Lecture 16
Simulation Based Inference for Astrophysics and Cosmology

Lecturer: **Dr Harry Bevins** (htjb2)

Overview

- What is Simulation Based Inference?
- Approximate Bayesian Computation
- Neural Posterior Estimation
- Neural Ratio Estimation

UNIVERSITY OF CAMBRIDGE

What is Simulation Based Inference?

Why not emulate the likelihood?

$$\log_e L(\theta) = \sum_i -\frac{1}{2}\log_e|2\pi\Sigma| - \frac{1}{2}\big(D_i - M_i(\theta)\big)^T \Sigma^{-1}(D_i - M_i(\theta))$$

- In the last lecture I discussed how the likelihood defines the noise distribution we expect in our data
- But often this is hard to know and approximations like the above Gaussian don't really describe our data well
- We looked at emulating the model describing our data but we can also think about emulating our likelihood function

Simulation Based Inference (SBI)

- Sometimes referred to as Likelihood-free Inference (not a good name)
- Or Implicit likelihood inference (a better name)
- The idea is to learn an approximation for the likelihood or indeed the posterior via simulations of the data
- Simulations here includes the physics we are interested in, and contaminating signals, the noise and the impact of the instrument we are observing with

Simulation Based Inference (SBI)

- A number of different Simulation-Based inference algorithms
- The idea has been around for a long time but has become more feasible with the advent of advance machine learning tools
- Examples include; Approximate Bayesian Computation, Neural Posterior, Neural Likelihood, Neural Joint and Neural Ratio estimation
- See K. Cranmer et a. [1911.01429] for a review

UNIVERSITY OF CAMBRIDGE

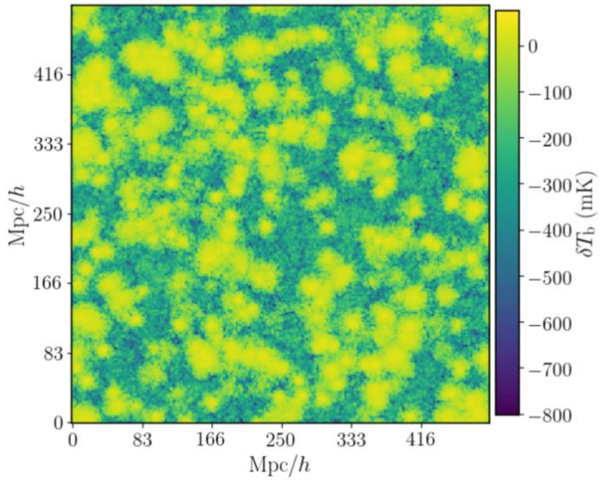
Simulation Based Inference (SBI)

- Beneficial when
  - we can not analytically write the likelihood
  - we do not know the noise distribution in our data
  - the analytic likelihood is too computationally expensive

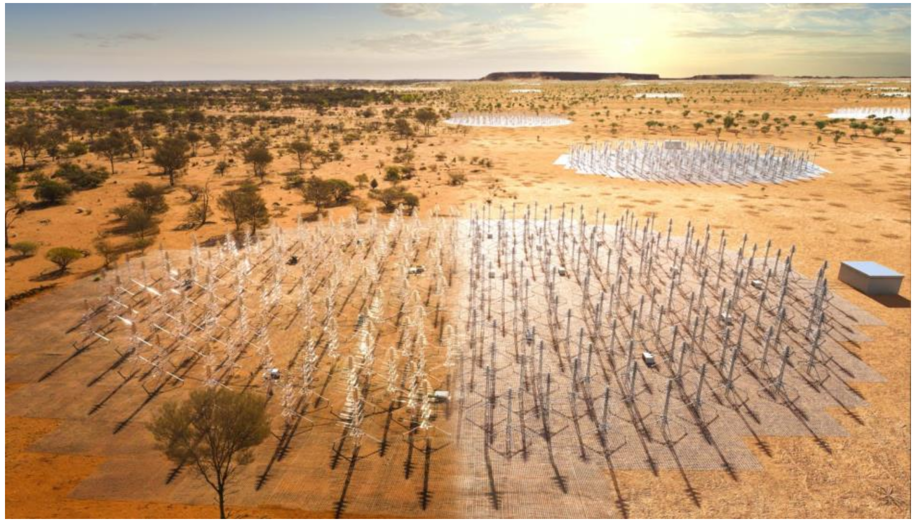
UNIVERSITY OF CAMBRIDGE

Field Level Inference

- What cosmological parameters give us this 21-cm intensity map?
- Could compress to summary statistics but there is potential loss of information
- Hard to write an analytic likelihood
- Turn to SBI instead

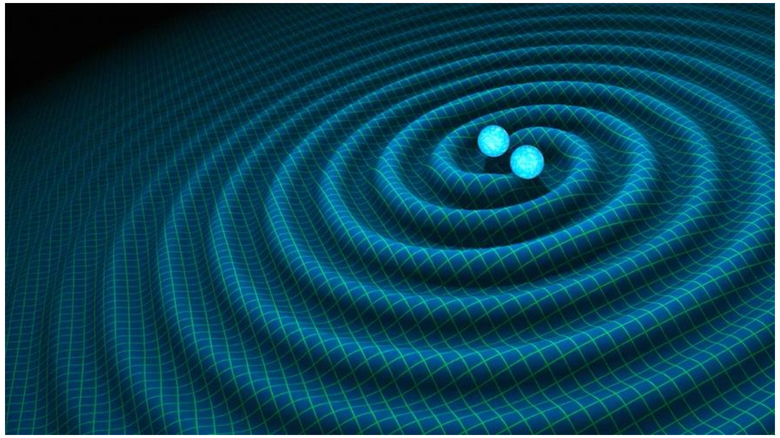


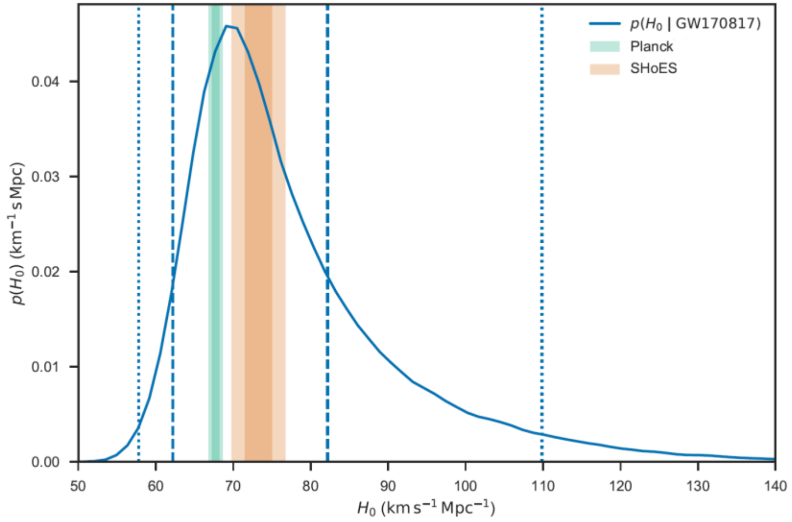
Bera et al. 2022
[2210.12164]
$z \approx 9$



GW Follow Ups

- GW chirps give us distance
- Electromagnetic follow up observations of GWs can help us pinpoint the redshift of the sources
- With distance and redshift we can learn about the expansion rate of the universe
- But GWs emitted from cataclysmic events with short lifetimes so follow ups have to be triggered quickly
- SBI allows us to do that





[1710.05835]

Approximate Bayesian Computation

The idea

- ABC originates from the 80s (Rubin 1984) and was one of the first proposed techniques for SBI
- The idea is to generate a set of simulations covering a broad prior parameter space
- Define a distance metric between the data and simulations $\rho(D, S)$
- And define a non-zero tolerance ϵ such that samples are accepted into the posterior based on $\rho(D, S) < \epsilon$

Approximating the posterior

- Effectively we are approximating $P(\theta|D)$ with $P(\theta|\rho(D,S) < \epsilon)$
- For small ϵ this approximation is usually okay
- However, if ϵ is set to large then the approximation will be poor
- What constitutes small and large will be data specific and so setting ϵ is hard

