Dong Yu and
Li Deng

Solving Nonlinear Estimation Problems Using Splines

We describe the use of splines for solving nonlinear model estimation problems, in which nonlinear functions with unknown shapes and values are involved, by converting the nonlinear estimation problems into linear ones at a higher-dimensional space. This contrasts with the typical use of the splines [1]–[3] for function interpolation where the functional values at some input points are given and the values corresponding to other input points are sought for via interpolation. The technique described in this column applies to arbitrary nonlinear estimation problems where one or more one-dimensional nonlinear functions are involved and can be extended to cases where higher-dimensional nonlinear functions are used.

The benefit of using the approach described here is obvious. Many real-world systems can only be appropriately modeled with nonlinear functions, while the estimation problem is much simpler if only linear functions are involved. It is thus highly desirable if a nonlinear estimation problem can be transformed into a linear estimation problem at a different space. In this column we use the cubic spline (i.e., piecewise third-order polynomials) [1], [2] to illustrate the technique. However, the same approach can be used with other types of spline as illustrated at the end. We demonstrate the applications of the technique in signal processing and pattern recognition with an example.

RELEVANCE

The topics presented here extend the standard cubic spline interpolation algorithms by finding a direct relationship between the interpolated values and those at the spline knots (i.e., the given or to be estimated input/output points) for an arbitrary nonlinear function. This direct relationship can be formulated as an inner product of location-dependent weights and the values at the knots. The approach presented here may find practical applications within pattern recognition, classification, system combination, speech recognition, and signal processing. Some existing applications are briefly discussed.

PREREQUISITES

The prerequisites consist of basic calculus and basic linear algebra. Optimization techniques could also be useful but not necessary.

BACKGROUND

Splines are piecewise or multiple-segment functions with pieces or segments connected smoothly with each other, where the connecting points are called knots or control points. Splines are typically used to approximate the values of a nonlinear function $y = f(x)$ within the range $[x_1, x_N]$ by interpolating the values at knots $\{(x_i, y_i) | i = 1, \ldots, N; x_i < x_{i+1}\}$. Spline interpolation has two properties that make it attractive: the interpolation value depends only on the nearby knots, and the interpolation accuracy can be improved by increasing the number of knots.

A widely used spline is the cubic spline in which a third-order polynomial is used to interpolate adjacent knots

$$y(x) = f_i(x) = s_{i,0} + s_{i,1}(x - x_i) + s_{i,2}(x - x_i)^2 + s_{i,3}(x - x_i)^3 \quad \forall x \in [x_i, x_{i+1}],\ i = 1, 2, \ldots, N-1 \tag{1}$$

with the constraint that the function is continuous in the first- and second-order derivatives at all knots

$$y(x_i) = y_i \quad \text{for } i = 1, 2, \ldots, N \tag{2}$$

$$y_{i+1} = y(x_{i+1}) = f_i(x_{i+1}) = f_{i+1}(x_{i+1}) \quad \text{for } i = 1, 2, \ldots, N-1 \tag{3}$$

$$y'_{i+1} = y'(x_{i+1}) = f'_i(x_{i+1}) = f'_{i+1}(x_{i+1}) \quad \text{for } i = 1, 2, \ldots, N-1 \tag{4}$$

$$y''_{i+1}(x_{i+1}) = y''(x_{i+1}) = f''_i(x_{i+1}) = f''_{i+1}(x_{i+1}) \quad \text{for } i = 1, 2, \ldots, N-1 \tag{5}$$

It can be easily verified that the above formulation can be rewritten as [1], [2]

$$y(x) = ay_j + by_{j+1} + cy''_j + dy''_{j+1}, \tag{6}$$

where $[x_j, x_{j+1}]$ is the segment to which the input value x belongs

$$a = \frac{x_{j+1} - x}{x_{j+1} - x_j}, \tag{7}$$

$$b = 1 - a = \frac{x - x_j}{x_{j+1} - x_j}, \tag{8}$$

$$c = \frac{1}{6}(a^3 - a)(x_{j+1} - x_j)^2, \text{ and} \tag{9}$$

$$d = \frac{1}{6}(b^3 - b)(x_{j+1} - x_j)^2 \tag{10}$$

are interpolation parameters, and y''_j is the second-order derivative of y with respect to x at point x_j at the boundary between two adjacent segments.

