
re-sampling
BOOTSTRAP!
estimator
estimator
empirical
distribution
parameter calculation
re-estimate
$q_{0.01}$ = -0.0392
$q_{0.01}$ = -0.0354

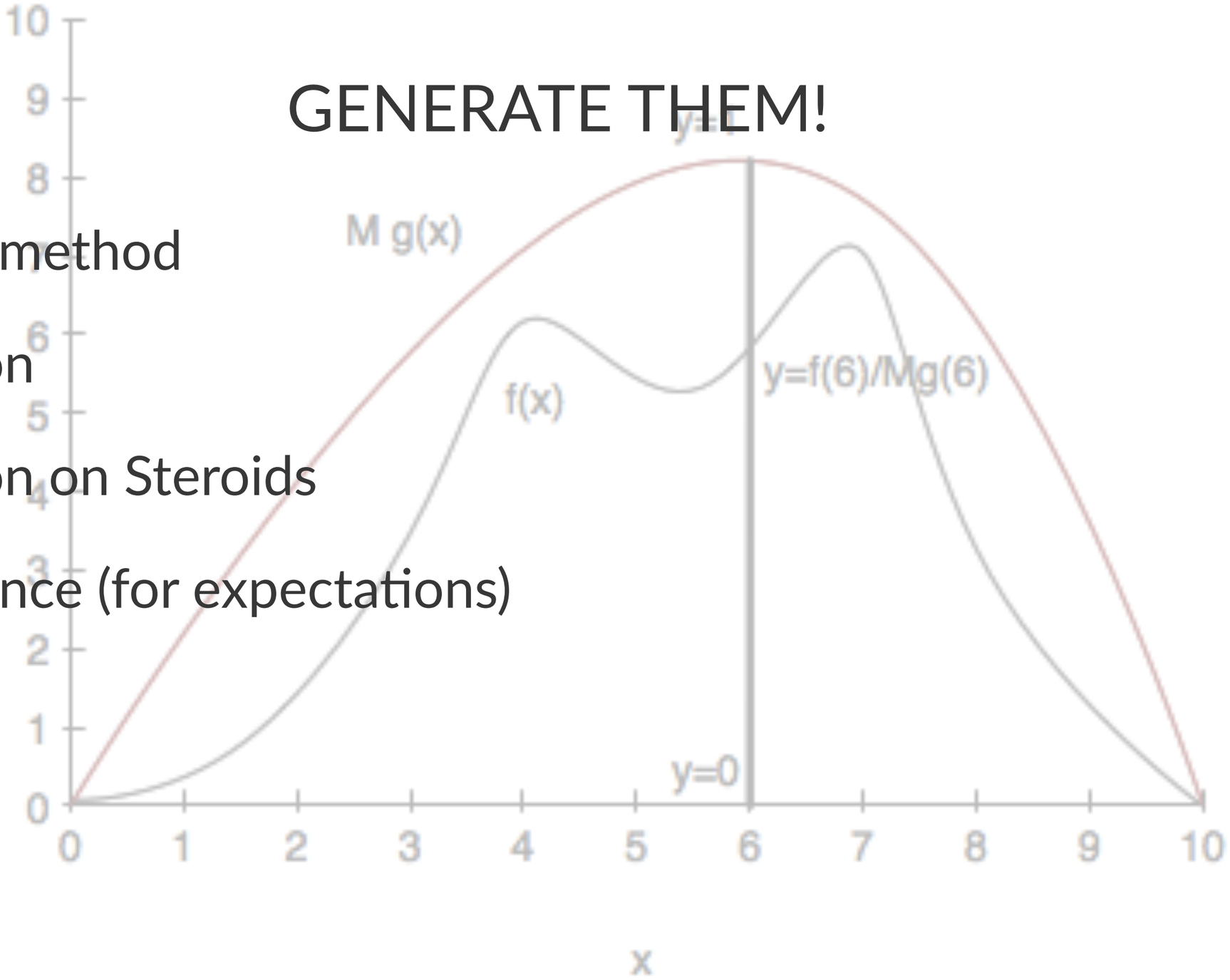
BUT

WE NEEDS

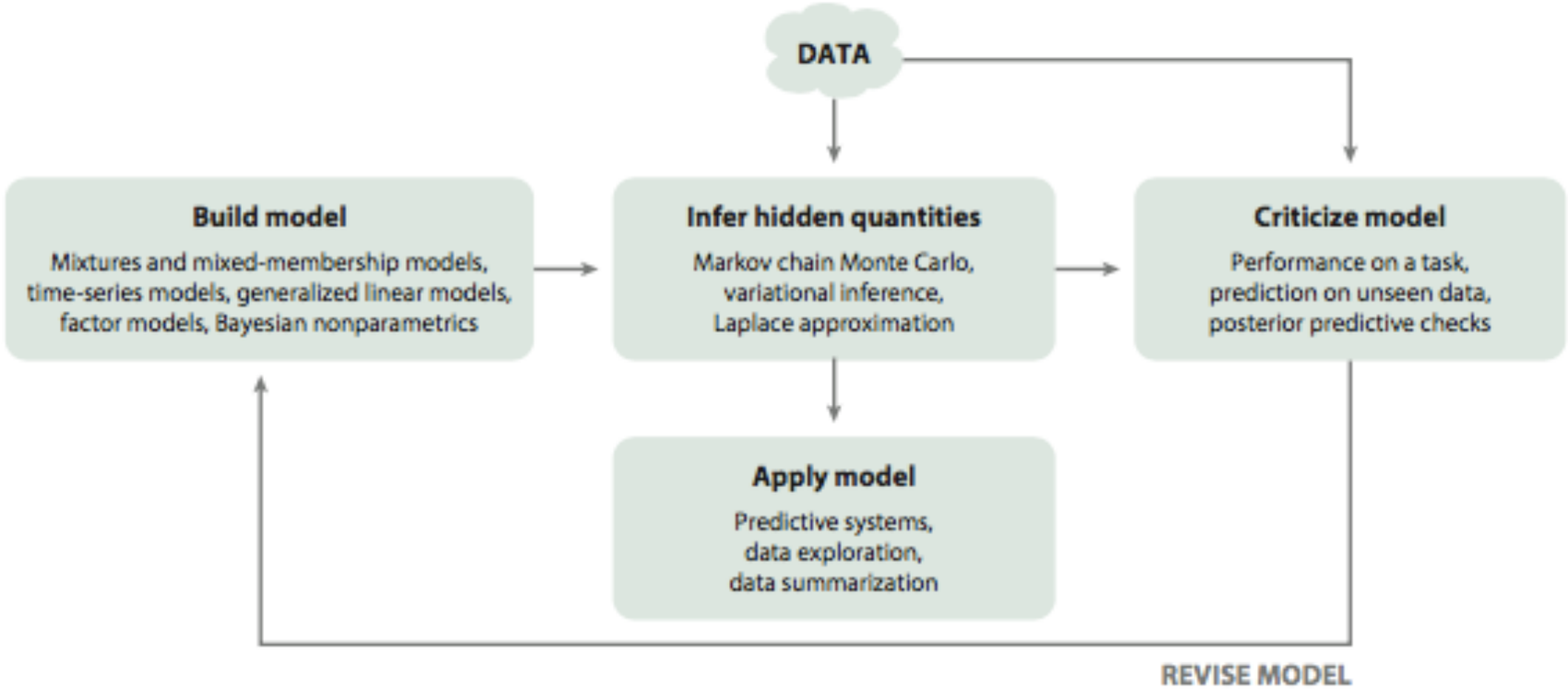
SAMPLES

GENERATE THEM!

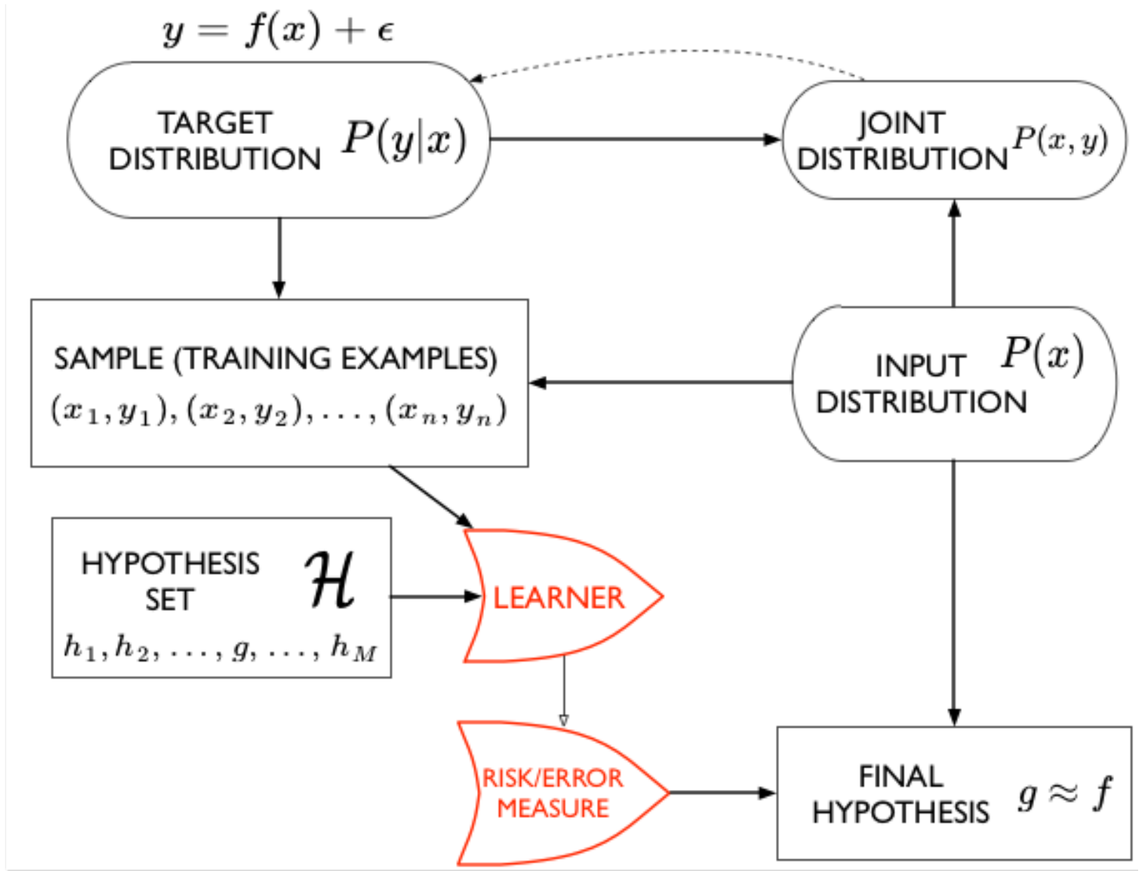
- Inverse method
- Rejection
- Rejection on Steroids
- Importance (for expectations)
- MCMC



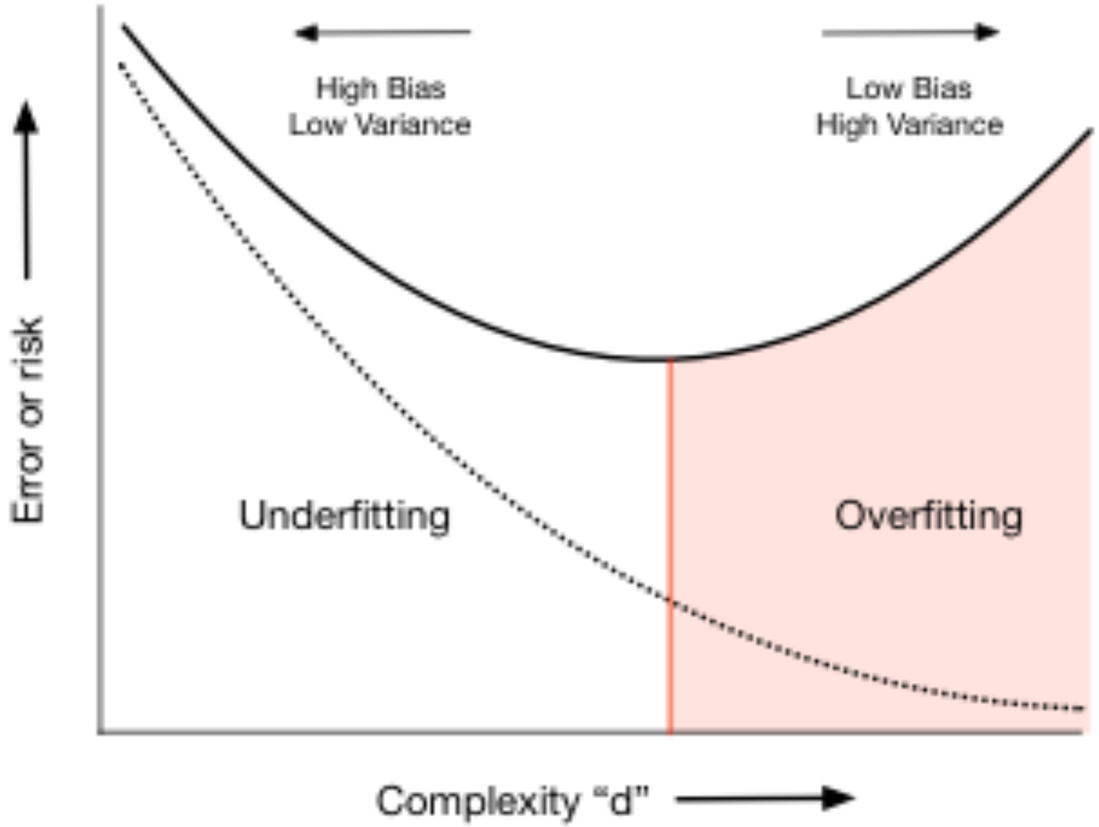
Box's loop


DATA
Build model
Mixtures and mixed-membership models, time-series models, generalized linear models, factor models, Bayesian nonparametrics
Infer hidden quantities
Markov chain Monte Carlo, variational inference, Laplace approximation
Criticize model
Performance on a task, prediction on unseen data, posterior predictive checks
Apply model
Predictive systems, data exploration, data summarization
REVISE MODEL

The nature of learning

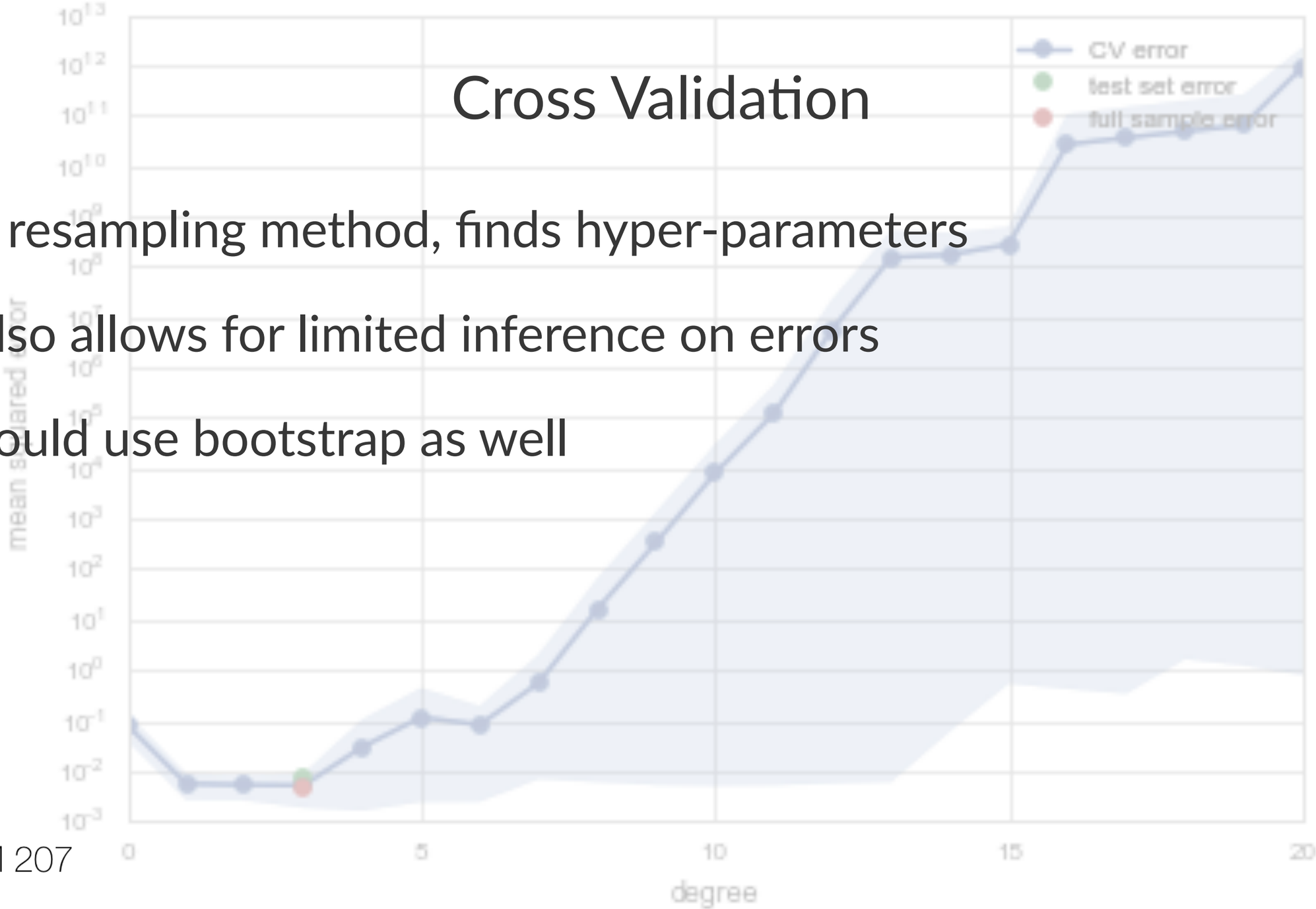


Complexity, Risk, Bias, and Variance



Cross Validation

- a resampling method, finds hyper-parameters
- also allows for limited inference on errors
- could use bootstrap as well



Grounding with Information Theory

KL-Divergence

$$D_{KL}(p,q) = E_p[log(p) - log(q)] = E_p[log(p/q)]$$
$$= \sum_i p_i log(\frac{p_i}{q_i}) \ or \int dPlog(\frac{p}{q})$$

KL divergence measures distance/dissimilarity of the two distributions p(x) and q(x).

