Shooting Methods for Parameter Estimation of Output Error Models ⋆

Antônio H. Ribeiro, * **Luis A. Aguirre** **

* *Graduate Program in Electrical Engineering, Federal University of Minas Gerais - Av. Antônio Carlos 6627, 31270-901, Belo Horizonte. (e-mail: antonio-ribeiro@ufmg.br).*
** *Department of Electronic Engineering, Federal University of Minas Gerais. (e-mail: aguirre@ufmg.br)*

Abstract: This paper studies parameter estimation of output error (OE) models. The commonly used approach of minimizing the free-run simulation error is called single shooting in contrast with the new multiple shooting approach proposed in this paper, for which the free-run simulation error of sub-datasets is minimized subject to equality constraints. The names "single shooting" and "multiple shooting" are used due to the similarities with techniques for estimating ODE (ordinary differential equation) parameters. Examples with nonlinear polynomial models illustrate the advantages of OE models as well as the capability of the multiple shooting approach to avoid undesirable local minima.

Keywords: Multiple shooting, output error models, simulation error minimization, nonlinear least-squares

1. INTRODUCTION

An important step in a typical *system identification* problem is to estimate model parameters using measured data. A widespread class of parameter estimation methods are the so-called *prediction error methods* (Ljung, 1998). These methods estimate the parameters by minimizing the error between the optimal output prediction and the measured value.

Two models from this class are to be discussed in this paper: output error (OE) and autoregressive with exogenous inputs (ARX) models. OE models can be estimated by minimizing the free-run simulation error and ARX models by minimizing the one-step-ahead prediction error. Both are optimal under certain noise assumptions: ARX models for white equation error and OE models for white output error.

For linear-in-the-parameters models (e.g. linear and polynomial models), the estimation of ARX models can be formulated as a linear least-squares problem, while for OE models it results in a non-convex optimization problem and may suffer from the existence of several local minima (Ljung, 1998). Even for general nonlinear models, for which both ARX and OE models estimation results in non-convex problems, the existence of undesirable local minima is more severe for OE models.

In order to ease the problem of several minima, this paper brings into the context of discrete time systems parameter estimation the so-called *multiple shooting* (MS). MS was first described for ordinary differential equation (ODE) parameter estimation by Van Domselaar and Hemker (1975), and further developed by Bock (1983); Baake et al. (1992). It has been successfully applied to estimate parameters for CO_2 lasers models (Horbelt et al., 2001), nonlinear circuits (Timmer et al., 2000) and biochemical processes (Peifer and Timmer, 2007).

MS can be seen as an improvement on the *single shooting* (SS) formulation. In the SS method, the system is simulated over the whole dataset and the parameters are estimated by minimizing the simulation error. On the other hand, MS methods subdivide the data into smaller sets, simulate the system over each set and minimize the sum of dataset errors, while equality constraints enforce the cohesion between the beginning and the end of simulation windows.

In using the MS method it become less likely that small parameter or initial condition variations cause large changes in the system trajectory. This makes the objective function smoother and the optimization algorithm less likely to be trapped in a "poor" local minima. These advantages come at the cost of including extra variables (the initial conditions of each subset) in the optimization problem.

The objective of this paper is to adapt and apply the MS method as a parameter estimation algorithm for OE models in system identification problems.

The rest of the paper is organized as follows: Section 2 explains the parameter estimation of discrete dynamic systems under different noise assumptions. Section 3 formulates SS and MS problems for output error models estimation. Section 4 gives two examples, the first illustrates advantages of working with OE models rather than ARX models and the second the advantages of using MS instead o SS. Section 5 provides concluding remarks.

⋆ This work has been supported by the Brazilian agencies CAPES, CNPq and FAPEMIG.

2. DISCRETE-TIME MODELS PARAMETER ESTIMATION

2.1 Output Error vs Equation Error

Consider the dataset $\mathcal{S} = \{(u[n], y[n]), n = 1, 2, \ldots, N\}$, where $u[n]$ and $y[n]$ are the input and the output of a dynamic system.

It is assumed that the data was generated by a *"true system"*, described by:

$$\begin{aligned} x[n] &= F(x[n-1], \ldots, x[n-n_y], u[n-\tau_d], \ldots, u[n-n_u]; \boldsymbol{\Theta}^*) + v[n] \\ y[n] &= x[n] + w[n], \end{aligned} \tag{1}$$

where F and $\boldsymbol{\Theta}^*$ are the *"true"* function and parameter vector that describe the system, $x[n]$ is the system state and $v[n]$ and $w[n]$ are random variables. The equation error (represented by $v[n]$) affects the system state, while the output error (represented by $w[n]$) only affect the measured values.

