SPLINES AND EFFICIENCY IN DYNAMIC PROGRAMMING

by

James W. Daniel*

September 1973 CNA-78

Abstract

It is shown how one can use splines, represented in the B-spline basis, to reduce the difficulties of large storage requirements in dynamic programming via approximations to the minimum-return function without the inefficiency associated with using polynomials to the same end.

*Departments of Mathematics and of Computer Sciences and Center for Numerical Analysis at The University of Texas at Austin.
Research supported in part by the Office of Naval Research under Contract N00014-67-A-0126-0015, NR0-044-425; reproduction in whole or in part is permitted for any purposes of the United States government.

CENTER FOR NUMERICAL ANALYSIS

THE UNIVERSITY OF TEXAS AT AUSTIN

SPLINES AND EFFICIENCY IN DYNAMIC PROGRAMMING

by

James W. Daniel

1. Introduction

A commonly discussed technique [Bellman (1971), Bellman-Dreyfus (1962), Bellman-Kalaba (1960), Bellman-Kalaba-Kotkin (1963), Peterson (1964)] for reducing storage requirements in the fundamental recursions for the minimum-return function in dynamic programming is to approximate this function by linear combinations of a relatively small number of fixed functions. To describe the approach simply, we suppose that we are using dynamic programming to solve a control problem and that the state variable, x, is one-dimensional and has been normalized so that $|x| \leq 1$; the problem we are treating could be, for example, that of minimizing $\Phi[x(T)]$ subject to $\frac{dx}{dt} = G(x,y)$ on $0 \leq t \leq 1$, $x(0) = a$, with y restricted say to $c \leq y \leq d$. Using a discrete time-step of Δ, the recurrence formula of dynamic programming becomes [Peterson (1964)]

$$f_{k+1}(x) \equiv \min_y \{f_k[x+G(x,y)\Delta]\}, \text{ for all } x \text{ in } [-1,1], \tag{1.1}$$

with

$$f_1(x) \equiv \min_y \{\Phi[x+G(x,y)\Delta]\} \tag{1.2}$$

The storage problem here is that we must store $f_k(x)$ at an enormous number of grid points x; the situation is even worse when x is multidimensional.

A standard way of reducing the storage requirements is to approximate each minimum-return function f_k as a linear combination of fixed functions $\{\varphi_i\}_1^M$, that is,

$$(1.3) \qquad f_k(x) \sim \sum_{i=1}^{M} a_{ik}\varphi_i(x)$$

We then only need store the M coefficients $\{a_{ik}\}_{i=1}^{M}$ to represent $f_k(x)$ for all x; if M is not terribly large then this results in a considerable storage saving. Clearly the storage savings depend essentially only on M and not on the form of the $\{\varphi_i\}$. How then should we choose among the various $\{\varphi_i\}$? Clearly we should consider questions of the approximation accuracy of the linear combinations of the $\{\varphi_i\}$, and we should consider the effort involved in computing with such approximations. Therefore in this paper we address ourselves to these questions of accuracy and efficiency. We use these two criteria for comparing the method based on spline approximations with the standard one [Bellman-Kalaba (1960), Bellman-Kalaba-Kotkin (1963), Peterson (1964)] using polynomial approximations.

Before we proceed to this, we state in general how one uses this general approach to approximate the fundamental recursion given by Equations 1.1 and 1.2. We choose to approximate via interpolation at given points $\{x_i\}_1^M$. We give a rough outline of the process:

Step 1: Perform whatever initial computations can be used to simplify later steps;

Step 2: Compute $\bar{f}_1(x_i) = \min_y \{\Phi[x_i + G(x_i,y)\Delta]\}$, $1 \le i \le M$;

Step 3: Set $k = 1$;

Step 4: Compute the coefficients $\{a_{jk}\}$ so that $\sum_{j=1}^{M} a_{jk}\varphi_j(x_i) = \bar{f}_k(x_i)$, $1 \le i \le M$;

Step 5: Compute $\bar{f}_{k+1}(x_i) = \min_y \left\{ \sum_{j=1}^{M} a_{jk}\varphi_j[x_i + G(x_i,y)\Delta] \right\}$, $1 \le i \le M$;

Step 6: Increase k by 1 and return to Step 4.

The above algorithm is, of course, terminated when $k+1 = N = \frac{1}{\Delta}$.

