Maxima 在線性代數上之應用

正準形式

國立屏東教育大學 應用數學系 研究助理 徐偉玲

weilinghsu@mail.npue.edu.tw

日期：2009/9/8

第七章 正準形式

7.1 Jordan 正準形式 I

2.對每一個矩陣 A，對 L_A 的每一個廣義特徵空間找一組由不相交的廣義特徵向量循環集之聯集所組成之基底，再求 A 的一 Jordan 正準形式 J

(a)$\begin{pmatrix} 1 & 1 \\ -1 & 3 \end{pmatrix}$

```
(%i1) load("diag")$
```
Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

```
(%i2) A:matrix([1,1],[-1,3]);
```
//定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 1，第二列元素有-1, 3

```
(%o2)  [ 1   1 ]
       [      ]
       [-1   3 ]
```

```
(%i3) jordan(A);
```
Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[2, 2]]
```

```
(%i4) dispJordan(%);
```
disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

```
(%o4)  [ 2   1 ]
       [      ]
       [ 0   2 ]
```

(b)A=$\begin{pmatrix} 1 & 2 \\ 3 & 2 \end{pmatrix}$

```
(%i1) load("diag")$
```
Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲

什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([1,2],[3,2])$ //定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 2，第二列元素有 3, 2

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [ [ 4 , 1 ] , [ - 1 , 1 ] ]
```

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4) $\begin{bmatrix} 4 & 0 \\ 0 & -1 \end{bmatrix}$

(a) $A=\begin{pmatrix} 11 & -4 & -5 \\ 21 & -8 & -11 \\ 3 & -1 & 0 \end{pmatrix}$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([11,-4,-5],[21,-8,-11],[3,-1,0])$ //定義一矩陣，矩陣名稱叫作 A，第一列元素有 11, -4, -5，第二列元素有 21, -8, -11，第三列元素有 3, -1, 0

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [ [ - 1 , 1 ] , [ 2 , 2 ] ]
```

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}$$

(b) A=
$$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 1 & -1 & 3 \end{pmatrix}$$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([2,1,0,0],[0,2,1,0],[0,0,3,0],[0,1,-1,3])$ //定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 1, 0, 0，第二列元素有 0, 2, 1, 0，第三列元素有 0, 0, 3, 0，第四列元素有 0, 1, -1, 3

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[2,2],[3,1,1]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

3.對每一個線性算子 T，對 T 的每一個廣義特徵空間找一組由不相交之廣義特徵向量循環集之聯集所組成的基底，再求 T 的一 Jordan 正準形式

(a) T 爲 $P_2(R)$上的線性算子定義爲$T(f(x)) = 2f(x) - f'(x)$

```
(%i1) load("diag")$
```
Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

```
(%i2) A:matrix([2,-1,0],[0,2,-2],[0,0,2])$
```
//定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, -1, 0，第二列元素有 0, 2, -2，第三列元素有 0, 0, 2

```
(%i3) jordan(A);
```
Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [ [ 2 , 3 ] ]
```

```
(%i4) dispJordan(%);
```
disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

$$(\%o4)\quad \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}$$

(b) V 爲由實值函數$\{1, t, t^2, e^t, te^t\}$所生成的函數實向量空間，且 T 爲 V 上的線性算子定義爲$T(f) = f'$

```
(%i1) load("diag")$
```
Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([0,0,0,0,0],[1,0,0,0,0],[0,2,0,0,0],[0,0,0,1,0],[0,1,0,0,1])$ //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, 0, 0, 0, 0，第二列元素有 1,,0,,0,,0,,0，第三列元素有 0, 2, 0, 0, 0，第四列元素有 0, 0, 0, 1, 0，第五列元素有 0, 1, 0, 0, 1