2.2 Free-run Simulation and One-step-ahead Prediction

Two concepts used during the paper are defined next:

Definition 1. (One-step-ahead prediction). The one-step-ahead prediction is defined as:

$$\hat{y}[n \mid n-1] = F(y[n-1], \ldots, y[n-n_y], u[n-\tau_d], \ldots, u[n-n_u]; \boldsymbol{\Theta}), \tag{2}$$

Definition 2. (Free-run simulation). The free-run simulation is defined using the recursive formula:

$$\hat{y}[n] = \begin{cases} y_0[n], \ 1 \leq n < n_y; \\ F(\hat{y}[n-1], \ldots, \hat{y}[n-n_y], u[n-\tau_d], \ldots, u[n-n_u]; \boldsymbol{\Theta}), \\ \qquad\qquad\qquad\qquad n \geq ny, \end{cases} \tag{3}$$

for which $y_0[n], \ n = 1, \cdots, n_y$ are the simulation initial conditions.

The simplified notation: $\underline{\mathbf{y}}[n] = [y[n-1], \ldots, y[n-n_y]]^T$, $\underline{\mathbf{u}}[n] = [u[n-\tau_d], \ldots, u[n-n_u]]^T$ will be used from now on. Furthermore, the vector of initial conditions is defined as $\underline{\mathbf{y}}_0 = [y_0[1], \ldots, y_0[n_y - 1]]^T$.

2.3 Optimal Predictor

If the measured values of y and u are known at all instants previous to n, the conditional expected value $E\{y[n]\}$ is an optimal prediction of $y[n]$ (in the sense the expected value of the squared error between the prediction and the actual value is minimum).

In the sequel the following situations will be considered in turn: (i) the equation error is white noise and the output error is zero ($w[n] = 0$); and, (ii) the output error is white noise and the equation error is zero ($v[n] = 0$).

For situation (i), $y[n] = x[n]$ and therefore Equation (1) reduces to $y[n] = F(\underline{\mathbf{y}}[n], \underline{\mathbf{u}}[n]; \boldsymbol{\Theta}^*) + v[n]$. And, because v has zero mean, it follows that:

$$E\{y[n]\} = F(\underline{\mathbf{y}}[n], \underline{\mathbf{u}}[n]; \boldsymbol{\Theta}^*) = \hat{y}[n \mid n-1],$$

and, therefore, the optimal prediction for situation (i) is the one-step-ahead prediction $\hat{y}[n \mid n-1]$.

On the other hand, for situation (ii), there is no equation error and therefore:

$$E\{y[n]\} = E\{x[n] + w[n]\} = x[n] = \hat{y}[n],$$

where it was used that $E\{x[n]\} = x[n]$ because $v[n] = 0$, $E\{w[n]\} = 0$ (zero mean noise) and that for true initial conditions and parameters $x[n] = \hat{y}[n]$. Therefore, the optimal prediction for situation (ii) is the free-run simulation $\hat{y}[n]$.

2.4 Prediction Error Methods

There is whole class of system identification methods that estimates the model parameters by minimizing the optimal prediction error. ARX models minimize one-step-ahead error $e_1[n] = \hat{y}[n \mid n-1] - y[n]$ while OE models minimize the free-run simulation error $e_s[n] = \hat{y}[n] - y[n]$, which are the optimal prediction error for situations (i) and (ii) respectively.

Let $\mathbf{e}_s = [e_s[1], \ldots, e_s[N]]^T$ and $\mathbf{e}_1 = [e_1[1], \ldots, e_1[N]]^T$ be error vectors. Prediction error methods usually minimize the sum of square errors. This choice is optimal (in the maximum likelihood sense) if the residuals are considered to be Gaussian white noise. Algorithms to solve this least square problem for the nonlinear case are discussed next.

