

# Computational Companion to “Quasi-Symmetric Nets: A Constructive Approach to the Equimodular Elliptic Type of Kokotsakis Polyhedra” — Example 1 Helper

A. Nurmatov, M. Skopenkov, F. Rist, J. Klein, D. L. Michels  
Tested on: Mathematica 14.0

In[5441]:=

```
(*****  
====*)  
(*****  
====*)  
(*****  
====*)  
(*Quit*)  
(*All angle sets in degrees*)  
alpha1 = 105;  
beta1 = 15;  
gamma1 = 120;  
delta1 = 90;  
  
sigma1 = (alpha1 + beta1 + gamma1 + delta1) / 2;  
  
anglesDeg = {  
  {alpha1, beta1, gamma1, delta1}, (*Vertex 1*)  
  {delta1, gamma1, beta1, alpha1}, (*Vertex 2*)  
  {180 - delta1, 180 - gamma1, 180 - beta1, 180 - alpha1}, (*Vertex 3*)  
  {alpha1, beta1, gamma1, delta1} (*Vertex 4*)};  
  
(*-----*)
```

```

(*Function to compute sigma from 4 angles*)
computeSigma[{α_, β_, γ_, δ_}] := (α + β + γ + δ) / 2;

(*-----*)
(*Function to compute a,b,c,d from angles*)
computeABCD[{α_, β_, γ_, δ_}] :=
Module[{alpha = α Degree, beta = β Degree, gamma = γ Degree,
  delta = δ Degree, sigma}, sigma = computeSigma[{α, β, γ, δ} Degree];
  {Sin[alpha] / Sin[sigma - alpha], Sin[beta] / Sin[sigma - beta],
  Sin[gamma] / Sin[sigma - gamma], Sin[delta] / Sin[sigma - delta]};

(*-----*)
(*Compute all σ values*)
sigmas = computeSigma /@ anglesDeg;

(*Compute all {a,b,c,d} values*)
results = computeABCD /@ anglesDeg;

(*Optional:extract individual values*)
{a1, b1, c1, d1} = results[[1]];
{a2, b2, c2, d2} = results[[2]];
{a3, b3, c3, d3} = results[[3]];
{a4, b4, c4, d4} = results[[4]];
{σ1, σ2, σ3, σ4} = sigmas;

(*=====
====*)
(*=====
CONDITION (N.0)=====*)
(*=====
====*)
uniqueCombos = {{1, 1, 1, 1}, {1, 1, 1, -1}, {1, 1, -1, -1}, {1, 1, -1, 1},
  {1, -1, 1, 1}, {1, -1, -1, 1}, {1, -1, 1, -1}, {1, -1, -1, -1}};

checkConditionN0Degrees[{α_, β_, γ_, δ_}] :=
Module[{angles = {α, β, γ, δ}, results},
  results = Mod[uniqueCombos.angles, 360] // Chop;
  ! MemberQ[results, 0]];

conditionsN0 = checkConditionN0Degrees /@ anglesDeg;
allVerticesPass = And @@ conditionsN0;

Column[{TextCell[Style["===== CONDITION (N.0) =====",
  Darker[Green], Bold, 16], "Text"],
  If[allVerticesPass,
    Style["✓ All vertices satisfy (N.0).", Darker[Green], Bold],

