Generate Admissible Functions using DOPbox

Matthew Harker and Paul O'Leary
Institute for Automation
University of Leoben
A-8700 Leoben, Austria
URL: automation.unileoben.ac.at

Original: January 9, 2013

Last Modified: April 11, 2013

Contents

This function demonstrates the use of the `DOPbox` for the generation of admissible functions for a BVP and for the solution of the BVP, more details on this method can be found in [1].

prepare tghe matlab environment

```
close all;
clear;
setUpGraphics;
```

1 Synthesize Admissible Functions

Define the number of points and the number of basis functions to be generated.

```
noPts = 100;
noBfs = 15;
%
% Generate the vector of x values at which the problem is to be solved
%
x = linspace(0,1,noPts)';
%
% Synthesize the basis functions
%
[B, dB] = dop( x , noBfs );
%
% generate a local differentiating matrix
%
ls = 3;
noBfsD = ls;
D = dopDiffLocal( x, ls, noBfsD );
%
% Compute the second derivative
%
D2 = D * D;
D3 = D2 * D;
%----------------------------------------
% Define the constraints
%
% 1) Position constraint
c1 = zeros( noPts, 1);
c1(1) = 1;
%
% 2) derivative constraint
c2 = D(1,:)';
%
% 3) second derivatice constraint
c3 = D2(end,:)';
%
% 4) a positional constraint at 0.7
c4 = zeros( noPts, 1 );
```

```
frac = 0.7;
at = round( noPts * frac );
c4( at ) = 1;
%
c5 = D3(end,:)';
%
% form the constraint matrix
%
C = [c1, c2, c3, c4, c5];
%
% Synthesize the admissible functions
%
Bc = dopConstrain( C, B );
%
% plot the first three admissible functions
%
fig1 = figure;
for k=1:3
   plot( x, Bc(:,k), 'k');
   hold on;
end;
title('Admissible Functions for the BVP');
xlabel('x');
ylabel('y(x)');
grid on;
```

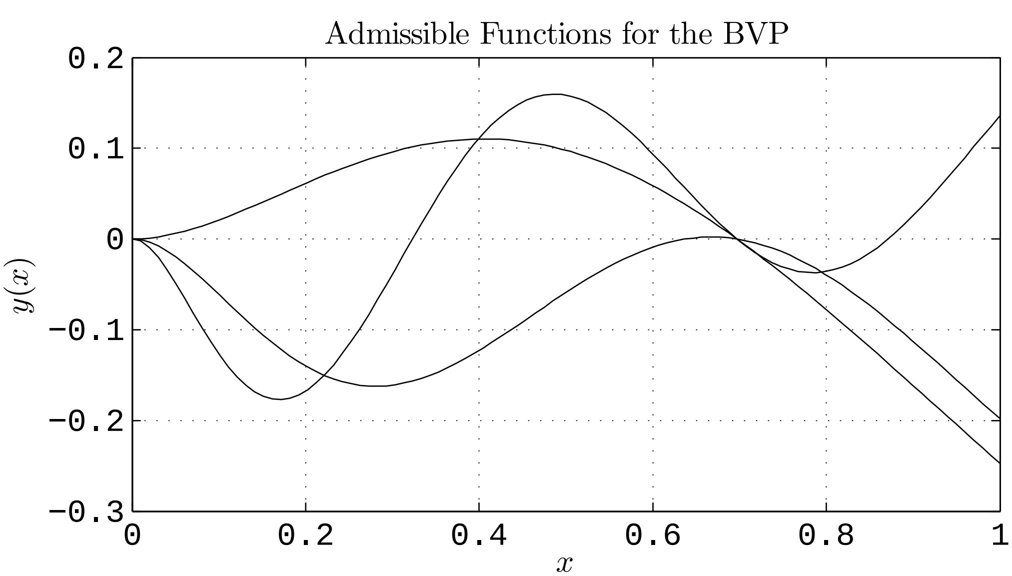

Figure 1: The first three admissible functions.

