

```

1 %%This class is a collection of different Kernels and integration✓
methods%%
2 %%to solve the Fredholm integralequation. It also provides a%%
3 %%Merverapproximation for the kernels. %%
4 %%written by Tim Jaschek as a part of his bachelor thesis%%
5
6 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7 classdef Kernels
8     properties (Constant)
9     end
10    methods (Static)
11        function K_st = Kernel(i,s,t)
12            %this is a collection of covariance functions%
13            if i == 1
14                %Brownian Motion%
15                K_st = min(s,t);
16            elseif i==2
17                %Brownian Bridge%
18                K_st = min(s,t) - s*t;
19            elseif i==3
20                %exponential kernel%
21                K_st = exp(-abs(t-s));
22            else
23                K_st = 0;
24            end
25        end
26        function Mat = KMat(i,N)
27            %for given N and i, this function will generate an NxN✓
matrix
28            %for the i-th kernel.
29            Mat = zeros(N+2,N+2);
30            for j = 1:N+2
31                for k = 1:N+2
32                    %use symmetry to save operations
33                    if k<j
34                        Mat(j,k) = Mat(k,j);
35                    else
36                        Mat(j,k) = Kernels.Kernel(i,(j-1)/(N+1),✓
(k-1)/(N+1));
37                    end
38                end
39            end
40        end
41        function [lambda,Phi] = uniform_Scheme(K)

```

```

42         %UNIFORM SCHEME
43         N = length(K)-2;
44         sqW = sqrt(1/(N+2)) * eye(N+2);
45         Mat = sqW*K*sqW;
46         [V,D]=eig(Mat);
47         [lambda,ind] = sort(diag(D),'descend');
48         E_vectors = V(:,ind);
49         Phi = sqrt(N+2)*E_vectors;
50         %%bring the EV in the right direction%%
51         for i = 1:N+2
52             if Phi(2,i)<0
53                 Phi(:,i)=-Phi(:,i);
54             end
55         end
56     end
57     function [lambda,Phi] = trapez_Sceme(K)
58         %TRAPEZ SCHEME
59         N = length(K)-2;
60         sqW = sqrt(1/(N+1)) * eye(N+2);
61         sqW(1,1) = sqrt(1/(2*(N+1)));
62         sqW(N+2,N+2) = sqW(1,1);
63         qW = sqrt(N+2)*eye(N+2);
64         qW(1,1) = sqrt(2*(N+1));
65         qW(N+2,N+2) = qW(1,1);
66         Mat = sqW*K*sqW;
67         [V,D]=eig(Mat);
68         [lambda,ind] = sort(diag(D),'descend');
69         E_vectors = V(:,ind);
70         Phi = qW*E_vectors;
71         %%bring the EV in the right direction%%
72         for i = 1:N+2
73             if Phi(2,i)<0
74                 Phi(:,i)=-Phi(:,i);
75             end
76         end
77     end
78     function [lambda,Phi] = simpson_Sceme(K)
79         %SIMPSON SCHEME
80         N = length(K)-2;
81         sqW = sqrt(1/(3*(N+1))) * eye(N+2);
82         qW = sqrt(3*(N+1))*eye(N+2);
83         for i = 2:N+1
84             sqW(i,i)=sqW(i,i)*sqrt(2);
85             qW(i,i)=qW(i,i)/sqrt(2);

```

```

86         end
87         for i = 2:(N+1)/2+1
88             j=2*(i-1);
89             sqW(j,j)=sqW(j,j)*sqrt(2);
90             qW(j,j)=qW(j,j)/sqrt(2);
91         end
92         Mat = sqW*K*sqW;
93         [V,D]=eig(Mat);
94         [lambda,ind] = sort(diag(D),'descend');
95         E_vectors = V(:,ind);
96         Phi = qW*E_vectors;
97         %%bring the EV in the right direction%%
98         for i = 1:N+2
99             if Phi(2,i)<0
100                 Phi(:,i)=-Phi(:,i);
101             end
102         end
103     end
104     function K = MercerApprox(lambda,Phi,n)
105         %INPUT: lambda - Eigenvalues,
106         %         Phi - Eigenfunctions,
107         %         n - summations
108
109         %OUTPUT: K as approximation of covariance matrix
110         N = length(lambda)-2;
111         K=zeros(N+2,N+2);
112         for s=1:N+2
113             for t=1:N+2
114                 if t<s
115                     K(s,t) = K(t,s);
116                 else
117                     for i=1:n
118                         K(s,t) = K(s,t) + lambda(i)*Phi(s,i) ✓
119                         *Phi(t,i);
120                     end
121                 end
122             end
123         end
124     end
125 end
126 end
127

```