Bank of Canada Banque du Canada

Working Paper 2002-29 / Document de travail 2002-29

Exponentials, Polynomials, and Fourier Series: More Yield Curve Modelling at the Bank of Canada

by

David Jamieson Bolder and Scott Gusba

ISSN 1192-5434

Printed in Canada on recycled paper

Contents

Acknowledgements

We would like to particularly thank Grahame Johnson, Marc Larson, and Peter Youngman from the Bank of Canada for creating the impetus for this project, patiently explaining some of the necessary background on fixed-income markets, and providing a *sanity check* to our analysis. We would also like to thank, without implication, Michel Krieber from TD Securities, Phelim Boyle from the University of Waterloo and the Centre for Advanced Studies in Finance, and Mark Reesor from the Applied Mathematics Department of the University of Western Ontario. As always, we entirely retain any and all responsibility for errors, omissions, and inconsistencies that may appear in this work.

Abstract

This paper continues the work started by Bolder and Stréliski (1999) and considers two alternative classes of models for extracting zero-coupon and forward rates from a set of observed Government of Canada bond and treasury-bill prices. The first class of term-structure estimation methods follows from work by Fisher, Nychka, and Zervos (1994), Anderson and Sleath (2001), and Waggoner (1997). This approach employs a B-spline basis for the space of cubic splines to fit observed coupon-bond prices—as a consequence, we call these the *spline-based* models. This approach includes a penalty in the generalized least-squares objective function—following from Waggoner (1997)—that imposes the desired level of smoothness into the term structure of interest rates. The second class of methods is called *function-based* and includes variations on the work of Li et al. (2001), which uses linear combinations of basis functions, defined over the entire term-to-maturity spectrum, to fit the discount function. This class of function-based models includes the model proposed by Svensson (1994). In addition to a comprehensive discussion of these models, the authors perform an extensive comparison of these models' performance in the Canadian marketplace.

JEL classification: C0, C6, E4, G1
Bank classification: Interest rates; Econometric and statistical methods; Financial markets

Résumé

Le présent document fait suite à l'étude de Bolder et Stréliski (1999) et examine deux classes de modèles différents dans le but de déterminer le taux des obligations à coupon zéro et les taux d'intérêt à terme à partir des cours observés des obligations et des bons du Trésor du gouvernement canadien. La première classe de modèles d'estimation, que nous appelons des modèles axés sur des splines, s'inscrit dans le prolongement des travaux de Fisher, Nychka et Zervos (1994), Anderson et Sleath (2001) et Waggoner (1997) et utilise une fonction spline cubique pour estimer les cours observés des obligations à coupon zéro. Dans cette approche, une pénalité ajoutée à la fonction objective des moindres carrés généralisés (proposée par Waggoner) permet d'intégrer le niveau désiré de lissage dans la structure à terme des taux d'intérêt. La seconde classe de modèles, les modèles fondés sur une fonction, est constituée de variantes du modèle de Li et coll. (2001). Elle utilise des combinaisons linéaires définies sur l'éventail entier des échéances pour estimer la fonction d'actualisation. Le modèle proposé par Svensson (1994) appartient à cette classe. La présente étude comprend, outre un examen approfondi de ces divers modèles, une comparaison détaillée de leur performance dans le contexte des marchés canadiens.

Classification JEL : C0, C6, E4, G1
Classification de la Banque : Taux d'intérêt; Méthodes économétriques et statistiques; Marchés financiers

1 Introduction

In the world of fixed-income, it is difficult to find a more fundamental object than a riskless *pure discount bond* or, as it is equivalently called, a zero-coupon bond. This is because the price of a pure discount bond represents the current value of one dollar paid with complete certainty—hence the word *riskless*—at some future point in time. Abstracting from the idea of risk premia for longer-term bond holdings, it is essentially a representation of the time value of money. A trivial transformation of the bond price is the rate of return on this simple instrument or, as it is more commonly termed, the *zero-coupon* interest rate. These building blocks of fixed-income finance are tremendously important for a wide array of different purposes, including bond pricing, discounting future cash flows, pricing fixed-income derivative products, constructing forward interest rates, and determining risk premia associated with holding bonds of different maturities. It often comes as a surprise, therefore, to those new to fixed-income markets that these objects are not directly observable. In most sovereign bond markets, pure discount bond prices are available only out to a one-year term to maturity in the form of treasury bills.

The lack of available pure discount bonds that can be used to compute zero-coupon interest rates is problematic. To solve this problem, we must employ various models and numerical techniques to extract zero-coupon interest rates from the prices of those risk-free debt instruments that are available: government coupon bonds. We will call this the *term-structure estimation* problem. How is this possible? It is possible because a coupon bond is, in fact, a portfolio of pure-discount bonds. Consequently, the price of a coupon bond is merely the sum of these pure discount bond prices. In short, if a model provides zero-coupon rates that are a good approximation to the set of coupon bonds in the economy, then it is probably a good model. Indeed, every model for extracting zero-coupon rates exploits this fundamental relationship—albeit in different ways. A large part of this paper is devoted to explaining, in substantial detail, exactly how a number of different models accomplish this task.

Even for those who are well aware of the unobservability of zero-coupon rates, there is a bewildering array of competing approaches for extracting zero-coupon rates from coupon bond prices. One reason for the proliferation of models is that any approach used to extract zero-coupon rates from government coupon bonds has little or no theoretical foundation. Indeed, all of these models are based on *curve-fitting* techniques.[1] A second complicating factor is the natural *tension* between the closeness of fit to the set of observed government coupon prices and the smoothness of the corresponding zero-coupon rate function. Zero-coupon curve smoothness is a relevant criterion for a term-structure estimation model, because overly non-smooth zero-coupon curves are highly oscillatory functions in a model. This implies the occurrence of dramatic swings in rates from one period to the next. Typically, one expects the term structure of interest rates to move gradually across the term-to-maturity spectrum. Dramatic moves, conversely, are *not*

[1] Curve-fitting is defined as fitting a continuous function to a set of discretely observed datapoints.

considered reasonable. What possible economic reason, for example, could explain a large difference between the price of a five-year pure discount bond and a five-year-and-one-week pure discount bond? An overly close fit to the data will tend to produce these types of ill-behaved zero-coupon and forward term structures. Specification of a smooth zero-coupon function, however, is not the solution. An overly smooth zero-coupon curve will not generally be capable of accurately pricing the set of coupon bonds in the economy. This tension is often described as the trade-off between *goodness of fit* and *smoothness*. Ideally, a model must strike a balance between these two competing criteria.

The natural question, of course, is which is the best model to use for this purpose? The answer, unfortunately, is that it *depends* on the application. If one is attempting to accurately price a set of off-the-run government bonds or the price of a derivative security, then smoothness is not the dominant criterion for the selection of a model. As will become evident in our analysis, however, a modicum of smoothness is necessary even for this purpose, as models that overfit the coupon bond prices typically perform poorly out of sample. Conversely, if one is attempting to use the term structure of zero-coupon rates to extract the aggregate interest-rate expectations of economic agents at a given point in time, then a relatively smooth curve is desirable. Again, any overly smooth specification of the zero-coupon curve may mask important economic information embedded in government coupon prices. A balance between goodness of fit and smoothness must be struck that leans towards the desired application. The final result is that, although there are a wide variety of models, it is reasonable for an institution to use more than one model, depending on the composition of its tasks. In a central bank, for example, the zero-coupon and forward term structures of interest rates are used for a wide variety of purposes. Hence, a central bank requires a wide variety of models.

This paper seeks, therefore, to extend the work of Bolder and Stréliski (1999) and examine a number of more recent models used in this area. Our objective is to enhance our understanding of term-structure estimation models at the Bank of Canada. To accomplish this, we treat eight separate models that fall into two main classes. First, we consider four separate piecewise-cubic polynomial-based approaches, which we call *spline-based* models, that are based on work by McCulloch (1971), Fisher, Nychka, and Zervos (1994), Waggoner (1997), and Anderson and Sleath (2001). Second, we examine four different *function-based* models. These models take linear combinations of various functions—exponential and trigonometric functions, to be precise—to model the zero-coupon term structure. These methodologies are based on the work of Vasicek and Fong (1981), Li et al. (2001), and Svensson (1994). This paper leans quite heavily on the contributions of these authors. Indeed, there is relatively little new in this paper aside from a few slight twists in the modelling, a comprehensive self-contained presentation, and the application of these models to the Government of Canada fixed-income market.

The paper is organized into three main sections. In section 2, we provide the necessary mathematical background for the spline-based models. The idea is to make this class of models more accessible to the end consumer. Armed with this background, we proceed in section 3 to work through the derivation of

the various spline-based and function-based methodologies considered in the paper. Both sections 2 and 3 make ample reference to the appendix, which provides illustrative MATLAB computer routines for the computation of various key mathematical objects. Using these model constructions, the paper proceeds to perform a more formal comparison of these models in section 4. Using almost 600 daily data points, we estimate each of our eight models and compare their performance on the basis of how well they fit the data, the nature of these pricing errors, and their computational speed. We then consider a subset of these models and perform an experiment to assess the overall stability of the models. In other words, in section 4, we perform a *horse race* among the models with a view towards recommending two models for general use at the Bank of Canada.

2 Mathematical Preliminaries

In this paper, we will make extensive use of *spline* models to fit a zero-coupon curve to a set of observed bond prices. A spline is a collection of piecewise polynomials of a given degree that, subject to certain conditions, are fit to a data set. While this is a popular technique, and indeed there is a surfeit of available software to accomplish this task, there is relatively little in the finance literature that works through the details of spline models. Unfortunately, although spline models are fairly simple, they can be somewhat intimidating from a notational perspective.[2] Moreover, to achieve reasonable numerical results, one must often pose the problem in a less-than-direct fashion. We believe, however, that one can gain substantial insight into the problem and its attendant numerical difficulties by considering the much simpler problem of polynomial interpolation. We will consider this problem in detail and then examine how it can be generalized into a spline model.

Imagine that we are given a set of data that consists of $N+1$ distinct x-coordinates,

$$\{x_0, x_1, ..., x_N\}, \tag{1}$$

and $N+1$ corresponding values of the unknown function, f, as follows,

$$\{f_0, f_1, ..., f_N\}. \tag{2}$$

Typically, we consider the domain of this function as $[a, b]$, where $a = x_0, b = x_N$, and $x_0 < x_1 ... < x_N$. We also have reason to believe that, in fact, this unknown function, f, is $C[a, b]$ (i.e., f is continuous on $[a, b]$). One possible method to find a continuous function for the observed set of values in equation (2) is to fit a polynomial through these points. There is actually a uniqueness theorem to help us out in this respect. In

[2] Fortunately, a number of excellent mathematical and engineering resources address this problem directly. This discussion is a distillation of the results so aptly presented in Lancaster and Salkauskas (1986), Dierckx (1993), deBoor (1978), Ralston and Rabinowitz (1978), Press et al. (1992), and Anderson et al. (1996).

particular, if we define $\mathcal{P}_N$ as the set of all polynomials of degree at most N, then we can state that for distinct values in equation (1) and the values in equation (2), there exists a unique $p \in \mathcal{P}_N$ such that,

$$p(x_i) = f_i, \tag{3}$$

for $i = 0, 1, ..., N$. This implies that with a polynomial of degree N we can uniquely fit $N + 1$ points. Moreover, this requires only that the points in the domain of $\{f_0, f_1, ..., f_N\}$ be distinct.

This is a very useful result. All we require, therefore, is an algorithm to help us determine the coefficients of this polynomial $p \in \mathcal{P}_N$. An obvious way to approach this problem is to write out the equations for these Nth degree polynomials and attempt to solve them directly. Ultimately, this is not the right way to proceed, but it is nonetheless educational. Consider, therefore, the following system of equations,

$$\begin{aligned} a_0 + a_1 x_0 + a_2 x_0^2 + \cdots + a_N x_0^N &= f_0, \\ a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_N x_1^N &= f_1, \\ &\vdots \\ a_0 + a_1 x_N + a_2 x_N^2 + \cdots + a_N x_N^N &= f_N. \end{aligned} \tag{4}$$

We can write this more conveniently in matrix notation as,

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^N \\ 1 & x_1 & x_1^2 & \cdots & x_1^N \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^N \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_N \end{bmatrix}, \tag{5}$$

or,

$$Va = f. \tag{6}$$

It seems rather obvious that one need only invert the matrix V (i.e., $a = V^{-1}f$) to find the solution to the linear system described in equation (4). Unfortunately, although the distinctness of the x-coordinates guarantees the non-singularity of V in a theoretical sense, it turns out in practice that V is often very poorly conditioned for N of even moderate size. The matrix V, often termed the *Vandermonde* matrix, is well known for its numerical difficulties. Engineering and mathematics textbooks are, therefore, unanimous in their advice to avoid this direct algebraic approach.

How, then, does one determine these coefficients? The solution dates back to a very clever idea from the French mathematician, Lagrange, who proposed a method whereby the problem is decomposed into $N + 1$ simple subproblems. It turns out that one may combine the solutions to these problems to find a solution to the initial problem. This will be made precise in a moment, but we will first work through the details and then discuss the reasoning behind the technique.

We begin with the same $N+1$ distinct x-coordinates described in equation (1), but instead of the function values in equation (2), we find the solution to the following $N+1$ problems,

$$\underbrace{\{1,0,...,0\}}_{\text{Problem 1}},\underbrace{\{0,1,...,0\}}_{\text{Problem 2}},...,\underbrace{\{0,0,...,1\}}_{\text{Problem } N+1}. \tag{7}$$

Our previously mentioned theorem ensures that each of these subproblems has a unique solution. Even better, as V is well-conditioned in this case, solving each of these problems is trivial. To see exactly how this works, consider the following example,[3]

$$\begin{aligned}\{x_0,x_1\}&=\{0,1\},\\ \{f_0,f_1\}&=\{10,13\}.\end{aligned} \tag{8}$$

We can solve this problem in three steps.

Step 1: Let's solve the first subproblem, which has the underlying form,

$$\begin{aligned}a_0+a_1x_0&=f_0,\\ a_0+a_1x_1&=f_1.\end{aligned} \tag{9}$$

Recall, however, that we are not solving this problem with the values $\{f_0,f_1\}$. Instead, in this first problem, $\{f_0,f_1\}=\{1,0\}$. The linear system is thus,

$$\begin{aligned}\begin{bmatrix}1 & x_0\\ 1 & x_1\end{bmatrix}\begin{bmatrix}a_0\\ a_1\end{bmatrix}&=\begin{bmatrix}f_0\\ f_1\end{bmatrix},\\ \begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}\begin{bmatrix}a_0\\ a_1\end{bmatrix}&=\begin{bmatrix}1\\ 0\end{bmatrix},\\ \begin{bmatrix}a_0\\ a_1\end{bmatrix}&=\begin{bmatrix}1 & 0\\ -1 & 1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}=\begin{bmatrix}1\\ -1\end{bmatrix}.\end{aligned} \tag{10}$$

Define the polynomial that solves this problem as $L_0(x)$. Its solution is thus,

$$L_0(x)=1-x. \tag{11}$$

Step 2: Here we merely repeat the first step with $\{f_0,f_1\}=\{0,1\}$. That is, we solve the second subproblem. Following directly from equation (10), the details are as follows,

$$\begin{bmatrix}a_0\\ a_1\end{bmatrix}=\begin{bmatrix}1 & 0\\ -1 & 1\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix}=\begin{bmatrix}0\\ 1\end{bmatrix}. \tag{12}$$

[3]This example is based on the presentation in Lancaster and Salkauskas (1986).

Using our previously defined notation, the solution is

$$L_1(x) = x. \tag{13}$$

Step 3: It turns out that the solution to the original problem, $p(x) \in \mathcal{P}_2$, is merely,

$$p(x) = \sum_{i=0}^{1} f_i L_i(x). \tag{14}$$

Our original solution is,

$$\begin{aligned} p(x) &= f_0 L_0(x) + f_1 L_1(x), \\ &= (10)(1-x) + 13x, \\ &= 10 + 3x. \end{aligned} \tag{15}$$

One can verify that this is the correct answer by solving the original system directly.

What, therefore, have we done? The success of this method stems from the fact that $\mathcal{P}_N$ is a vector space. Each of the polynomials we derived were members of $\mathcal{P}_N$ (i.e., $L_i(x) \in \mathcal{P}_N$ for $i = 0, 1$) and thus any linear combination of $L_0(x)$ and $L_1(x)$ is also an element of $\mathcal{P}_N$. $L_0(x)$ and $L_1(x)$, which are termed the Lagrange polynomials or *cardinal* functions, are linearly independent but *not* orthogonal. As such, the Lagrange polynomials form a basis for our polynomial space, $\mathcal{P}_N$. The solution to our original problem, therefore, is merely a linear combination of this basis. Lagrange's method provides both a technique for determining these basis functions and the appropriate manner for combining them.[4]

This simple polynomial interpolation approach is *not* used to fit the zero-coupon curve to bond prices. It does, however, provide us with some insight into the actual methodology employed for this purpose. In particular, we will be using polynomial functions to fit this zero-coupon curve. The difference is that instead of fitting a single polynomial of degree N, we will be fitting a collection of lower-order polynomials in a piecewise fashion to our $N + 1$ datapoints. This will, of course, complicate the analysis somewhat. Another similarity to Lagrange's method is that, owing to numerical difficulties, we will also employ the use of basis functions to find the coefficients for these piecewise polynomials. The logic behind the construction of this basis is identical to the previously outlined Lagrange polynomial example. In the subsequent discussion, we will take one step towards generalizing this basic result for the cubic spline models that we will be using in our applications.

[4]There are other possible bases for the space, $\mathcal{P}_N$. One can, for example, use so-called Hermite polynomials to accomplish the same task.

2.1 Linear splines

To ease our introduction to splines, and see how the previously described concepts generalize to our setting, we will begin with the easiest possible case, the linear spline model. Ultimately, the idea here is to fit a piecewise linear function through the set of observations in equation (2). In a spline model, one has to decide the endpoints of the individual piecewise functions. These are termed *knot* points and we will denote them as,

$$K = \{k_0, k_1, ..., k_m : k_0 < k_1 < \cdots < k_m\}. \tag{16}$$

In general, the knots need not coincide with the set of x-coordinates in equation (1), nor is it required that $m = N$. In the following discussion, we will make these two assumptions but we will relax them quite soon. That is, we assume for the moment that,

$$\{k_0, k_1, ..., k_N\} = \{x_0, x_1, ..., x_N\}. \tag{17}$$

With these definitions in hand, a linear spline has the following form,

$$l(x) = a_0|x - k_0| + a_1|x - k_1| + \cdots + a_N|x - k_N|, \tag{18}$$

for $a_0, a_1, ..., a_N \in \mathbb{R}$. Clearly, $|x - k_i|$ is a piecewise linear function for $i = 0, 1, ..., N$. The question, as usual, is how to find the coefficients $a_i, i = 0, 1, ..., N$. We can proceed directly with the underlying system,

$$\begin{aligned} a_0|x_0 - x_0| + a_1|x_0 - x_1| + \cdots + a_N|x_0 - x_N| &= f_0, \\ a_0|x_1 - x_0| + a_1|x_1 - x_1| + \cdots + a_N|x_1 - x_N| &= f_1, \\ &\vdots \\ a_0|x_N - x_0| + a_1|x_N - x_1| + \cdots + a_N|x_N - x_N| &= f_N. \end{aligned} \tag{19}$$

We can write this more conveniently in matrix notation as,

$$\begin{bmatrix} 0 & |x_0 - x_1| & \cdots & |x_0 - x_N| \\ |x_1 - x_0| & 0 & \cdots & |x_1 - x_N| \\ \vdots & \vdots & \ddots & \vdots \\ |x_N - x_0| & |x_1 - x_N| & \cdots & 0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_N \end{bmatrix}, \tag{20}$$

or,

$$Va = f. \tag{21}$$

This direct approach gives us the Vandermonde matrix, V. In this setting, V is equally poorly conditioned and thus we will need a more general approach to the problem.

2.2 Cubic splines

Let's look at the most direct way to construct a cubic spline. With this approach, it is easy to see what is going on, and it works quite well for small problems. This approach is still not terribly convenient from a computational perspective. Consider the following three knot points,

$$\{k_0, k_1, k_2\}, \tag{22}$$

with the corresponding function values,

$$\{f_0, f_1, f_2\}. \tag{23}$$

To ease the notation, therefore, we will derive the cubic spline for this extremely simple example. With three endpoints, we have two subintervals $[k_0, k_1]$ and $[k_1, k_2]$. This implies that we require a separate cubic polynomial for each interval. We will define the piecewise cubic polynomial, $S(x)$, in the following obvious manner,

$$S(x) = \begin{cases} a_0 + a_1(x-k_0) + a_2(x-k_0)^2 + a_3(x-k_0)^3 : x \in [k_0, k_1] \\ b_0 + b_1(x-k_1) + b_2(x-k_1)^2 + b_3(x-k_1)^3 : x \in [k_1, k_2] \end{cases}. \tag{24}$$

The whole point of this exercise is to find the parameters of $S(x)$ (i.e., $a_0, .., a_3, b_0, ..., b_3$). The trick is to find the parameters associated with two piecewise polynomials that are equal in the level, the first derivative, and the second derivative at the knots. The introduction of these constraints, however, will help us solve what is currently a system of two equations in eight unknowns. First, we impose the following conditions,

$$S(k_i) = f_i, \tag{25}$$

for $i = 0, 1, 2$. The fact that our piecewise polynomials must pass through the values in equation (23) provides four conditions. These arise from evaluating $S(k_0)$ and $S(k_1)$. They are as follows,

$$a_0 = f_0, \tag{26}$$

$$a_0 + a_1(k_1 - k_0) + a_2(k_1 - k_0)^2 + a_3(k_1 - k_0)^3 = f_1, \tag{27}$$

$$b_0 = f_1, \tag{28}$$

$$b_0 + b_1(k_2 - k_1) + b_2(k_2 - k_1)^2 + b_3(k_2 - k_1)^3 = f_2. \tag{29}$$

To solve this system, we need an additional four conditions. The first step is to consider the first derivative of our piecewise polynomial function, which follows from equation (24),

$$S'(x) = \begin{cases} a_1 + 2a_2(x-k_0) + 3a_3(x-k_0)^2 : x \in [k_0, k_1] \\ b_1 + 2b_2(x-k_1) + 3b_3(x-k_1)^2 : x \in [k_1, k_2] \end{cases}. \tag{30}$$

The next condition arises by equating the first derivatives of the two pieces of $S(x)$ at the interior knot point, k_1. This permits the elimination of a number of terms and leads to the condition,

$$\begin{aligned} a_1 + 2a_2(k_1 - k_0) + 3a_3(k_1 - k_0)^2 &= b_1 + 2b_2(k_1 - k_1) + 3b_3(k_1 - k_1)^2, \\ a_1 + 2a_2(k_1 - k_0) + 3a_3(k_1 - k_0)^2 - b_1 &= 0. \end{aligned} \tag{31}$$

We can repeat this step for the second derivative,

$$S''(x) = \begin{cases} 2a_2 + 6a_3(x - k_0) : x \in [k_0, k_1] \\ 2b_2 + 6b_3(x - k_1) : x \in [k_1, k_2] \end{cases}. \tag{32}$$

That is, we set the two piecewise second derivatives equal to one another at the interior knot, k_1,

$$\begin{aligned} 2a_2 + 6a_3(k_1 - k_0) &= 2b_2 + 6b_3(k_1 - k_1), \\ 2a_2 + 6a_3(k_1 - k_0) - 2b_2 &= 0. \end{aligned} \tag{33}$$

This step joins the pieces of the cubic together in a *smooth* way and the resulting function will be twice continuously differentiable. This provides us with six conditions. At this point, we have some choice. The typical decision is to set the second derivatives at our two exterior knots k_0 and k_2 to zero. These two conditions define what is termed the *natural* cubic spline.[5] The final two conditions to complete our linear system, therefore, are,

$$S''(k_0) = 2a_3 = 0, \tag{34}$$

$$S''(k_2) = 2b_2 + 6b_3(k_2 - k_1) = 0. \tag{35}$$

Combining equations (26-29), (31), and (33-35) generates our linear system. To ease the notation somewhat, define

$$h_i = k_i - k_{i-1}, \tag{36}$$

for $i = 1, 2$. In matrix format, therefore, we have

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & h_1 & h_1^2 & h_1^3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & h_2 & h_2^2 & h_2^3 \\ 0 & 1 & 2h_1 & 3h_1^2 & 0 & -1 & 0 & 0 \\ 0 & 0 & 2 & 6h_1 & 0 & 0 & -2 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 6h_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{37}$$

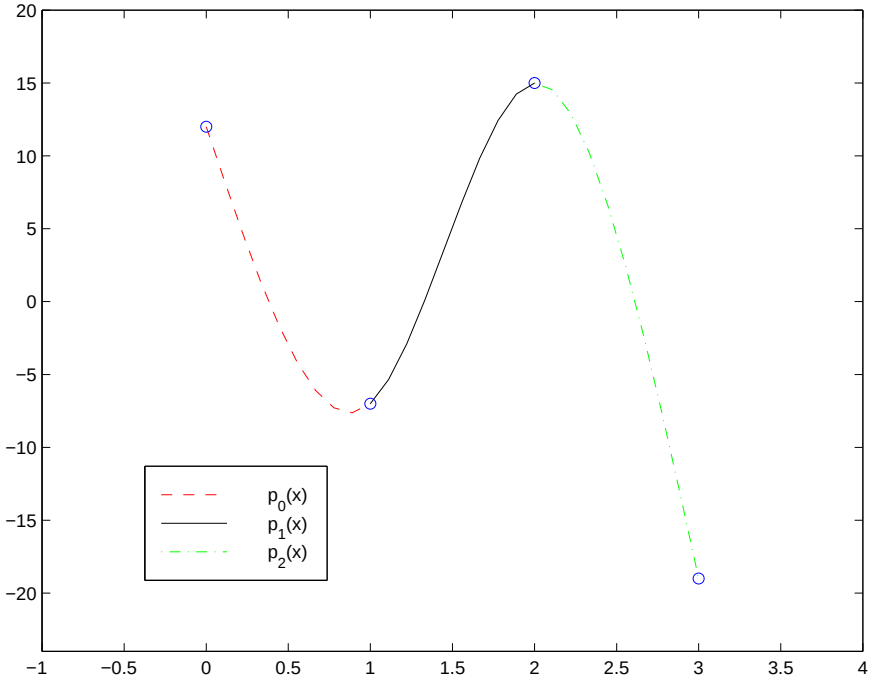
[5] In general, a natural cubic spline S has the property $S''(k_0) = S''(k_N) = 0$ for knot sequence $\{k_0, ..., k_N\}$.

or,

$$Va = f. \tag{38}$$

We would then, of course, solve this system in the usual way. Figure 1 demonstrates a simple natural cubic spline for four arbitrarily selected points fitted using this direct algorithm. Note that the cubic polynomials fit smoothly together at each of the function points occurring at the knots. These values are highlighted in Figure 1 with small circles.

