Pesquisa Operacional (2019) 39(3): 361-378

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
doi: 10.1590/0101-7438.2019.039.03.0361

THE EM ALGORITHM FOR STANDARD STOCHASTIC FRONTIER MODELS

Bernardo B. de Andrade[1*] and Geraldo S. Souza[2]

Received November 28, 2018 / Accepted September 11, 2019

ABSTRACT. The Expectation-Maximization (EM) algorithm is developed for the stochastic frontier models most used in practice with cross-section data. The resulting algorithms can be easily programmed into a computer and are shown to be worthy alternatives to general-purpose optimization routines currently used. The algorithms for the half normal and the exponential models have closed-form expressions whereas those for the truncated normal and gamma models will require the numerical solution of a nonlinear equation. Implementations of the EM algorithm either as a stand-alone routine or in accelerated form and also combined with Newton-like methods are discussed. We provide illustrations, along with R tools, for cost and production frontiers.

Keywords: efficiency, EM acceleration, gamma, maximum likelihood.

1 INTRODUCTION

In this paper we consider the Expectation-Maximization (EM) algorithm (Dempster et al., 1977; Wu, 1983) for doing stochastic frontier (SF) analysis, a technique for measuring economic efficiency. In statistical terminology, an SF model is a linear mixed-effects model, making it natural to consider the EM algorithm for maximum likelihood (ML) estimation.

The regression part of SF models represents the *production frontier* of the i-th firm: the response y is (possibly) some transformation of measured output and $\boldsymbol{x}$ is a vector of (possibly transformed) inputs. The relation between input and outputs is *stochastic*: the error v is the conventional Gaussian noise, $v_i \sim \mathcal{N}(0, \sigma_v^2)$, representing stochastic elements outside the control of the firm. Each firm possesses an inefficiency term $u_i \geq 0$ which makes the observed output smaller than its (stochastic) potential. Thus,

$$y_i = \boldsymbol{x}_i'\boldsymbol{\beta} + v_i - u_i. \tag{1}$$

We assume cross-sectional data for which the v_i's and the u_i's are independent of each other and across observations.

*Corresponding author – https://orcid.org/0000-0003-4688-9733
[1]Departamento de Estatística, Universidade de Brasília, DF, Brazil
[2]Empresa Brasileira de Pesquisa Agropecuária (Embrapa/Sede), Brasília, DF, Brazil
E-mails: bbandrade@unb.br, geraldo.souza@embrapa.br

In most applications, u is assumed to be either half-normally or exponentially distributed, $u_i \sim \mathscr{N}^+(0,\sigma_u^2)$ or $u_i \sim \mathscr{E}xp(\lambda)$, respectively. These canonical specifications for u have been generalized by letting $u_i \sim \mathscr{N}^+(\mu,\sigma_u^2)$ (truncated-normal) and $u_i \sim \mathscr{G}(\alpha,\lambda)$ (Gamma). These standard models are detailed in the comprehensive work of Kumbhakar & Lovell (2003).

The estimation method of choice for SF analysis is ML but variations of least squares, nonparametric and Bayesian estimation have also been developed and are surveyed by Greene (2008). SF models were preceded by (non-stochastic) frontier models whose estimates were computed by linear and quadratic programming (Aigner & Chu, 1968). Current software for SF analysis inspected by the authors (Bogetoft & Otto, 2015; Coelli & Henningsen, 2013; SAS Institute Inc., 2003; StataCorp., 2015; Greene, 2012) has been written around Newton-like routines based on either user-supplied derivatives or numerical approximations. Strictly speaking, the ML problem of SF analysis is a constrained optimization problem since there are parameters restricted to the positive axis. With luck and good starting values this may be of no practical effect in a given application but the usual practice with Newton-like methods is to reparameterize the model and retrieve standard errors by means of the delta method. EM updates will keep the iterates for the positive parameters positive and in our experience with SF estimation EM seems to work with more liberal starting values.

The use of EM in SF estimation is not new. Instead of using Newton, Greene (1982) devised a simple iterative scheme by manipulating the likelihood equations of the half normal SF model. This method was later improved by Lee (1983) who showed that Greene's scheme did not necessarily solve the likelihood equations and, in addition, did not restrict variance estimates to be positive. Lee devised a new scheme based on solving a set of rewritten equations which made explicit use of the conditional expectations $\mathrm{E}(u|v+u)$ and $\mathrm{E}(u^2|v+u)$. The new scheme actually solved the likelihood equations and preserved positivity of variance estimates. Lee did not mention that his scheme was in fact the EM algorithm for the half normal SF model. Huang (1984) devised the same algorithm, this time explicitly invoking EM, apparently independently of Lee. In Section 3.1 we will revisit (with different notation) the EM algorithm devised by Lee (1983) and Huang (1984) for the half normal model. In later sections we develop the EM algorithm for the other three standard models.

EM does not directly yield standard errors of estimates. Lee was not explicit about calculation of standard errors and Huang used least-squares errors for the regression coefficients. We will use more general methods (Section 3.5). Two illustrations are given in Section 4: estimation of a production frontier with the half normal model using data from the Brazilian Census of Agriculture and estimation of a cost function with exponential and gamma inefficiencies. Concluding remarks and some practical considerations are given in Section 5.

