UNIVERSITY OF ATHENS
Department of Economics

Economics Discussion Reports

A linearly distributed lag estimator with r-convex coefficients

E.E. Vassiliou and I.C. Demetriou

May 2010
No 2010 / 2

Department of Economics
8 Pesmazoglou Street, Athens 10559, Greece
www.econ.uoa.gr

A linearly distributed lag estimator with r-convex coefficients

E. E. Vassiliou[1], I. C. Demetriou[*,1]

Abstract

The purpose of linearly distributed-lag models is to estimate, from time series data, values of the dependent variable by incorporating prior information of the independent variable. A least squares calculation is proposed for estimating the lag coefficients subject to the condition that the rth differences of the coefficients are nonnegative, where r is a prescribed positive integer. Such priors do not assume any parameterization of the coefficients, and in several cases provide such an accurate representation of the prior knowledge, so as to compare favorably to established methods. In particular, the choice of the prior knowledge parameter r gives the lag coefficients interesting special features such as monotonicity, convexity, convexity/concavity, etc. The proposed estimation problem is a strictly convex quadratic programming calculation, where each of the constraint functions depends on $r+1$ adjacent lag coefficients multiplied by the binomial numbers with alternating signs that arise in the expansion of the rth power of $(1-1)$. The most distinctive feature of this calculation is the Toeplitz structure of the constraint coefficient matrix, which allows the development of a special active set method that is faster than general quadratic programming algorithms. Most of this efficiency is due to reducing the equality-constrained minimization calculations, which occur during the quadratic programming iterations, to unconstrained minimization ones that depend on much fewer variables. Some examples with real and simulated data are presented in order to illustrate this approach.

Key words: Almon polynomial, approximation, consumption, difference, distributed-lag model, least squares, r-convexity, regression, quadratic programming, time series, Toeplitz matrix

1. Introduction

The purpose of distributed-lag models is to estimate, from time series data, values y that incorporate prior information of the independent variable x. Specifically, the data are the pairs $(x_t, y_t), t = 1, 2, \ldots, n+m-1$, where we assume that y_t depends not only on x_t, but also on $m-1$ past values of x_t, where m is a prescribed positive number. In econometric and engineering applications (see, for instance, [10], [12], [18], [25], [26], [33], [44]) a linearly distributed-lag model of length m is defined as

$$y_t = \sum_{i=1}^{m} \beta_i x_{t-i+1} + \epsilon_t, \quad t \geq m, \tag{1}$$

where $\beta_1, \beta_2, \ldots, \beta_m$ are the unknown lag coefficients and ϵ_t is a random variable with zero mean and constant variance. For example, one may wish to model consumption expenditures (i.e. y_t) with respect to changes of income (i.e. x_t) over time. Another example from political science is to model reciprocity and enduring rivalries over some periods.

Distributed-lag modeling refers to only the last n observations of $y_t, t = 1, 2, \ldots, m-1, m, \ldots, m+n-1$, because $m-1$ degrees of freedom are lost due to (1). Further, the issue of the m selection depends on the data and may be decided with statistical means (see, for example, [28]:p.119). Adopting matrix notation, the unconstrained lag-distribution problem is to determine a vector $\underline{\beta}^T = (\beta_1, \beta_2, \ldots, \beta_m)$ that minimizes the objective function

$$F(\underline{\beta}) = (\underline{y} - \mathbf{X}\underline{\beta})^T (\underline{y} - \mathbf{X}\underline{\beta}), \tag{2}$$

[*]Corresponding author. Tel.: +302103689817; Fax: +302103689809
Email addresses: evagvasil@econ.uoa.gr (E. E. Vassiliou), demetri@econ.uoa.gr (I. C. Demetriou)
[1]Unit of Mathematics and Informatics, Department of Economics, University of Athens, 8 Pesmazoglou street, Athens 10559, Greece

April 11, 2010

where $\underline{y}^T = (y_m, y_{m+1}, \ldots, y_{m+n-1})$ and $\mathbf{X}$ is the $n \times m$ matrix of current and lagged values of x_t defined as

$$\mathbf{X} = \begin{pmatrix} x_m & x_{m-1} & x_{m-2} & \cdots & x_1 \\ x_{m+1} & x_m & x_{m-1} & \cdots & x_2 \\ x_{m+2} & x_{m+1} & x_m & \cdots & x_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{m+n-1} & x_{m+n-2} & x_{m+n-3} & \cdots & x_n \end{pmatrix}.$$

In the development of suitable algorithms for unconstrained least squares estimation of $\underline{\beta}$, one important consideration is that often there is high multicollinearity among the x_t's giving a notorious ill-posed inverse problem (for general references see, for example, [20], [43], [45]). Moreover, in the usual case of presence of errors of measurement in the y_t's, the resulted unconstrained estimates are corrupted by distortions. Often, however, there is a discernible trend in these estimates (which should depend on the choice of m) that may be helpful in considering some smoothness priors (see example in Section 4).

There have been several suggestions in the econometric literature to put some structure on the β_i's in (1). They all impose some a priori structure on the form of the lag, in order to combine prior and sample information in the estimation of the regression coefficients. A very popular approach is the Almon model [1], which assumes that the lag coefficients lie on a polynomial of degree $r-1$, where r is a prescribed integer. Shiller's method [41], as a variant of this model, assumes that the lag coefficients lie close to, rather than on, a polynomial. Jorgenson's method [24] approximates the lag distribution by the ratio of two polynomials. More models are found in [9], [21], [22], [29], [31], [33], [36], [42], etc, while some of them are implemented in software packages (see, for instance, [40]). All these models assume that the underlying function of the lag coefficients can be approximated closely by a form that depends on only a few parameters. However, over the years, literature on the subject agrees that some weak representation of the lag coefficients is a sensible requirement for a satisfactory model estimation (see, for example, [15], [19], [21], [33] and references therein).

In this paper a method is suggested that seeks lag coefficients $\beta_1, \beta_2, \ldots, \beta_m$ such that the rth differences (see, for example, [23])

$$\Delta_j^r \underline{\beta} = \frac{1}{r!} \sum_{i=j}^{j+r} (-1)^{r-i+j} \binom{r}{i-j} \beta_i, \quad j = 1, 2, \ldots, m-r \tag{3}$$

are nonnegative, where r is much smaller than m. Specifically the method minimizes (2) subject to the conditions

$$\Delta_j^r \underline{\beta} \geq 0, \quad j = 1, 2, \ldots, m-r, \tag{4}$$

but it may well be applied for the case where the differences $\Delta_j^r \underline{\beta}$, $j = 1, 2, \ldots, m-r$ are non-positive. Ideally, the fitted function of the lag coefficients is to have a nonnegative rth derivative. Functions like this are called r-convex (see [27] for their fundamental role in total positivity) and we analogously call r-convex a vector whose components satisfy the constraints (4).

If (4) are exactly satisfied, then our model is identical to Almon's polynomial of degree $r-1$. However, our model, due to the r-convexity properties of $\underline{\beta}$, allows some constraints to be amply satisfied, which is equivalent to relaxing some differences in Almon's polynomial. Hence our approximation is less tight than that of Almon's, providing a weaker structure on the lag coefficients. In order to answer the question on the nature of the r-convex vector $\underline{\beta}$, we consider these indices of the estimated components of $\underline{\beta}$, such that $\Delta_j^r \underline{\beta} \neq 0$, and seek the intervals on which the components of $\underline{\beta}$ satisfy the equations $\Delta_j^r \underline{\beta} = 0$. Let σ and τ be any indices such that $1 \leq \sigma < \tau \leq m$. If $\Delta_j^r \underline{\beta} = 0$, for $j = \sigma, \sigma+1, \ldots, \tau-m$, if $\sigma = 1$ or $\Delta_{\sigma-1}^r \underline{\beta} \neq 0$ and if $\tau = m$ or $\Delta_{\tau-r+1}^r \underline{\beta} \neq 0$, then $\beta_\sigma, \beta_{\sigma+1}, \ldots, \beta_\tau$ are interpolated by a polynomial of degree at most $r-1$. Then the estimated values $\{\beta_i : i = 1, 2, \ldots, m\}$ lie on a piecewise polynomial approximation, where the polynomial pieces are of degree at most $r-1$. The polynomial pieces usually overlap, while their breaks are found automatically by the optimization calculation. This not only makes our technique more flexible than the Almon polynomial, but also provides certain advantages over the spline distributed-lag method of [3], where the breakpoints have to be specified in advance.

