Modeling, Identification and Control, Vol. 43, No. 4, 2022, pp. 119–130, ISSN 1890–1328

# On subspace system identification methods

David Di Ruscio [1] Christer Dalen [2]

[1] *University of South-Eastern Norway, P.O. Box 203, N-3901 Porsgrunn, Norway. E-mail:* David.Di.Ruscio@usn.no

[2] *Skien, Norway. E-mail:* christerdalen@hotmail.com

**Abstract**

An open and closed loop subspace system identification algorithm DSR_e is compared to competitive open loop algorithms, DSR, and N4SID. Additionally, DSR_e is compared vs the optimal Prediction Error Method (PEM). Monte Carlo simulations with discrete random state space models are used for testing the subspace identification algorithms in the numerical simulation section.

*Keywords:* System identification, subspace methods, stochastic systems, monte carlo

## 1. Introduction

The research about building linear dynamic models from observed input and output data in the early '90s resulted in the N4SID algorithm in the work by Overschee and de Moor (1994); Di Ruscio (1996). N4SID was also implemented as a numerical function in the System Identification Toolbox for MATLAB (MATLAB (2020)). At the same time and in earlier years Di Ruscio (1994, 1995) worked with the same problem and published the DSR algorithm in Di Ruscio (1996). An overview of some other related works are given in Wang and Qin (2002) and Qin et al. (2005) where the DSR method is referred.

It was soon observed that these methods did not work completely for systems with feedback in the output data. Due to noise and feedback, there could be a bias in the estimates e.g. from the DSR/N4SID algorithms. For about ten years there was great research on solving the problem, i.e. solving the problem of subspace system identification of input and output data from closed loop systems. In the early 2000 s the problem was solved with the DSR_e algorithm. The DSR_e method was presented in Di Ruscio (2008); Di Ruscio (2009b). Note that, the DSR_e algorithm was published earlier in internal notes as documented in the Ph.D. thesis by Nilsen (2005). DSR_e is a simple subspace system identification method for closed as well as for open loop systems. This is a two-stage method where the innovation process is identified consistently in the 1st step, e.g. by filtering the output data into a signal and an innovations noise part. An overview of closed loop subspace identification is given by van der Veen et al. (2013). The CL_MOESP algorithm is in our view not a pure subspace system identification method because the deterministic part of the state space model and the Kalman filter gain are obtained by a prediction error method. Hence, the CL_MOESP algorithm is not considered further.

As mentioned above it was well known that the DSR_e algorithm also worked for open loop systems, but, recent works show that in most situations the DSR_e method performs better on open loop systems than the DSR algorithm, and even better on open loop systems than the N4SID algorithm. The aim of this paper is to document the performance of the DSR_e, DSR, and N4SID algorithms on open loop systems.

The contributions in this paper may be itemized as follows:

- Monte Carlo simulations using random state space models are used for comparing the subspace system identification algorithms DSR_e, DSR, N4SID, and the optimal Prediction Error Methods (PEM) (Ljung (1999)).
- The algorithms DSR_e, DSR, and the optimal

doi:10.4173/mic.2022.4.1

PEM have been applied on a JAS 39 Gripen fighter aircraft case inspired by the paper of Ljung (2013).

All numerical calculations and plotting facilities are provided by using the MATLAB software (MATLAB (2020)).

In Section 2.1 the system definitions are given. In Section 3 the subspace identification methods are presented. In Section 4 the numerical examples are given. In Section 5 the concluding and discussion remarks are given.

# 2. Theory

## 2.1. System Definition

We will restrict ourselves to linearized or linear state space dynamic models of the form

$$x_{k+1} = Ax_k + Bu_k + Ce_k, \quad (1)$$
$$y_k = Dx_k + Eu_k + Fe_k, \quad (2)$$

with $x_0$ as the initial predicted state and where a series of $N$ input and output data vectors $u_k$ and $y_k$ $\forall\ k = 0, 1, \ldots, N-1$ are known, and where there is possible feedback in the input data. In case of output feedback the feed through matrix is zero, i.e. $E = 0$. Also for open loop systems the feed through matrix may be zero. We will include a structure parameter $g = 0$ in case of feedback data or for open loop systems in which $E = 0$, and $g = 1$ for open loop systems when $E$ is to be estimated. Furthermore, for the innovations model (1) and (2) $e_k$ is white with unit covariance matrix, i.e. $\mathrm{E}(e_k e_k^T) = I$ .

Note that, corresponding to the model (1) and (2) on innovations form we may, if the system is not pure deterministic, define the more common Kalman filter on innovations form by defining the innovations as $\varepsilon_k = Fe_k$ and then $K = CF^{-1}$ is the Kalman filter gain. Hence, the Kalman filter on innovations form is defined as

$$x_{k+1} = Ax_k + Bu_k + K\varepsilon_k, \quad (3)$$
$$y_k = Dx_k + Eu_k + \varepsilon_k, \quad (4)$$

where the innovations process $\varepsilon_k$ have covariance matrix $\mathrm{E}(\varepsilon_k \varepsilon_k^T) = FF^T$.

The quintuple system matrices $(A, B, C, D, E, F)$ and the Kalman filter gain $K$ are of appropriate dimensions. The problem addressed in this paper is to determine these matrices from the known data. Both closed and open loop systems are addressed.

# 3. The DSR_e algorithm

In Di Ruscio (2008) a very simple, efficient subspace system identification algorithm that works for both open as well as for closed loop data was presented. This algorithm was developed earlier and presented in an internal report (2004) and used in Nilsen (2005). In this section, an improved and extended presentation of the algorithm is presented.

The following matrix equation is of fundamental importance in connection with subspace system identification algorithms, i.e.

$$Y_{J|L} = O_L X_J + H_L^d U_{J|L+g-1} + H_L^s E_{J|L}, \quad (5)$$

where the matrices in Eq. (5) are defined in Appendix A and B. One problem in case of closed loop subspace identification is that the future inputs, $U_{J|L+g-1}$ in the matrix Eq. (5) are correlated with the future innovations, in matrix $E_{J|L}$. However, a way of overcoming this is to put $L = 1$ in Eq. (5) and for closed loop systems it makes sense to put $g = 0$, i.e. no direct feed through matrix and $E = 0$ in the output Eq. (4). Hence we have

$$Y_{J|1} = \overbrace{DX_J}^{y^d_{J|1}} + \overbrace{FE_{J|1}}^{\varepsilon_{J|1}}. \quad (6)$$

In the DSR_e algorithm the signal content, $y^d_{J|1} = DX_J$, in Eq. (6), of the future data, $Y_{J|1} = \begin{bmatrix} y_J & y_{J+1} & \cdots & y_{N-1} \end{bmatrix} \in \mathbb{R}^{m\times(N-J)}$, is estimated by projecting the "past" onto the "future", and in case of closed loop data when the direct feed through term is zero ($E = 0$), i.e. estimated by the following projection

$$y^d_{J|1} = DX_J = Y_{J|1} / \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}, \quad (7)$$

where the projection operator "/" is defined in Eq. (39), and where we have used that for large $J$ or as $J \to \infty$ we have that

$$X_J / \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} = X_J, \quad (8)$$

which may be proved from Eq. (43) by using Eq. (40).

