THE UNIVERSITY OF NEWCASTLE IDENTIFICATION TOOLBOX (UNIT)

Brett Ninness [*,1] **Adrian Wills** [**] **Stuart Gibson** [***]

[*] *School of Elec. Eng. & Comp. Sci. University of Newcastle, Australia. email:*brett@ee.newcastle.edu.au*,*
FAX: +61 49 21 69 93
[**] *School of Elec. Eng. & Comp. Sci. University of Newcastle, Australia. email:*onyx@ee.newcastle.edu.au*,*
FAX: +61 49 21 69 93
[***] *Lehman Brothers, London, UK.*
*email:*stuart.gibson@lehman.com

Abstract: This paper describes a MATLAB-based software package for estimation of dynamic systems. A wide range of standard estimation approaches are supported. These include the use of non-parametric, subspace-based and prediction-error algorithms coupled (in the latter case) with either MIMO state space or MISO polynomial model structures. A key feature of the software is the implementation of several new techniques that have been investigated by the authors. These include the estimation of non-linear models, the use of non-standard model parametrizations, and the employment of Expectation Maximisation (EM) methods. *Copyright © 2005 IFAC.*

Keywords: Parameter Estimation, System Identification, Software.

1. INTRODUCTION

This paper describes the "University of Newcastle Identification Toolbox" (UNIT). This is software, usable as a MATLAB (Mathworks, 2004) toolbox, that implements a wide variety of algorithms for estimation of dynamic system models on the basis of observed operating records.

The essential features of this toolbox are as follows:

- It was developed by the authors in order to provide a unified platform to compare, contrast and profile various estimation approaches, both old and new;
- The authors believe that it has developed to a point where it may be useful to a wider group of researchers;
- In aid of the above, it is freely available under the GNU Public Licence (GPL) (GNU, 2004) which allows for further open source development by other interested parties;
- The work is inspired by the success, in allied research communities (numerical optimisation, numerical linear algebra), of promulgating research outcomes via non-proprietary software releases (Netlib, 2004).

Key features of the work as it stands are that the toolbox currently supports:

- Full multivariable system estimation via state space model structures;
- MISO estimation via both state space and polynomial form (ARMA,ARMAX, Output-Error, Box–Jenkins) model structures;
- State Space and Polynomial MISO estimation from frequency domain data via both state space and polynomial model structures;
- Recursive estimation of FIR and ARX model structures;

[1] This work was supported by the Australian Research Council.

- Subspace based estimation methods;
- Computation of error bounds both in parameter space and on associated frequency response estimates;
- Non-parametric estimation;
- Model validation;
- Kalman prediction, filtering and smoothing.

While these features would be considered 'standard', the toolbox also features certain non-standard embellishments which have arisen from the authors research activities, such as:

- The estimation of non-linear model structures including:
  (1) Bilinear systems;
  (2) MISO Hammerstein, Wiener and Wiener–Hammerstein systems.
- The use of non-standard optimisation methods including:
  (1) Expectation-Maximisation (EM) based methods;
  (2) Embellishments of standard gradient based search methods which hybridise recently developed Data Driven Local Co-Ordinate (DDLC) techniques (Wills and Ninness, 2004).
- The use of non-standard bases for model structures including:
  (1) Delta operators as an alternative to more standard shift q operators;
  (2) Rational orthonormal bases as an alternative to more standard FIR stuctures.

As already mentioned, it was the evaluation of these new techniques relative to more established methods that was the impetus for developing this first release of the toolbox.

We would welcome wider collaboration with other researchers to include further embellishments within the common framework established by this first release.

2. STANDARD UNDERLYING METHODS

We begin the description of the approach and algorithms underlying UNIT by considering the case of multiple-input $u_t \in \mathbf{R}^m$, single-output $y_t \in \mathbf{R}^1$ (MISO) data in which case, by default, the following polynomial form model structure is used,

$$y_t = G(q,\theta)u_t + H(q,\theta)e_t \tag{1}$$

where e_t is an i.i.d. zero mean noise sequence with variance $\mathbf{E}\left\{e_t^2\right\} = \sigma^2$, $G(q,\theta)$ is $1 \times m$ vector of scalar rational transfer functions $\{G_i(q,\theta)\}$ and $H(q,\theta)$ is also rational according to

