Model Fitting

I have data.

I have a model.*

I want to learn about my parameters of my model (and/or physics) based on my data.

* often based on physics

Bayesian Approach to Model Fitting

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Bayesian Approach to Model Fitting

D = Data
Theta = Model parameters

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Bayesian Approach to Model Fitting

D = Data
Theta = Model parameters

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Probability of Theta conditioned on (or given) on Data
“**Posterior Probability Distribution Function**”
aka
“**The Posterior**”

Bayesian Approach to Model Fitting

D = Data
Theta = Model parameters

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Probability of Data conditioned on (or given) Theta
“**Likelihood Function**” or “**Data Model**”
aka
“**The Likelihood**”

Bayesian Approach to Model Fitting

D = Data
Theta = Model parameters

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Marginal or Prior Probability(ies) for Theta
aka
“The Prior(s)”

Bayesian Approach to Model Fitting

D = Data
Theta = Model parameters

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Probability of the Data
or
“The Evidence”

Bayesian Approach to Model Fitting

Likelihood Prior

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Posterior

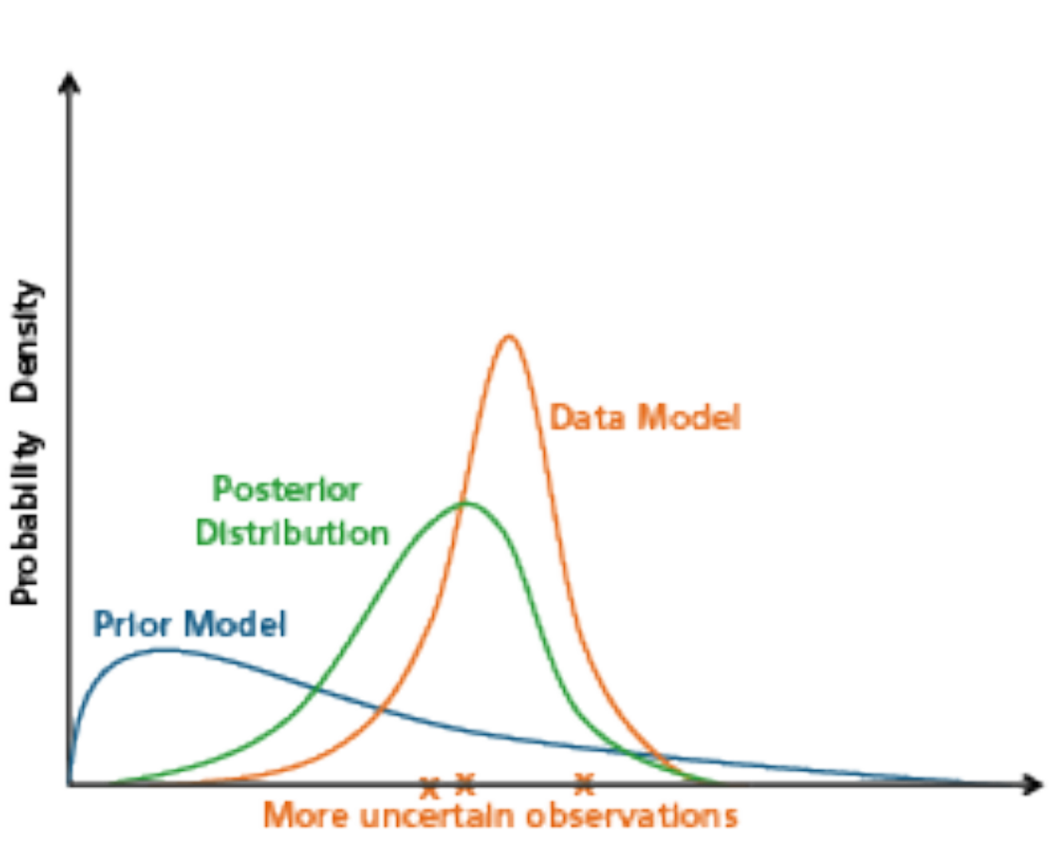
Evidence

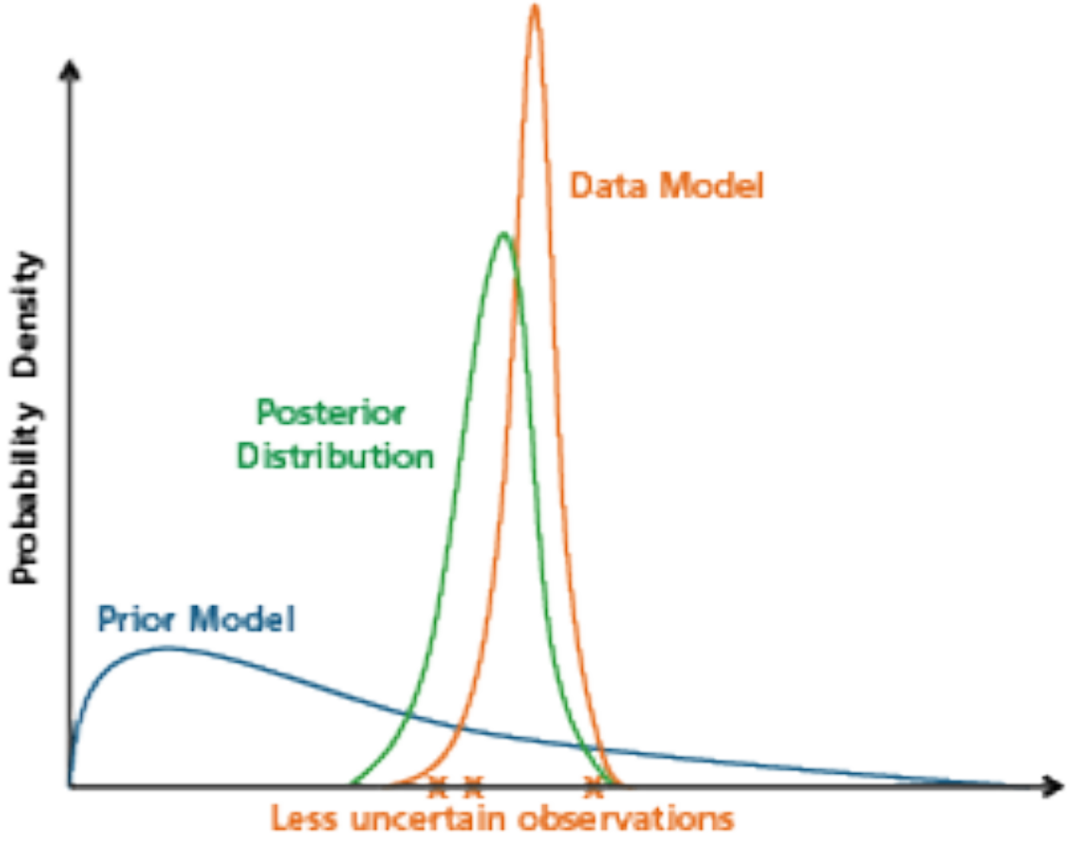
D = Data

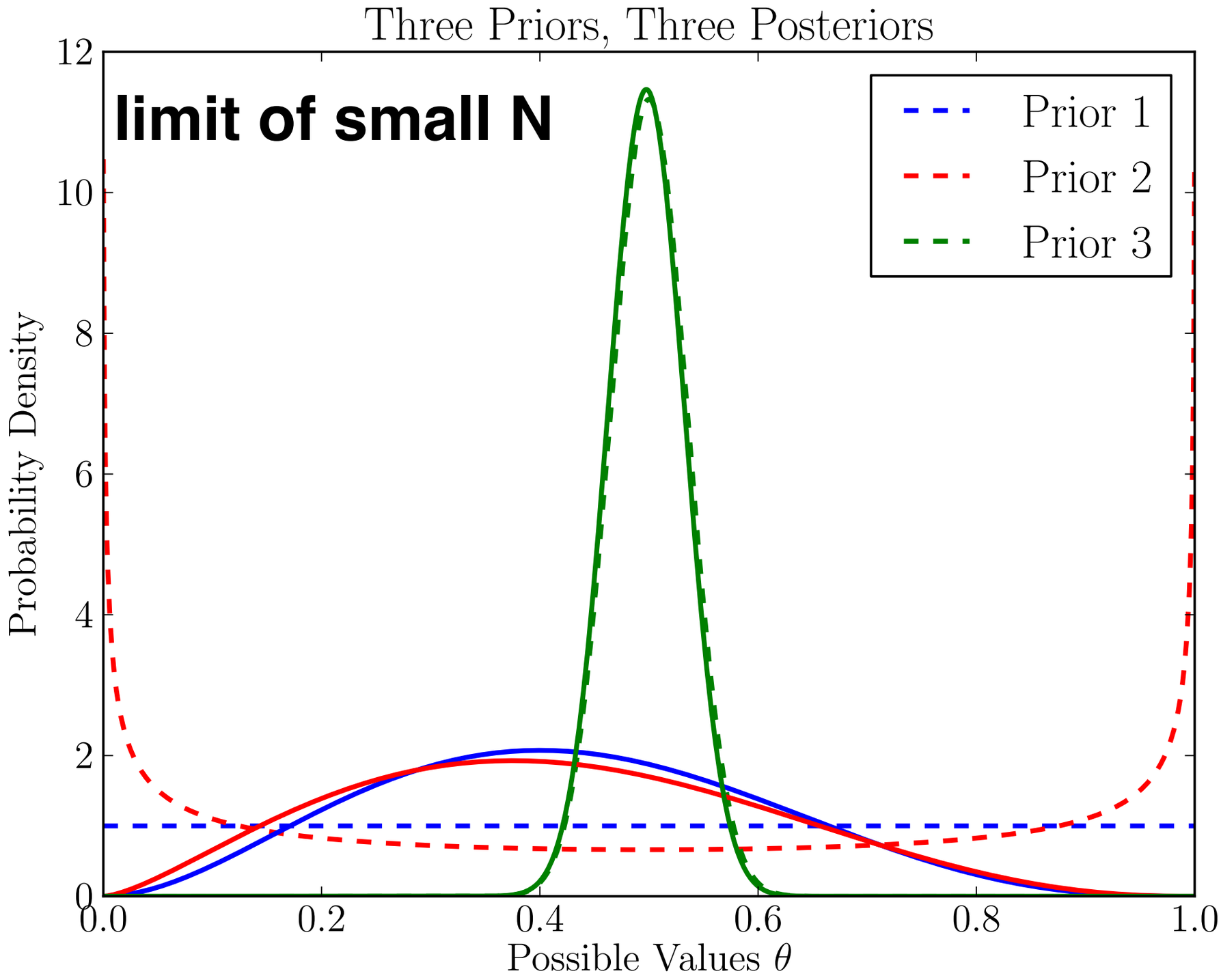
Theta = Model parameters

Updating Beliefs: Graphically

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$



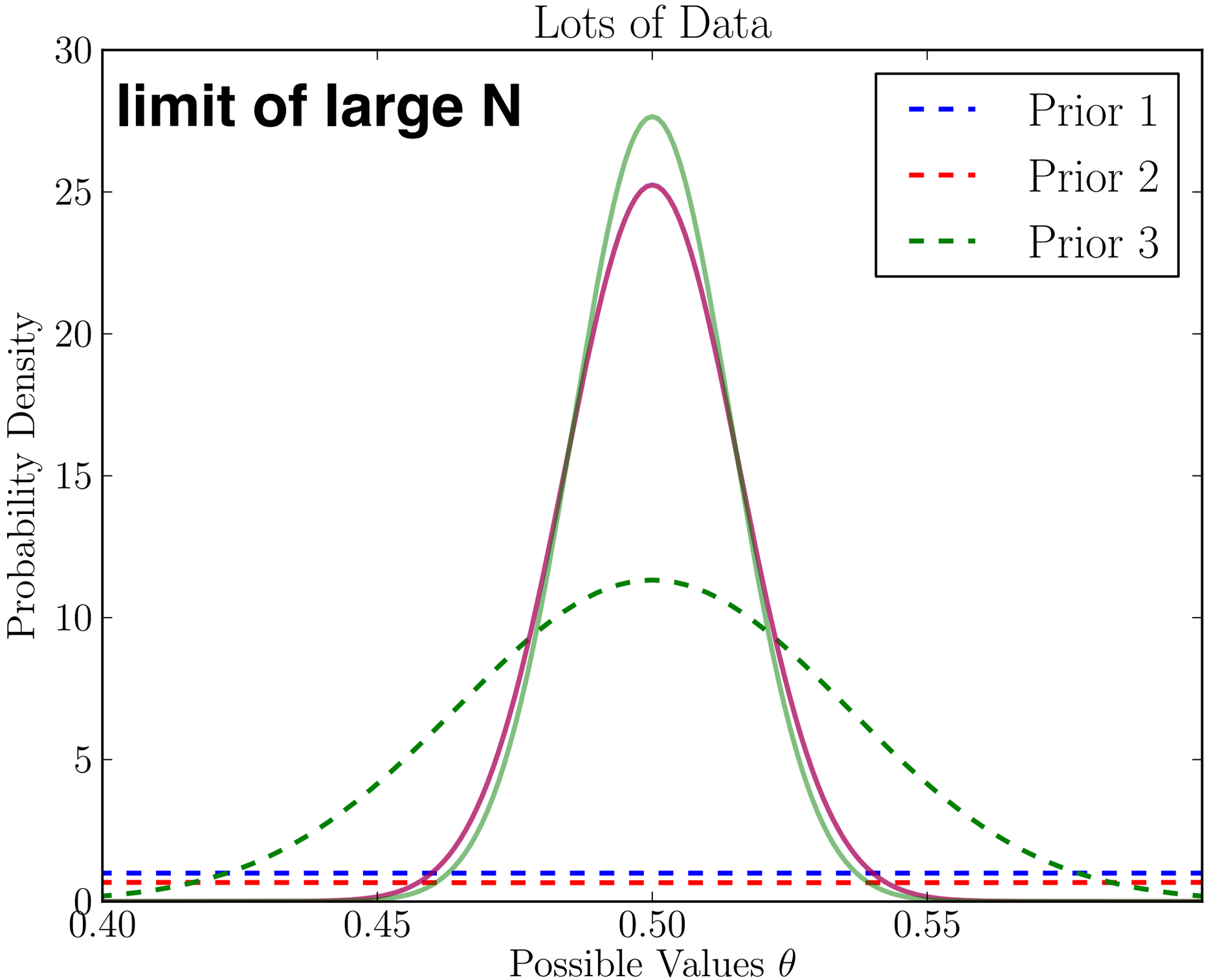




prior 1: “flat prior”, “prior ignorance”, “uninformative prior” (“improper”)

prior 2: emphasizing the extremes (“informative prior”)

