A System, Signals and Identification Toolbox in *Mathematica* with Symbolic Capabilities. ⋆

Jonas Sjöberg * **Håkan Hjalmarsson** **

* *Department of Signals and Systems, Chalmers University of Technology, SE41296 Göteborg, Sweden*
** *ACCESS Linnaeus Center, Electrical Engineering, KTH – Royal Institute of Technology, SE100 44 Stockholm, Sweden*

Abstract: In this contribution we describe a new signals, systems and identification toolbox for the symbolic and numerical computation system *Mathematica*. The toolbox provides functionality for computation of properties of systems and signals ranging from frequency responses, zeros and poles to signal spectra and spectral factorizations. It also includes a wide range of identification algorithms ranging from spectral analysis to subspace and prediction error identification of models for non-linear systems. The symbolic capabilities of *Mathematica* are used to allow the user to construct very general model structures, and for pre-processing, such as gradient calculations, when optimizing the parameters in such structures.

1. INTRODUCTION

Mathematica as an environment for analysis and synthesis in signals, systems, control and identification theory and practice is still quite under-developed when compared to the extensive list of functions available in MATLAB. However, symbolic computational software systems, such as *Mathematica*, offers interesting features that are useful in these areas. In particular for signal analysis and system identification, symbolic calculations allow to compute statistical properties such as covariances, correlations and spectra as explicit function of parameters. This may be useful for researchers as well as in education where theoretical calculations can parallel numerical computations of a signal's sample properties. Another, perhaps less obvious, use of symbolic computations is as a pre-processing tool in optimization of parameters. For example, for analytic computation of gradients of predictors in general model structures, in particular for non-linear systems. This opens up for the user herself to define tailor made model structures.

These driving forces lie behind the signals, systems and identification package for *Mathematica* that is described in this paper. The package is still under development and will be launched in 2009. In short, the package supports a wide range of algorithms for identification of linear and nonlinear models of dynamical systems, which can be used for simulation, prediction, and for control design. It also supports symbolic processing of stationary stochastic processes, i.e. covariance and spectral computations, spectral factorization etc. Thus the package is useful as a tool also for the theoretically oriented researcher and in education.

The symbolic features of *Mathematica* are used so that a large variety of model structures can be handled. Before the numerical estimation procedure, the symbolic definition of the model structure is simplified and modified before the numerical computations of the parameters. In this way a larger set of model structures can be supported, at the same time as the computational speed is improved. In respect to system identification, the package has the following features:

- *Linear models.* The package supports subspace methods for MIMO linear state-space models as well as traditional black box model structures such as FIR, ARX, ARMAX, BJ and for time-series AR and ARMA models. Also non-parametric methods in both time and frequency domain are available.

- *Non-linear models.* The package support of estimation of nonlinear model structures is unprecedented: Nonlinear versions of common linear black box models are supported (NLFIR, NLARX, NLARMAX, NLBJ) and for time-series NLAR and NLARMA models. In addition, nonlinear state-space models, Hammerstein and Wiener models are supported.

  Users can define their own nonlinearities, but guidance and support is available. Parametric models are estimated by the prediction error method.

Care has been taken to minimize computation time so that numerical computations match state-of-the art.

For control design there exists a toolbox for *Mathematica*, called *System Control Professional* (CSP), and models identified with the identification tool can be transformed to the format used in CSP so that one can continue with control design.

⋆

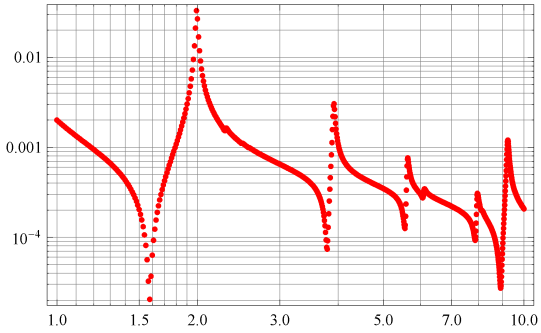

Fig. 1. Computed frequency response for one channel of the International Space Station. The *Mathematica* and MATLAB results are on top of each other.

