ENM 360: Introduction to Data-driven Modeling

Lecture #24: Sampling methods

Paris Perdikaris
November 19, 2020

Penn
UNIVERSITY *of* PENNSYLVANIA

The Metropolis algorithm for Bayesian inference

Goal: We want to sample from

$$p(\theta \mid y) = \frac{f(y \mid \theta)\pi(\theta)}{m(y)}.$$

Typically, we don't know $m(y)$.

The notation is a bit more complicated, but the set up is the same.

We'll approach it a bit differently, but the idea is exactly the same.

We know $\pi(\theta)$ and $f(y \mid \theta)$, so we can can draw samples from these.

Our notation here will be that we assume parameter values $\theta_1, \theta_2, \ldots, \theta_s$ which are drawn from $\pi(\theta)$.

We assume a new parameter value comes in that is θ^*.

The Metropolis algorithm for Bayesian inference

The Metropolis algorithm proceeds as follows:

1. Sample $\theta^* \sim J(\theta \mid \theta^{(s)})$.
2. Compute the acceptance ratio (r):

$$r = \frac{p(\theta^*|y)}{p(\theta^{(s)}|y)} = \frac{p(y \mid \theta^*)p(\theta^*)}{p(y \mid \theta^{(s)})p(\theta^{(s)})}.$$

3. Let

$$\theta^{(s+1)} = \begin{cases} \theta^* & \text{with prob min(r,1)} \\ \theta^{(s)} & \text{otherwise.} \end{cases}$$

Remark: Step 3 can be accomplished by sampling $u \sim \text{Uniform}(0, 1)$ and setting $\theta^{(s+1)} = \theta^*$ if $u < r$ and setting $\theta^{(s+1)} = \theta^{(s)}$ otherwise.

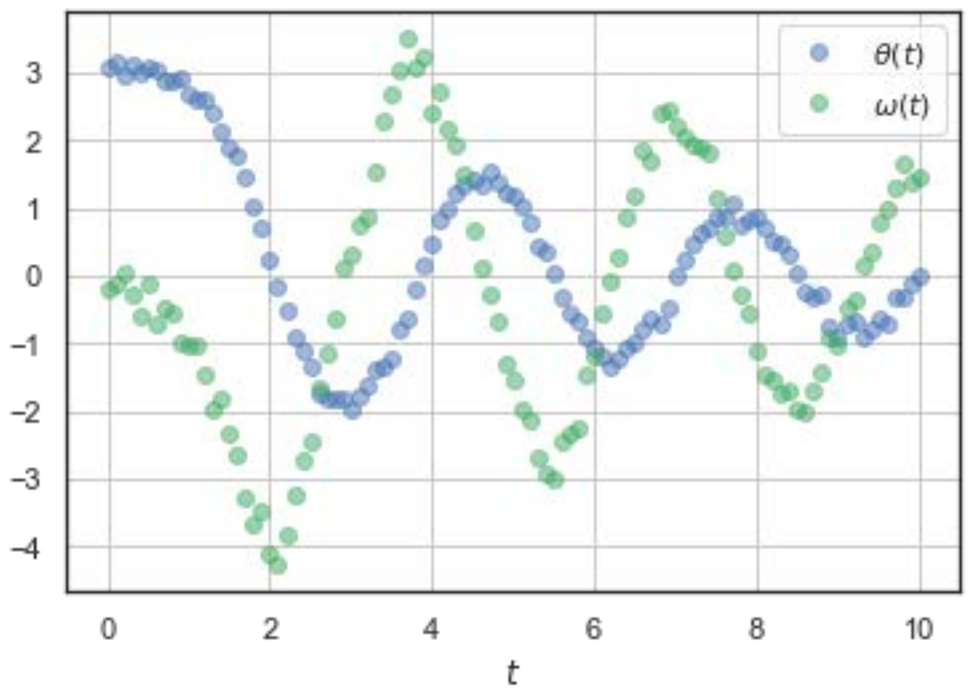
Bayesian calibration of dynamical systems

