Methods for Constructing a Yield Curve

Patrick S. Hagan
Chief Investment Office, JP Morgan 100 Wood Street London, EC2V 7AN, England,
e-mail: patrick.s.hagan@jpmorgan.com
Graeme West
Financial Modelling Agency, 19 First Ave East, Parktown North, 2193, South Africa
e-mail: graeme@finmod.co.za, www.finmod.co.za

Abstract
In this paper we survey a wide selection of the interpolation algorithms that are in use in financial markets for construction of curves such as forward curves, basis curves, and most importantly, yield curves. In the case of yield curves we also review the issue of bootstrapping and discuss how the interpolation algorithm should be intimately connected to the bootstrap itself.
As we will see, many methods commonly in use suffer from problems: they posit unreasonable expections, or are not even necessarily arbitrage free. Moreover, many methods result in material variation in large sections of the curve when only one input is perturbed (the method is not local). In Hagan and West [2006] we introduced two new interpolation methods—the monotone convex method and the minimal method. In this paper we will review the monotone convex method and highlight why this method has a very high pedigree in terms of the construction quality criteria that one should be interested in.

Keywords
yield curve, interpolation, fixed income, discount factors

1 Basic Yield Curve Mathematics

Much of what is said here is a reprise of the excellent introduction in [Rebonato, 1998, §1.2].

The term structure of interest rates is defined as the relationship between the yield-to-maturity on a zero coupon bond and the bond's maturity. If we are going to price derivatives which have been modelled in continuous-time off of the curve, it makes sense to commit ourselves to using continuously-compounded rates from the outset.

Now is denoted time 0. The price of an instrument which pays 1 unit of currency at time t—such an instrument is called a discount or zero coupon bond—is denoted $Z(0, t)$. The inverse of this amount could be denoted $C(0, t)$ and called the capitalisation factor: it is the redemption amount earned at time t from an investment at time 0 of 1 unit of currency in said zero coupon bonds. The first and most obvious fact is that $Z(0, t)$ is decreasing in t (equivalently, $C(0, t)$ is increasing). Suppose $Z(0,t_1) < Z(0,t_2)$ for some $t_1 < t_2$. Then the arbitrageur will buy a zero coupon bond for time t_1, and sell one for time t_2, for an immediate income of $Z(0,t_2)—Z(0,t_1) > 0$. At time t_1 they will receive 1 unit of currency from the bond they have bought, which they could keep under their bed for all we care until time t_2, when they deliver 1 in the bond they have sold.

What we have said so far assumes that such bonds do trade, with sufficient liquidity, and as a continuum i.e. a zero coupon bond exists for every redemption date t. In fact, such bonds rarely trade in the market. Rather what we need to do is impute such a continuum via a process known as bootstrapping.

It is more common for the market practitioner to think and work in terms of continuously compounded rates. The time 0 continuously compounded risk free rate for maturity t, denoted $r(t)$, is given by the relationship

$$C(0, t) = \exp(r(t)t) \tag{1}$$

$$Z(0, t) = \exp(-r(t)t) \tag{2}$$

$$r(t) = -\frac{1}{t} \ln Z(0, t) \tag{3}$$

In so-called normal markets, yield curves are upwardly sloping, with longer term interest rates being higher than short term. A yield curve which is downward sloping is called inverted. A yield curve with one or more turning points is called mixed. It is often stated that such mixed yield curves are signs of market illiquidity or instability. This is not the case. Supply and demand for the instruments that are used to bootstrap the curve may simply imply such shapes. One can, in a stable market with reasonable liquidity, observe a consistent mixed shape over long periods of time.

Figure 1: The arbitrage argument that shows that Z(0, t) must be decreasing.

The shape of the graph for $Z(0, t)$ does not reflect the shape of the yield curve in any obvious way. As already mentioned, the discount factor curve must be monotonically decreasing whether the yield curve is normal, mixed or inverted. Nevertheless, many bootstrapping and interpolation algorithms for constructing yield curves miss this absolutely fundamental point.

Interestingly, there will be at least one class of yield curve where the above argument for a decreasing Z function does not hold true — a real (inflation linked) curve. Because the actual size of the cash payments that will occur are unknown (as they are determined by the evolution of a price index, which is unknown) the arbitrage argument presented above does not hold. Thus, for a real curve the Z function is not necessarily decreasing (and empirically this phenomenon does on occasion occur).

1.1 Forward rates

If we can borrow at a known rate at time 0 to date t_1, and we can borrow from t_1 to t_2 at a rate known and fixed at 0, then effectively we can borrow at a known rate at 0 until t_2. Clearly

$$Z(0, t_1)Z(0; t_1, t_2) = Z(0, t_2) \tag{4}$$

is the no arbitrage equation: $Z(0; t_1, t_2)$ is the forward discount factor for the period from t_1 to t_2—it has to be this value at time 0 with the information available at that time, to ensure no arbitrage.

The forward rate governing the period from t_1 to t_2, denoted $f(0; t_1, t_2)$ satisfies

$$\exp(-f(0; t_1, t_2)(t_2 - t_1)) = Z(0; t_1, t_2)$$

Immediately, we see that forward rates are positive (and this is equivalent to the discount function decreasing). We have either of

$$f(0; t_1, t_2) = -\frac{ln(Z(0, t_2)) - ln(Z(0, t_1))}{t_2 - t_1} \tag{5}$$

$$= \frac{r_2 t_2 - r_1 t_1}{t_2 - t_1} \tag{6}$$

Let the instantaneous forward rate for a tenor of t be denoted f(t), that is, $f(t) = \lim_{\epsilon \downarrow 0} f(0; t, t + \epsilon)$, for whichever t this limit exists. Clearly then

$$f(t) = -\frac{d}{dt} \ln(Z(t)) \tag{7}$$

$$= \frac{d}{dt} r(t)t \tag{8}$$

So $f(t) = r(t) + r'(t)t$, so the forward rates will lie above the yield curve when the yield curve is normal, and below the yield curve when it is inverted. By integrating,[1]

$$r(t)t = \int_0^t f(s)ds \tag{9}$$

$$Z(t) = \exp\left(-\int_0^t f(s)ds\right) \tag{10}$$

Also

$$\frac{r_i t_i - r_{i-1} t_{i-1}}{t_i - t_{i-1}} = \frac{1}{t_i - t_{i-1}} \int_{t_{i-1}}^{t_i} f(s)ds \tag{11}$$

which shows that the average of the instantaneous forward rate over any of our intervals $[t_{i-1}, t_i]$ is equal to the discrete forward rate for that interval. Finally,

$$r(t)t = r_{i-1} t_{i-1} + \int_{t_{i-1}}^{t} f(s)ds, \ t \in [t_{i-1}, t_i] \tag{12}$$

which is a crucial interpolation formula: given the forward function we easily find the risk free function.

2 Interpolation And Bootstrap Of Yield Curves—Not Two Separate Processes

As has been mentioned, many interpolation methods for curve construction are available. What needs to be stressed is that in the case of bootstrapping yield curves, the interpolation method is intimately connected to the bootstrap, as the bootstrap proceeds with incomplete information. This information is 'completed' (in a non unique way) using the interpolation scheme.

In Hagan and West [2006] we illustrated this point using swap curves; here we will make the same points focusing on a bond curve. Suppose we have a reasonably small set of bonds that we want to use to bootstrap the yield curve. (To decide which bonds to include can be a non-trivial exercise. Excluding too many runs the risk of disposing of market information which is actually meaningful, on the other hand, including too many could result in a yield curve which is implausible, with a multitude of turning points, or even a bootstrap algorithm which fails to converge.) Recall that we insist that whatever instruments are included will be priced perfectly by the curve.

Typically some rates at the short end of the curve will be known. For example, some zero-coupon bonds might trade which give us exact rates. In some markets, where there is insufficient liquidity at the short end, some inter-bank money market rates will be used.

