

応用数値解析特論 第 11 回

～Navier-Stokes 方程式に対する有限要素法 (2)～
定常 Navier-Stokes 方程式に対する Newton 法

かつらだ まさし
桂田 祐史

<http://nalab.mind.meiji.ac.jp/~mk/ana2021/>

2021 年 12 月 6 日

目次

- 1 本日の講義内容、連絡事項
- 2 Navier-Stokes 方程式に対する有限要素解析
 - 定常 Navier-Stokes 方程式に対する Newton 法
 - 例題プログラムの紹介
 - 領域とその三角形分割
 - 方程式
 - 問題点と対処の方針
 - Newton 法の復習
 - 我々の問題での f と f' は?
 - Newton 反復の弱形式
 - プログラム中の弱形式解読
 - 初期値の選択
 - 余談: Oseen 方程式
- 3 補足
- 4 参考文献

本日の講義内容、連絡事項

- 定常 Navier-Stokes 方程式の境界値問題を解く 1 つのアルゴリズムを解説する。キーワードは Newton 法である。
- Newton 法の参考資料
 - 「非線形方程式、特に代数方程式の数値解法」 (桂田 [1])
 - 「多次元 Newton 法の例」
<http://nalab.mind.meiji.ac.jp/~mk/labo/text/Newton/>
 $-u''(x) = u(x)^2$ ($x \in (0, 1)$), $u(0) = u(1) = 0$ を差分法で解くプログラムの紹介。

11.4 定常 Navier-Stokes 方程式に対する Newton 法

11.4.1 例題プログラムの紹介

前回、定常 Stokes 方程式を有限要素法で解いた。今回は定常 Navier-Stokes 方程式を有限要素法で解いてみよう。

例題プログラムは FreeFem++ 4.9 に付属する

```
/usr/local/ff++/share/FreeFEM/4.9/examples/examples/NSNewton.edp  
(2012 January, Hecht)
```

である。

ターミナルでこうすればコピーできる (4.9 インストールした人向け)

```
cp -p /usr/local/ff++/share/FreeFEM/4.9/examples/examples/NSNewton.edp .
```

テキスト・エディターで開いて中身を読んでみよう。

これはマニュアル (Hecht [2]) の §2.12 に掲載されているプログラムとほぼ同じである (本来完全に同じにすべきであろう)。

11.4.2 領域とその三角形分割

Ω は $C_e + B_{eb} + B_{eo} + B_{eu} - C_c$ で囲まれる領域とする。ただし、 $R = 5$, $L = 15$,

$$C_e : (x, y) = (R \cos t, R \sin t) \quad (t \in [\pi/2, 3\pi/2]),$$

$$C_c : (x, y) = ((\cos t)/2, (\sin t)/2) \quad (t \in [0, 2\pi]),$$

$$B_{eb} : (x, y) = (Lt^{1.2}, -R) \quad (t \in [0, 1]),$$

$$-B_{eu} : (x, y) = (Lt^{1.2}, R) \quad (t \in [0, 1]),$$

$$B_{eo} : (x, y) = (L, t) \quad (t \in [-R, R]).$$

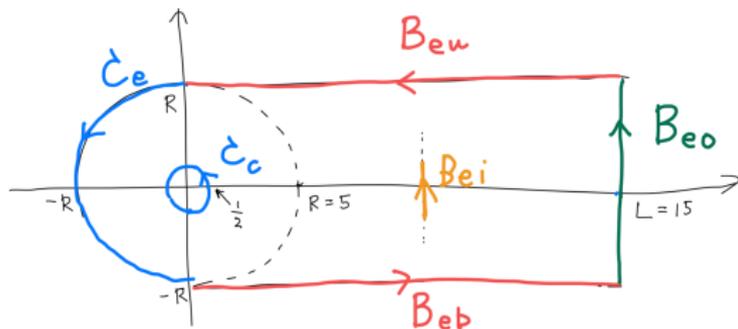


図 1: 領域 Ω ($\partial\Omega = C_e + B_{eb} + B_{eo} + B_{eu} - C_c$)

11.4.2 領域とその三角形分割

$$\Gamma_1 := -C_c + C_e + B_{eb} + B_{eu}, \quad \Gamma_2 := B_{eo}$$

とおく。 $\Gamma_1 \cup \Gamma_2 = \partial\Omega$ である (境界条件の種類に応じた分割)。

11.4.2 領域とその三角形分割

$$\Gamma_1 := -C_c + C_e + B_{eb} + B_{eu}, \quad \Gamma_2 := B_{eo}$$

とおく。 $\Gamma_1 \cup \Gamma_2 = \partial\Omega$ である (境界条件の種類に応じた分割)。

B_{eb} は $(x, y) = (Lt, R)$ ($t \in [0, 1]$) と書いても同じことであるが、FreeFem++ では t について等分することを想定してプログラムを書いたため、この式を採用しているのであろう。(こういう工夫が実際に効果があるのかどうかは分からない。)