Let's verify the equivalence of (1) and (6) by checking the boundary conditions. From (6) we get

$$\frac{dy}{dx} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{3a^2 - 1}{6}(x_{j+1} - x_j) \times y''_j + \frac{3b^2 - 1}{6}(x_{j+1} - x_j)y''_{j+1}, \tag{11}$$

Digital Object Identifier 10.1109/MSP.2009.932793

1053-5888/09/$25.00©2009IEEE

$$\frac{d^2y}{dx^2} = ay_j'' + by_{j+1}''. \tag{12}$$

Given $x = x_j$, from the right segment we have

$$a = \frac{x_{j+1} - x_j}{x_{j+1} - x_j} = 1, \tag{13}$$

$$b = 1 - a = 0, \tag{14}$$

$$c = \frac{1}{6}(a^3 - a)(x_{j+1} - x_j)^2 = 0, \tag{15}$$

and

$$d = \frac{1}{6}(b^3 - b)(x_{j+1} - x_j)^2 = 0, \tag{16}$$

and so

$$y(x_j) = ay_j + by_{j+1} + cy_j'' + dy_{j+1}'' = y_j, \tag{17}$$

$$y'(x_j) = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)\, y_j'' - \frac{1}{6}(x_{j+1} - x_j)\, y_{j+1}'', \text{ and} \tag{18}$$

$$y''(x_j) = ay_j'' + by_{j+1}'' = y_j''. \tag{19}$$

Similarly, from the left segment we have

$$a = \frac{x_j - x_j}{x_j - x_{j-1}} = 0, \tag{20}$$

$$b = 1 - a = 1, \tag{21}$$

$$c = \frac{1}{6}(a^3 - a)(x_{j+1} - x_j)^2 = 0, \tag{22}$$

and

$$d = \frac{1}{6}(b^3 - b)(x_{j+1} - x_j)^2 = 0, \tag{23}$$

and so

$$y(x_j) = ay_{j-1} + by_j + cy_{j-1}'' + dy_j'' = y_j, \tag{24}$$

$$y'(x_j) = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})\, y_{j-1}'' + \frac{1}{3}(x_j - x_{j-1})\, y_j'', \tag{25}$$

$$y''(x_j) = ay_{j-1}'' + by_j'' = y_j''. \tag{26}$$

Comparing (17) to (24) and (19) to (26), we can clearly see that the function and the second-order derivatives are continuous across segments. The continuity of the first-order derivative can be forced by choosing appropriate y_j'' s.

Note that (6) cannot be solved without additional constraints. These additional constraints are typically provided as boundary conditions. Under the boundary condition one, the first-order derivatives of y over x at knots x_1 and x_N are provided as z_1 and z_N, respectively. Under the boundary condition two, the second-order derivatives of y over x at knots x_1 and x_N are set to zero. The splines with boundary condition two are often called natural splines.

In the next several paragraphs, we further rewrite (6) by replacing the second-order derivatives with functions of knot values so that the values of the function only depend on the knots. The reason to rewrite (6) to the new formula is to find a direct relationship between the function values and the knots so that we may convert a function estimation problem into a knots estimation problem. This new formulation forms the basis for our objective of converting nonlinear estimation problems into linear ones at a higher-dimensional space.