The past data matrices, $U_{0|J}$ and $Y_{0|J}$, are uncorrelated with the future innovations sequence, $E_{J|1}$. In the same stage the innovations sequence $\varepsilon_{J|1} = FE_{J|1}$ in Eq. (6) is then consistently estimated as

$$\varepsilon_{J|1} = FE_{J|1} = Y_{J|1} - Y_{J|1} / \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}. \quad (9)$$

Note that, both the signal part, $y^d_{J|1}$, and the innovation part, $\varepsilon_{J|1}$, are used in the **dsr_e** algorithm.

For open loop systems we may have a direct feed through term in the output equation, i.e., $E \neq 0$ and with an output Eq. (2), $y_k = Dx_k + Eu_k + Fe_k$ we obtain

$$Y_{J|1} = \overbrace{DX_J + EU_{J|1}}^{y^d_{J|1}} + \overbrace{FE_{J|1}}^{\varepsilon_{J|1}}, \tag{10}$$

and the signal parts may in this case be computed as,

$$y^d_{J|1} = DX_J + EU_{J|1} = Y_{J|1}/\begin{bmatrix} U_{J|1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}, \tag{11}$$

where we have used Eq. (40). Furthermore, the innovations are determined by

$$\varepsilon_{J|1} = FE_{J|1} = Y_{J|1} - Y_{J|1}/\begin{bmatrix} U_{J|1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}. \tag{12}$$

The algorithm is a two step algorithm where in the 1st step the output data are filtered or split into a signal part, $y^d_{J|1}$, and a noise (innovations) part, $\varepsilon_{J|1} = FE_{J|1}$, i.e., as

$$y_{J|1} = y^d_{J|1} + \varepsilon_{J|1}. \tag{13}$$

This step of splitting or filtering the future outputs, $y_{J|1}$, into a signal part, $y^d_{J|1} = DX_J$ when $E = 0$ or $y^d_{J|1} = DX_J + EU_{J|1}$ when $E \neq 0$, and an innovations part $\varepsilon_{J|1} = FE_{J|1}$, is of particular importance in the algorithm.

We propose the following numerical efficient choices for solving this 1st filtering step in the algorithm:

- Using a QR (LQ) decomposition. Interestingly, the square root of the innovations covariance matrix, $F$, is also obtained in this 1st QR step, as in Di Ruscio (1996). Using the definitions or the QR decomposition gives approximately the same results in our simulation examples. One should here note that when the QR decomposition is used also the $Q$ factors as well as the $R$ factors are used. We have from the QR decomposition

$$\begin{bmatrix} U_{0|J+g} \\ Y_{0|J+1} \end{bmatrix} = \begin{bmatrix} R_{11} & 0 \\ R_{21} & R_{22} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}, \tag{14}$$

  where $g = 0$ for closed loop systems and systems with no direct feed through term in the output Eq., i.e., when $E = 0$. For open loop systems and when we want to estimate the direct feed through matrix $E$ we put $g = 1$. From, the decomposition (14) we have

$$y^d_{J|1} = R_{21}Q_1, \tag{15}$$
$$\varepsilon_{J|1} = R_{22}Q_2, \tag{16}$$

  and we notice that also the $Q$ sub matrices are used. Notice also that we may take $F = R_{22}$ or compute a new QR decomposition of $\varepsilon_{J|1}$ in order to estimate $F$.

- Another interesting solution to this step is to use a truncated Conjugate Gradient (CG) algorithm, Hestenes and Stiefel (1952), to compute the projection. The CG algorithm is shown to be equivalent to the Partial Least Squares algorithm in Di Ruscio (2000) for univariate (single output) systems. This will include a small bias but the variance may be small. This choice may be considered for noisy systems in which PEM and the DSR_e method have unsatisfactory variances or for problems where one has to choose a "large" past horizon parameter $J$. However, one has to consider a multivariate version, e.g. the one proposed in Di Ruscio (2000).

This step of splitting the future outputs, $y_{J|1}$, into a signal part, $y^d_{J|1}$, and an innovations part, $\varepsilon_{J|1}$, is consistent and believed to be close to optimal, see Section 4.

The 2nd step in the **dsr_e** algorithm is a deterministic subspace system identification step if the system order has to be found, or an optimal deterministic Ordinary Least Squares (OLS) step if the system order is known, for finding the Kalman filter model. Using PEM for solving the 2nd deterministic identification problem may also be an option.

Hence, the future innovations $\varepsilon_{J|1} = FE_{J|1}$ (noise part), as well as the given input and output data are given and we simply have to solve a deterministic identification problem. Note also that if the system order, $n$, is known this also is equivalent to a deterministic OLS or ARX problem for finding the model. This method is effectively implemented through the use of QR (LQ) factorization, see the D-SR Toolbox for Matlab function **dsr_e.p**.

At this stage the innovations sequence, $\varepsilon_{J|1}$, and the noise free part, $y^d_{J|1}$, of the output $y_{J|1}$ are known from the 1st step in the DSR_e algorithm. Hence we have to solve the following deterministic identification problem

$$x_{k+1} = Ax_k + \begin{bmatrix} B & K \end{bmatrix} \begin{bmatrix} u_k \\ \varepsilon_k \end{bmatrix}, \tag{17}$$
$$y^d_k = Dx_k, \tag{18}$$

where not only the input and output data, $u_k$, and, $y^d_k$, i.o., are known, but also the innovations, $\varepsilon_k$, are known. Hence, the data

$$\left.\begin{matrix} u_k \\ \varepsilon_k \\ y^d_k \end{matrix}\right\} \ \forall\ k = J, J+1, \ldots, N-1 \tag{19}$$

are known, and the model matrices $(A, B, K, D)$ may be computed in an optimal OLS problem.

