計算天文学 II 第 4 回 楕円型方程式

牧野淳一郎

2003 年 11 月 17 日

1 楕円型方程式

楕円型方程式は、典型的にはラプラス方程式

$$\nabla^2 u = 0 \tag{1}$$

またはポアソン方程式

$$\nabla^2 u = -4\pi G\rho \tag{2}$$

のようなものである。上のポアソン方程式はニュートン重力の場合で、 G は重力定数、ρ は物質の質量密度 (単位体積あたりの質量) である。ここで、 ρ はとりあえず与えられた空間座標だけの関数とする。空間 1 次元ではラプラシアンがただの 2 階微分演算子になってしまい、解くべき方程式は常微分方程式ということになる。

実際にポアソン方程式 (とさらに状態方程式を連立させててくるレーン・エムデン方程式) を球対称の場合に解くというのは I のほうでやったと思うので、ここでは空間 2 次元の場合を考える。3 次元も考え方は同様である。

工学的な応用では、楕円型方程式の応用として重要なのはラプラス方程式の境界値問題である。これは、構造解析、電場の解析などあらゆるところで現れる。特に複雑な形状の機械部品とか機械全体の中での応力場の解析といったものが重要な応用ということになる。

が、天文・物理のいろんな問題では、複雑な境界値問題というのはあんまり大事ではない。天体は大抵重力でまとまっているわけで、固定された表面 (境界) というものが与えられているとは限らないからである。もちろん、数値計算の方法によってはそういうものが出てくることもあるが、とりあえず今日は比較的に単純な境界条件だけをかんがえることにしよう。

ラプラス方程式やポアソン方程式は、基本的には拡散方程式から時間微分の項を落したものと考えることができる。つまり、なんらかの定常状態を記述しているわけである。多くの応用で、実際にそのような定常状態を求めているわけであるが、数値計算という観点からすると、ラプラス方程式やポアソン方程式に適用できる計算法は実は拡散方程式や波動方程式に陰解法を適用する時にも使える。

これは、前回見たように陰解法では空間方向の差分方程式を解く必要がでてくるからである。解くべき方程式は、ラプラス方程式やポアソン方程式の差分化からでてくるものと基本的には同じである。

というわけで、今日は、単純な 2 次元ポアソン方程式の境界値問題を例に、計算方法について考えていく。

2 次元なので、ラプラシアンは

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \tag{3}$$

である。例によって変数の方を u_{ij} と書き、これが $u(x_i, y_j)$ の近似解としよう。また、前回と同じ 2 次精度の中心差分を使うとすれば、ラプラシアンの差分近似は以下で与えられる。

$$\frac{u_{i,j+1} + u_{i,j-1} + u_{i+1,j} + u_{i-1,j} - 4u_{ij}}{\Delta x^2} \tag{4}$$

但し、ここでは $\Delta x = \Delta y$ とした。いま、解くべき問題を長方形領域でのポアソン方程式の境界値問題

$$\nabla^2 u(x,y) = \rho(x,y), \quad (x_0 \le x \le x_1, y_0 \le y <\le y_1) \tag{5}$$

$$u(x_0, y) = u(x_1, y) = u(x, y_0) = u(x, y_1) = 0 \tag{6}$$

ということにすれば (以下、面倒なので $-4\pi G$ の項は 1 とする)、これは 2 次元配列 `u[nx][ny]` についての線形連立方程式ということになる。ここで

$$n_x = (x_1 - x_0)/\Delta x + 1 \tag{7}$$

$$n_y = (y_1 - y_0)/\Delta x + 1 \tag{8}$$

であり、境界上では $u_{ij} = 0$, それ以外では

$$\frac{u_{i,j+1} + u_{i,j-1} + u_{i+1,j} + u_{i-1,j} - 4u_{ij}}{\Delta x^2} = \rho_{ij} \tag{9}$$

となる。

2 線形方程式と反復法

さて、というわけで、2 次元の長方形領域上でのポアソン方程式の差分近似は、連立線形 1 次方程式 (14) になるので、これを何らかの方法で解けばいい。

空間 1 次元の場合には、出てくる行列が 3 重対角という特別な性質を持ったものであったので、自由度 (要素数) に比例する計算量やメモリー使用量で方程式を解くことができた。

