Bayesian learning II

Bayes decision theory tells us that in this context we should consider the quantity $\Pr(\omega_i|\mathbf{s}, \mathbf{x})$ where the involvement of the training sequence has been made explicit.

$$\begin{aligned}
\Pr(\omega_i|\mathbf{s}, \mathbf{x}) &= \sum_{h \in \mathcal{H}} \Pr(\omega_i, h|\mathbf{s}, \mathbf{x}) \\
&= \sum_{h \in \mathcal{H}} \Pr(\omega_i|h, \mathbf{s}, \mathbf{x}) \Pr(h|\mathbf{s}, \mathbf{x}) \\
&= \sum_{h \in \mathcal{H}} \Pr(\omega_i|h, \mathbf{x}) \Pr(h|\mathbf{s}).
\end{aligned}$$

Here we have re-introduced $\mathcal{H}$ using marginalisation. In moving from line 2 to line 3 we are assuming some independence properties.

Bayesian learning II

So our classification should be

$$\omega = \underset{\omega \in \{\omega_1, \ldots, \omega_c\}}{\operatorname{argmax}} \sum_{h \in \mathcal{H}} \Pr(\omega|h, \mathbf{x}) \Pr(h|\mathbf{s})$$

If $\mathcal{H}$ is infinite the sum becomes an integral. So for example for a neural network

$$\omega = \underset{\omega \in \{\omega_1, \ldots, \omega_c\}}{\operatorname{argmax}} \int_{\mathbb{R}^W} \Pr(\omega|\mathbf{w}, \mathbf{x}) \Pr(\mathbf{w}|\mathbf{s}) \, d\mathbf{w}$$

where W is the number of weights in $\mathbf{w}$.

Bayesian learning II

Why might this make any difference? (Aside from the fact that we now know it's optimal!)

Example 1: Say $|\mathcal{H}| = 3$ and $h(\mathbf{x}) = \Pr(\mathbf{x} \text{ is in class } C_1)$ for a 2 class problem.

$$\Pr(h_1|\mathbf{s}) = 0.4$$
$$\Pr(h_2|\mathbf{s}) = \Pr(h_3|\mathbf{s}) = 0.3$$

Now, say we have an $\mathbf{x}$ for which

$$h_1(\mathbf{x}) = 1$$
$$h_2(\mathbf{x}) = h_3(\mathbf{x}) = 0$$

so h_{MAP} says that $\mathbf{x}$ is in class C_1.

Bayesian learning II

However,

$$\begin{aligned}
\Pr(\text{class } 1|\mathbf{s}, \mathbf{x}) &= 1 \times 0.4 + 0 \times 0.3 + 0 \times 0.3 \\
&= 0.4 \\
\Pr(\text{class } 2|\mathbf{s}, \mathbf{x}) &= 0 \times 0.4 + 1 \times 0.3 + 1 \times 0.3 \\
&= 0.6
\end{aligned}$$

so class C_2 is the more probable!

In this case *the Bayes optimal approach in fact leads to a different answer*.

A more in-depth example

Let's take this a step further and work through something a little more complex in detail. For a two-class classification problem, with $h(x)$ denoting $\Pr(C_1|h, x)$ and $x \in \mathbb{R}$:

Hypotheses: We have three hypotheses

$$
\begin{aligned}
h_1(x) &= \exp(-(x-1)^2) \\
h_2(x) &= \exp(-(2x-2)^2) \\
h_3(x) &= \exp(-(1/10)(x-3)^2)
\end{aligned}
$$

Prior: The prior is $\Pr(h_1) = 0.1$, $\Pr(h_2) = 0.05$ and $\Pr(h_3) = 0.85$.

A more in-depth example

We see the examples $(0.5, C_1)$, $(0.9, C_1)$, $(3.1, C_2)$ and $(3.4, C_1)$.

Likelihood: For the individual hypotheses the likelihoods are given by

$$\Pr(s|h) = h(x_1)h(x_2)[1 - h(x_3)]h(x_4)$$

Which in this case tells us

$$
\begin{aligned}
\Pr(s|h_1) &= 0.0024001365 \\
\Pr(s|h_2) &= 0.0031069836 \\
\Pr(s|h_3) &= 0.0003387476
\end{aligned}
$$

Posterior: Multiplying by the priors and normalising gives

$$
\begin{aligned}
\Pr(h_1|s) &= 0.3512575000 \\
\Pr(h_2|s) &= 0.2273519164 \\
\Pr(h_3|s) &= 0.4213905836
\end{aligned}
$$

A more in-depth example

Now let's classify the point $x' = 2.5$.