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[0,3],[1,1,1]]
```

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

$$(\%o4)\quad \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(c) T 爲 $M_{2\times 2}(R)$上的線性算子定義爲 $T(A)=\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}A$， $\forall A \in M_{2\times 2}(R)$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) B:matrix([1,1],[0,1]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 1，第二列元素有 0, 1

$$(\%o2)\quad \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

(%i3) B.matrix([1,0],[0,0]);　　//計算矩陣 B 乘上矩陣$\begin{pmatrix}1 & 0\\0 & 0\end{pmatrix}$

(%o3) $\begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}$

(%i4) B.matrix([0,0],[1,0]);　　//計算矩陣 B 乘上矩陣$\begin{pmatrix}0 & 0\\1 & 0\end{pmatrix}$

(%o4) $\begin{bmatrix}1 & 0\\1 & 0\end{bmatrix}$

(%i5) B.matrix([0,1],[0,0]);　//計算矩陣 B 乘上矩陣$\begin{pmatrix}0 & 1\\0 & 0\end{pmatrix}$

(%o5) $\begin{bmatrix}0 & 1\\0 & 0\end{bmatrix}$

(%i6) B.matrix([0,0],[0,1]);　//計算矩陣 B 乘上矩陣$\begin{pmatrix}0 & 0\\0 & 1\end{pmatrix}$=

(%o6) $\begin{bmatrix}0 & 1\\0 & 1\end{bmatrix}$

(%i7) A:matrix([1,0,0,0],[1,1,0,0],[0,0,1,0],[0,0,1,1])$　//定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 0, 0, 0，第二列元素有 1, 1, 0, 0，第三列元素有 0, 0, 1, 0，第四列元素有 0, 0, 1, 1 Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來　//求矩陣 A 的正準形式

(%i8) jordan(A);　Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來　//求矩陣 A 的正準形式

(%o8)　[[1,2,2]]

(%i9) dispJordan(%);　disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果　//A 的正準形式矩陣的輸出

(%o9)
$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(c) T(A)= $2A + A^t$ ， $\forall A \in M_{2\times 2}(R)$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([3,0,0,0],[0,2,1,0],[0,1,2,0],[0,0,0,3])$ //定義一矩陣，矩陣名稱叫作 A，第一列元素有 3, 0, 0, 0，第二列元素有 0, 2, 1, 0，第三列元素有 0, 1, 2, 0，第四列元素有 0, 0, 0, 3

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[1,1],[3,1,1,1]]
```

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

7.2 Jordan 正準形式 II

3.設 T 爲有限維度向量空間 V 上的一線性算子，使 T 的 Jordan 正準形式爲

$$\begin{pmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

(a)求 T 的特徵多項式

```
(%i1) load("diag")$
```

Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

```
(%i2)
T:matrix([2,1,0,0,0,0,0],[0,2,1,0,0,0,0],[0,0,2,0,0,0,0],[0,0,0,2,1,0,0],[0,0,0,0,2,0,0],[0,0,0,0,0,3,0],[0,0,0,0,0,0,3]);
```

//定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 1, 0, 0, 0, 0, 0，第二列元素有 0, 2, 1, 0, 0, 0, 0，第三列元素有 0, 0, 2, 0, 0, 0, 0，第四列元素有 0, 0, 0, 2, 1, 0, 0，第五列元素有 0, 0, 0, 0, 2, 0, 0，第六列元素有 0, 0, 0, 0, 0, 3, 0，第七列元素有 0, 0, 0, 0, 0, 0, 3

```
(%o2)
```

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

(%i3) f:charpoly(T,t); 特徵多項式指令：charpoly(矩陣，變數) //求矩陣 T 的特徵多項式，名稱訂爲 f

(%o3) $(2-t)^5(3-t)^2$

(b)求對應 T 每一特徵值的點圖

(c)使 $E_{\lambda_i}=K_{\lambda_i}$ 的特徵值 λ_i 爲何？

(d)對每一特徵值 λ_i ，求最小正整數 p_i 使 $K_{\lambda_i}=N((T-\lambda_i I))^{p_i}$

(e)令 U_i 表示 $T-\lambda_i I$ 在 K_{λ_i} 上的限制變換， $\forall i$ ，試求下列：

(Ⅰ) rank(U_i)

