プログラミング基礎演習 中間レポート

340481H　電子情報工学科内定　中里徳彦

2015/01/06

1　導入

多項式の項の次数が必ず非負の整数となることを利用し、多項式をベクトルのように見立て課題 1-4 を解いた。

2　手法

多項式は $a_0x^n + a_1x^{n-1} + \cdots + a_{n-1}x + a_n$ の形の代数的表現である。ここで $a_i(i = 0, 1, \ldots, n)$ は複素数で n は非負整数である [1]。したがって、n 次の多項式は n+1 次の

$$\boldsymbol{a} = \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_1 \\ a_0 \end{pmatrix} \tag{1}$$

というベクトルで表すことができる（多項式の添字に合わせたためベクトルの普通の添字とは順序が逆転している）。今回の課題では多項式の操作をこのベクトルに対する操作として考え実装した。ただし、$a_i(i = 0, 1, \ldots, n)$ は実数に限定する。上記のベクトルを構造体 Polynomial として実装した。この構造体は多項式の最大次数は int n、係数は double *a である。

課題を解く前に、多項式の定数倍、加算、減算、乗算を定義した。多項式の定数倍、加算は通常のベクトルの定数倍、加算と似たように

$$\boldsymbol{b} = c\boldsymbol{a} = \begin{pmatrix} ca_n \\ ca_{n-1} \\ \vdots \\ ca_1 \\ ca_0 \end{pmatrix} \tag{2}$$

$$\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{b} = \begin{pmatrix} a_n + b_n \\ a_{n-1} + b_{n-1} \\ \vdots \\ a_1 + b_1 \\ a_0 + b_0 \end{pmatrix} \tag{3}$$

として表すことができる。ただし c は実数。ただし、$\boldsymbol{a}$ と $\boldsymbol{b}$ の次数が一致しない場合にも加算が出来るように

定義する。例えば $\boldsymbol{b}$ の次数が $l < n$ だった場合、

$$\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{b} = \begin{pmatrix} a_n + b_l \\ a_{n-1} + b_{l-1} \\ \vdots \\ a_{n-l} + b_0 \\ a_{n-l-1} \\ \vdots \\ a_1 \\ a_0 \end{pmatrix} \tag{4}$$

とする。$\boldsymbol{a}$ の次数のほうが低かった場合も同様である。また計算の結果最高次数に対応する要素が 0 だった場合、最高次数の要素が 0 以外になるまで、ベクトルの次数を小さくするようにした。つまり、

$$\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{b} = \begin{pmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_1 \\ c_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_1 \\ c_0 \end{pmatrix} \tag{5}$$

という操作を加算の最後に行なった。ただし、係数が浮動小数点数であるため、係数が正確に 0 にならないこともあるので、係数が ERROR（このプログラムでは 0.0001）以下の時は 0 であるとみなした。減算は定数倍と加算を組合せて

$$\boldsymbol{c} = \boldsymbol{a} - \boldsymbol{b} = \boldsymbol{a} + (-1)\boldsymbol{b} \tag{6}$$

と考えた。乗算はベクトルや行列に定義されている通常の演算（内積、外積など）を用いて定義するのが困難であるため、多項式の乗算の定義に素直に従って

$$\boldsymbol{c} = \begin{pmatrix} a_m b_n \\ a_{m-1} b_n + a_m b_{n-1} \\ \vdots \\ a_0 b_1 + a_1 b_0 \\ a_0 b_0 \end{pmatrix} \tag{7}$$

として定義した。ただし $\boldsymbol{c}$ は $m + n + 1$ 次のベクトルである。

また、多項式の x^n 乗も実装した。n は整数である。これはベクトルで表現した多項式において、各要素を下に n 個シフトさせることに等しい。ただしその時ベクトルの次数も n 増える。$n < 0$ の場合は、ベクトルの次数を n 減らし、元々のベクトルの上 n 個は消去するようにした。要素が全て消去されてしまう場合は、NULL がかえってくるようにした。

2.1 課題 1、2

単項式は多項式の一部である。よって課題 1 と課題 2 は同一の課題とみなすことができるため同じセクションにまとめる。$a_0x^n + a_1x^{n-1} + \cdots + a_n$ を微分するために

$$\mathbf{a} = \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_1 \\ a_0 \end{pmatrix} \qquad \text{元のベクトル} \tag{8}$$

$$\rightarrow \begin{pmatrix} a_n \\ 1\ a_{n-1} \\ \vdots \\ (n-1)a_1 \\ na_0 \end{pmatrix} \qquad \text{次数倍する} \tag{9}$$

$$\rightarrow \begin{pmatrix} 1\ a_{n-1} \\ \vdots \\ (n-1)a_1 \\ na_0 \\ 0 \end{pmatrix} \qquad \text{各要素を一つ上にシフトする} \tag{10}$$

$$\rightarrow \begin{pmatrix} 1\ a_{n-1} \\ \vdots \\ (n-1)a_1 \\ na_0 \end{pmatrix} \qquad \text{最高次数を消去する} \tag{11}$$

とした。ただし、元の多項式が定数だった場合、ベクトルが NULL として帰ってくる。そのため、NULL とした場合は元の多項式が定数だったと考え、ベクトルとしての次数が 1 の $\mathbf{0}$ を計算結果とした。