We need

$$
\begin{aligned}
\Pr(C_1|s, x') &= \Pr(C_1|h_1)\Pr(h_1|s) + \Pr(C_1|h_2)\Pr(h_2|s) + \Pr(C_1|h_3)\Pr(h_3|s) \\
&= 0.6250705317
\end{aligned}
$$

So: it's most likely to be in class C_1, but not with great certainty.

The Bayesian approach to neural networks

Let's now see how this can be applied to *neural networks*. We have:

- A neural network computing a function $f(\mathbf{w}; \mathbf{x})$.
- A training sequence $\mathbf{s} = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$, split into

  $$\mathbf{y} = \begin{pmatrix} y_1 & y_2 & \cdots & y_m \end{pmatrix}$$

  and

  $$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \end{pmatrix}$$

The *prior distribution* $p(\mathbf{w})$ is now on the weight vectors, and Bayes' theorem tells us that

$$p(\mathbf{w}|\mathbf{s}) = p(\mathbf{w}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w}|\mathbf{X})}{p(\mathbf{y}|\mathbf{X})}$$

Nothing new so far...

The Bayesian approach to neural networks

As usual, we don't consider uncertainty in $\mathbf{x}$ and so $\mathbf{X}$ will be omitted. Consequently

$$p(\mathbf{w}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{y})}$$

where

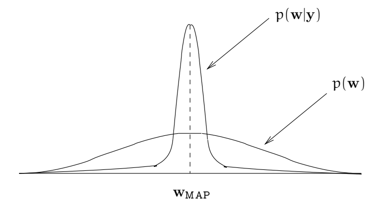
$$p(\mathbf{y}) = \int_{\mathbb{R}^W} p(\mathbf{y}|\mathbf{w})p(\mathbf{w})d\mathbf{w}$$

$p(\mathbf{y}|\mathbf{w})$ is a model of the noise corrupting the labels and as previously is the *likelihood function*.

The Bayesian approach to neural networks

$p(\mathbf{w})$ is typically a *broad distribution* to reflect the fact that in the absence of any data we have little idea of what $\mathbf{w}$ might be.

When we see some data the above equation tells us how to obtain $p(\mathbf{w}|\mathbf{y})$. This will typically be *more localised*.



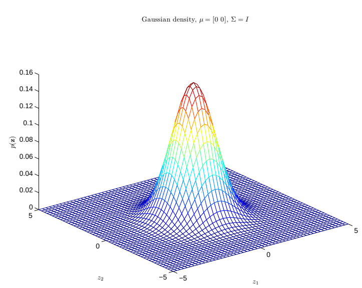
To put this into practice we need expressions for $p(\mathbf{w})$ and $p(\mathbf{y}|\mathbf{w})$.

Reminder: the general Gaussian density

Reminder: we're going to be making a lot of use of the general *Gaussian density* $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ in d dimensions

$$p(\mathbf{z}) = (2\pi)^{-d/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\left[-\frac{1}{2}\left((\mathbf{z}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\right)\right]$$

where $\boldsymbol{\mu}$ is the *mean vector* and $\boldsymbol{\Sigma}$ is the *covariance matrix*.



The Gaussian prior

A common choice for $p(\mathbf{w})$ is the *Gaussian prior* with zero mean and

$$\Sigma = \sigma^2\mathbf{I}$$

so

$$p(\mathbf{w}) = (2\pi)^{-W/2}\sigma^{-W} \exp\left[-\frac{\mathbf{w}^T\mathbf{w}}{2\sigma^2}\right]$$

Note that σ controls the distribution of other parameters.

- Such parameters are called *hyperparameters*.
- Assume for now that they are both fixed and known.

Hyperparameters can be learnt using s through the application of more advanced techniques.

The Bayesian approach to neural networks

Physicists like to express quantities such as $p(\mathbf{w})$ in terms of a measure of *"energy"*. The expression is therefore usually re-written as

$$\boxed{p(\mathbf{w}) = \frac{1}{Z_W(\alpha)} \exp\left(-\frac{\alpha}{2}||\mathbf{w}||^2\right)}$$

where

$$E_W(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$$
$$Z_W(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{d/2}$$
$$\alpha = \frac{1}{\sigma^2}$$

This is simply a re-arranged version of the more usual equation.