(Ⅱ) rank(U_i^2)

(Ⅲ) nullity(U_i)

(Ⅳ) nullity(U_i^2)

4.對下列各矩陣 A，求 Jordan 正準形式 J 及可逆矩陣 Q，使 $J=Q^{-1}AQ$，注意(a)，(b)及(c)的矩陣即例 5 中使用的矩陣

(a) A=$\begin{pmatrix}-3 & 3 & -2\\ -7 & 6 & -3\\ 1 & -1 & 2\end{pmatrix}$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([-3,3,-2],[-7,6,-3],[1,-1,2]);　//定義一矩陣，矩陣名稱叫作 A，第一列元素有-3, 3, -2，第二列元素有-7, 6, -3，第三列元素有 1, -1, 2

(%o2)
$$\begin{bmatrix} -3 & 3 & -2 \\ -7 & 6 & -3 \\ 1 & -1 & 2 \end{bmatrix}$$

(%i3) jordan(A);　Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來　//求矩陣 A 的正準形式

(%o3)　[[1,1],[2,2]]

(%i4) dispJordan(%);　disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果　//A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}$$

(b) A=$\begin{pmatrix} 0 & 1 & -1 \\ -4 & 4 & -2 \\ -2 & 1 & 1 \end{pmatrix}$

(%i1) load("diag")$　Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。　//我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。　//讀取 diag 模組

(%i2) A:matrix([0,1,-1],[-4,4,-2],[-2,1,1]);　//定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, 1, -1，第二列元素有-4, 4, -2，第三列元素有-2, 1, 1

(%o2)
$$\begin{bmatrix} 0 & 1 & -1 \\ -4 & 4 & -2 \\ -2 & 1 & 1 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[1,1],[2,1,1]]
```

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

(c) $A=\begin{pmatrix} 0 & -1 & -1 \\ -3 & -1 & -2 \\ 7 & 5 & 6 \end{pmatrix}$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([0,-1,-1],[-3,-1,-2],[7,5,6]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, -1, -1，第二列元素有-3, -1, -2，第三列元素有 7, 5, 6

(%o2)
$$\begin{bmatrix} 0 & -1 & -1 \\ -3 & -1 & -2 \\ 7 & 5 & 6 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[1,1],[2,2]]
```

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}$$

(d) A=
$$\begin{pmatrix} 0 & -3 & 1 & 2 \\ -2 & 1 & -1 & 2 \\ -2 & 1 & -1 & 2 \\ -2 & -3 & 1 & 4 \end{pmatrix}$$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([0,-3,1,2],[-2,1,-1,2],[-2,1,-1,2],[-2,-3,1,4]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, -3, 1, 2，第二列元素有-2, 1, -1, 2，第三列元素有-2, 1, -1, 2，第四列元素有-2, -3, 1, 4

(%o2)
$$\begin{bmatrix} 0 & -3 & 1 & 2 \\ -2 & 1 & -1 & 2 \\ -2 & 1 & -1 & 2 \\ -2 & -3 & 1 & 4 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[2,1,1],[0,2]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

5.對下列各線性算子 T，求 T 的 Jordan 的正準形式及 T 的一組 Jordan 正準基底 β

(a) V 爲由實值函數集$\{e^t, te^t, t^2e^t, e^{2t}\}$所生成的函數之實向量空間，且 T 爲 V 上的線性算子定義爲$T(f) = f'$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([1,1,0,0],[0,1,2,0],[0,0,1,0],[0,0,0,2]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 1, 0, 0，第二列元素有 0, 1, 2, 0，第三列元素有 0, 0, 1, 0，第四列元素有 0, 0, 0, 2

(%o2)
$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[1,3],[2,1]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