Maximum Likelihood justification

Use Empirical distribution for p.

$$D_{KL}(p,q) = E_p[log(p/q)] = \frac{1}{N}\sum_i (log(p_i) - log(q_i)$$

Minimizing KL-divergence $\implies$ maximizing $\sum_i log(q_i)$

Which is exactly the log likelihood! MLE!

Information Entropy, a measure of uncertainty

Desiderata:
- must be continuous so that there are no jumps
- must be additive across events or states, and must increase as the number of events/states increases

$$H(p) = -E_p[log(p)] = -\int p(x) log(p(x)) dx \;\; OR \; -\sum_i p_i log(p_i)$$

Maximum Entropy (MAXENT)

- finding distributions consistent with constraints and the current state of our information
- what would be the least surprising distribution?
- The one with the least additional assumptions?

The distribution that can happen in the most ways is the one with the highest entropy

Importance of MAXENT

- most common distributions used as likelihoods (and priors) are in the exponential family, MAXENT subject to different constraints.
- choosing a maxent distribution as a likelihood means that once the constraints has been met, no additional assumptions: The most conservative distribution we could choose consistent with our constraints!

Model Comparison: Deviance

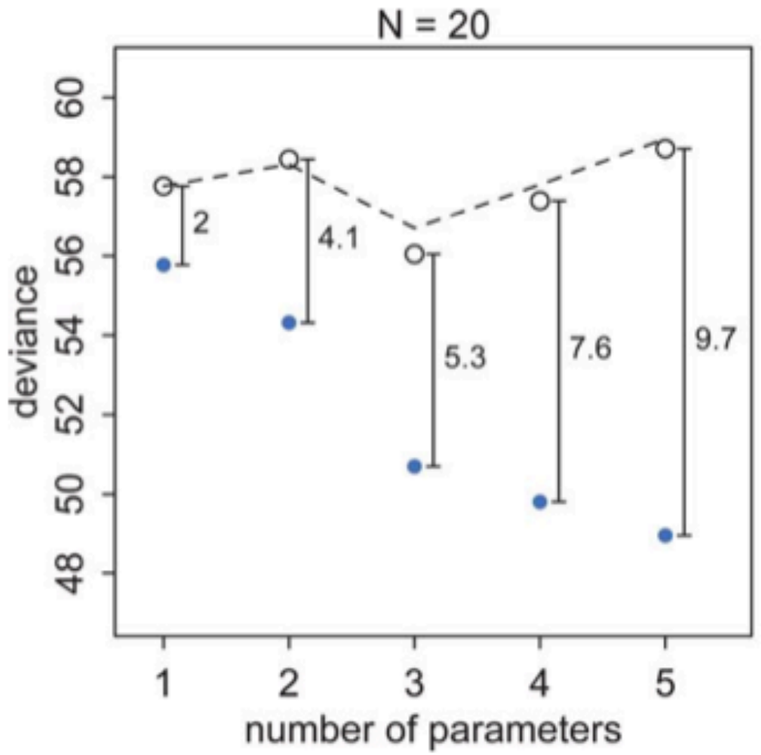
You only need the sample averages of the logarithm of r and q:

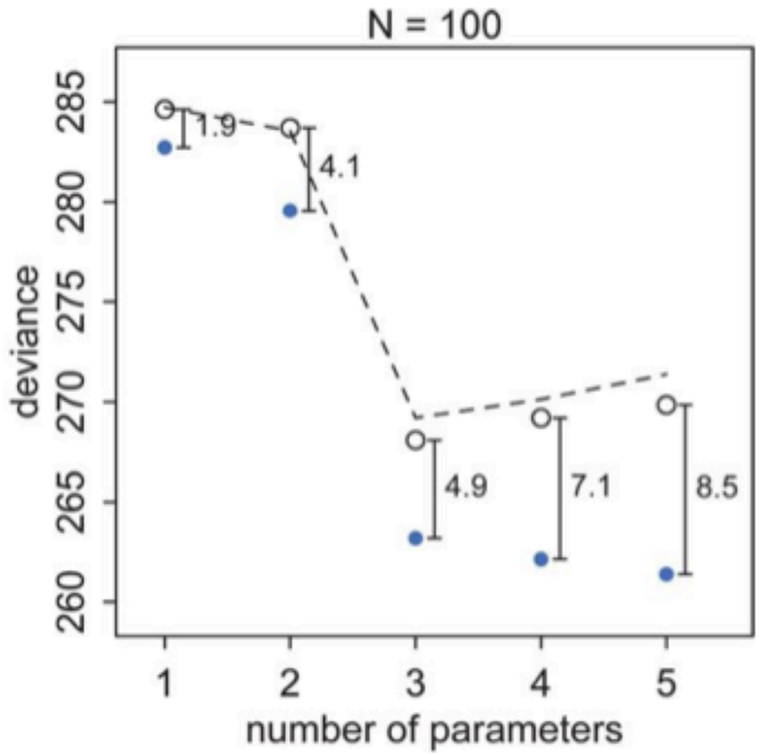
$$D_{KL}(p,q) - D_{KL}(p,r) = \langle log(r) \rangle - \langle log(q) \rangle$$

Define the deviance: $D(q) = -2 \sum_i log(q_i)$, a risk (e.g., -2 $\times$ ℓ, although the distribution need not be a likelihood)...

$$D_{KL}(p,q) - D_{KL}(p,r) = \frac{2}{N}(D(q) - D(r))$$

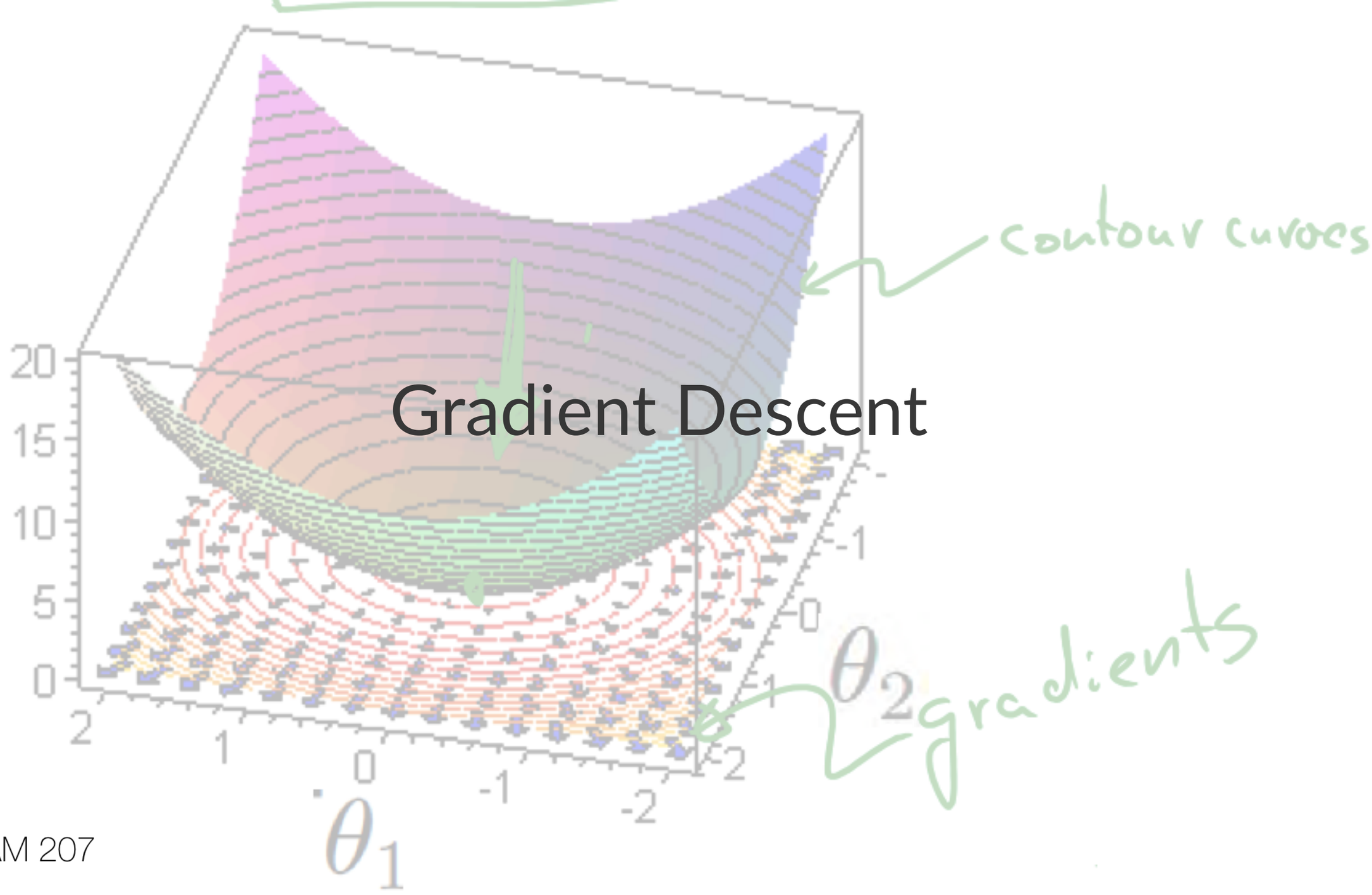
Deviance and AIC

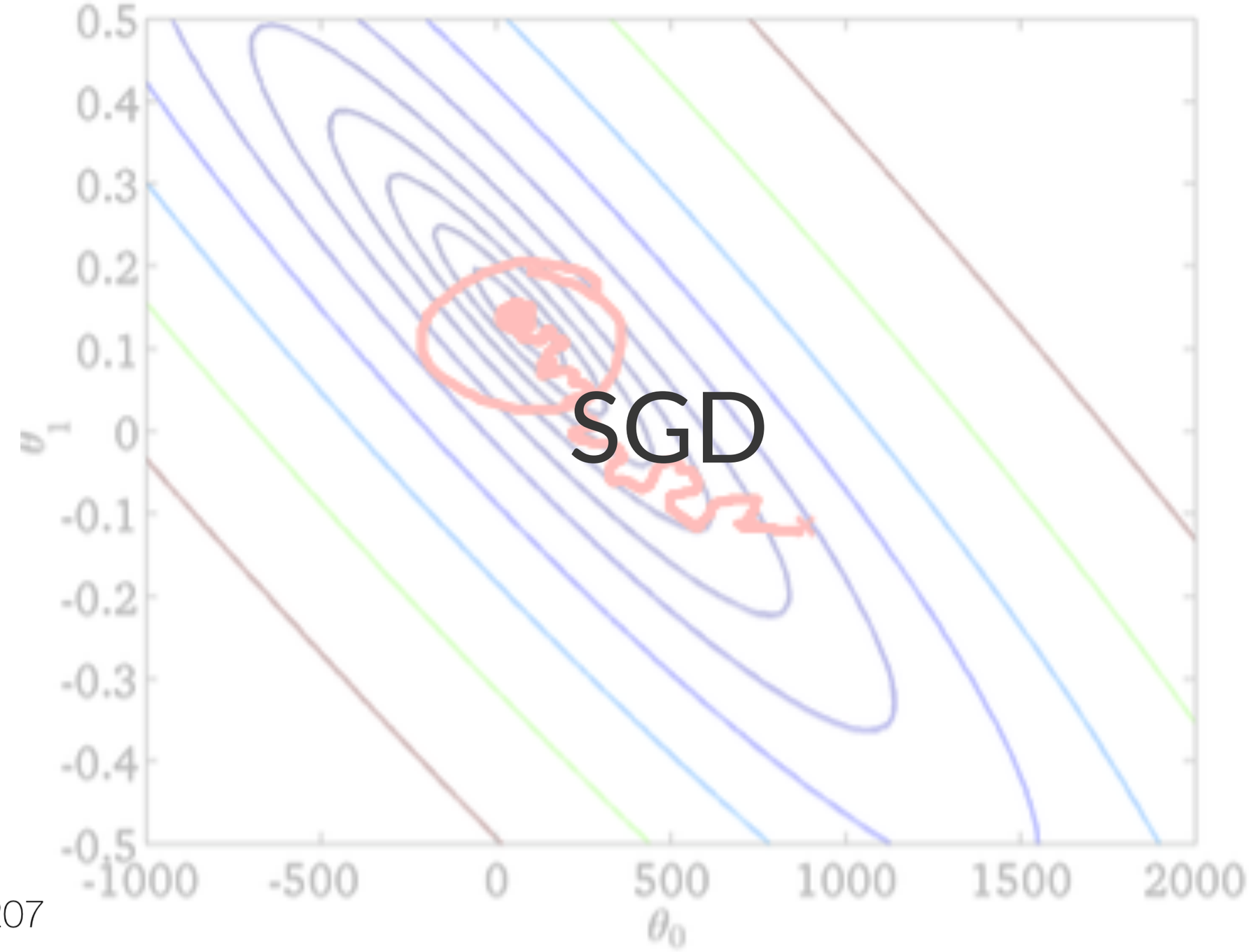




OPTIMIZATION

Gradient Descent




SGD
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
θ_1
-1000
-500
0
500
1000
1500
2000
θ_0

Simulated Annealing

Minimize f by identifying with the energy of an imaginary physical system undergoing an annealing process.

Move from x_i to x_j via a **proposal**.

If the new state has lower energy, accept x_j.

If the new state has higher energy, accept with $A = \exp\left(-\Delta f / kT\right)$

Lowering temperature slowly, the system reaches "thermal equilibrium" at each temperature. Boltzmann's distribution:

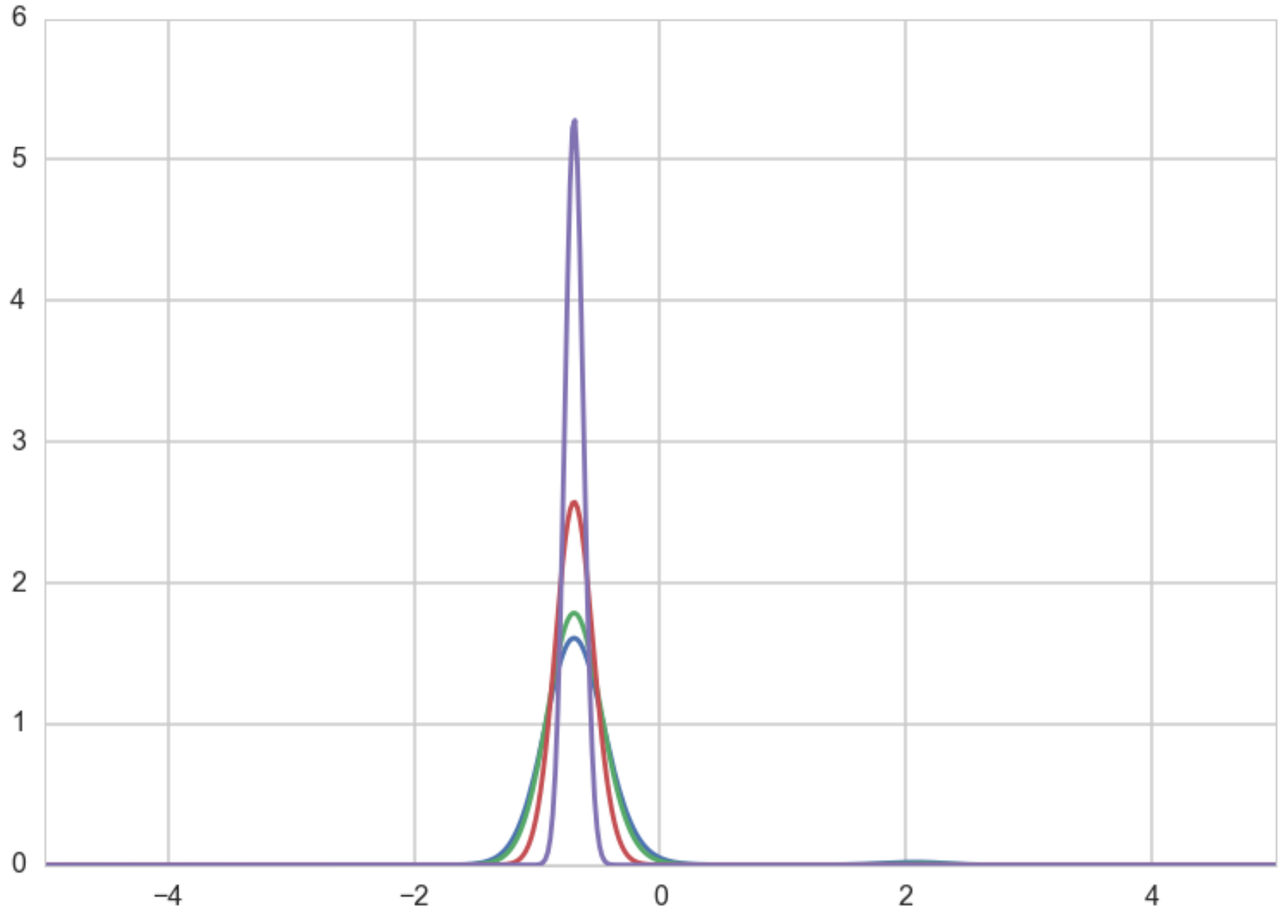
$$p(X = i) = \frac{1}{Z(T)} \exp\left(\frac{-E_i}{kT}\right)$$

If you identify

$$p_T(x) = e^{-f(x)/T} \text{ and } p(x) = e^{-f(x)}$$

Then:

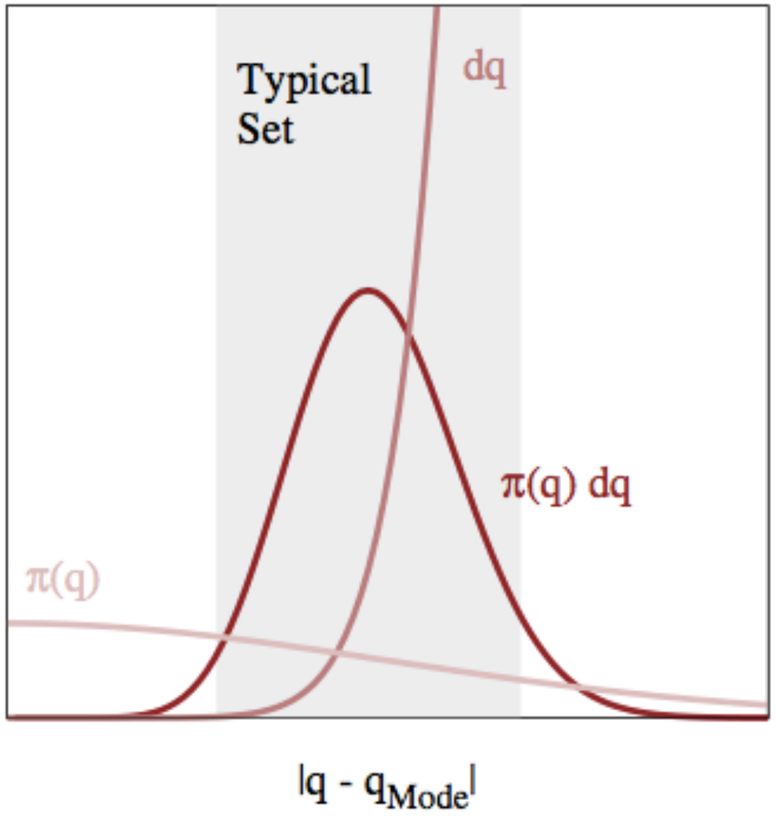
$$P_T(x) = P(x)^{1/T}$$



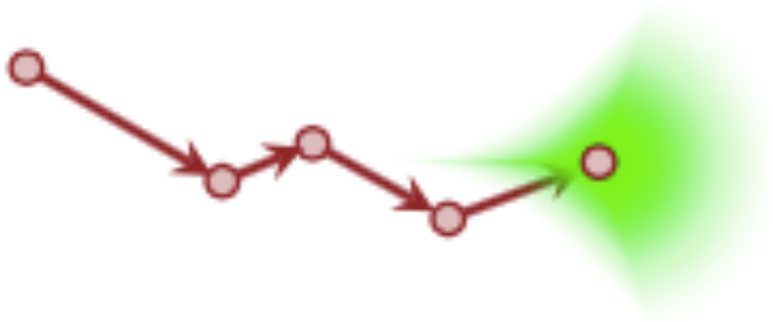
Sampling a Distribution

- Turn the question on its head.
- Suppose we wanted to sample from a distribution $p(x)$ (corresponding to a minimization of energy $-log(p(x))$).
- keep our symmetric proposal (reversibility!). Need irreducibility to sample from full distribution
- set T=1, and use our simulated annealing method: Metropolis

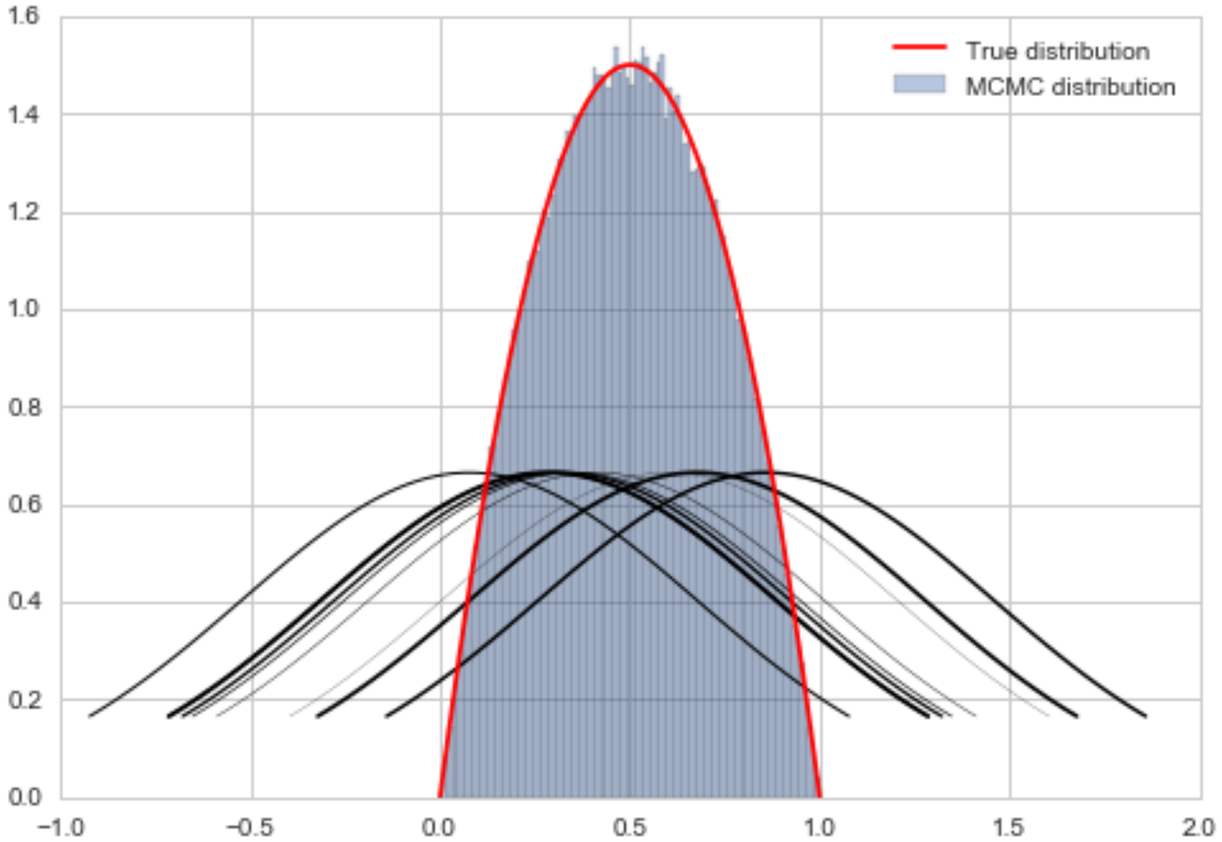
Critical: explore the typical set



Metropolis and MH



Sampling


True distribution
MCMC distribution
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

The idea of Gibbs

$$f(x) = \int f(x,y)dy = \int f(x|y)f(y)dy = \int dy f(x|y) \int dx' f(y|x')f(x')$$

Thus: $f(x) = \int h(x,x')f(x')dx'$ integral fixed point equation

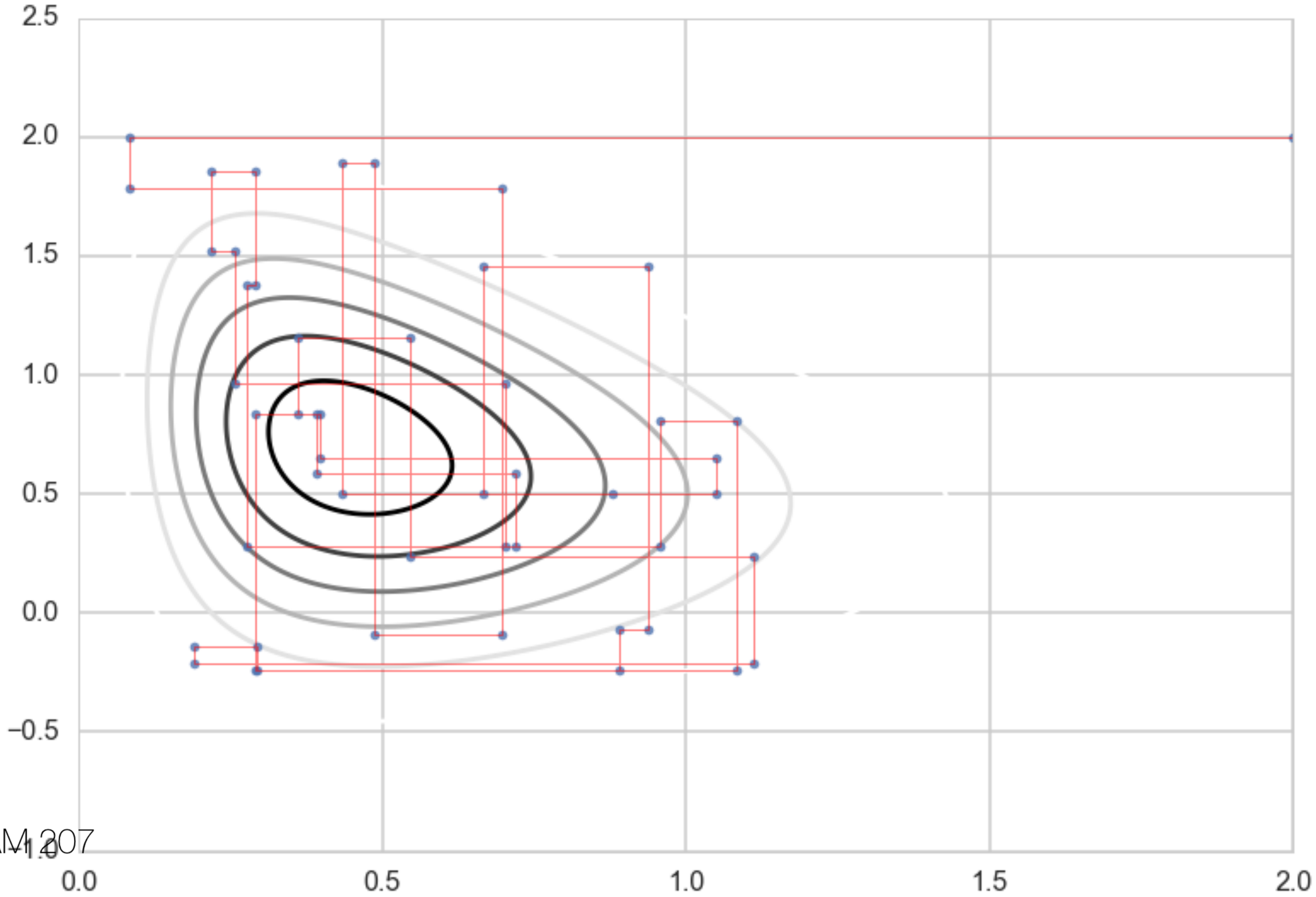
where $h(x,x') = \int dy f(x|y)f(y|x')$.

Iterative scheme in which the "transition kernel" $h(x, x')$ is used to create a proposal for metropolis-hastings moves:

$$f(x_t) = \int h(x_t, x_{t-1}) f(x_{t-1}) dx_{t-1},$$ a Stationary distribution.

$$h(x, x') = \int dy f(x|y) f(y|x').$$: Sample alternately to get transitions.

Can sample x marginal and $x|y$ so can sample the joint x, y.


2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
0.0
0.5
1.0
1.5
2.0

Data Augmentation

The difference from Gibbs Sampling: the other variable, say y, is to be treated as latent.

The game is to construct a joint $p(x, y)$ such that we can sample from $p(x|y)$ and $p(y|x)$, and then find the marginal

$$p(x) = \int dy\, p(x, y).$$

Latent Variables

- critical to our subsequent understanding
- dont think of bayes/frequentist, think of observed/Latent
- anything unobserved is latent (this is the posterior predictive point of view)
- standard bayesian viewpoint: nuisance parameters are latent
- latent factors in matrix factorization, mixtures, recommendations...

Simplest form of a DA algo:

1. Draw $Y \sim p_{Y|X}(.|x)$ and call the observed value y

2. Draw $X_{n+1} \sim p_{X|Y}(.|y)$

3. Histo the X

Example

Sample from $p(x) = e^{-x^2/2}/\sqrt{2\pi}$.

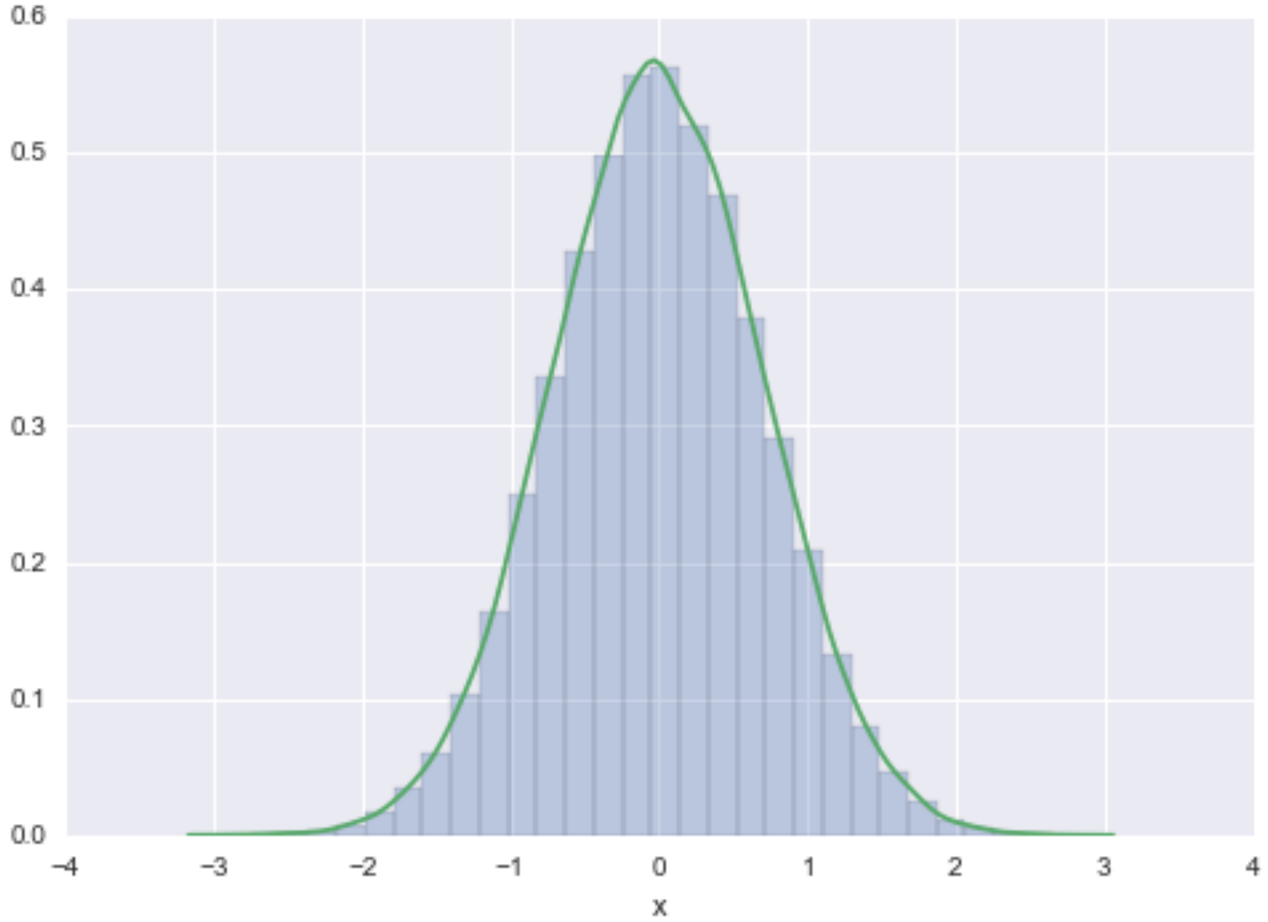
Take $p(x, y) = 1/(\sqrt{2\pi}) \exp\left\{-(x^2 - \sqrt{2}xy + y^2)\right\}$

$$Y|X = x \sim N(x/\sqrt{2}, 1/2) \text{ and } \mathrm{X}|\mathrm{Y} = \mathrm{y} \sim \mathrm{N}(\mathrm{y}/\sqrt{2}, 1/2)$$

The x-marginal is $\propto e^{-x^2/2} \int e^{-(y - x/\sqrt{2})^2} dy$

Example (contd)

```
N=100000
x=np.zeros(N)
x[0] = np.random.rand() # INITIALIZE
for i in np.arange(1,N):
    Y=sp.stats.norm.rvs(x[i-1]/np.sqrt(2), 0.5)
    x[i]=sp.stats.norm.rvs(Y/np.sqrt(2), 0.5)
```



Transition kernel

$h(x', x) = h(x'|x) = \int_Y p(x'|y)\, p(y|x)\, dy$ has stationarity by construction from Gibbs.

Its a probability: $\int h(x'|x)dx' = \int_X \int_Y p(x'|y)\, p(y|x)\, dy\, dx'$

$= \int_Y p(y|x) \left[\int_X p(x'|y)dx' \right] dy = \int_Y p(y|x)dy = 1$

$h(x'|x)\,p(x)$ is symmetric in (x, x'):

$$h(x'|x)\,p(x) = p(x)\int_Y p(x'|y)\,p(y|x)\,dy = \int_Y \frac{p(x',y)\,p(x,y)}{p(y)}\,dy.$$

The rhs is symmetric in (x, x') and so is $h(x'|x)p(x)$.

The Markov chain generated with transition probability $h(x'|x)$ is **REVERSIBLE** and thus supports detailed balance.

