Efficient and Direct Estimation of the Variance-Covariance Matrix in EM Algorithm with Interpolation Method

Lili Yu[a], Ding-Geng Chen[b] and Jun Liu[c]
[a]Jiann-Ping Hsu College of Public Health, Georgia Southern University, Statesboro, GA, 30460, USA.
[b]School of Social Work & Department of Biostatistics, Gillings School of Global Health, University of North Carolina,Chapel Hill, NC, 2759, USA.
[b]Department of statistics, University of Pretoria, South Africa.
[c]Department of Enterprise Systems & Analytics, Georgia Southern University, Statesboro, GA, 30460, USA.

Corresponding author: Lili Yu, Jiann-Ping Hsu College of Public Health, Georgia Southern University, Statesboro, GA, 30460, USA.

Abstract: The expectation-maximization (EM) algorithm is a seminal method to calculate the maximum likelihood estimators (MLEs) for incomplete data. However, one drawback of this algorithm is that the asymptotic variance-covariance matrix of the MLE is not automatically produced. Although there are several methods proposed to resolve this drawback, limitations exist for these methods. In this paper, we propose an innovative interpolation procedure to directly estimate the asymptotic variance-covariance matrix of the MLE obtained by the EM algorithm. Specifically we make use of the cubic spline interpolation to approximate the first-order and the second-order derivative functions in the Jacobian and Hessian matrices from the EM algorithm. It does not require iterative procedures as in other previously proposed numerical methods, so it is computationally efficient and direct. We derive the truncation error bounds of the functions theoretically and show that the truncation error diminishes to zero as the mesh size approaches zero. The optimal mesh size is derived as well by minimizing the global error. The accuracy and the complexity of the novel method is compared with those of the well-known SEM method. Two numerical examples and a real data are used to illustrate the accuracy and stability of this novel method.

Key words and phrases: Cubic spline interpolation, Hessian matrix, Incomplete data, Jacobian matrix, Maximum likelihood estimation.

1. Introduction

Maximum likelihood estimator (MLE) plays a key role in parameter estimation from observed data in statistics. It is calculated by finding the parameter values that maximize the likelihood function given the data. However, when the likelihood is complex or incomplete, the MLE is difficult to obtain. The expectation-maximization (EM) algorithm (Dempster, Laird and Rubin 1977) was proposed as an iterative procedure to obtain the MLE. It consists of two steps: in the E-step, a simple and complete-data likelihood is obtained and in the M-step, a standard maximum likelihood estimation is performed using the complete data obtained from the E-step. The simple calculation gains the EM algorithm a wide application in statistics. However, the asymptotic variance-covariance matrix of the MLE is not provided directly from the EM algorithm.

As remedies, several methods are proposed to obtain the asymptotic variance-covariance matrix. However, limitations exist for these methods. Louis (1982) provided an estimator with closed form, which requires conditional expectation (conditional on the observed data) of the Hessian matrix of the complete data and of the square of the complete-data score function, which is specific to each problem. Oakes (1999) proposed an estimator with closed form as well. However, this estimator requires the second derivatives of the expected complete-data likelihood, which is not always available. Meng and Rubin (1991) proposed the supplemented EM (SEM) algorithm, which only requires the code for EM itself. However, the SEM requires the calculation of the conditional expectation of the Hessian matrix and is sometimes infeasible (Beker 1992). Moreover, the computational cost of the SEM is high due to the requirement of running the EM algorithm for each parameter separately (Belin and Rubin 1995). Furthermore, it seems to be susceptible to numerical inaccuracies and instability, especially in high-dimensional settings (Beker 1992, McCulloch 1998, Segal et al. 1994). Monte Carlo method can be employed by multiple imputation (Rubin 1987) for missing data given the maximum likelihood estimate, but it is time consuming and inaccurate for small number of imputations. Recently, Meng and Spall (2017) showed that the simultaneous perturbation stochastic approximation (Spall 1992, 2005) method performs well for the approximation of the Hessian matrix of the incomplete data log-likelihood evaluated at the EM estimator. However, it is still based on numerical differentiation, which may be inaccurate and unstable.

Among all these methods to estimate the variance of the EM estimator, the SEM method is deemed to be the best so far and therefore we will compare our newly proposed method in this paper to the best SEM.

The fundamental difficulty to obtain the asymptotic variance-covariance matrix directly from the EM algorithm is due to the unavailability of the derivatives of some known functions. In this paper, we propose an innovative interpolation method to directly estimate the asymptotic variance-covariance matrix of the EM estimators (hereafter named as iEM). The basic idea of iEM is that we use cubic spline interpolatory functions to approximate these known functions. Then we use the derivatives of the cubic spline functions, which are very easy to obtain, to estimate the derivatives of the known functions, hence the variance-covariance matrix. The iEM overcomes all limitations in previous methods. First, it does not require anything other than those provided by the EM algorithm. Therefore, it is a general method that can apply to any problem where the EM algorithm is applied. Second, because it does not need iterative calculations, the iEM is computationally efficient. Third, We derive the optimal mesh size to minimize the global estimation error of the iEM method, so that the accurate and stable estimates can be obtained.

This paper is organized as follows. In Section 2, we give an review of the EM algorithm and the SEM method. We describe the cubic spline interpolation EM (i.e. iEM) in Section 3 and its applications to the EM algorithm with the corresponding theories and the instructions on how to select mesh and mesh size in Section 4. The comparison of the iEM method with the SEM method is given in Section 5. Two numerical examples are presented in Section 6 followed by discussions and conclusions in Section 7.

2. Review of the Classical EM Algorithm

2.1 The EM algorithm

Let $\mathbf{Y}_{obs}$ and $\mathbf{Y}_{mis}$ be the observed data and missing data, respectively. $\mathbf{Y} = \{\mathbf{Y}_{obs}, \mathbf{Y}_{mis}\}$ is the complete data with density $f(\mathbf{Y}|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a d-dimensional vector of d unknown parameters. The EM algorithm aims to find the maximum likelihood estimate (MLE) $\boldsymbol{\theta}^*$ of $\boldsymbol{\theta}$ based on the observed data $\mathbf{Y}_{obs}$.

The EM algorithm starts with an initial value $\boldsymbol{\theta}^{(0)}$ and iterates between the

E-step and the M-step. Specifically, at iteration $t+1$,

- E-step: Based on the current estimate $\boldsymbol{\theta}^{(t)}$, find the expected complete-data log-likelihood

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \int L(\boldsymbol{\theta}|\mathbf{Y}) f(\mathbf{Y}_{mis}|\mathbf{Y}_{obs}, \boldsymbol{\theta} = \boldsymbol{\theta}^{(t)}) d\mathbf{Y}_{mis}, \tag{2.1}$$

  where $L(\boldsymbol{\theta}|\mathbf{Y}) = logf(\mathbf{Y}|\boldsymbol{\theta})$.

- M-step: Obtain $\boldsymbol{\theta}^{(t+1)}$ by maximizing the expected complete-data log-likelihood (2.1),

$$\boldsymbol{\theta}^{(t+1)} = arg \max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}).$$

The iteration is repeated until $L(\boldsymbol{\theta}|\mathbf{Y})$ converges. Define the mapping $\boldsymbol{\theta}^{(t+1)} = \mathbf{M}(\boldsymbol{\theta}^{(t)})$, for $t = 0, 1, \cdots$, the EM algorithm indicates that if $\mathbf{M}(\boldsymbol{\theta})$ is continuous, then $\boldsymbol{\theta}^* = \mathbf{M}(\boldsymbol{\theta}^*)$.

