CS381: Numerical Computation & Softwares

Iterative method for System of Linear Equations

Jakramate Bootkrajang

Department of Computer Science

Chiang Mai University

Outline

Implementing Gauss-Seidel method

Approach

- กำหนดระบบสมการ $Ax = b$ เราต้องการหาค่า x โดย
- เดาสุ่มค่าเริ่มต้นของ x แล้วค่อยๆปรับปรุงค่า x ให้เข้าใกล้คำตอบเรื่อยๆ
- การ implement ไม่ได้ซับซ้อนมาก เนื่องจากมีสูตรในการคำนวณค่าของ x ไว้อยู่แล้ว

Calculating x_i

- การปรับปรุงค่าของ x_i แต่ละตัว

$$x_i = \frac{b_i - \sum_{j=1, j\neq i}^{n} a_{ij}x_j}{a_{ii}} \quad \text{for} \quad i = 1, \ldots, n$$

เริ่มเขียน function GaussSeidel

ฟังก์ชันนี้รับตัวแปรสามตัว คือ เมทริกซ์ A เวคเตอร์ b เวคเตอร์ของคำตอบเริ่มต้น x และ จำนวนรอบในการคำนวณ $iter$

```
function GaussSeidel(A,b,x,iter)


end
```

เริ่มเขียน function GaussSeidel

จากนั้นเราจะเริ่ม loop ในการคำนวณ x_i แต่ละตัว $x_i = \frac{b_i - \sum_{j=1, j\neq i}^{n} a_{ij}x_j}{a_{ii}}$ เราจะใช้การหา dot product ของ A ในแถว i กับเวคเตอร์ x ที่ถูกกำหนดค่าในตำแหน่งที่ i ให้เป็น 0 ในการคำนวณ $\sum_{j=1, j\neq i}^{n} a_{ij}x_j$ อย่างรวดเร็ว

```
function GaussSeidel(A,b,x,iter)
    for i=1:length(b)
        x[i] = 0
        x[i] = ((b[i] - (x*A[i,:]))/A[i,i])[1]
    end
end
```

สุดท้ายเราจะ loop การ update ค่า x_i

เราจะทำการ update ค่า x_i ให้ครบจำนวนครั้งที่กำหนดผ่านตัวแปร $iter$

```
...
for j=1:iter
    for i=1:length(b)
        ...
    end
end


return x
...
```

ทดสอบการทำงานของ GaussSeidel

ให้ทดลองค่าเริ่มต้นของ x ที่ต่างๆกันไปแล้วดูว่าจะได้คำตอบหรือไม่

```
A = [12 3 -5; 1 5 3; 3 7 13]
b = [1 28 76]
x = [1.0 0.0 1.0]
# x must be floating-point numbers

GaussSeidel(A,b,x,10)
```

Limitation

- เราได้เรียนไปแล้วว่า Gauss-Seidel จะใช้งานได้เมื่อเมทริกซ์ A เป็น diagonally dominant matrix
- หากเมทรีกซ์ A ไม่เป็นไปตามข้อกำหนด เราอาจไม่เจอคำตอบของระบบสมการ
- วิธีแก้ไขก็คือ เราต้องเชคก่อนว่า เมทริกซ์ A เป็น diagonally dominant matrix หรือไม่
- ซึ่งในการบ้านเราจะได้เขียนฟังก์ชันที่ทำหน้าที่ตรวจสอบว่า A เป็น diagonally dominant matrix

Limitation

- หาก A เป็น diagonally dominant matrix เราสามารถเรียกใช้ `GaussSeidel()` ได้เลย
- หากไม่ใช่ เราจำเป็นจะต้องลองจัดให้ A เป็น diagonally dominant matrix เสียก่อน

Hungarian algorithm (Munkres algorithm)

- เป็น algorithm ที่ใช้ในการจัดให้ ค่าบนเส้นทแยงมุมของเมทริกซ์มีผลรวมน้อยที่สุด เช่น
- ถ้า $A = \begin{bmatrix} 1 & 2 & 3 \\ 10 & 3 & 1 \\ 5 & 1 & 9 \end{bmatrix}$
- หากเรียกใช้ algorithm นี้ในการจัดแถวของ A จะได้ผลลัพธ์เป็น $\begin{bmatrix} 1 & 2 & 3 \\ 5 & 1 & 9 \\ 10 & 3 & 1 \end{bmatrix}$

สำหรับ diagonally dominant matrix

- สำหรับ diagonally dominant matrix เราอยากให้ค่าบนเส้นทแยงมุมของเมทริกซ์มีค่ามากสุด (หมายรวมถึงผลรวมของค่าเหล่านั้นควรจะมากสุดด้วย)
- เราสามารถเรียกใช้ Hungarian algorithm โดย run algorithm บน เมทริกซ์ $-A$

Hungarian algorithm in Julia

ลง package "Munkres"

```
Pkg.add("Munkres")
using Munkres

A = [12 3 -5; 3 7 13; 1 5 3; ]

row = munkres(-A)
print(row)

A = A[row,:]
```

Homework

- ให้ปรับปรุงโปรแกรมที่ได้เขียนไป โดยเพิ่มส่วนตรวจสอบว่า A เป็น diagonally dominant matrix หรือไม่
- หากใช่ ก็ทำการ solve เพื่อหาคำตอบโดยฟังก์ชัน GaussSeidel
- หากไม่ใช่ ก็ให้พยายามปรับให้ A เป็น diagonally dominant matrix
- ทั้งนี้หลังปรับแล้วอาจต้องทำการตรวจสอบอีกรอบว่า A เป็น diagonally dominant matrix จริง หากหลังจากใช้ Hungarian algorithm แล้ว ก็ยังไม่ได้ A ที่ต้องการ ก็ให้แสดงผลว่า ระบบสมการนี้ไม่สามารถ solve ได้