Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Software for Approximate Linear System Identification

Ivan Markovsky, Jan C. Willems, Sabine Van Huffel, and Bart De Moor

***Abstract*— The main features of the considered identification problem are that there is no a priori separation of the variables into inputs and outputs and the approximation criterion, called misfit, is representation invariant. The misfit is defined as the minimum of the ℓ_2-norm between the given time series and a time series that is consistent with the approximate model. The misfit is equal to zero if and only if the model is exact and the smaller the misfit is (by definition) the more accurate the model is. The considered model class consists of all linear time-invariant systems of bounded complexity and the complexity is specified by the number of inputs and the smallest number of lags in a difference equation representation.**

We present a MATLAB function for approximate identification based on misfit minimization. Although the problem formulation is representation independent, we use an input/state/output representation of the identified system in order to allow maximum compatibility with other software packages for system identification, analysis, and design.

***Index Terms*— System identification, behavioral systems, software development.**

I. INTRODUCTION

This paper describes MATLAB functions (m-files) for approximate linear time-invariant (LTI) system identification. We use the behavioral language. A discrete-time dynamical system $\mathscr{B} \subset (\mathbb{R}^q)^{\mathbb{Z}}$ is a collection of trajectories (q-variables time-series $w : \mathbb{Z} \to \mathbb{R}^q$). No a priori distinction of the variables in inputs and outputs is made and the system is not a priori bound to a particular representation. The variables w can be partitioned into inputs u (free variables) and outputs y (dependent variables) and the system can be represented in various equivalent forms, *e.g.*, the ubiquitous input/state/output representation

$$x(t+1) = Ax(t) + Bu(t), \quad y(t) = Cx(t) + Du(t). \quad \text{(I/S/O)}$$

The number of inputs m, the number of outputs p, and the minimal state dimension n of an input/state/output representation are invariant of the representation and in particular of the input/output partitioning.

The class of finite dimensional linear time-invariant systems with q variables and at most m inputs is denoted by $\mathscr{L}_m^q$. The number of inputs and the minimal state dimension specify the complexity of the system in the sense that the dimension of the restriction of $\mathscr{B}$ to the interval $[1,T]$, where $T \geq n$, is a $Tm+n$ dimensional subspace. Equivalently, the complexity of the system can be specified by the input dimension and the *lag* of the system. The lag of $\mathscr{B}$ is the minimal natural number l, for which there exists an lth order difference equation

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_l w(t+l) = 0 \quad \text{(DE)}$$

representation of the system, *i.e.*, $\mathscr{B} = \{ w \mid \text{(DE) holds} \}$. The subset of $\mathscr{L}_m^q$ with lag at most l is denoted by $\mathscr{L}_{m,l}^q$.

The considered identification problem is the global total least squares problem of Roorda and Heij [1], [2]:

> Given a time series $w_{\mathrm{d}} \in (\mathbb{R}^q)^T$ and a complexity specification (m,l), find the system
>
> $$\hat{\mathscr{B}} := \arg \min_{\mathscr{B} \in \mathscr{L}_{m,l}^q} M(w_{\mathrm{d}}, \mathscr{B}), \quad \text{(GTLS)}$$
>
> where $M(w_{\mathrm{d}}, \mathscr{B}) := \min_{\hat{w} \in \mathscr{B}} \ \|w_{\mathrm{d}} - \hat{w}\|_{\ell_2}^2$.

$M(w_{\mathrm{d}}, \mathscr{B})$ is the *misfit* (lack of fit) between w_{d} and $\mathscr{B}$. It shows how much the model $\mathscr{B}$ fails to "explain" the data w_{d}. The optimal approximate modeling problem (GTLS) aims to find the system $\hat{\mathscr{B}}$ in the model class $\mathscr{L}_{m,l}^q$ that best fits the data according to the misfit criterion.

