LECTURE 6,7: GENERALISATION REVISITED

In previous chapters we have seen how *overfitting* to some training data set will lead to poorer performance on unseen (new) data. In this chapter we explore a little more some of these concepts, starting with notions of complexity, regularisation and the *bias / variance tradeoff* and finish with looking at committees and the extremely powerful technique of Bayesian learning.

Complexity and Ockham's razor

More complex systems (ones with more free parameters) are needed to model more complex functions and *vice versa*. An over-complex systems will begin to fit noise and an under-complex one will not be able to model the function. How do we get the complexity right?

This issue was addressed from a philosophical perspective in the late 13th century by William of Ockham (sometimes spelled Occam). Ockham suggested that one should not

> ... *multiply explanations unnecessarily...*

In other words, we should seek the *simplest model which fits the data*. We need to specify what we mean by 'simplest' and 'fitting' though. One way was proposed by Rissannen in 1978, and is known as Minimum Description Length (MDL). Risannen proposed (from information-theoretic arguments) a penalisation of an error, or goodness-of-fit, function of the form

$$\langle E_{MDL} \rangle = \langle E_{data} \rangle + \frac{1}{2} N_p \frac{\log N}{N}$$

where N are the number of data and N_p the number of parameters in the model.

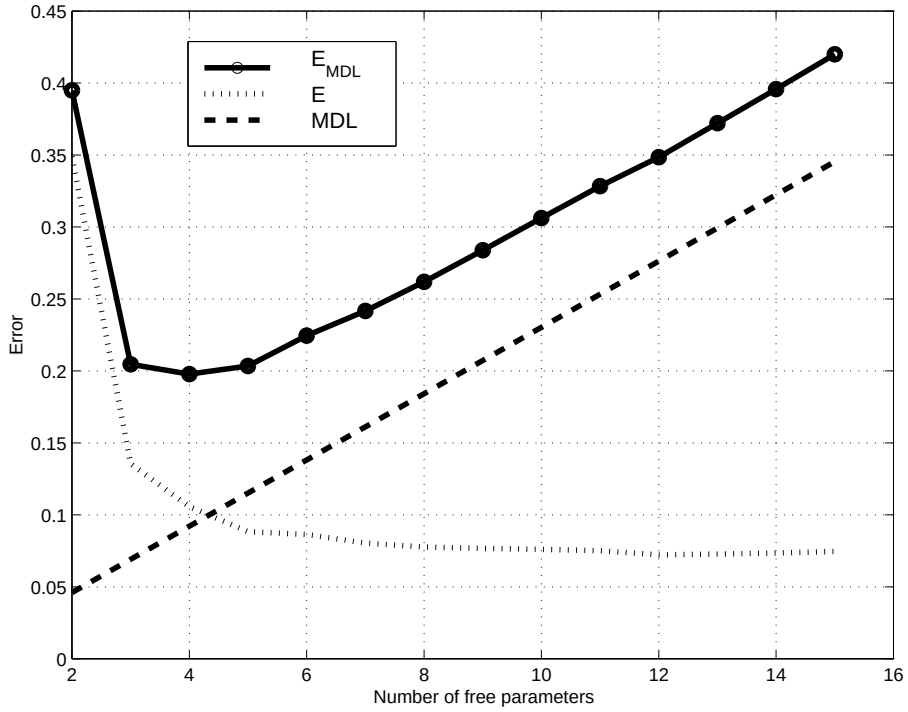

Figure 1: MDL on the noisy sine problem

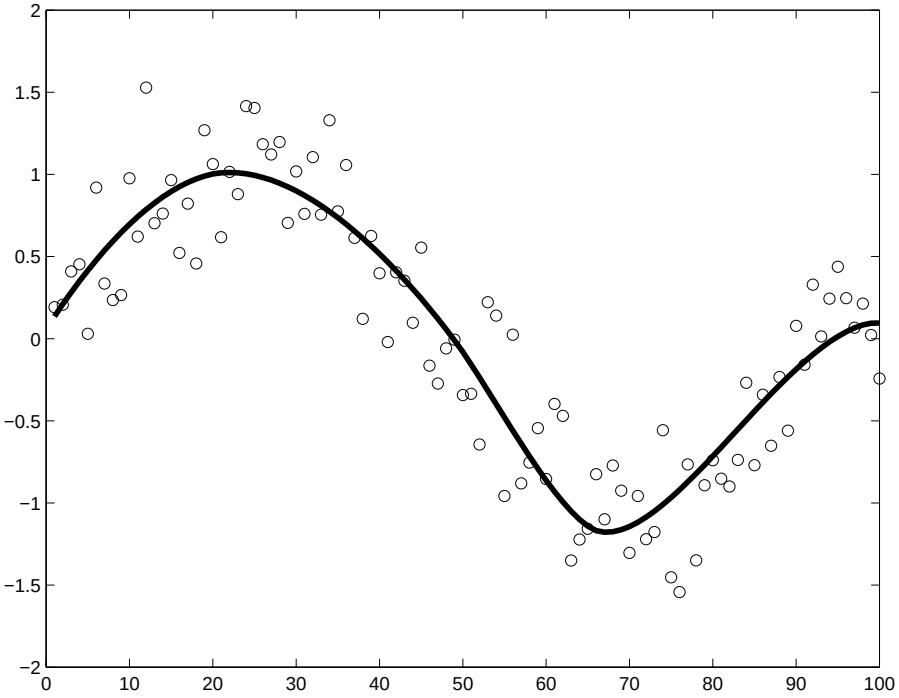

Figure 2: 4-component MDL solution.

This seems to work nicely. The only problem is this:

- the number of data is not necessarily the number of pieces of independent information and
- the number of parameters does not reflect the complexity of the model if we have regularisation (remember the smooth plots from complex models with Gaussians with large widths). What we really want is the *effective* number of parameters.

