Two-point Boundary Value Problems: Numerical Approaches

Math 615, Spring 2010
abbreviations

- ODE = ordinary differential equation
- PDE = partial differential equation
- IVP = initial value problem
- BVP = boundary value problem
- MOP = MATLAB or OCTAVE or PYLAB
Outline

1. classical IVPs and BVPs with by-hand solutions
2. a more serious example: a BVP for equilibrium heat
3. finite difference solution of two-point BVPs
4. shooting to solve two-point BVPs
5. a more serious example: solutions
6. exercises
classical ODE problems: IVP vs BVP

Example 1: ODE IVP. find $y(x)$ if

$$y'' + 2y' - 8y = 0, \quad y(0) = 1, \quad y'(0) = 0$$

Example 2: ODE BVP. find $y(x)$ if

$$y'' + 2y' - 8y = 0, \quad y(0) = 1, \quad y(1) = 0$$
classical ODE problems: IVP vs BVP

Example 1: ODE IVP. find $y(x)$ if

$$y'' + 2y' - 8y = 0, \quad y(0) = 1, \quad y'(0) = 0$$

Example 2: ODE BVP. find $y(x)$ if

$$y'' + 2y' - 8y = 0, \quad y(0) = 1, \quad y(1) = 0$$

• both problems can be solved by hand

• in fact, the ODE has constant coefficients so we can find characteristic polynomial and general solution ... like this:

if $y(x) = e^{rx}$ then $r^2 + 2r - 8 = (r + 4)(r - 2) = 0$ so

$$y(x) = c_1 e^{-4x} + c_2 e^{2x}$$

• Example 1 gives system $c_1 + c_2 = 1, \ -4c_1 + 2c_2 = 0$ for coefficients; get solution $y(x) = (1/3)e^{-4x} + (2/3)e^{2x}$

• Example 2 gives system $c_1 + c_2 = 1, \ e^{-4}c_1 + e^{2}c_2 = 0$ for coefficients; get solution

$$y(x) = (1 - e^{-6})^{-1}e^{-4x} + (1 - e^{6})^{-1}e^{2x}$$
just for practice: viewing solns with MATLAB/OCTAVE

```matlab
x = 0:.001:1;
y1 = exp(-4*x); y2 = exp(2*x);
yIVP = (1/3)*y1 + (2/3)*y2;
yBVP = (1/(1-exp(-6)))*y1 + (1/(1-exp(6)))*y2;
plot(x,yIVP,x,yBVP), grid on
legend('IVP soln','BVP soln')
```

![Graph showing IVP and BVP solutions](image_url)
obvious name: “two-point BVP”

again:

Example 2: ODE BVP. find $y(x)$ if

$$y'' + 2y' - 8y = 0, \quad y(0) = 1, \quad y(1) = 0$$

- Example 2 is called a “two-point BVP” because the solution is known at two points (duh!)
- a two-point BVP includes an ODE and the value of the solution at two different locations
- the ODE can be of any order, as long as it is at least two, because first-order ODEs cannot satisfy two conditions (generally)
- but there is no guarantee that a two-point BVP can be solved (see below), even though that is the usual case
- we will also be considering boundary value problems for PDEs in this course (i.e. problems including no initial values); these are “∞-point BVPs” I suppose
recall: a standard manipulation of a 2nd order ODE
Consider the general linear 2nd-order ODE:

\[ y'' + p(x)y' + q(x)y = r(x) \]  \hspace{1cm} (1)

Also consider the (almost-completely) general 2nd-order ODE:

\[ y'' = f(x, y, y') \]  \hspace{1cm} (2)

- these can be written as systems of coupled 1st-order ODEs
- in fact, equation (1) is equivalent to

\[
\begin{pmatrix}
y'
\end{pmatrix} =
\begin{pmatrix}
v \\
v
\end{pmatrix} =
\begin{pmatrix}
-v \\
-p(x)v - q(x)y + r(x)
\end{pmatrix}
\]

