OpenGamma

OpenGamma Quantitative Research

Piecewise Polynomial Interpolations

Yukinori Iwashita
yuki@opengamma.com

Abstract

This note reviews interpolation algorithms based on piecewise polynomial functions. We first introduce two basic interpolations: piecewise linear interpolation and cubic spline interpolation. Since the latter suffers from overshooting problems and nonlocal dependencies, we discuss alternative interpolation methods aiming at shape-preserving of given data and avoiding the nonlocality. Finally two-dimensional extensions of the spline interpolations are considered.

Contents

1 Introduction

Interpolation is a methodology of constructing new points between known data points. We often confront situations where only limited amount of data is accessible and it is necessary to estimate values between two consecutive given data points. In finance, as only a finite set of securities are traded in financial markets, it is a key role to construct a sensible curve or surface from discrete observable quantities. For example, the absence of arbitrage in yield curve construction requires that an interpolant $r(t)$ should satisfy

$$\frac{\partial\left(r(t)t\right)}{\partial t} \geq 0\,, \tag{1}$$

where the price of a zero coupon bond for a single payment at maturity t is $\exp(-r(t)t)$, whereas the conditions for an arbitrage-free implied volatility surface are

$$\frac{\partial C(t,K)}{\partial t} \geq 0\,, \qquad \frac{\partial^2 C(t,K)}{\partial K^2} \geq 0\,, \tag{2}$$

where C is the call option price for strike K (see [Hom11], [HW06] and references therein).

The purpose of this note is to review interpolation methods based on piecewise polynomial functions which are frequently used to estimate a structure hidden behind a set of market data. Any financial application of the interpolation methods is not described here and comprehensive review on piecewise polynomial interpolations in the context of yield curve construction is given in, e.g., [HW06], [Flo13] and see also [Whi12].

Let us first refine the interpolation problem. Given $N+1$ sets of data $(x_i, y_i) \in \mathbf{R}^2$ $(i = 0, \ldots, N)$ with the strictly increasing condition $x_0 < x_1 < \cdots x_N$, the one-dimensional interpolation problem is to find a function $f : [x_0, x_N] \to \mathbf{R}$ such that $f(x_i) = y_i$ for all i. For spline interpolation, the interpolant is expressed in terms of piecewise functions,

$$f(x) = f_i(x)\,, \quad x \in [x_i, x_{i+1}]\,, \tag{3}$$

where $f_i(x)$ is at least continuous everywhere in $[x_i, x_{i+1}]$.[1] Continuity condition should hold at every data point,

$$f_i(x_{i+1}) = f_{i+1}(x_{i+1})\,. \tag{4}$$

Smoothness of interpolation requires further conditions at the data points. Conditions for C^1 continuity at the grid points are

$$f_i'(x_{i+1}) = f_{i+1}'(x_{i+1})\,, \tag{5}$$

where the prime $'$ represents derivative with respect to x. In order to achieve C^2 continuity one should, in addition, impose continuity of the second derivative,

$$f_i''(x_{i+1}) = f_{i+1}''(x_{i+1})\,. \tag{6}$$

The interpolant which we discuss in this note is a polynomial function, that is, the piecewise function takes the following form,

$$f_i(x) = \sum_{k=0}^{n} a_k^i \left(x - x_i\right)^k\,. \tag{7}$$

[1] There are interpolation methods producing a curve with discontinuity. An example is a piecewise constant interpolation such as nearest neighbor interpolation. We do not consider the discontinuous cases in this note.

Hence the piecewise polynomial interpolation problem is to determine the coefficients a_k^i for all of the intervals such that the resulting interpolant has desirable properties.

Throughout our discussion on one-dimensional interpolations, we shall use the following notation for shifted coordinates, intervals and slopes,

$$\begin{aligned} &t = x - x_i\,, \quad x \in [x_i, x_{i+1}]\,, \\ &h_i = x_{i+1} - x_i\,, \qquad s_i = \frac{y_{i+1} - y_i}{h_i}\,, \end{aligned} \tag{8}$$

for $i = 0, 1, \ldots, N-1$.

This note is organised as follows.

We start discussion with piecewise linear interpolation in section 2, where two consecutive data points are interpolated by a straight line. Another basic interpolation method is cubic spline interpolation reviewed in section 3. This interpolator produces a C^2 curve once a data set (x_i, y_i) is given and the only ambiguity found at both the endpoints, x_0 and x_N, is fixed. We introduce three choices of the endpoint conditions. In section 4 interpolations based on Hermite polynomials are discussed. They require first derivative values as well as function values at the data points for the cubic case, whereas second derivative values are also required for quintic interpolations. The discussion includes Akima cubic interpolation in section 4.1, constrained cubic interpolation in section 4.2, monotone preserving interpolation in section 4.3, Hyman filters preserving nonnegativity in section 4.4 or monotonicity in section 4.5 and shape preserving C^2 cubic interpolation in section 4.6. Monotone convex interpolation designed for yield curve construction is reviewed in section 5. After introducing each one-dimensional interpolation algorithm, we apply the method to interpolate a data set (x_i, y_i) given in Table 1. Section 6 is devoted to introducing two-dimensional extensions of the one-dimensional piecewise polynomial interpolations. We close this note with brief comments on a couple of other interpolation methods which are also relevant in finance in section 7.

2 Linear Interpolation

The piecewise linear interpolation is piecewise straight lines connecting two consecutive data points. The interpolant in the interval $[x_i, x_{i+1}]$ is then

$$f_i(x) = y_i + s_i \cdot t\,, \tag{9}$$

where t and s_i are given in (8). Clearly the linear interpolation satisfies the condition (4). By construction this interpolation produces a C^0 line. As an example, let us interpolate the data in Table 1 by the piecewise linear function. The result is shown in Figure 1.

3 Cubic Spline Interpolations

There are many piecewise interpolations using cubic polynomials. Among them, cubic spline interpolation usually stands for a specific interpolation which produces C^2 cubic splines imposing the conditions (4), (5) and (6). Below we shall describe how the coefficients a_k^i in (7) are determined under these conditions.

Recalling the notation in (8) one finds that $f_i(x_i) = y_i$ and $f_i(x_{i+1}) = y_{i+1}$ for $i = 0, 1, \ldots, N-1$ imply

$$\begin{aligned} &a_0^i = y_i \,, \\ &a_0^i + h_i a_1^i + h_i^2 a_2^i + h_i^3 a_3^i = y_{i+1} \,, \end{aligned} \tag{10}$$

for $i = 0, 1, \ldots, N-1$ and the conditions (5), (6) are respectively expressed as

$$\begin{aligned} &a_1^i + 2h_i a_2^i + 3h_i^2 a_3^i = a_1^{i+1} \,, \\ &2a_2^i + 6h_i a_3^i = 2a_2^{i+1} \,, \end{aligned} \tag{11}$$

for $i = 0, 1, \ldots, N-2$. It is convenient to introduce new variables $m_i = f_i''(x_i)$. Then the equations to be solved are

$$h_i m_i + 2(h_i + h_{i+1}) m_{i+1} + h_{i+1} m_{i+2} = 6s_{i+1} - 6s_i \,, \tag{12}$$

for $i = 0, 1, \ldots, N-2$. Once m_i are determined by solving these equations, the coefficients can be computed by

$$a_1^i = s_i - \frac{h_i}{2} m_i - \frac{h_i}{6} (m_{i+1} - m_i) \,, \qquad a_2^i = \frac{m_i}{2} \,, \qquad a_3^i = \frac{m_{i+1} - m_i}{6h_i} \,, \tag{13}$$

and a_0^i are given by the first equation in (10). Since the cubic interpolation involves $4N$ parameters, 2 free parameters remain after imposing these conditions. There is no unique way to fix the remaining degrees of freedom, but common choices are $f'(x_0) = \alpha$, $f'(x_N) = \beta$ where α, β are constants (Clamped spline), $f''(x_0) = f''(x_N) = 0$ (Natural spline) and $f'''(x_0) = f'''(x_1)$, $f'''(x_N) = f'''(x_{N-1})$ (Not-A-Knot spline).

