# Week 6: Model Comparison

Probability Theory for Machine Learning

Daniel Worrall

AMLab, University of Amsterdam

October 16, 2019

## Probability and Statistics

We are mostly concerned with models which look like

$$p(\mathbf{x}|\boldsymbol{\theta}).$$

In many case $\mathbf{x}$ refers to an *observation* and $\boldsymbol{\theta}$ refers to a set of *parameters*.

**Probability**

$$\begin{tikzcd}\boldsymbol{\theta} \arrow[d, "p(\mathbf{x}|\boldsymbol{\theta})"] \\ \{\mathbf{x}_1, \mathbf{x}_2, ...\}\end{tikzcd}$$

**Statistics**

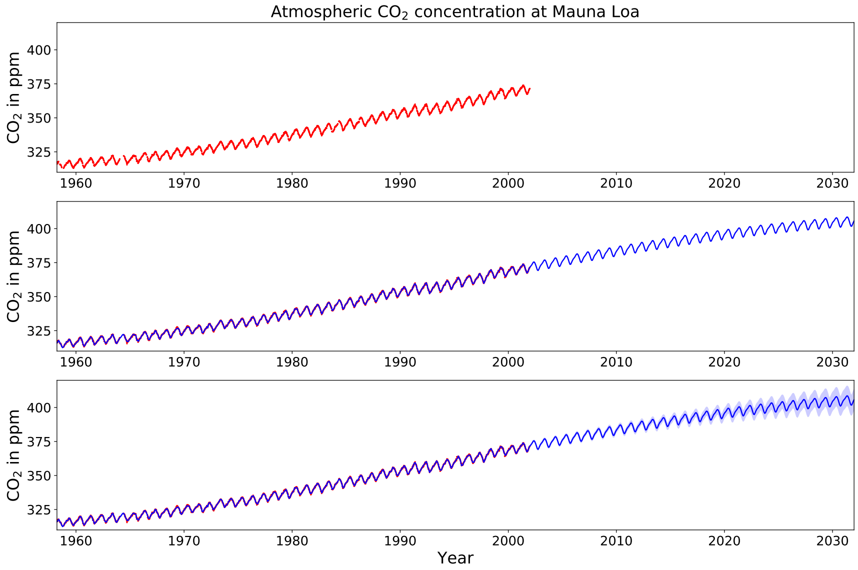
$$\begin{tikzcd}\{\mathbf{x}_1, \mathbf{x}_2, ...\} \arrow[d, "p(\mathbf{x}|\boldsymbol{\theta})\text{or}p(\boldsymbol{\theta}|\mathbf{x})"] \\ \boldsymbol{\theta}\end{tikzcd}$$

**Machine Learning**

$$\begin{tikzcd}\{\mathbf{x}_1, \mathbf{x}_2, ...\} \arrow[d, "p(\mathbf{x}|\boldsymbol{\theta})\text{or}p(\boldsymbol{\theta}|\mathbf{x})"] \\ \boldsymbol{\theta} \arrow[d, "p(\mathbf{x}_*|\boldsymbol{\theta})"] \\ \{\mathbf{x}_*\}\end{tikzcd}$$

# Machine learning

In machine learning, we use past data to make predictions about the future.



## I: Bayesian Model Comparison

## Model Comparison

Say I have some data $\mathcal{D} = \{x_i\}$, how do I pick a likelihood and a prior, aka models? A sensible idea would be to pick a few different *models* $\{\mathcal{M}_i\}$, and then find the posterior distribution over the models given the data.

$$p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{D}|\mathcal{M}_i)p(\mathcal{M}_i)$$

We can think as each choice of likelihood as a single model from a *model space* or *hypothesis space* $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, ...\}$.

Typically if we choose a uniform prior on models, so $p(\mathcal{M}_1) = p(\mathcal{M}_2) = ...$

## The Bayes' Factor

Let's consider a simple two-model comparison. To compare their probabilities we could take their ratio:

$$\frac{P(\mathcal{M}_1|\mathcal{D})}{P(\mathcal{M}_2|\mathcal{D})} = \frac{p(\mathcal{D}|\mathcal{M}_1)P(\mathcal{M}_1)/p(\mathcal{D})}{p(\mathcal{D}|\mathcal{M}_2)P(\mathcal{M}_2)/p(\mathcal{D})} = \underbrace{\frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)}}_{\text{Bayes' factor}} \cdot \underbrace{\cancel{\frac{P(\mathcal{M}_1)}{P(\mathcal{M}_2)}}}_{\text{typ. 1}}$$

This is called the *Bayes' factor*

$$K = \frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)}$$

If $K \gg 1$ then we prefer $\mathcal{M}_1$, if $K \ll 1$ then we prefer $\mathcal{M}_2$ otherwise both models are fairly equal. Sometimes we also use the log Bayes factor because it is better behaved

$$\log K = \log p(\mathcal{D}|\mathcal{M}_1) - \log p(\mathcal{D}|\mathcal{M}_2)$$

We compare this against 0.

## Example

We have some data. We know it contains values in the interval $[-1, 1]$, but we are not sure about which probability distribution to use as a good model. We consider a uniform model $\mathcal{M}_1$ and a semicircle distribution $\mathcal{M}_2$

$$\begin{aligned} p(x|\mathcal{M}_1) &= \frac{1}{2}, && -1 \leq x \leq 1 \\ p(x|\mathcal{M}_2) &= \frac{2}{\pi}\sqrt{1-x^2}, && -1 \leq x \leq 1 \end{aligned}$$

So we have that

$$\begin{aligned} p(\mathcal{D}|\mathcal{M}_1) &= \prod_{i=1}^{N} \frac{1}{2} = \left(\frac{1}{2}\right)^N \\ p(\mathcal{D}|\mathcal{M}_2) &= \prod_{i=1}^{N} \frac{2}{\pi}\sqrt{1-x_i^2} = \left(\frac{2}{\pi}\right)^N \prod_{i=1}^{N} \sqrt{1-x_i^2} \end{aligned}$$

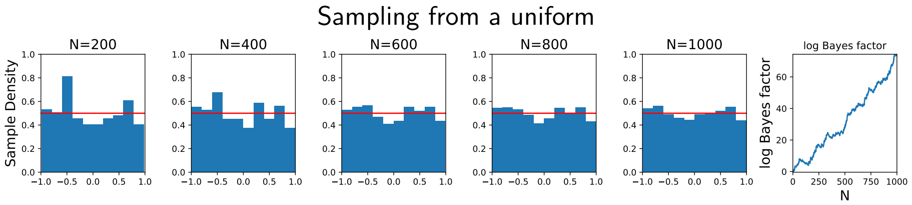
## Example

So the log Bayes factor is

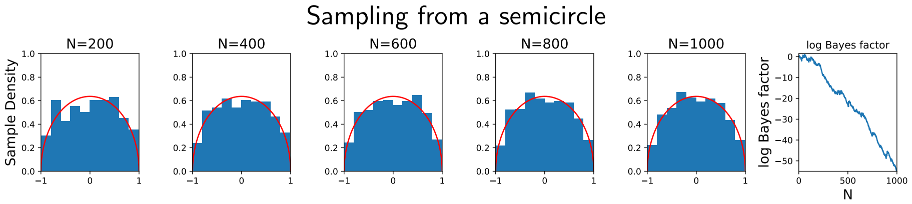
$$\begin{aligned}\log K &= \log p(\mathcal{D}|\mathcal{M}_1) - \log p(\mathcal{D}|\mathcal{M}_2) \\ &= N \log \frac{1}{2} - N \log \frac{2}{\pi} - \log \prod_{i=1}^{N} \sqrt{1 - x_i^2} \\ &= N \log \frac{\pi}{4} - \frac{1}{2} \sum_{i=1}^{N} \log(1 - x_i^2)\end{aligned}$$

Let's try this for some samples.

# Example

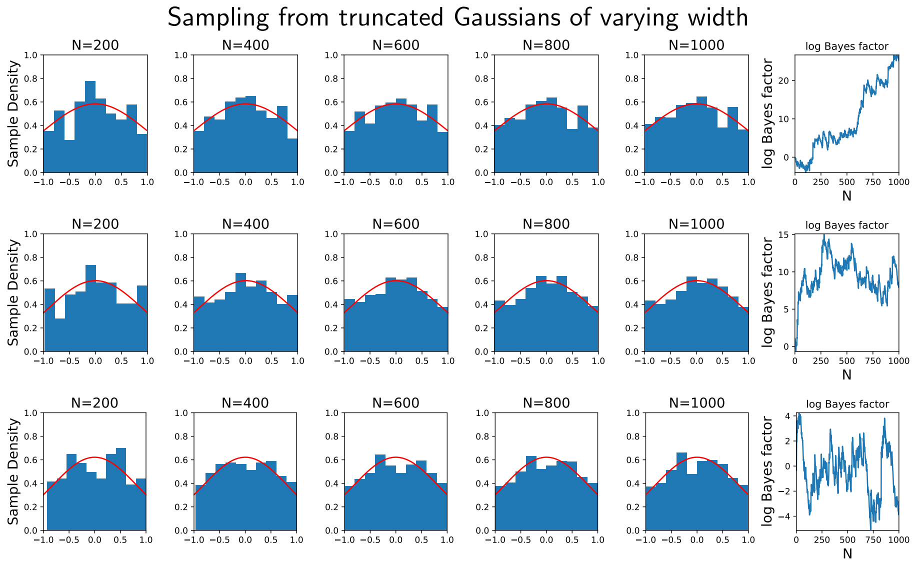


Sampling from a uniform
N=200
N=400
N=600
N=800
N=1000
log Bayes factor
Sample Density
log Bayes factor
N



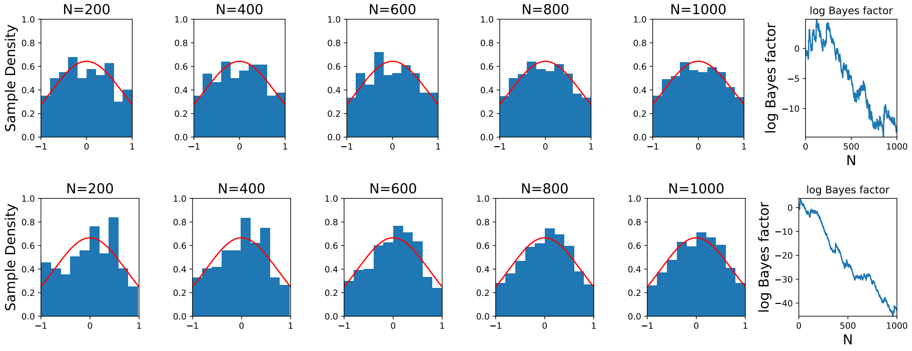
Sampling from a semicircle
N=200
N=400
N=600
N=800
N=1000
log Bayes factor
Sample Density
log Bayes factor
N

# Example



Sampling from truncated Gaussians of varying width
N=200
N=400
N=600
N=800
N=1000
log Bayes factor
Sample Density
N

# Example



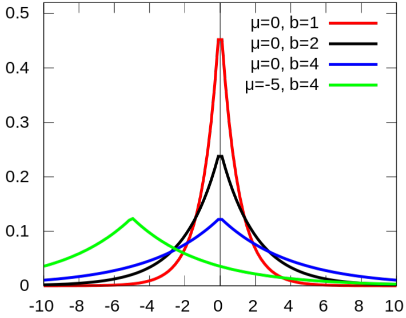
N=200
N=400
N=600
N=800
N=1000
log Bayes factor
Sample Density
log Bayes factor
N

