Module **Algebra**

Filename: Algebra.py Author: Rafel Amer (rafel.amer AT upc.edu) Copyright: Rafel Amer 2020–2021 Disclaimer: This program is provided "as is", without warranty of any kind, either expressed or implied, including, but not linmited to, the implied warranties of merchantability and fitness for a particular purpose. It has been written to generate random models of exams for the subject of Linear Algebra at ESEIAAT, Technic University of Catalonia License: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

```
            See <https://www.gnu.org/licenses/>
```

Functions

def matriu_latex(m, format=None, ampliada=False, tipus='p')

Retorna l'expressió en latex d'una matriu del tipus Matrix del sympy

Parametres

format: format de les columnes de la matriu. Per defecte "r" ampliada: si es vol separar amb una línia vertical l'última columna de la matriu

def matriu_mathematica(m)

Retorna l'expressió en Mathematica d'una matriu del tipus Matrix del sympy

def mcd_llista(list)

Retorna el màxim comú divisor d'una llista d'enters

def mcm_llista(list)

Retorna el mínim comú múltiple d'una llista d'enters

def mti(i, j)

Funció auxiliar per crear una matriu triangular superior i uns o menys uns a la diagonal Retorna zero si el coeficients està per sota de la diagonal principal

def mts(i, j, values)

Funció auxiliar per crear una matriu triangular inferior i uns o menys uns a la diagonal Retorna zero si el coeficients està per sobre de la diagonal principal

def mylatex(e)

def norma_maxim(m)

Retorna el màxim en valor absolut d'entre els coeficients d'una matriu del tipus Matrix del sympy

def nzeros(m)

Retorna el nombre de zeros d'una matriu del tipus Matrix del sympy

def primer_no_nul(list)

Retorna l'índex del primer coeficient no nul d'una llista

def vaps_veps(result)

Retorna la llista valors propis i els seus vectors propis.

Paràmetres

result: resultat de la funció eigenvects() del sympy

`def vaps_veps_amb_signe(result, signe=1)`

Donada una matriu A del sympy, result ha de ser el resultat de la funció r = A.eigenvects(). Aleshores aquesta funció retorna la llista valors propis positius (signe > 0) o negatius (signe < 0) i els seus vectors propis.

Paràmetres

result: resultat de la funció eigenvects() del sympy signe: positiu o negatiu, en funció de quins valors i vectors propis es volen

`def vectors_latex(l, sep=False)`

Retorna la llista de vectors l escrita en latex

Paràmetres

l: llista de vectors o punts sep: Si és False, retorna (1,2,3),(3,5,2),(1,-2,3) Si és True, retorna $(1,2,3)$, $(3,5,2)$ i $(1,-2,3)$

Classes

`class Base (vecs, unitaria=False)`

Classe que ens permetrà representar bases de R^n

Atributs

vecs: una llista de n vectors de R^n dimensio: el valor de n unitaria: En funcio de si volem imprimir els seus vectors unitaris o no

Static methods

`def aleatoria(dimensio=3, maxim=5, unitaria=False, mzeros=-1)`

Retorna una base aleàtoria

Paràmetres

dimensio: Dimensió de l'espai corresponent maxim: Nombre màxim de les components dels seus vectors unitaria: Si el determinant ha de ser 1 o -1 mzeros: Nombre màxim de zeros entre les components dels seus vectors

`def canonica(dimensio=3)`

Retorna la base canònica

Paràmetres

dimensio: Dimensió de l'espai corresponent

`def from_matriu(m)`

Crea una nova base a partir d'una matriu de la classe Matriu Si la matriu no és quadrada o no té rang màxim, retorna None

`def ortogonal(ordre=3, maxim=5, unitaria=False)`

Retorna una base ortogonal "aleatòria"

Paràmetres

ordre: dimensió maxim: màxim per a les components dels vectors de la base unitaria: si és True, la base serà ortonormal

Methods

`def canvi_de_base_a_la_base(self, B, p1=1, p2=0)`

Retorna en format latex l'expressió del canvi de base de la base actual a la base B

Paràmetres

p1: primes que s'escriuran a les components en la base actual p2: primes que s'escriuran a les components en la base B

`def components_del_vector(self, vec, base=None)`

Retorna un nou vector expressat en aquesta base del vector que en la base "base" té components "vec"

Paràmetres

vec: components en la base "base" base: Base

`def es_ortogonal(self)`

Retorna si la base és ortogonal

`def es_unitaria(self)`

Retorna si la base és unitària. Notem que els vectors no es guarden com a unitaris.

`def matriu(self)`

Retorna la matriu de la classe Matriu que té per columnes els vectors de la base

`def orientacio_positiva(self)`

Fa que tingui orientació positiva canviant, si cal, de signe l'últim vector

`def quadrats_longituds(self)`

Retorna els quadrats de les longituds dels vectors de la base sense tenir en compte si la base és unitària

`def set_unitaria(self)`

Si la matriu és ortogonal, la passa a unitària

`def te_orientacio_positiva(self)`

Retorna si té orientació positiva

`def vector_de_components(self, vec)`

Retorna un nou vector expressat en la base canònica del vector que en aquesta base té components "vec"

Paràmetres

vec: vector

`def vectors(self, unitaris=False)`

Retorna els vectors la base

Paràmetres

unitaris: si és True, els divideix per la seva longitud

`def vectors_latex(self)`

Retorna l'expressió en latex dels vector de la base, sense les claus inicial i final

`class CilindreElliptic (a2, b2, centre, eix1, eix2)`

Classe per treballar amb cilindres el·líptics

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un cilindre el·líptic de manera aleatòria

Methods

def centre(self)

Retorna un centre del cilindre el·líptic

def centres(self)

Retorna la recta de centres

def equacio_reduida(self)

Retorna l'equacio reduïda del cilindre el·líptic

def semieixos(self)

Retorna els semieixos del cilindre el·líptic

def semieixos_quadrats(self)

Retorna els semieixos al quadrat del cilindre el·líptic

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class CilindreHiperbolic (a2, b2, centre, eix1, eix2)

Classe per treballar amb cilindres hiperbòlic

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un cilindre hiperbòlic de manera aleatòria

Methods

def centre(self)

Retorna un centre del cilindre hiperbòlic

def centres(self)

Retorna la recta de centres

def equacio_reduida(self)

Retorna l'equacio reduïda del cilindre hiperbòlic

def semieixos(self)

Retorna els semieixos del cilindre hiperbòlic

def semieixos_quadrats(self)

Retorna els semieixos al quadrat del cilindre hiperbòlic

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class CilindreParabolic (p, vertex, eix1, eix2=None)

Classe per treballar amb cilindres parabòlics

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un cilindre parabòlic de manera aleatòria

Methods

def equacio_reduida(self)

Retorna l'equacio reduïda del cilindre parabòlic

def parametre(self)

Retorna el paràmetre de la paràbola

def vertex(self)

Retorna l'origen de la referència principal

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class Con (a2, b2, c2, centre, eix1, eix2)

Classe per treballar amb cons

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un hiperboloide de dues fulles de manera aleatòria aleatòria

Methods

def centre(self)

Retorna el centre o vèrtex del con

def equacio_reduida(self)

Retorna l'equacio reduïda del con

def semieixos(self)

Retorna els semieixos del con

def semieixos_quadrats(self)

Retorna els semieixos al quadrat del con

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class Conica (matriu, ref=None)

Classe per treballar amb còniques. L'objectiu no és classificar còniques, sinó generar coniques a partir dels elements característics o de manera aleatòria.