The Clamped spline conditions are rewritten in terms m_i,

$$\begin{aligned} &2h_0 m_0 + h_0 m_1 = 6\,(s_0 - \alpha) \,, \\ &h_{N-1} m_{N-1} + 2h_{N-1} m_N = 6\,(\beta - s_{N-1}) \,, \end{aligned} \tag{14}$$

and for the Natural spline we have

$$m_0 = m_N = 0 \,, \tag{15}$$

whereas the Not-A-Knot spline endpoint conditions are

$$\begin{aligned} &h_1\,(m_1 - m_0) = h_0\,(m_2 - m_1) \,, \\ &h_{N-1}\,(m_{N-1} - m_{N-2}) = h_{N-2}\,(m_N - m_{N-1}) \,. \end{aligned} \tag{16}$$

Hence the interpolation problem reduces into an $(N+1) \times (N+1)$ linear system with a band-diagonal matrix A,

$$Am = b \,, \tag{17}$$

where $m = (m_0, m_1, \cdots, m_N)^T$ and the matrix A and vector b are specified by (12) together

with the endpoint conditions (14), (15) or (16). For the Clamped spline we have

$$A = \begin{pmatrix} 2h_0 & h_0 & 0 & \cdots & \cdots & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & \\ 0 & h_1 & 2(h_1+h_2) & h_2 & 0 & \vdots \\ & 0 & h_2 & 2(h_2+h_3) & \cdots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \\ \vdots & & \cdots & 2(h_{n-3}+h_{n-2}) & h_{n-2} & 0 \\ & & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & \cdots & \cdots & 0 & h_{n-1} & 2h_{n-1} \end{pmatrix}, \tag{18}$$

and the Natural spline endpoint conditions yield

$$A = \begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & \\ 0 & h_1 & 2(h_1+h_2) & h_2 & 0 & \vdots \\ & 0 & h_2 & 2(h_2+h_3) & \cdots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \\ \vdots & & \cdots & 2(h_{n-3}+h_{n-2}) & h_{n-2} & 0 \\ & & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}, \tag{19}$$

and under the Not-A-Knot spline endpoint conditions the matrix A takes the form,

$$A = \begin{pmatrix} -h_1 & h_0+h_1 & -h_0 & 0 & \cdots & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & \\ 0 & h_1 & 2(h_1+h_2) & h_2 & 0 & \vdots \\ & 0 & h_2 & 2(h_2+h_3) & \cdots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \\ \vdots & & \cdots & 2(h_{n-3}+h_{n-2}) & h_{n-2} & 0 \\ & & 0 & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & \cdots & 0 & -h_{n-1} & h_{n-2}+h_{n-1} & -h_{n-2} \end{pmatrix}. \tag{20}$$

The vector b for the Clamped spline and the Natural spline is

$$b = \begin{pmatrix} 0 & s_1 - s_0 & \cdots & s_{n-1} - s_{n-2} & 0 \end{pmatrix}^T, \tag{21}$$

and the constants α, β should be inserted for the Not-A-Knot spline,

$$b = \begin{pmatrix} 6\,(s_0 - \alpha) & s_1 - s_0 & \cdots & s_{n-1} - s_{n-2} & 6\,(\beta - s_{N-1}) \end{pmatrix}^T. \tag{22}$$

For more detal, see e.g., [dB78].

We shall compare the results of the three interpolation methods described in this section on the data set given in Table 1 and $\alpha = \beta = 0$ for the Clamped Spline. Figure 2 illustrates

the curves for the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). The black dots correspond to the data points. This figure shows that the resulting interpolation functions, especially near the endpoints, are largely dependent on endpoint conditions. One can see a serious problem in these interpolation methods in $x \in [3.1, 5.1]$ where the spline functions take negative values although y values of the data are positive.

It is, in general, well known that the cubic spline interpolation functions sometimes exhibit unnatural wiggles and bumps and the interpolant can be negative even when all of the data points are positive, although the cubic spline interpolation is widely used due to its C^2 smoothness. Moreover, changing an input can alter the shape of piecewise polynomial functions far away from the changed data point meaning that the interpolation is nonlocal. There are primarily three ways to resolve these issues: relaxing C^2 continuity to C^1 continuity; using higher degree polynomial functions; introducing extra knots between two consecutive data points, which are addressed in the following sections.

4 Hermite Interpolations

The interpolation methods we have discussed so far require values of interpolants to be equal to y values of given data, $f(x_i) = y_i$ for all i. Hermite interpolation is to develop an interpolant equating derivative values of the interpolation function and given derivative values at data points, i.e.,

$$f(x_i) = y_i\,, \quad f'(x_i) = y_i'\,, \quad f''(x_i) = y_i''\,, \quad \cdots\,, \quad f^{(m)}(x_i) = y_i^{(m)}\,, \tag{23}$$

for $f \in C^m[x_0, x_N]$. For the cubic Hermite spline case the interpolant is computed once the first derivative values $f'(x_i) = y_i'$ are given at the data points while the quintic Hermite interpolation is obtained if $f''(x_i) = y_i''$ are given in addition.

Let $f_i = f(x_i)$ and $f_i' = f'(x_i)$, then the cubic Hermite interpolant is given by[2]

$$f_i(x) = \sum_{k=0}^{3} a_k^i \cdot t^k\,, \tag{26}$$

where t is given in (8) and

$$a_0^i = f_i\,, \qquad a_1^i = f_i'\,, \qquad a_2^i = \frac{3s_i - f_{i+1}' - 2f_i'}{h_i}\,, \qquad a_3^i = -\frac{2s_i - f_{i+1}' - f_i'}{h_i^2}\,. \tag{27}$$

Hence, for a given data set (x, y), the cubic Hermite interpolation is done if first derivative values f_i' are fixed. The resulting interpolant has a continuous first derivative for all $x \in \mathbf{R}$ and, if the given first derivative values are exact, the second derivative is also continuous.

[2] Formally, for $m = 1$ the Hermite polynomial with degree $2n + 1$ is given by

$$H_{2n+1}(x) = \sum_{j=0}^{n} f_j H_{n,j}(x) + \sum_{j=0}^{n} f_j' \hat{H}_{n,j}(x)\,, \tag{24}$$

where $H_{n,j}(x)$, $\hat{H}_{n,j}(x)$ are expressed in terms of the j-th Lagrange coefficient polynomial of degree n, $L_{n,j}(x)$,

$$H_{n,j}(x) = \big(1 - 2(x - x_j)L_{n,j}'(x)\big)\, L_{n,j}^2(x)\,, \qquad \hat{H}_{n,j}(x) = (x - x_j)\, L_{n,j}^2(x)\,. \tag{25}$$

Setting $n = 1$ (and the summations taken over $j = i, i + 1$) yields the form of a piecewise interpolation function in (26), (27). The Quintic interpolant (28), (29) is obtained similarly.

If the second derivative values at data points, $f_i'' = f''(x_i)$, are also computed, one can determine quintic Hermite interpolation,

$$f_i(x) = \sum_{k=0}^{5} a_k^i \cdot t^k \,, \tag{28}$$

where t is given in (8) and

$$\begin{aligned} &a_0^i = f_i \,, \quad a_1^i = f_i' \,, \quad a_2^i = \frac{f_i''}{2} \,, \quad a_3^i = \frac{f_{i+1}'' - 3f_i''}{2h_i} + 2\frac{5s_i - 3f_i' - 2f_{i+1}'}{h_i^2} \,, \\ &a_4^i = \frac{3f_i'' - 2f_{i+1}''}{2h_i^2} + \frac{8f_i' + 7f_{i+1}' - 15s_i}{h_i^3} \,, \quad a_5^i = \frac{f_{i+1}'' - f_i''}{2h_i^3} + 3\frac{2s_i - f_{i+1}' - f_i'}{h_i^4} \,. \end{aligned} \tag{29}$$

Generally the first and second derivatives are continuous for the quintic Hermite interpolation. Further smoothness is obtained if the derivative values are exact.

In the following subsections we shall review several methods determining the derivative values f_i', f_i''. If the derivative values are chosen locally, the cubic and quintic Hermite interpolations produce piecewise polynomials which only depend on given data at nearby mesh points.

4.1 Akima Cubic Interpolation

Akima proposed an algorithm determining f_i' by "geometric mean" [Aki70],

$$f_i' = \frac{|s_{i+1} - s_i|\, s_{i-1} + |s_{i-1} - s_{i-2}|\, s_i}{|s_{i+1} - s_i| + |s_{i-1} - s_{i-2}|} \,, \tag{30}$$

where s_i are the slopes defined in (8). Note that the formula does not determine the first derivative values if $s_{i+1} = s_i$ and $s_{i-1} = s_{i-2}$. In this case we shall employ a convention $f_i' = (s_{i-1} + s_i)/2$.

The formula above requires two more points at each endpoint. They are estimated by assuming that all of the last three points (x_{N-2}, y_{N-2}), (x_{N-1}, y_{N-1}), (x_N, y_N) and the extra points (x_{N+1}, y_{N+1}), (x_{N+2}, y_{N+2}) lie on a curve,

$$y = g_0 + g_1(x - x_N) + g_2(x - x_N)^2 \,, \tag{31}$$

where g_0, g_1, g_2 are constants and assuming

$$x_{N+2} - x_N = x_{N+1} - x_{N-1} = x_N - x_{N-2} \,. \tag{32}$$

The resulting equations are

$$\begin{aligned} \frac{y_{N+2} - y_{N+1}}{x_{N+2} - x_{N+1}} - \frac{y_{N+1} - y_N}{x_{N+1} - x_N} &= \frac{y_{N+1} - y_N}{x_{N+1} - x_N} - \frac{y_N - y_{N-1}}{x_N - x_{N-1}} \\ &= \frac{y_N - y_{N-1}}{x_N - x_{N-1}} - \frac{y_{N-1} - y_{N-2}}{x_{N-1} - x_{N-2}} \,. \end{aligned} \tag{33}$$

A similar argument applies for the other endpoint. Then the first derivative values for all the data points are determined by (30).

The resulting curve on the sample data given in Table 1 is depicted in Figure 3. The overshooting problem of the cubic spline interpolations in section 3 is not found in the present case.

4.2 Constrained Cubic Interpolation

While the first derivative value at one point is determined by using five consecutive data points in the Akima cubic interpolation, Kruger proposed a simpler formula by a harmonic mean [Kru02], in which three consecutive points are used for one derivative value,

$$f_i' = \begin{cases} 0 & \text{if slope changes sign at point} \\ \dfrac{2}{1/s_i + 1/s_{i-1}} & \text{otherwise} \end{cases} \tag{34}$$

for $i = 1, 2 \ldots, N-1$. The values at endpoints are given by

$$f_0' = \frac{3}{2}s_0 - \frac{1}{2}f_1' \,, \qquad f_N' = \frac{3}{2}s_{N-1} - \frac{1}{2}f_{N-1}' \,. \tag{35}$$

Although this simple algorithm works well and preserves shape of given data in many cases, drawbacks are sometimes observed if the intervals h_i have different lengths.