$$G_i(q,\theta) = q^{-k_i}\frac{B_i(q,\theta)}{A_i(q,\theta)}, \quad H(q,\theta) = \frac{C(q,\theta)}{D(q,\theta)} \tag{2}$$

where

$$\begin{aligned} A_i(q,\theta) &= 1 + a_1 q^{-1} + a_2 q^{-1} + \cdots + a_{m_a} q^{-m_a^i}, \\ B_i(q,\theta) &= b_0 + b_1 q^{-1} + b_2 q^{-1} + \cdots + b_{m_b} q^{-m_b^i}, \\ D(q,\theta) &= 1 + d_1 q^{-1} + d_2 q^{-1} + \cdots + d_{m_d} q^{-m_d}, \\ C(q,\theta) &= 1 + c_1 q^{-1} + c_2 q^{-1} + \cdots + c_{m_c} q^{-m_c}, \end{aligned}$$

and k_i is a specified delay acting on the i'th input.

Since any of the orders m_a^i, m_b^i, m_c, m_d may be set to zero, then UNIT can implement any of the common FIR, ARX, ARMAX, ARMA, Output-Error and Box–Jenkins model structures(Ljung, 1999).

In all of these situations, the mean-square optimal one-step ahead predictor $\widehat{y}_t(\theta)$ based on the model structure (1) is (Ljung, 1999)

$$\widehat{y}_t(\theta) = H^{-1}(q,\theta)G(q,\theta)u_t + \left[1 - H^{-1}(q,\theta)\right] y_t$$

with associated prediction error

$$\varepsilon_t(\theta) \triangleq y_t - \widehat{y}_t(\theta) = H^{-1}(q,\theta)\left[y_t - G(q,\theta_n)u_t\right] \tag{3}$$

Using this, assuming that N samples of u_t and y_t are available, a quadratic estimation criterion may be defined as

$$V_N(\theta) = \frac{1}{2N}\sum_{t=1}^{N} \|\varepsilon_t(\theta)\|^2 \tag{4}$$

and then used to construct the prediction error estimate $\widehat{\theta}_N$ of θ as

$$\widehat{\theta}_N \triangleq \arg\min_{\theta \in \mathbf{R}^n} V_N(\theta). \tag{5}$$

This estimate is computed in UNIT via an iterative 'damped' Gauss–Newton strategy in which, at the $k+1$'st iteration, a new approximation θ_{k+1} for $\widehat{\theta}_N$ is found according to

$$\theta_{k+1} = \theta_k + \mu\, p. \tag{6}$$

Here, $\mu \in (0,1]$ is a damping parameter (step length), and p is a search direction defined as a solution of

$$V_N''(\theta_k)p = V_N'(\theta_k) \tag{7}$$

($\cdot'$ denotes differentiation with respect to θ). If no initial estimate is provided, then in the MISO case UNIT uses a modified Steiglitz–McBride method to initialise $G(q,\theta)$, and uses the Hannan-Rissanen method to initialise $H(q,\theta)$. In the MIMO case, the N4SID (van Overschee and de Moor, 1994) subspace based variant is employed for initialisation purposes.

Analysis of estimation error is performed by assuming that N is large enough that, with $\theta_\circ$ denoting the true underlying parameters

$$\widehat{\theta}_N \sim \mathcal{N}\left(\theta_\circ, \widehat{\sigma}^2[V_N''(\widehat{\theta}_N)]^{-1}\right), \; \widehat{\sigma}^2 = \frac{1}{N}\sum_{t=1}^{N}\varepsilon_t^2(\widehat{\theta}_N)$$

is a good approximation.

Finally, in this MISO situation, recursive (in time) estimation may be performed for FIR and ARX model structures. This amounts to the update equation (6) being used. With $\mu = 1$ and p chosen via (7), an RLS scheme is achieved, while with $V_N''(\theta_k)$ set to I and μ freely chosen, an LMS scheme is implemented. A Kalman-Filter based update is also possible by similar modifications.

In the case of multivariable data, state space model structures of the form

$$\begin{bmatrix} x_{t+1} \\ y_t \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \begin{bmatrix} K \\ I \end{bmatrix} e_t \qquad (8)$$

are employed. The parameters in this structure may then be estimated by a subspace based method, via any of the N4SID (van Overschee and de Moor, 1994), MOESP (Verhaegen, 1994) or CVA (Larimore, 1990) variants.