2 STOCHASTIC FRONTIER MODELS IN EM FORM

The SF model (1) can be written as a hierarchy in terms of the total error $\varepsilon = y - \boldsymbol{x}'\boldsymbol{\beta}$ and the inefficiencies u,

$$\begin{aligned} \varepsilon_i | u_i &\sim \mathcal{N}(-u_i, \sigma_v^2), \\ u_i &\sim f_u, \end{aligned}$$

where f_u denotes the density for the inefficiency term. At times, we will make the dependence of ε on $\boldsymbol{\beta}$ explicit by writing $\varepsilon(\boldsymbol{\beta})$. The resulting joint density is

$$f_{\boldsymbol{\theta}}(u, \varepsilon) = f_u(u) \frac{1}{\sigma_v \sqrt{2\pi}} \exp\left[-\frac{(\varepsilon(\boldsymbol{\beta}) + u)^2}{2\sigma_v^2} \right],$$

where the vector $\boldsymbol{\theta}$ comprises $\boldsymbol{\beta}$, σ_v^2 and any parameters from f_u.

The loglikelihood of a set of n independent observations is thus given by

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \int_0^\infty f_{\boldsymbol{\theta}}(u_i, \varepsilon_i) du_i, \tag{2}$$

In order to maximize the likelihood, instead of directly using (2), the EM algorithm uses the *complete-data* likelihood (Dempster et al., 1977; Wu, 1983),

$$\ell_c(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log f_{\boldsymbol{\theta}}(u_i, \varepsilon_i). \tag{3}$$

The iterations of the algorithm result from two steps: Let $\boldsymbol{\theta}_t$ be the estimate at iteration t of the algorithm and let $\mathrm{E}_t(\cdot) \equiv \mathrm{E}_{\boldsymbol{\theta}_t}(\cdot)$. Then, the E-step is the evaluation of

$$\mathrm{E}_t(\ell_c(\boldsymbol{\theta}) | \boldsymbol{\varepsilon}) = \int f_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{\varepsilon}) f_{\boldsymbol{\theta}_t}(\boldsymbol{u} | \boldsymbol{\varepsilon}) d\boldsymbol{u}, \tag{4}$$

and the M-step is the maximization, with respect to $\boldsymbol{\theta}$, of the function

$$Q(\boldsymbol{\theta} | \boldsymbol{\theta}_t) = \mathrm{E}_t(\ell_c(\boldsymbol{\theta}) | \boldsymbol{\varepsilon}). \tag{5}$$

It has been shown that the sequence generated by iteratively solving (4) and (5) converges to a stationary point of ℓ. Furthermore, under regularity conditions (Wu, 1983), the ascent property holds: upward movements in Q imply upward movements in ℓ,

$$\ell(\boldsymbol{\theta}_{t+1}) - \ell(\boldsymbol{\theta}_t) \geq Q(\boldsymbol{\theta}_{t+1} | \boldsymbol{\theta}_t) - Q(\boldsymbol{\theta}_t | \boldsymbol{\theta}_t).$$

In order to present the steps of the EM algorithm in the context of SF models we start by denoting the first two conditional moments of $(u|\varepsilon)$ by

$$M_i(\boldsymbol{\theta}) = \mathrm{E}_{\boldsymbol{\theta}}(u_i | \varepsilon_i) \quad \text{and} \quad K_i(\boldsymbol{\theta}) = \mathrm{E}_{\boldsymbol{\theta}}(u_i^2 | \varepsilon_i). \tag{6}$$

Making explicit the dependence of the total error on the coefficients, $\varepsilon_i(\boldsymbol{\beta})$, we define

$$\begin{aligned} R_i(\boldsymbol{\theta}_t, \boldsymbol{\beta}) &= \mathrm{E}_t\left[(u_i + \varepsilon_i(\boldsymbol{\beta}))^2 | \varepsilon_i\right] \\ &= K_i(\boldsymbol{\theta}_t) + 2\varepsilon_i(\boldsymbol{\beta}) M_i(\boldsymbol{\theta}_t) + \varepsilon_i^2(\boldsymbol{\beta}). \end{aligned} \tag{7}$$

Note that it is crucial to distinguish $\boldsymbol{\beta}$ inside ε from $\boldsymbol{\beta}_t$ as part of $\boldsymbol{\theta}_t$.

Now let

$$\overline{M}_t = \frac{1}{n}\sum_i M_i(\boldsymbol{\theta}_t), \qquad \overline{K}_t = \frac{1}{n}\sum_i K_i(\boldsymbol{\theta}_t), \tag{8}$$

and

$$\overline{R}_t(\boldsymbol{\beta}) = \frac{1}{n}\sum_i R_i(\boldsymbol{\theta}_t, \boldsymbol{\beta}). \tag{9}$$

Finally, we define the "working" response w,

$$w_i(\boldsymbol{\theta}_t) = y_i + M_i(\boldsymbol{\theta}_t). \tag{10}$$

The above definitions will be used in the expressions for the Q function and its derivatives in the following sections. We close this section by noting that the integrals in the E-step can be carried out explicitly except in the gamma model (Section 3.4).

3 EM ALGORITHMS FOR THE STANDARD SF MODELS

The original models used in SF analysis were the half normal and exponential models, the former being the default option in most software for SF analysis. EM is developed for these two models in Sections 3.1 and 3.2 resulting in extremely simple algorithms with closed-form expressions and intuitive iterative schemes. The econometric literature on SF models has considered generalizations of those two models to allow for a nonzero mode for u leading to the truncated normal (Section 3.3) and gamma (Section 3.4) SF models. Another direction of generalization is to make some (or all) of the parameters in the distribution of the noise dependent on covariates. In any case, the resulting EM algorithm will not render explicit updates as with the simple half normal and exponential specifications.

