UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING
Department of Electrical & Computer Engineering

ECE 204 *Numerical methods*

Finite-difference methods for boundary-value problems

Douglas Wilhelm Harder, LEL, M.Math.
dwharder@uwaterloo.ca
dwharder@gmail.com

1

Introduction

- In this topic, we will
  - Describe finite-difference approximations of linear ordinary differential equations (LODEs)
  - See how this can be used to approximate solutions to boundary-value problems (BVPs)
  - Observe that this defines a system of linear equations
  - Look at examples with both constant coefficients and with variable coefficients
  - Describe implementations in MATLAB

2

2

Linear ordinary differential equations

- In this lecture, we will focus on a technique appropriate for linear ordinary differential equations (LODEs)
  - The most general form is a linear combination of $u(x)$ and its derivatives:
    $$a_2(x)u^{(2)}(x) + a_1(x)u^{(1)}(x) + a_0(x)u(x) = g(x)$$
  - The coefficients can be functions of x
- In your calculus course, you focused on solutions to LODEs with constant coefficients:
  $$a_2u^{(2)}(x) + a_1u^{(1)}(x) + a_0u(x) = g(x)$$
  - These approximation techniques will, however, generalize

Approximating the derivative

- Previously, we saw two approximations:
  $$u^{(1)}(x) \approx \frac{-u(x-h) + u(x+h)}{2h}$$
  $$u^{(2)}(x) \approx \frac{u(x-h) - 2u(x) + u(x+h)}{h^2}$$
  - How about substituting these two approximations into the LODE?
    $$a_2(x)u^{(2)}(x) + a_1(x)u^{(1)}(x) + a_0(x)u(x) = g(x)$$

A finite-difference method

- Thus, we go from

$$a_2(x)u^{(2)}(x) + a_1(x)u^{(1)}(x) + a_0(x)u(x) = g(x)$$

to

$$a_2(x)\left(\frac{u(x-h) - 2u(x) + u(x+h)}{h^2}\right) + a_1(x)\left(\frac{-u(x-h) + u(x+h)}{2h}\right)$$
$$+ a_0(x)u(x) \approx g(x)$$

5

5

A finite-difference method

- Let's expand this and collect on $u(x-h)$, $u(x)$ and $u(x+h)$:

$$a_2(x)\left(\frac{u(x-h) - 2u(x) + u(x+h)}{h^2}\right) + a_1(x)\left(\frac{-u(x-h) + u(x+h)}{2h}\right)$$
$$+ a_0(x)u(x) \approx g(x)$$

$$u(x-h)\left(\frac{a_2(x)}{h^2} - \frac{a_1(x)}{2h}\right) + u(x)\left(-\frac{2a_2(x)}{h^2} + a_0(x)\right) + u(x+h)\left(\frac{a_2(x)}{h^2} + \frac{a_1(x)}{2h}\right) \approx g(x)$$

6

6

- Finally, multiply by $2h^2$:

$$u(x-h)\left(\frac{a_2(x)}{h^2}-\frac{a_1(x)}{2h}\right)+u(x)\left(-\frac{2a_2(x)}{h^2}+a_0(x)\right)+u(x+h)\left(\frac{a_2(x)}{h^2}+\frac{a_1(x)}{2h}\right)\approx g(x)$$

$$u(x-h)\big(2a_2(x)-a_1(x)h\big)+u(x)\big(-4a_2(x)+2h^2a_0(x)\big)+u(x+h)\big(2a_2(x)+a_1(x)h\big)\approx 2g(x)h^2$$

7

7

- Therefore, if $u(x)$ satisfies this LODE,

$$a_2(x)u^{(2)}(x)+a_1(x)u^{(1)}(x)+a_0(x)u(x)=g(x)$$

then it must also be true that

$$u(x-h)\big(2a_2(x)-a_1(x)h\big)+u(x)\big(-4a_2(x)+2h^2a_0(x)\big)+u(x+h)\big(2a_2(x)+a_1(x)h\big)\approx 2g(x)h^2$$

