Communications in Applied Mathematics and Computational Science

AN EQUATION-BY-EQUATION METHOD FOR SOLVING THE MULTIDIMENSIONAL MOMENT CONSTRAINED MAXIMUM ENTROPY PROBLEM

WENRUI HAO AND JOHN HARLIM

vol. 13 no. 2 2018

msp

COMM. APP. MATH. AND COMP. SCI.
Vol. 13, No. 2, 2018
dx.doi.org/10.2140/camcos.2018.13.189

AN EQUATION-BY-EQUATION METHOD FOR SOLVING THE MULTIDIMENSIONAL MOMENT CONSTRAINED MAXIMUM ENTROPY PROBLEM

WENRUI HAO AND JOHN HARLIM

An equation-by-equation (EBE) method is proposed to solve a system of nonlinear equations arising from the moment constrained maximum entropy problem of multidimensional variables. The design of the EBE method combines ideas from homotopy continuation and Newton's iterative methods. Theoretically, we establish the local convergence under appropriate conditions and show that the proposed method, geometrically, finds the solution by searching along the surface corresponding to one component of the nonlinear problem. We will demonstrate the robustness of the method on various numerical examples, including (1) a six-moment one-dimensional entropy problem with an explicit solution that contains components of order 10^0–10^3 in magnitude, (2) four-moment multidimensional entropy problems with explicit solutions where the resulting systems to be solved range from 70–310 equations, and (3) four- to eight-moment of a two-dimensional entropy problem, whose solutions correspond to the densities of the two leading EOFs of the wind stress-driven large-scale oceanic model. In this case, we find that the EBE method is more accurate compared to the classical Newton's method, the MATLAB generic solver, and the previously developed BFGS-based method, which was also tested on this problem. The fourth example is four-moment constrained of up to five-dimensional entropy problems whose solutions correspond to multidimensional densities of the components of the solutions of the Kuramoto–Sivashinsky equation. For the higher-dimensional cases of this example, the EBE method is superior because it automatically selects a subset of the prescribed moment constraints from which the maximum entropy solution can be estimated within the desired tolerance. This selection feature is particularly important since the moment constrained maximum entropy problems do not necessarily have solutions in general.

Hao's research was partially supported by the American Heart Association (Grant 17SDG33660722), the National Science Foundation (Grant DMS-1818769) and an Institute for CyberScience Seed Grant. Harlim's research was partially supported by the Office of Naval Research (Grant N00014-16-1-2888) and the National Science Foundation (Grant DMS-1619661).

MSC2010: 65H10, 65H20, 94A17, 49M15.

Keywords: homotopy continuation, moment constrained, maximum entropy, equation-by-equation method.

1. Introduction

The maximum entropy principle provides a natural criterion for estimating the least biased density function subjected to the given moments [14]. This density estimation approach has a wide range of applications, such as the harmonic solid and quantum spin systems [20], econometrics [26], and geophysical applications [5; 13]. In a nutshell, this moment constrained method is a parametric estimation technique where the resulting density function is in the form of an exponential of polynomials. This is a consequence of maximizing the Shannon entropy subjected to the polynomial moment constraints, which is usually transformed into an unconstrained minimization problem of a Lagrangian function [27]. Standard approaches for solving this unconstrained minimization problem are based on Newton's iterative method [1; 27] or a quasi-Newton-based method such as the BFGS method [3; 4].

In the last two papers [3; 4], where the BFGS-based method was introduced and reviewed, Abramov considered minimization problems that involve 44–83 equations, resulting from a two-dimensional problem with moment constraints of up to order eight, a three-dimensional problem with moment constraints of up to order six, and a four-dimensional problem with moment constraints of up to order four. In this paper, we introduce a novel equation solver that can be used to find density functions of moderately high-dimensional problems (e.g., systems of 70–310 equations resulting from moments up to order four of four- to seven-dimensional density functions) provided that the solutions exist. The proposed method, which we called the equation-by-equation (EBE) method, is an iterative method that solves a one-dimensional problem at the first iterate, a two-dimensional problem at the second iterate, a three-dimensional problem at the third iterate, and eventually solves the full system of nonlinear equations corresponding to the maximum entropy problem at the last iterate. Technically, this method combines Newton's method with ideas from homotopy continuation. We will show that the EBE method is locally convergent under appropriate conditions. Furthermore, we will provide sufficient conditions for global convergence. Through the convergence analysis, we will show that, geometrically, the proposed method finds the solution of the nonlinear system of equations by tracking along the surface corresponding to one component of the system of nonlinear equations. The EBE method automatically selects a subset of the prescribed constraints from which the maximum entropy solution can be estimated within the desired tolerance. This is an important feature since the maximum entropy problems do not necessarily have solutions for general sets of moment constraints.

We shall find that the EBE method produces more accurate solutions (smaller error in the moments) compared to the classical Newton's method, MATLAB's built-in fsolve, and BFGS method on the test problem in [3; 4] and on test problems

based on the solutions of the Kuramoto–Sivashinsky equation. Numerically, we will demonstrate that the EBE method is able to solve problems where the true solutions consist of components of order 10^0–10^3. We shall also see that the EBE method can solve a system of hundreds of equations in various examples, including those with explicit solutions as well as those with densities estimated based on solutions of complex spatially extended dynamical systems.

The remaining part of the paper is organized as follows. In Section 2, we give a brief overview of the multidimensional maximum entropy problem. In Section 3, we introduce the EBE algorithm. In Section 4, we provide the local convergence analysis. In Section 5, we discuss the practical issues with the proposed method and provide remedies. In Section 6, we demonstrate the robustness of the EBE method on various numerical examples. In Section 7, we conclude the paper with a brief summary and discussion. We include an Appendix to show some computational details that are left out in the main text. Interested readers and users can access the EBE codes (written in MATLAB) at [10].

2. An overview of the maximum entropy problem

We consider the Haussdorf moment-constrained maximum entropy problem [1; 4; 8]. That is, find the optimal probability density $\rho^*(\boldsymbol{x})$ which maximizes the Shannon entropy

$$S(\rho) := -\int_\Omega \log(\rho(\boldsymbol{x}))\rho(\boldsymbol{x})\, d\boldsymbol{x}, \tag{1}$$

where $\boldsymbol{x} \in \Omega = [-1, 1]^d$ satisfies the linear constraints

$$\mathcal{F}_{\boldsymbol{j}} := \int_\Omega c_{\boldsymbol{j}}(\boldsymbol{x})\rho(\boldsymbol{x})\, d\boldsymbol{x} = f_{\boldsymbol{j}}, \quad |\boldsymbol{j}| = 0, 1, 2, \ldots, p. \tag{2}$$

In applications, one usually computes the statistics $f_{\boldsymbol{j}}$ from samples of data. For arbitrary finite domain, one can rescale the data to the domain Ω.

While $c_{\boldsymbol{j}}(\boldsymbol{x})$ can be arbitrary functions in $L^1(\Omega, \rho)$, we will focus on the usual uncentered statistical moments with monomial basis functions, $c_{\boldsymbol{j}}(\boldsymbol{x}) = \boldsymbol{x}^{\boldsymbol{j}}$ in this article, where we have adopted the notations $\boldsymbol{x} = (x_1, \ldots, x_d) \in \Omega$, $\boldsymbol{j} = (j_1, \ldots, j_d) \in \mathbb{Z}_+^d$ with $\mathbb{Z}_+ = \{0, 1, 2, \ldots\}$, and $\boldsymbol{x}^{\boldsymbol{j}} = \prod_{i=1}^d x_i^{j_i}$. In (2), the quantities $f_{\boldsymbol{j}}$ are the given $\boldsymbol{j}$-th moments that can be computed from the data. Since the total number of monomials $\boldsymbol{x}^{\boldsymbol{j}}$ where $|\boldsymbol{j}| = j$ is C_{d-1}^{j+d-1}, then the total number of constraints in (2) for moments up to order p is

$$n = \sum_{j=1}^{p} C_{d-1}^{j+d-1},$$

excluding the normalization factor corresponding to $c_{\boldsymbol{0}}(\boldsymbol{x}) = 1$. For example, in a two-dimensional problem, the total number of moments up to order $p = 4$ is $n = 14$.

To simplify the notation below, we will use a single index notation and understand that the total number of constraints to be satisfied is n, excluding the zeroth moment. The exclusion of the zeroth moment will be clear as we discuss below.

By introducing Lagrange multipliers, the above constrained optimization problem can be transformed into the unconstrained problem

$$\mathcal{L}(\rho(\boldsymbol{x}), \lambda_0, \ldots, \lambda_n) = S(\rho) + \sum_{j=0}^{n} \lambda_j (\mathcal{F}_j - f_j). \tag{3}$$

In order to find a solution of (3), we set $\frac{\partial \mathcal{L}}{\partial \rho} = 0$, which gives

$$\rho(\boldsymbol{x}) = \frac{1}{Z} \exp\left(\sum_{j=1}^{n} \lambda_j c_j(\boldsymbol{x}) \right), \tag{4}$$

where we have defined $Z = \exp(1 - \lambda_0)$. Since $\int_\Omega \rho(\boldsymbol{x})\, d\boldsymbol{x} = 1$, we have

$$Z(\lambda_1, \ldots, \lambda_n) = \int_\Omega \exp\left(\sum_{j=1}^{n} \lambda_j c_j(\boldsymbol{x}) \right) d\boldsymbol{x}, \tag{5}$$

which indicates that Z (or implicitly λ_0) is a function of $\lambda_1, \ldots, \lambda_n$. Therefore, the normalization factor Z can be computed via (5) once $\lambda_1, \ldots, \lambda_n$ are estimated. Therefore, we can just concentrate on finding the Lagrange multipliers $\lambda_1, \ldots, \lambda_n$ which satisfy n constraints in (2), excluding the case $c_{\mathbf{0}}(\boldsymbol{x}) = 1$. In particular, the constrained maximum entropy problem is to solve the nonlinear system of integral equations

$$\begin{aligned} F_j(\lambda_1, \ldots, \lambda_n) &:= \mathcal{F}_j(\lambda_1, \ldots, \lambda_n) - f_j \\ &= \int_\Omega (c_j(\boldsymbol{x}) - f_j) \exp\left(\sum_{k=1}^{n} \lambda_k c_k(\boldsymbol{x}) \right) d\boldsymbol{x} = 0, \quad j = 1, \ldots, n, \end{aligned} \tag{6}$$

for $\lambda_1, \ldots, \lambda_n$.

In our numerical implementation, the integral in system (6) will be approximated with a nested sparse grid quadrature rule [9]

$$\int_\Omega f(\boldsymbol{x})\, d\boldsymbol{x} \approx \sum_i f(\boldsymbol{x}_i) w_i,$$

where $\boldsymbol{x}_i$ are the nested sparse grid nodes, and w_i are the corresponding weights based on the nested Clenshaw–Curtis quadrature rule [25]. The number of nodes depends on the dimension of the problem d, and the number of the nested set (based on the Smolyak construction [23]) is denoted with the parameter ℓ (referred to as the level). In the numerical implementation, we need to specify the parameter ℓ.

