応用数理実験

Poisson 方程式, $-\triangle_D$ の固有値問題

桂田 祐史

2003 年 12 月 2 日

1 はじめに

授業中の演習は Octave を用いるが、MATLAB (1 ライセンスのみ) を使ってもらっても良い。以下の実例は基本的に Octave による。

Octave については、

『Octave Manual』http://www.math.meiji.ac.jp/~mk/labo/howto/octave.pdf

また自分用に作ったメモを

『MATLAB 手習い』http://www.math.meiji.ac.jp/~mk/labo/text/private-matlab-notebook/

で公開している。

2 Poisson 方程式

正方形領域 $\Omega = (0,1) \times (0,1)$ における Poisson 方程式の Dirichlet 境界値問題

$$\begin{aligned} -\triangle u &= f \quad (\text{in } \Omega), \\ u &= 0 \quad (\text{in } \partial\Omega) \end{aligned}$$

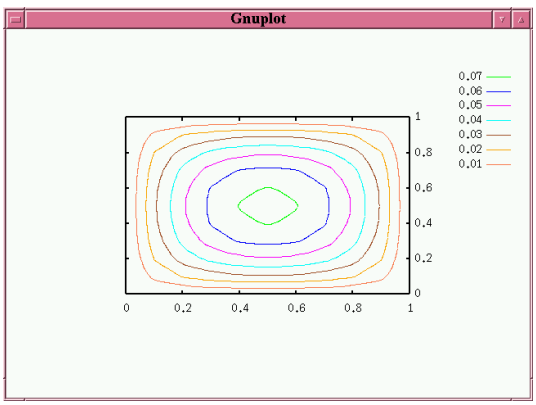
を $f \equiv 1$ の場合に差分法で解く。

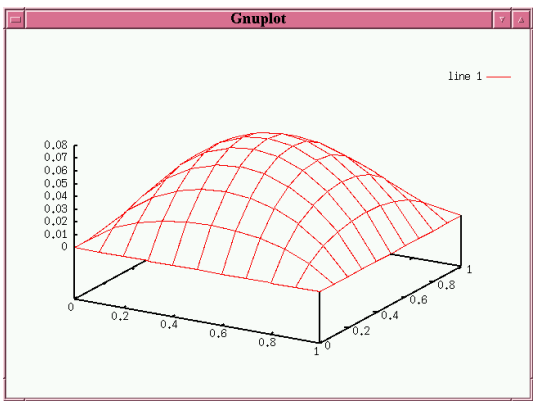
report_poisson.m

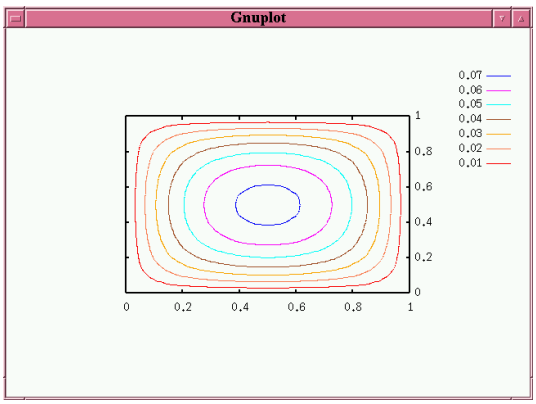
```
% report_poisson.m
%   正方形領域で -△ u=f, 同次 Dirichlet 境界値問題を解く (f ≡ 1)
%   正方形の各辺を n 等分して格子に切って差分法を用いる
function u = report_poisson(n)
  h=1/n;
  J=diag(ones(n-2,1),1)+diag(ones(n-2,1),-1);
  Id=eye(n-1,n-1);
  A=-4*kron(Id,Id)+kron(J,Id)+kron(Id,J);
  % b_{ij}=-h^2 f(x_i,y_j)
  b=-h*h*ones((n-1)*(n-1),1);
  U=A\b;
  u=zeros(n+1,n+1);
  for j=1:n-1
    u(2:n,j+1)=U((j-1)*(n-1)+1:j*(n-1));
  end
%endfunction
```

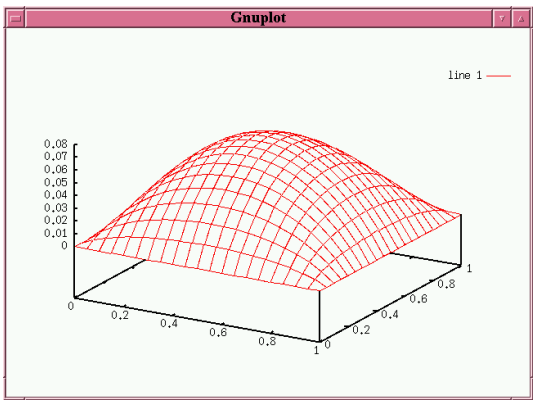
```
yurichan% octave
GNU Octave, version 2.0.16 (i386-unknown-freebsd3.4).
Copyright (C) 1996, 1997, 1998, 1999, 2000 John W. Eaton.
This is free software with ABSOLUTELY NO WARRANTY.
For details, type 'warranty'.

octave:1> n=10; u=report_poisson(n);
octave:2> x=0:1/n:1; y=x; contour(u,10,x,y);
octave:3> mesh(x,y,u);
octave:4> n=20; tic; u=report_poisson(n); toc
ans = 1.1213
octave:5> x=0:1/n:1; y=x; contour(u,10,x,y);
octave:6> mesh(x,y,u);
octave:7> n=40; tic; u=report_poisson(n); toc
ans = 16.870
octave:8> x=0:1/n:1; y=x; contour(u,10,x,y);
octave:9> quit
yurichan%
```