Figure 1: **A Simple Cubic Spline**: This figure illustrates the unique *natural* cubic spline fit through the points $\{12, -7, 15, -19\}$ with knot sequence $\{0, 1, 2, 3\}$. The computation was performed using the direct cubic-spline method.



There are results that demonstrate that V is theoretically non-singular and thus the solution to the system described in equation (37) is unique. Nevertheless, as the number of knot points increases, this approach is awkward to implement and numerically unstable. Section 2.3 provides, for completeness, another potential approach to the construction of a cubic spline that is somewhat better. While it also suffers from numerical problems, it is an intermediate step towards a numerically stable approach. As a consequence, it will help us better understand our final approach that is less obvious to derive, in an algebraic sense, but leads to greater numerical stability and ease of implementation.

2.3 An intermediate cubic-spline derivation

If one works backwards to derive the cubic spline, it is possible to find a constructive algorithm for fitting the spline to a given set of points. As usual, we start with an arbitrary interval, $I = [a, b]$, partitioned by $N + 1$ knots into N subintervals, $I_i = [k_{i-1}, k_i]$ for $i = 0, ..., N$. Moreover, the knots are selected such that,

$$a = k_0 < k_1 < \cdots < k_N = b. \tag{39}$$

The starting point of this derivation is the fact that, if $S(x)$ is continuous piecewise cubic, then $S'(x)$ is continuous piecewise quadratic, and finally $S''(x)$ is continuous piecewise linear. We can use a special case of the Lagrange polynomial interpolation to write out this second derivative,

$$S''(x) = m_{i-1}\frac{k_i - x}{k_i - k_{i-1}} + m_i\frac{x - k_{i-1}}{k_i - k_{i-1}}, \tag{40}$$

on I_i and $m_i, m_{i-1} \in \mathbb{R}$ for $i = 0, ..., N$. Note that m_i is *not* playing the same role as it does in the subsequent derivation in section 2.4.[6] Finally, at each knot we know the value of our otherwise unknown function. We define these function values as $\{f_0, f_1, ..., f_N\}$.

We now proceed to integrate equation (40) twice to recover the original function, $S(x)$. To ease the notation, let's define $h_i = k_i - k_{i-1}$. The first integration yields,

$$\begin{aligned} S'(x) &= \int \left(m_{i-1}\frac{k_i - x}{h_i} + m_i\frac{x - k_{i-1}}{h_i} \right) dx, \\ &= -\frac{m_{i-1}}{2h_i}(k_i - x)^2 + \frac{m_i}{2h_i}(x - k_{i-1})^2 + C_i, \end{aligned} \tag{41}$$

for some $C_i \in \mathbb{R}$. The second integration provides,

$$\begin{aligned} S(x) &= \int \left(-\frac{m_{i-1}}{2h_i}(k_i - x)^2 + \frac{m_i}{2h_i}(x - k_{i-1})^2 + C_i \right) dx, \\ &= \frac{m_{i-1}}{6h_i}(k_i - x)^3 + \frac{m_i}{6h_i}(x - k_{i-1})^3 + C_i x + D_i, \end{aligned} \tag{42}$$

again for some constants $C_i, D_i \in \mathbb{R}$. To solve for these constants, we need to be somewhat clever about their form. Let us write them as,

$$\begin{aligned} C_i &= -c_i + d_i, \\ D_i &= c_i k_i - d_i k_{i-1}. \end{aligned} \tag{43}$$

This implies that,

$$\begin{aligned} C_i x + D_i &= (-c_i + d_i)x + (c_i k_i - d_i k_{i-1}), \\ &= c_i(k_i - x) + d_i(x - k_{i-1}), \end{aligned} \tag{44}$$

[6] In the subsequent notation, we have $S'(k_i) = m_i$, but in the current derivation it turns out that $S''(k_i) = m_i$ (see equation (40)). Owing to this change in notation, we expect the equations we derive here to be of a different form than previously in equation (73).

and thus we have,

$$S(x) = \frac{m_{i-1}}{6h_i}(k_i - x)^3 + \frac{m_i}{6h_i}(x - k_{i-1})^3 + c_i(k_i - x) + d_i(x - k_{i-1}). \tag{45}$$

This intermediate step comes to our assistance when combined with the fact that we know that,

$$\begin{aligned} S(k_{i-1}) = f_{i-1} &= \frac{m_{i-1}}{6h_i}(k_i - k_{i-1})^3 + c_i(k_i - k_{i-1}), \\ &= \frac{m_{i-1}}{6h_i}h_i^3 + c_i h_i, \\ &= \frac{m_{i-1}}{6}h_i^2 + c_i h_i, \end{aligned} \tag{46}$$

which implies that,

$$c_i = \frac{1}{h_i}\left(f_{i-1} - \frac{m_{i-1}h_i^2}{6}\right). \tag{47}$$

A similar calculation using $S(k_i) = f_i$ provides,

$$d_i = \frac{1}{h_i}\left(f_i - \frac{m_i h_i^2}{6}\right). \tag{48}$$

The point of integrating equation (40) twice was to determine the two constants of integration. In fact, this process has ensured that the cubic splines will actually pass through each of the knots. The next step is to force the first derivatives to be equal at the knots. Thus, we will have to differentiate equation (45) after, of course, plugging in the appropriate values from equations (46) and (47). For $x \in (k_{i-1}, k_i)$ we have

$$\begin{aligned} S'(x) &= \frac{\partial}{\partial x}\left[\frac{m_{i-1}}{6h_i}(k_i - x)^3 + \frac{m_i}{6h_i}(x - k_{i-1})^3 + \frac{1}{h_i}\left(f_{i-1} - \frac{m_{i-1}h_i^2}{6}\right) + \frac{1}{h_i}\left(f_i - \frac{m_i h_i^2}{6}\right)\right] \\ &= -\frac{m_{i-1}}{2h_i}(k_i - x)^2 + \frac{m_i}{2h_i}(x - k_{i-1})^2 - \frac{1}{h_i}\left(f_{i-1} - \frac{m_{i-1}h_i^2}{6}\right) + \frac{1}{h_i}\left(f_i - \frac{m_i h_i^2}{6}\right) \\ &= -\frac{m_{i-1}}{2h_i}(k_i - x)^2 + \frac{m_i}{2h_i}(x - k_{i-1})^2 + \frac{f_i - f_{i-1}}{h_i} + \frac{m_{i-1} - m_i}{6}h_i. \end{aligned} \tag{49}$$

Our objective here is to compute the limit of the first derivative, $S'(x)$, of our cubic polynomial defined on $[k_{i-1}, k_i]$ as it approaches k_i from the left. We also need to determine the limit of the first derivative of $S'(x)$ defined on $[k_i, k_{i+1}]$ as it approaches k_i from the right. As stated earlier, these two first derivatives must be equal. Let's, therefore, calculate these quantities. The left-hand-side limit is,

$$\begin{aligned} \lim_{x \uparrow k_i} S'(x) = S'(k_i^-) &= \underbrace{-\frac{m_{i-1}}{2h_i}(k_i - k_i)^2 + \frac{m_i}{2h_i}(k_i - k_{i-1})^2 + \frac{f_i - f_{i-1}}{h_i} + \frac{m_{i-1} - m_i}{6}h_i}_{\text{Equation (49) evaluated at } k_i}, \\ &= \frac{m_i}{2h_i}h_i^2 + \frac{f_i - f_{i-1}}{h_i} + \frac{m_{i-1} - m_i}{6}h_i, \\ &= \frac{m_i h_i}{3} + \frac{m_{i-1}h_i}{6} + \frac{f_i - f_{i-1}}{h_i}, \end{aligned} \tag{50}$$

and the right-hand-side limit is,

$$\lim_{x\downarrow k_i} S'(x) = S'(k_i^+) = \lim_{x\downarrow k_i} \left(\underbrace{-\frac{m_i}{2h_{i+1}}(k_{i+1}-x)^2 + \frac{m_{i+1}}{2h_{i+1}}(x-k_i)^2 + \frac{f_{i+1}-f_i}{h_{i+1}} + \frac{m_i - m_{i+1}}{6}h_{i+1}}_{\text{Equation (49) on interval } [k_i, k_{i+1}]} \right), \quad (51)$$

$$\begin{aligned}
&= -\frac{m_i}{2h_{i+1}}(k_{i+1}-k_i)^2 + \frac{m_{i+1}}{2h_{i+1}}(k_i-k_i)^2 + \frac{f_{i+1}-f_i}{h_{i+1}} + \frac{m_i-m_{i+1}}{6}h_{i+1}, \\
&= -\frac{m_i}{2h_{i+1}}h_{i+1}^2\frac{f_{i+1}-f_i}{h_{i+1}} + \frac{m_i-m_{i+1}}{6}h_{i+1}, \\
&= -\frac{m_i h_{i+1}}{3} - \frac{m_{i+1}h_{i+1}}{6} + \frac{f_{i+1}-f_i}{h_{i+1}}.
\end{aligned}$$

All that remains is to set $S'(k_i^-) = S'(k_i^+)$ and solve for the resulting conditions on our cubic spline. The result is,

$$\underbrace{\frac{m_i h_i}{3} + \frac{m_{i-1}h_i}{6} + \frac{f_i - f_{i-1}}{h_i}}_{\text{Equation (50)}} = \underbrace{-\frac{m_i h_{i+1}}{3} - \frac{m_{i+1}h_{i+1}}{6} + \frac{f_{i+1}-f_i}{h_{i+1}}}_{\text{Equation (51)}}, \quad (52)$$

$$\frac{h_i}{6}m_{i-1} + \frac{h_i+h_{i+1}}{3}m_i + \frac{h_{i+1}}{6}m_{i+1} = \frac{f_{i+1}-f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i},$$

for $i = 1, ..., N-1$. We have derived $N-1$ conditions for our spline model that are consistent with continuous second derivatives, equal first derivatives at the knots, and interpolation of the function values. We can streamline this not terribly convenient representation to assist us in putting these conditions into matrix format. Consider the following definitions,

$$\sigma_i = \frac{f_i - f_{i-1}}{h_i}, \quad (53)$$

$$\lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}}, \quad (54)$$

$$1 - \lambda_i = \frac{h_i}{h_i + h_{i+1}}, \quad (55)$$

$$d_i = \frac{6\left(\frac{f_{i+1}-f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i}\right)}{h_i + h_{i+1}} = \frac{6\left(\sigma_{i+1} - \sigma_i\right)}{h_i + h_{i+1}}. \quad (56)$$

To see how we use these expressions, we multiply equation (52) by $\frac{6}{h_i+h_{i+1}}$. This provides the much-abridged version of our $N-1$ conditions,

$$\begin{aligned}
\frac{h_i}{h_i+h_{i+1}}m_{i-1} + 2m_i + \frac{h_{i+1}}{h_i+h_{i+1}}m_{i+1} &= \frac{6\left(\frac{f_{i+1}-f_i}{h_{i+1}} - \frac{f_i-f_{i-1}}{h_i}\right)}{h_i+h_{i+1}} \quad (57) \\
(1-\lambda_i)m_{i-1} + 2m_i + \lambda_i m_{i+1} &= d_i,
\end{aligned}$$

for $i = 1, ..., N-1$. The final issue to resolve before we can actually write out our linear system involves the boundary conditions. In particular, we have $N-1$ conditions, but we have $N+1$ coefficients (i.e., m_i

where $i = 0, ..., N$). There are a number of ways to approach this question, but we opt for the *natural* spline where we impose $S''(k_0) = S''(k_N) = 0$. In our situation this implies that $\lambda_0 = d_0 = 1 - \lambda_N = d_N = 0$. This implies that our first condition is,

$$\begin{aligned} 2m_0 + \lambda_0 m_1 &= d_0, \\ 2m_0 &= 0, \end{aligned} \tag{58}$$

and our second condition is,

$$\begin{aligned} (1 - \lambda_N)m_{i-1} + 2m_N &= d_N, \\ 2m_N &= 0. \end{aligned} \tag{59}$$

We now have all the pieces to write out our linear system in full,

$$\begin{bmatrix} 2 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 1-\lambda_1 & 2 & \lambda_1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1-\lambda_2 & 2 & \lambda_2 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1-\lambda_{N-2} & 2 & \lambda_{N-2} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1-\lambda_{N-1} & 2 & \lambda_{N-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{N-2} \\ m_{N-1} \\ mN \end{bmatrix} = \begin{bmatrix} 0 \\ d_1 \\ d_2 \\ \vdots \\ d_{N-2} \\ d_{N-1} \\ 0 \end{bmatrix}, \tag{60}$$

or,

$$Vm = d. \tag{61}$$

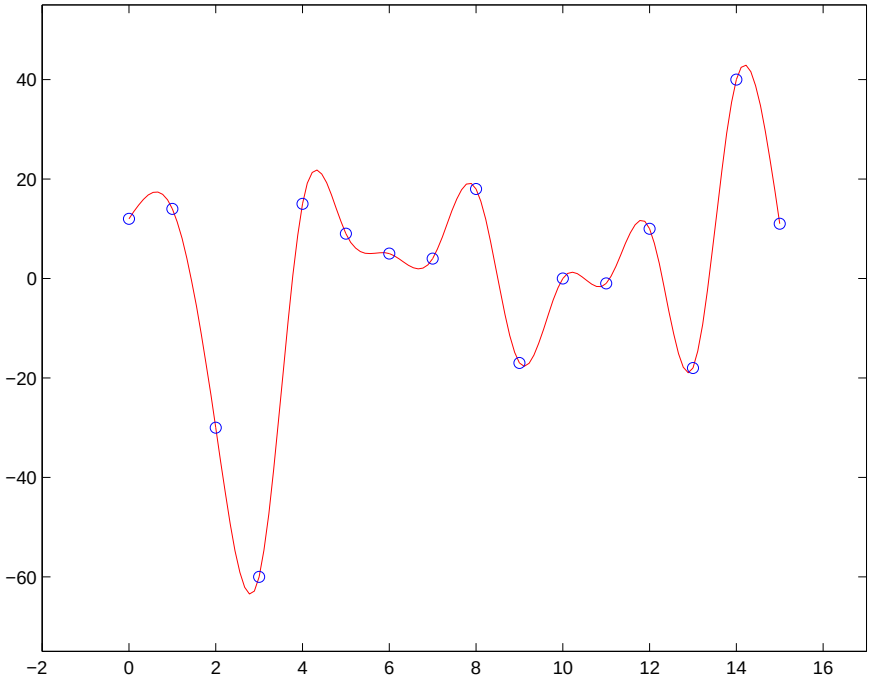
Observe that V is a tridiagonal matrix and there exist a variety of much more efficient algorithms for solving this system than merely inverting V.[7] This is the real advantage of this particular derivation of the cubic spline, compared with the direct approach described in section 2.2. That is, a more stable, general-purpose algorithm for the construction of a cubic spline can be created using this approach. Figure 2 shows the natural cubic spline fit to 15 distinct function values. This was performed using the straightforward MATLAB function provided in section A.1 of the appendix.

[7] One algorithm, in particular, involves the so-called LU-decomposition. That is, we decompose V into a lower-triangular matrix, L, and an upper triangular matrix, U, such that

$$LU = V. \tag{62}$$

The L and U matrices have quite convenient forms. The matrix L, for example, is composed of all ones on the main diagonal and a single non-zero value for each matrix entry just below these diagonal elements; all other elements are zero. The advantage is that this decomposition permits us to turn our initial matrix inversion into two subproblems that can be solved trivially by forward and backward substitution, given the simple form of our lower- and upper-triangular matrices, U and L. For a more detailed discussion of this algorithm see Press et al. (1992, pp. 43-48).

Figure 2: **The Tridiagonal Spline**: This figure illustrates the unique *natural* cubic spline, using the linear system described in equation (60), fit through the arbitrarily selected set of function points $\{12, 14, -30, -60, 15, 9, 5, 4, 18, -17, 0, -1, 10, -18, 40, 11\}$ with knot sequence $\{0, 1, ..., 14, 15\}$.



While this approach provides a fairly dramatic increase in the simplicity of implementation, it still has the potential to lead to numerical instability problems. Exactly why this is so can be seen from inspection of equations (53) to (56). Each of these key expressions in our linear system is a quotient of sums and differences of function values and knot points. Arbitrarily close function values and knot points, however, can lead to dividing a number by a value close to zero or dividing a very small number by another very small number. These types of computations can lead to significant roundoff errors and, hence, numerical instability. The approach to dealing with cubic splines, introduced in section 2.4, is a useful basis for the vector space of linear splines that greatly enhances the numerical stability of our calculations.

2.4 A final cubic-spline derivation: B-splines

In this section we use a basis, in a manner conceptually similar to the use of Lagrange polynomials, for the space of cubic splines—there are a number of possibilities but we use the popular *B-spline basis*. Constructing the B-spline basis is somewhat involved, but it provides a useful tool for the general construction of cubic splines. A B-spline is itself a cubic spline that takes positive values over only four adjacent subintervals in the

overall partition. On all other subintervals, the B-spline vanishes. When one defines a sequence of B-splines, each defined on its own four adjacent intervals, there are only four non-zero splines on any given subinterval in the overall partition of our arbitrary interval, $[a, b]$. We can show that the B-spline basis has the desirable property of the smallest possible support of any basis for the space of cubic splines. Moreover, and this is the key point, any cubic spline on $[a, b]$ can be constructed as a linear combination of this sequence of B-splines. Finally, because these B-splines are defined very narrowly, this linear combination is easy to compute and numerically stable.

The first step in the derivation of the B-spline follows Lancaster and Salkauskas (1986). We define the piecewise cubics as,

$$\Phi_i(x) = \begin{cases} 0 & x < k_{i-1} \\ -\frac{2}{h_i^3}(x - k_{i-1})^2(x - k_i - \frac{1}{2}h_i) & k_{i-1} \le x < k_i \\ \frac{2}{h_i^3}(x - k_i + \frac{1}{2}h_i)(x - k_{i+1})^2 & k_i \le x < k_{i+1} \\ 0 & x \ge k_{i+1} \end{cases}, \tag{63}$$

and

$$\Psi_i(x) = \begin{cases} 0 & x < k_{i-1} \\ \frac{1}{h_i^2}(x - k_{i-1})^2(x - k_i) & k_{i-1} \le x < k_i \\ \frac{1}{h_i^2}(x - k_i)(x - k_{i+1})^2 & k_i \le x < k_{i+1} \\ 0 & x \ge k_{i+1} \end{cases}, \tag{64}$$

for $i = 1, \ldots, N-1$. By construction, these piecewise cubics satisfy

$$\Phi_i(k_j) = \delta_{ij}, \tag{65}$$

$$\Phi_i'(k_j) = 0, \tag{66}$$

$$\Psi_i(k_j) = 0, \tag{67}$$

$$\Phi_i'(k_j) = \delta_{ij}, \tag{68}$$

and $\delta_{ij} = 1$ if $i = j$, and zero otherwise. Moreover, by a uniqueness theorem from Lancaster and Salkauskas (1986), we have the representation

$$S(x) = \sum_{i=0}^{N} f_i \Phi_i(x) + m_i \Psi_i(x), \tag{69}$$

where $f_i = S(k_i)$ and $m_i = S'(k_i)$.[8] To ensure that $S(x)$ is truly a spline, we demand that $S''(x)$ exists at each knot point. In other words, we impose the condition

$$S''(k_i^-) - S''(k_i^+) = 0, \tag{70}$$

[8]Technically, the definitions for $\Phi_0, \Phi_N, \Psi_0, \Psi_N$ are different. See Lancaster and Salkauskas (1986) for the details.

or equivalently

$$\sum_{j=i-1}^{i+1} f_j \left[\Phi_j''(k_i^-) - \Phi_j''(k_i^+)\right] + m_j \left[\Psi_j''(k_i^-) - \Psi_j''(k_i^+)\right] = 0, \tag{71}$$

for $i = 1, \ldots, N-1$. It is now easy to compute the second derivatives using the definitions of the Φ_j and Ψ_j above. We must be careful in choosing which part of the piecewise definition to use each time. As an example,

$$\begin{aligned}
\Phi_{i-1}(k_i^-) &= \lim_{x\uparrow k_i} \frac{2}{h_i^3}(x - k_{i-1} + \frac{1}{2}h_i)(x-k_i)^2, \\
\Phi'_{i-1}(k_i^-) &= \lim_{x\uparrow k_i} \frac{2}{h_i^3}\left[2(x - k_{i-1} + \frac{1}{2}h_i)(x-k_i) + (x-k_i)^2\right], \\
\Phi''_{i-1}(k_i^-) &= \lim_{x\uparrow k_i} \frac{2}{h_i^3}\left[4(x-k_i) + 2(x - k_{i-1} + \frac{1}{2}h_i)\right], \\
&= \frac{2}{h_i^3}\left[2(h_i + \frac{1}{2}h_i)\right], \\
&= \frac{6}{h_i^2}.
\end{aligned} \tag{72}$$

After doing the rest of the calculations similarly, the resulting $N-1$ conditions are

$$\frac{1}{h_i}m_{i-1} + 2\left(\frac{1}{h_i} + \frac{1}{h_{i+1}}\right)m_i + \frac{1}{h_{i+1}}m_{i+1} = 3\left(\frac{f_i - f_{i-1}}{h_i^2}\right) + 3\left(\frac{f_{i+1} - f_i}{h_{i+1}^2}\right), \tag{73}$$

for $i = 1, ..., N-1$. These equations could be compared with equation (52). The equations developed here, however, turn out to be much more convenient, particularly when we consider the case of equal spacing.

To better facilitate the construction of the B-spline, we will restrict our attention to four adjacent intervals $\{k_0, k_1, ..., k_4\}$ and set $h = h_i - h_{i-1}$ for all $i = 1, ..., 4$. Now, if we multiply equation (73) by $\frac{h}{2}$, we obtain,

$$\begin{aligned}
\frac{1}{2}m_{i-1} + 2m_i + \frac{1}{2}m_{i+1} &= \frac{3}{2h}(f_i - f_{i-1} + f_{i+1} - f_i), \\
\frac{1}{2}m_{i-1} + 2m_i + \frac{1}{2}m_{i+1} &= \frac{3}{2h}(-f_{i-1} + f_{i+1}),
\end{aligned} \tag{74}$$

for $i = 1, ..., 3$. If we set $f_0 = f_4 = m_0 = m_4 = 0$, we have

$$\begin{aligned}
2m_0 + m_1 &= \frac{3}{h}(-f_0 + f_1), \\
\frac{1}{2}m_0 + 2m_1 + \frac{1}{2}m_2 &= \frac{3}{2h}(-f_0 + f_2), \\
\frac{1}{2}m_1 + 2m_2 + \frac{1}{2}m_3 &= \frac{3}{2h}(-f_1 + f_3), \\
\frac{1}{2}m_2 + 2m_3 + \frac{1}{2}m_4 &= \frac{3}{2h}(-f_2 + f_4), \\
m_3 + 2m_4 &= \frac{3}{h}(-f_3 + f_4),
\end{aligned} \tag{75}$$

where the first and last expression are the boundary conditions necessary for a natural cubic spline. In matrix form, equation (75) translates into,

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ \frac{1}{2} & 2 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 2 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 2 & \frac{1}{2} \\ 0 & 0 & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ m_1 \\ m_2 \\ m_3 \\ 0 \end{bmatrix} = \frac{3}{h} \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ f_1 \\ f_2 \\ f_3 \\ 0 \end{bmatrix}. \tag{76}$$

Using this system, we will attempt to find those values for $m_1, ..., m_3, f_1, ..., f_3$ such that we create our desired B-spline basis for the space of cubic polynomials. This requires a bit of caution. Observe that if we select f_1, f_2, f_3 in an arbitrary manner, we cannot ensure that $f_0 = f_4 = 0$ as desired. In fact, it is our boundary conditions that provide the following two conditions relating our coefficients and function values. These are,

$$m_1 = \frac{3f_1}{h}, \tag{77}$$

$$m_3 = -\frac{3f_3}{h}. \tag{78}$$

The problem is that, given two equations and four unknowns, these restrictions are not particularly useful. The trick to solving this involves the interior linear system in equation (76) for m_1 and m_3. We can then proceed to find the necessary values of f_1 and f_3, in terms of f_2, to ensure that our desired conditions hold. The solution to the interior system, therefore, is,

$$\begin{bmatrix} 2 & \frac{1}{2} & 0 \\ \frac{1}{2} & 2 & \frac{1}{2} \\ 0 & \frac{1}{2} & 2 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = \frac{3}{2h} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}, \tag{79}$$

$$\begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = \begin{bmatrix} 2 & \frac{1}{2} & 0 \\ \frac{1}{2} & 2 & \frac{1}{2} \\ 0 & \frac{1}{2} & 2 \end{bmatrix}^{-1} \frac{3}{2h} \begin{bmatrix} f_2 \\ f_3 - f_1 \\ -f_2 \end{bmatrix},$$

$$= \begin{bmatrix} \frac{3}{14h}\left(\frac{14}{4}f_2 + f_1 - f_3\right) \\ 6(f_3 - f_1) \\ \frac{3}{14h}\left(-\frac{14}{4}f_2 - f_1 - f_3\right) \end{bmatrix}.$$

Equating the values in equations (77) and (78) with the solution from equation (79) creates the following two equations,

$$\begin{aligned} 13f_1 + f_3 &= \frac{14}{4}f_2, \\ -f_1 - 13f_3 &= -\frac{14}{4}f_2, \end{aligned} \tag{80}$$

or, in matrix form,

$$\begin{bmatrix} 13 & 1 \\ -1 & -13 \end{bmatrix} \begin{bmatrix} f_1 \\ f_3 \end{bmatrix} = \frac{14}{4} \begin{bmatrix} f_2 \\ -f_2 \end{bmatrix}, \tag{81}$$
$$\begin{bmatrix} f_1 \\ f_3 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} f_2 \\ f_2 \end{bmatrix},$$

implying that $f_1 = f_3 = \frac{f_2}{4}$. We are, of course, free to select f_2 as we wish, but it is convenient to set $f_2 = \frac{2}{3}$ because this permits $f_1 + f_2 + f_3 = \frac{1}{6} + \frac{2}{3} + \frac{1}{6} = 1$.

We have now defined our B-spline. It is the cubic spline on $\{k_0, ..., k_4\}$ such that the following set of straightforward conditions hold,

$$\begin{aligned} f_0 = f_4 = m_1 = m_4 = m_2 = 0, \\ f_1 = f_3 = \frac{1}{6}, \\ f_2 = \frac{2}{3}, \\ m_1 = \frac{1}{2h}, \\ m_3 = -\frac{1}{2h}. \end{aligned} \tag{82}$$

Let us denote this cubic spline as the B-spline, $\bar{B}_0(x)$.[9] Typically, a spline is defined more generally on an arbitrary interval $[k_i, k_{i+4}]$ in the following manner,

$$B_i(x) = \begin{cases} 0 : x \in (\infty, k_i) \\ \bar{B}_i(x) : x \in [k_i, k_{i+4}] \\ 0 : x \in (k_{i+4}, \infty) \end{cases}. \tag{83}$$

That is, in its formal definition, we add identically zero extensions to our B-spline defined on $[k_i, k_{i+4}]$.