For the data in Table 1 the constrained cubic interpolation provides the curve depicted in Figure 4. The overshooting problem of the cubic spline interpolations is avoided by the constrained interpolation.

4.3 Monotonicity Preserving Interpolation

Necessary and sufficient conditions for monotonicity preserving Hermite interpolation were discussed in [FC80] and a simpler algorithm for the monotonicity was proposed by Fritsch and Butland [FB84] in which the first derivative values f_i' are defined by a weighted harmonic mean,

$$\begin{aligned} f_i' &= 0 && \text{if } s_{i-1}s_i \le 0 \,, \\ \frac{1}{f_i'} &= \frac{h_{i-1} + 2h_i}{3(h_i + h_{i-1})}\frac{1}{s_{i-1}} + \frac{2h_{i-1} + h_i}{3(h_i + h_{i-1})}\frac{1}{s_i} && \text{if } s_{i-1}s_i > 0 \,, \end{aligned} \tag{36}$$

for $i = 1, 2 \ldots, N-1$ and the boundaries are defined by

$$f_0' = \begin{cases} 0 & \text{if } g_0 s_0 \le 0 \\ 3s_0 & \text{else if } s_0 s_1 \le 0, \;\; |g_0| > 3\,|s_0| \\ g_0 & \text{otherwise} \end{cases} \tag{37}$$

$$f_N' = \begin{cases} 0 & \text{if } g_N s_{N-1} \le 0 \\ 3s_{N-1} & \text{else if } s_{N-1} s_{N-2} \le 0, \;\; |g_N| > 3\,|s_{N-1}| \\ g_N & \text{otherwise} \end{cases} \tag{38}$$

where g_0, g_N are given by

$$g_0 = \frac{2h_0 + h_1}{h_0 + h_1}s_0 - \frac{h_0}{h_0 + h_1}s_1 \,. \qquad g_N = \frac{2h_{N-1} + h_{N-2}}{h_{N-1} + h_{N-2}}s_{N-1} - \frac{h_{N-1}}{h_{N-1} + h_{N-2}}s_{N-2} \,. \tag{39}$$

Applying the monotone preserving interpolation to the sample data set in Table 1, we then obtain the curve depicted in Figure 5. While this would look very similar to the constrained cubic interpolation shown in Figure 4, the interpolants for $x \in [3, 3.1]$ of these two methods are not exactly the same corresponding to the difference in the treatment of the $h_{i-1} \neq h_i$ cases.

4.4 Hyman Filters: Nonnegativity Preserving Interpolations

The Hermite interpolation methods which we have discussed directly determine first derivative values such that the resulting interpolant has desirable properties. In general, first derivative values in the cubic Hermite interpolation (26), (27) are free parameters and independent of each other, that is, one can change one of them without affecting other first derivative values. This means that if conditions on the first derivatives for the desirable properties are known, the interpolation problem can be solved by using another simple interpolation method and modifying part of the first derivative values only when they do not satisfy the conditions. We will review this approach in this section and the next.

In [DEH89] the authors discussed conditions for nonnegativity (nonpositivity) in the cases of cubic and quintic interpolations. The nonnegativity (nonpositivity) is defined as follows. The data are locally nonnegative (nonpositive) in the interval $[x_i, x_{i+1}]$ if $f_i, f_{i+1} \geq 0$ (≤ 0). The interpolant $f(x)$ is nonnegative (nonpositive) if $f(x) \geq 0$ (≤ 0) for all $x \in [x_i, x_{i+1}]$.

For positive (negative) given data, the piecewise linear interpolation always provides a positive (negative) interpolant. Thus a cubic Hermite interpolation preserves positivity (negativity) if it has the same sign as the linear interpolant. The condition for this is that the first derivative values in the following range,

$$-3\tau_i f_i/h_i \leq \tau_i f_i' \leq 3\tau_i f_i/h_{i-1}\,, \tag{40}$$

where $\tau_i = \mathrm{sgn}(f_i)$.

Figure 6 shows the resulting curves obtained by applying the nonnegativity preserving Hyman filter to the interpolants of the cubic interpolation in section 3 for the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line), that is, we first carry out the cubic spline interpolations, compute the first derivatives and modify the derivative values if the condition (40) is not satisfied. Clearly the interpolants do not take a negative value for all x. It turns out that the Hyman filter perfectly works in the present case. On the other hand the overshooting in the positive y region remains the same.

The restrictions to the first derivatives (40) are extended to the quintic case by replacing the coefficient 3 by 5,

$$-5\tau_i f_i/h_i \leq \tau_i f_i' \leq 5\tau_i f_i/h_{i-1}\,, \tag{41}$$

and the second derivatives are also restricted,

$$\tau_i f_i'' \geq \tau_i \max\left(8f_i'/h_{i-1} - 20f_i/h_{i-1}^2\,,\ \ -8f_i'/h_i - 20f_i/h_i^2\right)\,. \tag{42}$$

As pointed out in [HW06], spline interpolations with higher degree polynomials tend to have more wiggles and bumps. Still they have the advantage of C^2 smoothness even with the nonnegativity or monotonicity constraints imposed.

Since the cubic spline interpolations produce C^2 continuous curves, it is natural to use them in order to specify the first and second derivative values if other quintic interpolation methods are not available. Thus, as done in the cubic case, we interpolate the sample data in Table 1 by the cubic spline interpolations, compute the derivative values and modify them following the inequalities (41), (42). Then we obtain the curves shown in Figure 7.

4.5 Hyman Filters: Monotonicity Preserving Interpolations

Another version of the Hyman filter is preserving monotonicity of data. The definition of monotonicity given in [DEH89] is that the data are locally monotone at x_i if $s_i s_{i-1} \geq 0$ whereas an interpolant f is piecewise monotone if f' does not change sign in any interval (x_i, x_{i+1}).

It was shown in [Hym83] that monotonicity is preserved if the first derivative values satisfy the de Boor-Schwartz constraints [dBS77]. This is realised by applying the following filter on the first derivative values,

$$f_i' \leftarrow \begin{cases} \min\left(\max\left(0, f_i'\right), 3\min\left(|s_{i-1}|, |s_i|\right)\right) & \sigma_i > 0 \\ \max\left(\min\left(0, f_i'\right), -3\min\left(|s_{i-1}|, |s_i|\right)\right) & \sigma_i < 0 \\ 0 & \sigma_i = 0 \end{cases} \tag{43}$$

where $\sigma_i = \operatorname{sgn}(s_i)$ if $s_{i-1}s_i > 0$ and $\sigma_i = 0$ otherwise. However, these conditions are too restrictive since derivative values are constrained to be very small when a slope s_i takes a very small value. The modified algorithm for the monotonicity preserving interpolation is as follows. Let us define

$$\begin{aligned} p_i^{-1} &= \frac{s_{i-1}\left(2h_{i-1} + h_{i-2}\right) - s_{i-2}h_{i-1}}{h_{i-2} + h_{i-1}}, \\ p_i^0 &= \frac{s_{i-1}h_i + s_i h_{i-1}}{h_{i-1} + h_i}, \qquad p_i^1 = \frac{s_i\left(2h_i + h_{i+1}\right) - s_{i+1}h_i}{h_i + h_{i+1}}, \\ M_i &= 3\min\left(|s_{i-1}|, |s_i|, \left|p_i^0\right|\right). \end{aligned} \tag{44}$$

If $i > 1$ and p_i^0, p_i^{-1}, $s_{i-1} - s_{i-2}$ and $s_i - s_{i-1}$ have the same sign, redefine M_i by

$$M_i \leftarrow \max\left(M_i, 1.5\min\left(\left|p_i^0\right|, \left|p_i^{-1}\right|\right)\right). \tag{45}$$

If $i < N - 1$ and $-p_i^0$, $-p_i^1$, $s_i - s_{i-1}$ and $s_{i+1} - s_i$ have the same sign, redefine M_i by

$$M_i \leftarrow \max\left(M_i, 1.5\min\left(\left|p_i^0\right|, \left|p_i^1\right|\right)\right). \tag{46}$$

Then the first derivative values are modified as

$$f_i' \leftarrow \begin{cases} \left(\operatorname{sgn} f_i'\right)\min\left(\left|f_i'\right|, M_i\right) & \text{if } \operatorname{sgn} f_i' = \operatorname{sgn} p_i^0 \\ 0 & \text{otherwise} \end{cases} \tag{47}$$

The endpoints $i = 0, N$ are handled separately. For $i = 0$, set $f_i' = \left(\operatorname{sgn} f_i'\right)\min\left(\left|f_i'\right|, 3\left|s_i\right|\right)$ if $\operatorname{sgn} f_i' = \operatorname{sgn} s_i$ holds, and $f_i' = 0$ otherwise. The first derivative value at the other endpoint, f_n', is determined in a similar way.

