Tension Spline Algorithm for Building Forward Curves

Jake C. Fowler

February 2023

1 Introduction

2 Deriving the Algorithm

2.1 Functional Form

The base of the algorithm is a spline, which by definition is made up of piecewise polynomial functions.

$$p(t) = \begin{cases} p_1(t) & \text{for} \quad t \in [t_0, t_1) \\ p_2(t) & \text{for} \quad t \in [t_1, t_2) \\ \vdots & \\ p_{n-1}(t) & \text{for} \quad t \in [t_{n-2}, t_{n-1}) \\ p_n(t) & \text{for} \quad t \in [t_{n-1}, t_n] \end{cases} \tag{1}$$

Where $t_0 < t_1 < \ldots < t_{n-1} < t_n$ are the boundary points between the polynomials which make up the spline. In the context of building a forward curve, the variable t is defined as the time until start of delivery of a forward contract.

The boundary points are chosen to be start of the input forward prices. It is also assumed that the input forward prices are not for delivery periods which overlap with any other input. Gaps between input forward contracts are permitted, in which case a boundary point will exit for the start of the gap.

$$\begin{aligned} p_i(t) = \big[z_{i-1}\sinh(\tau(t_i - t)) + z_i \sinh(\tau(t - t_{i-1}))\big]/(\tau^2 \sinh(\tau h_i)) \\ + (y_{i-1} - z_{i-1}/\tau^2)(t_i - t)/h_i + (y_i - z_i/\tau^2)(t - t_{i-1})/h_i \end{aligned} \tag{2}$$

Where $h_i = t_i - t_{i-1}$. $z_i = p^{''}(t_i)$ and $y_i = p(t_i)$, i.e. the (as yet unknown) value of the function at the boundary points.

The curve fitting algorithm essentially involves solving for the parameters z_i, and y_i for $i = 0 \dots n$.

In many cases the spline described above is not sufficient to derive a forward curve which shows strong price seasonality, especially when this seasonality cannot be directly observed in the traded forward prices. An example of this is the day-of-week seasonality for gas and power prices, which generally are lower at the weekend when demand is lower. As such the function form is as follows:

$$f(t) = (p(t) + S_{add}(t))S_{mult}(t) \tag{3}$$

Where the forward price for the period starting delivery at time t is given by $f(t)$, which consists of $p(t)$ adjusted by two arbitrary seasonal adjustment functions $S_{add}(t)$ an additive adjustment, and $S_{mult}(t)$ a multiplicative adjustment.

2.2 Constraints

2.2.1 Polynomial Boundary Point Constraints

As usual with splines, constraints are put in place that adjascent polynomials have equal value, first derivative, and second derivatives at the boundary points. These three constraints can be respectively expressed as:

2.2.2 Forward Price Constraint

The most important constraints is that the derived forward curve averages back to the input traded forward prices. The market inputs to the forward curve model are traded forward prices F_i. Setting this equal to the average of the derived smooth curve:

$$F_j = \frac{\sum_{t \in T_j} (p(t) + S_{add}(t))S_{mult}(t)w(t)D(t)}{\sum_{t \in T_j} w(t)D(t)} \tag{4}$$

Where $D(t)$ is the discount factor from the settlement date of delivery period t. $w(t)$ is a weighting function and T_i is the set of all delivery start times for the delivery periods at the granularity of the curve being built. The weighting function has two meanings from a busines perspective.

- The volume of commodity delivered in each period. For example, an off-peak power forward contract in the UK delivers over 12 hours in on weekdays, and 24 hours on weekends, hence $w(t)$ would equal double for t representing weekends compared to $w(t)$ when t represents a weekday delivery. Clock changes can also cause the total volume delivered over a day in a fixed time zone to vary due to hours lost or gained. Hence $w(t)$ can be used to account for this.

- For swaps which only fix on certain days (usually business days) $w(t)$ can be used to account for this by returning the number of fixing days in the period starting at t. For example if deriving a a monthly curve $w(t)$ would evaluate to the number of fixing days in the month starting at t.