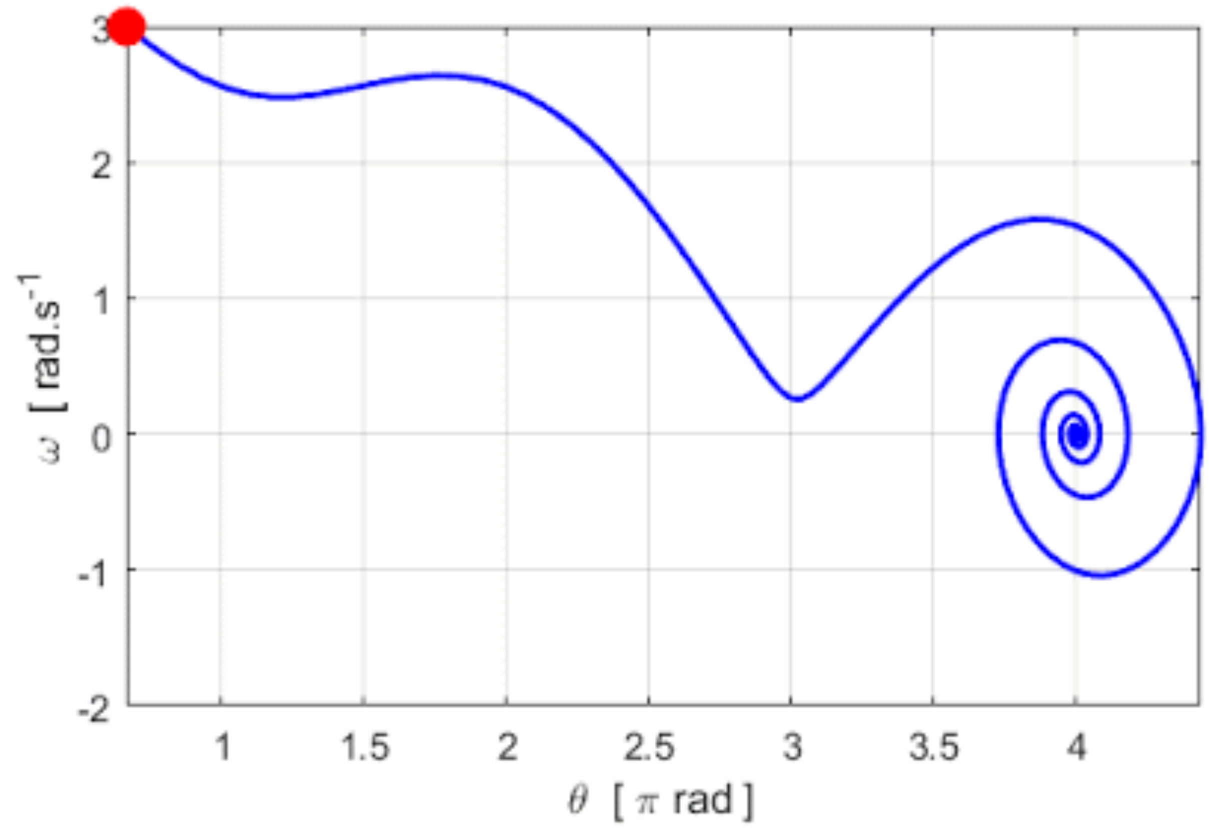
Example: Damped pendulum

$$\frac{d\theta}{dt} = \omega,$$
$$\frac{d\omega}{dt} = -b\omega - c\sin(\theta).$$