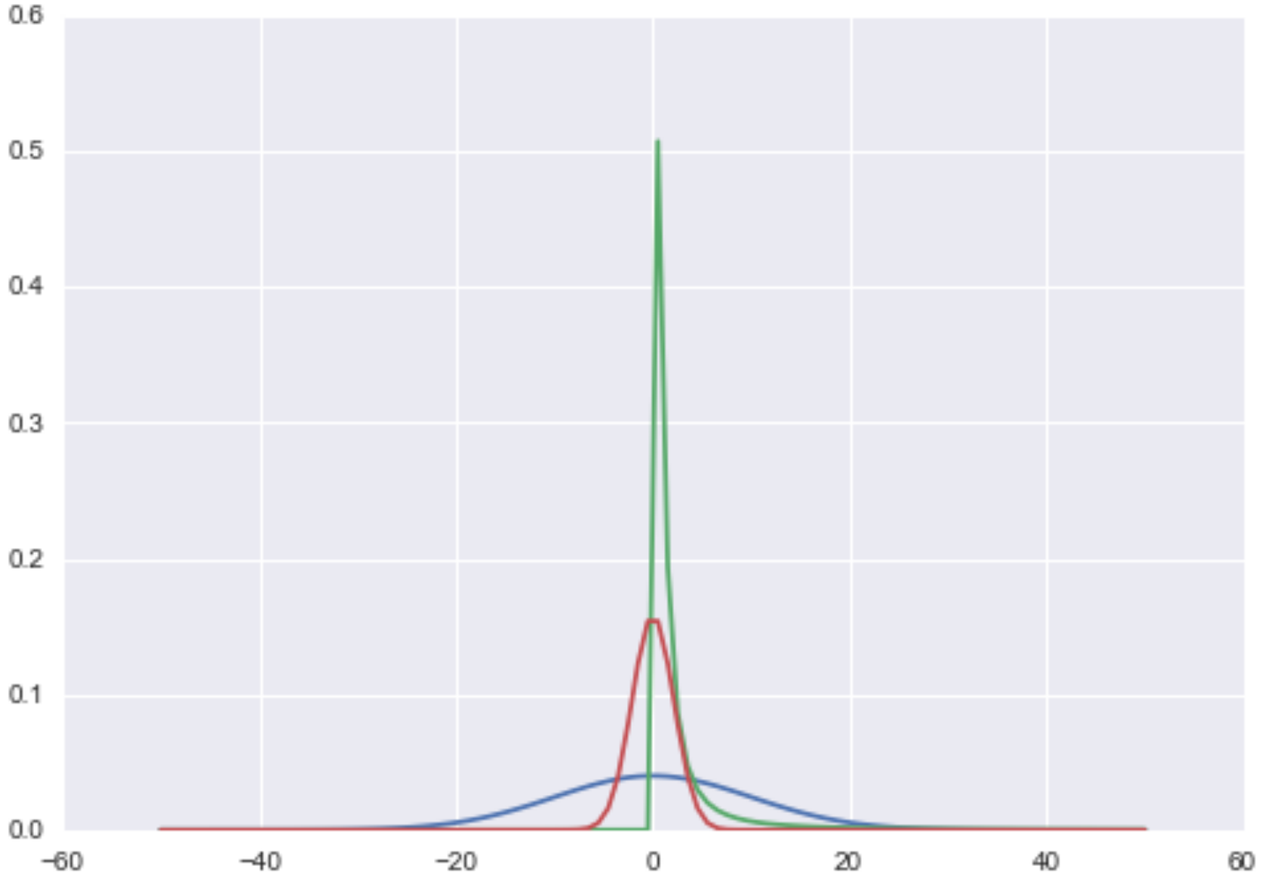
Bayesian

- sample is the data fixed
- parameter is stochastic, has prior and posterior distribution
- posterior: $p(\theta|y) = \frac{p(y|\theta)\,p(\theta)}{p(y)}$, can summarize via MAP
- just bayes rule: $posterior = \frac{likelihood \times prior}{evidence}$

- prior-predictive = evidence: $p(y) = E_{p(\theta)}[\mathcal{L}] = \int d\theta p(y|\theta)p(\theta)$ a normalization, irrelevant for sampling, useful for EB
- What if θ is multidimensional? Marginal posterior: $p(\theta_1|D) = \int d\theta_{-1} p(\theta|D).$
- posterior predictive: the distribution of a future data point y^*:

$$p(y^*|D = \{y\}) = E_{p(\theta|D)}[p(y|\theta)] = \int d\theta p(y^*|\theta)p(\theta|\{y\}).$$

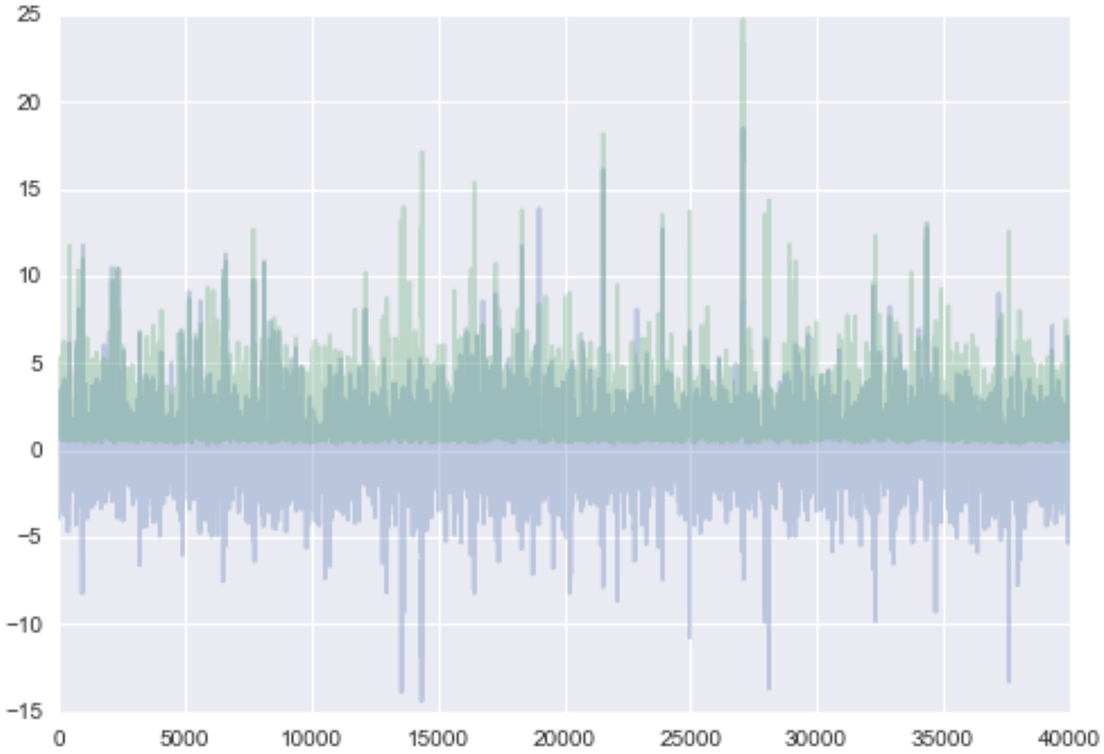
priors



- choose likelihoods with MAXENT
- choose priors as non-informative, e.g. uniform or Jeffreys
- better still: choose priors as weakly informative/regularizing
- helps with sampler performance

Bad->Good


7000
6000
5000
4000
3000
2000
1000
0
−1000
−2000
0
10000
20000
30000
40000
50000


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

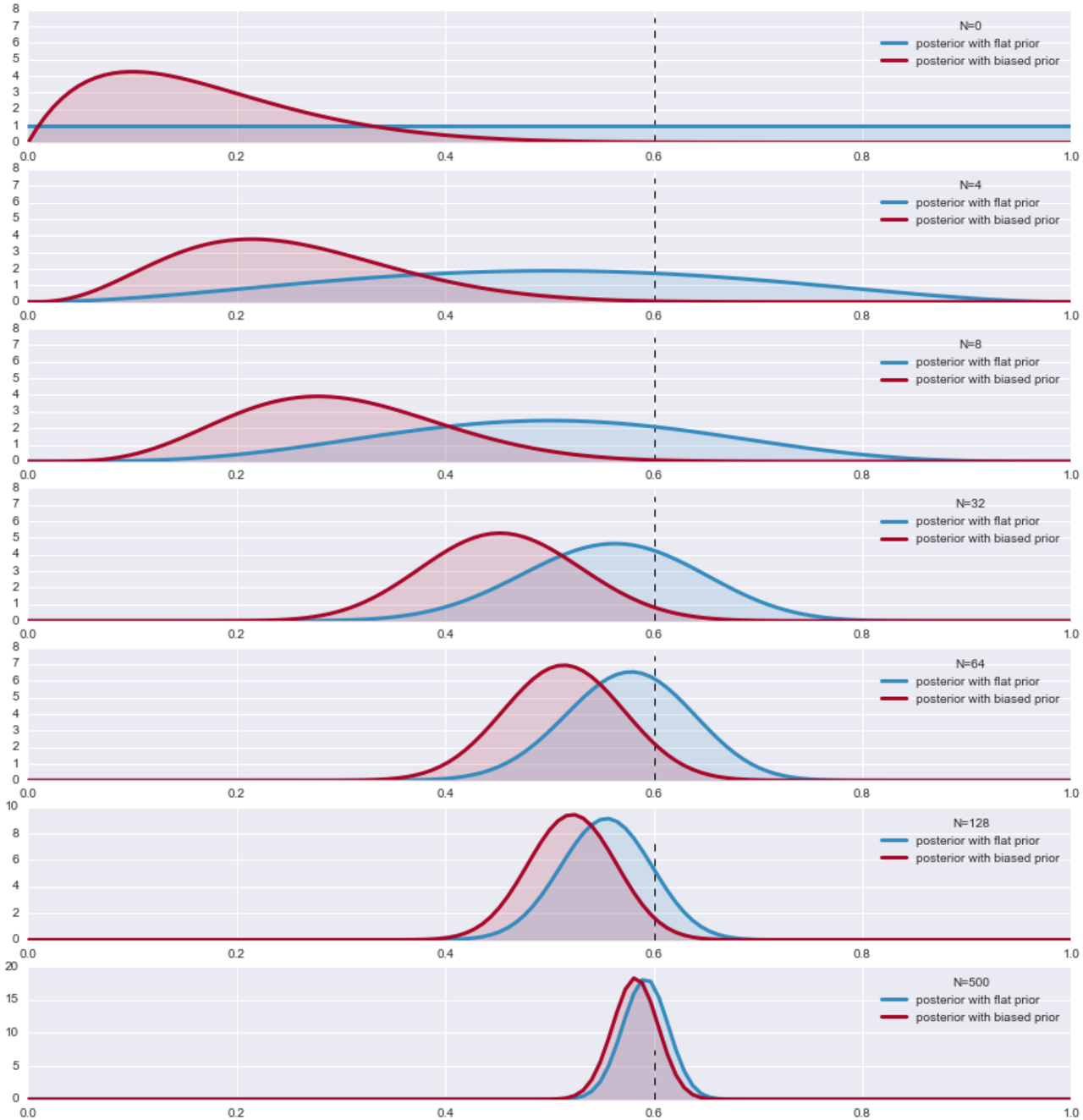
Data Overwhelms priors

Define $\kappa = \sigma^2/\tau^2$

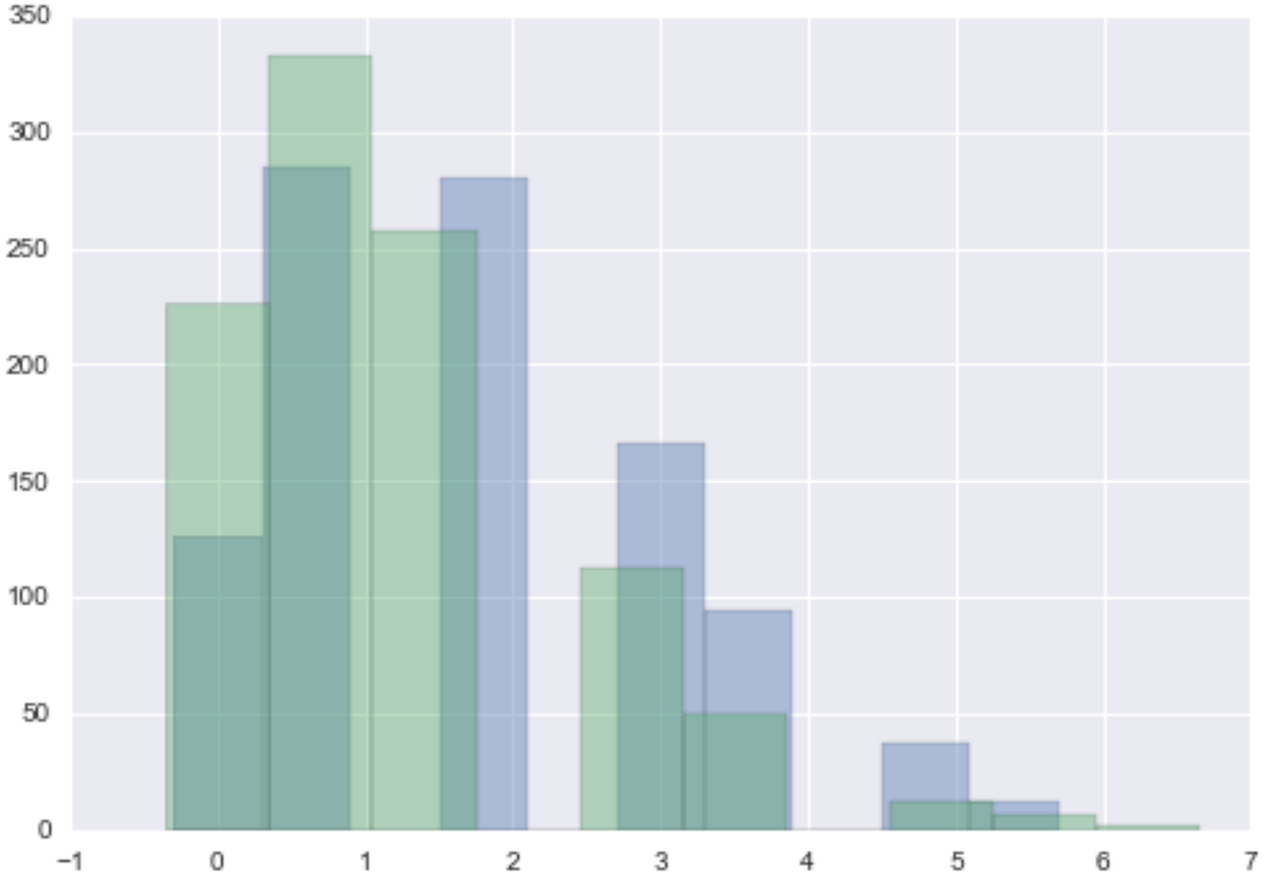
$$\mu_p = \frac{b}{a} = \frac{\kappa}{\kappa+n}\hat{\mu} + \frac{n}{\kappa+n}\bar{y}$$

$$\frac{1}{\tau_p^2} = \frac{1}{\tau^2} + \frac{n}{\sigma^2}.$$

- priors regularize data for small data
- but large data overwhelms priors



Posterior Predictives



$$p(y^*|D) = \int d\theta p(y^*|\theta)p(\theta|D)$$

Sampling easy (mothers poisson-gamma):

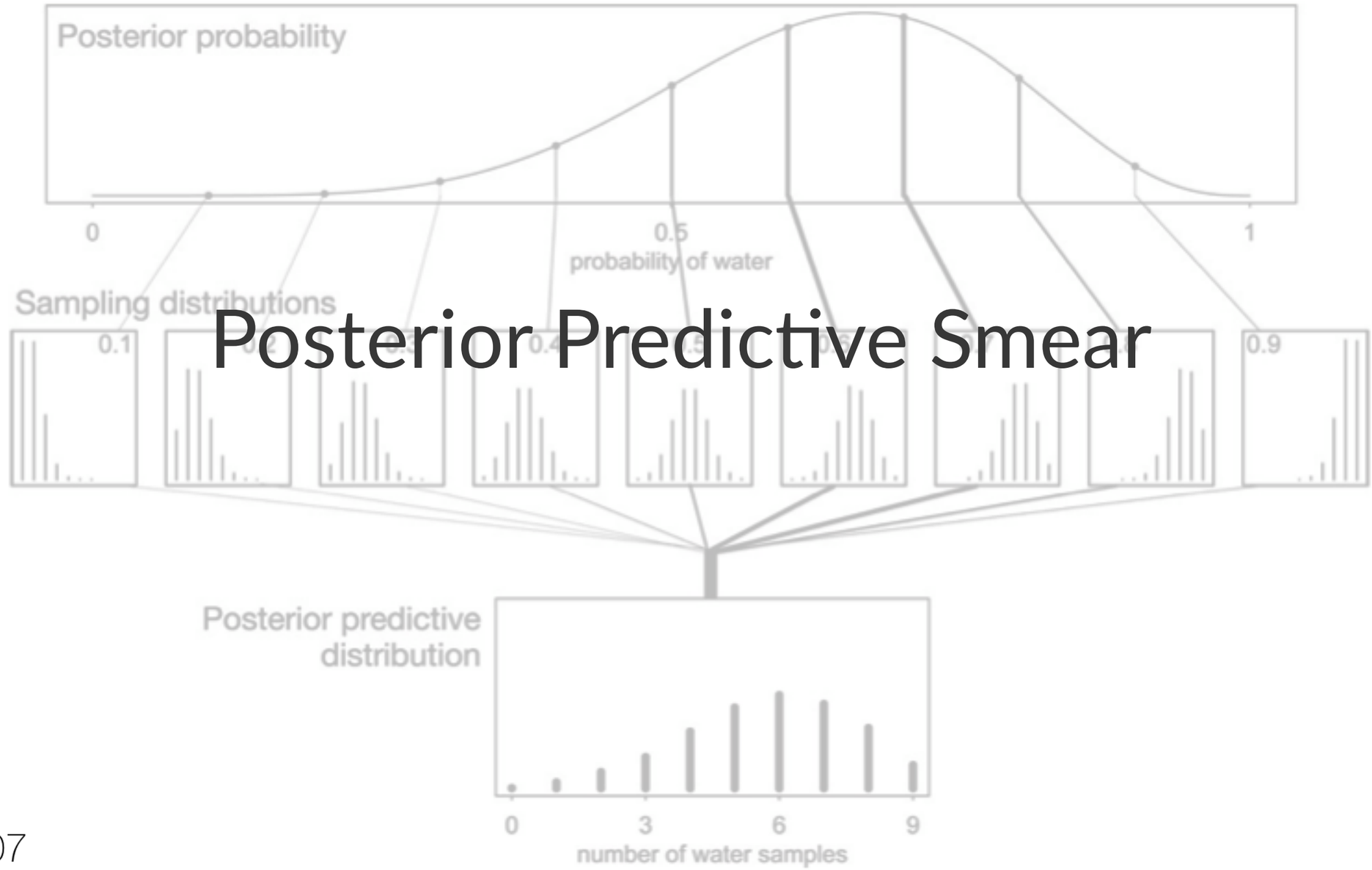
```
postpred1 = poisson.rvs(theta1trace)
postpred2 = poisson.rvs(theta2trace)
```