3.1 Half Normal Model

We now consider implementation of the EM scheme with the half normal model, $u_i \sim \mathcal{N}^+(0, \sigma_u^2)$, that is, for $u \geq 0$,

$$f_u(u) = \frac{1}{2\sigma_u}\varphi\left(\frac{u}{\sigma_u}\right),$$

where φ denotes the standard normal density. The EM algorithm for the half normal model was introduced by Huang (1984) but Lee (1983) arrived at the same algorithm from a different standpoint.

The Q function, after some simplifications using expressions (6)–(9), becomes (constant term omitted)

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}_t) = -n\left[\log(\sigma_u\sigma_v) + \frac{1}{2\sigma_u^2}\overline{K}_t + \frac{1}{2\sigma_v^2}\overline{R}_t(\boldsymbol{\beta})\right],$$

and thus,

$$\begin{aligned}
\frac{\partial Q}{\partial \boldsymbol{\beta}} = 0 &\Leftrightarrow \sum_i \boldsymbol{x}_i\left[M_i(\boldsymbol{\theta}_t) + \varepsilon_i(\boldsymbol{\beta})\right] = 0,\\
\frac{\partial Q}{\partial \sigma_v^2} = 0 &\Leftrightarrow \sigma_v^2 = \overline{R}_t(\boldsymbol{\beta}),\\
\frac{\partial Q}{\partial \sigma_u^2} = 0 &\Leftrightarrow \sigma_u^2 = \overline{K}_t.
\end{aligned}$$

We finally obtain the iterative scheme in Algorithm A below with w given by (10).

Algorithm A: EM for the half-normal SF model

Initialize with $(\boldsymbol{\beta}_0, \sigma^2_{v,0}, \sigma^2_{u,0})$.
Iterate:

A1. $\boldsymbol{\beta}_{t+1} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{w}_t, \quad \boldsymbol{w}_t = (w_1(\boldsymbol{\theta}_t), \ldots, w_n(\boldsymbol{\theta}_t))$.

A2. $\sigma^2_{v,t+1} = \overline{R}_t(\boldsymbol{\beta}_{t+1})$.

A3. $\sigma^2_{u,t+1} = \overline{K}_t$.

Stop if RC $\leq \zeta$.

(initialization and relative change (RC) criterion are illustrated in Section 4)

Some remarks are due. All of the conditional expectations in the calculations above are available in closed form since the conditional distribution of u given ε is truncated normal (Kumbhakar & Lovell, 2003, Chapter 3),

$$(u_i|\varepsilon_i) \sim \mathscr{N}^+(\widetilde{\mu}_i, b^2) \quad \text{with} \quad \widetilde{\mu}_i = \frac{-\varepsilon_i\sigma_u^2}{\sigma_u^2+\sigma_v^2}, \quad b^2 = \frac{\sigma_v^2\sigma_u^2}{\sigma_u^2+\sigma_v^2}.$$

Therefore, the conditional moments in (6) are given by

$$M_i(\boldsymbol{\theta}) = \widetilde{\mu}_i + b\delta_i$$

and

$$K_i(\boldsymbol{\theta}) = M_i^2(\boldsymbol{\theta}) + b^2[1 - \delta_i(\delta_i + \widetilde{\mu}_i/b)]$$

where $\delta_i = \dfrac{\varphi(-\widetilde{\mu}_i/b)}{1-\Phi(-\widetilde{\mu}_i/b)}$ and Φ is the cumulative distribution function of the standard normal density φ.

Note that Algorithm A has remarkably simple and intuitive updates. The regression coefficients are updated in A1 by OLS with the working response (10), which is a version of $y+\mathrm{E}(u|\varepsilon)$, a sort of "correction" for the presence of u in the overall error $\varepsilon = v-u$. In A2 the noise variance is updated as a mean square based on the squared "residuals" $(u+\varepsilon)^2$ given by (7). The inefficiency variance is updated last by the mean of the K_i's. Since K_i is the conditional mean of u_i^2 and its model mean is zero, the variance update A3 is simply a version of $\mathrm{Var}(u|\varepsilon)$.

3.2 Exponential Model

We now extend the EM algorithm to the other models starting with the exponential SF model. The notation developed in Section 2 and used in Section 3.1 remains useful. The inefficiencies are assumed to have an exponential density with mean $1/\lambda$.

Similar to Section 3.1, using definitions (6)–(9), the Q function for the exponential model is given by

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}_t) = n\left[\log\left(\frac{\lambda}{\sigma_v}\right) - \lambda\overline{M}_t - \frac{1}{2\sigma_v^2}\overline{R}_t(\boldsymbol{\beta})\right].$$

As before, the EM iterations are obtained by setting $\frac{\partial Q}{\partial \boldsymbol{\theta}} = \mathbf{0}$, resulting in Algorithm B below.

Algorithm B: EM for the exponential SF model

Initialize with $(\boldsymbol{\beta}_0, \sigma^2_{v,0}, \lambda_0)$.
Iterate:

1. $\boldsymbol{\beta}_{t+1} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{w}_t, \quad \boldsymbol{w}_t = (w_1(\boldsymbol{\theta}_t), \ldots, w_n(\boldsymbol{\theta}_t))$
2. $\sigma^2_{v,t+1} = \overline{R}_t(\boldsymbol{\beta}_{t+1})$
3. $\lambda_{t+1} = \overline{M}_t^{-1}$

Stop if RC $\leq \zeta$.

