PRE-PRINT
UD 2014-03

Curiosities on the Monotone Preserving Cubic Spline

Michele Maio[*]
and
Jorrit-Jaap de Jong[†]

uGly Duckling B.V.

Burgemeester le Fevre de Montignylaan 30,
3055LG Rotterdam, the Netherlands

December 9, 2014

Abstract

In this paper we describe some new features of the monotone-preserving cubic splines and the Hyman's monotonicity constraint, that is implemented into various spline interpolation methods to ensure monotonicity. We find that, while the Hyman constraint is in general useful to enforce monotonicity, it can be safely omitted when the monotone-preserving cubic spline is considered. We also find that, when computing sensitivities, consistency requires making some specific assumptions about how to deal with non-differentiable locations, that become relevant for special values of the parameter space.

Keywords: Yield curve, fixed-income, interpolation, Hyman, monotone preserving cubic splines.

[*]`m.maio@uglyduckling.nl`
[†]`j.jong@uglyduckling.nl`

Contents

1 Introduction

In this paper we review and streamline the content of our previous paper [1], where we considered one-dimensional interpolation methods that preserve the trend of the input data points. Useful reviews of various interpolation techniques are e.g. [2, 3, 4, 5]. We will mostly use [4][1]. One key issue is to be able to construct the interpolating function in such a way that it follows the trend of the data points. This property is usually referred to as monotonicity-preserving and it helps to remove spurious effects, such as oscillating behavior.

One of the most famous monotonicity-preserving algorithm was introduced by Hyman [8] and can be implemented in principle on almost any cubic-spline-like interpolation method. In fact, a large class of cubic splines is defined by specifying the vector of derivatives at the input points and the Hyman algorithm works exactly on these derivatives. In this paper we will focus on one particular spline only, namely the so-called *monotone preserving* spline. It is defined as that spline where Hyman's adjustment is enforced on top of the Fritsch-Butland prescription [3, 9, 10].

In many practical applications, we are interested in the behavior of the specific interpolation method under small changes of the input data. Continuity of the interpolating function is viewed in terms of small changes in the x-values, while stability is related to changes in the y-values of the input data. Typically, continuity is guaranteed, so we will not discuss it here. Instead, we will focus on stability, which is expressed by the derivative:

$$\frac{\partial f(x)}{\partial f_j} \tag{1}$$

Examples where such a quantity is relevant can be found in many places. For instance, in many financial application (e.g. portfolio replication and hedging) one needs to use

[1]The papers [4, 6] have become quite famous within the mathematical finance community, since their author have introduced a new interpolation method which is deeply intertwined with interest rates, forward rates, and discount factors. As noted e.g. in [7], this method sometimes produces discontinuous forward rates.

the so-called *PV01* (present value of one basis point[2]) or *DV01* (dollar value of one basis point) [11]. In these cases, the input y-values are the interest rates r_i given for specific maturities t_i (the x-values), and the associated continuous curve $r(t)$ is known as the yield curve. In this example, the derivative (1) becomes:

$$\frac{\partial r(t)}{\partial r_j}\,.$$

The main result of this note is that Hyman's constraint does not contribute to any calculation in the monotone-preserving cubic spline framework. In section 2, we define our set up and fix our notation, which follows [4]. Section 3 contains the main result. We show explicitly how Hyman's monotonicity constraint is redundant in the monotone-preserving cubic spline method. Moreover, we will see that an additional assumption about how to deal with non-differentiable locations must be included in order to get the correct answer for the sensitivities. Section 4 contains a summary of the work done and some conclusions. In Appendix A we show the Excel/VBA code used for testing purposes.