The 2nd step in the DSR_e method is best discussed as a deterministic identification problem where we will define new input and output data satisfying a deterministic system defined as follows

$$\left.\begin{array}{rl} y_k &:= y_k^d \\ u_k &:= \left[\begin{array}{c} u_k \\ \varepsilon_k \end{array}\right] \end{array}\right\} \ \forall\ k = 1, 2, \ldots, N \tag{20}$$

where $N := N - J$ is the number of samples in the time series to be used in the deterministic identification problem. Hence we have here defined new outputs, $y_k$, from all corresponding samples in the noise free part $y_{J|1}^d \in \mathbb{R}^{m\times K}$ where the number of columns is $K = N-J$. Similarly, new input data $u_k \in \mathbb{R}^{r+m}$, is defined from the sequence $u_{J|1} = \left[\begin{array}{cccc} u_J & u_{J+1} & \cdots & u_{N-1} \end{array}\right]$ of the original input data, and from the the computed innovations in $\varepsilon_{J|1} = \left[\begin{array}{cccc} \varepsilon_J & \varepsilon_{J+1} & \cdots & \varepsilon_{N-1} \end{array}\right]$.

In the examples of Section 4 we illustrate the statistical properties of the method on open loop data, and we show that DSR_e is as optimal as PEM.

We believe that the 1st step in the DSR_e algorithm is close to optimal. Since we have some possibilities for implementing the 2nd step in the algorithm, i.e., the deterministic identification problem, at least in the multiple output case. We discuss some possibilities for the 2nd step separately in the following subsection. This 2nd deterministic identification step is believed to be of interest in itself.

## 3.1. Deterministic subspace identification problem

Step 2 in the **dsr_e.p** implementation in the D-SR Toolbox for MATLAB is basically as presented in this subsection.

Consider an integer parameter $L$ such that the system order, $n$, is bounded by $1 \le n \le L\,m$. As an example for a system with $m = 2$ outputs and $n = 3$ states it is sufficient with $L = 2$.

From the known deterministic input and output data $\left.\begin{array}{c} u_k \\ y_k \end{array}\right\} \ \forall\ k = 0, 1, \ldots, N$ define the data matrix equation

$$Y_{1|L} = \tilde{A}_L Y_{0|L} + \tilde{B}_L U_{0|L+g}, \tag{21}$$

where the matrices are given by

$$\tilde{A}_L = O_L A (O_L^T O_L)^{-1} O_L^T, \tag{22}$$

$$\tilde{B}_L = \left[\begin{array}{cc} O_L B & H_L^d \end{array}\right] - \tilde{A}_L \left[\begin{array}{cc} H_L^d & 0_{Lm\times r} \end{array}\right] \tag{23}$$

The same data as used in Eq. (21), i.e. $Y_{0|L+1} = \left[\begin{array}{c} Y_{0|L} \\ Y_{L|1} \end{array}\right]$ and $U_{0|L+g}$ are used to form the matrix Eq.

$$Y_{0|L+1} = O_{L+1} X_{0|J} + H_{L+1}^d U_{0|L+g}. \tag{24}$$

There are some possibilities to proceed but we suggest estimating the extended observability matrix from Eq. (24) and the $B$, $E$ matrices as an optimal OLS problem from Eq. (21), using the corresponding $R$ sub matrices from the following LQ (transpose of QR) factorization, i.e.,

$$\left[\begin{array}{c} U_{0|L+g} \\ Y_{0|L+1} \end{array}\right] = \left[\begin{array}{cc} R_{11} & 0 \\ R_{21} & R_{22} \end{array}\right] \left[\begin{array}{c} Q_1 \\ Q_2 \end{array}\right]. \tag{25}$$

Due to the orthogonal properties of the QR factorization we have

$$R_{22} = O_{L+1} X_{0|J} Q_2^T, \tag{26}$$

and the system order, $n$, the extended observability matrix $O_{L+1}$, and thereafter $A$ and $D$, may be estimated from an SVD of $R_{22}$, and using the shift invariance technique. An alternative to this is to form the matrix Eq.

$$\bar{R}_{22} = \tilde{A}_L \underline{R}_{22}, \tag{27}$$

and estimate the system order as the $n$ largest non-zero singular values of the $Lm$ singular values of the matrix $\underline{R}_{22}$. Here $\bar{R}_{22}$ is obtained as $R_{22}$ with the 1st $m$ block row deleted and $\underline{\mathrm{R}}_{22}$ as $R_{22}$ with the last $m$ block row deleted. Using only the $n$ largest singular values we have from the SVD that $\underline{\mathrm{R}}_{22} = U_1 S_1 V_1^T$ and we may choose $O_L = U_1$ and find $A$ from Eq. (27), i.e., as

$$A = U_1^T \bar{R}_{22} V_1 S_1^{-1}. \tag{28}$$

Note that there are common block rows in $\underline{R}_{22}$ and $\bar{R}_{22}$. This may be utilized and we may use

$$A = U_1^T \left[\begin{array}{c} \bar{U}_1 \\ R_{22}(Lm+1 : (L+1)m, :) V_1 S_1^{-1} \end{array}\right], \tag{29}$$

which is used in the DSR algorithms. This means that we, for the sake of effectiveness, only use the truncated SVD of $\underline{\mathrm{R}}_{22}$ and the last block row in $R_{22}$ in order to compute an estimate of the $A$ matrix. The $D$ matrix is taken as the 1st $m$ block row in $O_L = U_1$. This way of computing the $A$ and $D$ matrices is a result of Di Ruscio (1994).

Finally, the parameters in the $B$ and $E$ matrices are estimated as an optimal OLS step, using the structure of matrix $\tilde{B}_L$ and the equation

$$\bar{R}_{21} = \tilde{A}_L \underline{\mathrm{R}}_{21} + \tilde{B}_L R_{11}, \tag{30}$$

where $\bar{R}_{21}$ is obtained as $R_{21}$ with the 1st $m$ block row deleted and $\underline{\mathrm{R}}_{21}$ as $R_{21}$ with the last $m$ block row deleted. Since $\tilde{A}_L$ now is known the problem $\tilde{B}_L R_{11} = \bar{R}_{21} - \tilde{A}_L \underline{\mathrm{R}}_{21}$ may be solved for the parameters in $B$ (and $E$ for open loop systems) as an optimal OLS problem in the unknown $(n+m)r$ parameters in $B$ and $E$, Di Ruscio (2003). We also mention in this connection that it is possible to include constraints in this OLS step, e.g. solve structural problems, e.g. with $K = 0$ as in output error models.

Note that, by using the realization algorithm in Ho and Kalman (1966), Hankel matrices $H_{1|L} = O_L C_J$ and $H_{2|L} = O_L A C_J$ are constructed from impulse response matrices $h_i = CA^{i-1}B \ \forall \ i = 1, \ldots, L+J$, where $L+J$ is the number of impulse response matrices. Using the SVD realization algorithm in Zeiger and McEwen (1974) gives the model matrices $A$, $B$ and $D$ and the system order, $n$. See Di Ruscio (2009b) for details.

