16
Term Structure of Interest Rates

16.1 Introduction

In financial markets, the term structure of interest rates is crucial to pricing of fixed income securities and derivatives. The last thirty years have seen great advances in the financial economics of term structure of interest rates. This chapter will focus on interpolating the term structure of interest rates from discrete bond yields. Refer to Campbell, Lo, and MacKinlay (1997) for basic concepts in fixed income calculations and Hull (1997) for an introduction to theoretical term structure modeling.

Section 16.2 first defines different rates, such as spot or zero coupon interest rate, forward rate, and discount rate, and documents how one rate can be converted to another. Section 16.3 shows how to interpolate term structure data using quadratic or cubic spline. Section 16.4 illustrates how to use smoothing splines to fit term structure data. Section 16.5 introduces the parametric Nelson-Siegel function and its extension and shows how it can be used to interpolate term structure data. Bliss (1997) and Ferguson and Raymar (1998) compared the performance of these different methods. Section 16.6 concludes this chapter.

16.2 Discount, Spot and Forward Rates

16.2.1 Definitions and Rate Conversion

Although many theoretical models in financial economics hinge on an abstract interest rate, in reality there are many different interest rates. For example, the rates of a three month U.S. Treasury bill are different from those of a six month U.S. Treasury bill. The relationship between these different rates of different maturity is known as the term structure of interest rates. The term structure of interest rates can be described in terms of spot rate, discount rate or forward rate.

The *discount function*, $d(m)$, gives the present value of \$1.00 which is repaid in m years. The corresponding *yield to maturity* of the investment, $y(m)$, or *spot interest rate*, or *zero coupon rate*, must satisfy the following equation under continuous compounding:

$$d(m)e^{y(m)\cdot m} = 1$$

or

$$d(m) = e^{-y(m)\cdot m} \tag{16.1}$$

Obviously, the discount function is an exponentially decaying function of the maturity, and must satisfy the constraint $d(0) = 1$.

The above equation easily shows that under continuous compounding

$$y(m) = -\frac{\log d(m)}{m}.$$

If discrete compounding is used instead, one can similarly show that

$$y(m) = p[d(m)^{-\frac{1}{p\cdot m}} - 1]$$

where p is the number of compounding periods in a year.

The spot interest rate is the single rate of return applied over the maturity of m years starting from today. It is also useful to think of it as the average of a series of future spot interest rates, or *forward rates*, with different maturities starting from a point in the future, and thus:

$$e^{y(m)\cdot m} = e^{\int_0^m f(x)dx}$$

from which one can easily obtain:

$$y(m) = \frac{1}{m}\int_0^m f(x)dx \tag{16.2}$$

with $f(m)$ denoting the forward rate curve as a function of the maturity m.

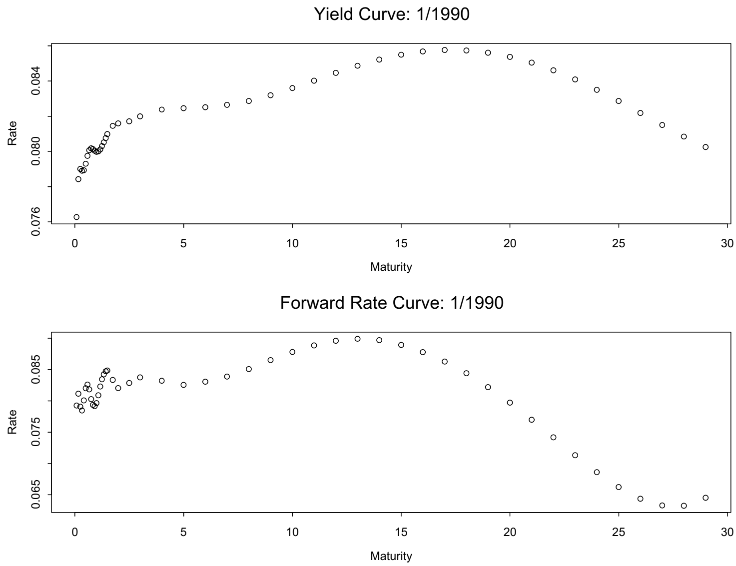

FIGURE 16.1. Yield curve and forward rate curve for January 1990.

From (16.1) and (16.2), the relationship between the discount function and forward rate can be derived:

$$d(m) = \exp\{-\int_0^m f(x)dx\}$$

or

$$f(m) = -\frac{d'(m)}{d(m)}.$$

Hence the forward rate gives the rate of decay of the discount function as a function of the maturity m. The relationship between these different rates under discrete compounding can be similarly obtained.

16.2.2 Rate Conversion in `S+FinMetrics`

To facilitate the interpolation of term structure from any of discount rate, spot rate, or forward rate, `S+FinMetrics` provides a group of functions for converting one rate into another rate. These functions will be illustrated using the `mk.zero2` and `mk.fwd2` data sets in `S+FinMetrics`, which contains the U.S. zero coupon rates and forward rates, respectively, as computed by McCulloch and Kwon (1993).