Given some noisy time-series data

$$\mathcal{D} := \{\theta(t_i), \omega_i(t_i)\}, \quad i = 1, \ldots, n$$

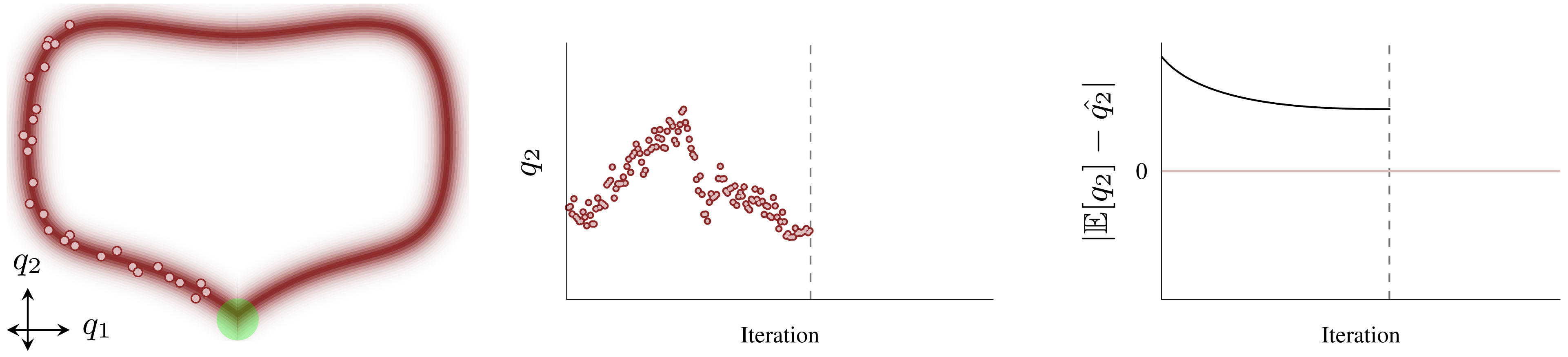




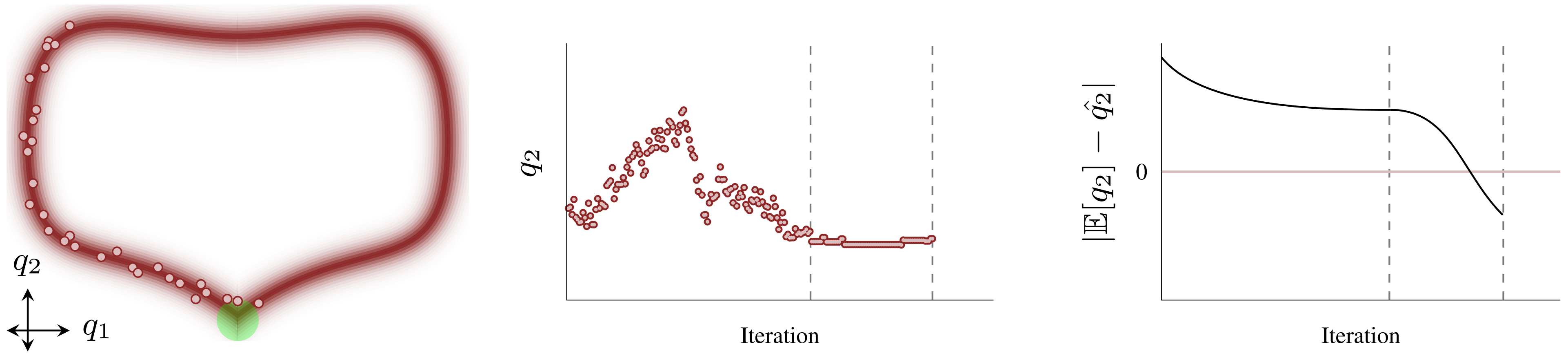
Infer a posterior distribution over the unknown parameters