Alternatively, the model structure (8) may be estimated via the afore-mentioned prediction error methods, by defining the parameter vector as

$$\theta^T \triangleq \left[\text{vec}\{A\}^T, \cdots, \text{vec}\{K\}^T\right] \qquad (9)$$

in which case the predictor can be expressed according to

$$\widehat{x}_{t+1|t} = (A - KC)\widehat{x}_{t|t-1} + (B - KD)u_t + Ky_t$$
$$\widehat{y}_{t|t-1}(\theta) = C\widehat{x}_{t|t-1} + Du_t.$$

A potential difficulty in employing the Gauss–Newton search direction (7) is that, since (9) is an over-parametrization, $V_N''(\widehat{\theta}_N)$ is singular(McKelvey *et al.*, 2004). The standard method, implemented in UNIT, for handling this situation is to switch to a new DDLC parametrization β defined as (McKelvey *et al.*, 2004; Bergboer *et al.*, 2002)

$$\theta(\beta) = \theta + P\beta \qquad (10)$$

where, using the indicated singular value decomposition

$$P = V_1, \; V_N''(\theta) = [V_1, V_2] \begin{bmatrix} S_1 & \varnothing \\ \varnothing & \varnothing \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}.$$

However, certain new techniques described in (Wills and Ninness, 2004) which build on this idea in order to deliver enhanced performance are also available in UNIT.

In the case of frequency domain data, only SISO structures are supported, which may be described in either state space or polynomial form. In the latter case, denoting by $Y(\omega_k)$ the measured frequency response at frequency ω_k the model structure becomes

$$Y(\omega_k) = G(\gamma_k, \theta) + H(\gamma_k, \theta)E(\omega_k) \qquad (11)$$

where $G(\gamma_k, \theta), H(\gamma_k, \theta)$ are rational in γ_k, $E(\omega_k)$ is a complex i.i.dżero mean random process, and γ_k is either $e^{j\omega_k}$ or $j\omega_k$ depending on whether a discrete time or continuous time estimate is sought.

In this case, UNIT finds an estimate $\widehat{\theta}_N$ again via (5) and the Gauss–Newton method (6),(7), but now with respect to the cost function

$$V_N(\theta) = \frac{1}{N}\sum_{k=1}^{N} |Y(\omega_k) - G(\gamma_k, \theta)|^2 \, |H(\gamma_k, \theta)|^2. \qquad (12)$$

In fact, when estimating from frequency domain data, UNIT supports only the Output-Error case of $C(\gamma, \theta) = D(\gamma, \theta) = 1$ or the ARX case of $C(\gamma, \theta) = 1$, $D(\gamma, \theta) = A(\gamma, \theta)$. Furthermore, if a continuous time model is requested, then the polynomials in $\gamma = j\omega$ that are implied by the rational form are re-parametrised in terms of other polynomials (of more limited dynamic range) in order to improve numerical robustness.

When a state-space structure is employed with respect to frequency domain data, the model structure implemented by UNIT becomes

$$Y(\omega_k) = G(\gamma_k, \theta) + E(\omega_k) \qquad (13)$$
$$G(\gamma, \theta) = C(\gamma I - A)^{-1}B + D. \qquad (14)$$

This is then estimated by means of the subspace based methods espoused in (McKelvey *et al.*, 1996).

Finally, UNIT also supports the computation of non-parametric estimates via either the Blackman–Tukey or ETFE methods, and implements robust square-root versions of Kalman Filtering and Smoothing using routines that have been hand-coded in 'C' using the ATLAS libraries(Netlib, 2004) in order to deliver maximum performance.

3. EMBELLISHMENTS

As mentioned in the introduction, the motivation for developing UNIT was to serve as a platform for testing and evaluating new methods. These new techniques, once refined and proven, are now also part of the toolbox.

Perhaps the most significant of these is the option to replace the gradient-based search (6), (7) with one based on the Expectation-Maximisation (EM) algorithm. This applies only to state-space model structures, and involves the iteration (6) being replaced with

$$\theta_{k+1} = \arg\max_{\theta} \mathcal{Q}(\theta, \theta_k) \qquad (15)$$

where

$$\mathcal{Q}(\theta, \theta_k) \triangleq \mathbf{E}_{\theta_k}\{\log p_\theta(X, Y) \mid Y\}. \qquad (16)$$

Here $X = \{x_1, \cdots, x_N\}$ and $Y = \{y_1, \cdots, y_N\}$ are (respectively) the state and observed output histories, $p_\theta(X,Y)$ is the their joint probability density for some θ of the form (9), and $\mathbf{E}_{\theta_k}(\cdot)$ is expectation with respect to $p_{\theta_k}(X,Y)$.