This is all interesting, but the question remains as to how we employ these mathematical objects in the construction of cubic splines. We must first discuss how we might construct a basis for the cubic splines on a given interval. We do not generally talk about a single B-spline, but rather consider a sequence of B-splines. For example, to create a basis for the knot sequence $\{k_0, ..., k_N\}$, we would require the collection,

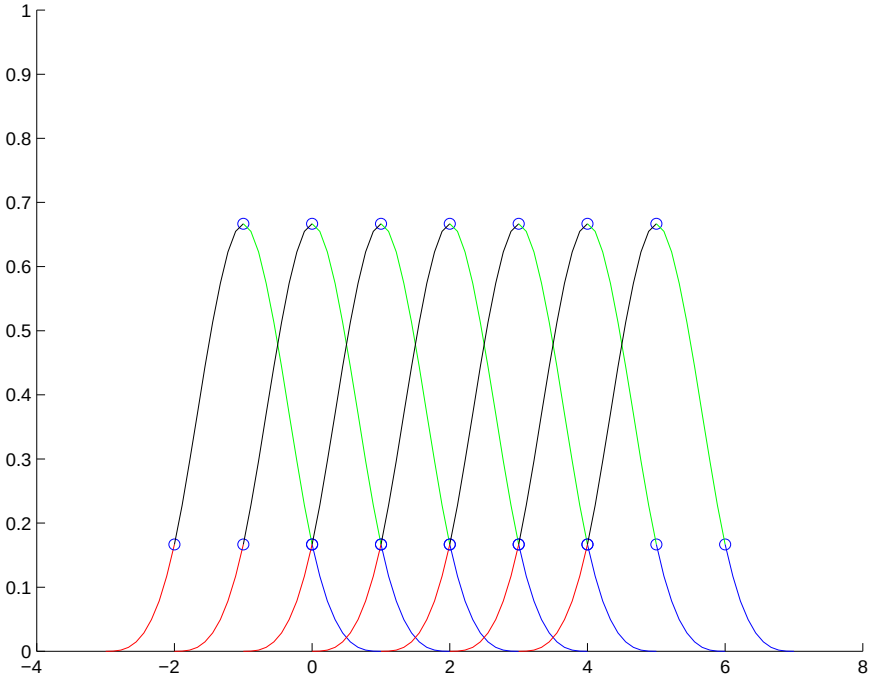
$$\{B_{-3}, B_{-2}, ..., B_{N-1}\}, \tag{84}$$

comprising $N + 3$ B-splines defined on $\{k_{-3}, ..., k_{N+3}\}$. Figure 3 illustrates the B-spline basis on the knot sequence $\{0, 1, 2, 3, 4\}$. Using Figure 3, we may visually verify that on any given interval, there are at most

[9] Apparently, the "B" in B-spline represents *basic* and was coined by the mathematician Schoenberg.

four non-zero B-splines.[10]

Figure 3: **The B-spline Basis on** $[0, 4]$: This figure illustrates the seven B-splines necessary to form a basis for the vector space of cubic polynomials defined on $[0, 4]$. Observe that on any given subinterval in $[0, 4]$ there are at most four non-zero splines.



Armed with this sequence of $N + 3$ splines for given a knot sequence $\{k_0, ..., k_N\}$, it turns out that,

$$S(x) = \sum_{i=-3}^{N} a_i B_i(x). \tag{86}$$

Or, in other words, a cubic spline can be written as a linear combination of the B-spline basis. Equation (86) has $N + 3$ coefficients for $N + 1$ function values, so this representation is not necessarily unique. For a given set of boundary conditions, such as the natural spline conditions $S''(k_0) = S''(k_N) = 0$, it is a unique representation.

Before we can actually proceed to demonstrate how to find the coefficients a_i for $i = -3, ..., N$ as described in equation (86), we need a general-purpose method for evaluating B-splines for an arbitrary point

[10] B-splines also have the interesting property that for any arbitrarily selected knot k_i,

$$\sum_{j=-3}^{1} B_{i-j}(x) = 1, \tag{85}$$

for all $x \in \{k_0, ..., k_N\}$. In the spline literature, this property is described as a partition of unity. This follows from our seemingly haphazard selection of $f_2 = \frac{2}{3}$.

$x \in (k_i, k_{i+1})$. We know, for example, the value of the B-spline at the knot points in this interval, k_i and k_{i+1}, but we need a simple way to find the intermediate points. This is essential if we are to construct a general algorithm for determining any given cubic spline as a linear combination of the B-spline basis.

Fortunately, there is a recursive formula that we can use to accomplish exactly this objective. To write out the recursion formula, we need to introduce the idea of the degree of a B-spline basis. We have, in our previous discussion, focused on a cubic B-spline basis. Technically, an n-order B-spline with knot sequence $\{k_0, ..., k_N\}$ is actually a $(n-1)$th degree polynomial that is $n-2$ times continuously differentiable (i.e., an element of the set $C^{(n-2)}$) on $\{k_{-3}, ..., k_3\}$.[11] Thus, a cubic B-spline has order equal to four; moreover, we denote the ith B-spline of order n as,

$$B_{i,n}(x). \tag{87}$$

The order of the B-spline is important because the B-spline recursion formula writes the B-spline in terms of B-splines of lesser order. It has the following, rather uninviting, form,

$$B_{i,n}(x) = \frac{x - k_i}{k_{i+n-1} - k_i} B_{i,n-1}(x) + \frac{k_{i+n} - x}{k_{i+n} - k_{i+1}} B_{i+1,n-1}(x), \tag{88}$$

for $i = -3, ..., N-1$ and $n = 1, ..., 4$.[12] To actually use this handy formula, one needs to know how to define $B_{i,1}$, because it is the final point in the recursion. Knowledge of $B_{i,1}$ is sufficient to determine any value of our cubic B-spline of interest, $B_{i,4}$. The first-order B-spline, therefore, is conventionally defined as the *right-continuous* indicator function,

$$B_{i,1}(x) = \mathbb{1}_{[k_i, k_{i+1})} = \begin{cases} 0 : x \in (\infty, k_i) \\ 1 : x \in [k_i, k_{i+1}) \\ 0 : x \in [k_{i+1}, \infty) \end{cases} . \tag{89}$$

Using equations (88) and (89), it is straightforward to evaluate a given cubic B-spline at any point $x \in (k_i, k_{i+1})$. See section A.2 of the appendix for a simple piece of code written in MATLAB operationalizing this algorithm.

The final step involves determination of the coefficients. We require the fact that, by construction,

$$\begin{aligned} B_{i-3}(k_i) &= \frac{1}{6}, \\ B_{i-2}(k_i) &= \frac{2}{3}, \\ B_{i-1}(k_i) &= \frac{1}{6}, \\ B_i(k_i) &= 0, \end{aligned} \tag{90}$$

[11] This means that the first $N-2$ derivatives on $\{k_{-3}, ..., k_3\}$ are continuous. In the case of a cubic B-spline, therefore, the first and second derivatives are continuous.

[12] This recursion relation follows from Leibniz's divided-difference formula. For a detailed derivation, see deBoor (1978, pp. 130-131).

and by equation (86), and the necessary boundary conditions, we can construct a linear system. It has the following form,

$$\begin{bmatrix} 1 & -2 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} a_{-3} \\ a_{-2} \\ a_{-1} \\ \vdots \\ a_{N-3} \\ a_{N-2} \\ a_N \end{bmatrix} = \begin{bmatrix} 0 \\ f_0 \\ f_1 \\ \vdots \\ f_{N-1} \\ f_N \\ 0 \end{bmatrix}, \tag{91}$$

or,

$$Va = f. \tag{92}$$

V is almost tridiagonal and consequently easy to invert. Moreover, it is both known entirely in advance, given N, and is not a function of differences in function values or knot points, as it was in the previous algorithm. As a result, it is numerically very stable. Figure 4 demonstrates two splines generated using the indirect tridiagonal approach and the B-spline approach. Section A.3 of the appendix gives the code to implement the cubic spline.

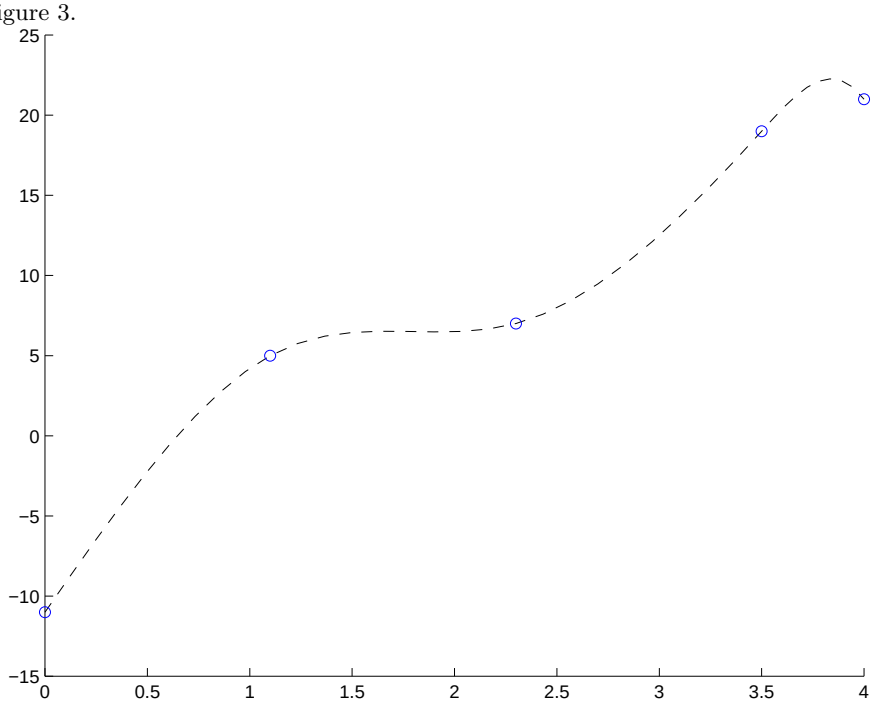
2.5 Least-squares estimation

One key difference between our financial problem and the previous situation is the fact that we do not directly observe the datapoints we are trying to fit. Instead, we are trying to extract zero-coupon rates from an observed sample of coupon-bearing bonds. We will have to use our curve to determine a set of *theoretical* bond prices associated with this set of zero-coupon rates. Our goal will be to find a set of parameters that provides the *best* fit to the observed bond prices. To make this work, we will have to define what *best* means in some quantitative sense. Because of its attractive mathematical properties, one generally attempts to minimize the sum of squared errors, or equivalently the ℓ^2-norm. Indeed, we will try to minimize the following quantity,

$$\ell^2(S) \triangleq \sum_{i=0}^{N} \left(S(x_i) - f_i\right)^2, \tag{93}$$

where, $S \in \mathcal{P}_m$ for $m < N$ where, in this context, N is the number of knot points. In other words, in the general problem one is trying to find a polynomial of degree m that minimizes the squared deviations from the observed data points. In our specific setting, however, we are attempting to find the cubic B-spline, S,

Figure 4: **The B-spline in Action**: This graph outlines a cubic spline, constructed using the B-spline basis, to five randomly selected data points. This was constructed as a linear combination of the B-spline basis functions summarized in Figure 3.



of the form described in equation (86),

$$S(x) = \sum_{j=-3}^{m} a_j B_j(x). \tag{94}$$

That is, we are trying to find the set of coefficients, $a_j, j = -3, ..., m$, that minimizes equation (93). The set of first-order conditions of the optimization problem requires that the partial derivatives of $\ell^2(S)$ with respect to the coefficients $a_j, j = -3, ..., m$ must vanish. Or,

$$\frac{\partial \ell^2(S)}{\partial a_j} = 0, \tag{95}$$

for $j = -3, ..., m$. We observe, from inspection of equation (94), that each of these partial derivatives has the following form,

$$\frac{\partial S(x)}{\partial a_j} = B_j(x), \tag{96}$$

and use this to evaluate our set of first-order conditions,

$$\begin{aligned}
\frac{\partial \ell^2(S)}{\partial a_j} &= 0, \qquad (97)\\
\frac{\partial}{\partial a_j}\left(\sum_{i=0}^{N}\left(S(x_i)-f_i\right)^2\right) &= 0,\\
\sum_{i=0}^{N} 2\left(\underbrace{\sum_{j=-3}^{m} a_j B_j(x_i)}_{\text{Equation (94)}} - f_i\right)\frac{\partial S(x)}{\partial a_j} &= 0,\\
\sum_{i=0}^{N} 2\left(\sum_{j=-3}^{m} a_j B_j(x_i) - f_i\right)\underbrace{B_j(x_i)}_{\text{Equation (96)}} &= 0,\\
\sum_{i=0}^{N}\left[B_j(x_i)\left(\sum_{j=-3}^{m} a_j B_j(x_i)\right) - B_j(x_i) f_i\right] &= 0.
\end{aligned}$$

The idea is to put this into a (hopefully) linear system and solve for the coefficients $a_j, j = -3, ..., m$. As a first step, we have the subsequent $m+3$ equations,

$$\begin{aligned}
a_{-3}\sum_{i=-3}^{N} B_{-3}^2(x_i) + a_{-2}\sum_{i=0}^{N} B_{-3}(x_i)B_{-2}(x_i) + \cdots + a_m\sum_{i=0}^{N} B_{-3}(x_i)B_m(x_i) &= \sum_{i=0}^{N} B_{-3}(x_i) f_i, \qquad (98)\\
a_{-3}\sum_{i=-3}^{N} B_{-2}(x_i)B_{-3}(x_i) + a_{-2}\sum_{i=0}^{N} B_{-2}^2(x_i) + \cdots + a_m\sum_{i=0}^{N} B_{-2}(x_i)B_m(x_i) &= \sum_{i=0}^{N} B_{-2}(x_i) f_i,\\
&\vdots\\
a_{-3}\sum_{i=-3}^{N} B_m(x_i)B_{-3}(x_i) + a_{-2}\sum_{i=0}^{N} B_m(x_i)B_{-2}(x_i) + \cdots + a_m\sum_{i=0}^{N} B_m^2(x_i) &= \sum_{i=0}^{N} B_m(x_i) f_i.
\end{aligned}$$

These are generally termed the *normal* equations. In matrix form, we have

$$\begin{bmatrix}
\sum_{i=-3}^{N} B_{-3}^2(x_i) & \sum_{i=0}^{N} B_{-3}(x_i)B_{-2}(x_i) & \cdots & \sum_{i=0}^{N} B_{-3}(x_i)B_m(x_i)\\
\sum_{i=-3}^{N} B_{-2}(x_i)B_{-3}(x_i) & \sum_{i=0}^{N} B_{-2}^2(x_i) & \cdots & \sum_{i=0}^{N} B_{-2}(x_i)B_m(x_i)\\
\vdots & \vdots & \ddots & \vdots\\
\sum_{i=-3}^{N} B_m(x_i)B_{-3}(x_i) & \sum_{i=0}^{N} B_m(x_i)B_{-2}(x_i) & \cdots & \sum_{i=0}^{N} B_m^2(x_i)
\end{bmatrix}
\begin{bmatrix} a_{-3}\\ a_{-2}\\ \cdots\\ a_m \end{bmatrix}
=
\begin{bmatrix} \sum_{i=0}^{N} B_{-3}(x_i) f_i\\ \sum_{i=0}^{N} B_{-2}(x_i) f_i\\ \sum_{i=0}^{N} B_m(x_i) f_i \end{bmatrix}. \qquad (99)$$

This is a well-known linear system and can be economically represented with a few clever matrix definitions. First, we define

$$V = \begin{bmatrix} B_{-3}(x_1) & B_{-2}(x_1) & \cdots & B_m(x_1) \\ B_{-3}(x_2) & B_{-2}(x_2) & \cdots & B_m(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ B_{-3}(x_N) & B_{-2}(x_N) & \cdots & B_m(x_N) \end{bmatrix}, \tag{100}$$

where $V \in \mathbb{R}^{N \times (m+3)}$. Then, we construct

$$a = \begin{bmatrix} a_{-3} & a_{-2} & \cdots & a_m \end{bmatrix}^T, \tag{101}$$

$$f = \begin{bmatrix} f_0 & f_1 & \cdots f_N \end{bmatrix}^T, \tag{102}$$

where $a \in \mathbb{R}^{m+3}$ and $f \in \mathbb{R}^N$, respectively. The definitions in equations (100) to (102) allow us to collapse equation (99) into,

$$V^T V a = V^T f, \tag{103}$$

which provides us with the well-known least-squares solution,

$$a = (V^T V)^{-1} V^T f. \tag{104}$$

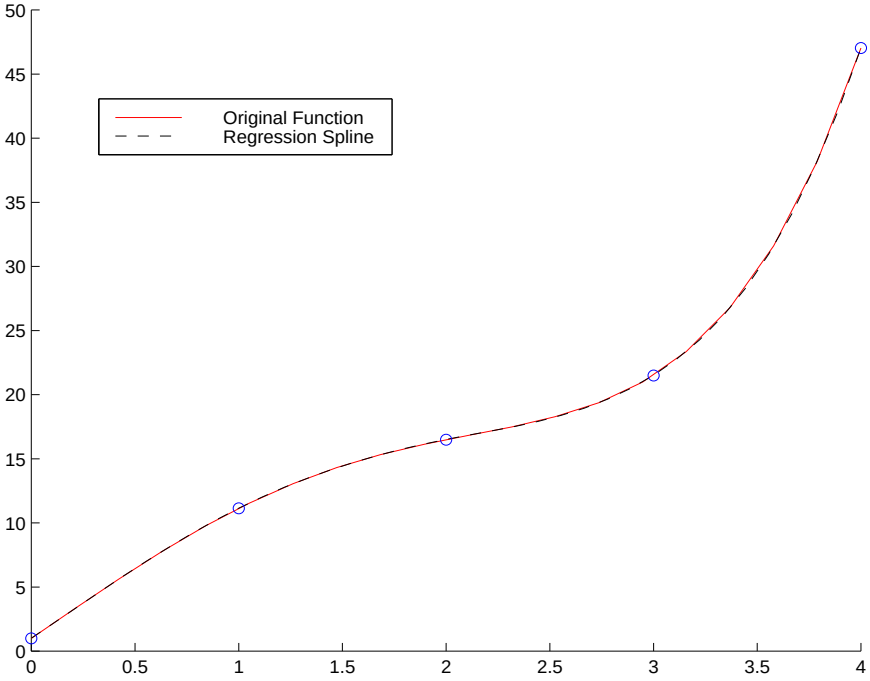
By construction, $V^T V$ is symmetric and positive definite. Moreover, given the nature of the B-splines, it has a large number of zero entries (i.e., it is a sparse matrix). Thus, solving this system is computationally straightforward and fast. Figure 5 illustrates the previously described approach using a B-spline with knot sequence $\{0, 1, 2, 3, 4\}$ fit by regression to the *made-up* example function, $f(x) = e^x + 10\sin(x)$, sampled at 20 x-coordinates in the interval, $[0, 4]$.[13] It does a good job of fitting the function, although f is not tremendously complicated. The MATLAB code for this implementation is outlined in section A.4 of the appendix.

2.6 Smoothing splines

In section 2.5, we considered the idea of fitting a cubic spline with m knots to N observations where, of course, $m < N$. As m approaches N, the ability of our cubic spline to fit the observed data increases. In the limit, the squared deviations from the observed function points will tend to zero. Forcing these errors to zero, however, may not be our objective. In fact, we may wish to impose some kind of smoothness onto the overall cubic spline. We could always reduce the number of m knots, but this may lead to an undesirable

[13] There is nothing special about f; it is an arbitrary function fabricated for illustrative purposes.

Figure 5: **The B-spline Basis on** $[0, 4]$: This figure illustrates the results of a B-spline with knot sequence $\{0, 1, 2, 3, 4\}$ fit by regression to the function $f(x) = e^x + 10\sin(x)$, sampled at 20 x-coordinates in the interval $[0, 4]$.



reduction in goodness of fit. The solution to this fit-smoothness trade-off involves the addition of an extra term to the least-squares objective function described in equation (93). Consider the following integral,

$$\mathcal{G}(S) \triangleq \int_a^b (S''(x))^2 dx. \tag{105}$$

This integral is a proxy for the smoothness of the function, x. More formally, it is a measure of curvature. Note that if S is a linear function, its first derivative is a constant and its second derivative vanishes. Thus, a linear fit to the data provides, by this criterion, an optimal level of smoothness. Nevertheless, a linear fit will not generally be optimal in terms of minimizing squared deviations from the observed data. As such, it is common practice to construct a new objective function composed of equations (93) and (105) as follows,

$$\begin{aligned} \mathcal{H}(S) &= \ell^2(S) + \lambda \mathcal{G}(S), \\ &= \sum_{i=0}^{N} \left(S(x_i) - f_i\right)^2 + \lambda \int_a^b (S''(x))^2 dx, \end{aligned} \tag{106}$$

where λ is a parameter that determines the relative importance of goodness *versus* smoothness of fit to the observed data.

To actually implement this idea, however, we will need to spend some time determining the derivatives of

our B-spline basis. It can be shown, although it is somewhat tedious, that the derivative of a cubic B-spline with equally spaced knots has the following form,

$$\begin{aligned}\sum_{j=-3}^{m} a_j B'_{j,4}(x) &= \frac{1}{h}\sum_{j=-3}^{m} a_j B_{j,3}(x) - \frac{1}{h}\sum_{j=-3}^{m} a_{j+1} B_{j+1,3}(x), \\ &= \frac{1}{h}\sum_{j=-2}^{m} (a_j - a_{j+1}) B_{j,3}(x), \\ &= -\frac{1}{h}\sum_{j=-2}^{m} \Delta a_{j+1} B_{j,3}(x),\end{aligned} \tag{107}$$

where $\Delta a_j = a_j - a_{j-1}$ is the first-difference operator.[14] Thus, not surprisingly, one represents the derivatives of a B-spline as a function of B-splines of lesser order. A second application of equation (107) generates the desired second derivative,

$$\sum_{j=-3}^{m} a_j B''_{j,4}(x) = \frac{1}{h^2}\sum_{j=-1}^{m} \Delta^2 a_j B_{j,2}(x), \tag{108}$$

where $\Delta^2 a_j = a_j - 2a_{j-1} + a_{j-2}$ is the second-difference operator. Using these identities, we can proceed to find a reasonable expression for $\mathcal{G}(S)$. Consider,

$$\begin{aligned}\mathcal{G}(S) &= \int_a^b (S''(x))^2 dx, \\ &= \int_a^b \left(\underbrace{\frac{1}{h^2}\sum_{j=-1}^{m} \Delta^2 a_j B_{j,2}(x)}_{\text{Equation (108)}} \right)^2 dx, \\ &= \frac{1}{h^4}\int_a^b \left(\sum_{j=-1}^{m}\sum_{k=-1}^{m} \Delta^2 a_j \Delta^2 a_k B_{j,2}(x) B_{k,2}(x) \right) dx.\end{aligned} \tag{109}$$

This is where the structure of the B-spline basis comes to our aid. Most of the terms $B_{j,2}(x)B_{k,2}(x)$ are zero, because second-order (linear) B-splines overlap only on $j = k-1, k, k+1$. Moreover, they are symmetric

[14] See deBoor (1978, pp. 138-139) or Nürnberger (1980, pp. 104-105) for a full description and proof of this property of B-splines.

about $j = k$, so we can replace our double sum as,

$$h^4\mathcal{G}(S) = \int_a^b \left[\sum_{j=-1}^{m} \left(\underbrace{\Delta^2 a_j B_{j,2}(x)}_{\text{For } j=k} \right)^2 + 2 \sum_{j=-1}^{m} \left(\underbrace{\Delta^2 a_j \Delta^2 a_{j-1} B_{j,2}(x) B_{j-1,2}(x)}_{\text{For } j=k-1,\,k+1} \right) \right] dx, \tag{110}$$

$$= \sum_{j=-1}^{m} (\Delta^2 a_j)^2 \underbrace{\int_a^b \left(B_{j,2}(x)\right)^2 dx}_{\text{Call this } \alpha} + 2 \sum_{j=-1}^{m} \Delta^2 a_j \Delta^2 a_{j-1} \underbrace{\int_a^b B_{j,2}(x) B_{j-1,2}(x) dx}_{\text{Call this } \beta},$$

$$= \alpha \sum_{j=-1}^{m} (\Delta^2 a_j)^2 + 2\beta \sum_{j=-1}^{m} \Delta^2 a_j \Delta^2 a_{j-1}.$$

For equally spaced knot points, the integrals, α and β, are constant. Despite the simpification, equation (110) is a fairly involved expression. Eilers and Marx (1996) realized that a good approximation for $\mathcal{G}(S)$ is given by,

$$\mathcal{G}(S) \approx \sum_{j=-1}^{m} (\Delta^2 a_j)^2. \tag{111}$$

From a computational perspective, this is a useful expression. Consider, in the context of the optimization of equation (106), the set of partial derivatives with respect to a_j,

$$\frac{\partial \mathcal{G}(S)}{\partial a_j} = \frac{\partial}{\partial a_j} \left(\sum_{j=-1}^{m} (\Delta^2 a_j)^2 \right), \tag{112}$$

$$= 2 \sum_{j=-1}^{m} (\Delta^2)^2 a_j,$$

$$= 2 D^T D a$$

in matrix form where D is the second-difference operator in matrix form. In matrix form, for a five-parameter problem, D is the following 3×5 tridiagonal matrix,

$$D = \begin{bmatrix} 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix}. \tag{113}$$

The consequence is that we can continue to solve our problem in a least-squares setting. Combining the results of equation (113) with equations (103) and (106), we have the following set of first-order conditions in matrix form,

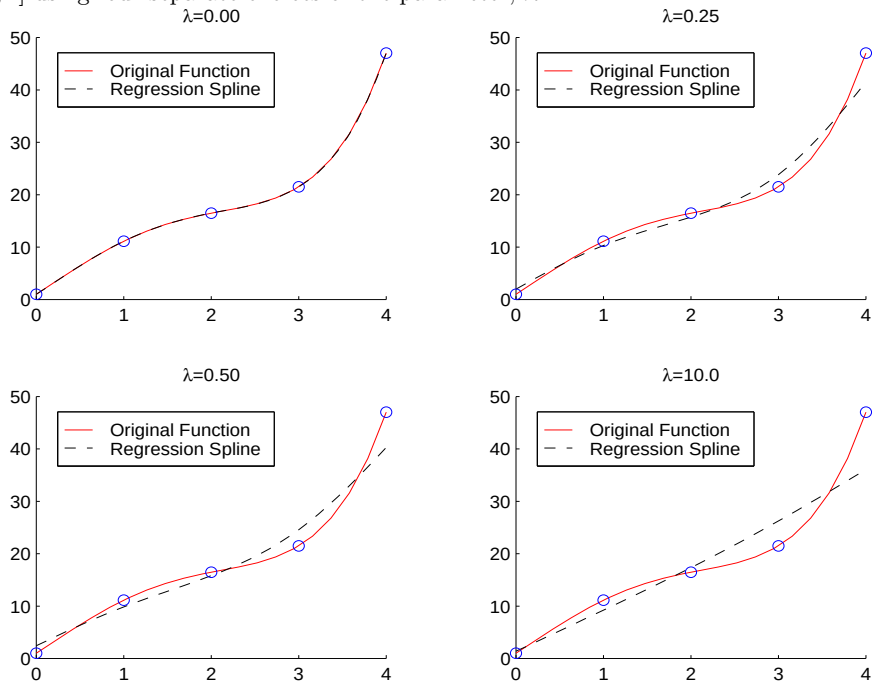
$$V^T V a - V^T f + \lambda D^T D a = 0, \tag{114}$$

$$(V^T V + \lambda D^T D) a = V^T f,$$

$$a = (V^T V + \lambda D^T D)^{-1} V^T f.$$

Figure 6 outlines the application of this smoothing spline using the same problem outlined in Figure 5 for four different values of λ. As λ grows large, the solution to the problem, in line with our intuition, tends towards a straight line.[15]

Figure 6: **Smoothed B-splines on** $[0, 4]$: This figure illustrates the results of four smoothed B-splines with knot sequence $\{0, 1, 2, 3, 4\}$ fit by smoothed regression to the function, $f(x) = e^x + 10\sin(x)$, sampled at 20 x-coordinates in the interval $[0, 4]$ using four separate choices of the parameter, λ.