(initialization and relative change (RC) criterion are illustrated in Section 4)

Remarks similar to those for the half normal model can be made. Again, all expectations involved are known analytically since the distribution of $(u|\varepsilon)$ is also truncated normal (Kumbhakar & Lovell, 2003, Chapter 3),

$$(u_i|\varepsilon_i) \sim \mathscr{N}^+(\widetilde{\mu}_i, \sigma_v^2), \quad \widetilde{\mu}_i = -(\varepsilon_i + \lambda\sigma_v^2).$$

Therefore, the conditional moments in (6) are given by

$$M_i(\boldsymbol{\theta}) = \widetilde{\mu}_i + \sigma_v\delta_i$$

and

$$K_i(\boldsymbol{\theta}) = M_i^2(\boldsymbol{\theta}) + \sigma_v^2[1 - \delta_i(\delta_i + \widetilde{\mu}_i/\sigma_v)]$$

where $\delta_i = \dfrac{\varphi(-\widetilde{\mu}_i/\sigma_v)}{1-\Phi(-\widetilde{\mu}_i/\sigma_v)}$.

The updates in Algorithm B mirror those of the half normal case. B1 and B2 are analogous to A1 and A2 (allowing, of course, for different expressions for M_i and K_i for each model). In B3, λ is updated by the inverse of the estimated mean of the inefficiencies, $\overline{M}^{-1}$, which is the conditional version of $\mathrm{E}^{-1}(u|\varepsilon)$.

3.3 Truncated Normal Model

We now present the EM scheme with the truncated normal model, $u_i \sim \mathscr{N}^+(\mu, \sigma_u^2)$, that is, for $u \geq 0$,

$$f_u(u) = \frac{\frac{1}{\sigma_u}\varphi\left(\frac{u-\mu}{\sigma_u}\right)}{\Phi\left(\frac{\mu}{\sigma_u}\right)}.$$

The Q function is, after some simplifications using (6)–(9),

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}_t) = -n\left[\log(\sigma_u\sigma_v) + \log\left(\Phi\left(\frac{-\mu}{\sigma_u}\right)\right) + \right.$$
$$\left. + \frac{1}{2\sigma_u^2}\left(\overline{K}_t - 2\mu\overline{M}_t + \frac{\mu^2}{n}\right) + \frac{1}{2\sigma_v^2}\overline{R}_t(\boldsymbol{\beta})\right].$$

After setting $\partial Q/\partial\boldsymbol{\theta}$ to zero, we arrive at the iterative scheme in Algorithm C below, using, once again, definitions (6)–(10).

Algorithm C: EM for the truncated normal SF model

Initialize with $(\boldsymbol{\beta}_0, \sigma^2_{v,0}, \mu_0, \sigma^2_{u,0})$.
Iterate:

1. $\boldsymbol{\beta}_{t+1} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{w}_t, \quad \boldsymbol{w}_t = (w_1(\boldsymbol{\theta}_t), \ldots, w_n(\boldsymbol{\theta}_t))$
2. $\sigma^2_{v,t+1} = \overline{R}_t(\boldsymbol{\beta}_{t+1})$
3. $\mu_{t+1} = m$ satisfying
$$\begin{cases} m < \dfrac{\overline{K}_t}{\overline{M}_t} \\ \Phi\left(\frac{-m}{s_t}\right)(\overline{M}_t - m) = \varphi\left(\frac{-m}{s_t}\right)\sqrt{\overline{K}_t - m\overline{M}_t} \end{cases}$$
4. $\sigma^2_{u,t+1} = \overline{K}_t - \mu_{t+1}\overline{M}_t$

Stop if RC $\leq \zeta$.

(initialization and relative change (RC) criterion are illustrated in Section 4)

As before, all conditional moments involved in Algorithm C have simple closed forms since (Kumbhakar & Lovell, 2003, Chapter 3),

$$(u_i|\varepsilon_i) \sim \mathscr{N}^+(\widetilde{\mu}_i, b^2), \quad \widetilde{\mu}_i = \frac{\mu\sigma^2_v - \varepsilon_i\sigma^2_u}{\sigma^2_u + \sigma^2_v}, \quad b^2 = \frac{\sigma^2_v\sigma^2_u}{\sigma^2_u + \sigma^2_v},$$

which leads to

$$M_i(\boldsymbol{\theta}) = \widetilde{\mu}_i + b\delta_i,$$

and

$$K_i(\boldsymbol{\theta}) = M^2_i(\boldsymbol{\theta}) + b^2[1 - \delta_i(\delta_i + \widetilde{\mu}_i/b)],$$

where $\delta_i = \dfrac{\varphi(-\widetilde{\mu}_i/b)}{1 - \Phi(-\widetilde{\mu}_i/b)}$.

The updates C1 and C2 are analogous to steps A1 and A2 in Algorithm A. Iteration C3 does not give an explicit update for μ since it requires the solution of a nonlinear equation with an upper bound which guarantees positivity in C4. The inefficiency variance is updated last by a term of the form $K - \mu M$ which is a version of $\mathrm{E}(u^2|\varepsilon) - \mathrm{E}^2(u|\varepsilon)$ with μM in the place of M^2.