- and equation (2) is equivalent to

\[
\begin{pmatrix}
y'
\end{pmatrix} =
\begin{pmatrix}
v \\
v
\end{pmatrix} =
\begin{pmatrix}
f(x, y, v)
\end{pmatrix}
\]

- first order systems are the form in which we can apply a numerical ODE solver to solve both IVPs and BVPs
- ... but BVPs generally require additional iteration
why IVP are *better* problems than BVPs

- IVPs with well-behaved parts do have unique solutions
- we say they are “well-posed”; specifically:
- **Theorem.** Consider the system of ODEs

\[ y = f(t, y), \quad (3) \]

where \( y(t) = (y_1(t), \ldots, y_d(t)) \) and \( f = (f_1, \ldots, f_d) \) are vector-valued functions. If \( f \) is continuous for \( t \) in an interval around \( t_0 \) and for \( y \) in some region around \( y_0 \), and if \( \partial f_i / \partial y_j \) is continuous for the same inputs and for all \( i \) and \( j \), then the IVP consisting of (3) and \( y(t_0) = y_0 \) has a unique solution \( y(t) \) for at least some small interval \( t_0 - \epsilon < t < t_0 + \epsilon \) for some \( \epsilon > 0 \).

- given comments on last slide, the theorem covers IVPs for 2nd-order scalar ODEs
**Warning about apparently-easy BVPs**

*Example 3: ODE BVP.* Find $y(x)$ if

$$y'' + \pi^2 y = 0, \quad y(0) = 1, \quad y(1) = 0$$

- This turns out to be impossible . . . there is no such $y(x)$.
- In fact, the general solution to the ODE is

$$y(x) = c_1 \cos(\pi x) + c_2 \sin(\pi x)$$

so the first boundary condition implies $c_1 = 1$ (because $\sin(0) = 0$).
- . . . but then the second condition says

  $$0 = y(1) = -1 + c_2 \sin(\pi)$$

and this has no solution because $\sin(\pi) = 0$.
- This is a constant-coefficient problem for which all the “parts” are “well-behaved”; we can even easily write down the general solution!
two-point BVPs related to eigenvalue problems

- homogeneous linear two-point BVPs like

\[ y'' + p(x)y' + q(x)y = \lambda y, \quad y(a) = 0, \quad y(b) = 0 \quad (4) \]

are called *Sturm-Liouville* problems

- they are analogous to eigenvalue problems “\( Ax = \lambda x \)” where the \( \lambda \) values *and* the vectors \( x \) are unknown
  - \( \lambda \) is an *eigenvalue*; there are finitely-many
  - \( x \neq 0 \) is an *eigenvector* associated to \( \lambda \)

- in the Sturm-Liouville problem (4), the “matrix” is the operator

\[ A = \frac{d}{dx} + p(x) \frac{d}{dx} + q(x) \]

(though the operator \( A \) must somehow also include the homogeneous boundary conditions)

- in (4) we seek eigenvalues \( \lambda = \lambda_n \), which come in an infinite-but-countable list, and their associated eigenfunctions \( y = y_n(x) \)

- Sturm-Liouville theory “explains” the impossible case on the previous slide . . . *but* this Sturm-Liouville thread will not be pursued further here . . .
Outline

1. classical IVPs and BVPs with by-hand solutions
2. a more serious example: a BVP for equilibrium heat
3. finite difference solution of two-point BVPs
4. shooting to solve two-point BVPs
5. a more serious example: solutions
6. exercises
an equilibrium heat example

- as noted in lecture and by Morton & Mayers, a PDE like this is a general description of heat flow in a rod:

\[ \rho c \frac{\partial u}{\partial t} = \frac{\partial}{\partial x} \left( k(x) \frac{\partial u}{\partial x} \right) + r(x)u + s(x) \quad (5) \]