Atributs

ref: referència afí matriu: matriu projectiva de la cònica en la referència "ref" canonica: matriu projectiva de la cònica en la referència canònica

Subclasses

Ellipse, Hiperbola, Parabola

Static methods

def aleatoria(maxim=30)

Retorna una el·lipse, hipèrbola o paràbola aleatòries

Paràmetres

maxim: nombre màxim de la matriu projectiva de la cònica

def ellipse(maxim=30)

Retorna una el·lipse aleatòria

Paràmetres

maxim: nombre màxim de la matriu projectiva de la cònica

def from_equacio(eq)

Retorna i classifica la cònica a partir de la seva equació. Només per a el·lipses, hipèrboles i paràboles

def hiperbola(maxim=30)

Retorna una hipèrbola aleatòria

Paràmetres

maxim: nombre màxim de la matriu projectiva de la cònica

def parabola(maxim=30)

`def parabola(maxim=30)`

Retorna una paràbola aleatòria

Paràmetres

maxim: nombre màxim de la matriu projectiva de la cònica

Methods

`def equacio(self)`

Retorna l'equació en latex de l'equació de la quàdrica

`def referencia_principal(self)`

Retorna la referencia principal

`def tipus(self)`

Retorna el tipus de cònica

`def vectors(self, unitaris=False)`

Retorna els vectors de la base de la referència principal

Paràmetres

unitaris: si és True, els retorna unitaris

`class Ellipse (a2, b2, centre, eix)`

Classe per treballar amb el·lipses

Ancestors

Conica

Static methods

`def aleatoria()`

Retorna una el·lipse aleatòria

Methods

`def centre(self)`

Retorna el centre de la el·lipse

`def equacio_reduida(self)`

Retorna l'equacio reduïda de l'el·lipse en format LaTeX

`def focus(self)`

Retorna els focus de l'el·lipse

`def semidistancia_focal(self)`

Retorna la simidistància focal

`def semieix_major(self)`

Retorna el semieix major

`def semieix_menor(self)`

def semieix_menor(self)

Retorna el semieix menor

def to_asy(self, x=13, y=10)

Retorna una expressió per fer servir amb el programa Asymtote

Paràmetres

x, y: nombres enters. El gràfic es representarà en una quadricula de límits (-x,x) i (-y,y)

def vertexs(self)

Retorna els vèrtex de l'el·lipse

Inherited members

Conica: `ellipse`, `equacio`, `from_equacio`, `hiperbola`, `parabola`, `referencia_principal`, `tipus`, `vectors`

class Ellipsoide (a2, b2, c2, centre, eix1, eix2)

Classe per treballar amb el·lipsoides

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un el·lipsoide de manera aleatòria aleatòria

Methods

def centre(self)

Retorna el centre de l'el·lipsoide

def equacio_reduida(self)

Retorna l'equacio reduïda de l'el·lipsoide

def semieixos(self)

Retorna els semieixos de l'el·lipsoide

def semieixos_quadrats(self)

Retorna els semieixos al quadrat de l'el·lipsoide

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class EquacioLineal (eq, amp=False, prime=0)

Classe per treballar amb equacions lineals.

Atributs

equacio: terme de l'esquerra en la equacio "eq = 0" unknowns: incògnites que apareixen a l'equació amp: True o False prime: nombre de primes que escriurem a l'equació

Constructor.

Paràmentres

eq: expressió lineal que ha de contenir tots els termes, aleshores l'equació serà "eq = 0". Només guardem la part "eq" amp: quan escrivim l'equació en latex ha d'aparèixer &= o només = prime: nombre de primes que s'han de posar a les incògnites

Per exemple: x, y, z, t = symbols('x y z t') eq = 2*x*-*3*y+4*z*-*3*t-4 e = EquacioLineal(eq)

Static methods

`def coeficients(a, b, amp=False, prime=0)`

Retorna una nova equació amb coeficients de les incògnites el vector "a" i terme independent b

Paràmetres

a: Vector amb els coeficients de les incògnites b: terme independents amp: si és True l'equació s'escriurà amb el &= per al LaTeX prime: nombre de primes que s'escriuran a les incògnites

Methods

`def set_coeficient_positiu(self, k)`

Si el coeficient de "k" és negatiu, canvia de signe tota l'equació, de manera que el coeficients de "k" passa a ser positiu

`def to_sistema_equacions(self)`

`class EquacioParametrica (eq, amp=True)`

Classe per treballar amb equacions paramètriques

Atributs

equacio: l'equació paramètrica b: terme independent de l'equaqció coefs: coeficients dels paràmetres unknown: incògina de l'equació paramètrica

Contructor.

Paràmetres

eq: equació paramètrica. Ha de ser del tipus -x + 2*t1* - *3*t2 + t3 - 4 amb el signe menys a la incògnita amp: True o False en funció si hem d'escriure &= o només = en la representació en LaTeX de l'equació

Static methods

`def coeficients(a, b, p=0, total=1, amp=True)`

Genera una equació amb coeficients dels paràmtres el vector "a", terme independent b i incògnita número p d'un total de "total".

Paràmetres

a: vector amb els coeficients dels paràmetres b: terme independent p: índex que representa la incògnita total: nombre total d'incònites

Exemple

e = EquacioParametrica(Vector(3,-2,1),5,1,4) genera l'equació - y + 3*t1* - *2*t2 + t3 + 5 e = EquacioParametrica(Vector(3,-2,1,3],-5,3,7) genera l'equació - x3 + 3*t1* - *2*t2 + t3 + 3*t4 + 7*t5 - 5

Observació

si hi ha un màxim de quatre incògnites, son (x,y,z,t) si n'hi ha més, són (x1,x2,x3,x4,...)

class EquacionsParametriques (a, b, amp=True)

Classe per treballar amb sistemes d'equacions paramètriques

Atributs

A: matriu dels coeficients dels paràmetres B: vector dels termes independents equacions: llista de EquacioParametrica nombre: nombre d'equacions

Methods

def eliminar_parametres(self, prime=0)

Retorna el SistemaEquacions que s'obté en eliminar els paràmetres dels sistema

class FormaQuadratica (matriu, base=None)

Classe per treballar amb formes quadràtiques

Atributs

matriu: matriu de la forma quadrètica en la base canònica dimensio: n vaps: valors propis base: base oronormal en la que diagonalitza

Static methods

def aleatoria(ordre=3, maxim=20, vapsnonuls=2)

Retorna una forma quadràtica aleatòria

Paràmetres

ordre: ordre de la matriu simètrica de la forma quadràtica maxim: nombre màxim dels coeficients de la matriu vapsnonuls: nombre mínim de valrs propis no nuls