11.4.2 領域とその三角形分割

$$\Gamma_1 := -C_c + C_e + B_{eb} + B_{eu}, \quad \Gamma_2 := B_{eo}$$

とおく。 $\Gamma_1 \cup \Gamma_2 = \partial\Omega$ である (境界条件の種類に応じた分割)。

B_{eb} は $(x, y) = (Lt, R)$ ($t \in [0, 1]$) と書いても同じことであるが、FreeFem++ では t について等分することを想定してプログラムを書いたため、この式を採用しているのであろう。(こういう工夫が実際に効果があるのかどうかは分からない。)

B_{eu} はプログラムでは

```
border beu(tt=1, 0){real t=tt^1.2; x=t*L; y=R; label=1;}
```

として与えられている。FreeFem++ では、 $tt=1,0$ と書くことで、パラメーターが 1 から 0 まで減少する曲線を表すことが出来る、ということであろう (こういうことが出来ることは、このプログラムを見て初めて知った)。

11.4.2 領域とその三角形分割

三角形分割をする際に

$$B_{ei}: (x, y) = (L/2, t) \quad (t \in [-R/4, R/4])$$

という曲線 (線分) も用いている。

```
Th=buildmesh(cc(-40)+ce(20)+beb(15)+beu(15)+beo(8)+bei(10));
```

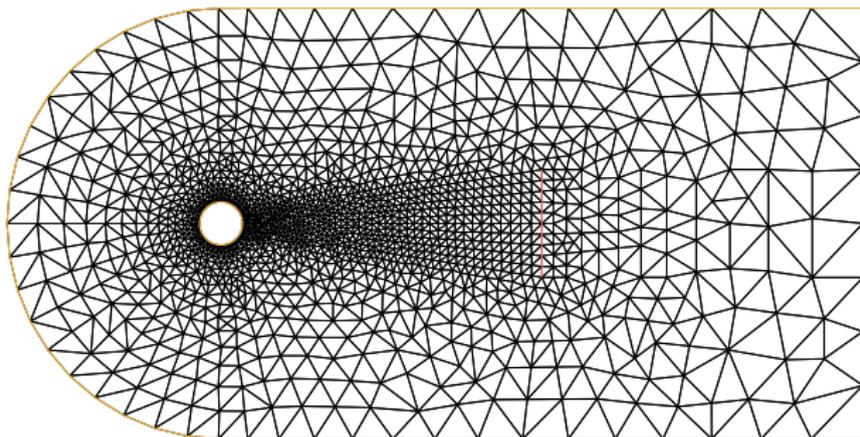


図 2: 領域の三角形分割. 中央部に B_{ei} (朱色の線分) が見える

11.4.3 方程式

このプログラムが解く問題の方程式は (境界条件が、マニュアルにも、プログラムの注釈にも明記されていないが (真面目にやってほしい)…)、多分、次のものである。

$$(1a) \quad (\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega,$$

$$(1b) \quad \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega,$$

$$(1c) \quad \mathbf{u} = \mathbf{U} \quad \text{on } \Gamma_1 \setminus C_c,$$

$$(1d) \quad \mathbf{u} = \mathbf{0} \quad \text{on } C_c,$$

$$(1e) \quad -pn + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

11.4.3 方程式

このプログラムが解く問題の方程式は (境界条件が、マニュアルにも、プログラムの注釈にも明記されていないが (真面目にやってほしい)…)、多分、次のものである。

$$(1a) \quad (\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega,$$

$$(1b) \quad \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega,$$

$$(1c) \quad \mathbf{u} = \mathbf{U} \quad \text{on } \Gamma_1 \setminus C_c,$$

$$(1d) \quad \mathbf{u} = \mathbf{0} \quad \text{on } C_c,$$

$$(1e) \quad -pn + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

(判断の根拠: 1 というラベルをつけた Γ_1 で $\delta \mathbf{u} = \mathbf{0}$ としていることから、そこで Dirichlet 境界条件をつけているのだろう。Newton 法の初期値 \mathbf{u}_0 は

$$\mathbf{u}_0 = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

であったから、 $\delta \mathbf{u}$ も Γ_1 では $\mathbf{0}$ である。)

