TP8

November 29, 2022

0.0.1 Exo 1 : calculs de résultants

[1]:
```
def Sylvester(P,Q):
    P=P.polynomial(Y)  # pour voir P et Q comme des polynomes en Y à coeff dans␣
 ↪Q[X]
    Q=Q.polynomial(Y)
    m=P.degree()
    n=Q.degree()
    LP=P.list()
    LP.reverse()
    LQ=Q.list()
    LQ.reverse()
    S=[]
    for i in range(n):
        L=[0]*i+LP+[0]*(n-1-i)
        S.append(L)
    for i in range(m):
        L=[0]*i+LQ+[0]*(m-1-i)
        S.append(L)
    return matrix(S)

K.<X,Y>=QQ[]
P=X**2+Y-1
Q=Y*X+Y**3-5
S=Sylvester(P,Q)
print(S)
Res1=S.determinant()
Res2=P.resultant(Q,Y)
print(Res1)
print(Res2)
```

```
[      1 X^2 - 1       0       0]
[      0       1 X^2 - 1       0]
[      0       0       1 X^2 - 1]
[      1       0       X      -5]
-X^6 + 3*X^4 - X^3 - 3*X^2 + X - 4
-X^6 + 3*X^4 - X^3 - 3*X^2 + X - 4
```

[2]:
```
def ResPGCD(P,Q):
    P=P.polynomial(Y)  # pour voir P et Q comme des polynomes en Y à coeff dans␣
 ↪Q[X]
    Q=Q.polynomial(Y)
    F=P.base_ring().fraction_field()  # declare Q(X)
    P=P.change_ring(F)  # met P, Q dans Q(X)[Y] pour avoir une division␣
 ↪euclidienne
    Q=Q.change_ring(F)
    c=1
    while True:
        m=P.degree()
        n=Q.degree()
        b=Q.leading_coefficient()
        R=P%Q
        if R!=0:
            c=c*(-1)**(m*n)*b**(m-R.degree())
            P,Q=Q,R
        elif R==0 and n==0: #dernier reste non nul est constant
            return Q*c
        else: # dernier reste non nul non constant, donc P,Q non premiers entre␣
 ↪eux, donc resultant=0
            return 0
```

[3]:
```
ResPGCD(P,Q)
```

[3]:
```
-X^6 + 3*X^4 - X^3 - 3*X^2 + X - 4
```

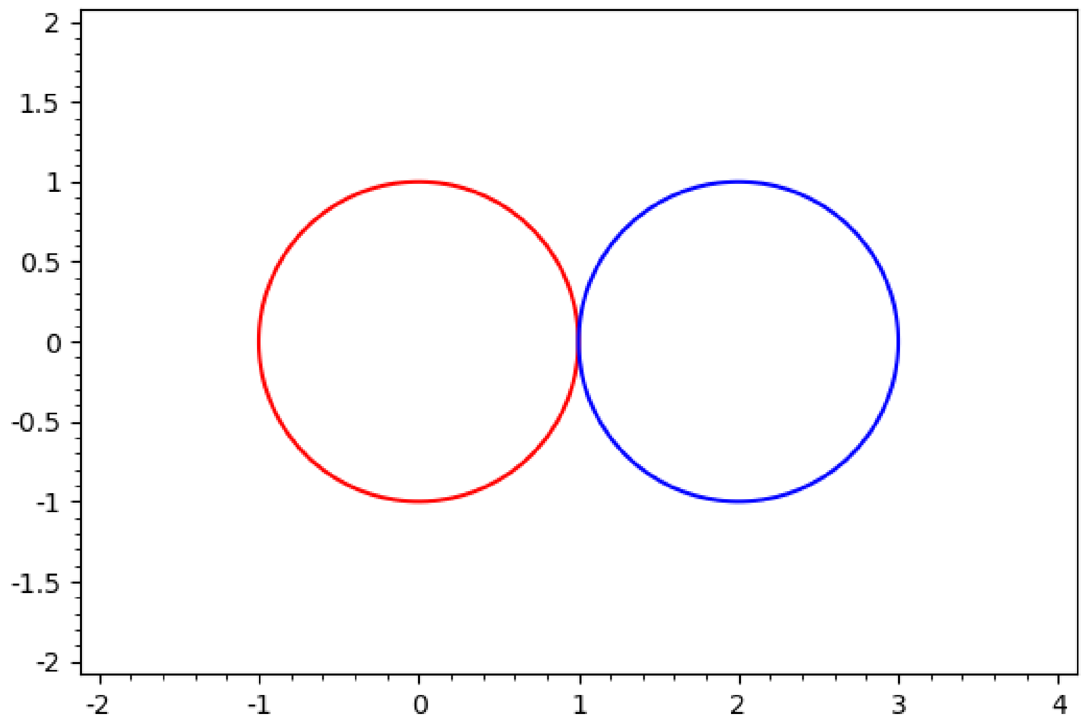
[4]:
```
N=10
P=K.random_element(degree=N)
Q=K.random_element(degree=N)
%time R1=ResPGCD(P,Q)
%time R2=Sylvester(P,Q).determinant()
%time R3=P.resultant(Q,Y)
print(R1==R3)
print(R2==R3)
```

```
CPU times: user 1.03 ms, sys: 122 µs, total: 1.15 ms
Wall time: 1.15 ms
CPU times: user 1.03 ms, sys: 123 µs, total: 1.15 ms
Wall time: 1.16 ms
CPU times: user 383 µs, sys: 0 ns, total: 383 µs
Wall time: 385 µs
True
True
```

0.0.2 Exo 2 : Résolution de systèmes à deux variables

[5]:
```
Rx.<x>=QQ['x']
Ry.<y>=QQ['y']
R.<x,y>=QQ['x,y']

A=x**2 + y**2-1
B=(x-2)**2 +y**2 - 1

GA=implicit_plot(A,(-2,4),(-2,2),color='red')
GB=implicit_plot(B,(-2,4),(-2,2),color='blue')

GA+GB
```

[5]:



[6]:
```
Res=A.resultant(B,y)
print(Res)
print(Res.parent())
Res=Rx(Res) # pour voir Res comme en un polynôme univarié en x
Res.roots()
```

```
16*x^2 - 32*x + 16
Multivariate Polynomial Ring in x, y over Rational Field
```

[6]: [(1, 2)]

[7]:
```
D=gcd(A(1,y),B(1,y))
print(D)
D=Ry(D)  # pour voir Res comme en un polynôme univarié en x
D.roots()   # a nouveau
```

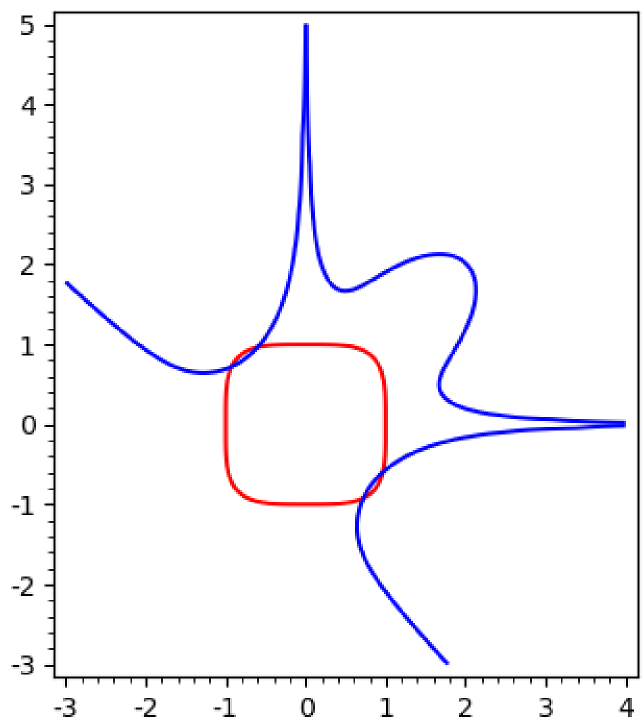
```
y^2
```

[7]:
```
[(0, 2)]
```

Le résultant a une unique racine (double) $x = 1$, et les polynômes $A(1, y)$ et $B(1, y)$ ont pour unique racine commune (double) $y = 1$. L'unique point d'intersection est donc $(1, 0)$. Puisque l'on a épuisé toutes les racines du résultant (y compris complexes), il n'y a pas d'autres points (y compris complexes). Notons tout de même qu'il y a en fait un point double caché ``à l'infini'', mais qui ne se voit pas ici sur des ``asymptotes''.

[8]:
```
F=x**4 + y**4-1
G=x**5*y**2 - 4*x**3*y**3+x**2*y**5-1

G1=implicit_plot(F,(-3,4),(-3,5),color='red')
G2=implicit_plot(G,(-3,4),(-3,5),color='blue')

G1+G2
```

[8]:



[9]:
```
Res=F.resultant(G,y)
Res=Rx(Res)
print(Res)
Abs=Res.roots(RR,False) # False pour cacher les multiplicités
Abs
```

```
2*x^28 - 16*x^27 + 32*x^26 + 249*x^24 + 48*x^23 - 128*x^22 + 4*x^21 - 757*x^20 -
112*x^19 + 192*x^18 - 12*x^17 + 758*x^16 + 144*x^15 - 126*x^14 + 28*x^13 -
251*x^12 - 64*x^11 + 30*x^10 - 36*x^9 - x^8 + 16*x^5 + 1
```

[9]:
```
[-0.924209668349044, -0.597428986963397, 0.721113386166218, 0.966506296874216]
```

On cherche les ordonnées des points d'intersections, qui correspondent aux zéros communs de F(a,y) et B(a,y) quand a parcourt les racines de Res. Attention, pas de tests à zéro (donc pas de pgcd) à notre disposition car a est une valeur numérique flottante ici. On remplace ici le test à zéro par plus petit que ``epsilon''.

