Tuesday – part 3

Petr Kropík

4. Examples of simple transient phenomena – RC circuit.
Settings of graphical outputs and graphical user interface (automated adjustment of figure size in depend on screen size etc.)

Example:
RC circuit

```
function duc=in_val(t,uc)
% diff. equation of RCcircuit (serial), DC el. source

% parameters of circuit settings
R=10;
C=10e-6;
U=50;

% diff. equation
duc=(U-uc)./(R.*C);


function rc_circ
% Solving of diff. equations
% Solve transient phenomena in serial RC circuit
% in case of connecting to DC voltage source in time t = 0
% using m-file in_val

% Numerical solution
% uc(0)=0 (capacitor is not charged at time T0)
% duc/dt = 1/RC*(U-uc), we compute uc
% ic = C*duc/dt
% Runge-Kutta of 2. and 3. order
% [t,uc]=ode23(@in_val,[begin_time end_time],[initial conditions],options);
% function ode23 has more parametrs, usually is possible to use implicit
% parameters can be set using function odeset

% circuit parameters setting
R=10;
C=10e-6;
U=50;

% starting time [s]
T0=0;

% ending time [s]
Tfinal=2e-3;

% initial condition - capacitor is not charged at time T0
Uc0=0;

% accuracy setting
% tol=1e-6;
options=odeset('AbsTol', 1e-6);

% graphical tracing of computation
% options=odeset('outputFCN', @odeplot);

% calling of ODE function - diff. eq. solving
[t,uc]=ode23(@in_val,[T0 Tfinal],[Uc0],options);

% current ic computing
% i.e. ic = C*duc/dt, use function diff to compute differential
current_c=C.*diff(uc)./diff(t);
```

```
% start to create graph
% get screen size to variable scrsz
scrsz = get(0,'ScreenSize');

% create graph windows at the left corner of screen, lower edge distance
% is 1/5 of screen size, 1/2 of screen size high and wide as screen
figure('Position',[1 scrsz(4)/5 scrsz(3) scrsz(4)/2])


% create two subplots
subplot(1,2,1);

% first subplot - time dependency uc (blue color)
% solid line
plot(t,uc,'-b');

% titles and labels
title('RC circuit - u_c');
xlabel('t');
ylabel('u_c');

% second subplot
subplot(1,2,2);

% second subplot - time dependency ic (red color)
% solid line
% draw one element less - current_c is shorter - side effect of diff function
% Test it, for example, with following values: diff([5 8 9 7 10])

plot(t(1:length(t)-1),current_c,'-r');
title('RC circuit - i_c');
xlabel('t');
ylabel('i_c');
```

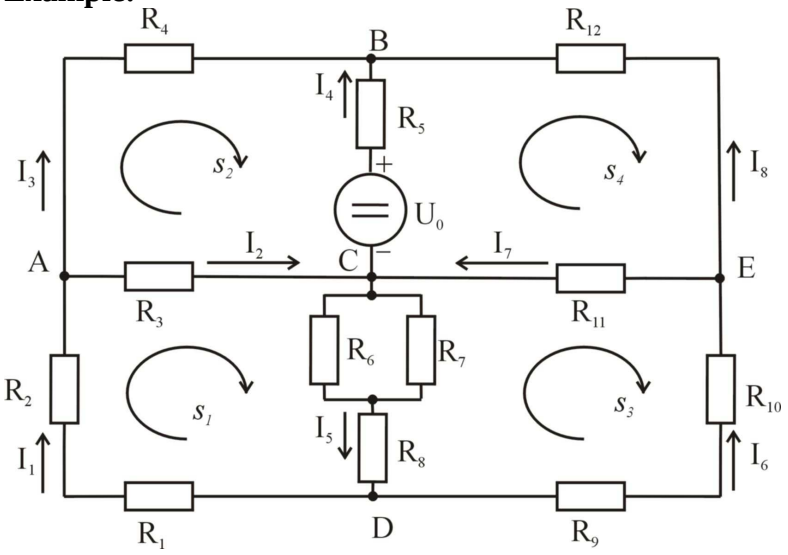
Example:
“Interactive Graphics”

```
function int_gr
handle_lines=plot(0,0);
axis([0 10 -5 5]);

handle_axis =get(handle_lines,'Parent');
handle_figury=get(handle_axis,'Parent');
set(handle_figury,'Renderer','OpenGL')
hold on;

pt = get(handle_axis,'CurrentPoint'); x=pt(1,1); y=pt(1,2);

xx=[]; yy=[];
while ishandle(handle_figury),
    pt = get(handle_axis,'CurrentPoint');
    if (x~=pt(1,1))||(y~=pt(1,2)),
        x=pt(1,1); y=pt(1,2);
        xx(length(xx)+1)=x;
        yy(length(yy)+1)=y;
        plot(xx,yy,'.-');
    end
    drawnow;
   % pause(0.2);
end
```