SOLUTION UNDER BOUNDARY CONDITION ONE

Under boundary condition one the first-order derivatives at boundaries are provided as z_1 and z_N respectively. By enforcing the first-order derivative continuity constraints, we obtain $\mathbf{y}'' = [y_1'' \;\; \ldots \;\; y_j'' \;\; \ldots \;\; y_N'']^t$ as the solution to the linear system of equations

$$\mathbf{A}_1 \mathbf{y}'' = \mathbf{B}_1, \tag{27}$$

where $\mathbf{A}_1$ is defined in (28) at the bottom of the page and

$$\mathbf{B}_1 = \begin{bmatrix} z_1 - \dfrac{y_2 - y_1}{x_2 - x_1} \\ \dfrac{y_3 - y_2}{x_3 - x_2} - \dfrac{y_2 - y_1}{x_2 - x_1} \\ \vdots \\ \dfrac{y_{j+1} - y_j}{x_{j+1} - x_j} - \dfrac{y_j - y_{j-1}}{x_j - x_{j-1}} \\ \vdots \\ \dfrac{y_N - y_{N-1}}{x_N - x_{N-1}} - \dfrac{y_{N-1} - y_{N-2}}{x_{N-1} - x_{N-2}} \\ z_N - \dfrac{y_N - y_{N-1}}{x_N - x_{N-1}} \end{bmatrix}. \tag{29}$$

Note that $\mathbf{B}_1$ can be rewritten as

$$\mathbf{B}_1 = \mathbf{C}_1 + \mathbf{D}_1 \mathbf{y} \tag{30}$$

where $\mathbf{C}_1 = [z_i \;\; 0 \; \cdots \; 0 \;\; z_N]^T$, $\mathbf{y} = [y_1 \;\; y_2 \; \cdots \; y_N]^T$, and $\mathbf{D}_1$ is defined in (31) at the top of the next page.

$$\mathbf{A}_1 = \begin{bmatrix} -\dfrac{x_2 - x_1}{3} & -\dfrac{x_2 - x_1}{6} & 0 & \cdots & \cdots & \cdots & 0 \\ \dfrac{x_2 - x_1}{6} & \dfrac{x_3 - x_1}{3} & \dfrac{x_3 - x_2}{6} & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & 0 & \dfrac{x_j - x_{j-1}}{6} & \dfrac{x_{j+1} - x_{j-1}}{3} & \dfrac{x_{j+1} - x_j}{6} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & \dfrac{x_{N-1} - x_{N-2}}{6} & \dfrac{x_N - x_{N-2}}{3} & \dfrac{x_N - x_{N-1}}{6} \\ 0 & \cdots & \cdots & \cdots & 0 & \dfrac{x_N - x_{N-1}}{6} & \dfrac{x_N - x_{N-1}}{3} \end{bmatrix} \tag{28}$$

$$
\mathbf{D}_1 = \begin{bmatrix}
\frac{1}{x_2-x_1} & -\frac{1}{x_2-x_1} & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{1}{x_2-x_1} & -\frac{1}{x_2-x_1}-\frac{1}{x_3-x_2} & \frac{1}{x_3-x_2} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{1}{x_j-x_{j-1}} & \frac{1}{x_j-x_{j-1}}-\frac{1}{x_{j+1}-x_j} & \frac{1}{x_{j+1}-x_j} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{x_{N-1}-x_{N-2}} & -\frac{1}{x_{N-1}-x_{N-2}}-\frac{1}{x_N-x_{N-1}} & \frac{1}{x_N-x_{N-1}} \\
0 & \cdots & \cdots & \cdots & 0 & \frac{1}{x_N-x_{N-1}} & -\frac{1}{x_N-x_{N-1}}
\end{bmatrix} \tag{31}
$$

If the knots are evenly distributed, i.e., $x_{j+1}-x_j = h$ for any j, $\mathbf{A}_1$ and $\mathbf{D}_1$ can be simplified to

$$
\mathbf{A}_1 = \begin{bmatrix}
-\frac{h}{3} & -\frac{h}{6} & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{h}{6} & \frac{2h}{3} & \frac{h}{6} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{h}{6} & \frac{2h}{3} & \frac{h}{6} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{h}{6} & \frac{2h}{3} & \frac{h}{6} \\
0 & \cdots & \cdots & \cdots & 0 & \frac{h}{6} & \frac{h}{3}
\end{bmatrix}
= h \begin{bmatrix}
-\frac{1}{3} & -\frac{1}{6} & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\
0 & \cdots & \cdots & \cdots & 0 & \frac{1}{6} & \frac{1}{3}
\end{bmatrix}. \tag{32}
$$