The MATLAB D-SR Toolbox is available upon request.

## 4. Numerical Results

In the following examples we will compare the algorithms **n4sid**, **pem**, **dsr_e** and **dsr** on Monte Carlo simulations using random state space models, viz. we generate Discrete Random State Space (DRSS) models using the MATLAB function **drss**. In the incoming examples, we will generate DRSS models by using the following MATLAB code.

```
m=randi(4);
r=randi([m,m+1]); n=randi([m+1,m+2]);
dsys=drss(n,m,r); dsys.d=0;
```

Note that, the system is stable, i.e., the eigenvalues of A, generated from the MATLAB function **drss**, are inside the unit circle

For each DRSS model we will perform $M = 100$ noise realizations. For each noise realization we will simulate the DRSS model with a pseudo random binary sequence as input signal of length, $N = 100$, using following MATLAB code.

```
for k=1:nu
   U(:,k)=prbs1(N,randi([4,9]),randi([16,25]));
end
```

The signal, $u_k$, is constant on random intervals, T, specified by the band, $T_{\mathbf{min}} \leq T \leq T_{\mathbf{max}}$. The MATLAB m-file function **prbs1.m** is available on request.

For comparing the results we will use the size of the covariance matrix of the error between the estimated and true parameter, i.e.

$$P_{alg} = \frac{N}{M-1}(\hat{\theta}_i - \theta_0)(\hat{\theta}_i - \theta_0)^T, \tag{31}$$

as

$$V_{alg} = \mathbf{trace}(P_{alg}), \tag{32}$$

where subscript $alg$ means the algorithms, **dsr**, **n4sid** and **dsr_e** or **pem**. The true parameter vector (i.e. in MATLAB notation), $\theta_0 = [A_c(:); B_c(:); D_c(:)]$, and the estimate parameter vector, $\hat{\theta}_i = [\hat{A}_c(:); \hat{B}_c(:); \hat{D}_c(:)]_i$, for each $i$ in $1 \leq i \leq M$. The subscript $c$ means observable canonical form.

Note that, the criterion defined in Eqs. (31) and (32) were copied from Eqs. (69) and (70) in the paper of Di Ruscio (2009a), i.o.

To compare the algorithms over a set of DRSS models we define the following scaling criterion,

$$\mathcal{V}_{alg} = \frac{V_{alg}}{\sum_{alg} V_{alg}}. \tag{33}$$

In the incoming examples, the future and the past horizon input parameters, $L$ and $J$, i.o., used in the **dsr** and **dsr_e** algorithms, are chosen such that the mean squared error (i.e. between the simulated output from the algorithm and the identification data) is minimized, where the system order, $n$, is assumed known. Also, the feed through matrix is set $E = 0$ (i.e. we have set $g = 0$). Note that, originally the **n4sid** algorithm only had, $L = J$, however in the MATLAB implementation of **n4sid** there exists an option to set the past horizon parameter, $J$. See the **opt_Lj.m** MATLAB function as follows.

```
function [opt_L,opt_J,n]=...
    opt_LJ(Y,U,Lmax,g,Jmax,alt,n)

[N,ny]=size(Y);
[~,nu]=size(U);
minV=inf;

for L=ceil(n/ny):Lmax
    for J=L:Jmax
        if (N - J - L < (J + L + g)*nu...
                + (J + L + 1)*ny)
         return
            disp('Need more data.');
        end
        if strcmp(alt,'dsr')
            [A,B,D,E,~,~,x0]=...
                dsr(Y,U,L,g,J,1,n);
        elseif strcmp(alt,'dsr_e')
```

```
            [A,B,D,E,~,~,x0]=....
                dsr_e(Y,U,L,g,J,n);
        elseif strcmp(alt,'n4sid')
            [dsys,x0] =...
                n4sid(iddata(Y,U,1),n,...
                'Feedthrough',g,'N4Horizon',...
                [L+1, J+1, J+1]);
            A=dsys.A;B=dsys.B;D=dsys.C;
            E=dsys.D;
        else
            return
        end
        Ym=dsrsim(A,B,D,E,U,x0);
        V = norm((Y-Ym)'*(Y-Ym)/N);
        if V < minV
            opt_L=L;
            opt_J=J;
            minV=V;
        end
    end
end
```

**Example 4.1 (Open loop MIMO)**

*In this example, we will compare* **dsr**, **n4sid** *and* **dsr_e**, *on 100 DRSS models.*

*The process noise* $v_k$ *and measurements noise,* $w_k$ *are white with covariance matrices* $E(v_k v_k^T) = 0.03^2\, I_n$ *and* $E(w_k w_k^T) = 0.03^2\, I_m$, *i.o.*

*It is seen in Table 1 (rows 2:4, column 2) that the* **dsr_e** *algorithm has an edge over both* **n4sid** *and* **dsr**, *in terms of the mean of the criterion* $\mathcal{V}$ *(defined in Eq. (33)), viz.* **dsr_e** *is seen to be,* $\mathbf{mean}(\mathcal{V}_{\mathbf{dsr}} - \mathcal{V}_{\mathbf{dsr\_e}}) = 0.3002 - 0.2081 = 0.0921 \approx 9\%$, *better than* **dsr** *and,* $\mathbf{mean}(\mathcal{V}_{\mathbf{n4sid}} - \mathcal{V}_{\mathbf{dsr\_e}}) = 0.4917 - 0.201 = 0.2836 \approx 28\%$, *better than* **n4sid**. *This also illustrated in Figure 1.*

Table 1: Example 4.1. The table shows the mean of the criterion $\mathcal{V}$ (defined in Eq. (33)) for **dsr_e**, **dsr** and **n4sid** based on random discrete space model Monte Carlo simulations.

| Algorithm | $\mathbf{mean}(\mathcal{V}_{alg})$ |
|---|---|
| **dsr_e** | 0.2081 |
| **dsr** | 0.3002 |
| **n4sid** | 0.4917 |

**Example 4.2 (Open loop MIMO PEM)**

*In this example, we will compare* **dsr_e** *and* **pem** *on 100 DRSS models.*

*The process noise* $v_k$ *and measurements noise,* $w_k$ *are white with covariance matrices* $E(v_k v_k^T) = 0.05^2\, I_n$ *and* $E(w_k w_k^T) = 0.03^2\, I_m$, *i.o.*

