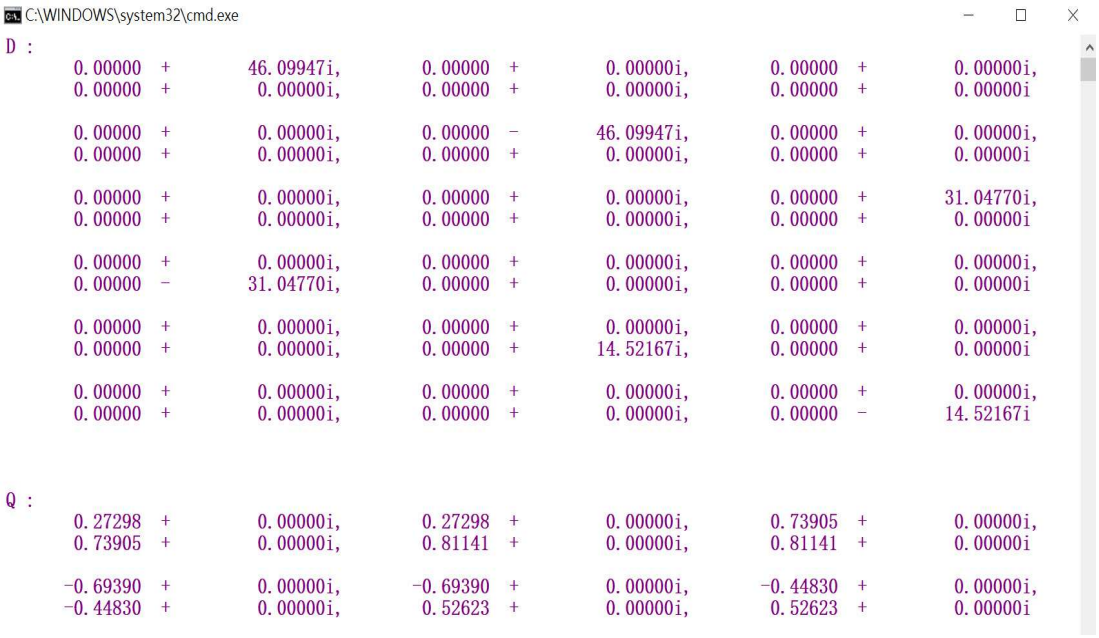
實例一 ：

```
using System;
using ClassLibrary1;
using ClassLibrary1A;

// 參考Clough and Penzien, "Dynamics of Structures"  P202 - P203
namespace ConsoleApp10
{
    class Program
    {
        static void Main(string[] args)
        {
            double[,] M = { {1, 0, 0}, {0, 1.5, 0}, {0, 0, 2.0} };
            double[,] C = { { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };
            double[,] K = { { 600, -600, 0 }, { -600, 1800, -1200 },
                { 0, -1200, 3000 } };
            MKCMatrix mkc = new MKCMatrix(M, K, C);
            ReMatrix A = mkc.Matrix;
            EIG eig = new EIG(A);
            CxMatrix D = eig.CxMatrixD;
            CxMatrix Q = eig.CxMatrixQ;
            Console.Write("\n D : \n{0}", new PR(D));
            Console.Write("\n Q : \n{0}", new PR(Q));
        }
    }
}

/* 以下是Visual Studio計算輸出結果 :
 D :
 0.00000 + 46.09947i,  0.00000 +  0.00000i,  0.00000 +  0.00000i,
 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 +  0.00000i

 0.00000 +  0.00000i,  0.00000 - 46.09947i,  0.00000 +  0.00000i,
 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 +  0.00000i

 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 + 31.04770i,
 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 +  0.00000i
```

```
 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 +  0.00000i,
 0.00000 - 31.04770i,  0.00000 +  0.00000i,  0.00000 +  0.00000i

 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 +  0.00000i,
 0.00000 +  0.00000i,  0.00000 + 14.52167i,  0.00000 +  0.00000i

 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 +  0.00000i,
 0.00000 +  0.00000i,  0.00000 +  0.00000i,  0.00000 - 14.52167i

 Q :
 0.27298 +  0.00000i,  0.27298 +  0.00000i,  0.73905 +  0.00000i,
 0.73905 +  0.00000i,  0.81141 +  0.00000i,  0.81141 +  0.00000i

-0.69390 +  0.00000i, -0.69390 +  0.00000i, -0.44830 +  0.00000i,
-0.44830 +  0.00000i,  0.52623 +  0.00000i,  0.52623 +  0.00000i

 0.66597 +  0.00000i,  0.66597 +  0.00000i, -0.50180 +  0.00000i,
-0.50180 +  0.00000i,  0.24492 +  0.00000i,  0.24492 +  0.00000i

 0.00000 -  0.00592i,  0.00000 +  0.00592i,  0.00000 -  0.02380i,
 0.00000 +  0.02380i,  0.00000 -  0.05588i,  0.00000 +  0.05588i

 0.00000 +  0.01505i,  0.00000 -  0.01505i,  0.00000 +  0.01444i,
 0.00000 -  0.01444i,  0.00000 -  0.03624i,  0.00000 +  0.03624i

 0.00000 -  0.01445i,  0.00000 +  0.01445i,  0.00000 +  0.01616i,
 0.00000 -  0.01616i,  0.00000 -  0.01687i,  0.00000 +  0.01687i
*/
```

