Experimental studies "Signals and Systems 1"		
Experiment No. 3 :	Convolution of Signals	
Fachgebiet:	Nachrichtentechnische Systeme	
Name:		Matr.-Nr.:
Supervisor:		Date:

The exercises in chapter 2 have to be solved before the practical exercise**!**

1. Introduction to the MATLAB system

In this practical exercise the convolution of some elementary functions should be carried out by means of MATLAB functions. The following module "AOSPraktA" cares for the MATLAB definition of these functions.

1.1 Example of MATLAB code

As with all continuos-time functions also in this case the computer can only process the discrete-time representation of the signals over a limited period. In this exercise this period is defined by means of the parameter Tmax. The sampling interval is denoted by dt.

In order to get familiar with the MATLAB notation the code of this module is presented in the following (including additional line numbers for easier reference of statements).

```
% AOSPraktA
% Defines 6 elementary funktions
% (rect, triangle, step, exponential, noise, Dirac)

% Begin of parameter block
dt=0.01;      % time interval
Tmax=3;       % functions will be evaluated between -Tmax to +Tmax

N=300;        % halve number of intervals dt

T=0.4; Tv=0.3; % Parameters for rect- and triangular-function
TG=0.5;        % Parameter for Gaussian impulse
Tvd = 100;     % Parameter for Dirac impulse
               (100 corresponds to a shift of 1.0)
NA=0.5;        % Noise amplitude
% End of parameter block

t=-Tmax:dt:Tmax;
t2= -2*Tmax:dt:2*Tmax;

%   s1(t) = rect((t-Tv)/T)
for i=-N:N
  t1=dt*i;
  s1(i+1+N)=1;
  if t1<Tv-T/2
   s1(i+1+N)=0;
  end
  if t1>Tv+T/2
   s1(i+1+N)=0;
  end
end
```

```
26    % s2(t)= Lambda((t-Tv)/T)
27    for i=-N:N
28     t1=dt*i;
29     s2(i+1+N)=1-abs((t1-Tv)/T);
30     if t1<Tv-T
31       s2(i+1+N)=0;
32     end
33     if t1>Tv+T
34       s2(i+1+N)=0;
35     end
36    end

37    % s3(t) = epsilon(t-Tv)*exp(-(t-Tv)/T)
38    for i=-N:N
39     t1=dt*i;
40     s3(i+1+N)=exp(-(t1-Tv)/T);
41     if t1<Tv-T
42       s3(i+1+N)=0;
43     end
44    end

45    % s4(t) = exp(-(t-Tv)*(t-Tv)/(TG*TG))
46    s4 = exp(-(t-Tv).*(t-Tv)/TG^2);

47    % s5(t): Gaussian white noise
48    s5 = NA*randn(size(t));

49    % s6(t) = Dirac-Impulse
50    s6=t*0.0;
51    s6(N+1+Tvd)=1/dt;
```

Table 1: Code of MATLAB module AOSPraktA

Most of this notation is well comparable to other programming languages. Some essential differences are explained as follows.

- As in lines 1 to 7 and other lines of the code comments may be included by means of the character "%" (starting anywhere in the line).
- A semicolon ";" at the end of the line directs MATLAB not to display the result of the line on the screen and may also be used to put several short instructions in one line. Note: Especially when operating with arrays a missing ";" leads to filling up the screen quickly (and in the case of large arrays may block any manual input for minutes).
- MATLAB code is case sensitive ("dt" is different from "Dt").
- In line 13 the time variable t is defined which runs from -3 to +3 with an interval dt of 0.01. Thus a row vector with 601 element's is defined. The same happens in line 14 where the vector t2 with 1201 elements is defined.

- In line 16 and 25 a "for" loop is defined. No semicolon appears in such control statements. All other statements to be carried out in such a loop - or other control structures - must appear in the lines between (and usually need the semicolon).

- Line 18 defines a new vector s1. Please note that only indices ≥ 1 are accepted.

- Lines 19 to 21 (and lines 22 to 24) define an "if" statement. The notation follows the structure of the "for" loop.

- If possible "for" or "while" loops should be avoided as they consume considerable more computing time than corresponding array operations. Line 46 gives an example of a "for" loop. These statements direct MATLAB to compute the elements of a new row vector s4 on the basis of the row vector t (representing the values of all time intervals).

- The multiplication of a constant like TG with a vector t e.g. is denoted by "TG*t". But the element-by-element product of two vectors or arrays t1 and t2 is denoted by "t1.*t2" resulting in a new vector which is different from the product t1*t2 (representing the scalar result of the inner product or the matrix result of the outer product). Also other element-by-element operations like ".\" and ".^N" follow this syntax.

- If matrices A and B e.g. have to be multiplied (or divided) the corresponding statements are simply expressed in the form of "A*B" and "A/B".

- Squaring is denoted by "^2". In general x^n is denoted by "x^n" and $\sqrt{x}$ is denoted by "sqrt(x)" - as usual.

- In line 48 the "randn()" function is used in order to define a vector of 601 random elements which are gaussian distributed and uncorrelated.

1.2 Other useful functions for this practical exercise

- The sequence

```
plot (t,s6)
zoom xon
title('signal s6(t)')
xlabel('Time t')
```

  plots the signal s6 over the vector t. The "title", "xlabel" (and "ylabel") statements provide for commenting the plot. The "zoom" statement switches on or off zooming of the x-axis (by meams of "xon" or "xoff"), zooming of the y-axis ("yon" or "yoff") or total zooming ("on" or "off").

