

---

**data analysis** with  
**Markov chain Monte Carlo**

---

**Dan Foreman-Mackey**  
CCPP@NYU



# Dan Foreman-Mackey

- 
- dan.iel.fm
  - dfm
  - @exoplaneteer

I'm a grad student  
in Camp Hogg  
at NYU



David W. Hogg

I'm an engineer  
in Camp Hogg  
at NYU



David W. Hogg

I build **tools**  
for **data analysis**  
**in astronomy**<sup>(mostly)</sup>



David W. Hogg



*it's hammer time!*

introducing **emcee** the MCMC Hammer

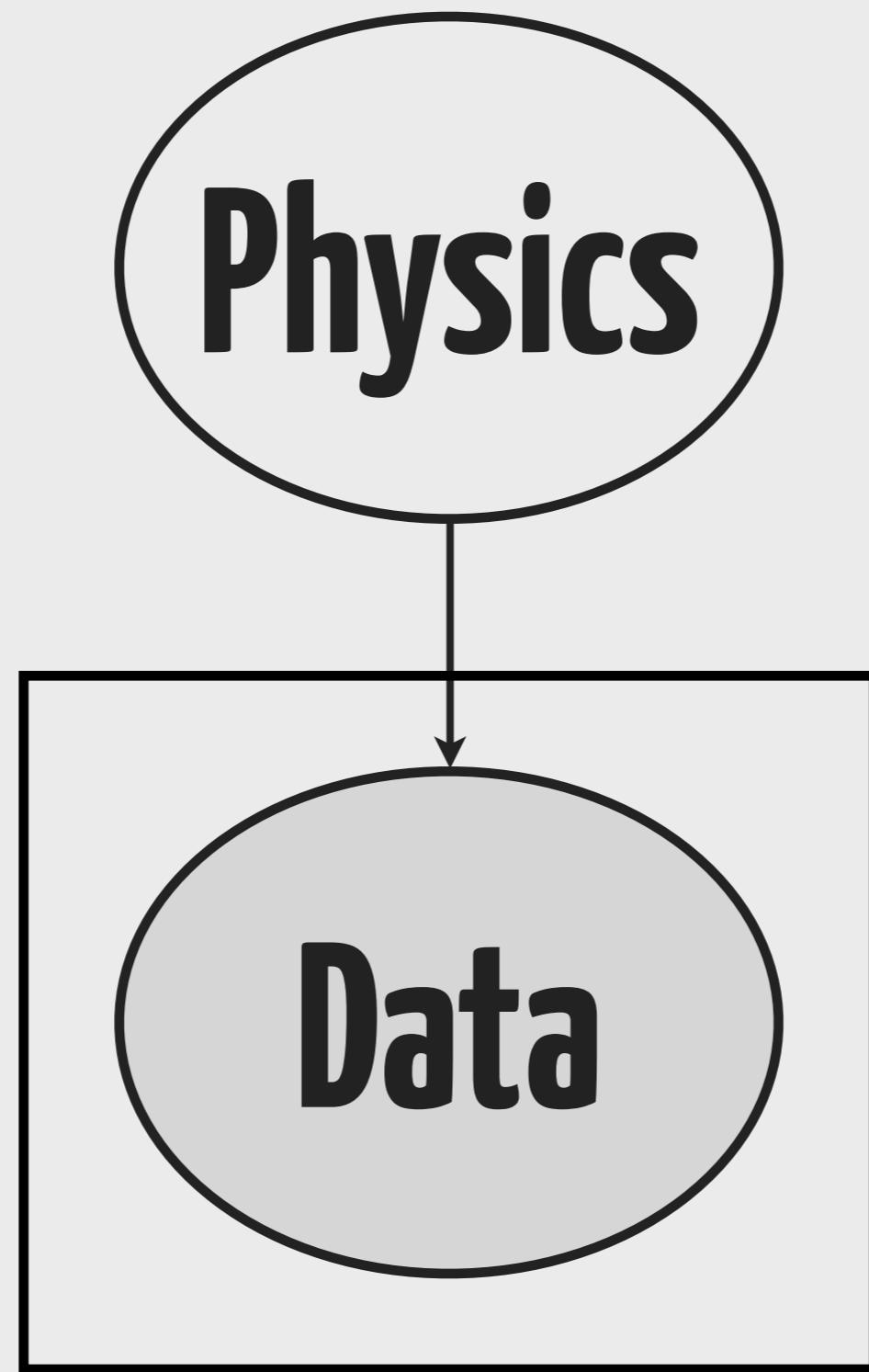
[arxiv.org/abs/1202.3665](https://arxiv.org/abs/1202.3665)

[dan.iel.fm/emcee](http://dan.iel.fm/emcee)

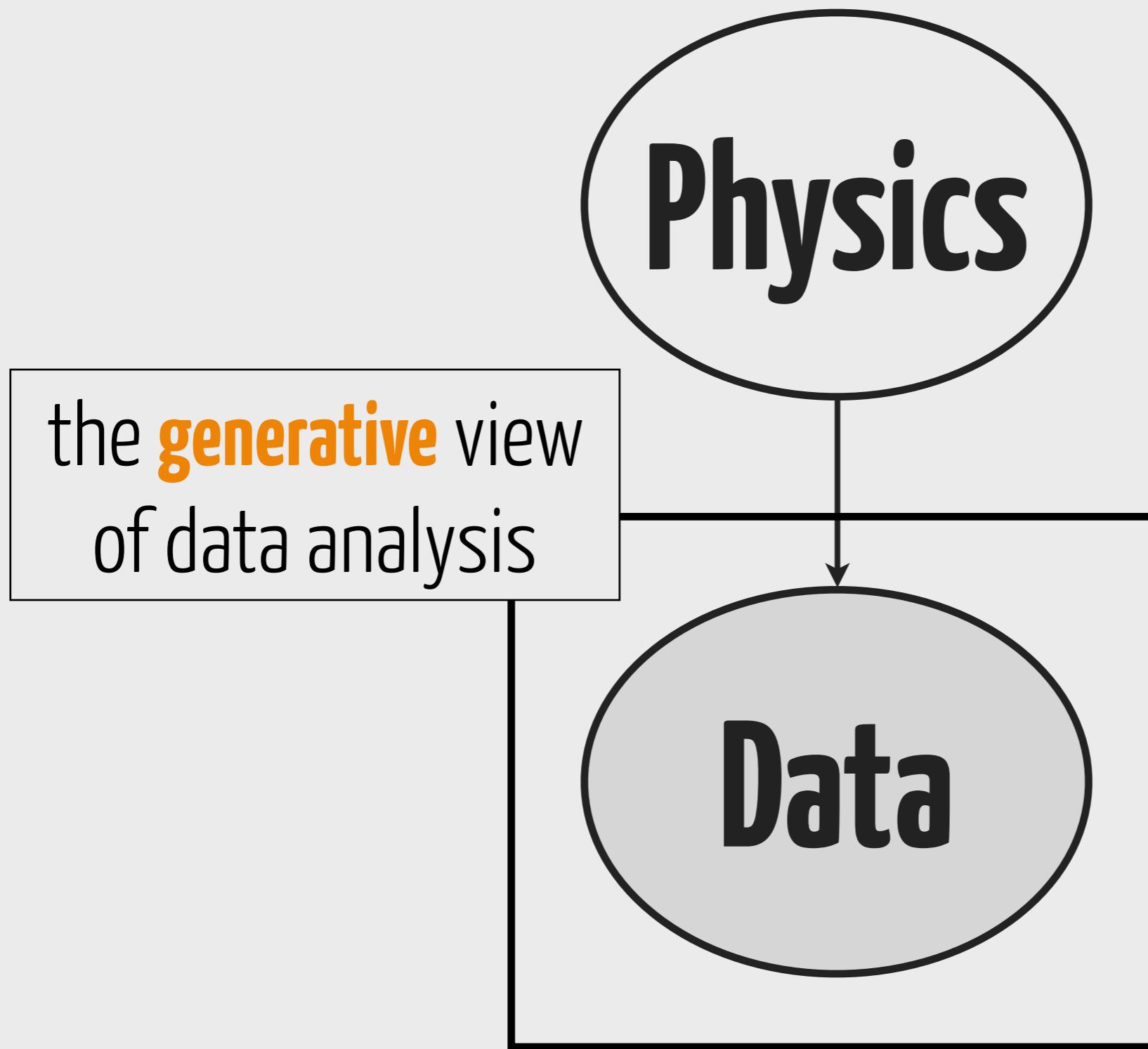
I work on the Local Group  
exoplanets  
variable stars  
image modeling  
calibration

I work on the Local Group  
exoplanets  
variable stars  
image modeling  
calibration  
not writing papers

I work on the Local Group  
exoplanets  
variable stars  
image modeling  
calibration  
not writing papers



a sketch of  
**The graphical model of my research.**



a sketch of  
**The graphical model of my research.**

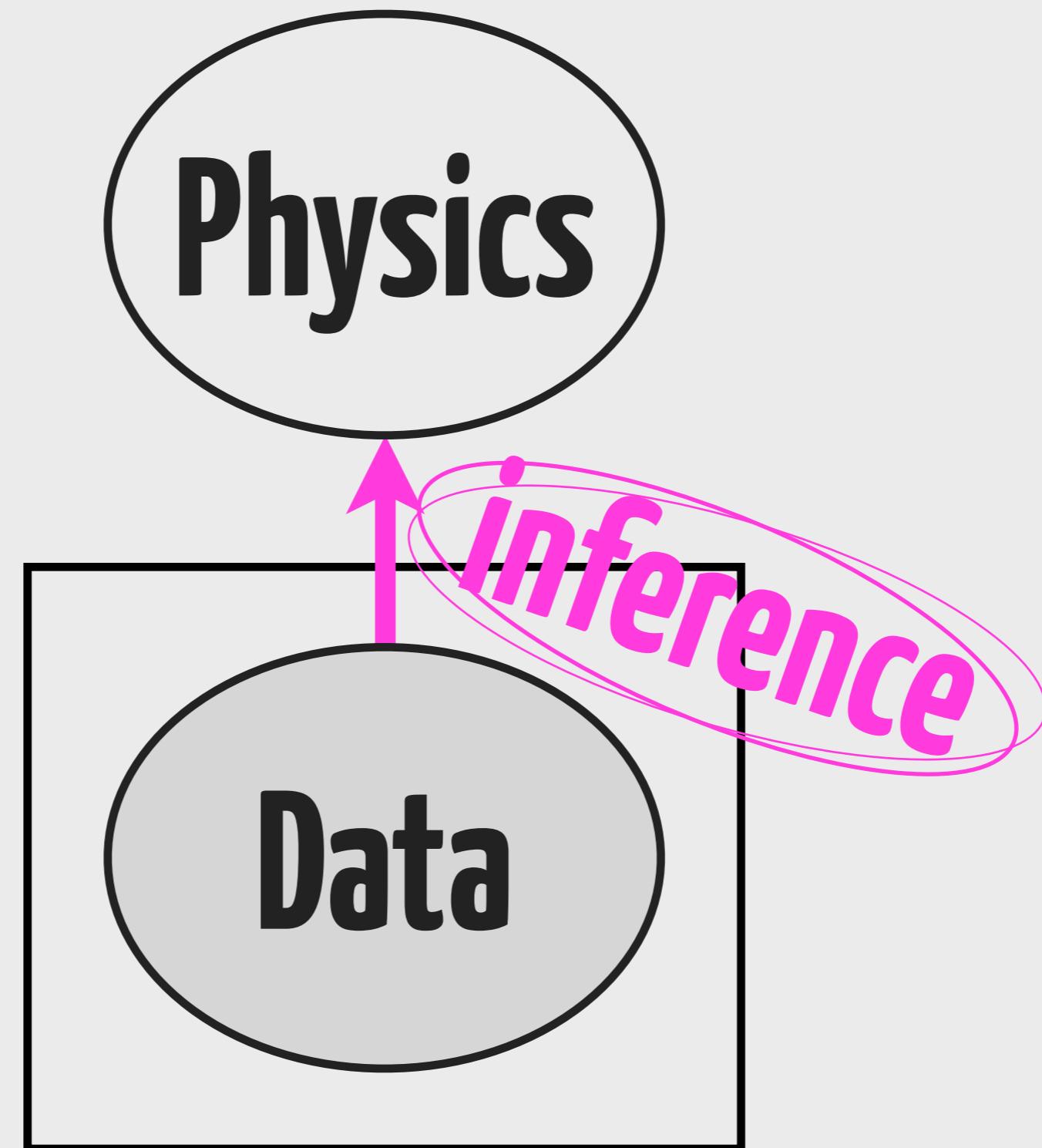
**for example**

$$\chi^2 = \sum_{n=1}^N \frac{[y_n - \overbrace{(m x_n + b)}^{\text{a line}}]^2}{\sigma_n^2}$$

$$p(\{y_n, x_n, \sigma_n^2\} | m, b) \propto \exp\left(-\frac{1}{2} \chi^2\right)$$

synthetic data:  $\hat{y}_n = m x_n + b + \epsilon_n$

noise model:  $\epsilon_n \sim \mathcal{N}(0; \sigma_n^2)$



a sketch of  
**The graphical model of my research.**

$$p(\text{data} \mid \text{physics})$$

likelihood function/generative model



$$p(\text{physics} \mid \text{data}) \propto p(\text{physics}) p(\text{data} \mid \text{physics})$$

posterior probability

good/bad news:

but physics  
is everything.

including things we don't care about!

# a more realistic model:

$$p(\mathbf{D} | \theta, \alpha)$$

cool physics

↓

↑ data

↑ garbage

# what if we underestimated our error bars?

(by some additive variance)

$$p(\{y_n, x_n, \sigma_n^2\} | \underbrace{m, b}_{\text{"physics"}}, \delta^2) =$$

"jitter"  
(not physics)

$$\prod_{n=1}^N \frac{1}{\sqrt{2\pi(\sigma_n^2 + \delta^2)}} \exp\left[-\frac{1}{2} \frac{[y_n - (m x_n + b)]^2}{\sigma_n^2 + \delta^2}\right]$$

# Do The Right Thing™

$$p(\mathbf{D} \mid \theta) \propto \int p(\alpha) p(\mathbf{D} \mid \theta, \alpha) d\alpha$$

marginalize.

**in our example**

$$p(\{y_n, x_n, \sigma_n\} | m, b) =$$

$$\int d\delta^2 p(\delta^2) \prod_{n=1}^N \frac{1}{\sqrt{2\pi(\sigma_n^2 + \delta^2)}} \exp\left[-\frac{1}{2} \frac{[y_n - (m x_n + b)]^2}{\sigma_n^2 + \delta^2}\right]$$

**BOOM!**

What do you **really** want?

What do you **really** want?

$$E_p[f(\theta)] = \frac{1}{Z} \int f(\theta) p(\theta) p(\mathbf{D} \mid \theta) d\theta$$

# What do you **really** want?

$$E_p[f(\theta)] = \frac{1}{Z} \int f(\theta) p(\theta) p(\mathbf{D} \mid \theta) d\theta$$

probably.

marginalization

$$p(\mathbf{D} \mid \theta) \propto \int p(\alpha) p(\mathbf{D} \mid \theta, \alpha) d\alpha$$

expectation

$$E_p[f(\theta)] = \frac{1}{Z} \int f(\theta) p(\theta) p(\mathbf{D} \mid \theta) d\theta$$

marginalization

$$p(\mathbf{D} \mid \theta) \propto \int p(\alpha) p(\mathbf{D} \mid \theta, \alpha) d\alpha$$

large number  
of dimensions

expectation

$$E_p[f(\theta)] = \frac{1}{Z} \int f(\theta) p(\theta) p(\mathbf{D} \mid \theta) d\theta$$

marginalization

$$p(\mathbf{D} \mid \theta) \propto \int p(\alpha) p(\mathbf{D} \mid \theta, \alpha) d\alpha$$

expectation

$$E_p[f(\theta)] = \frac{1}{Z} \int f(\theta) p(\theta) p(\mathbf{D} \mid \theta) d\theta$$

whoa!

large number  
of dimensions

marginalization

$$p(\mathbf{D} \mid \theta) \propto \int p(\alpha) p(\mathbf{D} \mid \theta, \alpha) d\alpha$$

expectation

$$E_p[f(\theta)] = \frac{1}{Z} \int f(\theta) p(\theta) p(\mathbf{D} \mid \theta) d\theta$$

whoa!

large number  
of dimensions

This is **HARD.**  
(in general)

**OK**

**now that we agree...**

as you learned in middle school

$$\int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \approx \frac{1}{N} \sum_{n=1}^N f(\mathbf{x}_n)$$

where:  $\mathbf{x}_n \sim p(\mathbf{x})$

error:  $\delta \propto \frac{1}{\sqrt{N'}}$

number of  
**independent**  
samples

as you learned in middle school

$$\int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \approx \frac{1}{N} \sum_{n=1}^N f(\mathbf{x}_n)$$

where:  $\mathbf{x}_n \sim p(\mathbf{x})$

error:  $\delta \propto \frac{1}{\sqrt{N'}}$

number of  
**independent**  
samples

# MCMC

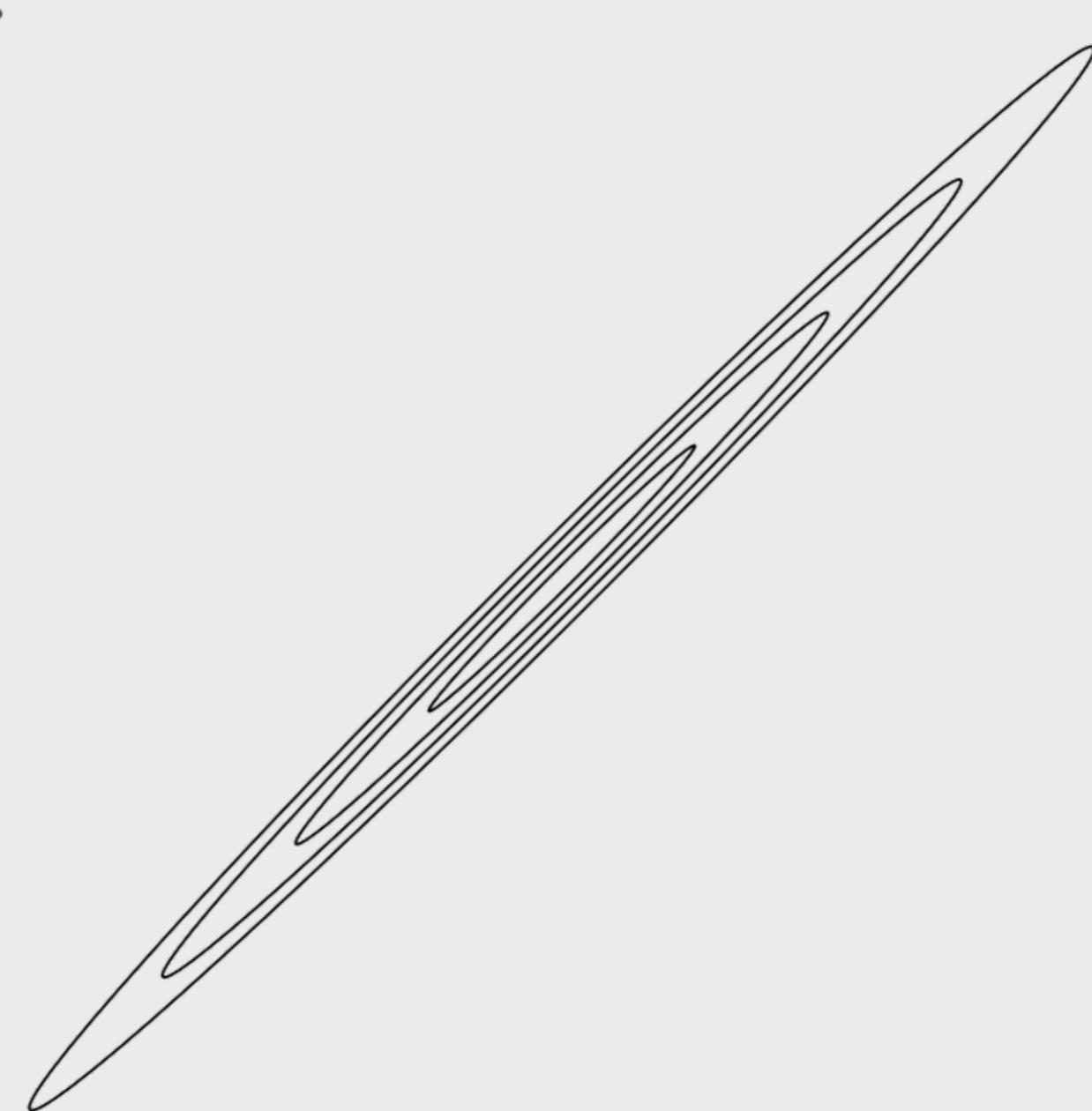
draws samples from a probability function

and all you need to be able to do is

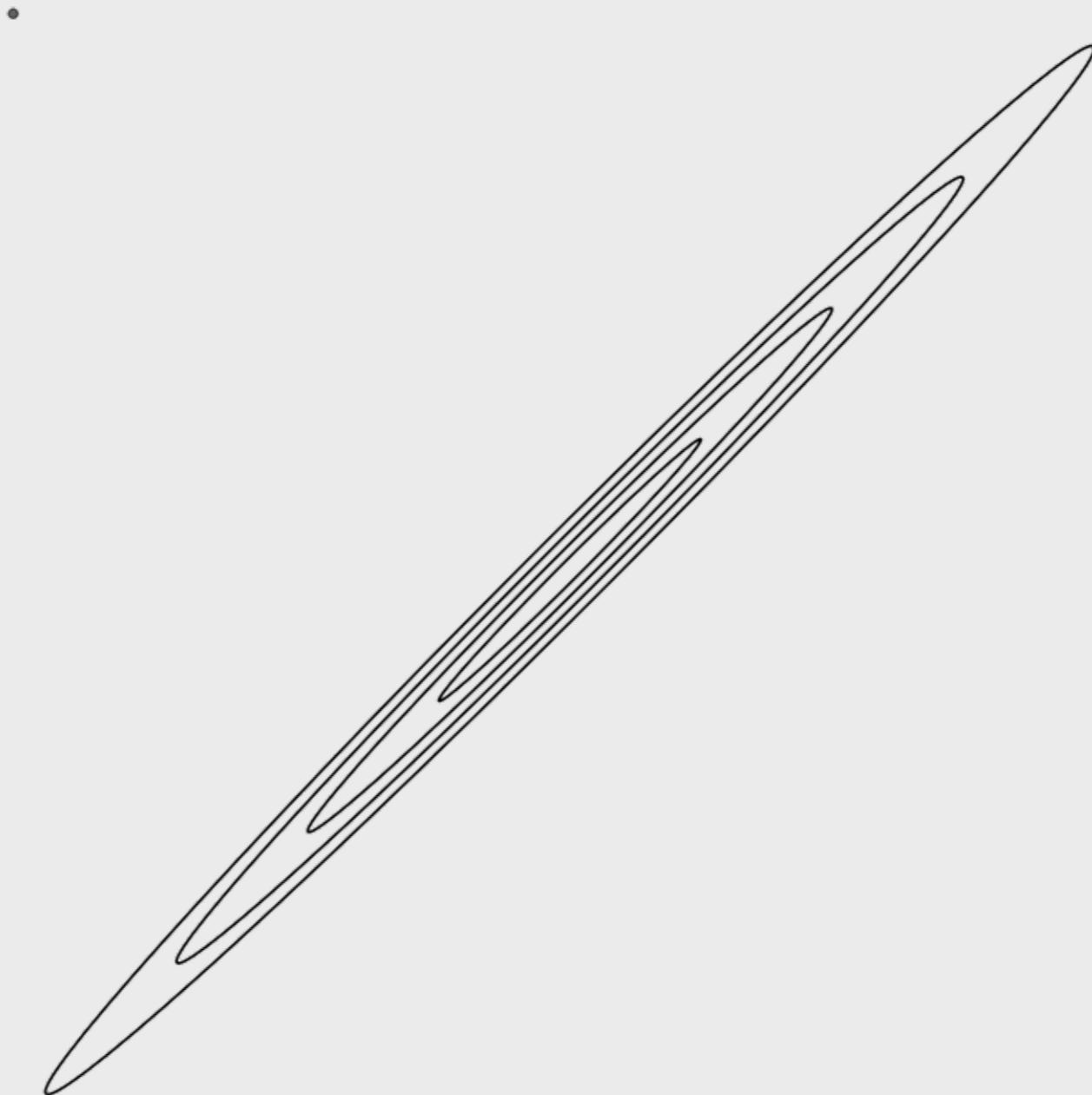
**evaluate**

the function

(up to a constant)

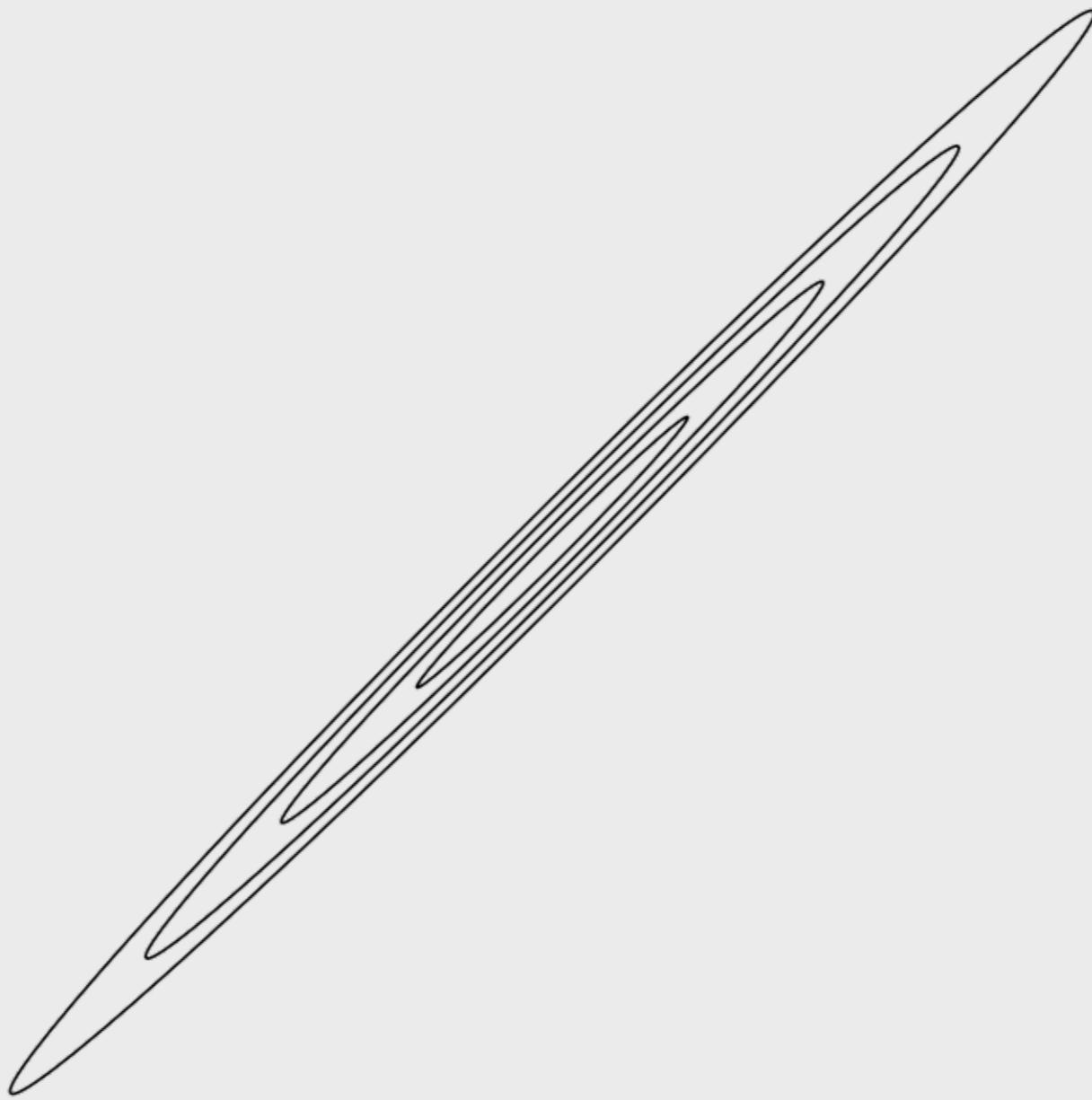


**Metropolis-Hastings**



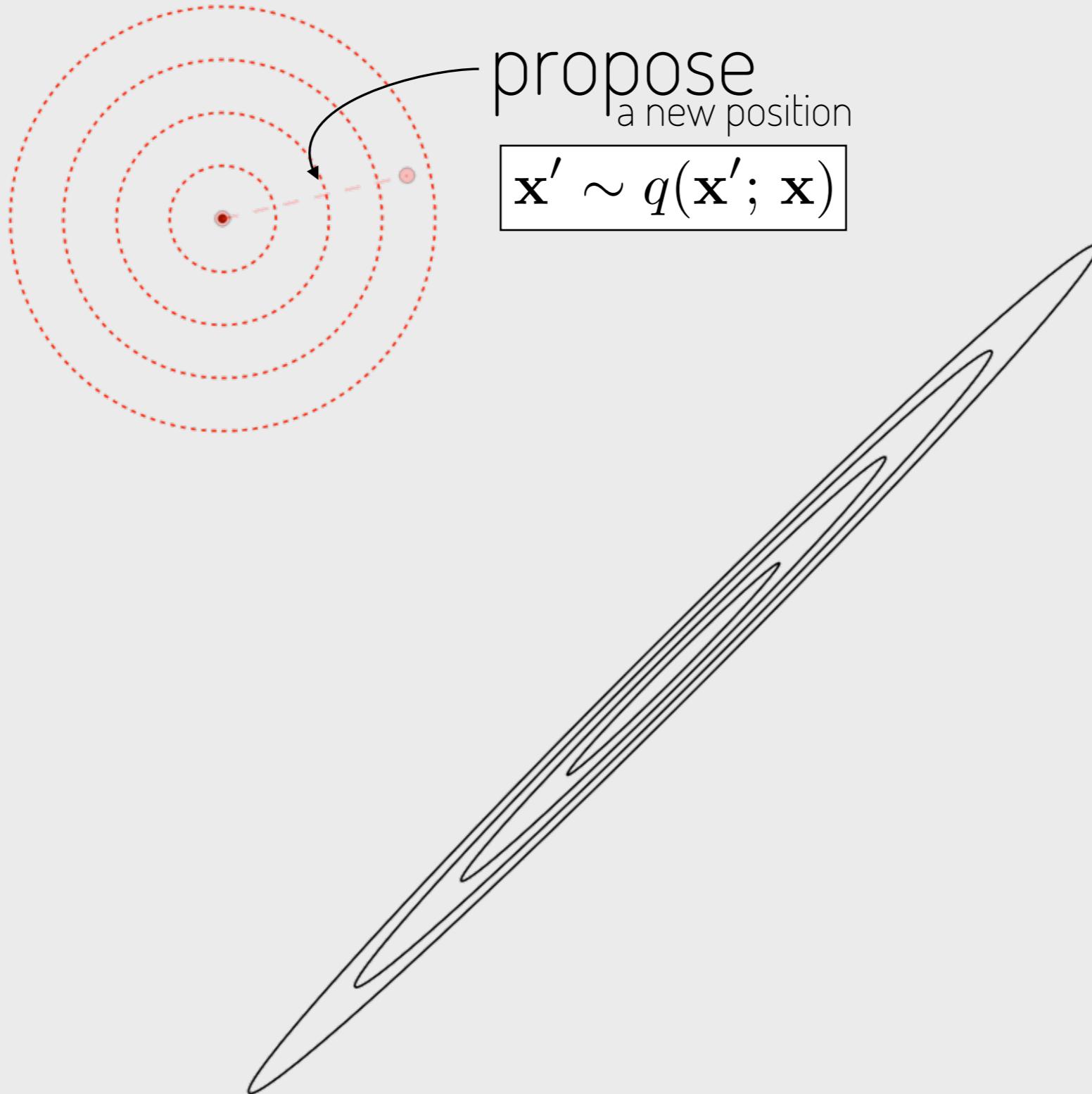
**Metropolis-Hastings**  
in an ideal world

