Python 入門

秋山進一郎
筑波大学計算科学研究センター

2025 年 10 月 31 日

目次

1　はじめに

例えば，10 行 10 列の二つ行列の積を計算したいとしよう．その計算方法は至ってシンプルであるが，実際に手で計算するのは大変骨が折れる．

$$(AB)_{ij} = \sum_{k=1}^{10} A_{ik} B_{kj} \tag{1.1}$$

式 (1.1) を見れば分かるように，10 行 10 列からなる行列同士の積を計算するためには，式 (1.1) の計算を ij に関して 100 回繰り返さなければならない．仮に，式 (1.1) の右辺で，それぞれの積 $A_{ik}B_{kj}$ を α 秒で，それらの k に関する和を β 秒で計算できたとしよう．計算に要する時間は $(\alpha \times 10 + \beta) \times 10 \times 10$ 秒だ．仮に $\alpha = \beta = 1$ でミスなく計算できた（！）とすると，18 分くらいで答えを得ることができる．もっと一般の行列積を考えて，M 行 K 列の行列と K 行 N 列の行列の積を計算するとしよう．この場合に必要な計算時間は $(\alpha \times K + \beta) \times M \times N$ 秒であり，M, N, K が大きい場合，計算時間は $O(MNK)$ でスケールすることになる．

この種の計算をやらねばならないというニーズが生じた場合，計算機は強力な助っ人になる．しかし，計算機は**機械語**という数字の羅列によって記述される言語しか理解することができない．行列積の計算を実行させたいだけなのに，機械語を一から学ばなければならない，と言われたら意気消沈するだろう．このギャップを埋めてくれるのが**プログラミング言語**である．プログラミング言語とは，人間が計算機に指示を与えるための言語であり，人間にとって分かりやすい言語体系になっている．プログラミング言語によって書かれた文章（プログラム）の実行方式は二種類に分けることができる．一つは，人間が書いたプログラムを上から一行ずつ機械語に翻訳しながら計算機に渡していくインタプリ型の言語である．このようなプログラミング言語は**スクリプト言語**と呼ばれ，Python はスクリプト言語の一例である．もう一つは，人間が書いたプログラムを最初から最後まで機械語に翻訳し切った後に計算機に一気に渡すコンパイル方式である．こうしたプログラミング言語は**コンパイル言語**と呼ばれ，C，C++，Fortran などが有名である．

2　Python の特徴

Python の特徴として，ここでは以下の三項目について説明する．

2.1 動的型付き言語

Python に限らず，プログラミング言語には**変数**と**型**という概念がある．変数とは，計算機上で再利用可能なように名前が付けられた“値”（より正確には，メモリ上の番地）のことであり，変数にはいくつかの型が存在する．Python で変数を扱う際には，変数の型を事前に宣言する必要がない．これは，Python が動的型付き言語だからである．具体例を見た方が早いので，まずはソースコード 1 を見てほしい．

ソースコード 1　整数 (int) 型

```
1  a = 1
```

これは，「a という変数を用意し，そこに 1 という値を代入せよ」という意味になる．この時，変数 a の型は**整数 (int) 型**になっている．ここで重要な点は，**a という変数に 1 という整数値が代入されたことで，a の型が整数 (int) 型に決まった**ということだ．[*1]

変数に小数点を含む数字を代入すると，その変数は**浮動小数点 (float) 型**になる（ソースコード 2）．

ソースコード 2　浮動小数点 (float) 型

```
1  b = 1.0
```

ダブルクォーテーションあるいはクォーテーションで囲まれた文字が代入された変数は，**文字列（str）型**になる（ソースコード 3）．

ソースコード 3　文字列（str）型

```
1  c = "string"
```

一般的に，動的型付き言語では変数の型宣言が不要なため，少ない記述量でコードが書けるが，実行中の処理速度は他の言語で書いた場合に比べて遅くなることも少なくない．

2.2 インデントによるコードブロックの表現

プログラムはいくつかの**文**から構成される．文はプログラムの実行単位であり，プログラムを実行すると，原則として**上の文から順番にプログラムが実行されていく**．Python では，インデント（行頭の字下げ）によって**コードブロック**を表現する．言い換えると，

[*1] 例えば，C や Fortran では変数の型を事前に宣言しないといけない．

Python ではプログラム上に書かれた文だけでなく，空白もプログラム上で意味を持つ．具体例として，以下のソースコード 4 を見てみよう．

ソースコード 4　インデントによるコードブロックの表現

```
1 if x > 0 :
2     print("Pro")
3 else:
4     print("Con")
```