This concludes the necessary background for our discussion of the spline-based models. We now examine the eight separate term-structure models that will participate in our *horse race* in section 4.

3 The Models

In this section, we will examine eight separate term-structure models in some depth. In the discussion that follows, we will be working with the primitive objects of fixed-income markets: pure-discount bonds, zero-coupon rates, discount functions, and instantaneous forward rates. Given any one of these objects, we

[15] No associated code is provided in the appendix, because implementation of this algorithm requires the additional two lines,

```
D=diff(eye(m-4),2);
c=inv(E'*E+g(r)*D'*D)*E'*f';
```

to `regSpline.m` in section A.4.

can perform the necessary transformation to find the others. As such, term-structure estimation techniques choose different points of entry for their modelling. Five of our models, for example, work with the discount function, two of the models in our group begin with the instantaneous forward rate, and one model uses the zero-coupon rate as its entry point. Understanding the relationships between these various fixed-income objects is, therefore, very important to understanding the distinctions between the different methodologies described in this section.

This introductory section is thus dedicated to a summary of the necessary notation and the derivation of these essential relationships. Let's begin with the most fundamental concept. We denote a pure discount bond price maturing at time T as $d(t,T)$. It has the following definition,

$$d(t,T) = e^{-(T-t)z(t,T)}, \tag{115}$$

where $z(t,T)$ denotes the zero-coupon interest rate prevailing from t until time T. Of course, inverting equation (115) provides the definition of the zero-coupon rate,

$$z(t,T) = -\frac{\ln d(t,T)}{T-t}. \tag{116}$$

The instantaneous forward interest rate at time t and for time T is given as,

$$f(t,T) = \frac{\partial}{\partial T}\left(-\ln d(t,T)\right). \tag{117}$$

This definition of the instantaneous forward rate is not particularly useful for computation. The underlying manipulation, however, is somewhat better,

$$\begin{aligned}
\frac{\partial}{\partial T}\left(-\ln d(t,T)\right) &= \underbrace{-\frac{1}{d(t,T)}d'(t,T)}_{\text{By chain rule}}, \\
&= -\frac{1}{d(t,T)}d'(t,T) + \frac{\ln d(t,T)}{T-t} - \frac{\ln d(t,T)}{T-t}, \\
&= -\frac{\ln d(t,T)}{T-t} - (T-t)\left(\frac{1}{(T-t)}\frac{1}{d(t,T)}d'(t,T) - \frac{\ln d(t,T)}{(T-t)^2}\right), \\
&= z(t,T) - (T-t)\left(\underbrace{\frac{\partial}{\partial T}\frac{\ln d(t,T)}{T-t}}_{\substack{\text{By chain and} \\ \text{quotient rules}}}\right), \\
&= z(t,T) + (T-t)z'(t,T).
\end{aligned} \tag{118}$$

It is also easy to see that, using equation (119), there is another way to represent the zero-coupon rate

defined in equation (116). In particular, we have,

$$f(t,T) = \frac{\partial}{\partial T}\left(-\ln d(t,T)\right), \tag{119}$$

$$\int_t^T f(t,u)du = \int_t^T \frac{\partial}{\partial T}\left(-\ln d(t,u)\right)du,$$

$$\frac{\int_t^T f(t,u)du}{T-t} = \frac{\overbrace{-\ln d(t,T) + \ln d(t,t)}^{\text{Recall that } d(t,t)=1}}{T-t},$$

$$\frac{\int_t^T f(t,u)du}{T-t} = \frac{-\ln d(t,T)}{T-t} = z(t,T).$$

This identity is important in the derivation of the Svensson model.

In the discussion that follows, for notational convenience, we will suppress the first argument for the discount function, the zero-coupon rate, and the instantaneous interest rate. We also introduce the following object defined as,

$$\ell(T) \stackrel{\triangle}{=} Tz(T). \tag{120}$$

This will prove useful in our examination of the Fisher, Nychka, and Zervos (1994) model. This concludes our brief review of our notation and the primitive objects in fixed-income markets. For more background, see the excellent references in Campbell, Lo, and MacKinlay (1997, Chapter 10), Musiela and Rutkowski (1998, Chapter 11), and Anderson et al. (1996, Chapter 1). We now turn our full attention to the details of the term-structure estimation models.

3.1 The spline-based models

The real challenge in applying the spline methodologies described in section 2 is that we do *not* actually observe zero-coupon rates, forward rates, or the discount function. The spline methodologies we have discussed thus far apply to situations where we are fitting a piecewise polynomial to a set of known function values; what is unknown in this setting is the intermediate values. In our situation, we do not actually observe even these function values. Instead, we observe the set of coupon bond prices that are traded in the bond market at a given point in time. In the following discussion, we will be considering a technique for fitting a cubic polynomial such that the resulting discount function fits these observed prices with minimal error. Minimal error in this context will mean minimizing a weighted sum of squared deviations of the resulting theoretical prices from actual observed prices. The first step, therefore, is to introduce the necessary notation

for us to write down the bond price equation. We define,

$$\begin{aligned} P_i &\triangleq \text{price of } i\text{th bond}, \\ c_{ij} &\triangleq \text{the } j\text{th payment of the } i\text{th bond}, \\ \tau_{ij} &\triangleq \text{the time when the } j\text{th payment of the } i\text{th bond occurs}, \\ m_i &\triangleq \text{remaining number of payments for the } i\text{th bond}. \end{aligned} \tag{121}$$

Armed with these definitions, the price of a coupon bond is merely the discounted sum of its cash flows or,

$$\begin{aligned} P_i &= \sum_{j=1}^{m_i} c_{ij} d(\tau_{ij}), \\ &= c_i^T \tilde{d}(\tau_i), \end{aligned} \tag{122}$$

where,

$$\begin{aligned} c_i &= \begin{bmatrix} c_{i1} & \cdots & c_{im_i} \end{bmatrix}^T, \\ \tilde{d}(\tau_i) &= \begin{bmatrix} d(\tau_{i1}) & \cdots & d(\tau_{im_i}) \end{bmatrix}^T, \end{aligned} \tag{123}$$

and,

$$\tau_i = \begin{bmatrix} \tau_{i1} & \cdots & \tau_{im_i} \end{bmatrix}. \tag{124}$$

We intend to use the B-spline basis to fit our cubic splines to the bond price data. We define our knot sequence as,

$$\{s_k, k = 1, ..., K : 0 = s_1 < s_2 < \cdots < s_{K-1} < s_K = M\}, \tag{125}$$

and the augmented knot sequence required for our B-spline basis as,

$$\{d_k, k = 1, ..., K + 6 : 0 = d_1 = d_2 = d_3 = s_1 < s_2 < \cdots < d_{K+4} = d_{K+5} = d_{K+6=} s_K = T\}. \tag{126}$$

We will use the B-spline basis as defined in equations (88) and (94). In total, we have $\kappa = K + 2$ B-splines defined over the interval $[0, T]$ with the augmented knot sequence described in equation (126). We can write any cubic spline as,

$$B(t)\theta, \tag{127}$$

for $t \in [0, T]$ where,

$$\theta = \begin{bmatrix} \theta_1 & \cdots & \theta_\kappa \end{bmatrix}^T, \tag{128}$$

and

$$B(t) = \begin{bmatrix} B_1(t) & \cdots & B_\kappa(t) \end{bmatrix}. \tag{129}$$

In this form, $B : \mathbb{R} \rightarrow \mathbb{R}^\kappa$ or, in words, B maps a scalar value t into a vector of κ B-spline values.[16] We will, however, require a slightly expanded notation to accommodate the following model construction. We define, therefore, the mapping $\tilde{B}_k : \mathbb{R}^{m_i} \rightarrow \mathbb{R}^{m_i}$ for $k = 1, ..., \kappa$ where,

$$\tilde{B}_k(\tau_i) = \begin{bmatrix} \tilde{B}_k(\tau_{i1}) & \cdots & \tilde{B}_k(\tau_{im_i}) \end{bmatrix}^T. \tag{130}$$

We then generalize equation (129) for all $k = 1, ..., \kappa$ in the following matrix,

$$\begin{aligned} \tilde{B}(\tau_i) &= \begin{bmatrix} \tilde{B}_1(\tau_{i1}) & \cdots & \tilde{B}_\kappa(\tau_{i1}) \\ \vdots & \ddots & \vdots \\ \tilde{B}_1(\tau_{im_i}) & \cdots & \tilde{B}_\kappa(\tau_{im_i}) \end{bmatrix}, \\ &= \begin{bmatrix} \tilde{B}_1(\tau_i) & \cdots & \tilde{B}_\kappa(\tau_i) \end{bmatrix}. \end{aligned} \tag{131}$$

Thus, $\tilde{B}(\tau_i)$ is a mapping such that $\tilde{B} : \mathbb{R}^{m_i} \rightarrow \mathbb{R}^{m_i \times \kappa}$. These definitions will prove useful.

How, then, do we actually fit a spline to the term structure of interest rates? We begin with an arbitrary function of the term structure, denoted as

$$h(t), \tag{132}$$

for all $t \in [0, T]$ where there exists a function, g, such that

$$g(h(\cdot), t) \equiv d(t). \tag{133}$$

This level of generality is introduced because we have some choice as to what part of the term structure we wish to fit. We could, for example, fit the discount function, the zero-coupon curve, the forward curve, or indeed any arbitrary function of the term structure such that equation (133) holds. Consider the simplest case, where we are fitting the discount function directly. This would imply that g is, in fact, the identity function.

The next step is to provide a specific form for $h(t)$. Not surprisingly, we will use a cubic spline to parameterize this function as,

$$\begin{aligned} h(t, \theta) &= \sum_{k=1}^{\kappa} \theta_k B_k(t), \\ &= B(t)\theta. \end{aligned} \tag{134}$$

[16] Recall, of course, that for any $t \in [d_k, d_{k+1}]$, only four of these B-splines are non-zero.

This permits us to rewrite equation (133) as,

$$g(h(\cdot,\theta),t) \equiv d(t,\theta). \tag{135}$$

We now return to our bond price formula, summarized in equation (122), and write out the form of the theoretical bond price using our parameterized function, g. We denote this value as $\hat{P}_i(\theta)$ and give it the following form,

$$\begin{aligned}
\hat{P}_i(\theta) &= \sum_{j=1}^{m_i} c_{ij} d(\tau_{ij},\theta), \\
&= \sum_{j=1}^{m_i} c_{ij} \underbrace{g(h(\cdot,\theta),\tau_{ij})}_{\text{Equation (134)}}, \\
&= \sum_{j=1}^{m_i} c_{ij} g(\underbrace{B(\cdot)\theta}_{\substack{\text{Equation} \\ (134)}}, \tau_{ij}), \\
&= c_i^T \tilde{g}(\tilde{B}(\cdot)\theta,\tau_i),
\end{aligned} \tag{136}$$

where,

$$\tilde{g}(\tilde{B}(\cdot)\theta,\tau_i) = \begin{bmatrix} g(\tilde{B}(\cdot)\theta,\tau_{i1}) & \cdots & g(\tilde{B}(\cdot)\theta,\tau_{im_i}) \end{bmatrix}^T. \tag{137}$$

We are now in a position where, given the appropriate form of g, we can construct a vector of theoretical prices for a given parameterization of our cubic B-spline basis. In general, we will observe N bond prices in the market with a set of prices described by the vector, P. We let

$$\hat{P}(\theta) = \begin{bmatrix} \hat{p}_1(\theta) & \cdots & \hat{p}_N(\theta), \end{bmatrix}^T, \tag{138}$$

be a vector of theoretical prices for the set of N bond observations. Our objective, therefore, is to solve the usual minimization problem,

$$\min_{\theta} \left((P - \hat{P}(\theta))^T W (P - \hat{P}(\theta)) \right), \tag{139}$$

where W is an $N \times N$ weighting matrix.[17] In a manner analogous to the discussion in section 2.5, the resulting $h(t,\theta^*)$ will be the regression spline for the term structure of interest rates. In general, however, the solution to equation (139) is not given by the linear least-squares estimator. Instead, for choices of g that are non-linear in θ this is a non-linear least-squares problem. One could always use a non-linear optimization algorithm.[18] Fisher, Nychka, and Zervos (1994) indicate that it is possible to use a linear first-order Taylor series approximation to solve this problem iteratively. The algorithm proceeds in the following sequence of steps:

[17] The weighting matrix, W, will be discussed further in section 3.2.1.

[18] There are a number of gradient-based hill-climbing algorithms, for example, that might be used.

Step 1: Compute the Taylor series approximation as,

$$\hat{P}(\theta) \approx \hat{P}(\theta^0) - (\theta - \theta^0)X(\theta^0), \tag{140}$$

where,

$$X(\theta^0) \triangleq \left.\frac{\partial \hat{P}(\theta)}{\partial \theta^T}\right|_{\theta=\theta^0}. \tag{141}$$

Step 2: Define the following quantity,

$$Y(\theta^0) = P - \hat{P}(\theta^0) + \theta^0 X(\theta^0), \tag{142}$$

Step 3: Solve the linear least-squares approximation to our original problem given as,

$$\min_{\theta} \left((Y(\theta^0) - \theta X(\theta^0))^T W (Y(\theta^0) - \theta X(\theta^0)) \right), \tag{143}$$

which is solved by,

$$\theta^1 = (X(\theta^0)^T W X(\theta^0))^{-1} X(\theta^0)^T W Y(\theta^0). \tag{144}$$

Step 4: We then iterate to convergence using something similar to this piece of pseudocode:

```
while( criterion > ϵ ) {
        θ^i = (X(θ^{i-1})^T W X(θ^{i-1}))^{-1} X(θ^{i-1})^T W Y(θ^{i-1});
        i=i+1;
        criterion=|| θ^i − θ^{i-1} ||;
};
```

Presumably, the convergence of this algorithm to the true solution of the non-linear least-squares problem, θ^*, will depend on how successfully the linear approximation does its job. One relatively easy, albeit heuristic, way to check this result is to compare and contrast the results of this algorithm with a standard hill-climbing algorithm. We did, indeed, try this and found that the suggested iterative technique works quite well and is, in fact, more stable than the non-linear optimizers employed for this task. As a final note, Fisher, Nychka, and Zervos (1994) indicate that the speed of this approach depends on the selection of reasonable starting values for the iterative algorithm. We used the assumption of a linear zero-coupon curve from 3 per cent to 6 per cent, or the equivalent for the forward curve and the discount curve. We found, similar to Fisher, Nychka, and Zervos (1994), that this was quite effective.

We have seen generally the approach to solve the problem, but how does one select $h(t)$? In their paper, Fisher, Nychka, and Zervos (1994) suggest *three* possible choices, which we will treat as three separate

term-structure estimation models. In the following three subsections, therefore, we will consider each Fisher, Nychka, and Zervos (FNZ) model in turn and derive the necessary quantities for the construction of the previously described minimization problem.

3.1.1 The McCulloch and FNZ-Discount models

The following three subsections will be quite repetitive, but we feel they are necessary to provide the requisite model details. The first choice is to set $h(t)$ equal to the discount function. That is, set $h(t) = d(t)$ for all $t \in [0, T]$. As previously discussed, this is the trivial case where g is the identity function, or rather,

$$\begin{aligned} g(h(\cdot), t) &= g(d(\cdot), t), \\ &= d(t). \end{aligned} \tag{145}$$

The next step is to determine what this implies for the form of the bond price function, $\hat{P}(\theta)$. The result follows from equation (136) and the underlying manipulation,

$$\begin{aligned} \hat{P}_i(\theta) &= \sum_{j=1}^{m_i} c_{ij} g(h(\cdot), \tau_{ij}), \\ &= \sum_{j=1}^{m_i} c_{ij} d(\tau_{ij}), \\ &= \sum_{j=1}^{m_i} c_{ij} B(\tau_{ij})\theta, \\ &= c_i^T \tilde{B}(\tau_i)\theta. \end{aligned} \tag{146}$$

This expression aids in the calculation of $X(\theta^0)$ required for the optimization algorithm. A key input is the partial derivative of the price function with respect to θ^T. It is given as,

$$\begin{aligned} \frac{\partial \hat{P}_i(\theta)}{\partial \theta^T} &= \frac{\partial}{\partial \theta^T}\left(c_i^T \tilde{B}(\tau_i)\theta\right), \\ &= c_i^T \tilde{B}(\tau_i), \end{aligned} \tag{147}$$

for $i = 1, ..., N$. Observe that this partial derivative is independent of the parameter vector, θ. As such, it is a constant and will not influence the results of the optimization problem. If we denote $X \equiv X(\theta^0)$, then we may minimize the linear problem,

$$\min_{\theta} (P - X\theta)^T W (P - X\theta), \tag{148}$$

with the usual solution,

$$\theta^* = (X^T W X)^{-1} X^T W P. \tag{149}$$

This is the original regression spline solution to the term structure suggested by McCulloch (1971). Indeed, the FNZ-Discount and McCulloch models are identical except for the inclusion of the smoothing term in the objective function. As it is similar for all of the Fisher, Nychka, and Zervos (1994) models, we will discuss the smoothing in the final part of this section.

3.1.2 The FNZ-Zero model

The second choice is to set $h(t)$ equal to a slight transformation of the zero-coupon rate. In particular, we set $h(t) = tz(t)$ for all $t \in [0, T]$. Recall that, using the definition in equation (115), we can write g as,

$$\begin{aligned} d(t) &= e^{-tz(t)} \\ &= e^{-h(t)} \\ &= g(h(\cdot), t). \end{aligned} \tag{150}$$

The computation of the theoretical price vector, $\hat{P}(\theta)$, follows from equation (136),

$$\begin{aligned} \hat{P}_i(\theta) &= \sum_{j=1}^{m_i} c_{ij} g(h(\cdot), \tau_{ij}), \\ &= \sum_{j=1}^{m_i} c_{ij} e^{-h(\tau_{ij})}, \\ &= \sum_{j=1}^{m_i} c_{ij} e^{-B(\tau_{ij})\theta}, \\ &= c_i^T e^{-\tilde{B}(\tau_i)\theta}. \end{aligned} \tag{151}$$

The partial derivative with respect to θ^T, needed for the computation of $X(\theta^0)$, is given as,

$$\begin{aligned} \frac{\partial \hat{P}_i(\theta)}{\partial \theta^T} &= \frac{\partial}{\partial \theta^T} \left(c_i^T e^{-\tilde{B}(\tau_i)\theta} \right), \\ &= c_i^T e^{-\tilde{B}(\tau_i)\theta} c_i^T \tilde{B}(\tau_i), \\ &= P_i(\theta) c_i^T \tilde{B}(\tau_i). \end{aligned} \tag{152}$$

This is a $1 \times \kappa$ vector of partial derivatives. Therefore, $X(\theta^0)$ is an $N \times \kappa$ matrix. That is, we have

$$X(\theta^0) = \begin{bmatrix} p_1(\theta) c_i^T \tilde{B}(\tau_1) \\ \vdots \\ p_N(\theta) c_N^T \tilde{B}(\tau_N) \end{bmatrix}. \tag{153}$$

Using equation (153), therefore, we employ the previously described optimization algorithm. Clearly, in this instance, the objective function is non-linear in the parameter vector and is solved using the previously outlined iterative algorithm.

3.1.3 The FNZ-Forward model

The final choice is to set $h(t)$ to the instantaneous forward rate,

$$h(t) = \frac{\partial}{\partial t}(-\ln d(t)), \tag{154}$$

for all $t \in [0, T]$. Now, using the definition in equation (117), we have that g can be written as,

$$\begin{aligned} f(t) = \frac{\partial}{\partial t}(-\ln d(t)) &= h(t) \\ -\ln d(t) &= \int_0^t h(u)du \\ d(t) &= e^{-\int_0^t h(u)du} = g(h(\cdot), t). \end{aligned} \tag{155}$$

This helps us to compute the bond price function. Again, the result follows from equation (136),

$$\begin{aligned} \hat{P}_i(\theta) &= \sum_{j=1}^{m_i} c_{ij} g(h(\cdot), \tau_{ij}), \\ &= \sum_{j=1}^{m_i} c_{ij} e^{-\int_0^{\tau_{ij}} h(u)du}, \\ &= \sum_{j=1}^{m_i} c_{ij} e^{-\int_0^{\tau_{ij}} B(u)\theta du}. \end{aligned} \tag{156}$$

We now define,

$$\begin{aligned} \beta(t) &= \left[\int_0^t B_1(u)du \quad \cdots \quad \int_0^t B_\kappa(u)\right], \\ &= \int_0^t B(u)du. \end{aligned} \tag{157}$$

This definition permits us to write equation (156) in a more convenient form as,

$$\begin{aligned} \hat{P}_i(\theta) &= \sum_{j=1}^{m_i} c_{ij} e^{-\beta(\tau_{ij})\theta}, \\ &= c_i^T e^{-\tilde{\beta}(\tau_i)\theta}, \end{aligned} \tag{158}$$

where,

$$\tilde{\beta}(\tau_i) = \begin{bmatrix} \int_0^{\tau_{i1}} B_1(u)du & \cdots & \int_0^{\tau_{i1}} B_\kappa(u) \\ \vdots & \ddots & \vdots \\ \int_0^{\tau_{im_i}} B_1(u)du & \cdots & \int_0^{\tau_{im_i}} B_\kappa(u) \end{bmatrix}. \tag{159}$$

Finally, we calculate $X(\theta^0)$. The necessary partial derivative of the price function with respect to θ^T is given as,

$$\begin{aligned}\frac{\partial \hat{P}_i(\theta)}{\partial \theta^T} &= \frac{\partial}{\partial \theta^T}\left(c_i^T e^{-\tilde{\beta}(\tau_i)\theta}\right), \\ &= c_i^T e^{-\tilde{\beta}(\tau_i)\theta} c_i^T \tilde{\beta}(\tau_i), \\ &= P_i(\theta) c_i^T \tilde{\beta}(\tau_i).\end{aligned} \tag{160}$$

Observe here, as before, that this is a $1 \times \kappa$ vector of partial derivatives. $X(\theta^0)$, therefore, is an $N \times \kappa$ matrix. That is, we have

$$X(\theta^0) = \begin{bmatrix} p_1(\theta) c_i^T \tilde{\beta}(\tau_1) \\ \cdots \\ p_N(\theta) c_N^T \tilde{\beta}(\tau_N) \end{bmatrix}. \tag{161}$$

Using equation (161), therefore, we employ the previously described optimization algorithm.

The actual solution for the instantaneous forward rate specification of $h(t)$, compared with the zero-coupon rate, is identical, except that we replace the usual B-spline basis with the integrated B-spline basis. This raises the question, of course, of how to integrate a B-spline basis. Dierckx (1993, page 9) provides this rather ugly recursion formula for the specification of an integrated B-spline,

$$\int_{k_i}^{x} B_{i,n}(u)du = \begin{cases} 0 : x \in (\infty, k_i] \\ \frac{k_{i+n}-k_i}{n} \sum_{j=0}^{n-1} \frac{x-k_{i+j}}{k_{i+n}-k_{i+j}} B_{i+j,n-j}(x) : x \in (k_i, k_{i+n}) \\ \frac{k_{i+n}-k_i}{n} : x \in [k_{i+n}, \infty] \end{cases}. \tag{162}$$

In fact, equation (162) computes the integral from k_1 to x. We require, however, the integral from 0 to x. Thus, we need to use equation (162) and the following computation to evaluate an integral on an arbitrary interval $[c, d]$ where $k_1 \leq c < d \leq T$,

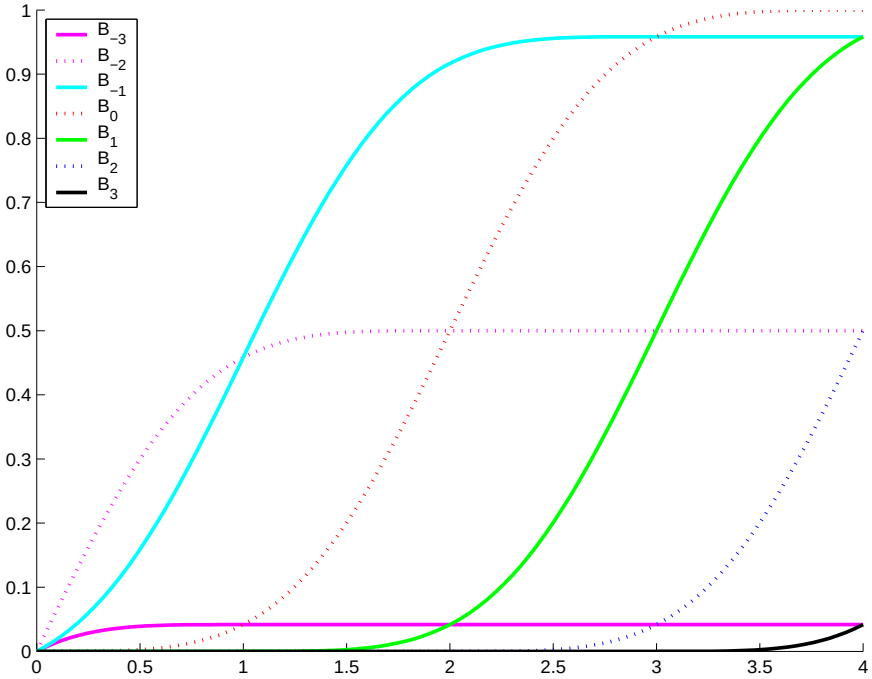
$$\int_c^d B_{i,n}(u)du = \int_{k_i}^d B_{i,n}(u)du - \int_{k_i}^c B_{i,n}(u)du. \tag{163}$$

Section A.5 of the appendix describes a sample computer program that can be used to perform this calculation. Figure 7 shows seven integrated B-splines over the interval $[0, 4]$.