8

8

A finite-difference method

Visualization

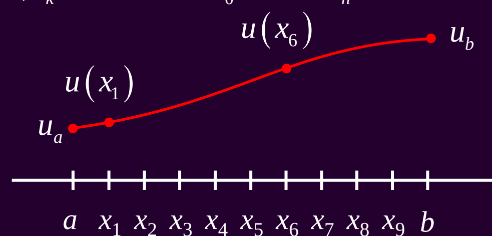
- Let's look at the problem visually:

$$a_2(x)u^{(2)}(x) + a_1(x)u^{(1)}(x) + a_0(x)u(x) = g(x)$$

$$u(a) = u_a$$

$$u(b) = u_b$$

  – Break the interval $[a, b]$ into n sub-intervals
    - Each is of width $h = \frac{b-a}{n}$
    - Thus, $x_k = a + kh$ with $x_0 = a$ and $x_n = b$



9

9

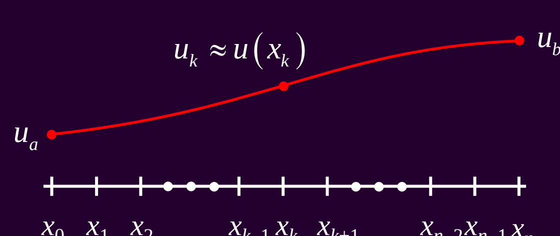
A finite-difference method

Visualization

- Let's focus on a single point x_k:
  – We don't know the value of $u(x_k)$,
    but the following equation should hold approximately true:

$$u(x_k - h)\left(2a_2(x_k) - a_1(x_k)h\right) + u(x_k)\left(-4a_2(x_k) + 2h^2 a_0(x_k)\right) + u(x_k + h)\left(2a_2(x_k) + a_1(x_k)h\right) \approx 2g(x_k)h^2$$

  – Represent our estimate of $u(x_k)$ with u_k so $u_k \approx u(x_k)$



10

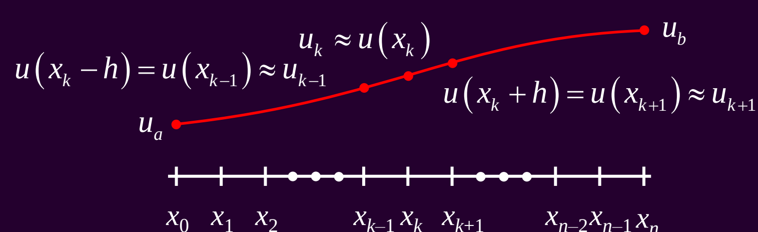
10

Visualization

- Note that $x_k - h = x_{k-1}$ and $x_k + h = x_{k+1}$,

  $u(x_k - h) = u(x_{k-1}) \approx u_{k-1}$

  $u(x_k + h) = u(x_{k+1}) \approx u_{k+1}$



Visualization

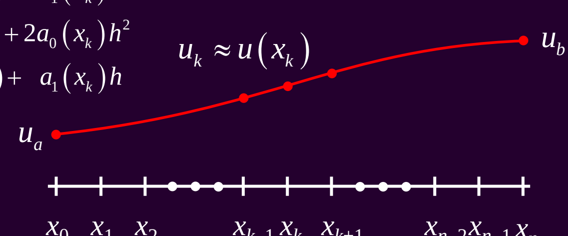
- This looks ugly, but all four functions a_2, a_1, a_0 and g as well as h are all known

$$u_{k-1}\left(2a_2(x_k) - a_1(x_k)h\right) + u_k\left(-4a_2(x_k) + 2h^2 a_0(x_k)\right) + u_{k+1}\left(2a_2(x_k) + a_1(x_k)h\right) \approx 2g(x_k)h^2$$

  – Therefore, this is a linear equation in three unknowns

$$p_k u_{k-1} + q_k u_k + r_k u_{k+1} \approx 2g(x_k)h^2$$

$$p_k = 2a_2(x_k) - a_1(x_k)h$$
$$q_k = -4a_2(x_k) + 2a_0(x_k)h^2$$
$$r_k = 2a_2(x_k) + a_1(x_k)h$$