```

```

Style["× Some vertices fail (N.0).", Red, Bold]]}]

(*=====
====*)
(*=====
    CONDITION (N.3)=====*)
(*=====
====*)
Ms = FullSimplify[Times@@@results];
allEqualQ = Simplify[Equal@@Ms];

Column[{TextCell[Style["===== CONDITION (N.3) =====",
    Blue, Bold, 16], "Text"], If[allEqualQ,
    Row[{Style["✓ M1 = M2 = M3 = M4 = ", Bold], Ms[[1]]},
    Style["× M_i are not all equal.", Red, Bold]]}]

(*=====
====*)
(*=====CONDITION (N.4)=====*)
(*=====
====*)
aList = results[[All, 1]];
cList = results[[All, 3]];
dList = results[[All, 4]];

rList = FullSimplify /@ (aList*dList); {r1, r2, r3, r4} = rList;
sList = FullSimplify /@ (cList*dList); {s1, s2, s3, s4} = sList;

Column[{TextCell[Style["===== CONDITION (N.4) =====",
    Darker[Blue], Bold, 16], "Text"],
    If[r1 == r2 && r3 == r4 && s1 == s4 && s2 == s3, Column[
        {Row[{Style["✓ r1 = r2 = ", Bold], r1, Style["; ✓ r3 = r4 = ", Bold], r3}],
        Row[{Style["✓ s1 = s4 = ", Bold], s1, Style["; ✓ s2 = s3 = ", Bold],
        s2}]]], Style["× Condition (N.4) fails.", Red, Bold]]
    ]}]

(*=====
====*)
(*=====
    CONDITION (N.5)=====*)
(*=====
====*)
fList = FullSimplify /@ (aList*cList); {f1, f2, f3, f4} = fList;
M1 = Ms[[1]];

```

```

m[M1_] := Piecewise[{{1 - M1, M1 < 1}, {(M1 - 1) / M1, M1 > 1}}] // N;
K = EllipticK[m[M1]] // N;
Kp = EllipticK[1 - m[M1]] // N;

computeTi[i_] :=
Module[{sigma = sigmas[[i]], r = rList[[i]], s = sList[[i]], f = fList[[i]], base},
  base = I * Im[InverseJacobiDN[Piecewise[
    {{Sqrt[f], M1 < 1}, {1 / Sqrt[f], M1 > 1}}], m[M1]]] / Kp;
  Which[sigma < 180, Which[r > 1 && s > 1, base, (r < 1 && s > 1) || (r > 1 && s < 1),
    1 + base, r < 1 && s < 1, 2 + base], sigma > 180, Which[r > 1 && s > 1,
    2 + base, (r < 1 && s > 1) || (r > 1 && s < 1), 3 + base, r < 1 && s < 1, base]]];

tList = computeTi /@ Range[4];
RoundWithTolerance[x_, tol_ : 10^-6] := Module[{nearest}, nearest = Round[x];
  If[Abs[x - nearest] ≤ tol, nearest, x]];

checkValidCombination[M1_, ε_ : 10^-6] :=
Module[{i, combo, dotProd, rePart, imPart, n2, expr, foundQ = False},
  Print[Style["△ Approximate validation using ε-tolerance. For rigorous
    proof, see the referenced paper.", Darker@Orange, Italic]];
  Do[combo = uniqueCombos[[i]];
    dotProd = tList.combo;
    rePart = Abs[Re[dotProd]];
    imPart = Abs[Im[dotProd]];
    If[M1 < 1,
      If[Mod[RoundWithTolerance[rePart], 4] < ε,
        If[Mod[RoundWithTolerance[imPart], 2] < ε, expr =
          tList[[1]] + combo[[2]] × tList[[2]] + combo[[3]] × tList[[3]] + combo[[4]] × tList[[4]];
          Print[Style["✔ Valid Combination Found (M < 1):",
            Darker[Green], Bold], "\n", Style["e1 = ", Bold], combo[[2]],
            Style["e2 = ", Bold], combo[[3]], Style["e3 = ", Bold], combo[[4]],
            "\n", Style["t1 = ", Bold], Re[tList[[1]]], "K + ", Im[tList[[1]]], "iK'",
            "\n", Style["t2 = ", Bold], Re[tList[[2]]], "K + ", Im[tList[[2]]], "iK'",
            "\n", Style["t3 = ", Bold], Re[tList[[3]]], "K + ", Im[tList[[3]]], "iK'",
            "\n", Style["t4 = ", Bold], Re[tList[[4]]], "K + ", Im[tList[[4]]],
            "iK'", "\n", Style["t1 + e1*t2 + e2*t3 + e3*t4 = ", Bold],
            Re[expr], "K + ", Im[expr], "iK'"];
          foundQ = True;
          Break[]]]];
    If[M1 > 1,
      If[Mod[RoundWithTolerance[imPart], 2] < ε,
        n2 = Quotient[RoundWithTolerance[imPart], 2];
        If[Mod[RoundWithTolerance[Abs[rePart - 2 n2]], 4] < ε, expr =
          tList[[1]] + combo[[2]] × tList[[2]] + combo[[3]] × tList[[3]] + combo[[4]] × tList[[4]];