Solution approach

Denote by $\mathscr{H}_{l+1}(w)$, where $w = \big(w(1), \ldots, w(T)\big)$, the block Hankel-matrix

$$\mathscr{H}_{l+1}(w) := \begin{bmatrix} w(1) & w(2) & \cdots & w(T-l) \\ w(2) & w(3) & \cdots & w(T-l+1) \\ \vdots & \vdots & & \vdots \\ w(l+1) & w(l+2) & \cdots & w(T) \end{bmatrix}.$$

The global total least squares problem is solved in the following generically equivalent [2] formulation:

$$\hat{X} = \arg\min_X \left(\min_{\hat{w}} \ \|w_{\mathrm{d}} - \hat{w}\|_{\ell_2}^2 \text{ s.t. } \mathscr{H}_{l+1}^\top(\hat{w}) \begin{bmatrix} X \\ -I \end{bmatrix} = 0 \right), \quad \text{(STLS)}$$

known as the structured total least squares problem [3]. The parameter X of (STLS) is related to the parameters $R_0, R_1, \ldots, R_l$ of the difference equation representation (DE) of the approximating system $\mathscr{B}$ as follows:

$$\begin{bmatrix} X^\top & -I \end{bmatrix} = \begin{bmatrix} R_0 & R_1 & \cdots & R_l \end{bmatrix}.$$

Software for solving the structured total least squares problem (STLS) is presented in [4]. It is used as the core computational tool for solving the system identification problem. In fact, the software presented in this paper can be viewed as an interface to the STLS solver for the purpose of LTI system identification.

I. Markovsky, J. C. Willems, S. Van Huffel, and B. De Moor are with the Electrical Engineering Department, K.U.Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium.

0-7803-9568-9/05/$20.00 ©2005 IEEE

The STLS solver gives as a result a difference equation representation of the optimal approximating system $\hat{\mathscr{B}}$. The function `stlsident`, described in Section II, converts the parameter $\hat{X}$ to the parameters $(\hat{A},\hat{B},\hat{C},\hat{D})$ of an input/state/output representation of $\hat{\mathscr{B}}$.

Note 1. The algorithms of Roorda and Heij are based on isometric state representations. Thus although the problem formulation is the same the algorithms of [1], [2] are essentially different than ours.

II. Software package

The function:

- `stlsident` solves the approximate identification problem (GTLS), and the function
- `misfit` computes the misfit $M(w_{\mathrm{d}},\mathscr{B})$.

Both functions use the input/state/output representation (I/S/O) of the systems that are returned as an output and accepted as an input, so that they can be viewed as implementations of the following mappings:

- `stlsident`: $(w_{\mathrm{d}},m,l)\mapsto(\hat{A},\hat{B},\hat{C},\hat{D})$, and
- `misfit`: $\big(w_{\mathrm{d}},(A,B,C,D)\big)\mapsto(M,\hat{w}_{\mathrm{d}})$.

The following are special case and extensions:

- the specification $m=0$ corresponds to an output-only system identification ($\hat{\mathscr{B}}$ autonomous);
- the functions work with multiple given time series $w_k=\big(w_k(1),\ldots,w_k(T)\big)$, $k=1,\ldots,N$; and
- some elements of w can be specified as "exact", in which case they appear unmodified in the approximation $\hat{w}$.

Using a combination of these options, one can solve approximately the realization problem, the finite time ℓ_2 model reduction problem (see [2, Section 5]), and the output error identification problem. Examples are given in Section III.

Calling sequences

```
[sysh,info,wh,xini]=stlsident(w,m,l,opt)
```

Inputs:

- `w` the given time series w_{d}; a real MATLAB array of dimension $T\times q\times N$, where T is the number of samples, q is the number of variables, and N is the number of time series.
- `m` the input dimension for the identified system.
- `l` the lag of the identified system.
- `opt` options for the optimization algorithm.
  - `opt.exct` (default []) a vector of indices for exact variables.
  - `opt.sys0` (default total least squares approximation) an initial approximation: an input/state/output representation of a system, given as MATLAB's `ss` object (see `help ss`), with `m` inputs, `w-m` outputs, and order `l*(w-m)`.
  - `opt.disp` (default `'notify'`) level of displayed information about the optimization process. The options are: `'off'`—silent, `'notify'`—only if not converged, `'final'`—convergence status, `'iter'`—per iteration.
  - `opt.maxiter` (default 100) a maximum number of iterations.
  - `opt.epsrel` (default 10^{-5}), `opt.epsabs` (default 10^{-5}), and `opt.epsgrad` (default 10^{-5}) convergence tolerances. The convergence condition is:

$$|X_{ij}^{(k+1)}-X_{ij}^{(k)}|<\texttt{opt.epsabs}+\texttt{opt.epsrel}\ |X_{ij}^{(k+1)}|,\quad\text{for all } i,j$$
$$\text{or}\quad \|M'(X^{(k+1)})\|<\texttt{opt.epsgrad}$$

    where

$$X^{(k)},\ k=1,2,\ldots,\texttt{info.iter}\le\texttt{opt.maxiter}$$

    are the successive iterates of the parameter X and $M'(X^{(k+1)})$ is the gradient of the cost function at the current iteration step.

Outputs:

- `sysh` an input/state/output representation of the identified system $\hat{\mathscr{B}}$.
- `info` information from the optimization solver:
  - `info.M` the misfit $M(w_{\mathrm{d}},\hat{\mathscr{B}})$.
  - `info.time` the execution time for the STLS solver. (Not equal to the execution time of `stlsident`.)
  - `info.iter` the number of iterations. Note that `info.iter = opt.maxiter` indicates lack of convergence to a desired convergence tolerance.
- `wh` the optimal approximating time series.
- `xini` a matrix whose columns are the initial condition, under which $\hat{w}_k$, $k=1,\ldots,N$ are obtained.

```
[M,wh,xini] = misfit(w,sys,exct)
```

Inputs:

- `w` the given time series w_{d}, a real MATLAB array of dimensions $T\times q\times N$, where T is the samples, q is the number of variables, and N is the number of time series.
- `sys` an input/state/output representation of a system $\mathscr{B}$, given as MATLAB's `ss` object (see `help ss`), with q external variables (inputs and outputs) and of order which is a multiple of the number of outputs.
- `exct` (default []) a vector of indices for exact variables;

Outputs:

- `M` the misfit $M(w_{\mathrm{d}},\mathscr{B})$.
- `wh` optimal approximating time series $\hat{w}$.
- `xini` a matrix whose columns are the initial condition, under which $\hat{w}_k$, $k=1,\ldots,N$ are obtained.

The package is available from [5].

Note 2. A MATLAB implementation of the algorithms of Roorda and Heij is included in [6]. Up to convergence problems due to the nonconvexity of the optimization problem (GTLS) the two implementations give equivalent results. We do not include in this paper, however, comparison results in terms of robustness with respect to local minima, speed, and efficiency of the two implementations.

III. EXAMPLES

The numerical examples shown in this section were originally designed as test examples for the function `stlsident`. At the same time, however, they illustrate the flexibility of the developed software and have a tutorial value. The MATLAB scripts are collected in a demo file accompanying the package.

A. Approximation by a static model

As static models are special case of dynamic models (with lag $l = 0$), we validate numerically that the software works correctly with complexity specification $(m, 0)$, *i.e.*, with the model class of linear static models $\mathscr{L}^q_{m,0}$. In this case the approximation problem (GTLS) coincides with the classical total least squares problem [7].

```
>> l = 0; m = 2; p = 3; T = 20;
>>
>> % Generate data
>> n    = l*p;           % order
>> sys0 = drss_(n,p,m);  % true system
>> u0   = randn(T,m);    % true input
>> y0   = lsim(sys0,u0); % true output
>> w0   = [u0 y0];       % true data
>> w    = w0;            % exact data
>>
>> % Identify the system
>> [sysh,info,wh] = stlsident(w,m,l);
>> info
info = iter: 1 time: 0 M: 1.5838e-16
>>
>> % Verify the results
>> err_sys = norm(sys0 - sysh)
err_sys =   0
>> err_w   = norm(w0 - wh,'fro')
err_w =     0
```