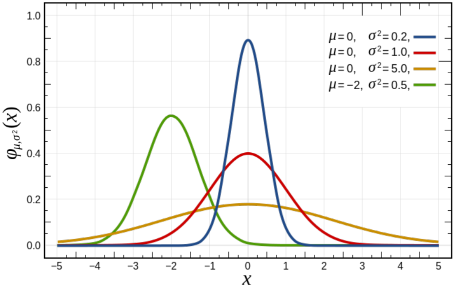
## Laplace vs. Gaussian

We have a set of data points $\{x_i\}$ with zero mean and unit variance, and we are not sure whether to model them as Laplacian distributed $\mathcal{M}_1$ or Gaussian distributed $\mathcal{M}_2$. What is the log Bayes factor for this model comparison if

$$p(x|\mathcal{M}_1) = \frac{1}{\sqrt{2}} \exp\left\{-\sqrt{2}|x|\right\}$$

$$p(x|\mathcal{M}_2) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{2}\right\}?$$





## Laplace vs. Gaussian

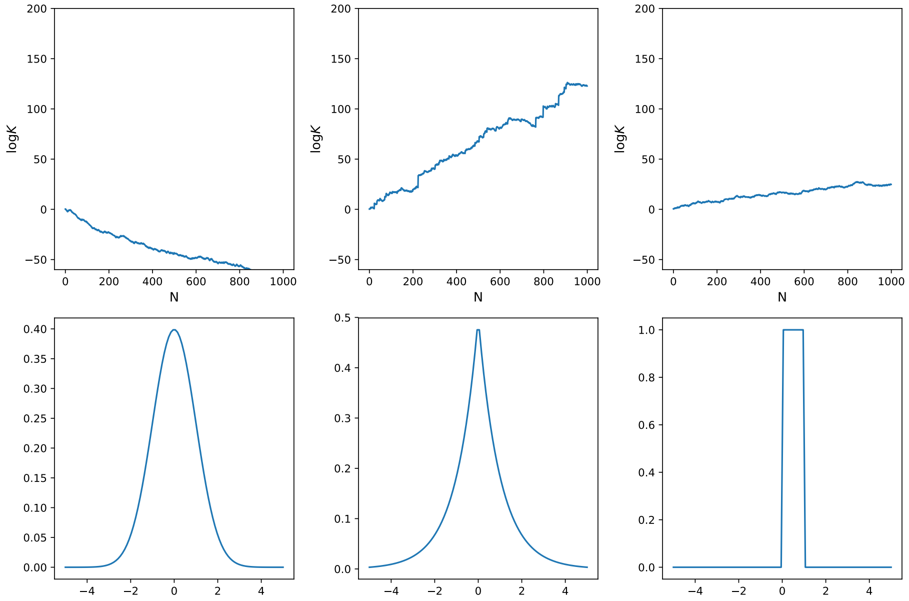
$$\begin{aligned}
\log K &= \sum_{i=1}^{N} \log\left(\frac{1}{\sqrt{2}} \exp\left\{-\sqrt{2}|x_i|\right\}\right) - \log\left(\frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x_i^2}{2}\right\}\right) \\
&= N \log \frac{1}{\sqrt{2}} - \sqrt{2}\sum_{i=1}^{N} |x_i| - N \log \frac{1}{\sqrt{2\pi}} + \frac{1}{2}\sum_{i=1}^{N} x_i^2 \\
&= N\left(\log\sqrt{\pi} - \sqrt{2}x_{\mathsf{MAD}} + \frac{1}{2}x_{\mathsf{MSE}}\right)
\end{aligned}$$

where

$$x_{\mathsf{MAD}} = \frac{1}{N}\sum_{i=1}^{N} |x_i|, \qquad x_{\mathsf{MSE}} = \frac{1}{N}\sum_{i=1}^{N} x_i^2$$

SE So if the mean absolute deviation is far larger than the mean squared error, we prefer the Gaussian and vice versa.

# Laplace vs. Gaussian



## The evidence

In the previous case the models were fairly simple, we were just given likelihood functions. What if our models had tunable parameters as well? In that case we have

$$p(\mathcal{D}|\mathcal{M}_i) = \int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{M}_i)\,\mathrm{d}\boldsymbol{\theta}$$

But this looks familiar, it's the model evidence!

$$p(\boldsymbol{\theta}|\mathcal{D},\mathcal{M}_i) = \frac{p(\mathcal{D}|\boldsymbol{\theta},\mathcal{M}_i)p(\boldsymbol{\theta}|\mathcal{M}_i)}{p(\mathcal{D}|\mathcal{M}_i)}.$$

Intuitively speaking, the evidence tells how how well a model explains the data, averaged over a variety of different parameter values.

## The evidence

Let's consider the Bayesian Bent coin from last week. In that example we had a Bernoulli likelihood model. So given $N$ coin flips with $H$ heads, we had

$$p(\pi|\mathcal{D}, \mathcal{M}_i) = \frac{\overbrace{\pi^H (1-\pi)^{N-H}}^{\text{likelihood}} \overbrace{p(\pi|\mathcal{M}_i)}^{\text{prior}}}{p(\mathcal{D}|\mathcal{M}_i)}$$

Let's compare a uniform prior ($\mathcal{M}_1$) with a Beta prior ($\mathcal{M}_2$). Last week we found that

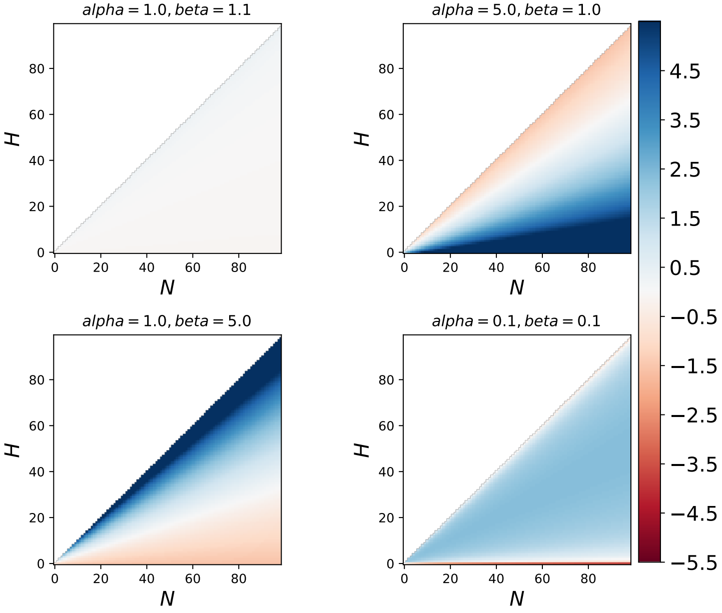
$$p(\mathcal{D}|\mathcal{M}_1) = \int_0^1 \pi^H (1-\pi)^{N-H} \mathbb{I}[\pi \in [0,1]] \, \mathrm{d}\pi = B(H+1, N-H+1)$$

$$p(\mathcal{D}|\mathcal{M}_2) = \frac{\int_0^1 \pi^H (1-\pi)^{N-H} \pi^{\alpha-1} (1-\pi)^{\beta-1} \, \mathrm{d}\pi}{B(\alpha, \beta)} = \frac{B(H+\alpha, N-H+\beta)}{B(\alpha,\beta)}$$

So the log Bayes factor is

$$\log \frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)} = \log B(H+1, N-H+1) - \log B(H+\alpha, N-H+\beta) + \log B(\alpha, \beta)$$

# The evidence



## The evidence

We are able to compare two models against one another. We can even compare a discrete set of models, by selecting the one with the highest $P(\mathcal{D}|\mathcal{M}_i)$. But did you notice that we could use this framework to select prior hyperparameters too?

$$\log p(\mathcal{D}|\mathcal{M}, \alpha, \beta) = \log B(H + \alpha, N - H + \beta) - \log B(\alpha, \beta)$$

**Question** How could we use this to select the best $\alpha$ and $\beta$?

**Answer** Pick the $\alpha$ and $\beta$ such that

$$\alpha^*, \beta^* = \arg\max_{\alpha,\beta} \log p(\mathcal{D}|\mathcal{M}, \alpha, \beta).$$

This technique goes by several names: *Type-II maximum likelihood, empircal Bayes*.

Obviously, we could add a hyperprior on the hyperparameters and perform Type-II MAP, or even find a posterior distribution over $\alpha$ and $\beta$, but after a while this gets too confusing and a bit silly.

## Exponential-Gamma example

A scientist is collecting exponentially distributed data with unknown decay parameter $\lambda$. She wants to select a good prior, so she decide to perform empirical Bayes:

$$\prod_{i=1}^{N} p(x_i|\lambda) = \prod_{i=1}^{N} \lambda e^{-\lambda x_i} = \lambda^N e^{-\lambda N \bar{x}}$$

$$p(\lambda|\alpha,\beta) = \underbrace{\frac{\beta^\alpha}{\Gamma(\alpha)}}_{\text{normalizer}} \lambda^{\alpha-1} e^{-\beta\lambda}$$

where $\bar{x} := \frac{1}{N}\sum_{i=1}^{N} x_i$. So the evidence is

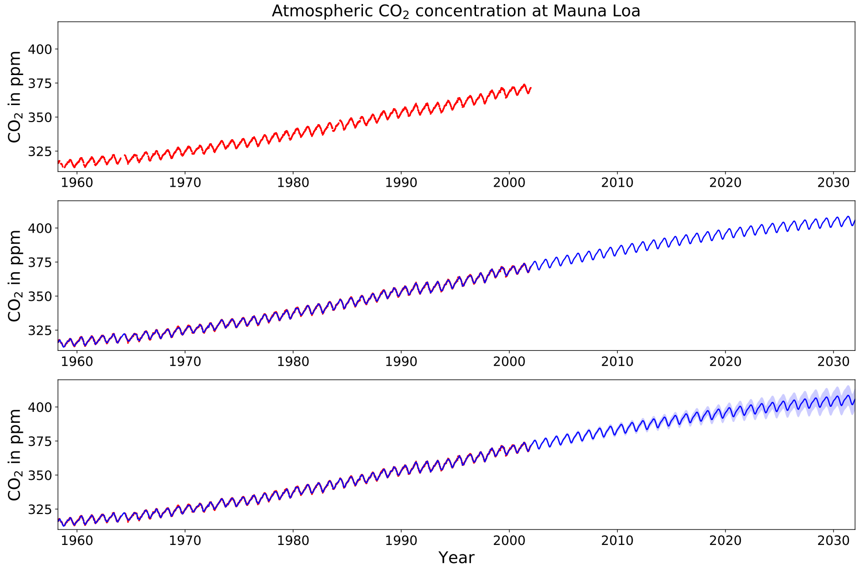
$$\begin{aligned} \int_0^\infty \lambda^N e^{-\lambda N\bar{x}} \cdot \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} \, \mathrm{d}\lambda &= \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^\infty \lambda^{N+\alpha-1} e^{-\lambda(\beta+N\bar{x})} \, \mathrm{d}\lambda \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\Gamma(N+\alpha)}{(\beta+N\bar{x})^{N+\alpha}}. \end{aligned}$$

How did we know the answer to the integral? It looks like the same form as the normalizer of the prior!

## II: Linear regression

# Regression

Recall we wanted to fit this Mauna Loa data



## Linear Regression

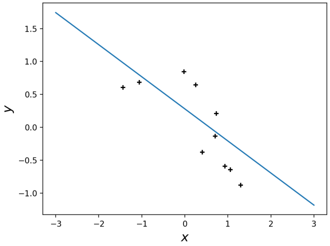
Before we had a dataset of the form
$\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_N\}$.

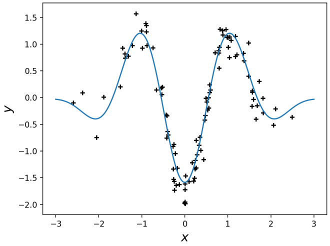
Now we have data pairs
$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), (\mathbf{x}_3, \mathbf{y}_3), ..., (\mathbf{x}_N, \mathbf{y}_N)\}$

Recall George Box *"All models are wrong"*.

- Want to predict $\mathbf{y}_* | \mathbf{x}_*$
- Want to measure 'goodness of fit'
- Want to handle noise

We use 'noise' to capture what our model cannot.