2.5 Nonlinear Least-Squares

Consider the error vector $\mathbf{e}(\boldsymbol{\Theta}) \in \mathbb{R}^{N_e}$, the parameter vector $\boldsymbol{\Theta} \in \mathbb{R}^{N_\Theta}$ is estimated minimizing the following sum of square errors:

$$V(\boldsymbol{\Theta}) = \tfrac{1}{2}\|\mathbf{e}(\boldsymbol{\Theta})\|^2 = \tfrac{1}{2}\mathbf{e}^T\mathbf{e} = \tfrac{1}{2}\sum_{i=0}^{N_e} e_i^2(\boldsymbol{\Theta}). \tag{4}$$

Let $J(\boldsymbol{\Theta}) \in \mathbb{R}^{N_e \times N_\Theta}$ be the Jacobian matrix associated with the error vector (Appendix A describes the computation of $J(\boldsymbol{\Theta})$ for $\mathbf{e}_s$ and $\mathbf{e}_1$). The gradient vector and Hessian matrix are (Nocedal and Wright, 2006, p. 246):

$$\nabla V(\boldsymbol{\Theta}) = J(\boldsymbol{\Theta})^T \mathbf{e}(\boldsymbol{\Theta}), \tag{5}$$

$$\nabla^2 V(\boldsymbol{\Theta}) = J(\boldsymbol{\Theta})^T J(\boldsymbol{\Theta}) + \sum_{i=1}^{N_e} e_i(\boldsymbol{\Theta})\, \nabla^2 e_i(\boldsymbol{\Theta}). \tag{6}$$

Nonlinear least-squares (NLS) algorithms update the solution iteratively ($\boldsymbol{\Theta}^{k+1} = \boldsymbol{\Theta}^k + \Delta\boldsymbol{\Theta}^k$). The Gauss-Newton method uses a search direction similar to the Newton's method computing the gradient according to (5) and approximating the Hessian matrix by the first term in (6). Hence, the Gauss-Newton update is:

$$\Delta\boldsymbol{\Theta}^k = -\mu \left[J(\boldsymbol{\Theta}^k)^T J(\boldsymbol{\Theta}^k)\right]^{-1} J(\boldsymbol{\Theta}^k)^T \mathbf{e}(\boldsymbol{\Theta}^k), \tag{7}$$

where the step length μ is computed using a line-search algorithm (Nocedal and Wright, 2006, Cap. 3).

The Levemberg-Marquardt algorithm, on the other hand, considers a parameter update (Marquardt, 1963):

$$\Delta\boldsymbol{\Theta}^k = -\left[J(\boldsymbol{\Theta}^k)^T J(\boldsymbol{\Theta}^k) + \lambda D^T D\right]^{-1} J(\boldsymbol{\Theta}^k)^T \mathbf{e}(\boldsymbol{\Theta}^k), \tag{8}$$

for which λ is a non-negative scalar and D is a non negative diagonal matrix. Fletcher et al. (1971) give heuristic rules for adapting λ through the iterations. Another view is to

consider the Levemberg-Marquardt algorithm as a trust-region method, for which, at each iteration, the following subproblem is (approximately) solved in an elliptical trust region: (Moré, 1978)

$$\min_{\Delta\boldsymbol{\Theta}} \ \tfrac{1}{2}\left\|\mathbf{e}(\boldsymbol{\Theta}^k) + J(\boldsymbol{\Theta}^k)\Delta\boldsymbol{\Theta}^k\right\|^2, \qquad (9)$$
$$\text{subject to:} \ \ \|D\Delta\boldsymbol{\Theta}^k\| \leq \delta.$$

where the trust-region radius δ is updated at each iteration.

3. SINGLE AND MULTIPLE SHOOTING

This section compares two different ways of estimating OE models. In analogy to ODE parameter estimation, the commonly used approach of minimizing the sum of squared simulation errors is called single shooting (SS) while the new multiple shooting (MS) approach investigated in this paper provides a way of improving SS convergence.

3.1 Single Shooting

Given the dataset $\mathcal{S}$ containing input-output pairs, in the SS formulation parameters of OE models are estimated using the following procedure: for a given set of initial conditions $\underline{\mathbf{y}}_0 \in \mathbb{R}^{n_y}$ and a given parameter choice $\boldsymbol{\Theta}$ the system free-run simulation $\hat{y}[n]$ runs over the entire dataset and the simulation error vector $\mathbf{e}_s$ is computed. The parameter vector $\boldsymbol{\Theta}$ is estimated minimizing $1/2\|\mathbf{e}_s\|^2$ using one of the unconstrained optimization algorithms described in Section 2.

There are two different ways to take into account the initial conditions $\underline{\mathbf{y}}_0$, they are: (i) to fix the initial conditions $\underline{\mathbf{y}}_0$ and estimate the model parameters $\boldsymbol{\Theta}$; and, (ii) to define an extended parameter vector $\boldsymbol{\Phi} = [\boldsymbol{\Theta}^T, \underline{\mathbf{y}}_0^T]^T$ and estimate $\underline{\mathbf{y}}_0$ simultaneously with $\boldsymbol{\Theta}$ using the NLS algorithm. Example 1 in this paper uses Formulation (ii).

