数理工学実験

2021年度版

実験演習ワーキンググループ編

数理工学実験 (2021 年度版)

京都大学 工学部情報学科 数理工学コース

実験演習ワーキンググループ：

岩崎淳 助教
上田仁彦 助教
大木健太郎 助教
加嶋健司 准教授
上岡修平 助教
Shurbevski, Aleksandar 助教
田口智清 教授
辻徹郎 助教
筒広樹 助教
中山優吾 助教
新納和樹 助教
原田健自 助教
福田エレン秀美 准教授
山川雄也 助教
山口義幸 助教

目 次

第1章　レポートの書法

本稿では, まず第 1.1 節でレポートの構成・形式・文章作成上の注意点などを述べる. 次に, レポート作成に使用する LaTeX の簡単な使い方を第 1.2 節で, gnuplot を使って図を作成する方法を第 1.3 節で紹介する.

1.1　レポートの内容

実験のレポートとは, その実験の報告書であり, 実験内容を知らない第三者が読んでも理解できるような内容でなければならない. 他人に読んで貰うためのものである以上, 読者の意欲を損なわないよう読みやすく仕上げることを心がける.

読みやすくするために注意すべきこととして,

- 構成 (第 1.1.1 節)
- 形式 (第 1.1.2 節)
- 文章 (第 1.1.3 節)

を順に説明する. また科学実験に付随する問題として

- 物理量の記法 (第 1.1.4 節)

を述べる.

1.1.1　構成

表紙

本実験のレポートには表紙を付ける. 表紙には以下の項目を記入すること.

- 科目名
- 実験テーマ
- 実験の実施年月日
- 実験の実施場所
- レポートの提出年月日
- 提出者の学年, 学生番号および氏名

レポートの構成の一例

1. はじめに
   何をしたいのかという目的や, 目的を達成するための方法の概略を書く. 方法の詳細は次の「原理」に譲る.
2. 原理・方法の詳細
   配布冊子の内容をそのまま写してはいけない. 原理を一旦自分なりに理解した後, 説明に必要な項目を取捨選択し, 必要であれば補足して記す.
3. 実験方法
   レポートで報告されている実験を, 読者が再現できるように書く. 読者が再現できなければ, それはレポートに書かれている情報 (方法の詳細やパラメータの値など) が足りていないということである.
4. 結果
   実験結果をわかりやすくまとめる. 必要に応じて, 図や表を用いる. 図を gnuplot を用いて用意する方法は第 1.3 節を参照せよ.
5. 考察・議論
   実験から得られた結果の意味や正当性の吟味, 問題点の議論などを行う. 理論値があればそれとの対比を行ない, 理論値と実験値が異なればその理由を考察する.

   「有意義だった」「大変だった」などの個人の主観による感想は要らない. 科学的文書に必要なのは事実とそれに基づく論理的推論である.
6. 結論
   実験結果と考察から得られた内容のまとめを行う. 読者は「はじめに」と「結論」に注目することが多いので, これまでに述べたことでも大事なことは再記載・強調してもよい.

7. 参考文献
   本文中で引用した参考文献を記すこと. 本文に記載した事の根拠とそれが誰の業績かを明らかにするため, 正しく引用することは重要である.

1.1.2 形式

- 用紙は A4 判を使用する.
- LaTeX を使う.
  数式が混じった文章を綺麗に書くソフトとして LaTeX がある. LaTeX の簡単な使用法については第 1.2 節を参照すること.
- 使用する記号にはすべて定義を与える.
- 式, 図, 表の参照は番号で行う.
  文中で式, 図, 表を参照する場合は番号を使用する. 例えば,
  式 (2) を式 (10) に代入すると…
  測定結果を表 2 にまとめる.
  などと使用する. 式番号は ( ) で囲み, 図や表の番号は囲まない. これらの番号は文中に現れる順番による. 文章構成の変更にともない式などの順番が変わっても自動的に正しく参照する方法については第 1.2 節を見ること.
- 図表の適切な使用.
  図や表は一つの独立した要素であるので, それを見ただけで何を表したものであるかが明確にされていなければならない. また適切なグラフ形式 (線形・片対数・両対数など) を選択し, 縦軸, 横軸は何を表しているのか, そしてそれらの単位は何であるかを明らかにしなければならない.

1.1.3 文章

ネイティブな日本語話者であっても、正しくかつ読みやすい日本語文章を書くには訓練が必要である. ここではいくつかの注意点を述べる. どう書くのが良いか迷った場合には, 文の自然さや格調を多少犠牲にしても, 意味の明確さを優先することを勧める. 文献 [1] も参考にすること.

主張は何か?

レポートを含め科学的文書の目的は,「自分の主張を読者に伝える」ことである. まずは「自分の主張が何であるか」をよく考え, それを強く意識しながら書く.

構成

主張すべきことが決まったら, その主張が明確になるよう構成を考える. だいたいは第 1.1.1 節で示したような例に沿えば良いが, 必要であれば順番を入れ替えたり, 項目を追加・削除することも考える.

段落

段落は気分で分けるものではない. 意味の塊ごとに段落を改める. ただしあまりに長い段落は読みにくさのもとなので, バランスを取って適切に段落分けをする.

例えば A1, A2, B1, B2, B3 というトピック毎のブロックがあるとする. (A1+A2), (B1+B2+B3) がそれぞれ適切な長さであれば, 2 つの段落に分ける. A1 などそれぞれのブロックが十分に長い場合は, 5 つの段落に分けることを考える.

文の順番

× A 君は朝起きた. B 君も朝起きて大学に行った. A 君は大学に行く途中で買い物をした. A 君と B 君は大学で会った.

内容は理解できるが, A 君のトピックが第 1,3 文に別れており読みにくい. 次のように交通整理する.

○ A 君は朝起きた. A 君は大学に行く途中で買い物をした. B 君も朝起きて大学に行った. A 君と B 君は大学で会った.

文の長さ

1 文がとても長くなると読みにくさのもとになる. 時に 1 文が 10 行ほどにも渡る場合を見かけるが, 避ける方が賢明である. ただし 1 文 1 文があまりに短いと文章が幼い印象になる. その意味では先の例文も改善の余地がある.

○ A 君は朝起き, 大学に行く途中で買い物をした. B 君も朝起きて大学に行った. 二人は大学で会った.

多重の意味に取れることを避ける

× ぼくは A 君と B 君と C 君の家に行った.

これは多重に意味が取れる.
1. A 君, B 君, C 君の家にそれぞれ行った.
2. A 君と一緒に, B 君, C 君の家にそれぞれ行った.
3. A 君,B 君と一緒に, C 君の家に行った.

どの意味であるかわからないような文は避けるべきである.

主語と述語

命令形など特殊な場合を除き, 主語と述語は文の最小セットである[1]. 主語に対する述語が正しく書けているか注意する. 途中で形容する部分が長く入る場合には特に注意する.

× 関数 $f(x)$ は原点付近で $f(x) = f(0)+xf'(x)+\cdots$ とテイラー展開する.

関数が主体になってテイラー展開することはないので、次のように改善する.

○ (我々は) 関数 $f(x)$を原点付近で $f(x) = f(0)+xf'(x)+\cdots$ とテイラー展開する.

○ 関数 $f(x)$ は原点付近で $f(x) = f(0)+xf'(x)+\cdots$ とテイラー展開される.

[1]以下の第 2 例文のように, 日本語では主語を書かないほうが自然な文になる場合も多いが, ここではこの事情に深入りしない.

代名詞

代名詞はその名の通り「名詞の代わり」だから代名詞をそれが指す名詞で置き換えても文章が成立していなければならない. 例えば

○ A 君は僕の友達だ. 彼は日本人だ.

において, 2 文目では「彼」=「A 君」だから

○ A 君は僕の友達だ. A 君は日本人だ.

と書いてもよい.

代名詞が指す名詞が明確か否かを常に意識すること. 明確でなければ, 代名詞が指す名詞をそのまま書く.

× 方法 A と B がある. ここではこれを用いる.

○ 方法 A と B がある. ここでは方法 B を用いる.

助詞

「て, に, を, は」 が間違っていないか, 意味が取りやすいか, に注意する. また「の」が連続する文章は読みにくいので避ける.

× 上の方程式の右辺の第 1 項の変数 x の次数は 2 である.

○ 上の方程式の右辺において、第 1 項に現れる変数 x の次数は 2 である.

接続詞

「および」「あるいは」「よって」「しかし」「一方」等, 文などを繋ぐ言葉である。論理的な文章を書く上では重要である.

1.1.4 物理量について

物理量についての注意を述べる. 数値計算による実験においては該当しないこともあるので取捨選択すること.

有効数字

実験で測定された結果は有限桁の精度しかない. 有効桁だけを残したものを有効数字という. 例えば有効桁 3 桁で 1.23×10^1 と書けば, この数値には 0.01×10^1 の誤差を含んでおり真値は $[12.25, 12.35)$ の間にあることを意味する. 1.230×10^1 の真値は $[12.295, 12.305)$ の間にあり, 両者の意味が異なることがわかる.

2 つの有効数字を四則演算した場合, 計算結果の有効桁は次の通り:

- 乗算・除算: 有効桁のうち小さい方
  $(1.2 \times 10^1) \times (3.45 \times 10^2) = 4.140 \times 10^3 \to 4.1 \times 10^3$
  (有効桁 2 桁と 3 桁の積の有効桁は 2 桁)
- 加算・減算: 有効数字末端の桁位の大きい方
  $1.250 + 56.7 = 57.950 \to 58.0$
  (10^{-3} の桁までと 10^{-1} の桁までの和は 10^{-1} の桁まで有効だから 10^{-2} の桁を四捨五入する)

字体の使い分け

1. 物理量の添字が物理量あるいは変数を表す場合にはイタリック体を用いる. それ以外はローマン体を用いる.
   周波数 f に対する物理量 A の値を添え字で書くと A_f だが, 例えば前方 (forward) の略字を意図する場合は A_{f} と書く.
2. 単位を表す記号はローマン体を用いる.
   質量の単位は kg ではなく kg である.
3. 数学定数を表す文字はローマン体である.
   自然対数は e (e ではない), 虚数単位は i (i ではない) と表す.
4. 特殊な数学関数 ($\sin, \cos, \log$ など) や常微分演算子 $\frac{\mathrm{d}}{\mathrm{d}x}$ はローマン体で表す.

詳しい説明は [2] を参照のこと.

関係式の表記

物理量を文字を使って表す場合の注意についても [2] に説明されている. 重要な点は, 物理量相互間の関係式は単位に無関係に普遍的に成立するということである.

良く見かける不適切な例を [2] から引用しておこう.

× 電磁波の波長 λ[m] と周波数 f[Hz] の間には以下の関係がある.

$$\lambda = 3.0 \times 10^8 / f$$

3.0×10^8 という無次元量を使った関係式は λ を m, f を Hz という「ものさし」を用いた場合に有効であって, そうでない場合は成立しない. λ と f の単位をどのようにとっても成立する関係式は, 以下のように書けばよい.

○ 電磁波の波長 λ と周波数 f の間には以下の関係がある.

$$\lambda = (3.0 \times 10^8 \mathrm{m/sec}) / f$$

図表の数値

図や表に現れる数値は無次元量で書かなければならない. 例えば長さ l を cm 単位で測った値を数直線上に表すとしよう. 数直線上の値が 1.2 という点は $l = 1.2$cm を表す. よって数直線のラベルを l/cm としておくと 1.2 は無次元量となる. 測定の単位を m に変えて $l = 1.2 \times 10^{-3}$m となるときはラベルを $10^3 l$/m とすればよい. この方法を使うと, グラフから読み取った値を 10^{-3} 倍するのか, 読み取った値はすでに 10^{-3} 倍されているのかという曖昧さを無くすことができる.

1.1.5 Activity 1

I. 「二次方程式 $x^2 - 2x - 3 = 0$ の解を求めよ」という問題に対して,
「$(x-1)^2 = 4$. $x = 3, -1$.」
という答案が提出された. この答案を改善するた

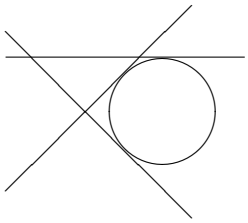
めに, 答案提出者へのアドバイスを考えよ. 箇条書きでも良い.

II. 先の問題に対する次の答案を改善せよ.

「$x^2-2x-3=0$ の問題を考える. 平方完成すればよい. この問題は平方完成できる. 解を求めると $x=3,-1$.」

III. $\sqrt{2}$ が無理数であることの証明を書け. 誰が読んでも誤解なくまた読みやすく書くこと. [2]

IV. 次の図形を文章のみで描写せよ. 図形を見たことがない人が読んだだけで描けるようにせよ. 大きさや多少の誤差は気にしなくて良い.



1.2 LaTeX の使い方

LaTeX とは, ソースファイルに書かれたコマンドをコンパイル (機械がわかるように翻訳) することにより, 数式が混じった文章を綺麗に仕上げるためのソフトである. ここでは基本的な使い方, および節・式・図・表番号の自動参照の仕方を説明する.

1.2.1 最初の一歩

最小限の LaTeX コマンドセットは

```
\documentclass{jarticle}
\begin{document}
本文を書く所
\end{document}
```

である. \begin と \end で挟まれた箇所を LaTeX では「環境」と呼ぶ. 上の例では document 環境が作成されたことになる. 節の題名や本文・式・図・表はすべて document 環境内に書く. documentclass が jarticle になっているのは日本語用だからであり, 英語用には article を用いる. なお環境によっては \ (バックスラッシュ) の代わりに ¥ (円マーク) が使われる場合もある.

1.2.2 節の作り方

```
\section{節の題名}

\subsection{小節 1 の題名}

\subsection{小節 2 の題名}
```

と書くと節や小節が作られ, 自動的に節・小節の番号が付けられる.

節番号を参照するときに手で書いていては, 構成を変更したときに面倒になる. 自動的に参照できるようにするには

```
\section{節の題名}
\label{sec:example}
....
第\ref{sec:example}節では...
```

と \section の後に \label でラベルを付けておいて, 参照したい箇所で \ref にラベルを書けばよい. 以下で見るように, 番号の自動参照は式番号, 図番号, 表番号でも可能である. ここでは節のラベルであることがわかるようにラベルの名前を sec:example としたが, ラベルの名前は自由にしてよい. ただしラベル毎に名前は変えなければならない.

なお正しく参照させるためには \label を付けて (あるいは変更して) から LaTeX ソースファイルを2 回コンパイルしなければならない.

1.2.3 文章の書き方

文章は LaTeX が適当なところで改行してくれる. ソースでの改行は LaTeX 文章には反映されな

[2] 証明の方法は, $\sqrt{2}$ が有理数になっている, つまり $\sqrt{2}=p/q$ (p,q は互いに素な整数, $q\neq 0$) なる p,q が存在すると仮定し, 両辺を二乗して矛盾を導けばよい.

い. コンパイル後の文章中で強制的に改行したい場所には \\ を書く[3].

段落を改めたい場合は, 前の段落の終わりから一行間を開けて書き始めればよい. 段落の先頭では自動的に一字さがるがこれはこのままで良い。

1.2.4 数式の書き方

文章中では $ で挟むことにより数式を書く。例えば $\alpha=\beta$ と書くと $\alpha = \beta$ と出力される.

式に番号を付けるには

```
\begin{equation}
  \alpha = \sin^{2}\beta
\end{equation}
```

とする。出力は

$$\alpha = \sin^2 \beta \tag{1.1}$$

となる。式番号は式が現れる順に付される. 本稿では実は documentclass が jarticle ではなく jbook のために (1.1) という番号になっているが, documentclass を jarticle にすると (1) などの番号になる.

式番号を自動的に参照するには,

```
\begin{equation}
  \alpha = \sin^{2}\beta
  \label{eq:example}
\end{equation}
```

と equation 環境の中で \label を付けておく. 参照する箇所では \ref を使って

```
式 (\ref{eq:example}) によると...
```

と書くと,

式 (1.1) によると...

と出力される.

番号なしの式は

```
\[ a=b \] あるいは $$ a=b $$
```

と書くと

$$a = b$$

と出力される.

[3]見た目を整える以外には使わない.

1.2.5 図の入れ方

図のファイル形式は様々あるが, 第 1.3 節で見るように gnuplot の epslatex を使うと LaTeX の書式で美しい数学記号を EPS 形式の図中に入れられる. そこでここでは EPS ファイル example1.eps と example2.eps があるとしてこれを LaTeX 中に取り入れる方法を述べる.

手順 1: \documentclass と \begin{document} の間で graphicx パッケージを dvipdfmx オプションで読み込む.

```
\documentclass{jarticle}
\usepackage[dvipdfmx]{graphicx} %←ココ
\begin{document}
...
\end{document}
```

手順 2: example1.eps と example2.eps を LaTeX ソースと同じディレクトリ (フォルダ) に入れた後, 図を入れたい場所に figure 環境で取り入れる.

```
\begin{figure}
 \centering
 \includegraphics{example1.eps}
 \includegraphics{example2.eps}
 \caption{図の取り入れ方法の例. 第\ref{sec:gnuplot}節に示した二通りの作成方法の出力を比較している.}
 \label{fig:example}
\end{figure}
```

図の大きさを変えたい場合は,
\includegraphics[width=6cm]{example1.eps}
などとサイズを指定する. このようにサイズ指定して取り込んだ例を図 1.1 に示す.

図についての注意

- 一つの figure 環境に複数枚の図がある場合には, (a) (b) などで区別する. 参照するときも図 1.1(a) などと書く.

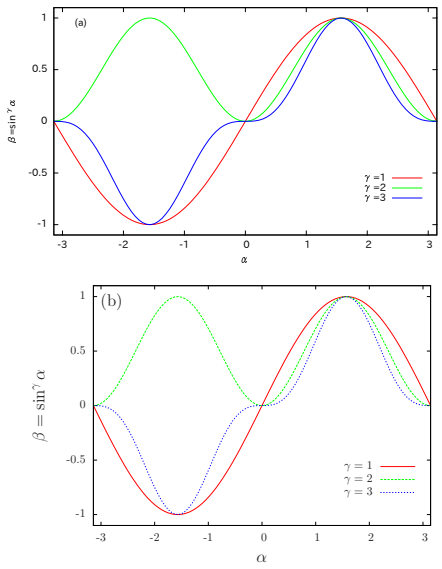


図 1.1: 図の取り入れ例. 第 1.3 節に示した二通りの作成方法の出力を比較している. (a) enhanced を使用. (b) epslatex を使用. パネル (b) だけが示された図に対するキャプション例は次の通り: $\beta = \sin^\gamma \alpha$ のグラフ. $\gamma = 1$ (赤, 実線), 2 (緑, 破線), 3 (青, 一点鎖線).

- \caption の中には図の表題や説明を書く. 「実験装置」など一語だけのこともあれば, パラメータの値などを全部書いて 10 行以上に及ぶこともある. これは分野の文化による. いずれにせよ, 本文中あるいは \caption 中に必要な説明を与えるべきである.
- 式番号と同様に \label で図番号も参照できる. もちろんラベルの名前は自由でよい.
- LaTeX において図の位置調整は難しい.

  `\begin{figure}[htbp]`

  と h (here), t (top), b (bottom), p (page: 図用ページ) のオプションがある. これらオプションを取捨選択したり, figure 環境のブロックをソースファイル中で上下に動かしたりすると, ある程度は図の位置を動かせるが, 完全に思い通りにならないこともある.

1.2.6 表の入れ方

表を入れるには table 環境を使う. 以下のソースに対する出力は表 1.1 となる.

```
\begin{table}[h]
  \centering
  \caption{表の出力例}
  \begin{tabular}{l|ll}
    \hline
    a & 1 & 2 \\
    b & 3 & 4 \\
    \hline
  \end{tabular}
  \label{tab:example}
\end{table}
```

表 1.1: 表の出力例

a	1	2
b	3	4

表についての注意

- caption の内容については, 図と同様である.
- 図の caption は下に置くが, 表の caption は上に置く. \caption を \tabular よりも上に置けばよい.
- 式番号, 図番号と同様に \label で表番号も参照できる. もちろんラベルの名前は自由でよい.
- 図と同様に h, t, b, p のオプションがあるが, やはり位置の調整は簡単ではない.

1.2.7 参考文献の書き方

参考文献は, それを出力する位置に以下のように書く.

```
\begin{thebibliography}{99}
\bibitem{Exp} 京都大学工学部電気系教室編，電気電子工学基礎実習.
\end{thebibliography}
```

Eexp は文献のラベルであり, 参照したい位置で \cite{Eexp} (\ref{Eexp} ではない) と書くと [2] と出る. 文献が複数ある場合は \bibitem を増やしていくと, 文献に付く番号が順に大きくなる. 文中に現れる順ではなく, thebibliography 環境に書いた順に番号が付されることに注意せよ.

文献情報の書式は分野の文化による. ここではその一例を紹介する.

書籍の場合

```
\bibitem{Feynman1} R. P. ファインマン,
ファインマン物理学 1 力学 (新装版), 坪井忠
二訳 (岩波書店, 1986).
\bibitem{LL1} L. ラ ン ダ ウ, E. リ フ シッ
ツ, 力学 (増訂第 3 版) ランダウ=リフシッツ理
論物理学教程 (東京図書, 1986).
```

それぞれ 1986 は出版年で, その前は出版社名である.

雑誌論文の場合

```
\bibitem{EPR} A. Einstein, B. Podolsky,
and N. Rosen, Can quantum-mechanical
description of physical reality be
considered complete?,
Phys. Rev. {\bf 47}, 777 (1935).
```

必要な情報は,

- 著者名 (全員. 英語では姓以外はイニシャル)
- 雑誌名 (各雑誌に通例の略し方がある)
- 巻数 (太字で書く)
- 始まりのページ番号 (論文番号の場合あり)
- 出版年

である. 例のように, 論文の題名を入れる場合もある.

1.2.8 newcommand

何度も使う記号や命令はなるべく簡単にタイプしたい. こういう場合は \newcommand を使って \documentclass と \begin{document} の間に新しいコマンドを定義する. 例えば,

```
\documentclass{jarticle}
\newcommand{\al}{\alpha}
\begin{document}
...
```

としておくと、本文中では $\al$ と書くと α が出力される. 引数を持つコマンドも可能で

```
\newcommand{\be}[1]{\beta_{#1}}
```

として $\be{3}$ と書くと β_3 が出力される.

```
\newcommand{\ga}[2]{\gamma_{#1}^{#2}}
```

として $\ga{3}{4}$ と書くと γ_3^4 が出力される. 当然ながら、コマンドの名前は重複がないようにすべきである.

1.2.9 Activity 2

I. 次の出力になるように LaTeX のソースを作成せよ. 式番号は \ref を使って参照せよ. 式番号の参照さえできれば, 式番号は問題の番号でなくともよい.

関数

$$f(x,y) = x^2 + y \tag{1.2}$$

および

$$g(x,y) = y^2 + x \tag{1.3}$$

を考える. 関数 (1.2) および (1.3) の値が同時に 0 となる (x,y) を求める.

II. table 環境を使って, 今後 1 週間の週間天気予報を作れ (予報は仮定でもよい). caption も適切に書け.

III. 式番号の参照 (\ref{...}) で ( ) を書く手間を減らすために newcommand で \eqref を定義せよ.

\eqref{eq:example} と打てばラベル eq:example を持つ式番号が ( ) で挟まれるようにせよ.

IV. 常微分 $\frac{\mathrm{d}x}{\mathrm{d}t}$ をタイプする手間を減らすために newcommand を使って \fracd を定義し \fracd{x}{t} と打てば $\frac{\mathrm{d}x}{\mathrm{d}t}$ が出力されるようにせよ. なお d の部分は \mathrm{d} で出る.

1.3 gnuplot で図を作成する

gnuplot で図を作成するには二通りの方法がある.

- gnuplot を立ち上げて一行ずつコマンドを打ち込む.
- スクリプトファイル (実行すべきコマンドを順に書いたファイル) を gnuplot に読み込ませる.

作った図の修正が容易なため, 後者を勧める. 本稿でも後者の方法を説明する. gnuplot のバージョンは 4 以上であることを確認されたい.

軸のラベルなどにギリシャ文字の出力を試したいので, 以下では例として, $\beta = \sin\alpha$, $\beta = \sin^2\alpha$, $\beta = \sin^3\alpha$ のグラフを描く. ただし gnuplot で関数をプロットするには sin(x) などとしなければならないことに注意せよ.

1.3.1 enhanced を使った図の作り方

手順 1

次のようなスクリプトファイル (example1.gp とする) を用意する. スクリプトファイル中で, 行頭が # の行はコメント行となる.

example1.gp

```
set terminal eps color enhanced
set output "example1.eps"
set xlabel "{/Symbol a}"
set ylabel "{/Symbol b}"
set label 1 '{(a)}' at -2.8,0.9
set key at 3, -0.5
plot [-pi:pi][-1.1:1.1]
 sin(x) title "{{/Symbol g}=1}",
 sin(x)**2 title "{{/Symbol g}=2}",
 sin(x)**3 title "{{/Symbol g}=3}"
#上の 4 行は実際には 1 行で書く
quit
```

enhanced を使うことによって, {/Symbol a} でギリシャ文字 α を出力できる. (他も試してみよ) label の (a) で at の後ろは出力する座標なので適切に指定する.

手順 2

ターミナル上で gnuplot に読み込ませる.

```
gnuplot example1.gp
```

これで図 1.1(a) に示すような example1.eps というファイルができる. example1.eps という名前は 2 行目の set output で指定されており, example1.gp と example1.eps という名前の一致は必要ではなくバラバラでも構わない.

1.3.2 LaTeX 数式を使った図の作り方

図中の数式をより美しく仕上げる方法として, gnuplot の epslatex を使った方法を紹介する. (") と (') の違いに注意せよ. 事前に以下の include.tex を用意しておく.

include.tex

```
\documentclass{article}
\usepackage{graphicx,color}
\pagestyle{empty}
\begin{document}
\begin{figure}[h]
\input{./temp.tex}
\end{figure}
\end{document}
```

include.tex に書かれている temp.tex は, 次の手順 1 におけるスクリプトファイル example2.gp 内に現れる temp.eps と関連している. temp という名前は変更してもよいが, include.tex と example2.gp のセットでの変更が必要である. 最終的な EPS ファイルの名前は手順 4 で選べる.

手順 1

gnuplot のスクリプトファイルを準備する.

example2.gp

```
set terminal epslatex color
set output "temp.eps"
set xlabel '{\Large $\alpha$}'
set ylabel '{\Large $\beta$}'
set label 1 '{\Large (b)}' at -2.8,0.9
set key at 3, -0.5
plot [-pi:pi][-1.1:1.1]
 sin(x) title "{{/Symbol g}=1}",
 sin(x)**2 title "{{/Symbol g}=2}",
 sin(x)**3 title "{{/Symbol g}=3}"
 lt 5 lc rgb "blue"
#上の 5 行は実際には 1 行で書く
quit
```

$\gamma = 3$ のグラフは, 線種 (lt 5) と色 (lc rgb "blue") を指定している. 必要に応じて他のグラフも同様に指定できる. 点種や線種の番号は gnuplot のバージョンに応じて異なる可能性があるので, 実際に作成して確認すること. gnuplot のプロンプトで "test" とタイプすると情報が出る.

手順 2

ターミナル上で gnuplot に example2.gp を読み込ませる.

```
gnuplot example2.gp
```

これで temp.eps と temp.tex ができる.

手順 3

ターミナル上で include.tex をコンパイルする.

```
latex include.tex
```

手順 4

ターミナル上で次の一連のコマンドを実行する.

```
dvips -E include.dvi
epstopdf --outfile=include.pdf include.ps
pdftops -eps include.pdf example2.eps
```

最後の行の example2.eps は, 最終的に得られる eps ファイルの名前指定であり, 自由に変えられる. 図 1.1(b) は以上の手順で作成した図である.

手順 3 でコンパイルに失敗するときには

データファイルを使ってプロットしている場合, データファイルの名前に data_1.dat の "_" など LaTeX で使うコマンドが含まれているとコンパイルができない場合がある. この問題の解決方法として次の 3 つが考えられる.

1. 該当するファイルの名前を変える.
   data_1.dat → data1.dat
2. ファイル名が表示されないように gnuplot のスクリプトの中で unset key を使う.
3. gnuplot スクリプトの plot コマンドで title 'This is the title' のように title を設定することでデータファイル名が表示されるのを避ける. 当然, 意味のあるタイトルにすべきである.

さらに便利に

上の手順 2,3,4 をシェルスクリプトに書いておくと一括処理ができて便利である. 意欲がある人は挑戦してみられたい.

1.3.3 データファイルのプロット法

上の例では関数 sin(x) などを直接プロットしていたが, データファイル data.dat に格納された点をプロットする場合は,

```
plot sin(x)
```

の代わりに

```
plot "data.dat"
```

とする. なお data.dat は第 1 列に x 軸の値, 第 2 列に y 軸の値が来るように作成する:

```
x1 y1
x2 y2
x3 y3
 :  :
```

データファイルに第 3 列以降があってもよい. 第 2 列を横軸の値, 第 3 列を縦軸の値に使いたい場合は

```
plot "data.dat" using 2:3
```

とする. さらに, グラフを左に 1 だけずらしたい場合には

```
plot "data.dat" using ($2-1):3
```

なども使える.

1.3.4 Activity 3

I. $\alpha \in [0.01, 1]$ の範囲において,

- $\beta = 2\alpha$
- $\beta = \exp(\alpha)$
- $\beta = \alpha^2$

のグラフを同時に描いた図を作成せよ. それぞれが, 線形関数, 指数関数, 冪関数であることが一見してわかるようにプロットの方法を工夫して 3 枚作成せよ. (a)(b)(c) で区別を付け, それぞれ軸の名前も書くこと.

ヒント 1: グラフのうち一見してそれとわかるのは直線である. よって各関数のグラフが直線になるようにプロットを工夫すればよい. 例えば $\beta = \exp(\alpha)$ では横軸 α, 縦軸 β では直線にならないが, 縦軸を $\log\beta$ にするとどうだろう? gnuplot で縦軸を対数スケールにするには

```
set log y
```

と打てば良い. ただし, 軸の目盛りと整合するように軸の名前を付けること. (軸を対数スケールにしたら軸名は $\log\beta$ なのか, β のままなのか?)

ヒント 2: 軸の目盛りが 1000000 などでは見にくいので, 10^n のうち n だけを出力したい場合がある. y 軸でこれを行うには, gnuplot のスクリプトファイル中に

```
set format y "%T"
```

と指定すればよい. もちろん y 軸の名前はこれに合わせて適切に選ばなければならない.

II. 図ができたら LaTeX のソースに figure 環境で取り込んでみよ. caption には図の説明 (軸をどうしたか, 傾きがどうか, 等) を必要十分に入れよ. 本文 (figure 環境ではない所) には, プロットをどのように工夫したかの説明を書け.

[山口義幸]

参考文献

[1] 木下是雄, 理科系の作文技術 (中公新書, 1981).

[2] 京都大学工学部電気系教室編, 電気電子工学基礎実習.

第2章　常微分方程式の数値解法

2.1　目的

自然現象や社会現象の数理モデルには，多くの場合微分方程式が現れる．それゆえ，微分方程式を解くことは数学的な興味だけでなく現象を理解する上で重要であるが，限られた場合を除いて解析的に解を得ることはできないため，数値解析の必要性が生じる．本節では，微分方程式の中でも，独立変数をひとつしか持たない常微分方程式の数値解析手法について学習する．

独立変数を t，未知変数を $\boldsymbol{u}(t)$ とする．ただし，$\boldsymbol{u}$ は n 次元のベクトルで，$\boldsymbol{u} = (u_1, u_2, ..., u_n)$ とする．物理的には t は時間変数を表すことが多い．次の 1 階の常微分方程式の初期値問題を考える：

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \boldsymbol{f}(t, \boldsymbol{u}(t)), \quad \boldsymbol{u}(0) = \boldsymbol{u}_0. \tag{2.1}$$

ここで，$\boldsymbol{f} = (f_1, f_2, ..., f_n)$ は既知関数あるいは作用素である．特に，$\boldsymbol{f}$ が $\boldsymbol{u}$ に対して線形（あるいは非線形）であるとき，常微分方程式は線形（あるいは非線形）であるという．また，$\boldsymbol{f}$ が t に依存しないとき，常微分方程式は自励系であるという．常微分方程式の初期値問題の解の存在や一意性については本節の範囲を超えるため，興味がある人は適宜参考文献（例えば [1, 2]）を参照せよ．表記の簡略化のため，必要に応じて $\mathrm{d}\boldsymbol{u}/\mathrm{d}t$ を $\boldsymbol{u}'$ と表す．

常微分方程式の例について簡単に知っておくと理解の一助となるので，いくつか代表的なものを紹介しておく．なお，下記の二つの例はともに線形問題である．

例 2.1.1 質点の運動

x 軸上を運動する質量 m の質点を考える．時間変数を t とし，質点の位置と速度をそれぞれ $x(t)$ および $v(t)$ とする．質点には速度に比例する摩擦力 $-cv(t)$ $(c > 0)$ が働く．また，初期位置と初期速度はそれぞれ $x(0) = x_0$ および $v(0) = v_0$ である．すでに学習しているように，この質点の運動を表す常微分方程式と初期条件は

$$x' = v, \quad mv' = -cv, \tag{2.2a}$$

$$x(0) = x_0, \quad v(0) = v_0, \tag{2.2b}$$

で与えられる．式 (2.1) の表記に合わせると，$u_1 = x$, $u_2 = v$, $f_1 = v$, $f_2 = -(c/m)v$ となる．一方で，物体の運動方程式は x に対する 2 階の常微分方程式と見ることができることに注意する．この例から，1 つの n 階の常微分方程式は，n 個の 1 階常微分方程式のシステムに置き換えることができることが分かる．

例 2.1.2 熱方程式

t を時間，x を位置変数とし，空間一次元の偏微分方程式を考える．具体的には，一様な材質からなる長さ L の棒の伝熱問題，すなわち棒の温度 $T(t, x)$ の振舞いを調べる．棒の温度は初期に $\bar{T}(x)$ とし，$x = 0$ および L における温度は $T(t, 0) = T_0$ および $T(t, L) = T_L$ で一定とする．また，初期条件は $\bar{T}(0) = T_0$ および $\bar{T}(L) = T_L$ を満たしている．このとき，棒の温度 $T(t, x)$ が以下の熱方程式に従う：

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}, \tag{2.3a}$$

$$\text{境界条件: } T(t, 0) = T_0,\ T(t, L) = T_L, \tag{2.3b}$$

$$\text{初期条件: } T(0, x) = \bar{T}(x). \tag{2.3c}$$

ただし $\alpha(> 0)$ は温度伝導率である．式 (2.3) は偏微分方程式なので本節の学習の範囲外と思うかもしれないが，以下のように考えると式 (2.3) も本節の対象となっていることが分かる．まず，区間 $x \in [0, L]$ を $n - 1$ 個に分割しよう．すなわち，整数 $i = 1, 2, ..., n$ に対して $x_i = (i - 1)\Delta x$ $(\Delta x = L/(n-1))$ を定義する．合わせて，$T_i(t) = T(t, x_i)$ と表しておこう．式 (2.3a) の右辺は，2 階の中心差分を用いて以下のように近似できる：

$$\alpha \frac{\partial^2 T}{\partial x^2} \approx \alpha \frac{T_{i-1}(t) - 2T_i(t) + T_{i+1}(t)}{\Delta x^2}. \tag{2.4}$$

この近似の下で，$\boldsymbol{u} = (T_1, T_2, ..., T_n)$ と考えれば，(2.3a) は n 個の連立常微分方程式になっていることが分かるだろう．もちろん，精度良く解を近似するためには n を大きくする必要があるため，偏微分方程式を解く際は巨大な連立方程式になることは想像に難くない．なお，境界条件の議論や $\boldsymbol{f}$ の具体形までは本節では紹介しない．

上記二つの例から想像ができるように，我々の身の回りの現象に対する数理モデルの多くは常微分方程式と結びついている．本節では，その数値解法の初歩的な部分を自分で構築し，実際の問題に適用できるようになることを目指す．

課題 1 身の回りにある現象からひとつ選んで，例に倣って常微分方程式の初期値問題を定式化せよ．

2.2 離散化

まず変数の離散化について述べる．これは，例 2.1.2 でも既に登場しているように，微分方程式を数値的に解く場合にしばしば行われる方法である．

簡単のため，$t \in [0, T]$ で定義される 1 変数の常微分方程式を考える：

$$u' = f(t, u(t)), \quad u(0) = u_0. \tag{2.5}$$

ここで，$n = 0, ..., N$ とし（n の意味が 2.1 節と異なることに注意），以下の離散変数を導入する：

$$t_n = n\Delta t, \quad u_n = u(t_n), \quad (\Delta t = T/N). \tag{2.6}$$

これは t を等間隔に分割した離散化の一例であり，t_n の形は必ずしも (2.6) でなくてよい．Δt をステップ幅と呼ぶ．これらの離散変数を用いると，例えばテイラー展開を用いて u_n を

$$\begin{aligned} u_n &= u(t_{n-1} + \Delta t) \\ &= u(t_{n-1}) + u'(t_{n-1})\Delta t + O(\Delta t^2), \end{aligned} \tag{2.7}$$

と表すことで，u' は

$$u'(t_{n-1}) = \frac{u_n - u_{n-1}}{\Delta t} + O(\Delta t), \tag{2.8}$$

と近似される．つまり，離散化した変数を用いれば，式 (2.5) は

$$u_n \approx u_{n-1} + f(t_{n-1}, u(t_{n-1}))\Delta t, \tag{2.9}$$

と変形できる．ここで，式 (2.9) の右辺の第二項目は $f(t_{n-1}, u(t_{n-1}))$ の代わりに $f(t_n, u(t_n))$ としても構わない．どちらを採用するか（あるいは他の形を採用するか）は数値解析手法の性質を決める重要な問題であり，次節以降で議論する．ここでは，初期条件 u_0 が与えられれば u_n $(n = 1, 2, ..., N)$ が式 (2.9) により逐次的に求められることに気づけば十分である．

2.3 代表的な解法

ここではいくつかの代表的な数値解法やその分類について説明する．ただし，u_m $(m = 0, 1, ..., n-1)$ は既知として，u_n を求める方法を記述する．また，ステップ幅は Δt で一定とする．必要に応じて，$f_n = f(t_n, u_n)$ と簡略化した表記を用いる．

2.3.1 オイラー法

さて，再び (2.5) に立ち戻ろう．この式を $t \in [t_{n-1}, t_n]$ の範囲で積分すると

$$u_n - u_{n-1} = \int_{t_{n-1}}^{t_n} f(\tau, u(\tau))\mathrm{d}\tau, \tag{2.10}$$