上式中，D是複數特徵值(矩陣)，Q是複數特徵向量(矩陣)。另以下是執行後之實際的影像 ：



實例二 ：

```
// 請參考J. L. Humar, "Dynamics of Structures" 第502-504頁。

using System;
using ClassLibrary1;
using ClassLibrary1A;

namespace ConsoleApp6B
{
    class Program
    {
        static void Main(string[] args)
        {

double[,] M = { { 2, 0 }, { 0, 1 } };
double[,] K = { { 3, -1 }, { -1, 1 } };
double[,] C = { { 0.4, -0.05 }, { -0.05, 0.2 } };
double[,] y0Start = { {0}, {0}, {1}, {2} };
double[,] yp0Start = { {0}, {0}, {0}, {0} };

ReMatrix y0 = new ReMatrix(y0Start);
ReMatrix yp0 = new ReMatrix(yp0Start);
```

```
MKCMatrix MKC = new MKCMatrix(M, K, C);
ReMatrix A = new ReMatrix(MKC.Matrix);
EIG eig = new EIG(A);
CxMatrix D = eig.CxMatrixD;
CxMatrix Q = eig.CxMatrixQ;
CxToDexp Dexp0 = new CxToDexp(D, 0);
CxMatrix DexpTime0 = Dexp0.GetCxMatrix;
CxMatrix d = ~DexpTime0 * ~Q * (y0 - yp0);

double step = 0.5;
int iRow = (int)(50 / step + 1);
int iCol = M.GetLength(1) + 1;
ReMatrix Displacement = new ReMatrix(iRow, iCol);
ReMatrix Velocity = new ReMatrix(iRow, iCol);
ReMatrix Acceleration = new ReMatrix(iRow, iCol);

for (int i = 0; i != iRow; i++)
{
    double t = step * i;

    double yp_0 = 0; double yp_1 = 0;
    double yp_dot_0 = 0; double yp_dot_1 = 0;
    double yp_2dot_0 = 0; double yp_2dot_1 = 0;

    CxToDexp Dexp = new CxToDexp(D, t);
    CxMatrix DexpTime = Dexp.GetCxMatrix;
    CxMatrix yh_Cx = Q * DexpTime * d;
    ReMatrix yh_Re = (ReMatrix) yh_Cx;

    Velocity.Matrix[i, 0] = t;
    Velocity.Matrix[i, 1] = yh_Re.Matrix[0, 0] + yp_dot_0;
    Velocity.Matrix[i, 2] = yh_Re.Matrix[1, 0] + yp_dot_1;
    Displacement.Matrix[i, 0] = t;
    Displacement.Matrix[i, 1] = yh_Re.Matrix[2, 0] + yp_0;
    Displacement.Matrix[i, 2] = yh_Re.Matrix[3, 0] + yp_1;

    CxMatrix yhDot_Cx = A * yh_Cx;
    ReMatrix yhDot_Re = (ReMatrix)yhDot_Cx;
```

```
        Acceleration.Matrix[i, 0] = t;
        Acceleration.Matrix[i, 1] = yhDot_Re.Matrix[0, 0] + yp_2dot_0;
        Acceleration.Matrix[i, 2] = yhDot_Re.Matrix[1, 0] + yp_2dot_1;
    }

    // 列印模態參數。
    Console.Write("\n***{0,12}模{0,7}態{0,7}參{0,7}數{0,12}***\n", "");
    Console.Write("\n***{0,5}特徵值矩陣D{0,5}***\n{1}\n","",new PR(D));
    Console.Write("\n***{0,5}特徵向量矩陣Q{0,5}***\n{1}\n","",new PR(Q));
    Console.Write("\n***{0,5}係數向量d{0,5}***\n{1}\n","",new PR(d));

    // 列印節點的變位，速度，和加速。
    Console.Write("\n{0,5}***位移反應量***{0,5}\n{0,8}時間(秒)" +
        "{0,8}第0點位移{0,8}第1點位移\n\n{1}", "", new PR(Displacement));
    Console.Write("\n{0,5}***速度反應量***{0,5}\n{0,8}時間(秒)" +
        "{0,8}第0點速度{0,8}第1點速度\n\n{1}", "", new PR(Velocity));
    Console.Write("\n***{0,5}加速度反應量{0,5}***\n{0,8}時間(秒)" +
        "{0,8}第0點加速度{0,7}第1點加速度\n\n{1}","", new
    PR(Acceleration));

            }
        }

    }
```

