SPARSE SPECTRAL-LINE ESTIMATION FOR NONUNIFORMLY SAMPLED MULTIVARIATE TIME SERIES : SPICE, LIKES AND MSBL

Prabhu Babu, Petre Stoica

Dept. of Information Technology, Uppsala University, Uppsala, SE 75105, Sweden.

ABSTRACT

In this paper we deal with the problem of spectral-line analysis of nonuniformly sampled *multivariate* time series for which we introduce two methods: the first method named SPICE (**sp**arse **i**terative **c**ovariance based **e**stimation) is based on a covariance fitting framework whereas the second method named LIKES (**lik**elihood-based **e**stimation of **s**parse parameters) is a maximum likelihood technique. Both methods yield sparse spectral estimates and they do not require the choice of any hyperparameters. We numerically compare the performance of SPICE and LIKES with that of the recently introduced method of **m**ultivariate **s**parse **B**ayesian **l**earning (MSBL).

Index Terms: Spectral analysis, multivariate data, nonuniform sampling, covariance fitting, maximum likelihood, majorization-minimization, expectation maximization.

1. DATA MODEL AND PROBLEM FORMULATION

Let $\{\boldsymbol{y}(t) \triangleq [y_1(t), \cdots, y_M(t)]^T \in \mathbb{C}^{M \times 1}\}$ denote an M-variate time series with each of its components satisfying the following spectral-line model:

$$y_m(t_n) = \sum_{l=1}^{C_m} r_{l,m} e^{i\Omega_{l,m} t_n} + e_m(t_n) \qquad \begin{array}{l} n = 1, \ldots, N \\ m = 1, \ldots, M \end{array} \quad (1)$$

where $\{t_n\}_{n=1}^N$ denote the sampling times which can be nonuniformly placed. For any m, $\{r_{l,m} \in \mathbb{C}\}_{l=1}^{C_m}$ denote the amplitudes of the C_m components located at the frequencies $\{\Omega_{l,m} \in [0, \Omega_{max}], \Omega_{max} \in \mathbb{R}\}_{l=1}^{C_m}$, respectively, and $\{e_m(t_n)\}$ denotes the noise in the data. Given $\{y_m(t_n)\}_{m=1,n=1}^{M,N}$ we want to estimate, for $m = 1, \ldots, M$, the number of components C_m and their corresponding amplitudes $\{r_{l,m}\}_{l=1}^{C_m}$ and frequencies $\{\Omega_{l,m}\}_{l=1}^{C_m}$. Thus the problem is to *detect* the components in the model as well as to *estimate* their amplitudes and frequencies, see e.g. [1] [2] and [3] for motivation of this problem and some possible applications.

To tackle this problem, we divide the frequency interval $[0, \Omega_{max}]$ into a set of uniformly spaced values with a spacing of Δ and form a fine grid $\{\omega_l\}_{l=1}^K$ such that the true frequencies lie on (or, practically, close to) the grid i.e., $\{\Omega_{l,m}\}_{l=1,m=1}^{C_m,M} \subset \{\omega_l\}_{l=1}^K$. A typical choice of Δ is $\frac{2\pi}{10(t_N-t_1)}$ [1] [4]. Making use of the grid, the data model in (1) can be re-written as follows :

$$\boldsymbol{y}_m \triangleq \begin{bmatrix} y_m(t_1) \\ \vdots \\ y_m(t_N) \end{bmatrix} = \begin{bmatrix} e^{i\omega_1 t_1} & \cdots & e^{i\omega_K t_1} \\ \vdots & \cdots & \vdots \\ e^{i\omega_1 t_N} & \cdots & e^{i\omega_K t_N} \end{bmatrix} \begin{bmatrix} x_{1,m} \\ \vdots \\ x_{K,m} \end{bmatrix}$$
$$+ \begin{bmatrix} e_m(t_1) \\ \vdots \\ e_m(t_N) \end{bmatrix} \quad m = 1, \ldots, M \quad (2)$$

for some $\{x_{l,m}\}$ (see below). Let

$$\boldsymbol{Y} \triangleq \begin{bmatrix} \boldsymbol{y}_1 & \cdots & \boldsymbol{y}_M \end{bmatrix}, \quad \boldsymbol{X} \triangleq \begin{bmatrix} x_{1,1} & \cdots & x_{1,M} \\ \vdots & \cdots & \vdots \\ x_{K,1} & \cdots & x_{K,M} \end{bmatrix}$$

$$\boldsymbol{A} \triangleq \begin{bmatrix} \boldsymbol{a}_1 & \cdots & \boldsymbol{a}_K \end{bmatrix} = \begin{bmatrix} e^{i\omega_1 t_1} & \cdots & e^{i\omega_K t_1} \\ \vdots & \cdots & \vdots \\ e^{i\omega_1 t_N} & \cdots & e^{i\omega_K t_N} \end{bmatrix} \quad (3)$$

$$\boldsymbol{E} \triangleq \begin{bmatrix} e_1(t_1) & \cdots & e_M(t_1) \\ \vdots & \cdots & \vdots \\ e_1(t_N) & \cdots & e_M(t_N) \end{bmatrix}.$$

