Einführung in Sage - Einheit 5

Datencontainer, Lineare Abbildungen, Eigenwert und Eigenvektoren

Jochen Schulz

Georg-August Universität Göttingen

Aufbau

Aufbau

Folgen - tuple

```
Folge = a,b,c,..
Folge = tuple(<sequence>)
```

- Ein Tuple ist eine Aneinanderreihung beliebiger Sage-Objekte, welche durch Kommata getrennt sind.
- ein Tuple ist in runde Klammern eingeschlossen.
- Tuple sind ein grundlegender Typ (Listen und Mengen bestehen daraus).
- Einfache Definition

```
t=var('a,b,c,d,e');Folge1 = a,b,c; Folge2 = (c,d,e);
    Folge1 ; Folge2
```

```
(a, b, c)
(c, d, e)
```

Konstruktion von Folgen

- Mit Listen erzeugte Folgen

```
tuple(i^2 for i in range(2,8))
```

```
(4, 9, 16, 25, 36, 49)
```

- Erzeugen von *n* identischen Objekten

```
tuple(sin(x) for i in range(0,5))
```

```
(sin(x),sin(x),sin(x),sin(x),sin(x))
```

- Erzeugen von *n* funktionalen Objekten

```
x = tuple(sin(i) for i in range(1,5)); x
```

```
(sin(1), sin(2), sin(3), sin(4))
```

- Erzeugen einer leeren Folge

```
folge = (); folge2 = folge,2,3; folge2
```

```
((), 2, 3)
```

Folgen - Zugriff und Verknüpfungen

- Verbinden von Folgen

```
Folge3 = Folge1+Folge2; Folge3
```

```
(a, b, c, c, d, e)
```

- Zugriff auf Elemente

```
x = 1,2,3,4; x[2]
```

```
3
```

```
x[2]; x[0:2]
```

```
3
(1, 2)
```

Listen

```
liste = [a,b,c,...]
liste = list(<sequence>)
```

- Eine Liste ist eine geordnete Folge beliebiger Sage Objekte.
- eine Liste is in eckigen Klammern eingeschlossen.
- Matrizen werden als geschachtelte Listen definiert.
- Baut auf `tuple` auf.
- einfache Konstruktion:

```
Liste = [1,[1,2], Set([1,2,3]),x]; Liste
```

```
[1, [1, 2], {1, 2, 3}, (1, 2, 3, 4)]
```

Konstruktion von Listen

- Erzeugen von Listen mit funktionalen Objekten

```
Liste = [2^i for i in range(1,9)]; Liste
```

```
[2, 4, 8, 16, 32, 64, 128, 256]
```

- Leere Listen:

```
Liste = []; Liste
```

```
[]
```

Zugriff auf Listen

- Der Zugriff funktioniert genau wie bei Folgen.

```
Liste=[(x,i) for x in range(1,4) for i in range(0,x)
    ];  Liste
```

```
[(1, 0), (2, 0), (2, 1), (3, 0), (3, 1), (3, 2)]
```

```
Liste[3], Liste[5]
```

```
((3, 0), (3, 2))
```

- Zuweisung:

```
Liste[5] = 42; Liste
```

```
[(1, 0), (2, 0), (2, 1), (3, 0), (3, 1), 42]
```

Weitere Befehle für Listen I

- Entfernen eines Elements aus der Liste mit `pop()`

```
Liste = [a,b,c]; Liste.pop(1); Liste
```

```
[a, c]
```

- Entfernen eines Elements aus der Liste mit `remove()`

```
Liste = [a,b,c]; Liste.remove(b); Liste
```

```
[a, c]
```

- Anhängen eines Elementes mittels `append()`

```
Liste.append([3,4,5]); Liste
```

```
[a, c, [3, 4, 5]]
```

Weitere Befehle für Listen II

- Zusammenfügen von Listen mit dem +- und ∗-Operator.

```
Liste2 = Liste+[3,4,5]; Liste2
```

```
[a, c, [3, 4, 5], 3, 4, 5]
```

```
Liste*2
```

```
[a, c, [3, 4, 5], a, c, [3, 4, 5]]
```

- Sortieren von Listen mit `sort()`

```
Liste = [4,-23,1,3]; Liste.sort(cmp=lambda x,y: int
    (2*(abs(x)>abs(y))-1)); Liste
```

```
[1, 3, 4, -23]
```

Inklusionen

- Ist Objekt `Objekt` in der Liste `Liste` enthalten?

```
<Objekt> in <Liste>
```

Beispiel:

```
Liste = [x+1,a,x+1,sin(b)]
x+1 in Liste
```

```
True
```

- Position des Objektes in der Liste (bei Fehlen gibt es eine entsprechende Meldung)

```
<Liste>.index(<Objekt>)
```

Beispiel:

```
Liste.index(sin(b))
```

```
3
```

map()

```
map(<f>,<Liste>)
```

wendet Funktion `f` auf alle Elemente der Liste `Liste` an.