となる．$\Delta t = t_n - t_{n-1}$ が十分小さければ，右辺の被積分関数の値は端点の値やそれらの平均で置き換えてもよさそうである．そこでまず，$t \in [t_{n-1}, t_n]$ における f を $f(t_{n-1}, u_{n-1})$ で置き換えてみよう．$f(t_{n-1}, u_{n-1})$ は積分変数 τ に依存しないので，得られる結果は

$$u_n \approx u_{n-1} + f(t_{n-1}, u_{n-1})\Delta t, \tag{2.11}$$

となり，これは式 (2.9) に他ならない．これを**前進オイラー法**と呼び，常微分方程式の数値解法の中で最も単純なものである．前進オイラー法では，未知数である u_n が既知である u_{n-1} のみを使って陽的に表されているため，**陽解法 (explicit method)** と呼ばれる．

一方，$t \in [t_{n-1}, t_n]$ における f を $f(t_n, u_n)$ で近似すると，

$$u_n \approx u_{n-1} + f(t_n, u_n)\Delta t, \tag{2.12}$$

を得る．式 (2.12) は**後退オイラー法**と呼ばれる．前進オイラー法と大きく異なるのは，式 (2.12) が u_n に対して陰的に表されていることであり，このような方法を**陰解法 (implicit method)** という．つまり，$u_n = ...$ と u_n について書き下すためには，式 (2.12) を u_n について解かなくてはならない．f が u_n に対して線形であればこれは容易だが，非線形である場合は，非線形方程式を解くためにニュートン法などの反復解法が必要となることが多い．

前進オイラー法は後退オイラー法も導出過程で Δt^2 の項を無視しているので，数値解法の次数は 1 次精度である．一見すると後退オイラー法を用いるうまみは無い．しかし，実際には陰解法の方が陽解法に比べて安定であり，問題の性質によっては陰解法が好まれる場合がある．なお，オイラー法は u_n を求めるときに u_{n-1} より過去の情報（例えば u_{n-2}）を用いていない．このような方法を**一段法 (one step method)** と呼ぶ．

2.3.2 クランク・ニコルソン法

オイラー法は 1 次精度であったが，これを 2 次精度に向上させる方法が**クランク・ニコルソン法**であり，これは**台形法**や**2 次のアダムス・ムルトン法**とも呼ばれる．アダムス・ムルトン法については 2.3.3 節で詳しく紹介する．式 (2.10) の中の f を，積分範囲の二つの端点

$$(t_{n-1}, f_{n-1}), \quad (t_n, f_n),$$

を通る 1 次関数で近似しよう．このとき f は

$$f(\tau, u(\tau)) \approx \frac{\tau - t_{n-1}}{t_n - t_{n-1}} f_n + \frac{\tau - t_n}{t_{n-1} - t_n} f_{n-1}, \tag{2.13}$$

と表すことができる．式 (2.13) は τ の 1 次関数なので，式 (2.10) の積分は容易に実行できて，

$$\int_{t_{n-1}}^{t_n} f(\tau, u(\tau)) \mathrm{d}\tau \approx \frac{\Delta t}{2}(f_{n-1} + f_n), \tag{2.14}$$

を得る．つまり，u_n は

$$u_n \approx u_{n-1} + \frac{\Delta t}{2}(f_{n-1} + f_n), \tag{2.15}$$

であり，これがクランク・ニコルソン法である．

式 (2.13) は f のテイラー展開において $O(\Delta t^2)$ を無視したものに他ならないので，式 (2.14) は結果的に $O(\Delta t^3)$ を無視している．つまり，クランク・ニコルソン法は 2 次精度の陰解法である．また，式 (2.15) の右辺に (u_n, u_{n-1}) しか含まれていないので，クランク・ニコルソン法は一段法である．

2.3.3 多段法

上記に紹介した一段法に対して，多段にして数値解析の精度の次数を向上させる**多段法（multi step method）**がある．式 (2.10) の中の f を，τ に関する N 次多項式 $p_N(\tau)$ で近似する（N の意味は式 (2.6) と異なることに注意せよ）:

$$\int_{t_{n-1}}^{t_n} f(\tau, u(\tau)) \mathrm{d}\tau \approx \int_{t_{n-1}}^{t_n} p_N(\tau) \mathrm{d}\tau. \tag{2.16}$$

$p_N(\tau)$ が τ の多項式なので，式 (2.16) の積分は解析的に実行できることに注意しよう．

p_N の構成方法は，u_n を用いるかどうかで二つに大別でき，それぞれ陽解法および陰解法となる．まず，最も一般的な陽解法について述べる．$p_N(\tau)$ を Lagrange 補間により構成することを考えよう．これには $N+1$ 個の点が必要である．陽解法なので，

$$(t_{n-1}, f_{n-1}), (t_{n-2}, f_{n-2}), ..., (t_{n-N-1}, f_{n-N-1}),$$

を使うのが自然だろう．結果として，例えば $N = 1$ のときは

$$u_n \approx u_{n-1} + \frac{\Delta t}{2}(3f_{n-1} - f_{n-2}), \tag{2.17}$$

また，$N = 2$ のときは

$$u_n \approx u_{n-1} + \frac{\Delta t}{12}(23f_{n-1} - 16f_{n-2} + 5f_{n-3}), \tag{2.18}$$

となる．式 (2.17) および式 (2.18) をそれぞれ 2 次および 3 次の**アダムス・バッシュフォース法**と呼ぶ．このような高精度化の考え方は，数値積分法と密接に関連している．

次に陰解法について述べる．先の陽解法の例と同様に $N+1$ 個の点を用いて f を Lagrange 補間するが，ここでは

$$(t_n, f_n), (t_{n-1}, f_{n-1}), ..., (t_{n-N}, f_{n-N}),$$

を使う．f_n すなわち $f(t_n, u_n)$ を用いて多項式を構成するため，結果が陰的になることが想像できるだろう．また，$N = 1$ のときはクランク・ニコルソン法と一致するため，これは一段法であることも分かる．$N = 2$ のときは，

$$u_n \approx u_{n-1} + \frac{\Delta t}{12}(5f_n + 8f_{n-1} - f_{n-2}), \quad (2.19)$$

を得る．式 (2.19) を 3 次の**アダムス・ムルトン法**と呼び，これは陰的な多段法である．

さて，アダムス・バッシュフォース法は陽解法なので，f の形に依らず簡単に構成できる．一方，アダムス・ムルトン法は陰解法なので安定性は高いが，場合によっては非線形方程式を解く必要がある．また，過去に参照できる点を制限した場合，アダムス・ムルトン法の方が精度が高い．これは，過去に参照している点が f_{n-2} である 2 次精度のアダムス・バッシュフォース法 (2.17) と 3 次精度のアダムス・ムルトン法 (2.19) を比較すれば分かる．これら二つの方法の長所を組み合わせた方法として，**予測子・修正子法**がある．予測子・修正子法では，まずアダムス・バッシュフォース法で u_n の予測値 $\tilde{u}_n$ を求めておく．次に，アダムス・ムルトン法 (2.19) の右辺で，f_n に含まれる u_n の代わりに $\tilde{u}_n$ を用いて，修正値として u_n を求める．予測子・修正子法は完全に陽的な方法であることに注意しよう．また，修正値を求める段階のアダムス・ムルトン法を繰り返すこともできる．この操作は反復法により非線形方程式 (2.19) を解くことに相当するため，数値解法は陰的になる．ただし，この反復法が収束するためには，予測値 $\tilde{u}_n$ が u_n に十分近く，また Δt が十分小さい必要がある（[3] の 5.2 章参照）．

この節で述べた多段法は，後に述べるルンゲ・クッタ法に比べて 1 ステップにおける f の計算回数が少ないため，全計算時間における f の計算の割合が大きい問題では有用である．

課題 2 式 (2.18) や式 (2.19) にならって，4 次のアダムス・バッシュフォース法とアダムス・ムルトン法を構成せよ．また，4 次のアダムス・バッシュフォース法と 4 次のアダムス・ムルトン法を用いて，予測子・修正子法を構成せよ．

2.3.4 高次精度一段法

これまで見てきたように，多段法を用いることで数値計算の精度の次数を向上させることができる．しかし，多段法においては，問題で与えられた初期値以外に，計算を開始するにあたって必要な初期値があり，これらを別途求めておく必要がある．これには，より低次の多段法を逐次的に適用していくことで対応できるが，計算全体の精度が低次部分の計算結果に依存してしまうため，例えばステップ幅を非一様にする工夫などが必要となる．このような問題を回避するためには，以下の高次精度一段法が便利である．

式 (2.11) を少し変形した次の形

$$\begin{cases} u_n = u_{n-1} + \frac{\Delta t}{2}(f_{n-1} + f(t_n, u_n^*)), \\ u_n^* = u_{n-1} + f_{n-1}\Delta t, \end{cases} \quad (2.20)$$

を考えよう．これは，クランク・ニコルソン法 (2.15) に似ているが，右辺で u_n を使う代わりに，$n-1$ ステップ目の情報だけで構成される u_n^* を使っている点が異なる．式 (2.20) は**ホイン法**あるいは 2 次の**ルンゲ・クッタ法**と呼ばれ，高次精度一段法の代表的なものである．ホイン法の考え方は既出の予測子・修正子法と似ており，未知の $f(t_n, u_n)$ を既知の $f(t_n, u_n^*)$ で近似して高次精度化を試みている．もっとも一般的に使われる高次精度一段法は以下の 4 次のルンゲ・クッタ法で，これを単にルンゲ・クッタ法と呼ぶこともある．

4 次のルンゲ・クッタ法は

$$u_n = u_{n-1} + \frac{\Delta t}{6}(F_1 + 2F_2 + 2F_3 + F_4), \tag{2.21a}$$

$$\begin{cases} F_1 = f(t_{n-1}, u_{n-1}), \\ F_2 = f(t_{n-1} + \Delta t/2, u_{n-1} + F_1 \Delta t/2), \\ F_3 = f(t_{n-1} + \Delta t/2, u_{n-1} + F_2 \Delta t/2), \\ F_4 = f(t_n, u_{n-1} + F_3 \Delta t), \end{cases} \tag{2.21b}$$

と書ける．上述の解法はすべて陽解法であることに注意する．一般のルンゲ・クッタ法の構築については [2] を参照するとよい．

課題 3 常微分方程式の初期値問題

$$u' = u, \quad u(0) = 1, \tag{2.22}$$

を考える．解析解は $u = \exp(t)$ で与えられる．式 (2.22) を $t \in [0, 1]$ の範囲で数値的に解いて，その精度を確認してみよう．

まず，ステップ幅を $\Delta t = 1/2^i$，ステップ数を $N = 2^i$，離散化した時間変数を $t_n = n\Delta t$ $(n = 0, ..., N)$ と定義する．また，ステップ幅が $\Delta t = 1/2^i$ の条件の下で得られた数値解を $u_n^{(i)}$ と表す．精度の確認は，i の値を $i = 1, 2, ..., i_{\max}$ の範囲で様々に変更した計算を実行し，それらの結果を比較すればよい．比較には，$t = 1$ における数値解 $u_N^{(i)}$ と解析解との差 $E^{(i)} = |u_N^{(i)} - u(1)|$ を用いる．p を計算手法の次数とし，$E_r^{(i)} = E^{(i)}/E^{(i-1)}$ $(i = 2, ..., i_{\max})$ と定義すると，$E_r^{(i)}$ は i の増加とともに $1/2^p$ に近づくはずである．

前進オイラー法 (FE)，2 次および 3 次のアダムス・バッシュフォース法 (AB2 および AB3)，ホイン法 (H)，4 次のルンゲ・クッタ法 (RK) の 5 種類の方法で，式 (2.22) を $t \in [0, 1]$ の範囲で数値的に解き，理論的に予測される次数を達成していることを確認せよ．結果は分かりやすく表にまとめるとよい．

ヒント： 単精度でなく倍精度で計算すること．$i_{\max}$ の値は自由に決めてよいが，$i \approx 10$ 程度より i が大きくなると，倍精度演算でも結果を正しく評価できない場合がある．また，多段法で必要となる初期条件に関しては，解析解の値を用いても構わない．

2.4 安定性

計算手法の安定性は手法の性質を議論する上で重要な要素のひとつである．ここでは，手法の安定性について簡単な例を題材に紹介する．より詳細な理論を知りたい場合は [2, 3] を参照せよ．常微分方程式の解それ自身の安定性ではなく，計算手法の安定性に着目していることに注意しよう．

手法の安定・不安定性を明確にするために，以下では $\lim_{t\to\infty} |u(t)| = \infty$ とならないような問題を考える．また，議論を簡単にするため，差分式が以下の形で書けるとする：

$$u_n = a_1 u_{n-1} + a_0. \tag{2.23}$$

例えば，線形方程式に対する前進オイラー法 (2.11) や後退オイラー法 (2.12) が式 (2.23) に対応することは容易に確認できるだろう．数値計算手法が不安定であるとき，$\lim_{n\to\infty} |u_n| = \infty$ となるはずである．そこで，式 (2.23) を満たす数列 $\{u_n\}$ を，$u_n = \lambda^n$ の形で探してみよう．$u_n = \lambda^n$ を代入して整理すると，式 (2.23) は

$$\lambda^n - a_1 \lambda^{n-1} - a_0 = 0, \tag{2.24}$$

と書ける．これを特性方程式と呼ぶ．まず，同次解を求める．$\lambda \neq 0$ として，式 (2.24) の同次方程式

$$\lambda^n - a_1 \lambda^{n-1} = 0, \tag{2.25}$$

を解くと，$\lambda = a_1$ を得る．一方，式 (2.23) の非同次解として $u_n = a_0/(1 - a_1)$ が見つかる．すなわち，(2.23) の解は，c_1 を未定係数として，

$$u_n = c_1 a_1^n + a_0/(1 - a_1), \tag{2.26}$$

である．u_0 が与えられているので，$c_1 = u_0 - a_0/(1 - a_1)$ と決まる．結局，式 (2.23) の解は，

$$u_n = \left(u_0 - \frac{a_0}{1 - a_1}\right) a_1^n + \frac{a_0}{1 - a_1}, \tag{2.27}$$

である．もし $|a_1| > 1$ だとすると，式 (2.27) は $n \to \infty$ で発散してしまうことが分かる．それでは，例題でこのことを確認してみよう．

例 2.4.1 オイラー法の安定性

$t \in [0, \infty)$ における常微分方程式の初期値問題

$$u' = -\alpha u + \beta, \quad u(0) = u_0, \quad (\alpha > 0), \quad (2.28)$$

を考える．式 (2.28) の解は

$$u(t) = \left(u_0 - \frac{\beta}{\alpha}\right) \exp(-\alpha t) + \frac{\beta}{\alpha}, \qquad (2.29)$$

であり．$\lim_{t\to\infty} u(t) = \beta/\alpha$ となる．

まず，式 (2.28) を前進オイラー法で離散化してみよう．このとき，

$$\begin{aligned} u_n &= u_{n-1} + (-\alpha u_{n-1} + \beta)\Delta t \\ &= (1 - \alpha\Delta t)u_{n-1} + \beta\Delta t, \end{aligned} \qquad (2.30)$$

が得られるので，$a_1 = 1 - \alpha\Delta t$, $a_0 = \beta\Delta t$ である．つまり，$|a_1| \le 1$ となるためには，$0 \le \Delta t \le 2/\alpha$ が要求される．一方，後退オイラー法では，

$$\begin{aligned} & u_n = u_{n-1} + (-\alpha u_n + \beta)\Delta t \\ \Longrightarrow \quad & u_n = \frac{u_{n-1}}{1 + \alpha\Delta t} + \frac{\beta\Delta t}{1 + \alpha\Delta t}, \end{aligned} \qquad (2.31)$$

と変形できるので，$a_1 = (1 + \alpha\Delta t)^{-1}$, $a_0 = (\beta\Delta t)/(1 + \alpha\Delta t)$ である．$\alpha > 0$ に対して，後退オイラー法では常に $|a_1| < 1$ が満たされるため，後退オイラー法は任意の $\Delta t > 0$ に対して安定である．

図 2.1 と 2.2 にそれぞれ前進オイラー法と後退オイラー法による数値解を示す．パラメータは $u_0 = 1$, $\alpha = 10$, $\beta = 1$ として，$\Delta t = 0.01$, 0.19, 0.205 とした．図 2.1 より，前進オイラー法で $\Delta t = 0.205 (> 2/\alpha = 0.2)$ の場合に，解が振動しながら発散していく様子が分かる．この振動は，$a_1 < -1$ であることと式 (2.27) の形からも想像できるだろう．一方，図 2.2 より，後退オイラー法では数値解の発散は見られない．この一例からも分かるように，一般に陰解法の方が安定性が高い．

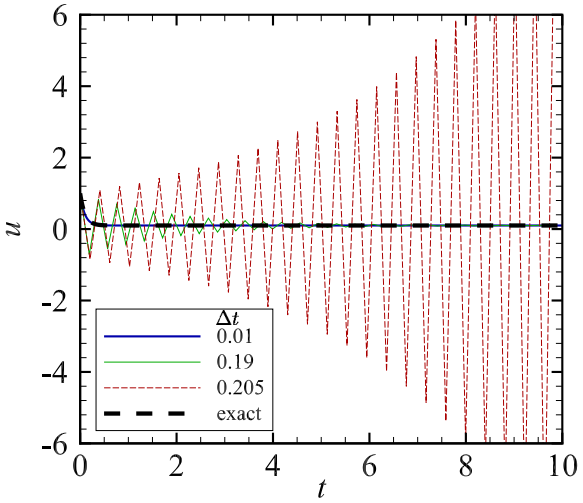


図 2.1: 前進オイラー法による式 (2.28) の数値解．

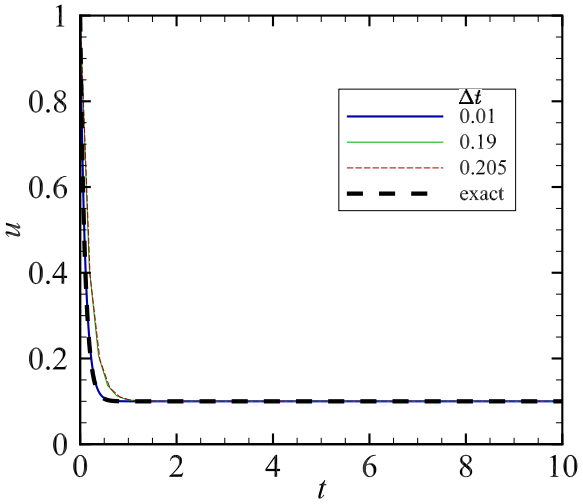


図 2.2: 後退オイラー法による式 (2.28) の数値解．

上記の例のように，陽解法を用いた場合，Δt を十分小さくとらなければ安定した数値解が得られないことがある．このような微分方程式は**硬い (stiff)** と表現される．式 (2.28) の場合は，α が Δt の大きさに制限を加えていることは明らかだが，一般の非線形連立常微分方程式系を扱う場合は安定性を議論することは容易ではない．

課題 4 式 (2.28) に対して，クランク・ニコルソン法，予測子・修正子法，ホイン法を適用した場合に安定性がどのように変わるか示し，数値的に確認せよ．

課題 5 これまで見てきたように，数値計算手法

は自由に作ることができる．例えば，式 (2.10) の代わりに，

$$u_n = u_{n-2} + \int_{t_{n-2}}^{t_n} f(\tau, u(\tau))\mathrm{d}\tau, \qquad (2.32)$$

として，$t \in [t_{n-2}, t_n]$ の範囲で $f \approx f_{n-1}$ と近似すれば，

$$u_n = u_{n-2} + 2f(t_{n-1}, u_{n-1})\Delta t, \qquad (2.33)$$

を得る．式 (2.33) は陽的な多段法であり一見問題なさそうだが，$t \in [0, \infty)$ における常微分方程式の初期値問題

$$u' = -2u + 1, \quad u(0) = 1, \qquad (2.34)$$

を解くことができない．このことを本節の議論にのっとって示し，数値計算で確かめてみよ．

ヒント：特性方程式は二次方程式になる．その解を λ_1, λ_2 として，$|\lambda_i| > 1$ を示せばよい．数値計算では，与えられた Δt に対しある $\tilde{t}$ が決まり，$t > \tilde{t}$ の範囲で数値解が発散していく様子が観察できるはずである．$\tilde{t}$ の値は Δt が小さいほど大きくなる．

2.5 爆発解

有限の T に対して，$t \to T$ で $u = u(t)$ が発散するとき，解は**爆発**するという．数値計算上では，微小なステップ幅に対しても大きな u の変動が生じるため，爆発解の計算は一般には難しい．例えば，ステップ幅が解の振舞いに応じて自動的に変化するようなアルゴリズムが必要である．ここでは，変数変換することで爆発解の数値計算が簡単になることを示そう [4]．

スカラー $u(t)$ に対する常微分方程式の初期値問題

$$u' = f(t, u), \quad u(0) = u_0, \qquad (2.35)$$

を考える．式 (2.35) が，ある t の値（あるいは t の範囲）で u' が非常に大きくなる性質を持っているとしよう．このような場合，独立変数として，t の代わりに曲線の長さを用いることが考えられる．t は新しい従属変数になることに注意する．t-u 平面において解が描く軌道に沿った長さを s とする．つまり，

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \sqrt{1 + (u')^2} = \sqrt{1 + (f(t,u))^2}, \qquad (2.36)$$

である．このとき，式 (2.35) は $u = u(s)$ および $t = t(s)$ を用いて

$$\frac{\mathrm{d}u}{\mathrm{d}s} = \frac{f(t,u)}{\sqrt{1 + (f(t,u))^2}}, \quad u(0) = u_0, \qquad (2.37a)$$

$$\frac{\mathrm{d}t}{\mathrm{d}s} = \frac{1}{\sqrt{1 + (f(t,u))^2}}, \quad t(0) = 0, \qquad (2.37b)$$

と表される．式 (2.37) は t, u を従属変数，s を独立変数とした連立常微分方程式に他ならない．式 (2.35) に比べて式 (2.37) が数値的に扱い易いことを，以下の例で確認してみよう．

例 2.5.1 オイラー法による爆発解の計算

非線形な常微分方程式の初期値問題

$$u' = u^2, \quad u(0) = 1, \qquad (2.38)$$

を考える．解析解は $u = (1-t)^{-1}$ なので，$t \to T = 1$ で $u \to \infty$ である．前進オイラー法により数値解析した結果を図 2.3 に示す．変数変換 (2.37) を使わない場合，数値解は $t = T = 1$ を超えた範囲にまで進んでいる．本来このような挙動は許されない．一方，変数変換 (2.37) を用いた結果ではこのような不具合は生じておらず，解析解により近い数値解を得られていることが分かる．

課題 6 生物の個体数の増減に関する数理モデルを考えよう．$t \geq 0$ を時刻，$u(t)$ を時刻 t における個体数，$u(0) = u_0 > 0$ を初期個体数とする．個体数の時間変化 u' は，個体数が多いほど大きいと考えるのが自然である．そこで，$u' = ru$ $(r > 0)$ と仮定するのが最も単純であろう．しかし，ある種の生物では，u が小さいと交配相手が見つかる確率が減り，結果として r が小さくなるという仮定が成り立つ．逆に，u が大きいと交配相手が見つかる確率が高まるので，r は増加する．これをアリー効果という．一例として，

$$u' = (u-1)u, \quad u(0) = u_0, \qquad (2.39)$$

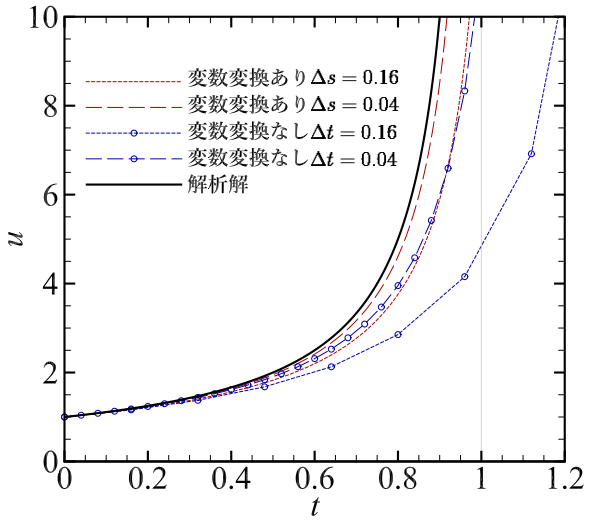


図 2.3: 前進オイラー法による式 (2.38) の数値解.

という数理モデルが考えられるだろう．ただし，式 (2.39) は適切に無次元化されているとする．式 (2.39) を数値解析し，解の振舞いを調べよ．式 (2.39) は解析解を求めることができるが，ここではあえて解析解を知らないふりをして，どのような結果になるか予想しながら取り組んでみよう．

ヒント：$u_0 > 1$, $u_0 = 1$, $u_0 < 1$ の 3 つの場合で，解の振舞いは大きく異なる．

2.6 カオス

最後に，解が興味深い振舞いをする例として，カオスを紹介する．数値計算の演習なので，物理的な背景や数学的な事柄はここでは紹介しない．興味がある人は，例えば [5] や，より数学的に発展的な内容については [6] を勉強するとよい．下記に示すローレンツ方程式について，物理的な背景や数学的な性質が紹介されている．

3 つの未知数 $x(t)$, $y(t)$, $z(t)$ が，次の**ローレンツ方程式**

$$x' = \sigma(y - x), \quad x(0) = x_0, \tag{2.40a}$$
$$y' = rx - y - xz, \quad y(0) = y_0, \tag{2.40b}$$
$$z' = xy - bz, \quad z(0) = z_0, \tag{2.40c}$$

に従うとしよう．σ, r, b はパラメータであり，すべて正とする．式 (2.40) は非線形連立常微分方程式であり，本章で紹介してきた数値計算手法で解析できる．一見シンプルな方程式であるが，その解の挙動は非常に複雑でカオス的である．なお，文献 [5] では，カオスを「決定論的なシステムにおける非周期的な長時間挙動であり，初期条件への鋭敏な依存性を示す」と定義している．式 (2.40) には乱数が用いられていないため，決定論的なシステムであることは明らかである．以下の課題では，カオス的な性質のうち，非周期的な長時間挙動と初期条件への鋭敏な依存性について調べることにしよう．なお，初期条件に鋭敏に依存する方程式は，解が数値計算上の誤差にも強く影響を受ける．そのため，数値計算の精度にも十分に気を配る必要がある．

課題 7 ローレンツ方程式 (2.40) において，$\sigma = 10$, $b = 8/3$, $r = 28$, $x_0 = 1 + \epsilon$, $y_0 = 0$, $z_0 = 0$ として，数値解析を行う．手法としては，前進オイラー法とルンゲ・クッタ法を用いよ．また，計算領域は $t \in [0, 100]$ とする．

1. まず，$\epsilon = 0$ とする．$x(t)$ を t の関数として図示せよ．また，$(x(t), y(t), z(t))$ の軌道を x-y-z 空間上に図示し，非周期的な長時間挙動を確認せよ．

2. $\Delta t = 0.01 \times 2^{-i}$ $(i = 0, ..., 13)$ として，前進オイラー法とルンゲ・クッタ法でそれぞれの i に対する数値解を求めよ．$t = 15, 30, 60$ の 3 つの時刻において，Δt を小さくするとともに $x(t)$ がどのように変化するか図示し，数値計算の誤差の影響が $x(t)$ に与える影響を議論せよ．

3. $\epsilon = 0$, 0.1, 0.01, 0.001 として，ϵ が $x(t)$ の振舞いに与える影響を議論せよ．ただし，解の振舞い対する数値計算の誤差の影響が ϵ の違いによる影響より小さい必要があるため，ルンゲ・クッタ法を用いて Δt も十分小さく設定すること．

参考文献

[1] 水野克彦編，“解析学”，学術図書出版 (2001)

[2] 齊藤宣一，“数値解析入門”，東京大学出版（2012）

[3] G. H. Golub and J. M. Ortega, “Scientific Computing and Differential Equations: An Introduction to Numerical Methods”, Academic Press (1992)

[4] 伊理正夫，藤野和建，“数値計算の常識”，共立出版（1985）

[5] ストロガッツ，“非線形ダイナミクスとカオス”，田中久陽，中尾裕也，千葉逸人訳，丸善出版（2015）

[6] アリグッド，サウアー，ヨーク，“カオス第 1 - 3 巻，力学系入門”，津田一郎監訳，シュプリンガー・ジャパン（2006）

[辻徹郎]

第3章　熱方程式の差分法

3.1　目的

この章では熱方程式（拡散方程式）の差分法について取り扱う [1, 2]。

3.2　偏微分方程式

偏微分方程式は様々な現象の数学的記述に現れる。偏微分方程式とは、2 つ以上の独立変数と未知関数およびその未知関数の偏微分を含むような方程式である。独立変数を x, t とし、未知関数を $u(x,t)$ とすると、偏微分方程式の一般系は

$$f\left(t, x, u, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial t^2}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial t \partial x}, \cdots\right) = 0 \tag{3.1}$$

となる。具体例としては、波動の時間発展を記述する波動方程式や、静電ポテンシャルの関数形を与えるラプラス方程式、そして本演習で扱う拡散方程式などがある。偏微分方程式を解くとは、決められた (x,t) に対して $u(x,t)$ を求めることである。解析的に解くのが困難な場合、数値計算を用いて近似的に解いてやることになる。

3.3　拡散方程式

拡散とは微粒子などが自発的に散らばり広がっていく現象である。1 次元空間内での拡散を考える。微粒子の濃度を $C(x,t)$ とする。微粒子の流れを $J(x,t)$ とすると、粒子数の保存を表す連続の式は

$$\frac{\partial}{\partial t}C(x,t) = -\frac{\partial}{\partial x}J(x,t) \tag{3.2}$$

と書ける。また、流れ $J(x,t)$ は濃度勾配に比例すると仮定すると

$$J(x,t) = -D\frac{\partial}{\partial x}C(x,t) \tag{3.3}$$

と表される。これらから拡散方程式

$$\frac{\partial}{\partial t}C(x,t) = D\frac{\partial^2}{\partial x^2}C(x,t) \tag{3.4}$$

が得られる。

以下では $D = 1$ とした拡散方程式

$$\frac{\partial}{\partial t}u(x,t) = \frac{\partial^2}{\partial x^2}u(x,t) \tag{3.5}$$

を考える。

拡散方程式 (3.5) を解くためには初期条件

$$u(x,0) = u_0(x) \tag{3.6}$$

と境界 $x = 0, L$ における境界条件を考える必要がある。代表的な境界条件には

- Dirichlet 境界条件: 境界上での u の値が指定されている。

$$u(0,t) = u_{\mathrm{L}} \tag{3.7}$$
$$u(L,t) = u_{\mathrm{R}} \tag{3.8}$$

- Neumann 境界条件: 境界上での u の微分の値が指定されている。

$$\frac{\partial u}{\partial x}(0,t) = J_{\mathrm{L}} \tag{3.9}$$
$$\frac{\partial u}{\partial x}(L,t) = J_{\mathrm{R}} \tag{3.10}$$

の 2 つがある。

3.4　数値解法

ここでは、時間と空間を離散化して拡散方程式 (3.5) を数値的に解くことを考える。時間と空間の刻み幅をそれぞれ Δt, Δx とする ($L = N\Delta x$)。離散化された時空間上での平均濃度を

$$u_j^n \equiv \frac{1}{\Delta x}\int_{(j-1)\Delta x}^{j\Delta x} u\left(y, n\Delta t\right) dy \tag{3.11}$$

と書く。u_j^n は区間 $[(j-1)\Delta x, j\Delta x]$ に対して割り当てられていることに注意が必要である。また、u_0^n, u_{N+1}^n は境界条件のために用いる。方程式 (3.2) を $[n\Delta t, (n+1)\Delta t]$ で積分すると

$$\begin{aligned} & u\left(x,(n+1)\Delta t\right) - u\left(x,n\Delta t\right) \\ = \; & -\int_{n\Delta t}^{(n+1)\Delta t} \frac{\partial}{\partial x} J(x,s) ds \end{aligned} \tag{3.12}$$

となる。(3.12) の右辺の積分を時刻 $n\Delta t$ の値で近似すると

$$\begin{aligned} & u\left(x,(n+1)\Delta t\right) - u\left(x,n\Delta t\right) \\ \simeq \; & -\Delta t \frac{\partial}{\partial x} J\left(x,n\Delta t\right) \end{aligned} \tag{3.13}$$

となる。この場合

$$\begin{aligned} & u_j^{n+1} - u_j^n \\ \simeq \; & -\frac{1}{\Delta x} \int_{(j-1)\Delta x}^{j\Delta x} \Delta t \frac{\partial}{\partial y} J\left(y,n\Delta t\right) dy \\ = \; & -\frac{\Delta t}{\Delta x} \left[J(j\Delta x, n\Delta t) - J((j-1)\Delta x, n\Delta t) \right] \end{aligned} \tag{3.14}$$

となる。ただし (3.3) より

$$J(j\Delta x, n\Delta t) = -\frac{\partial u}{\partial x}(j\Delta x, n\Delta t) \tag{3.15}$$

である。この量はさらに

$$J(j\Delta x, n\Delta t) \simeq -\frac{u_{j+1}^n - u_j^n}{\Delta x} \tag{3.16}$$

と近似されるので、差分方程式

$$u_j^{n+1} - u_j^n \simeq \frac{\Delta t}{\Delta x^2} \left(u_{j-1}^n - 2u_j^n + u_{j+1}^n \right) \tag{3.17}$$

を得る。

一方、(3.12) において積分をより精度の良い台形公式で近似すると

$$\begin{aligned} & u\left(x,(n+1)\Delta t\right) - u\left(x,n\Delta t\right) \\ \simeq \; & -\frac{\Delta t}{2} \left[\frac{\partial}{\partial x} J\left(x,(n+1)\Delta t\right) + \frac{\partial}{\partial x} J\left(x,n\Delta t\right) \right] \end{aligned} \tag{3.18}$$

となる。これを用いると

$$\begin{aligned} u_j^{n+1} - u_j^n \simeq \; & \frac{1}{2} \frac{\Delta t}{\Delta x^2} \left(u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1} \right) \\ & + \frac{1}{2} \frac{\Delta t}{\Delta x^2} \left(u_{j-1}^n - 2u_j^n + u_{j+1}^n \right) \end{aligned} \tag{3.19}$$

を得る。

(3.17) は $\left\{u_j^n\right\}_{j=1}^N$ から $\left\{u_j^{n+1}\right\}_{j=1}^N$ を直接求める漸化式になっている。このような数値解法を陽解法と呼ぶ。一方、(3.19) は右辺にも $\left\{u_j^{n+1}\right\}_{j=1}^N$ が現れており、$\left\{u_j^{n+1}\right\}_{j=1}^N$ の連立方程式として解いてやる必要がある。このような未知変数と既知変数の混在した差分方程式による数値解法を陰解法と呼ぶ。

3.5 オイラー陽解法

(3.17) のような最も単純な差分方程式による数値解法をオイラー陽解法と呼ぶ。以下では

$$c \equiv \frac{\Delta t}{\Delta x^2} \tag{3.20}$$

と書くことにする。

初期条件は $\left\{u_j^0\right\}_{j=1}^N$ と表される。また、境界条件は

- Dirichlet 境界条件:

$$u_0^n = u_{\mathrm{L}} \tag{3.21}$$

$$u_{N+1}^n = u_{\mathrm{R}} \tag{3.22}$$

- Neumann 境界条件:

$$u_0^n = u_1^n - J_{\mathrm{L}} \Delta x \tag{3.23}$$

$$u_{N+1}^n = u_N^n + J_{\mathrm{R}} \Delta x \tag{3.24}$$

と表される。

3.5.1 安定性解析

数値計算を行う際、c の値がある値を越えると差分方程式が不安定になることが知られている。(3.17) は線形方程式であるので、波数 k の成分ごとに議論を行える。（任意の解はそれらの重ね合わせとして得られる。）今、(3.17) において波数 k の特解

$$u_j^n = \lambda_k^n e^{ikj} \tag{3.25}$$

を考える。増幅因子 λ_k の絶対値 $|\lambda_k|$ が任意の波数 k に対して 1 より小さければ差分方程式は安定である（即ち、任意の波数のモードは時間とともに減衰する）。この特解を (3.17) に代入すると

$$\begin{aligned} & \lambda_k^{n+1} e^{ikj} \\ = \; & \lambda_k^n e^{ikj} \\ & + c\left(\lambda_k^n e^{ik(j-1)} - 2\lambda_k^n e^{ikj} + \lambda_k^n e^{ik(j+1)}\right) \end{aligned} \tag{3.26}$$

を得る。これより

$$\lambda_k = 1 - 4c\sin^2\frac{k}{2} \tag{3.27}$$

となる。従って、$|\lambda_k| < 1$ が任意の k に対して満たされるためには

$$c \leq \frac{1}{2} \tag{3.28}$$

でなければならない。この条件は、Δx を小さくとるとき、Δt もそれに合わせて小さくしなければならないことを表す。

3.5.2 アルゴリズム

オイラー陽解法のアルゴリズムは Algorithm 1 のようにまとめられる。

Algorithm 1 オイラー陽解法

Input: 初期条件 $\left\{u_j^0\right\}_{j=1}^N$, 境界条件
Output: 時刻 $n\Delta t$ $(n = 1, \cdots, T)$ における $\left\{u_j^n\right\}_{j=1}^N$

1: u_0^0, u_{N+1}^0 を境界条件に従って与える
2: **for** $n = 0$ to $T-1$ **do**
3: $\left\{u_j^n\right\}_{j=1}^N$ から $\left\{u_j^{n+1}\right\}_{j=1}^N$ を (3.17) に従って計算
4: u_0^{n+1}, u_{N+1}^{n+1} を境界条件に従って与える
5: $\left\{u_j^{n+1}\right\}_{j=1}^N$ を出力
6: **end for**

3.6 クランク-ニコルソン法

(3.19) はクランク-ニコルソン法と呼ばれる。この方法はオイラー陽解法よりも精度が良くなっている。(3.19) を書き直すと

$$\begin{aligned} & -\frac{c}{2}u_{j-1}^{n+1} + (1+c)u_j^{n+1} - \frac{c}{2}u_{j+1}^{n+1} \\ = \; & (1-c)u_j^n + \frac{c}{2}u_{j-1}^n + \frac{c}{2}u_{j+1}^n \end{aligned} \tag{3.29}$$