2.2 The SEM Algorithm

As shown in Meng and Robin (1991), the asymptotic variance-covariance matrix $\mathbf{V}$ of $\boldsymbol{\theta}^*$ is the inverse of the observed information matrix, which is defined as

$$\mathbf{I}_{obs} \equiv \mathbf{I}_o(\boldsymbol{\theta}|\mathbf{Y}_{obs}) = -\frac{\partial^2 logf(\mathbf{Y}_{obs}|\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}}.$$

This function is usually very difficult to obtain directly. To overcome this difficulty, Meng and Rubin (1991) derived the following formula for the asymptotic variance-covariance matrix $\mathbf{V}$,

$$\mathbf{V} = \mathbf{I}_{oc}^{-1}(\mathbf{I} - \mathbf{DM})^{-1}, \tag{2.2}$$

where $\mathbf{I}_{oc} = E[\mathbf{I}_o(\boldsymbol{\theta}|\mathbf{Y})|\mathbf{Y}_{obs}, \boldsymbol{\theta}]|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*}$ and $\mathbf{I}_o(\boldsymbol{\theta}|\mathbf{Y}) = -\frac{\partial^2 logf(\mathbf{Y}|\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\cdot\partial\boldsymbol{\theta}}$ is the complete-data observed information matrix. $\mathbf{DM}$ is the Jacobian matrix of $\mathbf{M}$ at $\boldsymbol{\theta}^*$ with elements

$$DM_{ji} = \left(\frac{\partial M_j(\boldsymbol{\theta})}{\partial\theta_i}\right)\Bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*}, \tag{2.3}$$

for $i, j = 1, \cdots, d$. In their calculation of $\mathbf{V}$, $\mathbf{I}_{oc}$ is assumed to be obtained directly from the complete data. To estimate $\mathbf{DM}$, they used numerical (forward) differentiation to estimate the first derivative of the $\mathbf{M}$ function. Specifically, using an EM sequence not equal to the one for calculating the EM estimator, $\boldsymbol{\theta}^{(t)}(i) = \left(\theta_1^*, \theta_2^*, \cdots, \theta_i^{(t)}, \theta_{i+1}^*, \cdots, \theta_d^*\right)$, for $i, j = 1, \cdots, d$, calculate

$$DM_{ji}^{(t)} = \frac{M_j(\boldsymbol{\theta}^{(t)}(i)) - M_j(\boldsymbol{\theta}^*)}{\theta_i^{(t)} - \theta_i^*},$$

stop the d^2 sequence until $DM_{ji}^{(t)}$ converges. The converged value is the estimate of DM_{ji}. Although this is a major breakthrough in EM algorithm, it has been known that SEM still suffers from some limitations as discussed in the introduction.

3. iEM: the Cubic Spline Interpolation EM

To overcome the limitations in the SEM method (Meng and Rubin 1991), we propose to use cubic spline interpolation method, instead of the numerical differentiation, to estimate $\mathbf{I}_{oc}$ and $\mathbf{DM}$ in (2.2). We first introduce one dimensional cubic spline interpolation and two dimensional cubic spline interpolation. Then we describe the applications of these interpolations to build iEM method.

3.1 One Dimensional Cubic Spline Interpolation

Spline interpolation (Ahlberg, Nilson and Walsh 1967) is first used by draftsmen to get a smooth curve passing through specified points. With the recent development of computer technology, its applications extended to many areas including mathematics, statistics, computer science and engineering (Su and Liu 2014). The properties of spline interpolation are thoroughly investigated (Birkhoff and De Boor 1964, Hall and Meyer 1976, Dolezal and Tewarson 1982, Dubeau and Savoie 1996). There are many interpolation methods available (Akima 1970, 1974, 1991, Franke 1982, Getreuer 2011, Schumaker 2015). We choose cubic spline interpolation method because the cubic spline interpolatory function has both continuous first and second derivatives, which are required to estimate the asymptotic variance-covariance matrix of the EM estimators. In addition, it has strong convergence property in mathematical sense.

In an interval $[a\ b]$, let $a = x_0 < x_1 < \cdots < x_n = b$ be a uniform mesh with mesh size $h_{\equiv}|x_i - x_{i-1}|, i = 1, \cdots, n$, and denote this mesh set as $\chi = \{a = x_0, x_1, \cdots, x_n = b\}$. A cubic spline $S(x)$ is defined as a C^2 piecewise cubic polynomial between consecutive knots x_i and it satisfies that $S(x)$ is cubic in each subinterval $[x_{i-1}, x_i]$ and $S(x)$ is continuous with continuous first and second derivatives in the support of x. To be specific,

$$S(x) = \begin{cases} s_1(x) = a_1 + b_1(x - x_0) + c_1(x - x_0)^2 + d_1(x - x_0)^3, & x \in [x_0, x_1], \\ s_2(x) = a_2 + b_2(x - x_1) + c_2(x - x_1)^2 + d_2(x - x_1)^3, & x \in [x_1, x_2], \\ \cdots \\ s_n(x) = a_n + b_n(x - x_{n-1}) + c_n(x - x_{n-1})^2 + d_n(x - x_{n-1})^3, & x \in [x_{n-1}, x_n]. \end{cases}$$

In short, the interpolation value at x is $s_i(x) = \mathbf{A}_i\mathbf{X}^T, x \in [x_{i-1}, x_i]$, where $\mathbf{A}_i = [a_i\ b_i\ c_i\ d_i]$ is the local coefficient and $\mathbf{X} = [1\ (x - x_{i-1})\ (x - x_{i-1})^2\ (x - x_{i-1})^3]$.

To interpolate a function $g(x)$ using a cubic spline function, suppose the data are $\{x_i, g(x_i)\}$, $i = 0, 1, \cdots, n$ and $h \equiv |x_i - x_{i-1}|$. $a = x_0 < x_1 < \cdots < x_n = b$ are considered as knots. To obtain a unique interpolatory cubic spline $S^g(x)$, we need $4n$ constraints on the $4n$ coefficients, $a_i, b_i, c_i, d_i, i = 1, \cdots, n$ in $S(x)$.

First, the cubic spline function requires that $S(x)$ is continuous and has continuous first and second derivatives. Notice that within each interval $[x_{i-1}, x_i]$, the corresponding cubic polynomial is continuous and hence has continuous first and second derivatives. However, the function $S(x)$ has two cubic pieces incident at the interior knot x_i, the function $s_i(x)$ to the left of x_i and the function $s_{i+1}(x)$ to the right of x_i. Therefore, we need constraints to ensure the function itself and its first and second derivatives are continuous at the interior knots. That is, $s_i(x)$ and $s_{i+1}(x)$, as well as their respective first and second derivatives, should match in value at x_i. Mathematically, the following conditions should be imposed on $S(x)$, $s_i(x_i) = s_{i+1}(x_i)$, $s_i^{(1)}(x_i) = s_{i+1}^{(1)}(x_i)$, and $s_i^{(2)}(x_i) = s_{i+1}^{(2)}(x_i)$, for $i = 1, \cdots, n - 1$. The same constraints can be expressed in the following matrix form, $\mathbf{A}_i\mathbf{X}_i^T = a_{i+1}$, $\mathbf{A}_i\mathbf{X}_i^{(1)} = b_{i+1}$, $\mathbf{A}_i\mathbf{X}_i^{(2)} = 2c_{i+1}$, where $\mathbf{X}_i = [1\ (x_i - x_{i-1})\ (x_i - x_{i-1})^2\ (x_i - x_{i-1})^3]$, $\mathbf{X}_i^{(1)} = [0\ 1\ 2(x_i - x_{i-1})\ 3(x_i - x_{i-1})^2]$ and $\mathbf{X}_i^{(2)} = [0\ 0\ 2\ 6(x_i - x_{i-1})]$ are the first and second derivatives of $\mathbf{X}_i$. It is obvious that we have $3(n - 1)$ linear constraints imposed on the coefficients to ensure $S(x)$ is continuous and has both continuous first and second derivatives.

Second, to interpolate the data, we require the $S(x)$ function passes through the data points $g(x_i)$. That is, for $i = 0, 1, \cdots, n$, $S(x_i) = g(x_i)$, which is equivalent to $\mathbf{A}_i\mathbf{X}_{i-1}^T = g(x_{i-1})$, and $\mathbf{A}_n\mathbf{X}_n^T = g(x_n)$. Clearly, this imposes $n + 1$ linear constraints on the coefficients. Now, we have $4n - 2$ linear constraints in total. Two more constraints are needed to obtain a unique cubic spline interpolatory function. These two constraints are usually imposed near the ends of the interval

$[a, b]$, therefore they are referred to as end conditions. Different end conditions (Holmes 2016) have been proposed. Because other conditions require known first or second derivatives of $g(a)$ and $g(b)$, we use the "not-a-knot" condition. It adds two linear constraints on the third derivatives, i.e., $s_1^{(3)}(x_1) = s_2^{(3)}(x_1)$ and $s_{n-1}^{(3)}(x_{n-1}) = s_n^{(3)}(x_{n-1})$, which are equivalent to $d_1 = d_2$, and $d_{n-1} = d_n$, respectively.