Both `mk.zero2` and `mk.fwd2` are "`timeSeries`" objects with 55 columns, with each column representing the rate with the corresponding maturity in the 55×1 vector `mk.maturity`. For example, the first element of the vector `mk.maturity` is 0.083, so the first columns of `mk.zero2` and `mk.fwd2` correspond to the rates with maturity of one month. Use the following code to plot the yield curve and forward rate curve for January 1990, and the graph is shown in Figure 16.1:

```
> par(mfrow=c(2,1))
> plot(mk.maturity,mk.zero2[54,],xlab="Maturity",ylab="Rate")
> title(paste("Yield Curve:", positions(mk.zero2[54,])))
> plot(mk.maturity,mk.fwd2[54,],xlab="Maturity",ylab="Rate")
> title(paste("Forward Rate Curve:",positions(mk.fwd2[54,])))
> par(mfrow=c(1,1))
```

To convert the spot interest rate or forward rate into the discount rate, use the `S+FinMetrics` function `bond.discount`. For example, to convert the first 48 spot rates in Figure 16.1 to discount rates, use the following command:

```
> disc.rate = bond.discount(mk.zero2[54, 1:48],
+ mk.maturity[1:48], input="spot", compounding=2)
```

The `bond.discount` function takes two required arguments: the first is a vector of rates, and the second is a vector of the corresponding maturity. Note that the optional argument `input` is used to specify the type of the input rates, and `compounding` to specify the number of compounding periods in each year. So `compounding=2` corresponds to semi-annual compounding.[1] If the input rates are forward rates, simply set `input="forward"`.

The functions `bond.spot` and `bond.forward` can be called in a similar fashion to compute the spot interest rate and forward rate, respectively, from different input rates. For all those three functions, the rates should be expressed as decimal numbers, and the maturity should be expressed in units of years. For example, to convert `disc.rate` back into the spot rates, use the following command:

```
> spot.rate = bond.spot(disc.rate, mk.maturity[1:48],
+ input="discount", compounding=2)
```

It can be easily checked that `spot.rate` is the same as `mk.zero2[54, 1:48]`.

16.3 Quadratic and Cubic Spline Interpolation

The interest rates are observed with discrete maturities. In fixed income analysis, the rate for a maturity which is not observed can sometimes be

[1] To use continuous compounding, specify `compounding=0`.

used. Those unobserved rates can usually be obtained by interpolating the observed term structure.

Since the discount rate should be a monotonically decreasing function of maturity and the price of bonds can be expressed as a linear combination of discount rates, McCulloch (1971, 1975) suggested that a spline method could be used to interpolate the discount function, or the bond prices directly. In particular, use k continuously differentiable functions $s_j(m)$ to approximate the discount rates:

$$d(m) = a_0 + \sum_{j=1}^{k} a_j s_j(m) \tag{16.3}$$

where $s_j(m)$ are known functions of maturity m, and a_j are the unknown coefficients to be determined from the data. Since the discount rate must satisfy the constraint $d(0) = 1$, set $a_0 = 1$ and $s_j(0) = 0$ for $j = 1, \cdots, k$. Note that once the functional form of $s_j(m)$ is determined, the coefficients a_j can be easily estimated by linear regression. Thus the discount rate, or forward rate, or spot rate, associated with an unobserved maturity can be easily interpolated using the above functional form, as long as the maturity is smaller than the largest maturity used in the estimation.

Figure 16.1 shows that there are usually more points in the short end of the term structure, and less points in the long end of the term structure. To obtain a reliable interpolation using the spline method, the functional form of $s_j(m)$ should be chosen so that it adapts to the density of maturity m. McCulloch (1971) gave a functional form of $s_j(m)$ using quadratic spline, which is based on piecewise quadratic polynomials, while McCulloch (1975) gave a functional form of $s_j(m)$ using cubic spline, which is based on piecewise cubic polynomials.

Term structure interpolation using quadratic or cubic spline methods can be performed by calling the `term.struct` function in `S+FinMetrics`. The arguments taken by `term.struct` are:

```
> args(term.struct)
function(rate, maturity, method = "cubic", input.type = "spot",
        na.rm = F, plot = T, compounding.frequency = 0,
        k = NULL, cv = F, penalty = 2, spar = 0, ...)
NULL
```

Similar to `bond.spot`, `bond.discount` and `bond.forward` functions, the first argument `rate` should be a vector of interest rates, while the second argument `maturity` specifies the corresponding maturity in units of years. The type of the input interest rate should be specified through the optional argument `input.type`. Note that the quadratic or cubic spline methods operate on discount rates. If the input interest rates are not discount rates, the optional argument `compounding.frequency` should also be set for proper conversion, which is set to zero for continuous compounding

