精度保証付き数値計算のアイディアお教えします

渡部 善隆

watanabe@cc.kyushu-u.ac.jp

九州大学情報基盤センター

Numerical Computation

Numerical Computation

$\boxed{\text{現象}}$

Numerical Computation

$\boxed{\text{現象}}$

$\Downarrow$ $\quad\longleftarrow$ モデル化

$\boxed{\text{数理モデル}}$

Numerical Computation

現象

$\Downarrow$ $\longleftarrow$ モデル化

数理モデル

$\Downarrow$ $\longleftarrow$ 離散化

計算機モデル

Numerical Computation

現象

⇓ ←— モデル化

数理モデル

⇓ ←— 離散化

計算機モデル

⇓ ←— 線形近似・反復

プログラム

Numerical Computation

現象

$\Downarrow$ $\longleftarrow$ モデル化

数理モデル

$\Downarrow$ $\longleftarrow$ 離散化

計算機モデル

$\Downarrow$ $\longleftarrow$ 線形近似・反復

プログラム

$\Downarrow$ $\longleftarrow$ 計算

数値の解析／可視化

Numerical Computation (example)

Numerical Computation (example)

流れ現象

Numerical Computation (example)

$\boxed{\text{流れ現象}}$

$\Downarrow$ $\longleftarrow$ モデル化

$$\boxed{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}}$$

Numerical Computation (example)

$$\boxed{\text{流れ現象}}$$

$$\Downarrow \qquad \longleftarrow \text{モデル化}$$

$$\boxed{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}}$$

$$\Downarrow \qquad \longleftarrow \text{離散化}$$

$$\boxed{F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n}$$

Numerical Computation (example)

$\boxed{\text{流れ現象}}$

$\Downarrow$ ⟵ モデル化

$$\boxed{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}}$$

$\Downarrow$ ⟵ 離散化

$$\boxed{F(\boldsymbol{x}) = \mathbf{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n}$$

$\Downarrow$ ⟵ 線形近似・反復

$$\boxed{\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})}$$

Numerical Computation (example)

$$\boxed{\text{流れ現象}}$$

$$\Downarrow \qquad \longleftarrow \text{モデル化}$$

$$\boxed{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}}$$

$$\Downarrow \qquad \longleftarrow \text{離散化}$$

$$\boxed{F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n}$$

$$\Downarrow \qquad \longleftarrow \text{線形近似・反復}$$

$$\boxed{\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})}$$

$$\Downarrow \qquad \longleftarrow \text{計算}$$

$$\boxed{\boldsymbol{x}_N = (1.223, 2.342, \ldots)}$$

‘Error’

流れ現象

$$\Downarrow$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}$$

$$\Downarrow$$

$$F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n$$

$$\Downarrow$$

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})$$

$$\Downarrow$$

$$\boldsymbol{x}_N = (1.223, 2.342, \ldots)$$

‘Error’

流れ現象

$$\Downarrow$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}$$

$\longleftarrow$ モデル化誤差

$$\Downarrow$$

$$F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n$$

$$\Downarrow$$

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})$$

$$\Downarrow$$

$$\boldsymbol{x}_N = (1.223, 2.342, \ldots)$$

‘Error’

$$\boxed{\text{流れ現象}}$$

$$\Downarrow \qquad \longleftarrow \text{モデル化誤差}$$

$$\boxed{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}}$$

$$\Downarrow \qquad \longleftarrow \text{離散化誤差}$$

$$\boxed{F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n}$$

$$\Downarrow$$

$$\boxed{\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})}$$

$$\Downarrow$$

$$\boxed{\boldsymbol{x}_N = (1.223, 2.342, \ldots)}$$

‘Error’

流れ現象

$$\Downarrow \qquad \longleftarrow \text{モデル化誤差}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} = -\nabla p + \nu\Delta\boldsymbol{u}$$

$$\Downarrow \qquad \longleftarrow \text{離散化誤差}$$

$$F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n$$

$$\Downarrow \qquad \longleftarrow \text{打ち切り誤差}$$

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1}F(\boldsymbol{x}_{n-1})$$

$$\Downarrow$$

$$\boldsymbol{x}_N = (1.223, 2.342, \ldots)$$

‘Error’

$$\boxed{\text{流れ現象}}$$

$$\Downarrow \qquad \longleftarrow \text{モデル化誤差}$$

$$\boxed{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}}$$

$$\Downarrow \qquad \longleftarrow \text{離散化誤差}$$

$$\boxed{F(\boldsymbol{x}) = \boldsymbol{0} \quad \boldsymbol{x} \in \boldsymbol{R}^n}$$

$$\Downarrow \qquad \longleftarrow \text{打ち切り誤差}$$

$$\boxed{\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})}$$

$$\Downarrow \qquad \longleftarrow \text{丸め誤差}$$

$$\boxed{\boldsymbol{x}_N = (1.223, 2.342, \ldots)}$$

Discretization Error

無限次元の問題を有限次元化した際に生じるギャップ

例) Emden’s equation:

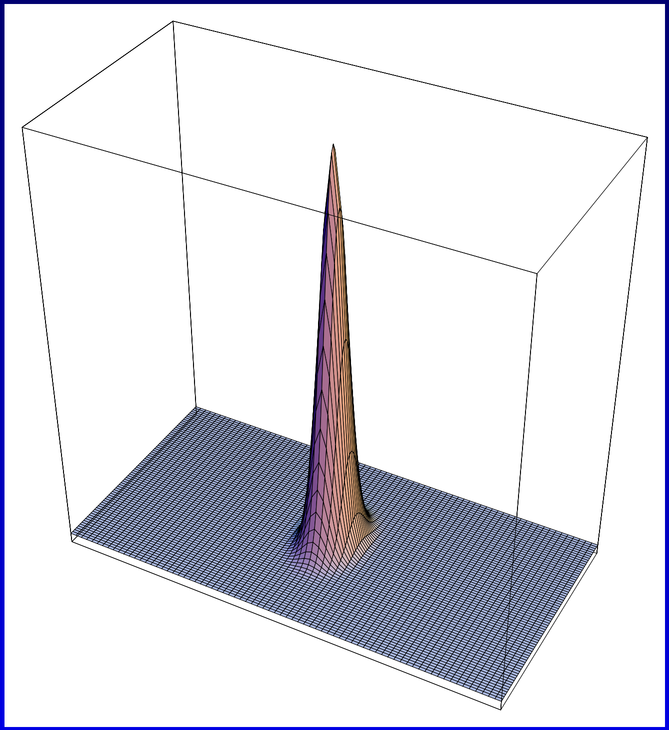
$$
\begin{cases}
-\Delta u = u^2 & \text{in} \quad \Omega_\xi = (0,\xi) \times (0, 1/\xi), \\
u = 0 & \text{on} \quad \partial\Omega_\xi.
\end{cases}
$$

Discretization Error

無限次元の問題を有限次元化した際に生じるギャップ

例) Emden’s equation:

$$\begin{cases} -\Delta u = & u^2 \quad \text{in} \quad \Omega_\xi = (0,\xi) \times (0, 1/\xi), \\ u = & 0 \quad \text{on} \quad \partial\Omega_\xi. \end{cases}$$



