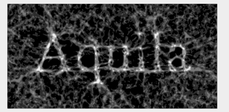
Parameter inference using neural networks

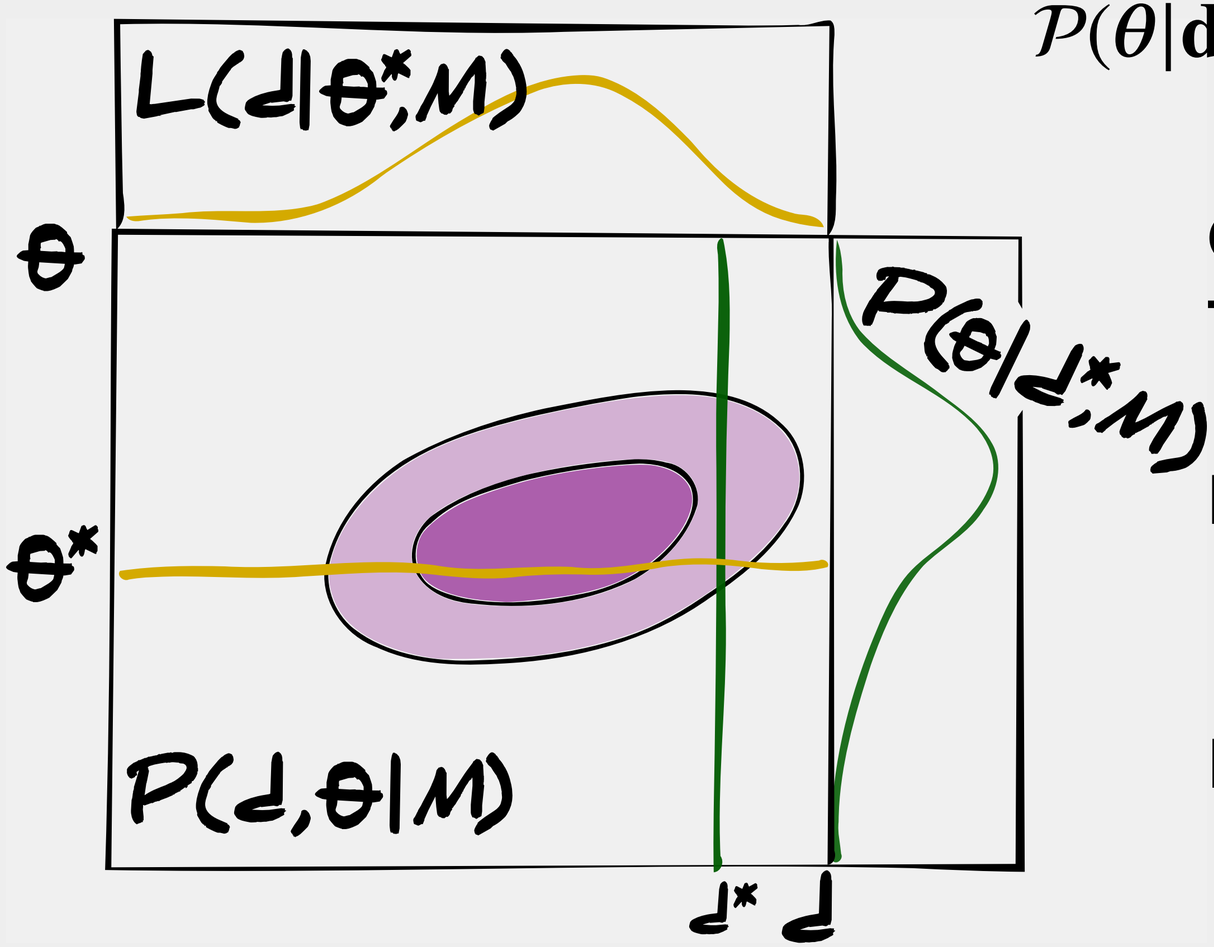
Tom Charnock

Institut d'Astrophysique de Paris



Joint distribution of parameters, θ, and possible data, $\mathbf{d}$,

which can be generated by a model, $\mathcal{M} : \theta \rightarrow \mathbf{d}$



$$\mathcal{P}(\theta|\mathbf{d}, \mathcal{M}) = \frac{\mathcal{L}(\mathbf{d}|\theta, \mathcal{M})p(\theta|\mathcal{M})}{p(\mathbf{d}|\mathcal{M})}$$

Obligatory Bayes Theorem

Likelihood

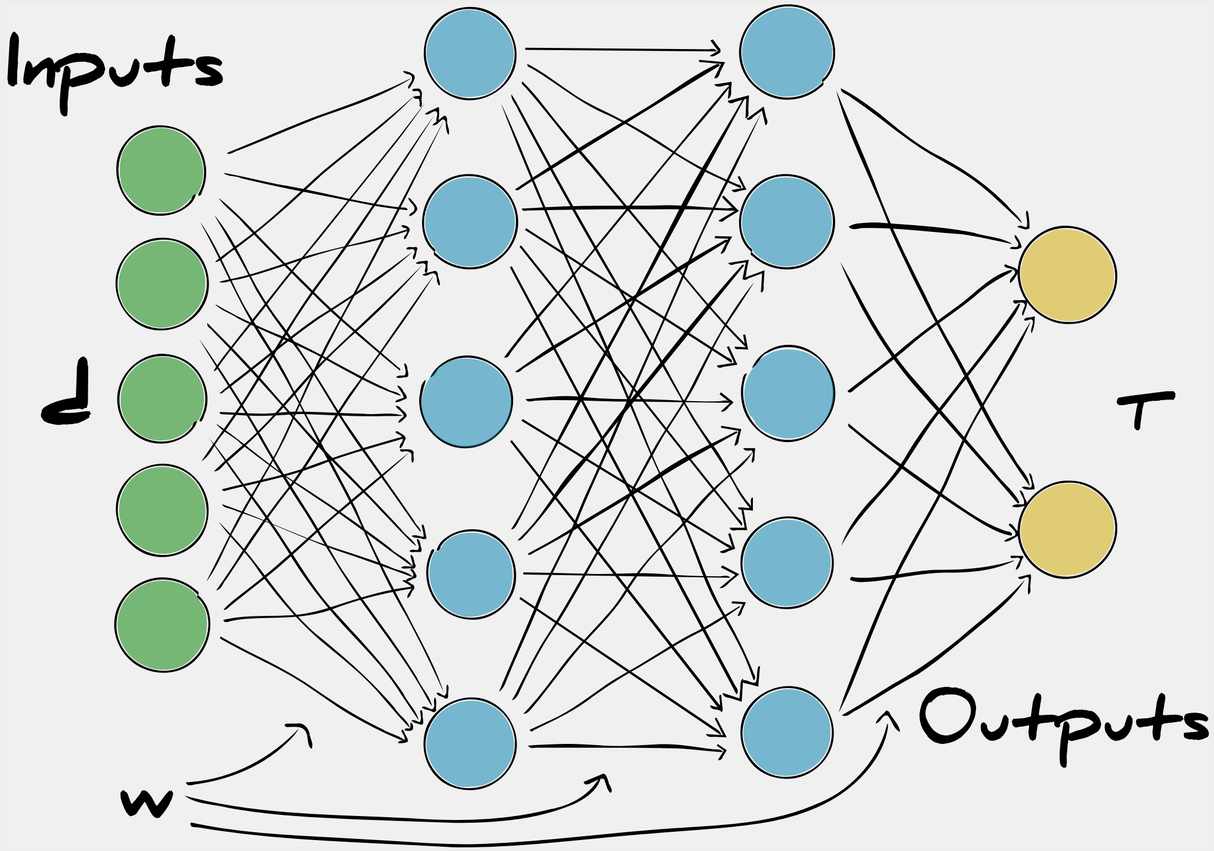
$$\mathcal{L}(\mathbf{d}|\theta^*, \mathcal{M})$$

Posterior

$$\mathcal{P}(\theta|\mathbf{d}^*, \mathcal{M})$$

Parameter inference using neural networks

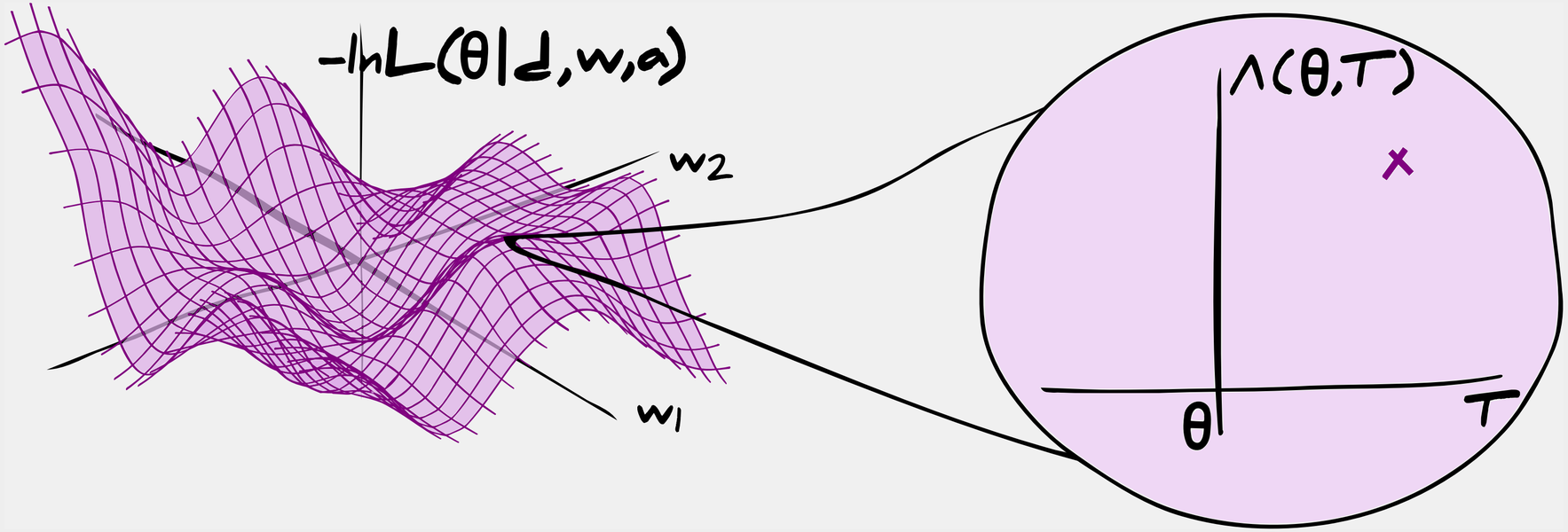
It's common to try and predict model parameters by regression



