SYMBOLIC SIGNAL PROCESSING AND SYSTEM ANALYSIS

Miroslav Lutovac and **Dejan Tošić**
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia, Yugoslavia

Abstract – We present a new software in MATLAB and *Mathematica* for symbolic signal processing and system analysis. Our mission is to encapsulate high-tech engineering and sophisticated mathematical knowledge into easy-to-use software that effectively solves practical problems.

INTRODUCTION

The design of signal processing systems typically involves (1) a high-level specification of the requirements of the system, such as speed, computational efficiency and modularity, (2) development of an appropriate algorithm to accomplish these requirements, and (3) the implementation of the algorithms in an appropriate technology. A design environment that remap signal processing algorithms, specified at a high level of abstraction, to algorithmic description that are more efficient in the context of particular implementation can be generated by experienced systems designers. A highly desirable outcome, at least in the short run, is an environment that achieves fast and efficient designs of signal processing algorithms.

Traditionally, signal processing used computers as high speed number crunchers for doing arithmetic. Computer came to be recognized as a symbol processor, and numbers are only one of the kinds of symbols that the computer can handle [1]. In this paper we demonstrate the advantages of signal processing environments that incorporate symbolic algorithm manipulation along with numerical processing. In particular, we focus on developing automated tools to support algorithm manipulation, symbolic analysis and symbolic design of signal processing systems.

We are creating new software modules that effectively solve practical problems, and provide solutions for wide range of users who need not to master complex mathematical background and algorithms. We are designing new software for signal processing that integrates analysis and design procedures into comprehensive easy-to-use ready-to-run *Mathematica* notebooks and MATLAB scripts. Our software integrates interactive graphics, high quality schematics, text, and "live" code providing immediate access to industry-tested software such as the MATLAB Signal Processing Toolbox or the *Mathematica* Control System Professional Pack (CSP) and Signals and Systems Pack (SSP).

Our software fully exploits power environments, such as *Mathematica* and MATLAB, in symbolic signal processing and system analysis and design. It combines very rich environments for the mixed symbolic-numeric signal processing needed in engineering. Principle targets of our software are (a) practitioners who are short of time to master theoretical background of the design procedures and algorithms, (b) educators who can write interactive lessons and solution sets, (c) students who want more efficient, and practical learning, and (d) industry designers responsible for products with short time-to-market.

Major benefits of our approach are (a) interactive graphic interface for analysis and design of systems, (b) high quality schematics, (c) easy-to-use ready-to-run software for deriving the

system characteristics directly from the schematic, and (d) post-processing of the results in other software, such as MATLAB Simulink or *Mathematica* CSP or SSP.

SYMBOLIC MATHEMATICS PROGRAMS

We choose two well-known symbolic mathematics programs MATLAB and *Mathematica*, the standardized high-level programming languages, as symbolic platforms for knowledge-based signal processing.

MATLAB

MATLAB is an environment for numerical analysis and computing [2, 3]. It is an inexpensive and easy-to-use software package and widely available commercial product which is in widespread use in both academia and industry. MATLAB is a powerful programming environment containing many built-in functions for doing signal processing, linear algebra and other mathematical calculations. MATLAB and toolboxes make possible the users to focus on the problem being solved rather than on the programming necessary to obtain a solution. MATLAB provides appropriate programs (scripts, functions) and background to proceed into areas such as communications, control systems, digital and analog signal processing. The open-system philosophy of MATLAB enables users to write new functions whenever the built-in functions fail to do a vital task.

The MATLAB **Signal Processing Toolbox** (SPT) provides a rich, customizable framework for digital signal processing (DSP). Built on a solid foundation of filter design and spectral analysis techniques, this toolbox contains powerful tools for algorithm development, signal and linear system analysis, and time-series data modeling. The SPT salient features are signal and linear system models; digital and analog filter design, analysis, and implementation; FFT, DCT, and other transforms; spectrum estimation and statistical signal processing; parametric time-series modeling; waveform generation; windowing; and export to embedded DSP platforms. **Simulink** is an interactive dynamic simulation environment for modeling, simulating, and analyzing dynamic, multi-domain systems. The designer can simulate various systems, generate C code for real-time prototyping and implementation, build a block diagram, simulate the system's behavior, evaluate its performance, and refine the design. Simulink uses the metaphor of a block diagram to represent a system. Instead of drawing the individual blocks, blocks are copied from libraries of blocks. The computational kernel of **Maple**, integrated into the MATLAB **Symbolic Toolbox**, performs computations using symbolic mathematics and variable precision arithmetic.

