Extended benchmark results for impulse response estimation

K Erik J Olofsson\textsuperscript{1,2}

Abstract—Recent attention has been given to the use of regularised least-squares methods for improving the quality of classical impulse response estimates from short/noisy datasets. This paper significantly augments and contextualises previously reported benchmark results by inclusion of a larger set of standard and non-standard methods in the comparison. The numerical experiments are summarised using several metrics. There is no obvious winner method.

I. INTRODUCTION

System identification deals with estimation of dynamical systems from measured input-output datasets [1] and may be seen as an inverse problem [2]. A typical assumption is that a linear time-invariant (LTI) system can approximate the recorded data. Parameterisation of this “ansatz” LTI system then enables a search for the “best” parameter vector value. This may make good sense if the parameterised class of systems can approximate the true underlying data-generating process “well enough”. Well-enough is dependent on the intended application of the model. Approximate modeling, with system identification providing a key set of methods, is important in e.g. the application of control theory [3].

This paper is concerned with benchmarking the accuracy of a catalogue of standard and nonstandard LTI system identification methods. The specific benchmark set, introduced in Chen et al. [4], is contrived to pose situations where the data record is short and noisy (decidedly finite-data/non-asymptotic cases). Both [5] and the sequel studies [6], [4], [7], [8], [9] appear to present a (very) strong case for the usage of least-squares (LS) regularisations based on a parameterised model of the “covariance” of the impulse response, either seen as a “smoothness” description and/or a stochastic “process”.

The purpose of this paper is to provide an extended collection of explorative numerical results for the prototypical LTI benchmark problems [4]. Several metrics and method rankings will be presented below. This extended benchmark is arguably important since similar experimental overviews are (too) rare in system identification research. The results and rankings could hence act as suggestions for interesting approaches to pursue further in detail. It is also the case that several of the evaluated methods are not commonly found in mainstream system identification research. The purpose is not to extrapolate a general winner method. In fact it will become clear that no single method is best for all subclasses of the benchmark problems in this study.

\(1\)EE School of Electrical Engineering, KTH Royal Institute of Technology, Stockholm, Sweden.
Corresponding author email: k.e.j.olofsson@gmail.com.

A set of 38 methods is let loose on the test suite of [4]. Section II details the essentials of the benchmark problems and the fitness metric. Section III lists key methods and concepts that act as building-blocks for the LTI estimation algorithms, which are catalogued in section IV. Results are found in section V. Conclusions are drawn in section VI.

II. THE BENCHMARK PROBLEM SET

A. Basics of the impulse response

This benchmark study compares estimation accuracy based on the impulse response of a single-input single-output (SISO) discrete-time LTI (DTLTI) system. The impulse response \(\{h_k\}\) of the system \(G(q)\) \((q^{-1}\) is the usual delay operator) can be defined as equal to the output \(y(k)\) of \(G(q)\), subject to noise-free conditions, with zero initial condition, for time-indices \(k = 1, 2, \ldots, +\infty\) given the input \(u(0) = 1\), \(u(k) = 0\), \(k \neq 0\). The impulse response \(\{h_k\}\) determines the properties of the DTLTI system and defines the output for an arbitrary input \(\{u(k)\}\) using the convolution summation (in this work it is always the case that \(h_0 = 0\))

\[
y(k) = G(q)u(k) = \sum_{j=1}^{\infty} h_j u(k - j) \tag{1}
\]

The impulse response is infinite in extent in general. For a stable system, the impulse response decays and approaches zero, \(h_j \rightarrow 0\) as \(j \rightarrow +\infty\). The slowest modes of the system \(G(q)\) determine the rate of this decay. The impulse response is in general not the most compact representation of a system.