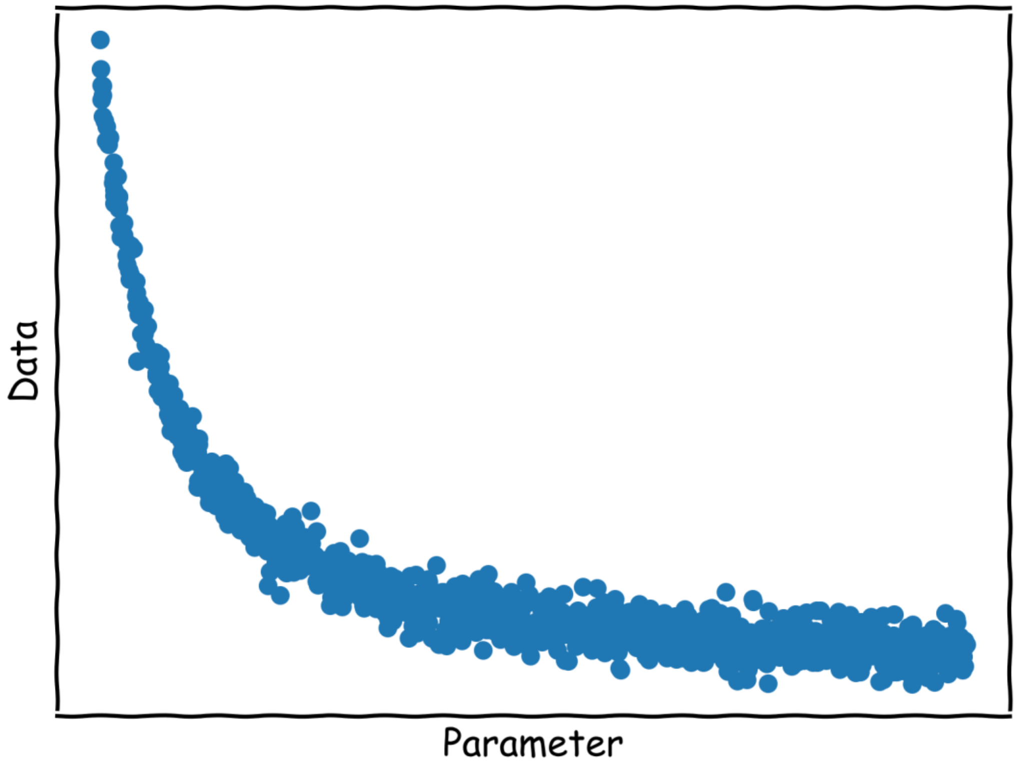
An Example

- Single parameter problem such that our data

$$D = \theta^{-1.5} + \sigma$$

- Where σ is Gaussian random noise

- Think about the joint distribution $P(D, \theta)$



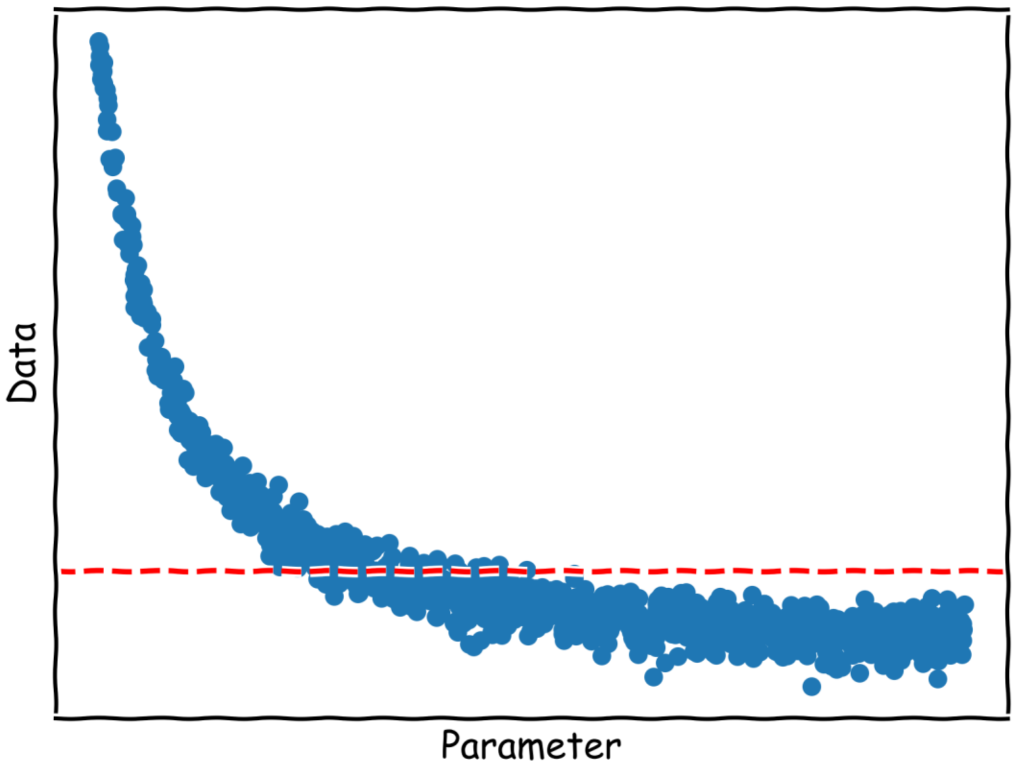
An Example

- Make an observation of our data

- Define a distance metric

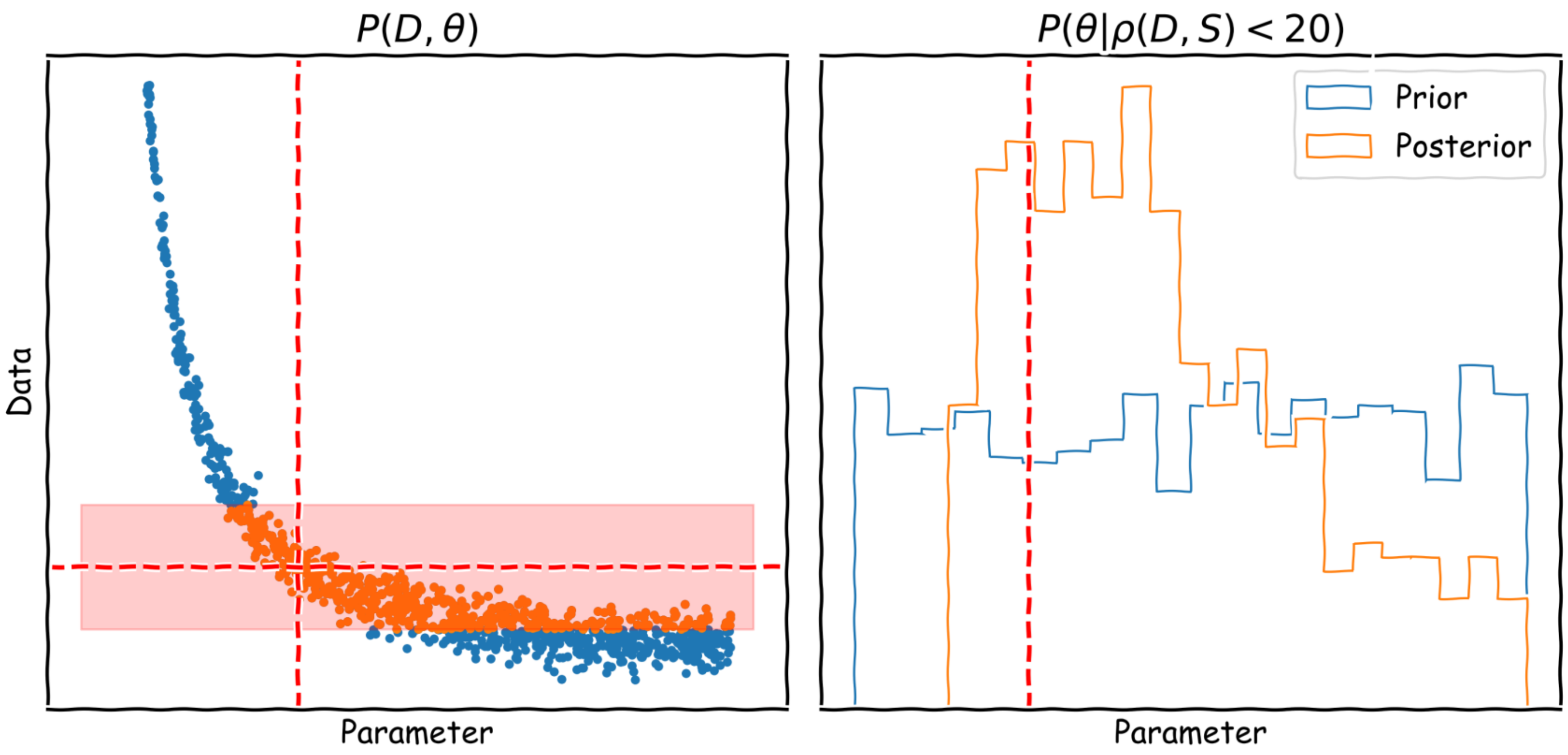
$$\rho(D,S) = |D - S|$$