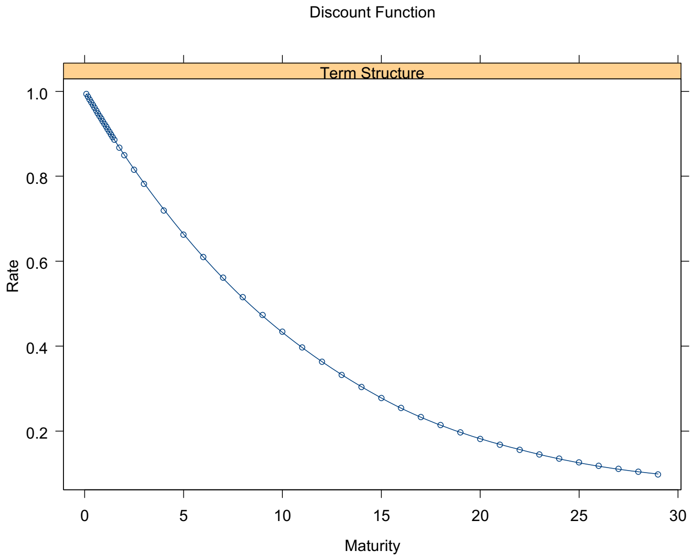

FIGURE 16.2. U.S. discount function for January 1990: quadratic spline.

by default. The optional argument `k` determines the number of functions in (16.3), also known as *knot points*. By default, follow McCulloch (1971, 1975) and set $k = [\sqrt{n}]$ where n is the length of the input rates. Other optional arguments will be discussed in later sections.

To illustrate the usage of the spline methods, in order to interpolate the term structure corresponding to January 1990, using `mk.zero2`, use the following command:

```
> disc.rate = term.struct(mk.zero2[54,], mk.maturity,
+ method="quadratic", input="spot", na.rm=T)
```

Note that `na.rm=T` is set to remove the missing values at the long end of the term structure. By default, the interpolated discount rate is plotted automatically, which is shown in Figure 16.2. The points in the figure represent the original discount rates, while the line represents the spline interpolation.

The returned object `disc.rate` is of class "`term.struct`". As usual, typing the name of the object at the command line invokes its `print` method:

```
> class(disc.rate)
[1] "term.struct"
> disc.rate

Call:
```

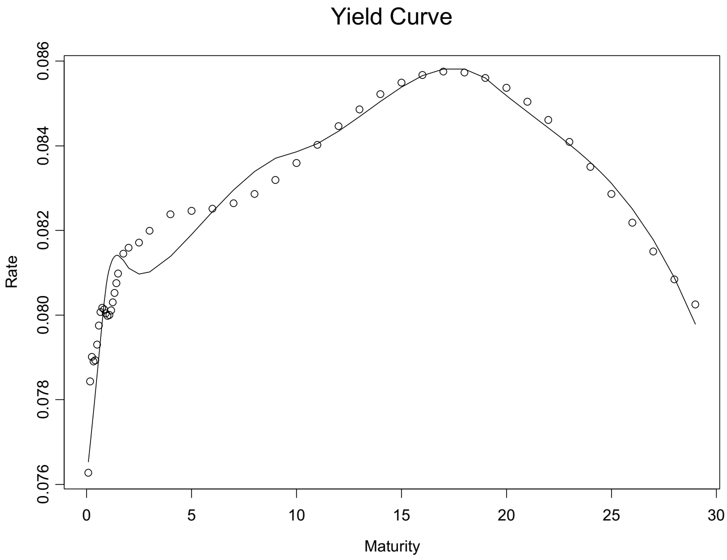

FIGURE 16.3. U.S. yield curve for January 1990: quadratic spline.

```
term.struct(rate = mk.zero2[54,  ], maturity = mk.maturity,
      method = "quadratic", input.type = "spot", na.rm = T)

Coefficients:
      a1      a2      a3      a4      a5      a6
 -0.0761 -0.0791 -0.0688 -0.0373 -0.0146 -0.0045

Degrees of freedom: 48 total; 42 residual
Residual standard error: 0.001067688
```

Since the unknown coefficients a_j of the spline are estimated by linear regression, the output looks very much similar to linear regression output. Since there are 48 spot rates available for January 1990, the number of knot points is chosen to be 6 by default.