11.4.3 方程式

このプログラムが解く問題の方程式は (境界条件が、マニュアルにも、プログラムの注釈にも明記されていないが (真面目にやってほしい)…)、多分、次のものである。

$$(1a) \quad (\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega,$$

$$(1b) \quad \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega,$$

$$(1c) \quad \mathbf{u} = \mathbf{U} \quad \text{on } \Gamma_1 \setminus C_c,$$

$$(1d) \quad \mathbf{u} = \mathbf{0} \quad \text{on } C_c,$$

$$(1e) \quad -pn + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

(判断の根拠: 1 というラベルをつけた Γ_1 で $\delta \mathbf{u} = \mathbf{0}$ としていることから、そこで Dirichlet 境界条件をつけているのだろう。Newton 法の初期値 \mathbf{u}_0 は

$$\mathbf{u}_0 = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

であったから、 $\delta \mathbf{u}$ も Γ_1 では $\mathbf{0}$ である。)

(1e) は自然境界条件である (弱形式には explicit に現れない — Poisson 方程式の問題における同次 Neumann 境界条件のようなもの)。

11.4.3 方程式

このプログラムが解く問題の方程式は (境界条件が、マニュアルにも、プログラムの注釈にも明記されていないが (真面目にやってほしい)…)、多分、次のものである。

$$(1a) \quad (\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega,$$

$$(1b) \quad \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega,$$

$$(1c) \quad \mathbf{u} = \mathbf{U} \quad \text{on } \Gamma_1 \setminus C_c,$$

$$(1d) \quad \mathbf{u} = \mathbf{0} \quad \text{on } C_c,$$

$$(1e) \quad -pn + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

(判断の根拠: 1 というラベルをつけた Γ_1 で $\delta \mathbf{u} = \mathbf{0}$ としていることから、そこで Dirichlet 境界条件をつけているのだろう。Newton 法の初期値 \mathbf{u}_0 は

$$\mathbf{u}_0 = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

であったから、 $\delta \mathbf{u}$ も Γ_1 では $\mathbf{0}$ である。)

(1e) は自然境界条件である (弱形式には explicit に現れない — Poisson 方程式の問題における同次 Neumann 境界条件のようなもの)。

一様流 ($\mathbf{U} := (1, 0)^\top$) の中に置かれた円柱の周りの流れを求めよという問題である。

11.4.3 方程式

このプログラムが解く問題の方程式は (境界条件が、マニュアルにも、プログラムの注釈にも明記されていないが (真面目にやってほしい)…)、多分、次のものである。

$$(1a) \quad (\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in } \Omega,$$

$$(1b) \quad \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega,$$

$$(1c) \quad \mathbf{u} = \mathbf{U} \quad \text{on } \Gamma_1 \setminus C_c,$$

$$(1d) \quad \mathbf{u} = \mathbf{0} \quad \text{on } C_c,$$

$$(1e) \quad -pn + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

(判断の根拠: 1 というラベルをつけた Γ_1 で $\delta \mathbf{u} = \mathbf{0}$ としていることから、そこで Dirichlet 境界条件をつけているのだろう。Newton 法の初期値 \mathbf{u}_0 は

$$\mathbf{u}_0 = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

であったから、 $\delta \mathbf{u}$ も Γ_1 では $\mathbf{0}$ である。)

(1e) は自然境界条件である (弱形式には explicit に現れない — Poisson 方程式の問題における同次 Neumann 境界条件のようなもの)。

一様流 ($\mathbf{U} := (1, 0)^\top$) の中に置かれた円柱の周りの流れを求めよという問題である。

前回スライド p. 15 「もう 1 つの弱形式」と類似点が多い。見比べてみよう。

11.4.4 問題点と対処の方針

問題になることと、それへの対処について、次の2点を指摘しておく。

11.4.4 問題点と対処の方針

問題になることと、それへの対処について、次の2点を指摘しておく。

- ① Navier-Stokes 方程式は非線形方程式であるので、一度に解を求めるのが難しい。Newton 法を用いて解くことにする (反復計算で近似解を求める)。

11.4.4 問題点と対処の方針

問題になることと、それへの対処について、次の2点を指摘しておく。

- a) Navier-Stokes 方程式は非線形方程式であるので、一度に解を求めるのが難しい。Newton 法を用いて解くことにする (反復計算で近似解を求める)。
- b) 動粘性係数 $\nu = \frac{1}{200}$ は素朴な数値計算にとっては、かなり小さい (“円柱” の直径 1 であるから、これを代表的な長さ、 $\|\mathbf{U}\| = 1$ を代表的な速さに選ぶと、 ν の逆数が Reynolds 数 R_e に相当するので、 $R_e = 200$ ということになり、これが実はかなり大きい、ということである)。Newton 法の初期値は真の解に十分近いものを選ばないと、反復が収束しないかもしれない。そのため、大きめの ν についての解を求め、それを小さい ν の問題の Newton 法の初期値とすることで、解きたい ν に近づいていく。