Exact: Negative Binomial (requires math):

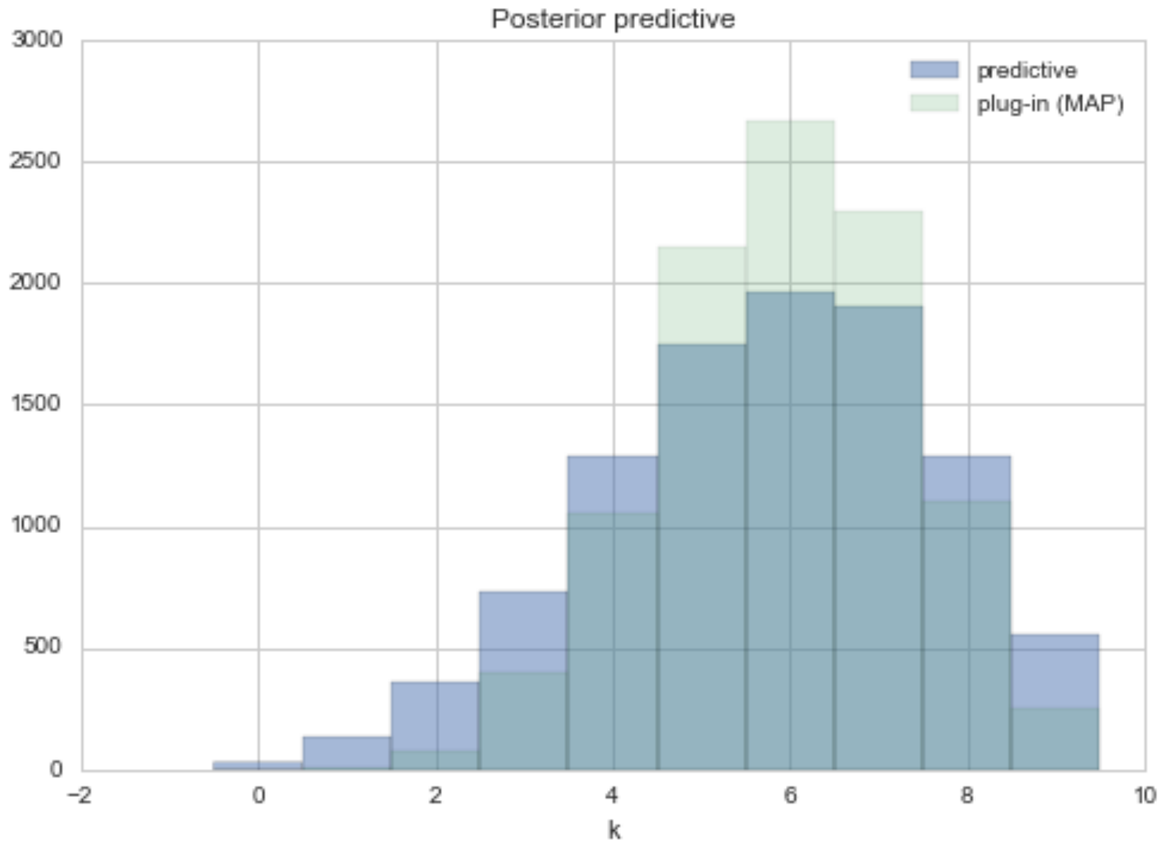
$$E[y^*] = \frac{(a + \sum y_i)}{(b+N)}$$

$$var[y^*] = \frac{(a + \sum y_i)}{(b+N)^2}(N + b + 1).$$

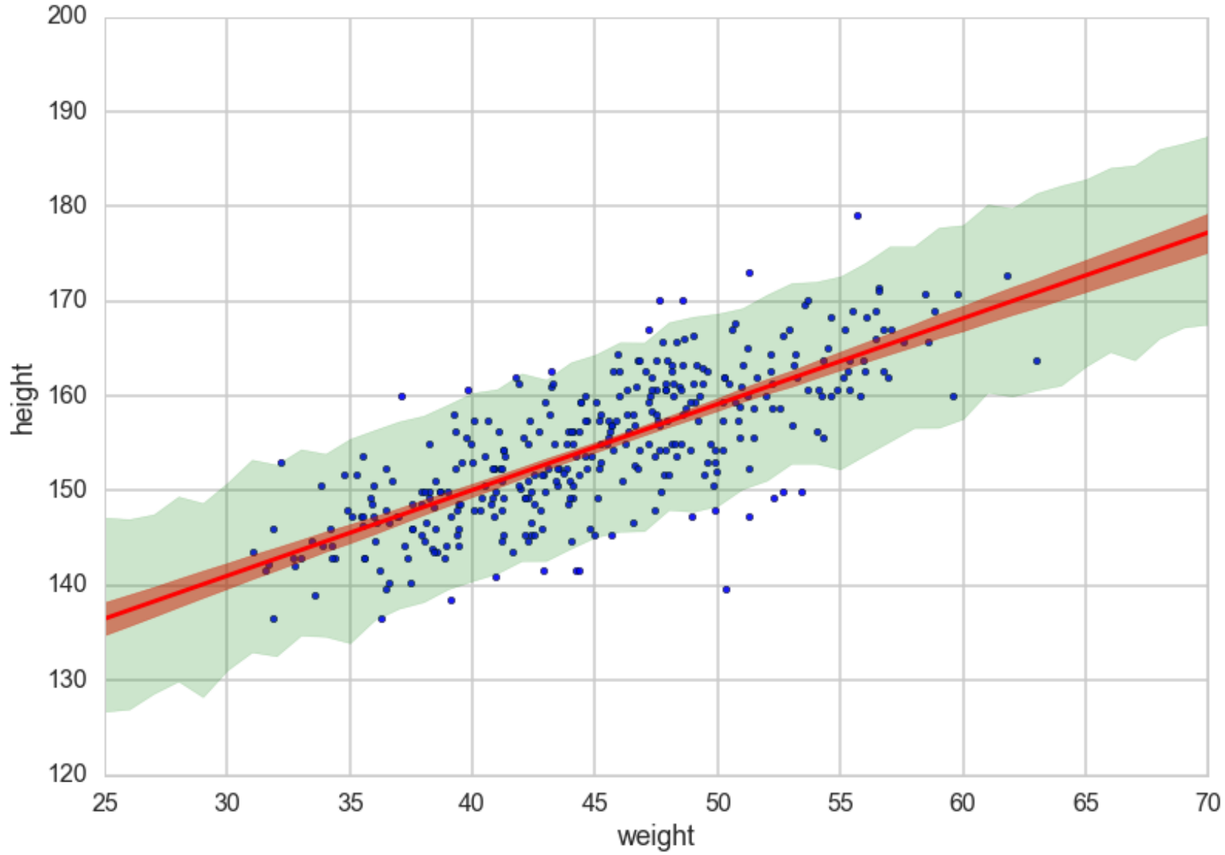
Posterior Predictive Smear



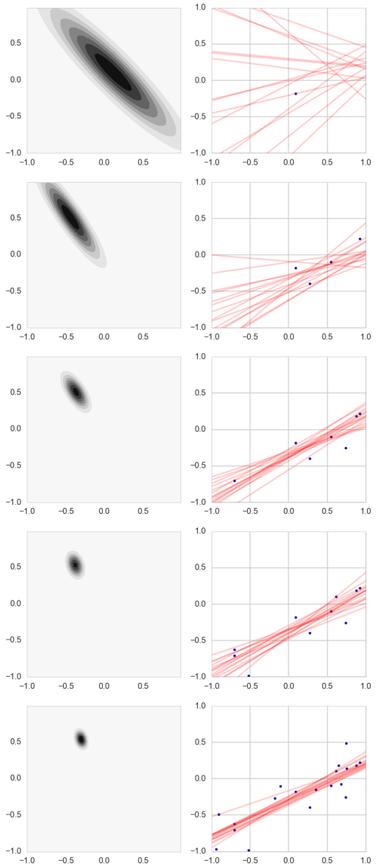
Posterior predictive vs sampling distrib (plugin approx)

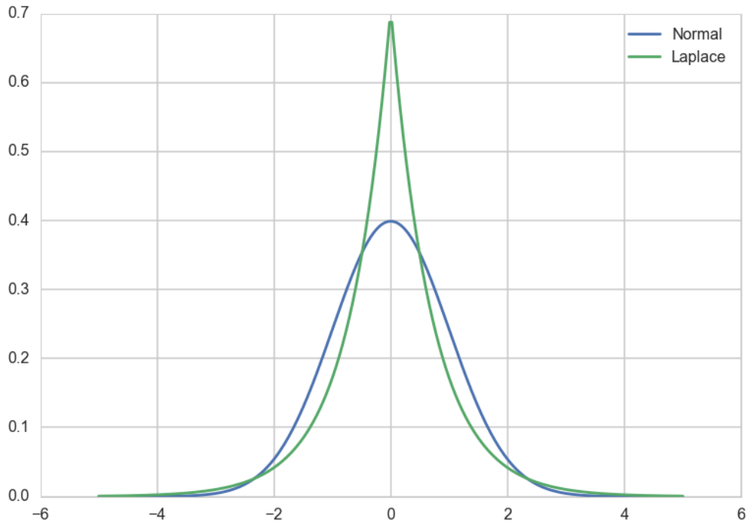


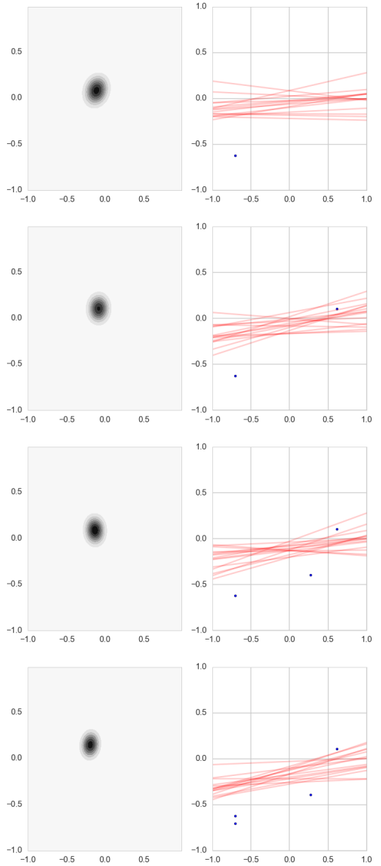
Bayesian Regression



Regression as a Bayesian Update; Regularization







Pymc3

Coal disasters Model:

$$y|\tau, \lambda_1, \lambda_2 \sim Poisson(r_t)$$

$$r_t = \lambda_1 \text{ if } t < \tau \text{ else } \lambda_2 \text{ for } t \in [t_l, t_h]$$
$$\tau \sim DiscreteUniform(t_l, t_h)$$

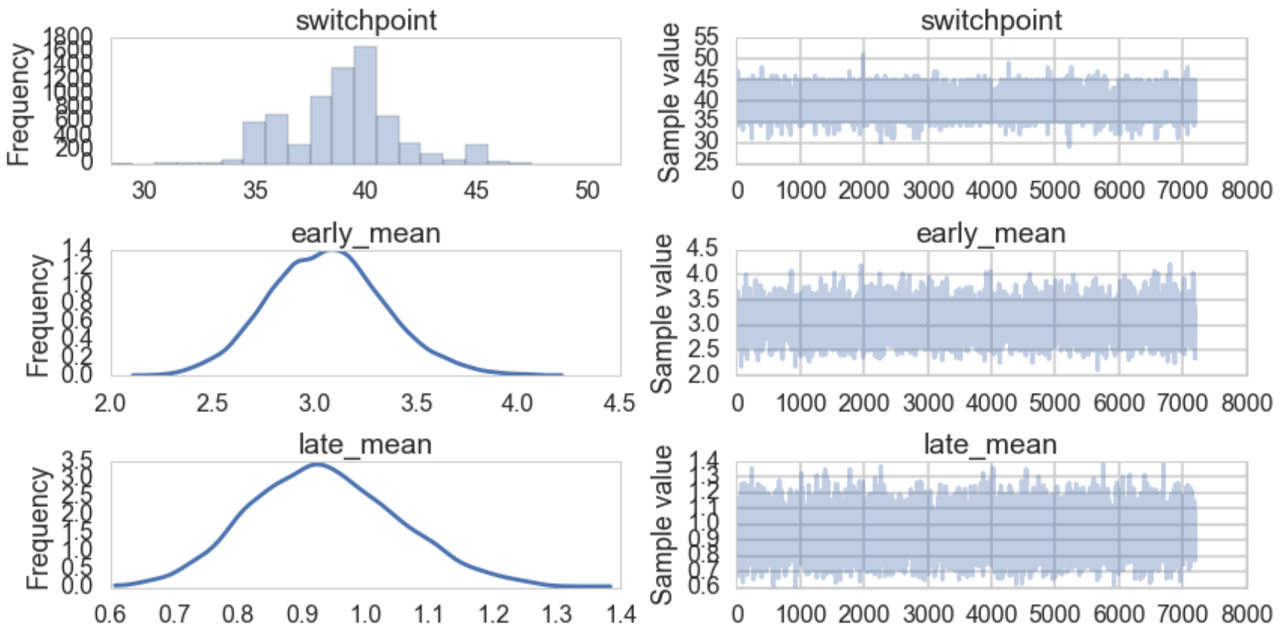
$$\lambda_1 \sim Exp(a)$$
$$\lambda_2 \sim Exp(b)$$

```
from pymc3.math import switch
with pm.Model() as coaldis1:
    early_mean = pm.Exponential('early_mean', 1)
    late_mean = pm.Exponential('late_mean', 1)
    switchpoint = pm.DiscreteUniform('switchpoint', lower=0, upper=n_years)
    rate = switch(switchpoint >= np.arange(n_years), early_mean, late_mean)
    disasters = pm.Poisson('disasters', mu=rate, observed=disasters_data)
```

```
with coaldis1:
    stepper=pm.Metropolis()
    trace = pm.sample(40000, step=stepper)
```

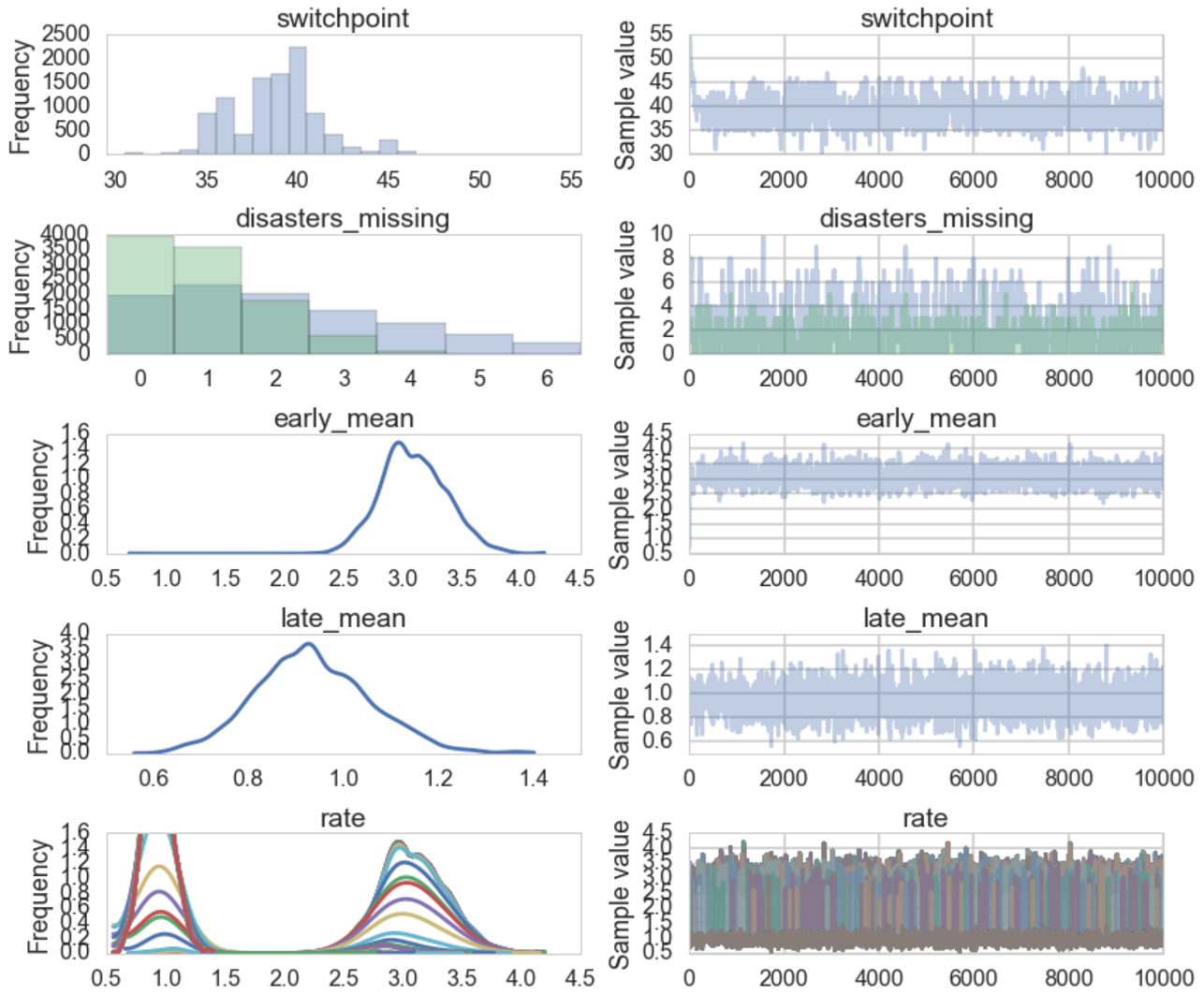
```
100%|██████████| 40000/40000 [00:12<00:00, 3326.53it/s] | 229/40000 [00:00<00:17, 2289.39it/s]
```

```
>>>coaldis1.vars #stochastics
[early_mean_log_, late_mean_log_, switchpoint]
>>>coaldis1.deterministics #deterministics
[early_mean, late_mean]
>>>coaldis1.observed_RVs
[disasters]
>>>ed=pm.Exponential.dist(1)
<class 'pymc3.distributions.continuous.Exponential'>
>>>ed.random(size=10)
array([ 1.18512233,  2.45533355,  0.04187961,  3.32967837,  0.0268889 ,
        0.29723148,  1.30670324,  0.23335826,  0.56203427,  0.15627659])
>>>type(switchpoint), type(early_mean)
(pymc3.model.FreeRV, pymc3.model.TransformedRV)
>>>switchpoint.logp({'switchpoint':55,
    'early_mean_log_':1, 'late_mean_log_':1})
array(-4.718498871295094)
```