When using formulation (i), a suitable choice is to set the initial conditions equal to the measured outputs ($y_0[n] = y[n], \ n = 1, \ldots, n_y - 1$). When using formulation (ii) the measured outputs may be used as an initial guess to be refined by the optimization algorithm.

The optimal choice for the initial condition would be $y_0[n] = x[n]$ for $n = 1, \cdots, n_y - 1$. Formulation (i) uses $y_0[n] = y[n]$ what is not optimal since:

$$y[n] = x[n] + w[n] \neq x[n].$$

Formulation (ii) goes one step further and include the initial conditions $y_0[n]$ in the optimization problem, so it converges to $x[n]$ and hence improves the parameter estimation.

3.2 Multiple Shooting

There is an alternative way to formulate the optimization problem. Suppose the dataset $\mathcal{S}$ is subdivided into m_s smaller datasets and, in each one, the free-run simulation $\hat{y}^{(i)}, \ i = 1, 2, \cdots, m_s$ is computed. If the initial conditions of one simulation $\underline{\mathbf{y}}_0^{(i+1)}$ always coincide with the end of the previous one $\underline{\hat{\mathbf{y}}}^{(i)}[\mathbf{end}]$, then the concatenated sequence is forced to be exactly equal to the free-run simulation over the entire dataset, as illustrated in Fig. 1 for $m_s = 3$.

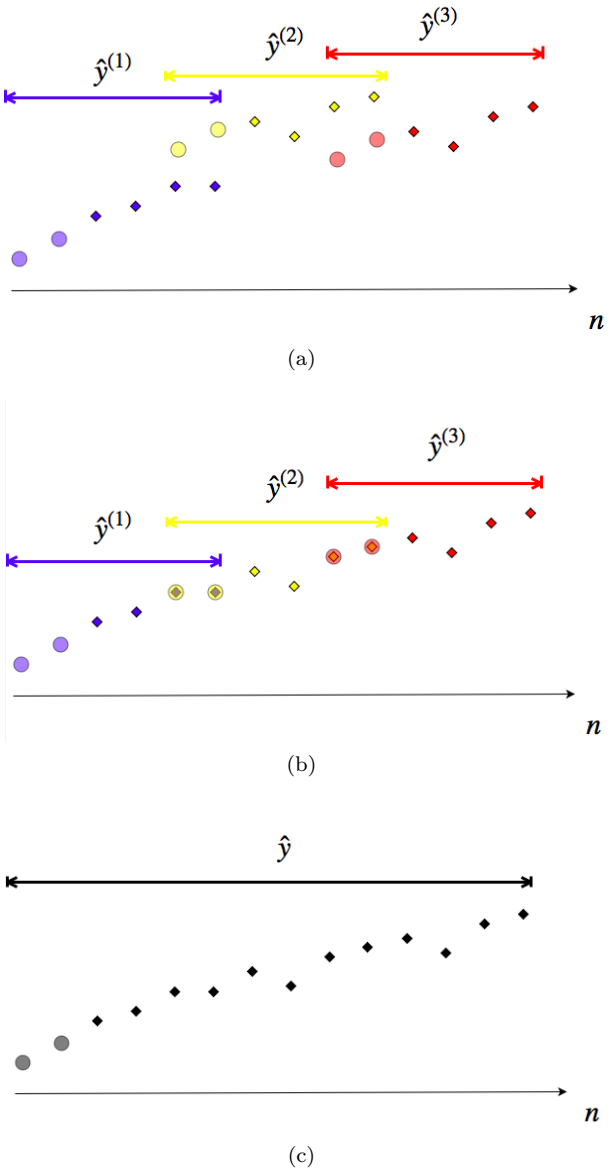


Fig. 1. Three consecutive simulations $\hat{y}^{(i)}, \ i = 1, 2, 3$ are indicated with different colors. The initial conditions are represented with circles ○ and subsequent simulated values with diamonds ◇. In (a), the end of one simulation does not coincide with the initial conditions of the next one. In (b) they do and the concatenation of those short simulations is equivalent to a single one carried over the entire set, shown in (c).