3.1.4 Some common details

Now that we have worked through most of the details of the Fisher, Nychka, and Zervos (1994) model, there is one remaining detail that we need to address. In particular, the Fisher, Nychka, and Zervos (1994) approach uses the *smoothing spline* methodology introduced in section 2.6. This involves the introduction of a penalty function to impose additional smoothness into the specific curve being estimated. What this

Figure 7: **Integrated B-splines on** $[0, 4]$: This figure plots the integrals of seven normalized B-splines defined over the interval $[0, 4]$. These correspond to the B-splines illustrated in Figure 3.



means technically is that we need to restate the minimization problem, as first stated in equation (139), as the following:

$$\min_{\theta} \left(\underbrace{(P - \hat{P}(\theta))^T W (P - \hat{P}(\theta))}_{\text{Original problem}} + \underbrace{\int_0^T \lambda(t) \left(\frac{\partial^2}{\partial t^2} h(t, \theta) \right)^2 dt}_{\text{Penalty function}} \right), \tag{164}$$

where h is our usual arbitrary function of the term structure and $\lambda(t)$ is a function of time that determines the importance of the penalty function in the overall minimization problem. The first question that arises is how to actually compute the penalty function. The answer is that we need to work directly with the B-spline basis to compute the requisite second derivatives and perform the necessary integration. The good news is that although it appears daunting, the penalty function depends only on the B-spline basis and is consequently completely determined by the choice of the knot sequence. Let's examine the penalty in more

detail using the definition in equation (134),[19]

$$\begin{aligned}\int_0^T \left(\frac{\partial^2}{\partial t^2}h(t,\theta)\right)^2 dt &= \int_0^T \left(\frac{\partial^2}{\partial t^2}\sum_{k=1}^{\kappa}\theta_k B_k(t)\right)^2 dt, \\ &= \int_0^T \left(\frac{\partial^2}{\partial t^2}B(t)\theta\right)^2 dt, \\ &= \int_0^T \theta^T B''(t)B''(t)^T\theta dt, \\ &= \theta^T\left(\int_0^T B''(t)B''(t)^T dt\right)\theta.\end{aligned} \tag{165}$$

This implies that we need to compute the second derivatives of our B-spline basis. This is accomplished with a straightforward generalization of equation (107). Some example MATLAB code, for both first and second derivatives of the B-spline basis, is outlined for this purpose in section A.6 of the appendix. This permits us to compute the matrix of squared second derivatives at each of the points in the knot sequence.

The next task, of course, is to compute the integral of this matrix of squared second derivatives. In fact, what we are doing here is computing the inner product of two arbitrary B-splines. Recall that the derivative of a fourth-order B-spline is actually a third-order B-spline. It follows, therefore, that the second derivative of a fourth-order B-spline is a second-order B-spline. Thus, the individual non-zero terms in the matrix $B''(t)B''(t)^T$, for $t = k_{-3}, ..., k_{N-1}$ have the form,

$$\int_0^T B_{j,2}(t)B_{k,2}(t)dt, \tag{166}$$

for $j, k = -3, ..., N-1$, which is the inner product of these two B-spline basis functions. Analytical solutions for equation (166) exist that are based on Gauss quadrature, but it is just as easy to perform this computation numerically. That is, we compute the second derivatives using the general formula and then numerically integrate this function over the interval $[0, T]$. In section 2.6, we saw another approach to this smoothing function based on the use of a second difference operator in matrix form. This was based on the approach suggested by Eilers and Marx (1996). We were tempted to use this approach in this paper, but felt that for a formal comparison of these models, it was best not to make any substantial changes. Some limited testing with this form showed that the Eilers and Marx (1996) approach is quite convenient. We would argue that this could prove a reasonable addition to the model that would be quite likely to speed the computation time and ease the implementation associated with these models in a significant manner.

How do we deal with the function $\lambda(t)$, which determines the relative importance of the penalty function? Fisher, Nychka, and Zervos (1994) suggest the use of a constant value λ selected each day using a technique termed generalized cross-validation. Waggoner (1997), however, first suggested the idea of introducing a

[19] For the moment, however, we will ignore the function $\lambda(t)$.

penalty as a function of term to maturity; he found that a piecewise linear specification for $\lambda(t)$ worked well. The resulting approach is termed the *variable roughness penalty* technique. Anderson and Sleath (2001) extended this idea by providing a continuous form for the function, $\lambda(t)$. We have adopted this latter approach and—based on trial and error—found two specifications for $\lambda(t)$ that seemed to work quite well for the Canadian market.[20] The primary reason for following these latter papers was that we felt it made more intuitive sense to impose greater smoothness at the long end of the term structure than at the short end.

In particular, we chose

$$\lambda(t) = \frac{\beta_0}{1 + \beta_1 e^{-\beta_2 t}}, \tag{168}$$

and,

$$\lambda(t) = \beta_0 \ln(t+1), \tag{169}$$

where in both cases $\beta_0 = 5000$ and in the first case $\beta_1 = 10$ and $\beta_2 = 0.2$. Figure 8 outlines these two functions. We found that the log-based approach was most effective with the application of the B-spline basis to the forward curve, while we used the logistic form in equation (168) for the discount and zero-coupon curve specifications. The reason for this difference was that we achieved better results by reducing the smoothing at the short end and increasing it at the long end for the zero-coupon and discount curves—hence the use of the logistic form, which inflects at approximately ten years. The forward curve needed a steady increase in smoothness over the entire 30-year term-to-maturity spectrum.

Some additional details require mention. The first relates to the number of knot points selected in our analysis. For the FNZ-Discount, FNZ-Zero, and FNZ-Forward models, we use 20 knot points over the interval $[0, T]$. For the McCulloch approach, which has $\lambda(t) \equiv 0$, we use only six knot points. The reason for this distinction is that the smoothness algorithm reduces the effective number of parameters and allows a greater number of knot points to be considered. This is one of the benefits of the Fisher, Nychka, and Zervos (1994) approach. With the McCulloch model, however, a selection of more than six knot points leads us dangerously close to a singular matrix for use in the solution of the least-squares problem. This can lead to potential numerical errors, so we have opted to retain a relatively low number of knot points.

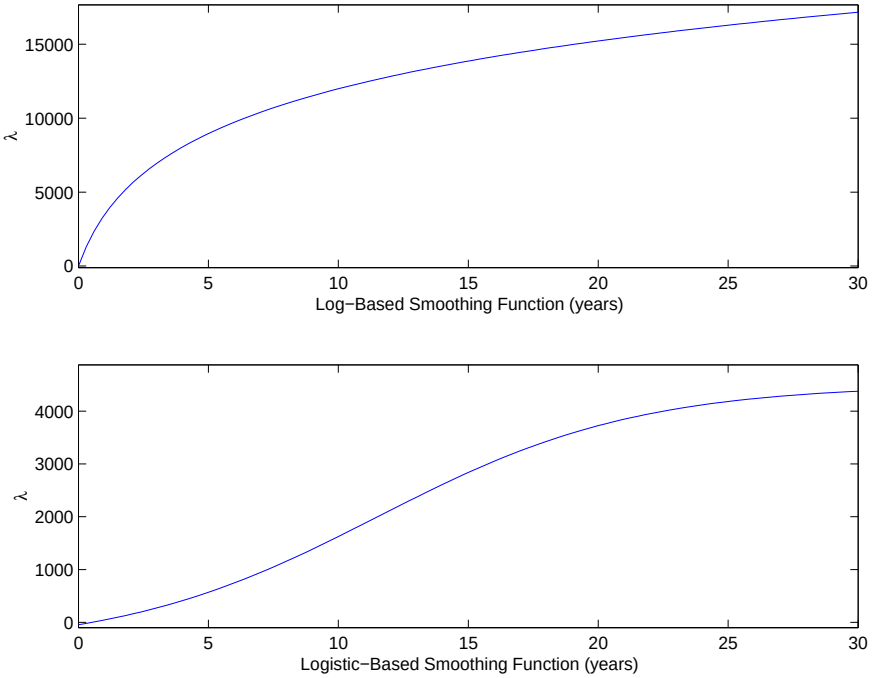
As a final note, it is useful to make this *effective parameter* concept more precise. Fisher, Nychka, and Zervos (1994) provide the following *effective parameter* formula, which we have generalized somewhat for a

[20] Note that the introduction of a variable roughness penalty does not complicate the computation of the penalty function. Indeed, the form of the individual non-zero terms in the matrix $B''(t)B''(t)^T$, for $t = k_{-3}, ..., k_{N-1}$ is,

$$\int_0^T \lambda(t) B_{j,2}(t) B_{k,2}(t) dt, \tag{167}$$

for $j, k = -3, ..., N-1$. We deal with this in the same manner as equation (166).

Figure 8: **Variable Roughness Penalty Functions**: This figure plots the two choices of variable roughness penalty functions in this paper—outlined in equations (168) and (169). This concept was introduced by Waggoner (1997) and extended by Anderson and Sleath (2001).



time-varying penalty parameter, λ_t,

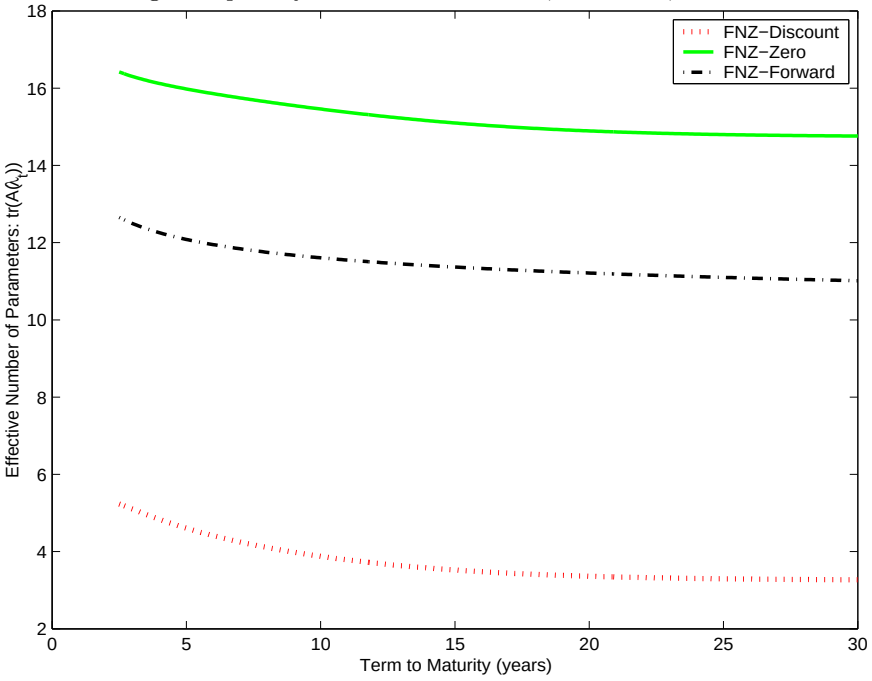
$$A(\lambda_t) = \text{trace}\left[X_{\theta^*(\lambda_t)}\left(X^T_{\theta^*(\lambda_t)}X_{\theta^*(\lambda_t)} + \int_0^T \lambda_t B''(t)B''(t)^T dt\right)^{-1} X^T_{\theta^*(\lambda_t)}\right], \tag{170}$$

where $X_{\theta^*(\lambda_t)}$ is as defined in equation (141) evaluated at the optimal parameter value for a given λ_t. The idea is that we should fix t and compute equation (170) for a number of values $t \in [0, T]$. We demonstrate this function—given our choice of variable roughness penalty—for the FNZ-Discount, FNZ-Zero, and FNZ-Forward models in Figure 9.

3.2 The function-based models

We now examine a different class of models that we term function-based. These models do not use piecewise cubic polynomials or splines, as was the case in the previous section, but instead use single-piece functions defined over the entire term-to-maturity domain. Beyond this difference in the choice of functions, the general approach is quite similar. Specifically, the parameters of these models are determined through the

Figure 9: **Effective Number of Parameters**: This figure plots the effective number of parameters associated with the choices of variable roughness penalty for the FNZ-Discount, FNZ-Zero, and FNZ-Forward models.



minimization of the squared deviations of theoretical prices from observed prices, and use is made of various basis functions. In the following examination, each of the four function-based models is discussed in turn.

3.2.1 The MLES model

In the Merrill Lynch exponential spline (MLES) model, as introduced in Li et al. (2001), the theoretical discount function $d(t)$ is modelled as a linear combination of exponential basis functions. This model does not actually involve splines at all, in the sense discussed elsewhere in this paper. Li et al. (2001) refer to modelling the discount function as a *single-piece* exponential spline, which is equivalent to simply fitting a curve on a single interval. The form of the discount function is given as,

$$d(t) = \sum_{k=1}^{D} \zeta_k \, e^{-k\alpha t}. \tag{171}$$

In other words, instead of using a linear combination of the B-spline basis, as used by Fisher, Nychka, and Zervos (1994), the MLES model employs a linear combination of exponentials. The ζ_k are unknown parameters for $k = 1, \ldots, N$ that must be estimated. The parameter α, while also an unknown parameter,

is interpretable as the long-term instantaneous forward interest rate. Notice that,

$$\sum_{k=1}^{N} \zeta_k = d(0) = 1, \tag{172}$$

which effectively reduces the number of unknown parameters by one. We choose the number of basis functions to be $N = 9$. To get a more accurate fit, a higher number of basis functions is desirable; however, for values of N higher than 9, there is not a substantial improvement in the residual error. Moreover, as N increases, the matrices used in the computations are more likely to become poorly conditioned, thereby potentially leading to unreliable numerical results.

For notational convenience, let us denote the basis functions as $f_k(t) = e^{-k\alpha t}$. It is reasonable to inquire as to why one would select this form for the basis functions. In fact, there is complete flexibility in the choice of the basis functions, f_k. That is, the MLES methodology can be used to model the discount function as a linear combination of any functions we might find interesting. Why, then, should we use these particular exponential functions? There are at least two reasons. First, there is good economic intuition to indicate that exponentials are strongly related to the discount function. This was first pointed out in Vasicek and Fong (1981). To see this, consider a hypothetical setting where interest rates are constant, say α_0. Note that if one interest rate, say the instantaneous forward rate, is constant, then all other types of interest rates will be the same constant. The true discount function is thus simply $d(t) = e^{-\alpha_0 t}$, which agrees with the theoretical linear combination above, taking $\zeta_1 = 1$ and $\zeta_k = 0$ for $k \geq 2$. Second, the parameter α appearing in the f_k represents a long-term instantaneous forward rate. Indeed,

$$\lim_{t\to\infty} \frac{d(t)}{\zeta_1 e^{-\alpha t}} = \lim_{t\to\infty} \left(1 + \sum_{k=2}^{D} \frac{\zeta_k}{\zeta_1} e^{-(k-1)\alpha t}\right) = 1. \tag{173}$$

When the limit of the ratio above is 1, we say that $d(t)$ is *asymptotic* to $\zeta_1 e^{-\alpha t}$, usually written $d(t) \sim \zeta_1 e^{-\alpha t}$. Therefore, for large values of t the discount function is approximately given by $e^{-\alpha t}$.[21] These two reasons, therefore, suggest that the use of an exponential basis has some theoretical appeal. Figure 10 shows a graph of the first three negative exponentials used in this approach.

When we think about a basis for a given vector space, we often think about the concept of orthogonality. Li et al. (2001) suggest that the $\{f_k\}$ basis be converted into an orthogonal basis $\{e_k\}$ under the inner product,

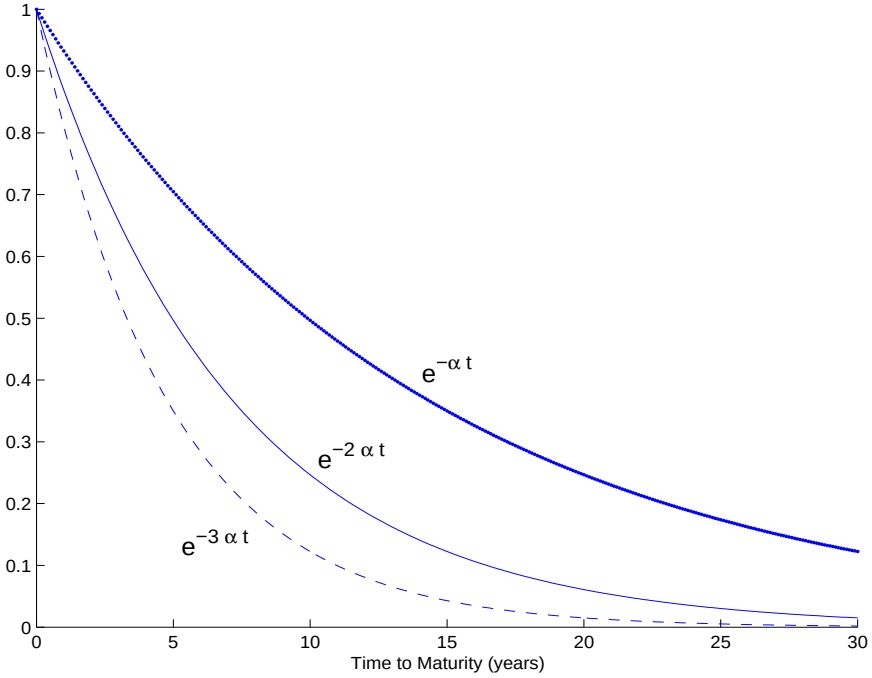
$$\langle g, h\rangle = \int_0^\infty g(t)h(t)\,dt, \tag{174}$$

using the Gram-Schmidt orthogonalization process.[22] Then, the discount function can be written in the

[21] If we interpret $\zeta_1 e^{-\alpha t}$ as a discount function, we must have $\zeta_1 = 1$, because $d(0) = 1$ for any discount function.

[22] While it may seem more natural to use the inner product $\langle g, h\rangle = \int_0^{30} g(t)h(t)\,dt$, integrating on $[0, \infty)$ greatly simplifies the calculations and does not overly influence the results when integrating our basis functions $\{f_k\}$. More to the point, we are free to use any linearly independent functions we like as basis functions.

Figure 10: **Negative Exponential Basis Functions**: This figure outlines the first three negative exponentials used to model the discount function as described in equation (171).



form

$$d(t) = \sum_{k=1}^{D} \lambda_k \, e_k(t). \tag{175}$$

Since each e_k is just a linear combination of the f_k, it follows that the new parameters λ_k are linear combinations of the previous parameters ζ_k. This simply amounts to reparameterizing the problem, and it is sufficient for our purposes to use the ζ_k parameterization (i.e., the $\{f_k\}$ basis).

Given this theoretical form for the discount function, how then do we compute the associated theoretical bond prices? The theoretical price of bond number j is given by the sum of the discounted values of its cash flows,

$$\hat{P}_i = \sum_{j=1}^{m_i} c_{ij} d(\tau_{ij}), \tag{176}$$

where we recall that m_i denotes the number of cash flows associated with the ith bond. In addition, note that the sum in equation (176) is taken over the coupon maturity dates of bond i, and c_{ij} is the magnitude

of the cash flow at time τ_{ij}. It is convenient to form the matrix H defined by

$$H_{ik} = \sum_{j=1}^{m_i} c_{ij} f_k(\tau_{ij}), \tag{177}$$

where the sum is taken over the cash-flow maturity times. The matrix H is an $N \times D$ matrix, where N is the number of bonds, and D is the number of basis functions. The matrix H depends only on the maturity times and coupon values of the bonds. As such, it need only be computed once daily. The important property of H is that $\hat{P} = HZ$, where $\hat{P}$ is the column vector of theoretical prices and $Z = (\zeta_1, \ldots, \zeta_D)^T$ is the column vector of unknown parameters. For simplicity of notation, we compute the ith entry,

$$\begin{aligned}
(HZ)_i &= (i\text{th row of } H)\, Z, \\
&= \sum_{k=1}^{D} \underbrace{\sum_{j=1}^{m_i} c_{ij} f_k(\tau_{ij})}_{\substack{\text{Equation} \\ (177)}} \zeta_k, \\
&= \sum_{j=1}^{m_i} c_{ij} \left(\sum_{k=1}^{D} \zeta_k f_k(\tau_{ij}) \right), \\
&= \sum_{j=1}^{m_i} c_{ij} d(\tau_{ij}), \\
&= \hat{P}_i.
\end{aligned} \tag{178}$$

The matrix H also emphasizes one of the most attractive features of the MLES method: each theoretical bond price is a *linear* function of the unknown parameters (i.e., $\hat{P} = HZ$). This is really a combination of two instances of linearity. First, (theoretical) bond prices are always a linear function of discount function values where the coefficients are the cash flows. Second, in this particular case, the discount function is modelled as a linear function of the unknown parameters. This is similar to the specification of h as the discount function in the Fisher, Nychka, and Zervos (1994) model. This underscores the general advantage of modelling the discount curve relative to the zero-coupon or forward curves.

The next step is to form a diagonal matrix, W, constructed from weights associated with each bond. We can make many choices for the weights, but the general idea is that higher weights should be placed on bonds that we believe have observed prices that are more accurate estimates of their true prices. The matrix W is a square diagonal matrix of size $N \times N$, with each diagonal entry equal to the corresponding bond weight.

Our choice for the weights was the *reciprocal* of the modified duration. Notice that this places less weight on longer-term, or equivalently higher-duration, bonds. This is because we expect the observed prices for these bonds to exhibit greater variability. Using the reciprocal of modified duration for the weights is also related to the idea that bonds are heteroscedastic in price, according to the modified duration. More

specifically, this particular weighting assumes that the variance of a bond's pricing error is approximately proportional to that bond's modified duration.

The final step is to actually estimate the parameters, $\zeta_1, \ldots, \zeta_D$. We assume that the pricing errors, $\hat{P}_j - P_j$, are normally distributed with a zero mean and a variance proportional to $1/w_j$, where $w_j = W_{jj}$ is the weight assigned to bond j. We wish to find the set of parameters $\zeta_1, \ldots, \zeta_D$ that maximizes the log-likelihood function (ignoring some multiplicative constants) given as,

$$l(\zeta_1, \ldots, \zeta_D) = -\sum_{j=1}^{N} \omega_j (\hat{P}_j - P_j)^2, \tag{179}$$

or equivalently in matrix form,

$$l(Z) = -\|W(HZ - P)\|^2. \tag{180}$$

Since the theoretical prices are linear functions of the unknown parameters, it follows that the maximum likelihood estimate is obtained as the following generalized least-squares (GLS) solution,

$$\hat{Z} = (H^T W H)^{-1} H^T W P. \tag{181}$$

We can also verify directly that this maximizes the log-likelihood if we recall that $\hat{P}_i = HZ$ depends on Z, whereas each P_j is a constant. Consider, therefore,

$$\frac{\partial l(Z)}{\partial \zeta_j} = -2 \sum_{i=1}^{N} \omega_i (\hat{P}_i - P_i) \frac{\partial \hat{P}_i}{\partial \zeta_j} = -2 \sum_{i=1}^{N} \omega_i (\hat{P}_i - P_i) \frac{\partial (HZ)_i}{\partial \zeta_j}. \tag{182}$$

Now, we set equation (182) equal to zero for all j and put the equations into matrix form using the fact that

$$\frac{\partial (HZ)_i}{\partial \zeta_j} = H_{ij}. \tag{183}$$

The following manipulation then yields

$$\begin{aligned}
0 &= (W(\hat{P} - P))^T H \quad \text{(dividing by } -2\text{)}, \\
0 &= H^T (W(\hat{P} - P)) \quad \text{(taking the transpose of both sides)}, \\
H^T W \hat{P} &= H^T W P, \\
H^T W (H\hat{Z}) &= H^T W P, \\
\hat{Z} &= (H^T W H)^{-1} H^T W P,
\end{aligned} \tag{184}$$

which agrees with equation (181).

Once $\hat{Z}$ is computed, we easily get the theoretical prices, using the result from equation (178), by computing $\hat{P} = H\hat{Z}$. Another attractive feature of the MLES model is that there is no numerical optimization

problem to solve—the optimization is handled automatically by the least-squares matrix calculation. This provides a significant advantage to this model in terms of computational speed.

How do we find the remaining parameter, α? The previous discussion provides an interpretation of α as a long-term instantaneous forward rate. The benefit of this interpretation is twofold. If we have some external estimate of α that we consider to be reliable—based, for example, on economic reasoning—then we can use it instead of treating α as an unknown parameter. On the other hand, without knowing anything about α, we can obtain an estimate for it using the Merrill-Lynch methodology. The easiest way to accomplish this is to choose the value of α that minimizes the root-mean squared pricing error (also called residual error), R, given as,

$$R = \frac{1}{\sqrt{N}}\|\hat{P} - P\| = \sqrt{\sum_{j=1}^{N} \frac{(\hat{P}_j - P_j)^2}{N}}. \tag{185}$$

Notice that R may actually be considered to be a function of α. This is just a one-dimensional numerical optimization problem that could be done by any mathematical software package. Li et al. (2001) recommend a range for α of 5 per cent to 9 per cent; however, any *economically reasonable* range for α, depending on the financial market in question, will also work.

3.2.2 The MLES-Fourier model

As previously discussed, while there are some theoretical reasons for use of the negative exponential basis functions, one is by no means restricted to these functions. First, in tests of the orthogonal basis, as described above, we found, much as expected, results identical to the non-orthogonal basis outcomes. Second, we examined the freedom allowed in choosing basis functions, by considering two additional examples. Each approach was motivated by Taylor's Theorem and Fourier analysis, respectively.

We first considered the standard basis,

$$\{1, x, \ldots, x^8\}, \tag{186}$$

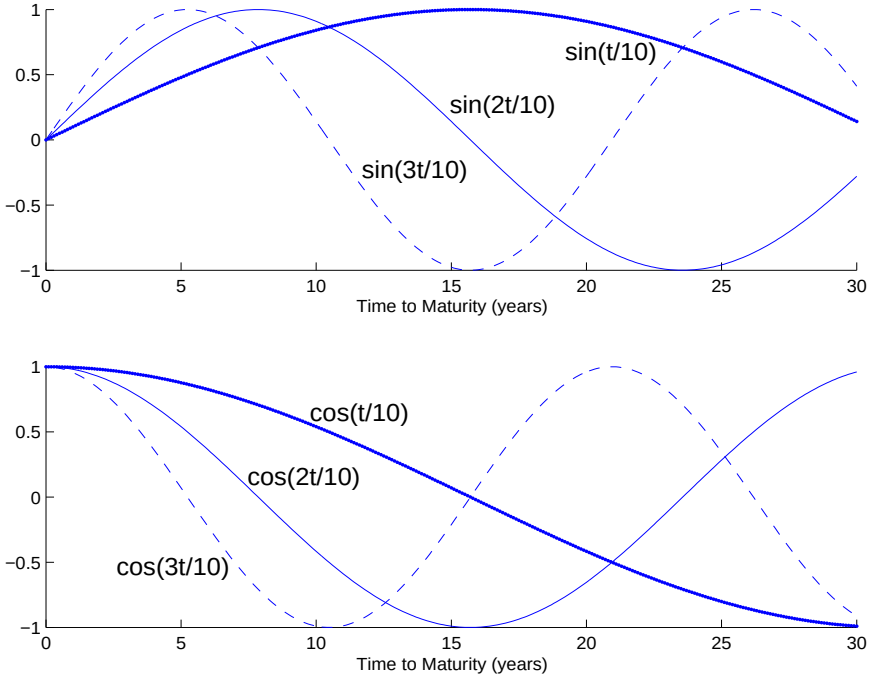
for the space of polynomials, $\mathcal{P}_8$. In some trial-and-error experimentation, we found that it generated reasonable-looking zero-coupon yield curves and pricing errors. There were, however, a number of concerns that the matrix H was poorly conditioned, which may have led to unreliable results. Also, this particular basis is extremely sensitive to the choice of the number of basis functions—here we choose $N = 9$ basis functions, and the results vary quite substantially choosing even $N = 8$ or 10.