*The MATLAB function* **pem** *is called as follows.*

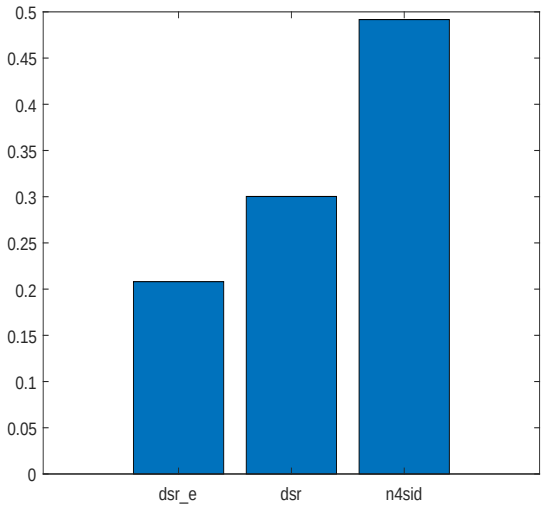


Figure 1: Example 4.1. The figure shows the mean of the criterion $\mathcal{V}$ (defined in Eq. (33)) for **dsr_e**, **dsr** and **n4sid** based on random discrete space model Monte Carlo simulations. **dsr_e** gives the best performance, i.e. lowest $\mathcal{V}$.

```
opt=n4sidOptions('Focus','simulation');
idd=iddata(Y,U,1);
n4sys=n4sid(idd,n,opt,'Feedthrough',0);
pemsys = pem(idd,n4sys);
```

*Surprisingly, at least in this example, it is seen in Table 2 (rows 2:3, column 2) that the* **dsr_e** *algorithm has an edge over* **pem** *in terms of the mean of the criterion* $\mathcal{V}$ *(defined in Eq. (33)). See also Figure 2, viz.* **dsr_e** *is seen to be,* $\mathbf{mean}(\mathcal{V}_{\mathbf{pem}} - \mathcal{V}_{\mathbf{dsr\_e}}) = 0.7482 - 0.2518 = 0.4964 \approx 50\%$, *better than* **pem**.

Table 2: Example 4.2. The table shows the mean of the criterion $\mathcal{V}$ (defined in Eq. (33)) for **dsr_e** and **pem** based on random discrete state space model Monte Carlo simulations.

| Algorithm | $\mathbf{mean}(\mathcal{V}_{alg})$ |
|---|---|
| **dsr_e** | 0.2518 |
| **pem** | 0.7482 |

**Example 4.3 (Swedish Fighter Gripen)**

*In this example, we will consider the JAS 39 Gripen fighter aircraft (developed jointly by Saab Military Aircraft and British Aerospace). This example is inspired by the paper of Ljung (2013). The raw dataset of length*

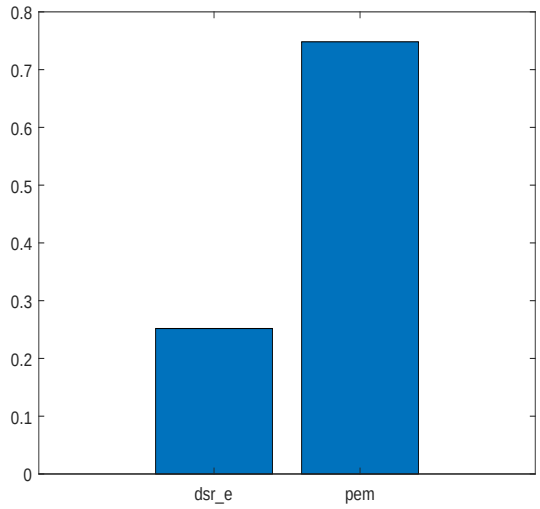


Figure 2: Example 4.2. The figure shows the mean of the criterion $\mathcal{V}$ (defined in Eq. (33)) for **dsr_e** and **pem** based on random discrete space model Monte Carlo simulations. **dsr_e** gives the best performance, i.e. lowest $\mathcal{V}$.

*equal to 815 samples which corresponds to 180 samples (i.e. 3 s sampled at 60 Hz) of test flight data is illustrated in Figure 4. Note that, these data are the result of reverse engineering of Figure 2 in Ljung (2013)[1] (the scaling is not preserved). Note that, the 1st 184 samples of the raw dataset are removed. The dataset is detrended using the 1st sample, i.e. the 1st sample is subtracted from the dataset. 631 samples are used for validation data and the 1st 300 samples of these are used for identification data. Note that, Figure 5 illustrates the identification and validation dataset. The authors suspect that the data from Figure 2 in Ljung (2013) has been filtered.*

*We will consider the following case:*

$$y_k \in \mathbb{R} := \Big\{ \textit{Pitch Rate},$$

$$u_k \in \mathbb{R}^3 := \begin{cases} u_1 : \textit{Leading Edge Flap}, \\ u_2 : \textit{Canard Angle}, \\ u_3 : \textit{Elevator Angle} \end{cases}$$

*Testing of different input-output pairings has been done, whereas using all three inputs, was found to produce the most accurate models. The model order was also tested, whereas choosing, $n = 1$, was seen as sufficient.*

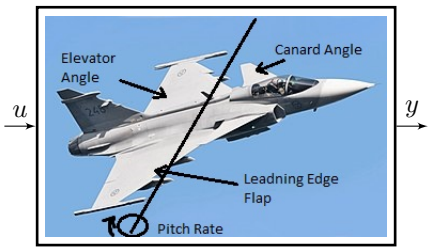


Figure 3: Example 4.3. The figure shows an open loop system of the JAS 39 Gripen fighter aircraft. The figure illustrates the inputs and output case. The figure shows an edited version of Tuomo Salonen / SIM Finnish Aviation Museum (2017).

*The state space models identified using* **dsr_e**, **pem** *and* **dsr** *was seen similar (wrt. the criterion in Eq. (32)), viz.* **pem** *(i.e. the SSM as in Eqs. (3) and (4) with matrices as in Eq. (34)) was seen in Table 3 (rows 2:4, column 2) to be,* $\frac{V_{\mathbf{dsr}}}{V_{\mathbf{pem}}} = 447.2/389 = 1.15$, *times better than* **dsr** *and,* $\frac{V_{\mathbf{dsr\_e}}}{V_{\mathbf{pem}}} = 440.9/389 = 1.13$, *times better than* **dsr_e**, *on the validation set. The eigenvalues for the* **pem**, **dsr** *and* **dsr_e** *models, shown in Table 4, are close to integrator.*

$$\overbrace{\begin{aligned} A &= 0.9988 \\ B &= \begin{bmatrix} 0.00000233 \\ 0.00001032 \\ -0.00002484 \end{bmatrix} \\ D &= 2412 \\ E &= 0 \\ K &= 0.0003762 \end{aligned}}^{\mathbf{pem}} \tag{34}$$

*In the paper of Ljung (2013), a 5-step ahead predictor was used (illustrated in Figure 3 in Ljung (2013)). This corresponds to,* $M = \mathbf{ceil}(815/180)\,5 = 25$ *step. In Figure 7 a 25-step ahead prediction of the output* $y_k$ *from the* **dsr**, **dsr_e** *and* **pem** *models are illustrated.*

Table 3: Example 4.3. The table shows the mean square error, i.e. the criterion Eq. (32) corresponding to the validation data shown in Figure 7.

| $V_{alg}$ | Validation set |
|---|---|
| $V_{\mathbf{dsr}}$ | 447.2 |
| $V_{\mathbf{dsr\_e}}$ | 440.9 |
| $V_{\mathbf{pem}}$ | 389.0 |

[1] The reverse engineering is done using a picture to data (pic2dat) tool, i.e. a tool for converting a picture depicting a plot from jpg, png, etc to a dataset. Request 2nd author.