The approach presented here may be generally applied to a variety of situations in which one knows some properties of an underlying relation, but one does not have sufficient information to put the relation into any simple parametric

$$\mathbf{D}_4 = \begin{bmatrix}
1 & & & & & & & & & & \\
-4 & 1 & & & & & & & & & \\
6 & -4 & 1 & & & & & & & & \\
-4 & 6 & -4 & 1 & & & & & & & \\
1 & -4 & 6 & -4 & 1 & & & & & & \\
 & 1 & -4 & 6 & -4 & 1 & & & & & \\
 & & 1 & -4 & 6 & -4 & 1 & & & & \\
 & & & 1 & -4 & 6 & -4 & 1 & & & \\
 & & & & 1 & -4 & 6 & -4 & 1 & & \\
 & & & & & 1 & -4 & 6 & -4 & 1 \\
 & & & & & & 1 & -4 & 6 & -4 \\
 & & & & & & & 1 & -4 & 6 \\
 & & & & & & & & 1 & -4 \\
 & & & & & & & & & 1
\end{bmatrix}$$

Figure 1: Matrix $\mathbf{D}_r$, when $m = 14$ and $r = 4$

form. Two immediate advantages are gained by this approach to lag estimation. The first one is that it avoids the assumption that the relation has a form that depends of a few parameters, which occurs in many other techniques as already being mentioned. The second one is that it obtains properties (i.e. r-convexity, see [4]) that occur in a variety of underlying relations of the lag coefficients, which depend on the value of r, such as monotonicity, convexity, concavity, convexity/concavity (see [6], [8], [39]) etc. For the particular case $r = 1$ we obtain the monotonically increasing components

$$\beta_1 \leq \beta_2 \leq \cdots \leq \beta_m, \tag{5}$$

which are useful if x_t in (1) becomes less significant as time proceeds. In many real cases, as in production situations, the case $r = 2$ in (4) may be considered. Now the lag-coefficients are subject to the increasing returns $\beta_2 - \beta_1 \leq \beta_3 - \beta_2 \leq \cdots \leq \beta_m - \beta_{m-1}$, which is equivalent in assuming that $\{\beta_i : i = 1, 2, \ldots, m\}$ satisfy the convexity conditions (i.e. they lie on a convex curve)

$$\beta_{i+2} - 2\beta_{i+1} + \beta_i \geq 0, \quad i = 1, 2, \ldots, m-2. \tag{6}$$

Respectively, the concavity conditions (cf. decreasing returns) are

$$\beta_{i+2} - 2\beta_{i+1} + \beta_i \leq 0, \quad i = 1, 2, \ldots, m-2. \tag{7}$$

Further, when the lag-coefficients follow a sigmoid trend or the underlying relation shows an inflection point and away from this the underlying relation seems to be convex and concave, then it would be suitable to allow $r = 3$ in (4), which gives the conditions

$$\beta_{i+3} - 3\beta_{i+2} + 3\beta_{i+1} - \beta_i \geq 0, \quad i = 1, 2, \ldots, m-3. \tag{8}$$

With these templates at hand we express the constraints (4) in the matrix form

$$\mathbf{D}_r^T \underline{\beta} \geq 0, \tag{9}$$

where $\mathbf{D}_r$ is the $m \times (m-r)$ rectangular matrix, whose elements $(\mathbf{D}_r)_{ij}$ are defined by the relation

$$(\mathbf{D}_r)_{ij} = \begin{cases} (-1)^{r+j-i} \binom{r}{i-j}, & j \leq i \leq j+r, \quad j = 1, 2, \ldots, m-r \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Now the nonzero components of the jth column of $\mathbf{D}_r$ are a constant multiple of the terms that occur in the differences (3) giving a Toeplitz pattern that depends on the value of r. For instance, we let $m = 14$ and display $\mathbf{D}_r$ in Figs. 1 and

$$\mathbf{D}_5 = \begin{bmatrix}
-1 & & & & & & & & \\
5 & -1 & & & & & & & \\
-10 & 5 & -1 & & & & & & \\
10 & -10 & 5 & -1 & & & & & \\
-5 & 10 & -10 & 5 & -1 & & & & \\
1 & -5 & 10 & -10 & 5 & -1 & & & \\
 & 1 & -5 & 10 & -10 & 5 & -1 & & \\
 & & 1 & -5 & 10 & -10 & 5 & -1 & \\
 & & & 1 & -5 & 10 & -10 & 5 & -1 \\
 & & & & 1 & -5 & 10 & -10 & 5 \\
 & & & & & 1 & -5 & 10 & -10 \\
 & & & & & & 1 & -5 & 10 \\
 & & & & & & & 1 & -5 \\
 & & & & & & & & 1
\end{bmatrix}$$

Figure 2: Matrix $\mathbf{D}_r$, when $m = 14$ and $r = 5$

2, for $r = 4$ and 5 respectively. General references to Toeplitz matrices may be found, for example, in [2], [17] and [34].

Since the constraints on $\underline{\beta}$ are linear and consistent and since the second derivative matrix of (2) with respect to $\underline{\beta}$ is twice the matrix $\mathbf{X}^T\mathbf{X}$, the problem of minimizing (2) subject to (9) is a convex quadratic programming problem. Given that $\mathbf{X}^T\mathbf{X}$ is positive semidefinite, there is a global solution and if $\mathbf{X}^T\mathbf{X}$ is positive definite the solution is unique, $\underline{\hat{\beta}}$ say. Of course, it is easy to construct a vector $\underline{x}$ such that some row of matrix $\mathbf{X}$ is a multiple of another row, but dependencies are readily detected in the algorithms. In this paper, it is assumed that $\mathbf{X}^T\mathbf{X}$ is positive definite. Therefore it is usually straightforward to calculate $\underline{\hat{\beta}}$ by standard quadratic programming methods (see, for example, [11], [13] and [30]), but careful attention is given to solving this problem efficiently by taking account of the Toeplitz structure of the constraint gradients for the various values of r.

Our model, as defined by (1) and the minimization of (2) subject to (9), is an instance of the general inequality constrained least-squares regression model of [32] that uses prior and sample information for parameter estimation. Sampling and statistical properties as well as some important questions of the general problem that reflect on our special problem are considered by [10] and [32], but they are beyond the scope of our paper.

In Section 2 we present and consider the method that is used. The method employs a quadratic programming algorithm, which is variant of the algorithms of [8] and [16]. The main feature of this method is a suitable transformation in the linear space of variables, which is used to convert the equality-constrained subproblems that arise during the quadratic programming iterations into unconstrained ones. Further, the calculation of the Lagrange multipliers becomes very efficient by making use of some methods that have been developed independently by [7]. The latter two calculations take advantage of a common submatrix that occurs during each iteration of the quadratic programming algorithm. Further, we compare the efficiency of our method with those in [16] and [14]. In Section 3 we present results from a simulation that demonstrate the performance of our method. In Section 4 we report results of our method on real data and compare with the method of Almon. In Section 5 we present some concluding remarks. A Fortran version of our method has been written by one of the authors (EEV). It consists of about 1300 lines including comments.

2. The method of calculation of the lag coefficients

This section develops a method for solving the problem of Section 1 that depends on the Karush-Kuhn-Tucker optimality conditions (see, for example, [35]). They state that $\underline{\beta} = \underline{\hat{\beta}}$ if and only if the constraints (9) are satisfied and

there exist Lagrange multipliers $\lambda_i \geq 0, i \in \mathcal{A}$ such that the equation

$$2\mathbf{X}^T(\mathbf{X}\underline{\beta} - \underline{y}) = \sum_{i \in \mathcal{A}} \lambda_i \underline{a}_i, \tag{11}$$

holds, where $\mathcal{A}$ is the subset $\{i : \Delta_i^r \beta = 0\}$ of constraint indices and $\underline{a}_i \in \mathbb{R}^m$ is the gradient of $\Delta_i^r \beta$ by a constant with respect to $\underline{\beta}$. We define, $\lambda_i = 0$, for $i \in [1, m-r] \backslash \mathcal{A}$ and denote the $(m-r)$-vector of Lagrange multipliers by $\underline{\lambda}$.

The method employs an extension of the strictly convex quadratic programming calculation of [8], which is based on the primal-dual active set method of [16]. It generates a sequence of subsets of the constraint indices $\{1, 2, \ldots, m-r\}$, where for each subset, $\mathcal{A}$ say, the equations (or active constraints)

$$\underline{a}_i^T \underline{\beta} = 0, \quad i \in \mathcal{A} \tag{12}$$

are satisfied and the vector $\underline{\beta}$ is obtained by minimizing the objective function (2) subject to the equations (12). Moreover, unique Lagrange multipliers $\lambda_i,\ i \in \mathcal{A}$ are defined by the first order optimality condition (11).

The efficiency of these operations depends on the remark that the minimization of function (2) subject to some prescribed differences being zero is reduced to an unconstrained minimization problem with fewer variables. The calculation begins from $\mathcal{A} = \{1, 2, \ldots, m-r\}$. Then it proceeds by dropping constraint indices from $\mathcal{A}$ that correspond to negative Lagrange multipliers, one at a time, until all multipliers become nonnegative. This provides the starting point for the quadratic programming procedure. Quadratic programming generates a finite number of sets of indices of different active constraints, so that at the beginning of an iteration the Lagrange multipliers are all nonnegative. If the associated estimate of $\underline{\beta}$ violates some constraints then the most violated constraint index is added to $\mathcal{A}$, and other constraint indices are removed from $\mathcal{A}$ if necessary, so that the Lagrange multipliers become nonnegative again, which completes an iteration. Termination occurs when the Karush-Kuhn-Tucker conditions are satisfied.

Algorithm 1 (Quadratic programming for the lag coefficients $\beta_i, i = 1, 2, \ldots, m$)
Input: n, m, r and $(x_t, y_t), t = 1, 2, \ldots, m+n-1$
Output: solution $\hat{\underline{\beta}}$ and the associated Lagrange multipliers

Step 0 (Initialization) Set $\mathcal{A} = \{1, 2, \ldots, m-r\}$ and call Algorithm 2 (which is described later) to calculate $\underline{\beta}$ and $\underline{\lambda}$. If any negative multipliers occur, then start removing the corresponding constraint indices from $\mathcal{A}$, one at a time, while recalculating each time $\underline{\beta}$ and $\underline{\lambda}$, until the multipliers become nonnegative.

Step 1 (Testing the constraints) If the constraints (9) are satisfied, then terminate. Otherwise record $\underline{\mu} = \underline{\lambda}$, find the most violated constraint, $\underline{a}_k^T \underline{\beta} < 0$ say, add k to $\mathcal{A}$ and calculate $\underline{\beta}$ and $\underline{\lambda}$ (by Algorithm 2).