Equation 4 can be transformed into an equation linear on the parameters of the piecewise polynomial by substituting in the polynomial representation of $p(t)$:

$$\sum_i \sum_{t \in T_j \cap [t_{i-1}, t_i)} p_i(t) S_{mult}(t) w(t) D(t) = F_j \sum_{t \in T_j} w(t) D(t) - \sum_{t \in T_i} S_{add}(t) S_{mult}(t) w(t) D(t) \tag{5}$$

Substituting in for $p_i(t)$:

$$\begin{aligned} \sum_i \sum_{t \in T_i \cap [t_{i-1}, t_i)} & \left(\left[z_{i-1} \sinh(\tau(t_i - t)) + z_i \sinh(\tau(t - t_{i-1})) \right] / (\tau^2 \sinh(\tau h_i)) \right. \\ & + (y_{i-1} - z_{i-1}/\tau^2)(t_i - t)/h_i + (y_i - z_i/\tau^2)(t - t_{i-1})/h_i + S_{add}(t) \Big) S_{mult}(t) w(t) D(t) \\ & = F_i \sum_{t \in T_i} w(t) D(t) - \sum_{t \in T_i} S_{add}(t) S_{mult}(t) w(t) D(t) \end{aligned} \tag{6}$$

Rearranging again gives a form linear with respect to the unknown polynomial coefficients z_i, z_{-1}, y_i and y_{i-1}.

$$\begin{aligned} \sum_i \Bigg(& z_i \sum_{t \in T_i \cap [t_{i-1}, t_i)} \left(\frac{\sinh(\tau(t - t_{i-1}))}{\tau^2 \sinh(\tau h_i)} - \frac{t - t_{i-1}}{\tau^2 h_i} \right) S_{mult}(t) w(t) D(t) \\ & + z_{i-1} \sum_{t \in T_i \cap [t_{i-1}, t_i)} \left(\frac{\sinh(\tau(t_i - t))}{\tau^2 \sinh(\tau h_i)} - \frac{t_i - t}{\tau^2 h_i} \right) S_{mult}(t) w(t) D(t) \\ & + y_i \sum_{t \in T_i \cap [t_{i-1}, t_i)} \frac{(t - t_{i-1})}{h_i} S_{mult}(t) w(t) D(t) \\ & + y_{i-1} \sum_{t \in T_i \cap [t_{i-1}, t_i)} \frac{(t_i - t)}{h_i} S_{mult}(t) w(t) D(t) \Bigg) \\ & = F_i \sum_{t \in T_i} w(t) D(t) - \sum_{t \in T_i} S_{add}(t) S_{mult}(t) w(t) D(t) \end{aligned} \tag{7}$$

2.2.3 Matrix Form of Constraints

Equations **??**, **??**, **??** and **??** can be expressed as the linear system $\mathbf{Ax} = \mathbf{b}$ where:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{n-1} \\ \mathbf{x}_n \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_{n-1} \\ \mathbf{b}_n \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & & \ddots & & \vdots \\ \mathbf{0} & \mathbf{0} & & \mathbf{A}_{n-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{A}_n \end{bmatrix}$$

And:

$$\mathbf{x_i} = \begin{bmatrix} a_i \\ b_i \\ c_i \\ d_i \\ e_i \end{bmatrix}$$

$$\mathbf{b_i} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ F_i \sum_{t \in T_i} w(t) - \sum_{t \in T_i} S_{add}(t) S_{mult}(t) w(t) \end{bmatrix}$$

$$\mathbf{A}_i = \begin{bmatrix} 1 & t_i & t_i^2 & t_i^3 & t_i^4 & -1 & -t_i & -t_i^2 & -t_i^3 & -t_i^4 \\ 0 & 1 & 2t_i & 3t_i^2 & 4t_i^3 & 0 & -1 & -2t_i & -3t_i^2 & -4t_i^3 \\ 0 & 0 & 2 & 6t & 12t^2 & 0 & 0 & -2 & -6t & -12t^2 \\ f_i^1 & f_i^2 & f_i^3 & f_i^4 & f_i^5 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Where the components in the last row are defined as:

$$f_i^1 = \sum_{t \in T_i} S_{mult}(t) w(t)$$
$$f_i^2 = \sum_{t \in T_i} S_{mult}(t) w(t) t$$
$$f_i^3 = \sum_{t \in T_i} S_{mult}(t) w(t) t^2$$
$$f_i^4 = \sum_{t \in T_i} S_{mult}(t) w(t) t^3$$
$$f_i^5 = \sum_{t \in T_i} S_{mult}(t) w(t) t^4$$