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

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

How likely are we to obtain any particular output from the network



$$\Lambda(\boldsymbol{\theta}, \boldsymbol{\tau}) \propto -\ln \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha})$$

The posterior becomes the posterior predictive density

$$\mathcal{P}(\theta|\mathbf{d}) = \int d\boldsymbol{\omega} d\boldsymbol{\alpha}\ \mathcal{L}(\theta|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha})\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha})$$

The posterior becomes the posterior predictive density

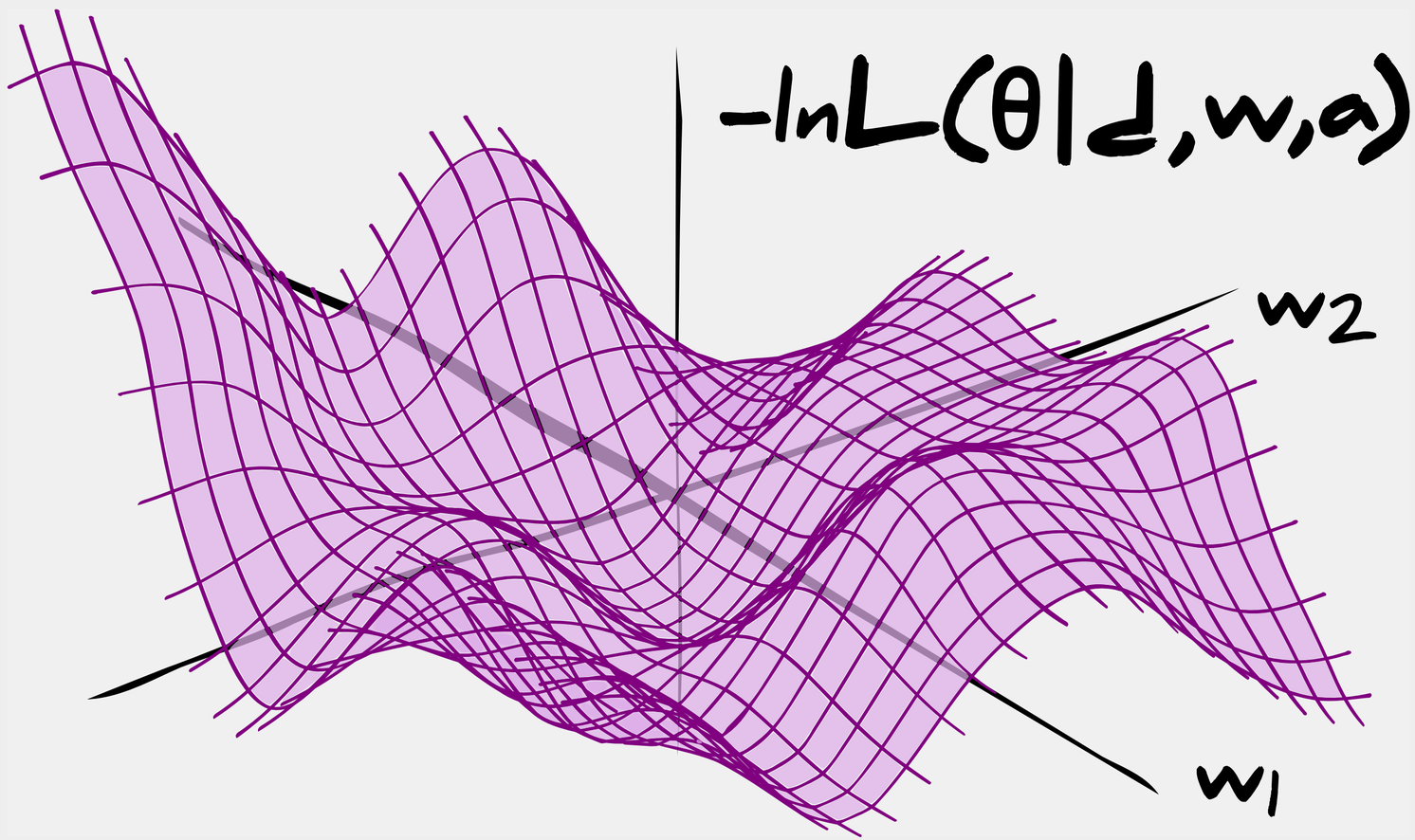
$$\mathcal{P}(\boldsymbol{\theta}|\mathbf{d}) = \int d\boldsymbol{\omega} d\boldsymbol{\alpha}\ \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}) \mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha})$$

$\mathcal{P}(\boldsymbol{\theta}|\mathbf{d})$ - Posterior predictive density
How likely are the true parameters given some data?

$\mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha})$ - Likelihood
How likely are the parameters to be generated by a particular network?

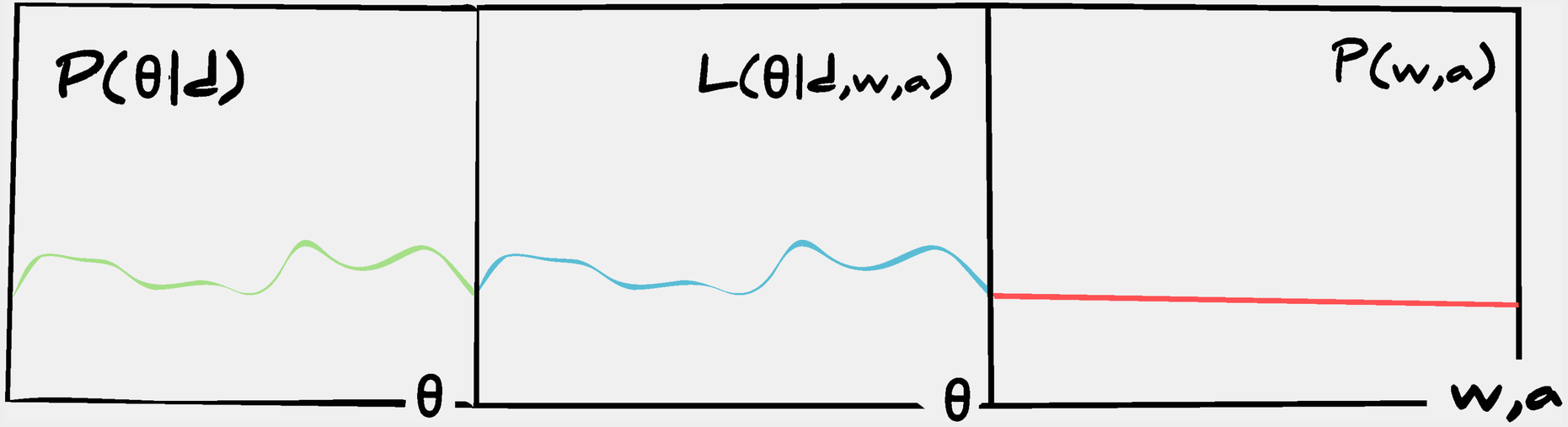
$\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha})$ - Probability density
What is the probability of obtaining a particular network with particular parameter values?

The chance of getting any parameter value from any network given some data is *almost* equal



So all the information comes from prior knowledge of the weights and hyperparameters

$$\mathcal{P}(\theta|\mathbf{d}) = \int d\boldsymbol{\omega} d\boldsymbol{\alpha}\ \mathcal{L}(\theta|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}) \mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha})$$



Where does the information about the weights and hyperparameters come from?

