NB.07.F7

July 23, 2024

1 Configuration (C_8)

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

1.1 Construction of a (C_8) configuration:

We define 4 generic points (P_1, P_2, P_4, P_7) in the plane and we define P_3, P_5, P_6 in such a way that the seven points are in a (C_8) configuration with the alignments:

$(1, 2, 3), (1, 4, 5), (1, 6, 7), (2, 4, 6), (2, 5, 7), (3, 4, 7) $

[138]:
```
P1 = vector((A1, B1, C1))
P2 = vector((A2, B2, C2))
P4 = vector((A4, B4, C4))
P7 = vector((A7, B7, C7))
P3 = intersection_lines(P1, P2, P7, P4)
P5 = intersection_lines(P1, P4, P2, P7)
P6 = intersection_lines(P1, P7, P2, P4)
```

The points P_3, P_5, P_6 are always defined (i.e. it is not possible that their coordinates are all zero), as follows from the block below:

[139]:
```
J1 = S.ideal(list(P3))
J1 = J1.saturation(S.ideal(matrix([P2, P7]).minors(2)))[0]
J1 = J1.saturation(S.ideal(matrix([P4, P7]).minors(2)))[0]
J1 = J1.saturation(matrix([P2, P4, P7]).det())[0]
J1 = J1.saturation(S.ideal(matrix([P1, P2]).minors(2)))[0]
assert(J1 == S.ideal(S(1)))

J1 = S.ideal(list(P5))
J1 = J1.saturation(S.ideal(matrix([P2, P4]).minors(2)))[0]
J1 = J1.saturation(S.ideal(matrix([P4, P7]).minors(2)))[0]
J1 = J1.saturation(matrix([P2, P4, P7]).det())[0]
J1 = J1.saturation(S.ideal(matrix([P1, P4]).minors(2)))[0]
assert(J1 == S.ideal(1))

J1 = S.ideal(list(P6))
J1 = J1.saturation(S.ideal(matrix([P2, P7]).minors(2)))[0]
J1 = J1.saturation(S.ideal(matrix([P4, P7]).minors(2)))[0]
```

```
J1 = J1.saturation(matrix([P2, P4, P7]).det())[0]
J1 = J1.saturation(S.ideal(matrix([P1, P7]).minors(2)))[0]
assert(J1 == S.ideal(1))
```

If configuration (C_8) is given by eigenpoints, we must have $e_1 = e_2 = e_3 = 0$, where:

$e_1 = \delta_1(P_5, P_1, P_2), e_2 = \delta_1(P_6, P_1, P_2), e_3 = \delta_1(P_3, P_1, P_4)$

[140]:
```
e1 = delta1(P5, P1, P2)
e2 = delta1(P6, P1, P2)
e3 = delta1(P3, P1, P4)
```

We have:

$$e_1 = (s_{12}s_{47} - s_{17}s_{24}) \cdot \det \begin{pmatrix} P_1 \\ P_2 \\ P_7 \end{pmatrix} \cdot \det \begin{pmatrix} P_1 \\ P_2 \\ P_4 \end{pmatrix}$$

and similarly for e_2 and e_3:

[141]:
```
assert(
    e1 ==
    matrix([P1, P2, P7]).det()*matrix([P1, P2, P4]).det()
    * (scalar_product(P1, P2)*scalar_product(P4, P7)-scalar_product(P1,␣
 ↪P7)*scalar_product(P2, P4))
)

assert(
    e2 ==
    matrix([P1, P2, P7]).det()*matrix([P1, P2, P4]).det()
    * (scalar_product(P1, P2)*scalar_product(P4, P7)-scalar_product(P1,␣
 ↪P4)*scalar_product(P2, P7))
)

assert(
    e3 ==
    -matrix([P1, P4, P7]).det()*matrix([P1, P2, P4]).det()
    * (scalar_product(P1, P4)*scalar_product(P2, P7)-scalar_product(P1,␣
 ↪P7)*scalar_product(P2, P4))
)
```

Hence we redefine e_1, e_2, e_3 as

- $e_1 = s_{12}s_{47} - s_{17}s_{24}$,
- $e_2 = s_{12}s_{47} - s_{14}s_{27}$, and
- $e_3 = s_{14}s_{27} - s_{17}s_{24}$