- recall that, roughly speaking, \( \rho \) is a density, \( c \) a specific heat, \( k \) a conductivity, \( r(x) \) a reaction coefficient (because \( r(x)u \) is the heat produced by a temperature-dependent chemical reaction, for example), and \( s(x) \) is an external (\( u \) independent) source of heat
an equilibrium heat example, cont

- **equilibrium** means no change in time; the equilibrium version of (5) is this:

\[ 0 = \frac{\partial}{\partial x} \left( k(x) \frac{\partial u}{\partial x} \right) + r(x)u + s(x) \]

- because we can use ordinary derivative notation, and slightly-rearrange, the equilibrium equation is an ODE:

\[ (k(x)u')' + r(x)u = -s(x) \]

(6)

- let’s suppose the rod has length \( L \), and \( 0 \leq x \leq L \)

- example boundary values are (i) insulation at the left end and (ii) pre-determined temperature at the right end:

\[ u'(0) = 0, \quad u(L) = 0 \]

(7)
an equilibrium heat example, cont, cont

- some concrete, generally-non-constant choices in my example include $L = 3$ and:

$$k(x) = \frac{1}{2} \arctan(20(x - 1)) + 1,$$

$$r(x) = r_0 = \frac{1}{2}, \quad s(x) = e^{-(x-2)^2}$$
Two-point Boundary Value Problems: Numerical Approaches

Bueler

classical IVPs and BVPs

serious example

finite difference

shooting

serious example: solved

exercises

• code used to produce the previous picture

L = 3;
k = @(x) 0.5 * atan((x-1.0) * 20.0) + 1.0;
r0 = 0.5;
s = @(x) exp(-(x-2.0).^2);

J = 300;
dx = L / J;
x = 0:dx:L;
plot(x,k(x),x,r0*ones(size(x)),x,s(x))
grid on, xlabel x
legend('k(x)', 'r(x)=r_0', 's(x)')
summary of “serious example”

- we now have a non-constant-coefficient boundary value problem to solve:
  \[(k(x)u')' + r_0 u = -s(x), \quad u'(0) = 0, \quad u(3) = 0\]  
  (8)

- \(u(x)\) represents the equilibrium distribution of temperature in a rod with these properties:
  - conductivity \(k(x)\): the first third \([0, 1]\) is a material with much lower conductivity than the last two-thirds \([2, 3]\)
  - reaction rate \(r_0 > 0\): constant rate of linear-in-temperature heating
  - source term \(s(x)\): an external heat source concentrated around \(x = 2\)

- worth drawing a picture of the rod and its surroundings: shading for \(k(x)\), candles for \(s(x)\), insulated end, refrigerated end, ...

- a concrete Question: what is \(u(0)\), the temperature at the left end?
plan from here

1. introduce finite difference approach on really-easy “toy” two-point BVP
2. introduce shooting method on same toy problem
3. demonstrate both approaches on “serious problem”
Outline

1. classical IVPs and BVPs with by-hand solutions
2. a more serious example: a BVP for equilibrium heat
3. finite difference solution of two-point BVPs
4. shooting to solve two-point BVPs
5. a more serious example: solutions
6. exercises
finite differences

- finite difference methods for two-point BVPs generalize to PDEs . . . as demonstrated in the rest of Math 615!
- but here we are just solving ODEs

- recall I showed this using a Taylor’s-theorem-with-remainder argument:

\[
\frac{f(x - h) - 2f(x) + f(x + h)}{h^2} = f''(x) + \frac{f^{(4)}(\nu)}{12} h^2
\]
toy example problem

- consider this easy BVP:
  
  \[ y'' = 12x^2, \quad y(0) = 0, \quad y(1) = 0 \]

- it has exact solution \( y(x) = x^4 - x \)

- ... please check my last claim

- ... and be sure you could construct this exact solution by integrating
toy example: approximated by finite differences

