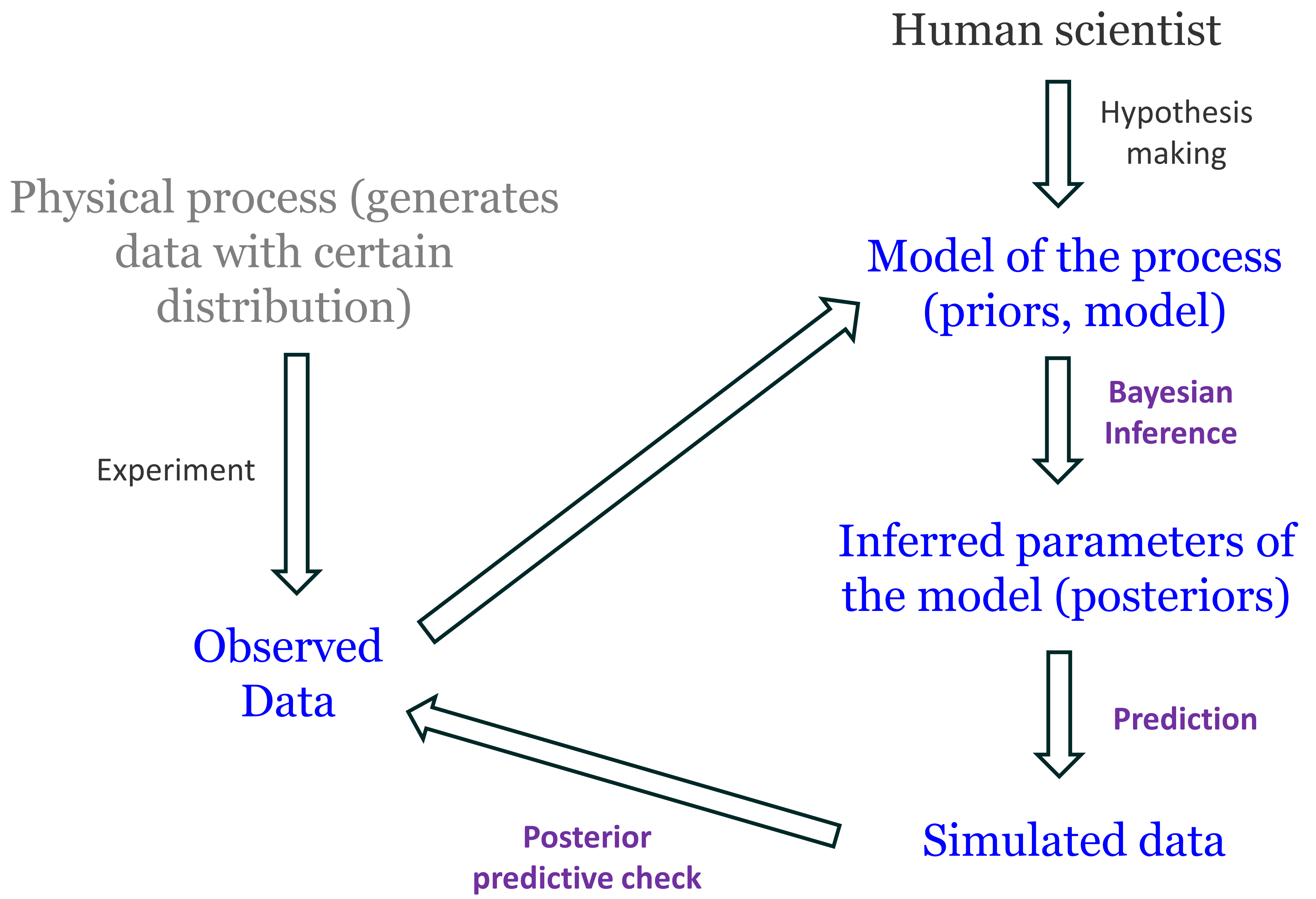
Lecture 20: Bayesian Inference

Instructor: Sergei V. Kalinin

How do we use Bayesian Methods in Science?



Probability of data is generally intractable

- Prior information $p(\theta)$ on parameters θ
- Likelihood of data given parameter values $f(x|\ \theta)$

Problem for physics/theory

Outside knowledge

$$p(\theta|x) = \frac{\boxed{f(x|\theta)}\boxed{p(\theta)}}{\boxed{f(x)}}$$

Problem for computation

$$f(x) = \int_{-\infty}^{\infty} f(x|\theta)p(\theta)d\theta$$

To use Bayesian methods, we need to be able to evaluate the denominator, which is the integral of the numerator over the parameter space. In general, this integral is very hard to evaluate.

Solution: Markov Chain Monte Carlo

We don't need to evaluate any integral, *we just sample from the distribution many times* (e.g., 50K times) and find (estimate) the posterior mean, middle 95%, etc., from that.

Metropolis-Hastings:

- An algorithm that generates a sequence $\{\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, \ldots\}$ from a Markov Chain whose stationary distribution is $\pi(\theta)$ (i.e., the posterior distribution)
- Fast computers and recognition of this algorithm has allowed Bayesian estimation to develop.

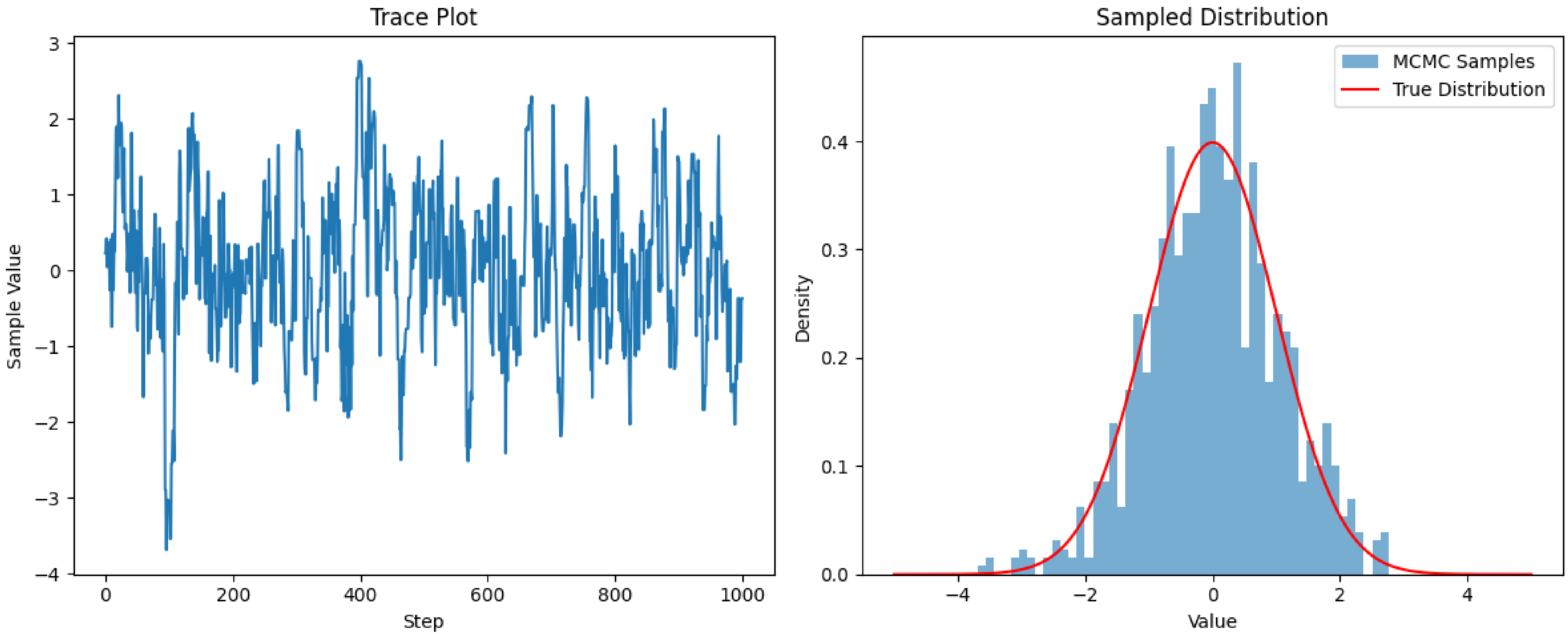
From Jeff Witmer, Introduction to Bayesian Statistics, slides 24 February 2016

Metropolis-Hastings algorithm

- Initial value $\theta^{(o)}$ to start the Markov Chain
- Propose new value
- Accepted value: θ'

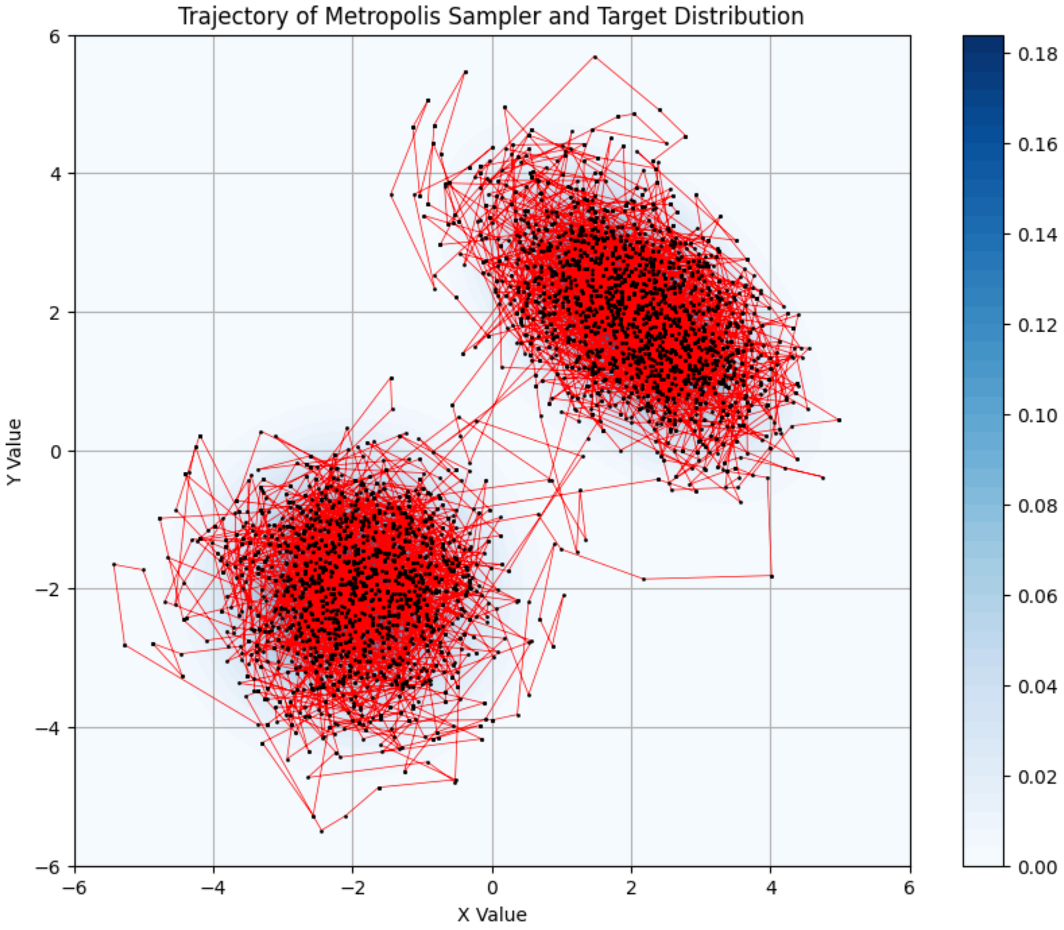
$$\theta^{(1)} = \begin{cases} \theta' \text{ with probability } \alpha \\ \theta^{(0)} \text{ with probability } 1-\alpha \end{cases} \qquad \text{where } \alpha = \min\left(1, \frac{\pi(\theta')}{\pi(\theta^{(0)})}\right)$$