Training and validation data

Training and validation data

$$\{\mathbf{d}, \theta\}^{\text{train}} \equiv \{\mathbf{d}_i^{\text{train}}, \theta_i^{\text{train}} | i \in [1, n_{\text{train}}]\}$$

Training and validation data

$$\{\mathbf{d}, \theta\}^{\text{train}} \equiv \{\mathbf{d}_i^{\text{train}}, \theta_i^{\text{train}} | i \in [1, n_{\text{train}}]\}$$

$$\{\mathbf{d}, \theta\}^{\text{val}} \equiv \{\mathbf{d}_i^{\text{val}}, \theta_i^{\text{val}} | i \in [1, n_{\text{val}}]\}$$

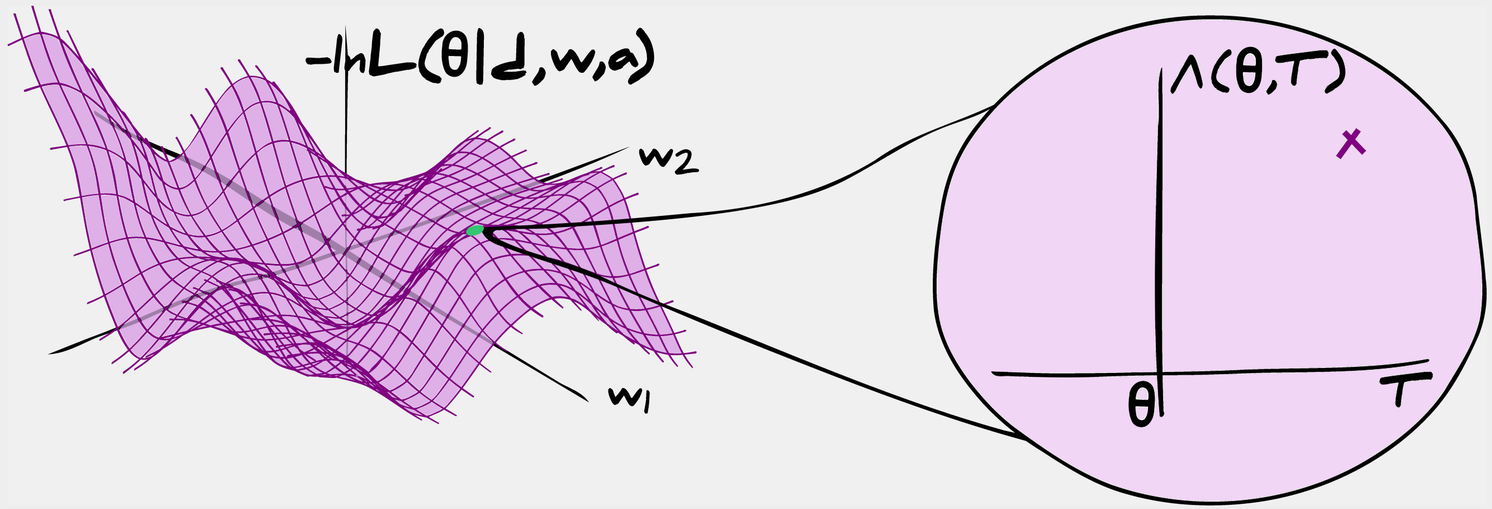
The prior distribution of weights and hyperparameters becomes the posterior

$$\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha} | \{\mathbf{d}, \theta\}^{\text{train}}, \{\mathbf{d}, \theta\}^{\text{val}}) \propto$$
$$\mathcal{L}(\boldsymbol{\omega}, \boldsymbol{\alpha} | \{\mathbf{d}, \theta\}^{\text{train}}, \{\mathbf{d}, \theta\}^{\text{val}}) p(\boldsymbol{\omega}, \boldsymbol{\alpha})$$

Training a network

Training a network

What are the maximum likelihood estimates of the weights?



$$\boldsymbol{\omega}^{\mathrm{MLE}} = \underset{\boldsymbol{\omega}}{\mathrm{argmax}} \left[\mathcal{L}(\{\boldsymbol{\theta}\}^{\mathrm{train}} | \{\mathbf{d}\}^{\mathrm{train}}, \boldsymbol{\omega}, \boldsymbol{\alpha}^{*}) \right]$$

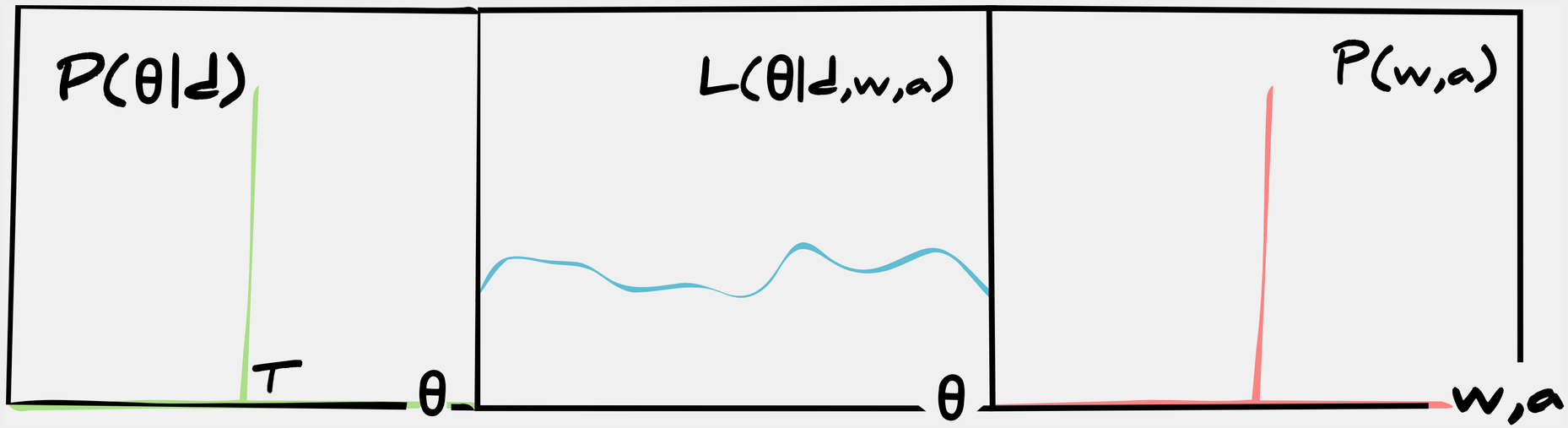
We degenerate the posterior

$$\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha} | \{\mathbf{d}, \theta\}^{\text{train}}) \propto \mathcal{L}(\boldsymbol{\omega}, \boldsymbol{\alpha} | \{\mathbf{d}, \theta\}^{\text{train}}) p(\boldsymbol{\omega}, \boldsymbol{\alpha})$$

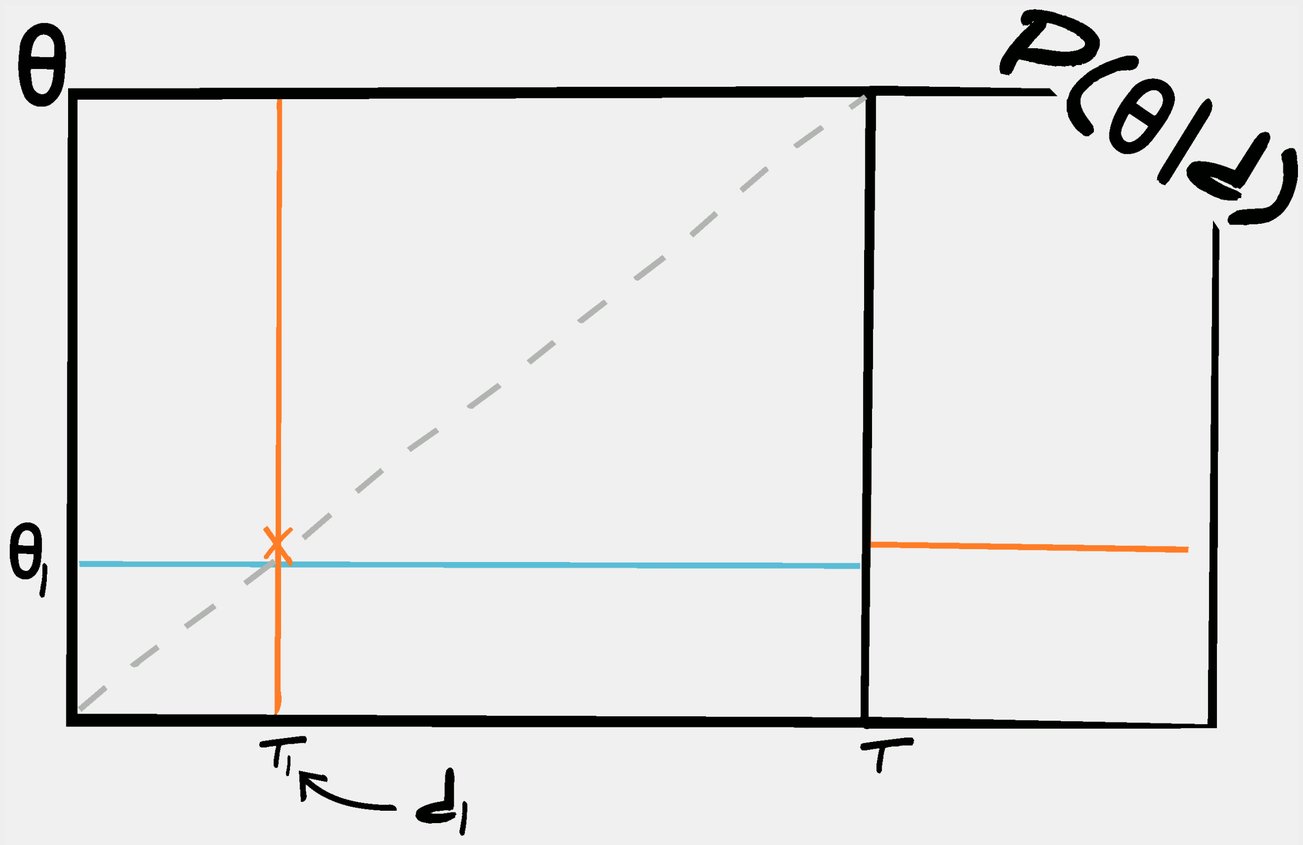
$$\rightarrow \delta(\boldsymbol{\omega} - \boldsymbol{\omega}^{\text{MLE}}, \boldsymbol{\alpha} - \boldsymbol{\alpha}^{*})$$

