PRECISION WORKSHOP

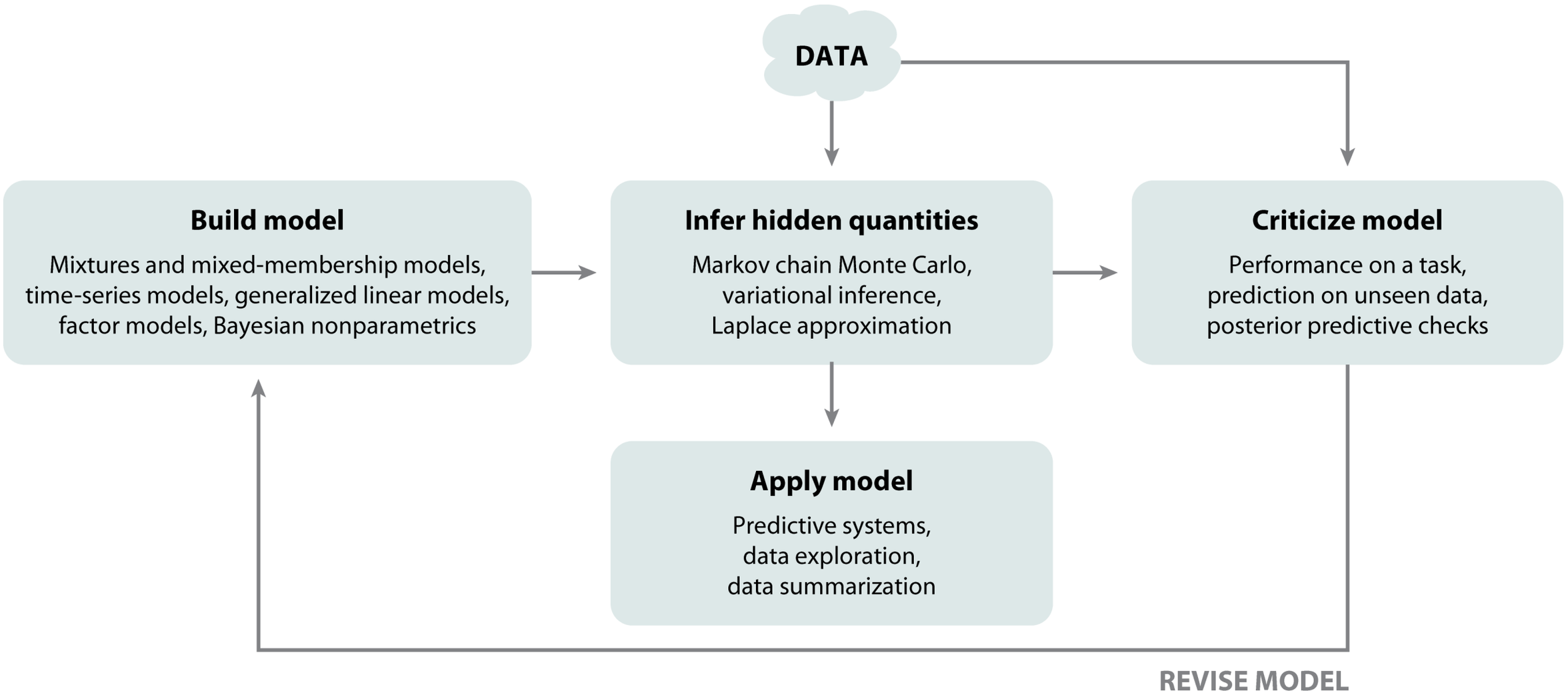
Practical Bayesian Computation with PyMC3

Junpeng Lao

May 2018 @ CEAi

Powered by 

A model-based Inference workflow

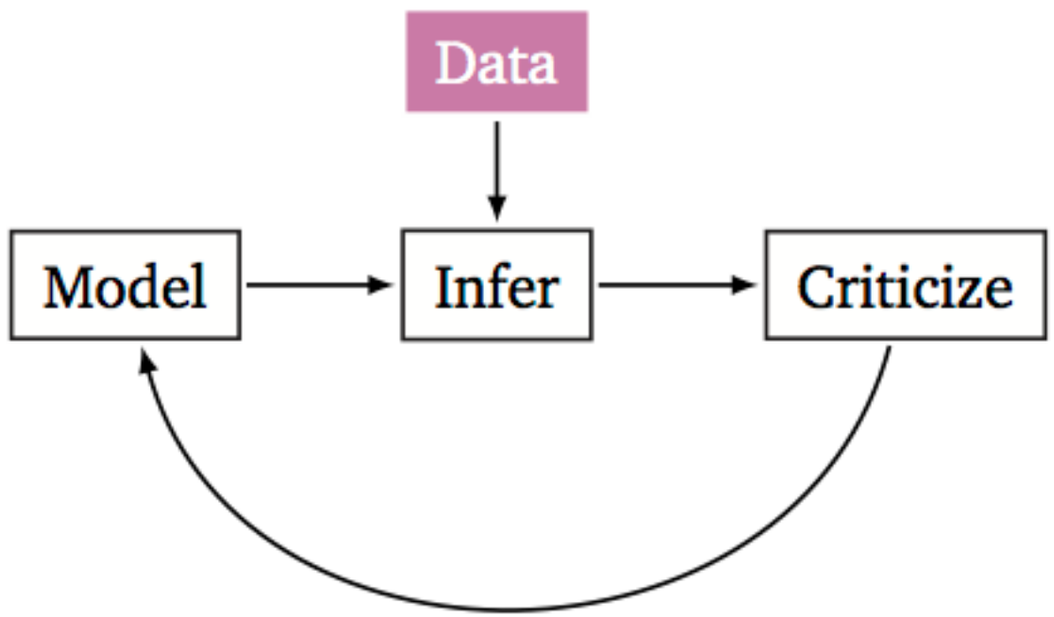


Blei DM. 2014.
Annu. Rev. Stat. Appl. 1:203–32

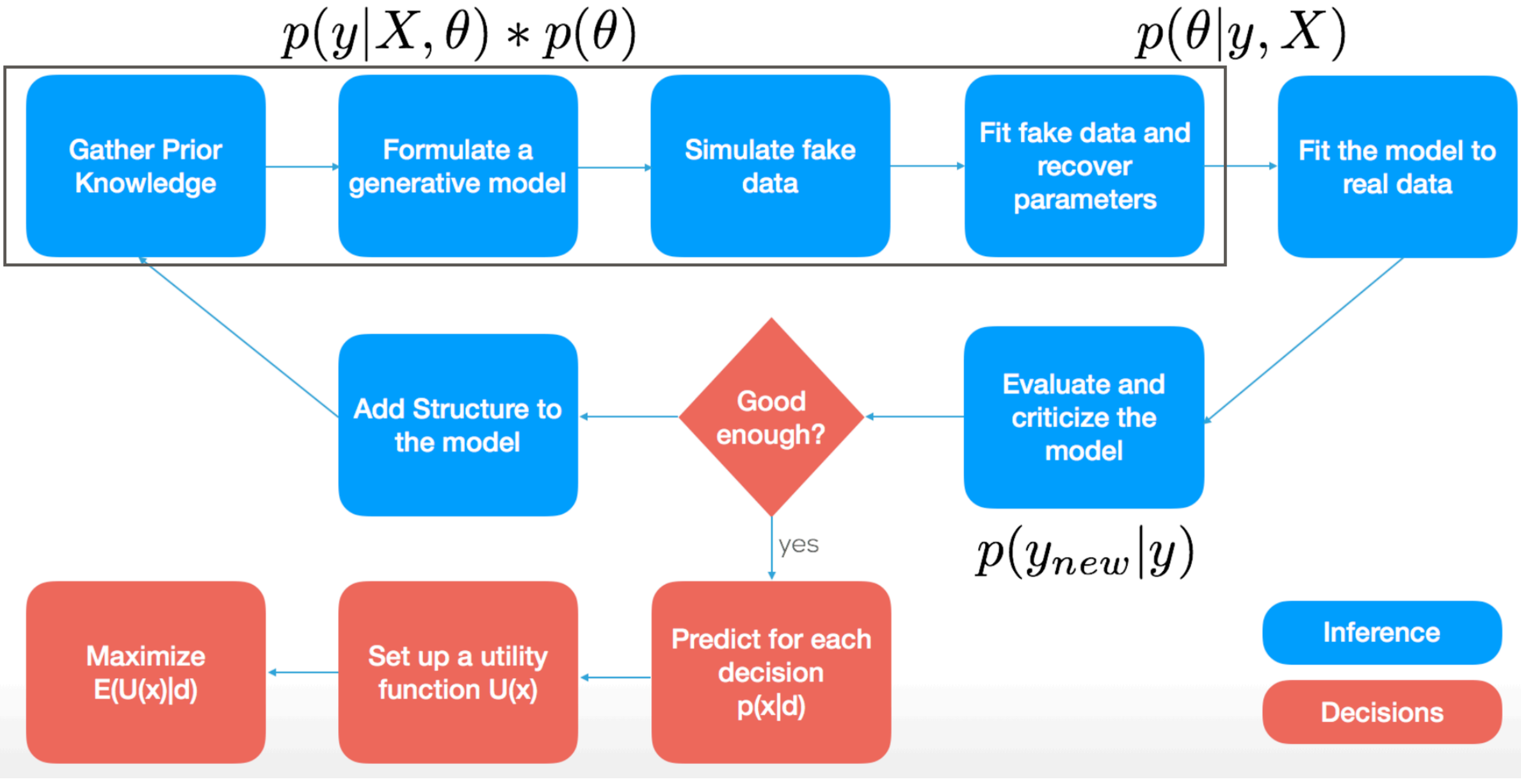
Box's Loop

First gather data from some real-world phenomena. Then cycle through Box's loop (Blei, 2014).

1. Build a probabilistic model of the phenomena.
2. Reason about the phenomena given model and data.
3. Criticize the model, revise and repeat.



Modern Bayesian workflow



http://rpubs.com/ericnovik/linreg

Building models

Statistical inference

Given a set of observed realizations of a random variable ***X***,

$$S = \boxed{?} x_1, x_2, \ldots, x_N$$

we want to infer on the underlying probability distribution that gives rise to the data S.

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

Why is it necessary to sample from the posterior distribution if we already KNOW the posterior distribution?

7

1

My understanding is that when using a Bayesian approach to estimate parameter values:

- The posterior distribution is the combination of the prior distribution and the likelihood distribution.
- We simulate this by generating a sample from the posterior distribution (e.g., using a Metropolis-Hasting algorithm to generate values, and accept them if they are above a certain threshold of probability to belong to the posterior distribution).
- Once we have generated this sample, we use it to approximate the posterior distribution, and things like its mean.

But, I feel like I must be misunderstanding something. It sounds like we have a posterior distribution and then sample from it, and then use that sample as an approximation of the posterior distribution. But if we have the posterior distribution to begin with why do we need to sample from it to approximate it?

asked 7 months ago

viewed 627 times

active 6 months ago

"The posterior is easy to construct (or at least the joint distribution over data and parameters) — it's *using the posterior* that's hard."

What do we want from the posterior?

for instance the completely artificial target

$$\pi(\theta|x) \propto \exp\{-||\theta - x||^2 - ||\theta + x||^4 - ||\theta - 2x||^6\}, \quad x, \theta \in \mathbb{R}^{18},$$

does not tell me what is

1. the posterior expectation of a function of θ, e.g., $\mathbb{E}[\mathfrak{h}(\theta)|x]$, posterior mean that operates as a Bayesian estimator under standard losses;
2. the optimal decision under an arbitrary utility function, decision that minimises the expected posterior loss;
3. a 90% or 95% range of uncertainty on the parameter(s), a sub-vector of the parameter(s), or a function of the parameter(s), aka HPD region

$$\{h = \mathfrak{h}(\theta);\ \pi^{\mathfrak{h}}(h) \geq \underline{h}\}$$

4. the most likely model to choose between setting some components of the parameter(s) to specific values versus keeping them unknown (and random).

These are only examples of many usages of the posterior distribution. In all cases but the most simple ones, I cannot provide the answers by staring at the posterior distribution density and do need to proceed through numerical resolutions like Monte Carlo and Markov chain Monte Carlo methods.

share cite edit flag

edited Nov 6 '17 at 22:37

amoeba

50.2k 11 172 232

answered Oct 14 '17 at 14:26

Xi'an

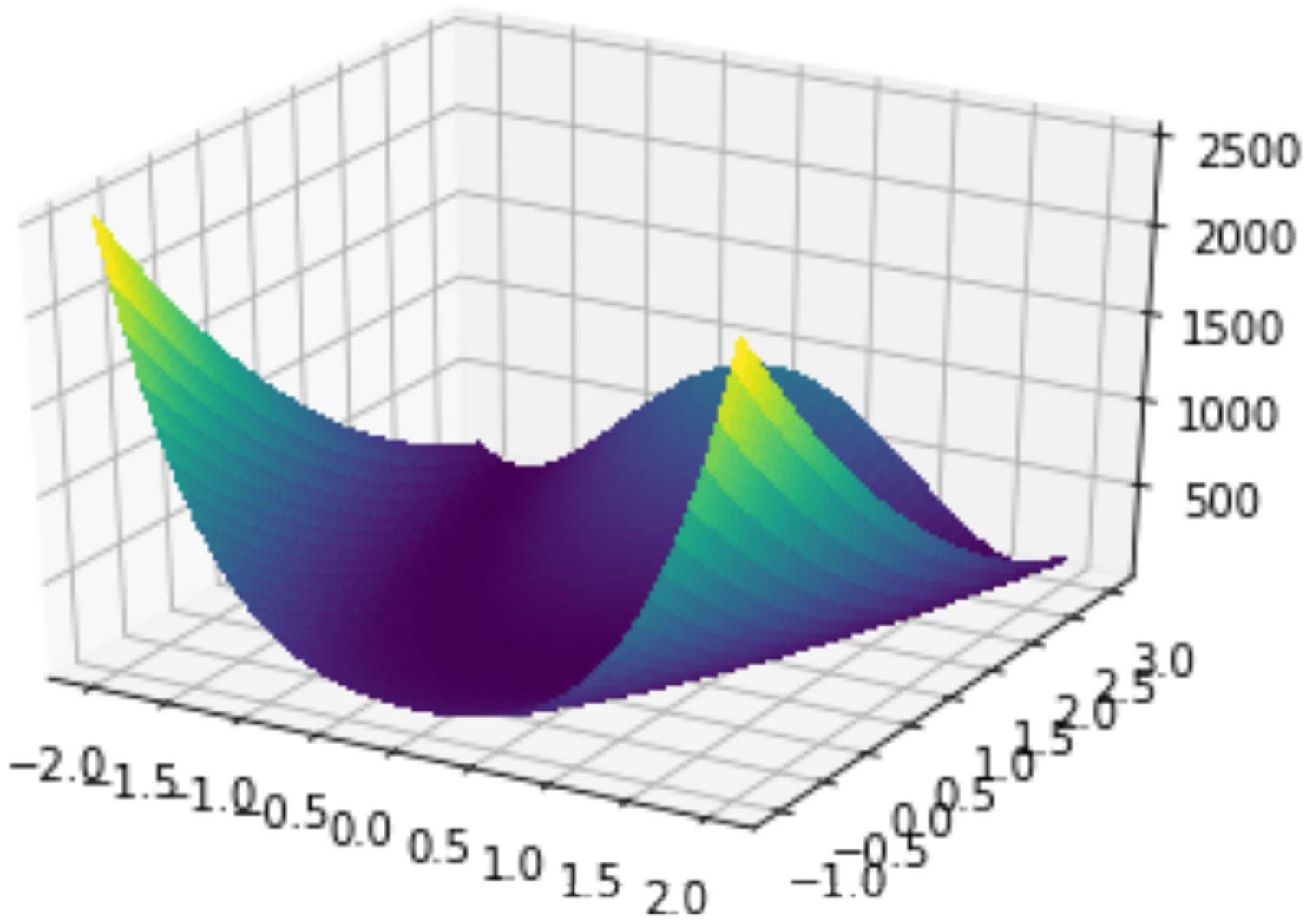
46.4k 6 80 303

Inference

We are operate in the parameter space that fully specified by the model:

- Point estimation: a specific location in this space
  - Estimators
    - Maximum likelihood Estimation
    - Iterative solvers
- Monte Carlo sampling
  - Different kind of samplers
- Approximation
  - Laplace approximation
  - Variational inference
  - Expectation propagation

Inference



Code8 - HMC_Leapfrog.ipynb

Monte Carlo Sampling

Bayesian statistics often requires the numerical evaluation of the expectation of any given function of interest with respect to a given density $\pi(\theta)$ defined as:

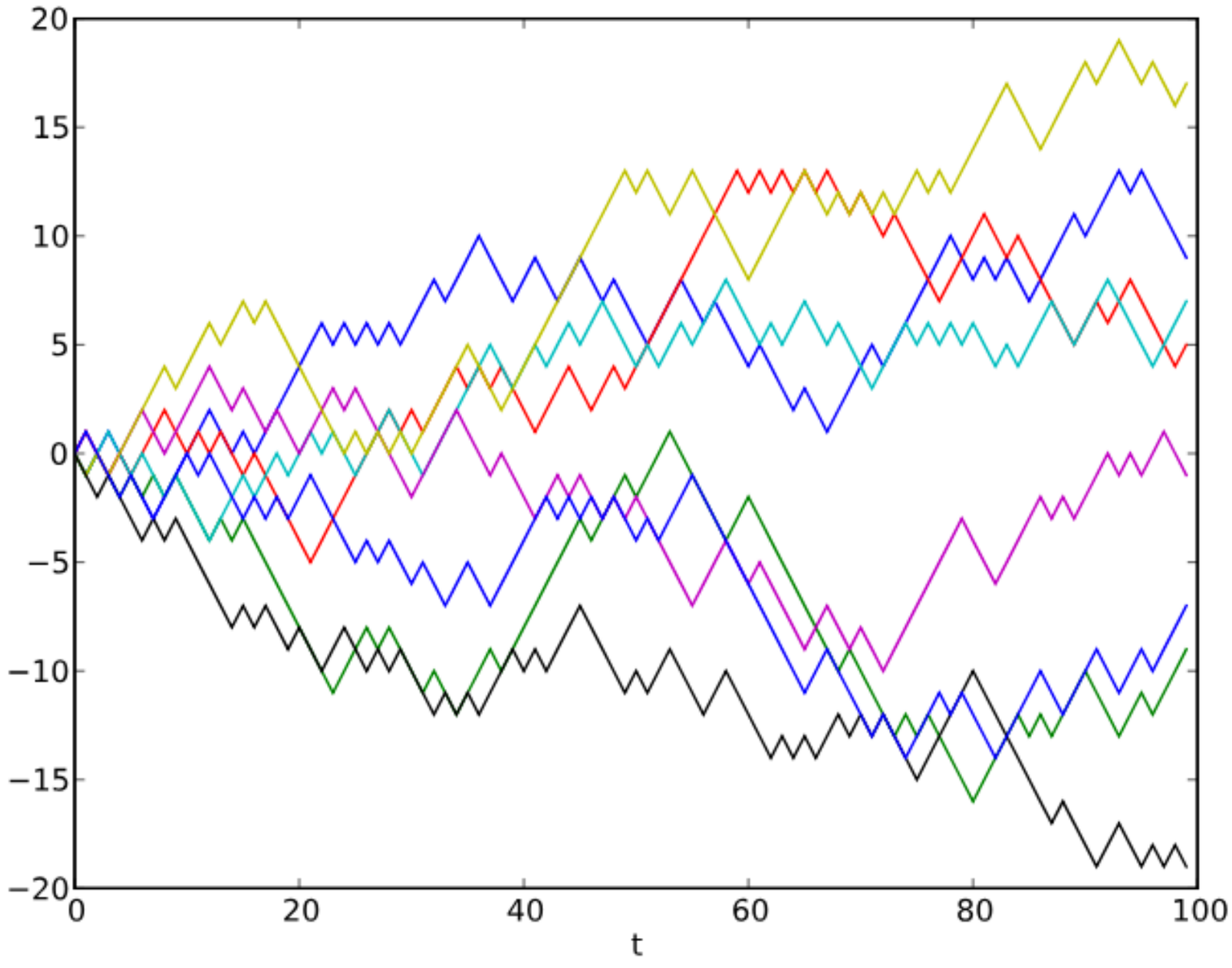
$$\mathsf{E}_\pi[f] \triangleq \int f(\theta)\pi(\theta)\,\mathrm{d}\theta, \quad \theta \in \mathbb{R}^{n_\theta},$$

- A high dimensional integration problem

$$\widehat{f}_M \triangleq \frac{1}{M}\sum_{k=1}^{M} f(\theta_k) \xrightarrow{a.s.} \mathsf{E}_\pi[f].$$

Markov Chain

$$Pr(X_{t+1} = x_{t+1} | X_t = x_t, X_{t-1} = x_{t-1}, \ldots, X_0 = x_0) = Pr(X_{t+1} = x_{t+1} | X_t = x_t)$$



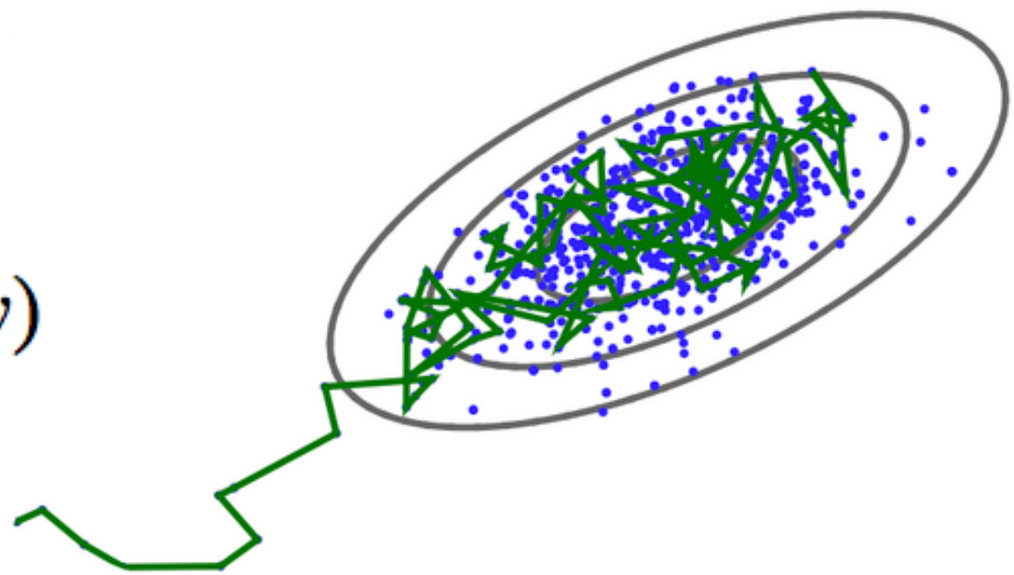
Markov Chain Monte Carlo

An invariant distribution with respect to some Markov chain with transition kernel Pr(y|x) implies that:

$$\int_x Pr(y \mid x)\pi(x)dx = \pi(y).$$

Detailed balance:

$$\pi(x)Pr(y \mid x) = \pi(y)Pr(x \mid y)$$



Generalised Metropolis–Hasting Algorithm

Algorithm 1. Generalised Metropolis–Hasting Algorithm

1. Draw a proposal $\mathcal{V}_k \sim q(\mathcal{V}_k \mid \theta_k)$ and compute the generalised acceptance probability as

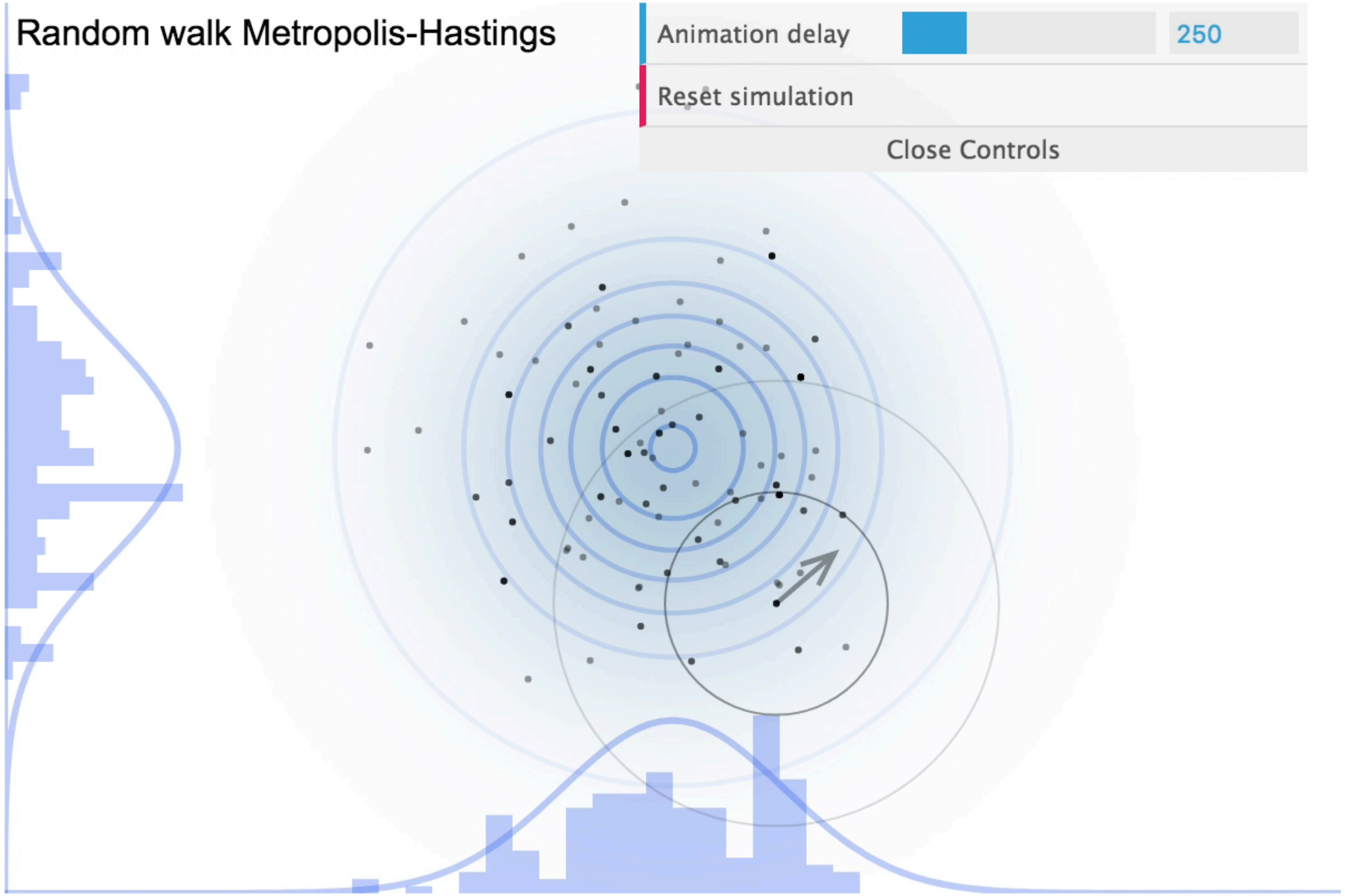
$$\alpha(\theta_k, \mathcal{V}_k) \triangleq \min\left\{1, \frac{\pi(\xi_k, \mathcal{W}_k)}{\pi(\theta_k, \mathcal{V}_k)} \left|J_{\mathcal{T}}(\theta_k, \mathcal{V}_k)\right|\right\}; \quad (\xi_k, \mathcal{W}_k) \triangleq \mathcal{T}(\theta_k, \mathcal{V}_k), \tag{13}$$

   where $\left|J_{\mathcal{T}}(\theta_k, \mathcal{V}_k)\right|$ is the determinant of the Jacobian of $\mathcal{T}$ evaluated at $\left[\theta_k, \mathcal{V}_k\right]$.

2. For some $u_k \sim \mathcal{U}[0, 1]$, we set $\theta_{k+1} = \begin{cases} \xi_k & \text{if } u_k \leq \alpha(\theta_k, \mathcal{V}_k), \\ \theta_k & \text{otherwise.} \end{cases}$

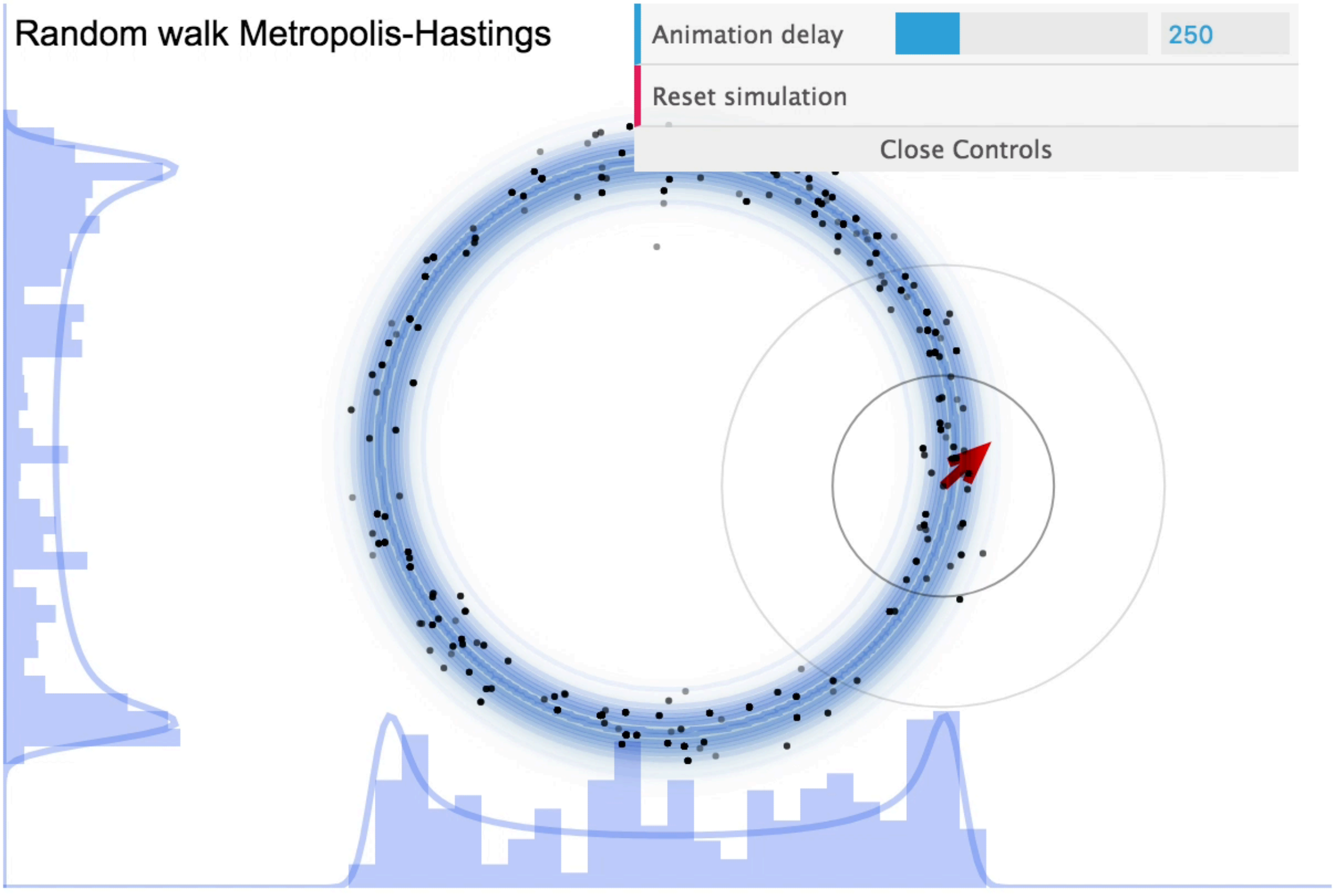
https://arxiv.org/abs/1607.01985

MCMC sampling



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

MCMC sampling



HMC and NUTS

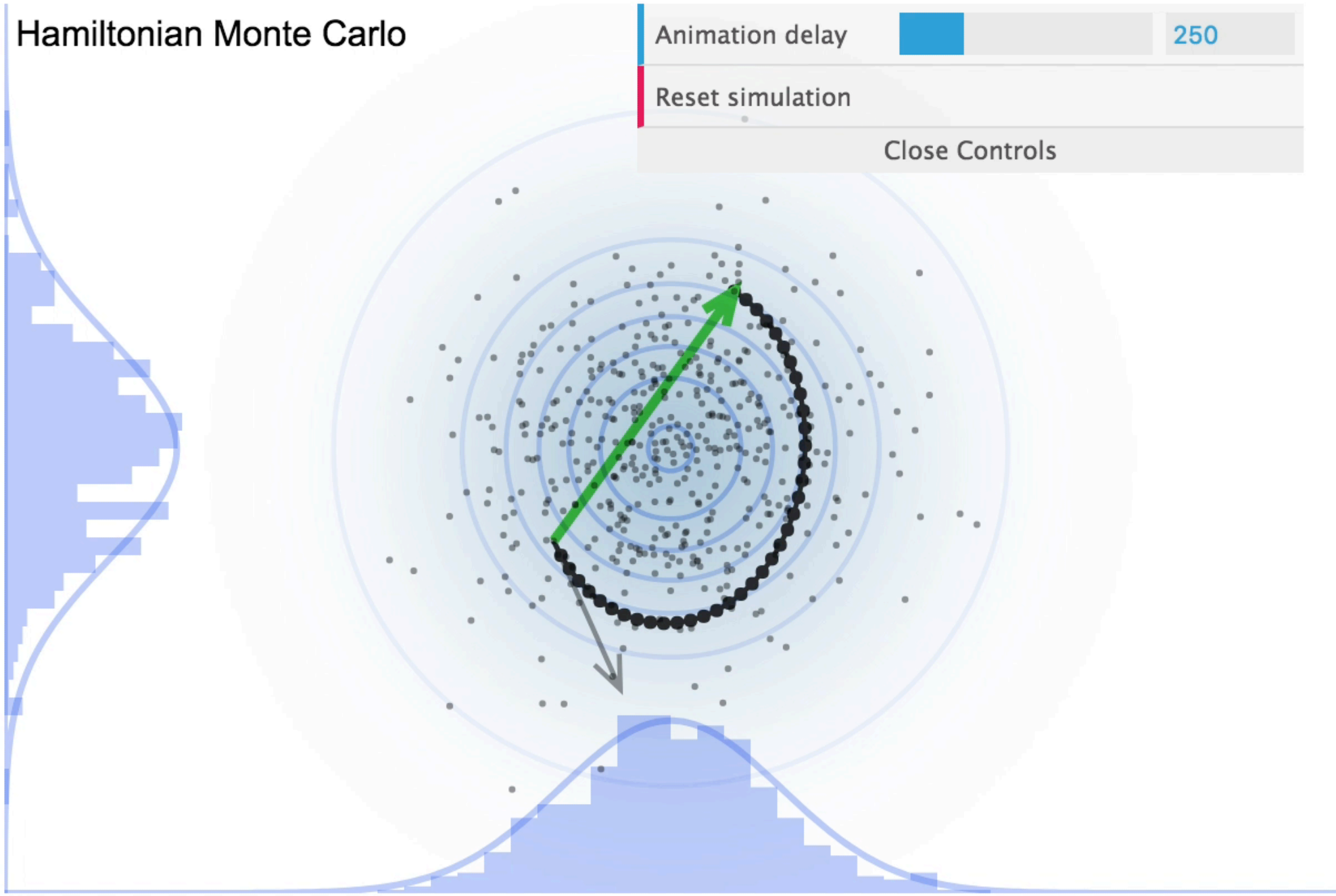
A Conceptual Introduction to Hamiltonian Monte Carlo