SOLUTION UNDER BOUNDARY CONDITION TWO

Under boundary condition two, the second-order derivatives at the boundaries equal to zero, i.e., $y''_1 = y''_N = 0$. Under this condition and by forcing the first-derivative continuity constraints, we have

$$
\mathbf{D}_1 = \begin{bmatrix}
\frac{1}{h} & -\frac{1}{h} & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{1}{h} & -\frac{2}{h} & \frac{1}{h} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{1}{h} & -\frac{2}{h} & \frac{1}{h} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{h} & -\frac{2}{h} & \frac{1}{h} \\
0 & \cdots & \cdots & \cdots & 0 & \frac{1}{h} & -\frac{1}{h}
\end{bmatrix}
$$

$$
\mathbf{A}_2 = \begin{bmatrix}
\frac{x_2-x_1}{6} & 0 & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{x_2-x_1}{6} & \frac{x_3-x_1}{3} & \frac{x_3-x_2}{6} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{x_j-x_{j-1}}{6} & \frac{x_{j+1}-x_{j-1}}{3} & \frac{x_{j+1}-x_j}{6} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{x_{N-1}-x_{N-2}}{6} & \frac{x_N-x_{N-2}}{3} & \frac{x_N-x_{N-1}}{6} \\
0 & \cdots & \cdots & \cdots & 0 & 0 & \frac{x_N-x_{N-1}}{6}
\end{bmatrix} \tag{35}
$$

$$
D_2 = \begin{array}{ccccccc}
0 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{1}{x_2 - x_1} & -\frac{1}{x_2 - x_1} - \frac{1}{x_3 - x_2} & \frac{1}{x_3 - x_2} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{1}{x_j - x_{j-1}} & -\frac{1}{x_j - x_{j-1}} - \frac{1}{x_{j+1} - x_j} & \frac{1}{x_{j+1} - x_j} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{x_{N-1} - x_{N-2}} & -\frac{1}{x_{N-1} - x_{N-2}} - \frac{1}{x_N - x_{N-1}} & \frac{1}{x_N - x_{N-1}} \\
0 & \cdots & \cdots & \cdots & 0 & 0 & 0
\end{array} \tag{38}
$$

$$
= \frac{1}{h} \begin{bmatrix}
1 & -1 & 0 & \cdots & \cdots & \cdots & 0 \\
1 & -2 & 1 & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & 1 & -2 & 1 & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & 1 & -2 & 1 \\
0 & \cdots & \cdots & \cdots & 0 & 1 & -1
\end{bmatrix}. \tag{33}
$$

$$
A_2 y'' = B_2, \tag{34}
$$

where A_2 is defined in (35) on the previous page and

$$
B_2 = \begin{array}{c}
0 \\
\frac{y_3 - y_2}{x_3 - x_2} - \frac{y_2 - y_1}{x_2 - x_1} \\
\vdots \\
\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}} \\
\vdots \\
\frac{y_N - y_{N-1}}{x_N - x_{N-1}} - \frac{y_{N-1} - y_{N-2}}{x_{N-1} - x_{N-2}} \\
0
\end{array}. \tag{36}
$$

Note that B_2 can be rewritten as

$$
B_2 = D_2 y \tag{37}
$$

where D_2 is defined in (38) at the top of the page.

