MATLAB LANGUAGE AS TOOL FOR SYMBOLIC, SEMI-SYMBOLIC AND NUMERICAL SIMULATION OF ELECTRICAL NETWORKS

Juraj Valsa

ABSTRACT
The paper deals with the application of MATLAB mathematical language [2] for simulation of electrical and electronic networks. A library of about 40 m-files (scripts and functions) is described that solves the following problem: automatic formulation of network equations in matrix form (both symbolic and numerical); evaluation of numerator and denominator polynomials for network functions; calculation of polynomial coefficients, zeros and poles; calculation and plotting of frequency-domain characteristics and sensitivities to variation of parameters of network elements; calculation and plotting of time-domain responses to unit impulse and step functions, to arbitrary input signals and periodic components of responses to periodic input signals. Virtually all the problems can be solved in various ways and the user has the possibility to choose individual solution procedures and to compare them from the point of view of accuracy, speed of solution and reliability of the results. The m-functions utilize the MATLAB built-in functions to realize most mathematical operations. To satisfy some needs, however, special functions are written. All m-functions are written in readable form and can be easily modified by the user if necessary. The prepared functions are used to support teaching simulation methods at both undergraduate and graduate level of study.

KEYWORDS
Simulation of electrical networks, linear networks, equations of networks in matrix form

INTRODUCTION

To support the teaching of circuit theory and simulation methods for electrical and electronic networks at the undergraduate, graduate and doctorate level we, at the Department of theoretical and experimental electrical engineering created a “building set” of MATLAB scripts. The prepared programs allow the students to experiment with the simulation procedures and compare individual possible approaches from the point of view of accuracy, speed of solution or reliability of the results. The overall conception of the programs offers the possibility to modify individual programs and write new versions to realize the original ideas of the students.

INDIVIDUAL SIMULATION LEVELS

The simulation is based on solution of network equations formulated by the modified node voltage method. This method makes it possible to describe a linear (linearized) network containing passive elements *G*, *R*, *C* and *L* (including mutual couplings), all 4 types of controlled sources and ideal operational amplifiers. The network parameters must be constant in time. The resultant network equations have the following form

$$(\mathbf{G} + p\mathbf{C})\,\mathbf{X} = \mathbf{B}\ ,$$

where
G is the matrix of parameters of non-inertial elements (conductances, resistances, parameters of controlled sources),
C is the matrix of parameters of inertial elements (capacitances, inductances, mutual inductances),
p is the operator of differentiation with respect to time $p = d/dt$,
X is the vector of unknown network quantities (node voltages and added currents),
B is right hand side vector of independent voltage and current sources.

The program solves immittance (i.e. impedance and/or admittance) and transfer functions of the network considered as a two-port. There are altogether 14 simulation levels offering specific forms of the results. Individual simulation levels are coded as follows:

a – netlist, basic tables
b – matrices *G* and *C* in symbolic form
c – matrices *G* and *C* in numerical form
d – coefficients of polynomials in network functions in symbolic form
e – coefficients of polynomials in numerical form
f – zeros and poles
g – frequency responses
h – frequency-domain sensitivities to variation of element parameters
i – group delay *tg*, slope of Bode characteristic *sa*
j – impulse response *g(t)*, step response *h(t)* in semi-symbolic form
k – impulse response *g(t)*, step response *h(t)* in numerical and graphical form
l – response to a general input signal
m – periodic component of response to a periodic input signal.

The user determines the simulation procedure by choosing a suitable sequence of calls to individual scripts. The name of each script has the form *Lxy*. The letter *x* denotes the starting level and the letter *y* the target level of the respective operation. So, for instance, the script *Lef*.m starts with polynomial coefficients and calculates zeros and poles. The script *Lfe.m* on the other hand starts with the roots and delivers polynomial coefficients. All possible sequences of simulation steps are shown in Fig.1.

DETAILED CHARACTERISTICS OF INDIVIDUAL SCRIPTS

Laa.m
Selects the file *.net and reads the netlist. Sets up tables with codes, serial numbers of nodes and parameters of network elements. Determines the number of independent nodes, added currents and total number of network variables (number of equations). Reads the suggested

limits of frequency range and number of solution points, calculates recommended limits of solution and suggests implicit form of standard input signal for time domain.

Lab.m
Based on netlist information, sets up system matrices *G* and *C* in symbolic form.