There is not sufficient space to provide more detail than this here (see (Gibson and Ninness, 2004) for a fuller development), but suffice to say that it is straightforward to compute (16) via use of a Kalman Smoother, and then the maximisation step involved in the iteration (15) may be solved simply and robustly in closed form.

While, as argued in (Gibson and Ninness, 2004), this feature of UNIT offers a robust and efficient solution for linear model estimation that is an attractive alternative to the gradient based approach (6),(7), as argued in the further work (Gibson *et al.*, 2004) it also provides an effective means for estimating parameters in the non-linear model structure

$$\begin{bmatrix} x_{t+1} \\ y_t \end{bmatrix} = \begin{bmatrix} A & F & B \\ C & G & D \end{bmatrix} \begin{bmatrix} x_t \\ u_t \otimes x_t \\ u_t \end{bmatrix} + \begin{bmatrix} w_t \\ v_t \end{bmatrix} \quad (17)$$

where $\otimes$ denotes the Kronecker tensor product, so that (17) represents a bilinear system.

Within this field of non-linear system identification, a further embellishment implemented in UNIT is the ability to handle Hammerstein–Wiener MISO model structures of the form

$$y_t = Z(\alpha, z_t(\theta,\eta)) + H(q,\theta)e_t,$$

$$z_t(\theta,\eta) = G(q,\theta)x_t(\eta), \quad x_t(\eta) = X(\eta, u_t).$$

Here, $Z(\alpha, z_t)$ and $X(\eta, u_t)$ are memoryless nonlinearities parametrized by vectors α and η respectively. In this case, the estimation problem becomes one of

$$\widehat{\theta}_N, \widehat{\alpha}_N, \widehat{\eta}_N \triangleq \arg\min_{\theta,\alpha,\eta} V_N(\theta,\alpha,\eta). \quad (18)$$

where V_N is still of the form given by (4), (3) but now with

$$\varepsilon_t(\theta,\alpha,\eta) = H^{-1}(q,\theta)[y_t - Z(\alpha, z_t(\theta,\eta))].$$

Within UNIT, these memoryless non-linearities may be chosen as tailored functions such as deadzones or saturations, or more general ones described by polynomials of a given order, or as piecewise linear with an arbitrary number of nodes. While differentiation of the ensuing cost V_N with respect to the parametrization of these functions requires some care, it is still straightforward (Ninness and Gibson, 2002).

With regard to time domain data, the discussion so far has discussed models such as $G(q,\theta)$ which are expressed with respect to the shift operator q. However, it has been argued (Middleton and Goodwin, 1990) that in the interests of improved numerical conditioning, the Euler difference (or delta) operator

$$\delta \triangleq \frac{q-1}{\Delta} \quad (19)$$

(where Δ is the sampling period) should be employed. UNIT transparently supports the use of this operator for all MISO polynomial structure estimation. However, the shift operator is used as a default.

4. SOFTWARE DESCRIPTION

For the purposes of finding a system estimate from available data, a user of the toolbox need know only one command:

```
g=est(z,m,opt);
```

All arguments, and the returned output `g` are structure variables, which (typically) contain individual elements of mixed type.

To be more specific, the structure `z` defines the observed data, and must have element `z.y` and `z.u` which (respectively) are matrices of output and input data, with each column representing a different input or output, and each row a different time sample.

The structure `m` defines the model structure to be used. At a minimum, it must contain the entry `m.A` which, if an integer, defines a model order, and if a matrix/vector defines an estimate. There will be more on this point in a moment. The structure `opt`, defines optional specifications pertaining to the algorithm being used. If there are none the user wished to dictate, then `g=est(z,m)` is a legal command.

