Autovalori e autovettori

Note

```
filename: autovalori.mws
 version: 2.0
  tested: Maple V Release 6.02 on Windows 2000
    date: 15 marzo 2003
  author: Claudio Marsan
          Liceo Cantonale di Mendrisio
          Via Agostino Maspoli
          CH-6850 Mendrisio
          e-mail: claudio.marsan@liceomendrisio.ch
```

```
> restart:
> with(linalg):
Warning, the protected names norm and trace have been redefined and unprotected
```

La matrice identità si indica con &*(). Introduciamo un simbolo più facile da ricordare:

```
> alias(Id=&*());
```

$$Id$$

Sia A una matrice quadrata di ordine n e **v** un vettore con n componenti. Prendiamo, nel seguito, n=2. La matrice A è la matrice di un'applicazione lineare f da R^2 verso R^2. Calcolare il prodotto A**v** equivale così a calcolare f(**v**). Talvolta ci si chiede se esistono punti fissi per la funzione f (ossia vettori **v** per i quali vale f(**v**)=**v**) o anche se esistono vettori **v** paralleli alla loro immagine sotto f, ossia se esistono vettori **v** con f(**v**)=λ**v** per un certo valore di λ non nullo. Il valore λ è detto autovalore (o valore proprio o valore caratteristico) di f (e di A) e **v** è il relativo autovettore (o vettore proprio o vettore caratteristico).
Per determinare gli autovalori di A basta risolvere l'equazione A**v**=λ**v**, che è equivalente a A**v** - λ**v** = 0, ossia (A - λId)**v** = 0.
Affinché ci siano delle soluzioni non nulle deve valere l'equazione caratteristica di A: det(A - λId) = 0. det(A - λId) è detto polinomio caratteristico di A.

Consideriamo la seguente matrice A e determiniamone gli autovalori.

```
> A := matrix(2,2,[[8,-5],[-1,4]]);
```

$$A := \begin{bmatrix} 8 & -5 \\ -1 & 4 \end{bmatrix}$$

```
> car_eq := det(A - lambda*Id) = 0;
```

$$car\_eq := 27 - 12\,\lambda + \lambda^2 = 0$$

```
> lambda := solve(car_eq, lambda);
```

$$\lambda := 9, 3$$

```
> lambda[1];
```

$$9$$

```
> lambda[2];
```

$$3$$

Per determinare gli autovettori basta sostituire gli autovalori in $A\mathbf{v}=\lambda\mathbf{v}$.

Autovettore relativo a λ_1

```
> v := vector([x,y]);
```

$$v := [x, y]$$

```
> A &* v - lambda[1] * v = [0, 0];
```

$$(A \&* v) - 9\, v = [0, 0]$$

```
> evalm(%);
```

$$[-x - 5\, y, -x - 5\, y] = [0, 0]$$

```
> solve({lhs(%)[1]=0, lhs(%)[2]=0},{x,y});
```

$$\{y = y, x = -5\, y\}$$

```
> nullspace(A  - lambda[1] * Id);
```

$$\{[-5, 1]\}$$

Autovettore relativo a λ_2

```
> v := vector([x,y]);
```

$$v := [x, y]$$

```
> A &* v - lambda[2] * v = [0, 0];
```

$$(A \&* v) - 3\, v = [0, 0]$$

```
> evalm(%);
```

$$[5\, x - 5\, y, -x + y] = [0, 0]$$

```
> solve({lhs(%)[1]=0, lhs(%)[2]=0},{x,y});
```

$$\{y = y, x = y\}$$

```
> nullspace(A  - lambda[2] * Id);
```

$$\{[1, 1]\}$$

Naturalmente con Maple si possono automatizzare molte operazioni:

```
> restart:
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> alias(Id=&*());
```

$$Id$$

```
> A := matrix(2,2,[[8,-5],[-1,4]]);
```

$$A := \begin{bmatrix} 8 & -5 \\ -1 & 4 \end{bmatrix}$$

Il comando **charmat**(A, λ) calcola la matrice caratteristica di A nell'incognita λ:

```
> charmat(A,lambda);
```