With the above $4n$ linear constraints, the system can be solved to obtain the unique cubic spline interpolatory function $S^g(x)$. Because all constraints are linear in the coefficients, it is obvious that

$$\mathbf{A}^T = \mathbf{k}(\chi)\mathbf{g}^T(\mathbf{x})$$

where $\mathbf{A} = [\mathbf{A}_1 \ \cdots \ \mathbf{A}_n]$, $\mathbf{k}(\chi)$ is the inverse of the matrix constructed by the constraints related to x_i $(i = 0, 1, \cdots, n)$ in the mesh set χ, and $\mathbf{g}(\mathbf{x})$ is the vector $[g(x_0) \ g(x_1) \ \cdots \ g(x_n)]$. Let $x \in [x_{i-1}, x_i]$ and $\mathbf{A}_i^T = [\mathbf{k}(\chi)\mathbf{g}^T(\mathbf{x})]_i$, then the interpolatory cubic spline function is

$$s_i^g(x) = \mathbf{X}\mathbf{A}_i^T = \mathbf{X}[\mathbf{k}(\chi)\mathbf{g}^T(\mathbf{x})]_i. \tag{3.1}$$

Now it is easy to obtain the first and second derivatives of $S^g(x)$. For $x \in [x_{i-1}, x_i]$, the first and the second derivatives are

$$\begin{aligned} s_i^{g(1)}(x) &= [1 \ 2(x - x_{i-1}) \ 3(x - x_{i-1})^2][b_i \ c_i \ d_i]^T, \text{ and} \\ s_i^{g(2)}(x) &= [2 \ 6(x - x_{i-1})][c_i \ d_i]^T. \end{aligned}$$

3.2 Two Dimensional Cubic Spline Interpolation on a Grid

When we seek an estimate of a function with two independent variables $g(x_1, x_2)$, we use the bicubic spline interpolation method. Bicubic spline interpolation is an extension of one dimensional cubic spline interpolation to two variables on a two dimensional grid. The basic idea is to break up the problem into a succession of one dimensional interpolations.

Suppose we have a two dimensional grid $\{x_{1i}, x_{2j}\}$, where $i = 0, 1, \cdots, M$, and $j = 0, 1, \cdots, K$. We know the function values $g_{ij} = g(x_{1i}, x_{2j})$ at these grid points. To interpolate the function $g(x_1, x_2)$ using bicubic spline, we first construct K one-dimensional cubic spline interpolation in x_1 direction across the $g(x_{1i}, x_{2j}), i = 0, 1, \cdots, M,$ values for each fixed x_{2j} to get a vector of the estimated function values $s_i^g(x_1, x_{2j}), j = 0, 1, \cdots, K, x_1 \in [x_{1(i-1)}, x_{1i}]$. To be specific, let $\mathbf{A}_j = [a_{1j}\ b_{1j}\ c_{1j}\ d_{1j}\ \cdots\ a_{Mj}\ b_{Mj}\ c_{Mj}\ d_{Mj}]$ and $A_{ij} = [a_{ij}\ b_{ij}\ c_{ij}\ d_{ij}]$. Then for each $j = 0, 1, \cdots, K$,

$$\mathbf{A}_j^T = \mathbf{k}(\chi_1)\mathbf{g}^T(\mathbf{x}_1, x_{2j}), \tag{3.2}$$

where $\mathbf{g}(\mathbf{x}_1, x_{2j}) = [g(x_{10}, x_{2j})\ g(x_{11}, x_{2j})\ \cdots\ g(x_{1M}, x_{2j})]$. $\mathbf{A}_j$ is estimated using the one dimensional cubic spline interpolation of the data $\{(x_{1i}, x_{2j}), g(x_{1i}, x_{2j}), i = 0, 1, \cdots, M\}$ introduced in Section 3.1. $\mathbf{k}(\chi_1)$ is the inverse of the matrix constructed by the constraints related to $x_{1i}, i = 0, 1, \cdots, M$ in the x_1-mesh set $\chi_1 = \{x_{1i}, i = 0, \cdots, M\}$. Then the interpolation value at $\{x_1, x_{2j}\}$ is

$$\begin{aligned} s_i^g(x_1, x_{2j}) &= [1\ (x_1 - x_{1(i-1)})\ (x_1 - x_{1(i-1)})^2\ (x_1 - x_{1(i-1)})^3][a_{ij}\ b_{ij}\ c_{ij}\ d_{ij}]^T \\ &= \mathbf{X}_1\mathbf{A}_{ij}^T = \mathbf{X}_1[\mathbf{k}(\chi_1)\mathbf{g}^T(\mathbf{x}_1, x_{2j})]_i, \end{aligned}$$

where $\mathbf{A}_{ij}^T = [\mathbf{k}(\chi_1)\mathbf{g}^T(\mathbf{x}_1, x_{2j})]_i$. Next we do another one dimensional cubic spline interpolation across these vector values $s_i^g(x_1, x_{2j}), j = 0, 1, \cdots, K$, to obtain the estimated function $s^g(x_1, x_2)$. Specifically, let $\mathbf{A}' = [a'_1\ b'_1\ c'_1\ d'_1\ \cdots\ a'_K\ b'_K\ c'_K\ d'_K]$ and $A'_j = [a'_j\ b'_j\ c'_j\ d'_j]$. Then

$$\mathbf{A}'^T = \mathbf{k}(\chi_2)\mathbf{s}_i^g(x_1, \mathbf{x}_2)^T, \tag{3.3}$$

where $\mathbf{s}_i^g(x_1, \mathbf{x}_2) = [s_i(x_1, x_{20})\ s_i(x_1, x_{21})\ \cdots\ s_i(x_1, x_{2K})]$ and $\mathbf{k}(\chi_2)$ is the inverse of the matrix constructed by the constraints related to the x_2-mesh set $\chi_2 = \{x_{2j}, j = 0, 1, \cdots, K\}$. Let $\mathbf{B} = \mathbf{k}(\chi_2)\mathbf{g}^T(\mathbf{x}_1, \mathbf{x}_2)\mathbf{k}^T(\chi_1)$, $\mathbf{x}_2 = [1\ (x_2 - x_{2(j-1)})\ (x_2 - x_{2(j-1)})^2\ (x_2 - x_{2(j-1)})^3]$, $\mathbf{x}_1 = [1\ (x_1 - x_{1(i-1)})\ (x_1 - x_{1(i-1)})^2\ (x_1 - x_{1(i-1)})^3]$, then the interpolation value at $\{x_1, x_2\}, x_2 \in [x_{2(j-1)}, x_{2j}], x_1 \in [x_{1(i-1)}, x_{1i}]$ is

$$\begin{aligned} s^g(x_1, x_2) &= \mathbf{x}_2[a'_j\ b'_j\ c'_j\ d'_j]^T\mathbf{x}_1^T, \\ &= \mathbf{x}_2[\mathbf{k}(\chi_2)\mathbf{g}^T(\mathbf{x}_1, \mathbf{x}_2)\mathbf{k}^T(\chi_1)]_{ij}\mathbf{x}_1^T \\ &= \mathbf{x}_2\mathbf{B}_{ij}\mathbf{x}_1^T \end{aligned} \tag{3.4}$$

where $\mathbf{B}_{ij}$ is $\mathbf{B}[(4j+1) : (4j+4), (4i+1) : (4i+4)]$, and

$$\mathbf{g}(\mathbf{x_1}, \mathbf{x_2}) = \begin{bmatrix} g(x_{10}, x_{20}) & g(x_{10}, x_{21}) & \cdots & g(x_{10}, x_{2K}) \\ g(x_{11}, x_{20}) & g(x_{11}, x_{21}) & \cdots & g(x_{11}, x_{2K}) \\ \cdots & \cdots & \cdots & \cdots \\ g(x_{1M}, x_{20}) & g(x_{1M}, x_{21}) & \cdots & g(x_{1M}, x_{2K}) \end{bmatrix}$$

Therefore, the first and second derivatives with respect to x_1 are respectively

$$s^{g(1,0)}(x_1, x_2) = \mathbf{x}_2\mathbf{B}_{ij}[0\ 1\ 2(x_1 - x_{1(i-1)})\ 3(x_1 - x_{1(i-1)})^2]^T$$

and

$$s^{g(2,0)}(x_1, x_2) = \mathbf{x}_2\mathbf{B}_{ij}[0\ 0\ 2\ 6(x_1 - x_{1(i-1)})]^T.$$

