Access provided by:
DUBLIN CITY UNIVERSITY

Sign Out
Attention Authors

Matlab based time-domain simulation of multiconductor transmission line systems

Publisher: IEEE Cite This PDF

L. Brancik **All Authors**

Downl
PDF

Abstract:
The paper deals with a technique of the time-domain domain simulation of transient phenomena in linear distributed systems using Matlab language. The distributed parts ar... **View more**

Metadata

Abstract:
The paper deals with a technique of the time-domain domain simulation of transient phenomena in linear distributed systems using Matlab language. The distributed parts are formed by multiconductor transmission lines (MTL) which can generally be nonuniform, frequency-dependent, and under nonzero initial conditions. In principle the solution is formulated in terms of a modified nodal admittance (MNA) equation method in the frequency domain. Then an improved fast method of numerical inversion of Laplace transforms (NILT) in vector or matrix form is applied to gain a solution in the time domain. This NILT method is based on the FFT in conjunction with the quotient-difference algorithm of Rutihauser to ensure both high speed of computation and sufficient accuracy of results.

Published in: The IEEE Region 8 EUROCON 2003. Computer as a Tool.

Date of Conference: 22-24 September 2003

DOI: 10.1109/EURCON.2003.1248066

Date Added to IEEE *Xplore*: 03 December 2003

Publisher: IEEE

Print ISBN:0-7803-7763-X

Conference Location: Ljubljana, Slovenia

Contents

SECTION I.
Introductlon

A possible method which is general enough to be used for an analysis of lumped-distributed circuits is a modified nodal admittance equation method (MNA) [1], [2]. In [3] this technique is applied for the time-domain simulation of multiconductor transmission line systems using Matlab language. It can be illustrated by a block diagram in Fig. 1.

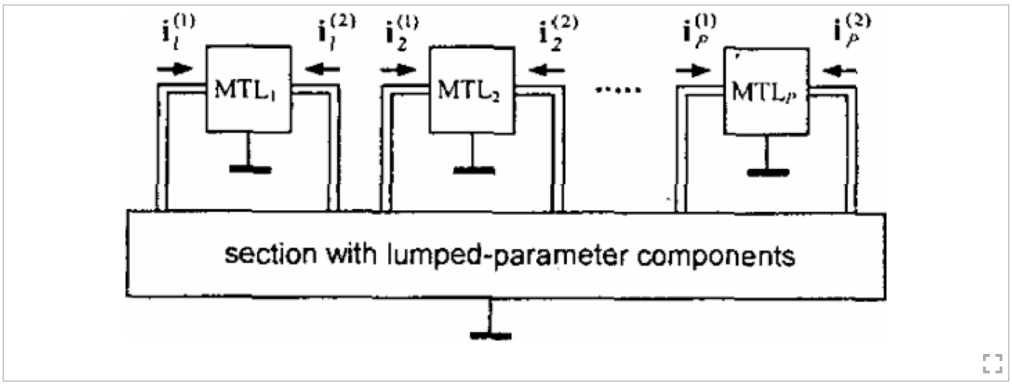

Fig. 1:
Linear system with MTL sections

Therein the MNA matrix equation describing MTL systems under nonzero initial conditions is formulated and an effectiveness of the solution in terms of the Matlab language is shown. Unlike [1], [2], where modal analysis technique is used, admittance matrices of the MTLs are computed by means of the chain matrices. In this way inhomogeneities of the MTLs can easiIy be taken into account if necessary. To incorporate nonzero initial conditions on the MTLs matrix convolution integrals must be solved. In Matlab language, this can effectively be made by the FFT when three-dimensional arrays are utilized [4].

From general point of view, the solution is performed in the frequency domain and then a fast NILT method in vector or matrix forms is used to obtain the solution in the time domain. Unlike mentioned work [3] an improved NIL T method based on the FFT and quotient-difference algorithm of Rutishauser is here used to ensure both high speed of the computation and the sufficient precision of the results. Moreover) to save a CPU time considerably the Matlab language capabilities to process multidimensional arrays in paralle1 are utilized with advantages.