3.4 Gamma Model

In the gamma model the inefficiencies are assumed to have a gamma density with mean α/λ. We will adopt the same definitions used so far in expressions (6)–(10) and in addition we define

$$L_i(\boldsymbol{\theta}) = \mathrm{E}_{\boldsymbol{\theta}}(\log(u_i)|\varepsilon_i) \tag{11}$$

and

$$\bar{L}_t = \frac{1}{n}\sum_i L_i(\boldsymbol{\theta}_t). \tag{12}$$

The Q function is thus (constant term omitted)

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}_t) = n\left[\log\left(\frac{\lambda^\alpha}{\Gamma(\alpha)\sigma_v}\right) + (\alpha-1)\bar{L}_t - \lambda\overline{M}_t - \frac{1}{2\sigma_v^2}\overline{R}_t(\boldsymbol{\beta})\right].$$

As opposed to the previous models, the conditional moments involved in the Q function for the gamma SF model do not have simple closed-form expressions. The conditional density $f(u|\varepsilon)$ has an intractable normalizing constant (Kumbhakar & Lovell, 2003, Chapter 3) which affects not only EM calculations but also the loglikelihood and its derivatives. More specifically, recalling (6), we have for the gamma model,

$$M_i(\boldsymbol{\theta}) = \frac{h_i(\alpha)}{h_i(\alpha-1)} \quad \text{and} \quad K_i(\boldsymbol{\theta}) = \frac{h_i(\alpha+1)}{h_i(\alpha-1)}$$

where

$$h_i(r) = \int_0^\infty u^r \varphi_i(u)du, \tag{13}$$

and $\varphi_i(\cdot)$ is the density function of a normal random variable with mean $-(\varepsilon_i + \lambda\sigma_v^2)$ and variance σ_v^2.

Evaluation of (13) was initially addressed by simulation-based methods (Greene, 2003; Kozumi & Zhang, 2005) but, recently, we showed that quadrature and Fourier methods can be used with higher accuracy (Andrade & Souza, 2018) and this is adopted in the implementations of Section 4. Similarly, $L_i = \int_0^\infty \log(u)\varphi_i(u)du$ may be computed via numerical quadrature.

Setting $\partial Q/\partial\boldsymbol{\theta}$ to zero yields Algorithm D below where we make use of the digamma function $\psi(a) = \Gamma'(a)/\Gamma(a)$.

Algorithm D: EM for the gamma SF model

Initialize with $(\boldsymbol{\beta}_0, \sigma^2_{v,0}, \alpha_0, \lambda_0)$.
Iterate:

1. $\boldsymbol{\beta}_{t+1} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{w}_t, \quad \boldsymbol{w}_t = (w_1(\boldsymbol{\theta}_t), \ldots, w_n(\boldsymbol{\theta}_t))$
2. $\sigma^2_{v,t+1} = \overline{R}_t(\boldsymbol{\beta}_{t+1})$
3. $\lambda_{t+1} = m$ satisfying
$$\log(m) + \bar{L}_t = \psi(m\overline{M}_t), \quad m > 0$$
4. $\alpha_{t+1} = \lambda_{t+1}\overline{M}_t$

Stop if RC $\leq \zeta$.
(initialization and relative change (RC) criterion are illustrated in Section 4)

Similar to the truncated normal model, Algorithm D requires the solution of a nonlinear equation in step D3. All other updates are straightforward. We also note that Algorithms A–D have analogous updates for $\boldsymbol{\beta}$ and σ^2_v. The differences occur only in the updates of the parameters related to the distribution of u, steps 3 and 4 in Algorithms A–D.

3.5 Standard Errors and Hybrid EM

EM does not directly yield standard errors. In addition, EM can be extremely slow to converge. Acceleration schemes have been proposed alongside ways of estimating standard errors (Jamshidian & Jennrich, 1997). We will call these methods *hybrid EM* since in order to obtain standard errors, the likelihood (not necessary for the updates) must be provided.

In the illustrations that follow, we have used different EM implementations:

I. Straightforward implementation of the EM updates (Algorithms A, B and C) with the variance of ML estimates, $\hat{\boldsymbol{\theta}}$, obtained with the outer product of gradients (OPG),
$$\widehat{\text{Var}}(\hat{\boldsymbol{\theta}}) = (\hat{G}'\hat{G})^{-1}, \tag{14}$$
where $\hat{G}$ is the $n \times p$ matrix of derivatives of the loglikelihood, $\hat{G}_{ij} = [\partial \ell_i / \partial \theta_j]|_{\hat{\boldsymbol{\theta}}}$. Here, derivatives were computed via numerical differentiation of the loglikelihood (2). Explicit computation of $\partial \ell_i / \partial \theta_j$ is possible for the half-normal, exponential and truncated normal SF models and may also be used to obtain $\hat{G}$ instead of numerical differentiation.

II. Hybridization schemes:

(II.1) Simple follow-up of EM by any Newton-like method: Running EM in the beginning then switching to a quasi-Newton method is a well established strategy (McLachlan

& Krishnan, 2007). The idea is to take advantage of the global convergence properties of EM at first and then bypass EM's slow convergence by switching to a Newton-like method such as DFP or BFGS, which features fast, superlinear, local convergence (Dennis & Moré, 1974). Such a hybrid algorithm is possible as long as the the log-likelihood (2) is available (at least numerically). In addition, regardless of whether the chosen Newton method uses user-supplied derivatives or some built-in numerical derivative routine, a Hessian matrix evaluated at the optimum, $\hat{H}$, will be generated at the final iteration which is then used for variance estimation,

$$\widehat{\mathrm{Var}}(\hat{\boldsymbol{\theta}}) = -\hat{H}^{-1}. \tag{15}$$

(II.2) Squared extrapolation method: Next we consider a hybridization scheme known as SQUAREM (Varadhan & Roland, 2008). It belongs to the non-monotone (partially monotone) class of acceleration schemes (Varadhan & Roland, 2008; Zhou et al., 2011) and along with the more elaborate scheme of Zhou et al. (2011) is considered the gold standard among acceleration schemes (Zhou et al., 2011). The general form of this accelerated EM algorithm is given is pseudo code form below. Details about the calculation of the steplength are given in Varadhan and Roland. In the R implementation we used (Bobb & Varadhan, 2018), standard errors are obtained by a numerically computed hessian if the loglikelihood is provided.