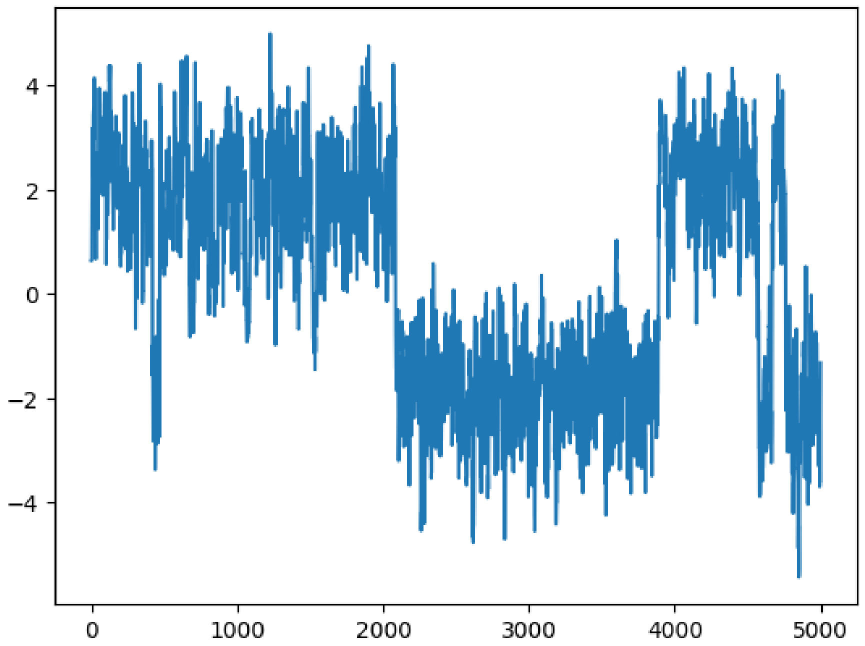
Sampling 1D Gaussian



We can calculate any function of the posterior by summing over the trace

Sampling 2D Gaussian





MCMC Solvers:

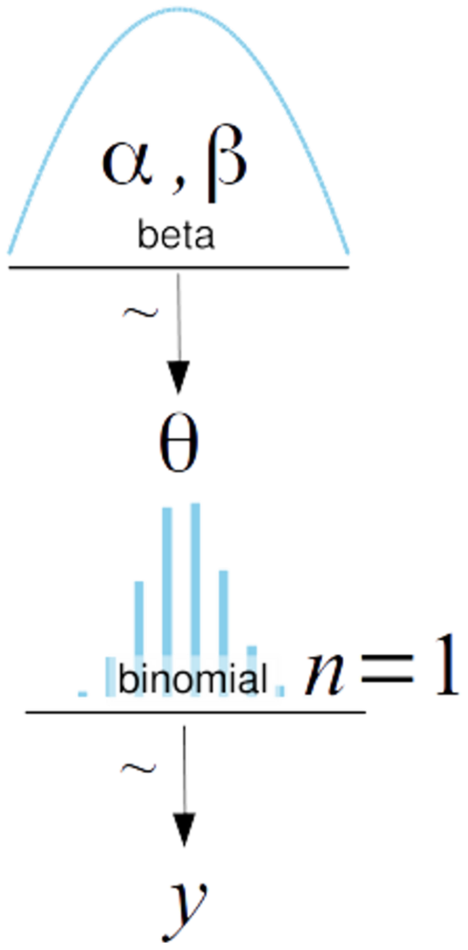
- BUGS – Bayes Using Gibbs Sampling
- JAGS – Just Another Gibbs Sampler
- Stan – uses Hamiltonian Monte Carlo
- NUTS – No U-Turn Sampler

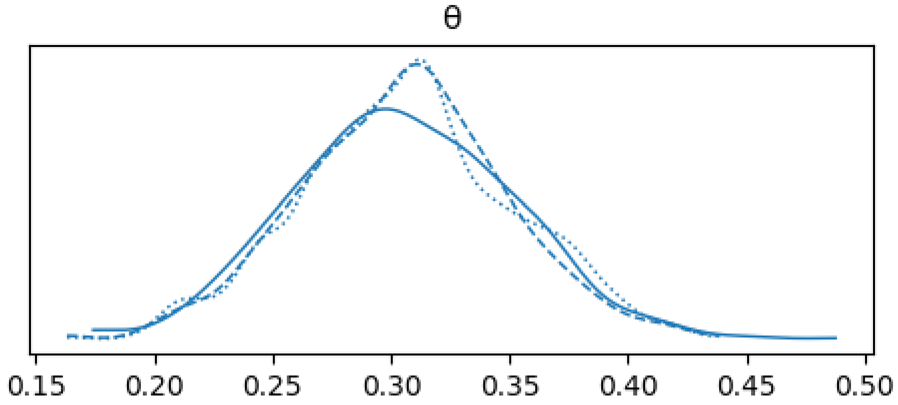
Let's toss a coin!

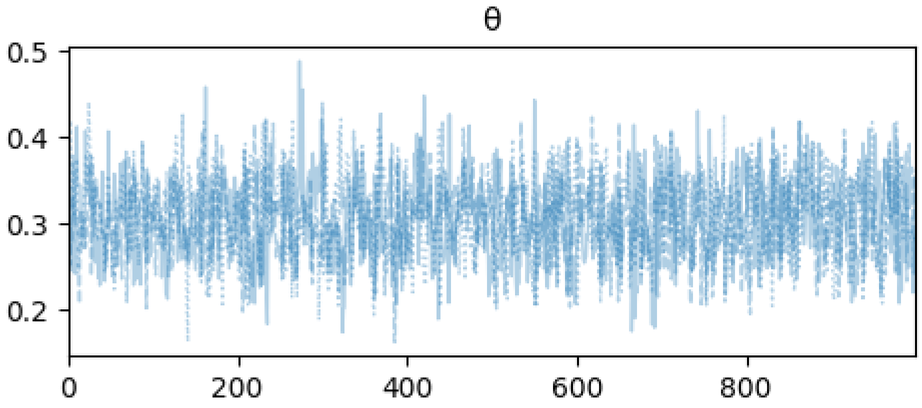
Colab: 26_PYMC_BayesianEZ.ipynb

Let's toss a coin (with a PYMC)!

```python
with pm.Model() as our_first_model:
    θ = pm.Beta('θ', alpha=1., beta=1.)
    y = pm.Bernoulli('y', p=θ, observed=data)
    trace = pm.sample(1000, random_seed=123, chains = 3)
```



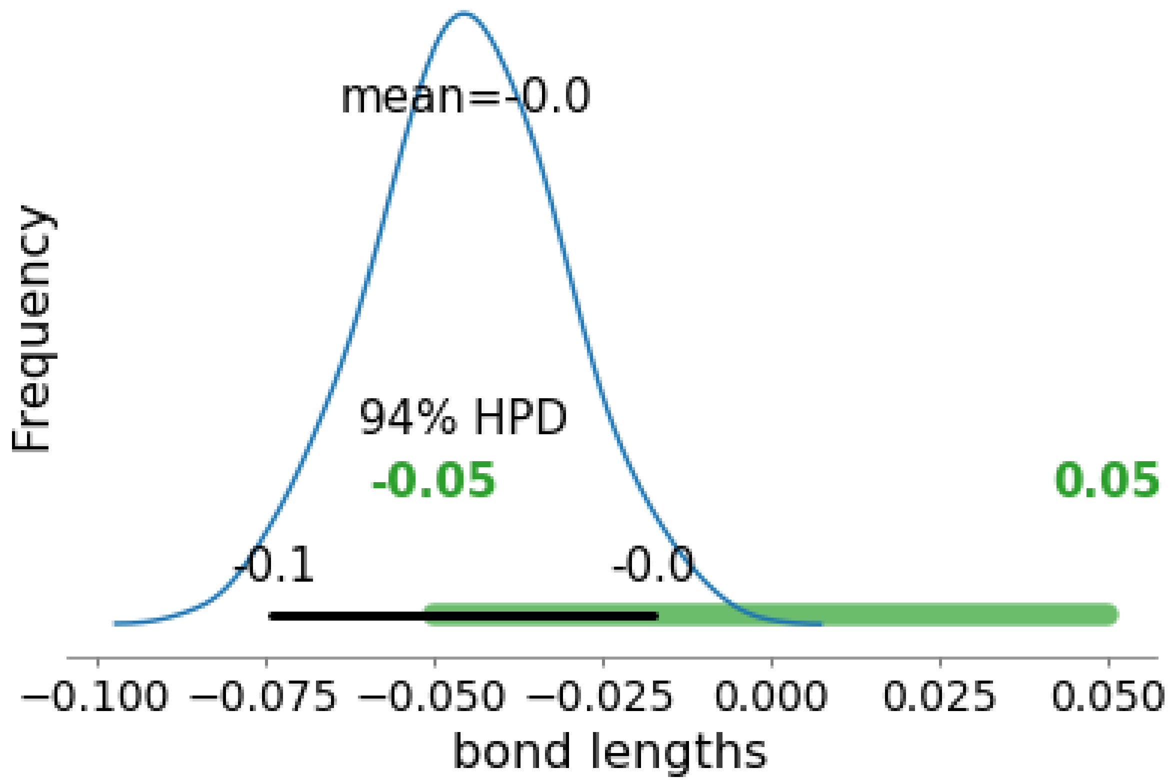




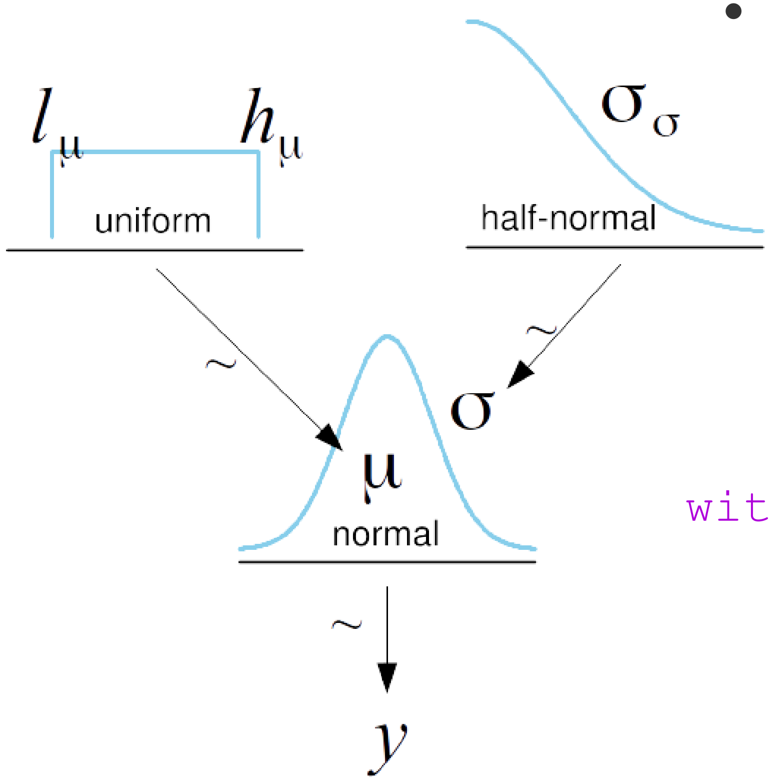
Colab: 19_PYMC_BayesianEZ.ipynb

Decision making: ROPE

- We construct an interval (ROPE) around the hypothesis and a decision can be made by comparing the intervals of HDI (94% credible interval) and ROPE.
- Decision rules for different criteria are listed in “Bayesian Analysis with Python” by Osvaldo Martin