B. Exact data

The data is called exact if it is generated by a model in the considered model class and the identification problem is called exact if the data is exact. As approximate identification includes as a special case exact identification, next we validate that the software works correctly with exact data.

```
>> l = 2; m = 2; p = 2; T = 100;
>>
>> % Generate data
>> n  = l*p;           % order
>> sys0 = drss_(n,p,m); % true system
>> u0 = randn(T,m);     % true input
>> y0 = lsim(sys0,u0);  % true output
>> w0 = [u0 y0];        % true data
>> w  = w0;             % exact data
>>
>> % Identify the system
>> [sysh,info,wh] = stlsident(w,m,l);
>> info
info = iter: 1 time: 0.0200 M: 1.8817e-15
>>
>> % Verify the results
>> err_sys = norm(sys0 - sysh)
err_sys = 2.4896e-15
>> err_w   = norm(w0 - wh, 'fro')
err_w =   2.0814e-15
```

In the above example, the data generating system `sys0` has m inputs and lag l. If `sys0` were of order less than $n := pl$, *i.e.*, if it were in the interior of the model class $\mathscr{L}^q_{m,l}$, then the function `stlsident` would still recover `sys0` exactly but the identified system `sysh` would be nonminimal.

In the following examples, if not redefined, the true system `sys0` and the true data `w0` are the same as in the "exact data" example.

C. Errors-in-variables system identification

The basic use of `stlsident` is for identification from data generated according to the errors-in-variables model:

$$w_{\mathrm{d}} = \bar{w} + \tilde{w}, \text{ where } \bar{w} \in \bar{\mathscr{B}} \in \mathscr{L}^q_{m,l}, \ \tilde{w} \sim \mathrm{N}(0, \bar{\sigma}^2 I). \quad \text{(EIV)}$$

The system $\bar{\mathscr{B}}$ is the to-be-estimated system, called the "true" system, $\bar{w}$ is the "true" data, and $\tilde{w}$ is an additive disturbance, called "measurement noise". The optimization problem (GTLS) defines the maximum-likelihood estimator in the errors-in-variables setup. Its solution $\hat{\mathscr{B}}$ (the estimator) is consistent and asymptotically normal.

The following script shows a simulation example of an errors-in-variables identification problem.

```
>> % Perturb the "true" data w0 with noise
>> w = w0 + 0.5 * randn(T,m+p);
>>
>> % Identify the system
>> [sysh,info,wh] = stlsident(w,m,l);
>> info
info = iter: 100 time: 1.0100 M: 6.7320
>>
>> % Verify the results
>> err_data = norm(w0-w,'fro')/norm(w0,'fro')
err_data =    0.2605
>> err_appr = norm(w0-wh,'fro')/norm(w0,'fro')
err_appr =    0.1963
```

D. Output error identification

The output error system identification problem is the (GTLS) problem with the addition constrain that the first m variables of the given time series w_{d} are preserved unmodified in the approximation $\hat{w}$. One can think of these variables as "exact". The following script shows a simulation example of output error identification problem.

```
>> % Perturb the "true" output y0 with noise
>> w = w0 + 0.75 * [zeros(T,m) randn(T,p)];
>>
>> % Identify the system
>> opt.exct = 1:m; % exact inputs
>> [sysh,info,wh] = stlsident(w,m,l,opt);
>> info
info = iter: 20 time: 0.2200 M: 7.4936
>>
>> % Verify the results
>> e_data = norm(w0-w,'fro')/norm(w0,'fro')
e_data =    0.2590
>> e_appr = norm(w0-wh,'fro')/norm(w0,'fro')
e_appr =    0.1786
```

E. Output-only identification

The (GTLS) problem with complexity specification $(0,l)$ is an output-only identification problem. (The model class $\mathscr{L}^q_{0,l}$ consists of all autonomous systems with q variables of lag less than or equal to l.) The next script shows a simulation example of an output-only identification problem. Fig. 1 shows the fit of the true and the data sequences by the estimated sequences.

```
>> % Generate data
>> T     = 20; % length of the data sequence
>> xini0 = randn(n,1);
>> y0    = initial(sys0,xini0,T);
>> w0    = y0;
>> % Perturb the "true" data w0 with noise
>> w = w0 + 0.25 * randn(T,p); % given data
>>
>> % Identify the system
>> [sysh,info,wh] = stlsident(w,0,l);
>> info
info = iter: 100 time: 0.1500 M: 1.0719
>>
>> % Verify the results
>> e_data = norm(w0-w,'fro')/norm(w0,'fro')
e_data =    0.7269
>> e_appr = norm(w0-wh,'fro')/norm(w0,'fro')
e_appr =    0.4184
```

F. Identification from step response observations

Step s_d and impulse h_d response observations of a multi-input systems consist of multiple (equal length) time series. In addition, in both cases the initial conditions are a priori known to be zero and the inputs are exactly known. These features make the identification problems from step and impulse response challenging test examples for `stlsident`.

