応用複素関数

数値計算実習

桂田 祐史

katurada ＡＴ meiji.ac.jp

2017年6月21日, 2017年8月11日

http://nalab.mind.meiji.ac.jp/~mk/complex2/

ポテンシャル問題の数値計算法については、別の文書 [1] を用意した。(もっとも、この「応用複素関数」では、数値計算法そのものを学ぶものでなく、とにかく数値計算する方法はある、という「紹介だけする」姿勢に徹する。興味を持った人が、適当な時期に見てくれれば良い。)

1　2017/6/21 のスケジュール

(今日はなるべく前の方に座りましょう。)

1. まず「応用複素関数」[1] から FreeFem++ のインストール用の .pkg ファイルを入手してもらう。
2. レポート課題について説明する (§2)。
3. 学生の大部分がファイルを入手出来た段階で、FreeFem++ のインストールを実演するので、それを真似てインストールしてもらう (§§4.3)。
4. FreeFem++ を紹介して、サンプル・プログラムを解説し、試しに実行してみる (§§4.4)。
5. 時間があれば、レポート課題 1 に取り掛かることを勧める (§3)。

2　レポート課題 part 1

締め切りは 7 月 21 日 19:30 (以前 18:00 と言いましたが、ゼミをやっている時間で、中途半端であると言われたので、少しずらしました), 提出方法は Oh-o! Meiji. もし容量制限に引っかかった場合は、早目にメールで相談して下さい。(FreeFem++ がうまくインストール出来ない、という場合も相談に乗りますが、〆切が近づくとこちらも忙しいので、早目早目でお願いします。)

以下 [1]〜[5] から 1 つ以上解いて下さい。使用するプログラミング言語の選択は基本的に自由ですが、桂田が相談に乗るためには、自分の MacBook Air で実行して見せてもらう必要があります[2]。

［1］正則関数の定める流れ (Mathematica を使うことを想定)

[1] http://nalab.mind.meiji.ac.jp/~mk/complex2/

[2] ずっと、以前、家のパソコンでしか動きません、というプログラムがあって、うまく行かなかったことがあるため。

(1) 渦糸の等ポテンシャル線、流線、ベクトル場を適当に (流れの様子が良く分かるように) 可視化する。
(湧き出しのサンプル・プログラムはあるので簡単のはず。追記 プログラムは、講義内容と対応するように書かれていて、きちんと解読すること。もちろん細かいところは問題に合うように書き換える必要がある。)

(2) 自分で思いつく正則関数を 5 つ以上試し (「係数だけを変えて数合わせ」ではなく、なるべく「違う」ものを選ぶこと)、そのうちの 1 つを選んで、それを複素速度ポテンシャルとする流れについて、等ポテンシャル線、流線、ベクトル場を適切に可視化し、それをもとにどういう流れであるか説明する (流量の計算をしてみるとか…)。
正則関数の実部・虚部の等高線自体は、関数論のテキストの多くに載っている。凝りたければ、今井 [2] 等を見て関数を選ぶと良い。追記 「簡単な流れの合成」という話をしてある。それや初等関数の定める流れを描いてみることを授業で勧めた。

[2] ポテンシャル問題を解くことによる流れの決定 (FreeFem++ を使うことを想定)
2 次元渦なし非圧縮の定常流で、流体の占める領域の境界での流速が分かっている場合に、速度ポテンシャル ϕ、流れ関数 ψ を計算して、等ポテンシャル線, 流線, ベクトル場を可視化せよ。— ϕ は

$$\triangle\,\phi = 0 \quad \text{in } \Omega, \quad \frac{\partial \phi}{\partial n} = \boldsymbol{v}\cdot\boldsymbol{n} \quad \text{on } \partial\Omega$$

の解として求まる。ψ はどうすれば良いか？(念のため注意: 講義で説明したように、解 ϕ が存在するためには $\displaystyle\int_{\partial\Omega} \boldsymbol{v}\cdot\boldsymbol{n}\, d\sigma = 0$ が必要である。$\overline{\Omega}$ で定義された $\boldsymbol{v}$ で $\operatorname{div}\boldsymbol{v} = 0$ を満たすものはこの条件を満たすが、$\partial\Omega$ の上だけで $\boldsymbol{v}$ を与える場合は注意すること。)