以下是列印輸出結果。

```
/* 輸出結果:
***                模          態          參          數                ***
***        特徵值矩陣 D        ***
 -0.11666 + 1.40933i,   0.00000 + 0.00000i,   0.00000 + 0.00000i,   0.00000 + 0.00000i
  0.00000 + 0.00000i,  -0.11666 - 1.40933i,   0.00000 + 0.00000i,   0.00000 + 0.00000i
  0.00000 + 0.00000i,   0.00000 + 0.00000i,  -0.08334 + 0.70221i,   0.00000 + 0.00000i
  0.00000 + 0.00000i,   0.00000 + 0.00000i,   0.00000 + 0.00000i,  -0.08334 - 0.70221i

***        特徵向量矩陣 Q        ***
   0.57745 + 0.00000i,   0.57745 + 0.00000i,   0.25800 + 0.00000i,   0.25800 + 0.00000i
  -0.57707 - 0.01365i,  -0.57707 + 0.01365i,   0.51648 - 0.00609i,   0.51648 + 0.00609i
  -0.03369 - 0.40695i,  -0.03369 + 0.40695i,  -0.04300 - 0.36231i,  -0.04300 + 0.36231i
```

0.02405 + 0.40748i, 0.02405 - 0.40748i, -0.09463 - 0.72428i, -0.09463 + 0.72428i

*** 係數向量 d ***

0.00484 + 0.00006i

0.00484 - 0.00006i

-0.01084 + 1.37914i

-0.01084 - 1.37914i

位移反應量

時間(秒)	第 0 點位移	第 1 點位移
0.00000	1.00000	2.00000
0.50000	0.93967	1.87981
1.00000	0.77181	1.54694
.....		
49.00000	-0.01639	-0.03270
49.50000	-0.01619	-0.03246
50.00000	-0.01408	-0.02838

速度反應量

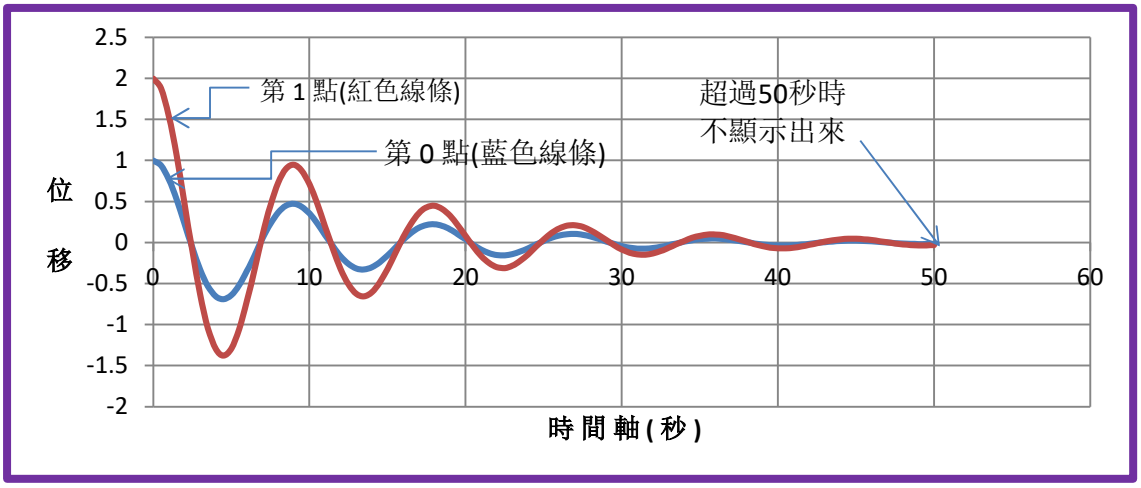
時間(秒)	第 0 點速度	第 1 點速度
0.00000	0.00000	0.00000
0.50000	-0.23583	-0.46888
1.00000	-0.42608	-0.84336
.....		
49.00000	-0.00167	-0.00368
49.50000	0.00241	0.00452
50.00000	0.00587	0.01152