SECTION II.
Mna Matrix Equation Formulation

As is shown e. g. in [1], [2] the modified nodal admittance matrix equation in the time dom, ain can be written as

$$C_M\frac{d\mathrm{v}_M(t)}{dt}+\mathrm{G}_{At}\mathrm{v}_M(t)+\sum_{k=1}^{l}\mathrm{D}_k\mathrm{i}_k(l)=\mathrm{i}_M(t), \tag{1}$$

View Source

where C_M and G_M are $N\times N$ constant matrices with entries determined by the lumped memory and mernoryless components) respectively, $\mathrm{v}_M(t)$ is the $N\times 1$ vector of node voltages appended by currents of independent voltage sources and inductors, $\mathrm{i}_M(t)$ is the $N\times 1$ vector of source waveforms, $\mathrm{i}_k(t)$ is the $n_k\times 1$ vector of currents entering the k·th MTL, and D_k is the $N\times n_k$ selector matrix with entries $d_{i,j}\in\{0,1\}$ mapping the vector $\mathrm{i}_k(t)$ into the node space of the circuit. Applying Laplace transformation the frequency-domain representation has the form

$$[\mathrm{G}_M+\mathrm{sC}_M]\mathrm{V}_M(s)+\sum_{k=1}^{p}\mathrm{D}_k\mathrm{I}_k(s)=\mathrm{I}_M(s)+\mathrm{C}_M\mathrm{v}_M(0). \tag{2}$$

View Source

The MTLs consist of $N_k = n_k/2$ active conductors, i.e. they can be regarded aS $2N_k$. ports. Then the $\mathrm{I}_k(s)$ in (2) is formed to contain vectors of currents entering the input and output ports as $\mathrm{I}_k(s) = [\mathrm{I}_k^{(1)}(s), \mathrm{I}_k^{(2)}(s)]^T$, and they are stated from the basic MTL matrix equation as follows.

Suppose a generally nonuniform MTL of a length l) with per-unit-length matrices $\mathrm{R}(x),\ \mathrm{L}(x),\ \mathrm{G}(x),\ \mathrm{C}(x),\ \ln$ the time domain a basic MTL matrix equation has a form as [5]

$$\frac{\partial}{\partial r}\begin{bmatrix}\mathrm{v}(x,t)\\ \mathrm{i}(x,t)\end{bmatrix} = \begin{bmatrix}0 & -\mathrm{R(x)}\\ \mathrm{G}(x) & 0\end{bmatrix}\cdot\begin{bmatrix}\mathrm{v}(x,t)\\ \mathrm{i}(x,t)\end{bmatrix} - \begin{bmatrix}0 & \mathrm{L}\\ C(x) & 0\end{bmatrix}\cdot\frac{\partial}{\partial t}\begin{bmatrix}\mathrm{v}(x,t)\\ \mathrm{i}(x,t)\end{bmatrix}, \tag{3}$$

View Source

and, after a Laplace transformation is applied, the (3) leads to

$$\frac{d}{dx}\begin{bmatrix}\mathrm{V}(x,s)\\ \mathrm{I}(x,s)\end{bmatrix} = \begin{bmatrix}0 & -\mathrm{Z}(x,s)\\ -\mathrm{Y}(x,s) & 0\end{bmatrix}\cdot\begin{bmatrix}\mathrm{V}(x,s)\\ \mathrm{I}(x,s)\end{bmatrix} + \begin{bmatrix}0 & \mathrm{L}(x)\\ \mathrm{C}(x) & 0\end{bmatrix}\cdot\begin{bmatrix}\mathrm{v}(x,0)\\ \mathrm{i}(x,0)\end{bmatrix}. \tag{4}$$

