Kısa Sage Kılavuzu: Doğrusal Cebir

Robert A. Beezer (Türkçeleştiren Kürşat Aker)

Sage Sürüm 3.4

`http://wiki.sagemath.org/quickref`

P. Jipsen ve W. Stein'ın çalışmalarını esas alarak

Vektörlerin Tanımlanması

Uyarı: Bir vektörün ilk indisi 0'dır.

`u = vector(QQ, [1, 3/2, -1])` $= (1, \frac{3}{2}, -1) \in \mathbb{Q}^3$

`v = vector(QQ, {2:4, 95:4, 210:0})`

toplam 211 kutu, 2. ve 95. kutular = 4, seyrek

Vektör İşlemleri

`u = vector(QQ, [1, 3/2, -1])`

`v = vector(ZZ, [1, 8, -2])`

`2*u - 3*v` doğrusal bileşim

`u.dot_product(v)`

`u.dot_product(v)` iç çarpım

`u.cross_product(v)` sıralama: `u×v`

`u.inner_product(v)` u'nun anasından gelen iç çarpım

`u.pairwise_product(v)` $= (u_0v_0, u_1v_1, u_2v_2, \ldots)$

`u.norm() == u.norm(2)` Öklid normu

`u.norm(1)` kutu içeriklerinin toplamı

`u.norm(Infinity)` en büyük kutu içeriği

`A.gram_schmidt()` `A` matrisinin satırlarını dönüştürür

Matrislerin Tanımlanması

Uyarı: Satır ve sütunların isimlendirilmesi 0'dan başlar

`A = matrix(ZZ, [[1,2],[3,4],[5,6]])`

Tamsayılar üzerinde 3×2-lik matris

`B = matrix(QQ, 2, [1,2,3,4,5,6])`

Listeden 2-şer satır al, kesirlerde 2×3-lük matris yap

`C = matrix(CDF, 2, 2, [[5*I, 4*I], [I, 6]])`

kutuları karmaşık sayılar, 53-bit hassasiyet

`Z = matrix(QQ, 2, 2, 0)` sıfır matris

`D = matrix(QQ, 2, 2, 8)`

köşegendeki terimler 8, geri kalanlar 0

`I = identity_matrix(5)` 5×5-lik birim matris

`J = jordan_block(-2,3)`

3×3-lük matris, köşegende -2, köşegenüstünde 1'ler

`var('x y z'); K = matrix(SR, [[x,y+z],[0,x^2*z]])`

Simgesel ifadeler, `SR` halkasında yaşarlar

`L=matrix(ZZ, 20, 80, {(5,9):30, (15,77):-6})`

20×80-lik matris, iki tane $\neq 0$ kutu, seyrek temsil

Matris Çarpımı

`u = vector(QQ, [1,2,3]), v = vector(QQ, [1,2])`

`A = matrix(QQ, [[1,2,3],[4,5,6]])`

`B = matrix(QQ, [[1,2],[3,4]])`

`u*A, A*v, B*A, B^6, B^(-3)`'nün tümü geçerlidir.

`B.iterates(v, 6)` $vB^0, vB^1, \ldots, vB^5$'u üretir

`rows = False` v vektörünü matrislerin sağına koyar

`f(x)=x^2+5*x+3` tanımlıysa, `f(B)` geçerlidir

`B.exp()` $= e^B = \sum_{k=0}^{\infty} \frac{B^k}{k!}$

Matris Uzayları

`M = MatrixSpace(QQ, 3, 4)`

3×4-lük matrislerin 12 boyutlu uzayı

`A = M([1,2,3,4,5,6,7,8,9,10,11,12])`

3×4-lük bir matris, `M` uzayının bir üyesidir

`M.basis()` M'nin tabanı

`M.dimension()` M'nin boyutu

`M.zero_matrix()` M'nin sıfır matrisi

Matris İşlemleri

`5*A+2*B` doğrusal bileşim

`A.inverse(), A^(-1), ~A` = A'nın tersi

Eğer `A` tersinir değilse, `ZeroDivisionError` hatası gelir

`A.transpose()` $= A^t$ = A'nın devriği

`A.antitranspose()` A^t'nin satır ve sütunları ters sıralı

`A.adjoint()` Kofaktörler matrisi

`A.conjugate()` kutu kutu karmaşık eşlenikler

`A.restrict(V)` A'nın korunan `V` altuzayına kısıtlaması

Satır/Sütun İşlemleri

Satır İşlemleri: (matrisi değiştirirler)