2 Set up and notation

In this section we review the monotone preserving cubic splines. The main reference are [4, 6]. We start with a mesh of data points $\{t_1, t_2, \ldots, t_n\}$ (we will think of the x-values as times) and corresponding values $\{f_1, f_2, \ldots, f_n\}$ for a generic but unknown function $f(t)$. Cubic splines are generically defined by piece-wise cubic polynomial that pass through consecutive points:

$$f(t) = a_i + b_i(t - t_i) + c_i(t - t_i)^2 + d_i(t - t_i)^3\,, \tag{2}$$

with $t \in [t_i, t_{i+1}]$ and $i = 1, \ldots, n$. We will use the following definitions:

$$h_i = t_{i+1} - t_i \tag{3}$$

$$m_i = \frac{f_{i+1} - f_i}{h_i}\,, \tag{4}$$

with $i = 1, \ldots, n-1$. The coefficients a_i, b_i, c_i, and d_i, depends on the details of the method, and are related to the values of $f(t)$ and its derivatives at the node points. In general,

$$a_i = f(t_i) \equiv f_i\,, \qquad b_i = f'(t_i)\,, \qquad \text{etc.} \tag{5}$$

where the prime denotes the derivative of the interpolating function $f(t)$ w.r.t. its argument t. Moreover, given a_i and b_i, we can express c_i and d_i as follows:

$$c_i = \frac{3m_i - b_{i+1} - 2b_i}{h_i} \tag{6}$$

$$d_i = \frac{b_{i+1} + b_i - 2m_i}{h_i^2}\,. \tag{7}$$

We can use (2) to compute the derivative

$$\frac{\partial f(t)}{\partial f_j} = \frac{\partial a_i}{\partial f_j} + \frac{\partial b_i}{\partial f_j}(t - t_i) + \frac{\partial c_i}{\partial f_j}(t - t_i)^2 + \frac{\partial d_i}{\partial f_j}(t - t_i)^3\,. \tag{8}$$

[2]1 basis point = 0.01%.

We have

$$\begin{aligned}
\frac{\partial a_i}{\partial f_j} &= \delta_i^j && (9)\\
\frac{\partial m_i}{\partial f_j} &= \frac{1}{h_i}\left(\delta_{i+1}^j - \delta_i^j\right) && (10)\\
\frac{\partial c_i}{\partial f_j} &= \frac{1}{h_i}\left(3\frac{\partial m_i}{\partial f_j} - \frac{\partial b_{i+1}}{\partial f_j} - 2\frac{\partial b_i}{\partial f_j}\right) && (11)\\
\frac{\partial d_i}{\partial f_j} &= \frac{1}{h_i^2}\left(\frac{\partial b_{i+1}}{\partial f_j} + \frac{\partial b_i}{\partial f_j} - 2\frac{\partial m_i}{\partial f_j}\right), && (12)
\end{aligned}$$

which depends on the matrix elements $\frac{\partial b_i}{\partial f_j}$. Here δ_i^j is the Kronecker delta, which is equal to one if $i = j$ and zero otherwise.

Due to our previous formulas, once the derivatives at the points, or equivalently the b_i coefficients, are specified, everything else is fixed. In particular, if we are interested in computing $\frac{\partial f(t)}{\partial f_j}$, then all the work will be in the calculation of the derivatives of b_i w.r.t. f_j. This calculation is however tricky if we use monotone preserving splines (or any other method which enforces monotonicity in a similar fashion), where the b_i's are non-differentiable functions of the f_j's (which involve the min and max functions). Let us consider this case in more detail.

2.1 Monotone preserving cubic splines

Let us start by recalling the formulas for the b_i's in the monotone preserving cubic spline method as defined in the Hagan-West paper [4]. First of all, at the boundaries:

$$b_1 = 0\,, \qquad b_n = 0\,. \tag{13}$$

For the internal data, if the curve is not monotone at t_i, i.e. $m_{i-1} \cdot m_i \leq 0$, then

$$b_i = 0 \qquad (\text{if } m_{i-1} \cdot m_i \leq 0)\,, \tag{14}$$

so that it will have a turning point there. Instead, if the trend is monotone at i, i.e. $m_{i-1} \cdot m_i > 0$, one defines

$$\beta_i = \frac{3m_{i-1} \cdot m_i}{\max(m_{i-1}, m_i) + 2\min(m_{i-1}, m_i)} \tag{15}$$

and

$$b_i = \begin{cases} \min\left(\max(0, \beta_i), 3\min(m_{i-1}, m_i)\right) & \text{if } m_{i-1},\, m_i > 0 \\ \max\left(\min(0, \beta_i), 3\max(m_{i-1}, m_i)\right) & \text{if } m_{i-1},\, m_i < 0 \end{cases} \tag{16}$$

The former choice is made when the curve is increasing (positive slopes), the latter when decreasing (negative slopes). Equation (16) represents the monotonicity constraint introduced by Hyman [8] and based on the Fritsch-Butland algorithm [3, 9, 10].