5. Predefined Dialog Boxes

dialog	Create and display dialog box
errordlg	Create and display error dialog box
helpdlg	Create and display help dialog box
inputdlg	Create and display input dialog box
listdlg	Create and display list selection dialog box
msgbox	Create and display message dialog box
pagesetupdlg	Display page setup dialog box
printdlg	Display print dialog box
questdlg	Display question dialog box
uigetdir	Display standard dialog box for retrieving a directory
uigetfile	Display standard dialog box for retrieving files
uiputfile	Display standard dialog box for saving files
uisave	Display standard dialog box for saving workspace variables
uisetcolor	Display standard dialog box for setting an object's
ColorSpecuisetfont	Display standard dialog box for setting an object's font characteristics
waitbar	Display waitbar warndlgDisplay warning dialog box

Example:



Kirchhoff‘s laws for this circuit ($m = 4$, $n = 4$):

I. Kirchhoff’s law: A: $I_1 - I_2 - I_3 = 0$ B: $I_3 + I_4 + I_8 = 0$

D: $-I_1 + I_5 - I_6 = 0$ E: $I_6 - I_7 - I_8 = 0$

II. Kirchhoff’s law: s_1: $R_1 I_1 + R_2 I_1 + R_3 I_2 + R_{67} I_5 + R_8 I_5 = 0$

s_2: $-R_3 I_2 + R_4 I_3 - R_5 I_4 + U_0 = 0$

s_3: $-R_9 I_6 - R_{10} I_6 - R_{11} I_7 - R_8 I_5 - R_{67} I_5 = 0$

s_4: $R_5 I_4 - R_{12} I_8 + R_{11} I_7 - U_0 = 0$, where $R_{67} = \frac{R_6 R_7}{R_6 + R_7}$

```
function currents_R=calc_curr_04(U0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12)
% calculating of current in cicrcuit using Kirchhoff's law
% (c) Lenka Sroubova 10/2004, Petr Kropik 2005

R67=(R6*R7)/(R6+R7);

currents_R = zeros(10,1);
A=[1,-1,-1,0,0,0,0,0;0,0,1,1,0,0,0,1;-1,0,0,0,1,-1,0,0;0,0,0,0,0,1,-1,-
1;(R1+R2),R3,0,0,(R67+R8),0,0,0;0,-R3,R4,-R5,0,0,0,0;0,0,0,0,-(R67+R8),-(R9+R10),-
R11,0;0,0,0,R5,0,0,R11,-R12];
b=[0;0;0;0;0;-U0;0;U0];
```

```
currents_R=A\b;

% calculation of the dependecy R7 on R6 increasing, with step 10
% to 10 * R6
% and the dependecy R7 on R6 dereasing, with step 2,
% with no change of R67
% graph's boundary values:
% R6=R67    R7=Inf
% R6=10*R6
% R7 is calculated using next formula:
R7=R6*10*R67/(R6*10-R67);

currents_R(length(b)+1) = R6;
currents_R(length(b)+2) = R67;

function genzad
% (c) Petr Kropik 3/2003, Lenka Sroubova 10/2004
% generates requested number of submissions and calculate results and graphs

% number of currents
num_of_currents = 8;
checking_R = 2;

message1 = sprintf('Number of submissions: ');
message2 = sprintf('Start value of numbering.\n(i.e. if you insert 20, numbering of
submissions will be 20, 21, 22 etc.)\nIf you insert a number less than 1, a number 1 will
be used automatically.');
message = {message1, message2};
dlg_title = 'Submissions generating';
num_of_lines = 1;
preselection = {'20', '1'};
usr_inp = char(inputdlg(message, dlg_title, num_of_lines, preselection));

% Cancel press handling
if strcmp(usr_inp,'')
    return;
end;

num_of_submissions = str2num(usr_inp(1,:));
start_of_numbering = str2num(usr_inp(2,:));
if start_of_numbering < 1
    start_of_numbering = 1;
end;

[file_name, folder_name, filterindex] = uiputfile({'*.txt','Text file (*.txt)'; '*.*',