Mathematica

Mathematica is a general computer software system intended for mathematical applications. It can be used as (1) a numerical and symbolic calculator, (2) a visualization system, (3) a high-level programming language, (4) a modeling and data analysis environment, (5) a system for representing knowledge, (6) a software platform, and (7) a way to create interactive documents that mix text, graphics, sound and formulas. *Mathematica* handles numerical, symbolic and graphical classes of computation in a unified way [3, 4].

The **Signals and Systems Pack** (SSP) is a signal processing tool with complete symbolic power. SSP is a *Mathematica* implementation of the necessary tools for working with signals (functions) and linear systems (operators) and it is based on the transform theory. SSP can perform a variety of symbolic, graphical, and numerical operations on signals and systems.

Digital signal processing deals with discrete-time signals (sequences) and systems that act on them. The signals are viewed as a stream of numbers. The computer provides an efficient way to compute the DSP operator that transforms one stream of numbers into another. On the other hand, the design of a signal processing system treats the signals as functions in the mathematical sense. In symbolic processing, the signal is represented in a computer as a formula, rather than as a sequence of numbers [1]. Thus, the value of a signal might only be known in terms of a formula, instead of a number. In a similar manner, signal processing operators, the building blocks for systems, are maintained in symbolic form. This enables a machine to simplify, rearrange, and rewrite symbolic expressions until they take a desired form. When one of the operators is applied to a function, no evaluation takes place; the resulting function is stored in the symbolic form until it becomes convenient to compute it explicitly.

DrawFilt AND DFSYM SOFTWARE

Symbolic analysis is a modern technique, for analyzing systems and circuits, which gives insight in the system behavior. The linear, time-invariant systems and electric circuits are the focus of symbolic analysis. The system parameters, such as multiplier coefficients of a digital filter, are kept as symbols and various transfer functions are computed analytically as closed-form expressions rather than sequences of numbers. In this paper we consider linear, time-invariant, discrete-time systems. Our goal is to derive the transfer function of these systems, such as digital filters, by using MATLAB.

We present our original software **DFSYM** for symbolic computation of the transfer function of an arbitrary time-invariant, linear, discrete-time system. The software uses standard MATLAB commands and MATLAB Symbolic Toolbox. We automate the transfer function computation by deriving it directly from the schematic created by **DrawFilt** [5]. DrawFilt is our original software, written in MATLAB, for drawing schematics of systems and circuits in MATLAB figure window. It allows users to use mouse point and click operations to draw schematics with the following properties: (a) the program is easy to use, intuitive and is based on standard built-in MATLAB commands, (b) standard file handling routines are provided to save existing figures and to open them for modification, (c) the schematic can be saved and called as stand-alone schematic function, (d) the schematic function can be used in MATLAB script or by another function to illustrate or document analyzed system, and (e) the schematic that are produced are of publication quality and can be imported into word processor as eps, wmf or bmp figures.

DFSYM finds transfer function of any system with arbitrary, real or complex, coefficients. The transfer function is presented as an analytic expression in terms of the z-variable and the symbols representing system parameters (such as the multiplier coefficients). Using DFSYM is a simple, straightforward and intuitive three-step process: (1) run MATLAB, and run DrawFilt from the MATLAB command window, (2) draw the schematic of the system that you want to analyze symbolically, or open the existing schematic file from your working directory, (3) run DFSYM from the MATLAB command window. DFSYM automatically (a) scans the components that constitute the system, (b) counts and reports annotation components (Text, Node, PolyLine), line components, and the components which are effectively used in symbolic analysis (Adder, Multiplier, Delay, Block, Input, and Output), (c) checks for errors in the schematic (missing input, missing output, floating nodes, etc.), (d) assigns to nodes consecutive integer numbers starting from one, (e) sets up the required equations, (f) draws another schematic on which possible errors are indicated, and designation of the variables Y1, Y2, ..., is shown, (g) carries out symbolic analysis and computes the

transfer function. The results of the symbolic analysis are shown in the MATLAB command window.