The Gaussian noise model for regression

We've already seen that for a regression problem with zero mean Gaussian noise having variance σ_n^2

$$y_i = f(\mathbf{x}_i) + \epsilon_i$$
$$p(\epsilon_i) = \frac{1}{\sqrt{2\pi\sigma_n^2}} \exp\left(-\frac{\epsilon_i^2}{2\sigma_n^2}\right)$$

where f corresponds to some unknown network, the likelihood function is

$$p(\mathbf{y}|\mathbf{w}) = \frac{1}{(2\pi\sigma_n^2)^{m/2}} \exp\left(-\frac{1}{2\sigma_n^2}\sum_{i=1}^{m}(y_i - f(\mathbf{w};\mathbf{x}_i))^2\right)$$

Note that there are now two variances: σ^2 *for the prior and* σ_n^2 *for the noise.*

The Bayesian approach to neural networks

This expression can also be rewritten in physicist-friendly form

$$\boxed{p(\mathbf{y}|\mathbf{w}) = \frac{1}{Z_y(\beta)} \exp\left(-\beta E_y(\mathbf{w})\right)}$$

where

$$\beta = \frac{1}{\sigma_n^2}$$
$$Z_y(\beta) = \left(\frac{2\pi}{\beta}\right)^{m/2}$$
$$E_y(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}(y_i - f(\mathbf{w};\mathbf{x}_i))^2$$

Here, β is a second *hyperparameter*. Again, we assume it is fixed and known, although it can be learnt using s using more advanced techniques.

The Bayesian approach to neural networks

Combining the two boxed equations gives

$$\boxed{p(\mathbf{w}|\mathbf{y}) = \frac{1}{Z_S(\alpha,\beta)} \exp(-S(\mathbf{w}))}$$

where

$$\boxed{S(\mathbf{w}) = \alpha E_W(\mathbf{w}) + \beta E_y(\mathbf{w})}$$

The quantity

$$Z_S(\alpha,\beta) = \int_{\mathbb{R}^W} \exp(-S(\mathbf{w}))d\mathbf{w}$$

normalises the density. Recall that this is called the *evidence*.

Example I: gradient descent revisited...

To find h_{MAP} (in this scenario by finding $\mathbf{w}_{MAP}$) we therefore maximise

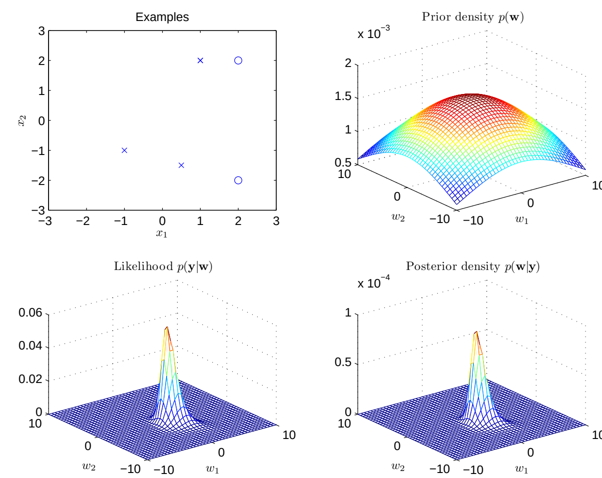
$$p(\mathbf{w}|\mathbf{y}) = \frac{1}{Z_S(\alpha, \beta)} \exp(-(\alpha E_W(\mathbf{w}) + \beta E_{\mathbf{y}}(\mathbf{w})))$$

or equivalently find

$$\mathbf{w}_{MAP} = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{\alpha}{2}||\mathbf{w}||^2 + \frac{\beta}{2} \sum_{i=1}^{m} (y_i - f(\mathbf{w}; \mathbf{x}_i))^2$$

This algorithm has also been used a lot in the neural network literature and is called the *weight decay* technique.

Example II: two-class classification in two dimensions



The Bayesian approach to neural networks

What happens as the number m of examples increases?

- The first term *corresponding to the prior* remains fixed.
- The second term *corresponding to the likelihood* increases.

So for small training sequences the prior dominates, but for large ones h_{ML} is a good approximation to h_{MAP}.