'All files (*.*)'}, 'Save to file...');
name_size = length(file_name);
if name_size > 4
    % is extension type right?
    if strcmpi(file_name(name_size-3:name_size),'.txt')
        usr_path = sprintf('%s%s', folder_name, file_name);
    else
        usr_path = sprintf('%s%s%s', folder_name, file_name, '.txt');
    end;
elseif name_size > 0
    usr_path = sprintf('%s%s%s', folder_name, file_name, '.txt');
else
    msgbox('Empty file name, it is wrong...','Error','error');
    return;
end;

question = sprintf('Would You like to create graphs of depencies R67 to each
submission?\n\n(this operation can take some time ;) and occupy more disk space)');
button = questdlg(question, 'Create graph files', 'Yes','No','No');
if strcmp(button,'Yes')
% flag set to 1 - graphs will be created, path and file types will be set
```

```
    generate_graphs = 1;
    pict_folder = uigetdir(folder_name,'Select folder to save pictures:');
    format_list = [{'fig - MATLAB figure'}, {'jpg - JPEG'}, {'bmp - Windows bitmap'},
{'emf - Enhanced metafile'}, {'eps - EPS Level 1'}, {'m - MATLAB M-file'}, {'pbm -
Portable bitmap'}, {'pcx - Paintbrush 24-bit'}, {'pgm - Portable Graymap'}, {'png -
Portable Network Graphics'}, {'ppm - Portable Pixmap'}, {'tif - TIFF (kompr.)'}];
    [selection, was_OK] = listdlg('PromptString', 'Select file formats to save
pictures:', 'ListSize', [250 200], 'SelectionMode', 'single', 'ListString', format_list);
    switch selection
       case 1
          pict_extension = '.fig';
       case 2
          pict_extension = '.jpg';
       case 3
          pict_extension = '.bmp';
       case 4
          pict_extension = '.emf';
       case 5
           pict_extension = '.eps';
       case 6
           pict_extension = '.m';
       case 7
           pict_extension = '.pbm';
       case 8
           pict_extension = '.pcx';
       case 9
           pict_extension = '.pgm';
       case 10
           pict_extension = '.png';
       case 11
           pict_extension = '.ppm';
       case 12
           pict_extension = '.tif';
    end;
elseif strcmp(button,'Ne')
% flag set to 0 - graphs will not be created
    generate_graphs = 0;
end;

% ------------------------------------------------------------

U0 = round(rand(num_of_submissions,1)*100)+50;
R1 = round(rand(num_of_submissions,1)*90)+10;
R2 = round(rand(num_of_submissions,1)*90)+10;
R3 = round(rand(num_of_submissions,1)*90)+10;
R4 = round(rand(num_of_submissions,1)*90)+10;
R5 = round(rand(num_of_submissions,1)*90)+10;
R6 = round(rand(num_of_submissions,1)*44)+10;
R7 = round(rand(num_of_submissions,1)*45)+55;
R8 = round(rand(num_of_submissions,1)*90)+10;
R9 = round(rand(num_of_submissions,1)*90)+10;
R10 = round(rand(num_of_submissions,1)*90)+10;
R11 = round(rand(num_of_submissions,1)*90)+10;
R12 = round(rand(num_of_submissions,1)*90)+10;

wait_handler = waitbar(0,'Calculating of currents I1 to I8 in progress...');
currents_R = zeros(num_of_submissions, num_of_currents + checking_R);
for cycle = 1:num_of_submissions
    % attention - result of calc_curr_04 must be transposed