We will refer to $\{\boldsymbol{y}_m\}$ as the data snapshots. By using the above matrix notation, (2) can be re-written as follows :

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{E} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{E} \end{bmatrix} \triangleq \boldsymbol{B}\boldsymbol{Z} \quad (4)$$

where $\boldsymbol{I}$ denotes the identity matrix of dimension $N \times N$, and $\boldsymbol{B} \triangleq [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{K+N}] = [\boldsymbol{A} \ \boldsymbol{I}]$. Usually, the number of components $\{C_m\}$ is much smaller than the grid dimension K, so that only a few values of $\{x_{l,m}\}$ are non-zero i.e., for any m there exist a set $\{m_{\bar{l}}\}$ such that $\{x_{m_{\bar{l}},m} = r_{l,m}\}$ or, in other words, the matrix $\boldsymbol{X}$ is row-sparse. Thus, we have transformed the nonlinear detection and estimation problem associated with (1) into a linear sparse parameter estimation problem for (4).

By assuming that the noise sequences in the data snapshots have zero means but possibly different variances, namely $\{\sigma_k\}_{k=1}^N$, and that they are uncorrelated with each other as well as with $\boldsymbol{X}$, the (normalized) covariance matrix of $\boldsymbol{Y}$ can be expressed as :

$$\boldsymbol{R} \triangleq E\left[\boldsymbol{Y}\boldsymbol{Y}^*\right]/M = \sum_{l=1}^K \sum_{m=1}^M \frac{E[|x_{l,m}|^2]}{M} \boldsymbol{a}_l \boldsymbol{a}_l^*$$
$$+ \text{diag}(\sigma_1, \cdots, \sigma_N) \quad (5)$$
$$\triangleq \boldsymbol{B}\boldsymbol{P}\boldsymbol{B}^*$$

where

$$\boldsymbol{P} = \text{diag}\left(\sum_{m=1}^M \frac{E[|x_{1,m}|^2]}{M}, \cdots, \sum_{m=1}^M \frac{E[|x_{K,m}|^2]}{M}, \sigma_1, \cdots, \sigma_N\right)$$
$$\triangleq \text{diag}(p_1, p_2, \cdots, p_{K+N}) \quad (6)$$

where $E(\cdot)$ denotes the expectation operation. The data sample covariance matrix can be obtained as :

$$\hat{\boldsymbol{R}} = \frac{1}{M} \sum_{m=1}^M \boldsymbol{y}_m \boldsymbol{y}_m^* = \boldsymbol{Y}\boldsymbol{Y}^*/M. \quad (7)$$

The quantities $\{p_l\}_{l=1}^K$ in $\boldsymbol{R}$ represent the powers at the frequencies $\{\omega_l\}_{l=1}^K$, respectively. Although our primary interest is in the

estimation of $\boldsymbol{X}$ from $\boldsymbol{Y}$, we will also estimate $\{p_l\}$. In fact, in the methods described in the following section we will start with the problem of estimating $\{p_l\}$ and an estimate of $\boldsymbol{X}$ will be obtained as a byproduct of the method. We would also like to point out that apart from estimating the powers at different frequencies, we also estimate the noise variances by estimating the quantities $\{p_l\}_{l=K+1}^{K+N}$.

In Section 2, we introduce two new methods, namely SPICE and LIKES, for estimating $\{p_l\}$ and $\boldsymbol{X}$, briefly describe the multivariate-SBL (MSBL) algorithm and duly refer to the previous relevant literature. In Section 3, we compare the statistical performance of SPICE, LIKES and MSBL, as well as their computational complexities and convergence properties, by means of numerical simulations.