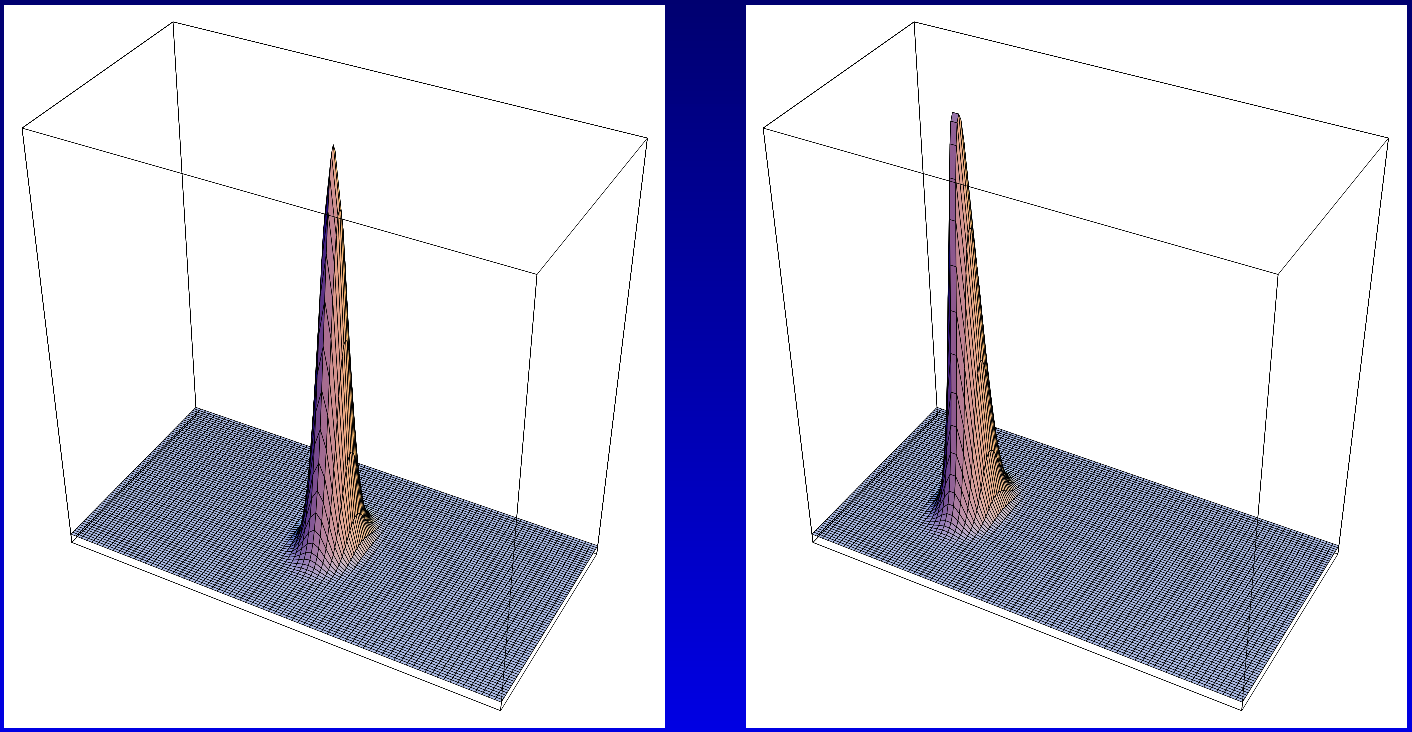
symmetric solution

Discretization Error

無限次元の問題を有限次元化した際に生じるギャップ

例) Emden's equation:

$$\begin{cases} -\Delta u = u^2 & \text{in} \quad \Omega_\xi = (0,\xi) \times (0, 1/\xi), \\ u = 0 & \text{on} \quad \partial\Omega_\xi. \end{cases}$$



symmetric solution　　“spurious” approximation

Truncation Error

$F(\boldsymbol{x}) = 0, \;\; \boldsymbol{x} \in \boldsymbol{R}^n$ を求める反復計算:

Truncation Error

$F(\boldsymbol{x}) = 0,\ \ \boldsymbol{x} \in \boldsymbol{R}^n$ を求める反復計算:

$$\boldsymbol{x}_0 \longrightarrow \boldsymbol{x}_1 \longrightarrow \boldsymbol{x}_2 \longrightarrow \cdots \longrightarrow \boldsymbol{x}_m$$

Truncation Error

$F(\boldsymbol{x}) = 0,\ \ \boldsymbol{x} \in \boldsymbol{R}^n$ を求める反復計算:

$$\boldsymbol{x}_0 \longrightarrow \boldsymbol{x}_1 \longrightarrow \boldsymbol{x}_2 \longrightarrow \cdots \longrightarrow \boldsymbol{x}_m$$

$\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|$ や $\|F(\boldsymbol{x}_m)\|$ が小さいからといって,
必ずしも $\|\boldsymbol{x} - \boldsymbol{x}_m\|$ が小さいとは限らない！

Truncation Error

$F(\boldsymbol{x}) = 0,\ \ \boldsymbol{x} \in \boldsymbol{R}^n$ を求める反復計算:

$$\boldsymbol{x}_0 \longrightarrow \boldsymbol{x}_1 \longrightarrow \boldsymbol{x}_2 \longrightarrow \cdots \longrightarrow \boldsymbol{x}_m$$

$\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|$ や $\|F(\boldsymbol{x}_m)\|$ が小さいからといって,
必ずしも $\|\boldsymbol{x} - \boldsymbol{x}_m\|$ が小さいとは限らない！

例)

100×100 乱数行列の最小特異値を逆反復法により計算

連立 1 次方程式の解法として QMR 法を使用

停止基準 $\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|_2 / \|\boldsymbol{x}_{m-1}\|_2 < 10^{-6}$

Truncation Error

$F(\boldsymbol{x}) = 0,\ \ \boldsymbol{x} \in \boldsymbol{R}^n$ を求める反復計算:

$$\boldsymbol{x}_0 \longrightarrow \boldsymbol{x}_1 \longrightarrow \boldsymbol{x}_2 \longrightarrow \cdots \longrightarrow \boldsymbol{x}_m$$

$\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|$ や $\|F(\boldsymbol{x}_m)\|$ が小さいからといって,
必ずしも $\|\boldsymbol{x} - \boldsymbol{x}_m\|$ が小さいとは限らない！

例)

100×100 乱数行列の最小特異値を逆反復法により計算

連立 1 次方程式の解法として QMR 法を使用

停止基準 $\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|_2 / \|\boldsymbol{x}_{m-1}\|_2 < 10^{-6}$

近似解: 11326919.76359431

Truncation Error

$F(\boldsymbol{x}) = 0,\ \ \boldsymbol{x} \in \boldsymbol{R}^n$ を求める反復計算:

$$\boldsymbol{x}_0 \longrightarrow \boldsymbol{x}_1 \longrightarrow \boldsymbol{x}_2 \longrightarrow \cdots \longrightarrow \boldsymbol{x}_m$$

$\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|$ や $\|F(\boldsymbol{x}_m)\|$ が小さいからといって,
必ずしも $\|\boldsymbol{x} - \boldsymbol{x}_m\|$ が小さいとは限らない！

例)

100×100 乱数行列の最小特異値を逆反復法により計算

連立 1 次方程式の解法として QMR 法を使用

停止基準 $\|\boldsymbol{x}_m - \boldsymbol{x}_{m-1}\|_2 / \|\boldsymbol{x}_{m-1}\|_2 < 10^{-6}$

近似解:	11326919.76359431
真の解:	0.06339481925842853
相対誤差:	0.113×10^8

Rounding Error

計算機で取り扱える実数桁が有限であることから時には計算結果に思わぬ大きな誤差をもたらす.

Rounding Error

計算機で取り扱える実数桁が有限であることから時には計算結果に思わぬ大きな誤差をもたらす.

例) IEEE754 double precision

$$A\boldsymbol{x} = \boldsymbol{b}$$

Rounding Error

計算機で取り扱える実数桁が有限であることから時には計算結果に思わぬ大きな誤差をもたらす.

例) IEEE754 double precision

$$A\boldsymbol{x} = \boldsymbol{b}$$

$$A = \begin{pmatrix} 6491912 & -159018721 \\ 41869520.5 & -102558961 \end{pmatrix}, \quad \boldsymbol{b} = (1, 0)^T$$

$$x_1 = a_{22}/(a_{11}a_{22} - a_{12}a_{21}), \;\; x_2 = -a_{21}/(a_{11}a_{22} - a_{12}a_{21})$$

Rounding Error

計算機で取り扱える実数桁が有限であることから時には計算結果に思わぬ大きな誤差をもたらす.