- Kapselung bei Funktionen mit mehreren Argumenten.
- Beispiele:

```
map(sin,[x,1,0,pi,0.3])
```

```
[sin(x), sin(1), 0, 0, 0.295520206661340]
```

```
map(max,[2,3,4,5,6,7],[8,7,6,5,4,3])
```

```
[8, 7, 6, 5, 6, 7]
```

map_threaded()

```
map_threaded(<f>,<Liste>)
```

führt die Funktion `f` rekursiv auf alle Elemente in der Liste `Liste` an.

- Beispiele:

```
map_threaded(sin,[x,[1,0],[pi,[0.3]]])
```

```
[sin(x), [sin(1), 0], [0, [0.29552020666134]]]
```

```
map_threaded(sin,matrix([[1,2,3],[4,5,6]]))
```

```
[sin(1) sin(2) sin(3)]
[sin(4) sin(5) sin(6)]
```

filter()

Ein Filter erzeugt eine Teilmenge aus einer größeren Menge.

```
M1 = filter(<f>,<M>)
```

- `f(x)` ist eine Abbildung auf die Boolschen Werte `True`/`False`.
- `M1` ist die Teilmenge die aus den Elementen $x \in M$ besteht, für die `f(x)` eine wahre Aussage ergibt.

Beispiele für filter

```
filter(bool,[1==1, 1==2, 3==3, 4==5, 7==7])
```

```
[True, True, True]
```

```
var('x,y,z,a')
def f(x): return a in x.operands()
filter(f,[a+2,x,y,z,sin(a)])
```

```
[a + 2, sin(a)]
```

zip()

```
zip(<Liste1>, <Liste2>)
```

die Elemente zweier Listen werden paarweise zu einer neuen Liste verknüpft.

Beispiel:

```
Liste1 = [a,b,c]; Liste2 = [e,f,g]
zip(Liste1,Liste2)
```

```
[(a, e), (b, f), (c, g)]
```

Wörterbücher (Dictionaries)

```
d = {<Index1>:<Wert1>,<Index2>:<Wert2>,...}
```

- Dictionaries bestehen aus Assoziationen der Form `<Index>:<Wert>`.
- Zugriff:

```
d[<Index>]
```

- Sind gut geeignet für das Speichern großer Datenmengen, der indizierte Zugriff sehr schnell ist.
- der `Index` ist eindeutig

Konstruktion von Dictionaries

- Erzeugen eines leeren Dictionaries

```
T = {}; T
```

```
{}
```

- Einträge können durch Zuweisungen der Form `<Dict>[<Index>] = <Wert>` erzeugt oder verändert werden.

```
T[f] = sin(x); T[1,2] = 5
T[1,2,3] = [a,b,c]; T[a] = d;
T
```

```
{(1, 2): 5, f: sin(x), a: d, (1, 2, 3): [a, b, c]}
```

Zugriff auf Dictionaries

- Zugriff auf ein Dictionary

```
T[a],T[1,2],T[c]
```

```
(d, 5, d)
```

```
float(T[f](x=4))
```

```
-0.7568024953079282
```

Wird ein `Index` nicht gefunden, gibt es eine Fehlermeldung

- Löschen von Einträgen mit `pop()`

```
T.pop(a)
```

Befehle (nicht) wie bei Listen

Alle Funktionen wirken erstmal direkt auf den Index, nicht den Werten:

- `<a> in <M>`: Es wird überprüft, ob der Index `a` in einem Dictionary vorkommt.
- `filter()`: Filtert die Dictionary-Indizes nach Kriterien.
- `map()`: Wendet eine Funktion auf die Indizes an.

Man kann aber die Objektfunktion `values()` benutzen um eine Liste der Werte zu bekommen (Dann hat man allerdings kein Dictionary mehr!)

Beispiele

- `in` prüft ob `a` oder `b` als Index in einem Dictionary auftaucht.

```
Z = {}; Z[a] = b; Z[c] = d; Z[x] = b
a in Z, b in Z
```

```
(True, False)
```

`a` taucht als Index auf, `b` nur als Wert.

- `filter` prüft ebenfalls nur den Index.

```
f(y) = y == c; filter(f,Z)
```

```
[c]
```

```
f(y) = y == d; filter(f,Z.values())
```

```
[d]
```

Dictonaries - `update()`

- Dictionaries können aneinander gehängt werden:

```
T={1:a,2:b}; S={3:c,4:d}
T.update(S); T
```

```
{1: a, 2: b, 3: c, 4: d}
```

Vorsicht: Doppelt auftretenden Indizes werden überschrieben!

