応用数値解析特論 第 7 回

～C 言語によるサンプル・プログラム, FreeFem++紹介～

かつらだ まさし
桂田 祐史
http://nalab.mind.meiji.ac.jp/~mk/ana/

2020 年 11 月 9 日

目次

本日の内容

本日は二本立て。

- 菊地 [1] 掲載のプログラムを元にしたサンプル・プログラム (C 言語で記述) の紹介
- FreeFem++ の紹介

本日の内容

本日は二本立て。

- 菊地 [1] 掲載のプログラムを元にしたサンプル・プログラム (C 言語で記述) の紹介
- FreeFem++ の紹介

(授業資料公開が遅れたことについてのおわび)

6 C 言語によるサンプル・プログラムの紹介

6.1 進行表

6 C 言語によるサンプル・プログラムの紹介

6.1 進行表

1. 百聞は一見しかず。まず実行例を見てもらう。

1. 百聞は一見しかず。まず実行例を見てもらう。
2. プログラムが何をするか、入力と出力を理解する。
   有限要素解を求めるプログラム (`naive`, `band`) では、領域や三角形分割の情報を入力データとする。そのため一般性が高くなっている。

1. 百聞は一見しかず。まず実行例を見てもらう。
2. プログラムが何をするか、入力と出力を理解する。
   有限要素解を求めるプログラム (`naive`, `band`) では、領域や三角形分割の情報を入力データとする。そのため一般性が高くなっている。
3. naive と band の比較をする。数学的にはやること同じ。効率の違いは？

6 C 言語によるサンプル・プログラムの紹介

6.1 進行表

1. 百聞は一見しかず。まず実行例を見てもらう。
2. プログラムが何をするか、入力と出力を理解する。
   有限要素解を求めるプログラム (`naive`, `band`) では、領域や三角形分割の情報を入力データとする。そのため一般性が高くなっている。
3. naive と band の比較をする。数学的にはやること同じ。効率の違いは？
4. プログラムの心臓部分 `assem()` と `ecm()` の読解 (説明したことの確認)。

6.2 試しに実行

授業 WWW サイトの「有限要素法のサンプル C プログラム」にアクセスしよう。

入手、展開、ファイル名確認

```
curl -O http://nalab.mind.meiji.ac.jp/~mk/program/fem/kikuchi-fem-mac.tar.gz
tar xzf kikuchi-fem-mac.tar.gz
cd kikuchi-fem-mac
ls
```

6.2 試しに実行

授業 WWW サイトの「有限要素法のサンプル C プログラム」にアクセスしよう。

入手、展開、ファイル名確認

```
curl -O http://nalab.mind.meiji.ac.jp/~mk/program/fem/kikuchi-fem-mac.tar.gz
tar xzf kikuchi-fem-mac.tar.gz
cd kikuchi-fem-mac
ls
```

とりあえず動作チェック (実行には、cc, cglsc, make 等が必要)

コンパイル&テスト

`make`	プログラムのコンパイル
`make test1`	naive の動作確認 (辺を 2,4,8 分割したときの有限要素解の数値データ)
`make test2`	band の動作確認 (辺を 2,4,8 分割したときの有限要素解の数値データ)
`make test3`	band の動作確認 (辺を 2,4,8,16,32 分割したときの有限要素解の等高線表示)

6.2 試しに実行

授業 WWW サイトの「有限要素法のサンプル C プログラム」にアクセスしよう。

入手、展開、ファイル名確認

```
curl -O http://nalab.mind.meiji.ac.jp/~mk/program/fem/kikuchi-fem-mac.tar.gz
tar xzf kikuchi-fem-mac.tar.gz
cd kikuchi-fem-mac
ls
```

とりあえず動作チェック (実行には、cc, cglsc, make 等が必要)

コンパイル&テスト

`make`	プログラムのコンパイル
`make test1`	naive の動作確認 (辺を 2,4,8 分割したときの有限要素解の数値データ)
`make test2`	band の動作確認 (辺を 2,4,8 分割したときの有限要素解の数値データ)
`make test3`	band の動作確認 (辺を 2,4,8,16,32 分割したときの有限要素解の等高線表示)

途中で引っかかった場合、相談して下さい。

6.3 有限要素解を求めるプログラム naive, band の理解

2 次元多角形領域 Ω における Poisson 方程式の同次 Dirichlet, Neumann 境界値問題

$$-\triangle u(x,y) = f(x,y) \equiv 1 \quad ((x,y) \in \Omega), \tag{1}$$

$$u(x,y) = 0 \quad ((x,y) \in \Gamma_1), \tag{2}$$

$$\frac{\partial u}{\partial \boldsymbol{n}}(x,y) = 0 \quad ((x,y) \in \Gamma_2) \tag{3}$$

を有限要素法で解くプログラムである。

6.3 有限要素解を求めるプログラム naive, band の理解

2 次元多角形領域 Ω における Poisson 方程式の同次 Dirichlet, Neumann 境界値問題

$$-\triangle u(x,y) = f(x,y) \equiv 1 \quad ((x,y) \in \Omega), \tag{1}$$

$$u(x,y) = 0 \quad ((x,y) \in \Gamma_1), \tag{2}$$

$$\frac{\partial u}{\partial \boldsymbol{n}}(x,y) = 0 \quad ((x,y) \in \Gamma_2) \tag{3}$$