3. An equation-by-equation algorithm

In this section, we describe the new equation-by-equation (EBE) technique to solve the system of equations in (6),

$$\boldsymbol{F}_n(\boldsymbol{\lambda}_n) = \mathbf{0}, \tag{7}$$

where we have defined

$$\boldsymbol{F}_n(\boldsymbol{\lambda}_n) := (F_1(\boldsymbol{\lambda}_n), \ldots, F_n(\boldsymbol{\lambda}_n)),$$

and $\boldsymbol{\lambda}_n = (\lambda_1, \ldots, \lambda_n)$. In the following iterative scheme, we start the iteration with an initial condition $(\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$. We define $\boldsymbol{\mu}^{(i)} \in \mathbb{R}^i$ as the exact solution to the i-dimensional system

$$\boldsymbol{F}_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = \mathbf{0}, \quad i = 1, \ldots, n, \tag{8}$$

where we have fixed the last $n - i$ coefficients, $\lambda_{i+1} = \alpha_{i+1}, \ldots, \lambda_n = \alpha_n$. With this notation, the exact solution for (7) is $\boldsymbol{\mu}^{(n)} \in \mathbb{R}^n$. We also define $\hat{\boldsymbol{\mu}}^{(i)}$ to be the numerical estimate of $\boldsymbol{\mu}^{(i)}$. With these notations, we now describe the algorithm.

Generally speaking, at each iteration i, where $i = 1, \ldots, n$, the EBE algorithm solves i-dimensional system in (8). At each step i, given the numerical solution at the previous step $\hat{\boldsymbol{\mu}}^{(i-1)} \in \mathbb{R}^{i-1}$ and initial condition α_i, we apply an idea from homotopy continuation to find the solution $\boldsymbol{\mu}^{(i)} \in \mathbb{R}^i$ that solves the i-dimensional system of equations (8). Notice that we do not only add a new equation $F_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = 0$ but we also estimate the i-th variable in the previous $i - 1$ equations $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = \mathbf{0}$. The scheme proceeds by solving the larger systems one by one until $i = n$ so we eventually solve (7).

Now let us describe how to numerically estimate $\boldsymbol{\mu}^{(i)}$ at every step i. For the first step $i = 1$, we solve the one-dimensional problem

$$\boldsymbol{F}_1(\boldsymbol{\lambda}_1, \alpha_2, \ldots, \alpha_n) = 0$$

for $\boldsymbol{\lambda}_1$ with Newton's method. For the steps $i = 2, \ldots, n$, we have $\hat{\boldsymbol{\mu}}^{(i-1)}$ which are the numerical estimates of $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \alpha_i, \ldots, \alpha_n) = \mathbf{0}$. To simplify the expression below, let us use $F_i(\boldsymbol{\lambda}_{i-1}, \lambda_i)$ as a short-hand notation for $F_i(\boldsymbol{\lambda}_{i-1}, \lambda_i, \alpha_i, \ldots, \alpha_n)$ to emphasize the independent variables.

We proceed to estimate λ_i using Newton's method with Tol_1 on the i-th equation. That is, we iterate

$$\begin{aligned} \lambda_i^{m+1} &= \lambda_i^m - \left(\frac{\partial F_i}{\partial \lambda_i}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m) \right)^{-1} F_i(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m), \quad m = 0, 1, \ldots, \\ \lambda_i^0 &= \alpha_i, \quad \boldsymbol{\lambda}_{i-1}^0 = \hat{\boldsymbol{\mu}}^{(i-1)} \end{aligned} \tag{9}$$

assuming that $\frac{\partial F_i}{\partial \lambda_i}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m) \neq 0$. Here, the partial derivative of F_i with respect to λ_i evaluated at λ_i^m is defined as

$$\frac{\partial F_i}{\partial \lambda_i}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m) = \int_\Omega (c_i(\boldsymbol{x}) - f_i) c_i(\boldsymbol{x}) \exp\left(\sum_{j=1}^{i-1} \lambda_j^m c_j(\boldsymbol{x}) + \lambda_i^m c_i(\boldsymbol{x}) \right) d\boldsymbol{x}, \quad (10)$$

where we have denoted $\boldsymbol{\lambda}_{i-1}^m = (\lambda_1^m, \ldots, \lambda_{i-1}^m)$. Notice that to proceed the iteration in (9), we need to update $\boldsymbol{\lambda}_{i-1}^m$ for $m > 0$. We propose to follow the homotopy continuation method for this update. In particular, we are looking for $\boldsymbol{\lambda}_{i-1}^{m+1}$ that solves $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}^{m+1}, \lambda_i^{m+1}) = \mathbf{0}$, given the current estimate λ_i^{m+1} from (9) as well as $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m) = \mathbf{0}$. At $m = 0$, this last constraint is numerically estimated by $\boldsymbol{F}_{i-1}(\hat{\boldsymbol{\mu}}^{(i-1)}, \alpha_i) \approx \mathbf{0}$.

One way to solve this problem is through the following predictor-corrector step which is usually used in the homotopy continuation method [7; 24]. In particular, we apply Taylor's expansion to

$$\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}^{m+1}, \lambda_i^{m+1}) = \boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}^m + \Delta\boldsymbol{\lambda}, \lambda_i^m + (\lambda_i^{m+1} - \lambda_i^m)) = \mathbf{0}$$

at $(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)$, which gives

$$\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m) + \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)\Delta\boldsymbol{\lambda} + \boldsymbol{F}_{i-1,\lambda_i}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)(\lambda_i^{m+1} - \lambda_i^m) = \mathbf{0},$$

which means that

$$\Delta\boldsymbol{\lambda} = -\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}^{-1}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)\boldsymbol{F}_{i-1,\lambda_i}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)(\lambda_i^{m+1} - \lambda_i^m),$$

assuming that $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)$ is invertible. Based on this linear prediction, $\boldsymbol{\lambda}_{i-1}^{m+1}$ is approximated by

$$\begin{aligned} \tilde{\boldsymbol{\lambda}}_{i-1}^{m+1} &= \boldsymbol{\lambda}_{i-1}^m + \Delta\boldsymbol{\lambda} \\ &= \boldsymbol{\lambda}_{i-1}^m - \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}^{-1}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)\boldsymbol{F}_{i-1,\lambda_i}(\boldsymbol{\lambda}_{i-1}^m, \lambda_i^m)(\lambda_i^{m+1} - \lambda_i^m). \end{aligned} \quad (11)$$

Subsequently, when $\|\boldsymbol{F}_i(\tilde{\boldsymbol{\lambda}}_{i-1}^{m+1}, \lambda_i^{m+1})\| \geq \mathrm{Tol}_2$, apply a correction using Newton's method by expanding

$$\mathbf{0} = \boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}^{m+1}, \lambda_i^{m+1}) = \boldsymbol{F}_{i-1}(\tilde{\boldsymbol{\lambda}}_{i-1}^{m+1}, \lambda_i^{m+1}) + \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\tilde{\boldsymbol{\lambda}}_{i-1}^{m+1}, \lambda_i^{m+1})\Delta\tilde{\boldsymbol{\lambda}},$$

assuming that $\boldsymbol{\lambda}_{i-1}^{m+1} = \tilde{\boldsymbol{\lambda}}_{i-1}^{m+1} + \Delta\tilde{\boldsymbol{\lambda}}$, to find that

$$\boldsymbol{\lambda}_{i-1}^{m+1} = \tilde{\boldsymbol{\lambda}}_{i-1}^{m+1} - \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\tilde{\boldsymbol{\lambda}}_{i-1}^{m+1}, \lambda_i^{m+1})^{-1}\boldsymbol{F}_{i-1}(\tilde{\boldsymbol{\lambda}}_{i-1}^{m+1}, \lambda_i^{m+1}). \quad (12)$$

This expression assumes that $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\tilde{\boldsymbol{\lambda}}_{i-1}^{m+1}, \lambda_i^{m+1})$ is invertible.

In summary, at each step i, we iterate (9), (11), and (12). So the outer loop i corresponds to adding one equation to the system at the time, and for each i, we apply an inner loop, indexed with m, to find the solution $\boldsymbol{\mu}^{(i)}$ for $\boldsymbol{F}_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = \mathbf{0}$.

We denote the approximate solution as $\hat{\boldsymbol{\mu}}^{(i)}$. An adaptive tolerance technique is employed to compute the initial guess of $\boldsymbol{F}_i$ by using Newton's method. In particular, when the current tolerance Tol_2 is not satisfied after executing (12), then we divide Tol_1 by ten until Tol_2 is met.

Recall that the standard Newton's method assumes that the Jacobian $\boldsymbol{F}_{n,\boldsymbol{\lambda}_n} \in \mathbb{R}^{n\times n}$ is nonsingular at the root of the full system in (6) to guarantee the local convergence. In the next section, we will show that the EBE method requires the following conditions for local convergence.

Assumption 1. Let $\boldsymbol{\mu}^{(i)} \in \mathbb{R}^i$ be a solution of $\boldsymbol{F}_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = \mathbf{0}$, for each $i = 1, \ldots, n$. The EBE method assumes the conditions

(1) $\frac{\partial F_i}{\partial \lambda_i}(\boldsymbol{\mu}^{(i)}, \alpha_{i+1}, \ldots, \alpha_n) \neq 0$,

(2) $\boldsymbol{F}_{i,\boldsymbol{\lambda}_i}(\boldsymbol{\mu}^{(i)}, \alpha_{i+1}, \ldots, \alpha_n)$ are nonsingular, and

(3) each component of $\boldsymbol{F}_i$ is twice differentiable in a close region whose interior contains the solution $\boldsymbol{\mu}^{(i)}$.

These conditions are similar to the standard Newton's assumptions on each system of i equations. The smoothness condition will be used in the proof of the local convergence in the next section. Of course if one can specify initial conditions that are sufficiently close to the true solution, then one can simply apply Newton's method directly. With the EBE method, we can start with any arbitrary initial condition. Theoretically, this will require an additional condition beyond Assumption 1 for global convergence as we shall discuss in Section 4. In Section 5, we will provide several remedies when the initial condition is not close to the solution. In fact, we will always set the initial condition to zero in our numerical implementation in Section 6, $\alpha_i = 0$ for all $i = 1, \ldots, n$, and demonstrate that the EBE method is numerically accurate in the test problems with solutions that are far away from zero.