Methods

def classificacio(self)

Retorna la classificació de la forma quadràtica

def expressio_euclidiana(self, prime=1)

Retorna l'expressió euclidiana reduïda de la forma de polinomi expressat en LaTeX

Paràmetres

prime: nombre de primes que s'escriuran a les variables

def latex(self, base=None, prime=0)

Retorna l'expressió en latex de la forma quadràtica com a polinomi de segon grau en la base "base"

Paràmetres

base: base de R^n. No cal que sigui ortormal. Si és None, serà la base canònica prime: nombre de primes que s'escriuran a les variables del polinomi segon grau

def polinomi_caracteristic(self)

Retorna el polinomi característic de la forma quadràtica

def rank(self)

Retorna el rang de la forma quadràtica

def signatura(self)

Retorna la signatura o índexs d'inèrcia de la forma quadràtica

class Hiperbola(a2, b2, centre, eix)

Classe per treballar amb hipèrboles

Ancestors

Conica

Static methods

def aleatoria()

Retorna una hipèrbola aleatòria

Methods

def centre(self)

Retorna el centre de la el·lipse

def equacio_continua_asimptota(self, v)

Retorna l'equació contínua de l'asímptota amb direccio v en format LaTeX

Paràmetres

v: vector director de l'asímptota

def equacio_reduida(self)

Retorna l'equacio reduïda de la hipèrbola en format LaTeX

def focus(self)

Retorna els focus de la hipèrbola

def semidistancia_focal(self)

Retorna la simidistància focal

def semieix_imaginari(self)

Retorna el semieix imaginari

def semieix_real(self)

Retorna el semieix real

def to_asy(self, x=10, y=10)

Retorna una expressió per fer servir amb el programa Asymtote

Paràmetres

x, y: nombres enters. El gràfic es representarà en una quadricula de límits (-x,x) i (-y,y)

def vectors_directors_asimptotes(self)

Retorna els vectors directors de les asímptotes expressats en la base canònica

def vertexs(self)

Retorna els vèrtexs de la hipèrbola

Inherited members

Conica: `ellipse`, `equacio`, `from_equacio`, `hiperbola`, `parabola`, `referencia_principal`, `tipus`, `vectors`

class HiperboloideDuesFulles(a2, b2, c2, centre, eix1, eix2)

Classe per treballar amb hiperboloides de dues fulles

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un hiperboloide de dues fulles de manera aleatòria aleatòria

Methods

def centre(self)

Retorna el centre de l'hiperboloide de dues fulles

def equacio_reduida(self)

Retorna l'equacio reduïda de l'hiperboloide de dues fulles

def semieixos(self)

Retorna els semieixos de l'hiperboloide de dues fulles

def semieixos_quadrats(self)

Retorna els semieixos al quadrat de l'hiperboloide de dues fulles

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class HiperboloideUnaFulla(a2, b2, c2, centre, eix1, eix2)

Classe per treballar amb hiperboloides d'una fulla

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un hiperboloide d'una fulla de manera aleatòria

Methods

def centre(self)

Retorna el centre de l'el·lipsoide

def equacio_reduida(self)

Retorna l'equacio reduïda de l'hiperboloide d'una fulla

def semieixos(self)

Retorna els semieixos de l'el·lipsoide

def semieixos_quadrats(self)

Retorna els semieixos al quadrat de l'el·lipsoide

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class Impresora (settings=None)

La funció latex() del sympy té la mania d'escriure les variables x, y, z i t en l'ordre t, x, y i z. L'única manera que, de moment, he trobat per resoldre aquest inconvenient és definir la classe Impresora i la funció mylatex(). Ho he trobat a StackOverflow.

Ancestors

sympy.printing.printer.Printer

Class variables

var printmethod

class Matriu (matrix=Matrix([[1, 0, 0], [0, 1, 0], [0, 0, 1]]))

Classe que ens permetrà representar matrius. El problema de la classe Matrix del sympy és que només es poden multiplicar per elements del tipus Matrix. Ens insteressa poder multiplicar Matrius per Vectors

Atributs

dimensio: nombre de files de la matriu columnes: nombre de columnes de la matriu matriu: matriu de la classe Matrix del sympy

Només s'utilitzen quan generem una matriu diagonalitzble vaps: llista de vectors propis de la matriu veps: llista de vectors propis de la matriu

Constructor.

Paràmetres

matrix: matriu del tipus Matrix de sympy per defecte és la matriu unitat d'ordre 3

Static methods

def aleatoria(f=3, c=3, maxim=5, nuls=True)

Genera una matriu aleatoria.

Paràmetres

f: nombre de files de la matriu c: nombre de columnes de la matriu maxim: tots els elements tindran valor absolut menor que "maxim" nuls: la matriu pot contenir coeficients nuls o no

def amb_rang(f=3, c=3, r=3, maxim=5, mzeros=-1)

Retorna una matriu aleatoria amb rang r.

Paràmetres

f: nombre de files de la matriu c: nombre de columnes de la matriu r: rang de la matriu maxim: tots els elements tindran valor absolut menor que "maxim" nuls: la matriu pot contenir coeficients nuls o no

def diagonal(vals)

Retorna una matriu diagonal amb valors "vals" a la diagonal

Paràmetres

vals: llista d'escalars o vector (class Vector o Punt)

def diagonalitzable(ordre=3, maxim=5, mzeros=-1, mvaps=3, vapsnuls=False, vapsrepetits=True)

Retorna una matriu quadrada aleatoria diagonalitzable.

Paràmetres

ordre: nombre de files i columnes de la matriu maxim: tots els elements tindran valor absolut menor o igual que "maxim" mzeros: si nzeros >= 0, nombre màxim de zeros que tindrà la matrius si nzeros < 0, el nombre de zeros no està limitat mvaps: tots els valors propis tindran valor absolut menor o igual que "mvaps" vapsnuls: si hi pot aparèixer el valor propi nul vapsrepetits: si hi pot aparèixer valors propis repetits

def from_vectors_columna(vecs)

Retorna una nova matriu a partir d'una llista de vectors. Les components dels vectors seran les columnes de la nova matriu

Paràmetres

v: llista de vectors o punts

def from_vectors_fila(vecs)

Retorna una nova matriu a partir d'una llista de vectors. Les components dels vectors seran les files de la nova matriu

Paràmetres

v: llista de vectors o punts

def gram(ordre=3, maxim=5, mzeros=-1)

Retorna una matriu quadrada aleatoria que serà d'un producte escalar, és a dir, una matriu de Gram

Paràmetres

ordre: nombre de files i columnes de la matriu maxim: tots els elements tindran valor absolut menor o igual que "maxim" mzeros: si nzeros >= 0, nombre màxim de zeros que tindrà la matrius si nzeros < 0, el nombre de zeros no està limitat

def identitat(ordre)

def invertible(ordre=3, maxim=5, mzeros=-1, unitaria=False)

Retorna una matriu quadrada aleatoria invertible.

