New address: Department of Mathematics, King's College London,
Strand, London WC2R 2LS, U.K.
Email: peter.sollich@kcl.ac.uk
Web: www.mth.kcl.ac.uk/~psollich/

Learning curves for Gaussian processes

Peter Sollich
Department of Physics, University of Edinburgh
Edinburgh EH9 3JZ, U.K., email: P.Sollich@ed.ac.uk

Abstract

I consider the problem of calculating learning curves (i.e., average generalization performance) of Gaussian processes used for regression. A simple expression for the generalization error in terms of the eigenvalue decomposition of the covariance function is derived, and used as the starting point for several approximation schemes. I identify where these become exact, and compare with existing bounds on learning curves; the new approximations, which can be used for any input space dimension, generally get substantially closer to the truth.

1 INTRODUCTION: GAUSSIAN PROCESSES

Within the neural networks community, there has in the last few years been a good deal of excitement about the use of Gaussian processes as an alternative to feedforward networks [1]. The advantages of Gaussian processes are that prior assumptions about the problem to be learned are encoded in a very transparent way, and that inference—at least in the case of regression that I will consider—is relatively straightforward. One crucial question for applications is then how 'fast' Gaussian processes learn, *i.e.*, how many training examples are needed to achieve a certain level of generalization performance. The typical (as opposed to worst case) behaviour is captured in the *learning curve*, which gives the average generalization error ϵ as a function of the number of training examples n. Several workers have derived bounds on $\epsilon(n)$ [2, 3, 4] or studied its large n asymptotics. As I will illustrate below, however, the existing bounds are often far from tight; and asymptotic results will not necessarily apply for realistic sample sizes n. My main aim in this paper is therefore to derive approximations to $\epsilon(n)$ which get closer to the true learning curves than existing bounds, and apply both for small and large n.

In its simplest form, the regression problem that I am considering is this: We are trying to learn a function θ^* which maps inputs x (real-valued vectors) to (real-valued scalar) outputs $\theta^*(x)$. We are given a set of training data D, consisting of n input-output pairs (x_l, y_l); the training outputs y_l may differ from the 'clean' target outputs $\theta^*(x_l)$ due to corruption by noise. Given a test input x, we are then asked to come up with a prediction $\theta(x)$ for the corresponding output, expressed either in the simple form of a mean prediction $\hat{\theta}(x)$ plus error bars, or more comprehensively in terms of a 'predictive distribution' $P(\theta(x)|x, D)$. In a Bayesian setting, we do this by specifying a prior $P(\theta)$ over our hypothesis functions, and a likelihood $P(D|\theta)$ with which each θ could have generated the training data; from this we deduce the posterior distribution $P(\theta|D) \propto P(D|\theta)P(\theta)$. In the case of feedforward networks, where the hypothesis functions θ are parameterized by a set of network weights, the predictive distribution then needs to be extracted by integration over this posterior, either by computationally intensive Monte Carlo techniques or by approximations which lead to analytically tractable integrals. For a Gaussian process, on the other hand, obtaining the predictive distribution is trivial (see below); one reason for this is that the prior $P(\theta)$ is defined directly over input-output functions θ. How is this done? Any θ is uniquely determined by its output values $\theta(x)$ for all x from the input domain, and for a Gaussian process, these are simply assumed to have a joint Gaussian distribution (hence the name). This distribution can be specified by the mean values $\langle\theta(x)\rangle_\theta$ (which I assume to be zero in the following, as is commonly done), and the covariances $\langle\theta(x)\theta(x')\rangle_\theta = C(x, x')$; $C(x, x')$ is called the *covariance function* of the Gaussian process. It encodes in an easily interpretable way prior assumptions about the function to be learned. Smoothness, for example, is controlled by the behaviour of $C(x, x')$ for $x' \to x$: The Ornstein-Uhlenbeck (OU) covariance function $C(x, x') \propto \exp(-|x-x'|/l)$ produces very rough (non-differentiable) functions, while functions sampled from the squared exponential (SE) prior with $C(x, x') \propto \exp(-|x - x'|^2/(2l^2))$ are infinitely differentiable. The 'length scale' parameter l, on the other hand, corresponds directly to the distance in input space over which we expect our function to vary significantly. More complex properties can also be encoded; by replacing l with different length scales for each input component, for example, relevant (small l) and irrelevant (large l) inputs can be distinguished.