4. Convergence analysis

In this section, we study the convergence of this method. First, let's concentrate on the convergence of the iteration (9), (11), and (12) for solving the i-dimensional system $\boldsymbol{F}_i(\boldsymbol{\lambda}_{i-1}, \lambda_i, \alpha_{i+1}, \ldots, \alpha_n) := \boldsymbol{F}_i(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \mathbf{0}$ for $\boldsymbol{\lambda}_{i-1}$ and λ_i. In compact form, these three steps can be written as an iterative map

$$(\boldsymbol{\lambda}_{i-1}^{m+1}, \lambda_{i+1}^{m+1}) = \boldsymbol{H}_i(\boldsymbol{\lambda}_{i-1}^{m}, \lambda_i^{m}), \tag{13}$$

where the map $\boldsymbol{H}_i : \mathbb{R}^i \to \mathbb{R}^i$ is defined as

$$\boldsymbol{H}_i(\boldsymbol{\lambda}_{i-1}, \lambda_i) := \begin{pmatrix} \boldsymbol{g}_i - \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{g}_i, H_{i,2})^{-1}\boldsymbol{F}_{i-1}(\boldsymbol{g}_i, H_{i,2}) \\ \lambda_i - \left(\frac{\partial F_i}{\partial \lambda_i}(\boldsymbol{\lambda}_{i-1}, \lambda_i)\right)^{-1} F_i(\boldsymbol{\lambda}_{i-1}, \lambda_i) \end{pmatrix}. \tag{14}$$

In (14), the notation $H_{i,2}$ denotes the second component of (14) and

$$\boldsymbol{g}_i := \boldsymbol{\lambda}_{i-1} - \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}, \lambda_i)^{-1} \boldsymbol{F}_{i-1,\lambda_i}(\boldsymbol{\lambda}_{i-1}, \lambda_i)(H_{i,2} - \lambda_i) \tag{15}$$

is defined exactly as in (11).

For notational convenience in the discussion below, we let the components of the exact solution of (8) be defined as $\boldsymbol{\mu}^{(i)} := (\boldsymbol{\mu}^{(i)}_{i-1}, \mu^{(i)}_i) \in \mathbb{R}^i$. Here, we denote the first $i-1$ components as $\boldsymbol{\mu}^{(i)}_{i-1} = (\mu^{(i)}_1, \ldots, \mu^{(i)}_{i-1}) \in \mathbb{R}^{i-1}$. Similarly, we also denote $\boldsymbol{H}_i = (\boldsymbol{H}_{i,1}, H_{i,2})$. First, we can deduce:

Theorem 4.1. *Let $\boldsymbol{\mu}^{(i)} \in \mathbb{R}^i$ be a fixed point of (13). Assume that $\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}} := \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\mu}^{(i)})$ is nonsingular and $\frac{\partial F^*_i}{\partial \lambda_i} := \frac{\partial F_i}{\partial_{\lambda_i}}(\boldsymbol{\mu}^{(i)}) \neq 0$; then $\boldsymbol{F}^*_i := \boldsymbol{F}_i(\boldsymbol{\mu}^{(i)}) = \boldsymbol{0}$.*

Proof. Evaluating the second equation in (14) at the fixed point, we obtain

$$\mu^{(i)}_i = \mu^{(i)}_i - \left(\frac{\partial F^*_i}{\partial \lambda_i}\right)^{-1} F^*_i,$$

which means that $F^*_i := F_i(\boldsymbol{\mu}^{(i)}) = 0$. This also implies that $H^*_{i,2} = \mu^{(i)}_i$, where $H^*_{i,2}$ denotes the second component of (14) evaluated at the fixed point. Subsequently,

$$\boldsymbol{g}^*_i := \boldsymbol{g}_i(\boldsymbol{\mu}^{(i)}_{i-1}, \mu^{(i)}_i) = \boldsymbol{\mu}^{(i)}_{i-1}.$$

Substituting $H^*_{i,2} = \mu^{(i)}_i$ and $\boldsymbol{g}^*_i = \boldsymbol{\mu}^{(i)}_{i-1}$ into $\boldsymbol{\mu}^{(i)}_{i-1} = \boldsymbol{H}^*_{i,1}$, where $\boldsymbol{H}^*_{i,1}$ denotes the first equation in (14) evaluated at the fixed point $\boldsymbol{\mu}^{(i)}$, we immediately obtain $\boldsymbol{F}^*_{i-1} := \boldsymbol{F}_{i-1}(\boldsymbol{\mu}^{(i)}) = \boldsymbol{0}$. □

This theorem says that the fixed points of (13) are indeed the solutions of

$$\boldsymbol{F}_i(\boldsymbol{\lambda}_{i-1}, \lambda_i, \alpha_{i+1}, \ldots, \alpha_n) = \boldsymbol{0},$$

which is what we intend to solve on each iteration $i = 2, \ldots, n$. Next, we will establish the condition for the fixed point to be locally attracting. This condition will ensure that if we iterate the map in (14) with an initial condition that is close to the solution, then we will obtain the solution.

For local convergence, we want to show that eigenvalues of the Jacobian matrix $D\boldsymbol{H}^*_i := D\boldsymbol{H}_i(\boldsymbol{\mu}^{(i)})$ are in the interior of the unit ball of the complex plane. One can verify that the components of the Jacobian matrix $D\boldsymbol{H}^*_i$ are given by

$$\frac{\partial \boldsymbol{H}^*_{i,1}}{\partial \lambda_j} = -(\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1} \boldsymbol{F}^*_{i-1,\lambda_i} \frac{\partial H^*_{i,2}}{\partial \lambda_j}, \tag{16}$$

$$\frac{\partial H^*_{i,2}}{\partial \lambda_j} = \delta_{j,i} - \left(\frac{\partial F^*_i}{\partial \lambda_i}\right)^{-1} \frac{\partial F^*_i}{\partial \lambda_j}, \tag{17}$$

for $j = 1, \ldots, i$, where we have used all three conditions in Assumption 1 (see the Appendix for the detailed derivation). Here, $\delta_{j,i}$ is one only if $j = i$ and zero

otherwise. To simplify the discussion below, let's define the notations

$$\begin{aligned} J &:= \boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}}, \\ \boldsymbol{v} &:= \boldsymbol{F}^*_{i-1,\lambda_i}, \\ \boldsymbol{c} &:= \left(\frac{\partial H^*_{i,2}}{\partial \lambda_1}, \ldots, \frac{\partial H^*_{i,2}}{\partial \lambda_{i-1}}\right)^\top \end{aligned} \tag{18}$$

such that

$$D\boldsymbol{H}^*_{i+1} = \begin{pmatrix} J^{-1}\boldsymbol{v}\boldsymbol{c}^\top & \mathbf{0} \\ \boldsymbol{c}^\top & 0 \end{pmatrix} \in \mathbb{R}^{i\times i}. \tag{19}$$

We can now obtain the following result.

Theorem 4.2. *Let $\boldsymbol{\mu}^{(i)} \in \mathbb{R}^i$ be a fixed point of (13) such that the conditions in Assumption 1 are satisfied. Let $\sigma_j(\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}})$ be the eigenvalues of $\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}}$, and assume that they satisfy the order $|\sigma_1| \geq |\sigma_2| \geq \cdots \geq |\sigma_{i-1}|$. If*

$$\left|\left(\frac{\partial F^*_i}{\partial \lambda_i}\right)^{-1} \sum_{j=1}^{i-1} \frac{\partial F^*_j}{\partial \lambda_i}\frac{\partial F^*_i}{\partial \lambda_j}\right| < |\sigma_{i-1}(\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}})|, \tag{20}$$

then $\boldsymbol{\mu}^{(i)}$ is locally attracting.

Proof. From (19), we only need to analyze the eigenvalues of $J^{-1}\boldsymbol{v}\boldsymbol{c}^\top$. From basic matrix theory, recall that the magnitude of the largest eigenvalue can be bounded above as

$$|\sigma_1(J^{-1}\boldsymbol{v}\boldsymbol{c}^\top)| = \|J^{-1}\boldsymbol{v}\boldsymbol{c}^\top\|_2 \leq \|J^{-1}\|_2\|\boldsymbol{v}\boldsymbol{c}^\top\|_2,$$

where $\|\cdot\|_2$ denotes the matrix ℓ_2-norm. For the fixed point to be locally attracting, all of the eigenvalues of $J^{-1}\boldsymbol{v}\boldsymbol{c}^\top$ have to be in the interior of the unit ball in the complex plane. This means that we only need to show that $\|J^{-1}\|_2\|\boldsymbol{v}\boldsymbol{c}^\top\|_2 < 1$ or $\|\boldsymbol{v}\boldsymbol{c}^\top\|_2 < |\sigma_{i-1}(J)|$, where $\sigma_{i-1}(J)$ denotes the smallest eigenvalue of the $(i-1)\times(i-1)$ matrix J following the ordering in the hypothesis.

Since $\mathrm{Tr}(\boldsymbol{v}\boldsymbol{c}^\top) = \sum_{j=1}^{i} \sigma_j(\boldsymbol{v}\boldsymbol{c}^\top)$ and $\boldsymbol{v}\boldsymbol{c}^\top$ is a rank-one matrix, then its nontrivial eigenvalue is given by

$$\sigma(\boldsymbol{v}\boldsymbol{c}^\top) = \mathrm{Tr}(\boldsymbol{v}\boldsymbol{c}^\top) = \sum_{j=1}^{i-1} \frac{\partial F^*_j}{\partial \lambda_i}\frac{\partial H^*_{i,2}}{\partial \lambda_j} = -\sum_{j=1}^{i-1} \frac{\partial F^*_j}{\partial \lambda_i}\frac{\partial F^*_i}{\partial \lambda_j}\left(\frac{\partial F^*_i}{\partial \lambda_i}\right)^{-1},$$

where we have used the definitions in (18) and the second component in (17). From the assumption in (20), we have

$$\|\boldsymbol{v}\boldsymbol{c}^\top\|_2 = |\sigma(\boldsymbol{v}\boldsymbol{c}^\top)| = \left|\left(\frac{\partial F^*_i}{\partial \lambda_i}\right)^{-1} \sum_{j=1}^{i-1} \frac{\partial F^*_j}{\partial \lambda_i}\frac{\partial F^*_i}{\partial \lambda_j}\right| < |\sigma_{i-1}(J)|. \qquad \square$$

This theorem provides the conditions for local convergence on each iteration i. In particular, if the hypothesis in Theorem 4.2 is satisfied, we will find the solutions to (8) by iterating (13) provided that we start with a sufficiently close initial condition. Notice also that this condition suggests that in practice the local convergence will be difficult to satisfy if the Jacobian matrix $F_{i-1,\boldsymbol{\lambda}_{i-1}}$ is close to singular. With these two theorems, we can now establish:

Theorem 4.3. *Let $\boldsymbol{\mu}^{(n)} \in \mathbb{R}^n$ be the solution of the n-dimensional system of equations in (7). We assume the hypothesis in Theorem 4.2; then the EBE method is locally convergent.*

Proof. Choose an initial condition $(\alpha_1, \ldots, \alpha_n)$ that is sufficiently close to the solution $\boldsymbol{\mu}^{(n)}$ of $\boldsymbol{F}_n(\boldsymbol{\lambda}_n) = \mathbf{0}$. First, let us define the surface $F_1(\lambda_1, \ldots, \lambda_n) = 0$ as $\mathcal{M}_n$; here, the dimension of $\mathcal{M}_n$ is at most $n-1$. Subsequently, we define the surfaces $\boldsymbol{F}_2(\boldsymbol{\lambda}_n) = \mathbf{0}$ as $\mathcal{M}_{n-1}$, $\boldsymbol{F}_3(\boldsymbol{\lambda}_n) = \mathbf{0}$ as $\mathcal{M}_{n-2}$, and so on. The dimension of $\mathcal{M}_j$ is at most $j-1$. We assume that $\boldsymbol{F}_n(\boldsymbol{\lambda}_n) = \mathbf{0}$ has at least one solution; then $\mathcal{M}_1$ contains the solution $\boldsymbol{\mu}^{(n)}$. It is clear that $\mathcal{M}_n \supset \mathcal{M}_{n-1} \supset \cdots \supset \mathcal{M}_1$.