Outline

The paper unfolds as follows. Section 2 describes functions assessing system properties, and commands for handling models, mainly for linear models. This is followed by a description of functions for stochastic systems theory in Section 3. Functions for spectral analysis are described in Section 4 and the support for nonlinear system identification is explained in Section 5. Some conclusions are then given in Section 6.

2. SYSTEMS THEORY

The standard underlying systems representation is the state-space form, either in continuous time

$$\dot{x}(t) = f(x(t), u(t), w(t), \theta) \quad (1)$$
$$y(t) = h(x(t), u(t), v(t), \theta) \quad (2)$$

where $u(t)$ is a measured signal (the input), and where $w(t)$ and $v(t)$ are disturbances, or in discrete-time

$$x(t+1) = f(x(t), u(t), w(t)\theta) \quad (3)$$
$$y(t) = h(x(t), u(t), v(t), \theta) \quad (4)$$

2.1 Interconnections

Complex systems and models can be constructed by interconnection of simpler building blocks. For example with G and H being models with the same number of inputs and outputs, $G + H$ will result in a model where G and H are connected in parallel and $G \cdot H$ will be G in series with H.

2.2 Frequency response

Different methods are used to compute the frequency response depending on whether or not the state-space model representation contains symbolic elements and whether or not the frequencies are symbolic or not. When both model and the frequency variable are numeric, the well known method using the Heisenberg form as intermediate is used, see Laub (1981). The accuracy is similar to `freqresp` in MATLAB. In Figure 1, the computed frequency response for one of the channels in the model of the International Space Station in Antoulas et al. (2001) is shown both using the *Mathematica* toolbox and `freqresp` in MATLAB.

2.3 Poles and zeros

The well known algorithm in Emami-Naeini and van Dooren (1982) has been implemented for the computation of zeros of a system.

2.4 Partial realization theory

There are several methods to compute a state-space representation from a finite impulse response sequence. The algorithm in Ho and Kalman (1966) forms the basis of the function `ImpulseResponseToStateSpace` but the N4SID subspace identification method can also be used.

Example 1. Consider the system

$$x(t+1) = \begin{bmatrix} a & -0.25 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 2 & 2 \\ 2 & 1 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 2 & 1 \\ 3 & 1 \end{bmatrix} x(t)$$

where a is a free parameter.

From the three first impulse response coefficients

$$h(0) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \; h(1) = \begin{bmatrix} 6 & 8 \\ 5 & 7 \end{bmatrix}, \; h(2) = \begin{bmatrix} 1+4a & 0.5+6a \\ 1.5+4a & 1.25+6a \end{bmatrix}$$

`ImpulseResponseToStateSpace` gives the following state space model

$$x(t+1) = \begin{bmatrix} 1.509 + 0.770a & -0.0597 - 0.0579a \\ 4.565 - 3.058a & -0.151 + 0.230a \end{bmatrix} x(t) + \begin{bmatrix} -0.758 & -0.652 \\ -0.652 & 0.758 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} -7.809 & -0.122 \\ -10.630 & 0.0898 \end{bmatrix} x(t)$$

which has the same input output behavior as the original system.

3. STOCHASTIC SYSTEMS THEORY

The perhaps most common functions in stochastic systems theory have been implemented.

3.1 State covariances and covariance functions

The function `CoVar` computes the state covariance of a state space system when the input is white noise. For a linear time invariant system where the input is stationary with spectrum Φ_u, the cross-covariance between input and output and the output correlations are computed with `CrossCov` and `OutputCov`, respectively.

