Chandler Justice - A02313187

Computational Convergence Study

As $h \to 0$ we want $||E^h|| \to 0$

$$\underline{Uu}: \begin{cases} u'' = f(x) & \text{on } (0,1) \Rightarrow (ku')' = f(x_j) + \epsilon \\ u(0) = \alpha \\ u(1) = \beta \end{cases}$$

For $j = 1, 2, ..., m$

$$\frac{1}{h^2}(U_{j-1} - 2U_j + U_{j+1} = f(x_j)$$

We know if everything goes okay, that this process of discretization will generate an $O(h^2)$ approximation to the exact result.This is a mathematical result that leads to convergence, but there are errors occur in any program we might write/implement.

- Roundoff errors (you cant do *shit* about that)
- measurement error
- this will work as $h \to 0$
- what if the problem is more complicated?

Computational convergence analysis

- chose a decreasing sequence of h that makes

$$\{h_0, \frac{h_0}{2}, \frac{h_0}{2^2}, \frac{h_0}{2^3}\}$$

- for each $h \leq h_0$, we can compute an approximation of U

$$U = \begin{bmatrix} U_1 \\ U_2 \\ ... \\ U_m \end{bmatrix}$$

- evaluate $u(x)$ of parts

$$\hat{U} = \begin{bmatrix} u(x_j 1) \\ u(x_j 2) \\ u(x_j 3) \\ ... \\ u(x_j m) \end{bmatrix}$$

- Compute $||E_h|| = ||U - \hat{U}||$
- this gives us the data we need to determine convergence

$$||E^h|| \leq Ch^2$$
$$\log ||E^h|| \leq \log C + p \log h$$

We can fit the error to ?????

h	E^h	$\log h$	$\log \|\|E_h\|\|$
h_0	$\|\|E^{h_0}\|\|$	$\log(h_0)$	$\log \|\|E^{h_0}$
h_1	$\|\|E^{h_1}\|\|$	$\log(h_1)$	$\log \|\|E^{h_1}$
...			
h_n	$\|\|E^{h_n}\|\|$	$\log(h_n)$	$\log \|\|E^{h_n}$

$$(x_i, y_i), i = 0, 1, 2, ..., n \Rightarrow \text{ fit to a function}$$

$$y(x) = a + px$$
$$y(x_0) = a + px_0$$
$$y(x_1) = a + px_1$$
$$...y(x_n) = a + px_n$$

Which can be represented as

$$\begin{bmatrix} 1 & x_0 \\ 1 & x_1 \\ ... & \\ 1 & x_n \end{bmatrix} \begin{bmatrix} a \\ p \end{bmatrix} = \begin{bmatrix} y(x_0) \\ y(x_1) \\ ... \\ y(x_m) \end{bmatrix} \Rightarrow X \begin{bmatrix} a \\ p \end{bmatrix} = Y$$

We can project this to the column space of X using

$$X^{-1} X \begin{bmatrix} a \\ p \end{bmatrix} = X^{-1} Y$$

Which gives us the result

$$\begin{bmatrix} \sum_{k=0}^{n} 1 & \sum_{k=0}^{n} x_n \\ \sum_{k=0}^{n} h & \sum_{k=0}^{n} x_h^2 \end{bmatrix} \begin{bmatrix} a \\ p \end{bmatrix} = \begin{bmatrix} \sum_{k=0}^{n} y_k \\ \sum_{k=0}^{n} y_k x_k \end{bmatrix}$$

We can write the following code to achieve this

```
a[1][1] = n + 1
a_12 = 0.0
for i in range(n + 1):
    a_12 = a_12 + x[i]
a_21 = a_12
a22 = 0.0
for i in range(nn):
    a[2][2] = a[2][2] + x[i] * x[i]

b[1] = 0
b[2] = 0
for i in range (nn):
    b[1] = b[1] + y[i] * x[i]
    b[2] = b[2] + y[i] * x[i]
```

<u>matrix inversion</u>

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} a \\ p \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \Rightarrow \begin{bmatrix} a \\ p \end{bmatrix} = \frac{1}{a_{11} a_{21} - a_{21} a_{12}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\det A = a_{11}a_{22} - a_{12}a21$$
$$a = \frac{1}{\det A}(b_1 a_{22} - b_2 a_{12})$$
$$p = \frac{1}{\det A}(-b_1 a_{21} + b_2 a_{11})$$

what if we don't have u?

h	E^h	$\log h$	$\log \|\|E_h\|\|$
h_0	$\|\|E^{h_0}\|\|$	...	...
h_1	$\|\|E^{h_1}\|\|$	...	...
...			
$h_m = \frac{h^0}{2^m}$	$\|\|E^{h_n}\|\|$	$\log(h_n)$	$\log \|\|E^{h_n}$