start here  
perhaps



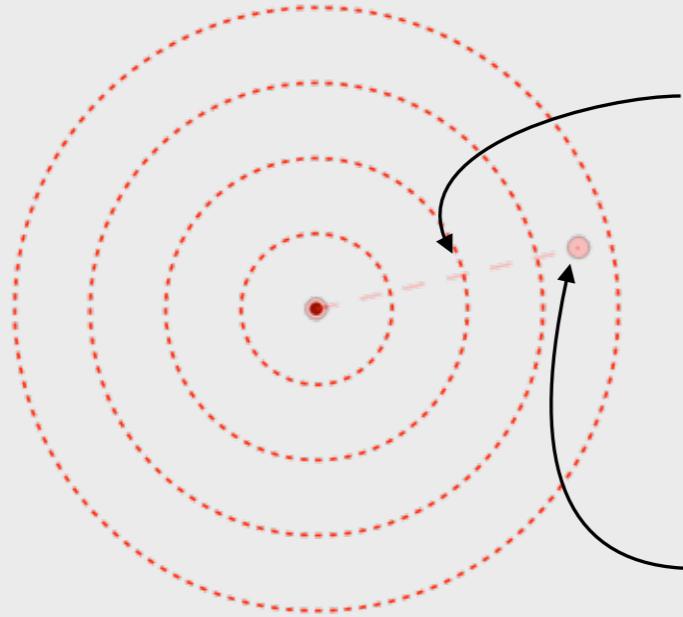
# Metropolis-Hastings

in an ideal world



# Metropolis-Hastings

in an ideal world

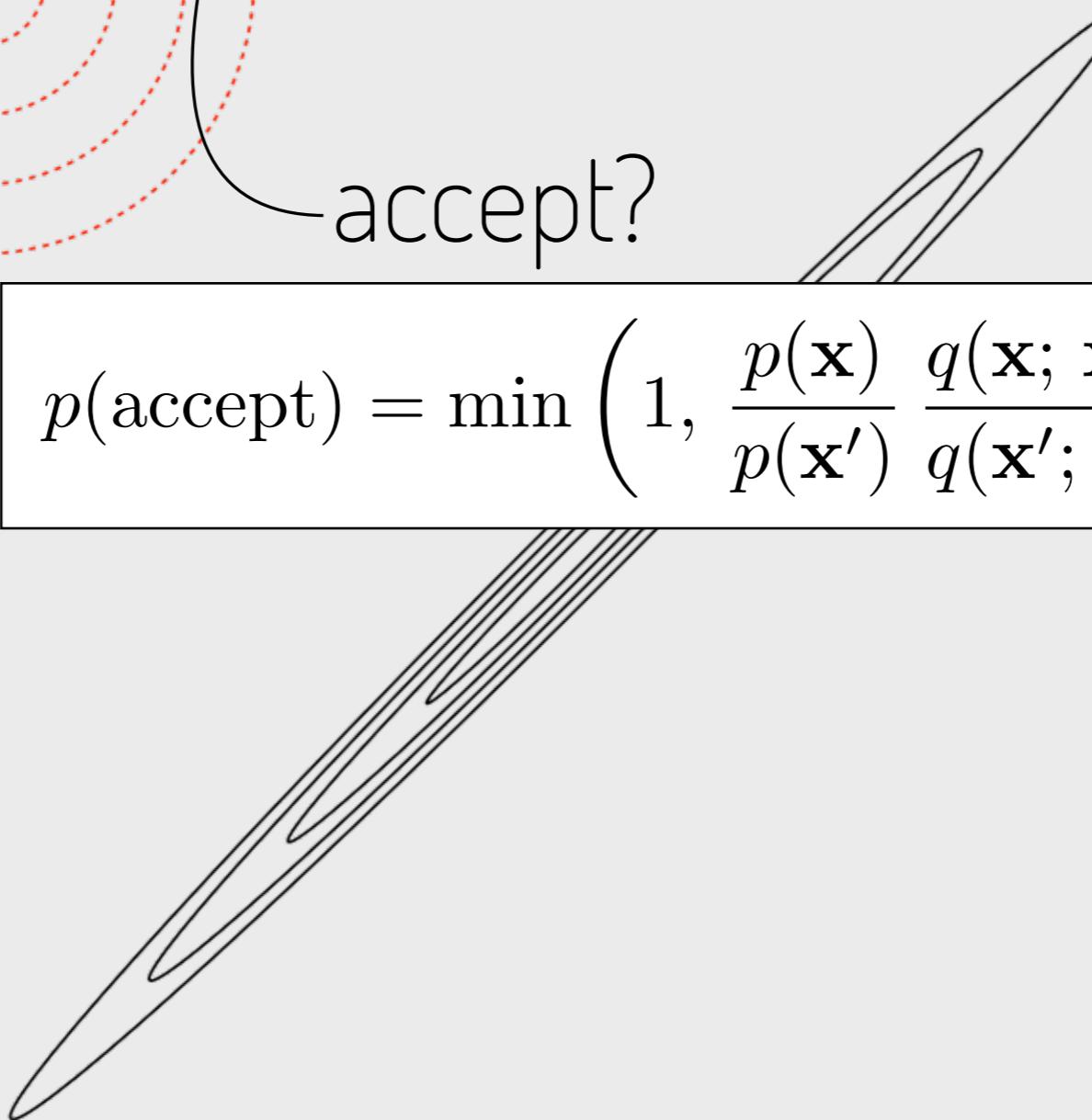


propose  
a new position

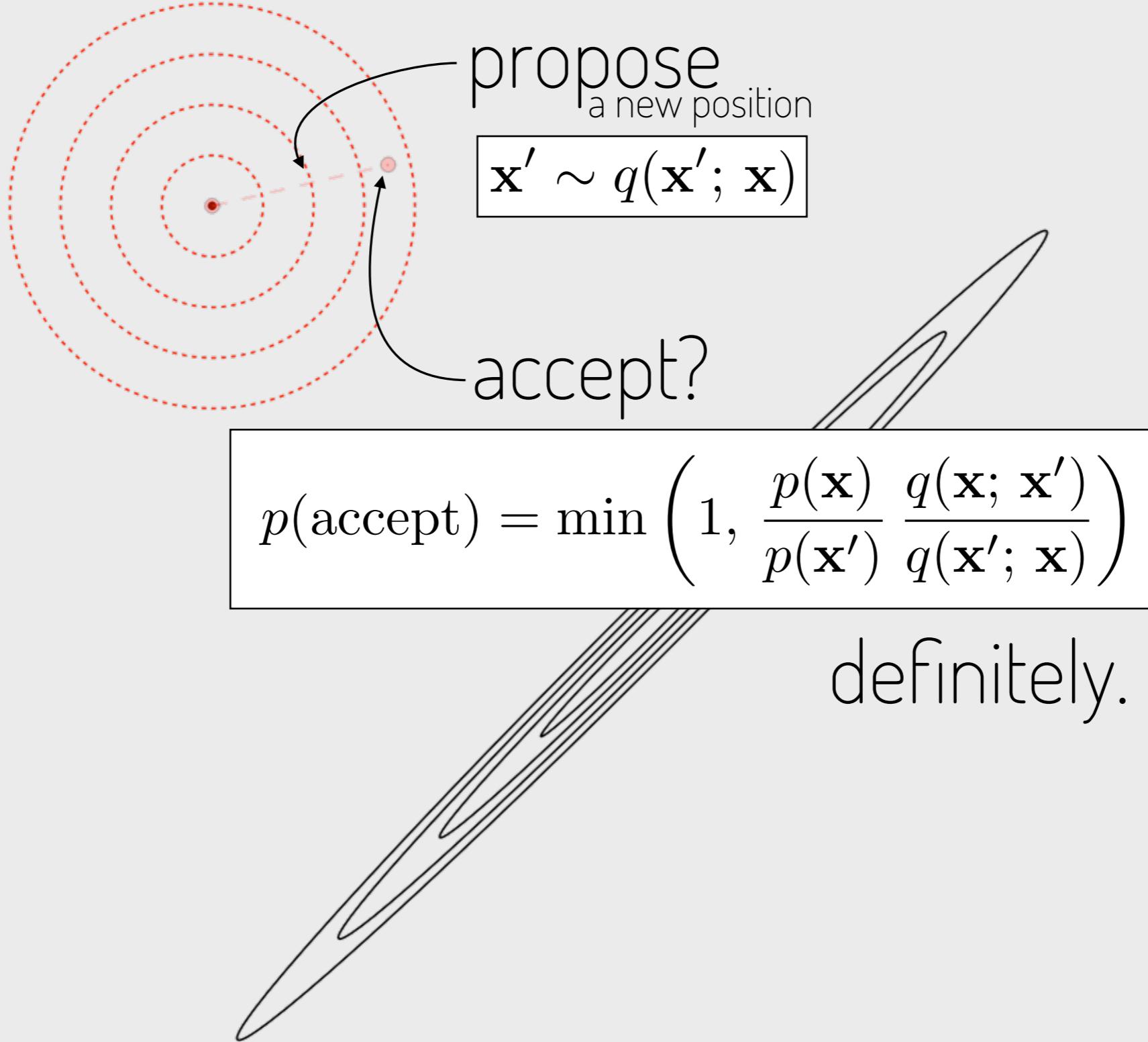
$$\mathbf{x}' \sim q(\mathbf{x}'; \mathbf{x})$$

accept?

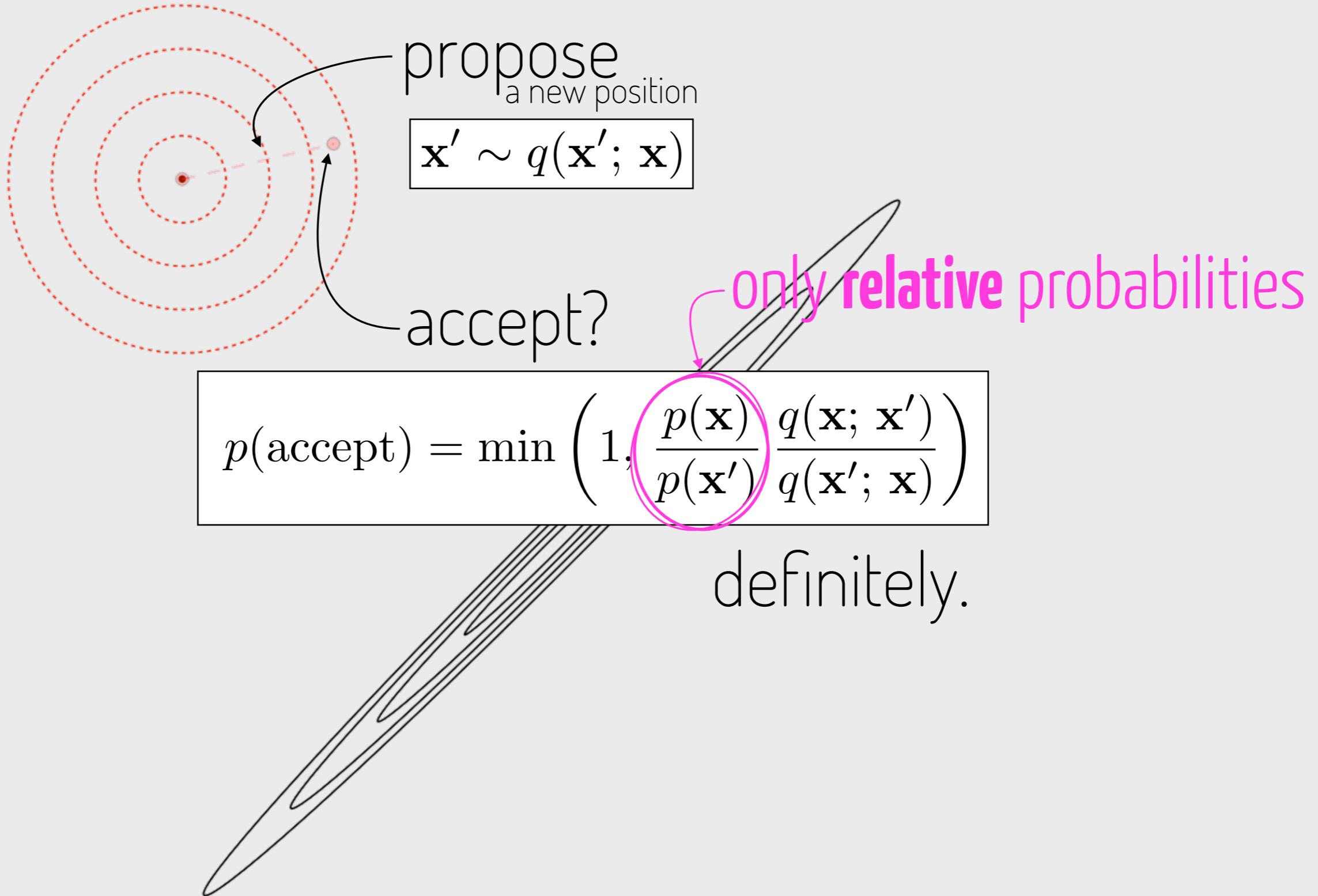
$$p(\text{accept}) = \min \left( 1, \frac{p(\mathbf{x})}{p(\mathbf{x}')}, \frac{q(\mathbf{x}; \mathbf{x}')}{q(\mathbf{x}'; \mathbf{x})} \right)$$



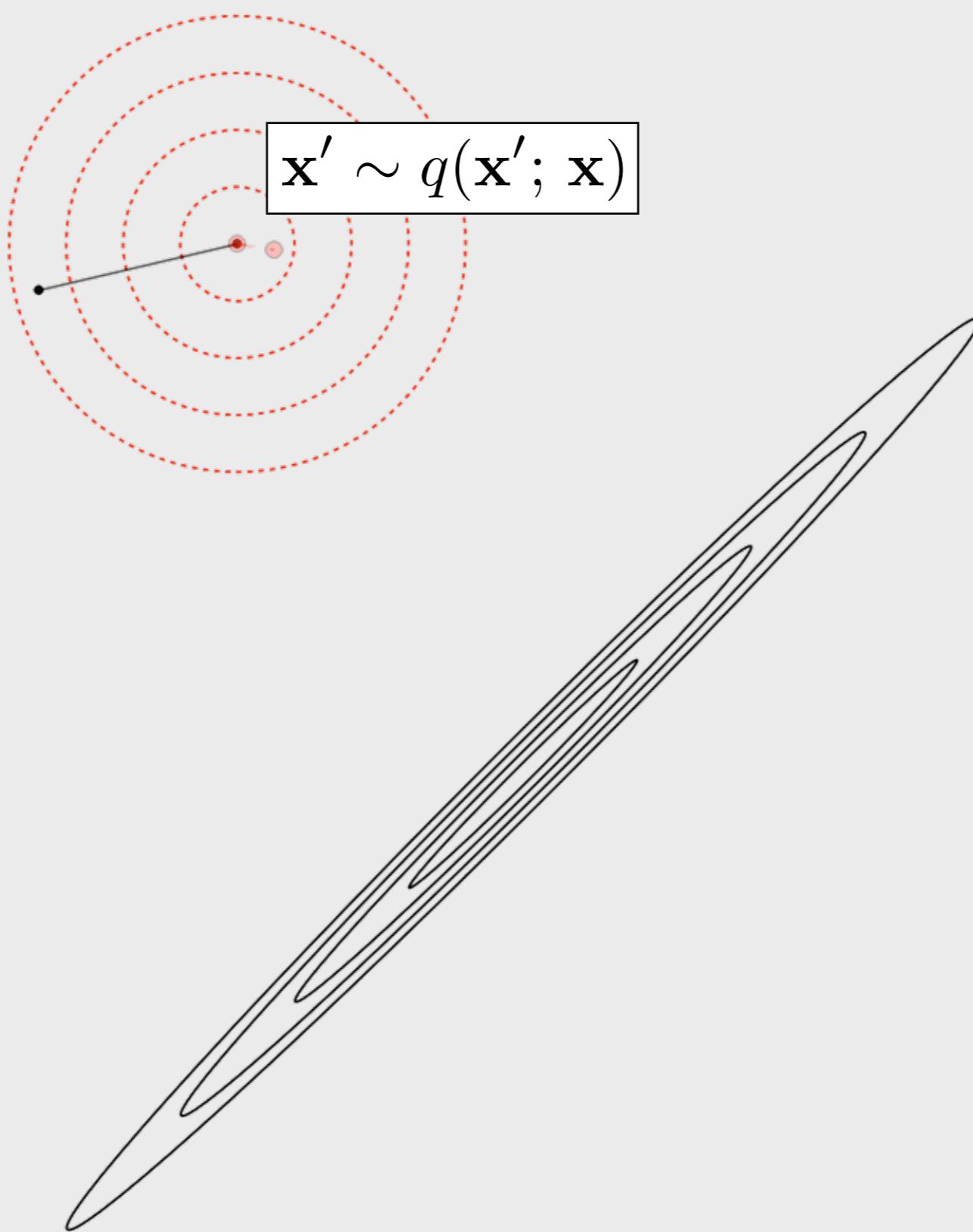
**Metropolis-Hastings**  
in an ideal world



**Metropolis-Hastings**  
in an ideal world

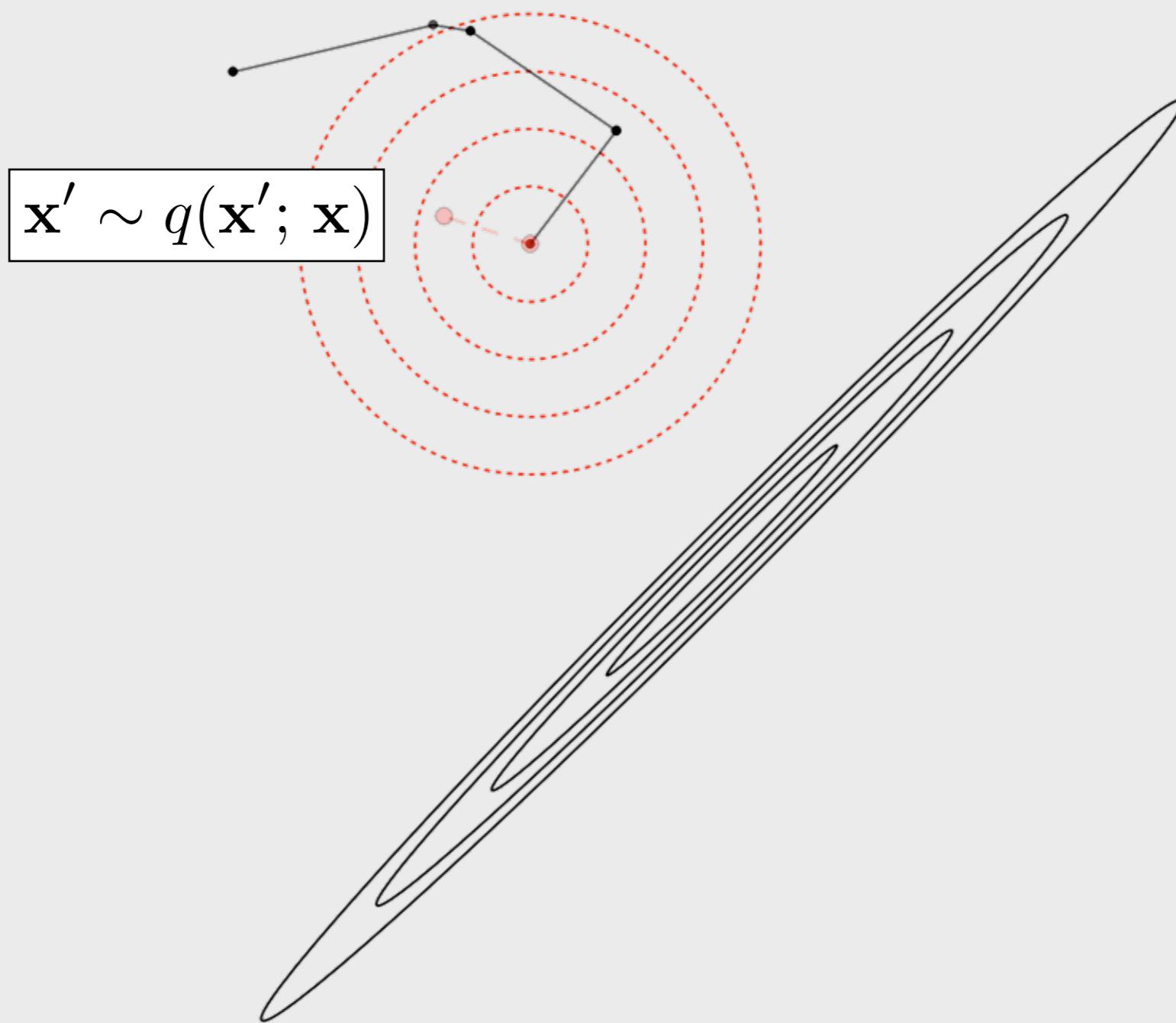


**Metropolis-Hastings**  
in an ideal world



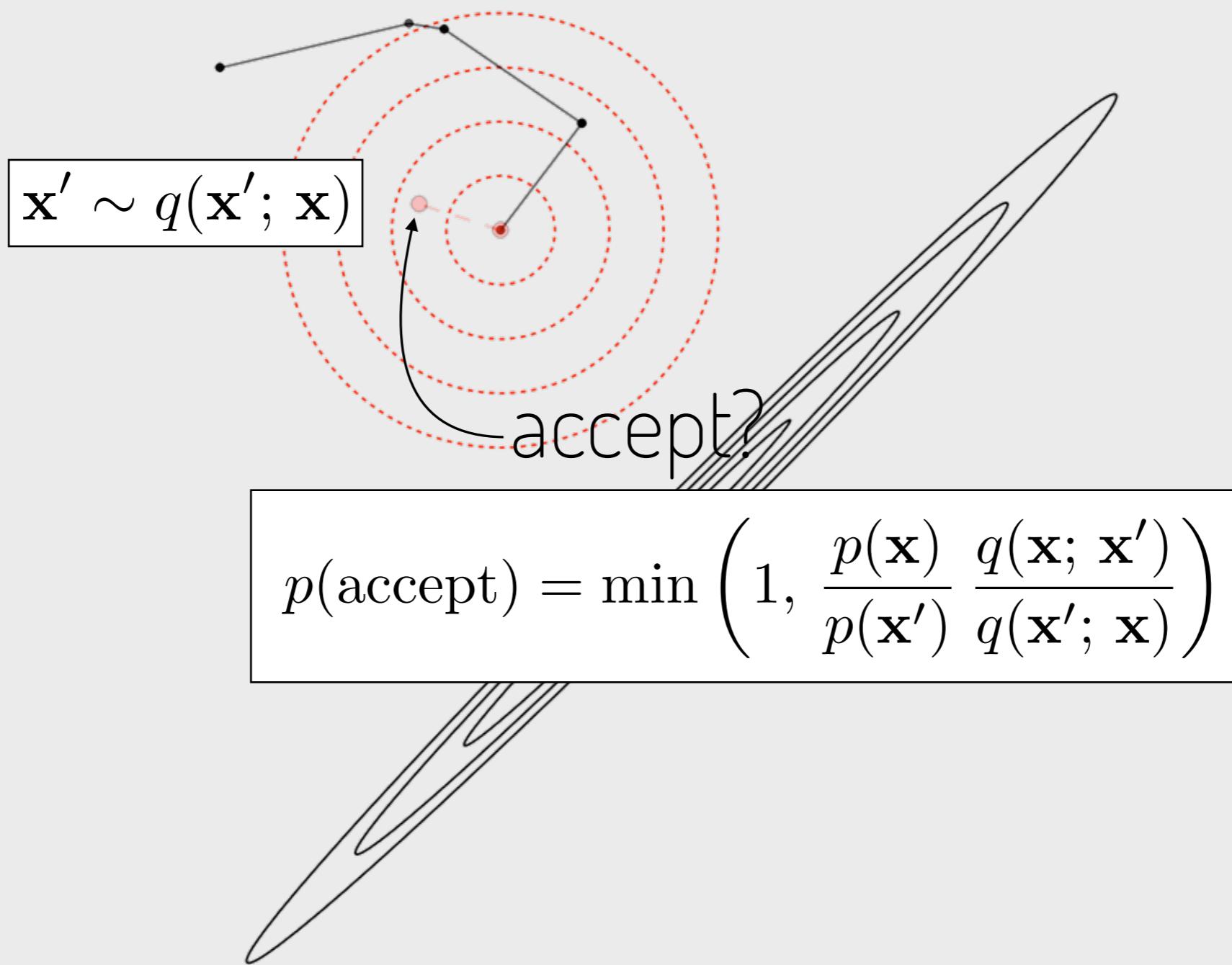
# Metropolis-Hastings

in an ideal world

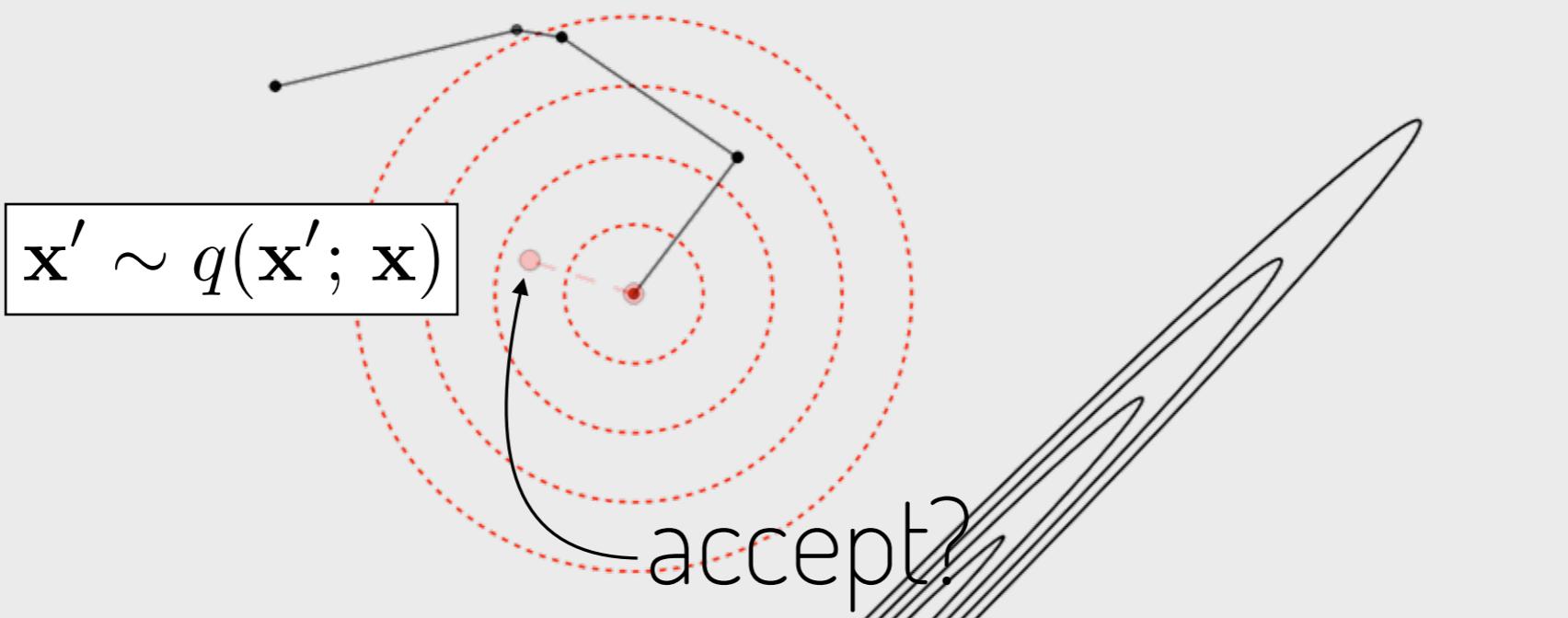


# Metropolis-Hastings

in an ideal world



**Metropolis-Hastings**  
in an ideal world

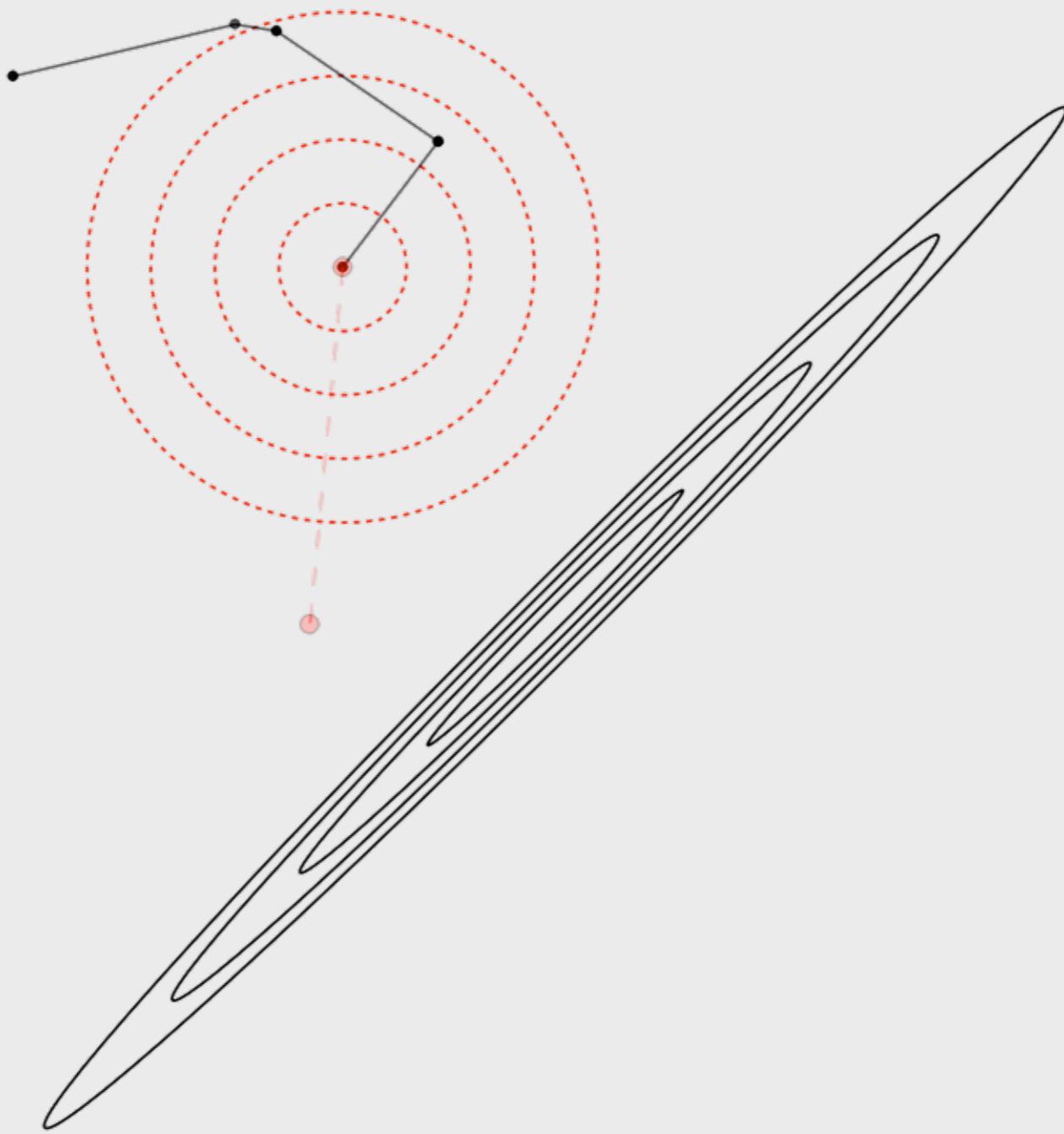


$$p(\text{accept}) = \min \left( 1, \frac{p(\mathbf{x})}{p(\mathbf{x}')}, \frac{q(\mathbf{x}; \mathbf{x}')}{q(\mathbf{x}'; \mathbf{x})} \right)$$

not this time.