2. SPICE, LIKES AND MSBL

2.1. SPICE

Given $\hat{\boldsymbol{R}}$, the $\{p_k\}$ in $\boldsymbol{R}$ can be estimated as the solutions to the following minimization problem :

$$\min_{\boldsymbol{p}} \left\| \boldsymbol{R}^{-1/2}(\boldsymbol{R} - \hat{\boldsymbol{R}}) \right\|^2 \qquad (8)$$

where $\| \cdot \|$ denotes the Frobenius matrix norm, $\boldsymbol{R}^{-1/2}$ denotes a Hermittian square root of $\boldsymbol{R}^{-1}$, and $\boldsymbol{p} \triangleq [p_1, \cdots, p_{K+N}]^T$ with each $p_k \geq 0$. We have used this type of covariance fitting criterion in [4] (for the spectral analysis of univariate time series, i.e. $M = 1$) and, in a related form, in [5] (for spatial spectral analysis, which is essentially equivalent to multivariate time series analysis with $M \geq N$) to derive a sparse parameter estimation technique named SPICE (**sp**arse **i**terative **c**ovariance based **e**stimation). Here we extend SPICE to the multivariate case with $M \in (1, N)$. By substituting the expression for $\boldsymbol{R}$ in (8) and expanding the cost function we get the following equivalent formulation of the problem:

$$\min_{\boldsymbol{p}} \operatorname{tr}(\hat{\boldsymbol{R}}^* \boldsymbol{R}^{-1} \hat{\boldsymbol{R}}) + \sum_{k=1}^{K+N} w_k^2 p_k \qquad (9)$$

where $w_k = \|\boldsymbol{b}_k\|$ and $\operatorname{tr}(\cdot)$ denotes the matrix trace. The minimization problem in (9) is convex and has a unique global minimum. In fact, (9) can be cast as the following semi-definite program (SDP),

$$\begin{aligned} &\min_{\boldsymbol{p},\{\alpha_l\}} \sum_{l=1}^{N} \alpha_l + \sum_{k=1}^{K+N} w_k^2 p_k \\ &\text{s.t.} \begin{bmatrix} \alpha_l & \boldsymbol{g}_l^* \\ \boldsymbol{g}_l & \boldsymbol{R} \end{bmatrix} \geq 0 \quad l = 1, \ldots, N \end{aligned} \qquad (10)$$

where $\{\alpha_l\}$ are auxiliary variables and $[\boldsymbol{g}_1, \cdots, \boldsymbol{g}_N] = \hat{\boldsymbol{R}}$. However, solving the SDP in (10) can be quite time consuming: for instance this SDP cannot be solved on a general purpose PC even for relatively modest dimensions (say $N = 100$, $M = 10$ and $K = 1000$). To tackle this computational issue, we follow [4] to derive an iterative algorithm for the problem in (9).

Consider the following augmented problem:

$$\begin{aligned} &\min_{\boldsymbol{p},\boldsymbol{Q}} \operatorname{tr}(\boldsymbol{Q}^* \boldsymbol{P}^{-1} \boldsymbol{Q}) + \sum_{k=1}^{K+N} w_k^2 p_k \\ &\text{s.t.} \ \boldsymbol{BQ} = \hat{\boldsymbol{R}}. \end{aligned} \qquad (11)$$

Minimization over $\boldsymbol{Q}$ (for fixed $\boldsymbol{p}$) is straightforward: the solution is given by $\boldsymbol{Q}_0 = \boldsymbol{P}\boldsymbol{B}^* \boldsymbol{R}^{-1} \hat{\boldsymbol{R}}$ (see, e.g., [1], Appendix A, Result R35 and [5], Section III, Page 632). It is easy to verify that substituting $\boldsymbol{Q}_0$ back into the cost function in (11) yields the original problem in (9). Hence the $\boldsymbol{p}$'s obtained from (9) and (11) must be identical.

The minimization over $\boldsymbol{p}$ (for a given $\boldsymbol{Q}$) can also be done analytically as follows. Using $\boldsymbol{Q} = [\boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_{K+N}]^*$, the optimization problem in (11) (for fixed $\{\boldsymbol{\beta}_k\}$) can be reduced to

$$\min_{\boldsymbol{p}} \sum_{k=1}^{K+N} \frac{\|\boldsymbol{\beta}_k\|^2}{p_k} + \sum_{k=1}^{K+N} w_k^2 p_k. \qquad (12)$$

A simple calculation shows that

$$\begin{aligned} &\sum_{k=1}^{K+N} \left(\frac{\|\boldsymbol{\beta}_k\|}{\sqrt{p_k}} - w_k\sqrt{p_k} \right)^2 \geq 0 \Longleftrightarrow \\ &\sum_{k=1}^{K+N} \left(\frac{\|\boldsymbol{\beta}_k\|^2}{p_k} + w_k^2 p_k - 2w_k\|\boldsymbol{\beta}_k\| \right) \geq 0 \Longleftrightarrow \\ &\sum_{k=1}^{K+N} \left(\frac{\|\boldsymbol{\beta}_k\|^2}{p_k} + w_k^2 p_k \right) \geq \sum_{k=1}^{K+N} 2w_k\|\boldsymbol{\beta}_k\|. \end{aligned} \qquad (13)$$