Paràmetres

ordre: nombre de files i columnes de la matriu maxim: tots els elements tindran valor absolut menor que "maxim" mzeros: si nzeros >= 0, nombre màxim de zeros que tindrà la matrius si nzeros < 0, el nombre de zeros no està limitat unitaria: si volem que el determinant sigui 1 o -1

def matriu_columna(v)

Retorna una nova matriu columna a partir de les components del vector v

Paràmetres

v: vector o punt

def matriu_fila(v)

Retorna una nova matriu fila a partir de les components del vector v

Paràmetres

v: vector o punt

def transformacio_elemental(ordre, i, j, s, t)

Methods

def adjunta(self)

Retorna una nova matriu que és l'adjunta de l'actual

def clona(self)

def det(self)

Retorna el determiant de la matriu

def determinant(self)

Retorna el determiant de la matriu

def es_diagonal(self)

Retorna True si és una matriu diagonal

def es_simetrica(self)

Retorna True si és simètrica

def factor_comu(self)

Retorna quin factor comú podem treure de la matriu

def inserta_columna(self, pos, columna)

Retorna una nova matriu amb la columna "columna" insertada a la posició "pos"

Paràmetrers

columna: nova columna de la matriu pos: posició que ha d'ocupar la nova columna

def inserta_fila(self, pos, fila)

Retorna una nova matriu amb la fila "fila" insertada a la posició "pos"

Paràmetrers

fila: nova fila de la matriu pos: posició que ha d'ocupar la nova fila

def intercanvia_columnes(self, i, j)

Retorna una matriu amb les columnes i i j permutades

Paràmetres

i, j: índexs de les columnes

def inversa(self)

Retorna una nova matriu que és la inversa de l'actual

def latex(self, format=None, tipus='p')

def max_diagonal(self)

Retorna el màxim en valor absolut dels coeficients de la diagonal Si la matriu no és quadrada retorna None

def norma_maxim(self)

Retorna la norma del màxim de la matriu

def nucli(self)

Retorna una llista de vectors que formen una base del nucli de la matriu

def nzeros(self)

Retorna el nombre de zeros de la matriu

def polinomi_caracteristic(self)

Retorna el polinomi característic de la matriu

def rang(self)

Retorna el rang de la matriu

def rank(self)

Retorna el rang de la matriu

def reordena_aleatoriament_columnes(self)

Retorna una nova matriu amb les columnes reordenades aleatòriament

def reordena_aleatoriament_files(self)

Retorna una nova matriu amb les files reordenades aleatòriament

def set_vaps(self, vaps)

Assigna un llista de valors propis a la variable self.vaps

Paràmetres

vaps: llista de nombres

def set_veps(self, veps)

Assigna un llista de vectors propis simplificats a la variable self.veps

Paràmetres

vaps: llista de vectors

def simplificar(self)

Simplifica la matriu, és a dir, converteix les seves entrades en una llista d'enters amb mcd igual a 1. Només té sentit si totes les components del vector són nombres enters o racionals

def sistema_propi(self)

Retorna el sistema d'equacions en format latex corresponent al càlcul dels valors propis de la matriu

def submatriu(self, files, columnes)

Retorna la submatriu determinada per les files "files" i les columnes "columnes".

Paràmetrers

files: llista de files columnes: llista de columnes

def subs(self, l)

def transposada(self)

Retorna la transposada de la matriu

def vectors_columna(self, simplificar=False)

Retorna una llista amb els vectors columna de la matriu

Paràmetres

simplificar: si és True retornarà els vectors simplificats

def vectors_fila(self, simplificar=False)

Retorna una llista amb els vectors fila de la matriu

Paràmetres

simplificar: si és True retornarà els vectors simplificats

class Parabola (vertex, focus)

Classe per treballar amb paràboles

Ancestors

Conica

Static methods

def aleatoria()

Retorna una paràbola aleatòria

Methods

def equacio_reduida(self)

Retorna l'equacio reduïda de la paràbola en format LaTeX

def focus(self)

Retorna el focus de la paràbola

def parametre(self)

Retorna el paràmetre de la paràbola

def to_asy(self, x=10, y=10)

Retorna una expressió per fer servir amb el programa Asymtote

Paràmetres

x, y: nombres enters. El gràfic es representarà en una quadricula de límits (-x,x) i (-y,y)

def vertex(self)

Retorna el vèrtex de la paràbola

Inherited members

Conica: ellipse, equacio, from_equacio, hiperbola, parabola, referencia_principal, tipus, vectors

class ParaboloideElliptic (a2, b2, vertex, eix1, eix2)

Classe per treballar amb paraboloides el·líptics

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un paraboloide el·líptic de manera aleatòria

Methods

def equacio_reduida(self)

Retorna l'equacio reduïda del paraboloide el·líptic

def semieixos(self)

Retorna els semieixos del paraboloide el·líptic

def semieixos_quadrats(self)

Retorna els semieixos al quadrat del paraboloide el·líptic

def vertex(self)

Retorna el vèrtex del paraboloide el·líptic

Inherited members

Quadrica: cilindreelliptic, cilindrehiperbolic, cilindreparabolic, con, ellipsoide, equacio, from_equacio, hiperboloideduesfulles, hiperboloideunafulla, paraboloideelliptic, paraboloidehiperbolic, referencia_principal, tipus, vectors

class ParaboloideHiperbolic (a2, b2, vertex, eix1, eix2)

Classe per treballar amb paraboloides hiperbòlics

Ancestors

Quadrica

Static methods

def aleatoria()

Retorna un paraboloide hiperbòlic de manera aleatòria

Methods

def equacio_reduida(self)

Retorna l'equacio reduïda del paraboloide hiperbòlic

def semieixos(self)

Retorna els semieixos del paraboloide hiperbòlic

def semieixos_quadrats(self)

Retorna els semieixos al quadrat del paraboloide hiperbòlic

def vertex(self)

Retorna el vèrtex del paraboloide hiperbòlic

Inherited members

Quadrica: `cilindreelliptic`, `cilindrehiperbolic`, `cilindreparabolic`, `con`, `ellipsoide`, `equacio`, `from_equacio`, `hiperboloideduesfulles`, `hiperboloideunafulla`, `paraboloideelliptic`, `paraboloidehiperbolic`, `referencia_principal`, `tipus`, `vectors`

class PlaAfi(p, u1, u2, ref=None)

Classe per treballar amb plans afins.

