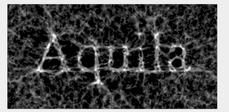
Parameter inference using neural networks

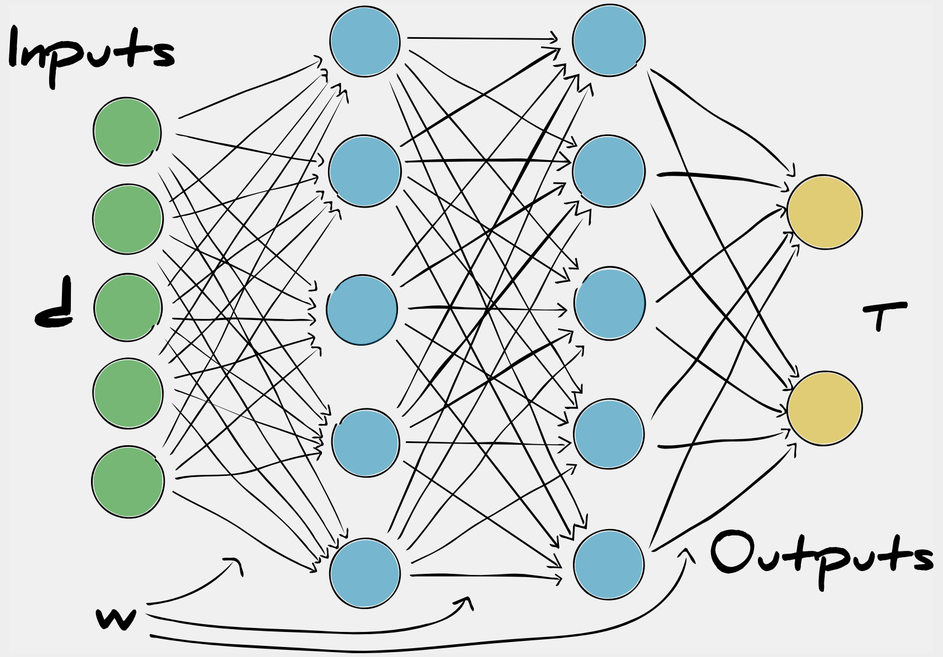
Tom Charnock

Institut d'Astrophysique de Paris



What is a neural network?

What is a neural network?



$$\mathbb{NN}(w, \boldsymbol{\alpha}) : \mathbf{d} \rightarrow \boldsymbol{\tau}$$

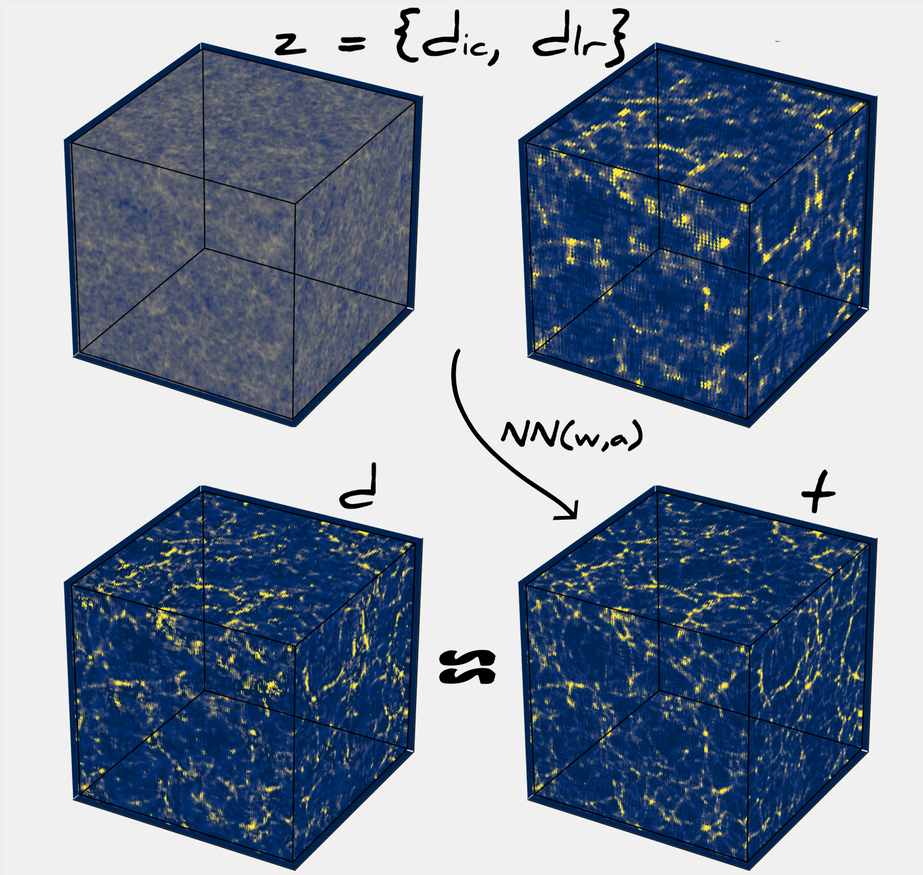
An approximation to a model, $\mathcal{M} : \mathbf{d} \rightarrow \mathbf{t}$

What are they used for?

Emulation (generative networks)

$$\mathbb{NN}(w, \boldsymbol{\alpha}) : \mathbf{z} \rightarrow \mathbf{t}$$

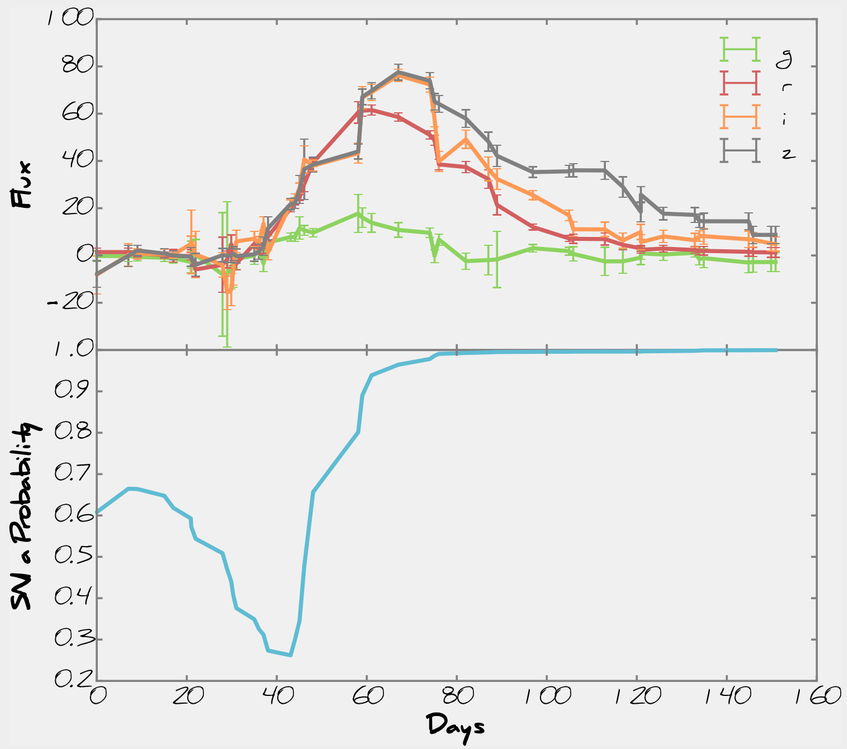
$$\mathbf{t} \in \mathcal{P}(\mathbf{t}|\mathbf{z}) \simeq \mathbf{d} \in \mathcal{P}(\mathbf{d})$$



Classification

$$\mathbb{NN}(w, \alpha) : \mathbf{d} \rightarrow \mathbf{t}$$

$$\mathbf{t} \in \mathcal{P}(\mathbf{t}) \simeq \mathbf{d} \in \mathcal{P}(\text{type}|\mathbf{d})$$



Extremely popular for cosmological parameter estimation

Unfortunately...

Unfortunately...

None of these work in a scientifically rigourous way!

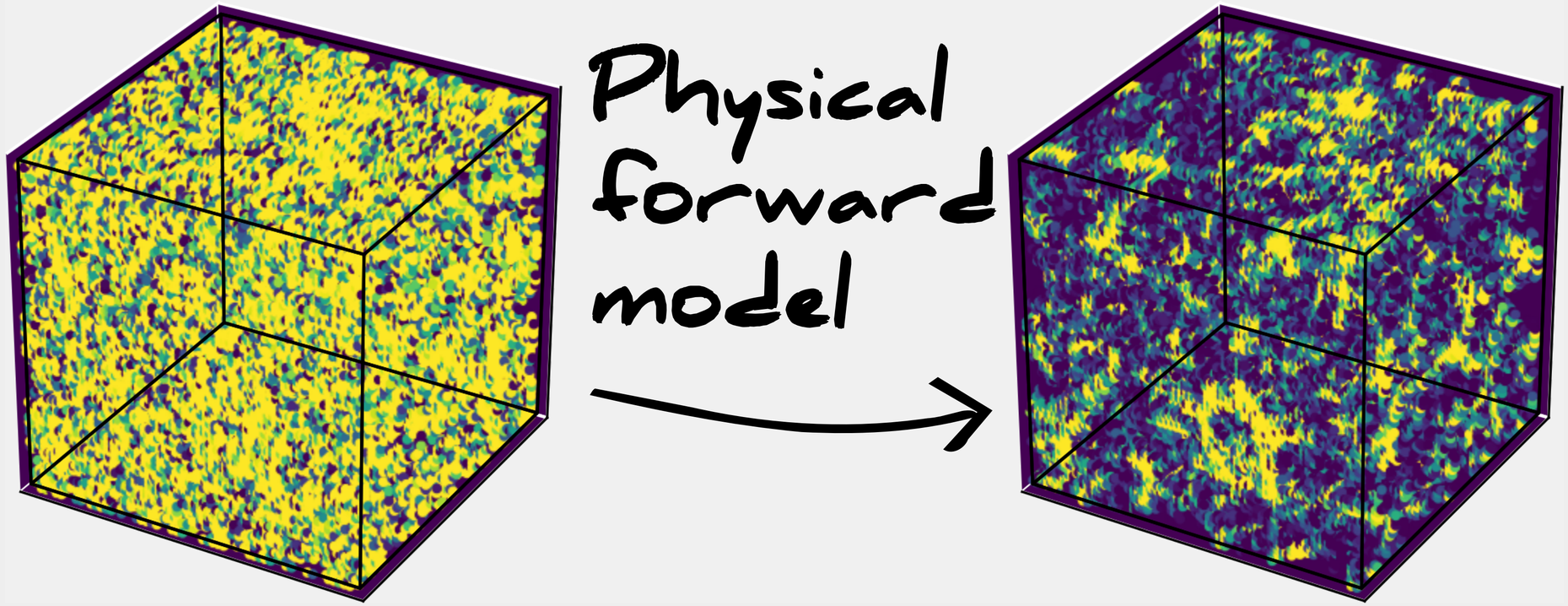
To see why we need to do some stats

A physical model $\mathcal{M} : \theta \rightarrow \mathbf{d}$

The generator of data ($\mathbf{d}$) with parameters (θ)

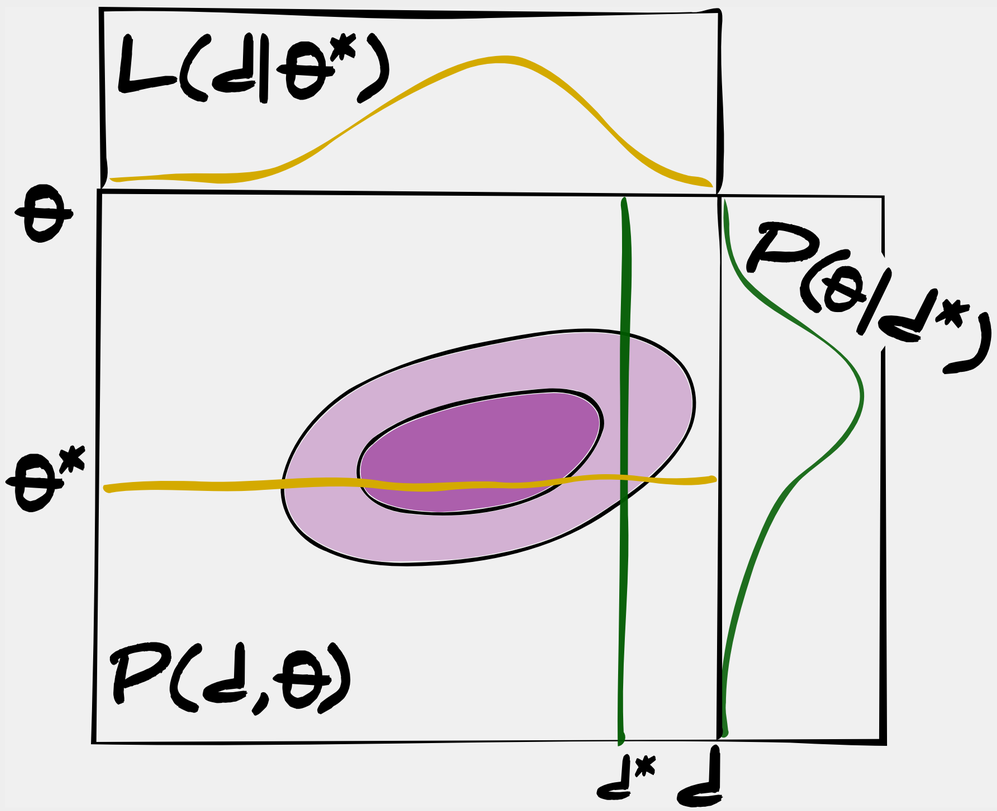
A physical model $\mathcal{M} : \theta \rightarrow \mathbf{d}$

The generator of data ($\mathbf{d}$) with parameters (θ)



The likelihood and the posterior

The likelihood and the posterior

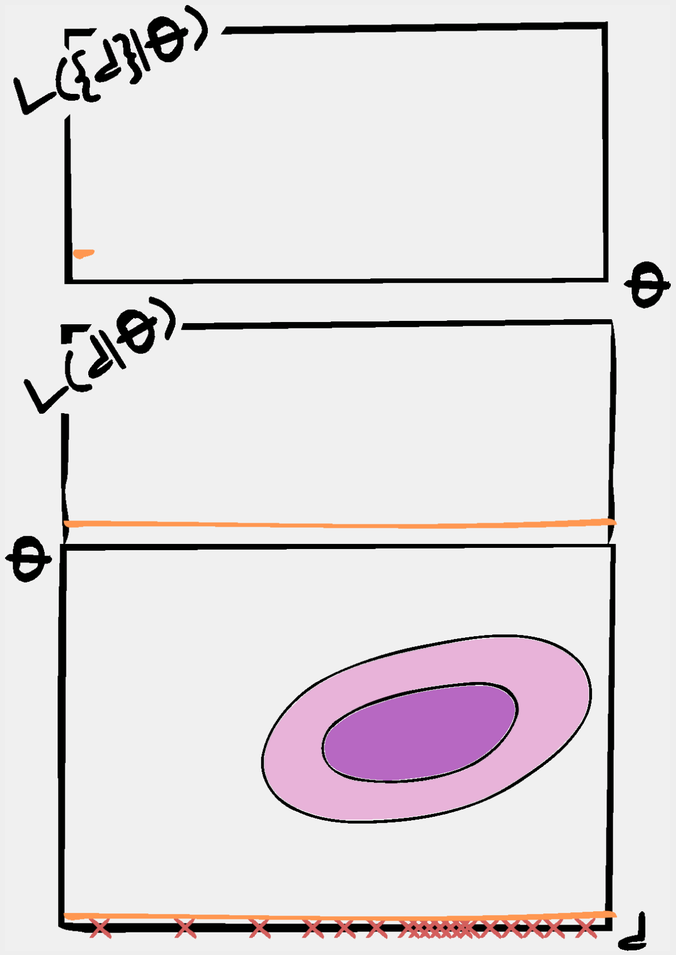


Likelihood $\mathcal{L}(\mathbf{d}|\theta^*)$
What is the probability that the model generates data $\mathbf{d}$ given a set of model parameters θ^*?

