

```

1 %%This class is a collection of functions related to the✓
Brownian%%
2 %%motion in the context of Karhunen-Loeve Expansion%
3 %%written by Tim Jaschek as a part of%%
4 %%-a presentation about the Levy construction of Brownian Motion%
5 %%-a presentation about the idea of Karhunen-Loeve Expansion%
6 %%-a part of his bachelor thesis%%
7
8 %%This Programm is used to generate FIGURE 3 in the thesis%
9 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
10
11 classdef BrMo
12     properties (Constant)
13         steps = 1000 ;
14         time = linspace(1/BrMo.steps,1,BrMo.steps) ;
15     end
16     methods (Static)
17         function [Z,lambda,psi] = compute_KLT_components(n)
18             %this function computes the analytically solved✓
eigenvalues and
19             %eigenfunctions of a Brownian motion
20             steps = BrMo.steps ;
21             Z = randn(1,n) ;
22             lambda = zeros(1,n) ;
23             psi = zeros(n,steps) ;
24             for j = 1:n
25                 lambda(j) = 1 / ((j-0.5) * pi);
26                 for k = 1:steps
27                     psi(j,k) = sqrt(2)*sin((j-0.5)*pi*k/steps) ;
28                 end
29             end
30         end
31         function BM = BrownianMotion(n,steps)
32             %using the previous function, this function computes✓
the n-th
33             %partial sum of the Karhunen-Loeve Expansion of✓
Brownian motion
34             BM = zeros(1,steps) ;
35             [Z,lambda,psi] = BrMo.compute_KLT_components(n);
36             for j = 1:n
37                 for k = 1:steps
38                     BM(k) = BM(k) + Z(j)*lambda(j)*psi(j,k);
39                 end
40             end

```

```

41         %OPPORTUNITY to plot:
42         %figure
43         %plot(BrMo.time,BM)
44         %xlabel('time')
45         %str=sprintf('Brownian Motion via Karhunen-Loeve✓
Approx. for n = %d',n);
46         %title(str)
47     end
48     function plotDevelop()
49         %generates the first plot of FIGURE 3
50         BM = zeros(1,BrMo.steps) ;
51         [Z,lambda,psi] = BrMo.compute_KLT_components(80);
52         figure
53         subplot(1,2,1)
54         title('Brownian Motion via Karhunen-Loeve Approx. for✓
different n')
55         hold on;
56         for j = 1:80
57             for k = 1:BrMo.steps
58                 BM(k) = BM(k) + Z(j)*lambda(j)*psi(j,k);
59             end
60             if j==4 || j==16 || j==32 || j==64
61                 plot(BrMo.time,BM)
62             end
63         end
64         legend('n = 4','n = 16', 'n = 32','n = 64')
65         hold off;
66     end
67     function plotMovie()
68         %an exciting movie that shows the behaviour of the✓
partial sums
69         %of the Karhunen-Loeve Expansion is generated here
70         BM = zeros(1,BrMo.steps) ;
71         [Z,lambda,psi] = BrMo.compute_KLT_components(500);
72         for j = 1:500
73             for k = 1:BrMo.steps
74                 BM(k) = BM(k) + Z(j)*lambda(j)*psi(j,k);
75             end
76             plot(BrMo.time,BM)
77             title('Animation of KLT Series');
78             pause(0.01)
79         end
80     end
81     function LevyBase()