Atributs

u1, u2: vectors directors del pla p: punt de pas

Static methods

def amb_associat(w, p)

Genera el pla afí que té vector perpendicular "w" i passa pel punt p

Paràmetres

w: vector associat p: punt de pas

def from_equacio_implicita(eq)

Retorna el pla afí que té equació implícita eq

Paràmetres

eq: EquacioLineal

Methods

def associat(self, base=None)

Retorna un vector perpendicular al pla expressat en la base "base"

Paràmetres

base: base de l'espai vectorial. Si és None, serà la canònica

def base_ortogonal(self)

Retorna una base orogonal (vectors directors perpendiculars) del pla

def distancia(self, other)

Retorna la distància entre el pla actual i un punt, una recta o un altre pla

Paràmetres

other: un punt (classe Punt), una recta (classe RectaAfi) o un pla (class PlaAfi)

`def equacio_implicita(self, ref=None, prime=0)`

Retorna l'expressió de l'equació implícita del pla en la referència "ref"

Paràmetres

ref: referència afí prime: nombre de primes que s'escriuran a les incògnites

`def projeccio_ortogonal(self, punt)`

Retorna la projecció ortogonal del punt "punt" sobre el pla

Paràmetres

punt: Punt

`def punt_de_coordenades_enteres(self, p=None, u=None, v=None)`

Retorna, si és possible, un punt de coordenades enteres del pla afí que passa pel punt p i té vectors directors u i v

`def simetric(self, punt)`

Retorna el simètric del punt "punt" respecte al pla

Paràmetres

punt: Punt

`class PlaVectorial (u1, u2)`

Classe per treballar amb plans vectorials

Static methods

`def amb_associat(w)`

Genera el pla vectorial que té vector perpendicular "w"

Paràmetres

v: vector no nul de dimensió 3

`def from_matriu(m)`

Crea el pla vectorial generat per les columnes de la matriu "m"

Paràmetres

m: matriu. Ha de tenir 3 files, dues columnes i rang 2

Methods

`def base_ortogonal(self)`

Retorna una base ortogonal del pla vectorial

`def conte(self, u)`

Retorna si el vector u pertany al pla.

Paràmetres

u: Vector

def equacio_implicita(self, base=None, prime=0)

Retorna l'equació implícita del pla en la base "base" en format LaTeX. Normalment, si la base no és la canònica es posa prime > 0 perquè el resultat sigui de l'estil 2x'-3y'+4z'= 0

Paràmetres

base: una base (classe Base). Si és None, serà la canònica prime: nombre de primes que s'escriuran a les incògnites

def es_associat(self, u)

Retorna si el vector és perpendicular al pla.

Paràmetres

u: Vector

def projeccio_ortogonal(self, u)

Retorna la projecció ortogonal del vector u sobre el pla.

Paràmetres

u: Vector

def simetric(self, u)

Retorna el simètric del vector u respecte al pla.

Paràmetres

u: Vector

class Punt (*args)

Classe per treballar amb punts. Internament un punt és el mateix que un vector

Constructor.

Paràmetres

c: Una única llista de nombres o una llista de paràmetres que han de ser nombres

Exemples

u = Vector([2,3,1,2]) v = Vector(3,1,-2)

Ancestors

Vector

Methods

def coordenades_en_referencia(self, ref)

Retorna les coordenades del punt en la referència "ref"

Paràmetres

ref: referència de la classe ReferenciaAfi

Inherited members

Vector: aleatori, components_en_base, cross, dot, factor_comu, latex, length, maxim,

`normalitzar`, `normalitzat`, `nul`, `nzeros`, `radsimplificar`, `reordena_aleatoriament`, `simplificar`, `tots_enters`

class Quadrica (matriu, ref=None)

Classe per treballar amb quàdriques. L'objectiu no és classificar quàdriques, sinó generar-les a partir dels elements característics o de manera aleatòria.

Atributs

ref: Referència afí matriu: matriu projectiva de la quàdrica en la referència "ref" canonica: matriu projectiva de la quàdrica en la referència canònica

Subclasses

CilindreElliptic, CilindreHiperbolic, CilindreParabolic, Con, Ellipsoide, HiperboloideDuesFulles, HiperboloideUnaFulla, ParaboloideElliptic, ParaboloideHiperbolic

Static methods

def aleatoria(maxim=30, diagonal=15)

Retorna una quàdrica de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

def cilindreelliptic(maxim=30, diagonal=15)

Retorna un cilindre el·líptic de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

def cilindrehiperbolic(maxim=30, diagonal=15)

Retorna un cilindre hiperbòlic de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

def cilindreparabolic(maxim=30, diagonal=15)

Retorna un cilindre parabòlic de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

def con(maxim=30, diagonal=15)

Retorna un con de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

def ellipsoide(maxim=30, diagonal=15)

Retorna un el·lipsoide de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

`def from_equacio(eq)`

Retorna i classifica la quadrica a partir de la seva equació. Només per als nou tipus de quàdriques definides en aquesta llibreria

`def hiperboloideduesfulles(maxim=30, diagonal=15)`

Retorna un hiperboloide de dues fulles de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

`def hiperboloideunafulla(maxim=30, diagonal=15)`

Retorna un hiperboloide d'una fulla de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

`def paraboloideelliptic(maxim=30, diagonal=15)`

Retorna un paraboloide el·líptic de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

`def paraboloidehiperbolic(maxim=30, diagonal=15)`

Retorna un paraboloide hiperbòlic de manera aleatòria

Paràmetres

maxim: valor màxim de la matriu projectiva de la quàdrica diagonal: valor màxim de la diagonal de la matriu projectiva de la quàdrica

Methods

`def equacio(self)`

Retorna l'equació en latex de l'equació de la quàdrica

`def referencia_principal(self)`

Retorna la referencia principal

`def tipus(self)`

Retorna el tipus de quàdrica

`def vectors(self, unitaris=False)`

Retorna els vectors de la base de la referència principal

Paràmetres

unitaris: si és True, els retorna unitaris

class Radicals

Classe per treure factor comú en expressions on hi apareixen arrels quadrades

Methods

def busca_fraccions(self, el)

Afegeix els termes que apareixen als denominadors a la llista self.fraccions

Paràmetres

el: expressió del sympy

def busca_quadrats(self, el)

Afegeix els termes que apareixen dins d'arrels quadrades a la llista self.quadrats

Paràmetres

el: expressió del sympy

def mcd(self)

Retorna el màxim comú divisor dels elements de la llista self.quadrats

def mcm(self)

Retorna el mínim comú múltiple dels elements de la llista self.faccions

class RectaAfi(p, u, ref=None)

Classe per treballar amb rectes afins, dimensió 2 o 3

Atributs

u: generador de la recta vectorial p: punt de pas

Static methods

def from_equacions_implicites(s)

Retorna la recta afí que té equacions implícites s

Paràmetres

s: SistemaEquacions

Methods

def conte(self, punt)

Retorna si el punt "punt" pertany a la recta

Paràmetres

punt: Punt

def distancia(self, other)

Retorna la distància entre la recta actual i un punt, una recta o un pla

Paràmetres

other: un punt (classe Punt), una recta (classe RectaAfi) o un plan (class PlaAfi)

`def equacio_continua(self, ref=None, prime=0)`

Retorna l'expressió en latex de l'equqció contínua de la recta afí en la referència "ref".

Paràmetres

ref: referència afí. Si és None, serà la canònica prime: nombre de primes que s'escriuran a les incògnites

`def equacions_implicites(self, ref=None, prime=0, aleatori=False)`

Retorna l'equació implícita (dimensió 2) o el sistema d'equacions implícites (dimensió 3) de la recta afí en la referència "ref".

