PDE Project: Problem 7.2.4

Timothy Fraizer

Lani Oramas

Jason Williams

Ryan Wojtyla

December 6, 2018

Part A

Part A of the problem asks us to plot $y_n = e^{\frac{x^2}{2}} H_n(x)$ when $n = [0, 1, 2, 3]$, where $H_n(x)$ is the Hermite polynomial. The plot, Figure 2, was generated with the code displayed in Figure 1.

```
% declare symbol to be used in function
syms x;

% calculate hermite polynomials
h = hermiteH([0,1,2,3],x);

% calculate the functions to be plotted
y=exp((x^2)/2) * h;

% plot the functions
fplot(y)
```

Figure 1: The MATLAB code for generating the plots of $y_n = e^{\frac{x^2}{2}} H_n(x)$ for $n = [0, 1, 2, 3]$.

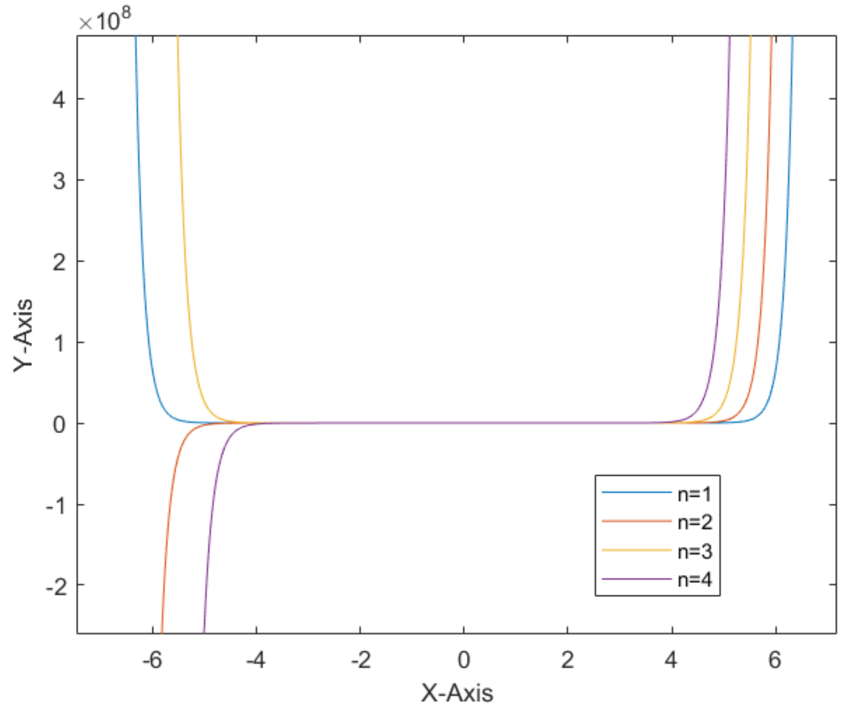

Figure 2: The plots of $y_n = e^{\frac{x^2}{2}} H_n(x)$ for $n = [0, 1, 2, 3]$.

Part B

Part B asks us to use the MATLAB function `bvp4c` to find the first four eigenfunctions of the second order differential equation $y\prime\prime + (\lambda - x^2)y = 0$ in the range $-L \leq x \leq L$ with the boundary conditions $y(-L) = y(L) = 0$. Unfortunately, we had some trouble implementing `bvp4c` in code, Figure 3. We could not come up with a valid initial guess of the solution, the function on line 27. MATLAB complained that a singular Jacobian was encountered, which indicates that the solution diverges. We do not know how to guess a solution that does not produce that error, i.e. one that does not diverge.

```matlab
function mat4bvp

lambda = 0;
solinit = bvpinit(linspace(-100,100),@mat4init,lambda);
sol = bvp4c(@mat4ode,@mat4bc,solinit);

fprintf('The fourth eigenvalue is approximately %7.3f.\n',...
          sol.parameters)

xint = linspace(-100,100);
Sxint = deval(sol,xint);
plot(xint,Sxint(1,:))
axis([-100 100 -100 100])
title('Eigenfunction of Hermite polynomial.')
xlabel('x')
ylabel('solution y')
% ------------------------------------------------------------------
function dydx = mat4ode(x,y,lambda)
dydx = [  y(2)
          -(lambda - (x^2)) * y(1) ];
% ------------------------------------------------------------------
function res = mat4bc(ya,yb,lambda)
res = [  ya(1)
         yb(1)
         0 ];
% ------------------------------------------------------------------
function yinit = mat4init(x)
yinit = [ x
          1 ];
% singular Jacobian error is because guess is divergent
```

Figure 3: The attempted code for finding the eigenfunctions of Part B.