To make this concrete, consider the data set shown in figure 1 which was plotted according to

```
subplot(211); plot(y);
subplot(212); plot(u);
subplot(211); title('Output');
subplot(212); title('Input');
```

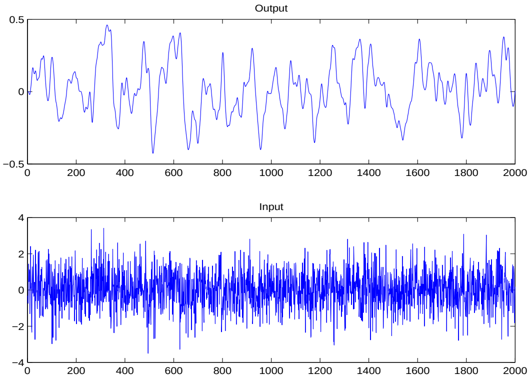


Fig. 1. *Observed Input/Output Data.*

Then the following sequence of commands will estimate a fifth order model from the data.

```
>> z.y=y; z.u=u; m.A=5;
>> g=est(z,m);
```

As mentioned, g is a structure, and in the above example it will contain 26 elements. In order to extract a summary of the properties of the estimate, the `details` command provides

```
>> details(g)
-------------------------------------
Details for Estimated Model Structure
-------------------------------------
Operator used in model          = q
Sampling Period                 = 1.000000 seconds
Estimated Innovations Variance = 5.059506e-03
Model Structure Used            = Output Error
Estimation algorithm            = Gauss-Newton search
Input #1 block type             = linear
Output block type               = linear
-------------------------------------

---------------------------------------------------------
Input #1 to Output #1 Estimated T/F model + standard devs:
---------------------------------------------------------
      1        q^-1    q^-2    q^-3    q^-4    q^-5
B =  0.0011 -0.0019  0.0031 -0.0005 -0.0029  0.0081
SD=  0.0016  0.0029  0.0029  0.0033  0.0033  0.0024

      1        q^-1    q^-2    q^-3    q^-4    q^-5
A =  1.0000 -1.4324 -0.5590  1.4602 -0.3570 -0.1047
SD=  0        0.2955  0.4923  0.1017  0.4801  0.2049

delay = 0 samples

Poles at 0.8372*exp(+-j0.0797), -0.9698, -0.1704, 0.9035.
```

This illustrates a main point. A philosophy underlying UNIT is that, in order to maximise utility for those inexperienced with the toolbox, defaults are used as opposed to issuing error messages. In particular, the above indicates that since only an order was specified, the operator type (q), sampling period, and model order type (Output Error) have all been set as defaults.

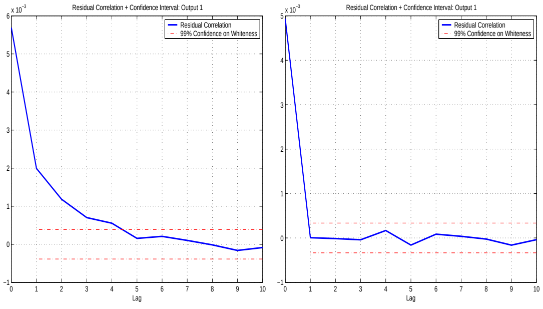
In order to assess the quality of this model, a standard model validation test may be performed in UNIT according to

```
validate(z,g);
```

which provides the sample-correlation of the error residuals as shown in figure 2(a). Clearly, there appears to be some undermodelling. In order to illustrate the specification of a Box–Jenkins structure, the following commands specify that a first order noise model should be added to the model structure (`m.D=1`), that the progress of the Gauss–Newton iterations should be displayed (`opt.dsp=1`) and that this new model should also be validated

```
>> m.D=1;
>> opt.dsp=1;
>> g1=est(z,m,opt);
Finding initial dynamics model via Steiglitz-McBride...
Finding initial noise Model by Hannan-Rissanen...
bisec# = 2,cost = 4.9459e-03,gn norm =4.9212e+00:G-N Direction
bisec# = 6,cost = 4.9448e-03,gn norm =2.2722e+01:G-N Direction
bisec# = 3,cost = 4.9392e-03,gn norm =1.1786e+00:G-N Direction
bisec# = 3,cost = 4.9374e-03,gn norm =7.0141e-02:G-N Direction
bisec# = 1,cost = 4.9361e-03,gn norm =5.4293e-02:G-N Direction
bisec# = 2,cost = 4.9358e-03,gn norm =9.6350e-01:G-N Direction
--------------------------------------------------
Termination due to relative cost decrease < OPT.mdec
--------------------------------------------------
>> validate(z,g1);
```

The ensuing validation results are shown in figure 2(b). Of course, in practise, data other than that used for estimation should be employed for this validation purpose, but we ignore this in order to streamline the presentation.