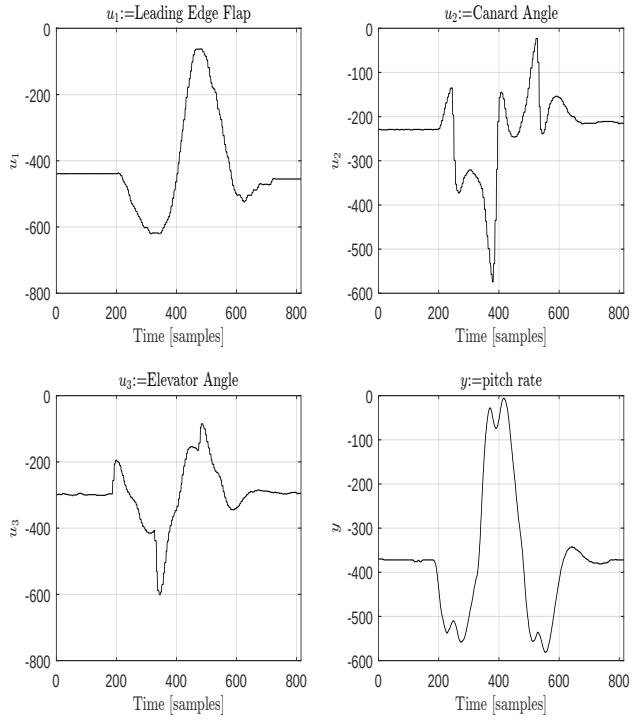


Figure 4: Example 4.3. JAS Gripen test flight raw data. 817 samples correspond to Gripen test flight over 3 s with a sample interval of 1/60 s. These data are the result of reverse engineering of figure 2 in Ljung (2013).

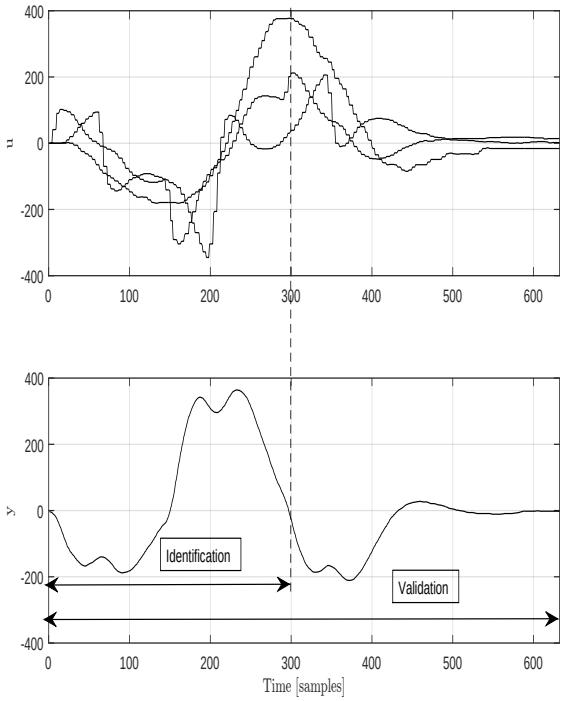


Figure 5: Example 4.3. The figure shows the identification and validation data.

Table 4: Example 4.3. The table shows the eigenvalues of the models identified using the algorithms **dsr**, **dsr_e** and **pem**.

| *alg* | eigenvalue |
|---|---|
| **dsr** | 0.9989 |
| **dsr_e** | 0.9988 |
| **pem** | 0.9988 |

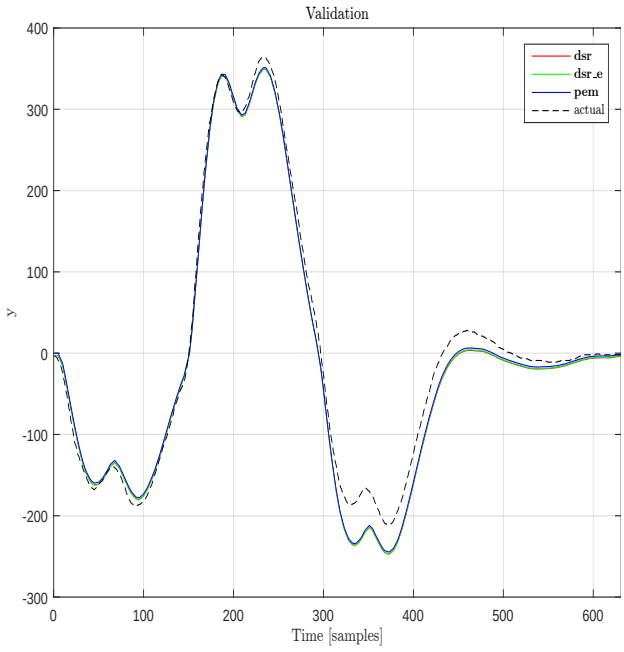


Figure 6: Example 4.3. The figure shows the comparison of the simulated output, $y_k^d$, from the algorithms **dsr**, **dsr_e** and **pem**, with the actual output, $y_k$ (validation). The corresponding criterion, i.e. the mean square error, are given in Table 3.