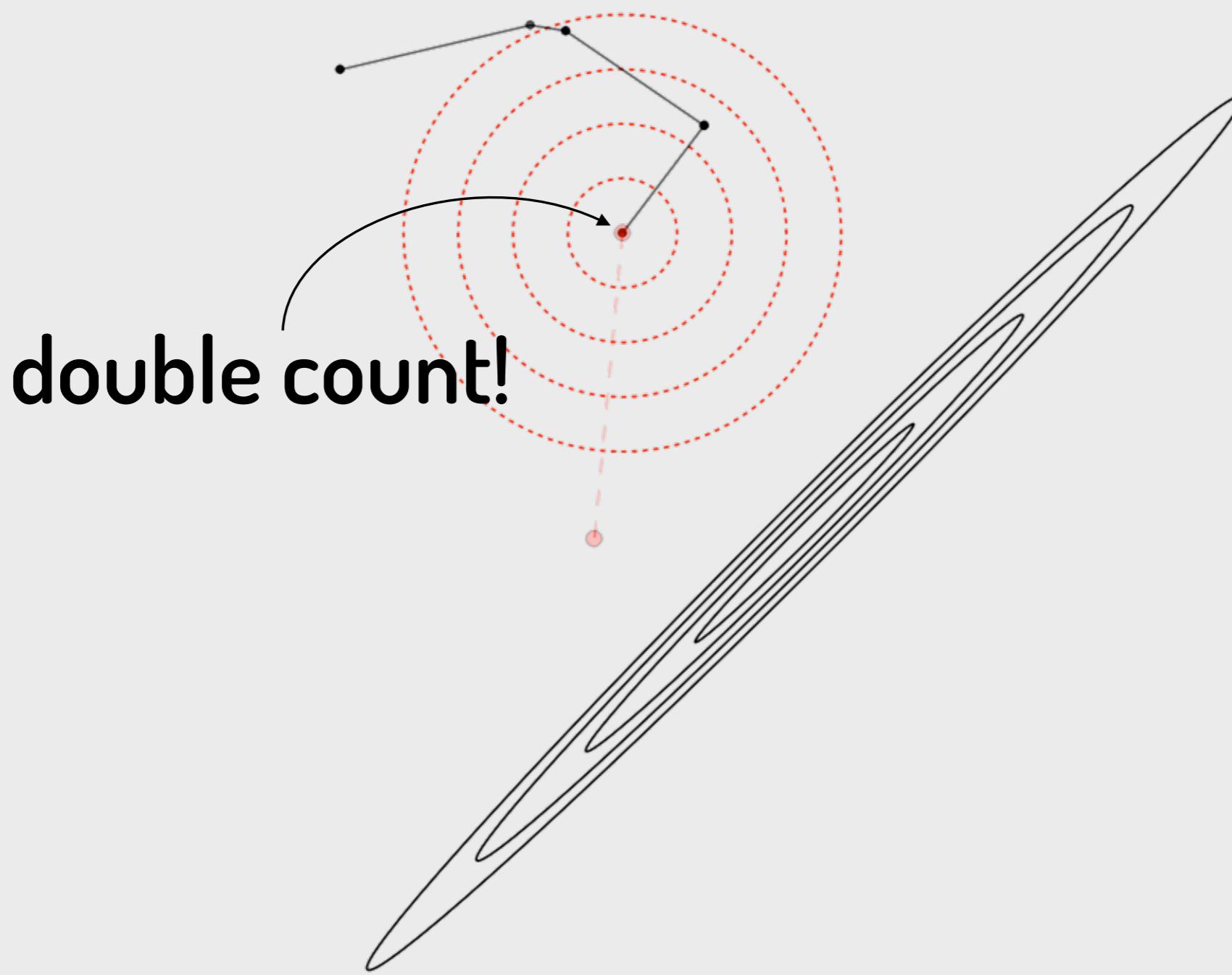
# Metropolis-Hastings

in an ideal world

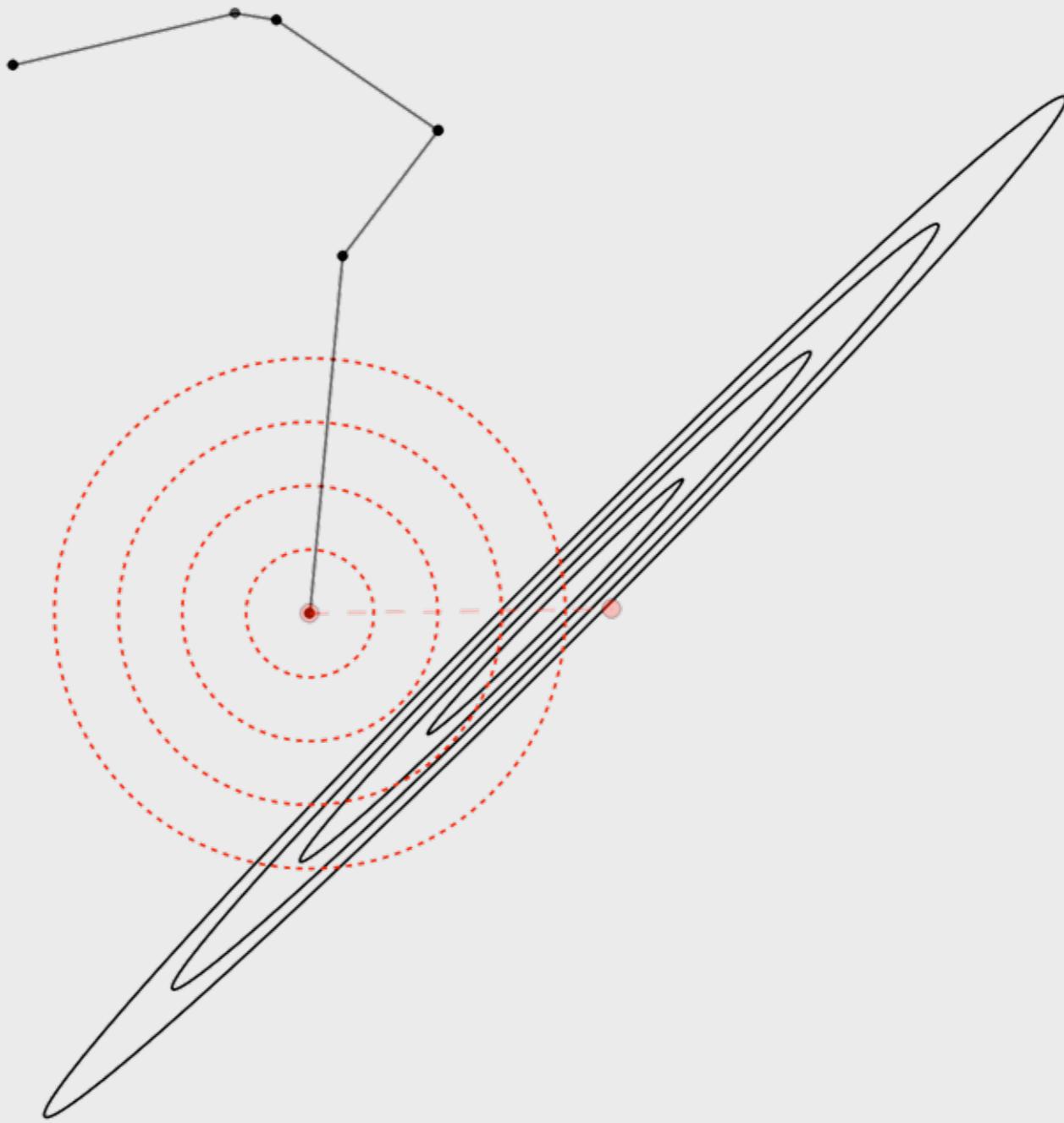


# Metropolis-Hastings

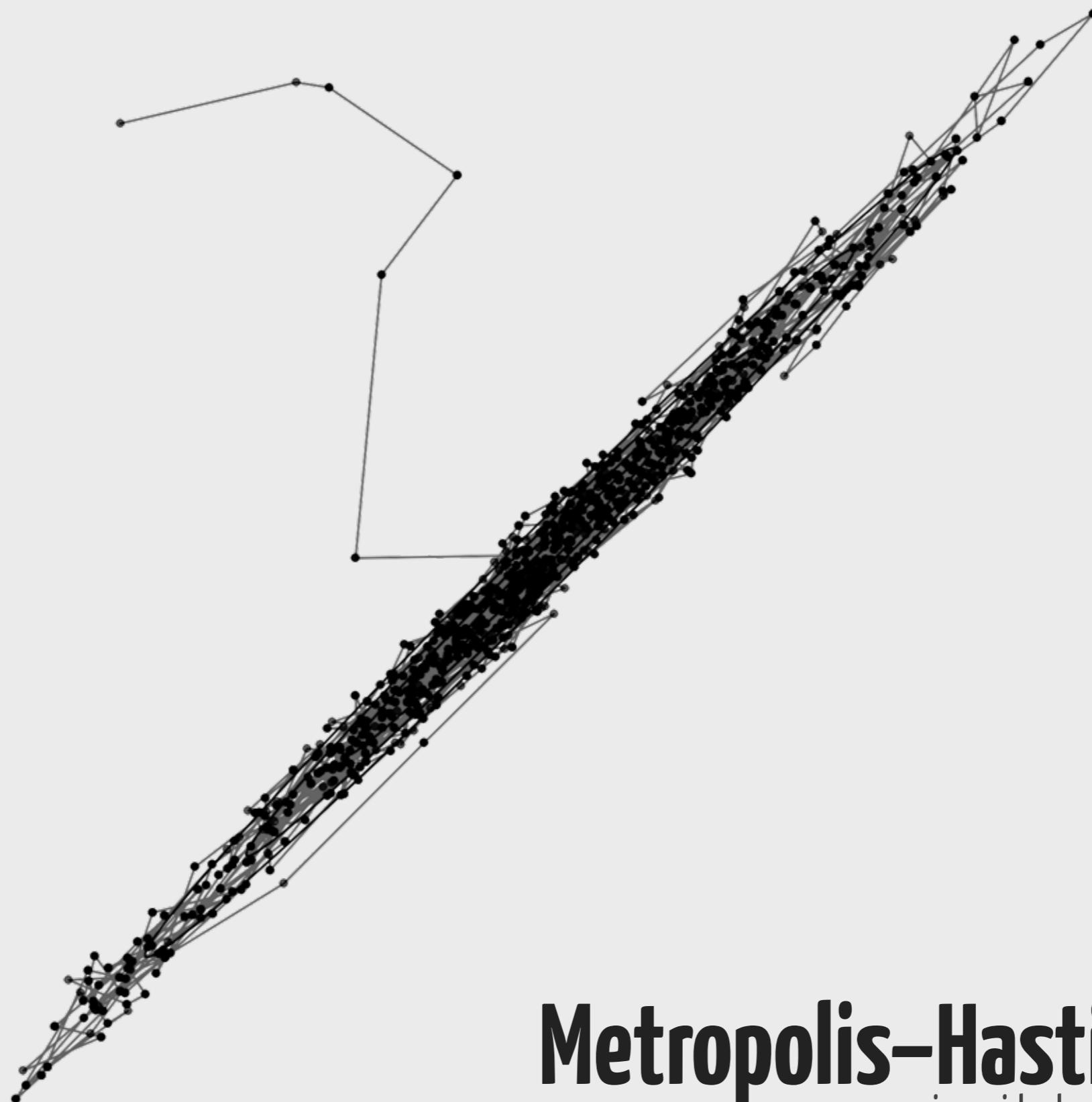
in an ideal world



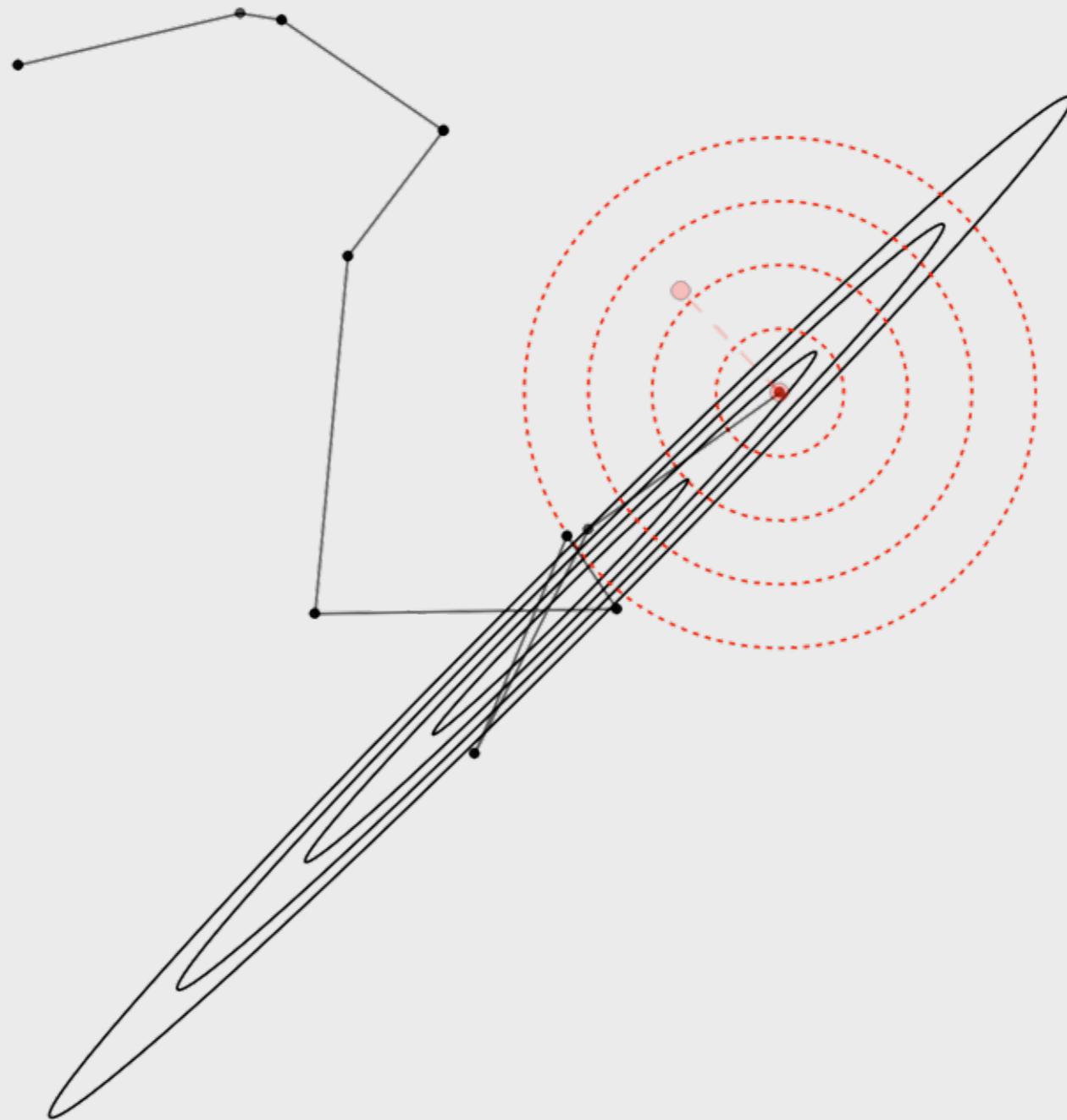
**Metropolis-Hastings**  
in an ideal world



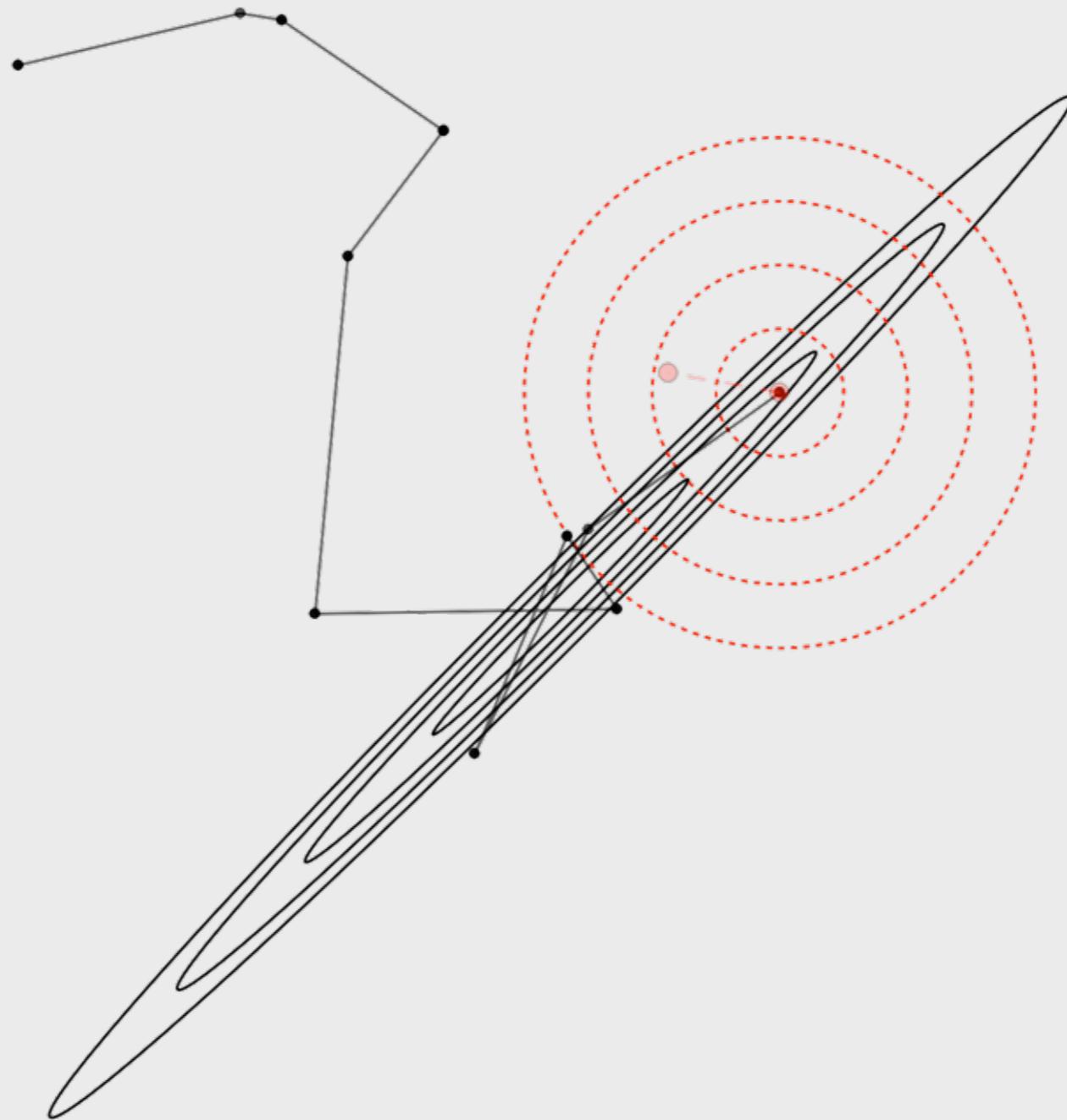
**Metropolis-Hastings**  
in an ideal world



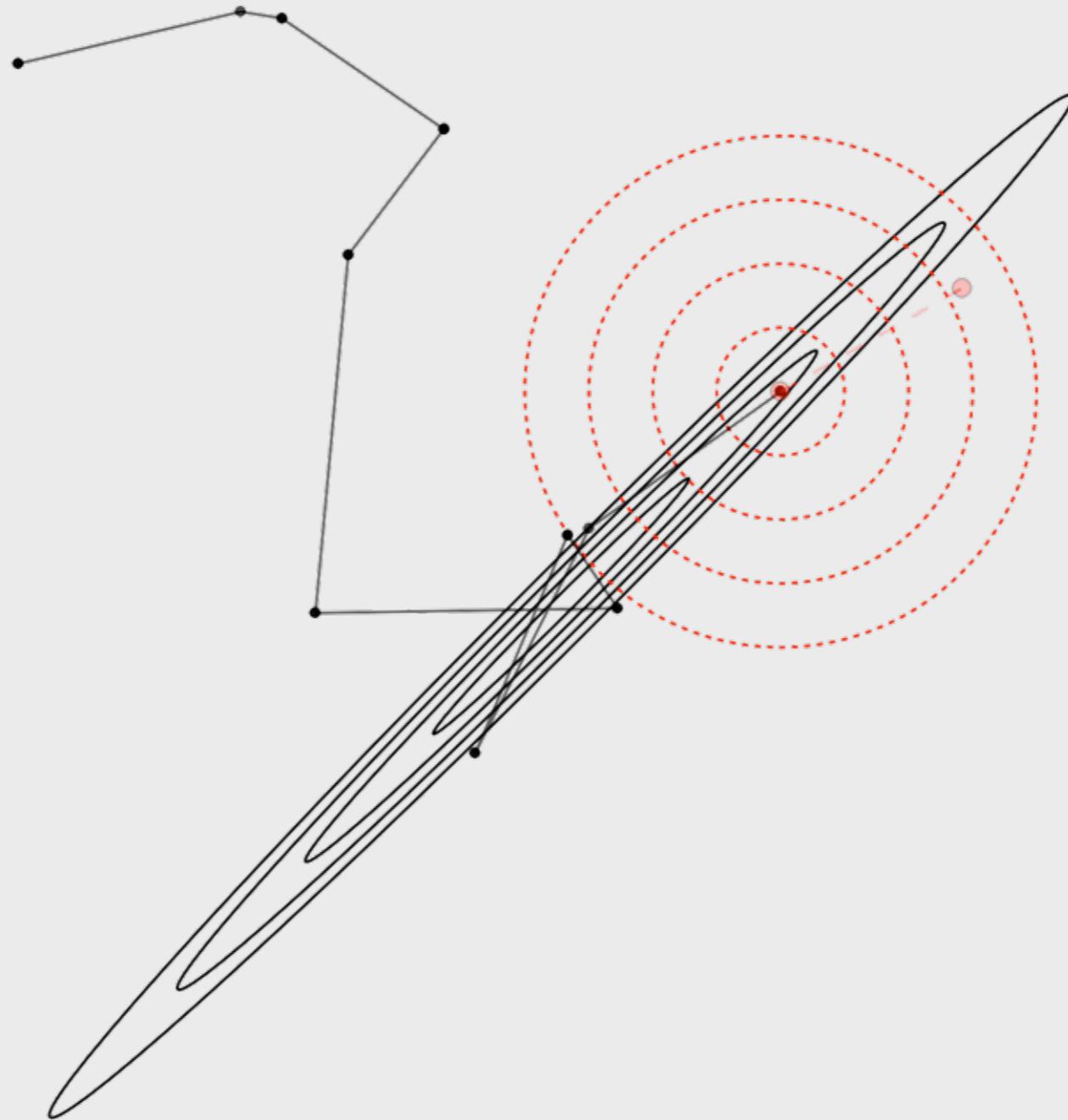
**Metropolis-Hastings**  
in an ideal world



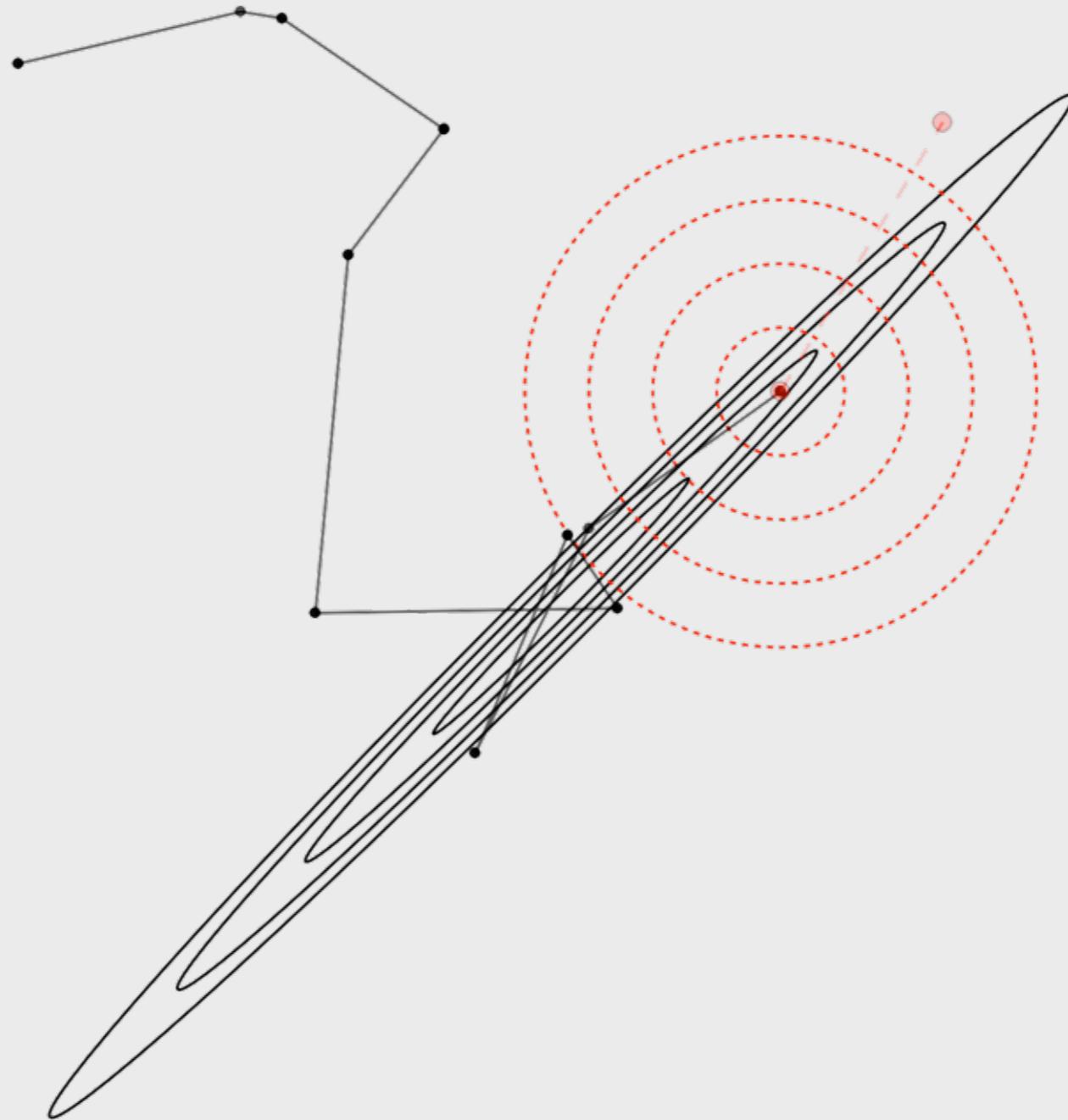
**Metropolis-Hastings**  
in the real world



**Metropolis-Hastings**  
in the real world

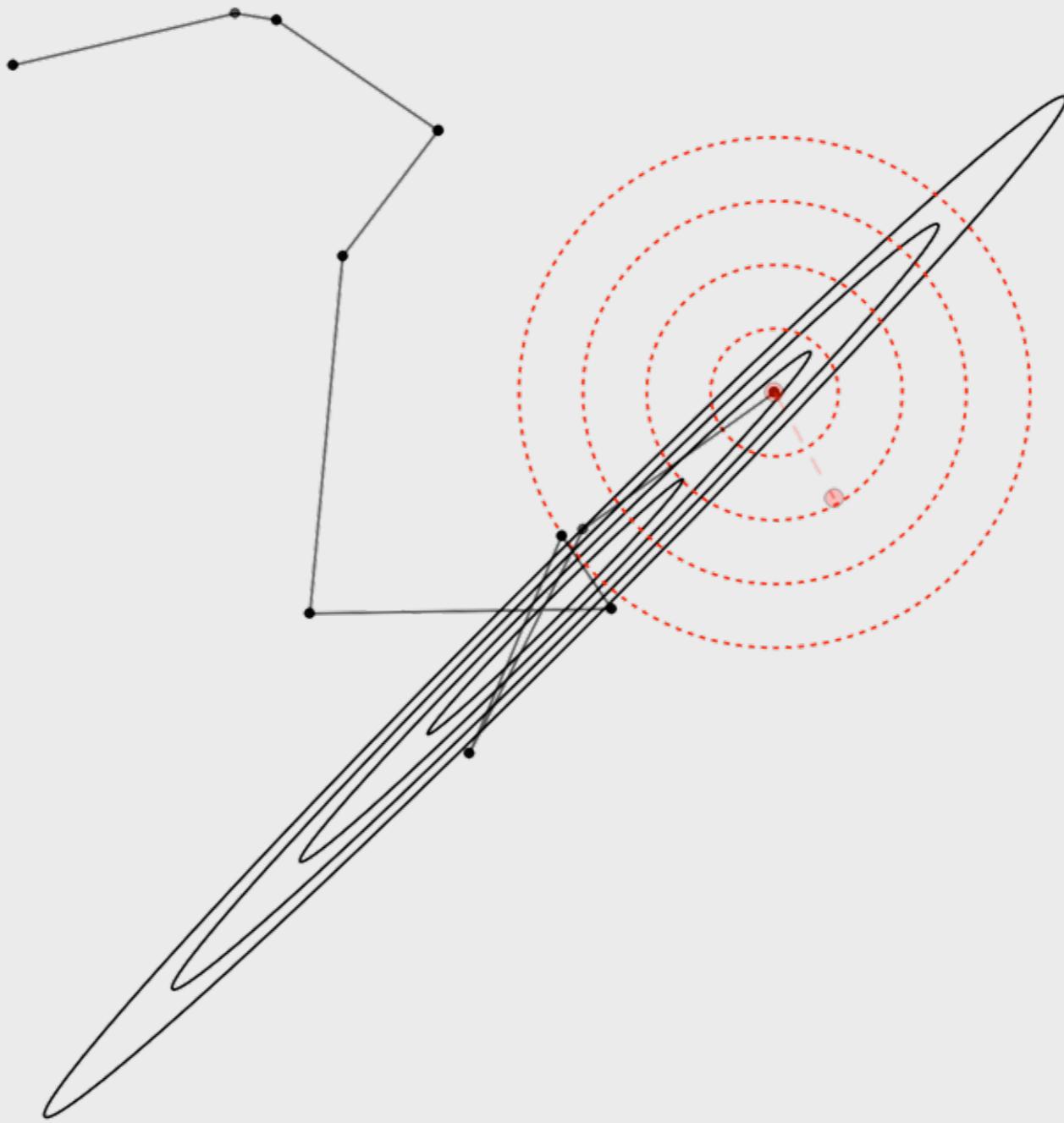


**Metropolis-Hastings**  
in the real world

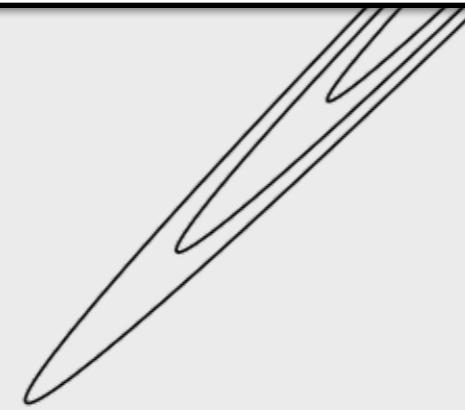
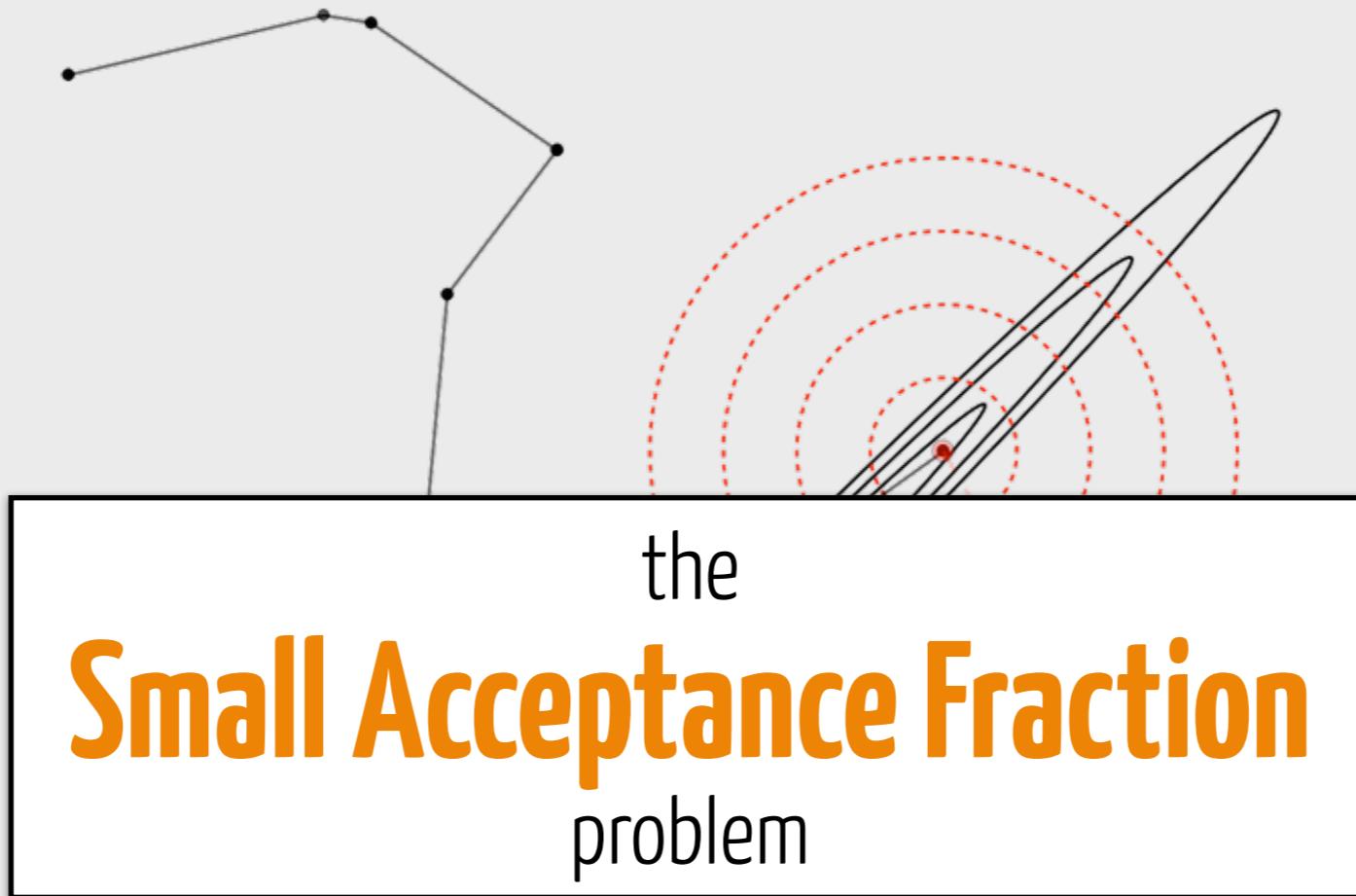