例) IEEE754 double precision

$$A\boldsymbol{x} = \boldsymbol{b}$$

$$A = \begin{pmatrix} 6491912 & -159018721 \\ 41869520.5 & -102558961 \end{pmatrix}, \quad \boldsymbol{b} = (1, 0)^T$$

$$x_1 = a_{22}/(a_{11}a_{22} - a_{12}a_{21}), \;\; x_2 = -a_{21}/(a_{11}a_{22} - a_{12}a_{21})$$

$$\Downarrow$$

$$\hat{\boldsymbol{x}} = (102558961, 41869520.5)^T$$

Rounding Error

計算機で取り扱える実数桁が有限であることから時には計算結果に思わぬ大きな誤差をもたらす.

例) IEEE754 double precision

$$A\boldsymbol{x} = \boldsymbol{b}$$

$$A = \begin{pmatrix} 6491912 & -159018721 \\ 41869520.5 & -102558961 \end{pmatrix}, \quad \boldsymbol{b} = (1, 0)^T$$

$$x_1 = a_{22}/(a_{11}a_{22} - a_{12}a_{21}), \;\; x_2 = -a_{21}/(a_{11}a_{22} - a_{12}a_{21})$$

$$\Downarrow$$

$$\hat{\boldsymbol{x}} = (102558961, 41869520.5)^T$$

$$\boldsymbol{x} = (205117927, 83739041)^T$$

Numerical Computation

現象

⇓ ⟵ モデル化誤差

数理モデル

⇓ ⟵ 離散化誤差

計算機モデル

⇓ ⟵ 打ち切り誤差

プログラム

⇓ ⟵ 丸め誤差

数値の解析／可視化

Self-validating Numerics 1

数値計算において発生する「誤差」

- 丸めの誤差 `for (i=0; i<N; i++){ s+=a[i]*b[i]; }`
- 打ち切り誤差 $u_0 \to u_1 \to \cdots \to u_n$ ／ stop!
- 離散化誤差 $u''(x) \approx \dfrac{u(x+h) - 2u(x) + u(x-h)}{h^2}$

Self-validating Numerics 1

数値計算において発生する「誤差」

- 丸めの誤差

```
for (i=0; i<N; i++){
      s+=a[i]*b[i]; }
```

- 打ち切り誤差 $u_0 \to u_1 \to \cdots \to u_n$ ／ stop!
- 離散化誤差 $u''(x) \approx \dfrac{u(x+h) - 2u(x) + u(x-h)}{h^2}$

$$\Downarrow$$

レベルの異なる誤差を正確に把握したい

Self-validating Numerics 2

Self-validating Numerics 2

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

Self-validating Numerics 2

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

$\Downarrow$

> 理論的に証明することが困難な解析学上の問題に対する数値的解決としての計算機支援証明へと進展

Self-validating Numerics 2

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

$$\Downarrow$$

> 理論的に証明することが困難な解析学上の問題に対する数値的解決としての計算機支援証明へと進展

《実現技法》

- 区間演算
- Newton 型作用素
- 不動点定理
  - 適切な関数空間の設定
  - 近似解に対する誤差評価の理論

Guide-books

Guide-books

「精度保証付き数値計算」

中尾 充宏，山本 野人

日本評論社, 1998

ISBN4-535-78258-X

Guide-books

「精度保証付き数値計算」

中尾 充宏，山本 野人

日本評論社, 1998

ISBN4-535-78258-X

「精度保証付き数値計算」

大石 進一

コロナ社, 2000

ISBN4-339-02605-0

Take Account of Rounding Error 1

プログラミングの観点から

Take Account of Rounding Error 1

プログラミングの観点から

- 桁数を増やす

  単精度→倍精度，倍精度→ 4 倍精度

Take Account of Rounding Error 1

プログラミングの観点から

- 桁数を増やす
  単精度→倍精度，倍精度→ 4 倍精度
- 丸め誤差の影響を調べる
  コンパイラの精度調整機能

Take Account of Rounding Error 1

プログラミングの観点から

- 桁数を増やす
  単精度→倍精度，倍精度→ 4 倍精度
- 丸め誤差の影響を調べる
  コンパイラの精度調整機能
- 専門家に相談する
  アルゴリズムの改良，手続きの関数化

Take Account of Rounding Error 1

プログラミングの観点から

- 桁数を増やす
  単精度→倍精度，倍精度→4倍精度
- 丸め誤差の影響を調べる
  コンパイラの精度調整機能
- 専門家に相談する
  アルゴリズムの改良，手続きの関数化
- 既存のソフトウェアを使う
  SAS, SPSS, BMDP, SPICE, NASTRAN, Gaussian, etc.

Take Account of Rounding Error 1

プログラミングの観点から

- 桁数を増やす
  単精度→倍精度，倍精度→ 4 倍精度
- 丸め誤差の影響を調べる
  コンパイラの精度調整機能
- 専門家に相談する
  アルゴリズムの改良，手続きの関数化
- 既存のソフトウェアを使う
  SAS, SPSS, BMDP, SPICE, NASTRAN, Gaussian, etc.
- 複数の計算機で実行させてみる
  アーキテクチャ，最適化，デバッグ

Take Account of Rounding Error 2

数学的信頼性を保証するには

Take Account of Rounding Error 2

数学的信頼性を保証するには

- 入力データに注意！

  10 進: 0.2 → 2 進: 0.0011001100110011001100 ...

Take Account of Rounding Error 2

数学的信頼性を保証するには

- 入力データに注意！
  10 進: 0.2 → 2 進: 0.001100110011001100 ...
- 多倍長演算
  任意の桁数を確保
  GNU MP, FMLIB
  誤差評価が必要

Take Account of Rounding Error 2

数学的信頼性を保証するには

- 入力データに注意！

  10 進: 0.2 → 2 進: 0.0011001100110011001100 ...

- 多倍長演算

  任意の桁数を確保

  GNU MP, FMLIB

  誤差評価が必要

- 有理数演算

  Mathematica, MATLAB, Maple, Reduce, etc.

  CALC

Take Account of Rounding Error 2

数学的信頼性を保証するには

- 入力データに注意！

  10 進: 0.2 → 2 進: 0.0011001100110011001100 ...

- 多倍長演算

  任意の桁数を確保

  GNU MP, FMLIB

  誤差評価が必要

- 有理数演算

  Mathematica, MATLAB, Maple, Reduce, etc.

  CALC

- **区間演算**

Rational Number Arithmetic

Rational Number Arithmetic

- 厳密計算

Rational Number Arithmetic

- 厳密計算

$$\frac{1}{100} + \frac{7}{300} \times \frac{3}{28} \quad / \quad \frac{1}{3}$$

Rational Number Arithmetic

- 厳密計算

$$\frac{1}{100} + \frac{7}{300} \times \frac{3}{28} \quad / \quad \frac{1}{3} = \frac{13}{1200}$$

Rational Number Arithmetic

- 厳密計算

$$\frac{1}{100} + \frac{7}{300} \times \frac{3}{28} \;\; / \;\; \frac{1}{3} = \frac{13}{1200}$$

- 計算とともに分母分子が肥大化する

Rational Number Arithmetic

● 厳密計算

$$\frac{1}{100}+\frac{7}{300}\times\frac{3}{28} \;\Big/\; \frac{1}{3}=\frac{13}{1200}$$

● 計算とともに分母分子が肥大化する

《*Mathematica* での計算例》

```
In[1]:= a=0
In[2]:= Do[a=a+i/j,{i,100},{j,100}]
In[3]:= a
```

Rational Number Arithmetic

- 厳密計算

$$\frac{1}{100} + \frac{7}{300} \times \frac{3}{28} \;\; / \;\; \frac{1}{3} = \frac{13}{1200}$$

- 計算とともに分母分子が肥大化する

《*Mathematica*での計算例》

```
In[1]:= a=0
In[2]:= Do[a=a+i/j,{i,100},{j,100}]
In[3]:= a
          365282566057888867955933058837931905470275
Out[3]= -------------------------------------------
          1394407504594249543290676178706246071136
```

Rational Number Arithmetic

- 厳密計算

$$\frac{1}{100}+\frac{7}{300}\times\frac{3}{28} \;/\; \frac{1}{3}=\frac{13}{1200}$$