The Bayesian approach to neural networks

Where have we got to...? We have obtained

$$p(\mathbf{w}|\mathbf{y}) = \frac{1}{Z_S(\alpha, \beta)} \exp(-(\alpha E_W(\mathbf{w}) + \beta E_{\mathbf{y}}(\mathbf{w})))$$

$$Z_S(\alpha, \beta) = \int_{\mathbb{R}^W} \exp(-(\alpha E_W(\mathbf{w}) + \beta E_{\mathbf{y}}(\mathbf{w})))d\mathbf{w}$$

Translating the expression for the *Bayes optimal* solution given on the first slide of this handout into the current scenario, we need to compute

$$p(Y|\mathbf{y}, \mathbf{x}) = \int_{\mathbb{R}^W} p(y|\mathbf{w}, \mathbf{x}) p(\mathbf{w}|\mathbf{y})\, d\mathbf{w}$$

Easy huh? *Unfortunately not...*

The Bayesian approach to neural networks

In order to make further progress it's therefore necessary to perform integrals of the general form

$$\int_{\mathbb{R}^W} F(\mathbf{w})p(\mathbf{w}|\mathbf{y})d\mathbf{w}$$

for various functions F and this is generally not possible.

There are two ways to get around this:

1. We can use an *approximate form* for $p(\mathbf{w}|\mathbf{y})$.
2. We can use *Monte Carlo* methods.

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

The first approach introduces a *Gaussian approximation* to $p(\mathbf{w}|\mathbf{y})$ by using a *Taylor expansion* of

$$S(\mathbf{w}) = \alpha E_W(\mathbf{w}) + \beta E_y(\mathbf{w})$$

at $\mathbf{w}_{\text{MAP}}$.

This allows us to use a *standard integral*.

The result will be *approximate* but we hope it's good!

Let's recall how Taylor series work...

Reminder: Taylor expansion

In one dimension the Taylor expansion about a point $x_0 \in \mathbb{R}$ for a function $f : \mathbb{R} \to \mathbb{R}$ is

$$f(x) \approx f(x_0) + \frac{1}{1!}(x - x_0)f'(x_0) + \frac{1}{2!}(x - x_0)^2 f''(x_0) + \cdots + \frac{1}{k!}(x - x_0)^k f^k(x_0)$$

What does this look like for the kinds of function we're interested in? We can try to approximate

$$\exp(-f(x))$$

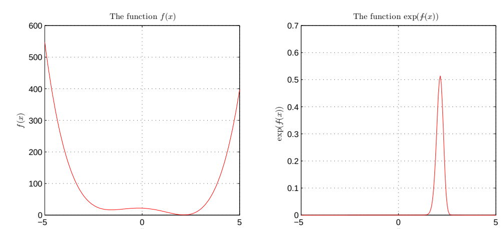
where

$$f(x) = x^4 - \frac{1}{2}x^3 - 7x^2 - \frac{5}{2}x + 22$$

This has a form similar to $S(\mathbf{w})$, but in one dimension.

Reminder: Taylor expansion

The functions of interest look like this:



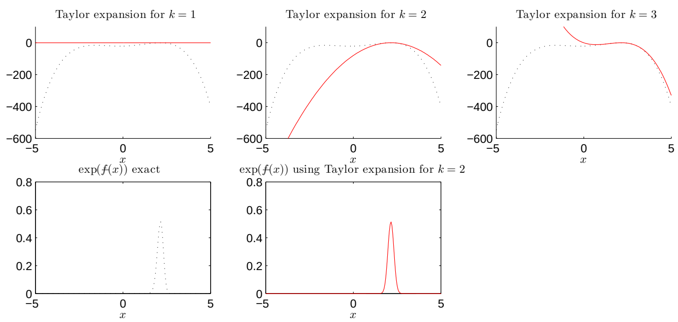
By replacing $-f(x)$ with its Taylor expansion about its maximum, which is at

$$x_{\max} = 2.1437$$

we can see what the approximation to $\exp(-f(x))$ looks like. Note that the exp hugely emphasizes peaks.

Reminder: Taylor expansion

Here are the approximations for $k = 1$, $k = 2$ and $k = 3$.



The use of $k = 2$ looks promising...

Reminder: Taylor expansion

In *multiple dimensions* the Taylor expansion for $k = 2$ is

$$f(\mathbf{x}) \approx f(\mathbf{x}_0) + \frac{1}{1!}(\mathbf{x} - \mathbf{x}_0)^T \left.\nabla f(\mathbf{x})\right|_{\mathbf{x}_0} + \frac{1}{2!}(\mathbf{x} - \mathbf{x}_0)^T \left.\nabla^2 f(\mathbf{x}_0)\right|_{\mathbf{x}_0} (\mathbf{x} - \mathbf{x}_0)$$

where ∇ denotes *gradient*

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} & \frac{\partial f(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f(\mathbf{x})}{\partial x_n} \end{pmatrix}$$

and $\nabla^2 f(\mathbf{x})$ is the matrix with elements

$$M_{ij} = \frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}$$

(Although this looks complicated, it's just the obvious extension of the 1-dimensional case.)