Each bond and the curve must satisfy the following relationship:

$$[\mathbb{A}] = \sum_{i=0}^{n} p_i Z(0; t_{\text{settle}}, t_i)$$

where

- $\mathbb{A}$ is the all-in (dirty) price of the bond;
- t_{settle} is the date on which the cash is actually delivered for a purchased bond;
- $p_0, p_1, \ldots, p_n$ are the cash flows associated with a unit bond (typically $p_0 = e\frac{c}{2}$, $p_i = \frac{c}{2}$ for $1 \le i < n$ and $p_n = 1 + \frac{c}{2}$ where c is the annual coupon and e is the cum-ex switch);
- $t_0, t_1, \ldots, t_n$ are the dates on which those cash flows occur.

On the left is the price of the bond trading in the market. On the right is the price of the bond as stripped from the yield curve. We rewrite this in the computationally more convenient form

$$[\mathbb{A}]\, Z(0, t_{\text{settle}}) = \sum_{i=0}^{n} p_i Z(0, t_i) \tag{13}$$

Suppose for the moment that the risk free rates (and hence the discount factors) have been determined at $t_0, t_1, \ldots, t_{n-1}$. Then we solve $Z(0, t_n)$ easily as

$$Z(0, t_n) = \frac{1}{p_n}\left[[\mathbb{A}]\, Z(0, t_{\text{settle}}) - \sum_{i=0}^{n-1} p_i Z(0, t_i)\right]$$

which is written in the form of risk-free rates rather than discount factors as

$$r_n = \frac{1}{t_n}\left[\ln p_n - \ln\left[[\mathbb{A}]\, e^{-r_{\text{settle}}\, t_{\text{settle}}} - \sum_{i=0}^{n-1} p_i e^{-r_i t_i}\right]\right] \tag{14}$$

where the t_i's are now denominated in years and the relevant day-count convention is being adhered to. Of course, in general, we do not know the earlier rates, neither exactly (because it is unlikely that any money market instruments expire exactly at t_i) nor even after some interpolation (the rates for the smallest few t_i might be available after interpolation, but the later ones not at all). However, as in the case of swap curves, (14) suggests an iterative solution algorithm: we guess r_n, indeed other expiry-date rates for other bonds, and take the rates already known from e.g. the money market, and insert these rates into our interpolation algorithm. We then determine r_{settle} and $r_0, r_1, \ldots, r_{n-1}$. Next, we insert these rates into the right-hand side of (14) and solve for r_n. We then take this new guess for this bond, and for all the other bonds, and again apply the interpolation algorithm. We iterate this process. Even for fairly wild curves (such as can often be the case in South Africa) this iteration will reach a fixed point with accuracy of about 8 decimal places in 4 or 5 iterations. This then is our yield curve.

3 How To Compare Yield Curve Interpolation Methodologies

In general, the interpolation problem is as follows: we have some data x as a function of time, so we have $\tau_1, \tau_2, \ldots, \tau_n$ and $x_1, x_2, \ldots, x_n$ known. An interpolation method is one that constructs a continuous function $x(t)$ satisfying $x(\tau_i) = x_i$ for $i = 1, 2, \ldots, n$. In our setting, the x values might be risk free rates, forward rates, or some transformation of these—the log of rates, etc. Of course, many choices of interpolation function are possible — according to the nature of the problem, one imposes requirements additional to continuity, such as differentiability, twice differentiability, conditions at the boundary, and so on.

The Lagrange polynomial is a polynomial of degree $n - 1$ which passes through all the points, and of course this function is smooth. However, it is well known that this function is inadequate as an interpolator, as it demonstrates remarkable oscillatory behaviour.

The typical approach is to require that in each interval the function is described by some low dimensional polynomial, so the requirements of continuity and differentiability reduce to linear equations in the coefficients, which are solved using standard linear algebraic techniques. The simplest example are where the polynomials are linear, and these methods are surveyed in §4. However, these functions clearly will not be differentiable. Next, we try quadratics—however here we have a remarkable 'zig-zag' instability which we will discuss. So we move on to cubics—or even quartics—they overcome these already-mentioned difficulties, and we will see these in §5.

All of the interpolation methods considered in Hagan and West [2006] appear in the rows of Table 1.

We will restrict attention to the case where the number of inputs is reasonably small and so the bootstrapping algorithm is able to price the instruments exactly, and we restrict attention to those methods where the instruments are indeed always priced exactly.

The criteria to use in judging a curve construction and its interpolation method that we will consider are:

(a) In the case of yield curves, how good do the forward rates look? These are usually taken to be the 1m or 3m forward rates, but these are virtually the same as the instantaneous rates. We will want to have positivity and continuity of the forwards.

It is required that forwards be positive to avoid arbitrage, while continuity is required as the pricing of interest sensitive instruments is sensitive to the stability of forward rates. As pointed out in McCulloch and Kochin [2000], 'a discontinuous forward curve implies either implausible expectations about future short-term interest rates, or implausible expectations about holding period returns'. Thus, such an interpolation method should probably be avoided, especially when pricing derivatives whose value is dependent upon such forward values.

Smoothness of the forward is desirable, but this should not be achieved at the expense of the other criteria mentioned here.

(b) How local is the interpolation method? If an input is changed, does the interpolation function only change nearby, with no or minor spill-over elsewhere, or can the changes elsewhere be material?

(c) Are the forwards not only continuous, but also stable? We can quantify the degree of stability by looking for the maximum basis point change in the forward curve given some basis point change (up or down) in one of the inputs. Many of the simpler methods can have this quantity determined exactly, for others we can only derive estimates.

(d) How local are hedges? Suppose we deal an interest rate derivative of a particular tenor. We assign a set of admissible hedging instruments, for example, in the case of a swap curve, we might (even should)

decree that the admissible hedging instruments are exactly those instruments that were used to bootstrap the yield curve. Does most of the delta risk get assigned to the hedging instruments that have maturities close to the given tenors, or does a material amount leak into other regions of the curve?

We will now survey a handful of these methods, and highlight the issues that arise.

4 Linear Methods

4.1 Linear on rates

For $t_{i-1} < t < t_i$ the interpolation formula is

$$r(t) = \frac{t - t_{i-1}}{t_i - t_{i-1}} r_i + \frac{t_i - t}{t_i - t_{i-1}} r_{i-1} \tag{15}$$

Using (8) we get

$$f(t) = \frac{2t - t_{i-1}}{t_i - t_{i-1}} r_i + \frac{t_i - 2t}{t_i - t_{i-1}} r_{i-1} \tag{16}$$

Of course f is undefined at the t_i, as the function $r(t)t$ is clearly not differentiable there. Moreover, in the actual rate interpolation formula, by the time t reaches t_i, the import of r_{t-1} has been reduced to zero—that rate has 'been forgotten'. But we clearly see that this is not the case for the forward, so the left and right limits $f(t_i^+)$ and $f(t_i^-)$ are different—the forward jumps. Furthermore, the choice of interpolation does not prevent negative forward rates: suppose we have the (t, r) points (1y, 8%) and (2y, 5%). Of course, this is a rather contrived economy: the one year interest rate is 8% and the one year forward rate in one year's time is 2%. Nevertheless, it is an arbitrage free economy. But using linear interpolation the instantaneous forwards are negative from about 1.84 years onwards.

4.2 Linear on the log of rates

Now for $t_{i-1} \le t \le t_i$ the interpolation formula is

$$\ln(r(t)) = \frac{t - t_{i-1}}{t_i - t_{i-1}} \ln(r_i) + \frac{t_i - t}{t_i - t_{i-1}} \ln(r_{i-1})$$

which as a rate formula is

$$r(t) = r_i^{\frac{t - t_{i-1}}{t_i - t_{i-1}}} r_{i-1}^{\frac{t_i - t}{t_i - t_{i-1}}} \tag{17}$$

A simple objection to the above formula is that it does not allow negative interest rates. Also, the same argument as before shows that the forward jumps at each node, and similar experimentation will provide an example of a Z function which is not decreasing.

