二次元多重連結領域内における
構造安定な非圧縮流れの木表現の可視化手法

2016SE024 亀谷拓磨
2016SE076 田島嘉人
2016SE090 渡辺康平
指導教員: 横山哲郎

目次

流体力学のあらまし

・離散解析の概要

本研究の背景

・木表現のメリット・デメリット

本研究の目的

目的達成の為に

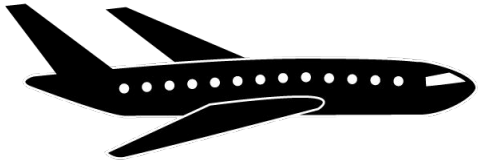
・Asymptote・pythonを用いたソフトウェアの作成

流体力学のあらまし

流体力学とは…？

気体や液体の運動を取り扱う力学において主要な研究分野

なぜボールは曲がるのか？



なぜ飛行機は空を飛ぶのか？

流体力学における離散解析

離散解析とは…？

ある流れから、**トポロジー**の考え方を利用し**本質的な構造**を探し出す

一見すると複雑な形の流れを単純な構造に変換できる

・トポロジー

連続的に変形可能な図形を同じ形と考える

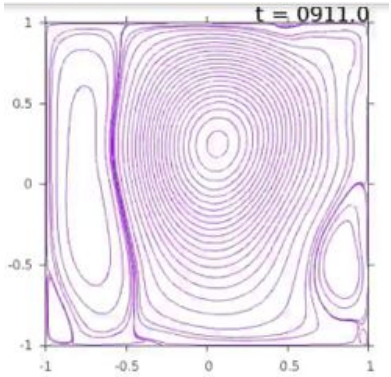
（例  と  は同じ形

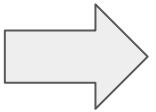
離散解析：流れの木表現

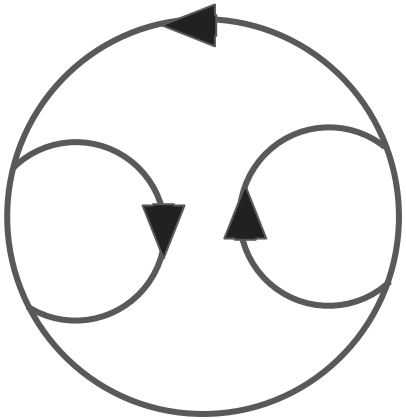
流れの木表現とは…？

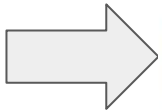
二次元多重連結領域上の非圧縮流に適用可能な離散解析の手法

流れを木表現に変換する過程









$$b_{\emptyset+}\ (1,\{\mathrm{cons}(c\_(1,\lambda),\mathrm{cons}(c\_(1,\ \lambda),\lambda))\})$$

実際の流れ　　トポロジー　　木表現

離散解析と流れの木表現のメリット

なぜ離散解析…？

細かい流れを無視した、大雑把な見方

・流れの**本質的な構造**のみに着目する

大まかな構造にのみ着目し解析する場合の**効率が良い**

扱うデータ量が少ないため**計算量が少ない**

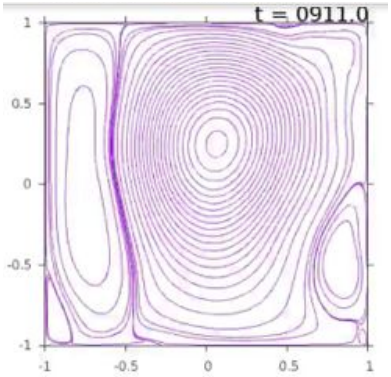
なぜ木表現…？

・すべての二次元上の非圧縮流を数学的に**厳密に分類可能**

・流れを代数的に表現できる

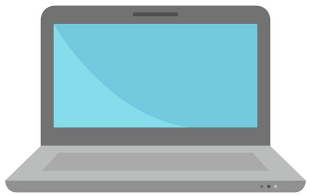
流れの構造の特徴を説明する**共通言語**

研究背景



変換

$b_{\emptyset+}\ (1,\{\mathrm{cons}(c\_(1,\lambda),\mathrm{cons}(c\_(1,\ \lambda),\lambda))\})$



