```
%%This class is a collection of functions related to signal detection via
%%Karhunen Loeve Transformation
%%written by Tim Jaschek as a part of his bachelor thesis%%

%%Used to generate FIGURE 5 %%
%%...to generate it, type the following in your MATLAB command:
%%Signal;
%%Signal.compare2();
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

classdef Signal
   properties (Constant)
   end
   methods (Static)
        function tone = SinTone(toneFreq,sampleFreq)
            %build sine tone
            t = (1:sampleFreq) / sampleFreq;             % build time steps of length 1 second
            tone = sin(2 * pi * toneFreq * t);           % sinusoidal modulation
        end
        function playTone(tone,sampleFreq)
            %play tone
            sound(tone, sampleFreq);    % sound function from Matlab
            pause(1.5);                 % wait
        end
        function spect = Spectrum(coeff)
            spect = abs(coeff);
            %know spectrum is two sided. Make it one sided:
            spect = spect(1:length(coeff)/2+1);
            spect(2:end-1) = 2*spect(2:end-1);
        end
        function K = AutoCo(data)
            [M,N] = size(data);
            K = zeros(N,N);
            AK = zeros(N);
            for j=1:N
                for l=1:M
                    AK(j) =  AK(j) + data(l,1)*data(l,j);
                end
                AK(j)=AK(j)/M;
            end
            for j=1:N
                for k=1:N
                    K(j,k) = AK(abs(j-k)+1);
                end
            end
        end
        function K = AutoCo2(data)
           [M,N] = size(data);
           K = zeros(N,N);
           for j = 1:N
                for k = 1:N
                    %use symmetry to save operations
                    if k<j
                        K(j,k) = K(k,j);
                    else
                        for l=1:M
                            K(j,k)=  K(j,k) + data(l,k)*data(l,j);
                        end
                        K(j,k)=K(j,k)/M;
                    end
                end
           end
        end
        function coeff = KLT(K,E)
            Kernels;
            [lambda,Phi] = Kernels.trapez_Sceme(K);
            Phi(:,1)=sqrt(lambda(1))*Phi(:,1);
            %for i = 2:5
            %    Phi(:,1)=Phi(:,1)+sqrt(lambda(i))*Phi(:,i);
            %end
            N = length(E);
            A = zeros(1,N);
            for j=1:N
                A(j) = Phi(j,1)+E(j);
            end
            coeff = fft(A);
        end
        function compare()
            %build measure values
            N = 1400;
            M = 40000;
            figure
            %different factor for the noise amplitude
            for i = 1:4
```

```
            data = zeros(M,N);
            if i ==1
                z=2;
            elseif i ==2
                z=4;
            elseif i == 3
                z=10;
            else
                z=100;
            end
            for j = 1:M
                %generate M times tone + noise
                tone = Signal.SinTone(300,N);
                noise = z*randn(1,N);
                data(j,:) = tone + noise;
            end
            %first line is B
            B = data(1,:);
            %take the time
            tic;
            %build covariance matrix
            K = Signal.AutoCo(data);
            %KARHUNEN-LOEVE TRANSFORMATION
            %KLT returns first Eigenfunction in Fourier base
            coeff = Signal.KLT(K,zeros(1,N));
            spectrum_KLT = Signal.Spectrum(coeff);
            toc
            tic;
            %FAST FOURIER TRANSFORM
            spectrum_FFT = Signal.Spectrum(fft(B));
            toc
            subplot(4,2,1+2*(i-1));
            plot(spectrum_FFT);
            if i == 1
                title('SNR=0.5 - FFT');
            elseif i == 2
                title('SNR=0.25  - FFT');
            elseif i == 3
                title('SNR=0.1  - FFT');
            elseif i == 4
                title('SNR=0.01  - FFT');
            end
            xlabel('Frequenz in Hz');
            ylabel('Magnitude');
```

```
                subplot(4,2,2+2*(i-1));
                plot(spectrum_KLT);
                if i == 1
                    title('SNR=0.5 - KLT');
                elseif i == 2
                    title('SNR=0.25  - KLT');
                elseif i == 3
                    title('SNR=0.1  - KLT');
                elseif i == 4
                    title('SNR=0.01  - KLT');
                end
                xlabel('Frequenz in Hz');
                ylabel('Magnitude');
            end
        end
        function compare2()
            %build measure values
            N = 1400;
            M = 300;
            figure
            %different factor for the noise amplitude
            for i = 1:4
                data = zeros(M,N);
                if i ==1
                    z=2;
                elseif i ==2
                    z=4;
                elseif i == 3
                    z=10;
                else
                    z=50;
                end
                for j = 1:M
                    %generate M times tone + noise
                    tone = Signal.SinTone(300,N);
                    noise = z*randn(1,N);
                    data(j,:) = tone + noise;
                end
                %compute Expectation
                E = zeros(1,N);
                for j = 1:N
                    E(j) = sum(data(:,j))/M;
                end
                for j=1:M
                    data(j,:)=data(j,:)-E;
                end
                %first line is B
                B = data(1,:);
                %take the time
                tic;
                %build covariance matrix
                K = Signal.AutoCo2(data);
                %KARHUNEN-LOEVE TRANSFORMATION
                %KLT returns first Eigenfunction in Fourier base
                coeff = Signal.KLT(K,E);
                spectrum_KLT = Signal.Spectrum(coeff);
                toc
                tic;
                %FAST FOURIER TRANSFORM
                spectrum_FFT = Signal.Spectrum(fft(B+E));
                toc
                subplot(4,2,1+2*(i-1));
                plot(spectrum_FFT);
                if i == 1
                    title('SNR=0.5 - FFT');
                elseif i == 2
                    title('SNR=0.25  - FFT');
                elseif i == 3
                    title('SNR=0.1  - FFT');
                elseif i == 4
                    title('SNR=0.02  - FFT');
                end
                xlabel('Frequenz in Hz');
                ylabel('Magnitude');
                subplot(4,2,2+2*(i-1));
                plot(spectrum_KLT);
                if i == 1
                    title('SNR=0.5 - KLT');
                elseif i == 2
                    title('SNR=0.25  - KLT');
                elseif i == 3
                    title('SNR=0.1  - KLT');
                elseif i == 4
                    title('SNR=0.02  - KLT');
                end
                xlabel('Frequenz in Hz');
                ylabel('Magnitude');
            end
        end
    end
end

```