となる。ただし、境界条件は

- Dirichlet 境界条件: $u_0^n = u_{\mathrm{L}}$, $u_{N+1}^n = u_{\mathrm{R}}$ より

$$\begin{aligned} & (1+c)u_1^{n+1} - \frac{c}{2}u_2^{n+1} \\ = \; & (1-c)u_1^n + cu_{\mathrm{L}} + \frac{c}{2}u_2^n \end{aligned} \tag{3.30}$$

$$\begin{aligned} & -\frac{c}{2}u_{N-1}^{n+1} + (1+c)u_N^{n+1} \\ = \; & (1-c)u_N^n + cu_{\mathrm{R}} + \frac{c}{2}u_{N-1}^n \end{aligned} \tag{3.31}$$

- Neumann 境界条件: $u_0^n = u_1^n - J_{\mathrm{L}}\Delta x$, $u_{N+1}^n = u_N^n + J_{\mathrm{R}}\Delta x$ より

$$\begin{aligned} & \left(1+\frac{c}{2}\right)u_1^{n+1} - \frac{c}{2}u_2^{n+1} \\ = \; & \left(1-\frac{c}{2}\right)u_1^n - cJ_{\mathrm{L}}\Delta x + \frac{c}{2}u_2^n \end{aligned} \tag{3.32}$$

$$\begin{aligned} & -\frac{c}{2}u_{N-1}^{n+1} + \left(1+\frac{c}{2}\right)u_N^{n+1} \\ = \; & \left(1-\frac{c}{2}\right)u_N^n + cJ_{\mathrm{R}}\Delta x + \frac{c}{2}u_{N-1}^n \end{aligned} \tag{3.33}$$

である。これらの連立方程式を解く際には LU 分解という方法が用いられる。

3.6.1 安定性解析

3.5.1 と同様に安定性解析を行う。(3.29) において波数 k の特解

$$u_j^n = \lambda_k^n e^{ikj} \tag{3.34}$$

を代入すると、

$$\begin{aligned} & -\frac{c}{2}\lambda_k^{n+1}e^{ik(j-1)} + (1+c)\lambda_k^{n+1}e^{ikj} \\ & -\frac{c}{2}\lambda_k^{n+1}e^{ik(j+1)} \\ = \ & (1-c)\lambda_k^n e^{ikj} + \frac{c}{2}\lambda_k^n e^{ik(j-1)} + \frac{c}{2}\lambda_k^n e^{ik(j+1)} \end{aligned} \tag{3.35}$$

となる。これから

$$\lambda_k = \frac{1-2c\sin^2\frac{k}{2}}{1+2c\sin^2\frac{k}{2}} \tag{3.36}$$

と求まる。$c>0$ より、全ての k に対して $|\lambda_k|<1$ であることがわかり、クランク-ニコルソン法は全ての c について安定であると言える。

3.6.2 LU 分解

(3.29) と境界条件 (3.30), (3.31) あるいは (3.32), (3.33) を合わせた連立方程式は $\boldsymbol{x} \equiv \left(u_1^{n+1}, \cdots, u_N^{n+1}\right)^{\mathsf{T}}$ と既知な量を成分とするベクトル $\boldsymbol{z}$ 及び、ある行列 A を用いて

$$A\boldsymbol{x} = \boldsymbol{z} \tag{3.37}$$

のように表される。A は Dirichlet 境界条件では

$$A = \begin{pmatrix} 1+c & -\frac{c}{2} & 0 & \cdots & 0 \\ -\frac{c}{2} & 1+c & -\frac{c}{2} & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -\frac{c}{2} & 1+c & -\frac{c}{2} \\ 0 & \cdots & 0 & -\frac{c}{2} & 1+c \end{pmatrix} \tag{3.38}$$

Neumann 境界条件では

$$A = \begin{pmatrix} 1+\frac{c}{2} & -\frac{c}{2} & 0 & \cdots & 0 \\ -\frac{c}{2} & 1+c & -\frac{c}{2} & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -\frac{c}{2} & 1+c & -\frac{c}{2} \\ 0 & \cdots & 0 & -\frac{c}{2} & 1+\frac{c}{2} \end{pmatrix} \tag{3.39}$$

となる。また、$\boldsymbol{z}$ は Dirichlet 境界条件では

$$\begin{aligned} z_1 &= (1-c)u_1^n + cu_{\mathrm{L}} + \frac{c}{2}u_2^n \\ z_j &= (1-c)u_j^n + \frac{c}{2}u_{j-1}^n + \frac{c}{2}u_{j+1}^n \\ & \qquad (2 \le j \le N-1) \\ z_N &= (1-c)u_N^n + cu_{\mathrm{R}} + \frac{c}{2}u_{N-1}^n \end{aligned} \tag{3.40}$$

Neumann 境界条件では

$$\begin{aligned} z_1 &= \left(1-\frac{c}{2}\right)u_1^n - cJ_{\mathrm{L}}\Delta x + \frac{c}{2}u_2^n \\ z_j &= (1-c)u_j^n + \frac{c}{2}u_{j-1}^n + \frac{c}{2}u_{j+1}^n \\ & \qquad (2 \le j \le N-1) \\ z_N &= \left(1-\frac{c}{2}\right)u_N^n + cJ_{\mathrm{R}}\Delta x + \frac{c}{2}u_{N-1}^n \end{aligned} \tag{3.41}$$

である。(3.37) から $\boldsymbol{x}$ を求めるのが目標である。

LU 分解とは A を上三角行列 U と下三角行列 L の積 $A=LU$ に分解することである。今、A を

$$A = \begin{pmatrix} a_1 & b_1 & 0 & \cdots & 0 \\ c_1 & a_2 & b_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & c_{N-2} & a_{N-1} & b_{N-1} \\ 0 & \cdots & 0 & c_{N-1} & a_N \end{pmatrix} \tag{3.42}$$

のように書く。

$$L = \begin{pmatrix} \alpha_1 & 0 & 0 & \cdots & 0 \\ c_1 & \alpha_2 & 0 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & c_{N-2} & \alpha_{N-1} & 0 \\ 0 & \cdots & 0 & c_{N-1} & \alpha_N \end{pmatrix} \tag{3.43}$$

$$U = \begin{pmatrix} 1 & \beta_1 & 0 & \cdots & 0 \\ 0 & 1 & \beta_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 0 & 1 & \beta_{N-1} \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix} \tag{3.44}$$

と置くと

$$LU = \begin{pmatrix} \alpha_1 & \alpha_1\beta_1 & 0 & \cdots & 0 \\ c_1 & \alpha_2 + c_1\beta_1 & \alpha_2\beta_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & c_{N-2} & \alpha_{N-1} + c_{N-2}\beta_{N-2} & \alpha_{N-1}\beta_{N-1} \\ 0 & \cdots & 0 & c_{N-1} & \alpha_N + c_{N-1}\beta_{N-1} \end{pmatrix} \tag{3.45}$$

となる。(3.42) と (3.45) を比較すると

$$\begin{aligned}
&\alpha_1 = a_1 \\
&\beta_1 = \frac{b_1}{\alpha_1} \\
&\alpha_2 = a_2 - c_1\beta_1 \\
&\beta_2 = \frac{b_2}{\alpha_2} \\
&\vdots \\
&\alpha_{N-1} = a_{N-1} - c_{N-2}\beta_{N-2} \\
&\beta_{N-1} = \frac{b_{N-1}}{\alpha_{N-1}} \\
&\alpha_N = a_N - c_{N-1}\beta_{N-1}
\end{aligned} \tag{3.46}$$

のように昇順に L と U を求められる。

次に LU 分解を用いて連立方程式 (3.37) を解く方法を説明する。$LU\boldsymbol{x} = \boldsymbol{z}$ において $U\boldsymbol{x} \equiv \boldsymbol{y}$ とおくと

$$L\boldsymbol{y} = \boldsymbol{z} \tag{3.47}$$

となる。この式から $\boldsymbol{y}$ を求め、$U\boldsymbol{x} = \boldsymbol{y}$ から $\boldsymbol{x}$ を求める。まず $L\boldsymbol{y} = \boldsymbol{z}$ は

$$\begin{pmatrix} \alpha_1 & 0 & 0 & \cdots & 0 \\ c_1 & \alpha_2 & 0 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & c_{N-2} & \alpha_{N-1} & 0 \\ 0 & \cdots & 0 & c_{N-1} & \alpha_N \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_{N-1} \\ z_N \end{pmatrix} \tag{3.48}$$

より、$\boldsymbol{y}$ は昇順に

$$\begin{aligned}
y_1 &= \frac{z_1}{\alpha_1} \\
y_2 &= \frac{z_2 - c_1 y_1}{\alpha_2} \\
&\vdots \\
y_{N-1} &= \frac{z_{N-1} - c_{N-2} y_{N-2}}{\alpha_{N-1}} \\
y_N &= \frac{z_N - c_{N-1} y_{N-1}}{\alpha_N}
\end{aligned} \tag{3.49}$$

のように求まる。次に $U\boldsymbol{x} = \boldsymbol{y}$ は

$$\begin{pmatrix} 1 & \beta_1 & 0 & \cdots & 0 \\ 0 & 1 & \beta_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 0 & 1 & \beta_{N-1} \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix} \tag{3.50}$$

より、$\boldsymbol{x}$ は降順に

$$\begin{aligned}
x_N &= y_N \\
x_{N-1} &= y_{N-1} - \beta_{N-1} x_N \\
&\vdots \\
x_2 &= y_2 - \beta_2 x_3 \\
x_1 &= y_1 - \beta_1 x_2
\end{aligned} \tag{3.51}$$

のように求まる。実際の計算では $\{\alpha_j\}$, $\{\beta_j\}$ は配列に保存して置くと良い。

3.6.3 アルゴリズム

以上をまとめて、クランク-ニコルソン法では Algorithm 2 のように計算を行う。

3.7 その他の陰解法

一般に (3.12) において $0 \leq \theta \leq 1$ を用いて

$$\begin{aligned}
u_j^{n+1} - u_j^n = \ & \theta c \left(u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1} \right) \\
& + (1-\theta) c \left(u_{j-1}^n - 2u_j^n + u_{j+1}^n \right)
\end{aligned} \tag{3.52}$$

という差分方程式を考えることができる。$\theta = 0$ がオイラー陽解法、$\theta = 1/2$ がクランク-ニコルソ

Algorithm 2 クランク-ニコルソン法

Input: 初期条件 $\left\{u_j^0\right\}_{j=1}^N$, 境界条件

Output: 時刻 $n\Delta t$ $(n = 1, \cdots, T)$ における $\left\{u_j^n\right\}_{j=1}^N$

1: LU 分解：境界条件に合わせて $\{\alpha_j\}$, $\{\beta_j\}$ を (3.46) に従って計算
2: **for** $n = 0$ to $T-1$ **do**
3: $\left\{u_j^n\right\}_{j=1}^N$ から $\boldsymbol{z}$ を (3.40) または (3.41) に従って計算
4: (3.49) に従って $\boldsymbol{y}$ を計算
5: (3.51) に従って $\boldsymbol{x}$ を計算し、これを $\left\{u_j^{n+1}\right\}_{j=1}^N$ とする
6: $\left\{u_j^{n+1}\right\}_{j=1}^N$ を出力
7: **end for**

ン法に対応する。また、$\theta = 1$ を完全陰的スキームと呼ぶ。(3.52) を書き直すと

$$\begin{aligned} & -\theta c u_{j-1}^{n+1} + (1+2\theta c) u_j^{n+1} - \theta c u_{j+1}^{n+1} \\ = \; & \{1 - 2(1-\theta)c\} u_j^n \\ & + (1-\theta) c u_{j-1}^n + (1-\theta) c u_{j+1}^n \end{aligned} \tag{3.53}$$

となる。ただし、境界条件は

- Dirichlet 境界条件: $u_0^n = u_{\mathrm{L}}$, $u_{N+1}^n = u_{\mathrm{R}}$ より

$$\begin{aligned} & (1+2\theta c) u_1^{n+1} - \theta c u_2^{n+1} \\ = \; & \{1 - 2(1-\theta)c\} u_1^n + c u_{\mathrm{L}} \\ & + (1-\theta) c u_2^n \end{aligned} \tag{3.54}$$

$$\begin{aligned} & -\theta c u_{N-1}^{n+1} + (1+2\theta c) u_N^{n+1} \\ = \; & \{1 - 2(1-\theta)c\} u_N^n + c u_{\mathrm{R}} \\ & + (1-\theta) c u_{N-1}^n \end{aligned} \tag{3.55}$$

- Neumann 境界条件: $u_0^n = u_1^n - J_{\mathrm{L}}\Delta x$, $u_{N+1}^n = u_N^n + J_{\mathrm{R}}\Delta x$ より

$$\begin{aligned} & (1+\theta c) u_1^{n+1} - \theta c u_2^{n+1} \\ = \; & \{1 - (1-\theta)c\} u_1^n - c J_{\mathrm{L}} \Delta x \\ & + (1-\theta) c u_2^n \end{aligned} \tag{3.56}$$

$$\begin{aligned} & -\theta c u_{N-1}^{n+1} + (1+\theta c) u_N^{n+1} \\ = \; & \{1 - (1-\theta)c\} u_N^n + c J_{\mathrm{R}} \Delta x \\ & + (1-\theta) c u_{N-1}^n \end{aligned} \tag{3.57}$$

である。

これらの連立方程式を解く際にも LU 分解が用いられる。今の場合は (3.37) の行列 A は Dirichlet 境界条件では

$$A = \begin{pmatrix} 1+2\theta c & -\theta c & 0 & \cdots & 0 \\ -\theta c & 1+2\theta c & -\theta c & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -\theta c & 1+2\theta c & -\theta c \\ 0 & \cdots & 0 & -\theta c & 1+2\theta c \end{pmatrix} \tag{3.58}$$

Neumann 境界条件では

$$A = \begin{pmatrix} 1+\theta c & -\theta c & 0 & \cdots & 0 \\ -\theta c & 1+2\theta c & -\theta c & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -\theta c & 1+2\theta c & -\theta c \\ 0 & \cdots & 0 & -\theta c & 1+\theta c \end{pmatrix} \tag{3.59}$$

となる。また、$\boldsymbol{z}$ は Dirichlet 境界条件では

$$\begin{aligned} z_1 &= \{1-2(1-\theta)c\}\,u_1^n + cu_{\mathrm{L}} \\ &\quad +(1-\theta)cu_2^n \\ z_j &= \{1-2(1-\theta)c\}\,u_j^n + (1-\theta)cu_{j-1}^n \\ &\quad +(1-\theta)cu_{j+1}^n \\ &\qquad (2 \le j \le N-1) \\ z_N &= \{1-2(1-\theta)c\}\,u_N^n + cu_{\mathrm{R}} \\ &\quad +(1-\theta)cu_{N-1}^n \end{aligned} \tag{3.60}$$

Neumann 境界条件では

$$\begin{aligned} z_1 &= \{1-(1-\theta)c\}\,u_1^n - cJ_{\mathrm{L}}\Delta x \\ &\quad +(1-\theta)cu_2^n \\ z_j &= \{1-2(1-\theta)c\}\,u_j^n + (1-\theta)cu_{j-1}^n \\ &\quad +(1-\theta)cu_{j+1}^n \\ &\qquad (2 \le j \le N-1) \\ z_N &= \{1-(1-\theta)c\}\,u_N^n + cJ_{\mathrm{R}}\Delta x \\ &\quad +(1-\theta)cu_{N-1}^n \end{aligned} \tag{3.61}$$

である。

3.7.1 安定性解析

3.5.1 と同様に安定性解析を行う。(3.53) において波数 k の特解

$$u_j^n = \lambda_k^n e^{ikj} \tag{3.62}$$

を代入すると、

$$\lambda_k = \frac{1-4(1-\theta)c\sin^2\frac{k}{2}}{1+4\theta c\sin^2\frac{k}{2}} \tag{3.63}$$

と求まる。これより、無条件安定（全ての $c>0$, k に対して $|\lambda_k|<1$ が成り立つ）であるための条件は $1/2 \le \theta \le 1$ であることがわかる。

3.8 演習問題

以下の 3 つの問題の解答をレポートとして提出せよ。

問題 1 (拡散方程式の数値解)

拡散方程式 (3.5) ($L=10$) を初期条件

$$u_0(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(x-5)^2} \tag{3.64}$$

の下で数値的に解く。以下の 4 つの場合を考えよ。

1. オイラー陽解法 (3.17) で境界条件は Dirichlet 境界条件 $u_{\mathrm{L}}=u_{\mathrm{R}}=0$ を用いて数値的に解け。刻み幅は $\Delta t=0.01$, $\Delta x=0.5$ ($N=20$) とせよ。

2. オイラー陽解法 (3.17) で境界条件は Neumann 境界条件 $J_{\mathrm{L}}=J_{\mathrm{R}}=0$ を用いて数値的に解け。刻み幅は $\Delta t=0.01$, $\Delta x=0.5$ ($N=20$) とせよ。

3. クランク-ニコルソン法 (3.19) で境界条件は Dirichlet 境界条件 $u_{\mathrm{L}}=u_{\mathrm{R}}=0$ を用いて数値的に解け。刻み幅は $\Delta t=0.01$, $\Delta x=0.05$ ($N=200$) とせよ。

4. クランク-ニコルソン法 (3.19) で境界条件は Neumann 境界条件 $J_{\mathrm{L}}=J_{\mathrm{R}}=0$ を用いて数値的に解け。刻み幅は $\Delta t=0.01$, $\Delta x=0.05$ ($N=200$) とせよ。

いずれの場合も $t = 1,2,3,4,5$ ($n = 100,200,300,400,500$) における $u(x,t)$ の値を出力すること。また、得られた $u(x,t)$ の値をそれぞれ一つの図（横軸 x, 縦軸 u）に図示せよ。

*（注意）本演習では (3.11) のように u_j^n に区間 $[(j-1)\Delta x, j\Delta x]$ を割り当てるという離散化を行なっている。従って、初期条件は

$$u_j^0 = \frac{1}{2}\left[u_0((j-1)\Delta x) + u_0(j\Delta x)\right] \tag{3.65}$$

であると考えるのが自然である。また、$\left\{u_j^n\right\}_{j=1}^N$ をプロットする場合は、各 j に対応する座標として $x=(j-1/2)\,\Delta x$ を採用するのが自然である。ただし、これらと異なる方法をとったとしても減点はしない。問題 2, 3 も同様である。

*（注意）本問題では方程式と初期条件の対称性より $x=5$ について対称な結果が得られるはず

である。もし解答のグラフが $x = 5$ について対称となっていない場合はプログラムが間違っていると考えてほしい。

*（注意）グラフを生成する部分についてはソースコードに含める必要はない。問題 2, 3 も同様である。

問題 2 (Fisher 方程式)

拡散方程式に非線形項 $f(u) = u(1-u)$ を加えた偏微分方程式 (Fisher 方程式)

$$\frac{\partial u}{\partial t} = u(1-u) + \frac{\partial^2 u}{\partial x^2} \tag{3.66}$$

を考える。この方程式は遺伝学における優性遺伝子の空間伝播を表現するために提案された。初期条件

$$u_0(x) = \frac{1}{(1+e^{bx-5})^2} \tag{3.67}$$

境界条件 ($L = 200$)

$$u(0,t) = 1 \tag{3.68}$$
$$u(L,t) = 0 \tag{3.69}$$

の下で $b = 0.25,\ 0.5,\ 1.0$ の場合に Fisher 方程式をオイラー陽解法

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = f\left(u_j^n\right) + \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{\Delta x^2} \tag{3.70}$$

を用いて数値的に解け。ただし、刻み幅は $\Delta x = 0.05$ ($N = 4000$), $\Delta t = 0.001$ とし、$t = 10, 20, 30, 40$ ($n = 10000, 20000, 30000, 40000$) の $u(x,t)$ の値を出力すること。また、得られた $u(x,t)$ の値を b の値ごとに一枚の図（横軸 x, 縦軸 u）に図示せよ。

また、得られた結果について考察せよ。（例えば b の値を様々に変えた時に振る舞いがどのように変わるかを見るなど。）

問題 3 (1 次元調和振動子の Schrödinger 方程式)

量子力学において、1 次元調和振動子のダイナミクスは、次の Schrödinger 方程式

$$i\hbar\frac{\partial \psi}{\partial t}(x,t) = \left(-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + \frac{k}{2}x^2\right)\psi(x,t) \tag{3.71}$$

に従う。$\psi(x,t) = \psi_{\mathrm{R}}(x,t) + i\psi_{\mathrm{I}}(x,t)$ ($\psi_{\mathrm{R}}, \psi_{\mathrm{I}} \in \mathbb{R}$) のように波動関数を実部と虚部に分けると、それぞれの従う方程式は

$$\hbar\frac{\partial \psi_{\mathrm{R}}}{\partial t}(x,t) = \left(-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + \frac{k}{2}x^2\right)\psi_{\mathrm{I}}(x,t) \tag{3.72}$$
$$-\hbar\frac{\partial \psi_{\mathrm{I}}}{\partial t}(x,t) = \left(-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + \frac{k}{2}x^2\right)\psi_{\mathrm{R}}(x,t) \tag{3.73}$$

となり、2 つの拡散型方程式の組で表される。この際、粒子の位置の確率密度は $P(x,t) \equiv |\psi(x,t)|^2 = \psi_{\mathrm{R}}(x,t)^2 + \psi_{\mathrm{I}}(x,t)^2$ で与えられる。

今、パラメータを $\hbar = 1$, $m = 1$, $k = 1$ とし、初期条件

$$\psi(x,0) = \frac{\sqrt{2}}{\pi^{\frac{1}{4}}}e^{-2(x-5)^2} \tag{3.74}$$

の下で Schrödinger 方程式を数値的に解くことを考える。ただし、空間領域は $[-L/2, L/2]$ とし、境界条件は周期境界条件

$$\psi\left(\frac{L}{2}, t\right) = \psi\left(-\frac{L}{2}, t\right) \tag{3.75}$$

とする。

数値的に解く際には Visscher のスキーム [3] と呼ばれる、実部と虚部の時間発展を半ステップだけずらして解く陽解法を用いる。即ち、離散化された波動関数の実部、虚部をそれぞれ $\left\{R_j^n\right\}$, $\left\{I_j^n\right\}$ と書くと、$1 \le j \le N$ ($L = N\Delta x$) に対し

て時間発展は

$$\begin{aligned} & R_j^{n+1} \\ = \; & R_j^n \\ & +\Delta t \left(-\frac{1}{2} \frac{I_{j-1}^n - 2I_j^n + I_{j+1}^n}{\Delta x^2} + \frac{1}{2} x_j^2 I_j^n \right) \end{aligned} \tag{3.76}$$

$$\begin{aligned} & I_j^{n+1} \\ = \; & I_j^n \\ & -\Delta t \left(-\frac{1}{2} \frac{R_{j-1}^{n+1} - 2R_j^{n+1} + R_{j+1}^{n+1}}{\Delta x^2} \right. \\ & \left. +\frac{1}{2} x_j^2 R_j^{n+1} \right) \end{aligned} \tag{3.77}$$

と記述される。ただし、

$$x_j \equiv \left(j - \frac{1}{2} \right) \Delta x - \frac{L}{2} \tag{3.78}$$

であり、境界条件は全ての n で

$$R_0^n = R_N^n \tag{3.79}$$

$$R_{N+1}^n = R_1^n \tag{3.80}$$

$$I_0^n = I_N^n \tag{3.81}$$

$$I_{N+1}^n = I_1^n \tag{3.82}$$

と表される。また、初期条件は $1 \leq j \leq N$ で

$$\begin{aligned} R_j^0 = \; & \frac{1}{2} \left[\psi \left((j-1)\Delta x - \frac{L}{2}, 0 \right) \right. \\ & \left. +\psi \left(j\Delta x - \frac{L}{2}, 0 \right) \right] \end{aligned} \tag{3.83}$$

$$\begin{aligned} I_j^0 = \; & -\Delta t \left(-\frac{1}{2} \frac{R_{j-1}^0 - 2R_j^0 + R_{j+1}^0}{\Delta x^2} \right. \\ & \left. +\frac{1}{2} x_j^2 R_j^0 \right) \end{aligned} \tag{3.84}$$

と表される。$N = 400$, $\Delta x = 0.05$ ($L = 20$), $\Delta t = 0.001$ の下でこの系を数値的に解き、$t = 1, 2, 3, 4, 5, 6, 7, 8$ ($n = 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000$) における確率密度 $\left(R_j^n \right)^2 + I_j^n I_j^{n-1}$ ($1 \leq j \leq N$) の値を出力せよ。また、得られた確率密度 $\left(R_j^n \right)^2 + I_j^n I_j^{n-1}$ の値を一つの図（横軸は x_j）に図示せよ。

参考文献

[1] 山本哲朗, 数値解析入門 (サイエンスライブラリ 現代数学への入門 14), サイエンス社 (1976)

[2] 高橋大輔, 数値計算 (理工系の基礎数学 8), 岩波書店 (1996)

[3] P. B. Visscher, Computers in Physics **5**, 596 (1991)

[上田仁彦]

第4章　関数の補間と数値積分

4.1　目的

4 章では積分を何らかの近似により数値的に計算する手法である**数値積分**について学習する．積分は工学や物理学上の様々なモデルに現れるが，その不定積分が陽に得られるのは被積分関数が特別な形をしているときのみであり，多くの場合積分値を評価するためには何らかの近似を導入して数値的に計算する必要がある．広く用いられている数値積分法は積分領域の次元によって大きく異なるが，本章では積分領域が比較的低次元の場合（高々3,4 次元程度）に有効な，関数補間に基づく数値積分公式について述べる．なお本章では簡単のため一次元上の区間 (a,b) における積分

$$\int_a^b f(x)dx$$

の数値積分を取り扱う（2 次元以上の積分については 4.4.4 参照）．

一般に未知関数 $f(x)$ について，いくつかの点 x_i において関数値 $f(x_i)$ が与えられたときに，これらの点 x_i の間の部分における $f(x)$ の値を推定することを**補間**という．関数補間に基づく数値積分法は，被積分関数を積分計算が容易な既知関数で補間することで積分値を近似的に求める手法である．4.2 節で扱う Newton-Cotes 公式は，最も基本的な数値積分公式の一つであり，被積分関数を Lagrange 補間（多項式補間）することで積分値を計算する方法である．この積分公式では積分区間を等間隔に分割し，その分点上における被積分関数の値を使って積分値を求める．4.3 節で扱う Gauss 型積分公式は直交多項式による補間に基づく数値積分公式であり，積分区間の分割を工夫することで Newton-Cotes 積分公式よりも精度を高めた数値積分法であると捉えられる．4.4 節ではこれらの数値積分公式の代表的な応用例として，偏微分方程式の数値解法である有限要素法の解説を行う．

4.2　Lagrange 補間と Newton-Cotes 公式

本節では最も基本的な数値積分公式の一つである Newton-Cotes 公式について解説する．4.2.1 節でこの積分公式の基礎となる Lagrange 補間を解説し，4.2.2 節で Newton-Cotes 公式について解説を行う．

4.2.1　Lagrange 補間

相異なる n 個の点 $x_1 < \cdots < x_n$ において関数 f の値 $f(x_i), (i = 1, \ldots, n)$ が与えられたとき，

$$P(x_i) = f(x_i) \quad (i = 1, \cdots, n) \tag{4.1}$$

を満たす $n-1$ 次多項式 $P(x)$ を作り，$x \in [x_1, x_n]$ における $f(x)$ の値を $P(x)$ と推定することを**多項式補間**または **Lagrange 補間**と呼ぶ．このような $n-1$ 次の補間多項式 $P(x)$ は一意に存在する．実際，$P(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}$ とおくと，与えられた条件は

$$a_0 + a_1 x_i + \cdots + a_{n-1}x_i^{n-1} = f(x_i) \quad (i = 1, \cdots, n)$$

となり，この式をまとめて書くと

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ & \cdots & & \cdots & \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \cdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \cdots \\ f(x_n) \end{pmatrix} \tag{4.2}$$

となる．係数行列の行列式は **Vandermonde の行列式**と呼ばれ

$$\begin{vmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ & \cdots & & \cdots & \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{vmatrix} = \prod_{i>j}(x_i - x_j) \neq 0$$

となるため，線形方程式 (4.2) の解は一意に存在する．

式 (4.1) を満たす $n-1$ 次の補間多項式 $P(x)$ は

$$P(x) = \sum_{i=1}^{n} f(x_i) l_i(x) \tag{4.3}$$

$$l_i(x) = \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)} \tag{4.4}$$

で与えられる．実際，$l_i(x_j) = \delta_{ij}$ より $P(x_i) = f(x_i)$ となる．また $P(x)$ は $n-1$ 次多項式 $l_i(x)$ の線型結合であるため，高々 $n-1$ 次の多項式である．これらの事実と上で示した補間多項式の一意性より，(4.3) は $f(x)$ の $n-1$ 次補間多項式である．

4.2.2 Newton-Cotes 公式

定積分

$$I(f) = \int_{\widetilde{a}}^{\widetilde{b}} f(x)dx$$

を近似的に計算するために，4.2.1 節で述べた Lagrange 補間で関数 f を近似する方法を考える．$\widetilde{x}_i$ を n 個の等間隔分点

$$\widetilde{x}_i = \widetilde{x}_1 + (i-1)h \quad (1 \leq i \leq n,\, h > 0)$$

とし，この分点における関数 $f(x)$ の Lagrange 補間 (4.3) を用いると，

$$\begin{aligned} I(f) &= \int_{\widetilde{a}}^{\widetilde{b}} f(x)dx \\ &\approx \int_{\widetilde{a}}^{\widetilde{b}} P(x)dx \\ &= \int_{\widetilde{a}}^{\widetilde{b}} \sum_{i=1}^{n} f(\widetilde{x}_i) l_i(x) \\ &= \sum_{i=0}^{n} c_i f(\widetilde{x}_i) \end{aligned}$$

となる．ここに

$$c_i = \int_{\widetilde{a}}^{\widetilde{b}} l_i(x)dx \tag{4.5}$$

である．この数値積分公式を n **点 Newton-Cotes 公式**という．

式 (4.4) より関数 l_i は分点 x_i から決まるので，等間隔分点 x_i の個数およびそのとり方を決めることで様々な積分公式を作ることができるが，実際によく使われるのは以下の 3 種類の公式である．

単一区間における中点公式

区間 $(\widetilde{a}, \widetilde{b})$ において，

$$\widetilde{x}_1 = \frac{\widetilde{a} + \widetilde{b}}{2}$$

の 1 点 Newton-Cotes 公式を考える．$l_1(x) \equiv 1$ に注意して，式 (4.5) より

$$c_1 = \int_{\widetilde{a}}^{\widetilde{b}} dx = \widetilde{b} - \widetilde{a}$$

となる．したがって以下の数値積分公式を得る．

$$\int_{\widetilde{a}}^{\widetilde{b}} f(x)dx \approx (\widetilde{b} - \widetilde{a}) f(\widetilde{x}_1). \tag{4.6}$$

これを**中点公式**または中点則という（図 4.1）．

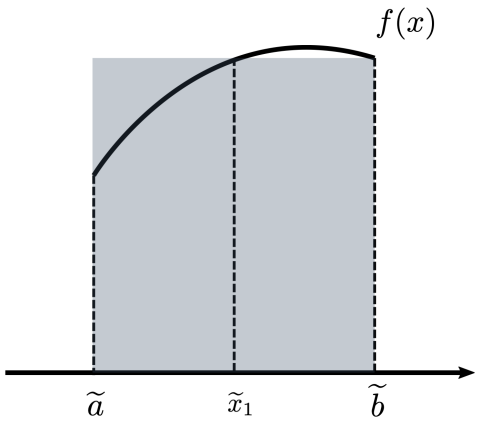


図 4.1: 積分区間 $(\widetilde{a}, \widetilde{b})$ における中点公式

単一区間における台形公式

区間 $(\widetilde{a}, \widetilde{b})$ において

$$\widetilde{x}_1 = \widetilde{a},\ \widetilde{x}_2 = \widetilde{b}$$

の 2 点 Newton-Cotes 公式を考える．分点の間隔は $\widetilde{h} = \widetilde{x}_2 - \widetilde{x}_1 = b - a$ であり，また式 (4.5) より，$c_1 = c_2 = h/2$．ゆえに以下の数値積分公式を得る．

$$\int_{\widetilde{a}}^{\widetilde{b}} f(x)dx \approx \frac{\widetilde{h}}{2}\{f(\widetilde{x}_1) + f(\widetilde{x}_2)\}. \qquad (4.7)$$

これを**台形公式**または台形則という（図 4.2）．

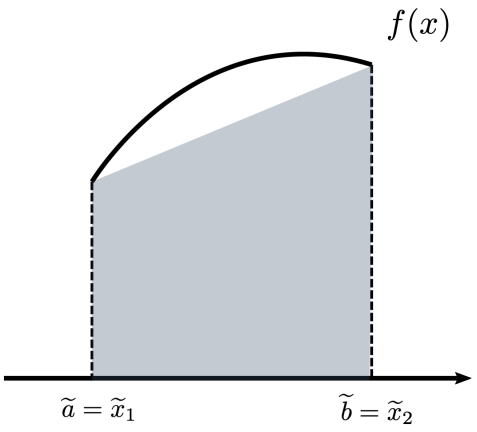


図 4.2: 積分区間 $(\widetilde{a}, \widetilde{b})$ における台形公式

単一区間における Simpson の公式

区間 $(\widetilde{a}, \widetilde{b})$ において

$$\widetilde{x}_i = \widetilde{a} + (i-1)\widetilde{h}, \quad (i = 1, 2, 3)$$
$$\widetilde{h} = \frac{\widetilde{b} - \widetilde{a}}{2}$$

の 3 点 Newton-Cotes 公式を考えると，式 (4.5) より

$$c_1 = \frac{\widetilde{h}}{3},\ c_2 = \frac{4}{3}\widetilde{h},\ c_3 = \frac{\widetilde{h}}{3}.$$

ゆえに以下の数値積分公式を得る．

$$\int_{\widetilde{a}}^{\widetilde{b}} f(x)dx \approx \frac{\widetilde{h}}{3}\{f(\widetilde{x}_1) + 4f(\widetilde{x}_2) + f(\widetilde{x}_3)\}. \qquad (4.8)$$

これは **Simpson の公式**または Simpson 則と呼ばれる（図 4.3）．

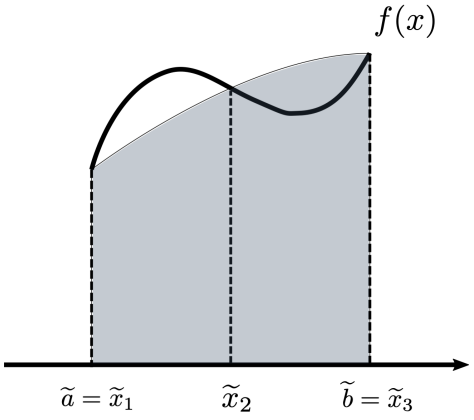


図 4.3: 積分区間 $(\widetilde{a}, \widetilde{b})$ における Simpson 公式

4.2.3 複合公式

前節において一般の n に対する n 点 Newton-Cotes 公式を求めたが，実際によく使われるのは中点公式，台形公式，Simpson の公式，すなわち $n = 1, 2, 3$ と小さい値の n に対する Newton-Cotes 公式である．「より大きな n に対する Newton-Cotes 公式を考えることで，より高精度な数値積分公式が得られないか」と考えるかもしれないが，これは大体の場合誤りである [1]（この資料の 4.2.4 節や 4.2.5 節の課題 1 も参照)．一方上で示した 3 つの公式は，被積分関数を定数，一次関数，二次関数で近似して得られる数値積分公式であるため，複雑な形状の被積分関数 $f(x)$ に対して，積分区間全体に上の公式をそのまま適用してもまともな精度は得られない．そこで実際の計算では積分区間 $[a, b]$ を等分して小区間を作り，各区間に対して上の数値積分公式を適用する．このようにして得られる積分公式を複合公式と言い，上の 3 つの積分公式の複合公式もそれぞれ複合中点公式，複合台形公式，複合 Simpson 公式と呼ぶ（但し，実際の計算ではほぼ必ず複合公式を用いるため，複合中点公式のことを単に中点公式などと呼ぶことが多い．）．

以下に 3 つの複合公式の陽な形を示す．

（複合）中点公式

$h=(b-a)/n$, $x_i=a+ih$, $(0\le i\le n)$ とおき，積分区間 (a,b) を n 個の小区間 (x_i,x_{i+1}) $(0\le i\le n-1)$ に分割し，各々の小区間に対して中点公式を適用すると，

$$\int_a^b f(x)dx \approx h\sum_{i=0}^{n-1} f\left(\frac{x_i+x_{i+1}}{2}\right) \qquad (4.9)$$

を得る．

（複合）台形公式

前例と同じ刻み幅 h と分点 x_i をとり，n 個の小区間 (x_i,x_{i+1}) に台形公式 (4.7) を適用すると，（複合）台形公式

$$\int_a^b f(x)dx \approx \frac{h}{2}\left(f(x_0)+2\sum_{i=1}^{n-1} f(x_i)+f(x_n)\right)$$

を得る．

（複合）Simpson 公式

前例と同じ刻み幅 h と分点 x_i をとり，各小区間にシンプソン公式 (4.8) を適用すると，（複合）シンプソン公式

$$\int_a^b f(x)dx \approx \frac{h}{6}\left(f(x_0)+2\sum_{i=1}^{n-1} f(x_i)+f(x_n) + 4\sum_{i=0}^{n-1} f\left(\frac{x_i+x_{i+1}}{2}\right)\right)$$

を得る．

4.2.4 Newton-Cotes 公式の誤差

区間 $(\widetilde{a},\widetilde{b})$ における単一区間の中点公式 (4.6) の左辺において $f(x)$ を $x=\widetilde{x}_1:=(a+b)/2$ 周りで Taylor 展開すると，

$$\begin{aligned}\int_{\widetilde{a}}^{\widetilde{b}} f(x)dx \approx& \int_{\widetilde{a}}^{\widetilde{b}} (f(\widetilde{x}_1)+f'(\widetilde{x}_1)(x-\widetilde{x}_1) \\ &+\frac{1}{2}f''(\widetilde{x}_1)(x-\widetilde{x}_1)^2)dx \\ =&(\widetilde{b}-\widetilde{a})f(\widetilde{x}_1)+\frac{(\widetilde{b}-\widetilde{a})^3}{24}f''(\widetilde{x}_1)\end{aligned}$$

となる．したがって積分公式 (4.6) の誤差は

$$\int_a^b f(x)dx-(b-a)f(x_1)\approx\frac{(b-a)^3}{24}f''(x_1)$$

となる．これを用いると n 点複合中点公式 (4.9) の誤差は

$$\begin{aligned}&\int_a^b f(x)dx-h\sum_{i=0}^{n-1} f\left(\frac{x_i+x_{i+1}}{2}\right)\\ =&\sum_{i=0}^{n-1}\left\{\int_{x_i}^{x_{i+1}} f(x)dx-hf\left(\frac{x_i+x_{i+1}}{2}\right)\right\}\\ =&\sum_{i=0}^{n-1}\frac{(x_{i+1}-x_i)^3}{24}f''\left(\frac{x_i+x_{i+1}}{2}\right)\\ =&\frac{n}{24}f''(\eta)h^3=\frac{(b-a)}{24}f''(\eta)h^2\end{aligned}$$

となる．ただし η は次式を満たすように選んだ．

$$\sum_{i=0}^{n-1} f''\left(\frac{x_i+x_{i+1}}{2}\right)=nf''(\eta).$$

これより，固定した積分区間 (a,b) における積分に対する（複合）中点公式の誤差は，分割幅 h に対して h^2 程度のオーダーであることがわかる．

（複合）台形公式,（複合）Simpson 公式についても同様の評価を得ることができ，それぞれ

$$-\frac{b-a}{12}f''(\eta)h^2,\quad -\frac{b-a}{2880}f^{(4)}(\eta)h^4$$

となり，それぞれ h^2, h^4 程度のオーダーである [1].