2 Solve the Differential Equation

Define the linear differential opertor.

```
L = Bc' * D^4 * Bc;
%
% Compute the eigenvalues and eigenvectors
%
[vec, val] = eig( L );
%
% Sort the eigenvalues and vectors in assending order.
%
[val, inds] = sort( diag(val) );
vec = vec(:,inds);
%
% Compute the solution vectors from the spectrum.
%
solV = Bc * vec;
%
% Plot the solution vectors
%
fig2 = figure;
for k=1:3
   plot( x, solV(:,k), 'k');
   hold on;
end;
title('Solution to BVP');
xlabel('x');
ylabel('y(x)');
grid on;
```

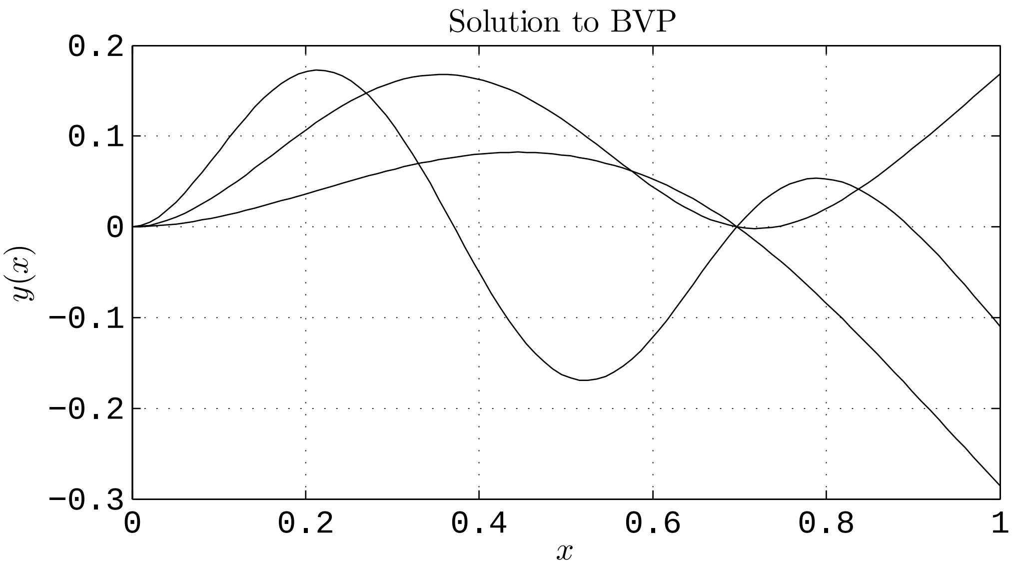

Figure 2: Eigenvectors for the Boundary Value Problem.

3 The spectrum of the Eigenvectors w.r.t. the Admissible Functions

The method implemented here is a discrete equivalent of a Rayleigh-Ritz method. The constrained polynomials are used for the series approximation. The `vec` matrix contains the spectrum. Only the specrum of the first 4 eigenvectors w.r.t. the first 10 basis functions are displayed here.

```
disp('The spectrum of the first 4 eigenvectors with respect to the first ');
disp('10 basis functions');
vec(1:10,1:4)
```

```
The spectrum of the first 4 eigenvectors with respect to the first
10 basis functions

ans =

    0.9790    0.1863    0.0260   -0.0002
    0.2021   -0.9539   -0.2252   -0.0505
   -0.0282    0.2332   -0.9354   -0.1464
   -0.0011    0.0063   -0.2500    0.8095
    0.0037   -0.0274    0.0906    0.5598
    0.0021   -0.0131    0.0525    0.0796
   -0.0000    0.0005    0.0012    0.0235
    0.0007   -0.0043    0.0120    0.0225
    0.0004   -0.0022    0.0059    0.0069
    0.0000   -0.0001   -0.0002   -0.0032
```

References

[1] Paul O'Leary and Matthew Harker. A framework for the evaluation of inclinometer data in the measurement of structures. *IEEE T. Instrumentation and Measurement*, 61(5):1237–1251, 2012.