理解

木表現からの直感的理解の難しさ

目的: 二次元流れの木表現から図への変換の自動化

木表現

- ○ 計算機での扱いが**容易**
  - ・流線の代数的構造の表現
- × 形状の把握が困難

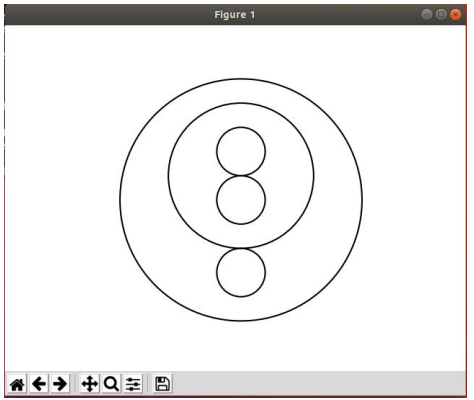
  例: $b_{++}b_{++}\{b_{++}\{I,I\},I\}$

自動変換

図

- × 計算機での扱いが煩雑
- ○ 形状を直観的に**把握可能**

  例:



※ 変換の自動化 ⇒ **変換ミスの防止**

流線図に求められる条件

1. 見やすさ

線が重ならない
線を滑らかに描く
線が適切な距離を保つ

2. 正確性

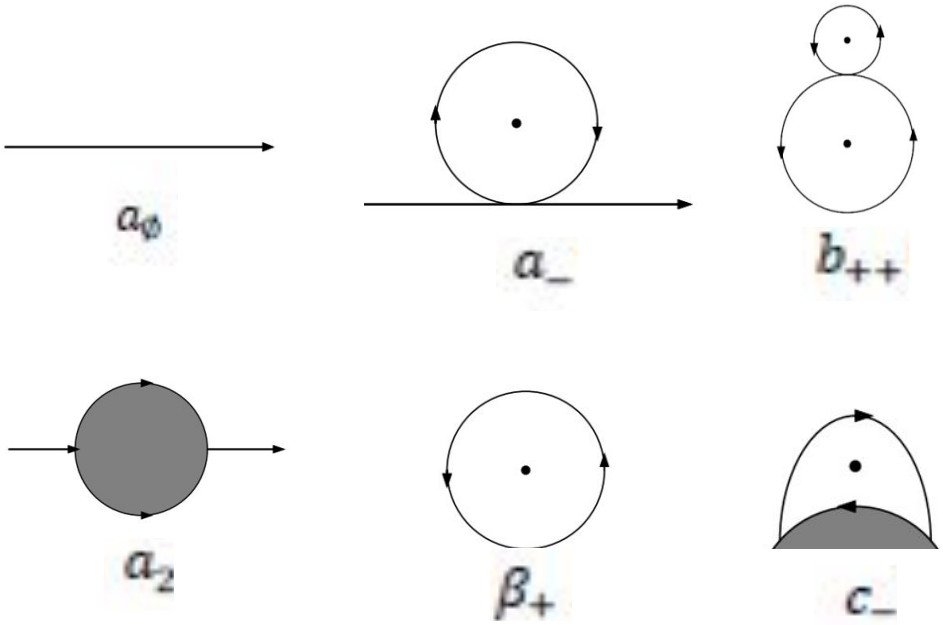
木表現と同じトポロジーを
正確に表現

目的

以上を満たす木表現と同じトポロジーを
表現するプログラムの作成

木文法

木文法の生成規則に準じて流線図を生成



流れの生成規則

S → $a_\emptyset\ (A^*) \mid b_{\emptyset+}\ (B_+,\{C_-^*\}) \mid b_{\emptyset-}\ (B_-,\{C_+^*\})$