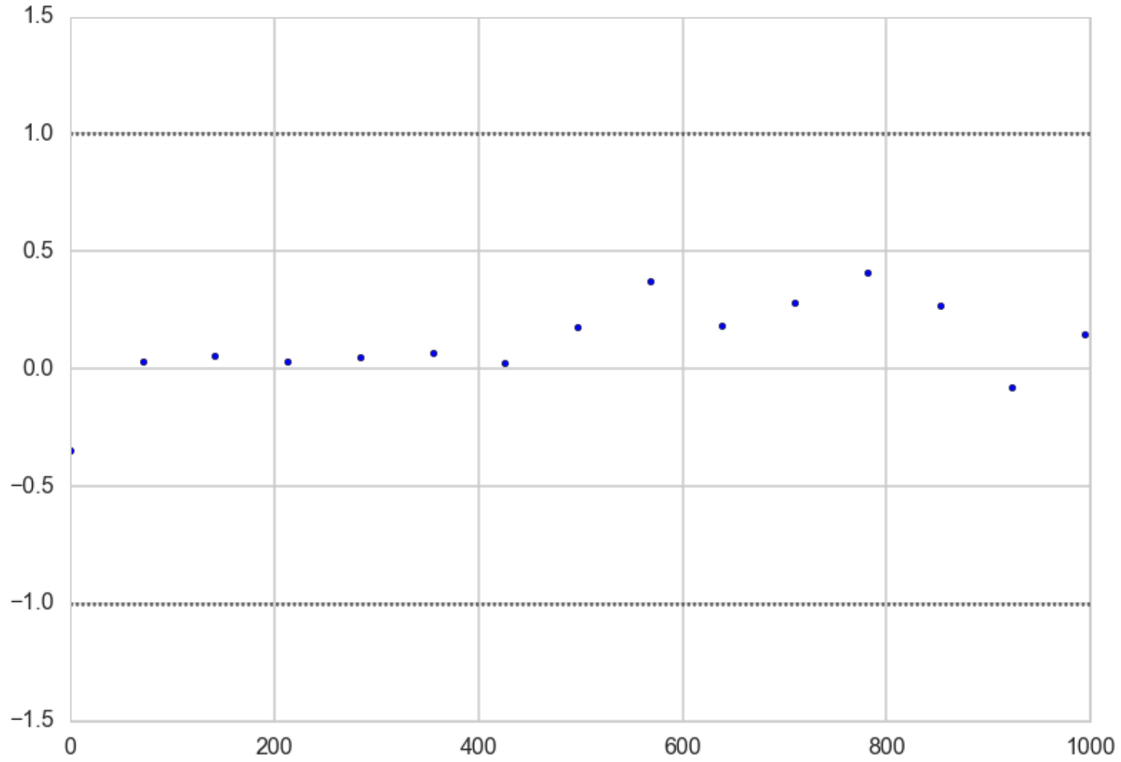


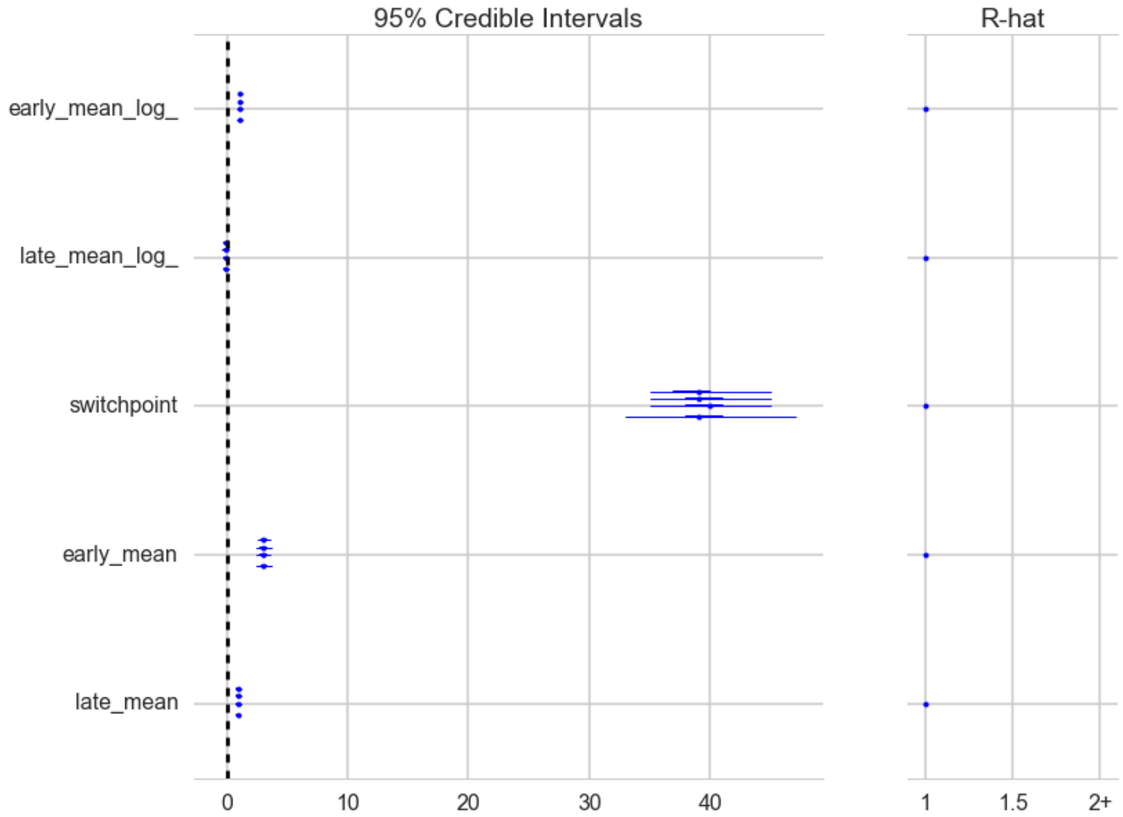
```python
with pm.Model() as missing_data_model:
    switchpoint = pm.DiscreteUniform('switchpoint', lower=0, upper=len(disasters_masked))
    early_mean = pm.Exponential('early_mean', lam=1.)
    late_mean = pm.Exponential('late_mean', lam=1.)
    idx = np.arange(len(disasters_masked))
    rate = pm.Deterministic('rate', switch(switchpoint >= idx, early_mean, late_mean))
    disasters = pm.Poisson('disasters', rate, observed=disasters_masked)

with missing_data_model:
    stepper=pm.Metropolis()
    trace_missing = pm.sample(10000, step=stepper)

pm.summary(trace_missing, varnames=['disasters_missing'])
```

```
disasters_missing:

  Mean             SD               MC Error         95% HPD interval
  ------------------------------------------------------------------


  2.189            1.825            0.078            [0.000, 6.000]
  0.950            0.980            0.028            [0.000, 3.000]

  Posterior quantiles:
  2.5              25               50               75               97.5
  |--------------|===============|===============|--------------|

  0.000            1.000            2.000            3.000            6.000
  0.000            0.000            1.000            2.000            3.000
```



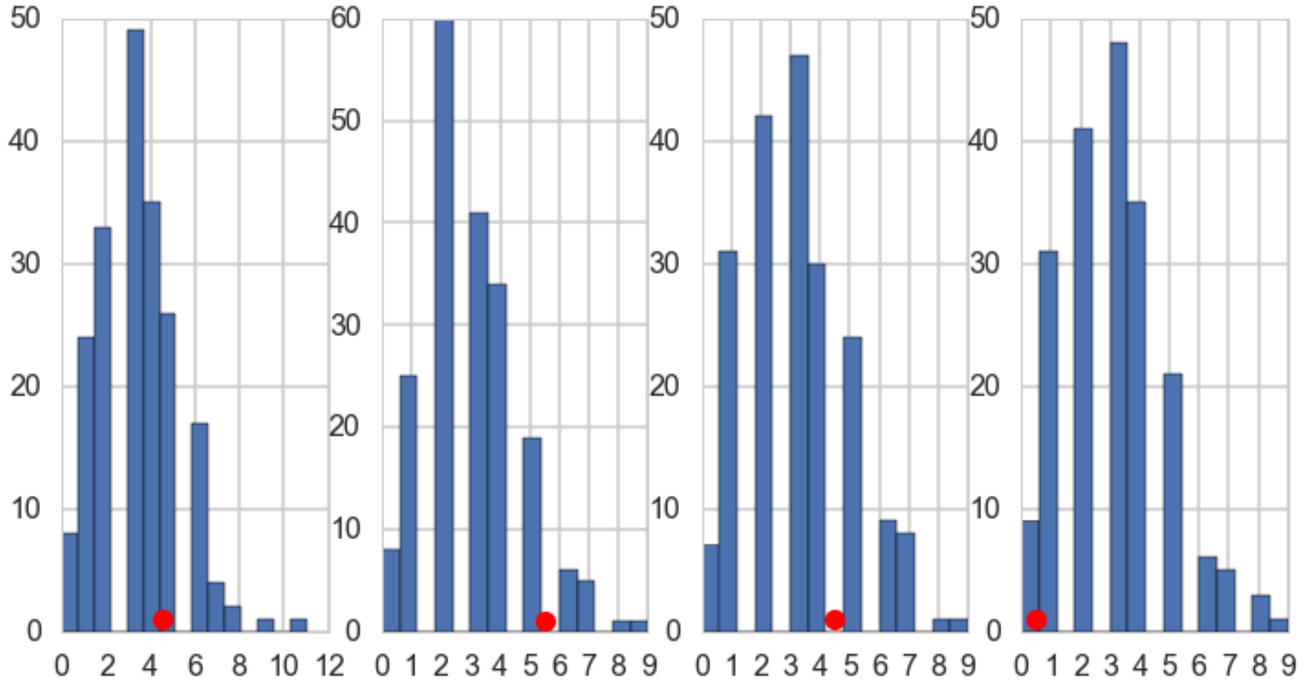
Gewecke and Gelman Rubin



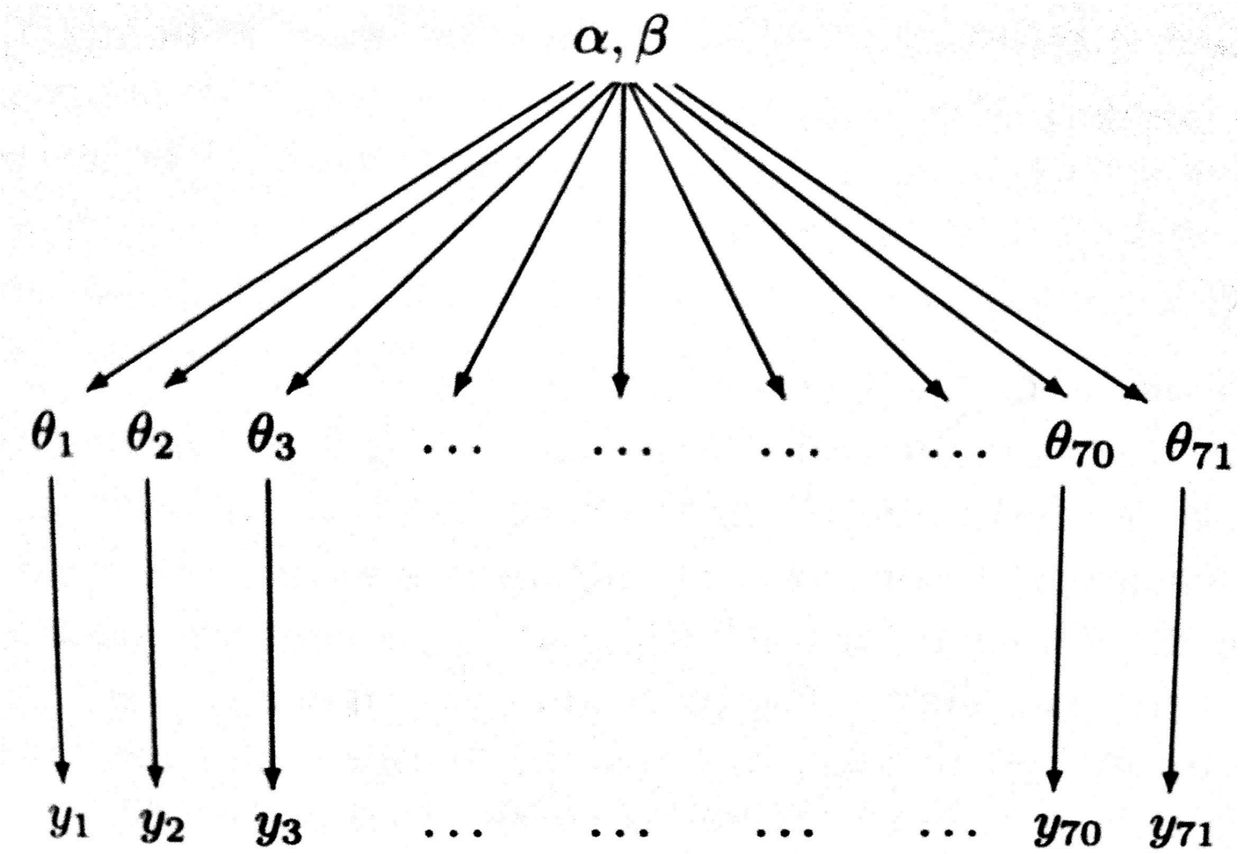


Posterior Predictive Checks

```
with coaldis1:
    sim = pm.sample_ppc(t2, samples=200)
```



Hierarchical models



Key Idea: Share statistical strength

- Some **units** (experiments) statistically more robust
- Non-robust experiments have smaller samples or outlier like behavior
- Borrow strength from all the data as a whole through the estimation of the hyperparameters
- **regularized partial pooling model** in which the "lower" parameters (θs) tied together by "upper level" hyperparameters.

First idea: estimate directly from data

Posterior-predictive distribution, as a function of upper level parameters $\eta = (\alpha, \beta)$.

$$p(y^*|D, \eta) = \int d\theta \, p(y^*|\theta) \, p(\theta|D, \eta)$$

A likelihood with parameters η and simply use maximum-likelihood with respect to η to estimate these η using our "data" y^*

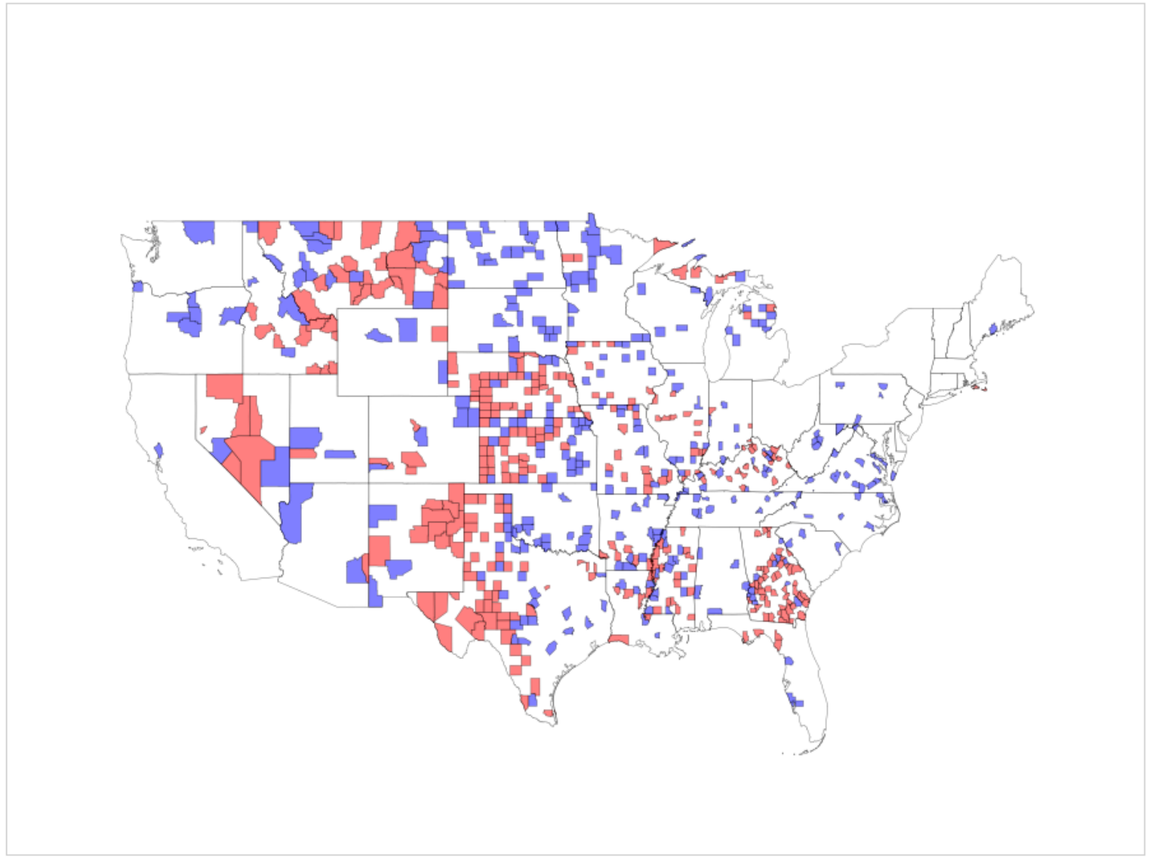
Full Sampling

- Fix α and β, we have a Gibbs step for all of the θ_is
- For α and β, everything else fixed, use stationary metropolis step, as conditionals not simply sampled distributions
- when we sample for α, we will propose a new value using a normal proposal, while holding all the θs and β constant at the old value. ditto for β.
- OR just specify in pymc and go!

