```matlab
%%This class is a collection of functions related to the Brownian%%
%%motion in the context of Karhunen-Loeve Expansion%%
%%written by Tim Jaschek as a part of%%
%%-a presentation about the Levy construction of Brownian Motion%%
%%-a presentation about the idea of Karhunen-Loeve Expansion%%
%%-a part of his bachelor thesis%%

%%This Programm is used to generate FIGURE 3 in the thesis%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

classdef BrMo
   properties (Constant)
      steps = 1000 ;
      time = linspace(1/BrMo.steps,1,BrMo.steps) ;
   end
   methods (Static)
       function [Z,lambda,psi] = compute_KLT_components(n)
           %this function computes the analytically solved eigenvalues and
           %eigenfunctions of a Brownian motion
           steps = BrMo.steps ;
           Z = randn(1,n) ;
           lambda = zeros(1,n) ;
           psi = zeros(n,steps) ;
           for j = 1:n
                lambda(j) = 1 / ((j-0.5) * pi);
                for k = 1:steps
                    psi(j,k) = sqrt(2)*sin((j-0.5)*pi*k/steps) ;
                end
           end
       end
       function BM = BrownianMotion(n,steps)
           %using the previous function, this function computes the n-th
           %partial sum of the Karhunen-Loeve Expansion of Brownian motion
           BM = zeros(1,steps) ;
           [Z,lambda,psi] = BrMo.compute_KLT_components(n);
           for j = 1:n
                for k = 1:steps
                    BM(k) = BM(k) + Z(j)*lambda(j)*psi(j,k);
                end
           end
```

```
            %OPPORTUNITY to plot:
            %figure
            %plot(BrMo.time,BM)
            %xlabel('time')
            %str=sprintf('Brownian Motion via Karhunen-Loeve Approx. for n = %d',n);
            %title(str)
        end
        function plotDevelop()
            %generates the first plot of FIGURE 3
            BM = zeros(1,BrMo.steps) ;
            [Z,lambda,psi] = BrMo.compute_KLT_components(80);
            figure
            subplot(1,2,1)
            title('Brownian Motion via Karhunen-Loeve Approx. for different n')
            hold on;
            for j = 1:80
                for k = 1:BrMo.steps
                    BM(k) = BM(k) + Z(j)*lambda(j)*psi(j,k);
                end
                if j==4 || j==16 || j==32 || j==64
                    plot(BrMo.time,BM)
                end
            end
            legend('n = 4','n = 16', 'n = 32','n = 64')
            hold off;
        end
        function plotMovie()
            %an exciting movie that shows the behaviour of the partial sums
            %of the Karhunen-Loeve Expansion is generated here
            BM = zeros(1,BrMo.steps) ;
            [Z,lambda,psi] = BrMo.compute_KLT_components(500);
            for j = 1:500
                for k = 1:BrMo.steps
                    BM(k) = BM(k) + Z(j)*lambda(j)*psi(j,k);
                end
                plot(BrMo.time,BM)
                title('Animation of KLT Series');
                pause(0.01)
            end
        end
        function LevyBase()
```

```
            %plots the Schauder base
            Bn = [0,1] ;
            List = [0,1] ;
            figure
            title('Schauder Base - Base from the Construction by Lèvy');
            xlabel('time');
            hold on;
            plot(List,Bn);
            for n = 1:4
                 if n==1
                     Bn = [Bn, 0];
                     List = linspace(0,1,3);
                 else
                     Bn = [Bn, 1 , 0];
                     List = linspace(0,1,length(Bn));
                 end
                 plot(List,Bn);
            end
            hold off;
        end
        function KLTBase()
            %plots the Karhunen-Loeve Eigenfunction base
            [Z,lambda,psi] = BrMo.compute_KLT_components(6);
            figure
            title('Fourier Base - Base from Karhunen-Loève Theorem');
            xlabel('time');
            hold on;
            for j = 1:6
                plot(BrMo.time,psi(j,:))
            end
            hold off;
        end
        function levy()
            %generates the second plot of FIGURE 3
            D_0 = linspace(0,1,2^0 + 1) ; %intervalls of dyadic points
            B_0 = [ 0 , randn]; %approximation of Brownian Motin in step 0
            subplot(1,2,2);
            title('Brownian Motion via Levy Construction for different n')
            hold on;
```

```
            for n=1:6
                D_next = linspace(0,1,2^n+1);
                B_next = zeros(1,length(D_next));
                j=1;
                for i=1:2^n+1
                    if mod(i,2)== 0;
                        B_next(i)= sqrt(1/(2^(n+1)))*randn;
                    else
                        B_next(i)= B_0(j);
                        j=j+1;
                    end
                end
                for i=1:2^n+1
                    if mod(i,2)== 0;
                        B_next(i)=B_next(i)+(B_0(i/2)+B_0(i/2+1))/2;
                    end
                end
                D_0 = D_next;
                B_0 = B_next;
                if n==2 || n==4 || n==5 || n==6
                    plot(D_0,B_0)
                end
            end
            legend('n = 4','n = 16', 'n = 32','n = 64')
            hold off;
        end
        function CoV()
            %plots the covariancefunction of Brownian motion
            [X,Y] = meshgrid(0:.05:1);
            Z = min(X,Y);
            surf(X,Y,Z)
            title('Kovarianzfunktion der Brownschen Bewegung')
            xlable('time')
        end
        function KLTlambda()
            %plots the Eigenvalues
            [Z,lambda,psi] = BrMo.compute_KLT_components(10);
            plot(linspace(1,10,10),lambda,'*')
            title('KLT Coefficients')
            xlabel('index')
            ylabel('lambda_i')
        end
        function CoVBB()
```

```
            %plots the covariancefunction of Brownian bridge
            [X,Y] = meshgrid(0:.05:1);
            Z = min(X,Y) - X.*Y
            surf(X,Y,Z)
            title('Kovarianzfunktion der Brownschen Brücke')
        end
        function Mercer()
            %plots different approximations of the covariancefunction of
            %Brownian motion using Mercers theorem
            [Z,lambda,psi] = BrMo.compute_KLT_components(10);
            figure
            A = zeros(20,20);
            title('Mercers Theorem - covariance function of Brownian Motion');
            hold on;
            for n = 1:4
                for i = 1 : 20
                    for j = 1:20
                        A(i,j) = A(i,j) + lambda(n)*psi(n,50*i)*psi(n,50*j);
                    end
                end
            end
            [X,Y] = meshgrid(0.05:0.05:1);
            surf(X,Y,A)
        end
        function value = mother(t)
            %%%Haar mother function%%%%
            if ((t<0.5) && (t >= 0))
                value = 1;
            elseif ((t>=0.5) && (t<1))
                value = -1;
            else
                value = 0;
            end
        end
        function psi = Haar(N)
            steps = linspace(1/N,1,N);
            %%Use the Haar Mother function%%%%
            psi(:,1) = ones(1,N);
            psi(N,1) = 0;
            n=0;
            k=0;
```

```
           count =2;
           while count <N+1
               max = 2^n;
               if k == max
                   n=n+1;
                   k=0;
                   continue
               end
               for t=1:N
                    psi(t,count) = 2^(n/2)*BrMo.mother(2^n*steps(t)-k);
               end
               count = count+1;
               k=k+1;
           end
       end
   end
end
```