- Pick a value for epsilon say 20

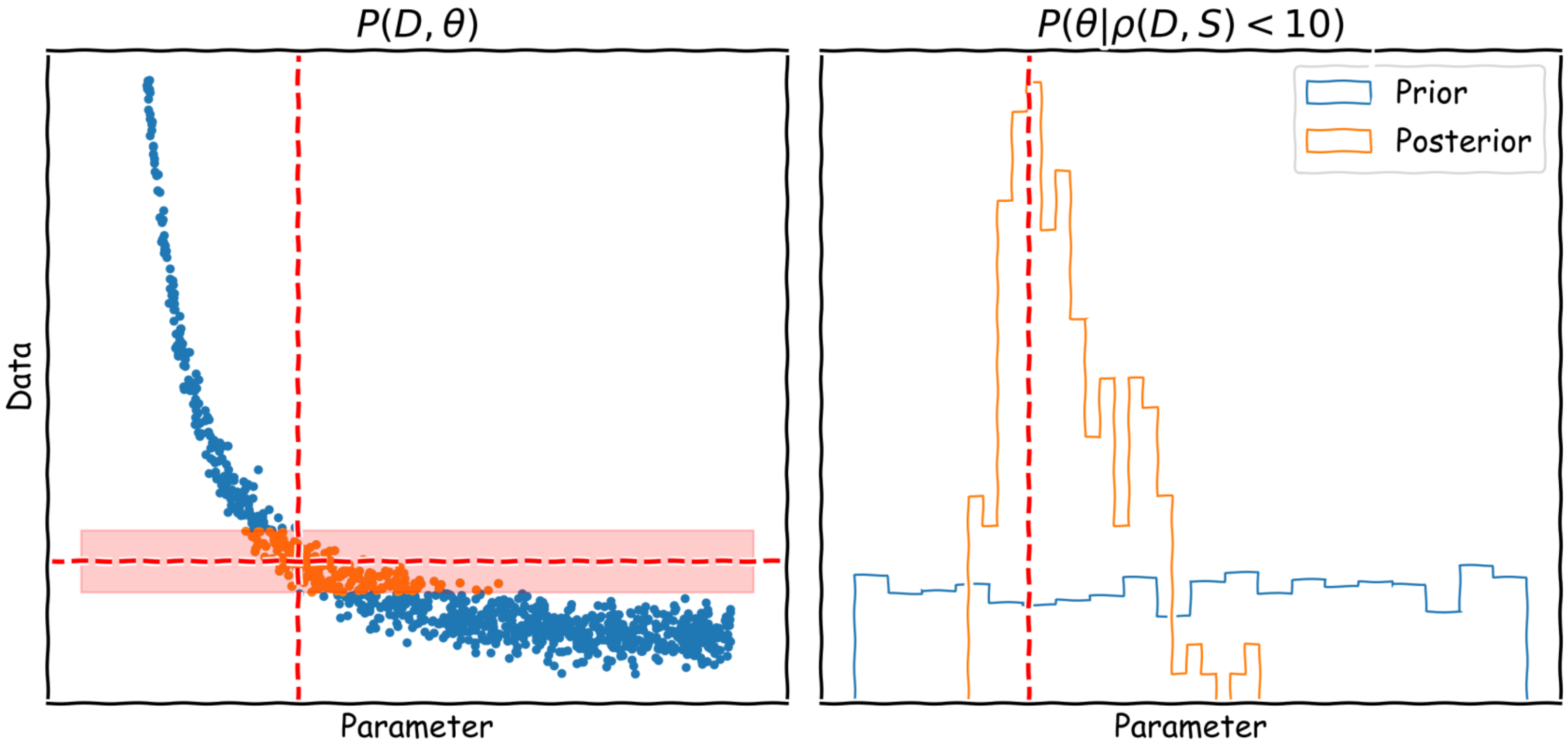


UNIVERSITY OF CAMBRIDGE

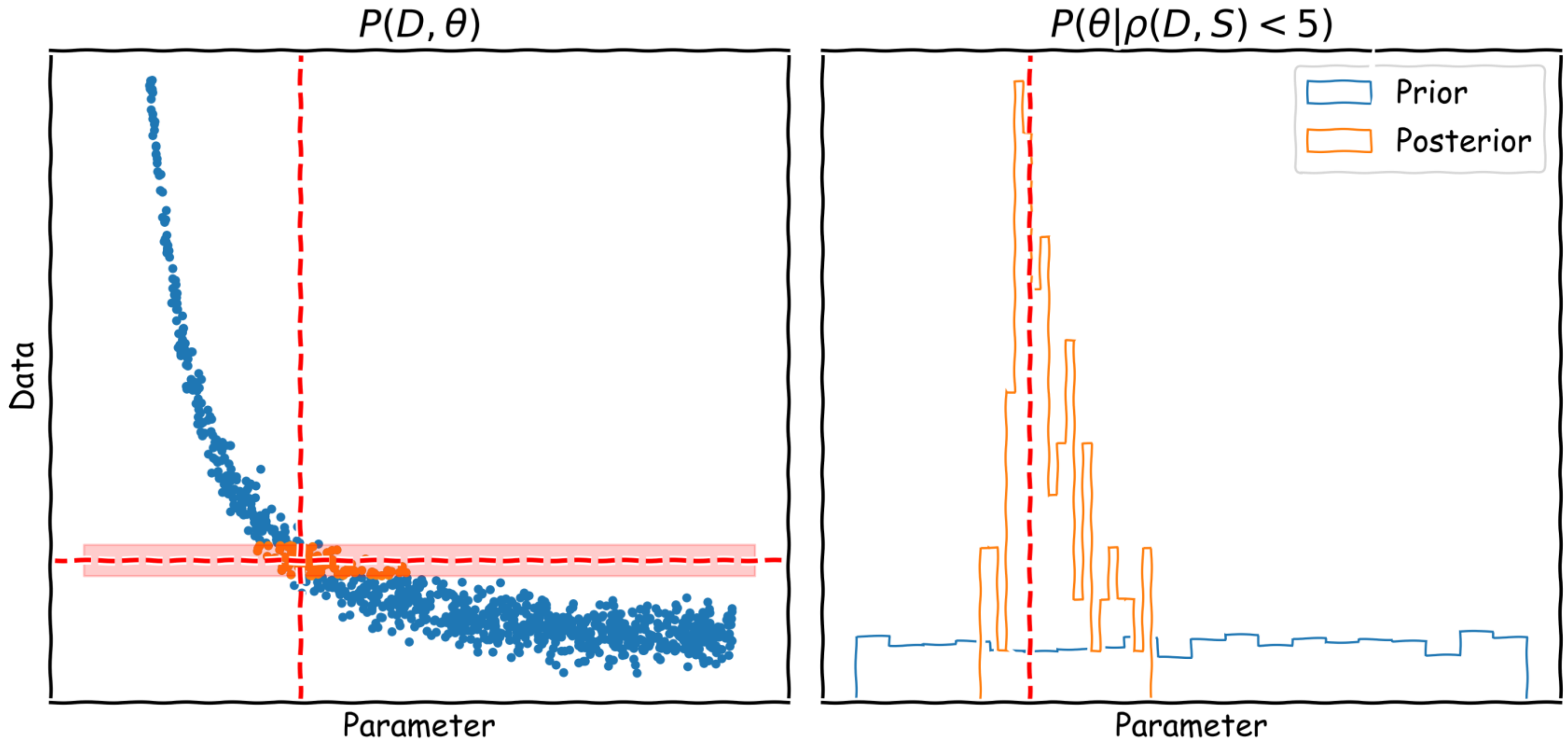
$\epsilon = 20$



$\epsilon = 10$


$P(D, \theta)$
Data
Parameter
$P(\theta|\rho(D, S) < 10)$
Prior
Posterior
Parameter

$\epsilon = 5$



$P(D, \theta)$
Data
Parameter
$P(\theta|\rho(D, S) < 5)$
Prior
Posterior
Parameter

UNIVERSITY OF CAMBRIDGE

What does the algorithm look like?