We then proceeded to examine the following Fourier-series basis,

$$\left\{1, \sin\left(\frac{nt}{10}\right), \cos\left(\frac{nt}{10}\right); n = 1, 2, 3, 4\right\}. \tag{187}$$

The horizontal-stretch factor of $\frac{1}{10}$ was chosen ad hoc, and is meant to extend the wavelength of each basis function to avoid excessive oscillation. This approach also produced a reasonable-looking zero-coupon yield curve, although the forward curve was somewhat more oscillatory than the standard negative exponential basis. After substantial experimentation, however, we found the MLES-Fourier approach created using the basis functions in equation (187) to be a stable and potentially useful extension of the MLES methodology. Figure 11 shows the first three terms for the Fourier basis; the sine and cosine terms are graphed individually. Observe the flexibility of this functional form.

Figure 11: **Fourier Series Basis Functions**: This figure outlines the first three Fourier terms—both the cosine and sine functions are graphed separately—used to model the discount function as described in equation (171).



3.2.3 The MLES-Benchmark model

In the Government of Canada bond market, at any given point in time, there are normally four bonds outstanding that are considered to be *benchmarks*. These bonds are considered to be the most liquid debt instruments in the market, and the observed prices of these benchmark bonds are considered to be a highly accurate indication of their true prices. We decided, therefore, given their relative importance, to place more weight on the benchmark bonds in our estimation algorithm. After forming the weighting matrix, W, the benchmarks' weights are then multiplied by a constant value, K. With this structure we can choose

K to be whatever we want—the higher the value of K, of course, the more closely the theoretical prices of the benchmark bonds will match their observed prices. For example, a choice of $K = 1$ indicates no special treatment for the benchmarks. In some preliminary testing with actual bond data, we found $K = 30$ represented about a 5 cent pricing error for a notional \$100 bond in the 30-year benchmark while a choice of $K = 350$ represented a 30-year benchmark pricing error of less than one cent. Note, however, that when you impose a tighter fit on a specific subcollection of bonds, you expect slightly larger errors on the remaining ones. This is not necessarily problematic if you are confident with the accuracy of the benchmark prices. Furthermore, this adjusted form of the MLES model can help expose which non-benchmark issues' observed prices are relatively expensive or inexpensive compared with the set of benchmark bonds.

This *preferential weighting* idea can also be easily adapted to any particular subset of bonds besides just the benchmarks. Fractional weights could be used to represent lower confidence in accurate pricing.[23] This benchmark idea can also be extended to incorporate a different approach to the optimization problem. Earlier, we stated that the observed prices of the benchmark bonds are thought to be accurate representations of the benchmark bonds' theoretical true prices. We can reformulate our problem by demanding that the benchmark bond pricing errors must be exactly zero. This formulation is known generally as a *constrained optimization* problem. In this case, the benchmark bond pricing errors being set to zero give rise to the constraints, and the optimization part of the problem is to minimize the pricing errors of the non-benchmarks.[24]

Let's consider the mathematical details of this constrained optimization approach.[25] To set things up, we construct the matrix H_B for the benchmarks-only model, which corresponds to the matrix H in the previous model. That is,

$$(H_B)_{ik} = \sum_{j=1}^{m_i} c_{ij}\, f_k(\tau_{ij}) \tag{188}$$

but now the index i runs only over the *benchmark* bonds. We also form the vector of observed benchmark prices, P_B.

Our constrained optimization problem in matrix form is,

$$\begin{aligned} \text{Min} \quad & \|W(\hat{P} - P)\|^2 \\ \text{s.t.} \quad & H_B Z = P_B. \end{aligned} \tag{189}$$

The second line, which includes the constraints, represents the fact that we want the theoretical benchmark prices to be exactly equal to the observed benchmark prices.

[23] A fractional weight is a strictly positive weighting that is less than unity.

[24] Equivalent results can be obtained by taking an extremely large value of K in our previously described *benchmark weighting* approach. Recall that we can make the benchmark pricing errors arbitrarily close to zero by choosing a sufficiently large value of K.

[25] We would like to thank Michel Krieber of TD Securities for bringing this formulation to our attention.

We now introduce a vector of Lagrange multipliers, denoted γ. The least-squares solution, $\hat{Z}$, to the constrained optimization problem is given by the block matrix equation,

$$\begin{bmatrix} H^T W H & (H_B)^T \\ H_B & 0 \end{bmatrix} \begin{bmatrix} Z \\ \gamma \end{bmatrix} = \begin{bmatrix} H^T W P \\ P_B \end{bmatrix}. \tag{190}$$

Notice that looking only at the $(1,1)$-block of the matrix equation, we get a similar matrix equation to the one that arose in our previous non-constrained least-squares optimization problem. Let L denote the first matrix on the left-hand side. The optimal solution $\hat{Z}$ is obtained by computing

$$\begin{bmatrix} \hat{Z} \\ \gamma \end{bmatrix} = L^{-1} \begin{bmatrix} H^T W P \\ P_B \end{bmatrix}, \tag{191}$$

where the values of the Lagrange multipliers, γ, can be ignored.

3.2.4 The Svensson model

The final model considered in our experiment is the so-called Svensson model. The basic idea for this model originated with Nelson and Siegel (1987), who suggested a parsimonious estimation methodology for the term structure by postulating a relatively simple functional form for the instantaneous forward curve. Svensson (1994) extended this work by altering the functional form of the instantaneous forward curve suggested by Nelson and Siegel (1987).[26] How does this work? It begins with the following, rather straightforward, three-parameter function of time,

$$g(t) = b_0 + b_1 e^{-\frac{t}{a_0}}, \tag{192}$$

for $b_0, b_1, b_2 \in \mathbb{R}$ and $a_0 > 0$. This is, in fact, a simple exponential function. The b_0 parameter essentially anchors g at a given level, while the sign of b_1 determines the slope of the instantaneous forward curve. Figure 12 illustrates some possible parameterizations of g.

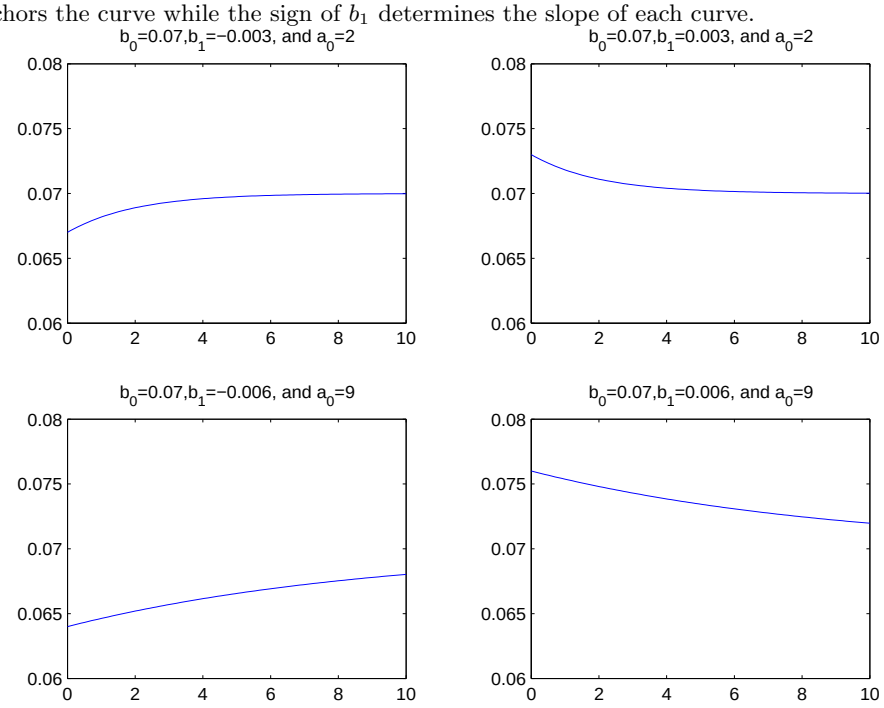
The function g would nevertheless be a rather uneventful model for the forward term structure. What is required now is some additional flexibility to permit the instantaneous forward-rate curve to take a number of different shapes. Consider, therefore, a similar two-parameter function of time,

$$h(t) = \frac{b_2 t}{a_0} e^{-\frac{t}{a_0}}, \tag{193}$$

for $b_2, a_0 \in \mathbb{R}$ with $a_0 > 0$. This is, in fact, a positive or negative *U-shaped* function, depending on the choice of the parameter, b_2. In Figure 13, we illustrate a number of possible combinations of values for b_2 and a_0. Observe that the location of the U-shape is governed by the second parameter, a_0.

[26] Indeed, the Svensson model should probably rightly be termed the extended Nelson and Siegel model.

Figure 12: **An Exponential Function**: This figure illustrates four different parameterizations of equation (192). Note that b_0 anchors the curve while the sign of b_1 determines the slope of each curve.



The Svensson model linearly combines the functions g and h into a single function for the instantaneous forward-rate curve, as follows,

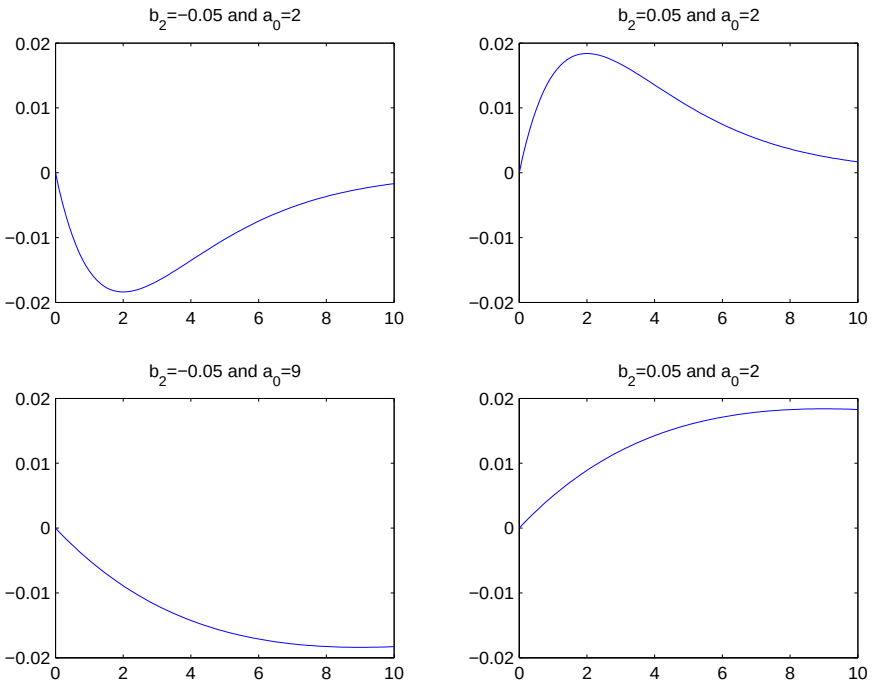
$$f(t) = \underbrace{b_0 + b_1 e^{\frac{-t}{a_0}}}_{\text{Equation (192)}} + \underbrace{\frac{b_2 t}{a_0} e^{\frac{-t}{a_0}}}_{\substack{\text{Equation} \\ (193)}} + \underbrace{\frac{b_3 t}{a_1} e^{\frac{-t}{a_1}}}_{\substack{\text{Equation} \\ (193)}}. \tag{194}$$

Observe that the function h is repeated twice, with different parameters, in this formulation. In the original work (Nelson and Siegel 1987), there was only one incidence of the function h in the specification of the instantaneous forward-rate curve.

The key question in this model is how do we actually determine the parameter set? We have seen in previous sections that, ultimately, we need to transform this forward-rate curve into a discount function to price the set of government coupon bonds. Once we have a theoretical price vector, we can optimize over the parameter set to minimize the distance between the observed prices and the theoretical prices. To transform equation (194) into a discount function, therefore, we use the result from equation (119). Repeated for convenience, we have,

$$z(t) = \frac{\int_0^t f(u)du}{t}. \tag{195}$$

Figure 13: **A Hump-Shaped Function**: This figure illustrates four different parameterizations of equation (192). The sign of parameter b_2 determines the direction of the U-shape, while the parameter a_0 determines its location.



Integrating equation (194) might seem difficult but a simple integration by parts of equation (193) is possible, yielding,

$$\begin{aligned} \int u e^{-\frac{u}{a_0}} du &= u a_0 e^{-\frac{u}{a_0}} - \int a_0 e^{-\frac{u}{a_0}} du, \\ &= u a_0 e^{-\frac{u}{a_0}} + a_0^2 e^{-\frac{u}{a_0}} + C, \\ &= a_0 e^{-\frac{u}{a_0}} (u + a_0) + C, \end{aligned} \tag{196}$$

for $C \in \mathbb{R}$. With this result, equation (195), and some tedious algebra we obtain the zero-coupon curve. This is then subsequently transformed into the discount function (using equation (115)), which is subsequently used to derive a theoretical set of bond prices given the values of the parameters. The optimal parameter set is found using a non-linear optimization algorithm. Note that this problem is very non-linear in the parameters. Bolder and Stréliski (1999) discuss the issues relating to finding reasonable parameter estimates in detail.

4 Results

In this section, we will consider the eight different models described in section 3. These include the four spline-based models: McCulloch and three versions of the Fisher, Nychka, and Zervos (1994) model that fit a *cubic spline* to the discount, zero, and forward curves. We will refer to these models as the FNZ-Discount, FNZ-Zero, and FNZ-Forward approaches, respectively. In addition, we will consider the four function-based models. These include Li et al. (2001) with exponential and Fourier-series bases, which we will call the MLES-Exponential and the MLES-Fourier models. The function-based models include the exponential basis with forced-zero error on the benchmarks, termed the MLES-Benchmark, and the Svensson model. Our objective is to estimate these models over a reasonably large number of dates and assess the relative strengths and weaknesses of each of these approaches. This analysis should help us identify those models that are most useful for our purposes at the Bank of Canada.

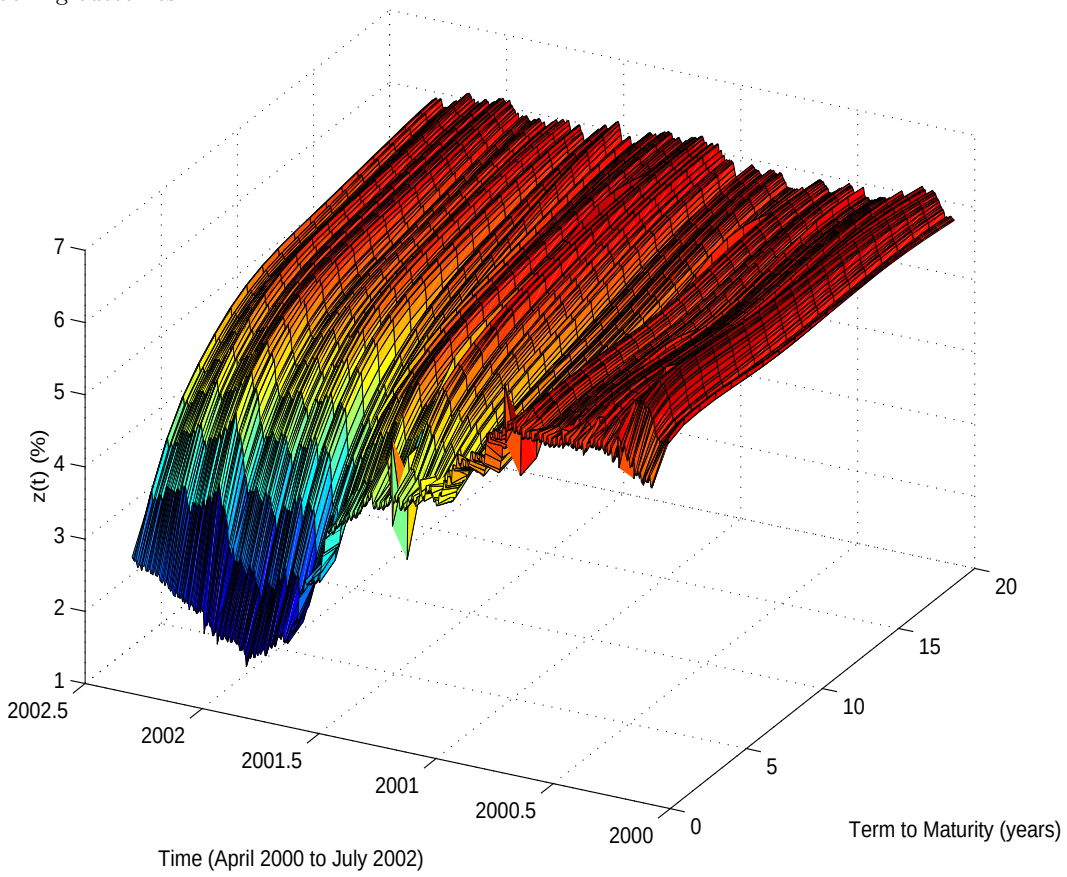
We begin our discussion with a brief review of our data. We used a daily set of bond prices from 1 April 2000 to 11 July 2002. With the elimination of several dates owing to data problems, this implies that we have 561 consecutive dates to perform our empirical examination. On each date, the data include all of the outstanding Government of Canada coupon bonds and five treasury-bill observations with one-month, two-month, three-month, six-month, and one-year maturities. There are an average of approximately 70 bonds and treasury bills in our collection of observations for each individual date. This does not imply, however, that we use all of the bonds in our estimation of these curves. Some bond prices—given relatively low liquidity or special features—are not representative of the general prices of government bond prices in the economy. We must, therefore, filter the set of available bonds to arrive at a smaller set of bonds that is reasonable for our estimation approach.

The *filtering* issue is discussed in detail in Bolder and Stréliski (1999), and using this work we have adopted a similar set of filters. First, we require that a bond have at least Can$500 million outstanding. This provides a minimum amount of liquidity. Second, we exclude any bonds that have more than 500 basis points difference between their market yield and coupon rate. This is essentially a measure of the size of the premium or discount on a given government coupon bond. A filter of this nature is important: investors typically avoid bonds trading at large premiums or discounts because of the relative tax treatment of capital gains versus interest income.[27] Third, we exclude bonds with less than one year remaining to maturity. In general, there was some concern that these bonds are illiquid and thus trade at erratic prices. Finally, we eliminate Real Return bonds and we force inclusion of benchmark bonds and treasury bills. The end result is that, of the approximately 70 bonds that are available on average for each date in the sample, we use an average of 38.7 bonds in the daily estimation algorithm.

Figure 14 graphically illustrates the term structure of zero-coupon interest rates from April 2000 until

[27] In recent years this has become a much less important market distortion.

Figure 14: **Our Data Period**: This figure illustrates the term structure of zero-coupon interest rates from April 2000 until July 2002—these rates were estimated using the FNZ-Zero model, although all models provide quite similar-looking outcomes.



July 2002. These rates were estimated using the FNZ-Zero model, although all models provide quite similar outcomes. Observe that the term structure was quite flat at the beginning of the period, but steepened dramatically throughout the latter part of 2001 and into 2002.

4.1 The first experiment

To assess our eight different models, we require some methodology to measure their relative usefulness. Our first, and perhaps most important, model criterion is the ability of each model to produce theoretical prices that are a close fit to the set of bond prices used to estimate the model. In short, we require a group of measures to assess the *goodness of fit* of our collection of models. The first measure we will use is called the

root-mean-squared error (RMSE) and is defined as,

$$\text{RMSE} = \sum_{i=1}^{N} \sqrt{\frac{\left(\hat{P}_i - P_i\right)^2}{N}}, \tag{197}$$

where N is the total number of bonds used in the estimation, $\hat{P}_i$ is the theoretical price of the ith bond, and P_i is the observed price of the ith bond. This is a useful measure given that, for all the models, we are determining the *optimal* parameter set by minimizing the squared distance of the theoretical prices from the observed prices. The RMSE measure provides us with a *pseudo* average error for the given set of bonds; we apply the square root simply to return it to the original units.[28]

The second measure that we use to gauge the goodness of fit of our eight models is the *mean absolute error* (MAE), which is defined as,

$$\text{MAE} = \sum_{i=1}^{N} \frac{\left|\hat{P}_i - P_i\right|}{N}. \tag{198}$$

MAE is thus the average distance between the theoretical bond prices and the observed set of bonds in absolute value terms. This measure is not as easily influenced by extreme observations as the RMSE measure. As the RMSE squares the distance between observed and theoretical prices, a single large error will have a larger relative contribution to the overal RMSE than with the MAE. As such, these are two complementary measures.

Since we will be using the RMSE and MAE measures quite heavily, let's take a moment to examine these quantities and their relationship to each other. Let $e = (e_1, \ldots, e_N)$ be a vector in $\mathbb{R}^N$; we can think of e as being a vector of bond pricing errors (i.e., $e = \hat{P} - P$). In this case, N represents the number of bonds. Define the following quantities,

$$\|e\|_1 = \sum_{i=1}^{N} |e_i|, \tag{199}$$

and,

$$\|e\|_2 = \sqrt{\sum_{i=1}^{N} (e_i)^2}. \tag{200}$$

Both of these quantities provide a notion of *length* of a vector in $\mathbb{R}^N$. They are usually called the l^1-norm and l^2-norm, respectively. In fact, $\|e\|_2$ is just the usual *Euclidean* length of a vector. There is a simple way

[28] As we will see in a moment, the RMSE measure is *not* strictly an average.

we can compare these two quantities; namely, we have $\|x\|_2 \leq \|x\|_1$ for all vectors $x \in \mathbb{R}^N$. Indeed,

$$\begin{aligned} \|x\|_1^2 &= \left(\sum_{i=1}^{N} |x_i|\right)^2, \\ &= \sum_{i=1}^{N} (x_i)^2 + 2\sum_{i<j} |x_i x_j|, \\ &\geq \sum_{i=1}^{N} (x_i)^2, \\ &= \|x\|_2^2. \end{aligned} \tag{201}$$

Taking the square root of both sides of equation (201), which is order-preserving, gives the result. This result allows us to relate the RMSE and MAE, because

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (e_i)^2}{N}} = \frac{1}{\sqrt{N}} \|e\|_2, \tag{202}$$

and,

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |e_i| = \frac{1}{N} \|e\|_1. \tag{203}$$

Thus, we can apply the result above to the vector of errors,

$$\begin{aligned} \|e\|_2 &\leq \|e\|_1 \\ \frac{1}{N}\|e\|_2 &\leq \frac{1}{N}\|e\|_1 \\ \frac{1}{\sqrt{N}}\left(\frac{1}{\sqrt{N}}\|e\|_2\right) &\leq \frac{1}{N}\|e\|_2 \\ \frac{1}{\sqrt{N}}\text{RMSE} &\leq \text{MAE}. \end{aligned} \tag{204}$$

As a rough idea of how the RMSE and MAE relate to each other in this paper, let's use the estimate $\sqrt{N} \approx 6$, since the number of bonds we input into our models is typically close to 36. So we see that we should expect the relationship RMSE $\leq 6\,$(MAE).

Despite the word *mean* that occurs in RMSE, this measure is *not* actually an average of squared errors. In fact, the square $(\text{RMSE})^2$ is precisely that. Of course, MAE is a true average, so it is important to keep in mind that these two measures of error are slightly different statistical objects. We nevertheless use RMSE because it is a common measure of error in many areas, such as engineering and statistics, and because it has a connection to the notion of standard deviation. As such, the RMSE is also helpful in spotting unusually large errors in data, since large errors would contribute more towards the RMSE than they would to the MAE.

We now have the necessary background to actually compare the models in terms of both RMSE and MAE. Table 1 outlines the goodness-of-fit results, in price terms, for the 561 individual dates in our example. For both the RMSE and MAE, we report the mean and median as well as the standard deviation and interquartile range (IQR) for each individual model across the range of 561 data points. We include the order statistics median and IQR to demonstrate the impact of extreme observations on a small number of dates that could unduly impact the mean and standard deviation.

Table 1: **Goodness of Fit in Price Space**: This table summarizes the *price* RMSE and MAE measures for each of the eight term-structure estimation models over the 561 days in our sample. Units are in Canadian dollars.

Models	**Price RMSE**				**Price MAE**			
	Mean	**Median**	**S. Dev.**	**IQR**	**Mean**	**Median**	**S. Dev.**	**IQR**
McCulloch	0.23	0.22	0.05	0.06	0.17	0.16	0.04	0.04
FNZ-Discount	0.32	0.31	0.06	0.07	0.21	0.20	0.04	0.05
FNZ-Zero	0.40	0.40	0.03	0.03	0.21	0.21	0.02	0.02
FNZ-Forward	1.20	1.17	0.32	0.54	0.69	0.68	0.20	0.32
MLES-Fourier	0.21	0.20	0.05	0.05	0.14	0.13	0.03	0.03
MLES-Exponential	0.22	0.22	0.05	0.05	0.14	0.14	0.03	0.04
MLES-Benchmark	0.28	0.28	0.07	0.10	0.18	0.17	0.04	0.07
Svensson	0.95	0.52	3.68	0.28	0.73	0.35	3.68	0.16

The McCulloch model seems to do the best job among the spline-based models, with an average RMSE of 23 cents on a $100 notional government bond. Conversely, the FNZ-Forward model falls into a dramatic last place, among the spline-based models, with an average RMSE of $1.20 that is almost five times worse than the McCulloch. The MAE measure tells a similar story, although, quite interestingly, the performance of the FNZ-Forward model is relatively improved. This suggests that the FNZ-Forward model has a number of large pricing errors over the sample that have led to a somewhat upward biasing of the RMSE measure. Among the function-based models, it is difficult to distinguish between the best two approaches, which are the MLES-Fourier and MLES-Exponential with average RMSE values of just over 20 cents and an average MAE of 14 cents. The MLES benchmark is slightly worse, which is understandable, given that it is more highly constrained to fit the benchmark prices perfectly. The Svensson model, finally, fares quite poorly relative to the other function-based models.

Overall, in price terms, the function-based models generally outperform the spline-based models. This is natural, given that the smoothing splines used in the Fisher, Nychka, and Zervos (1994) paper favour smooth curves at the expense of goodness of fit. Observe, however, that the non-smoothed McCulloch model compares well with the MLES models using the exponential and Fourier-basis functions.