prior 3: good prior knowledge (“informative prior”)


Lots of Data
limit of large N
Prior 1
Prior 2
Prior 3
Probability Density
30
25
20
15
10
5
0
0.40
0.45
0.50
0.55
Possible Values θ

Data analysis recipes: Fitting a model to data*

David W. Hogg
Center for Cosmology and Particle Physics, Department of Physics, New York University
Max-Planck-Institut für Astronomie, Heidelberg

Jo Bovy
Center for Cosmology and Particle Physics, Department of Physics, New York University

Dustin Lang
Department of Computer Science, University of Toronto
Princeton University Observatory

Abstract

We go through the many considerations involved in fitting a model to data, using as an example the fit of a straight line to a set of points in a two-dimensional plane. Standard weighted least-squares fitting is only appropriate when there is a dimension along which the data points have negligible uncertainties, and another along which all the uncertainties can be described by Gaussians of known variance; these conditions are rarely met in practice. We consider cases of general, heterogeneous, and arbitrarily covariant two-dimensional uncertainties, and situations in which there are bad data (large outliers), unknown uncertainties, and unknown but expected intrinsic scatter in the linear relationship being fit. Above all we emphasize the importance of having a "generative model" for the data, even an approximate one. Once there is a generative model, the subsequent fitting is non-arbitrary because the model permits direct computation of the likelihood of the parameters or the posterior probability distribution. Construction of a posterior probability distribution is indispensible if there are "nuisance parameters" to marginalize away.