Distance Metric

Prior samples and simulations

```
def distance(sim_d, d):
    return np.abs(d - sim_d)

def sampling(samples, data, epsilon):

    posterior = []
    accepted_data = []
    for i in range(len(samples)):
        if distance(data[i], true_data) < epsilon:
            posterior.append(samples[i])
            accepted_data.append(data[i])
    posterior = np.array(posterior)
    accepted_data = np.array(accepted_data)
    return posterior, accepted_data
```

$\rho(D, S) < \epsilon$

Accepted samples and simulations

UNIVERSITY OF CAMBRIDGE

No likelihood calls

- We did not make a single call to any likelihood function here
- But there is a cost to generating our simulations (so it might still be slow)
- We could use emulators to generate the simulations if they are prohibitively costly (e.g. the VAE we saw last lecture)
- Typically need fewer simulation calls than during an MCMC run
- Likelihood might be intractable anyway so MCMC might not be an option

Problems with ABC?

- How do we set ϵ?
- Becomes very expensive when we have many dimensions and big data sets (curse of dimensionality)
- Sensitive to accuracy of simulations (more general for SBI)
- Not amortized? Basically, means that if our data changes we have to repeat most of the process

UNIVERSITY OF CAMBRIDGE

Neural Posterior Estimation (NPE)

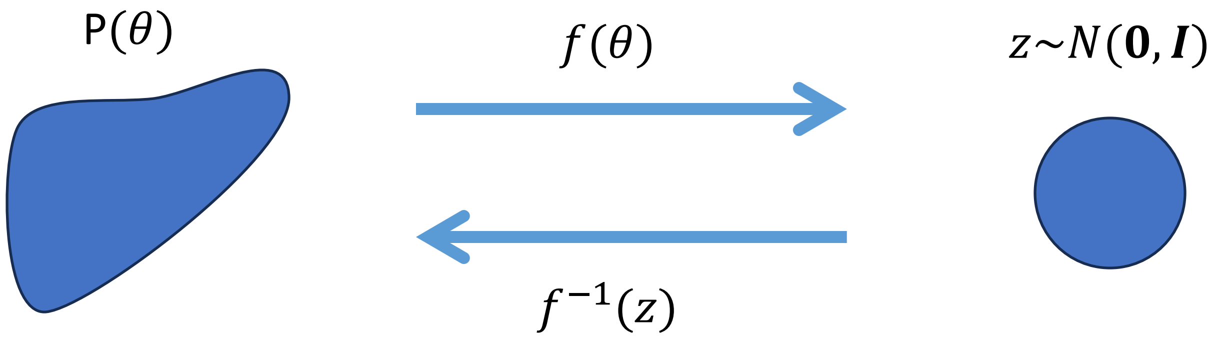
What is NPE?

- Here we use neural networks to approximate the posterior distribution $P(\theta|D)$
- Specifically, we use normalising flows (NFs)
- NPEs are amortized
- After an initial cost to generate simulations and train the NF we can apply the algorithm to many different data sets

Normalizing Flows (NFs)

- Invertible transformation from some known distribution (a multi-variate Gaussian) to a more complex target distribution (a posterior for example)

- Focusing on Masked Autoregressive Flows (MAF) which comprises of a series of Masked Autoencoder for Distribution Estimation (MADE, Papamakarios et al 2017) neural networks



Normalizing Flows (NFs)

- We choose to represent our data as a series of conditionals where i represents the dimension

$$P(\theta) = \prod_i P(\theta_i | \theta_{j \in N \neq i})$$

- And model each conditional as a Gaussian distribution

$$P(\theta_i | \theta_{j \in N \neq i}) = N(\mu_i, \sigma_i)$$

Normalizing Flows (NFs)

$$P(\theta_i|\theta_{j \in N \neq i}) = N(\mu_i, \sigma_i)$$

- Our conditionals are just a shifted and scaled version of the base distribution (standard normal)

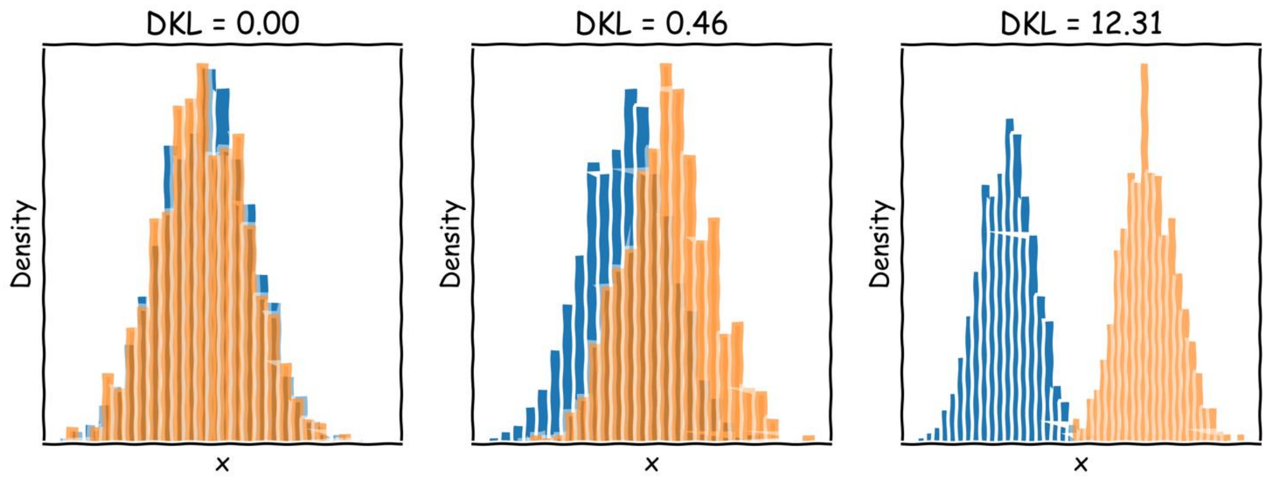
- μ_i and σ_i are functions of the neural network hyperparameters and so we have to optimize them with some loss function

UNIVERSITY OF CAMBRIDGE

Normalizing Flows (NFs)

- Minimize the KL divergence between the probability of some samples on the network $P_\phi(\theta)$ and the true target probability $P(\theta)$

$$D_{KL} = \int P(\theta) \log \frac{P(\theta)}{P_\phi(\theta)} d\theta$$



UNIVERSITY OF CAMBRIDGE

Normalizing Flows (NFs)

- Minimize the KL divergence between the probability of some samples on the network $P_\phi(\theta)$ and the true target probability $P(\theta)$

$$D_{KL} = \int P(\theta) \log \frac{P(\theta)}{P_\phi(\theta)} d\theta$$

$$D_{KL} = \int P(\theta) \log P(\theta)\, d\theta \; - \int P(\theta) \log P_\phi(\theta)\, d\theta$$

- The first term is independent of ϕ so we can ignore it and the last term is just

$$\int P(\theta) \log P_\phi(\theta)\, d\theta = < \log P_\phi(\theta) >_{P(\theta)} = \frac{1}{N} \sum_j \log P_\phi(\theta_j)$$

Normalizing Flows (NFs)

- To evaluate $\log P_\phi(\theta_j)$ we use the change of variables formula between the target and the standard normal distribution

$$\log P_\phi(\theta) = \log N\big(f_\phi(\theta)|\mathbf{0}, \boldsymbol{I}\big) + \log(\det\left|\frac{d f_\phi(\theta)}{d\,\theta}\right|)$$

- We give the network a sample on the target distribution and it transforms it into a sample on the base distribution

- We then evaluate the Jacobian between the base and target and the probability of that sample on the base