The left hand side in the above inequality is nothing but the cost function in (12) and the equality holds only when $p_k = \|\boldsymbol{\beta}_k\|/w_k$. Thus the minimizer of (12) is

$$p_k = \frac{\|\boldsymbol{\beta}_k\|}{w_k} \quad k = 1, \ldots, K + N. \qquad (14)$$

Since the cost function in (11) is convex in both $\boldsymbol{Q}$ and $\boldsymbol{p}$, the cyclic minimization over $\boldsymbol{Q}$ and $\boldsymbol{p}$, i.e. minimization over $\boldsymbol{Q}$ (for fixed $\boldsymbol{p}$) and vice-versa, starting from any arbitrary initial point will lead to the global minimum of (11). The $(i + 1)$-th iteration of the so-obtained cyclic algorithm consists of the following steps:

$$\begin{aligned} \boldsymbol{Q}^{i+1} &= \boldsymbol{P}^i \boldsymbol{B}^* \boldsymbol{R}^{-1}(i)\hat{\boldsymbol{R}} \\ p_k^{i+1} &= \frac{\|\boldsymbol{\beta}_k^{i+1}\|}{w_k} \quad k = 1, \ldots, K + N \\ \boldsymbol{R}(i+1) &= \boldsymbol{B}\boldsymbol{P}^{i+1}\boldsymbol{B}^*. \end{aligned} \qquad (15)$$

(for initialization of (15)), as well as of the other yet-to-be derived algorithms, see Section 3) An estimate of $\boldsymbol{Z}$, and hence of $\boldsymbol{X}$ can then be obtained as follows :

$$\begin{aligned} \hat{\boldsymbol{Z}} &= \boldsymbol{Q}_c \boldsymbol{Y} \left(\boldsymbol{Y}^* \boldsymbol{Y}/M \right)^{-1} \\ \hat{\boldsymbol{X}} &= \text{the first } K \text{ rows of } \hat{\boldsymbol{Z}} \end{aligned} \qquad (16)$$

where $\boldsymbol{Q}_c$ denotes the value of $\boldsymbol{Q}$ obtained at the convergence of (15).

2.2. LIKES

LIKES, which stands for **lik**elihood-based **e**stimation of **s**parse parameters, estimates $\boldsymbol{p}$ by minimizing the Gaussian negative log-likelihood (NLL) function :

$$\begin{aligned} f(\boldsymbol{p}) &= \operatorname{tr}(\boldsymbol{R}^{-1}\hat{\boldsymbol{R}}) + \ln|\boldsymbol{R}| \\ &= \frac{1}{M}\operatorname{tr}(\boldsymbol{Y}^* \boldsymbol{R}^{-1}\boldsymbol{Y}) + \ln|\boldsymbol{R}| \end{aligned} \qquad (17)$$

The minimization problem in (17) is non-convex. In fact, it can be shown that the two terms in $f(\boldsymbol{p})$ are convex and concave in $\boldsymbol{p}$, respectively. In [6] we have derived a LIKES iterative algorithm based on a majorization-minimization technique that minimizes the above cost function in the univariate case. Here, following a similar approach to that in [6], we extend the LIKES algorithm to the multivariate case.

Since the second term in $f(\boldsymbol{p})$, viz. $\ln|\boldsymbol{R}|$, is a concave function in $\boldsymbol{p}$, it can be majorized by its tangent plane at any point. Let $\tilde{\boldsymbol{p}}$ be an arbitrary point in the parameter space, and let $\tilde{\boldsymbol{R}}$ denote the corresponding covariance matrix; then :

$$\begin{aligned} \ln|\boldsymbol{R}| &\leq \ln|\tilde{\boldsymbol{R}}| + \sum_{k=1}^{K+N} \mathrm{tr}\left(\tilde{\boldsymbol{R}}^{-1}\boldsymbol{b}_k\boldsymbol{b}_k^*\right)(p_k - \tilde{p}_k) \\ &= \ln|\tilde{\boldsymbol{R}}| - N + \sum_{k=1}^{K+N} \tilde{w}_k^2 p_k \end{aligned} \tag{18}$$

where

$$\tilde{w}_k^2 = \boldsymbol{b}_k^* \tilde{\boldsymbol{R}}^{-1} \boldsymbol{b}_k. \tag{19}$$