$$p(b, c|\mathcal{D})$$

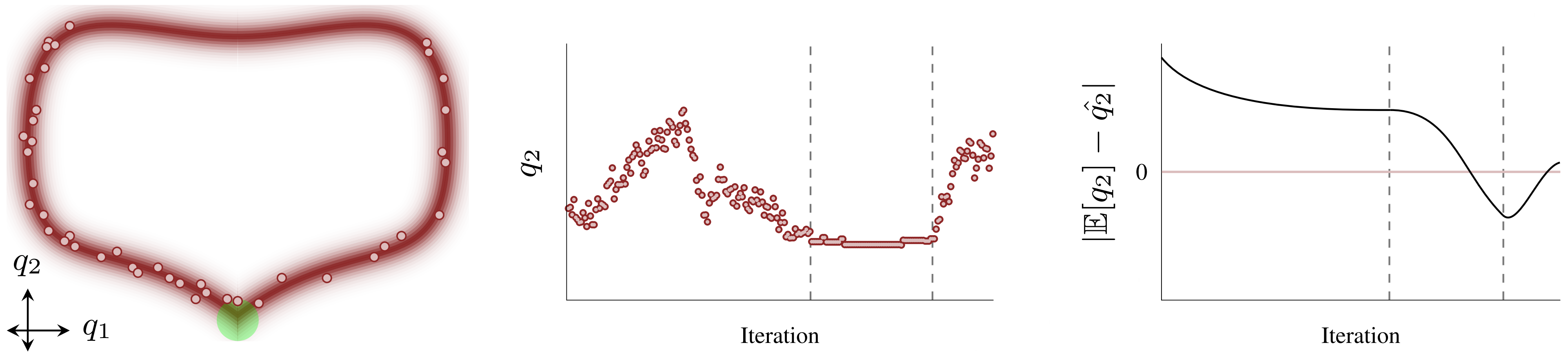
Hamiltonian Monte Carlo



(a)



(b)



(c)

In practice, pathological regions of the typical set usually cause Markov chains to get "stuck".

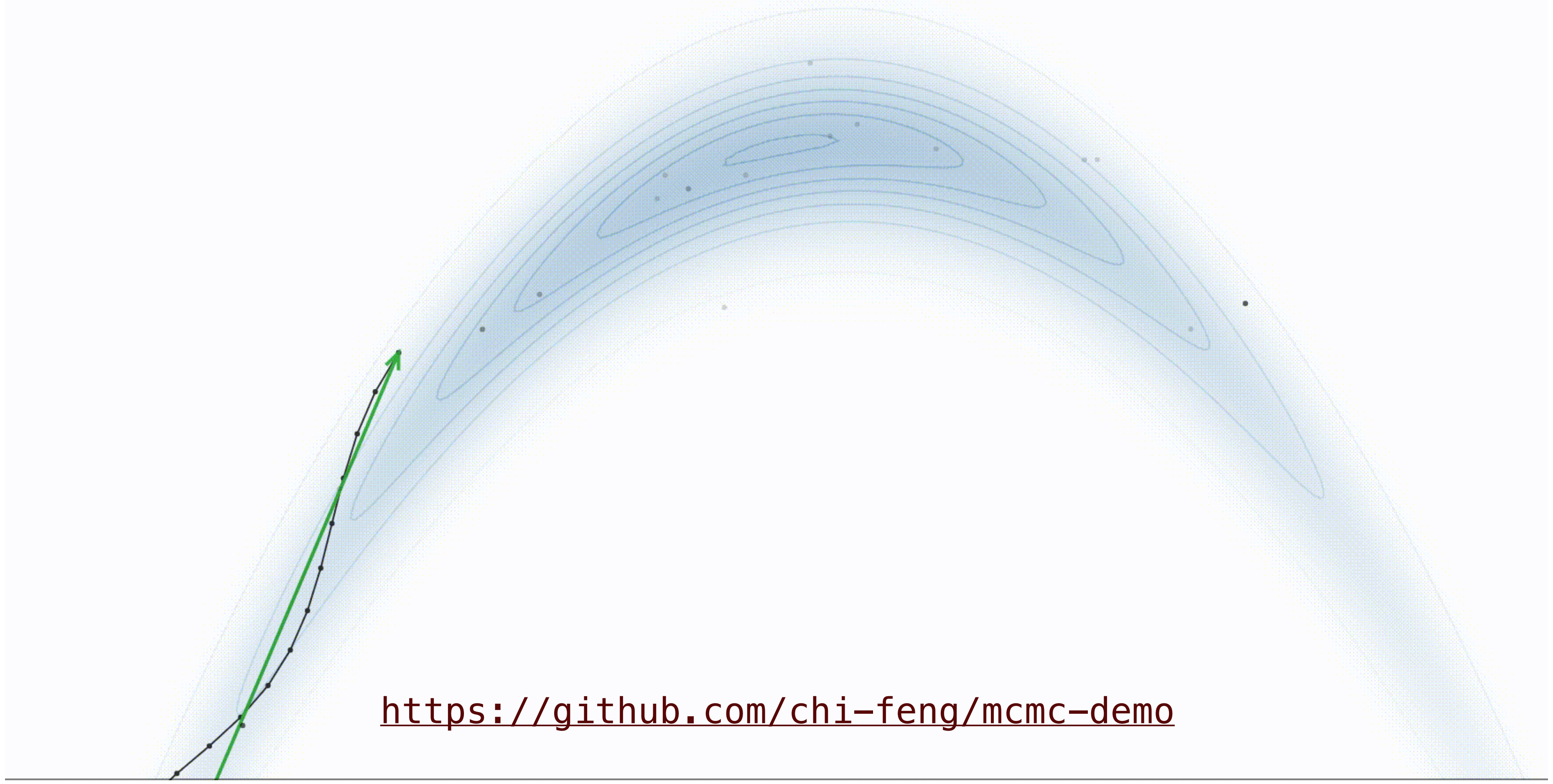
(a) Initially the Markov chain explores well-behaved regions of the typical set, avoiding the pathological neighborhood entirely and biasing Markov chain Monte Carlo estimators.