Learning Gaussian Distribution

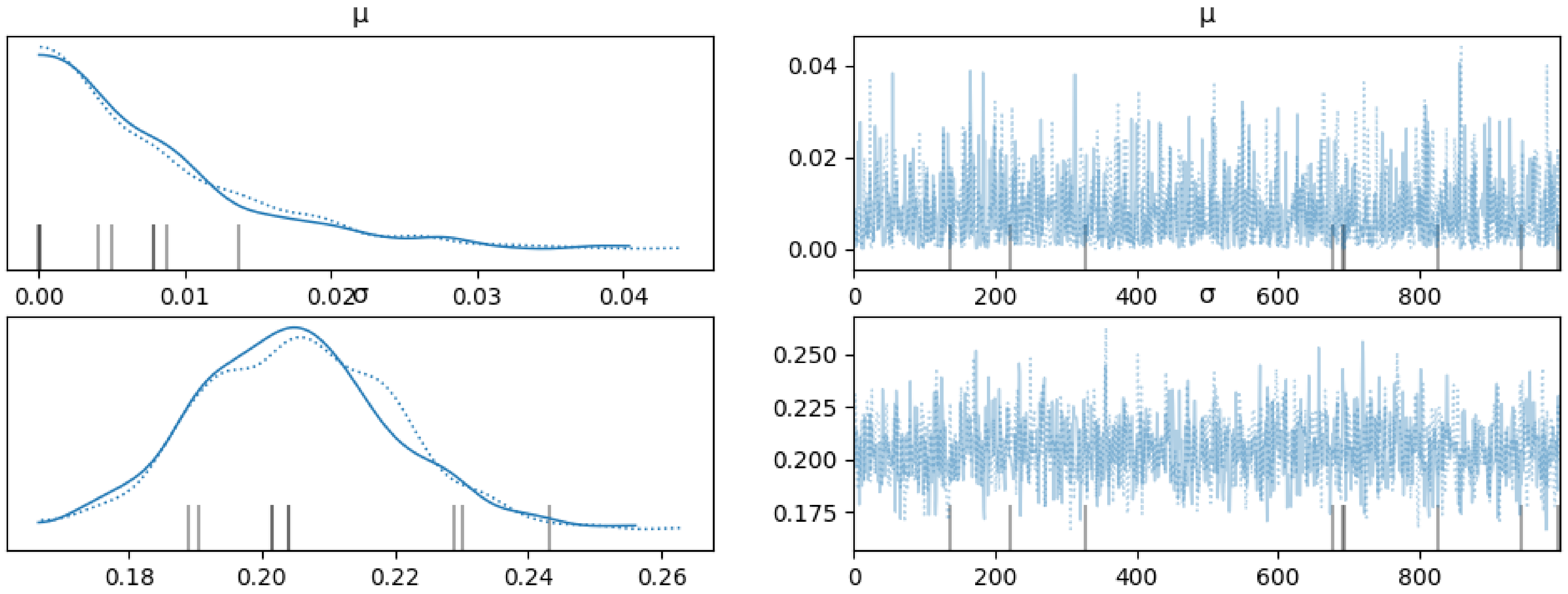


- Imagine that we have some observational data
- Can we say which distribution has it come from?

Define Bayesian model!

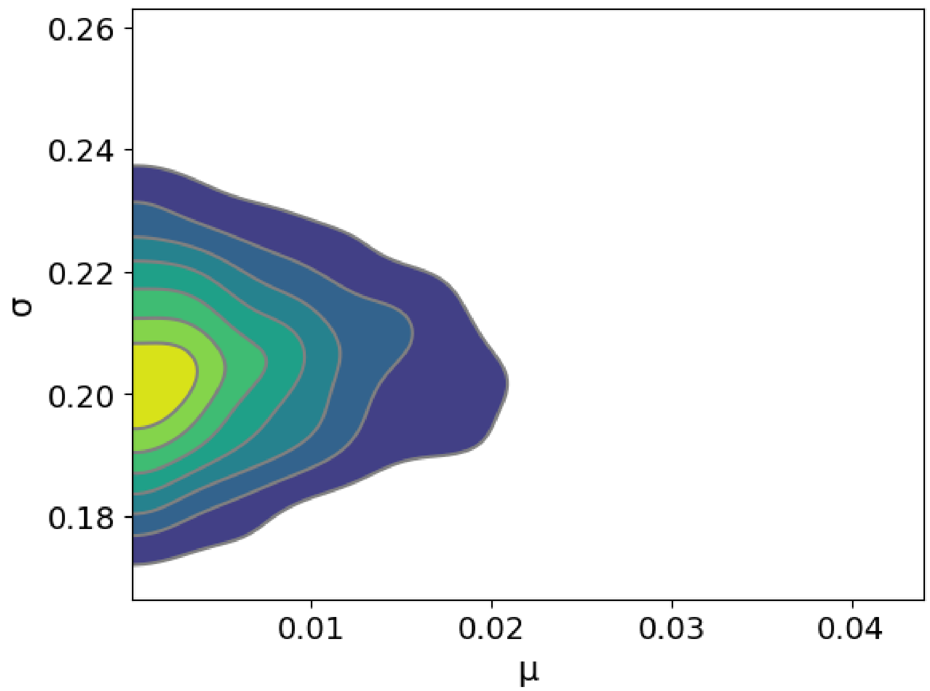
```
with pm.Model() as model_g:
    μ = pm.Uniform('μ', lower=0, upper=2)
    σ = pm.HalfNormal('σ', sigma=1)
    y = pm.Normal('y', mu=μ, sigma=σ, observed=arr)
    idata_g = pm.sample(1000)
```

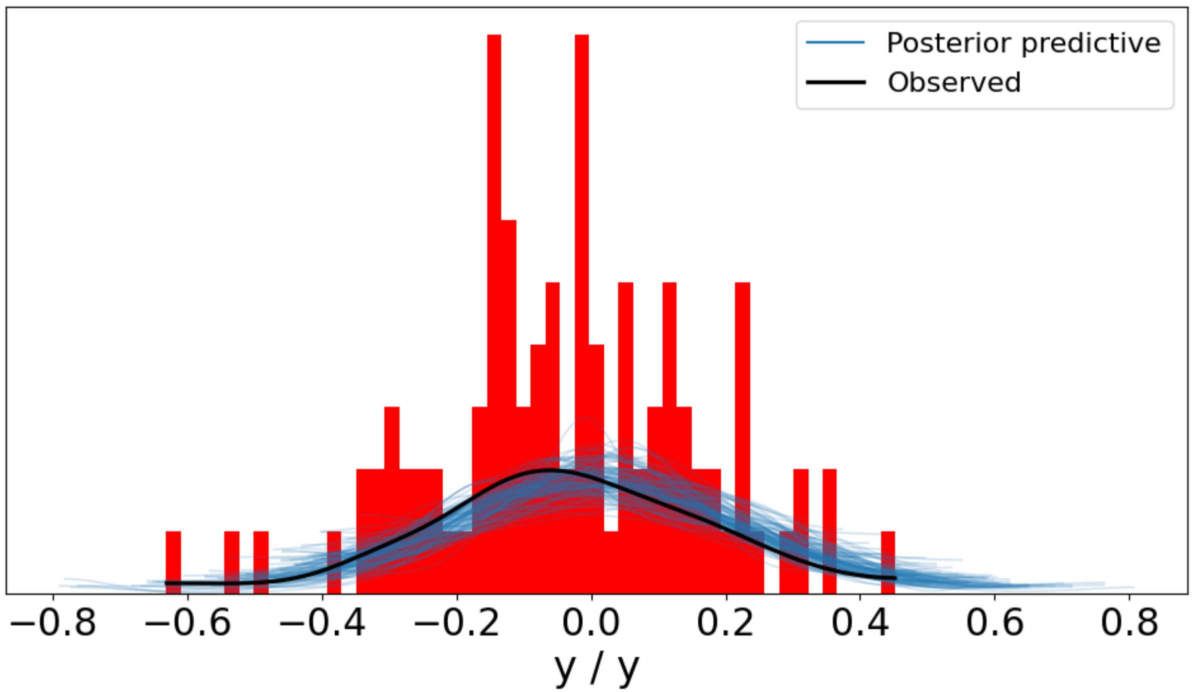
Presto!



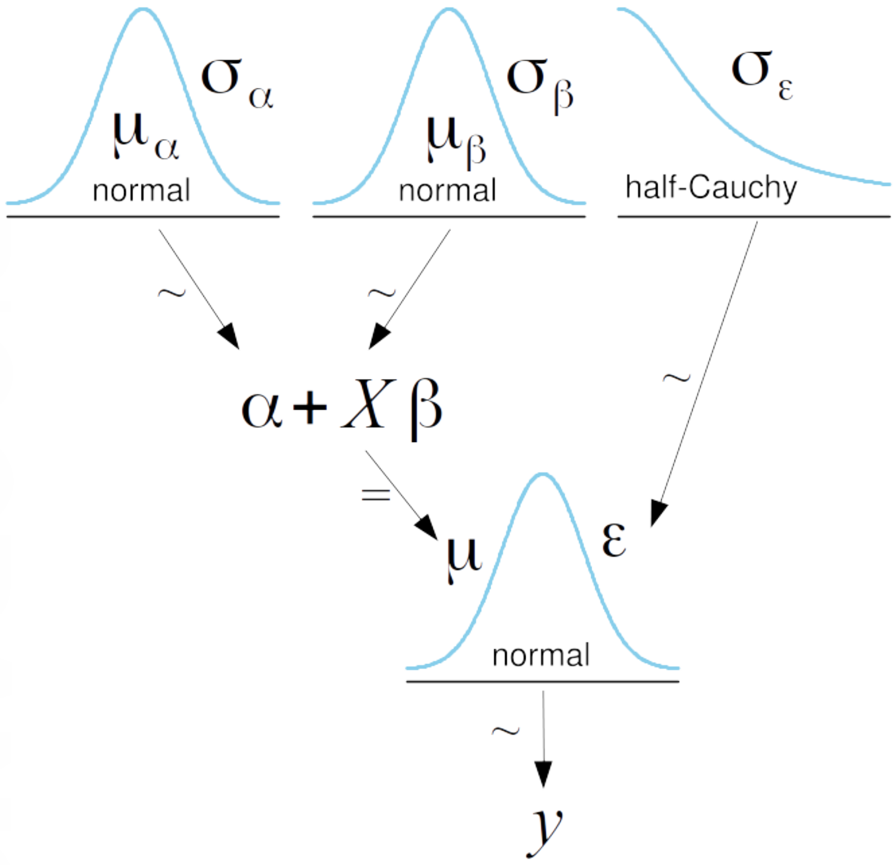
μ
0.00
0.01
0.02
0.03
0.04
σ
0.18
0.20
0.22
0.24
0.26
μ
0.04
0.02
0.00
0
200
400
600
800
σ
0.250
0.225
0.200
0.175
0
200
400
600
800

Presto!


0.26
0.24
0.22
0.20
0.18
σ
0.01
0.02
0.03
0.04
μ


Posterior predictive
Observed
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
y / y

Linear regression Bayesian style



- Imagine that we have some observational data
- Can we fit it by linear function?

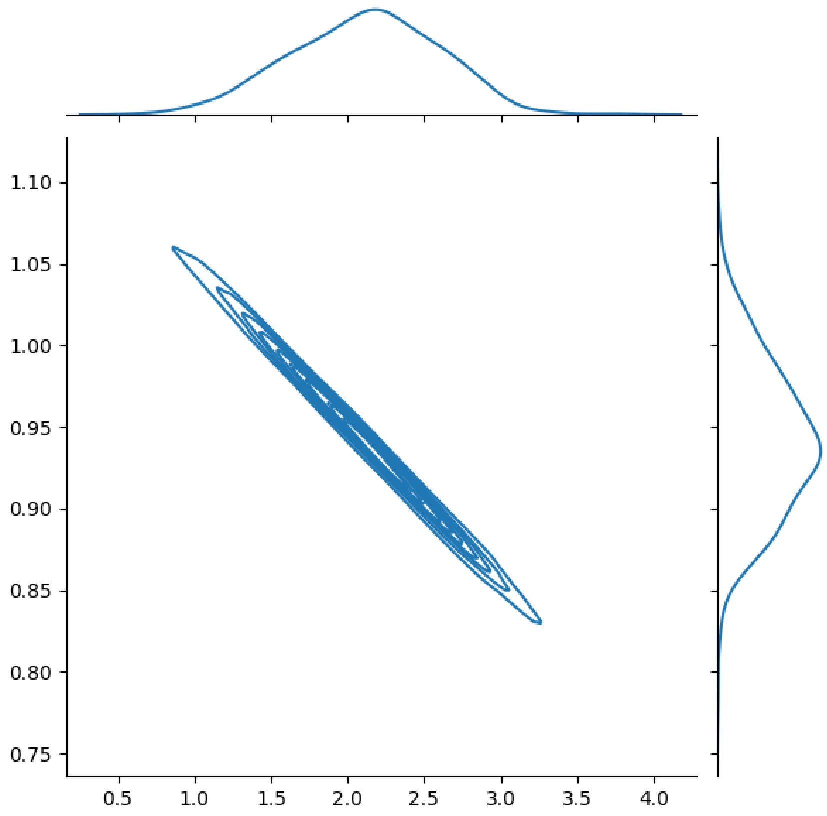
Define Bayesian model!