It is conventional to begin any scientific document with an introduction that explains why the subject matter is important. Let us break with tradition and observe that in almost all cases in which scientists fit a straight line to their data, they are doing something that is simultaneously *wrong* and *unnecessary*. It is wrong because circumstances in which a set of two

*The notes begin on page 39, including the license[1] and the acknowledgements[2].

“All models are wrong, but some are useful.”

~ George Box, 1978

Well-known example

$$PV = nkT$$

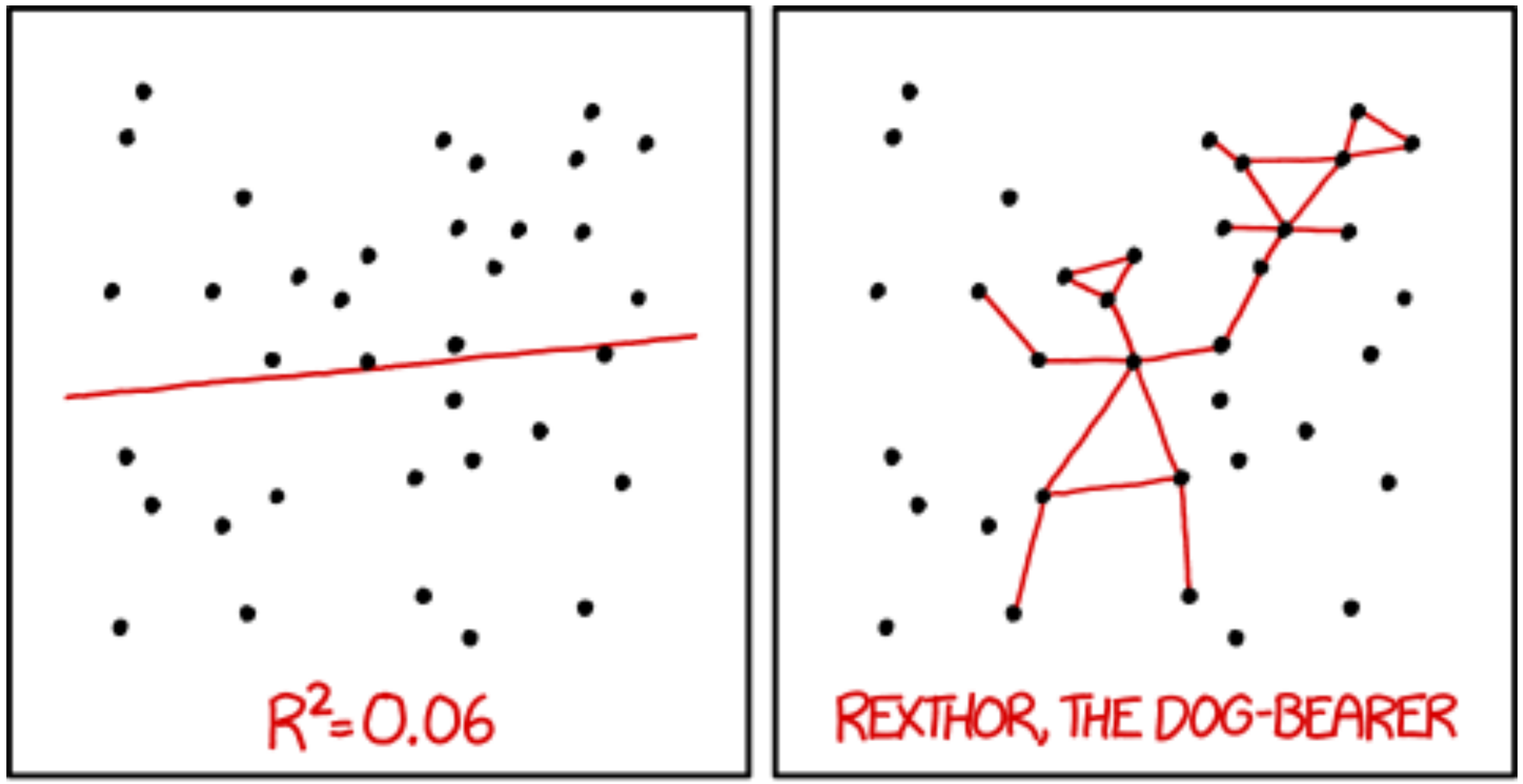
Well-known example

$$y = mx + b$$

“A model is a simplification or approximation of reality and hence will not reflect all of reality. …

Box noted that “all models are wrong, but some are useful.” While a model can never be “truth,” a model might be ranked from very useful, to useful, to somewhat useful to, finally, essentially useless.”

~ Burnham & Anderson, 1998



I DON'T TRUST LINEAR REGRESSIONS WHEN IT'S HARDER TO GUESS THE DIRECTION OF THE CORRELATION FROM THE SCATTER PLOT THAN TO FIND NEW CONSTELLATIONS ON IT.

http://xkcd.com/1725

“Parameter Estimation”

Given a model, what can we learn about its parameters

e.g., **y = mx + b**

“Model Selection”

Given multiple models, can we decide which one better represents the data

e.g.,

y = mx + b

VS.

y = b + mx + nx^2 + px^3

“Parameter Estimation”

We can compute the posterior up to a normalization constant:

$$P(\theta|D) = \frac{1}{Z} P(D|\theta)\, P(\theta)$$

This constant, **Z**, is the evidence:

$$Z = P(D) = \int P(D|\theta)\, P(\theta) d\theta$$

“Parameter Estimation”

Usually done in (natural) log-probability space.

$$P(\theta|D) = \frac{1}{Z} P(D|\theta)\, P(\theta) \longrightarrow$$

$$\ln P(\theta|D) = \ln P(D|\theta) + \ln P(\theta) + \ln\left(\frac{1}{Z}\right)$$

