固有値と固有ベクトル： QL 法

固有値と固有ベクトルを QL 法によって求めるプログラム例をリスト１に示す。

リスト１　QL 法の利用例。

```
#include "stdafx.h"
#include <iostream>
#include "mymat.h"
#include "myEigen.h"
#include "smplRN.h"

using namespace System;
using namespace std;
using namespace smplRN;
using namespace myLib;

int main(array<System::String ^> ^args)
{
	const int n = 4;

	double u1[n] = { 1, 1, 1, 1 };
	double u2[n] = { 1, -1, 1, -1 };
	double u3[n] = { 1, 1, -1, -1 };
	double u4[n] = { 1, -1, -1, 1 };

	myMat a(n,n);
	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++)
			a[i][j] = 4*u1[i]*u1[j] + 3*u2[i]*u2[j] +
				2*u3[i]*u3[j] + u4[i]*u4[j];

	cout << "a = " << endl;
	WriteMat( 9, a );

	myMat StoreA(n,n);
	StoreA = a;

	myMat EigenVec(n,n);
	myMat EigenVal(n,1);
	int n_eigens;
	QL_decom ql;
	ql.DoQL( a, EigenVec, EigenVal, n_eigens );

	cout << "Eigen values and Eigen vectors..." << endl;
	for (int i = 0; i < n; i++){
		cout << "Eigen Value = " << EigenVal[i][0] << endl;
		for (int j = 0; j < n; j++)
			cout << EigenVec[j][i] << endl;
	}
```

```
        myMat CkA(n,n);
        for (int i = 0; i < n; i++)
                for (int j = 0; j < n; j++){
                        double tv = 0.0;
                        for (int k = 0; k < n; k++)
                            tv += EigenVal[k][0] * EigenVec[i][k] * EigenVec[j][k];
                        CkA[i][j] = tv;
                }
        cout << "a =" << endl;
        WriteMat( StoreA );
        cout << "a(constructed) =" << endl;
        WriteMat( CkA );

        Console::WriteLine("Enterキーを押して終了");
        Console::ReadLine();
        return 0;
}
```

リスト１では、QL 法による固有分解をヘッダーファイル myEigen.h に宣言されているクラス型 QL_decom によって行っている。行列 a の固有分解は

```
QL_decom ql;
ql.DoQL( a, EigenVec, EigenVal, n_eigens );
```

と実行すると、EigenVec, EigenVal, n_eigens にそれぞれ固有ベクトル、固有値、０でない固有値の数が返される。

リスト１において#include の対象となっているヘッダーファイル mymat.h、myEigen.h および smplRN.h を利用するときは、それらのファイルで宣言されているものの定義が置かれているファイル mymat.cpp、myEigen.cpp および smplRN.cpp も他の*.cpp ファイルや*.h ファイルと同じフォルダにコピーして、メニュ「プロジェクト｜既存項目の追加」により追加しておく。ヘッダーファイルは#include により挿入されているが、*.cpp ファイルはこの「プロジェクト｜既存項目の追加」によって追加しておかないとビルド時にエラーとなる。

リスト１のプログラムの実行例を図１に示す。

```
a =
        10         2         4         0
         2        10         0         4
         4         0        10         2
         0         4         2        10
Eigen values and Eigen vectors...
Eigen Value = 16
0.5
0.5
0.5
0.5
Eigen Value = 12
```

```
-0.5
0.5
-0.5
0.5
Eigen Value = 8
0.5
0.5
-0.5
-0.5
Eigen Value = 4
-0.5
0.5
0.5
-0.5
a =
   10    2    4    0
   2    10    0    4
   4    0    10    2
   0    4    2    10
a(constructed) =
   10    2    4    0
   2    10    -5.55112e-016    4
   4    -5.55112e-016    10    2
   0    4    2    10
Enter キーを押して終了
```

図１　リスト１のプログラムの実行例。

固有分解の対象である行列は

$$\mathbf{u1} = (1 \quad 1 \quad 1 \quad 1),$$
$$\mathbf{u2} = (1 \quad -1 \quad 1 \quad -1),$$
$$\mathbf{u3} = (1 \quad 1 \quad -1 \quad -1),$$
$$\mathbf{u4} = (1 \quad -1 \quad -1 \quad 1),$$

$$\mathbf{v1} = \frac{1}{2}\mathbf{u1}, \quad \mathbf{v2} = \frac{1}{2}\mathbf{u2}, \quad \mathbf{v3} = \frac{1}{2}\mathbf{u3}, \quad \mathbf{v4} = \frac{1}{2}\mathbf{u4}$$

とおくとき、次式

$$\mathrm{a} = 4 \times \mathbf{u1'u1} + 3 \times \mathbf{u2'u2} + 2 \times \mathbf{u3'u3} + 1 \times \mathbf{u4'u4}$$
$$= 16 \times \mathbf{v1'v1} + 12 \times \mathbf{v2'v2} + 8 \times \mathbf{v3'v3} + 4 \times \mathbf{v4'v4}$$

で与えられるものである。固有値 16，12，8，4 と対応する固有ベクトル **v1**、**v2**、**v3**、**v4** が求められていることが図１よりわかる。