## The linear model

In high school you will have learned the equation of a straight line

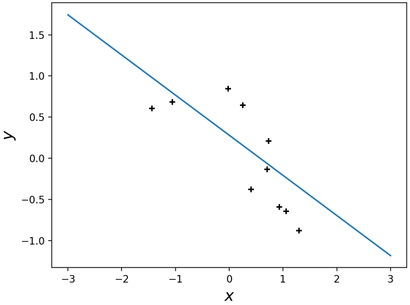
$$y = wx + b = \begin{bmatrix} w & b \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix}$$

$w$ is the gradient of the line and $b$ is the $y$-intercept. This is a linear equation.

Typically in machine learning, you will see this rewritten as

$$y = \begin{bmatrix} w & b \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = \mathbf{w}^\top \mathbf{x}$$

This separates the model parameters $\mathbf{w} = \begin{bmatrix} w & b \end{bmatrix}$ from the inputs $\mathbf{x} = \begin{bmatrix} x & 1 \end{bmatrix}^\top$.



In general a function $f : X \to Y$ is *linear* if

- $f(x_1 + x_2) = f(x_1) + f(x_2)$
- $f(ax) = af(x)$

So $y = \mathbf{w}^\top \mathbf{x}$ is linear in $\mathbf{w}$ (and $\mathbf{x}$).

## The linear model

Given a *training set* $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), ..., (\mathbf{x}_N, y_N)\}$, we might assume a linear model + a noise term $n_i \sim \mathcal{N}(n_i; 0, \sigma^2)$, so

$$y_i = \mathbf{w}^\top \mathbf{x}_i + n_i$$

This is equivalent to writing a likelihood of the form

$$p(y_i | \mathbf{w}, \mathbf{x}_i, \sigma^2) = \mathcal{N}(y_i; \mathbf{w}^\top \mathbf{x}_i, \sigma^2).$$

i.e. $y_i$ is Gaussian distributed about a mean $\mathbf{w}^\top \mathbf{x}_i$.

Now how do we fit the parameters **w** and $\sigma^2$? Next we will consider:

- Maximum likelihood
- Bayesian inference
- Type-II maximum likelihood

## The linear model

The log-likelihood is

$$\mathcal{L}(\mathbf{w}, \sigma^2) = \sum_{i=1}^{N} \log \left( \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{(y_i - \mathbf{w}^\top \mathbf{x}_i)^2}{2\sigma^2} \right\} \right)$$
$$= -\frac{N}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

To make the maths easier, we stack the data in a matrix $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_N \end{bmatrix}$ and a vector $\mathbf{y} = \begin{bmatrix} y_1 & y_2 \dots & y_N \end{bmatrix}^\top$. This leads to an alternate formulation of $\mathcal{L}$

$$\mathcal{L}(\mathbf{w}, \sigma^2) = -\frac{N}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}^\top \mathbf{w})^\top (\mathbf{y} - \mathbf{X}^\top \mathbf{w})$$

This is the same likelihood as for a model $\mathcal{L}(\mathbf{w}, \sigma^2) = \mathcal{N}(\mathbf{y} | \mathbf{X}^\top \mathbf{w}, \sigma^2 \mathbf{I})$.

## The linear model

The maximum likelihood parameter estimates are

$$\begin{aligned}\mathbf{w}_{\mathrm{ML}} &= (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y} \\ \sigma^2_{\mathrm{ML}} &= \frac{1}{N}(\mathbf{y} - \mathbf{X}^\top \mathbf{w}_{\mathrm{ML}})^\top (\mathbf{y} - \mathbf{X}^\top \mathbf{w}_{\mathrm{ML}})\end{aligned}$$

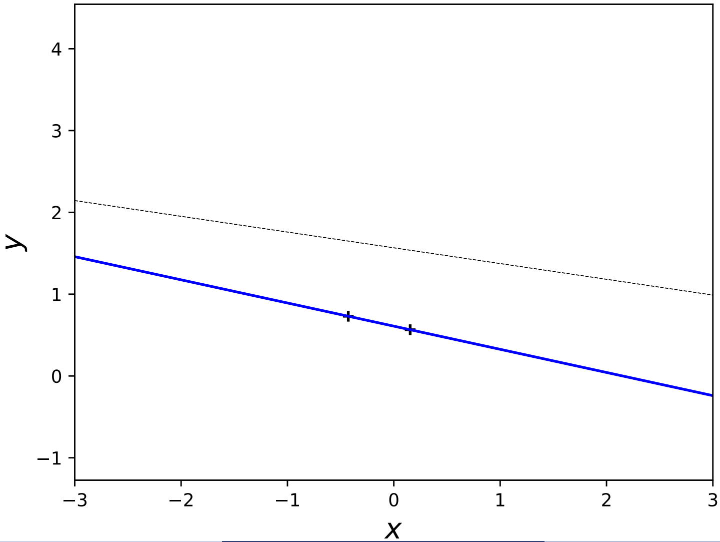
These are sometimes referred to as a the *normal equations*.

Looking at the first line, if $\mathbf{X}$ were square and full rank, then we could write

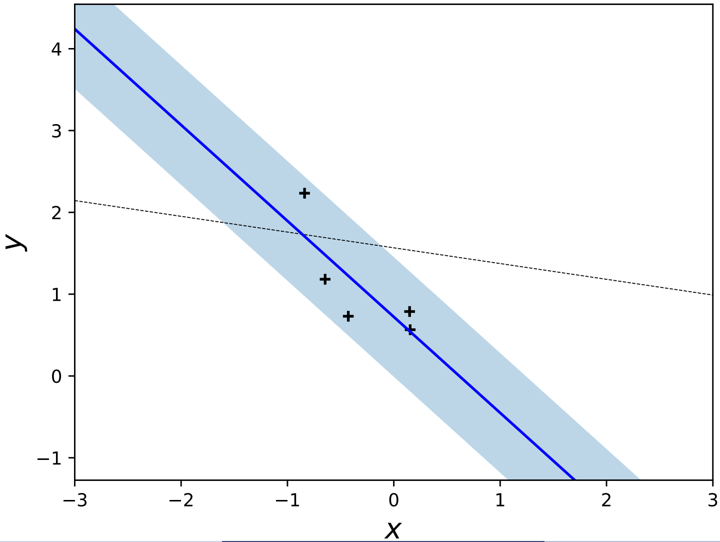
$$\mathbf{w} = (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y} = \mathbf{X}^{-\top}\mathbf{X}^{-1}\mathbf{X}\mathbf{y} = \mathbf{X}^{-\top}\mathbf{y}$$

so $\mathbf{X}^\top \mathbf{w} = \mathbf{y}$, which is the equation for the mean of the regression model. So why we use $(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}$ instead of $\mathbf{X}^{-\top}$? It is the pseudoinverse of $\mathbf{X}^\top$. Recall that the pseudoinverse is $\mathbf{X}^\dagger = (\mathbf{X}^\top\mathbf{X})\mathbf{X}^\top$.