It follows from (18) that :

$$f(\boldsymbol{p}) \leq \left(\ln|\tilde{\boldsymbol{R}}| - N\right) + \tfrac{1}{M}\mathrm{tr}(\boldsymbol{Y}^*\boldsymbol{R}^{-1}\boldsymbol{Y}) + \sum_{k=1}^{K+N} \tilde{w}_k^2 p_k \triangleq g(\boldsymbol{p}) \tag{20}$$

for any vectors $\tilde{\boldsymbol{p}}$ and $\boldsymbol{p}$. Note also that

$$f(\tilde{\boldsymbol{p}}) = g(\tilde{\boldsymbol{p}}). \tag{21}$$

The main implication of (20) and (21) is that we can decrease the function $f(\boldsymbol{p})$ from $f(\tilde{\boldsymbol{p}})$ to, let us say, $f(\hat{\boldsymbol{p}})$ by choosing $\hat{\boldsymbol{p}}$ as a minimizer of the majorizing function $g(\boldsymbol{p})$ or at least such that $g(\tilde{\boldsymbol{p}}) > g(\hat{\boldsymbol{p}})$:

$$f(\hat{\boldsymbol{p}}) \leq g(\hat{\boldsymbol{p}}) < g(\tilde{\boldsymbol{p}}) = f(\tilde{\boldsymbol{p}}) \tag{22}$$

This is precisely the basic idea behind the majorization-minimization approach to solve the problem in (17), see, e.g. [7] and the references therein. The usefulness of this approach depends on how easier the minimization (or the decrease) of $g(\boldsymbol{p})$ is compared to minimizing $f(\boldsymbol{p})$ directly. In the present case, minimizing $g(\boldsymbol{p})$ is much easier than minimizing $f(\boldsymbol{p})$ because $g(\boldsymbol{p})$ is (to within an additive constant) a SPICE-like convex criterion function (compare it with (9) after replacing $\hat{\boldsymbol{R}}$ by $\boldsymbol{Y}/\sqrt{M}$). Consequently, the approach used in the previous subsection can be adopted verbatim to find a vector $\hat{\boldsymbol{p}}$ with the above property, for any given $\tilde{\boldsymbol{p}}$.

Following the said approach, we consider the augmented optimization problem :

$$\min_{\boldsymbol{p},\tilde{\boldsymbol{Q}}} \tfrac{1}{M}\mathrm{tr}(\tilde{\boldsymbol{Q}}^*\boldsymbol{P}^{-1}\tilde{\boldsymbol{Q}}) + \sum_{k=1}^{K+N} \tilde{w}_k^2 p_k \tag{23}$$
$$\text{s.t. } \boldsymbol{B}\tilde{\boldsymbol{Q}} = \boldsymbol{Y}$$

The minimizer $\tilde{\boldsymbol{Q}}$ (for fixed $\boldsymbol{p}$) is given by $\tilde{\boldsymbol{Q}}_0 = \boldsymbol{P}\boldsymbol{B}^*\boldsymbol{R}^{-1}\boldsymbol{Y}$ and substituting $\tilde{\boldsymbol{Q}}_0$ into (23) yields the cost function in (20) (to within an additive constant). By using $\tilde{\boldsymbol{Q}} = \left[\tilde{\boldsymbol{\beta}}_1, \cdots, \tilde{\boldsymbol{\beta}}_{K+N}\right]^*$ and a calculation similar to (12)-(13), the minimizer $\boldsymbol{p}$ (for fixed $\tilde{\boldsymbol{Q}}$) can also be derived. The resulting LIKES algorithm comprises an inner loop to minimize (or to decrease) $g(\boldsymbol{p})$ and an outer loop

to recompute the weights $\{\tilde{w}_k\}$, and can be summarized as follows:

The inner loop:
At iteration $i + 1$:
$$\tilde{\boldsymbol{Q}}^{i+1} = \boldsymbol{P}^i\boldsymbol{B}^*\boldsymbol{R}^{-1}(i)\boldsymbol{Y}$$
$$p_k^{i+1} = \frac{\|\tilde{\boldsymbol{\beta}}_k^{i+1}\|}{\sqrt{M}\tilde{w}_k} \quad k = 1, \ldots, K + N$$
$$\boldsymbol{R}(i + 1) = \boldsymbol{B}\boldsymbol{P}^{i+1}\boldsymbol{B}^*$$
The outer loop:
Letting $\boldsymbol{R}_c(\{p_k^c\})$ denote the $\boldsymbol{R}(\{p_k\})$ obtained at convergence (or after a pre-specified number of iterations) of the inner loop, compute
$$\tilde{w}_k = \sqrt{\boldsymbol{b}_k^*\boldsymbol{R}_c^{-1}\boldsymbol{b}_k}$$
and then go to the inner loop. The inner loop will then be initialized with $\{p_k^c\}$.
Final step :
The estimates of $\boldsymbol{Z}$ and $\boldsymbol{X}$ are obtained as
$$\hat{\boldsymbol{Z}} = \tilde{\boldsymbol{Q}}_c$$
$$\hat{\boldsymbol{X}} = \text{the first } K \text{ rows of } \hat{\boldsymbol{Z}}$$
where $\tilde{\boldsymbol{Q}}_c$ denotes the value of $\tilde{\boldsymbol{Q}}$ obtained at convergence of the outer loop.