4.2.5 4.2 節の課題

1. $n=2,4,8,16$ に対して，以下の関数 $f(x)$ の $n-1$ 次多項式による Lagrange 補間のグラフを図示せよ．但し関数が定義される区間を (a,b) としたとき，分点 $x_i,(i=1,\cdots,n)$ は

$$x_i=a+\frac{b-a}{n-1}(i-1)$$

とする．また得られたグラフからわかることを述べよ．(Hint: Lagrange 補間で得られる関数 $P(x)$ は多項式なので，gnuplot などで

多項式をプロットさせるコードを書けばよいが，もしそれが難しければ，x_i よりもさらに細かい分点 y_i を取り，$P(y_i)$ 計算するコードを書いてもよい．)

(a) $f(x) = \log x \quad x \in (1, 2)$

(b) $f(x) = \frac{1}{x^3} \quad x \in (0.1, 2)$

(c) $f(x) = \frac{1}{1+25x^2} \quad x \in (-1, 1)$

2. 区間 (a, b) における定積分

$$I(f) := \int_a^b f(x)dx$$

を分割数 $n = 2^i\ (i = 0, 1, \cdots, 10)$ の（複合）中点公式，台形公式，Simpson 公式により求めるコードを作成し，以下の関数 $f(x)$ に対して適用した結果をグラフか表にまとめよ．(Hint: 結果の値によってグラフの方が見やすい場合と表の方が見やすい場合があるので適宜使い分けるとよい．またいくつかのグラフを重ねて表示するなど工夫して結果をまとめること)

(a) $f(x) = \frac{1}{x} \quad x \in (1, 2)$

(b) $f(x) = \mathrm{e}^{5x} \quad x \in (-1, 1)$

(c) $f(x) = 1 + \sin x \quad x \in (0, \pi)$

(d) $f(x) = 1 + \sin x \quad x \in (0, 2\pi)$

3. 課題 2. の定積分 $I(f)$ を分割数 n の数値積分で近似的に求めた値を $I_n(f)$ とし，その誤差を

$$E_n(f) = I_n(f) - I(f)$$

とする．課題 2 のときと同様に，$n = 2^i\ (i = 0, 1, \cdots, 10)$ に対する（複合）中点公式，台形公式，Simpson 公式を使って，(a)〜(d) に対する積分の誤差をグラフか表にまとめよ．

4. 課題 2. と課題 3. で作成したグラフや表からわかることを述べよ．

4.3 Gauss 型積分公式

前節で扱った Newton-Cotes 公式は等間隔分点上での被積分関数の値を使った積分公式であったが，本節で扱う Gauss 型積分公式は分点を適切に取ることで積分公式の精度をある意味で最良にした方法であると解釈できる．このアイデアに基づくと結果的に直交多項式の零点を分点とすればよいことがわかる．本節では 4.3.1 節で直交多項式について簡単に解説を行い，4.3.2 節で Gauss 型積分公式について解説する．

4.3.1 直交多項式

$w(x) > 0$ を区間 (a, b) で定義された連続な関数とし，実係数多項式 $p(x), q(x)$ に対して

$$(p, q)_w = \int_a^b w(x)p(x)q(x)dx \tag{4.10}$$

とおく．このとき，$(\cdot, \cdot)_w$ は内積を与える．すなわち次の性質を満たすことがわかる．

正値性 $(p, p)_w \geq 0$ で，$(p, p)_w = 0 \Rightarrow p \equiv 0$

対称性 $(p, q)_w = (q, p)_w$

線形性 多項式 $p(x), q(x), r(x)$ と実数 α, β に対して $(\alpha p + \beta q, r)_w = \alpha(p, r)_w + \beta(q, r)_w$

とくに $(p, q)_w = 0$ のとき，多項式 p, q は重み関数 w に関して**直交する**という．さらに $\phi_n(x),\ (n = 0, 1, 2, \cdots)$ が n 次多項式 $(\phi_0(x) \neq 0)$ で $(\phi_m, \phi_n)_w = 0,\ (m \neq n)$ を満たすとき，$\{\phi_n\}$ を直交多項式系という．

直交多項式系は以下の性質を満たす．

直交多項式系の存在 任意の内積に対して直交多項式系は存在する．

直交性 $\phi_n(x)$ は $n-1$ 次以下の任意の多項式と直交する．

一意性 二つの直交多項式系 $\{\phi_n(x)\}, \{\widetilde{\phi}_n(x)\}$ に対して，ある定数 $c_n \neq 0$ が存在して，$\phi_n(x) = c_n\widetilde{\phi}_n(x)$．すなわち直交多項式系は定数倍を除いて一意に定まる．

[直交多項式系の存在の証明] $\{1, x, x^2, \cdots\}$ に対して Gramm-Schmidt の直交化法を行えばよい．

[直交性の証明] $n-1$ 次以下の多項式 $p(x)$ は，$\phi_m(x)$, $(0 \le m \le n-1)$ の線型結合で表せる．すなわち $p(x) = \sum_{m=0}^{n-1} c_m \phi_m(x)$．したがって

$$\begin{aligned}(\phi_n, p)_w &= (\phi_n, \sum_{m=0}^{n-1} c_m \phi_m)_w \\ &= \sum_{m=0}^{n-1} c_m (\phi_n, \phi_m) \\ &= 0\end{aligned}$$

[一意性の証明] $\phi_n(x)$ と $\widetilde{\phi}_n(x)$ の x^n の係数をそれぞれ $c_n, \widetilde{c}_n (\neq 0)$ とすると，$q(x) = \widetilde{c}_n \phi_n(x) - c_n \widetilde{\phi}_n(x)$ は $n-1$ 次以下の多項式であるため，上で示した直交性より

$$\begin{aligned}(q, q)_w &= (q, \widetilde{c}_n \phi_n - c_n \widetilde{\phi}_n)_w \\ &= \widetilde{c}_n (q, \phi_n)_w - c_n (q, \widetilde{\phi}_n)_w \\ &= 0\end{aligned}$$

したがって $q \equiv 0$．$c_n/\widetilde{c}_n$ をあらためて c_n とおくことで $\phi_n(x) = c_n \widetilde{\phi}_n(x)$ を得る．

以上より内積（すなわち重み関数 w と区間 (a, b)）を一つ定めると直交多項式系が一つ定まることがわかる．よく用いられる直交多項式として，内積 (4.10) を

$$(p, q) = \int_{-1}^{1} p(x) q(x) dx$$

（すなわち，重み関数 $w \equiv 1$，区間 $(a, b) = (-1, 1)$）とすると得られる直交多項式系 $P_n(x)$ は **Legendre 多項式**と呼ばれる．他にも重み関数 $w = (1 - x^2)^{-1/2}$，区間 $(a, b) = (-1, 1)$ の **Chebyshev 多項式**，重み関数 $w = \mathrm{e}^{-x}$，区間 $(a, b) = (0, \infty)$ の **Laguerre 多項式**，重み関数 $w = \mathrm{e}^{-x^2}$，区間 $(a, b) = (-\infty, \infty)$ の **Hermite 多項式**などがよく知られている．

4.3.2 Gauss 型積分公式

Newton-Cotes 積分公式は，n 個の等間隔分点 x_i が与えられたときに，n 個の重み w_i を適切に設定することで，少なくとも $n-1$ 次までの多項式に対して誤差なく積分を行える積分公式と捉えることができる（→4.2.4 節）．そうすると，「重み w_i だけでなく分点 x_i も適切に決めれば，より高次の多項式に対しても誤差の無い積分公式が作れないか」と考えられる．これを実際に実現するのが **Gauss 型積分公式**である．ここではこのアイデアに基づき，Gauss 型積分公式を構成的に得る方法を示す．

n 個の重み w_i と分点 x_i, $(i = 1, \cdots, n)$ に対して積分公式

$$\int_{-1}^{1} f(x) dx = \sum_{i=1}^{n} w_i f(x_i) \tag{4.11}$$

が m 次までの多項式に対して正確であるとする（簡単のため積分範囲は $(-1, 1)$ とする）．このとき分点 x_i を零点とする n 次多項式を $G(x) = \prod_{i=1}^{n} (x - x_i)$，$r$ 次の Legendre 多項式を P_r とすると，$0 \le r \le m - n$ に対して

$$\int_{-1}^{1} G(x) P_r(x) dx = \sum_{i=1}^{n} w_i G(x_i) P_r(x_i) = 0 \tag{4.12}$$

が成り立つ（$\because G(x)P_r(x)$ は高々m 次多項式であるから）．つまり $G(x)$ と $P_r(x)$ $(0 \le r \le m-n)$ は直交する．これと Legendre 多項式の直交性より $G(x)$ は $m - n + 1$ 次以上の多項式でなければならない（$\because$ もし $G(x)$ が $m-n$ 次以下の多項式ならば，$G(x)$ は $P_r(x)$ $(0 \le r \le m-n)$ の線型結合で表されるため，いずれかの $P_r(x)$ と直交しないから）．すなわち

$$\begin{aligned}&n \ge m - n + 1 \\ &\Rightarrow m \le 2n - 1\end{aligned}$$

積分公式 (4.11) は m 次までの多項式に対して正確であるとしたので，m を最大化するために $m = 2n-1$ とする．このとき n 次多項式 $G(x)$ に対して式 (4.12) は，$0 \le r \le n-1$ で成り立たなければならない．すなわち $G(x)$ は n 次の Legendre 多項式（の定数倍）である必要がある．さらにこの事実と式 (4.12) より x_i は n 次 Legendre 多項式の零点となることがわかる．

数値積分公式の分点 x_i は n 次 Legendre 多項式の零点とすればよいことがわかったので，次に $m = 2n-1$ を実現する重み w_i を求める．$f(x)$ を $2n-1$ 次以下の多項式とするとき，$f(x)$ を $P_n(x)$ で割った商を $Q(x)$，余りを $R(x)$ とする：

$$f(x) = P_n(x)Q(x) + R(x).$$

ただし $Q(x), R(x)$ は $n-1$ 次以下の多項式である．$P_n(x)$ は $n-1$ 次以下の任意の多項式と直交するので，

$$\begin{aligned}\int_{-1}^{1} f(x)dx &= \int_{-1}^{1} (P_n(x)Q(x) + R(x))dx \\ &= \int_{-1}^{1} R(x)dx\end{aligned}$$

となる．一方，分点 x_i は Legendre 多項式の零点であったから

$$\begin{aligned}\sum_{i=1}^{n} w_i f(x_i) &= \sum_{i=1}^{n} w_i (P_n(x_i)Q(x_i) + R(x_i)) \\ &= \sum_{i=1}^{n} w_i R(x_i)\end{aligned}$$

となる．以上より，数値積分公式が $n-1$ 次以下の多項式 $R(x)$ に対して正確な値を与えるように重み w_i を定めれば，$2n-1$ 次以下の多項式 $f(x)$ に対しても正確な値を与えることがわかる．$n-1$ 次多項式 $R(x)$ の Lagrange 補間は $R(x)$ に等しいから，$R(x) = \sum_{i=1}^{n} R(x_i)l_i(x)$ である．これを積分すると

$$\int_{-1}^{1} R(x)dx = \sum_{i=1}^{n} R(x_i) \int_{-1}^{1} l_i(x)dx$$

となるので，重みは

$$w_i = \int_{-1}^{1} l_i(x)dx \tag{4.13}$$

で与えられる．n 次の Legendre 多項式の零点 x_i と重み (4.13) による積分公式 (4.11) を n 次の **Gauss 型積分公式**または **Gauss-Legendre 積分公式**と呼ぶ．

4.3.3 複合公式

Gauss 型積分公式も Newton-Cotes 公式と同様の理由（→ 4.2.3 節）で積分区間を小区間に分割し，それぞれの小区間に低次の積分公式を用いる複合公式を使うのが一般的である．一般に区間 (a, b) における積分を分点 $x_i = a + i(b-a)/n\ (i = 0, \cdots, N)$ で N 等分すると

$$\int_a^b f(x)dx = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)dx$$

区間 (x_i, x_{i+1}) の積分は変数変換

$$y = 2\frac{x - x_i}{x_{i+1} - x_i} - 1$$

で $(-1, 1)$ での積分に変換できるので，4.3.2 節で求めた n 次 Gauss 型積分公式の積分点と重みをそれぞれ $y_m, w_m\ (m = 1, \cdots n)$ とすると，

$$\begin{aligned}&\sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)dx \\ &= \sum_{i=0}^{N-1} \int_{-1}^{1} f\left\{\frac{(y+1)(x_{i+1} - x_i)}{2} + x_i\right\} \frac{x_{i+1} - x_i}{2} dy \\ &\approx \sum_{i=0}^{N-1} \sum_{m=1}^{n} w_m f\left\{\frac{(y_m+1)(x_{i+1} - x_i)}{2} + x_i\right\} \frac{x_{i+1} - x_i}{2}\end{aligned}$$

となり，Gauss 型積分公式に対する複合公式を得る．

4.3.4 Gauss 型積分公式の誤差

Gauss 型積分公式の誤差について以下の事実が知られている．区間 $(-1, 1)$ における積分の Gauss 型積分公式 (4.11) の誤差はある $\xi \in (a, b)$ に対して

$$\int_{-1}^{1} f(x)dx - \sum_{m=1}^{n} w_m f(y_m) \approx \frac{f^{(2n)}(\xi)}{(2n)!} \int_{-1}^{1} \phi_n(x)^2 dx$$

$$\phi_n(x) = (x - y_1)^2 \cdots (x - y_m)^2$$

となる．これを用いると（複合）Gauss 型積分公式の誤差は

$$\begin{aligned}
&\int_a^b f(x)dx - \sum_{i=0}^{N-1}\sum_{m=1}^{n} w_m f(x(y_m))\frac{x_{i+1}-x_i}{2} \\
=&\sum_{i=0}^{N-1}\int_{x_i}^{x_{i+1}} f(x)dx - \sum_{i=0}^{N-1}\sum_{m=1}^{n} w_m f(x(y_m))\frac{x_{i+1}-x_i}{2} \\
=&\sum_{i=0}^{N-1}\left\{\int_{-1}^{1} f(x(y))dy - \sum_{m=1}^{n} w_m f(x(y_m))\right\}\frac{x_{i+1}-x_i}{2} \\
\approx&\sum_{i=0}^{N-1}\frac{f^{(2n)}(\xi_i)}{(2n)!}\int_{-1}^{1}\phi_n(x)^2dx\left(\frac{x_{i+1}-x_i}{2}\right)^{2n+1} \\
=&N\frac{f^{(2n)}(\eta)}{(2n)!}\int_{-1}^{1}\phi_n(x)^2dx\left(\frac{x_{i+1}-x_i}{2}\right)^{2n+1}
\end{aligned}$$

となる．したがって複合公式の刻み幅を $h=(b-a)/N=x_{i+1}-x_i$ とすると，この式は

$$\frac{(b-a)f^{(2n)}(\eta)}{(2n)!2^{2n+1}}\int_{-1}^{1}\phi_n(x)^2dx\,h^{2n}$$

となり，誤差は h^{2n} 程度となることがわかる（4.2.4 節の Newton-Cotes 公式の誤差と比較せよ）．

4.3.5 4.3 節の課題

1. 4.2 節の課題 2 と同じ積分を分割数 $N=2^i\,(i=0,1,\cdots,10)$ の（複合）M 次 Gauss 型積分公式で計算し結果をグラフか表にまとめよ．ただし積分区間を N 分割した各小区間では表 4.1, 4.2 に従い，$M=2$ または $M=3$ の Gauss 型積分公式を用いよ．

表 4.1: $M=2$ の Gauss 型積分公式

m	1	2
y_m	$-\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$
w_m	1	1

2. 以下の積分を考える．

$$\int_0^1 \frac{\mathrm{e}^{-x}}{\sqrt{x}}dx \tag{4.14}$$

表 4.2: $M=3$ の Gauss 型積分公式

m	1	2	3
y_m	$-\sqrt{\frac{3}{5}}$	0	$\sqrt{\frac{3}{5}}$
w_m	$\frac{5}{9}$	$\frac{8}{9}$	$\frac{5}{9}$

(a) 定積分 (4.14) に対して，積分区間 $(0,1)$ を $N=2^i\,(i=0,1,\cdots,10)$ 等分した小区間に $m=2,3$ の Gauss 型積分公式を適用し，積分値とその誤差 E_n を求めよ．ただし

$$\int_0^1 \frac{\mathrm{e}^{-x}}{\sqrt{x}}dx \approx 1.49364826562$$

である．

また Gauss 型積分公式の代わりに台形則や Simpson 則を適用するとどうなるか？

(b) 定積分 (4.14) を $\sqrt{x}=t$ で変数変換した積分を求めよ．得られた不定積分に対して，その積分区間を $N=2^i\,(i=0,1,\cdots,10)$ 等分し，それぞれの小区間に $m=2,3$ の Gauss 型積分公式を適用し，積分値とその誤差 E_n を求めよ．

(c) 定積分 (4.14) を

$$\int_0^1 \frac{\mathrm{e}^{-x}}{\sqrt{x}}dx = \int_0^1 \frac{1}{\sqrt{x}}dx + \int_0^1 \frac{\mathrm{e}^{-x}-1}{\sqrt{x}}dx$$

と変形し，第一項の積分を手計算で，第二項の積分を (a),(b) と同じ数値積分でそれぞれ求めた積分値とその誤差 E_n を求めよ．

(d) (a),(b),(c) の 3 つの方法の精度にはどのような違いがあるか，理由と共に述べよ．

4.4 有限要素法

4.4.1 概要

本節では数値積分の応用例として，有限要素法について述べる．**有限要素法 (Finite Element**

Method, FEM) は変分問題に基づく数値解法であり，差分法と並んで代表的な偏微分方程式の数値解法である．元々はエンジニアによって開発された数値解法であり，機械工学，土木工学，電磁気学などに現れる様々な現象を解析するために広く用いられている．通常，有限要素法は偏微分方程式の解析に用いられるが，偏微分方程式は（当然「偏」微分を含むために）2 次元以上の領域で定式化されるため，実装が複雑になりがちである．そこで本節では，1 次元の Strum-Liouville 型の常微分方程式を例にとり，これに対する有限要素法について解説する．またこの資料では Strum-Liouville 型微分方程式に有限要素法を適用した結果の式のみを示し，有限要素法の理論については深く立ち入らない．有限要素法の詳しい解説は [1, 5] などを参照されたい．

4.4.2 Strum-Liouville 型微分方程式と弱形式

Strum-Liouville 型の 2 点境界値問題,

$$-\frac{d}{dx}\left(p(x)\frac{du}{dx}(x)\right)+q(x)u(x)=f(x), \qquad a<x<b \quad (4.15)$$

$$u(a)=u(b)=0 \quad (4.16)$$

を考える．ただし，$p(x), q(x), f(x)$ は与えられた滑らかな関数とする．

$v(a)=v(b)=0$ を満たす関数 v を式 (4.15) の両辺にかけ，区間 (a,b) で積分すると，左辺第一項は

$$\begin{aligned}&\int_a^b v(x)\left\{-\frac{d}{dx}\left(p(x)\frac{du}{dx}\right)\right\}dx\\ &=-\left[v(x)\left(p(x)\frac{du}{dx}\right)\right]_a^b+\int_a^b \frac{dv}{dx}(x)p(x)\frac{du}{dx}(x)dx\\ &=\int_a^b p(x)\frac{dv}{dx}(x)\frac{du}{dx}(x)dx\end{aligned}$$

となり，以下の式を得る:

$$\begin{aligned}&a(u,v)\\ &:=\int_a^b\left(p(x)\frac{du}{dx}(x)\frac{dv}{dx}(x)+q(x)u(x)v(x)\right)dx\\ &=\int_a^b f(x)v(x)dx. \qquad (4.17)\end{aligned}$$

式 (4.17) は**弱形式**や**変分形式**と呼ばれる．有限要素法では問題 (4.15),(4.16) を解く代わりに，任意の関数 v に対して弱形式 (4.17) を満たす関数 u を求めることで解を求める．

4.4.3 弱形式の離散化

任意の関数 v に対して，弱形式 (4.17) を満たす関数 u を計算機を使って求めたいが，このままでは関数 u, v の取りうる自由度が無限にあるため，計算機で扱うことは困難である．そこで**離散化**を行うことで関数 u, v が属する空間を有限次元空間に制限し，弱形式 (4.17) を計算機で扱える式に帰着する．

例えば関数 $u(x)$ が n 次元ベクトル空間に属する場合，u は**基底関数** $t_i(x), (i=1,\cdots n)$ と係数 $c_i\in\mathbb{R}$ を用いて，

$$u(x)=\sum_{i=1}^n c_i t_i(x) \quad (4.18)$$

と書ける．基底 t_i の選び方は様々なものが考えられるが，ここでは折れ線関数による近似を考えよう．区間 (a,b) の分割 $a=x_0<x_1<\cdots<x_{n-1}<x_n=b$ に対して，$h_i=x_i-x_{i-1}\,(i=1,\cdots n)$ とし，$t_i(x)$ を

$$t_i(x)=\begin{cases}\frac{x-x_{i-1}}{h_i} & (x_{i-1}\le x<x_i)\\ \frac{x_{i+1}-x}{h_{i+1}} & (x_i\le x<x_{i+1})\\ 0 & \text{その他}\end{cases} \quad (4.19)$$

とすると，式 (4.18) で表される関数 $u(x)$ は点 $u(x_i)=c_i$ を満たす折れ線関数になる（→4.4.4 節演習課題）．

関数 $u(x)$ に式 (4.18) で表される折れ線関数を代入し，さらに $v=t_i$ を代入すると，式 (4.17) の

左辺は

$$
\begin{aligned}
&a\left(\sum_{j=1}^{n} u_j t_j, t_i\right) \\
&= \int_a^b \left\{ p(x)\frac{d}{dx}\left(\sum_{j=1}^{n} u_j t_j(x)\right)\frac{dt_i}{dx}(x) \right. \\
&\quad \left. + q(x)\left(\sum_{j=1}^{n} u_j t_j(x)\right) t_i(x) \right\} dx \\
&= \sum_{j=1}^{n} u_j \int_a^b \left(p(x)\frac{dt_j}{dx}(x)\frac{dt_i}{dx}(x) + q(x)t_i(x)t_j(x) \right) dx \\
&\quad (i = 1, \cdots n),
\end{aligned}
$$

右辺は

$$\int_a^b f(x)t_i(x)dx \quad (i = 1, \cdots n)$$

となるので，これら n 本の式をまとめて行列表示で書き下すと

$$Au = b \tag{4.20}$$

となる．ここに n 次正方行列 A の ij 成分，および n 次ベクトル u, b の i 成分は，それぞれ

$$(A)_{ij} = \int_a^b \left(p\frac{dt_i}{dx}\frac{dt_j}{dx} + qt_i t_j \right) dx \tag{4.21}$$

$$(u)_i = u_i$$

$$(b)_i = \int_a^b f(x)t_i(x)dx \tag{4.22}$$

である．

4.4.4 4.4 節の課題

1. 式 (4.19) で定義された $t_i(x)$ によって，点 x_i を頂点とする任意の折れ線関数が式 (4.18) で表せることを確かめよ．すなわち，式 (4.18) で表される関数 $u(x)$ は $u(x_i) = c_i\,(i = 0, \cdots, n)$ を満たし，区間 $(x_{i-1}, x_i)\,(i = 0, \cdots, n)$ において一次関数になることを示せ．

2. 式 (4.20) の解を u_i とし，$\overline{u}(x) := \sum_{j=1}^{n} u_j t_j(x)$ とする．このとき任意の折れ線関数 $\widetilde{v}(x) = \sum_{i=1}^{n} \widetilde{v}_i t_i(x)$ に対して，

$$a(\overline{u}, \widetilde{v}) = \int_a^b f(x)\widetilde{v}(x)dx$$

が成り立つことを示せ．

3. 以下の 1 次元 Strum-Liouville 型境界値問題

$$-\frac{d}{dx}\left(\mathrm{e}^{-x^2}\frac{du}{dx}(x)\right) - 6\mathrm{e}^{-x^2}u(x) = 0, \quad 0 < x < 1$$

$$u(0) = 0, \quad u(1) = -4$$

を有限要素法で解くコードを作成する．

   (a) $u(x) = 8x^3 - 12x$ がこの境界値問題の解であることを確かめよ．

   (b) この境界値問題を式 (4.15), (4.16) の形の境界値問題へ変形せよ．(Hint: $v(x) = u(x) + 4x$ として，v の満たす境界値問題を導出する．)

   (c) $\frac{dt_i}{dx}(x)$ を求めよ．

   (d) 式 (4.21) と (4.22) を計算するコードを作成せよ（Hint: 与えられた x に対して被積分関数を計算する関数を作り，これに数値積分を適用すれば良い．）．

   (e) 有限要素法のコードを完成させよ（Hint: (d) で作った線形方程式を Gauss の消去法などで解けば良い）．

付録 A. 重積分

本章では全て 1 次元の数値積分を取り上げたが，より高次元の領域において数値積分を行う場合は，次元の大きさによって扱いが大きく異なる．次元が 4 以下程度の比較的低い場合においては，重積分を複数の一次元積分に帰着し，本章で紹介した数値積分公式を用いる方法が一般的である．例えば $\mathbb{R}^2$ における長方形領域 $(x_1, x_2) \times (y_1, y_2)$

上での積分は

$$\int\int_{(x_1,x_2)\times(y_1,y_2)} f(x,y)dxdy = \int_{y_1}^{y_2}\int_{x_1}^{x_2} f(x,y)dxdy$$

として，1 次元の数値積分公式をそれぞれの積分に適用すればよい．3 次元における直方体領域に対しても同じようにできることが容易にわかる．また長方形や直方体以外の領域上での数値積分も工夫することで 1 次元積分に帰着できることがある．特によく使われる 2 次元の三角形領域や 3 次元の四面体領域上での数値積分は，積分点や重みの数表が作られ，半ば数値積分公式として扱われている [5].

また次元が 5 次以上の高次元領域における積分は，本章で扱った関数補間に基づく積分公式が使われることはあまりなく，**モンテカルロ法**などの統計的な方法で計算されることが多い．

付録 B. 被積分関数が特異性を持つ場合

被積分関数が積分区間内に特異性を持つ場合，Newton-Cotes 公式や Gauss 型積分公式をそのまま適用すると精度が悪化することが多い．この問題の対処法はいくつか考えられる．(→4.3.5 節問題 2)

変数変換 積分変数に対して適切な変数変換を施すことで特異性を消すことができる場合がある（→ 演習課題）．また上手い変数変換を施すことで特異性のある関数の積分の精度を高めた積分公式として，**二重指数関数型数値積分公式** [4] が知られている．

解析的なアプローチ 被積分関数の特異性がどの程度かわかっている場合は，被積分関数を “特異性があるが積分が解析的に行える関数” と “特異性の無い関数” の和に分割できることがある．

特異性に対応した Gauss 型数値積分公式 被積分関数の特異性がどの程度かわかっている場合には，その特異性に対応した Gauss 型積分公式を構成できることがある．例えば原点で $\log x$ の特異性 [6] や $1/x$ の特異性（Cauchy の主値積分）[7] を持っている場合の Gauss 型積分公式が知られている．

参考文献

[1] 山本哲朗, “数値解析入門 [増訂版]”, サイエンス社，2003 年 6 月．

[2] 杉原正顯，室田一雄，“数値計算法の数理”，岩波書店，1994 年 11 月．

[3] 一松信，“数値解析”，朝倉書店，1982 年 10 月．

[4] 森正武，“数値解析”，共立出版，1973 年 9 月．

[5] 菊地文雄，齊藤宣一 ，“数値解析の原理: 現象の解析をめざして”，岩波書店，2016 年 8 月

[6] S. Kapur and V. Rokhlin, “High-order corrected trapezoidal quadrature rules for singualr functions”, *Siam Journal on Numerical Analysis*, Vol. 34, No. 4, pp. 1331–1356, 1997

[7] D. Elliot and D. F. Paget, “Gauss type quadrature rules for Cauchy principal value integrals”, *Mathematics of computation*, Col. 33, Num. 145, pp. 301–309, 1979

[新納和樹]

第5章　連続最適化

5.1　目的

与えられた関数 $f\colon \mathbb{R}^n \to \mathbb{R}$ の零点，および最小値または最大値を与える点 $x \in \mathbb{R}^n$ を求める手法を学び，プログラミング言語 Python を用いた演習を通じて，その手法を理解する．内容は，5.2〜5.5 節の大きく 4 つに分かれており，5.2 節は関数の零点を求める二分法とニュートン法について学び，Python を導入する．5.3 節は降下法の概要を述べ，最適化手法として根幹を成す最急降下法およびニュートン法について学ぶ．5.4 節は，直線探索を用いたニュートン法および最適化手法の収束性について学ぶ．5.5 節は，ニュートン法に修正を加えた手法について学び，総合演習を行う．

5.2　関数の零点探索と Python 導入

ここでは，関数の零点探索のための二分法とニュートン法を学び，Python の導入を行う．

5.2.1　関数の零点探索

与えられた関数 $f\colon \mathbb{R} \to \mathbb{R}$ に対して，$f(x) = 0$ を満たす点 x のことを関数 f の零点と呼ぶ．関数の零点探索は，様々な場面で必要とされる．例えば，行列 $A \in \mathbb{R}^{n\times n}$ の固有値を求める際には，固有多項式 $p_A(x) = \det(xI - A)$ の零点を求めることに帰着され，一方，連続関数 $F\colon \mathbb{R} \to \mathbb{R}$ の最適化では，導関数 F' の零点を求めることに帰着される．

このように場面に応じて関数 f は異なるが，零点探索はそれぞれの場面で重要な役割を果たしている．ここでは，計算機を用いた関数の零点探索を考える．はじめに，関数の零点探索のための反復法として知られる二分法を以下に与える．

二分法

Step 0: $f(a) < 0$, $f(b) \geq 0$ を満たす初期点 a, b を選び，終了条件 $\varepsilon > 0$ を決める．

Step 1: a と b の中間点 $c := (a+b)/2$ を求める．もし，c が終了条件 $|f(c)| \leq \varepsilon$ を満たしていれば，c を解として終了する．

Step 2: もし，$f(c) < 0$ であれば $a \leftarrow c$ とし，$f(c) \geq 0$ であれば $b \leftarrow c$ として，Step 1 へ戻る．

二分法は，関数 f が $\mathbb{R}$ 上で連続であれば必ず収束する．実際，Step 1 の初期条件を満たす a, b を選べば，関数 f の零点は a と b の間に存在するので，a と b の間隔を繰り返し 1/2 に縮小していけば，中間点 c は以下の図で示すように関数 f の零点へ近づいていくことがわかる．

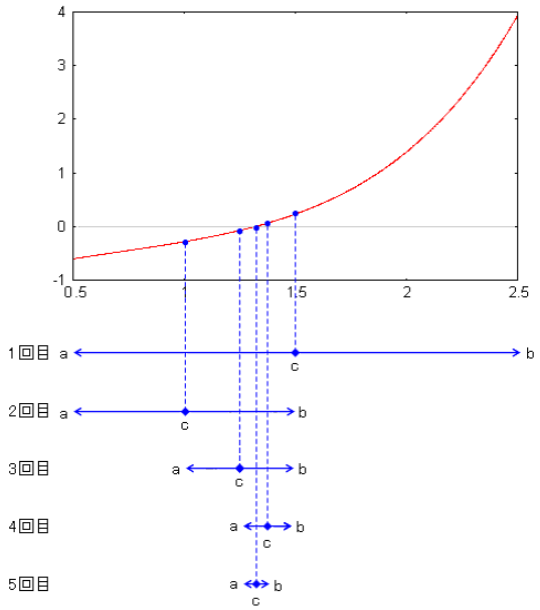


図 5.1: 二分法の生成点列

二分法は，アルゴリズムが簡易的であり，二つの異なる初期点 a, b の間に零点が存在すれば，必ず収束するという優れた性質を持つ．しかし，一般的には収束が遅いことで知られる．より収束が速い手法としては，次に与えるニュートン法が良く知られている．

ニュートン法

Step 0: 初期点 x_0 を選び，終了条件 $\varepsilon > 0$ を決める．$k := 0$ とする．

Step 1: ニュートン方程式

$$f'(x_k)\Delta x_k = -f(x_k)$$

を解き，Δx_k を求める．

Step 2: 点列を $x_{k+1} := x_k + \Delta x_k$ により更新する．もし，$|f(x_{k+1})| \leq \varepsilon$ を満たしていれば，x_{k+1} を解として終了する．

Step 3: $k \leftarrow k+1$ として，Step 1 へ戻る．

ニュートン法は，点 x_k の周りで関数 f を一次近似し，接線の方程式 $y = f'(x_k)(x-x_k)+f(x_k)$ を考える．この方程式と x 軸の交点を求め，更新点とする（つまり，$f'(x_k)(x-x_k)+f(x_k)=0$ を解き，$x = x_k - f(x_k)/f'(x_k)$ を更新点とする）ことで点列を生成する反復法である．一般的に収束は速いが，解の近くに初期点を選ばなければ収束しないということが知られている．

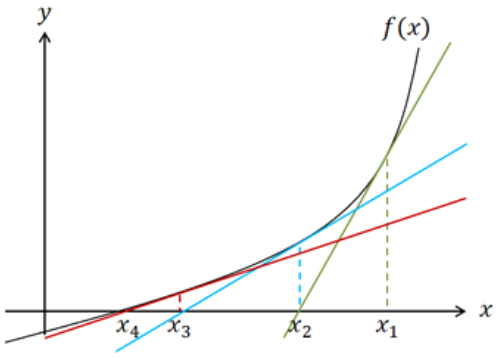


図 5.2: ニュートン法の生成点列

5.2.2 Python 導入

ここでは，Python の導入を行う．Python 3.0，数値計算ライブラリの Numpy，グラフ描写ライブラリの Matplotlib が PC にインストールされている場合，以下の課題は飛ばしても良い．

課題 1 インターネット上のチュートリアルを参考にして，Anaconda3 を用いて Python を PC にインストールせよ．例えば，以下のリンクが参考となる．

- `https://sukkiri.jp/technologies/ides/anaconda-win_install.html`

Python ヒント

Python はスクリプト言語なので，実行方法が 2 通りある：(a) 直接プログラムを 1 行ずつ打ち込む方法と，(b) プログラムファイルを作成し，インタプリタに一気に実行させる方法．(a) の場合は，簡単な計算の確認などのテストをするために便利であり，ターミナル上に `python` コマンドを入力することによってインタプリタが実行される．(b) の場合は，「`.py`」という拡張子のファイル（例えば，`test.py`）を用いて，ターミナル上に `python test.py` を入力し，プログラムを実行する．また，anaconda3 をインストールした際に，Spyder と呼ばれる Python 専用のエディターが同時にインストールされるので，こちらを用いるのも良い．

Python ヒント

本演習では，ライブラリ Numpy・Matplotlib が必要となる．インポートするために

```
import numpy as np
import matplotlib.pyplot as plt
```

を実行する，またはファイルに入力する．ここでは，Numpy の組込み関数を有効に使う．コンパイラ言語 C と違って，例えば，

```
w = np.random.rand(4)
```

を入力すると，ランダムな 4 次元ベクトルが `while` 文や `for` 文なしで生成される．同様に，

```
y = np.random.rand(4)
sum = w + y
```

を追加すると，2 つのランダムなベクトルの和が簡単に得られる．

課題 2 次の関数 $f\colon \mathbb{R} \to \mathbb{R}$ を考える．

$$f(x) := x^3 + 2x^2 - 5x - 6$$

Python を用いて，

(a) 関数 f を $-10 \leq x \leq 10, -10 \leq y \leq 10$ の範囲で描写せよ．

(b) 関数 f の零点を二分法を用いて求めよ．ただし，二つの異なる初期点は，(a) で描写した関数 f の概形を参考にして適切に設定せよ．

(c) 関数 f の零点をニュートン法を用いて求めよ．ただし，(a) で描写した関数 f の概形を参考にして，零点の近くに初期点を設定せよ．

Python ヒント

グラフ描写は以下を参考にすると良い．例えば，$y = \sin(x)$ を $-5 \leq x \leq 5, -5 \leq y \leq 5$ の平面内に描写する際は，Numpy・Matplotlib をインポートしてから以下を入力する．

```
x = np.arange(-5,5,0.1)
y = np.sin(x)
plt.plot(x, y, 'red')
plt.xlim(-5,5)
plt.ylim(-5,5)
plt.axhline(0, c='black')
plt.axvline(0, c='black')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Graph of y=sin(x)')
plt.show()
```

5.3 数理最適化の基礎と降下法

ここでは，まず最適化の基礎を学ぶ．さらに，最適化手法として代表的な降下法の概要を説明し，具体例として最急降下法とニュートン法を学ぶ．

5.3.1 関数の最大化・最小化

工学的分野に限らず，様々な分野において，与えられた関数の最小値，または最大値を求めることは重要である．例えば，企業においては，その利益を最大化することを目的としており，また，統計の分野における最小二乗法は，与えられたデータとの誤差を最小にする近似曲線を求めることを目的としている．

各分野において，その最小化または最大化しようとする関数は異なるが，ここでは，一般に $f\colon \mathbb{R}^n \to \mathbb{R}$ で与えられているとする．このとき，その関数の最小化問題は，