Let $\mathbf{e}_s^{(i)}$ be the error between the free-run simulation $\hat{y}^{(i)}$ and the correspondent measured output, the multiple shooting error is defined as the concatenation of those errors $\mathbf{e}_{\text{ms}} = [(\mathbf{e}_s^{(1)})^T, \cdots, (\mathbf{e}_s^{(m_s)})^T]^T$. It follows from the previous discussion that $\mathbf{e}_{\text{ms}} = \mathbf{e}_s$ when the initial conditions of one simulation coincide with the end of the previous.

Based on this idea, it is proposed to estimate OE model parameters by solving the following constrained optimization problem:

$$\min_{\mathbf{\Phi}} \ \tfrac{1}{2}\|\mathbf{e}_{\text{ms}}\|^2 \qquad (10)$$
$$\text{subject to: } \ \underline{\hat{\mathbf{y}}}^{(i)}[\mathbf{end}] = \underline{\mathbf{y}}_0^{(i+1)}, \ i = 1, \cdots, m_s - 1.$$

where $\underline{\hat{\mathbf{y}}}^{(i)}[\mathbf{end}]$ denotes the last n_y values of the i-th simulated window. The variables for the optimization problem are both the parameter vector and the initial conditions:

$$\mathbf{\Phi} = \left[\ \mathbf{\Theta}^T \ \underline{\mathbf{y}}_0^{(1)^T} \ \cdots \ \underline{\mathbf{y}}_0^{(m_s)^T} \ \right]^T$$

In analogy with ODE parameter estimation methods, this formulation is called multiple shooting (MS). In the next section the advantages of the MS over the SS formulation will be illustrated.

4. IMPLEMENTATION AND TEST RESULTS

To solve the constrained optimization problem that arises in the MS formulation a penalty method was used. In this method, a cost is added to penalize constraint violations, resulting in the following NLS problem:

$$\min_{\mathbf{\Phi}} \ \tfrac{1}{2}\|\mathbf{e}_{\text{ms}}\|^2 + \mu \sum_{i=1}^{m_s-1} \|\underline{\mathbf{y}}_0^{(i+1)} - \underline{\mathbf{y}}^{(i)}[\mathbf{end}]\|^2.$$

where μ is a large penalization constant.

The NLS problems were solved using SciPy (Jones et al., 2001–) implementation of Trust Region Reflective method. Jacobian matrices are computed according to Appendix A and a sparse matrix representation is used for the Jacobian matrix that arises in the MS formulation. The NLS algorithm stop criteria is based on: i) the cost function rate of change; ii) the step norm; and, iii) the gradient norm.

The tests were performed on a computer with a processor Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz; running Arch Linux operational system. No parallelization was used.

4.1 Example 1: Output Error

Consider the nonlinear system:

$$y[n] = \Theta_1 y[n-1] + \Theta_2 u[n-2] + \Theta_3 u^2[n-1] + \Theta_4 y^2[n-2] + \Theta_5 \qquad (11)$$

for which $\mathbf{\Theta} = [0.5, \ 0.8, \ 1, \ -0.05, \ 0.5]^T$

Zero-mean Gaussian distributed values with unit standard deviation were generated. Each randomly generated value was held for 5 samples thus producing the input signal $u[n]$. For this input, the system was simulated for 600 samples and white Gaussian output error with standard deviation σ_e was added to it. The parameter vector $\hat{\mathbf{\Theta}}$ is estimated using the provided data.

This experiment was repeated 100 times for different values of σ_e in order to estimate the expected value and the standard deviation of $\hat{\mathbf{\Theta}}$. These are shown in Figure 2 for Θ_1 as a function of the signal noise ratio ($\text{SNR} = 10\log_{10}\sigma_y^2/\sigma_e^2$). The results for the remaining parameter estimates are similar and are not shown.

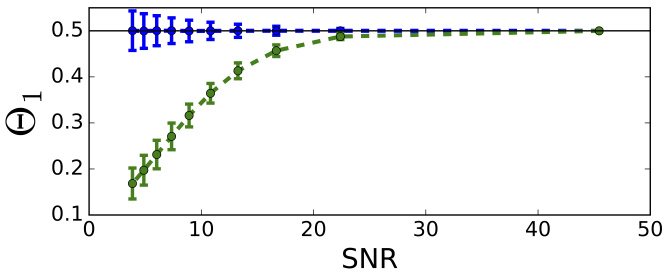


Fig. 2. Expected values (dots) and standard deviation $\pm\sigma_{\hat{\Theta}_1}$ (bars) of $\hat{\Theta}_1$. Estimated using an OE model (in blue) and an ARX model (in green). The true value, Θ_1, is represented as the horizontal line.