[10]:
```
Ry.<y>=RR['y']
L=[]
for a in Abs:
    Fa=Ry(F(a,y))  # pour voir F(a,y) en un polynôme univarié en y
    Orda=Fa.roots(RR,False)
    for b in Orda:
        if abs(G(a,b))<1e-15:  # on remplace le test à zéro par "est plus petit␣
 ↪que epsilon"
            L.append((a,b))
L
```

[10]:
```
[(-0.924209668349044, 0.721113386166218),
 (-0.597428986963397, 0.966506296874216),
 (0.721113386166218, -0.924209668349044),
 (0.966506296874216, -0.597428986963397)]
```

Comparaison avec la résolution de Sage : soit avec ``solve'', soit avec ``ideal'' and ``variety''

[11]:
```
x,y=var('x,y')
D=solve([F(x,y)==0,G(x,y)==0],x,y,solution_dict=True)
LR=[(d[x],d[y]) for d in D if d[x] in RR]  # pour garder les solutions réelles
print(LR)
```

```
[(-0.5974289580514208, 0.9665062916358254), (0.7211134453781513,
-0.924209614551754), (-0.924209614551754, 0.7211134453781513),
(0.9665062916358254, -0.5974289580514208)]
```

[12]:
```
I=R.ideal(F,G)
I.variety(RR)
```

[12]:
```
[{y: -0.924209668349044, x: 0.721113386166217},
 {y: -0.597428986963397, x: 0.966506296874216},
 {y: 0.721113386166218, x: -0.924209668349045},
 {y: 0.966506296874216, x: -0.597428986963401}]
```

0.0.3 Exo 3 (implicitation avec résultant)

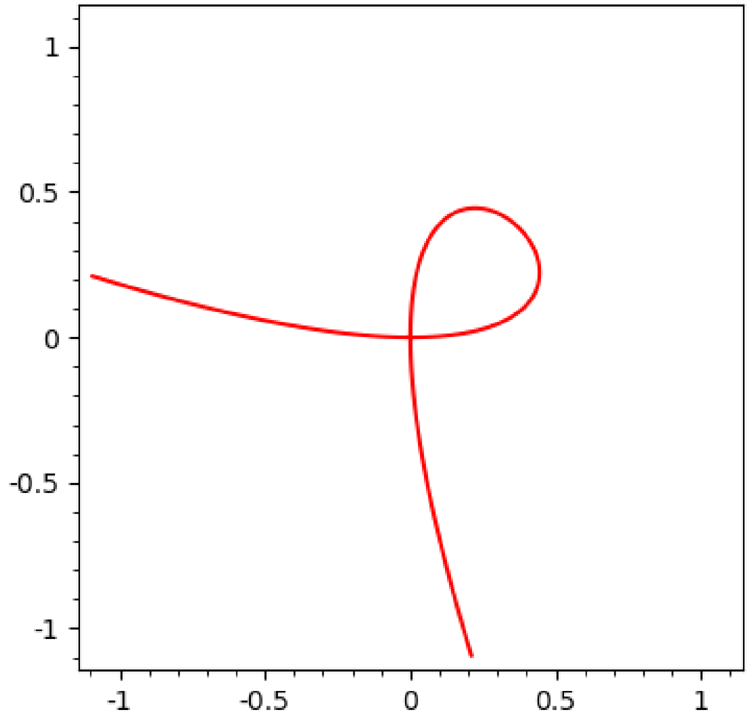
[13]:
```
R.<X,Y,T>=QQ['X,Y,T']

def EquationImplicite(A,B):
    P=X*A.denominator()-A.numerator()
    Q=Y*B.denominator()-B.numerator()
    Res=P.resultant(Q,T)
    return Res//Res.content()
```

[14]:
```
A=3*T/(1+T)**3
B=3*T^2/(1+T)**3
Res=EquationImplicite(A,B)
print(Res)
print(Res(A,B,1))    # Res constant en T mais considéré dans Q[X,Y,T] par Sage␣
 ↪(dommage...).
```

```
-X^3 - 3*X^2*Y - 3*X*Y^2 - Y^3 + 3*X*Y
0
```

[15]:
```
R1.<X,Y>=QQ['X,Y'] # On force Res à être dans Q[X,Y] pour pouvoir tracer la␣
 ↪courbe
Res=R1(Res)
implicit_plot(Res,(-1.1,1.1),(-1.1,1.1),color='red')
```

[15]:

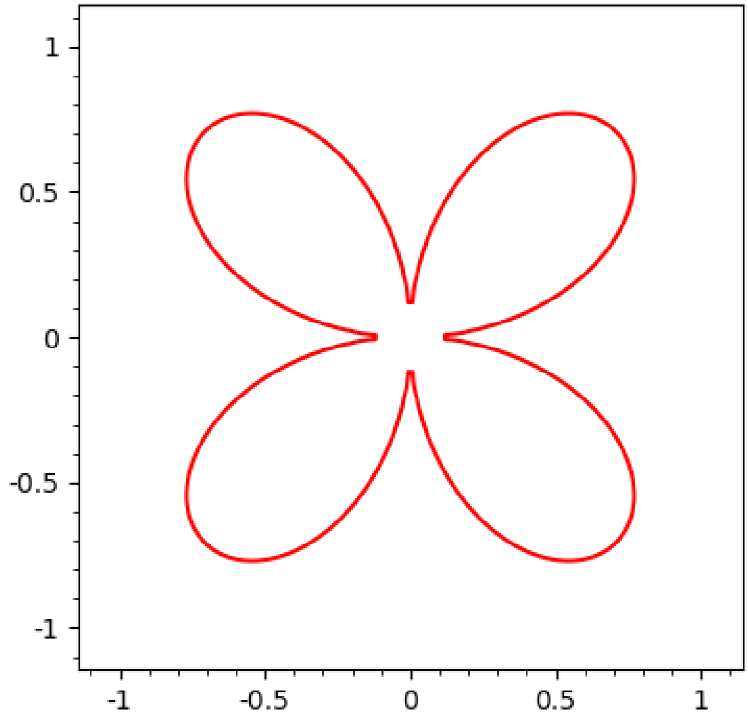


[16]:
```
A=4*T*(1-T**2)**2/(1+T**2)**3
B=8*T**2*(1-T**2)/(1+T**2)**3
Res=EquationImplicite(A,B)
print(Res)
print(Res(A,B,1))
```

```
X^6 + 3*X^4*Y^2 + 3*X^2*Y^4 + Y^6 - 4*X^2*Y^2
0
```

[17]:
```
Res=R1(Res)
implicit_plot(Res,(-1.1,1.1),(-1.1,1.1),color='red')
```

[17]:



0.0.4 Exo 4

[18]:
```
R4=PolynomialRing(QQ,'x',4,order='lex')
x=R4.gens()
P=(sum (x[i] for i in range(4)))**2
print(P)
print(P.lt())
P.parent()
```

```
x0^2 + 2*x0*x1 + 2*x0*x2 + 2*x0*x3 + x1^2 + 2*x1*x2 + 2*x1*x3 + x2^2 + 2*x2*x3 +
x3^2
x0^2
```

[18]: Multivariate Polynomial Ring in x0, x1, x2, x3 over Rational Field

[19]:
```
def sigma(n):
    Rn=PolynomialRing(QQ,'x',n,order='lex')
    x=Rn.gens()
    L=[0..n-1]
    LL=list(subsets(L))
    Res=[]
    for k in range(1,n+1):
        LLk=[l for l in LL if len(l)==k]
```

```
        sigmak=sum(prod(x[i] for i in l) for l in LLk)
        Res.append(sigmak)
    return Res

sigma(4)
```

[19]:
```
[x0 + x1 + x2 + x3,
 x0*x1 + x0*x2 + x0*x3 + x1*x2 + x1*x3 + x2*x3,
 x0*x1*x2 + x0*x1*x3 + x0*x2*x3 + x1*x2*x3,
 x0*x1*x2*x3]
```

[20]:
```
## Avec Sage :

Sym = SymmetricFunctions(QQ)
e=Sym.elementary()

def sigmaSage(n):
    Res=[]
    for k in range(1,n+1):
        P=e[k].expand(n)
        Res.append(P)
    return Res

sigmaSage(5)==sigma(5)
```

[20]:
```
True
```

[21]:
```
def TestSymetrique(P):
    n=len(P.parent().gens()) # nombre de variables
    for i in range(n-1):
        if P!=P.subs({x[i]:x[i+1],x[i+1]:x[i]}):
            return False
    return True
```

[22]:
```
P=sigma(4)[2]
print(P)
print(TestSymetrique(P))
Q=P+x[2]+x[3]
print(Q)
print(TestSymetrique(Q))
```

```
x0*x1*x2 + x0*x1*x3 + x0*x2*x3 + x1*x2*x3
True
x0*x1*x2 + x0*x1*x3 + x0*x2*x3 + x1*x2*x3 + x2 + x3
False
```

[23]:
```
def SymConvert(P):
    n=len(P.parent().gens())
    Sigma=sigma(n)
    Sn=PolynomialRing(QQ,'s',n,order='lex')
    s=Sn.gens()
    Q=0
    while P!=0:
        M=P.lm()  # leading monomial
        c=P.lc()  # leading coefficient
        I=M.exponents()[0]  # exposants de M
        MQ=c*prod(s[k]**(I[k]-I[k+1]) for k in range(n-1))*s[n-1]**I[n-1]
        Q=Q+MQ
        MP=c*prod(Sigma[k]**(I[k]-I[k+1]) for k in␣
↪range(n-1))*Sigma[n-1]**I[n-1]
        P=P-MP
    return Q
```