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && x \in X \end{aligned} \tag{5.1}$$

として定式化される．これは，「集合 $X \subseteq \mathbb{R}^n$ の中から，関数値 $f(x)$ を最小とする x を求めよ」という問題である．ここで，f をこの最小化問題の**目的関数**と呼び，集合 X を**制約集合**と呼ぶ．さらに，制約条件を満たす点を**実行可能解**という．制約集合 X が $X = \mathbb{R}^n$ のとき，問題 (5.1) を**無制約最小化問題**という．本演習では，制約無し最小化問題のみを扱う．また，最大化問題については，最大化する関数が $\tilde{f}$ で与えられているとき，この問題は $f(x) := -\tilde{f}(x)$ の最小化問題へと同値に変形することができるため，ここでは最小化問題のみを考える．

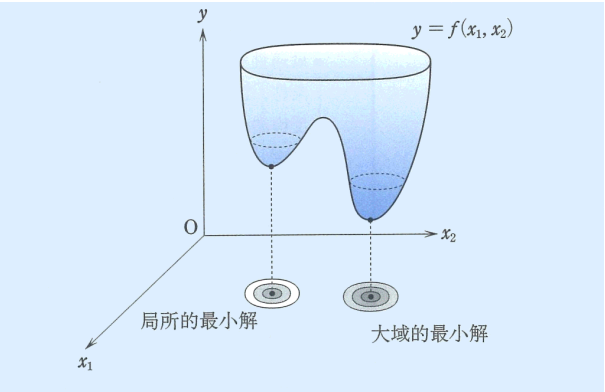


図 5.3: 大域的最小解，局所的最小解

最小化問題において，全ての実行可能解 x の関数値よりも多くならない関数値をもつ点 $\bar{x} \in \mathbb{R}^n$ を，その問題の**大域的最小解**と呼ぶ (図 5.3)．つまり，次が成り立つ．

$$f(\bar{x}) \leq f(x) \quad \text{for all } x \in \mathbb{R}^n$$

しかし，大域的最小解であるかどうかの判定には，$\mathbb{R}^n$ 上の全ての点の情報が必要となるため，一般的に大域的最小解を求めることは難しい．このため，次の局所的最小解という概念が重要となる．ここで，$\hat{x}$ が**局所的最小解**とは，$\hat{x}$ を含むある近傍 $N \subseteq \mathbb{R}^n$ を取ることができて，その近傍内では $f(\hat{x})$ よりも関数値が小さくなる実行可能解が存在しないことである (図 5.3)．つまり，$\hat{x}$ を含むある近傍 N が存在し，次を満たすことである．

$$f(\hat{x}) \leq f(x) \quad \text{for all } x \in N$$

もし，$N = \mathbb{R}^n$ ならば，局所的最小解は大域的最小解となる．関数 f が微分可能であるとき，

$$\nabla f(x) := \left(\frac{\partial f(x)}{\partial [x]_0}, \ldots, \frac{\partial f(x)}{\partial [x]_{n-1}} \right)^\top = 0$$

を満たす点 x のことを f の停留点と呼ぶ．ただし，$[x]_k$ $(k = 0, 1, \ldots, n-1)$ はベクトル x の第 k 成分であり，$\top$ は転置を表す．また，$\nabla f(x)$ は点 x における f の勾配ベクトルと呼ばれる．もし，$\hat{x}$ が局所的最小解であれば，$\hat{x}$ は f の停留点となる．しかし，f の停留点が f の局所的最小解であるとは限らない．局所的最大解や鞍点の場合もあるからである (図 5.4)．

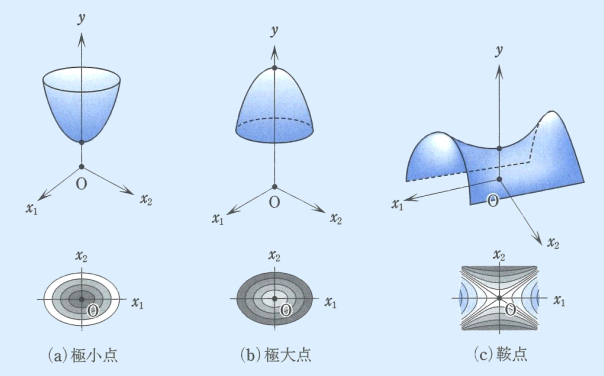


図 5.4: 局所的最大解と鞍点

本演習で扱う最急降下法およびニュートン法は，基本的に停留点を求める手法であり，必ずしも局所的最小値を求められるわけではない．しかしながら，大抵の問題に対しては，それらの手法によって局所的最小解を求めることができる．また，局所的最小値を求めることができる複雑な手法も，その根本的なところで，最急降下法やニュートン法を用いている．このため，それらの手法を理解することは重要である．

5.3.2 降下法

最急降下法やニュートン法は降下法と呼ばれる手法である．降下法は，関数の値を減少させる点列 $\{x^0, x^1, x^2, \ldots\}$ を生成していく反復法である[1]．つまり，

$$f(x^0) > f(x^1) > \cdots$$

となる点列 $\{x^k\}$ を生成する．この点列は

$$x^{k+1} := x^k + t_k d^k$$

で与えられる．ここで，d^k を**探索方向**，t_k を**ステップサイズ**と呼ぶ．降下法では，前提として，d^k は次の条件を満たすと仮定する．

$$\langle d^k, \nabla f(x^k) \rangle < 0$$

ここで，$\langle y, z \rangle$ はベクトル $y, z \in \mathbb{R}^n$ の内積を表している．つまり，$\langle y, z \rangle := \sum_{i=0}^{n-1} [y]_i [z]_i$ である．このような条件を満たしてるベクトル d^k のことを**降下方向**と呼ぶ．

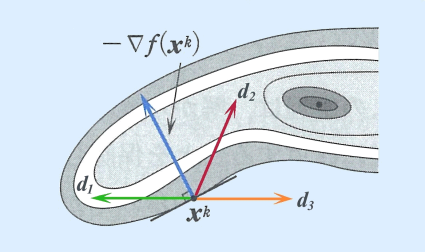


図 5.5: 降下方向

図 5.5 は関数 f の等高線を表しており，図中の d_1 および d_2 は降下方向であるが，d_3 は降下方向ではない．降下方向へ適切に進めば，関数値は減少するが，進みすぎると増加する場合がある．そのため，$f(x^k) > f(x^{k+1})$ となるように，t_k を

[1] $\mathbb{R}^n (n \geq 2)$ の点列を表す際は，上付き添え字を用いる．つまり，x^k は $\mathbb{R}^n$ の点列 $\{x^0, x^1, x^2, \ldots\}$ における k 番目の項を表す．なお，$\mathbb{R}$ や $\mathbb{R}^{n \times n}$ の点列を表す際は，冪乗との混同を避けるため下付き添え字を用いる．

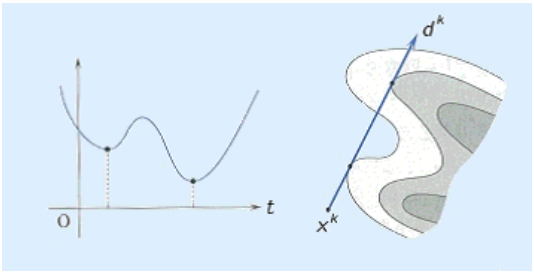


図 5.6: 直線探索

うまく設定する必要がある．この t_k を決める方法は**直線探索法**と呼ばれ，いくつか提案されているが，本テキストではバックトラック法を学ぶ．バックトラック法の詳細は 5.4 節で述べる．図 5.6（右）は探索方向を表す．図 5.6（左）は探索方向 d^k へ進んだ際の目的関数値の変化を表すグラフである．

バックトラック法を組み込んだ降下法を以下に与える．ただし，$\|z\| := (\sum_{i=0}^{n-1}[z]_i^2)^{1/2}$ である．

降下法

Step 0: 初期点 x^0，終了条件 $\varepsilon > 0$ を決める．$k := 0$ とする．

Step 1: もし，x^k が終了条件 $\|\nabla f(x^k)\| \leq \varepsilon$ を満たしていれば，x^k を解として終了する．

Step 2: 次式を満たす降下方向 d^k を定める．

$$\langle d^k, \nabla f(x^k)\rangle < 0$$

Step 3: バックトラック法を用いて，ステップサイズ t_k を決定する．

Step 4: 点 x^k を $x^{k+1} := x^k + t_k d^k$ と更新する．$k \leftarrow k+1$ として，Step 1 へ戻る．

注意 5.3.1 Step 1 における終了条件のパラメータ ε は，$\varepsilon = n \times 10^{-6}$ のように設定されることが多い．これは，一般に問題の次元 n が大きくなるにつれて，計算誤差が大きくなるため，この誤差を許容するという意味がある．

注意 5.3.2 反復回数 (k) が十分大きいとき，「解が求まらない」と判断する．よって，許容される最大反復回数の設定が必要である．

この降下法によって生成される点列 $\{x^k\}$ に対して，次の収束に関する定理が示されている．証明に興味がある方は文献 [1] を読んでもらいたい．

定理 5.3.1 点列 $\{x^k\}$ および $\{d^k\}$ は，それぞれ降下法により生成される有界列とし，$\{d^k\}$ は降下方向の列であると仮定する．このとき，ある正の定数 γ が存在し，全ての $k \in \mathbb{N} \cup \{0\}$ に対して，

$$\langle d^k, \nabla f(x^k)\rangle \leq -\gamma \|\nabla f(x^k)\|^2$$

が成り立つならば，点列 $\{x^k\}$ の任意の集積点 x^* は，関数 f の停留点となる．つまり，$\nabla f(x^*) = 0$ が成り立つ．

この定理により，上手に降下方向を選んでやれば，問題の停留点 (多くの場合，局所最小解) を求めることができる．次節以降で紹介する最急降下法とニュートン法は，この定理の降下方向に関する仮定を満たす手法である．

5.3.3 最急降下法とニュートン法

ここでは，最急降下法およびニュートン法の順に説明をする．まず，最急降下法は，点 x^k が与えられたときに，降下方向として

$$\text{[最急降下方向]} \qquad d^k := -\nabla f(x^k)$$

を用いる降下法である．いま，$\nabla f(x^k) \neq 0$ と仮定する．(もし，$\nabla f(x^k) = 0$ であれば，点 x^k は既に f の停留点となっている．) このとき，

$$\langle d^k, \nabla f(x^k)\rangle = -\|\nabla f(x^k)\|^2 < 0$$

が成り立つため，d^k は降下方向であり，定理 5.3.1 の仮定を満たすことがわかる．(実際，上記の等号により，定理 5.3.1 における正の定数 γ として，$\gamma = 1$ の存在が確かめられる．)

一方，ニュートン法は，x^k が与えられたときに，降下方向として

$$\text{[ニュートン方向]} \qquad d^k := -\nabla^2 f(x^k)^{-1} \nabla f(x^k)$$

を用いる降下法である．ただし，$\nabla^2 f(x)$ は x における f のヘッセ行列を表しており，

$$\begin{bmatrix} \frac{\partial^2 f(x)}{\partial [x]_0^2} & \frac{\partial^2 f(x)}{\partial [x]_0 [x]_1} & \cdots & \frac{\partial^2 f(x)}{\partial [x]_0 [x]_{n-1}} \\ \frac{\partial^2 f(x)}{\partial [x]_1 [x]_0} & \frac{\partial^2 f(x)}{\partial [x]_1^2} & \cdots & \frac{\partial^2 f(x)}{\partial [x]_1 [x]_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial [x]_{n-1} [x]_0} & \frac{\partial^2 f(x)}{\partial [x]_{n-1} [x]_1} & \cdots & \frac{\partial^2 f(x)}{\partial [x]_{n-1}^2} \end{bmatrix}$$

で与えられる．ここで，$\{\nabla^2 f(x^k)\}$ は有界かつ一様正定値（詳細は 5.5 節で解説する）であると仮定する．これは，ある $\mu > 0$ が存在して，全ての $k \in \mathbb{N} \cup \{0\}$ と $v \in \mathbb{R}^n$ に対して，

$$\frac{1}{\mu}\|v\|^2 \leq \langle \nabla^2 f(x^k)^{-1} v, v \rangle \leq \mu \|v\|^2$$

が成り立つことと同値である．このとき，すべての k に対して，$\nabla f(x^k) \neq 0$ であれば，

$$\begin{aligned} \langle d^k, \nabla f(x^k) \rangle &= -\langle \nabla^2 f(x^k)^{-1} \nabla f(x^k), \nabla f(x^k) \rangle \\ &\leq -\frac{1}{\mu}\|\nabla f(x^k)\|^2 \\ &< 0 \end{aligned}$$

となるため，d^k は降下方向であり，定理 5.3.1 の仮定を満たすことがわかる．

課題 3 関数 $f\colon \mathbb{R} \to \mathbb{R}$ を以下で定義する．

$$f(x) := \frac{1}{3}x^3 - x^2 - 3x + \frac{5}{3}$$

Python を用いて，

(a) 最急降下法を実装し，関数 f の停留点を求めよ．ただし，初期点は $x_0 := 1/2$ と設定し，各反復 $k \in \mathbb{N} \cup \{0\}$ において，ステップサイズは $t_k := 1/(k+1)$ を採用せよ．

(b) ニュートン法を実装し，関数 f の停留点を求めよ．ただし，初期点は $x_0 := 5$ と設定し，各反復 $k \in \mathbb{N} \cup \{0\}$ において，ステップサイズは $t_k := 1$ を採用せよ．

5.4 直線探索と最適化手法の収束性

ここでは，前半で 5.3 節の降下法の概説で触れた直線探索法について学ぶ．特に，直線探索法の一種であるバックトラック法を詳説する．後半では，反復法の収束性について学ぶ．

5.4.1 直線探索法

直線探索法は，ステップサイズ $t_k > 0$ を決定するための手法であり，点 x_k を x_{k+1} へ更新する際に，$f(x_{k+1}) < f(x_k)$ を満たすようにするために行う．これまでにいくつかの直線探索法が提案されているが，以下に与えるバックトラック法は理論的にも実用的にも優れていることが知られている．

バックトラック法

Step 0: $\xi \in (0,1)$，$\rho \in (0,1)$，初期ステップサイズ $\bar{t} > 0$ を選ぶ．$t = \bar{t}$ とする．

Step 1: 次式を満たすまで $t \leftarrow \rho t$ とする．

$$f(x^k + td^k) \leq f(x^k) + \xi t \langle d^k, \nabla f(x^k) \rangle$$

Step 2: $t_k = t$ として終了．

Step 1 の不等式は**アルミホ条件**と呼ばれ，その意味は図 5.7 で表す．横軸はステップサイズ t，縦軸は $\varphi(t) := f(x^k + td^k)$ を表す．原点における φ の接線は $y = f(x^k) + t\langle d^k, \nabla f(x^k) \rangle$ である．アルミホ条件を満たすステップサイズを選ぶことは，φ の接線の傾きを緩和して得られる直線 $y = f(x^k) + \xi t \langle d^k, \nabla f(x^k) \rangle$ よりも関数値が小さくなる t の区間からステップサイズを選ぶことに対応していると言える．ξ を小さく設定すれば，アルミホ条件を満たすステップサイズを速く見つけることができる．

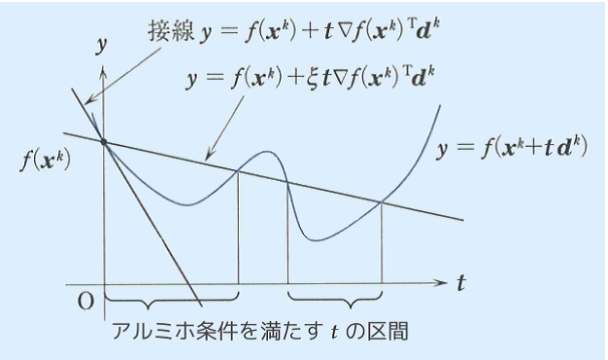


図 5.7: アルミホ条件を満たすステップサイズ

5.4.2 反復法の収束性について

ここまで，最急降下法およびニュートン法について学び，定理 5.3.1 によって，どちらの手法を用いても問題の解（多くの場合，局所最小解）を求められることがわかった．しかし，ここまでの議論では，どれだけ速く解を得ることができるかはわからない．これについて議論するためには，反復法の解への収束の速さに関する概念が必要となる．

ここで，x^* を求める問題の解とし，$\{x_k\}$ は x^* へ収束するとしよう．このとき，ある $\kappa \in (0,1)$ と $m_0 \in \mathbb{N}$ が存在して，

$$\|x^{k+1} - x^*\| \leq \kappa \|x^k - x^*\| \quad \text{for all } k \geq m_0$$

が成り立つとき，点列 $\{x^k\}$ は解 x^* に **1 次収束**するという．一方，

$$\lim_{k\to\infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|^2} < \infty$$

が成り立つとき，点列 $\{x^k\}$ は解 x^* に **2 次収束**するという．これは，ある $\eta > 0$ と $n_0 \in \mathbb{N}$ が存在して，

$$\|x^{k+1} - x^*\| \leq \eta \|x^k - x^*\|^2 \quad \text{for all } k \geq n_0$$

となることを意味しており，一旦 $\|x^k - x^*\| < 1$ となると，非常に速く収束する．

ここで，次の点列 $\{y^k\}$ と $\{z^k\}$ を考えてみよう．

$$y_k = 0.5^k, \quad z_k = 0.5 z_{k-1}^2$$

ただし，$z_0 = 1$ とする．点列 $\{y_k\}$ は，0 に 1 次収束することが確かめられる．一方, 点列 $\{z_k\}$ は，0 に 2 次収束することが確かめられる．これらの点列を計算してみると，表 5.1 となる．表 5.1 より，2 次収束する点列は，1 次収束する点列に比べて，極めて速い収束性を持つことがわかる．

ここまでに学んだ最急降下法は，適切な仮定の下で一次収束することが示されている．しかし，最小化する関数の一階微分（つまり一次の情報）しか用いていないため，二次収束しないことが多い．一方，ニュートン法は関数のヘッセ行列（つまり二次の情報）を用いた方法であり，適切な仮定の下で二次収束することが知られている．よって，最急降下法とニュートン法の解への収束性を比較すると，一般にニュートン法に軍配が上がる．

表 5.1: 1 次収束と 2 次収束

k	y_k	z_k
0	1	1
1	0.5	0.5
2	0.25	0.125
3	0.125	0.0078125
4	0.0625	0.0000305
⋮	⋮	⋮

このように収束性の観点から両手法を見ると，ニュートン法の方が最急降下法に比べて有能な反復法であるかのように思える．しかし，ニュートン法は毎回の反復において，降下方向の生成のためにヘッセ行列 $\nabla^2 f(x^k)$ および勾配 $\nabla f(x^k)$ を計算し，ニュートン方程式 $\nabla^2 f(x^k) d^k = -\nabla f(x^k)$ を解かなければならないため，勾配 $\nabla f(x^k)$ を計算するだけの最急降下法に比べて一反復当たりの計算負荷が大きい．従って，問題の規模が大きくなれば，最急降下法では計算が可能であっても，ニュートン法では計算が困難となることも有り得る．このように，解への収束が速いニュートン法の方が，最急降下法に比べて有能な反復法であるとは限らないことに注意して欲しい．

課題 4 次の 2 次元の最適化問題を考える．

$$\begin{aligned} \text{minimize} \quad & f(x) := x_0^2 + e^{x_0} + x_1^4 + x_1^2 \\ & \qquad\qquad -2x_0x_1 + 3 \\ \text{subject to} \quad & x := (x_0, x_1)^\top \in \mathbb{R}^2 \end{aligned}$$

Python を用いて，

(a) 点 x が与えられたとき，目的関数の値 $f(x)$ を出力する関数を作成せよ．

(b) 点 x が与えられたとき，勾配ベクトル $\nabla f(x)$ を出力する関数を作成せよ．

(c) 点 x が与えられたとき，ヘッセ行列 $\nabla^2 f(x)$ を出力する関数を作成せよ．

Python ヒント

- 課題 4 を実施する前に，$f(x) := x_0^2 + x_1$ の場合を考える．このとき，

$$\nabla f(x) = \begin{bmatrix} 2x_0 \\ 1 \end{bmatrix}, \nabla^2 f(x) = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$$

である．

- 以下のコードをファイルに入力し，実行してみる．コードも理解しておく[a]．

```
#!/usr/bin/env python
# -*- coding:  utf-8 -*-

import numpy as np

def evalf(x):
  # 目的関数の値 f(x) を計算する
  f = x[0]**2 + x[1]
  return f

def evalg(x):
  # f の勾配を計算する
  g = np.array([2*x[0], 1])
  return g

def evalh(x):
  # f のヘッセ行列を計算する
  H = np.array([[2, 0], [0, 0]])
  return H

def main():
  x = np.array([0.3, 5])
  f = evalf(x)
  g = evalg(x)
  H = evalh(x)
  print('f =', f, '\ng =', g, \
    '\nH = \n', H)
if __name__ == '__main__':
  main()
```

[a] Python 3 を使った例である．

課題 5 Python を用いて，

(a) バックトラック法を用いた最急降下法を実装し，課題 4 の最適化問題を解け．ただし，初期点は $x^0 = (1,1)^\top$ を用いることとし，バックトラック法における ξ，ρ，$\bar{t}$ は，それぞれ $\xi = 10^{-4}$，$\rho = 0.5$，$\bar{t} = 1$ と設定せよ．

(b) バックトラック法を用いたニュートン法を実装し，課題 4 の最適化問題を解け．ただし，初期点は $x^0 = (1,1)^\top$ を用いることとし，バックトラック法における ξ，ρ，$\bar{t}$ は，それぞれ $\xi = 10^{-4}$，$\rho = 0.5$，$\bar{t} = 1$ と設定せよ．

5.5 最適化手法の実用的な修正方法と総合演習

5.3 節で学習したニュートン法は，$\{\nabla^2 f(x^k)\}$ が一様正定値であることを仮定していたが，この仮定は非常に強い仮定であり，成り立たないこともしばしばある．ここでは，この仮定を満たさない場合にも適用可能となるように，ニュートン法を修正する方法について学ぶ．また，Python を用いて最急降下法とニュートン法を実装し，いくつかの最適化問題を解く総合演習を行う．

5.5.1 ニュートン方向の修正

まず，対称行列の正定値性と固有値の関係について学ぶ．対称行列 $M \in \mathbb{R}^{n\times n}$ が正定値であるとは，以下を満たすことである．

$$\langle Mv, v\rangle > 0 \quad \text{for all } v \in \mathbb{R}^n \backslash \{0\}$$

以降，対称行列 M が正定値であることを $M \succ O$ と表す．次の定理は，正定値性と固有値の関係を表すものであり，非常に重要な性質である．

定理 5.5.1 対称行列 $M \in \mathbb{R}^{n\times n}$ が正定値であることは，その固有値 $\lambda_j(M)$ $(j = 1, \ldots, n)$ が全て正であることと同値である．

また，$\mathbb{R}^{n\times n}$ の点列 $\{M_k\}$ が有界かつ一様正定値であるとは，ある $\kappa > 0$ が存在して，全ての $k \in \mathbb{N} \cup \{0\}$ に対して，

$$\frac{1}{\kappa}\|v\|^2 \le \langle M_k v, v\rangle \le \kappa \|v\|^2 \quad \text{for all } v \in \mathbb{R}^n$$

が成り立つことであり，これは，

$$\frac{1}{\eta}\|v\|^2 \le \langle M_k^{-1}v, v\rangle \le \eta\|v\|^2 \quad \text{for all } v \in \mathbb{R}^n$$

と同値である．ただし，$\eta := 1/\kappa$ である．

5.3 節で学んだニュートン法は，「$\{\nabla^2 f(x^k)\}$ が有界かつ一様正定値である」と仮定したが，一様正定値性については成り立たないことも多々あるため，このような場合に対応するために，次に述べるような修正を行う．各反復 k に対して，ヘッセ行列 $\nabla^2 f(x^k)$ に単位行列 I の τ 倍を加え，次の方程式を考える．

$$\big(\nabla^2 f(x^k) + \tau I\big)d^k = -\nabla f(x^k) \qquad (5.2)$$

ただし，$\tau > 0$ は $\nabla^2 f(x^k) + \tau I \succ O$ となるようにとる．もし，$\nabla^2 f(x^k) \succ O$ ならば，$\tau = 0$ とすれば良いので，$\nabla^2 f(x^k) \not\succ O$ の場合を考えよう．いま，$\nabla^2 f(x^k) + \tau I$ の固有値は $\lambda_j(\nabla^2 f(x^k)) + \tau$ $(j = 1, \ldots, n)$ となることを考慮すると，例えば τ として，$\nabla^2 f(x^k)$ の最小固有値の絶対値にある定数（10^{-1} や 10^{-2} など）を加えた値を設定すれば，$\lambda_j(\nabla^2 f(x^k)) + \tau > 0$ $(j = 1, \ldots, n)$ となり，定理 5.5.1 より $\nabla^2 f(x^k) + \tau I \succ O$ となる．よって，この τ を用いて，式 (5.2) から探索方向 d^k を生成する．このようなニュートン方向の修正は実用上，良く行われるものであり有用である．

5.5.2 総合演習

ここでは，Python を用いて，いくつかの最適化問題を解く演習を行う．

課題 6 Python を用いて，最急降下法およびニュートン法を実装し，次の最適化問題を解け．ただし，初期点は $x^0 = [2, 0]^\top$ を用いることとし，バックトラック法における ξ，ρ，$\bar{t}$ は，それぞれ $\xi = 10^{-4}$，$\rho = 0.5$，$\bar{t} = 1$ と設定せよ．

$$\begin{array}{ll} \text{minimize} & f(x) := \displaystyle\sum_{i=0}^{2} f_i(x)^2 \\ \text{subject to} & x \in \mathbb{R}^2 \end{array}$$

ただし，$f_i(x) := y_i - [x]_0(1 - [x]_1^{i+1})$ $(i = 0, 1, 2)$ と定義し，$y_0 = 1.5$, $y_1 = 2.25$, $y_2 = 2.625$ である．（参考：最適値は 0，最適解は $[3, 0.5]^\top$ である．）

ヒント

課題 6 の f の勾配とヘッセ行列は，それぞれ次のようになる．

$$\begin{aligned} \nabla f(x) &= 2\textstyle\sum_{i=0}^{2} f_i(x)\nabla f_i(x) \\ \nabla^2 f(x) &= 2\textstyle\sum_{i=0}^{2} \big(f_i(x)\nabla^2 f_i(x) \\ &\quad + \nabla f_i(x)\nabla f_i(x)^\top\big) \end{aligned}$$

課題 7 Python を用いて，最急降下法およびニュートン法を実装し，次の最適化問題を解け．ただし，初期点は要素が全て 1 のベクトルを用いることとし，バックトラック法における ξ，ρ，$\bar{t}$ は，それぞれ $\xi = 10^{-4}$，$\rho = 0.5$，$\bar{t} = 1$ と設定せよ．

$$\begin{array}{ll} \text{minimize} & f(x) := \frac{1}{2}\langle Ax, x\rangle \\ \text{subject to} & x \in \mathbb{R}^n \end{array}$$

ただし，$A \in \mathbb{R}^{n\times n}$ は，$[0, 1]$ の範囲の乱数を各要素に持つ行列 $Z \in \mathbb{R}^{n\times n}$ を用いて $A = Z^\top Z$ により定義される半正定値対称行列である．このとき，$n = 2, 5, 10$ のそれぞれに対して 5 回計算を実行すること．また，各手法と各 n に対して，反復回数（k）の平均を計算すること．

ヒント

課題 7 の f の勾配とヘッセ行列は，それぞれ次のようになる．

$$\begin{aligned} \nabla f(x) &= Ax \\ \nabla^2 f(x) &= A \end{aligned}$$

参考文献

[1] D.P. Bertsekas, “Nonlinear Programming”, 2nd edition, Athena Scientific (1999).

[2] 福島雅夫,「新版・数理計画入門」，朝倉書店 (2011).

[3] J. Nocedal and S.J. Wright, “Numerical Optimization”, 2nd edition, Springer Verlag (2006).

[山川雄也]

第6章　最小二乗法

6.1　目的

最小二乗法とは，与えられたデータに合う数理モデルあるいはそのパラメータを決定するために，二乗誤差を目的関数とした最小化問題を解いて推定値を得る手法のことである．この手法は最小二乗誤差推定とも呼ばれ，原理は単純ながらも強力で，実験データを扱う多くの場面で標準的に用いられる手法の 1 つである．最小二乗法を単に最適化問題の 1 つとみなすのではなく，最適化問題を構成するデータの取得方法や用いるデータ数，得られた最適化解の評価の仕方を工夫することも重要であり，課題を通して簡単なデータ分析ができるようになってほしい．本実験では，最小二乗法の基礎から逐次的な処理方法，さらに交互最小二乗法までを使えるようになり，データ処理の初歩を学ぶことを目的とする．

本稿では，２回前期配当科目の全学共通科目「確率論基礎」の知識があることが望ましいが，内容や課題の多くは高校数学の範囲内の確率・統計学の知識でカバーできるように記述した．そのため，回りくどい部分や天下り的な箇所もあるいが，それらを含む最小二乗法の統計的性質や発展的な内容は，3 回前期の「確率と統計」を受講するか，あるいは章末に挙げた参考文献 [1–5] などで自習してもらいたい．

- 課題ではデータをダウンロードする必要がある．PandA にファイルを置くので，適宜アクセスしてダウンロードすること．
- 課題を取り組むためのプログラミング言語は指定しないが，C 言語は補助資料を PandA に置いておく．

記法

$\mathbb{R}$ は実数全体を表し，n 次の数ベクトル空間を $\mathbb{R}^n$ で表す．また，$m \times n$ の実行列全体を $\mathbb{R}^{m\times n}$ で表す．とくに I_m は m 次の単位行列を表す．行列 A の転置は，$A^\top$ で表す．正方行列 A に対して $A \geq 0$ で半正定値対称行列を，$A > 0$ で正定値対称行列を表す．$\|\bullet\|$ は Euclid ノルムを表す．ただしノルムに下添字をつけることで，別のノルムを表すことがある．$\mathbb{E}$ は適当な確率分布による期待値演算であり，確率分布あるいは確率密度関数を陽に表すときには $\mathbb{E}_P$ と書く．確率変数 $w \in \mathbb{R}^m$ に対し $\mathcal{N}(\mu, V)$ は平均 $\mu \in \mathbb{R}^m$, 共分散行列 $V \in \mathbb{R}^{m\times m}$ の正規分布を表す．

6.2　最小二乗法とその評価方法

次の関係式を満たす方程式が与えられているとする．

$$z = f(\theta, x) \tag{6.1}$$

ここで $z \in \mathbb{R}^m$, $\theta \in \mathbb{R}^n$, $x \in \mathbb{R}^p$ とし，$f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m$ とする．「適当に固定されたパラメータ θ の下で，x を入れると z を返す」という意味で，これはパラメータを θ とし，x を入力，z を出力とするシステム f であるという．一般に観測には誤差を伴うため，入力 x や出力 z の値には誤差が乗る．最近の応用で多く見られるように，ここでは出力にのみ観測誤差のある場合を考えよう．入力に観測誤差がある場合も考えられるが，基本的には同じように議論できるため，本項では考えない．

誤差を $w \in \mathbb{R}^m$ とおき，加法的な誤差を考えると，観測値は次で与えられる．

$$y = z + w. \tag{6.2}$$

観測誤差は偶然誤差と系統誤差の 2 種類に分類されるが，本稿では偶然誤差のみを考える．特に断らない限り，本稿では偶然誤差は次の性質を持つものと仮定する．

1. 各試行ごとに観測誤差は同じ確率分布 P から確率的に発生する．

2. 入力 x やパラメータ θ とは独立で，観測ごとに独立に定まる．

3. i 回目の観測に対する誤差を $w_i \in \mathbb{R}^m$ で表す．このとき，次の 2 つが成り立つものとする．

$$\mathbb{E}_P[w_i] = 0,\ \mathbb{E}_P[w_i w_j^\top] = \delta_{i,j} V. \qquad (6.3)$$

ここで 0 は m 次のゼロベクトルを表し，V は m 次の正定値対称行列である．$\delta_{i,j}$ は Kronecker のデルタである．

同じ計測機器で計測することを考えるのならば，同じ条件で測定する限り，観測誤差は独立同一分布に従うものとみてもよいであろう．ただし，このような確率分布を正確に知ることは難しい．以降断らない限り，我々はこの観測誤差に関する確率分布は正確に分からないものとし，観測誤差は平均ゼロ，共分散行列が有界に存在することのみを知っているものとする．例として，図 6.1 に $f(x) = \theta x$，$\theta, x \in \mathbb{R}$ の場合の例を表す．

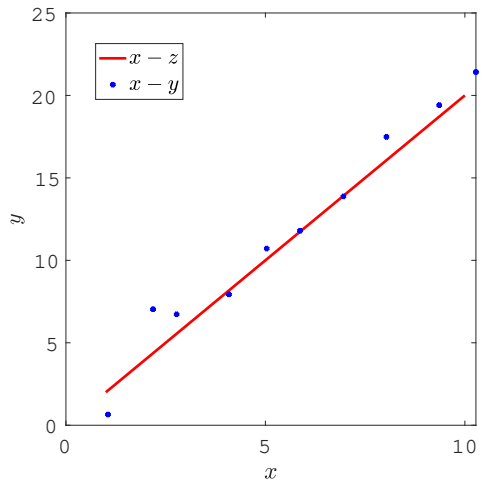


図 6.1: $f(x) = \theta x$ の場合．青色の点は観測値 y_i で，赤色の線が z_i の値．

ここで関数 f が既知であるが，パラメータ θ が未知である場合を考えよう．このとき，入力と観測値のデータから θ を決定したい．例えば，図 6.1 の青点から赤線を求める問題を考えたい．θ さえ適切に決められれば，未知のデータに対する出力の予測ができるようになり，様々な場面で恩恵を受けることができるからである．N 組の入力および観測値のデータ $\{(x_i, y_i)\}_{i=1}^N$ が与えられているとしよう．このとき，

$$\min_{\theta \in \mathbb{R}^n} \sum_{i=1}^N \|y_i - f(\theta, x_i)\|^2 \qquad (6.4)$$

を解くことで，真のパラメータ θ^* を得ることを考える．ここで $\|\bullet\|$ は Euclid ノルムである．このパラメータを求めることを，で f が θ に関して線形である場合を**線形最小二乗法（linear least squares）**単に**最小二乗法（least mean squares）**，非線形である場合を**非線形最小二乗法（non-linear least squares）**と呼ぶ（least の代わりに minimum が用いられることもある）．一般に最小二乗法は，適切な最適化の手法を使って解けばよい．しかし，以下で見るように，特別な場合は解を陽に書き下せたり，後からデータが追加されても逐次的に解を得られる場合があり，データ容量を抑えられるなどメリットがある．

注意 6.2.1 最小二乗法において誤差の確率解釈は必ずしも必要ではない．例えば，A 君は四条河原町に，B さんは吉田キャンパスに，C 君は御所にいるとき，それぞれが最短距離で落ち合う場所を求める問題も最小二乗法として解くことができるが，最適な落合場所からそれぞれの現在地が確率的に決まっているかどうかは考える必要もない．

注意 6.2.2 実際のデータ分析の現場では，関数 f が既知であるというのも強い仮定であることに注意．関数 f はデータの統計的性質や物理的考察から決めるか，とりあえず簡単なものから試していけばよい．深層ニューラルネットワークでは，関数を柔軟に表現できる基底の組み合わせを使うことで f を表現するし，最近では Gauss 過程回帰 [3] と呼ばれる手法も使われる．

6.2.1 回帰問題

まずは θ が f に対して線形である場合を扱おう．このとき，適当な行列値関数 $\varphi : \mathbb{R}^p \to \mathbb{R}^{m \times n}$ が

存在して，

$$f(\theta, x) = \varphi(x)\theta$$

となる．ここで，$\sum_{i=1}^{N} \varphi(x_i)^\top \varphi(x_i)$ が正則ならば，最適化問題 (6.4) の最適解は次で求められる．

$$\hat{\theta}_N = \left(\sum_{i=1}^{N} \varphi(x_i)^\top \varphi(x_i) \right)^{-1} \sum_{j=1}^{N} \varphi(x_j)^\top y_j. \tag{6.5}$$

この $\hat{\theta}_N$ を，**最小二乗誤差推定値**（**Minimum square error estimate**）あるいは**最小二乗推定値**という[1]．このデータをもらったときの処理の仕方（推定規則）のことを，**最小二乗誤差推定量**（**minimum square error estimator**）という．また，このとき観測誤差 $w_i, i = 1, \ldots, N$ の共分散行列の推定値は，

$$\hat{V}_N = \frac{1}{N-n} \sum_{i=1}^{N} \left(y_i - \varphi(x_i)\hat{\theta}_N \right) \left(y_i - \varphi(x_i)\hat{\theta}_N \right)^\top \tag{6.6}$$

で与えられる[2]．以降，表記の簡単のために，

$$\Phi_N := \left(\sum_{i=1}^{N} \varphi_i^\top \varphi_i \right)^{-1}, \quad \varphi_i := \varphi(x_i)$$

とおく．次の問題は時間があるときに確認されたい．

問題 1 $\sum_{i=1}^{N} \varphi_i^\top \varphi_i$ が正則行列であるとき，式 (6.5) が成り立つことを示せ．

式 (6.5) より推定値 $\hat{\theta}_N$ が得られたからといって，喜んでばかりいてはいけない．得られた推定値がどのくらいよいか，検討することも重要である．真のパラメータ θ と $\hat{\theta}_N$ の間には

$$\begin{aligned} \theta - \hat{\theta}_N =& \Phi_N \left(\Phi_N^{-1}\theta - \sum_{j=1}^{N} \varphi_j^\top y_j \right) \\ =& \Phi_N \sum_{j=1}^{N} \varphi_j^\top (\varphi_j \theta - y_j) \\ =& \Phi_N \sum_{j=1}^{N} \varphi_j^\top w_j \end{aligned}$$

の関係がある．入力のデータは既知の数であるので，確率的な変動は w_j のみに依存することに注意されたい．したがって，w_j が平均ゼロであることは既知なので，推定が**不偏的**（**unbiased**）であるという性質

$$\mathbb{E}[\theta - \hat{\theta}_N] = 0$$

が成り立つ．また，もし観測誤差の共分散行列 V が既知であるならば，推定誤差の共分散行列は

$$\mathbb{E}[(\theta - \hat{\theta}_N)(\theta - \hat{\theta}_N)^\top] = \Phi_N \sum_{i=1}^{N} \varphi_i^\top V \varphi_i \Phi_N \tag{6.7}$$

