

Computational Convergence Study

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

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

For $j = 1, 2, \dots, m$

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

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 $h \rightarrow 0$
- 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

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ u_m \end{bmatrix}$$

- evaluate $u(x)$ of parts

$$\hat{u} = \begin{bmatrix} u(x_j1) \\ u(x_j2) \\ u(x_j3) \\ \dots \\ u(x_jm) \end{bmatrix}$$

- 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}\ $
\dots			
h_n	$\ E^{h_n}\ $	$\log(h_n)$	$\log \ E^{h_n}\ $

$(x_i, y_i), i = 0, 1, 2, \dots, n \Rightarrow$ fit to a function

$$\begin{aligned}
 y(x) &= a + px \\
 y(x_0) &= a + px_0 \\
 y(x_1) &= a + px_1 \\
 \dots y(x_n) &= a + px_n
 \end{aligned}$$

Which can be represented as

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

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

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

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]
```

matrix inversion

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

$$\det A = a_{11}a_{22} - a_{12}a_{21}$$

$$a = \frac{1}{\det A} (b_1 a_{22} - b_2 a_{12})$$

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

what if we don't have u ?

h	E^h	$\log h$	$\log \ E_h\ $
h_0	$\ E^{h_0}\ $	\dots	\dots
h_1	$\ E^{h_1}\ $	\dots	\dots
\dots			
$h_m = \frac{h_0}{2^m}$	$\ E^{h_m}\ $	$\log(h_m)$	$\log \ E^{h_m}\ $

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:

$$\|E^h\|_\infty = \max_{1 \leq i \leq m} |u_i - \hat{u}_i| \approx \|e(x)\| = \max_{a \leq x \leq b} |u(x) - \hat{u}(x)|$$

$$\|E^h\|_1 = \sum_{j=1}^m |u_j - \hat{u}_j| \approx \|e(x)\|_1 = \int |u(x) - \hat{u}(x)| dx$$

$$\|E^h\|_2 = \left(\sum_{j=1}^m |u_j - \hat{u}_j|^2 \right)^{1/2} \approx \|e(x)\|_2 = \left(\int_b^a |u(x) - \hat{u}(x)|^2 dx \right)^{1/2}$$

PDEs

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

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

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

Ex: Elliptic equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f \quad \text{poisson equation}$$

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \quad \text{laplace equation}$$

$$\Rightarrow a_1 = 1, a_2 = 0, a_3 = 1$$

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad \text{heat equation}$$

$$\Rightarrow a_1 = 1, a_2 = 0, a_3 = 0$$

Notation:

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

Mesh: In 1-d we have

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

in 2-d we have

$$\begin{array}{ccccccc} & & & & & \Delta y & \\ & & & & & \Delta y & \\ & & & & & \Delta y & \\ & & & & & \Delta y & \\ \Delta x & \Delta x & \Delta x & \Delta x & & & \end{array}$$

probabilities

$$\begin{aligned}\Delta u &= f(x, y) && \text{on } \Omega \in \mathbb{R}^m \\ u|_{\partial\Omega} &= g(x, y)\end{aligned}$$

February 16, 2024

We still talking about heat equations (Potential equation)

$$\begin{aligned}\Delta u &= f \text{ on } \Omega \\ u|_{\partial\Omega} &= g && (\partial \text{ is the boundary})\end{aligned}$$

Solutions are called harmonic functions. 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.

$$\begin{array}{cccccc} \cdot & \cdot & \cdot & \cdot & 4 & \\ \cdot & 7 & 8 & 9 & 3 & \\ \cdot & 4 & 5 & 6 & 2 & \\ \cdot & 1 & 2 & 3 & 1 & \\ 0 & 1 & 2 & 3 & 4/0 & \end{array} \quad (i, j) \rightarrow \text{ind} \quad \Rightarrow \text{ind} = i + (j - 1) * m$$

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

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

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

spout stencil

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

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

$$A^h = \frac{1}{h^2} \begin{bmatrix} -4 & 1 & 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ 1 & -4 & 1 & 0 & \dots & 0 & 0 & 1 & \dots & 0 \\ 0 & 1 & -4 & 1 & \dots & 0 & 0 & 0 & 1 & \dots \\ 1 & 0 & 0 & -4 & 1 & \dots & 0 & 0 & \dots & 1 \\ \dots & & & & & & & & & \\ \dots & & & & & & & & & \\ \dots & & & & & & & & & \\ \dots & & & & & & & & & \\ \dots & & & & & & & & & -4 \end{bmatrix} \Rightarrow \frac{1}{h^2} \begin{bmatrix} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \end{bmatrix}$$

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

$$\text{LTE: } \tau_{i,j}(h) = ? = O(h^2)$$

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

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

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

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