Algorithm SQUAREM: Pseudo code (Varadhan & Roland, 2008)

Iterate:

i. $\boldsymbol{\theta}_a$ = EMupdate($\boldsymbol{\theta}_t$)

ii. $\boldsymbol{\theta}_b$ = EMupdate($\boldsymbol{\theta}_a$)

iii. $\boldsymbol{r} = \boldsymbol{\theta}_a - \boldsymbol{\theta}_t$

iv. $\boldsymbol{s} = \boldsymbol{\theta}_b - \boldsymbol{\theta}_a - \boldsymbol{r}$

v. Compute steplength α

vi. $\boldsymbol{\theta}_c = \boldsymbol{\theta}_t - 2\alpha\boldsymbol{r} + \alpha^2\boldsymbol{s}$

vii. $\boldsymbol{\theta}_{t+1}$ = EMupdate($\boldsymbol{\theta}_c$)

viii. Stop if RC $\leq \zeta$.

(initialization and relative change (RC) criterion are illustrated in Section 4)

(II.3) Quasi-newton acceleration of EM: the method developed by Zhou et al. (2011) surpasses previous quasi-Newton acceleration methods in that it does not handle the observed information matrix or the Hessian of the algorithm map (`EMupdate`), a

property shared with the above squared extrapolation method. The general form of the update is given by

$$\boldsymbol{\theta}_{t+1} = \texttt{EMupdate}(\boldsymbol{\theta}_t) - \Delta[\boldsymbol{\theta}_t - \texttt{EMupdate}(\boldsymbol{\theta}_t)].$$

The matrix Δ is related to a low-rank approximation of the differential of the EM update. Its construction is detailed in Zhou et al. (2011).

In both SQUAREM and quasi-Newton acceleration the Hessian of the loglikelihood evaluated at the estimate is obtained numerically thus providing standard errors, according to (15). Note that the likelihood is not needed in the updates, just for Hessian computation.

4 ILLUSTRATIONS

In the following illustrations we have monitored the relative change in estimates,

$$\text{RC} = \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}\|}{\zeta + \|\boldsymbol{\theta}_{t-1}\|},$$

with stopping rule $\text{RC} < \zeta$. We have set $\zeta = 10^{-6}$. The hybrid scheme (II.1) runs EM until RC reaches a less stringent level (say $\zeta = 10^{-3}$), followed by a built-in BFGS routine (which computes the necessary derivatives of the loglikelihood numerically).

Starting values for all schemes are obtained by adapting ordinary least squares (OLS) and the results for half normal and exponential models were checked against those provided by Stata14's `rfrontier` StataCorp. (2015) which uses a Newton routine.

Versioned code in R language for the applications below are available from the authors' website as illustrated in the Appendix.

4.1 Agricultural Production Frontier

As a first illustration, we use data from the Brazilian Census of Agriculture (2006) (IBGE, 2006)[1] including 219 municipalities in the Brazilian Midwest region to fit the half normal and exponential models using EM. The specific technology considered is Cobb-Douglas,

$$\log(Y_i) = \beta_0 + \beta_1 \log(K_i) + \beta_2 \log(L_i) + v_i - u_i,$$

where Y is gross output of farms, ranches and agricultural enterprises in the municipality and K and L are aggregate measures of capital and labor, respectively.

For the half normal model, a combination of method of moments and OLS, known as modified OLS (MOLS), is available (with closed-form expressions) (Kumbhakar & Lovell, 2003) and the parameter estimates could be used as (excellent) starting values for likelihood maximization.

[1] The latest available census is from 2017, followed by the one in 2006, but we, as of now, have not been able to compile the 2017 microdata.

A rougher starting point is simply $\boldsymbol{\beta}_0 =$ OLS estimates and $\sigma^2_{v,0} = \sigma^2_{u,0} =$ half of the residual variance from OLS. As expected, with either of these sets of starting values we obtained convergence with different generic optimization routines available in R and Stata14. EM converged in about 370 iterations when started from MOLS and it took close to 600 iterations when started from the rougher point. The acceleration scheme SQUAREM (II.2) used close to 20 iterations with either starting point. Computing times for all scenarios considered were less than a second on a personal computer running Linux OS (i5-6500 CPU 3.20GHz) with no parallelization. Regardless of method and software, results for point estimates agreed within reasonable numerical accuracy. However, standard errors differ slightly for the technology coefficients and more so for the parameters of the random components. The results are presented in Table 1.

For the exponential model, starting our routines at $\boldsymbol{\beta}_0 =$ OLS estimates and $\sigma^2_{v,0} = 1/\lambda^2_0 =$ half of the residual variance from OLS, the EM algorithm took far less iterations (around 100) to reach the desired relative precision with SQUAREM requiring only 10 iterations. As opposed to the half normal model, standard errors for the parameters of the random components were less variable across methods. In passing, we note that the likelihood-ratio test of $\sigma_u = 0$ rejected the null (as opposed to the half normal case). The results are presented in Table 2.

Table 1 – Estimation results from half normal SF model with 219 agricultural DMUs and different routines. $\ell(\hat{\boldsymbol{\theta}}) = -170.45$.

	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\sigma}^2_u$	$\hat{\sigma}^2_v$
MLE	1.18	0.73	0.24	0.15	0.22
Standard Errors					
OPG (see eq. (14))	(.59)	(.03)	(.03)	(.10)	(.04)
Hybrid EM (see II.1)	(.60)	(.06)	(.05)	(.12)	(.05)
SQUAREM (see II.2)	(.56)	(.04)	(.03)	(.14)	(.05)
Stata14	(.60)	(.06)	(.05)	(.12)	(.05)

Table 2 – Estimation results from exponential SF model with 219 agricultural DMUs and different routines. $\ell(\hat{\boldsymbol{\theta}}) = -167.94$.