を得る．とくに $V = \sigma^2 I_n$ ならば，

$$\mathbb{E}[(\theta - \hat{\theta}_N)(\theta - \hat{\theta}_N)^\top] = \sigma^2 \Phi_N$$

となり，Φ_N は N に対して（正定値行列の意味で）単調減少なので，漸近的に推定誤差がゼロに近づくことが示唆される．観測誤差の推定値が分からない場合は，式 (6.7) 右辺の V を $\hat{V}_N$ に置き換えればよい．

回帰推定誤差も求めておこう．元の最小化問題 (6.4) のコスト関数を正規化するため，データ数 N で割ると，

$$R_N := \frac{1}{N} \sum_{i=1}^{N} \| y_i - \varphi_i \hat{\theta}_N \|^2 \tag{6.8}$$

とおくと，

$$\begin{aligned} R_N =& \frac{1}{N} \boldsymbol{y}^\top \left(I_{mN} - \boldsymbol{\varphi}(\boldsymbol{\varphi}^\top \boldsymbol{\varphi})^{-1} \boldsymbol{\varphi}^\top \right) \boldsymbol{y} && (6.9) \\ =& \frac{1}{N} \| \left(I_{mN} - \boldsymbol{\varphi}(\boldsymbol{\varphi}^\top \boldsymbol{\varphi})^{-1} \boldsymbol{\varphi}^\top \right) \boldsymbol{y} \|^2 && (6.10) \\ =& \frac{1}{N} \| \left(I_{mN} - \boldsymbol{\varphi}(\boldsymbol{\varphi}^\top \boldsymbol{\varphi})^{-1} \boldsymbol{\varphi}^\top \right) \boldsymbol{w} \|^2 && (6.11) \end{aligned}$$

[1] $\hat{\theta}_N$ を求める問題は，本質的には連立 1 次方程式の解を求める問題と同じになるため，逆行列の計算は Gauss の消去法などを利用して解けばよい．

[2] パラメータ数分をデータから引いて $N - n$ としているのは不偏推定値にするため．$N \gg n$ なら引かなくてもあまり変わらない．

となる．ただし，

$$\boldsymbol{y} := \begin{bmatrix} y_1^\top & \dots & y_N^\top \end{bmatrix}^\top \in \mathbb{R}^{mN},$$
$$\boldsymbol{w} := \begin{bmatrix} w_1^\top & \dots & w_N^\top \end{bmatrix}^\top \in \mathbb{R}^{mN},$$
$$\boldsymbol{\varphi} := \begin{bmatrix} \varphi_1^\top & \dots & \varphi_N^\top \end{bmatrix}^\top \in \mathbb{R}^{mN\times n}$$

とおいた．Φ_N の正則性は，$\boldsymbol{\varphi}$ が列フルランクであることを意味することに注意されたい．また，式 (6.9) から式 (6.10) への変形には $(I_{mN} - \boldsymbol{\varphi}(\boldsymbol{\varphi}^\top\boldsymbol{\varphi})^{-1}\boldsymbol{\varphi}^\top)$ が射影行列になることを用いた．式 (6.11) から，誤差の二乗平均を表す R_N は観測誤差と入力から定まる行列 $\boldsymbol{\varphi}$ で定まる．もし各ベクトルやデータの数が $mN = n$ を満たす場合，$\boldsymbol{\varphi}$ そのものも逆行列をもつため，$R_N = 0$ になる．R_N の最小値を達成していることになるが，これはよい推定と言えるだろうか？一方，データを増やすと $mN > n$ となり，$R_N > 0$ と一般にはなってしまうが，これは悪い推定値を得ていることになるのだろうか？

推定問題が「データが与えられたときの単なる最適化問題である」と言えないのは，推定問題の本来の目的が入出力関係を適切に定める関数 φ とパラメータ θ を決定することであり，最適化問題はそのための手段だからである．そのため，そもそも適切な問題設定になっていなければ，そうなるようにデータを増やすなどの工夫をしなければならない．上に挙げた例は，観測誤差に対してパラメータを合わせてしまった**過学習（over learning** あるいは**過適合（over fitting）**と呼ばれる現象が起こりうる例であり，一般的に歓迎するものではない．それでは，推定性能はどのように評価すればよいだろうか？θ が何かの分布に従って発生しており，雑音の分布も分かっているのならば，R_N の期待値，あるいは $\|\theta - \hat{\theta}_N\|^2$ の期待値を取って評価することができる．しかし，ここで考えている条件下では，確率分布が分からないため，（観測誤差を含む）観測値と答え合わせをするしかない．1 つのやり方が，**交差検証（cross validation）**である（和名よりもクロスバリデーションと呼ばれることが多い）．簡単なやり方は，データを学習用と検証用の 2 つに分け，学習用データでパラメータ推定を行い，検証用データで誤差を評価する方法である．検証用のデータを用いた誤差のことを**テスト誤差（testing error）**という．学習用のデータで得られた推定値が検証用のデータでも良い推定誤差をもつのならば，良い推定ができていると言えるだろう．学習用のデータはできるだけ多くとることは重要であるが，そのために検証用のデータがなくなるのは困るので，多くの場合はそれらの兼ね合いを考えて行う必要がある．学習データがどの程度あれば十分そうかを見積もるには，$\hat{\theta}_N$ がどのくらいのデータ数 N で収束しているかを確認すればよい．例えば，$y = 2x + w$, $w \sim \mathcal{N}(0, 4)$ の場合の推定値のデータ数に対する収束の様子を図 6.2 に示す．また，区間 $[0, 10]$ 内から一様乱数を用いて x_i を発生させデータセットを 10000 組用意し，学習用データ N を増やしたときの観測誤差 R_N の様子と，次で定義される学習用データを除いた残りの $M = 10000 - N$ 個のデータを使った誤差を図 6.3 に示す．

$$R_N^M := \frac{1}{M}\sum_{i=N+1}^{10000} \|y_i - \varphi_i\hat{\theta}_N\|^2,\ M > 0$$

これによると大体 $N = 200$ 程度から $\hat{\theta}_N$ はほとんど変化しなくなるので，この例においては学習用のデータは 200 くらいあればよさそうである．また，この例では $R_1 = 0$ となるが，$\hat{\theta}_1 \simeq 1.2$ なので真値から大きく異なる．N を増やしていくと学習誤差はどこか一定の値に収束していく様子も見られる．また，テスト誤差もだいたい学習誤差と同じくらい値に収束していく様子も見られる．データが十分ある場合はテスト誤差と学習誤差は変わらないが，データが十分あるかどうかわからない場合は両方を併用するとよい．

なお，上で用いた R_N や R_N^M の大きさはどの単位で測定しているかによって決まるので（例えば m か cm では 100 倍違うので），得られたパラメータがどれほどよくデータを説明するかの指標として用いるには向いてない．観測誤差の確率分布が不明であるならば，**決定変数（coefficient of determination, determination coefficient）**

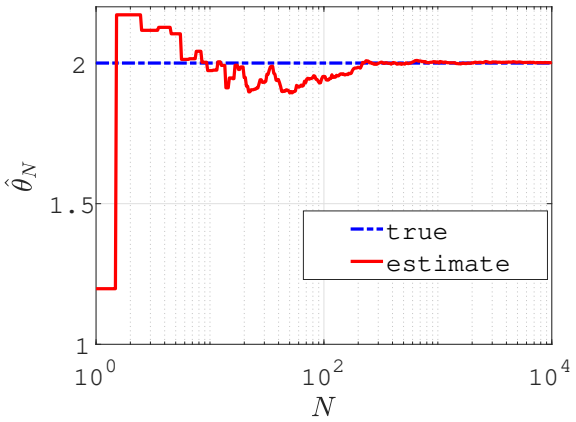


図 6.2: $\hat{\theta}_N$ の収束の様子

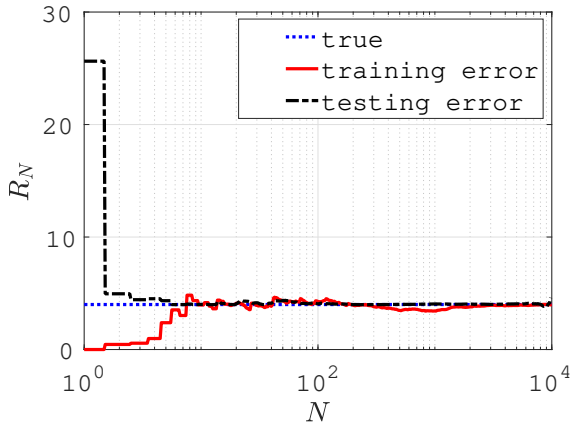


図 6.3: N に対する R_N と R_N^M

と呼ばれる量が用いられる．

$$C := \frac{\sum_{i=1}^{N} \|\varphi_i \hat{\theta}_N - \bar{y}\|^2}{\sum_{i=1}^{N} \|y_i - \bar{y}\|^2} \tag{6.12}$$

ここで $\bar{y} := \frac{1}{N}\sum_{i=1}^{N} y_i$ とした．最小二乗推定値 $\hat{\theta}_N$ の決定変数は $[0,1]$ の範囲に値を取り，その値が 1 に近いほどよい推定であるとみなされる．逆に 0 に近い場合は得られたパラメータ $\hat{\theta}_N$ が出力 y の推定に役に立たないことを意味する．

なお，観測誤差が正規分布に従うのであれば，$\hat{\theta}_N$ の誤差の評価も様々な統計なツールで分析することができる [2]．以下の 2 つの問題を通して，観測誤差の大きさが異なる場合の最小二乗法を比較検討されたい．

課題 8 (重回帰問題) 各入力 $x_i \in \mathbb{R}^2$, $i = 1, 2, \ldots, 10000$ に対し，次の式で観測データ $y_i \in \mathbb{R}$ が生成されているとする．

$$y_i = x_i^\top \theta + w_i \tag{6.13}$$

ここで観測誤差は $\mathcal{N}(0,1)$ に従って発生している（ただし，平均の情報のみ知っているとし，分散も分布も知らないものとする）．このとき，次の問いに答えよ．

1. θ の最小二乗誤差推定量を，与えられた全てのデータ $\{(x_i, y_i)\}_{i=1}^{10000}$ を使って求めよ．また，そのときの推定誤差共分散行列を求めよ．
2. 図 6.2 のように用いるデータ $\{(x_i, y_i)\}_{i=1}^{N}$ を増やしたとき，$\hat{\theta}_N$ の各要素が収束していくことを片対数グラフでプロットして確認せよ．ただし，$N = 2, 4, 8, \ldots, 2^{13} = 8192$ のときのみでよい．
3. 全てのデータを用いたときの決定変数を求めよ．

課題 9 (多項式回帰) 各入力 $x_i \in \mathbb{R}$, $i = 1, 2, \ldots, 10000$ に対し，次の式で観測データ $y_i \in \mathbb{R}$ が生成されているとする．

$$y_i = \varphi(x_i)\theta + w_i, \ \varphi(x) := \begin{bmatrix} 1 & x & x^2 & x^3 \end{bmatrix} \tag{6.14}$$

ここで観測誤差は $\mathcal{N}(0,\,9)$ に従って発生している（ただし，平均の情報のみ知っているとし，分散も分布も知らないものとする）．このとき，次の問いに答えよ．

1. θ の最小二乗誤差推定量を，与えられた全てのデータ $\{(x_i, y_i)\}_{i=1}^{10000}$ を使って求めよ．また，そのときの推定誤差共分散行列を求めよ．
2. 図 6.2 のように用いるデータ $\{(x_i, y_i)\}_{i=1}^{N}$ を増やしたとき，$\hat{\theta}_N$ の各要素が収束していくことを片対数グラフでプロットして確認せよ．ただし，$N = 4, 8, \ldots, 2^{13} = 8192$ のときのみでよい．
3. 全てのデータを用いたときの決定変数を求めよ．

また，推定値が収束しない例もあることを次の問題を通して確認されたい．このような問題に最小二乗法を適用することは意味がなく，別の推定手法が必要になる．

課題 10 (分散∞の観測誤差の場合) ここで扱うデータの観測誤差は，各入力 $x_i \in \mathbb{R}^2$, $i = 1, 2, \ldots, 10000$ に対し，次の式でデータ $y_i \in \mathbb{R}$ が生成されているとする．

$$y_i = x_i^\top \theta + w_i \tag{6.15}$$

ただし，観測誤差は標準 Cauchy 分布に従うとする．このとき，次の問いに答えよ．

1. 与えられた全データ $\{(x_i, y_i)\}_{i=1}^{10000}$ を用いて θ の最小二乗誤差推定量を式 (6.5) から求めよ．
2. 図 6.2 のように用いるデータ $\{(x_i, y_i)\}_{i=1}^{N}$ を増やしたとき，$\hat{\theta}_N$ の各要素が収束していかないことを片対数グラフでプロットして確認せよ．ただし，$N = 2, 4, 8, \ldots, 2^{13} = 8192$ のときのみでよい．

データが求めたい関数を表現するのに十分であるかどうかを確認することも重要である．

課題 11 課題 9 と同じ $\varphi(x)$，θ および観測誤差の分布 $\mathcal{N}(0, 9)$ を考える．また，$x_i \in [0, 1]$, $i = 1, \ldots, 10000$ のときに，データ $\{(x_i, y_i)\}_{i=1}^{10000}$ が与えられているものとする．このとき，$\hat{\theta}_{N=10000}$ を求め，課題 9.1 の $N = 10000$ の結果と比較せよ．また，どのようにデータを取るべきか考察せよ．

注意 6.2.3 データの取り方が悪い場合やデータの数が少ない場合は，$\sum_{i=1}^{N} \varphi_i^\top \varphi_i$ が正則にならない，あるいはそれに近い状況（条件数が悪い）になることがあり，逆行列が求められないことがある．正則にならない正方行列は線形従属な行または列をもつため，統計学では**多重共線性**（**multicolinear**）と呼ばれる．これは $\hat{\theta}_N$ が求められないことを意味するのではなく，一意に求められないことを意味するので，そのような場合，下記のように正則化と呼ばれる項 $g(\theta)$ を追加して解くことがよく行われる．

$$\min_{\theta \in \mathbb{R}^n} \sum_{i=1}^{N} \|y_i - \varphi_i \theta\|^2 + \lambda g(\theta)$$

ここで λ は正の実数である．この正則化は，単に解を得るための手段としてのみでなく，過学習を防ぐ役割や求めたいパラメータの特徴を反映させるために付加することもある．例えば，$g(\theta) = \|\theta\|^2$ とすれば，リッジ回帰と呼ばれる種類の問題になる．また，$g(\theta) = \|\theta\|_{\ell^1}$ と ℓ^1 ノルムを用いれば，Lasso と呼ばれるスパースな解を得るための問題になる [3, 7]．リッジ回帰は逐次最小二乗法の項で用いる．

注意 6.2.4 観測誤差が Cauchy 分布の場合を課題 10 に出したが，最小二乗法の弱点は外れ値（outlier）と呼ばれる大きな誤差に弱い（大きな誤差に合わせるようにパラメータが選ばれてしまう）点である．このような外れ値がデータに混入する場合，最小二乗法を適用する前に，事前にデータから外れ値を除いた方がよい．修正トンプソン法など，データから外れ値を自動的に判断して除去する方法はいくつか提案されているので，外れ値の影響が疑われる場合はデータの前処理を行うとよい．

6.2.2 重み付き最小二乗法

次に，最小二乗法における観測誤差に関する仮定を再考する．具体的には，次の 3 つを考える

- 観測誤差の同一分布性
- 観測誤差の独立性
- 観測誤差の共分散行列の情報の有無

これまでは観測誤差は独立同一分布であることや，共分散行列の値が未知であることを仮定していたが，経年劣化したセンサと新品のセンサで同じ対象を計測した場合，その観測誤差は独立ではあっても同じ統計量をもつとはいえない場合も出てくるだろう．データ数 N_1 までは同じ共分散行列 V_1 だったものが，センサを変えたので新たに取れるデータでは共分散行列が $V_2 = 0.1 \times V_1$ になっているかもしれない．仮にどちらかの分散が小さくなることが分かっているのならば（あるいは推定されているのならば），得られるデータの質は同等とはみなせないだろう．このように，データ $\{(x_i, y_i)\}_{i=1}^N$ を得たとき，それぞれのデータの信用度を考えて最小二乗法を解きたい場合もある．このとき，一般には重み行列 $Q_i \geq 0$, $i = 1, 2, \ldots, N$ を用いた次の**重み付き最小二乗法（weighted least squares）**

$$\min_{\theta \in \mathbb{R}^n} \sum_{i=1}^{N} (y_i - \varphi_i \theta)^\top Q_i (y_i - \varphi_i \theta) \tag{6.16}$$

を考えればよい．もちろん，全ての重みがゼロ行列の場合は意味がないので，そのような例は本稿では考えない．この場合，$\sum_{i=1}^N \varphi_i^\top Q_i \varphi_i$ が正則ならば，最適な推定値は

$$\hat{\theta}_N = \left(\sum_{i=1}^{N} \varphi_i^\top Q_i \varphi_i \right)^{-1} \sum_{j=1}^{N} \varphi_j^\top Q_j y_j \tag{6.17}$$

となる．重み付き最小二乗法とは通常，Q_i は対角行列とすることが多いが，本項では重み行列の場合も重み付き最小二乗法と呼ぶ．

観測誤差の共分散行列 V が既知である場合，推定誤差共分散行列 (6.7) は最小の推定誤差共分散行列ではないことが知られている（Gauss–Markov の定理，[2, 4]）．V は正定値対称行列であるので，$V = W^2$ を満たす正定値行列 W が一意に存在する（V の平方根行列）[3]．これを用いると，式 (6.2) は

$$W^{-1} y = W^{-1} \varphi(x) \theta + \underbrace{W^{-1} w}_{=:w'}, \tag{6.18}$$

と書き直すことができる．ここで新たな観測誤差 w_i' は単位行列を分散としてもつ観測誤差である．この解は，

$$\hat{\theta}_N = \left(\sum_{i=1}^{N} \varphi_i^\top W^{-2} \varphi_i \right)^{-1} \sum_{j=1}^{N} \varphi_j^\top W^{-2} y_j \tag{6.19}$$

となるので，$Q_i = aW^{-2} = aV^{-1}$ と選んだものに相当する（$a \in \mathbb{R}$ は正定数）．この観測誤差の統計量を用いて推定値 $\hat{\theta}_N$ を得る上記の手法は，**最良線形不偏推定量（best linear unbiased estimator, BLUE）**と呼ばれる推定量と一致する．「最良」の名前の通り，これは最も推定誤差および訓練誤差の分散を最小にする推定規則である．正しい観測誤差の情報を持っていれば，素朴な定式化から得られる推定規則よりもいい結果が得られることを意味する．

課題 12 $x_i \in \mathbb{R}$, $i = 1, \ldots, 1000$ に対し，観測値の次元が 2 次元となる場合を考える．

$$y_i = \varphi(x_i)\theta + w_i.$$

ここで w_i は $\mathcal{N}(0, V)$ の独立同一分布に従うとし，

$$\varphi = \begin{bmatrix} 1 & x \\ 1 & x^2 \end{bmatrix}, \quad V = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$$

とする．このとき，式 (6.5) と式 (6.17) をそれぞれ用いて推定値を求め，それぞれの推定誤差共分散を求めよ．ただし，$Q_i = V^{-1}$ とする．また，$\hat{\theta}_N$ の各要素の収束の仕方をプロットせよ．

[3] 一般的に，正則行列 U で $V = UU^\top$ となるものも無数に存在する．

課題 13 先ほどの課題と同様，$x_i \in \mathbb{R},\ i = 1,\dots,1000$ に対し，観測値の次元が 2 次元となる場合を考える．

$$y_i = \varphi(x_i)\theta + w_i.$$

ここで w_i は，$i = 1,\dots,500$ は $\mathcal{N}(0, V_1)$ の独立同一分布に，$i = 501,\dots,1000$ は $\mathcal{N}(0, V_2)$ の独立同一分布に従うとし，

$$\varphi(x) = \begin{bmatrix} 1 & x \\ 1 & x^2 \end{bmatrix},\ V_1 = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}, V_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

とする．このとき，式 (6.5) と式 (6.17) のそれぞれで推定値を求め，それぞれの収束の仕方を片対数グラフでプロットせよ．

より一般的には，観測誤差の独立性が仮定できない場合も考えられる．観測誤差が独立でない場合，次のように相関をもつ．

$$V_{i,j} = \mathbb{E}[w_i w_j^\top]$$

$\boldsymbol{V} := \mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top]$ とおくと，重み付き最小二乗法の解法を拡張して，次で得られる推定値が BLUE であることが分かる．

$$\hat{\theta}_N = (\boldsymbol{\varphi}^\top \boldsymbol{V}^{-1} \boldsymbol{\varphi})^{-1} \boldsymbol{\varphi}^\top \boldsymbol{V}^{-1} \boldsymbol{y}^\top \tag{6.20}$$

ここで $\boldsymbol{V}$ はデータ数 N によってはかなり大きなサイズの行列になりえるので，仮にこの行列が分かっていたとしても，この逆行列を求めるのは時間がかかってしまう．そのため，計算時間がかかりすぎる場合や条件数が悪い場合には，何かしら工夫して逆行列を計算する必要がある．

6.3 推定値の合成と逐次最小二乗法

式 (6.5) のように得られている全てのデータをまとめて処理することは，**バッチ処理**と呼ばれる．とくに新しくデータが追加されなければバッジ処理だけ知っておけば十分だが，データ処理の現場では入力と観測値のデータが追加されることはよくあるので，そのたびに最適化問題を解き直すことは煩わしい．とくに計測機からの情報が送られ続ける場合，データを保存するメモリの総量も膨大になるので，別の手段を用いる必要がある．幸いなことに，最小二乗推定値は出力に関して線形な解になる．このことを利用して，ここでは新たにデータが得られたときの処理の仕方について学ぶ．

6.3.1 異なるデータセットからの推定値の合成

データ $\{(x_i, y_i)\}_{i=1}^N$ を得たとき，$f(\theta, x) = \varphi(x)\theta$ の場合の式 (6.4) の問題を解くと，式 (6.5) で解が得られることがわかったが，実際のデータ処理では，別のデータセット $\{(x'_j, y'_j)\}_{j=1}^M$ を用いた推定値 $\hat{\theta}'_M$ が独立に存在する場合もある．それらは適切な仮定の下で最小二乗法を用いて以下のように推定値を得ることができる．

$$\hat{\theta}'_M = \underbrace{\left(\sum_{i=1}^M \varphi(x'_i)^\top \varphi(x'_i)\right)^{-1}}_{=:\Phi'_M} \sum_{j=1}^M \varphi(x'_j)^\top y'_j. \tag{6.21}$$

2 組のデータセットからそれぞれの推定値が得られているが，より多くのデータセットから推定する方が推定精度は改善することが期待される．このとき，データを合わせて改めて解き直してもよいが，2 つの推定値を用いて，推定値を更新することができる．便宜的に $\{(x'_j, y'_j)\}_{j=1}^M$ のラベルをつけ直して $\{(x_i, y_i)\}_{i=N+1}^{N+M} = \{(x'_j, y'_j)\}_{j=1}^M$ とす

ると，

$$
\begin{aligned}
&\hat{\theta}_{N+M} \\
&= \left(\Phi_N^{-1} + \Phi_M'^{-1}\right)^{-1} \sum_{i=1}^{N+M} \varphi_i^\top y_i \\
&= \left(\Phi_N^{-1} + \Phi_M'^{-1}\right)^{-1} \sum_{i=1}^{N} \varphi_i^\top y_i \\
&\quad + \left(\Phi_N^{-1} + \Phi_M'^{-1}\right)^{-1} \sum_{i=N+1}^{N+M} \varphi_i^\top y_i \\
&= \left(\Phi_N^{-1} + \Phi_M'^{-1}\right)^{-1} \Phi_N^{-1}\Phi_N \sum_{i=1}^{N} \varphi_i^\top y_i \\
&\quad + \left(\Phi_N^{-1} + \Phi_M'^{-1}\right)^{-1} \Phi_M'^{-1}\Phi_M' \sum_{i=N+1}^{N+M} \varphi_i^\top y_i \\
&= \left(\Phi_N^{-1} + \Phi_M'^{-1}\right)^{-1} \left(\Phi_N^{-1}\hat{\theta}_N + \Phi_M'^{-1}\hat{\theta}'_M\right)
\end{aligned}
$$

となる．したがって，新たなデータを得ることが想定されるならば，推定値 $\hat{\theta}$ のみでなく Φ_N も保存しておけば，元のデータを保存しておく必要はない[4]．

課題 14 $x_i \in \mathbb{R}$, $i = 1, \ldots, 10000$ に対し，次の式で観測データ $y_i \in \mathbb{R}$ が生成されているとする．

$$
y_i = \varphi(x_i)\theta + w_i \tag{6.22}
$$

ここで

$$
\varphi(x) := \begin{bmatrix} 1 & \exp\left(-\frac{(x_i-1)^2}{2}\right) & \exp\left(-(x_i+1)^2\right) \end{bmatrix}
$$

であり，w_i は平均 0，分散有限の独立同一分布から生成されているものとする．10000 組のデータの組 $\{(x_i, y_i)\}_{i=1}^{10000}$ のうち，最初の 6000 組と残りの 4000 組のデータそれぞれで $\theta \in \mathbb{R}^3$ の推定値を求め，それらから推定値を合成せよ．また，全データを使った場合の推定値と比較し，一致することを確かめよ．

観測誤差の大きさが異なる場合，重み付きの最小二乗法を使った方が推定結果がよくなることが期待される．データ数がそれぞれ N_1, N_2 のデータセットが 2 組与えられているとき，重み行列 Q_1 と Q_2 を使ったときのそれぞれのデータセットにおける推定値を $\hat{\theta}_{N_1}$，$\hat{\theta}_{N_2}$ とおくと，

$$
\begin{aligned}
\hat{\theta}_{N_1+N_2} = &\left(\Phi_{Q_1,N_1}^{-1} + \Phi_{Q_2,N_2}^{-1}\right)^{-1} \\
&\times \left(\Phi_{Q_1,N_1}^{-1}\hat{\theta}_{N_1} + \Phi_{Q_2,N_2}^{-1}\hat{\theta}_{N_2}\right)
\end{aligned} \tag{6.23}
$$

となる．ここで

$$
\Phi_{Q,N} := \left(\sum_{i=1}^{N} \varphi_i^\top Q \varphi_i\right)^{-1}
$$

とした．次の問題もそれほど難しくないので，時間があるときに考えてもらいたい．

問題 2 式 (6.23) が成り立つことを示せ．

課題 15 $x_i \in \mathbb{R}$, $i = 1, \ldots, 10000$ に対し，次の式でデータ $y_i \in \mathbb{R}$ が生成されているとする．

$$
y_i = \varphi(x_i)\theta + w_i \tag{6.24}
$$

ここで w_i は最初の 6000 組と残りの 4000 組のデータで，平均ゼロの異なる独立な分布に従うものとする（$i = 1, \ldots, 6000$ では独立同一分布，$i = 6001, \ldots, 10000$ では異なる独立同一分布）．

$$
\varphi(x) := \begin{bmatrix} 1 & \exp\left(-\frac{(x_i-1)^2}{2}\right) & \exp\left(-(x_i+1)^2\right) \end{bmatrix}
$$

である．10000 組のデータの組 $\{(x_i, y_i)\}_{i=1}^{10000}$ のうち，最初の 6000 組と残りの 4000 組のデータそれぞれで $\theta \in \mathbb{R}^3$ の推定値と偶然誤差 w_i の分散の推定値を求め，推定された分散を用いて推定値を合成せよ．また，全データを使い，式 (6.5) を用いて得た $\hat{\theta}_{10000}$ と，どちらが優れているか比較考察せよ（比較の仕方を考えること）．

6.3.2 逐次最小二乗法

これまでの考え方を応用して，最小二乗法を実時間処理で解くことを考えよう．N 個のデータから推定された $\hat{\theta}_N$ と新たなデータ (x_{N+1}, y_{N+1})

[4] あくまで最小二乗法で行う場合の結果であり，他の評価基準を用いる場合は全データを用いる必要が生じることがほとんどである．

を用いて再帰的に $\hat{\theta}_{N+1}$ を求めることを考えよう．最小二乗誤差推定 (6.5) より，次のように式変形できる．

$$\begin{aligned}
&\hat{\theta}_{N+1}\\
=&\Phi_{N+1}\sum_{j=1}^{N+1}\varphi_j^\top y_j\\
=&\left(\Phi_N^{-1}+\varphi_{N+1}^\top\varphi_{N+1}\right)^{-1}\\
&\times\left(\varphi_{N+1}^\top y_{N+1}+\sum_{j=1}^{N}\varphi_j^\top y_j\right)\\
=&\Big\{\Phi_N-\Phi_N\varphi_{N+1}^\top\\
&\times\left(I_m+\varphi_{N+1}\Phi_N\varphi_{N+1}^\top\right)^{-1}\varphi_{N+1}\Phi_N\Big\}\\
&\times\left(\varphi_{N+1}^\top y_{N+1}+\sum_{j=1}^{N}\varphi_j^\top y_j\right)\\
=&\hat{\theta}_N+K_{N+1}\left(y_{N+1}-\varphi_{N+1}\hat{\theta}_N\right).
\end{aligned}$$

ただし

$$K_{N+1}:=\Phi_N\varphi_{N+1}^\top\left(I_m+\varphi_{N+1}\Phi_N\varphi_{N+1}^\top\right)^{-1}.$$

ここで式変形に逆行列補題[5]を用いて，

$$\begin{aligned}
&\Phi_{N+1}\\
=&\Phi_N-\Phi_N\varphi_{N+1}^\top\\
&\times\left(I_m+\varphi_{N+1}\Phi_N\varphi_{N+1}^\top\right)^{-1}\varphi_{N+1}\Phi_N
\end{aligned}$$

を得た．したがって，新たなデータ (x_{N+1},y_{N+1}) が得られるたびに，新たな推定値 $\hat{\theta}_{N+1}$ はそれまでのデータによる推定値 $\hat{\theta}_N$ と Φ_N を用いることで更新できるので，バッジ処理の必要がない．オンラインで行う場合，重要なのは Φ_0 の選び方である．Φ_N の定義から，Φ_0 は定義できない．そこで Φ_N そのものを用いずに，小さな正の実数 ϵ を用いて $\tilde{\Phi}_0=\frac{1}{\epsilon}I_n$ とし，次のように更新すればよい．

$$\tilde{\theta}_{N+1}=\tilde{\theta}_N+\tilde{K}_{N+1}\left(y_{N+1}-\varphi_{N+1}\tilde{\theta}_N\right), \tag{6.25}$$

$$\tilde{K}_{N+1}=\tilde{\Phi}_N\varphi_{N+1}^\top\left(I_m+\varphi_{N+1}\tilde{\Phi}_N\varphi_{N+1}^\top\right)^{-1} \tag{6.26}$$

$$\tilde{\Phi}_{N+1}=\tilde{\Phi}_N-\tilde{K}_{N+1}\varphi_{N+1}\tilde{\Phi}_N \tag{6.27}$$

ただし，

$$\tilde{\theta}_0=0,\quad \tilde{\Phi}_0=\frac{1}{\epsilon}I_n$$

とした．

ここでこの $\tilde{\Phi}_N$ と Φ_N の関係は，Φ_N が正則になる N に対して，

$$\tilde{\Phi}_N^{-1}=\Phi_N^{-1}+\epsilon I_n=\sum_{i=1}^{N}\varphi_i^\top\varphi_i+\epsilon I_n$$

であることに注意されたい．Φ_N^{-1} はデータを得るたびに正定値対称行列の意味で大きくなるため，$\tilde{\Phi}_N$ と Φ_N は Taylor 展開すると，

$$\tilde{\Phi}_N=\Phi_N-\epsilon\Phi_N^2+O(\epsilon^2)$$

であり，適切なデータ数 N が増えるにつれて Φ_N は正定値対称行列の意味で小さくなるので，ϵ の 1 次の項がゼロに近づくことが期待される．実際に ϵ の 1 次以上の項は N が増えると小さくなる項である．以下では，添字に時間の順序関係がある場合には k を用いる．

課題 16 (システム同定) 次のバネ・マス・ダンパ系を表す直線上の運動方程式を考える．

$$M\frac{d^2}{dt^2}y(t)+D\frac{d}{dt}y(t)+Ky(t)=F(t)$$

ここで M,K,D は正定数で，$F(t)$ は外力である．$y(t)$ は観測でき，$F(t)$ はこちらで設計できるものとする．このとき，得られるデータから (M,K,D) を決定したい．微小な δt では

$$\begin{aligned}
\frac{d}{dt}y(t)&\simeq\frac{y((k+1)\delta t)-y(k\delta t)}{\delta t},\\
\frac{d^2}{dt^2}y(t)&\simeq\frac{y((k+2)\delta t)-2y((k+1)\delta t)+y(k\delta t)}{\delta t^2}
\end{aligned}$$

[5] $A\in\mathbb{R}^{n\times n}$ を正則な行列とするとき，$(A+BC)^{-1}=A^{-1}-A^{-1}B(I+CA^{-1}B)^{-1}CA^{-1}$．ただし，$B$ と C は $BC\in\mathbb{R}^{n\times n}$ となる任意のサイズの行列．

と近似できることを利用し，$y_k := y(k\delta t)$，$F_k := F(k\delta t)$ とすると，

$$\begin{aligned} y_k = &\left(2 - \frac{D}{M}\delta t\right) y_{k-1} \\ &+ \left(1 + \frac{D}{M}\delta t - \frac{K}{M}\delta t^2\right) y_{k-2} \\ &+ \frac{\delta t^2}{M} F_{k-2} + w_k \end{aligned}$$

を得る．ここで w_k は近似の際に生じる誤差である．したがって，問題は

$$y_k = x_k^\top \theta + w_k,\ x_k := \begin{bmatrix} y_{k-1} \\ y_{k-2} \\ F_{k-2} \end{bmatrix}$$

の θ を求める問題にすることができる．パラメータの真値を $(M, D, K) = (2, 1, 3)$ とし，$\delta t = 0.01$，w_k は $[-1, 1]$ の一様分布に従う各時刻で独立な確率変数であるとして，以下の問いに答えよ．

1. $y_{-2} = 0$, $y_{-1} = 0$ とし，$F_k = 1$, $k = -2, -1, 0, 1, 2, \ldots$ として一定の外力を加えたとき，パラメータ θ を逐次最小二乗法で求めよ．ただし，パラメータの初期値は 0 とし，$k = 10000$ までよい．また正則化項は工夫せよ．

2. $y_{-2} = 0$, $y_{-1} = 0$ とし，$F_k = \sin(\pi k/5)$, $k = -2, -1, 0, 1, 2, \ldots$ として一定の外力を加えたとき，パラメータ θ を逐次最小二乗法で求めよ．ただし，パラメータの初期値は 0 とし，$k = 10000$ までよい．また正則化項は工夫せよ．

3. $y_0 = 0$, $y_1 = 0$ とする．このとき，どのような外力を加えればパラメータの推定がうまくいくか考え実装し，逐次最小二乗法で求めよ．ただし，パラメータの初期値は 0 とし，$k = 10000$ までよい．

重み付き逐次最小二乗法の更新アルゴリズムも，逐次最小二乗法と同様の計算で求められる．とくに重み行列が単位行列の定数倍（$Q_i = \rho_i I$）のとき，次のように計算される．

$$\begin{aligned} \hat{\theta}_{N+1} =& \hat{\theta}_N + K_{N+1}(y_{N+1} - \varphi_{N+1}\hat{\theta}_N), \\ K_{N+1} =& \rho_{N+1}\Phi_N\varphi_{N+1}^\top \\ &\times \left(I_m + \rho_{N+1}\varphi_{N+1}\Phi_N\varphi_{N+1}^\top\right)^{-1}, \\ \Phi_{N+1} =& \Phi_N - K_{N+1}\varphi_{N+1}\Phi_N. \end{aligned}$$

とくに時系列データの場合には，古いデータの影響を小さく，新しいデータの影響を大きく使いたい場面が多い．そこで N 組のデータがあるときに $\rho_i = \gamma^{N-i}$，$\gamma \in (0, 1)$ とおくと，その推定値は

$$\hat{\theta}_{\gamma,N+1} = \left(\sum_{i=1}^{N} \gamma^{N-i}\varphi_i^\top\varphi_i\right)^{-1} \sum_{j=1}^{N} \gamma^{N-j}\varphi_j^\top y_j$$

となる．同様に $N+1$ 組のデータでは

$$\begin{aligned} \hat{\theta}_{\gamma,N+1} = &\left(\varphi_{N+1}^\top\varphi_{N+1} + \gamma\sum_{i=1}^{N}\gamma^{N-i}\varphi_i^\top\varphi_i\right)^{-1} \\ &\times \left(\varphi_{N+1}^\top y_{N+1} + \gamma\sum_{j=1}^{N}\gamma^{N-j}\varphi_j^\top y_j\right) \end{aligned}$$

となるので，これを整理すると次の更新アルゴリズムを得る．

$$\hat{\theta}_{\gamma,N+1} = \hat{\theta}_{\gamma,N} + K_{\gamma,N+1}(y_{N+1} - \varphi_{N+1}\hat{\theta}_{\gamma,N}) \tag{6.28}$$

$$K_{\gamma,N+1} = \Phi_{\gamma,N}\varphi_{N+1}^\top \left(\gamma I_m + \varphi_{N+1}\Phi_{\gamma,N}\varphi_{N+1}^\top\right)^{-1} \tag{6.29}$$

$$\Phi_{\gamma,N+1} = \frac{1}{\gamma}\Phi_{\gamma,N} - \frac{1}{\gamma}K_{\gamma,N+1}\varphi_{N+1}\Phi_{\gamma,N} \tag{6.30}$$

この γ を**忘却係数（forgetting factor）**と呼び，古いデータに指数的な重みをつけ，新しいデータの影響をパラメータ推定に強く反映させる効果を与える．

課題 17 (非定常時系列データの推定) 次の正弦波でゆっくりと変動する信号を考える．

$$y_k = \sin(0.0001k) + w_k,\ k = 1, \ldots, 10000.$$

ただし，w_k は -1 と $+1$ を確率 $\frac{1}{2}$ で出す Bernoulli 過程であるとし，$\theta_k := \sin(0.0001k)$ は未知とす

る．このとき，$\gamma = 0.99$ の忘却係数付きの逐次最小二乗法で θ_k を推定し，各時刻の $\hat{\theta}_{\gamma,k}$ をプロットせよ．