Example 2. When the input to the system in Example 1 is white noise, `CoVar` gives that the state covariance is given by

$$\frac{1}{(a-1.25)(a+1.25)} \cdot \begin{pmatrix} 37(a-2.06)(a+1.40) & 45(a-2.11)(a+1.50) \\ 45(a-2.11)(a+1.50) & 53(a-2.16)(a+1.67) \end{pmatrix}$$

We see that the covariance matrix increases as $1/(|a| - 1.25)$ when a approaches the stability region $a = \pm 1.25$.

3.2 Positive real part

The positive real part of a spectrum can be computed using `Spectrum2PositiveRealPart`.

Example 3. Consider the signal

$$y(t) = \frac{1}{1 - 0.7q^{-1}} e(t) + w(t)$$

where $\{e(t)\}$ and $\{w(t)\}$ are zero mean white noise sequences with variance 1 and λ, respectively.

The spectrum of y is simple to compute in *Mathematica*:

First we define $G(z)$ and $G(z^{-1})$

```
G = 1/(z - 0.7);
Gstar = G/.z → 1/z;
```

Then the spectrum is given by

```
φy = GGstar + λ
```

$$\frac{1}{\left(-0.7+\frac{1}{z}\right)(-0.7+z)} + \lambda$$

The positive real part for this spectrum is obtained by

```
F = Spectrum2PositiveRealPart [{{φy}} , z, 1]
```

$$\text{LinearStateSpace}\left[\left\{\{\{0.7\}\},\{\{1.\}\},\left\{\left\{\frac{12+2\pi\lambda}{4\pi}\right\}\right\}\right\}\right]$$

The result is a linear state space model.

3.3 Spectral factorization

There is also a facility to compute the spectral factorization of a spectrum. This is done in two steps where first the positive real part of the spectrum is computed and then the spectral factorization. The algorithm for the latter uses a Riccati equation solver that at present only is able to handle numerical data.

Example 4. (Example 3 continued). The spectral factorization of Φ_Y in Example 3, with $\lambda = 2$, is computed as follows:

```
H = SpectralFactorization[F/.λ → 2]
```

$$\text{LinearStateSpace}[\{\{\{0.7\}\},\{\{0.39\}\},\{\{1.37\}\},\{\{1.85\}\}\}]$$

The result is a linear state space model.

4. SPECTRAL ANALYSIS

There are functions available for computing sample correlations and smoothing such estimates, and from these estimates also non-parametric spectral and frequency function estimates (spectral analysis). We illustrate this with an example.

Example 5. We generate 500 samples from the system $G(z) = 0.6 + 0.3z^{-1} + 0.1z^{-2}$:

```
u = {Table[RandomReal[NormalDistribution[0, 1]], {500}]};
y = {Flatten[{MovingAverage[Flatten[u], {0.6, 0.3, 0.1}], 0, 0}]};
G[z_]:={{0.6 + 0.3/z + 0.1/z^2}};
```

Using a window of size $M = 50$, a frequency function estimate of G is obtained by first computing sample correlations and then performing spectral analysis:

```
Ruu=SampleCorrelationFunction[u,u,-M,M];
Ryu=SampleCorrelationFunction[u,y,-M,M];
```

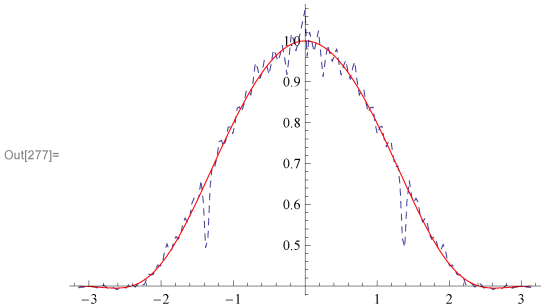

Fig. 2. Magnitude plot of frequency function estimate obtained using spectral analysis together with the true frequency function in Example 5.