(b) T 爲 $P_3(R)$上的線性算子定義爲$T(f(x)) = xf''(x)$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([0,0,0,0],[0,0,0,0],[0,2,0,0],[0,0,6,0]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, 0, 0, 0，第二列元素有 0, 0, 0, 0，第三列元素有 0, 2, 0, 0，第四列元素有 0, 0, 6, 0

(%o2)
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 6 & 0 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[0,3,1]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(c) T 爲 $P_3(R)$上的線性算子定義爲$T(f(x)) = f''(x) + 2f(x)$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([2,0,0,0],[0,2,0,0],[2,0,2,0],[0,6,0,2]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 0, 0, 0，第二列元素有 0, 2, 0, 0，第三列元素有 2, 0, 2, 0，第四列元素有 0, 6, 0, 2

(%o2)
$$\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 2 & 0 & 2 & 0 \\ 0 & 6 & 0 & 2 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[2,2,2]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

(d) T 爲 $M_{2\times 2}(R)$上的線性算子定義爲 $T(A)=\begin{pmatrix} 3 & 1 \\ 0 & 3 \end{pmatrix} \bullet A - A^t$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([2,0,0,0],[0,3,-1,0],[1,-1,3,0],[0,1,0,2]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 0, 0, 0，第二列元素有 0, 3, -1, 0，第三列元素有 1, -1, 3, 0，第四列元素有 0, 1, 0, 2

(%o2)
$$\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & -1 & 0 \\ 1 & -1 & 3 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[4,1],[2,3]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

(d) T 爲 $M_{2\times 2}(R)$上的線性算子定義爲 $T(A)=\begin{pmatrix} 3 & 1 \\ 0 & 3 \end{pmatrix} \bullet (A - A^t)$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 disjordan 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算正準形式。 //我們要算一矩陣的 jordan 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form。 //讀取 diag 模組

(%i2) A:matrix([0,0,0,0],[-1,3,-3,0],[1,-3,3,0],[0,0,0,0]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, 0, 0, 0，第二列元素有-1, 3, -3, 0，第三列元素有 1, -3, 3, 0，第四列元素有 0, 0, 0, 0

(%o2)
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 3 & -3 & 0 \\ 1 & -3 & 3 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) [[0,1,1,1],[6,1]]

(%i4) dispJordan(%); disjordan 指令是將上列所算出 jodan form 以矩陣形式表現出來，%是引用前面計算出來的結果 //A 的正準形式矩陣的輸出

(%o4)
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 \end{bmatrix}$$

(e) V 爲項數至多爲 2 的兩變數 x 及 y 之多項式函數所成的向量空間，如例 4 所定義，且 T 爲 V 上的線性算子定義爲 $T(f(x,y)) = \frac{\delta}{\delta x} f(x,y) + \frac{\delta}{\delta y} f(x,y)$

24.利用 23 求下列各微分方程組的通解，其中 x，y，及 z 是實變數 t 的實值可微分函數

(a)
$$\begin{aligned} x' &= 2x + y \\ y' &= 2y - z \\ z' &= 3z \end{aligned}$$

(b)
$$\begin{aligned} x' &= 2x + y \\ y' &= 2y + z \\ z' &= 2z \end{aligned}$$

7.3 最小多項式

2.求下列各矩陣的最小多項式

(a) $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的

jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

```
(%i2) A:matrix([2,1],[1,2]);
```

//定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 1，第二列元素有 1, 2

$$(\%o2)\quad \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

```
(%i3) jordan(A);
```

Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[3,1],[1,1]]
```

```
(%i4) minimalPoly(%);
```

最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

```
(%o4)  (x-3)(x-1)
```

(b) $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$

```
(%i1) load("diag")$
```

Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

```
(%i2) A:matrix([1,1],[0,1]);
```

//定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 1，第二列元素有 0, 1

$$(\%o2)\quad \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

```
(%i3) jordan(A);
```
Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[1,2]]
```

```
(%i4) minimalPoly(%);
```
最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

```
(%o4)  (x-1)^2
```