6.4 Kalman フィルタと Kalman スムーザ

重み付け最小二乗法の項で述べたとおり，確率分布を知っている場合（統計的な性質を知っている場合），最小二乗誤差推定よりも**最小平均二乗誤差推定（minimum mean square error estimation）**と呼ばれる手法を使った方が未知のデータに対する精度が上がる[6]．議論を簡単にするため，N 個のデータ $\mathcal{D}_N := \{(x_i, y_i)\}_{i=1}^N$ の関数として，入力 x を入れたときの出力の推定値を $\hat{y}(\mathcal{D}_N, x)$ と表す．また，入力 x のときの出力を $y(x)$ で表す．このとき，入力 x に対する最小平均二乗誤差推定は，次の「関数 $\hat{y}$」に関する最適化問題の解になる（期待値は存在するとする）．

$$\min_{\hat{y}} \mathbb{E}_P \left[|y(x) - \hat{y}(\mathcal{D}_N, x)|^2 \right] \tag{6.31}$$

期待値を用いることで未知のデータに対しても二乗誤差の統計量を計算することができ，それを最小にするように推定規則を作成することができる．データとして用いている $\{y_i\}_{i=1}^N$ も確率変数であることに注意されたい．また，$\hat{y}$ は x にも依存する関数であるが，以下では $x_i = i$ と時間を表すパラメータだと思うことにする[7]．この問題 (6.31) の解は，条件付き期待値になることが知られている．条件付き期待値の厳密な定義や性質は「確率と統計」の授業や統計学の文献で勉強していただきたい．とくに $y(x) = \varphi(x)\theta + w$ と表される場合，最適化問題 (6.31) の解は θ の推定値を求める BLUE に等しい．そこで，重み行列を使った逐次最小二乗法の考え方を応用してみよう．とくにここでは線形なダイナミカルシステムの**潜在変数（hidden variable）**の推定問題を考える．

6.4.1 Kalman フィルタ

次の線形差分方程式を考える．

$$\theta_k = a_k \theta_{k-1} + v_k, \tag{6.32}$$

$$y_k = c_k \theta_k + w_k,\ k = 1, 2, \dots \tag{6.33}$$

ここで $a_k, c_k \in \mathbb{R}$ であり，$v_k, w_k \in \mathbb{R}$ は互いに独立な平均ゼロ，分散がそれぞれ σ_v^2, σ_w^2 の Gauss 分布に従って発生し，さらに異なる時刻で独立である（白色性をもつ）とする[8]．また，$c_k \neq 0$, $k = 1, \dots,$ とする．θ_0 は平均 $\bar{\theta}$, 分散 P を持つ分布に従って発生するとする（Gauss 性は不要）．ここで v_k や w_k および初期値 θ_0 は観測できないが，それらの “分布”の情報は既知であり，a_k, c_k も既知であるとしよう．観測値 y_k もセンサから各時刻で得られるものとするが，θ_k は潜在変数で直接知ることはできないとする．この節で扱う内容は，各時刻 k までの観測値 $\{y_\ell\}_{\ell=1}^k$ を用いて θ_k を推定することである．これまで扱ってきた問題は推定すべきパラメータが確定的だったが，ここでは θ_k も確率変数になっていることに注意されたい．

時刻 k までのデータによる推定値を $\hat{\theta}_k \in \mathbb{R}$ で表すとする．式 (6.25) の逐次最小二乗法では，1 ステップ前の推定値に，新たなデータによる修正項が加えられる．式 (6.32) と (6.33) より，

$$y_k = c_k(a_k \theta_{k-1} + v_k) + w_k$$

となるため，前の時刻の推定値 $\hat{\theta}_{k-1}$ を a_k 倍したものに新たなデータから修正する項を考え，次の規則で推定値を更新することを考える．

$$\begin{aligned}\hat{\theta}_k &= a_k \hat{\theta}_{k-1} + F_k(y_k - c_k a_k \hat{\theta}_{k-1}) \\ &= a_k \hat{\theta}_{k-1} + a_k c_k F_k(\theta_{k-1} - \hat{\theta}_{k-1}) \\ &\quad + F_k(w_k + c_k v_k)\end{aligned}$$

ここで $F_k \in \mathbb{R}$ は自由に設計できるパラメータである．このとき，推定誤差 $\theta_k - \hat{\theta}_k$ を最小二乗平

[6] least mean square error (LMSE) と minimum mean square error (MMSE) は，意味するものが違うので注意．LMSE は二乗誤差の「標本平均」に対する推定を意味し，MMSE は二乗誤差の「期待値」に対する推定を意味することが多いようである．

[7] そうでない場合は確率場の推定になり，Gauss 過程回帰などで用いられる．その場合，x の依存性を消すために x に関する積分や $\sup_x$ などを期待値の内側か外側につける．

[8] Gauss 性は本来は不要だが議論を簡単にするため．

均の意味で最小化する推定器を設計したい．推定誤差分散を $V_k := \mathbb{E}[(\theta_k - \hat{\theta}_k)^2]$ とすると，

$$\begin{aligned} V_k =& a_k^2(1-c_kF_k)^2V_{k-1} \\ &+ (1-c_kF_k)^2\sigma_v^2 + \sigma_w^2F_k^2 \\ =& (a_k^2c_k^2V_{k-1} + c_k^2\sigma_v^2 + \sigma_w^2) \\ &\times \left(F_k - \frac{a_k^2c_kV_{k-1} + c_k\sigma_v^2}{a_k^2c_k^2V_{k-1} + c_k^2\sigma_v^2 + \sigma_w^2}\right)^2 \\ &+ a_k^2V_{k-1} + \sigma_v^2 - \frac{(a_k^2c_kV_{k-1} + c_k\sigma_v^2)^2}{a_k^2c_k^2V_{k-1} + c_k^2\sigma_v^2 + \sigma_w^2} \end{aligned}$$

より，

$$F_k = \frac{a_k^2c_kV_{k-1} + c_k\sigma_v^2}{a_k^2c_k^2V_{k-1} + c_k^2\sigma_v^2 + \sigma_w^2} \tag{6.34}$$

が推定誤差の分散 V_k の増加を最も抑えるパラメータになる．見やすくなるように変数 X_k を入れて整理すると，$\hat{\theta}_0 = \bar{\theta}$, $V_0 = P$ として，次の推定規則を導くことができる．

$$\hat{\theta}_k = a_k\hat{\theta}_{k-1} + F_k(y_k - c_ka_k\hat{\theta}_{k-1}), \tag{6.35}$$

$$X_k = a_k^2V_{k-1} + \sigma_v^2, \tag{6.36}$$

$$V_k = X_k - \frac{c_k^2X_k^2}{c_k^2X_k + \sigma_w^2} = \frac{\sigma_w^2X_k}{c_k^2X_k + \sigma_w^2}, \tag{6.37}$$

$$F_k = \frac{c_kX_k}{c_k^2X_k + \sigma_w^2}. \tag{6.38}$$

この推定規則は **Kalman フィルタ（Kalman filter）**と呼ばれ，システム制御や時系列解析，信号処理等で広く用いられる．θ_k が n 次元，y_k が m 次元であっても，行列やベクトルの平方根などを使って同様に求められる．

課題 18 (Kalman フィルタ) 次の離散時間ダイナミクスおよび観測方程式が与えられているとする．

$$\theta_k = 0.9\theta_{k-1} + v_k, \tag{6.39}$$

$$y_k = 2\theta_k + w_k \tag{6.40}$$

ここで $\theta_k, y_k \in \mathbb{R}$ であり，v_k, w_k は互いに独立な $\mathcal{N}(0,1)$ に従う確率変数である．また，θ_0 は平均 3, 分散 2 をもつ確率分布に従うとする．このとき，$\{y_\ell\}_{\ell=1}^k$ までを使った θ_k, $k = 1, \dots 100$ の推定値 $\hat{\theta}_k$ を求め，θ_k と共に時間変化をプロットせよ．

Kalman フィルタの Bayes 的な導出や応用例は，文献 [8, 9] とその参考文献を参照されたい．また，確率制御問題へどのように応用されるかは文献 [10] を参照されたい．

6.4.2 Kalman スムーザ

Kalman フィルタは，新たにデータ y_k を得るたびに，新たな時刻の潜在変数 θ_k の推定を行うアルゴリズムである．$\alpha_i := a_i(1 - c_iF_i)$ として $\hat{\theta}_k$, $k = 1, \dots, N$ を改めて書き直すと，

$$\begin{aligned} \hat{\theta}_k =& \alpha_k\hat{\theta}_{k-1} + F_ky_k \\ =& \alpha_k\alpha_{k-1}\hat{\theta}_{k-2} + \alpha_kF_{k-1}y_{k-1} + F_ky_k \\ =& \prod_{i=1}^k \alpha_i\bar{\theta} + \sum_{\ell=1}^{k-1}\left(\prod_{j=\ell+1}^k \alpha_j\right)F_\ell y_\ell + F_ky_k \end{aligned}$$

となるので，次のように書き直すことができる．

$$\begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \vdots \\ \hat{\theta}_N \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_1\alpha_2 \\ \vdots \\ \prod_{i=1}^N \alpha_i \end{bmatrix} \bar{\theta} + M_N \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

$$M_N := \begin{bmatrix} F_1 & 0 & \cdots & 0 \\ \alpha_2F_1 & F_2 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ \prod_{2=1}^N \alpha_iF_1 & \prod_{3=1}^N \alpha_iF_2 & \cdots & F_N \end{bmatrix}$$

M_N が下三角行列になることが，各時刻 k で k までのデータ $\{y_\ell\}_{\ell=1}^k$ を使っていることと等価になる．一方，差分方程式 (6.32) の解 θ_k は，過去の情報 $\theta_{k-\ell}$, $\ell = 1, \cdots$ に影響を受けているため，

$$\begin{aligned} y_k =& c_k\theta_k + w_k = c_k(a_k\theta_{k-1} + v_k) + w_k \\ =& c_k\prod_{j=i+1}^k a_j\theta_i + (\text{noise terms}) \end{aligned}$$

より，y_k には過去の θ に関する情報も含まれている．そのため，新たなデータ y_k から推定値 $\hat{\theta}_\ell$, $\ell < k$ を更新することもできるはずである．これを可能にするのが **Kalman スムーザ (Kalman smoother)** であり，いくつかアルゴリズムは知

られているが，ここではよく用いられる **Rauch–Tung–Striebel アルゴリズム**と呼ばれるオフラインの手法を紹介する（オンラインのアルゴリズムもあるが，あまり使われない）．

時刻 1 から N までのデータ $\{y_k\}_{k=1}^N$ から Kalman フィルタによる推定値 $\{\hat{\theta}_k\}_{k=1}^N$, 推定誤差分散 $\{V_k\}_{k=1}^N$, 一段先予測誤差分散 $\{X_k\}_{k=1}^N$ が得られているものとする．時刻 k の潜在変数 θ_k を，全てのデータを用いた推定を $\hat{\theta}_k^s$ で表す．このとき，終端条件 $\hat{\theta}_N^s = \hat{\theta}_N$, $V_N^s = V_N$ とした後退方程式

$$\hat{\theta}_k^s = \hat{\theta}_k + g_k(\hat{\theta}_{k+1}^s - a_k\hat{\theta}_k) \tag{6.41}$$

$$g_k = a_k\frac{V_k}{X_{k+1}} \tag{6.42}$$

$$V_k^s = V_k + g_k^2(V_{k+1}^s - X_{k+1}) \tag{6.43}$$

の解 $\hat{\theta}_k^s$, $k = 0, \dots, N$ は，時刻 N までの全ての情報を用いた最小平均二乗誤差推定値である．導出に関しては文献 [9, 10] を参照されたい．また，V_k^s は推定誤差分散 $\mathbb{E}[(r_k - \hat{r}_k^s)^2]$ を表す．

課題 19 (Kalman スムーザ) 課題 18 の設定の元で，初期値 θ_0 の推定値とその推定誤差分散を求めよ．また，初期値の分散と比べて式 (6.41)–(6.43) からなる RTS アルゴリズムによる推定誤差分散がどれほど改善できたか，推定誤差分散と初期分散の比（分母が初期分散）で表せ．

6.5 交互最小二乗法

非線形最小二乗法は，問題によって様々な解き方があるので，一般論は存在しない．ここでは，機械学習などで多く用いられる K–**平均法（K-means clustering）**（あるいは K–平均クラスタリング）を題材に，非線形最小二乗法の一種である**交互最小二乗法（Alternating least squares）**を取り上げる．

6.5.1 交互最小二乗法

関数 $f(\theta, x)$ が $\theta = (\alpha, \beta)^\top$ に関して双線形であるとは，例えば

$$f(\theta, x) = \alpha\beta x + \beta$$

のように，推定したいパラメータ同士が掛け算の形になってしまっているというものである．$\alpha\beta$ を改めて別の変数に置き換えて推定してもよいが，それぞれのパラメータに意味のある場合もある．このような関数 $f(\theta, x)$ は，パラメータ $\theta \in \mathbb{R}^n$ を 2 つに分割すれば，分割したパラメータ $\alpha \in \mathbb{R}^{n_1}$, $\beta \in \mathbb{R}^{n_2}$，$n_1 + n_2 = n$ それぞれに対して線形になる．そこで，$f(\theta, x)$ を改めて $g(\alpha, \beta, x)$ とおいて，次のようにパラメータを求める．

Step 1 初期値 $\hat{\alpha}_0$, $\hat{\beta}_0$ を設定する．

Step 2 $j = 1, \dots$ に対し，

Step 2-1 $\beta = \hat{\beta}_{j-1}$ に固定し，α を最小化する

$$\hat{\alpha}_j = \arg\min_{\alpha\in\mathbb{R}^{n_1}} \sum_{i=1}^N \|y_i - g(\alpha, \hat{\beta}_{j-1}, x_i)\|^2$$

Step 2-2 $j = 1, \dots$ に対し，$\alpha = \hat{\alpha}_j$ を固定し，β を最小化する．

$$\hat{\beta}_j = \arg\min_{\beta\in\mathbb{R}^{n_2}} \sum_{i=1}^N \|y_i - g(\hat{\alpha}_j, \beta, x_i)\|^2$$

Step 3 収束判定条件（例えば事前に設定した $\varepsilon > 0$ に対して $(\|\alpha_{j-1} - \alpha_j\|^2 + \|\beta_{j-1} - \beta_j\|^2) < \varepsilon$）を満たすか判定し，満たさなければ j を 1 つ繰り上げて Step 2 に戻る．

同じデータを使ったまま，最小二乗問題を繰り返し解く問題になる．各最小化問題は，式 (6.5) の形で解けばよい．交互に最小化問題を解くため，交互最小二乗法と呼ばれる．アルゴリズムの性質から，$j = 1, \dots$ に対して

$$\sum_{i=1}^N \|y_i - g(\hat{\alpha}_j, \hat{\beta}_j, x_i)\|^2$$
$$\geq \sum_{i=1}^N \|y_i - g(\hat{\alpha}_{j+1}, \hat{\beta}_{j+1}, x_i)\|^2$$

なる関係も成り立つ．したがって交互最小二乗法は「悪くはならない」手法である．しかし，一般には局所的最適解への収束は必ずしも保証されるとは限らないため，初期値の工夫などが必要である．

課題 20 $x_i \in [-1,1]$, $i = 1,\dots,10000$, に対して，次の方程式で定められる入出力データが与えられているとする．

$$y_i = \alpha^\top \varphi(x_i)\beta + w_i.$$

ここで $\alpha \in \mathbb{R}^2$, $\beta \in \mathbb{R}^3$，w_i は区間 $[-1,1]$ の一様分布であるとし，

$$\varphi(x) = \begin{bmatrix} 1 & x & x^2 \\ x & x^2 & x^3 \end{bmatrix}$$

の場合を考える．10000 組の $\{(x_i, y_i)\}_{i=1}^{10000}$ から，交互最小二乗法を用いて α, β を推定せよ．このとき，10 組の異なる初期値を用意し，最もよかったパラメータとそのときに使った初期値を示せ．

異なる複数の初期値を使って最適化問題を解く方法をマルチスタート法あるいは多点スタート法という．

6.5.2 K-平均法

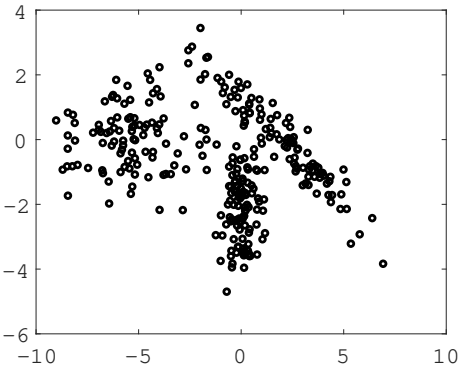


図 6.4: 観測されたデータ

K–平均法は，与えられたデータを K 個のクラスタに分類する手法であり，似たようなデータを集める代表的な方法である [4, 11]．例えば，図 6.4 は元々別々の分布から発生したデータが混在しており，元のデータは図 6.5 のように色分けされた 3 種類のデータからなる．$\{x_i\}_{i=1}^N \subset \mathbb{R}^p$ がデータとして与えられているものとする．これを K 個のクラスタに分類したい．ここで K はこちらで与える正整数とする．K をどのように決めるかの議論はここでは行わず，与えられているものとする．$\mathcal{V} := \{1,\dots,N\}$ とし，$\mathcal{V}$ の空でない部分集合 $\mathcal{V}_1,\dots,\mathcal{V}_K$ を，$\cup_{\ell=1}^K \mathcal{V}_\ell = \mathcal{V}$ かつ $\mathcal{V}_\ell \cap \mathcal{V}_k = \emptyset$, $\ell \neq k$ を満たすようにとる．この $\mathcal{V}_\ell$ がクラスタを表し，そのクラスタを特徴づける量を次で定める．

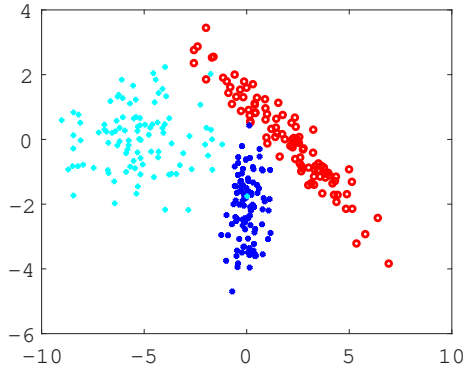


図 6.5: それぞれのデータ点を生成分布別に表した図

$$\mu(\mathcal{V}_\ell) := \arg\min_{\mu\in\mathbb{R}^p} \sum_{i_\ell \in \mathcal{V}_\ell} \|x_{i_\ell} - \mu\|^2,\ \ell = 1,\dots,K \tag{6.44}$$

これは，次のようにクラスタの中心を表す．

$$\mu(\mathcal{V}_\ell) = \frac{1}{|\mathcal{V}_\ell|} \sum_{i_\ell \in \mathcal{V}_\ell} x_{i_\ell}. \tag{6.45}$$

ただし，$|\mathcal{V}_\ell|$ は $\mathcal{V}_\ell$ の要素数を表す．問題となるのはどのようにクラスタを選ぶかであるが，これは次の最小化問題を考えればよい．

$$\min_{\{\mathcal{V}_1,\dots,\mathcal{V}_K\}} \sum_{\ell=1}^K \sum_{i_\ell \in \mathcal{V}_\ell} \|x_{i_\ell} - \mu(\mathcal{V}_\ell)\|^2. \tag{6.46}$$

ここで

$$r_{i,\ell} := \begin{cases} 1, & i \in \mathcal{V}_\ell \\ 0, & i \notin \mathcal{V}_\ell \end{cases}$$

を導入すれば，式 (6.46) は

$$\sum_{\ell=1}^{K}\sum_{i=1}^{n} r_{i,\ell}\|x_i-\mu(\mathcal{V}_\ell)\|^2$$

となる．x_i がどのクラスタに入るかを更新するには，$\mathcal{V}_\ell$ をどう変更するかの問題になるが，これは $r_{i,\ell}$ と $\mu(\mathcal{V}_\ell)$ の両方を変更しなければならないので難しい．そこで現在のクラスタ中心 $\mu(\mathcal{V}_\ell)$ を固定して μ_ℓ と表し，$r_{i,\ell}$ を更新する．$r_{i,\ell}=1$ は，x_i が $\mathcal{V}_\ell$ に所属することを意味したが，改めて x_i がどこのクラスタに所属するかは，クラスタの代表点の中で最も近いものに変更する．

$$r_{i,\ell}=\begin{cases}1, & \ell=\arg\min\limits_{k=1,\dots,K}\|x_i-\mu_k\|^2\\ 0, & \text{otherwise}\end{cases} \tag{6.47}$$

各クラスタに所属するデータが変更されたので，$\mu(\mathcal{V}_\ell)$ も計算し直す必要があり，交互に最適化問題を解くことになる．アルゴリズムをまとめると，次の通りである．

Step 1 初期値 $\mu_\ell\in\mathbb{R}^p$, $\ell=1,\dots,K$ を設定する．

Step 2 $j=1,\dots$ に対し，

Step 2-1 式 (6.47) より $r_{i,\ell}^{(j)}$, $i=1,\dots,N$, $\ell=1,\dots,K$ を定める．

Step 2-2 次の式より $\mu_\ell^{(j)}$, $\ell=1,\dots,K$ を定める．

$$\mu_\ell^{(j)}=\frac{1}{\sum_{i=1}^{N} r_{i,\ell}}\sum_{i=1}^{N} r_{i,\ell}x_i.$$

Step 3 収束判定条件（例えば事前に設定した $\varepsilon>0$ に対して $\max_\ell\|\mu_\ell^{(j-1)}-\mu_\ell^{(j)}\|^2<\varepsilon$）を満たすか判定し，満たさなければ j を 1 つ繰り上げて Step 2 に戻る．

$U:=[\mu_1,\dots,\mu_K]$, $R=\{r_{i,\ell}\}$, $X=[x_1,\dots,x_N]$ と置くと，K 平均法は次のコスト関数の最小化問題となる．

$$\|X-UR\|^2 \tag{6.48}$$

R が 0 か 1 の要素しか持たないことから，連続最適化と離散最適化の両方の性質を持った難しめの最小化問題になっており，大域的最適解を求められる保証はない．また，初期値の選び方が非常に重要になる．

課題 21 与えられたデータ $x_i\in\mathbb{R}^2$, $i=1,\dots,1000$ を 3 つのクラスターに分類せよ．異なる初期値を 10 組以上選び，最も良さそうな分類をその理由とともに示せ．

K 平均法の初期値の選び方は色々工夫されているので，インターネットでも調べてみるとよい．

6.6 発展課題

余力がある学生やより実践的な問題に取り組みたい学生へ，発展的な課題として下記の 3 つを挙げる．これまでと同様に，必要なデータは全て PandA に挙げる．それぞれの問題は下記の通りで，実機を使う 3 回生のシステム工学実験で実際に直面するものである．

1. 与えられているデータは，入力と出力がそれぞれ 1 次元の実数であり，それ以外の情報はない．このとき，与えられた入出力データを用いて $\varphi(x)$ を適切に決定して線形最小二乗推定値をもとめ，推定の良さを議論せよ．また，複数の φ を用いて比較せよ．
   - 可視化できるデータは，入出力データをプロットしてから関数の検討をつけること．
2. 単振り子の方程式が与えられているとする．

$$ml\frac{d^2\phi(t)}{dt^2}+K\frac{d\phi(t)}{dt}+mg\sin(\phi(t))=F(t) \tag{6.49}$$

ここで m は質点の質量（1 kg），l は中心から質点までの長さ (0.5 m)，K は粘性係数 (0.01 N·s)，g は重力加速度（9.8 N·kg^{-1}），$F(t)$ は外力である．ここで $t=0$ で静止状態の単振り子に $F(t)=\sin(t)$ の外力を加えたとき

の，時間幅 $\delta t = 0.002$ ごとに得られたデータ $\{\phi(k\delta t), F(k\delta t)\}_{k=1}^{N}$ が与えられているものとする．長さ l および重力加速度の値は既知としたとき，未知パラメータ (m, K) を与えられたデータから推定せよ．

- ３回後期のシステム工学実験では，正解を知らないまま，このような問題に取り組まなければならない．どのくらいのデータがあれば適切な推定値とみなしてよいかを考えながら行えば，実際にどのくらいデータを取る必要があるかの目安になる．

3. 時系列データが周期信号であることが分かっているとき，その周波数成分を調べたい．しかし，時系列データを取得してから Fourier 変換するには，データが膨大すぎて特別な計算機でなければメモリが不足し，計算できない．そのため，実験を行いながらデータ処理をする方法として，逐次最小二乗法が用いられる．$\alpha_j, \beta_j \in \mathbb{R}$ として

$$y(t) := \sum_{j=1}^{11} \alpha_j \sin(\omega_j t) + \beta_j \cos(\omega_j t),$$
$$\omega_j = 2\pi 10^{-1+2(j-1)/10},\ j = 1, \dots, 11$$

とし，与えられたデータ $\{(t_k, y(t_k))\}$, $t_k = 0.002 \times k$ から未知パラメータ $\{(\alpha_j, \beta_j)\}_{j=1}^{11}$ を求めよ．また，ω_j を横軸に，縦軸に $(\alpha_j^2 + \beta_j^2)$ を両対数グラフで表示せよ．

- （システム解析入門を取っている学生向け）この時系列データは，線形システムに周期的な入力（周期は問題設定と同じ）を複数同時に入れたときの出力信号である．プロットされた値は伝達関数の Bode ゲイン線図に相当する．

欠測値のある場合の推定や LASSO などのスパースモデリングなど，扱わなかった題材でよく用いられている手法もたくさんあるので，それらは適宜補ってもらいたい．

参考文献

[1] 松原 望, 統計学, 東京図書, 2013.

[2] 久保川 達也, 現代数理統計学の基礎, 共立出版, 2017.

[3] 小西 貞則, 多変量解析入門, 岩波書店, 2010.

[4] 森, 黒田, 足立, 最小二乗法・交互最小二乗法, 共立出版, 2017.

[5] 北川 源四郎, 時系列解析入門, 岩波書店, 2005.

[6] 持橋, 大羽，ガウス過程と機械学習, 講談社, 2019.

[7] I. Rosh, G. Y. Grabarnik, Sparse Modeling: Theory, Algorithms, and Applications, Chapman & Hall/Crc, 2006.
（Rish, Grabarnik，スパースモデリング: 理論，アルゴリズム，応用, ジャムハウス, 2019）

[8] 足立, 丸田, カルマンフィルタの基礎, 東京電機大学出版, 2012.

[9] Särkkä, Bayesian Filtering and Smoothing, Cambridge University Press, 2013.

[10] F. L. Lewis, L. Xie, D. Popa, Optimal and Robust Estimation with an Introduction to Stochastic Control Theory, CRC Press, 2008.

[11] C. M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2006.
（ビショップ，パターン認識と機械学習 上下, 丸善出版, 2012）

[大木健太郎]

付録　データの取り扱い

付録 1　データの読み込み

各課題にに取り組むにあたって必要なデータファイルは panda に用意されている．それらデー

タファイルの C 言語上での読み込み方を示しておく．例として図 6.6 に示したようなデータファイ

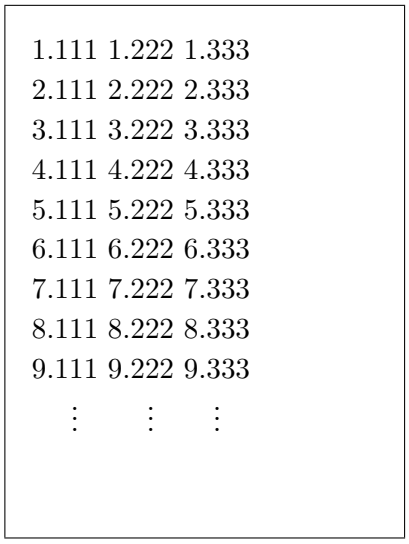


図 6.6: データファイル sample.txt

ル sample.txt を読み込むことを考える．このデータファイルはテキストファイルであり，3 列で数字が並んでいる．このとき，以下のようなプログラムで sample.txt の各列をそれぞれ double 型の配列 a[N]，b[N]，c[N] に読み込むことができる．

```
#include<stdio.h>
#define N 1000//データファイルの行数

int main(){

  double a[N],b[N],c[N];
  FILE *fp;

  fp=fopen("sample.txt","r");
  int i=0;
  while(fscanf(fp, "%lf %lf %lf",
        &a[i],&b[i],&c[i])!=EOF){
          i++;
  }
  fclose(fp);

}
```

読み込むデータが整数値でそれを int 型として読み込みたいときは，対応する配列を int 型に，対応する%lf を%d にすればよい．

付録 2　データのプロット

データのプロットについて説明する．方法は無数に考えられるが，ここでは学術的な文献などでよく用いられる Gnuplot の使い方の基礎を述べておく．インストールの方法は検索すればすぐに出てくるので割愛する．（例えば https://mtkbirdman.com/c-gnuplot-install などを参照されたい.）また，Gnuplot の機能もほとんどここでは説明しきれない．必要な時に検索し，少しずつ身に着けていくのが良い．

散布図

Gnuplot ではどのような図を描くかをコマンドで指示する．例えば図 6.6 の sample.txt の 1 列目と 2 列目を散布図として描きたいときには

```
plot "sample.txt" u 1:2
```

と入力する．なお，これはカレントディレクトリ（作業しているフォルダ）に sample.txt が置かれている場合であり，別のところに置いてある場合にはパスを指定する必要がある．同様に sample.txt の 2 列目と 3 列目の散布図を描くには

```
plot "sample.txt" u 2:3
```

と入力する．重ねて書きたい場合には

```
plot "sample.txt" u 1:2,
     "sample.txt" u 2:3
```

と入力する．

なお，何列目のデータを使うかを指定しているところ（‘‘u 1:2’’ や ‘‘u 2:3’’）を省略すると，自動的に 1 列目と 2 列目が使われる．

線グラフ

線グラフを描きたい場合には，

```
plot ”sample.txt” u 1:2 w l
```

または

```
plot ”sample.txt” u 1:2 w lp
```

のように書く．こうすると，散布図の点が順に線で結ばれる．線は sample.txt の一行目に対応する点から二行目に対応する点，二行目に対応する点から三行目に対応する点，三行目から四行目，··· というふうな順番で結ばれていく．そのため，関数のグラフを図示したい場合など（通常,「折れ線グラフ」という言葉でイメージするもの）を描きたい場合には横軸に対応するデータは大きさの順でソートされていなければならない．

また，

```
plot sin(x)
```

のように関数形を与えてやると直接そのグラフを描いてくれる．

その他

- 凡例の名称を変更することもできる．

  ```
  plot ”sample.txt” u 1:2 title ”hoge”
  ```

  と書くことによって名称が hoge になる．

- 横軸にラベル（横軸が何を表せているかの説明）を付けるには

  ```
  set xlabel ”hoge”
  ```

  のように指定する．指定した後にもう一度描画してやるとラベルが付される．縦軸にラベルを付けるには xlabel を ylabel に置き換える．

- 描画する範囲を指定することもできる．

  ```
  plot [a:b][c:d] ”sample.txt”
  ```

  とすると横軸が a から b，縦軸が c から d の範囲の図が描かれる． なお，横軸の範囲のみを指定することもできる．

- 横軸を対数表示にするには

  ```
  set log x
  ```

  と入力した後に再度描画する．縦軸を対数表示したければ x を y に置き換える．x と y のどちらも書かなければ両方の軸が対数表示になる．

- その他，点や線の色・種類・大きさ・太さ，文字のサイズ等も変更できる．調べてみよ．

[岩崎淳]

第7章　組合せ最適化

7.1　目的

組合せ最適化（離散最適化）とは，解が離散的に定義されていたり，順序や割当のように組合せ的な構造によって表現できる最適化問題のことである．現実の多くの場面において自然に現れる問題であるが，問題の構造をうまく捉えなければ効率よく解くことは難しい．本実験では最短路問題という問題を通して組合せ最適化問題の難しさを体感し，代表的な解法の一つである分枝限定法について学ぶことを目的とする．

7.2　最短路問題

節点集合 $V = \{1, 2, \ldots, n\}$ と有向枝集合 $E \subseteq V \times V$ から成るグラフ $G = (V, E)$ と，各枝 $(u, v) \in E$ の長さ $d(u, v)$ が与えられたとき，$(v_i, v_{i+1}) \in E$, $i = 0, 1, 2, \ldots, l-1$ を満たす節点の列 $P = \langle u = v_0, v_1, v_2, \ldots, v_l = v \rangle$ を始点 u から節点 v への**路**（または有向路）P と呼び，路に含まれる枝の長さの総和を路の**長さ** $\ell(P)$ と言う．グラフ G において，点 u から点 v への有向路 P の中で長さ $\ell(P)$ を最小にするものを探す問題は応用上も離散最適化の理論においても重要な問題である．枝重みは一般的には負の場合も考えることがある．とくに，長さの和 $\ell(C)$ が負である有向閉路 C が存在することもある（これを**負閉路** (negative cycle) と呼ぶ）．この場合，選択する有向路 P を「単純」（同じ点を二度以上通らない）としておかないと負閉路を何度も通り，路の長さ $\ell(P)$ がいくらでも小さくなってしまうことがある．そこで，選択する有向路は単純なものに限って議論する．点 u から点 v への単純有向路 P の中で長さ $\ell(P)$ を最小にするものを点 u から点 v への最短路と呼び，その長さ $\ell(P)$ を $\mathrm{dist}(u, v)$ で表す．

与えらた二点 $u, v \in V$ に対して，点 u から点 v への最短路を求める問題は負の枝重みを許す場合は一般に NP-困難な問題となる（多項式時間のアルゴリズムは存在しないと予想されている）[1]．当実験の目的はこの場合の最短路問題に対してアルゴリズムを設計及び実装することである．

7.3　分枝限定法

組織的な場合分け（分枝）に基づく列挙法に不要な場合の排除機能（限定）を付したほとんど全ての組合せ最適化問題に適用できる幅広い計算方法である．よい構造を持たない問題における厳密最適解を求めるためによく使われる．

代表的な離散問題が解決できる方法として詳しく説明する参考文献は少なくない [2, 3, 6]．特に，以下の説明は参考文献 [5] に基づくものである．

分枝限定法は，問題をいくつかの小規模な問題に分割し，その全てを解くことで等価的に元の問題を解くという考え方に基づいている．小規模な問題への分割は，例えば，ある一つの決定変数の値を固定し（グラフ探索の場合，枝や節点を決め），それぞれの場合を個別に考察することによって実現できる．このように問題を分割する操作を**分枝操作**（branching operation）という．分枝操作を繰り返し行うことで，すべての場合を列挙することができるが，その過程は生成木と呼ばれる根付き木を用いて表現できる．生成木において，根は元の問題に対応し，その二つの子はそれぞれある変数の値を 0 または 1 に固定した 2 つの場合に対応する．その他の節点も同様である．よって，内部の節点は根からその節点への路に対応していくつかの変数の値を 0 か 1 に固定した問題に対応し，葉は全ての変数の値が定まった解に対応する．一部の変数の値が固定された問題を**部分問題**（partial problem）と呼び，この例の場合，部分問題も元の問題と等しい構造である．

分枝限定法では，実際に全ての葉を列挙するわけではなく，その一部分だけを生成する．生成木のうち実際に生成された節点のみから成るものを

探索木（search tree）という．ある部分問題（節点）に対して，

(i) その最適値，

(ii) その部分問題が実行不可能であること，

(iii) その部分問題の最適解が元の問題の最適解とはならないこと，

のいずれかが分かれば，その部分問題をこれ以上調べる必要はなく，その下の節点（子孫と呼ぶ）の探索を省略できる（**終端する**（terminate）という）．これを**限定操作**（bounding operation）と呼ぶ．探索の過程で，まだ終端されていない部分問題を活性（active）であるといい，全ての部分問題が終端されたとき，つまり活性部分問題がなくなったとき，分枝限定法は厳密な最適解を与える（もしくは実行可能解が存在しないことを証明する）．

限定操作の代表例である**下界値テスト**について簡単に述べる．近似解法等を用いて実行可能解を一つでも得ることができれば，その目的関数値は最適値の上界（upper bound）値となる．また，探索が生成木の葉に到達したときにも実行可能解が得られる場合がある．これまでの探索で得られている最良の上界値を暫定値（incumbent value）という．一方，制約を緩めた緩和問題を解くなどして得られた最適値は，元の問題の最適値の下界（lower bound）値となる．このようにして得られた上下界値を用い，「ある部分問題の下界値が暫定値以上であれば，その子孫を調べる必要はない」ことが結論できる．その部分問題の子孫を全て調べても，暫定値よりもよい目的関数値を持つ解は見つからないからである．以上が下界値テストである．（最大化問題の場合は上界と下界の役割が入れ替わる．）

なお，計算時間の都合などにより活性部分問題が残っている状態で探索を終了し，その時点での暫定値を出力する方法もしばしば用いられる．このとき，出力した値が最適である保証はなく，分枝限定法を近似解法として用いることになるが，この場合でも，活性部分問題の下界値の最小値が元の問題の下界値となり，出力値の精度を測る指標としてこれを用いることができる．

7.3.1 アルゴリズムの説明：擬似コード

擬似コードを書く時以下の点に注意すること：

- 代入 (substitution) には，「:=」を使う（「←」を使うこともある）．
- **goto** 文又は **break** 文は使わないようにする．
- 構造を分かりやすくするため，
  if ... **then** ... **end if**
  if ... **then** ... **else** ... **end if**
  for ... **do** ... **end for**
  while ... **do** ... **end while**
  の形を守り，字下げ (indentation) を深さに応じて使う．とくに入れ子構造が分かりやすくなるようにする．
- 関数の返す出力には **return**，印字・ファイルへの出力には **output** を使う．
- 連続するコマンドは「;」で区切る（機械的にコマンドの終わりに付けるのではない）．

グラフ $G = (V, E)$ と節点 $s, t \in V$ が与えられた時，点 s から点 t へ至るすべての路を出力するアルゴリズムの例は以下の通りである．

再帰的関数 3 PathAll(x, F)

入力: グラフ G 上の節点 x 点 s から点 x へ至る有向路に含まれている節点集合 F;

出力: グラフ G 上点 s から点 t へ至り，かつ節点集合 F を含む有向路.

```
1: for すべての有向枝 (x, y) ∈ E, y ∉ F について do
2:   if y = t then
3:     output F ∪ {y}
4:   else
5:     PathAll(x, F ∪ {y})
6:   end if
7: end for.
```

尚，すべての有向路を出力するためには PathAll$(s, \emptyset)$ を実行する．

7.3.2 実験課題

課題 1
グラフ $G = (V, E)$ ，節点 $s, t \in V$ 及び各枝 $(u, v) \in E$ の長さ $d(u, v)$ が与えられたとき，点 s から点 t へ至る最短路を求める分枝アルゴリズムを設計し，その擬似コードを書け.