[3] 差分法による Poisson 方程式の解法 (1) (MATLAB を使うことを想定)
こちらが提示した MATLAB のサンプル・プログラムは、$f \equiv 1$, $g_1 \equiv 0$ という特殊な場合に

$$-\triangle\,u = f \quad \text{in } \Omega, \quad u = g_1 \quad \text{on } \partial\Omega$$

を解くものであるが、これをより一般の条件のもとで解けるようにプログラムを拡張する。そのプログラムを別のプログラミング言語 (例えば Python) に移植出来たら加点する。

[4] 差分法による Poisson 方程式の解法 (2)
Poisson 方程式の Neumann 境界値問題、すなわち

$$-\triangle\,u = f \quad \text{in } \Omega, \quad \frac{\partial u}{\partial n} = g_2 \quad \text{on } \partial\Omega$$

の解が存在するためには、$\displaystyle\int_{\partial\Omega} g_2\, d\sigma = -\int_{\Omega} f\, d\boldsymbol{x}$ という条件が必要であり、解が存在するときも解の一意性は成立しない (u に任意の定数を加えたものはまた解になるから)。この事情は、離散化して作った差分方程式 $A\boldsymbol{U} = \boldsymbol{F}$ でも変わらない。$\det A = 0$ となっているため、解が存在するためには $\boldsymbol{F}$ に条件が必要で、解が存在する場合も一意性は成立しない (分からなければ線型代数を復習すること)。この問題を解決し、差分解を求めるプログラムを作る。1 次元、すなわち

$$-u''(x) = f(x) \quad (x \in (0,1)), \quad u'(0) = a, \quad u'(1) = b$$

の場合で構わない。講義では、$u(0) = \alpha$, $u'(1) = \beta$ という問題に対して、行列 A を与えてあるので、それをたたき台にして考察すると良い (なお、プログラムも C で書いたものを提供してある http://nalab.mind.meiji.ac.jp/~mk/complex2/potential/node5.html)。
この問題に対して、“いい加減な” アプローチをしているプログラムが結構ある。実は、この

講義で提供している FreeFem++ のプログラムもそうである (笑)。一度じっくり考えてみる価値のある問題である。

[5] (基本解の方法に関する課題)
a, b を $a > b > 0$ を満たす実数とするとき

$$\Omega := \left\{ z = x + yi \in \mathbb{C} \;\middle|\; \frac{x^2}{a^2} + \frac{y^2}{b^2} < 1 \right\}$$

とおく (つまり、Ω は楕円領域である)。Ω の等角写像 $\varphi\colon \Omega \to D_1$ を、基本解の方法を用いて求めよ。(もし、厳密解を文献で見つけて、それを計算する方法が得られたら、上で計算した解と比べて見よ。)

3　正則関数を複素速度ポテンシャルとする流れとその可視化

3.1　用語と基本的な性質のおさらい

流体の 2 次元の流れを考える。速度場を

$$\boldsymbol{v}(x,y)=\begin{pmatrix}u(x,y)\\v(x,y)\end{pmatrix}$$

とする。

$$\operatorname{div}\boldsymbol{v}:=u_x+v_y$$

で定義される $\operatorname{div}\boldsymbol{v}$ ($\nabla\cdot\boldsymbol{v}$ とも書く) を**発散**と呼び、いたるところ

$$\operatorname{div}\boldsymbol{v}=0$$

であるとき、流れは**非圧縮**であるという。

$$\omega:=\begin{vmatrix}\frac{\partial}{\partial x} & u\\ \frac{\partial}{\partial y} & v\end{vmatrix}=v_x-u_y \quad (\text{これも } \operatorname{rot}\boldsymbol{v} \text{ と書くことがある})$$

で定義される ω を**渦度**と呼び、いたるところ

$$\omega=0$$

であるとき、流れは**渦なし**であるという。

- 渦なしならば速度ポテンシャル ϕ が存在: $\exists\phi$ s.t. $\operatorname{grad}\phi=\begin{pmatrix}\phi_x\\\phi_y\end{pmatrix}=\boldsymbol{v}$. (ただし ϕ は、一般には多価関数である。単連結領域の場合は一価関数である。)

  $\triangle\phi=\operatorname{div}\boldsymbol{v}$ であるので、流れが非圧縮ならば $\triangle\phi=0$.