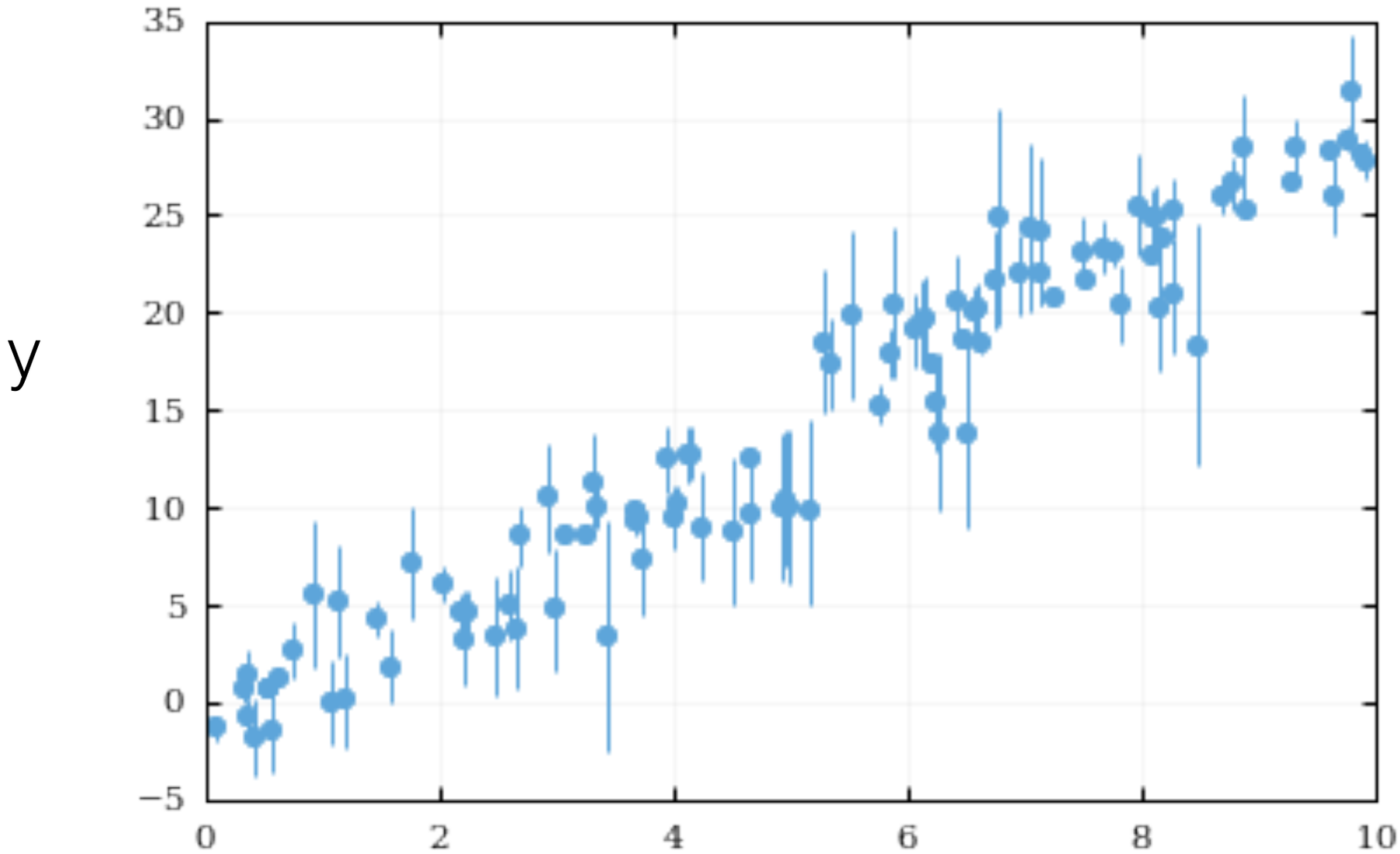
“ln P”

“ln like”
“log-likelihood”

“ln Prior”
“Prior”

just a constant, don’t need to compute

“Parameter Estimation”:
fitting a line to noisy data

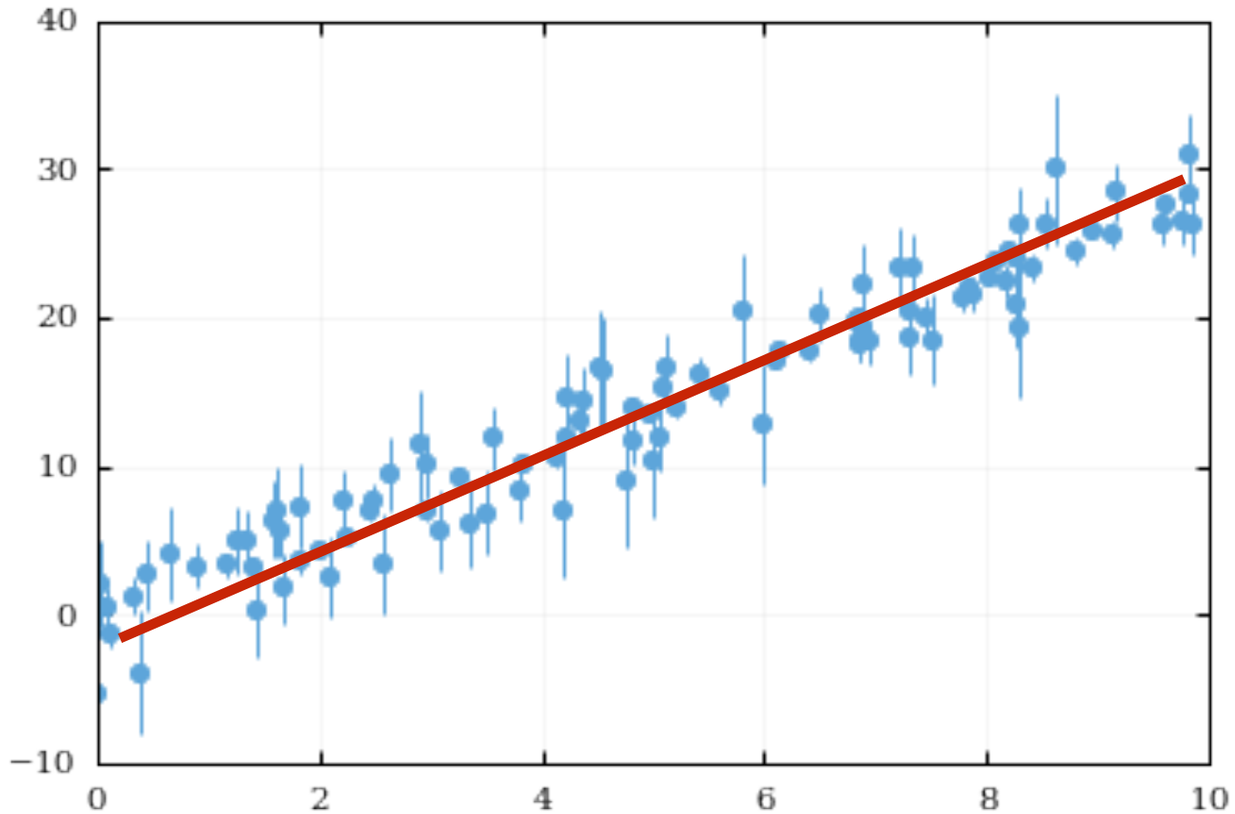


“Parameter Estimation”:
Graphically

The “scene”

y = mx + b

y



x

Observations

y_i

sigma_i

find values of **m** and **b**, given data (i.e., **y** and **sigma**)

“Parameter Estimation”:

Define Likelihood Function

Assume Gaussian error distribution for data, yields: χ^2

$$P(D|\theta) = \frac{1}{\sqrt{(2\pi\sigma^2)}}\, e^{\frac{-(y-\bar{y})^2}{2\sigma^2}}$$

<u>the data (or observations)</u>

$y = set\, of\, data\, points\, (\text{“}D\text{”})$

$\sigma = uncertainties$

<u>the model</u>

$\bar{y} = mx + b$

$\theta = (m, b)$

“Parameter Estimation”:
Define Likelihood Function

Assume Gaussian error distribution for data, yields: χ^2

$$P(D|\theta) = \prod_i^N \frac{1}{\sqrt{(2\pi\sigma_i^2)}}\, e^{\frac{-(y_i-\bar{y_i})^2}{2\sigma_i^2}}$$

assume data are drawn **“identically and independently”** (iid)

the data (or observations)

$y_i = ith\, data\, point$

$\sigma_i = uncertainty\, on\, ith\, data\, point$

the model

$\bar{y_i} = mx_i + b_i$

$\theta = (m, b)$

“Parameter Estimation”:
Define Likelihood Function

$$P(D|\theta) = \prod_{i}^{N} \frac{1}{\sqrt{(2\pi\sigma_i^2)}}\, e^{\frac{-(y_i - \bar{y_i})^2}{2\sigma_i^2}}$$

write as log-likelihood ⟶

$$\ln \mathrm{P}(\mathrm{D}|\theta) = -0.5 \sum_{\mathrm{i}}^{\mathrm{N}} \left(\frac{(\mathrm{y_i} - \bar{\mathrm{y_i^2}})}{\sigma_\mathrm{i}^2} + \ln(2\pi\sigma_\mathrm{i}^2) \right)$$

“Parameter Estimation”:

How do we actually get the values for m and b?

$$\ln P(\theta|D) = \ln P(D|\theta) + \ln P(\theta) + \ln(\frac{1}{Z})$$

“Parameter Estimation”:

How do we actually get the values for m and b?