$$\tag{24}$$

2.3. MSBL

Besides the majorization-minimization technique, the NLL function in (17) can also be minimized by an expectation-maximization (EM) algorithm. Such an approach named **s**parse **B**ayesian **l**earning (SBL) has been suggested in [8] for the univariate case; and in [9] for the multivariate case where it was called MSBL. For conciseness we present only the main steps of the MSBL algorithm:

Iterative step :
$$\tilde{\boldsymbol{Q}}^{i+1} = \boldsymbol{P}^i\boldsymbol{B}^*\boldsymbol{R}^{-1}(i)\boldsymbol{Y}$$
$$p_k^{i+1} = p_k^i - \left(p_k^i\right)^2 \boldsymbol{b}_k^*\boldsymbol{R}^{-1}(i)\boldsymbol{b}_k + \|\tilde{\boldsymbol{\beta}}_k^{i+1}\|^2/M$$
$$k = 1, \ldots, K + N$$
$$\boldsymbol{R}(i + 1) = \boldsymbol{B}\boldsymbol{P}^{i+1}\boldsymbol{B}^*.$$
Final step :
The estimates of $\boldsymbol{Z}$ and $\boldsymbol{X}$ are obtained as:
$$\hat{\boldsymbol{Z}} = \tilde{\boldsymbol{Q}}_c$$
$$\hat{\boldsymbol{X}} = \text{the first } K \text{ rows of } \hat{\boldsymbol{Z}}$$
where $\tilde{\boldsymbol{Q}}_c$ denotes the value of $\tilde{\boldsymbol{Q}}$ obtained at the convergence.
$$\tag{25}$$

3. NUMERICAL SIMULATIONS AND CONCLUDING REMARKS

In this section we numerically compare the performance of SPICE, LIKES and MSBL. The data were generated via the model in (2) with $N = 100$, $M = 10$ and $K = 1000$. The sampling times $\{t_n\}$ were uniformly randomly distributed between $[0 - 20]$ sec. The value of Ω_{max} was chosen to be 10π rad/sec. In each of the 10 data snapshots, 5 sinusoidal components were present. Each snapshot shares three common frequencies with its neighboring snapshots. Table 1 shows the values of the frequencies with non-zero amplitudes in different snapshots. The amplitudes of all existing sinusoidal components were chosen as 5. The noise was Gaussian distributed with zero mean and variance equal to σ. The signal to

noise ratio (SNR) is defined by $10\log(25/\sigma)$. The three methods were initialized with the periodogram estimate $\{p_k^0 = \boldsymbol{b}_k^* \hat{\boldsymbol{R}} \boldsymbol{b}_k / N\}$. For all methods the convergence criterion used to terminate the iterations was : $\frac{\|\boldsymbol{p}^{i+1} - \boldsymbol{p}^i\|}{\|\boldsymbol{p}^i\|} < 10^{-3}$. In the case of LIKES, this convergence criterion has been used in the outer loop while the inner loop has been run for 10 iterations.

3.1. Statistical performance

Figure 1 shows 100 superimposed plots of amplitude spectra corresponding to the (randomly picked) 7-th data snapshot obtained with SPICE, LIKES and MSBL. The spectra obtained with all methods are sparse and they correctly indicate the presence of sinusoidal components in most of the Monte-Carlo runs; furthermore the likelihood based approaches provide more accurate amplitude estimates than SPICE. Figure 2 shows the plots of average mean square error (AMSE) of the amplitude estimates at the true frequency locations, as well as the probability of detection of the true frequencies vs SNR. The AMSE of the amplitude estimates was calculated as:

$$\text{AMSE} = \frac{1}{5000} \sum_{k=1}^{100} \sum_{m=1}^{10} \sum_{\bar{l}=1}^{5} \left| \hat{x}_{m_{\bar{l}},m}^k - 5 \right|^2 \qquad (26)$$

where $\{m_{\bar{l}}\}_{\bar{l}=1}^5$ denote the frequency indices corresponding to the 5 sinusoidal components present in the m-th snapshot, and the superscript k in $\hat{x}_{m_{\bar{l}},m}^k$ denotes the Monte-Carlo run. The probability of detection was computed by first picking the 5 dominant peaks of the estimated spectrum, sorting the frequencies corresponding to those peaks, and then calculating the mean absolute error of those frequency estimates: if the mean absolute error is equal to zero then we have a detection, else we declare a miss. It can be observed from the plots in Figure 2 that SPICE is less accurate than LIKES and MSBL for both amplitude and frequency estimation, and that LIKES is slightly better than MSBL in the case of amplitude estimation.