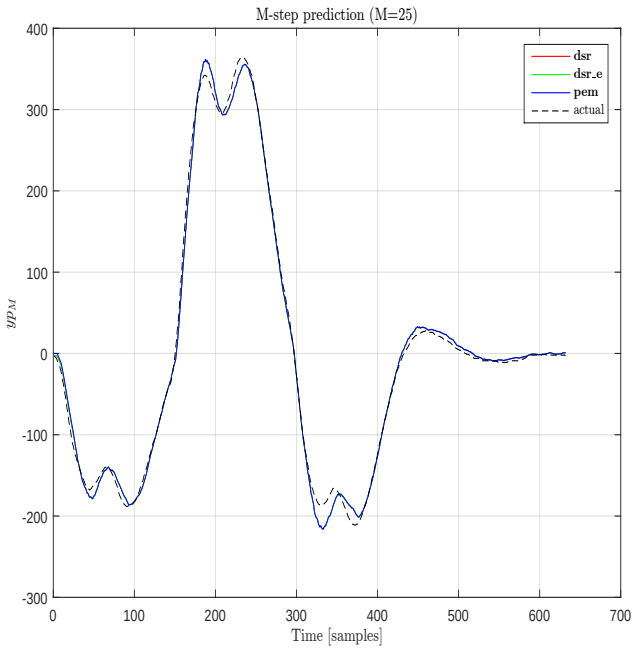


Figure 7: Example 4.3. The figure shows an M-step ahead prediction of the output $y_k$ from the **dsr**, **dsr_e** and **pem** models with $M = 25$.

## 5. Concluding Remarks

The discussion and concluding remarks are itemized as follows:

- The results from Monte Carlo simulations with 100 DRSS models, each with 100 noise realizations have been documented in the numerical examples in Sec. 4.

  In Example 4.1, the **dsr_e** algorithm is seen to have an edge over both the **dsr** and **n4sid** algorithms on open loop data generated from DRSS models by at least 9%.

  Surprisingly, the **dsr_e** algorithm is seen in Example 4.2 to outperform **pem** by 50%. Note that, for enhancing performance, more emphasis could have been put on choosing the options used in the **pem** algorithm. It may also be explained by a trade-off between bias and variance (Di Ruscio (2009a)).

- **pem** was seen slightly better than **dsr_e** and **dsr** in a Example 4.3, i.e. on flight data from a Swedish JAS Gripen Fighter Case (see Ljung (2013)). The **pem** algorithm was seen at least 1.13 times better than **dsr_e** and **dsr**, in terms of performance criterion $V_{alg}$ in Eq. (32).

## 6. Acknowledgement

The theory of the DSR_e method is from the references, 1st author. The motivation behind the paper and simulation experiments are performed by the 2nd author.

## A. Notations and definitions

Hankel matrices are frequently used in realization theory and subspace system identification. The special structure of a Hankel matrix as well as some matching notations, which are frequently used throughout the paper, are defined in the following.

**Definition A.1 (Hankel matrix)** *Given a vector sequence*

$$s_k \in \mathbb{R}^{nr} \ \forall \ k = 0, 1, 2, \ldots, \tag{35}$$

*where $nr$ is the number of rows in $s_k$.*

*Define integer numbers $j$, $i$ and $nc$ and define the matrix $S_{j|i} \in \mathbb{R}^{inr \times nc}$ as follows*

$$S_{j|i} \stackrel{def}{=} \begin{bmatrix} s_j & s_{j+1} & s_{j+2} & \cdots & s_{j+nc-1} \\ s_{j+1} & s_{j+2} & s_{j+3} & \cdots & s_{j+nc} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{j+i-1} & s_{j+i} & s_{j+i+1} & \cdots & s_{j+i+nc-2} \end{bmatrix},$$

*which is defined as a Hankel matrix because of the special structure. A Hankel matrix is symmetric and the elements are constant across the anti-diagonals. The integer numbers $j$, $i$ and $nc$ have the following interpretations:*

- *$j$ start index or initial time in the sequence used to form $S_{j|i}$, i.e., $s_j$, is the upper left vector element in the Hankel matrix.*
- *$i$ is the number of $nr$-block rows in $S_{j|i}$.*
- *$nc$ is the number of columns in the Hankel matrix $S_{j|i}$.*

One should note that this definition of a Hankel matrix $S_{j|i}$ is slightly different from the notation used in Overschee and de Moor (1996) (pp. 34-35) where the subscripts denote the 1st and last element of the 1st column in the block Hankel matrix, i.e. using the notation in Overschee and de Moor (1996) a Hankel matrix $U_{0|i}$ is defined by putting $u_0$ in the upper left corner and $u_i$ in the lower left corner and hence $U_{0|i}$ would have $i+1$ rows.

Examples of such vector processes, $s_k$, to be used in the above Hankel-matrix definition, are the measured process outputs, $y_k \in \mathbb{R}^m$, and possibly known inputs, $u_k \in \mathbb{R}^r$.

**Definition A.2 (Basic matrix definitions)**
*The extended observability matrix, $O_i$, for the pair $(D, A)$ is defined as*

$$O_i \stackrel{def}{=} \begin{bmatrix} D \\ DA \\ \vdots \\ DA^{i-1} \end{bmatrix} \in \mathbb{R}^{im \times n}, \tag{36}$$

*where the subscript $i$ denotes the number of block rows.*

*The reversed extended controllability matrix, $C_i^d$, for the pair $(A, B)$ is defined as*

$$C_i^d \stackrel{def}{=} \begin{bmatrix} A^{i-1}B & A^{i-2}B & \cdots & B \end{bmatrix} \in \mathbb{R}^{n \times ir}, \tag{37}$$

*where the subscript $i$ denotes the number of block columns. A reversed extended controllability matrix, $C_i^s$, for the pair $(A, C)$ is defined similar to (37), i.e.,*

$$C_i^s \stackrel{def}{=} \begin{bmatrix} A^{i-1}C & A^{i-2}C & \cdots & C \end{bmatrix} \in \mathbb{R}^{n \times im}, \tag{38}$$

*i.e., with $B$ substituted with $C$ in (37). The lower block triangular Toeplitz matrix, $H_i^d \in \mathbb{R}^{im \times (i+g-1)r}$, for the quadruple matrices $(D, A, B, E)$.*

$$H_i^d \stackrel{def}{=} \begin{bmatrix} E & 0_{m\times r} & 0_{m\times r} & \cdots & 0_{m\times r} \\ DB & E & 0_{m\times r} & \cdots & 0_{m\times r} \\ DAB & DB & E & \cdots & 0_{m\times r} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}B & DA^{i-3}B & DA^{i-4}B & \cdots & E \end{bmatrix}$$

*where the subscript $i$ denotes the number of block rows and $i+g-1$ is the number of block columns, and where $0_{m\times r}$ denotes the $m \times r$ matrix with zeroes. A lower block triangular Toeplitz matrix $H_i^s \in \mathbb{R}^{im\times im}$ for the quadruple $(D, A, C, F)$ is defined as*

$$H_i^s \stackrel{def}{=} \begin{bmatrix} F & 0_{m\times m} & 0_{m\times m} & \cdots & 0_{m\times m} \\ DC & F & 0_{m\times m} & \cdots & 0_{m\times m} \\ DAC & DC & F & \cdots & 0_{m\times m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}C & DA^{i-3}C & DA^{i-4}C & \cdots & F \end{bmatrix}$$

Given two matrices $A \in \mathbb{R}^{i\times k}$ and $B \in \mathbb{R}^{j\times k}$, the orthogonal projection of the row space of $A$ onto the row space of $B$ is defined as

$$A/B = AB^T(BB^T)^\dagger B. \tag{39}$$

The following property is used

$$A/\begin{bmatrix} A \\ B \end{bmatrix} = A. \tag{40}$$

Proof of Eq. (40) can be found in e.g., Di Ruscio (1997).