2.2 課題 3

割られる多項式に対応する n 次ベクトルを $\boldsymbol{a}$、割る多項式に対応する m 次ベクトルをを $\boldsymbol{b}$、商の n-m 次ベクトルを $\boldsymbol{q}$、余りの m-1 次ベクトルを $\boldsymbol{r}$ とする。筆算での方法に対応して商と余りを求めると次のようにな

る。まず q_0 は a_0/b_0 で求まる。次に、

$$\boldsymbol{c} = \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{m+1} \\ a_m \\ a_{m-1} \\ \vdots \\ a_1 \\ a_0 \end{pmatrix} - q_0 \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ b_m \\ b_{m-1} \\ \vdots \\ b_1 \\ b_0 \end{pmatrix} \qquad \boldsymbol{b} \text{をシフトして引き算} \tag{12}$$

$$= \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{m+1} \\ a_m - q_0 b_m \\ a_{m-1} - q_0 b_{m-1} \\ \vdots \\ a_1 - q_0 b_1 \\ 0 \end{pmatrix} \tag{13}$$

$$= \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{m+1} \\ a_m - q_0 b_m \\ a_{m-1} - q_0 b_{m-1} \\ \vdots \\ a_1 - q_0 b_1 \end{pmatrix} \qquad \text{最高次数を削除} \tag{14}$$

$$= \begin{pmatrix} c_{n-1} \\ c_{n-2} \\ \vdots \\ c_0 \end{pmatrix} \tag{15}$$

と計算する。q_1 は c_0/b_0 で求まる。同様の計算を q_{n-m} が求まるまで行うと、商が導出できる。

余りは

$$\boldsymbol{r} = \boldsymbol{a} - \boldsymbol{bq} \tag{16}$$

であることから求めた。

2.3 課題 4

ユークリッドの互除法を用いた [2]。n 次方程式を表すベクトルを $\boldsymbol{a}$、$m(<n)$ 次方程式を表すベクトル $\boldsymbol{b}$ とするとユークリッドの互除法は次のような操作となる。

$$\begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{m+1} \\ a_m \\ a_{m-1} \\ \vdots \\ a_1 \\ a_0 \end{pmatrix} - (a_0/b_0) \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ b_m \\ b_{m-1} \\ \vdots \\ b_1 \\ b_0 \end{pmatrix} \qquad \boldsymbol{b} \text{をシフトして引き算} \tag{17}$$

$$= \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{m+1} \\ a_m - (a_0/b_0)b_m \\ a_{m-1} - (a_0/b_0)b_{m-1} \\ \vdots \\ a_1 - (a_0/b_0)b_1 \\ 0 \end{pmatrix} \tag{18}$$

$$\rightarrow \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{m+1} \\ a_m - (a_0/b_0)b_m \\ a_{m-1} - (a_0/b_0)b_{m-1} \\ \vdots \\ a_1 - (a_0/b_0)b_1 \end{pmatrix} \qquad \text{最大次数が } 0 \text{ にならないよう調整} \tag{19}$$

(20)

このようにして求めたベクトルと $\boldsymbol{b}$ を用いてこの操作を繰り返し、$\boldsymbol{0}$ が生じたら、その時点で操作をやめるともう一方のベクトルは最大公約元になっている。

3 結果

3.1 課題 1、2

単項式を引数とすると次のようになる。

```
$./kadai1 "10*x^3"
diff +0.0 +0.0*x +30.0*x^2
```

定数を引数とすると次のように 0 が出力される。

```
$ ./kadai1 "2"
diff +0.0
```

多項式の場合は次のように同類項がまとめられて出力される。

```
$ ./kadai2 "10*x^3-32*x^5+3*x^3+5"
diff +0.0 +0.0*x +39.0*x^2 +0.0*x^3 -160.0*x^4
```

3.2 課題 3

次のように出力される。

```
$ ./kadai3 "x^2+7*x+3" "x+1"
Quotient +6.0 +1.0*x
Remainder -3.0
```

```
$ ./kadai3 "x^4+x+1" "x^2+1"
Quotient -1.0 -0.0*x +1.0*x^2
Remainder +2.0 +1.0*x
```

また引く数が 0 のときは次のように出力される。

```
$ ./kadai3 "x^4+x+1" "0"
argv[2] == "0"
NULL
argv[2] == "0"
NULL
Quotient Remainder
```

3.3 課題 4

```
$ ./kadai4 "x^6+3*x^5+x^3+2*x^2-4*x-3" "x^3+5*x^2+3*x-9"
GCD -3.0 +2.0*x +1.0*x^2
```

4 考察

今回は係数に倍精度浮動小数点数を用いたが、その場合どうしても誤差が出てしまう。また循環小数を扱うことができないという欠点も存在する。係数に分数を用いることでそれらの欠点をなくすことできると考えられる。

参考文献

[1] 一松　信　伊藤雄二　監訳　『数学辞典』(朝倉書店，1993)

[2] 多項式の基本 (2015/1/6 アクセス)
http://www.math.nagoya-u.ac.jp/ shinichi/K105.pdf