4.3 Linear on discount factors

Now for $t_{i-1} \le t \le t_i$ the interpolation formula is

$$Z(t) = \frac{t - t_{i-1}}{t_i - t_{i-1}} Z_i + \frac{t_i - t}{t_i - t_{i-1}} Z_{i-1}$$

which as a rate formula is

$$r(t) = \frac{-1}{t} \ln\left[\frac{t - t_{i-1}}{t_i - t_{i-1}} e^{-r_i t_i} + \frac{t_i - t}{t_i - t_{i-1}} r^{-r_{i-1} t_{i-1}} \right] \tag{18}$$

Again, the forward jumps at each node, and the Z function may not be decreasing.

4.4 Raw interpolation (linear on the log of discount factors)

This method corresponds to piecewise constant forward curves. This method is very stable, is trivial to implement, and is usually the starting point for developing models of the yield curve. One can often find mistakes in fancier methods by comparing the raw method with the more sophisticated method.

By definition, raw interpolation is the method which has constant instantaneous forward rates on every interval $t_{i-1} < t < t_i$. From (11) we see that that constant must be the discrete forward rate for the interval, so $f(t) = \frac{r_i t_i - r_{i-1} t_{i-1}}{t_i - t_{i-1}}$ for $t_{i-1} < t < t_i$. Then from (12) we have that

$$r(t)t = r_{i-1} t_{i-1} + (t - t_{i-1}) \frac{r_i t_i - r_{i-1} t_{i-1}}{t_i - t_{i-1}}$$

By writing the above expression with a common denominator of $t_i - t_{i-1}$, and simplifying, we get that the interpolation formula on that interval is

$$r(t)t = \frac{t - t_{i-1}}{t_i - t_{i-1}} r_i t_i + \frac{t_i - t}{t_i - t_{i-1}} r_{i-1} t_{i-1} \tag{19}$$

which explains yet another choice of name for this method: 'linear rt'; the method is linear interpolation on the points $r_i t_i$. Since $\pm r_i t_i$ is the logarithm of the capitalisation/discount factors, we see that calling this method 'linear on the log of capitalisation factors' or 'linear on the log of discount factors' is also merited.

This raw method is very attractive because with no effort whatsoever we have guaranteed that all instantaneous forwards are positive, because every instantaneous forward is equal to the discrete forward for the 'parent' interval. As we have seen, this is an achievement not to be sneezed at. It is only at the points $t_1, t_2, \ldots, t_n$ that the instantaneous forward is undefined, moreover, the function jumps at that point.

4.5 Piecewise linear forward

Having decided that the raw method is quite attractive, what happens if we try to remedy its only defect in the most obvious way? What we will do is that instead of the forwards being piecewise constant we will demand that they be a piecewise continuous linear function. What could be more natural than to simply ask to gently rotate the raw interpolants so that they are now not only piecewise linear, but continuous as well? Unfortunately, this very plausible requirement gives rise to at least two types of very unpleasant behaviour indeed. This is easily understood by means of an example.

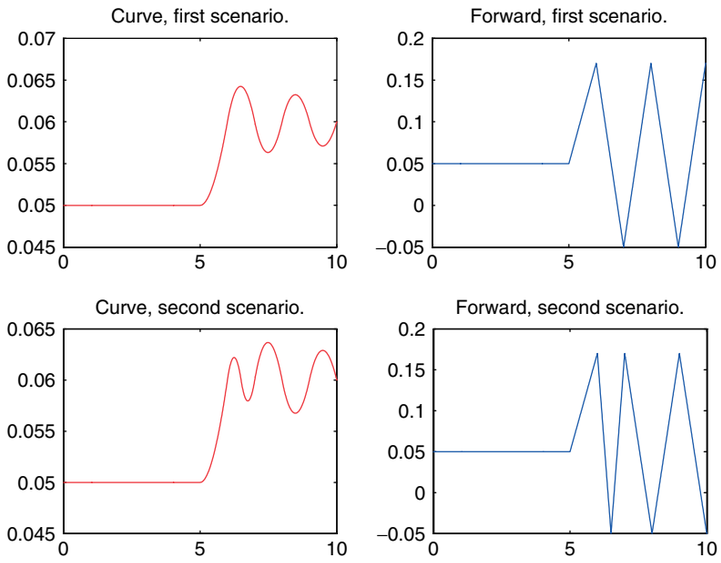

Figure 2: The piecewise linear forward method.

Firstly, suppose we have a curve with input zero coupon rates at every year node, with a value of $r(t) = 5\%$ for $t = 1, 2, \ldots, 5$ and $r(t) = 6\%$ for $t = 6, 7, \ldots, 10$. We must have $f(t) = r(1)$ for $t \leq 1$. In order to assure continuity, we see then we must have $f(t) = r(i)$ for every $i \leq 5$. Now, the discrete forward rate for [5, 6] is 11%. In order for the average of the piecewise linear function f on the interval [5, 6] to be 11% we must have that $f(6) = 17\%$. And now in turn, the discrete forward rate for [6, 7] is 6% and so in order for the average of the piecewise linear function f on the interval [6, 7] to be 6% we must have that $f(7) = -5\%$. This zig-zag feature continues recursively; see Figure 2. Note also the implausible shape of the actual yield curve itself.

Secondly, suppose now we include a new node, namely that $r(6.5) = 6\%$. It is fairly intuitive that this imparts little new information[2]. Nevertheless, the bootstrapped curve changes dramatically. The 'parity of the zig-zag' is reversed. So we see that the localness of the method is exceptionally poor.

5 Splines

The various linear methods are the simplest examples of polynomial splines: a polynomial spline is a function which is piecewise in each interval a polynomial, with the coefficients arranged to ensure at least that the spline coincides with the input data (and so is continuous). In the linear case that is all that one can do—the linear coefficients are now determined. If the polynomials are of higher degree, we can use up the degrees of freedom by demanding other properties, such as differentiability, twice differentiability, asymptotes at either end, etc.

The first thing we try is a quadratic spline.

5.1 Quadratic splines

To complete a quadratic spline of a function x, we desire coefficients (a_i, b_i, c_i) for $1 \leq i \leq n-1$. Given these coefficients, the function value at any term τ will be

$$x(\tau) = a_i + b_i(\tau - \tau_i) + c_i(\tau - \tau_i)^2 \quad \tau_i \leq \tau \leq \tau_{i+1} \tag{20}$$

The constraints will be that the interpolating function indeed meets the given data (and hence is continuous) and the entire function is differentiable. There are thus $3n-4$ constraints: $n-1$ left hand function values to be satisfied, $n-1$ right hand function values to be satisfied, and $n-2$ internal knots where differentiability needs to be satisfied. However, there are $3n-3$ unknowns. With one degree of freedom remaining, it makes sense to require that the left-hand derivative at τ_n be zero, so that the curve can be extrapolated with a horizontal asymptote.

Suppose we apply this method to the rates (so $x_i = r_i$). The forward curves that are produced are very similar to the piecewise linear forward curves—the curve can have a 'zig-zag' appearance, and this zig-zag is subject to the same parity of input considerations as before.

So, next we try a cubic spline.