Howto Hierarchical models

- a DAG, with observations at the bottom of a tree, next layer intermediate parameters, upper layers hyper-parameters
- sample conditionals from parents up the tree.
- the y_j were exchangeable since we had no additional information about experimental conditions.
- if specific groups of experiments came from specific laboratories, assume experiments interchangeable if from the same lab.

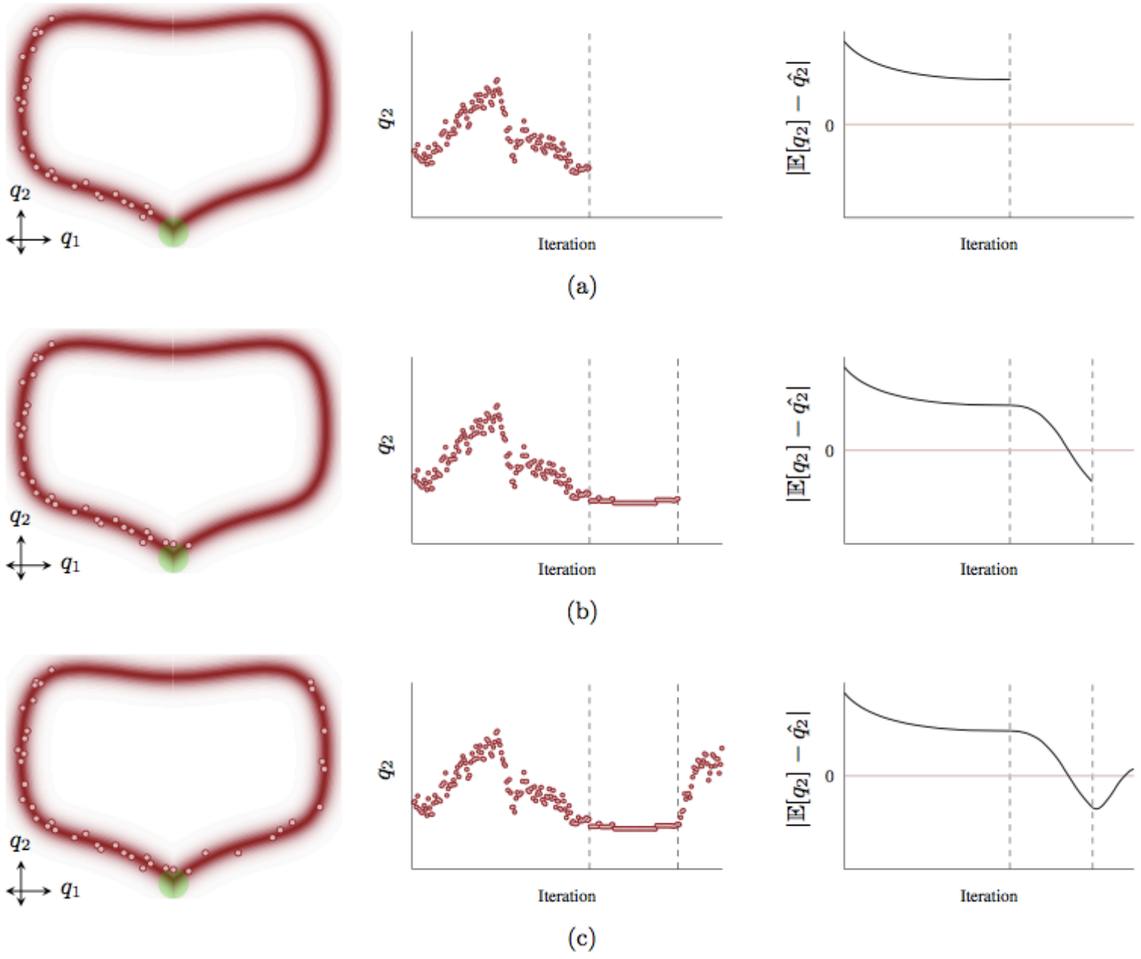
Homework



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	$p(\theta, \eta\|D) \propto p(D\|\theta)p(\theta\|\eta)p(\eta)$

Problems with MCMC

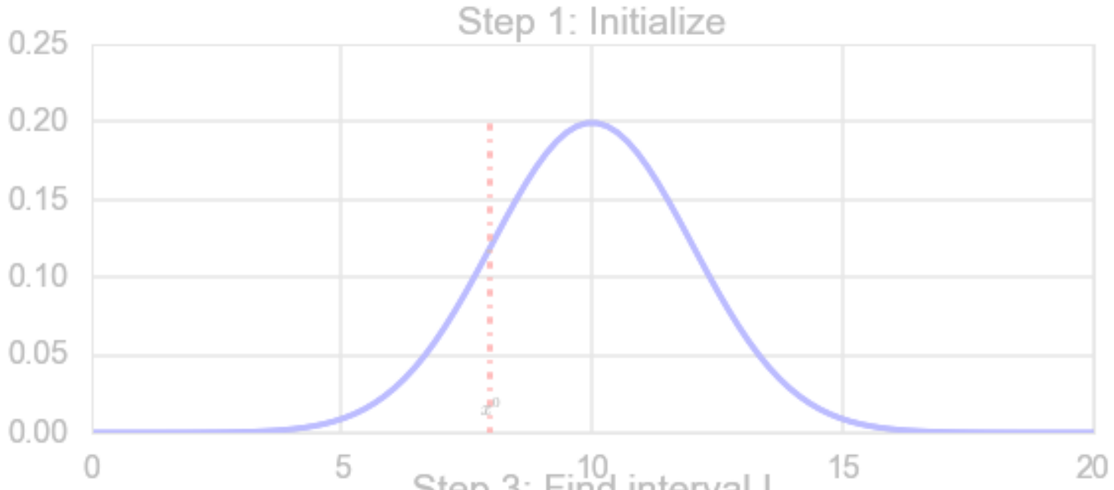


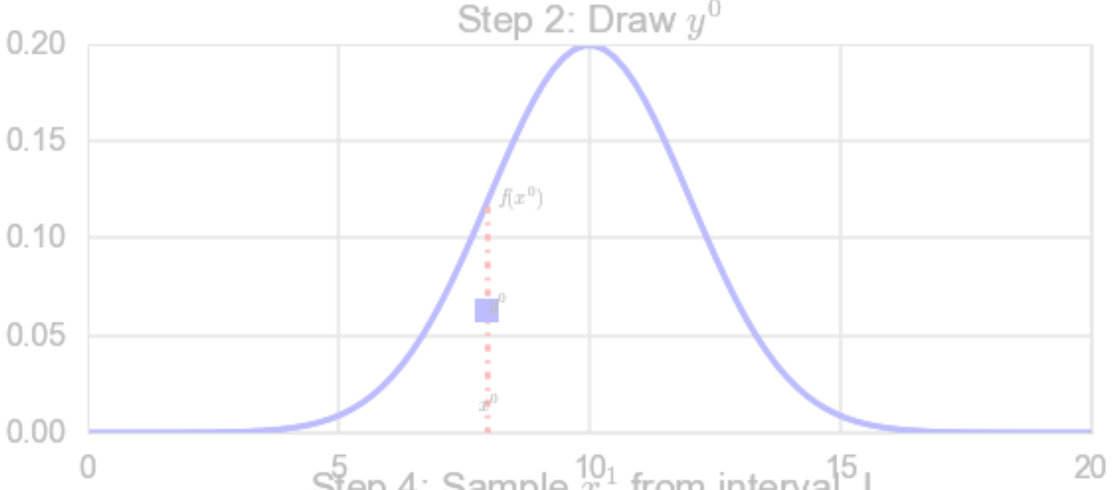
- overshoot and oscillate at pinches
- need to specify step step sizes
- large steps go outside typical set and are not accepted
- small steps accepted but go nowhere
- large correrlations

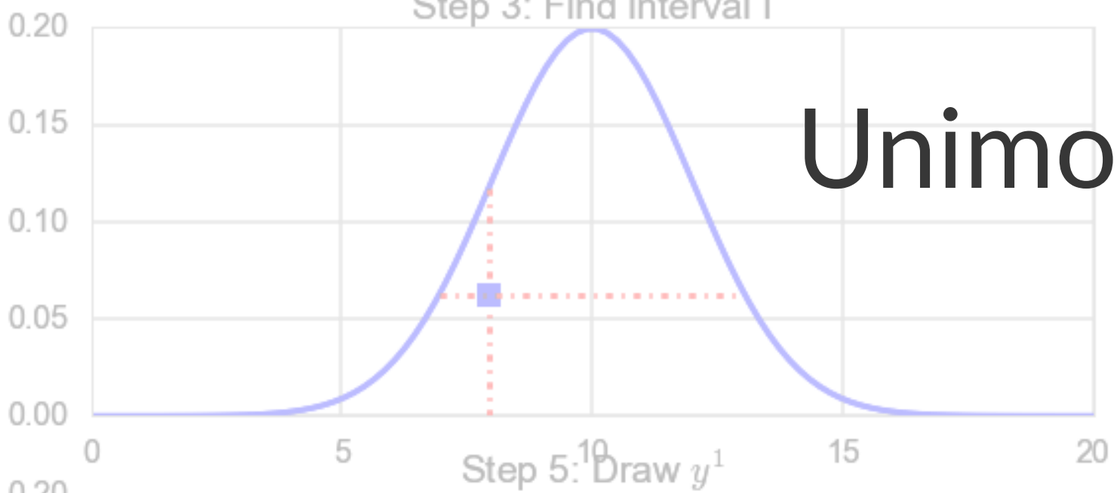
SLICE

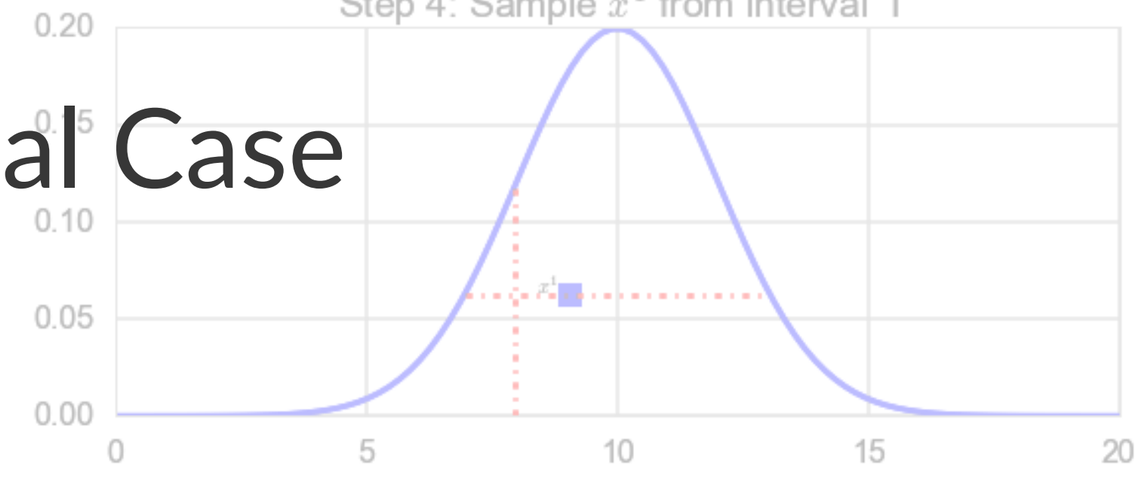
to the rescue

- Pick an initial point x_0 from our posterior
- Draw y_0 from U(0, f(x_0))
- Repeat for N samples
  - Select the interval (e.g. stepping out, etc)
  - Sample x_i from that interval (e.g. shrinkage)
  - Draw y_i from U(0, f(x_i))