(b) If the Markov chain is run long enough then it will get stuck near the boundary of the pathological region, slowly correcting the Markov chain Monte Carlo estimators.

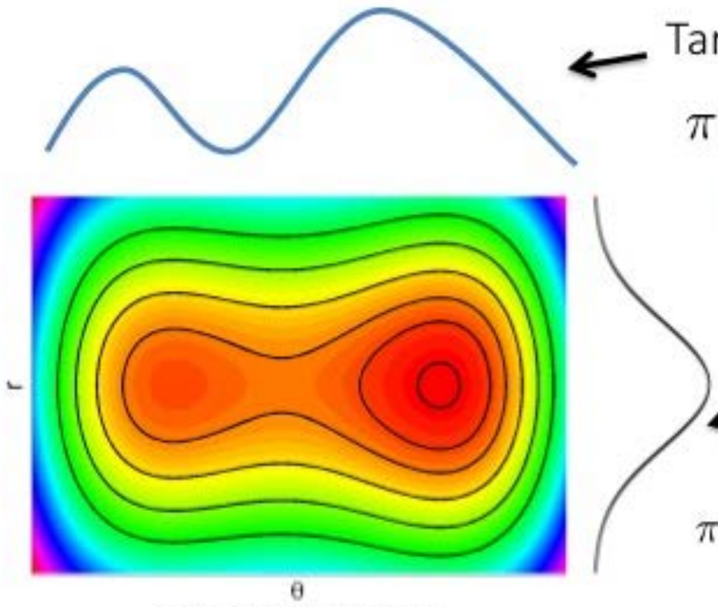
(c) Eventually the Markov chain escapes to explore the rest of the typical set. This process repeats, causing the resulting estimators to oscillate around the true expectations and inducing strong biases unless the chain is improbably stopped at exactly the right time.

Betancourt, M. (2017). A conceptual introduction to Hamiltonian Monte Carlo. arXiv preprint arXiv:1701.02434.