3 The main results

In the case of non-monotonic trend the b_i coefficients are zero, and hence also their derivatives. So we will focus on case of monotonic trend from now on. The main result will be that within the framework of the monotone preserving spline the Hyman adjustment (16) can be omitted, since

$$b_i = \beta_i \,, \tag{17}$$

if the trend is monotonic at the node i. As a trivial consequence of (17), one has

$$\frac{\partial b_i}{\partial f_j} = \frac{\partial \beta_i}{\partial f_j} \,. \tag{18}$$

As a side result, one finds [1] that, when computing the derivative above, special care is needed to handle the non-differentiable max and min functions. In fact, if $m_i = m_{i-1}$, the solution is to average the derivatives of m_i and m_{i-1}:

$$\frac{\partial}{\partial f_j} \max(m_{i-1}, m_i) = \frac{1}{2}\left(\frac{\partial m_i}{\partial f_j} + \frac{\partial m_{i-1}}{\partial f_j}\right) \tag{19a}$$

$$\frac{\partial}{\partial f_j} \min(m_{i-1}, m_i) = \frac{1}{2}\left(\frac{\partial m_i}{\partial f_j} + \frac{\partial m_{i-1}}{\partial f_j}\right) \tag{19b}$$

and similarly for the derivatives of the β_i's. This was confirmed in [1] by performing numerical checks. While (17) is limited to the monotone-preserving algorithm, (19) is always valid. The remaining of this section is dedicated to proving (17).

3.1 The proof of the main results

Here we recall the relevant formulas that have been worked out in detail in [1].

3.1.1 The Fritsch-Butland prescription

If we denote the denominator of the β_i's with $L \equiv \max(m_{i-1}, m_i) + 2\min(m_{i-1}, m_i)$, then the derivatives of (15) will be given by:

- $\frac{\partial \beta_i}{\partial f_i}$

$$\frac{\partial \beta_i}{\partial f_i} = \frac{3}{L^2} \cdot \left(\alpha \frac{m_i^2}{h_{i-1}} - \beta \frac{m_{i-1}^2}{h_i}\right) \,,$$

where

$$\alpha = \begin{cases} 2 & \text{if } m_{i-1} > m_i \\ 1 & \text{if } m_{i-1} < m_i \end{cases} \qquad \text{and} \qquad \beta = \begin{cases} 1 & \text{if } m_{i-1} > m_i \\ 2 & \text{if } m_{i-1} < m_i \end{cases}$$

or explicitly

$$\frac{\partial \beta_i}{\partial f_i} = \frac{3}{L^2} \cdot \begin{cases} \frac{2m_i^2}{h_{i-1}} - \frac{m_{i-1}^2}{h_i} & \text{if } m_{i-1} > m_i \\ \frac{m_i^2}{h_{i-1}} - \frac{2m_{i-1}^2}{h_i} & \text{if } m_{i-1} < m_i \,. \end{cases} \tag{20}$$

- $\frac{\partial \beta_i}{\partial f_{i-1}}$

$$\frac{\partial \beta_i}{\partial f_{i-1}} = -k \cdot \frac{3}{L^2} \cdot \frac{m_i^2}{h_{i-1}} \,,$$

where

$$k = \begin{cases} 2 & \text{if } m_{i-1} > m_i \\ 1 & \text{if } m_{i-1} < m_i \,, \end{cases}$$

or explicitly

$$\frac{\partial \beta_i}{\partial f_{i-1}} = -\frac{3}{L^2} \cdot \frac{m_i^2}{h_{i-1}} \cdot \begin{cases} 2 & \text{if } m_{i-1} > m_i \\ 1 & \text{if } m_{i-1} < m_i \,. \end{cases} \tag{21}$$