[24]:
```
P=sum(x[i]**2 for i in range(4))
print(P)
Q=SymConvert(P)
print(Q)
```

```
x0^2 + x1^2 + x2^2 + x3^2
s0^2 - 2*s1
```

[25]:
```
S=sigma(4)
s=Q.parent().gens()
print(Q.subs({s[i]:S[i] for i in range(4)}))
print(P)
```

```
x0^2 + x1^2 + x2^2 + x3^2
x0^2 + x1^2 + x2^2 + x3^2
```

[ ]:
```

```

0.0.5 Exo 7

[26]:
```
def Division(P,G): # G=[G_1,...,G_r]
    r=len(G)
    Q=[0 for i in range(r)]
    R=0
    F=P
    while F!=0:
        i=0
        while i<r and F.lm()%G[i].lm()!=0:
            i=i+1
        if i<r:
```

```
            c=F.lt()//G[i].lt()
            Q[i]=Q[i]+c
            F=F-c*G[i]
        else:
            R=R+F.lt()
            F=F-F.lt()
    return [Q,R]
```

[27]:
```
P=R4.random_element(degree=5)
x=R4.gens()
G1=R4.random_element()
G2=R4.random_element()
G=[G1,G2]
[Q,R]=Division(P,G)
print(Q,R)
P==sum(Q[i]*G[i] for i in range(len(G)))+R
```

```
[-2*x3, -x0^2*x3 - x1*x2^2*x3^2 + x2^2*x3^3 + 3*x2^2*x3] 0
```

[27]: True

[28]:
```
P=R4.random_element(degree=5)
G1=R4.random_element()
G=[G1]
[Q,R]=Division(P,G)
print(Q[0])
print(R)
print()
print(P//G1)
print(P%G1)
```

```
1/13*x1*x3 - 1/169*x1
-x0^2*x2^3 + 1/4*x0*x1^4 + 1/169*x0*x1 - 5/2*x1^4 + 1/13*x1*x3^3 - 1/169*x1*x3^2

-x0*x1
13*x0^2*x1*x3 + x0^2*x1 - x0^2*x2^3 + 1/4*x0*x1^4 - 5/2*x1^4
```

0.0.6 Exo 8 : tester si une famille est une base de Gröbner

[29]:
```
def Syzygy(F,G):
    S=(G*F.lt()-F*G.lt())//G.lm().gcd(F.lm())
    return S

def TestGrobner(I):
    n=len(I)
    for i in range(n):
        for j in range(i+1,n):
            S=Syzygy(I[i],I[j])
```

```
            R=Division(S,I)[1]
            if R!=0:
                return False
    return True

R3.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='lex')

F=2*x**2*y-x+z**2
G=3*x-y**2-z*y

print(TestGrobner([F,G]))
print()
I=R3.ideal(F,G)
J=I.groebner_basis()
print(I)
print()
print(J)
print()
TestGrobner(J)
```

```
False

Ideal (2*x^2*y - x + z^2, 3*x - y^2 - y*z) of Multivariate Polynomial Ring in x,
y, z over Rational Field

[x - 1/3*y^2 - 1/3*y*z, y^5 + 2*y^4*z + y^3*z^2 - 3/2*y^2 - 3/2*y*z + 9/2*z^2]
```

[29]:

```
True
```

0.0.7 Exo 9 : Résolution de systèmes, idéaux éliminants, quotient

[30]:

```
R3.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='lex')
G1=x**2 + y**2 + z**2 - 4
G2=x**2 + 2*y**2 - 5
G3=x*z - 1
I=R3.ideal(G1,G2,G3)
f=x**2+ 2*x*z**3 - 3*x*z - y**2* z + z**3 + z
g=x*y+x**3
print('solutions dans la cloture algebrique',I.variety(QQbar))
print('solutions rationnelles',I.variety(QQ))
print('f s\'annule sur V(I) :',f in I.radical())  # f s'annule sur la variété␣
↪ssi il est dans le radical de I (Nullstellensatz)
print('base de Grobner:',I.groebner_basis())
print('premier idéal éliminant:', I.elimination_ideal([x]))
print('second idéal éliminant:',I.elimination_ideal([x,y]))
print('base monomiale du quotient:',I.normal_basis())
```

```
print('reduction naïve de g:',g.reduce(I))
print('reduction totale de g:',g.reduce(I.groebner_basis()))
```

```
solutions dans la cloture algebrique [{z: 1, y: -1.414213562373095?, x: 1}, {z:
1, y: 1.414213562373095?, x: 1}, {z: -1, y: -1.414213562373095?, x: -1}, {z: -1,
y: 1.414213562373095?, x: -1}, {z: -0.7071067811865475?, y: -1.224744871391589?,
x: -1.414213562373095?}, {z: -0.7071067811865475?, y: 1.224744871391589?, x:
-1.414213562373095?}, {z: 0.7071067811865475?, y: -1.224744871391589?, x:
1.414213562373095?}, {z: 0.7071067811865475?, y: 1.224744871391589?, x:
1.414213562373095?}]
solutions rationnelles []
f s'annule sur V(I) : True
base de Grobner: [x + 2*z^3 - 3*z, y^2 - z^2 - 1, z^4 - 3/2*z^2 + 1/2]
premier idéal éliminant: Ideal (2*z^4 - 3*z^2 + 1, y^2 - z^2 - 1) of
Multivariate Polynomial Ring in x, y, z over Rational Field
second idéal éliminant: Ideal (2*z^4 - 3*z^2 + 1) of Multivariate Polynomial
Ring in x, y, z over Rational Field
base monomiale du quotient: [y*z^3, z^3, y*z^2, z^2, y*z, z, y, 1]
reduction naïve de g: -2*y*z^3 + 3*y*z - 8*z^9 + 36*z^7 - 54*z^5 + 27*z^3
reduction totale de g: -2*y*z^3 + 3*y*z - 6*z^3 + 7*z
```

Les monômes dominants de la base de Gröbner sont x, y^2 et z^4. On vérifie que les monômes qui ne sont pas dans l'idéal engendré par ces éléments sont bien $yz^3, z^3, yz^2, z^2, yz, z, y, 1$, qui forment donc une base de $\mathbb{Q}[x, y, z]/(I)$, conformément au théorème de décomposition du quotient.

La réduction ``naïve'' de g modulo I est le reste d'une division multivariée par la suite des générateurs G_1, G_2, G_3. Pour avoir l'image dans la base monomiale du quotient, il faut faire une division multivariée par une base de Gröbner.