Similarly, the first and second derivatives with respect to x_2 are respectively

$$s^{g(0,1)}(x_1, x_2) = [0\ 1\ 2(x_2 - x_{2(j-1)})\ 3(x_2 - x_{2(j-1)})^2]\mathbf{B}_{ij}\mathbf{x}_1^T$$

and

$$s^{g(0,2)}(x_1, x_2) = [0\ 0\ 2\ 6(x_2 - x_{2(j-1)})]\mathbf{B}_{ij}\mathbf{x}_1^T.$$

The cross derivative $\partial s^g(x_1, x_2)/\partial x_1 \partial x_2$ is

$$s^{g(1,1)}(x_1, x_2) = [0\ 1\ 2(x_2-x_{2(j-1)})\ 3(x_2-x_{2(j-1)})^2]\mathbf{B}_{ij}[0\ 1\ 2(x_1-x_{1(i-1)})\ 3(x_1-x_{1(i-1)})^2]^T.$$

3.3 iEM: Estimate the Variance-Covariance Matrix in EM Algorithm

To estimate the asymptotic variance-covariance matrix, we need to estimate $\mathbf{I}_{oc}$ and $\mathbf{DM}$ in (2.2), which may be difficult to calculate directly. However, the elements of these two matrices are derivatives of known functions $Q(\boldsymbol{\theta}|\hat{\boldsymbol{\theta}}^*) \equiv E[logf(\mathbf{Y}|\boldsymbol{\theta})|\mathbf{Y}_{obs}, \boldsymbol{\theta}^*]$ and $\mathbf{M}(.)$ in the EM algorithm, respectively. Therefore, the iEM is developed to estimate these two functions, which allows the derivatives to be calculated easily.

Note as in Meng and Rubin (1991), we are estimating the complete-data observed information matrix, $\mathbf{I}_o(\boldsymbol{\theta}|\mathbf{Y})$, instead of the Fisher information matrix $\mathbf{I}_{oc}$. Therefore, the iEM is to estimate the elements of $\mathbf{DM}$ by the first derivatives of the one dimensional cubic spline interpolatory functions of the $\mathbf{M}(.)$. And for $\mathbf{I}_o(\boldsymbol{\theta}|\mathbf{Y})$, the iEM is to estimate the diagonal elements by the second derivatives of one dimensional cubic spline interpolatory functions of $Q(\boldsymbol{\theta}|\hat{\boldsymbol{\theta}}^*)$, and to estimate the off-diagonal elements by the cross derivatives of the bicubic spline interpolatory functions of $Q(\boldsymbol{\theta}|\hat{\boldsymbol{\theta}}^*)$.

4. iEM: Theory and Implementation

4.1 Truncation Error Bounds of iEM

The numerical analysis is affected by a global error including truncation error and rounding error. The following two theorems establish the truncation error bounds of the elements of $\mathbf{DM}$ and $\mathbf{I}_{oc}$ based on the cubic spline interpolation, we defer the theoretical proofs to Appendix.

Theorem 1. *Assume* $M_j(\boldsymbol{\theta}(i)) \in C^4[a,b]$ *and* $\max_{a\le\theta_i\le b}|M_j^{(4)}(\boldsymbol{\theta}(i))| = H_{ij}$. *Furthermore,* $S^{M_j}(\boldsymbol{\theta}(i))$ *is the unique cubic spline interpolatory function with not-a-knot boundary conditions, the uniform mesh* $a=\theta_{i0}\le\theta_{i1}\le\theta_{i2}\le\cdots\le\theta_{iM}=b$ *and* $h=i(m+1)-i(m), m=0,1,\cdots,(M-1)$, *then for* $\theta_i^*\in[a,b]$,

$$|DM_{ij}-S^{M_j(1)}(\boldsymbol{\theta}^*(i))|\le\frac{7}{2}H_{ij}h^3.$$

Theorem 2. *Assume* $Q(\boldsymbol{\theta}(i,j))\in C^{4,4}[\Omega]$ *where* $\Omega=[a_1,b_1]\times[a_2,b_2]$, *and* $\max_{a_1\le\theta_i\le b_1}Q^{(4,0)}(\boldsymbol{\theta}(i,j))=H_i$, $\max_\Omega Q^{(4,4)}(\boldsymbol{\theta}(i,j))=H_{ij}$, $\max_{a_2\le\theta_j\le b_2}Q^{(0,4)}(\boldsymbol{\theta}(i,j))=H_j$. *The uniform mesh for* θ_i *is* $a_1=\theta_{i0}\le\theta_{i1}\le\theta_{i2}\le\cdots\le\theta_{iM}=b_1$ *and* $h_1=i(m+1)-i(m), m=0,1,\cdots,(M-1)$. *The uniform mesh for* θ_j *is* $a_2=\theta_{j0}\le\theta_{j1}\le\theta_{j2}\le\cdots\le\theta_{jK}=b_2$ *and* $h_2=j(k+1)-j(k), k=0,1,\cdots,(K-1)$.

(i) $S^Q(\boldsymbol{\theta}(i))$ *is the unique cubic spline interpolatory function of* $Q(\boldsymbol{\theta}(i))$ *with not-a-knot boundary conditions, then for* $\theta_i^*\in[a_1,b_1]$, *the error bound for the ith diagonal element of* I_{oc} *is*

$$|I_{oc}^{ii}-S^{Q(2)}(\boldsymbol{\theta}^*(i))|\le\frac{7}{2}H_ih_1^2.$$

(ii) $S^Q(\boldsymbol{\theta}(i,j))$ *is the unique bicubic spline interpolatory function of* $Q(\boldsymbol{\theta}(i,j))$ *with not-a-knot boundary conditions, then for* $(\theta_i^*,\theta_j^*)\in\Omega$, *the error bound for the* (i,j)*th off-diagonal element of* I_{oc} *is*

$$|I_{oc}^{ij}-S^{Q(1,1)}(\boldsymbol{\theta}^*(i,j))|\le\frac{7}{2}H_ih_1^3+\frac{49}{4}H_{ij}h_1^3h_2^3+\frac{7}{2}H_jh_2^3.$$

4.2 Rounding Error of iEM

Now, we derive the rounding error for iEM method. It is shown in Appendix B that the rounding error (RE) for the local coefficients in the intervals of a one dimensional cubic spline interpolation are

$$RE(a_i)\propto O(\epsilon), RE(b_i)\propto O(\epsilon/h), RE(c_i)\propto O(\epsilon/h^2) \text{ and } RE(d_i)\propto O(\epsilon/h^3), \tag{4.1}$$

where $i=1,\cdots,n$ are the local coefficients, ϵ is the machine accuracy and h is the uniform mesh size. Based on these rounding errors, it is easy to see that the rounding errors for the first and second derivatives of a cubic spline interpolatory function are $O(\epsilon/h)$ and $O(\epsilon/h^2)$, respectively. For the cross derivative, we use two one dimensional cubic spline interpolation in succession. For the second interpolation, the first derivatives of the one dimensional cubic spline interpolatory functions are used as the function values. Therefore, the rounding error is also $O(\epsilon/h^2)$.

4.3 Knots and Mesh Selection in iEM

In practice, the cubic spline interpolation method requires selecting an interval and mesh for each parameter. We use a uniform mesh not only because it is convenient but also because it results in estimators with smaller truncation error than non-uniform mesh (De Boor 1984, Tyagi 2016). The computational time of the interpolation increases with the increase of the number of knots. However, as shown in the Theorems above, the number of knots does not affect the truncation error and the rounding error. Therefore, we can use a small number of knots to improve computational efficiency. However, as discussed in Tyagi (2016), the interpolatory cubic spline function and its derivatives at the boundary (around two end knots) of the interval might have larger truncation errors than those not at the boundary of the interval, we therefore put the EM estimator value as a knot in the middle of the interval and use five knots.

To obtain the optimal mesh size for the first derivatives, we make use of the conclusions from the Theorem that the truncation error is $O(h^3)$ and the rounding error is $O(\epsilon/h)$. Therefore, for the case of double precision where $\epsilon = 10^{-16}$, the optimal mesh size minimizing the global error is 10^{-4}. Similarly, we can derive the optimal mesh size for the second derivatives which is also 10^{-4}. For cross derivative, the optimal mesh size is shown to be $10^{-16/5}$.