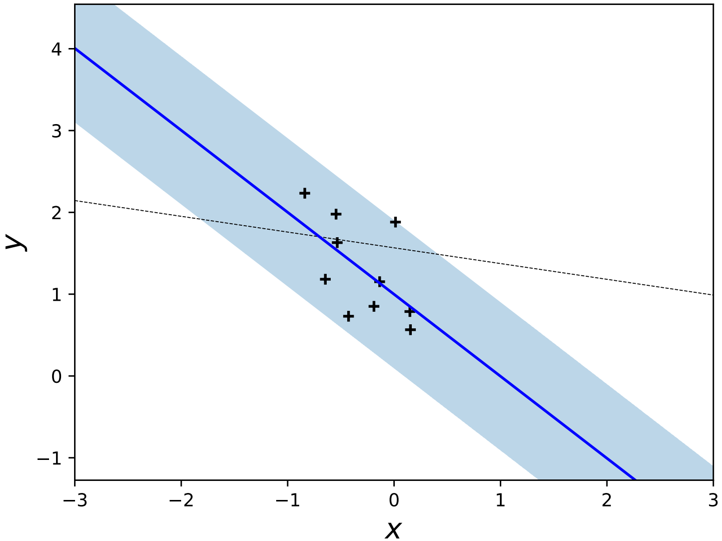
## A 2D example: N=2



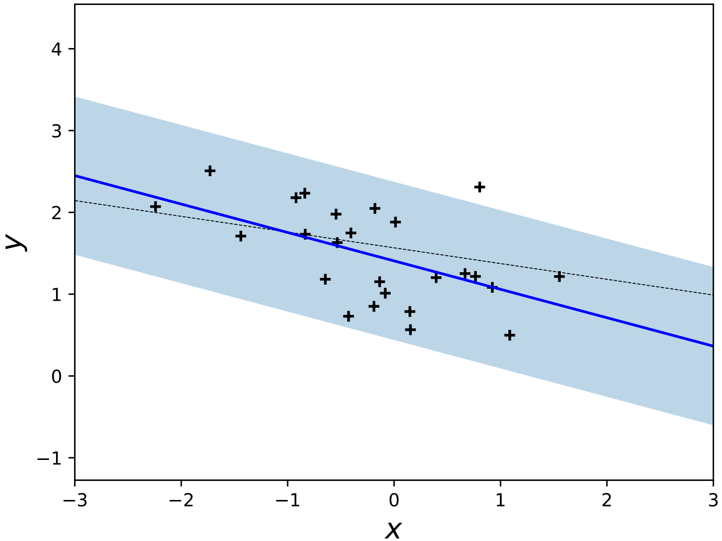
## A 2D example: N=5



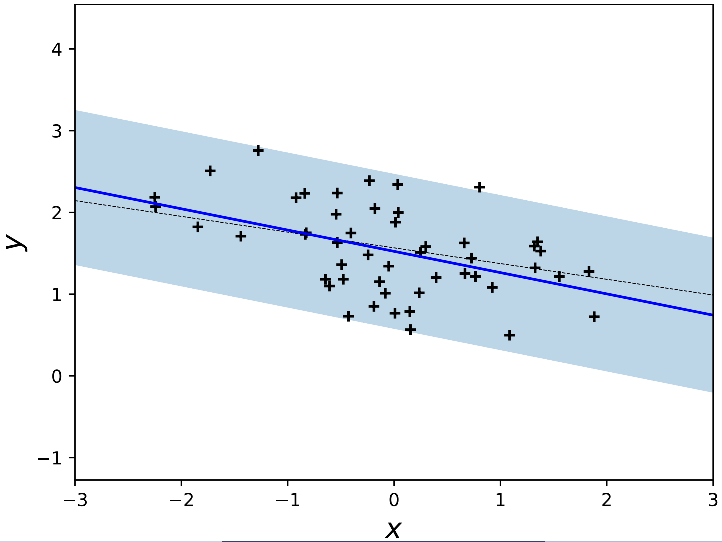
## A 2D example: N=10



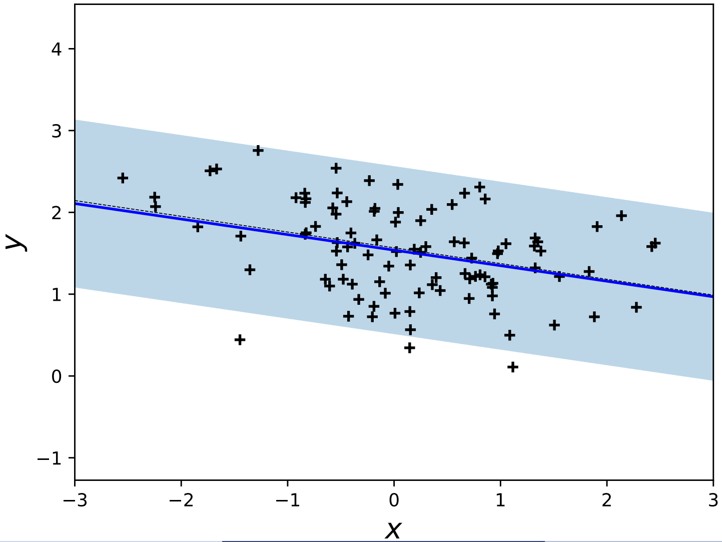
# A 2D example: N=25



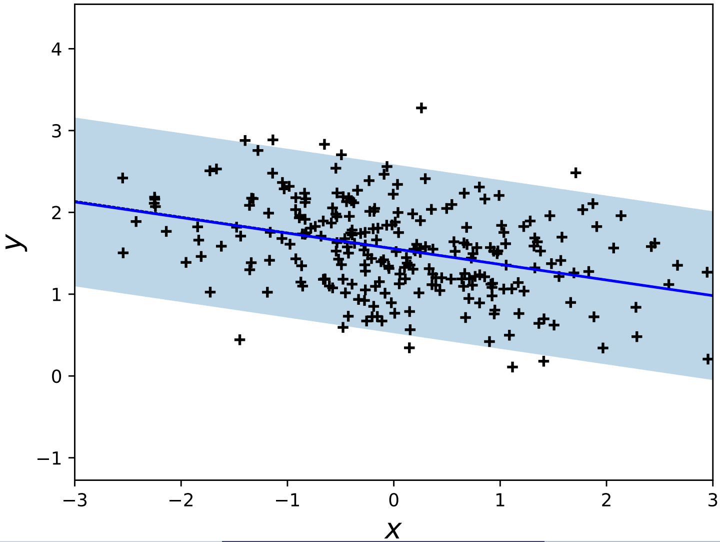
## A 2D example: N=50



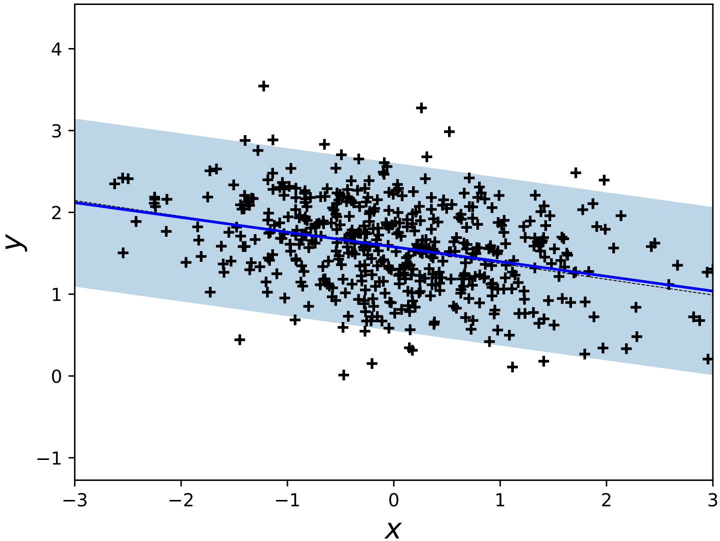
## A 2D example: N=100



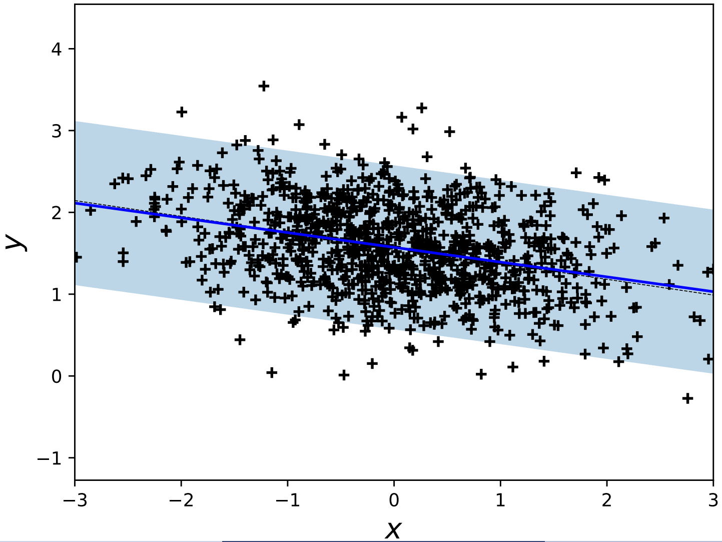
# A 2D example: N=250



## A 2D example: N=500

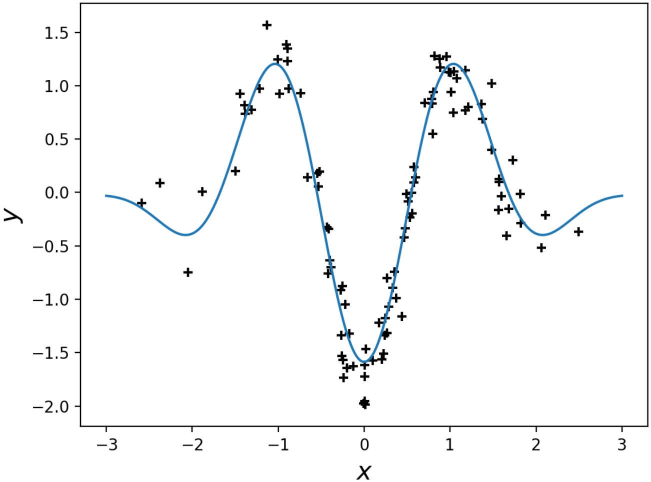


## A 2D example: N=1000



## Feature transformations

The model we just presented is easy to use, but how useful is it really? What happens if we are presented with data like this?



## Feature transformations

The solution is very simple, and is one of the cornerstones of machine learning. Say instead of the linear equation

$$y = wx + b,$$

we had something more complex such as

$$\begin{aligned} y &= w_1 \cdot \phi_1(x) + w_2 \cdot \phi_2(x) + w_3 \cdot \phi_3(x) + w_4 \cdot \phi_4(x) + b \cdot 1 \\ &= \begin{bmatrix} w_1 & w_2 & w_3 & w_4 & b \end{bmatrix} \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \\ \phi_3(x) \\ \phi_4(x) \\ 1 \end{bmatrix} \\ &= \mathbf{w}^\top \boldsymbol{\phi}(x). \end{aligned}$$

The function $\boldsymbol{\phi}(x)$ is called a feature transform. It is generally a nonlinear transformation of the data space $\mathcal{X}$, typically lifting to higher dimensions.

## Feature transformations

An example of a feature transformation is the trigonometric mapping

$$\boldsymbol{\phi}(x) = \begin{bmatrix} \cos x & \sin x & \cos 2x & \sin 2x & \cdots & \cos Nx & \sin Nx & 1 \end{bmatrix}^\top .$$

This is a good way to represent periodic functions. It is related to a common decomposition of functions you may have come across, the Fourier transform.

Another feature transform is the monomials

$$\boldsymbol{\phi}(x) = \begin{bmatrix} x & x^2 & x^3 & \cdots & x^N & 1 \end{bmatrix}^\top .$$

This is good for modelling polynomial functions.

Another is the radial basis function

$$\boldsymbol{\phi}(x) = \begin{bmatrix} e^{\lambda_1 \|x-\mu_1\|} & e^{\lambda_2 \|x-\mu_2\|} & e^{\lambda_3 \|x-\mu_3\|} & \dots & e^{\lambda_N \|x-\mu_N\|} & 1 \end{bmatrix}^\top$$

This is handy for locally approximating smooth functions.

## Feature transformations

So the equation we now have is

$$y = \mathbf{w}^\top \boldsymbol{\phi}(x).$$

This is nonlinear in $x$, but it is linear in **w** still since

$$(\mathbf{w}_1 + \mathbf{w}_2)^\top \boldsymbol{\phi}(x) = \mathbf{w}_1^\top \boldsymbol{\phi}(x) + \mathbf{w}_2^\top \boldsymbol{\phi}(x) = y_1 + y_2$$

and we can also show that it is linear in $\boldsymbol{\phi}(x)$.

So given a training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), ..., (\mathbf{x}_N, y_N)\}$ we might try to fit the model

$$y_i = \mathbf{w}^\top \boldsymbol{\phi}(x_i) + n_i$$

where $n_i \sim \mathcal{N}(n_i; 0, \sigma^2)$ again. This can also be written

$$p(y_i|\mathbf{w}, \boldsymbol{\phi}(x_i), \sigma^2) = \mathcal{N}(y_i, \mathbf{w}^\top \boldsymbol{\phi}(x_i), \sigma^2)$$

## Feature transformations

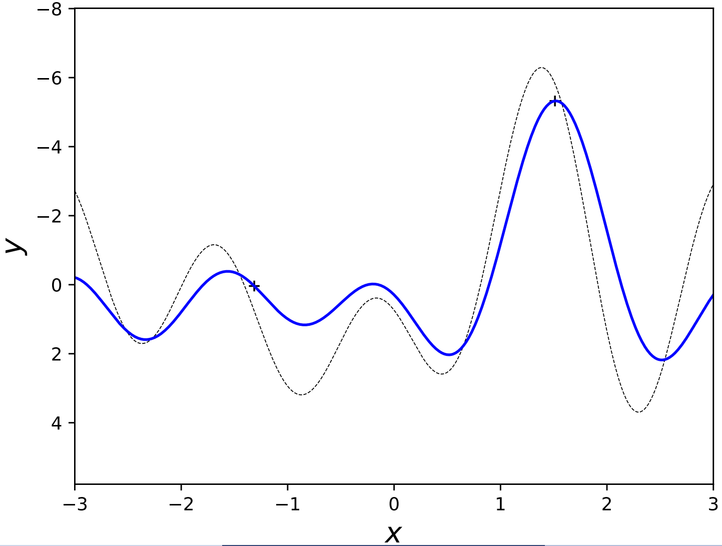
Going through the movements of maximum likelihood again, we end up at the normal equations

$$\mathbf{w}_{\mathsf{ML}} = (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top)^{-1}\boldsymbol{\Phi}\mathbf{y}$$
$$\sigma^2_{\mathsf{ML}} = \frac{1}{N}(\mathbf{y} - \boldsymbol{\Phi}^\top\mathbf{w}_{\mathsf{ML}})^\top(\mathbf{y} - \boldsymbol{\Phi}^\top\mathbf{w}_{\mathsf{ML}})$$

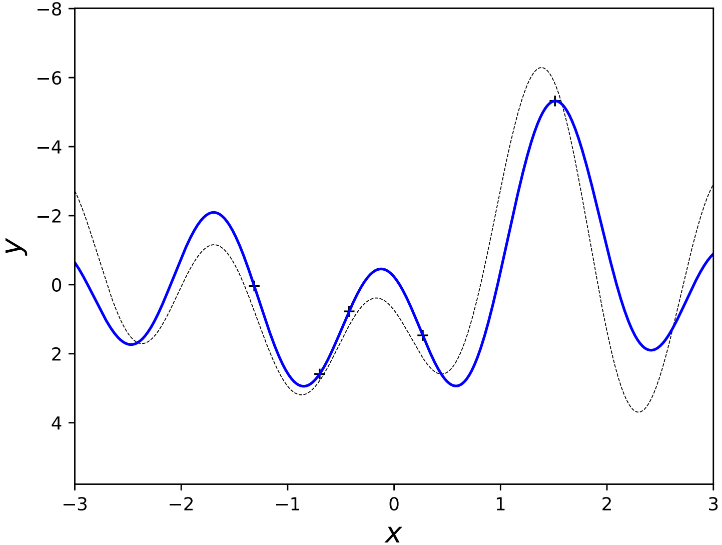
Notice that the only difference with the linear equations on $\mathbf{X}$ is that we have replaced $\mathbf{X}$ with $\boldsymbol{\Phi}$. This is because the original equations are linear in $\mathbf{w}$ and $\boldsymbol{\Phi}$ (but not $\mathbf{X}$).

A useful way to think about what we have done is to see the feature mapping $x \mapsto \phi(\mathbf{x})$ as mapping the data to a space, a *feature space*, where it lies on a straight line.