Lbd.m
Calculates and displays the numerator and denominator polynomials (determinants of the respective matrices) in symbolic form.

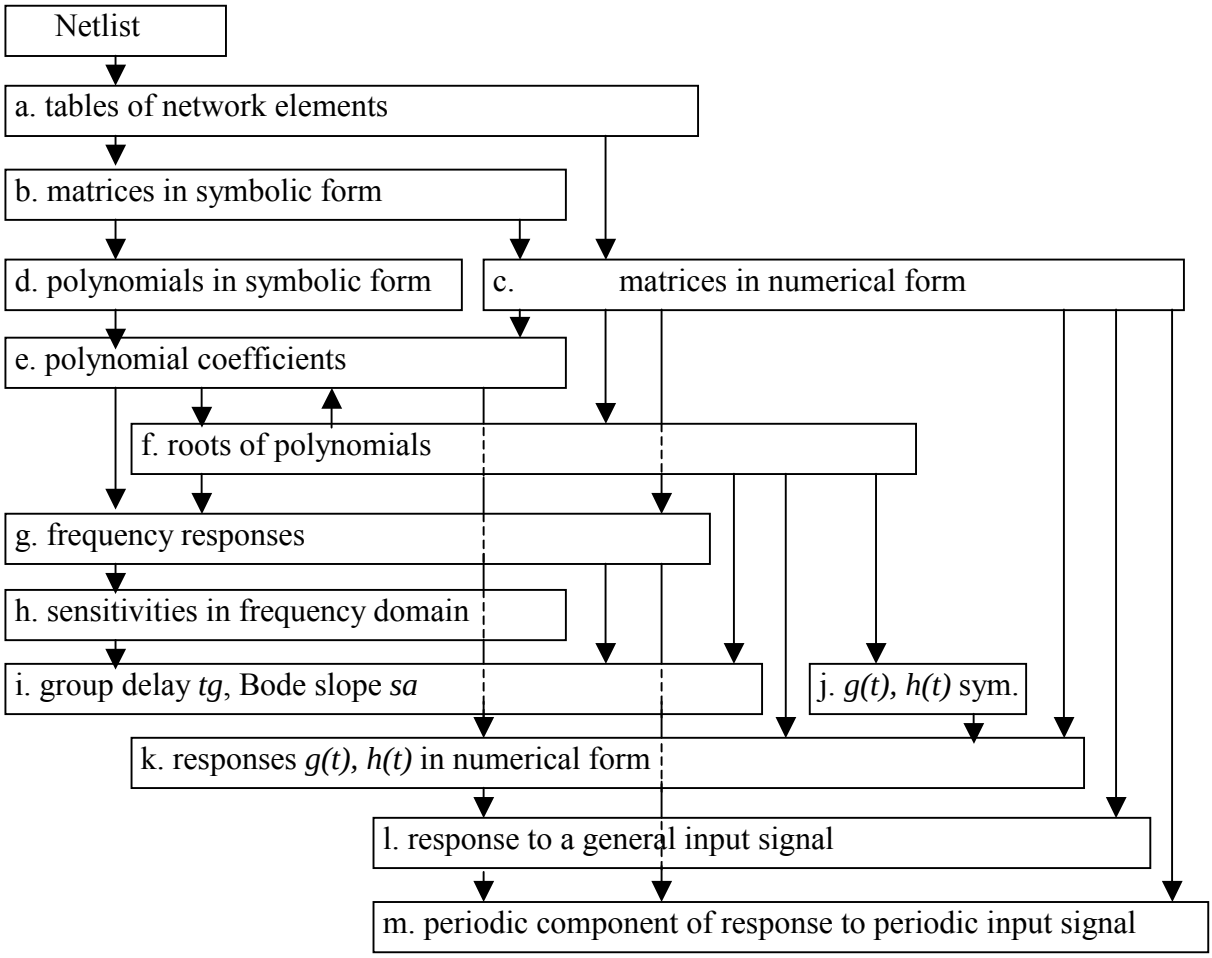

Fig. 1. Flow chart of simulation

Lde.m
Substitutes parameters of network elements into the symbolic polynomials to get polynomial coefficients.

Lbc.m
Substitutes parameters of network elements into the symbolic matrices to obtain numerical forms of these matrices.