For $i = 1$, we solve $F_1(\lambda_1, \alpha_2, \ldots, \alpha_n) = 0$ for λ_1. Geometrically, we look for the first coordinate on the surface $\mathcal{M}_n$. From Assumption 1(2), we have the local convergence of the usual Newton's iteration. If α_1 is sufficiently close to the solution $\boldsymbol{\mu}^{(1)} = \mu_1^{(1)} \in \mathbb{R}$, as $m \to \infty$ we obtain the solution $(\mu_1^{(1)}, \alpha_2, \ldots, \alpha_n) \in \mathcal{M}_n$. By the smoothness assumption, $(\mu_1^{(1)}, \alpha_2, \ldots, \alpha_n)$ is also close to $\boldsymbol{\mu}^{(n)}$.

Continuing with $i > 1$, we want to solve $\boldsymbol{F}_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = \mathbf{0}$ for $\boldsymbol{\lambda}_i$. Numerically, we will apply the iterative map $\boldsymbol{H}_i$ in (13) starting from $(\boldsymbol{\mu}^{(i-1)}, \alpha_i, \ldots, \alpha_n) \in \mathcal{M}_{n-i+2}$. By Assumption 1(2), the Jacobian $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\mu}^{(i-1)}, \alpha_i, \ldots, \alpha_n)$ is nonsingular, so by the implicit function theorem, for any local neighborhood V of $\boldsymbol{\mu}^{(i-1)}$, there exists a neighborhood U of α_i and a C^1 function $\boldsymbol{h}_{i-1} : U \to V$ such that $\boldsymbol{\mu}^{(i-1)} = \boldsymbol{h}_{i-1}(\alpha_i)$ and $\boldsymbol{F}_{i-1}(\boldsymbol{h}_{i-1}(\lambda_i), \lambda_i, \alpha_{i+1}, \ldots, \alpha_n) = 0$ for all $\lambda_i \in U$. Since the initial condition α_i is close to $\mu_i^{(n)}$, by the smoothness assumption it is also close to $\mu_i^{(i)}$ that solves $\boldsymbol{F}_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = 0$. The continuity of $\boldsymbol{h}_{i-1}$ on U means that $(\boldsymbol{\mu}_{i-1}^{(i)}, \mu_i^{(i)}) \in V \times U$. Geometrically, this means the surface $F_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = 0$ intersects with the curve $\boldsymbol{\lambda}_{i-1} = \boldsymbol{h}_{i-1}(\lambda_i)$ at $\boldsymbol{\mu}^{(i)} = (\boldsymbol{\mu}_{i-1}^{(i)}, \mu_i^{(i)})$. Therefore, we can find the solution for this i-dimensional system by tracking along the curve $\boldsymbol{\lambda}_{i-1} = \boldsymbol{h}_{i-1}(\lambda_i)$ where we consider λ_i as an independent parameter. The iterative map $\boldsymbol{H}_i$ in (14) is to facilitate this tracking, and the conditions in Theorem 4.2 guarantee convergence to the solution. Notice that during this iteration, the solution remains on $\mathcal{M}_{n-i+2}$. The solution for this i-dimensional problem is $(\boldsymbol{\mu}^{(i)}, \alpha_{i+1}, \ldots, \alpha_n) \in \mathcal{M}_{n-(i+1)+2} \subset \mathcal{M}_{n-i+2} \subset \cdots \subset \mathcal{M}_n$. Continuing with the same argument, we find that for $i = n$, $\boldsymbol{\mu}^{(n)} \in \mathcal{M}_1 \subset \mathcal{M}_n$. □

This iterative procedure finds the solution by searching along the manifold $\mathcal{M}_n$ in the direction of the hypersurfaces of a single parameter at a time, whose

local existence is guaranteed by Assumption 1. It is clear that after each step i, the estimated solution may not necessarily be closer to the true solution since the estimates do not minimize the closest path to the true solution along the manifold $\mathcal{M}_n$ (or the geodesic distance). This means that, locally,

$$\|(\boldsymbol{\mu}^{(i+1)}, \alpha_{i+2}, \ldots, \alpha_n) - \boldsymbol{\mu}^{(n)}\| \leq \|(\boldsymbol{\mu}^{(i)}, \alpha_{i+1}, \ldots, \alpha_n) - \boldsymbol{\mu}^{(n)}\|$$

for $i < n-1$ is not true.

In practice, when initial conditions are not close to the solution, the (global) convergence of EBE requires the additional condition that, for every i, there exists a nonempty connected set that contains $(\boldsymbol{\mu}^{(i)}, \alpha_{i+1})$ and $\boldsymbol{\mu}^{(i+1)}$ such that $\boldsymbol{F}_{i,\lambda_i}$ evaluated at any point in this set is nonsingular. The existence of this set will allow us to build a path to connect these two points that are far apart. If this condition is not met, we need an additional treatment to overcome this issue which will be discussed in the next section.

5. Practical challenges

In this section, we will discuss several practical challenges related to our algorithm with remedies. They include nonlocality of the initial condition, mistracking due to multiple solutions, nonexistence of solutions within the desired numerical tolerance, and the computational complexity.

Adaptive tracking. As we mentioned in the previous section, the EBE method only converges locally, which means that it requires an adequate initial condition which is practically challenging. In our numerical simulations below, in fact, we always start from zero initial condition, $\alpha_i = 0$ for all $i = 1, \ldots, n$. In this case, notice that even when we obtain an accurate solution at step i, that is, $\boldsymbol{F}_i(\hat{\boldsymbol{\mu}}^{(i)}) \approx \mathbf{0}$, as we proceed to the next iteration, $|F_{i+1}(\hat{\boldsymbol{\mu}}^{(i)}, \alpha_{i+1})| \gg 0$, meaning that $(\hat{\boldsymbol{\mu}}^{(i)}, \alpha_{i+1})$ is not close to the solution $\boldsymbol{\mu}^{(i+1)}$. Even when $\frac{\partial F_{i+1}}{\partial \lambda_{i+1}}(\hat{\boldsymbol{\mu}}^{(i)}, \alpha_{i+1})$ is not singular, according to (9), λ_i^{m+1} could be very far away from λ_i^m. In this case, Newton's method could fail in (12) because the initial guess could be very far from the solution.

As a remedy, we employ an adaptive tracking on λ_i to guarantee that the application of Newton's method is within its zone of convergence for each predictor-corrector step. The idea of the adaptive tracking is that we cut the tracking step, $\Delta\lambda_i := \lambda_{i+1} - \lambda_i$, by half until the prediction-correction step in (11)–(12) converges. The algorithm is outlined in Algorithm 1.

Bifurcation. In order to solve $F_i(\lambda_1, \lambda_2, \ldots, \lambda_i) = 0$, we track $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \mathbf{0}$ along λ_i as a parameter. During this parameter tracking, we may have some bifurcation points of λ_i for the nonlinear system $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \mathbf{0}$. This means that the Jacobian $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}, \lambda_i)$ is rank deficient such that $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \mathbf{0}$ has

Input minimum step size $\lambda_{\min}$ and threshold value of Tol.
Compute $\Delta\lambda_i$ by using Newton's method to solve $F_i = 0$.
Set Final $= \Delta\lambda_i$.
while |Final| > 0 **do**
    Solve $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i + \Delta\lambda_i) = 0$ by using Newton's method.
    if Newton's method fails **then**
        $\Delta\lambda_i = \Delta\lambda_i/2$
        if $\Delta\lambda_i < \lambda_{\min}$ **then**
            Discard the i-th equation.
        end
    else
        Final $=$ Final $- \Delta\lambda_i$
        $\Delta\lambda_i = \min\{\Delta\lambda_i, \text{Final}\}$
    end
end

Algorithm 1. Summary of adaptive tracking algorithm.

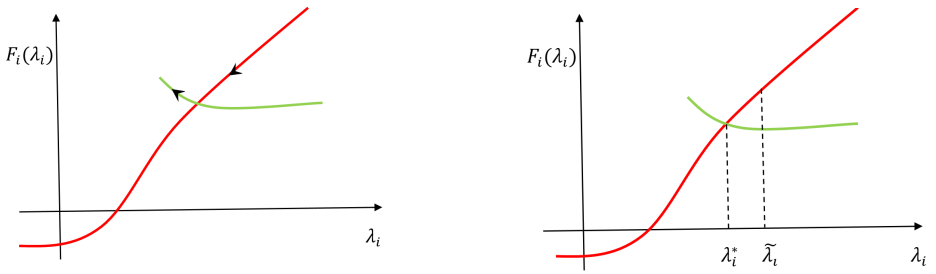

Figure 1. Plot of $F_i(\lambda_i)$ versus λ_i. There are two bifurcation branches for the nonlinear system $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \mathbf{0}$. The left part is a mistracking example; the right part is the illustration of a numerical method to avoid the bifurcation point.

multiple solutions $\boldsymbol{\lambda}_{i-1}$ for a given λ_i. In this situation, F_i has multiple realization functions of λ_i. See the illustration in Figure 1 where the bifurcation point is the intersection of the two possible realizations of F_i. In this illustration, the goal is to track along the red branch to find the root, $F_i(\lambda_i) = 0$. As we get closer to the bifurcation point, the Jacobian $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}, \lambda_i)$ is singular such that we can't evaluate (11). Intuitively, the existence of multiple solutions near the bifurcation point induces a possibility of mistracking from the red curve to the green curve (as shown by the arrows), which prohibits one from finding the solution.