If it is assumed that the output is observed subject to some random i.i.d. zero-mean Gaussian measurement error \(e(k)\) then the following process is obtained

\[
y(k) = G(q)u(k) + e(k) \tag{2}
\]

where \(e(k) \sim N(0, \sigma^2)\). This situation will correspond to the benchmark problems detailed in subsection II-B below. Truncation of the convolution implied by (2) at some lag-length \(m\) forms the basis of the finite-impulse-response (FIR) estimation approach:

\[
y(k) = \sum_{j=1}^{m} h_j u(k - j) + e(k) \tag{3}
\]

B. Benchmark plant collection

The revisited collection of randomised/categorised SISO plants/input-output data is declared in [4, Sect. 2] and was retrieved from [10]. The benchmark test bank was derived from 5000 calls to the \texttt{rss} function in \texttt{matlab} (Control System Toolbox, [11]) with subsequent time-discretisation. The order of each system is 30. All systems are stable. The systems are...
sorted into “fast” and “slow” types which are distinguished by the spectral radii of the DTLTI A-matrices. “Slow” systems are defined by $0.95 < \rho(A) < 1$, where $\rho(A)$ is the spectral radius operator. “Fast” systems are the remaining, with $\rho(A) \leq 0.95$. The signal-to-noise ratio (SNR) is defined in the usual sense: $\text{SNR} = \sqrt{\sum_k (y(k) - e(k))^2 / \sum_k e(k)^2}$. The full benchmark set is partitioned into four classes of problems:

(i) Abbreviated S1D1; “fast systems”, $\text{SNR} = 10$.
(ii) Abbreviated S2D1; “slow systems”, $\text{SNR} = 10$.
(iii) Abbreviated S1D2; “fast systems”, $\text{SNR} = 1$.
(iv) Abbreviated S2D2; “slow systems”, $\text{SNR} = 1$.

The high-SNR cases (i)-(ii) provide datasets with length $N = 500$. The low-SNR cases (iii)-(iv) provide datasets with length $N = 375$. Each problem class (i)-(iv) provides one realisation for each system. This means there are 2500 sets of input-output data for each problem class. Hence the total number of estimation problems is $10 \times 10^3$.

It seems important to note the following:

- The input has been generated by realisations of white noise. This may restrict the extrapolation of the ranking results to be presented below to more general cases.
- This is a SISO benchmarking case. Some of the methods below are easy to generalise to MIMO. Other methods are not trivial to extend.
- The noise in this benchmark set is additive-on-output, i.e. equation (2). Some of the methods are formulated more generally and do not utilise this given fact.

C. Impulse response estimation

The problem setting is to estimate the impulse response of the systems as accurately as possible. The fitness of an estimate $\{\hat{h}_j\}_{j=1}^l$ is evaluated based on the score used in [4]:

$$W_l = 100 \times \left[ 1 - \frac{\sum_{j=1}^l (\hat{h}_j - h_0)^2}{\sum_{j=1}^l (h_j - \bar{h}_0)^2} \right]$$

(4)

where $\{h_0\}$ is the true impulse response and $\bar{h}_0 = (1/l) \sum_{j=0}^l h_0$, Scalar $W_l$ quantifies a fitness score for the first $l$ impulse response coefficients [4]. In this work (i) $l = 125$ will be used for accuracy evaluation (ii) whenever $W_l \leq 0$, the estimation will be regarded as a “failure”.

III. OVERVIEW OF ESTIMATION TOOLS

A. Least-squares (LS) estimation

The direct LS estimate of the $m$-truncated FIR coefficients are given by posing the observation equations from (3):

$$d = Kf + \eta$$

(5)

following the notation of [2]. Let $n = N - m$. For the FIR case, $d, \eta \in \mathbb{R}^{n \times 1}$, $f \in \mathbb{R}^{m \times 1}$, and $K \in \mathbb{R}^{n \times m}$. The element structures are

$$d^T = (y(m + 1) \ldots y(N))$$

$$K^T = (u_m(m + 1) \ldots u_m(N))$$

$$u_m^T(k) = (u(k - 1) \ldots u(k - m))$$

$$f^T = (h_1 h_2 \ldots h_m)$$

(6a)

(6b)

(6c)

(6d)

and the solution is provided by

$$\hat{f} = (K^T K)^{-1} K^T d$$

(7)