11.4.5 Newton 法の復習

方程式 $f(u) = 0$ の近似解を求めるための Newton 法とは、まず適当な初期値 u_0 を選び、漸化式

$$(2) \quad u_{i+1} = u_i - f'(u_i)^{-1}f(u_i)$$

により $\{u_i\}$ を定め、十分大きな i に対して、 u_i を近似解に採用する、というものである。

11.4.5 Newton 法の復習

方程式 $f(u) = 0$ の近似解を求めるための Newton 法とは、まず適当な初期値 u_0 を選び、漸化式

$$(2) \quad u_{i+1} = u_i - f'(u_i)^{-1} f(u_i)$$

により $\{u_i\}$ を定め、十分大きな i に対して、 u_i を近似解に採用する、というものである。

(2) は、 $f(u) = 0$ を $u = u_i$ で線形近似した

$$f'(u_i)(u - u_i) + f(u_i) = 0$$

の解を u_{i+1} とする、ということである。

11.4.5 Newton 法の復習

方程式 $f(u) = 0$ の近似解を求めるための Newton 法とは、まず適当な初期値 u_0 を選び、漸化式

$$(2) \quad u_{i+1} = u_i - f'(u_i)^{-1}f(u_i)$$

により $\{u_i\}$ を定め、十分大きな i に対して、 u_i を近似解に採用する、というものである。

(2) は、 $f(u) = 0$ を $u = u_i$ で線形近似した

$$f'(u_i)(u - u_i) + f(u_i) = 0$$

の解を u_{i+1} とする、ということである。

$w_i := f'(u_i)^{-1}f(u_i)$ とおくと、 w_i は $f'(u_i)w_i = f(u_i)$ の解であることに注意すると、次のアルゴリズムを得る (FreeFem++ のマニュアルでは、 F の微分は DF と表している)。

Find $u \in V$ such that $f(u) = 0$ where $f: V \rightarrow V$.

1. choose $u_0 \in V$
2. for ($i=0$; $i<niter$; $i=i+1$)
 1. Solve $f'(u_i)w_i = f(u_i)$
 2. $u_{i+1} := u_i - w_i$
break if $\|w_i\| < \varepsilon$ (修正量が小さくなったら反復を停止する)

11.4.6 我々の問題での f と f' は？

$f = 0$ が (1a)–(1e) と一致するようにするには

$$f(\mathbf{u}, p) := \begin{pmatrix} (\mathbf{u} \cdot \nabla)\mathbf{u} - \nu \Delta \mathbf{u} + \nabla p \\ \nabla \cdot \mathbf{u} \\ \mathbf{u}|_{\Gamma_1} - \mathbf{u}_b \\ -p\mathbf{n} + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} \Big|_{B_{e_0}} \end{pmatrix}$$

とすれば良いだろう。ここで \mathbf{u}_b は次式で定まる Γ_1 上の関数である。

$$\mathbf{u}_b := \begin{cases} (1, 0)^\top & (\text{on } \Gamma_1 \setminus C_c) \\ (0, 0)^\top & (\text{on } C_c). \end{cases}$$

11.4.6 我々の問題での f と f' は？ (続き)

$$\begin{aligned}
 & f(\mathbf{u} + \delta\mathbf{u}, p + \delta p) - f(\mathbf{u}, p) \\
 &= \begin{pmatrix} ((\mathbf{u} + \delta\mathbf{u}) \cdot \nabla) \mathbf{u} - \nu \Delta(\mathbf{u} + \delta\mathbf{u}) + \nabla(p + \delta p) - ((\mathbf{u} \cdot \nabla) \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p) \\ \nabla \cdot (\mathbf{u} + \delta\mathbf{u}) - \nabla \cdot \mathbf{u} \\ (\mathbf{u} + \delta\mathbf{u})|_{\Gamma_1} - \mathbf{u}_b - (\mathbf{u}|_{\Gamma_1} - \mathbf{u}_b) \\ -(p + \delta p)\mathbf{n} + \nu \frac{\partial(\mathbf{u} + \delta\mathbf{u})}{\partial \mathbf{n}} \Big|_{B_{eo}} - \left(-p\mathbf{n} + \nu \frac{\partial \mathbf{u}}{\partial \mathbf{n}} \Big|_{B_{eo}} \right) \end{pmatrix} \\
 &= \begin{pmatrix} (\delta\mathbf{u} \cdot \nabla) \mathbf{u} + (\mathbf{u} \cdot \nabla) \delta\mathbf{u} - \nu \Delta(\delta\mathbf{u}) + \nabla(\delta p) + \delta\mathbf{u} \cdot \nabla(\delta\mathbf{u}) \\ \nabla \cdot (\delta\mathbf{u}) \\ \delta\mathbf{u}|_{\Gamma_1} \\ -(\delta p)\mathbf{n} + \nu \frac{\partial(\delta\mathbf{u})}{\partial \mathbf{n}} \Big|_{B_{eo}} \end{pmatrix}.
 \end{aligned}$$

