Preventing Overshoot of Splines with Application to Reinforcement Learning

Chris G. Drummond
Department of Computer Science
University of Ottawa
Ottawa,Ontario
Canada,K1N 6N5
cdrummon@csi.uottawa.ca

Abstract

Overshoot is caused by the interaction of a function approximator with rapid changes in the input data. In many applications this produces a visually confusing result. In reinforcement learning it can cause the iterative value estimator to diverge. This research investigates b-splines, a very general family of function approximators with a wide range of applications. Smoothing splines allow the addition of a bias which balances the fit to the data against some desired smoothness of the resulting function. This paper shows that one such bias can be used to eliminate overshoot.

1 INTRODUCTION

Spline function approximation is a general method of reducing the dimensionality of input data. This is typically motivated because the data is corrupted by noise. Through this reduction it is hoped to regress to the underlying function, eliminating the noise. Such function approximation is produced through the linear sum of a set of basis functions. The choice of basis function determines the form of the resulting function. Ideally this form is the same as that of the underlying function, which minimises the number of basis functions needed to represent it accurately. In many problems the abstract properties of the input space may suggest a good choice of basis function. Unless otherwise known, a reasonable heuristic might be to assume that neighbours have similar function values, and derivatives, predicating smooth basis functions. But if this assumption of smoothness is not always met, some undesirable interactions between the approximator and the data may be produced. A common problem is the "ringing" associated with discontinuities in the underlying function. The ringing produces overshoot, which can occur even if the first derivative is finite. Figure 1, "least squared", shows the approximation of a step function by cubic b-splines.

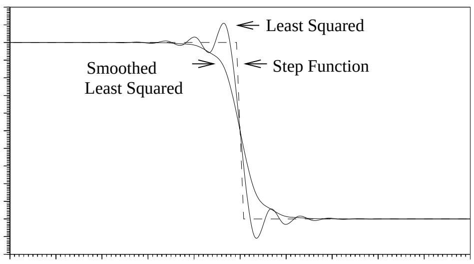

Figure 1: Overshoot

This paper presents a method of eliminating this effect through the use of a smoothing spline. The spline used will be a membrane spline, which intuitively can be likened to a elastic sheet. The tension in the sheet discourages large derivatives in the approximating function. This paper shows that there is a particular choice of tension which prevents overshoot. The "least squared" curve in figure 1 is the best approximation, in terms of minimising the least squared error, to the step function shown. The "smoothed" curve shows the result of smoothing the function just enough to eliminate overshoot. This seems a better approximation to the human eye. In many applications, the overshoot can be problematical. In visual processing, for instance, ringing may produce false features or obscure existing features some distance from the step. The main interest in this research is the application of splines to reinforcement learning. Within this context, algorithms that approximate the value function are iterative, each local estimate of the function is derived from the estimated value of its neighbours. Overshoot, by exaggerating the local value function, can cause the process to diverge, a problem that has been the focus of much research in the reinforcement learning community (Boyan and Moore,1995,Baird,1995,Gordon,1995).

In the rest of the paper, section 2, discusses the form of the smoothing spline. It then shows, by relating it to the solution of a partial differential equation, how the critical tension parameter can be derived. Section 3 gives a brief overview of reinforcement learning and the modifications necessary to work with splines. Section 4 presents experiments with a commonly used test bed for reinforcement learning.

2 SMOOTHING SPLINE APPROXIMATION

A spline approximates a function by a curve consisting of piecewise polynomials. Additional flexibility is achieved over a single polynomial by allowing discontinuities in certain derivatives at the point two neighbours join. Splines are typically constructed from a series of basis functions which are translates of a single canonical function. One such family is the b-splines. The cubic b-spline in figure 2 is of order four, each piece is a cubic polynomial and the second derivative is continuous. The order one b-spline is a box function, no continuity, the order two a hat function, the function is continuous but the first derivative is not. Thus increasing the order of the basis increases the order of the polynomial pieces and the highest derivative that is continuous. This idea is readily extended to any number of dimensions. Typically higher dimension basis functions are produced by a simple product, the basis function value at a point is just the product of the values of one dimensional basis functions at their respective coordinates.