Step 2 (Testing the sign of the Lagrange multipliers) If $\lambda_i \geq 0$, $i \in \mathcal{A}$, then branch to Step 1, otherwise proceed to Step 3.

Step 3 (Recovering nonnegativity of the Lagrange multipliers) Seek the greatest value of θ such that the numbers $(1-\theta)\mu_i + \theta\lambda_i$, $i \in \mathcal{A}$ are nonnegative, which implies $0 \leq \theta < 1$. If ρ is the value of i that gives $(1-\theta)\mu_i + \theta\lambda_i = 0$, then remove ρ from $\mathcal{A}$, replace $\underline{\mu}$ by $(1-\theta)\underline{\mu} + \theta\underline{\lambda}$, calculate $\underline{\beta}$ and $\underline{\lambda}$ (by Algorithm 2) and branch to Step 2. ■

Because the number of constraints is finite, Algorithm 1 either terminates or cycles. The following propositions prevent cycling in absence of rounding errors. Step 0 provides the starting point. Step 1 is entered either from Step 0 or from Step 2. Inserting constraints at Step 1 makes the value of the objective function strictly higher than the value it had at the previous occurrence of Step 1. Step 2 is entered from Step 1 or Step 3 and checks the sign of the Lagrange multipliers. Step 3 is entered from Step 2 in order to recover nonnegativity of the Lagrange multipliers. Now both μ and $\underline{\lambda}$ are available, the components of $\underline{\mu}$ are nonnegative and one or more of the components of $\underline{\lambda}$ are negative. The first visit to Step 3 picks an index ρ that is different from k of Step 1. This ρ cannot be in $\mathcal{A}$ on the next visit to Step 1. Deleting constraints at Step 3 decreases the value of the objective function, but keeps it strictly larger than the value it had at the previous instance of Step 2. Hence at least one of the deleted constraint indices cannot be inserted until after the algorithm branches to Step 2. Since Step 2 is usually reached after some insertions and the objective function strictly increases at consecutive instances of Step 2, cycling is impossible in exact arithmetic, so the algorithm terminates. These propositions are proved in [8].

Lemma 1. *Algorithm 1 terminates at the solution of the problem that minimizes the objective function (2) subject to the constraints (9).*

Proof: See Theorem 2 of [8]. ■

An important feature of any active set method concerns efficient solution of the equality constrained minimization problems that occur when changes are made to $\mathcal{A}$. Below in this section we take advantage of the Toeplitz structure of matrix $\mathbf{D}_r$ and address two topics. First, we construct a basis for the linear subspace defined by (12). Second, we present efficient methods for the minimization of function (2) subject to the linear constraints (12) and the calculation of the corresponding Lagrange multipliers from (11).

Since there are no redundant equations in (12), as it may be proved by taking account of the fact that each $\underline{a}_i$ depends on only $r+1$ adjacent components, there are $m-p$ degrees of freedom in the variables $\beta_1, \beta_2, \ldots, \beta_m$, where $p = |\mathcal{A}|$ is the number of elements of $\mathcal{A}$. It follows that we can find $m-p$ linearly independent m-vectors $\{\underline{u}_s : s \in S\}$, for some $S \subseteq \{1, 2, \ldots, m\}$ to be defined later, such that $\underline{a}_j^T \underline{u}_s = 0, s \in S$, for all $j \in \mathcal{A}$. Now for any vector $\underline{\beta}$ that satisfies (12) there exist $m-p$ real numbers $\{\theta_i : i = 1, 2, \ldots, m-p\}$ such that

$$\underline{\beta} = \sum_{i=1}^{m-p} \theta_i \underline{u}_{\sigma(i)}, \tag{13}$$

where we let $\sigma(1), \sigma(2), \ldots, \sigma(m-p)$ be the elements of S in ascending order.

Because $\mathcal{A}$ is usually kept large during the iterations of Algorithm 1, working with the $m-p$ variables $\{\theta_i : i = 1, 2, \ldots, m-p\}$ instead of with $\{\beta_i : i = 1, 2, \ldots, m\}$ is advantageous in that there are fewer variables and that the equations (12) are satisfied automatically. Suppose that a basis $\{\underline{u}_s : s \in S\}$ is known and let $\mathbf{U}$ be the $m \times (m-p)$ matrix whose columns are the basis elements $\underline{u}_s$. Then we substitute (13) into (2) and we obtain the reduced quadratic function

$$\psi(\underline{\theta}) = \|\mathbf{X}\mathbf{U}\underline{\theta} - \underline{y}\|_2^2, \tag{14}$$

where $\underline{\theta}^T = (\theta_1, \theta_2, \ldots, \theta_{m-p})$. Since the second derivative matrix with respect to $\underline{\theta}$ of function (14) is the $(m-p) \times (m-p)$ positive definite matrix $(\mathbf{X}\mathbf{U})^T(\mathbf{X}\mathbf{U})$, a unique minimizer of $\psi(\underline{\theta})$ exists and is obtained by applying Cholesky factorization to the first order condition of (14), namely the normal equations

$$(\mathbf{X}\mathbf{U})^T(\mathbf{X}\mathbf{U})\underline{\theta} = (\mathbf{X}\mathbf{U})^T\underline{y}. \tag{15}$$

As p in practice is close to $m-r$, $\underline{\theta}$ is obtained by solving a $r \times r$ (about) system, where r is a small number. Thus, this procedure provides a stable and economical calculation for obtaining $\underline{\theta}$. Then $\underline{\beta}$ is derived by substituting $\underline{\theta}$ into (13).

The question is whether we can find a suitable basis for the linear subspace of vectors $\underline{\beta}$ defined by the active constraints (12) or equivalently by the equations $\Delta_i^r \beta = 0, i \in \mathcal{A}$. Motivated by [4], we develop a basis, which is a convenient and very efficient choice for our problem, because it is naturally defined from the non-active constraints throughout the calculation and because the matrices that appear are banded and positive definite. The rest of the section includes technical details for defining and calculating this basis as well as for calculating the Lagrange multipliers.

Having in mind the paragraph that follows relation (4), we associate a break in the mentioned piecewise polynomial approximation with the central coefficient (there are two such coefficients with opposite sign, if r is odd) of a non-active difference. Thus, let $K = \{1, 2, \ldots, m-r\} \backslash \mathcal{A}$ be the set of indices of constraints that give $\underline{a}_j^T \underline{\beta} \neq 0, 1 \leq j \leq m-r$, let $\bar{q}$ be the least integer such that $2\bar{q} > r$ and let $q = \bar{q} - 1$. Then the indices of the central coefficients of non-active differences are

$$\{k + q : k \in K\}$$

and they are are going to be associated with $|K| = m - r - p$ basis elements. The additional r basis elements derive from those additional q indices at the left end and $r-q$ indices at the right end of the interval $[1+q, m-r+q]$, as they are specified by the the index set

$$Q = \{1, \ldots, q\} \cup \{m - r + q + 1, \ldots, m\}.$$

We also let

$$S = \{k + q : k \in K\} \cup Q.$$

$$\mathbf{M}_4 = \begin{bmatrix} 6 & 1 & & & & \\ 1 & 6 & -4 & 1 & & \\ & -4 & 6 & -4 & & \\ & 1 & -4 & 6 & & \\ & & & & 6 & -4 \\ & & & & -4 & 6 \end{bmatrix}$$

Figure 3: Matrix $\mathbf{M}_r$ associated with $\mathbf{D}_r$, when $m = 14$, $r = 4$ and $\mathcal{A} = \{1, 3, 4, 5, 8, 9\}$. $\mathbf{D}_r$ is displayed in Fig. 1

.

Then we define the vectors $\underline{u}_s,\ \text{for } s \in S$ as a solution to the equations

$$(\underline{u}_s)_t = \delta_{st}, \quad \text{for } s, t \in S \tag{16}$$

and

$$\underline{a}_i^T \underline{u}_s = 0, \quad \text{if } i \in \mathcal{A} \text{ and } s \in S, \tag{17}$$

where δ_{st} is the Kronecker's delta, and we prove that they form a basis.