The plot generated in Figure 16.2 shows the interpolated discount function because the quadratic or cubic spline methods are designed to operate on discount function. This plot can be later regenerated by calling the generic `plot` function on a "`term.struct`" object. However the yield curve or forward rate curve is usually of more interest. These can also be easily plotted using the components of a "`term.struct`" object. For example, use the S-PLUS `names` function to find out the components of `disc.rate`:

```
> names(disc.rate)
 [1] "coefficients" "residuals"  "fitted.values" "effects"
```

```
 [5] "R"            "rank"      "assign"      "df.residual"
 [9] "contrasts"    "terms"     "call"        "fitted"
[13] "knots"        "method"    "maturity"    "rate"
```

The first 10 components are inherited from an "lm" object, because the S-PLUS lm function is used for the linear regression. The fitted (instead of the fitted.values) component represents the estimated discount rates associated with the maturity component. To plot the interpolated yield curve or forward rate curve, simply convert the estimated discount rates into the rates you want. For example, use the following code to plot the interpolated yield curve:

```
> spot.rate = bond.spot(disc.rate$fitted, disc.rate$maturity,
+ input="discount", compounding=0)
> plot(mk.maturity[1:48], mk.zero2[54,1:48],
+ xlab="Maturity", ylab="Rate", main="Yield Curve")
> lines(disc.rate$maturity, spot.rate)
```

and the plot is shown in Figure 16.3. Note that in the plot the points represent the original zero coupon rates, while the line represents the quadratic spline interpolation.

16.4 Smoothing Spline Interpolation

The previous section demonstrated that the polynomial spline methods proposed by McCulloch (1971, 1975) can fit the discount rate and yield curve very well. However, since the methods operate on (linear combinations of) discount functions, the implied forward rate curve usually has some undesirable features. For example, use the following code to generate the implied forward rate curve from the object disc.rate fitted in the previous section:

```
> fwd.rate = bond.forward(disc.rate$fitted, disc.rate$maturity,
+ input="discount", compounding=0)
> plot(disc.rate$maturity, fwd.rate, type="l",
+ xlab="Maturity", ylab="Rate", main="Forward Rate")
> points(mk.maturity[1:48], mk.fwd2[54, 1:48])
```

The plot is shown in Figure 16.4. The implied forward rate is way off at the long end of the term structure.

In addition to the undesirable behavior of implied forward rate, the choice of knot points for polynomial splines is rather *ad hoc*. For a large number of securities, the rule can imply a large number of knot points, or coefficients a_j. To avoid these problems with polynomial spline methods, Fisher, Nychka and Zervos (1995) proposed to use *smoothing splines* for interpolating the term structure of interest rates.

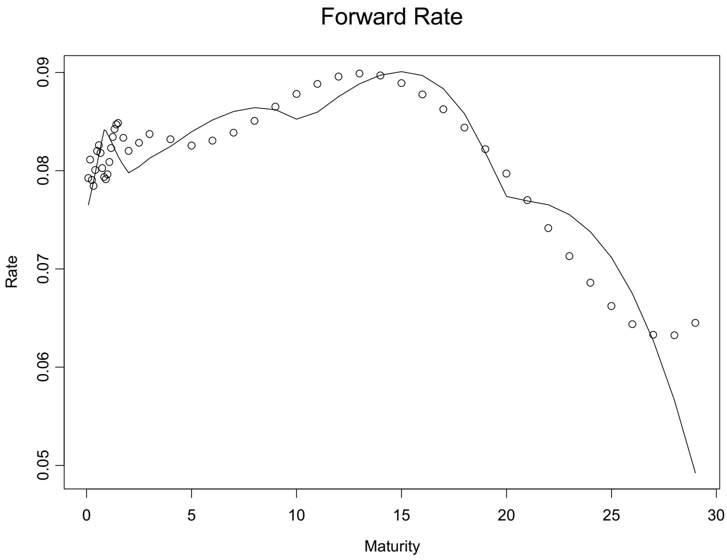

FIGURE 16.4. U.S. forward rate for January 1990: quadratic spline.

In general, for an explanatory variable x_i and a response variable y_i, the smoothing spline tries to find a smooth function $f(\cdot)$ to minimize the penalized residual sum of squares (PRSS):

$$\text{PRSS} = \sum_{i=1}^{n}[y_i - f(x_i)]^2 + \lambda \int [f''(t)]^2 dt \tag{16.4}$$

where the first term is the residual sum of squares (RSS), and the second term is the penalty term, and the parameter λ controls the trade-off between goodness-of-fit and parsimony. By using the penalty term, the spline function can be over-parameterized, while using λ to reduce the effective number of parameters.

Let S denote the $n \times n$ implicit smoother matrix such that $f(x_i) = \sum_{j=1}^{n} S(x_i, x_j)y_j$. Fisher, Nychka and Zervos (1995) suggested using generalized cross validation (GCV) to choose λ. That is, λ is chosen to minimize

$$\text{GCV} = \frac{\text{RSS}}{n - \theta \cdot \text{tr}(S)}$$

where θ is called the *cost*, and $\text{tr}(S)$ denotes the trace of the implicit smoother matrix and is usually used as the measure of effective number of parameters.