B. Regularised least-squares estimation

A useful class of modifications to (7) is defined by

$$\hat{f}_{\alpha,L} = (K^T K + \alpha L)^{-1} K^T d$$

(8)

Standard Tikhonov regularisation corresponds to $L = I$ in (8). There are several prototype choices of the penalty “operator” $L$. Basic non-smoothness-regularisation can be found in e.g. [12, Ch. 18]. For example, if the solution is supposed to be “close” to a vector with constant elements, $L$ can be constructed as $L = D^T D_1$ from a finite-difference matrix $D_1 \in \mathbb{R}^{(m-1) \times m}$ where

$$D_1 = \begin{pmatrix} -1 & 1 & 0 & \ldots & 0 \\ 0 & -1 & 1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & -1 \end{pmatrix}$$

(9)

The optimal choice of $L$ (and $\alpha$) depends on the true solution, which is unknown. Whenever (robust) prior knowledge can be modeled into $L$, this may improve the estimation accuracy. Also a stochastic view of the estimation problem (5) naturally leads to regularisation. If $f$ and $\eta$ both are regarded as mutually independent zero-mean random vectors with respective covariances $R_f$ and $R_\eta$, then the minimum variance estimate $\hat{f}$ of $f$ given the data $d$ becomes

$$\hat{f}_{MV} = R_f K^T (K_R f K^T + R_\eta)^{-1} d$$

(10a)

$$= (K^T R_\eta^{-1} K + R_f)^{-1} K^T R_\eta^{-1} d$$

(10b)

which is equivalent to (8) for the specific prescription $R_\eta = \sigma^2 I$, $\sigma^2 R_f^{-1} = \alpha L$.

C. Generalised cross-validation (GCV)

For the observational model (5) and the solution ansatz (8) a popular method for selection of $\alpha \geq 0$ is to use the minimising argument of the generalised cross-validation (GCV) function [13]. Following the notations of [2, Ch. 7] define:

$$\text{GCV}(\alpha) = \frac{(1/n) r_\alpha^T r_\alpha}{(\text{trace}(I - A_\alpha))^2}$$

(11)

where $r_\alpha = K \hat{f}_\alpha - d$, $\hat{f}_\alpha = Q_\alpha d$, $A_\alpha = K Q_\alpha$ and $Q_\alpha = (K^T K + \alpha L)^{-1} K^T$. It is understood, in the case of (8), that these quantities depends on some underlying $L$. The same expression (11) can also be used in selecting the number of terms to include in truncated SVD solutions (TSVD) to (5). The TSVD approach is based on the SVD of $K$. See [2] for
details. Note that one could insert the estimator from (10) as \(Q_\alpha\) for the GCV evaluation; hence \(\alpha\) in (11) may be seen as a placeholder symbol for some generic linear estimator.

\[E. \text{The linear one-step-ahead predictor}\]

The linear predictor is usually known as the autoregressive-exogenous (ARX) model. It is very efficiently estimated by LS and may model infinite impulse responses with few parameters. The basic equations can be modeled into the same form as the (originally) FIR-formulated equation (5) as detailed below. The lag-\(m\) predictor is defined by

\[\hat{y}(k) = \sum_{j=1}^{m} a_j y(k-j) + \sum_{j=1}^{m} b_j u(k-j) \quad (13)\]

with the coefficients determined such that a norm of the innovations process \(e(k) = y(k) - \hat{y}(k)\) is minimised. The corresponding linear equation can be cast as (5) with

\[K = \begin{pmatrix} u_{m}^T(m+1) & y_{m}^T(m+1) \\ u_{m}^T(m+2) & y_{m}^T(m+2) \\ \vdots & \vdots \\ u_{m}^T(N) & y_{m}^T(N) \end{pmatrix} \quad (14a)\]

\[y_{m}^T(k) = (y(k-1), \ldots, y(k-m)) \quad (14b)\]

\[f^T = (b_m^T a_m^T) \quad (14c)\]

\[a_m = (a_1 a_2 \ldots a_m) \quad (14d)\]

\[b_m = (b_1 b_2 \ldots b_m) \quad (14e)\]