Hamiltonian Monte Carlo



https://github.com/chi-feng/mcmc-demo



Target posterior of θ

$$\pi(\theta) \triangleq p(\theta|\mathcal{D}) \propto \exp(-U(\theta))$$

Focus on $\theta \in \mathbb{R}^d$

Potential Energy
$U = -\log\left[p(\mathcal{D}|\theta)p(\theta)\right]$

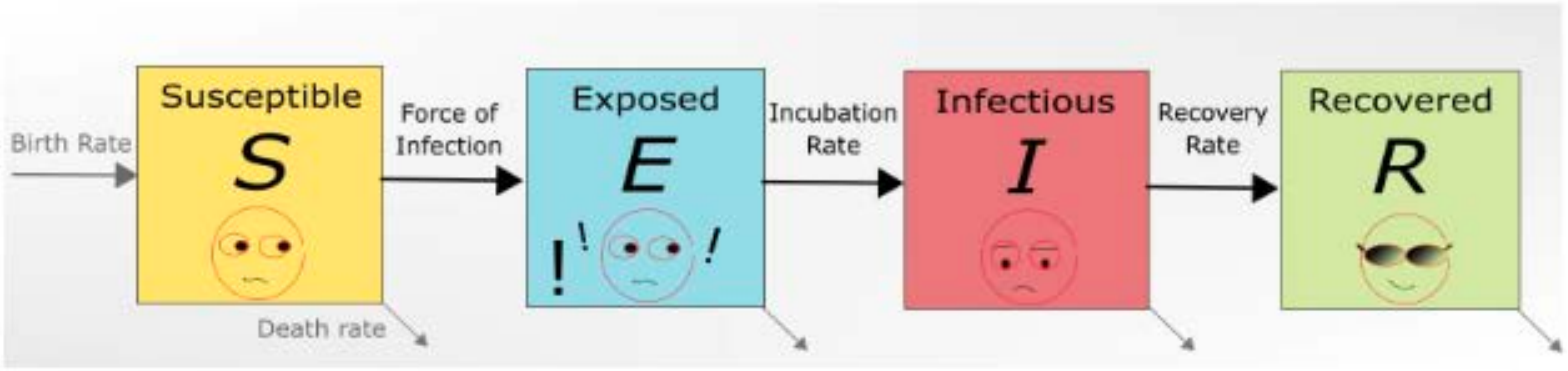
Add auxiliary "momentum" variable r

$$\pi(\theta, r) \propto \exp\left(-U(\theta) - \underbrace{\frac{1}{2}r^T M^{-1} r}_{\text{Kinetic Energy}}\right)$$

$$= \exp(-H(\theta, r))$$

Hamiltonian
(total energy H)

Example: Epidemiology models



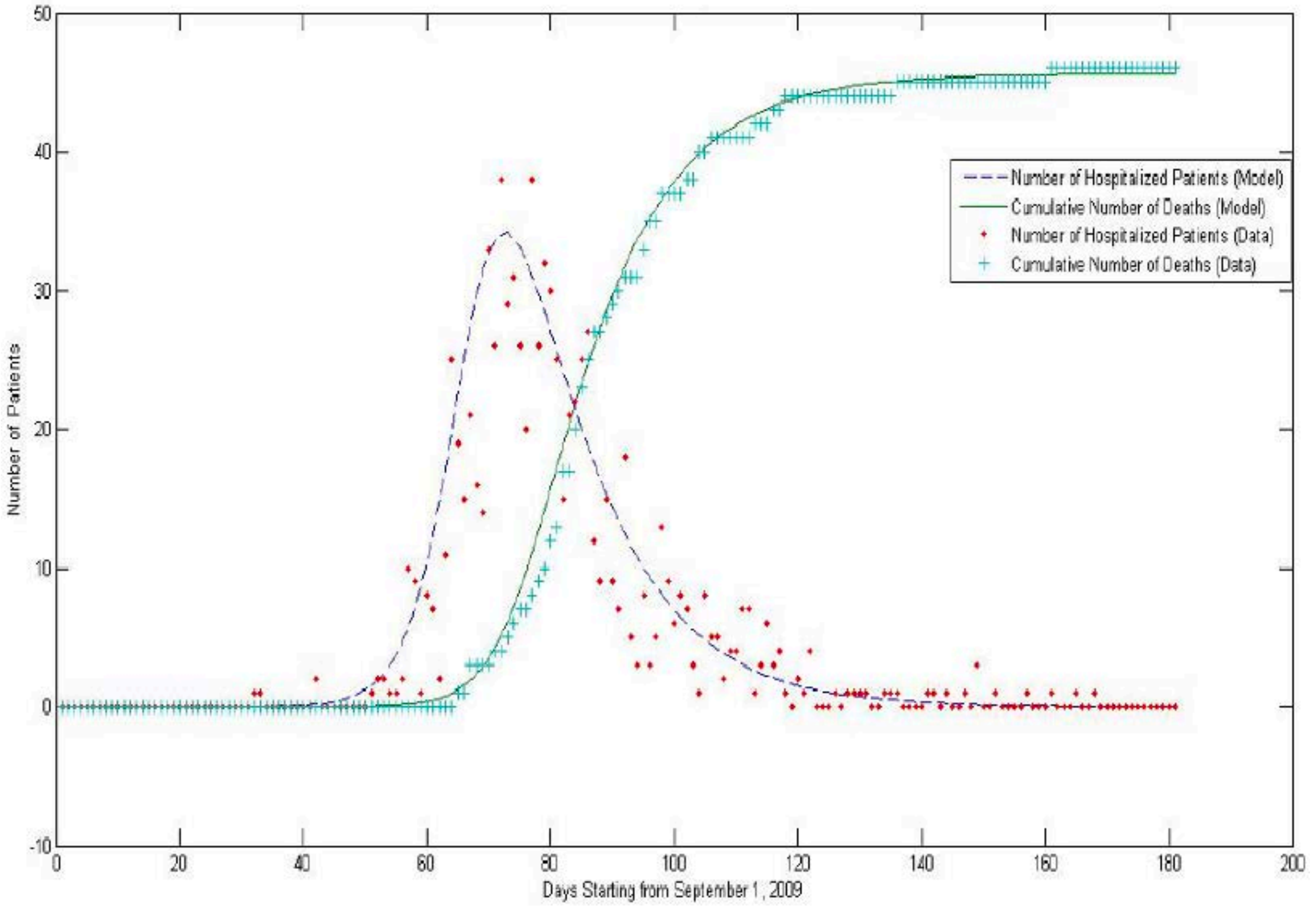
$$\frac{dS}{dt} = -\frac{\beta S I}{N}$$