Paràmetres

ref: referència en la que calculem les equacions implícites prime: nombre de primes que s'escriran a les incògnites aleatori: només s'aplica a dimensió 3 i genera unes equacions implícites amb tots els coeficients de les incògnites no nuls

`def projeccio_ortogonal(self, punt)`

Retorna la projecció ortogonal del punt "punt" sobre la recta

Paràmetres

punt: Punt

`def punt(self, t)`

Retorna el punt de la recta amb paràmetre t

Paràmetres

t: escalar

`def punt_de_coordenades_enteres(self, p=None, u=None)`

Retorna, si és possible, un punt de coordenades enteres de la recta que passa pel punt p i té vector director u

`def simetric(self, punt)`

Retorna el simètric del punt "punt" respecte a la recta

Paràmetres

punt: Punt

`class RectaRegressio (punts)`

Classe per treballar amb rectes de regressió

Atributs

punts: llista de punts A: matriu dels coeficients de les incògnites B: vector de termes independents solucio: solució del sistema d'equacions A^tAX = A^tB

Static methods

`def aleatoria(l=4, max=4)`

Retorna un problema aleatori amb l punts

Paràmetres

l: nombre de punts max: valor màxim de les ys

Methods

`def equacio(self)`

Retorna l'equació de la recta de regressió expressada en LaTeX

`def error_quadratic(self)`

Retorna d'error quadràtic

`def llista_punts(self)`

Retorna la llista de punts en format LaTeX

`def taula_punts(self)`

Retorna una taula en format LaTeX dels punts

`class RectaVectorial (u)`

Classe per treballar amb rectes vectorials, dimensió 2 o 3

Atributs

u: generador de la recta vectorial

Methods

`def equacio_continua(self, base=None, prime=0)`

Retorna l'expressió en LaTeX de l'equació contínua de la recta vectorial en la base "base".

Paràmetres

base: base del pla o de l'espai vectorial (classe Base) prime: nombre de primes amb el que s'escriuran les incògnites

`def equacions_implicites(self, base=None, prime=0, aleatori=False)`

Retorna l'equació implícita (dimensió 2) o el sistema d'equacions implícites (dimensió 3) en la base "base".

Paràmetres

base: Base en la que calculem les equacions implícites prime: Quantes primes volem posar a les equacions aleatori: només s'aplica a dimensió 3 i genera unes equacions implícites no trivials

`def projeccio_ortogonal(self, u)`

Retorna la projecció ortogonal del vector u sobre la recta.

Paràmetres

u: Vector

`def simetric(self, u)`

Retorna el simètric del vector u respecte a la recta.

Paràmetres

u: Vector

`class ReferenciaAfi (origen, base)`

Classe per treballar amb referències afins de P^n

Atributs

origen: origen de la referència (classe Punt) base: base de la referència (classe Base) dimensio: dimensió de l'espai corresponent

Static methods

`def aleatoria(dimensio=3, maxim=3, mzeros=0, unitaria=False)`

Retorna una referència aleatòria

Paràmetres

dimensio: dimensió de l'espai corresponent maxim: Màxim nombre que hi apareix mzeros: Màxim nombre de zeros que apareixen a la base unitaria: si és True la matriu del canvi de base tindrà determinant 1 o -1

`def canonica(dimensio=3)`

Retorna la referència canònica

Paràmetres

dimensio: dimensió de l'espai corresponent

Methods

`def canvi_coordenades(self, prime1=0, prime2=1)`

Restorna en format latex l'expressió del canvi de coordenades de la referència actual a la referència canònica

Paràmetres

prime1: primes que s'escriuran a les coordenades en la referència canònica prime2: primes que s'escriuran a les coordenades en la referència actual

`def canvi_de_referencia_a_la_referencia(self, R, p1, p2)`

Retorna en format latex l'expressió del canvi de coordenades de la referència actual a la referència R

Paràmetres

p1: primes que s'escriuran a les coordenades en la referència actual p2: primes que s'escriuran a les coordenades en la referència R

`def coordenades_del_punt(self, punt, ref=None)`

Retorna un nou punt expressat en aquesta referència del punt que en la referència "ref" té coordenades "punt". Si ref és None, serà la referència canònica

Paràmetres

punt: coordenades en la referència "ref" ref: ReferenciaAfi

`def punt_de_coordenades(self, punt)`

Retorna un nou punt expressat en la referència canònica del punt que en aquesta referencia té coordenades "punt"

Paràmetres

punt: coordenades d'un punt en la referència actual

`def referencia_inversa(self)`

Retorna una referència que es correspon amb el canvi de coordenades de de la referència canònica a l'actual

`def vectors(self, unitaris=False)`

Retorna els vectors de la base de la referència

Paràmetres

unitaris: si és True els retorna dividits per la seva longitud

class SistemaEquacions (a, b, unknowns=None, prime=0)

Classe per treballar amb sistemes d'equacions lineals

Atributs

A: matriu dels coeficients de les incógnites B: vector de termes independents equacions: llista de EquacioLineal nombre: nombre d'equacions solucio: solucio del sistema d'equacions parametrica: solucio paramètrica del sistema d'equacions parametres: paràmetres que apareixen a la solució paramètrica unknowns: llista d'incògnites prime: nombre de primes que escriurem a l'equació

Static methods

def from_equacions(eqs, nombre, prime=0)

Retorna un sistema d'equacions amb equacions "eqs"

Paràmetres

eqs: llista de EquacioLineal nombre: nombre d'incògnites prime: nombre de primes que s'escriuran a les incògnites

Methods

def matriu_ampliada(self)

Retorna la matriu ampliada del sistema d'equacions expressada en LaTeX

def matriu_incognites(self)

Retorna la matriu dels coeficients de les incògnites expressada en LaTeX

def resol(self, unknowns=None)

Resol el sistema d'equacions utilitzant la funció linsolve del sympy. El resultat és una llista d'expressions on hi poden aparèixer les incògnites del sistema com a paràmetres

def solucio_latex(self, linia=False, unknowns=None)

Retorna l'expressió en LaTex de la solució del sistema d'equacions

Paràmetres

linia: si és True escriu la solució en una línia, en cas contrari ho fa com un sistema d'equacions

class SubespaiVectorial (vecs, basern=None)

Classe per treballar amb subespais vectorials

Atributs

generadors: generadors del subespai base: base del subespai dimensio: dimensio del subespai espai: n si és un suespai de R^n

Static methods

def from_equacio_implicita(eq, basern=None)

def from_equacions_implicites(eqs, basern=None)

Retorna el subespai vectorial que té equacions implícites "eqs"

Paràmetres

eqs: equacions implícites (classe SistemaEquacions) basern: Base de R^n

Methods

`def amplia_base(self, unitaria=False)`

Retorna una base ortogonal amb orientació positiva de R^n que comença amb una base del subespai

Paràmetres

unitaria: si és True, retorna una base ortonormal

`def amplia_base_suplementari(self, unitaria=False)`

Retorna una base ortogonal amb orientació positiva de R^n que comença amb una base del suplementari ortogonal del subespai