- cut up the interval $[0, 1]$ into $J$ subintervals:
  \[ \Delta x = \frac{1}{J} \]
  \[ x_j = 0 + (j - 1)\Delta x \quad (j = 1, \ldots, J + 1) \]
- note that my indices run from $j = 1$ to $j = J + 1$
- let $Y_j$ be the approximation to $y(x_j)$
- for each of $j = 2, \ldots, J$ we approximate \( y'' = 12x^2 \)
  by
  \[ \frac{Y_{j-1} - 2Y_j + Y_{j+1}}{\Delta x^2} = 12x_j^2 \]
- the boundary conditions are: $Y_1 = 0$, $Y_{J+1} = 0$
so now we have a linear system of $J + 1$ equations in $J + 1$ unknowns:

\[
\begin{align*}
Y_1 &= 0 \\
Y_1 - 2Y_2 + Y_3 &= 12\Delta x^2 x_2^2 \\
Y_2 - 2Y_3 + Y_4 &= 12\Delta x^2 x_3^2 \\
&\vdots & \vdots \\
Y_{J-1} - 2Y_J + Y_{J+1} &= 12\Delta x^2 x_J^2 \\
Y_{J+1} &= 0
\end{align*}
\]
**toy example: as matrix problem**

- this is a matrix problem:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & \ldots & 0 \\
1 & -2 & 1 & 0 & \ldots & 0 \\
0 & 1 & -2 & 1 & 0 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \ldots & 1 & -2 & 1 \\
0 & \ldots & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
Y_1 \\
Y_2 \\
Y_3 \\
\vdots \\
Y_J \\
Y_{J+1}
\end{bmatrix}
= \begin{bmatrix}
0 \\
12\Delta x^2 x_2^2 \\
12\Delta x^2 x_3^2 \\
\vdots \\
12\Delta x^2 x_J^2 \\
0
\end{bmatrix}
\]

- i.e.

\[AY = b\]
toy example: as matrix problem in OCTAVE

- the matrix $A$ is \textit{tridiagonal}
- which is usually true of finite difference methods for two-point boundary value problems for second order ODEs
- $A$ has lots of zero entries, so in MATLAB/OCTAVE we store it as a “sparse” matrix
- this means that the \textit{locations} of nonzero entries, and the matrix entries at those locations, are stored; this saves space
- also there are “expert systems” in MATLAB/OCTAVE which recognize sparsity and then try to exploit it to speed up matrix/vector operations
- practical MATLAB/OCTAVE advice: learn how to use \texttt{spy} and \texttt{full} to see these sparse matrix structures
toy example: as matrix problem in OCTAVE, cont

- setting up the matrix problem looks like:
  
  ```matlab
  J = 10; dx = 1/J; x = (0:dx:1)';
  b = zeros(J+1,1);
  b(2:J) = 12 * dx^2 * x(2:J).^2;
  A = sparse(J+1,J+1);
  A(1,1) = 1.0; A(J+1,J+1) = 1.0;
  for j=2:J
    A(j,[j-1, j, j+1]) = [1, -2, 1];
  end
  
  ```

- solving the matrix problem looks like:
  
  ```matlab
  Y = A \ b; % solve A Y = b
  
  ```

- plot on next page from
  
  ```matlab
  % also get exact soln on fine grid:
  xf = 0:1/1000:1; yexact = xf.^4 - xf;
  plot(x,Y,'o','markersize',12,xf,yexact)
  grid on, xlabel x, legend('finite diff','exact')
  ```
toy example: as matrix problem in OCTAVE, cont, cont

- gives result which is better than we have any reason to expect:

![Graph showing finite difference and exact solutions]

- gives result which is better than we have any reason to expect:
toy example with finite differences: brief analysis

regarding how the result on the previous slide can be so suspiciously nice:

- recall that the exact solution is $y(x) = x^4 - x$
- recall we had

$$\frac{f(x - h) - 2f(x) + f(x + h)}{h^2} = f''(x) + \frac{f^{(4)}(\nu)}{12} h^2$$