- $\frac{\partial \beta_i}{\partial f_{i+1}}$

$$\frac{\partial \beta_i}{\partial f_{i+1}} = k \cdot \frac{3}{L^2} \cdot \frac{m_{i-1}^2}{h_i} \,,$$

where

$$k = \begin{cases} 1 & \text{if } m_{i-1} > m_i \\ 2 & \text{if } m_{i-1} < m_i \,, \end{cases}$$

or explicitly

$$\frac{\partial \beta_i}{\partial f_{i+1}} = \frac{3}{L^2} \cdot \frac{m_{i-1}^2}{h_i} \cdot \begin{cases} 1 & \text{if } m_{i-1} > m_i \\ 2 & \text{if } m_{i-1} < m_i \,. \end{cases} \tag{22}$$

- $\frac{\partial \beta_i}{\partial f_j}$

$$\frac{\partial \beta_i}{\partial f_j} = 0 \,, \qquad \text{if} \quad j \neq i-1, i, i+1 \,. \tag{23}$$

As already mentioned, if $m_{i-1} = m_i$, then $\frac{\partial \beta_i}{\partial f_j}$ will be the average of the answers corresponding to the two options $m_{i-1} < m_i$ and $m_{i-1} > m_i$.

3.2 The theorem

In this section we will prove that in the locally-monotonic case one has:

$$b_i = \beta_i \,. \tag{24}$$

Let us consider the case of local monotonicity at the node i and let us define the following four regions in the parameter space that are relevant for the Hyman adjustment:

i) $\beta_i > 0$ and $\beta_i < 3\min(m_{i-1}, m_i)$

ii) $\beta_i > 0$ and $\beta_i > 3\min(m_{i-1}, m_i)$

iii) $\beta_i < 0$ and $\beta_i > 3\max(m_{i-1}, m_i)$

iv) $\beta_i < 0$ and $\beta_i < 3\max(m_{i-1}, m_i)$

The first two region refer to the case of increasing trend ($m_i, m_{i-1}, b_i > 0$), while the remaining two are for the case of decreasing trend ($m_i, m_{i-1}, b_i < 0$).

Theorem 1. *Given β_i as defined in (15), the following statements hold both true*

- *if m_{i-1}, $m_i > 0$, then $\beta_i < 3\min(m_{i-1}, m_i)$;*
- *if m_{i-1}, $m_i < 0$, then $\beta_i > 3\max(m_{i-1}, m_i)$.*

Proof. The proof is straightforward.

Consider the monotonically increasing case first, $m_{i-1},\ m_i > 0$. By working out the inequality

$$\beta_i < 3\min(m_{i-1}, m_i)$$

and using the positivity of the denominator in (15), we end up with:

$$0 < 2\left(\min(m_{i-1}, m_i)\right)^2 . \tag{25}$$

which is always satisfied.

Similarly, for the monotonically decreasing case $m_{i-1},\ m_i < 0$, by working out the inequality

$$\beta_i > 3\max(m_{i-1}, m_i)$$

and using the negativity of the denominator in (15), we end up with:

$$0 < \left(\max(m_{i-1}, m_i)\right)^2 + m_{i-1}m_i \tag{26}$$

which is always satisfied. □

Theorem 2 (Main Result)**.** *In the monotone-preserving cubic spline interpolation method that uses Hyman monotonicity constraint (15)-(16), if the data trend is locally monotonic at node i, then we have*

$$b_i = \beta_i \tag{27}$$

else

$$b_i = 0\,. \tag{28}$$

Proof. This follows from the calculation:

$$\begin{aligned} b_i &= \begin{cases} \min\left(\max(0, \beta_i), 3\min(m_{i-1}, m_i)\right) & \text{if } m_{i-1},\ m_i > 0 \\ \max\left(\min(0, \beta_i), 3\max(m_{i-1}, m_i)\right) & \text{if } m_{i-1},\ m_i < 0 \end{cases} && (29) \\ &= \begin{cases} \min\left(\beta_i, 3\min(m_{i-1}, m_i)\right) & \text{if } m_{i-1},\ m_i > 0 \\ \max\left(\beta_i, 3\max(m_{i-1}, m_i)\right) & \text{if } m_{i-1},\ m_i < 0 \end{cases} && (30) \\ &= \begin{cases} \beta_i & \text{if } m_{i-1},\ m_i > 0 \\ \beta_i & \text{if } m_{i-1},\ m_i < 0 \end{cases} && (31) \\ &= \beta_i\,. && (32) \end{aligned}$$