Two different models were considered when estimating the parameters: (1) OE model (single shooting) and (2) ARX model. The initial guess used for the parameter vector was $\mathbf{\Theta}^0 = \mathbf{0}$.

Figure 2 shows that the ARX model is biased unlike the OE model. This example confirms that the OE model can provide more accurate parameters estimation in the presence of output error, as happens in several practical applications.

The average running time of the NLS algorithm (when generating Figure 2) was 0.8 seconds for the estimation of the ARX model and 7 seconds for the estimation of the OE model.

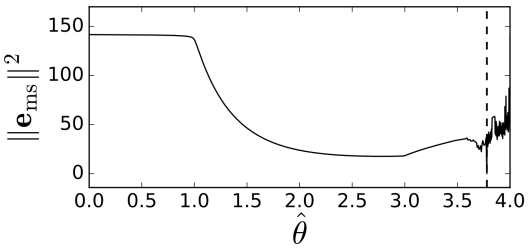
4.2 Example 2: Multimodal Problem

This example illustrates the robustness of the MS method with respect to the initial parameter guess. In order to do so a dataset with 300 samples for $\theta = 3.78$ were generated using the logistic map (May, 1976):

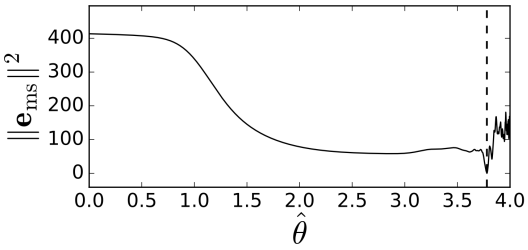
$$y[n] = \theta y[n-1](1 - y[n-1]). \qquad (12)$$

A twofold approach is adopted to illustrate the limitations of SS method for this example: i) For the generated dataset, a slice of the optimization objective function is obtained by varying $\hat{\theta}$ from 0 to 4 and computing the corresponding $\|\mathbf{e}_{\text{ms}}\|^2$ for fixed initial conditions $\underline{\mathbf{y}}_0^{(i)}$; and, ii) for 100 initial guesses θ^0 chosen randomly (uniform) between 0 and 4 the NLS algorithm was used to estimate the parameter θ.

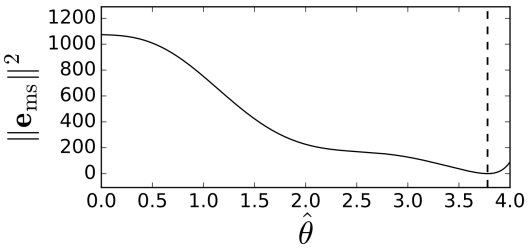
Both experiment were performed for $m_{\text{s}} = 1$, $m_{\text{s}} = 30$, $m_s = 100$ and $m_{\text{s}} = 300$, where m_{s} is the number of divisions used for the MS method. The results are displayed in Figures 3 and 4. Figure 3 displays $\|\mathbf{e}_{\text{ms}}\|^2$ as a function of $\hat{\theta}$ for fixed initial conditions, as described in (i). Figure 4 shows the histograms of the estimated parameter values for random initial guess, as described in (ii). The logistic map is in chaotic regime and if the simulation runs for long enough very small parameters and initial conditions modifications may cause large variations on the simulation trajectory. The intricate error surface that arises for $m_s = 1$ (SS formulation) is a direct consequence of it. As m_s increases, the length of each individual simulation shrinks and the effect of parameter variations become less abrupt, resulting in progressively less intricate error surface (Figure 3).