Michael Betancourt

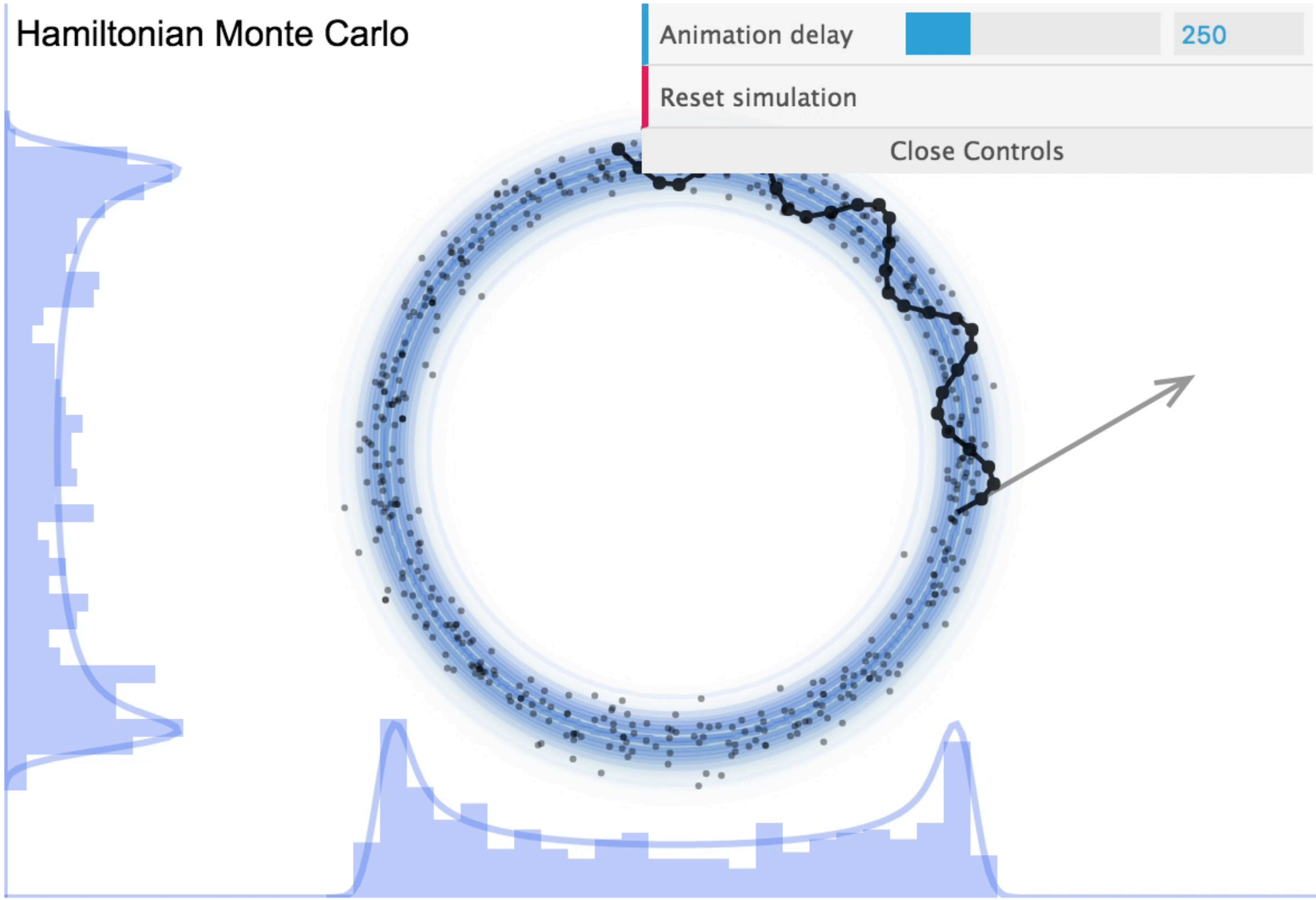
Abstract. Hamiltonian Monte Carlo has proven a remarkable empirical success, but only recently have we begun to develop a rigorous understanding of why it performs so well on difficult problems and how it is best applied in practice. Unfortunately, that understanding is con-