5.2 Cubic splines

This time we desire coefficients (a_i, b_i, c_i, d_i) for $1 \leq i \leq n-1$. Given these coefficients, the function value at any term τ will be

$$x(\tau) = a_i + b_i(\tau - \tau_i) + c_i(\tau - \tau_i)^2 + d_i(\tau - \tau_i)^3 \tau_i \leq \tau \leq \tau_{i+1} \tag{21}$$

As before we have $3n-4$ constraints, but this time there are $4n-4$ unknown coefficients. There are several possible ways to proceed to find another n constraints. Here are the ones that we have seen:

- $x_i = r_i$. The function is required to be twice differentiable, which for the same reason as previously adds another $n-2$ constraints. For the final two constraints, the function is required to be linear at the extremes i.e. the second derivative of the interpolator at τ_1 and at τ_n are zero. This is the so-called natural cubic spline.
- $x_i = r_i$. The function is again required to be twice differentiable; for the final two constraints we have that the function is linear on the left and horizontal on the right. This is the so-called financial cubic spline Adams [2001].
- $x_i = r_i\tau_i$. The function is again required to be twice differentiable; for the final two constraints we have that this function is linear on the right and quadratic on the left. This is the quadratic-natural spline proposed in McCulloch and Kochin [2000].
- $x_i = r_i$. The values of b_i for $1 < i < n$ are chosen to be the slope at τ_i of the quadratic that passes through (τ_j, r_j) for $j = i-1, i, i+1$. The value of b_1 is chosen to be the slope at τ_1 of the quadratic that passes through (τ_j, r_j) for $j = 1, 2, 3$; the value of b_n is chosen likewise. This is the Bessel method [de Boor, 1978, 2001, Chapter IV], although often somewhat irregularly called the Hermite method by software vendors.

- $x_i = r_i\tau_i$. Again, Bessel interpolation.
- Going one step further, quartic splines. According to Adams [2001] the quartic spline gives the smoothest interpolator of the forward curve. The spline can proceed on instantaneous forward rates, this time there are $5n-5$ unknowns and 3 additional conditions at τ_1 or τ_n required. Although one must ask: when does one actually have a set of instantaneous forwards as inputs for interpolation? Alternatively if we apply (9) then the inputs are risk free rates, and the spline is of the form $r(\tau) = \frac{a_i}{\tau} + b_i + c_i\tau + d_i\tau^2 + e_i\tau^3 + g_i\tau^4$, with $6n-6$ unknowns and 4 additional conditions required.
- $x_i = r_i$. The monotone preserving cubic spline of Hyman [1983]. The method specifies the values of b_i for $1 \le i \le n$, in a way to be discussed in more detail shortly.

Significant problems can become apparent when using some of these methods. The spline is supposed to alleviate the problem of oscillation seen when fitting a single polynomial to a data set (the Lagrange polynomial), nevertheless, significant oscillatory behaviour can still be present. Furthermore, the various types of clamping we see with some of the methods above (clamping refers to imposing conditions at the boundaries τ_1 or τ_n) can compromise localness of the interpolator, sometimes grossly. In fact, the iterative procedure from §2 often fails to converge for the quartic interpolation methods, and we exclude them from further analysis.

The method of Hyman is a method which attempts to address these problems. This method is quite different to the others; it is a local method—the interpolatory values are only determined by local behaviour, not global behaviour. This method ensures that in regions of monotonicity of the inputs (so, three successive increasing or decreasing values) the interpolating function preserves this property; similarly if the data has a minimum/maximum then the output interpolator will have a minimum/maximum at the node.

6 Monotone Convex

Many of the ideas of the method of Hyman will now have a natural development—the monotone convex method was developed to resolve the only remaining deficiency of Hyman [1983]. Very simply, none of the methods mentioned so far are aware that they are trying to solve a financial problem—indeed, the breeding ground for these methods is typically engineering or physics. As such, there is no mechanism which ensures that the forward rates generated by the method are positive, and some simple experimentation will uncover a set of inputs to a yield curve which give some negative forward rates under all of the methods mentioned here, as seen in Hagan and West [2006]. Thus, in introducing the monotone convex method, we use the ideas of Hyman [1983], but explicitly ensure that the continuous forward rates are positive (whenever the discrete forward rates are themselves positive).

The point of view taken in the monotone convex method is that the inputs are (or can be manipulated to be) discrete forwards belonging to intervals; the interpolation is not performed on the interest rate curve itself. We may have actual discrete forwards—FRA rates. On the other hand if we have interest rates $r_1, r_2, \ldots, r_n$ for periods $\tau_1, \tau_2, \ldots, \tau_n$ then the first thing we do is calculate $f_i^d = \frac{r_i\tau_i - r_{i-1}\tau_{i-1}}{\tau_i - \tau_{i-1}}$ for $1 \le i \le n$, $r_0 = 0$. (Here we also check that these are all positive, and so conclude that the curve is legal i.e. arbitrage free (except in those few cases where forward rates may be negative). As an interpolation algorithm the monotone convex method will now bootstrap a forward curve, and then if required recover the continuum of risk free rates using (12).

One rather simple observation is that all of the spline methods we saw in §5 fail in forward extrapolation beyond the interval $[\tau_1, \tau_n]$. Clearly if the interpolation is on rates then we will apply horizonal extrapolation to the rate outside of that interval: $r(\tau) = r_1$ for $\tau < \tau_1$ and $r(\tau) = r_n$ for $\tau > \tau_n$. So far so good. What happens to the forward rates? Perhaps surprisingly we cannot apply the same extrapolation rule to the forwards, in fact, we need to set $f(\tau) = r_1$ for $\tau < \tau_1$ and $f(\tau) = r_n$ for $\tau > \tau_n$—consider (8). This makes it almost certain that the forward curve has a material discontinuity at τ_1, and probably one at τ_n too (the latter will be less severe as the curve, either by design or by nature, probably has a horizontal asymptote as $\tau \uparrow \tau_n$).

In order to avoid this pathology, we now have terms $0 = \tau_0, \tau_1, \ldots, \tau_n$ and the generic interval for consideration is $[\tau_{i-1}, \tau_i]$. A 'short rate' (instantaneous) rate may be provided, if not, the algorithm will model one. Usually the shortest rate that might be input will be an overnight rate, if it is provided, the algorithm here simply has some 'overkill'—there will be an overnight rate and an instantaneous short rate—but it need not be modified.

f_i^d is the discrete rate which 'belongs' to the entire interval $[\tau_{i-1}, \tau_i]$; it would be a mistake to model that rate as being the instantaneous rate at τ_i. Rather, we begin by assigning it to the midpoint of the interval, and then modelling the instantaneous rate at τ_i. as being on the straight line that joins the adjacent midpoints. Let this rate $f(\tau_i)$ be denoted f_i. This explains (22). In (23) and (24) the values $f_0 = f(0)$ and $f_n = f(\tau_n)$ are selected so that $f\prime(0) = 0 = f\prime(\tau_n)$. Thus

$$f_i = \frac{\tau_i - \tau_{i-1}}{\tau_{i+1} - \tau_{i-1}} f_{i+1}^d + \frac{\tau_{i+1} - \tau_i}{\tau_{i+1} - \tau_{i-1}} f_i^d, \quad \textit{for } i = 1, 2, \ldots, n-1 \tag{22}$$

$$f_0 = f_1^d - \tfrac{1}{2}(f_1 - f_1^d) \tag{23}$$

$$f_n = f_n^d - \tfrac{1}{2}(f_{n-1} - f_n^d) \tag{24}$$

Note that if the discrete forward rates are positive then so are the f_i for $i = 1, 2, \ldots, n-1$.

We now seek an interpolatory function f defined on $[0, \tau_n]$ for $f_0, f_1, \ldots, f_n$ that satisfies the conditions below (in some sense, they are arranged in decreasing order of necessity).