Visualization

- There are unknowns from $k = 1, 2, \ldots, n - 1$

$$p_1 u_0 + q_1 u_1 + r_1 u_2 = 2g(x_1)h^2$$
$$p_2 u_1 + q_2 u_2 + r_2 u_3 = 2g(x_2)h^2$$
$$p_3 u_2 + q_3 u_3 + r_3 u_4 = 2g(x_2)h^2$$
$$p_4 u_3 + q_4 u_4 + r_4 u_5 = 2g(x_2)h^2$$

$$p_{n-2} u_{n-3} + q_{n-2} u_{n-2} + r_{n-2} u_{n-1} = 2g(x_{n-2})h^2$$
$$p_{n-1} u_{n-2} + q_{n-1} u_{n-1} + r_{n-1} u_n = 2g(x_{n-1})h^2$$

  – This gives $n - 1$ equations in the $n + 1$ unknowns $u_0, u_1, \ldots, u_{n-1}, u_n$

A finite-difference method

Visualization

- Fortunately, we have two boundary values, so:

$$u_0 = u_a$$
$$u_n = u_b$$

  – Thus, Equations for $k = 1$ and $k = n - 1$ may be slightly modified:

$$p_1 u_a + q_1 u_1 + r_1 u_2 = 2g(x_1)h^2$$
$$q_1 u_1 + r_1 u_2 = 2g(x_1)h^2 - p_1 u_a$$

$$p_{n-1} u_{n-2} + q_{n-1} u_{n-1} + r_{n-1} u_b = 2g(x_{n-1})h^2$$
$$p_{n-1} u_{n-2} + q_{n-1} u_{n-1} = 2g(x_{n-1})h^2 - r_{n-1} u_b$$

- Thus, we have a system of $n-1$ linear equations in $n-1$ unknowns
  - This is a tri-diagonal matrix
    - It can be solved in O(n) time, and not O(n^3) time

$$\begin{pmatrix} q_1 & r_1 & & & & & \\ p_2 & q_2 & r_2 & & & & \\ & p_3 & q_3 & r_3 & & & \\ & & p_4 & q_4 & r_4 & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & p_{n-2} & q_{n-2} & r_{n-2} \\ & & & & & p_{n-1} & q_{n-1} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{pmatrix} = \begin{pmatrix} 2g(x_1)h^2 - p_1 u_a \\ 2g(x_2)h^2 \\ 2g(x_3)h^2 \\ 2g(x_4)h^2 \\ \vdots \\ 2g(x_{n-2})h^2 \\ 2g(x_{n-1})h^2 - r_{n-1} u_b \end{pmatrix}$$

- Suppose we have a LODE with constant coefficients:

$$u_{k-1}(2a_2 - a_1 h) + u_k\left(-4a_2 + 2h^2 a_0\right) + u_{k+1}(2a_2 + a_1 h) \approx 2g(x_k)h^2$$

  - Now the matrix entries are identical:

$$\begin{aligned} p &= 2a_2 - a_1 h \\ q &= -4a_2 + 2a_0 h^2 \\ r &= 2a_2 + a_1 h \end{aligned}$$

- Our matrix is now greatly simplified:
  - All entries on the diagonal, the super-diagonal and the sub-diagonal are the same

$$\begin{pmatrix} q & r & & & & & \\ p & q & r & & & & \\ & p & q & r & & & \\ & & p & q & r & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & p & q & r \\ & & & & & p & q \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{pmatrix} = \begin{pmatrix} 2g(x_1)h^2 - pu_a \\ 2g(x_2)h^2 \\ 2g(x_3)h^2 \\ 2g(x_4)h^2 \\ \vdots \\ 2g(x_{n-2})h^2 \\ 2g(x_{n-1})h^2 - ru_b \end{pmatrix}$$

Examples

- We will first look at both an implementation of, and an example of a BVP with constant coefficients
  - That is, a_2, a_1 and a_0 fixed real values