2.3 Smoothness Criteria

Maximum smoothness is obtained by by finding the spline parameters which minimise the integral of the second derivative.

$$min \int_{t_0}^{t_n} p''(t)^2 dt = \sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} p_i''(t)^2 dt$$
$$= \sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} (2c_i + 6d_i t + 12e_i t^2)^2 dt$$
$$= \sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} (4c_i^2 + 24c_i d_i t + 48c_i e_i t^2 + 36d_i^2 t^2 + 144 d_i e_i t^3 + 144 e_i^2 t^4) dt$$
$$= \sum_{i=1}^{n} 4c_i^2 \Delta_i^1 + 12 c_i d_i \Delta_i^2 + 16 c_i e_i \Delta_i^3 + 12 d_i^2 \Delta_i^3 + 36 d_i e_i \Delta_i^4 + \frac{144}{5} e_i^2 \Delta_i^5 \quad (8)$$

Where Δ_i^j is defined as the difference between t^j at the polynomial boundary points, i.e. $\Delta_i^j = t_i^j - t_{i-1}^j$. Recognising 8 as a quadratic form it can be reformulating in the following matrix form:

$$\sum_{i=1}^{n} \mathbf{x_i^T H_i x_i} \quad (9)$$

Where:

$$\mathbf{H_i} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4\Delta_i^1 & 6\Delta_i^2 & 8\Delta_i^3 \\ 0 & 0 & 6\Delta_i^2 & 12\Delta_i^3 & 18\Delta_i^4 \\ 0 & 0 & 8\Delta_i^3 & 18\Delta_i^4 & \frac{144}{5}\Delta_i^5 \end{bmatrix}$$

The objective function 9 can be arranged into a single matrix quadratic form without the summation as:

$$\mathbf{x^T H x} \tag{10}$$

Where:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{n-1} \\ \mathbf{x}_n \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & & \ddots & & \vdots \\ \mathbf{0} & \mathbf{0} & & \mathbf{H}_{n-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{H}_n \end{bmatrix}$$

2.4 Minimisation Problem

The sections above show that finding the maximum smoothness curve comes down to finding the polynomial coefficients, vector $\mathbf{x}$, which minimises $\mathbf{x^T H x}$, subject to the linear constraints $\mathbf{Ax} = \mathbf{b}$. This problem is well suited to the method of Lagrange multipliers for which we first define the vector λ.

$$\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{4n-2} \\ \lambda_{4n-1} \end{bmatrix} \tag{11}$$

$$\mathcal{L}(\mathbf{x}, \lambda) = \mathbf{x^T H x} + \lambda^{\mathbf{T}}(\mathbf{Ax} - \mathbf{b}) \tag{12}$$

The minima $\min_{x,\lambda} \mathcal{L}(\mathbf{x}, \lambda)$ is found as the solution where the partial derivatives of $\mathcal{L}(\mathbf{x}, \lambda)$ with respect to $\mathbf{x}$ and λ are zero.

$$\frac{\mathcal{L}(\mathbf{x}, \lambda)}{\partial \mathbf{x}} = \mathbf{2Hx} + \mathbf{A^T}\lambda = 0 \tag{13}$$

$$\frac{\mathcal{L}(\mathbf{x}, \lambda)}{\partial \lambda} = \mathbf{Ax} - \mathbf{b} = 0 \tag{14}$$

These can be arranged into a single linear system:

$$\begin{bmatrix} \mathbf{2H} & \mathbf{A^T} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x_{min}} \\ \lambda_{\mathbf{min}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix} \tag{15}$$

Hence the vector of spline polynomial coefficients for the maximum smoothness curve, $\mathbf{x_{min}}$, can be found by solving this system.

$$\begin{bmatrix} \mathbf{x_{min}} \\ \lambda_{\mathbf{min}} \end{bmatrix} = \begin{bmatrix} \mathbf{2H} & \mathbf{A^T} \\ \mathbf{A} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix} \tag{16}$$

Once 16 is solved, the spline parameters are taken from $\mathbf{x_{min}}$ and the derived forward prices are calculated by evaluating 3.