であるから

$$f'(\mathbf{u}, p) \begin{pmatrix} \delta\mathbf{u} \\ \delta p \end{pmatrix} = \begin{pmatrix} (\delta\mathbf{u} \cdot \nabla) \mathbf{u} + (\mathbf{u} \cdot \nabla) \delta\mathbf{u} - \nu \Delta(\delta\mathbf{u}) + \nabla(\delta p) \\ \nabla \cdot (\delta\mathbf{u}) \\ \delta\mathbf{u}|_{\Gamma_1} \\ -(\delta p)\mathbf{n} + \nu \frac{\partial(\delta\mathbf{u})}{\partial \mathbf{n}} \Big|_{B_{eo}} \end{pmatrix}.$$

(書くと手間だが、要するに2次式であるから、慣れると結果がすぐに分かる。)

11.4.7 Newton 反復の弱形式

$f'(\mathbf{u}_i, p_i) \begin{pmatrix} \delta \mathbf{u} \\ \delta p \end{pmatrix} = f(\mathbf{u}_i, p_i)$ を具体的に書くと

$$(3a) \quad (\delta \mathbf{u} \cdot \nabla) \mathbf{u}_i + (\mathbf{u}_i \cdot \nabla) \delta \mathbf{u} - \nu \Delta(\delta \mathbf{u}) + \nabla(\delta p) = (\mathbf{u}_i \cdot \nabla) \mathbf{u}_i - \nu \Delta \mathbf{u}_i + \nabla p_i \quad \text{in } \Omega,$$

$$(3b) \quad \nabla \cdot \delta \mathbf{u} = \nabla \cdot \mathbf{u}_i \quad \text{in } \Omega,$$

$$(3c) \quad \delta \mathbf{u} = 0 \quad \text{on } \Gamma_1,$$

$$(3d) \quad -(\delta p) \mathbf{n} + \nu \frac{\partial(\delta \mathbf{u})}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

当たり前のことであるが、 $\delta \mathbf{u}$, δp についての線形方程式である。

11.4.7 Newton 反復の弱形式

$f'(\mathbf{u}_i, p_i) \begin{pmatrix} \delta \mathbf{u} \\ \delta p \end{pmatrix} = f(\mathbf{u}_i, p_i)$ を具体的に書くと

$$(3a) \quad (\delta \mathbf{u} \cdot \nabla) \mathbf{u}_i + (\mathbf{u}_i \cdot \nabla) \delta \mathbf{u} - \nu \Delta(\delta \mathbf{u}) + \nabla(\delta p) = (\mathbf{u}_i \cdot \nabla) \mathbf{u}_i - \nu \Delta \mathbf{u}_i + \nabla p_i \quad \text{in } \Omega,$$

$$(3b) \quad \nabla \cdot \delta \mathbf{u} = \nabla \cdot \mathbf{u}_i \quad \text{in } \Omega,$$

$$(3c) \quad \delta \mathbf{u} = 0 \quad \text{on } \Gamma_1,$$

$$(3d) \quad -(\delta p) \mathbf{n} + \nu \frac{\partial(\delta \mathbf{u})}{\partial \mathbf{n}} = \mathbf{0} \quad \text{on } B_{eo}.$$

当たり前のことであるが、 $\delta \mathbf{u}$, δp についての線形方程式である。

最初の 2 つに試験関数 (それぞれ \mathbf{v} と q) をかけて積分すると

$$\begin{aligned} & \int_{\Omega} [((\delta \mathbf{u} \cdot \nabla) \mathbf{u}_i) \cdot \mathbf{v} + ((\mathbf{u}_i \cdot \nabla) \delta \mathbf{u}) \cdot \mathbf{v} - \nu \Delta(\delta \mathbf{u}) \cdot \mathbf{v} + \nabla(\delta p) \cdot \mathbf{v}] dx, \\ & \quad = \int_{\Omega} ((\mathbf{u}_i \cdot \nabla) \mathbf{u}_i - \nu \Delta \mathbf{u}_i + \nabla p_i) \cdot \mathbf{v} dx \\ & \int_{\Omega} q (\nabla \cdot (\delta \mathbf{u})) dx = 0. \end{aligned}$$