(i) $\frac{1}{\tau_i - \tau_{i-1}} \int_{\tau_{i-1}}^{\tau_i} f(t)dt = f_i^d$, so the discrete forward is recovered by the curve, as in (11).
(ii) f is positive.
(iii) f is continuous.
(iv) If $f_{i-1}^d < f_i^d < f_{i+1}^d$ then $f(\tau)$ is increasing on $[\tau_{i-1}, \tau_i]$, and if $f_{i-1}^d > f_i^d > f_{i+1}^d$ then $f(\tau)$ is decreasing on $[\tau_{i-1}, \tau_i]$.

Let us first normalise things, so we seek a function g defined on [0, 1] such that[3]

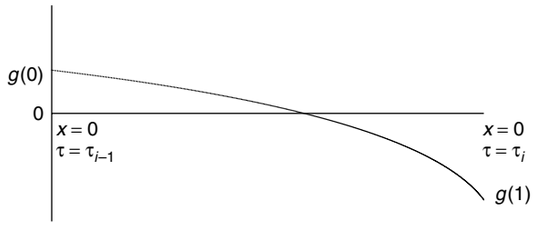

Figure 3: The function *g*.

$$g(x) = f(\tau_{i-1} + (\tau_i - \tau_{i-1})x) - f_i^d. \tag{25}$$

Before proceeding, let us give a sketch of how we will proceed. We will choose g to be piecewise quadratic in such a way that (i) is satisfied by construction. Of course, g is continuous, so (iii) is satisfied. As a quadratic, it is easy to perform an analysis of where the minimum or maximum occurs, and we thereby are able to apply some modifications to g to ensure that (iv) is satisfied, while ensuring (i) and (iii) are still satisfied.

Also, we see a posteriori that if the values of f_i had satisfied certain constraints, then (ii) would have been satisfied. So, the algorithm will be to construct (22), (23) and (24), then modify the f_i to satisfy those constraints, then construct the quadratics, and then modify those quadratics. Finally,

$$f(\tau) = g\left(\frac{\tau - \tau_{i-1}}{\tau_i - \tau_{i-1}}\right) + f_i^d. \tag{26}$$

Thus, the current choices of f_i are provisional; we might make some adjustments in order to guarantee the positivity of the interpolating function f.

Here follow the details. We have only three pieces of information about g: $g(0) = f_{i-1} - f_i^d$, $g(1) = f_i - f_i^d$, and $\int_0^1 g(x)dx = 0$. We postulate a functional form $g(x) = K + Lx + Mx^2$, having 3 equations in 3 unknowns we get $\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & \frac{1}{2} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} K \\ L \\ M \end{bmatrix} = \begin{bmatrix} g(0) \\ g(1) \\ 0 \end{bmatrix}$, and easily solve to find that

$$g(x) = g(0)[1 - 4x + 3x^2] + g(1)[-2x + 3x^2] \tag{27}$$

Note that by (22) that (iv) is equivalent to requiring that if $f_{i-1} < f_i^d < f_i$ then $f(\tau)$ is increasing on $[\tau_{i-1}, \tau_i]$, while if $f_{i-1} > f_i^d > f_i$ then $f(\tau)$ is decreasing on $[\tau_{i-1}, \tau_i]$. This is equivalent to requiring that if $g(0)$ and $g(1)$ are of opposite sign then g is monotone.
Now

$$g'(x) = g(0)(-4 + 6x) + g(1)(-2 + 6x)$$
$$g'(0) = -4g(0) - 2g(1)$$
$$g'(1) = 2g(0) + 4g(1)$$

g being a quadratic it is now easy to determine, simply by inspecting $g'(0)$ and $g'(1)$, the behaviour of g on [0, 1]. The cases where $g'(0) = 0$ and $g'(1) = 0$ are crucial; these correspond to $g(1) = -2g(0)$ and $g(0) = -2g(1)$ respectively. These two lines divide the g(0)/g(1) plane into eight sectors. We seek to modify the definition of g on each sector, taking care that on the boundary of any two sectors, the formulae from those two sectors actually coincide (to preserve continuity). In actual fact the treatment for every diametrically opposite pair of sectors is the same, so we really have four cases to consider, as follows (refer Figure 4):

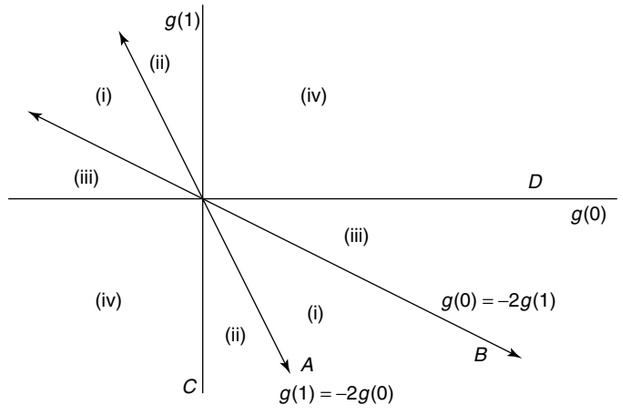

Figure 4: The reformulated possibilities for *g*.

(i) In these sectors $g(0)$ and $g(1)$ are of opposite signs and $g'(0)$ and $g'(1)$ are of the same sign, so g is monotone, and does not need to be modified.

(ii) In these sectors $g(0)$ and $g(1)$ are also of opposite sign, but $g'(0)$ and $g\prime(1)$ are of opposite sign, so g is currently not monotone, but needs to be adjusted to be so. Furthermore, the formula for (i) and for (ii) need to agree on the boundary $\mathcal{A}$ to ensure continuity.

(iii) The situation here is the same as in the previous case. Now the formula for (i) and for (iii) need to agree on the boundary $\mathcal{B}$ to ensure continuity.

(iv) In these sectors $g(0)$ and $g(1)$ are of the same sign so at first it appears that g does not need to be modified. Unfortunately this is not the case: modification will be needed to ensure that the formula for (ii) and (iv) agree on $\mathcal{C}$ and (iii) and (iv) agree on $\mathcal{D}$.

The origin is a special case: if $g\prime(0) = 0 = g\prime(1)$ then $g(x) = 0$ for all x, and $f_{i-1}^d = f_i^d = f_{i+1}^d$, and we put $f(\tau) = f_i^d$ for $\tau \in [\tau_{i-1}, \tau_i]$.
So we proceed as follows:

(i) As already mentioned g does not need to be modified. Note that on $\mathcal{A}$ we have $g(x) = g(0)(1 - 3x^2)$ and on $\mathcal{B}$ we have $g(x) = g(0)(1 - 3x + \frac{3}{2}x^2)$.

(ii) A simple solution is to insert a flat segment, which changes to a quadratic at exactly the right moment to ensure that $\int_0^1 g(x)dx = 0$. So we take

$$g(x) = \begin{cases} g(0) & \text{for} \quad 0 \le x \le \eta \\ g(0) + (g(1) - g(0))\left(\frac{x-\eta}{1-\eta}\right)^2 & \text{for} \quad \eta < x \le 1 \end{cases} \tag{28}$$

$$\eta = 1 + 3\frac{g(0)}{g(1) - g(0)} = \frac{g(1) + 2g(0)}{g(1) - g(0)} \tag{29}$$

Note that $\eta \longrightarrow 0$ as $g(1) \longrightarrow -2g(0)$, so the interpolation formula reduces to $g(x) = g(0)(1 - 3x^2)$ at $\mathcal{A}$, as required.

(iii) Here again we insert a flat segment. So we take

$$g(x) = \begin{cases} g(1) + (g(0) - g(1)) \left(\frac{\eta - x}{\eta}\right)^2 & \text{for} \quad 0 < x < \eta \\ g(1) & \text{for} \quad \eta \le x < 1 \end{cases} \tag{30}$$

$$\eta = 3\frac{g(1)}{g(1) - g(0)} \tag{31}$$

Note that $\eta \longrightarrow 1$ as $g(1) \longrightarrow -\frac{1}{2}g(0)$, so the interpolation formula reduces to $g(x) = g(0)(1 - 3x + \frac{3}{2}x^2)$ at $\mathcal{B}$, as required.