0.0.8 Exo 10 : Influence de l'ordre monomial

[31]:
```
R3glex.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='degrevlex')
I=R3.ideal(x**5 + y**4 + z**3 -1, x**3 + y**2 + z**2 -1)
print("Base gröbner lex de I:\n",I.groebner_basis())
print()
I=R3glex.ideal(x**5 + y**4 + z**3 -1, x**3 + y**2 + z**2 -1)
print("Base gröbner grevlex de I:\n",I.groebner_basis())
print()
J=R3.ideal(x**5 + y**4 + z**3 -1, x**3 + y**3 + z**2 -1)
print("Base gröbner lex de J:\n",J.groebner_basis())
print()
J=R3glex.ideal(x**5 + y**4 + z**3 -1, x**3 + y**3 + z**2 -1)
print("Base gröbner grevlex de J\n:",J.groebner_basis())
print()
```

```
Base gröbner lex de I:
 [x^3 + y^2 + z^2 - 1, x^2*y^2 + x^2*z^2 - x^2 - y^4 - z^3 + 1, x^2*z^4 +
x^2*z^3 - 2*x^2*z^2 - 1/6*x*z^10 - 2/3*x*z^9 - 1/4*x*z^8 + 7/6*x*z^7 -
```

```
1/12*x*z^6 - x*z^5 + x*z^4 + 1/2*y^10*z^2 + 1/3*y^10*z - 5/12*y^10 - 1/6*y^8*z^4
+ 1/3*y^8*z^3 + 1/4*y^8*z^2 + 1/2*y^8 + y^6*z^5 - 5/3*y^6*z^4 - 17/6*y^6*z^3 +
17/6*y^6*z^2 + 2/3*y^6*z - 1/3*y^6 - 1/3*y^4*z^7 - 7/2*y^4*z^6 - 3*y^4*z^5 +
131/12*y^4*z^4 + 23/4*y^4*z^3 - 13*y^4*z^2 - 8/3*y^4*z + 29/6*y^4 - 17/6*y^2*z^8
- 4*y^2*z^7 + 119/12*y^2*z^6 + 17/2*y^2*z^5 - 16*y^2*z^4 - 14/3*y^2*z^3 +
12*y^2*z^2 + 5/3*y^2*z - 55/12*y^2 - 7/6*z^10 - 7/6*z^9 + 55/12*z^8 + 21/4*z^7 -
31/4*z^6 - 15/2*z^5 + 83/12*z^4 + 41/12*z^3 - 31/12*z^2, x*y^4 + x*z^3 - x + y^4
+ 2*y^2*z^2 - 2*y^2 + z^4 - 2*z^2 + 1, x*y^2*z - x*y^2 - 1/8*x*z^10 - 7/24*x*z^9
+ 1/2*x*z^8 + 7/8*x*z^7 - 23/24*x*z^6 + 1/2*x*z^4 - 1/2*x*z^3 - x*z + x +
3/8*y^10*z^2 - 3/8*y^10*z - 7/24*y^10 - 1/8*y^8*z^4 + 11/24*y^8*z^3 -
3/8*y^8*z^2 + 1/4*y^8*z - 1/4*y^8 + 3/4*y^6*z^5 - 5/2*y^6*z^4 + 5/6*y^6*z^3 +
31/12*y^6*z^2 - y^6*z - 1/3*y^6 - 1/4*y^4*z^7 - 53/24*y^4*z^6 + 11/6*y^4*z^5 +
113/12*y^4*z^4 - 55/8*y^4*z^3 - 29/4*y^4*z^2 + 15/4*y^4*z + 19/12*y^4 -
17/8*y^2*z^8 + 13/24*y^2*z^7 + 239/24*y^2*z^6 - 59/12*y^2*z^5 - 151/12*y^2*z^4 +
85/12*y^2*z^3 + 43/8*y^2*z^2 - 29/8*y^2*z + 7/24*y^2 - 7/8*z^10 + 7/12*z^9 +
31/8*z^8 - 11/12*z^7 - 103/12*z^6 + 17/6*z^5 + 127/24*z^4 - 3/2*z^3 - 17/24*z^2
+ z - 1, x*z^11 + 4*x*z^10 + x*z^9 - 10*x*z^8 - 4*x*z^7 + 8*x*z^6 - 3*y^10*z^3 -
2*y^10*z^2 + 4*y^10*z + 4*y^10 + y^8*z^5 - 2*y^8*z^4 - 2*y^8*z^3 - 6*y^6*z^6 +
10*y^6*z^5 + 20*y^6*z^4 - 16*y^6*z^3 - 24*y^6*z^2 + 8*y^6*z + 8*y^6 + 2*y^4*z^8
+ 21*y^4*z^7 + 17*y^4*z^6 - 78*y^4*z^5 - 64*y^4*z^4 + 90*y^4*z^3 + 76*y^4*z^2 -
32*y^4*z - 32*y^4 + 17*y^2*z^9 + 24*y^2*z^8 - 68*y^2*z^7 - 80*y^2*z^6 +
106*y^2*z^5 + 108*y^2*z^4 - 77*y^2*z^3 - 70*y^2*z^2 + 20*y^2*z + 20*y^2 + 7*z^11
+ 7*z^10 - 31*z^9 - 42*z^8 + 55*z^7 + 77*z^6 - 39*z^5 - 62*z^4 + 8*z^3 + 20*z^2,
y^12 - y^10 + 3*y^8*z^3 - 5*y^8*z^2 + 2*y^8 - 10*y^6*z^4 + 20*y^6*z^2 - 10*y^6 -
7*y^4*z^6 + 30*y^4*z^4 - 6*y^4*z^3 - 30*y^4*z^2 + 13*y^4 - 5*y^2*z^8 +
20*y^2*z^6 - 30*y^2*z^4 + 20*y^2*z^2 - 5*y^2 - z^10 + z^9 + 5*z^8 - 13*z^6 +
10*z^4 + 3*z^3 - 5*z^2]
```

Base gröbner grevlex de I:

```
 [y^6 - y^4*z^2 + x^2*z^4 + 2*x*y^2*z^2 + x^2*z^3 + y^2*z^3 + x*z^4 - z^5 -
2*x^2*z^2 - 2*y^2*z^2 - x*z^3 - z^4 - 2*x*y^2 - 2*x*z^2 + z^3 + y^2 + 3*z^2 +
2*x - 2, x*y^4 + y^4 + 2*y^2*z^2 + x*z^3 + z^4 - 2*y^2 - 2*z^2 - x + 1, x^2*y^2
- y^4 + x^2*z^2 - z^3 - x^2 + 1, x^3 + y^2 + z^2 - 1]
```

Base gröbner lex de J:

```
 [x^3 + y^3 + z^2 - 1, x^2*y^3 + x^2*z^2 - x^2 - y^4 - z^3 + 1, x^2*y*z - x^2*y
+ x^2*z^6 + 2*x^2*z^5 + 3*x^2*z^4 - 3*x^2*z^3 - 3*x^2*z^2 - x^2*z + x^2 +
7*x*y^3*z - 7*x*y^3 - 12*x*y^2*z + 12*x*y^2 - 62093/7776*x*z^20 -
15037/243*x*z^19 - 1974401/7776*x*z^18 - 2274307/3888*x*z^17 -
2701393/3888*x*z^16 + 1471927/7776*x*z^15 + 160163/81*x*z^14 + 820093/288*x*z^13
+ 1775483/2592*x*z^12 - 6997327/2592*x*z^11 - 8397505/2592*x*z^10 -
615715/1296*x*z^9 + 10699009/7776*x*z^8 + 7091971/7776*x*z^7 + 217021/7776*x*z^6
+ x*z^5 + 5/4*x*z^4 + 3/4*x*z^3 + 5*x*z - 5*x - 62093/7776*y^14*z^11 +
450211/7776*y^14*z^10 + 1224587/3888*y^14*z^9 + 572525/864*y^14*z^8 +
61387/81*y^14*z^7 + 999865/2592*y^14*z^6 - 364105/864*y^14*z^5 -
40125/32*y^14*z^4 - 3812519/2592*y^14*z^3 - 3424889/3888*y^14*z^2 -
945749/3888*y^14*z - 33719/3888*y^14 + 310465/3888*y^13*z^11 +
644123/1944*y^13*z^10 + 2700925/3888*y^13*z^9 + 3905987/3888*y^13*z^8 +
```

4334645/3888*y^13*z^7 + 1493281/1944*y^13*z^6 - 380585/1944*y^13*z^5 - 293434/243*y^13*z^4 - 2832059/1944*y^13*z^3 - 1726417/1944*y^13*z^2 - 942347/3888*y^13*z - 39065/3888*y^13 + 62093/1296*y^12*z^12 + 287125/1944*y^12*z^11 + 431569/1944*y^12*z^10 + 153943/486*y^12*z^9 + 2668523/3888*y^12*z^8 + 4595723/3888*y^12*z^7 + 530431/486*y^12*z^6 + 24677/486*y^12*z^5 - 281903/243*y^12*z^4 - 705055/486*y^12*z^3 - 3447245/3888*y^12*z^2 - 955469/3888*y^12*z - 27887/3888*y^12 - 62093/3888*y^11*z^13 + 1195327/3888*y^11*z^12 + 11665603/7776*y^11*z^11 + 20826185/7776*y^11*z^10 + 7257919/3888*y^11*z^9 - 1093649/864*y^11*z^8 - 2191181/432*y^11*z^7 - 18490621/2592*y^11*z^6 - 3942919/864*y^11*z^5 + 15841297/7776*y^11*z^4 + 47780995/7776*y^11*z^3 + 5657215/1296*y^11*z^2 + 4704931/3888*y^11*z + 186577/3888*y^11 + 62093/7776*y^10*z^14 + 884813/2592*y^10*z^13 + 64403/48*y^10*z^12 + 4517213/1944*y^10*z^11 + 2017553/864*y^10*z^10 + 128209/96*y^10*z^9 - 4288277/3888*y^10*z^8 - 43983913/7776*y^10*z^7 - 8543879/972*y^10*z^6 - 14644769/2592*y^10*z^5 + 13952111/7776*y^10*z^4 + 47175827/7776*y^10*z^3 + 2849465/648*y^10*z^2 + 4704445/3888*y^10*z + 196783/3888*y^10 + 310465/1944*y^9*z^14 + 131917/324*y^9*z^13 + 165697/1296*y^9*z^12 - 147217/648*y^9*z^11 + 426449/324*y^9*z^10 + 2550619/648*y^9*z^9 + 9873259/3888*y^9*z^8 - 17557669/3888*y^9*z^7 - 4920281/486*y^9*z^6 - 1112293/162*y^9*z^5 + 37493/24*y^9*z^4 + 3906707/648*y^9*z^3 + 5695771/1296*y^9*z^2 + 4786579/3888*y^9*z + 124369/3888*y^9 - 62093/7776*y^8*z^15 + 1350535/2592*y^8*z^14 + 18125525/7776*y^8*z^13 + 7103509/1944*y^8*z^12 + 798427/648*y^8*z^11 - 2014181/486*y^8*z^10 - 69123395/7776*y^8*z^9 - 10136435/972*y^8*z^8 - 17653931/3888*y^8*z^7 + 66547639/7776*y^8*z^6 + 115182683/7776*y^8*z^5 + 24044455/3888*y^8*z^4 - 269935/72*y^8*z^3 - 4162441/972*y^8*z^2 - 4603357/3888*y^8*z - 270655/3888*y^8 + 62093/3888*y^7*z^16 + 212147/432*y^7*z^15 + 3571399/1944*y^7*z^14 + 1206181/486*y^7*z^13 + 1186199/1296*y^7*z^12 - 515045/432*y^7*z^11 - 744575/216*y^7*z^10 - 11042377/1296*y^7*z^9 - 7025765/648*y^7*z^8 - 2247317/3888*y^7*z^7 + 2195017/144*y^7*z^6 + 66891659/3888*y^7*z^5 + 1642999/486*y^7*z^4 - 15580525/1944*y^7*z^3 - 13532807/1944*y^7*z^2 - 3768173/1944*y^7*z - 142895/1944*y^7 - 62093/7776*y^6*z^17 + 357047/2592*y^6*z^16 + 525857/3888*y^6*z^15 - 10057949/7776*y^6*z^14 - 10786067/3888*y^6*z^13 + 6413327/7776*y^6*z^12 + 1096481/144*y^6*z^11 + 4906787/972*y^6*z^10 - 78227581/7776*y^6*z^9 - 47738575/2592*y^6*z^8 - 16274735/3888*y^6*z^7 + 132719201/7776*y^6*z^6 + 49492879/2592*y^6*z^5 + 1078505/288*y^6*z^4 - 61589201/7776*y^6*z^3 - 13586267/1944*y^6*z^2 - 3828923/1944*y^6*z - 89921/1944*y^6 + 683023/1944*y^5*z^16 + 915737/648*y^5*z^15 + 1035847/648*y^5*z^14 - 1285447/972*y^5*z^13 - 5011187/972*y^5*z^12 - 3184595/486*y^5*z^11 - 3796163/648*y^5*z^10 - 19495/216*y^5*z^9 + 25380091/1944*y^5*z^8 + 11892013/648*y^5*z^7 + 7211327/1944*y^5*z^6 - 22648957/1944*y^5*z^5 - 4551529/486*y^5*z^4 - 31405/54*y^5*z^3 + 3113069/1944*y^5*z^2 + 109258/243*y^5*z + 12082/243*y^5 + 62093/7776*y^4*z^18 + 2157695/7776*y^4*z^17 + 1180093/1296*y^4*z^16 + 3247837/3888*y^4*z^15 - 115379/486*y^4*z^14 + 110635/972*y^4*z^13 + 693821/3888*y^4*z^12 - 16622791/2592*y^4*z^11 - 105032993/7776*y^4*z^10 - 15910187/3888*y^4*z^9 + 129546847/7776*y^4*z^8 + 83691797/3888*y^4*z^7 + 7163699/1944*y^4*z^6 -