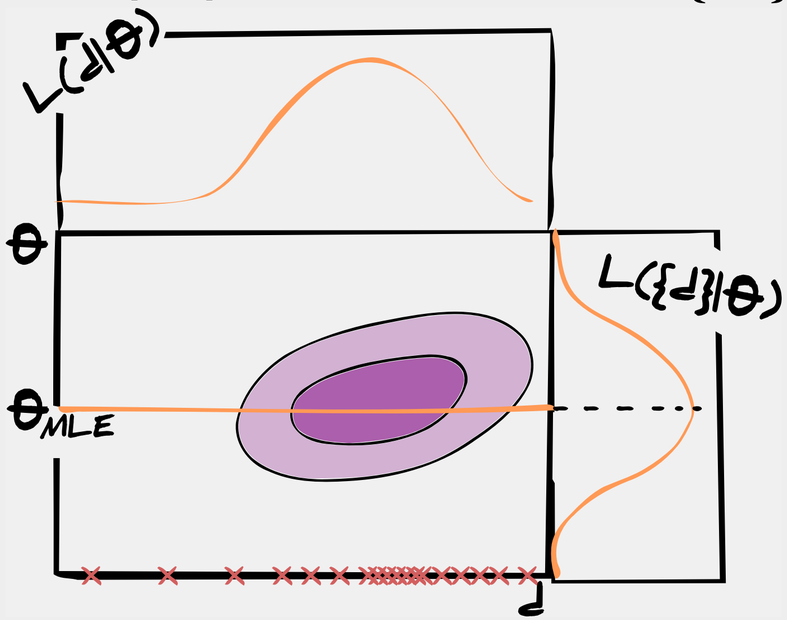
Posterior $\mathcal{P}(\theta|\mathbf{d}^*)$
What is the probability that model parameters θ generate the data $\mathbf{d}^*$?

Maximum likelihood estimates

Maximum likelihood estimates



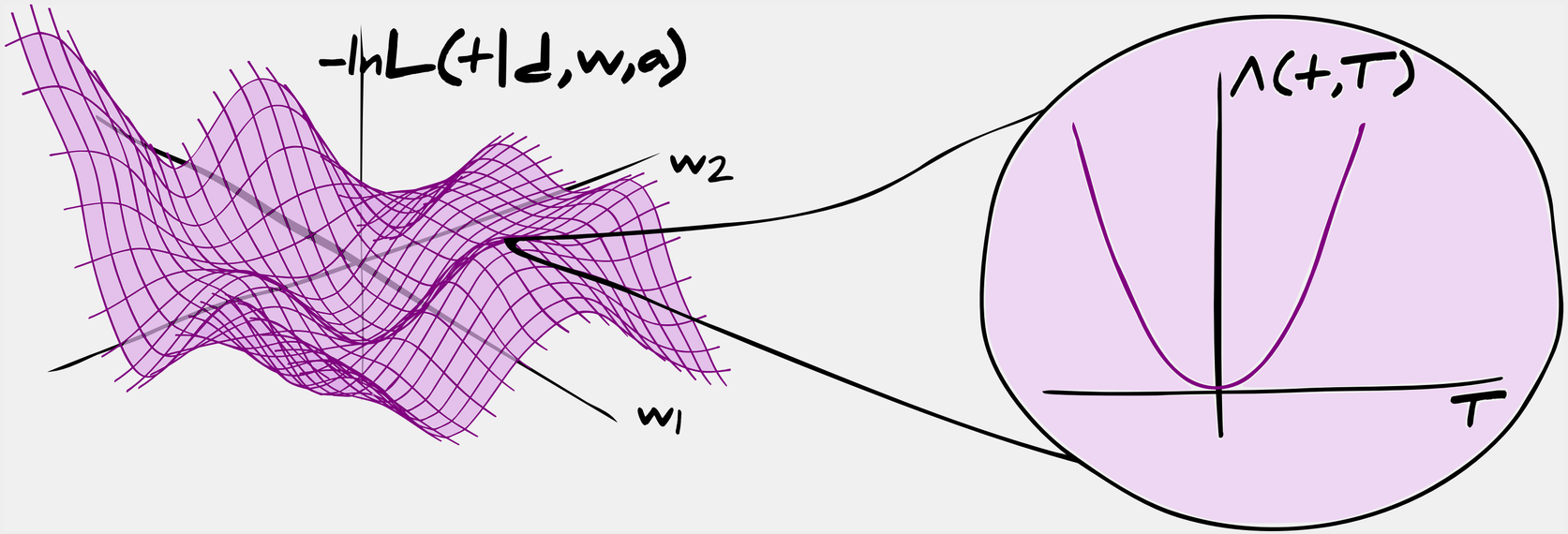
Which value of the model parameters θ is most likely to generate the observed population of data $\{\mathbf{d}\}$?



$$\mathcal{L}(\{\mathbf{d}\}|\theta_{\mathrm{MLE}}) = \sup_{\theta \in \Theta} \mathcal{L}(\{\mathbf{d}\}|\theta)$$

Training a network

Training a network



$\boldsymbol{\Lambda}(\mathbf{t}, \boldsymbol{\tau})$
smooth and convex

$\mathcal{M} : \mathbf{d} \rightarrow \mathbf{t}$ **approximator**

$\mathbb{NN}(w, \boldsymbol{\alpha}) : \mathbf{d} \rightarrow \boldsymbol{\tau}$

$\boldsymbol{\mathcal{L}}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha})$
complex, discrete and possibly non-convex in w and $\boldsymbol{\alpha}$

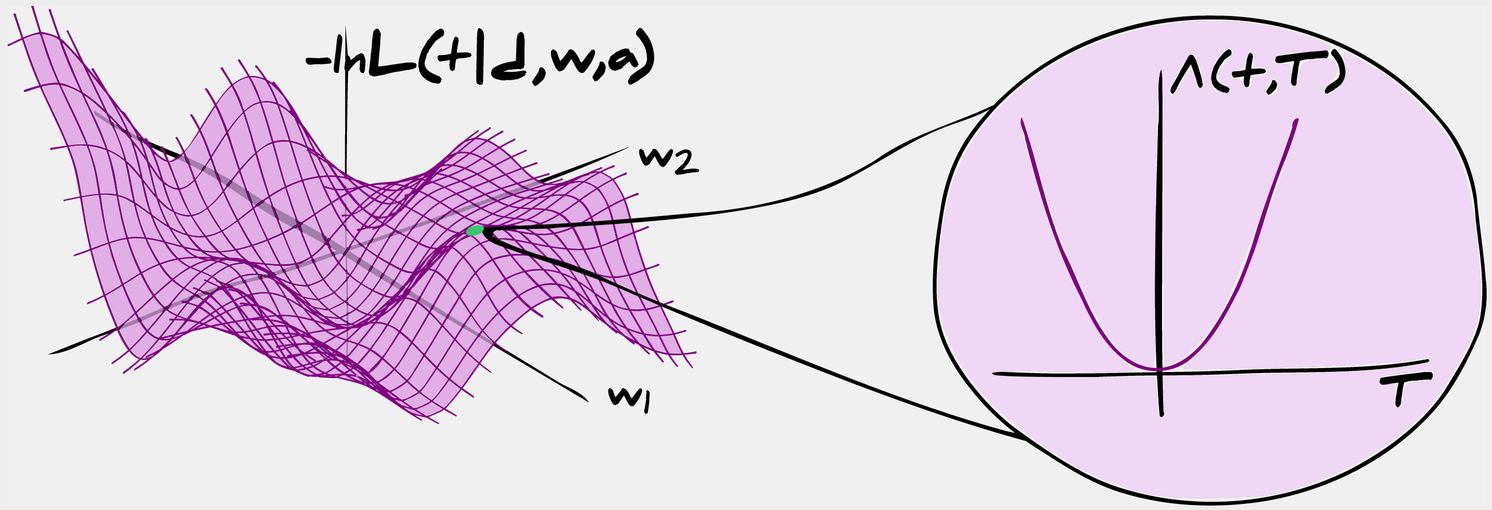
Cost function and likelihood

$$\boldsymbol{\Lambda}(\mathbf{t}, \boldsymbol{\tau}) = -\ln \boldsymbol{\mathcal{L}}(\mathbf{t}|\mathbf{d}, w^*, \boldsymbol{\alpha}^*)$$

Optimising a network

Optimising a network

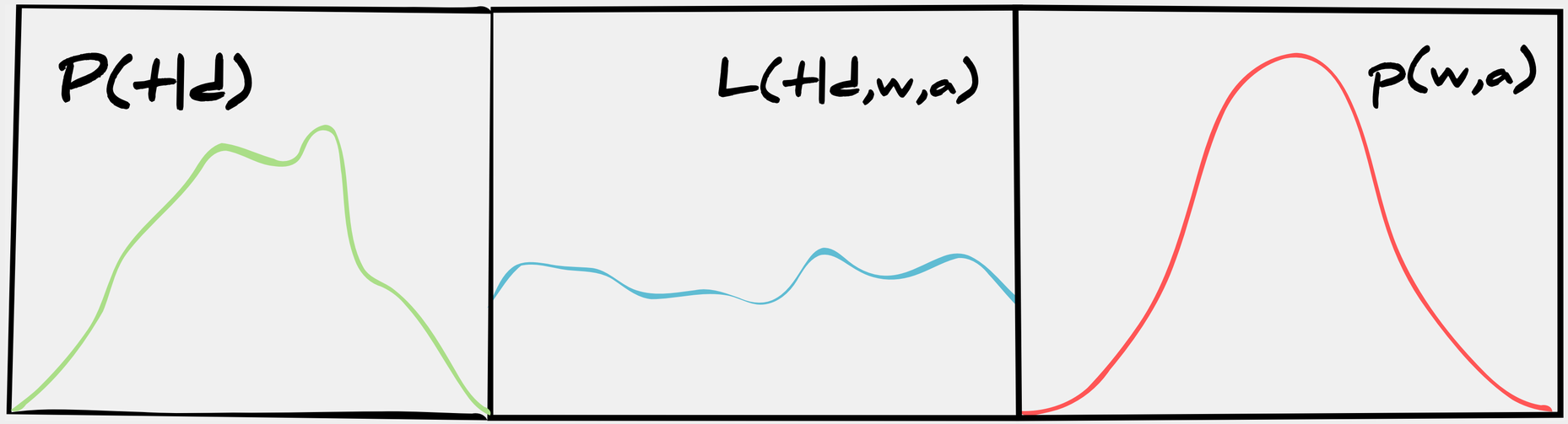
Given a set of training data $\{\mathbf{d}_i^{\text{train}}, \mathbf{t}_i^{\text{train}} | i \in [1, n_{\text{train}}]\}$ what is the maximum likelihood estimate of the weights?



$$w^{\text{MLE}} = \underset{w}{\text{argmax}} \left[\mathcal{L}(\{\mathbf{t}_i^{\text{train}} | i \in [1, n_{\text{train}}]\} | \{\mathbf{d}_i^{\text{train}} | i \in [1, n_{\text{train}}]\right.$$

Where the problem starts

The posterior predictive density $\mathcal{P}(\mathbf{t}|\mathbf{d})$

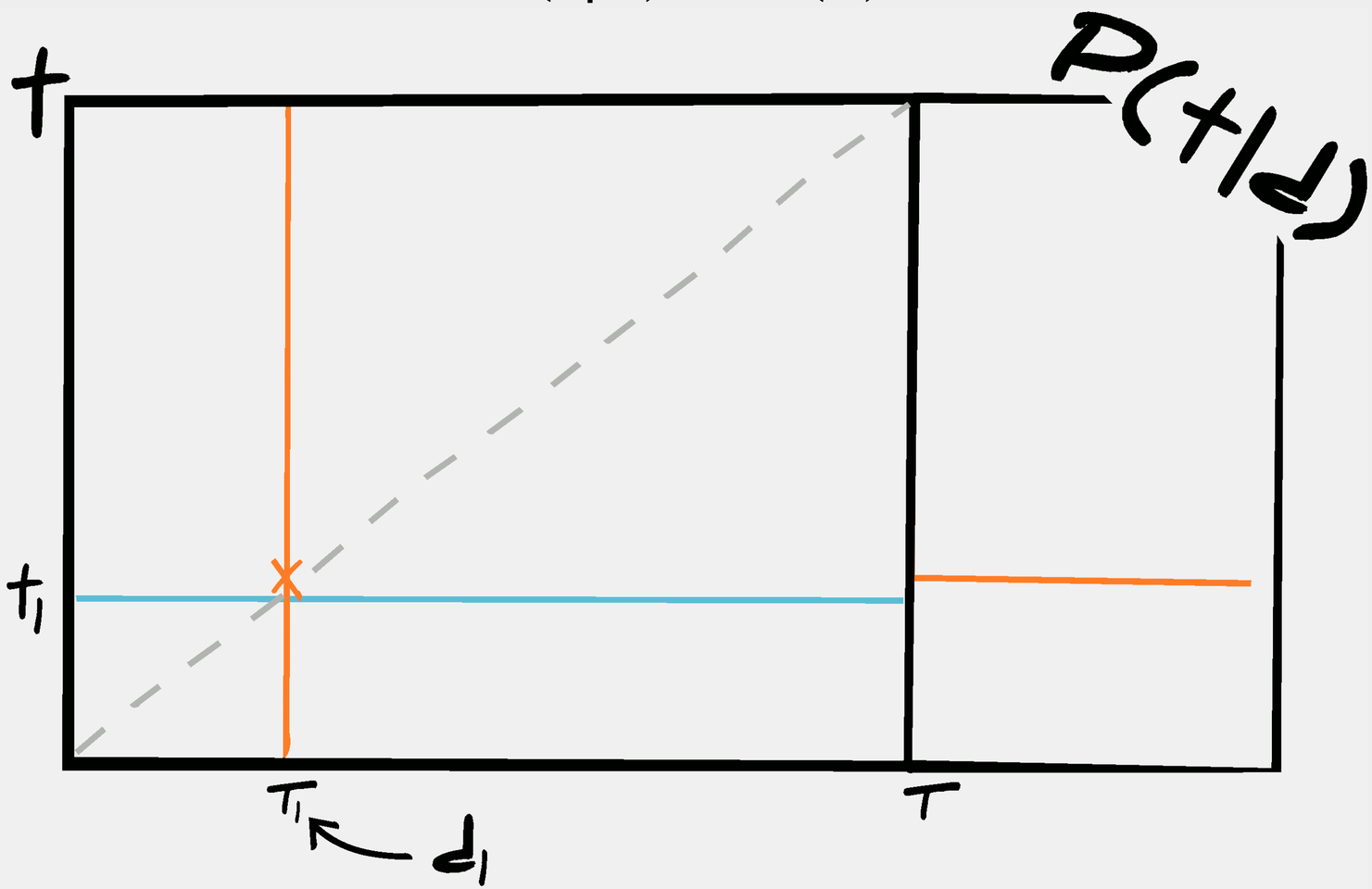


$$\mathcal{P}(\mathbf{t}|\mathbf{d}) = \int d\,w\boldsymbol{\alpha}\mathcal{L}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha})p(w, \boldsymbol{\alpha})$$