11.4.7 Newton 反復の弱形式 (続き)

それぞれ部分積分して (境界積分を消去するのに、(3d) を用いる)

$$(4a) \quad \int_{\Omega} [((\delta \mathbf{u} \cdot \nabla) \mathbf{u}_i) \cdot \mathbf{v} + ((\mathbf{u}_i \cdot \nabla) \delta \mathbf{u}) \cdot \mathbf{v} - \nu \nabla(\delta \mathbf{u}) : \nabla \mathbf{v} - \delta p(\nabla \cdot \mathbf{v})] dx,$$
$$= \int_{\Omega} ((\mathbf{u}_i \cdot \nabla) \mathbf{u}_i - \nu \Delta \mathbf{u}_i + \nabla p_i) \cdot \mathbf{v} dx$$

$$(4b) \quad \int_{\Omega} q(\nabla \cdot (\delta \mathbf{u})) dx = 0.$$

これが弱形式である。

11.4.8 プログラム中の弱形式解読

次のようなマクロが定義されている。

```
Grad(u1,u2) [dx(u1),dy(u1),dx(u2),dy(u2)]
```

11.4.8 プログラム中の弱形式解読

次のようなマクロが定義されている。

```
Grad(u1,u2) [dx(u1),dy(u1),dx(u2),dy(u2)]
```

これは $\nabla \mathbf{u}$ を 4次元ベクトルにしたものである。

11.4.8 プログラム中の弱形式解読

次のようなマクロが定義されている。

$$\text{Grad}(u1, u2) \quad [\text{dx}(u1), \text{dy}(u1), \text{dx}(u2), \text{dy}(u2)]$$

これは $\nabla \mathbf{u}$ を 4次元ベクトルにしたものである。これから

$$\text{Grad}(du1, du2)' * \text{Grad}(v1, v2)$$

は

$$\text{dx}(du1) * \text{dx}(v1) + \text{dy}(du1) * \text{dy}(v1) + \text{dx}(du2) * \text{dx}(v2) + \text{dy}(du2) * \text{dy}(v2)$$

となる。これは $\nabla(\delta \mathbf{u}) : \nabla \mathbf{v}$ である。

11.4.8 プログラム中の弱形式解読

次のようなマクロが定義されている。

```
Grad(u1,u2) [dx(u1),dy(u1),dx(u2),dy(u2)]
```

これは $\nabla \mathbf{u}$ を 4次元ベクトルにしたものである。これから

```
Grad(du1,du2)'*Grad(v1,v2)
```

は

```
dx(du1)*dx(v1)+dy(du1)*dy(v1)+dx(du2)*dx(v2)+dy(du2)*dy(v2)
```

となる。これは $\nabla(\delta \mathbf{u}) : \nabla \mathbf{v}$ である。

注 FreeFem++ では、' は転置を表す (MATLAB でも使われる、古い記法の利用)。

ベクトルの内積

```
real[int] a=[1,2,3,4];  
real[int] b=[-4,-3,-2,-1];  
cout << a'*b << endl;
```

これで -20 という値が表示される。

11.4.8 プログラム中の弱形式解読 (続き)

一方

$$\text{UgradV}(u1, u2, v1, v2) \quad [[u1, u2]' * [dx(v1), dy(v1)], [u1, u2]' * [dx(v2), dy(v2)]]$$

は

11.4.8 プログラム中の弱形式解読 (続き)

一方

$$\text{UgradV}(u_1, u_2, v_1, v_2) \quad [[u_1, u_2]' * [dx(v_1), dy(v_1)], [u_1, u_2]' * [dx(v_2), dy(v_2)]]$$

は

$$\begin{pmatrix} u_1 * dx(v_1) + u_2 * dy(v_1) \\ u_1 * dx(v_2) + u_2 * dy(v_2) \end{pmatrix},$$

すなわち

$$(\mathbf{u} \cdot \nabla) \mathbf{v} = \left(u_1 \frac{\partial}{\partial x_1} + u_2 \frac{\partial}{\partial x_2} \right) \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} u_1 \frac{\partial v_1}{\partial x_1} + u_2 \frac{\partial v_1}{\partial x_2} \\ u_1 \frac{\partial v_2}{\partial x_1} + u_2 \frac{\partial v_2}{\partial x_2} \end{pmatrix}$$