```

```

82         %plots the Schauder base
83         Bn = [0,1] ;
84         List = [0,1] ;
85         figure
86         title('Schauder Base - Base from the Construction by
Lèvy');
87         xlabel('time');
88         hold on;
89         plot(List,Bn);
90         for n = 1:4
91             if n==1
92                 Bn = [Bn, 0];
93                 List = linspace(0,1,3);
94             else
95                 Bn = [Bn, 1 , 0];
96                 List = linspace(0,1,length(Bn));
97             end
98             plot(List,Bn);
99         end
100        hold off;
101    end
102    function KLTBase()
103        %plots the Karhunen-Loeve Eigenfunction base
104        [Z,lambda,psi] = BrMo.compute_KLT_components(6);
105        figure
106        title('Fourier Base - Base from Karhunen-Loève
Theorem');
107        xlabel('time');
108        hold on;
109        for j = 1:6
110            plot(BrMo.time,psi(j,:))
111        end
112        hold off;
113    end
114    function levy()
115        %generates the second plot of FIGURE 3
116        D_0 = linspace(0,1,2^0 + 1) ; %intervalls of dyadic
points
117        B_0 = [ 0 , randn]; %approximation of Brownian Motin in
step 0
118        subplot(1,2,2);
119        title('Brownian Motion via Levy Construction for
different n')
120        hold on;

```

```

121         for n=1:6
122             D_next = linspace(0,1,2^n+1);
123             B_next = zeros(1,length(D_next));
124             j=1;
125             for i=1:2^n+1
126                 if mod(i,2)== 0;
127                     B_next(i)= sqrt(1/(2^(n+1)))*randn;
128                 else
129                     B_next(i)= B_0(j);
130                     j=j+1;
131                 end
132             end
133             for i=1:2^n+1
134                 if mod(i,2)== 0;
135                     B_next(i)=B_next(i)+(B_0(i/2)+B_0(i/2+1))/2; ✓
136                 end
137             end
138             D_0 = D_next;
139             B_0 = B_next;
140             if n==2 || n==4 || n==5 || n==6
141                 plot(D_0,B_0)
142             end
143         end
144         legend('n = 4','n = 16','n = 32','n = 64')
145         hold off;
146     end
147     function CoV()
148         %plots the covariancefunction of Brownian motion
149         [X,Y] = meshgrid(0:.05:1);
150         Z = min(X,Y);
151         surf(X,Y,Z)
152         title('Kovarianzfunktion der Brownschen Bewegung')
153         xlabel('time')
154     end
155     function KLTlambda()
156         %plots the Eigenvalues
157         [Z,lambda,psi] = BrMo.compute_KLT_components(10);
158         plot(linspace(1,10,10),lambda,'*')
159         title('KLT Coefficients')
160         xlabel('index')
161         ylabel('lambda_i')
162     end
163     function CoVBB()

```

```

164         %plots the covariancefunction of Brownian bridge
165         [X,Y] = meshgrid(0:.05:1);
166         Z = min(X,Y) - X.*Y
167         surf(X,Y,Z)
168         title('Kovarianzfunktion der Brownschen Brücke')
169     end
170     function Mercer()
171         %plots different approximations of the
covariancefunction of
172         %Brownian motion using Mercers theorem
173         [Z,lambda,psi] = BrMo.compute_KLT_components(10);
174         figure
175         A = zeros(20,20);
176         title('Mercers Theorem - covariance function of
Brownian Motion');
177         hold on;
178         for n = 1:4
179             for i = 1 : 20
180                 for j = 1:20
181                     A(i,j) = A(i,j) + lambda(n)*psi(n,50*i)
*psi(n,50*j);
182                 end
183             end
184         end
185         [X,Y] = meshgrid(0.05:0.05:1);
186         surf(X,Y,A)
187     end
188     function value = mother(t)
189         %%Haar mother function%%
190         if ((t<0.5) && (t >= 0))
191             value = 1;
192         elseif ((t>=0.5) && (t<1))
193             value = -1;
194         else
195             value = 0;
196         end
197     end
198     function psi = Haar(N)
199         steps = linspace(1/N,1,N);
200         %%Use the Haar Mother function%%
201         psi(:,1) = ones(1,N);
202         psi(N,1) = 0;
203         n=0;
204         k=0;

```

```
205         count =2;
206         while count <N+1
207             max = 2^n;
208             if k == max
209                 n=n+1;
210                 k=0;
211                 continue
212             end
213             for t=1:N
214                 psi(t,count) = 2^(n/2)*BrMo.mother(2^n*steps✓
(t)-k);
215             end
216             count = count+1;
217             k=k+1;
218         end
219     end
220 end
221 end
```