How does inference with Gaussian processes work? I only give a brief summary here and refer to existing reviews on the subject (see *e.g.* [5, 1]) for details. It is simplest to assume that outputs y are generated from the 'clean' values of a hypothesis function $\theta(x)$ by adding Gaussian noise of x-independent variance σ^2. The joint distribution of a set of training outputs $\{y_l\}$ and the function values $\theta(x)$ is then also Gaussian, with covariances given by

$$\langle y_l y_m\rangle = C(x_l, x_m) + \sigma^2\delta_{lm} = (\mathbf{K})_{lm}, \qquad \langle y_l\theta(x)\rangle = C(x_l, x) = (\mathbf{k}(x))_l;$$

here I have defined an $n \times n$ matrix $\mathbf{K}$ and x-dependent n-component vectors $\mathbf{k}(x)$. The posterior distribution $P(\theta|D)$ is then obtained by simply conditioning on the $\{y_l\}$. It is again Gaussian and has mean and variance

$$\langle\theta(x)\rangle_{\theta|D} = \hat{\theta}(x) = \mathbf{k}(x)^{\mathbf{T}}\mathbf{K}^{-1}\mathbf{y} \tag{1}$$

$$\left\langle(\theta(x) - \hat{\theta}(x))^2\right\rangle_{\theta|D} = C(x, x) - \mathbf{k}(x)^{\mathbf{T}}\mathbf{K}^{-1}\mathbf{k}(x) \tag{2}$$

Eqs. (1,2) solve the inference problem for Gaussian process: They provide us directly with the predictive distribution $P(\theta(x)|x, D)$. The posterior variance, eq. (2), in fact also gives us the expected generalization error at x. Why? If the teacher is θ^*, the squared deviation between our mean prediction and the teacher output is[1] $(\hat{\theta}(x) - \theta^*(x))^2$; averaging this over the posterior distribution of teachers $P(\theta^*|D)$ just gives (2). The underlying assumption is that our assumed Gaussian process prior is the true one from which teachers are actually generated (and that we are using the correct noise model). Otherwise, a more complicated expression for the

[1] One can also one measure the generalization by the squared deviation between the prediction $\hat{\theta}(x)$ and the *noisy* teacher output; this simply adds a term σ^2 to eq. (3).

expected generalization error results; in line with most other work on the subject, I only consider the 'correct prior' case in the following. Averaging the generalization error at x over the distribution of inputs gives then

$$\epsilon(D) = \left\langle C(x,x) - \mathbf{k}(x)^{\mathbf{T}}\mathbf{K}^{-1}\mathbf{k}(x)\right\rangle_x \tag{3}$$

This form of the generalization error (which is well known [2, 3, 4, 5]) still depends on the training inputs (the fact that the training outputs have dropped out already is a signature of the fact that Gaussian processes are *linear* predictors, compare (1)). Averaging over data sets yields the quantity we are after,

$$\epsilon = \langle \epsilon(D) \rangle_D \,. \tag{4}$$

This average expected generalization error (I will drop the 'average expected' in the following) only depends on the number of training examples n; the function $\epsilon(n)$ is called the *learning curve*. Its exact calculation is difficult because of the joint average in eqs. (3,4) over the training inputs x_l and the test input x.

2 LEARNING CURVES

As a starting point for an approximate calculation of $\epsilon(n)$, I first derive a representation of the generalization error in terms of the eigenvalue decomposition of the covariance function. Mercer's theorem (see e.g. [6]) tells us that the covariance function can be decomposed into its eigenvalues λ_i and eigenfunctions $\phi_i(x)$:

$$C(x,x') = \sum_{i=1}^{\infty} \lambda_i \phi_i(x)\phi_i(x') \tag{5}$$

This is simply the analogue of the eigenvalue decomposition of a finite symmetric matrix; the eigenfunctions can be taken to be normalized such that $\langle \phi_i(x)\phi_j(x)\rangle_x = \delta_{ij}$. Now write the data-dependent generalization error (3) as $\epsilon(D) = \langle C(x,x)\rangle_x - \mathrm{tr}\,\left\langle \mathbf{k}(x)\mathbf{k}(x)^{\mathbf{T}}\right\rangle_x \mathbf{K}^{-1}$ and perform the x-average in the second term:

$$\left\langle (\mathbf{k}(x)\mathbf{k}(x)^{\mathbf{T}})_{lm}\right\rangle_x = \sum_{ij} \lambda_i\lambda_j \phi_i(x_l)\,\langle \phi_i(x)\phi_j(x)\rangle\,\phi_j(x_m) = \sum_i \lambda_i^2 \phi_i(x_l)\phi_i(x_m)$$