$$\mathcal{P}(\theta | \mathbf{d}) \propto \int d\boldsymbol{\omega} d\boldsymbol{\alpha} \mathcal{L}(\theta | \mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}) \delta(\boldsymbol{\omega} - \boldsymbol{\omega}^{\text{MLE}}, \boldsymbol{\alpha} - \boldsymbol{\alpha}^{*})$$

$$= \delta(\boldsymbol{\tau})$$

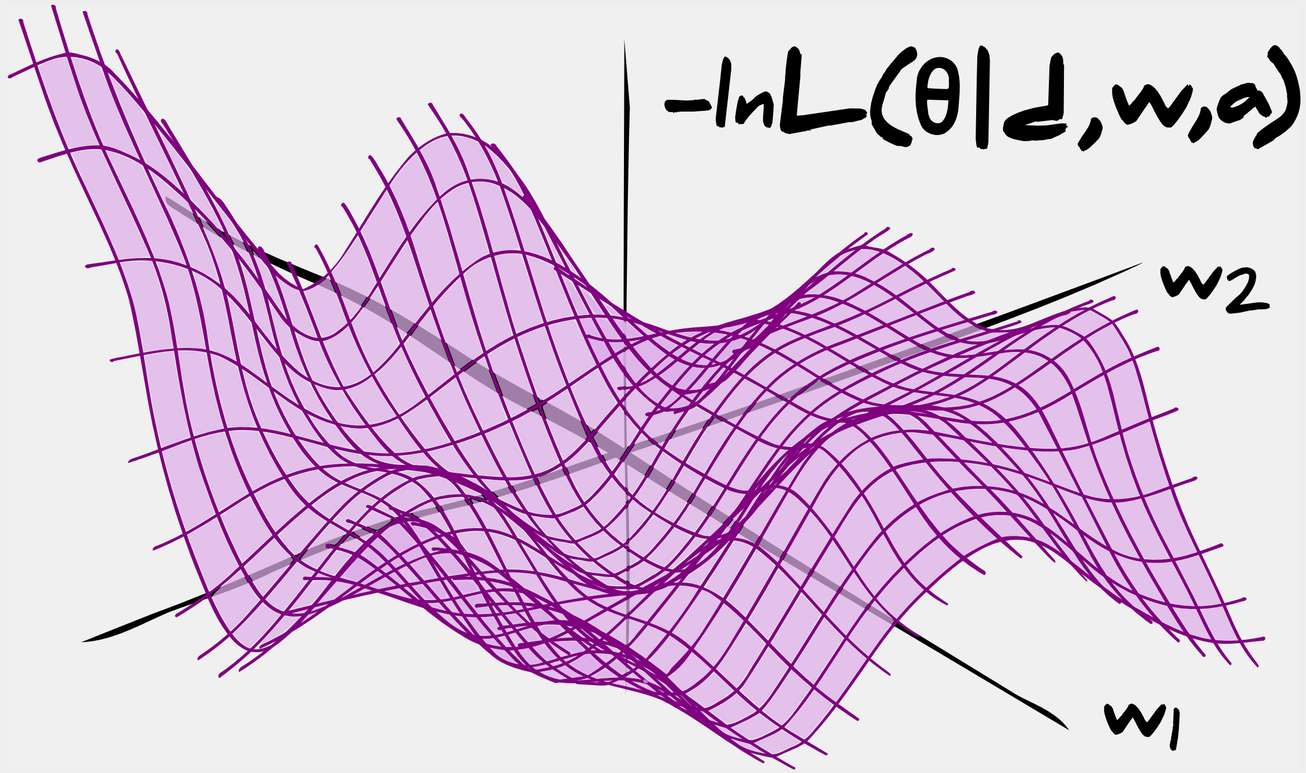


All predictions are (unknowably incorrect) estimates

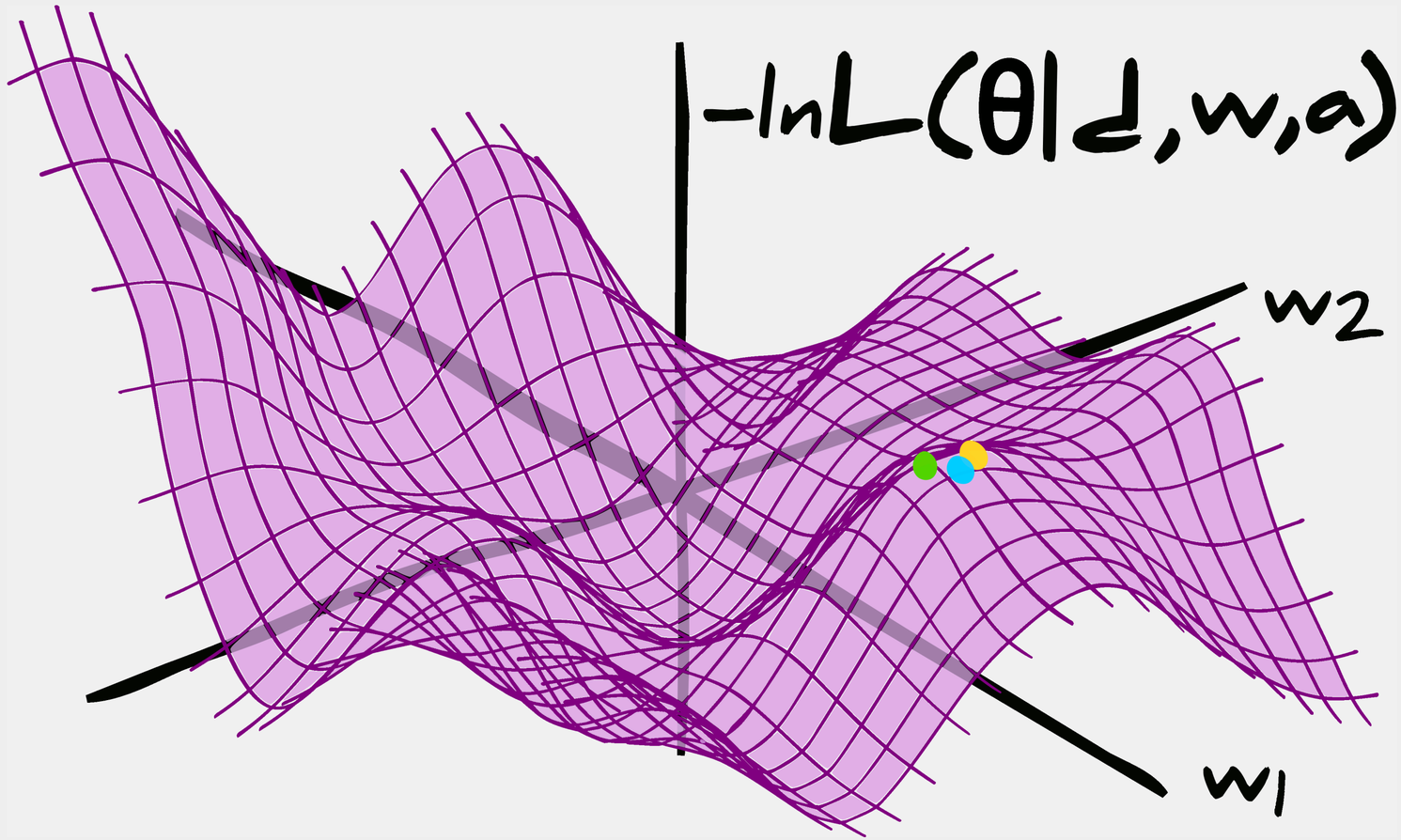


$$\mathcal{P}(\theta|\mathbf{d}) = \delta(\tau)$$

There is no way to interpret how close τ is to θ...