Note that when applying regularisation techniques to the ARX equation, the partitioned structure of the solution vector \(f \in \mathbb{R}^{2m \times 1}\) ought to be taken into account.

\[F. \text{Subspace system identification methods (SIMs)}\]

Three SIMs will be benchmarked in this work: (i) standard N4SID method/routine as implemented in the System Identification Toolbox in \texttt{matlab}, (ii) SSARX introduced in [14] and (iii) the DSRe method [15]. Methods (ii) and (iii) are based on a first step that employs an ARX predictor equation. Method (ii) uses canonical correlation analysis to project a state sequence [16]. Methods (ii) and (iii) can both be used to identify plants operating under feedback and return a general innovations-model \((A, B, C, K)\):

\[x(k+1) = Ax(k) + Bu(k) + Ke(k) \quad (15a)\]

\[y(k) = Cx(k) + e(k) \quad (15b)\]

and an optional direct feedthrough \(D\) if requested (not relevant for the present benchmark). Specifically (and importantly), these methods are here tried out generically such that a noise model is simultaneously implied by the \(K\)-matrix (which is not assumed to be zero). The SIM implementations tested are fully MIMO capable.

\[G. \text{Empirical Bayes and hyper-parameters}\]

Consider (10). The basic idea here is (i) parameterise the “covariances” \(R_f\) and \(R_n\) and (ii) estimate these parameters using a maximum-likelihood (ML) argument and (iii) usage of (10) to find the impulse response. These specific parameters are called hyper-parameters. Let the hyper-parameters be denoted by the vector \(\beta\). Define

\[S(\beta) = R_n(\beta) + KR_f(\beta)K^T \quad (16)\]

then the ML hyper-parameter value should solve

\[\hat{\beta} = \arg \min_{\beta} \{d^TS^{-1}d + \ln \det S\} \quad (17)\]

which is a nonconvex programming problem. Two particular hyper-parameterisations are included in the benchmark [4], [17]: (i) \textbf{DC}: A DTLTI-inspired “diagonal-correlated” kernel. Defined by the matrix elements \(R_{k-j}(\beta) = \exp(-j(k-j)^2/\lambda^2)\). The hyper-parameters are \(c\), \(\rho\) and \(\lambda\), and (ii) \textbf{SE}: “squared-exponential” kernel defined by the matrix elements \(R_{k-j}(\beta) = c \exp\{-(k-j)^2/(2\lambda^2)\}\). Its hyper-parameters are \(c\) and \(\lambda\). The reason for using the DC kernel is that it is reportedly one of the best performing methods for this benchmark. The implementation for the DC kernel recently became available in \texttt{matlab} (R2012b) [11] (the function name is \texttt{impulseest}). The SE kernel is included as a representative of generic (non-LTI-specific) kernels.

\[H. \text{Nonparametric sub- and resampling}\]

One possible approach for obtaining a “low-resolution” prior covariance for the minimum-variance estimator is to perform randomised downsampling of the solution vector \(f\) exemplified as follows (for a subsampling of order 2). Draw a uniformly random integer \(z\) on the interval \([1, l]\). Introduce the vectors \(w_1\) and \(w_2\) where \(w_1\) has ones at elements \(i = 1 \ldots z\) and \(w_2\) has ones at elements \(i = (z+1) \ldots l\). The rest of the elements are zeros. Let \(W = (w_1 \ w_2)\). Let \(f = Wc\) in (5) and solve for \(c\) using LS. Repeat this
randomisation many times and aggregate the “covariance” $R_{\text{fsub}}$ of the “downsampled” vectors $\hat{\mathbf{f}} = W \hat{\mathbf{c}}$. The selection of either the regulariser $R_{\text{fsub}}$ or its diagonal is based on evaluation of the GCV value \((11)\) in conjunction with \((10)\). Finally, an adjustment of the $\hat{a}$-value is made based on GCV. By subsampling with higher order (i.e. $c$ of dimension higher than two) faster systems may be handled more accurately.