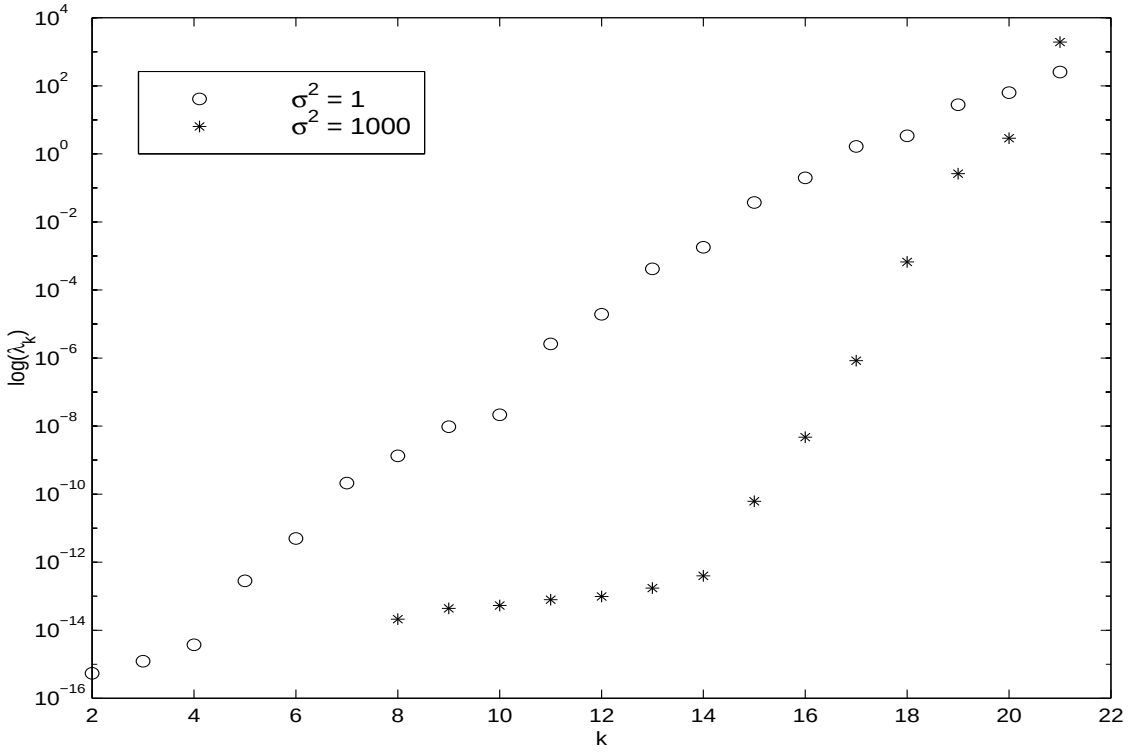

Figure 3: Eigenspectra of Hessian for RBF with Gaussians of variable width.

Fig. 3 shows the eigenspectra of the Hessian matrix for Gaussian kernel RBF systems with 20 kernels. The variances of the Gaussians are $\sigma^2 = 1, 1000$. Look at the difference in the number of 'well-determined' components to the Hessian. This means that there are big differences in the number of parameters which actually effect the error surface. It has been suggested that the N_p term in the MDL Eqn. be replaced with this estimated number of 'well-determined' components.

The bias / variance tradeoff

Remember from previous chapters that the best model output is the one which equals the conditional average of the targets given the input,

$$y(x) = \langle t \mid x \rangle$$

We can (for regression) measure the deviation from the ideal situation via the error,

$$\mathcal{E}_D[(y(x) - \langle t \mid x \rangle)^2]$$

where the expectation is over the data set, D, which the model is conditioned on. Expanding and completing the square gives,

$$\mathcal{E}_D[(y(x) - \langle t \mid x \rangle)^2] = \underbrace{(\mathcal{E}_D[y(x)] - \langle t \mid x \rangle)^2}_{(\text{bias})^2} + \underbrace{\mathcal{E}_D[(y(x) - \mathcal{E}_D[y(x)])^2]}_{\text{variance}}$$

 Have a think what these terms represent.

Regularisation

As we saw in chapter 4, the role of regularisation is to penalise solutions which have high changes in curvature - i.e. are not smooth. This was approached via the modified error function,

$$E_{reg} = E_{data} + \alpha R$$

where α is the regularisation parameter and R a measure of the un-smoothness of the solution (assuming $\alpha > 0$). We saw that R for a RBF system of Gaussian kernels depended on two things we can change,

- The width of the kernel representation - larger widths give more overlap between kernels and hence more correlation, so the effective number of free parameters goes down.
- The value of $|\mathbf{w}|$.

Define a regularisation term as:

$$R = \frac{1}{2} \sum_i w_i^2$$

The total error functional is hence

$$E_{reg} = E_{data} + \frac{\alpha}{2} \sum_i w_i^2$$

the error function has a gradient, w.r.t. parameter w_i of:

$$\nabla E_{reg} = \nabla E_{data} + \alpha w_i$$

if we adjust the parameters according to some scheme such as

$$\Delta w_i = -\eta \nabla E_{reg} = -\eta \nabla E_{data} - \eta \alpha w_i$$

If the data contributes little to the adjustment of w_i (i.e. w_i contributes little to the error function, so is irrelevant) then w_i decays exponentially towards zero. For this reason this regularisation scheme is referred to as *weight decay*.

Writing the error in a quadratic form, i.e.

$$E(\mathbf{w}) \propto \frac{1}{2}\mathbf{w}^{\mathsf{T}}\mathbf{H}\mathbf{w}$$

gives a minimum error gives which solves

$$\nabla E = \mathbf{H}\mathbf{w}^* = 0$$

With a regulariser this becomes,

$$\nabla E_{reg} = \mathbf{H}\mathbf{w}^+ + \alpha \mathbf{w}^+ = 0$$

We can expand both $\mathbf{w}^*$ and $\mathbf{w}^+$ in terms of the eigenvectors of $\mathbf{H}$ hence,

$$\mathbf{w}^* = \sum_i w_i^* \mathbf{u}_i \qquad \text{and} \qquad \mathbf{w}^+ = \sum_i w_i^+ \mathbf{u}_i$$

Equating the two gradient equations gives

$$\sum_i w_i^* \lambda_i \mathbf{u}_i = \sum_i w_i^+ \mathbf{u}_i (\lambda_i + \alpha)$$

where $\{\lambda_i\}$ are the eigenvalues of $\mathbf{H}$. The solution to this is (remembering that all $\lambda_i \geq 0$ as $\mathbf{H}$ is positive semi-definite):

$$w_i^+ = \frac{\lambda_i}{\lambda_i + \alpha} w_i^*$$

 Have a think what this means.

One way we can interpret this is to see that, if λ_i is small compared to α (weight is insignificant) then the regularised weight will end up at near zero - so it will not contribute to the model. This has two nice effects

- We can use the expression

$$N_p^{eff} = \sum_i \frac{\lambda_i}{\lambda_i + \alpha}$$

  as representing the *effective number of parameters in the model*, and

- we can speed up computation by removing all those parameters whose contribution to the Hessian curvature (the eigenvalues) is small compared α.

The first case leads us to consider an alternative to the validation-set approach looked at in chapter 1. Instead of varying the number of parameters we can vary the constant α. We will come back to this form of regularisation in the context of Bayesian

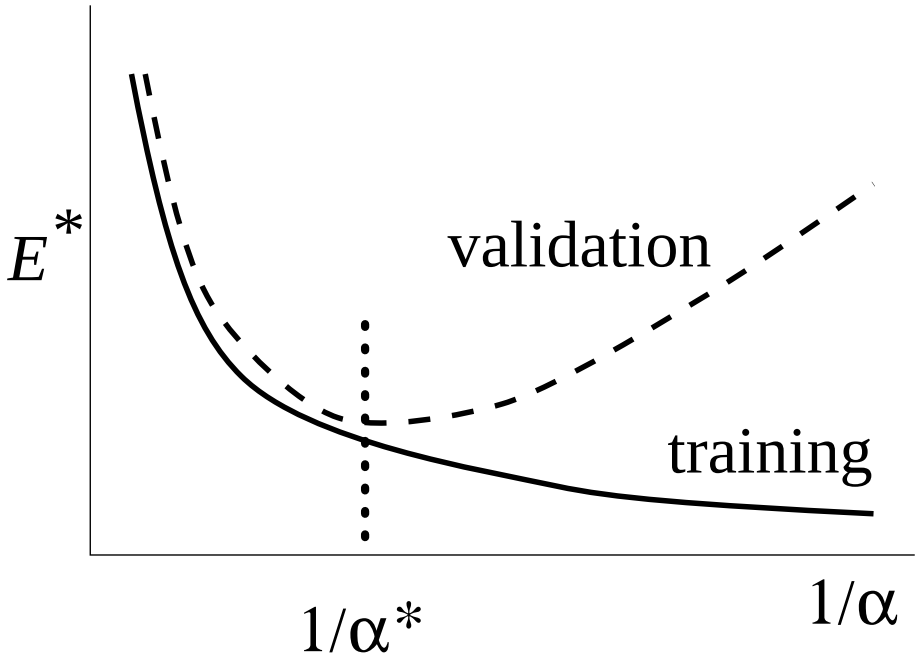

Figure 4: Training & validation error against (reciprocal) regularisation constant.

learning later in this chapter.