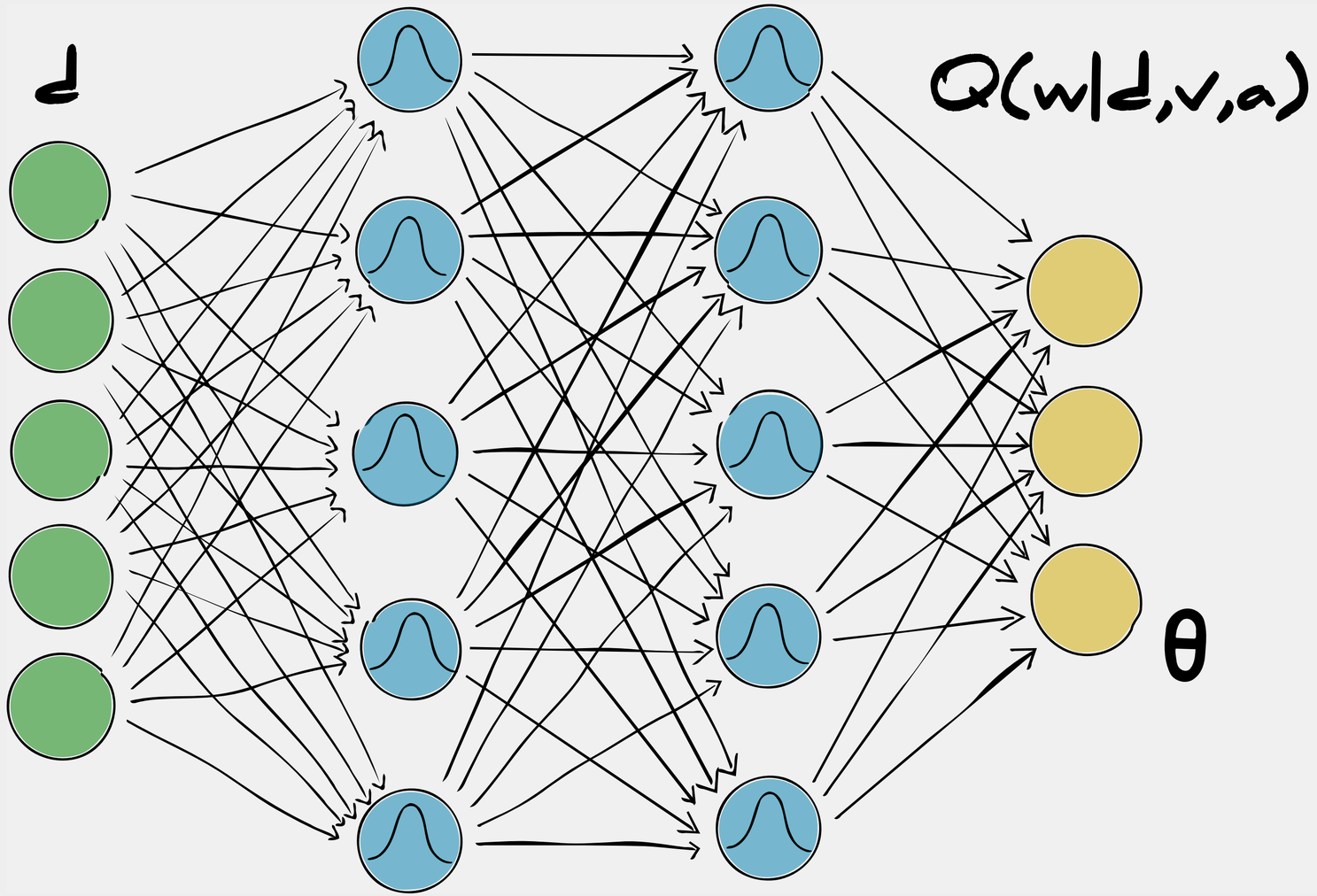


Local maximum likelihood estimates

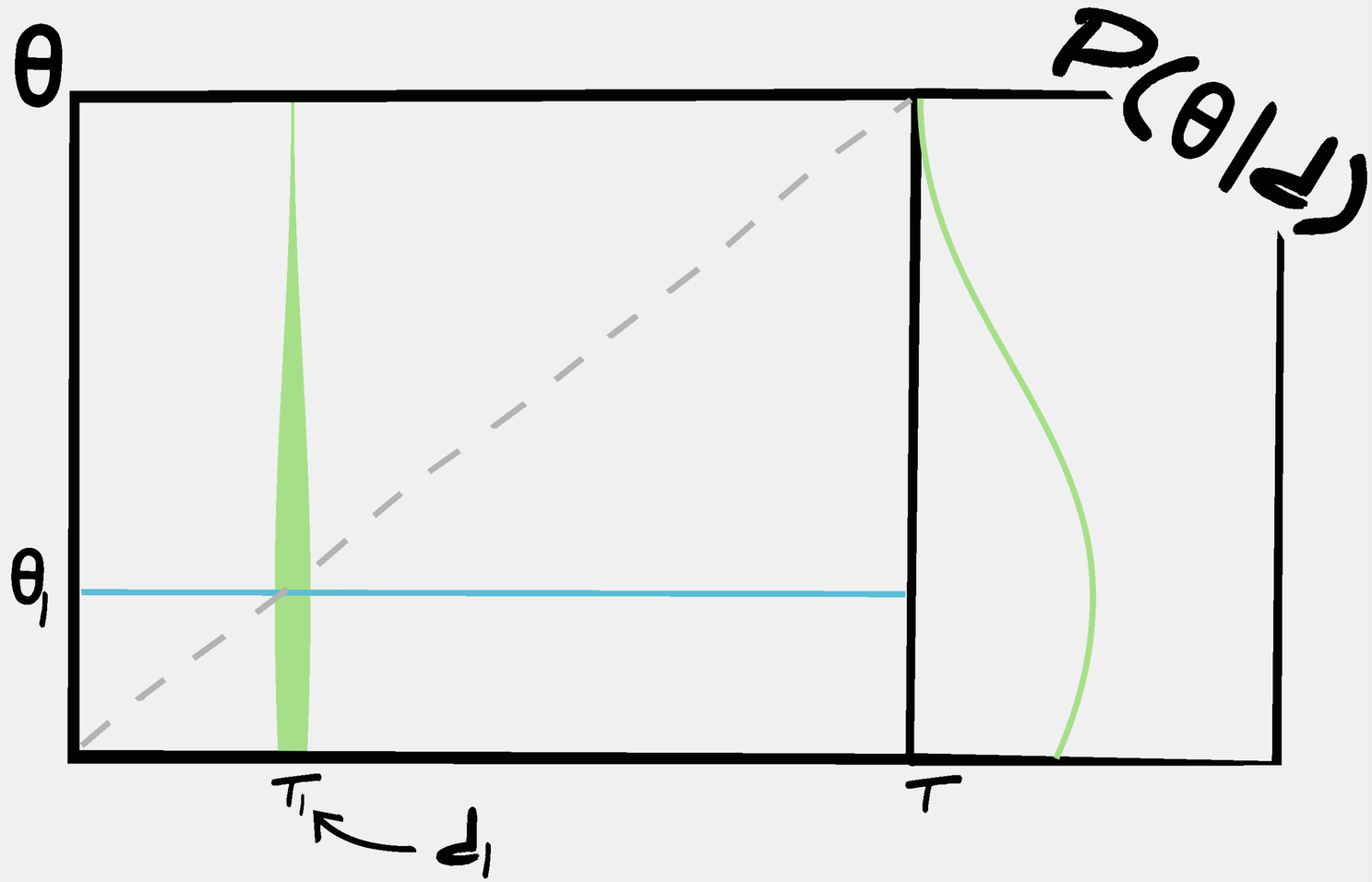


Are there better methods?

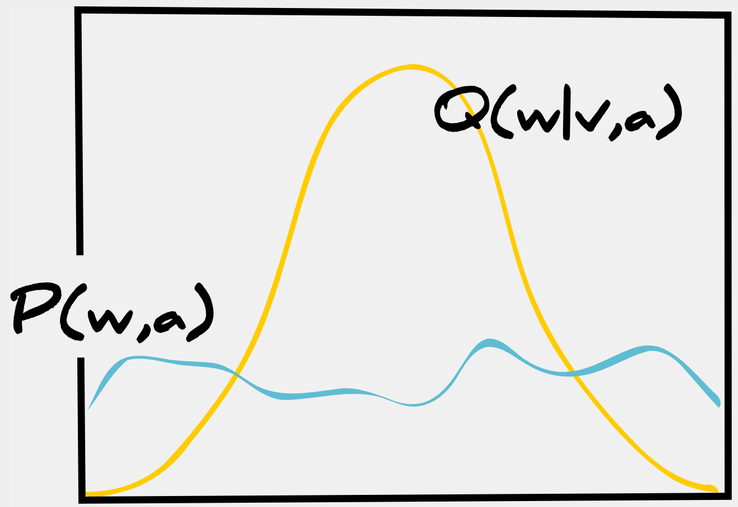
Variational inference



$$\mathcal{P}(\boldsymbol{\theta}|\mathbf{d}) = \int d\boldsymbol{\omega} d\boldsymbol{\nu} d\boldsymbol{\alpha}\ \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}) \mathcal{Q}(\boldsymbol{\omega}|\boldsymbol{\nu}, \boldsymbol{\alpha}, \{\mathbf{d}, \boldsymbol{\theta}\}^{\text{train}}) p(\boldsymbol{\nu}, \boldsymbol{\alpha})$$



Still depends on fixed weights in the complex likelihood surface and choice of variational distribution