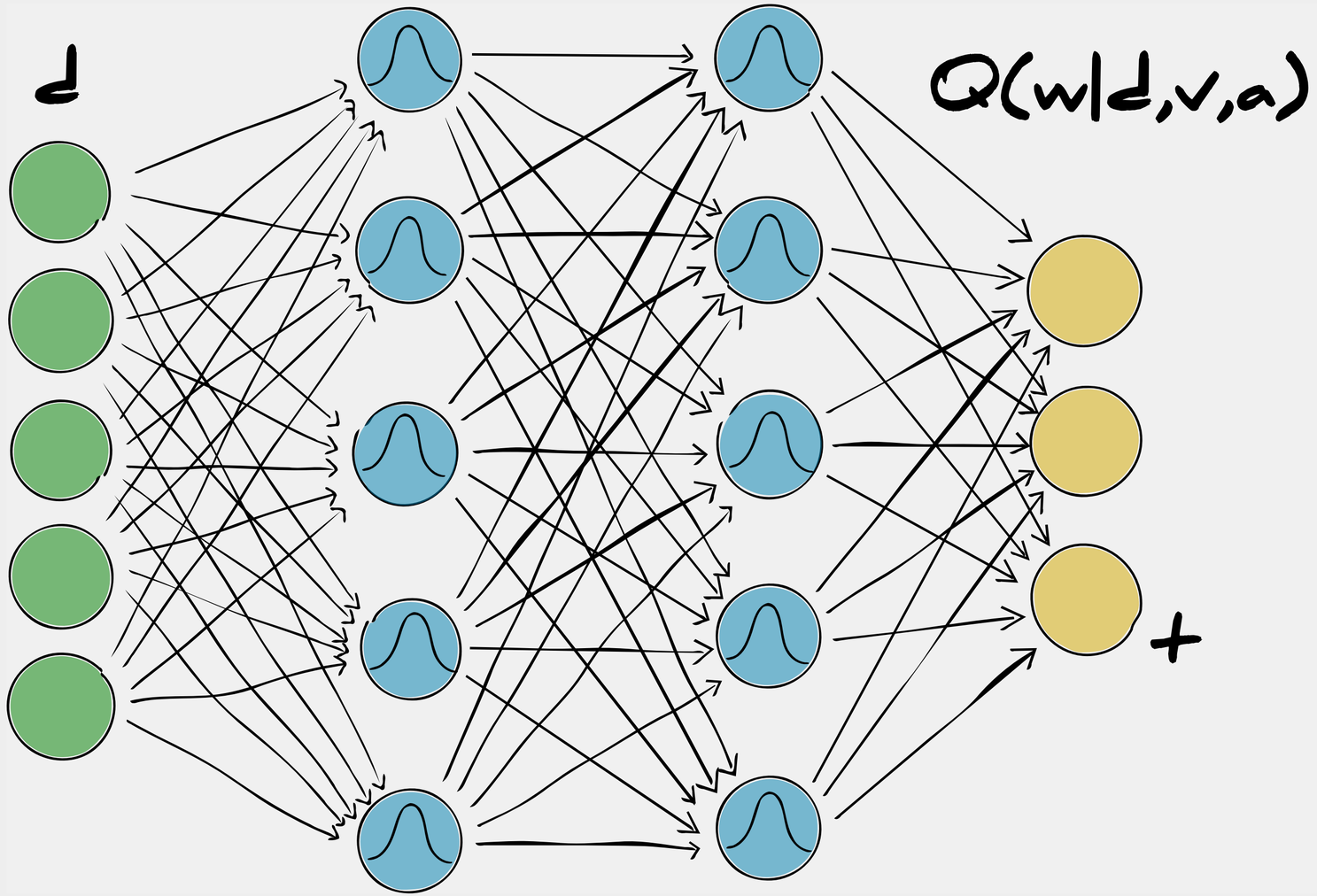
Failure of traditional networks

$$p(w, \boldsymbol{\alpha}) = \delta(w - w^{\text{local MLE}}, \boldsymbol{\alpha} - \boldsymbol{\alpha}^*)$$

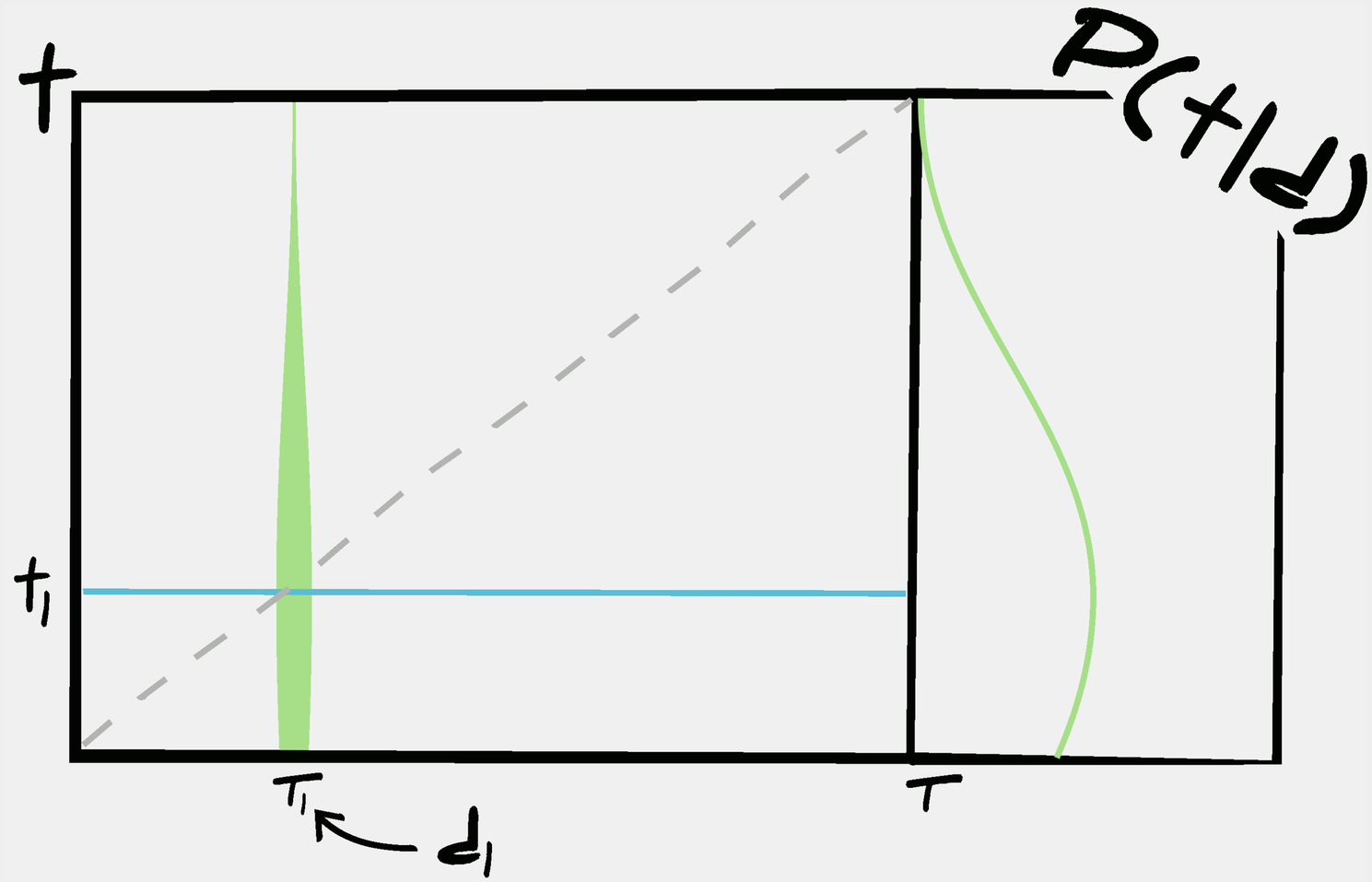
$$\mathcal{P}(\mathbf{t}|\mathbf{d}) = \delta(\tau)$$



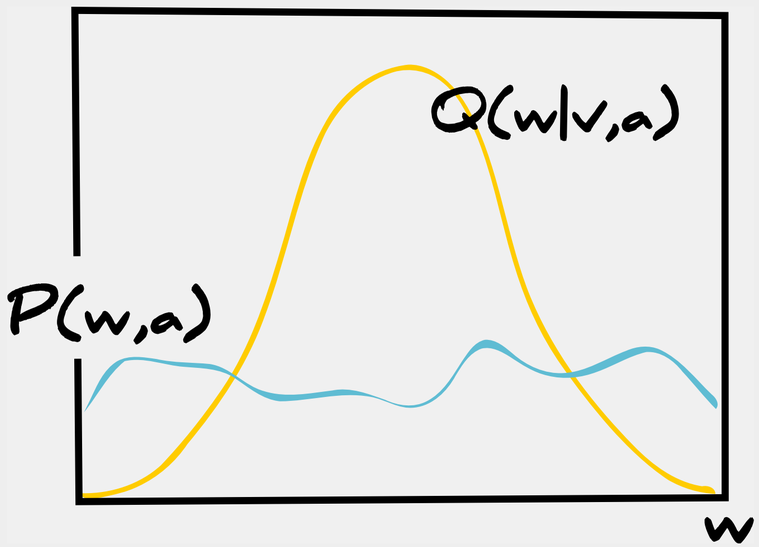
Variational inference



$$\mathcal{P}(\mathbf{t}|\mathbf{d}) = \int dw dv d\boldsymbol{\alpha}\, \mathcal{L}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha}) \mathcal{Q}(w|v, \boldsymbol{\alpha}, \{\mathbf{d}_i^{\text{train}}, \mathbf{t}_i^{\text{train}} | i \in [1, n_{\text{train}}]\}) p(v, \boldsymbol{\alpha})$$



Still depends on training the complex likelihood surface and choice of variational distribution



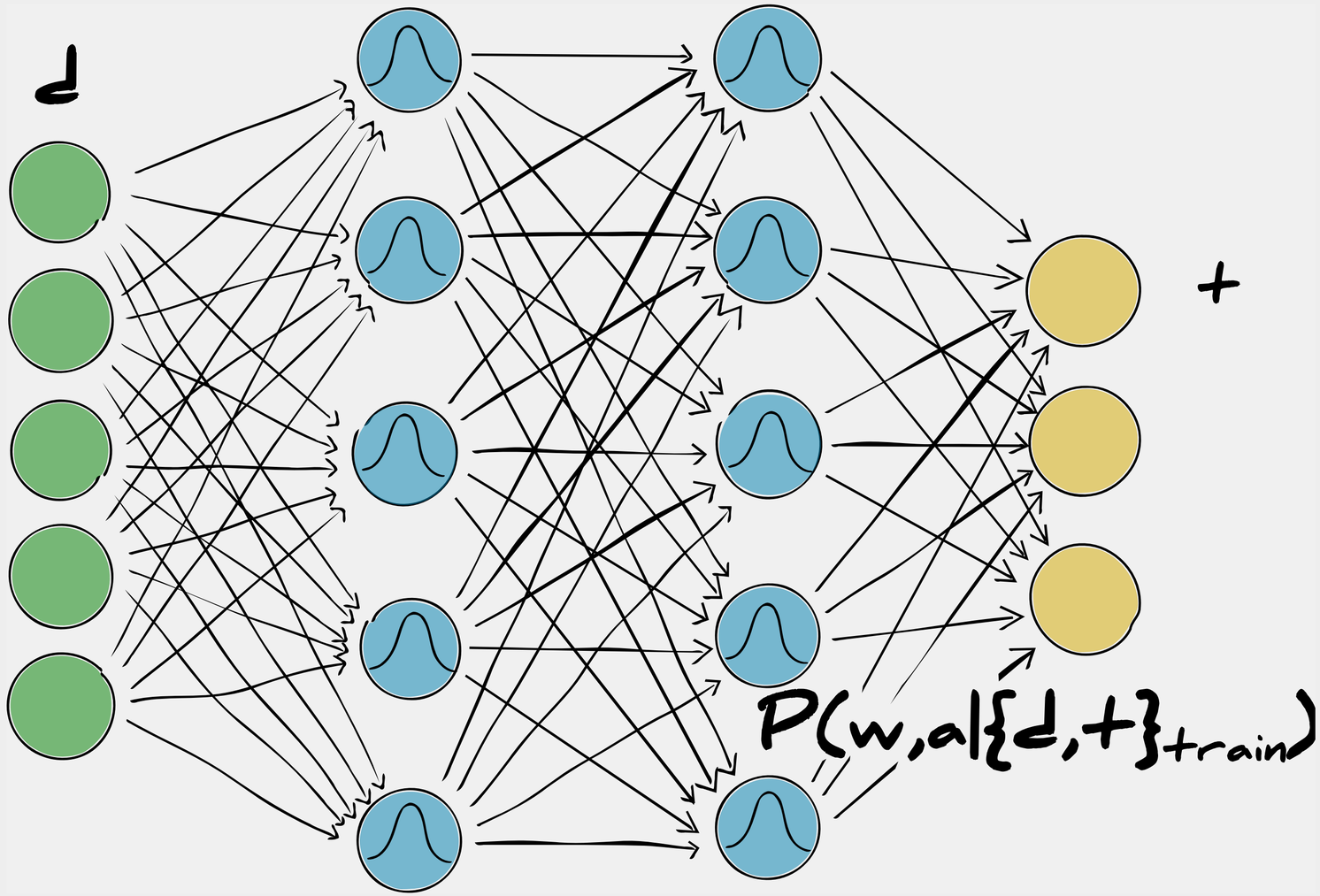
$$\mathcal{P}(\mathbf{t}|\mathbf{d}) = \int dw dv d\boldsymbol{\alpha}\, \mathcal{L}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha}) \mathcal{Q}(w|v, \boldsymbol{\alpha}, \{\mathbf{d}_i^{\text{train}}, \mathbf{t}_i^{\text{train}} | i \in [1, n_{\text{train}}]\})$$

$$\times \delta(v - v^{\text{local MLE}}, \boldsymbol{\alpha} - \boldsymbol{\alpha}^*)$$

$$= \int dw\, \mathcal{L}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha}^*) \mathcal{Q}(w|v^{\text{local MLE}}, \boldsymbol{\alpha}^*, \{\mathbf{d}_i^{\text{train}}, \mathbf{t}_i^{\text{train}} | i \in [1, n_{\text{train}}]\}).$$

Bayesian neural networks

Bayesian neural networks



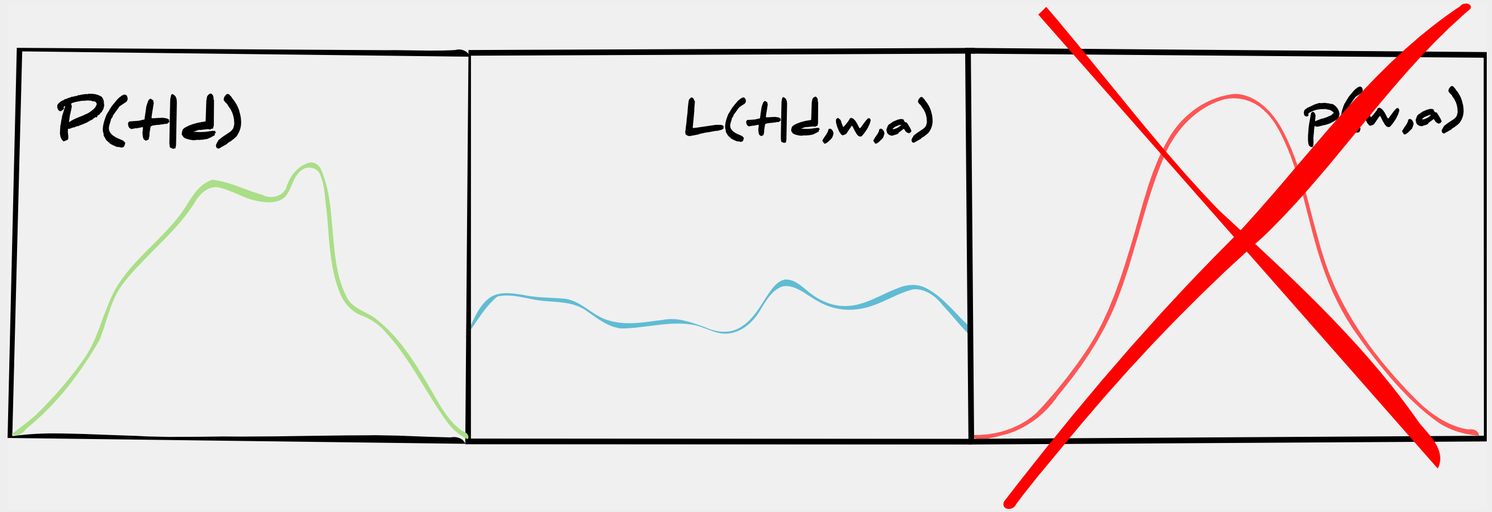
$$\begin{aligned}\mathcal{P}(\mathbf{t}|\mathbf{d}) &= \int dw d\boldsymbol{\alpha}\ \mathcal{L}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha}) \mathcal{P}(w, \boldsymbol{\alpha}|\{\mathbf{d}_i^{\text{train}}, \mathbf{t}_i^{\text{train}}|i \in [1, n_{\text{train}}]\}) \\ &= \int dw d\boldsymbol{\alpha}\ \mathcal{L}(\mathbf{t}|\mathbf{d}, w, \boldsymbol{\alpha}) \prod_i^{n_{\text{train}}} \mathcal{L}(\mathbf{t}_i^{\text{train}}|\mathbf{d}_i^{\text{train}}, w, \boldsymbol{\alpha}) p(w, \boldsymbol{\alpha}).\end{aligned}$$

Still dependent on the training data!