  一価関数である場合、例えば領域の境界 $\partial\Omega$ 上で $\boldsymbol{v}$ が得られれば、

  $$\triangle\phi=0 \quad (\text{in } \Omega), \qquad \frac{\partial\phi}{\partial\boldsymbol{n}}=\boldsymbol{v}\cdot\boldsymbol{n} \quad (\text{on } \partial\Omega).$$

  これは Laplace 方程式の境界値問題として、解くことが出来る。

- 非圧縮ならば流れ関数 ψ が存在: $\exists\psi$ s.t. $\begin{pmatrix}\psi_y\\-\psi_x\end{pmatrix}=\boldsymbol{v}$. (ただし ψ は、一般には多価関数である。単連結領域の場合は一価関数である。)

  $\triangle\psi=-\omega$ であるので、流れが渦なしならば $\triangle\psi=0$.

  一価関数である場合、領域の境界上の $\boldsymbol{v}$ が得られれば、やはり ψ は Laplace 方程式の境界値問題 (どんな？これは各自の練習問題) の解として得られる。

- ϕ, ψ のいずれかが求まれば、微分するだけで速度 $\boldsymbol{v}$ が求まる (流れが分かったことになる)。

- 渦なし非圧縮流に対して、**複素速度ポテンシャル** $f:=\phi+i\psi$ は正則関数で、$f'=u-iv$. (逆に任意の正則関数は、ある渦なし非圧縮流の複素速度ポテンシャルである。)

- 曲線で、その接線ベクトルが速度ベクトル $\boldsymbol{v}$ と方向が同じものを**流線**と呼ぶ。定常流の場合、流線は流体の粒子の軌跡と一致する。流れ関数 ψ の等高線 ($\psi=$ 定数) は流線を表す。

- $\phi=$ 定数 で表される曲線を**等ポテンシャル線**と呼ぶ。

- 渦なし非圧縮流では、流線と等ポテンシャル線は互いに直交する (正則関数の実部・虚部の等高線は互いに直交する)。

3.2　簡単な流れ

3.2.1　一様な流れ

$c \in \mathbb{C} \setminus \{0\}$, $f(z) = cz$ の場合、$c = Ue^{-i\alpha}$ $(U > 0,\ \alpha \in \mathbb{R})$ とする。複素速度は

$$u - iv = f'(z) = Ue^{-i\alpha}.$$

すなわち

$$\boldsymbol{v} = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} U\cos\alpha \\ U\sin\alpha \end{pmatrix}.$$

速度ポテンシャルと流れ関数は

$$\begin{cases} \phi(x,y) = \mathrm{Re}\left((U\cos\alpha - i\sin\alpha)(x+iy)\right) = U(x\cos\alpha + y\sin\alpha), \\ \psi(x,y) = \mathrm{Im}\left((U\cos\alpha - i\sin\alpha)(x+iy)\right) = U(-x\sin\alpha + y\cos\alpha). \end{cases}$$

等ポテンシャル線も、流線も平行直線群で、それらは互いに直交する。

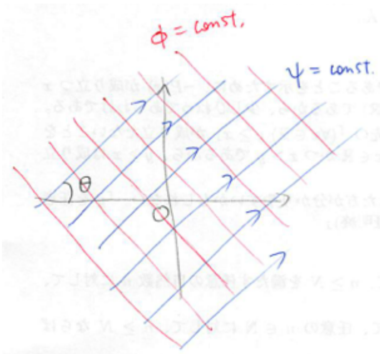

図 1: 一様な流れ

3.2.2　湧き出し、吸い込み

$m \in \mathbb{R}$, $f(z) = m\log z$ $(z \in \mathbb{C} \setminus \{0\})$ の場合 (多価関数！)。(多価関数は気持ち悪いかもしれないが、しばらく我慢。$\sqrt{z}$ などと違って微分すると一価関数になるので、案外面倒なことにはならない。)

$z = re^{i\theta} = x + iy$ とすると、

$$u(x,y) - iv(x,y) = f'(z) = \frac{m}{z} = \frac{m}{r}e^{-i\theta} = \frac{m}{r}\left(\cos\theta - i\sin\theta\right).$$

すなわち

$$\boldsymbol{v} = \begin{pmatrix} u \\ v \end{pmatrix} = \frac{m}{r}\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}.$$

(方向は $\begin{pmatrix} x \\ y \end{pmatrix}$ と同じ、$m > 0$ ならば向きも同じ、$m < 0$ ならば反対向き、大きさ $|\boldsymbol{v}| = \frac{|m|}{r}$ は原点から遠いほど小さい。)