1. Solve the problem exactly.
2. Compute P(x) at gridded values of x.
3. Optimize.
4. Sample.

Markov chain Monte Carlo Sampling aka MCMC

MCMC is a general purpose technique for generating fair samples from a probability distribution.

The resulting density of samples is proportional to the underlying probability distribution

You get marginalization for free

Markov chain Monte Carlo Sampling aka MCMC

Goal is to determine the form of:

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

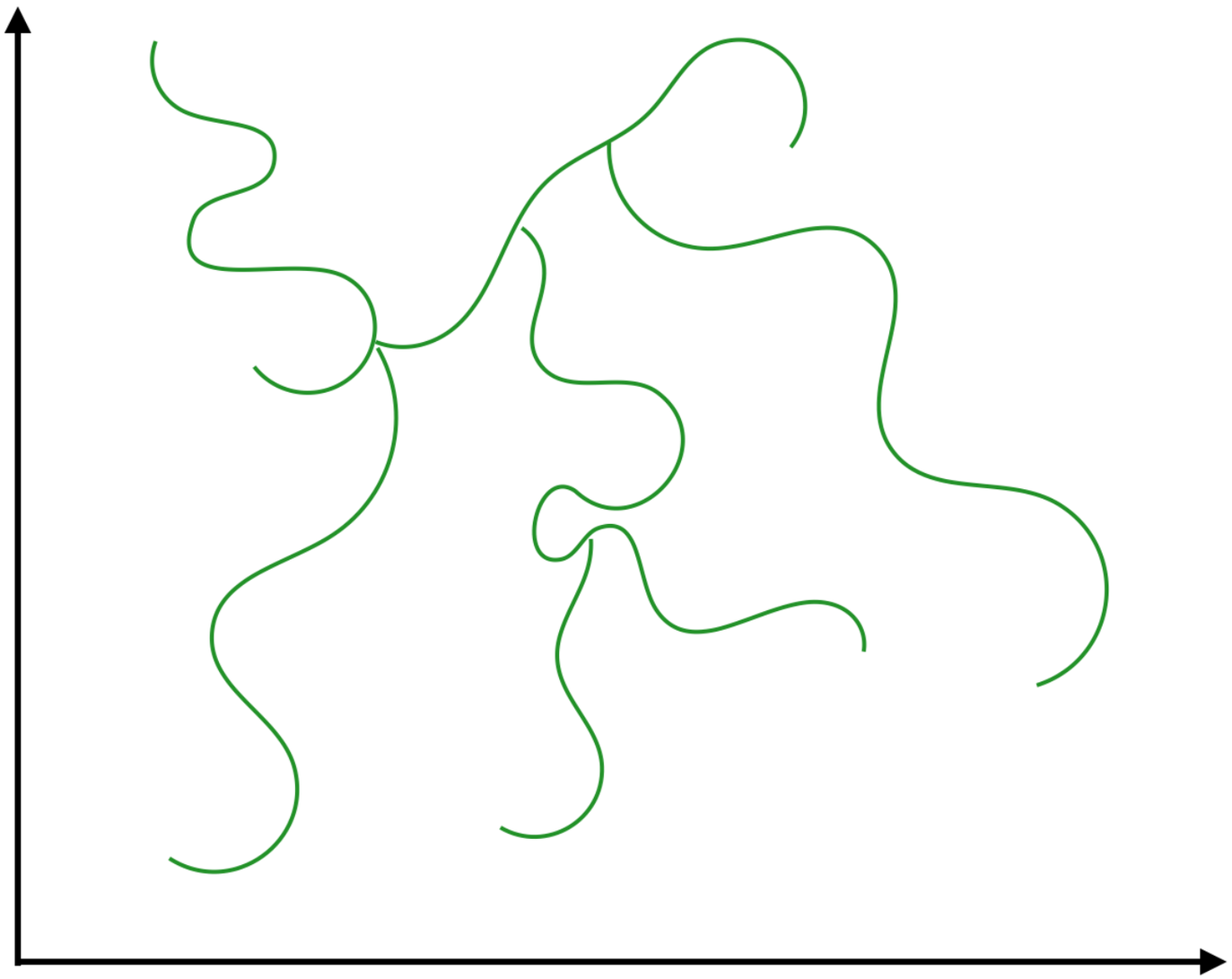
In low dimensions, this can be done efficiency in many ways

In high dimensions, it gets more complicated.

Markov chain Monte Carlo Sampling aka MCMC

Imagine we have some complicated function

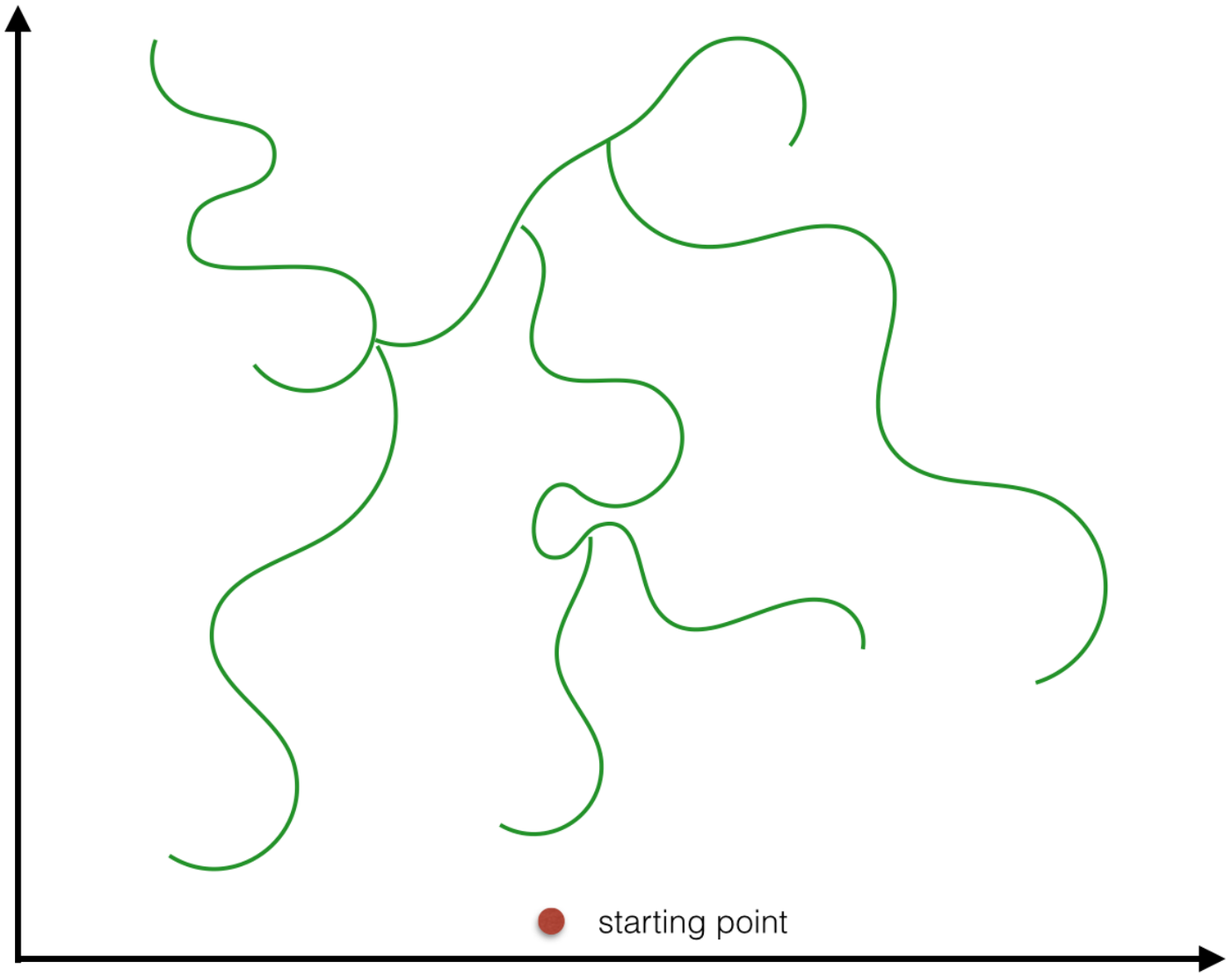
With high probability regions represented in green.