- Next, we will look at the implementation of, and an example of a BVP with non-constant coefficients
  - That is, a_2, a_1 and a_0 are functions of x

Constant coefficient example

```
function [xs, us] = bvpcc( ode, g, x_rng, u_bndry, n )
    h = (x_rng(2) - x_rng(1))/n;

    p =  2.0*ode(1) -     ode(2)*h;
    q = -4.0*ode(1) + 2.0*ode(3)*h^2;
    r =  2.0*ode(1) +     ode(2)*h;

    A = diag( q*ones( n - 1, 1 ) )      ...
      + diag( r*ones( n - 2, 1 ),  1 )  ...
      + diag( p*ones( n - 2, 1 ), -1 );

    xs = linspace( x_rng(1) + h, x_rng(2) - h, n - 1 )';
    v = 2.0*g( xs )*h^2;

    v(1)   = v(1)   - p*u_bndry(1);
    v(end) = v(end) - r*u_bndry(2);

    us = A \ v;
    xs = [x_rng(1);   xs; x_rng(2)];
    us = [u_bndry(1); us; u_bndry(2)];
end
```


Constant coefficient example

- Suppose we have the following BVP:

$$u^{(2)}(x) + 3u^{(1)}(x) + 2u(x) = \sin(x)$$
$$u(-1) = 1$$
$$u(1) = 2$$

- If n = 10, then h = 0.2, so

$$\begin{aligned} p &= 2a_2 - a_1 h && = 2\cdot 1 - 3\cdot 0.2 && = 1.4 \\ q &= -4a_2 + 2a_0 h^2 && = -4\cdot 1 + 2\cdot 2\cdot 0.04 && = -3.84 \\ r &= 2a_2 + a_1 h && = 2\cdot 1 - 3\cdot 0.2 && = 2.6 \end{aligned}$$

- Also, the x-values are –1, –0.8, –0.6, –0.4, –0.2, 0, 0.2, 0.4, 0.6, 0.8, 1

Constant coefficient example

- Thus, we have our system of linear equations

$$\begin{pmatrix} -3.84 & 2.6 & & & & & & & \\ 1.4 & -3.84 & 2.6 & & & & & & \\ & 1.4 & -3.84 & 2.6 & & & & & \\ & & 1.4 & -3.84 & 2.6 & & & & \\ & & & 1.4 & -3.84 & 2.6 & & & \\ & & & & 1.4 & -3.84 & 2.6 & & \\ & & & & & 1.4 & -3.84 & 2.6 & \\ & & & & & & 1.4 & -3.84 & 2.6 \\ & & & & & & & 1.4 & -3.84 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{pmatrix} = \begin{pmatrix} 2\sin(-0.8)0.04 - 1.4\cdot 1 \\ 2\sin(-0.6)0.04 \\ 2\sin(-0.4)0.04 \\ 2\sin(-0.2)0.04 \\ 2\sin(0)0.04 \\ 2\sin(0.2)0.04 \\ 2\sin(0.4)0.04 \\ 2\sin(0.6)0.04 \\ 2\sin(0.8)0.04 - 2.6\cdot 2 \end{pmatrix}$$

A finite-difference method

Constant coefficient example

- Solving this system of linear equations yields:

$$\mathbf{u} = \begin{pmatrix} 3.265385521974507 \\ 4.262189198888517 \\ 4.519267459652307 \\ 4.367603345860092 \\ 4.011049560817661 \\ 3.572225242052188 \\ 3.122218881076646 \\ 2.699753319940751 \\ 2.323506812667992 \end{pmatrix} \begin{matrix} \approx u(-0.8) \\ \\ \\ \\ \\ \\ \approx u(0.4) \\ \\ \\ \end{matrix}$$