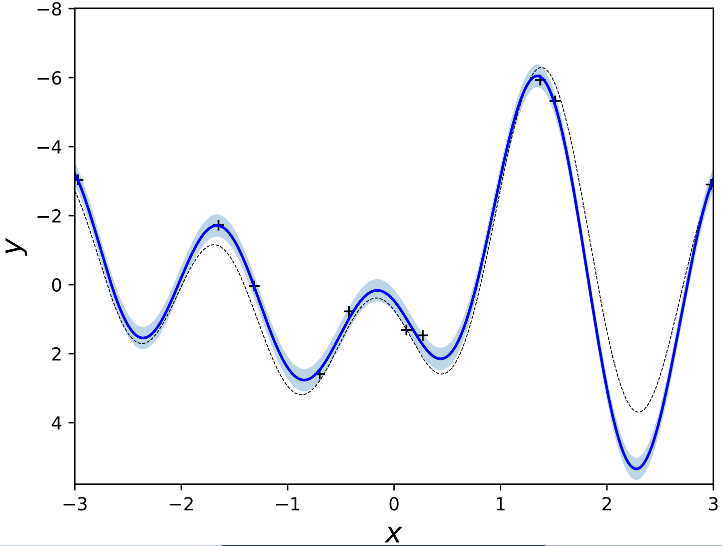
## A 2D example: N=2



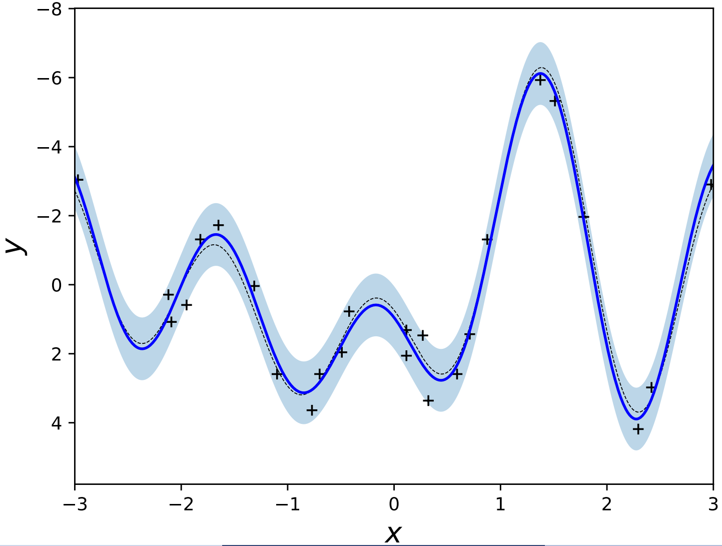
## A 2D example: N=5



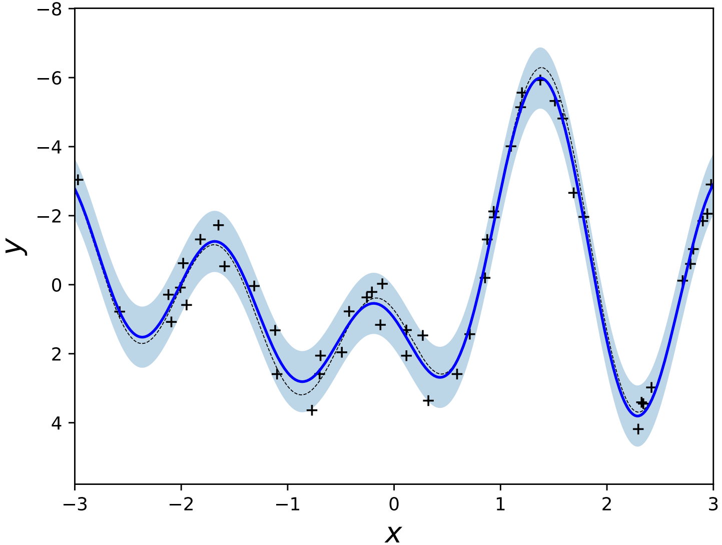
## A 2D example: N=10



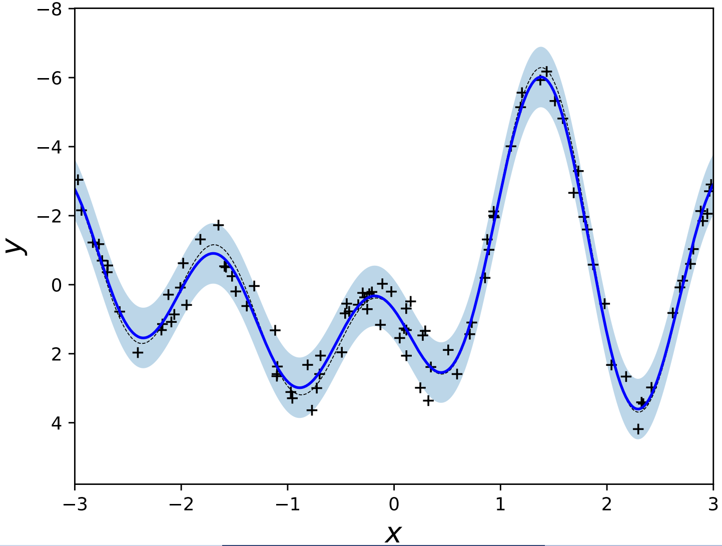
# A 2D example: N=25



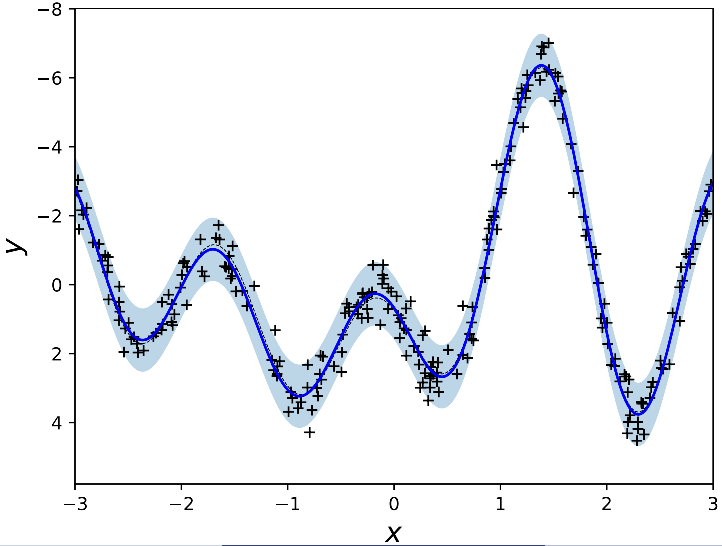
# A 2D example: N=50



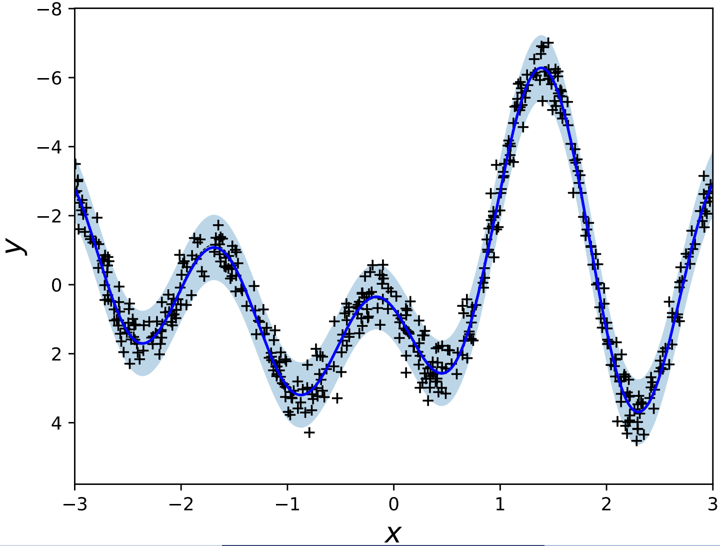
# A 2D example: N=100



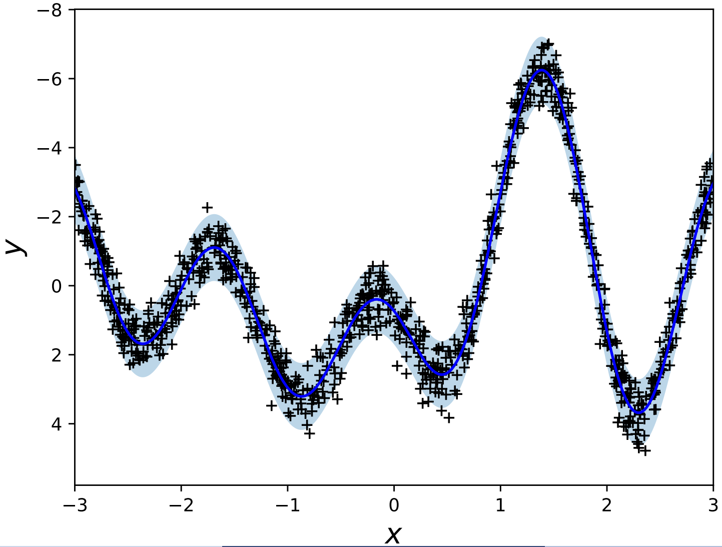
## A 2D example: N=250



## A 2D example: N=500

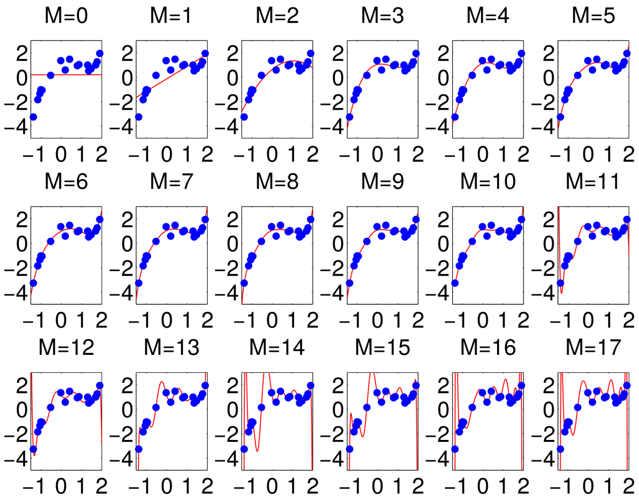


# A 2D example: N=1000



## Feature transformations

The issue with maximum likelihood is that it tends to *overfit*.



Slide from Rasmussen and Gharamani 4F13

## Bayesian linear regression

The Bayesian way to do regression protects against overfitting in the low-data scenario. We shall place a prior on the parameters **w**. Let's just do this on the mean. We choose a zero mean Gaussian with covariance matrix $\sigma^2_{\text{prior}}\mathbf{I}$. This leads to the following equations

**Forward model**

$$p(\mathbf{w}|\mathbf{0}, \sigma^2_{\text{prior}}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma^2_{\text{prior}}\mathbf{I})$$

$$p(\mathbf{y}|\mathbf{w}, \mathbf{X}, \sigma^2_{\text{lik}}) = \mathcal{N}(\mathbf{y}|\mathbf{w}^\top\mathbf{X}, \sigma^2_{\text{lik}}\mathbf{I})$$

**Posterior**

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}, \sigma^2_{\text{lik}}) = \frac{\mathcal{N}(\mathbf{y}|\mathbf{w}^\top\mathbf{X}, \sigma^2_{\text{lik}}\mathbf{I})\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma^2_{\text{prior}}\mathbf{I})}{\int_{\mathbf{W}} \mathcal{N}(\mathbf{y}|\mathbf{w}^\top\mathbf{X}, \sigma^2_{\text{lik}}\mathbf{I})\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma^2_{\text{prior}}\mathbf{I})\,\mathrm{d}\mathbf{w}} = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

$$\boldsymbol{\mu} = \frac{1}{\sigma^2_{\text{lik}}}\boldsymbol{\Sigma}\mathbf{X}\mathbf{y}, \qquad \boldsymbol{\Sigma} = \left(\frac{1}{\sigma^2_{\text{prior}}}\mathbf{I} + \frac{1}{\sigma^2_{\text{lik}}}\mathbf{X}\mathbf{X}^\top\right)^{-1}$$

Notice how the posterior precision is a mean of the prior precision and the data-weighted likelihood precision

## Bayesian linear regression

**Posterior predictive**

We need to figure out the posterior predictive distribution (PPD). Lucky the PPD is also a Gaussian! So

$$p(y_*|\mathbf{x}_*, \mathcal{D}, \sigma^2_{\text{lik}}) = \int_{\mathbf{w}} \mathcal{N}(y_*|\mathbf{w}^\top \mathbf{x}, \sigma^2_{\text{lik}}) \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \, \mathrm{d}\mathbf{w} = \mathcal{N}(y_*|\boldsymbol{\mu}^\top \mathbf{x}, \sigma^2_{\text{lik}} + \mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x})$$

The PPD uses the mean of the posterior distribution. The variance is a sum of $\sigma^2_{\text{lik}}$ and $\mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x}$. This extra term gets larger the larger the data term is, as measured under the posterior covariance.