https://arxiv.org/abs/1701.02434

HMC

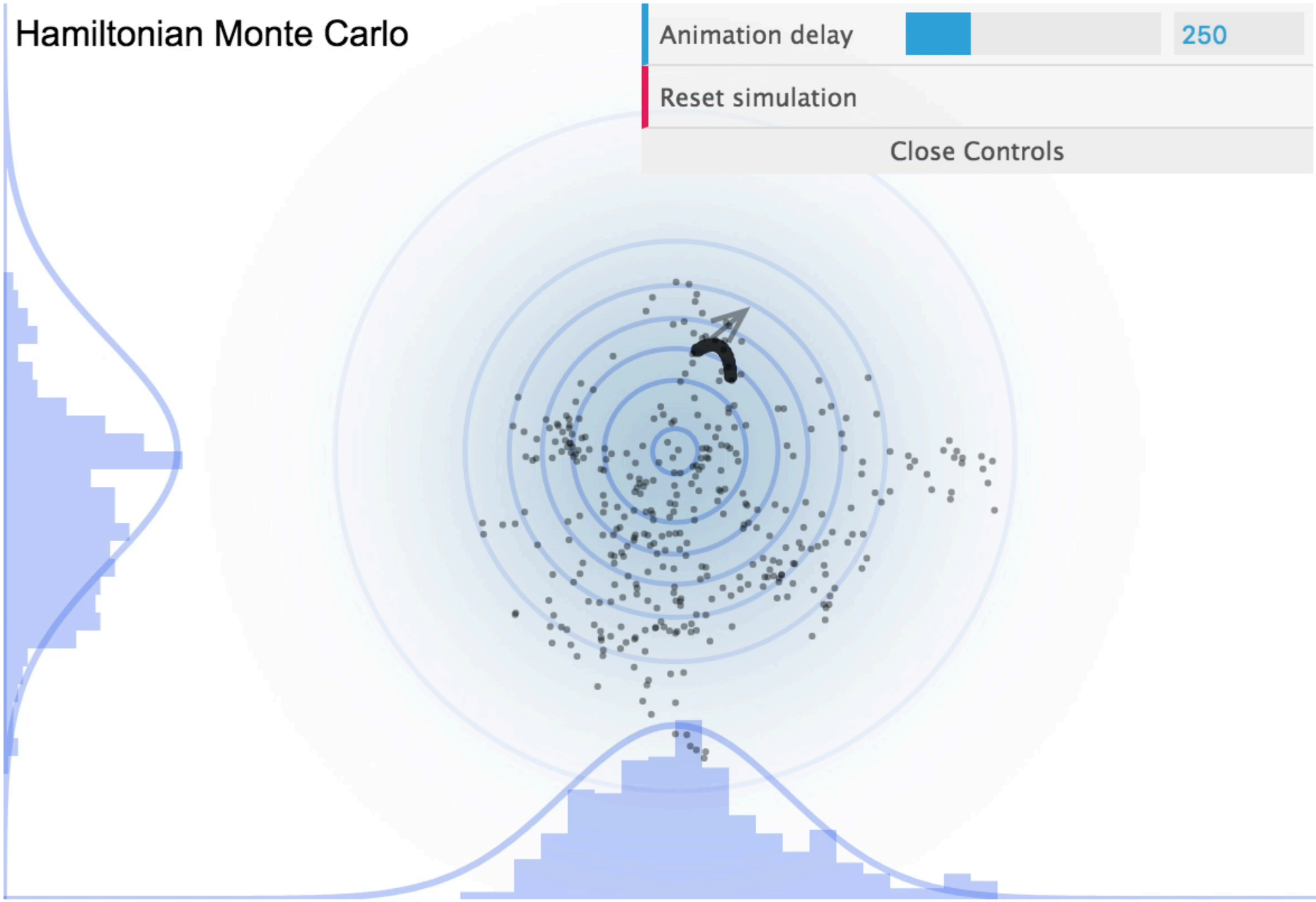

Hamiltonian Monte Carlo
Animation delay
250
Reset simulation
Close Controls

HMC

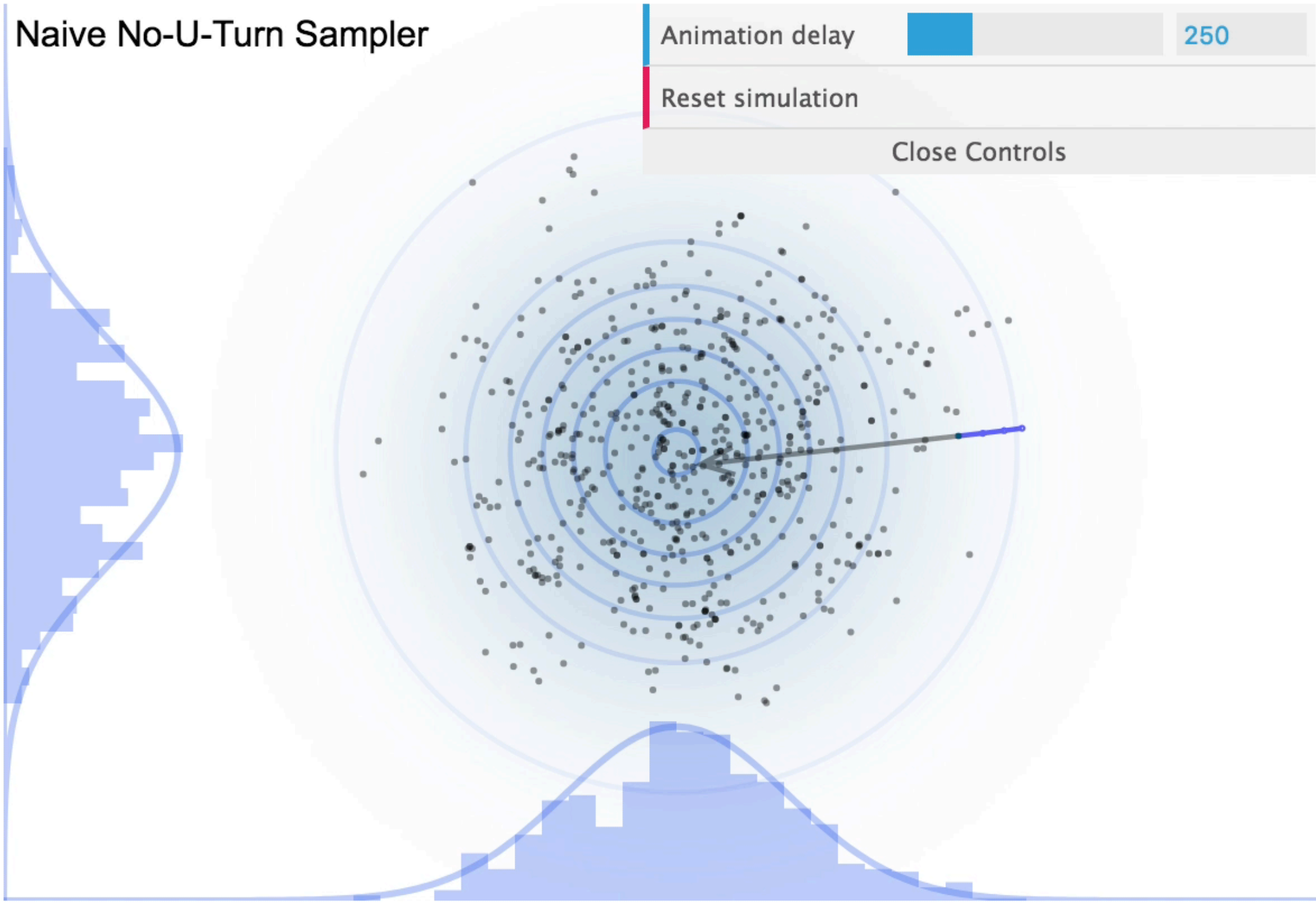


Code8 - HMC_Leapfrog.ipynb

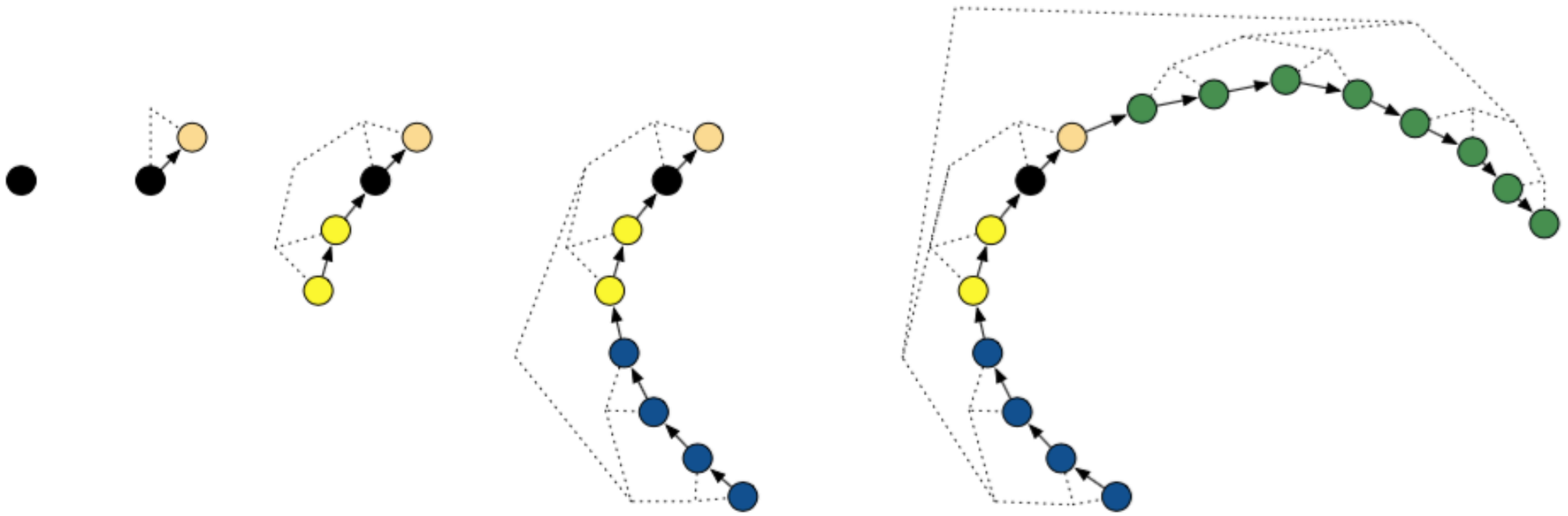
HMC

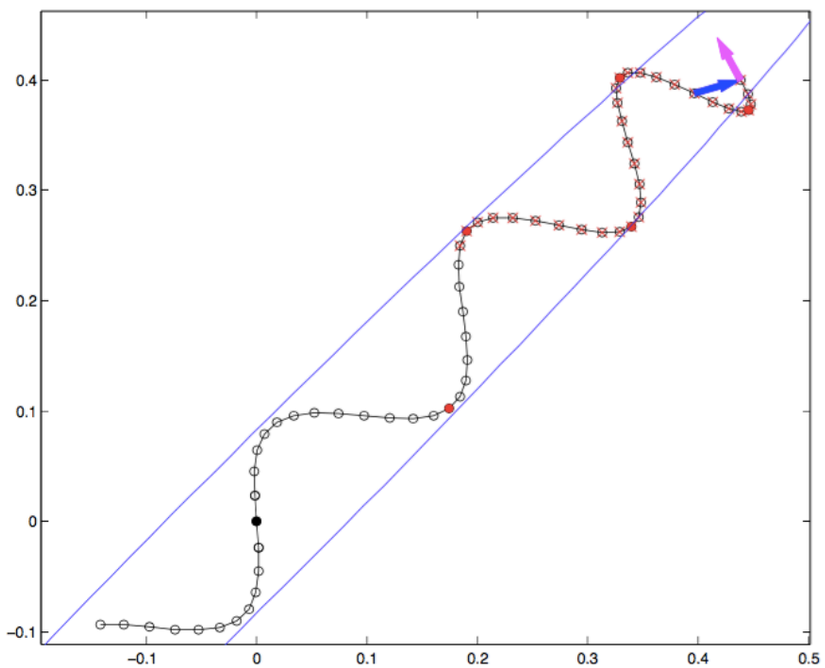

Hamiltonian Monte Carlo
Animation delay
250
Reset simulation
Close Controls

NUTS

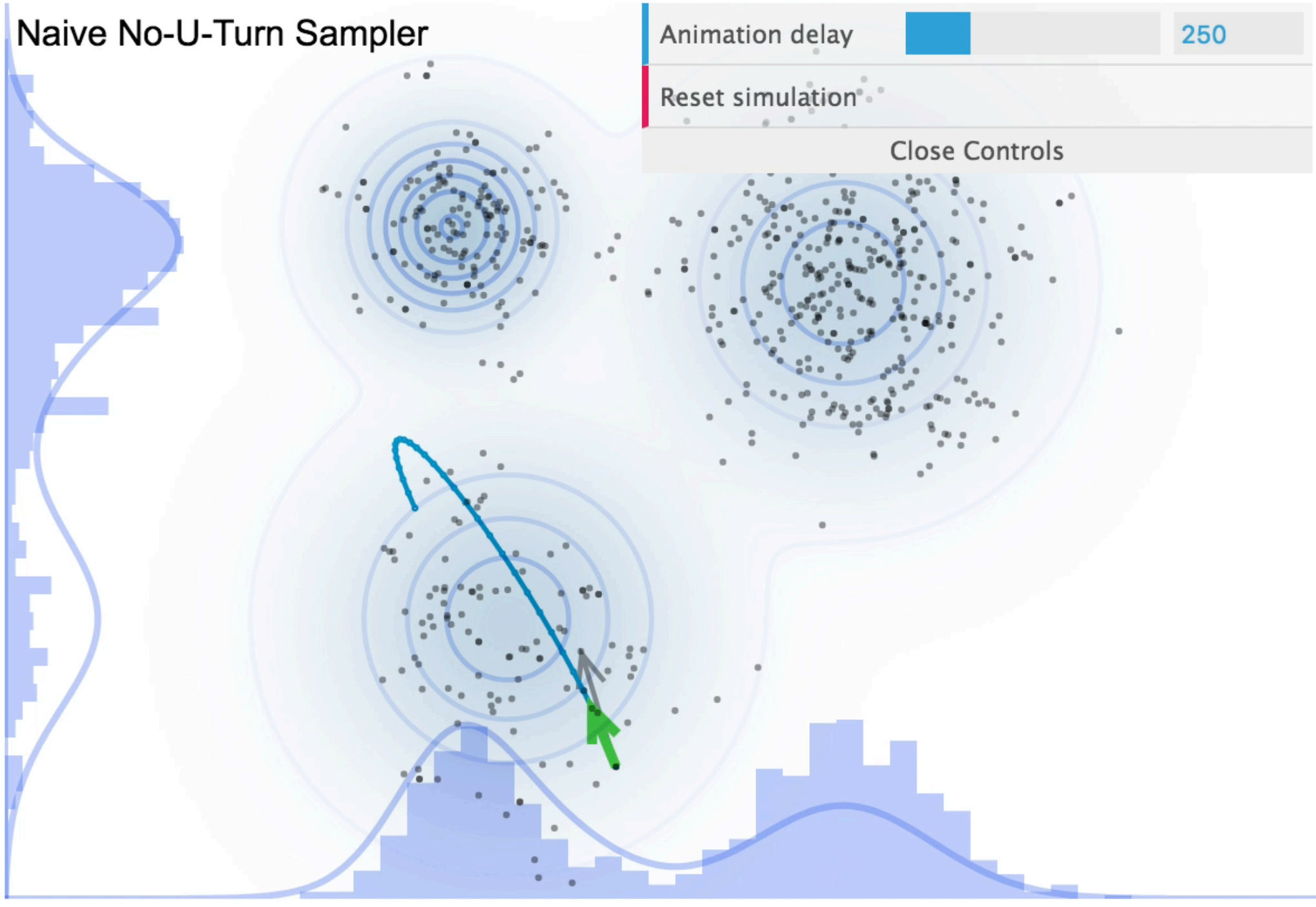

Naive No-U-Turn Sampler
Animation delay
250
Reset simulation
Close Controls

NUTS tree building





NUTS


Naive No-U-Turn Sampler
Animation delay
250
Reset simulation
Close Controls

HMC and NUTS related diagnostic

```
There were 31 divergences after tuning. Increase `target_accept` or reparameterize.
The acceptance probability does not match the target. It is 0.5134372680489251, but should be close to 0.8. Try to in
crease the number of tuning steps.
There were 91 divergences after tuning. Increase `target_accept` or reparameterize.
The acceptance probability does not match the target. It is 0.14324708680515164, but should be close to 0.8. Try to i
ncrease the number of tuning steps.
```

Dealing with divergence(s)

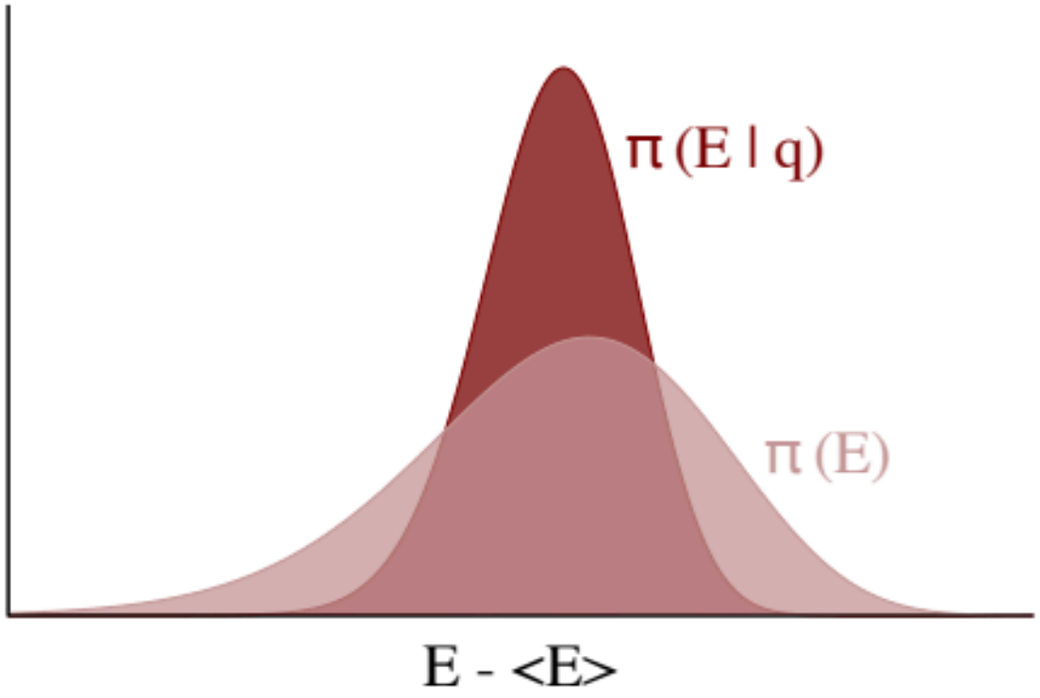
- Increase `target_accept` by doing
  - nuts_args=dict(target_accept=.95)
- Reparameterize
- Avoid heavy tail prior (especially with small n of observation)
  - Cauchy+ for sd
- Careful with bounded variable with large volume around the edge
  - Dirichlet with a[…] << 1.

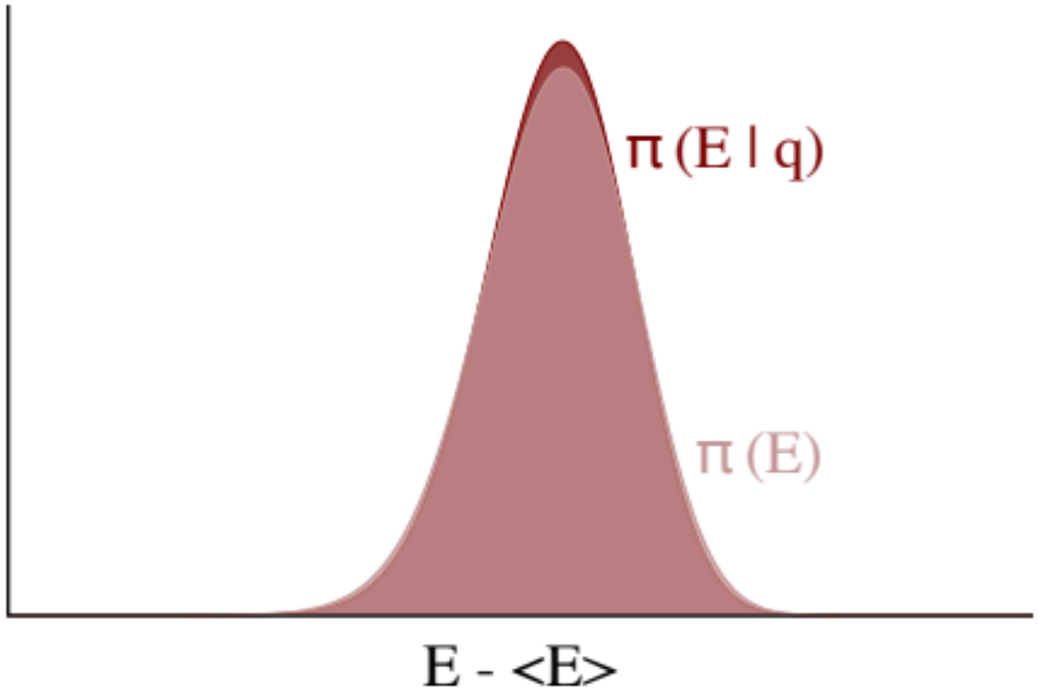
Code8 - HMC_Leapfrog.ipynb

HMC and NUTS related diagnostic

Energy plot and Bayesian Fraction of Missing Information

```
bfmi = pm.bfmi(trace)
```