課題 2
課題 1 で設定したアルゴリズムを実装し，グラフの節点数及び枝数に対して実行時間及び探索したノード数を調べ，表とグラフにまとめよ.

課題 3
課題 1 で設定したアルゴリズムに関して限定操作を導入し，その妥当性を説明および証明せよ．限定操作別の擬似コードを書き，最後に全体の限定操作アルゴリズムの擬似コードを書け.

課題 4
限定操作を実装し，適当なインスタンスで課題 2 の実装と実行時間及び探索したノード数を比較せよ．結果を表とグラフにまとめよ.

参考文献

[1] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979.

[2] 浅野孝夫, 情報の構造 (上)(下), 日本評論社, 1994.

[3] 茨木俊秀, C によるアルゴリズムとデータ構造, 昭晃堂, 1999.

[4] S. バーセ，アルゴリズム入門 – 設計と解析，ピアソンエデュケーション，2002.

[5] 柳浦睦憲，野々部 宏司，分枝限定法— さらなる計算効率の希求 —,「システム／制御／情報」 第 50 巻 第 9 号，350-356，2006.

[6] T. コルメン，C. ライザーソン，R. リベスト，C. シュタイン，アルゴリズムイントロダクション第 2 巻，アルゴリズムの設計と解析手法 – 改訂 2 版–，近代科学社, 2007.

付録: プログラムの実行時間の計測方法

ここでは，gettimeofday() を使用する方法を説明する．gettimeofday() を使用するにはプログラムの最初で以下のように記述し，time.h と sys/time.h をインクルードし，関数 gettimeofday_sec() を定義する必要がある.

```
#include <time.h>
#include <sys/time.h>

double gettimeofday_sec(){
    struct timeval tv;
    gettimeofday(&tv, NULL);
    return tv.tv_sec
      + (double)tv.tv_usec*1e-6;
}
```

下の記述のように実行時間を計測したい操作の前後で gettimeofday_sec() を呼び出して値を保存すると，その差に 1000 をかけたものが実行時間（ミリ秒）になる.

```
/* 時間計測開始 */
double start =
gettimeofday_sec();

(計測したい操作の記述)

/* 時間計測終了 */
double end = gettimeofday_sec();

printf("time = %lf msec.\n",
(end-start)*1000);
```

この方法で計測される時間は実時間ではなく，プロセスによって消費された CPU 時間であるので，計算機で同時に実行されている他のプロセスによる影響は受けない.

付録: 代表的な解決法（非負閉路はない場合）

節点集合 $V = \{1, 2, \ldots, n\}$ と枝集合 $E \subseteq V \times V$ から成るグラフ $G = (V, E)$ と各枝 $(u, v) \in E$ の長さ $d(u, v)$ が与えられたとき，$d(u, v)$ の値によって最短路問題が解ける方法は以下のようになる．ただし，以下ではグラフ G の節点数を n，枝の数を m とする．

(a) 枝重みが全て非負の場合：ダイクストラ法により多項式時間で解ける．

(b) 負の枝重みがある場合：

- **(i)** 負閉路が存在するかどうかの判定が $O(nm)$ 時間で行える（ベルマン･フォード法）．
- **(ii)** 負閉路が存在しない場合：二点間の最短路の長さは $O(nm)$ 時間で求まる（ベルマン・フォード法）．
- **(iii)** 負閉路が存在する場合：二点間の最短路の長さを求める問題は NP-困難である（多項式時間のアルゴリズムは存在しないと予想されている）．

当実験テーマではあるグラフ G とその枝重みによって負閉路が存在する場合二点間の最短路の長さを求める問題に対して分枝限定法のアルゴリズムを考えた．続いて，ベルマン・フォード法とダイクストラ法を軽く紹介するためそれぞれの擬似コードを提供する．

ベルマン・フォード法

ベルマン･フォード法では各 $v \in V$ と $k \in \{0\} \cup [n-1]$ について変数 $\mathrm{label}(v, k)$ と $\mathrm{prev}(v)$ が定義される．

次の擬似コードでは，簡単のため，すべての枝重みは非負であると仮定している．

$\mathrm{label}(v, k)$ には, 高々 k 本の枝しか含まない始点 s から節点 v へ至る路の中で最短なものの長

アルゴリズム 4 ベルマン・フォード法

入力: 節点の数が n である 無向グラフ $G = (V, E)$, 始点 $s \in V$ 及び各枝 $(u, v) \in E$ の長さ $d(u, v) \geq 0$;

出力: 各節点 $v \in V$ に対して s から v までの最短路の長さ $\ell(v)$.

```
1:  label(s, 0) := 0;
2:  for すべての節点 v ∈ V\{s} do
3:    label(v, 0) := ∞
4:  end for;
5:  for k := 1, 2, ..., n − 1 do
6:    for すべての節点 v ∈ V do
7:      label(v, k) := label(v, k − 1)
8:    end for;
9:    for 各枝 (u, v) ∈ E do
10:     if label(u, k − 1) + d(u, v) < label(v, k)
        then
11:       label(v, k) := label(u, k − 1) + d(u, v)
12:     end if;
13:     if label(v, k−1)+d_vu < label(u, k) then
14:       label(u, k) := label(v, k − 1) + d_vu
15:     end if
16:   end for
17: end for;
18: for すべての節点 v ∈ V do
19:   ℓ(v) := label(v, n − 1)
20: end for;
21: return ℓ.
```

さが計算される．枝の長さが非負であれば，最短路のうち高々$n-1$本の枝しか含まないものが必ず存在するので，$\mathrm{label}(v, n-1)$ が節点 v までの最短路の長さとなる．また，$\mathrm{prev}(v)$ がその最短路において v の直前の節点を指す．上の記述では，アルゴリズムが何を出力するか示していないが，最短路は prev の指す節点をたどっていくことによって求められる．

ダイクストラ法

ベルマン・フォード法とは違い，ダイクストラ法では任意の $v \in V$ についてのみ，変数 $\mathrm{label}(v)$ が定義される．

アルゴリズム 5 ダイクストラ法

入力: 節点の数が n である 無向グラフ $G = (V, E)$, 始点 $s \in V$ 及び各枝 $(u, v) \in E$ の長さ $d(u, v) \geq 0$;

出力: 各節点 $v \in V$ に対して s から v までの最短路の長さ $\ell(v)$.

```
1:  P := ∅;
2:  label(s) := 0;
3:  for すべての節点 v ∈ V\{s} do
4:    label(v) := ∞
5:  end for;
6:  while P ≠ V do
7:    u := V\P の中で label の値が最小の節点;
8:    P := P ∪ {u};
9:    for 節点 u に接続する枝 (u, v) ∈ E のうち、
            v ∉ P であるものすべて do
10:     if label(u) + d(u, v) < label(v) then
11:       label(v) := label(u) + d(u, v)
12:     end if
13:   end for
14: end while;
15: for すべての節点 v ∈ V do
16:   ℓ(v) := label(v)
17: end for;
18: return ℓ.
```

アルゴリズムが終了した時点での $\mathrm{label}(v)$ が節点 v までの最短路の距離，$\mathrm{prev}(v)$ が最短路において v の直前の節点である．

[Aleksandar Shurbevski]

第8章　ニューラルネットワークによる機械学習

v3のテキスト使う

8.1　目的

本節では、ニューラルネットワーク (Neural Network; NN) を用いた機械学習の基礎を学習する。この枠組みでは、手書き文字認識など多くのタスクが、NN に含まれるパラメータに関する特殊な構造を有した最適化問題として定式化される。そこで、8.2 節においてこれらの最適化問題を効率よく解く手法である確率勾配法や誤差逆伝播法などの考え方とその実装方法を概説する。さらに、8.3 節以降では、Tensorflow と呼ばれるフレームワークを用いてアルゴリズムを実装する。ただし、独力でこれらのコードが書けるようになることは目指さず、準備されたサンプルコードを利用して、動作を理解することを目標とする。

8.2　数学的準備

8.2.1　ニューラルネットワーク

まずは図 8.1 で表現される関数を考えよう。それぞれのノード（円）は活性化関数と呼ばれる非線形関数の入力と出力の関係

$$h_1^{(1)} = f^{(1)}(u_1^{(1)}) \tag{8.1}$$

などを表現している。こうしたノードの縦方向の集まりのことを層と呼ぶ。層と層をつなぐエッジ（矢印）は重み $w_{ij}^{(l)}$ をもつ線型結合

$$u_j^{(l)} = \sum_i w_{ij}^{(l)} h_i^{(l-1)}$$

をあらわす。NN はこうした構造をもつ関数であり、深層学習とは層の数が非常に多い NN を用いた機械学習の手法や、その周辺の研究領域のことを指す。

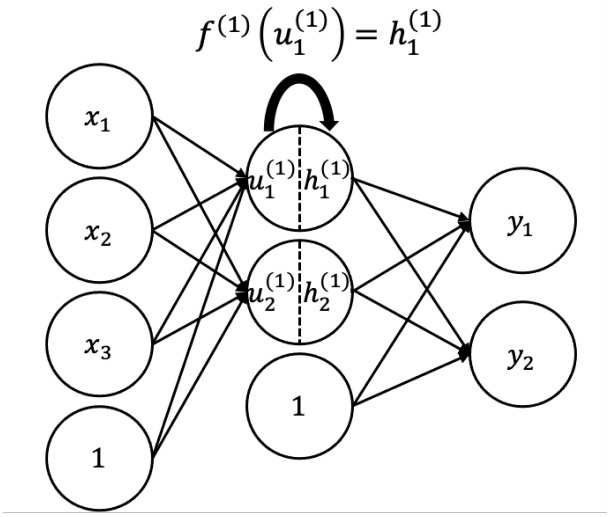


図 8.1: 3 層のニューラルネットワーク

より一般的に L 層からなる NN を考え、第 l 層のノードの個数とそのノードを d_l, $\boldsymbol{u}^{(l)} = (u_1^{(l)}, \ldots, u_{d_l}^{(l)})^T$ と表記する。まず、入力層と呼ばれる第 1 層（input layer）は活性化関数を持たず、入力ベクトル $\boldsymbol{x} = (x_j)$ をそのまま出力する。

$$\boldsymbol{u}^{(1)} = \boldsymbol{x} \tag{8.2}$$

つぎに中間層 (internal layer) や隠れ層 (hidden layer) と呼ばれる第 l 層（$1 < l < L$）における演算処理は、パラメータ $w_{ij}^{(l)}$ と $b_j^{(l)}$ をまとめて $\boldsymbol{W}^{(l)}, \boldsymbol{b}^{(l)}$ と表記すると、

$$\boldsymbol{u}^{(l)} = \boldsymbol{W}^{(l)}\boldsymbol{h}^{(l-1)} + \boldsymbol{b}^{(l)}, \quad \boldsymbol{h}^{(l)} = f^{(l)}\left(\boldsymbol{u}^{(l)}\right) \tag{8.3}$$

と書ける。活性化関数 $f^{(l)}$ はノードごとに変えても構わないが、一般的には層ごとに共通の関数を用いる。最後の第 L 層が出力層に相当し、出力 $\hat{\boldsymbol{y}}$ は次で与えられる。

$$\hat{\boldsymbol{y}} = f^{(L)}(\boldsymbol{u}^{(L)}) \tag{8.4}$$

このように、入力が与えられたとき、NN の各層を順番に計算していき、出力まで計算を行うアーキテクチャを順伝播 (feed forward; FF)NN と呼ばれる。

注意 8.2.1 隠れ層の活性化関数をタスクに応じて定める一般的な基準は未だ存在しない。現場では経験則や試行錯誤に頼ることが多いが、役に立つ活性化関数の選び方がいくつも知られている。例えば、学習をスムーズに進行させるには活性化関数として、正規化線形関数 (rectified linear unit; ReLU)

$$f(u) = \max\{0, u\} = \begin{cases} u & (u \geq 0) \\ 0 & (u < 0) \end{cases}$$

を用いると良いことがわかっている。活性化関数を持つノードを ReLU とも呼ぶが、関数自体も略して ReLU と呼ぶ。

8.2.2 関数回帰と最適化

設計する NN に課すタスクに応じてその構造を決定し、重み係数などのパラメータ集合 $\boldsymbol{w}$ に関する最適化問題を設定しなければならない。代表例として、訓練データ $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i\in\mathcal{B}}$, $\mathcal{B} = \{1, 2, \ldots, |\mathcal{B}|\}$ に対して、$\hat{\boldsymbol{y}}(\boldsymbol{x}_i) \approx \boldsymbol{y}_i$ が成り立つ関数 $\hat{\boldsymbol{y}}$ を求める関数回帰問題を考える。この問題に対して、$\boldsymbol{w}$ によりパラメトライズされる関数の集合 $\{\hat{\boldsymbol{y}}(\boldsymbol{x}; \boldsymbol{w})\}_{\boldsymbol{w}}$ から求める関数を探すことにしよう。このとき、ある $\boldsymbol{w}$ を用いたモデルの予測値 $\hat{\boldsymbol{y}}(\boldsymbol{x}_i; \boldsymbol{w})$ と実際のデータ $\boldsymbol{y}_i$ ができるだけ近くなるように、平均二乗誤差

$$L(\boldsymbol{w}) = \frac{1}{|\mathcal{B}|} \sum_{i\in\mathcal{B}} \mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i; \boldsymbol{w}), \boldsymbol{y}_i), \tag{8.5}$$

$$\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \|\hat{\boldsymbol{y}} - \boldsymbol{y}\|^2 \tag{8.6}$$

に対して、

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} L(\boldsymbol{w}) \tag{8.7}$$

とすることは妥当であろう。特別な場合として、$\boldsymbol{x} \in \mathbb{R}^{n_x}, \boldsymbol{y} \in \mathbb{R}^{n_y}$ に対して、アファイン関数

$$\hat{\boldsymbol{y}}(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \boldsymbol{W} \in \mathbb{R}^{n_y \times n_x}, \boldsymbol{b} \in \mathbb{R}^{n_y}$$

によりフィッティングを行う線形回帰を含む。以降では、線形回帰は困難なデータにも適用することを目指し、前節で導入した NN を関数集合として採用する。このとき、$\boldsymbol{w} = (\boldsymbol{W}^{(1)}, \boldsymbol{b}^{(1)}, \ldots, \boldsymbol{W}^{(L)}, \boldsymbol{b}^{(L)})$ であり、層の数や活性化関数などは適切に選択する必要がある。

他の代表的なタスクとして、データを 2 つのクラスに分類する 2 値分類がある。これは、各 $\boldsymbol{x}_i$ に対して $\boldsymbol{y}_i \in \{0, 1\}$ が与えられていると考えれば良い。求めたい関数は $\{0, 1\}$ に値をとる関数であるため、出力層がシグモイド関数

$$f^{(L)}(x) = 1/(1 + e^{-x}) \tag{8.8}$$

である NN が標準的に用いられ、$\hat{\boldsymbol{y}}(\boldsymbol{x}; \boldsymbol{w}) < 1/2$ であるとき所属クラスを 0、それ以外の場合は 1 と判定する。誤差関数としては、

$$\mathcal{L}_{ce}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -(\boldsymbol{y} \log \hat{\boldsymbol{y}} + (1 - \boldsymbol{y}) \log(1 - \hat{\boldsymbol{y}})) \tag{8.9}$$

を採用すれば $\boldsymbol{y}_i = 1$ であれば $\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{w}) \approx 1$、$\boldsymbol{y}_i = 0$ であれば $\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{w}) \approx 0$ となることが期待できる。

8.2.3 勾配降下法による学習

NN の学習は損失関数 $L(\boldsymbol{w})$ の最小化として定式化されていた。NN における損失関数は一般に非常に複雑な非凸関数であるため、大域的極小値 (global minimum) 以外にも、膨大な数の局所的極小値 (local minima) を持つ。実は、深層学習では真の最小値を見つけずとも、損失関数の良い極小値さえ見つければ十分であることが予想されている。そこで以下では方程式

$$\frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}} = 0$$

を満たす停留点を求める。さまざまな場面で用いられるニュートン法は、ヘシアン逆行列の評価にかかる計算コストが大きいため、深層学習ではほとんど用いられない。以下では、(8.7) が

- 決定変数 $\boldsymbol{w}$ が NN という特殊な関数に含まれるパラメータである、
- 損失関数 (8.5) が各データ i に対する誤差 $\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i; \boldsymbol{w}), \boldsymbol{y}_i)$ の和である

という特徴を利用した効率的な最適化手法を紹介する。以下では記述を簡単化し、中心となる考え方に注目するために $d_l = 1,\ \boldsymbol{b}^{(l)} = 0$ とするが、一般の場合も同様の議論が成立する。

誤差逆伝播法　パラメータ $\boldsymbol{w}$ に関する損失関数 $\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x};\boldsymbol{w}))$ の $\boldsymbol{w}$ に関する勾配を計算する。ここで、NN の特別な構造を用いると、素朴に $\boldsymbol{w}$ の各要素に関する勾配を独立に計算するよりも大幅に計算量を削減することができる。この手法は誤差逆伝播法 (back propagation) と呼ばれ、1986 年にラメルハートらによって発表され、NN の第 2 次ブームを巻き起こした。

まず、微分の連鎖律と $\boldsymbol{h}^{(l-1)}$ が $\boldsymbol{W}^{(l)}$ に依存しないという性質を用いると、

$$\frac{\partial \mathcal{L}}{\partial w_{ij}^{(l)}} = \frac{\partial \mathcal{L}}{\partial u_i^{(l)}} \frac{\partial u_i^{(l)}}{\partial w_{ij}^{(l)}} = \delta_i^{(l)} h_j^{(l-1)} \tag{8.10}$$

が $l \leq L$ に対して成り立つ。ここで、$\delta_i^{(l)} = \partial \mathcal{L}/\partial u_i^{(l)}$ とした。一方で、$l < L$ に対して

$$\begin{aligned}\delta_i^{(l)} &= \sum_{k=1}^{d_{l+1}} \frac{\partial \mathcal{L}}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial u_i^{(l)}} \\ &= \sum_{k=1}^{d_{l+1}} \delta_k^{(l+1)} w_{ki}^{(l+1)} f'(u_i^{(l)})\end{aligned} \tag{8.11}$$

および $l = L$ に対して

$$\delta_i^{(L)} = \frac{\partial \mathcal{L}}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial u_i^{(L)}} = \frac{\partial \mathcal{L}}{\partial \hat{y}_i} f'(u_i^{(L)}) \tag{8.12}$$

が成り立つ。

したがって、与えられた $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ および $\boldsymbol{w}$ に対して、具体的に $\mathcal{L}(\boldsymbol{y}(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i)$ の $\boldsymbol{w}$ に関する勾配を求めるには、

1. 順方向に (8.2),(8.3),(8.4) により $\boldsymbol{h}^{(l)}, l < L$ を求める、
2. (8.12) により、$\delta^{(L)}$ を求める、
3. (8.11) により $\delta^{(l)}, l < L$ を逆方向に求める、
4. (8.10) により $\partial \mathcal{L}(\boldsymbol{y}(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i)/\partial w_{ij}^{(l)}$ を求める、

という手順をたどればよい。この手順は L によらないが、層が深くなるに連れて勾配消失問題や勾配爆発問題などが起こることが知られている。パラメータの事前学習や活性化関数の工夫などにより、こうした問題を避ける手法の開発が盛んにおこなわれている。

課題 22 スカラー変数 $x,\ w_2,\ w_3$ に対して

$$\begin{aligned}&u^{(1)} = x,\ h^{(l)} = f(u^{(l)}), \\ &u^{(l)} = w^{(l)} h^{(l-1)},\ \hat{y} = f^{(3)}(u^{(3)})\end{aligned}$$

を定義し，$f(u) = u^2,\ \mathcal{L}(\hat{y}) = \hat{y}^2$ とする．このとき，$u^{(l)},\ h^{(l)},\ \hat{y}, \mathcal{L}$ を x, w_2, w_3 の関数としてあらわし、$\frac{\partial \mathcal{L}}{\partial w_l},\ l = 2, 3$ を求めよ。またこの例において

$$\delta^{(3)} = (2\hat{y})(2u^{(3)}),\ \delta^{(2)} = \delta^{(3)} w^{(3)} (2u^{(2)})$$

を用いて誤差逆伝播法の構造を説明せよ。

この課題において、たとえば $(x, w^{(2)}, w^{(3)}) = (1, 2, 3)$ における勾配を求める場合、$u,\ h,\ \delta$ は簡単な代入操作を繰り返すことで計算することができ、勾配もそれらの積で表現される。一方で、$\mathcal{L}$ およびその偏微分の解析表現を求め、値を代入することも可能であるが効率が悪い。また、変数が増えると、誤差逆伝播法においては使い回せる値が増えるため、その差はさらに顕著となる。

課題 23 活性化関数にシグモイド関数は勾配消失問題を起こすため、現在使われない。一方で、ReLU は勾配消失を起こす可能性が低いことが知られている。このことについて説明せよ。

確率的勾配降下法　勾配降下の過程において、局所的極小値にはまる (図 8.2.3 左) 状況をできるだけ回避するために確率的要素を取り入れる。各更新ステップ t で、ミニバッチと呼ばれる一部の訓練サンプル $\mathcal{B}^{(t)}$ のみを用いる方法をミニバッチ学習と呼ぶ（$\mathcal{B}^{(t)} = \mathcal{B}$ の場合をバッチ学習と呼ぶ）。通常ミニバッチは学習前にランダムに作成しておく。そしてステップ t では、ミニバッチ上で平均した誤差関数

$$L^{(t)}(\boldsymbol{w}) = \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i) \tag{8.13}$$

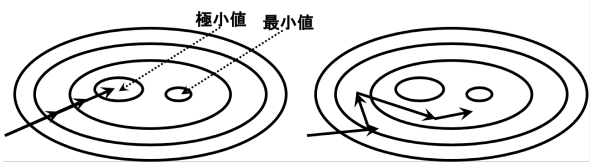


図 8.2: ミニバッチで極小値を避けるイメージ。左は極小値にはまってしまうが、ミニバッチによって極小値を避けて最小値に向かうことができる。

の勾配にもとづいて $\boldsymbol{w}$ を更新する。特に、各時刻のミニバッチに 1 つの訓練しか含まない $|\mathcal{B}^{(t)}| = 1$ という場合をオンライン学習や確率的勾配降下法と呼ぶ。ミニバッチ学習ではミニバッチ選択のランダム性により、時刻ごとに損失関数 $L^{(t)}(\boldsymbol{w})$ もランダムに変化し、望ましくない臨界点にはまり込む可能性がぐっと小さくなる (図 8.2.3)。なお、各 i に対する $\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i)$ の勾配計算は容易に並列化できるため、並列計算環境がある場合には、ある程度のサイズのミニバッチを利用する方が良い。

ミニバッチ $\mathcal{B}^{(t)}$ 内の訓練データに対して計算された $\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i)$ を用いて、勾配

$$\Delta \boldsymbol{W}^{(t,l)} = -\frac{1}{|\mathcal{B}^{(t)}|}\sum_{i\in\mathcal{B}^{(t)}}\frac{\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i)}{\partial \boldsymbol{W}^{(l)}}$$

を計算し

$$\boldsymbol{W}^{(t+1,l)} = \boldsymbol{W}^{(t,l)} + \eta^{(t)}\Delta \boldsymbol{W}^{(t,l)}$$

と重みパラメータを更新する。行列での微分は、∇ によるベクトルの成分ごとでの微分を行ったように行列の成分ごとの微分を表している．ベクトルステップサイズ $\eta^{(t)}$ の選び方や更新アルゴリズムによっても挙動は大きく変化するため、様々な手法が提案されている。

8.3 計算機実装

本節では、具体的に前節の最適化アルゴリズムを計算機上で実装する。

8.3.1 Google Colaboratory

Google Colaboratory (略称: Colab) はインターネット環境と Google のアカウントさえ持っていれば、環境構築なしでブラウザから Jupyther という Python のインターフェイスを使うことができるサービスである。深層学習の際には不可欠となる GPU も含めて無料で使うことができる。ブラウザに Google アカウントでログインした後に、以下のサンプルファイルを開く。

`https://bit.ly/39JsDg8`

サンプルファイルの中に、実装方法や本節以降の実装例も含まれている。

本節では MNIST (Mixed National Institute of Standards and Technology database) を使った画像認識を演習する。

注意 8.3.1 Colaboratory を立ち上げると、ノートブックを指定することが求められるので、「アップロード」のタブを選択する。Colaboratory が使用するノートブックはログインしている Google Drive 上にある。また、右上の「共有」のボタンを押すことにより、ノートブックの共有設定を変えられる。その上で、ノートブックが表示されているブラウザ上の URL を伝えることにより、教員や TA にノートブックを見せることができる。Colab でノートブックを実行する際は、実行環境として GPU や TPU が接続された環境を利用することができ、本演習では、GPU を接続した環境での環境での利用を想定している (デフォルトでは CPU の設定になっている)。

8.3.2 レポートの作成

Colab 内では、コード形式とテキスト形式で入力ができる。特に、テキスト形式ではマークダウン記法ができるので、レポートとして提出する際は第 1 章を参考にしてほしい。Google Drive 上のノートブックをパソコンにダウンロードするには、Colab のファイルメニューで「.ipynb をダウンロード」を選択し、提出することができる。

Colab による課題は、サンプルファイルを参照すること。

注意 8.3.2 本演習では、最適化の違いまでは解説していないが、実装で用いる TensorFlow にはいくつか Optimizer が登録されている。AdamOptimizer と FtrlOptimizer は、比較的安定して精度の高い値を与えることが知られているので、本演習では基本的には AdamOptimizer を用いることとする。

8.3.3 NN による画像認識

手書き文字認識や画像認識は典型的な多クラス分類である。K クラスの分類は、各 $\boldsymbol{x}_i$ に対して $\boldsymbol{y}_i \in \{0, 1, \ldots, K\}$ が与えられていると考えれば良い。ここでは、$\hat{\boldsymbol{y}}(\boldsymbol{x}; \boldsymbol{w})$ の出力は K 次元確率ベクトル (要素が非負かつ総和が 1) であるとして、その k 番目の要素が最大であるとき、所属クラスを k と判定する。出力層の活性化関数にソフトマックス関数（$\mathbb{R}^K \to \mathbb{R}^K$）

$$\mathrm{softmax}_i(x_1, \ldots, x_K) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}},\ i = 1, \ldots, K$$

を用いれば、このような NN が設計できる。

課題 24 損失関数を”MSE”にしてクロスエントロピーとの結果を比較せよ。

課題 25 次の関数

$$\begin{aligned} f(x_1, x_2) =& 10 - 10\exp(-0.2(x_1^2 + x_2^2)) \\ & - \exp((\cos(x_1) + \cos(x_2))/2) \end{aligned}$$

に対して回帰問題を考える。n 個のデータ (x_{i1}, x_{i2}), $i = 1, \ldots, n$ からこの関数上の値 $y_i = f(x_{i1}, x_{i2}) + \varepsilon$ を生成し、データセット $\{(x_{i1}, x_{i2}, y_i)\}_{i=1}^n$ をもとに，NN による関数 $f_{NN}(x_1, x_2)$ を学習し、$(x_1, x_2, f_{NN}(x_1, x_2))$ を 3 次元プロットとして図示せよ。

課題 26 TF や keras のフレームワークを使わずに、numpy を用いて、勾配法と SGD での実装せよ。 データは MNIST を使い、計算時間や損失の挙動について報告せよ。

課題 27 ノードの数を $10, 10^2, 10^3, 10^4, 10^5$ の 5 つで比較せよ。

課題 28 多層 NN の層数、ノード、バッチの設定において、エポック 100 としてエポックを増やしたときにの学習の様子を報告せよ。

課題 29 Optimizer クラスに 6 つの Optimizer（GradientDescent、Adadelta、Adagrad、Adam、Ftrl、RMSProp）を使用することで比較せよ。

課題 30 CPU と GPU の場合の結果を比較せよ。

課題 31 損失関数を”MSE”にしてクロスエントロピーとの結果を比較せよ。

8.4 次元削減による特徴量抽出

前節までは入出力データ $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i \in \mathcal{B}}$ の関数回帰問題を念頭に、一般的に教師あり学習と呼ばれる問題に対する NN の議論を進めてきた。本節では s、（正解ラベルのついていない）データ集合 $\{\boldsymbol{x}_i\}_{i \in \mathcal{B}}$ の特徴量を次元削減により抽出する教師なし学習を考える。

8.4.1 オートエンコーダ

上記の目的のために、$\boldsymbol{y}_i = \boldsymbol{x}_i$ として前節と同様の関数回帰をおこなう。このように、$\hat{y}(\boldsymbol{x}_i) \approx \boldsymbol{x}_i$ を達成し、図 8.4.1 のように中間層のノード数が少ない「くびれた」構造をもつ NN をオートエンコーダ (Autoencoder; AE、自己符号化器) と呼ぶ。ここで、中間層の出力を潜在変数、入力から潜在変数への関数をエンコーダ、潜在変数から出力までの関数をデコーダと呼ぶ。与えられたデータの集合は、低次元の潜在変数をデコーダに入力することで再現できるという意味で、潜在変数をある種の特徴量と捉えることができる。

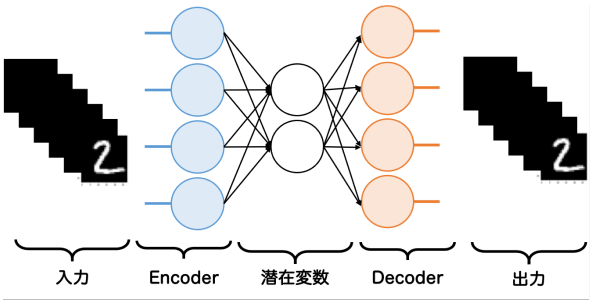


図 8.3: オートエンコーダのアーキテクチャ

8.4.2 手書き文字の特徴量抽出

MNIST から 0 と 1 の画像をそれぞれ 100 個ずつ取り出し、エンコーダ、潜在変数、デコーダ、出力の順に結合し、学習を行なった結果が図 8.5 である。学習を行った結果エンコーダにより特徴が圧縮され、デコーダによりもとのデータが復元できるのであれば潜在変数には圧縮された特徴が得られていることがわかる。

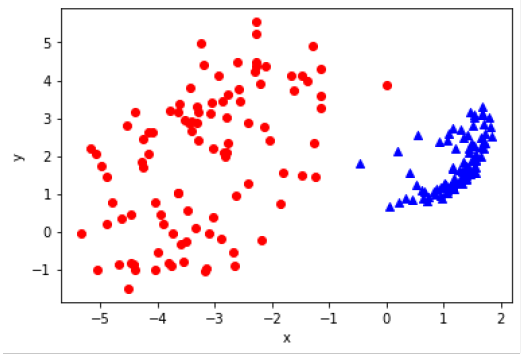


図 8.4: MNIST の 0 と 1 のそれぞれ 100 個のデータを AE で 2 次元に圧縮した散布図

課題 32 0, 1, 2 の 3 つの画像をそれぞれ 100 枚ずつ準備し、中間層で 2 次元まで圧縮し、散布図を表示せよ

8.5 生成モデル学習

前節の手書き文字の特徴量抽出の例において、図 8.5 の赤（青）の特徴量をデコーダに入力すると 0（1）に「近い」画像が出力される。一方で、これらの特徴量の分布に関しては、$\{\boldsymbol{x}_i\}_i$ をエンコーダに入力することで知ることができるものの、学習時には特に考慮していない。そこで、本節ではこうした元データや特徴量の分布を陽に考慮した生成モデル学習と呼ばれる手法を概観する。

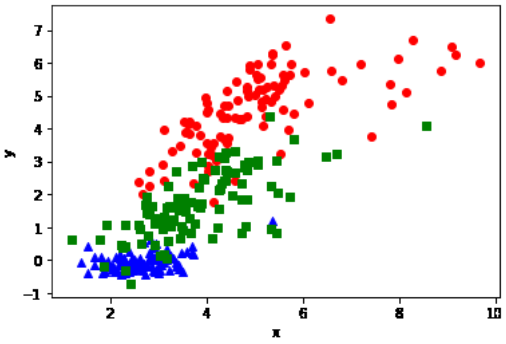


図 8.5: MNIST の 0 と 1 と 2 のそれぞれ 100 個のデータを AE で 2 次元に圧縮した散布図

8.5.1 変分オートエンコーダ

本節では、図 8.5.1 のような変分オートエンコーダ (Variational autoencorder; VAE) を紹介する。VAE での目的は潜在変数の分布が正規分布に従う（$\boldsymbol{z} \sim \mathcal{N}(0, I)$）ようなエンコーダ・デコーダを構成することである。準備として、確率密度関数 $q(x), p(x)$ に対してカルバック・ライブラー (Kullback-Leibler; KL) ダイバージェンスは次のように定義される。

$$D_{KL}(q||p) = \int q(x) \log(q(x)/p(x)) dx.$$

KL ダイバージェンスは 2 つの確率分布がどの程度似ているかを表す尺度である。エンコーダはデータ $\boldsymbol{x}$ に対して

$$\boldsymbol{z}(\boldsymbol{x}; \phi) = \mu(\boldsymbol{x}; \phi) + \sigma(\boldsymbol{x}; \phi)\varepsilon, \ \varepsilon \sim \mathcal{N}(0, I) \tag{8.14}$$

により、確率的に $\boldsymbol{z}$ を割り当てるとする。

デコーダの出力として生成されるデータの周辺尤度 $p_\theta(\boldsymbol{x}) = \int p(\boldsymbol{z}) p_\theta(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{z}$ が与えられたデー

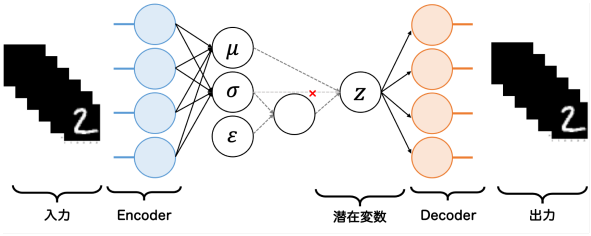


図 8.6: 変分オートエンコーダのアーキテクチャ

タの分布に一致するように、

$$\sum_i \log p_\theta(\boldsymbol{x}_i) \tag{8.15}$$

を最大化する θ が望ましい。ここで、推定困難な $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ の推定のために、その近似として $p_\phi(\boldsymbol{z}|\boldsymbol{x})$ を導入する。$\log p_\theta(\boldsymbol{x})$ に対して

$$\begin{aligned}\log p_\theta(\boldsymbol{x}) \geq & - D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) \\ & + \int \log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \log p_\theta(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{z} \\ & (= L_{ELBO}(\boldsymbol{x};\theta,\phi)) \end{aligned} \tag{8.16}$$

が任意の $\phi, \theta, \boldsymbol{x}$ に対して成り立つ。これを利用して、(8.15) の最大化の代わりに ELBO（Evidence Lower Bound）と呼ばれる L_{ELBO} の和

$$\sum_{i \in \mathcal{B}} L_{ELBO}(\boldsymbol{x}_i;\theta,\phi) \tag{8.17}$$

の最大化を考える。第 1 項は解析的に求めることができる。例えば、潜在変数が 1 次元で $f_\theta(z)$ に $N(0,1)$、$q_\phi(z|x)$ に $N(\mu,\sigma^2)$ を仮定すると、$D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) = (\sigma^2 + \mu^2 - \log\sigma^2 - 1)/2$ となる。一方で、再構成誤差に対応する第 2 項は積分計算が必要となるので、バッチ内のサンプルを用いてモンテカルロ法で近似される。例えば、二値画像の分類では、出力にベルヌーイ分布を仮定し、誤差関数にクロスエントロピー (8.9) とすれば良い。すなわち、$\int \log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \log p_\theta(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{z} = x \log \hat{y} + (1-x)\log(1-\hat{y})$ となる。以上の議論により、前節と同様の手法により θ, ϕ の最適化をおこなうことができる。学習後は、$\boldsymbol{z} \sim N(0,1)$ にしたがって、$\boldsymbol{z}$ のサンプルを生成し、それをデコーダに入力することでデータを生成できる。これが生成モデルと呼ばれる所以である。

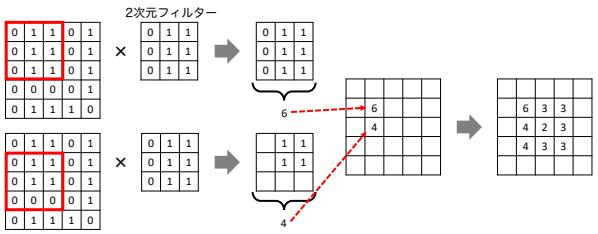


図 8.7: 5×5 の画像に対して 2 次元フィルターを適用した場合

課題 33 (8.16) を示せ。

課題 34 VAE を用いて 0, 1, 2 の 3 つの画像をそれぞれ 100 枚ずつ準備し、中間層で 2 次元まで圧縮し、散布図を表示せよ。

課題 35 CIFAR-10 に対して VAE を用いて画像を生成せよ。

8.6 (オプション) 畳み込みニューラルネットワーク

画像ソフトにおけるぼかしやエッジ抽出などに応用されている画像処理手法にフィルターがある。畳み込みニューラルネットワーク (CNN) は図 8.6 のように画像入力データに 2 次元フィルターを適用して畳み込み演算を行うことで、元の画像の情報を保持しつつ、サイズの小さな画像に変換する手法であり、多くの場面で性能を向上させることができる。CNN の性能を引き上げるための様々な手法が考案されているが、最も効果があるとされているのは畳み込み層や全結合層の間に挿入するプーリング層（最大プーリングや平均プーリング）である。図 8.6 からもわかるように、フィルターを適用した後は画像のサイズが小さくなり、連続して適用していくと画像が小さくなりすぎてしまう。このような場合の対策として、あらかじめ元の画像の周りを 0 で埋めておくことで元のサイズを維持するゼロパディングなどがある。

参考文献

[1] 瀧雅人，機械学習スタートアップシリーズ これならわかる深層学習入門，講談社，2017.

[2] Jakub Langr，Vladimir Bok，大和田茂，実践 GAN 敵対的生成ネットワークによる深層学習，マイナビ出版，2020.

[3] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning (Adaptive Computation and Machine Learning series) , The MIT Press, 2016.

[4] 中井悦司，TensorFlow で学ぶディープラーニング入門: 畳み込みニューラルネットワーク徹底解説，マイナビ出版，2019.

[中山、加嶋]