速度ポテンシャルと流れ関数は、$f(re^{i\theta}) = m\left(\log r + i\left(\theta + 2n\pi\right)\right)$ $(n \in \mathbb{Z})$ より

$$\begin{cases} \phi = m\log r, \\ \psi = m\left(\theta + 2n\pi\right) \quad (n \in \mathbb{Z}). \end{cases}$$

(ϕ は一価関数、ψ は多価関数である。)

等ポテンシャル線は、原点を中心とする円であり、流線は、原点を端点とする半直線である。

原点の周りを一周する任意の閉曲線 C を取ると、C から外に湧き出る流量 (流束) は

$$\int_C \boldsymbol{v}\cdot\boldsymbol{n}\,ds = \int_C d\psi = \mathrm{Im}\int_C df = \mathrm{Im}\int_C f'(z)dz = \mathrm{Im}\int_C \frac{m}{z}dz = 2\pi m$$

と一定値である。

(計算の確認: 既に見たように、$\boldsymbol{n}\,ds = \begin{pmatrix} -dy \\ dx \end{pmatrix}$ であり、$\boldsymbol{v}\cdot\boldsymbol{n}\,ds = -v\,dx + u\,dy = \psi_x\,dx + \psi_y\,dy = d\psi$ である。また、$df = f'(z)\,dz = (\phi_x + i\,\psi_x)(dx + i\,dy) = (\phi_x\,dx - \psi_x\,dy) + i(\psi_x\,dx + \phi_x\,dy) = (\phi_x\,dx + \phi_y\,dy) + i\,(\psi_x\,dx + \psi_y\,dy) = d\phi + i\,d\psi$.)

$m > 0$ ならば原点に置かれた**湧き出し**、$m < 0$ ならば原点に置かれた**吸い込み**をいう。

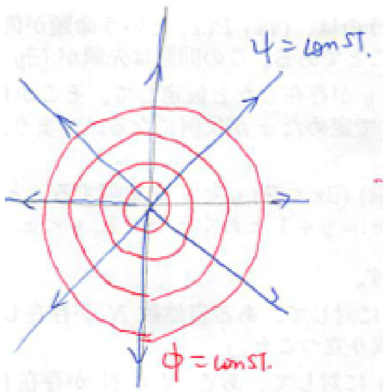

図 2: 湧き出し

3.2.3 渦糸

(上の f の m を純虚数に変えてみる。)

$\kappa \in \mathbb{R}$, $f(z) = i\kappa\log z$ $(z \in \mathbb{C}\setminus\{0\})$ の場合。$z = re^{i\theta} = x + iy$ とすると、

$$u(x,y) - iv(x,y) = f'(z) = \frac{i\kappa}{z} = \frac{i\kappa}{r}e^{-i\theta} = \frac{i\kappa}{r}(\cos\theta - i\sin\theta) = \frac{\kappa}{r}(\sin\theta + i\cos\theta).$$

すなわち

$$\boldsymbol{v} = \begin{pmatrix} u \\ v \end{pmatrix} = \frac{\kappa}{r}\begin{pmatrix} \sin\theta \\ -\cos\theta \end{pmatrix}.$$

($\begin{pmatrix} \cos\theta \\ -\sin\theta \end{pmatrix} = \begin{pmatrix} \cos\frac{-\pi}{2} & -\sin\frac{-\pi}{2} \\ \sin\frac{-\pi}{2} & \cos\frac{-\pi}{2} \end{pmatrix}\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$ であるから、$\boldsymbol{v}$ の方向は、$\begin{pmatrix} x \\ y \end{pmatrix}$ を $-\pi/2$ 回転した方向である。「応用複素関数」なんだから、行列でなくて、$\sin\theta + i\,(-\cos\theta) = -i\,(\cos\theta + i\sin\theta)$ と説明すべきか？)

速度ポテンシャルと流れ関数は、$f\left(re^{i\theta}\right) = i\kappa\,(\log r + i\,(\theta + 2n\pi))$ $(n \in \mathbb{Z})$ より

$$\begin{cases} \phi = -\kappa\,(\theta + 2n\pi) \quad (n \in \mathbb{Z}) \\ \psi = \kappa\log r. \end{cases}$$

流線は原点を中心とする円で、等ポテンシャル線は原点を端点とする半直線である。

原点に置かれた**渦糸** (**渦点**) の流れと呼ばれる。渦度は $\mathbb{C}\setminus\{0\}$ 全体で 0 であることに注意しよう。

