NB.06.F4

July 23, 2024

1 Proposition

Any cubic that has a line t in the eigenscheme is the limit of a family of cubics whose general member has a 0-dimensional eigenscheme with an aligned triple.

[1]:
```
load("basic_functions.sage")
```

1.1 CASE 1: CUBICS WITH A LINE OF EIGENPOINTS (NOT TANGENT TO THE ISOTROPIC CONIC)

1.2 A cubic of the form $r^2 l$ is the limit of a family of cubics with three

1.3 aligned points (depending on one parameter u1).

We choose three points P_1, P_2, and P_3 aligned on the line $z = 0$

[2]:
```
P1 = vector(S, (1, 0, 0))
P2 = vector(S, (0, 1, 0))
P3 = P1 + P2
```

We construct the space of all the cubics that have P_1, P_2, P_3 among the eigenpoints.

[3]:
```
M = condition_matrix([P1, P2, P3], S, standard="all")
```

We extract 6 linearly independent rows from M:

[4]:
```
M1 = M.matrix_from_rows([0, 1, 3, 5, 6, 7])
assert(M1.rank() == 6)
```

We construct 4 cubics that have P_1, P_2, P_3 among the eigenpoints.

[5]:
```
M2 = M1.stack(vector(S, (0, 2, 0, 4, 0, 0, 0, 6, -7, 3)))
M2 = M2.stack(vector(S, (1, 1, 0, 2, 0, 0, 0, 8, 5, 3)))
M2 = M2.stack(vector(S, (4, 7, 0, 1, 0, 0, 0, 11, -4, 2)))
M2 = M2.stack(vector(S, mon))
cb1 = M2.det()

M2 = M1.stack(vector(S, (4, 1, 0, -2, 0, 0, 0, 3, -1, 5)))
M2 = M2.stack(vector(S, (2, -3, 0, 1, 0, 0, 0, 2, 1, 4)))
M2 = M2.stack(vector(S, (1, 2, 0, -1, 0, 0, 0, 9, -1, -2)))
M2 = M2.stack(vector(S, mon))
```

```
cb2 = M2.det()

M2 = M1.stack(vector(S, (7, 7, 0, 1, 0, 0, 0, 5, -2, -5)))
M2 = M2.stack(vector(S, (9, 7, 0, 2, 0, 0, 0, 1, -1, 3)))
M2 = M2.stack(vector(S, (5, 3, 0, 8, 0, 0, 0, 1, 11, 3)))
M2 = M2.stack(vector(S, mon))
cb3 = M2.det()

M2 = M1.stack(vector(S, (3, 5, 0, 2, 0, 0, 0, -1, 6, -1)))
M2 = M2.stack(vector(S, (2, 3, 0, 11, 0, 0, 0, -1, 1, 2)))
M2 = M2.stack(vector(S, (1, 2, 0, 1, 0, 0, 0, -1, 1, 4)))
M2 = M2.stack(vector(S, mon))
cb4 = M2.det()
```

cb_1, cb_2, cb_3, cb_4 are linearly independent:

[6]:
```
assert(matrix(
    [
        [cb1.coefficient(mm) for mm in mon],
        [cb2.coefficient(mm) for mm in mon],
        [cb3.coefficient(mm) for mm in mon],
        [cb4.coefficient(mm) for mm in mon]
    ]
).rank() == 4)
```

We use cb_1, cb_2, cb_3, cb_4 to construct a simpler basis of the 3-dim space of all the cubics with P1, P2, P3 eigenpoints:

[7]:
```
Ma = matrix(
    [
        [cb1.coefficient(mm) for mm in mon],
        [cb2.coefficient(mm) for mm in mon],
        [cb3.coefficient(mm) for mm in mon],
        [cb4.coefficient(mm) for mm in mon]
    ]
)

Ma = Ma.echelon_form()
```

We redefine cb_1, cb_2, cb_3, cb_4:

[8]:
```
cb1 = add([Ma[0][i]*mon[i] for i in range(10)])
cb2 = add([Ma[1][i]*mon[i] for i in range(10)])
cb3 = add([Ma[2][i]*mon[i] for i in range(10)])
cb4 = add([Ma[3][i]*mon[i] for i in range(10)])
```

[9]:
```
assert(cb1 == x^3+y^3)
assert(cb2 == x*z^2)
assert(cb3 == y*z^2)
```

```
assert(cb4 == z^3)
```

Now cb_1, cb_2, cb_3, cb_4 is a good basis. From it, we construct the generic cubic that has P_1, P_2, P_3 among the eigenpoints.

