数値解析レポート No.1

32 番 平田 蓮

1　任意の行列の 2 行を交換する

作成した行列の任意の 2 行を交換する関数 `swap_row()` をリスト 1 に示す.

リスト 1　swap.c

```
void swap_row(double **a, int row, int col, int x, int y) {
    if (x < 0 || x >= row || y < 0 || y >= row) {
        puts("Defective␣Input␣(swap_row())");
        return;
    }

    double *tmp = a[x];
    a[x] = a[y];
    a[y] = tmp;
}
```

引数の `a` には NAbasic.c の `csvRead()` から得られる行列を渡す. `row`, `col`, `x`, `y` はそれぞれ行列の行数, 列数, 交換する行である. 正方行列を掛け合わせることで行を交換することも可能だが, それだと正方行列以外の行列に対応できないため, リスト 1 のように 2 行のポインタを入れ替えることで実装した.

この関数を使って課題 4 の $A = \begin{pmatrix} 2 & -1 & 5 \\ -4 & 2 & 1 \\ 8 & 2 & -1 \end{pmatrix}$ の 1, 3 行目を交換した際の出力を下に示す.

```
Before swapping
2.000000 -1.000000 5.000000
-4.000000 2.000000 1.000000
8.000000 2.000000 -1.000000

After swapping
8.000000 2.000000 -1.000000
-4.000000 2.000000 1.000000
2.000000 -1.000000 5.000000
```

正しく交換できていることがわかる.

2　行列積を利用して内積を求める

2 つのベクトル $\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$ の内積は $\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = x_1 y_1 + x_2 y_2 + x_3 y_3$ と表される．これは次の行列積 $\begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$ で表すことができる.

行列積を計算する関数 `matrix_p()` をリスト 2 に示す.

リスト 2　prod.c

```
double **matrix_p(
    double **x, int xrow, int xcol,
    double **y, int yrow, int ycol
) {
    int i, j, k;
    double **ret = allocMatrix(xrow, ycol);

    if (xrow != ycol || xcol != yrow) {
        puts("Defective␣data!␣(matrix_p())");
        return ret;
    }

    for (i = 0; i < xrow; i++) {
        for (j = 0; j < ycol; j++) {
            for (k = 0; k < xcol; k++) {
                ret[i][j] += x[i][k] * y[k][j];
            }
        }
    }

    return ret;
}
```

引数は計算をする 2 つの行列とそれぞれの行数と列数である.

2 つのベクトルを読み込む際に, 行ベクトルと列ベクトルとして読み込まなければ計算ができない.

この関数を使用して課題 2 の 2 つのベクトルの内積を求めた結果を下に示す.

```
Vector1
1.000000 5.000000 8.000000

Vector2
3.000000
7.000000
```

-2.000000

Inner product: 22.000000

正しく計算できていることがわかる.

3　2 つのベクトルの成す角を求める

2 つのベクトル $\boldsymbol{v_1}, \boldsymbol{v_2}$ の内積は 2 つの成す角を θ としたとき, $\boldsymbol{v_1} \cdot \boldsymbol{v_2} = v_1 v_2 \cos\theta$ と定義される. よって, 2 つの成す角は, $\theta = \arccos \dfrac{\boldsymbol{v_1} \cdot \boldsymbol{v_2}}{v_1 v_2}$ として求めることができる. これを求める関数 `calc_angle()` をリスト 3 に示す.

リスト 3　calc_angle.c

```
double calc_angle(
    double **x, int xrow, int xcol,
    double **y, int yrow, int ycol
) {
    if (xrow != 1 || ycol != 1 || xcol != yrow) {
        puts("Defective␣data!␣(calc_angle())");
        return -1;
    }

    double p = matrix_p(x, y, xrow, xcol, yrow, ycol)[0][0];
    return acosf(
        p /
        norm(matrix2colVector(x, xrow, xcol), xrow * xcol) /
        norm(matrix2colVector(y, yrow, ycol), yrow * ycol)
    );
}
```

引数は `matrix_p()` と同様である.

これを使用して課題 2 の 2 つのベクトルの成す角を求めた結果を下に示す.

Vector1
1.000000 5.000000 8.000000

Vector2
3.000000
7.000000
-2.000000

Inner product: 22.000000

```
Angle:  1.271850 [rad]
       72.871616 [°]
```

わかりやすいように孤度法と度数法 2 通りで出力するようにした.

4 感想

課題の内容は比較的簡単に熟すことができた. 2 本のベクトルの成す角について, グラフにプロットをして実際の角度を確かめようとしたが, 3 次元プロットではわかりづらかったため, ここには載せなかった.