Aufbau

Lineare Abbildungen

Seien K-Vektorräume V und W gegeben. Eine Abbildung

$$F: \; V \to \; W$$

heißt linear, falls für $v, w \in V$ und $\lambda \in K$ gilt:

(L1) $F(v + w) = F(v) + F(w)$

(L2) $F(\lambda \cdot v) = \lambda \cdot F(v)$

- Isomorphismus: F bijektiv.
- Endomorphismus: $V = W$.
- Automorphismus: $V = W$ und F bijektiv.

Bemerkungen

- Sei $(v_i)_{i\in I}$ eine Basis in V und $(w_i)_{i\in I}$ seien Vektoren in W. Dann gibt es genau eine lineare Abbildung $F: V \to W$ mit $F(v_i) = w_i$ für alle $i \in I$.
- Bild von F: $Im(F) = F(V) := \{F(v), v \in V\}$.
- Kern von F: $Ker(F) := \{v \in V \mid F(v) = 0\}$
- Kern und Bild sind Untervektorräume.
- Dimensionsformel:

$$\dim V = \dim F(V) + \dim Ker(F)$$

- $\mathrm{Hom}_K(V, W)$: Die Menge der linearen Abbildungen von V nach W. Sie ist ein Vektorraum durch punktweise Addition und Skalarmultiplikation.

Lineare Abbildungen und Matrizen

- Jeder Matrix $A \in K^{m \times n}$ läßt sich durch

$$L_A : K^n \to K^m, \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \longmapsto A \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

  eine lineare Abbildung zuordnen.
- Es gilt $\dim(L_A(K^m)) = \mathrm{Rang}(A)$.

Koordinatenvektor

Sei V ein K-Vektorraum mit Basis $\mathcal{V} = (v_1, \ldots, v_n)$.

- Die lineare Abbildung $\Phi_{\mathcal{V}} : K^n \to V$ mit

$$\Phi_{\mathcal{V}}(x_1, \ldots, x_n) = x_1 v_1 + \cdots + x_n v_n$$

  ist ein Isomorphismus. Man nennt $\Phi_{\mathcal{V}}$ ein Koordinatensystem in V und $x = (x_1, \ldots, x_n) = \Phi_{\mathcal{V}}^{-1}(v)$ den Koordinatenvektor zu $v \in V$.
- Basiswechselabbildung von $\mathcal{V}$ nach Basis $\mathcal{Z}$:

$$T := \Phi_{\mathcal{Z}}^{-1} \circ \Phi_{\mathcal{V}}$$

.

Isomorphismus

Seien K-Vektorräume V und W mit Basen $\mathcal{V} = (v_1, \ldots, v_n)$ und $\mathcal{W} = (w_1, \ldots, w_m)$ gegeben.
Für eine Matrix $A \in K^{m\times n}$ wird durch

$$\begin{array}{rcl} F(v_1) & := & a_{11}w_1 + \cdots + a_{m1}w_m \\ \vdots & & \vdots \\ F(v_n) & := & a_{1n}w_1 + \cdots + a_{mn}w_m \end{array}$$

eine lineare Abbildung F definiert. Dies ergibt einen Isomorphismus

$$L^{\mathcal{V}}_{\mathcal{W}} : K^{m\times n} \to \mathrm{Hom}_K(V, W), \quad A \mapsto F.$$

Kanonisches Beispiel

Seien K^n und K^m mit den kanonischen Basen $\mathcal{K}_n$ und $\mathcal{K}_m$ versehen.

- Die Abbildungen $\Phi_{\mathcal{K}_n}$ und $\Phi_{\mathcal{K}_m}$ sind Identitäten.
- Die Abbildung $L_{\mathcal{K}_m}^{\mathcal{K}_n}$ ist gegeben durch

$$L_{\mathcal{K}_m}^{\mathcal{K}_n}(A)(x) = Ax, \; x \in K^n.$$

- Die Spaltenvektoren von A sind die Bilder der Einheitsvektoren unter der Abbildung $L_{\mathcal{K}_m}^{\mathcal{K}_n}(A)$.