UNIVERSITY OF CAMBRIDGE

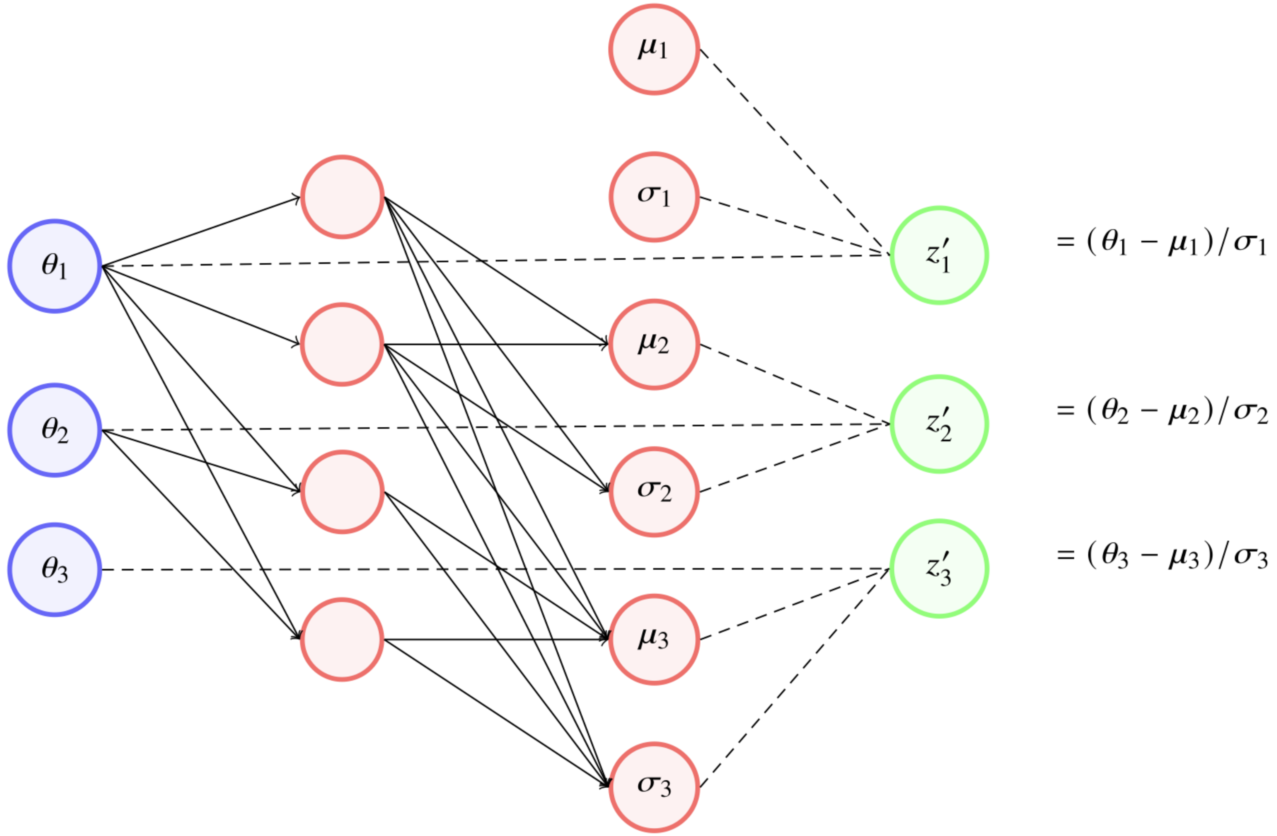
Normalizing Flows (NFs)

$$\log P_\phi(\theta) = \log N\big(f_\phi(\theta)|\mathbf{0}, \boldsymbol{I}\big) + \log(\det\left|\frac{df_\phi(\theta)}{d\ \theta}\right|)$$

- The Jacobian is evaluated across the network
- It accounts for the change in volume between the base and target
- It comes for free from tensorflow/pytorch

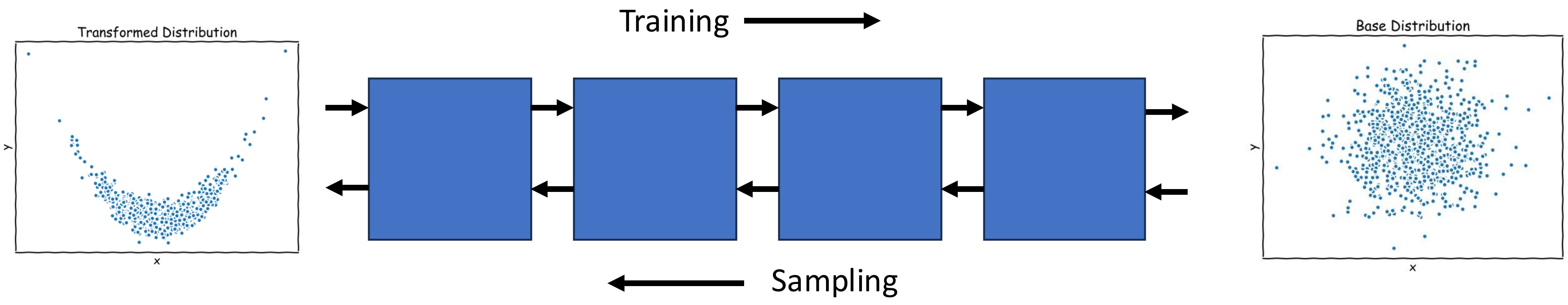
UNIVERSITY OF CAMBRIDGE

Masked Autoencoder for Distribution Estimation (MADE)



[Bevins et al 2023]

Masked Autoregressive Flow (MAF)



- Chain a series of MADEs together to get MAFs
- Idea is that we can learn more complex distributions with more complex architectures
- Essentially shifting and scaling repeatedly the base distribution
- The modelled conditionals in each MADE will be slightly different

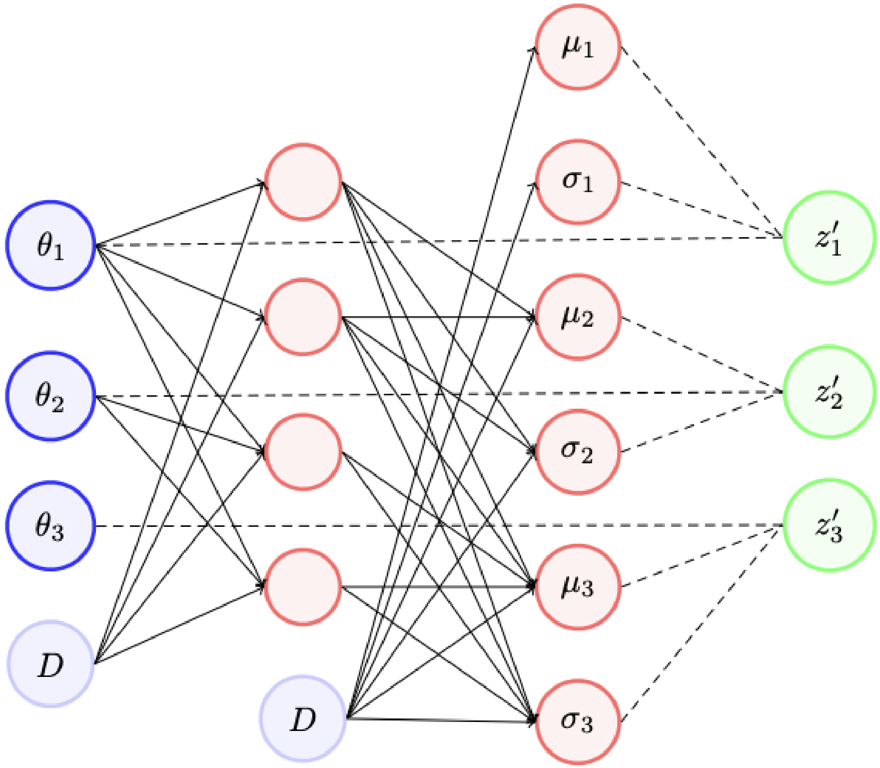
But this is just density estimation?

- NFs estimate the probability of $P(\theta)$ but what we really want is a posterior distribution i.e. $P(\theta|D)$

- To do this we use conditional MAFs

- Essentially the transformation (modelled by the network hyperparameters) from samples on the base distribution to the target becomes conditional on the data