One may also attempt to reduce variance via bootstrap resampling/aggregation (a possible DTLTI-instance of “bagging” \([18]\)): (i) draw synthetic data from a high-order SSARX-estimated model on innovations form \((15)\) (ii) re-sample impulse responses using SSARX (iii) use the replicated impulse responses $\hat{\mathbf{f}}$ to approximate the “prior covariance” of $\mathbf{f}$: $R_{\text{f}}^{\text{rep}}$ (iv) apply the minimum-variance estimator \((10)\).

I. Sparse regression with a modal dictionary

From recent developments in sparse methods and compressive sampling it is known that if a signal has a sparse representation in some “basis” it is possible to recover it using a small number of randomised projections \([19]\), \([20]\). The benchmark problem can be cast as a sparse-recovery-like problem by postulating an expansion of the impulse response as a sum of atomic modal system impulses (these are decaying oscillations of various frequencies and damping-rates).

In particular, consider the problem \((5)\) and now let $\mathbf{f} = D \mathbf{c}$ where each column of $D$ is a normalised impulse response of some modal subsystem. The dictionary $D$ consists of several thousands of such modal impulses (enumerated according to some scheme). This means the estimation situation is very different from all other approaches above since $\dim c \gg \dim f$. However, only a small subset of the entries of $c$ become nonzero in this $\ell_1$-penalised problem

$$\hat{c} = \arg \min_c \| \mathbf{d} - \Phi c \|_2^2 + \lambda \| c \|_1 \quad (18)$$

for some $\lambda > 0$ (there are numerous similar ways of approaching this) where $\Phi = KD$ is a very fat matrix. The impulse reconstruction is then $\hat{\mathbf{f}} = D \hat{\mathbf{c}}$.

A rudimentary implementation has been attempted (using (i) a grid of pole radii and frequencies and (ii) randomised holdout cross-validation for selecting the number of terms to keep). The implementation for \((18)\) of \([21]\) (based on least-angle regression \([22]\)) was used to solve all the sparse programs.

IV. CATALOGUE OF COMPETING METHODS

The following mnemonics are used when presenting the results in section V below. The methods are composed of the various techniques presented in section III above.