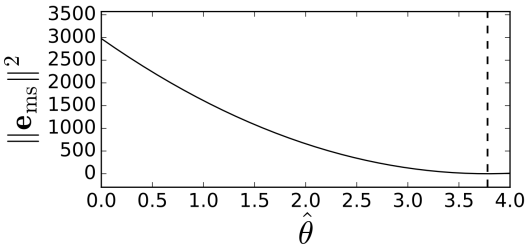
(a) $m_{\mathrm{s}} = 1$



(b) $m_{\mathrm{s}} = 30$

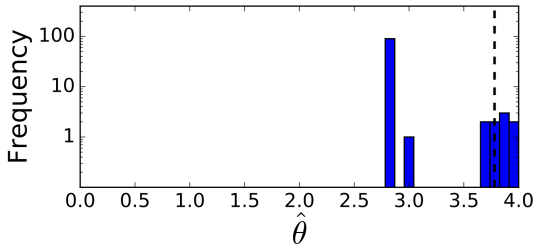


(c) $m_{\mathrm{s}} = 100$

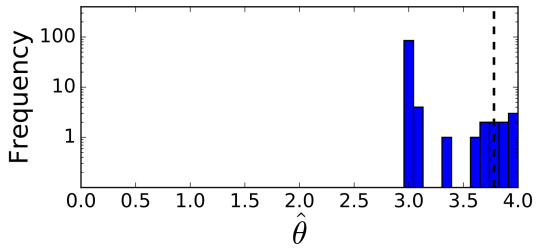


(d) $m_{\mathrm{s}} = 300$

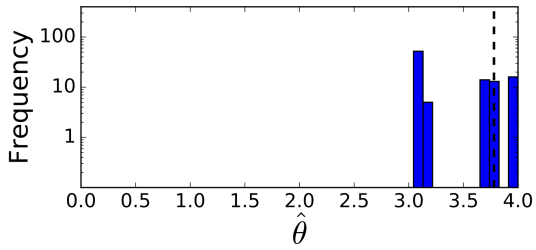
Fig. 3. Shows $\|\mathbf{e}_{\mathrm{ms}}\|^2$ as a function of $\hat{\theta}$ for fixed initial conditions. The vertical dashed line $(--)$ represents the true parameter value.



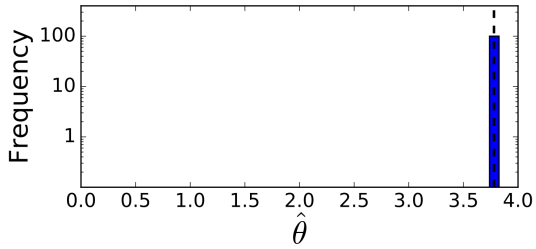
(a) $m_{\mathrm{s}} = 1$



(b) $m_{\mathrm{s}} = 30$



(c) $m_{\mathrm{s}} = 100$



(d) $m_{\mathrm{s}} = 300$

Fig. 4. Histograms (in log scale) of the estimated parameter values to which the algorithm converged. The vertical dashed line $(--)$ represents the true parameter value.

For $m_{\mathrm{s}} = 1$ (SS formulation), the objective function has many minima hence the final value of the optimization process largely depends on the initial guess (Figure 4). On the other hand, for $m_{\mathrm{s}} = 300$ the optimization process seems to always converge to the real value of θ regardless of the initial parameter guess θ^0, due to its single local minima.

It is worth noticing that $m_s = 300$ is an extreme case of the MS formulation, for which only a single step ahead is computed in each sub-dataset (the continuity between these is reinforced by the equality constraints). The optimization result being independent from the initial guess θ^0 only for $m_s = 300$ (Figure 4) seems to suggest the simulation can go, for small perturbation in the parameters, into different trajectories very quickly, resulting in the need to restrict the simulation length to a single step ahead to get the desired convergence.

The running time increases with the number of subdivisions m_s. Nevertheless, for this example, this is not very pronounced, with the average running time varying between 0.3 and 0.5 seconds depending on m_s.

5. FINAL REMARKS

This paper provide two formulations for estimate OE models. The approach usually described in the literature of minimizing the free-run simulation error is called single shooting and is contrasted with the multiple shooting approach. For Example 1 a single shooting approach provides satisfactory results, while for Example 2 it does not. This second example illustrates the effect of MS formulation, that results in a problem less sensitive to the initial guess and less likely to be trapped in a bad local minima. This extra robustness of the MS formulation comes at the cost of including extra optimization variables and equality constraints.

ACKNOWLEDGEMENTS

This work has been supported by the Brazilian agencies CAPES, CNPq and FAPEMIG.

REFERENCES

Aguirre, L.A., Barbosa, B.H., and Braga, A.P. (2010). Prediction and simulation errors in parameter estimation for nonlinear systems. *Mechanical Systems and Signal Processing*, 24(8), 2855–2867.

Baake, E., Baake, M., Bock, H., and Briggs, K. (1992). Fitting ordinary differential equations to chaotic data. *Physical Review A*, 45(8), 5524.

Billings, S.A. (2013). *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons.

Bock, H. (1983). Recent advances in parameter identification problems for ODE. *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, 95–121.