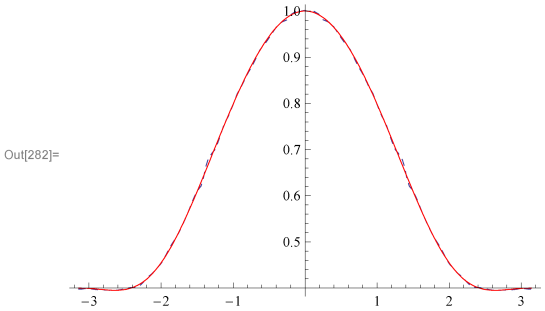

Fig. 3. Magnitude plot of frequency function estimate obtained using spectral analysis on smoothed sample correlations together with the true frequency function in Example 5.

```
Ryy=SampleCorrelationFunction[y,y,-M,M];
{Ghat,Seehat}=SpectralAnalysis[Ruu,Ryu,Ryy];
```

The result is shown in Figure 2. It is simple to smooth the estimates:

```
Ruus = Smooth[Ruu, Hamming];
Ryus = Smooth[Ryu, Hamming];
Ryys = Smooth[Ryy, Hamming];
{Ghats, Seehats} = SpectralAnalysis[Ruus, Ryus, Ryys];
```

which results in the estimate in Figure 3.

5. IDENTIFICATION OF NON-LINEAR SYSTEMS

5.1 Parameter estimation

The tool supports prediction error identification of the parameters in the nonlinear models. Due to the symbolic computational possibility in *Mathematica* it is easy to handle any criterion of fit. Hence, there are some standard alternatives like the sum of squared errors, but the user can also specify any criterion of their choice if they prefer.

Initial parameter estimate The criterion of fit is minimized by a gradient based iterative algorithm, typical Gauss-Newton or levenberg-Marquardt, but any of the supported algorithms of *Mathematica* can be chosen. The minimization starts at an *initial parameter estimate* and its quality is crucial for the success of the minimization. With a bad initialization the algorithm can be stuck in a local minima corresponding to a model far from the best possible one.

The tool supports algorithms to define model structures and produce reasonable initial parameter values where the risk of getting stuck in local minima is reduced. These algorithms are described in Sjöberg (1997).

Stability issues One requirement for the prediction error algorithm is that the model and its derivative with respect to the parameters are stable for the parameter values. For linear models this is easily tests and monitored during the minimization. Stability of nonlinear models is however not a system property and it depends in general also in the input signal. The tool has algorithms which facilitates definitions of model structures and parameter initializations which gives stable models. These are described in Sjöberg and Ngia (1998).

Initial state estimation The initial state of the model is estimated together with all parameters of the model. For state-space models the user can influence the estimation in various ways, for example, indicating some states as known and other as unknowns.

5.2 Nonlinear Model Structures

The tool can handle, basically, any model structure which can be expressed in the state-space form (4). The only requirement is that the derivative of the output exists so that the derivative based criterion minimization can be applied.

Tailored model structures A tailored model means that the user specifies the model equations on the form (4). The functions f and h are specified based on prior knowledge from, for example, physics. The user must also supply the initial parameter estimate.

Black box model structures Nonlinear counterparts of the linear black-box models are supported. Any type of nonlinear basis function can be used for the nonlinear mapping which gives models of the following types NL-FIR, NLARX, NLARMAX, NLBJ, NLAR and NLARMA. These are defined and motivated as in Sjöberg et al. (1995).

Special model structures Well known special nonlinear model structures, such as Wiener models and Hammerstein models are supported. Also, nonlinear model structures with, for example, affine input signal can be obtained. These model can be either state space models or input-output models.

5.3 Stepwise modeling approach

One possible way to work with the tool for nonlinear system identification is to use a *stepwise* approach where nonlinear parts are, stepwise, added. Starting from a simple model, normally a linear one, the tool has special commands for adding nonlinear parts to different places of the original model. The stepwise process, described in Sjöberg and Ngia (1998) and close to that in Bohlin (1991), resembles the general system, identification work process where validation and examination takes place after each model increment. Often a modification is rejected and the user can try to add a nonlinearity to the model in an different way. An illustration of one step of this procedure

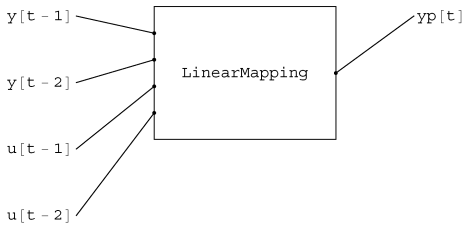

Fig. 4. Graphical illustration of a linear ARX model. This is one of the possible starting points for nonlinear system identification where the the linear mapping is changed to a nonlinear one.