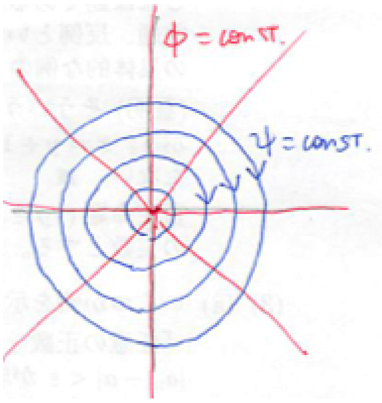

図 3: 渦糸 ($\kappa > 0$ の場合は時計回り)

3.3 正則関数の定める流れを Mathematica で可視化

複素数の計算、2 変数関数の等高線と 2 次元ベクトル場の描画が出来れば良い。

虚数単位は `I`, 実部 `Re[ ]`, 虚部 `Im[ ]`, 共役複素数 `Conjugate[ ]`, 絶対値 `Abs[ ]`, 偏角 (の主値) `Arg[ ]`, 式の中のすべての変数を実数と仮定して実部・虚部に展開する `ComplexExpand[ ]` などが用意されている。

2 変数関数の等高線の描画には `ContourPlot[ ]`, ベクトル場の描画には `VectorPlot[ ]` が用意されている。これらの使い方はオンライン・ヘルプを見よ (例えば `?ContourPlot`)。

「応用複素関数」[3] に、`uniformflow.nb`[4] `source.nb`[5] というサンプル・プログラムを用意してある (保存してから Mathematica で開くこと)。

[3] http://nalab.mind.meiji.ac.jp/~mk/complex2/

[4] http://nalab.mind.meiji.ac.jp/~mk/complex2/uniformflow.nb

[5] http://nalab.mind.meiji.ac.jp/~mk/complex2/source.nb

$c = 1 - 2i$ の場合の $f(z) = cz$ を複素速度ポテンシャルとする流れ

```
c=1-2I;
f[z_]:=c z;
ComplexExpand[f[x+I y]]
phi[x_,y_]:=ComplexExpand[Re[f[x+I y]]];
psi[x_,y_]:=ComplexExpand[Im[f[x+I y]]];

phi[x,y]

psi[x,y]

g1=ContourPlot[phi[x,y]==Table[c,{c,-5,5,1.0}],{x,-2,2},{y,-2,2},
  ContourStyle->Directive[Red,Thin]]

g2=ContourPlot[psi[x,y]==Table[c,{c,-5,5,1.0}],{x,-2,2},{y,-2,2},
  ContourStyle->Directive[Blue,Thin]]

u[x_, y_] := ComplexExpand[Re[f'[x + I y]]];
v[x_, y_] := -ComplexExpand[Im[f'[x + I y]]];

g3 = VectorPlot[{u[x, y], v[x, y]}, {x, -2, 2}, {y, -2, 2}]

g12=Show[g1,g2]

g13=Show[g1,g3]
```

ComplexExpand[ ] は、事前に Evaluate させる効果も考えて採用したが、いつもこれを使うのが良いかは分からない。

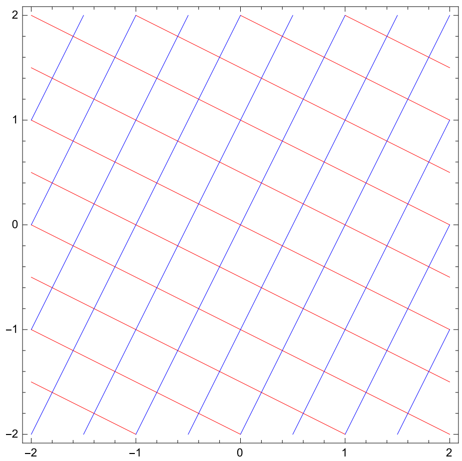

図 4: 等ポテンシャル線と流線は直交する

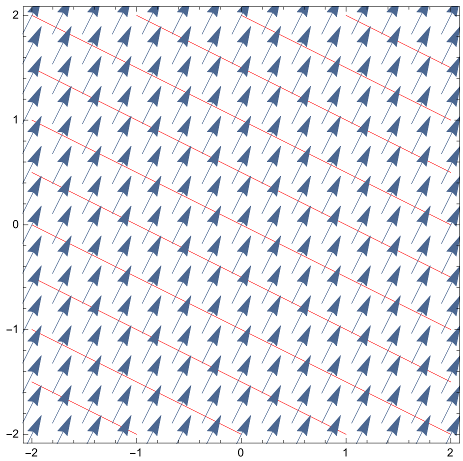

図 5: 等ポテンシャル線と速度ベクトル

このように、一様流は素直なので簡単に描画できるが、そうでないものは色々調整が必要になったりする (湧き出しの場合の source.nb が参考になるかも)。

4　ポテンシャル問題の有限要素法による数値計算 (しばらく工事中)