```

```

Print[Style["✔ Valid Combination Found (M > 1):",
  Darker[Green], Bold], "\n", Style["e1 = ", Bold], combo[[2]],
  Style["e2 = ", Bold], combo[[3]], Style["e3 = ", Bold], combo[[4]],
  "\n", Style["t1 = ", Bold], Re[tList[[1]], "K + ", Im[tList[[1]], "iK'",
  "\n", Style["t2 = ", Bold], Re[tList[[2]], "K + ", Im[tList[[2]], "iK'",
  "\n", Style["t3 = ", Bold], Re[tList[[3]], "K + ", Im[tList[[3]], "iK'",
  "\n", Style["t4 = ", Bold], Re[tList[[4]], "K + ", Im[tList[[4]],
  "iK'", "\n", Style["t1 + e1*t2 + e2*t3 + e3*t4 = ", Bold],
  Re[expr], "K + ", Im[expr], "iK'"];
foundQ = True;
Break[]]]], {i, Length[uniqueCombos]}}];
If[! foundQ, Print[Style["✗ No valid combination found.", Red, Bold], "\n",
  Style["t1 = ", Bold], Re[tList[[1]], "K + ", Im[tList[[1]], "iK'", "\n",
  Style["t2 = ", Bold], Re[tList[[2]], "K + ", Im[tList[[2]], "iK'", "\n",
  Style["t3 = ", Bold], Re[tList[[3]], "K + ", Im[tList[[3]], "iK'", "\n",
  Style["t4 = ", Bold], Re[tList[[4]], "K + ", Im[tList[[4]], "iK'", "\n"]];];

Column[{TextCell[Style["===== CONDITION (N.5) =====",
  Purple, Bold, 16], "Text"]}
res = checkValidCombination[M1];

(*=====
====*)
(*=====
OTHER PARAMETER=====*)
(*=====
====*)
Column[
{TextCell[Style["===== OTHER PARAMETERS =====",
  Darker[Purple], Bold, 16], "Text"],
Row[{Style["u = ", Bold], 1 - M1}],
Row[{Style["σ1 = ", Bold], σ1 Degree, Style["σ2 = ", Bold], σ2 Degree,
  Style["σ3 = ", Bold], σ3 Degree, Style["σ4 = ", Bold], σ4 Degree}],
Row[{Style["f1 = ", Bold], f1, Style["f2 = ", Bold],
  f2, Style["f3 = ", Bold], f3, Style["f4 = ", Bold], f4}],
Row[{Style["z1 = ", Bold], FullSimplify[1 / (f1 - 1)], Style["z2 = ", Bold],
  FullSimplify[1 / (f2 - 1)], Style["z3 = ", Bold], FullSimplify[1 / (f3 - 1)],
  Style["z4 = ", Bold], FullSimplify[1 / (f4 - 1)]}],
Row[{Style["x1 = ", Bold], FullSimplify[1 / (r1 - 1)], Style["x2 = ", Bold],
  FullSimplify[1 / (r2 - 1)], Style["x3 = ", Bold], FullSimplify[1 / (r3 - 1)],
  Style["x4 = ", Bold], FullSimplify[1 / (r4 - 1)]}],
Row[{Style["y1 = ", Bold], FullSimplify[1 / (s1 - 1)], Style["y2 = ", Bold],
  FullSimplify[1 / (s2 - 1)], Style["y3 = ", Bold], FullSimplify[1 / (s3 - 1)],
  Style["y4 = ", Bold], FullSimplify[1 / (s4 - 1)]}],