**Evidence**

$$\begin{aligned}
\log p(\mathcal{D}|\sigma^2_{\text{lik}}, \sigma^2_{\text{prior}}) &= p(\mathbf{y}|\mathbf{X}, \sigma^2_{\text{lik}}, \sigma^2_{\text{prior}}) = \log \int_{\mathbf{w}} \mathcal{N}(\mathbf{y}|\mathbf{w}^\top \mathbf{X}, \sigma^2_{\text{lik}}) \mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma^2_{\text{prior}} \mathbf{I}) \, \mathrm{d}\mathbf{w} \\
&= \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \underbrace{\sigma^2_{\text{lik}} \mathbf{I} + \sigma^2_{\text{prior}} \mathbf{X}^\top \mathbf{X}}_{\mathbf{K}}) \\
&= -\frac{1}{2} \mathbf{y}^\top \mathbf{K}^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K}| - \frac{N}{2} \log 2\pi
\end{aligned}$$

## Conclusion

**Rounding up**

Given data $\mathcal{D} = \{\mathbf{x}_i\}$ and a forward model $p(\mathbf{x}|\boldsymbol{\theta})$, you should be able to

- Find $\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta})$
- Find $p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$
- Know what the posterior predictive is $p(x_*|\mathcal{D})$

Given data $\mathcal{D} = \{\mathbf{x}_i\}$ and a collections of models $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, ...\}$, you should be able to

- Find the (log) Bayes' factor $\log K = p(\mathcal{D}|\mathcal{M}_1)/p(\mathcal{D}|\mathcal{M}_2)$
- Find best model in $\mathcal{M}$: $\mathcal{M}^* = \arg\max_{\mathcal{M}_i \in \mathcal{M}} p(\mathbf{x}|\mathcal{M}_i)$

## Mixture models

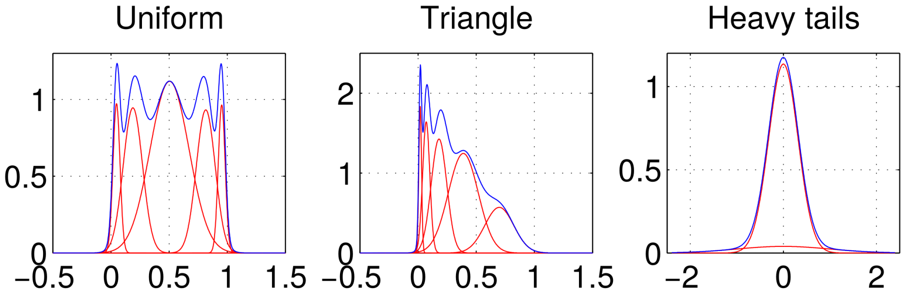
III: Mixture Models

## Mixture of Gaussians

Some complicated distributions can be modelled as *mixture models*. Take for instance the *mixture of Gaussians* (MoG) in 1D

$$s_i \sim \mathsf{Cat}(\boldsymbol{\pi})$$
$$x_i | s_i \sim \mathcal{N}(x_i; \mu_{s_i}, \sigma^2_{s_i})$$

where $s_i \in \{1, 2, ..., M\}$.



## Clustering

This is a canonical example of what is known as a *latent variable model*. The cluster identity $s_i$ is known as a *latent variable*. Note that there is one latent variable per datum $x_i$. Denoting $\theta_m = \{\mu_m, \sigma_m^2\}$, this has a single-point likelihood

$$
\begin{aligned}
p(x_i|\boldsymbol{\pi}) &= \sum_{m=1}^{M} P(s=m)p(x_i|s=m) \\
&= \sum_{m=1}^{M} \pi_m p(x_i|\theta_m)
\end{aligned}
$$

Before observing any data, the term $P(s=m) = \pi_m$ is the *prior* probability that we expect a datum $x$ to be produced by cluster $m$.

The posterior probability of assigning a point $x_i$ to cluster $m$ is

$$P(s_i = m|x_i) = \frac{p(x_i|s_i = m)\pi_m}{\sum_{m'} p(x_i|s_i = m')\pi_{m'}} = r_{mi}$$

This is often called the *responsibility*.

## Training

The total log-likelihood is

$$\mathcal{L}(\theta, \boldsymbol{\pi}) = \sum_{i=1}^{N} \log \sum_{m=1}^{M} \pi_m p(x_i|\theta_m)$$

The $\log \sum$ is a notorious computationally intractable term[1]. So we have to use numerical methods to optimize this log-likelihood. Thus we need derivatives

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \theta_m} &= \frac{\partial}{\partial \theta_m} \sum_{i=1}^{N} \log \sum_{m=1}^{M} \pi_m p(x_i|\theta_m) \\ &= \sum_{i=1}^{N} \frac{\pi_m}{\sum_{m=1}^{M} \pi_m p(x_i|\theta_m)} \frac{\partial p(x_i|\theta_m)}{\partial \theta_m}\end{aligned}$$

[1]In practice, we actually optimize a lower bound on the log-likelihood using a technique known as variational inference.

## Training

A very important trick, which comes up again and again is the *log-derivative trick*

$$\frac{\partial \log p(x_i|\theta_m)}{\partial \theta_m} = \frac{1}{p(x_i|\theta_m)} \frac{\partial p(x_i|\theta_m)}{\partial \theta_m} \implies p(x_i|\theta_m) \frac{\partial \log p(x_i|\theta_m)}{\partial \theta_m} = \frac{\partial p(x_i|\theta_m)}{\partial \theta_m}$$

So

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \theta_m} &= \sum_{i=1}^{N} \frac{\pi_m}{\sum_{m=1}^{M} \pi_m p(x_i|\theta_m)} \frac{\partial p(x_i|\theta_m)}{\partial \theta_m} \\ &= \sum_{i=1}^{N} \underbrace{\frac{\pi_m p(x_i|\theta_m)}{\sum_{m=1}^{M} \pi_m p(x_i|\theta_m)}}_{r_{mi}} \frac{\partial \log p(x_i|\theta_m)}{\partial \theta_m} \\ &= \sum_{i=1}^{N} r_{mi} \frac{\partial \log p(x_i|\theta_m)}{\partial \theta_m}\end{aligned}$$

The gradient of the log-likelihood of each point $x_i$ wrt each cluster parameters $\theta_m$ is reweighted by the posterior probability that that cluster explains data point $x_i$.

# Training

We also find

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \pi_m} &= \frac{\partial}{\partial \pi_m} \sum_{i=1}^{N} \log \sum_{m=1}^{M} \pi_m p(x_i|\theta_m) \\ &= \sum_{i=1}^{N} \frac{p(x_i|\theta_m)}{\sum_{m=1}^{M} \pi_m p(x_i|\theta_m)} \\ &= \sum_{i=1}^{N} \frac{r_{mi}}{\pi_m}\end{aligned}$$

## The Expectation-Maximization Algorithm

By optimizing $\mathcal{L}$ we can cluster. This is a specific instance of a famous algorithm called the *Expectation-Maximization algorithm*

E-step

$$r_{mi}^k = \frac{\pi_m^k p(x_i|\theta_m^k)}{\sum_{m=1}^M \pi_m^k p(x_i|\theta_m^k)}$$

M-step

$$\theta^{k+1}, \boldsymbol{\pi}^{k+1} = \arg\max_{\theta, \boldsymbol{\pi}} \mathcal{L}(\theta, \boldsymbol{\pi})$$

The E-step gets its name from the fact that you are computing the expected assignments.

# The Expectation-Maximization Algorithm

For the M-step, we notice that

$$\frac{\partial \mathcal{L}}{\partial \pi_m} = \sum_{i=1}^{N} \frac{r_{mi}^k}{\pi_m} = 0 \implies \pi_m^{k+1} = \sum_{i=1}^{N} r_{mi}^k$$

If we use a Mixture of Gaussians, then

$$\frac{\partial \mathcal{L}}{\partial \mu_m} = \sum_{i=1}^{N} r_{mi}^k \frac{x_i - \mu_m}{\sigma_m^2} = 0 \qquad \implies \mu_m^{k+1} = \frac{\sum_{i=1}^N r_{mi}^k x_i}{\sum_{i=1}^N r_{mi}}$$

$$\frac{\partial \mathcal{L}}{\partial \sigma_m} = -\sum_{i=1}^{N} r_{mi}^k \left( \frac{\sigma_m^2 + (x_i - \mu_m)^2}{\sigma_m^3} \right) = 0 \qquad \implies \sigma_m^{2,k+1} = \frac{\sum_{i=1}^N r_{mi}^k (x_i - \mu_m)^2}{\sum_{i=1}^N r_{mi}^k}$$

The mean update is the responsibility weighted

# The Expectation-Maximization Algorithm

Notice that each update can be rewritten as

$$\mu_m^{k+1} = \sum_{i=1}^{N} \tilde{r}_{mi}^k x_i$$

$$\sigma_m^{2,k+1} = \sum_{i=1}^{N} \tilde{r}_{mi}^k (x_i - \mu_m)^2$$

where

$$\tilde{r}_{mi}^k = \frac{r_{mi}^k}{\sum_{i=1}^N r_{mi}^k}$$

represents the contribution of point $x_i$ to cluster $m$.