(c) $\begin{pmatrix} 4 & -14 & 5 \\ 1 & -4 & 2 \\ 1 & -6 & 4 \end{pmatrix}$

```
(%i1) load("diag")$
```
Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

```
(%i2) A:matrix([4,-14,5],[1,-4,2],[1,-6,4]);
```
//定義一矩陣，矩陣名稱叫作 A，第一列元素有 4, -14, 5，第二列元素有 1, -4, 2，第三列元素有 1, -6, 4

```
(%o2)  [ 4  -14  5 ]
       [ 1   -4  2 ]
       [ 1   -6  4 ]
```

```
(%i3) jordan(A);
```
Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[2,1],[1,2]]
```

(%i4) minimalPoly(%); 最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

```
(%o4)  (x-2)(x-1)^2
```

(d) $\begin{pmatrix} 3 & 0 & 1 \\ 2 & 2 & 2 \\ -1 & 0 & 1 \end{pmatrix}$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([3,0,1],[2,2,2],[-1,0,1]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 3, 0, 1，第二列元素有 2, 2, 2，第三列元素有-1, 0, 1

```
(%o2)  [ 3   0   1 ]
       [           ]
       [ 2   2   2 ]
       [           ]
       [-1   0   1 ]
```

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [[2,2,1]]
```

(%i4) minimalPoly(%); 最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

```
(%o4)  (x-2)^2
```

3.對下列 V 上的各線性算子 T，求 T 的最小多項式

(a) $V=R^3$，且 T(a, b)=(a+b, a-b)

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([1,1],[1,-1]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 1, 1，第二列元素有 1, -1

(%o2) $\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) $[[-\sqrt{2},1],[\sqrt{2},1]]$

(%i4) minimalPoly(%); 最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

(%o4) $(x-\sqrt{2})(x+\sqrt{2})$

(b) V=P_2(R)，且$T(g(x))=g'(x)+2g(x)$

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([2,1,0],[0,2,2],[0,0,2]);　//定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 1, 0，第二列元素有 0, 2, 2，第三列元素有 0, 0, 2

(%o2) $\begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 2 \\ 0 & 0 & 2 \end{bmatrix}$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來　//求矩陣 A 的正準形式

(%o3) [[2,3]]

(%i4) minimalPoly(%);　最小多項式指令：minimalpoly(矩陣)　//求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

(%o4) $(x-2)^3$

(c) V=P_2(R)，且 $T(f(x)) = -xf''(x) + f'(x) + 2f(x)$

(%i1) load("diag")$　Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。　//我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([2,1,0],[0,2,0],[0,0,2]);　//定義一矩陣，矩陣名稱叫作 A，第一列元素有 2, 1, 0，第二列元素有 0, 2, 0，第三列元素有 0, 0, 2

(%o2) $\begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$

(%i3) jordan(A);　Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來　//求矩陣 A 的正準形式

```
(%o3)  [[2,2,1]]
```

(%i4) minimalPoly(%); 最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

```
(%o4)  (x-2)^2
```

(d) V=$M_{n\times n}(R)$，且 T(A)=A^t，提示：注意 T^2=I

7.4 有理正準形式

2.對下面的每一個矩陣 A∈ $M_{n\times n}(F)$，求矩陣 A 的有理正準形式 C 及矩陣，

Q∈ $M_{n\times n}(F)$，使 $Q^{-1}AQ=C$

(a) A=$\begin{pmatrix}3&1&0\\0&3&1\\0&0&3\end{pmatrix}$，F=R

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([3,1,0],[0,3,1],[0,0,3]);　//定義一矩陣，矩陣名稱叫作 A，第一列元素有 3, 1, 0，第二列元素有 0, 3, 1，第三列元素有 0, 0, 3

$$(\%o2)\quad \begin{bmatrix} 3 & 1 & 0 \\ 0 & 3 & 1 \\ 0 & 0 & 3 \end{bmatrix}$$

(%i3) jordan(A);　Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來　//求矩陣 A 的正準形式

(%o3)　[[3,3]]