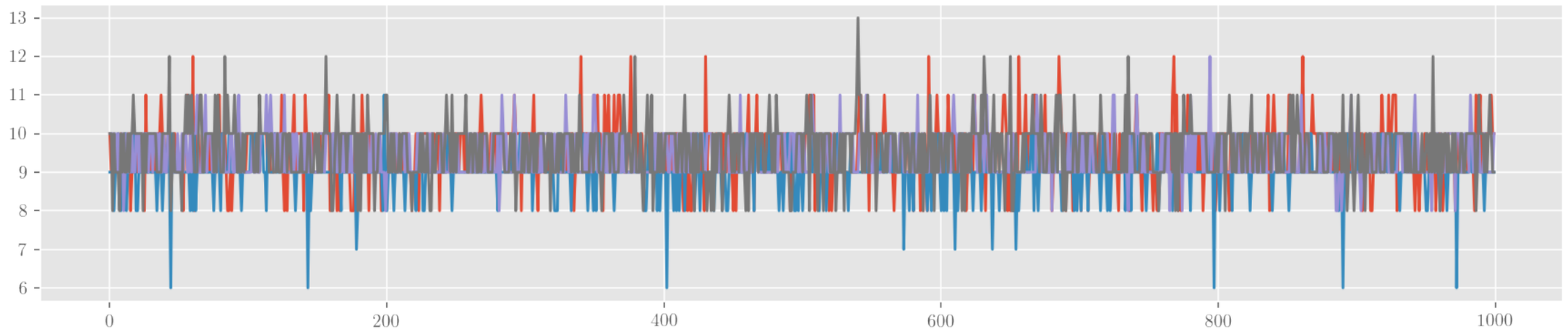


https://arxiv.org/abs/1701.02434

NUTS related diagnostic

Tree depth

```
nuts_kwargs=dict(max_treedepth=15)
```

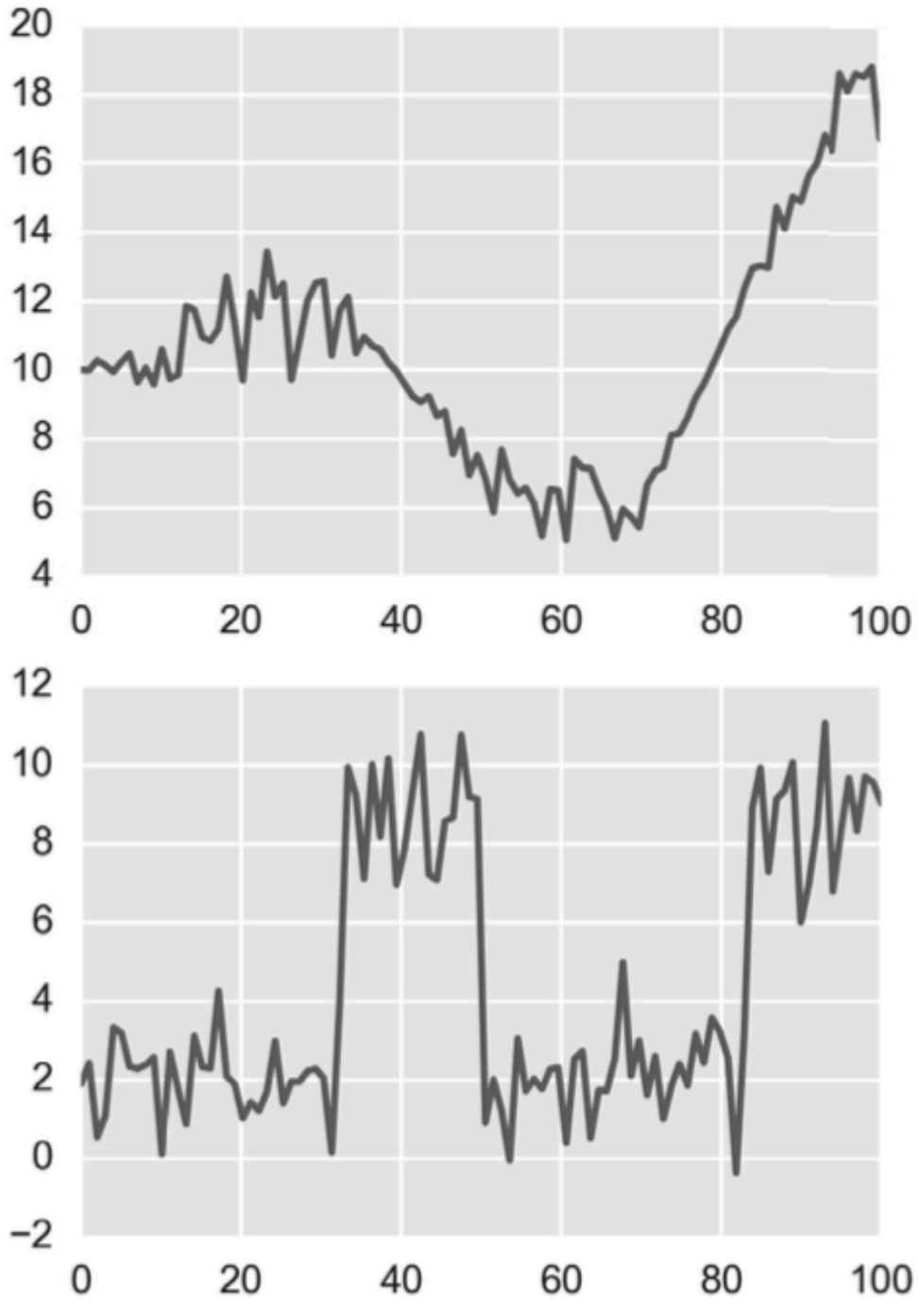


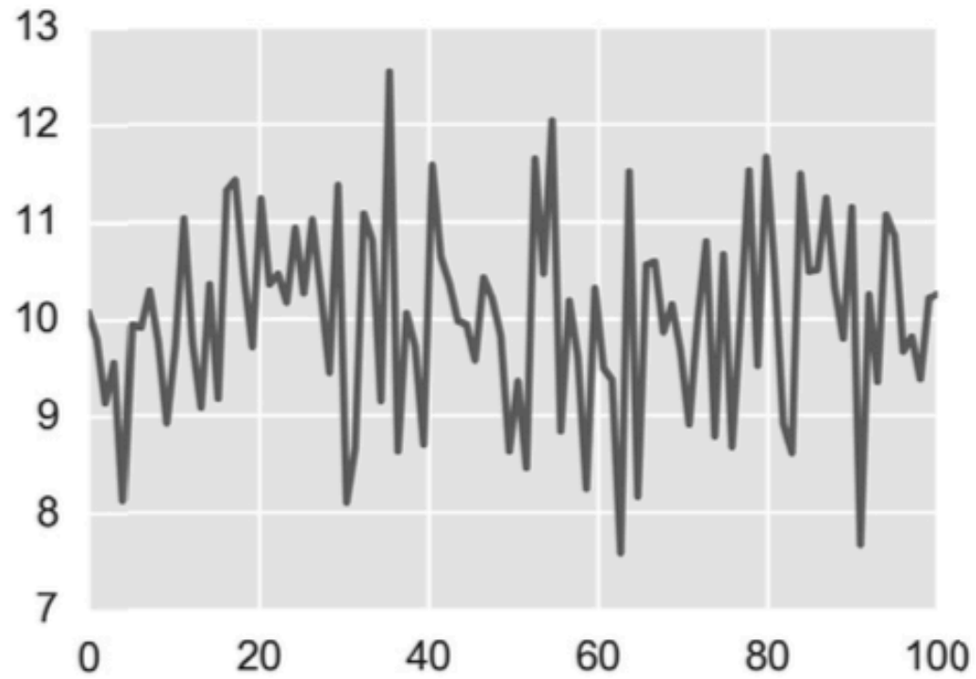
Challenges of MCMC sampling

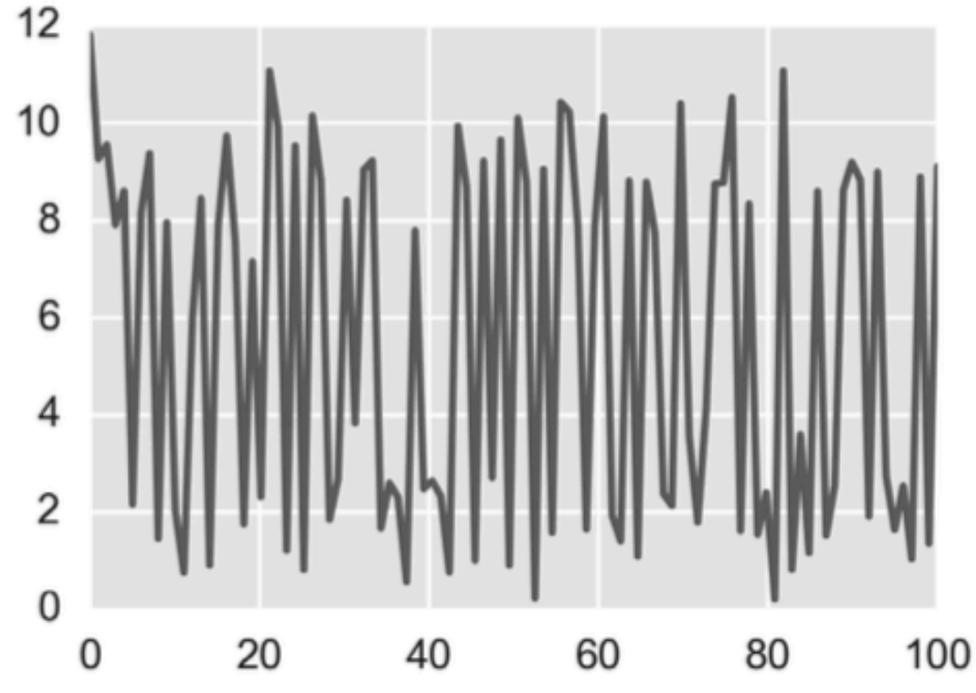
- Has a randomly initialized Markov chain converged to its equilibrium distribution?

- The draws from a Markov chain are correlated —> the central limit theorem’s bound on estimation error no longer applies.

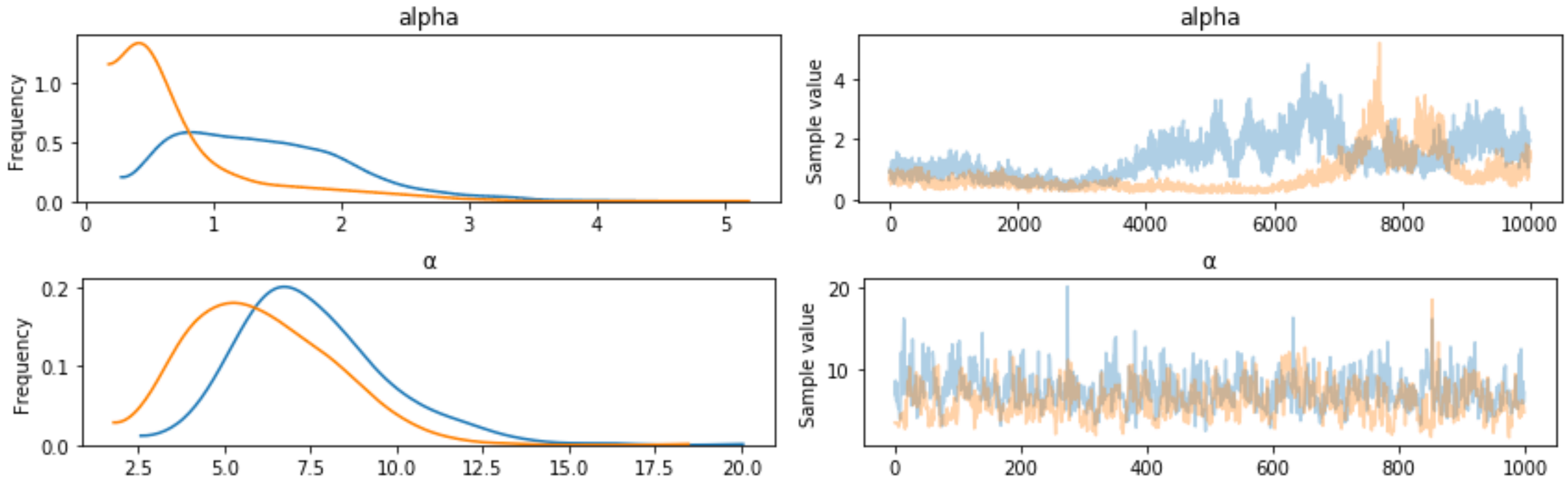
What is a *good* MCMC chain





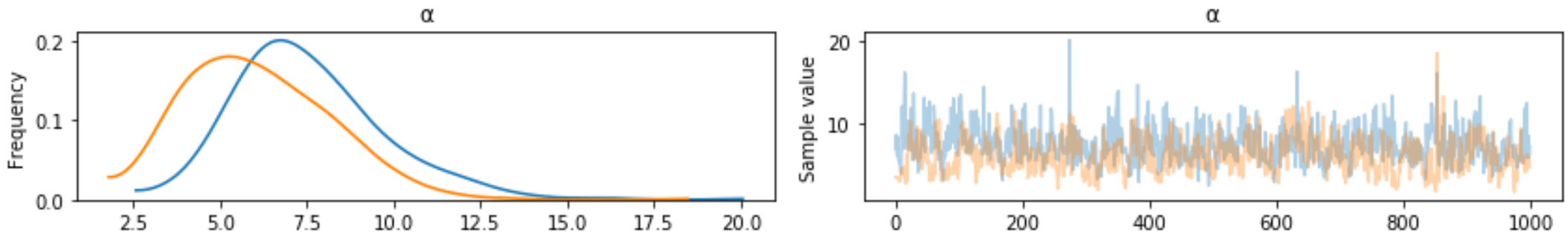


Potential scale reduction



- Better known as R-hat.
- The R-hat statistic measures the ratio of the average variance of samples within each chain to the variance of the pooled samples across chains
  - R-hat = 1 —> Good
  - R-hat > 1 —> Bad

Rhat in PyMC3




```
NUTS: [μ_log__, p_logodds__, α_log__]
100%|██████████| 3000/3000 [17:37<00:00,  3.18it/s]
The gelman-rubin statistic is larger than 1.4 for some parameters. The sampler did not converge.
```

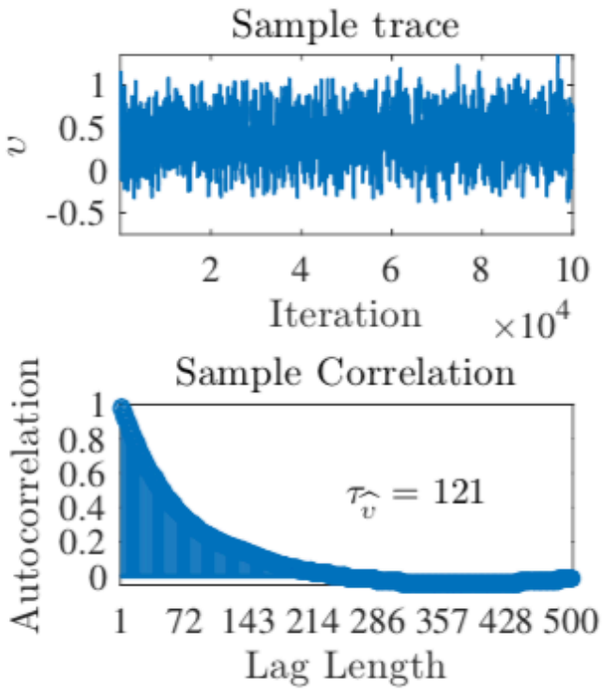
Convergence is Global

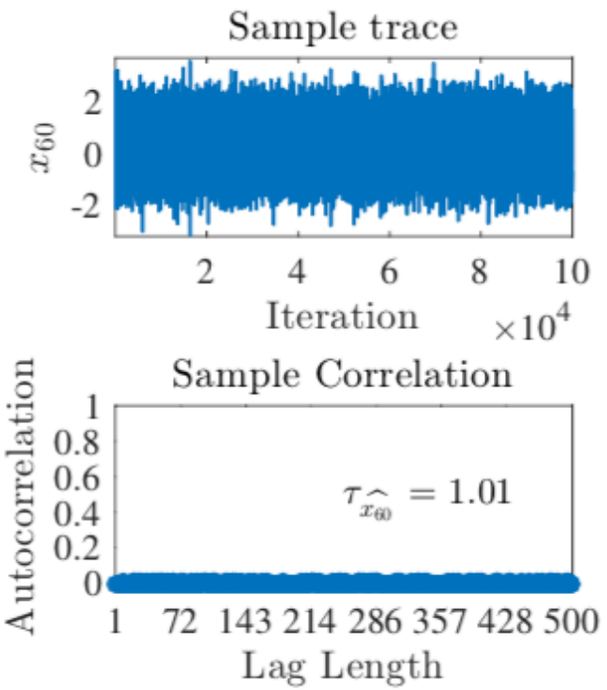
Is it acceptable to monitor convergence of only a subset of the parameters or generated quantities?

“If Rhat is close enough to one for all of your variables except for logp then the expectation value estimates for those variables are probably okay, especially if none of the other diagnostics are indicating problems.”

- use Rhat of logp samples as upper bound.

Effective Sample Size





The effective sample size of N samples generated by a process with autocorrelations ρ_t

$$N_{\text{eff}} = \frac{N}{\sum_{t=-\infty}^{\infty} \rho_t} = \frac{N}{1 + 2\sum_{t=1}^{\infty} \rho_t}.$$

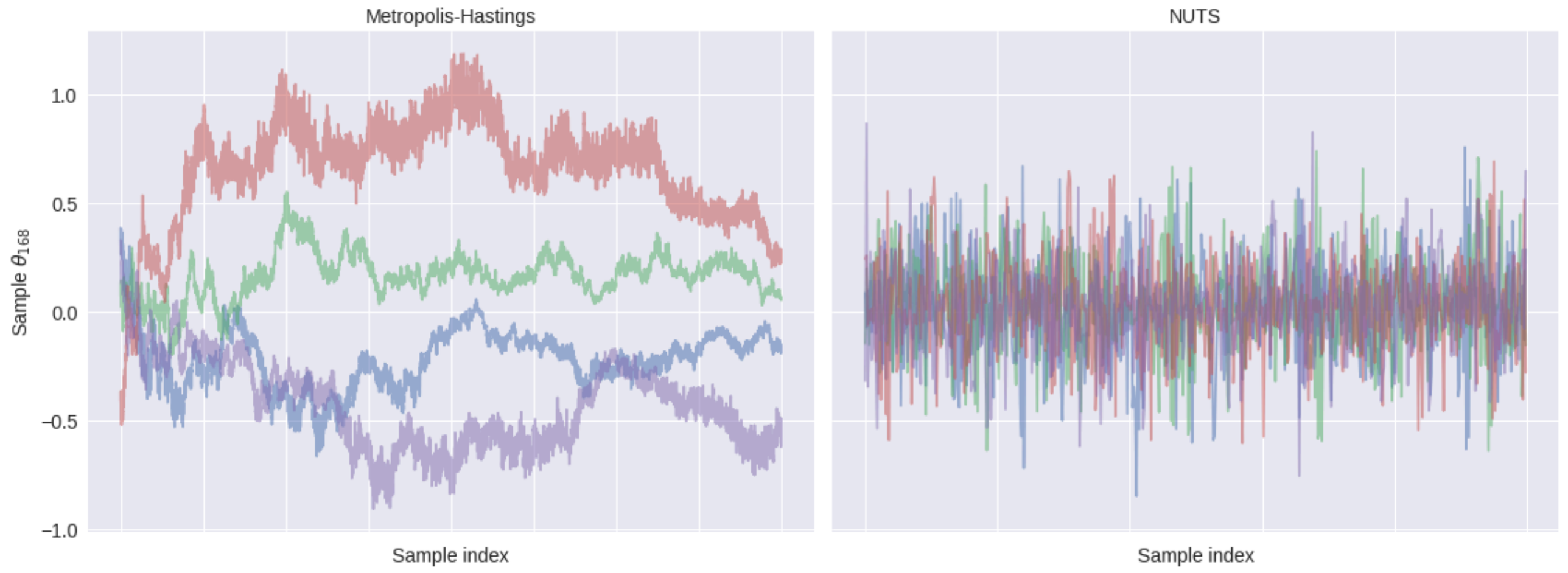
Larger than *N* ess in NUTS

NUTS can be antithetical

The desideratum for a sampler Andrew laid out to Matt was to maximze expected squared transition distance. Why? Because that's going to maximize effective sample size. (I still hadn't wrapped my head around this when Andrew was laying it out.) Matt figured out how to achieve this goal by building an algorithm that simulated the Hamiltonian forward and backward in time at random, doubling the time at each iteration, and then sampling from the path with a preference for the points visited in the final doubling. This tends to push iterations away from their previous values. In some cases, it can lead to anticorrelated chains.

http://andrewgelman.com/2018/01/18/measuring-speed-stan-incorrectly-faster-thought-cases-due-antithetical-sampling/

Superiority of NUTS



https://twitter.com/AustinRochford/status/992135745057054721

Superiority of NUTS

More problematically, Random walk Metropolis can fail silently:

Inferencing trigonometric time series model

Questions

narendramukherjee 2 6d

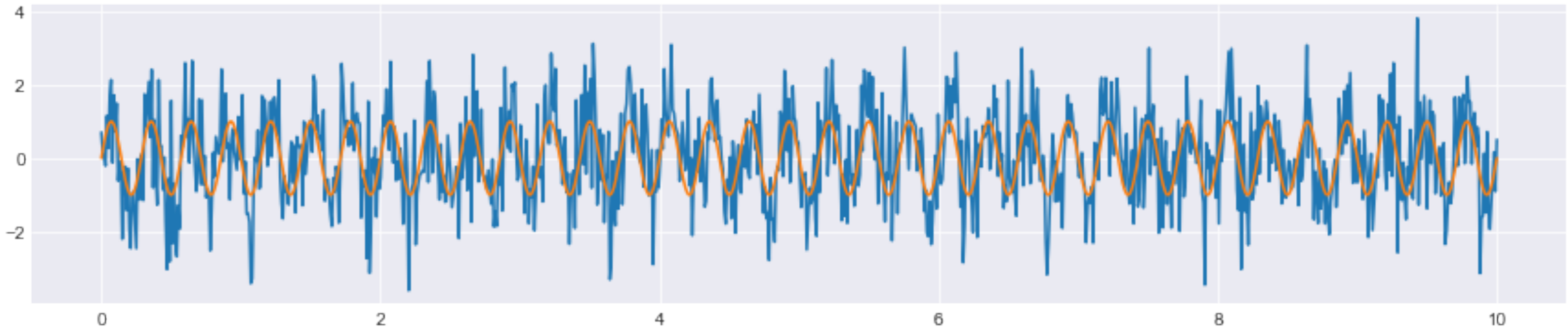
Following the conversation in How to model sinusoids in white gaussian noise, I have been trying to fit a sinusoidal model as well. I have tried a lot of different parametrizations with NUTS, some of which are as follows:

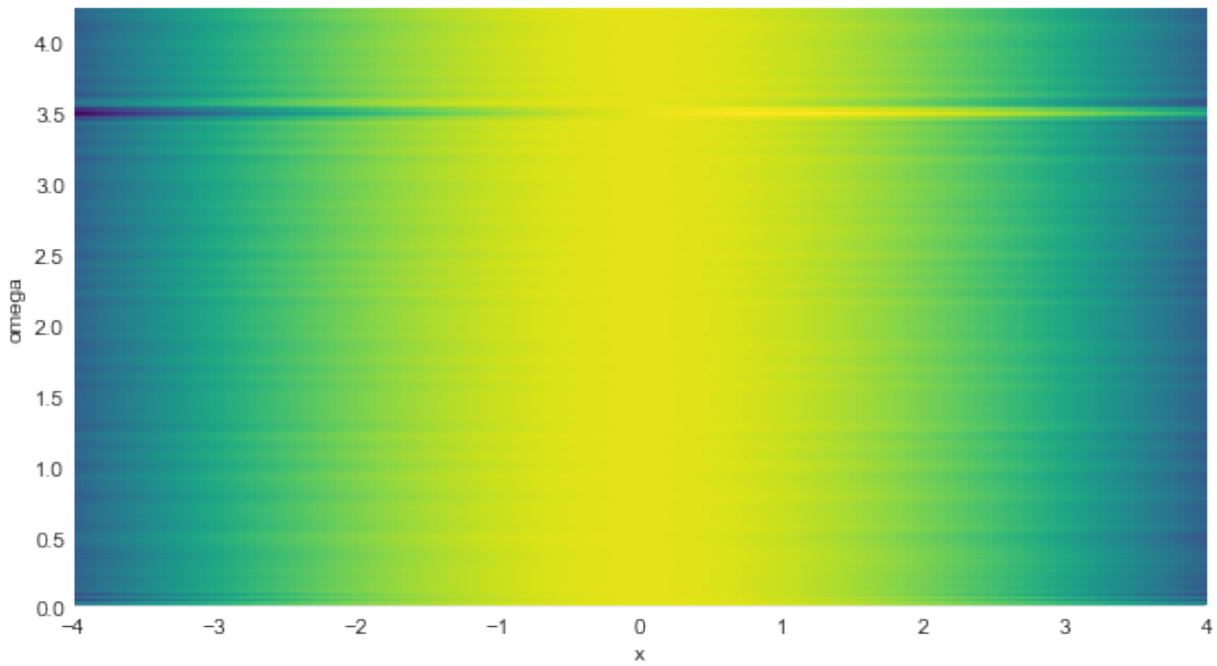
1. Try to fit $y(t) = Asin(\omega t) + Bcos(\omega t) + noise$ where A and B are drawn as Normal random variables.
2. Draw A and B as Normal random variables, and fit $y(t) = Csin(\omega t + \phi)$ where $C = \sqrt{A^2 + B^2}$ and $\phi = tan^{-1}\frac{B}{A}$.
   I tried some others which were worse than these two.

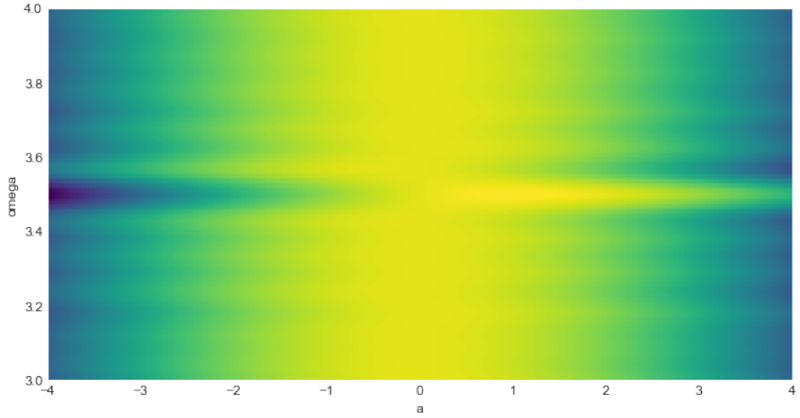
The conversation has revolved around the phase parameter mostly in this discussion, but I think the issue lies more with NUTS having problems in a nonlinear model. VERY suprisingly (to me at least!), Metropolis works fabulously in this model (I tried the parametrization number 1 above). There has been at least one previous report of NUTS being miserable in a nonlinear model when Metropolis worked well, and it seems that issue wasn't resolved back then:

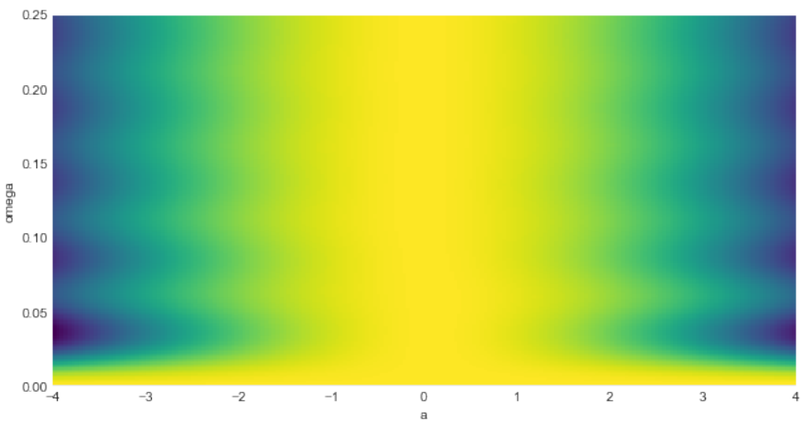
https://discourse.pymc.io/t/inferencing-trigonometric-time-series-model/1190

Trigonometric time series model









Code9 - Timeserie_model.ipynb

Alternative sampler in PyMC3

Slice sampler

Sequential Monte Carlo (SMC) sampler

Differential Evolution Metropolis sampling (DEMetropolis)

Stochastic Gradient Fisher Scoring (SGFS)

(upcoming) Likelihood-Free Inference

Challenge in implementing NUTS

axch commented 9 days ago — Collaborator

We are all happy to know you're interested in helping! You should know that we're going for an implementation that can run in graph mode, and can batch together multiple independent chains running in tandem (or at least their leapfrog steps). Since different chains can take different numbers of leapfrog steps during one update, the solution is proving surprisingly complicated---it basically amounts to writing a multi-stage compiler targeting TF graphs, which implements the logic needed for batching and recursion (which, however, should be reusable).

Given the scope, off the top of my head I would guess the implementation roadmap will take at least another full-time engineer-month. If you are interested in diving in, we can look for a place that would make sense, but it may take a while to get you up to speed on the design and what has happened so far.

The new age of Bayesian computation that is NUTS

- We can fit large complex model much easier.
  - Hierarchical linear model and mixed effect model with complex random structure
- Need to evaluate the model assumption more carefully
  - More realistic prior
- Older model that doesn't work out of the box
  - MLE or MAP does not care about the volume
- Mixing with other sampler

Code10 - Schizophrenic_case_study.ipynb

PyMC Developers
Probabilistic Programming in Python
May 17 · 4 min read

Theano, TensorFlow and the Future of PyMC

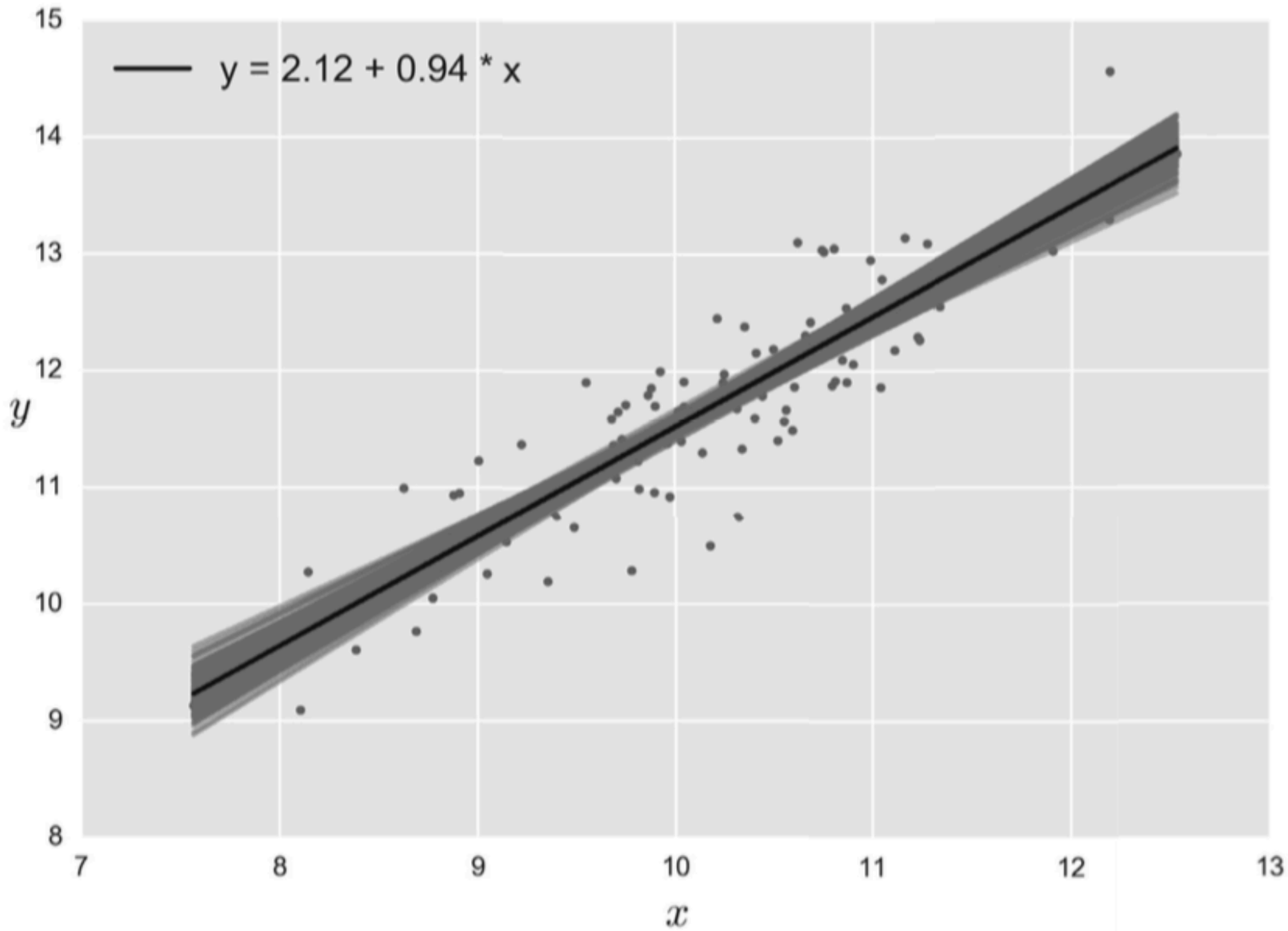
PyMC3 is an open-source library for Bayesian statistical modeling and inference in Python, implementing gradient-based Markov chain Monte Carlo, variational inference, and other approximation methods. These algorithms currently rely on Theano for computation, specifically for providing gradients.

. . .

Since the Theano team announced that it would cease development and maintenance of Theano within a year, we, the PyMC developers, have been actively discussing what to do about this. In this post we want to make two big announcements:

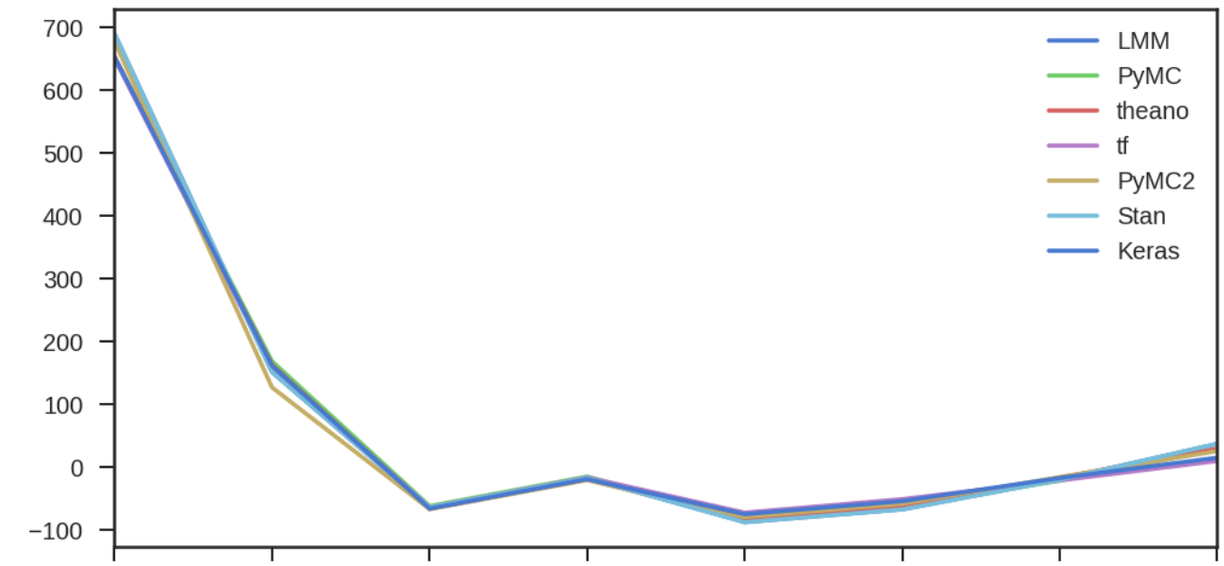
https://medium.com/@pymc_devs/theano-tensorflow-and-the-future-of-pymc-6c9987bb19d5

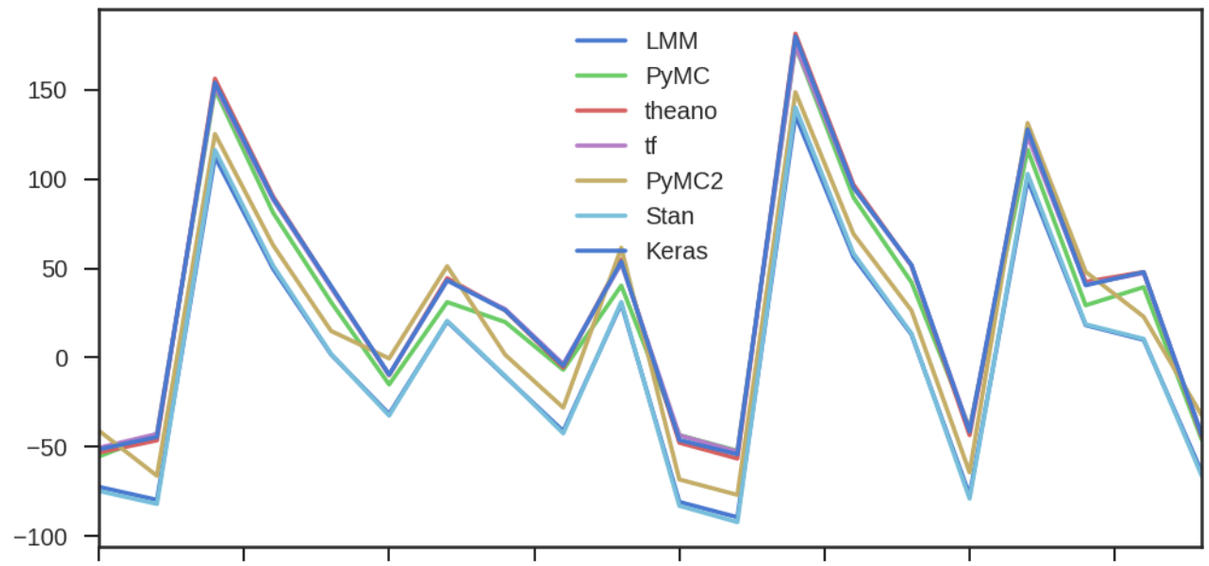
How good is the fit?