If the knots are evenly distributed, A_2 and D_2 can be simplified to

$$
A_2 = \begin{array}{ccccccc}
\frac{h}{6} & 0 & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{h}{6} & \frac{2h}{3} & \frac{h}{6} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{h}{6} & \frac{2h}{3} & \frac{h}{6} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{h}{6} & \frac{2h}{3} & \frac{h}{6} \\
0 & \cdots & \cdots & \cdots & 0 & 0 & \frac{h}{6}
\end{array}
$$

$$
= h \begin{array}{ccccccc}
\frac{1}{6} & 0 & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\
0 & \cdots & \cdots & \cdots & 0 & 0 & \frac{1}{6}
\end{array}. \tag{39}
$$

THE UNIFIED FORM OF SOLUTIONS

If we choose $z_1 = z_N = 0$ in boundary condition one, then the two solutions discussed above can be written in the unified form

$$
D_2 = \begin{array}{ccccccc}
0 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\
\frac{1}{h} & -\frac{2}{h} & \frac{1}{h} & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & \frac{1}{h} & -\frac{2}{h} & \frac{1}{h} & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & \frac{1}{h} & -\frac{2}{h} & \frac{1}{h} \\
0 & \cdots & \cdots & \cdots & 0 & 0 & 0
\end{array}
$$

$$
= \frac{1}{h} \begin{bmatrix}
0 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\
1 & -2 & 1 & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\cdots & 0 & 1 & -2 & 1 & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & 1 & -2 & 1 \\
0 & \cdots & \cdots & \cdots & 0 & 0 & 0
\end{bmatrix} \tag{40}
$$

$$
A_m y'' = D_m y, \tag{41}
$$

where $m = 1$ or 2. Thus

$$
y'' = A_m^{-1} D_m y. \tag{42}
$$

Substituting (42) into (6) to eliminate the dependency on y_j'' and $y_{(j+1)}''$, we have

$$
\begin{aligned}
y &= a y_j + b y_{j+1} + c y_j'' + d y_{j+1}'' \\
&= (E_x + F_x A_m^{-1} D_m) y,
\end{aligned} \tag{43}
$$

where

$$\mathbf{E}_x = \begin{bmatrix} 0 & \cdots & a_j & b_{j+1} & \cdots & 0 \end{bmatrix}, \quad (44)$$

$$\mathbf{F}_x = \begin{bmatrix} 0 & \cdots & c_j & d_{j+1} & \cdots & 0 \end{bmatrix}. \quad (45)$$

Note that since a, b, c, d are functions of x, $\mathbf{E}_x$ and $\mathbf{F}_x$ are also functions of x. It is obvious from (43) that

$$\frac{dy}{d\mathbf{y}} = (\mathbf{E}_x + \mathbf{F}_x\mathbf{A}_m^{-1}\mathbf{D}_m)^T \quad (46)$$

Note that $\mathbf{A}_m$ is tridiagonal, and its inverse can be obtained in $O(N)$ steps. If evenly distributed knots are used, $\mathbf{A}_m^{-1}\mathbf{D}_m$ only needs to be evaluated once and may be precalculated by taking h outside of the equations.

As a summary, we have shown that the interpolated value y can be estimated as a linear interpolation of the values at knots $\mathbf{y}$ as

$$y = \mathbf{a}_x^T\mathbf{y}, \quad \text{where } \mathbf{a}_x = (\mathbf{E}_x + \mathbf{F}_x\mathbf{A}_m^{-1}\mathbf{D}_m)^T. \quad (47)$$

Equation (47) indicates that if we want to estimate a nonlinear function over some variables x, we can simplify the problem by converting it to estimating N knot values $\mathbf{y}$. We will use an example to show how it can be used to solve model estimation problems where nonlinear functions are involved.