Conditional MADE



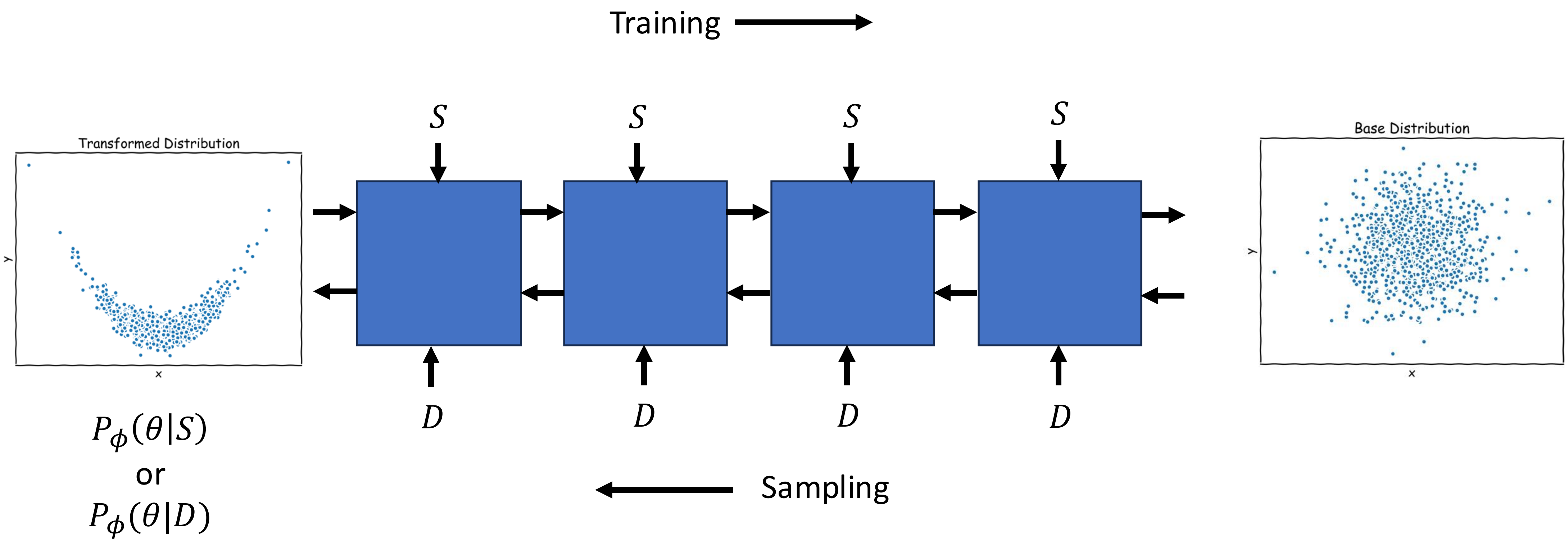
$= (x_1 - \mu_1)/\sigma_1$

$= (x_2 - \mu_2)/\sigma_2$

$= (x_3 - \mu_3)/\sigma_3$

UNIVERSITY OF CAMBRIDGE

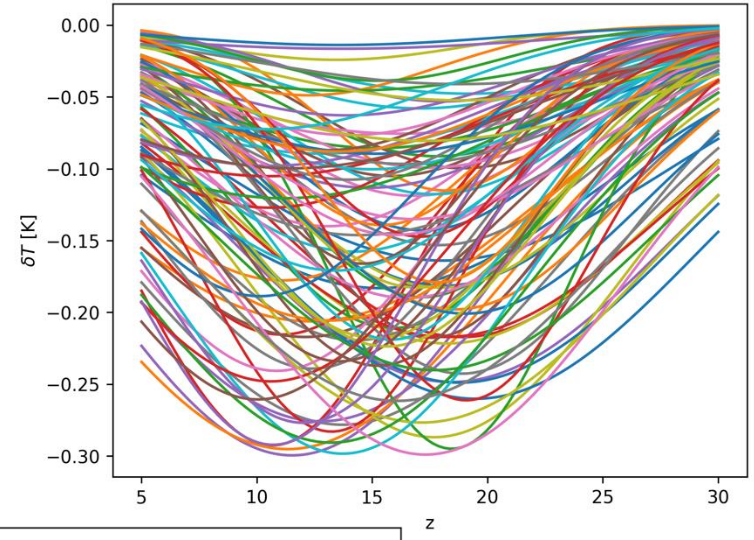
Conditional MAF

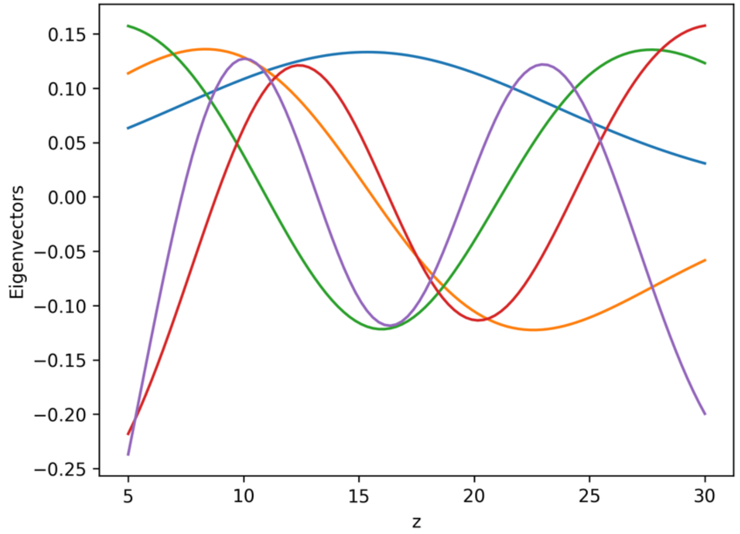


UNIVERSITY OF CAMBRIDGE

An example: 21cm Cosmology

- Think again about sky-averaged 21-cm Cosmology
- Transforming a 3D space of Amplitude, central redshift and width
- No noise for simplicity
- Conditioned on 100D data space!!
- Need to do some compression

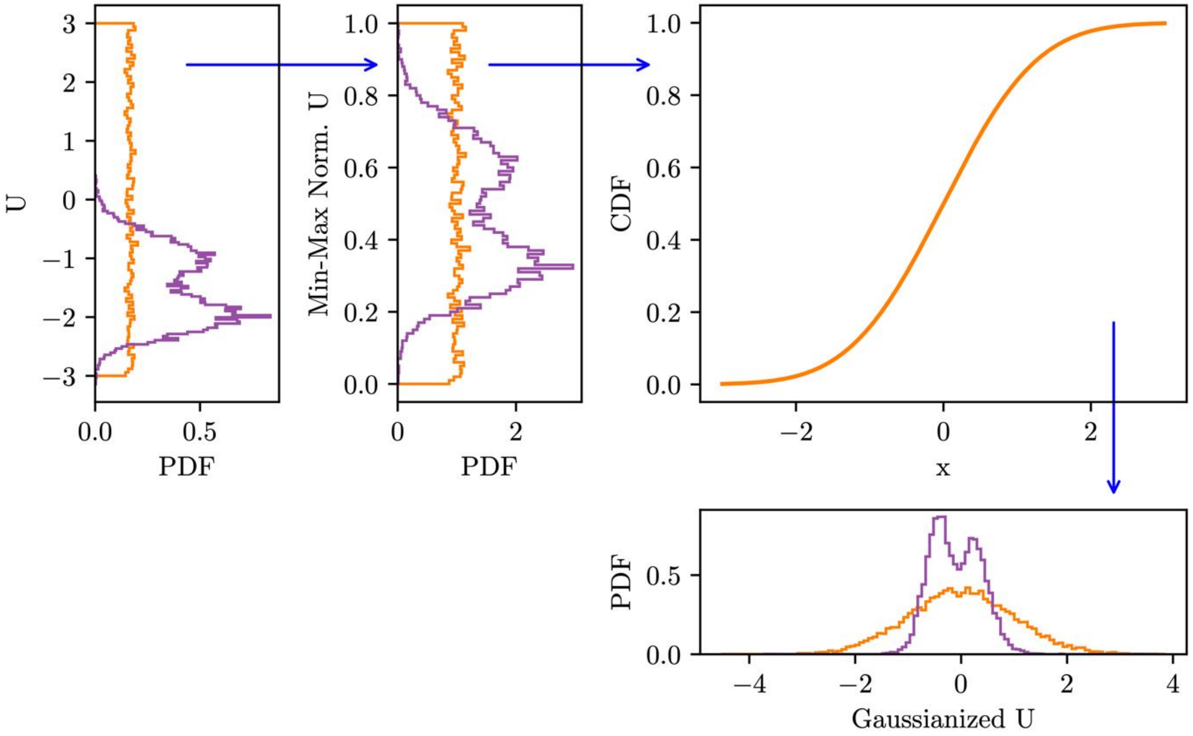




UNIVERSITY OF CAMBRIDGE

An example: 21cm Cosmology

- Example code on https://github.com/htjb/Talks
- Quite non-trivial tensorflow set up but hopefully the basic idea is clear
- We have to do the usual tricks normalising our parameter and data space