(iv) We want a formula that reduces in form to that defined in (ii) as we approach $\mathcal{C}$, and to that defined in (iii) as we approach $\mathcal{D}$. This suggests

$$g(x) = \begin{cases} A + (g(0) - A) \left(\frac{\eta - x}{\eta}\right)^2 & \text{for} \quad 0 < x < \eta \\ A + (g(1) - A) \left(\frac{x - \eta}{1 - \eta}\right)^2 & \text{for} \quad \eta < x < 1 \end{cases} \tag{32}$$

where $A = 0$ when $g(1) = 0$ - so the first line satisfies (iii)) and $A = 0$ when $g(0) = 0$ (so the second line satisfies (ii). Straightforward calculus gives

$$\int_0^1 g(x)dx = \frac{2}{3}A + \frac{\eta}{3}g(0) + \frac{1 - \eta}{3}g(1)$$

and so

$$A = -\frac{1}{2}[\eta g(0) + (1 - \eta)g(1)]$$

A simple choice satisfying the various requirements is

$$\eta = \frac{g(1)}{g(1) + g(0)} \tag{33}$$

$$A = -\frac{g(0)g(1)}{g(0) + g(1)} \tag{34}$$

6.1 Ensuring positivity

Suppose we wish to guarantee that the interpolatory function f is everywhere positive.

Clearly from the formula (26) it suffices to ensure that $g(x) > -f_i^d$ for $x \in [0, 1]$. Now $g(0) = f_{i-1} - f_i^d > -f_i^d$ and $g(1) = f_i - f_i^d > -f_i^d$ since f_{i-1}, f_i are positive. Thus the inequality is satisfied at the endpoints of the interval. Now, in regions (i), (ii) and (iii), g is monotone, so those regions are fine.

In region (iv) g is not monotone. g is positive at the endpoints and has a minimum of A (as in (34)) at the x-value η (as in (33)). So, it now suffices to prove that $\frac{g(0)g(1)}{g(0)+g(1)} < f_i^d$. This is the case if $f_{i-1}, f_i < 3f_i^d$. To see this, note that then $0 < g(0), g(1) < 2f_i^d$ and the result follows, since if $0 < y, z < 2a$ then $\frac{y+z}{yz} = \frac{1}{z} + \frac{1}{y} > \frac{1}{2a} + \frac{1}{2a} = \frac{1}{a}$ and so $\frac{yz}{y+z} > a$.
We choose the slightly stricter condition $f_{i-1}, f_i < 2f_i^d$. Thus, our algorithm is

(1) Determine the f_i^d from the input data.
(2) Define f_i for $i = 0, 1, \ldots, n$ as in (22), (23) and (24).
(3) If f is required to be everywhere positive, then collar f_0 between 0 and $2f_1^d$, for $i = 1, 2, \ldots, n-1$ collar f_i between 0 and $2\min(f_i^d, f_{i+1}^d)$, and collar f_n between 0 and $2f_n^d$. If f is not required to be everywhere positive, simply omit this step.
(4) Construct g with regard to which of the four sectors we are in.
(5) Define f as in (26).
(6) If required recover r as in (12). Integration formulae are easily established as the functions forms of g are straightforward.

Pseudo-code for this recipe is provided in an Appendix. Working code for this interpolation scheme is available from the second author's website.

6.2 Amelioration

In Hagan and West [2006] an enhancement of this method is considered where the curve is ameliorated (smoothed). This is achieved by making the interpolation method slightly less local i.e. by using as inputs not only neighbouring information but also information which is two nodes away.

7 Hedging

We can now ask the question: how do we use the instruments which have been used in our bootstrap to hedge other instruments? In general

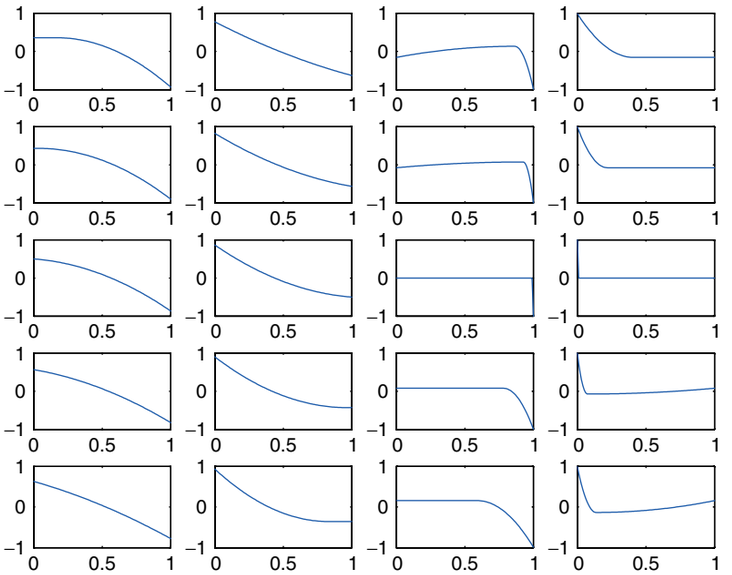

Figure 5: The g function as we cross the boundaries. From left to right: boundaries $\mathcal{A}, \mathcal{B}, \mathcal{C}$ and $\mathcal{D}$. From top to bottom: approaching the boundary, at the boundary (central), leaving the boundary. Only at the boundary of $\mathcal{C}$and $\mathcal{D}$ are there discontinuities.

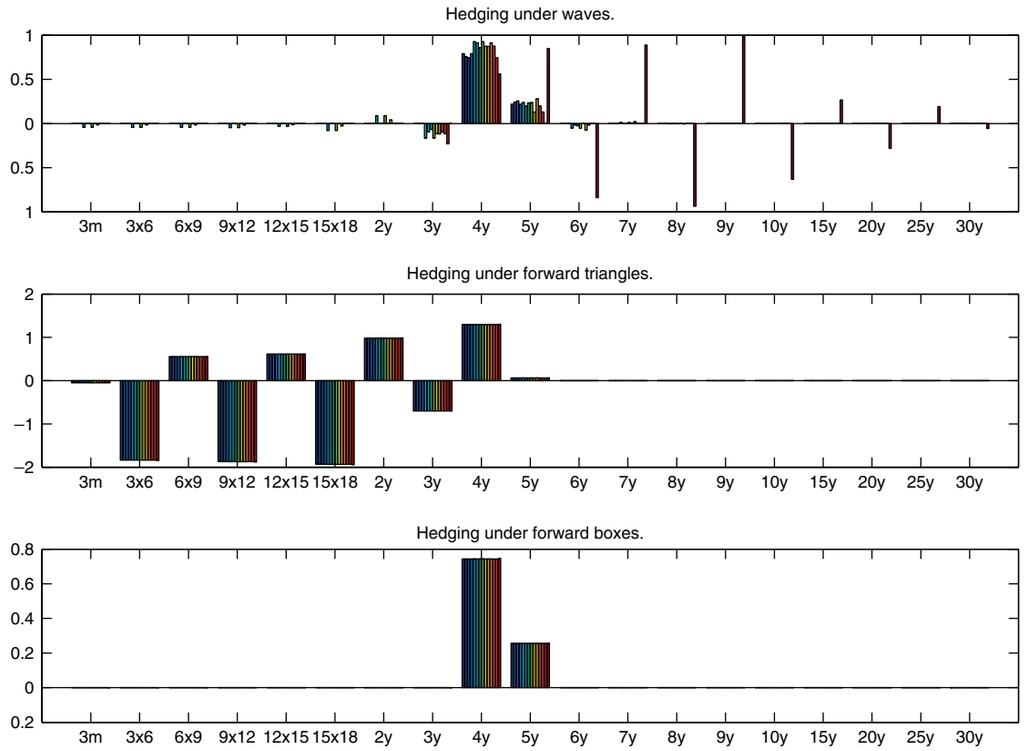

Figure 6: The obvious superiority of using forward boxes to determine hedge portfolios: not only is the hedge portfolio simple and intuitive, but the portfolio composition is practically invariant under the interpolation method.

the trader will have a portfolio of other, more complicated instruments, and will want to hedge them against yield curve moves by using liquidly traded instruments (which, in general, should exactly be those instruments which were used to bootstrap the original curve). For simplicity, we will assume that these instruments are indeed available for hedging, and the risky instrument to be hedged is nothing more complicated that another vanilla swap: for example, one with term which is not one of the bootstrap terms, is a forward starting swap, or is a stubbed swap.