Classical network : $p(w, \boldsymbol{\alpha}) = \delta(w - w^{\mathrm{MLE}}, \boldsymbol{\alpha} - \boldsymbol{\alpha}^*)$

Variational inference : $p(w, \boldsymbol{\alpha}) = \mathcal{Q}(w|v^{\text{local MLE}}, \boldsymbol{\alpha}^*, \{\mathbf{d}_i^{\text{train}}, \mathbf{t}_i^{\text{train}} | i \in [1, n_{\text{train}}]\})$

Bayesian neural networks : $p(w, \boldsymbol{\alpha}) = \prod_i^{n_{\text{train}}} \mathcal{L}(\mathbf{t}_i^{\text{train}}|\mathbf{d}_i^{\text{train}}, w, \boldsymbol{\alpha}) p(w, \boldsymbol{\alpha})$



How can we use a neural network then?

Build it into the physical model

Method 1 : Model extension

Generate initial conditions
of dark matter distribution

$P(\hat{\delta}_{ic}|\Omega)$

Draw power spectrum from
prior cosmological prameters

δ_{ic}

Evolve forward to dark
matter distribution today

δ_{LPT}

Use neural physical engine to obtain
informative summaries about the
abundance of halos of
a certain mass

Ψ_{NPE}

observed
catalogue

	X	Y	Z	M
1				
2				
3				
4				
5				

Neural
bias model

Calculate Poission likelihood between
observed catalogue and the halo mass
distribution function from the neural
density estimators

We infer the neural network

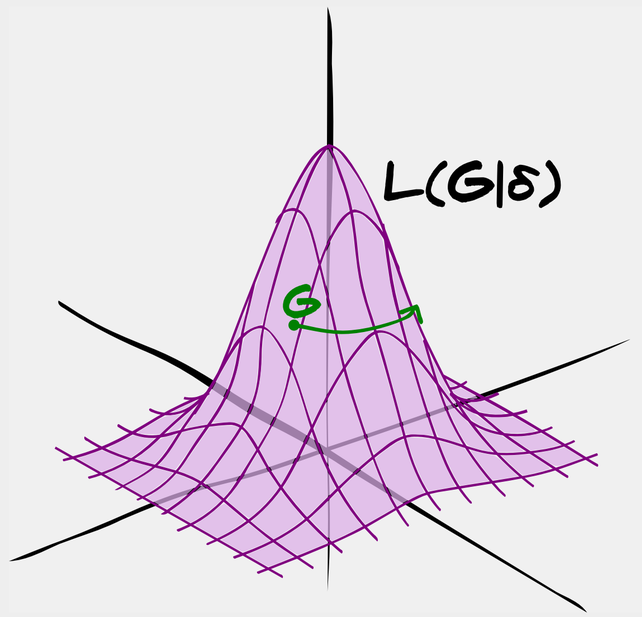
HMCLET

Hamiltonian Monte Carlo

Introduce momentum $\mathbf{p} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{M})$ and solve

Hamilton's equations

$$\begin{aligned}\mathcal{H}(\theta, \mathbf{p}) &= \mathcal{V}(\theta) + \mathcal{K}(\mathbf{p}) \\ &= \mathcal{L}(\theta|\delta) - \log[\pi(\theta)] + \frac{1}{2}\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}\end{aligned}$$

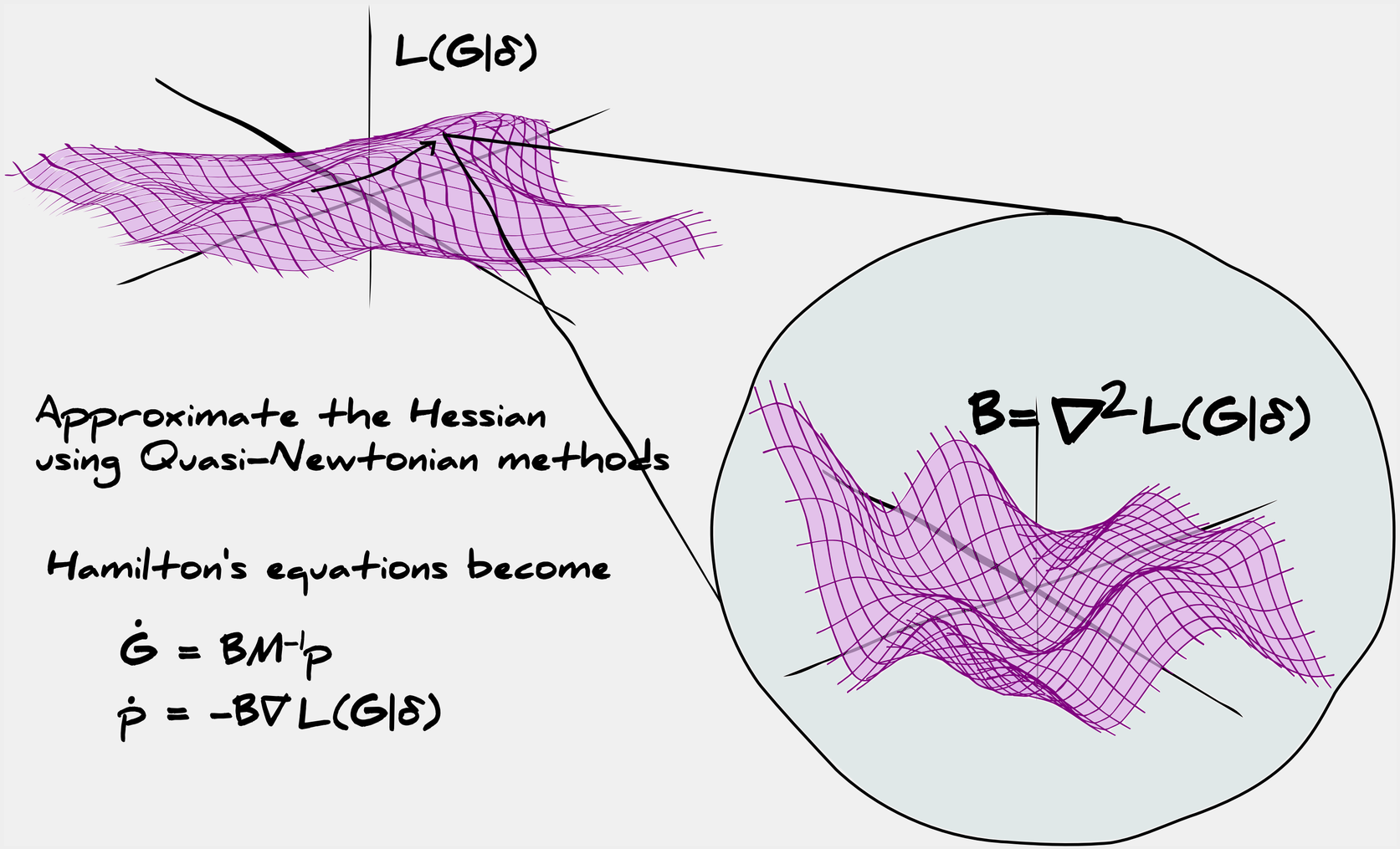


Accept samples according to probability

$$\alpha = \text{Min}\left[\exp(\Delta\mathcal{H}), 1\right]$$

Evolve using $\dot{\theta} = \mathbf{M}^{-1}\mathbf{p}$ and $\dot{\mathbf{p}} = -\nabla\mathcal{V}(\theta)$

Use second order geometric information



Approximate the Hessian
using Quasi-Newtonian methods

Hamilton's equations become

$$\dot{G} = BM^{-1}p$$

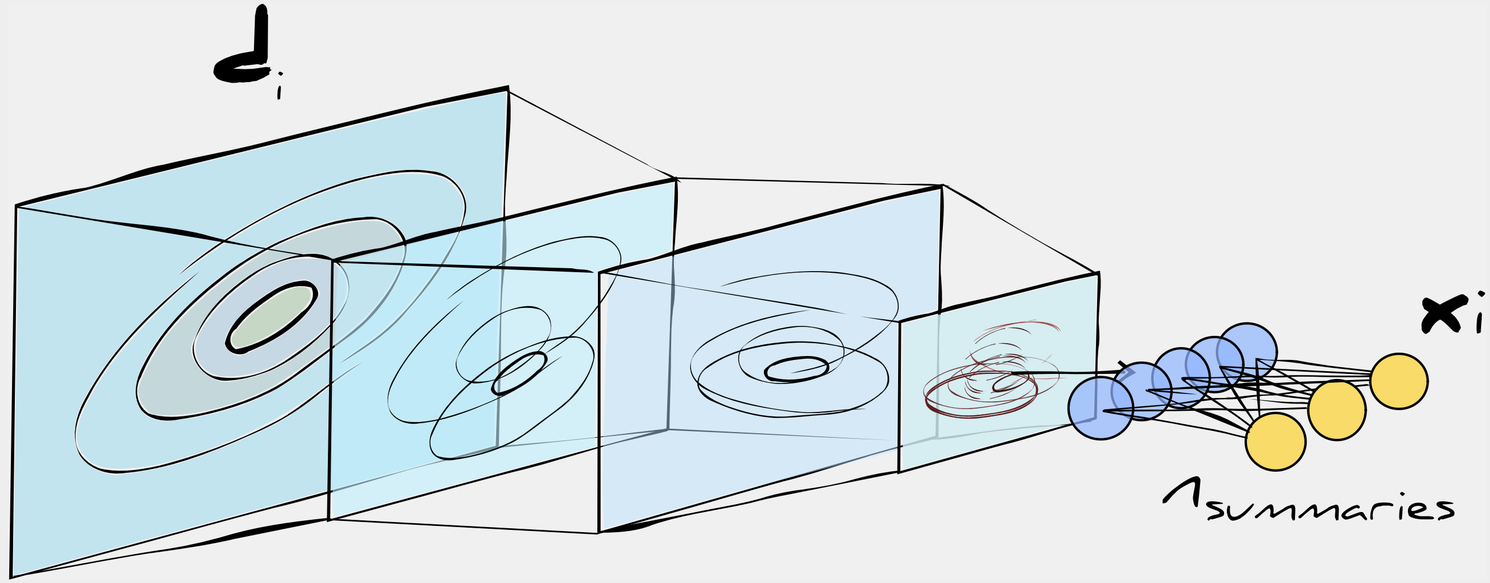
$$\dot{p} = -B\nabla L(G|\delta)$$

Neural physical engines

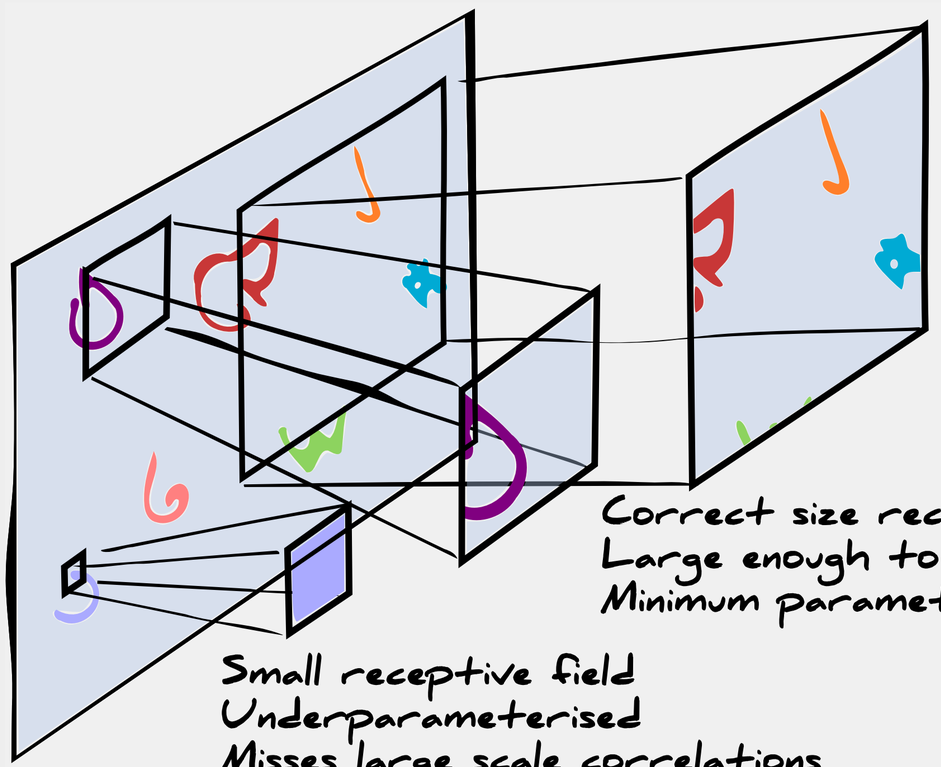
Build networks using physical principles.