4.1　ポテンシャル問題とは

Laplace 方程式 $\triangle u = 0$ の境界値問題を**ポテンシャル問題**という。正則関数の実部・虚部は調和関数 (ラプラス方程式の解) であるため、関数論のあちこちの重要な場面でポテンシャル問題が登場する。

(2 次元渦無し非圧縮流の速度ポテンシャル ϕ は、Laplace 方程式の Neumann 境界値問題

$$\triangle \phi = 0 \quad \text{in } \Omega, \tag{1}$$

$$\frac{\partial \phi}{\partial n} = \boldsymbol{v} \cdot \boldsymbol{n} \quad \text{on } \partial\Omega \tag{2}$$

の解である、ということを既に紹介したが、後日他のポテンシャル問題も紹介する。)

以下では、これを少し一般化した Poisson 方程式の境界値問題 (3), (4), (5) を考える。

実は、この問題は非常に筋の良い問題である。そのため、様々な数値計算法が適用出来る。この講義では、(1) 差分法, (2) 有限要素法, (3) 基本解の方法を紹介する。

ポテンシャル問題の数値計算については、解説文書「ポテンシャル問題の数値計算」(`http://nalab.mind.meiji.ac.jp/~mk/complex2/potential.pdf`) にまとめる予定で、差分法 (とサンプル・プログラム) については、そちらを参照のこと。

この節では、有限要素法による数値計算を説明する。

4.2　Poisson 方程式の境界値問題とそれに対する弱形式

Ω を $\mathbb{R}^2$ の有界な領域、その境界 $\partial\Omega$ が Γ_1, Γ_2 に分かれているとする。

$$\partial\Omega = \Gamma_1 \cup \Gamma_2, \quad \Gamma_1 \cap \Gamma_2 = \emptyset.$$

また領域 Ω の境界 $\partial\Omega$ 上の点における外向きの単位法線ベクトルを n とする。

$f \colon \Omega \to \mathbb{R}$, $g_1 \colon \Gamma_1 \to \mathbb{R}$, $g_2 \colon \Gamma_2 \to \mathbb{R}$ が与えられたとき、

$$-\triangle u = f \quad (\text{in } \Omega) \tag{3}$$

$$u = g_1 \quad (\text{on } \Gamma_1) \tag{4}$$

$$\frac{\partial u}{\partial n} = g_2 \quad (\text{on } \Gamma_2) \tag{5}$$

を満たす u を求めよ、というのが Poisson 方程式の境界値問題である。

注: $\dfrac{\partial u}{\partial n}(\boldsymbol{x}) = \lim\limits_{h \to -0} \dfrac{u(\boldsymbol{x} + hn) - u(\boldsymbol{x})}{h} = \nabla u(\boldsymbol{x}) \cdot n.$

この問題は、次のように変形できる (弱定式化, weak formulation)。

Find $u \in X_{g_1}$ s.t.

$$\int_\Omega \nabla u \cdot \nabla v \, d\boldsymbol{x} = \int_\Omega f v \, d\boldsymbol{x} + \int_{\Gamma_2} g_2 v \, ds \quad (v \in X). \tag{6}$$

ここで

$$X_{g_1} := \left\{ w \in H^1(\Omega) \mid w = g_1 \quad \text{on } \Gamma_1 \right\}, \quad X := \left\{ w \in H^1(\Omega) \mid w = 0 \quad \text{on } \Gamma_1 \right\}.$$

(念のため: $\nabla u \cdot \nabla v = \operatorname{grad} u \cdot \operatorname{grad} v = u_x v_x + u_y v_y$.)