First we consider the problem of identifying a MIMO system from step response observations. Multiple time series are naturally treated by `stlsident` and exact variables can be specified by the user defined options. The possibility to fix zero initial conditions, however, is not yet implemented. In order to account for the zero initial conditions, we precede the given data s_d by l zero samples. If it were possible to specify that these samples are exact, we would have achieved optimal approximation of the observations, according to the misfit

$$M(s_\mathrm{d},\mathscr{B}) = \min \|s_\mathrm{d} - \hat{s}\|_{\ell_2} \quad \text{subject to} \quad \hat{s} \text{ is a step response of } \mathscr{B}.$$

The added zero entries corresponding to the outputs of the system, however, are in general modified and therefore the solution is suboptimal.

The following script shows a simulation example of a step response identification problem.

```
>> % Generate data
>> T   = 150;  % length of the data sequence
>> y0  = step(sys0,T);
>> % Perturb the "true" data y0 with noise
>> y   = y0 + 0.25 * randn(T,p,m);
>>
>> % Construct the inputs
>> u0   = zeros(T,m,m);
>> for i = 1:m, u0(:,i,i) = ones(T,1); end
>>
>> w = [u0 y]; % input/output data
>> % Precede it with l zeros
>> wext = [zeros(l,m+p,m); w];
>>
>> % Identify the system from the ext. data
>> opt.exct = [1:m]; % exact inputs
>> [sysh,info,whext]=stlsident(wext,m,l,opt);
>> info
info = iter: 100 time: 3.2200 M: 5.9504
>>
>> % Remove the trailing part
>> wh = whext(l+1:end,:,:);
>>
>> % Verify the results
>> w0 = [u0 y0];
>> e_data = norm(w0(:)-w(:))/norm(w0(:))
e_data =    0.1711
>> e_appr = norm(w0(:)-wh(:))/norm(w0(:))
e_appr =    0.0384
```

G. Identification from impulse response observations

Identification from exact impulse response is the topic of (partial) realization theory. When the given data (impulse response observations) is not exact, an approximation is needed. Kung's algorithm is a well known solution for this problem. It is, however, suboptimal in terms of the misfit criterion

$$M(h_\mathrm{d},\mathscr{B}) = \min \|h_\mathrm{d} - \hat{h}\|_{\ell_2} \quad \text{subject to} \quad \hat{h} \text{ is an impulse response of } \mathscr{B}.$$

The (GTLS) problem can be used to find optimal in terms of $M(h_\mathrm{d},\cdot)$ approximate model.

One possibility is to treat the problem as an input/output identification problem. As in the identification problem from step response observations, however, the zero initial conditions can not be specified in the current implementation of `stlsident`. Another possibility is to treat the identification problem from impulse response observations as an output only-identification problem. This approach is based on the equivalence of the impulse response of the system (I/S/O) and the free responses of the autonomous system

$$x(t+1) = Ax(t), \qquad y(t) = Cx(t) \qquad \text{(AUT)}$$

under initial conditions—the columns of B.

```
>> % Generate data
>> T  = 50; % length of the data sequence
>> y0 = impulse(sys0,T);
>>
>> % Perturb the "true" data y0 with noise
>> y  = y0 + 0.25 * randn(T,p,m);
>>
>> % Solution 1: I/O identification.
>>
>> % Construct the inputs
>> u0 = zeros(T,m,m); u0(1,:,:) = eye(m);
>>
>> w = [u0 y]; % given input/output data
>> % Precede w with l zeros
>> wext = [zeros(l,m+p,m); w];
```