- applied to $f(x) = y(x)$, for which $y^{(4)}(x) = 24$ is constant, we see that the finite difference approximation to the second derivative in the ODE $y'' = 12x^2$ has error at most

$$\frac{y^{(4)}(\nu)}{12} \Delta x^2 = \frac{24}{12} (0.1)^2 = 0.02$$

because $\Delta x = 0.1$

- this is a rare case where the local truncation error is a known constant . . . and fairly small
toy example with finite differences: brief analysis, cont

- let $e_j = Y_j - y(x_j)$
- by subtraction,
  $$\frac{e_{j-1} - 2e_j + e_{j+1}}{\Delta x^2} = 0.02$$
  and $e_0 = e_{J+1} = 0$
- so (after bit of not-too-hard thought)
  $$e_j = 0.01x_j(x_j - 1)$$
- so
  $$\max_j |Y_j - y(x_j)| = \max_j |e_j| = 0.0025$$
- which explains why picture a few slides back was good ... but showed slight errors close to screen resolution
Outline

1. classical IVPs and BVPs with by-hand solutions
2. a more serious example: a BVP for equilibrium heat
3. finite difference solution of two-point BVPs
4. shooting to solve two-point BVPs
5. a more serious example: solutions
6. exercises
toy example problem again: shooting

- recall this “toy” ODE BVP:

\[ y'' = 12x^2, \quad y(0) = 0, \quad y(1) = 0 \]

(which has exact solution \( y(x) = x^4 - x \))

- this time we think: *if only it were an ODE IVP then we could apply a numerical ODE solver like ode45 or lsode*

- indeed, this ODE IVP

\[ w'' = 12x^2, \quad w(0) = 0, \quad w'(0) = A \]

*can* be solved by a numerical ODE solver, for any \( A \)

- solving this ODE IVP involves “aiming” by guessing an initial slope \( w'(0) = A \)

- … and “hitting the target” is getting the desired boundary value \( w(1) = 0 \) correct, so that \( y(x) = w(x) \) in that case
toy example shooting, cont

- for illustrating the method, I’ll skip the use of a numerical ODE solver because the ODE IVP

\[ w'' = 12x^2, \quad w(0) = 0, \quad w'(0) = A \]

has a solution we can get by-hand:

\[ w(x) = x^4 + Ax \]

- plotting for \( A = -2.5, -1.5, -0.5, 0.5, 1.5 \) gives this figure:
toy example shooting, cont, cont

- we have “aimed” (by choosing $A$) and “shot” five times
- “shot” = (computed the solution to an ODE IVP); generally this would be solving the ODE IVP numerically
- we missed every time
- but we have bracketed the correct right-hand boundary condition $y(1) = 0$ with the two values $A = -1.5$ and $A = -0.5$
- a numerical equation solver can refine the search to converge to the correct $A$ value . . . which we know would by $A = -1$ in this case
- . . . the last idea is best illustrated by example
shooting: solving the boundary condition equation

- recall our ODE BVP

\[ y'' = 12x^2, \quad y(0) = 0, \quad y(1) = 0 \]

is replaced by this ODE IVP when “shooting”:

\[ w'' = 12x^2, \quad w(0) = 0, \quad w'(0) = A \quad (9) \]

- the \( x = 1 \) endpoint value of \( w(x) \) is a function of \( A \):

\[ F(A) = \left( w(1), \text{ where } w \text{ solves (9)} \right) \]

- and so we solve this equation because we want \( y(1) = 0 \):

\[ F(A) = 0 \]

- in this easy problem, \( w(x) = x^4 + Ax \)
- so we solve \( F(A) = 1 + A = 0 \) and get \( A = -1 \)
- generally we solve \( F(A) = 0 \) numerically, e.g. by the \textit{bisection} or \textit{secant} methods
shooting: general strategy for two-point ODE BVPs