- Reduces number of parameters
- Increases computational efficiency
- Decreases overfitting
- Improves interpretability

Convolutions for translational invariance



Use only causally relevant features

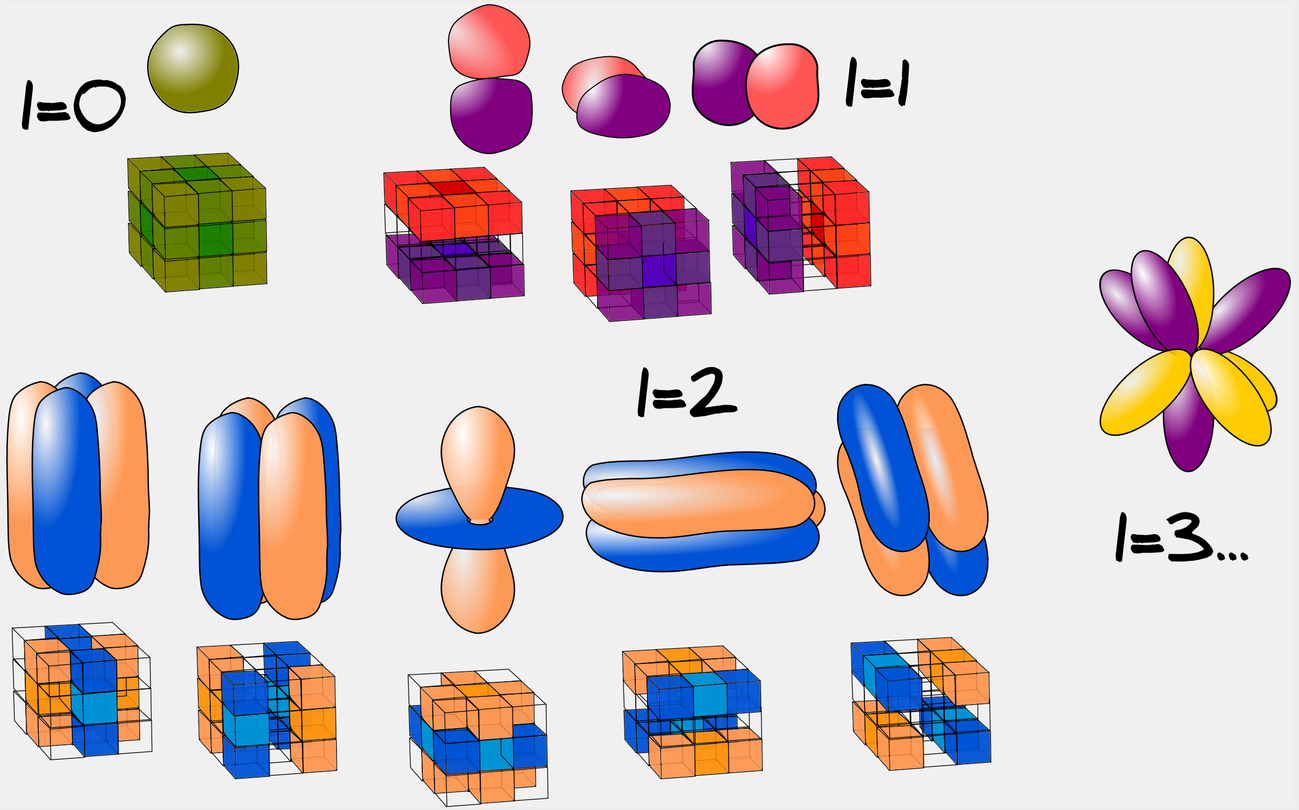


Large receptive field
Overparameterised
Information sparsely represented
Difficult to train

Correct size receptive field
Large enough to capture longest causal correlations
Minimum parameters to maximum information

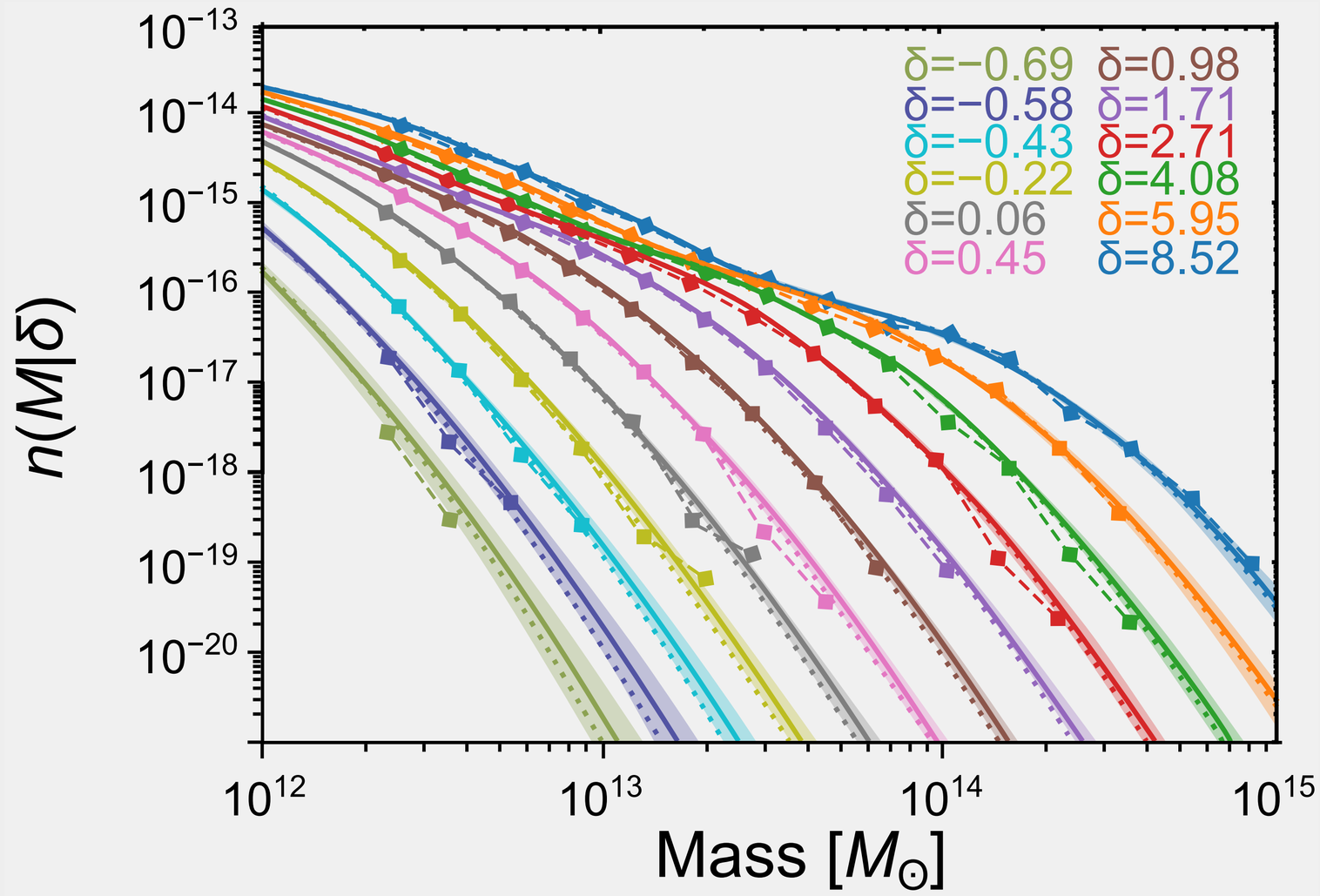
Small receptive field
Underparameterised
Misses large scale correlations
Averages out information content

Use only informative features



github:multipole_kernels

Halo mass distribution function

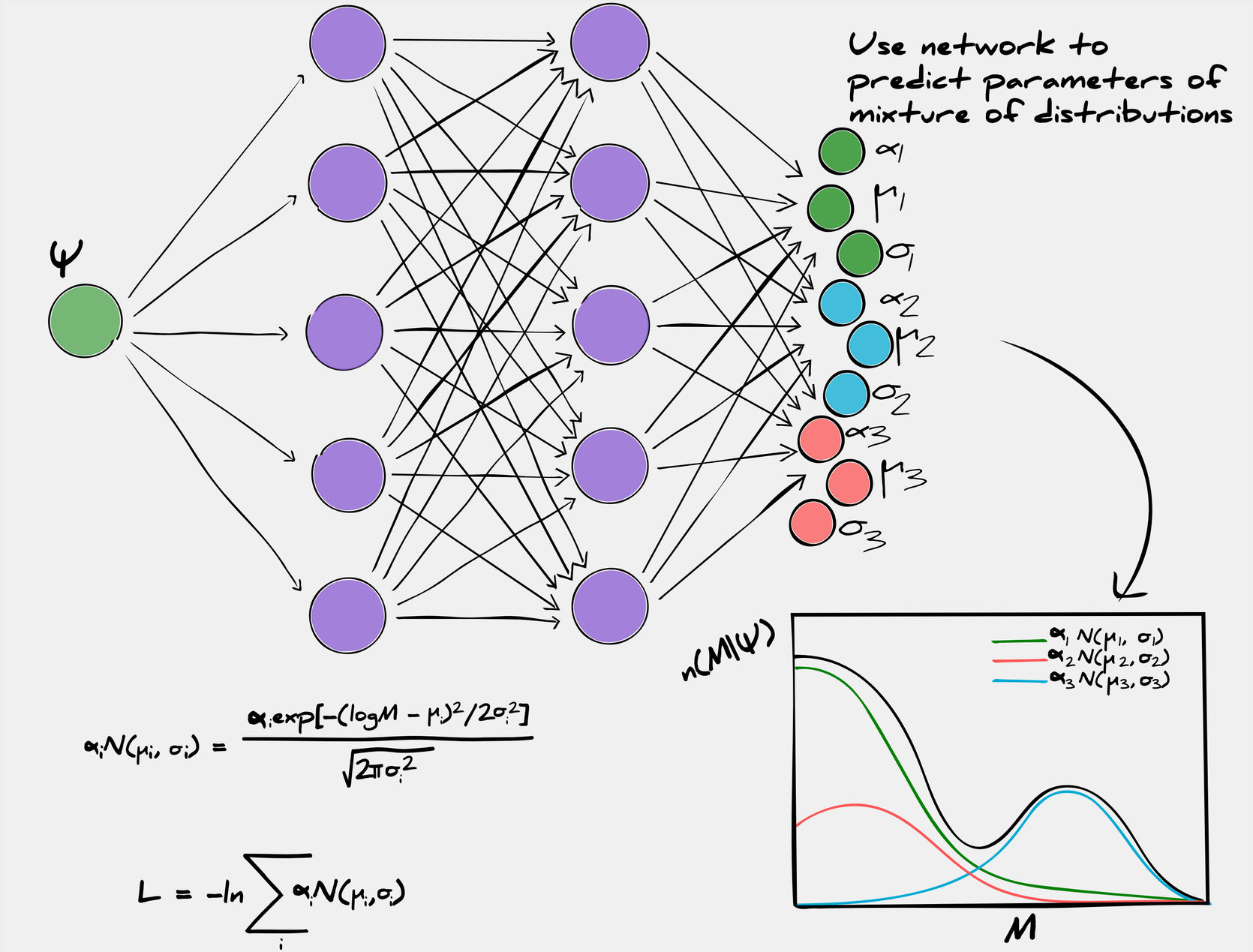


Neural density estimators

Halo mass distribution function is a smooth function of mass given a density environment

Halo mass distribution function is a smooth function of mass given a density environment

Use a mixture density network



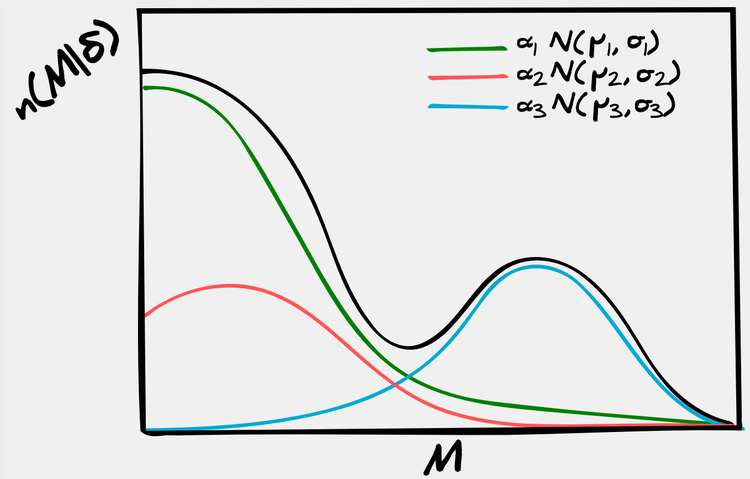
$$\alpha_i N(\mu_i, \sigma_i) = \frac{\alpha_i \exp[-(\log M - \mu_i)^2 / 2\sigma_i^2]}{\sqrt{2\pi\sigma_i^2}}$$

$$L = -\ln \sum_i \alpha_i N(\mu_i, \sigma_i)$$

Our neural density estimator

$$n(M|\delta) = \sum_i^N \alpha(\boldsymbol{\psi}, \theta_i^\alpha) \mathcal{N}\left(\mu(\boldsymbol{\psi}, \theta_i^\mu), \sigma(\boldsymbol{\psi}, \theta_i^\sigma)|M\right),$$

$$= \sum_i^N \frac{\alpha(\boldsymbol{\psi}, \theta_i^\alpha)}{\sqrt{2\pi\left(\sigma(\boldsymbol{\psi}, \theta_i^\sigma)\right)^2}} \exp\left[-\frac{\left(\log(M) - \mu(\boldsymbol{\psi}, \theta_i^\mu)\right)^2}{2\left(\sigma(\boldsymbol{\psi}, \theta_i^\sigma)\right)^2}\right],$$



$$\alpha_i = \text{softplus}(w_i^\alpha \boldsymbol{\psi} + b_i^\alpha),$$

$$\mu_i = \begin{cases} w_i^\mu \boldsymbol{\psi} + b_i^\mu & i = 0 \\ \text{Max}\left[0,\ w_i^\mu \boldsymbol{\psi} + b_i^\mu\right] + \mu_{i-1} & i > 0 \end{cases},$$

$$\sigma_i = \text{softplus}(w_i^\sigma \boldsymbol{\psi} + b_i^\sigma) \quad ,$$

2 Gaussians with

$b^\alpha \to b^\alpha + \log(10^{-3})$, $b_0^\mu \to b_0^\mu + \log\left(2 \times 10^{12}\right)$ and $b^\sigma \to b^\alpha + \log(10^3)$.

Likelihood

Likelihood

$$\mathcal{L} = \sum_{j \in \text{catalogue}} \log\left[\sum_{i}^{N} \frac{\alpha_{i,j}}{\sqrt{2\pi\sigma_{i,j}^2}} \exp\left[-\frac{\left(\log(M_j) - \mu_{i,j}\right)^2}{2\sigma_{i,j}^2}\right]\right]$$

$$-V \sum_{j \in \text{voxels}, i=1}^{N} \frac{\alpha_{i,j}}{2} \exp\left[\frac{\sigma_{i,j}^2}{2}\right] \operatorname{erfc}\left[\frac{\log(M_{\text{th}}) - \mu_{i,j} - \sigma_{i,j}^2}{\sqrt{2\sigma_{i,j}^2}}\right].$$

Generate initial conditions
of dark matter distribution

$P(\hat{\delta}_{ic}|\Omega)$

Draw power spectrum from
prior cosmological prameters

δ_{ic}

Evolve forward to dark
matter distribution today

δ_{LPT}

Use neural physical engine to obtain
informative summaries about the
abundance of halos of
a certain mass