31024403/2592*y^4*z^5 - 24365701/2592*y^4*z^4 - 5850743/7776*y^4*z^3 +
407632/243*y^4*z^2 + 115333/243*y^4*z + 6250/243*y^4 - 62093/3888*y^3*z^19 -
15511/1296*y^3*z^18 - 871439/2592*y^3*z^17 - 14439323/7776*y^3*z^16 -
10235095/3888*y^3*z^15 + 20484593/7776*y^3*z^14 + 41801113/3888*y^3*z^13 +
51887995/7776*y^3*z^12 - 46707793/3888*y^3*z^11 - 80508941/3888*y^3*z^10 -
13363721/7776*y^3*z^9 + 56751797/2592*y^3*z^8 + 68978845/3888*y^3*z^7 -
45834331/7776*y^3*z^6 - 128446523/7776*y^3*z^5 - 55627499/7776*y^3*z^4 +
27747047/7776*y^3*z^3 + 16868221/3888*y^3*z^2 + 4824487/3888*y^3*z +
63133/3888*y^3 + 310465/3888*y^2*z^18 + 325927/1296*y^2*z^17 -
69823/1944*y^2*z^16 - 1641901/1296*y^2*z^15 - 8062141/3888*y^2*z^14 -
2135233/1944*y^2*z^13 + 1660715/3888*y^2*z^12 + 10035251/3888*y^2*z^11 +
2132911/324*y^2*z^10 + 12725305/1944*y^2*z^9 - 882389/324*y^2*z^8 -
10152877/972*y^2*z^7 - 7703591/1296*y^2*z^6 + 8885575/3888*y^2*z^5 +
14077583/3888*y^2*z^4 + 1111699/972*y^2*z^3 + 269197/3888*y^2*z^2 + 12*y^2*z -
12*y^2 + 62093/1296*y*z^19 + 387971/3888*y*z^18 - 214169/1296*y*z^17 -
2316565/3888*y*z^16 - 69113/3888*y*z^15 + 2592223/3888*y*z^14 -
2217817/1296*y*z^13 - 17385025/3888*y*z^12 + 645061/1296*y*z^11 +
2433767/243*y*z^10 + 18230531/1944*y*z^9 - 1576189/486*y*z^8 - 4934963/432*y*z^7
- 23785315/3888*y*z^6 + 1410127/648*y*z^5 + 3527597/972*y*z^4 +
1169533/972*y*z^3 + 123883/3888*y*z^2 - y*z + y - 62093/7776*z^21 -
294905/7776*z^20 - 1896895/7776*z^19 - 2836079/3888*z^18 - 1896163/3888*z^17 +
4339531/1944*z^16 + 18930365/3888*z^15 + 4109437/7776*z^14 - 75897065/7776*z^13
- 89966545/7776*z^12 + 21700007/7776*z^11 + 5657717/324*z^10 + 32836639/2592*z^9
- 19506331/3888*z^8 - 51629797/3888*z^7 - 50374091/7776*z^6 + 17195287/7776*z^5
+ 13970663/3888*z^4 + 604571/486*z^3 + 47581/3888*z^2 - 4*z + 4, x^2*z^7 +
3*x^2*z^6 + 5*x^2*z^5 - 5*x^2*z^3 - 4*x^2*z^2 + 18*x*y^3*z - 18*x*y^3 -
27*x*y^2*z + 27*x*y^2 - 62197/3888*x*z^20 - 477679/3888*x*z^19 -
1943665/3888*x*z^18 - 275920/243*x*z^17 - 5079481/3888*x*z^16 +
1881863/3888*x*z^15 + 1277051/324*x*z^14 + 585259/108*x*z^13 +
1223257/1296*x*z^12 - 1792241/324*x*z^11 - 494006/81*x*z^10 - 588079/1296*x*z^9
+ 5526469/1944*x*z^8 + 6288353/3888*x*z^7 - 171599/1944*x*z^6 + 3/2*x*z^5 +
3*x*z^4 + 9*x*z - 9*x - 62197/3888*y^14*z^11 + 113819/972*y^14*z^10 +
2422655/3888*y^14*z^9 + 69253/54*y^14*z^8 + 1841305/1296*y^14*z^7 +
853763/1296*y^14*z^6 - 24436/27*y^14*z^5 - 1059965/432*y^14*z^4 -
447539/162*y^14*z^3 - 1500209/972*y^14*z^2 - 683551/1944*y^14*z + 18625/972*y^14
+ 310985/1944*y^13*z^11 + 1268849/1944*y^13*z^10 + 2612831/1944*y^13*z^9 +
3714277/1944*y^13*z^8 + 4052521/1944*y^13*z^7 + 2667577/1944*y^13*z^6 -
495835/972*y^13*z^5 - 4658429/1944*y^13*z^4 - 2664679/972*y^13*z^3 -
379001/243*y^13*z^2 - 680149/1944*y^13*z + 16195/972*y^13 + 62197/648*y^12*z^12
+ 562279/1944*y^12*z^11 + 102851/243*y^12*z^10 + 1169275/1944*y^12*z^9 +
322844/243*y^12*z^8 + 4410193/1944*y^12*z^7 + 3911539/1944*y^12*z^6 -
126505/1944*y^12*z^5 - 2273447/972*y^12*z^4 - 5320567/1944*y^12*z^3 -
1513331/972*y^12*z^2 - 694243/1944*y^12*z + 22513/972*y^12 -
62197/1944*y^11*z^13 + 150205/243*y^11*z^12 + 11520437/3888*y^11*z^11 +
5006425/972*y^11*z^10 + 12994009/3888*y^11*z^9 - 610735/216*y^11*z^8 -
1438795/144*y^11*z^7 - 17560223/1296*y^11*z^6 - 436225/54*y^11*z^5 +
18625895/3888*y^11*z^4 + 11704235/972*y^11*z^3 + 2534063/324*y^11*z^2 +
3392969/1944*y^11*z - 82919/972*y^11 + 62197/3888*y^10*z^14 +