## The Expectation-Maximization Algorithm

The complete algorithm is

E-step

$$r_{mi} = \frac{\pi_m^k p(x_i|\theta_m^k)}{\sum_{m=1}^M \pi_m^k p(x_i|\theta_m^k)}$$

M-step

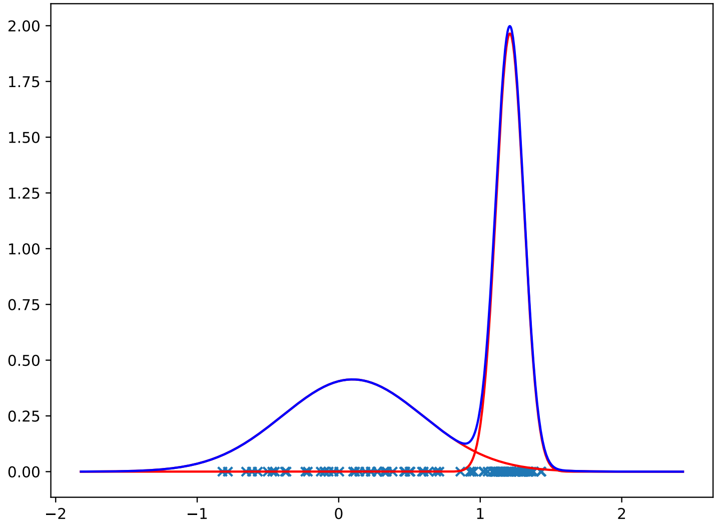
$$\pi_m^{k+1} = \sum_{i=1}^N r_{mi}$$

$$\mu_m^{k+1} = \sum_{i=1}^N \tilde{r}_{mi}^k x_i$$

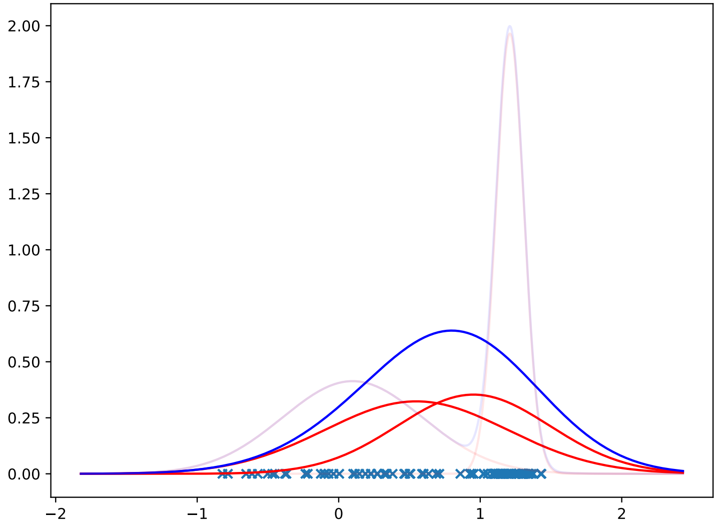
$$\sigma_m^{2,k+1} = \sum_{i=1}^N \tilde{r}_{mi}^k (x_i - \mu_m)^2$$

$$\tilde{r}_{mi}^k = \frac{r_{mi}^k}{\sum_{i=1}^N r_{mi}^k}$$

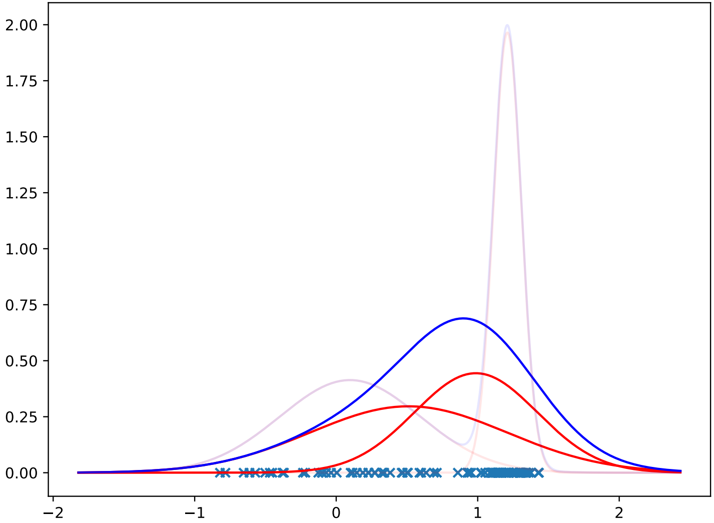
## The Expectation-Maximization Algorithm



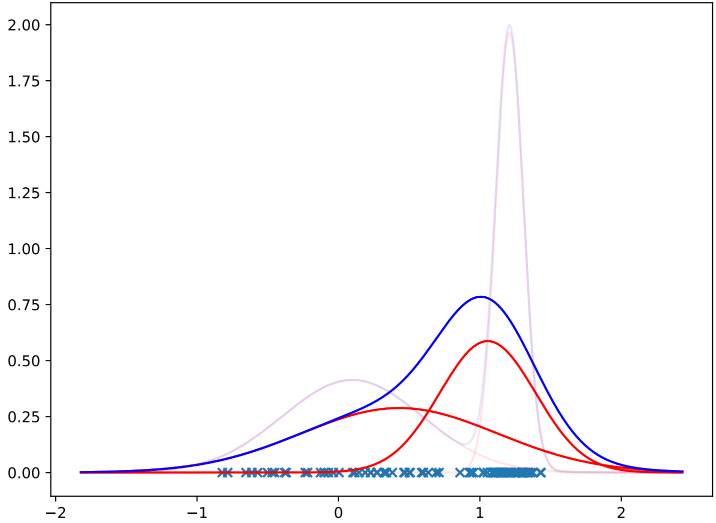
# The Expectation-Maximization Algorithm



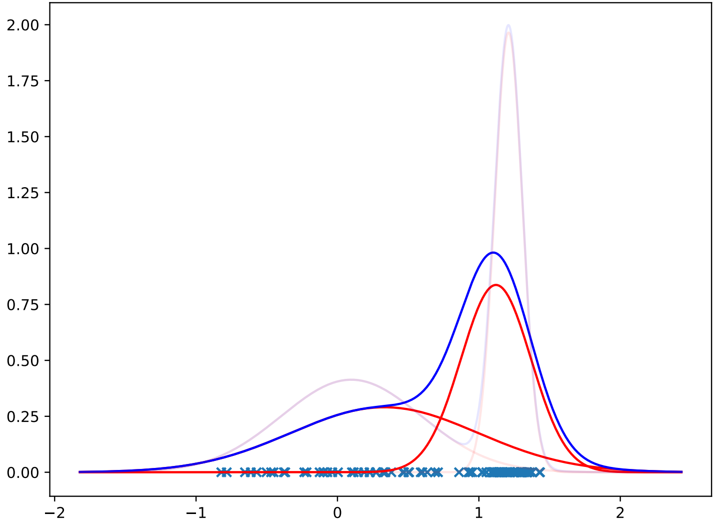
## The Expectation-Maximization Algorithm



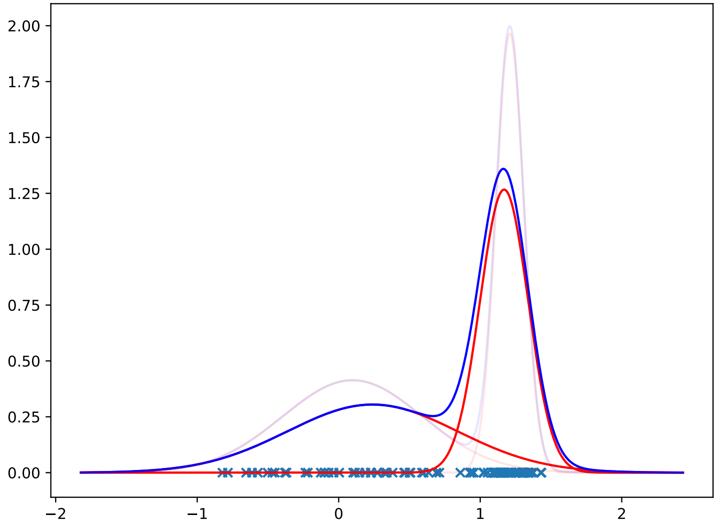
## The Expectation-Maximization Algorithm



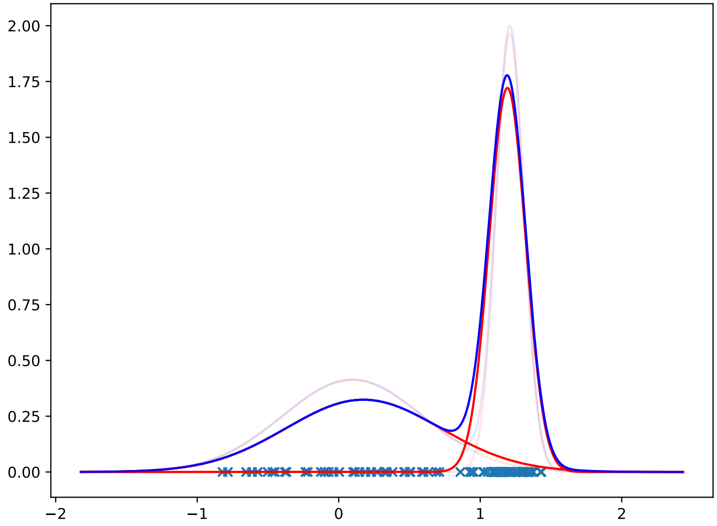
# The Expectation-Maximization Algorithm



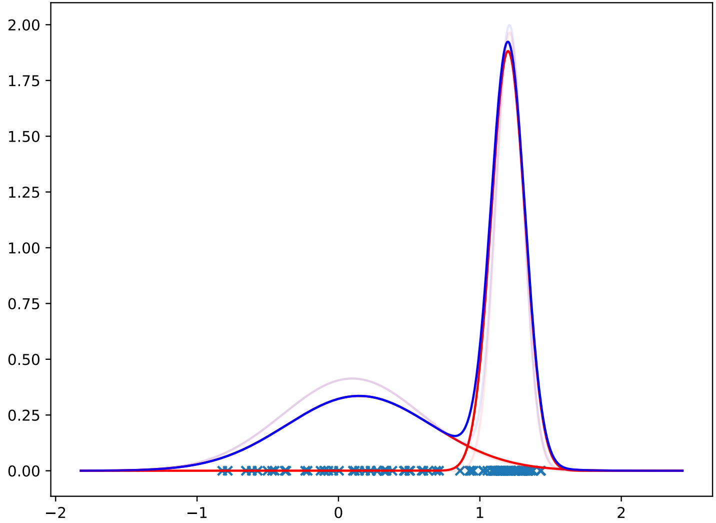
# The Expectation-Maximization Algorithm



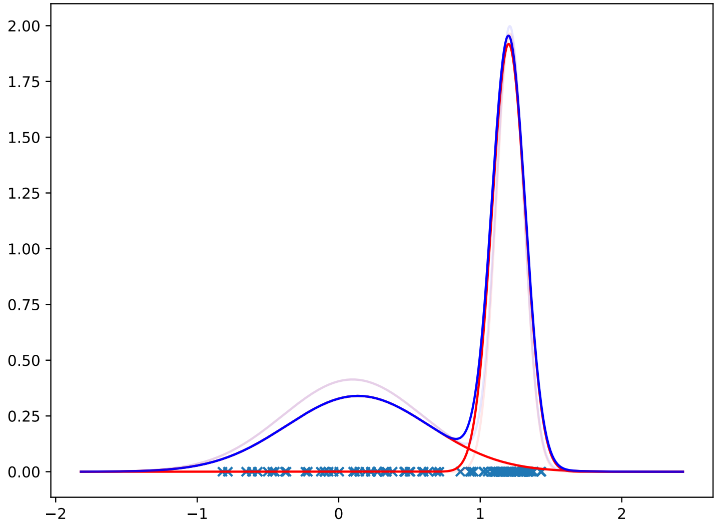
## The Expectation-Maximization Algorithm



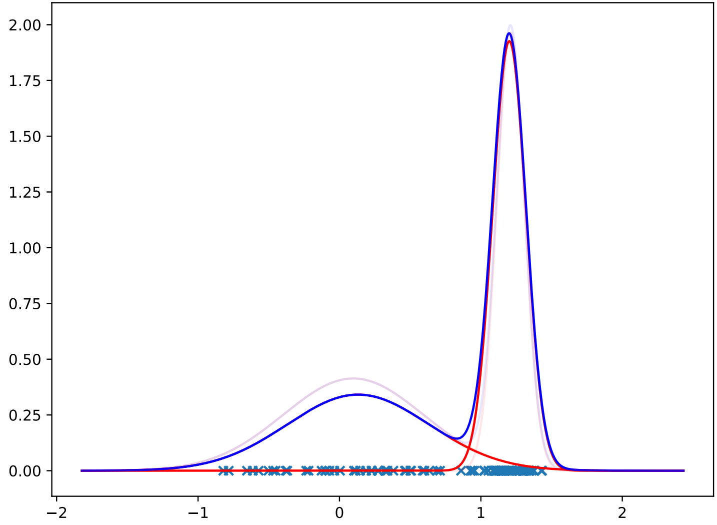
## The Expectation-Maximization Algorithm