(a) OE Model (b) BJ Model

Fig. 2. *Model Validation*

To illustrate a more complex example, consider the MISO data shown in figure 3. This comes from a system with saturation and deadzone nonlinearities on inputs 1 and 2 respectively, that then pass through linear systems of order 4 and 3 respectively with the output then passing through a further deadzone non-linearity.

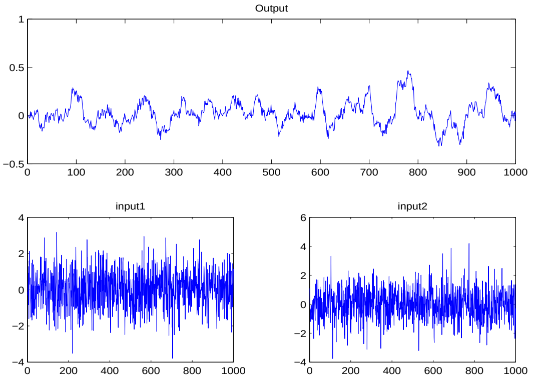


Fig. 3. *Observed MISO Input/Output Data from Hammerstein System.*

In order to estimate a model structure of this form we proceed as follows

```
>> z.y=y; z.u=u;
>> m.A=[4;3]; m.B=[3;2]; m.delay=[1;1];
>> m.in(1).type='hinge';
>> m.in(2).type='deadzone';
>> m.out.type='deadzone';
>> m.out.upper=0.1; m.out.lower=-0.1;
>> g=est(z,m);
```

Note that now two model orders are specified, which are different, for the two linear components associated with the two inputs. It is important that the different orders are specified as a column vector. If they had been a row vector, UNIT would interpret them not as orders, but an initial estimate of a polynomial denominator and/or numerator.

In the above, a time delay of one sample is specified for both of the inputs. Furthermore, it is specified that a piecewise linear structure should be used to model the non-linearity on the first input, while for the second, prior knowledge that it is of deadzone type is employed. Finally, a deadzone non-linearity is specified for the output, with initial estimates for the deadzone region as $[-0.1, 0.1]$.

As before, the results of this estimation experiment can be summarised via use of the `details` command.

```
>> details(g)
-------------------------------------
Details for Estimated Model Structure
-------------------------------------
Operator used in model         = q
Sampling Period                = 1.000000 seconds
Estimated Innovations Variance = 1.069568e-04
Model Structure Used           = Output Error
Estimation algorithm           = Gauss-Newton search
Input #1 block type            = hinge
Input #2 block type            = deadzone
Output block type              = deadzone
-------------------------------------

----------------------------------------------------------
Input #1 to Output #1 Estimated T/F model + standard devs:
----------------------------------------------------------
     1       q^-1    q^-2    q^-3
B =  0.0002  0.0001  0.0005  0.0004
SD=  0.0001  0.0001  0.0001  0.0002

     1       q^-1    q^-2    q^-3    q^-4
A =  1.0000 -1.4152 -0.3569  1.2182 -0.4318
SD=  0       0.2449  0.5575  0.4215  0.1074

delay = 1 samples

Poles at -0.9274, 0.6689, 0.7719, 0.9018.

----------------------------------------------------------
Input #2 to Output #1 Estimated T/F model + standard devs:
----------------------------------------------------------
     1       q^-1    q^-2
B =  0.0033  0.0007  0.0048
SD=  0.0006  0.0013  0.0012

     1       q^-1    q^-2    q^-3
A =  1.0000 -2.2436  1.6332 -0.3816
SD=  0       0.0402  0.0718  0.0321

delay = 1 samples

Poles at 0.5064, 0.8365, 0.9008.
----------------------------------------------------------
Input Non-linearity Parameters and standard deviations:
----------------------------------------------------------
Input block #1 of type hinge has estimates and standard dev:

eta =

   -6.1438    0.1417    7.5736  -12.8909    6.3845   12.8978
    0.1249    0.0875    0.1424    0.1980    0.1265    0.2156

Input block #2 of type deadzone has estimates and standard dev:

upper limit = 0.6064, sd = 0.0058
lower limit = -0.4900, sd = 0.0051

----------------------------------------------------------
Output Non-linearity Parameters and standard deviations:
----------------------------------------------------------
Output block of type deadzone has estimates and standard dev:

upper limit = 0.0618, sd = 0.0014
lower limit = -0.0398, sd = 0.0015
```

By way of information, the true noise variance was $\sigma^2 = 10^{-4}$, the true deadzone region on input 2 was (lower, then upper limit) $[-0.5, 0.6]$. Furthermore, the above hinge parametrization corresponds to a deadzone with limits $[-0.49, 0.59]$ while the underlying true one was $[-0.5, 0.6]$. There was no output non-linearity in the true system, which has essentially also been estimated in the model, even though it was allowed for.