Uyarı: İlk satırın indisi 0'dır

`A.rescale_row(i,a)` `a*`(`i` satırı)

`A.add_multiple_of_row(i,j,a)` `a*`(`j` satırı) + `i` satırı

`A.swap_rows(i,j)` `i` satırıyla `j` satırı yer değiştirir

Her satır işleminin sütun karşılığı vardır, `row`→`col`

Yeni bir matris için, örn. `B = A.with_rescaled_row(i,a)`

Matrisleri Merdiven Haline Getirilmesi

`A.echelon_form()`, `A.echelonize()`, `A.hermite_form()`

Uyarı: Sonuçlar, tanım halkasına bağlıdır

`A = matrix(ZZ,[[4,2,1],[6,3,2]])`

`B = matrix(QQ,[[4,2,1],[6,3,2]])`

`A.echelon_form()`	`B.echelon_form()`
$\begin{pmatrix} 2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$

`A.pivots()` sütun uzayını geren sütunların indisleri

`A.pivot_rows()` satır uzayını geren satırların indisleri

Matrislerin Parçaları

Uyarı: Satır ve sütunların isimlendirilmesi 0'dan başlar

`A.nrows()`

`A.ncols()`

`A[i,j]` `i`-nci satır ve `j`-nci sütundaki kutunun içeriği

Uyarı: Doğru: `A[2,3] = 8`, Yanlış: `A[2][3] = 8`

`A[i]` `i` satırı (değiştirilemez bir Python destesi olarak)

`A.row(i)` `i` satırı (Sage vektörü)

`A.column(j)` `j` sütunu (Sage vektörü)

`A.list()` Matris, satır öncelikli dizilmiş (Python listesi)

`A.matrix_from_columns([8,2,8])`

seçilen sütunlardan matris yap (sütunlar yinelenebilir)

`A.matrix_from_rows([2,5,1])`

seçilen satırlardan matris yap (satırlar sırasız olabilir)

`A.matrix_from_rows_and_columns([2,4,2],[3,1])`

satır ve sütun numaralı kutulardan yeni matris yap

`A.rows()` tüm satırlar (destelerden bir liste halinde)

`A.columns()` tüm sütunlar (destelerden bir liste halinde)

`A.submatrix(i,j,nr,nc)`

sol üst köşesi `(i,j)` kutusu olan `nr` × `nc` altmatris

`A[2:4,1:7]`, `A[0:8:2,3::-1]` Python tarzı liste kesiti

Matrisleri Birleştirme

`A.augment(B)` = `[A|B]`, `A` matrisi `B`'nin soluna konur

`A.stack(B)` `A` matrisi `B`'nin üzerine konur

`A.block_sum(B)` `A` solüstte, `B` sağaltta blok köşegen mat.

`A.tensor_product(B)` `B`'nin katları `A`'ya uygun dizilmiş

Matrislere Sayı Atayan Fonksiyonlar

`A.rank()`

`A.nullity() == A.left_nullity()`

`A.right_nullity()`

`A.determinant() == A.det()`

`A.permanent()`

`A.trace()`

`A.norm() == A.norm(2)` Öklid normu

`A.norm(1)` en büyük sütun toplamı

`A.norm(Infinity)` en büyük satır toplamı

`A.norm('frob')` Frobenius normu

Matrislerin Özellikleri

`.is_zero()` (sıfır mı?), `.is_one()` (birim matris mi?), `.is_scalar()` (birimin katı mı?), `.is_square()`, `.is_symmetric()`, `.is_invertible()`, `.is_nilpotent()`