	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\sigma}^2_u$ †	$\hat{\sigma}^2_v$
MLE	1.59	0.70	0.26	0.07	0.20
Standard Errors					
OPG (see eq. (14))	(.58)	(.04)	(.03)	(.02)	(.03)
Hybrid EM (see II.1)	(.60)	(.06)	(.05)	(.03)	(.03)
SQUAREM (see II.2)	(.58)	(.02)	(.03)	(.03)	(.03)
Stata14	(.60)	(.06)	(.05)	(.03)	(.03)

† The routine used is parameterized in terms of $\lambda = 1/\sigma_u$.

The errors for σ^2_u reported here were obtained by delta method.

4.2 Electricity Cost Function

Here we use the EM algorithm to fit the gamma model to electricity data from 1970 ($n = 158$ firms and holding companies) which has appeared in Greene (2003) and other places. The cost function is Cobb-Douglas with a quadratic term in log output ($\log Q$). The other variables involved are total cost of generation (C) and the unit prices of fuel (F), capital (K) and labor (L),

$$\log(C/F)_i = \beta_0 + \beta_1 \log(L/F)_i + \beta_2 \log(K/F)_i + \beta_3 \log Q_i + \beta_4 \log^2 Q_i + v_i + u_i.$$

We note that software regularly used by us (R, Stata and SAS) do not fit the SF gamma model, most likely due to issues in integration and identifiability. LIMDEP (Greene, 2012) uses a simulated likelihood method Greene (2003) to estimate the gamma SF model. Our use of the gamma model (Algorithm D) has been operationalized by numerical integration which is the focus of (Andrade & Souza, 2018). The estimates of the gamma SF model produced by EM (Algorithm D with the appropriate sign changes due to this being a cost function and therefore $\varepsilon = v + u$) are shown in Table 3.

The starting point used was the OLS estimates for the β's, $\alpha_0 = 1$ (exponential model) and $\sigma^2_{v,0}$ and $1/\lambda_0^2 =$ half of the residual variance from OLS.

We note that different choices of initial values have led to other stationary points with lower likelihood. In fact, several local maxima may be the reason for the identifiability questions raised by Ritter & Simar (1997) with respect to the gamma SF model. Simulated maximum likelihood (SML) estimates reported by Greene (2003) are shown in the last line of Table 3. The SML estimates suggest an exponential model and have lower likelihood than that achieved by the EM estimates. We have not succeeded in estimating standard errors by numerical computation of the hessian due to issues involving numerical integration. The OPG method yields the errors reported in Table 3; the standard error associated with $\hat{\lambda}$ is extremely low relative to the estimate and could be the result of numerical inaccuracy.

Running time differences were irrelevant in the model fits of the previous section. Here, we must note that, because numerical integration and root finding are involved in the EM steps, the computing time becomes an issue. EM took approximately 2 minutes to converge and the gain from using SQUAREM is noticeable, with running time of 36 seconds.

Table 3 – Estimation results from gamma SF model with 158 DMUs.

	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\sigma}^2_v$	$\hat{\alpha}$	$\hat{\lambda}$	$\hat{\ell}$
EM	-7.044	0.146	0.135	0.455	0.028	0.012	0.258	5.876	93.4
	(.22)	(.04)	(.04)	(.04)	(.002)	(.002)	(.15)	(.01)	
SML	-7.034	0.145	0.138	0.445	0.028	0.011	1.082	9.577	91.6

5 FINAL REMARKS AND PRACTICAL CONSIDERATIONS

The hierarchical structure of SF models motivated our study of the EM algorithm. We have given the necessary information to implement the EM algorithm in four standard SF models. EM calculations resulted in simple algorithms with closed-form expressions for the half normal and exponential models and more elaborate versions for the truncated normal and gamma models.

This paper does not bring any comparative study of optimization methods for SF models. In general, the user's needs, programming environment, programming ability and objectives, access to built-in routines (for optimization and nonlinear equation solving) and data characteristics will dictate the methods and criteria for useful and specific comparisons. We offer the following general remarks. Since the loglikelihood is available for the standard models (and at least numerically for the gamma model), several general-purpose optimization methods may be used to compute ML estimates. Such methods may require up to second derivatives or be derivative-free, including trust regions, Newton, quasi-Newton, conjugate gradient, Nelder-Mead, pattern search, etc. Specific implementations may require user-supplied derivatives or be automated to compute them numerically. EM is another tool in this toolbox. It does not require derivatives in the estimation but we have used numerically calculated first and second derivatives in the computation of standard errors by means of the hessian and the outer product of gradients estimate (14). It is well known that EM is relatively slow and Newton is very fast when close to the solution (Dennis & Moré, 1974). Differences in running time may be negligible depending on the data size and model used. Yet, small differences may matter if one is performing large simulations. The ascent property (Section 2) gives EM more room for choosing initial values than most methods. Implementing Algorithms A–D in most languages should be straightforward whereas general-purpose methods are much more elaborate. Yet, built-in versions of quasi-Newton are readily available in systems like R and Matlab. EM's computer storage and sensitivity to data sparsity have not been studied by us relative to any other method. Since units may vary widely for different kinds of inputs (hours, tons, dollars, counts) we suggest rescaling $\boldsymbol{x}$ for better numerical stability.

Acceleration schemes for EM have been discussed and the examples showed that SQUAREM accelaration substantially reduces the computing time. We would certainly consider them in implementations of EM for more complex models with covariates and heterocedasticity. The above considerations, as usual, call for further research, especially with regards to the gamma SF model.