Interpolation of term structure using smoothing spline can also be performed using the `term.struct` function by setting the optional argument

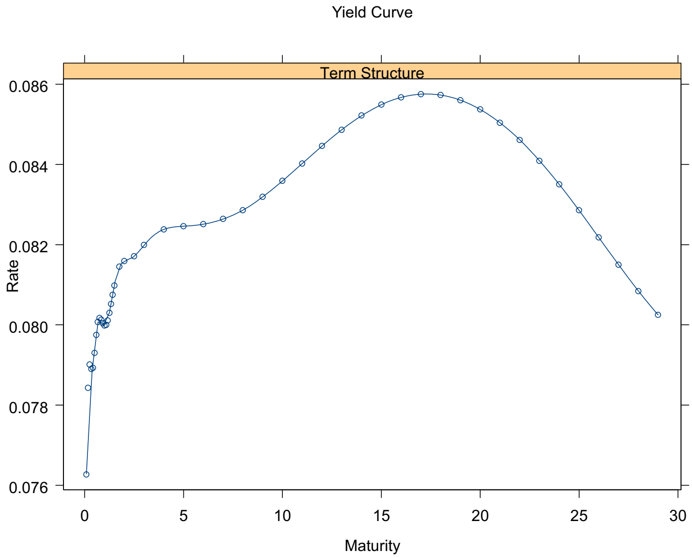

FIGURE 16.5. U.S. yield curve for January 1990: smoothing spline.

`method="smooth"`. The procedure uses the `S-PLUS` `smooth.spline` function as the workhorse.[2] In particular, for all the arguments taken by the function `term.struct`, `cv`, `penalty` and `spar` are specifically used for smoothing spline methods and passed to the `smooth.spline` function. By default, use GCV by setting `cv=F` and thus `spar`, which specifies the value of λ, is ignored.[3] The optional argument `penalty` is used to specify the value for θ. Following Fisher, Nychka, and Zervos (1995), set $\theta = 2$ by default.

For example, use the following command to interpolate the yield curve for January 1990, with the smoothing spline method:

```
> fnz.fit = term.struct(mk.zero2[54,], mk.maturity,
+ method="smooth", input="spot", na.rm=T)
```

Again, the interpolated yield curve is plotted automatically, as shown in Figure 16.5. Although the returned object `fnz.fit` is of class “`term.struct`”, its components are different from the `disc.rate` object fitted in the previous section, because now the `smooth.spline` function is used as the workhorse:

---

[2]Refer to Hastie (1993) and *S-PLUS Guide to Statistics* for the description of `smooth.spline` function.

[3]For further details regarding these arguments, see the on-line help file for `smooth.spline` function.

```
> class(fnz.fit)
[1] "term.struct"
> names(fnz.fit)
 [1] "x"        "y"      "w"      "yin"    "lev"    "cv.crit"
 [7] "pen.crit" "df"     "spar"   "fit"    "call"   "method"
[13] "maturity" "rate"
```

The first 10 components are inherited from a "smooth.spline" object, while the last four components are generated by the term.struct function. For the same reason, the print function now shows different information:

```
> fnz.fit
Call:
term.struct(rate = mk.zero2[54,  ], maturity = mk.maturity,
        method = "smooth", input.type = "spot", na.rm = T)

Smoothing Parameter (Spar): 4.767984e-11
Equivalent Degrees of Freedom (Df): 47.57122
Penalized Criterion: 4.129338e-10
GCV: 3.605842e-14
```

which shows the optimal smoothing parameter λ, and its associated GCV, penalized criterion, and equivalent degrees of freedom.

For "term.struct" objects, S+FinMetrics also implements a predict method, which can be used to obtain the interpolated rate associated with an arbitrary vector of maturity. For example, to recover the fitted spot rates from fnz.fit, use the predict method as follows:

```
> fnz.spot = predict(fnz.fit, fnz.fit$maturity)
```

From the fitted spot rates, one can compute the implied forward rates for the smoothing spline:

```
> fnz.forward = bond.forward(fnz.spot, fnz.fit$maturity,
+ input="spot", compounding=0)
> plot(mk.maturity[1:48], mk.fwd2[54,1:48],
+ xlab="Maturity", ylab="Rate", main="Forward Rate")
> lines(fnz.fit$maturity, fnz.forward)
```

The "real" forward rates and the smoothing spline interpolations are shown together in Figure 16.6. The interpolations agree very well with the "real" forward rates. The slight difference is partly caused by the fact that mk.zero2[54,] and the spot rates implied by mk.fwd2[54,] are slightly different.

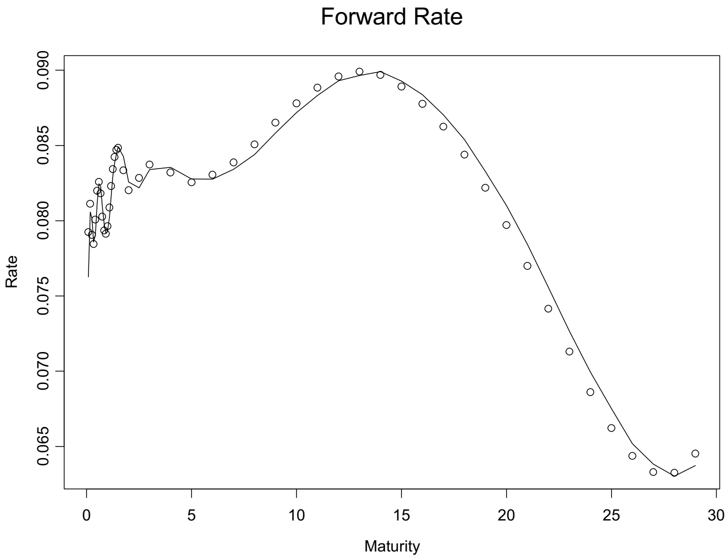

FIGURE 16.6. U.S. forward rate for January 1990: smoothing spline.