We can use the h_m to approximate u. We can then do the following to get the other needed approximations

$$||U - U_k + U_k - \hat{U}|| \leq ||U - U_h|| + ||U_h - \hat{U}||$$

February 14, 2024

Errors:

$$||E^h||_\infty = \max_{1\leq i\leq m} |U_i - \hat{U}_i| \approx ||e(x)|| = \max_{a\leq x\leq b} |u(x) - \hat{u}(x)|$$
$$||E^h||_1 = \sum_{j=1}^{m} |U_j - \hat{U}_j| \approx ||e(x)||_1 = \int |u(x) - \hat{u}(x)| dx$$
$$||E^h||_2 = (\sum_{j=1}^{m} |U_j - \hat{U}_j|^2)^{1/2} \approx ||e(x)||_2 = (\int_b^a |u(x) - \hat{u}(x)|^2 dx)^{1/2}$$

PDEs

$$a_1(x,y)u_{xx} + a_2(x,y)u_{xy} + a_3(x,y)u_{yy} + ... + a_6(x,y) = f$$

Def: $\rho = a_2^2(x,y) - 4a_1(x,y) * a_3(x,y)$

$$\text{if}: \begin{cases} \rho < 0 & \rightarrow \text{ elliptic equation} \\ \rho = 0 & \rightarrow \text{ parabolic equation} \\ \rho > 0 & \rightarrow \text{ hyperbolic equation} \end{cases}$$

Ex: Elliptic equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f \qquad \text{poisson equation}$$
$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \qquad \text{laplace equation}$$
$$\Rightarrow a_1 = 1, a_2 = 0, a_3 = 1$$
$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \qquad \text{heat equation}$$
$$\Rightarrow a_1 = 1, a_2 = 0a_3 = 0$$

Notation:

$$
\begin{aligned}
\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial u^2}{\partial y^2} &= \Delta\Delta u \\
&= (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}) \\
&= \nabla^2 u
\end{aligned}
$$

Mesh: In 1-d we have

$$x_j = a + j * h = a + jx\Delta x$$

in 2-d we have

$$
\begin{array}{ccccc}
. & . & . & . & \Delta y \\
. & . & . & . & \Delta y \\
. & . & . & . & \Delta y \\
. & . & . & . & \Delta y \\
\Delta x & \Delta x & \Delta x & \Delta x &
\end{array}
$$

<u>probabilities</u>

$$\Delta u = f(x, y) \qquad \text{on } \Omega \in \mathbb{R}^m$$

$$u\Big|_{\partial\Omega} = g(x, y)$$

February 16, 2024

<u>We still talking about heat equations (Potential equation)</u>

$$\Delta u = f \text{ on } \Omega$$

$$u\Big|_{\partial\Omega} = g \qquad (\partial \text{ is the boundary})$$

Solutions are called <u>harmonic functions</u>. Building a linear system alike to the 1D problem, where are are finding the unknowns between two bounds. When we *step it up* to 2D we are still finding the unknowns at specific points, we just add an axis.

$$
\begin{array}{cccc|c}
. & . & . & . & 4 \\
. & 7 & 8 & 9 & 3 \\
. & 4 & 5 & 6 & 2 \\
. & 1 & 2 & 3 & 1 \\
\hline
0 & 1 & 2 & 3 & 4/0
\end{array}
\quad (i, j) \to \text{ind} \quad \Rightarrow \text{ind} = i + (j-1) * m
$$

Where ind is a conversion from 2d to 1d indices.

We can use the following variables to convert from 1D to 2D indices

$$\text{ind} \to (i, j) \Rightarrow \quad j = \frac{\text{ind}}{m}\%1, \quad i = \text{ind} - (j-1) * m$$

<u>spout stencil</u>

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho$$

Which means the equation at (i,j) is

$$\frac{1}{h^2}(u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j+1})) = f(x_i, y_j)$$

When we build a system of equations it will look like

$$A^h = \frac{1}{h^2}\begin{bmatrix} -4 & 1 & 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ 1 & -4 & 1 & 0 & \dots & 0 & 0 & 1 & \dots & 0 \\ 0 & 1 & -4 & 1 & \dots & 0 & 0 & 0 & 1 & \dots \\ 1 & 0 & 0 & -4 & 1 & \dots & 0 & 0 & \dots & 1 \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots & & & & & & & & & -4 \end{bmatrix} \Rightarrow \frac{1}{h^2}\begin{bmatrix} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{bmatrix}\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \\ U_7 \\ U_8 \\ U_9 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \end{bmatrix}$$