Özdeğerler

`A.charpoly('t')` $= \det(\mathtt{t} - \mathtt{A})$ (varsayılan değişken `x`)
  `A.characteristic_polynomial() == A.charpoly()`
`A.fcp('t')` çarpanlarına ayrılmış karakteristik polinom
`A.minpoly()` minimum polinom
  `A.minimal_polynomial() == A.minpoly()`
`A.eigenvalues()` Özdeğerlerin listesi (sırasız, katlılık)
`A.eigenvectors_left()` vektörler solda (sağ →`_right`)
  Özdeğerlere karşılık üçlülerden `(e,V,n)` bir liste verir:
    `e`: Özdeğer
    `V`: Özuzayın bir tabanı
    `n`: Cebirsel Katlılık
`A.eigenmatrix_right()` vektörler sağda `_left` too
  İki matris verir:
    `D`: özdeğerlerden oluşan köşegen matris
    `P`: sütunları `A`'nın özvektörleri (`_left` satırlar özvek.)
      eğer `A` köşegenlenemezse, sıfır sütunları var

Matris Ayrışmaları

Önemli: Bu yöntemler, uygun halkalarda tanımlıdır.
`A.jordan_form(transformation=True)`
  iki matris verir:
    `J`: Jordan blokların oluşan bir matris
    `P`: tersinir bir matris
    öyle ki,   `A == P^(-1)*J*P`
`A.smith_form()` üç matris verir:
  `D`: köşegen boyunca temel bölenler
  `U, V`: determinantları tersinir
  öyle ki,   `D == U*A*V`
`A.LU()` üç matris verir:
  `P`: bir permütasyon matrisi
  `L`: alt-üçgen matris
  `U`: üst-üçgen matris
  öyle ki,   `P*A == L*U`
`A.QR()` iki matris verir:
  `Q`: dik matris
  `R`: üst-üçgen matris
  öyle ki,   `A == Q*R`
`A.SVD()` üç matris verir:
  `U`: dik matris
  `S`: köşegen dışında sıfır, `A` ile aynı boyutlarda
  `V`: dik matris
  öyle ki,   `A == U*S*(V'nin devrik eşleniği)`
`A.symplectic_form()`
`A.hessenberg_form()`
`A.cholesky()`

Doğrusal Denklem Takımlarının Çözümleri

`A.solve_right(B)`, `A.solve_left(B)` sırasıyla `A*X = B`
  ve `X*A = B` denklemlerini çözer (`X` vektör veya matris)
`A = matrix(QQ, [[1,2],[3,4]])`
`b = vector(QQ, [3,4])`
  için `A*x = b`'nin çözümü `A\b`$= \mathtt{A}^{-1}\mathtt{b} =$`(-2, 5/2)`'tir.

Doğrusal Uzaylar

`U = VectorSpace(QQ, 4)` kesirler üstünde 4 boyutlu uzay
`V = VectorSpace(RR, 4)` "cisim" 53-bit hassas gerçeller
`W = VectorSpace(RealField(200), 4)`
  "cisim" 200-bit hassaslıkta
`X = CC^4` 4-boyutlu, 53-bit hassas karmaşıklar
`Y = VectorSpace(GF(7), 4)` sonlu doğrusal uzay
  `Y.finite()` sorusu Doğru (`True`) yanıtı verir
  `len(Y.list())` $= 7^4 = 2401 =$ `Y`'nin üye sayısı

Doğrusal Uzayların Özellikleri

`V.dimension()` `V`'nin boyutu
`V.basis()` `V`'nin tabanı
`V.echelonized_basis()`
`V.has_user_basis()` kullanıcı `V` için bir taban vermiş mi?
`V.is_subspace(W)` `W`, `V`'nin altuzayı mı?
`V.is_full()` modül olarak, rank mertebeye eşit mi?
`Y = GF(7)^4`, `T = Y.subspaces(2)`
  `T`, `Y`'nin 2 boyutlu altuzaylarını üreten nesne
  `[U for U in T]` `Y`'nin 2 boyutlu altuzay listesi (=2850)