16.5 Nelson-Siegel Function

The previous sections have shown that both the polynomial and smoothing spline methods can fit the term structure very well, except that the implied forward rates from polynomial spline methods have some undesirable features at the long end of the term structure. However, the non-parametric spline based methods usually do not generate good out-of-sample forecasts. There is substantial evidence showing that a parametric function suggested by Nelson and Siegel (1987) has better out-of-sample forecasting performance.

Using a heuristic argument based on the expectation theory of the term structure of interest rates, Nelson and Siegel (1987) proposed the following parsimonious model for the forward rate:

$$f(m) = \beta_0 + \beta_1 \cdot e^{-m/\tau} + \beta_2 \cdot m/\tau \cdot e^{-m/\tau}.$$

They suggested that the model may also be viewed as a constant plus a Laguerre function, and thus can be generalized to higher-order models. Based on the above equation, the corresponding yield curve can be derived as follows:

$$y(m) = \beta_0 + \beta_1 \frac{1 - e^{-m/\tau}}{m/\tau} + \beta_2 \left[\frac{1 - e^{-m/\tau}}{m/\tau} - e^{-m/\tau} \right]. \qquad (16.5)$$

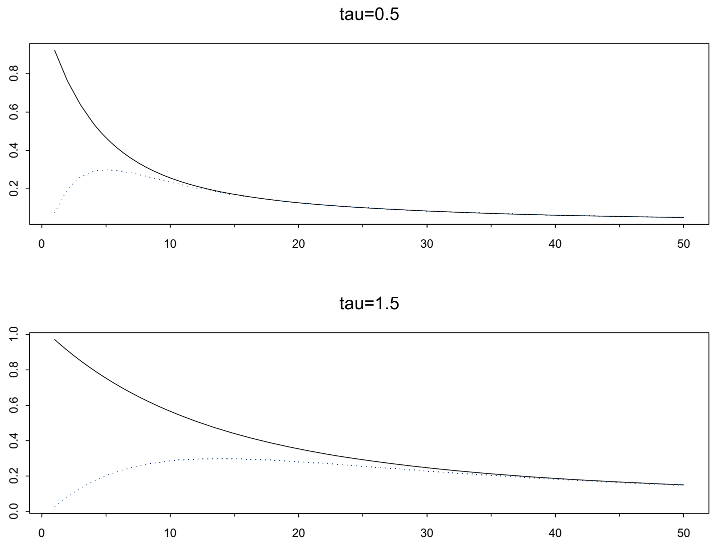

FIGURE 16.7. Short term and medium term components of Nelson-Siegel function.

For a given constant τ, both the forward rate curve and the yield curve are linear functions of the coefficients β_0, β_1 and β_2. Nelson and Siegel (1987) showed that, depending on the values of β_1 and β_2, the yield curve can assume the common shapes of observed yield curves, such as upward sloping, downward sloping, humped, or inverted humped. In addition, consistent with stylized facts of the yield curve, the three components in (16.5) can be interpreted as the long term, short term and medium term component, or the level, slope, and curvature component of the yield curve.[4]

Example 115 *Interpretation of Nelson-Siegel function*

The function `term.struct.nsx` in `S+FinMetrics` can be used to generate the regressors in (16.5) given a vector of maturity and a value for τ. Use the following code to visualize these components for different values of τ:

```
> ns.maturity = seq(1/12, 10, length=50)
> ns05 = term.struct.nsx(ns.maturity, 0.5)
> ns15 = term.struct.nsx(ns.maturity, 1.5)
> par(mfrow=c(2,1))
> tsplot(ns05[,2:3], main="tau=0.5")
> tsplot(ns15[,2:3], main="tau=1.5")
```

[4]Refer to Diebold and Li (2002) for a detailed explanation.