View Source

Here $\mathrm{V}(x,s) = \mathrm{L}[\mathrm{v}(x,t)]$ and $\mathrm{I}(x,s) = \mathrm{L}[\mathrm{i}(x,t)]$ are column vectors of the Laplace transforms of voltages and currents at a distance x from MTL's left end, respectively, $\mathrm{v}(x,0)$ and $\mathrm{i}(x,0)$ are column vectors of initial voltage and current distributions, respectively and 0 means zero matrix. The $\mathrm{Z}(x,s) = \mathrm{R}(x) + s\mathrm{L}(x)$ and $\mathrm{Y}(x,s) = \mathrm{G}(x) + s\mathrm{C}(x)$ are series iropedance and shunting admittance matrices, respectively. In a compact matrix form the (4) looks like

$$\frac{d}{dx}\mathrm{W}(x,s) = \mathrm{M}(x,s)\mathrm{W}(x,s) + \mathrm{N}(x)\mathrm{w}(x,0). \tag{5}$$

View Source

Then taking $\mathrm{W}(0,s)$ as the solution at $x=0$ (MTL s input) the solution at $x=l$ (MTL's output) can be written as [6]

$$\mathrm{W}(l,s) = \Phi_0'(s)\mathrm{W}(0,s) + \int_0^l \Phi_\xi'(s)\mathrm{N}(\xi)\mathrm{w}(\xi,0)d\xi, \tag{6}$$

View Source

where $\Phi_0'(s)$ is an integra1 matrix (matrizant) defined with an infinite series of matrix integrals or with so-called product-integral, see e.g. [6] In the case of a uniform MTL the matrix exponential function is used for its exact calculation as

$$\Phi_0'(s)|_{\mathrm{M}(x,s)=\mathrm{M}(s)} = e^{\mathrm{M}(s)s-1}. \tag{7}$$

View Source

In general case, however, only an approximate integral matrix can be calculated. It can be done by dividing the MTL into a sufficiently large number m of sections presupposing $\mathrm{M}(s)$ constant in each of them. Then taking basic property of the matrizant into account a recurrent formula is valid as follows

$$\tilde{\Phi}_0^{x_i}(s) = e^{\mathrm{M}(\zeta_j,s)\Delta x_j}\cdot\tilde{\Phi}_0^{x_{j-1}}(s),\ \text{with}\ \tilde{\Phi}_0^0(s) = \mathrm{E} \tag{8}$$

View Source

as an identity matrix $\Delta x_j = x_j - x_{j-1}, j = 1, 2, \ldots, m,\ x_0 = 0x_m = l$ and $\zeta_j \in \langle x_{j-1}, x_j\rangle$. In the case of a uniform MTL the resuIt is the same as according to (7).

In terms of the multipart theory the integral matrix acts as the chain matrix $\Phi(s)$. Thus after denoting

$$\mathrm{W}(0,s)=\mathrm{W}^{(1)}(s)=[\mathrm{V}^{(1)}(s),\mathrm{I}^{(1)}(s)]^{T}, \tag{9}$$
$$\mathrm{W}(l,s)=\mathrm{W}^{(2)}(s)=[\mathrm{V}^{(2)}(s),-\mathrm{I}^{(2)}(s)]^{T}, \tag{10}$$
$$\int_0^l \Phi'_{\xi}(s)\mathrm{N}(\xi)\mathrm{w}(\xi,0)d\xi=\mathrm{W}^{(0)}(s)=\mathrm{V}^{(0)}(s),\mathrm{I}^{(0)}(s)]^{T} \tag{11}$$

View Source

the MTL can be described by (6) in a decomposed form as

$$\begin{bmatrix}\mathrm{V}^{(2)}(s)\\ -\mathrm{I}^{(2)}(s)\end{bmatrix}=\begin{bmatrix}\Phi_{11}(s) & \Phi_{12}(s)\\ \Phi_{21}(s) & \Phi_{22}(s)\end{bmatrix}\cdot\begin{bmatrix}\mathrm{V}^{(1)}(s)\\ \mathrm{I}^{(1)}(s)\end{bmatrix}+\begin{bmatrix}\mathrm{V}^{(0)}(s)\\ \mathrm{I}^{(0)}(s)\end{bmatrix}. \tag{12}$$