To avoid such mistracking, we apply the deflation technique to compute the bifurcation point directly [12; 16]. Once the bifurcation point is estimated, we approximate the correct branches using Richardson extrapolation to avoid mistracking. Denoting the bifurcation point as λ_i^*, the nonlinear system $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \mathbf{0}$

is difficult to solve when λ_i is close to λ_i^* since the Jacobian of $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i)$ becomes near singular. If the last attempt is $(\tilde{\boldsymbol{\lambda}}_{i-1}, \tilde{\lambda}_i)$, we compute $(\boldsymbol{\lambda}_{i-1}^*, \lambda_i^*)$ by solving the deflated system

$$G(\boldsymbol{\lambda}_{i-1}^*, \lambda_i^*, \boldsymbol{v}) = \begin{pmatrix} \boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) \\ \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}, \lambda_i)\boldsymbol{v} \\ \boldsymbol{\xi}^\top \boldsymbol{v} - 1 \end{pmatrix} = \boldsymbol{0},$$

where $\boldsymbol{v}$ is the kernel of $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{\lambda}_{i-1}, \lambda_i)$ and $\boldsymbol{\xi}$ is a random vector to guarantee that $\boldsymbol{v}$ is not a zero eigenvector. In this case, $G(\boldsymbol{\lambda}_{i-1}^*, \lambda_i^*, \boldsymbol{v})$ is well conditioned [12; 16]. Once the bifurcation point $(\boldsymbol{\lambda}_{i-1}^*, \lambda_i^*)$ is estimated, we can avoid mistracking by setting $\lambda_i = 2\lambda_i^* - \tilde{\lambda}_i$ and solve $\boldsymbol{F}_{i-1}(\boldsymbol{\lambda}_{i-1}, \lambda_i) = \boldsymbol{0}$ by using Newton's method with an initial guess $2\boldsymbol{\lambda}_{i-1}^* - \tilde{\boldsymbol{\lambda}}_{i-1}$ (which is a Richardson extrapolation).

Nonexistence of solutions. In general, the moment constrained maximum entropy problems may not necessarily have solutions. Even when the solutions exist theoretically, they could be difficult to find numerically due to the noisy dataset, error in the numerical integration, etc. In this case, we simply discard the equation F_i when the minimum is larger than the desired tolerance. This feature (discarding the constraints that give no solutions) is only feasible in the EBE algorithm. However, some theories are needed to preserve the convexity of the polynomials in the exponential term of (4) while discarding some of these constraints. In our numerical simulations below, we handle this issue by reordering the constraints. In particular, for a problem with moment constraints up to order four, we include the constraints corresponding to $\mathbb{E}[x_i^4]$, $i = 1, \ldots, d$, in the earlier step of the EBE iterations to avoid these constraints being discarded. Note that this method is sensitive to ordering, that is, different ordering of constraints yields different paths to compute the solution. Therefore, a systematic ordering technique that simultaneously preserves the convexity of the polynomial in the exponential term of (4) is an important problem to be addressed in the future.

Computational complexity. The most expensive computational part in EBE is the numerical evaluation of (6). For a fast numerical integration, we store the monomial basis $c_j(\boldsymbol{x})$ as a matrix of size $N_\ell \times n$, where N_ℓ is the number of sparse grid points and n is number of monomial basis. In this case, the computational cost in evaluating F_j is $(2j+1)N_\ell$ ($j-1$ additions, $j+1$ multiplications, and 1 subtraction for each grid point), excluding the computational cost for exponential function evaluation, which is on the order of $\log^2 m$ to obtain an error of resolution 2^{-m} [6]. For the i-th iteration of the EBE algorithm, the computational cost to evaluate the i-dimensional system $\boldsymbol{F}_i$ is $\sum_{j=1}^{i}(2j+1)N_\ell = \frac{1}{2}(i^2+i)N_\ell$, excluding the exponentiation.

6. Numerical results

In this section, we show numerical results of the EBE method on five examples. In all of the simulations below, unless stated, we set the Newton's tolerance $\text{Tol}_1 = 10^{-1}$ and the predictor tolerance $\text{Tol}_2 = 10^{-10}$. In the first test example, we will describe how the EBE method works on each iteration. The goal of the second example is to demonstrate the global convergence with solutions that are far away from initial condition, $\alpha_j = 0$. In particular, we will test the EBE method on a problem with solutions, λ_j, that have magnitudes ranging from orders 10^0–10^3. In this example, we will show the robustness of the estimate as a function of the number of integration points (or the sparse grid level ℓ). The third example demonstrates the performance on high-dimensional problems (with $70 \le n \le 310$ of order one hundred), induced from order-four moments of four- to seven-dimensional density functions. While these first three examples involve estimating densities of the form (4), in the next two examples, we also test the EBE method to estimate densities from a given data set where the maximum entropy solutions may or may not exist. The first data-driven problem is to estimate densities of the first two leading EOFs of the wind stress-driven large-scale oceanic model [3; 4]. The second data-driven problem is to estimate two- to five-dimensional densities arising from solutions of the Kuramoto–Sivashinsky equation. In these two problems, we compare our method with the classical Newton's method, the MATLAB built-in solver fsolve, and the previously developed BFGS-based method [3; 4].

Example 1. We consider a simple example $\rho(x) \propto \exp(x + x^2 + x^3)$ for $x \in [-1, 1]$ so that the exact solution is $\boldsymbol{\lambda} = (1, 1, 1)$. Here, the moments f_j can be computed numerically by

$$f_j = \frac{\int_{-1}^{1} x^j \rho(x)\, dx}{\int_{-1}^{1} \rho(x)\, dx} \quad \text{for } i = 1, 2, 3.$$

In order to numerically integrate both the denominator and numerator, we used a regular one-dimensional sparse grid of level $\ell = 7$ (the number of nodes is 65). Our goal here is to illustrate the method and to show the trajectory of the solutions after each iteration of the inner loop m and outer loop i. In Figure 2, we show the surface of $F_1(\lambda_1, \lambda_2, \lambda_3) = 0$ (gray). For $i = 1$, we solve $F_1(\lambda_1, 0, 0) = 0$; after three iterations ($m = 3$) the solution converges to $\lambda_1 = 2.3$ (see Table 1). For $i = 2$, we start with this solution and introduce the second variable λ_2 for solving the second equation $F_2(\lambda_1, \lambda_2, 0) = 0$ with constraint $F_1(\lambda_1, \lambda_2, 0) = 0$. Here, the solution follows the path $\lambda_1 = h_1(\lambda_2)$ thanks to the implicit function theorem (black curve). Numerically, a sequence of (green) points following this path converges to a point that satisfies $F_1(\lambda_1, \lambda_2, 0) = F_2(\lambda_1, \lambda_2, 0) = 0$ (the green point in the intersection between black and red curves in Figure 2). In the next iteration $i = 3$, we

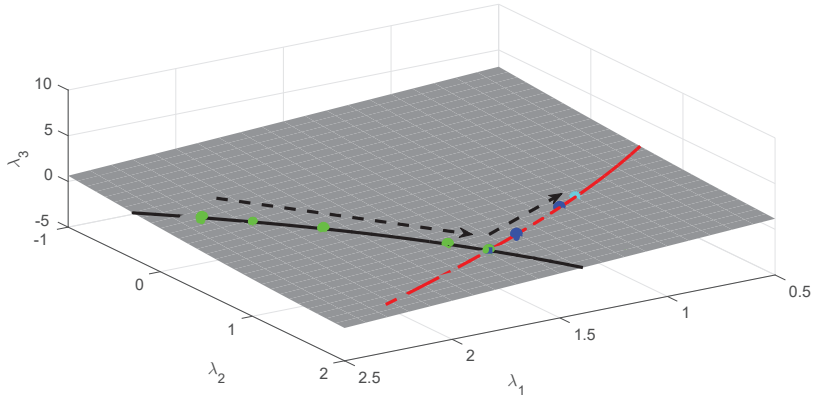

Figure 2. The illustration of Example 1. The black curve is $\lambda_1 = h_1(\lambda_2)$, the green points are the iterations when we solved $F_1(\lambda_1, \lambda_2, 0) = 0$, the red curve is $(\lambda_1, \lambda_2) = h_2(\lambda_3)$, the blue points are the iterations when we solved $F_1(\lambda_1, \lambda_2, \lambda_3) = F_2(\lambda_1, \lambda_2, \lambda_3) = 0$, and the cyan point is the numerical solution.

$m \downarrow \ i \rightarrow$	1	2	3
0	(0, 0, 0)	(2.30, 0, 0)	(1.58, 1.43, 0)
1	(1.76, 0, 0)	(2.23, 0.22, 0)	(1.52, 1.38, 0.26)
2	(2.23, 0, 0)	(1.87, 0.57, 0)	(1.12, 1.09, 0.76)
3	(2.30, 0, 0)	(1.67, 1.21, 0)	(1, 1, 1)
4		(1.58, 1.43, 0)	

Table 1. The coordinate of the solutions of Example 1 for each iteration, starting from (0, 0, 0). For each outer loop i, the EBE takes few iterates (m) to find the i-dimensional solution, fixing $\lambda_j = \alpha_j = 0$ for $j > i$.

introduce the third variable λ_3 for solving the third equation $F_3(\lambda_1, \lambda_2, \lambda_3) = 0$ with constraints $F_1(\lambda_1, \lambda_2, \lambda_3) = F_2(\lambda_1, \lambda_2, \lambda_3) = 0$. By the implicit function theorem, we have $(\lambda_1, \lambda_2) = h_2(\lambda_3)$ that satisfies $F_1(h_2(\lambda_3), \lambda_3) = F_2(h_2(\lambda_3), \lambda_3) = 0$, which is shown by the red curve in Figure 2. On this red curve, we have a sequence of (blue) points which converges to the solution of the full system (cyan point shown in Figure 2). The coordinate of the solution on each iteration is shown in Table 1. Notice that the solutions always lie on the surface $F_1(\lambda_1, \lambda_2, \lambda_3) = 0$.

Example 2. We consider a one-dimensional example with up to order-six moment constraints with explicit solution given by

$$\rho(x) \propto \exp(2x + 16x^2 + 24x^3 + 96x^4 - 256x^5 - 1024x^6),$$

as shown in Figure 3. This example is a tough test problem since the solution, $\boldsymbol{\lambda} = (2, 16, 24, 96, -256, 1024)$, has components of order 10^0–10^3. Following

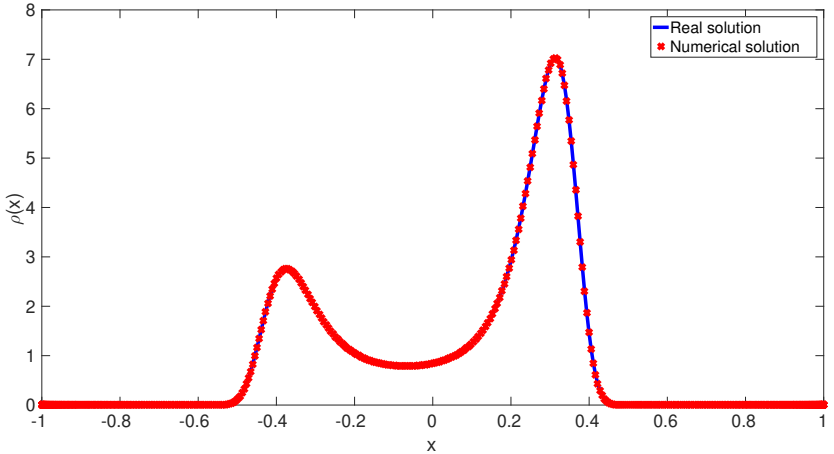

Figure 3. The unnormalized density $\rho(x)$ in Example 2.

Example 1, we compute the moments f_i by using a one-dimensional sparse grid of level $\ell = 7$ (65 nodes). The EBE algorithm converges to the exact solution with error $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| = 5.44 \times 10^{-13}$. Since the numerical experiment is performed with an initial condition $\alpha_j = 0$ that is far from the solution, this result demonstrates a global convergence of the EBE method.