- 計算とともに分母分子が肥大化する

《*Mathematica* での計算例》

```
In[1]:= a=0
In[2]:= Do[a=a+i/j,{i,100},{j,100}]
In[3]:= a
         36528256605788886679559333058837931905470275
Out[3]= ---------------------------------------------
         1394407504594249543290676178706246071136
```

↑ 工夫が必要

IEEE754 Standard

IEEE754 Standard

$\boldsymbol{F} \subset \boldsymbol{R}$: 計算機で表現できる数, $c \in \boldsymbol{R}$

IEEE754 Standard

$\boldsymbol{F} \subset \boldsymbol{R}$: 計算機で表現できる数, $c \in \boldsymbol{R}$

$\triangle : \boldsymbol{R} \to \boldsymbol{F}$: c 以上の最も小さい $\boldsymbol{F}$ の数への丸め

$\bigtriangledown : \boldsymbol{R} \to \boldsymbol{F}$: c 以下の最も大きい $\boldsymbol{F}$ の数への丸め

IEEE754 Standard

$\boldsymbol{F} \subset \boldsymbol{R}$: 計算機で表現できる数, $c \in \boldsymbol{R}$

$\triangle : \boldsymbol{R} \to \boldsymbol{F}$: c 以上の最も小さい $\boldsymbol{F}$ の数への丸め

$\bigtriangledown : \boldsymbol{R} \to \boldsymbol{F}$: c 以下の最も大きい $\boldsymbol{F}$ の数への丸め

$$* \in \{+,\ -,\ \cdot,\ /\},\ \bigcirc \in \{\triangle, \bigtriangledown\}$$

$$x \circledast y = \bigcirc(x * y) \quad \forall x, y \in \boldsymbol{F}$$

IEEE754 Standard

$\boldsymbol{F} \subset \boldsymbol{R}$: 計算機で表現できる数, $c \in \boldsymbol{R}$

$\triangle : \boldsymbol{R} \to \boldsymbol{F}$: c 以上の最も小さい $\boldsymbol{F}$ の数への丸め

$\bigtriangledown : \boldsymbol{R} \to \boldsymbol{F}$: c 以下の最も大きい $\boldsymbol{F}$ の数への丸め

$* \in \{+,\ -,\ \cdot,\ /\},\ \bigcirc \in \{\triangle, \bigtriangledown\}$

$$x \circledast y = \bigcirc(x * y) \quad \forall x, y \in \boldsymbol{F}$$

計算機での四則演算の結果 $x \circledast y$ は，(数学的に正しい) 実数としての四則演算の結果 $x * y$ に指定された丸めを行なって得られた数 $\bigcirc(x * y)$ に一致

IEEE754 Standard

$\boldsymbol{F} \subset \boldsymbol{R}$: 計算機で表現できる数，$c \in \boldsymbol{R}$

$\triangle : \boldsymbol{R} \to \boldsymbol{F}$: c 以上の最も小さい $\boldsymbol{F}$ の数への丸め

$\bigtriangledown : \boldsymbol{R} \to \boldsymbol{F}$: c 以下の最も大きい $\boldsymbol{F}$ の数への丸め

$$* \in \{+,\ -,\ \cdot,\ /\},\ \bigcirc \in \{\triangle, \bigtriangledown\}$$

$$x \circledast y = \bigcirc(x * y) \quad \forall x, y \in \boldsymbol{F}$$

計算機での四則演算の結果 $x \circledast y$ は，(数学的に正しい) 実数としての四則演算の結果 $x * y$ に指定された丸めを行なって得られた数 $\bigcirc(x * y)$ に一致

$$\Downarrow$$

計算機内の演算結果を評価することが可能

Interval Arithmetic 1

計算機で取り扱える数は実数の近似のため，1つの $\boldsymbol{F}$ の数では表現できない

Interval Arithmetic 1

計算機で取り扱える数は実数の近似のため，1つの $\boldsymbol{F}$ の数では表現できない

$$\Downarrow$$

下界と上界を表す $\boldsymbol{F}$ のペアで丸め誤差を含めた形で実数を表現する

$$[a, b] = \{t \mid a \le t \le b,\ \ a, b \in \boldsymbol{F}\}$$

Interval Arithmetic 1

計算機で取り扱える数は実数の近似のため，１つの $\boldsymbol{F}$ の数では表現できない

$$\Downarrow$$

下界と上界を表す $\boldsymbol{F}$ のペアで丸め誤差を含めた形で実数を表現する

$$[a, b] = \{t \mid a \leq t \leq b,\ \ a, b \in \boldsymbol{F}\}$$

- 区間に含まれる**無限個の実数**を表現できる

Interval Arithmetic 1

計算機で取り扱える数は実数の近似のため，1つの $\boldsymbol{F}$ の数では表現できない

$$\Downarrow$$

下界と上界を表す $\boldsymbol{F}$ のペアで丸め誤差を含めた形で実数を表現する

$$[a, b] = \{t \mid a \leq t \leq b,\ \ a, b \in \boldsymbol{F}\}$$

- 区間に含まれる**無限個の実数**を表現できる
- 有理数演算に対しても同様の区間表現が可能

Interval Arithmetic 2

For intervals $X = [a, b]$, $Y = [c, d] \subset \boldsymbol{R}$ and a operation $* \in \{+,\ -,\ \cdot,\ /\}$,

$$X * Y \equiv \{x * y \mid x \in X,\ y \in Y\}.$$

Interval Arithmetic 2

For intervals $X = [a, b], Y = [c, d] \subset \boldsymbol{R}$ and a operation $* \in \{+,\ -,\ \cdot,\ /\}$,

$$X * Y \equiv \{x * y \mid x \in X,\ y \in Y\}.$$

$$\begin{aligned} X + Y &= [a + c, b + d] \\ X - Y &= [a - d, b - c] \\ X \cdot Y &= [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}] \\ X/Y &= [a, b] \cdot [1/d, 1/c] \qquad (0 \notin Y) \end{aligned}$$

Interval Arithmetic 2

For intervals $X = [a, b], Y = [c, d] \subset \boldsymbol{R}$ and a operation $* \in \{+,\ -,\ \cdot,\ /\}$,

$$X * Y \equiv \{x * y \mid x \in X,\ y \in Y\}.$$

$$\begin{aligned} X + Y &= [a + c, b + d] \\ X - Y &= [a - d, b - c] \\ X \cdot Y &= [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}] \\ X / Y &= [a, b] \cdot [1/d, 1/c] \qquad (0 \notin Y) \end{aligned}$$

$$\begin{aligned} 2/3 \cdot \sqrt{2} &\in [0.6666, 0.6667] \cdot [1.414, 1.415] \\ &\subset [0.9425, 0.9434] \end{aligned}$$

Interval Arithmetic 3

$$f(\xi) = \ \xi^3 - 3\xi + 4\xi + 5, \quad x = [0, 1]$$

Interval Arithmetic 3

$$\begin{aligned} f(\xi) = &\ \xi^3 - 3\xi + 4\xi + 5, \quad x = [0,1] \\ &\Rightarrow f(x) = [-3, 10] \end{aligned}$$

Interval Arithmetic 3

$$f(\xi) = \xi^3 - 3\xi + 4\xi + 5, \quad x = [0, 1] \Rightarrow f(x) = [-3, 10]$$

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \; B = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}, \; C = \begin{pmatrix} [-1, 1] & 0 \\ 0 & [-1, 1] \end{pmatrix}$$

Interval Arithmetic 3

$$f(\xi) = \xi^3 - 3\xi + 4\xi + 5, \quad x = [0, 1]$$
$$\Rightarrow f(x) = [-3, 10]$$

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \; B = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}, \; C = \begin{pmatrix} [-1, 1] & 0 \\ 0 & [-1, 1] \end{pmatrix}$$

$$(AB)C = \begin{pmatrix} 0 & [-1, 1] \\ [-1, 1] & [-1, 1] \end{pmatrix}$$

Interval Arithmetic 3

$$\begin{aligned} f(\xi) = \ & \xi^3 - 3\xi + 4\xi + 5, \quad x = [0,1] \\ & \Rightarrow f(x) = [-3, 10] \end{aligned}$$