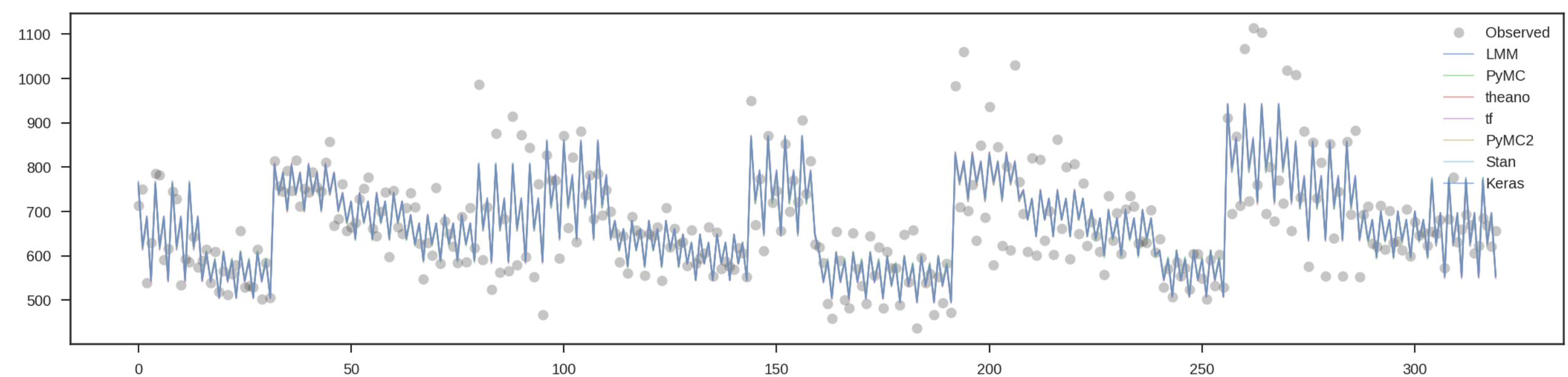


Point prediction

Different estimation can have (seemly) identical fit!



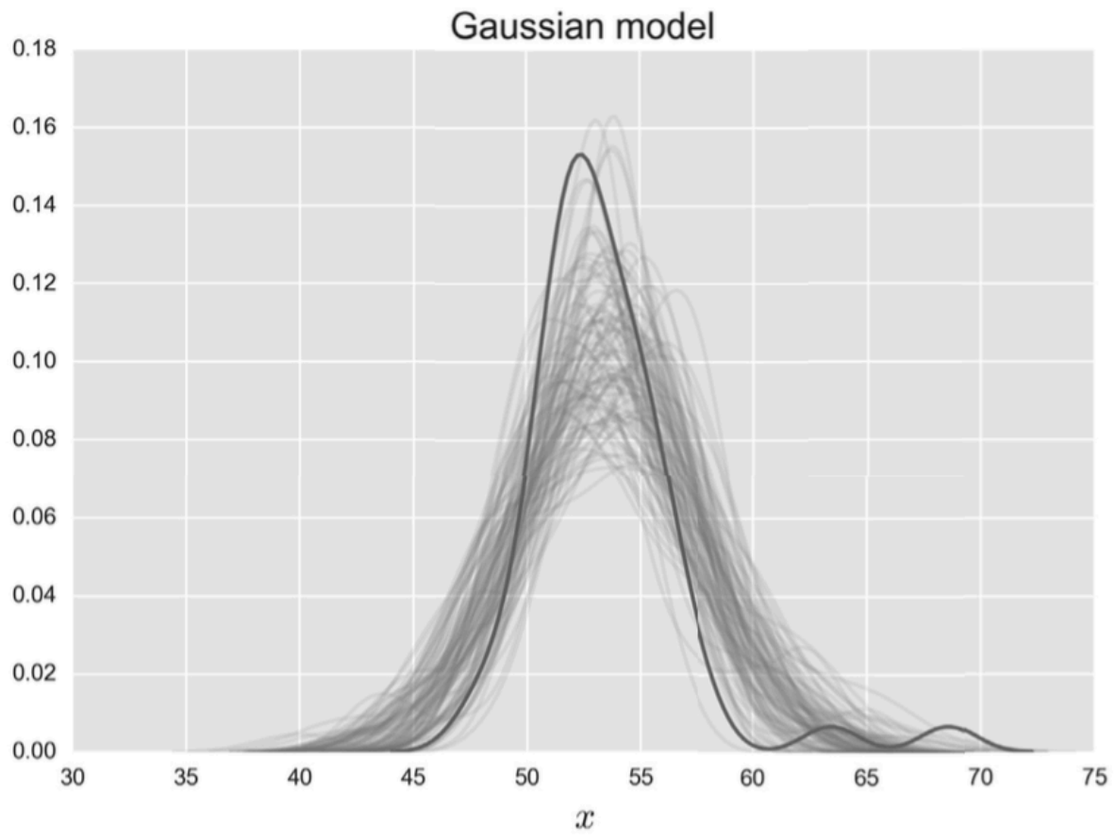


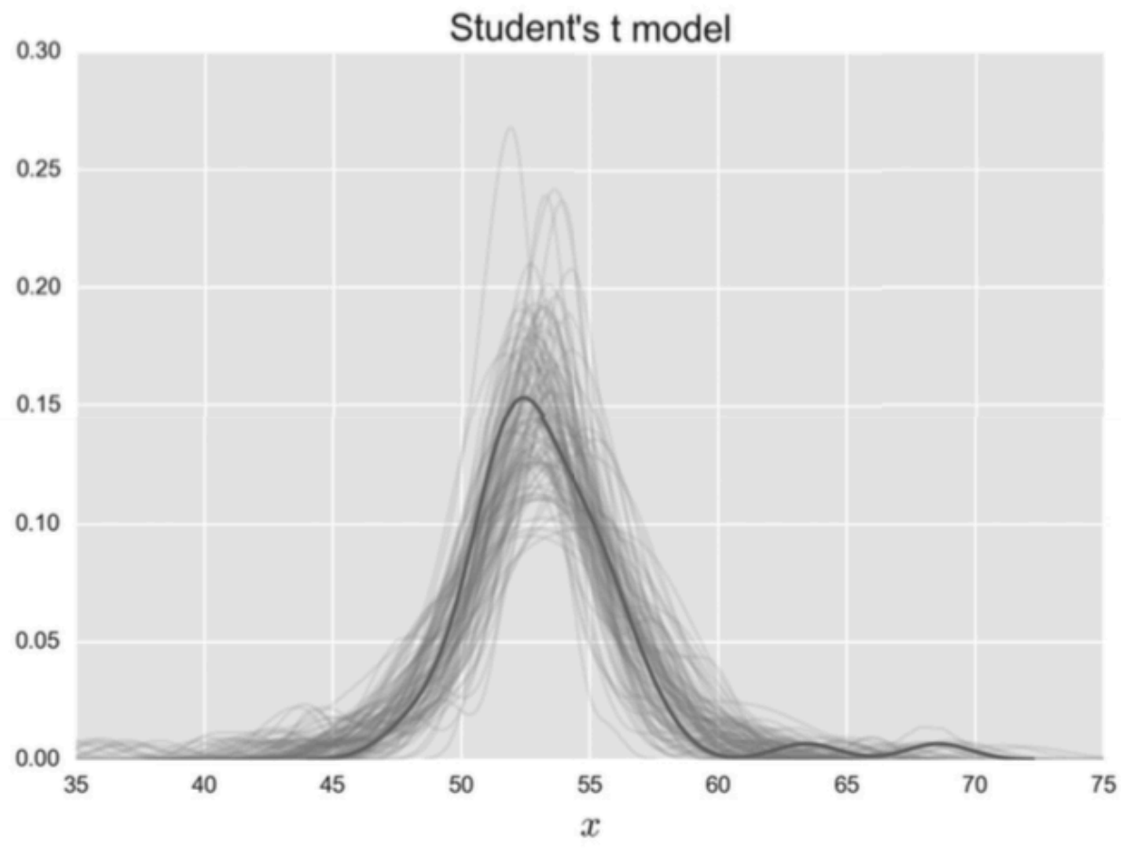


How good is the fit?

The posterior predictive distribution:

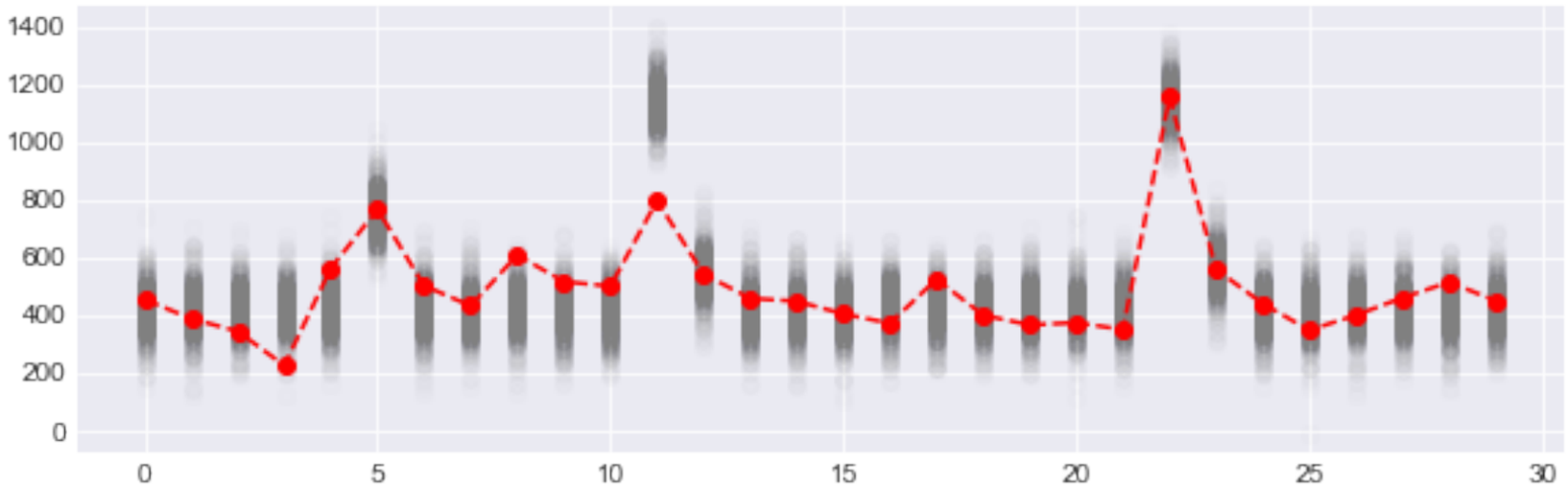
$$p(\tilde{y}|y) = \int p(\tilde{y}|\theta) f(\theta|y) d\theta$$

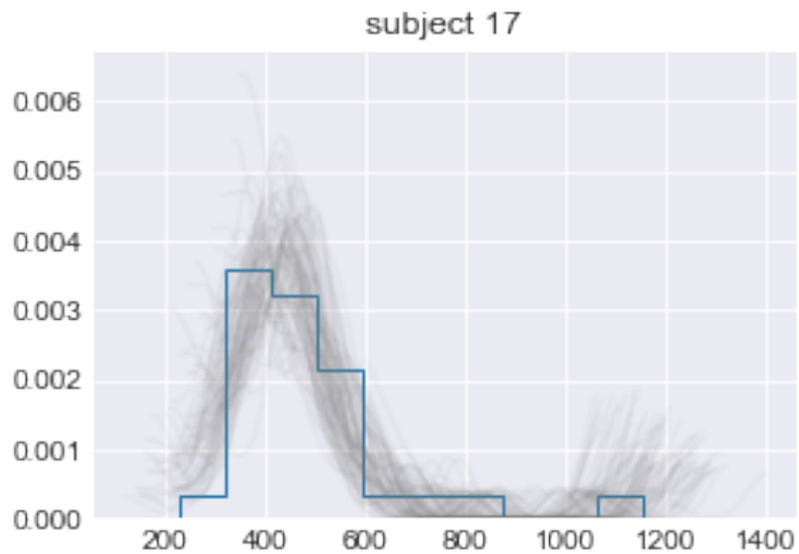


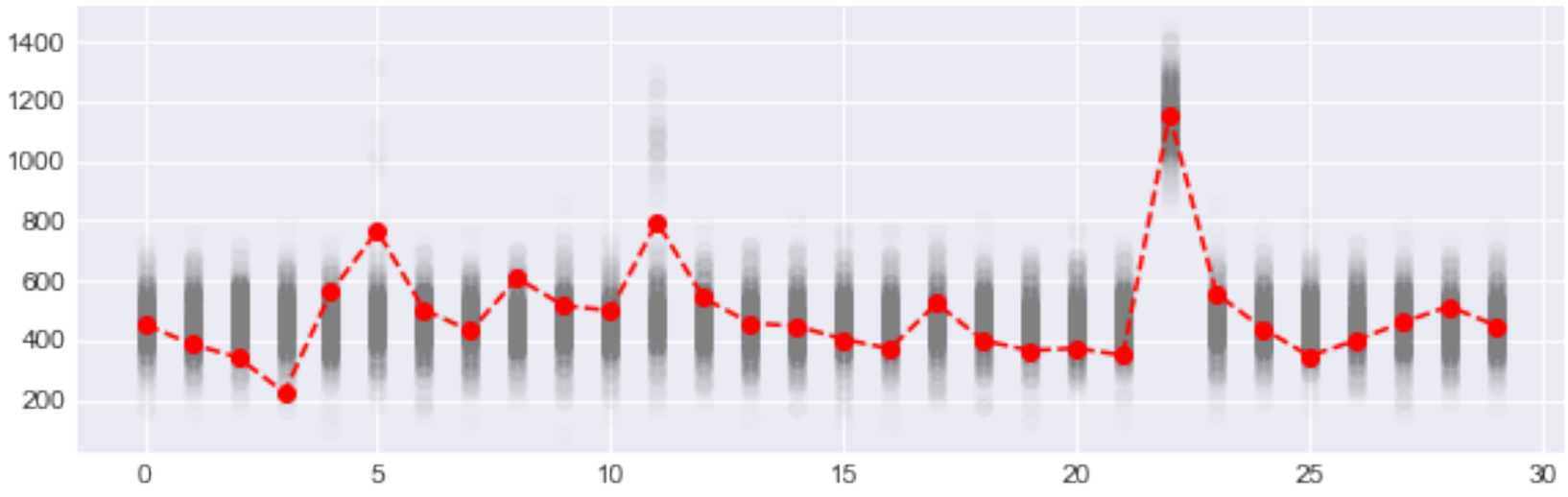


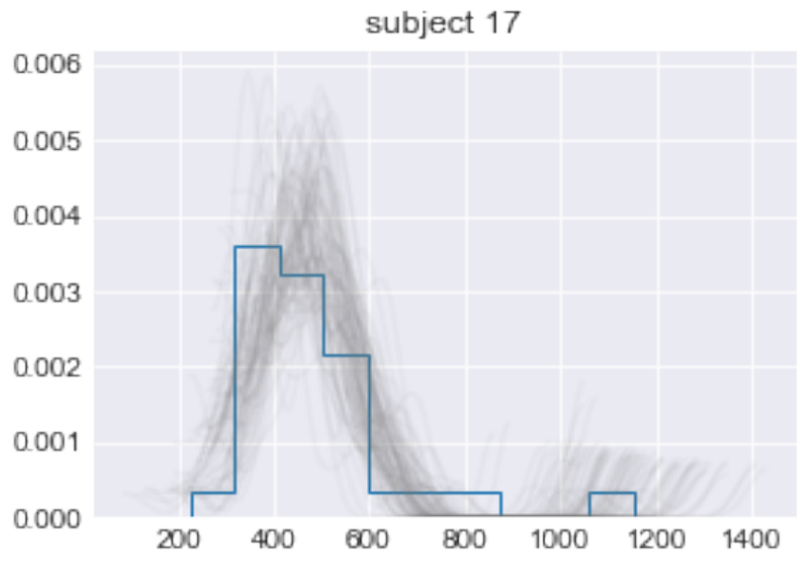
Code10 - Schizophrenic_case_study.ipynb

How good is the fit?