5. Comparison between iEM and SEM

Because the SEM method cannot estimate $\mathbf{I}_{oc}$, we compare these two algorithms in estimating $\mathbf{DM}$.

5.1 Global Error

As shown in the Theorems 1 and 2, the truncation error of the iEM method (**TEi**) is $O(h^3)$. It is well known that the truncation error of forward differentiation used in SEM (**TEs**) is $O(h)$. Therefore, when the same small mesh size h is used, the iEM estimator of $\mathbf{DM}$ has significantly smaller truncation error than the SEM estimator.

To show the relationship between **TEi** and **TEs**, we put $\boldsymbol{\theta}^*$ as a knot in iEM. For each $M_j, j = 1, \cdots, d$, and $\theta_i^*, i = 1, \cdots, d$, let $[al_j \ bl_j \ cl_j \ dl_j]$ be the local coefficients of the cubic spline in the subinterval to the left of θ_i^* and $[ar_j \ br_j \ cr_j \ dr_j]$

be the local coefficients of the cubic spline in the subinterval to the right θ_i^*. It is easy to show that

$$TEs_j = TEi_j + \frac{M_j(\theta_i^* + h) - M_j(\theta_i^*)}{h} - br_j.$$

Note $\frac{M_j(\theta_i^*+h)-M_j(\theta_i^*)}{h}$ is the slope of the linear interpolation that connects the two points $M_j(\theta_i^* + h)$ and $M_j(\theta_i^*)$, while br_j is the slope of the cubic spline interpolation at θ_i^*. It is obvious that the difference of these two slopes is $O(h)$.

When the EM algorithm is slow, Jamshidian and Jennrich (2000) showed the truncation error of $\mathbf{V}$ has an *error magnification*, because the truncation error in $\mathbf{DM}$ is multiplied by a large number. In SEM, the size of increments h is chosen automatically by the EM algorithm. When the algorithm is slow, the increment h increases. This results in a larger truncation error in the estimation of $\mathbf{DM}$. Hence the SEM algorithm has larger truncation error in estimating $\mathbf{V}$ when the EM algorithm is slow. In contrast, the iEM chooses the mesh size h by minimizing the global error. It does not increase the truncation error in estimating $\mathbf{DM}$. Consequently, the *error magnification* will have smaller effect on the estimation of $\mathbf{V}$ for the iEM method. In this sense, the iEM method is more stable than the SEM algorithm.

Kim and Shao (2013) showed that asymptotically,

$$\boldsymbol{\theta}^{(t)} = (\mathbf{I} - \mathbf{J}_{mis})\boldsymbol{\theta}^* + \mathbf{J}_{mis}\boldsymbol{\theta}^{(t-1)},$$

where $\mathbf{J}_{mis} \equiv \mathbf{I}_{mis}\mathbf{I}_{oc}^{-1}$ is the fraction of missing information matrix (Rubin, 1987) and $\mathbf{I}_{mis} \equiv \mathbf{I}_{oc} - \mathbf{I}_{obs}$ is the missing information matrix. This means that the true $\mathbf{M}$ function is linear asymptotically. In other words, the forward differentiation used in the SEM algorithm has no truncation error. Similarly, the iEM method has no truncation error as well. Therefore the global error should be small because there is only rounding error for both methods asymptotically.

Regarding the rounding error, both the SEM algorithm and the iEM method have rounding error $O(\epsilon/h)$. Apparently, small h results in larger rounding error.

5.2 Complexity and Computation

Both the SEM and iEM require d^2 sequences of approximate derivatives. For one element in $\mathbf{DM}$, the computational complexity of the iEM method is $O(N)$, where N is the number of knots. However as discussed above, N can be as small as 5, so the iEM is very efficient computationally. In contrast, the SEM algorithm is much more time-consuming due to the iterative nature of the algorithm, especially when the EM algorithm is slow.

In addition, the computational implementation of the iEM method is general enough for any EM algorithm. In other words, the coding of the interpolation will be essentially the same for different problems. However, the SEM algorithm is specific to each problem, i.e., the coding of the E-step and the M-step is specific to each problem. This not only adds complexity to the algorithm, but also increases the chances of programming errors.

6. Numerical Examples

To study the finite-sample performance of the iEM, in this section we apply the iEM method on two classic examples and one real data example.

6.1 Univariate Contaminated Normal Mixture

This example is previously analyzed by Meng and Rubin (1991) and Little and Rubin (1987). Let $x_1, \cdots, x_n$ be an independent sample from the univariate contaminated normal model

$$f(x|\mu, \sigma^2) = (1-\pi)N(\mu, \sigma^2) + \pi N(\mu, \sigma^2/\lambda),$$

where $0 < \pi < 1$ and $\lambda > 0$ are known. The goal is to find the MLE $\boldsymbol{\theta}^* = (\mu^*, \log\sigma^{2*})$ for $\theta = (\mu, \log\sigma^2)$.

To facilitate the calculation of MLE, let

$$\begin{aligned} h(q) = &\quad 1-\pi, \quad \text{if } \ q = 1,\\ = &\quad \pi, \quad \text{if } \ q = \lambda,\\ = &\quad 0, \quad \text{otherwise.} \end{aligned}$$

Consider an independent sample $x_1, \cdots, x_n$ from a population with density

$$x_i|\theta, q_i \sim N(\mu, \sigma^2/q_i), \tag{6.1}$$

where the q_i are unobserved iid random variables with known density $h(q_i)$. The EM algorithm can be employed to compute $\boldsymbol{\theta}^*$. In the E-step, the unobserved q_i can be obtained,

$$\begin{aligned} w_i(\boldsymbol{\theta}^{(t)}) &= E(q_i|x_i, \boldsymbol{\theta}^{(t)})\\ &= \frac{1-\pi+\pi\lambda^{3/2}exp\{(1-\lambda)z_i^2/2\}}{1-\pi+\pi\lambda^{1/2}exp\{(1-\lambda)z_i^2/2\}} \end{aligned} \tag{6.2}$$

where $z_i^2 = (x_i - \mu^{(t)})^2/\sigma^{2(t)}$. In the M-step, we calculate the complete data MLE using

$$\mu^{(t+1)} = \sum_{i=1}^{n} q_i x_i / \sum_{i=1}^{n} q_i, \tag{6.3}$$

and

$$\log \sigma^{2(t+1)} = \log \left\{ \frac{1}{n} \sum_{i=1}^{n} q_i (x_i - \mu^{(t+1)})^2 \right\}. \tag{6.4}$$

It is clear that the $\mathbf{M} = [M_1(\mu, \log \sigma^2)\ M_2(\mu, \log \sigma^2)]^T$ where $M_1(\mu, \log \sigma^2)$ is (6.3) and $M_2(\mu, \log \sigma^2)$ is (6.4) with q_i obtained by (6.2). In addition,

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^*) \equiv E[logf(Y|\boldsymbol{\theta})|Y_{obs}, \boldsymbol{\theta}^*] = -1/2 \sum_{i=1}^{n} \log(2\pi\sigma^2) q_i (x_i - \mu)^2/(2\sigma^2). \tag{6.5}$$

To investigate the performance of the proposed method, we generate a random sample of 100 observations from the distribution (6.1) with $\mu = 0, \sigma^2 = 1, \lambda = 2, \pi = .1$. The EM estimator obtained is $\boldsymbol{\theta}^* = (-0.05357040, -0.02361871)$.

We calculate both the true $\mathbf{DM}$ and $\mathbf{I}_{oc}$ by taking derivatives with respect to the $\mathbf{M}$ and $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^*)$ at $\boldsymbol{\theta}^*$. The corresponding estimated values are calculated by iEM method. In order to investigate the effect of mesh size, several different sizes are used in the iEM method. Table 1 shows the true values and the estimated values for $\mathbf{DM}$ and $\mathbf{I}_{oc}$.

From Table 1, we can see that the optimal mesh size works well for both $\mathbf{DM}$ and $\mathbf{I}_{oc}$. We notice that all mesh sizes work well in estimating $\mathbf{DM}$. The reason, as discussed above (Kim and Shao 2013), is that $\mathbf{M}$ is asymptotically linear, so the truncation error is very small. Consequently, compared with the true values, the estimator of $\mathbf{DM}$ is more accurate than that of $\mathbf{I}_{oc}$ because of the smaller truncation error.