Next, we investigate the sensitivity of the estimates to the number of sparse grid points used in approximating the integral. In our numerical experiments, we estimate the true moments f_i using a one-dimensional sparse grid of level $\ell = 20$ (524 289 nodes) and feed these moment estimates into the EBE algorithm. In Figure 4, we show the error in λ (with ℓ_2-metric) for different levels of the sparse grid from 6 to 15 that are used in the EBE method. Notice that the error decreases as a function of ℓ and the improvement becomes negligible for $\ell > 8$.

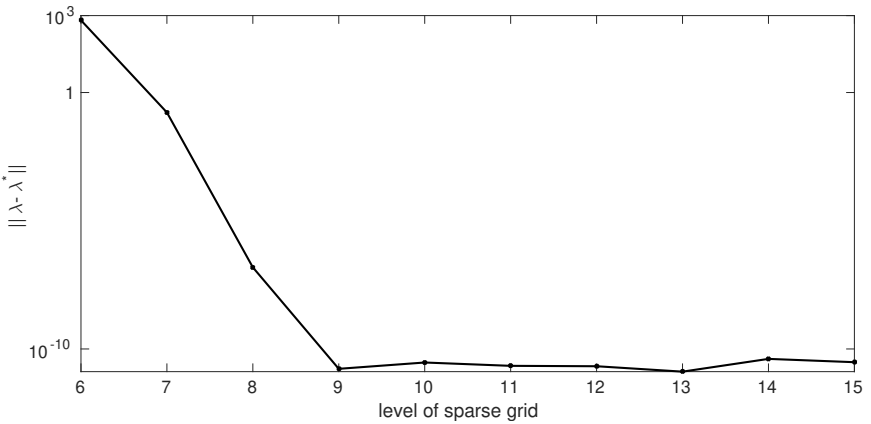

Figure 4. The solution error as a function of the number of sparse grid.

Methods	Order		
	4	6	8
BFGS algorithm with uniform grid	4.07×10^{-2}	1.45×10^{-4}	1.14×10^{-2}
EBE algorithm with uniform grid	1.27×10^{-11}	9.84×10^{-15}	7.75×10^{-13}
EBE algorithm with sparse grid	7.54×10^{-12}	8.12×10^{-15}	2.43×10^{-13}
MATLAB fsolve with sparse grid	4.70×10^{-7}	1.19×10^{-4}	1.74×10^{-4}
Newton with sparse grid	5.12×10^{-11}	divergence	divergence

Table 2. Summary of solutions for Example 4: moment errors for different algorithms with different grids.

Example 3. In this example, we consider a d-dimensional example with an explicit solution,

$$\rho(x) \propto \exp(-2x_1^4 + x_2^3 - x_2^4 - x_3^4 - 1.8x_4^4),$$

on domain $\Omega = [-1, 1]^d$ where we will vary $d = 4, \ldots, 7$. For these simulations, we consider up to order-four moment constraints and fix the sparse grid level $\ell = 8$ to compute the integration.

Here, the EBE method is able to estimate $\boldsymbol{\lambda}$ with ℓ_2-errors of order 10^{-13} (the error in $\boldsymbol{\lambda}$ is 1.11×10^{-13} and moments error is 3.15×10^{-15}). In this computation, the dimensions of the nonlinear system are 70 for $d = 4$, 126 for $d = 5$, 210 for $d = 6$, and 310 for $d = 7$. Here, the EBE method is able to recover the true density even if we prescribe more constraints, corresponding to d larger than four.

Example 4. Next, we consider estimating a two-dimensional probability density of the two leading empirical orthogonal functions of a geophysical model for a wind stress-driven large-scale oceanic model [18; 19]. This is exactly the same test example as in the previously developed BFGS-based method [3; 4]. In fact, the two-dimensional density that we used here was supplied by Rafail Abramov. First, we compare the EBE method with the BFGS algorithm of [3], whose code can be downloaded from [2]. In this comparison, we use the same uniformly distributed grid points where the total number of nodes is $85 \times 85 = 7\,225$. We set the Newton's tolerance of the EBE algorithm to be 10^{-10}. In Table 2 notice that the moment errors of the EBE are much smaller compared to those of the BFGS method.

While the EBE is superior compared to BFGS, we should note that the BFGS method does not use the Hessian of $\boldsymbol{F}_i$ whereas the EBE does. For a fair comparison, we include results using the MATLAB built-in function fsolve, whose default algorithm is the trust-region-dogleg (see the documentation for detail [17]). In our numerical implementation, we apply fsolve with a specified Hessian function $\boldsymbol{F}_n$. We also include the classical Newton's method with a specified Hessian function $\boldsymbol{F}_n$. In this comparison, we use the same sparse grid of level $\ell = 11$ (or 7 169 nodes) to

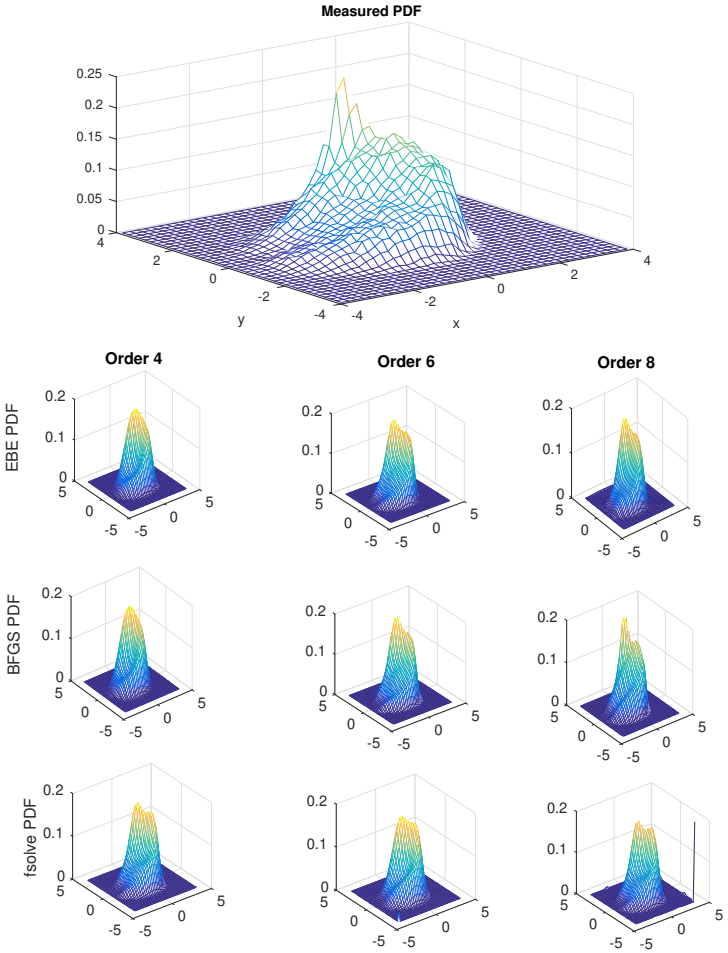

Figure 5. The 2D measured probability density functions supplied by R. Abramov (first row), and PDFs computed by the EBE method (second row), BFGS algorithm (third row), and the MATLAB fsolve function (fourth row).

compute the two-dimensional integral. Notice that the EBE method is still superior compared to these two schemes as reported in Table 2. In fact, Newton's method does not converge for higher-order moment constraints. The joint two-dimensional PDFs are shown in Figure 5. The first row is the two-dimensional density function provided by R. Abramov. The second row shows the EBE estimates using up to order-four, -six, and -eight moment constraints. The third and fourth rows show the BFGS and MATLAB fsolve estimates, respectively.

Example 5. In this example, we consider estimating multidimensional densities of the solutions of the Kuramoto–Sivashinsky equation. Here, the solutions are integrated with a fourth-order time-differencing method on 128 equally spaced grid

d	EBE method	fsolve	Newton
2	1.098×10^{-15}	9.779×10^{-7}	8.128×10^{-14}
3	4.29×10^{-13}	3.150×10^{-2}	divergence
4	1.19×10^{-14}	0.021	divergence
5	2.47×10^{-11}	0.018	divergence

Table 3. Summary of solutions for Example 5.

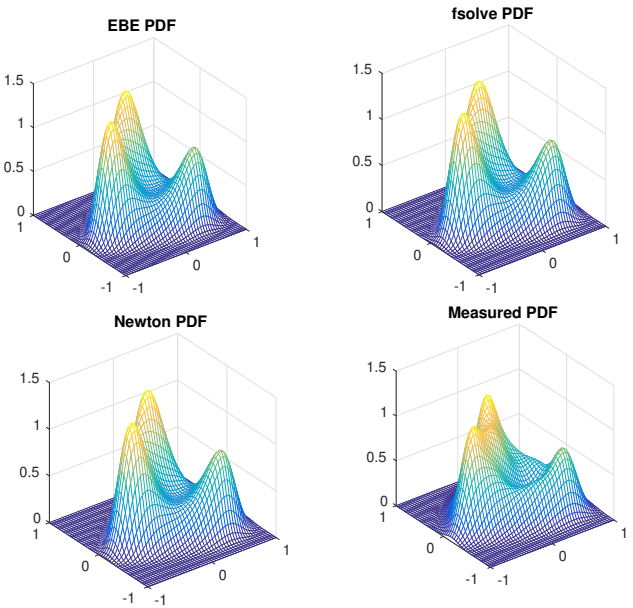

Figure 6. The comparison of the density functions obtained by the EBE algorithm, the MATLAB fsolve function, Newton's method, and the kernel density estimate (denoted as the measured PDF) for the two-dimensional case.

points over a domain of $[0, 32\pi]$ as in [15]. We use initial condition $u(x, 0) = \cos(x/(16\xi))(1 + \sin(x/16))$, with $\xi \sim U[0, 1]$ and integration time step 0.25. The data is generated by integrating 10 000 time steps. Based on this data set, we randomly select d components and estimate the d-dimensional joint density associated to these components. For visual comparison, we also show the results from a two-dimensional kernel density estimation method [22; 21] as a reference. Numerically, we use the MATLAB built-in function, ksdensity. Note that the BFGS algorithm [3] does not work on this data set while the classical Newton's method only converges for the two-dimensional case. We also show the corresponding results with the MATLAB fsolve with specified Hessian function as in the previous example. The moment errors of these three schemes are reported in Table 3.