を意味する。

11.4.8 プログラム中の弱形式解読 (続き)

一方

$$\text{UgradV}(u1, u2, v1, v2) \quad [[u1, u2]' * [dx(v1), dy(v1)], [u1, u2]' * [dx(v2), dy(v2)]]$$

は

$$\begin{pmatrix} u1 * dx(v1) + u2 * dy(v1) \\ u1 * dx(v2) + u2 * dy(v2) \end{pmatrix},$$

すなわち

$$(\mathbf{u} \cdot \nabla) \mathbf{v} = \left(u_1 \frac{\partial}{\partial x_1} + u_2 \frac{\partial}{\partial x_2} \right) \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} u_1 \frac{\partial v_1}{\partial x_1} + u_2 \frac{\partial v_1}{\partial x_2} \\ u_1 \frac{\partial v_2}{\partial x_1} + u_2 \frac{\partial v_2}{\partial x_2} \end{pmatrix}$$

を意味する。ゆえに

$$\text{UgradV}(du1, du2, u1, u2)' * [v1, v2]$$

は $((\delta \mathbf{u} \cdot \nabla) \mathbf{u}) \cdot \mathbf{v}$ を、

$$\text{UgradV}(u1, u2, du1, du2)' * [v1, v2]$$

は $((\mathbf{u} \cdot \nabla) (\delta \mathbf{u})) \cdot \mathbf{v}$ を計算する。

11.4.8 プログラム中の弱形式解読 (続き)

```
int2d(Th) (  
    nu*(Grad(du1,du2)')*Grad(v1,v2))  
+ UgradV(du1,du2, u1, u2) '*[v1,v2]  
+ UgradV( u1, u2,du1,du2) '*[v1,v2]  
- div(du1,du2)*q - div(v1,v2)*dp  
- 1e-8*dp*q // stabilization term  
)
```

は次式を意味する。

$$\int_{\Omega} [\nu \nabla(\delta \mathbf{u}) : \nabla \mathbf{v} + ((\delta \mathbf{u} \cdot \nabla) \mathbf{u}) \cdot \mathbf{v} + ((\mathbf{u} \cdot \nabla)(\delta \mathbf{u})) \cdot \mathbf{v} - q \nabla \cdot (\delta \mathbf{u}) - \delta p \nabla \cdot \mathbf{v} - 10^{-8}(\delta p)q] dx$$

```
- int2d(Th) (  
    nu*(Grad(u1,u2)')*Grad(v1,v2))  
+ UgradV(u1,u2, u1, u2) '*[v1,v2]  
- div(u1,u2)*q - div(v1,v2)*p  
- 1e-8*p*q  
)
```

は次式を意味する。

$$- \int_{\Omega} [\nu \nabla \mathbf{u}_i : \nabla \mathbf{v} + ((\mathbf{u}_i \cdot \nabla) \mathbf{u}_i) \cdot \mathbf{v} - q \nabla \cdot \mathbf{u}_i - p_i \nabla \cdot \mathbf{v} - 10^{-8} p_i q] dx$$

11.4.9 初期値の選択

最初に $\nu = \frac{1}{50}$ について、

$$\mathbf{u}_0(x, y) = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

を Newton 法の初期値として Navier-Stokes 方程式の解を求め、それを ν を小さくした Navier-Stokes 方程式に対する Newton 法の初期値としている。

11.4.9 初期値の選択

最初に $\nu = \frac{1}{50}$ について、

$$\mathbf{u}_0(x, y) = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

を Newton 法の初期値として Navier-Stokes 方程式の解を求め、それを ν を小さくした Navier-Stokes 方程式に対する Newton 法の初期値としている。

最終的には、 $\nu = \frac{1}{200}$ の場合の Navier-Stokes 方程式の解を求めている。

11.4.9 初期値の選択

最初に $\nu = \frac{1}{50}$ について、

$$\mathbf{u}_0(x, y) = \begin{cases} (1, 0)^\top & (x^2 + y^2 > 2) \\ (0, 0)^\top & (x^2 + y^2 \leq 2) \end{cases}$$

を Newton 法の初期値として Navier-Stokes 方程式の解を求め、それを ν を小さくした Navier-Stokes 方程式に対する Newton 法の初期値としている。

最終的には、 $\nu = \frac{1}{200}$ の場合の Navier-Stokes 方程式の解を求めている。

Newton 法の収束が早ければ、 ν をより速く小さくしたり、Newton 法が収束しなければ、 ν を少し (戻して) 大きくしたり、細かい制御を行っている。余裕があればプログラムを読んでみよう。