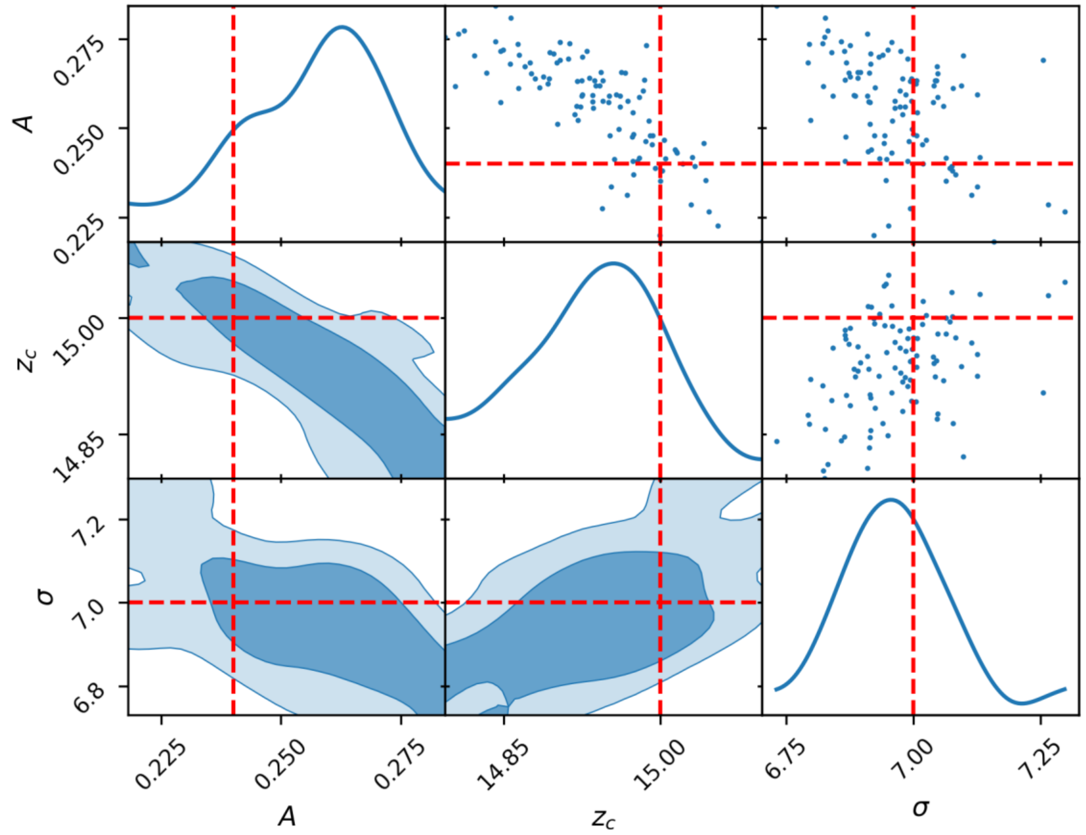


Bevins et al 2023

UNIVERSITY OF CAMBRIDGE

An example: 21cm Cosmology

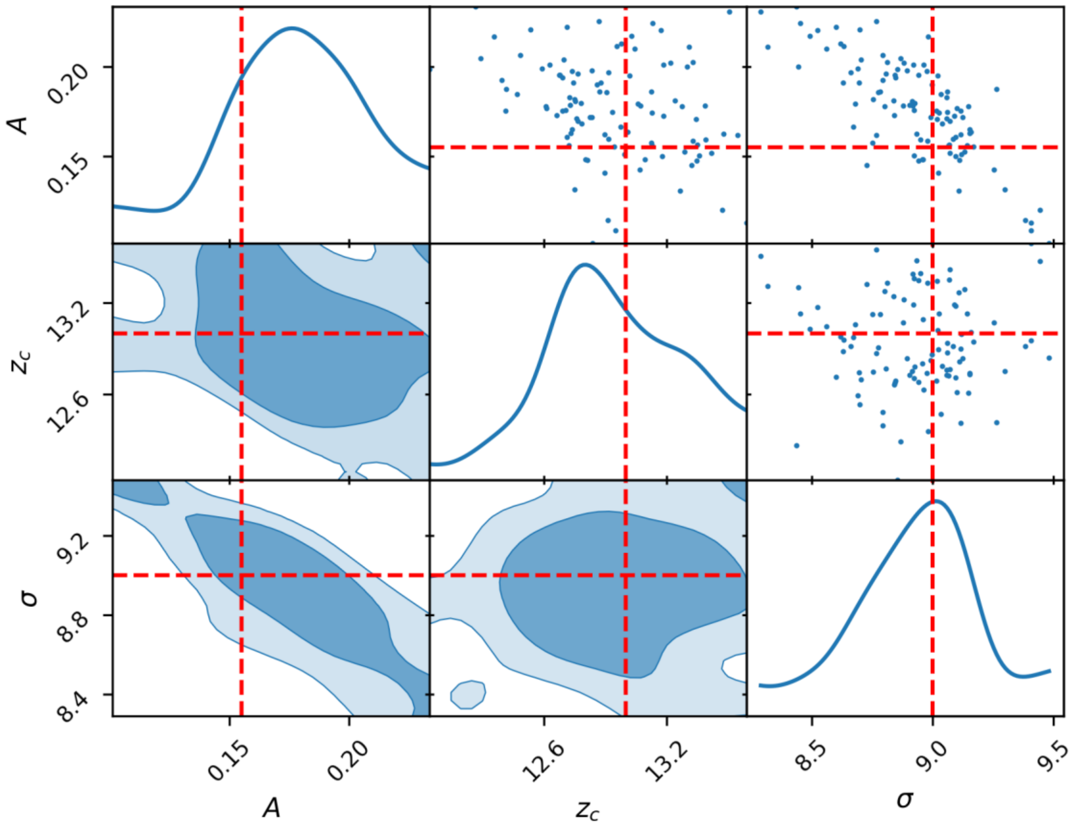
- We train on pairs of parameters and simulated data
- But once trained we pass the network samples from the base distribution and the observed data
- Returns a posterior distribution!



UNIVERSITY OF CAMBRIDGE

An example: 21cm Cosmology

- Once trained the NPEs are very fast
- They are amortized so we can show them another observation and recover a posterior
- But we have to be confident that our observation is well represented by the simulations



UNIVERSITY OF CAMBRIDGE

Neural Ratio Estimation (NRE)

What is NRE?

- NREs are an alternative to ABC and NPE and work by predicting ratios of densities

- Typically, we set NREs up to give us the ratio $\frac{P(D|\theta)}{P(D)} = \frac{L(\theta)}{Z}$

- We can sample over our NRE and multiply the ratio with a prior probability to recover a properly normalised posterior

As a classification problem

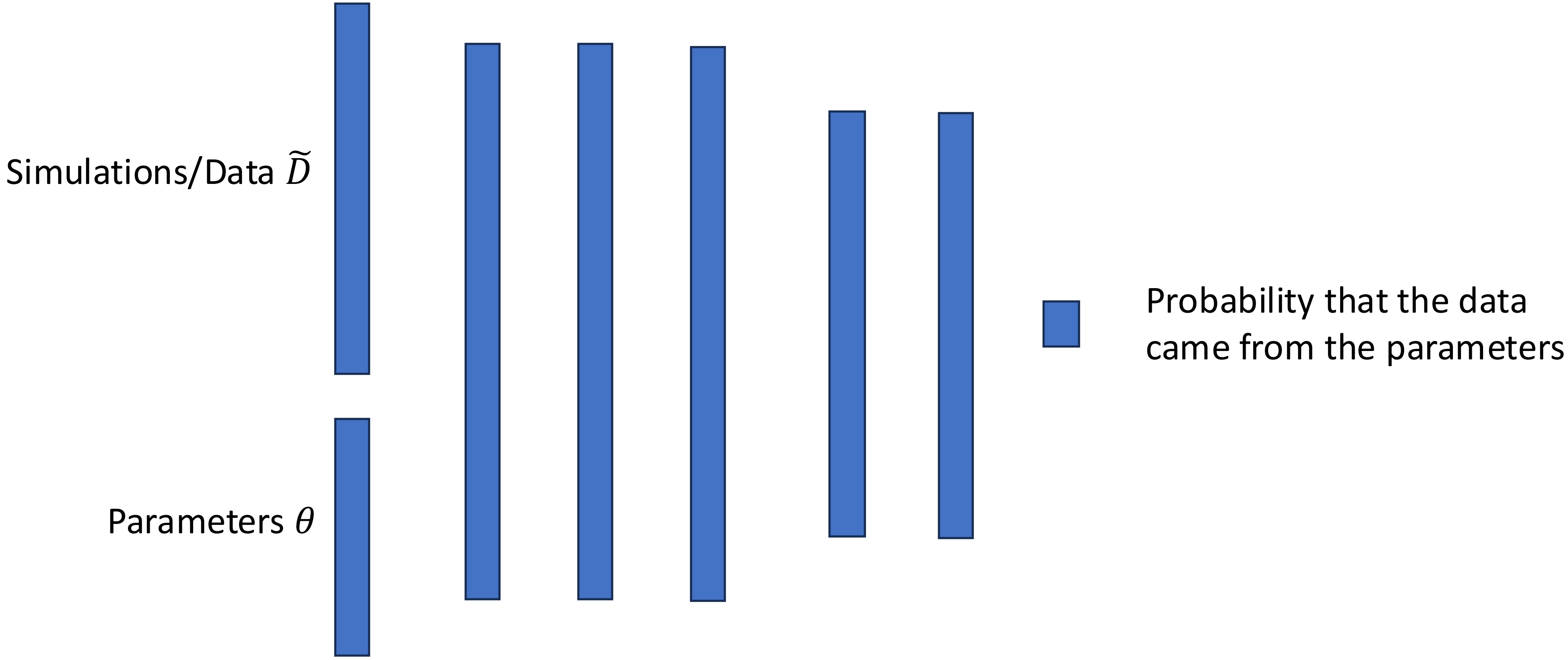
- NREs are essentially classifiers
- We train with pairs of data and parameters that go together with a label (probability) of 1 and pairs that do not go together with label 0
- Use a binary cross entropy loss function

$$L = \frac{1}{N}\sum_i \mathrm{y_i}\log\left(\sigma \cdot f\left(\widetilde{D}_i, \theta_i\right)\right) + (1 - y_i)\log\left(1 \ - \sigma \cdot f\left(\widetilde{D}_i, \theta_i\right)\right)$$