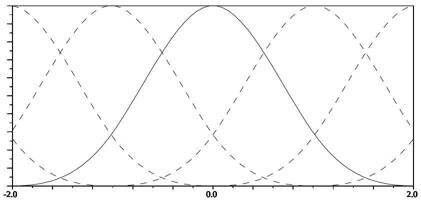

Figure 2: Cubic B-splines

The approximation is the linear sum of the product of the basis function ϕ with its coefficient, equation 1. The accuracy of the fit to the data is defined by a measure, such as L_2 norm. Equation 2 shows the norm for a continuous function, for discrete data the integral is replaced by a summation. This measure is commonly used, being intuitive and perhaps more importantly being mathematically easy to determine. Finding the coefficients that minimise this functional results in the least squared error fit of the spline to the input function.

$$f(x) = \sum_{j=1}^{m} c_j \phi_i(x) \qquad (1) \qquad\qquad D(f) = \int_R (f(x) - d)^2 dx \qquad (2)$$

One extension to this approach is the idea of a smoothing spline. Instead of just using the least squares error, an additional functional term is added. This term, often called the stabiliser, penalises functions that are not sufficiently smooth. What is meant by smoothness dictates the type of stabiliser used. The focus of this paper is the membrane spline. This intuitively acts like an elastic sheet or in one dimension as an elastic band, equation 3, discouraging high gradients. To control the balance between fit and smoothness the factor $1/k^2$ is introduced effectively controlling the tension in the band. There are alternative stabilisers, a popular one is the thin plate spline. This intuitively acts like a thin metal plate discouraging high curvatures. The minimising of such functionals is a variational problem. By differentiating with respect to the coefficients of the basis functions and equating the resulting equations to zero, a set of simultaneous equations is produced. These can readily be solved in matrix form.

$$E(f) = D(f) + 1/k^2 \int_R (\frac{df(x)}{dx})^2 dx \qquad (3)$$

2.1 FINDING THE RIGHT TENSION

In this section,the process of finding the least squares solution for the smoothing spline is rerepresented as an integration of the input function with a kernel. The same analysis should apply to discrete data using a weighted average as opposed to an integral. The kernel integrates to one. But for small values of tension it will be sometimes negative. It is this property which causes the overshoot. By increasing the tension, a point can be found where this kernel is always non-negative and thus will not create overshoot. This can be viewed as finding the critical damping value. The smoothed kernel is essentially an averager. Gordon (Gordon,1995) has proved that such approximators do not diverge in reinforcement learning. This work shows that a broad class of approximators can have this desirable characteristic.

The form of the smoothing spline is closely connected with the solution of a partial differential equation. The membrane spline determines the least squares approximation of the solution of the Euler equation $-\Delta U + k^2 U = F$. There are many ways of finding the exact solution of partial differential equations, the one of interest here is using Green's function. A Green's function, G(x) in equation 4, is essentially an impulse response solution of the differential equation. To produce the membrane spline approximation, one could convolve the input function with the appropriate Green's function, equation 5 and then find the least squares fit to the result.

$$-\Delta G(x) + k^2 G(x) = \delta(x) \qquad (4) \qquad U(x) = \int_{-\infty}^{\infty} G(x-\eta) f(\eta) d\eta \qquad (5)$$

In a partial differential equation problem, the boundary conditions must be considered. Instead of periodic boundary conditions, here a reflecting boundary is used, each boundary acts as mirror reflecting the function back and forth. For the purpose of this analysis the function, and basis functions, can be unfolded so they repeat over the appropriate grid. The Green's functions for free space can then be used. Equation 6 shows the Green's functions, for free space, for one and two dimensions (Stakgold,1979), K_0 is the modified Bessel function of the second kind, order 0.

$$G(k,x) = \frac{e^{-k|x|}}{2k} \qquad G(k,x,y) = \frac{K_0(k\sqrt{(}x^2+y^2))}{2\pi} \qquad (6)$$

There is also an alternative method to the variational method used above for finding the least squares fit. Had the original bases been orthogonal, the coefficient of any one basis could have been found by integrating it with the input function. The same process can still be carried out with b-splines even though they are not orthogonal. Instead of integrating with the original basis, a special basis function called the dual basis is used. The dual basis for b-splines introduced by Chui (Chui and Wang,1992,Sakakibara,1994) spans the same space as the b-spline. So the dual basis itself can be represented as a sum of b-splines. Reconstructing the function within any region needs the sum of m^d bases, where m is the order of the b-spline and d the dimension. Equation 7 shows this for one dimension, ω is the dual basis.

$$\hat{f}(\eta) = \sum_{i=1}^{m} \left(\int_{-\infty}^{\infty} \omega_l(x) f(x) dx \right) \phi_i(\eta) \qquad (7)$$

The next stage is to combine equation 7 with 5, rearranging the terms, reordering the integrals, and expanding the dual basis in terms of the original basis. As the basis functions are positioned on a grid, the resulting kernel would depend on the relative position within a cell. So for ease of analysis the variable η in equation 7 is replaced by a constant for one particular position. What needs to be determined is the worst position relative to the cell, the one that produces a kernel that requires the most smoothing. A set of m^d values for one particular position, will be effectively convolved discretely with the dual basis coefficients. The values that least smooth these coefficients are assumed determine this position.