Lemma 2. *The vectors $\underline{u}_s$, for $s \in S$, which are defined by (16) and (17), form a basis for the $m - p$ dimensional subspace of the vectors $\underline{\beta}$ that satisfy the equality constraints (12).*

Proof: Let $\eta_s, s \in S$ be scalars such that

$$\sum_{s \in S} \eta_s \underline{u}_s = \underline{0}. \tag{18}$$

Considering the components t, $t \in S$ of equation (18), in view of the definition (16), gives immediately $\eta_s = 0$, for all $s \in S$. Hence, the vectors $\underline{u}_s,\ s \in S$ are linearly independent. Because of (17), these vectors belong to the subspace that satisfies the active constraints (12). Since $\{k + q :\ k \in K\} \cap Q = \emptyset$, the vectors $\underline{u}_s,\ s \in S$ are all different. Moreover, $|K| = m - r - p$ and $|S| = |K| + |Q| = (m - r - p) + r = m - p$, so the dimension of the subspace under consideration is $m - p$. ■

We remind that $\{i + q : i \in \mathcal{A}\} \cap \{k + q : k \in K\} = \emptyset$, namely that $\mathcal{A}$ and K have no common elements and we turn to the calculation of the basis elements. Since the $m - p$ components $\{(\underline{u}_s)_t,\ t \in S\}$ of $\underline{u}_s$ are defined by (16) and since $\{i + q : i \in \mathcal{A}\} \cup S = \{1, 2, \ldots, m\}$, the remaining p components are $\{(\underline{u}_s)_{i+q},\ i \in \mathcal{A}\}$. These components are obtained by solving a $p \times p$ system of equations, whose coefficient matrix elements are derived by deleting a column of the $p \times m$ coefficient matrix of (17) for each $s \in S$ (we shall see later that the transpose of this reduced matrix occurs also in the calculation of the Lagrange multipliers) and whose right hand side is own to (16). Specifically, we let $\alpha(1), \alpha(2), \ldots, \alpha(p)$ be the elements of $\mathcal{A}$ in ascending order and obtain the reduced $p \times p$ matrix $\tilde{\mathbf{D}}_r$ by

$$(\tilde{\mathbf{D}}_r)_{cd} = (\mathbf{D}_r)_{\alpha(c)+q, \alpha(d)},\ 1 \le c, d \le p. \tag{19}$$

Hence and in view of (17) and (16), the components $\{(\underline{u}_s)_{i+q},\ i \in \mathcal{A}\}$ for a specific $s \in S$ can be found by solving the system

$$\sum_{d=1}^{p} (\tilde{\mathbf{D}}_r^T)_{cd} (\underline{u}_s)_{\alpha(d)+q} = (\underline{b}_s)_c,\ 1 \le c, d \le p, \tag{20}$$

where vector $\underline{b}_s$, $s \in S$, due to (16), is the negative of the sth column of the coefficient matrix of system (17).

Motivated by the sign pattern of $\mathbf{D}_r$ (as, for instance, it is exhibited by Figs. 1 and 2), we claim that $\tilde{\mathbf{D}}_r$ is positive definite if $r \bmod 4 = 0, 3$ and negative definite if $r \bmod 4 = 1, 2$. In order to prove this claim we set

$$(\mathbf{M}_r)_{cd} = \begin{cases} (\tilde{\mathbf{D}}_r)_{\alpha(c)+q, \alpha(d)}, & \text{if } r \bmod 4 = 0, 3 \\ -(\tilde{\mathbf{D}}_r)_{\alpha(c)+q, \alpha(d)}, & \text{if } r \bmod 4 = 1, 2 \end{cases} \quad 1 \le c, d \le p, \tag{21}$$

$$\mathbf{M}_5 = \begin{bmatrix} 10 & 1 & & & & \\ 5 & 10 & -5 & 1 & & \\ -1 & -10 & 10 & -5 & & \\ & 5 & -10 & 10 & & \\ & & & -1 & 10 & -5 \\ & & & & -10 & 10 \end{bmatrix}$$

Figure 4: As in Fig. 3, but $r = 5$. $\mathbf{D}_r$ is displayed in Fig. 2

.

while, by way of illustration, in Figs. 3 and 4 we see $\mathbf{M}_r$ when $m = 14$, $r = 4, 5$ and $\mathcal{A} = \{1, 3, 4, 5, 8, 9\}$. Then we define the $(m-r) \times (m-r)$ matrix $\mathbf{L}_r$ by

$$(\mathbf{L}_r)_{ij} = \begin{cases} (\mathbf{D}_r)_{i+q,j}\,, & \text{if } r \bmod 4 = 0, 3 \\ -(\mathbf{D}_r)_{i+q,j}\,, & \text{if } r \bmod 4 = 1, 2 \end{cases} \quad 1 \le i, j \le m - r, \tag{22}$$

which has the Toeplitz property

$$(\mathbf{L}_r)_{ij} = (\mathbf{L}_r)_{i+1,j+1}, \ 1 \le i, j \le m - r - 1,$$

as it follows immediately from the Toeplitz structure of $\mathbf{D}_r$. Furthermore, it has been proved by [7] that if r is an even integer, then $\mathbf{L}_r$ is a symmetric positive definite Toeplitz matrix, and if r is an odd integer, then $\mathbf{L}_r$ is a nonsymmetric positive definite Toeplitz matrix. Since

$$(\mathbf{M}_r)_{cd} = (\mathbf{L}_r)_{\alpha(c),\alpha(d)}, 1 \le c, d \le p, \tag{23}$$

matrix $\mathbf{M}_r$ consists of the elements from the intersection of the rows and columns of $\mathbf{L}_r$, whose elements are given in $\mathcal{A}$. In other words, $\mathbf{M}_r$ is a principal submatrix of $\mathbf{L}_r$. Given the positive definiteness of $\mathbf{L}_r$, the positive definiteness of $\mathbf{M}_r$ follows for any r (see, for example, [17]: p.141).

Moreover, in view of the band structure of $\mathbf{D}_r$ and (22), it follows that $(\mathbf{L}_r)_{ij} = 0$, unless $q - r \le j - i \le q$. Hence $\mathbf{L}_r$ is a banded matrix and since $\mathbf{M}_r$ is a principal submatrix of $\mathbf{L}_r$, it is a banded matrix too. Since the Toeplitz property of $\mathbf{L}_r$ is not inherited by $\mathbf{M}_r$, we can make use of only the positive definiteness of $\mathbf{M}_r$ when solving (20). Thus, (20) is solved for the components $\{(\underline{u}_s)_{i+q},\ i \in \mathcal{A}\}$ efficiently and stably by Cholesky factorization if r is even and by band LU factorization if r is odd. Since this process has to be repeated for each $s \in S$ in order to generate *all* the basis elements, we factorize $\mathbf{M}_r$ only once and subsequently use this factorization in solving system (20) (of course, each time, the right hand side $\underline{b}_s$ changes). Hence the method for generating the basis elements $\{\underline{u}_s : s \in S\}$ for the linear space defined by (12) is complete.

Once β is available, the corresponding Lagrange multipliers $\{\lambda_i : i \in \mathcal{A}\}$ are obtained by the first order conditions (11), which represent an overdetermined system with $m - p$ redundant equations. So p equations may be chosen in order to specify the p unknowns λ_i. All possible choices will give the same solution, provided the chosen system is nonsingular. Relying on the work of [7], it is remarkable that the choice of p out of the m equations (11) that resulted to matrix (21) is suitable to the calculation of the Lagrange multipliers as well. Specifically, we observe that the central element (there are two such elements with opposite sign, if r is odd) of each column of $\mathbf{D}_r$ is also the largest in absolute value element of each column and consequently it suffices to let the $p \times p$ matrix $\mathbf{M}_r$ be defined by (21) and define also the p-vectors $\underline{\tilde{\lambda}}$ and $\underline{\tilde{b}}$ by

$$\tilde{\lambda}_d = \lambda_{\alpha(d)} \quad \text{and} \quad \tilde{b}_c = 2[\mathbf{X}^T(\mathbf{X}\underline{\beta} - \underline{y})]_{\alpha(c)+q}, \quad 1 \le c, d \le p.$$

Hence in view of (11), we derive the positive definite system of equations

$$\sum_{d=1}^{p} \tilde{\lambda}_d (\mathbf{M}_r)_{cd} = \tilde{b}_c, \quad 1 \le c \le p. \tag{24}$$

Since the transpose of $\mathbf{M}_r$ is the coefficient matrix of system (20), its factorization is already available from the calculation that provided $\{(\underline{u}_s)_{i+q}, i \in \mathcal{A}\}$.

The following algorithm implements the procedures described in this section for the calculation of $\underline{\beta}$ and $\underline{\lambda}$ for each $\mathcal{A}$.

Algorithm 2 (Determining $\underline{\beta}$ and $\underline{\lambda}$ for each $\mathcal{A}$)
Input: $\mathcal{A}$
Output: $\underline{\beta}$, namely the solution of the problem that minimizes (2) subject to the equality constraints (12), and $\underline{\lambda}$, namely the corresponding Lagrange multipliers that satisfy (11)

Step 1 After replacing $\tilde{\mathbf{D}}_r$ by $\mathbf{M}_r$ in (20), solve (20) for each $s \in S$, and obtain the components $\{(\underline{u}_s)_{i+q}, i \in \mathcal{A}\}$ as follows: if r is even, then apply band Cholesky factorization to $\mathbf{M}_r$, and if r is odd, then apply band LU factorization to $\mathbf{M}_r$.

Step 2 Form $(\mathbf{XU})^T(\mathbf{XU})$ and solve (15) for $\underline{\theta}$ by Cholesky factorization of $(\mathbf{XU})^T(\mathbf{XU})$. The required vector $\underline{\beta}$ that minimizes (2) subject to (12) is obtained by substituting $\underline{\theta}$ into (13).

Step 3 Determine the Lagrange multipliers $\{\lambda_i : i \in \mathcal{A}\}$, as follows: If r is even, then solve (24) for $\{\lambda_i : i \in \mathcal{A}\}$, by making use of the Cholesky factors of $\mathbf{M}_r$ already available from Step 1, and if r is odd, then solve (24) by making use of the LU factors of $\mathbf{M}_r$ already available from Step 1. ■

Matrix $\mathbf{M}_r$ appears twice in Algorithm 2. First at Step 1, in generating the basis vectors, and second at Step 3, in obtaining the Lagrange multipliers. Step 1 obtains the solution of system (20) for each $s \in S$ in the expense of $O((m - |\mathcal{A}|)\,|\mathcal{A}|\,r^2)$ computer operations if r is even, where Cholesky factorization is applied to $\mathbf{M}_r$, and in $O((m - |\mathcal{A}|)\,|\mathcal{A}|\,q(r-q))$, where $q \simeq \frac{r}{2}$, if r is odd, by LU factorization of $\mathbf{M}_r$. Step 2 requires $O(m(m-|\mathcal{A}|)^2)$ operations in order to form the matrix $(\mathbf{XU})^T(\mathbf{XU})$ in (15) and $O((m - |\mathcal{A}|)^3)$ operations in order to solve the $(m - |\mathcal{A}|) \times (m - |\mathcal{A}|)$ system (15). Step 3 obtains the Lagrange multipliers in only $O(|\mathcal{A}|\,r)$ operations, because the factorization of $\mathbf{M}_r$ has already been carried out at Step 1. Thus the amount of work of Algorithm 2 is of the order of $m(m - |\mathcal{A}|)^2 + (m - |\mathcal{A}|)\,|\mathcal{A}|\,r^2$ computer operations, when $m - |\mathcal{A}|$ is a small number.