(%i4) f:minimalPoly(%);　最小多項式指令：minimalpoly(矩陣)　//求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

$$(\%o4)\quad (x-3)^3$$

(%i5) expand(f);　多項式展開指令：expand(多項式名稱)　//將最小多項式 f 展開

$$(\%o5)\quad x^3-9x^2+27x-27$$

(%i6) C:matrix([0,0,27],[1,0,-27],[0,1,9]);　////定義一矩陣，矩陣名稱叫作 C，第一列元素有 0, 0, 27，第二列元素有 1, 0, -27，第三列元素有 0, 1, 9

$$(\%o6)\quad \begin{bmatrix} 0 & 0 & 27 \\ 1 & 0 & -27 \\ 0 & 1 & 9 \end{bmatrix}$$

(b) $A=\begin{pmatrix} 0 & -1 \\ 1 & -1 \end{pmatrix}$，F=R

(c) $A=\begin{pmatrix} 0 & -1 \\ 1 & -1 \end{pmatrix}$，F=C

(%i1) load("diag")$　Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些

指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([0,-1],[1,-1]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, -1，第二列元素有 1, -1

(%o2) $\begin{bmatrix} 0 & -1 \\ 1 & -1 \end{bmatrix}$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) $[[-\frac{\sqrt{3}\,\%i+1}{2},1],[\frac{\sqrt{3}\,\%i-1}{2},1]]$

(%i4) f:minimalPoly(%); 最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

(%o4) $\left(x-\frac{\sqrt{3}\,\%i-1}{2}\right)\left(x+\frac{\sqrt{3}\,\%i+1}{2}\right)$

(%i5) expand(f); 多項式展開指令：expand(多項式名稱) //將最小多項式 f 展開

(%o5) x^2+x+1

(%i6) C:matrix([-(sqrt(3)*%i-1)/2,0],[0,(sqrt(3)*%i+1)/2]); //定義一矩陣，矩陣名稱叫作 C，第一列元素有$\frac{1-\sqrt{3}i}{2}$, 0，第二列元素有 0, $\frac{\sqrt{3}i+1}{2}$

(%o6) $\begin{bmatrix} \frac{1-\sqrt{3}\,\%i}{2} & 0 \\ 0 & \frac{\sqrt{3}\,\%i+1}{2} \end{bmatrix}$

(d) $A=\begin{pmatrix} 0 & -7 & 14 & -6 \\ 1 & -4 & 6 & -3 \\ 0 & -4 & 9 & -4 \\ 0 & -4 & 11 & -5 \end{pmatrix}$，F=R

```
(%i1) load("diag")$
```
Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

```
(%i2) A:matrix([0,-7,14,-6],[1,-4,6,-3],[0,-4,9,-4],[0,-4,11,-5]);
```
//定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, -7, 14, -6，第二列元素有 1, -4, 6, -3，第三列元素有 0, -4, 9, -4，第四列元素有 0, -4, 11, -5

```
(%o2)  [  0  -7  14  -6 ]
       [                ]
       [  1  -4   6  -3 ]
       [                ]
       [  0  -4   9  -4 ]
       [                ]
       [  0  -4  11  -5 ]
```

```
(%i3) jordan(A);
```
Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

```
(%o3)  [ [ - %i , 2 ] , [ %i , 2 ] ]
```

```
(%i4) f:minimalPoly(%);
```
最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

```
(%o4)  (x - %i)^2 (x + %i)^2
```

```
(%i5) expand(f);
```
多項式展開指令：expand(多項式名稱) //將最小多項式 f 展開

```
(%o5)  x^4 + 2 x^2 + 1
```

```
(%i6) C:matrix([0,-1,0,0],[1,0,0,0],[0,0,0,-1],[0,0,1,0]);
```
//定義一矩陣，矩陣名稱叫作 C，第一列元素有 0, -1, 0, 0，第二列元素有 1, 0, 0, 0，第三列元素有 0, 0, 0, -1，