The second case, the removal of parameters or connections which contribute little to the solution, is referred to as *pruning*.

Committees

If we go back to the bias / variance components of error (earlier in this chapter) we see that if we have several data sets and combine the results of systems conditional on each then the *bias* will tend to reduce (as for over-fitted systems the learned 'noise' on each instance of the data set will tend to be different) at the expense of the variance (which will tend to increase). The former of these terms may be larger than the latter though. We would expect to get an improvement in results if we combine the results of several analysers in a *committee.*

Consider the classification case first. Let the error on the n-th datum be the cross-entropy error which for the i-th classifier is:

$$E_i = -t \ln(y_i/t)$$

Now let us form a convex combination of the classifiers, mixed such that

$$y = \sum_i \gamma_i y_i$$

where $\gamma_i \geq 0$ and $\sum_i \gamma_i = 1$. We write the resultant error as

$$E_{comm} = -t \ln(y/t) = t \ln t - t \ln \left(\sum_i \gamma_i y_i \right)$$

Now let's consider the mixture (weighted average) of errors from each of the classifiers,

$$E_{av} = \sum_i \gamma_i E_i = \sum_i \gamma_i (t \ln t - t \ln y_i)$$

as the t terms are independent of i so,

$$E_{av} = t \ln t - t \sum_i \gamma_i \ln y_i$$

Using Jensen's inequality ($\ln \sum_i \gamma_i x_i \geq \sum_i \gamma_i \ln x_i$) so

$$\ln \left(\sum_i \gamma_i y_i \right) \geq \sum_i \gamma_i \ln y_i$$

and hence (as E is negative in these quantities)

$$E_{comm} \leq E_{av}$$

This means that a combination of classifier outputs will give *always* a lower error than the (weighted) average of errors for each classifier. A remarkable result! The optimal choice of γ_i requires the estimation of the between classifier covariance, but just a simple averaging ($\gamma_i = 1/C$) is often used. Again, we will (briefly) revisit the issue of committees later.

What about the regression case? We appeal to the simple fact that, *if the errors between a set of regressors are uncorrelated* then any weighted averaging (convex combination) will give a smaller error than the average error of the individuals. If σ_i^2 is the error variance of the i-th regressor then a combination of C gives (assuming independence) the resultant variance of the committee is, taking the simple case of equal weighting and letting $e_i = y_i - t$,

$$\sigma_{comm}^2 = \left(\frac{1}{C} \sum_i e_i \right)^2$$

which, if errors are independent with zero mean (i.e. there is no bias)

$$\sigma_{comm}^2 = \frac{1}{C^2} \sum_i \sigma_i^2$$

and the average error variance of the members is

$$\sigma_{av}^2 = \frac{1}{C}\sum_i \sigma_i^2$$

Clearly, the errors will not really be independent. In the worst case we have C redundant error correlations, and need to multiple the committee variance by C. This is still only equal to the average case. Hence

$$E_{comm} \leq E_{av}$$

So we have proved that using committees gives, on average, better results, *irrespective of whether they are formed from systems trained on different data sets*. Fig. 5 shows a simple example of committee benefits in the regression case.

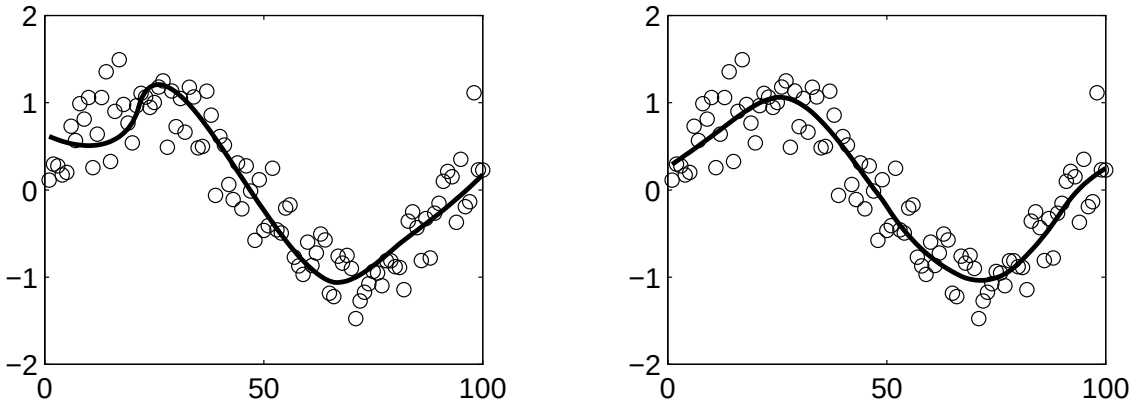

Figure 5: [left] Output of single 5-spline regressor. [right] Output of committee of 10 uniformly weighted 5-spline regressors.

Bayesian learning

We have already looked at a simple example of Bayesian learning in previous chapters. It is worth considering what advantages Bayesian learning has:

- On average it gives better results.
- Regularisation arises naturally and has a simple interpretation.
- Committees have a simple interpretation and arise naturally.
- As distributions, not single values, are dealt with so error bars are naturally obtained in regression and uncertainty dealt with in a principled framework.
- Overfitting is avoided even when training without a validation set, so all available data can be used for training.

You may wonder why anyone bothers not to use the Bayesian approach.

> *There are two ways of solving a data analysis problem: the Bayesian way and the wrong way.* [Bishop]
>
> *There are many excuses for not using a Bayesian approach. The only true one is incompetence.* [Skilling]

Bishop gives a nice picture of Bayesian learning. We consider a data set D and a set of models M_i of increasing complexity (number of free parameters). As the complexity of the model increases so the range of data sets which cane be modelled increases. Consider Bayes theorem in the form

$$P(M_i \mid D) = \frac{P(D \mid M_i)P(M_i)}{P(D)}$$

we have no reason, ahead of time, to favour one model more than another so both the model prior and the evidence are constant across models. So ranking models with $P(M_i \mid D)$ becomes equivalent to ranking using the *model evidence on* D, i.e. $P(D \mid M_i)$. As this is a density so it integrates to unity, so more complex models, with wider coverage on the set $\mathcal{D}$ of data, will model *any one* D less well then a lower complexity model tuned to D. Fig. 6 shows this idea graphically. This means that over-complex models are naturally penalised. Bayesian methodology forms a nice trade-off between fitting the model to the data and minimising the complexity penalty (often called the *Ockham factor*).