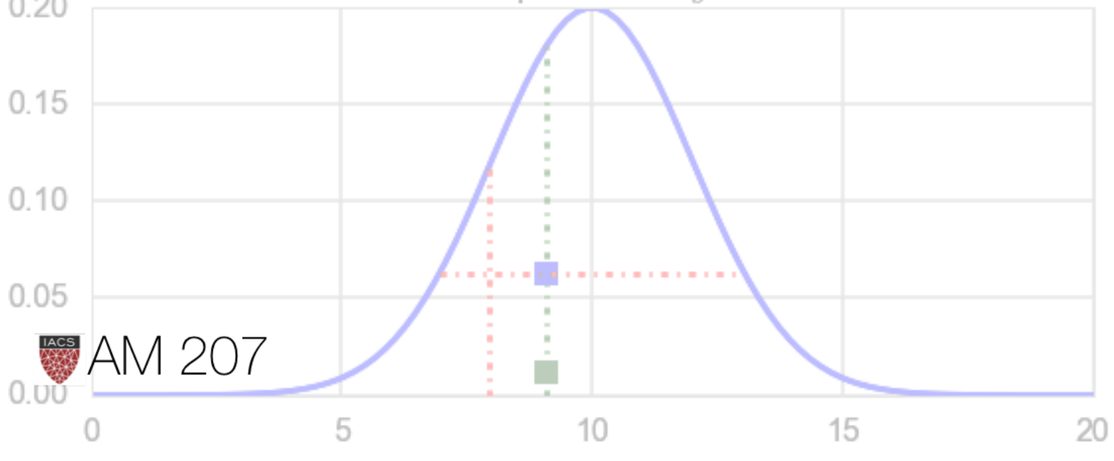




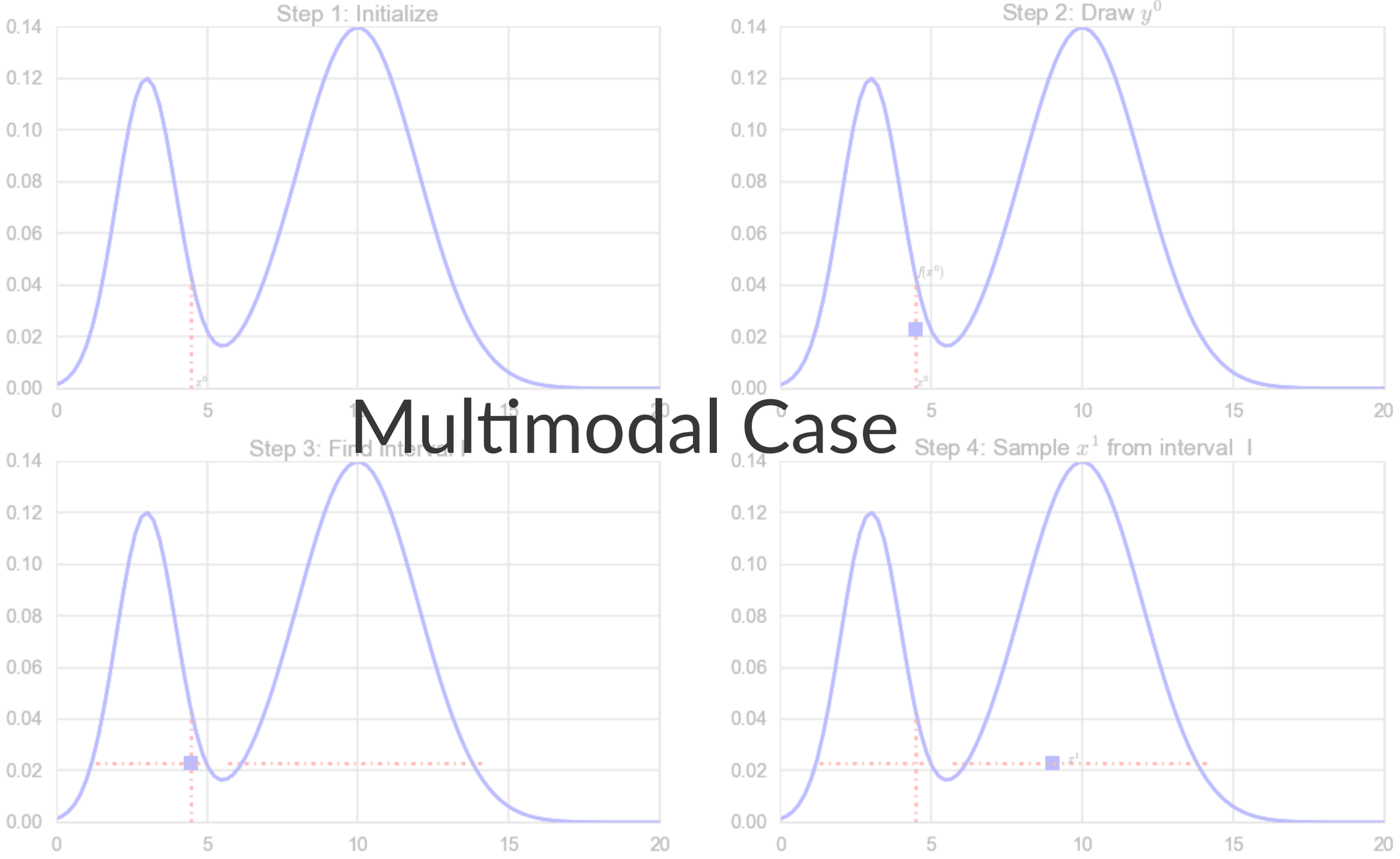




Unimodal Case



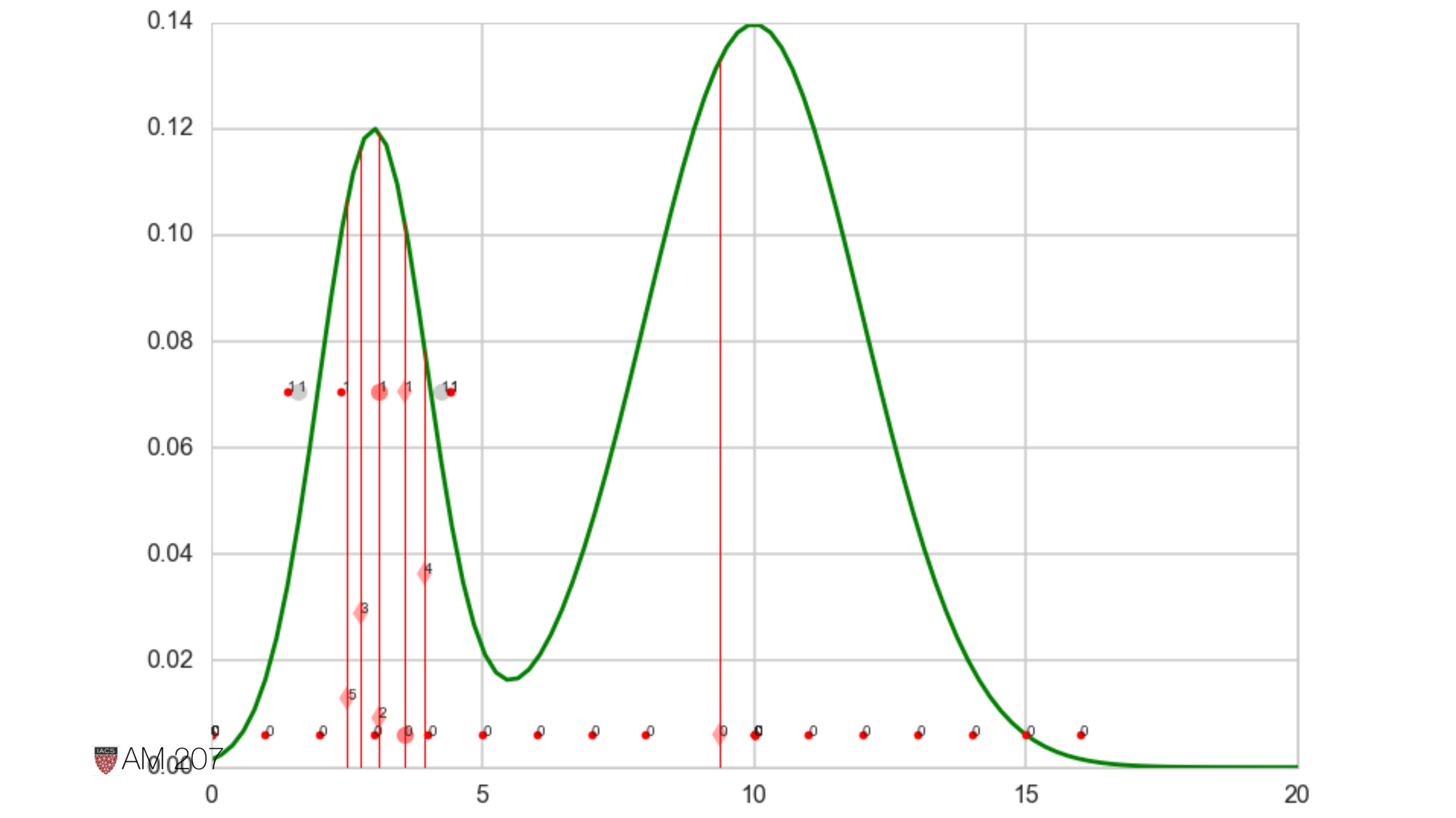
Multimodal Case



Stepping Out

- set w width and draw u ~ Unif(0,1)
- set L = $x^{(0)} - wu, R = L + w$ (so $x^{(0)}$ lies in [L, R])
- while y < f(L) (here's where we extend left interval) L = L - w
- while y < f(R) (here's where we extend the right interval) R = R + w

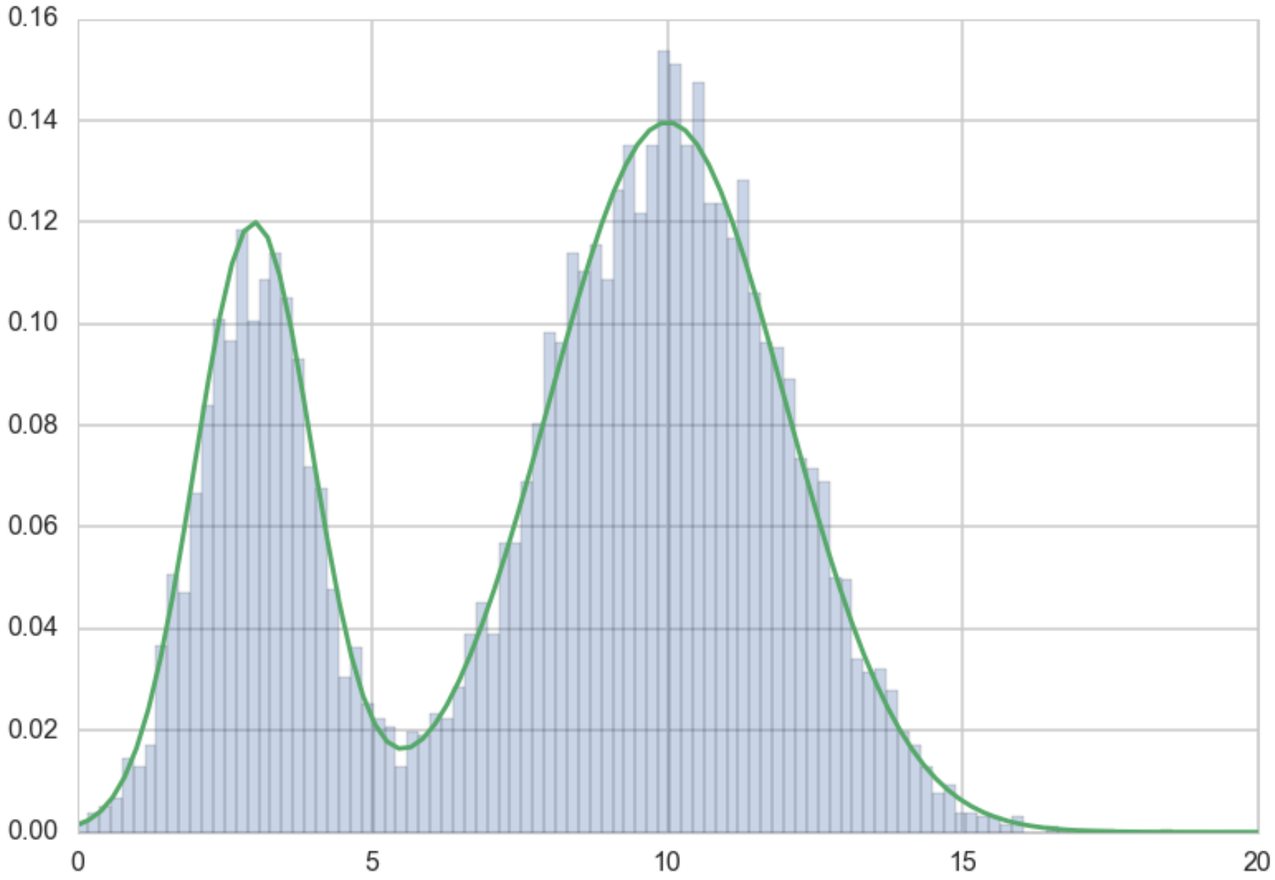
The final interval will be larger than S.


0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20

Shrinkage

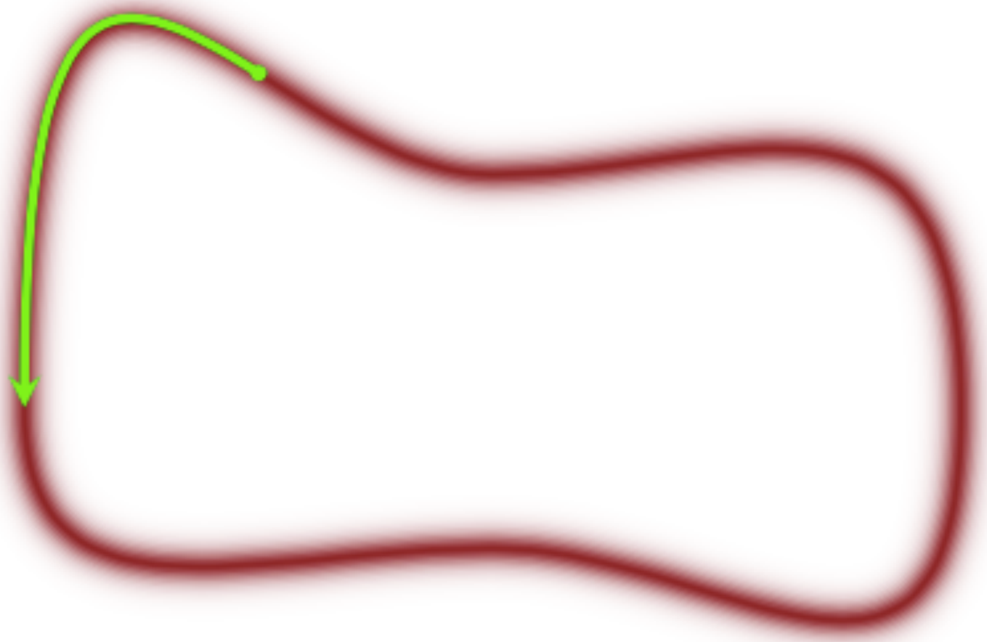
- start with interval $I = (L, R)$
- current sample is $x^{(k)}$ and $y^{(k)}$
- repeat until loop exits
  - sample x^* uniformly from (L, R)
  - if $y^{(k)} < f(x^*)$

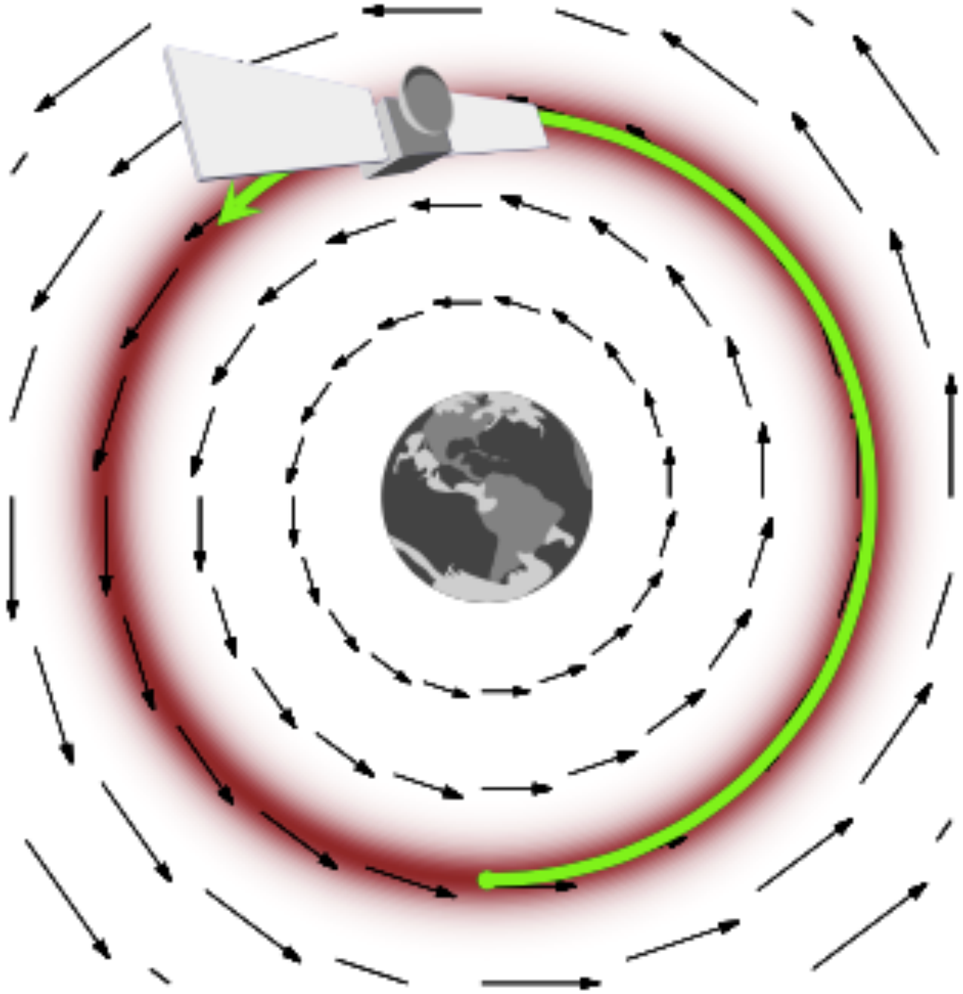
```
w=1.0
L=0; R=0;
x_prev = np.random.uniform(low=0, high=17)
iters=10000
trace=[]
kmax=1
for k in range(iters):
    y_samp = np.random.uniform(low=0, high=fun(x_prev))
    # widen left
    U = np.random.rand()
    L=x_prev-U*w
    R=x_prev+w*(1.0-U)
    while fun(L)>y_samp:
        L = L-w
    while fun(R)>y_samp:
        R = R+w
    #now propose new x on L,R

    while 1:
        x_prop= np.random.uniform(low=L, high=R)
        if k <= kmax:
            print("L,R, xprop", L, R, x_prop)
        if y_samp < fun(x_prop):
            x_prev = x_prop
            trace.append(x_prop)
            break
        elif x_prop > x_prev:
                R = x_prop
        elif x_prop < x_prev:
                L = x_prop
```



Hamiltonian Monte Carlo

Need a Coherent Glide





Balance between gravity and momentum in a rocket provides it

Now, like in annealing, let
$p(p,q) = e^{-Energy}$

Carry out an augmentation with an additional momentum with the energy **Hamiltonian**

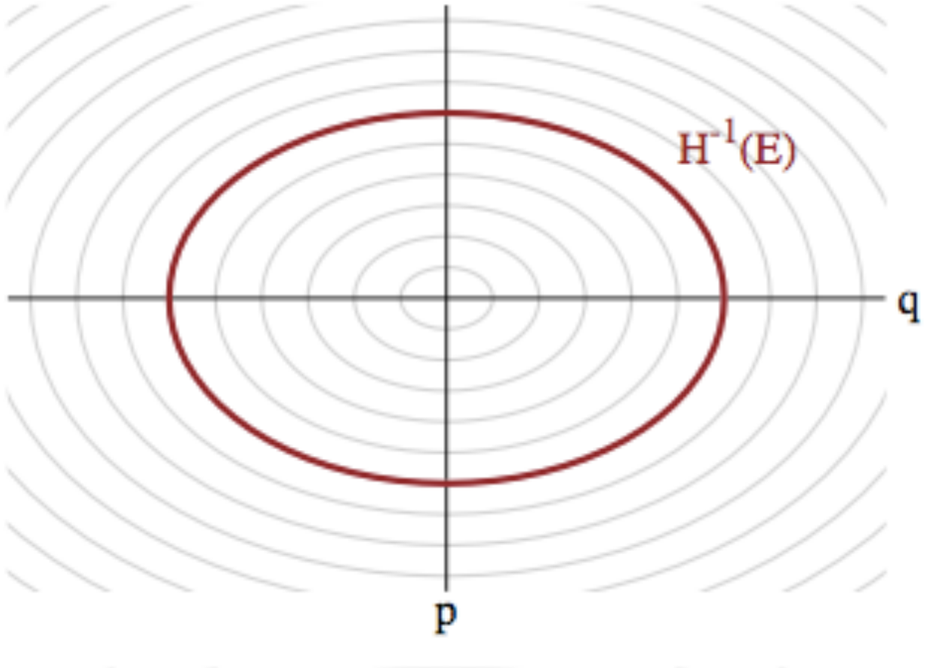
$$H(p,q) = \frac{p^2}{2m} + V(q)$$

Typical Set decomposes into level sets of constant probability

The energy **Hamiltonian**

$$H(p,q) = \frac{p^2}{2m} + V(q) = E_i,$$

with E_i constants (constant energies) for each level-set foliate and where the **potential energy** $V(q) = -log(p(q))$ replaces the energy term we had earlier in simulated annealing.



$$p(p,q) = e^{-H(p,q)} = e^{-K(p,q)}e^{-V(q)} = p(p|q)p(q)$$

and thus: $H(p,q) = -log(p(p,q)) = -logp(p|q) - logp(q)$

The choice of a kinetic energy term then is the choice of a conditional probability distribution over the "augmented" momentum which ensures that

$$\int dp p(p,q) = \int dp p(p|q)p(q) = p(q)\int p(p|q)dp = p(q).$$

Momentum resampling (thruster fire) moves us between level sets