A finite-difference method

Constant coefficient example

- In MATLAB, we can find this by calling:

```
>> [xs, us] = bvpcc( [1 3 2], @sin, [-1 1], [1 2], 10 );
>> u = @(x)( -0.32523483298690115734*exp( 2.0    - 2.0*x) ...
            - 0.19505820123410984121*exp(-1.0*x - 1.0)   ...
            + 0.32523483298690115734*exp(-1.0*x + 3.0)   ...
            + 0.19505820123410984121*exp(-2.0*x)         ...
            - 0.3*cos(x) + 0.1*sin(x);
```

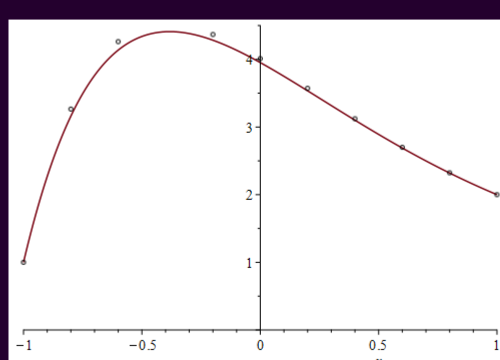
23

23

A finite-difference method

Constant coefficient example

- Here is a plot of the solution and the approximations:



24

24

General example

```
function [xs, us] = bvp( a2, a1, a0, g, x_rng, u_bndry, n )
    h = (x_rng(2) - x_rng(1))/n;

    p = @(x)( 2.0*a2(x) -     a1(x)*h  );
    q = @(x)(-4.0*a2(x) + 2.0*a0(x)*h^2);
    r = @(x)( 2.0*a2(x) +     a1(x)*h  );

    xs = linspace( x_rng(1) + h, x_rng(2) - h, n - 1 )';

    A = zeros( n - 1, n - 1 );

    for k = 1:(n - 1)
        A(k, k) = q(xs(k));
    end

    for k = 1:(n - 2)
        A(k + 1, k    ) = p(xs(k + 1));
        A(k,     k + 1) = r(xs(k));
    end
```


General example

```
    v = 2.0*g( xs )*h^2;

    v(1)   = v(1)   - p(xs(1)  )*u_bndry(1);
    v(end) = v(end) - r(xs(end))*u_bndry(2);

    us = A \ v;
    xs = [x_rng(1);   xs; x_rng(2)];
    us = [u_bndry(1); us; u_bndry(2)];
end
```

A finite-difference method

General example

- Suppose we have the following BVP:

$$13x^2u^{(2)}(x) - 5u^{(1)}(x) + 8xu(x) = \sin(x)$$
$$u(-1) = 1$$
$$u(1) = 2$$

- If n = 10, then h = 0.2, so

$$\begin{aligned} p_k &= 2a_2(x_k) - a_1(x_k)h &&= 2\cdot 13x_k^2 - (-5)\cdot 0.2 \\ q_k &= -4a_2(x_k) + 2a_0(x_k)h^2 &&= -4\cdot 13x_k^2 + 2\cdot 8x\cdot 0.04 \\ r_k &= 2a_2(x_k) + a_1(x_k)h &&= 2\cdot 13x_k^2 + (-5)\cdot 0.2 \end{aligned}$$

- As before,
  the x-values are –1, –0.8, –0.6, –0.4, –0.2, 0, 0.2, 0.4, 0.6, 0.8, 1

27

27

A finite-difference method

General example

- Thus, we have our system of linear equations

$$\begin{pmatrix} -33.8 & 15.6 & & & & & & & \\ 10.4 & -19.1 & 8.4 & & & & & & \\ & 5.2 & -8.6 & 3.2 & & & & & \\ & & 2.0 & -2.2 & 0.0 & & & & \\ & & & 1.0 & 0.0 & -1.0 & & & \\ & & & & 2.0 & -2.0 & 0.0 & & \\ & & & & & 5.2 & -8.1 & 3.2 & \\ & & & & & & 10.4 & -18.3 & 8.4 \\ & & & & & & & 17.6 & -32.8 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{pmatrix} = \begin{pmatrix} 2\sin(-0.8)0.04 - 17.6\cdot 1 \\ 2\sin(-0.6)0.04 \\ 2\sin(-0.4)0.04 \\ 2\sin(-0.2)0.04 \\ 2\sin(0)0.04 \\ 2\sin(0.2)0.04 \\ 2\sin(0.4)0.04 \\ 2\sin(0.6)0.04 \\ 2\sin(0.8)0.04 - 15.6\cdot 2 \end{pmatrix}$$