As mentioned already, a strong reason that favors the calculation of $\underline{\beta}$ by means of a basis in the linear space defined by the equality constraints (12) and solving the normal equations (15) is the large size of set $|\mathcal{A}|$ that usually occurs in practice (see numerical results in Section 3). Indeed, now $m - |\mathcal{A}|$, the order of the normal equations, can be very small, and, although the normal equations can sometimes be ill-conditioned, ill-conditioning in our case is most unlikely, because all calculations for deriving the mentioned basis are based on positive definite systems. Moreover, the sparsity feature of the constraint normals promotes the use of LU and Cholesky factorizations, instead of using orthogonal factorizations.

Further, we elucidate some differences between Algorithm 1 and QPROG and QPSOL, which are two broadly used subroutines for general quadratic programming calculations. QPROG is the implementation of the algorithm of [16] for convex quadratic programming that was developed by [38] and is provided by IMSL. QPSOL is a general purpose subroutine for quadratic programming that was developed by [14] and versions of it are in NAG and Matlab. These subroutines are not aimed at large scale problems; the constraint matrices and the Hessian $\mathbf{X}^T\mathbf{X}$ are specified in dense storage format and all numbers are calculated by updating techniques that deal with full matrices, whose storage requirements are $O(m(m - r))$. The total number of multiplications for QPROG in an iteration that makes one addition to and one deletion from the active set has the value (see [38])

$$W(\text{QPROG}) = m(m - r) + 16m^2/3 - 3m\,|\mathcal{A}| + 5\,|\mathcal{A}|^2/3$$

and the corresponding value for QPSOL is (see [14])

$$W(\text{QPSOL}) = m(m - r) + 13.5m^2 - 22m\,|\mathcal{A}| + 12\,|\mathcal{A}|^2\,.$$

Because of differences in the number of iterations to solve a quadratic programming problem, the complexity of Algorithm 2 and these two expressions provide only a rough guide to the relative efficiencies of Algorithm 1 and

subroutines QPROG and QPSOL. However, besides the advantage that we derive from $O(rm)$ storage, Algorithm 2, due to taking account of constraint sparsity and Toeplitz property, requires much less work in order to carry out the tasks that give W(QPROG) and W(QPSOL). So the efficiency of Algorithm 1 seems to be quite competent and the apparent gain over a general quadratic programming algorithm is due to the fact that our method is very suitable for a large scale calculation.

3. Numerical results

This section presents results from simulation experiments in order to demonstrate the model accuracy and the performance of Algorithm 1. The data were produced in two steps. First, the values $x_t, t = 1, 2, \ldots, n + m - 1$ were chosen to be the daily U.S. Dollar/Euro Foreign Exchange Rate derived from the Board of Governors of the Federal Reserve System for the period 1/4/1999 - 5/8/2007, which amounts to $n + m - 1 = 2099$ observations. Second, each of the components $\{y_t : t = m, m+1, \ldots, n+m-1\}$ was generated from (1) after a function value $\beta(z_i)$ was substituted for β_i and a number from the uniform distribution $U[-0.05, 0.05]$ was substituted for e_t, where

$$\beta(z) = \exp(z), z \in [0, 1] \tag{25}$$

and

$$\beta(z) = 1 + \sin(2z + 1)/(1 + z^2), z \in [0, 2]. \tag{26}$$

Function (25) was chosen because it has nonnegative derivatives of all orders and its measurements appear particularly suitable for fitting with nonnegative differences. Function (26) is a concave/convex function and was chosen to diversify the final numbers of active constraints, due to the sign changes that occur in its derivatives of all orders. In this case one may assume that fitting by nonnegative differences is likely to be poor, but the examples below show that this can be false for $r \geq 3$, because the set of vectors defined by (4) is strictly larger than the set of points that can be interpolated by functions with nondecreasing derivatives of order $r - 1$ (see, [4], [5]). For each of the two underlying functions, the lag length was chosen $m = 26, 51, 76, 101$ and for each m the data points z_i have equally spaced values.

All the experiments required the calculation of $\underline{\beta}$ by minimizing the objective function (2) subject to the constraints (9), while $r = 2, 3, 4$ and 5. The actual values of m, r and the following list of calculated parameters are given in Tables 1 and 2 for the underlying functions (25) and (26) respectively:

1. $S_{\beta\hat{\beta}} = \sqrt{\sum_{i=1}^{m} (\beta(i) - \hat{\beta}_i)^2}$, the distance between the function values $\beta(i), i = 1, 2, \ldots, m$ and the estimated lag coefficients $\hat{\beta}_i, i = 1, 2, \ldots, m$.
2. $P_{RelError} = \max_{1 \leq i \leq n} |y_i - \underline{\hat{\beta}}^T \underline{\xi}^{(i)}| / (\max_{1 \leq i \leq n} y_i - \min_{1 \leq i \leq n} y_i) \times 100$, the percent relative error of the time series estimation, which relates the error to the scale of values taken by the data, where $\underline{\xi}^{(i)}$ is the ith column of matrix $\mathbf{X}$.
3. $|\mathcal{A}^*|$, the number of constraints at final active set.
4. The number of active set changes (additions and deletions) required by Algorithm 1 to calculate the lag coefficients.
5. The CPU time in seconds for calculating the lag coefficients.
6. $R_{KKT} = \max_{i \in \mathcal{A}^*} |2(\mathbf{X}\underline{\hat{\beta}} - \underline{y})^T \underline{\xi}^{(i)} - \sum_{k \in \mathcal{A}^*} \lambda_k(\mathcal{A}^*)(\mathbf{D}_r)_{ik}|$, the maximum component of the residuals of the Karush-Kuhn-Tucker (abrev. KKT) conditions (11). In view of (11), the quantity R_{KKT} is zero in exact arithmetic.

The parameter 1 requires the a-priori knowledge of the underlying function of the lag coefficients, therefore it can be used only for testing purposes. The parameter 2 is the actual time series smoothing quality indicator that the user has available at the end of the calculation. The parameters 3, 4 and 5 present the computational performance of the method. The parameter 6 provides a measure of the accuracy of the computer program that implements the method.

The amount of work of the main iterations of Algorithm 1 can be deduced from the fifth column of Tables 1 and 2. Specifically, because each visit to Step 1 adds a constraint and each visit to Step 3 deletes one, column 5 gives the total number of occurrences of Steps 1 and 3. Column 7 presents the times to perform the calculations in double precision arithmetic using the standard Fortran 77 compiler of Compaq Visual Fortran 6.1 on a Personal Computer with an Intel 2.4 GHz processor operating in Microsoft Windows XP with 32 bits word length. A direct comparison

of the number of active set changes and CPU time indicates the work required by a single call of Algorithm 2 in order to calculate the lag coefficients and the corresponding Lagrange multipliers.

In almost all cases presented in Tables 1 and 2, Algorithm 1 terminated in fewer than $m - r$ active set changes (column 5) with a large active set (column 6), while for $r \leq 4$ the accuracy of the computer program seems to be very good (column 8). In some cases, the algorithm terminated with no active set changes, because all the final active constraints were identified at Step 0. The best results for Table 1 were obtained when r equals 3 or 4 and for Table 2 when r equals 3 or 5, after which the coefficient error $S_{\beta\hat{\beta}}$ and the time series error reduction $P_{RelError}$ start to increase. For all values of m, as r increased the value of R_{KKT} decreased. For instance, the result $R_{KKT} = 1.37E-02$ in Table 1, obtained when $m = 101$ and $r = 5$, shows that the arithmetic for calculating the Lagrange multipliers in this case is less accurate than when $r \leq 4$. The reason is that, as m becomes larger (as, for instance, in the cases with $m = 101$), the second derivative matrix $\mathbf{X}^T\mathbf{X}$ is very ill-conditioned, while the calculation is further aggravated by the errors in the time series measurements y_t. However, it is worth mentioning that cancelation errors do not occur when calculating the coefficients (10) of scaled higher differences (9), as opposed to the case that makes use of general divided differences ([37]:p.47).

m	r	$S_{\beta\hat{\beta}}$	$P_{RelError}$	Active set changes	$\lvert\mathcal{A}^*\rvert$	CPU time (sec)	R_{KKT}
	2	0.0517	0.3463	3	22	0.04	2.42E-09
26	3	0.0338	0.3506	4	22	0.04	1.38E-07
	4	0.0571	0.3509	12	19	0.09	3.54E-08
	5	0.0752	0.3527	24	20	0.18	1.20E-06
	2	0.0393	0.1679	25	44	0.31	1.04E-08
51	3	0.0279	0.1655	52	47	0.60	3.76E-06
	4	0.0385	0.1678	41	45	0.59	1.06E-06
	5	0.0391	0.1674	83	44	0.89	1.79E-04
	2	0.0363	0.1012	43	69	0.85	4.37E-08
76	3	0.0155	0.1000	38	72	0.46	3.07E-05
	4	0.0062	0.0969	23	71	0.32	2.15E-06
	5	0.0426	0.1025	47	68	0.70	2.78E-03
	2	0.0641	0.0673	58	93	1.53	1.55E-07
101	3	0.0104	0.0662	87	94	1.87	3.84E-05
	4	0.0078	0.0657	1	96	0.04	5.20E-06
	5	0.0117	0.0684	0	96	0.00	1.37E-02