*** 加速度反應量 ***

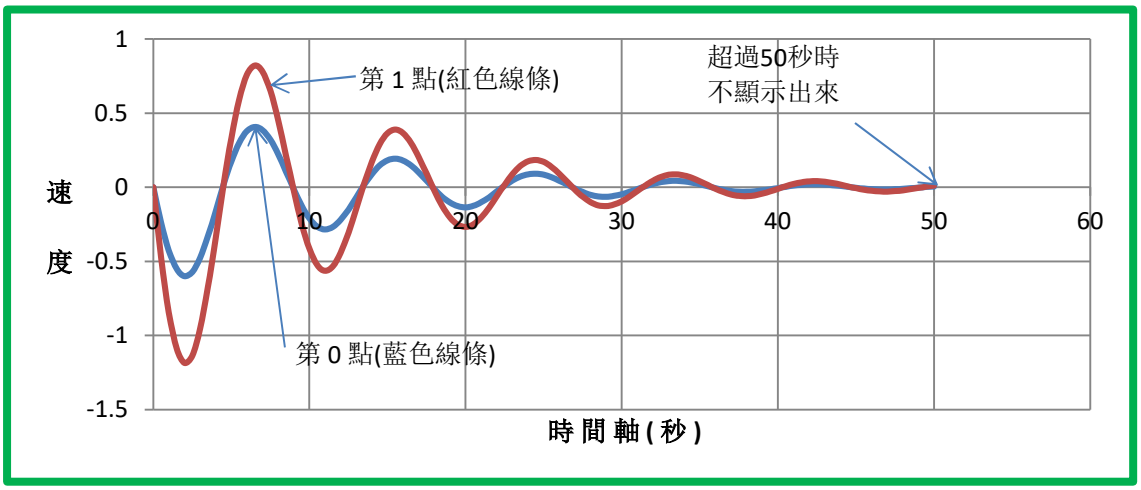
時間(秒)	第 0 點加速度	第 1 點加速度
0.00000	-0.50000	-1.00000
0.50000	-0.43415	-0.85816
1.00000	-0.32011	-0.62777
.....		
49.00000	0.00847	0.01696
49.50000	0.00768	0.01549
50.00000	0.00605	0.01229