A → $a_+(B_+) \mid a_-(B_-) \mid a_2(C_+^*, C_-^*)$

A^* → $1 \mid \mathrm{cons}(\mathrm{A}, A^*)$

B_+ → $1 \mid b_{++}(B_+,B_+) \mid b_{+-}\ \{B_+,B_-\} \mid \beta_+\ \{C_+^*\}$

B_- → $1 \mid b_{--}(B_-, B_-) \mid b_{-+}\ \{B_-,B_+\} \mid \beta_-\ \{C_-^*\}$

C_+ → $c_+(B_+,C_-^*)$

C_- → $c_-(B_-,C_+^*)$

C_+^* → $\lambda \mid \mathrm{cons}(C_+,C_+^*)$

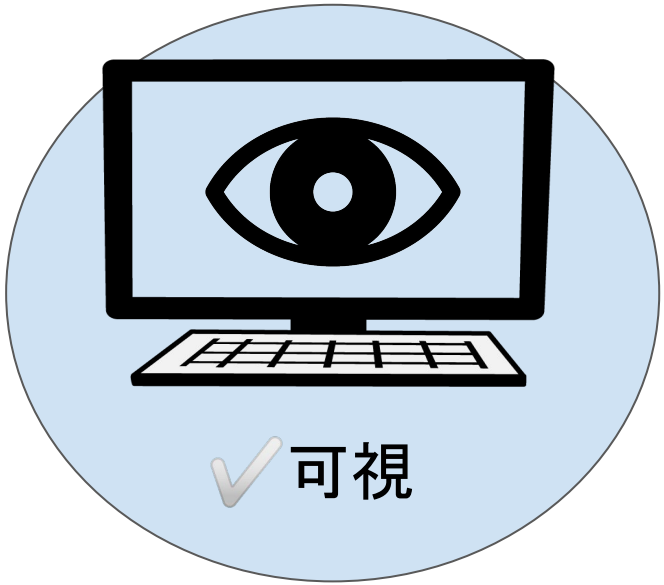
C_-^* → $\lambda \mid \mathrm{cons}(C_-,C_-^*)$