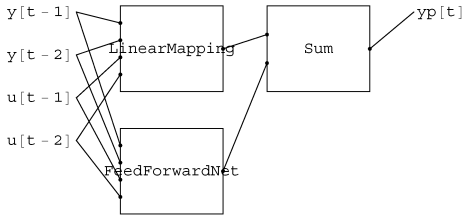

Fig. 5. A NLARX model obtained by adding a block consisting of a feed-forward neural network in parallel to the linear mapping of an ARX model.

can be seen in Figures 4 and 5. Figure 4 illustrates a linear black box model and in 5 a nonlinearity has been added in parallel to the linear model.

The user can choose where to put the added nonlinearity, in parallel, after or before the original block. The user can also choose what kind of nonlinearity to add, and which signals should feed into the new nonlinear part.

The stepwise procedure can be used in combination with all supported nonlinear model structures.

This stepwise approach makes it possible to initialize the parameters in the added nonlinear part so that stability is preserved which increases the possibility to obtain a good model.

The stepwise approach is further illustrated in the papers Sjöberg (2000); Sjöberg and Gutman (1999).

5.4 Model validation

Standard tools for model validation are supported such as simulation, prediction, correlation tests where the model output is compared to the measured output.

5.5 Linearization

Nonlinear models can be linearized at any given state and value of the input signal.

6. CONCLUSIONS

While exploring the symbolic opens up new avenues, it is important to keep in mind that the capabilities of handling

several of the operations that commonly appear in signals and systems theory at present is restricted to very simple problems. One example of such an operation is singular value decomposition.

An interesting research topic is how to balance symbolic and numerical computations. It is usually more computationally expensive to make a symbolic solution but once this is available, it is easy to compute the solution for any parameter.

REFERENCES

Antoulas, A., Sorenson, D., and Guercin, S. (2001). A survey of model reduction methods for large-scale systems. *Contemporary Mathematics*, 280, 193–219.

Bohlin, T. (1991). *Interactive System Identification: Prospects and Pitfalls*. Springer Verlag.

Emami-Naeini, A. and van Dooren, P. (1982). Computation of zeros of linear multivariable systems. *Automatica*, 18(4), 415–430.

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

Laub, A. (1981). Efficient multivariable frequency response computations. *IEEE Transactions on Automatic Control*, 26(2), 407–408.

Sjöberg, J. (1997). On estimation of nonlinear black-box models: How to obtain a good initialization. In *IEEE Workshop in Neural Networks for Signal Processing, Amelia Island Plantation, Florida, Sep. 24-26*, 72–81.

Sjöberg, J. (2000). A nonlinear grey-box example using a stepwise system identification approach. In *Proceedings of the 11th IFAC Symposium on Identification, Santa Barbara, USA*.

Sjöberg, J. and Gutman, P. (1999). Nonlinear identification of the position sled dynamics of a cd player. In *Proceedings of the 7th Mediterranean Conference on Control and Automation (MED99), Haifa, Israel*, 738–751.

Sjöberg, J. and Ngia, L. (1998). Neural nets and related model structures for nonlinear system identification. In J. Suykens and J. Vanderwalle (eds.), *Nonlinear Modelling, Advanced Black-Box Techniques*, 1–28. Kluwer Academic Publisher.

Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Deylon, B., Glorennec, P.Y., Hjalmarsson, H., and Juditsky, A. (1995). Non-linear black-box modeling in system identification: a unified overview. *Automatica*, 31(12), 1691–1724.