Lac.m
Based on netlist information prepares the network matrices *G* and *C* and right-hand side vector *B* in numerical form. According to the input and output ports of the network determines which network function (transfer impedance or admittance, voltage or current transfer factor) is to be simulated and formulates the numerator matrices *Gnum*, *Cnum* accordingly.

Lcg.m
Calculates frequency responses of the selected network function by solving the network equations in complex frequency domain. Calls *setfrq* to specify range of frequencies or angular frequencies and mode of frequency variation (logarithmic or linear). Calls *fchar* to plot graphs of frequency responses (modulus in dB, argument in degrees). Corrects the argument response if necessary. Plots the hodograph.

Lcf.m
Performs semi-symbolic analysis of the network function. Calls *defl* to prepare the matrices *Gnum*, *Cnum* for solving the generalized eigenvalue problem. Determines zeros as negatively taken eigenvalues and value of the coefficient *knum*. In a similar way, deflates matrices *G* and *C* and calculate the poles and coefficient *kden*.

Lfg.m
Calculates frequency responses of the selected network function by using the zeros and poles and coefficients *knum*, *kden*. Calls *fchar* to plot the results.

Lfe.m
Using the known zeros and poles and coefficients *knum*, *kden*, calculates coefficients of the polynomials in numerator and denominator of the simulated network function.

Lce.m
Calculates the coefficients of polynomials in the simulated network function by interpolation in complex frequency domain. Uses test frequencies lying on a normalized circle in complex plane to get complex values of the respective determinants and obtains the polynomial coefficients by help of inverse discrete Fourier transformation.

Lef.m
Using the known polynomial coefficients, calculates the zeros and poles of the simulated network function.

Leg.m
Using the known polynomial coefficients, calculates the frequency responses of the selected network function. Calls *fchar* to plot the results.

Lgi.m
Calculates the group delay *tg* and slope of the Bode modulus characteristic *sa* (in dB/decade) by approximate numerical differentiation of frequency responses. Plots the graphs of results.

Lgh.m
Calculates frequency dependence of absolute sensitivities of the simulated network function to the variation of selected elements. Plots the graphs of results.

Lhi.m
Calculates the group delay *tg* and slope of the Bode modulus characteristic *sa* (in dB/decade) exactly by using the relative sensitivities to the variation of capacitances and inductances in the network. Plots the graphs of results.

Lfi.m
Calculates the group delay *tg* and slope of the Bode modulus characteristic *sa* (in dB/decade) exactly by using the zeros and poles of the network functions. Plots the graphs of results.

Lfj.m
Derives formulae for impulse and step responses by expanding the network functions into partial fractions and evaluating the residue.

Ljk.m
Substitutes numerical values and calculates and displays impulse and step responses.

Lck.m
Calculates impulse *g(t)* and step *h(t)* responses of the network function by numerical inversion of the Laplace transform of the function. Solves the basic network equations with complex frequencies *p* to get the necessary network function samples. There are 4 versions of the procedure based on different approaches to the numerical Laplace transform inversion:
1 - using Padé approximation with M=10, N=8 [5],
2 - using Padé approximation with M=24, N=22 [5],
3 - using approximation with hyperbolic sine [4],
4 - using FFT and epsilon algorithm [1].
Plots the graphs of results.

Lek.m
Calculates impulse *g(t)* and step *h(t)* responses of the network function by numerical inversion of the Laplace transform of the function. Calculates the network function samples using the coefficient of polynomials. Again, there exist 4 possibilities for numerical inversion.

Lfk.m
Calculates impulse *g(t)* and step *h(t)* responses of the network function by numerical inversion of the Laplace transform of the function. Calculates the network function samples using the zeros and poles. Again, there exist 4 possibilities for numerical inversion.

Lkl.m
Finds the response in time domain to an arbitrary input signal by evaluating the convolution integral. Calls *setsig* to describe the input signal waveform. Plots graphs of both the input and output signals.

Lgm.m
Finds the periodic steady state response to an arbitrary periodic input signal. Calls *setsig* to describe the input signal waveform and to determine its period. By discrete Fourier transformation finds amplitudes and phases of a selected number of input signal harmonics and superimposes corresponding responses to get the signal at output. Plots graphs of both the input and output signals.

Lcl.m
Calculates the response to an arbitrary input signal by solving the ordinary differential equations (ODEs in MATLAB terminology) or differential-algebraic equations (DAEs) of the network with the given initial conditions. Uses *ode45* solver for ODEs, *ode15s* solver for DAEs.