This suggests introducing the diagonal matrix $(\mathbf{\Lambda})_{ij} = \lambda_i\delta_{ij}$ and the 'design matrix' $(\Phi)_{li} = \phi_i(x_l)$, so that $\left\langle \mathbf{k}(x)\mathbf{k}(x)^{\mathbf{T}}\right\rangle_x = \Phi\mathbf{\Lambda}^2\Phi^{\mathbf{T}}$. One then also has $\langle C(x,x)\rangle_x = \mathrm{tr}\,\mathbf{\Lambda}$, and the matrix $\mathbf{K}$ is expressed as $\mathbf{K} = \sigma^2\mathbf{I} + \Phi\mathbf{\Lambda}\Phi^{\mathbf{T}}$, $\mathbf{I}$ being the identity matrix. Collecting these results, we have

$$\epsilon(D) = \mathrm{tr}\,\mathbf{\Lambda} - \mathrm{tr}\,(\sigma^2\mathbf{I} + \Phi\mathbf{\Lambda}\Phi^{\mathbf{T}})^{-1}\Phi\mathbf{\Lambda}^2\Phi^{\mathbf{T}}$$

This can be simplified using the Woodbury formula for matrix inverses (see *e.g.* [7]), which applied to our case gives $(\sigma^2\mathbf{I}+\Phi\mathbf{\Lambda}\Phi^{\mathbf{T}})^{-1} = \sigma^{-2}[\mathbf{I} - \Phi(\sigma^2\mathbf{I}+\mathbf{\Lambda}\Phi^{\mathbf{T}}\Phi)^{-1}\mathbf{\Lambda}\Phi^{\mathbf{T}}]$; after a few lines of algebra, one then obtains the final result

$$\epsilon = \langle \epsilon(D)\rangle_D\,, \qquad \epsilon(D) = \mathrm{tr}\,\sigma^2\mathbf{\Lambda}(\sigma^2\mathbf{I} + \mathbf{\Lambda}\Phi^{\mathbf{T}}\Phi)^{-1} = \mathrm{tr}\,(\mathbf{\Lambda}^{-1} + \sigma^{-2}\Phi^{\mathbf{T}}\Phi)^{-1} \tag{6}$$

This exact representation of the generalization error is one of the main results of this paper. Its advantages are that the average over the test input x has already been carried out, and that the remaining dependence on the training data is contained entirely in the matrix $\Phi^{\mathbf{T}}\Phi$. It also includes as a special case the well-known result for linear regression (see *e.g.* [8]); $\mathbf{\Lambda}^{-1}$ and $\Phi^{\mathbf{T}}\Phi$ can be interpreted as suitably generalized versions of the weight decay (matrix) and input correlation matrix. Starting from (6), one can now derive approximate expressions for the learning curve $\epsilon(n)$. The most naive approach is to entirely neglect the fluctuations in $\Phi^{\mathbf{T}}\Phi$

over different data sets and replace it by its average, which is simply $\left\langle(\mathbf{\Phi^T\Phi})_{ij}\right\rangle_D = \sum_l \left\langle\phi_i(x_l)\phi_j(x_l)\right\rangle_D = n\delta_{ij}$. This leads to the *N*aive approximation

$$\epsilon_{\mathrm{N}}(n) = \mathrm{tr}\,(\mathbf{\Lambda}^{-1} + \sigma^{-2}n\mathbf{I})^{-1} \tag{7}$$

which is not, in general, very good. It does however become exact in the large noise limit $\sigma^2 \to \infty$ at constant n/σ^2: The fluctuations of the elements of the matrix $\sigma^{-2}\mathbf{\Phi^T\Phi}$ then become vanishingly small (of order $\sqrt{n}\sigma^{-2} = (n/\sigma^2)/\sqrt{n} \to 0$) and so replacing $\mathbf{\Phi^T\Phi}$ by its average is justified.