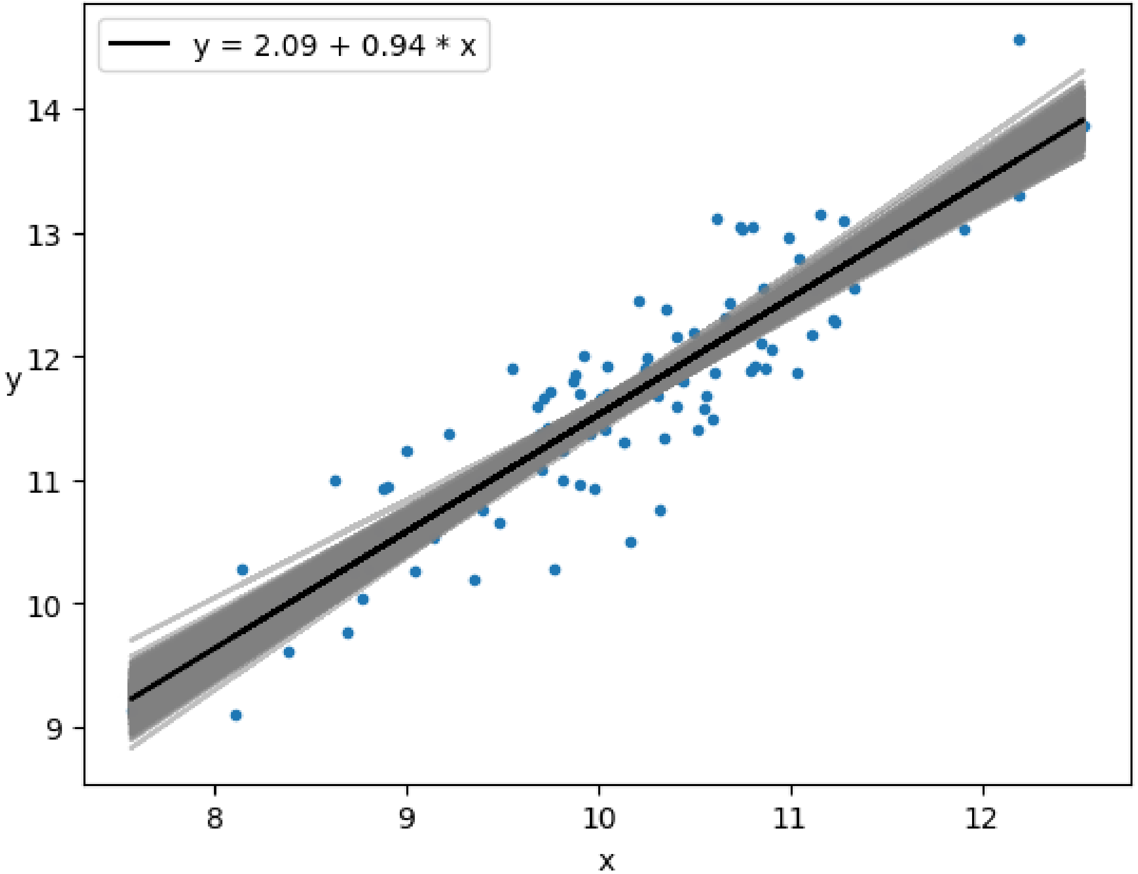
```
with pm.Model() as model_g:
    α = pm.Normal('α', mu=0, sigma=10)
    β = pm.Normal('β', mu=0, sigma=1)
    ϵ = pm.HalfCauchy('ϵ', 5)

    μ = pm.Deterministic('μ', α + β * x)
    y_pred = pm.Normal('y_pred', mu=μ,
                       sigma=ϵ, observed=y)

    idata_g = pm.sample(2000, tune=2000)
```

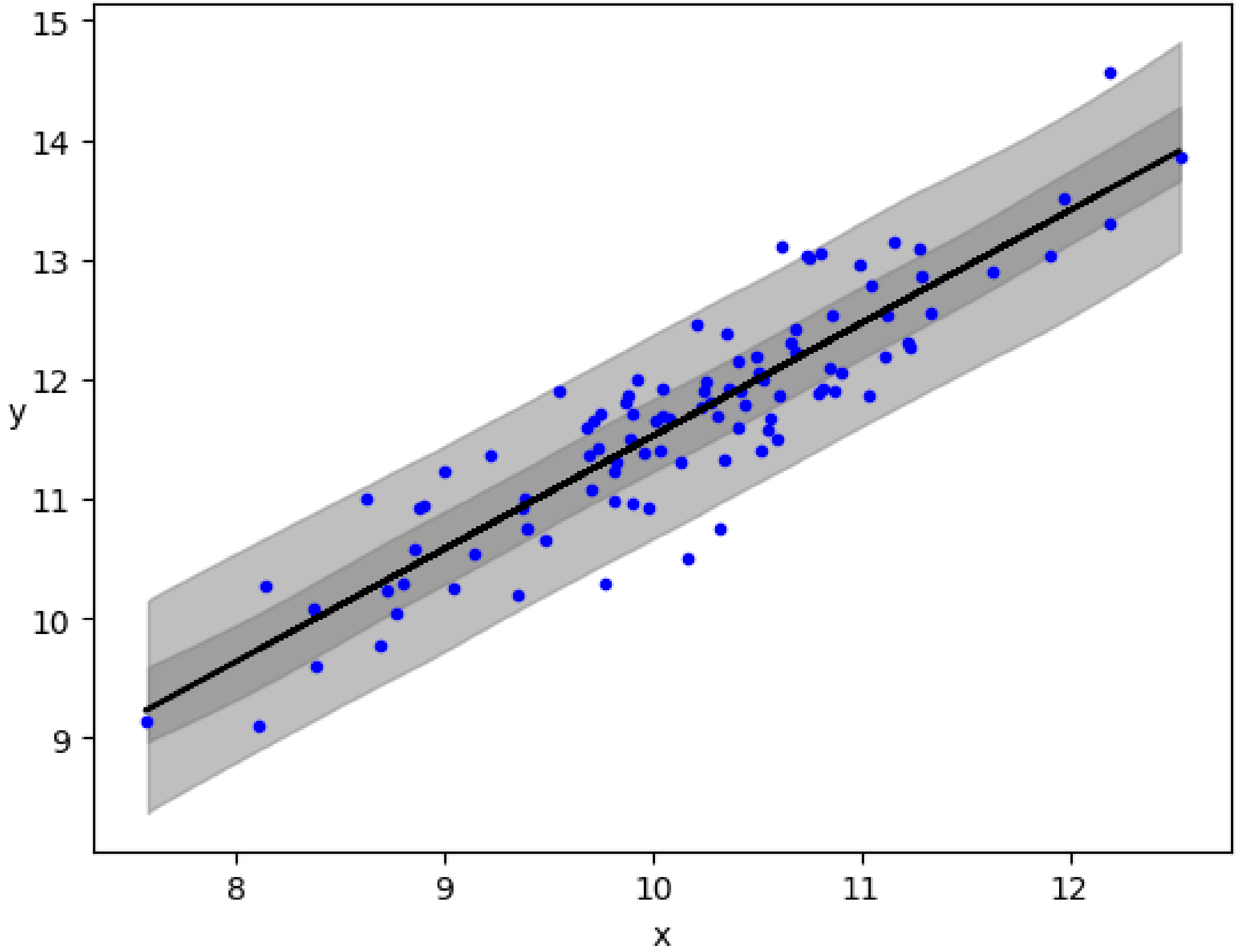
The outputs?





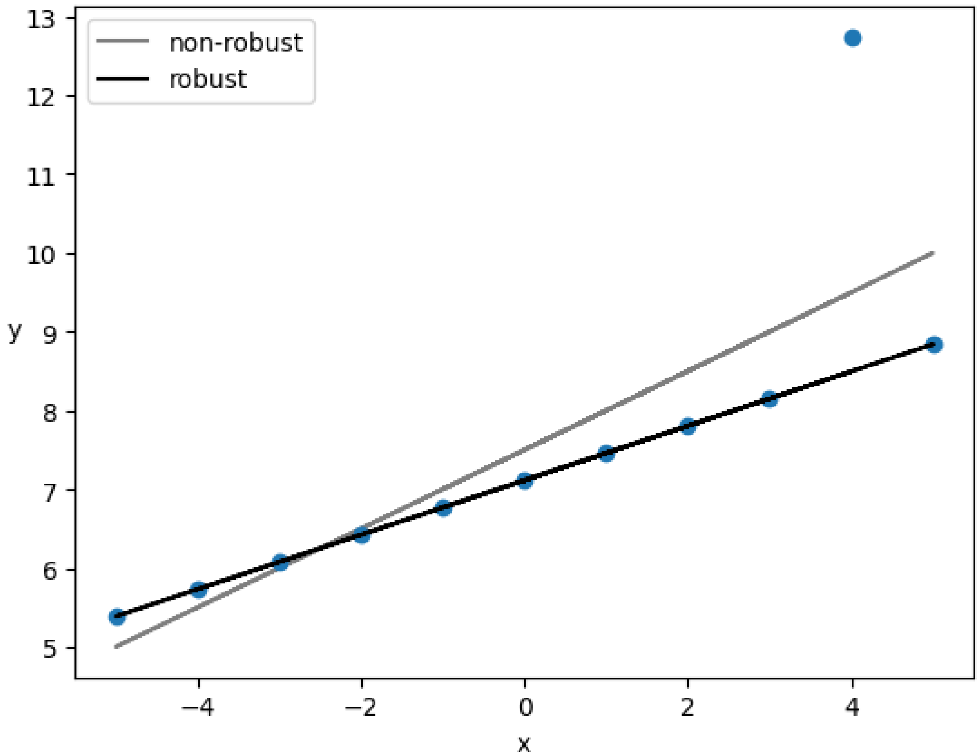
- The output of the Bayesian model will be draws for the slope and offset: families of lines consistent with data
- We can analyze their joint distribution
- And plot these lines

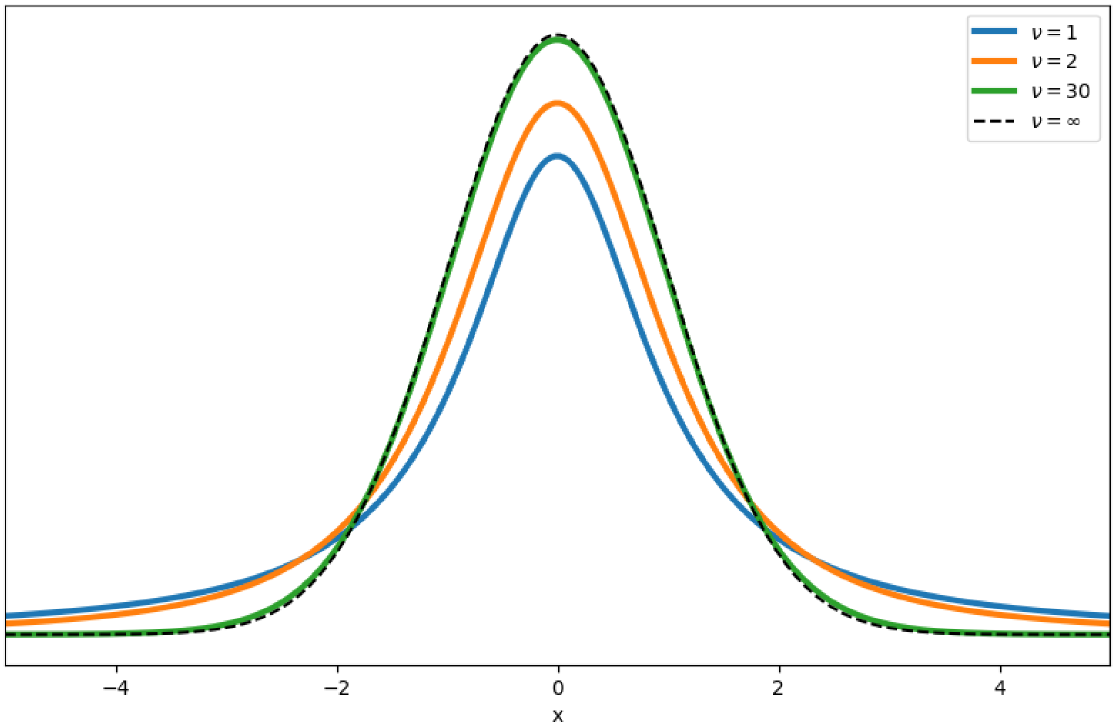
How do we decide if the “data is reasonable”?