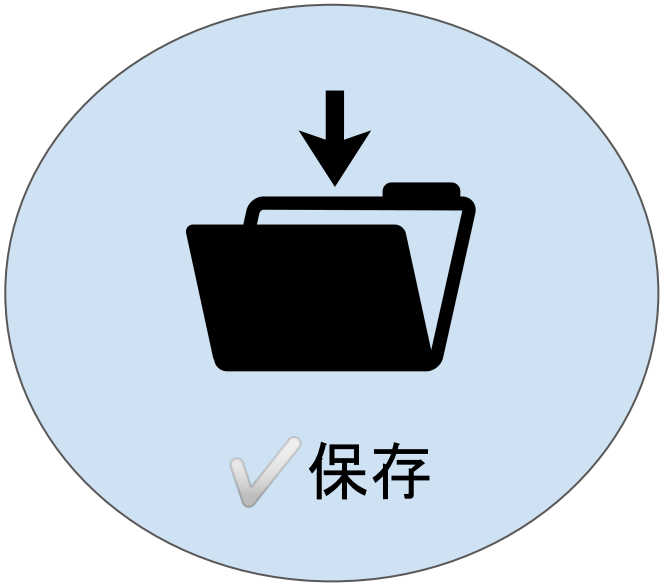
＋：反時計回り
－：時計回り

参考文献[2]より

アプローチ

作成した流線図が・・・

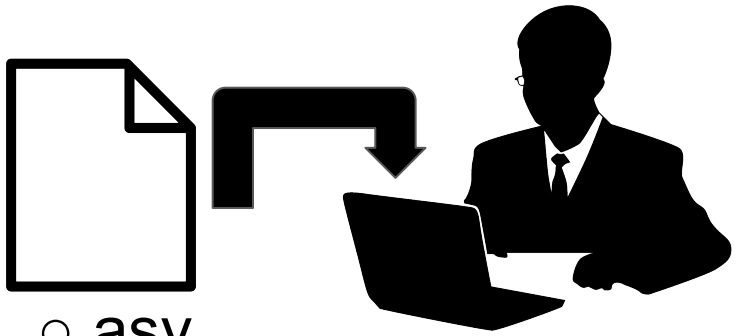




以上を可能とする**プログラミング言語**

Asymptote

1. ライブラリを提供

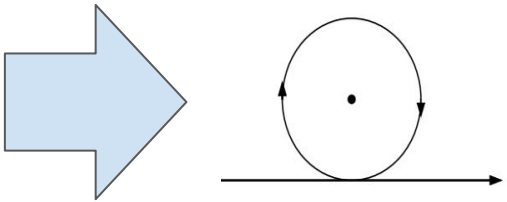


○.asy

2. 目的の達成

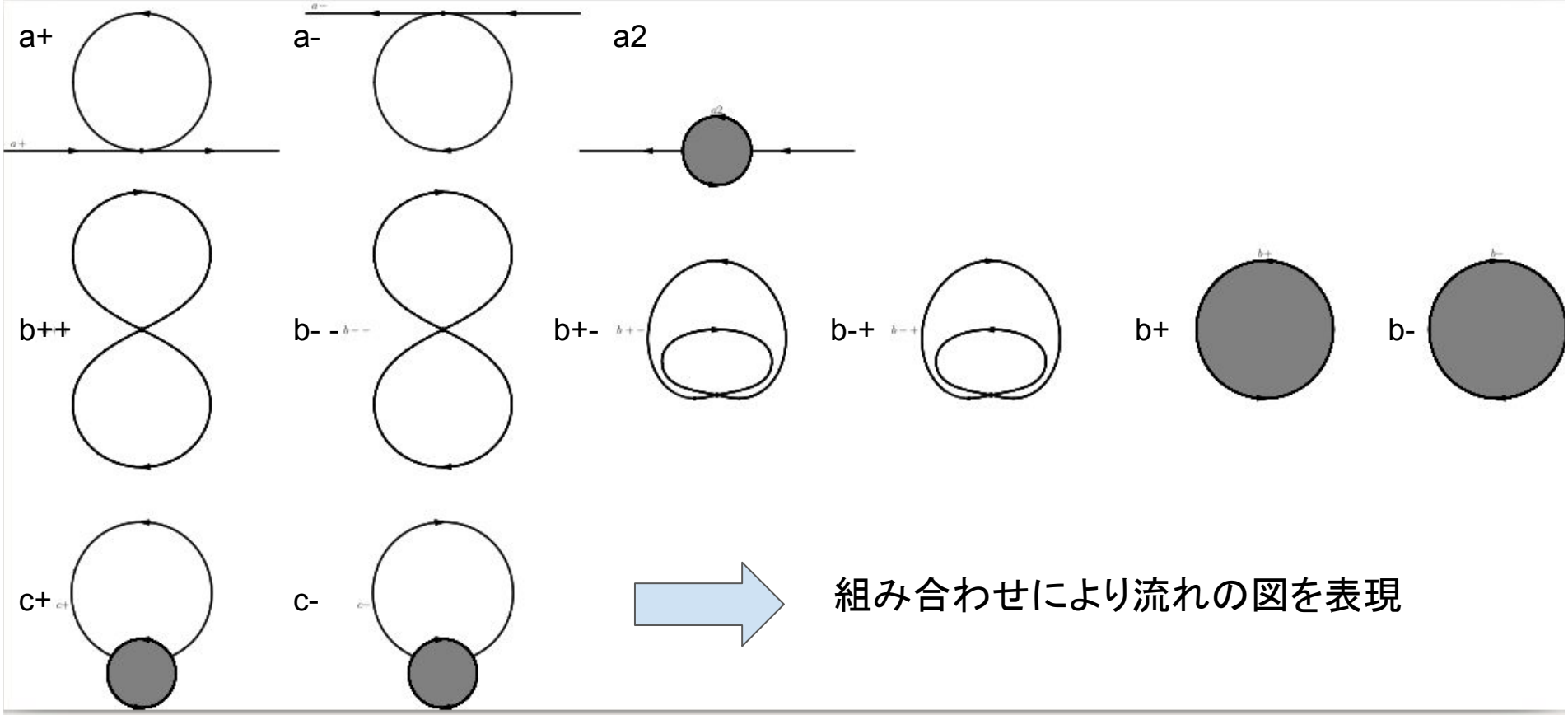
a-

木表現



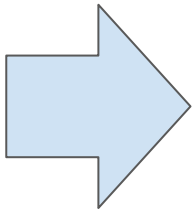
図

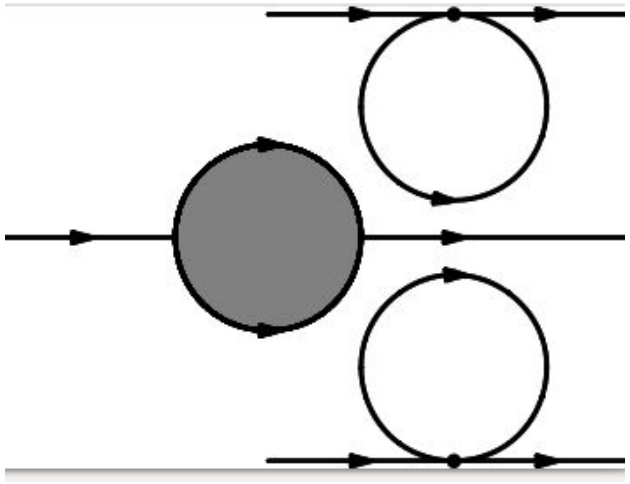
提供するライブラリの図（Asymptote）