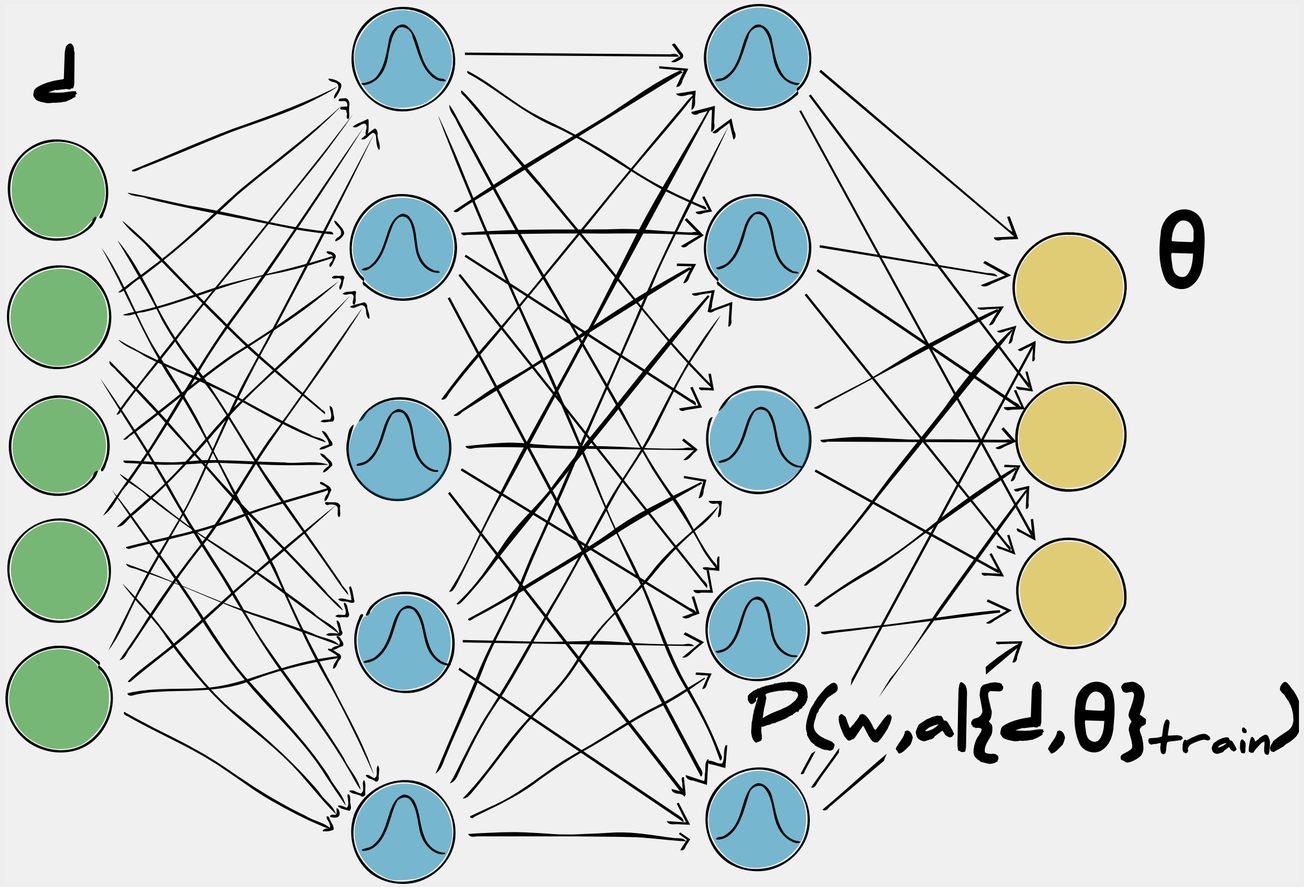
$$\mathcal{P}(\boldsymbol{\theta}|\mathbf{d}) = \int d\boldsymbol{\omega} d\boldsymbol{\nu} d\boldsymbol{\alpha}\, \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}) \mathcal{Q}(\boldsymbol{\omega}|\boldsymbol{\nu}, \boldsymbol{\alpha}, \{\mathbf{d}, \boldsymbol{\theta}\}^{\text{train}})$$

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

$$= \int d\boldsymbol{\omega}\, \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}^*) \mathcal{Q}(\boldsymbol{\omega}|\boldsymbol{\nu}^{\text{MLE}}, \boldsymbol{\alpha}^*, \{\mathbf{d}, \boldsymbol{\theta}\}^{\text{train}}).$$

Bayesian neural networks

Bayesian neural networks



Approximate the posterior distribution of weights and hypermeters

$$\mathcal{P}(\boldsymbol{\theta}|\mathbf{d}) = \int d\boldsymbol{\omega} d\boldsymbol{\alpha}\ \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha}) \mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha}|\{\mathbf{d}, \boldsymbol{\theta}\}^{\text{train}})$$

$$\propto \int d\boldsymbol{\omega} d\boldsymbol{\alpha}\ \mathcal{L}(\boldsymbol{\theta}|\mathbf{d}, \boldsymbol{\omega}, \boldsymbol{\alpha})$$

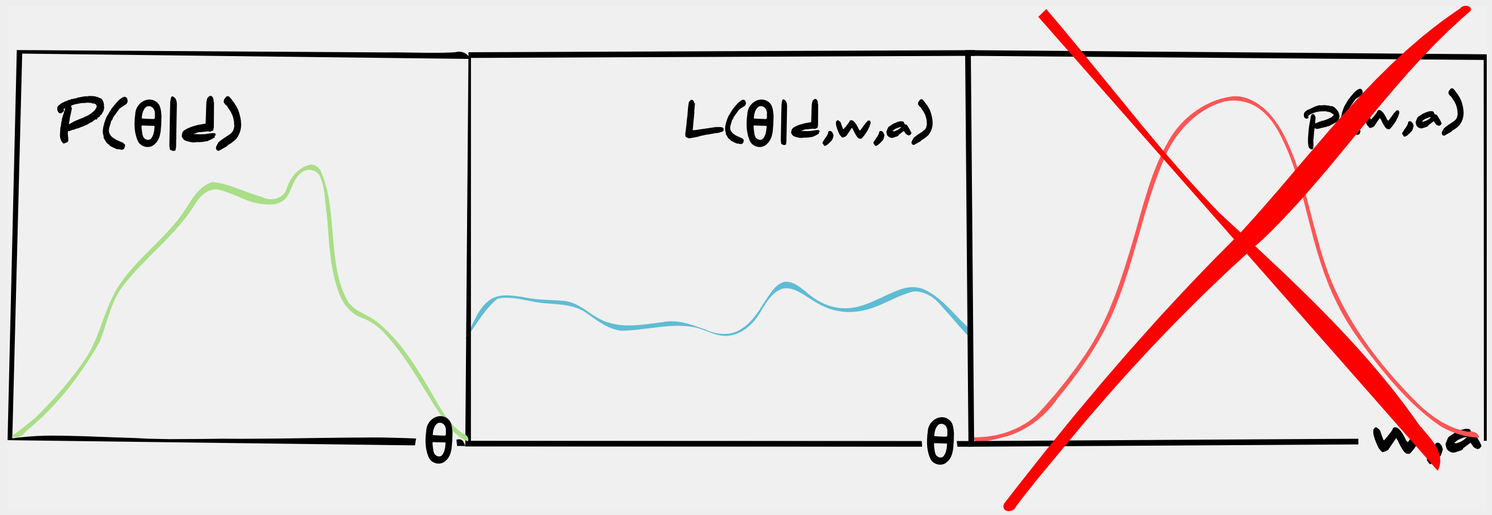
$$\times \prod_{i}^{n_{\text{train}}} \mathcal{L}(\boldsymbol{\theta}_i^{\text{train}}|\mathbf{d}_i^{\text{train}}, \boldsymbol{\omega}, \boldsymbol{\alpha}) p(\boldsymbol{\omega}, \boldsymbol{\alpha}).$$

Everything is dependent on the training data!

Classical network : $\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha}|\{\mathbf{d}, \theta\}^{\text{train}}) \rightarrow \delta(\boldsymbol{\omega} - \boldsymbol{\omega}^{\text{MLE}}, \boldsymbol{\alpha} - \boldsymbol{\alpha}^{*})$

Variational inference : $\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha}|\{\mathbf{d}, \theta\}^{\text{train}}) = \mathcal{Q}(\boldsymbol{\omega}|\boldsymbol{\nu}^{\text{MLE}}, \boldsymbol{\alpha}^{*}, \{\mathbf{d}, \theta\}^{\text{train}})$

Bayesian networks : $\mathcal{P}(\boldsymbol{\omega}, \boldsymbol{\alpha}|\{\mathbf{d}, \theta\}^{\text{train}}) = \prod_{i}^{n_{\text{train}}} \mathcal{L}(\mathbf{t}_i^{\text{train}}|\theta_i^{\text{train}}, \boldsymbol{\omega}, \boldsymbol{\alpha}) p(\boldsymbol{\omega}, \boldsymbol{\alpha})$



If the training data, network optimisation, or (pretty much) anything else is not perfect, the inference will be biased

Parameter inference with summaries

Consider when the likelihood of some data, $\mathcal{L}(\mathbf{d}|\theta)$ is unknown

Consider when the likelihood of some data, $\mathcal{L}(\mathbf{d}|\theta)$ is unknown

Summarise the data, $\mathit{\ell} : \mathbf{d} \rightarrow \mathbf{t}$, to have a nicer likelihood for the model $\mathcal{M} \otimes \mathit{\ell} : \theta \rightarrow \mathbf{t}$

$$\mathcal{P}(\theta|\mathbf{t}, \mathcal{M} \otimes \mathit{\ell}) = \frac{\mathcal{L}(\mathbf{t}|\theta, \mathcal{M} \otimes \mathit{\ell})p(\theta|\mathcal{M})}{p(\mathbf{t}|\mathcal{M} \otimes \mathit{\ell})}$$