- That’s what the posterior predictive checks are for!

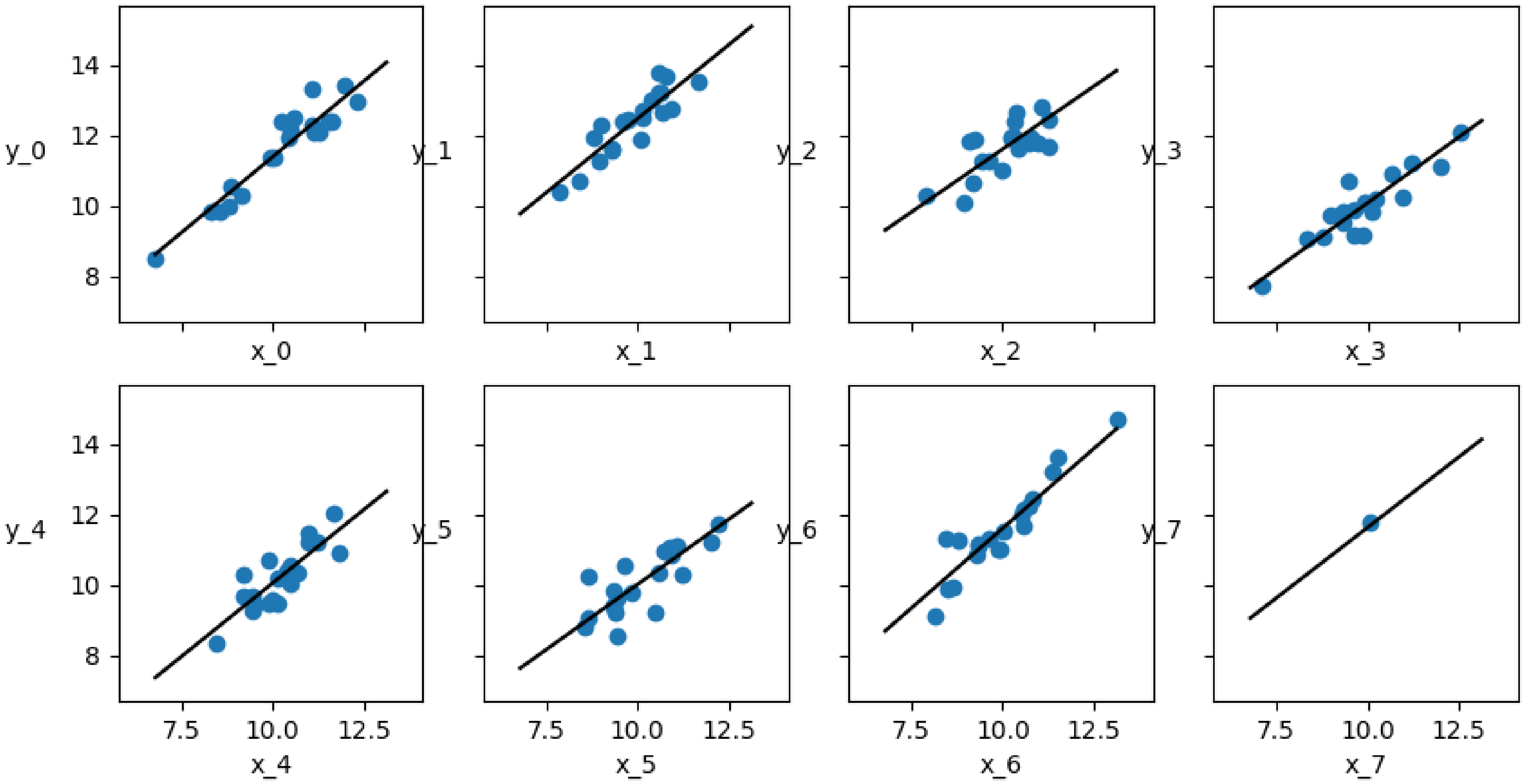
Bayesian methods for “impossible” problems





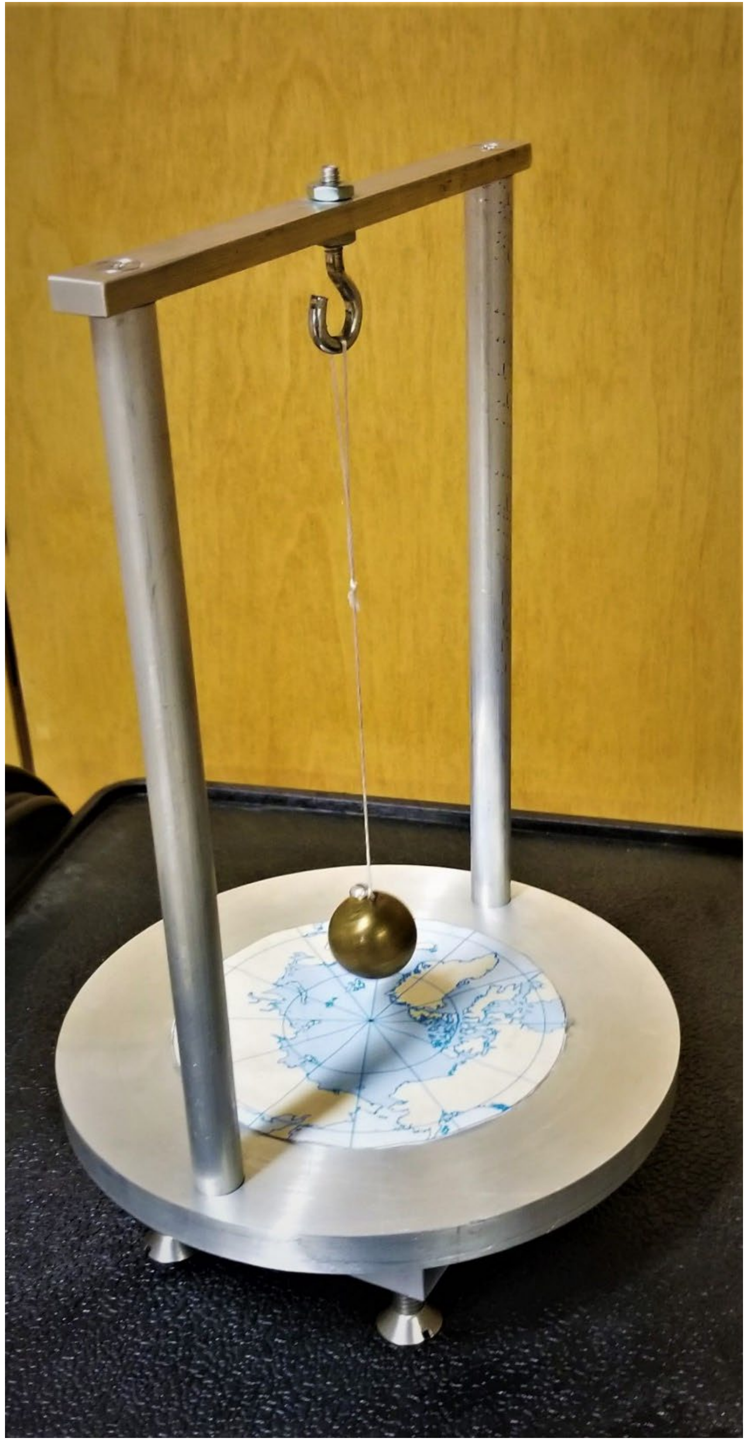

```
with pm.Model() as model_t:
    α = pm.Normal('α', mu=y_3.mean(), sigma=1)
    β = pm.Normal('β', mu=0, sigma=1)
    ϵ = pm.HalfNormal('ϵ', 5)
    ν_ = pm.Exponential('ν_', 1/29)
    ν = pm.Deterministic('ν', ν_ + 1)
    y_pred = pm.StudentT('y_pred', mu=α + β * x_3,sigma=ϵ, nu=ν, observed=y_3)
    idata_t = pm.sample(2000)
```

Bayesian methods for “impossible” problems



Least Square Fits

- We have some observational data
- And physical model expressed by formula
- All we have to do is to fit formula to the observations



Theory: For small angles, a simple pendulum follows a simple harmonic motion, where the period of a full swing back and forth (the time for one complete cycle) is given by the formula:

T=2π Sqrt(L/g)

- *T* is the period (time for one complete cycle, in seconds).
- *L* is the length of the pendulum
- *g* is the acceleration due to gravity (in *m/s*2).

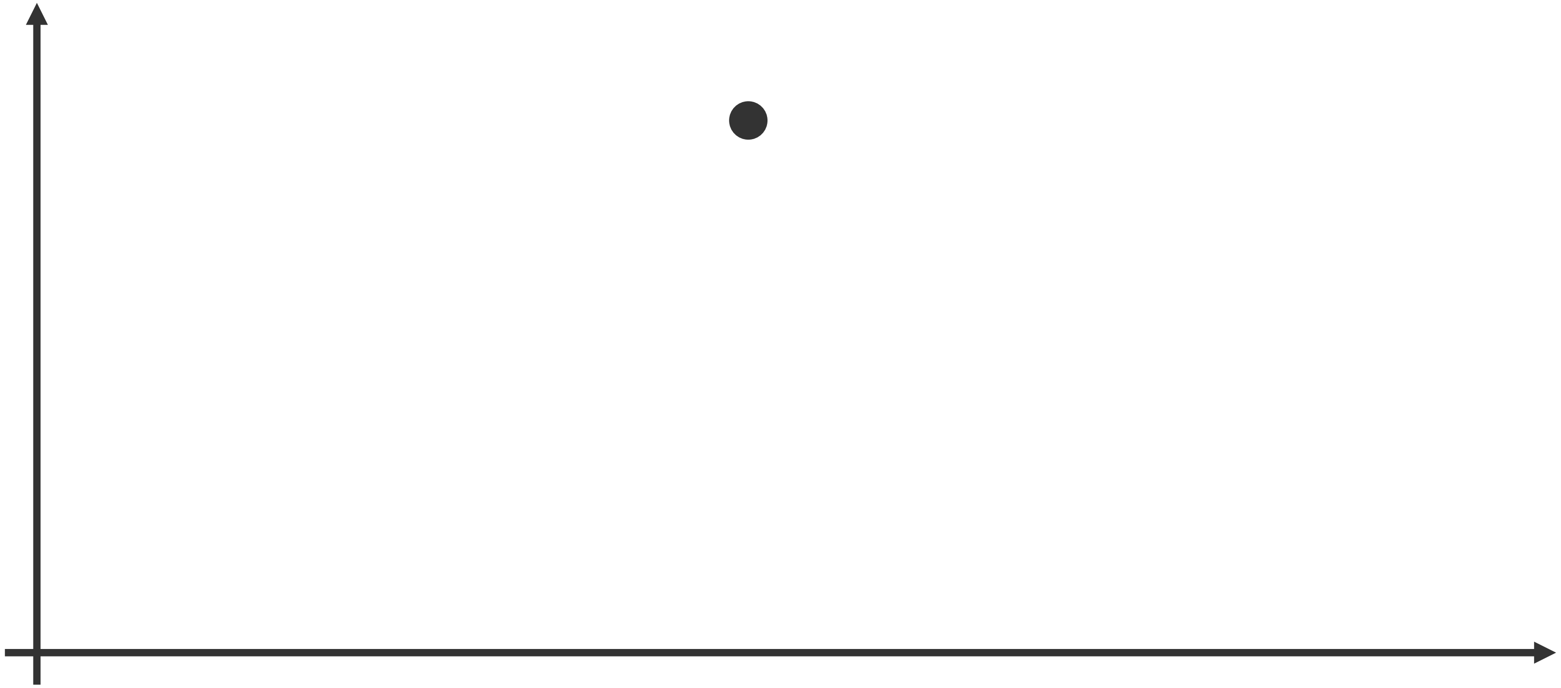
Procedure:

1. Measure the length of the pendulum (*L*) from the pivot point to the center of mass of the bob.
2. Displace the pendulum to a small angle (less than 15∘) to ensure that the motion approximates simple harmonic motion and release it.
3. Use the stopwatch to measure the time it takes for the pendulum to complete a number of oscillations.
4. To reduce error, measure the time for multiple oscillations (say, 10 or 20) and then divide by the number of oscillations to find the average period (*T*).
5. Repeat a few times and average to minimize random errors.