実行例

例）カルマン渦



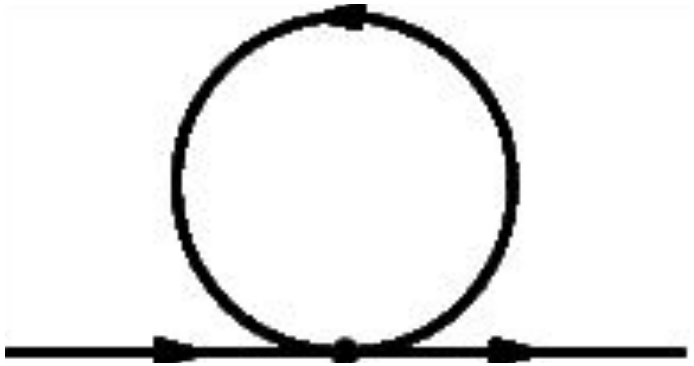


実際に存在する流れを再現することも可能

【今回提供したライブラリ関数について】

例）void apRU(real d, pair s)
①　②③④　⑤　⑥

①：void関数
②：a+を表す（マイナスはm）
③：右方向（Right）を表す
④：上向き（Up）を表す
⑤：大きさを表す
⑥：淀み点の座標を表す

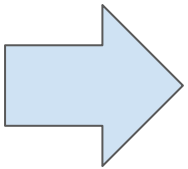


void apRU(5 , (0 , 0))

結果（Asymptote）

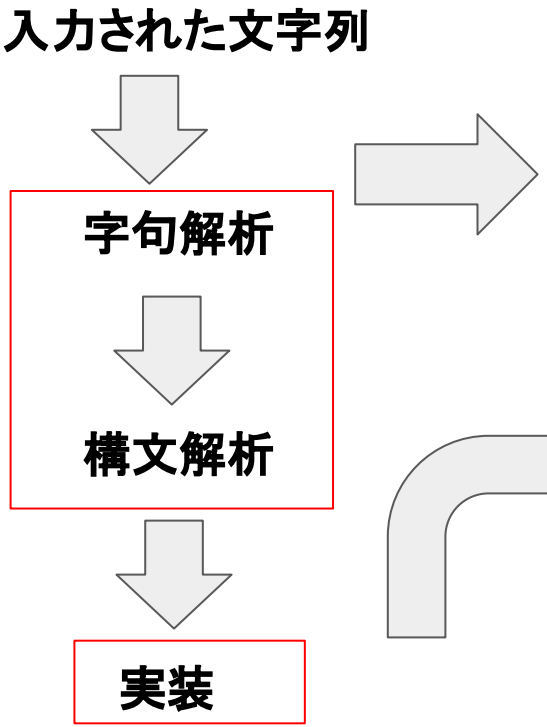
・パラメータの調節、プログラムの組み合わせを駆使することで流れの図を作成　することとができた