2 次元の場合はどうなるかというと、そんなにうまくはいかないということがわかる。もちろん、2 次元の場合でも、 $u_{ij} = u_{i \cdot n_x + j}$ というように添字をふりなおして (以下、添字が 1 つしかないときにはこのように付け変えたものを考えるということにする)、以下のように行列の形に書くことはできる。

$$A\boldsymbol{u} = \boldsymbol{\rho} \tag{10}$$

ここで、 $\boldsymbol{u}$は u_i のベクトル表示である。しかし、 A の要素を考えてみると、対角要素、その両側の他に、 $\pm n_x$ だけ離れたところに 0 でない要素が現れる。

このために、計算量としては $O(n_x^2 n)$ ($n = n_x n_y$ として) になり、1 次元のときよりずっと計算量が多い。例えば $n_x = n_y$、つまり正方形領域とすれば計算量は結局 $O(n^2)$ になってしまう。

もちろん、これでも一般の場合のガウスの消去法の計算量である n^3 よりは少ないが、しかし非常に多いことには変わりはない。また、3 次元の場合にはこの状況はいっそう悪化する。

このために、ガウスの消去法のようにまともに行列を解くのではなく、適当な近似を繰り返していくことで解を求めることはできないかということが昔から考えられてきた。以下、その方法のいくつかを紹介する。これらを反復法という。

2.1 ガウス反復

なんらかの方法で適当な近似解 $\boldsymbol{u}^i$ (ここで上つき添字 i は i 番目の近似解ということで、別に $\boldsymbol{u}$ の i 乗という意味ではない。下につけると他の添字と混乱するので上につけてみた) があるとする。これをもうちょっとましな解にできないかということを考える。

一つの方法は、行列 A を形式的に以下のように分解することである。

$$A = D + N \tag{11}$$

ここで、 D は対角要素 a_{ii} だけをとりだした行列で、 N はそれ以外の残りである。これを使って、以下のような式を考える

$$D\boldsymbol{u}^{i+1} = -N\boldsymbol{u}^i + \boldsymbol{\rho} \tag{12}$$

D は対角行列なので、これは解けていて、実際のプログラムとしては、

```
for (i=1; i<nx-1;i++){
    for (j=1; j<ny-1;j++){
        unew[i][j] = 0.25*(u[i][j-1]+u[i][j+1]+u[i-1][j]+u[i+1][j])
                    - rho[i][j]*0.25*deltax*deltax;
    }
}
```

ということになる。1 で始まって $n_x - 2$ 等で終っているのは、境界は 0 固定だからである。その分の配列を節約することも考えられるが、いじましいしちょっとわかりにくくなるので上の形に書くことにしよう。

この反復で、収束すれば、つまり、 $\boldsymbol{u}^{i+1} = \boldsymbol{u}^i$ となれば、これらが元の方程式を満たしていることは明らかであろう。収束するかどうかは面倒なのでここでは省くが、上のような簡単なポアソン方程式の離散化で妙な非線形項とかがなければ収束するということが証明できる。

実用上は、収束したかどうか判定する必要がある。これには、普通は $|\boldsymbol{u}^{i+1} - \boldsymbol{u}^i|^2$ を計算してそれが適当な設定した値よりも小さくなったらいいことにする。

理論的には、数列は連続する 2 項間の差が小さくなっても収束に近付いているとは限らないが、ガウス反復の場合には、丸め誤差がなければこれで判定できることが証明できる。

が、この方法には 2 つ問題がある。一つは収束が非常に遅いことであり、もう一つは u と unew で配列が 2 ついるのでメモリがたくさん必要になることである。

2.2 ガウス・ザイデル法

さて、実は、上の 2 つの問題は一度に解決することができる。上のプログラムを、以下のように変えてしまうのである。

```
for (i=1; i<nx-1;i++){
    for (j=1; j<ny-1;j++){
        u[i][j] = 0.25*(u[i][j-1]+u[i][j+1]+u[i-1][j]+u[i+1][j])
                    - rho[i][j]*0.25*deltax*deltax;
    }
}
```