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \ B = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}, \ C = \begin{pmatrix} [-1,1] & 0 \\ 0 & [-1,1] \end{pmatrix}$$

$$(AB)C = \begin{pmatrix} 0 & [-1,1] \\ [-1,1] & [-1,1] \end{pmatrix}$$

$$A(BC) = \begin{pmatrix} [-2,2] & [-1,1] \\ [-1,1] & [-1,1] \end{pmatrix}$$

Interval Arithmetic Software

INTLAB	Matlab toolbox `http://www.ti3.tu-harburg.de/~rump/intlab/`
PROFIL	C++ class library `http://www.ti3.tu-harburg.de/knueppel/profil/`
INTLIB	Fortran 90 module `ftp://interval.louisiana.edu/pub/interval_math/Fortran_90_software/`
Sun WorkShop	interval operation on Fortran 95 `http://www.sun.com/forte/fortran/interval/`
Mathematica	`Interval` `http://www.wolfram.co.jp/`

Interval Arithmetic Software

INTLAB	Matlab toolbox http://www.ti3.tu-harburg.de/˜rump/intlab/
PROFIL	C++ class library http://www.ti3.tu-harburg.de/knueppel/profil/
INTLIB	Fortran 90 module ftp://interval.louisiana.edu/pub/interval_math/Fortran90_software/
Sun WorkShop	interval operation on Fortran 95 http://www.sun.com/forte/fortran/interval/
Mathematica	`Interval` http://www.wolfram.co.jp/

算術式・行列演算・初等関数の保証付き計算などは，通常の実数型演算と同じ感覚でプログラミング可能

Interval Arithmetic Software

INTLAB	Matlab toolbox http://www.ti3.tu-harburg.de/˜rump/intlab/
PROFIL	C++ class library http://www.ti3.tu-harburg.de/knueppel/profil/
INTLIB	Fortran 90 module ftp://interval.louisiana.edu/pub/interval_math/Fortran90_software/
Sun WorkShop	interval operation on Fortran 95 http://www.sun.com/forte/fortran/interval/
Mathematica	`Interval` http://www.wolfram.co.jp/

算術式・行列演算・初等関数の保証付き計算などは，通常の実数型演算と同じ感覚でプログラミング可能

```
INTERVAL X(1,2), Y(3.14,3.15);
cout << X*Y;
```

The Idea

- 丸め誤差の制御は区間演算によって可能

$$\pi \in [3.14, 3.15]$$

Take Account of Truncation Error

Take Account of Truncation Error

不動点定理

Take Account of Truncation Error

不動点定理

Brouwer の不動点定理

T: $\boldsymbol{R}^n$ の有界凸閉集合 U 上で定義された連続関数.

$$T(U) \subset U$$

ならば，U の中に不動点 $u = T(u)$ が存在.

Take Account of Truncation Error

不動点定理

Brouwer の不動点定理
T: $\boldsymbol{R}^n$ の有界凸閉集合 U 上で定義された連続関数.

$$T(U) \subset U$$

ならば, U の中に不動点 $u = T(u)$ が存在.

- 縮小写像の原理
- Kantorovich (Newton-Kantorovich)
- Schauder
- Banach

Take Account of Truncation Error

不動点定理 の**条件を検証**する！

Brouwer の不動点定理
T: $\boldsymbol{R}^n$ の有界凸閉集合 U 上で定義された連続関数.

$$T(U) \subset U$$

ならば，U の中に不動点 $u = T(u)$ が存在.

- 縮小写像の原理
- Kantorovich (Newton-Kantorovich)
- Schauder
- Banach

Linear Equations 1

$$A\boldsymbol{x} = \boldsymbol{b} \quad A \in \boldsymbol{R}^{n\times n}, \boldsymbol{b} \in \boldsymbol{R}^n$$

Linear Equations 1

$$A\boldsymbol{x} = \boldsymbol{b} \quad A \in \boldsymbol{R}^{n\times n}, \boldsymbol{b} \in \boldsymbol{R}^{n}$$

Brouwer の不動点定理を適用

Linear Equations 1

$$A\boldsymbol{x} = \boldsymbol{b} \quad A \in \boldsymbol{R}^{n\times n}, \boldsymbol{b} \in \boldsymbol{R}^n$$

Brouwer の不動点定理を適用

Lemma

$R \in \boldsymbol{R}^{n\times n}$, n 次元区間ベクトル $\boldsymbol{X}$

$$R\boldsymbol{b} + \{I - RA\} \cdot \boldsymbol{X} \subset \overset{\circ}{\boldsymbol{X}}$$

($\overset{\circ}{X}$ は集合 X の内点全体)

$\Rightarrow R, A$ は正則行列かつ $\exists^1 \widehat{\boldsymbol{x}} \in \overset{\circ}{\boldsymbol{X}}$ s.t. $A\widehat{\boldsymbol{x}} = \boldsymbol{b}$

Linear Equations 1

$$A\boldsymbol{x} = \boldsymbol{b} \quad A \in \boldsymbol{R}^{n\times n}, \boldsymbol{b} \in \boldsymbol{R}^n$$

Brouwer の不動点定理を適用

Lemma

$R \in \boldsymbol{R}^{n\times n}$, n 次元区間ベクトル $\boldsymbol{X}$

$$R\boldsymbol{b} + \{I - RA\} \cdot \boldsymbol{X} \subset \overset{\circ}{\boldsymbol{X}}$$

($\overset{\circ}{X}$ は集合 X の内点全体)

$\Rightarrow R, A$ は正則行列かつ $\exists^1 \widehat{\boldsymbol{x}} \in \overset{\circ}{\boldsymbol{X}}$ s.t. $A\widehat{\boldsymbol{x}} = \boldsymbol{b}$

R は通常 A の近似逆行列を用いる

Linear Equations 2

Example

$$A = \begin{pmatrix} 33 & 16 & 72 \\ -24 & -10 & -57 \\ -8 & -4 & -17 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} 359 \\ -281 \\ -85 \end{pmatrix}$$

Linear Equations 2

Example

$$A = \begin{pmatrix} 33 & 16 & 72 \\ -24 & -10 & -57 \\ -8 & -4 & -17 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} 359 \\ -281 \\ -85 \end{pmatrix}$$

$R \sim A^{-1}$ obtained by Gaussian elimination

Linear Equations 2

Example

$$A = \begin{pmatrix} 33 & 16 & 72 \\ -24 & -10 & -57 \\ -8 & -4 & -17 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} 359 \\ -281 \\ -85 \end{pmatrix}$$

$R \sim A^{-1}$ obtained by Gaussian elimination

$$\Rightarrow A^{-1}b \in \begin{pmatrix} [-1.001, -0.9999] \\ [1.999, 2.001] \\ [4.999, 5.002] \end{pmatrix}$$

Nonliner Equations $f(x) = 0$

Nonliner Equations $f(x) = 0$

Krawczyk 作用素:

$$K(\boldsymbol{X}) \equiv \boldsymbol{y} - Y\boldsymbol{f}(\boldsymbol{y}) + \{I - Y\boldsymbol{F}'(\boldsymbol{X})\}(\boldsymbol{X} - \boldsymbol{y})$$

- $\boldsymbol{X}$: n 次元区間ベクトル
- $\boldsymbol{f}'(\boldsymbol{X})$: $\boldsymbol{f}$ の $\boldsymbol{X}$ における Jacobi 行列の値域
- $\boldsymbol{F}'(\boldsymbol{X})$: $\boldsymbol{f}'(\boldsymbol{X})$ を区間行列として拡張したもの
- $\boldsymbol{y}$: 与えられた $\boldsymbol{X}$ 内の点
- Y は適当に定められた正則行列

Nonliner Equations $f(x) = 0$

Krawczyk 作用素:

$$K(\boldsymbol{X}) \equiv \boldsymbol{y} - Y\boldsymbol{f}(\boldsymbol{y}) + \{I - Y\boldsymbol{F}'(\boldsymbol{X})\}(\boldsymbol{X} - \boldsymbol{y})$$