View Source

After some manipulations the admittance equations taking nonzero initial conditions into account have a form as

$$\begin{bmatrix}\mathrm{I}^{(1)}(s)\\ \mathrm{I}^{(2)}(s)\end{bmatrix}=\begin{bmatrix}\mathrm{Y}_{11}(s) & \mathrm{Y}_{12}(s)\\ \mathrm{Y}_{21}(s) & \mathrm{V}_{22}(s)\end{bmatrix}\cdot\begin{bmatrix}V^{(1)}(s)\\ \mathrm{V}^{(2)}(s)\end{bmatrix}-\begin{bmatrix}\mathrm{Y}_{12}(s) & 0\\ \mathrm{Y}_{22}(s) & \mathrm{E}\end{bmatrix}\cdot\begin{bmatrix}\mathrm{V}^{(0)}(s)\\ \mathrm{I}^{(0)}(s)\end{bmatrix}, \tag{13}$$

View Source

where submatrices $\mathrm{Y}_{11}(s)=-\Phi_{12}^{-1}(s)\Phi_{11}(s),\mathrm{Y}_{12}(s)=\Phi_{12}^{-1}(s)\mathrm{Y}_{22}(s)=-\Phi_{22}(s)\Phi_{12}^{-1}(s)$, and $\mathrm{Y}_{21}(s)=\mathrm{Y}_{12}^{T}(s)$ because of a reciprocity of the MTL. In the case of a uniform MTL the equality $\mathrm{Y}_{11}(s)=\mathrm{Y}_{22}(s)$ is then valid. Considering now the *k-th* MTL the (13) can be written in a compact matrix form

$$\mathrm{I}_k(s)=\mathrm{Y}_k(s)\mathrm{V}_k(s)-\mathrm{X}_k(s)\Gamma_k(s). \tag{14}$$

View Source

Finally after substituting (14) into basic Mna equation (2) the resultant MNA equation can be written in the form as

$$\mathrm{V}_M(s)=\left[\mathrm{G}_M+\mathrm{sC}_M+\sum_{k=1}^{P}\mathrm{D}_k\mathrm{V}_k(s)\mathrm{D}_k^T\right]^{-1}\cdot$$
$$\left[\mathrm{I}_M(s)+\mathrm{C}_M\mathrm{v}_M(0)+\sum_{k=1}^{P}\mathrm{D}_k\mathrm{X}_k(s)\mathrm{W}_k^{(0)}(s)\right]. \tag{15}$$

View Source

To solve voltages and currents. at a coordinate x from the beginning (I) of the MTL the (12) can look like

$$\begin{bmatrix}\mathrm{V}(x,s)\\ \mathrm{I}(x,s)\end{bmatrix}=\begin{bmatrix}\Phi_{11}(x,s) & \Phi_{12}(x,s)\\ \Phi_{21}(x,s) & \Phi_{22}(x,s)\end{bmatrix}\cdot\begin{bmatrix}\mathrm{V}^{(1)}(s)\\ \mathrm{I}^{(1)}(s)\end{bmatrix}+\begin{bmatrix}\mathrm{V}^{(0)}(x,s)\\ \mathrm{I}^{(0)}(x,s)\end{bmatrix}, \tag{16}$$

View Source

where $\Phi(x,s)$ is a partial chain matrix computed according to (8) and $[\mathrm{V}^{(0)}(x,s),\mathrm{I}^{(0)}(x,s)]^{\mathrm{T}}=\mathrm{W}^{(0)}(x,s)$ is expressed by the matrix integral (11) while replacing indices l by x As this integral expression is that of a convolution type the method based on the FFT can be used for its calculation. In the work [4] there is proposed to utilize three-dimensional arrays when capabilities of the Matlab language to treat multidimensional arrays in parallel are utilized with advantages. The necessary voltage $\mathrm{V}^{(1)}(s)$ and the current $\mathrm{I}^{(1)}(s)$ appertaining to the k-th MTL can be extracted from the equations

$$\mathrm{V}_k(s)=\mathrm{D}_k^t\mathrm{V}_M(s), \tag{17}$$

View Source

and (14), respectively.

SECTION III.

