Table of Contents

COMPARISON BETWEEN CUSTOM AND MATLAB BUILT-IN SPLINE INTERPOLATION 1
--> Function for calculating the determinant of a square matrix 2
--> Function for calculating the inverse of a square matrix 3

COMPARISON BETWEEN CUSTOM AND MATLAB BUILT-IN SPLINE INTERPOLATION

```
close all; clearvars; clc

Fc = 5e3;
Nsamps = 500;
Fs1 = 240e3;
Fs2 = 250e3;

Time1 = 1/Fs1*(0:Nsamps-1);
Wave1 = sin(2*pi*Fc*Time1);                     % Original signal

Time2 = 0:1/Fs2:Time1(end);
Wave2 = spline(Time1,Wave1,Time2);              % Matlab built-in
 interpolation

Time2 = 0:1/Fs2:Time1(end);
Wave3 = Interp(Time1,Wave1,Time2);              % Custom interpolation

figure; hold on
plot(Time1,Wave1,'g*-.')
plot(Time2,Wave2,'b*-')
plot(Time2,Wave3,'ro-')
legend('Original','MaBI','Custom')
ylim(1.2*[-1 1]);
grid on; hold off

function [ WaveB ] = Interp( TimeA, WaveA, TimeB )

    Ts = TimeA(2);                  % Sample period of WaveA [s]
    LenA = length(WaveA);           % Length of WaveA [Sa]
    dx = Ts*ones(LenA-1,1);
    DerWaveA = diff(WaveA)/Ts;      % Derivate function of WaveA
    if LenA < 3
        error('Input waveform must have a sample length > 2');
    end
    TmpWave = zeros(1,LenA);
    TmpWave(2:LenA-1) = 3*Ts*(DerWaveA(1:LenA-2)+DerWaveA(2:LenA-1));
    TmpWave(1) = (5*Ts*DerWaveA(1)+Ts*DerWaveA(2))/2;
    TmpWave(LenA) = (Ts*DerWaveA(LenA-2)+5*Ts*DerWaveA(LenA-1))/2;

    InSpDiag = [ [2*Ts;dx(1:LenA-2);0]
 [dx(2);2*(dx(2:LenA-1)+dx(1:LenA-2));dx(LenA-2)] [0;dx(2:LenA-1);2*Ts] ];
```

```
    c = GenSpMtx(InSpDiag,[-1 0 1]',LenA);
    s = TmpWave/c;
%     s = TmpWave*GetInvMtx(c);

    pp = pwch(TimeA,WaveA,s,dx',DerWaveA); pp.dim = 1;
    WaveB = ppval(pp,TimeB);

end


function [ res1 ] = GenSpMtx( In1, In2, Dim )

    p = length(In2);
    len = [0 Dim-1 2*Dim-1 3*Dim-2];      % Compute lengths of diagonals
    a = zeros(len(p+1),3);
      for k = 1:p
         % Append new In2(k)-th diagonal to compact form
         i = (max(1,1-In2(k)):min(Dim,Dim-In2(k)))';
         a((len(k)+1):len(k+1),:) = [i i+In2(k) In1(i+(Dim>=Dim)*In2(k),k)];
      end

    res1 = sparse(a(:,1),a(:,2),a(:,3),Dim,Dim);
%     find(a(:,1)==1)

end
```

--> Function for calculating the determinant of a square matrix

```
function [ Det ] = GetMtxDet( Mtx )

    Dim = length(Mtx(1,:));
    if Dim < 2
        Det = Mtx;
    elseif Dim == 2
        Det = Mtx(1,1)*Mtx(2,2)-Mtx(1,2)*Mtx(2,1);
    elseif Dim == 3
        Det = Mtx(1,1)*Mtx(2,2)*Mtx(3,3)+Mtx(1,2)*Mtx(2,3)*Mtx(3,1)+ ...
            Mtx(2,1)*Mtx(3,2)*Mtx(1,3)-Mtx(1,3)*Mtx(2,2)*Mtx(3,1)- ...
            Mtx(3,2)*Mtx(2,3)*Mtx(1,1)-Mtx(3,3)*Mtx(2,1)*Mtx(1,2);
    else
        Det = 0;
        for j = 1:Dim
            if Mtx(1,j) ~= 0
                Sign = 2*mod(j,2)-1;
                SubMtx = Mtx;
                SubMtx(1,:) = [];
                SubMtx(:,j) = [];
                Det = Det+Sign*Mtx(1,j)*GetMtxDet(SubMtx);
            end
        end
    end
```

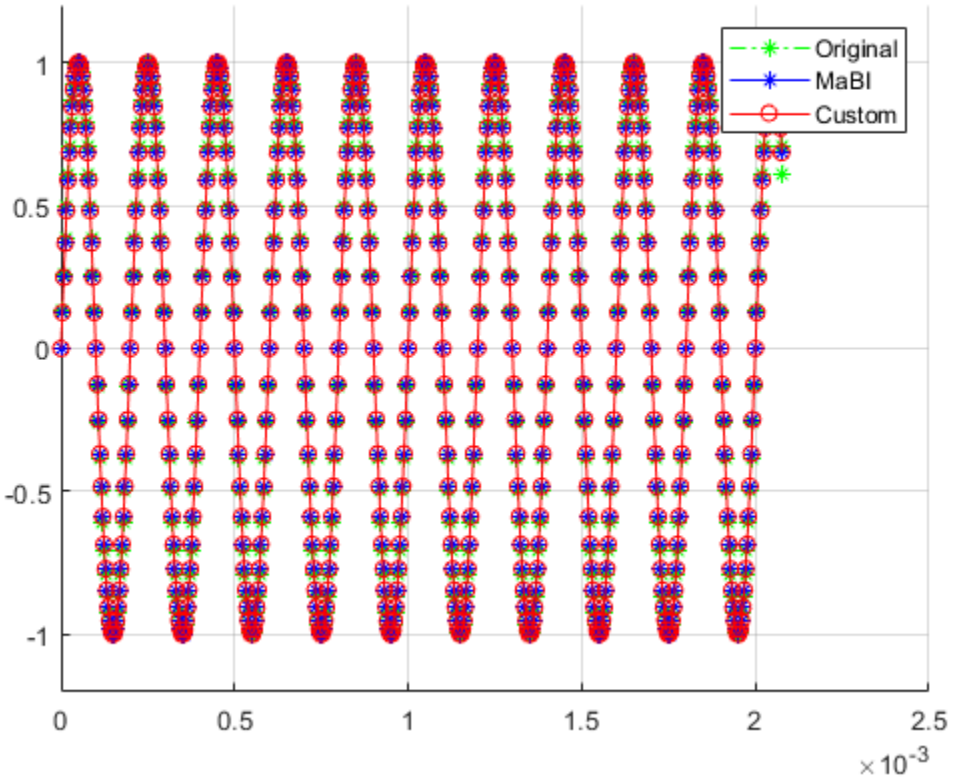
```
end
```

--> Function for calculating the inverse of a square matrix

```
function [ InvMtx ] = GetInvMtx( Mtx )

    Dim = length(Mtx(1,:));
    Det = GetMtxDet(Mtx);
    AdjMtx = zeros(Dim);
    for i = 1:Dim
        for j = 1:Dim
            if Mtx(i,j) ~= 0
                Sign = 1-2*mod(i+j,2);
                SubMtx = Mtx;
                SubMtx(i,:) = [];
                SubMtx(:,j) = [];
                AdjMtx(j,i) = Sign*GetMtxDet(SubMtx);
            end
        end
    end
    InvMtx = AdjMtx/Det;

end
```



Published with MATLAB® R2022a