Figure 1 shows a portion of a typical DrawFilt session and DFSYM error handling. The figure illustrates an obvious error in the schematic – a missing line between two nodes. DFSYM puts boxes around the floating nodes, so the user can easily identify the problem in a large schematic that contains many components.

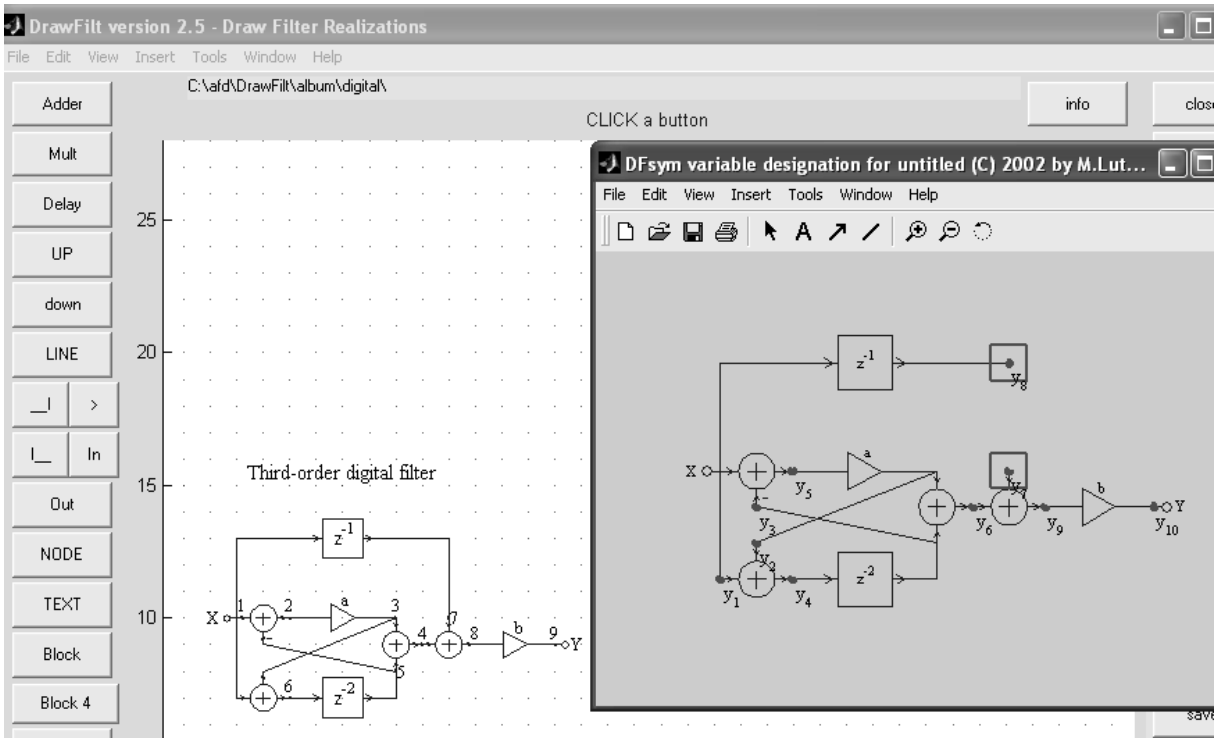

Figure 1. Example DrawFilt schematic and DFSYM error handling demonstration.

For the system drawn on Figure 1, DFSYM computed symbolically the transfer function as follows:

```
Transfer function H(z) = Y9/Y1 =

                          2
(z + 1) b (a z  + z - z a + a)
------------------------------
                  3
                 z  + z a
```

SchematicSolver AND ADVANCED SIGNAL PROCESSING PACK

SchematicSolver is our new *Mathematica* application for creating quality schematics of systems, and for symbolic solving the systems directly from the schematic. The results returned by SchematicSolver can be immediately used for further post-processing with *Mathematica* power functions or packs such as Signals and Systems, and Control System Professional.