なにをしたかというと、 unew というのをやめにして u にしただけである。つまり、一回の反復の中では u の要素を順に変えていくわけだが、その時に、既に新しい値が求まっていればそっちを使おうというだけのことである。新しい値のほうが、多分より正確であろうと考えられるからである。

形式的には、これは以下のように書くことができる。 行列 A を、対角成分 D、非対角成分の上半三角分 U、残り L の 3 個に分ける $(A = D + L + U)$ と、

$$(L + D)\boldsymbol{u}^{i+1} = -U\boldsymbol{u}^{i} + \boldsymbol{\rho} \tag{13}$$

ということになる。

2.3 SOR

というわけで、ガウス・ザイデルはガウス反復よりはましだが、まだもうちょっとなんとかならないものかという気もする。実際に近似解の挙動を見るとわかるが、どちらの方法でも、誤差の減り方にはパターンがあって、1 方向から真の解に近付いている。

というわけで、以下のようなことが考えられる：ガウス・ザイデル法で、修正量 $\Delta\boldsymbol{u} = \boldsymbol{u}^{i+1} - \boldsymbol{u}^{i}$ に適当な係数 ω を掛けて水増ししてやればもうちょっとうまくいくのではないか？

こんな安直な方法で大丈夫なのかと思うであろうが、うまくいってしまうのがすばらしいところである。プログラムとしては、ガウス・ザイデルの時の式をちょっと変えて、

```
u[i][j] += omega*(0.25*(u[i][j-1]+u[i][j+1]+u[i-1][j]+u[i+1][j])
            - rho[i][j]*0.25*deltax*deltax -u[i][j])
```

というだけで済む。実際的な問題は、 ω の値をどうとるかということである。理論的には $1 \leq \omega < 2$ でないといけないことがわかっていて、さらに本当はもちろん線形のポアソン方程式とかなら最適な ω の値が計算できるが、ここではそこまでは立ち入らないことにする。

この方法を SOR (Successive Over Relaxation, 本によっては Simultaneous ... となっているものもあるかもしれない) という。

3　もっと高度な解法

天文等で出てくるポアソン方程式の場合には、特にうまく ω を決められれば SOR で非常にうまくいくことが多い。が、まあ、世の中はそういう問題ばかりでもないので、それ以外の方法を紹介だけしておく。

3.1　ADI

ADI は alternating direction implicit の略である。ここでの基本的な考察は、「3 重対角なら簡単に解ける」ということである。これを使って、 x 方向と y 方向を交代で解いているうちになんとかなって欲しいというのが ADI の考え方ということになる。

もうちょっとちゃんと書いてみると、例えば j 方向に解くときには、

$$\frac{u_{i,j+1}^{new} + u_{i,j-1}^{new} - 2u_{ij}^{new} + u_{i+1,j} + u_{i-1,j} - 2u_{ij}}{\Delta x^2} = \rho_{ij} \tag{14}$$

となる。ここで、 new とつけたのが更新後の値で、これを $i = 1$ から順に解いていく。 j 方向が終ったら次は i 方向をやる。

ただし、実際に使う上では、 SOR と同じように、修正量の補正をしたほうが収束が速い。 SOR とは逆に修正量をすこし減らす必要があり、どれくらい減らすべきかを決めるのにはまたややこしい理屈がいる。

ただし、適当に修正量を決めても最適な SOR よりも遅くはない程度で収束するという利点がある。

3.2　共役勾配法

共役勾配法は、実は、現在大規模な行列を解くのにはもっとも広く使われている方法である。これは、特にいろいろな前処理と組み合わせることで、例えば拡散方程式の定常問題で拡散係数が空間に強く依存する場合にも収束させられるとか、刻みを場所によって変えるような高度な方法でも同様になんとかできるとかいう利点があるからである。

しかし、これは安直にプログラムを書いただけではこれまでの紹介してきた方法に比べて速くなくて、いろいろ高度な変形をして初めて速くなる。また、ほとんど共役勾配法だけについて議論した良い教科書がいくつもあるので、ここでは省略する。ここでは、とりあえずそういう名前のものがあるということくらいをちょっと憶えておいてほしい。