2. Polynomial Approximation

It is widely recommended [Bellman-Kalaba-Kotkin (1963), Peterson (1964), Bellman (1971)] that one choose an orthonormal set of functions $\{\varphi_i\}$ such as trigonometric functions or classical orthogonal polynomials. To be precise, we follow the suggestion of [Peterson (1964)] and assume that φ_i is a classical orthogonal polynomial of degree i-1, such as

$$\varphi_i(x) = T_{i-1}(x) \equiv \cos[(i-1)\arccos x], \tag{2.1}$$

the classical Chebyshev polynomials. Thus each f_k is approximated by a polynomial of degree M-1. If f_k has a bounded r-th derivative, then f_k can be approximated by such polynomials to degree $\mathcal{O}\left(\frac{1}{M^r}\right)$ as M increases; this answers the question of the accuracy attainable by this choice of $\{\varphi_i\}$ so we turn our attention to matters of efficiency.

By choosing the interpolation points $\{x_i\}_1^M$ as the zeros of φ_{M+1}, for example $x_i = \cos\frac{(2i-1)\pi}{2M}$ in the case of Equation 2.1, we can exploit the orthogonality of the $\{\varphi_i\}$ in the algorithm of Section 1. In particular, the computation of the a_{jk} in Step 4 reduces to

$$a_{jk} = \beta_j \sum_{i=1}^{M} \bar{f}_k(x_i)\varphi_j(x_i), \qquad 1 \le j \le M \tag{2.2}$$

for some constants β_j depending only on the set $\{\varphi_i\}$. Thus all the coefficients a_{jk}, $1 \le j \le M$, for fixed k, can be evaluated in about M^2 multiplications and M^2 additions.

Whatever

Whatever method used to implement Step 5, certainly for each y we must evaluate $\sum_{j=1}^{M} a_{jk}\varphi_j[x_i+G(x_i,y)\Delta]$. By careful exploitation of the usual three-term recursion among orthogonal polynomials [Cheney (1966), Clenshaw (1962)], we can evaluate this expression in about M multiplications and M additions for each i and each y. To be explicit, if we have to evaluate at Q different points y, Step 5 will require about QM^2 multiplications and QM^2 additions.

Since we see that the costs in Step 2 are independent of $\{\varphi_i\}$ and since the costs in Step 1 for this case are just the evaluations of $\{x_i\}$, we find that our total costs for each value of k, $1 \le k \le N-1$, is about $(Q+1)M^2$ multiplications and additions. Since generally Q is much greater than one, the following is valid:

(2.3) degree M polynomial approximation costs on the order of NQM^2 operations and yields $\sigma\left(\frac{1}{M^r}\right)$ accuracy for return functions having a bounded r-th derivative.

3. Spline Approximation

We now consider approximation by splines of order r (or degree $< r$) having joints at the interpolation points $\{x_i\}_1^M$. For convenience we assume that the x_i are equally spaced in $[-1,1]$, so that $x_i = -1 + (i-1)h$ with $h = \frac{2}{M-1}$. Then such a spline S_r is an $(r-2)$-times continuously differentiable function on $[-1,1]$ which equals a polynomial of degree not greater than $(r-1)$ in each interval (x_i,x_{i+1}) for $1 \le i \le M-1$. For information on splines, see, for example, [Ahlberg, et al. (1967), Greville (1969), Schoenberg (1970), Schultz (1973)]. If f_k has a bounded r-th derivative then it can be approximated by such a spline S_r to an accuracy of $\sigma\left(\frac{1}{M^r}\right)$ just as for polynomials; since the dimension of this

spline space is $M+r-2$ and we do not usually take r to be large, we see that splines of order r give essentially the same approximation power as polynomials with roughly the same number of free parameters. Much more importantly, however, we can prepresent each spline as a linear combination of $M+r-2$ special B-splines φ_i having the property that they vanish everywhere outside the interval $[x_i, x_{i+r}]$. Thus in evaluating any function $\sum a_j\varphi_j(x)$ no more than r different B-splines φ_j can be non-zero at each x; since each B-spline can be efficiently evaluated [de Boor (1971, 1972), Cox (1971)] at a cost independent of M we see that the cost of evaluating $\sum a_j\varphi_j(x)$ is a small constant C independent of M. One can show that C is roughly $\frac{3}{2}r^2$ [de Boor (1971, 1972)]. Therefore in Step 5 of the algorithm in Section 1 we need perform only CMQ operations as opposed to M^2Q for polynomials.

