```
  1 %%This class is a collection of functions related to signal detection via
  2 %%Karhunen Loeve Transformation
  3 %%written by Tim Jaschek as a part of his bachelor thesis%%
  4
  5 %%Used to generate FIGURE 5 %%
  6 %%...to generate it, type the following in your MATLAB command:
  7 %%Signal;
  8 %%Signal.compare2();
  9 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 10
 11 classdef Signal
 12    properties (Constant)
 13    end
 14    methods (Static)
 15         function tone = SinTone(toneFreq,sampleFreq)
 16             %build sine tone
 17             t = (1:sampleFreq) / sampleFreq;             % build time steps of length 1 second
 18             tone = sin(2 * pi * toneFreq * t);           % sinusoidal modulation
 19         end
 20         function playTone(tone,sampleFreq)
 21             %play tone
 22             sound(tone, sampleFreq);    % sound function from Matlab
 23             pause(1.5);                 % wait
 24         end
 25         function spect = Spectrum(coeff)
 26             spect = abs(coeff);
 27             %know spectrum is two sided. Make it one sided:
 28             spect = spect(1:length(coeff)/2+1);
 29             spect(2:end-1) = 2*spect(2:end-1);
 30         end
 31         function K = AutoCo(data)
 32             [M,N] = size(data);
 33             K = zeros(N,N);
 34             AK = zeros(N);
 35             for j=1:N
 36                 for l=1:M
 37                     AK(j) =  AK(j) + data(l,1)*data(l,j);
 38                 end
 39                 AK(j)=AK(j)/M;
 40             end
```

```
41              for j=1:N
42                  for k=1:N
43                      K(j,k) = AK(abs(j-k)+1);
44                  end
45              end
46          end
47          function K = AutoCo2(data)
48             [M,N] = size(data);
49             K = zeros(N,N);
50             for j = 1:N
51                  for k = 1:N
52                      %use symmetry to save operations
53                      if k<j
54                          K(j,k) = K(k,j);
55                      else
56                          for l=1:M
57                              K(j,k)=  K(j,k) + data(l,k)*data(l,j);
58                          end
59                          K(j,k)=K(j,k)/M;
60                      end
61                  end
62             end
63          end
64          function coeff = KLT(K,E)
65              Kernels;
66              [lambda,Phi] = Kernels.trapez_Sceme(K);
67              Phi(:,1)=sqrt(lambda(1))*Phi(:,1);
68              %for i = 2:5
69              %    Phi(:,1)=Phi(:,1)+sqrt(lambda(i))*Phi(:,i);
70              %end
71              N = length(E);
72              A = zeros(1,N);
73              for j=1:N
74                  A(j) = Phi(j,1)+E(j);
75              end
76              coeff = fft(A);
77          end
78          function compare()
79              %build measure values
80              N = 1400;
81              M = 40000;
82              figure
83              %different factor for the noise amplitude
84              for i = 1:4
```

```
85                  data = zeros(M,N);
86                  if i ==1
87                      z=2;
88                  elseif i ==2
89                      z=4;
90                  elseif i == 3
91                      z=10;
92                  else
93                      z=100;
94                  end
95                  for j = 1:M
96                      %generate M times tone + noise
97                      tone = Signal.SinTone(300,N);
98                      noise = z*randn(1,N);
99                      data(j,:) = tone + noise;
100                 end
101                 %first line is B
102                 B = data(1,:);
103                 %take the time
104                 tic;
105                 %build covariance matrix
106                 K = Signal.AutoCo(data);
107                 %KARHUNEN-LOEVE TRANSFORMATION
108                 %KLT returns first Eigenfunction in Fourier base
109                 coeff = Signal.KLT(K,zeros(1,N));
110                 spectrum_KLT = Signal.Spectrum(coeff);
111                 toc
112                 tic;
113                 %FAST FOURIER TRANSFORM
114                 spectrum_FFT = Signal.Spectrum(fft(B));
115                 toc
116                 subplot(4,2,1+2*(i-1));
117                 plot(spectrum_FFT);
118                 if i == 1
119                     title('SNR=0.5 - FFT');
120                 elseif i == 2
121                     title('SNR=0.25  - FFT');
122                 elseif i == 3
123                     title('SNR=0.1  - FFT');
124                 elseif i == 4
125                     title('SNR=0.01  - FFT');
126                 end
127                 xlabel('Frequenz in Hz');
128                 ylabel('Magnitude');
```

```
129                 subplot(4,2,2+2*(i-1));
130                 plot(spectrum_KLT);
131                 if i == 1
132                     title('SNR=0.5 - KLT');
133                 elseif i == 2
134                     title('SNR=0.25  - KLT');
135                 elseif i == 3
136                     title('SNR=0.1  - KLT');
137                 elseif i == 4
138                     title('SNR=0.01  - KLT');
139                 end
140                 xlabel('Frequenz in Hz');
141                 ylabel('Magnitude');
142             end
143         end
144         function compare2()
145             %build measure values
146             N = 1400;
147             M = 300;
148             figure
149             %different factor for the noise amplitude
150             for i = 1:4
151                 data = zeros(M,N);
152                 if i ==1
153                     z=2;
154                 elseif i ==2
155                     z=4;
156                 elseif i == 3
157                     z=10;
158                 else
159                     z=50;
160                 end
161                 for j = 1:M
162                     %generate M times tone + noise
163                     tone = Signal.SinTone(300,N);
164                     noise = z*randn(1,N);
165                     data(j,:) = tone + noise;
166                 end
167                 %compute Expectation
168                 E = zeros(1,N);
169                 for j = 1:N
170                     E(j) = sum(data(:,j))/M;
171                 end
172                 for j=1:M
```

```
173                     data(j,:)=data(j,:)-E;
174                 end
175                 %first line is B
176                 B = data(1,:);
177                 %take the time
178                 tic;
179                 %build covariance matrix
180                 K = Signal.AutoCo2(data);
181                 %KARHUNEN-LOEVE TRANSFORMATION
182                 %KLT returns first Eigenfunction in Fourier base
183                 coeff = Signal.KLT(K,E);
184                 spectrum_KLT = Signal.Spectrum(coeff);
185                 toc
186                 tic;
187                 %FAST FOURIER TRANSFORM
188                 spectrum_FFT = Signal.Spectrum(fft(B+E));
189                 toc
190                 subplot(4,2,1+2*(i-1));
191                 plot(spectrum_FFT);
192                 if i == 1
193                     title('SNR=0.5 - FFT');
194                 elseif i == 2
195                     title('SNR=0.25  - FFT');
196                 elseif i == 3
197                     title('SNR=0.1  - FFT');
198                 elseif i == 4
199                     title('SNR=0.02  - FFT');
200                 end
201                 xlabel('Frequenz in Hz');
202                 ylabel('Magnitude');
203                 subplot(4,2,2+2*(i-1));
204                 plot(spectrum_KLT);
205                 if i == 1
206                     title('SNR=0.5 - KLT');
207                 elseif i == 2
208                     title('SNR=0.25  - KLT');
209                 elseif i == 3
210                     title('SNR=0.1  - KLT');
211                 elseif i == 4
212                     title('SNR=0.02  - KLT');
213                 end
214                 xlabel('Frequenz in Hz');
215                 ylabel('Magnitude');
216             end
```

```
217         end
218     end
219 end
220
```