- Where $y_i = 1\ (0)$ for matched (mis-matched) data and parameters and $f\left(\widetilde{D}_i, \theta_i\right)$ is the predicted probability that the data goes together or not

UNIVERSITY OF CAMBRIDGE

As a classification problem



As a classification problem

- We can interpret exponential of $f\left(\tilde{D},\theta\right)$ as

$$r = \frac{P(\tilde{D},\theta)}{P\left(\tilde{D}\right)P(\theta)}$$

- Which is the ratio of the probability that the simulation and parameters are from the joint distribution or are independent

- We can express this in more familiar language as

$$r = \frac{P(\tilde{D},\theta)}{P\left(\tilde{D}\right)P(\theta)} = \frac{P(\tilde{D}|\theta)P(\theta)}{P\left(\tilde{D}\right)P(\theta)} = \frac{L(\theta)\pi(\theta)}{Z\,\pi(\theta)} = \frac{L(\theta)}{Z}$$

As a classification problem

- The network learns what kind of data different sets of parameters produce

- We then feed the network the real data that we have observed and samples of θ from a prior to recover a posterior as

$$P(\theta|D) = r\,\pi(\theta) = \frac{L(\theta)}{Z}\,\pi(\theta)$$

- Watch out some python packages implement their own sampling algorithms!

- NREs are amortized like NPEs

UNIVERSITY OF CAMBRIDGE

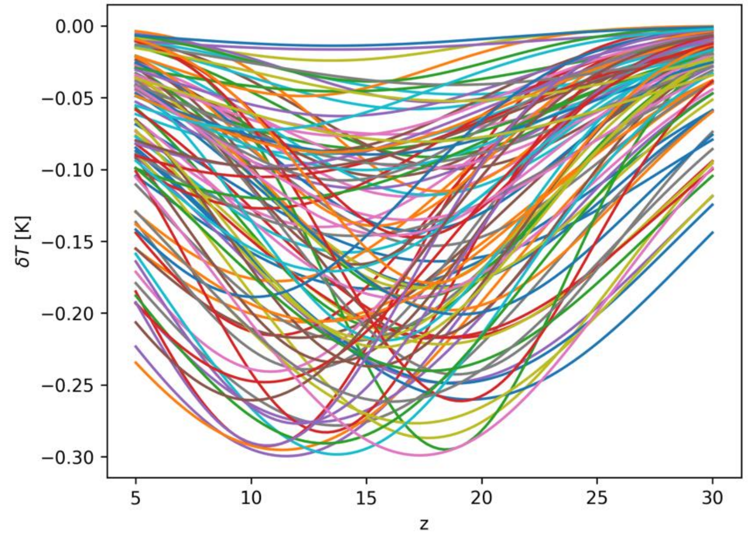
An Example: 21cm Cosmology

- Lets do the same example again

- Generate a set of example signals with matching parameters $s = \{\tilde{D}(\theta), \theta\}$ and assign them a label of 1

- Shuffle the parameters and signals so that we have $s' = \{\tilde{D}(\theta), \phi\}$ with label 0

- No data compression here (although it is often needed)

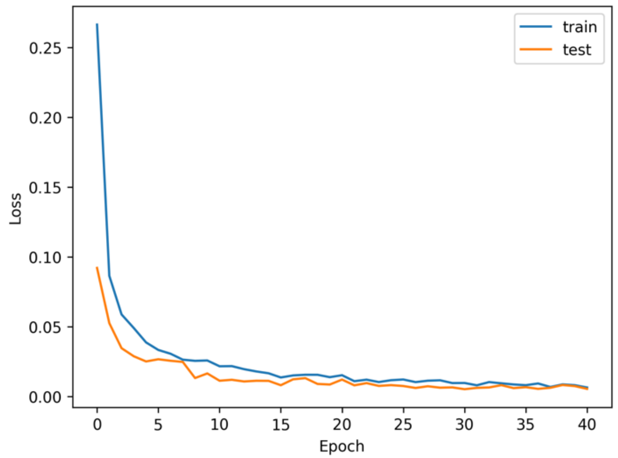


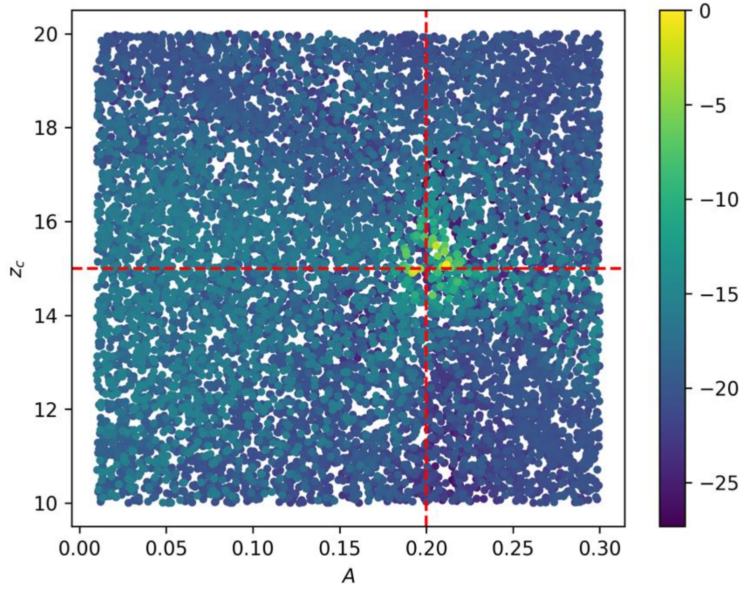

```python
norm_signals = (signals - signals.mean())/signals.std()
norm_params = (theta- theta.mean(axis=0))/theta.std(axis=0)

norm_data = np.hstack([norm_signals, norm_params])
norm_labels = np.ones(n)

idx = np.arange(0, len(norm_data))
random.shuffle(idx)
shuffled_params = norm_params[idx, :]
shuffled_data = np.hstack([norm_signals, shuffled_params])
shuffled_labels = np.zeros(n)

data = np.vstack([shuffled_data, norm_data])
labels = np.hstack([shuffled_labels, norm_labels])

idx = np.arange(0, 2*n)
random.shuffle(idx)
data = data[idx, :]
labels = labels[idx]
```

UNIVERSITY OF CAMBRIDGE

An Example: 21cm Cosmology

- Train the NRE
- Sample a prior and evaluate $P(\theta|D)$ using the NRE
- Here I am just sampling uniformly across the prior
- But really want to wrap this inside an NS or MCMC implementation





Summary

Simulation Based Inference

- A growing field of interest
- Advantageous when we cannot analytically write a likelihood function
- But strongly dependent on the accuracy of our simulations
- We are still learning about how SBI methods scale to higher dimensions and cope with model misspecification

UNIVERSITY OF CAMBRIDGE

Join the research group!

- You've heard a lot from a lot of people during this course!
- We are always interested in prospective PhD students
- The group works on a diverse range of topics so hopefully we are doing something that interests you!
- Related opportunities in Will Handley's Group (Cavendish/Kavli) and Anastasia Fialkov's Group (IoA/Kavli)
- Remember there are code examples at https://github.com/htjb/Talks/tree/master/Lectures/MPhil_Data_Intensive_Science_Lectures

Will Handley's Group

Eloy's Group

Anastasia Fialkov Group