- identify one end of the interval \( x = b \) as the target
- at the other end \( x = a \), identify some additional initial conditions which would give a well-posed ODE IVP
- for various guesses of those additional initial conditions, “shoot” by solving the corresponding ODE IVP from \( x = a \) to \( x = b \)
- ask whether you “hit the target” by asking whether the boundary conditions at \( x = b \) are satisfied
- automate the adjustment process by using an equation solver (e.g. bisection or secant method) on the equation that says “the discrepancy between the solution of the ODE IVP at \( x = b \) and the desired boundary conditions at \( x = b \), as a function of the additional initial conditions, should be zero: \( F(A) = 0 \)”
Outline

1. classical IVPs and BVPs with by-hand solutions
2. a more serious example: a BVP for equilibrium heat
3. finite difference solution of two-point BVPs
4. shooting to solve two-point BVPs
5. a more serious example: solutions
6. exercises
recall the serious example

- recall the “serious” non-constant-coefficient BVP:
  \[
  (k(x)u')' + r_0 u = -s(x), \quad u'(0) = 0, \quad u(3) = 0, \quad (10)
  \]

- \(u(x)\) is the equilibrium temperature in a rod
- the conductivity \(k(x)\) has a big jump at \(x = 1\) and the heat source \(s(x)\) is concentrated at \(x = 2\):
finite differences: need staggered grid

- finite difference approach first
- as before: \( J \) subintervals, \( \Delta x = 1/J \), and
  \[
  x_j = (j - 1)\Delta x \quad \text{for } j = 1, \ldots, J + 1
  \]
- let \( U_j \) be our finite diff. approx. to \( u(x_j) \)
- let \( k_j = k(x_j) \) and \( s_j = s(x_j) \); we know these exactly
- note: if \( q(x) = -k(x)u'(x) \)—think Fourier!—then we are solving
  \[
  -q' + r_0 u = -s(x)
  \]
- the finite difference version looks like
  \[
  -q_{j+1/2} - q_{j-1/2} \Delta x + r_0 U_j = -s(x_j)
  \]
- or
  \[
  k(x_{j+1/2}) \frac{U_{j+1} - U_j}{\Delta x} - k(x_{j-1/2}) \frac{U_j - U_{j-1}}{\Delta x} \Delta x + r_0 U_j = -s(x_j)
  \]
finite differences: need staggered grid, cont

- ... or (just notation)

\[
\frac{k_{j+\frac{1}{2}}(U_{j+1} - U_j) - k_{j-\frac{1}{2}}(U_j - U_{j-1})}{\Delta x^2} + r_0 U_j = -s_j
\]

- or (clear denominators)

\[
k_{j+\frac{1}{2}}(U_{j+1} - U_j) - k_{j-\frac{1}{2}}(U_j - U_{j-1}) + r_0 \Delta x^2 U_j = -s_j \Delta x^2
\]

- or

\[
k_{j-\frac{1}{2}} U_{j-1} - \left(k_{j-\frac{1}{2}} + k_{j+\frac{1}{2}} - r_0 \Delta x^2\right) U_j + k_{j+\frac{1}{2}} U_{j+1} = -s_j \Delta x^2
\]

- like the “toy” example earlier, this last form is a tridiagonal matrix equation \( AU = b \)

- note we actually evaluate the conductivity \( k(x) \), and the flux \( q \), on the staggered grid

- the deeper reason why we use the staggered grid will be revealed later in class ...
finite differences: remember the boundary conditions

- recall we have boundary condition \( u'(0) = 0 \)
- approximate this by

\[
\frac{U_2 - U_1}{\Delta x} = 0
\]

- or

\[-U_1 + U_2 = 0\]

- we will see there is a more-accurate way later . . .
- also we have \( u(L) = 0 \) so

\[U_{J+1} = 0\]
finite differences for the “serious problem”