$$\hat{f}(x_i) = \int_{-\infty}^{\infty} \underbrace{(G(k,x) * (\sum_{j=1}^{m} \sum_{l=1}^{n} a_{l-j} \phi_{l-j}(x) \phi_j(x_i)))}_{Kern_i(k,x)} f(x) dx \qquad (8)$$

This kernel, equation 8, is sometimes negative. All that is necessary now is to find the value of k that smooths the function sufficiently to make it always non-negative. Such a value must exist as the function integrates to one and the Green's function tends to a straight line as k decreases. The convolution can be solved in the fourier domain and followed by an inverse transform. This has been done for one dimension. Its complexity, however, makes it difficult to solve. Instead, at present, the correct value is found by using a discrete convolution and a binary search over the values of k. Figure 3 shows, for one dimension, the initial kernel and how the smoothing makes it always non-negative. In the two dimensional case, the accuracy of this convolution was reduced to speed up the convolution of the kernel with the Bessel function. The resulting value for k, used in the experiments of the next section, was twenty.

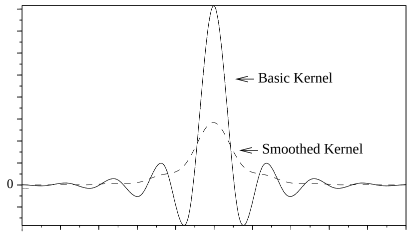

Figure 3: The Kernel

3 REINFORCEMENT LEARNING

A well known and general learning method is reinforcement learning. Much of its generality comes from treating a task as an accumulation of reward from the environment. Both the reward and result of an action can be stochastic. The system explores the world, learning for each action in each state the expectation of the sum of future rewards. These are often discounted the farther they are into the future, $R = E[\sum_{t=1}^{\infty} \gamma^t r_t]$ where γ, the discount factor, is less than or equal to one.

The algorithm most frequently used to address this problem is Q-learning. This algorithm assumes the world is a discrete Markov process thus both states and actions actions are discrete. For each action a in each state s, Q-learning maintains a rolling average of the immediate reward r plus the value for the best action a' of the next state s', formula 9. Watkins (Watkins and Dayan,1992) proved that this will converge to the optimal value with certain constraints on the reward and the learning rate α. Because of the nature of the problem representation, the optimal solution from any state is to take the best action from the present state and then take the optimal solution from then on.

$$Q^{t+1}_{s,a} = (1-\alpha)Q^t_{s,a} + \alpha(r + \gamma max_{a'} Q^t_{s',a'}) \tag{9}$$

Even assuming the world is a discrete, as the dimensionality increases the size of the state space can increase exponentially. For economy of representation it is desirable to approximate the surface by some function. A more general assumption is that the world is a continuous Markov process so unless the functional form is known beforehand some approximation scheme is necessary. Just applying existing techniques of function approximation has led to mixed results. Boyan and Moore (Boyan and Moore,1995) show how various common approximators can diverge on simple problems.

One can view discretisation as just one choice of basis, box functions. If the functional form is not discrete other approximators may produce a more compact representation for the same accuracy of fit. There are many other choices of functions, some having more inherent smoothness and are thus better able to represent smooth functions. Dependent on the order of the b-spline different degrees of smoothness can be obtained. When using splines, instead of updating an entry in a lookup table, each action produces a new sample which is added to the existing representation and a least squares fit to all points returned. How to formally include a learning rate is not clear in this case. Although in the experiments of the next section, some heuristic approach was used to accelerate learning.

4 EXPERIMENTS

The following experiments demonstrate the effect of the stabiliser in preventing divergence. The experiment uses the "car on the hill" problem (Moore,1992) that has become a de facto test bed for reinforcement learning algorithms. The problem, simply stated, is to get a car up a steep hill. The resultant state space is the position and velocity of the car. If the car is stationary part way up the hill, then it has insufficient acceleration to make to the top. So the car must reverse down the hill and then get up sufficient forward velocity before accelerating up the hill. This particular effect is what makes the problem interesting as it introduces a discontinuity in the value function. It is just such discontinuities that cause overshoot. In the experiments a reward of one is received if the car gets sufficiently up the hill, beyond position 0.6, within a limited velocity ± 0.4. Gamma is set to 0.99.