Markov chain Monte Carlo Sampling aka MCMC

Imagine we have some complicated function

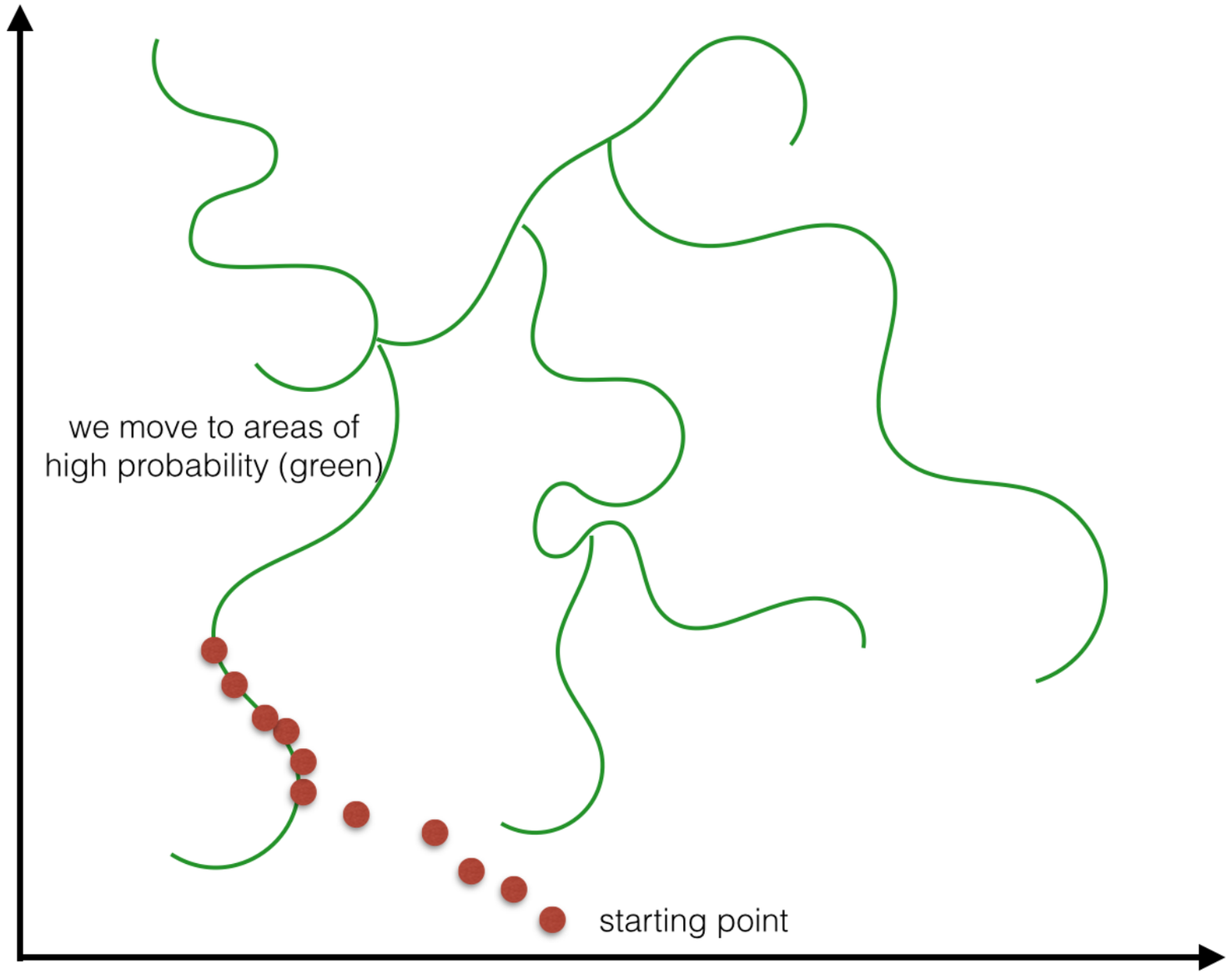
With high probability regions represented in green.



Markov chain Monte Carlo Sampling aka MCMC

Imagine we have some complicated function

With high probability regions represented in green.



Markov chain Monte Carlo Sampling aka MCMC

Markov chain — where we go next only depends on the previous state (Markov)

Monte Carlo — simulating the data

Metropolis MCMC Sampling

Metropolis Algorithm (1953)

1. Choose a starting point, θ^0.
2. For $t = 1, 2, \ldots$:
   (a) Sample a proposal θ^* from a proposal (jumping) distribution, $J_t(\theta^*|\theta^{t-1})$.
   The proposal distribution must be symmetric for the Metropolis algorithm, with $J_t(\theta_a|\theta_b) = J_t(\theta_b|\theta_a) \quad \forall\ \theta_a, \theta_b, t$
   (b) Calculate the ratio of the posteriors,

$$r = \frac{p(\theta^*|\mathbf{x})}{p(\theta^{t-1}|\mathbf{x})}$$

   (c) Set the next point in the Markov chain to

$$\theta^t = \begin{cases} \theta^* & \text{with probability } \min(r, 1) \\ \theta^{t-1} & \text{otherwise} \end{cases}$$