を有限要素法で解くプログラムである。

Q ここで $\Omega, \Gamma_1, \Gamma_2$ は何か？

6.3 有限要素解を求めるプログラム naive, band の理解

2 次元多角形領域 Ω における Poisson 方程式の同次 Dirichlet, Neumann 境界値問題

$$-\triangle u(x,y) = f(x,y) \equiv 1 \quad ((x,y) \in \Omega), \tag{1}$$

$$u(x,y) = 0 \quad ((x,y) \in \Gamma_1), \tag{2}$$

$$\frac{\partial u}{\partial \boldsymbol{n}}(x,y) = 0 \quad ((x,y) \in \Gamma_2) \tag{3}$$

を有限要素法で解くプログラムである。

Q ここで $\Omega, \Gamma_1, \Gamma_2$ は何か？

A 実は $\Omega, \Gamma_1, \Gamma_2$ についてはデータとして入力する。

naive, band ともに、任意の領域&境界についての計算ができる。

6.3 有限要素解を求めるプログラム naive, band の理解

入力データの例 `input.dat`

```
9     8     5
0.0         0.0
0.0         0.5
0.0         1.0
0.5         0.0
0.5         0.5
0.5         1.0
1.0         0.0
1.0         0.5
1.0         1.0
 0     3     4     0     4     1
 1     4     5     1     5     2
 3     6     7     3     7     4
 4     7     8     4     8     5
 0     1     2     3     6
```

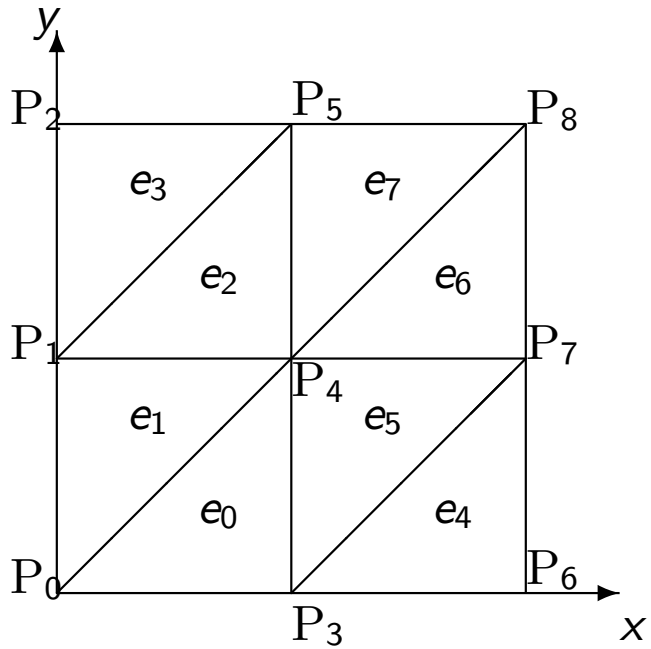


図 1: 要素分割 (各辺を 2 等分してから要素分割)

1. 1 行目には、節点数 (nnode)、要素数 (nelmt)、Γ_1 に属している節点数 (nbc)
2. 2〜10 行は、節点の座標 (x_i, y_i) $(i = 0, 1, \cdots, \mathtt{nnode} - 1)$
3. 11〜14 行は、各要素を構成する節点の全体節点番号 (0 から nelmt−1 までの通し番号)
   節点は各要素を左回りに回るように順序付けてある。
4. 最後に Γ_1 に属する節点の全体節点番号

6.3 有限要素解を求めるプログラム naive, band の理解

この形式のデータがあれば、図が描ける (幾何的状況が分かる) ことを理解しよう。

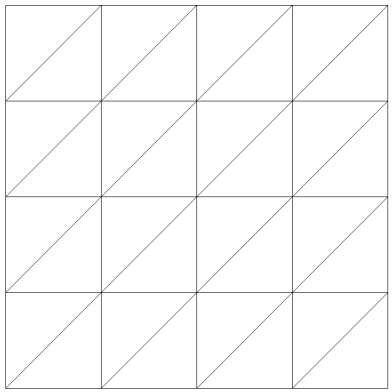
三角形と (結果として)Ω を描く

```
./disp-glsc input.dat
./disp-glsc input4.dat
cat input4.dat | ./disp-glsc
./make-input | ./disp-glsc
```

(最後のコマンドに対して、辺を何等分するか、数値を入力しよう。)