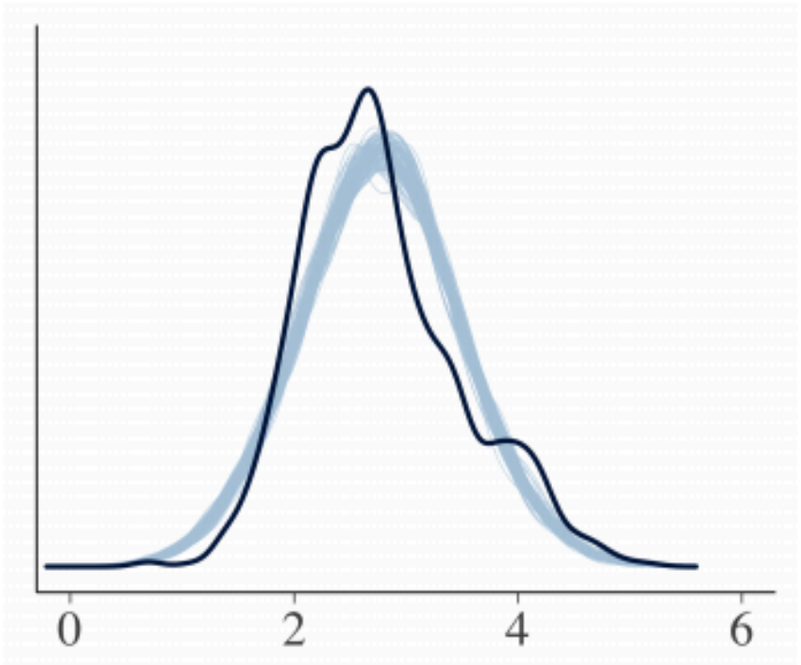




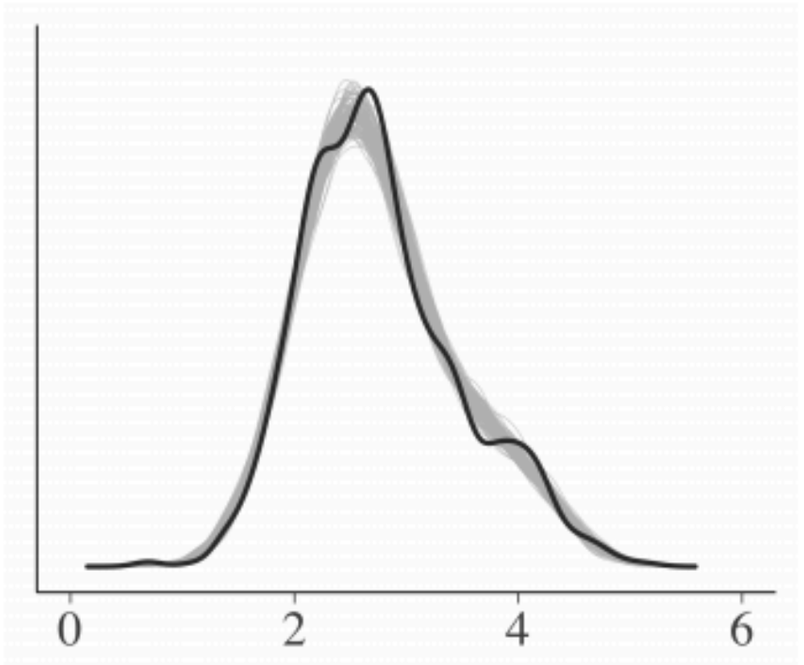




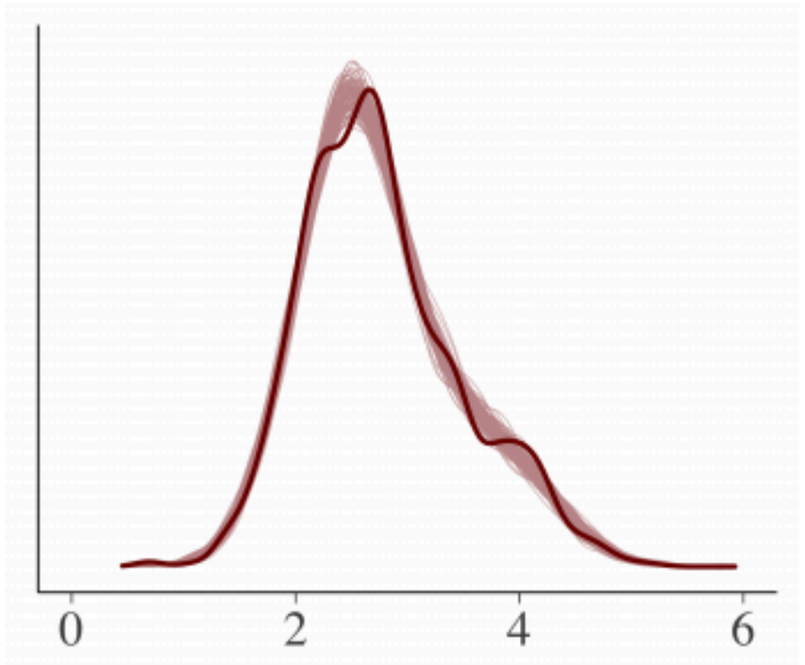
Visualization in Bayesian workflow



(a) Model 1



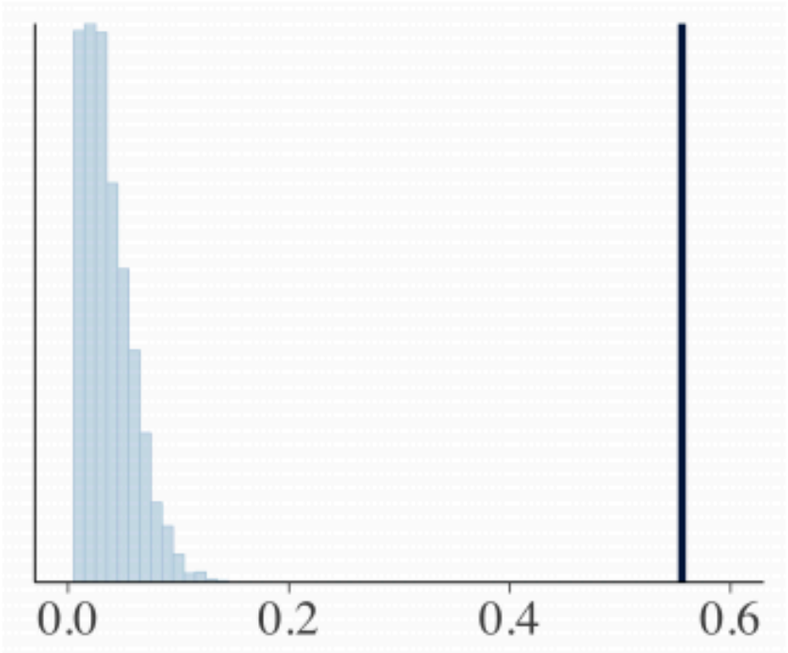
(b) Model 2



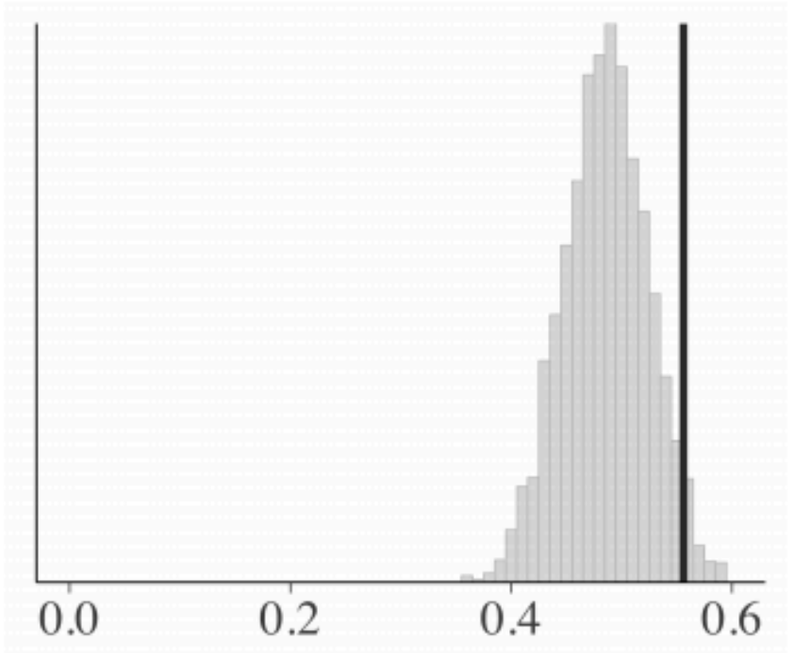
(c) Model 3

https://arxiv.org/pdf/1709.01449.pdf

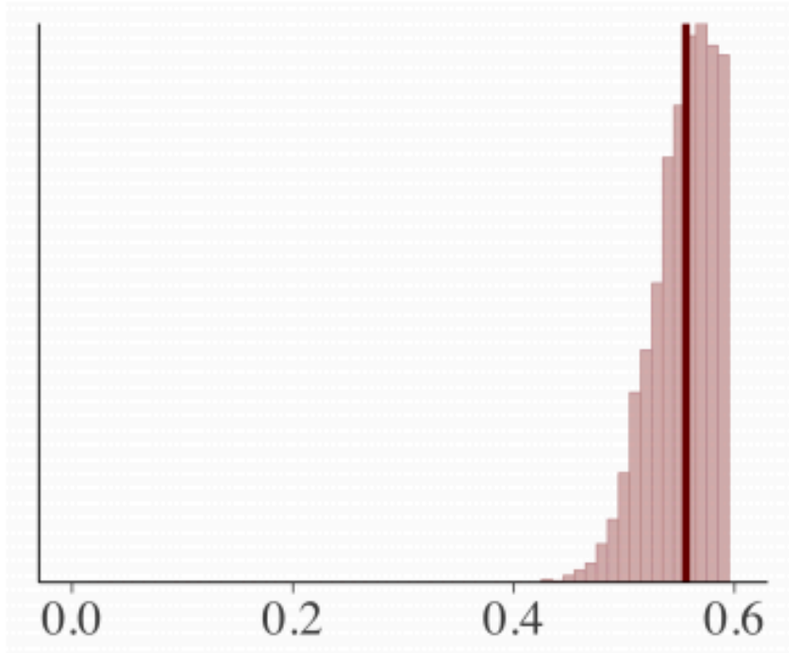
Visualization in Bayesian workflow



(a) Model 1



(b) Model 2



(c) Model 3

https://arxiv.org/pdf/1709.01449.pdf

Model evaluation

- Deviance information criterion (DIC)
- Bayesian Predictive Information Criterion (BPIC)
- The widely applicable or Watanabe-Akaike information criterion (WAIC)
- Pareto smoothed importance sampling leave-one-out cross-validation (PSIS-LOO)

Model evaluation

PSIS-LOO, WAIC, DIC, AIC, etc are all approximations to a fundamentally-uncalculatable number that exactly quantifies predictive accuracy. Each of those approximations require different assumptions to be reasonably accurate.

One of the great features of PSIS-LOO and WAIC is the self-diagnostic.

PSIS-LOO and WAIC in PyMC3

```
pm.loo(trace2, mariginal)
```

```
/Users/jlao/Documents/Github/pymc3/pymc3/stats.py:292: UserWarning: Estimated shape parameter
of Pareto distribution is
        greater than 0.7 for one or more samples.
        You should consider using a more robust model, this is because
        importance sampling is less likely to work well if the marginal
        posterior and LOO posterior are very different. This is more likely to
        happen with a non-robust model and highly influential observations.
  happen with a non-robust model and highly influential observations.""")
```

```
LOO_r(LOO=6419.625727828014, LOO_se=94.11283847318855, p_LOO=99.06232388543867, shape_warn=1)
```

```
pm.waic(trace2, mariginal)
```

```
/Users/jlao/Documents/Github/pymc3/pymc3/stats.py:211: UserWarning: For one or more samples t
he posterior variance of the
        log predictive densities exceeds 0.4. This could be indication of
        WAIC starting to fail see http://arxiv.org/abs/1507.04544 for details

  """)
```

```
WAIC_r(WAIC=6409.393861614912, WAIC_se=93.911097482862, p_WAIC=93.94639077888765, var_warn=1)
```

Posterior predictive distribution

$$\begin{aligned}\text{lpd} &= \text{log pointwise predictive density} \\ &= \sum_{i=1}^{n} \log p(y_i|y) = \sum_{i=1}^{n} \log \int p(y_i|\theta)p(\theta|y)d\theta.\end{aligned}$$

$$\begin{aligned}\text{elpd} &= \text{expected log pointwise predictive density for a new dataset} \\ &= \sum_{i=1}^{n} \int p_t(\tilde{y}_i) \log p(\tilde{y}_i|y)d\tilde{y}_i,\end{aligned}$$

https://arxiv.org/pdf/1507.04544.pdf

Code11 - Roaches_case_study.ipynb

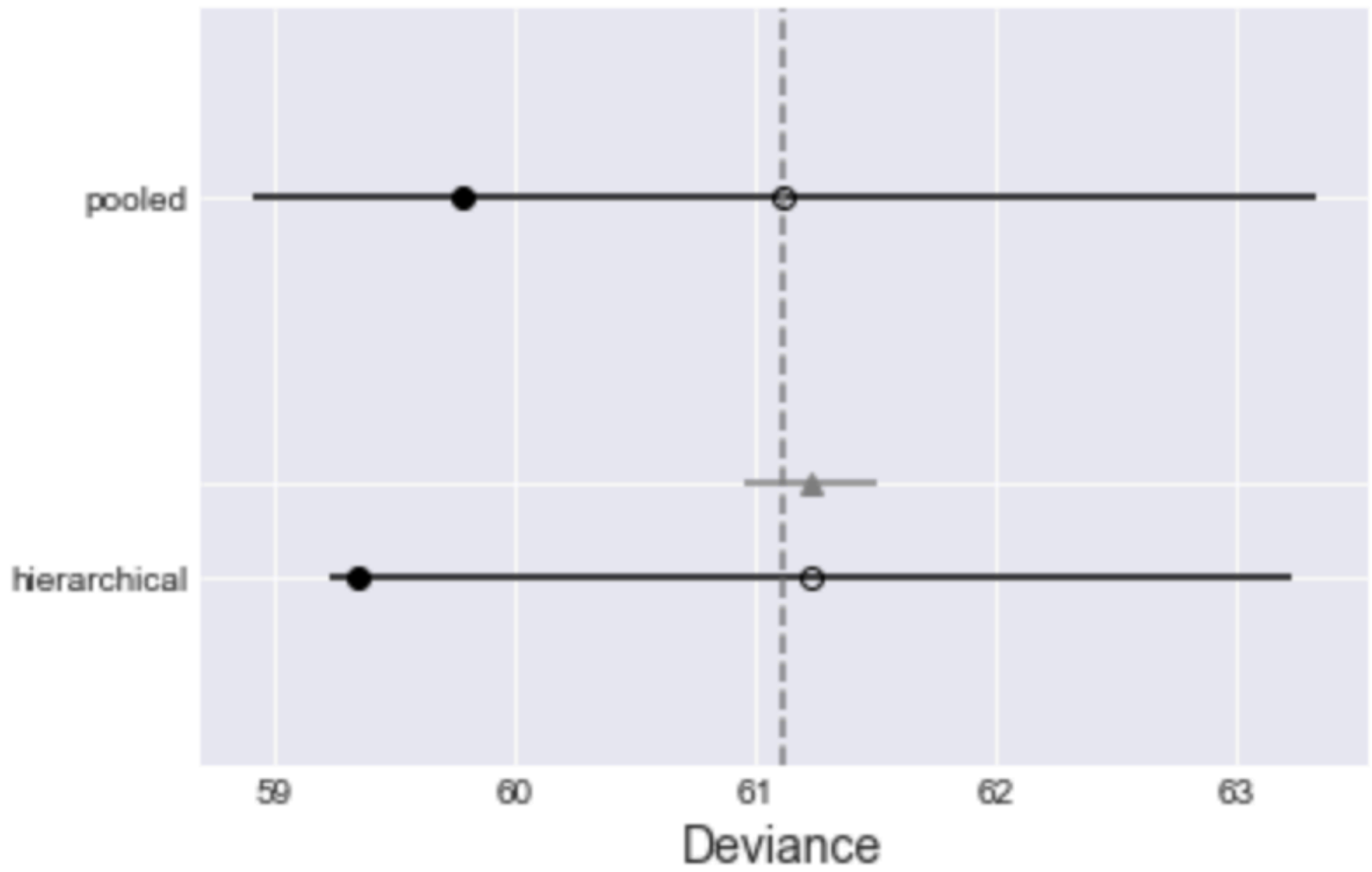
Some remark on WAIC and PSIS- LOO

Aki Vehtari: “I also recommend using PSIS-LOO instead of WAIC, because it’s more reliable and has better diagnostics as discussed in our paper), but if you insist to have one information criterion then leave WAIC”.

Alternatively, Watanabe says: “WAIC is a better approximator of the generalization error than the pareto smoothing importance sampling cross validation. The Pareto smoothing cross validation may be the better approximator of the cross validation than WAIC, however, it is not of the generalization error”.

Model comparison and selection

```
pm.compareplot(df_comp_WAIC);
```



http://discourse.mc-stan.org/t/interpreting-elpd-diff-loo-package/1628/2

Bayesian model averaging

Using the weight provided from pm.compare, and make Weighted posterior predictive samples using pm.sample_ppc_w

http://docs.pymc.io/notebooks/model_averaging.html

CASE STUDIES

Advance topics

- Compound Step Explanation
- Laplace approximation in PyMC3
- Inferencing Linear Mixed Model with EM
- Box-Cox transformation
- Simulation Based Calibration

SESSION 2: MODEL INFERENCE AND COMPARISON