Altuzaylar

`span([v1,v2,v3], QQ)` `v1,v2,v3`'ün gerdiği altuzay/`QQ`

Bir `A` matrisinden başlayarak,
  tanım halkası cisim ise, doğrusal uzaylar
  tanım halkası yalnızca halka ise, modüller üretilir
`A.left_kernel() == A.kernel()` sağ yan için `right_`
`A.row_space() == A.row_module()`
`A.column_space() == A.column_module()`
`A.eigenspaces_right()` vektörler sağda, sol için `_left`
  Pairs, having eigenvalue with its right eigenspace

Eğer `V` ve `W` altuzaylarsa,
`V.quotient(W)` $=$ `V/W` $=$ `V`'nin `W` altuzayı ile bölümü
`V.intersection(W)` $= \mathtt{V} \cap \mathtt{W} =$ `V`'nin `W` ile kesişimi
`V.direct_sum(W)` $= \mathtt{V} \oplus \mathtt{W} =$ `V` ile `W`'nın direkt toplamı
`V.subspace([v1,v2,v3])`

Yoğun ya da Seyrek

Not: Algoritmalar, nesnelerin temsil şekline göre değişir
Vektörler ve matrislerin iki tür temsili vardır:
  Yoğun: listeler (vektör) ve listelerin listeleri (matris)
  Seyrek: Python sözlükleri
`.is_dense()` yoğun mu?, `.is_sparse()` seyrek mi?
`A.sparse_matrix()` `A`'nın seyrek temsili
`A.dense_rows()` (yoğun) satır vektörleri olarak `A`
Bazı komutlarda `sparse` seçeneğini seçebilirsiniz

Halkalar

Not: Farklı halkalar üzerinde farklı algoritmalar kullanılır
<nesne>`.base_ring(R)` vektör, matris gibi bir `nesne`
  için üzerinde tanımlandığı halkayı verir
<nesne>`.change_ring(R)` vektör, matris gibi bir `nesne`
  için üzerinde tanımlandığı halkayı (cismi) değiştirir
`R.is_ring()` halka mı? , `R.is_field()` cisim mi?
`R.is_integral_domain()`, `R.is_exact()`

Bazı Halkalar ve Cisimler
  `ZZ` tamsayılar, halka
  `QQ` kesir sayılar, cisim
  `QQbar` cebirsel cisim, kayıpsız
  `RDF` real double field, kayıpsız
  `RR` 53-bit hassas gerçeller, kayıplı
  `RealField(400)` 400-bit hassas gerçeller, kayıpsız
  `CDF`, `CC`, `ComplexField(400)` karmaşıklar
  `RIF` gerçel aralık cismi
  `GF(2)` mod 2, cisim, özel algoritmalar
  `GF(p) == FiniteField(p)` `p` asal, sonlu cisim
  `Integers(6)` mod 6'da tamsayılar, yalnızca halka
  `CyclotomicField(7)` 1'in 7. kökleri eklenmiş kesirler
  `QuadraticField(-5, 'x')` kesirlere $\mathtt{x}=\sqrt{-5}$ eklenmiş
  `SR` simgesel ifadeler halkası

Doğrusal Uzaylarla Modüller

Bir halka üzerinde tanımlanan doğrusal uzaya modül denir
Yukarıdaki pek çok komut modüller için de geçerlidir
Bazı "vektörler" de aslında bir modülün elemanlarıdır

Biraz Daha Yardım

Kısmen yazılmış bir komut "tab"a basarak tamamlanır
`nesne.` 'den sonra "tab", ilgili tüm yöntemleri gösterir
<komut>? özet ve örnekler
<komut>?? tüm kaynak kodu