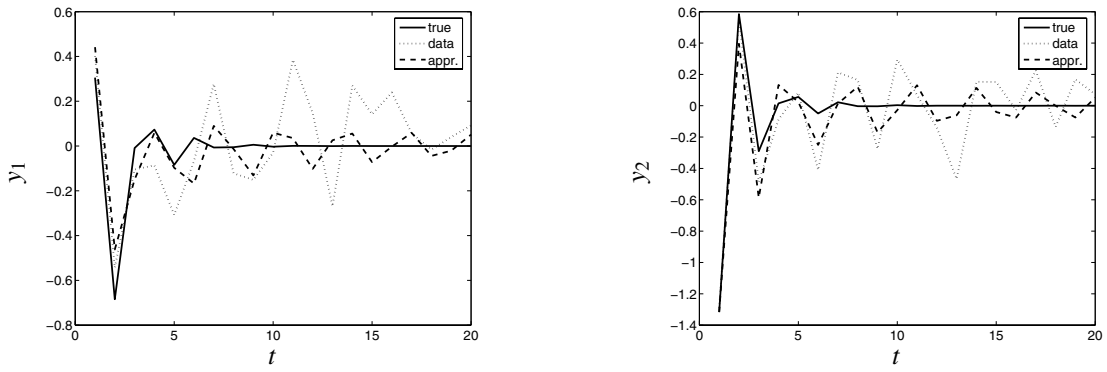

Fig. 1. Output-only identification. Solid line—exact trajectory $\bar{w}$, dotted line—data w_{d}, dashed line—approximating trajectory $\hat{w}$.

```
>> % Identify the system from the ext. data
>> opt.exct = [1:m]; % exact input
>> [sysh1,info1,whext] = ...
                      stlsident(wext,m,l,opt);
>> info1 = info1
info1 = iter: 100 time: 1.1300 M: 3.4258
>>
>> % Remove the trailing part
>> wh1 = whext(l+1:end,:,:);
>>
>> % Solution 2: output-only identification.
>>
>> % Identify an autonomous system
>> [sysh2,info2,yh2,xini2] = ...
                   stlsident(y(2:end,:,:),0,l);
>> yh2 = [y(1,:,:); yh2];
>> info2 = info2
info2 = iter: 100 time: 0.6300 M: 3.3443
>>
>> % Recover the I/O system
>> sysh2 = ss(sysh2.a,xini2,sysh2.c, ...
               reshape(yh2(1,:,:),p,m),-1);
>> wh2 = [u0 yh2];
>>
>> % Verify the results
>> w0 = [u0 y0];
>> e_data  = norm(w0(:)-w(:))/norm(w0(:))
e_data =     0.9219
>> e_appr1 = norm(w0(:)-wh1(:))/norm(w0(:))
e_appr1 =    0.2716
>> e_appr2 = norm(w0(:)-wh2(:))/norm(w0(:))
e_appr2 =    0.2608
```

H. Finite time ℓ_2 model reduction

The finite time T, ℓ_2 norm of a system $\mathscr{B} \in \mathscr{L}^q_{m,l}$ with an impulse response h is defined as

$$\|\mathscr{B}\|_{\ell_2,T} := \||h|_{[0,T]}\|_{\ell_2} = \sqrt{\textstyle\sum_{t=0}^{T} \|h(t)\|_{\mathrm{F}}^2}.$$

For a strictly stable system $\mathscr{B}$, $\|\mathscr{B}\|_{\ell_2,\infty}$ is well defined and is equal to its $\mathscr{H}_2$ norm. The finite time ℓ_2 model reduction problem is:

> Given a system $\bar{\mathscr{B}} \in \mathscr{L}^q_{m,l}$, a natural number $l_{\mathrm{red}} < l$, and a time horizon T, find a system $\hat{\mathscr{B}} \in \mathscr{L}^q_{m,l_{\mathrm{red}}}$, that minimizes the finite time T, ℓ_2 norm $\|\bar{\mathscr{B}} - \hat{\mathscr{B}}\|_{\ell_2,T}$ of the error system.