Python では，この例のように，「キーワード + コロン（:）」でコードブロックが始まり，続くコードブロックはインデント（行頭の字下げ）で表される．インデントでコードブロックを表現する方法は**オフサイドルール**と呼ばれ，Python の大きな特徴の一つである．インデントは空白何文字でも構わないが，**スペース 4 つ**で入れるのが推奨されている．[*2] インデントそのものもプログラム上で意味を持つため，ソースコード 4 を下のように書くとエラーになる．

ソースコード 5　インデントに起因してエラーとなる例

```
1 if x > 0 :
2 print("Pro")
3 else:
4 print("Con")
```

2.3　豊富なライブラリ

Python は豊富な**ライブラリ**を有している．ライブラリとは，よく使う機能がパッケージ化されたものである．ライブラリを活用することで，自力でプログラムを書く手間が大幅に削減される．Python を使ったプログラミングでは，プログラムを書き始める前に自分がやりたい処理に使えそうなライブラリがないかインターネットで検索することから始めるのが良い．特に数値計算では，自分でプログラムを書くよりもライブラリを利用した方が計算の高速化に繋がることが多い．[*3] この授業でも，ライブラリは積極的に活用して

[*2] この他にも Python にはいくつかのコーディング規約が定められている．詳しくは `https://peps.python.org/pep-0020/` を参照．コーディング規約に従うに越したことはないが，初学の段階では過度に気にしなくても良い．自分の書いたプログラムを外部に公開するような際には気にしておいた方が良い．

[*3] 数値計算のアルゴリズムそのものに対する理解を深めるといった目的であれば，一度は自力でプログラムを書いてみるのが一番勉強になる．

いく．

なお，全学計算機システムに導入されているソフトウェアは原則として授業で使用されるものに限られており，管理者権限が必要なソフトウェアをユーザがインストールすることはできない．ただし，pip コマンドからインストール可能な Python のパッケージ（ライブラリ）についてはユーザディレクトリにインストールして利用しても良いことになっている．インストールの際は，pip install --user [package] を使うこと（全学計算機システムの Web サイト https://www.u.tsukuba.ac.jp/ufaq/python1/ 参照）．

3 型

3.1 数値型

数値型には，**整数（int）**，**浮動小数点（float）**，**複素数（complex）**の三種類がある．文字通り整数は整数型であり，小数点を含む数字は型になる．Python では複素数も扱うことができる．虚数単位には i ではなく j を使うので注意．例えば，$1 + 2i$ は 1+2j と書く．あるいは，虚数単位を書かずに，complex(1,2) と書くこともできる．複素数の実部は real，虚部は imag で取り出すことができる（ソースコード 6）．なお，複素数はその実部や虚部を整数で書いても浮動小数点型として扱われる．

ソースコード 6　複素数の実部，虚部を取り出す

```
1 (1+2j).real
2 (1+2j).complex
```

これらの数値型に対する主要な演算については表 1 を参照．以下にいくつかの注意点を列挙する．

- 同じ型同士の四則演算の結果は原則としてその型になる．
- ただし，整数同士の割り算（/）の結果は整数型ではなく，浮動小数点型になる．
- 整数型と浮動小数点型の演算結果は浮動小数点型になる．
- int() や float() で囲めば，文字列も数値型へ変換可能．
- **浮動小数点型の数値は内部的にはその数値に最も近い近似値を扱っているため誤差を持つ．**

表 1　主な演算

演算	結果	注
a + b	a と b の和	
a - b	a と b の差	
a * b	a と b の積	
a / b	a と b の商	a, b が int 型でも結果は float 型
a // b	a と b の商を切り下げたもの	complex 型には使えない
a % b	a/b の剰余	complex 型には使えない
a ** b	a の b 乗	
pow(a, b)	a の b 乗	
abs(a)	a の絶対値	
int(a)	a を int 型に変換	a が float 型なら小数点以下切り捨て
float(a)	a を float 型に変換	
complex(a, b)	実部 a，虚部 b の複素数	a+bj と同じ
a.conjugate()	複素数 a の共役複素数	

3.2　真偽値

真偽値（bool）とは，真（True）か偽（False）の二値だけを取る型である．真偽値は条件分岐などで用いられる．例えば，二つの数値を比較するとその結果が真偽値として与えられ，条件分岐やループの終了条件に用いられる．数値の比較には**比較演算子**を使う．代表的な比較演算子を表 2 にまとめておく．注意すべき点として，

- **浮動小数点数同士の等号比較は信頼できない．**

表 2　主な比較演算子

比較演算子	意味
`a < b`	a は b よりも小さい
`a <= b`	a は b 以下
`a > b`	a は b よりも大きい
`a >= b`	a は b 以上
`a == b`	a と b は等しい
`a != b`	a と b は等しくない

また，主要な bool 演算を表 3 にまとめておく．

表 3　主な bool 演算を表

演算	結果
`not x`	x が偽なら True，そうでなければ False
`x or y`	x が真なら x，そうでなければ y
`x and y`	x が偽なら x，そうでなければ y

4　Python の予約語

表 4 の単語は，プログラム中で変数として使ってはいけない．これらは**予約語**と呼ばれ，変数として使用した場合は SyntaxError になる．

表 4　Python の予約語

and	class	except	global	None	return
as	continue	exec	if	nonlocal	True
assert	def	False	import	not	try
async	del	finally	in	or	while
await	elif	for	is	pass	with
break	else	from	lambda	raise	yield

5　リストに対する処理

使用頻度の高いリストメソッドについては表 5 を見よ.

