応用数理工学特論
第六回　講義ノート

高橋　一樹・近藤　祐史

1　差分法により生じる疎行列

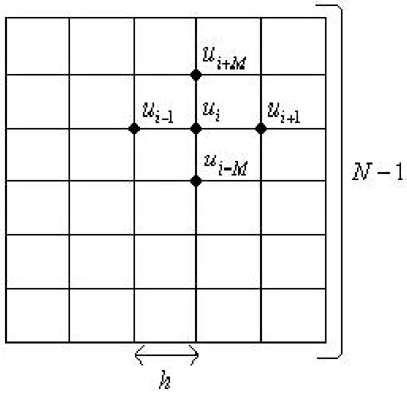
1.1　差分法で生じる疎行列

疎行列 とは行列の要素がほとんどが0である行列である
差分法により疎行列となるものの一つとして **ポアソン方程式** がある。
ポアソン方程式は $\nabla^2\phi = f$ で表される方程式で二次元のポアソン方程式は

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y) \tag{1}$$

である。
今回はこのポアソン方程式の一つとして二次元の熱伝導方程式について考える。

二次の微分に対して中心差分をとると

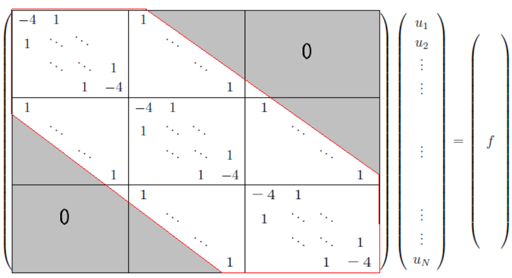


$$
\begin{aligned}
\frac{\partial^2 u}{\partial x^2} &= \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta h^2} && (2)\\
\frac{\partial^2 u}{\partial y^2} &= \frac{u_{i+M} - 2u_i + u_{i-M}}{\Delta h^2} && (3)
\end{aligned}
$$

これらを元の式に入れると

$$
\frac{1}{h^2}(u_{i+1} + u_{i-1} + u_{i+M} + u_{i-M} - 4u_i) = f \qquad (4)
$$

となるから、これを行列で表すと



となり、差分部分の行列は
非零要素が最大で５の帯行列となっている疎行列であることが分かる。
これを数値的に解く方法を考えてゆく。

2　疎行列向け一次の連立方程式の分類

2.1　直接法

直接法とは **ガウスの消去法に基づく方法** である。

長所

1. 安定性
2. 必要なメモリ量、演算時間が予測可能
3. 係数行列が同じで右辺ベクトルのみ異なる複数の連立一次方程式を解くのに有利

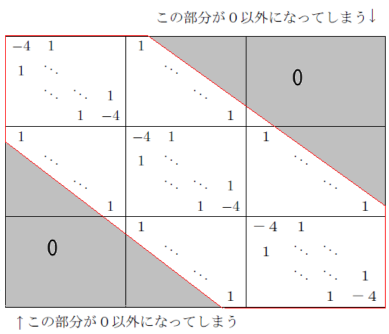
短所

1. 必要なメモリ量が大きい。$\longleftarrow$*フィルインのため**(後述参照)
2. この後記述する反復法が上手くいく場合計算時間の面で不利

＊フィルイン

フィルインとは最初 0 であった要素が消去などの過程により非零となること。
このことによりメモリ量、演算量が増加してしまう

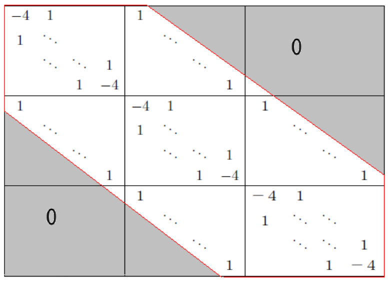
今回の例（二次の熱伝導方程式）では主に対角部分以外の０の要素が０以外の数になることが計算量の増加につながる（下図参照）

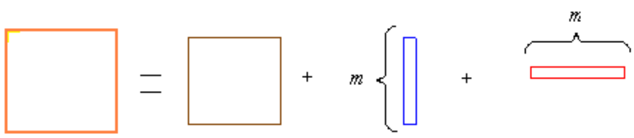


直接法の例
帯ガウス法
スパースソルバ

2.1.1　帯ガウス法

　帯の内部のみを記憶、計算して演算量とメモリ量を節約





ピポット列の長さ＝ピポット列の長さ＝m
一ステップの演算量 $O(m^2)$（半帯幅 m のとき）
メモリ：　$(2m+1) \times n \ll n^2$
演算量：　$2m^2 \times n \ll \frac{2}{3}n^3$
例えば 100×100 格子場合
行列サイズ $n = 10^4$
帯幅　　　$m = 100$
ガウスの消去法の演算量 $n^3 = (10^4)^3 = 10^{12}$
帯ガウス法の演算量　　$m^2 \cdot n = (10^2)^2 * 10^4 = 10^8$
となり大体 10^4 倍ほど演算量が減っている。