## B. Background theory

Consider the "past-future" data matrix, Eq. (4) in the paper Di Ruscio (1997). This same equation is also presented in Eq. (27) in the Di Ruscio (1997) paper. We have

$$\begin{aligned} Y_{J|L} &= \begin{bmatrix} H_L^d & O_L\tilde{C}_J^d & O_L\tilde{C}_J^s \end{bmatrix} \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \\ &+ O_L(A-KD)^J X_0 + H_L^s E_{J|L}, \end{aligned} \tag{41}$$

where $\tilde{C}_J^s = C_J^s(A-KD, K)$ is the reversed extended controllability matrix of the pair $(A-KD, K)$, and $\tilde{C}_J^d = C_J^d(A-KD, B-KE)$ is the reversed extended controllability matrix of the pair $(A-KD, B-KE)$, and where $C_J^d$ and $C_J^s$ are defined in Eqs. (37) and (38), i.o..

One should here note that the term

$$\lim_{J\to\infty} O_L(A-KD)^J X_0 = 0 \tag{42}$$

Also note Eq. (43) in Di Ruscio (2003) with proof, i.e.,

$$X_J = \begin{bmatrix} \tilde{C}_J^d & \tilde{C}_J^s \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} + (A-KD)^J X_0, \tag{43}$$

which relates the "past" data matrices, $U_{0|J}$ and $Y_{0|J}$ to the "future" states

$$X_J = \begin{bmatrix} x_J & x_{J+1} & \cdots & x_{N-(J+L)} \end{bmatrix}. \tag{44}$$

Note that, Eq. (41) is obtained by putting (43) into Eq. (5).

## References


Di Ruscio, D. Methods for the identification of state space models from input and output measurements. 1994. The 10 IFAC symposium on system identification SYSID94, Copenhagen, 4-6 July.

Di Ruscio, D. Methods for the identification of state space models from input and output measurements. In *10th IFAC Symp. on System Identif.* 1994.

Di Ruscio, D. A method for the identification of state space models from input and output measurements. *Modeling, Identification and Control*, 1995. 16(3):129–143. doi:10.4173/mic.1995.3.2.

Di Ruscio, D. Combined Deterministic and Stochastic System Identification and Realization: DSR-a subspace approach based on observations. *Modeling, Identification and Control*, 1996. 17(3):193–230. doi:10.4173/mic.1996.3.3.

Di Ruscio, D. On subspace identification of the extended observability matrix. In *36th Conf. on Decision and Control*. 1997.

Di Ruscio, D. A weighted view of the partial least squares algorithm. *Automatica*, 2000. 36(6):831–850. doi:10.1016/S0005-1098(99)00210-1.

Di Ruscio, D. Subspace System Identification of the Kalman Filter. *Modeling, Identification and Control*, 2003. 24(3):125–157. doi:10.4173/mic.2003.3.1.

Di Ruscio, D. Subspace system identification of the Kalman filter: open and closed loop systems. In *Proc. Intl. Multi-Conf. on Engineering and Technological Innovation*. 2008.

Di Ruscio, D. A Bootstrap Subspace Identification Method: Comparing Methods for Closed Loop Subspace Identification by Monte Carlo Simulations. *Modeling, Identification and Control*, 2009a. 30(4):203–222. doi:10.4173/mic.2009.4.2.

Di Ruscio, D. Closed and Open Loop Subspace System Identification of the Kalman Filter. *Modeling, Identification and Control*, 2009b. 30(2):71–86. doi:10.4173/mic.2009.2.3.

Hestenes, M. R. and Stiefel, E. Methods for Conjugate gradients for Solving Linear Systems. *J. Res. National Bureau of Standards*, 1952. 49(6):409–436.

Ho, B. L. and Kalman, R. E. Effective construction of linear state-variable models from input/output functions. *Regelungstechnik*, 1966. 14(12):545–592.

Ljung, L. *System Identification: Theory for the User*. Prentice Hall information and system sciences series. Prentice Hall PTR, 1999.

Ljung, L. Some classical and some new ideas for identification of linear systems. *Journal of Control, Automation and Electrical Systems*, 2013. 24. doi:10.1007/s40313-013-0004-7.

MATLAB. *version 9.9.0.1718557 (R2020b)*. The MathWorks Inc., Natick, Massachusetts, USA, 2020. Control System Toolbox, Version 10.9. System Identification Toolbox, Version 9.13.

Nilsen, G. W. *Topics in open and closed loop subspace system identification: finite data based methods*. Ph.D. thesis, NTNU-HiT, 2005. ISBN 82-471-7357-3.

Overschee, P. V. and de Moor, B. N4SID: Subspace Algorithms for the Identification of Combined Deterministic Stochastic Systems. *Automatica*, 1994. 30(1):75–93. doi:10.1016/0005-1098(94)90230-5.

Overschee, P. V. and de Moor, B. *Subspace identification for linear systems*. Kluwer Acad. Publ., 1996.

Qin, S., Lin, W., and Ljung, L. A novel subspace identification approach with enforced causal models. *Automatica*, 2005. 41(12):2043–2053. doi:10.1016/j.automatica.2005.06.010.

Tuomo Salonen / SIM Finnish Aviation Museum. Saab gripen at the kaivopuisto air show in june. 2017. URL `https://en.wikipedia.org/wiki/Saab_JAS_39_Gripen#/media/File:Saab_JAS_39_Gripen_at_Kaivopuisto_Air_Show,_June_2017_(altered)_copy.jpg`. Online; accesed 08-0.5-22. The original picture has been edited.

van der Veen, G., van Wingerden, J.-W., Bergamasco, M., Lovera, M., and Verhaegen, M. Closed-loop subspace identification methods: an overview. *IET Control Theory & Applications*, 2013. 7(10):1339–1358. doi:10.1049/iet-cta.2012.0653.

Wang, J. and Qin, S. J. A new subspace identification approach based on principal component analysis. *Journal of Process Control*, 2002. 12:841–855.

Zeiger, H. and McEwen, A. Approximate linear realizations of given dimensions via Ho's algorithm. *IEEE Trans. on Automatic Control*, 1974. 19(2):153. doi:10.1109/TAC.1974.1100525.