Table 1: Parameters of time series estimation performance and efficiency of Algorithm 1, when $\beta(z) = \exp(z), z \in [0, 1]$

4. An example on U.S.A. consumption data and a comparison with the method of Almon

In order to illustrate our method we present an application on real annual macroeconomic data derived from the Bureau of Economic Analysis of the U.S.A. Department of Commerce for the period 1/1/1929 - 1/1/2006, but we do not discuss the economic implications of the data or the results. The dependent variable is the Real Personal Consumption Expenditures (PCE) and the independent variable is the Real Gross Domestic Product (GDP) for U.S.A., both measured in billions of chained 2000 dollars. The data of our application are reported in Table 3 and amount to 78 pairs of observations. We assume that a change in the GDP will affect not only current consumption, but also future consumption for a number of time periods. Therefore we calculated the coefficients of the distributed-lag model with lag length $m = 5, 6, 7, 8, 9$ and 10 subject to the constraints (9) on the components of β by allowing $r = 2, 3, 4$ and 5. In order to compare results, for each value of m we calculated the lag coefficients by Almon's polynomials of degree $k = 1, 2, 3$ and 4. Almon's coefficients are shown in the third ($k = 1$), fourth ($k = 2$), fifth ($k = 3$) and sixth ($k = 4$) column of the relevant part of Table 4 for each m, while the r-convex coefficients are shown in the seventh ($r = 2$), eighth ($r = 3$), ninth ($r = 4$) and tenth ($r = 5$) column respectively. Finally, the unconstrained lag coefficients for each m, obtained by minimizing (2), are shown in the last column of Table 4. We see that the r-convex lag coefficients

m	r	$S_{\beta\hat{\beta}}$	$P_{RelError}$	Active set changes	$\lvert\mathcal{A}^*\rvert$	CPU time (sec)	R_{KKT}
	2	0.3966	0.3835	6	22	0.06	3.16E-09
26	3	0.1314	0.3297	12	21	0.10	1.13E-07
	4	0.2790	0.3597	0	22	0.00	4.35E-08
	5	0.1300	0.3297	13	20	0.12	6.32E-07
	2	0.5209	0.2567	28	46	0.39	1.43E-08
51	3	0.0652	0.1714	20	45	0.23	1.66E-06
	4	0.3496	0.2277	0	47	0.00	8.45E-07
	5	0.0778	0.1711	24	44	0.28	7.95E-05
	2	0.6208	0.2437	37	72	0.62	1.41E-08
76	3	0.0633	0.1201	47	71	0.68	1.20E-05
	4	0.4184	0.2069	0	72	0.00	1.62E-06
	5	0.0938	0.1210	85	69	1.12	1.00E-03
	2	0.7054	0.2644	59	93	1.26	3.64E-08
101	3	0.0766	0.0974	90	94	1.42	3.63E-05
	4	0.4793	0.2021	0	97	0.00	4.49E-06
	5	0.0842	0.0991	113	93	1.94	5.87E-03

Table 2: As in Table 1, but $\beta(z) = 1 + \sin(2z+1)/(1+z^2), z \in [0,2]$

cannot deviate far from the polynomial of degree $r-1$ and, indeed, they do so in a smooth manner alternating above and below the polynomial curve.

In Figs 5, 6, 7 and 8 we display the unconstrained, the r-convex and the kth degree Almon polynomial lag coefficients of Table 4 for $m = 8$, while $r = 2, 3, 4$ and 5, and $k = 1, 2, 3$ and 4 respectively. The condition number of the 8×8 matrix $\mathbf{X}^T\mathbf{X}$ was found by Matlab to be equal to 170080, which exhibits the ill-conditioned character of the problem. Since the r-convex model is a piecewise polynomial, it seems to be more suitable in following the pattern of the unconstrained lag coefficients, than the corresponding polynomial of $(r-1)$th degree. Indeed, in Fig. 5 the 2-convex model is a linear spline with interior knots at the second and fourth data point as opposed to Almon's straight line model. In Fig. 6 the 3-convex model coincides with Almon's 2nd degree polynomial, because all the constraints (9) are satisfied as equalities. In Fig. 7 the 4-convex model contains two overlapping cubics and is obtained by minimizing (2) subject to the equality constraints $\beta_6 - 4\beta_5 + 6\beta_4 - 4\beta_3 + \beta_2 = 0$ and $\beta_7 - 4\beta_6 + 6\beta_5 - 4\beta_4 + \beta_3 = 0$. In Fig. 8 the 5-convex model contains just one quartic and is obtained by minimizing (2) subject to the equality constraint $-\beta_7 + 5\beta_6 - 10\beta_5 + 10\beta_4 - 5\beta_3 + \beta_2 = 0$. Moreover, a standard result from sensitivity analysis is that if a Lagrange multiplier is large, then the optimal value $F(\hat{\beta})$ is sensitive to the perturbation of the corresponding constraint, while if a Lagrange multiplier is small, the dependence is much weaker. Thus, the larger the multiplier, the stronger the dependence upon the corresponding constraint. The following table indicates these dependencies by reporting the Lagrange multipliers for each case (the zero Lagrange multipliers correspond to non-active constraints).

	$r=2$	$r=3$	$r=4$	$r=5$
λ_1	0.00	95836.10	0.00	0.00
λ_2	1779.47	217696.21	830.12	438.44
λ_3	0.00	265146.41	90.66	0.00
λ_4	22643.45	187651.75	0.00	
λ_5	26744.57	65369.76		
λ_6	10783.24			

5. Conclusions

We have developed a new method for calculating distributed-lag coefficients in time series estimation subject to the condition that the rth consecutive differences of the m coefficient estimates are nonnegative, which may well be

GDP	PCE	GDP	PCE	GDP	PCE
865.2	661.4	2212.8	1385.5	5291.7	3422.2
790.7	626.1	2255.8	1425.4	5189.3	3470.3
739.9	606.9	2301.1	1460.7	5423.8	3668.6
643.7	553.0	2279.2	1472.3	5813.6	3863.3
635.5	541.0	2441.3	1554.6	6053.7	4064.0
704.2	579.3	2501.8	1597.4	6263.6	4228.9
766.9	614.8	2560.0	1630.3	6475.1	4369.8
866.6	677.0	2715.2	1711.1	6742.7	4546.9
911.1	702.0	2834.0	1781.6	6981.4	4675.0
879.7	690.7	2998.6	1888.4	7112.5	4770.3
950.7	729.1	3191.1	2007.7	7100.5	4778.4
1034.1	767.1	3399.1	2121.8	7336.6	4934.8
1211.1	821.9	3484.6	2185.0	7532.7	5099.8
1435.4	803.1	3652.7	2310.5	7835.5	5290.7
1670.9	826.1	3765.4	2396.4	8031.7	5433.5
1806.5	850.2	3771.9	2451.9	8328.9	5619.4
1786.3	902.7	3898.6	2545.5	8703.5	5831.8
1589.4	1012.9	4105.0	2701.3	9066.9	6125.8
1574.5	1031.6	4341.5	2833.8	9470.3	6438.6
1643.2	1054.4	4319.6	2812.3	9817.0	6739.4
1634.6	1083.5	4311.2	2876.9	9890.7	6910.4
1777.3	1152.8	4540.9	3035.5	10048.8	7099.3
1915.0	1171.2	4750.5	3164.1	10301.0	7295.3
1988.3	1208.2	5015.0	3303.1	10703.5	7577.1
2079.5	1265.7	5173.4	3383.4	11048.6	7841.2
2065.4	1291.4	5161.7	3374.1	11415.3	8091.4

Table 3: The values of GDP and PCE for the years 1929-2006 (U.S.A. Department of Commerce)

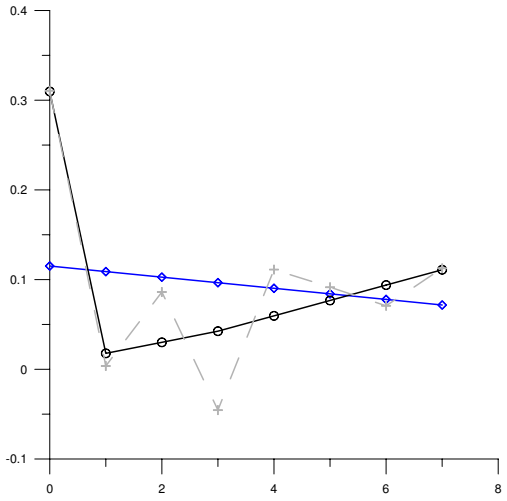

Figure 5: The unconstrained (+), the r-convex (o) and the kth degree Almon polynomial ($\diamond$) lag coefficients of Table 4, when $m = 8$, $r = 2$ and $k = 1$

applied for the case where the differences are non-positive.