Advanced Fft-Based Nilt Method

An original $f(t)$ to a Laplace transform $F(s)$ can be expressed by the Bromwich integral

$$f(t)=\frac{1}{2\pi \mathrm{j}}\int_{c-j\infty}^{c+j\infty}F(s)e^{st}ds, \tag{18}$$

View Source

under the basic assumption $|f(t)|\leq Ke^{\varpi}, K$ real positive, α as the exponential order of the real function $f(t), t\geq 0$ and $F(s)$ defined for $\mathrm{Re}[s]>\alpha$. Integrating the integral (18) numerically an approximate formula in the discrete form $\tilde{f}_{k}=\tilde{f}(kT), k=0, \cdots, N-1$, can be derived as [7]

$$\tilde{f}_{k}=\mathrm{C}_{k}\{2\mathrm{Re}[\sum_{n=0}^{N-1}F_{n}z_{k}^{n}+\sum_{n=0}^{\infty}G_{n}z_{k}^{n}]-F_{0}\}, \tag{19}$$

View Source

where

$$C_{k}=\frac{\Omega}{2\pi}\cdot e^{ckT}, z_{k}=e^{-jkT\Omega}, F_{n}=F(c-jn\Omega), \mathrm{G}_{n}=F_{N+n}, \tag{20}$$

View Source

with T and $\Omega=2\pi/(NT)$ as sampling periods in the original and the transform domain, respectively. The error analysis has resulted in an approximate formula for the coefficient c as

$$c\approx a-\Omega/2\pi\cdot\ln E_{r}, \tag{21}$$

View Source

where E_{r} denotes the desired relative error. The finite sum in (19) is evaluated by the FFT supposing $N=2^{m}, m$ integer. This enables to get the set of N points in a single calculation step. Consequently the required maximum time is taken as $t_{m}=(M-1)T$, with $M=N/2$ as the number of resultant computed points. To minimize an error towards its theoretical value E_{r} the infinite sum in (19) must be evaluated as much accurately as possible. For this purpose just the quotient. difference algorithm is used to accelerate its convergence. Thus taking only $2P+1$ first terms of this sum into account the continued fraction is determined as [8]

$$v(z_{k}, P)=d_{0}/(1+d_{1}z_{k}/(1+\cdots+d_{2P}z_{k})), \quad \forall k, \tag{22}$$

View Source

that corresponds to Padé rational approximation of the power series. The q-d algorithm can shortly be explained as follows, see Fig. 2.

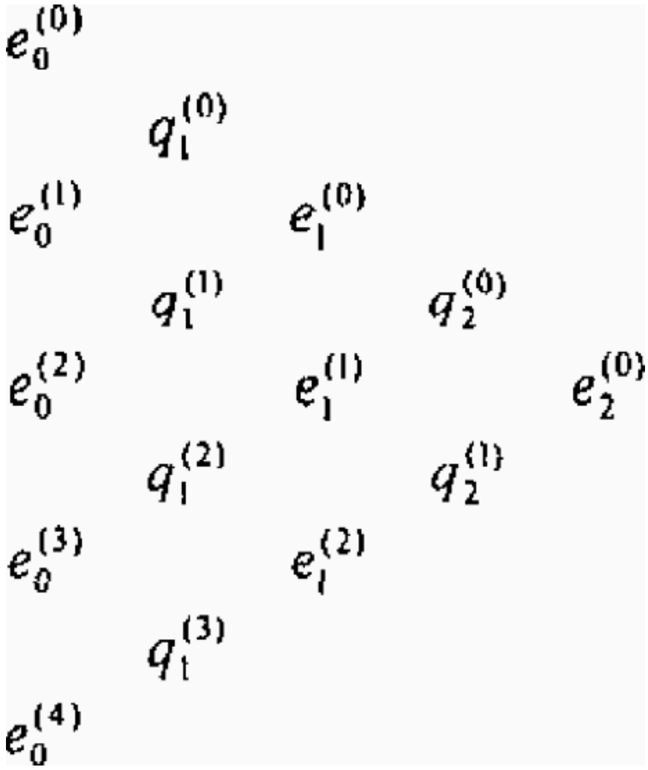

Fig. 2:
Quotient-difference algorithm diagram

The first two columns are formed as

$$e_0^{(j)} = 0, \qquad i = 0, \cdots, 2P, \tag{23}$$
$$q_1^{(i)} = G_{i+1}/G_i, \;\; i = 0, \cdots, 2P-1, \tag{24}$$