We then can worry about stability and truncation error to determine convergence.

$$\text{LTE: } \tau_{i,j}(h) = ? = O(h^2)$$
$$\text{stability: } \lambda_{1,1} = -2\pi + O(h^2)$$

$$\therefore \text{contains number } \rightarrow K_1(A) = (\frac{8}{h^2}\frac{1}{2\epsilon}) = O\left(\frac{1}{h^2}\right)$$

Each row represents the approximate equation at a point. We can then take the top 4x4 subset of this matrix to solve our system

$$\Rightarrow \begin{bmatrix} T & I & 0 & 0 \\ I & T & I & 0 \\ 0 & I & T & I \\ 0 & 0 & I & T \end{bmatrix}$$

February 21, 2024

Relevant topics for this class

Direct methods of determining finite differences

- Gaussian Elimination & back substitution
- LU factorization

Iterative Methods

- Jacobi Iteration
- Gauss-sidel elimination
- SOR
- Gradient methods
- Conjugate-gradients

Preconditioning

Approximations

1. LTE $\rightarrow 0 \quad ah \rightarrow 0$
2. stability

Local Truncation Error

$$\Delta u = u_{xx} + u_{yy} = f \Rightarrow \frac{U_{i+i,j} - 2U_{i,j}U_{i-1,j}}{h^2} + \frac{U_{i,j+1} - 2U_{i,j}U_{i,j-1}}{h^2} = f_{i,j}$$

$$-\tau_{i,j} = \frac{U_{i+i,j} - 2U_{i,j}U_{i-1,j}}{h^2} + \frac{U_{i,j+1} - 2U_{i,j}U_{i,j-1}}{h^2} = f_{i,j}$$

$$\frac{1}{h^2}u_{i+1,j} = u_{i,j} + h(u_x)_{i,j} + \frac{1}{2}h^2(u_{xx})_{i,j} + \frac{1}{6}h^3(u_{xxx})_{i,j} + \frac{1}{24}h^4(u_{xxxx})_{i,j} + ...$$

$$\frac{1}{h^2}u_{i-1,j} = u_{i,j} - h(u_x)_{i,j} + \frac{1}{2}h^2(u_{xx})_{i,j} - \frac{1}{6}h^3(u_{xxx})_{i,j} + \frac{1}{24}h^4(u_{xxxx})_{i,j} + O(h^4)$$

$$-\frac{4}{h^2}u_{i,j} = u_{i,j}$$

$$\frac{1}{h^2}u_{i,j+1} = u_{i,j} + h(u_y)_{i,j} + \frac{1}{2}h^2(u_{yy})_{i,j} + \frac{1}{6}h^3(u_{yyy})_{i,j} + \frac{1}{24}h^4(u_{yyyy})_{i,j} + O(h^4)$$

$$\frac{1}{h^2}u_{i,j-1} = u_{i,j} - h(u_y)_{i,j} + \frac{1}{2}h^2(u_{yy})_{i,j} - \frac{1}{6}h^3(u_{yyy})_{i,j} + \frac{1}{24}h^4(u_{yyyy})_{i,j} + O(h^4)$$

$$\Rightarrow \frac{h^2(u_{xx})_{i,j} + h^2(u_{yy})_{i,j}}{h^2}$$

all this bullshit gives us

$$\tau_{i,j} = (u_{xx})_{i,j} + (u_{yy})_{i,j} - f_{i,j} + O(h^3)$$

For the problem we are trying to solve

$$A^h E^h = -\tau^h$$

We can now determine the stability using

$$\|A^h\|_2 \leq C$$

and

$$\|A^h E^h\| \leq \|A_h\| * \|E^h\| \Rightarrow \|A^h E^h\| \leq \|\tau^h\|$$

$$\Rightarrow \|(A^h)^{-1}\tau^h\| \leq \|(A^h)^{-1}\|_2 * \|\tau^h\| \leq C * O(h^2)$$

Then our error presents inself as

$$\lambda_{1,1} = -2\pi^2 + O(h^2) \Rightarrow p((A^h)^{-1}) = \frac{1}{\lambda_{1,1}} = -\frac{1}{2\pi^2}$$

$$\kappa(A) = \|A\|_2 * \|A^{-1}\|_2 = \lambda_{m,m} * \lambda_{1,1}$$

Iterative methods

Jacobi Method

$$\frac{1}{h^2}(U_{i+1,j} - 2U_{i,j} + U_{i,1,j}) + \frac{1}{h^2}(U_{i,j+1} - 2I_{i,j} + U_{i,j+1})$$

In code...

```
4 * U[i][j] = U[i+1][j] + U[i][j-1] + U[i][j+1] - f[i][j] * h**2
```