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

Applying this to $S(\mathbf{w})$ and expanding around $\mathbf{w}_{MAP}$

$$S(\mathbf{w}) \approx S(\mathbf{w}_{MAP}) + (\mathbf{w} - \mathbf{w}_{MAP})^T \left.\nabla S(\mathbf{w})\right|_{\mathbf{w}_{MAP}} + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{MAP})^T \mathbf{A} (\mathbf{w} - \mathbf{w}_{MAP})$$

notice the following:

- As $\mathbf{w}_{MAP}$ *minimizes* the function the first derivatives are zero and the corresponding term in the Taylor expansion *disappears*.
- The quantity $\mathbf{A} = \left.\nabla\nabla S(\mathbf{w})\right|_{\mathbf{w}_{MAP}}$ can be simplified.

This is because

$$\begin{aligned} \mathbf{A} &= \left.\nabla\nabla(\alpha E_W(\mathbf{w}) + \beta E_y(\mathbf{w}))\right|_{\mathbf{w}_{MAP}} \\ &= \alpha \mathbf{I} + \beta \nabla\nabla E_y(\mathbf{w}_{MAP}) \end{aligned}$$

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

Defining

$$\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}_{MAP}$$

we now have

$$S(\mathbf{w}) \approx S(\mathbf{w}_{MAP}) + \frac{1}{2}\Delta\mathbf{w}^T \mathbf{A} \Delta\mathbf{w}$$

The vector $\mathbf{w}_{MAP}$ can be obtained using any standard optimization method (such as *backpropagation*).

The quantity $\nabla\nabla E_y(\mathbf{w})$ can be evaluated using an *extended form of backpropagation*.

A useful integral

Dropping *for this slide only* the special meanings usually given to vectors $\mathbf{x}$ and $\mathbf{y}$, here is a useful standard integral:

If $\mathbf{A} \in \mathbb{R}^{n \times n}$ is symmetric then for $\mathbf{b} \in \mathbb{R}^n$ and $c \in \mathbb{R}$

$$\int_{\mathbb{R}^n} \exp\left(-\frac{1}{2}\left(\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{b} + c\right)\right) d\mathbf{x} = (2\pi)^{n/2}|\mathbf{A}|^{-1/2} \exp\left(-\frac{1}{2}\left(c - \frac{\mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}}{4}\right)\right)$$

At the beginning of the course, two exercises were set involving the evaluation of this integral.

To make this easy to refer to, let's call it the *BIG INTEGRAL*.

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

We now have

$$p(\mathbf{w}|\mathbf{y}) \approx \frac{1}{Z(\alpha,\beta)} \exp\left(-S(\mathbf{w}_{MAP}) - \frac{1}{2}\Delta\mathbf{w}^T\mathbf{A}\Delta\mathbf{w}\right)$$

where $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}_{MAP}$ and using the *BIG INTEGRAL*

$$Z(\alpha,\beta) = (2\pi)^{W/2}|\mathbf{A}|^{-1/2}\exp(-S(\mathbf{w}_{MAP}))$$

Our earlier discussion tells us that given a new input $\mathbf{x}$ we should calculate

$$p(Y|\mathbf{y},\mathbf{x}) = \int_{\mathbb{R}^W} p(y|\mathbf{w},\mathbf{x})p(\mathbf{w}|\mathbf{y})d\mathbf{w}$$

$p(y|\mathbf{w},\mathbf{x})$ is just the *likelihood* so...

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

The likelihood we're using is

$$p(y|\mathbf{w},\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(y - f(\mathbf{w};\mathbf{x}))^2}{2\sigma^2}\right) \propto \exp\left(-\frac{\beta}{2}(y - f(\mathbf{w};\mathbf{x}))^2\right)$$

and plugging it into the integral gives

$$p(y|\mathbf{x},\mathbf{y}) \propto \int_{\mathbb{R}^W} \exp\left(-\frac{\beta}{2}(y - f(\mathbf{w};\mathbf{x}))^2\right)\exp\left(-\frac{1}{2}\Delta\mathbf{w}^T\mathbf{A}\Delta\mathbf{w}\right) d\mathbf{w}$$

which *has no solution!*

We need *another approximation...*

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

If we *assume* that $p(\mathbf{w}|\mathbf{y})$ is narrow (this depends on $\mathbf{A}$) then we can introduce a *linear approximation* of $f(\mathbf{w};\mathbf{x})$ at $\mathbf{w}_{MAP}$:

$$f(\mathbf{w};\mathbf{x}) \approx f(\mathbf{w}_{MAP};\mathbf{x}) + \mathbf{g}^T\Delta\mathbf{w}$$

where $\mathbf{g} = \nabla f(\mathbf{w};\mathbf{x})|_{\mathbf{w}_{MAP}}$.