# Metropolis-Hastings

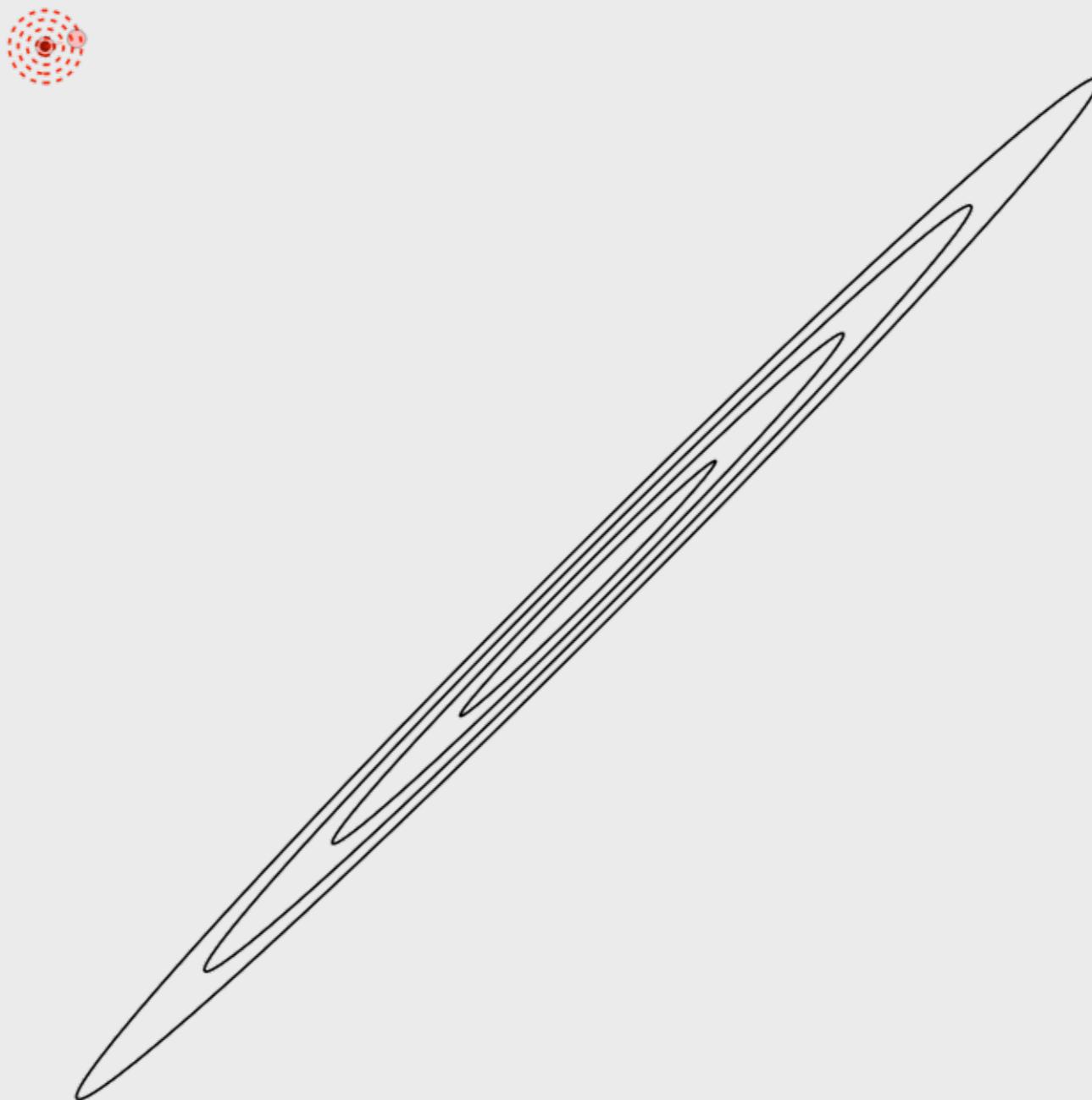
in the real world



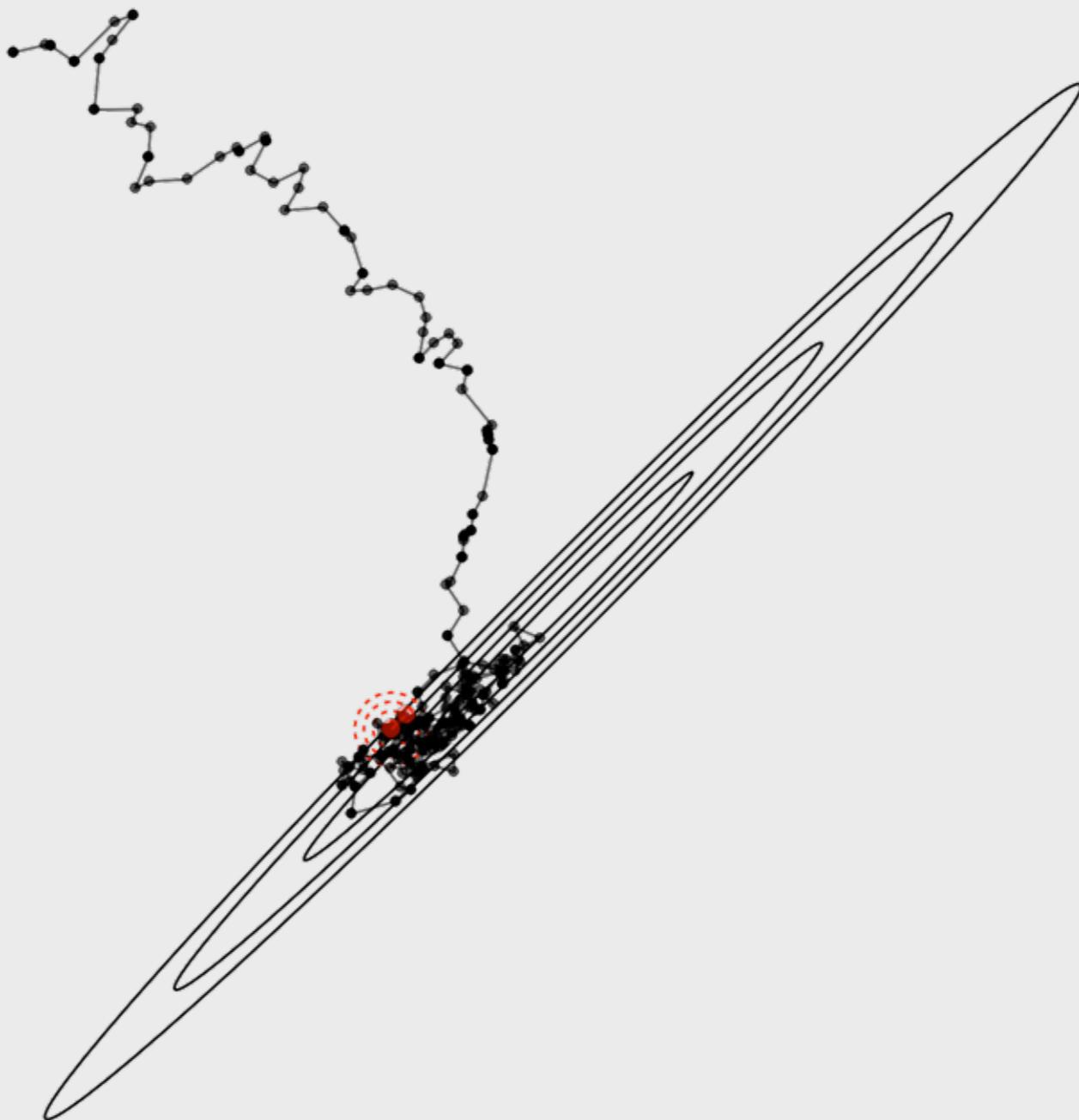
**Metropolis-Hastings**  
in the real world



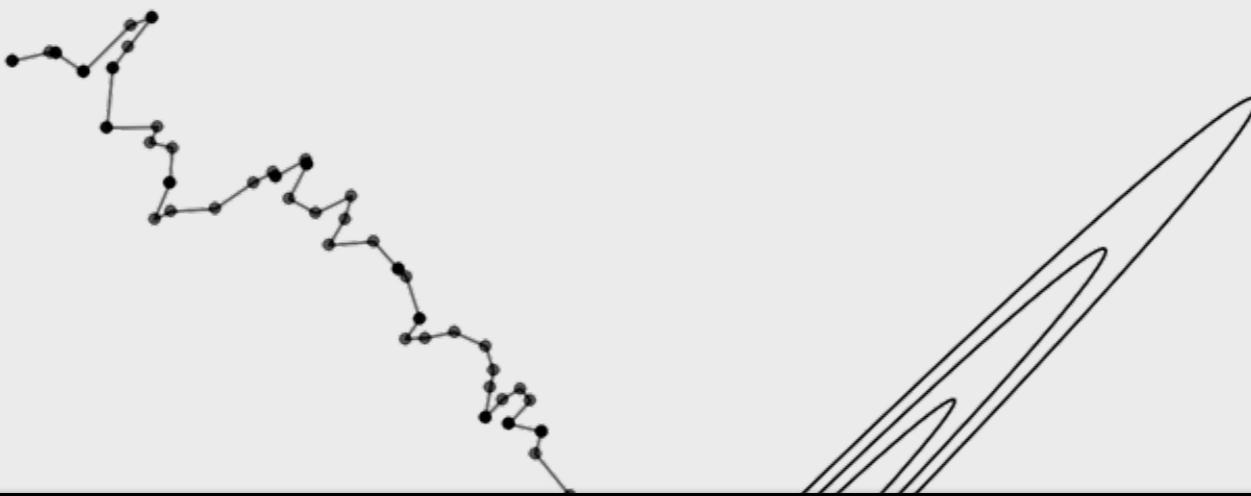
**Metropolis-Hastings**  
in the real world



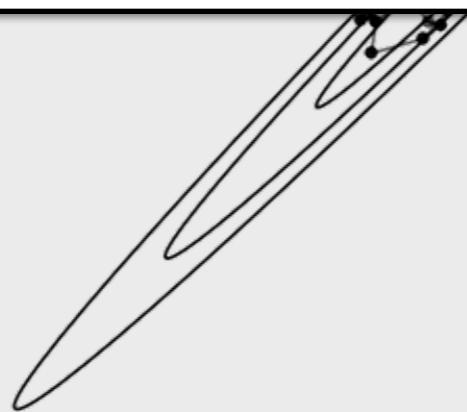
**Metropolis-Hastings**  
in the real world



**Metropolis-Hastings**  
in the real world



the  
**Huge Acceptance Fraction**  
problem

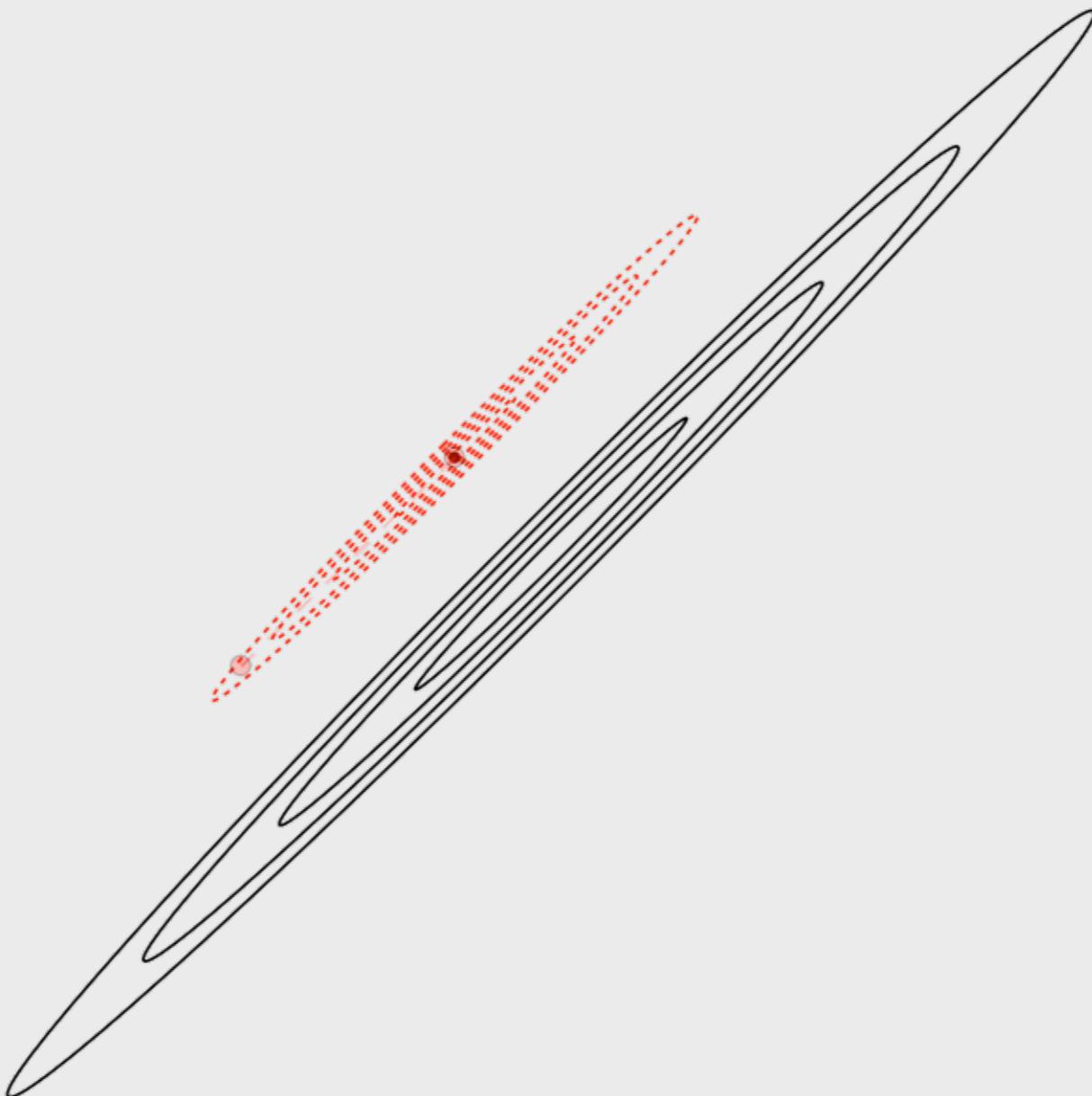


**Metropolis-Hastings**  
in the real world

a brief aside.



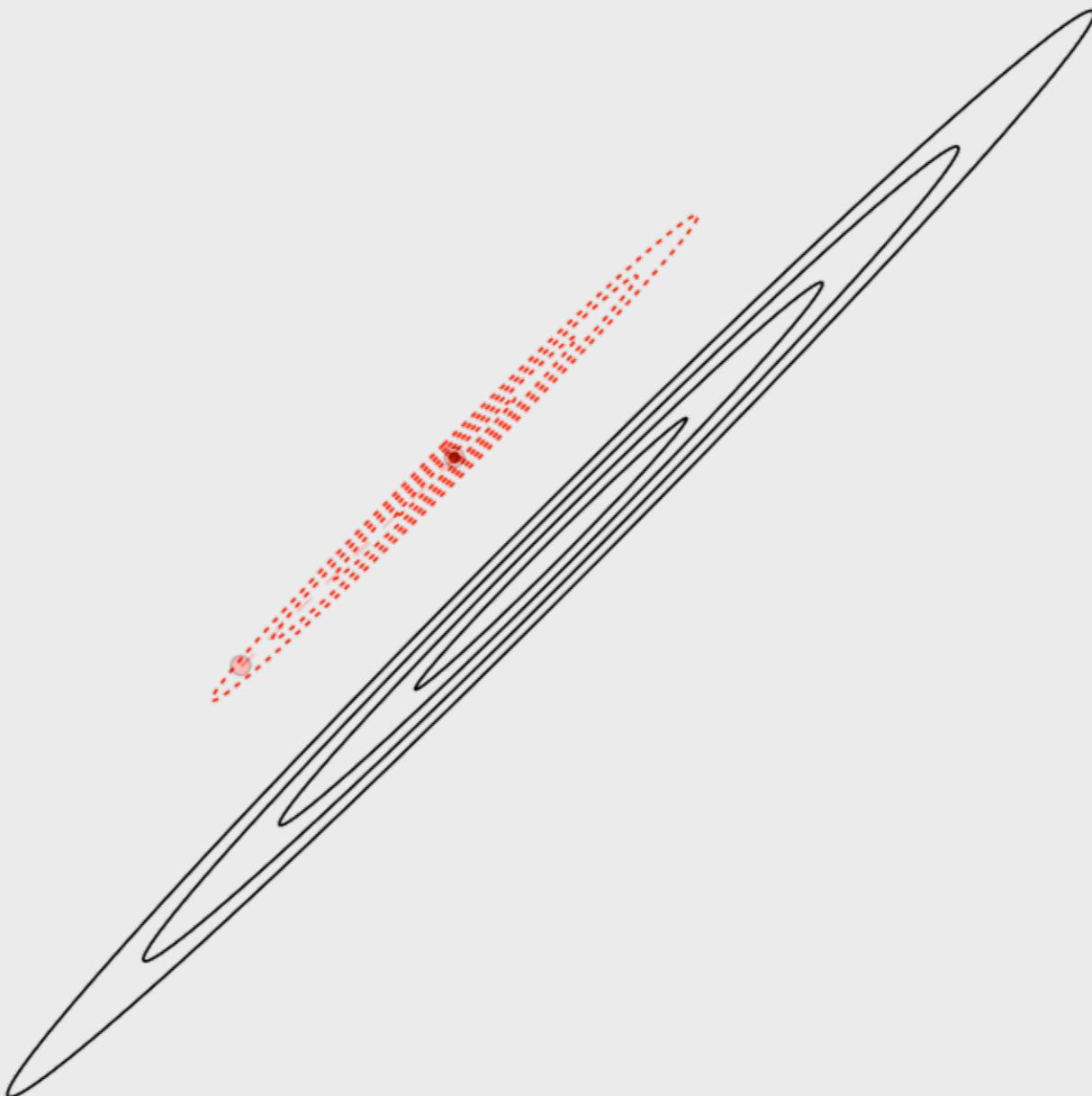
# YOUR LIKELIHOOD FUNCTION IS EXPENSIVE



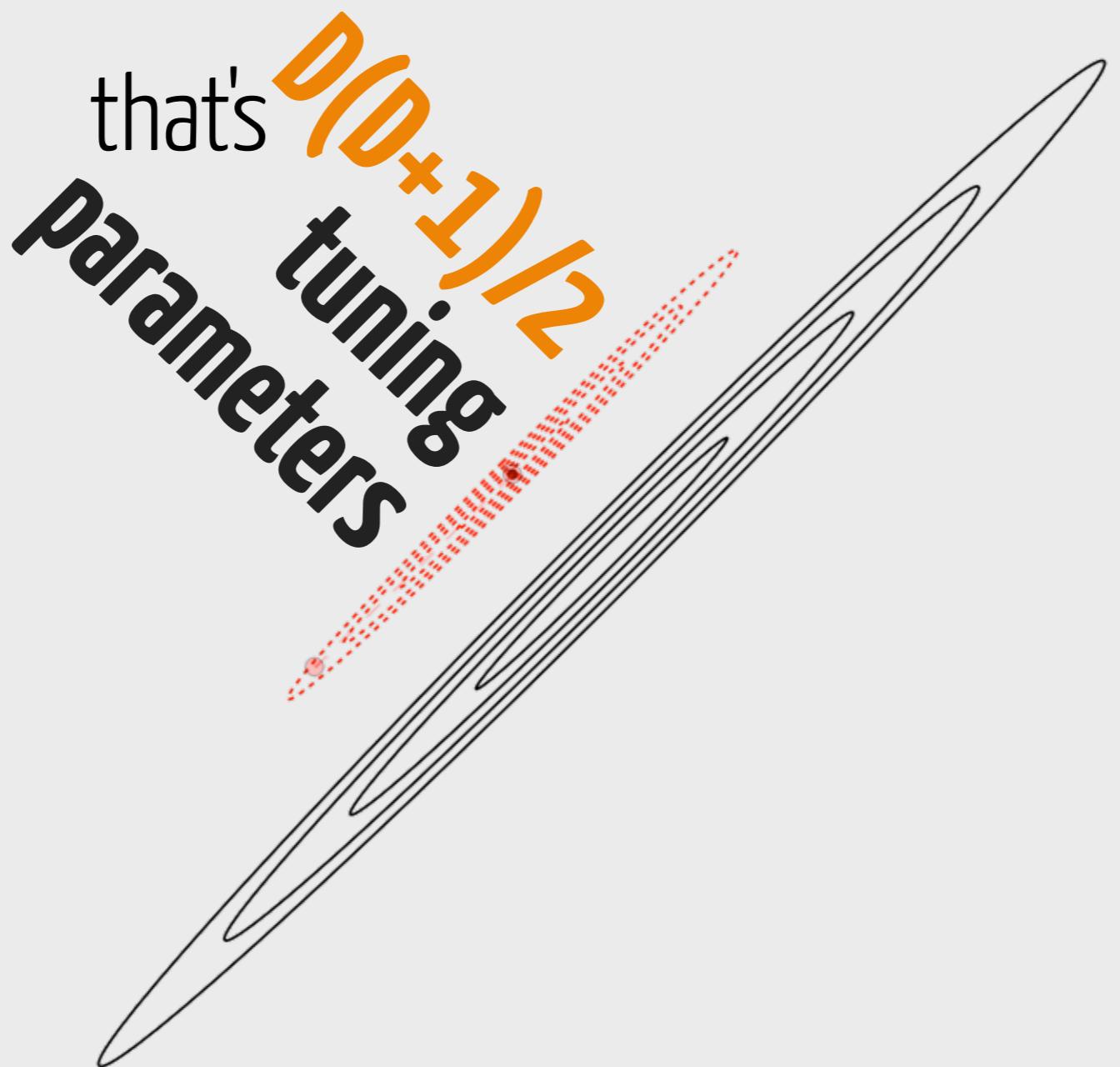
**Metropolis-Hastings**  
in the real world

# REMEMBER?

# YOUR LIKELIHOOD FUNCTION IS EXPENSIVE

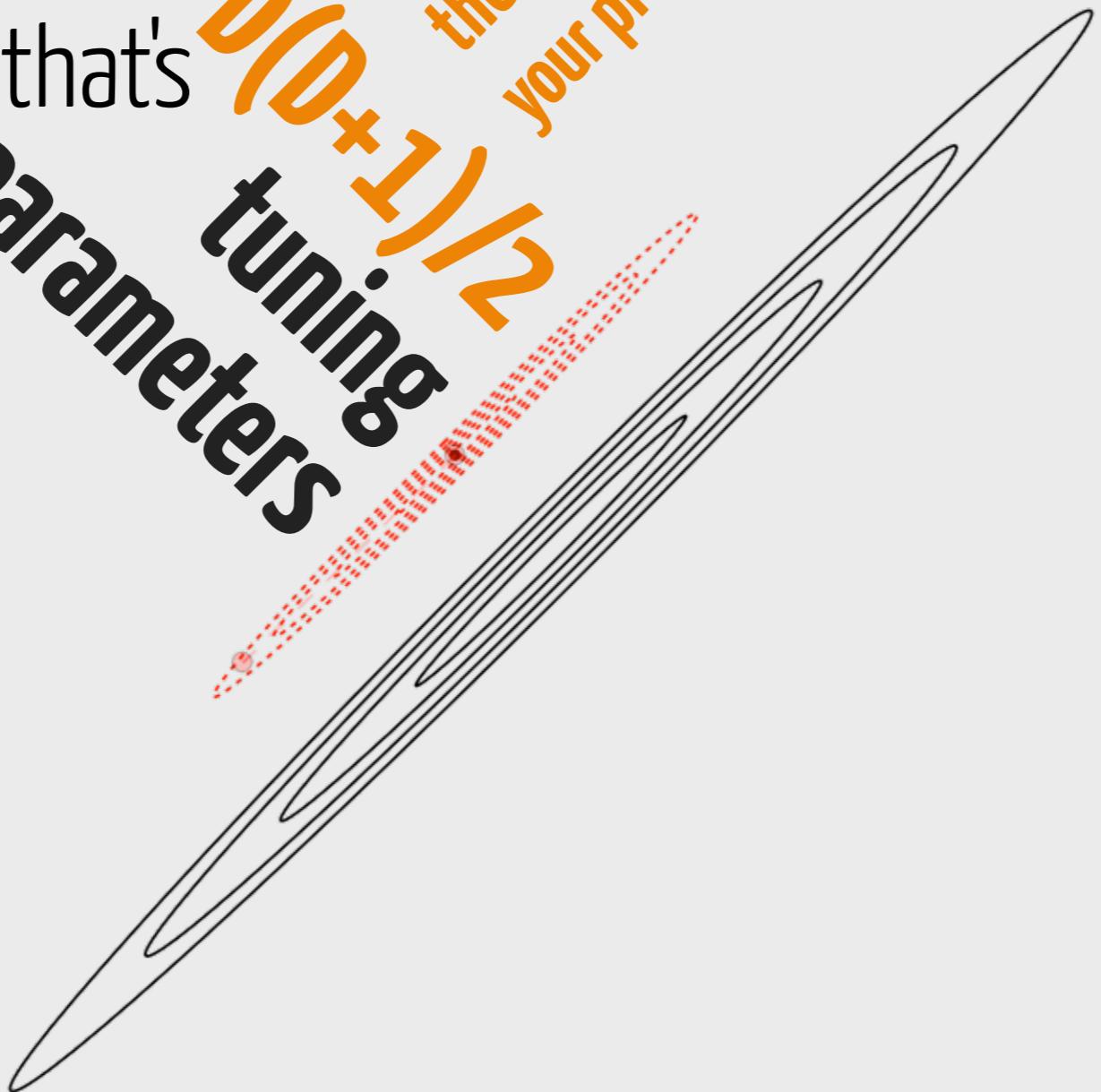


**Metropolis-Hastings**  
in the real world

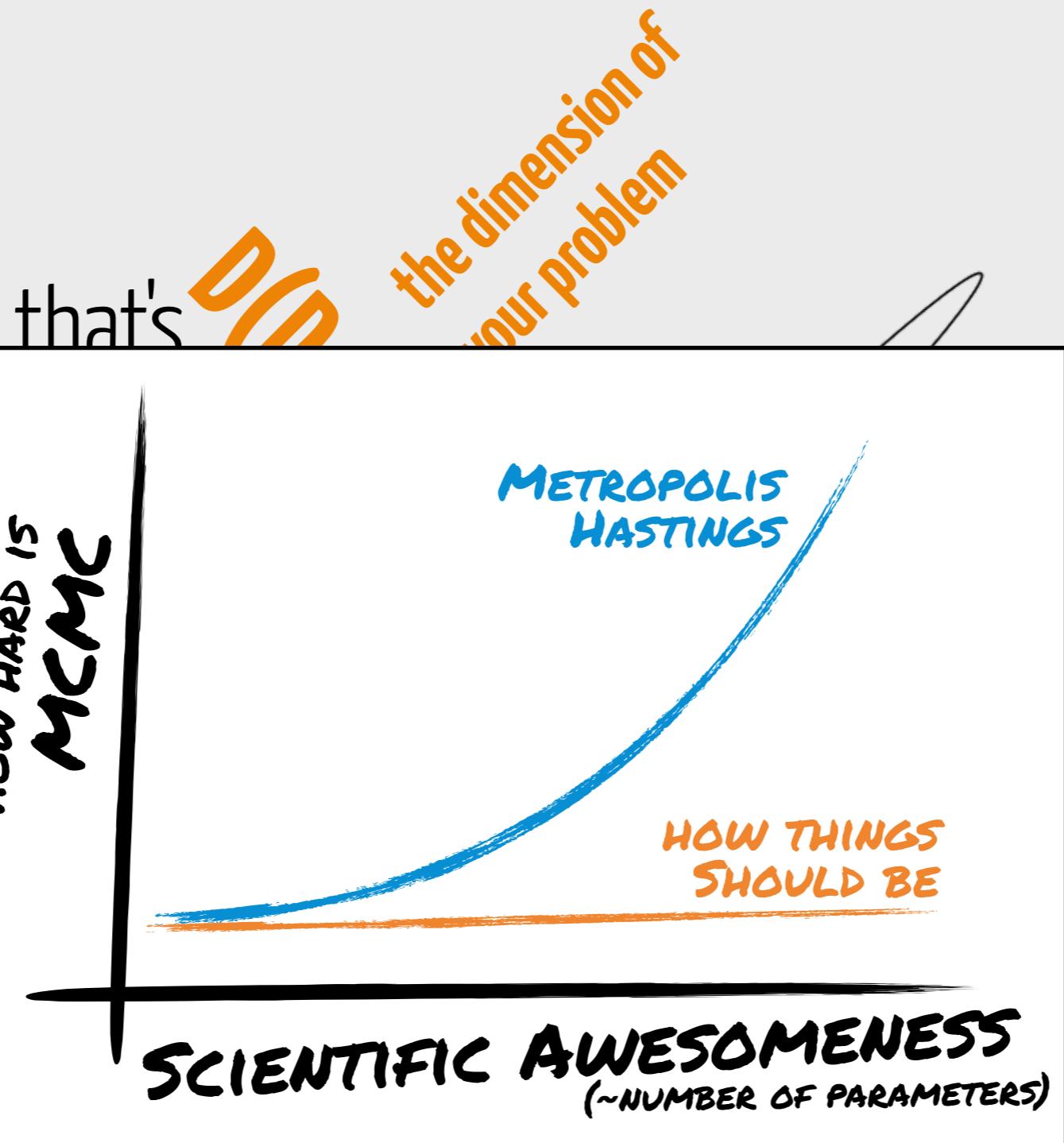


**Metropolis-Hastings**  
in the real world

that's  $\frac{D(D+1)}{2}$  tuning parameters  
the dimension of your problem



**Metropolis-Hastings**  
in the real world



# Metropolis-Hastings

in the real world

that being said,

go code up your own

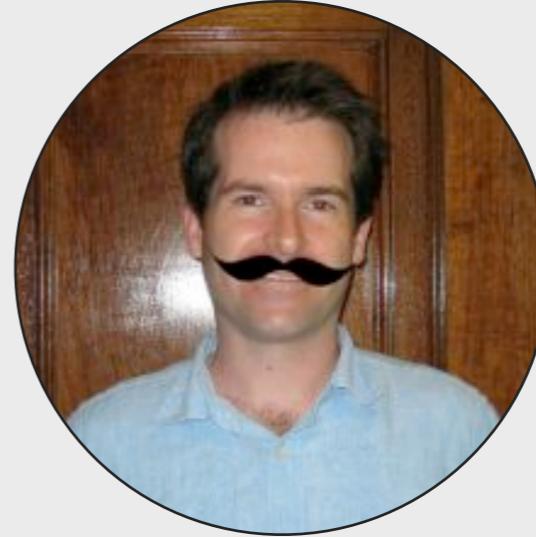
**Metropolis-Hastings**

code right now.