To perform Step 4 we need to be able to interpolate by splines in the B-spline representation, that is, to solve $\sum_j a_j\varphi_j(x_i) = f(x_i)$ for the a_j. Actually one usually interpolates at an additional $r-2$ points so as to determine uniquely the spline; this does not disturb the structure of the system of equations. Because of the small supports of the B-splines we see that this system of equations is essentially r-banded and hence can be decomposed into a product of banded lower- and upper-triangular matrices in the order of M operations. Once we do this decomposition in Step 1, we can solve the interpolation problems of Step 4 in the order of M operations using this decomposition.

In summary, we see that there is a small integer C such that

(3.1) degree r spline approximation with M joints costs on the order of $CMQM$ operations and yields $O\left(\frac{1}{M^r}\right)$ accuracy for return functions having a bounded r-th derivative.

This should be compared with Equation 2.3 where we see that polynomial approximation with the same accuracy and the same storage requirements involves more work by a factor of roughly M, certainly a significant increase for even moderate M. Clearly practitioners should seriously consider using spline approximation.

4. Higher Dimensions

Similar approximations can be employed in higher dimensions by using tensor products: in two dimensions, for example, we could approximate via $f(u,v) \sim \sum_{i,j} a_{ij}\varphi_i(u)\varphi_j(v)$. In p dimensions one sees that polynomial approximation costs roughly QNM^{2p} compared with $QQNM^{p}$ for spline approximation.

5. References

1. Ahlberg, J. H., E. N. Nilson, J. L. Walsh (1967), The Theory of Splines and Their Applications, Academic Press, New York.

2. Bellman, R. (1971), Introduction to the Mathematical Theory of Control Processes, Volume II: Nonlinear Processes, Academic Press, New York.

3. Bellman, R., S. Dreyfus (1962), Applied Dynamic Programming, Princeton Univ. Press.

4. Bellman, R., R. Kalaba (1960), "Reduction of dimensionality, dynamic programming, and control processes," P-1964, The RAND Corporation, Santa Monica, California.

5. Bellman, R., R. Kalaba, B. Kotkin (1963), "Polynomial approximation--a new computational technique in dynamic programming: allocation processes," Math. Comp., vol. 17, 155-161.

6. de Boor, C. (1971), "Subroutine package for calculating with B-splines," Los Alamos Scientific Lab. Report LA-4728-MS.

7. de Boor, C. (1972), "On calculating with B-splines," J. Approx. Th., vol. 6, 50-62.

8. Cheney, E. W. (1966), Introduction to Approximation Theory, McGraw-Hill, New York.

9. Clenshaw, C. W. (1962), "Chebyshev series for mathematical functions," National Physical Laboratory Math. Tables, vol. 5, Teddington, England.

10. Cox, M. G. (1971), "The numerical evaluation of B-splines," National Physical Laboratory Report DNAC4, Teddington, England.

11. Greville, T.N.E., editor (1969), Theory and Applications of Spline Functions, Academic Press, New York.

12. Peterson, E. L. (1964), "Optimum multivariable control," in Applied Combinatorial Mathematics, E. F. Beckenbach (ed.), Wiley, New York, 253-283.

13. Schoenberg, I. J. (1970), Approximations with Special Emphasis on Spline Functions, Academic Press, New York.

14. Schultz, M. H. (1973), Spline Analysis, Prentice Hall, Englewood Cliffs, New Jersey.

unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified

ORIGINATING ACTIVITY (Corporate author) The University of Texas at Austin	2a. REPORT SECURITY CLASSIFICATION unclassified 2b. GROUP

3. REPORT TITLE

SPLINES AND EFFICIENCY IN DYNAMIC PROGRAMMING

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Center for Numerical Analysis Report

5. AUTHOR(S) (First name, middle initial, last name)

James W. Daniel

6. REPORT DATE September 1973	7a. TOTAL NO. OF PAGES 8	7b. NO. OF REFS 14
8a. CONTRACT OR GRANT NO. N00014-67-A-1026-0015 b. PROJECT NO.	9a. ORIGINATOR'S REPORT NUMBER(S) CNA-78 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Mathematics Branch, Office of Naval Research, Washington, D. C.

13. ABSTRACT

It is shown how one can use splines, represented in the B-spline basis, to reduce the difficulties of large storage requirements in dynamic programming via approximations to the minimum-return function without the inefficiency associated with using polynomials to the same end.

DD FORM 1473 (PAGE 1)

S/N 0102-007-6801

unclassified
Security Classification