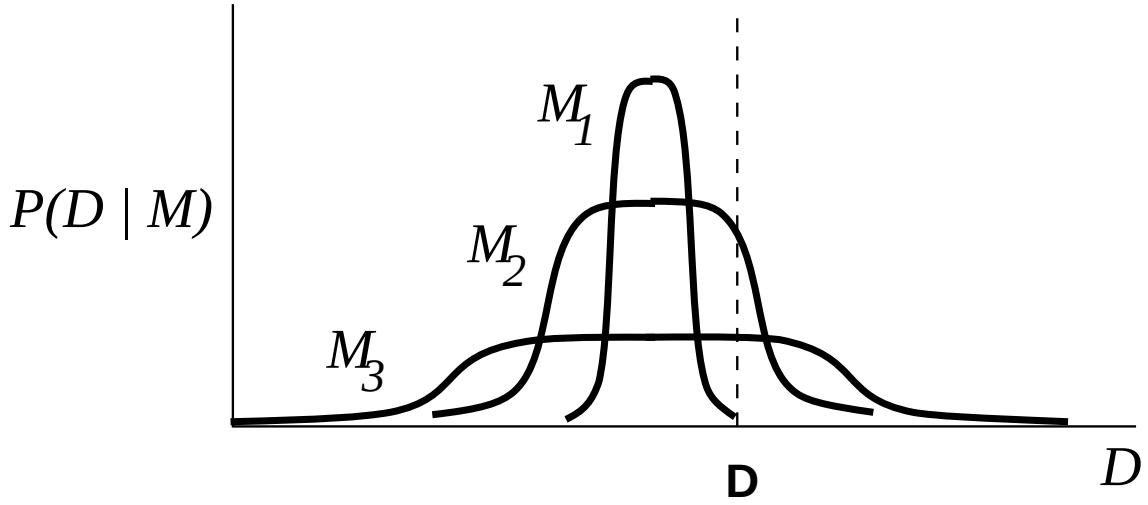

Figure 6: Increasing complexity models, $M_{1,2,3}$ and data set D.

Bayes in action

Consider the kind of system we have been looking at for a while, where an output is modelled as

$$y = g(a)$$

where a is the *latent* variable and $g(.)$ is e.g. a sigmoid for classification and linear for regression. The latent variables are weighted combinations of some other variables. In the case of linear discriminants the inputs and for more flexible models (such as neural networks) a non-linear mapping of the inputs.

Consider now the density over the latent variable $P(a \mid D)$. An analysis model will provide us with $P(a \mid D, \mathbf{v})$ where $\mathbf{v}$ is all the variables which parameterise the model (the weights etc.). The Bayesian approach is to integrate out these nuisance variables using the marginal integral

$$P(a \mid D) = \int P(a \mid D, \mathbf{v}) P(\mathbf{v} \mid D) d\mathbf{v}$$

For now we will consider the regression case and come back to classification later. Hence we use a linear $g(.)$ and the least-squares error, which is the same in the latent space. This means that the density over a in the latent space is Gaussian. Denote the variables of the model $\mathbf{v} = \{\mathbf{w}, \alpha, \beta\}$ where $\mathbf{w}$ are the weights and α, β are parameters which will be defined shortly.

First we write the density of $P(a \mid D, \mathbf{w})$ as a Gaussian (as the error is least squares) giving:

$$P(a \mid D, \mathbf{w}, \beta) \propto \exp\left\{-\frac{\beta}{2}(a-t)^2\right\}$$

where β is the precision (inverse variance) term.

It is worth noting that $1/\beta$ corresponds to the expected (believed) variance of errors we make. This is normally taken to be noise on the targets.

Now, using Bayes theorem, we write

$$P(\mathbf{w}|D) \propto P(D \mid \mathbf{w})P(\mathbf{w})$$

where the last term is the *prior* distribution over the weights. What form could this take? An appropriate prior is just a Gaussian, centred on zero (weights can be $\pm$) i.e.

$$P(\mathbf{w}) \propto \exp\left\{-\frac{\alpha}{2}\mathbf{w}^2\right\}$$

so the other mystery parameter, α, governs the precision of the prior. Strictly then, the prior is $P(\mathbf{w} \mid \alpha)$. This kind of parameter, which governs the scale of another parameter, is referred to as a *hyper-parameter*. OK - putting it all together we can then write,

$$P(a \mid D, \alpha, \beta) \propto \int P(a \mid D, \mathbf{w}, \beta)P(D \mid \mathbf{w})P(\mathbf{w} \mid \alpha)d\mathbf{w}$$

Now we use the fact that the distribution

$$P(\mathbf{w} \mid D) \propto \exp\{\ln P(D \mid \mathbf{w}) + \ln P(\mathbf{w})\} = \exp\{-E(\mathbf{w})\}$$

and so we have that the effective error functional is

$$E(\mathbf{w}) = \underbrace{-\ln P(D \mid \mathbf{w})}_{\text{log likelihood}}\underbrace{-\ln P(\mathbf{w})}_{\text{prior}}$$

The first term, the negative log likelihood is the 'traditional' model fit term which is the error in modelling the data, E_{data} say. The second, the prior, is something which occurs due to the Bayesian approach. Plugging in the form of the prior into the above gives an error functional, which after ignoring terms that are independent of D or $\mathbf{w}$ and noting that $P(D \mid \mathbf{w}) \propto \exp\{\beta E_{data}(\mathbf{w})\}$, is of the form:

$$E(\mathbf{w}, \alpha, \beta) = \beta E_{data}(\mathbf{w}) + \frac{\alpha}{2}\mathbf{w}^2$$

(the factor of 2 'missing' from the first term arises as $E_{data} = 1/2\sum(a-t)^2$). Hence

$$E = \beta E_{data} + \alpha E_{weights}$$

We see that this is the same as the regularisation approach (as $\mathbf{w}^2 = \sum_i w_i^2$).

> *Regularisation parameters arise naturally as part of the Bayesian paradigm as they are just hyperparameters which govern the prior distributions over the model weights (α) or the expected noise on the targets (β).*

Putting these things together gives:

$$P(a \mid D, \alpha, \beta) \propto \int \exp\left\{-\frac{\beta}{2}(a-t)^2 - E(\mathbf{w}, \alpha)\right\} d\mathbf{w}$$

How do we proceed now?

One very powerful method, if we do not want to numerically integrate the above, is to make an approximation. This approach, detailed now, is known as the *Laplace approximation* and aims at expanding the error function quadratically and the variable a linearly. It turns out that this gives a neat, analytic solution and is not too gross an approximation.

Expanding the error to second order, making the assumption that we can find a local minimum in the error at $\mathbf{w} = \mathbf{w}^*$, gives

$$E(\mathbf{w}, \alpha, \beta) \approx E(\mathbf{w}^*, \alpha, \beta) + \frac{1}{2}\Delta\mathbf{w}^\mathsf{T}\mathbf{H}\Delta\mathbf{w}$$

where $\mathbf{H}$ is the Hessian matrix of the total error function $E(\mathbf{w}, \alpha, \beta)$, above and $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}^*$.