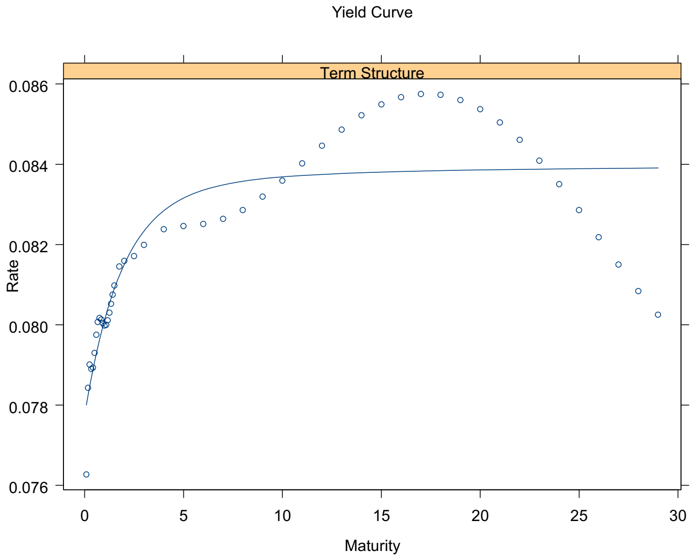

FIGURE 16.8. U.S. yield curve for January 1990: Nelson-Siegel function.

```
> par(mfrow=c(1,1))
```

A vector of maturity was created from one month to ten years. The regressor matrix has three columns, and only the last two columns were plotted because the first column is always one, and the plot is shown in Figure 16.7. The parameter τ controls the rate of decay of those components. When τ is smaller, the short and medium term components decay to zero at a faster rate. Asymptotically, both the short and medium term components approach zero, and thus β_0 can be interpreted as the long term component, or the level of the yield curve.

To interpolate yield curves using the Nelson-Siegel function, choose the value of τ which gives the best fit for equation (16.5). The `term.struct` function employs this procedure if the optional argument `method` is set to `"ns"`. For example, use the following command to interpolate the yield curve for January 1990:

```
> ns.fit = term.struct(mk.zero2[54,], mk.maturity,
+ method="ns", input="spot", na.rm=T)
> ns.fit

Call:
term.struct(rate = mk.zero2[54,  ], maturity = mk.maturity,
      method = "ns", input.type = "spot", na.rm = T)
```

```
Coefficients:
      b0      b1      b2
  0.0840 -0.0063  0.0044

Degrees of freedom: 48 total; 45 residual
Residual standard error: 0.001203026
Tau estimate: 1.7603
```

Again, the fit is plotted by default as shown in Figure 16.8. The graph shows that although the Nelson-Siegel generally captures the shape of the yield curve, the in-sample fit is usually not as good as the non-parametric spline methods because it only uses three coefficients. The output shows the estimates of those coefficients, along with the estimate of τ.

Since the Nelson-Siegel function does not fit the data very well when the yield curve has a rich structure as in the above example, Svensson (1994) proposed to extend the Nelson-Siegel forward function as follows:

$$f(m) = \beta_0 + \beta_1 e^{-m/\tau_1} + \beta_2 \cdot m/\tau_1 \cdot e^{-m/\tau_1} + \beta_3 \cdot m/\tau_2 \cdot e^{-m/\tau_2}$$

which adds another term to the Nelson-Siegel function to allow for a second hump. The corresponding yield function can be shown to be:

$$y(m) = \beta_0 + \beta_1 \frac{1 - e^{-m/\tau_1}}{m/\tau_1} + \beta_2 \left[\frac{1 - e^{-m/\tau_1}}{m/\tau_1} - e^{-m/\tau_1} \right] + \beta_3 \left[\frac{1 - e^{-m/\tau_2}}{m/\tau_2} - e^{-m/\tau_2} \right]. \quad (16.6)$$

To use the above function for interpolating yield curve, simply call the function `term.struct` with `method="nss"`:

```
> nss.fit = term.struct(mk.zero2[54,], mk.maturity,
+ method="nss", input="spot", na.rm=T)
> nss.fit

Call:
term.struct(rate = mk.zero2[54,  ], maturity = mk.maturity,
      method = "nss", input.type = "spot", na.rm = T)

Coefficients:
     b0     b1     b2     b3
 0.0000 0.0761 0.1351 0.0104

Degrees of freedom: 48 total; 44 residual
Residual standard error: 0.0005997949
Tau estimate: 21.8128 0.5315
```

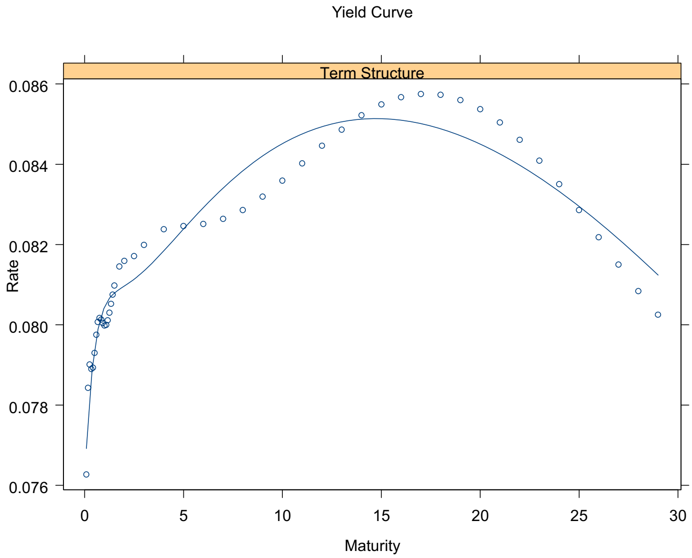

FIGURE 16.9. U.S. yield curve for January 1990: Svensson function.