By linear approximation we just mean the Taylor expansion for $k = 1$.

This leads to

$$p(Y|\mathbf{y},\mathbf{x}) \propto \int_{\mathbb{R}^W} \exp\left(-\frac{\beta}{2}\left(y - f(\mathbf{w}_{MAP};\mathbf{x}) - \mathbf{g}^T\Delta\mathbf{w}\right)^2 - \frac{1}{2}\Delta\mathbf{w}^T\mathbf{A}\Delta\mathbf{w}\right) d\mathbf{w}$$

and this integral can be evaluated using the *BIG INTEGRAL* to give *THE ANSWER...*

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

Finally

$$p(Y|\mathbf{y},\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(y - f(\mathbf{w}_{\mathrm{MAP}};\mathbf{x}))^2}{2\sigma_y^2}\right)$$

where

$$\sigma_y^2 = \frac{1}{\beta} + \mathbf{g}^T \mathbf{A}^{-1}\mathbf{g}.$$

Hooray! But what does it mean?

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$

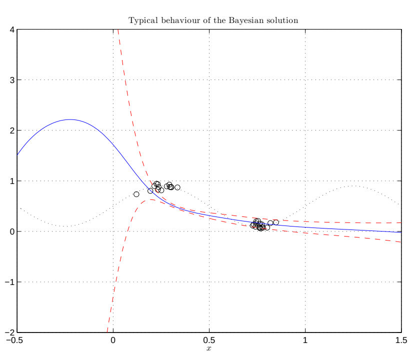
This is a *Gaussian density*, so we can now see that $p(Y|\mathbf{y},\mathbf{x})$ *peaks* at $f(\mathbf{w}_{\mathrm{MAP}};\mathbf{x})$. That is, the *MAP solution*.

The *variance* σ_y^2 can be interpreted as a measure of *certainty*.

- The first term of σ_y^2 is $1/\beta$ and corresponds to the noise.
- The second term of σ_y^2 is $\mathbf{g}^T\mathbf{A}^{-1}\mathbf{g}$ and corresponds to the width of $p(\mathbf{w}|\mathbf{y})$.

Or *interpreted graphically...*

Method 1: approximation to $p(\mathbf{w}|\mathbf{y})$



Method II: Markov chain Monte Carlo (MCMC) methods

The second solution to the problem of performing integrals

$$I = \int F(\mathbf{w})p(\mathbf{w}|\mathbf{y})d\mathbf{w}$$

is to use *Monte Carlo* methods. The basic approach is to make the approximation

$$I \approx \frac{1}{N}\sum_{i=1}^{N} F(\mathbf{w}_i)$$

where the $\mathbf{w}_i$ have distribution $p(\mathbf{w}|\mathbf{y})$. Unfortunately, generating $\mathbf{w}_i$ with a *given distribution* can be non-trivial.

MCMC methods

A simple technique is to introduce a random walk, so

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \boldsymbol{\epsilon}$$

where $\boldsymbol{\epsilon}$ is zero mean spherical Gaussian and has small variance. Obviously the sequence $\mathbf{w}_i$ does not have the required distribution. However we can use the *Metropolis algorithm*, which does *not* accept all the steps in the random walk:

1. If $p(\mathbf{w}_{i+1}|\mathbf{y}) > p(\mathbf{w}_i|\mathbf{y})$ then accept the step.
2. Else accept the step with probability $\frac{p(\mathbf{w}_{i+1}|\mathbf{y})}{p(\mathbf{w}_i|\mathbf{y})}$.

MCMC methods

In practice, the Metropolis algorithm has several shortcomings, and a great deal of research exists on improved methods, see:

R. Neal, "Probabilistic inference using Markov chain Monte Carlo methods," University of Toronto, Department of Computer Science Technical Report CRG-TR-93-1, 1993.