442429/648*y^10*z^13 + 3421945/1296*y^10*z^12 + 17348077/3888*y^10*z^11 +
5615981/1296*y^10*z^10 + 752717/324*y^10*z^9 - 4695277/1944*y^10*z^8 -
5435419/486*y^10*z^7 - 65292989/3888*y^10*z^6 - 810656/81*y^10*z^5 +
17408485/3888*y^10*z^4 + 23167685/1944*y^10*z^3 + 1279303/162*y^10*z^2 +
3393455/1944*y^10*z - 79517/972*y^10 + 310985/972*y^9*z^14 + 85697/108*y^9*z^13
+ 42407/216*y^9*z^12 - 103399/216*y^9*z^11 + 216385/81*y^9*z^10 +
4993523/648*y^9*z^9 + 4383565/972*y^9*z^8 - 18382733/1944*y^9*z^7 -
38206769/1944*y^9*z^6 - 7934489/648*y^9*z^5 + 1361629/324*y^9*z^4 +
7696165/648*y^9*z^3 + 2558201/324*y^9*z^2 + 3481421/1944*y^9*z - 119855/972*y^9
- 62197/3888*y^8*z^15 + 677117/648*y^8*z^14 + 4468807/972*y^8*z^13 +
6790313/972*y^8*z^12 + 1245179/648*y^8*z^11 - 33102991/3888*y^8*z^10 -
66936577/3888*y^8*z^9 - 9515029/486*y^8*z^8 - 14650573/1944*y^8*z^7 +
69509327/3888*y^8*z^6 + 13836461/486*y^8*z^5 + 19730045/1944*y^8*z^4 -
230074/27*y^8*z^3 - 3896771/486*y^8*z^2 - 3283619/1944*y^8*z + 34805/972*y^8 +
62197/1944*y^7*z^16 + 318035/324*y^7*z^15 + 7021567/1944*y^7*z^14 +
9150649/1944*y^7*z^13 + 945481/648*y^7*z^12 - 274297/108*y^7*z^11 -
20197/3*y^7*z^10 - 5368237/324*y^7*z^9 - 13276025/648*y^7*z^8 +
461875/972*y^7*z^7 + 19909601/648*y^7*z^6 + 31341785/972*y^7*z^5 +
4016549/972*y^7*z^4 - 16177499/972*y^7*z^3 - 6190703/486*y^7*z^2 -
2719867/972*y^7*z + 69883/486*y^7 - 62197/3888*y^6*z^17 + 19949/72*y^6*z^16 +
979745/3888*y^6*z^15 - 5079635/1944*y^6*z^14 - 20912123/3888*y^6*z^13 +
8030785/3888*y^6*z^12 + 19665773/1296*y^6*z^11 + 35006123/3888*y^6*z^10 -
81555203/3888*y^6*z^9 - 1916497/54*y^6*z^8 - 21773417/3888*y^6*z^7 +
135926551/3888*y^6*z^6 + 1287217/36*y^6*z^5 + 5916637/1296*y^6*z^4 -
32085167/1944*y^6*z^3 - 3112483/243*y^6*z^2 - 2784991/972*y^6*z + 99529/486*y^6
+ 684167/972*y^5*z^16 + 75122/27*y^5*z^15 + 485837/162*y^5*z^14 -
1406837/486*y^5*z^13 - 2468096/243*y^5*z^12 - 12022495/972*y^5*z^11 -
1161359/108*y^5*z^10 + 225721/324*y^5*z^9 + 12742687/486*y^5*z^8 +
5653507/162*y^5*z^7 + 4520299/972*y^5*z^6 - 23357087/972*y^5*z^5 -
16545389/972*y^5*z^4 + 35243/108*y^5*z^3 + 1603811/486*y^5*z^2 +
150658/243*y^5*z + 9277/243*y^5 + 62197/3888*y^4*z^18 + 1078499/1944*y^4*z^17 +
771331/432*y^4*z^16 + 5993371/3888*y^4*z^15 - 147232/243*y^4*z^14 +
124853/486*y^4*z^13 + 1373297/3888*y^4*z^12 - 5566013/432*y^4*z^11 -
25434103/972*y^4*z^10 - 12031801/1944*y^4*z^9 + 132624275/3888*y^4*z^8 +
19792573/486*y^4*z^7 + 15863623/3888*y^4*z^6 - 887429/36*y^4*z^5 -
22074205/1296*y^4*z^4 - 65795/1944*y^4*z^3 + 1690805/486*y^4*z^2 +
330233/486*y^4*z - 4817/243*y^4 - 62197/1944*y^3*z^19 - 1175/54*y^3*z^18 -
290107/432*y^3*z^17 - 892627/243*y^3*z^16 - 19475671/3888*y^3*z^15 +
1379251/243*y^3*z^14 + 82420621/3888*y^3*z^13 + 45846113/3888*y^3*z^12 -
97789063/3888*y^3*z^11 - 154747355/3888*y^3*z^10 - 1148833/3888*y^3*z^9 +
9568087/216*y^3*z^8 + 125604427/3888*y^3*z^7 - 56776217/3888*y^3*z^6 -
15699683/486*y^3*z^5 - 45915757/3888*y^3*z^4 + 8058817/972*y^3*z^3 +
7925977/972*y^3*z^2 + 3522245/1944*y^3*z - 149015/972*y^3 + 310985/1944*y^2*z^18
+ 39911/81*y^2*z^17 - 210943/1944*y^2*z^16 - 821969/324*y^2*z^15 -
3862621/972*y^2*z^14 - 3672235/1944*y^2*z^13 + 2003287/1944*y^2*z^12 +
4967135/972*y^2*z^11 + 8306009/648*y^2*z^10 + 23634265/1944*y^2*z^9 -
1395853/216*y^2*z^8 - 40032863/1944*y^2*z^7 - 1675367/162*y^2*z^6 +
5383087/972*y^2*z^5 + 13455157/1944*y^2*z^4 + 3326843/1944*y^2*z^3 -