表 5　よく使うリストメソッド

メソッド	使用方法	注
append	aList.append(item)	リストの末尾に新しい要素 item を追加
insert	aList.insert(i,item)	リストの i 番目の位置に新しい要素 item を挿入
pop	aList.pop()	リストの最後尾の要素を削除
pop	aList.pop(i)	リストの i 番目の位置の要素を削除
sort	aList.sort(key=None, reverse=False)	リストの要素を小さい値から大きい値の順番に並び替える． key にソートルールを決める関数を指定できる．reverse が True の時，リストは小さい値から大きい値へソートする.
reverse	aList.reverse()	リストを逆順にする
count	aList.count(item)	要素 item の個数を数える

6　matplotlib による作図

matplotlib の最も基本的な使い方をソースコード 7 に示す．ここでは，Python で数値計算を行う際に広く使われている Numpy ライブラリと併用している.

ソースコード 7　matplotlib の最も基本的な使い方

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 x = np.arange(-np.pi, np.pi, 0.1)
5 y = np.sin(x)
6
7 plt.plot(x, y, "-ob")
8
9 plt.savefig("sinx.pdf")
```

ソースコード 7 の出力結果を図 1 に示す．ソースコード 7 の 7 行目の「`-ob`」について，「`-`」は実線，「`o`」は o の形のマーカー，「`b`」は青色を意味する．線の種類，マーカーの種類，色を指定する順番は任意である．つまり，「`o-b`」や「`bo-`」のように書いても良い．いくつかの代表的な線，マーカー，色の指定方法をそれぞれ表 6，7，8 に示す．

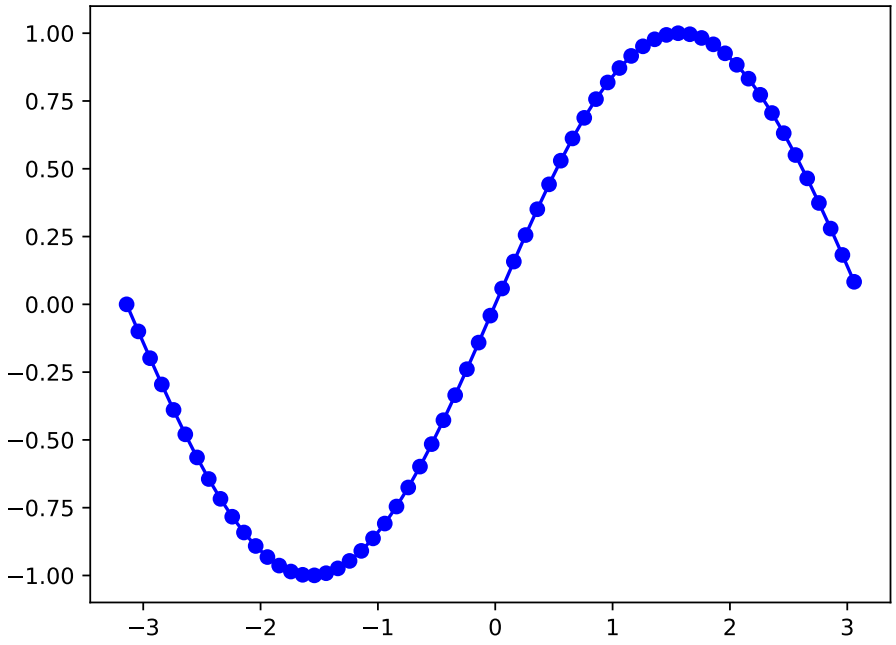

図 1　ソースコード 7 の出力結果

表 6　主な線の種類

スクリプト上での記号	線の種類
-	実線
--	破線
-.	一点鎖線
:	点線

表 7　主なマーカーの種類

スクリプト上での記号	線の種類
.	点
,	ピクセル
o	丸
^	三角
v	下三角
<	左三角
>	右三角
s	四角
d	菱形
+	＋
x	×

表 8　主な色の種類

スクリプト上での記号	色
r	赤
g	緑
b	青
c	シアン
m	マゼンタ
y	黄
k	黒
w	白