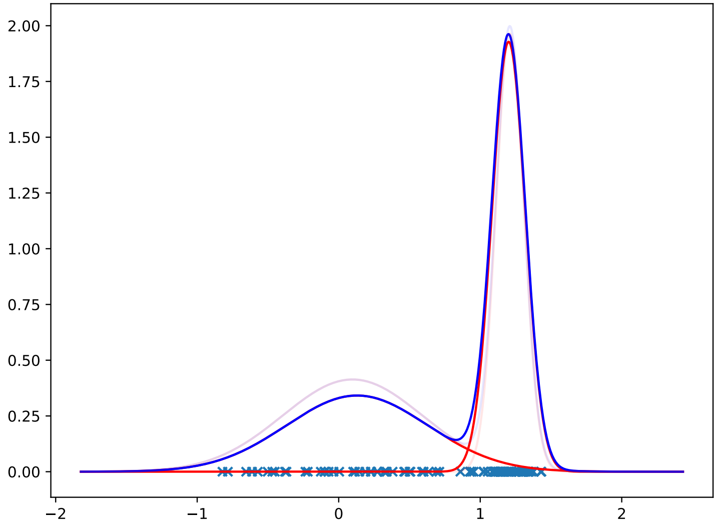
## The Expectation-Maximization Algorithm



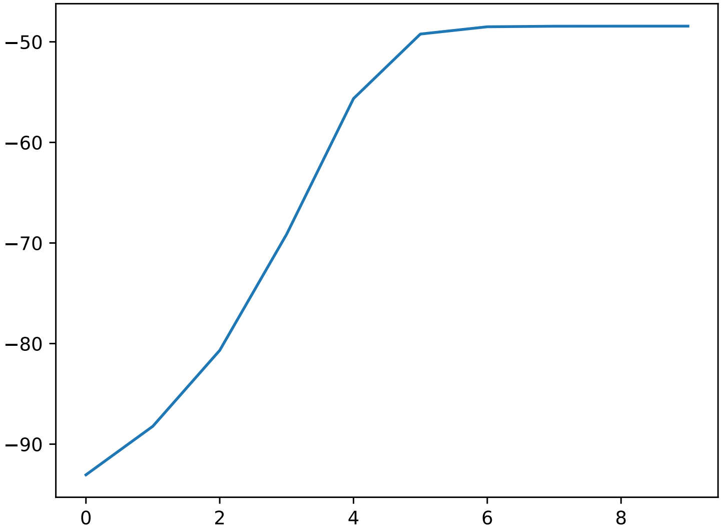
# The Expectation-Maximization Algorithm



## The Expectation-Maximization Algorithm



# Training curve



## Clustering Summary

- Why is the fit not perfect?
- Is this a Bayesian model?
- If not Bayesian, is a Bayesian formulation possible?
- Convergence is guaranteed, but not always to the global maximum
- How could we select the number of cluster centers?

To answer the last question, we could use model comparison

$$M^* = \arg\max_M p(\mathcal{D}|M)$$

## Summary of second half

You should be able to

- write down the log likelihood of a model $\log p(\mathcal{D}|\theta)$
- perform maximum likelihood $\arg\max_\theta p(\mathcal{D}|\theta)$
- perform Bayesian inference in conjugate models $p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$
- perform maximum a posteriori $\arg\max_\theta p(\mathcal{D}|\theta)p(\theta)$
- perform posterior predictive inference $p(x_*|\mathcal{D}) = \int p(x_*|\theta)p(\theta|\mathcal{D})\,\mathrm{d}\theta$
- identify normalizing constants in distributions
- perform Bayesian model comparison $\arg\max_{M\in\mathcal{M}} p(\mathcal{D}|M)$

## Summary of second half

Good luck in the exam!

## Appendix: Matrix Calculus

## Multivariate Calculus*

We often deal with derivatives of functions. The derivatives you will have encountered already will have been derivatives of *scalar functions* $f$ with respect to *scalar arguments* $x$.

$$f : \text{scalar argument} \to \text{scalar output}$$

The derivative is thus a scalar function, which we can understand from the definition of the derivative

$$\frac{\mathrm{d}f}{\mathrm{d}x} := \lim_{\delta x \to 0} \frac{f(x + \delta x) - f(x)}{\delta x} = \frac{\text{scalar}}{\text{scalar}} = \text{scalar}.$$

Note the common alternative notations

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \frac{\mathrm{d}}{\mathrm{d}x} f = f'(x)$$

## Multivariate Calculus*

Multivariate calculus concerns itself with derivatives of functions that have arguments and outputs (input and output) that are not necessarily scalars.

e.g.

$$\begin{array}{ccc} f : \text{vector} \to \text{scalar} & f : \text{matrix} \to \text{scalar} & f : \text{vector} \to \text{vector} \\ f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top\mathbf{x} & f(\mathbf{X}) = \det \mathbf{X} & f(\mathbf{x}) = \frac{\mathbf{x}}{\sqrt{\mathbf{x}^\top\mathbf{x}}} \end{array}$$

Whenever there is more than one argument, we always use a $\partial$ instead of a $\mathrm{d}$ for derivatives.

## Multivariate Calculus: vector $\to$ scalar*

In a practical setting, the way I find easiest to compute gradients is to do it element by element.

**e.g.** Compute the derivative of $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{x}$ wrt $\mathbf{x}$. We know that $f$ : vector $\to$ scalar so

$$\begin{aligned}
\frac{\partial}{\partial x_i} f &= \frac{\partial}{\partial x_i} \frac{1}{2} \mathbf{x}^\top \mathbf{x} \\
&= \frac{1}{2} \frac{\partial}{\partial x_i} \sum_{j=1}^{J} x_j x_j \\
&= \frac{1}{2} \frac{\partial}{\partial x_i} (x_1^2 + \ldots + x_i^2 + \ldots + x_J^2) \\
&= \frac{1}{2} (0 + 0 + \ldots + 2x_i + \ldots + 0) \\
&= x_i
\end{aligned}$$

We write the answer as a vector

$$\frac{\partial f}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_N} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \mathbf{x}$$

In maths and physics, $\frac{\partial}{\partial \mathbf{x}}$ is called the *gradient*, sometimes written grad or $\nabla$.

## Multivariate Calculus: vector $\to$ vector*

**e.g.** Compute the derivative of $f(\mathbf{x}) = \mathbf{A}\mathbf{x}$ wrt $\mathbf{x}$. We know that $f$ : vector $\to$ vector so

$$\begin{aligned}
\frac{\partial}{\partial x_j} f_i &= \frac{\partial}{\partial x_j} [\mathbf{A}\mathbf{x}]_i \\
&= \frac{\partial}{\partial x_j} \sum_k A_{ik} x_k \\
&= \frac{\partial}{\partial x_j} (A_{i1}x_1 + ... + A_{ij}x_j + ...) \\
&= A_{ij}
\end{aligned}$$

We write the answer as a matrix

$$\begin{aligned}
\frac{\partial f}{\partial \mathbf{x}} &= \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_1} \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_M}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_N} & \frac{\partial f_2}{\partial x_N} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} \\
&= \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{M1} \\ A_{12} & A_{22} & \cdots & A_{M2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1N} & A_{2N} & \cdots & A_{MN} \end{bmatrix} \\
&= \mathbf{A}^\top
\end{aligned}$$

## Multivariate Calculus*

Other results are

$$
\begin{aligned}
\frac{\partial}{\partial \mathbf{A}} \mathbf{x}^\top \mathbf{A} \mathbf{y} &= \mathbf{x}\mathbf{y}^\top \\
\frac{\partial}{\partial \mathbf{x}} \frac{1}{2} \mathbf{x}^\top \mathbf{A} \mathbf{x} &= \mathbf{A}\mathbf{x} \\
\frac{\partial}{\partial \mathbf{x}} \mathbf{x}^\top \mathbf{A} \mathbf{y} &= \frac{\partial}{\partial \mathbf{x}} \mathbf{y}^\top \mathbf{A}^\top \mathbf{x} = \mathbf{A}\mathbf{y} \\
\frac{\partial}{\partial \mathbf{A}} \log \det \mathbf{A} &= \mathbf{A}^{-1}
\end{aligned}
$$

**The Matrix Cookbook** In general, most of us are fairly lazy human beings and so except for a few rules, which are good to know off by heart, most of the juicy derivatives have been tabulated in a important resourse called *The Matrix Cookbook* (Wikipedia is pretty good too).

## Multivariate Calculus*

How would we fit a multivariate Gaussian to data? The likelihood term is easy[2]

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \sum_{i=1}^{N} \log \left( \left|\det\ 2\pi\boldsymbol{\Sigma}\right|^{-1/2} \exp\left\{ -\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}) \right\} \right) \\
&= -\frac{1}{2}\sum_{i=1}^{N} \log\left|\det\ 2\pi\boldsymbol{\Sigma}\right| - \frac{1}{2}\sum_{i=1}^{N} \log\exp\left\{(\mathbf{x}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})\right\} \\
&= -\frac{N}{2}\log\det\ 2\pi\boldsymbol{\Sigma} - \frac{1}{2}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})
\end{aligned}
$$

Note that $|\det\ \boldsymbol{\Sigma}| = \det\ \boldsymbol{\Sigma}$ from the definition that $\boldsymbol{\Sigma}$ is positive-definite. How do we take the gradient $\frac{\partial}{\partial\boldsymbol{\mu}}\mathcal{L}$ or $\frac{\partial}{\partial\boldsymbol{\Sigma}}\mathcal{L}$?

In general matrix derivatives are just plain nasty. You will not be asked to do these yourselves, but below is a sample of the ugliness that I am sparing you.

[2] I hope that when I have been saying 'easy' students have been able to detect the slight tone of irony in my voice.

## Multivariate Calculus: Mean*

Let's begin with the mean. First of all, we see that the first term doesn't depend on $\boldsymbol{\mu}$ at all, so we can drop it

$$
\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\mu}} \mathcal{L} &= \frac{\partial}{\partial \boldsymbol{\mu}} \left[ -\frac{N}{2} \log \det 2\pi \boldsymbol{\Sigma} - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \right] \\
&= \frac{\partial}{\partial \boldsymbol{\mu}} \left[ -\frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \right] \\
&= \frac{\partial}{\partial \boldsymbol{\mu}} \left[ -\frac{1}{2} \sum_{i=1}^{N} \cancel{\mathbf{x}_i^\top \boldsymbol{\Sigma}^{-1} \mathbf{x}_i} - \boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1} \mathbf{x}_i - \mathbf{x}_i^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} + \boldsymbol{\mu} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \right] \\
&= -\frac{1}{2} \sum_{i=1}^{N} -\boldsymbol{\Sigma}^{-1} \mathbf{x}_i - \boldsymbol{\Sigma}^{-1} \mathbf{x}_i + 2 \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \\
&= -\frac{1}{2} \boldsymbol{\Sigma}^{-1} \sum_{i=1}^{N} (\boldsymbol{\mu} - \mathbf{x}_i) = \mathbf{0} \implies \boxed{\boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i}
\end{aligned}
$$

## Multivariate Calculus: Variance*

Again it is easier to use the precision parameterization where $\mathbf{\Lambda} = \Sigma^{-1}$.

$$\frac{\partial}{\partial \mathbf{\Lambda}} \mathcal{L} = \frac{\partial}{\partial \mathbf{\Lambda}} \left[ -\frac{N}{2} \log \det 2\pi \mathbf{\Lambda}^{-1} - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^\top \mathbf{\Lambda} (\mathbf{x}_i - \boldsymbol{\mu}) \right]$$

$$= \frac{\partial}{\partial \mathbf{\Lambda}} \left[ \cancel{-\frac{N}{2} \log \det 2\pi} + \frac{N}{2} \log \det \mathbf{\Lambda} - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^\top \mathbf{\Lambda} (\mathbf{x}_i - \boldsymbol{\mu}) \right]$$

$$= \frac{N}{2} \mathbf{\Lambda}^{-1} - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^\top (\mathbf{x}_i - \boldsymbol{\mu}) = 0$$

$$\implies \boxed{\mathbf{\Sigma} = \mathbf{\Lambda}^{-1} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top}$$

## Multivariate Calculus*

The maximum likelihood mean vector and covariance matrix for the Gaussian are

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$$

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top$$

Again, do these look familiar?