The data no longer enters the inference, only the summaries

We can use a neural network as this summarising function

We can use a neural network as this summarising function

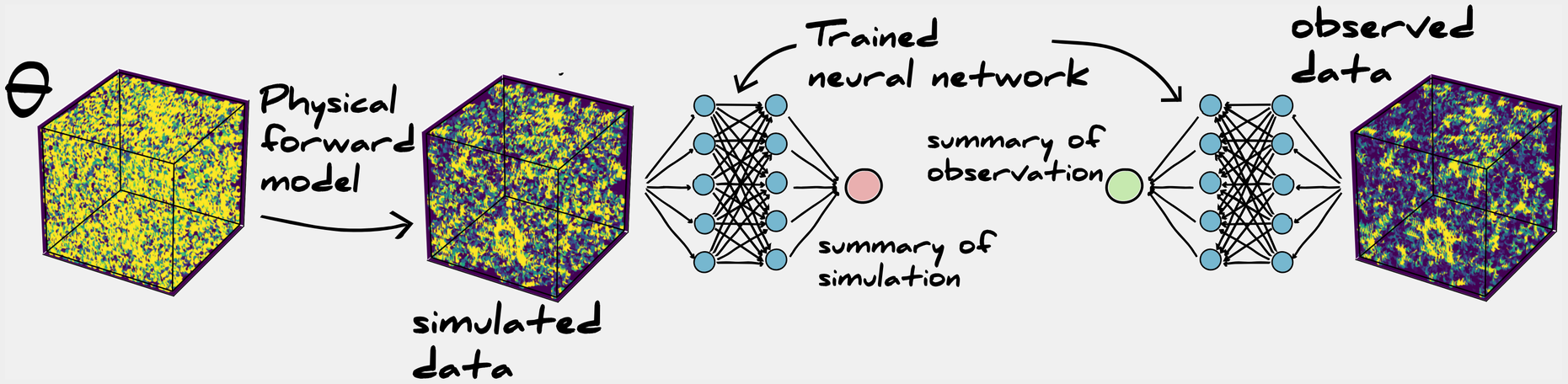
But the likelihood of the summaries is still difficult in this situtation

Forward modelling

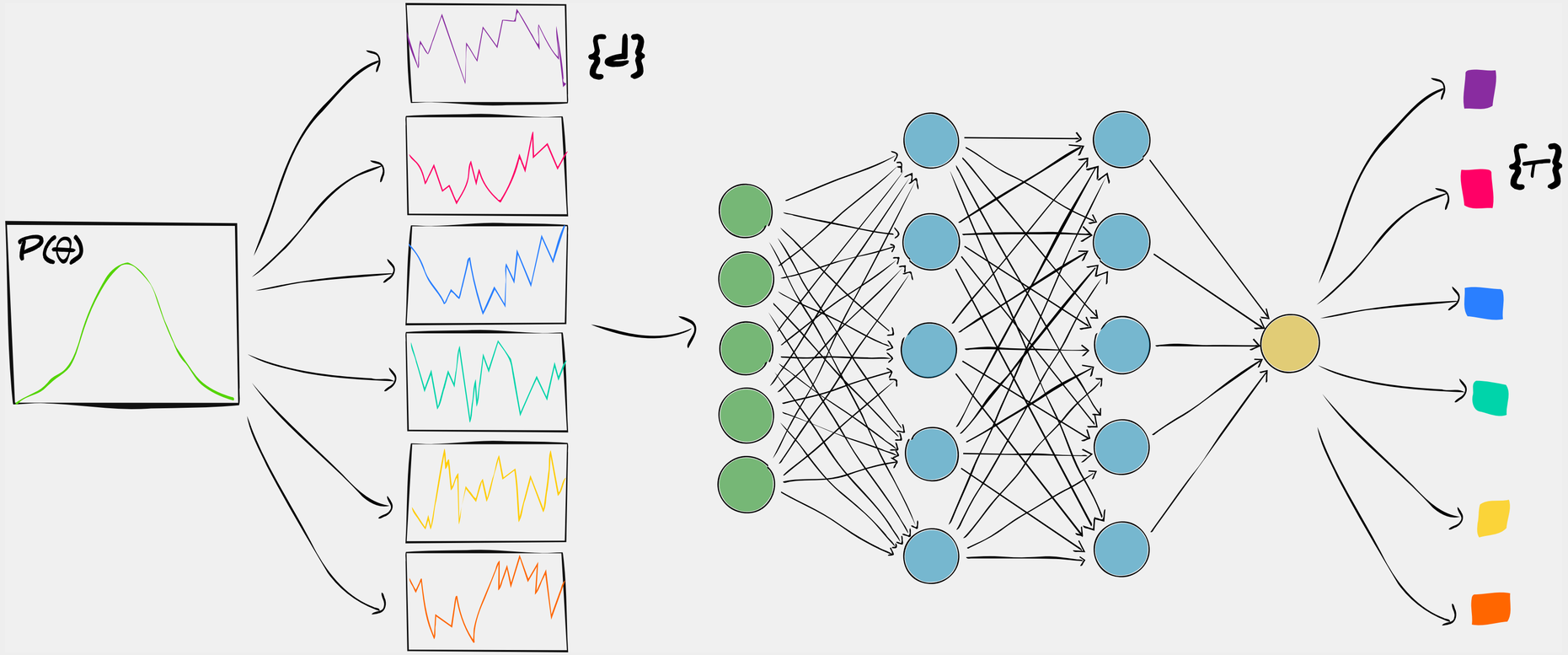
Method 1 :

Likelihood-free inference

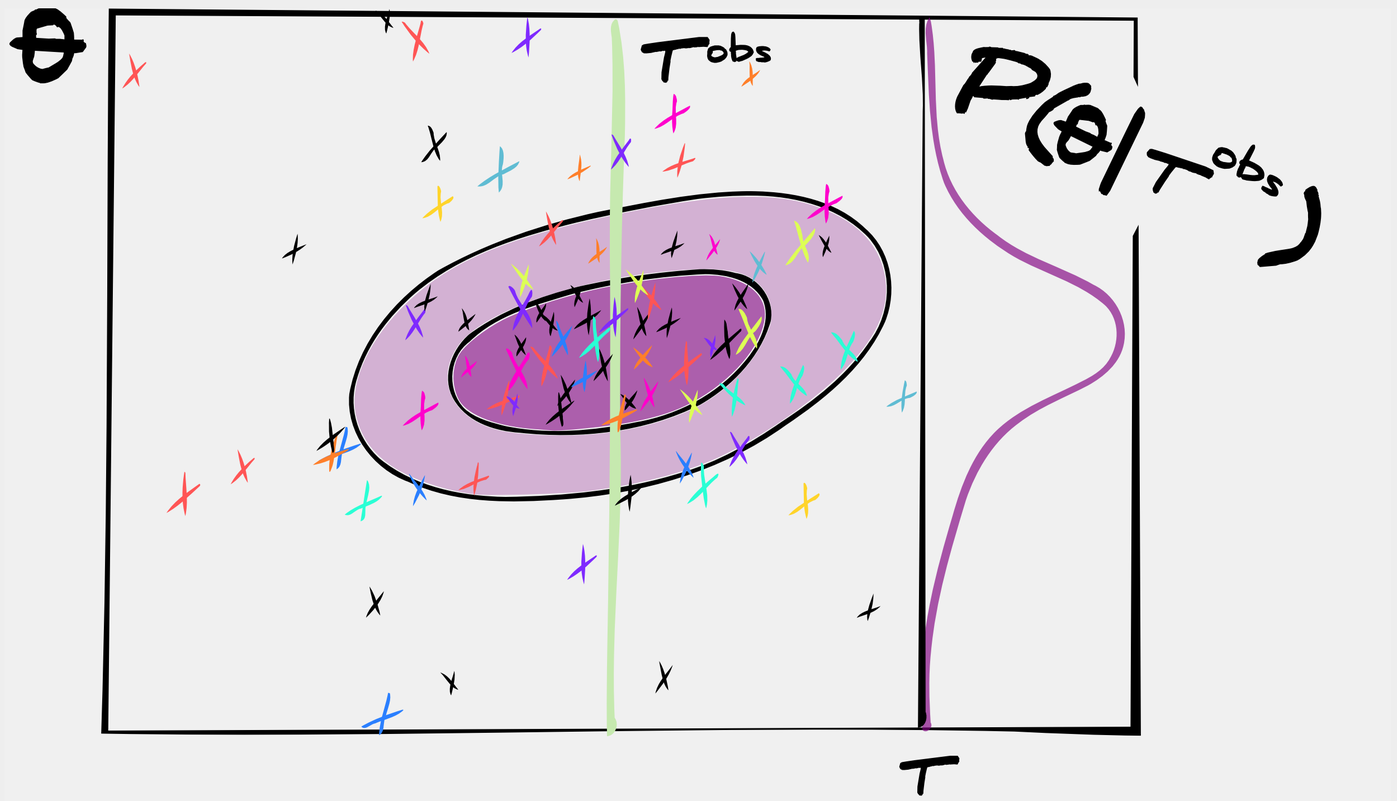
Use the same neural network to summarise observations and simulations