References

[1] AIGNER D & CHU DS. 1968. On estimating the industry production function. *American Economic Review*, **58**: 826–839.

[2] ANDRADE BB & SOUZA GS. 2018. Likelihood computation in the normal-gamma stochastic frontier model. *Computational Statistics*, **33**(2): 967–982.

[3] BOBB JF & VARADHAN R. 2018. *turboEM: A Suite of Convergence Acceleration Schemes for EM, MM and Other Fixed-Point Algorithms*. Available at: https://CRAN.R-project.org/package=turboEM. R package version 2018.1.

[4] BOGETOFT P & OTTO L. 2015. *Benchmarking with DEA and SFA*. R package version 0.26.

[5] COELLI T & HENNINGSEN A. 2013. *frontier: Stochastic Frontier Analysis*. Available at: http://CRAN.R-Project.org/package=frontier. R package version 1.1-0.

[6] DEMPSTER AP, LAIRD NM & RUBIN DB. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, **39**(1): 1–38.

[7] DENNIS JE & MORÉ JJ. 1974. A characterization of superlinear convergence and its application to quasi-Newton methods. *Math. Comp.*, **28**: 549–560.

[8] GREENE WH. 1982. Maximum likelihood estimation of stochastic frontier production models. *Journal of Econometrics*, **18**(2): 285–289.

[9] GREENE WH. 2003. Simulated Likelihood Estimation of the Normal-Gamma Stochastic Frontier Function. *Journal of Productivity Analysis*, **19**(2): 179–190.

[10] GREENE WH. 2008. The Econometric Approach to Efficiency Analysis. In: FRIED HO, LOVELL CAK & SCHMIDT SS (Eds.), *The Measurement of Productive Efficiency and Productivity Growth*. chap. 2. Oxford University Press.

[11] GREENE WH. 2012. *LIMDEP 10*. Available at: http://www.limdep.com/. Econometric Software, Inc.

[12] HUANG CJ. 1984. Estimation of Stochastic Frontier Production Function and Technical Inefficiency via the EM Algorithm. *Southern Economic Journal*, **50**(3): 847–856.

[13] IBGE. 2006. *Census of Agriculture 2006*. Available at: http://www.ibge.gov.br/english. Accessed: 2016-04-18.

[14] JAMSHIDIAN M & JENNRICH RI. 1997. Acceleration of the EM Algorithm by Using Quasi-Newton Methods. *Journal of the Royal Statistical Society. Series B*, **59**(3): 569–587.

[15] KOZUMI H & ZHANG X. 2005. Bayesian and non-bayesian analysis of gamma stochastic frontier models by Markov Chain Monte Carlo methods. *Computational Statistics*, **20**(4): 575–593.

[16] KUMBHAKAR S & LOVELL C. 2003. *Stochastic Frontier Analysis*. Cambridge University Press.

[17] LEE LF. 1983. On maximum likelihood estimation of stochastic frontier production models. *Journal of Econometrics*, **23**(2): 269–274.

[18] MCLACHLAN G & KRISHNAN T. 2007. *The EM Algorithm and Extensions*. Wiley.

[19] RITTER C & SIMAR L. 1997. Pitfalls of Normal-Gamma Stochastic Frontier Models. *Journal of Productivity Analysis*, **8**(2): 167–182.

[20] SAS INSTITUTE INC. 2003. `proc qlim`. *SAS/STAT Software, Version 9.1*. Cary, NC. Available at: http://www.sas.com/.

[21] STATACORP. 2015. `frontier`. *Stata Statistical Software: Release 14*. College Station, TX. Available at: http://www.stata.com/.

[22] VARADHAN R & ROLAND C. 2008. Simple and Globally Convergent Methods for Accelerating the Convergence of Any EM Algorithm. *Scandinavian Journal of Statistics*, **35**(2): 335–353.

[23] WU CFJ. 1983. On the Convergence Properties of the EM Algorithm. *Ann. Statist.*, **11**(1): 95–103.

[24] ZHOU H, ALEXANDER D & LANGE K. 2011. A quasi-Newton acceleration for high-dimensional optimization algorithms. *Statistics and Computing*, **21**(2): 261–273.

APPENDIX: R TOOLS

```
############################################################
#  R code using:
#    sfaEM: EM tools for stochastic frontier analysis_
#    R package under development; version 0.0.1 (june/2019)
#    http://probest-unb.weebly.com/sobre.html
#    Author:  Bernardo B. de Andrade
############################################################

install.packages('sfaEM.tar.gz', type='source')

# Half normal production frontier
require(sfaEM)
data(AGRO2006)
x <- agro2006[,2:3]
y <- agro2006$y

# EM and SQUAREM
out <- sfaem(x=x, y=y, dist='halfnorm')
out
pars(out)
stderror(out)   # from numerical Hessian

# OPG standard errors
sqrt(diag(opg(out)))

# EM with lower tolerance, followed by Newton (II.1)
out0 <- sfaem(x=x, y=y, reltol=1e-3)
out1 <- bfgs(out0)
out1
sqrt(diag(out1$invhessian))


# Gamma cost frontier
data(Electricity1970)
x <- with(Electricity1970, {
  data.frame(x1 = log(labor) - log(fuel),
             x2 = log(capital) - log(fuel),
             x3 = log(output),
             x4 = log(output)^2)
       }
       )


y <-  with(Electricity1970, log(cost) - log(fuel))

out <- sfaem(x=x, y=y, dist='gamma', frontier='cost')
out
pars(out)
sqrt(diag(opg(out)))
```