Ψ_{NPE}

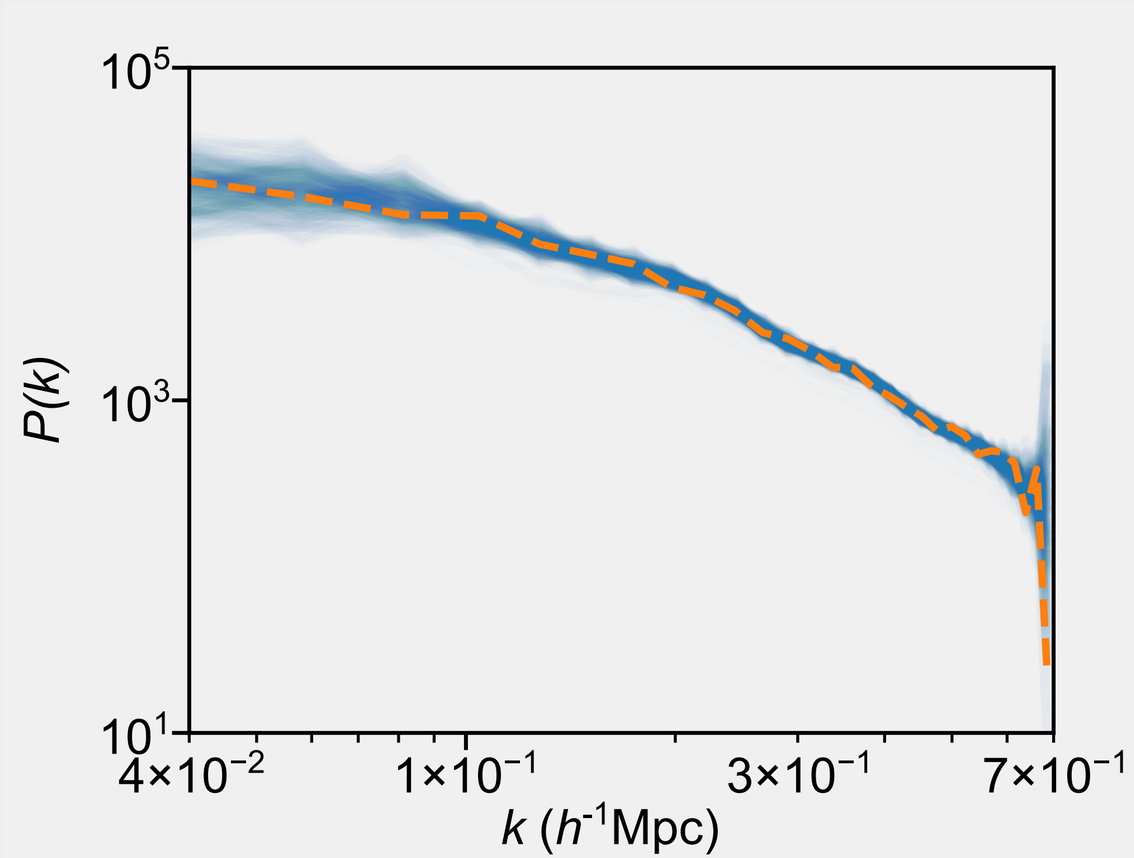
observed
catalogue

	X	Y	Z	M
1				
2				
3				
4				
5				

Neural
bias model

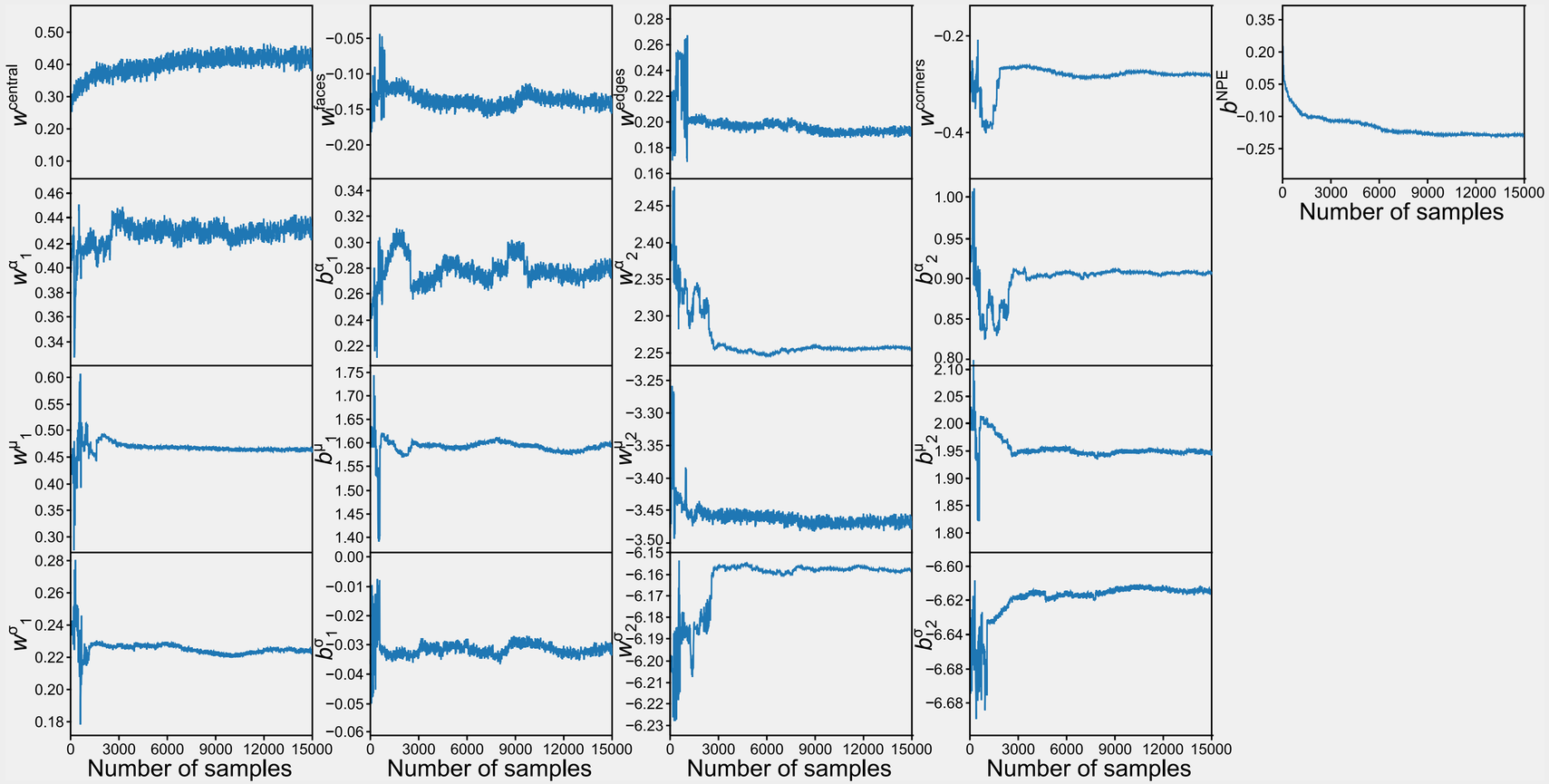
Calculate Poission likelihood between
observed catalogue and the halo mass
distribution function from the neural
density estimators

Sample the power spectrum



✓ Power spectra remain smooth and close to prior

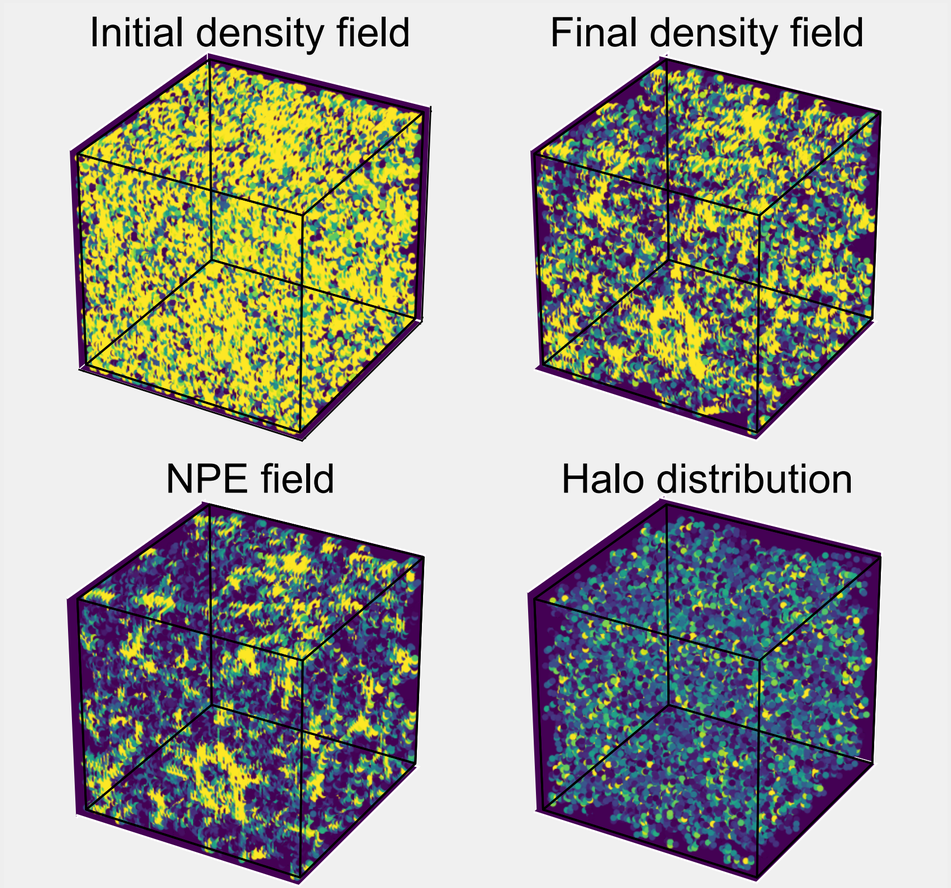
Sample the weights of the neural bias model



✓ Weight values are properly sampled after burn in

✓ NPE acts as a contrast enhancer

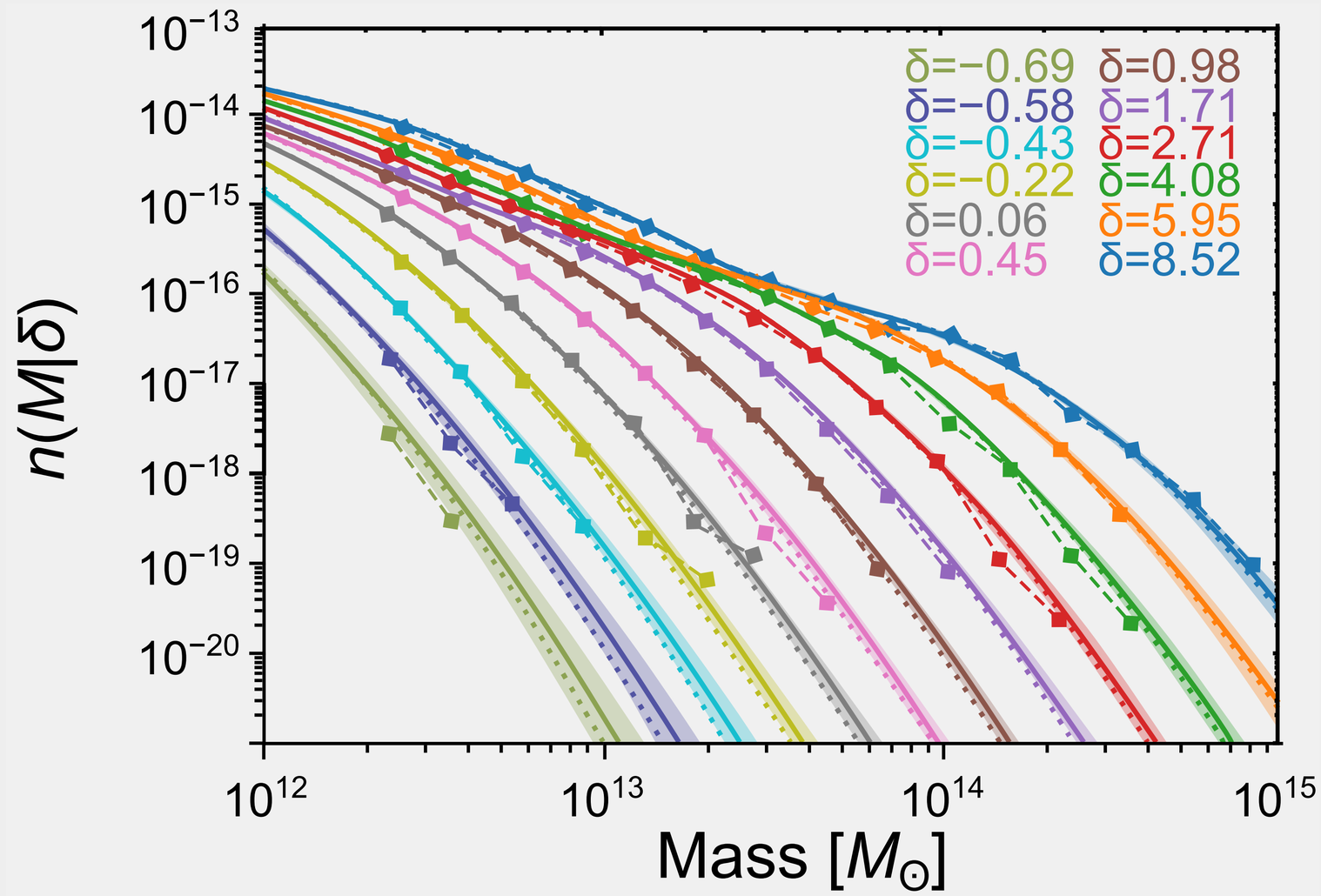
See the effects of the neural physical engine



✓ Non-local information is used to improve fit

And what does the halo mass distribution function look like?

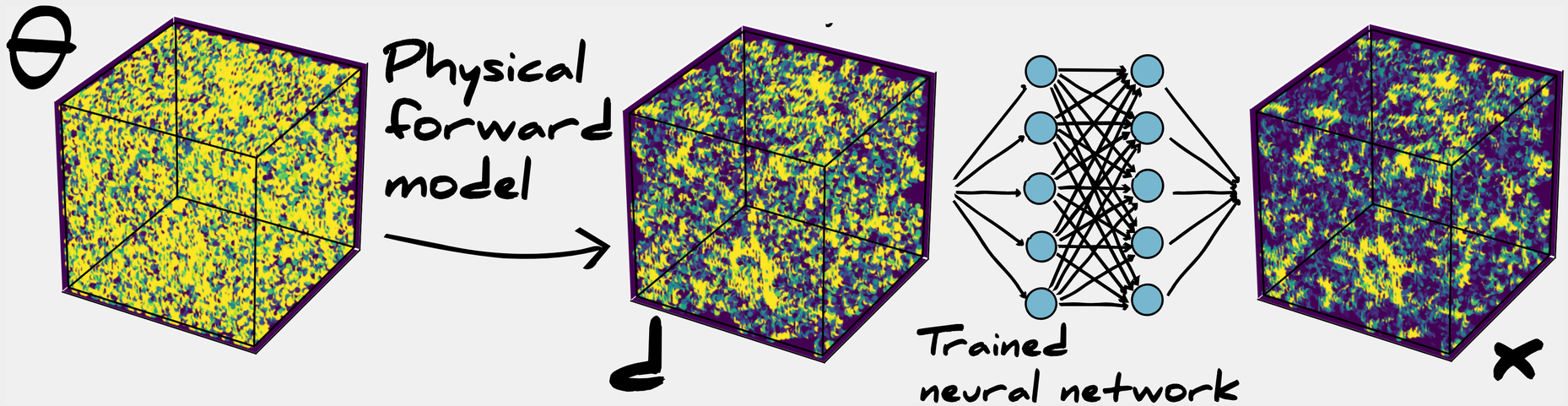
Halo mass distribution function from neural bias model



✓ Fits data!