$$\begin{bmatrix} \lambda - 8 & 5 \\ 1 & \lambda - 4 \end{bmatrix}$$

Il comando **charpoly**(A, λ) calcola il polinomio caratteristico di A nell'incognita λ:

```
> charpoly(A,lambda);
```

$$\lambda^2 - 12\,\lambda + 27$$

Il comando **eigenvals**(A) calcola gli autovalori di A:

```
> eigenvals(A);
```

$$9,\ 3$$

Il comando **eigenvects**(A) calcola gli autovalori di A, la loro molteplicità e i relativi autovettori:

```
> eigenvects(A);
```

$$[3,\ 1,\ \{[1,\ 1]\}],\ [9,\ 1,\ \{[-5,\ 1]\}]$$

(la riga sopra va così interpretata: 3 è un autovalore di A con molteplicità 1 e relativo autovettore [1,1]; 9 è un autovalore di A con molteplicità 1 e relativo autovettore [-5, 1])

Applicazione ai sistemi di equazioni differenziali lineari del primo ordine

```
> restart:
> with(linalg):
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> alias(Id=&*());
```

$$Id$$

```
> ode1 := diff(y1(t), t) = 2*y2(t);
```

$$ode1 := \frac{\partial}{\partial t}\,\mathrm{y1}(t) = 2\,\mathrm{y2}(t)$$

```
> ode2 :=  diff(y2(t), t) = -y1(t) - 3*y2(t);
```

$$ode2 := \frac{\partial}{\partial t}\,\mathrm{y2}(t) = -\mathrm{y1}(t) - 3\,\mathrm{y2}(t)$$

```
> initconds := y1(0) = 1, y2(0) = -4;
```

$$initconds := \mathrm{y1}(0) = 1,\ \mathrm{y2}(0) = -4$$

La matrice del sistema è:

```
> A := matrix(2,2,[[0,2],[-1,-3]]);
```

$$A := \begin{bmatrix} 0 & 2 \\ -1 & -3 \end{bmatrix}$$

Gli autovalori e ralativi autovettori:

```
> v := eigenvects(A);
```

$$v := [-1,\ 1,\ \{[-2,\ 1]\}],\ [-2,\ 1,\ \{[1,\ -1]\}]$$

```
> lambda[1] := v[1][1]; lambda[2] := v[2][1];
```

$$\lambda_1 := -1$$

$$\lambda_2 := -2$$

```
> v1 := (v[1][3][1]); v2 := v[2][3][1];
```

$$v1 := [-2,\ 1]$$

$$v2 := [1,\ -1]$$

```
> w1 := C1 * exp(lambda[1]*t) * v1;
```

$$w1 := C1\,\mathbf{e}^{(-t)}\,v1$$

```
> w2 := C2 * exp(lambda[2]*t) * v2;
```

$$w2 := C2\, \mathbf{e}^{(-2\, t)}\, v2$$

```
> Y := w1 + w2;
```

$$Y := C1\, \mathbf{e}^{(-t)}\, v1 + C2\, \mathbf{e}^{(-2\, t)}\, v2$$

```
> Y := evalm(%);
```

$$Y := [-2\, C1\, \mathbf{e}^{(-t)} + C2\, \mathbf{e}^{(-2\, t)},\, C1\, \mathbf{e}^{(-t)} - C2\, \mathbf{e}^{(-2\, t)}]$$

Condizioni iniziali: y1(0) = 1, y(0) = -4:

```
> sol := solve({eval(subs(t=0,Y[1]))=1,
  eval(subs(t=0,Y[2]))=-4},{C1,C2});
```

$$sol := \{C1 = 3,\, C2 = 7\}$$

```
> assign(sol);
```

La soluzione:

```
> 'y1(t)' = Y[1]; 'y2(t)' = Y[2];
```

$$\mathrm{y1}(t) = -6\, \mathbf{e}^{(-t)} + 7\, \mathbf{e}^{(-2\, t)}$$

$$\mathrm{y2}(t) = 3\, \mathbf{e}^{(-t)} - 7\, \mathbf{e}^{(-2\, t)}$$

Naturalmente si poteva procedere direttamente con il comando **dsolve**:

```
> dsolve({ode1, ode2, initconds}, {y1(t), y2(t)});
```

$$\{\mathrm{y1}(t) = -6\, \mathbf{e}^{(-t)} + 7\, \mathbf{e}^{(-2\, t)},\, \mathrm{y2}(t) = 3\, \mathbf{e}^{(-t)} - 7\, \mathbf{e}^{(-2\, t)}\}$$