Figure 8 illustrates the resulting curves of the monotonicity preserving Hyman filter on the cubic interpolations in section 3 for the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). Since the monotonicity condition is relaxed compared with other monotone preserving methods, e.g., the monotone preserving interpolation in section 4.3, the resulting interpolant can have a local extremum between consecutive data points.

In the case of the quintic interpolation, constraints on the first derivative values corresponding to (43) are not relaxed in a simple manner. Thus we shall employ

$$f_i' \leftarrow \begin{cases} \min\left(\max\left(0, f_i'\right), 5\min\left(|s_{i-1}|, |s_i|\right)\right) & \sigma_i \geq 0 \\ \max\left(\min\left(0, f_i'\right), -5\min\left(|s_{i-1}|, |s_i|\right)\right) & \sigma_i \leq 0 \end{cases} \tag{48}$$

where $\sigma_i = \mathrm{sgn}\,(f'_i)$ if $s_{i-1}s_i < 0$ and $\sigma_i = 0$ otherwise. The second derivatives f''_i are constrained by $f''_i \in A$ and $f''_i \in B$, where A, B are intervals,

$$\begin{aligned} A &= [-7.9d^+ - 0.26d^+b\,,\ \ (20-2b)\,s_i - 8d^+ - 0.48d^+b]\,h_i^{-1}\,, \\ B &= [(-20+2a)\,s_{i-1} + 8d^- + 0.48d^-a\,,\ 7.9d^- + 0.26d^-a]\,h_{i-1}^{-1}\,. \end{aligned} \tag{49}$$

The constants a, b, $d^\pm$ are defined by

$$\begin{aligned} &a = \max\left(0, f'_i/s_{i-1}\right)\,, \qquad b = \max\left(0, f'_{i+1}/s_i\right)\,, \\ &d^+ = \begin{cases} f'_i & \text{if } f'_i s_i > 0 \\ 0 & \text{otherwise} \end{cases} \qquad d^- = \begin{cases} f'_i & \text{if } f'_i s_{i-1} > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{50}$$

If $A \cap B = \emptyset$ is found for some i, we should redefine the relevant first derivative value,

$$f'_i = \frac{(20-2b)\,s_i h_i^{-1} + (20-2a)\,s_{i-1}h_{i-1}^{-1}}{(8+0.48b)\,h_i^{-1} + (8+0.48a)\,h_{i-1}^{-1}}\,, \tag{51}$$

which ensures that the two intervals, A and B, intersect.

The resulting curves of the monotonicity preserving quintic interpolation are depicted in Figure 9. Here we used the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line) to specify the primary first and second derivatives. These values are modified such that the constraints (48), (49) are satisfied.

4.6 Shape Preserving C^2 Cubic Interpolation

In general it is very hard to add C^2 smoothness to cubic Hermite interpolations keeping its shape-preserving feature and avoiding global dependencies. Pruess constructed a C^2 cubic interpolation by introducing two extra breakpoints between each data interval [Pru93]. Then the interval $[x_i, x_{i+1}]$ is divided into 3 pieces, $[x_i, \xi_i]$, $[\xi_i, \eta_i]$ and $[\eta_i, x_{i+1}]$ with $x_i \le \xi_i \le \eta_i \le x_{i+1}$. The piecewise polynomial functions in these intervals are written as

$$\begin{aligned} f(x) &= f_i + f'_i\,(x-x_i) + \frac{f''_i}{2}\,(x-x_i)^2 + \frac{f'''\,(x_i)}{6}\,(x-x_i)^3\,, && x \in [x_i, \xi_i]\,, \\ f(x) &= f\,(\xi_i) + f'\,(\xi_i)\,(x-\xi_i) + \frac{f''\,(\xi_i)}{2}\,(x-\xi_i)^2 + \frac{f'''\,(\xi_i)}{6}\,(x-\xi_i)^3\,, && x \in [\xi_i, \eta_i]\,, \\ f(x) &= f\,(\eta_i) + f'\,(\eta_i)\,(x-\eta_i) + \frac{f''\,(\eta_i)}{2}\,(x-\eta_i)^2 + \frac{f'''\,(\eta_i)}{6}\,(x-\eta_i)^3\,, && x \in [\eta_i, x_{i+1}]\,. \end{aligned} \tag{52}$$

Let us denote the coordinates of the extra knots as $\xi = x_i + \sigma_i h_i$, $\eta_i = x_{i+1} - \tau_i h_i$ for $\sigma_i, \tau_i \in (0, 1/2)$. We will eventually set $\tau_i = \sigma_i$. It was shown in [Pru93] that the interpolant $f(x)$ is C^2

continuous everywhere in $[x_0, x_N]$ if the following conditions are satisfied,

$$
\begin{aligned}
f(\xi_i) &= f_i + \sigma_i h_i f_i' + \frac{\sigma_i^2 h_i^2}{6}\left(2f_i'' + f''(\xi_i)\right), \\
f(\eta_i) &= f_{i+1} - \tau_i h_i f_{i+1}' + \frac{\tau_i^2 h_i^2}{6}\left(2f_{i+1}'' + f''(\eta_i)\right), \\
f'(\xi_i) &= f_i' + \frac{\sigma_i h_i}{2}\left(f_i'' + f''(\xi_i)\right), \qquad f'(\eta_i) = f_{i+1}' - \frac{\tau_i h_i}{2}\left(f_{i+1}'' + f''(\eta_i)\right), \\
f''(\xi_i) &= \frac{6s_i - 2(2+\sigma_i-\tau_i)f_i' - 2(1-\sigma_i+\tau_i)f_{i+1}' - \sigma_i h_i(2-\tau_i)f_i'' + \tau_i h_i(1-\sigma_i)f_{i+1}''}{h_i(1-\tau_i)}, \\
f''(\eta_i) &= \frac{-6s_i + 2(1+\sigma_i-\tau_i)f_i' + 2(2-\sigma_i+\tau_i)f_{i+1}' + \sigma_i h_i(1-\tau_i)f_i'' - \tau_i h_i(2-\sigma_i)f_{i+1}''}{h_i(1-\sigma_i)}, \\
f'''(x_i) &= \frac{f''(\xi_i) - f_i''}{\sigma_i h_i}, \qquad f'''(\xi_i) = \frac{f''(\eta_i) - f''(\xi_i)}{(1-\sigma_i-\tau_i)h_i}, \qquad f'''(\eta_i) = \frac{f_{i+1}'' - f''(\eta_i)}{\tau_i h_i}.
\end{aligned}
\tag{53}
$$

Thus the interpolation is completed once we specify f_i', f_i'', τ_i and σ_i.

The definition of the locally monotonicity preserving interpolant given in the literature is that a function $f(x)$ is locally monotonicity preserving for data (x_i, y_i) if: when for some i, $1 \le i \le N-2$, s_{i-1}, s_i, and s_{i+1} all have the same sign, then $f'(x)$ must have this sign everywhere in $[x_i, x_{i+1}]$. On the other hand, a function $f(x)$ is locally convexity preserving for data (x_i, y_i) if and only if the following holds: (1) $f(x)$ has an inflection point in $[x_i, x_{i+1}]$ if and only if $\beta_i\beta_{i-1} < 0$; (2) where β_i and β_{i-1} have the same sign, $f''(x)$ also must have this same sign in $[x_i, x_{i+1}]$.

To preserve these properties we first compute f_i', for example, by using [FB84], and check inequalities,

$$
\beta_i R_{2i-1} \ge 0, \qquad \beta_{i+1} R_{2i} \ge 0, \tag{54}
$$

where

$$
R_{2i-1} = 6s_i - 4f_i' - 2f_{i+1}', \qquad R_{2i} = 2f_i' + 4f_{i+1}' - 6s_i, \qquad \beta_i = \operatorname{sgn}(s_i - s_{i-1}). \tag{55}
$$

If the inequalities hold, the primary interpolant also preserves convexity. Otherwise, the first derivative values should be modified such that $f_i' \in [c_i, d_i]$ where c_i, d_i are obtained by the following double sweep algorithm,

$$
a_i = \begin{cases}
\max\left[s_i,\ (3s_i - b_i)/2\right] & \beta_i > 0, \beta_{i+1} > 0 \\
\max\left[3s_i - 2b_i,\ (3s_i - b_i)/2\right] & \beta_i < 0, \beta_{i+1} > 0 \\
\max\left[s_{i+1},\ 3s_i - 2b_i\right] & \beta_i < 0, \beta_{i+1} < 0
\end{cases} \tag{56}
$$

$$
b_i = \begin{cases}
3s_i - 2a_i & \beta_i > 0, \beta_{i+1} > 0 \\
\min\left[3s_i - 2a_i,\ (3s_i - a_i)/2\right] & \beta_i > 0, \beta_{i+1} < 0 \\
(3s_i - a_i)/2 & \beta_i < 0, \beta_{i+1} < 0
\end{cases} \tag{57}
$$

starting with

$$
a_0 = \begin{cases}
3s_0 - 2s_1 & \beta_0 = \beta_1 > 0 \\
s_0 & \beta_0 = \beta_1 < 0
\end{cases} \tag{58}
$$

$$b_0 = \begin{cases} s_0 & \beta_0 = \beta_1 > 0 \\ 3s_0 - 2s_1 & \beta_0 = \beta_1 < 0 \end{cases} \tag{59}$$