1) PEM-em: Standard prediction-error method (PEM) matlab approach for output-error model structures \((2)\). A fixed-order call of the oe function in the System Identification Toolbox with orders \([5,5,1]\).
2) PEM-oem: A "textbook approach". See PEM-em. Two-fold cross validation to determine the "best" order $n \in [1,30]$ for the oe function with parameters \([n,n,1]\). This is more rigorous than is the case in \([4]\) (but it is very time-consuming).
3) FIR-name: Non-regularised LS FIR estimation \((7)\).
4) FIR-tsvd: Truncated SVD solution of \((5)\). Truncation selected by GCV.
5) FIR-0th: Classical Tikhonov regularisation. $L = I$ in \((8)\). Selection of $\alpha$ by GCV.
6) FIR-1st: Regularisation matrix that penalises the first discretised derivative of the solution vector. Use $L = D_1^2 D_1$, with $D_1$ given by \((9)\). Selection of $\alpha$ by GCV.
7) FIR-2nd: Regularisation matrix that penalises the second discretised derivative of the solution vector. See \([12]\) for structure of $L$. Selection of $\alpha$ by GCV.
8) FIR-3rd: Regularisation matrix that penalises the third discretised derivative of the solution vector. See \([12]\) for structure of $L$. Selection of $\alpha$ by GCV.
9) FIR-chb-1st: Estimation using a baseline impulse response obtained using the first $n_c = 20$ Chebyshev polynomial basis functions. The perturbation equation is solved as FIR-1st.
10) FIR-chb-2nd: Estimation using a baseline impulse response based on a two-state model obtained from SSARX. The perturbed equation is solved as FIR-1st.
11) FIR-bt: Non-regularised LS solution of the FIR coefficients $\text{FIR-none}$, followed by balanced truncation to a state-space order $n = 15$.
12) FIR-reg: FIR estimation using the SE kernel (subsection III-G).
13) FIR-reg/dc: FIR estimation using the DC kernel (subsection III-G).
14) FIR-reg: Estimation using the DC kernel (subsection III-G). Pads the data with zeros to emulate knowledge of the zero initial condition (i.e. uses more prior information).
15) B&G-gcv: Standard Backus-Gilbert method, using GCV to select $\alpha$. Due to space constraints the reader is referred to the treatments in e.g. \([23]\), \([12]\).
16) ARX-cv: This may be considered as a classical and conventional technique. Data is grouped in testing and training sets. ARX models of increasing lag orders $m = 1 \ldots 30$ are fitted using LS. The lag order yielding the smallest testing error is selected as the final model.
17) ARX-chb: Same regularisation as for FIR-1st but applied to the solution of the ARX predictor of fixed lag $m = 30$.
18) ARX-2nd: Same regularisation as for FIR-2nd but applied to the solution of the ARX predictor of fixed lag $m = 30$.
19) ARX-reg: Usage of the SE kernel described in subsection III-G applied to the ARX predictor equation with lag $m = 40$.
20) ARX-reg/dc: Usage of the DC kernel described in subsection III-G applied to the ARX predictor equation with lag $m = 40$. Based on the code behind $\text{FIR-reg/dc}$.
21) ARX-cv-bt: Starts by running ARX-cv. Then balanced truncation is applied to the implied predictor state-space model. The state-space truncation is set to $n = 15$.
22) HOARX-no: A long (high-order) ARX model (HOARX) with order $m = 50$ is fitted with LS.
23) HOARX-tsvd: A HOARX model with lag order $m = 50$ is fitted using Tsvd and GCV.
24) HOARX-0th: A HOARX model with lag order $m = 50$ is fitted using Tikhonov regularisation $L = I$ and GCV selection of $\alpha$.
25) HOARX-bt: A HOARX model with lag order $m = 50$ is fitted using non-regularised LS. Then balanced truncation is applied to the implied predictor state-space model. The state-space truncation is set to $n = 15$.
26) NSID: Plain application of the $\text{n4sid}$ SIM function in matlab using fully automatic options.
27) SSARX: Plain application of the SSARX SIM. Using a “one-size-fits-all” future horizon $f = 25$, past horizon $p = 25$ and a pre-impulse step with lag $q = 25$. Truncates the state space model to $n = 15$ states.
28) DSRe: Plain application of the $\text{DSRe}$ SIM using a “one-size-fits-all” future horizon $f = 25$, past horizon $p = 25$. Truncates the state space model to $n = 15$ states.
29) SSARX-ave: Application of the SSARX SIM: $f = 25$, $p = 25$, $q = 25$. Employs the SVC criterion to select the system order $n = 24$.
30) SSARX-knee: The SSARX SIM: $f = 25$, $p = 25$, $q = 25$. Selects system order $n$ by finding the first $n$ that fulfills $\ln n \leq (1/2) \ln \left( \max(\sigma) + \ln \left( \min(\sigma) \right) \right)$ for the singular value sequence $\sigma_n$.
31) SSARX-cv: Similar to SSARX but the selection of the state-space order $n \in [10,30]$ is determined by 3-fold cross-validation.
32) SSARX-ave: This approach is inspired by the "stabilisation diagram" which is practical of use in experimental modal analysis (EMA). Obtain, from SSARX, a sequence of state-space systems with state-dimensions $n = 10 \ldots 30$. Compute the impulse response for each $n$. Return the average impulse response.
33) SIM-sub: Randomly “subsampled” nonparametric realisation of the FIR model. See section III-H. Using 500 subsamples each for a range of subsampling orders $2 \ldots 6$.
34) SIM-sub: Randomly “subsampled” nonparametric realisation of an ARX model with lag $m_n = 50$. See section III-H. Using 500 subsamples. Subsampling orders $2 \ldots 6$.
35) SIM-bag: The bootstrapped SIM approach outlined in the last paragraph of subsection III-H, using aggregation of 100 synthetically redrawn datasets. The data generating process is pre-estimated using a high order SSARX application with $f = p = 40$, $q = 45$, and state-dimension $n_m = 30$.
36) L1-modal: Sparse regression approach with a modal impulse response dictionary $D$ with a total of 5951 columns (see subsection III-I) and a simple cross-validation approach to select the number of nonzero components to use (maximum terms to keep is limited to 40).
37) L1-modal: Same as L1-modal but padding the data with zeros to emulate prior knowledge of the zero initial condition.
38) FIR-opt: Non-feasible near-optimal LS regularisation. Estimates the noise variance using the LS equation but has access to the true impulse response
to form the regularisation matrix [7].