Un altro esempio:

```
> restart:
> with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
> alias(Id=&*());
```

$$Id$$

```
> ode1 := diff(y1(t), t) = y2(t);
```

$$ode1 := \frac{\partial}{\partial t}\mathrm{y1}(t) = \mathrm{y2}(t)$$

```
> ode2 :=  diff(y2(t), t) = -y1(t);
```

$$ode2 := \frac{\partial}{\partial t}\mathrm{y2}(t) = -\mathrm{y1}(t)$$

```
> initconds := y1(0) = 1, y2(0) = 0;
```

$$initconds := \mathrm{y1}(0) = 1,\, \mathrm{y2}(0) = 0$$

La matrice del sistema è:

```
> A := matrix(2,2,[[0,1],[-1,0]]);
```

$$A := \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Gli autovalori e ralativi autovettori:

```
> v := eigenvects(A);
```

$$v := [I, 1, \{[1, I]\}], [-I, 1, \{[1, -I]\}]$$

```
> lambda[1] := v[1][1]; lambda[2] := v[2][1];
```

$$\lambda_1 := I$$

$$\lambda_2 := -I$$

```
> v1 := (v[1][3][1]); v2 := v[2][3][1];
```

$$v1 := [1, I]$$

$$v2 := [1, -I]$$

```
> w1 := C1 * exp(lambda[1]*t) * v1;
```

$$w1 := C1\, \mathbf{e}^{(I\,t)}\, v1$$

```
> w2 := C2 * exp(lambda[2]*t) * v2;
```

$$w2 := C2\, \mathbf{e}^{(-I\,t)}\, v2$$

```
> Y := w1 + w2;
```

$$Y := C1\, \mathbf{e}^{(I\,t)}\, v1 + C2\, \mathbf{e}^{(-I\,t)}\, v2$$

```
> Y := evalm(%);
```

$$Y := [C1\, \mathbf{e}^{(I\,t)} + C2\, \mathbf{e}^{(-I\,t)}, I\, C1\, \mathbf{e}^{(I\,t)} - I\, C2\, \mathbf{e}^{(-I\,t)}]$$

Condizioni iniziali: y1(0) = 1, y(0) = 0:

```
> sol := solve({eval(subs(t=0,Y[1]))=1,
  eval(subs(t=0,Y[2]))=0},{C1,C2});
```

$$sol := \{C1 = \frac{1}{2}, C2 = \frac{1}{2}\}$$

```
> assign(sol);
```

La soluzione:

```
> 'y1(t)' = Y[1]; 'y2(t)' = Y[2];
```

$$\mathrm{y1}(t) = \frac{1}{2}\mathbf{e}^{(I\,t)} + \frac{1}{2}\mathbf{e}^{(-I\,t)}$$

$$\mathrm{y2}(t) = \frac{1}{2} I\, \mathbf{e}^{(I\,t)} - \frac{1}{2} I\, \mathbf{e}^{(-I\,t)}$$

I numeri complessi non li vogliamo nella soluzione:

```
> 'y1(t)' = convert(Y[1],trig);
```

$$\mathrm{y1}(t) = \cos(t)$$

```
> 'y2(t)' = convert(Y[2],trig);
```

$$\mathrm{y2}(t) = \frac{1}{2} I\,(\cos(t) + I\sin(t)) - \frac{1}{2} I\,(\cos(t) - I\sin(t))$$

```
> simplify(%);
```

$$\mathrm{y2}(t) = -\sin(t)$$

Naturalmente si poteva procedere direttamente con il comando **dsolve**:

```
> dsolve({ode1, ode2, initconds}, {y1(t), y2(t)});
```

$$\{\mathrm{y1}(t) = \cos(t), \mathrm{y2}(t) = -\sin(t)\}$$