Now expanding a linearly

$$a(x, \mathbf{w}) = a(x, \mathbf{w}^*) + \mathbf{g}(x)^\mathsf{T}\Delta\mathbf{w}$$

where $\mathbf{g}(x) = \left(\frac{\partial a}{\partial \mathbf{w}}\right)(x)$ evaluated at $\mathbf{w} = \mathbf{w}^*$. This means that we can write:

$$P(a \mid D, \alpha, \beta) \propto \int \exp\left\{-\frac{\beta}{2}\left(a(x, \mathbf{w}^*) + \mathbf{g}(x)^\mathsf{T}\Delta\mathbf{w} - t\right)^2 - \frac{1}{2}\Delta\mathbf{w}^\mathsf{T}\mathbf{H}\Delta\mathbf{w}\right\} d\mathbf{w}$$

Note that the $E(\mathbf{w}^*, \alpha)$ term has been dropped, as this is a constant in the integral (does not depend upon $\mathbf{w}$).
This is just a Gaussian integral whose mean is given by:

$$a^*(x) = a(x, \mathbf{w}^*)$$

which is the *most-likely* value for a given some input x, and a variance given as:

$$\sigma^2 = \frac{1}{\beta} + \mathbf{g}^\mathsf{T}(x)\mathbf{H}^{-1}\mathbf{g}(x)$$

Bishop gives a hand-holding work-through of how to prove this in exercises 10.1,10.2 in his book. It is worth looking at this.
Before we go on to look at classification under this approach, we need to estimate α, β. The exact formalism will not be gone through here (it is covered in detail in

Bishop): the basic aproach, however, is as follows.

We can write

$$P(\mathbf{w} \mid D) = \int \int P(\mathbf{w} \mid \alpha, \beta, D) P(\alpha, \beta \mid D) d\alpha d\beta$$

This integration could be performed using numerical techniques. Many favour an alternative approach in which it is assumed that $P(\alpha, \beta \mid D)$ is very sharply peaked around the most-probable values for these hyper-parameters, then we may write

$$P(\mathbf{w} \mid D) \approx P(\mathbf{w} \mid \alpha^*, \beta^*)$$

This approach is often referred to as the *evidence* scheme as α^*, β^* can be estimated from the derivative of $P(D \mid \alpha, \beta)$ (the evidence), or its log. It turns out that

$$\ln P(D \mid \alpha, \beta) \propto -\alpha E_{weights}(\mathbf{w}^*) - \beta E_{data}(\mathbf{w}^*) - \frac{1}{2} \ln |\mathbf{H}| + \frac{W}{2} \ln \alpha + \frac{N}{2} \ln \beta$$

In which W, N are the number of weights and data and $\mathbf{H}$ is the Hessian of the total (regularised) error function. As the log determinant of a matrix can be written as the sum of log eigenvalues so

$$\ln |\mathbf{H}| = \sum \ln(\lambda_i + \alpha)$$

where $\{\lambda_i\}$ are the eigenvalues of the Hessian of the *un*-regularised (βE_{data}) error. We thence obtain,

$$\alpha^* = \frac{\gamma}{2E_{weights}}$$

and

$$\beta^* = \frac{N - \gamma}{2E_{data}}$$

where γ is given by

$$\gamma = \sum_i \frac{\lambda_i}{\lambda_i + \alpha}$$

which is just the number of 'well-determined' weights in the analyser.

Note that $\gamma, E_{data}, E_{weights}$ all change during optimisation, so the above formulae must be used as sucessive estimators. This means that, periodically, we use the current values of the errors etc. to re-estimate values for α^*, β^* and then these these to continue optimisation.

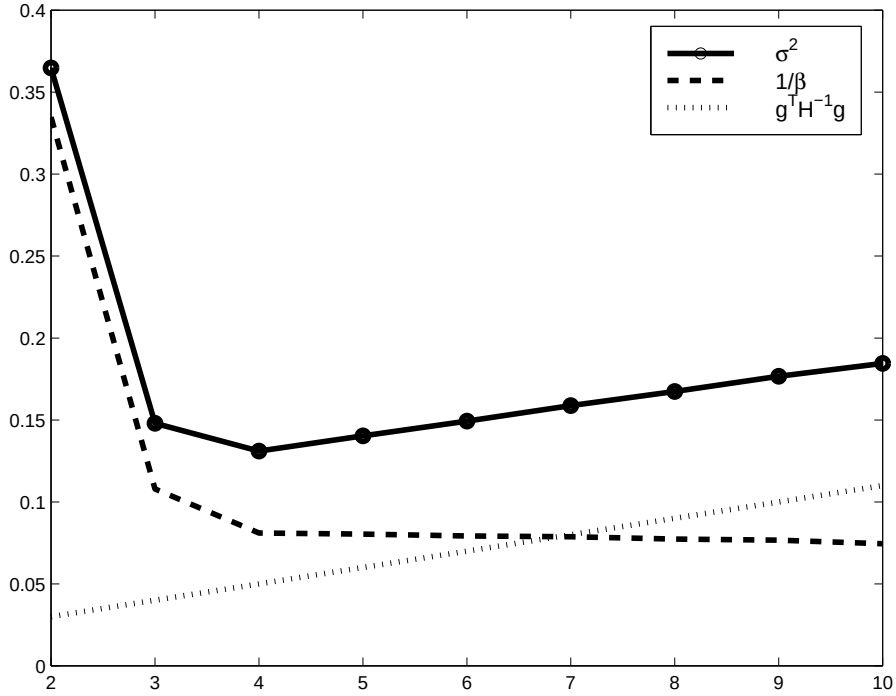

Figure 7: σ^2 terms from estimated variance Eqn. Training set only using spline RBF model.

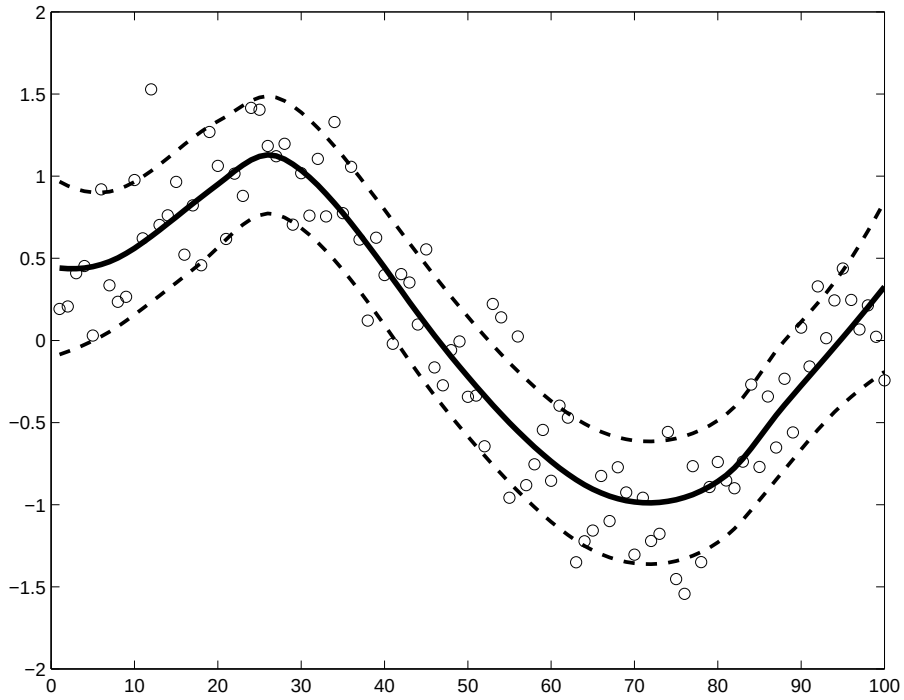

Figure 8: Mean and $\pm\sigma$ from Bayesian approach with 4-spline RBF model.