6.2 Poisson Mixture

As pointed out in Section 5.1, when the EM algorithm is slow, the *error magnification* has smaller effect on the iEM method than that on the SEM. We demonstrate this effect by using an example with slow EM algorithm in Jamshidian and Jennrich (2000).

Let $y_1, \cdots, y_n$ be an independent sample from the mixture of two Poisson densities

$$f(y|\theta) = \gamma \frac{e^{-\theta_1}\theta_1^y}{y!} + (1-\gamma)\frac{e^{-\theta_2}\theta_2^y}{y!}$$

where $0 < \gamma < 1$, $\theta_1 > 0$ and $\theta_2 > 0$. Let c_j be the number of y_i equal to $j = 0, 1, 2, \cdots$. Then the log-likelihood for $\boldsymbol{\theta}$ given $y_1, \cdots, y_n$ can be expressed as

$$\sum_{j=0}^{\infty} c_j log\{\gamma e^{-\theta_1}\theta_1^j + (1-\gamma)e^{-\theta_2}\theta_2^j\} + \sum_{j=0}^{\infty} c_j log j!$$

The goal is to find the MLE $\boldsymbol{\theta}^* = (\gamma^*, \theta_1^*, \theta_2^*)$ for $\boldsymbol{\theta} = (\gamma, \theta_1, \theta_2)$.

To apply the EM algorithm, we consider a latent variable z with density

$$\begin{aligned} h(z|\gamma) &= \quad \gamma, \quad \text{if } z = 1, \\ &= 1-\gamma, \quad \text{if } z = 2, \\ &= \quad 0, \quad \text{otherwise.} \end{aligned}$$

Therefore, the density of the complete data $(y_1, z_1), \cdots, (y_n, z_n)$ is

$$\tilde{f}(y, z|\theta) = \left\{ e^{-\theta_z}\theta_z^y / y! \right\} h(z|\gamma). \tag{6.6}$$

In the E-step, we calculate the $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$ function as

$$\begin{aligned} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) &= \sum_{j=0}^{\infty} \left\{ c_j (log\gamma - \theta_1 + j log\theta_1) w(j, \boldsymbol{\theta}^{(t)}) \right\} \\ &+ \sum_{j=0}^{\infty} \left[c_j \{log(1-\gamma) - \theta_2 + j log\theta_2\}\{1 - w(j, \boldsymbol{\theta}^{(t)})\} \right] \end{aligned} \tag{6.7}$$

where $w(j, \boldsymbol{\theta}^{(t)}) = \left\{ \gamma^{(t)} e^{-\theta_1^{(t)}} \theta_1^{(t)j} / j! \right\} f\left(j|\boldsymbol{\theta}^{(t)} \right)$. In the subsequent M-step, the complete data MLE is obtained as

$$\gamma^{(t+1)} = \sum_{j=0}^{\infty} c_j w(j, \boldsymbol{\theta}^{(t)}) / \sum_{j=0}^{\infty} c_j, \tag{6.8}$$

$$\theta_1^{(t+1)} = \sum_{j=0}^{\infty} j c_j w(j, \boldsymbol{\theta}^{(t)}) / \sum_{j=0}^{\infty} c_j w(j, \boldsymbol{\theta}^{(t)}), \tag{6.9}$$

$$\theta_2^{(t+1)} = \sum_{j=0}^{\infty} j c_j (1 - w(j, \boldsymbol{\theta}^{(t)})) / \sum_{j=0}^{\infty} c_j (1 - w(j, \boldsymbol{\theta}^{(t)})). \tag{6.10}$$

Obviously, the **M** functions are (6.8) to (6.10).

To investigate the performance of the proposed method, we generate a random sample of 2000 observations from the distribution (6.6) with $\gamma = 0.3, \theta_1 = 1,$

and $\theta_2 = 5$. The EM estimator is $\boldsymbol{\theta}^* = (0.285070, 0.855829, 4.917305)$. The true and estimated values by iEM method of $\mathbf{DM}$ and $\mathbf{I}_{oc}$ are provided in Table 2.

Compared with Table 1, the estimated $\mathbf{DM}$ has similar accuracy as those in Table 1. Because the $\mathbf{I}_{oc}$ in Table 2 is much larger than those in Table 1, we compare the relative accuracy of the estimators. The relative accuracy for each element is defined as

$$IR_{oc}^{ij} = \frac{I_{oc}^{ij} - S^{Q(1,1)}(\boldsymbol{\theta}^*(i,j))}{I_{oc}^{ij}},$$

where I_{oc}^{ij} is the true value and $S^{Q(1,1)}(\boldsymbol{\theta}^*(i,j))$ is the estimated value based on the optimal mesh size. The largest relative accuracy of $\mathbf{I}_{oc}$ in both Table 1 and Table 2 is in the order of 10^{-7}. This indicates the slow EM algorithm does not affect the estimation accuracy for iEM method.

6.3 A Real Data Example

Hartley (1958) studied the pollution of seeds of Phleum pratense by the presence of a few noxious weed seeds. It is observed the number of the noxious weed seeds ranges from 2 to 9 in 78 quarter-ounce samples. The corresponding frequencies are 26, 16, 18, 9, 3, 5, 0, and 1, respectively. In addition, it is known that the numbers of the noxious weed seeds are 0 and 1 in some samples, but their frequencies are unknown. The goal is to estimate the average number of the noxious weed seeds. We will apply the proposed iEM algorithm on the data used in Hartley (1958) to estimate the average number of noxious weed seeds.

Let X represent the number of the noxious weed seeds. It follows Poisson distribution with probability density function

$$f(X \mid \theta) = \frac{exp(-\theta)\theta^X}{X!}.$$

Let $x_i, i = 1, \cdots, 8$ be the observed distinct values of X and n_{xi} be the corresponding frequencies. Let n_{00}, n_{01} represent the frequencies of the samples with 0 and 1 noxious weed seeds, respectively. In the E-step, the unobserved n_{00} and n_{01} are calculated by

$$w_{00}(\theta^{(t)}) \equiv E(n_{00} \mid \theta^{(t)}) = 78 \times f_{00}/(1 - f_{00} - f_{01}),$$

$$w_{01}(\theta^{(t)}) \equiv E(n_{01} \mid \theta^{(t)}) = 78 \times f_{01}/(1 - f_{00} - f_{01}),$$

where f_{00} and f_{01} are the probabilities when $X = 0$ and $X = 1$, respectively. In the M-step, we calculate the complete data MLE using

$$\theta^{(t+1)} = (\sum_{i=1}^{8} x_i n_{xi} + w_{01}(\theta^{(t)}))/(78 + w_{00}(\theta^{(t)}) + w_{01}(\theta^{(t)})). \tag{6.11}$$

Then the M function is (6.11) and the expected log-likelihood is

$$Q(\theta \mid \theta^*) = -(78 + n_{00} + n_{01})\theta^* + (\sum_{i=1}^{8} x_i n_{xi} + w_{01}) log(\theta^*) - \sum_{i=1}^{8} n_{xi} log(x_i!)).$$

The estimated mean number of noxious weed seeds by the proposed iEM estimator is 3.025 and the corresponding estimated variance is 0.036. The estimated variance is very close to the true asymptotic variance of the EM estimator (0.033), calculated based on the inverse of the observed information matrix. It shows that a very accurate variance estimate can be obtained by the proposed variance estimator in a more computationally efficient way.

7. Discussions and Conclusions

In this paper, we proposed a novel iEM algorithm using cubic spline interpolation method to directly estimate the asymptotic variance-covariance matrix in the classical EM algorithm. This method is general enough for many methods of calculating the asymptotic variance-covariance matrix, such as Oaks (1992) formula and the second derivative of the observed data likelihood. We used the form in Meng and Rubin (1991) because we would like to compare the newly proposed iEM method with the best SEM method. Future research can be done to investigate its performance on other forms of the asymptotic variance-covariance matrix.

Compared with other numerical differentiation methods, such as forward differentiation and Richardson extrapolation, the iEM is more accurate. The SEM uses forward differentiation, which can be derived from Lagrange polynomial interpolation with two points. Similarly, the Richardson extrapolation can be obtained by Lagrange polynomial interpolation with five points. In contrast, the iEM uses cubic spline interpolation. Spline functions are usually superior to Lagrangian polynomial fits in respect to interpolation (Forsythe, Malcolm and Moler 1977, Mcnamee 1985). One exception to this preference is when the Lagrangian polynomial performs better than the splines for moderate to large mesh sizes and fairly accurate to accurate data (Mcnamee 1985). In the iEM method, we can usually choose the optimal mesh size and it is small, so spline interpolation usually performs better when calculating asymptotic variance-covariance matrix. Therefore, we prefer cubic spline interpolation method to the numerical differentiation method.