V. BENCHMARK RESULTS

Figures 1, 2, and 3 show respectively the mean ranking and score, the failed-case ranking and count, and the estimation rate ranking for all the methods outlined in section IV. There is no guarantee that a given method is implemented in the most efficient and robust manner. The numerical values of the estimation rates are dependent on the host computer hardware and load (all reported computations were performed using the same computer).

Notice that the ranking of methods for the mean score is quite different from the ranking for fewest failed cases. E.g. FIR-sub scores high on the S2D2 dataset but underperforms on the fast systems. There are orders of magnitude in difference in the execution speed of the algorithms as seen in Figure 3. The point of Figure 3 is the logarithmic scale. Only the infeasible FIR-opt beats the score-per-time rate achieved by FIR-none (which is a very poor method for this benchmark).

In Figure 4, an outlier-protected method ranking is presented. The plot summarises scoring distributions with lines connecting the quantiles [1/10, 9/10] for each of the benchmark cases, S1D1 (blue), S2D1 (green), S1D2 (red), and S2D2 (cyan). The total suite mean of the scores within these quantile brackets is marked with a black cross. Figure 4 suggests that the FIR-reg/dc method (which arguably represents state-of-the-art in [9]) meets respectable competition from both (i) wildly different methods (SIM-bag, L1-modal0) and (ii) variations on the same empirical-bayes programming (ARX-reg/dc, FIR0-reg/dc). Specifically FIR0-reg/dc implies that initial condition estimation can be significant. Note that a robust joint selection of both SIM horizons and order is yet to be demonstrated (the convention for setting the SIM horizons in this study has been to cut these at a fraction \( \sim 1/10 \) of the dataset length to enable reasonable statistics for the implied covariances, which may work well if the horizon covers a slow time-constant). Note also that some of the benchmark problems seem extremely difficult for any method (Figure 2).

VI. CONCLUSIONS

The outcome of these assorted numerical experiments, as summarised in section V, corroborates the (not surprising) claim that regularised LS is a highly competitive method for short and noisy data records. But there are strong competi-
The effectiveness of SIMs on the benchmark is important to emphasise (certain flavours of SSARX scores highest on the S1D1 suite). There seems further to be untapped potential in resampling techniques. Dictionary creation for impulse basis pursuit appears significantly underdeveloped. For example, it could be possible to use online dictionary learning techniques to adapt the sparse estimator to the underlying statistical properties of the benchmark (if the method is given persistent memory access and this is not considered “cheating”). Remarkably, the naive modal dictionary used in this work shows performance comparable to the best regularised methods.

No single method “wins” for all combinations of SNR and system types. The method of choice would seem to depend on the metric one intends to prioritise. Note also that many of the best methods are actually using the “wrong” model structure. For the purpose of plausible assessment of these issues, randomised (and specialised) benchmarking need to become standardised and more central in system identification research.

ACKNOWLEDGEMENTS

The author is very much grateful for substantial constructive help provided by T. Chen and L. Ljung in discussing ideas, preparing the manuscript and for access to reliable kernel implementations. The author would like to thank C. Rojas, H. Hjalmarsson, M. Jansson and P. Valenzuela for useful discussions. The DSRe implementation was provided by D. Di Ruscio. J. Höök administered the computational resources.

REFERENCES