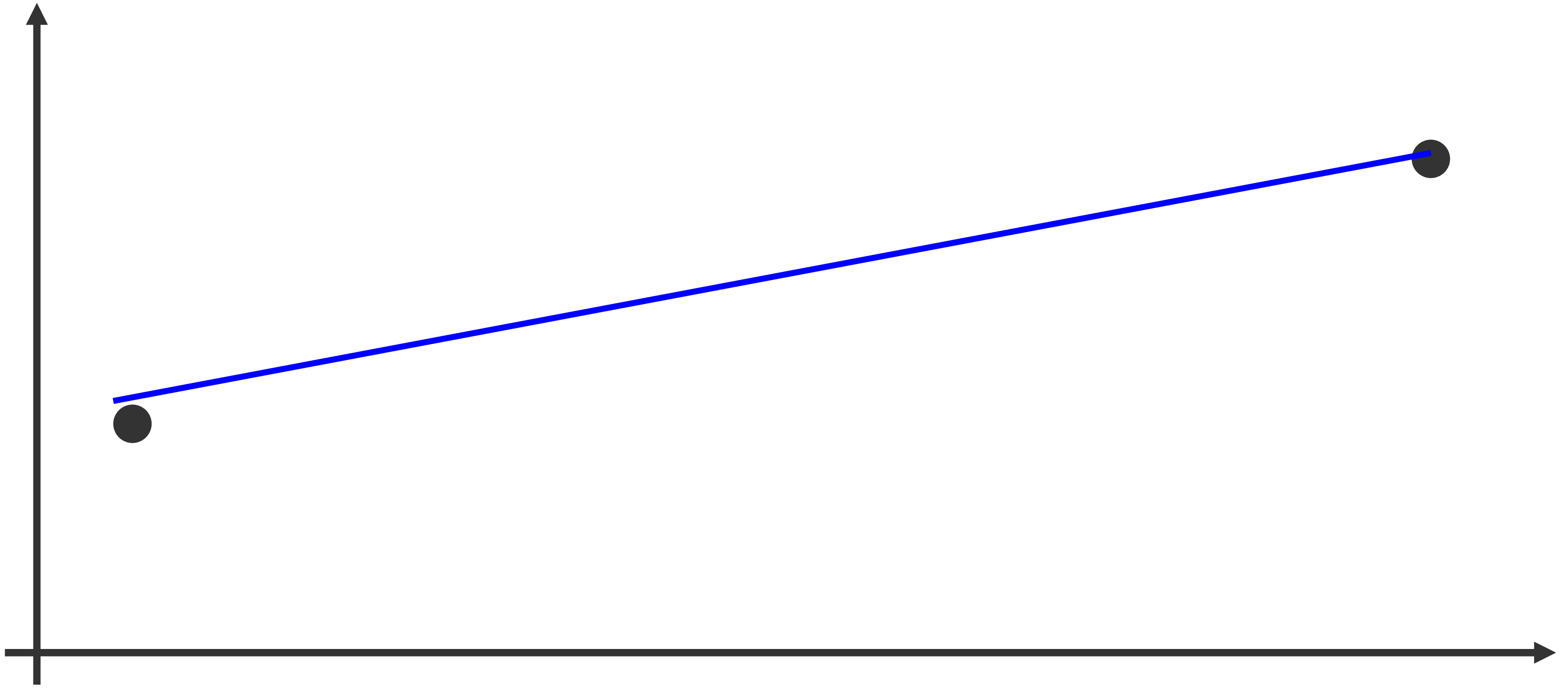
Pendulum example

- Let's assume that we have a ruler, balance, and stop watch
- However, the measurements of g gives us 12 m/s^2. We know that the true value is 9.8 m/s^2
- How can we analyze the uncertainties?

Physics vs. data science

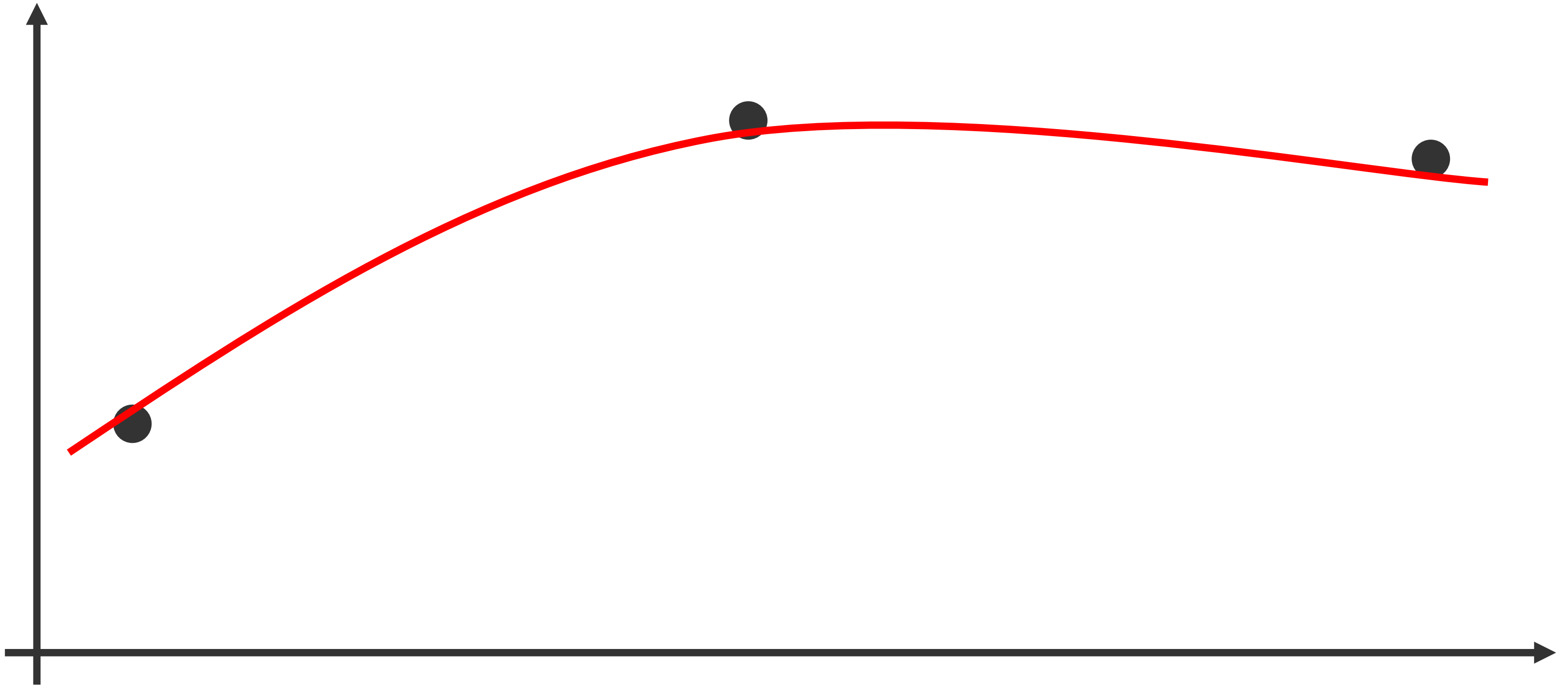


Physics vs. data science



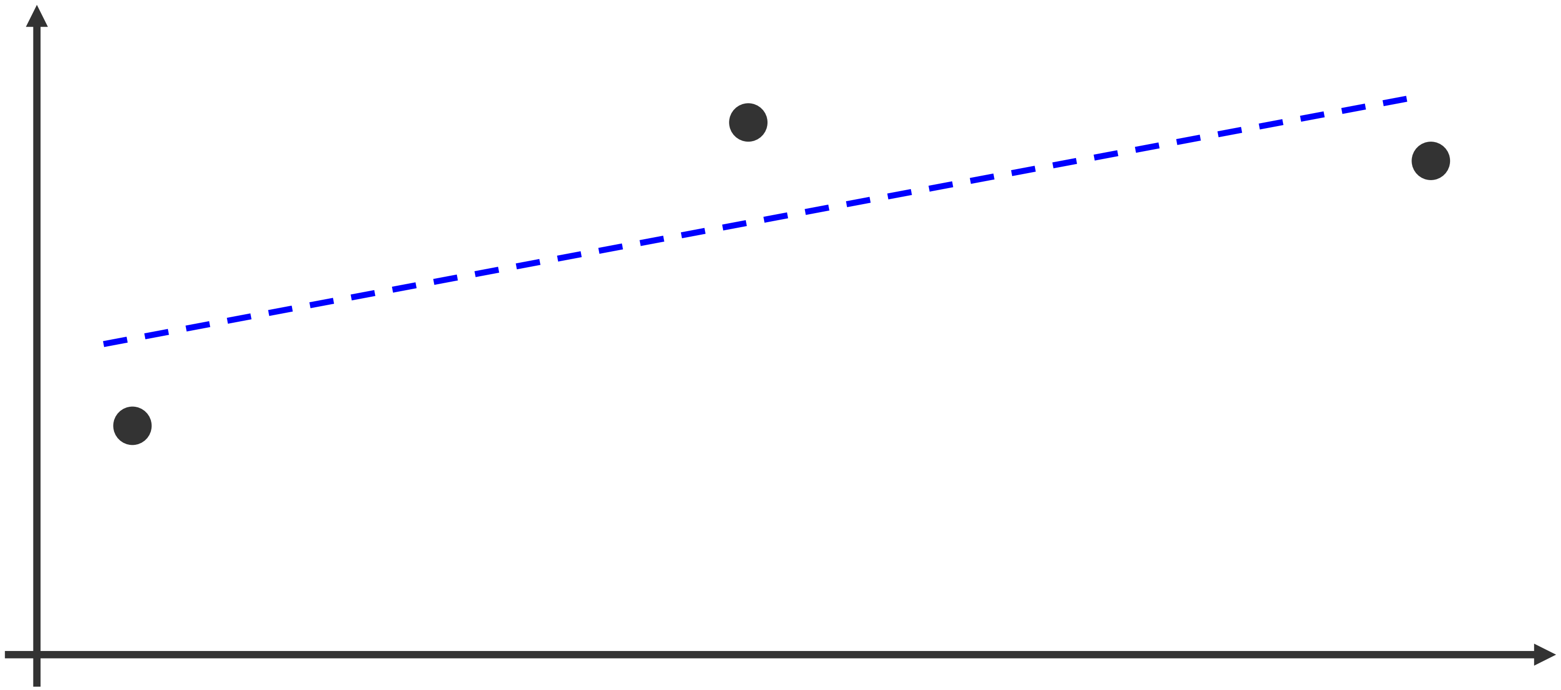
- If we have 2 data points, we “naturally” use linear model
- What should we use if we have three data points? Parabola or linear?
- What if we have one data point?