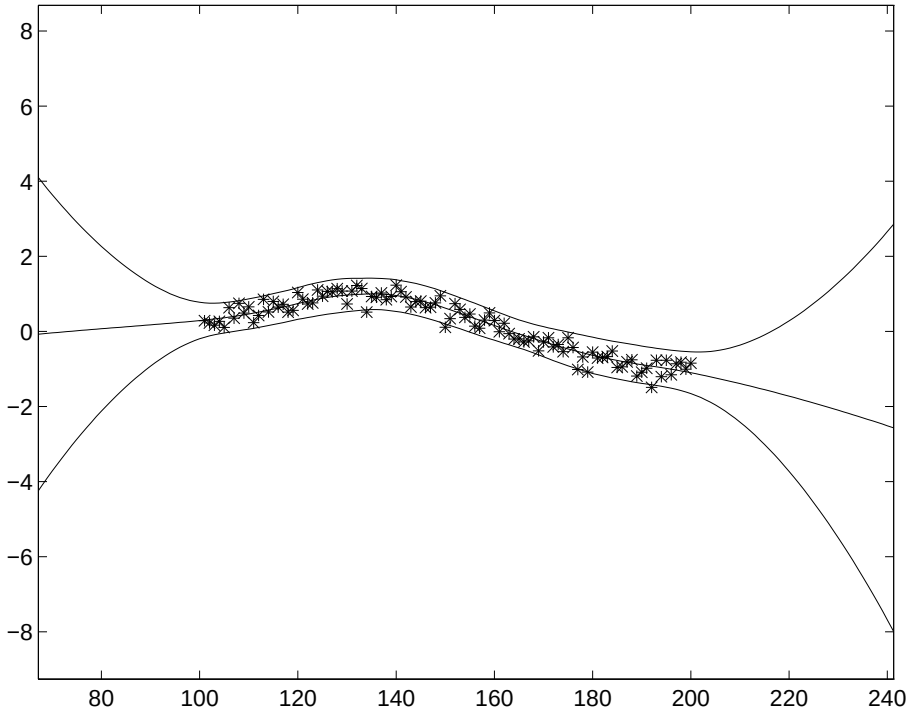

Figure 9: Mean and $\pm\sigma$ from Bayesian approach with 4-spline RBF model and extrapolation. Note the increasing error bars when there is no data.

The classification case

Remember the hyper-parameter β corresponded to the expected noise on the targets (labels) we have. In the case of classification we don't have any noise - we assume our labels are 1-of-n coded and so noiseless. This means that we drop the dependence on β in the above formulation. The mean and variance of the latent distribution are hence

$$a^*(x) = a(x, \mathbf{w}^*)$$

and

$$\sigma^2 = \mathbf{g}^\mathsf{T}(x)\mathbf{H}^{-1}\mathbf{g}(x)$$

The output of our classifier (and for now let us just look at a single output i.e. a two-class problem) is

$$y(x, D) = P(C_1 \mid x, D) = \int g(a)P(a|x, D)da$$

where $y = g(a)$ is the sigmoid transfer function. The above is non-analytic though and we resort to an approximation,

$$P(C_1 \mid x, D) \approx g(\kappa(\sigma^2)a^*)$$

where

$$\kappa(\sigma^2) = \left(1 + \frac{\pi\sigma^2}{8}\right)^{-1/2}$$

Let us see what effect changes in the latent variance (uncertainty) have on our classification probability. Consider the line $a^*(x) = 2$. Note that the resultant estimated posterior probability goes down towards 1/2 as the uncertainty in a increases. Remember that the uncertainty in a decision is the distance from unity of the largest posterior, which is worst when the posterior equals the class prior (1/2 in this 2-class problem).

> *In a principled way so uncertainty (high variance) in the latent distribution is automatically represented as a lower certainty of decision.*

Note also that this process, known as *moderation*, will not make the decisions we make different unless we have a non-uniform cost matrix or use a reject option. Then it can have dramatic effects.

The general linear model case

So far we have not specified what model is used in the above. Note that the theory is general, but that we need to estimate the Hessian matrix under whatever model we