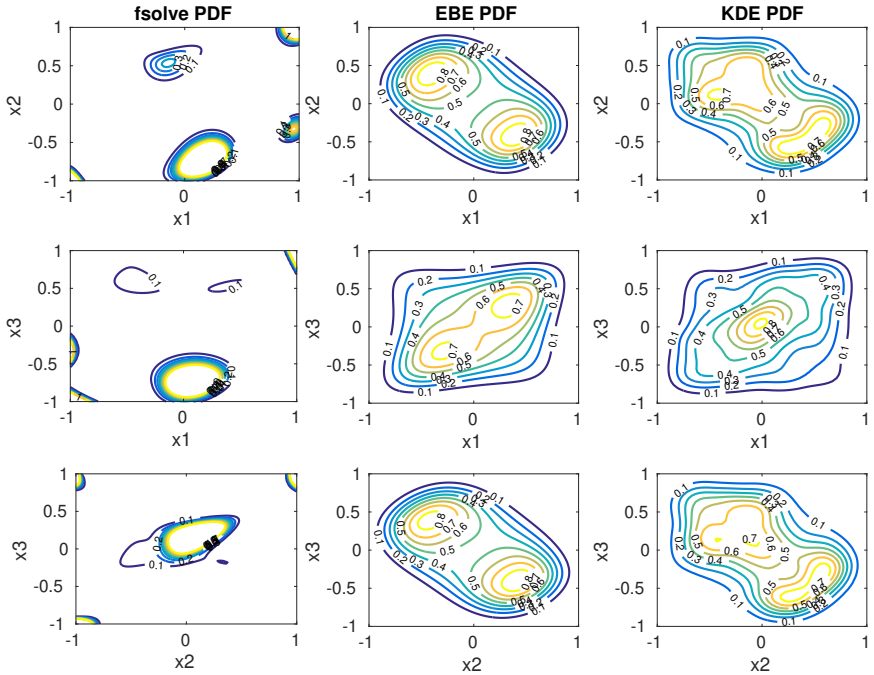

Figure 7. The comparison of the two-dimensional marginal density functions obtained by the MATLAB fsolve function (first column), the EBE algorithm (second column) that solves a three-dimensional problem accounting for up to order-four moment constraints, and the two-dimensional kernel density estimate (third column).

In Figure 6, we show the two-dimensional density estimated by the EBE algorithm compared to those from fsolve, the classical Newton's method, and the 2D kernel density estimate. For the two-dimensional case, the resulting densities are visually identical although the corresponding moment error of the EBE method is still the smallest compared to Newton's and the MATLAB fsolve (see Table 3). In Figure 7, we show the contour plot of the two-dimensional marginal densities obtained from solving the three-dimensional problem given four-moment constraints with the EBE method and the MATLAB fsolve. For diagnostic purposes, we also provide the corresponding contour plots of the two-dimensional kernel density estimates. Notice that the MATLAB fsolve produces a completely inaccurate estimate. The EBE method produces an estimate that qualitatively agrees to the corresponding two-dimensional KDE estimates. The slight disagreement between these estimates is expected since we only provide up to order-four moment information.

In Figure 8, we show the results for the four-dimensional problem. We do not show the estimate from the MATLAB fsolve since it is not accurate at all. Here, we include more than four-order moments. Specifically, the total number of constraints for up to order-four moments is 70 while this result is based on 87 constraints,

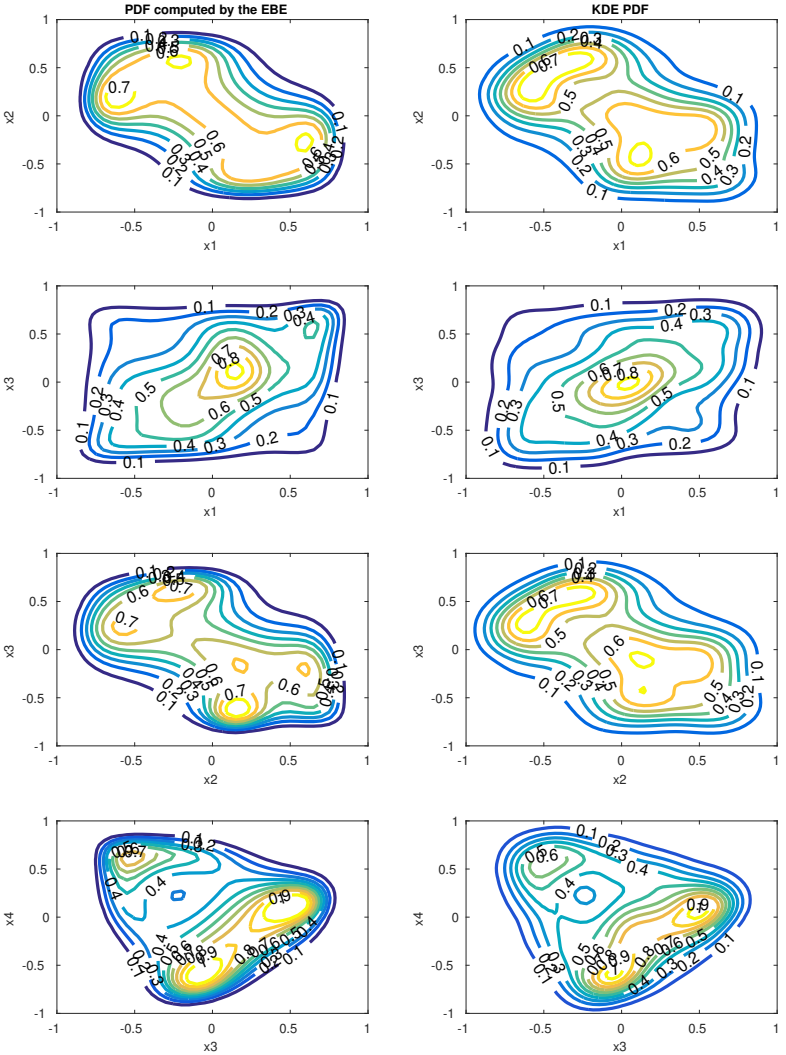

Figure 8. The comparison of the two-dimensional marginal density functions obtained by the EBE algorithm (first column) that solves a four-dimensional problem accounting for more than order-four moment constraints (see text for detail) and the two-dimensional kernel density estimate (second column).

including 17 additional higher-order moment constraints that include order-six moments, $\mathbb{E}[x_i^6]$, $i = 1, \ldots, 4$. See the movie of the density estimates for each iteration in the supplementary material [11]. Notice that the marginal densities estimated by the EBE look very similar to those estimated by the two-dimensional kernel density estimation. If more constraints are included, we found that we lose the convexity of the polynomial terms in (4). As we mentioned before, we need better criteria to preserve the convexity of the solutions.

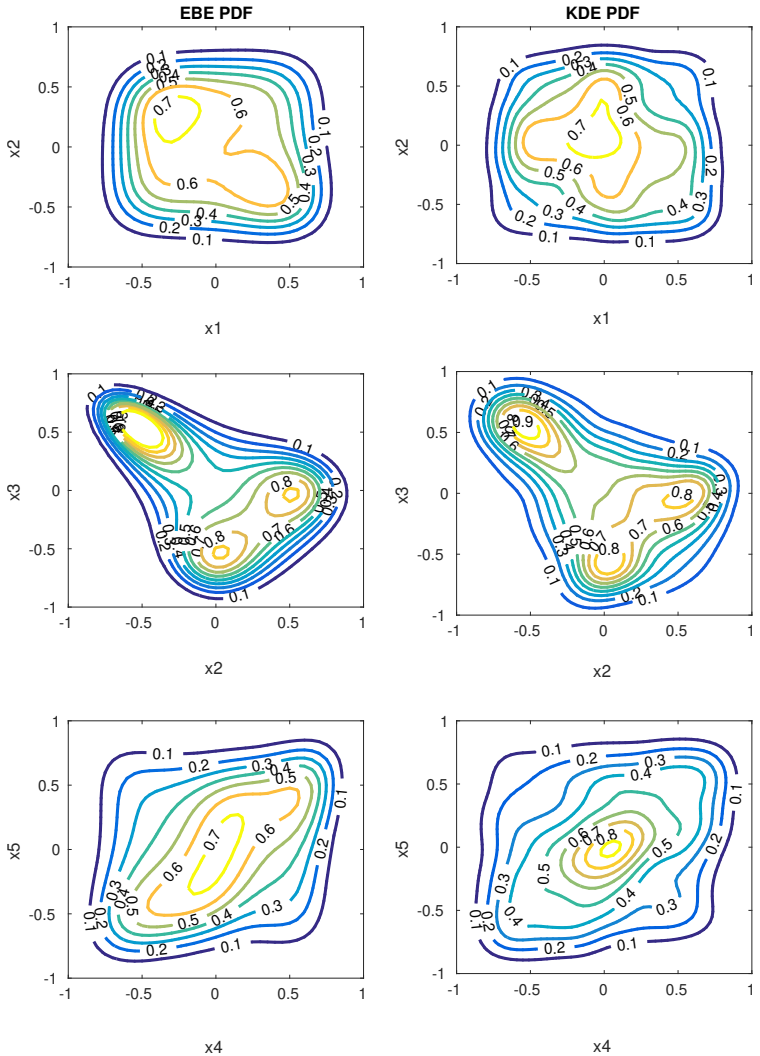

Figure 9. The comparison of the two-dimensional marginal density functions obtained by the EBE algorithm (first column) that solves a five-dimensional problem accounting for the automatically selected 91 out of the prescribed 125 moments, and the two-dimensional kernel density estimate (second column).

In Figure 9, we include the result from a five-dimensional simulation. We also do not show the estimate from the MATLAB fsolve since it is not accurate at all. In this five-dimensional case, the EBE method automatically discards 34 equations (moment constraints). In this case, we suspect that either the maximum entropy solution that accounts for all of the constraints does not exist or the EBE method cannot find the solution. Here, the EBE method just estimates the best-fitted solution within the tolerance of 10^{-10} by solving 91 out of 125 moment constraints.

7. Summary

In this paper, we introduced a novel equation-by-equation algorithm for solving a system of nonlinear equations arising from the moment constrained maximum entropy problem. Theoretically, we have established the local convergence and provided a sufficient condition for global convergence. Through the convergence analysis, we understood that the method, geometrically, finds the solution by searching along the surface corresponding to one component of the nonlinear equations. Numerically, we have demonstrated its accuracy and efficiency on various examples. In one of the examples, we found that the EBE algorithm produces more accurate solutions compared to the previously developed BFGS-based algorithm which does not use the Hessian information [3; 4]. In this same example, we also found that the EBE is superior compared to two schemes that use the Hessian information, including the current MATLAB built-in solver which uses the trust-region-dogleg algorithm and the classical Newton's method.

We also found that the proposed EBE algorithm is able to solve a system of 70–310 equations when the maximum entropy solution exists compared to the previously developed BFGS method which was shown to work for a system of size 44–83 equations. On the Kuramoto–Sivashinsky example, the EBE method is able to reconstruct the density of a four-dimensional problem accounting for up to order-four moments (or 70 constraints). In this case, we showed that the estimate is improved by accounting for 17 additional constraints of order-six moments. For the five-dimensional problem with moments up to order four, the EBE method reconstructs the solution within the desired precision, 10^{-10}, by automatically selecting a subset of 91 constraints from the total prescribed 125 constraints induced by moments of up to order four.