Generate simulations drawn from prior and summarise

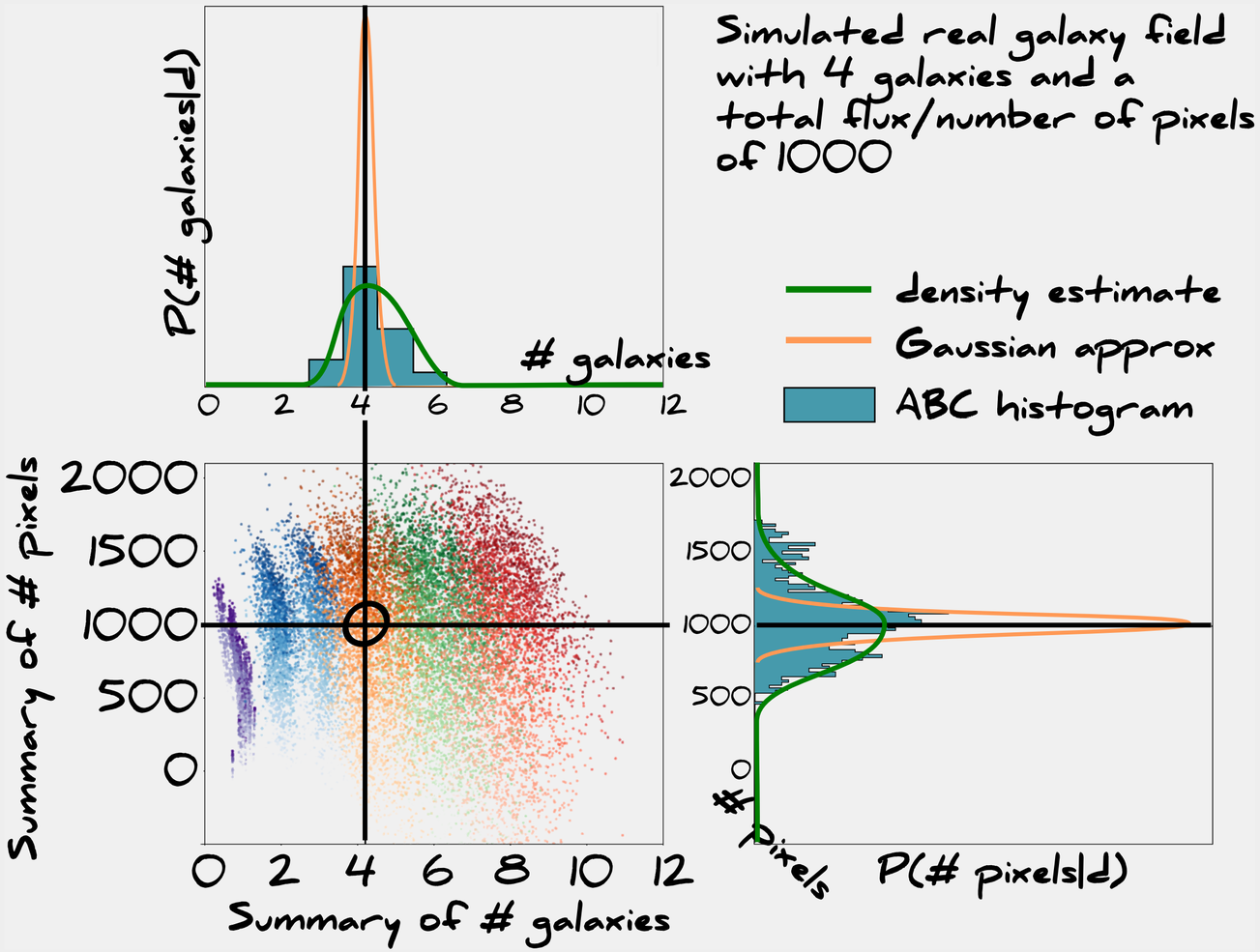


Make a kernel density estimate (or use ABC) on the simulations



A slice through at the summarised observation is the approximate posterior

Posterior distribution of galaxy counts and fluxes in fields



We're still biased by the form of the density estimator we use, and by the number of simulations we use for approximating the posterior

Method 2 :

Infer the data, physics and neural network

Need a good control on the likelihood of weights and hyperparameters

- remove degeneracies

- force to be convex

- etc.

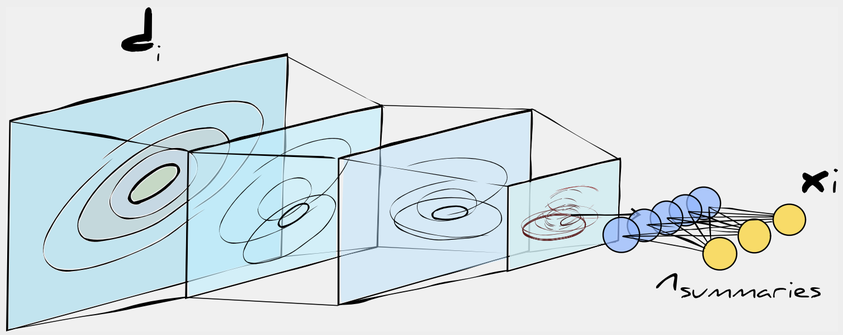
Need a good control on the likelihood of weights and hyperparameters

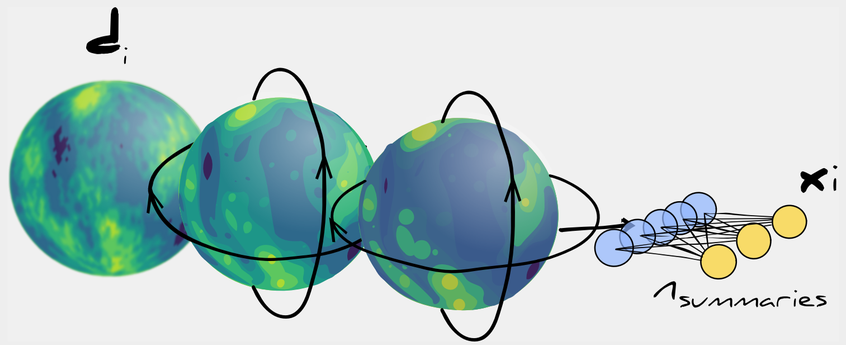
- remove degeneracies

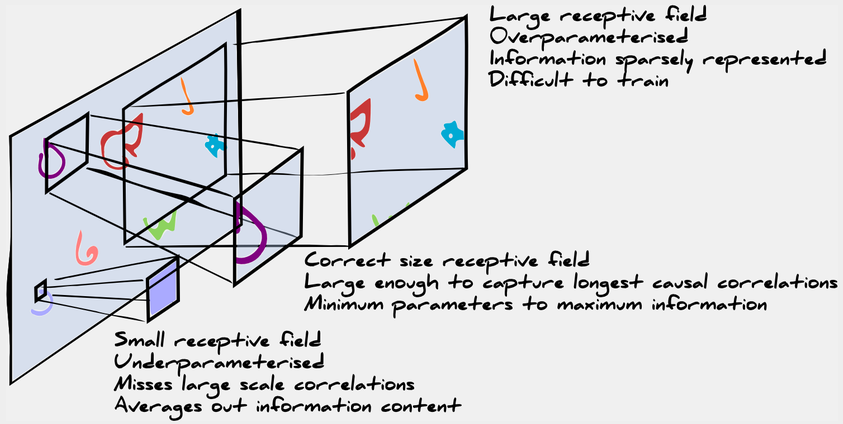
- force to be convex

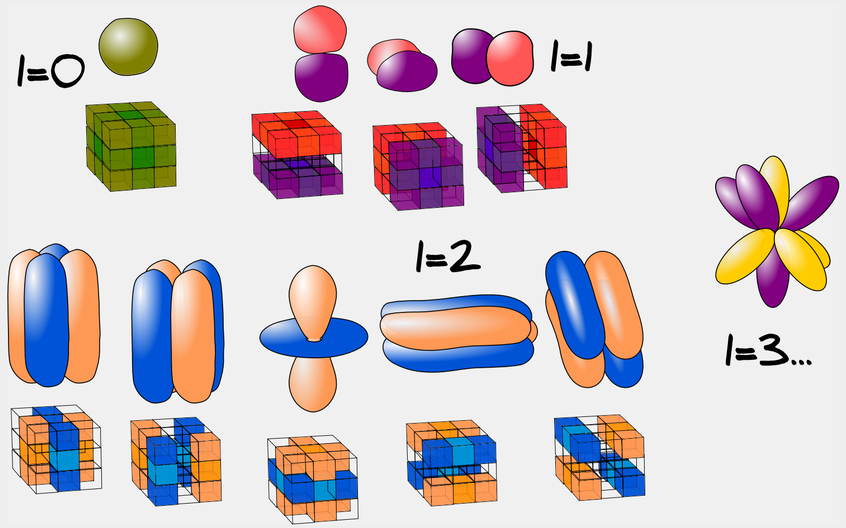
- etc.

Use neural physical engines


d_i
x_i
summaries


d_i
x_i
summaries


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


l=0
l=1
l=2
l=3...

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

Conclusions

Conclusions

Doing parameter estimation with neural networks is not as easy as predicting parameter values

We can still use results from neural networks as highly informative summaries

We can also infer all parameters of the forward model, including network to get unbiased results

Questions

When do we say we trust a neural network?

How approximate is too approximate?

Are using neural networks for parameter estimation worse than using other summary statistics?