$$\frac{dE}{dt} = \frac{\beta S I}{N} - \sigma E$$

$$\frac{dI}{dt} = \sigma E - \gamma I$$

$$\frac{dR}{dt} = \gamma I$$

$$N = S + E + I + R$$



Probabilistic programming

Stan

http://mc-stan.org/

PyMC3

https://github.com/pymc-devs/pymc3

Edward

http://edwardlib.org/

https://github.com/uber/pyro

Bayesian linear regression

$$y_i \sim \mathcal{N}(\beta_0 + \beta_1 x_i, 1/\tau)$$

or equivalently

$$y_i = \beta_0 + \beta_1 x_i + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1/\tau)$$

The likelihood for this model may be written as the product over N iid observations

$$L(y_1, \ldots, y_N, x_1, \ldots, x_N | \beta_0, \beta_1, \tau) = \prod_{i=1}^{N} \mathcal{N}(\beta_0 + \beta_1 x_i, 1/\tau)$$

Priors on the model parameters:

$$\beta_0 \sim \mathcal{N}(\mu_0, 1/\tau_0)$$

$$\beta_1 \sim \mathcal{N}(\mu_1, 1/\tau_1)$$

$$\tau \sim \mathrm{Gamma}(\alpha, \beta)$$

Bayesian linear regression

$$y_i \sim \mathcal{N}(\beta_0 + \beta_1 x_i, 1/\tau)$$

or equivalently

$$y_i = \beta_0 + \beta_1 x_i + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1/\tau)$$

The likelihood for this model may be written as the product over N iid observations

$$L(y_1, \ldots, y_N, x_1, \ldots, x_N | \beta_0, \beta_1, \tau) = \prod_{i=1}^{N} \mathcal{N}(\beta_0 + \beta_1 x_i, 1/\tau)$$