Suppose initially that, with n instruments being used in our bootstrap, there are exactly n yield curve movements that we wish to hedge against. It is easy to see that we can construct a perfect hedge. First one calculates the square matrix P where P_{ij} is the change in price of the j^{th} bootstrapping instrument under the i^{th} curve. Next we calculate the change in value of our risk instrument under the i^{th} curve to form a column vector ΔV. The quantity of the i^{th} bootstrapping instrument required for the perfect replication is the quantity Q_i where Q is the solution to the matrix equation $PQ = \Delta V$. Assuming for the moment that P is invertible, we find the solution.

Of course, in reality, the set of possible yield curve movements is far, far greater. What one wants to do then is find a set of n yield curve changes whose moves are somehow representative. Some methods have been suggested as follows:

- Perturbing the curve: creation of bumps. In bumping, we form new curves indexed by i: to create the i^{th} curve one bumps up the i^{th} input rate by say 1 basis point, and bootstraps the curve again.
- Perturbing the forward curve with triangles. One approach is to again form new curves, again indexed by i: the i^{th} curve has the original forward curve incremented by a triangle, with left hand endpoint at t_{i-1}, fixed height say one basis point and apex at t_i, and right hand endpoint at t_{i+1}. (The first and last triangle will in fact be right angled, with their apex at the first and last time points respectively.)
- Perturbing the forward curve with boxes. In boxes: the i^{th} curve has the original forward curve incremented by a rectangle, with left hand endpoint at t_{i-1} and right hand endpoint at t_i, and fixed height say one basis point. Such a perturbation curve corresponds exactly with what we get from bumping, if one of the inputs is a $t_{i-1} \times t_i$ FRA rate, we bump this rate, and we use the raw interpolation method.

More generally the user might want to define generic key terms e.g. 1w, 1m, 3m, 6m, 1y, 2y, etc. and define triangles or boxes relative to these

Table 1: A synopsis of the comparison between methods.

Yield curve type	Forwards positive?	Forward smoothness	Method local?	Forwards stable?	Bump hedges local?
Linear on discount	no	not continuous	excellent	excellent	very good
Linear on rates	no	not continuous	excellent	excellent	very good
Raw (linear on log of discount)	yes	not continuous	excellent	excellent	very good
Linear on the log of rates	no	not continuous	excellent	excellent	very good
Piecewise linear forward	no	continuous	poor	very poor	very poor
Quadratic	no	continuous	poor	very poor	very poor
Natural cubic	no	smooth	poor	good	poor
Hermite/Bessel	no	smooth	very good	good	poor
Financial	no	smooth	poor	good	poor
Quadratic natural	no	smooth	poor	good	poor
Hermite/Bessel on rt function	no	smooth	very good	good	poor
Monotone piecewise cubic	no	continuous	very good	good	good
Quartic	no	smooth	poor	very poor	very poor
Monotone convex (unameliorated)	yes	continuous	very good	good	good
Monotone convex (ameliorated)	yes	continuous	good	good	good
Minimal	no	continuous	poor	good	very poor

terms - the inputs to the bootstrap do not necessarily correspond to these nodes.

In either case we have (an automated or user defined) set of dates $t_1, t_2, \ldots, t_n$ which will be the basis for our waves, where the triangles are defined as above.

Some obvious ideas which are just as obviously rejected are to form corresponding perturbations to the yield curve itself - such curves will not be arbitrage free (the derived Z function will not be decreasing).

As an example, consider a 51m swap, where (for simplicity, and indeed, in some markets, such as the second author's domestic market) both fixed and floating payments in swaps occur every 3 months. Thus our swap lies between the 4 and 5 year swap, which let us assume are inputs to the curve.

The type of results we get are in Figure 6. The very plausible and popular bump method performs adequately for many methods, but some methods - for example, the minimal method - can be rejected out of hand if bumping is to be used. Furthermore, for all of the cubic splining methods, there is hedge leakage of varying degrees. Perturbing with triangles can be rejected out of hand as a method - indeed, the pathology that occurs here is akin to the pathology that arises when one uses the piecewise forward linear method of bootstrap: adding or removing an input to the bootstrap will reverse the sign of the hedge quantities before the input in question. Anyway, to have these magnitudes in the hedge portfolio is simply absurd. Perturbing with boxes is the method of choice, but unfortunately does not enable us to distinguish between the quality of the different interpolation methods.

8 Conclusion

The comparison of the methods we analyse in Hagan and West [2006] appears in Table 1.

It is our opinion that the new method derived in Hagan and West [2006], namely monotone convex (in particular, the unameliorated version) should be the method of choice for interpolation. To the best of our knowledge this is the only published method where simultaneously

(1) all input instruments to the bootstrap are exactly reproduced as outputs of the bootstrap,
(2) the instantaneous forward curve is guaranteed to be positive if the inputs allow it (in particular, the curve is arbitrage free), and
(3) the instantaneous forward curve is typically continuous.

In addition, as bonuses

(4) the method is local i.e. changes in inputs at a certain location do not affect in any way the value of the curve at other locations.
(5) the forwards are stable i.e. as inputs change, the instantaneous forwards change more or less proportionately.
(6) hedges constructed by perturbations of this curve are reasonable and stable.

In Hagan and West [2006] we have reviewed many interpolation methods available and have introduced a couple of new methods. In the final analysis, the choice of which method to use will always be subjective, and needs to be decided on a case by case basis. But we hope to have provided some warning flags about many of the methods, and have outlined several qualitative and quantitative criteria for making the selection on which method to use.

FOOTNOTES & REFERENCES

1. We have $r(s)s + C = \int f(s)ds$, so $r(t)t = [r(s)s]_0^t = \int_0^t f(s)ds$.
2. It would be wrong to say that there is no new information; that would be the case under linear interpolation of rates, but not necessarily here.
3. Strictly speaking, we are defining functions g_i, each corresponding to the interval $[\tau_{i-1}, \tau_i]$. As the g_i are constructed one at a time, we suppress the subscript.

[1] Ken Adams. Smooth interpolation of zero curves. *Algo Research Quarterly,* 4(1/2):11—22, 2001.
[2] Carl de Boor. *A Practical Guide to Splines: Revised Edition,* volume 27 of *Applied Mathematical Sciences.* Springer-Verlag New York Inc., 1978, 2001.
[3] Patrick S. Hagan and Graeme West. Interpolation methods for curve construction. *Applied Mathematical Finance,* 13 (2):89—129, 2006.
[4] James M. Hyman. Accurate monotonicity preserving cubic interpolation. *SIAM Journal on Scientific and Statistical Computing,* 4(4):645—654, 1983.
[5] J. Huston McCulloch and Levis A. Kochin. The inflation premium implicit in the US real and nominal term structures of interest rates. Technical Report 12, Ohio State University Economics Department, 2000. URL http://www.econ.ohio-state.edu/jhm/jhm.html.
[6] Ricardo Rebonato. *Interest-Rate Option Models.* John Wiley and Sons Ltd, second edition, 1998.