For the backward sweep, choose s'_N in $[a_N\,, b_N]$ and let

$$c_i = \begin{cases} 3s_i - f'_{i+1} & \beta_i > 0\,, \beta_{i+1} > 0 \\ \max\left[3s_i - 2f'_{i+1}\,,\ \left(3s_i - f'_{i+1}\right)/2\right] & \beta_i < 0\,, \beta_{i+1} > 0 \\ \left(3s_i - f'_{i+1}\right)/2 & \beta_i < 0\,, \beta_{i+1} < 0 \end{cases} \tag{60}$$

$$d_i = \begin{cases} \left(3s_i - f'_{i+1}\right)/2 & \beta_i > 0\,, \beta_{i+1} > 0 \\ \min\left[3s_i - 2f'_{i+1}\,,\ \left(3s_i - f'_{i+1}\right)/2\right] & \beta_i < 0\,, \beta_{i+1} > 0 \\ 3s_i - 2f'_{i+1} & \beta_i < 0\,, \beta_{i+1} < 0 \end{cases} \tag{61}$$

Once we confirm the inequalities (54), the second derivatives are computed by using the first derivatives,

$$f''_i = \beta_i \min\left(\beta_i R_{2i-1}/h_i\,,\ \beta_i R_{2i-2}/h_{i-1}\right)\,. \tag{62}$$

To find appropriate τ_i and σ_i, we set $\tau_i = \sigma_i = 1/3$ and check inequalities,

$$\begin{aligned} &\left(4f'_i + 2f'_{i+1} + h_i f''_i \tau_i(2-\tau_i) - h_i f''_{i+1}\tau_i(1-\tau_i)\right)\beta_i \le 6s_i\beta_i\,, \\ &\left(2f'_i + 4f'_{i+1} + h_i f''_i \tau_i(1-\tau_i) - h_i f''_{i+1}\tau_i(2-\tau_i)\right)\beta_{i+1} \ge 6s_i\beta_{i+1}\,. \end{aligned} \tag{63}$$

If these inequalities are satisfied, we employ $\tau_i = \sigma_i = 1/3$. Otherwise we should reduce the values of τ_i, σ_i. Finally plugging f'_i, f''_i, τ_i and σ_i into the equations (53) yields the shape preserving C^2 interpolation. We should note that one drawback of this interpolation method is increased data storage requirement due to the additional breakpoints. For details including mathematical proofs, see the original paper [Pru93].

For the sample data in Table 1 the resulting interpolant of the shape preserving C^2 cubic interpolation is depicted in Figure 10. While the curve would look very similar to the C^1 case in Figure 5, these two interpolants are quantitatively and qualitatively different and one can confirm that the C^2 continuity holds everywhere in $x \in [1, 8]$ in the present case.

5 Monotone Convex Interpolation

The monotone convex interpolation introduced by Hagan and West [HW06] is designed for yield curve construction. Given a data set (x_i, y_i), let us construct the quantities,

$$F_i^d = \frac{y_i x_i - y_{i-1}x_{i-1}}{x_i - x_{i-1}}\,, \tag{64}$$

for $i = 1, 2, \ldots, N$ and

$$\begin{aligned} &F_i = \frac{x_i - x_{i-1}}{x_{i+1} - x_{i-1}} F_{i+1}^d + \frac{x_{i+1} - x_i}{x_{i+1} - x_{i-1}} F_i^d\,, \quad i = 1, 2, \ldots, N-1\,, \\ &F_0 = F_1^d - \left(F_1 - F_1^d\right)/2\,, \qquad F_N = F_N^d - \left(F_{N-1} - F_N^d\right)/2\,. \end{aligned} \tag{65}$$

The monotone convex interpolation method is to find a quadratic function $F(x)$ in every interval $[x_{i-1}, x_i]$ such that

$$F(x_i) = F_i\,, \qquad F(x_{i-1}) = F_{i-1}\,, \qquad \frac{1}{x_i - x_{i-1}} \int_{x_{i-1}}^{x_i} F(\xi) d\xi = F_i^d\,, \tag{66}$$

then integrate the piecewise polynomial functions $F(x)$ over $[x_0, x_1]\,,\ [x_1, x_2]\,, \ldots$ to obtain a function $f(x)$ on which the data points (x_i, y_i) lie. The method in general offers a C^1 cubic spline.

It turns out that the quadratic function $F(x)$ should take the following form under the conditions (66),

$$\begin{gathered} F(x) = \left(1 - 4X(x) + 3X(x)^2\right) F_{i-1} + \left(-2X(x) + 3X(x)^2\right) F_i + \left(6X(x) - 6X(x)^2\right) F_i^d\,, \\ X(x) = \frac{x - x_{i-1}}{x_i - x_{i-1}}\,, \qquad x \in [x_{i-1}, x_i]\ . \end{gathered} \tag{67}$$

Although the function form is uniquely determined, modification of the piecewise quadratic function is allowed if we introduce extra breakpoints. This enables us to obtain an interpolation preserving the monotonicity and convexity. For analysing monotonicity and convexity conditions, it is convenient to shift the function $F(x)$ as $G(X) = F(X(x)) - F_i^d$, that is,

$$G(X) = \left(1 - 4X + 3X^2\right) G(0) + \left(-2X + 3X^2\right) G(1)\,, \tag{68}$$

and its first derivative with respect to X is

$$G'(X) = (-4 + 6X)\, G(0) + (-2 + 6X)\, G(1)\,. \tag{69}$$

Depending on the signs of $G(X)$ and $G'(X)$ at the knot points $X = 0, 1$ we have four regions on the $G(0)$-$G(1)$ plane. The boundaries of the four regions are $G(1) = -2G(0)$ (boundary $\mathcal{A}$), $G(0) = -2G(1)$ (boundary $\mathcal{B}$), $G(0) = 0$ (boundary $\mathcal{C}$) and $G(1) = 0$ (boundary $\mathcal{D}$).

On boundary $\mathcal{A}$ the shifted function is $G(X) = G(0)(1 - 3X^2)$ whereas we have $G(X) = G(0)(1 - 3X + 3X^2/2)$ on boundary $\mathcal{B}$. At the origin on the $G(0)$-$G(1)$ plane we have $G(X) = 0$ for all X and $F_{i-1}^d = F_i^d = F_{i+1}^d$ meaning that the interpolant is $F(x) = F_i^d$.

We observe that the four regions are

(i) $G(0)$, $G(1)$ are of opposite sign while $G'(0)$, $G'(1)$ are of the same sign. Thus $G(X)$ is monotone and need not be modified.

(ii) $G(0)$, $G(1)$ are of opposite sign and $G'(0)$, $G'(1)$ are also of the opposite sign. Thus $G(X)$ is not monotone and should be modified such that the formulas for (i) and (ii) agree on boundary $\mathcal{A}$.

(iii) $G(0)$, $G(1)$ are of opposite sign and $G'(0)$, $G'(1)$ are also of the opposite sign. The relation of the signatures is the same as (ii), but $G(x)$ for (i) and (iii) should match on boundary $\mathcal{B}$.

(iv) $G(0)$, $G(1)$ are of the same sign. $G(X)$ should be modified such that the continuity holds on boundaries $\mathcal{C}$ and $\mathcal{D}$.

The pictorial description of this classification is given in [HW06].

One simple modification of $G(X)$ in the regions (ii), (iii) and (iv) can be done by introducing extra knots between $X = 0$ and $X = 1$, and inserting new functions keeping the conditions in (66) satisfied. We redefine the interpolation functions in (ii),

$$G(X) = \begin{cases} G(0) & 0 \le X \le \eta \\ G(0) + (G(1) - G(0)) \left(\frac{X-\eta}{1-\eta}\right)^2 & \eta \le X \le 1 \end{cases} \qquad \eta = \frac{G(1) + 2G(0)}{G(1) - G(0)} \,. \tag{70}$$

The modification in (iii) is

$$G(X) = \begin{cases} G(1) + (G(0) - G(1)) \left(\frac{\eta-X}{\eta}\right)^2 & 0 \le X \le \eta \\ G(1) & \eta \le X \le 1 \end{cases} \qquad \eta = \frac{3G(1)}{G(1) - G(0)} \,. \tag{71}$$

In (iv) we have

$$G(X) = \begin{cases} A + (G(0) - A) \left(\frac{\eta-X}{\eta}\right)^2 & 0 \le X \le \eta \\ A + (G(1) - A) \left(\frac{X-\eta}{1-\eta}\right)^2 & \eta \le X \le 1 \end{cases} \tag{72}$$

where η and A are not uniquely determined in this case, but a simple choice is

$$\eta = \frac{G(1)}{G(1) + G(0)} \,, \qquad A = -\frac{G(0)G(1)}{G(1) + G(0)} \,. \tag{73}$$

One can confirm that the modified functions above satisfy the boundary conditions on $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}$, respectively, and the constraints in (66).