		Almon's coeffs				r-convex coeffs				unconstrained
m	β_i	$k=1$	$k=2$	$k=3$	$k=4$	$r=2$	$r=3$	$r=4$	$r=5$	lag coefficients
	β_0	0.1751	0.3955	0.3383		0.3572	0.3347	0.3828		0.3828
	β_1	0.1593	0.0474	0.1344		0.0848	0.1226	0.0014		0.0014
5	β_2	0.1434	-0.0787	-0.0786		-0.0171	-0.0316	0.0984		0.0984
	β_3	0.1276	0.0172	-0.0693		-0.1190	-0.1280	-0.2024		-0.2024
	β_4	0.1118	0.3352	0.3936		0.4125	0.4211	0.4380		0.4380
	β_0	0.1399	0.3119	0.2900	0.3214	0.3174	0.2476	0.3272	0.3759	0.3346
	β_1	0.1325	0.0974	0.1191	0.0490	0.0622	0.1409	0.0441	-0.1532	0.0074
6	β_2	0.1252	-0.0134	-0.0003	0.0385	0.0286	0.0445	0.0310	0.2955	0.1044
	β_3	0.1178	-0.0205	-0.0334	0.0050	-0.0050	-0.0415	0.0142	-0.1602	-0.0613
	β_4	0.1105	0.0762	0.0547	-0.0159	0.0048	-0.0107	-0.0152	0.0741	0.0263
	β_5	0.1031	0.2766	0.2990	0.3312	0.3210	0.3508	0.3278	0.2974	0.3176
	β_0	0.1219	0.2452	0.2807	0.3099	0.3135	0.2271	0.3198	0.3097	0.3283
	β_1	0.1165	0.1158	0.0923	0.0436	0.0326	0.1364	0.0266	0.0441	-0.0157
	β_2	0.1110	0.0360	0.0102	0.0121	0.0296	0.0683	0.0197	0.0117	0.1060
7	β_3	0.1056	0.0060	0.0059	0.0407	0.0267	0.0228	0.0389	0.0405	-0.0544
	β_4	0.1002	0.0258	0.0513	0.0525	0.0522	0.0257	0.0587	0.0534	0.1163
	β_5	0.0948	0.0953	0.1181	0.0689	0.0777	0.0769	0.0608	0.0680	0.0411
	β_6	0.0894	0.2146	0.1780	0.2089	0.2041	0.1764	0.2120	0.2092	0.2147
	β_0	0.1152	0.1963	0.2691	0.2964	0.3097	0.1963	0.3021	0.3030	0.3105
	β_1	0.1089	0.1194	0.0886	0.0530	0.0177	0.1194	0.0418	0.0430	0.0037
	β_2	0.1027	0.0663	0.0149	0.0033	0.0301	0.0663	0.0114	0.0039	0.0862
8	β_3	0.0965	0.0369	0.0147	0.0344	0.0425	0.0369	0.0305	0.0408	-0.0455
	β_4	0.0903	0.0313	0.0544	0.0736	0.0596	0.0313	0.0682	0.0632	0.1112
	β_5	0.0841	0.0493	0.1006	0.0877	0.0768	0.0493	0.1057	0.1033	0.0915
	β_6	0.0779	0.0911	0.1199	0.0839	0.0939	0.0911	0.0643	0.0678	0.0707
	β_7	0.0717	0.1567	0.0787	0.1090	0.1110	0.1567	0.1171	0.1161	0.1127
	β_0	0.1114	0.1646	0.2321	0.3001	0.2778	0.1646	0.2795	0.2795	0.2961
	β_1	0.1045	0.1167	0.1001	0.0309	0.0343	0.1167	0.0619	0.0619	-0.0085
	β_2	0.0976	0.0807	0.0378	-0.0062	0.0413	0.0807	-0.0055	-0.0055	0.1159
	β_3	0.0907	0.0565	0.0258	0.0458	0.0483	0.0565	0.0204	0.0204	-0.0848
9	β_4	0.0838	0.0442	0.0447	0.0935	0.0553	0.0442	0.0828	0.0828	0.1330
	β_5	0.0769	0.0438	0.075	0.0933	0.0623	0.0438	0.1250	0.1250	0.0824
	β_6	0.0700	0.0553	0.0975	0.0513	0.0692	0.0553	0.0901	0.0901	0.1649
	β_7	0.0631	0.0787	0.0927	0.0234	0.0762	0.0787	-0.0785	-0.0785	-0.1513
	β_8	0.0562	0.1140	0.0413	0.1151	0.0832	0.1140	0.1732	0.1732	0.2013
	β_0	0.1050	0.1477	0.1817	0.3093	0.2357	0.1334	0.2911	0.2748	0.2975
	β_1	0.0986	0.1118	0.1079	0.0071	0.0557	0.1093	-0.0053	0.0526	-0.0308
	β_2	0.0922	0.0835	0.0644	-0.0279	0.0540	0.0885	0.0430	-0.0092	0.0970
	β_3	0.0858	0.0626	0.0447	0.0430	0.0523	0.0712	0.0384	0.0096	-0.0404
10	β_4	0.0794	0.0492	0.0422	0.1103	0.0506	0.0572	-0.0048	0.0616	0.0737
	β_5	0.0730	0.0434	0.0504	0.1166	0.0489	0.0467	0.1585	0.1318	0.1120
	β_6	0.0667	0.0450	0.0627	0.0561	0.0472	0.0395	0.1392	0.1313	0.1583
	β_7	0.0603	0.0542	0.0726	-0.0248	0.0455	0.0358	-0.0033	0.0040	-0.0173
	β_8	0.0539	0.0708	0.0734	-0.0280	0.0438	0.0355	-0.2098	-0.2061	-0.1963
	β_9	0.0475	0.0949	0.0586	0.1967	0.1263	0.1494	0.3145	0.3113	0.3077

Table 4: Almon's ($k = 1, 2, 3, 4$), r-convex ($r = 2, 3, 4, 5$) and the unconstrained lag coefficients for $m = 5, 6, 7, 8, 9, 10$

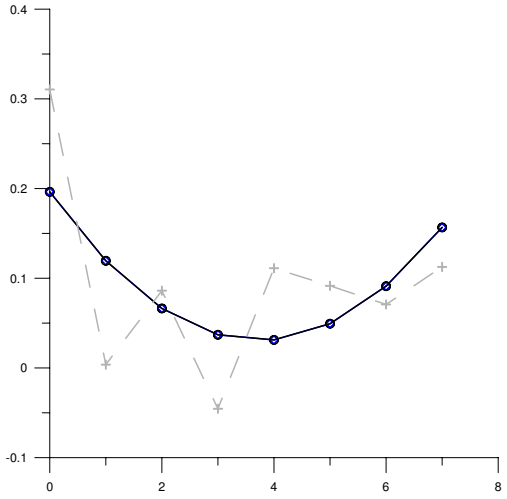

Figure 6: As in Fig. 5, but $r = 3$ and $k = 2$

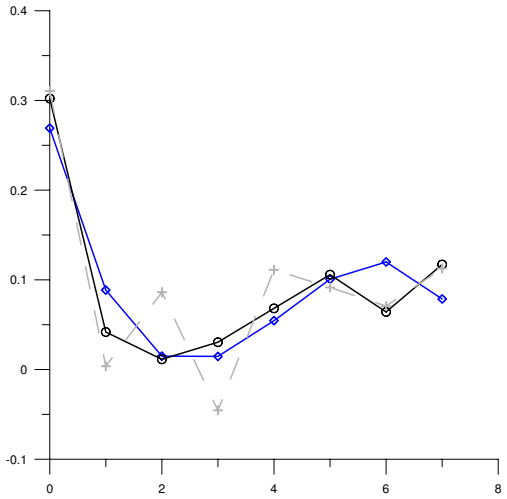

Figure 7: As in Fig. 5, but $r = 4$ and $k = 3$

The method is a strictly convex quadratic programming algorithm that takes account of the least squares objective function and the fact that each of the constraint functions depends on only $r + 1$ adjacent components of the lag coefficients that give a Toeplitz structure. We have developed efficient procedures for the solution of the equality constrained minimization problem and the calculation of the corresponding Lagrange multipliers that occur during the active set revisions of the quadratic programming iterations. Specifically, we have considered a particularly convenient basis $\{\underline{u}_s : s \in S\}$ in the linear space of active constraint gradients, where the definition of S takes account of the non-active constraints, and we worked with reduced quantities throughout the calculation. Four advantages were gained. One is that the number of variables that occurs in practice for our calculation is much lower than m, the original number of variables. The second is that the active constraints are satisfied automatically due to the choice of the basis. The third is that the calculation of the basis depends on a positive definite subsystem of equations derived from the active constraints, where positive definiteness comes from the Toeplitz structure. The fourth is that the matrix that occurs in the calculation of the Lagrange multipliers is the transpose of the matrix of the subsystem that was already used for obtaining $\{\underline{u}_s : s \in S\}$. Therefore, a factorization of this matrix is available and the Lagrange multipliers can be found only by a small amount of extra work after the basis elements are calculated. The particular choice of

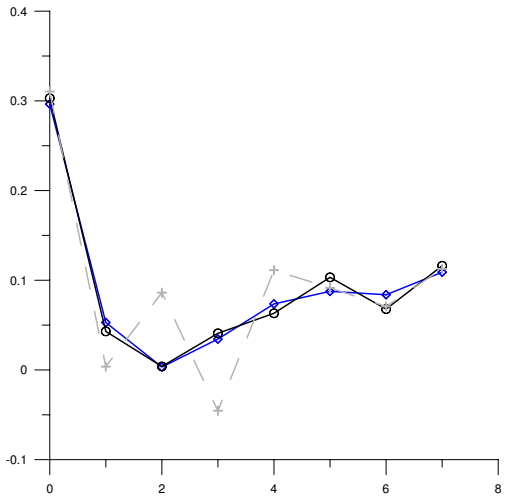

Figure 8: As in Fig. 5, but $r = 5$ and $k = 4$

$\{\underline{u}_s : s \in S\}$ is also important in its own right, because it provides a basis for the linear space of piecewise polynomials that are defined by the difference equations $\Delta_j^r \beta = 0,\ j \in \mathcal{A}$. This choice provides a major computing saving, because although the size of the active sets that are carried out during the calculation is usually quite close to $m - r$ (which is the number of constraints of the quadratic programming problem), we solve only $r \times r$ (about) systems, where r is a small number in practice. Moreover, the calculation is stable, because all the coefficient matrices of the mentioned subsystems of equations are positive definite.

