Implementing shooting method for boundary value problems

$$\frac{d^2\theta}{dt} = -sin(\theta(t)) \tag{1}$$

Given $\theta(0) = 0$ and $\theta(2) = \pi/2$, find $\theta(t)$.

CODE:

Define right-hand side, as for initial value ODEs.

```
function rhs = pendulum_rhs(t,y)

rhs = [y(2)
      -sin(y(1))];
```

SETUP: `pendulum_shoot_bisection.m`

```
TOLERANCE = 10^(-4);  % Tolerance for the bisection
max_iterations = 100;  % Maximum number of bisection steps
tspan = [0 2];  % Domain

%bounds for range of A that will search for solution
AL=.1 ;  %left and right-hand endpoints
AR=3 ;
```

Check whether bisection will work: D(AL) and D(AR) must have opposite signs

```
    % Calculate D(AL)
    y0 = [0 AL];  % Construct the initial condition
    [t,y] = ode45('pendulum_rhs',tspan,y0);  % Solve the ODE
    DAL=y(end,1)-pi/2;

    % Calculate D(AR)
    y0 = [0 AR];  % Construct the initial condition
    [t,y] = ode45('pendulum_rhs',tspan,y0);  % Solve the ODE
    DAR=y(end,1)-pi/2;
```

If this is not satisfied, code breaks – must choose different starting values of AL and AR

NEXT: begin bisection loop

```
for i=1:max_iterations
    AC = (AL+AR)/2 % Find the midpoint of the interval
```

- compute D(AL), D(AR), D(AC)
- Bisection step: if D(AC) and D(AL) have same sign, throw away left half (move left endpt. to center)

```
    if DAC*DAL>0
        AL=AC
```

- else throw away right half (move right endpt. to center)

```
    else
        AR=AC
    end
```

- Check to see if we are close to solution (is $|D(AC)|$ below tolerance?)

```
    if ( abs(DAC)<TOLERANCE )
        break
    end

end % end for loop
```

Schrödinger equation from quantum mechanics:

$$\frac{d^2\Psi_n}{dx^2} = (\beta_n - n_0)\Psi_n \qquad \Psi_n(\pm 1) = 0 \tag{2}$$

Rewrite as a first order system:

$$\frac{d\phi_1}{dx} = \phi_2 \tag{3}$$

$$\frac{d\phi_2}{dx} = (\beta_n - n_0)\phi_1 \tag{4}$$

Boundary conditions:

$$\phi_1(-1) = 0 \qquad \phi_1(1) = 0 \tag{5}$$

$$\phi_2(-1) = 1 \tag{6}$$

A NEW TYPE OF SHOOTING PROBLEM:

- SPECIFY FULL set of initial conditions.
- Shoot to find $\bar{\beta}$: values such that if $\beta_n = \bar{\beta}$, we satisfy $\phi_1(1) = 0$
- $\bar{\beta}$ are energy levels for Schrodinger eqn!
- There are MANY $\bar{\beta}$. Say we want to find 5 of them.

Search-and-bisection method:
Given prior knowledge: e.g. $\phi_1(1)$ *decreases as* β *decreases*.

- Start at $\beta = \beta_{start}$
- If $\phi_1(1) > 0$, $\beta \to \beta - \Delta\beta$
- If $\phi_1(1) < 0$, $\beta \to \beta + \Delta\beta/2$ ..... and $\Delta\beta \to \Delta\beta/2$
- After converge to $\bar{\beta}$, restart with $\beta_{start} = \bar{\beta} - 1$

[[sketch of search algorithm]]

CODES: First, `shoot2.m`

define right hand side for the schrodinger equation
2 parameters: n0, beta

```
function rhs=shoot2(xspan,phi,dummy,n0,beta)
rhs = [phi(2) ; (beta-n0)*phi(1)];
```

CODE: `schrodinger_shoot.m`

INNER LOOP: BISECTION TO FIND EACH $\bar{\beta}$:

```
for j=1:1000    %number of bisection iterations
        [x,y] = ode45('shoot2',xspan,x0,[],n0,beta);  % Solve the ODE

        if abs(y(end,1)-0) < TOLERANCE  % converged yet?
            break;
        end

        if (-1)^(modes+1)*y(end,1)>0    %...then must decrease beta
                     beta=beta - dbeta;
        else                            % ... must increase beta
              beta=beta + dbeta/2;
              dbeta=dbeta/2;
        end

end
```

CODE: `schrodinger_shoot.m`

Key idea: NEST BISECTION LOOP INSIDE "MODE" LOOP TO FIND MULTIPLE $\bar{\beta}$:

```
for modes=1:5  %loop over number of beta that want

    beta = beta_start;  % Reset for next solution beta
    dbeta = n0/100;  %  Reset for next solution beta

         [[BISECTION LOOP, CONVERGING TO beta]]

    beta_start = beta-0.1;  % Change the starting value of beta  so w
                                        % find the next one
end  %loop over modes
```