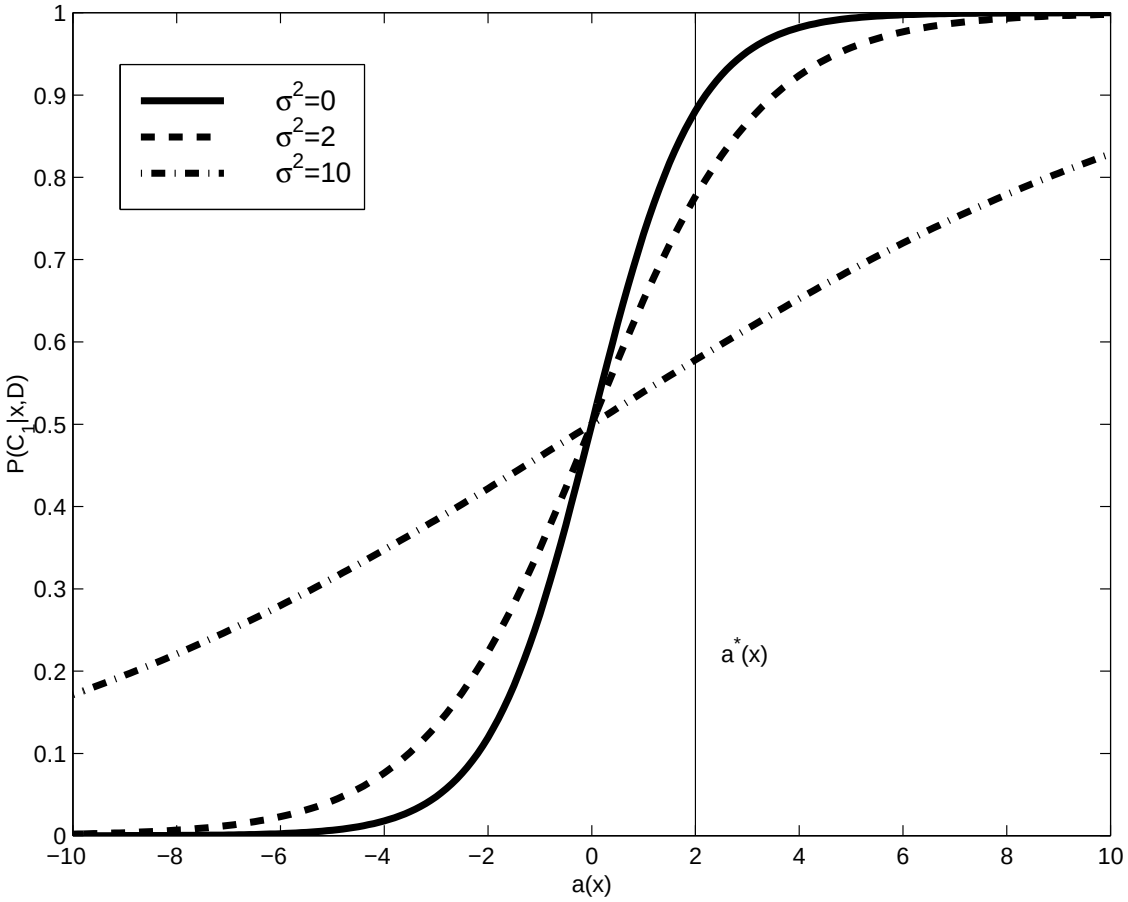

Figure 10: Changes in slope of sigmoid due to latent variable uncertainty.

choose. If the model is *linear* in the parameters (weights) then the Hessian is very easy to obtain.

Consider first the simple case of a $y = a$ where

$$a = \mathbf{w}^{\mathsf{T}}\mathbf{x}$$

(I have included the bias weight in here by 'augmenting' the input, $\mathbf{x}$, by a column of ones). The Hessian matrix is given as

$$\mathbf{H} = \frac{\partial}{\partial \mathbf{w}} \left(\frac{\partial E}{\partial \mathbf{w}} \right)^{\mathsf{T}}$$

The error function is given as

$$E = \frac{\beta}{2} \sum_n (y(\mathbf{x}_n) - t_n)^2 + \frac{\alpha}{2} \mathbf{w}^T \mathbf{w}$$

The Hessian is hence:

$$\mathbf{H} = \beta \sum_n \mathbf{x}_n \mathbf{x}_n^{\mathsf{T}} + \alpha \mathbf{I}$$

denoting the sample covariance matrix as,

$$\mathbf{C} = \langle \mathbf{x}_n \mathbf{x}_n^\mathsf{T} \rangle \approx \frac{1}{N} \sum_n \mathbf{x}_n \mathbf{x}_n^\mathsf{T}$$

so the Hessian is

$$\mathbf{H} = \beta N \mathbf{C} + \alpha \mathbf{I}$$

Note that the uncertainty (variance) is dependent upon $\mathbf{H}^{-1}$. This means that larger values of N give a smaller Hessian. The more points you have the better you can fit the model. Large α gives a smoother model and reduces the uncertainty and a large β means little noise on the targets and again gives more certain models - all of this makes common sense! Another way of looking at the above is with the eigenspectrum of $\mathbf{H}$ which will be of the form $\beta N \lambda_i + \alpha$ giving the number of well-determined parameters given by

$$\sum_i \frac{\lambda_i}{\lambda_i + \frac{\alpha}{\beta N}}$$

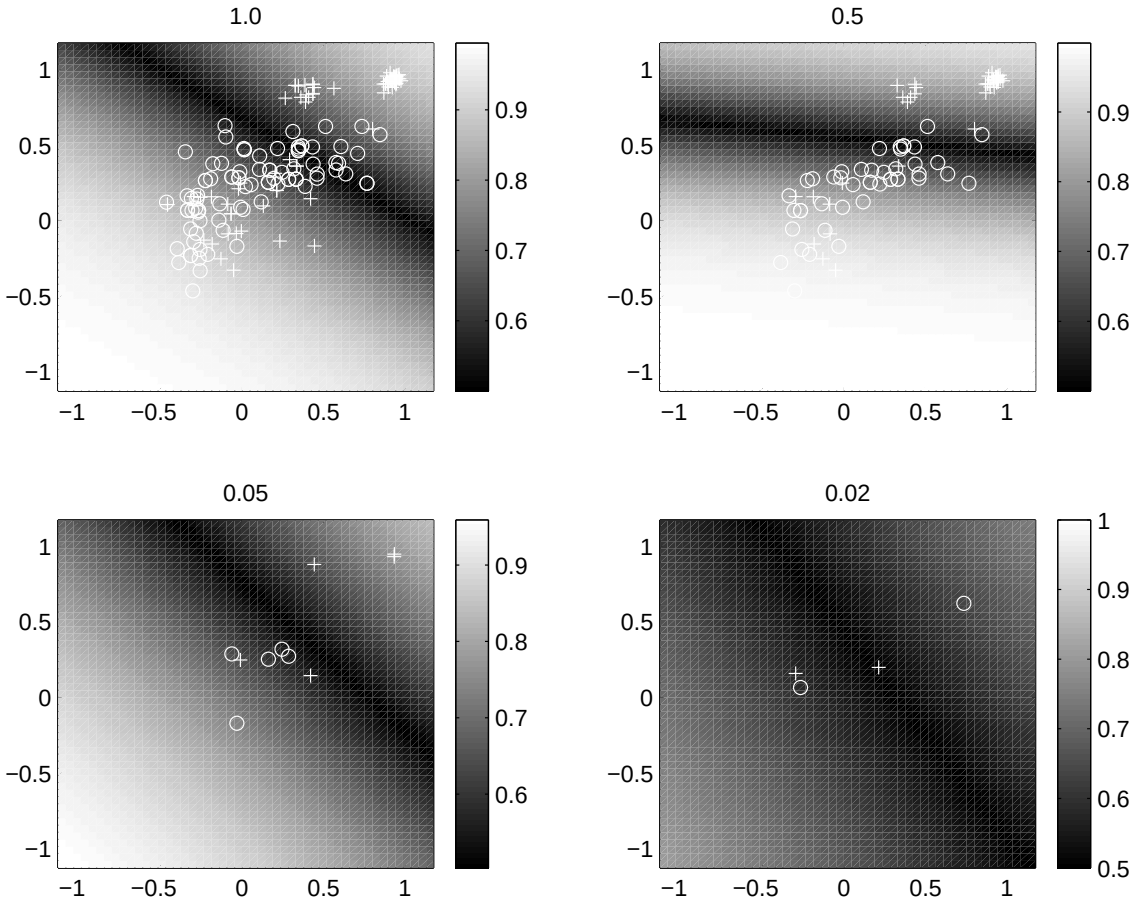

Figure 11: Changes in the number of points give lower certainty (plot of maximum posterior).

We can easily extend the above to the case of the basis-function approach. Here the variable $a(x)$ is modelled as

$$a(x) = \mathbf{w}^{\mathsf{T}} \boldsymbol{\phi}(x)$$

where $\boldsymbol{\phi}(x)$ are the basis function responses to input x, i.e. some non-linear mapping of x. For example, this can be the spline function or a Gaussian. The same approach as above gives identical expressions for the Hessian etc. save for $\mathbf{x}$ replaced by $\boldsymbol{\phi}$.