3.2. Complexity and convergence rate

The computational complexity per iteration (in flops) of the considered methods is on the order of $O(2N^3 + 2N^2M + 2KMN + KN^2 + KM + NM)$ with MSBL requiring an additional $KN^2 + N^3$ flops to compute the terms in the power update formula, see (25). However, the convergence rates of the three methods differ quite a bit from one another, which leads to rather different execution times. Figure 3a shows the average computation times (in sec) per run vs SNR; it is seen from this figure that the times decrease with increasing SNR, which is primarily due to the fact that the methods converge faster as the SNR increases. For a fixed SNR, SPICE is faster than LIKES which is faster than MSBL.

As both LIKES and MSBL minimize the same NLL criterion, it is interesting to compare their convergence rates and the values of NLL they attain at convergence. Figure 3b shows the NLL value vs the iteration number for LIKES and MSBL in one Monte-Carlo run; it can be seen from this plot that LIKES converges faster than MSBL and also that the value of NLL that it attains at convergence is slightly lower than the NLL value attained by MSBL.

3.3. Concluding remarks

To conclude, based on our recent work we have introduced two methods to solve the problem of spectral-line analysis for nonuniformly sampled multivariate time series. Both methods yield sparse

	Snapshot number									
	1	2	3	4	5	6	7	8	9	10
Frequencies ×π (rad/sec)	0.05	0.79	0.85	1.24	1.45	1.53	1.66	1.84	2.40	2.63
	0.76	0.84	1.07	1.37	1.46	1.63	1.82	2.29	2.60	2.64
	0.79	0.85	1.24	1.45	1.53	1.66	1.84	2.40	2.63	3.12
	0.84	1.07	1.37	1.46	1.63	1.82	2.29	2.60	2.64	3.51
	0.85	1.24	1.45	1.53	1.66	1.84	2.40	2.63	3.12	4.00

Table 1. The frequencies of the sinusoidal components in the different data snapshots.

parameter estimates and they do not require any selection of hyperparameters. We have also considered the previously proposed method of MSBL that minimizes the same NLL function as LIKES. We compared these three methods via numerical simulations and observed that LIKES and MSBL are more accurate than SPICE but at the cost of extra computation time. Regarding LIKES and MSBL, we showed that LIKES converges faster than MSBL and also that the NLL value that LIKES attains at convergence can be slightly smaller than the NLL value attained by MSBL.

Acknowledgments

4. REFERENCES

[1] P. Stoica and R. Moses, *Spectral Analysis of Signals*. Prentice Hall, Upper Saddle River, NJ, 2005.

[2] N. Butt and A. Jakobsson, "Coherence spectrum estimation from nonuniformly sampled sequences," *IEEE Signal Processing Letters*, vol. 17, no. 4, pp. 339–342, 2010.

[3] A. Jakobsson, S. Alty, and J. Benesty, "Estimating and time-updating the 2-D coherence spectrum," *IEEE Transactions on Signal Processing*, vol. 55, no. 5, pp. 2350–2354, 2007.

[4] P. Stoica, P. Babu, and J. Li, "New method of sparse parameter estimation in separable models and its use for spectral analysis of irregularly sampled data," *IEEE Transactions on Signal Processing*, vol. 59, no. 1, pp. 35–47, 2011.

[5] ——, "SPICE: A sparse covariance based estimation method for array processing," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 629–638, 2011.

[6] P. Stoica and P. Babu, "SPICE and LIKES : Two hyperparameter-free methods for sparse-parameter estimation," *Signal Processing*, vol. 92, no. 7, pp. 1580–1590, , 2012.

[7] P. Stoica and Y. Selen, "Cyclic minimizers, majorization techniques, and the expectation-maximization algorithm: a refresher," *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 112–114, 2004.

[8] D. Wipf and B. Rao, "Sparse Bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.