コマンド 1 | コマンド 2 でコマンド 1 の出力をコマンド 2 に入力できる (パイプ機能)。

`disp-glsc` は上の形式のデータを図示するプログラム、`make-input` は正方形領域に対して上の形式のデータを作成するプログラムである。



6.3 有限要素解を求めるプログラム naive, band の理解

naive, band は上の形式の入力データから、有限要素解を計算するプログラム。

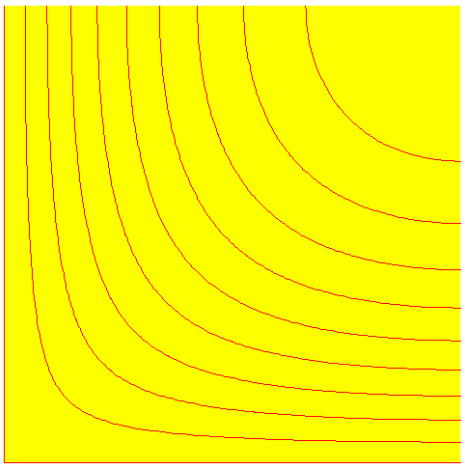
両者は同じ計算を行う。連立 1 次方程式の係数行列が**帯行列** (band matrix) であることを利用して、計算の効率化の工夫をしたのが band で、それをせず素朴な計算をするのが naive である。

一辺 64 分割で解き比べ (CPU 時間計測), 解の等高線表示

```
echo 64 | ./make-band-input > input64.dat
time ./naive input64.dat
time ./band input64.dat
```

あるマシンで 17.5 秒 vs 0.02 秒　(naive は実際的ではない)

```
./contour band.out
```



6.4 プログラム naive の内部構造

主な関数には以下のようなものがある。

`main()`	
`input()`	入力データ読み込み
`assem()`	全体係数行列 A, 全体自由項ベクトル f の計算 (直接剛性法)
`ecm()`	要素係数行列 A_e, 要素自由項ベクトル f_e の計算
`solve()`	
`output()`	節点パラメーター (節点での解の値) を出力
`f()`	Poisson 方程式 $-\triangle u = f$ の右辺の既知関数 f

6.4 プログラム naive の内部構造

主な関数には以下のようなものがある。

`main()`	
`input()`	入力データ読み込み
`assem()`	全体係数行列 A, 全体自由項ベクトル f の計算 (直接剛性法)
`ecm()`	要素係数行列 A_e, 要素自由項ベクトル f_e の計算
`solve()`	
`output()`	節点パラメーター (節点での解の値) を出力
`f()`	Poisson 方程式 $-\triangle u = f$ の右辺の既知関数 f

主な変数名

`nnode`	節点の総数
`nelmt`	有限要素の総数
`nbc`	Γ_1 (Dirichlet 境界条件を課す) 上の節点の総数
`x[nnode], y[nnode]`	節点の座標
`ielmt[nelmt][3]`	各有限要素を構成する節点の番号
`ibc[nbc]`	基本境界条件を課す節点の番号
`am[][]`	全体係数行列
`fm[]`	全体自由項ベクトル

6.4 プログラム naive の内部構造　assem()

assem() は連立 1 次方程式を組み立てる関数。

$A^* := \sum_{k=0}^{N_e-1} A_k^*$, $\boldsymbol{f}^* := \sum_{k=0}^{N_e-1} \boldsymbol{f}_k^*$ を次のように計算する。

```
/* assemblage of total matrix and vector; */
    for (k = 0; k < nelmt; k++) {
        ecm(k, ielmt, x, y, ae, fe);
        for (i = 0; i < 3; i++) {
            ii = ielmt[k].node[i];
            fm[ii] += fe[i];
            for (j = 0; j < 3; j++) {
                jj = ielmt[k].node[j];
                am[ii][jj] += ae[i][j];
            }
        }
    }
```

ielmt[k].node[i] は要素 e_k の、局所節点番号が i の節点 N_i の全体節点番号

6.4 プログラム naive の内部構造　ecm()

ecm() は要素係数行列 A_k、要素自由項ベクトル $\boldsymbol{f}_k$ を求める関数。

```
/* 節点の座標を求める */
for (i = 0; i < 3; i++) {
    j = ielmt[k].node[i];
    xe[i] = x[j];
    ye[i] = y[j];
}
```

節点の座標さえ求まれば、A_k, $\boldsymbol{f}_k$ の成分は公式に従って計算するだけである。

6.5 参考課題

以前、FreeFem++ が使えなかった頃は、授業で次のような課題を出していた。

私は「百見は一験にしかず」と考えていて、次のような実験をすることは有益と思っているが、この科目では要求しない。

(a) このプログラムで解ける問題は、境界条件が同次境界条件

$$u = 0 \quad \text{on } \Gamma_1, \qquad \frac{\partial u}{\partial \boldsymbol{n}} = 0 \quad \text{on } \Gamma_2$$

であるが、これを非同次境界条件

$$u = g_1 \quad \text{on } \Gamma_1, \qquad \frac{\partial u}{\partial \boldsymbol{n}} = g_2 \quad \text{on } \Gamma_2$$

に変える。

(b) 自分で選んだ領域を三角形分割して、このプログラムに入力できるデータを生成するプログラムを書く。

7 FreeFem++の紹介

「FreeFem++ の紹介」 を見ながら、実際に Mac にインストールして、サンプル・プログラムを動かしてみる。

参考文献

[1] 菊地文雄：有限要素法概説, サイエンス社 (1980), 新訂版 1999.