Paràmetres

unitaria: si és True, retorna una base ortonormal

`def base_ortogonal(self)`

Retorna una base ortogonal del subespai

`def equacions_implicites(self, basern=None, prime=0)`

Retorna unes equacions implícites del subespai

Paràmetres

basern: base en la que s'escriuran les equacions implícites prime: nombre de primes a les incògnites

`def es_total(self)`

Retorna True si és el subespai R^n

`def es_zero(self)`

Retorna True si és el subespai {0}

`def projeccio_ortogonal(self, u)`

Retorna la projeccio ortogonal del vector u sobre el subespai

Parametres

u: Vector

`def simetric(self, u)`

Retorna el simètric del vector u sobre respecte al subespai

Parametres

u: Vector

`def suplementari_ortogonal(self)`

Retorna el suplementari ortogonal

`class TransformacioAfi (p, t)`

Classe per treballar amb transformacions afins T:P^n ----> P^n, on T(p) = T + A(p)

Atributs

transformacio: Transformació lineal donada per la matriu A translacio: translació de la transformació afí en la referència canònica dimensio: n

Static methods

`def gir(angle, origen, radiants=False)`

Retorna el gir d'angle "angle" al voltant del punt "origen"

Paràmetres

origen: centre del gir (classe Punt) angle: angle de rotació radiants: si és True, l'angle ha d'estar expressat en radiants

`def moviment_helicoidal(recta, angle, radiants=False, alpha=0)`

Retorna el moviment helicoidal de P_3 que consisteix en la rotació d'angle "angle" al voltant de la recta "recta" seguit d'una translació de vector alpha * vector director de la recta.

Paràmetres

recta: eix de rotació angle: angle de rotació radiants: si és True, l'angle ha d'estar expressat en radiants alpha: factor de la translació

`def projeccio_ortogonal(v)`

Retorna la projecció ortogonal sobre la variatat lineal "v"

Paràmetres

v: varietat lineal (classe VarietatLineal)

`def rotacio(recta, angle, radiants=False)`

Retorna la rotació d'angle "angle" al voltant de la recta "recta"

Paràmetres

recta: eix de rotació angle: angle de rotació radiants: si és True, l'angle ha d'estar expressat en radiants

`def simetria(v)`

Retorna la simetria respecte a la variatat lineal "v"

Paràmetres

v: varietat lineal (classe VarietatLineal)

Methods

`def latex(self, ref=None, prime=0)`

Retorna l'expressió en latex de la transformació afí en la referència "ref". Si ref és None, serà en la referència canònica

Paràmetres

base: referència de P^n prime: nombre de primes que s'escriuran

`def transforma(self, p, ref=None)`

Calcula el transformat o imatge del punt "p". p seran les coordenades d'aquest punt en la referència "ref" i el transformat també estarà expressat en aquesta referència

Paràmetres

punt: punt (classe Punt) ref: referència afí. Si és None, serà la referència canònica

`class TransformacioLineal (matriu, base=None)`

Classe per treballar amb transformacions lineals T:R^n ----> R^n

Atributs

dimensio: n canonica: matriu de la transformació en la base canònica

Static methods

`def gir(angle, radiants=False)`

Retorna el gir d'angle "angle" en dimensió 2

Paràmetres

angle: angle de rotació radiants: si l'angle està en radiants, ha de ser True

`def projeccio_ortogonal(s)`

Retorna la projecció ortogonal sobre el subespai "s"

Paràmetres

s: subespai vectorial (classe SubespaiVectorial)

`def rotacio(eix, angle, radiants=False)`

Retorna la rotació d'angle "angle" al voltant del vector "eix"

Paràmetres

eix: vector al voltant del qual fem la rotació angle: angle de rotació radiants: si l'angle està en radiants, ha de ser True

`def simetria(s)`

Retorna la simetria respecte al subespai "s"

Paràmetres

s: subespai vectorial (classe SubespaiVectorial)

Methods

`def angles_euler(self, radiants=False)`

Retorna els angles d'Euler de la rotació

Paràmetres

radiants: si és True retorna els angles en radiants, en cas contrari, ho fa en graus

`def determinant(self)`

Retorna el deterinant de la transformació lineal

`def eix_angle_rotacio(self, radiants=False)`

Retorna l'eix i l'angle de rotació

Paràmetres

radiants: si és True retorna l'angle en radiants, en cas contrari, ho fa en graus

`def es_rotacio(self)`

Ens diu si és una rotació tridimensional o no

`def es_simetrica(self)`

Retorna si la transformació lineal és simètrica

def latex(self, base=None, prime=0)

Retorna l'expressió en latex de la transformació lineal en la base "base" Si base és None, serà en la base canònica

Paràmetres

base: base de R^n prime: nombre de primes que s'han d'escriure

def matriu_en_base(self, base)

Retorna la matriu de la transformacio lineal en la base "base"

Paràmetres

base: base de R^n (classe Base)

def polinomi_caracteristic(self)

Retorna el polinomi característic de la transformació lineal

def transforma(self, vec, base=None)

Calcula el transformat (image) del vector "vec".

Paràmetres

vec: vector base: si no és None, vec seran les compoents del vectors en aquesta base. El transformat o imatge també estarà expressat en aquesta base

class VarietatAfi(p, s, ref=None)

Classe per treballar amb varietats lineal

Atributs

punt: punt de pas subespai: SubespaiVectorial

Static methods

def from_equacio_implicita(eq, ref=None)

def from_equacions_implicites(eqs, ref=None)

Retorna la varietat afí que té equacions implícites "eqs" en la referència afí "ref"

Paràmetres

eqs: equacions implícites (classe SistemaEquacions) ref: Referència de P^n

Methods

def base_ortogonal(self)

Retorna una base ortogonal del subespai director de la varietat lineal

def equacions_implicites(self, ref=None, prime=0)

Retorna unes equacions implícites de la varietat lineal

Paràmetres

ref: referència en la que s'escriuran les equacions implícites prime: nombre de primes de les incògnites

def es_total(self)

Retorna True si la varietat lineal és P_n

def es_un_punt(self)

Retorna True si la varietat lineal és un punt

def projeccio_ortogonal(self, punt)

Retorna la projecció ortogonal del punt "punt" sobre la varietat afi

Paràmetres

punt: Punt

def simetric(self, punt)

Retorna el simètric del punt "punt" respecte a la varietat afí

Paràmetres

punt: Punt

def varietat_ortogonal(self, p)

Retorna la varietat ortogonal a l'actual que passa pel punt p

class Vector (*args)

Classe que ens permetrà representar vectors i punts

Atributs

dimensio: el nombre de components o longitud del vector components: llista amb les components del vector

Constructor.