[10]:
```
cb = u1*cb1+v1*cb2+w1*cb3+l1*cb4
```

If $u_1 = 0$, cb is the double line $z = 0$ and a generic line:

[11]:
```
assert(cb.subs(u1=0) == z^2*(v1*x+w1*y+l1*z))
```

We extract the eigenpoints of cb (and we erase P_1, P_2, P_3 and we assume $u_1 \neq 0$).

[12]:
```
Jc = S.ideal(
    matrix(
        [
            [x, y, z],
            [cb.derivative(x), cb.derivative(y), cb.derivative(z)]
        ]
    ).minors(2)
)
```

[13]:
```
Jc = Jc.saturation(S.ideal(matrix([(x, y, z), P1]).minors(2)))[0]
Jc = Jc.saturation(S.ideal(matrix([(x, y, z), P2]).minors(2)))[0]
Jc = Jc.saturation(S.ideal(matrix([(x, y, z), P3]).minors(2)))[0]
Jc = Jc.saturation(u1)[0]
```

[14]:
```
assert(Jc.is_prime())
```

Jc is generated by g_1, g_2 below:

[15]:
```
g1 = 3*y^2*u1 - 2*x*y*v1 - 2*y^2*w1 + z^2*w1 - 3*y*z*l1
g2 = 3*x^2*u1 - 2*x^2*v1 + z^2*v1 - 2*x*y*w1 - 3*x*z*l1
```

[16]:
```
assert(Jc == S.ideal(g1, g2))
```

The zeros of Jc should be 4 points which are (in general) distinct and with no other collinearities).

1.4 CASE 2: CUBICS WITH EIGENPOINTS ON A LINE TANGENT TO THE ISOTROPIC CONIC

We choose three points P_1, P_2, and P_3 aligned on the line $x + i\,y = 0$

[17]:
```
P1 = vector(S, (1, ii, 0))
P2 = vector(S, (0, 0, 1))
P3 = P1 + P2
```

We construct the space of all the cubics that have P_1, P_2, P_3 among the eigenpoints.

[18]:
```
M = condition_matrix([P1, P2, P3], S, standard="all")
```

We extract 5 linearly independent rows:

[19]:
```
M1 = M.matrix_from_rows([0, 1, 4, 5, 7])
```

[20]:
```
assert(M1.rank() == 5)
```

We construct 5 cubics which are a basis for the space of allo the cubics with P_1, P_2, P_3 as eigenpoints.