Physics vs. data science



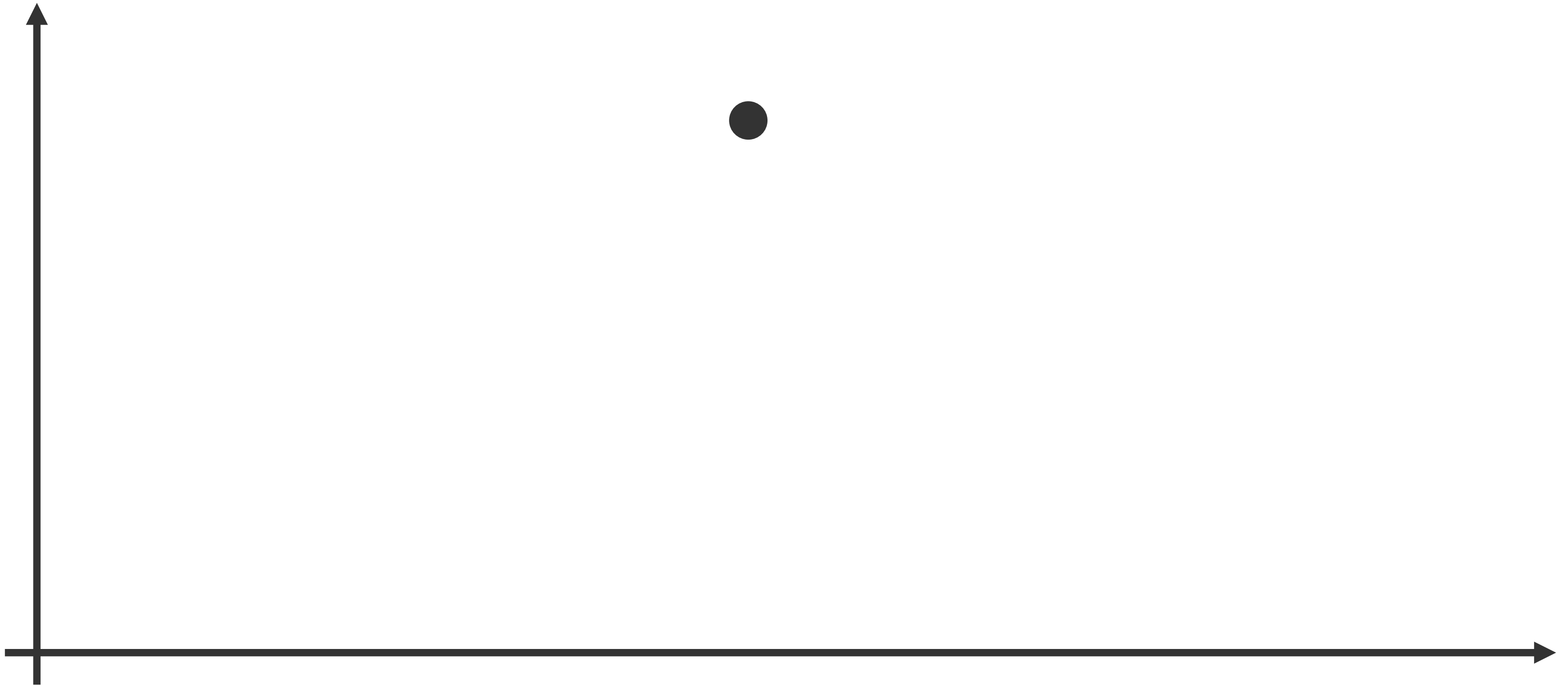
- If we have 2 data points, we "naturally" use linear model
- What should we use if we have three data points? Parabola or linear?
- What if we have one data point?

Physics vs. data science



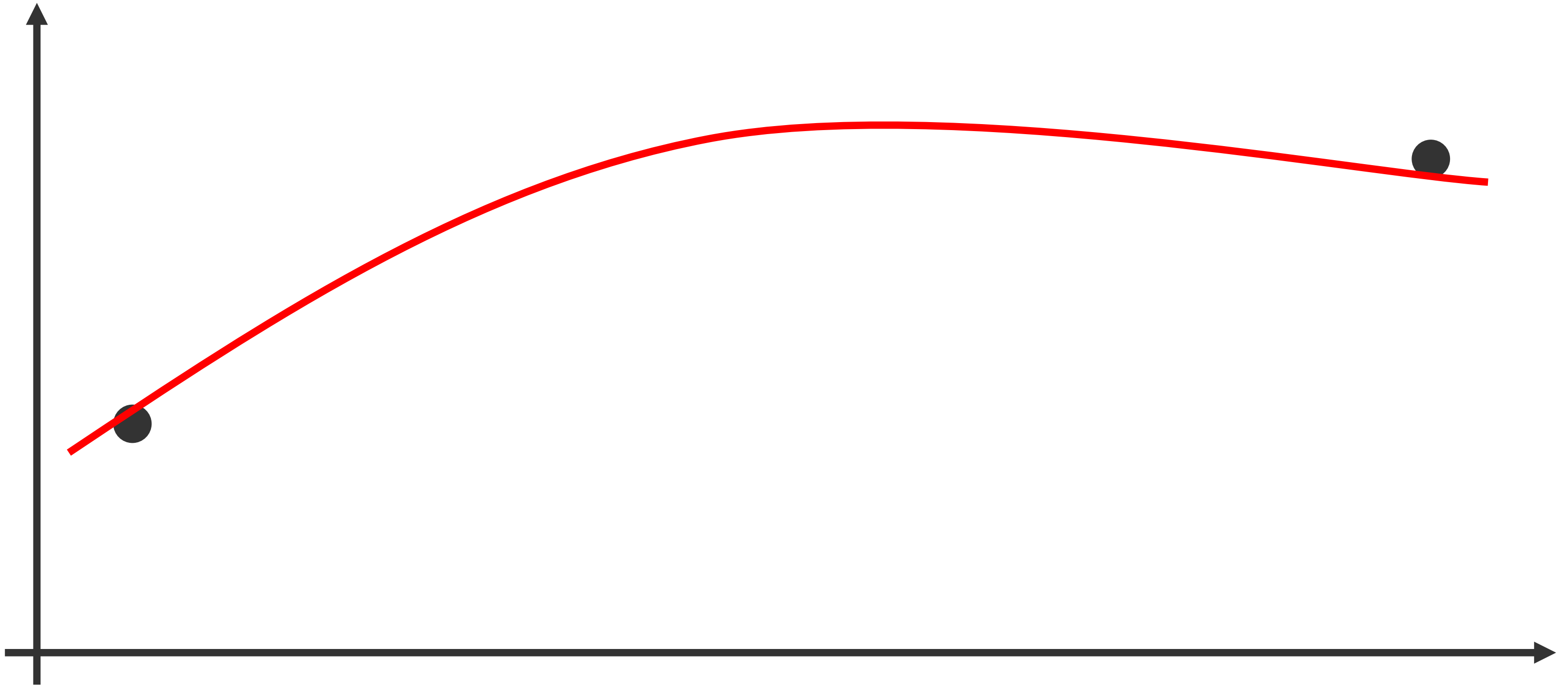
- If we have 2 data points, we "naturally" use linear model
- What should we use if we have three data points? Parabola or linear?
- What if we have one data point?

Physics vs. data science



- If we have 2 data points, we “naturally” use linear model
- What should we use if we have three data points? Parabola or linear?
- What if we have one data point?

Physics vs. data science

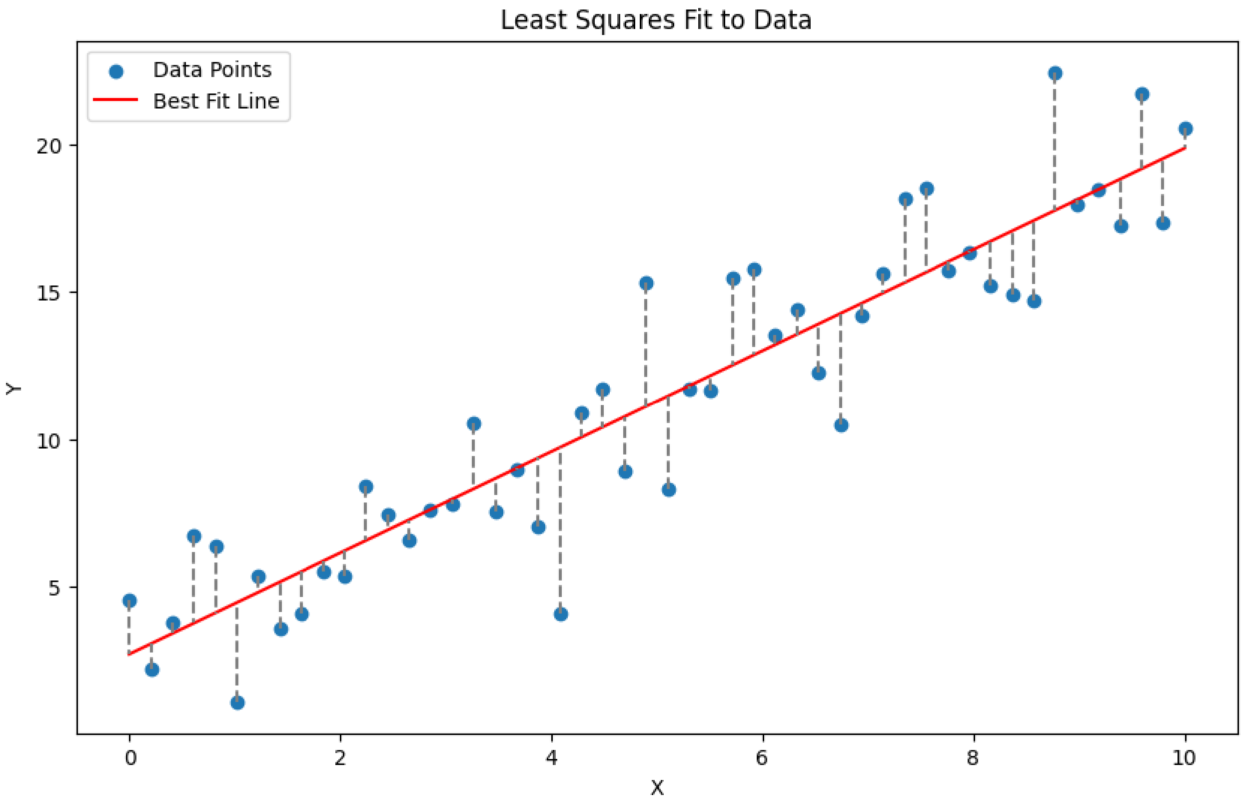


- If we have 2 data points, we “naturally” use linear model
- What should we use if we have three data points? Parabola or linear?
- What if we have one data point?

Let's start linear!

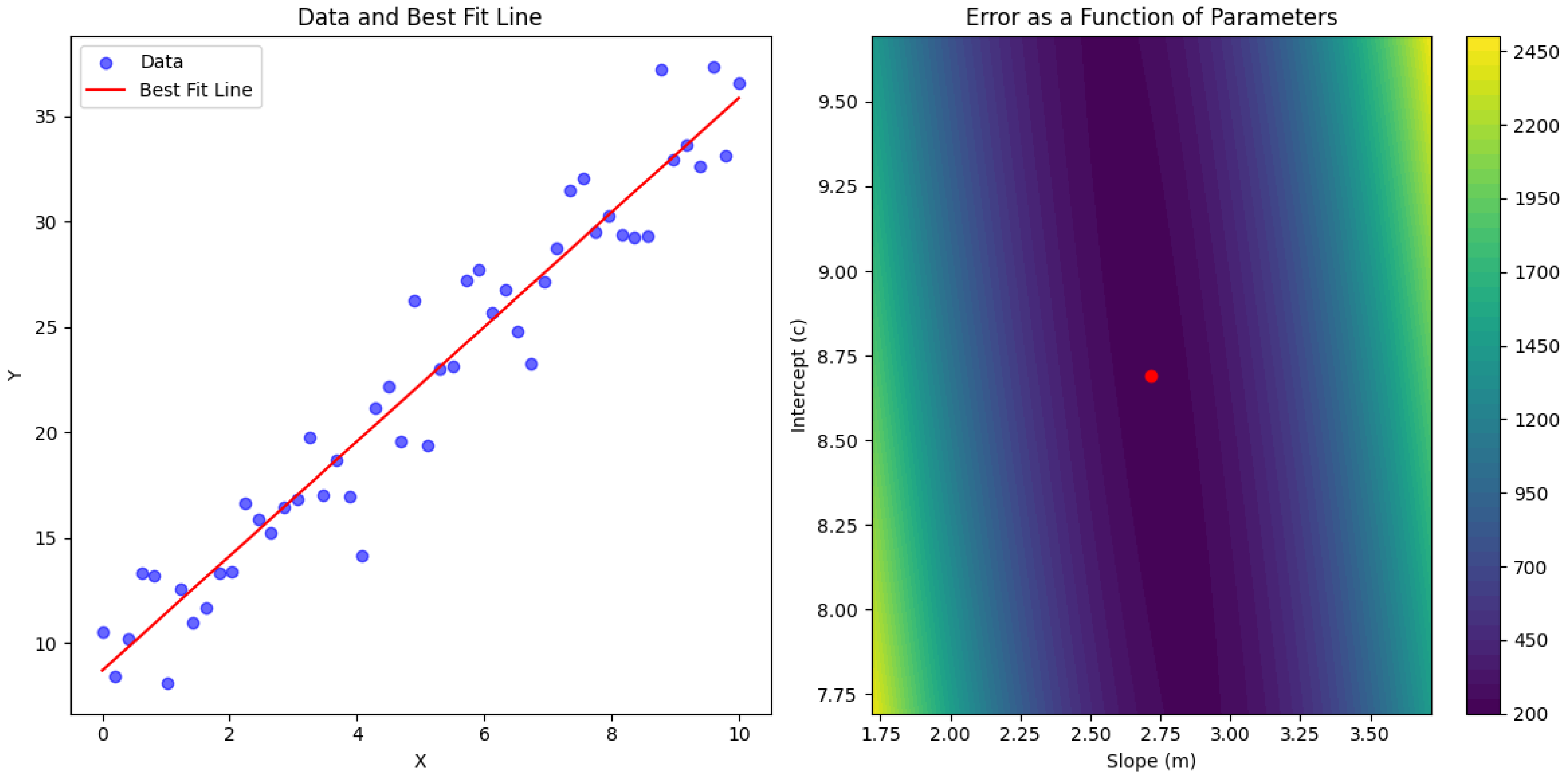
Least Squares Fitting: a method to determine the best-fitting line through a set of points.