(well not right right now)



Jonathan Goodman



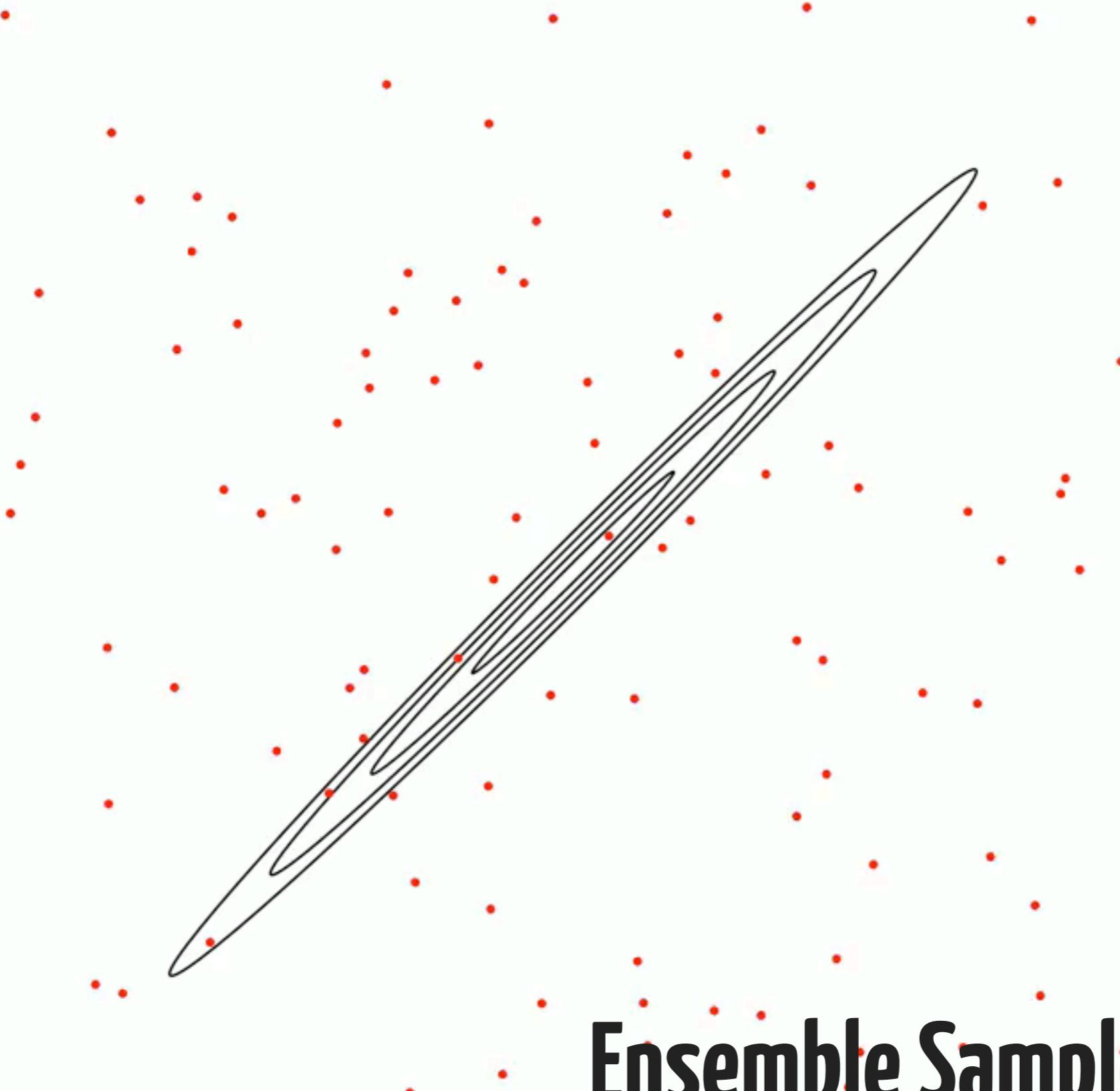
Jonathan Weare

# "Ensemble samplers with affine invariance"

([dfm.io/mcmc-gw10](http://dfm.io/mcmc-gw10))

# **Ensemble Samplers**

in the real world

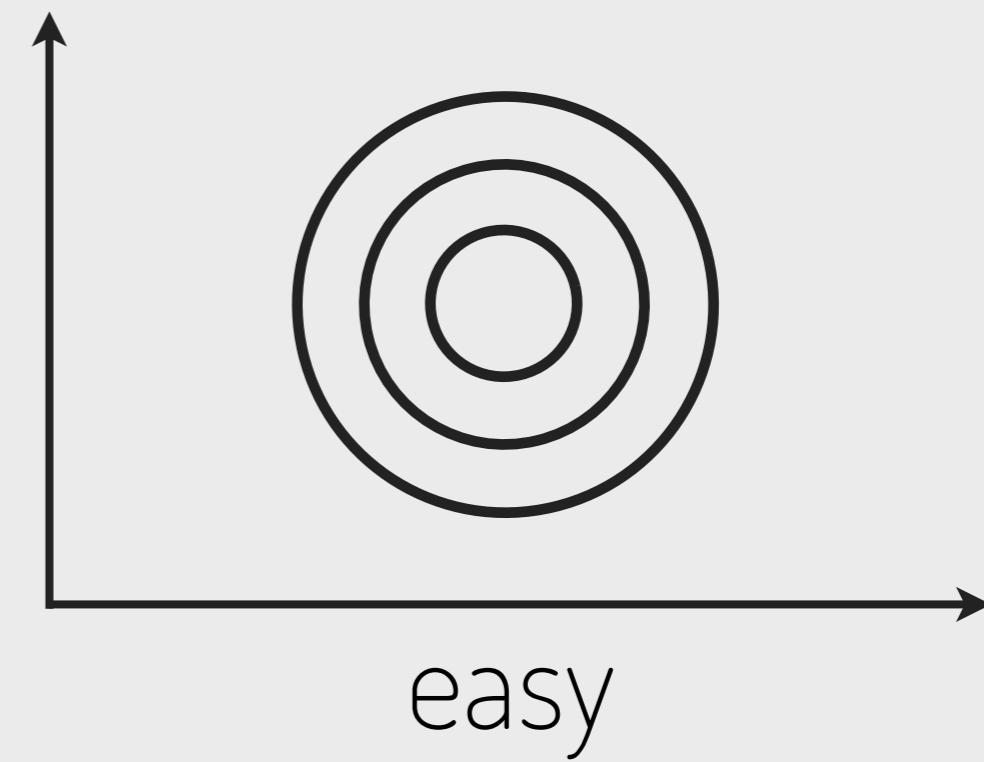
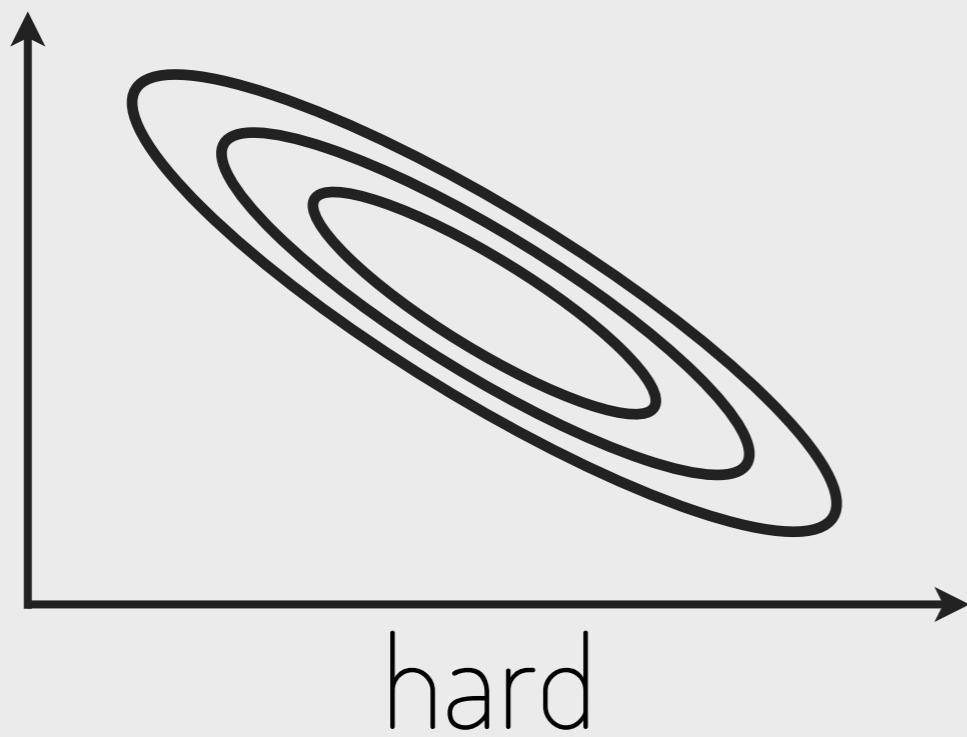