とはいえ、極めて重要な方法ではあるので、これについては、もうちょっと数学的な準備をした後でもう一度やることにする。

3.3　FFT を使った方法

ポアソン方程式を解く方法の一つで、応用上は重要なのはフーリエ変換を使った方法である。いま、固定境界をやめて周期境界条件でいいことにすると、ポアソン方程式の場合、 ρ をフーリエ級数展開して、出てきた係数に $1/k^2$ (k は波数) を掛けてから逆フーリエ変換すればポアソン方程式の解になっているわけである。固定境界なら sin 関数だけで展開すればよい。フーリエ変換の計算量は

多次元でも格子点の総数を n として $O(n \log n)$ の程度なので、これは反復法でいうと回数がせいぜい $\log n$ で済むということに相当する。

これは極めて強力かつ高速な方法であり、

- 長方形領域でいい
- 周期境界または固定境界でいい
- 空間微分の係数が座標によらない
- 格子が等間隔でよい

という条件がすべて満足されていれば、必ずこの方法を使うべきといってもさしつかえない。

星の構造を解くとかいう場合だと、極座標を使って球面調和関数展開ということも多い。この場合の計算には FFT のようなうまい方法があるわけではない (最近高速ルジャンドル変換の研究もあるが、これはよほど項数が大きくないと速くならない) が、r 方向は展開しないで済ませられるのでまあつかえなくはない。

4 練習

1. ガウス・ザイデル法で以下のポアソン方程式の境界値問題を解くプログラムを作成せよ。

$$\nabla^2 u(x,y) = \rho(x,y), \quad (x_0 \le x \le x_1, y_0 \le y <\le y_1) \tag{15}$$

$$u(x_0,y) = u(x_1,y) = u(x,y_0) = u(x,y_1) = 0 \tag{16}$$

   ただし、 $x_0 = y_0 = 0$, $x_1 = y_1 = 1$ としてよい (入力パラメータとして読める汎用的なプログラムを作ってくれればもちろんそのほうがよい)。さらに、$\rho(x,y) = \sin \pi x \sin \pi y$ として、初期推定 $\boldsymbol{u} = 0$ から出発した時に

   - 残差 $|A\boldsymbol{u} - \boldsymbol{\rho}|$ が反復を繰り返した時にどのように 0 に近付くか
   - 収束した時に、真の解との差 (誤差) はどうなっているか

   を空間刻み n_x ($n_y = n_x$ としてよい) のいくつかの値について調べ、それから精度が n_x の何乗になっているか議論せよ。

2. さらに、 SOR について同様に調べよ。最適な ω の値は n_x に依存するかどうかも議論すること。

3. $\rho(x,y) = \sin k\pi x \sin k\pi y$ (k は整数) の時に、誤差はどうなるか、理論的または数値的 (両方あればそれも OK) 議論せよ。

4. ρ がデルタ関数 $\delta(x-0.5, y-0.5)$ の時に、厳密解をフーリエ級数の形で表せ。

5. 上と同じデルタ関数の時に数値解を求めて、 $y = 0.5$ の線上で誤差がどのように分布するか調べよ。また、誤差の等高線表示や 3 次元表示を適当なツールを使って作ってみよ。(PGPLOT には等高線表示をするサブルーチンが準備されている)

6. 例えば 3 次元的な星の自己重力を計算するためには、境界条件を「無限遠で 0 」という形に置く必要がある。これにはどうすればよいか検討せよ。

5　レポート

前回と今回の「練習」の中から、2 つ以上を選んで解き、

1. プログラム
2. 実行結果
3. 考察

をまとめて提出すること。なお、提出は、メイルで

makino@astron.s.u-tokyo.ac.jp

あてに、 Subject を kadai-2 として提出すること。〆切は 12/1。

注意

今週の資料では、牧野は実際にプログラムを組んで動かしてみていないので、差分式とかには間違いがある可能性がある。プログラムを書いて動かない時には、資料の式が間違っている可能性も含めて調べること。

6　次週予告

次回から、常微分方程式の初期値問題を扱う。これはすでに I でやっているわけだが、もうちょっといろいろなことを考える。