APPLICATION EXAMPLE

This example draws from the authors' recent work on applying the result of (47) to a pattern recognition problem. It was shown in [4] that for continuous variables with distribution constraints, the solution to the popular maximum entropy model in pattern recognition has the form of

$$p(c|x) = \frac{1}{Z_\lambda(x)} \exp\left(\sum_{i\in\{\text{continuous}\}} \lambda_i(f_i(x,c)) f_i(x,c) + \sum_{j\in\{\text{binary}\}} \lambda_j f_j(x,c)\right) \quad (48)$$

where c is the class index, $Z_\lambda(x)$ is the normalization term, and $\lambda_i(f_i(x,c))$ is a nonlinear function of the extracted features $f_i(x,c)$. Note that this nonlinear function can have any shape. By using the technique we just described we can approximate $\lambda_i(f_i(x,c))$ as

$$\lambda_i(f_i) \cong a^T(f_i)\lambda_i, \quad (49)$$

which further gives

$$\begin{aligned} \lambda_i(f_i)f_i &\cong a^T(f_i)\lambda_i f_i \\ &= [a^T(f_i)f_i]\lambda_i \\ &= \sum_k \lambda_{ik}[a_k(f_i)f_i]. \end{aligned} \quad (50)$$

By substituting (50) into (48), we can convert (48) to

$$p(c|x) = \frac{1}{Z_\lambda(x)} \exp\left(\sum_{i\in\{\text{continuous}\}k} \lambda_{ik} f_{ik}(x,c) + \sum_{j\in\{\text{binary}\}} \lambda_j f_j(x,c)\right), \quad (51)$$

where

$$f_{ik}(x,c) = a_k(f_i(x,c)) f_i(x,c). \quad (52)$$

Equation (51) is in the standard log-linear form at a higher-dimensional space and can be solved with existing algorithms for the maximum entropy models.

The same technique can also be applied to other areas such as nonlinear system combination and speech recognition. For example, it has also been successfully applied to the variable parameter hidden Markov model [5].

SUMMARY

In this column we have shown an approach to solving nonlinear estimation problems by casting the problem into a linear estimation problem at a higher dimensional space using the cubic spline interpolation technique. We demonstrated an example of applying this technique.

This technique is easy to implement and can be applied to many different problems. It is robust to data sparseness problems since the value at a particular point is determined by many surrounding points. This technique, however, can be expensive if applied to functions with more than two variables. In addition, the number of knots used in the solution needs to be determined empirically based on the amount of training data available and is typically tuned with a development set.

Although we have used the cubic spline as an example, the same technique can be implemented using other polynomial splines such as linear and quadratic splines. The key idea is to find a direct relationship between the interpolated value and the knots in the spline so that the optimization problem can be converted to the problem of finding the best knots.

AUTHORS

Dong Yu (dongyu@microsoft.com) is a researcher in the Microsoft Speech Research Group, Redmond, Washington.

Li Deng (deng@microsoft.com) is a principal researcher in the Microsoft Speech Research Group, Redmond, Washington.

REFERENCES

[1] R. H. Bartels, J. C. Beatty, and B. A. Barsky, "Hermite and cubic spline interpolation," in *An Introduction to Splines for Use in Computer Graphics and Geometric Modelling*. San Francisco, CA: Morgan Kaufmann, pp. 9–17, Ch. 3, 1998.

[2] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, "Cubic spline interpolation," in *Numerical Recipes in FORTRAN: The Art of Scientific Computing*, 2nd ed. Cambridge, England: Cambridge Univ. Press, 1992, pp. 107–110, §3.3.

[3] M. Unser, "Splines—A perfect fit for signal and image processing," *IEEE Signal Processing Mag.*, vol. 16, pp. 22–38, Nov. 1999.

[4] D. Yu, L. Deng, and A. Acero, "The maximum entropy model with continuous features," in *Proc. NIPS 2008 Workshops*, Whistler, Canada. [Online]. Available: http://research.microsoft.com/en-us/um/people/xiaohe/NIPS08/PaperAccepted/nips2008WSL1_01.pdf

[5] D. Yu, L. Deng, Y. Gong, and A. Acero, "Discriminative training of variable-parameter HMMs for noise robust speech recognition," in *Proc. Interspeech 2008*, Brisbane, Australia, pp. 285–288.

[SP]