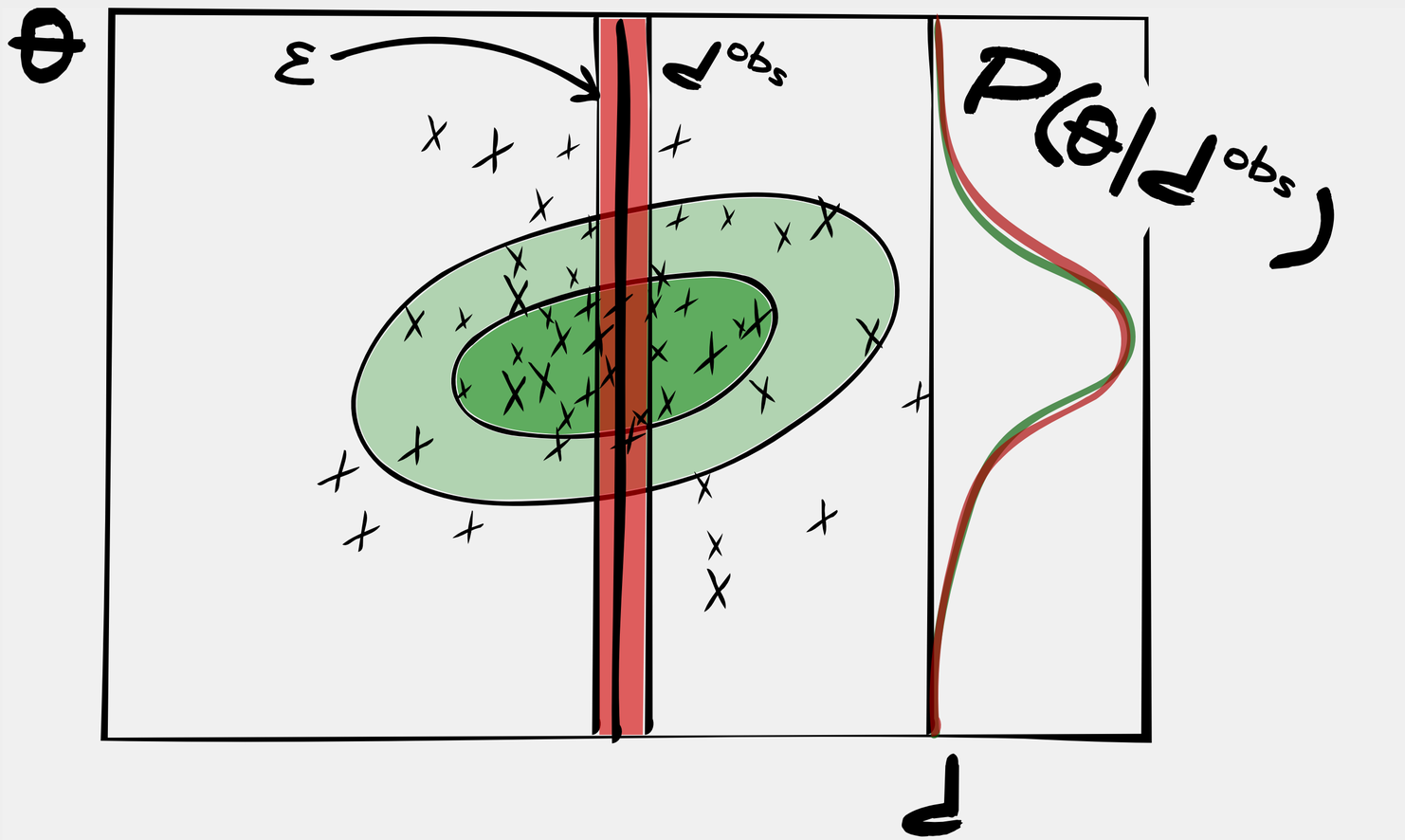
Method 2 : Likelihood-free inference

Method 2 : Likelihood-free inference

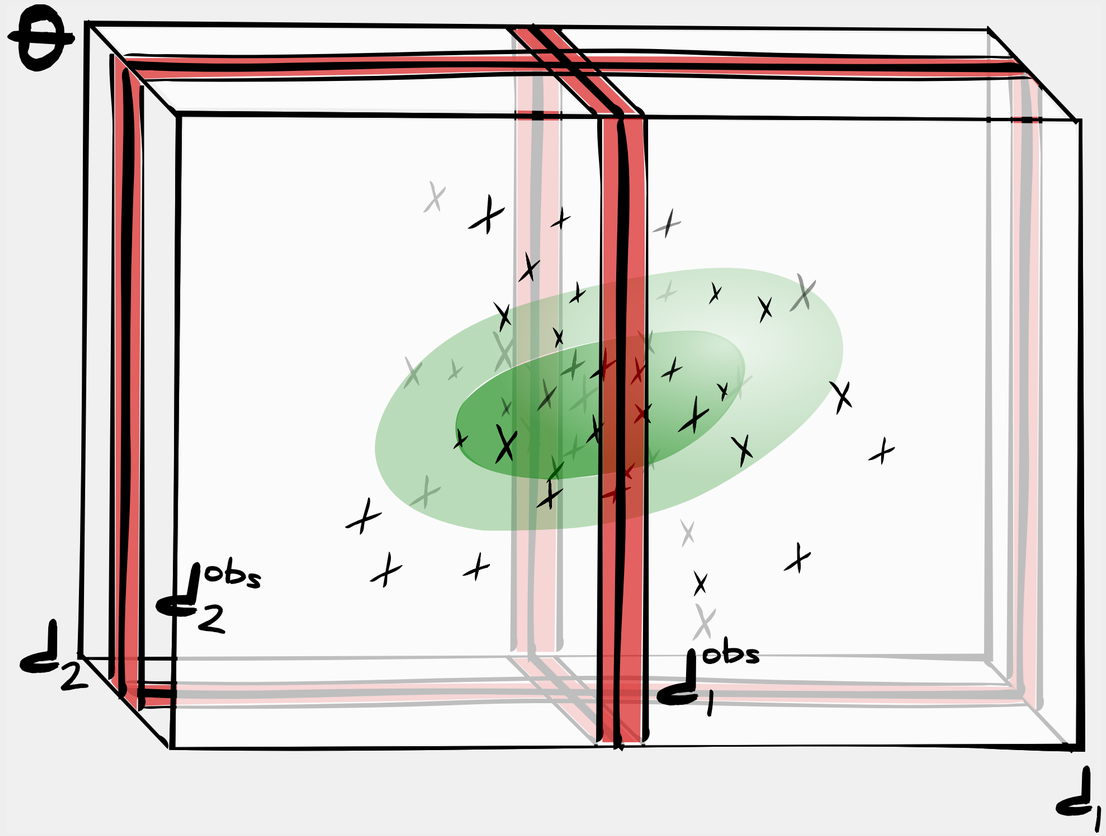


Likelihood-free inference using machine learning

Approximate Bayesian computation

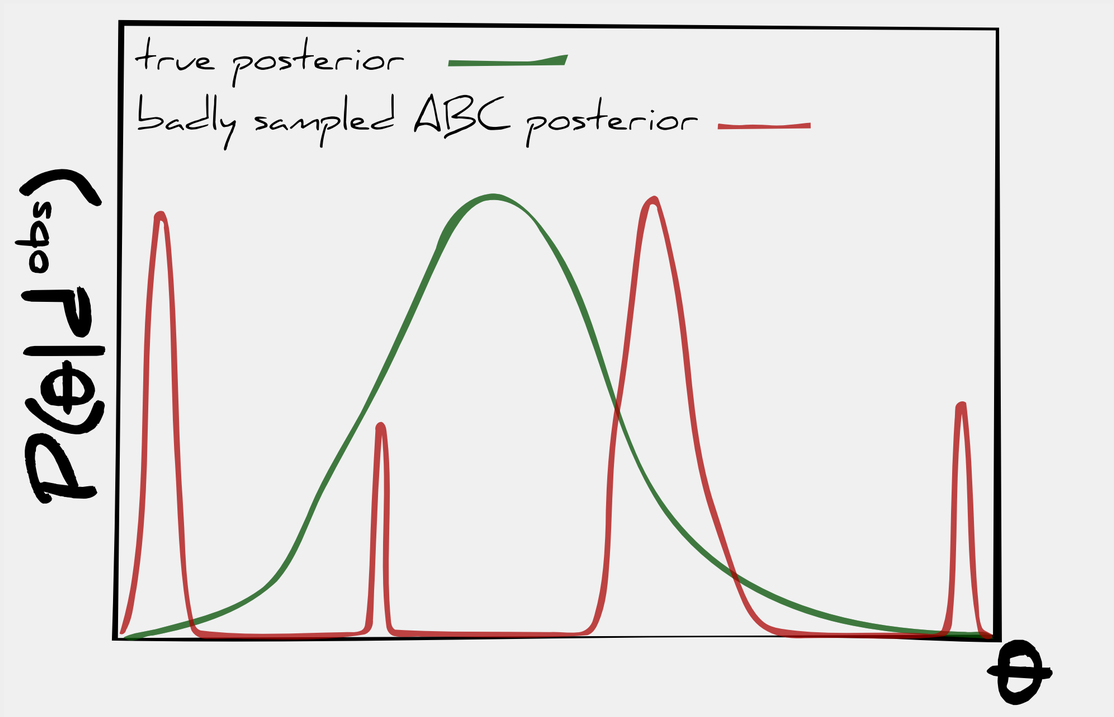


Why even introduce the neural network?



Inadequate sampling

Impossibly large numbers of simulations become necessary to correctly sample the posterior



We need to do some compression

Any pretrained neural network *could* be used

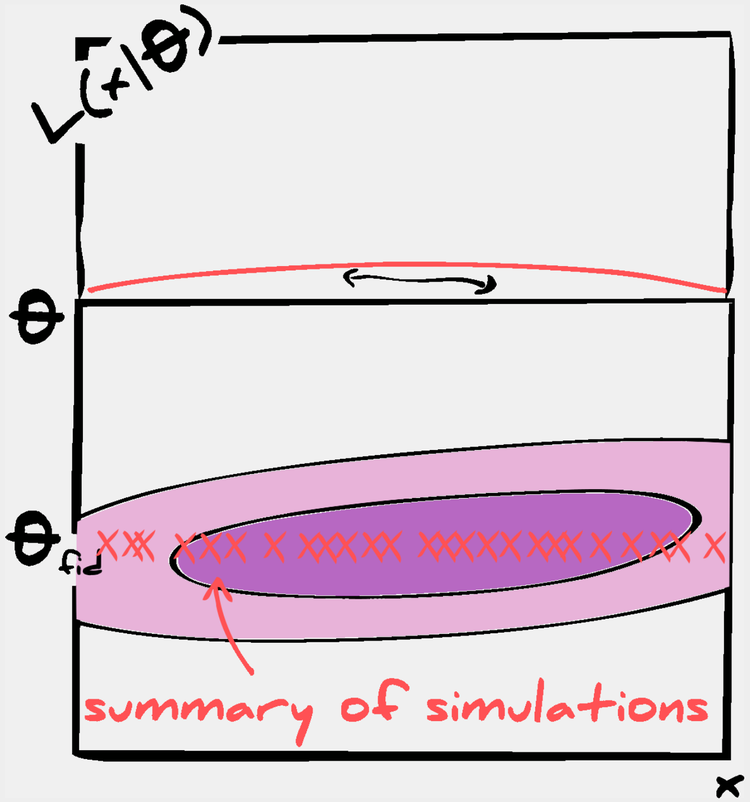
Any pretrained neural network *could* be used

Is there a well motivated choice?

Information maximising neural networks

Which function $\mathscr{l} : \mathbb{R}^{\mathbf{d}} \rightarrow \mathbb{R}^{\theta}$ maximises the Fisher information of the summaries $\mathbf{x}$ from that function?

$$\mathcal{L}(\mathbf{d}|\theta) \rightarrow \mathcal{L}(\mathbf{x}|\theta, \mathbf{d})$$

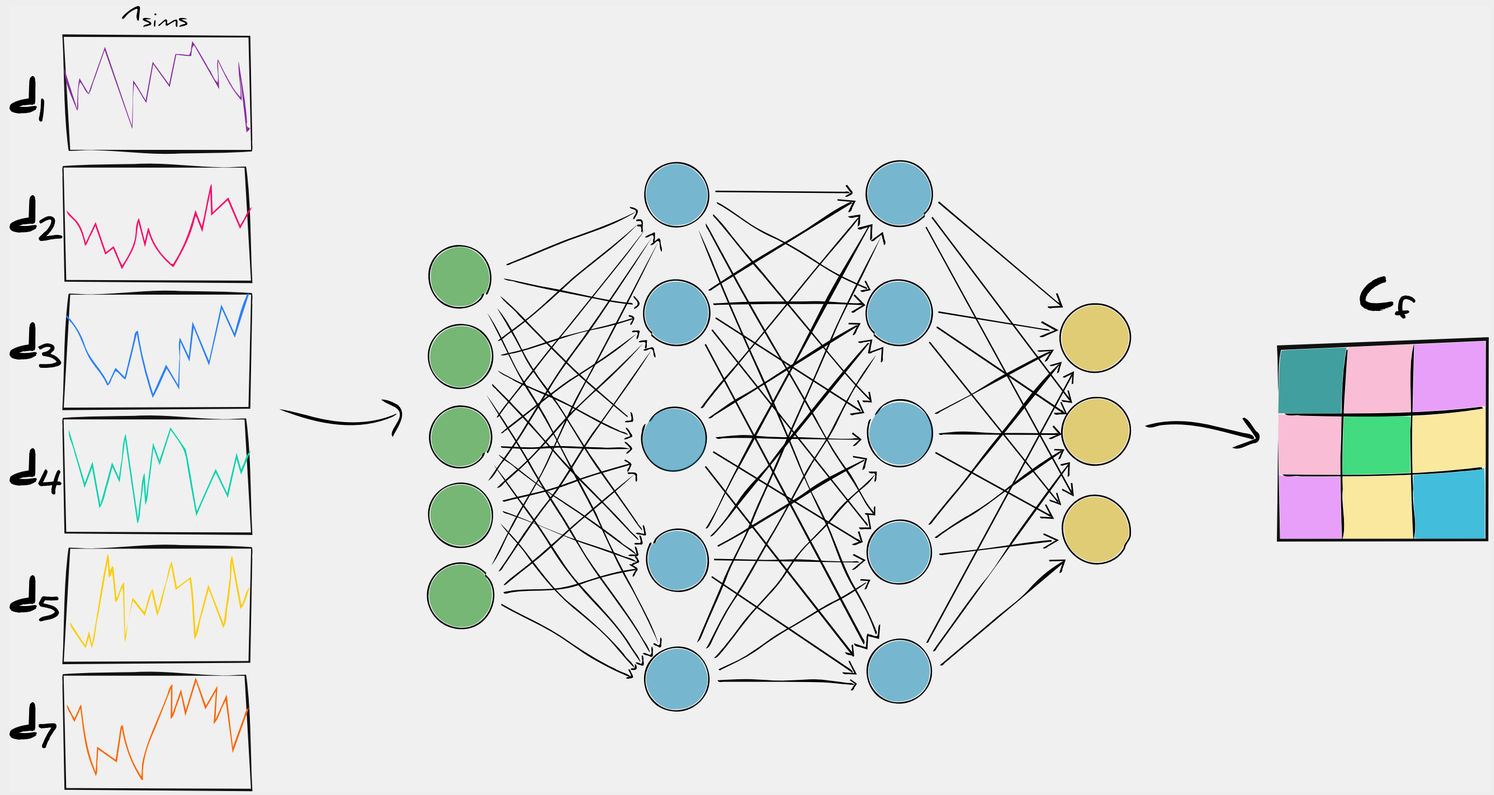


Fisher information of the network summaries

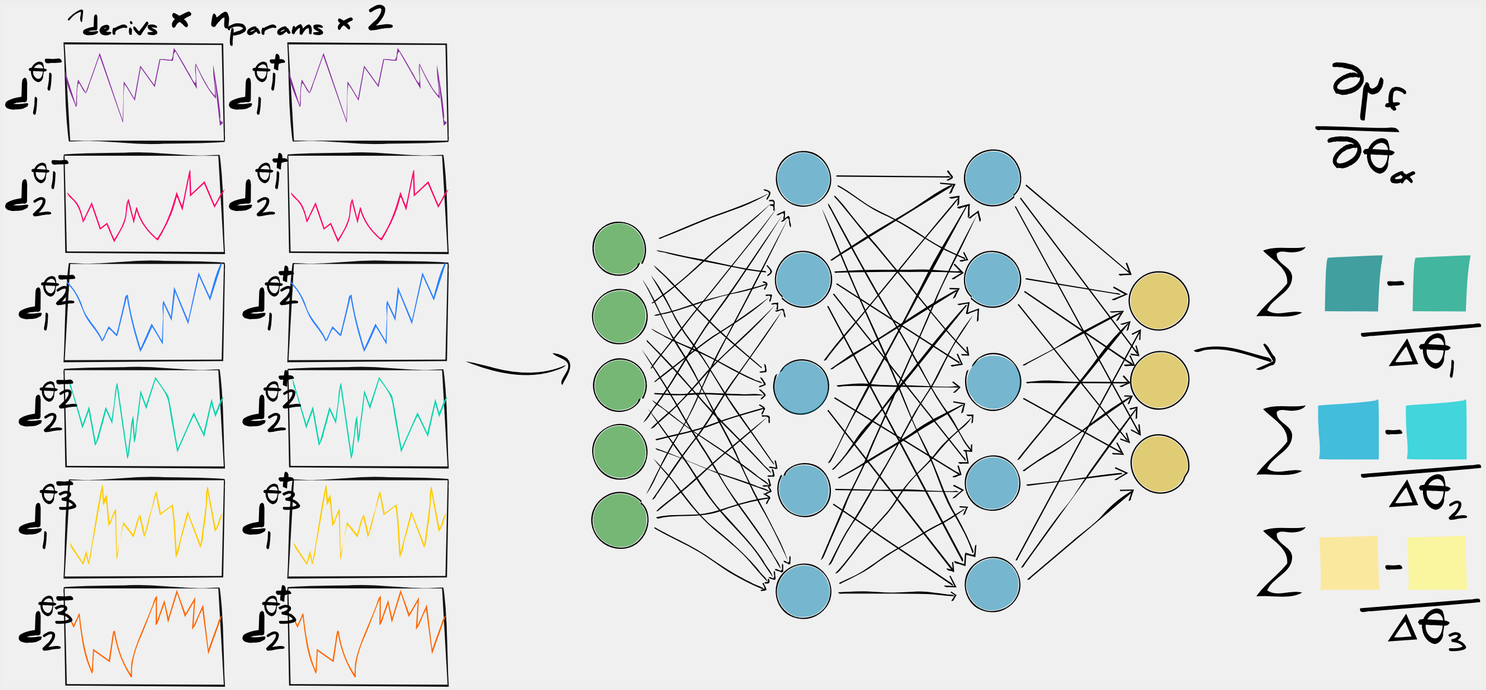
Simulations $\{\mathbf{d}_i | i \in [1, n_\mathbf{d}]\}$ at a single parameter value θ^*