- $\boldsymbol{X}$: n 次元区間ベクトル
- $\boldsymbol{f}'(\boldsymbol{X})$: $\boldsymbol{f}$ の $\boldsymbol{X}$ における Jacobi 行列の値域
- $\boldsymbol{F}'(\boldsymbol{X})$: $\boldsymbol{f}'(\boldsymbol{X})$ を区間行列として拡張したもの
- $\boldsymbol{y}$: 与えられた $\boldsymbol{X}$ 内の点
- Y は適当に定められた正則行列

《検証条件》 $K(\boldsymbol{X}) \subset \boldsymbol{X}$

$\Rightarrow \boldsymbol{f}(\hat{\boldsymbol{x}}) = \boldsymbol{0}$ の解 $\hat{\boldsymbol{x}} \in \mathring{\boldsymbol{X}}$ が局所一意に存在

The Ideas

- 丸め誤差の制御は区間演算によって可能

  $\pi \in [3.14, 3.15]$

The Ideas

- 丸め誤差の制御は区間演算によって可能

$$\pi \in [3.14, 3.15]$$

- 打ち切り誤差は不動点定理を用いて評価

$$F(U) \subset U$$

Take Account of Discretization Error 1

Take Account of Discretization Error 1

無限次元非線形問題 $u = F(u)$ の精度保証

Take Account of Discretization Error 1

無限次元非線形問題 $u = F(u)$ の精度保証

- F の不動点 u を探す．ただし u は無限次元関数空間 S の要素; $u = \sum_{i=1}^{\infty} u_i \phi_i$

Take Account of Discretization Error 1

無限次元非線形問題 $u = F(u)$ の精度保証

- F の不動点 u を探す．ただし u は無限次元関数空間 S の要素; $u = \displaystyle\sum_{i=1}^{\infty} u_i \phi_i$
- S_h: S の有限次元近似空間

Take Account of Discretization Error 1

無限次元非線形問題 $u = F(u)$ の精度保証

- F の不動点 u を探す．ただし u は無限次元関数空間 S の要素; $u = \displaystyle\sum_{i=1}^{\infty} u_i \phi_i$
- S_h: S の有限次元近似空間
- P_h: $S \longrightarrow S_h$: 近似射影; $P_h u = \displaystyle\sum_{i=1}^{N} u_i \phi_i$

Take Account of Discretization Error 1

無限次元非線形問題 $u = F(u)$ の精度保証

- F の不動点 u を探す．ただし u は無限次元関数空間 S の要素; $u = \displaystyle\sum_{i=1}^{\infty} u_i \phi_i$
- S_h: S の有限次元近似空間
- P_h: $S \longrightarrow S_h$: 近似射影; $P_h u = \displaystyle\sum_{i=1}^{N} u_i \phi_i$
- $v = P_h v + v_* \quad P_h v \in S_h,\ v_* \in S - S_h$

Take Account of Discretization Error 1

無限次元非線形問題 $u = F(u)$ の精度保証

- F の不動点 u を探す．ただし u は無限次元関数空間 S の要素; $u = \displaystyle\sum_{i=1}^{\infty} u_i \phi_i$
- S_h: S の有限次元近似空間
- P_h: $S \longrightarrow S_h$: 近似射影; $P_h u = \displaystyle\sum_{i=1}^{N} u_i \phi_i$
- $v = P_h v + v_* \quad P_h v \in S_h,\ v_* \in S - S_h$

$$\sum_{i=1}^{\infty} u_i \phi_i = \sum_{i=1}^{N} u_i \phi_i + \sum_{i=N+1}^{\infty} u_i \phi_i$$

Take Account of Discretization Error 2

《無限次元の集合》を計算機で表現するには?

$$\sum_{i=1}^{\infty} u_i \phi_i = \sum_{i=1}^{N} u_i \phi_i + \sum_{i=N+1}^{\infty} u_i \phi_i$$

Take Account of Discretization Error 2

《無限次元の集合》を計算機で表現するには?

$$\sum_{i=1}^{\infty} u_i\phi_i = \sum_{i=1}^{N} u_i\phi_i + \sum_{i=N+1}^{\infty} u_i\phi_i$$

- 有限次元部分は区間係数: $\sum_{i=1}^{N}[\underline{u}_i, \overline{u}_i]\phi_i \subset S_h$

Take Account of Discretization Error 2

《無限次元の集合》を計算機で表現するには?

$$\sum_{i=1}^{\infty} u_i\phi_i = \sum_{i=1}^{N} u_i\phi_i + \sum_{i=N+1}^{\infty} u_i\phi_i$$

- 有限次元部分は区間係数: $\displaystyle\sum_{i=1}^{N}[\underline{u}_i, \overline{u}_i]\phi_i \subset S_h$
- 無限次元部分はノルム評価: $\{v \in S - S_h \mid \|v\| < K\}$

Take Account of Discretization Error 2

《無限次元の集合》を計算機で表現するには?

$$\sum_{i=1}^{\infty} u_i \phi_i = \sum_{i=1}^{N} u_i \phi_i + \sum_{i=N+1}^{\infty} u_i \phi_i$$

- 有限次元部分は区間係数: $\displaystyle\sum_{i=1}^{N} [\underline{u}_i, \overline{u}_i] \phi_i \subset S_h$
- 無限次元部分はノルム評価: $\{v \in S - S_h \mid \|v\| < K\}$
- K は何らかの方法で定量的に分かっているとする

$$u = \sum_{i=1}^{\infty} a_i \sin(i\pi x), \quad u_h = \sum_{i=1}^{N} a_i \sin(i\pi x) \ \ \text{on}(0,1)$$

$$\|u - u_h\|_{L^2} \le \frac{1}{(N+1)\pi} \|u''\|_{L^2}$$

Take Account of Discretization Error 3

Take Account of Discretization Error 3

- $U = U_h + U_*$

Take Account of Discretization Error 3

- $U = U_h + U_*$
  - $U_h = \sum_{i=1}^{N} [\underline{A}_i, \overline{A}_i]\phi_i \subset S_h$
  - $U_* = \{\phi \in S - S_h \mid \ \|\phi\| \leq K\}$

Take Account of Discretization Error 3

- $U = U_h + U_*$
  - $U_h = \sum_{i=1}^{N} [\underline{A}_i, \overline{A}_i]\phi_i \subset S_h$
  - $U_* = \{\phi \in S - S_h \mid \ \|\phi\| \le K\}$

- 検証条件

$$\begin{cases} P_h F(U) & \subset & U_h \\ (I - P_h)F(U) & \subset & U_* \end{cases}$$

$$\Rightarrow F(U) \subset U$$

$$\Rightarrow \exists u \in F(U) \ \text{ such that } \ u = F(u)$$

The Ideas

- 丸め誤差の制御は区間演算によって可能

  $\pi \in [3.14, 3.15]$
- 打ち切り誤差は不動点定理を用いて評価

  $F(U) \subset U$

The Ideas

- 丸め誤差の制御は区間演算によって可能
  $$\pi \in [3.14, 3.15]$$
- 打ち切り誤差は不動点定理を用いて評価
  $$F(U) \subset U$$
- 離散化誤差は無限次元の不動点定理と区間演算を用いて評価

The Ideas

- 丸め誤差の制御は区間演算によって可能

  $\pi \in [3.14, 3.15]$
- 打ち切り誤差は不動点定理を用いて評価

  $F(U) \subset U$
- 離散化誤差は無限次元の不動点定理と区間演算を用いて評価
  - できることもある

The Ideas

- 丸め誤差の制御は区間演算によって可能
  $$\pi \in [3.14, 3.15]$$
- 打ち切り誤差は不動点定理を用いて評価
  $$F(U) \subset U$$
- 離散化誤差は無限次元の不動点定理と区間演算を用いて評価
  - できることもある
  - 微分方程式の場合，個別撃破の戦略が必要

The Ideas

- 丸め誤差の制御は区間演算によって可能

  $$\pi \in [3.14, 3.15]$$

- 打ち切り誤差は不動点定理を用いて評価

  $$F(U) \subset U$$

- 離散化誤差は無限次元の不動点定理と区間演算を用いて評価
  - できることもある
  - 微分方程式の場合，個別撃破の戦略が必要
  - Newton 法を活用