View Source

and successive columns are given by the rules as follows:

For $r = 1, \cdots, P,$

$$e_r^{(i)} = q_r^{(i+1)} - q_r^{(i)} + e_{r-i}^{(i+1)} \;\; i = 0, \cdots, 2P-2r, \tag{25}$$

View Source

for $r = 2, \cdots, P,$

$$q_r^{(i)} = q_{r-1}^{(il+1)} e_{r-1}^{(i+1)} / e_{r-1}^{(i)} \;\; i = 0, \cdots, 2P-2r-1. \tag{26}$$

View Source

Then the coefficients $d_n, n = 0, \cdots, 2P$, are given by

$$d_0 = G_0, \;\; d_{2m-1} = -q_m^{(0)} \;\; d_{2m} = -e_m^{(0)}, \tag{27}$$

View Source

$m = 1, \cdots, P$ A practical evaluation of the continued fraction (22) can also be based on recurrent formulae [8]. Finally the $v(z_k, P)$ is used in (19) instead of the original infinite sum.

To invert the equation (15) very fast the vector version of the NILT method is used as follows [9]. ·1 f a transform to be inverted is a vector $\mathrm{F}^{J}(S)=[F_{1}(s),F_{2}(s),\cdots F_{J}(s)]^{T}$ then a NILT formula in a matrix form can be written as

$$\tilde{\mathrm{f}}^{J\times M}=\mathrm{C}^{J\times M}\circ\{2\mathrm{Re}^{J\times M}\{FFT_{<2>}(\mathrm{F}^{J\times N})\}+\mathrm{V}_{f}^{J\times M}]-\mathrm{F}_{0}^{J\times M}\},\tag{28}$$

View Source

where all terms are matrices of upper indexed sizes computed according to (20), but formed for all the vector components. The subscript < 2 > means the FFT operation runs along the 2nd dimension (columns) but in parallel for all the rows. The $\mathrm{V}_{P}^{J\times M}$ is the matrix resulted from (22), the $\mathrm{R}^{J\times M}\{.\}$ denotes the operator of $N\rightarrow M$ matrix dimension reduction and the symbol 0 means Hadamard product of matrices. Similarly to find the time-domain solution (16) the matrix version of the NILT method is the most effective to use [10]. The formula is

$$\tilde{\mathrm{f}}^{J\times M\times L}=\mathrm{C}^{J\times M\times L}\circ\{2\mathrm{Re}[\mathrm{R}^{J\times M\times L}\{FFT_{<2>}(\mathrm{F}^{J\times M\times L})\}+\mathrm{V}_{P}^{J\times M\times L}]-\mathrm{F}_{0}^{J\times M\times L}\},\tag{29}$$

View Source

with terms as 3-dimensional arrays of upper indexed sizes.