There are further features of UNIT that have not been profiled by these brief examples but are still worth mentioning.

- An estimated model structure `g` may be used to specify the model structure `m` in cascaded estimation experiments;
- It is not necessary to use the `details` command to assess the features of an estimated model `g`, since the latter is simply a structure, whose elements will be echoed to the screen by typing it at the `>>` prompt;
- Frequency domain plots, complete with error bounds may be simply generated via the `showbode(g)` and `shownyq(g)` commands;
- A model structure may be forced by setting `m.type` to `fir`,`arx`,`armax`,`oe`, `bj` or `ss`;
- A delta operator parametrization may be specified by setting `m.op='d'`;
- The EM algorithm may be specified by setting `opt.alg='em'`. This and `m.type='ss'` will be used as a default if `z.y` contains multiple columns (outputs);
- Finally, online help for all functions is available. In particular `help est` gives a catalogue of all options available.

5. CONCLUSION

The UNIT software profiled here is intended as an open source platform to support further research developments in the science of system identification. It will co-exist without conflict in a MATLAB environment where the propriety system identification toolbox available for that product (Ljung, 2004) is installed. While there is overlap between the functionality offered by these two software packages, UNIT is intended mainly for researchers and experienced practitioners. It does not pretend to offer the same level of online help, error-checking and user guidance (for example) that are necessary in a package suitable for widespread industrial use. Having said that, an extensive GUI is also available for UNIT, but lack of space preclude a description of it here.

REFERENCES

Bergboer, N., Vincent Verdult and Michel H.G. Verhaegen (2002). An efficient implementation of Maximum Likelihood identification of LTI state-space models by local gradient search. In 'Proceedings of the 41st IEEE CDC, Las Vegas, USA'.

Gibson, S., Adrian Wills and Brett Ninness (2004). 'Maximum-likelihood parameter estimation of bilinear systems'. *Submitted to IEEE Transactions on Automatic Control.*

Gibson, S. and Brett Ninness (2004). 'Robust ML estimation of MIMO dynamic systems'. *To appear, Automatica.*

GNU (2004). 'Free software foundation'. www.gnu.org.

Larimore, W. (1990). Canon. variate analysis in identification, filtering and adaptive control. In 'Proc. 29th IEEE Conf. Decision and Control, Hawaii'. pp. 596–604.

Ljung, L. (1999). *System Identification: Theory for the User, (2nd edition).* Prentice-Hall, Inc.. New Jersey.

Ljung, L. (2004). *MATLAB System Identification Toolbox Users Guide, Version 6.* The Mathworks.

Mathworks (2004). *MATLAB Users Guide, Version 7.* The Mathworks.

McKelvey, T., A. Helmersson and T. Ribarits (2004). 'Data driven local coordinates for multivariable linear systems and their application to system identification'. *Automatica* **40**, 1629–1635.

McKelvey, T., Hüseyin Akçay and Lennart Ljung (1996). 'Subspace-based multivariable system identification from frequency response data'. *IEEE Transactions on Automatic Control* **41**, 960–979.

Middleton, R. and G.C. Goodwin (1990). *Digital Estimation and Control: A Unified Approach.* Prentice-Hall, Inc.. New Jersey.

Netlib (2004). 'The Netlib Repository'. www.netlib.org.

Ninness, B.& S. Gibson. 'Quantifying the accuracy of hammerstein model estimation'. *Automatica* **38**(12), 2037–2051.

van Overschee, P. and Bart de Moor (1994). 'N4SID:Subspace algortihms for the identification of combined deterministic-stochastic systems'. *Automatica* **30**(1), 75–93.

Verhaegen, M. (1994). 'Identification of the deterministic part of MIMO state space models in innovations form from input-output data'. *Automatica* **30**(1), 61–74.

Wills, A. and Brett Ninness (2004). On gradient-based search for multivariable system estimates. '2005 IFAC World Congress'.