The Ideas

- 丸め誤差の制御は区間演算によって可能

  $$\pi \in [3.14, 3.15]$$

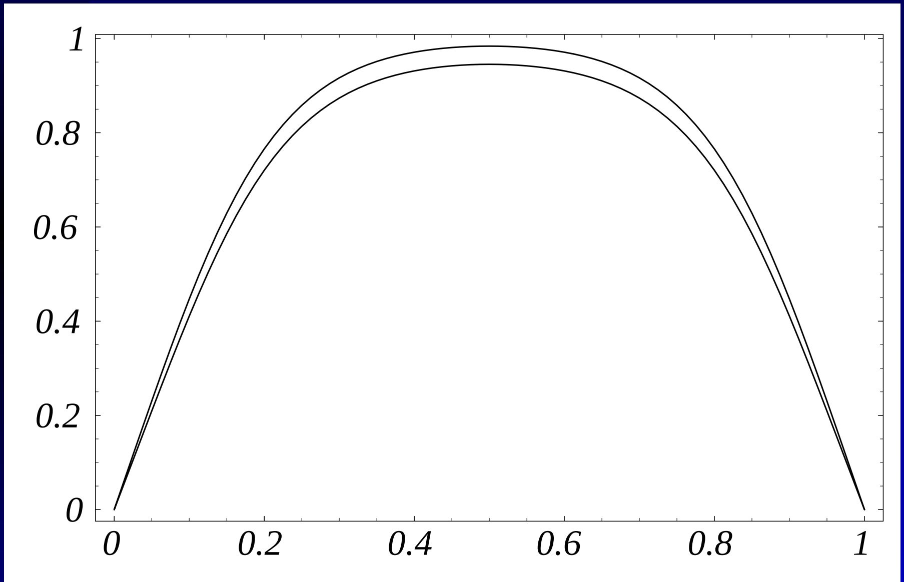
- 打ち切り誤差は不動点定理を用いて評価

  $$F(U) \subset U$$

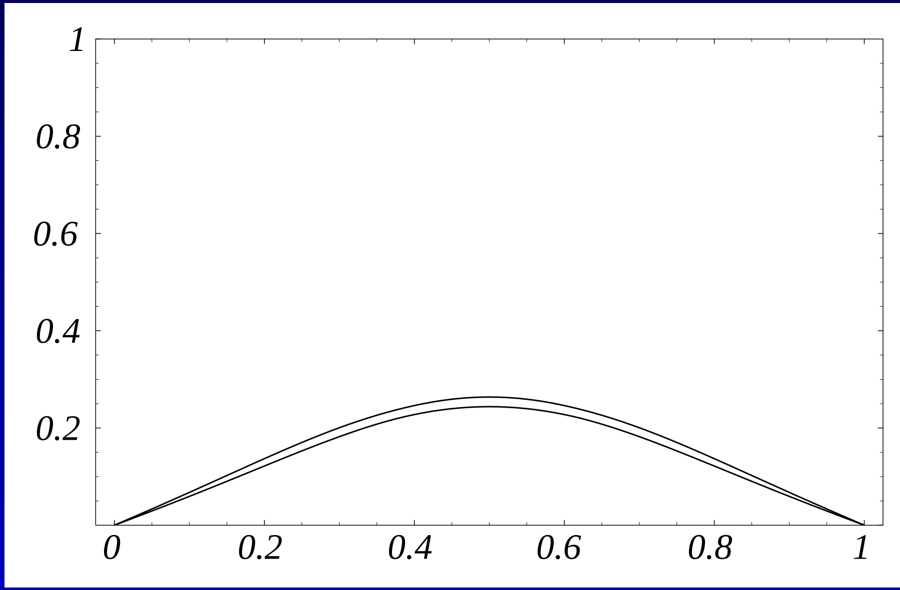
- 離散化誤差は無限次元の不動点定理と区間演算を用いて評価
  - できることもある
  - 微分方程式の場合，個別撃破の戦略が必要
  - Newton 法を活用
  - 区間演算は《最後の手段》

A: Allen-Cahn equation

$$\begin{cases} -\Delta u = \lambda u(u-a)(1-u), & x \in \Omega, \\ u = 0, & x \in \partial\Omega \end{cases}$$



上側分岐解の包み込み ($y = 1/2$)

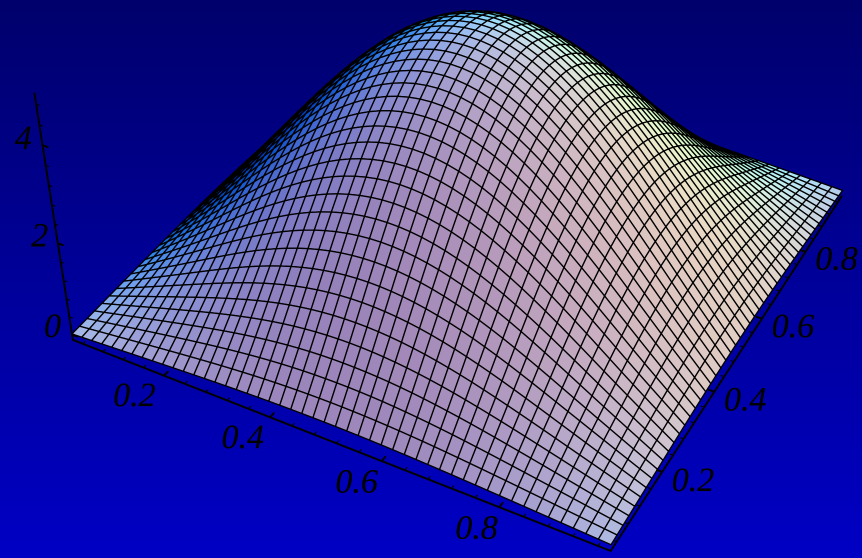


下側分岐解の包み込み ($y = 1/2$)

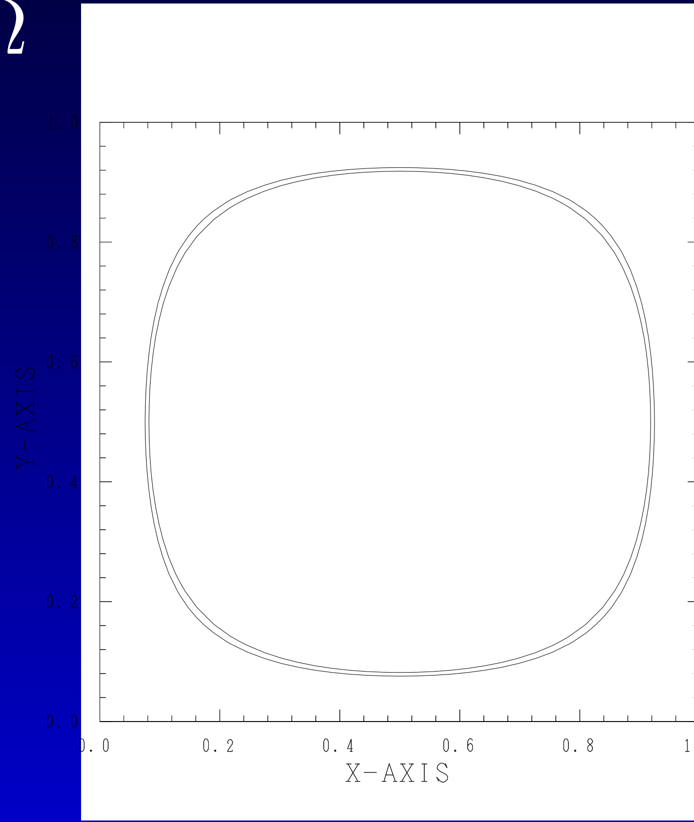
$$\lambda = 150,\ a = 0.01$$

B: MHD equilibrium

$$\begin{cases} -\Delta v &= \lambda \max\{v, 0\}, & x \in \Omega, \\ v &= -1, & x \in \partial\Omega \end{cases}$$