Although we actually use bond prices to determine the parameter set, it is also useful to consider how

well the models fit the observed yields of the bonds used in the sample.[29] Table 2, therefore, summarizes the RMSE and MAE in yield terms in the same format as Table 1. In general, the results are similar but there are a few surprises. With regard to spline-based models, in particular, the McCulloch model, with an average RMSE of 8.4 basis points, no longer appears to be the forerunner. In fact, the McCulloch model is outperformed by both the FNZ-Discount and FNZ-Zero models, which demonstrate average RMSE values of 6.6 and 4.4 basis points, respectively. The same trend is also evident with the mean absolute yield errors.

Table 2: **Goodness of Fit in Yield Space**: This table summarizes the *yield* RMSE and MAE measures for each of the eight term-structure estimation models over the 561 days in our sample.

Models	**Yield RMSE**				**Yield MAE**			
	Mean	**Median**	**S. Dev.**	**IQR**	**Mean**	**Median**	**S. Dev.**	**IQR**
McCulloch	8.4	7.0	5.1	5.5	5.0	4.6	2.3	2.6
FNZ-Discount	6.6	5.6	3.2	2.5	4.8	4.2	1.8	1.8
FNZ-Zero	4.4	3.9	2.3	1.5	3.1	2.9	1.0	1.0
FNZ-Forward	15.9	14.9	9.0	7.4	10.7	10.6	3.5	4.2
MLES-Fourier	5.9	5.1	3.0	3.1	3.6	3.3	1.2	1.4
MLES-Exponential	4.2	3.7	2.5	2.0	2.8	2.6	1.1	1.2
MLES-Benchmark	4.8	4.4	2.7	1.8	3.4	3.1	1.2	1.3
Svensson	12.6	4.5	68.7	2.0	10.5	3.7	62.9	1.4

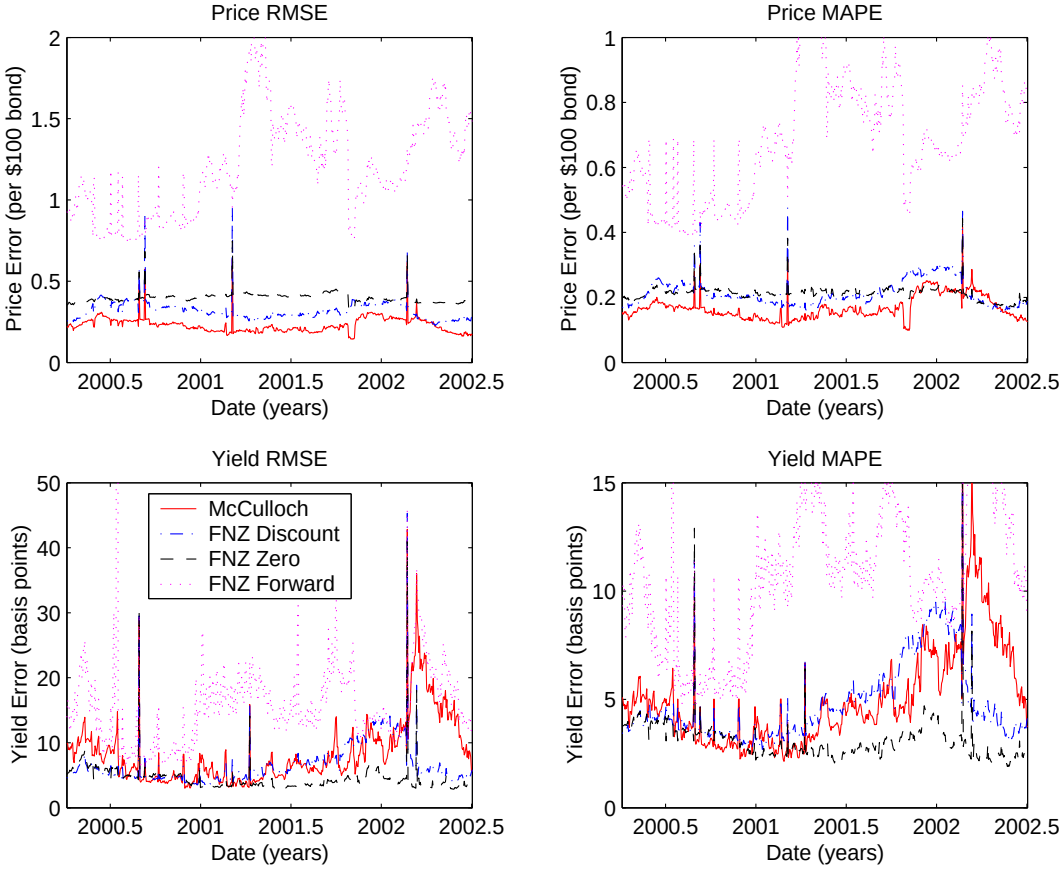
While somewhat less dramatic, there is also a change in the relative performance of the function-based models. Specifically, the previously close competition between the MLES-Exponential and MLES-Fourier models is now being led by the MLES-Exponential model. Not surprisingly, given the binding benchmark constraints, these two models are again trailed by the MLES-Benchmark approach. The Svensson model, nonetheless, continues to perform poorly, with average RMSE and MAE values at multiples of 2 or 3 of the other models.

What is responsible for this change in model ordering as we move from price to yield space? We believe it has to do with the nature of the various models. The McCulloch and MLES-Fourier models, given their functional forms, are highly flexible curve-fitting techniques. That is, linear combinations of trigonometric functions and cubic splines are easily capable of sufficient oscillation to obtain a close fit to the data. This implies that they are actually somewhat *overfitting* bond prices. When we consider their performance in yield space, however, this overfitting inhibits their relative ability to replicate the observed bond yields. In effect, the use of a weighting matrix cannot adequately compensate for the overfitting of these models in price space. Conversely, the Fisher, Nychka, and Zervos (1994) models and the MLES-Exponential models are

[29] In an ideal world, we would actually fit the bond yields directly. The incremental computational expense associated with the translation from prices to yields, however, implies that it is more efficient to use a weighting matrix to avoid overfitting to the long end of the term structure.

smoother; the former models, of course, exhibit this property by construction, while the MLES-Exponential model demonstrates this behaviour given the generally smooth properties of exponentials. We suggest that this is why the FNZ-Discount, FNZ-Zero, and MLES-Exponential models make such a successful transition into yield space.

Figure 15: **The Spline-Based Models**: This figure illustrates graphically the evolution of the price and yield-based measures of goodness of fit for the McCulloch, FNZ-Discount, FNZ-Zero, and FNZ-Forward models.



It is often difficult to interpret tables filled with numbers. As a consequence, we have graphed the evolution of the RMSE and MAE measures, in both price and yield space, over the entire data period. Figure 15 illustrates this evolution for the spline-based models. There are, at least, three things to observe in Figure 15. First, and most strikingly, is the consistently poor performance of the FNZ-Forward model. It performs worse than the alternative spline-based models for almost every observation in the 561-day series. Second, we note that, for all spline-based models, the price errors are quite stable over the entire 2.5-year interval. The yield errors, however, seem to increase towards the end of 2001 and the beginning of 2002.

As evidenced by Figure 14, the earlier period of the sample was typified by a generally flat term-structure environment. We wonder, therefore, whether this might be due to the increased steepness in the term structure over this time interval. The actual reason is nonetheless unclear. Third, the clear outperformance of the FNZ-Zero and FNZ-Discount models relative to the McCulloch methodology's dominance in price space is clearly evident in the various graphs.

Figure 16: **The Function-Based Models**: This figure illustrates graphically the evolution of the price and yield-based measures of goodness of fit for the MLES-Exponential, MLES-Fourier, MLES-Benchmark, and Svensson models.

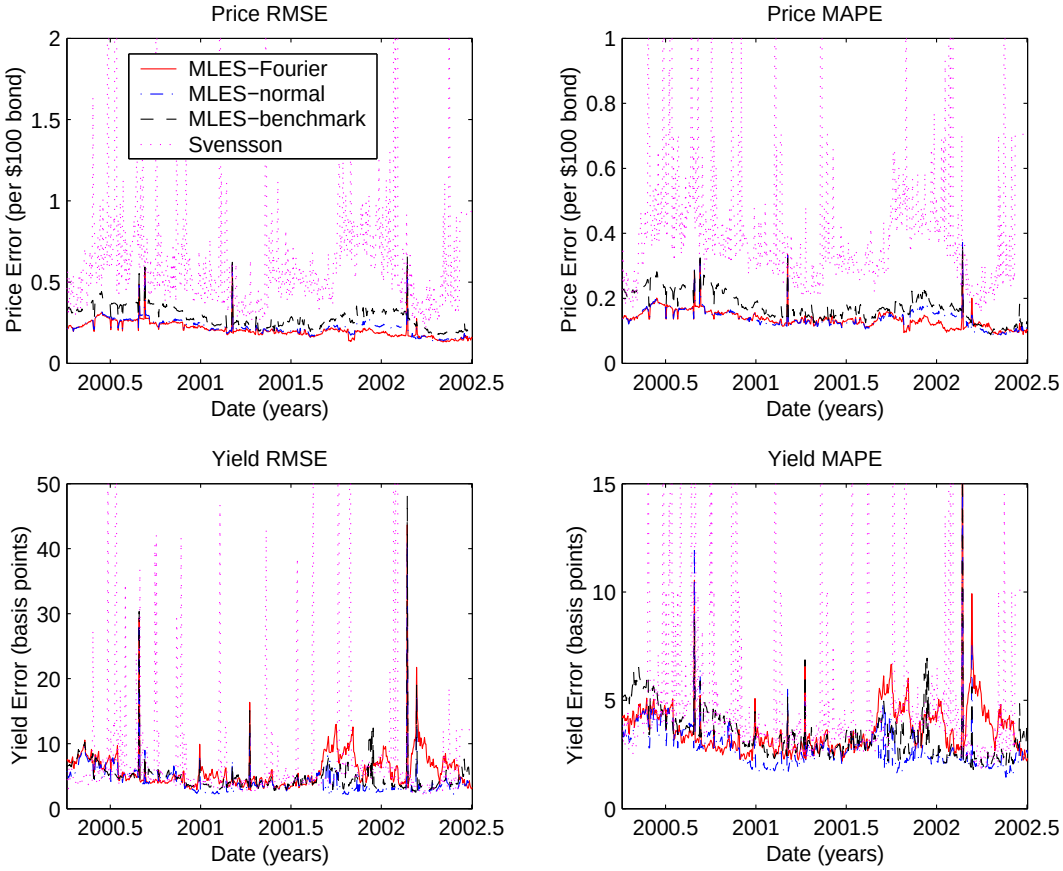


Figure 16 shows the dynamics of the RMSE and MAE measures for the group of function-based models. Again, the relatively poor performance of the Svensson model is the most striking aspect of these graphics. Observe, however, that in yield space the Svensson model appears to perform on par with the other models for a significant number of dates in the sample. It nevertheless exhibits enormous variation from one day to the next. We suspect that this occurs for two related reasons. First, the high degree of non-linearity of

the Svensson model implies that the optimization algorithm quite frequently gets stranded at local minima, instead of reaching the desired global minimum. Second, as discussed in Cairns (1997), the Svensson approach lends itself to so-called *catastrophic jumps* in parameter values from one day to another. Therefore, if one could solve these two problems with the parameter estimation, the Svensson model may not be as bad as one might conclude from examining Tables 1 and 2.[30]

The general trends in yield and price errors observed in Tables 1 and 2 are also evident in Figure 16. In particular, the MLES-Fourier and MLES-Exponential dominate in price space while the MLES-Fourier model tends to underperform in yield space. Once again, among the yield measures there appears to be somewhat more variance and a poor fit towards the end of the sample. Among these function-based models, however, the MLES exponential appears to be the most stable over this time interval.

The next element in our analysis is an attempt to understand the *nature* of the pricing errors. Each of the individual pricing algorithms requires a point estimate for each of the observed government coupon bond prices. In reality, these observed prices are quoted as a spread; one purchases at the *offer* price and sells at the *bid* price. We denote the offer and bid price of the ith bond as as P_i^{o} and P_i^{b}, respectively. For the purposes of our pricing algorithms, therefore, we use the midpoint of this bid-offer interval as a point estimate for the theoretical bond price. Clearly, this is an assumption and thus we would like to characterize the errors into three categories: inside the bid-offer spread, above the offer, and below the bid. More specifically, a theoretical bond price that lies between the offer and bid price is, for all intents and purposes, a correctly priced bond. We define the number of theoretical bond prices, as a proportion of the overall number of bonds in the daily sample, as the *hit ratio*. If, we let $\mathcal{A}$ represent the entire set of N bonds on a given date used to estimate the term structure, then mathematically we define the hit ratio as,

$$\text{Hit ratio} = \frac{\text{card}\left(\{\hat{P}_i \in \mathcal{A} : P_i^{\text{b}} \leq \hat{P}_i \leq P_i^{\text{o}}\}\right)}{N}, \tag{205}$$

where $\text{card}(\cdot)$ represents the cardinality of the set.

We define the same concepts for the overpricing and underpricing of bonds in the data sample. If, for example, the theoretical price of a bond is high relative to the observed price (i.e., it exceeds the offer price), then we would characterize this bond as being relatively inexpensive, or *cheap*. In other words, the theoretical model is suggesting that this bond is a good deal. In an effort to quantitatively describe the proportion of theoretical bond prices that *overprice* the bonds in the sample, we examine an object called the *cheap ratio*. Using the notation from equation (205), it is defined as,

$$\text{Cheap ratio} = \frac{\text{card}\left(\{\hat{P}_i \in \mathcal{A} : \hat{P}_i \geq P_i^{\text{o}}\}\right)}{N}. \tag{206}$$

[30] Of course, these problems are structural and, as such, quite difficult if not impossible to solve.

We should, of course, also consider the opposite case, where the theoretical model underprices a given bond. In this case, the observed bond price is higher than the estimated theoretical price and suggests that the bond is relatively expensive, or *rich*. This leads to the idea of the *rich ratio*—or the proportion of bonds in the sample that are underpriced by the theoretical model—and is defined as,

$$\text{Rich ratio} = \frac{\text{card}\left(\{\hat{P}_i \in \mathcal{A} : \hat{P}_i \leq P_i^{\text{b}}\}\right)}{N}. \tag{207}$$

Having defined these three measures, we can now turn to the empirical results. Table 3 summarizes the average and standard deviation for the hit ratio, cheap ratio, and rich ratio across the 561 data points in our sample. Let's begin with our focus on the hit ratio. Among the spline-based models, the FNZ-Zero model is the clear winner, with an average hit ratio of greater than 12 per cent. This compares with 7 per cent and 8 per cent, respectively, for the McCulloch and FNZ-Discount models; we also observe a rather disappointing 3.5 per cent hit ratio for the FNZ-Forward model. The FNZ-Zero model does exhibit a rather more substantial hit-ratio volatility than the other models.

Table 3: **Nature of Pricing Errors**: This table outlines the average hit ratio, cheap ratio, and the rich ratio as well as their standard deviation for the 561 dates in our sample. These ratios are defined in equations (205) to (206), respectively.

Models	**Hit ratio**		**Cheap ratio**		**Rich ratio**	
	Mean	**S. Dev.**	**Mean**	**S. Dev.**	**Mean**	**S. Dev.**
McCulloch	8.0%	5.1%	44.4%	4.9%	47.9%	5.0%
FNZ-Discount	7.0%	4.4%	47.5%	4.3%	46.6%	4.9%
FNZ-Zero	12.3%	7.9%	48.7%	4.9%	39.2%	6.7%
FNZ-Forward	3.5%	2.4%	72.9%	6.1%	24.6%	5.4%
MLES-Fourier	11.1%	6.4%	43.3%	5.6%	45.9%	5.3%
MLES-Exponential	11.5%	6.9%	43.5%	5.2%	45.4%	5.5%
MLES-Benchmark	16.6%	6.3%	55.0%	12.6%	28.4%	9.6%
Svensson	9.0%	5.0%	54.7%	10.7%	37.2%	10.6%

The function-based models do a better job in terms of the hit ratio. The dominant model is the MLES-Benchmark model at almost 17 per cent, followed by the MLES-Exponential and MLES-Fourier models at 11 per cent. As usual, the Svensson model lags the other models, albeit with a respectable 9 per cent average hit ratio. The MLES-Benchmark model, however, enjoys a substantial structural advantage in this measure. As it is forced to fit the four benchmark bonds with a zero error, it will always enjoy a minimum hit ratio of $\frac{4}{38.7}$, or about 10 per cent.[31] It should, therefore, be no surprise that the MLES-Benchmark model dominates the others on this measure.

[31] Recall that we use an average of 38.7 bonds for each daily estimation algorithm.

We now examine the cheap and rich ratio results. How should we interpret these measures? In general, we expect that a good model should *not* have a strong bias in one direction or another. That is, at first glance, it would be desirable to have a model that produces theoretical bond prices that do not systematically over- or underestimate the observed bond prices. A quick examination of Table 3 reveals that five of the eight models do not demonstrate a noticeable bias. Specifically, the McCulloch, FNZ-Discount, FNZ-Zero, MLES-Exponential, and MLES-Fourier exhibit average rich and cheap ratios of very similar magnitude with relative low levels of volatility. The FNZ-Forward, MLES-Benchmark, and Svensson models tell a different story. The FNZ-Forward model overprices, on average, approximately 73 per cent of the bonds, and underprices about 25 per cent of the bonds. The MLES-Benchmark and Svensson models demonstrate a weaker trend in the same direction. This is a matter for some concern.

The only model that has a reasonable explanation for this overpricing behaviour is the MLES-Benchmark model. The additional constraint of zero error for the four benchmark bonds will, on average, tend to overprice the other non-benchmark bonds, because the benchmark bonds are the most liquid bonds in the marketplace and will consequently trade at higher prices. This liquidity is valuable to market participants and they will thus pay a premium for it. On average, therefore, the MLES-Benchmark will generate higher price estimates for the remaining bonds. These bonds, however, owing to their relative lower liquidity, will tend to trade at lower prices. The result is a model that will tend to overprice rather than underprice the set of observed bonds; this explains the MLES-Benchmark model's higher cheap ratio. Given that the benchmark prices are generally assumed to be the most reliable estimates of bonds prices—and thus the most reliable data for estimating zero-coupon and forward interest rates—this high cheap ratio is not necessarily an unreasonable feature in a term-structure estimation model.

The final group of measures relates to the amount of computational effort required for each of these models. A terrific term-structure model that requires, for example, a long time to converge to an optimal parameterization may not be of much practical assistance. We consider two separate measures: the amount of central-processing unit (CPU) time required and the number of model iterations. The first category applies to all models, while the number of iterations is relevant only to the Fisher, Nychka, and Zervos (1994) models, where the optimization problem is solved using an iterative, approximating linear solution to the non-linear least-squares problem.

Table 4 lists the average and the standard deviation of our two measures of computational effort. Clearly, the Svensson model is an outlier in terms of computational expense, with an average of one hour of CPU time per datapoint.[32] At a substantially faster average pace, the Fisher, Nychka, and Zervos (1994) models are the next slowest group. The FNZ-Discount model is the fastest at an average of about 6.5 minutes per day, while the FNZ-Zero model is the slowest, at almost 8.5 minutes per day. This is evident in the larger

[32] All computation was performed using MATLAB running on a Sun Microsystems Blade with the Solaris 2.8 operating system.

number of iterations required to the solve the non-linear least-squares problem. The MLES-Exponential and McCulloch models require an average of one and two minutes of CPU time, respectively, while the remaining models all need, on average, less than one minute of computation. In aggregate, however, with the exception of the Svensson model, all of these models are reasonably fast. As we are not using these models in a real-time setting, it is unnecessary to distinguish between a few minutes of computational expense. All else equal, of course, one would still lean towards a faster model.

Table 4: **Computational Effort**: This table summarizes the average amount of CPU time required to run each model as well as the number of iterations of the numerical procedure required for model convergence.

Models	**CPU time (minutes)**		**Number of iterations**	
	Mean	**S. Dev.**	**Mean**	**S. Dev.**
McCulloch	2.0	0.4	1.0	0.0
FNZ-Discount	6.4	1.0	1.0	0.0
FNZ-Zero	8.4	1.1	1325.6	57.2
FNZ-Forward	7.3	1.0	589.2	89.0
MLES-Fourier	0.1	0.0	1.0	0.0
MLES-Exponential	1.0	0.2	1.0	0.0
MLES-Benchmark	0.6	0.1	1.0	0.0
Svensson	60.0	0.0	0.0	0.0

We have now examined all eight models on a number of different levels, including goodness of fit, nature of pricing errors, and computational expense. We can make some general conclusions as to the desirabilty of the various approaches. Our plan is to select *two* estimation methodologies from each group of models (i.e., spline-based and function-based) and consider a range of stability measures. The reason we do not consider all these stability measures for all models is twofold. First, the preceding analysis is sufficient to identify some clear winners among the considered eight models. Second, and more importantly, the calculation of the stability measures is highly computationally intensive and thus, in the interest of time, we need to reduce the number of models in our analysis.

Among the spline-based models, the FNZ-Forward methodology appears to perform consistently poorly across all measures. Ultimately, we believe this is because of the indirect link—and numerous requisite intermediate calculations—between the discount function and the forward-rate curve. As a result, we will immediately eliminate this model from our analysis. The FNZ-Discount and FNZ-Zero models, conversely, appear to perform quite well. Although, in general, they seem quite similar, the FNZ-Zero model demonstrated a superior hit ratio and better performance in yield space. As such, we select the FNZ-Zero model as the first spline-based model for advancement to the stability analysis. The McCulloch model offers a number of advantages, including a close fit to the data, unbiased pricing errors, and speed of computation,

and therefore we select it as our second spline-based model.

The logic used in deciding among the function-based models is similar. Based on its consistently poor performance—all due primarily to the vagaries associated with its highly non-linear structure—we immediately reject the Svensson model. The structure of the MLES-Benchmark model, which performs well on all accounts, nevertheless does not lend itself to stability analysis, because its high degree of dependence on the benchmark bonds makes it somewhat unique. As such, we select the MLES-Fourier and MLES-Exponential models for closer examination in section 4.2.

4.2 The second experiment

The analysis of stability, in this context, relates to a simple question. How do the results change if one bond is excluded from the sample of bonds used in the estimation of the model? Ideally, the results should not change in an important manner. If, however, the results do change dramatically, then this would suggest that the model lacks stability. One common way to approach this problem—performed primarily in studies using American data—is to split the sample of bonds into two groups. One group is used to estimate the parameters of the model, while the other is used to estimate these *out-of-sample* price errors. The analyst then reshuffles the choice of bonds in each sample and attempts to see how the results might change. Again, if the results change dramatically, this may indicate that the model is sensitive to the choice of bonds in the estimation algorithm. Or, put more simply, the model would appear to lack stability.

In the Canadian market, however, we do not have this luxury. With an average of only (approximately) 39 bonds available on any given day to estimate our models, we cannot split our sample without desperately jeopardizing the performance of our models. Fortunately, we are not the only country facing this problem. The United Kingdom faces a similar challenge and, consequently, Anderson and Sleath (2001) have developed some interesting ideas for circumventing it. We thus adopt, more or less wholesale, some of their suggestions in this area.

The first idea is quite straightforward. Instead of splitting the sample of bonds in two, we eliminate a single bond and re-estimate the model parameters. We then look to see how well the model reproduces the price of the excluded bond. The bond is returned to the sample and another bond is excluded, the parameters are again re-estimated with the slightly altered data set, and the error of the newly excluded bond is computed. This is repeated for all the bonds in the sample. In this manner, we can construct a set of out-of-sample price and yield errors (Tables 5 and 6) while still having sufficient bonds to parameterize our term-structure estimation algorithms. This technique, termed *cross-validation*, is used quite commonly in statistics.

There is, however, a complication. Some reflection will reveal that computation of these out-of-sample statistics is a fairly intensive endeavour. With an average of almost 39 bonds in each daily sample, 39 re-

estimations of the model parameters are required for each individual date. Were we to use the entire sample, we would be obliged to perform more than 20,000 estimations! Our solution to this difficulty is to examine a subset of dates from the main 561-day sample. Specifically, we compute our results using 15 different dates spread evenly across our 2.5-year data sample.[33]

Table 5: **Out-of-Sample Price Errors**: This table lists the out-of-sample performance, in price space, of the subset of four models selected from the analysis in the previous section.

Models	**Price RMSE**			**Price MAE**		
	Mean	**S. Dev.**	**Max**	**Mean**	**S. Dev.**	**Max**
McCulloch	3.36	2.51	7.53	0.79	0.45	1.56
FNZ-Zero	0.45	0.04	0.50	0.24	0.03	0.28
MLES-Exponential	0.34	0.11	0.59	0.20	0.05	0.31
MLES-Fourier	1.03	0.83	3.32	0.36	0.17	0.79

Table 5 describes the result of this out-of-sample statistic in price space. For each of the 15 dates in our sample, we construct an average out-of-sample pricing error for the excluded bonds. We then report the average, standard deviation, and maximum error across our sample of 15 dates. The results are interesting. The MLES-Exponential and FNZ-Zero models demonstrate a dramatically better out-of-sample performance than the McCulloch and MLES-Fourier models. The McCulloch model, in particular, has a difficult time pricing the bonds excluded from the estimation algorithm. An almost identical set of conclusions can be drawn from the out-of-sample yield statistics listed in Table 6. The slightly less shocking size of the out-of-sample *yield* errors for the McCulloch and MLES-Fourier models suggests that the majority of the out-of-sample errors relates to the long end of the term structure. This is caused by the natural heteroscedasticity of price errors that we attempt to deal with by introducing the weighting matrix in our estimation algorithms.

Table 6: **Out-of-Sample Yield Errors**: This table lists the out-of-sample performance, in price space, of the subset of four models selected from the analysis in the previous section.

Models	**Yield RMSE**			**Yield MAE**		
	Mean	**S. Dev.**	**Max**	**Mean**	**S. Dev.**	**Max**
McCulloch	26.0	17.4	69.3	10.8	5.9	26.0
FNZ-Zero	8.4	4.2	16.7	4.4	1.4	7.4
MLES-Exponential	7.9	4.1	17.2	4.1	1.4	6.6
MLES-Fourier	11.8	7.3	31.4	6.1	2.4	12.6

We argue that the reason for this behaviour relates to the basic nature of the models. The McCulloch

[33]The most problematic model was the FNZ-Zero. At an average of 8.5 minutes per estimation, and an average of 39 bonds per sample, this requires 3.5 days of CPU time for a subset of 15 days from the original 561-day sample.

model is a non-smoothed cubic spline, while the basis functions of the MLES-Fourier are linear combinations of trigonometric functions. These are very flexible functional forms that have the ability to make the necessary adjustments to very accurately price the bonds provided to the estimation algorithm. Indeed, it appears, on the basis of this analysis, that they have a tendency to *overfit* the data. In other words, they place too much emphasis on the individual bonds in their sample, at the expense of the general trend provided by the data. The relatively less flexible, or smoother, MLES-Exponential and FNZ-Zero models do not place as much emphasis on individual observations and consequently exhibit a greater degree of model stability.