Lcm.m
Finds the periodic steady state response to an arbitrary periodic input signal. Calls *setsig* to describe the input signal waveform and to determine its period. Calculates the response by solving the ODEs or DAEs of the network and accelerates the steady state solution by epsilon algorithm.

RECOMMENDED CALLING SEQUENCES

1. Symbolic simulation
   Laa – Lab – Lbd

2. Semi-symbolic simulation
   Laa – Lab – Lbd – Lde – Lef
   or
   Laa – Lab – Lbc – Lce – Lef
   or better
   Laa – Lac – Lce – Lef
   or even better
   Laa – Lac – Lcf – Lfe

3. Frequency responses including group delay and slope of Bode characteristic
   Laa – Lac – Lcg – Lgi
   or
   Laa – Lac – Lcgf – Lcf - Lfi

4. Responses *g(t)* and *h(t)* to standard signals by numerical inversion of Laplace transforms
   Laa – Lac – Lck
   or
   Laa – Lac – Lcf – Lfk
   or
   Laa – Lac – Lcf – Lfe - Lek

5. Responses *g(t)* and *h(t)* to standard signals in semi-symbolic and numerical form
   Laa – Lac – Lcf – Lfj - Ljk

6. Response to an arbitrary input signal
   Laa – Lac – Lck – Lkl
   or
   Laa – Lac – Lcl

7. Periodic part of response to a periodic input signal

Laa – Lac – Lgm
or
Laa – Lac - Lcm

EXAMPLE

Passive RLC low - pass filter

Laa

```
% Cauer3a - passive Cauer 3-rd order LFP with losses
r.2  2 3  1000      R2
c.1  2 3  322.3e-9  C1
c.2  1 3  322.3e-9  C2
c.3  1 2  37.56e-9  C3
l.1  2 1  0.131  104.3  L  rL
ein  1 3  1000      R1
vout 2 3
#
flog 100  10e3  201
```

Lab

```
Gsnum =
[  0,  0, -1,  1]
[  0,  1,  1,  0]
[  0, R2,  0,  0]
[  1,  0, rL,  0]
Csnum =
[ C2+C3,     0,     0,     0]
[   -C3,     0,     0,     0]
[     0,     0,     0,     0]
[     0,     0,     L,     0]
Gs =
[  0,  0,  0, -1,  1]
[  0,  0,  1,  1,  0]
[  0, -1, R2,  0,  0]
[  1, -1,  0, rL,  0]
[ -1,  0,  0,  0, R1]
Cs =
[ C2+C3,   -C3,     0,     0,     0]
[   -C3, C1+C3,     0,     0,     0]
[     0,     0,     0,     0,     0]
[     0,     0,     0,     L,     0]
[     0,     0,     0,     0,     0]
```

Lbd

```
Numerator polynomial
    2
   s C3 R2 rL + s  C3 R2 L + R2
Denominator polynomial
                                                      3
  (C2 R2 R1 C3 L + C2 R2 R1 C1 L + C3 R2 R1 C1 L) s  + (C3 R1 L
```

```
  + C3 R2 R1 C1 rL + R2 C1 L + C2 R2 R1 C1 rL + C2 R1 L + C3 R2 L

                     2
 + C2 R2 R1 C3 rL) s  +

(C2 R2 R1 + C2 R1 rL + C3 R1 rL + R2 C1 rL + C3 R2 rL + L + R2 R1 C1)
s

+ R2 + R1 + rL

Lde
Numerator polynomial

            2890610699308390289         2904467893456561     2
           ---------------------- s + ---------------------- s  + 1000
           73786976294838206464O      590295810358705651712
num =
 -4.9204e-006 -3.9175e-003 -1.0000e+003
Denominator polynomial

     1267829902514513      3     20333177801922345379    2
  ---------------------- s  + ----------------------- s
  75557863725914323419136      188894659314785808547840

           153242506619282151029     21043
         + ---------------------- s + -----
           180143985094819840000      10
den =
  1.6780e-008  1.0764e-004  8.5067e-001  2.1043e+003


Lac
G =
            0            0 -1.0000e+000  1.0000e+000
            0  1.0000e-003  1.0000e+000            0
  1.0000e+000 -1.0000e+000  1.0430e+002            0
 -1.0000e+000            0            0  1.0000e+003

C =
  3.5986e-007 -3.7560e-008            0            0
 -3.7560e-008  3.5986e-007            0            0
            0            0  1.3100e-001            0
            0            0            0            0


Lcf
Multiplying coefficient and roots of numerator
knum =
 -4.9204e-009
zer =
 -3.9809e+002 -1.4251e+004i
 -3.9809e+002 +1.4251e+004i
Multiplying coefficient and roots of denominator
kden =
  1.6780e-011
```

```
pol =
 -1.6562e+003 -6.1381e+003i
 -1.6562e+003 +6.1381e+003i
 -3.1027e+003
koef =
  2.9323e+002
```