28

28

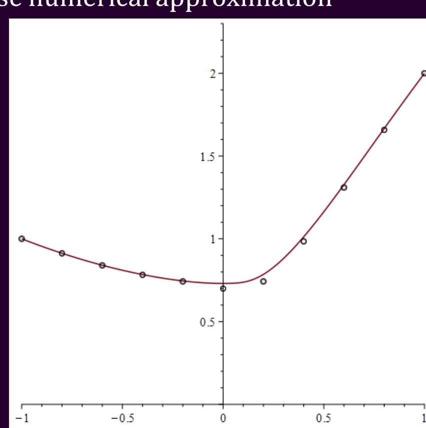
General example

- Solving this system of linear equations yields:

$$\mathbf{u} = \begin{pmatrix} 0.912226606004807 \\ 0.839422824989929 \\ 0.782355814895478 \\ 0.742692476016105 \\ 0.699139454829521 \\ 0.742692476016104 \\ 0.984619294870424 \\ 1.309756419296422 \\ 1.657919761630765 \end{pmatrix} \begin{matrix} \\ \\ \\ \approx u(-0.2) \\ \\ \\ \\ \\ \approx u(0.8) \end{matrix}$$

General example

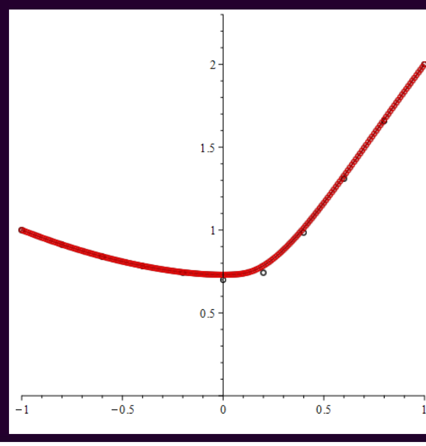
- Here is a plot of the solution and the approximations:
  - In this case, there is no exact solution, so the *solution* is actually a very precise numerical approximation



General example

- We can also use more points:

```
>> [xs, us] = bvp( @(x)(x^2*13), @(x)(-5.0), @(x)(8*x),...
                   @sin, [-1 1], [1 2], 200 )
```



31

31

Error analysis

- Beyond the scope of this course,
  even though we are using two O(h^2) approximations of the derivative and second derivative
  - The overall error of this method is still O(h^2) as we are solving these simultaneously
  - Thus, doubling the number of intervals reduces the error by four

32

32

Summary

- Following this topic, you now
  - Understand the idea finite difference approximations
    - For a LODE, we substitute approximations of the derivative and second derivatives
  - Know that this defines a system of linear equations
  - Understand that this solution gives the approximations at the equally spaced points between *a* and *b*
  - Are aware that if the LODE has constant coefficients, all entries on the diagonal, super-diagonal and sub-diagonal are equal, respectively
  - Have seen implementations in MATLAB

References

[1] https://en.wikipedia.org/wiki/Finite_difference_method

Acknowledgments

None so far.

35

35

Colophon

These slides were prepared using the Cambria typeface. Mathematical equations use Times New Roman, and source code is presented using `Consolas`.
Mathematical equations are prepared in MathType by Design Science, Inc.
Examples may be formulated and checked using Maple by Maplesoft, Inc.

The photographs of flowers and a monarch butter appearing on the title slide and accenting the top of each other slide were taken at the Royal Botanical Gardens in October of 2017 by Douglas Wilhelm Harder. Please see

https://www.rbg.ca/

for more information.

36

36

Disclaimer

These slides are provided for the ECE 204 *Numerical methods* course taught at the University of Waterloo. The material in it reflects the author's best judgment in light of the information available to them at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. The authors accept no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.