Byrd, R.H., Schnabel, R.B., and Shultz, G.A. (1988). Approximate solution of the trust region problem by minimization over two-dimensional subspaces. *Mathematical Programming*, 40(1-3), 247–263.

Fletcher, R., Authority, U.K.A.E., and H.M.S.O. (1971). *A Modified Marquardt Subroutine for Non-linear Least Squares*. AERE report. Theoretical Physics Division, Atomic Energy Research Establishment.

Horbelt, W., Timmer, J., Bünner, M., Meucci, R., and Ciofini, M. (2001). Identifying physical properties of a CO2 LASER by dynamical modeling of measured time series. *Physical Review E*, 64(1), 016222.

Jones, E., Oliphant, T., Peterson, P., et al. (2001–). SciPy: Open source scientific tools for Python. URL `http://www.scipy.org/`. [Online; accessed 2016-09-22].

Ljung, L. (1998). *System identification*. Springer.

Marquardt, D.W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2), 431–441.

May, R.M. (1976). Simple mathematical models with very complicated dynamics. *Nature*, 261(5560), 459–467.

Moré, J.J. (1978). The Levenberg-Marquardt algorithm: implementation and theory. In *Numerical Analysis*, 105–116. Springer.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer Science & Business Media, second edition.

Peifer, M. and Timmer, J. (2007). Parameter estimation in ordinary differential equations for biochemical processes using the method of multiple shooting. *IET Systems Biology*, 1(2), 78–88.

Timmer, J., Rust, H., Horbelt, W., and Voss, H. (2000). Parametric, nonparametric and parametric modelling of a chaotic circuit time series. *Physics Letters A*, 274(3), 123–134.

Van Domselaar, B. and Hemker, P.W. (1975). Nonlinear parameter estimation in initial value problems. Technical report, SIS-76-1121.

Voglis, C. and Lagaris, I. (2004). A rectangular trust region dogleg approach for unconstrained and bound constrained nonlinear optimization. In *WSEAS Conference*, 17–19.

Appendix A. COMPUTING THE DERIVATIVES

The derivatives of $\hat{y}[n \mid n-1]$, $\hat{y}[n]$ and $\hat{y}^{(i)}[n]$ in relation to the parameters and initial conditions can be computed accordingly to the following propositions and used to construct the Jacobian matrices of $\mathbf{e}_1$, $\mathbf{e}_s$ and $\mathbf{e}_{\text{ms}}$.

Proposition 1. The partial derivative of $\hat{y}[n \mid n-1]$ in relation to the j-th parameter being estimated is:

$$\frac{\partial \hat{y}[n|n-1]}{\partial \Theta_j} = \frac{\partial F}{\partial \Theta_j}(\underline{\mathbf{y}}[n], \underline{\mathbf{u}}[n]; \boldsymbol{\Theta})$$

Proposition 2. The partial derivatives of $\hat{y}[n]$ can be computed using the following recursive formulas:

$$\frac{\partial \hat{y}[n]}{\partial \Theta_j} = \begin{cases} 0, \text{ if } 1 \leq n < n_y; \\ \frac{\partial F}{\partial \Theta_j}(\hat{\mathbf{y}}, \mathbf{u}, \boldsymbol{\Theta}) + \sum_{i=1}^{N} \frac{\partial F}{\partial y[n-i]}(\hat{\mathbf{y}}, \mathbf{u}, \boldsymbol{\Theta}) \frac{\partial \hat{y}[n-i]}{\partial \Theta_j}, \\ \qquad \text{if } n \geq n_y, \end{cases}$$

$$\frac{\partial \hat{y}[n]}{\partial y_0[j]} = \begin{cases} 1, \text{ if } 1 \leq n < n_y \text{ and } n = j; \\ 0, \text{ if } 1 \leq n < n_y \text{ and } n \neq j; \\ \sum_{i=1}^{N} \frac{\partial F}{\partial y[n-i]}(\hat{\mathbf{y}}, \mathbf{u}, \boldsymbol{\Theta}) \frac{\partial \hat{y}[n-i]}{\partial y_0[j]}, \text{ if } n \geq n_y, \end{cases} \tag{A.1}$$

for which $\frac{\partial \hat{y}[n]}{\partial \Theta_j}$ and $\frac{\partial \hat{y}[n]}{\partial y_0[j]}$ are, respectively, the derivatives of $\hat{y}[n]$ with respect to the j-th element of $\boldsymbol{\Theta}$ and to the initial condition y_0 at time j.