[21]:
```
M2 = M1.stack(vector(S, (0, 0, 1, 2, 0, -1, 3, 0, 0, 1)))
M2 = M2.stack(vector(S, (0, 0, -2, 1, 0, 1, 1, 0, 0, 2)))
M2 = M2.stack(vector(S, (0, 0, 4, -3, 0, 1, 1, 0, 0, 3)))
M2 = M2.stack(vector(S, (0, 0, 2, 1, 0, -3, 2, 0, 0, 1)))
M2 = M2.stack(vector(S, mon))
cb1 = M2.det()

M2 = M1.stack(vector(S, (0, 0, 6, 3, 0, 5, 2, 0, 0, -1)))
M2 = M2.stack(vector(S, (0, 0, -2, 1, 0, -3, -1, 0, 0, 2)))
M2 = M2.stack(vector(S, (0, 0, 4, 6, 0, 2, 5, 0, 0, 3)))
M2 = M2.stack(vector(S, (0, 0, 1, 7, 0, -5, 4, 0, 0, 4)))
M2 = M2.stack(vector(S, mon))
cb2 = M2.det()

M2 = M1.stack(vector(S, (0, 0, 5, 3, 0, 3, 2, 0, 0, -3)))
M2 = M2.stack(vector(S, (0, 0, -3, 1, 0, -6, -1, 0, 0, 2)))
M2 = M2.stack(vector(S, (0, 0, 5, 6, 0, 5, 5, 0, 0, 6)))
M2 = M2.stack(vector(S, (0, 0, 2, 7, 0, -5, 7, 0, 0, 5)))
M2 = M2.stack(vector(S, mon))
cb3 = M2.det()

M2 = M1.stack(vector(S, (0, 0, 1, 3, 0, 3, 2, 0, 0, -2)))
M2 = M2.stack(vector(S, (0, 0, -3, 1, 0, -7, -4, 0, 0, 5)))
M2 = M2.stack(vector(S, (0, 0, 5, 3, 0, 5, 8, 0, 0, 2)))
M2 = M2.stack(vector(S, (0, 0, 2, 1, 0, -5, 7, 0, 0, 3)))
M2 = M2.stack(vector(S, mon))
cb4 = M2.det()

M2 = M1.stack(vector(S, (0, 0, 7, 1, 0, 3, 2, 0, 0, -2)))
M2 = M2.stack(vector(S, (0, 0, -3, 1, 0, -5, -2, 0, 0, 5)))
M2 = M2.stack(vector(S, (0, 0, 1, 3, 0, 5, 1, 0, 0, 7)))
M2 = M2.stack(vector(S, (0, 0, 3, 2, 0, -1, 5, 0, 0, 3)))
M2 = M2.stack(vector(S, mon))
cb5 = M2.det()
```

[22]:
```
Ma = matrix(
    [
        [cb1.coefficient(mm) for mm in mon],
        [cb2.coefficient(mm) for mm in mon],
        [cb3.coefficient(mm) for mm in mon],
```

```
        [cb4.coefficient(mm) for mm in mon],
        [cb5.coefficient(mm) for mm in mon]
    ]
)
```

[23]:
```
assert(Ma.rank() == 5)
```

$cb_1, \dots, cb_5$ is a basis. We want a better basis. We choose:

[24]:
```
ccb1 = (x+ii*y)^2*x
ccb2 = (x+ii*y)^2*y
ccb3 = (x+ii*y)^2*z
ccb4 = z*(x^2+y^2+2/3*z^2)
ccb5 = x^3 + (-ii)*y^3 + z^3
```

We verify that also this is a basis:

[25]:
```
MMa = matrix(
    [
        [ccb1.coefficient(mm) for mm in mon],
        [ccb2.coefficient(mm) for mm in mon],
        [ccb3.coefficient(mm) for mm in mon],
        [ccb4.coefficient(mm) for mm in mon],
        [ccb5.coefficient(mm) for mm in mon]
    ]
)
```

[26]:
```
assert(Ma.echelon_form() == MMa.echelon_form())
```

We define the generic cubic linear combinations of $ccb_1, \dots, ccb_5$

[27]:
```
cb = u1*ccb1+u2*ccb2+v1*ccb3+v2*ccb4+w1*ccb5
```

For $w_1 = 0$, we get that cb is of the form $t^2\ell + \lambda C(r)$, so it is the generic cubic with a line of eigenpints tangent to the isotropic conic; for $w_1 \neq 0$, the ideal of the remaining 4 eigenpoints is given by:

[28]:
```
Jc = S.ideal(
    matrix(
        [
            [x, y, z],
            [cb.derivative(x), cb.derivative(y), cb.derivative(z)]
        ]
    ).minors(2)
)
```

[29]:
```
Jc = Jc.saturation(S.ideal(matrix([(x, y, z), P1]).minors(2)))[0]
Jc = Jc.saturation(S.ideal(matrix([(x, y, z), P2]).minors(2)))[0]
Jc = Jc.saturation(S.ideal(matrix([(x, y, z), P3]).minors(2)))[0]
```

```
Jc = Jc.saturation(w1)[0]
```

[30]:
```
Jcrad = Jc.radical()
```

The ideal Jcrad gives the 4 eigenpoints.