Chandler Justice - A02313187

January 22, 2024

Homework:
For question 4: determine

$$D_n h(y) = c_1 u(x_1) + c_2 u(x_2) + ... + c_n u(x_n)$$

What we will want to do is take the vanderbaun matrix and multiply it by our constants to get a vector of all 0s and a 1
Example:

$$u''(x) = \frac{u(x-h) - 2u(x) + u(x+h)}{h^2} = err$$

$$|E| = |u''(x) - \frac{1}{h^2}(u(x+h) - 2u(x) + u(x-h)|$$

$$= |u''(x) - \frac{1}{h^2}\{(u(x) + hu'(x) + \frac{1}{2}h^2u''(x) + \frac{h^3}{6}u'''(x) + \frac{h^4}{24}u''''(\zeta_1))$$

$$-2u(u(x) - hu(x) + \frac{1}{2}h^2u''(x) - \frac{1}{6}h^3(x) + \frac{h^4}{24}u''''(\zeta_2))\}$$

$$= |u'' - \frac{1}{h^2}\{h^2u''(x) + \frac{1}{24}h^4u''''(\zeta_3)\}|$$

Heat Equation: Heat equation:

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial k}k(x)\frac{\partial u}{\partial x} + t(x,t)$$

If we assume k is continuous

$$\frac{\partial u}{\partial t} = K\frac{\partial^2 u}{\partial x^2} + t(x,t)$$

steady state = 1 $\Rightarrow \frac{\partial u}{\partial t} = 0 \Rightarrow Ku'' + t(x,t) = 0$

$$\begin{cases} u'' = f(x) \\ u(0) = \alpha \\ u(1) = \beta \end{cases}$$

Exact solution:

$$\int u''(x)dx = \int f(x)dx = g(x) + c$$

$$u' = g(x) + c_1$$

$$u = \int g(x) + c_1x + c_2$$

We could decide to get an approximation at discrete points in the domain. Lets our domain be $[0,1]$.

So we will use equally spaced (for now) points in $[0,1]$, say $m+2$ points. Then

$$h = \frac{1}{m+1} \Rightarrow \{u_0, u_1, ..., u_m, u_{m+1}\} \quad \text{(size = } m+2\text{)}$$

$$x_j = j * h$$

$$u_0 = \alpha$$

$$u_{m+1} = \beta$$

our u_0, u_{m+1} variables will be exact values.
$u'' = f(x)$ at $x_0, x_1, ...x_n$

$$D^2 u_j = \frac{u_{j-1} 2u_j + u_{j+1}}{h^2}$$

so

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

for $j = 0, 1, 2, ..., m+1$

$$\begin{array}{cc}
j = 0 & u_0 = \alpha \\
j = 1 & \frac{1}{h^2}(u_0 - 2u_1 + u_2) = f(x_1) \\
j = 2 & \frac{1}{h^2}(u_1 - 2u_2 + u_3) = f(x_1) \\
... & ... \\
j = m & \frac{1}{h^2}(u_{m-1} - 2u_m + u_{m+1}) = f(x_m)
\end{array}$$

$$\begin{bmatrix} 1 & 0 & ... & 0 & 0 \\ -\frac{1}{h^2} & -\frac{2}{h^2} & ... & 0 & \\ 0 & -\frac{1}{h^2} & -\frac{2}{h^2} & ... & 0 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_n \end{bmatrix} = \begin{bmatrix} \alpha \\ f_1 \\ f_2 \\ f_n \\ \beta \end{bmatrix}$$

Then we get

$$\begin{bmatrix} \frac{\alpha}{n^2} \\ 0 \\ ... \\ 0 \\ \frac{\beta}{n^2} \end{bmatrix} \begin{bmatrix} -2 & 1 & 0 & ... & 0 & 0 \\ 1 & -2 & 1 & 0 & ... & 0 \\ 0 & 1 & -2 & 1 & 0 & ... \\ ... & ... & ... & ... & & \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ ... \\ f_m \end{bmatrix}$$

To summarize we get a matrix A that we multiply by a vector U to get the vector of functions F.