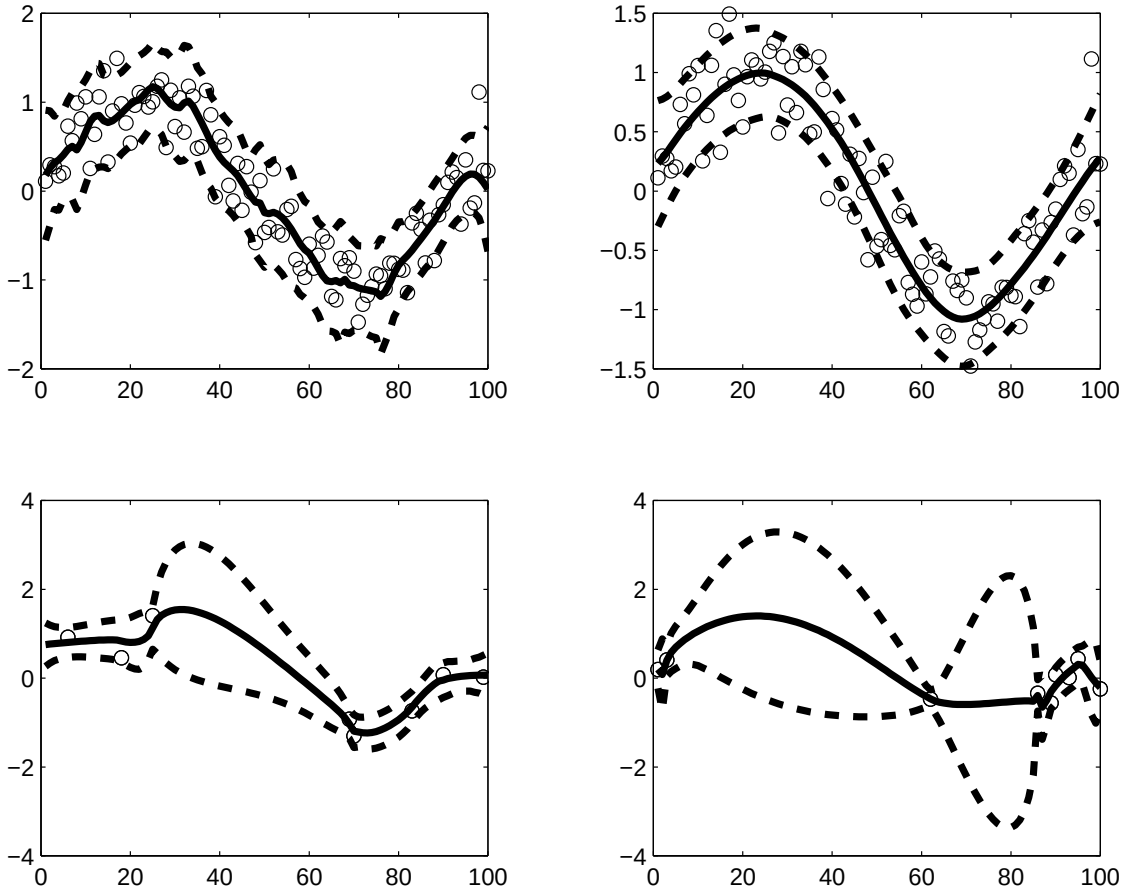

Figure 12: Changes in the number of points and parameters change the error bars. [top left] too many parameters. [top right] just right. [bottom left & right] too few points. Note how the error bars drop where data is. Note also the change of scale in the bottom plots.

Bayes and committees

If we go back to the idea of the Laplace approximation we see that what it does is to find a mode of the posterior $P(\mathbf{w} \mid D)$ (finds a minimum of the error function) and then expands the error quadratically, which is the same as fitting a Gaussian bump to the mode of the distribution. What we *really* want to do is to integrate over all the parameter space. A reasonable approach might be just to integrate round each bump (mode) in the probability space though and then add up the results. So we integrate

over small regions, r_i around each of the many non-equivalent minima, m_i, of the energy function

$$a(x, D) = \sum_{m_i} P(m_i \mid D) \int_{r_i} a_i(x, \mathbf{w}) P(\mathbf{w} \mid D, m_i) d\mathbf{w}$$

which we will rewrite as

$$a(x, D) = \sum_{m_i} P(m_i \mid D) a_i(x, D)$$

which is just a committee of models in which

$$a_i(x, D) = \int_{r_i} a_i(x, \mathbf{w}) P(\mathbf{w} \mid D, m_i) d\mathbf{w}$$

If we denote the mixings in the committee as

$$\gamma_i = P(m_i \mid D)$$

then we may get new expressions for the mean and variance associated with the distribution of a, noting that for a combination of random variables the total variance is the sum of the variances plus the variance of the means. The mean is just the weighted average over committee members, i.e.

$$\overline{a} = \sum_i \gamma_i a_i$$

and the variance

$$\sigma^2_{tot} = \sigma^2_c + \sigma^2_e + \sigma^2_{wu}$$

where σ^2_c represents the committee *variance*,

$$\sigma^2_c = \text{ var } [a_i(\mathbf{x})]$$

σ^2_e the target noise (and residual bias - though this is assumed to be small compared to the target noise for well-formed models) estimated by

$$\sigma^2_e = \sum_i \frac{\gamma_i}{\beta^*_i}$$

and σ^2_{wu} the parameter uncertainty of the committee,

$$\sigma^2_{wu} = \sum_i \gamma_i \mathbf{g}^T_i(\mathbf{x}) \mathbf{H}^{-1}_i \mathbf{g}_i(\mathbf{x})$$

This is an intuitively pleasing result as the total error may be regarded in a common-sense manner as arising from three distinct causes.

- The first term penalises variant decisions *between* committee members,
- the second penalises the committee *as a whole* if the output is associated with a region of input space with high target noise levels
- and the third penalises solutions which have poorly set parameters.

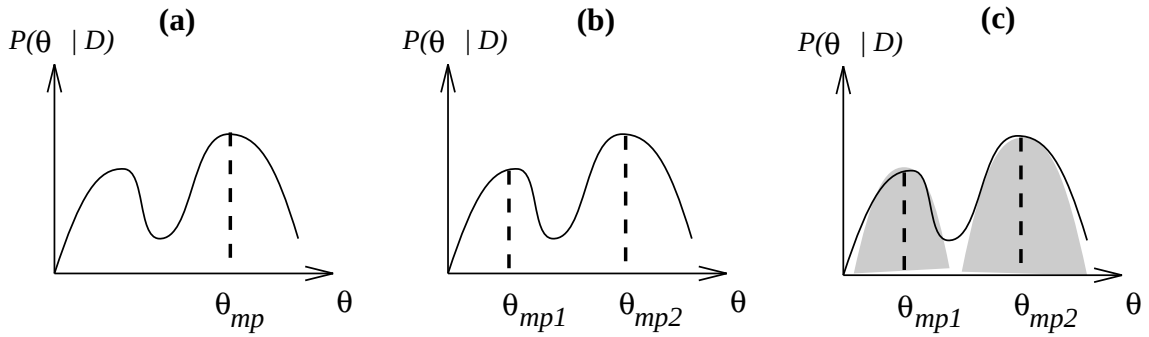

Figure 13: (a) single model, (b) multiple models & (c) local Gaussian approximation around each mode.