Acknowledgement

We are grateful to the editor, associate editor, and one referee for their insightful comments, which significantly improved this manuscript.

This work is based upon research supported by the South Africa National Research Foundation (NRF) and South Africa Medical Research Council (SAMRC) (South Africa DST-NRF-SAMRC SARChI Research Chair in Biostatistics to Ding-Geng Chen, Grant number 114613). Opinions expressed and conclusions arrived at are those of the author and are not necessarily to be attributed to the NRF and SAMRC.

Appendix A

Proof of Theorem 1:

Tyagi (2016) showed for $\theta_i \in [a, b]$,

$$|M_j^{(2)}(\boldsymbol{\theta}(i)) - S^{M_j(2)}(\boldsymbol{\theta}(i))| \leq \frac{7}{2} H_{ij} h^2.$$

Now consider $e(t) = M_j(t) - S^{M_j}(t)$, because $e(\theta_{im}) = 0, m = 0, 1, \cdots, M$, by Rolle's theorem there exist ξ_{im} in each subinterval $[\theta_{i(m-1)}, \theta_{im}], m = 1, \cdots, M$, such that $e^{(1)}(\xi_{im}) = 0$. Then for every $\theta \in [\theta_{i(m-1)}, \theta_{im}]$, we have $|\theta - \xi_{im}| \leq h$. Write $e^{(1)}(\theta)$ as

$$|M_j^{(1)}(\boldsymbol{\theta}(i)) - S^{M_j(1)}(\boldsymbol{\theta}(i))| = \int_{\xi_{im}}^{\theta} (M_j^{(2)}(t) - S^{M_j(2)}(t))dt$$

Thus,

$$|e^{(1)}| \leq |\theta - \xi_{im}| ||e^{(2)}||_\infty \leq \frac{7}{2} H_{ij} h^3,$$

and this completes the proof of Theorem 1.

Proof of Theorem 2:

The proof of (i) in Theorem 2 is obvious based on Theorem 5 of Tyagi (2016). To prove (ii), we make use of equations (3.1), (3.2), (3.3) and (3.4), and write

$$S_i^Q(\boldsymbol{\theta}(i,j)) = \sum_{m=0}^{M} Q(\boldsymbol{\theta}(im,j))\phi_i(\boldsymbol{\theta}(im)),$$

$$S_j^Q(\boldsymbol{\theta}(i,j)) = \sum_{k=0}^{K} Q(\boldsymbol{\theta}(i,jk)\varphi_j(\boldsymbol{\theta}(jk)),$$

$$S^Q(\boldsymbol{\theta}(i,j))) = \sum_{m=0}^{M}\sum_{k=0}^{K} Q(\boldsymbol{\theta}(im,jk))\phi_i(\boldsymbol{\theta}(im))\varphi_j(\boldsymbol{\theta}(jk)),$$

where $S_i^Q(\boldsymbol{\theta}(i,j))$ is one dimensional cubic spline interpolatory function based on $\{\boldsymbol{\theta}(im,j),\, Q(\boldsymbol{\theta}(im,j))\}$ for fixed j; $S_j^Q(\boldsymbol{\theta}(i,j))$ is one dimensional cubic spline interpolatory function based on $\{\boldsymbol{\theta}(i,jk), Q(\boldsymbol{\theta}(i,jk))\}$ for fixed i; $S^Q(\boldsymbol{\theta}(i,j)))$ is bicubic spline interpolatory function based on $\{\boldsymbol{\theta}(im,jk), Q(\boldsymbol{\theta}(im,jk))\}$. $\phi_i(\boldsymbol{\theta}(im))$ is a function based on θ_i and $\varphi_j(\boldsymbol{\theta}(jk))$ is based on θ_j. Then for $(\theta_i,\theta_j) \in \Omega$,

$$Q^{(1,1)}(\boldsymbol{\theta}(i,j)) - S^{Q(1,1)}(\boldsymbol{\theta}(i,j)) = (Q^{(1,1)}(\boldsymbol{\theta}(i,j)) - S_j^{Q(1,1)}(\boldsymbol{\theta}(i,j))) + (S_j^{Q(1,1)}(\boldsymbol{\theta}(i,j)) - S^{Q(1,1)}(\boldsymbol{\theta}(i,j))) \tag{8.1}$$

The first term of (8.1) is obvious based on theorem 1, which is

$$Q^{(1,1)}(\boldsymbol{\theta}(i,j)) - S_j^{Q(1,1)}(\boldsymbol{\theta}(i,j)) \leq \frac{7}{2}H_j h_2^3.$$

For the second term of (8.1), let $R_1(\boldsymbol{\theta}(i,jk)) = Q(\boldsymbol{\theta}(i,jk) - \sum_{m=0}^{M} Q(\theta(im,jk))\phi_i(\boldsymbol{\theta}(im))$ and $\varphi_{R_1} = \sum_{k=0}^{K} R_1(\boldsymbol{\theta}(i,jk))\varphi_j(\boldsymbol{\theta}(jk))$,

$$S_j^Q(\boldsymbol{\theta}(i,j)) - S^Q(\boldsymbol{\theta}(i,j)) = \sum_{k=0}^{K} R_1(\boldsymbol{\theta}(i,jk))\varphi_j(\boldsymbol{\theta}(jk)),$$

and

$$\begin{aligned} S_j^{Q(1,1)}(\boldsymbol{\theta}(i,j)) - S^{Q(1,1)}(\boldsymbol{\theta}(i,j)) &= (\varphi_{R_1}^{(1,1)} - R_1^{(1,1)}) + R_1^{(1,1)} \\ &\leq \frac{49}{4}\max_{\Omega}|Q^{(4,4)}|h_1^3 h_2^3 + R_1^{(1,1)} \end{aligned}$$

Then the proof of Theorem 2 is complete.

Appendix B

Let $x_1 < x_2 < x_3$ represent the uniform mesh with mesh size $h = x_i - x_{i-1}, i = 2, 3$. The cubic spline interpolation can be written as

$$a_1 + b_1(x - x_1) + c_1(x - x_1)^2 + d_1(x - x_1)^3 = f(x), x \in [x_1, x_2],$$

and

$$a_2 + b_2(x - x_1) + c_2(x - x_1)^2 + d_2(x - x_1)^3 = f(x), x \in [x_2, x_3].$$

Now evaluate $f(x), f^{(1)}, f^{(2)}$ at x_2, which belongs to both $[x_1, x_2]$ and $[x_2, x_3]$. We have, respectively

$$a_1 + b_1 h + c_1 h^2 + d_1 h^3 = a_2, \tag{9.1}$$

$$b_1 + 2c_1 h + 3d_1 h^2 = b_2, \tag{9.2}$$

$$2c_1 + 6d_1 h = 2c_2. \tag{9.3}$$

From (9.3), we obtain $d_1 = \frac{c_2 - c_1}{3h}$. Similarly, from (9.2) and (9.1), we derive $c_1 + c_2 = \frac{b_2 - b_1}{h}$ and $b_1 h + (c_2/3 + 2c_1/3)h^2 = a_2 - a_1$. Note $a_i, i = 1, 2, 3$ are the function values and b_i, c_i are the first and second derivatives at the knots. Then if we denote the rounding error for $a_i, i = 1, 2, 3$, as $O(\epsilon)$, we can obtain (4.1) immediately.

References

[1] Ahlberg, JH., Nilson, EN., and Walsh, JL. (1967). *The theory of splines and their applications.* Academic Press, New York.

[2] Akima, H. (1970). A new method of interpolation and smooth curve fitting based on local procedures. *Journal of the association for computing machinery,* **17**, 589-602.

[3] Akima, H. (1974). A method of bivariate interpolation and smooth surface fitting based on local procedures. *Numerical mathematics,* **17**, 18-20.

[4] Akima, H. (1991). A new method of univariate interpolation that has the accuracy of a third-degree polynomial. *ACM transactions on mathematical software,* **17**, 341-366.

[5] Becker, N. G. (1992). Statistical challenges of AIDS. *Australian Journal of Statistics,* **34**, 129-l44.