Kommutierendes Diagramm

Seien K-Vektorräume V und W mit Basen $\mathcal{V} = (v_1, \ldots, v_n)$ und $\mathcal{W} = (w_1, \ldots, w_m)$ und eine lineare Abbildung F gegeben. Dann gilt das folgende kommutierende Diagramm:

$$\begin{tikzcd} V \arrow[r, "F"] & W \\ K^n \arrow[u, "\Phi_V"] \arrow[r, "(L_{\mathcal{W}}^{\mathcal{V}})^{-1}(F)"'] & K^m \arrow[u, "\Phi_W"'] \end{tikzcd}$$

Drehung

Drehung um den Winkel α - Drehmatrix G:

$$G(\alpha) := \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}$$

```
var('a,b');A = matrix([[cos(a),-sin(a)],[sin(a),cos(a)]])
A(a=pi/2)*vector([1,1])
```

```
(-1, 1)
```

Spiegelung

Spiegelung bezüglich der Ebene

$$H(a) := \{x \in \mathbb{R}^3 | x^T a = 0\}, \|a\| = 1$$

durch

$$S(a) := I - 2aa^T.$$

```
a = matrix(3,1,[1,2,3])
a = a/norm(a)
I_n = identity_matrix(3)
S = I_n - 2*a*a.transpose()
norm(S*S-I_n)
```

```
0.0
```

Aufbau

Eigenwerte und Eigenvektoren

Sei $A \in K^{n \times n}$. Ein Element $\lambda \in K$ heißt Eigenwert von A, wenn ein $x \in K^n \setminus \{0\}$ existiert,

$$Ax = \lambda x$$

gilt. Der Vektor $x \in K^n$ heißt Eigenvektor zum Eigenwert λ.

- Die Eigenwerte sind die Nullstellen des charakteristischen Polynoms

$$p(t) := \det(A - t\, I_n).$$

- Es gibt höchstens n Eigenwerte.

Bemerkungen

- Eigenvektoren zu paarweise verschiedenen Eigenwerten sind linear unabhängig.
- Gibt es eine Basis aus Eigenvektoren, so ist A diagonalisierbar, d.h. man kann die Abbildung L_A bei geeigneter Basiswahl durch eine Diagonalmatrix repräsentieren.
- Jeder Endomorphismus eines komplexen Vektorraums läßt sich durch eine Matrix in Jordanscher Normalform darstellen.

Eigenwerte in Sage

- Bestimmung von Eigenwerten

```
_=var('al');A = matrix([[cos(al), sin(al)],[sin(al),-
    cos(al)]])
[ m.full_simplify() for m in  A.eigenvalues()]
```

```
[-1, 1]
```

- Bestimmung von Eigenvektoren

```
A.eigenvectors_right()
```

Eigenwerte in Sage

- Bestimmung des charakteristischen Polynoms

```
E = identity_matrix(2)
p = (A-x*E).det()
A.charpoly(); p
```

```
(x - cos(al))*(x + cos(al)) - sin(al)^2
```

```
[m.full_simplify() for m in solve(p==0,x)]
```

```
[x == -1, x == 1]
```

Lineare Gleichungssysteme (LGS)

Sei $A \in K^{m \times n}$ und $b \in K^m$. Gesucht ist die Menge der Lösungen (Lösungsraum):

$$\{x \in K^n \mid Ax = b\}$$

- Ist $b = 0$, so spricht man von einem homogenen System. Ansonsten spricht man von einem inhomogenen System.
- Der Lösungsraum W des homogenen Systems bildet einen Untervektorraum des K^n. Die Dimension ist

$$\dim(W) = n - \operatorname{rang}(A).$$

Struktur des Lösungsraums

- affiner Unterraum $X \subset K^n$: wenn ein Unterraum W von K^n und ein $v \in K^n$ existiert, so dass
$$X = v + W$$
- Die Lösungen des inhomogenen Systems ($b \neq 0$) bilden einen affinen Unterraum des K^n.
- Ist W der Lösungsraum des homogenen Systems und $v \in K^n$ eine beliebige Lösung von $Ax = b$, dann ist der Lösungsraum X von $Ax = b$ gegeben durch $X = v + W$.
- Zwei Lösungen des inhomogenen Systems unterscheiden sich durch eine Lösung des homogenen Systems.

Lösbarkeit

- Das inhomogene System ist genau dann für alle b lösbar, wenn $\mathrm{rang}(A) = m$ gilt.
- Das homogene bzw. das inhomogene System besitzt höchstens eine Lösung, genau dann wenn $\mathrm{rang}(A) = n$ gilt.
- Der Lösungsraum des inhomogenen Systems ist genau dann nicht leer, wenn $\mathrm{rang}(A) = \mathrm{rang}(A, b)$ gilt.
- Praktisch kann ein LGS mit dem Gausschen Eliminationsverfahren gelöst werden.

LGS in Sage

Berechnung der Lösungen von $Ax = b$:

```
A = matrix([[1,2,3],[4,5,6],[7,8,9]])
b1 = vector([0,0,0])
b2 = vector([1,0,0])
b3 = A*b2
print A\b1
print A\b3
```

```
(0, 0, 0)
(1, 0, 0)
```