    currents_R(cycle, :) = (calc_curr_04(U0(cycle), R1(cycle), R2(cycle), R3(cycle),
R4(cycle), R5(cycle), R6(cycle), R7(cycle), R8(cycle), R9(cycle), R10(cycle), R11(cycle),
R12(cycle)))';
    waitbar(cycle / num_of_submissions, wait_handler);
end;
close(wait_handler);

final_result = [U0, R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, currents_R];
```

```
file_id = fopen(usr_path,'w');
if file_id == -1
    message2 = sprintf('File %s can not be opened...', usr_path);
    msgbox(message2,'Error','error');
    return;
end;

message2 = sprintf('Saving of submissions to file %s is in progress.', file_name);
wait_handler = waitbar(0,message2);
for cycle = 1:num_of_submissions
% first variable is id of submission

fprintf(file_id,'%d;%d;%d;%d;%d;%d;%d;%d;%d;%d;%d;%d;%d;%6.4f;%6.4f;%6.4f;%6.4f;%6.4f;
%6.4f;%6.4f;%6.4f;%d;%d\n', cycle + start_of_numbering - 1, final_result(cycle,:));
    waitbar(cycle / num_of_submissions, wait_handler);
end;
close(wait_handler);

% close file
file_status = fclose(file_id);
if file_status ~= 0
    message2 = sprintf('File %s can not be closed...', usr_path);
    msgbox(message2,'Error','error');
    return;
elseif generate_graphs == 1
    message2 = sprintf('Graphs generating and saving to folder %s', pict_folder);
    wait_handler = waitbar(0,message2);
    for cycle = 1:num_of_submissions
        graph_R67_04(R6(cycle), R7(cycle), cycle + start_of_numbering - 1, pict_folder,
pict_extension);
        waitbar(cycle / num_of_submissions, wait_handler);
    end;
    close(wait_handler);
    message2 = sprintf('\n\nFile %s was saved...\n\n\nData are separed by
semicolon.\n\n\nFirst item is sequential number followed by:\n\nInputs: U0, R1, R2, R3,
R4, R5, R6, R7, R8, R9, R10, R11, R12\n\nResults: I1, I2, I3, I4, I5, I6, I7, I8, R6,
R67\n\nGraphs are in folder %s\n\nThis greatfull software was created by: Lenka Šroubová
and Petr Kropík.\n\n\n', usr_path, pict_folder);
    msgbox(message2,'Finished','help');
else
    message2 = sprintf('\n\nFile %s was saved...\n\n\nData are separed by
semicolon.\n\n\nFirst item is sequential number followed by:\n\nInputs: U0, R1, R2, R3,
R4, R5, R6, R7, R8, R9, R10, R11, R12\n\nResults: I1, I2, I3, I4, I5, I6, I7, I8, R6,
R67\n\nThis greatfull software was created by: Lenka Šroubová and Petr Kropík.\n\n\n',
usr_path);
    msgbox(message2,'Finished','help');
end;

function graph_R67_04(R6, R7, number_of_submission, my_path, extension)
% generating of requested number of graphs
% (c) Petr Kropik 3/2003, Lenka Sroubova 10/2004

path_file = sprintf('%s\\Graf_%04d%s', my_path, number_of_submission, extension);

% open figure window and save it's window id to variable handler
scrsz = get(0,'ScreenSize');
handler = figure('Position',[20 scrsz(4)-250 200 200]);

R67=(R6*R7)/(R6+R7);

m=1;
for n=(R67+1):2:R6
    r6(m)=n;
    r7(m)=r6(m)*R67/(r6(m)-R67);
    m=m+1;
```

```
end;
for n=R6:10:(10*R6)
    r6(m)=n;
    r7(m)=r6(m)*R67/(r6(m)-R67);
    m=m+1;
end;
plot(r6,r7);
title('Parallel-connected resistors');
xlabel('R_6');
ylabel('R_7');

% saving of figure to a picture file
saveas(handler, path_file);

% close figure window
close(handler);
```