近似解の形状



$v = 0$ (自由境界) の包込

$$\lambda = 30$$

C: Heat Convection Problem 1

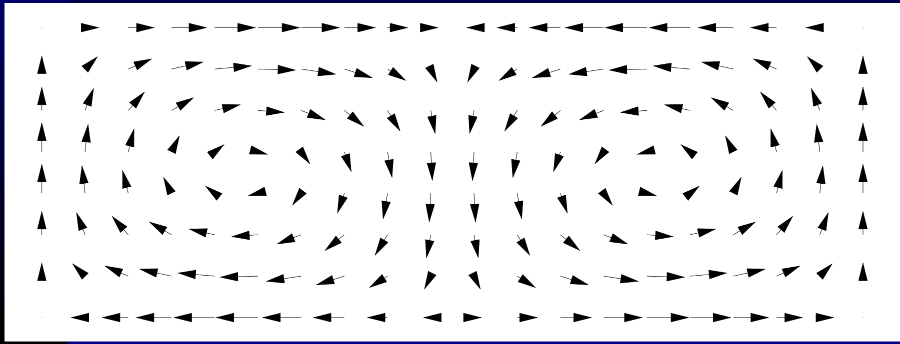
2 次元 (x-z 座標) Oberbeck-Boussinesq 方程式:

$$\begin{cases} u_t + uu_x + \omega u_z &= p_x + \mathcal{P}\Delta u, \\ \omega_t + u\omega_x + \omega\omega_z &= p_z - \mathcal{P}\mathcal{R}\,\theta + \mathcal{P}\Delta\omega, \\ u_x + \omega_z &= 0, \\ \theta_t + \omega + u\theta_x + \omega\theta_z &= \Delta\theta \end{cases}$$

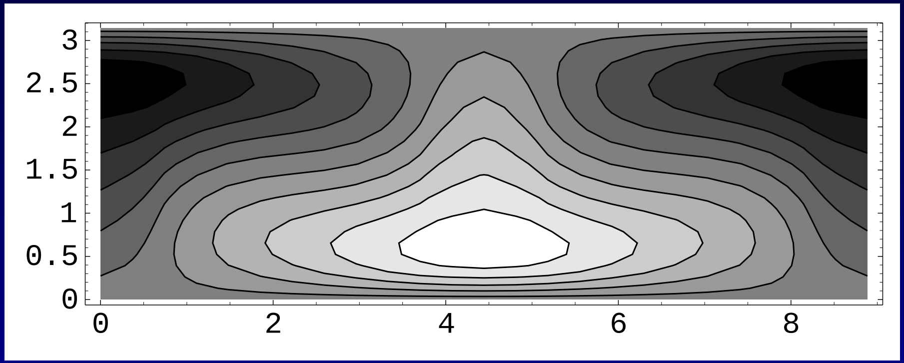
- (u, w): 速度場, p:圧力場, θ: 温度場, $\mathcal{P}$: Prandtl 数, $\mathcal{R}$: Rayleigh 数
- 定常分岐解の存在検証
- 長方形領域 $\{0 < x < 2\pi/a,\ 0 < z < \pi\}$
- $z = 0$, $z = h$ で接線応力が 0 となる自由表面
- $x = 0$, $x = 2\pi/a$ において周期境界条件

C: Heat Convection Problem 2

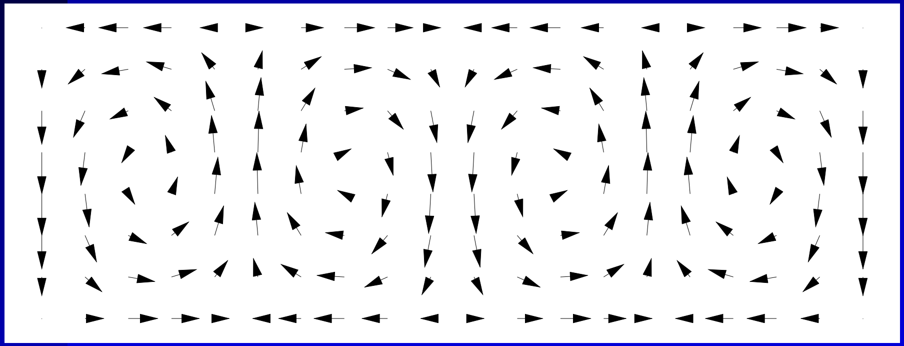
$\mathcal{P} = 10, \mathcal{R} = 40,$ 温度場は自明解からの摂動



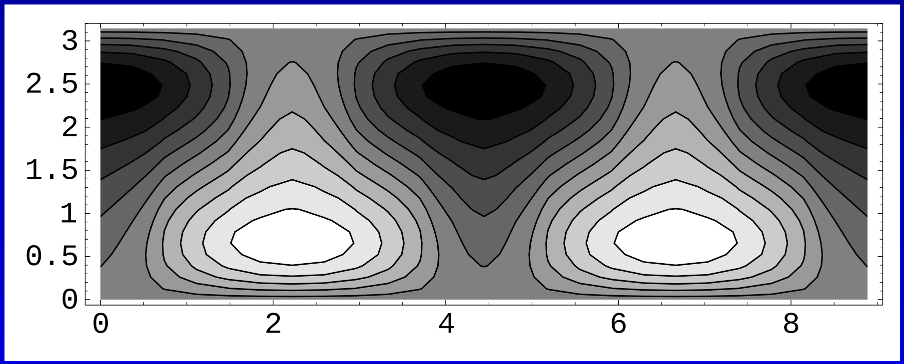
第 1 分岐解の形状: 流れ場



等温場



第 2 分岐解の形状: 流れ場



等温場

‘Error’

流れ現象

$\Downarrow$ ⟵ モデル化誤差

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \nu \Delta \boldsymbol{u}$$

$\Downarrow$ ⟵ 離散化誤差

$$F(\boldsymbol{x}) = 0 \quad \boldsymbol{x} \in \boldsymbol{R}^n$$

$\Downarrow$ ⟵ 打ち切り誤差

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - F'(\boldsymbol{x}_{n-1})^{-1} F(\boldsymbol{x}_{n-1})$$

$\Downarrow$ ⟵ 丸め誤差

$$\boldsymbol{x}_N = (1.223, 2.342, \ldots)$$

Conclusion

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

$\Downarrow$

数値計算誤差の《最悪の》結果を見積もる

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

数値計算誤差の《最悪の》結果を見積もる

⇑

地味

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

数値計算誤差の《最悪の》結果を見積もる

⇑

地味

『告白と呪詛』シオラン (Cioran), 出口 裕弘 訳

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

数値計算誤差の《最悪の》結果を見積もる

⇑

地味

自分がしたことを誇るのもよかろう。

『告白と呪詛』シオラン (Cioran), 出口 裕弘 訳

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

数値計算誤差の《最悪の》結果を見積もる

⇑

地味

自分がしたことを誇るのもよかろう。
だが、それよりも私たちは、

『告白と呪詛』シオラン (Cioran), 出口 裕弘 訳

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

数値計算誤差の《最悪の》結果を見積もる

⇑

地味

自分がしたことを誇るのもよかろう。
だが、それよりも私たちは、自分がしなかったことを、

『告白と呪詛』シオラン (Cioran), 出口 裕弘 訳

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

数値計算誤差の《最悪の》結果を見積もる

⇑

地味

自分がしたことを誇るのもよかろう。
だが、それよりも私たちは、自分がしなかったことを、大いに誇るべきではなかろうか。

『告白と呪詛』シオラン (Cioran), 出口 裕弘 訳

Conclusion

精度保証付き数値計算法とは, 問題に対する解の**存在** (一意性) と**誤差限界**を数学的に保証するような数値計算法

⇓

$\boxed{\text{数値計算誤差の《最悪の》結果を見積もる}}$

⇑

地味

> 自分がしたことを誇るのもよかろう。
> だが、それよりも私たちは、自分がしなかったことを、大いに誇るべきではなかろうか。
> その種の誇りを、ぜひとも創り出すべきだ。
>
> 『告白と呪詛』シオラン (Cioran), 出口 裕弘 訳