条件 (6) を**弱形式**と呼ぶ。適当な条件のもとで、弱形式を満たす $u \in X_{g_1}$ は一意に定まり、それがもとの問題の解となる。弱形式を満たす u を求めることでもとの問題を解く方法を**弱解の方法**と呼ぶ。(そのルーツとして有名なのは、Riemann による写像定理の、Dirichlet の原理を用いた証明である— 後日説明する。)

4.3 有限要素法, FreeFem++

有限要素法 (finite element method) は、弱解の方法を原理とする数値計算法である。それはかなりの部分を自動化出来るため、専用のソフトウェアが開発されている。

その 1 つである、**FreeFem++**[6] は、パリ第 6 大学 J. L. Lions 研究所の Frédéric Hecht, Oliver Pironneau, A. Le Hyaric, 広島国際学院大学の大塚厚二氏らが開発した、2 次元, 3 次元問題を有限要素法で解くための、 一種の PSE (problem solving environment) である。ソースコード、マニュアル (約 400 ページ, 幸い英文)、主なプラットホーム (Windows, Mac, Linux) 向けの 実行形式パッケージが公開されている。

FreeFem++ については、大塚・高石 [3] という解説書も出ているが、簡単なことは、「FreeFEM++の紹介」[7] を読めば分かるであろう。

インストールは次の手順で行う。授業でやって見せるので、なるべくその時に一緒にやってみて、分からなかったら質問して下さい。

1. 自分が使っている Mac OS X (macOS) のバージョンを確認する (「この Mac について」を見るとよい)。

2. 「応用複素関数」の WWW サイト (または **FreeFem++**[8]) 上記の WWW サイトから、自分の OS のバージョンに合ったインストール用の .pkg ファイル (`FreeFem++-3.53-1-MacOS_10.11.pkg` のような名前) を入手する。

3. 入手した .pkg ファイルを実行する (Control+クリックする必要があるかも)。
   もしも検証に時間がかかるならば、`/System/Library/CoreServices` にある “インストーラ” を起動し、そこから .pkg ファイルを指定する。

4. `/usr/local/ff++` の下にある `bin` という名前のディレクトリィを探し、PATH にそれを設定する。`FreeFem++-3.53-1-MacOS_10.11.pkg` の場合は、`/usr/local/ff++/openmpi-2.1/3.53/bin` というディレクトリィなので、`~/.profile` (または `~/.bash_profile`) の下の方に

   ```
   export PATH=$PATH:/usr/local/ff++/openmpi-2.1/3.53/bin:/usr/local/ff++/openmpi-2.1/bin
   ```

   (従来の `$PATH` に `:` で区切って追加する。)

   という行を書き込む。ターミナルを起動し直すか、`source .profile` のようにして設定を読み直す。

`/usr/local/ff++/share/freefem++/freefem++doc.pdf` というマニュアルがある。

[6] `http://www.freefem.org/ff++/`
[7] `http://nalab.mind.meiji.ac.jp/~mk/labo/text/welcome-to-freefem/`
[8] `http://www.freefem.org/ff++/`

4.4 (1), (2) を解くための FreeFem++ のプログラム

速度ポテンシャルを求める境界値問題

(再 1) $$\triangle\phi = 0 \quad (\text{in } \Omega)$$

(再 2) $$\frac{\partial\phi}{\partial\boldsymbol{n}} = \boldsymbol{v}\cdot\boldsymbol{n} \quad (\text{on } \partial\Omega)$$

の場合は、$\Gamma_1=\emptyset$, $\Gamma_2=\partial\Omega$, $f=0$, $g_2=\boldsymbol{v}\cdot\boldsymbol{n}$.

特に $\Omega=\{(x,y)\in\mathbb{R}^2 \mid x^2+y^2<1\}$, $\boldsymbol{v}\equiv\begin{pmatrix}1\\2\end{pmatrix}$ のときは、$(x,y)\in\partial\Omega$ において $\boldsymbol{n}=\begin{pmatrix}x\\y\end{pmatrix}$ であるから、

$$g_2=\boldsymbol{v}\cdot\boldsymbol{n}=\begin{pmatrix}1\\2\end{pmatrix}\cdot\begin{pmatrix}x\\y\end{pmatrix}=x+2y.$$