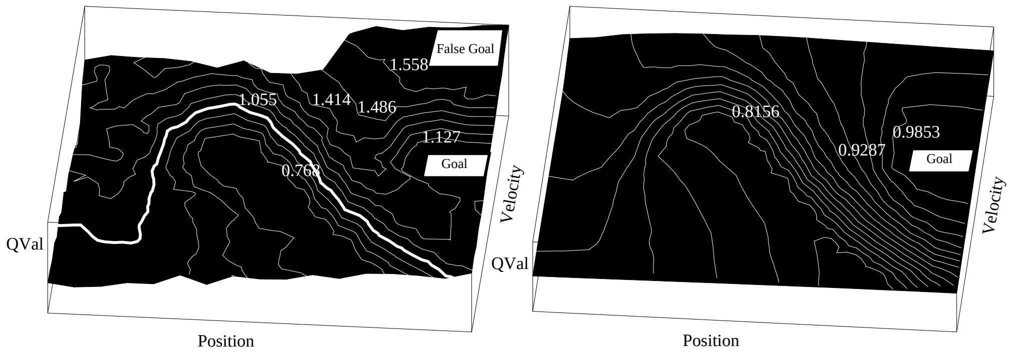

Figure 4: a) Divergent Approximation b) Convergent Approximation

Figure 4 shows the result of the experiments, firstly without the stabiliser and then using the stabiliser with the calculated value of tension. Figure 4b shows the essential characteristics of the space. The cluster of close contour lines approximates the discontinuity. It demarks the area where the car can accelerate directly up the hill towards the goal, largely towards the top of the figure, from the area where the car must initially accelerate away from the goal, the central region. In figure 4a the process is diverging. As the discontinuity is approximated, overshoot occurs along the top of the resultant edge. This becomes the local best value and other states reaching it become exaggerated. Eventually the whole top right corner achieves a value greater than the goal. Essentially a false goal is created. This continues to grow dragging up the rest of the space. The thick white line indicates a contour at approximately the same level as the true goal. Figure 4b shows how using the stabiliser with the appropriate value of tension eliminates this problem.

Although the use of the stabiliser prevents divergence, the resultant function has some error, in the least squares sense, most notably close to the discontinuity. To improve accuracy, without overshoot, more basis functions could be used. Extra basis function need only be added in the area of the discontinuity. The use of a hierarchical system of bases such as spline wavelets is one natural approach to this problem (Gortler at al.,1993). It should then be possible to locally adjust the tension just in those areas where new basess are added. Using the analysis given earlier, the amount necessary to adjust tension as the basis functions become smaller can be determined.

5 CONCLUSIONS

This paper has shown that the correct choice of tension in a membrane spline can remove the problem of overshoot. It has demonstrated that, through this means, general function approximators like b-splines can be used in reinforcement learning without divergence. Hopefully other areas that use such approximators might also benefit from this work. The ideas in this paper are built around basis functions that are translates of a canonical basis and should be readily applicable to wavelets. Some research within the neural network community has investigated stabilisers (Bishop,1992). Certainly where such networks use bases that are translates, this work should be relevant. A case in point is spline and wavelet networks.

References

L. C. Baird (1995) Residual Algorithms: Reinforcement Learning with Function Approximation *Proc. 12th Int. Conf. of Machine Learning* pp. 30-37

Chris M. Bishop (1992) Curvature-Driven Smoothing: A Learning Algorithm for Feedforward Networks *IEEE Trans. on Neural Networks* V. 4 No. 5 pp 882-884

Justin A. Boyan and Andrew W. Moore (1995) Generalization in Reinforcement Learning: Safely Approximating the Value Function *Advances in Neural Infosmation Processing Systems* V. 7 pp. 30-37

C. K. Chui and J. Z. Wang (1992) On compactly supported spline wavelets and a duality principle *Trans. Amer. Math Soc.* 330 pp 903-915

Geoffrey J. Gordon (1995) Stable Function Approximation in Dynamic Programming *Proc. 12th Int.Conf. of Machine Learning*, pp. 261-268

S. Gortler, P. Schroder, M. Cohen and P. Hanrahan (1993) Wavelet Radiosity *Computer Graphics, Annual Conference Series Siggraph* V. 1 pp 205-212

Andrew W. Moore (1992) Variable Resolution Dynamic Programming: Efficiently Learning Action Maps in Multivariate Real-valued State Spaces *Proc. 9th Int. Conf. on Machine Learning* pp 333-337

Susumu Sakakibara (1994) A Practice of Data Smoothing by B-spline Wavelets *Wavelets Theory, Algorithms and Applications* pp 179-196

Ivar Stakgold (1979) *Green's functions and boundary value problems.*

Christopher J.C.H. Watkins and Peter Dayan (1992) Technical Note:Q-Learning *Machine Learning* V. 8 No 3-4 pp 279-292