[142]:
```
e1 = scalar_product(P1, P2)*scalar_product(P4, P7)-scalar_product(P1,␣
 ↪P7)*scalar_product(P2, P4)
```

```
e2 = scalar_product(P1, P2)*scalar_product(P4, P7)-scalar_product(P1,␣
 ↪P4)*scalar_product(P2, P7)
e3 = -scalar_product(P1, P4)*scalar_product(P2, P7)+scalar_product(P1,␣
 ↪P7)*scalar_product(P2, P4)
```

We have: e1-e2+e3 == 0, hence e3 is not necessary

[143]:
```
assert(e1-e2+e3 == 0)
```

We solve the system $e_1 = 0, e_2 = 0$, linear in A_4, B_4, C_4 and we get the solution that we denote by ss6. We verify that ss6 is the solution of the two equations $e_1 = 0, e_2 = 0$

[144]:
```
M1 = matrix(
    [
        [e1.coefficient(A4), e1.coefficient(B4), e1.coefficient(C4)],
        [e2.coefficient(A4), e2.coefficient(B4), e2.coefficient(C4)]
    ]
)

minM1 = M1.minors(2)

ss6 = {A4: minM1[2], B4: -minM1[1], C4: minM1[0]}

assert(e1.subs(ss6) == 0)
assert(e2.subs(ss6) == 0)
```

The solution ss6 is not unique (degenerate case) iff $(s_{12} = 0, s_{17} = 0)$ or $(s_{27} = 0, s_{17} = 0)$ or $(s_{12} = 0, s_{27} = 0)$ as we can obtain from the computations below

[145]:
```
pd = S.ideal(minM1).saturation(det(matrix([P1, P2, P7])))[0].radical().
 ↪primary_decomposition()
assert(len(pd) == 3)

assert(pd[0] == S.ideal(scalar_product(P1, P2), scalar_product(P1, P7)))
assert(pd[1] == S.ideal(scalar_product(P2, P7), scalar_product(P1, P7)))
assert(pd[2] == S.ideal(scalar_product(P1, P2), scalar_product(P2, P7)))
```

From a previous computation (NB.07.F6) we know that each of these 3 conditions gives $s_{12} = 0, s_{17} = 0, s_{27} = 0$ and this condition is not possible

Now we redefine the points P_4, P_3, P_5, P_6 using the substitution ss6

[146]:
```
P4 = P4.subs(ss6)
P3 = P3.subs(ss6)
P5 = P5.subs(ss6)
P6 = P6.subs(ss6)
```

And we get that 4 δ_2 conditions are zero:

[147]:
```
assert(delta2(P1, P2, P3, P4, P5) == 0)
assert(delta2(P4, P1, P5, P2, P6) == 0)
assert(delta2(P7, P3, P4, P2, P5) == 0)
assert(delta2(P2, P1, P3, P4, P6) == 0)
```

We have that the lines $P_1 \vee P_2$ and $P_3 \vee P_4$ are orthogonal,

the lines $P_1 \vee P_6$ and $P_2 \vee P_4$ are orthogonal and

the lines $P_1 \vee P_4$ and $P_2 \vee P_5$ are orthogonal:

[148]:
```
assert(scalar_product(wedge_product(P1, P2), wedge_product(P3, P4)) == S(0))
assert(scalar_product(wedge_product(P1, P6), wedge_product(P2, P4)) == S(0))
assert(scalar_product(wedge_product(P1, P4), wedge_product(P2, P5)) == S(0))
```

Moreover,it holds:

$P_4 = (P_1 \times P_2)s_{17}s_{27} - s_{12}(P_1 \times P_7)s_{27} + s_{12}s_{17}(P_2 \times P_7)$

Indeed:

[149]:
```
Q4 = (
    wedge_product(P1, P2)*scalar_product(P1, P7)*scalar_product(P2, P7)
    - scalar_product(P1, P2)*wedge_product(P1, P7)*scalar_product(P2, P7)
    + scalar_product(P1, P2)*scalar_product(P1, P7)*wedge_product(P2, P7)
)

assert(matrix([P4, Q4]).minors(2) == [0, 0, 0])
```

1.1.1 CONCLUSION:

If we start from configuration (C_8) of eigenpoints, then we have the above orthogonalities among the lines joining the points and the point P_4 is defined by the above formula.