Pseudo Code For Monotone Convex Interpolation

First the estimates for $f_0, f_1, \ldots, f_n$. This implementation assumes that it is required that the output curve is everywhere positive. Various arrays have already been dimensioned, the raw data inputs have already been provided, and it has been specified with the boolean variable 'InputsareForwards' where those inputs are rates $r_1, r_2, \ldots, r_n$ or discrete forwards $f_1^d, f_2^d, \ldots, f_n^d$. Further, a 'collar' and 'min' utility functions are used (not shown). Of course, $\text{collar}(a, b, c) = \max(a, \min(b, c))$.

```
Private Sub fi_estimates()
  'extend the curve to time 0
  Terms(0) = 0
  Values(0) = Values(1)
  'step 1
  If InputsareForwards = False Then
    For j = 1 To n
      fdiscrete(j) = (Terms(j) * Values(j) - Terms(j - 1) _
                   * Values(j - 1)) / (Terms(j) _
                   - Terms(j - 1))
    Next j
  Else
    For j = 1 To n
      fdiscrete(j) = Values(j)
    Next j
  End If
  'step 2
  For j = 1 To n - 1
    f(j) = (Terms(j) - Terms(j - 1)) / (Terms(j + 1) _
         - Terms(j - 1)) * fdiscrete(j + 1) + (Terms(j + 1) _
         - Terms(j)) / (Terms(j + 1) - Terms(j - 1)) _
         * fdiscrete(j)
  Next j
  'step 3
  f(0) = Util.collar(0, fdiscrete(1) - 0.5
       * (f(1) - fdiscrete(1)), 2 * fdiscrete(1))
  f(n) = Util.collar(0, fdiscrete(n) - 0.5  _
       * (f(n - 1) - fdiscrete(n)), 2 * fdiscrete(n))
  For j = 1 To n - 1
    f(j) = Util.collar(0, f(j), 2 _
         * Util.Min(fdiscrete(j), fdiscrete(j + 1)))
  Next j
  fi_estimates_are_Calced = True
End Sub
```

Having found the estimates for $f_0, f_1, \ldots, f_n$, we can find the value of $f(\tau)$ for any τ. The key function here is 'LastIndex', which determines the unique value of i for which $\tau \in [\tau_i, \tau_{i+1})$. Extrapolation is as in the third paragraph of §6.

```
Public Function Forward(Term As Double) As Double
  If fi_estimates_are_Calced = False Then fi_estimates
  If Term <= 0 Then
    Forward = f(0)
  ElseIf Term >= Terms(n) Then
    Forward = Interpolant(Terms(n))
  Else
    i = Util.LastIndex(dTerms, Term)
    'the x in (25)
    x = (Term - Terms(i)) / (Terms(i + 1) - Terms(i))
    g0 = f(i) - fdiscrete(i + 1)
    g1 = f(i + 1) - fdiscrete(i + 1)
    If x = 0 Then
      G = g0
    ElseIf x = 1 Then
      G = g1
    ElseIf (g0 < 0 And -0.5 * g0 <= g1 And g1 <= -2 * g0) _
      Or (g0 > 0 And -0.5 * g0 >= g1 And g1 >= -2 * g0) Then
      'zone (i)
      G = g0 * (1 - 4 * x + 3 * x ^ 2) + g1 _
        * (-2 * x + 3 * x ^ 2)
    ElseIf (g0 < 0 And g1 > -2 * g0) Or (g0 > 0 _
      And g1 < -2 * g0) Then
      'zone (ii)
      '(29)
      eta = (g1 + 2 * g0) / (g1 - g0)
      '(28)
```

```
      If x <= eta Then
        G = g0
      Else
        G = g0 + (g1 - g0) * ((x - eta) / (1 - eta)) ^ 2
      End If
    ElseIf (g0 > 0 And 0 > g1 And g1 > -0.5 * g0) _
      Or (g0 < 0 And 0 < g1 And g1 < -0.5 * g0) Then
      'zone (iii)
      '(31)
      eta = 3 * g1 / (g1 - g0)
      '(30)
      If x < eta Then
        G = g1 + (g0 - g1) * ((eta - x) / eta) ^ 2
      Else
        G = g1
      End If
    ElseIf g0 = 0 And g1 = 0 Then
      G = 0
    Else
      'zone (iv)
      '(33)
      eta = g1 / (g1 + g0)
      '(34)
      A = -g0 * g1 / (g0 + g1)
      '(32)
      If x <= eta Then
        G = A + (g0 - A) * ((eta - x) / eta) ^ 2
      Else
        G = A + (g1 - A) * ((eta - x) / (1 - eta)) ^ 2
      End If
    End If
    '(26)
    Forward = G + fdiscrete(i + 1)
  End If
End Function



Public Function Interpolant(Term As Double) As Double
  If fi_estimates_are_Calced = False Then fi_estimates
  If Term <= 0 Then
    Interpolant = f(0)
  ElseIf Term > Terms(n) Then
    Interpolant = Interpolant(Terms(n))
  Else
    i = Util.LastIndex(dTerms, Term)
    L = Terms(i + 1) - Terms(i)
    'the x in (25)
    x = (Term - Terms(i)) / L
    g0 = f(i) - fdiscrete(i + 1)
    g1 = f(i + 1) - fdiscrete(i + 1)
    If x = 0 Or x = 1 Then
      G = 0
    ElseIf (g0 < 0 And -0.5 * g0 <= g1 _
      And g1 <= -2 * g0) \ Or (g0 > 0 And -0.5 _
      * g0 >= g1 And g1 >= -2 * g0) Then
      'zone (i)
      G = L * (g0 * (x - 2 * x ^ 2 + x ^ 3) + g1 _
        * (-x ^ 2 + x ^ 3))
    ElseIf (g0 < 0 And g1 > -2 * g0) Or (g0 > 0 _
      And g1 < -2 * g0) Then
      'zone (ii)
      '(29)
      eta = (g1 + 2 * g0) / (g1 - g0)
      '(28)
      If x <= eta Then
        G = g0 * (Term - Terms(i))
      Else
        G = g0 * (Term - Terms(i)) + (g1 - g0) _
          * (x - eta) ^ 3 / (1 - eta) ^ 2 / 3 * L
      End If
    ElseIf (g0 > 0 And 0 > g1 And g1 > -0.5 * g0) _
      Or (g0 < 0 And 0 < g1 And g1 < -0.5 * g0) Then
      'zone (iii)
      '(31)
      eta = 3 * g1 / (g1 - g0)
      '(30)
      If x < eta Then
        G = L * (g1 * x - 1 / 3 * (g0 - g1) * ((eta - x) ^ 3 _
          / eta ^ 2 - eta))
      Else
        G = L * (2 / 3 * g1 + 1 / 3 * g0) * eta + g1 _
          * (x - eta) * L
      End If
    ElseIf g0 = 0 And g1 = 0 Then
      G = 0
    Else
      'zone (iv)
      '(33)
      eta = g1 / (g1 + g0)
      '(34)
      A = -g0 * g1 / (g0 + g1)
      '(32)
      If x <= eta Then
        G = L * (A * x - 1 / 3 * (g0 - A) * ((eta - x) ^ 3 _
          / eta ^ 2 - eta))
      Else
        G = L * (2 / 3 * A + 1 / 3 * g0) * eta + L _
          * (A * (x - eta) + (g1 - A) / 3 * (x - eta) ^ 3 _
          / (1 - eta) ^ 2)
      End If
    End If
    '(12)
    Interpolant = 1 / Term * (Terms(i) * Values(i) _
                + fdiscrete(i + 1) * (Term - Terms(i)) + G)
  End If
End Function
```

Working code for this interpolation scheme, with proper dimensioning of all arrays and code for all the missing functions, is available from the second author's website on the resources page.