**Ensemble Samplers**  
in the real world

# Ensemble samplers with affine invariance

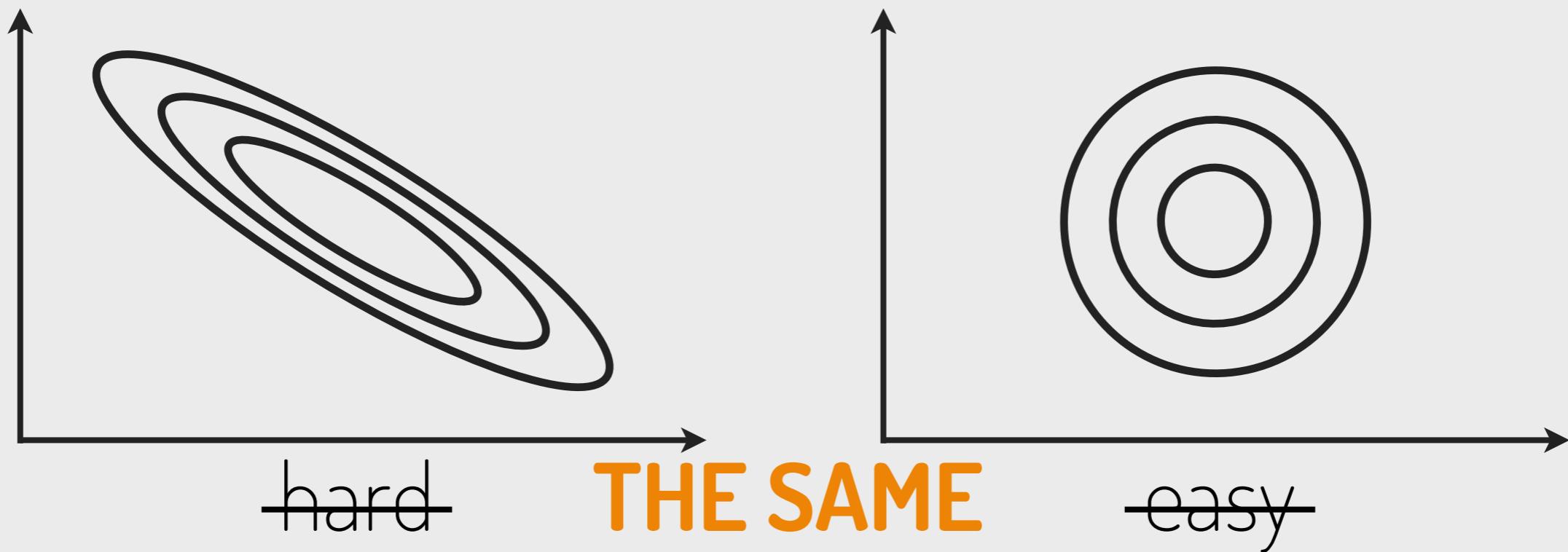
# Affine Transformation

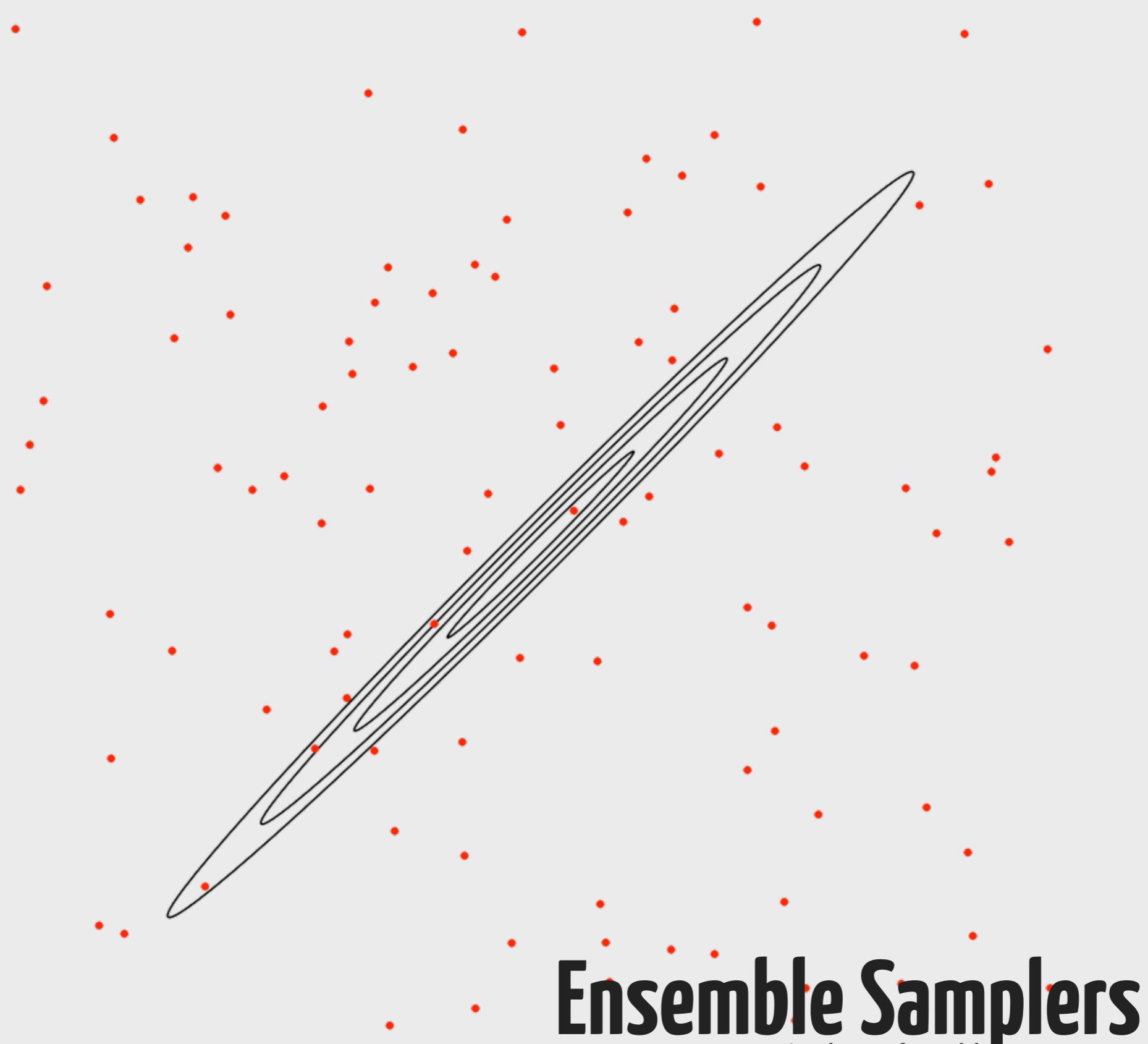
$$y \leftarrow A x + b$$



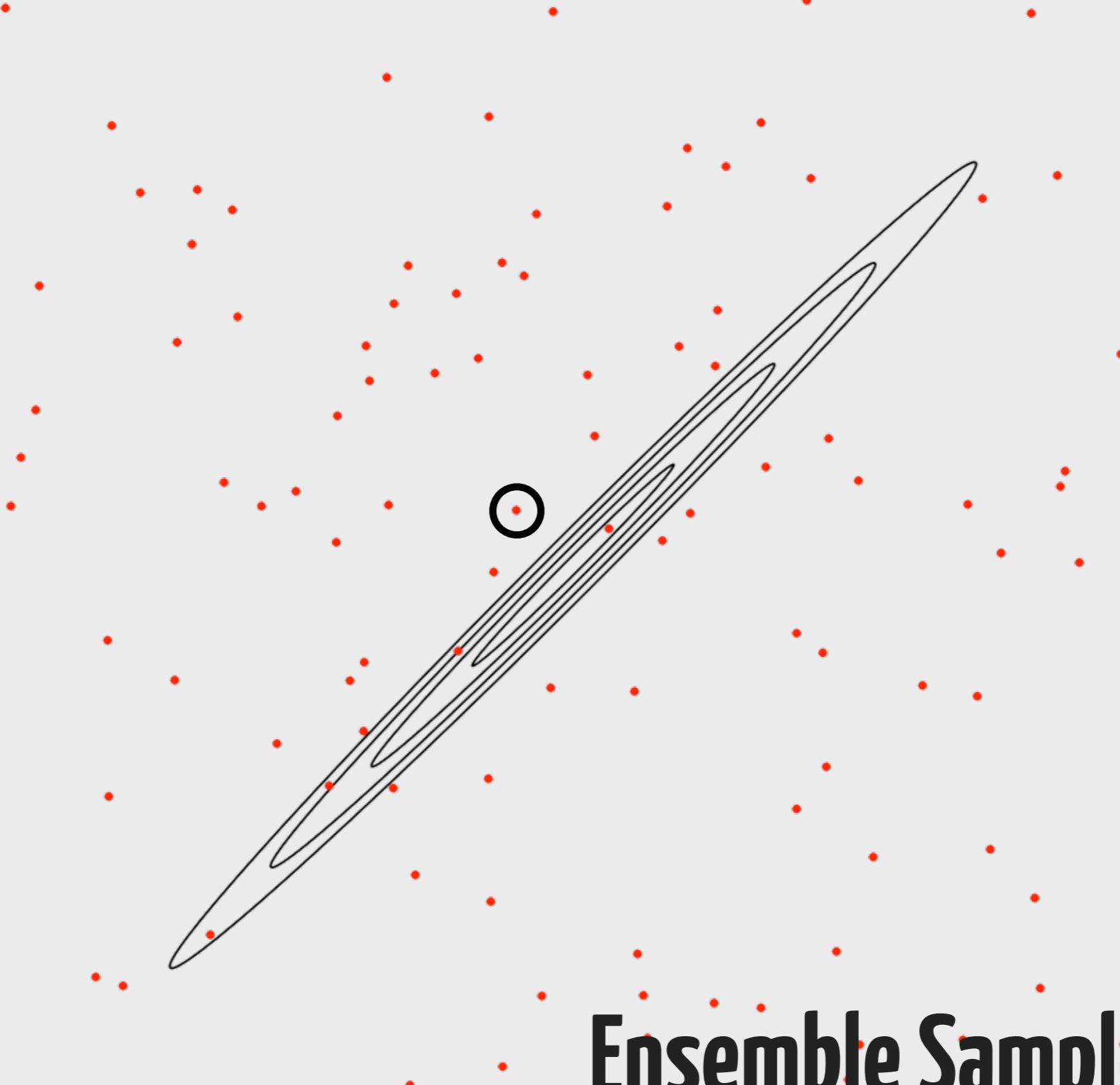
# Affine Transformation

$$y \leftarrow A x + b$$

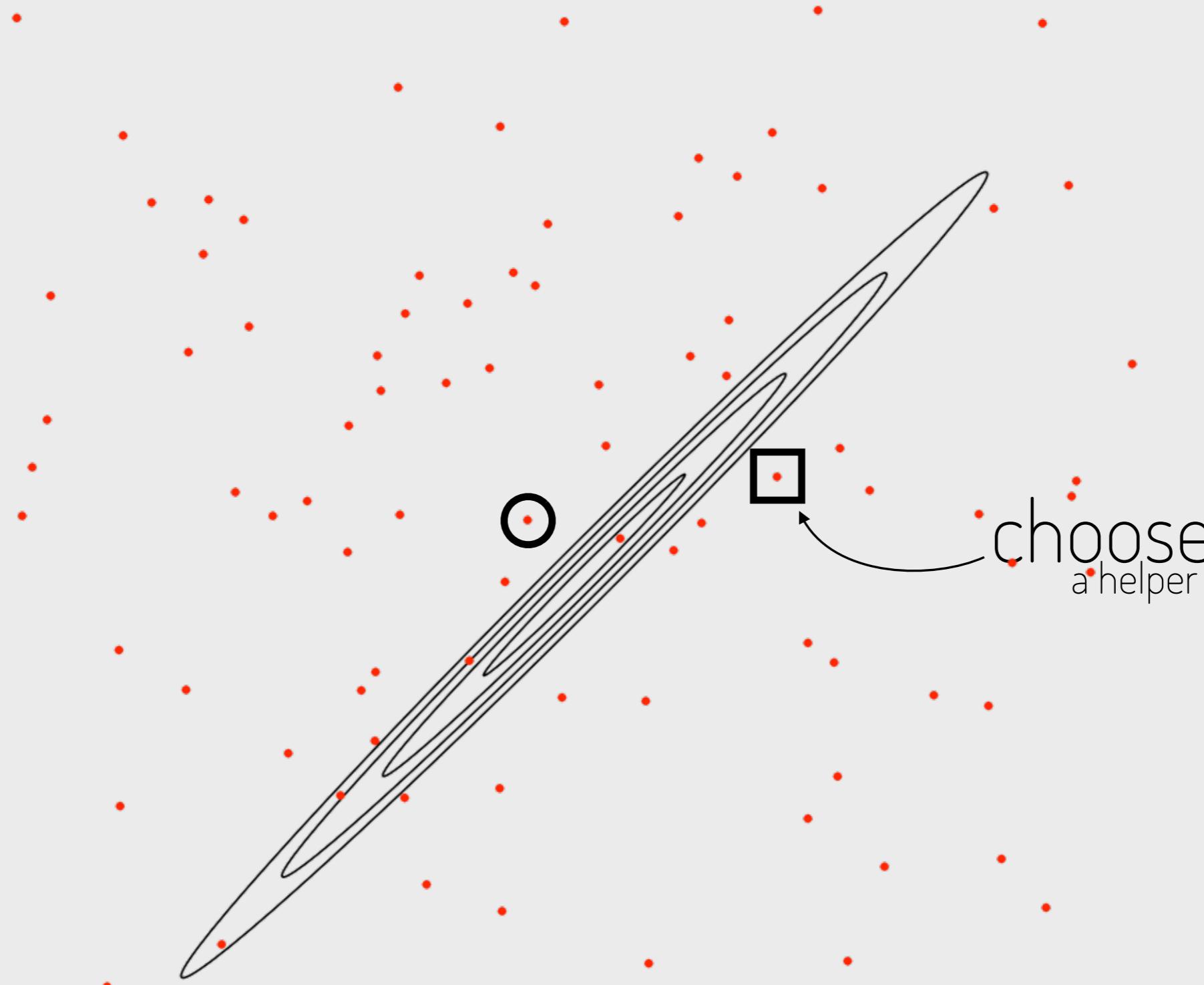




**Ensemble Samplers**  
in the real world

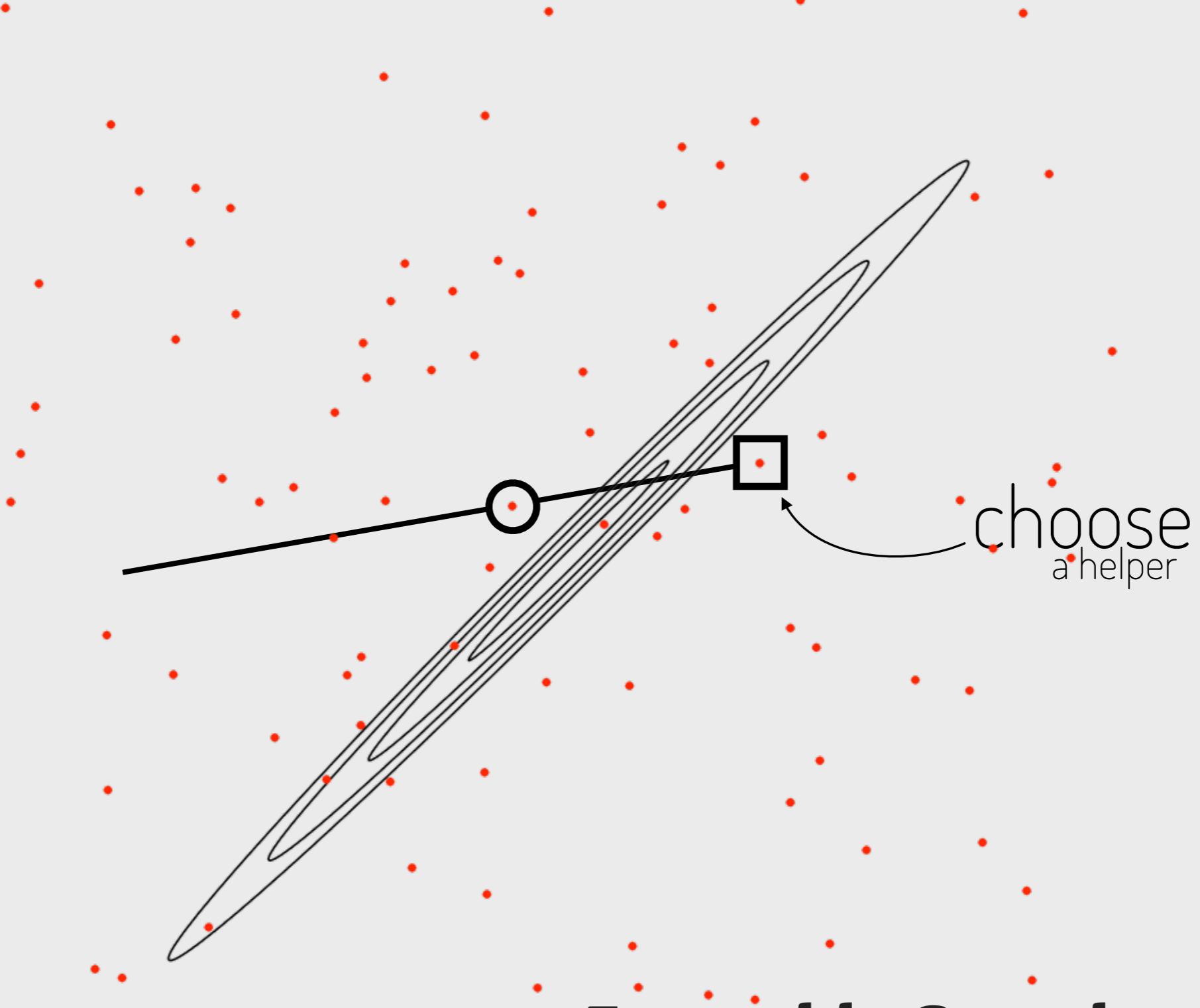


**Ensemble Samplers**  
in the real world



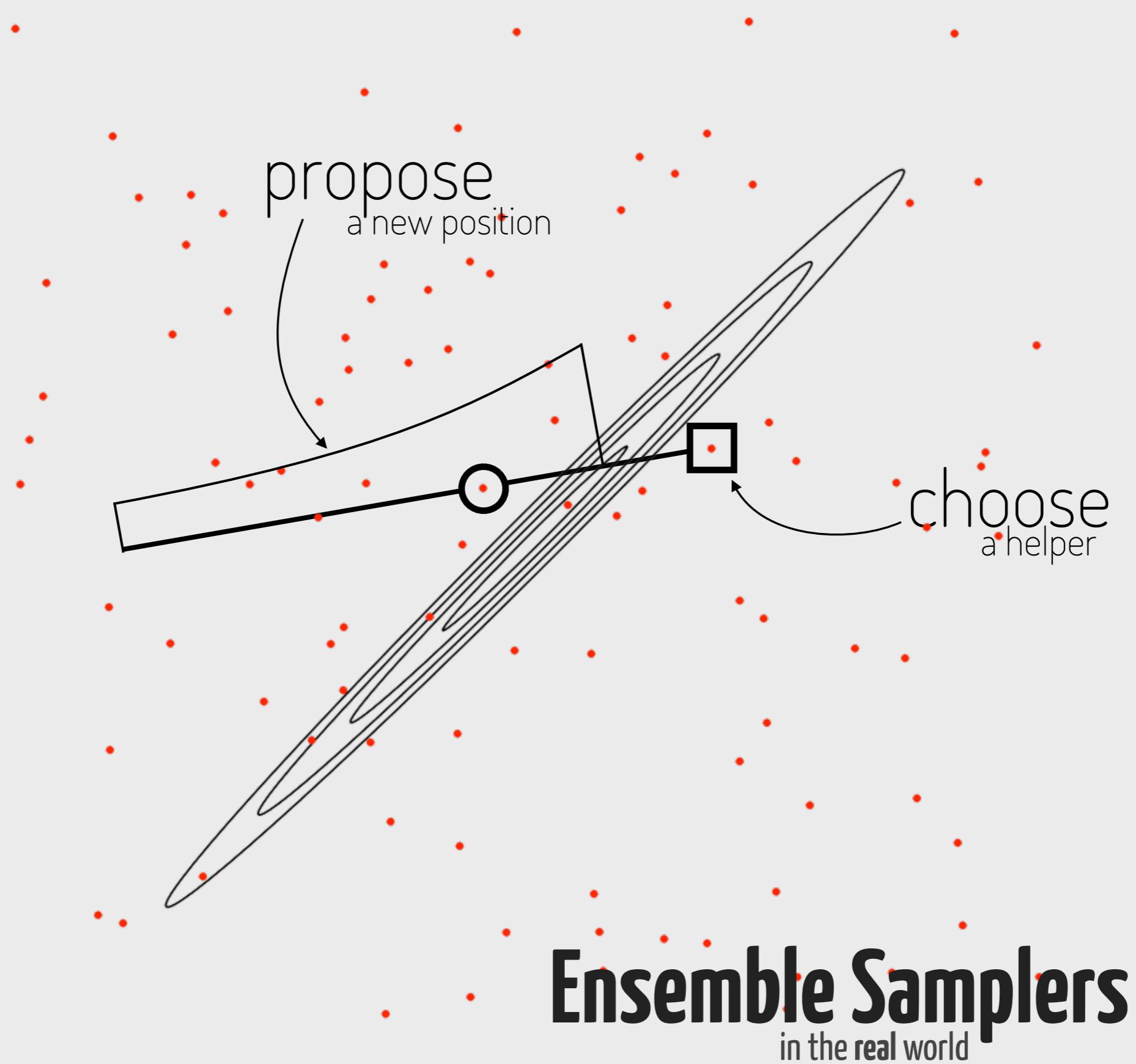
# Ensemble Samplers

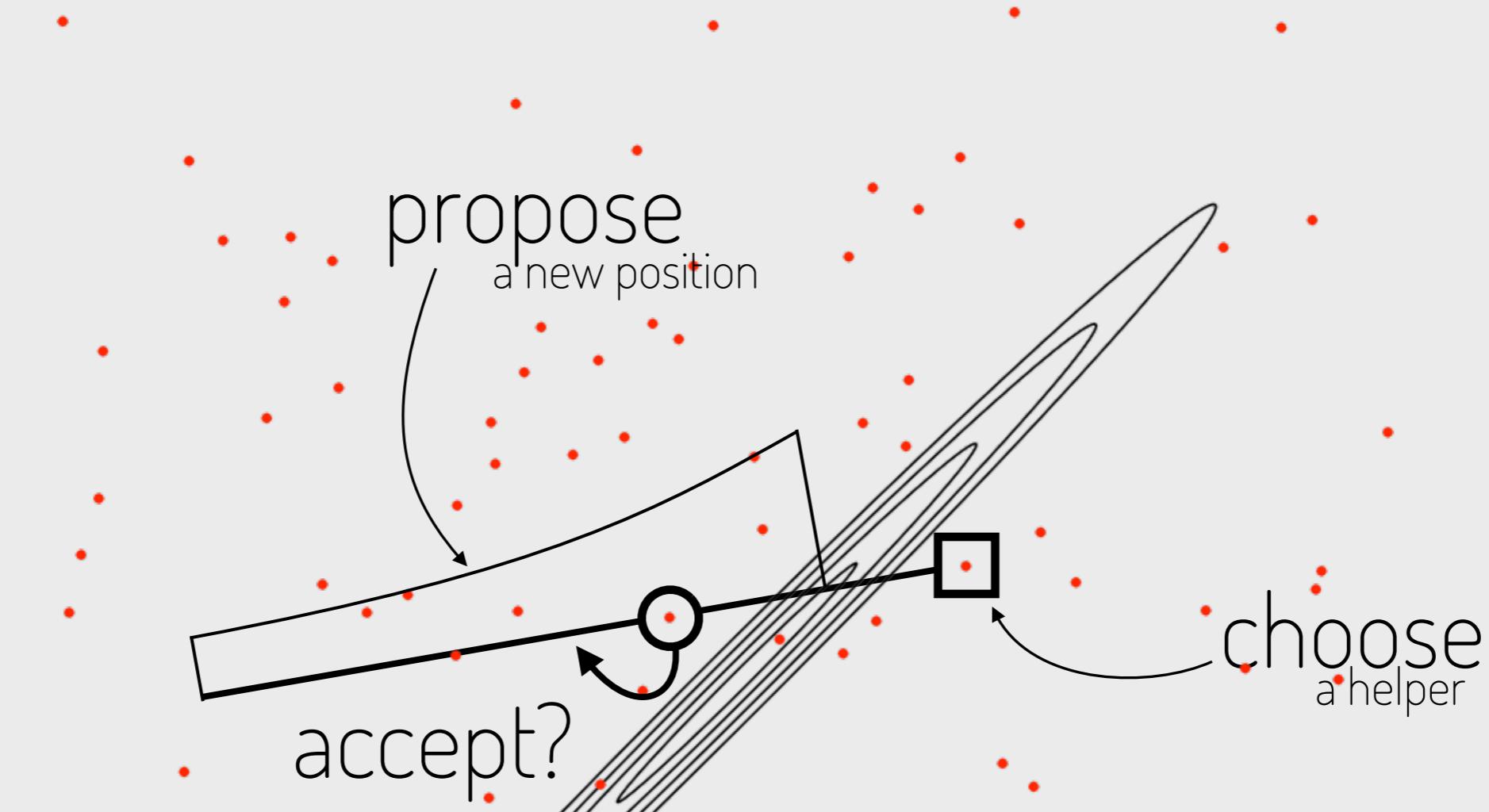
in the real world



# Ensemble Samplers

in the real world

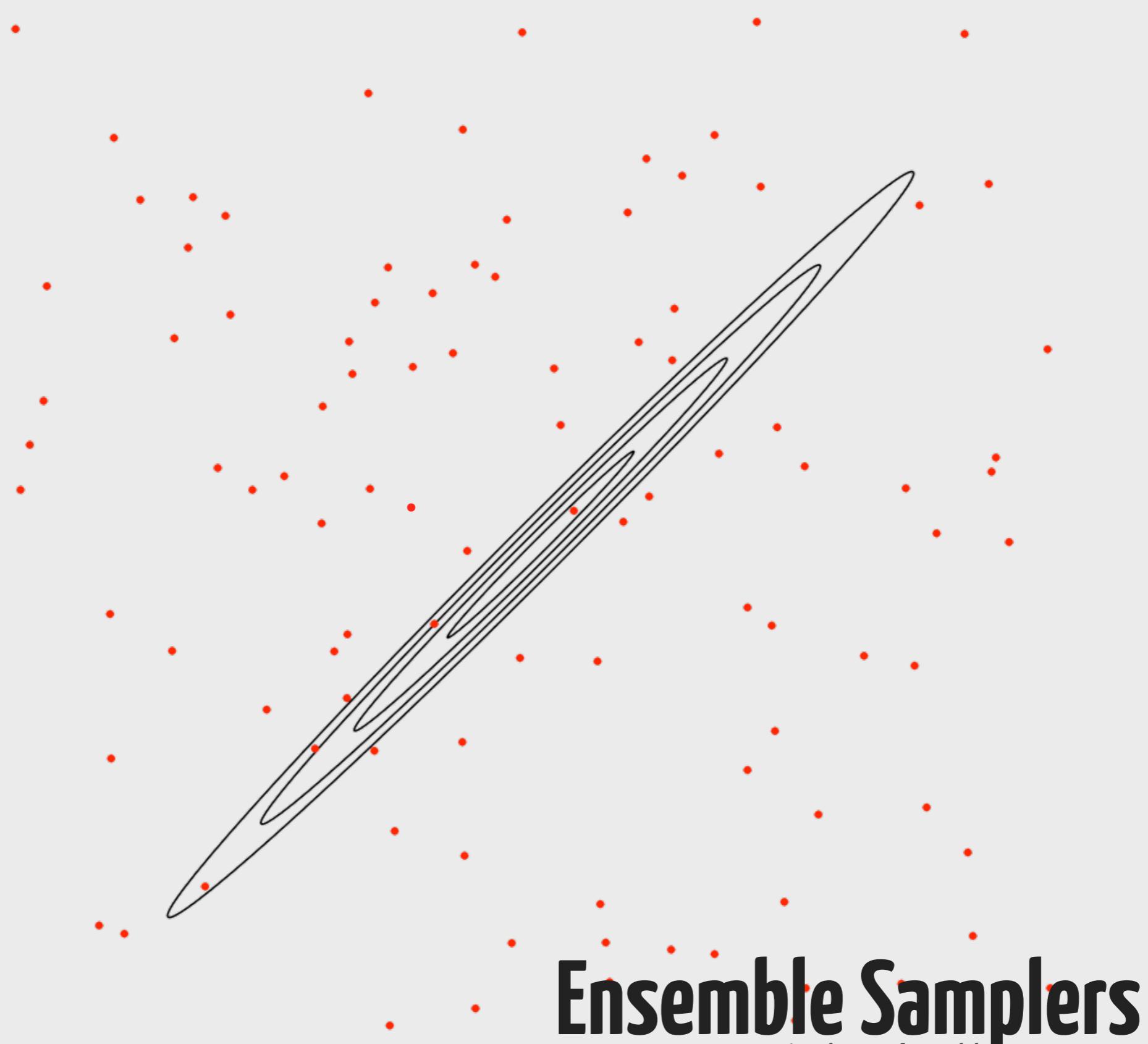


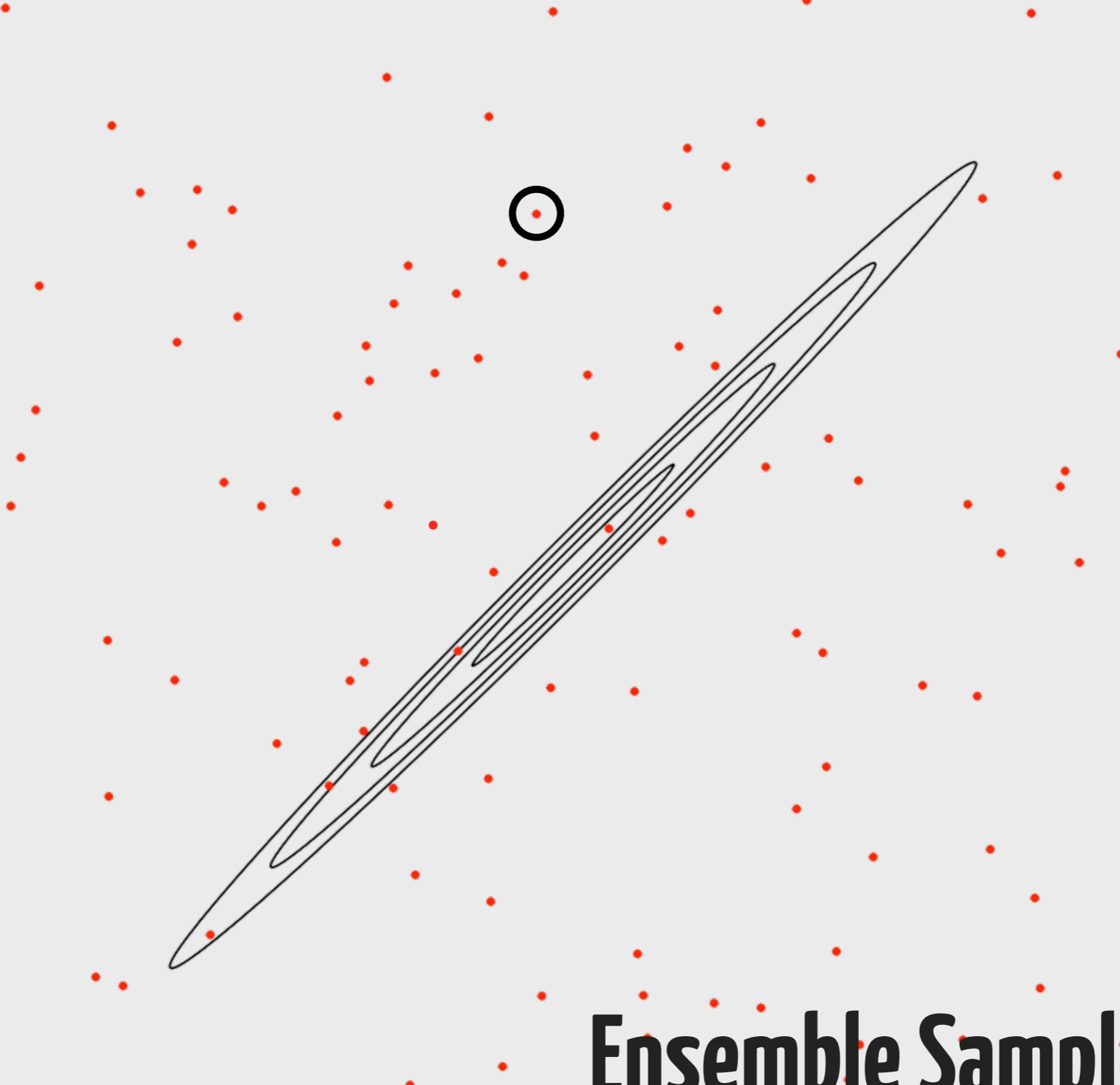


$$p(\text{accept}) = \min \left( 1, Z^{D-1} \frac{p(\mathbf{x})}{p(\mathbf{x}')}\right)$$

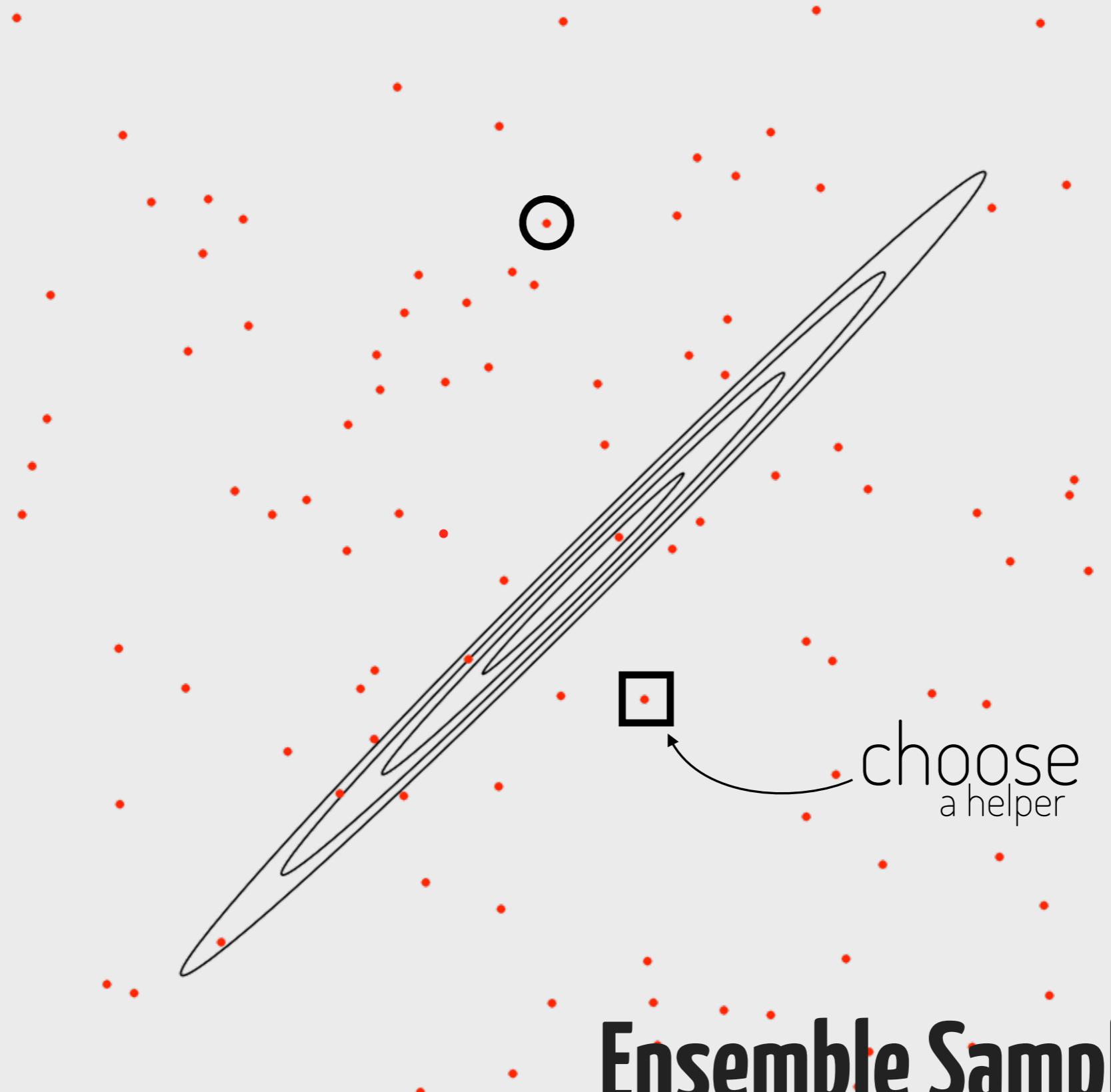
# Ensemble Samplers

in the real world



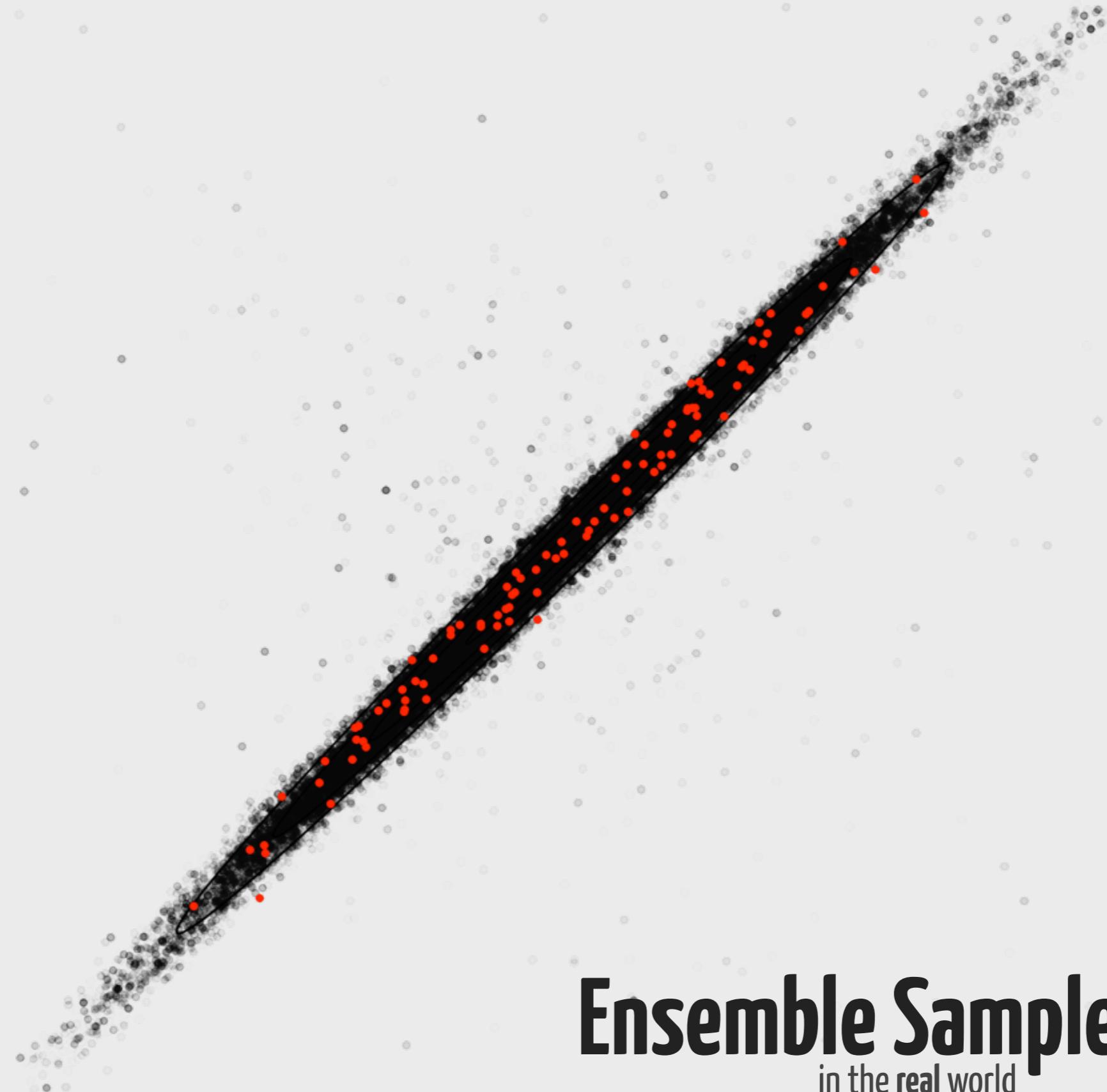


**Ensemble Samplers**  
in the real world



# Ensemble Samplers

in the real world



# Ensemble Samplers

in the real world

go code up your own

# Ensemble Sampler

code right now?



introducing **emcee** the **MCMC**  
Hammer

[arxiv.org/abs/1202.3665](https://arxiv.org/abs/1202.3665)  
[dan.iel.fm/emcee](http://dan.iel.fm/emcee)



to install:

**pip install emcee**

introducing **emcee** the MCMC Hammer

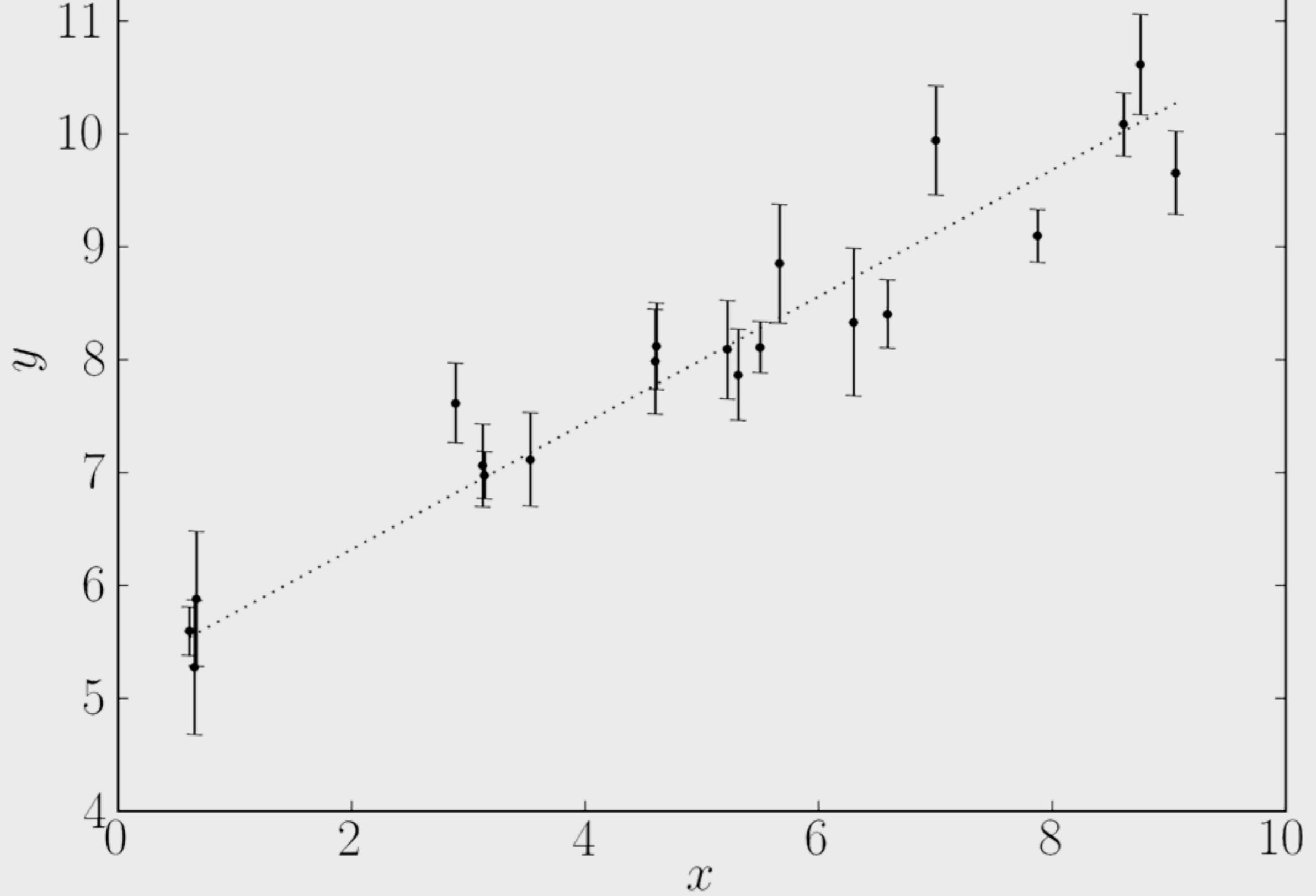
[arxiv.org/abs/1202.3665](https://arxiv.org/abs/1202.3665)  
[dan.iel.fm/emcee](https://dan.iel.fm/emcee)

using **emcee**  
is **easy**



let me show you...

$$p(\{y_n, x_n, \sigma_n^2\} | m, b, \delta^2) = \prod_{n=1}^N \frac{1}{\sqrt{2\pi(\sigma_n^2 + \delta^2)}} \exp\left[-\frac{1}{2} \frac{[y_n - (m x_n + b)]^2}{\sigma_n^2 + \delta^2}\right]$$



```
import numpy as np
import emcee

def lnprobfn(p, x, y, yerr):
    m, b, d2 = p

    if not 0 <= d2 <= 1:
        return -np.inf

    ivar = 1.0 / (yerr ** 2 + d2)
    chi2 = np.sum((y - m * x - b) ** 2 * ivar)

    return -0.5 * (chi2 - np.sum(np.log(ivar)))

# Load data.
# x, y, yerr = ...

# Set up sampler and initialize the walkers.
nwalkers, ndim = 100, 3
sampler = emcee.EnsembleSampler(nwalkers, ndim, lnprobfn, args=(x, y, yerr))
p0 = [[np.random.rand(), 10 * np.random.rand(), np.random.rand()]
       for k in range(nwalkers)]

# Go.
sampler.run_mcmc(p0, 1000)
m, b, d2 = sampler.flatchain.T
```

$$\ln p(\{x_n, y_n, \sigma_n\} | m, b, \delta^2) = -\frac{1}{2}\chi^2 - \frac{1}{2} \sum_{n=1}^N \ln (\sigma_n^2 + \delta^2)$$