1.2 Conversely:

Given: Suppose P_1, P_2, P_7 are three arbitrary points of the plane and define P_4 as above, by the formula: $P_4 = (P_1 \times P_2)s_{17}s_{27} - s_{12}(P_1 \times P_7)s_{27} + s_{12}s_{17}(P_2 \times P_7)$ then define $P_3 = (P_1 \vee P_2) \cap (P_4 \vee P_7)$, $P_5 = (P_1 \vee P_4) \cap (P_2 \vee P_7)$, $P_6 = (P_1 \vee P_7) \cap (P_2 \vee P_4)$ then the points $P_1, \ldots, P_7$ are in configuration (C_8) and are eigenpoints of a suitable cubic.

It is enough to verify that the rank of the matrix $\Phi(P_1, P_2, P_3, P_4, P_5, P_6, P_7)$ is ≤ 9.

We redefine the points according to the above constrains and we split the problem in two cases: $P_1 = (1 : 0 : 0)$ and $P_1 = (1 : i : 0)$.

[157]:
```
p1 = vector(S, (1, 0, 0))

p2 = vector(S, (A2, B2, C2))
p7 = vector(S, (A7, B7, C7))
p4 = (
    wedge_product(p1, p2)*scalar_product(p1, p7)*scalar_product(p2, p7)
```

```
    - scalar_product(p1, p2)*wedge_product(p1, p7)*scalar_product(p2, p7)
    + scalar_product(p1, p2)*scalar_product(p1, p7)*wedge_product(p2, p7)
)
p3 = intersection_lines(p1, p2, p4, p7)
p5 = intersection_lines(p1, p4, p2, p7)
p6 = intersection_lines(p1, p7, p2, p4)
```

We have: the matrix $\Phi([p_1, p_2, p_3, p_4, p_5, p_6, p_7])$ has rank 9:

[158]:
```
assert(condition_matrix([p1, p2, p3, p4, p5, p6, p7], S, standard="all").rank()␣
↪== 9)
```

Similarly, we can verify the result for the case in which p_1 is $(1 : ii : 0)$, but in this case , to speed up the computation, it is better to make some simplifications.

We redefine the points.

[152]:
```
p1 = vector(S, (1, ii, 0))

p2 = vector(S, (A2, B2, C2))
p7 = vector(S, (A7, B7, C7))
p4 = (
    wedge_product(p1, p2)*scalar_product(p1, p7)*scalar_product(p2, p7)
    - scalar_product(p1, p2)*wedge_product(p1, p7)*scalar_product(p2, p7)
    + scalar_product(p1, p2)*scalar_product(p1, p7)*wedge_product(p2, p7)
)
p3 = intersection_lines(p1, p2, p4, p7)
p5 = intersection_lines(p1, p4, p2, p7)
p6 = intersection_lines(p1, p7, p2, p4)
```

The coordinates of these points are big, but we can simplify some of them:

[153]:
```
gg = gcd(list(p3))
p3 = vector(S, [pp.quo_rem(gg)[0] for pp in p3])

gg = gcd(list(p5))
p5 = vector(S, [pp.quo_rem(gg)[0] for pp in p5])

gg = gcd(list(p6))
p6 = vector(S, [pp.quo_rem(gg)[0] for pp in p6])
```

Now we can verify that in general matrix $\Phi([p_1, p_2, p_3, p_4, p_5, p_6, p_7])$ has rank 9.

This computation requires about 10 minutes.

[154]:
```
ttA = cputime()
assert(condition_matrix([p1, p2, p3, p4, p5, p6, p7], S, standard="all").rank()␣
↪== 9)
print(cputime()-ttA)
```

```
601.2928419999998
```

1.2.1 CONCLUSION:

If we fix three points P_1, P_2, P_7 in an arbitrary way and we define

$P_4 = (P_1 \times P_2)s_{17}s_{27} - s_{12}(P_1 \times P_7)s_{27} + s_{12}s_{17}(P_2 \times P_7)$

then we define

$P_3 = (P_1 \vee P_2) \cap (P_4 \vee P_7)$

$P_5 = (P_1 \vee P_4) \cap (P_2 \vee P_7)$

$P_6 = (P_1 \vee P_7) \cap (P_2 \vee P_4)$

and we get a (C_8) configuration of eigenpoints.