In going from (29) to (30) we have used the fact that β_i is positive (negative) if the trend is increasing (decreasing), while from (30) to (31) we have used Theorem 1. Hence, in the case of local monotonicity we are left with

$$b_i = \beta_i\,. \tag{33}$$

□

Therefore, the whole Hyman constraint (16) is completely superfluous in the monotone-preserving spline and the identity between the derivatives follows trivially:

$$\frac{\partial b_i}{\partial f_j} = \frac{\partial \beta_i}{\partial f_j}\,, \tag{34}$$

where the r.h.s. is given by formulas (20)-(23).

As a general remark, the Hyman constraint (16) is trivial when the β_i's are defined by the Fritsch-Butland algorithm [10] as in (15). This is what happens for example in the notation of Hagan and West [4]. The reason for this identity is the fact that the β_i's as defined by the Fritsch-Butland algorithm already guarantee that the resulting curve will be monotonic. However, one can still implement the Hyman constraint (16) to ensure monotonicity in a non-trivial way when different first derivatives β_i at the node points are chosen, and in that case the identities (33) and (34) will not hold anymore.

4 Summary and conclusions

In this paper we have consider monotone preserving interpolation methods and shown that Hyman's monotonicity constraint does not contribute within the framework of the monotone-preserving cubic spline.

However this is generically not true anymore when the Hyman constraint is applied on any other spline in order to construct a monotonic curve. In fact, in this case Theorem 1 does not need to hold and consequently regions ii) and iv) will in general contribute. In this case one will need to use to complete formulas for all the four regions.

This result is important for practical as well as conceptual reasons. Moreover this quantity is important in many areas (e.g. in finance for pricing and for risk management of interest rate derivatives).

Acknowledgements

We would like to thank Patrick Hagan for discussions, the ABN AMRO Model Validation team for providing us with feedback and Dmytro Makogon in particular for his useful comments and remarks. We are also grateful with the Lawrence Livermore National Laboratory for sharing with us copies of the their reports.

References

[1] M. Maio, J. de Jong (2014), *On the Non-differentiability of Hyman's Monotonicity Constraint*, pre-print nr UD2014-01 http://uglyduckling.nl/library_files/PRE-PRINT-UD-2014-01.pdf.

[2] W. H. Press , S. A. Teukolsky , W. T. Vetterling , B. P. Flannery (2007), *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, Third Edition, 1256 pp., ISBN-10: 0521880688 http://www.nr.com/.

[3] F. N. Fritsch, R. E. Carlson (1980), *Monotone Piecewise Cubic Interpolation*, SIAM J. Numer. Anal., 17 (1980) 238-246 http://epubs.siam.org/doi/abs/10.1137/0717021.

[4] P. S. Hagan, G. West (2006), *Interpolation Methods for Curve Construction*, Applied Mathematical Finance, 13: 2, 89 - 129 http://dx.doi.org/10.1080/13504860500396032, http://finmod.co.za/Hagan_West_curves_AMF.pdf.

[5] T. M. Lehmann, C. Gonner, K. Spitzer (1999), *Survey: Interpolation Methodsin Medical Image Processing*, IEEE Transactions on Medical Imaging, Vol. 18, No. 11, 1049

- 1075 http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=816070&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D816070.

[6] P. S. Hagan, G. West (2008), *Methods for Constructing a Yield Curve*, Wilmott Magazine, 70 - 81 http://www.math.ku.dk/~rolf/HaganWest.pdf.

[7] P. F. du Preez, E. Mare (2013), *Interpolating Yield Curve Data in a Manner that Ensures Positive and Continuous Forward Curves*, SAJEMS NS 16 (2013) No 4:395-406 http://sajems.org/index.php/sajems/article/view/388.

[8] J. M. Hyman (1983), *Accurate Monotonicity Preserving Cubic Interpolation*, SIAM J. Sci. Stat. Comput. Vol. 4, No. 4, 645 - 654 http://www.researchgate.net/publication/230676072_Accurate_monotonicity_preserving_cubic_interpolation/file/d912f50c0a8c9585cd.pdf.