・Asymptote内では入力を完全な木表現で図を作成する段階までに至らなかった



他の言語内で抽象的に書くことにより描画できる可能性あり

pythonの実装方法



PLY（pythonライブラリ）で作成

インタプリタパターンを利用

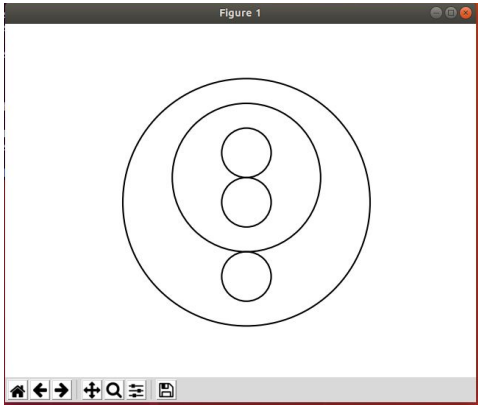
メリット

- プログラムの変更が容易
- テストが容易
- 手順に則った処理が可能

実装：流線図の描画方法

matplotlibを利用して描画する

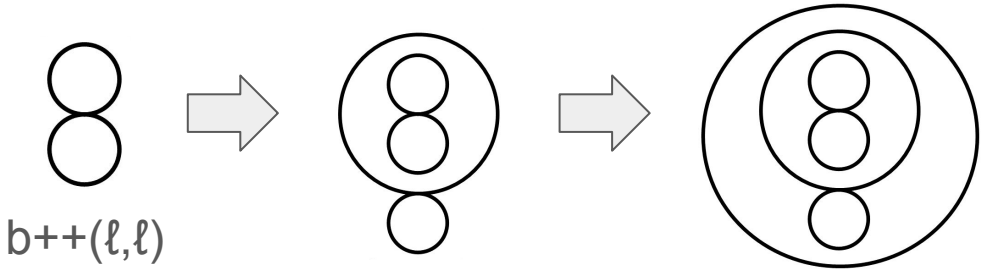
python用のグラフ描画ライブラリ



図：作成中のプログラムで描画した流線

描画の考え方

例．a+(b++(b++(ℓ,ℓ),ℓ)))



b++(ℓ,ℓ)

b++(b++(ℓ,ℓ),ℓ)

a+(b++(b++(ℓ,ℓ),ℓ)))

葉から図を作成することで

条件1

線が重ならない
線を滑らかに描く
線が適切な距離を保つ

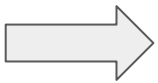
を満たす図を描画できる

課題

主にC系の流れにループが存在し、その部分の要素は無限に増やすことが可能である。そのため、要素数や要素の種類に合わせ柔軟に生成される図を変更できることが求められる。

・物理モデル

主にグラフ描画の際に用いられる技術。グラフの要素同士に物理演算を用いることによってグラフの見た目を整える。



・C系の流れをグラフとしてモデル化し、物理モデルで位置を整え描画する
・局所的な物理モデルの適用

まとめ

pythonを用いた手法では、最初に定めた条件を満たすことができる描画方法を発見した。

しかし、C系の表現において条件を満たすためには工夫が必要である。

今後の課題

・プログラムの完成

・作成したプログラムが目的の条件を満たしているか調査

・作成したプログラムの実際の流体への応用

参考文献

[1] 坂上貴之, 横山知郎, 澤村陽一: 二次元多重連結領域内における構造安定な非圧縮流れの文字列表現アルゴリズム:数理解析研究所講究録 , Vol.1900, pp.11–25(2014).

[2] 横山哲郎, 横山知郎: 多重連結領域上の非圧縮流を表す木文法の深化 , プレプリント.

[3] 加藤舞, 内藤綾香: 多重連結領域上の安定非圧縮流の解析 , 南山大学2018年度卒業論文(2019)

[4] 荒井迅: トポロジカルな流れ構造の理解へ向けて , 日本流体力学会誌 , Vol.33, pp.23–28(2014) (仮)