それが筋が通ったものであるか分からないが「そこまでやるか」と感心した。こういうプログラムを公開してもらえるのは、大変ありがたいことである。

11.4.10 余談: Oseen 方程式

NSNewton.edp の弱形式を定義しているところで `solve Oseen()` と書いてある。

11.4.10 余談: Oseen 方程式

NSNewton.edp の弱形式を定義しているところで `solve Oseen()` と書いてある。

オゼーン

Oseen 方程式は、Navier-Stokes 方程式を一様な定常流の周りで線形近似したものである。

11.4.10 余談: Oseen 方程式

NSNewton.edp の弱形式を定義しているところで `solve Oseen()` と書いてある。

オゼーン

Oseen 方程式は、Navier-Stokes 方程式を一様な定常流の周りで線形近似したものである。

無限遠で小さいが 0 ではない、一様な速度 $U\mathbf{e}_x = (U, 0, 0)^\top$ を持つ流れを扱いたい。

$$\mathbf{v} := \mathbf{u} - U\mathbf{e}_x$$

とおく (\mathbf{v} は $\delta\mathbf{u}$ と書くのが良いかもしれない)。

11.4.10 余談: Oseen 方程式

NSNewton.edp の弱形式を定義しているところで `solve Oseen()` と書いてある。

オゼーン Oseen 方程式は、Navier-Stokes 方程式を一様な定常流の周りで線形近似したものである。

無限遠で小さいが 0 ではない、一様な速度 $U\mathbf{e}_x = (U, 0, 0)^\top$ を持つ流れを扱いたい。

$$\mathbf{v} := \mathbf{u} - U\mathbf{e}_x$$

とおく (\mathbf{v} は $\delta\mathbf{u}$ と書くのが良いかもしれない)。このとき

$$(\mathbf{u} \cdot \nabla) \mathbf{u} = \left((U + v_1) \frac{\partial}{\partial x_1} + v_2 \frac{\partial}{\partial x_2} + v_3 \frac{\partial}{\partial x_3} \right) \begin{pmatrix} U + v_1 \\ v_2 \\ v_3 \end{pmatrix} = U \frac{\partial \mathbf{v}}{\partial x_1} + (\mathbf{v} \cdot \nabla) \mathbf{v} \doteq U \frac{\partial \mathbf{v}}{\partial x_1}.$$

そこで近似方程式として

$$(5) \quad \frac{\partial \mathbf{v}}{\partial t} + U \frac{\partial \mathbf{v}}{\partial x_1} = -\frac{1}{\rho} \nabla p + \nu \Delta \mathbf{v}$$

の採用が考えられる。この方程式が普通 Oseen 方程式と呼ばれる (と私は習った)。

Wikipedia 等を見ると、もう少し意味を広げて解釈されることもあるようである。しかし、Hecht のプログラムで、Oseen と言っているのは、かなりの拡張解釈であろう。

“Oseen 近似” という検索語でヒットした玉田 [3] は、Oseen 方程式が研究された経緯が分かる解説である (と私は感じた)。

(工事中)

説明がイマイチのところがあり、一部は式レベルでミスをしている気がする。時間が取れしだい直す。今は要点のみ以下に示す。

例えば、 A が線形作用素、 a が双線形作用素とすると、

$$f(\mathbf{u}) = \begin{pmatrix} A\mathbf{u} - \mathbf{b} \\ a(\mathbf{u}, \mathbf{u}) \end{pmatrix}$$

の微分は

$$f'(\mathbf{u})\mathbf{v} = \begin{pmatrix} A\mathbf{v} \\ a(\mathbf{u}, \mathbf{v}) + a(\mathbf{v}, \mathbf{u}) \end{pmatrix}.$$

これを言っておくと、計算の見通しが良くなる。定理に格上げしておいてそれを使って議論すると、説明がかなり短縮されそうである。

参考文献

- [1] 桂田祐史：非線形方程式、特に代数方程式の数値解法,
<http://nalab.mind.meiji.ac.jp/~mk/labo/text/nonlinear-algebraic.pdf> (2007/3/24～).
- [2] Hecht, F.: Freefem++,
<https://doc.freefem.org/pdf/FreeFEM-documentation.pdf>, 以前は <http://www3.freefem.org/ff++/ftp/freefem++doc.pdf> にあった。
- [3] 玉田琮：近似解法：Oseen 近似, 境界層近似等について, 京都大学数理解析研究所講究録, Vol. 52, pp. 37–52 (1968), <https://repository.kulib.kyoto-u.ac.jp/dspace/handle/2433/107760>.