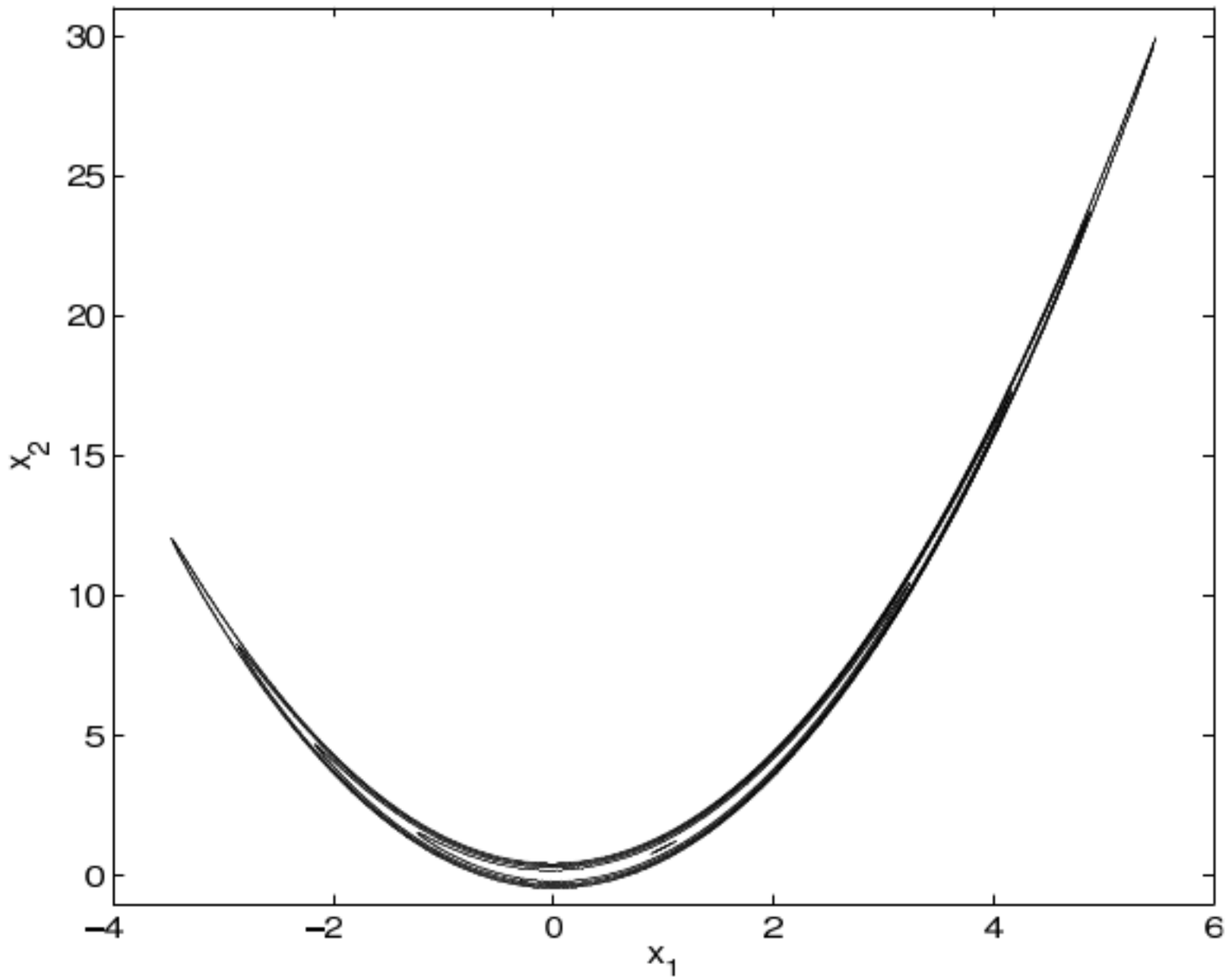
- We always accept proposals for which $p(\theta^*|\mathbf{x}) \geq p(\theta^{t-1}|\mathbf{x})$
- We sometimes accept proposals for which $p(\theta^*|\mathbf{x}) < p(\theta^{t-1}|\mathbf{x})$
- So the MCMC does not always go uphill, and is thus does not get stuck in local minima (MCMC techniques are useful for optimization)

Metropolis MCMC Sampling Visual

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

Challenge for M-H sampler: Rosenbrock

$$p(X) \propto \exp\left(-\frac{100(X_2 - X_1^2)^2 + (1 - X_1)^2}{20}\right)$$

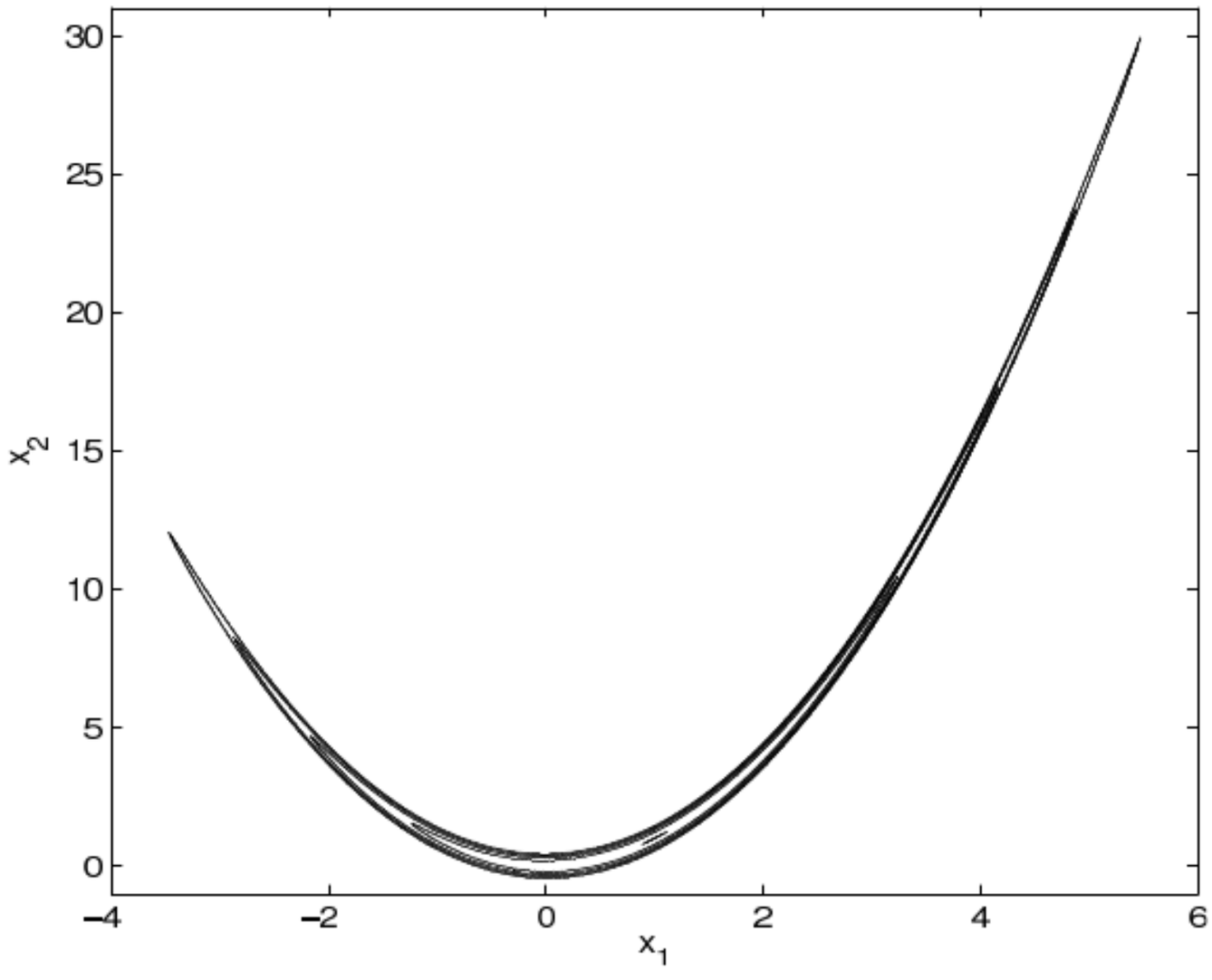


Challenge for M-H sampler: Rosenbrock

M-H sampler is inefficient

Proposal distribution is Gaussian (for example)