[9] ——, "An empirical Bayesian strategy for solving the simultaneous sparse approximation problem," *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3704–3716, 2007.
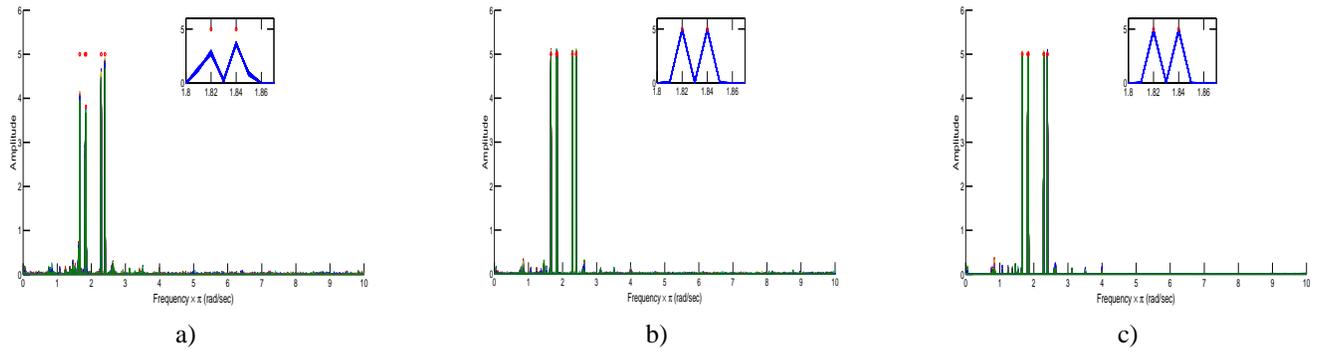
Fig. 1. One hundred superimposed plots of the amplitude spectra corresponding to the 7-th data snapshot estimated via a) SPICE, b) LIKES and c) MSBL. The circles indicate the locations and amplitudes of the true components in the data. The SNR was 40 dB. The peaks at the closely-spaced frequencies of 1.82π and 1.84π appear almost merged but in actuality they are distinct. The zoom-in plots show the spectrum in the interval $[1.8 - 1.86] \times \pi$ rad/sec to confirm this fact.
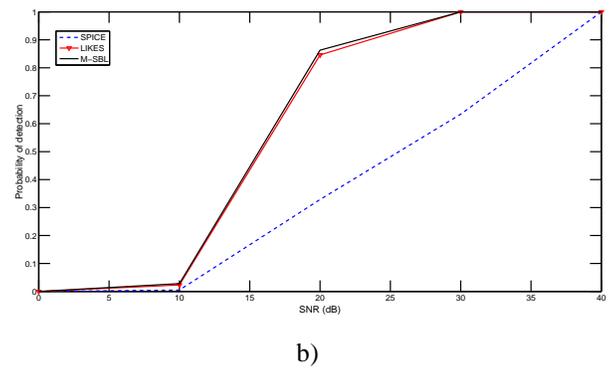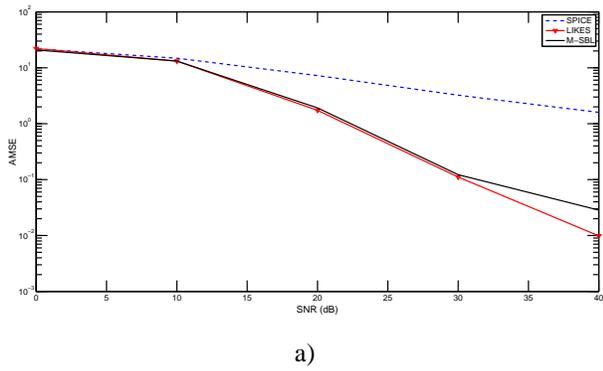
Fig. 2. a) AMSE of amplitude estimates at the true frequency locations vs SNR b) Probability of detection of true frequencies vs SNR. The number of Monte-Carlo runs was 100.
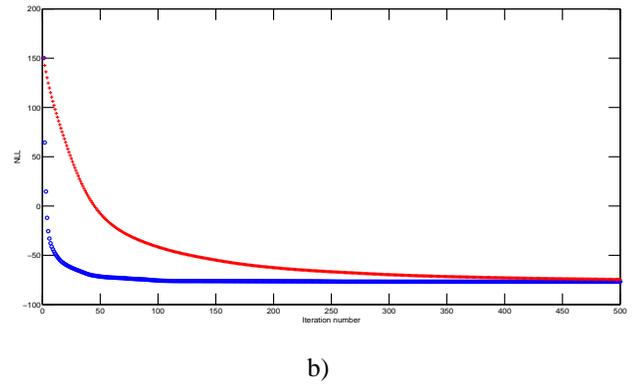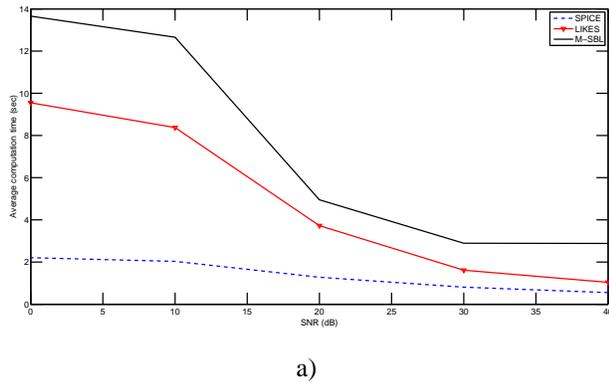
Fig. 3. a) Average computation times (sec) of SPICE, LIKES and MSBL vs SNR, b) NLL values of LIKES and MSBL vs iteration number (**o** : LIKES, + : MSBL).