The solution of this problem is equivalent to the identification problem from an impulse response observation. The following script shows a numerical example. Figure 2 shows the fitting of the impulse response of the original (high order) system by the impulse response of the reduced order system.

```
>> l  = 10; % lag of the high order system
>> lr = 1;  % lag of the reduced system
>> m  = 2;  % # inputs
>> p  = 2;  % # outputs
>>
>> % High order system
>> sys = drss_(p*l,p,m);
>>
>> % Simulate impulse response
>> % (determine automatically T)
>> h = impulse(sys);
>> T = size(h,1);
>>
>> % Find reduced model
>> [sysr,info,hr,xini] = ...
>>          stlsident(h(2:end,:,:),0,lr);
>> hr = [h(1,:,:); hr];
>> sysr = ss(sysr.a,xini,sysr.c, ...
              reshape(hr(1,:,:),p,m),-1);
>> info
info = iter: 18 time: 0.1100 M: 3.5908
>>
>> % Relative Hinf and H2 norms of the
>> % error system
>> h2_err=norm(sys-sysr,2)/norm(sys,2)
h2_err = 0.5253
>> hi_err=norm(sys-sysr,'inf')/norm(sys,'inf')
hi_err = 0.5853
```

I. The optimal misfit is generically independent of the input/output partitioning

The optimal misfit value of the (GTLS) problem is invariant to permutation of the elements of w_{d}, *i.e.*, for an arbitrary permutation matrix $\Pi \in \mathbb{R}^{q\times q}$,

$$\min_{\mathscr{B}\in\mathscr{L}^q_{m,l}} M(w_{\mathrm{d}},\mathscr{B}) = \min_{\mathscr{B}\in\mathscr{L}^q_{m,l}} M(\Pi w_{\mathrm{d}},\mathscr{B}).$$

This property of the problem allows to make no distinction of the ordering of the variables. In the next simulation example, we verify the invariance property numerically.

```
>> % Identify the system from the original
```

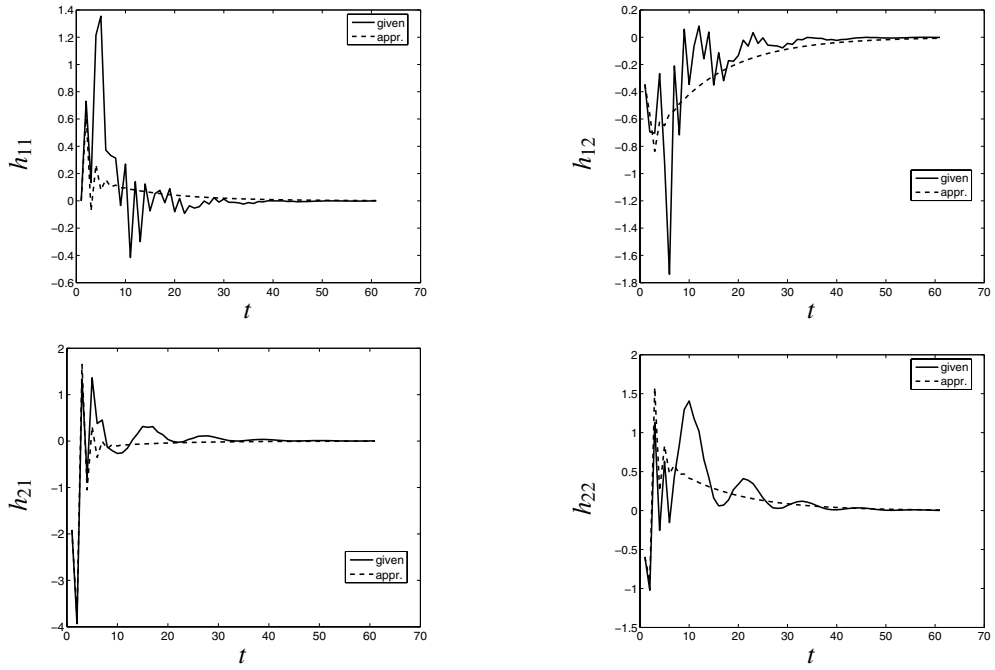

Fig. 2. Finite time ℓ_2 model reduction. Solid line—impulse response of the high-order system, dashed line—impulse response of the reduced order system.

```
>> % exact data
>> [sysh1,info1,wh1,xini1]=stlsident(w0,m,l);
>> info1 = iter: 1 time: 0.0200 M: 3.5617e-32
>>
>> % Identify the system from exact data
     with randomly permuted variables
>> perm = randperm(m+p)
perm =    3     4     1     2
>> w2    = w0(:,perm);
>> [sysh2,info2,wh2,xini2]=stlsident(w2,m,l);
>> info2 = iter: 1 time: 0 M: 0
>> % Compare the results
>> norm(wh1(:,perm) - wh2)
ans =   2.0822e-32
```