Price and yield errors are not the only items of interest in this analysis. A related issue is how the zero-coupon curve itself changes as we exclude a given bond from the estimation algorithm. Again, it would be desirable for the zero-coupon curve to be relatively insensitive to the exclusion of a given bond from the dataset. The question is how to describe the difference between two curves. Using a slight variation on the clever approach suggested by Anderson and Sleath (2001), we use two standard measures of distance between functions used in mathematical analysis. First, we let $z(t)$ denote the zero-coupon curve estimated using all available bonds, and we let $z_{\mathcal{A}\setminus i}(t)$ denote the zero-coupon curve estimated excluding bond i.[34] Second, we define a general, and well known, notion of distance between these two curves as,

$$\left\|z(t) - z_{\mathcal{A}\setminus i}(t)\right\|_p = \left(\int_0^T \left|z(t) - z_{\mathcal{A}\setminus i}(t)\right|^p dt\right)^{\frac{1}{p}}, \tag{208}$$

where T is the maturity of the longest bond in the data sample. This is what is termed an $\mathcal{L}^p$ norm. For our purposes, we will consider two special cases when $p = 1, 2$. Loosely speaking, we are essentially summing up the absolute deviations or squared deviations between the two functions over the interval of interest. Clearly, when $p = 2$, more weight is placed on large deviations between the two curves, whereas when $p = 1$ all deviations receive similar weighting.

Recall from section 4.1 that the l^1 norm will, in a finite-dimensional setting, always dominate the l^2 norm. However, the comparison of the $\mathcal{L}^1$ and $\mathcal{L}^2$ is a little different in this setting. To perform a formal comparison, we use Hölder's Inequality. Specifically, if $p, q > 0$ satisfy $\frac{1}{p} + \frac{1}{q} = 1$, then we have,

$$\|fg\|_1 \leq \|f\|_p \|g\|_q, \tag{209}$$

for all $f \in \mathcal{L}^p$ and $g \in \mathcal{L}^q$. We will take $p = q = 2$, and $g = \mathbb{1}_{[0,T]}$. This gives,

$$\|f\|_1 \leq T \cdot \|f\|_2, \tag{210}$$

for all $f \in \mathcal{L}^2$. Compared with the finite-dimensional case, the inequality is reversed, but a constant, which turns out to be the full measure of the space $[0, T]$, is introduced.[35]

[34] Recall that we defined $\mathcal{A}$ to be the set of N bonds used in each estimation algorithm.

[35] This argument is valid only for *finite* measure spaces (i.e., spaces where the function $\mathbb{1}$ is integrable).

Table 7 provides an overview of the sensitivity of the zero-coupon curve to a systematic exclusion of one bond at a time from the estimation algorithm. The values in the table do not have much meaning when considered individually. Instead, the idea is to focus on a relative comparison of the measures associated with different models. The results are, in fact, consistent with the out-of-sample error comparison. In particular, the FNZ-Zero and MLES-Exponential models exhibit roughly *half* of the variation in the zero-coupon curve when a given bond is excluded from the estimation algorithm, compared with the McCulloch and MLES-Fourier model. This result is invariant under the $\mathcal{L}^1$ and $\mathcal{L}^2$ norms. As a conclusion, this should hardly be surprising. Large differences in out-of-sample errors between the models should correspond to large differences in not only the zero-coupon curve but also the discount function and the associated forward curve. The results of Table 7 do, however, lend credibility to our argument that the McCulloch and MLES-Fourier models tend to overfit the data relative to the less flexible, but more stable, MLES-Exponential and FNZ-Zero models.

Table 7: **Zero-Coupon Curve Stability**: Using the form of equation (208) with $p = 1, 2$, this table compares the average distance between the base zero-coupon curve on a given date and the curve computed by systematically excluding one bond at a time throughout the entire sample. All values are scaled by a factor of 10^5.

Models	$\mathcal{L}^1$ **Distance**			$\mathcal{L}^2$ **Distance**		
	Mean	**S. Dev.**	**Max**	**Mean**	**S. Dev.**	**Max**
McCulloch	19.60	5.32	28.10	1.96	0.60	3.18
FNZ-Zero	10.30	3.00	15.86	1.01	0.27	1.63
MLES-Exponential	9.40	2.96	13.58	1.01	0.40	1.63
MLES-Fourier	17.34	6.60	26.96	2.16	0.99	3.82

5 Conclusion

In this paper, we have considered two separate approaches to term-structure estimation: spline-based and function-based models. In section 2, we worked through the necessary mathematical preliminaries to gain a thorough understanding of the spline-based and function-based methodologies described in section 3. The goal in these sections was to provide a thorough, relatively straightforward and self-contained discussion of the mathematical underpinnings of these curve-fitting techniques. To further contribute to the clarity of these models, the appendix shows associated computer programs for a substantial number of the numerical techniques.

In section 4 we performed an extended quantitative comparison of these various approaches. We estimated each of the eight different models using almost 600 different dates spanning the approximately 2.5-year period from April 2000 to July 2001. We compared and contrasted our eight term-structure estimation models on

the basis of goodness of fit, composition of pricing errors, and computational efficiency. In our analysis, we did not observe any systematic difference between spline-based and function-based models. We did, however, identify the FNZ-Forward and Svensson models as being quite clearly undesirable.

In the second step of our numerical experiment we halved the number of models in our comparison down to four, based on the results of this first part of the analyis. We ultimately selected the McCulloch, FNZ-Zero, MLES-Exponential, and MLES-Fourier models for more detailed comparison on the basis of their relative performance in the first experiment. The form of this comparison involved re-estimation of the parameters of each of these four models systematically excluding each bond from the data set one at a time. This cross-validation procedure permitted the construction of out-of-sample price and yield errors as well as a measure of the volatility of the zero-coupon curve from the base curve estimated with the full complement of bonds. We concluded from this analysis that the MLES-Fourier and McCulloch models tend to overfit the data and consequently tend to perform relatively poorly out-of-sample. Furthermore, this poor out-of-sample performance contributes to a lower degree of stability among these two models.

In conclusion, therefore, we suggest that the MLES-Exponential and the FNZ-Zero approaches are the most desirable term-structure estimation models among the eight separate models when applied to the Government of Canada bond market. There is relatively little to distinguish between these two remaining models, other than the significantly faster computational speed of the MLES model. We nevertheless recommend that the Bank of Canada implement and estimate *both* of these models on a regular basis. The fundamentally different construction of the two models will permit a reasonable error-checking mechanism for the other model, and will subsequently contribute to a stronger long-run understanding of the evolution of the zero-coupon and forward term structure.

Appendix: MATLAB Code

A word of warning is in order for the code appearing in this appendix. We are neither computer scientists nor experts in the area of numerical analysis. There are most certainly more efficient and desirable ways to structure the computer programs that perform the tasks in this appendix. In short, our objective is *not* to demonstrate a correct numerical technique for solving these problems. Our goal instead is to provide some of the code that we used in estimating the results in this paper and thereby provide additional conceptual clarity. We hope that, considered in this light, these might prove useful to some readers.

A.1 Tridiagonal cubic spline approach: tSpline.m

```
function [p]=tSpline(k,f)
step=10;
N=length(k);
for(i=2:N)
  h(i-1)=k(i)-k(i-1);
  s(i-1)=(f(i)-f(i-1))/h(i-1);
end
for(i=1:N)
  if(i==1)
    g(i)=0;
    d(i)=0;
  elseif(i==N)
    g(i)=1;
    d(i)=0;
  else
    g(i)=h(i)/(h(i-1)+h(i));
    d(i)=6*((s(i)-s(i-1))/(h(i-1)+h(i)));
  end
end
% Build and solve our tridiagonal matrix
V = spalloc(N,N,3*N);
V(1,1:2) = [2 g(1)];
for i = 2:N-1
  V(i,i-1:i+1) = [1-g(i) 2 g(i)];
end
V(N,N-1:N) = [1-g(N) 2];
m=V\d';
% Building the piecewise polynomial
for(i=1:N-1)
  x=linspace(k(i),k(i+1),step);
  p((1+(i-1)*step):i*step)=((m(i)*((k(i+1)-x).^3)))/(6*h(i)) + ...
                           ((m(i+1)*((x-k(i)).^3)))/(6*h(i)) + ...
                           ((f(i)-((m(i)*(h(i)^2))/6))*(k(i+1)-x))/h(i) +...
                           ((f(i+1)-((m(i+1)*(h(i)^2))/6))*(x-k(i)))/h(i);
end
```

A.2 B-spline recursion formula: recurse.m

```
function [B]=recurse(x,i,n,k)
% x => point to be evaluated,
% i => position in knot sequence,
% n => order of B-spline,
% k => knot sequence.
  if(n~=1)
    a=(x-k(i))/(k(i+n-1)-k(i));
    b=(k(i+n)-x)/(k(i+n)-k(i+1));
    B=a*recurse(x,i,n-1,k)+b*recurse(x,i+1,n-1,k);
  elseif(n==1)
    if(x<k(i))
      B=0;
    elseif(x>=k(i) & x<k(i+1))
      B=1;
    elseif(x>=k(i+1))
      B=0;
    end
  end
```

A.3 Cubic B-spline approach: bSpline.m

```
function [p]=bSpline(f,k,step,a)
% Generate & solve linear system
N=length(k)-4;
if(nargin<4)
  V = spalloc(N,N,3*N);
  V(1,1:3) = [1 -2 1];
  for i = 2:N-1
    V(i,i-1:i+1) = [1/6 2/3 1/6];
  end
  V(N,N-2:N) = [1 2 1];
  a=V\f';
else
  % Compute associated B-splines
  for(j=1:4)
      x=linspace(k(j+3),k(j+4),step);
      for(i=1:length(x))
        B1(i)=recurse(x(i),j,4,k);
        B2(i)=recurse(x(i),j+1,4,k);
        B3(i)=recurse(x(i),j+2,4,k);
        B4(i)=recurse(x(i),j+3,4,k);
      end
      p((1+(j-1)*step):j*step)=a(j)*B1+a(j+1)*B2+a(j+2)*B3+a(j+3)*B4;
      X((1+(j-1)*step):j*step)=x;
  end
end
```

A.4 Least-squares cubic B-spline: regSpline.m

```
function [p]=regSpline(f,k,x,step)
```

```
% Construct and solve linear system
N=length(x);
m=length(k);
E=zeros(N,m-4);
for(i=1:N)
  for(j=4:m-4)
    if(x(i)>=k(j) & x(i)<=k(j+1))
      for(w=1:m-4)
        E(i,w)=recurse(x(i),w,4,k);
      end
    end
  end
end
c=inv(E'*E)*E'*f';
% Generate linear combination of B-splines
p=bSpline(f,k,step,c)
```

A.5 Definite integral of a B-spline: integrateB.m

```
function [B]=integrateB(i,d,x,n,k)
% i=>element of B-spline to integrate
% d=>index for lower bound of integration
% x=>value for upper bound of integration
% n=>order of B-spline
% k=>knot sequence
for(w=1:2)
  if(w==2)
    x=k(d);
  end
  if(x<=k(i))
    Bint(w)=0;
  elseif(x>k(i) & x<k(i+n))
    a=(k(i+n)-k(i))/n;
    for(j=1:n-1)
      temp(j)=((x-k(i+j))/(k(i+n)-k(i+j)))...
              *recurse(x,i+j,n-j,k);
    end
    temp(n)=((x-k(i))/(k(i+n)-k(i)))...
            *recurse(x,i,n,k);
    Bint(w)=a*sum(temp);
    clear temp;
  elseif(x>=k(i+n))
    Bint(w)=(k(i+n)-k(i))/n;
  end
end
B=Bint(1)-Bint(2);
```

A.6 Derivative of a B-spline: differentiateB.m

```
function [B]=differentiateB(i,x,n,k,order)
% i=>element of B-spline to integrate
```

```
% x=>value for upper bound of integration
% n=>order of B-spline
% k=>knot sequence
% order=> 1 computes derivative
%         2 computes derivative
if(order==1)
  a=recurse(x,i,n-1,k)/(k(i+n-1)-k(i));
  b=recurse(x,i+1,n-1,k)/(k(i+n)-k(i+1));
  B=(n-1)*(a-b);
elseif(order==2)
  a=differentiateB(i,x,n-1,k,1)/(k(i+n-1)-k(i));
  b=differentiateB(i+1,x,n-1,k,1)/(k(i+n)-k(i+1));
  B=(n-1)*(a-b);
end
```

A.7 MLES: weighted benchmark commands

```
% N=> number of basis functions
% K=> weighting of benchmarks
N = 9;
K = 1;
prices = (spoffer + spbid)./2;
smod_dur = sdur./(1+(syld./200));
weights = diag(1./smod_dur) * (diag(sbk*(K-1))+diag(ones(length(sbk),1)));
H = construct_H(scpmt,scttm,alpha,N);
lambda_hat = gls(H, weights,prices);
[e,p,p_th] = priceerrors(scpmt,scttm,weights,prices,alpha,N,lambda_hat);
[scpn sttm sbk p p_th p-p_th]
```

A.8 MLES: construct_H.m

```
function [H] = construct_H(C,T,alpha,N)
% Constructs a matrix H based on cash flows and times to maturity
% C is a matrix where each row is a vector of cash flows for a specific
% bond and the matrix T contains the corresponding times (i.e. maturities)
% of each cash flow. C and T must have the same size
[num_bonds , num_time_divisions] = size(C);
for j = 1:num_bonds
  for k = 1:N
    temp = 0;
    for m = 1:num_time_divisions
      temp = temp + C(j,m)*e_basis(alpha,k,T(j,m));
    end
    H(j,k) = temp;
  end
end
```

A.9 MLES: gls.m

```
function [lambda_hat] = gls(H,W,p)
% GLS estimate of the MLES basis parameters
```

```
% H = matrix obtained from cash flows and basis functions
% W = diagonal matrix of bond weights
% p = column vector of observed bond prices
lambda_hat = (H'*W*H)\(H'*(W*p));
```

A.10 MLES: priceerrors.m

```
function [e,p,p_th] = priceerrors(C,T,W,p,alpha,N,lambda_hat)
% Computes bond price errors
% Use norm(e)^2 to get the sum of the squared errors
% e(i) is the (observed price - theoretical price)
% e(i) < 0 indicates cheap
% e(i) > 0 indicates rich
[num_bonds num_times] = size(C);
% initializing: -1 will never be a legitimate time value
time_list=[-1 -1];
p_th=zeros(num_bonds,1);
% Calculates the theoretical bond prices, keeping a list
% to avoid redundant computations
for i = 1:num_bonds
  temp = 0;
  for j = 1:num_times
    temp2 = 0;
    if C(i,j)~=0
      [length two] = size(time_list);
      for k = 1:length
        if T(i,j) == time_list(k,1)
          temp = temp + C(i,j)*time_list(k,2);
          break
        else
          if k == length
            temp2 = discount(T(i,j),alpha,N,lambda_hat);
            time_list = [time_list; T(i,j) temp2];
            temp = temp + C(i,j)*temp2;
          end
        end
      end
    end
  end
  p_th(i,1) = temp;
end
e = p - p_th;
residual_error = norm(e)/sqrt(num_bonds)
```

A.11 MLES: zero-error benchmark commands

```
N = 8;
prices = (spoffer + spbid)./2;
smod_dur = sdur./(1+(syld./200));
weights = diag(1./smod_dur)
H = construct_H(scpmt,scttm,alpha,N);
L = construct_L(scpmt,scttm,alpha,N,sbk,weights,H);
```

```
lambda_hat_bench = lambda_hat_B(scpmt,scttm,alpha,L,weights,prices,sbk,N,H);
[e,p,p_th] = priceerrors_bench(scpmt,scttm,weights,prices,alpha,N,sbk,H,L...
        ,lambda_hat_bench);
[scpn sttm sbk p p_th p-p_th]
```

A.12 MLES: construct_L.m

```
function [L] = construct_L(C,T,alpha,N,bk,W,H)
% Constructs a matrix L based on cash flows and times to maturity
% L is a matrix used for the constrained (benchmark error = 0) GLS
% optimization problem. C is a matrix where each row is a vector of cash
% flows for a specific bond and the matrix T contains the corresponding
% times (i.e. maturities) of each cash flow. C and T must have the same size
[num_bonds , num_time_divisions] = size(C);
H_B = zeros(sum(bk),N);
inc_B = 0;
for n = 1:num_bonds
  if bk(n) == 1
    inc_B = inc_B + 1;
    for l = 1:N
      temp = 0;
      for i = 1:num_time_divisions
        temp = temp + C(n,i)*e_basis(alpha,l,T(n,i));
      end
      H_B(inc_B,l) = temp;
    end
  end
end
L = [ H'*W*H H_B' ; H_B zeros(sum(bk),sum(bk)) ];
```

A.13 MLES: lambda_hat_B.m

```
function [lambda_hat_B] = lambda_hat_B(C,T,alpha,L,W,p,bk,N,H)
% Gives the weighted Least Squares estimate of the MLES basis
% co-efficients. This is the zero-error benchmark case
% W = diagonal matrix of bond weights
% p = column vector of observed bond prices
[num_bonds, num_time_divisions] = size(C);
%Initialize the variables
H_B = zeros(sum(bk),N);
inc_B = 0;
p_B = zeros(sum(bk),1);p_N = zeros(length(p)-sum(bk),1);
lambda_hat_B = zeros(N,1);
%Constructs the matrix H_B which is the "H" matrix associated with the
%benchmarks only
for n = 1:num_bonds
  if bk(n) == 1
    inc_B = inc_B + 1;  %a running index of the benchmark bond number
    for l = 1:N
      temp = 0;
      for i = 1:num_time_divisions
```

```
        temp = temp + C(n,i)*e_basis(alpha,l,T(n,i));
      end
      H_B(inc_B,l) = temp;
    end
    p_B(inc_B) = p(n);
  end
end
temp = L\[H'*W*p ; p_B];
for i = 1:N
  lambda_hat_B(i) = temp(i); %disregards the Lagrange multipliers
end
```

A.14 MLES: priceerrors_bench.m

```
function [e,p,p_th] = priceerrors_bench(C,T,W,p,alpha,N,sbk,H,L,lambda_hat)
%Gives bond pricing errors in the zero-error benchmark case
%Use norm(e)^2 to get the sum of the squared errors
% e(i) is the observed price - theoretical price
% e(i) < 0 indicates cheap
% e(i) > 0 indicates rich
[num_bonds num_times] = size(C);
time_list=[-1 -1];  % initializing: -1 will never be a legitimate time value
p_th=zeros(num_bonds,1);
%Computes the theoretical price of each bond
for i = 1:num_bonds
  temp = 0;
  for j = 1:num_times
    temp2 = 0;
    if C(i,j)~=0 %avoids calculations we don't need to do
      [length two] = size(time_list); %list of times we have computed already
      for k = 1:length
        if T(i,j) == time_list(k,1)  %checks for a previous match
          temp = temp + C(i,j)*time_list(k,2);
          break
        else    %a new time
          if k == length
            temp2 = discount(T(i,j),alpha,N,lambda_hat);
            time_list = [time_list; T(i,j) temp2];  %adds time and
                                                    %discount value to the list
            temp = temp + C(i,j)*temp2;
          end
        end
      end
    end
  end
  p_th(i,1) = temp;
end
e = p - p_th;
residual_error = norm(e)/sqrt(num_bonds)
```

Bibliography

Anderson, N., F. Breedon, M. Deacon, A. Derry, and G. Murphy (1996): *Estimating and Interpreting the Yield Curve*. John Wiley and Sons, West Sussex, England.

Anderson, N., and J. Sleath (2001): "New Estimates of the UK Real and Nominal Yield Curves," Bank of England Working Paper.

Bliss, R. R. (1996): "Testing Term Structure Estimation Methods," Federal Reserve Bank of Atlanta: Working Paper 96-12.

Bolder, D., and D. Stréliski (1999): "*Yield Curve Modelling at the Bank of Canada*," Technical Report No. 84. Ottawa: Bank of Canada.

Cairns, A. J. G. (1997): "Descriptive Bond-Yield and Forward-Rate Models for the Pricing of British Government Securities," Department of Actuarial Mathematics and Statistics, Heriot-Watt University: Working Paper.

Campbell, J. Y., A. W. Lo, and A. C. MacKinlay (1997): *The Econometrics of Financial Markets*. Princeton University Press, Princeton, New Jersey.

Deacon, M. (2000): "The DMO's Yield Curve Model," United Kingdom Debt Management Office: Working Paper.

deBoor, P. (1978): *A Practical Guide to Splines*. Springer Verlag, Berlin, Germany.

Dierckx, P. (1993): *Curve and Surface Fitting with Splines*. Clarendon Press, Walton Street, Oxford.

Eilers, P. H., and B. D. Marx (1996): "Flexible Smoothing with B-splines and Penalties," *Statistical Science*, 11, 89–102.

Fisher, M. (1996): "Fitting and Interpreting the U.S. Yield Curve at the Federal Reserve Board," U.S. Federal Reserve Board Working Paper.

——— (2001): "Forces That Shape the Yield Curve: Parts 1 and 2," U.S. Federal Reserve Board Working Paper.

Fisher, M., D. Nychka, and D. Zervos (1994): "Fitting the Term Structure of Interest Rates with Smoothing Splines," U.S. Federal Reserve Board Working Paper.

Jeffrey, A., O. Linton, and T. Nguyen (2000): "Flexible Term Structure Estimation: Which Method is Preferred," Yale International Centre for Finance: Discussion Paper No. ICF-00-25.

Knott, G. D. (2000): *Interpolating Cubic Splines*. Birkhäuser, Boston.

Lancaster, P., and K. Salkauskas (1986): *Curve Fitting and Surface Fitting: An Introduction*. Academic Press, Orlando, Florida.

Li, B., E. DeWetering, G. Lucas, R. Brenner, and A. Shapiro (2001): "Merrill Lynch Exponential Spline Model," Merrill Lynch Working Paper.

Linton, O., E. Mammen, J. Nielsen, and C. Tanggaard (1999): "Estimating Yield Curves by Kernel Smoothing Methods," Yale International Centre for Finance: Discussion Paper.

McCulloch, J. H. (1971): "Measuring the Term Structure of Interest Rates," *Journal of Business*, 44, 19–31.

Musiela, M., and M. Rutkowski (1998): *Martingale Methods in Financial Modelling*. Springer-Verlag, Berlin, first edn.

Nelson, C., and A. Siegel (1987): "Parsimonious Modelling of Yield Curves," *Journal of Business*, 60, 473–489.

Nürnberger, G. (1980): *Approximation by Spline Functions*. Springer Verlag, Berlin, Germany.

Nychka, D. (1995): "Splines as Local Smoothers," *Annals of Statistics*, 23, 1175–1197.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992): *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Trumpington Street, Cambridge, second edn.

Ralston, A., and P. Rabinowitz (1978): *A First Course in Numerical Analysis*. Dover Publications, Mineola, New York, second edn.

Rice, J., and M. Rosenblatt (1983): "Smoothing Splines: Regression, Derivatives and Deconvolution," *Annals of Statistics*, 11, 141–156.

Schich, S. T. (1997): "Estimating the German Term Structure," Economic Research Group of the Deutsche Bundesbank: Discussion Paper 4/97.

Schumaker, L. L. (1978): *Spline Functions: Basic Theory*. John Wiley and Sons, West Sussex, England.

Seppällä, J., and P. Viertiö (1996): "The Term Structure of Interest Rates: Estimation and Interpretation," Bank of Finland: Discussion Paper 19/96.

Shea, G. S. (1985): "Interest Rate Term Structure Estimation with Exponential Splines: A Note," *The Journal of Finance*, 40, 319–325.

Svensson, L. E. (1994): "Estimating and Interpreting Forward Interest Rates: Sweden 1992-94," International Monetary Fund: Working Paper No. 114.

Vasicek, O. A., and H. G. Fong (1981): "Term Structure Modeling Using Exponential Splines," *The Journal of Finance*, 37, 339–348.

Waggoner, D. F. (1997): "Spline Methods for Extracting Interest Rate Curves from Coupon Bond Prices," Federal Reserve Bank of Atlanta: Working Paper 97-10.

Walsh, J. H., E. N. Nilson, and J. L. Walsh (1967): *The Theory of Splines and Their Applications*. Academic Press, Fifth Avenue, New York.

Wegman, E. J., and I. W. Wright (1983): "Splines in Statistics," *Journal of American Statistical Association*, 78, 351–365.

Yandell, B. S. (1992): "Smoothing Splines: A Tutorial," *Statistician*, 42, 317–319.

Bank of Canada Working Papers
Documents de travail de la Banque du Canada

Working papers are generally published in the language of the author, with an abstract in both official languages. ***Les documents de travail sont publiés généralement dans la langue utilisée par les auteurs; ils sont cependant précédés d'un résumé bilingue.***

2002

2002-28	Filtering for Current Analysis	S. van Norden
2002-27	Habit Formation and the Persistence of Monetary Shocks	H. Bouakez, E. Cardia, and F.J. Ruge-Murcia
2002-26	Nominal Rigidity, Desired Markup Variations, and Real Exchange Rate Persistence	H. Bouakez
2002-25	Nominal Rigidities and Monetary Policy in Canada Since 1981	A. Dib
2002-24	Financial Structure and Economic Growth: A Non-Technical Survey	V. Dolar and C. Meh
2002-23	How to Improve Inflation Targeting at the Bank of Canada	N. Rowe
2002-22	The Usefulness of Consumer Confidence Indexes in the United States	B. Desroches and M-A. Gosselin
2002-21	Entrepreneurial Risk, Credit Constraints, and the Corporate Income Tax: A Quantitative Exploration	C.A. Meh
2002-20	Evaluating the Quarterly Projection Model: A Preliminary Investigation	R. Amano, K. McPhail, H. Pioro, and A. Rennison
2002-19	Estimates of the Sticky-Information Phillips Curve for the United States, Canada, and the United Kingdom	H. Khan and Z. Zhu
2002-18	Estimated DGE Models and Forecasting Accuracy: A Preliminary Investigation with Canadian Data	K. Moran and V. Dolar
2002-17	Does Exchange Rate Policy Matter for Growth?	J. Bailliu, R. Lafrance, and J.-F. Perrault
2002-16	A Market Microstructure Analysis of Foreign Exchange Intervention in Canada	C. D'Souza
2002-15	Corporate Bond Spreads and the Business Cycle	Z. Zhang
2002-14	Entrepreneurship, Inequality, and Taxation	C.A. Meh
2002-13	Towards a More Complete Debt Strategy Simulation Framework	D.J. Bolder

Copies and a complete list of working papers are available from:
Pour obtenir des exemplaires et une liste complète des documents de travail, prière de s'adresser à :

Publications Distribution, Bank of Canada
234 Wellington Street, Ottawa, Ontario K1A 0G9
E-mail: publications@bankofcanada.ca
Web site: http://www.bankofcanada.ca

Diffusion des publications, Banque du Canada
234, rue Wellington, Ottawa (Ontario) K1A 0G9
Adresse électronique : publications@banqueducanada.ca
Site Web : http://www.banqueducanada.ca