Objective: Minimize the sum of the squares of the differences (residuals) between observed and predicted values.

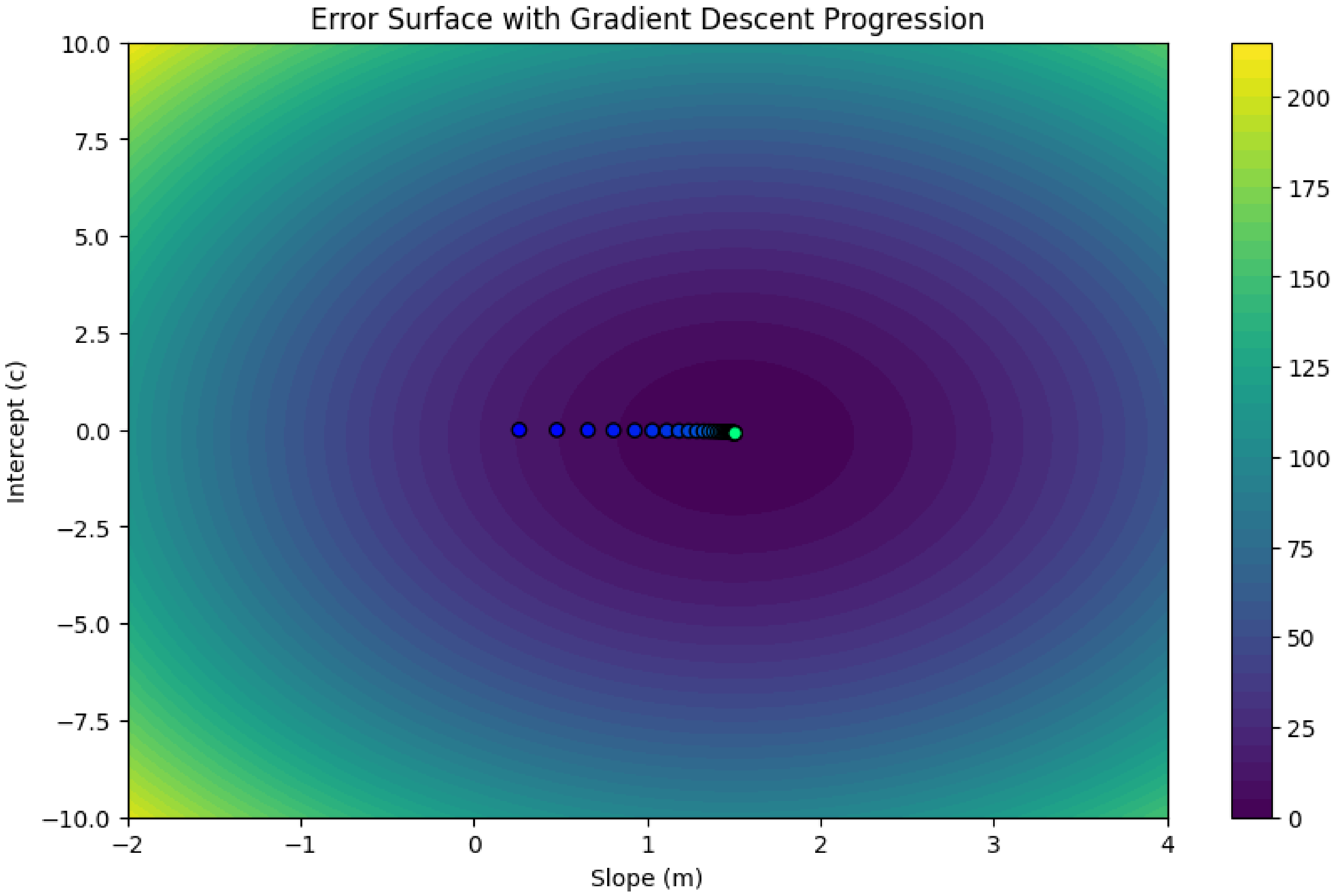


- We have data, meaning collection of points $(\mathbf{x_i}, \mathbf{y_i})$
- We choose function, here $\mathbf{y = a + b\,x}$
- We calculate total error, $\mathbf{L(a,b) = \sum((y_i - (a + b\,x_i))^2)}$
- We find parameters **a, b** for which **L(a,b)** is minimal

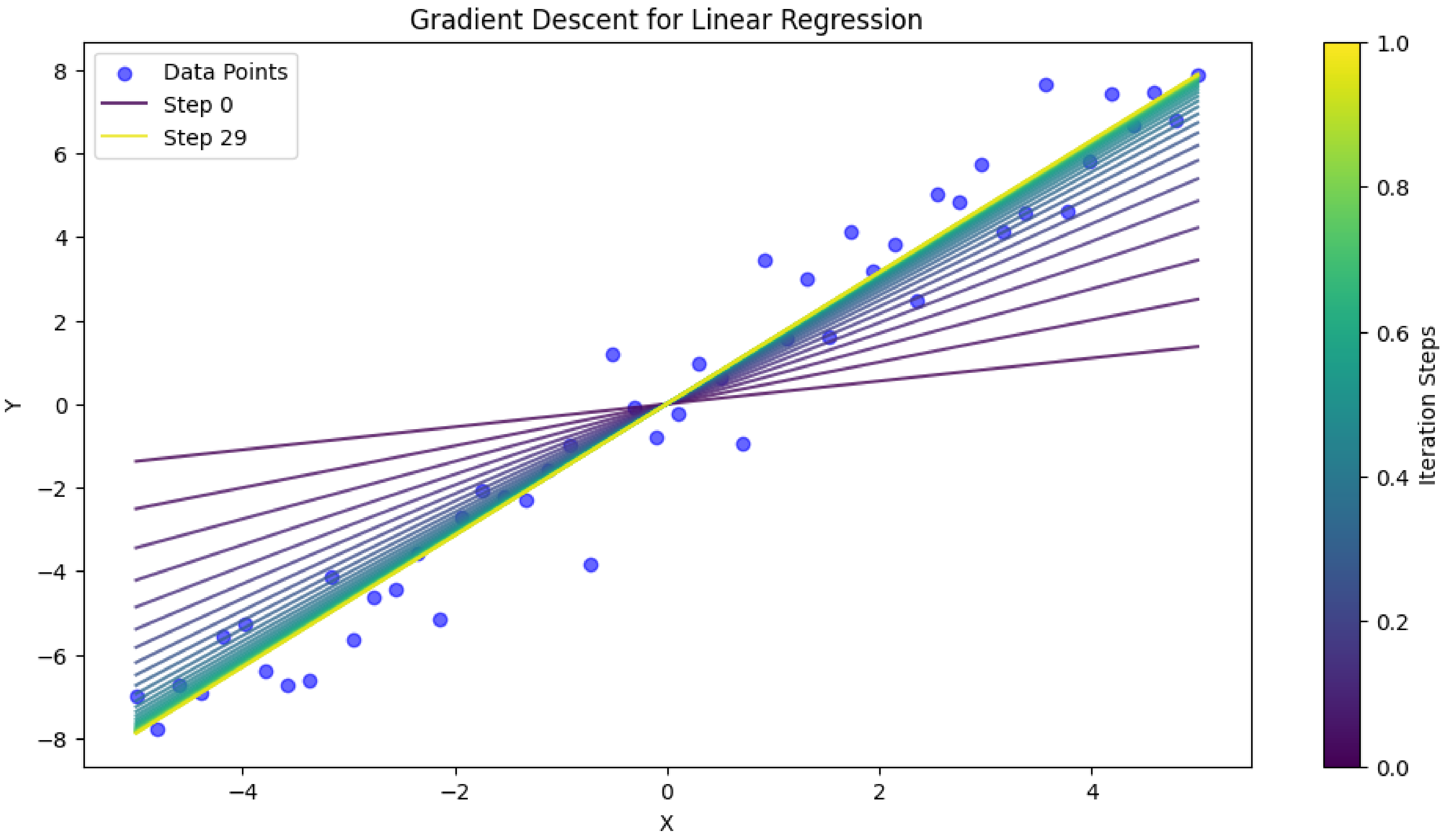
Let's start linear!



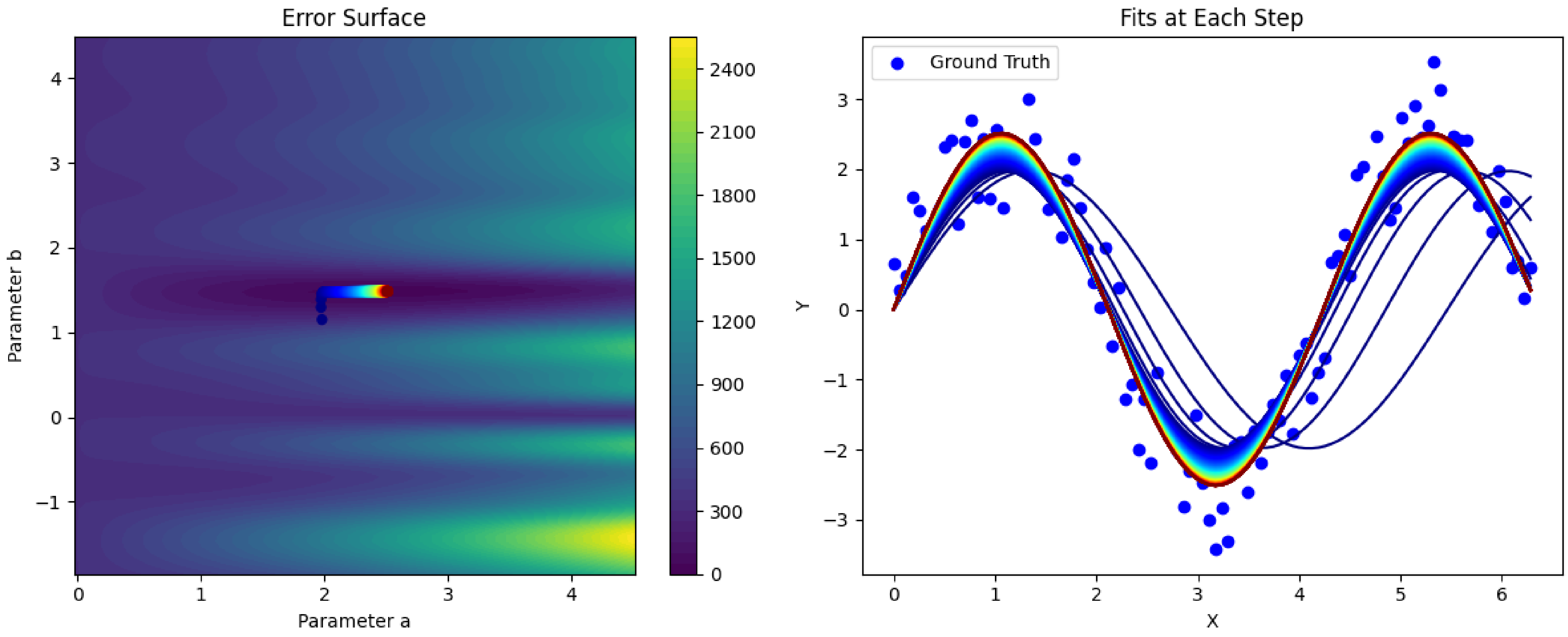
Finding minimum with gradient descent



Finding minimum with gradient descent



The sine that works



The sine that doesn't