- now for an actual code: see `varheatFD.m` online
- the ODE setup:

```matlab
L = 3;
k = @(x) 0.5 * atan((x-1.0) * 20.0) + 1.0;
s = @(x) exp(-(x-2.0).^2);
r0 = 0.5;

dx = L / J;
x = (0:dx:L)'; % regular grid
xstag = ((dx/2):dx:L-(dx/2))'; % staggered grid
kstag = k(xstag); % k(x) on staggered grid

% right side is J+1 length column vector
b = [0;
    - dx^2 * s(x(2:J));
    0];

% matrix is tridiagonal
A = sparse(J+1,J+1);
A(1,[1 2]) = [-1.0 1.0];
for j=1:J-1
    A(j+1,j) = kstag(j);
    A(j+1,j+1) = - kstag(j) - kstag(j+1) + r0 * dx^2;
    A(j+1,j+2) = kstag(j+1);
end
A(J+1,J+1) = 1.0;
```
finite differences for the “serious problem”, cont

- it is good to use \( \text{spy}(A) \) at this point to see the matrix structure; this is the \( J = 10 \) case
finite differences for the “serious problem”, cont, cont

- the matrix solve:
  \[ U = A \ \backslash \ b; \quad \text{% soln is J+1 column vector} \]

- the plot details:
  ```matlab
  figure(1)
  plot(x,k(x),'r',x,s(x),'b',...
       x,U','g*','markersize',3)
  grid on, xlabel x
  legend('k(x)','s(x)','solution U_j')
  ```
finite difference solution to “serious problem”

• the picture when $J = 60$: 

![Graph showing finite difference solution for a problem with $J = 60$. The graph displays a plot of $u(0) = -5.666658$.](image)
finite difference solution to “serious problem”, cont

- recall our concrete goal was to estimate $u(0)$
- clearly we should try different $J$ values to estimate:

<table>
<thead>
<tr>
<th>J</th>
<th>estimate of $u(0)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>-13.86507</td>
</tr>
<tr>
<td>20</td>
<td>-7.20263</td>
</tr>
<tr>
<td>60</td>
<td>-5.66666</td>
</tr>
<tr>
<td>200</td>
<td>-5.27443</td>
</tr>
<tr>
<td>1000</td>
<td>-5.15199</td>
</tr>
<tr>
<td>4000</td>
<td>-5.12965</td>
</tr>
</tbody>
</table>

- this suggests that $u(0) \approx -5.13$
- *How do we know how wrong we are?*
shooting for the “serious problem”

- shooting is implemented these codes online:
  - `varheatSHOOT.m`: OCTAVE version using `lsode`
  - `varheatSHOOTmat.m`: MATLAB version using `ode45`

- the setup (OCTAVE version):

  ```octave
  L = 3;
k = @(x) 0.5 * atan((x-1.0) * 20.0) + 1.0;
s = @(x) exp(-(x-2.0).^2);
r0 = 0.5;

  % ODE Y’ = G(Y,x) is described by this right-hand side:
  G = @(Y,x) [- Y(2) / k(x); % Y(1) = u
                  r0 * Y(1) + s(x)]; % Y(2) = q

  % bracket unknown u(0)
a = -10.0; % produces u(3) which is too high
b = 0.0; % ... u(3) which is too low
  ```
shooting for the “serious problem”, cont

- the \textit{bisection} implementation (OCTAVE version), which
  starts from initial bracket \([a, b] = [-10.0, 0.0]\):

\begin{verbatim}
N = 100;
for n = 1:N
    fprintf(‘.’)
    c = (a+b)/2;
    % evaluate F(c) = (estimate of u(3) using u(0)=c)
    Y = lsode(G,[c; 0.0],[0.0 3.0]);
    F = Y(2,1);
    if abs(F) < 1e-12
        break % we are done
    elseif F >= 0.0
        a = c;
    else
        b = c;
    end
end
\end{verbatim}
shooting for the “serious problem”, cont

- the finish:

```matlab
% redo to get final version on a grid for plot
x = 0:0.05:3.0;
Y = lsode(G,[c; 0.0],x);
u = Y(:,1)';
qu = Y(:,2)';
figure(2)
plot(x,k(x),'r',x,s(x),'b',x,u,'g*',x,q,'k')
grid on, xlabel x
legend('k(x)','s(x)','u(x)','q(x)')
```
shooting solution to “serious problem”