We need to be able to define and interpret an error. Suppose we define

$$\hat{U} = \begin{bmatrix} u(x_1) \\ u(x_2) \\ ...u(x_m) \end{bmatrix}, \quad U = \begin{bmatrix} u_1 \\ u_2 \\ ...u_n \end{bmatrix} \in \mathbb{R}$$

Then

$$E = U - \hat{U}$$

Definition: *Vector norm -* Any function $||*||{:}\mathbb{R}^m \in \mathbb{R}$ is a norm if

1. for any vector $v \in \mathbb{R}^m$, $||v|| \geq 0$ and $||v|| = 0$ iff $x = 0$
2. for any vector $v \in \mathbb{R}^m$ and scaler a, $||av|| = |a| * ||v||$
3. for $u, v \in \mathbb{R}^m$, $||u + v|| \leq ||u|| + ||v||$

The norms:

- $||E||_\infty = max_{1 \leq j \leq m} |u_j - u(xj)|$
- $||E||_1 = h \sum_{j=1}^{m} |u_j - u(x_j)|$
- $||E||_2 = (h \sum_{j=1}^{n} |u - u(x_j)|^2)^{\frac{1}{2}}$

January 24, 2024

Example: 2 point boundary problem

$$\begin{cases} u'' = f(x) & 0 < x < 1 \\ u(0) = \alpha \\ u(1) = \beta \end{cases}$$

Lets employ a centered difference

$$u''(x) \approx \frac{u(x-h) - 2u(x) + u(x+h)}{h^2}$$

If we want the error we can compute

$$|e(x)| = |u''(x) - \frac{u(x-h) - 2u(x) + u(x+h)}{h^2}| \leq O(h^2)$$

We want to approximate u at discrete points. To do this, we need to pick points over an interval where each point is h apart from the next, $h = 1/(m+1)$, $x_j = j * h$, and $j = \{1, 2, 3, 4, ..., m+1\}$.

Definition:

$$u(x_j) = u_j \rightarrow u''(x_j) = \frac{u_{j-1} - 2u_j + u_{j+1}}{h^2} \quad j = 1, 2, ...m$$

$$\rightarrow \frac{u_{j-1} - 2u_j + u_{j+1}}{h^2}$$

Working through these computations we get $AU = F$

$$\begin{bmatrix} -2 & 1 & 0 & ... & 0 \\ 1 & -2 & 1 & ... & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ... & 1 & -2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ ... \\ u_m \end{bmatrix} = \begin{bmatrix} f_1 - \alpha/h^2 \\ f_2 \\ ... \\ f_{m-1} \\ f_{m-1} - Ch^2 \end{bmatrix}$$

In general, we can use the classic $Ax = b$ method from linear algebra to solve these systems of equations. We can do this in code with

```
for k = 1, m = 1
    for i = k + 1, m
        factor = a[i][k]
        for j = k + 1, m
            a[i][j] = a[i][j] - factor * a[i][j]
        end
        b[i] = b[i] - factor * b[j]
    end
end
```

For a tridiagonal matrix we can perform elimination via

```
for k=1, m=1
    factor = a[k + 1][k] / a[k][k]
    a[k + 1][k+1] = a[k+1][k+1] - factor * a[k][k+1]
    b[k+1] = b[k+1] - factor * b[k]
end
```

After running this algorithm on a tridiagonal matrix, it will result in a matrix with all zeros except for along the two center diagonal entries where $a'_{i,i}, a'_{i+1,i}$. From here, we can to our $AU = F$ compuattion, where

$$u_m = (b'_m / a'_m m)$$

$$u_{m-1} = (b'_{m=1} - a_{m-1,m-1} a_m) / a_{m-1,m-1}$$

$$u_k = (b'_k - a'_{k,m+1} * u_{m+1}) / a'_{k,k}$$

Putting all this shit together is referred to as the **Thomas algorithm**.

Lets say we don't want to waste computation time and memory storing a bunch of zeros. We can avoid this by decomposing the nonzero elements into vectors where

$$A = \begin{cases} a_d = \text{main diagonal} \\ a_{l1} = \text{1st ??? diagonal} \\ a_{s1} = \text{1nd super diagonal} \end{cases}$$

We can implement this in code using

```
# forward elimination
for k=1, m=1
    factor = al1[k] / ad[k]
    ad[k+1][k+1] = ad[k+1][k+1] - factor * as1[k]
    b[k+1] = b[k+1] - factor * b[k]
end

# back substitution
u(m) = k[m] / ad[m]
for k = m-1, 1
    u[k] = (b[k] - as1[k] * u[k+1]) / ad[k])
end
```