Paràmetres

c: Una única llista de nombres o una llista de paràmetres que han de ser nombres

Exemples

u = Vector([2,3,1,2]) v = Vector(3,1,-2)

Subclasses

Punt

Static methods

def aleatori(l=3, maxim=5, nuls=True, positius=False)

Retorna un vector aleatori de longitud l

Paràmetres

l: longitud del vector maxim: Nombre màxin que pot contenir en valor absolut nuls: Si pot contenir el valor 0 o no positius: Si els coeficients han de set tots positius

def nul(dim)

Retorna el vector nul de longitud dim

Methods

def components_en_base(self, base=None)

Retorna un nou vector amb les components del vector (donem per fet que estan en la base canònica) en la base "base"

Paràmetres

base: una base (classe Base)

`def cross(self, other, simplificar=False)`

Retorna un nou vector que és el producte vectorial de dos vectors: Si simplificar és True, simplifica el vector resultant

Paràmetres

other: un altre vector simplificar: si es vol simplificar el resultat o no

Exemple

u = Vector.aleatori(l=3) v = Vector.aleatori(l=3) w = u.cross(v,simplificar=True)

`def dot(self, other)`

Retorna el producte escalar de dos vectors.

Paràmetres

other: un altre vector

Exemple

u = Vector.aleatori(l=3) v = Vector.aleatori(l=3) p = u.dot(v)

`def factor_comu(self)`

En cas que totes les components del vector sigui enteres o racionals, torna el racional que es pot treure factor comú i el vector simplificat. Si hi ha components que no són ni enteres ni racionals, torna 1 i el mateix vector

`def latex(self, unitari=False)`

Retorna l'expressió en latex del vector.

Paràmetres

unitari: si unitari és True, el retorna dividit per la seva longitud

`def length(self)`

Retorna la longitud o mòdul del vector

`def maxim(self)`

Retorna el màxim dels valors absoluts de les seves components

Exemple

u = Vector.aleatori(l=5,maxim=4,nuls=False) m = u.maxim()

`def normalitzar(self)`

Converteix el vector en unitari

Exemple

u = Vector(1,2,3) u.normalitzar()

`def normalitzat(self)`

Retorna el vector unitari en la direcció i sentit del vector

Exemple

u = Vector(1,2,3) v = u.normalitzat()

def nzeros(self)

Retorna el nombre de zeros del vector

def punt(self)

def radsimplificar(self)

Simplifica el vector quan alguna de les seves components té radicals

Exemple

u = Vector(sqrt(2),-3*sqrt(2),-sqrt(2)) u.radsimplificar()

def reordena_aleatoriament(self)

Retorna un nou vector amb les components reordenades aleatoriament

def simplificar(self)

Simplifica el vector, és a dir, converteix les seves components en una llista d'enters amb mcd igual a 1. Només té sentit si totes les components del vector són nombres enters o racionals

Exemple

u = Vector(-1,2,2,Rational(3,2)) u.simplificar()

def tots_enters(self)

Retorna True si totes les components del vector són nombres enters

Index

Functions

Classes

parametre
vertex

Con
aleatoria
centre
equacio_reduida
semieixos
semieixos_quadrats

Conica
aleatoria
ellipse
equacio
from_equacio
hiperbola
parabola
referencia_principal
tipus
vectors

Ellipse
aleatoria
centre
equacio_reduida
focus
semidistancia_focal
semieix_major
semieix_menor
to_asy
vertexs

Ellipsoide
aleatoria
centre
equacio_reduida
semieixos
semieixos_quadrats

EquacioLineal
coeficients
set_coeficient_positiu
to_sistema_equacions

EquacioParametrica
coeficients

EquacionsParametriques
eliminar_parametres

FormaQuadratica
aleatoria
classificacio
expressio_euclidiana
latex
polinomi_caracteristic
rank
signatura

Hiperbola

aleatoria
centre
equacio_continua_asimptota
equacio_reduida
focus
semidistancia_focal
semieix_imaginari
semieix_real
to_asy
vectors_directors_asimptotes
vertexs

HiperboloideDuesFulles
aleatoria
centre
equacio_reduida
semieixos
semieixos_quadrats

HiperboloideUnaFulla
aleatoria
centre
equacio_reduida
semieixos
semieixos_quadrats

Impresora
printmethod

Matriu
adjunta
aleatoria
amb_rang
clona
det
determinant
diagonal
diagonalitzable
es_diagonal
es_simetrica
factor_comu
from_vectors_columna
from_vectors_fila
gram
identitat
inserta_columna
inserta_fila
intercanvia_columnes
inversa
invertible
latex
matriu_columna
matriu_fila
max_diagonal
norma_maxim
nucli
nzeros

polinomi_caracteristic
rang
rank
reordena_aleatoriament_columnes
reordena_aleatoriament_files
set_vaps
set_veps
simplificar
sistema_propi
submatriu
subs
transformacio_elemental
transposada
vectors_columna
vectors_fila

Parabola
aleatoria
equacio_reduida
focus
parametre
to_asy
vertex

ParaboloideElliptic
aleatoria
equacio_reduida
semieixos
semieixos_quadrats
vertex

ParaboloideHiperbolic
aleatoria
equacio_reduida
semieixos
semieixos_quadrats
vertex

PlaAfi
amb_associat
associat
base_ortogonal
distancia
equacio_implicita
from_equacio_implicita
projeccio_ortogonal
punt_de_coordenades_enteres
simetric

PlaVectorial
amb_associat
base_ortogonal
conte
equacio_implicita
es_associat
from_matriu
projeccio_ortogonal

simetric

Punt

coordenades_en_referencia

Quadrica

aleatoria
cilindreelliptic
cilindrehiperbolic
cilindreparabolic
con
ellipsoide
equacio
from_equacio
hiperboloideduesfulles
hiperboloideunafulla
paraboloideelliptic
paraboloidehiperbolic
referencia_principal
tipus
vectors

Radicals

busca_fraccions
busca_quadrats
mcd
mcm

RectaAfi

conte
distancia
equacio_continua
equacions_implicites
from_equacions_implicites
projeccio_ortogonal
punt
punt_de_coordenades_enteres
simetric

RectaRegressio

aleatoria
equacio
error_quadratic
llista_punts
taula_punts

RectaVectorial

equacio_continua
equacions_implicites
projeccio_ortogonal
simetric

ReferenciaAfi

aleatoria
canonica
canvi_coordenades
canvi_de_referencia_a_la_referencia
coordenades_del_punt
punt_de_coordenades

referencia_inversa
vectors

SistemaEquacions
from_equacions
matriu_ampliada
matriu_incognites
resol
solucio_latex

SubespaiVectorial
amplia_base
amplia_base_suplementari
base_ortogonal
equacions_implicites
es_total
es_zero
from_equacio_implicita
from_equacions_implicites
projeccio_ortogonal
simetric
suplementari_ortogonal

TransformacioAfi
gir
latex
moviment_helicoidal
projeccio_ortogonal
rotacio
simetria
transforma

TransformacioLineal
angles_euler
determinant
eix_angle_rotacio
es_rotacio
es_simetrica
gir
latex
matriu_en_base
polinomi_caracteristic
projeccio_ortogonal
rotacio
simetria
transforma

VarietatAfi
base_ortogonal
equacions_implicites
es_total
es_un_punt
from_equacio_implicita
from_equacions_implicites
projeccio_ortogonal
simetric
varietat_ortogonal

Vector

aleatori

components_en_base

cross

dot

factor_comu

latex

length

maxim

normalitzar

normalitzat

nul

nzeros

punt

radsimplificar

reordena_aleatoriament

simplificar

tots_enters