The question arises of a suitable choice of r in general. If there exists appropriate prior information to be taken into account, then our method may have advantages. If the right choice of r is a matter of experimentation, the user may try iteratively some values of r, while simultaneously keep checking adjacent values of the parameters $F(\hat{\beta})$, $P_{RelError}$ and R_{KKT}. If each parameter possesses approximately the same value for successive r, while R_{KKT} is sufficiently small, then these values of r are most likely to provide adequate choices. Instead, a model underfit is usually indicated by a large $P_{RelError}$ and small R_{KKT}, where the latter indicates accurate termination and the former large deviations of the estimated components. Furthermore, if for some values of r, the parameters $F(\hat{\beta})$ and $P_{RelError}$ take isolated values, while R_{KKT} is large, then the user should be suspicious of inaccuracies due to round-off, for the reasons given in Section 3.

The proposed method seems to be a useful one for distributed-lag estimation, which is at least more general and comparably more competent than Almon's polynomial family method. The connection with Almon polynomial is that this polynomial satisfies all our rth difference inequality constraints as equalities, so the Almon lag coefficients lie on a polynomial of degree $r - 1$. Our model, as a piecewise polynomial, is much more successful in practice, because it allows *that* number of overlapping polynomial pieces of degree $r - 1$, *which* the quadratic programming method automatically provides.

Three modeling advantages of using the new method are that it achieves a rather weak representation of the lag coefficients, which is highly desirable in lag estimation practices, it obtains well recognized structures due to prior knowledge of the r-convexity property and it provides estimates of rth derivative values of the underlying relation. The Fortran program we have developed for the r-convex model required indeed a good deal of effort and hopefully it would be very helpful for empirical analyses and applications to real problems. Since no assumption is made about the nature of an underlying relation, it is also indicated that this method may be of general use.

References

[1] Almon, S., 1965. The Distributed Lag between Capital Appropriations and Expenditures. *Econometrica*, 33(1), 178-196.

[2] Bunch, J. R., 1985. Stability of Methods for Solving Toeplitz Systems of Equations. *SIAM J. Sci. Stat. Comp.*, 6, 349-364.

[3] Corradi, C., 1977. Smooth Distributed Lag Estimators and Smoothing Spline Functions in Hilbert Spaces. *Journal of Econometrics*, 5, 211-219.
[4] Cullinan, M. P., 1990. Data smoothing using non-negative divided differences and $l2$ approximation. *IMA J. of Numerical Analysis*, 10, 583-608.
[5] Cullinan, M. P. & Powell, M. J. D., 1982. Data smoothing by divided differences. In: *Numerical Analysis Proc. Dundee 1981* (ed. G. A. Watson), LNIM 912, Berlin: Springer-Verlag, 26-37.
[6] Demetriou, I. C., 1995. Algorithm 742: L2CXFT: A Fortran subroutine for least squares data fitting with non-negative second divided differences. *ACM Trans. on Math. Software*, 21(1), 98-110.
[7] Demetriou, I. C. & Lipitakis, E. A., 2001. Certain positive definite submatrices that arise from binomial coefficient matrices. *Applied Numerical Mathematics*, 36, 219-229.
[8] Demetriou, I. C. & Powell, M. J. D., 1991. The minimum sum of squares change to univariate data that gives convexity. *IMA J. of Numerical Analysis*, 11, 433-448.
[9] Fisher, I., 1937. Note on a Short-Cut Method for Calculating Distributed Lags. *International Statistical Institute Bulletin*, 323-327.
[10] Fomby, T. B., Hill, R. C. & Johnson, S. R., 1984. *Advanced Econometric Methods*, New York: Springer.
[11] Fletcher, R., 2003. *Practical Methods of Optimization, Second Edition*. Chichester: J. Wiley and Sons.
[12] Gershenfeld, N., 1999. *The Nature of Mathematical Modelling*. Cambridge: Cambridge University Press.
[13] Gill, P. E. & Murray, W., 1978. Numerically stable methods for quadratic programming. *Math. Programming*, 14, 349-372.
[14] Gill, P. E., Murray, W., Saunders, M. A. & Wright, M. H., 1983. User's guide for SOL/QPSOL: a Fortan package for quadratic programming. Report SOL 83-7, Stanford University.
[15] Gujarati, D., 2003. *Basic Econometrics, Fourth Edition*. London: McGraw-Hill Book Co.
[16] Goldfarb, D. & Idnani, A., 1983. A numerically stable dual method for solving strictly convex quadratic programs. *Math. Programming*, 27, 1-33.
[17] Golub, G. & van Loan, C. F., 1989. *Matrix Computations, Second Edition*. Baltimore and London: The John Hopkins University Press.
[18] Gray, R. M., 2006. *Toeplitz and Circulant Matrices: A Review*. Stanford: Department of Engineering, http://ee.stanford.edu/~gray/toeplitz.pdf.
[19] Griliches, Z., 1967. Distributed Lags: A Survey. *Econometrica*, 35(1), 16-49.
[20] Groetch, C. W., 1993. *Inverse Problems in the Mathematical Sciences*, Braunschchweig: Vieweg.
[21] Hannan, E. J., 1965. The Estimation of Relations Involving Distributed Lags. *Econometrica*, 33(1), 206-224.
[22] Harezlak, J., Coull, B. A., Laird, N. M., Magari, S. R. & Christiani, D. C. 2007. Penalized solutions to functional regression problems. *Computational Statistics and Data Analysis*, 51 (10), 4911-4925.
[23] Hildebrand, F. B., 1974. *Introduction to Numerical Analysis*, Second Edition. New York: Dover Publication, Inc.
[24] Jorgenson, D., 1966. Rational Distributed Lag Functions. *Econometrica*, 34(1), 135-149.
[25] Jorgenson, D. W. & Kun-Young Yun, 2001. *Investment, Volume 3 Lifting the Burden: Tax Reform, the Cost of Capital, and U.S. Economic Growth*. Cambrdige, Mass: The MIT Press.
[26] Kailath, T. (editor), 1977. *Linear Least-Squares Estimation*. Stroudsburg, Pennsylvania: Dowden, Hutchinson and Ross, Inc., Benchmark Papers in Electrical Engineering and Computer Science, 17.
[27] Karlin, S., 1968. *Total Positivity*. Volume 1. Stanford, California: Stanford University Press.
[28] Koop, G., 2000. *Analysis of Economic Data*. Chichester: J. Wiley and Sons.
[29] Koyck, L., 1954. *Distributed Lags and Investment Analysis*. Amsterdam: North-Holand Pub. Co.
[30] Lawson, C. L. & Hanson, R. J., 1995. *Solving Least Squares Problems*. Philadelphia: SIAM Publications.
[31] DeLeeuw, F., 1962. The Demand for Capital Goods by Manufactures: A Study of Quarterly Time Series. *Econometrica*, 30, 407-423.
[32] Liew, C. K., 1976. Inequality Constrained Least-Squares Estimation *J. of the American Statistical Association*, 71, 746-751.
[33] Maddala, G. S., 1977. *Econometrics*. London: McGraw-Hill Book Co.
[34] Ng, M. T., 2004. *Iterative Methods for Toeplitz Systems*. Oxford: Oxford University Press.
[35] Nocedal, J. & Wright, S. J., 1999. *Numerical Optimization*. New York: Springer.
[36] Polasek, W., 1990 Vector distributed lag models with smoothness priors. *Computational Statistics and Data Analysis*, 10(2), 133-141.
[37] Powell, M. J. D., 1981. *Approximation Theory and Methods*. Cambridge: Cambridge University Press.
[38] Powell, M. J. D., 1985. On the quadratic programming algorithm of Goldfarb and Idnani. *Math. Programming Studies*, 25, 46-61.
[39] Robertson, T., Wright, F. T. & Dykstra, R. L., 1988. *Order Restricted Statistical Inference*. Chichester: J. Wiley and Sons.
[40] SAS User Guide, 2008. SAS Institute Inc. http://www.sas.com/software/, 100 SAS Campus Drive Cary, NC 27513-2414 USA.
[41] Shiller, R. J., 1973. A Distributed Lag Estimator Derived from Smothness Priors. *Econometrica*, 41(4), 775-788.
[42] Solow, R. M., 1960. On a Family of Lag Distributions. *Econometrica*, 28(2), 393-406.
[43] Tikhonov, A. N., 1977. *Solutions for Ill-posed Problems*. Washington: Winston and Sons.
[44] Tsay, R. S., 2002. *Analysis of Financial Time Series*. New York: J. Wiley and Sons.
[45] Vogel, C. R., 2003. *Computational Methods for Inverse Problems*. Philadelphia: SIAM FR23.