- The functions "zeros(n1,n2)" and "ones(n1,n2)" provide a vector containing the value of 1 or 0 for all elements with indices ranging between n1 and n2. The zeros function is therfore quite useful for initialising of vectors or filling up vector elements with zeros.

- The functions "min(a,b)" and "max(a,b)" determine the minimum and the maximum value of the parameters a and b.

- The functions "real(f)" and "imag(f)" determine the real and imaginary part of the complex value f. The functions "abs(f)" and "angle(f)" determine magnitude and phase angle of f. The function "unwrap(angle(f))" may be used to prepare a phase plot without jumps.

- The function "fft(s)" determines the Discrete Fourier transform of the vector s, whereas "ifft(S)" determines the inverse discrete Fourier transform of S.

1.3 Some additional vector/matrix operations

- By means of A=[1 1]; B=[2 2 2]; C = [A B 3 4 5] ; the row vector C with the elements [1 1 2 2 2 3 4 5] results. Changing of row to column vectors is carried out by means of the transpose operator " ' " as in y = x'; Elements of a row vector C are addressed by means of the corresponding index value as in C(4), which displays the 4^{th} element of C.

- The statement D = [1; 1; 2; 2; 2; 3; 4; 5] represent a column vector (8-by-1 matrix) which is the transpose of C. A 4-by-2 matrix E is defined by means of E = [1 2; 3 4; 5 6; 7 8] .

- The statement D = C(3:6); results in D = [2 2 2 3]. In this way parts of vectors can be addressed and isolated. For isolating the second row of a matrix E into a new vector F the following statements F = E(2,:) would work and leads to the same result as F = [3 4]. If only halve of the second column e.g. should be isolated the statement G = (1:length(E)/2,2) would be appropriate and would give the result G = [2; 4; 6; 8] .

2. Questions and problems for the preparation of the exercise

2.1

The signals $s(t) = \delta(t-T) - 2\delta(t-2T) + 3\delta(t-3T)$ and $f(t) = \delta(t-2T) + 2\delta(t-3T)$ are given. Determine *s(t)*f(t)* and sketch it.

2.2

If instead of a signal *s(t)* its derivative $s'(t)$ is given, the convolution of *s'(t)* and *f(t)* results in $s'(t) * f(t) = y'(t)$ with $y(t) = s(t) * f(t)$.

Please also note that *y(t)* can be obtained from $y'(t)$ by integrating it!

Use these properties in order to determine the convolution $y(t) = s(t) * f(t)$ for

$$s(t) = r(\frac{t}{T}) \cdot rect(\frac{t}{T} - 0.5) \;+\; (1 - 2r(\frac{t-T}{T})) \cdot rect(\frac{t-T}{0..5T}) \text{ with } f(t) = \delta(t-2T) + 2\delta(t-3T).$$

Procedure for the solution:

At first make a sketch of *s(t)* and its derivative. Then determine $y'(t)$ and sketch it. Finally determine *y(t)* and sketch it.

2.3

The signals $s(t) = (S_0 + S_1 \cdot (t/T)) \cdot rect(t/T - 0.5)$ and $f(t) = F_0 \cdot rect(t/T - 0.5)$ are given. Determine the convolution product $y(t) = s(t) * f(t)$. Please note that S_0, S_1 and F_0 are constants.

2.4

Describe what happens if instead of *s(t)* the shifted signal $s_V(t) = s(t - NT)$ has to be convoluted with f(t)?

3. The exercise

3.1

Write a MATLAB program which carries out the convolution of two signal *s(t)* and *h(t),* represented by the MATLAB vectors s and h, by means of an approximation of the convolution integral

$$g(t) = \int_{-\infty}^{+\infty} s(\tau)h(t-\tau)d\tau \quad .$$

Verify the program by means of the convolution of *s(t) = rect(t/T)* with *h(t) = s(t)* and *T = 1.0* in the form of *g1(t) = s(t) * h(t)* .

Note: Please observe that in MATLAB no indices ≤ 0 are allowed!

3.2

Determine the following convolution operations

*g1(t) = rect(t/T) * rect (t/T) * rect(t/T) * rect(t/T)*

g2(t) = $\varepsilon(t) \cdot e^{-t/T} * \varepsilon(t) \cdot e^{-t/T}$

Discuss the results.

3.3

Write a MATLAB program which carries out the convolution of *s(t)* and *h(t)* by means of the MATLAB functions fft and ifft. Verify the program once again by means of the example in 3.1.

Note:
MATLAB determines the Fourier transform *S = fft(s)* by means of the Discrete Fourier transform, which transforms an Nth element real input vector s into a Nth element output vector S whith the property of $S(k) = S^*(N-k+1) \;\; \forall \, 1 \le k \le N/2$. If s(t) is represented by N elements of s the transform S only exhibits N/2 different elements in magnitude e.g..

To get correct convolution results the input vector s with M elements therefore has to be extended by additional M zeros.

3.4

Determine the following convolution operations

$$g_3(t) = rect(t/T) * \Lambda(t/2T)$$
$$g_4(t) = rect(t/T) * rect(t/4T)$$

3.5

Determine the following convolution operations (as an example on signal averaging)

$$g_5(t) = (\ \exp(-(t/T)^2) + s_5(t)\ ) * rect(t/aT)$$
$$g_6(t) = (\ \sin(2\pi\ t/T) + s_5(t)\ ) * rect(t/aT)$$

with varying parameter value a.

4. References

[1] Matlab, The MathWorks Inc., http//www.mathworks.com
[2] The Student Edition of Matlab, The MathWorks Inc.
[3] Transmission and Modulation of Signals- Experiment1: Introduction to Matlab, http://fb9nt.uni-duisburg.de (Section 'Lehre')