Although (GTLS) is permutation invariant, the optimization problem may behave differently for different permutation matrices Π. The reason for this is the possible existence of multiple local minima of the misfit function.

IV. Conclusions

We presented a software package for approximate system identification and illustrated on numerical examples its application for approximate realization and model reduction. The structured total least squares method allows to treat identification problems without input/output partitioning of the variables and errors-in-variables identification problems. In addition, multiple time series and prior knowledge about exact variables can be taken into account.

Acknowledgments

Research supported by: **Research Council KUL:** GOA-Mefisto 666, GOA-Ambiorics, IDO/99/003 and IDO/02/009 (Predictive computer models for medical classification problems using patient data and expert knowledge), several PhD/postdoc & fellow grants; **Flemish Government: FWO:** PhD/postdoc grants, projects, G.0078.01 (structured matrices), G.0269.02 (magnetic resonance spectroscopic imaging), G.0270.02 (nonlinear Lp approximation), G.0240.99 (multilinear algebra), G.0407.02 (support vector machines), G.0197.02 (power islands), G.0141.03 (Identification and cryptography), G.0491.03 (control for intensive care glycemia), G.0120.03 (QIT), G.0452.04 (QC), G.0499.04 (robust SVM), research communities (ICCoS, ANMMM, MLDM); AWI: Bil. Int. Collaboration Hungary/ Poland; **IWT:** PhD Grants; GBOU (McKnow) **Belgian Federal Government:** DWTC (IUAP IV-02 (1996-2001) and Belgian Federal Science Policy Office IUAP V-22 (2002-2006) (Dynamical Systems and Control: Computation, Identification & Modelling)); PODO-II (CP/01/40: TMS and Sustainibility); **EU:** PDT-COIL, BIOPATTERN, eTUMOUR, FP5-Quprodis; ERNSI; Eureka 2063-IMPACT; Eureka 2419-FliTE; Contract Research/agreements: ISMC/IPCOS, Data4s, TML, Elia, LMS, IPCOS, Mastercard; HPC-EUROPA (RII3-CT-2003-506079), with the support of the European Community—Research Infrastructure Action under the FP6 "Structuring the European Research Area" Program.

References

[1] B. Roorda and C. Heij, "Global total least squares modeling of multivariate time series," *IEEE Trans. on Aut. Control*, vol. 40, no. 1, pp. 50–63, 1995.

[2] I. Markovsky, J. C. Willems, S. Van Huffel, B. D. Moor, and R. Pintelon, "Application of structured total least squares for system identification," in *Proc. of the Conf. on Decision and Control*, Atlantis, Paradise Island, Bahamas, 2004, pp. 3382–3387.

[3] I. Markovsky, S. Van Huffel, and R. Pintelon, "Block-Toeplitz/Hankel structured total least squares," *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 4, pp. 1083–1099, 2005.

[4] I. Markovsky and S. Van Huffel, "High-performance numerical algorithms and software for structured total least squares," *J. of Comput. and Appl. Math.*, vol. 180, no. 2, pp. 311–331, 2005.

[5] I. Markovsky, J. C. Willems, S. V. Huffel, and B. De Moor, "Software for approximate linear system identification," Dept. EE, K.U.Leuven, Tech. Rep. 04–221, 2004.

[6] B. Roorda, "Global total least squares—a method for the construction of open approximate models from vector time series," Ph.D. dissertation, Tinbergen Institute Series No. 88, Thesis Publishers, 1995.

[7] G. Golub and C. Van Loan, "An analysis of the total least squares problem," *SIAM J. Numer. Anal.*, vol. 17, pp. 883–893, 1980.