With SchematicSolver users can easily (a) derive the equations describing the system, (b) compute the symbolic system response and transfer functions, and (c) generate the software implementation. Users don't have to be skillful in *Mathematica*, nor experts in signal processing, to fully exploit SchematicSolver. SchematicSolver has intuitive interface and

comprehensive online documentation that leads the user step-by-step through the process of creating and analyzing the schematic.

For signal processing experts, SchematicSolver is a quick solution to the frequently used systems. Moreover, the user can exploit the power of *Mathematica* to full extent for additional processing of symbolic results, such as, mixed symbolic-numeric optimization not available with other software. In short, our paradigm is straightforward: Draw-Solve-Implement.

Advanced Signal Processing Pack (ASPP) is a collection of *Mathematica* packages and notebooks for signal processing and system design. ASPP combines mixed symbolic-numeric signal processing. It integrates analysis and design procedures into comprehensive easy-to-use ready-to-run *Mathematica* notebooks. Also, it fully exploits *Mathematica* in symbolic signal processing and system analysis and/or design.

ASPP solves real-life problems. It has easy-to-use self-contained notebooks and a collection of systems – schematics that are most frequently used. Each notebook contains the system schematic, the analysis equations, the design equations, and the signal processing procedures. ASPP is ready-to-use without programming – the user enters the system parameters, only.

Figure 2 presents a portion of a SchematicSolver notebook. It shows three principle parts: (1) the system description as a list of components, (2) the system graphic representation, and (3) the palette for placing schematic elements, that is, for drawing the schematic.