SECTION IV.
Matlab Language Implementations

Below the Matlab listings are presented both for the vector and for the matrix version of the NILT method:

```
%******* NILTV – FUNCTION DEFINITION (vector version) *******%
function [ft,t]=niltv(F,tm,pl);
global ft t;
alfa=0; M=256; P=3; Er=1e-10;                          % adjustable
N=2*M; qd=2*P+1; t=linspace(0,tm,M);
NT=2*tm*N/(N-2); omega=2*pi/NT; c=alfa-log(Er)/NT;
s=c-i*omega*(0:N+qd-1); Fsc=feval(F,s); ft=fft(Fsc,N,2);
ft=ft(:,1:M); delv=size(Fsc,1); d=zeros(delv,qd); e=d;
q=Fsc(:,N+2:N+qd)./Fsc(:,N+1:N+qd-1);
d(:,1)=Fsc(:,N+1); d(:,2)=-q(:,1);
for r=2:2:qd-1
  w=qd-r; e(:,1:w)=q(:,2:w+1)-q(:,1:w)+e(:,2:w+1); d(:,r+1)=-e(:,1);
  if r>2
  q(:,1:w-1)=q(:,2:w).*e(:,2:w)./e(:,1:w-1); d(:,r)=-q(:,1);
  end
end
A2=zeros(delv,M); B2=ones(delv,M); A1=repmat(d(:,1),[1,M]); B1=B2;
z=repmat(exp(-i*omega*t),[delv,1]);
for n=2:qd
  Dn=repmat(d(:,n),[1,M]); A=A1+Dn.*z.*A2; B=B1+Dn.*z.*B2;
  A2=A1; B2=B1; A1=A; B1=B;
end
ft=2*real(ft+A./B)-repmat(Fsc(:,1),[1,M]);
ft=repmat(exp(c*t)/NT,[delv,1]).*ft; ft(:,1)=2*ft(:,1);
feval(pl);
%*********************************************************************%

%*************** PLOT1 – FUNCTION DEFINITION ****************%
function pl1                     % multiple plotting into a single figure
global ft t;
plot(t,ft); xlabel('t'); ylabel('f(t)'); grid on;
%*********************************************************************%
%*************** PLOT2 – FUNCTION DEFINITION ****************%
function pl2                     %  plotting into separate figures
global ft t;
for k=1:size(ft,1)
  figure;
  plot(t,ft(k,:)); xlabel('t'); ylabel('f(t)'); grid on;
end
%*********************************************************************%

%******* NILTM – FUNCTION DEFINITION (matrix version) ******%
function [ft,t,x]=niltm(F,tm,pl);
global ft t x;
alfa=0; M=256; P=3; Er=1e-10;                          % adjustable
N=2*M; qd=2*P+1; t=linspace(0,tm,M);
NT=2*tm*N/(N-2); omega=2*pi/NT; c=alfa-log(Er)/NT;
s=c-i*omega*(0:N+qd-1); Fsc=feval(F,s); ft=fft(Fsc,N,2);
ft=ft(:,1:M,:); dim1=size(Fsc,1); dim3=size(Fsc,3);
d=zeros(dim1,qd,dim3); e=d;
q=Fsc(:,N+2:N+qd,:)./Fsc(:,N+1:N+qd-1,:);
d(:,1,:)=Fsc(:,N+1,:); d(:,2,:)=-q(:,1,:);
for r=2:2:qd-1
  w=qd-r; e(:,1:w,:)=q(:,2:w+1,:)-q(:,1:w,:)+e(:,2:w+1,:);
  d(:,r+1,:)=-e(:,1,:);
  if r>2
  q(:,1:w-1,:)=q(:,2:w,:).*e(:,2:w,:)./e(:,1:w-1,:); d(:,r,:)=-q(:,1,:);
  end
end
A2=zeros(dim1,M,dim3); B2=ones(dim1,M,dim3);
A1=repmat(d(:,1,:),[1,M]); B1=B2;
z=repmat(exp(-i*omega*t),[dim1,1,dim3]);
for n=2:qd
  Dn=repmat(d(:,n,:),[1,M]); A=A1+Dn.*z.*A2; B=B1+Dn.*z.*B2;
  A2=A1; B2=B1; A1=A; B1=B;
end
ft=ft+A./B; ft=2*real(ft)-repmat(real(Fsc(:,1,:)),[1,M]);
ft=repmat(exp(c*t)/NT,[dim1,1,dim3]).*ft; ft(:,1,:)=2*ft(:,1,:);
feval(pl);
%*********************************************************************%

%*************** PLOT3 – FUNCTION DEFINITION ****************%
function pl3
global ft t x;
m=length(t); tgr=[1:m/64:m,m];                      % 65 time points
for k=1:size(ft,3)
  figure; mesh(t(tgr),x(k,:),ft(:,tgr,k));
  xlabel('t'); ylabel('x'); zlabel(strcat('f_{',num2str(k),'}'));
end
%*********************************************************************%
```

The niltv and niltm functions are called with 3 parameters: 'F' the name of another function in which Laplace transform in the vector or matrix form is defined, tm the maximum time, "pl the name of the plotting function.