To derive better approximations, it is useful to see how the matrix $\mathcal{G} = (\mathbf{\Lambda}^{-1} + \sigma^{-2}\mathbf{\Phi^T\Phi})^{-1}$ changes when a new example is added to the training set. One has

$$\mathcal{G}(n+1) - \mathcal{G}(n) = \left[\mathcal{G}^{-1}(n) + \sigma^{-2}\psi\psi^{\mathbf{T}}\right]^{-1} - \mathcal{G}(n) = -\,\frac{\mathcal{G}(n)\psi\psi^{\mathbf{T}}\mathcal{G}(n)}{\sigma^2 + \psi^{\mathbf{T}}\mathcal{G}(n)\psi} \tag{8}$$

in terms of the vector ψ with elements $(\psi)_i = \phi_i(x_{n+1})$; the second identity uses again the Woodbury formula. To get exact learning curves, one would have to average this update formula over both the new training input x_{n+1} and all previous ones. This is difficult, but progress can be made by again neglecting some fluctuations: The average over x_{n+1} is approximated by replacing $\psi\psi^{\mathbf{T}}$ by its average, which is simply the identity matrix; the average over the previous training inputs by replacing $\mathcal{G}(n)$ by its average $\mathbf{G}(n) = \langle\mathcal{G}(n)\rangle_D$. This yields the approximation

$$\mathbf{G}(n+1) - \mathbf{G}(n) = -\,\frac{\mathbf{G}^2(n)}{\sigma^2 + \mathrm{tr}\,\mathbf{G}(n)} \tag{9}$$

Iterating from $\mathbf{G}(n=0) = \mathbf{\Lambda}$, one sees that $\mathbf{G}(n)$ remains diagonal for all n, and so (9) is trivial to implement numerically. I call the resulting $\epsilon_{\mathrm{D}}(n) = \mathrm{tr}\,\mathbf{G}(n)$ the *D*iscrete approximation to the learning curve, because it still correctly treats n as a variable with discrete, integer values. One can further approximate (9) by taking n as continuously varying, replacing the difference on the left-hand side by the derivative $d\mathbf{G}(n)/dn$. The resulting differential equation for $\mathbf{G}(n)$ is readily solved; taking the trace, one obtains the generalization error

$$\epsilon_{\mathrm{UC}}(n) = \mathrm{tr}\,(\mathbf{\Lambda}^{-1} + \sigma^{-2}n'\mathbf{I})^{-1} \tag{10}$$

with n' determined by the self-consistency equation $n' + \mathrm{tr}\,\ln(\mathbf{I} + \sigma^{-2}n'\mathbf{\Lambda}) = n$. By comparison with (7), n' can be thought of as an 'effective number of training examples'. The subscript UC in (10) stands for *U*pper *C*ontinuous approximation. As the name suggests, there is another, *lower* approximation also derived by treating n as continuous. It has the same form as (10), but a different self-consistent equation for n', and is derived as follows. Introduce an auxiliary offset parameter v (whose usefulness will become clear shortly) by $\mathcal{G}^{-1} = v\mathbf{I} + \mathbf{\Lambda}^{-1} + \sigma^{-2}\mathbf{\Phi^T\Phi}$; at the end of the calculation, v will be set to zero again. As before, start from (8)—which also holds for nonzero v—and approximate $\psi\psi^{\mathbf{T}}$ and $\mathrm{tr}\,\mathcal{G}$ by their averages, but retain possible fluctuations of $\mathcal{G}$ in the numerator. This gives $\mathbf{G}(n+1) - \mathbf{G}(n) = -\left\langle\mathcal{G}^2(n)\right\rangle/[\sigma^2 + \mathrm{tr}\,\mathbf{G}(n)]$. Taking the trace yields an update formula for the generalization error ϵ, where the extra parameter v lets us rewrite the average on the right-hand side as $-\mathrm{tr}\,\left\langle\mathcal{G}^2\right\rangle = (\partial/\partial v)\mathrm{tr}\,\langle\mathcal{G}\rangle = \partial\epsilon/\partial v$. Treating n again as continuous, we thus arrive at the partial differential equation $\partial\epsilon/\partial n = (\partial\epsilon/\partial v)/(\sigma^2 + \epsilon)$. This can be solved using the method of characteristics [8] and (for $v = 0$) gives the *L*ower *C*ontinuous approximation to the learning curve,

$$\epsilon_{\mathrm{LC}}(n) = \mathrm{tr}\,(\mathbf{\Lambda}^{-1} + \sigma^{-2}n'\mathbf{I})^{-1}, \qquad n' = \frac{n\sigma^2}{\sigma^2 + \epsilon_{\mathrm{LC}}} \tag{11}$$

By comparing derivatives w.r.t. n, it is easy to show that this is always lower than the UC approximation (10). One can also check that all three approximations that I have derived (D, LC and UC) converge to the exact result (7) in the large noise limit as defined above.