[9] J. Butland (1980), *A Method of Interpolating Reasonable-Shaped Curves Through Any Data*, Proceedings of Computer Graphics 80, Online Publications Ltd, 409-422.

[10] F. N. Fritsch, J. Butland (1980), *An Improved Monotone Piecewise Cubic Interpolation Algorithm*, Lawrence Livermore National Laboratory preprint UCRL-85104.

[11] M. Maio, J. de Jong (2014), *Geometry of Interest Rate Risk*, pre-print nr UD2014-02, http://uglyduckling.nl/library_files/PRE-PRINT-UD-2014-02.pdf.

A Appendix: VBA Code

In this appendix we show the Excel/VBA code that can be used to test our results. Here we will only reproduce the most important methods. The complete code, together with a working spreadsheet and examples, can be found at the link http://uglyduckling.nl/library_files/PRE-PRINT-UD-2014-03_COMPANION-SPREADSHEET.xlsm.

The first step is to compute the b_i coefficients. This calculation can be done using equations (13)-(16) and is implemented into the `B_Coefficients` function. Within this function one can choose whether to switch on or off the Hyman constraint (16), enclosed in the `HymanConstraint` method. If the constraint is turned off, then only the Fritsch-Butland prescription is used with $b_i = \beta_i$ (15). The boolean parameter `isHymanEnforced` is used for this purpose and is specified in the initialization method.

```
Private Function HymanConstraint(index As Integer, b() As Double) As Double
    Dim max As Double, min As Double

    max = WorksheetFunction.max(m(index), m(index - 1))
    min = WorksheetFunction.min(m(index - 1), m(index))

    If m(index) > 0 Then
        HymanConstraint = WorksheetFunction.min(WorksheetFunction.max(0, b(index)), 3 * min)
    End If

    If m(index) < 0 Then
       HymanConstraint = WorksheetFunction.max(WorksheetFunction.min(0, b(index)), 3 * max)
    End If
End Function
```

```
Private Function B_coefficients() As Double()
    Dim i As Integer, n As Integer
    Dim max As Double, min As Double

    If numberOfNodes > 2 Then
        n = numberOfNodes - 1

        Dim b() As Double
        ReDim b(0 To n)
        b(0) = 0#
        b(n) = 0#

        For i = 1 To n - 1
            If m(i - 1) * m(i) <= 0 Then
                b(i) = 0
            Else
                max = WorksheetFunction.max(m(i), m(i - 1))
                min = WorksheetFunction.min(m(i - 1), m(i))

                b(i) = 3 * m(i - 1) * m(i) / (max + 2 * min)

                If isHymanEnforced Then
                    b(i) = HymanConstraint(i, b)
                End If
            End If
        Next i
    End If

    B_coefficients = b
End Function
```

A crucial ingredient of the code is the `FindLowerIndexWithBinarySearch` method, which performs a search using the binary algorithm and returns the index i of the interval that contains x, given any input value x, i.e. $x \in [x_i, x_{i+1})$. For x values that fall outside the input range, we use flat extrapolation. We can then compute all the necessary quantities, such as the length of the relevant interval (3) with method `h(i)`, its slope (4) with method `m(i)`, and the remaining interpolation coefficients (6) and (7) with methods `c(i)` and `d(i)` respectively. Finally we apply the `Interpolate` method that computes the interpolated function for any input value.

```
Public Function Interpolate(xInput As Double) As Double
    Dim yOutput As Double, index As Integer

    If numberOfNodes < 3 Then
        MsgBox "At least 3 points needed for Monotone Preserving"
        Exit Function
    End If

    If xInput >= xValues(numberOfNodes - 1) Then
        yOutput = yValues(numberOfNodes - 1)
    ElseIf xInput < xValues(0) Then
        yOutput = yValues(0)
    Else
        index = FindLowerIndexWithBinarySearch(xInput)
        yOutput = yValues(index)
            + b(index) * (xInput - xValues(index))
            + C_coefficients(index) * (xInput - xValues(index))^2
            + D_coefficients(index) * (xInput - xValues(index))^3
    End If

    Interpolate = yOutput
 End Function
```