*/

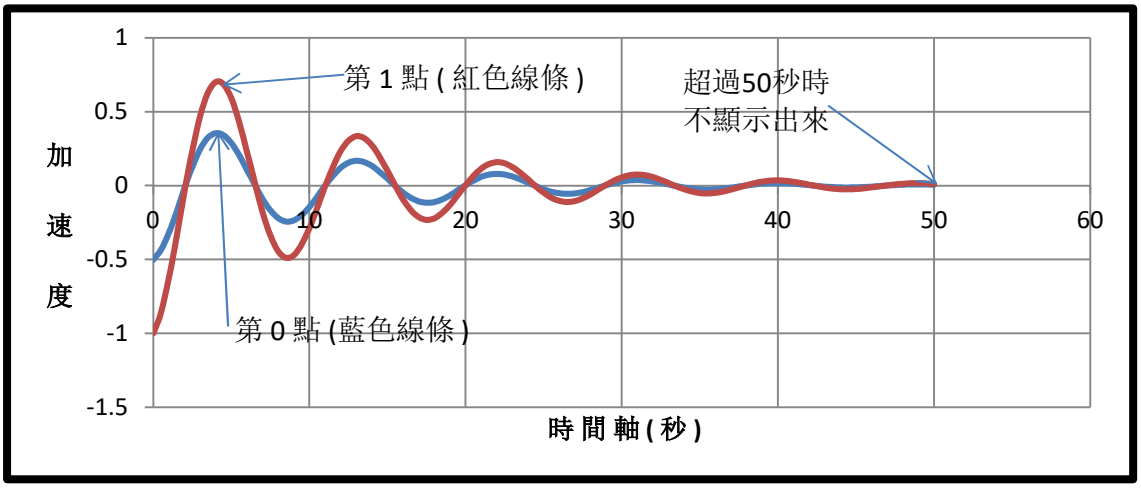
以下是將輸出結果，畫成圖表。



位 移 - 時 間 關 係 圖

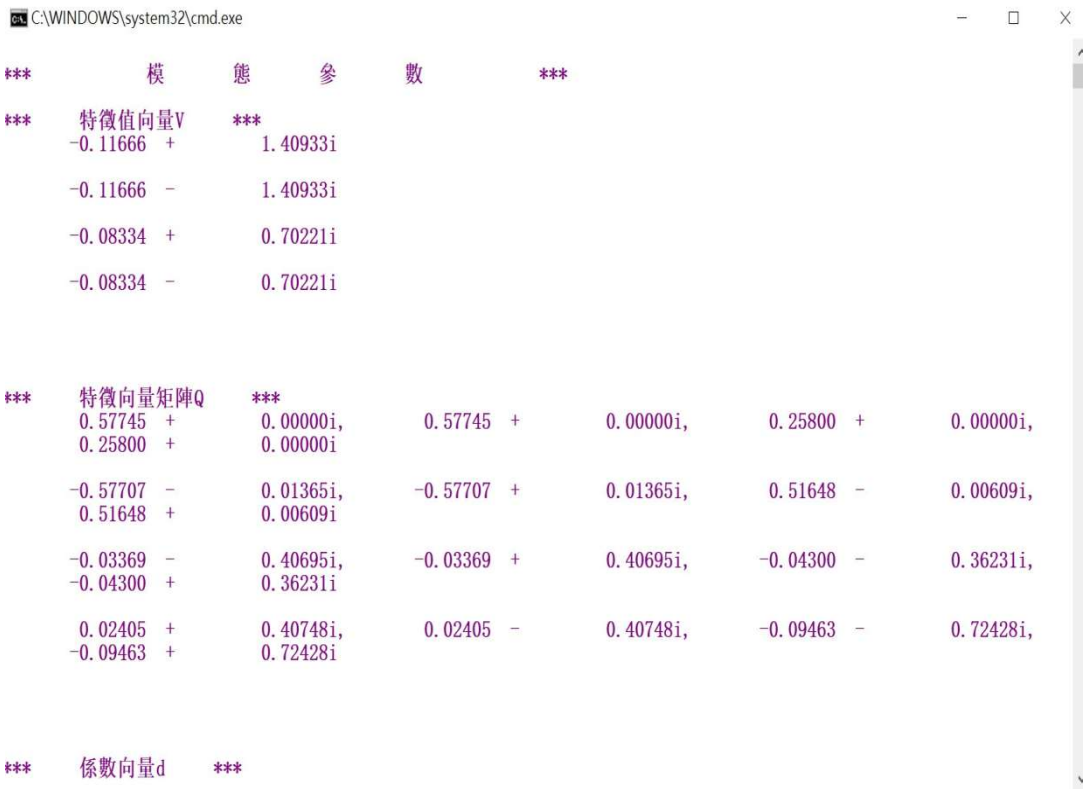


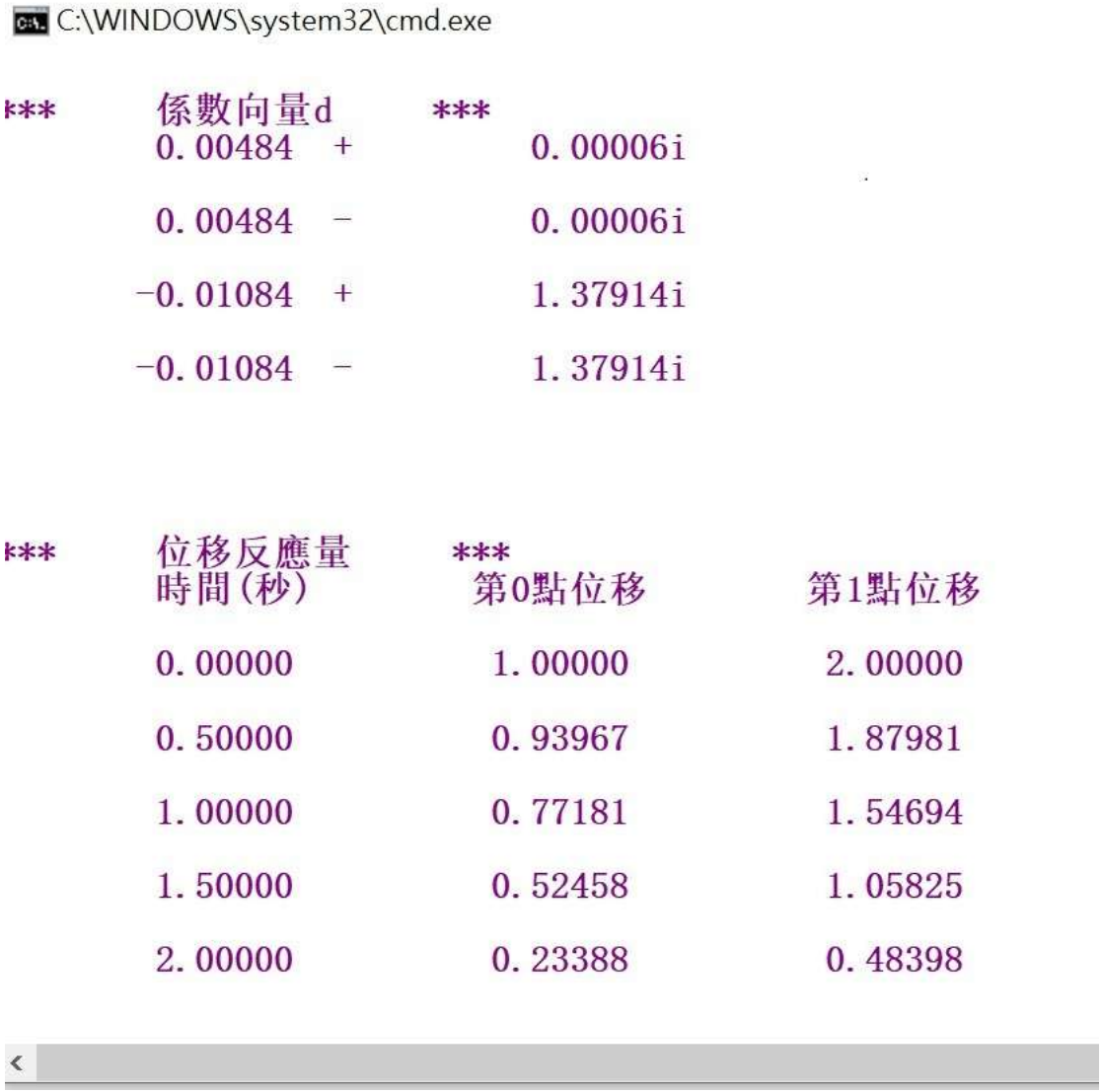
速 度 - 時 間 關 係 圖



加 速 度 - 時 間 關 係 圖

以下是影像 ：





實例三 ：

已知

A = { {3, 6}, {7, 4}, {2, 6} }　B = { {3, -4}, {2, 6} }　C = { {3, 5}, {2, 5}, {3, -1}}

求　A/ B + C * 3.5

程式碼如下 ：

```
using System;
using ClassLibrary1;
using ClassLibrary1A;

namespace ConsoleApp11
{
```

```
    class Program
    {
        static void Main(string[] args)
        {
            double[,] A = { { 3, 6 }, { 7, 4 }, { 2, 6 } };
            double[,] B = { { 3, -4 }, { 2, 6 } };
            double[,] C = { { 3, 5 }, { 2, 5 }, { 3, -1 } };
            // 求 D = A / B + C * 3.5
            ReMatrix D = (ReMatrix)A / B + (ReMatrix)C * 3.5;
            Console.Write("\n矩陣 D : \n{0}\n", new PR(D));
        }
    }
}
/* 輸出結果 :
矩陣 D :
       10.73077          18.65385
        8.30769          19.03846
       10.50000          -2.50000
*/
```

*****　以上輸出結果與 Matlab 執行的結果完全相同　*****

以下是輸出的影像 :