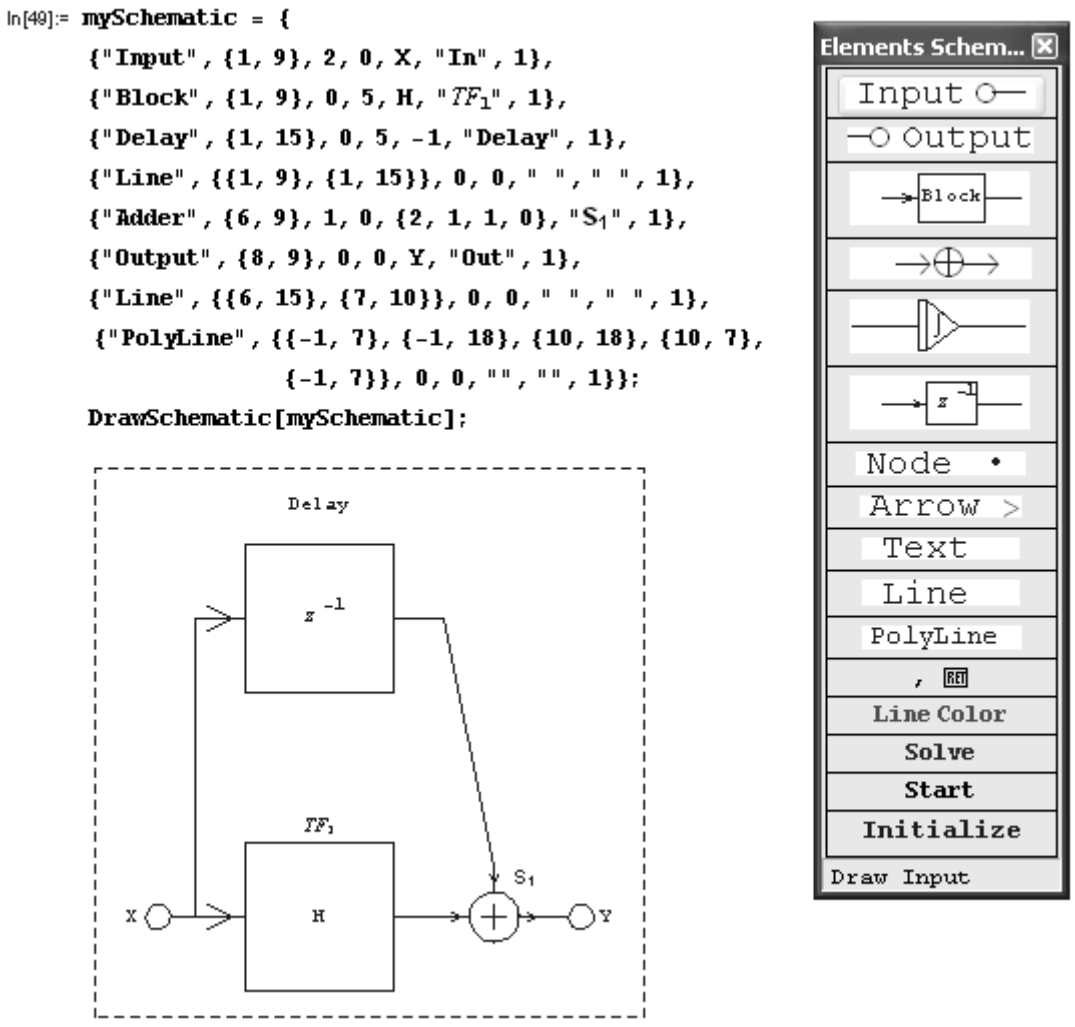

Figure 2. Example SchematicSolver system representation and the drawing palette.

The **Solve** palette button launches the symbolic analyzer that computes the system characteristics, such as the transfer function. For the system drawn on Figure 2, SchematicSolver computed symbolically the transfer function as follows:

```
In[22]  =  Y[{8, 9}]/X /. tf[[1]] // Simplify
Out[22] =  H + z^-1
```

CONCLUSIONS

We propose software that is appropriate for (a) drawing system schematics in MATLAB and *Mathematica*, (b) generating system equations, (c) computing response and system characteristics, (d) constructing system realization, (e) symbolic-numeric digital signal processing, and (f) symbolic-numeric analog signal processing.

The proposed software solves real-life problems and involves (a) easy-to-use self-contained notebooks and scripts, (b) collection of systems – schematics that are most frequently used, (c) each notebook/script contains system schematic, analysis equations, design equations, (d) signal processing procedures, and (e) ready-to-use modules without programming – user enters the system parameters only.

Traditional powerful programs, such as MATLAB Simulink and SPICE, are numeric-only software; the presented software performs both, symbolic and numeric analysis.

Our symbolic design algorithms were used in symbolic performance optimization of analog and digital systems that is not possible by using traditional numeric algorithms [6 - 8]. Closed form formulas were derived for designing high-speed low-consumption systems known as quadrature mirror filter banks. Also, using symbolic optimization, a very efficient digital signal processing systems were implemented using programmable logic devices and very large-scale integrated circuits.

ACKNOWLEDGMENTS

This work was supported in part by the Serbian Ministry for Science, Technologies and Development (contract IT.1.21.0045.B, 2002) and Wolfram Research, Inc. (Visiting Scholars Grant Program, July 2002).

References

[1] A. V. Oppenheim and S. H. Nawab Eds., *Symbolic and Knowledge-Based Signal Processing*, Prentice Hall, 1992.
[2] The MathWorks, Inc., *Using MATLAB*, Version 6.5 Release 13, Natick, MA, 2002.
[3] M. D. Lutovac, D. V. Tošić and B. L. Evans, *Filter Design for Signal Processing Using MATLAB and Mathematica*, Prentice Hall, 2001.
[4] S. Wolfram, *The Mathematica Book*, Wolfram Media, Cambridge University Press, 1999.
[5] M. D. Lutovac and D. V. Tošić, "DRAWFILT - Drawing Filter Realizations in MATLAB", in *PC Programs for Engineers*, IEEE Circuits and Devices Magazine, L. P. Huelsman Ed., vol. 17, no. 1, pp. 3-4, Jan. 2001.
[6] Lj. Milić and M. D. Lutovac, "Efficient Multirate Filtering", in *Multirate Systems: Design and Applications*, Idea Group Publishers, G. Dolecek Ed., 2002, pp. 105-142.
[7] P. Karantzalis, "Free FilterCAD 3.0 Software Designs Filters Quickly and Easily*", Linear Technology Design Notes*, No. 245, Dec. 2000.
[8] Lj. D. Milić and M. D. Lutovac, "Design of Multiplierless Elliptic IIR Filters with a Small Quantization Error*", IEEE Trans. Signal Processing*, vol. 47, no.2, pp.469-479, Feb.1999.