Lcg
frequency range and number of frequency points as given in netlist

Lgm
input signal triangular; rise time 1e-3, fall time 0.5e-3, period=3e-3
Fourier expansion with 10 harmonic components. See Fig.3.

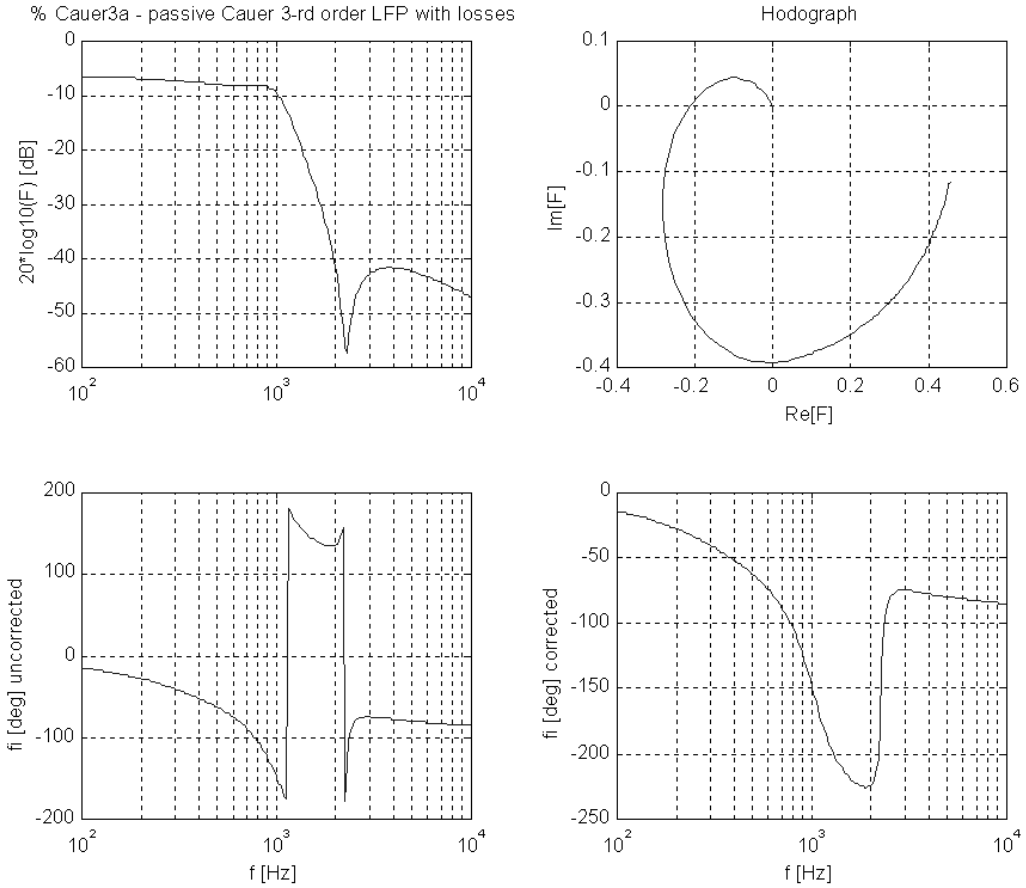

Fig. 2. Frequency-domain responses of the low-pass filter Cauer3a

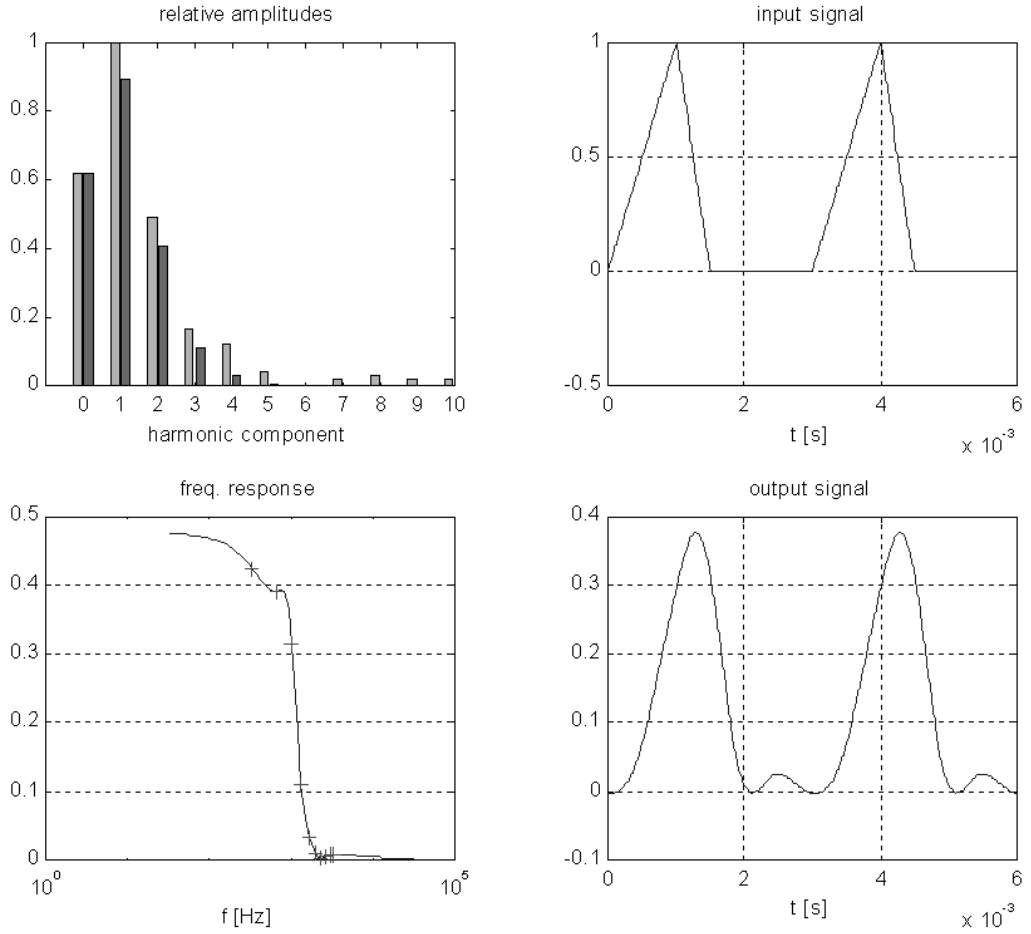

Fig. 3. Periodic triangular signal at input and output of the low-pass filter Cauer3a

CONCLUSION

The described program units are written for simulation of linear or linearized electrical and electronic network with lumped parameters. Future work will be concentrated on the problem of simulation of combined lumped-distributed networks which also contain sections of transmission lines or distributed *RC* or *RL* elements and networks with non-linear elements.

REFERENCES

[1] Brančík, L.: Programs for Fast Numerical Inversion of Laplace Transforms in MATLAB Language Environment with Some Applications, Proceedings of the 23rd Conference SPETO 2000, Gliwice – Ustron (Poland), pp. 197 – 200

[2] MATLAB, The Language of Technical Computing, Version 5.3, Release 11. The Math Works, Inc., 1999

[3] Valsa, J.: Simulation of electrical and electronic networks in MATLAB (in Czech), Proc. of the 7th Conference MATLAB'99, Prague 1999, pp. 213 – 226

[4] Valsa, J., Brančík, L.: Approximate Formulae for Numerical Inversion of Laplace Transforms, International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, Vol. 11, (1998), pp. 153 – 166

[5] Vlach, J., Singhal, K.: Computer Methods for Circuit Analysis and Design, second edition, Van Nostrand Reinhold, New York 1994

Professor Juraj Valsa
Faculty of Electrical Engineering and Computer Science
Brno University of Technology
Purkynova 118, 612 00 Brno, Czech Republic

E-mail: valsa@utee.fee.vutbr.cz