3 COMPARISON WITH BOUNDS AND SIMULATIONS

I now compare the D, LC and UC approximations with existing bounds, and with the 'true' learning curves as obtained by simulations. A lower bound on the generalization error was given by Michelli and Wahba [2] as

$$\epsilon(n) \geq \epsilon_{\mathrm{MW}}(n) = \textstyle\sum_{i=n+1}^{\infty} \lambda_i \tag{12}$$

This is derived for the noiseless case by allowing 'generalized observations' (projections of $\theta^*(x)$ along the first n eigenfunctions of $C(x, x')$), and so is unlikely to be tight for the case of 'real' observations at discrete input points. Based on information theoretic methods, a different *L*ower bound was obtained by *O*pper [3]:

$$\epsilon(n) \geq \epsilon_{\mathrm{LO}}(n) = \frac{1}{4}\mathrm{tr}\,(\mathbf{\Lambda}^{-1} + 2\sigma^{-2}n\mathbf{I})^{-1} \times [\mathbf{I} + (\mathbf{I} + 2\sigma^{-2}n\mathbf{\Lambda})^{-1}]$$

This is always lower than the naive approximation (7); both incorrectly suggest that ϵ decreases to zero for $\sigma^2 \to 0$ at fixed n, which is clearly not the case (compare (12)). There is also an *U*pper bound due to *O*pper [3],

$$\tilde{\epsilon}(n) \leq \epsilon_{\mathrm{UO}}(n) = (\sigma^{-2}n)^{-1}\,\mathrm{tr}\,\ln(\mathbf{I} + \sigma^{-2}n\mathbf{\Lambda}) + \mathrm{tr}\,(\mathbf{\Lambda}^{-1} + \sigma^{-2}n\mathbf{I})^{-1} \tag{13}$$

Here $\tilde{\epsilon}$ is a modified version of ϵ which (in the rescaled version that I am using) becomes identical to ϵ in the limit of small generalization errors ($\epsilon \ll \sigma^2$), but never gets larger that $2\sigma^2$; for small n in particular, $\epsilon(n)$ can therefore actually be much larger than $\tilde{\epsilon}(n)$ and its bound (13). An upper bound on $\epsilon(n)$ itself was derived by Williams and Vivarelli [4] for one-dimensional inputs and stationary covariance functions (for which $C(x, x')$ is a function of $x - x'$ alone). They considered the generalization error at x that would be obtained from each individual training example, and then took the minimum over all n examples; the training set average of this 'lower envelope' can be evaluated explicitly in terms of integrals over the covariance function [4]. The resulting upper bound, $\epsilon_{\mathrm{WV}}(n)$, never decays below σ^2 and therefore complements the range of applicability of the UO bound (13).

In the examples in Fig. 1, I consider a very simple input domain, $x \in [0, 1]^d$, with a uniform input distribution. I also restrict myself to stationary covariance functions, and in fact I use what physicists call periodic boundary conditions. This is simply a trick that makes it easy to calculate the required eigenvalue spectra of the covariance function, but otherwise has little effect on the results as long as the length scale of the covariance function is smaller than the size of the input domain[2], $l \ll 1$. To cover the two extremes of 'rough' and 'smooth' Gaussian priors, I consider the OU $[C(x, x') = \exp(-|x-x'|/l)]$ and SE $[C(x, x') = \exp(-|x-x'|^2/2l^2)]$ covariance functions. The prior variance of the values of the function to be learned is simply $C(x, x) = 1$; one generically expects this 'prior ignorance' to be significantly larger than the noise on the training data, so I only consider values of $\sigma^2 < 1$. I also fix the covariance function length scale to $l = 0.1$; results for $l = 0.01$ are qualitatively similar. Several observations can be made from Figure 1. (1) The MW lower bound is not tight, as expected. (2) The bracket between Opper's lower and upper bounds (LO/UO) is rather wide (1-2 orders of magnitude); both give good representations of the overall shape of the learning curve only in the asymptotic regime (most clearly visible for the SE covariance function), *i.e.*, once ϵ has dropped below σ^2. (3) The WV upper bound (available only in $d = 1$) works