速度ポテンシャルを求める potential2d-v0.edp[9] ← これを保存する

```
// potential2d-v0.edp
//   http://nalab.mind.meiji.ac.jp/~mk/complex2/potential2d-v0.edp
//   2 次元非圧縮ポテンシャル流
//     速度ポテンシャル、流れ関数、速度を求め
//     等ポテンシャル線，流線，速度場を描く

border Gamma(t=0,2*pi) { x = cos(t); y = sin(t); } // 円盤領域
int m=40;
mesh Th=buildmesh(Gamma(m));
plot(Th, wait=1, ps="Th.eps");

fespace Vh(Th,P1);
Vh u, v, phi, psi;
func Vn=x+2*y; // Ωが単位円で，V=(1,2) のとき V・n=x+2y

// 速度ポテンシャルφを求め、その等高線（等ポテンシャル線）を描く
solve Laplace(phi,v) =
  int2d(Th)(dx(phi)*dx(v)+dy(phi)*dy(v))
  -int1d(Th,Gamma)(Vn*v);
plot(phi,ps="contourpotential.eps",wait=1);

// ベクトル場 (u,v)=∇φ を描く (ちょっと雑なやり方)
u=dx(phi);
v=dy(phi);
plot([u,v],ps="vectorfield.eps",wait=1);

// 等ポテンシャル線とベクトル場を同時に描く
plot([u,v],phi,ps="both.eps", wait=1);
```

プログラムはテキスト・エディター (mi, emacs, Xcode, テキスト・エディット[10]など) で作成し、ターミナルから、

こんなふうにして実行

```
FreeFem++ potential2d.edp
```

とタイプして実行できる。リターン・キーを打つごとに次の図に進み、最後 (ベクトル場と等ポテンシャル線を描いてお終い) は Escape キーを打って終了する。

このプログラムは、図の PostScript データ “counterpotential.eps”, “vectorfield.eps”, “both.eps” を出力している。

[10] テキストエディットで、テキスト・ファイルを入力・編集するには若干の設定が必要である。http://nalab.mind.meiji.ac.jp/~mk/knowhow-2015/node1.html を見よ。

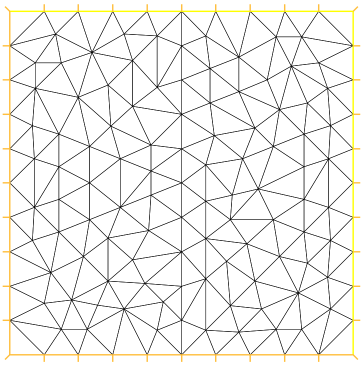

図 6: Ω の三角形分割

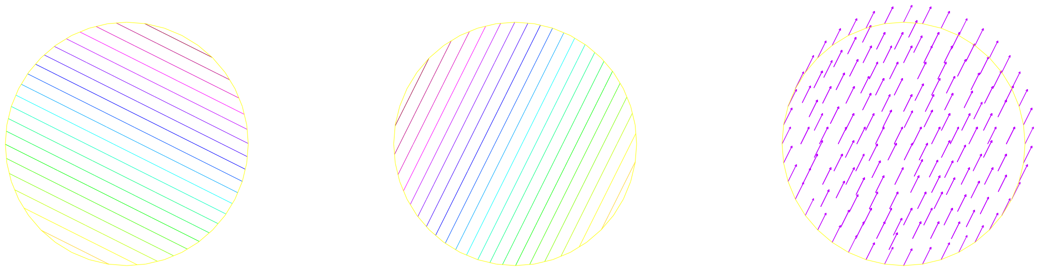

図 7: 一様流の等ポテンシャル線, 流線, ベクトル場

(実は、上の弱形式は解の一意性がないので、このプログラムは少し危ういところがある。)

FreeFem++ はポテンシャル問題に限らず、色々な問題を解くことが出来る。時間が余った時の脇道用: Navier-Stokes 方程式による、有名な円柱の周りの水の流れのシミュレーション・プログラム NS-cylinder.edp[11] (鈴木 厚先生の Finite element programming by FreeFem++ – intermediate course[12] から)

参考文献

[1] 桂田祐史：ポテンシャル問題の数値計算, http://nalab.mind.meiji.ac.jp/~mk/complex2/potential.pdf (2017).

[2] 今井功：複素解析と流体力学, 日本評論社 (1989).

[3] 大塚厚二, 高石武史：有限要素法で学ぶ現象と数理 — FreeFem++数理思考プログラミング —, 共立出版 (2014), http://comfos.org/jp/ffempp/book/ というサポート WWW サイトがある。

[11] https://www.ljll.math.upmc.fr/~suzukia/FreeFempp-tutorial-JSIAM2016/EDP/NS-cylinder.edp
[12] https://www.ljll.math.upmc.fr/~suzukia/FreeFempp-tutorial-JSIAM2016/