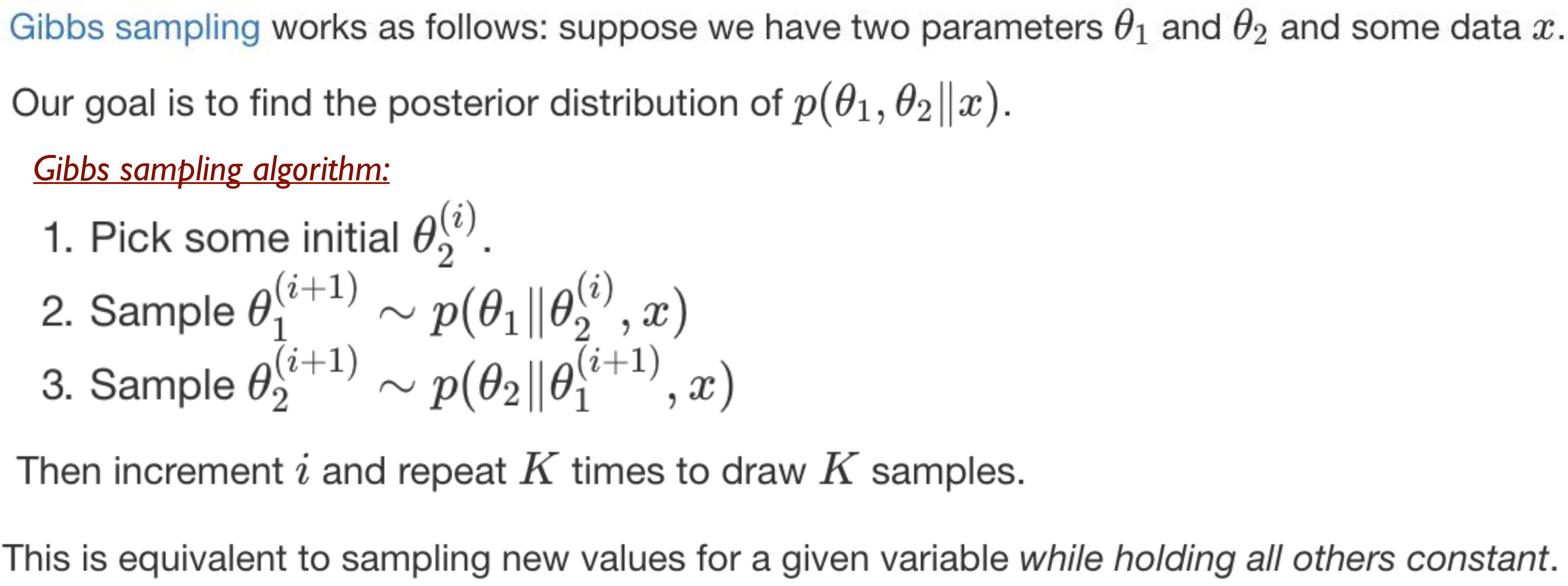
Priors on the model parameters:

$$\beta_0 \sim \mathcal{N}(\mu_0, 1/\tau_0)$$

$$\beta_1 \sim \mathcal{N}(\mu_1, 1/\tau_1)$$

$$\tau \sim \mathrm{Gamma}(\alpha, \beta)$$

Gibbs sampling



Gibbs sampling works as follows: suppose we have two parameters θ_1 and θ_2 and some data x.

Our goal is to find the posterior distribution of $p(\theta_1, \theta_2 || x)$.

Gibbs sampling algorithm:

1. Pick some initial $\theta_2^{(i)}$.
2. Sample $\theta_1^{(i+1)} \sim p(\theta_1 || \theta_2^{(i)}, x)$
3. Sample $\theta_2^{(i+1)} \sim p(\theta_2 || \theta_1^{(i+1)}, x)$

Then increment i and repeat K times to draw K samples.

This is equivalent to sampling new values for a given variable *while holding all others constant*.

The general approach to deriving an update for a variable is

1. Write down the posterior conditional density in log-form
2. Throw away all terms that don't depend on the current sampling variable
3. Pretend this is the density for your variable of interest and all other variables are fixed. What distribution does the log-density remind you of?
4. That's your conditional sampling density!

Pros: No parameters need to be tuned (e.g. vs MCMC that needs a proposal distribution)

Cons: It might be hard to analytically derive the conditional distributions.