33347/972*y^2*z^2 + 27*y^2*z - 27*y^2 + 62197/648*y*z^19 + 46961/243*y*z^18 -
37357/108*y*z^17 - 2278337/1944*y*z^16 + 12601/243*y*z^15 + 2618267/1944*y*z^14
- 2286719/648*y*z^13 - 16970189/1944*y*z^12 + 359333/216*y*z^11 +
38981425/1944*y*z^10 + 4210861/243*y*z^9 - 15481069/1944*y*z^8 -
4854419/216*y*z^7 - 2558140/243*y*z^6 + 3457595/648*y*z^5 + 13502785/1944*y*z^4
+ 3592685/1944*y*z^3 - 117425/972*y*z^2 - 62197/3888*z^21 - 18193/243*z^20 -
939481/1944*z^19 - 2773717/1944*z^18 - 423455/486*z^17 + 17694533/3888*z^16 +
9179513/972*z^15 + 324893/972*z^14 - 38272073/1944*z^13 - 84649877/3888*z^12 +
14325983/1944*z^11 + 44942477/1296*z^10 + 29531183/1296*z^9 - 11709841/972*z^8 -
50399609/1944*z^7 - 42613627/3888*z^6 + 10608523/1944*z^5 + 13367677/1944*z^4 +
3763271/1944*z^3 - 162623/972*z^2 - 9*z + 9, x*y^4 + x*z^3 - x + y^6 + 2*y^3*z^2
- 2*y^3 + z^4 - 2*z^2 + 1, x*y*z^2 - 2*x*y*z + x*y - 44425/69984*x*z^20 -
163535/34992*x*z^19 - 639107/34992*x*z^18 - 2698855/69984*x*z^17 -
1286021/34992*x*z^16 + 2619449/69984*x*z^15 + 443767/2916*x*z^14 +
1277093/7776*x*z^13 - 436135/11664*x*z^12 - 2790745/11664*x*z^11 -
125515/729*x*z^10 + 1600469/23328*x*z^9 + 4427383/34992*x*z^8 +
946153/34992*x*z^7 - 1980847/69984*x*z^6 + 1/4*x*z^5 - 1/2*x*z^4 + 1/4*x*z^3 -
x*z^2 + 2*x*z - x - 44425/69984*y^14*z^11 + 339305/69984*y^14*z^10 +
1628711/69984*y^14*z^9 + 111115/2592*y^14*z^8 + 457223/11664*y^14*z^7 +
72931/11664*y^14*z^6 - 59791/1296*y^14*z^5 - 168541/1944*y^14*z^4 -
1797889/23328*y^14*z^3 - 196225/8748*y^14*z^2 + 158203/17496*y^14*z +
250613/34992*y^14 + 222125/34992*y^13*z^11 + 208925/8748*y^13*z^10 +
785005/17496*y^13*z^9 + 1011689/17496*y^13*z^8 + 1980241/34992*y^13*z^7 +
891121/34992*y^13*z^6 - 354533/8748*y^13*z^5 - 3162047/34992*y^13*z^4 -
1351039/17496*y^13*z^3 - 830827/34992*y^13*z^2 + 171325/17496*y^13*z +
237491/34992*y^13 + 44425/11664*y^12*z^12 + 89815/8748*y^12*z^11 +
455137/34992*y^12*z^10 + 631759/34992*y^12*z^9 + 389551/8748*y^12*z^8 +
158494/2187*y^12*z^7 + 860033/17496*y^12*z^6 - 68578/2187*y^12*z^5 -
3310039/34992*y^12*z^4 - 2749969/34992*y^12*z^3 - 101393/4374*y^12*z^2 +
158203/17496*y^12*z + 250613/34992*y^12 - 44425/34992*y^11*z^13 +
872405/34992*y^11*z^12 + 7686377/69984*y^11*z^11 + 11626231/69984*y^11*z^10 +
4404103/69984*y^11*z^9 - 420709/2592*y^11*z^8 - 1417253/3888*y^11*z^7 -
4737577/11664*y^11*z^6 - 32731/243*y^11*z^5 + 5470091/17496*y^11*z^4 +
29810057/69984*y^11*z^3 + 287521/1944*y^11*z^2 - 795389/17496*y^11*z -
1248691/34992*y^11 + 44425/69984*y^10*z^14 + 209105/7776*y^10*z^13 +
2242763/23328*y^10*z^12 + 4995035/34992*y^10*z^11 + 2600449/23328*y^10*z^10 +
717805/23328*y^10*z^9 - 9140507/69984*y^10*z^8 - 14566699/34992*y^10*z^7 -
36482669/69984*y^10*z^6 - 215965/1296*y^10*z^5 + 11468003/34992*y^10*z^4 +
29956543/69984*y^10*z^3 + 598613/3888*y^10*z^2 - 869747/17496*y^10*z -
1174333/34992*y^10 + 222125/17496*y^9*z^14 + 40025/1458*y^9*z^13 -
31325/11664*y^9*z^12 - 133003/5832*y^9*z^11 + 435073/3888*y^9*z^10 +
354365/1296*y^9*z^9 + 336577/4374*y^9*z^8 - 972461/2187*y^9*z^7 -
1454837/2187*y^9*z^6 - 430759/1944*y^9*z^5 + 4099939/11664*y^9*z^4 +
560825/1296*y^9*z^3 + 293839/1944*y^9*z^2 - 777893/17496*y^9*z -
1266187/34992*y^9 - 44425/69984*y^8*z^15 + 323995/7776*y^8*z^14 +
5923523/34992*y^8*z^13 + 952247/4374*y^8*z^12 - 149135/7776*y^8*z^11 -
26096155/69984*y^8*z^10 - 20168129/34992*y^8*z^9 - 19074349/34992*y^8*z^8 -
1947773/69984*y^8*z^7 + 29182693/34992*y^8*z^6 + 63744949/69984*y^8*z^5 +

```
165673/17496*y^8*z^4 - 489187/972*y^8*z^3 - 7674947/34992*y^8*z^2 +
817259/17496*y^8*z + 1226821/34992*y^8 + 44425/34992*y^7*z^16 +
449615/11664*y^7*z^15 + 4580449/34992*y^7*z^14 + 2445551/17496*y^7*z^13 -
84311/11664*y^7*z^12 - 54575/432*y^7*z^11 - 915223/3888*y^7*z^10 -
3314095/5832*y^7*z^9 - 6944063/11664*y^7*z^8 + 1333129/4374*y^7*z^7 +
1811375/1458*y^7*z^6 + 30909727/34992*y^7*z^5 - 2537123/8748*y^7*z^4 -
1660150/2187*y^7*z^3 - 2559353/8748*y^7*z^2 + 696235/8748*y^7*z +
939029/17496*y^7 - 44425/69984*y^6*z^17 + 261185/23328*y^6*z^16 +
456911/69984*y^6*z^15 - 7506655/69984*y^6*z^14 - 6334411/34992*y^6*z^13 +
5389457/34992*y^6*z^12 + 13651993/23328*y^6*z^11 + 11243399/69984*y^6*z^10 -
34018261/34992*y^6*z^9 - 26889559/23328*y^6*z^8 + 1149559/4374*y^6*z^7 +
52849991/34992*y^6*z^6 + 11403491/11664*y^6*z^5 - 1946329/5832*y^6*z^4 -
52987825/69984*y^6*z^3 - 1278583/4374*y^6*z^2 + 613129/8748*y^6*z +
1022135/17496*y^6 + 488675/17496*y^5*z^16 + 592115/5832*y^5*z^15 +
80563/972*y^5*z^14 - 343673/2187*y^5*z^13 - 6493751/17496*y^5*z^12 -
6239263/17496*y^5*z^11 - 742009/2916*y^5*z^10 + 1044131/5832*y^5*z^9 +
2290939/2187*y^5*z^8 + 6131711/5832*y^5*z^7 - 5292641/17496*y^5*z^6 -
18535223/17496*y^5*z^5 - 1604699/4374*y^5*z^4 + 56383/216*y^5*z^3 +
2626225/17496*y^5*z^2 - 43378/2187*y^5*z - 29413/2187*y^5 + 44425/69984*y^4*z^18
+ 1526545/69984*y^4*z^17 + 1489043/23328*y^4*z^16 + 661867/17496*y^4*z^15 -
1622543/34992*y^4*z^14 + 545069/34992*y^4*z^13 + 112471/8748*y^4*z^12 -
3012031/5832*y^4*z^11 - 30731213/34992*y^4*z^10 + 7525711/69984*y^4*z^9 +
103064435/69984*y^4*z^8 + 82980083/69984*y^4*z^7 - 28598633/69984*y^4*z^6 -
12665851/11664*y^4*z^5 - 234601/648*y^4*z^4 + 18247517/69984*y^4*z^3 +
674599/4374*y^4*z^2 - 54313/2187*y^4*z - 47891/4374*y^4 - 44425/34992*y^3*z^19 -
5365/11664*y^3*z^18 - 614141/23328*y^3*z^17 - 9605029/69984*y^3*z^16 -
10602997/69984*y^3*z^15 + 20587543/69984*y^3*z^14 + 27215087/34992*y^3*z^13 +
6637291/34992*y^3*z^12 - 82735585/69984*y^3*z^11 - 89174777/69984*y^3*z^10 +
9389951/17496*y^3*z^9 + 42424103/23328*y^3*z^8 + 24933521/34992*y^3*z^7 -
37215199/34992*y^3*z^6 - 4967245/4374*y^3*z^5 - 343051/17496*y^3*z^4 +
36291181/69984*y^3*z^3 + 3891055/17496*y^3*z^2 - 699161/17496*y^3*z -
1344919/34992*y^3 + 222125/34992*y^2*z^18 + 2525/144*y^2*z^17 -
376315/34992*y^2*z^16 - 43247/432*y^2*z^15 - 4391987/34992*y^2*z^14 -
718225/34992*y^2*z^13 + 2475415/34992*y^2*z^12 + 836129/4374*y^2*z^11 +
2579827/5832*y^2*z^10 + 2744041/8748*y^2*z^9 - 5028437/11664*y^2*z^8 -
6578429/8748*y^2*z^7 - 521381/3888*y^2*z^6 + 6628477/17496*y^2*z^5 +
7556995/34992*y^2*z^4 - 1000777/34992*y^2*z^3 - 1187455/34992*y^2*z^2 +
44425/11664*y*z^19 + 225985/34992*y*z^18 - 95083/5832*y*z^17 -
1486403/34992*y*z^16 + 76165/4374*y*z^15 + 1918787/34992*y*z^14 -
918205/5832*y*z^13 - 666692/2187*y*z^12 + 2133181/11664*y*z^11 +
27617131/34992*y*z^10 + 7490587/17496*y*z^9 - 19862143/34992*y*z^8 -
351121/432*y*z^7 - 256829/2187*y*z^6 + 30862/81*y*z^5 + 1904011/8748*y*z^4 -
1005151/34992*y*z^3 - 1218073/34992*y*z^2 + 2*y*z - y - 44425/69984*z^21 -
193795/69984*z^20 - 637177/34992*z^19 - 440941/8748*z^18 - 1109995/69984*z^17 +
13565093/69984*z^16 + 22302653/69984*z^15 - 7954033/69984*z^14 -
56030677/69984*z^13 - 21454273/34992*z^12 + 10510465/17496*z^11 +
30624971/23328*z^10 + 326326/729*z^9 - 7179065/8748*z^8 - 63266461/69984*z^7 -
335959/4374*z^6 + 27268337/69984*z^5 + 7320799/34992*z^4 - 607117/34992*z^3 -
```

```
1309927/34992*z^2 - 2*z + 1, x*z^21 + 9*x*z^20 + 42*x*z^19 + 117*x*z^18 +
195*x*z^17 + 123*x*z^16 - 232*x*z^15 - 678*x*z^14 - 660*x*z^13 + 45*x*z^12 +
783*x*z^11 + 744*x*z^10 + 115*x*z^9 - 300*x*z^8 - 240*x*z^7 - 64*x*z^6 +
y^14*z^12 - 6*y^14*z^11 - 48*y^14*z^10 - 136*y^14*z^9 - 219*y^14*z^8 -
210*y^14*z^7 - 57*y^14*z^6 + 198*y^14*z^5 + 408*y^14*z^4 + 422*y^14*z^3 +
264*y^14*z^2 + 96*y^14*z + 16*y^14 - 10*y^13*z^12 - 54*y^13*z^11 - 144*y^13*z^10
- 256*y^13*z^9 - 342*y^13*z^8 - 336*y^13*z^7 - 168*y^13*z^6 + 132*y^13*z^5 +
384*y^13*z^4 + 418*y^13*z^3 + 264*y^13*z^2 + 96*y^13*z + 16*y^13 - 6*y^12*z^13 -
26*y^12*z^12 - 54*y^12*z^11 - 84*y^12*z^10 - 150*y^12*z^9 - 276*y^12*z^8 -
366*y^12*z^7 - 254*y^12*z^6 + 66*y^12*z^5 + 360*y^12*z^4 + 414*y^12*z^3 +
264*y^12*z^2 + 96*y^12*z + 16*y^12 + 2*y^11*z^14 - 36*y^11*z^13 - 235*y^11*z^12
- 590*y^11*z^11 - 750*y^11*z^10 - 308*y^11*z^9 + 711*y^11*z^8 + 1770*y^11*z^7 +
2019*y^11*z^6 + 922*y^11*z^5 - 792*y^11*z^4 - 1642*y^11*z^3 - 1240*y^11*z^2 -
480*y^11*z - 80*y^11 - y^10*z^15 - 44*y^10*z^14 - 222*y^10*z^13 - 523*y^10*z^12
- 743*y^10*z^11 - 684*y^10*z^10 - 223*y^10*z^9 + 795*y^10*z^8 + 2058*y^10*z^7 +
2449*y^10*z^6 + 1232*y^10*z^5 - 672*y^10*z^4 - 1622*y^10*z^3 - 1240*y^10*z^2 -
480*y^10*z - 80*y^10 - 20*y^9*z^15 - 76*y^9*z^14 - 90*y^9*z^13 - 18*y^9*z^12 -
138*y^9*z^11 - 684*y^9*z^10 - 1018*y^9*z^9 - 88*y^9*z^8 + 1806*y^9*z^7 +
2738*y^9*z^6 + 1542*y^9*z^5 - 552*y^9*z^4 - 1602*y^9*z^3 - 1240*y^9*z^2 -
480*y^9*z - 80*y^9 + y^8*z^16 - 64*y^8*z^15 - 373*y^8*z^14 - 856*y^8*z^13 -
877*y^8*z^12 + 88*y^8*z^11 + 1680*y^8*z^10 + 2968*y^8*z^9 + 2781*y^8*z^8 +
314*y^8*z^7 - 2900*y^8*z^6 - 3648*y^8*z^5 - 1472*y^8*z^4 + 718*y^8*z^3 +
1080*y^8*z^2 + 480*y^8*z + 80*y^8 - 2*y^7*z^17 - 64*y^7*z^16 - 308*y^7*z^15 -
630*y^7*z^14 - 622*y^7*z^13 - 156*y^7*z^12 + 558*y^7*z^11 + 1686*y^7*z^10 +
2916*y^7*z^9 + 2318*y^7*z^8 - 1112*y^7*z^7 - 4496*y^7*z^6 - 4116*y^7*z^5 -
656*y^7*z^4 + 1916*y^7*z^3 + 1872*y^7*z^2 + 768*y^7*z + 128*y^7 + y^6*z^18 -
16*y^6*z^17 - 38*y^6*z^16 + 132*y^6*z^15 + 541*y^6*z^14 + 414*y^6*z^13 -
902*y^6*z^12 - 1874*y^6*z^11 - 26*y^6*z^10 + 3544*y^6*z^9 + 4061*y^6*z^8 -
270*y^6*z^7 - 4787*y^6*z^6 - 4584*y^6*z^5 - 848*y^6*z^4 + 1884*y^6*z^3 +
1872*y^6*z^2 + 768*y^6*z + 128*y^6 - 44*y^5*z^17 - 232*y^5*z^16 - 444*y^5*z^15 -
176*y^5*z^14 + 748*y^5*z^13 + 1712*y^5*z^12 + 2096*y^5*z^11 + 1356*y^5*z^10 -
1244*y^5*z^9 - 4336*y^5*z^8 - 4180*y^5*z^7 - 324*y^5*z^6 + 2748*y^5*z^5 +
2320*y^5*z^4 + 512*y^5*z^3 - 288*y^5*z^2 - 192*y^5*z - 32*y^5 - y^4*z^19 -
36*y^4*z^18 - 158*y^4*z^17 - 265*y^4*z^16 - 160*y^4*z^15 - 32*y^4*z^14 -
25*y^4*z^13 + 770*y^4*z^12 + 2683*y^4*z^11 + 3037*y^4*z^10 - 565*y^4*z^9 -
5025*y^4*z^8 - 4922*y^4*z^7 - 497*y^4*z^6 + 2820*y^4*z^5 + 2368*y^4*z^4 +
520*y^4*z^3 - 288*y^4*z^2 - 192*y^4*z - 32*y^4 + 2*y^3*z^20 + 4*y^3*z^19 +
45*y^3*z^18 + 286*y^3*z^17 + 642*y^3*z^16 + 202*y^3*z^15 - 1587*y^3*z^14 -
2686*y^3*z^13 - 240*y^3*z^12 + 4030*y^3*z^11 + 4238*y^3*z^10 - 1108*y^3*z^9 -
5533*y^3*z^8 - 3486*y^3*z^7 + 1823*y^3*z^6 + 3882*y^3*z^5 + 1768*y^3*z^4 -
642*y^3*z^3 - 1080*y^3*z^2 - 480*y^3*z - 80*y^3 - 10*y^2*z^19 - 44*y^2*z^18 -
40*y^2*z^17 + 148*y^2*z^16 + 460*y^2*z^15 + 544*y^2*z^14 + 254*y^2*z^13 -
318*y^2*z^12 - 1258*y^2*z^11 - 2018*y^2*z^10 - 1108*y^2*z^9 + 1306*y^2*z^8 +
2552*y^2*z^7 + 1334*y^2*z^6 - 418*y^2*z^5 - 872*y^2*z^4 - 432*y^2*z^3 -
80*y^2*z^2 - 6*y*z^20 - 20*y*z^19 + 2*y*z^18 + 94*y*z^17 + 106*y*z^16 -
42*y*z^15 + 112*y*z^14 + 784*y*z^13 + 746*y*z^12 - 1040*y*z^11 - 2768*y*z^10 -
1716*y*z^9 + 1318*y*z^8 + 2770*y*z^7 + 1442*y*z^6 - 398*y*z^5 - 872*y*z^4 -
432*y*z^3 - 80*y*z^2 + z^22 + 6*z^21 + 37*z^20 + 132*z^19 + 191*z^18 - 156*z^17
```

```
- 927*z^16 - 972*z^15 + 822*z^14 + 2938*z^13 + 2097*z^12 - 1862*z^11 - 4500*z^10
- 2504*z^9 + 1617*z^8 + 3228*z^7 + 1614*z^6 - 378*z^5 - 872*z^4 - 432*z^3 -
80*z^2, y^15 + 5*y^12*z^2 - 6*y^12 + 10*y^9*z^4 - 20*y^9*z^2 + 10*y^9 -
3*y^8*z^3 + 3*y^8 + 10*y^6*z^6 - 30*y^6*z^4 + 30*y^6*z^2 - 10*y^6 - 3*y^4*z^6 +
6*y^4*z^3 - 3*y^4 + 5*y^3*z^8 - 20*y^3*z^6 + 30*y^3*z^4 - 20*y^3*z^2 + 5*y^3 +
z^10 - z^9 - 5*z^8 + 13*z^6 - 10*z^4 - 3*z^3 + 5*z^2]

Base gröbner grevlex de J
: [y^6 + x*y^4 + 2*y^3*z^2 + x*z^3 + z^4 - 2*y^3 - 2*z^2 - x + 1, x^2*y^3 - y^4
+ x^2*z^2 - z^3 - x^2 + 1, x^3 + y^3 + z^2 - 1]
```