3 ラプラシアンの固有値問題

まずは Octave で解いてみよう。

eigen_square.m

```
1  % eigen_square.m
2  % 正方形領域のラプラシアンの固有値
3  function retval = eigen_square(n)
4    h = 1/n;
5    J=diag(ones(n-2,1),1)+diag(ones(n-2,1),-1);
6    I=eye(n-1,n-1);
7    A = - n * n * (- 4 * kron(I,I) + kron(J,I) + kron(I,J));
8    retval = eig(A);
9  %endfunction
```

eigen_square2.m

```
1  % eigen_square2.m
2  % 正方形領域のラプラシアンの固有値、固有関数
3  function [v,lambda] = eigen_square2(n)
4    h = 1/n;
5    J=diag(ones(n-2,1),1)+diag(ones(n-2,1),-1);
6    I=eye(n-1,n-1);
7    A = - n * n * (- 4 * kron(I,I) + kron(J,I) + kron(I,J));
8    [v,lambda] = eig(A);
9  %endfunction
```

$n = 10$ で解いて最初の 10 個の固有値を表示

```
octave:1> a=eigen_square(10); a(1:10)
```

`tic;a=eigen_square(10);toc` のようにして計算時間を計測した。

n	計算時間 (秒)
10	0.14443
20	7.4554
25	32.129
30	105.79

計算時間は n^6 に比例している？

4 課題

正方形領域における Laplacian の固有値問題を差分法で解け。

- 差分方程式の導出は自分で行うこと (プログラムと照らし合わせる)。
- Octave あるいは MATLAB を用いると良い。
- 得られた近似固有値の計算精度を調べよ。
- 重複固有値について調べよ。
- 近似固有関数の可視化をせよ。
- 特に絶対値の小さい近似固有関数について、厳密な固有関数と比較せよ。重複固有値に対応するものはどうなっているか？
- MATLAB で疎行列用の命令を使うとどうなるか？

5 参考書案内

菊地・山本 [7] には、正方形領域における Laplacian の固有値問題を差分法で解く方法の解説が載っている (プログラムは FORTRAN である)。

Kronecker 積 ($\otimes$) については、伊理 [2] または明治大学の線形代数の教科書 [8] を参照せよ。

Octave については、WWW 上に解説文書があふれているが、日本語で読める書籍としては大石 [3] がある。

6 MATLAB を使う場合

- 注釈に # は使えず、代わりに % を使わないといけない。
- endfunction は使えない (削除する)。

eigen_square_m.m

```
% 正方形領域のラプラシアンの固有値
function retval = eigen_square_m(n)
  h = 1/n;
  J=spdiags([ones(n-1,1) zeros(n-1,1) ones(n-1,1)], -1:1, n-1,n-1);
  I=speye(n-1,n-1);
  A = - n * n * (- 4 * kron(I,I) + kron(J,I) + kron(I,J));
  retval = eig(A);
%endfunction
```

このプログラムの oyabun 上での実行時間は、

n	実行時間 (秒)
10	0.0121
20	0.2181
40	8.0817
80	509.2671

Octave と比較してスピードもさることながら、大きい問題が解けるのが嬉しい。
疎行列の扱いに慣れるために、少し調べてみる。

例 **6.1** (単位行列を作る) n 次単位行列 I を記憶した sparse matrix `Id` を作れ。

```
Id=sparse(eye(n,n));              素朴
Id=speye(n,n);                    専用の命令 (ベストの方法)
Id=sparse(1:n,1:n,ones(1,n));     sparse() の利用
Id=sparse(1:n,1:n,ones(n,1));
Id=sparse(1:n,1:n,1);             こう簡略化できる
Id=spdiags(ones(n,1),0,n,n));     spdiag() の利用
```

例 **6.2** n 次正方行列 $J = \begin{pmatrix} 0 & 1 & & \\ 1 & 0 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 0 \end{pmatrix}$ を記憶した sparse matrix `J` を作れ。

```
J=sparse(diag(ones(n-1,1),1)+diag(ones(n-1,1),-1));
e=ones(n,1); J=spdiags([e zeros(n,1) e],-1:1,n,n);
J=sparse(1:n-1,2:n,ones(n-1,1),n,n)+sparse(2:n,1:n-1,ones(n-1,1),n,n) ;
J=sparse(1:n-1,2:n,1,n,n)+sparse(2:n,1:n-1,1,n,n)
J=spdiags(ones(n,1),1,n,n)+spdiags(ones(n,1),-1,n,n);
```

その他、調べて解説すべき命令として、

- `spconvert()`　各行が i, j, a_{ij} という 3 数である行列を入力する。

```
sdata = [1 1 5; 1 2 1; ....]
s = spconvert(sdata)
```

- 零非零パターンを見る `spy(s)` とメモリー使用量を見る `whos`
- 普通の行列に直す `full()`

参考文献

[1] 有木進, 工学のための線形代数, 日本評論社 (200?).

[2] 伊理正夫, 線形代数 I, 岩波講座 応用数学, 岩波書店 (1993).

[3] 大石進一, Linux 数値計算ツール, コロナ社 (2000).

[4] 大石進一, MATLAB による数値計算, 培風館 (2001).

[5] 小国力/Dongarra, Jack J., MATLAB による線形計算ソフトウェア, 丸善 (1998).

[6] 小国力, MATLAB と利用の実際 — 現代の応用数学と CG —, サイエンス社 (1995).

[7] 菊地 文雄, 山本 昌宏, 微分方程式と計算機演習, 山海堂 (1991).

[8] 大学数学教育研究会 編, 大学課程 線形代数, 共立出版 (1986).