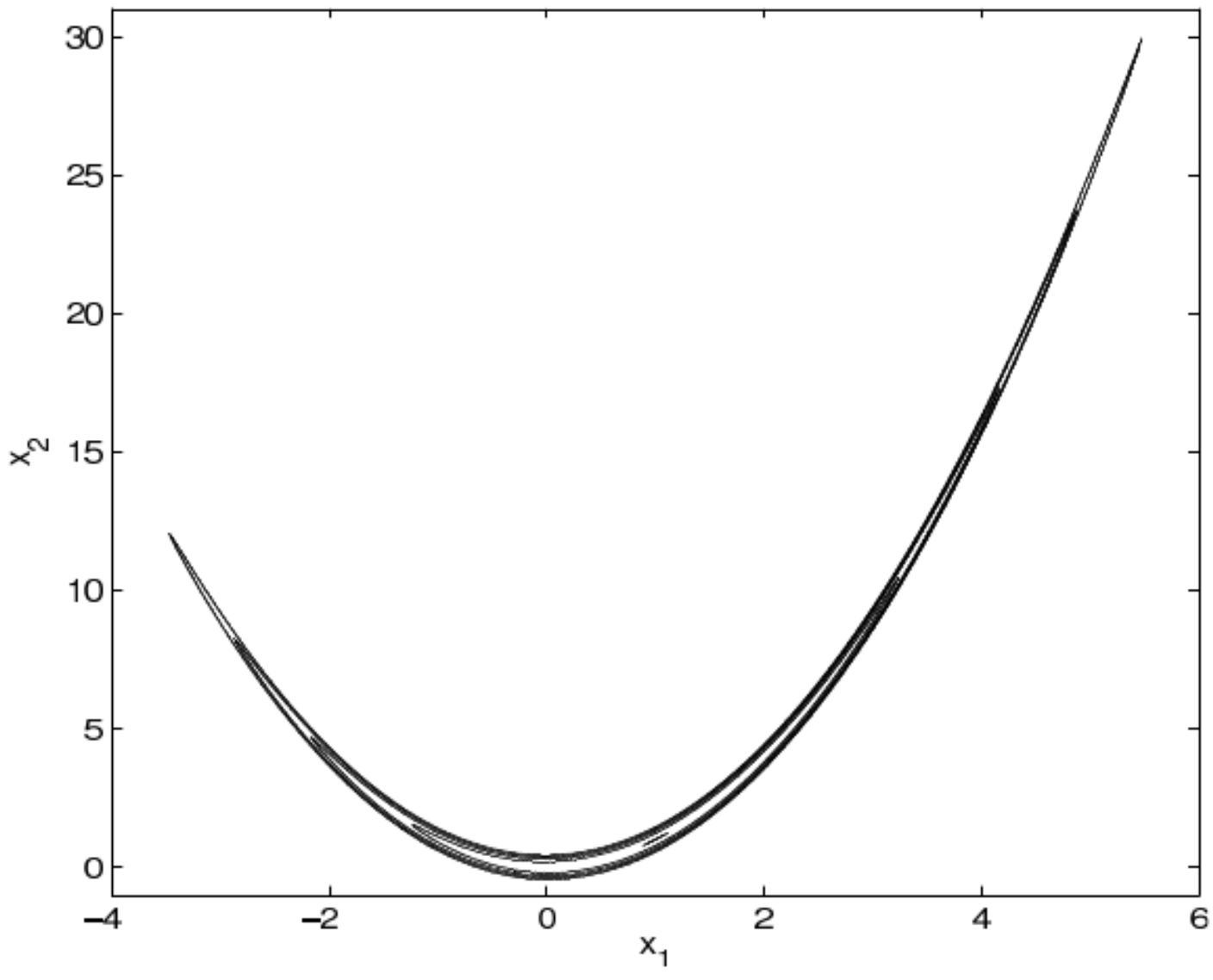
Sampling distribution is clearly non-Gaussian



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

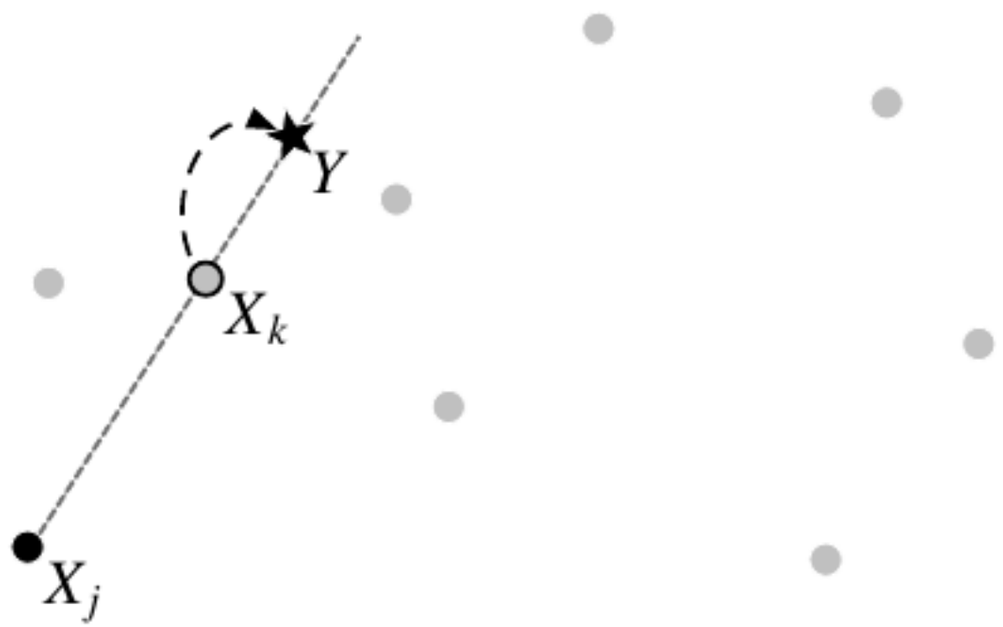
Affine Invariant MCMC Sampling

- **Affine transformation** is a linear mapping method that preserves points, straight lines, and planes.
- Sets of parallel lines remain parallel after an **affine transformation**.
- The **affine transformation** technique is typically used to correct for geometric distortions.



Goodman & Weare (2010

Affine Invariant MCMC Sampling

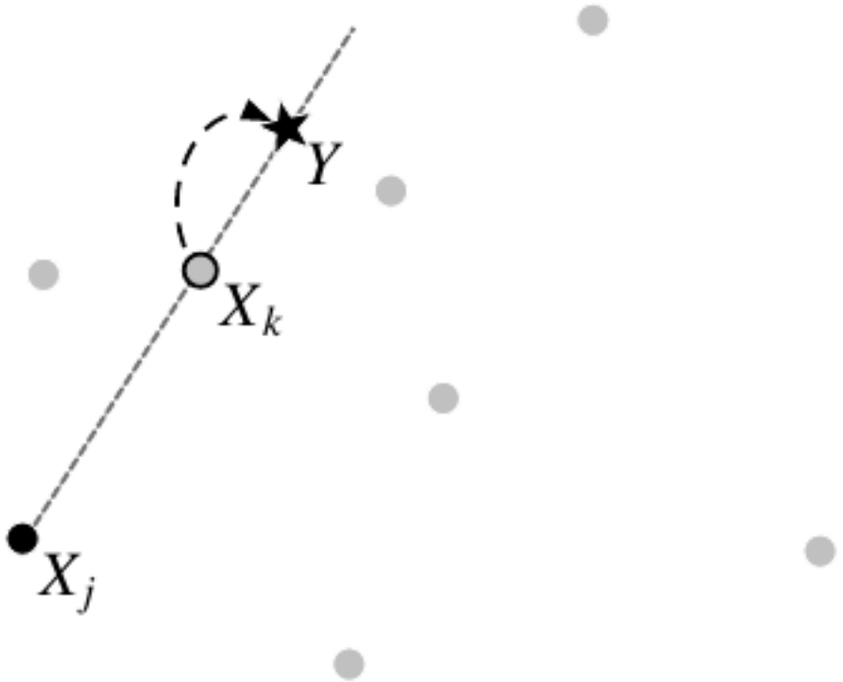


- Ensemble of “walkers”
- Use information from all walkers to choose next step
- Each walker gets a partner
- A walker’s next step is a “stretch move” along a line of its partner
- Markov properties satisfied
- Very little computational overhead

Goodman & Weare (2010

Affine Invariant MCMC Sampling

Pure python implementation: *emcee*



http://dfm.io/emcee/current/

https://www.youtube.com/watch?v=yow7Ol88DRk