L'influence de l'ordre monomial est flagrante sur ces exemples.

0.0.9 Exo 11 : implicitation de courbes et surfaces paramétrées

[32]:
```
RR.<t,u,x,y,z>=PolynomialRing(QQ,'t,u,x,y,z',order='lex')
G1=x-t-u
G2=y-t**2-2*t*u
G3=z-t**3-3*t**2*u
I=RR.ideal(G1,G2,G3)
J=I.groebner_basis()
print(J)
print()
print("equation :", J[-1])

R3.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='lex')
P=R3(J[-1])
implicit_plot3d(P,(-5,5),(-5,5),(-2,5),color='blue')
```

```
[t + u - x, u^2 - x^2 + y, u*x^2 - u*y - x^3 + 3/2*x*y - 1/2*z, u*x*y - u*z -
x^2*y - x*z + 2*y^2, u*x*z - u*y^2 + x^2*z - 1/2*x*y^2 - 1/2*y*z, u*y^3 - u*z^2
- 2*x^2*y*z + 1/2*x*y^3 - x*z^2 + 5/2*y^2*z, x^3*z - 3/4*x^2*y^2 - 3/2*x*y*z +
y^3 + 1/4*z^2]

equation : x^3*z - 3/4*x^2*y^2 - 3/2*x*y*z + y^3 + 1/4*z^2
```

[32]: Graphics3d Object

[33]:
```
RR.<u,v,x,y,z>=PolynomialRing(QQ,'u,v,x,y,z',order='lex')
G1=3*u + 3*u*v**2 - u**3-x
G2=3*v + 3*u**2*v - v**3-y
G3=3*u**2 - 3*v**2-z
I=RR.ideal(G1,G2,G3)
J=I.groebner_basis()
print("équation : ",J[-1])
R3.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='lex')
P=R3(J[-1])
```

```
implicit_plot3d(P,(-5,5),(-5,5),(-2,5),color='blue')
```

```
équation :  x^6 - 3*x^4*y^2 + 5/9*x^4*z^3 + 6*x^4*z^2 - 3*x^4*z + 3*x^2*y^4 +
26/9*x^2*y^2*z^3 + 6*x^2*y^2*z + 16/243*x^2*z^6 + 16/9*x^2*z^5 + 80/9*x^2*z^4 -
16*x^2*z^3 - y^6 + 5/9*y^4*z^3 - 6*y^4*z^2 - 3*y^4*z - 16/243*y^2*z^6 +
16/9*y^2*z^5 - 80/9*y^2*z^4 - 16*y^2*z^3 - 64/19683*z^9 + 128/243*z^7 - 64/3*z^5
```

[33]: Graphics3d Object

[34]:
```
### Paramétrisation rationnelle : on ajoute une equation et une variable
### G4=1-w*(produit des denominateurs) pour s'assurer que les denominateurs ne
↪s'annulent pas !!

RR.<u,v,w,x,y,z>=PolynomialRing(QQ,'u,v,w,x,y,z',order='lex')
G1=x*v-u**2
G2=y*u - v**2
G3=z-u**3
G4=1-w*u*v  #

I=RR.ideal(G1,G2,G3,G4)
J=I.groebner_basis()
print("équation : ",J[-1])

R3.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='lex')
P=R3(J[-1])
implicit_plot3d(P,(-5,5),(-5,5),(-2,5),color='blue')
```

```
équation :  x^2*y - z
```

[34]: Graphics3d Object

[35]:
```
RR.<t,x,y,z>=PolynomialRing(QQ,'t,x,y,z',order='lex')
G1=x-t**4
G2=y-t**3
G3=z-t**2

I=RR.ideal(G1,G2,G3)
J=I.groebner_basis()
print(J)


R3.<x,y,z>=PolynomialRing(QQ,'x,y,z',order='lex')
```

```
[t^2 - z, t*y - z^2, t*z - y, x - z^2, y^2 - z^3]
```

[36]:
```
P=R3(J[-2])
Q=R3(J[-1])
```

```
var('x,y,z')
G1=implicit_plot3d(P,(x,-5,5),(y,-5,5),(z,-2,2),color='blue')
G2=implicit_plot3d(Q,(x,-5,5),(y,-5,5),(z,-2,2),color='red')
G1+G2
```

[36]: Graphics3d Object

[37]:
```
var('t')
parametric_plot3d((t**4,t**3,t**2),(t,-1,1))
```

[37]: Graphics3d Object