The output now shows two estimates for τ and one more coefficient for the additional term. The plot of the interpolated yield curve is shown in Figure 16.9.

16.6 Conclusion

For all the term structure interpolation methods discussed in this chapter, they all work with the yield curve for a given time, and thus do not consider the time series aspect of the yield curve. Recently Diebold and Li (2002) considered estimating the three components β_0, β_1 and β_2 of the Nelson-Siegel function for each available time, and building a time series model (in particular, an AR(1)-GARCH(1,1) model) for the estimated β_0, β_1 and β_2. By employing the times series forecasts of β_0, β_1 and β_2, they are able to generate reliable forecasts of yield curve. However, in this approach, the coefficients β_0, β_1 and β_2 are still estimated ignoring the time series aspect.

In recent years, many researchers have proposed to use state space models and Kalman filter to estimate the term structure of interest rates using a panel data, for example, see Duan and Simonato (1999), Geyer and Pichler (1999), Babbs and Nowman (1999), de Jong and Santa-Clara (1999) and de Jong (2000). Most of these models are special cases of the affine term structure model proposed by Duffie and Kan (1996), which can be readily expressed in a state space model by discretizing the continuous-time models. These models can be easily implemented using the state space modeling functions in `S+FinMetrics` as illustrated in Zivot, Wang and Koopman (2004).

16.7 References

Babbs, S. H., and Nowman, K. B. (1999). "Kalman Filtering of Generalized Vasicek Term Structure Models," *Journal of Financial and Quantitative Analysis*, 34(1), 115-129.

Bliss, R. R. (1997). "Testing Term Structure Estimation Methods", in P. Boyle, G. Pennacchi, and P. Ritchken (eds.), *Advances in Futures and Options Research*, Volume 9. Elsevier, Amsterdam, pp. 197-231.

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

de Jong, F. (2000). "Time Series and Cross Section Information in Affine Term-Structure Models," *Journal of Business and Economic Statistics*, 18(3), 300-314.

de Jong, F., and Santa-Clara, P. (1999). "The Dynamics of the Forward Interest Rate Curve: a Formulation with State Variables," *Journal of Financial and Quantitative Analysis*, 34(1), 131-157.

Diebold, F. X., and Li, C. (2003). "Forecasting the Term Structure of Government Bond Yields," NBER Working Paper No. 10048.

Duan, J.-C., and Simonato, J. (1999). "Estimating and Testing Exponential-Affine Term Structure Models by Kalman Filter," *Review of Quantitative Finance and Accounting*, 13, 111-135.

Duffie, D., and Kan, R. (1996). "A Yield-Factor Model of Interest Rates," *Mathematical Finance*, 6(4), 379-406.

Ferguson, R., and Raymar, S. (1998). "A Comparative Analysis of Several Popular Term Structure Estimation Models," *Journal of Fixed Income*, March 1998, 17-33.

Fisher, M., Nychka, D., and Zervos, D. (1995). "Fitting the Term Structure of Interest Rates with Smoothing Splines," Finance and Economics Discussion Series #1995-1, Board of Governors of the Federal Reserve System.

Geyer, A. L. J., and Pichler, S. (1999). "A State-Space Approach to Estimate and Test Multifactor Cox-Ingersoll-Ross Models of the Term Structure," *Journal of Financial Research*, 22(1), 107-130.

Hastie, T. J. (1993). "Generalized Additive Models," in J. M. Chambers and T. J. Hastie (eds.), *Statistical Models in S*. Chapman & Hall.

Hull, J. C. (1997). *Options, Futures, and Other Derivatives*, Prentice Hall, New York.

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

McCulloch, J. H. (1975). "The Tax-Adjusted Yield Curve", *Journal of Finance*, 30(3), 811-830.

McCulloch, J. H., and Kwon, H.-C. (1993). "U.S. Term Structure Data: 1947-1991," Department of Economics, Working Paper #93-6, Ohio State University.

Nelson, C. R., and Siegel, A. F. (1987). "Parsimonious Modeling of Yield Curves," *Journal of Business*, 60(4), 473-489.

Svensson, L. E. O. (1994). "Estimating and Interpreting Forward Interest Rates: Sweden 1992-1994", NBER Working Paper No. 4871.

Zivot, E., Wang, J. and S.J. Koopman (2004). "State Space Models in Economics and Finance Using SsfPack in S+FinMetrics," in A. Harvey, S.J. Koopman, and N. Shephard (eds.), *Unobserved Components Models*. Cambridge University Press, Cambridge.