The positivity of the interpolant requires further modification. The definition of $G(X)$ implies that the function $F(X)$ is positive anywhere in $[0, 1]$ if

$$G(X) \ge -F_i^d \,. \tag{74}$$

For positive F_i^d, this condition is satisfied at the knot points, x_i. Since the function $G(X)$ are monotone between the knots and the inequality (74) is always satisfied for (i), (ii) and (iii), we need to check the inequality in the region (iv). If $G'(0) < 0$ and $G'(1) > 0$, $G(X)$ has a minimum at $X = X_{\min} \in [0, 1]$. In this case we should impose $G\left(X_{\min}\right) \ge -F_i^d$, which can be realised by applying the transformation on F_i,

$$\begin{aligned} &F_i \leftarrow \min\left(\max\left(0, F_i\right), 2\min\left(F_i^d, F_{i+1}^d\right)\right) \,, \quad i = 1, 2, \ldots, N-1 \,, \\ &F_0 \leftarrow \min\left(\max\left(0, F_0\right), 2F_1^d\right) \,, \qquad F_N \leftarrow \min\left(\max\left(0, F_N\right), 2F_N^d\right) \,. \end{aligned} \tag{75}$$

Finally the interpolant $f(x)$ is obtained by integrating the resulting function $F(x)$ over $[x_0, x_1]$, $[x_1, x_2], \ldots,\ [x_i, x]$.

As done in the other interpolation methods, the sample data in Table 1 are interpolated by the monotone convex interpolation in Figure 11.[3] This interpolation should be compared with the monotonicity preserving interpolation discussed in section 4.3. Reflecting the convexity of the curve and introduction of extra knots, the individual interpolants between data points exhibit small wiggles in the present case.

[3] The monotone convex interpolation is originally designed to interpolate $(x_i, y_i \cdot x_i)$ rather than (x_i, y_i) and particularly useful when $(x_i, y_i \cdot x_i)$ are monotone and monotonicity of $f(x) \cdot x$ is expected. Here we apply the interpolation method to (x_i, y_i) assuming the structure $y_i = \hat{y}_i \cdot x_i$ for all i. Generally interpolating $(x_i, y_i \cdot x_i)$ and dividing an interpolant $f(x) \cdot x$ by x yield a different curve.

6 Two-Dimensional Spline Interpolations

Given mesh-gridded data $(x_i, y_j, z_{i,j})$ with $i = 0, 1, \ldots, M$ and $j = 0, 1, \ldots, N$ in three-dimensional space, the two-dimensional spline interpolation problem is to find a surface defined by a function $f : [x_0, x_M] \times [y_0, y_N] \to \mathbf{R}$ such that $z_{i,j} = f(x_i, y_j)$ for all i, j. In fact the variety of interpolations based on piecewise polynomial functions is very limited in two or more dimensions. Here we shall review two primary two-dimensional spline interpolation methods, bilinear interpolation and bicubic interpolation.

For notational simplicity we shall employ the following coordinate transformations for $x \in [x_i, x_{i+1}]$, $y \in [y_j, y_{j+1}]$,

$$t = \frac{x - x_i}{x_{i+1} - x_i}\,, \qquad u = \frac{y - y_j}{y_{j+1} - y_j}\,. \tag{76}$$

Note that the definition used in this section is different from the one used in the one-dimensional case (8). Thus the interpolation function for $x \in [x_i, x_{i+1}]$, $y \in [y_j, y_{j+1}]$ takes the form,

$$f^{i,j}(t, u) = \sum_{k=0}^{m} \sum_{l=0}^{n} a_{kl}^{i,j} t^k u^l\,. \tag{77}$$

We have $m, n = 1$ for the bilinear interpolation and $m, n = 3$ for the bicubic interpolation.

For both of the two-dimensional interpolation methods, we interpolate a sample data set,

$$\begin{aligned} x &= \{-2,\ -1.5,\ -1,\ -0.5,\ 0,\ 0.5,\ 1,\ 1.5,\ 2\}\,, \\ y &= \{-2,\ -1.5,\ -1,\ -0.5,\ 0,\ 0.5,\ 1,\ 1.5,\ 2\}\,, \\ z_{i,j} &= \frac{1}{2} \exp\left(-\frac{x_i^2 + y_j^2}{4}\right)\,. \end{aligned} \tag{78}$$

The results are shown in Figure 12 and Figure 13, respectively.

6.1 Bilinear Interpolation

In a grid square $(x, y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}]$, the bilinear interpolation is defined by

$$\begin{gathered} a_{00}^{i,j} = z_{i,j}\,, \qquad a_{10}^{i,j} = -z_{i,j} + z_{i+1,j}\,, \qquad a_{01}^{i,j} = -z_{i,j} + z_{i,j+1}\,, \\ a_{11}^{i,j} = z_{i,j} - z_{i+1,j} - z_{i,j+1} + z_{i+1,j+1} \end{gathered} \tag{79}$$

As in the one-dimensional linear interpolation, this interpolation provides a C^0 surface: the first derivative values are discontinuous on the grid square boundaries. For the sample data in (78), the bilinear interpolation produces the surface shown in Figure 12.

6.2 Bicubic Interpolation

The bicubic interpolation guarantees the continuity of function value $f(x, y)$, gradients $f_x(x, y)$, $f_y(x, y)$ and cross derivative $f_{xy}(x, y)$.[4] To construct bicubic spline we need to specify the

[4] Here we use the notation,

$$f_x(x, y) = \partial f(x, y)/\partial x\,, \qquad f_y(x, y) = \partial f(x, y)/\partial y\,, \qquad f_{xy}(x, y) = \partial^2 f(x, y)/\partial x \partial y\,. \tag{80}$$

function values, the values of their gradient and the value of their cross derivative at all of the grid points, (x_i, y_j). One way to estimate the derivative values is to use one-dimensional interpolation methods along the x, y directions. Once the derivatives values are computed, the interpolation problem for $(x, y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}]$ is reformulated into a linear system,

$$A\alpha = \beta\,, \tag{81}$$

where

$$\begin{aligned}\alpha &= (a_{00}\, a_{10}\, a_{20}\, a_{30}\, a_{01}\, a_{11}\, a_{21}\, a_{31}\, a_{02}\, a_{12}\, a_{22}\, a_{32}\, a_{03}\, a_{13}\, a_{23}\, a_{33})^T\,,\\ \beta &= \big(f(0,0)\, f(1,0)\, f(0,1)\, f(1,1)\, f_x(0,0)\, f_x(1,0)\, f_x(0,1)\, f_x(1,1)\\ &\qquad f_y(0,0)\, f_y(1,0)\, f_y(0,1)\, f_y(1,1)\, f_{xy}(0,0)\, f_{xy}(1,0)\, f_{xy}(0,1)\, f_{xy}(1,1)\big)^T\,,\end{aligned} \tag{82}$$

with indices ^{i,j} suppressed. The inverse form of A is known, e.g., in [PFTV88],

$$A^{-1} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
-3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 3 & 0 & 0 & -2 & -1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 & 1 & 1 & 0 & 0\\
-3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & -3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0\\
9 & -9 & -9 & 9 & 6 & 3 & -6 & -3 & 6 & -6 & 3 & -3 & 4 & 2 & 2 & 1\\
-6 & 6 & 6 & -6 & -3 & -3 & 3 & 3 & -4 & 4 & -2 & 2 & -2 & -2 & -1 & -1\\
2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0\\
-6 & 6 & 6 & -6 & -4 & -2 & 4 & 2 & -3 & 3 & -3 & 3 & -2 & -1 & -2 & -1\\
4 & -4 & -4 & 4 & 2 & 2 & -2 & -2 & 2 & -2 & 2 & -2 & 1 & 1 & 1 & 1
\end{pmatrix} \tag{83}$$

The two-dimensional surface of the bicubic interpolation on the sample data set in (78) is depicted in Figure 13. As found in the case of one-dimensional cubic spline interpolation in section 3, the surface exhibits unnatural bumps.

7 Other useful interpolations

Let us briefly comment on other interpolation methods which are also frequently used in finance.

In the interpolation algorithms described in this note, the shape-preserving properties are realised by introducing extra breakpoints or determining the derivative values locally. Another way of preserving shape of given data is to introduce a tension parameter which enables one to make a cubic interpolant arbitrarily close to the piecewise linear function keeping the continuous second derivative values [Sch66]. Renka developed a tension-spline algorithm of using piecewise exponential functions with a tension factor in each interval [Ren87], where the tension parameters are automatically determined and the resulting interpolant is C^1 continuous globally and C^2 smoothness can be added by an iterative procedure. Application of the tension spline in the context of finance is found in [Flo13].

A parallel approach to the piecewise polynomial interpolation problem is provided by B-spline [dB78]. A piecewise polynomial function of degree n has an alternative representation,

$$f(x) = \sum_{-\infty<j<\infty} a_j B_j^{(n)}(x)\,, \tag{84}$$

where a_j are constants to be fixed and the basis functions $B_j^{(n)}(x)$ satisfy

$$\begin{aligned} B_j^{(n)}(x) &\geq 0 \quad x_j \leq x \leq x_{j+1} \\ B_j^{(n)}(x) &= 0 \quad \text{otherwise} \end{aligned} \tag{85}$$

for knot points $\cdots < x_{-1} < x_0 < x_1 < \cdots < x_j < \ldots$, and

$$\sum_{-\infty<j<\infty} B_j^{(n)}(x) = 1\,, \tag{86}$$

that is, the basis functions form a partition of unity. An objective curve is obtained by solving the least square problem,

$$\chi^2 = \sum_{i=0}^{N} \left(y_i - \sum_{-\infty<j<\infty} a_j B_j^{(n)}(x_i) \right)^2 . \tag{87}$$

The coefficients a_j are determined by solving the linear problem and the resulting curve is C^{n-1} continuous. The positions of the knots are not necessarily the same as the data points but free parameters which allow for limited control over fitting and smoothing. As discussed in [EM96] the choice of the knots involves computational problems. Smoothing techniques with the B-splines are considered, for example, in [EM96, HN99].