2.1.2　スパースソルバ

長所

- フィルイン現象の削減
- 非ゼロ要素のみの格納・演算
- 演算の並列性の抽出

方法　節点番号の付け替え (オーダリング) をしガウスの消去法を変形

例:Nasted Dissection オーダリング

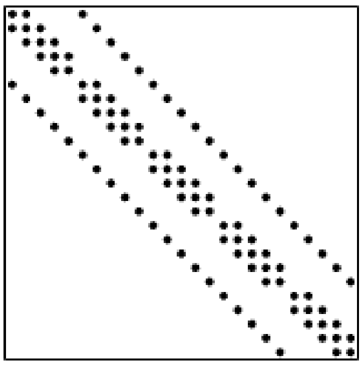
格子を部分領域 A,B とセパレーター S の 3 つの領域に分解し A,B,S の順に節点番号を振り直す (変数の並べ替えに対応)
これを再帰的に行う。
ただし、A に属する接点と B に属する接点が互いに隣接しないように S を設定することが重要

A に属する節点と B に属する節点とが隣接しないため，係数行列は縁付きブロック対角行列となり，この構造が LU 分解した後も保存されること (フィルインの抑制)、LU 分解時に計算の大部分を占める過程の並列が可能なことがうれしい。

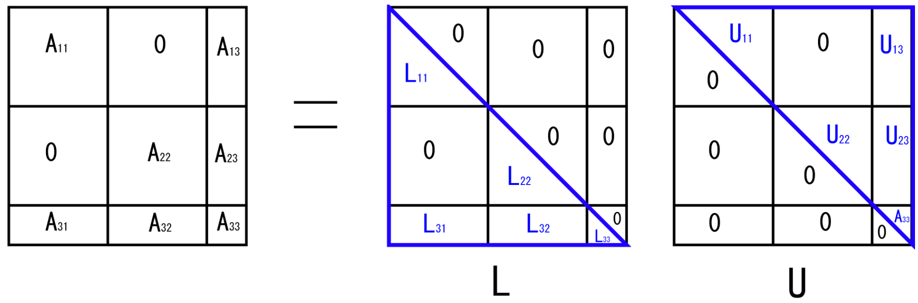
格子点:

$$
\begin{array}{|cc|c|cc|}
\hline
u_1 & u_2 & u_3 & u_4 & u_5 \\
u_6 & u_7 & u_8 & u_9 & u_{10} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
u_{21} & u_{22} & u_{23} & u_{24} & u_{25} \\
\hline
\multicolumn{2}{c}{A} & \multicolumn{1}{c}{S} & \multicolumn{2}{c}{B}
\end{array}
\Longrightarrow
\begin{array}{|cc|c|cc|}
\hline
u_1 & u_2 & u_{21} & u_{11} & u_{12} \\
u_3 & u_4 & u_{22} & u_{13} & u_{14} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
u_9 & u_{10} & u_{25} & u_{19} & u_{20} \\
\hline
\multicolumn{2}{c}{A} & \multicolumn{1}{c}{S} & \multicolumn{2}{c}{B}
\end{array}
$$

係数行列:



$$
\Longrightarrow
\left[\begin{array}{c|c|c}
A_{11} & 0 & A_{13} \\
\hline
0 & A_{22} & A_{23} \\
\hline
A_{31} & A_{32} & A_{33}
\end{array}\right]
$$



という形の分解が存在することは数学的に保証されている。

$$
\begin{aligned}
L_{11}U_{11} &= A_{11} &(5)\\
L_{11}U_{13} &= A_{13} &(6)\\
L_{22}U_{22} &= A_{22} &(7)\\
L_{23}U_{23} &= A_{23} &(8)\\
L_{31}U_{11} &= A_{31} &(9)\\
L_{32}U_{22} &= A_{32} &(10)\\
L_{31}U_{13} + L_{32}U_{23} + L_{33}U_{33} &= A_{33} &(11)
\end{aligned}
$$

この LU 分解では A_{11}, A_{22} が演算のほとんどを占め並列化が可能
また $U_{13}, U_{23}, L_{31}, L_{32}$ を求める作業も大きな計算量ではないが並列化が可能
最後に残った $A_{33} - L_{31}U_{13} - L_{32}U_{23}$ の LU 分解は並列化出来ないがセパレーター S が小さければ少ない演算

この方法は対角ブロックに対して再帰的に適用可能であるので並列化をする上でも良い解法になっている。

ポイント

- 並列化した時の計算負荷が均等になるようにする
- セパレーター S は可能な限り小さく

2.2 反復法

反復法とは解を逐次的に改良する方法である。

長所

1. 必要なメモリ量が少ない。
2. 収束が早くすむ時などうまくいく場合は非常に高速。

短所

1. 収束しない場合や非常に時間がかかる場合がある。

反復法の例
ヤコビ法、ガウスザイゼル法、SOR 法
クロリフ部分空間法（CGS,Bi-CGSTAB,GPBi-CG）