SECTION V.
Example

Consider the circuit with three uniform $(2+1)$-conductor transmission lines in Fig. 3, with the per-unit-length matrices stated in [1]. The MTLs differ only by their lengths, namely $l_1 = 0.05m,\ l_2 = 0.04m, l_3 = 0.03m \cdot$ A 1 V pulse with 1.5 ns rise/fall times and 7.5 ns width is applied at the input.

To gain waveforms of nodal voltages or branch currents a Matlab function describing the solution (15) is called by the niltv function. In Fig. 4 the input and output voltages and the current i_2 are done as the examples. However, to get voltage or current distributions along MTLs' wires a Matlab function describing the solution (16) is called by the niltm function. In Fig. 5 the voltage distributions are done as the examples.

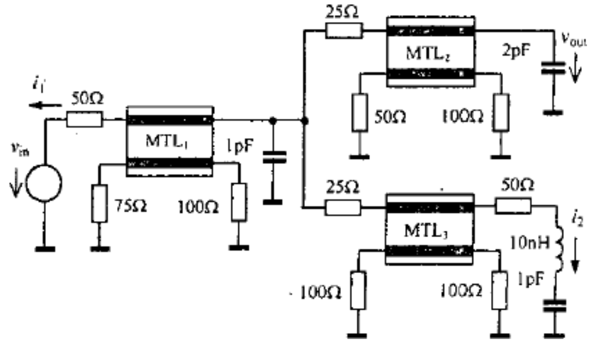

Fig. 3:
Linear circuit containing three MTLs

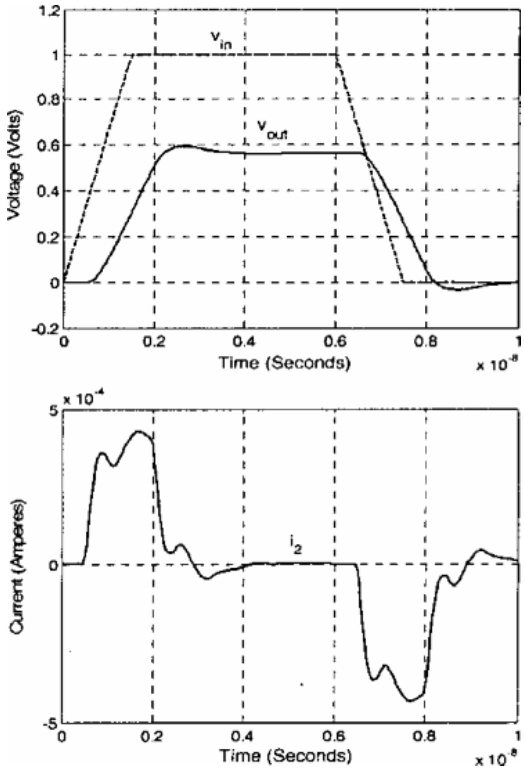

Fig. 4:
Voltage and current waveforms

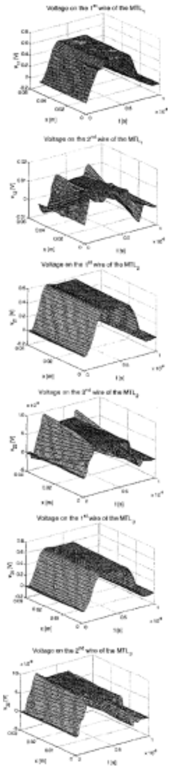

Fig. 5:
Voltage distributions along mtls' wires - cont.

SECTION VI.
Conclusion

All the computations were done on a PC 2G Hz/25 6MB. The CPU times were about 0.5 second and 6 seconds when the niltv and niltm functions were called, respectively. The used novel NILT method based on the FFT and quotient-difference algorithm of Rutishauser leads to approximatelly 20% saving CPU time compared to the previous one using s-algorithm [3]. Moreover, the method seems to be more numerically stable. The results obtained in the example under consideration correspond very well with those gained by the NIL T method in [1] and the AWE technique in [2].

ACKNOWLEDGEMENTS

This work was supported by the grant GACR No. 102/03/0241 and realized within the scope of Research programmes MSM 262200011 and MSM 262200022.

Authors

Figures

References

Citations

Keywords

Metrics

< Previous | Back to Results | Next >