- the picture:

```
result of SHOOTING:  u(0) = −5.144434
```

- default use of `lsode` gives estimate $u(0) = −5.14443$

- *How do we know how wrong we are?*
minimal conclusion

- finite difference and shooting methods give comparable solutions to this “serious problem”
- closer inspection of the programs above will help understand the methods
- better understanding will also follow from doing the exercises 1 through 5 on the last three slides
- ... which forms Assignment # 3
Outline

1. classical IVPs and BVPs with by-hand solutions
2. a more serious example: a BVP for equilibrium heat
3. finite difference solution of two-point BVPs
4. shooting to solve two-point BVPs
5. a more serious example: solutions
6. exercises
exercises

1. Solve by-hand this ODE BVP to find $y(x)$:

   $$y'' + 2y' + 2y = 0, \quad y(0) = 1, \quad y(1) = 0.$$ 

2. Recall Example 3, an impossible-to-solve ODE BVP. Nonetheless there are some values of $A$ in the following problem which allow a solution: find $y(x)$ if

   $$y'' + \pi^2 y = 0, \quad y(0) = 1, \quad y(1) = A.$$ 

   What values of $A$ are allowed? For an allowed value of $A$, how many solutions are there?

3. Equation (6) has non-constant coefficients, and essentially it cannot be solved exactly by hand. To develop some sense of the effect of the source term $s(x)$, solve by-hand this ODE BVP

   $$(k_0 u')' = -s(x), \quad u'(0) = 0, \quad u(L) = 0,$$

   merely assuming the source is quadratic ($s(x) = ax^2 + bx + c$) and the conductivity is constant ($k_0 > 0$). Compute by-hand $u(0)$. How does the solution $u(x)$ depend on $s(x)$? (For example, how does $u$ depend on the sign, values, slope, or concavity of $s(x)$?)
4 Apply the finite difference method to solve this ODE BVP:

\[ y'' + \sin(5x)y = x^3 - x, \quad y(0) = 0, \quad y(1) = 0. \]

In particular, use \( J = 10, \Delta x = 1/J \), and \( x_j = j\Delta x \) for \( j = 0, \ldots, J \). Construct the system

\[ A \mathbf{y} = \mathbf{b} \]

where \( A \) is a \((J+1) \times (J+1)\) matrix, \( \mathbf{y} = \{ Y_j \} \) approximates the unknowns \( \{ y(x_j) \} \), and \( \mathbf{b} \) contains the right-side function “\( x^3 - x \)” in the ODE. Arrange things so that the first equation in the system represents the boundary condition “\( y(0) = 0 \)” and the last equation the condition “\( y(1) = 0 \)”. The remaining equations in the system will each hold finite difference approximations of the ODE. Show me your matrix \( A \) in a non-wasteful way. Solve the system to find \( \mathbf{y} \), and plot it appropriately. Also write a few sentences addressing how to know qualitatively and quantitatively the degree to which your answer is a good approximation.
Consider the nonlinear ODE BVP

\[ u'' + u^3 = 0, \quad u(0) = 1, \quad u(1) = 0. \]

This problem is well-suited to the shooting method. Specifically, write a MOP program that uses an ODE solver to solve the following ODE IVP

\[ u'' + u^3 = 0, \quad u(0) = 1, \quad u'(0) = A \]

for each of the eleven values \( A = -5, -4, \ldots, 4, 5 \). Plot all eleven solutions, and identify on the plot the \( A \) value for each curve. Which two \( A \) values make the computed value \( u(1) \) bracket the desired value (boundary condition) “\( u(1) = 0 \)”?

(With this information in hand you could make a program like varheatSHOOT.m, which uses bisection to converge to an \( A \) value so that \( u(1) \approx 0 \) to many-digit-accuracy.)