In section 6 we have discussed the bilinear interpolation and the bicubic interpolation. Another two-dimensional spline interpolation is thin plate spline [Duc76, Duc77]. This corresponds to the generalization of the Natural spline discussed in section 3 and the interpolant $f(x, y)$ is such that $z_{i,j} = f(x_i, y_j)$ and the "bending energy" $E[f]$ is minimised,

$$E[f] = \int_{R^2} \left| D^2 f \right|^2 dxdy\,, \tag{88}$$

where $\left|D^2 f\right| = f_{xx} + 2f_{xy} + f_{yy}$. The minimum is found by solving the biharmonic equation and its solution is known for arbitrary dimensions. Thus the multi-dimensional generalization of the thin plate spline is achieved straightforwardly. For financial applications, see [Hom11] and references therein.

References

[Aki70] Hiroshi Akima. A new method of interpolation and smooth curve fitting based on local procedures. *Journal of the Association for Computing Machinery*, 17(4):589–602, 1970. 6

[dB78] Carl de Boor. *A Practical Guide to Splines*. Springer, 1978. 4, 16

[dBS77] Carl de Boor and B. Schwartz. Piecewise monotone interpolation. *Journal of Approximation Theory*, 21:411–416, 1977. 9

[DEH89] Randall L. Dougherty, Alan S. Edelman, and James M. Hyman. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic hermite interpolation. *Mathematics of Computation*, 52(186):471–494, 1989. 8

[Duc76] J. Duchon. Interpolation des fonctions de deux variables suivant le principe de la minces. *RAIRO Analyse Numérique*, 10:5–12, 1976. 17

[Duc77] J. Duchon. Splines minimizing rotation-invariant semi-norms in sobolev spaces,. *Lecture Notes in Mathematics (ZAMP)*, 57:85–100, 1977. 17

[EM96] Paul H. C. Eilers and Brian D. Marx. Flexible smoothing with b-splines and penalties. *Statistical Science*, 11(2):89–121, 1996. 17

[FB84] Fred N. Fritsch and J Butland. A method for constructing local monotone piecewise cubic interpolants. *SIAM Journal on Scientific and Statistical Computing*, 5(2):300–304, 1984. 7, 11

[FC80] Fred N. Fritsch and Ralph E. Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):2386, 1980. 7

[Flo13] Fabien Le Floc'h. Stable interpolation for the yield curve. *Working Papers Series*, 2013. 1, 16

[HN99] Xuming He and Pin Ng. Cobs: Qualitatively constrained smoothing via linear programming. 1999. 17

[Hom11] Cristian Homescu. Implied volatility surface: construction. 2011. arXiv:1107.1834v1 [q-fin.CP]. 1, 17

[HW06] Patrick S. Hagan and Graeme West. Interpolation methods for curve construction. *Applied Mathematical Finance*, 13(2):899, 2006. 1, 8, 12, 14

[Hym83] James M. Hyman. Accurate monotonicity preserving cubic interpolation. *SIAM Journal on Scientific and Statistical Computing*, 4(4):645–654, 1983. 9

[Kru02] CJC Kruger. Constrained cubic spline interpolation for chemical engineering applications. 2002. http://www.korf.co.uk/spline.pdf. 7

[PFTV88] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1988. 16

[Pru93] Steven Pruess. Shape preserving c^2 cubic spline interpolation. *IMA Journal of Numerical Analysis*, 13(4):493–507, 1993. 10, 12

[Ren87] Robert J. Renka. Interpolatory tension splines with automatic selection. *SIAM Journal of Scientific and Statistical Computing*, 8:393–415, 1987. 16

[Sch66] D. G. Schweikert. An interpolation curve using a spline in tension. *Journal of Mathematics and Physics*, 45:312–313, 1966. 16

[Whi12] Richard White. Multiple curve construction. Technical report, OpenGamma, 2012. . 1

x	1	2	3	3.1	5.1	6	7	8
y	1.8	1.9	1.7	1.1	1.1	1.7	1.4	1.9

Table 1: Sample data set. The number of elements is $N + 1 = 8$.

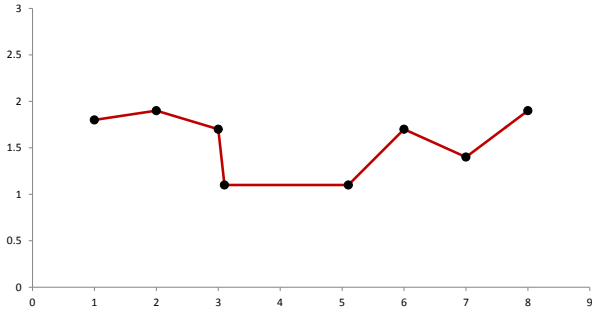

Figure 1: Linear interpolation (red line) and the data points (black dots). Two consecutive data points are connected by a straight line.

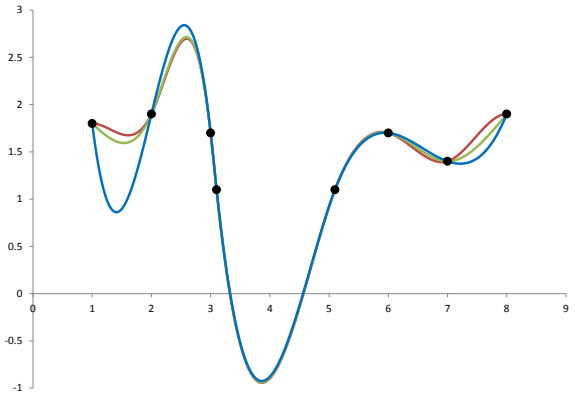

Figure 2: Cubic spline interpolations with different endpoint conditions: the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). The black dots represent the data points. The resulting interpolation functions are dependent on endpoint conditions. One can see a serious problem in these interpolation methods in $x \in [3.1, 5.1]$ where the spline functions take negative values although y values of all of the data are positive.

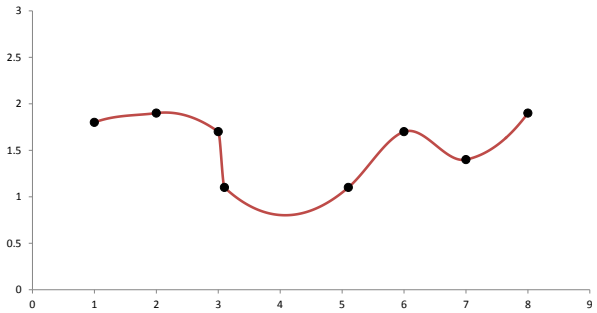

Figure 3: Akima cubic interpolation (red line) and the data points (black dots). The overshooting problem of the cubic spline interpolations in section 3 is not found in the present case.

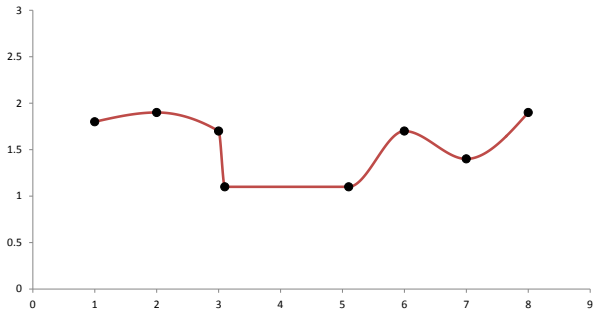

Figure 4: Constrained cubic interpolation (red line) and the data points (black dots). The overshooting problem of the cubic spline interpolations is avoided by the constrained interpolation.

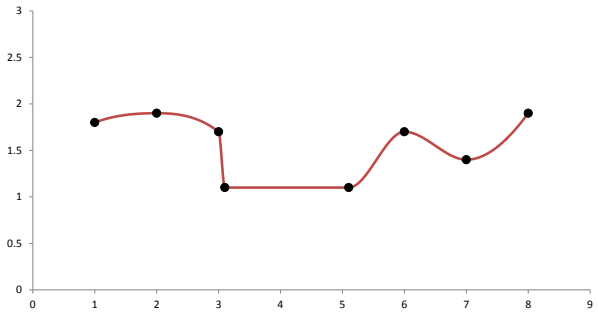

Figure 5: Monotonicity preserving interpolation (red line) and the data points (black dots). Two consecutive data points are connected by a monotonic function and the interpolant has extrema at breakpoints.

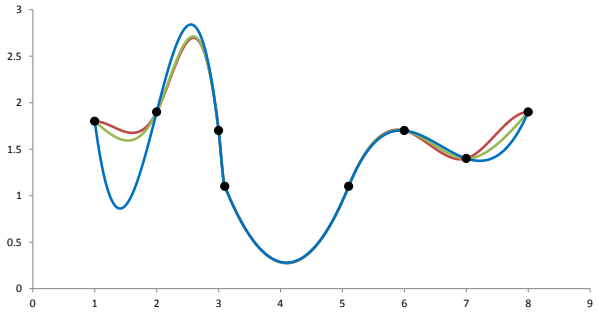

Figure 6: Nonnegativity preserving cubic interpolations starting with the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). The black dots represent the data points. The resulting interpolants do not take a negative value for all x while the overshooting behaviour is still observed.