[6] Belin, T. and Rubin, DB. (1995). A Method for Calibrating False-Match Rates in Record Linkage. *Journal of the American Statistical Association,* **90**, 694-707.

[7] Birkhoff, G. and De Boor, C. (1964). Error bounds for spline interpolation. *Journal of mathematics and mechanics,* **13**, 827-835.

[8] De Boor, C. (1984). *Convergence of cubic spline interpolation with the not-a-knot condition.* Mathematics Research Center preprint, University of Wisconsin, Madison.

[9] Dempster, AP., Laird, NM., and Rubin, DB. (1977). Maximum Likelihood Estimation From Incomplete Data Via the EM Algorithm" (with discussion). *Journal of the Royal Statistical Society, Ser. B,* **39**, 1-38.

[10] Dolezal, V. and Tewarson, R. (1982). Error bounds for spline-on-spline interpolation. *Journal of approximation theory,* **36**, 213-225.

[11] Dubeau, F. and Savoie, J. (1996). Optimal error bounds for quadratic spline interpolation. *Journal of mathematical analysis and applications,* **198**, 49-63.

[12] Forsythe, GE., Malcolm, MA., and Moler, CB. (1977). *Computer Methods for Mathematical Computations.* Prentice-Hall, Englewood Cliffs, NJ.

[13] Franke, R. (1982). Scattered data interpolation: Tests of some methods. *Mathematics of computation,* **157**, 181-200.

[14] Getreuer, P. (2011). Linear methods for image interpolation. *Image processing on line,* **1**, 238-259.

[15] Hall, C. and Meyer, W. (1976). Optimal error bounds for cubic spline interpolation. *Journal of approximation theory,* **16**, 105-122.

[16] Holmes, M. (2016). *Introduction to Scientific Computing and Data Analysis.* Springer, Switzerland.

[17] Jamshidian, M. and Jennrich, R. (2000). Standard errors for EM estimation. *Journal of the Royal Statistical Society, Ser. B,***62**, 257-270.

[18] Little, RJA., and Rubin, DB. (1987). *Statistical Analysis With Missing Data.* New York, John Wiley.

[19] Louis, TA. (1982). Finding the Observed Information Matrix When Using the EM Algorithm. *Journal of the Royal Statistical Society, Ser. B,* **44**, 226-233.

[20] McCulloch, CE. (1998). Review of "EM Algorithm and Extensions". *Journal of the American Statistical Association,* **93**, 403-404.

[21] McNAMEE, J. (1986). Comparison of spline and Lagrangian interpolation. *Journal of Computational and Applied Mathematics,* **16**, 237-240.

[22] Meng, XL. and Rubin, DB. (1991). Using EM to Obtain Asymptotic Variance-Covariance Matrix: The SEM Algorithm. *Journal of the American Statistical Association,* **86**, 899-909.

[23] Meng, L. and Spall. J. (2017). *Efficient computation of the Fisher information matrix in the EM algorithm.* Baltimore, MD.

[24] Oakes, D. (1999). Direct calculation of the information matrix via the EM algorithm. *Journal of the Royal Statistical Society, Ser. B,* **61**, 479-482.

[25] Rubin, DB. (1987). *Multiple imputation for Nonresponse in surveys.* New York, John Wiley.

[26] Schumaker, L (2015). *Spline Functions: Computational Methods.* SIAM-Society for Industrial and Applied Mathematics, Philadelphia, PA.

[27] Segal, MR., Bacchetti, P. and Jewell, NP. (1994). Variances for Maximum Penalized Likelihood Estimates Obtained via the EM Algorithm. *Journal of the Royal Statistical Society. Ser B,* **56**, 345-352.

[28] Smith, C. A. B. (1977). Discussion of "Maximum Likelihood Estimation From Incomplete Data Via the EM Algorithm," by A. P. Dempster, N. M. Laird, and D. B. Rubin. *Journal of the Royal Statistical Society, Ser. B,* **39**, 24-25.

[29] Spall, JC. (1992). Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation. *IEEE Transactions on Automatic Control,* **37**, 332-341.

[30] Spall, JC. (2005). Monte Carlo Computation of the Fisher Information Matrix in Nonstandard Settings. *Journal of Computational and Graphical Statistics,* **14**, 889-909.

[31] Su, BQ. and Liu, DY. (2014). *Computational Geometry: Curve and Surface Modeling.* Academic Press, Inc. San Diego, CA.

[32] Tyagi, H. (2016). *On low dimensional models for functions in high dimensions*. Dissertation. https://doi.org/10.3929/ethz-a-010666821.

Tables

Table 1: Values of “True”, estimated **DM** and $\mathbf{I_{oc}}$ for the univariate contaminated normal example.

h	DM_{11}	DM_{12}	DM_{21}	DM_{22}
True	0.044758195347040	-0.001097609405997	-0.002468769797289	0.038461732545600
10^{-2}	0.044758195312330	-0.001097609402873	-0.002468769787017	0.038461732584883
10^{-3}	0.044758195347042	-0.001097609405996	-0.002468769786110	0.038461732587692
10^{-4}	0.044758195347352	-0.001097609406090	-0.002468769782901	0.038461732593630
10^{-5}	0.044758195343133	-0.001097609402152	-0.002468769723693	0.038461732602420
10^{-6}	0.044758195311436	-0.001097609362199	-0.002468769761824	0.038461732682825

h	I_{oc}^{11}	I_{oc}^{12}	I_{oc}^{22}	
True	-112.4612167553670	0	-50	
10^{-2}	-112.4612167526351	0.0000000004401	-49.9991666463017	
10^{-3}	-112.4612167515691	-0.0000000159872	-49.9999916669940	
10^{-4}	-112.4611976877246	0.0000024079503	-49.9999991632200	
10^{-5}	-112.4612225568687	0.0001717144945	-50.0012475868102	
10^{-6}	-112.3154902413744	-0.0301980662698	-49.7450969309843	

Table 2: Values of “True”, estimated $\mathbf{DM}$ and $\mathbf{I_{oc}}$ for the Poisson mixture example.

h	DM_{11}	DM_{12}	DM_{13}
True	0.3228126176426	0.0735845071369	0.0415333630905
10^{-2}	0.3228123247310	0.0735845067503	0.0415333630917
10^{-3}	0.3228126176134	0.0735845071369	0.0415333630905
10^{-4}	0.3228126176417	0.0735845071367	0.0415333630915
10^{-5}	0.3228126176569	0.0735845071351	0.0415333630955
10^{-6}	0.3228126175968	0.0735845071338	0.0415333630901

h	DM_{21}	DM_{22}	DM_{23}
True	1.0839440899135	0.5213031071185	0.0917347609661
10^{-2}	1.0839434308930	0.5213031054224	0.0917347609764
10^{-3}	1.0839440898480	0.5213031071181	0.0917347609658
10^{-4}	1.0839440899153	0.5213031071191	0.0917347609623
10^{-5}	1.0839440898597	0.5213031071309	0.0917347609416
10^{-6}	1.0839440899167	0.5213031070213	0.0917347610338

h	DM_{31}	DM_{32}	DM_{33}
True	1.4016696744534	0.2101657848169	0.1993704571866
10^{-2}	1.4016684866946	0.2101657832012	0.1993704571911
10^{-3}	1.4016696743348	0.2101657848164	0.1993704571872
10^{-4}	1.4016696744650	0.2101657848189	0.1993704571892
10^{-5}	1.4016696743487	0.2101657847513	0.1993704572635
10^{-6}	1.4016696751628	0.2101657836862	0.1993704570785

h	I_{oc}^{11}	I_{oc}^{12}	I_{oc}^{13}
True	-9813.2975	0	0
10^{-2}	-9804.0666	-0.0000	-0.0000
10^{-3}	-9813.2057	0.0000	-0.0000
10^{-4}	-9813.2963	0.0000	-0.0000
10^{-5}	-9813.2977	-0.0004	0.0043
10^{-6}	-9812.7657	-0.0505	-0.0505

h	I_{oc}^{22}	I_{oc}^{23}	I_{oc}^{33}
True	-666.1844	0	-290.7812
10^{-2}	-666.0934	0.0000	-290.7800
10^{-3}	-666.1835	-0.0000	-290.7812
10^{-4}	-666.1844	0.0000	-290.7813
10^{-5}	-666.1798	-0.0020	-290.8041
10^{-6}	-665.5227	0.0000	-294.4489