[2] In $d = 1$ dimension, for example, a 'periodically continued' stationary covariance function on $[0, 1]$ can be written as $C(x, x') = \sum_{r=-\infty}^{\infty} c(x - x' + r)$. For $l \ll 1$, only the $r = 0$ term makes a significant contribution, except when x and x' are within $\approx l$ of opposite ends of the input space. With this definition, the eigenvalues of $C(x, x')$ are given by the Fourier transform $\int_{-\infty}^{\infty} dx\, c(x) \exp(-2\pi iqx)$, for integer q.

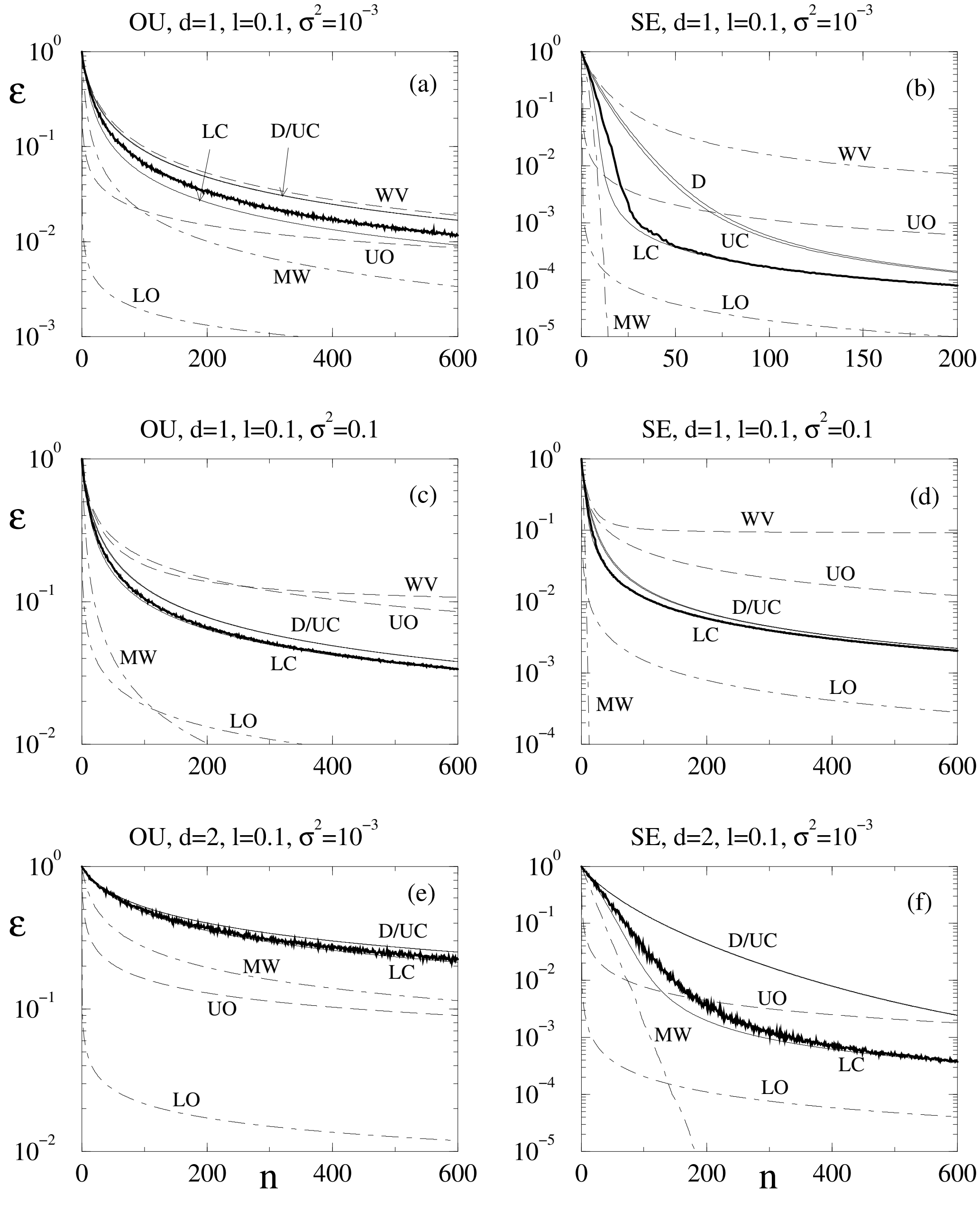

Figure 1: Learning curves $\epsilon(n)$: Comparison of simulation results (thick solid lines; the small fluctuations indicate the order of magnitude of error bars), approximations derived in this paper (thin solid lines; D = discrete, UC/LC = upper/lower continuous), and existing upper (dashed; UO = upper Opper, WV = Williams-Vivarelli) and lower (dot-dashed; LO = lower Opper, MW = Michelli-Wahba) bounds. The type of covariance function (Ornstein-Uhlenbeck/Squared Exponential), its length scale l, the dimension d of the input space, and the noise level σ^2 are as shown. Note the logarithmic y-axes. On the scale of the plots, D and UC coincide (except in (b)); the simulation results are essentially on top of the LC curve in (c-e).

well for the OU covariance function, but less so for the SE case. As expected, it is not useful in the asymptotic regime because it always remains above σ^2. (4) The discrete (D) and upper continuous (UC) approximations are very similar, and in fact indistinguishable on the scale of most plots. This makes the UC version preferable in practice, because it can be evaluated for any chosen n without having to step through all smaller values of n. (5) In all the examples, the true learning curve lies between the UC and LC curves. In fact I would conjecture that these two approximations provide upper and lower bounds on the learning curves, at least for stationary covariance functions. (6) Finally, the LC approximation comes out as the clear winner: For $\sigma^2 = 0.1$ (Fig. 1c,d), it is indistinguishable from the true learning curves. But even in the other cases it represents the overall shape of the learning curves very well, both for small n and in the asymptotic regime; the largest deviations occur in the crossover region between these two regimes.