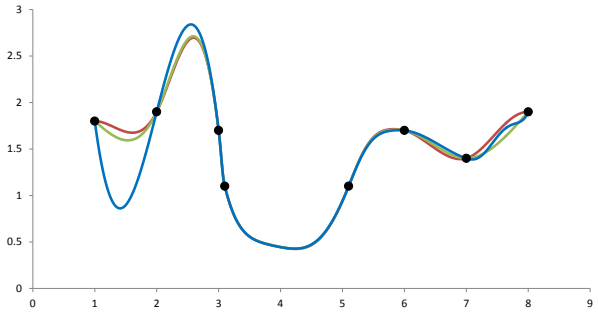

Figure 7: Nonnegativity preserving quintic interpolations where the primary first and second derivatives are computed by the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). The black dots represent the data points. The resulting interpolants are positive for all x while the overshooting behaviour is observed.

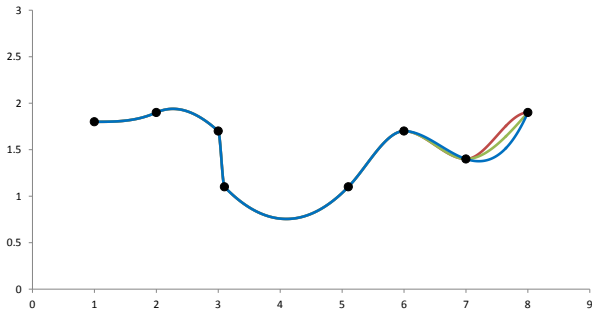

Figure 8: Monotonicity preserving cubic interpolations starting with the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). The black dots represent the data points. Since the monotonicity condition is relaxed compared with other monotone preserving method, e.g., the monotone preserving interpolation in section 4.3, the resulting interpolant can have a local extremum between consecutive data points.

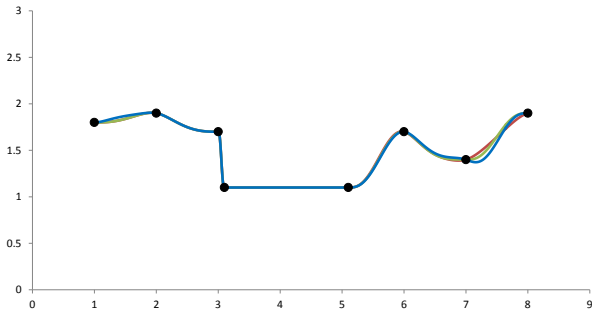

Figure 9: Monotonicity preserving quintic interpolations where the primary first and second derivatives are computed by the Clamped spline (red line), the Natural spline (green line) and Not-A-Knot spline (blue line). The black dots represent the data points. Since the monotonicity constraints are not relaxed for the quintic interpolation, the shape of the interpolant is largely different from the cubic case.

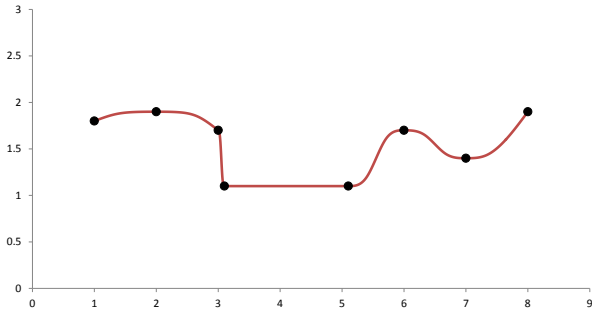

Figure 10: Shape preserving C^2 cubic interpolation (red line) and the data points (black dots). While the curve would look very similar to the C^1 case in Figure 5, the interpolant is C^2 continuous everywhere in $x \in [1, 8]$ in the present case.

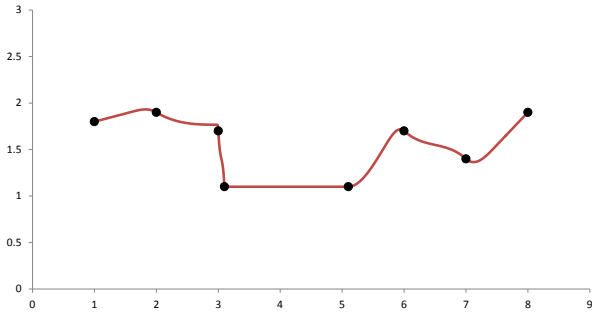

Figure 11: Monotone convex interpolation (red line) and the data points (black dots). This interpolation should be compared with the monotonicity preserving interpolation discussed in section 4.3. In the present case the individual interpolants between data points exhibit small wiggles due to the convexity constraints on the curve and introduction of extra knots.

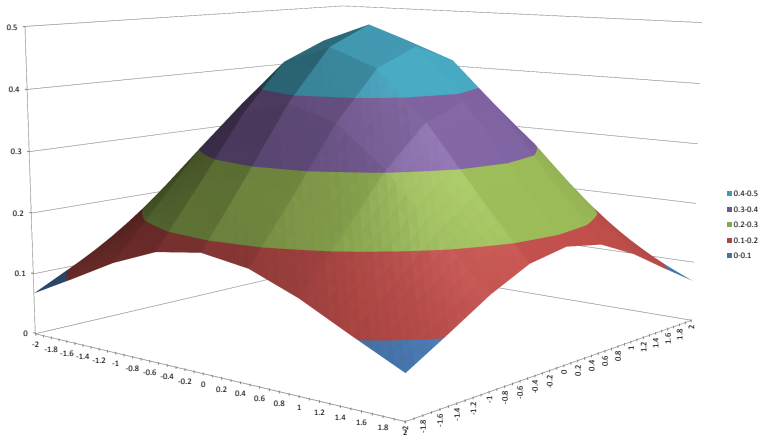

Figure 12: Bilinear interpolation on the data set in (78). The first derivative values are discontinuous on the grid square boundaries.

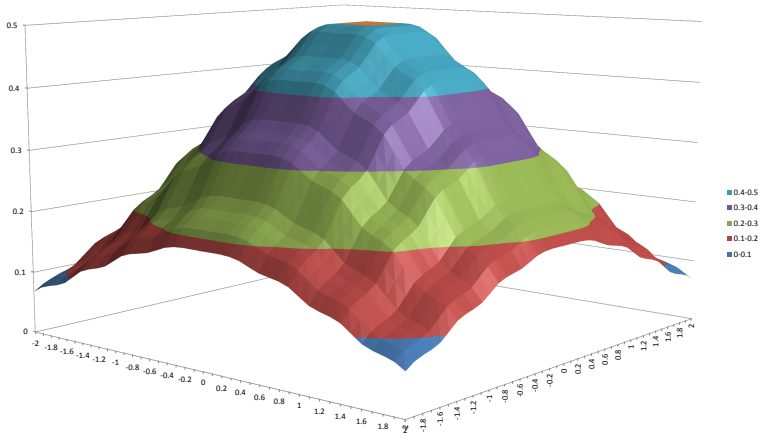

Figure 13: Bicubic interpolation on the data set in (78). While the function value $f(x, y)$, its gradients $f_x(x, y)$, $f_y(x, y)$ and cross derivative $f_{xy}(x, y)$ are continuous, the surface exhibits unnatural wiggles as found in the case of the one-dimensional cubic spline interpolation.

OpenGamma Quantitative Research

1. Marc Henrard. Adjoint Algorithmic Differentiation: Calibration and implicit function theorem. November 2011.
2. Richard White. Local Volatility. January 2012.
3. Marc Henrard. My future is not convex. May 2012.
4. Richard White. Equity Variance Swap with Dividends. May 2012.
5. Marc Henrard. Deliverable Interest Rate Swap Futures: Pricing in Gaussian HJM Model. September 2012.
6. Marc Henrard. Multi-Curves: Variations on a Theme. October 2012.
7. Richard White. Option pricing with Fourier Methods. April 2012.
8. Richard White. Equity Variance Swap Greeks. August 2012.
9. Richard White. Mixed Log-Normal Volatility Model. August 2012.
10. Richard White. Numerical Solutions to PDEs with Financial Applications. February 2013.
11. Marc Henrard. Multi-curves Framework with Stochastic Spread: A Coherent Approach to STIR Futures and Their Options. March 2013.
12. Marc Henrard. Algorithmic Differentiation in Finance: Root Finding and Least Square Calibration. January 2013.
13. Marc Henrard. Multi-curve Framework with Collateral. April 2013.
14. Yukinori Iwashita. Mixed Bivariate Log-Normal Model for Forex Cross. January 2013.
15. Yukinori Iwashita. Piecewise Polynomial Interpolations May 2013

About OpenGamma

OpenGamma helps financial services firms unify their calculation of analytics across the traditional trading and risk management boundaries.

The company's flagship product, the OpenGamma Platform, is a transparent system for front-office and risk calculations for financial services firms. It combines data management, a declarative calculation engine, and analytics in one comprehensive solution. OpenGamma also develops a modern, independently-written quantitative finance library that can be used either as part of the Platform, or separately in its own right.

Released under the open source Apache License 2.0, the OpenGamma Platform covers a range of asset classes and provides a comprehensive set of analytic measures and numerical techniques.

Find out more about OpenGamma
`www.opengamma.com`

Download the OpenGamma Platform
`developers.opengamma.com/downloads`

Europe
OpenGamma
185 Park Street
London SE1 9BL
United Kingdom

North America
OpenGamma
280 Park Avenue
27th Floor West
New York, NY 10017
United States of America