$$\chi^2 = \sum_{n=1}^N \frac{(y_n - m x_n + b)^2}{\sigma_n^2 + \delta^2}$$

$$p(\delta^2) = \begin{cases} 1, & \text{if } 0 \leq \delta^2 \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

```

import numpy as np

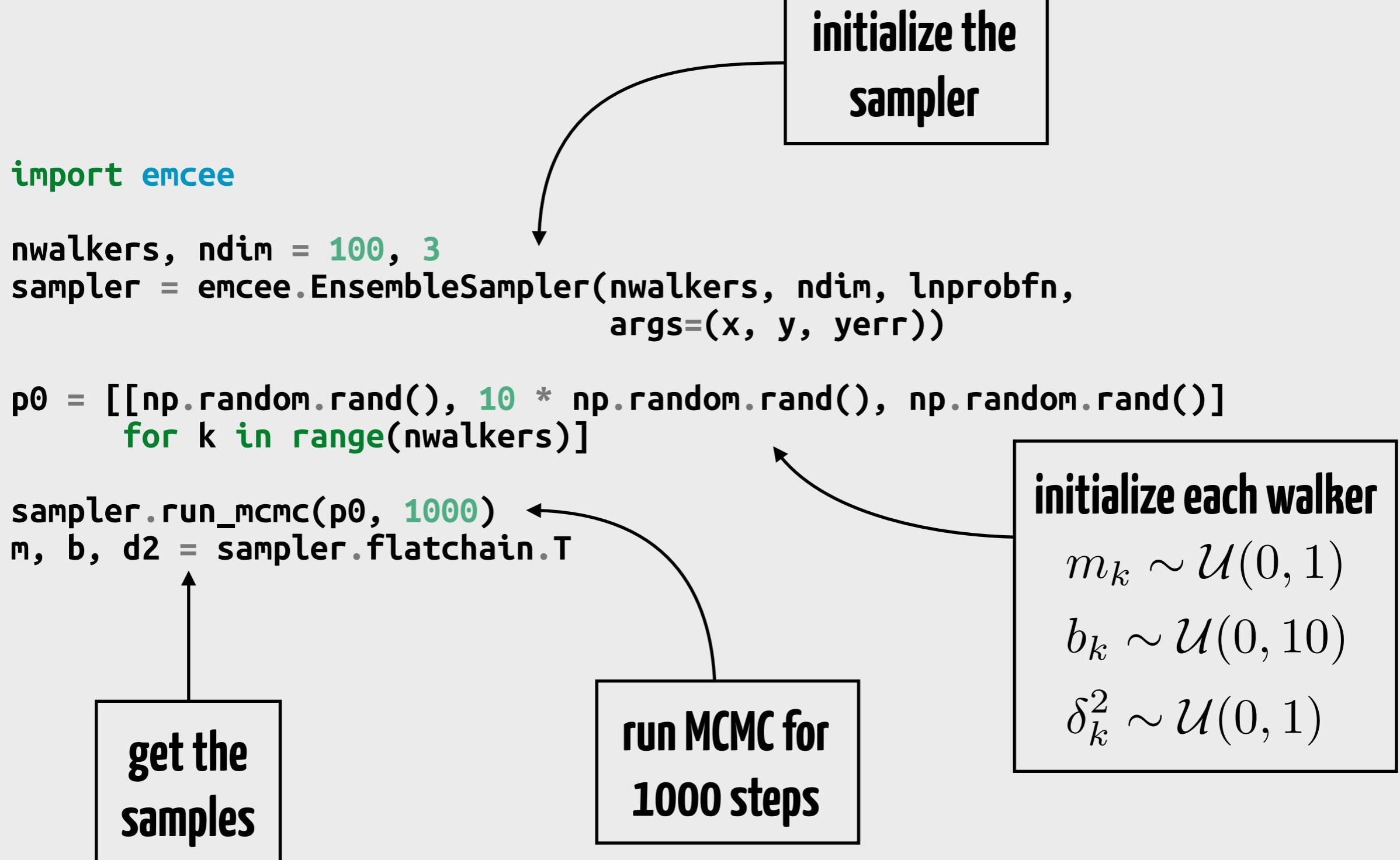
def lnprobfn(p, x, y, yerr):
    m, b, d2 = p

    if not 0 <= d2 <= 1:
        return -np.inf

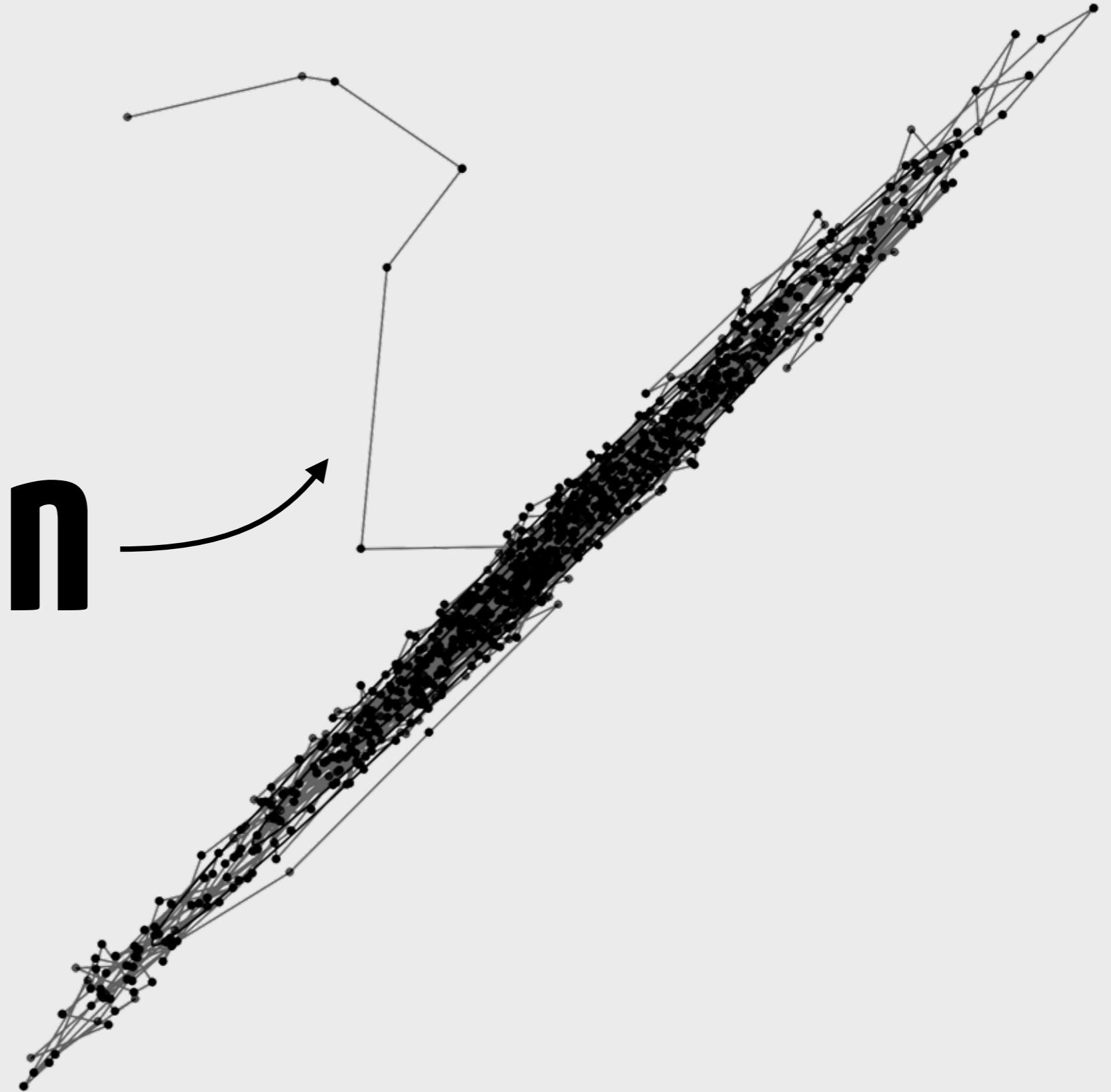
    ivar = 1.0 / (yerr ** 2 + d2)
    chi2 = np.sum((y - m * x - b) ** 2 * ivar)

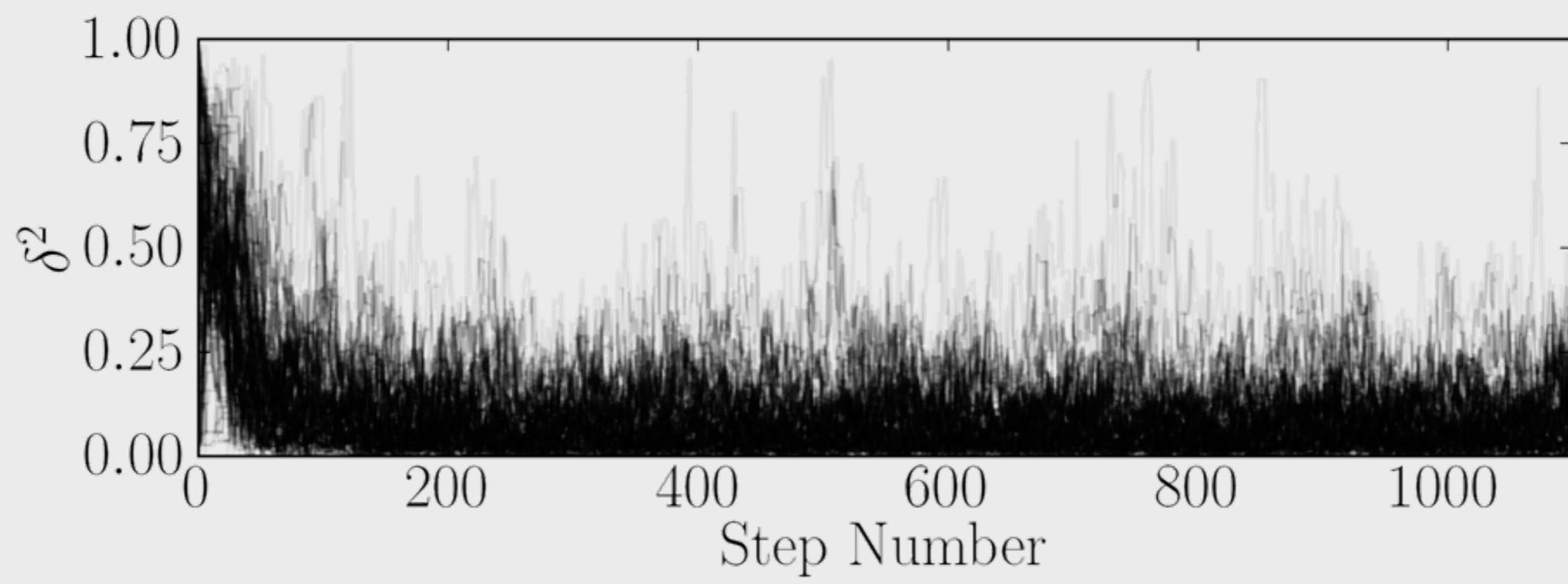
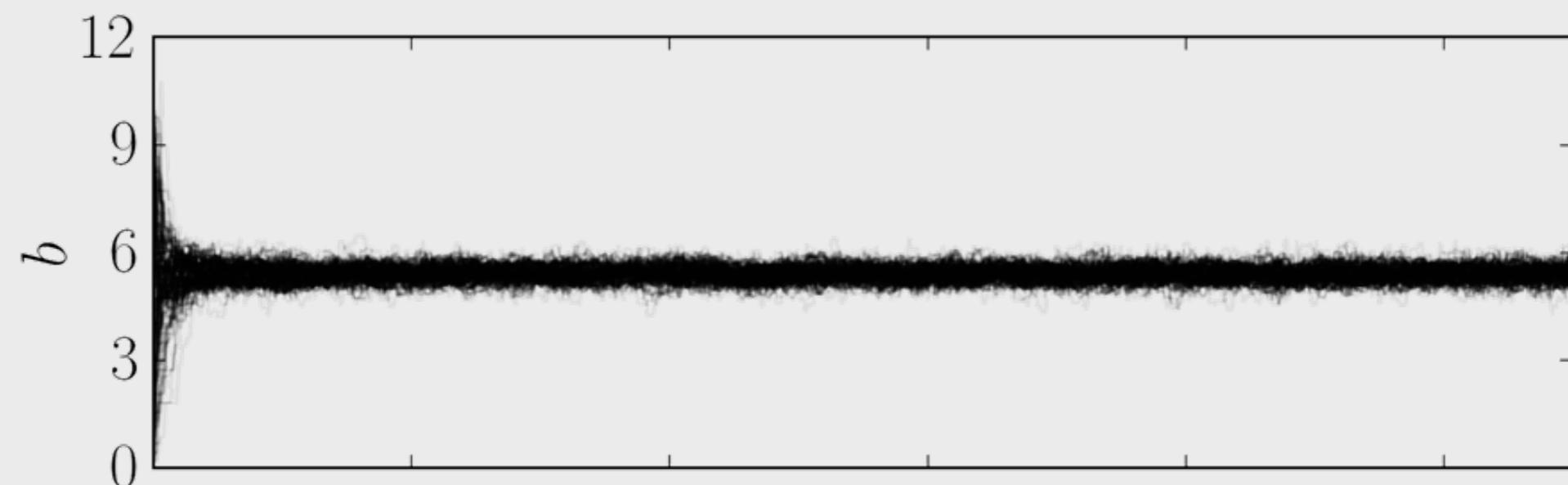
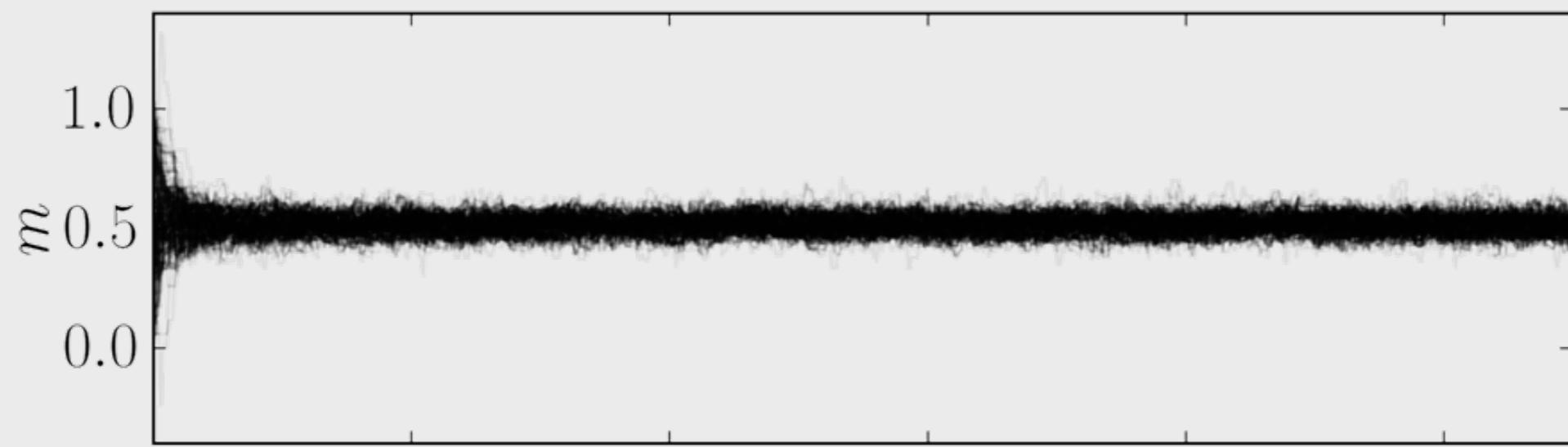
    return -0.5 * (chi2 - np.sum(np.log(ivar)))

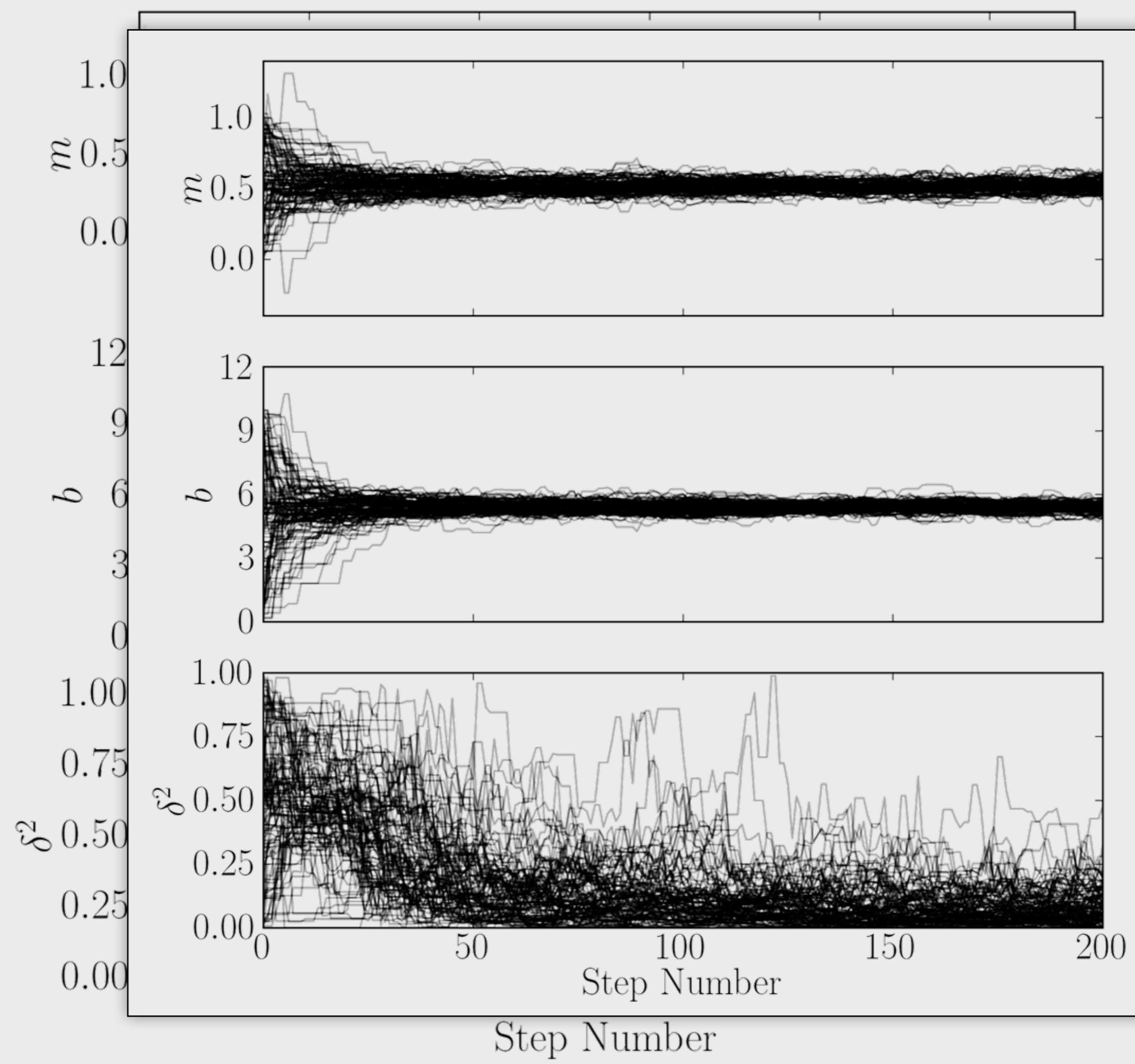
```



**burn-in**







**convergence**

# acceptance fraction?

The limiting diffusion approximation admits a straight-forward efficiency maximization problem, and the resulting asymptotically optimal policy is related to the asymptotic acceptance rate of proposed moves for the algorithm. The asymptotically optimal acceptance rate is 0.234 under quite general conditions.

# acceptance fraction?

The limiting diffusion approximation admits a straight-forward efficiency maximization problem, and the resulting asymptotically optimal policy is related to the asymptotic acceptance rate of proposed moves for the algorithm. The asymptotically optimal acceptance rate is 0.234 under quite general conditions.



whoa!

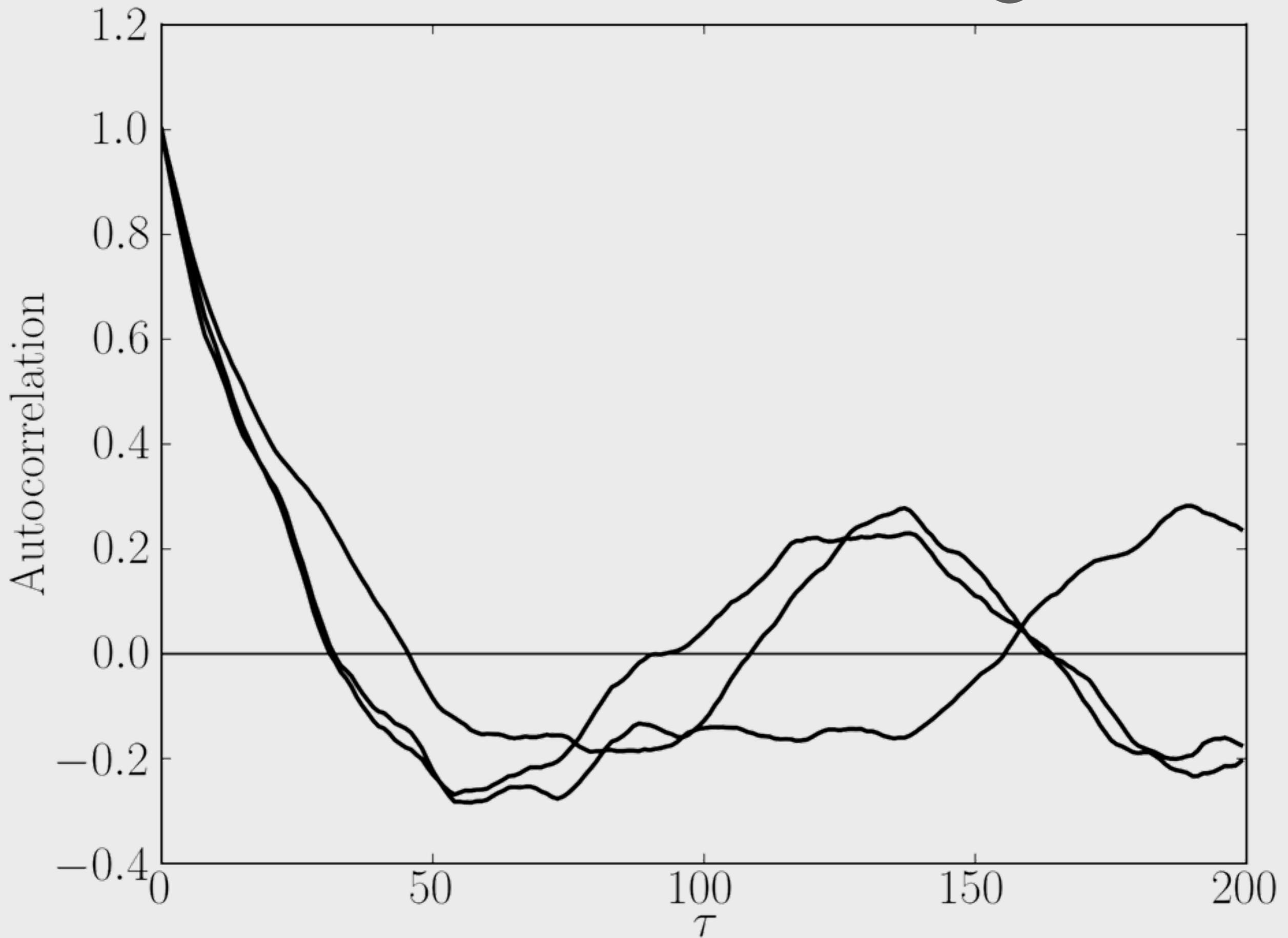
# acceptance fraction?

The limiting diffusion approximation admits a straight-forward efficiency maximization problem, and the resulting asymptotically optimal policy is related to the asymptotic acceptance rate of proposed moves for the algorithm. The asymptotically optimal acceptance rate is 0.234 under quite general conditions.

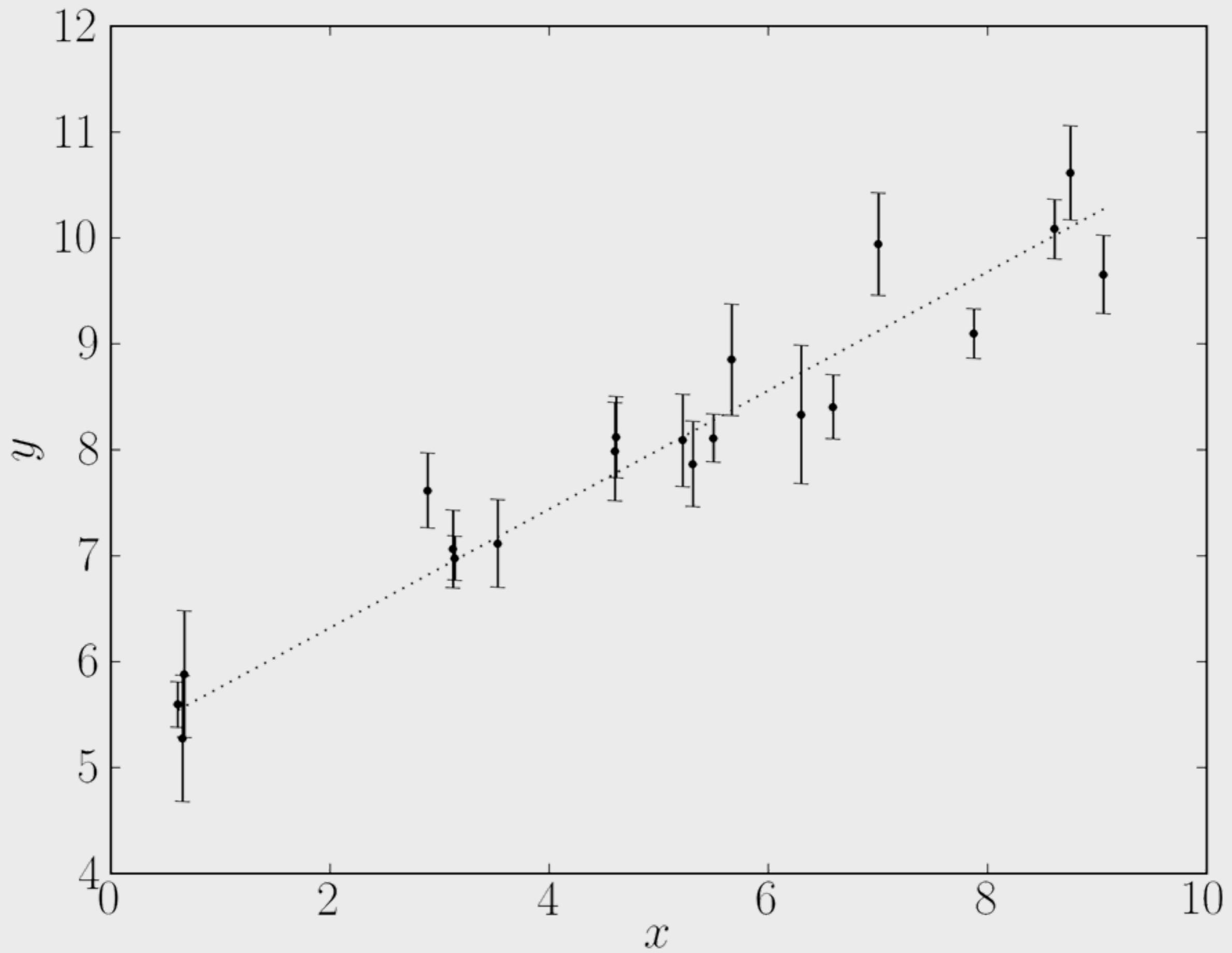


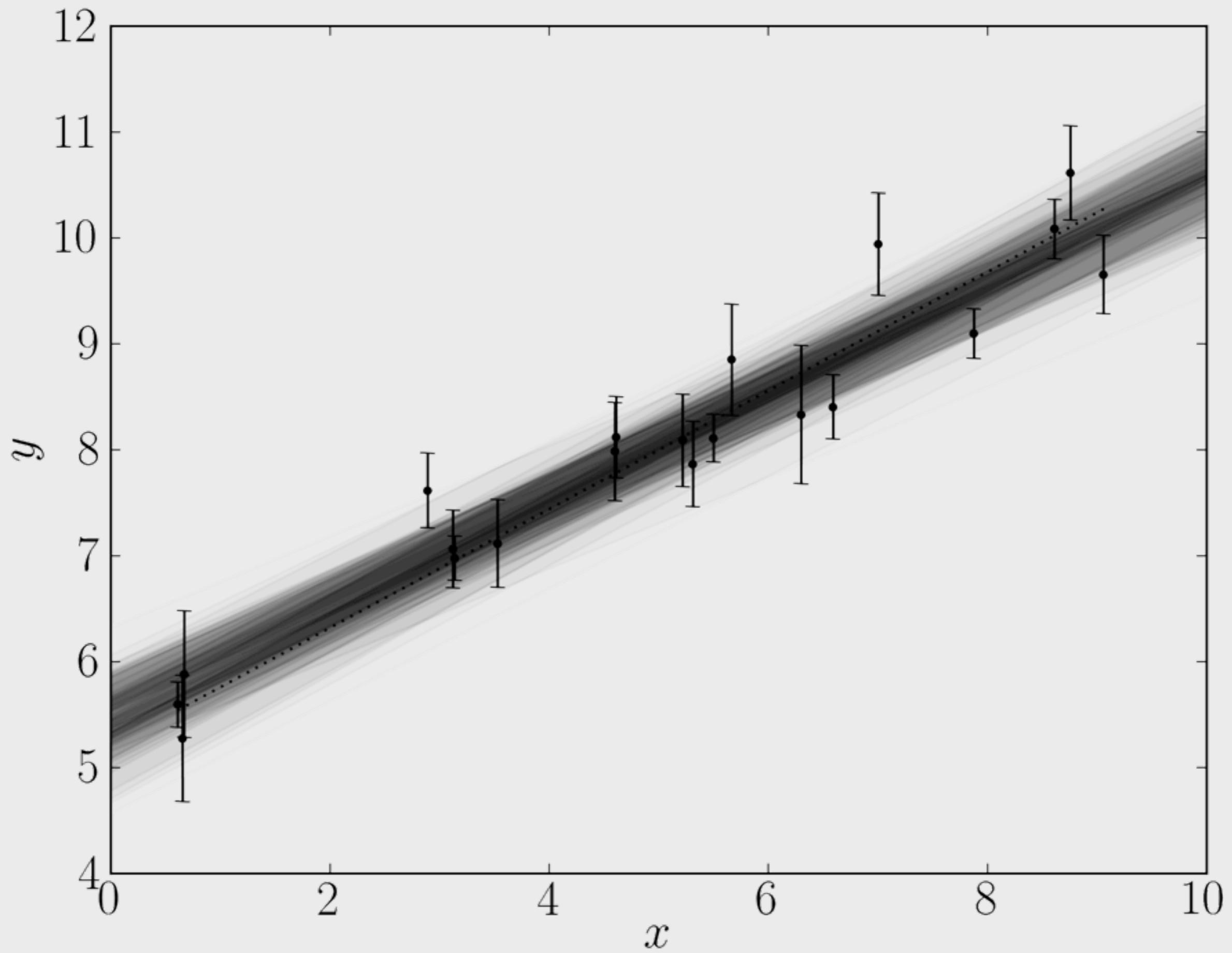
**whoa!**

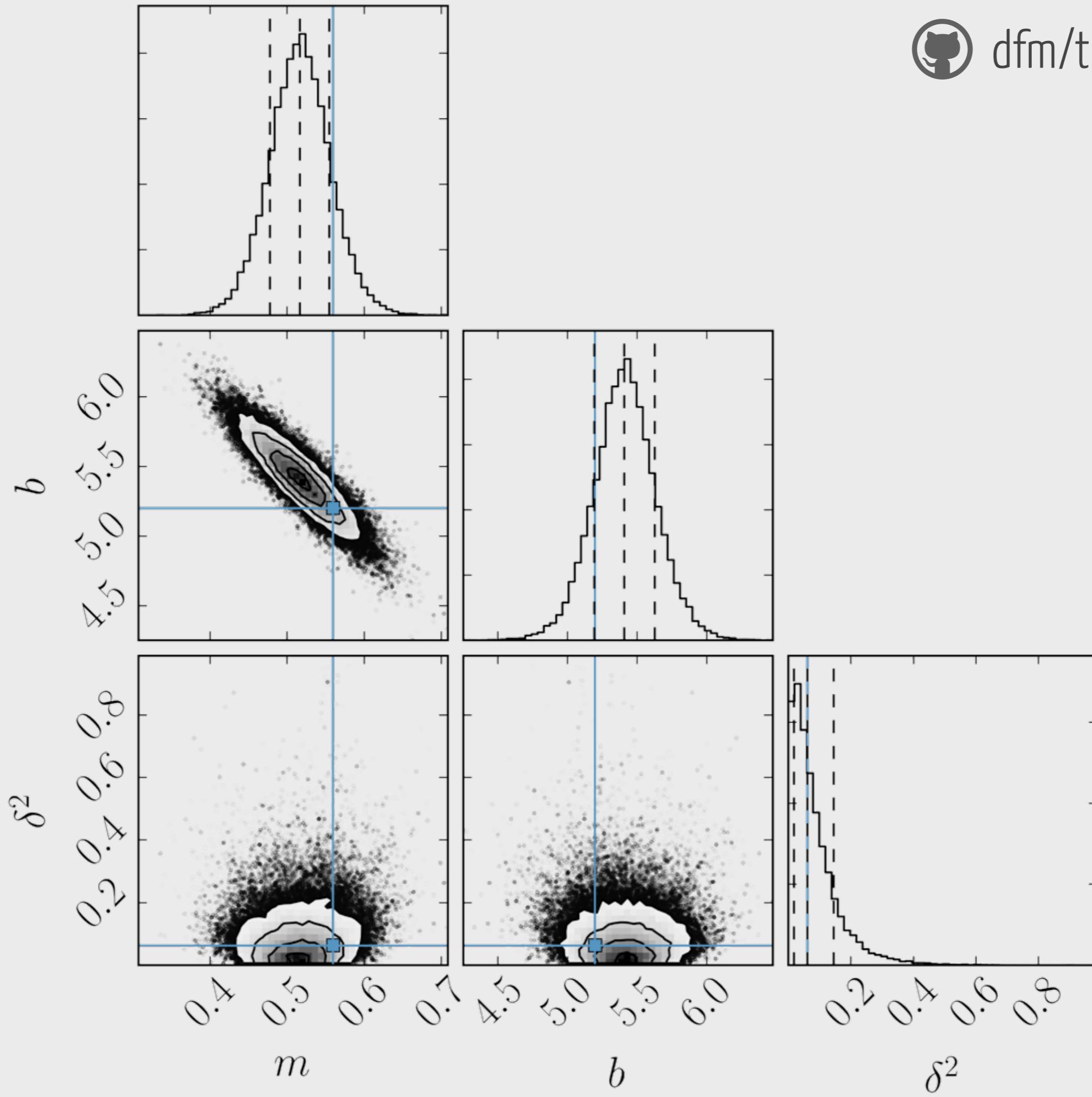
that's a lot of significant digits...



**displaying results**







# sharing results

you need a **number** for your abstract?!?

- 1 sort the samples
- 2 compute moments/quantiles  
 $(0.16, 0.5, 0.84)$

- 1 sort the samples
- 2 compute moments/quantiles  
 $(0.16, 0.5, 084)$

$$X = \bar{X}^{+\delta_+}_{-\delta_-}$$

**what does it MEAN?**

1

sort the samples

2

compute moments/quantiles  
(0.16, 0.5, 084)

$$X = \bar{X}^{+\delta_+}_{-\delta_-}$$

what does it MEAN?

3

machine readable samplings  
(including priors values)

1

sort the samples

2

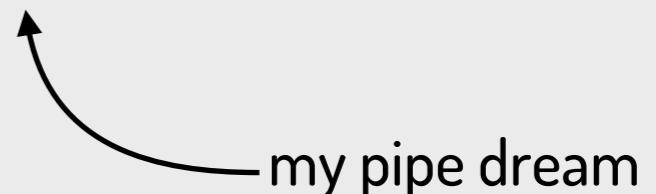
compute moments/quantiles  
(0.16, 0.5, 084)

$$X = \bar{X}^{+\delta_+}_{-\delta_-}$$

what does it MEAN?

3

machine readable samplings  
(including priors values)



my pipe dream



**Brendon Brewer's** words of wisdom...

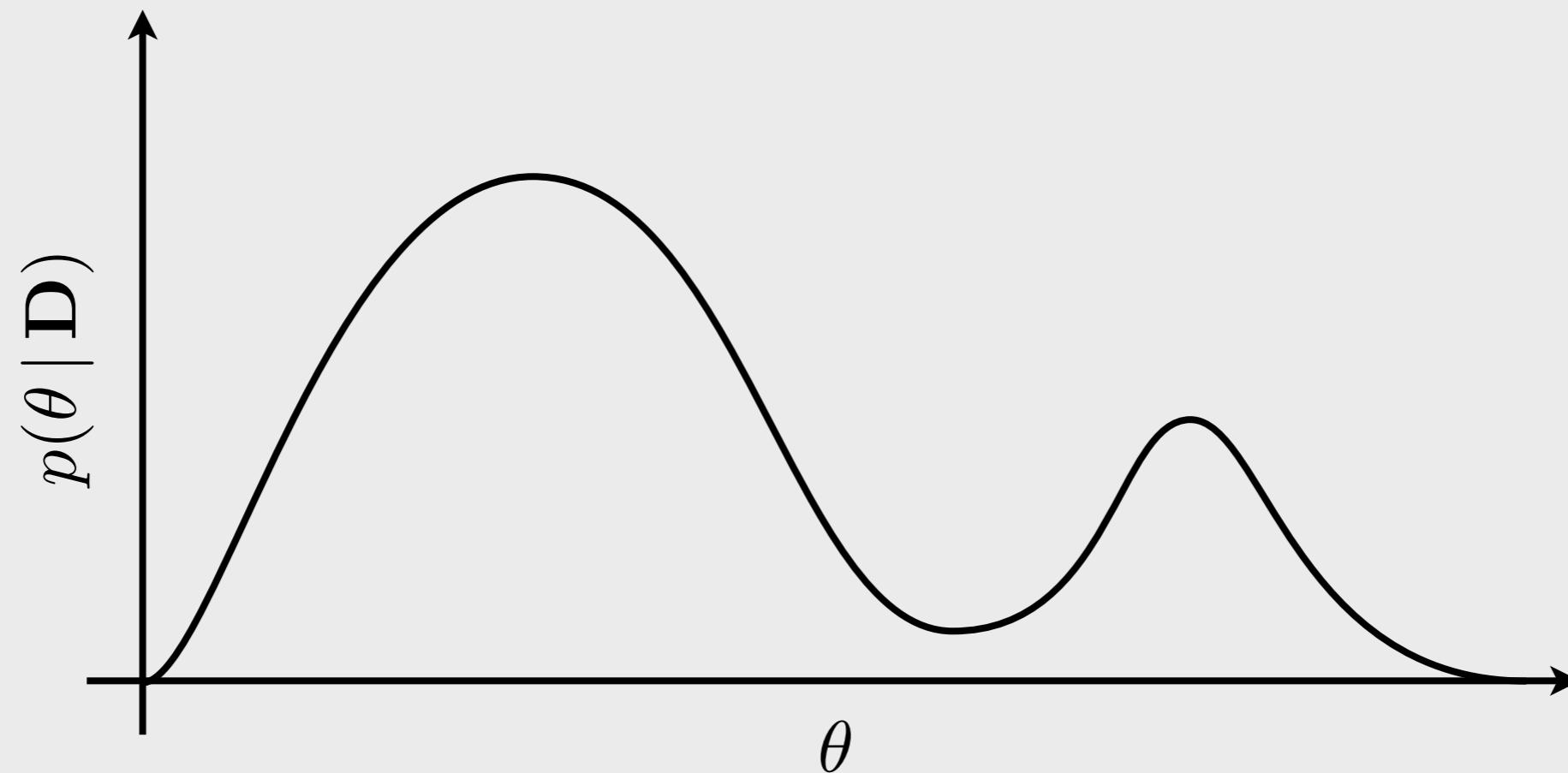
Remember:

**emcee isn't  
always  
The Right Choice™**

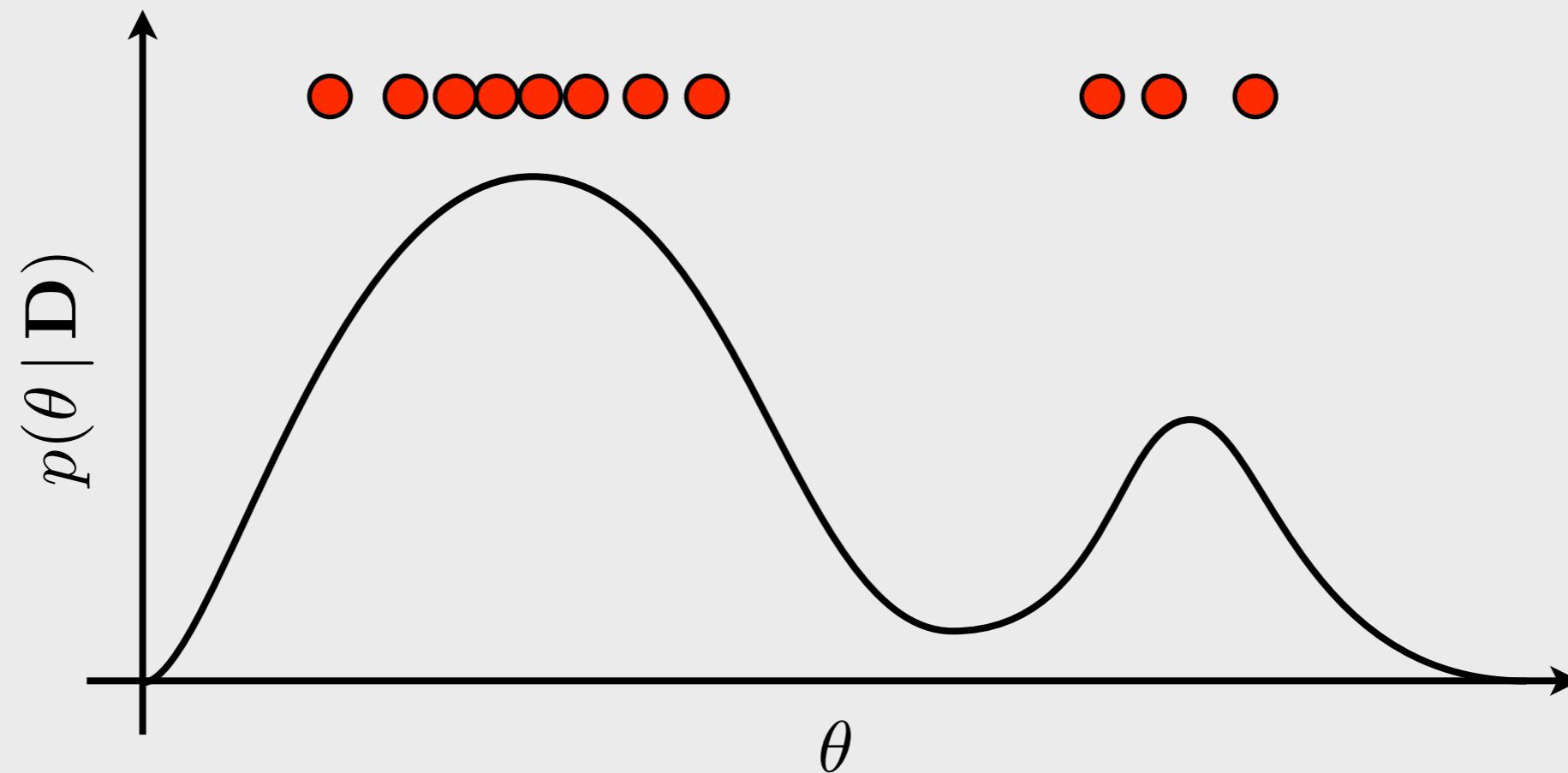


**Brendon Brewer's words of wisdom...**

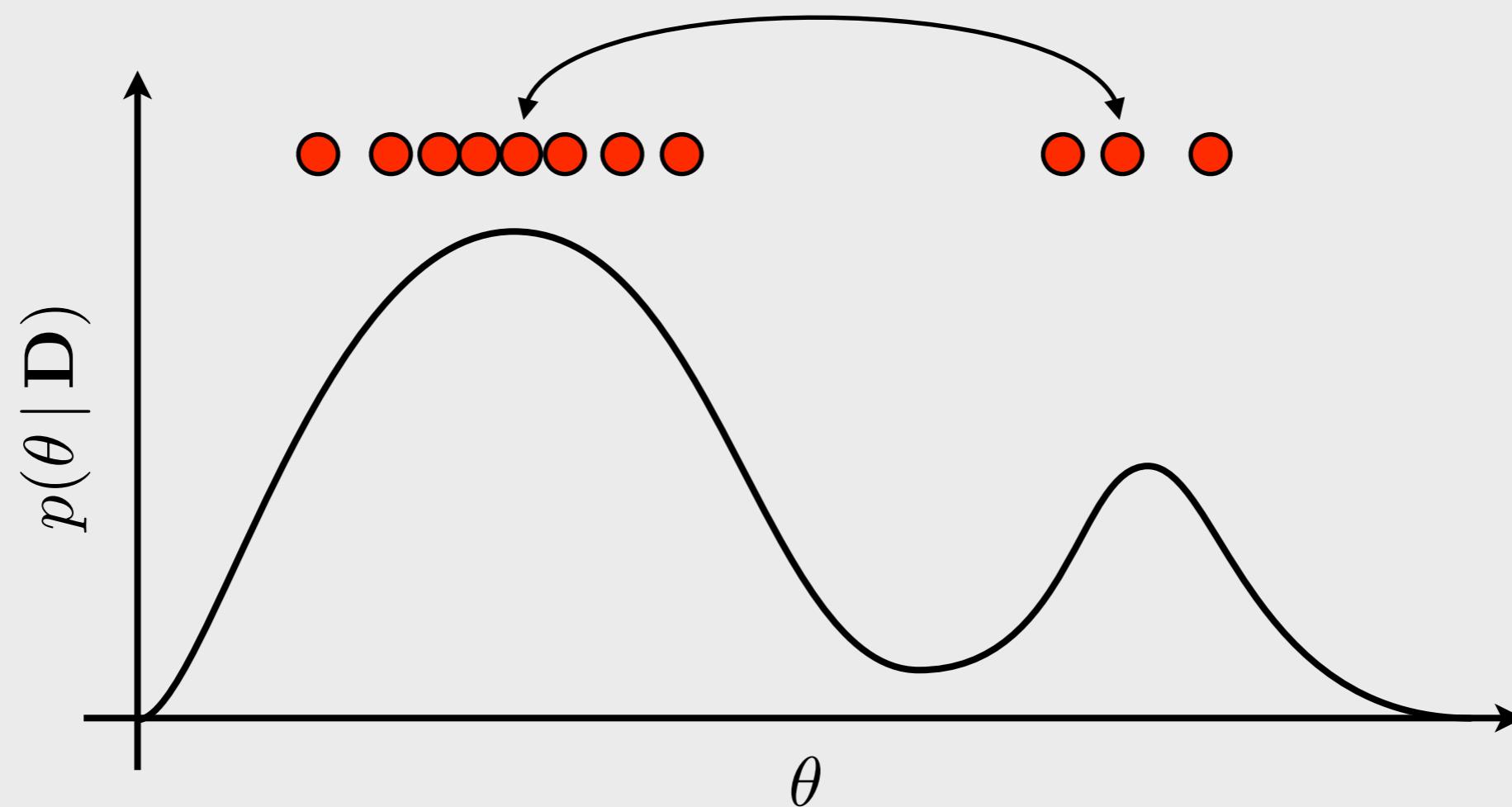
# what about **multimodal densities**?



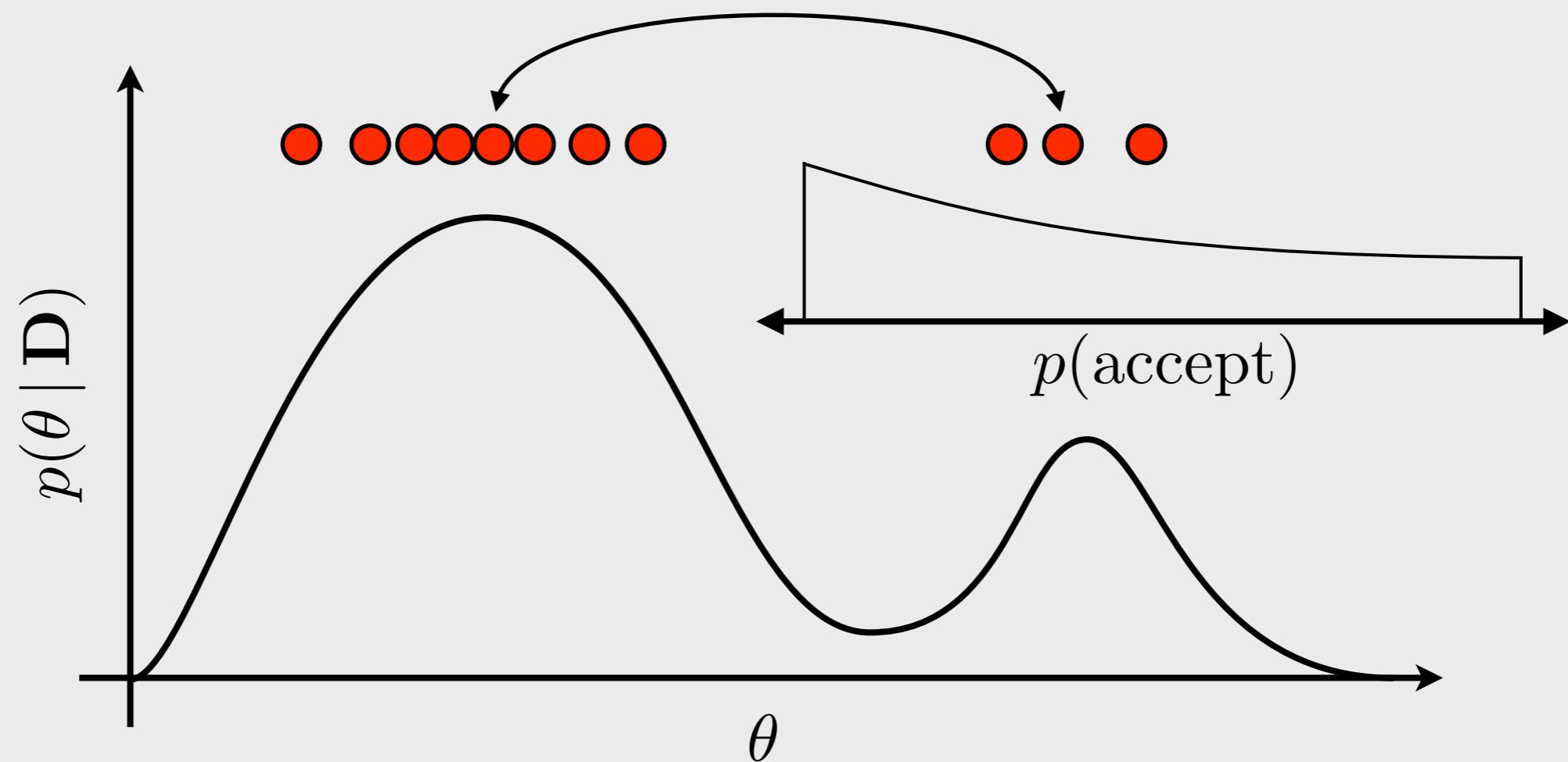
# what about multimodal densities?



# what about multimodal densities?



# what about multimodal densities?



What would happen?

$$p(\theta | \mathbf{D})$$



BOMB CAT

$\theta$

**what if we want to compare models?**

$$p(\theta | \mathbf{D}, M) = \frac{1}{Z(M)} p(\theta | M) p(\mathbf{D} | \theta, M)$$

$$Z = p(\mathbf{D} | M) = \int p(\mathbf{D}, \theta | M) d\theta$$

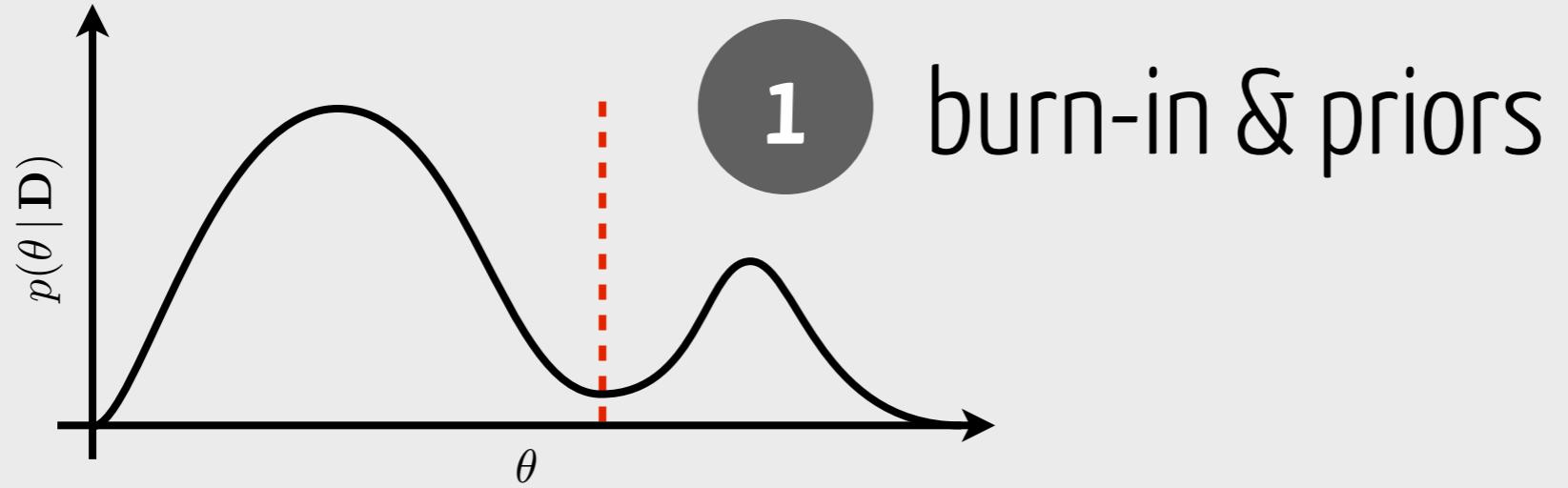
what if we want to compare models?

$$p(\theta | D, \text{BOMB CAT}) | \theta, M)$$

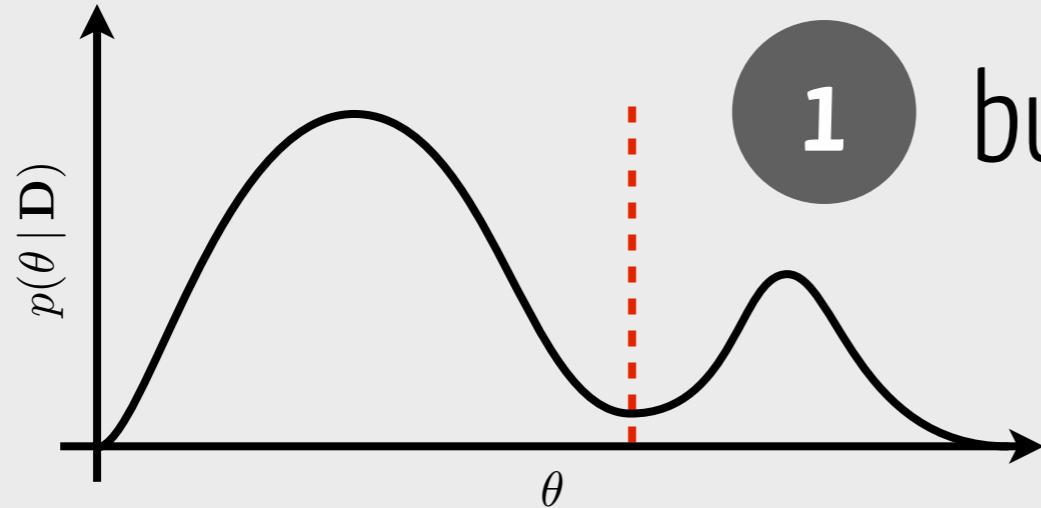


$$Z = \int \text{BOMB CAT} d\theta$$

**what to do?**



**what to do?**



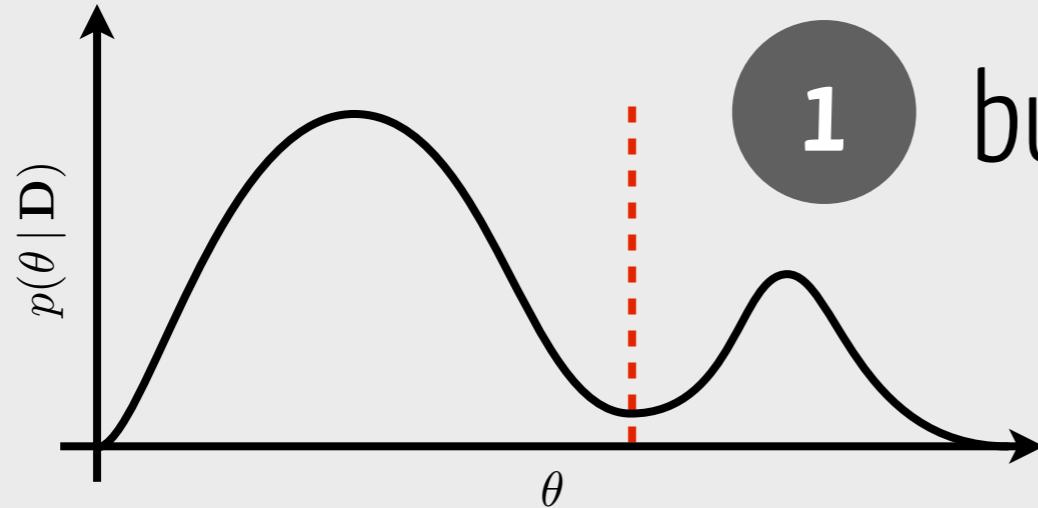
1 burn-in & priors

# what to do?

use a **different algorithm**  
(gasp!)

2

$$Z = p(\mathbf{D} \mid M) = \int p(\mathbf{D}, \theta \mid M) d\theta$$



burn-in & priors

# what to do?

use a **different algorithm**  
(gasp!)

$$Z = p(\mathbf{D} \mid M) = \int p(\mathbf{D}, \theta \mid M) d\theta$$



[github.com/eggplantbren/DNest3](https://github.com/eggplantbren/DNest3)

2

# sometimes it is useful

The screenshot shows a web browser window with the title bar "ads emcee: The MCMC Hammer". The address bar contains the URL "adsabs.harvard.edu/abs/2012arXiv1202.3665F". Below the address bar is a toolbar with various icons. The main content area displays a list of links and a table of bibliographic information.

- [Find Similar Abstracts \(with default settings below\)](#)
- [arXiv e-print](#) (arXiv:1202.3665)
- [References in the Article](#)
- [Citations to the Article \(37\) \(Citation History\)](#)
- [Refereed Citations to the Article](#)
- [Also-Read Articles \(Reads History\)](#)
- 
- [Translate This Page](#)

<b>Title:</b>	emcee: The MCMC Hammer
<b>Authors:</b>	<a href="#">Foreman-Mackey, Daniel</a> ; <a href="#">Hogg, David W.</a> ; <a href="#">Lang, Dustin</a> ; <a href="#">Goodman, Jonathan</a>
<b>Publication:</b>	eprint arXiv:1202.3665
<b>Publication Date:</b>	02/2012
<b>Origin:</b>	ARXIV
<b>Keywords:</b>	Astrophysics - Instrumentation and Methods for Astrophysics, Physics - Computational Physics, Statistics - Computation
<b>Comment:</b>	Updated to version accepted for publication in PASP
<b>Bibliographic Code:</b>	<a href="#">2012arXiv1202.3665F</a>

# emcee is **community supported**.

The screenshot shows a GitHub repository page for 'dfm/emcee'. The repository is described as 'The Python ensemble sampling toolkit for affine-invariant MCMC'. It has 11 issues, 53 forks, and 21 pull requests. The 'Code' tab is selected. The repository URL is <https://github.com/dfm/emcee>. The latest commit is by dfm, 6 days ago, with the commit hash 4ec12a4d2b. The commit message is 'Addressing initial Inprob=nan comment in #55'. Other commits listed include 'docs' (a month ago), 'document' (2 months ago), 'emcee' (6 days ago), and 'examples' (25 days ago).

PUBLIC **dfm / emcee** Pull Request Unwatch Unstar 53 Fork 21

Code Network Pull Requests 1 Issues 11 Wiki Graphs Settings

The Python ensemble sampling toolkit for affine-invariant MCMC — [Read more](#)

<http://dan.iel.fm/emcee>

Clone in Mac ZIP HTTP SSH Git Read-Only <https://github.com/dfm/emcee.git> Read+Write access

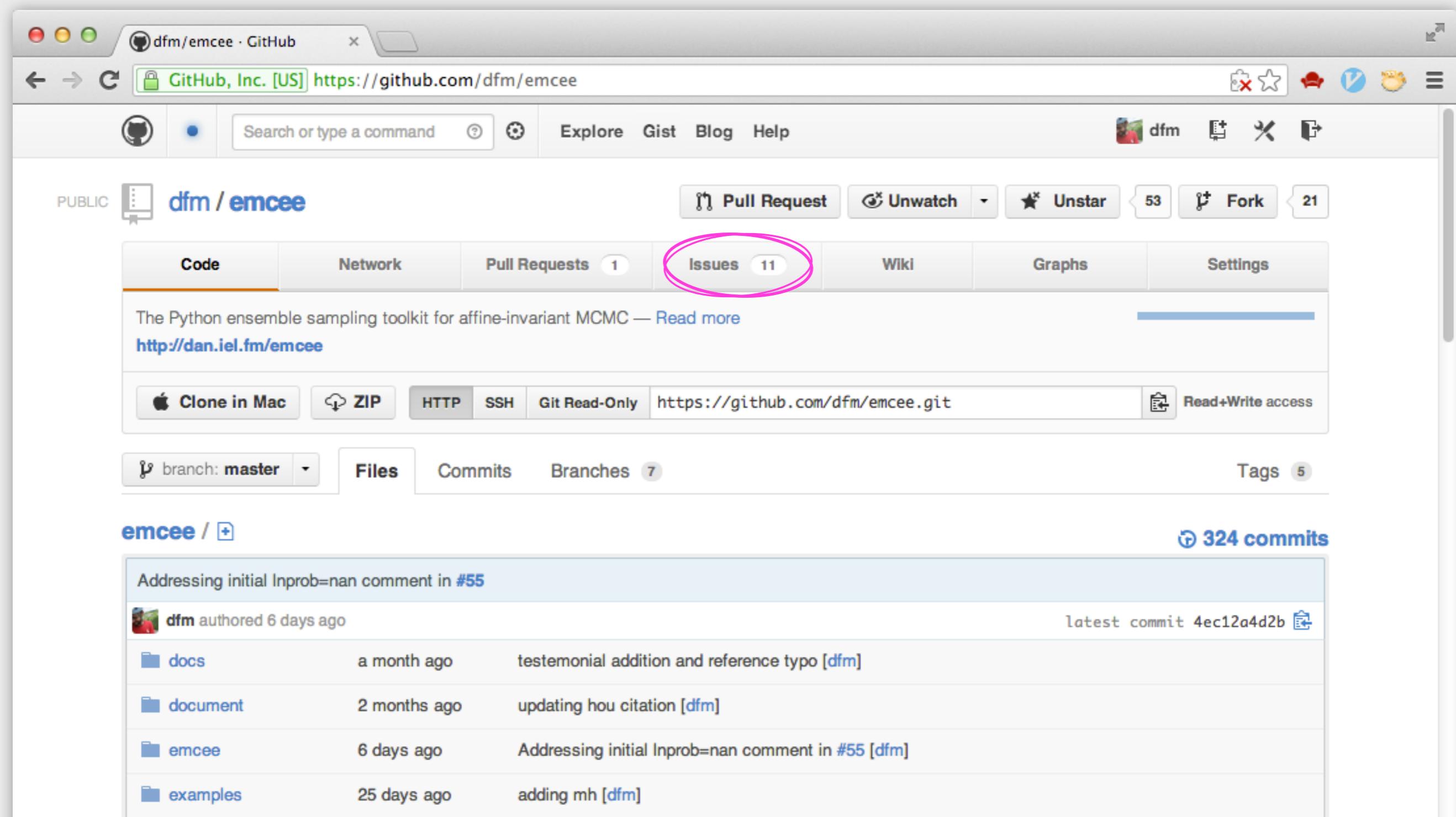
branch: master Files Commits Branches 7 Tags 5

**emcee /**  324 commits

Addressing initial Inprob=nan comment in [#55](#)

Author	Date	Message
dfm	authored 6 days ago	latest commit 4ec12a4d2b
docs	a month ago	testemorial addition and reference typo [dfm]
document	2 months ago	updating hou citation [dfm]
emcee	6 days ago	Addressing initial Inprob=nan comment in #55 [dfm]
examples	25 days ago	adding mh [dfm]

# emcee is **community supported**.



The screenshot shows a GitHub repository page for `dfm/emcee`. The top navigation bar includes links for Explore, Gist, Blog, and Help. Below the header, there are buttons for Pull Request, Unwatch, Unstar, Fork (with 53 forks), and Issues (with 11 issues). A pink oval highlights the Issues button. The main content area displays the repository's description: "The Python ensemble sampling toolkit for affine-invariant MCMC — [Read more](#)". Below the description are links for cloning the repository (Clone in Mac, ZIP, HTTP, SSH) and its Git URL (`https://github.com/dfm/emcee.git`). The repository has Read+Write access. At the bottom, there are tabs for Files, Commits, Branches (7), and Tags (5). The commit history shows 324 commits, with the latest being "Addressing initial Inprob=nan comment in #55" by `dfm` 6 days ago.

The Issues tab is highlighted with a pink oval.

**Issues** 11

The Python ensemble sampling toolkit for affine-invariant MCMC — [Read more](#)

<http://dan.iel.fm/emcee>

[Clone in Mac](#) [ZIP](#) [HTTP](#) [SSH](#) [Git Read-Only](#) <https://github.com/dfm/emcee.git> [ReadWrite access](#)

branch: master [Files](#) [Commits](#) [Branches](#) 7 [Tags](#) 5

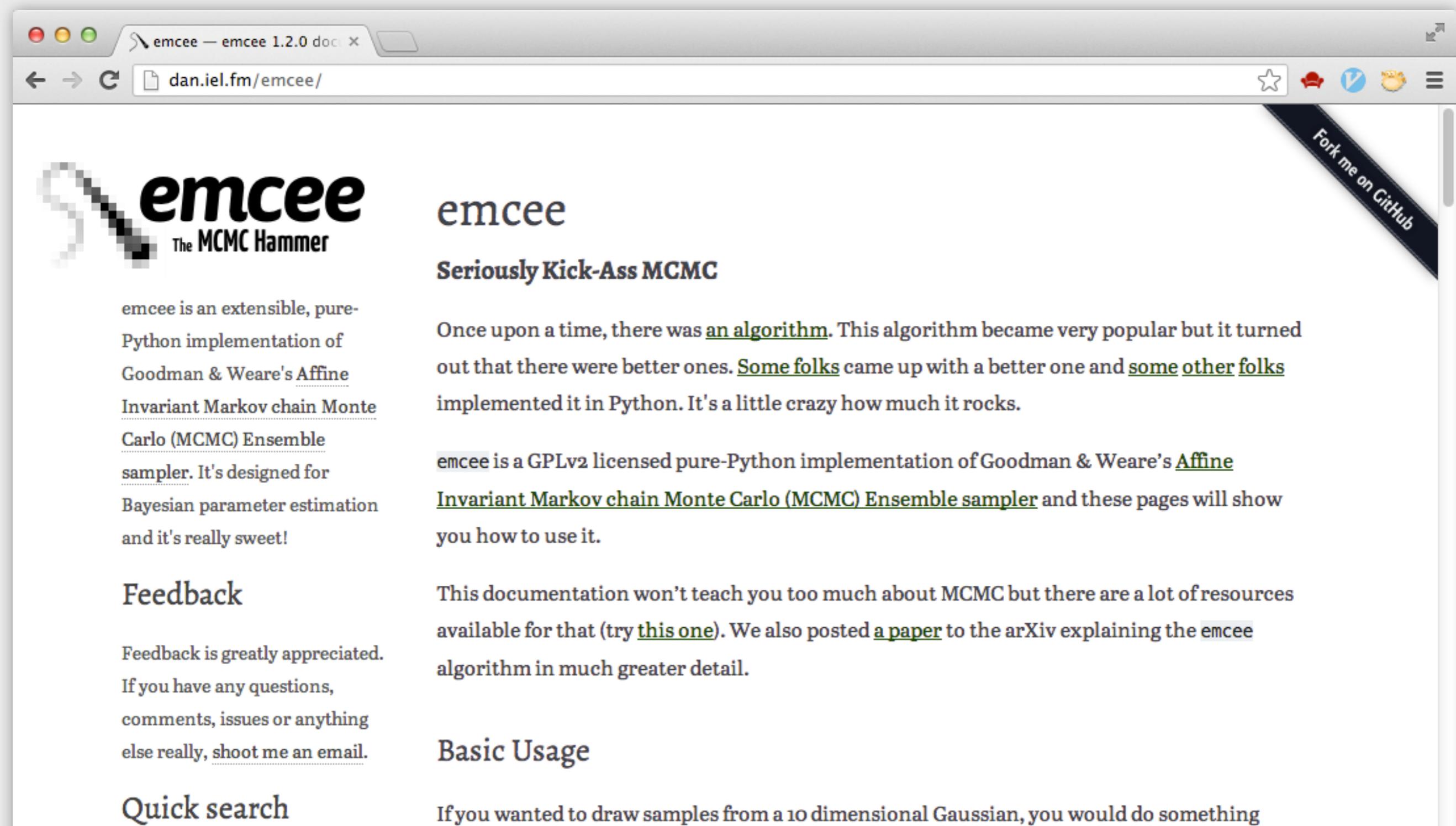
**emcee / +** [324 commits](#)

Addressing initial Inprob=nan comment in [#55](#)

 `dfm` authored 6 days ago latest commit `4ec12a4d2b` [Edit](#)

<a href="#">docs</a>	a month ago	testemorial addition and reference typo [dfm]
<a href="#">document</a>	2 months ago	updating hou citation [dfm]
<a href="#">emcee</a>	6 days ago	Addressing initial Inprob=nan comment in <a href="#">#55</a> [dfm]
<a href="#">examples</a>	25 days ago	adding mh [dfm]

# emcee has good documentation.



The screenshot shows a web browser window with the title bar "emcee — emcee 1.2.0 doc" and the URL "dan.iel.fm/emcee/". The page content is as follows:

**emcee**  
The MCMC Hammer

emcee is an extensible, pure-Python implementation of Goodman & Weare's Affine Invariant Markov chain Monte Carlo (MCMC) Ensemble sampler. It's designed for Bayesian parameter estimation and it's really sweet!

## Feedback

Feedback is greatly appreciated. If you have any questions, comments, issues or anything else really, shoot me an email.

## Basic Usage

If you wanted to draw samples from a 10 dimensional Gaussian, you would do something

**emcee**

**Seriously Kick-Ass MCMC**

Once upon a time, there was an algorithm. This algorithm became very popular but it turned out that there were better ones. Some folks came up with a better one and some other folks implemented it in Python. It's a little crazy how much it rocks.

emcee is a GPLv2 licensed pure-Python implementation of Goodman & Weare's Affine Invariant Markov chain Monte Carlo (MCMC) Ensemble sampler and these pages will show you how to use it.

This documentation won't teach you too much about MCMC but there are a lot of resources available for that (try this one). We also posted a paper to the arXiv explaining the emcee algorithm in much greater detail.

*Fork me on GitHub*

**emcee has a live support team.**



**danfm@nyu.edu**

**Alex Conley** (UC Boulder)

**Jason Davies** (Jason Davies Ltd.)

**Will Meierjurgen Farr** (Northwestern)

**David W. Hogg** (NYU)

**Dustin Lang** (CMU)

**Phil Marshall** (Oxford)

**Ilya Pashchenko** (ASC LPI, Moscow)

**Adrian Price-Whelan** (Columbia)

**Jeremy Sanders** (Cambridge)

**Joe Zuntz** (Oxford)

**Eric Agol** (UW)

**Jo Bovy** (IAS)

**Jacqueline Chen** (MIT)

**John Gizis** (Delaware)

**Jonathan Goodman** (NYU)

**Marius Millea** (UC Davis)

**Jennifer Piscionere** (Vanderbilt)

**contributors**

**take home.**

your model:

$$p(\mathbf{D} \mid \theta, \alpha)$$

# take home.

your model:

$$p(\mathbf{D} | \theta, \alpha)$$

$$p(\mathbf{D} | \theta) \propto \int p(\alpha) p(\mathbf{D} | \theta, \alpha) d\alpha$$



# take home.

your model:

$$p(\mathbf{D} | \theta, \alpha)$$

$$p(\mathbf{D} | \theta) \propto \int p(\alpha) p(\mathbf{D} | \theta, \alpha) d\alpha$$



$$\theta =$$



**Alex Conley** (UC Boulder)

**Jason Davies** (Jason Davies Ltd.)

**Will Meierjurgen Farr** (Northwestern)

**David W. Hogg** (NYU)

**Dustin Lang** (CMU)

**Phil Marshall** (Oxford)

**Ilya Pashchenko** (ASC LPI, Moscow)

**Adrian Price-Whelan** (Columbia)

**Jeremy Sanders** (Cambridge)

**Joe Zuntz** (Oxford)

**Eric Agol** (UW)

**Jo Bovy** (IAS)

**Jacqueline Chen** (MIT)

**John Gizis** (Delaware)

**Jonathan Goodman** (NYU)

**Marius Millea** (UC Davis)

**Jennifer Piscionere** (Vanderbilt)

**thanks!**