Seed matched simulations $\{\mathbf{d}_i^\pm | i \in [1, n_\mathbf{p}]\}$ at perturbed fiducial paremeters $\Delta\theta^\pm = \theta \pm \delta$

Calculate the covariance



Calculate the derivative of the mean of the summaries with respect to the parameters



Calculate the Fisher information

$$\mathbf{F}_{\alpha\beta} = \frac{\partial \mu_\ell}{\partial \theta_\alpha}^T \mathbf{C}_\ell^{-1} \frac{\partial \mu_\ell}{\partial \theta_\beta}$$

This Gaussian form forces the summaries to be Gaussianised

Optimise the tunable parameters of the network such that $\ln |\mathbf{F}_{\alpha\beta}|$ is maximised!

Once converged the network compresses *and* Gaussianises the data *without* losing information*

*in the optimal case...

And we get free maximum likelihood estimates...

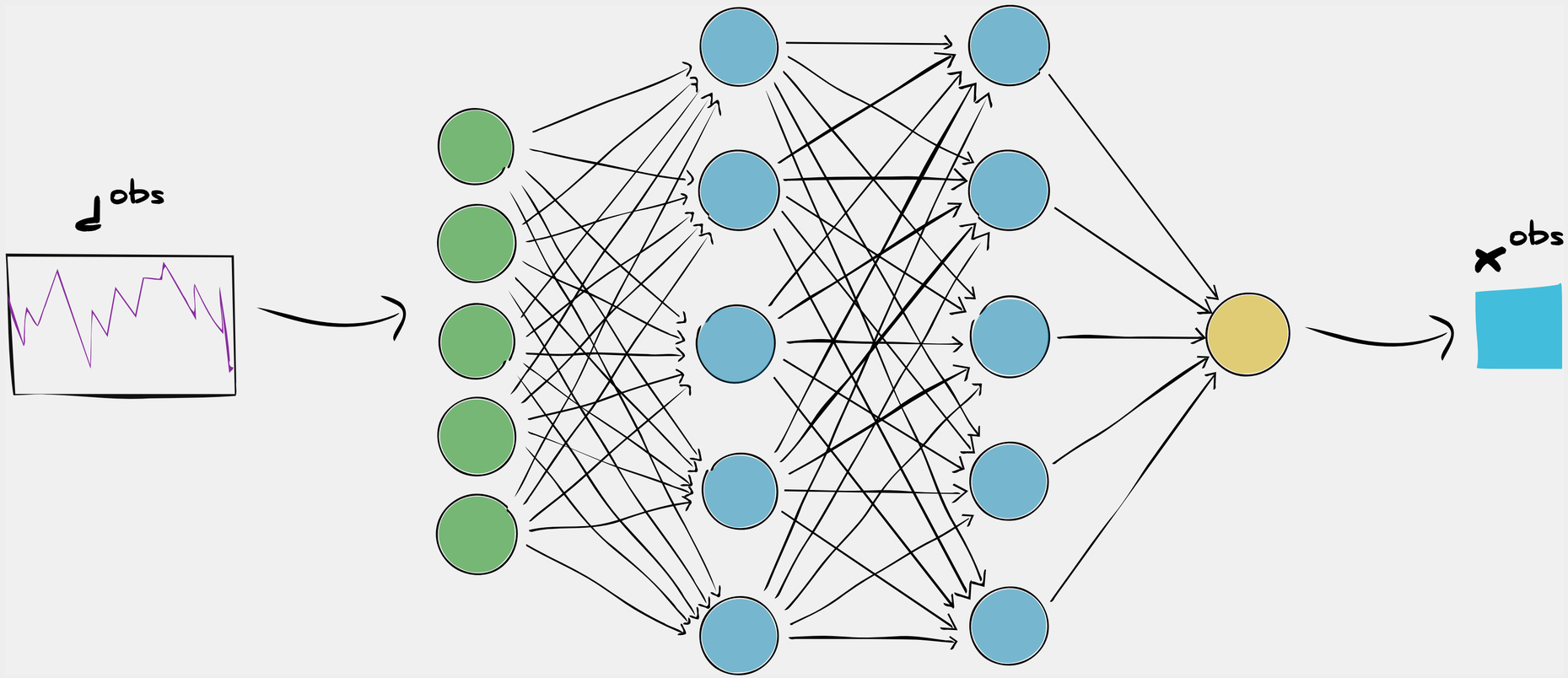
$$\theta_\alpha^{\mathsf{MLE}} = \theta_\alpha^* + \mathbf{F}_{\alpha\beta}^{-1} \mathbf{C}_\ell^{-1} \frac{\partial \mu_\ell}{\partial \theta_\beta} (\mathbf{x} - \mu_\ell)$$

And we can get free approximations of the posterior

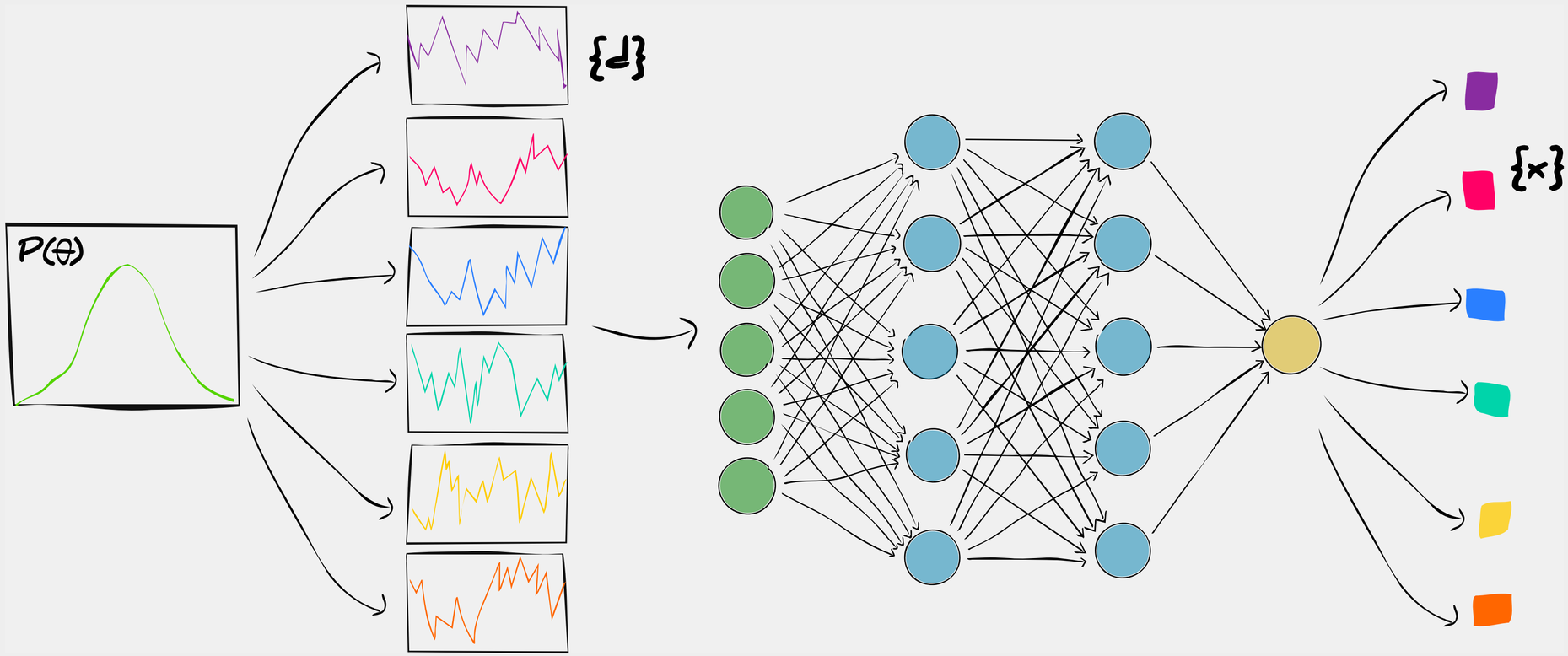
$$\mathrm{Cov}[\boldsymbol{\theta}_\alpha^{\mathrm{MLE}}, \boldsymbol{\theta}_\beta^{\mathrm{MLE}}] \geq \mathbf{F}^{-1}_{\alpha\beta}$$

Or do likelihood-free inference

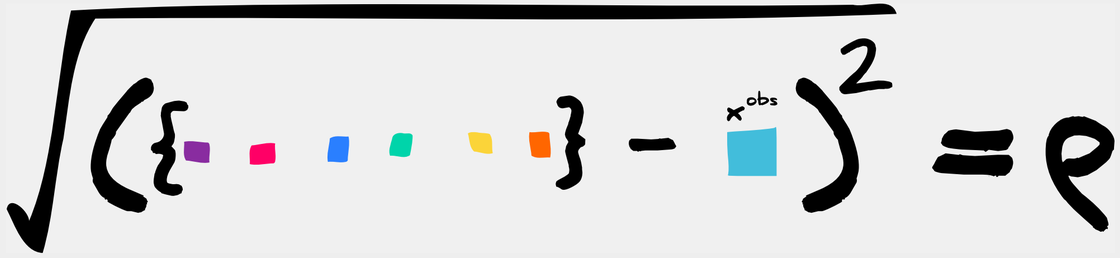
We pass our observed data through the network (put it to the side)



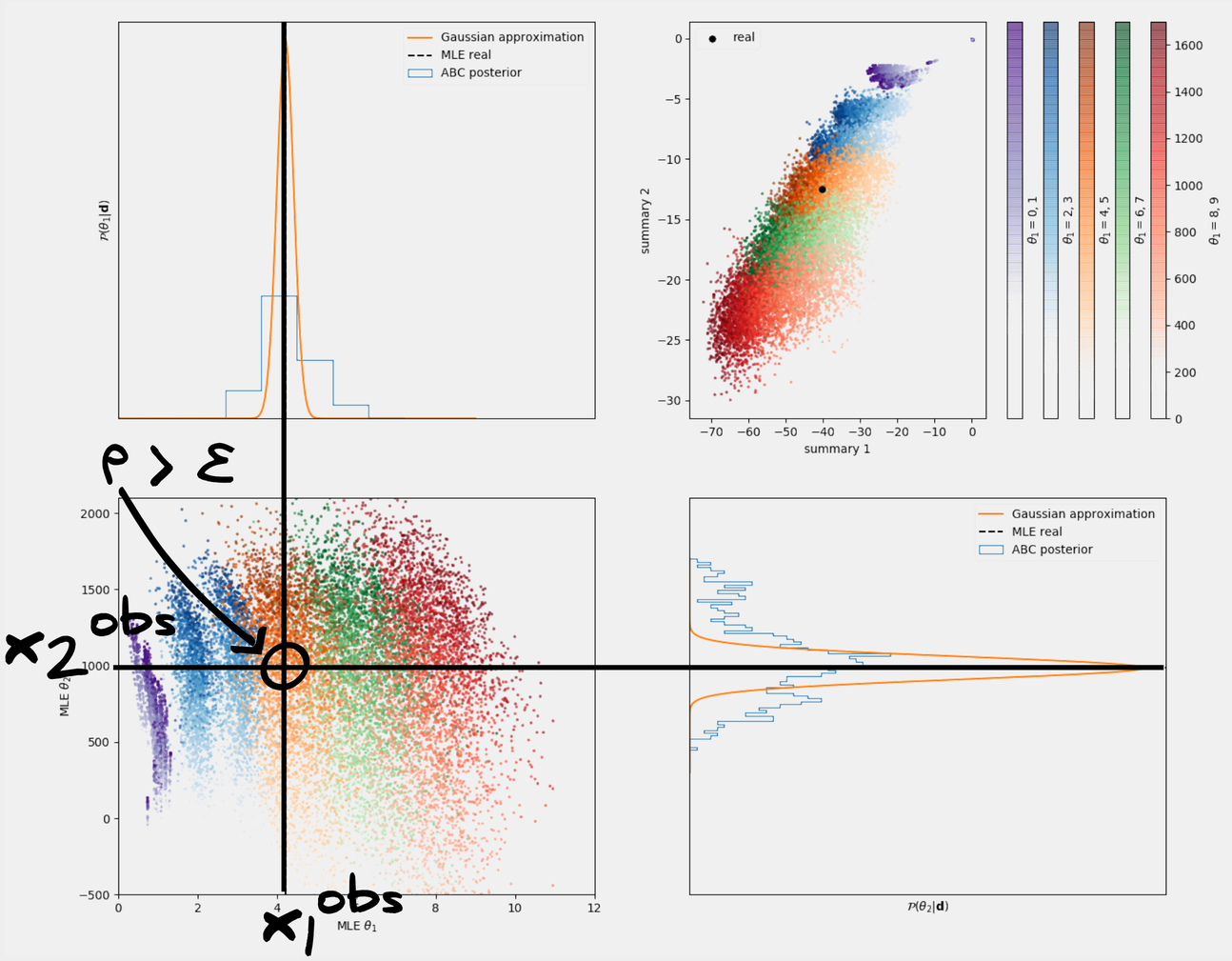
Draw simulations (cleverly?) from the prior model parameters, $\mathcal{P}(\theta)$ and pass them through the network



Measure the difference between the summarised simulations and the observed summary



Compare distance between observed summaries and simulation summaries and select results within ϵ



Conclusions

Conclusions

Neural networks are not to be trusted

They can make trusty companions - when the correct framework is introduced

Using statistics we can build them into the forward model to give us lossless summaries

Take home message

Take home message

Stop doing machine learning, think, then start doing machine learning again!