In summary, I have derived an exact representation of the average generalization ϵ error of Gaussian processes used for regression, in terms of the eigenvalue decomposition of the covariance function. Starting from this, I have obtained three different approximations to the learning curve $\epsilon(n)$. All of them become exact in the large noise limit; in practice, one generically expects the opposite case ($\sigma^2/C(x,x) \ll 1$), but comparison with simulation results shows that even in this regime the new approximations perform well. The LC approximation in particular represents the overall shape of the learning curves very well, both for 'rough' (OU) and 'smooth' (SE) Gaussian priors, and for small as well as for large numbers of training examples n. It is not perfect, but does get substantially closer to the true learning curves than existing bounds. Future work will have to show how well the new approximations work for non-stationary covariance functions and/or non-uniform input distributions, and whether the treatment of fluctuations in the generalization error (due to the random selection of training sets) can be improved, by analogy with fluctuation corrections in linear perceptron learning [8].

Acknowledgements: I would like to thank Chris Williams for several very stimulating discussions, and him and Manfred Opper for providing me with copies of their papers [3, 4] prior to publication. I am grateful for financial support through a Royal Society Dorothy Hodgkin Research Fellowship.

[1] See *e.g.* D J C MacKay, Gaussian Processes, Tutorial at *NIPS 10*, and recent papers by Goldberg/Williams/Bishop (in *NIPS 10*), Williams and Barber/Williams (*NIPS 9*), Williams/Rasmussen (*NIPS 8*).

[2] C A Michelli and G Wahba. Design problems for optimal surface interpolation. In Z Ziegler, editor, *Approximation theory and applications*, pages 329–348. Academic Press, 1981.

[3] M Opper. Regression with Gaussian processes: Average case performance. In I K Kwok-Yee, M Wong, and D-Y Yeung, editors, *Theoretical Aspects of Neural Computation: A Multidisciplinary Perspective*. Springer, 1997.

[4] C K I Williams and F Vivarelli. An upper bound on the learning curve for Gaussian processes. Submitted for publication.

[5] C K I Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In M I Jordan, editor, *Learning and Inference in Graphical Models*. Kluwer Academic. In press.

[6] E Wong. *Stochastic Processes in Information and Dynamical Systems*. McGraw-Hill, New York, 1971.

[7] W H Press, S A Teukolsky, W T Vetterling, and B P Flannery. *Numerical Recipes in C (2nd ed.)*. Cambridge University Press, Cambridge, 1992.

[8] P Sollich. Finite-size effects in learning and generalization in linear perceptrons. *Journal of Physics A*, 27:7771–7784, 1994.