第四列元素有 0, 0, 1, 0

(%o6)
$$\begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(e)A=$\begin{pmatrix} 0 & -4 & 12 & -7 \\ 1 & -1 & 3 & -3 \\ 0 & -1 & 6 & -4 \\ 0 & -1 & 8 & -5 \end{pmatrix}$，F=R

(%i1) load("diag")$ Maxima 本身是「不會」算 jordan 跟 minimal polynomial 的，我們可以用適當的模組來做這件事，所謂模組就是一段小程式，通常是增加一些指令，爲什麼 Maxima 一開始就把這些模組都加進來？那是因爲一來太佔記憶體，一來是很多對某些人很重要的指令也許你永遠也沒用到。我們要算一矩陣的 jordanc、minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令和 minimalpoly 指令去計算正準形式跟最小多項式。 //我們要算一矩陣的 jordan 和 minimalpoly 要使用 diag 這個模組，此模組提供了 jordan 指令去計算 jordan canonical form 和 minimalpoly 指令去算最小多項式。

(%i2) A:matrix([0,-4,12,-7],[1,-1,3,-3],[0,-1,6,-4],[0,-1,8,-5]); //定義一矩陣，矩陣名稱叫作 A，第一列元素有 0, -4, 12, -7，第二列元素有 1, -1, 3, -3，第三列元素有 0, -1, 6, -4，第四列元素有 0, -1, 8, -5

(%o2)
$$\begin{bmatrix} 0 & -4 & 12 & -7 \\ 1 & -1 & 3 & -3 \\ 0 & -1 & 6 & -4 \\ 0 & -1 & 8 & -5 \end{bmatrix}$$

(%i3) jordan(A); Jordan form 指令：jordan(矩陣)，但 jordan 指令只會把算出的數字在 Maxima 上分類列舉出來 //求矩陣 A 的正準形式

(%o3) $[[-\sqrt{2}\,i,1],[\sqrt{2}\,i,1],[-\sqrt{3}\,i,1],[\sqrt{3}\,i,1]]$

(%i4) f:minimalPoly(%); 最小多項式指令：minimalpoly(矩陣) //求矩陣 A 的最小多項式，%是表示引用前面所計算出來的結果

(%o4) $(x-\sqrt{2}\,\%i)(x+\sqrt{2}\,\%i)(x-\sqrt{3}\,\%i)(x+\sqrt{3}\,\%i)$

(%i5) expand(f); 多項式展開指令：expand(多項式名稱) //將最小多項式 f 展開

(%o5) x^4+5x^2+6

(%i6) C:matrix([0,-2,0,0],[1,0,0,0],[0,0,0,-3],[0,0,1,0]); //定義一矩陣，矩陣名稱叫作 C，第一列元素有 0, -2, 0, 0，第二列元素有 1, 0, 0, 0，第三列元素有 0, 0, 0,-3，第四列元素有 0, 0, 1, 0

(%o6) $\begin{bmatrix} 0 & -2 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -3 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

3.對下面各線性算子 T，求其基本約式，有理正準形式 C，及一組有理正準基底 β

(a) T 爲 $P_3(R)$上的線性算子定義爲 $T(f(x))=f(0)x-f'(1)$

(b)令 S={sinx，cosx，xsinx，xcosx}爲 F(R，R)的一子集合，且令 V=span(S)，定義 T 爲 V 上的線性算子滿足 $T(f)=f'$

(c) T 爲 $M_{2\times 2}(R)$上的線性算子定義爲 $T(A)=\begin{pmatrix} 0 & 1 \\ -1 & 1 \end{pmatrix}\bullet A$

(d)令 S={sinxsiny，sinxcosy，cosxsiny，cosxcosy}爲 F(R×R，R)的一子集合，且令 V=span(S)，定義 T 爲 V 上的線性算子滿足 $T(f)(x,y)=\frac{\partial f(x,y)}{\partial x}+\frac{\partial f(x,y)}{\partial y}$