While the automatic constraint selection is a desirable feature since the maximum entropy solutions within the tolerance may not be easily estimated (nor theoretically available), further study is required to fully take advantage of this feature. In particular, an important open problem is to develop a mathematical theory for ordering the constraints since the path of the solution is sensitive to the order of the constraints. Simultaneously, the ordering of the constraints needs to preserve the convexity of the polynomials in the exponential term of (4). We should stress that the EBE method is computationally not the most efficient method since it is designed to avoid singularities by tracking along the surface corresponding to one component of the nonlinear equations. Therefore, a more efficient EBE method will be one of future goals.

Appendix: The detailed calculation of the Jacobian of the map H_i

In this appendix, we will give the detailed computation for the Jacobian of the map H_i in (14) evaluated at $\boldsymbol{\mu}^{(i)}$, the solution of $\boldsymbol{F}_i(\boldsymbol{\lambda}_i, \alpha_{i+1}, \ldots, \alpha_n) = \mathbf{0}$. Recall

that for $\boldsymbol{H}_i = (\boldsymbol{H}_{i,1}, H_{i,2})$ in (14),

$$\boldsymbol{H}_{i,1}(\boldsymbol{\lambda}_i) = \boldsymbol{g}_i - \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}(\boldsymbol{g}_i, H_{i,2})^{-1}\boldsymbol{F}_{i-1}(\boldsymbol{g}_i, H_{i,2}),$$
$$H_{i,2}(\boldsymbol{\lambda}_i) = \lambda_i - \left(\frac{\partial F_i}{\partial \lambda_i}(\boldsymbol{\lambda}_i)\right)^{-1} F_i(\boldsymbol{\lambda}_i),$$

where $\boldsymbol{g}_i : \mathbb{R}^{i-1} \to \mathbb{R}^{i-1}$ is defined as in (15).

To take another derivative of $\boldsymbol{H}_{i,1}$ with respect to λ_j, we use the fact that if $\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}$ is a nonsingular matrix, then

$$\frac{\partial}{\partial \lambda_j}(\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1} = (\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}\frac{\partial \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}}{\partial \lambda_j}(\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1},$$

and the Hessian $\frac{\partial \boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}}}{\partial \lambda_j}$ is well defined, which are Assumption 1(2)–(3). We can deduce that for $j = 1, \ldots, i$,

$$\frac{\partial \boldsymbol{H}_{i,1}}{\partial \lambda_j} = \frac{\partial \boldsymbol{g}_i}{\partial \lambda_j} - (\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}(\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}\frac{\partial \boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}}{\partial \lambda_j}(\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}\boldsymbol{F}_{i-1}$$
$$- (\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}\left(\boldsymbol{F}_{i-1,\boldsymbol{\lambda}_{i-1}}\frac{\partial \boldsymbol{g}_i}{\partial \lambda_j} + \frac{\partial \boldsymbol{F}_{i-1}}{\partial \lambda_i}\frac{\partial H_{i,2}}{\partial \lambda_j}\right), \quad (21)$$

$$\frac{\partial H_{i,2}}{\partial \lambda_j} = \frac{\partial \lambda_i}{\partial \lambda_j} - \frac{\partial}{\partial \lambda_j}\left(\frac{\partial F_i}{\partial \lambda_i}\right)^{-1} F_i - \left(\frac{\partial F_i}{\partial \lambda_i}\right)^{-1}\frac{\partial F_i}{\partial \lambda_j}. \quad (22)$$

Evaluating these two equations at $\boldsymbol{\mu}^{(i)}$ and using the fact that $\boldsymbol{F}^*_i := \boldsymbol{F}_i(\boldsymbol{\mu}^{(i)}) = \boldsymbol{0}$, the second terms in the right-hand-side of (21)-(22) vanish and we have

$$\frac{\partial \boldsymbol{H}^*_{i,1}}{\partial \lambda_j} = \frac{\partial \boldsymbol{g}^*_i}{\partial \lambda_j} - (\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}\left(\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}}\frac{\partial \boldsymbol{g}^*_i}{\partial \lambda_j} + \frac{\partial \boldsymbol{F}^*_{i-1}}{\partial \lambda_i}\frac{\partial H^*_{i,2}}{\partial \lambda_j}\right)$$
$$= -(\boldsymbol{F}^*_{i-1,\boldsymbol{\lambda}_{i-1}})^{-1}\left(\frac{\partial \boldsymbol{F}^*_{i-1}}{\partial \lambda_i}\frac{\partial H^*_{i,2}}{\partial \lambda_j}\right),$$
$$\frac{\partial H^*_{i,2}}{\partial \lambda_j} = \delta_{j,i} - \left(\frac{\partial F^*_i}{\partial \lambda_i}\right)^{-1}\frac{\partial F^*_i}{\partial \lambda_j}.$$

where $\delta_{j,i}$ is one only if $j = i$ and zero otherwise.

Acknowledgment

We thank Rafail Abramov for supplying the two-dimensional density data set for Example 4. The BFGS code that we used for comparison in Example 4 was downloaded from [2].

References

[1] R. V. Abramov, *An improved algorithm for the multidimensional moment-constrained maximum entropy problem*, J. Comput. Phys. **226** (2007), no. 1, 621–644. MR Zbl

[2] ———, *The multidimensional moment-constrained maximum entropy algorithm*, 2007. Zbl

[3] ———, *The multidimensional moment-constrained maximum entropy problem: a BFGS algorithm with constraint scaling*, J. Comput. Phys. **228** (2009), no. 1, 96–108. MR Zbl

[4] ———, *The multidimensional maximum entropy moment problem: a review on numerical methods*, Commun. Math. Sci. **8** (2010), no. 2, 377–392. MR Zbl

[5] R. V. Abramov, A. Majda, and R. Kleeman, *Information theory and predictability for low-frequency variability*, J. Atmospheric Sci. **62** (2005), no. 1, 65–87. MR

[6] T. Ahrendt, *Fast computations of the exponential function*, STACS 99 (Trier) (C. Meinel and S. Tison, eds.), Lecture Notes in Comput. Sci., no. 1563, Springer, 1999, pp. 302–312. MR Zbl

[7] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler, *Numerically solving polynomial systems with Bertini*, Software, Environments, and Tools, no. 25, Society for Industrial and Applied Mathematics, 2013. MR Zbl

[8] M. Frontini and A. Tagliani, *Maximum entropy in the finite Stieltjes and Hamburger moment problem*, J. Math. Phys. **35** (1994), no. 12, 6748–6756. MR Zbl

[9] T. Gerstner and M. Griebel, *Numerical integration using sparse grids*, Numer. Algorithms **18** (1998), no. 3–4, 209–232. MR Zbl

[10] W. Hao and J. Harlim, *Supplementary material: MATLAB software for the equation-by-equation method for solving the maximum entropy problem*, 2018.

[11] ———, *Supplementary movie for the equation-by-equation method for solving the maximum entropy problem*, 2018.

[12] W. Hao, J. D. Hauenstein, B. Hu, Y. Liu, A. J. Sommese, and Y.-T. Zhang, *Continuation along bifurcation branches for a tumor model with a necrotic core*, J. Sci. Comput. **53** (2012), no. 2, 395–413. MR Zbl

[13] K. Haven, A. Majda, and R. V. Abramov, *Quantifying predictability through information theory: small sample estimation in a non-Gaussian framework*, J. Comput. Phys. **206** (2005), no. 1, 334–362. MR Zbl

[14] E. T. Jaynes, *Information theory and statistical mechanics*, Phys. Rev. (2) **106** (1957), 620–630. MR Zbl

[15] A.-K. Kassam and L. N. Trefethen, *Fourth-order time-stepping for stiff PDEs*, SIAM J. Sci. Comput. **26** (2005), no. 4, 1214–1233. MR Zbl

[16] A. Leykin, J. Verschelde, and A. Zhao, *Newton's method with deflation for isolated singularities of polynomial systems*, Theoret. Comput. Sci. **359** (2006), no. 1–3, 111–122. MR Zbl

[17] MathWorks, MATLAB *fsolve.m*, 2017.

[18] J. D. McCalpin, *The statistics and sensitivity of a double-gyre model: the reduced-gravity, quasigeostrophic case*, J. Phys. Oceanogr. **25** (1995), no. 5, 806–824.

[19] J. D. McCalpin and D. B. Haidvogel, *Phenomenology of the low-frequency variability in a reduced-gravity, quasigeostrophic double-gyre model*, J. Phys. Oceanogr. **26** (1996), no. 5, 739–752.

[20] L. R. Mead and N. Papanicolaou, *Maximum entropy in the problem of moments*, J. Math. Phys. **25** (1984), no. 8, 2404–2417. MR

[21] E. Parzen, *On estimation of a probability density function and mode*, Ann. Math. Statist. **33** (1962), 1065–1076. MR Zbl

[22] M. Rosenblatt, *Remarks on some nonparametric estimates of a density function*, Ann. Math. Statist. **27** (1956), 832–837. MR Zbl

[23] S. A. Smolyak, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Dokl. Akad. Nauk SSSR **148** (1963), no. 5, 1042–1045, In Russian; translated in Soviet Math. Dokl. **4** (1963), 240–243. MR Zbl

[24] A. J. Sommese and C. W. Wampler, II, *The numerical solution of systems of polynomials: arising in engineering and science*, World Scientific, 2005. MR

[25] L. N. Trefethen, *Is Gauss quadrature better than Clenshaw–Curtis?*, SIAM Rev. **50** (2008), no. 1, 67–87. MR Zbl

[26] X. Wu, *Calculation of maximum entropy densities with application to income distribution*, J. Econometrics **115** (2003), no. 2, 347–354. MR Zbl

[27] Z. Wu, G. N. Phillips, Jr., R. Tapia, and Y. Zhang, *A fast Newton algorithm for entropy maximization in phase determination*, SIAM Rev. **43** (2001), no. 4, 623–642. MR Zbl

Received July 4, 2017. Revised January 17, 2018.

WENRUI HAO: wxh64@psu.edu
Department of Mathematics, The Pennsylvania State University, University Park, PA, United States

JOHN HARLIM: jharlim@psu.edu
Department of Mathematics, Department of Meteorology and Atmospheric Science, The Pennsylvania State University, University Park, PA, United States

Communications in Applied Mathematics and Computational Science

msp.org/camcos

See inside back cover or msp.org/camcos for submission instructions.

The subscription price for 2018 is US $100/year for the electronic version, and $150/year (+$15, if shipping outside the US) for print and electronic. Subscriptions, requests for back issues from the last three years and changes of subscriber address should be sent to MSP.

Communications in Applied Mathematics and Computational Science (ISSN 2157-5452 electronic, 1559-3940 printed) at Mathematical Sciences Publishers, 798 Evans Hall #3840, c/o University of California, Berkeley, CA 94720-3840, is published continuously online. Periodical rate postage paid at Berkeley, CA 94704, and additional mailing offices.

CAMCoS peer review and production are managed by EditFLOW® from MSP.

PUBLISHED BY

mathematical sciences publishers

nonprofit scientific publishing

http://msp.org/

Communications in Applied Mathematics and Computational Science

vol. 13 no. 2 2018