```

```

Row[{Style["p1 = ", Bold], Simplify[Sqrt[r1 - 1]],
  Style["", p2 = ", Bold], Simplify[Sqrt[r2 - 1]], Style["", p3 = ", Bold],
  Simplify[Sqrt[r3 - 1]], Style["", p4 = ", Bold], Simplify[Sqrt[r4 - 1]]}],
Row[{Style["q1 = ", Bold], Simplify[Sqrt[s1 - 1]],
  Style["", q2 = ", Bold], Simplify[Sqrt[s2 - 1]], Style["", q3 = ", Bold],
  Simplify[Sqrt[s3 - 1]], Style["", q4 = ", Bold], Simplify[Sqrt[s4 - 1]]}],
Row[{Style["p1.q1 = ", Bold], Simplify[Sqrt[r1 - 1] Sqrt[s1 - 1]],
  Style["", p2.q2 = ", Bold], Simplify[Sqrt[r2 - 1] Sqrt[s2 - 1]],
  Style["", p3.q3 = ", Bold], Simplify[Sqrt[r3 - 1] Sqrt[s3 - 1]],
  Style["", p4.q4 = ", Bold], Simplify[Sqrt[r4 - 1] Sqrt[s4 - 1]]}],
Row[{Style["!\(\(*OverscriptBox[\(\alpha\), \(_\)]\) = ", Bold],
   $\sigma_1 - \text{anglesDeg}[1, 1]$ , ""], Style[
  "", \!\(\(*OverscriptBox[\(\beta\), \(_\)]\) = ", Bold],  $\sigma_1 - \text{anglesDeg}[1, 2]$ ,
  ""], Style["", \!\(\(*OverscriptBox[\(\gamma\), \(_\)]\) = ", Bold],
   $\sigma_1 - \text{anglesDeg}[1, 3]$ , ""],
  Style["", \!\(\(*OverscriptBox[\(\delta\), \(_\)]\) = ", Bold],
   $\sigma_1 - \text{anglesDeg}[1, 4]$ , ""}]
}]

(*=====
====*)
(*=====
BRICARD'S EQUATIONS=====*)
(*=====
====*)

(*Bricard's equation as defined in terms of angles and variables*)
BricardsEquation[{ $\alpha$ _,  $\beta$ _,  $\gamma$ _,  $\delta$ _,  $\sigma$ _, x_, y_] := Module[
  {c22, c20, c02, c11, c00},
  c22 = Sin[ $\sigma - \delta$ ] Sin[ $\sigma - \delta - \beta$ ];
  c20 = Sin[ $\sigma - \alpha$ ] Sin[ $\sigma - \alpha - \beta$ ];
  c02 = Sin[ $\sigma - \gamma$ ] Sin[ $\sigma - \gamma - \beta$ ];
  c11 = -Sin[ $\alpha$ ] Sin[ $\gamma$ ];
  c00 = Sin[ $\sigma$ ] Sin[ $\sigma - \beta$ ];
  c22 x^2 y^2 + c20 x^2 + c02 y^2 + 2 c11 x y + c00
];

(*Step 1: Bricard's system of equations*)
Column[{(*Header*)TextCell[
  Style["===== Bricard's System of Equations =====",
    Red, Bold, 16], "Text"], (*Explanatory note*)
Row[
  {TextCell[Style["We introduce new notation for the cotangents of half of
    the dihedral angles. Denote w, := ", GrayLevel[0.3], 13],
    "Text"], TraditionalForm[cot[Subscript[ $\theta$ , 1] / 2]], TextCell[

```

```

Style[" t:= ", GrayLevel[0.3], 13], "Text"],
TraditionalForm[cot[Subscript[0, 2] / 2]],
TextCell[Style[" w2:= ", GrayLevel[0.3], 13], "Text"],
TraditionalForm[cot[Subscript[0, 3] / 2]],
TextCell[Style[" and z:= ", GrayLevel[0.3], 13], "Text"],
TraditionalForm[cot[Subscript[0, 4] / 2]]
}], Spacer[12],
(*Traditional form results*)Row[{"P1(w1, z) = ",
TraditionalForm[Collect[FullSimplify[BricardsEquation[anglesDeg[[1]] Degree,
sigmas[[1]] Degree, w1, z]], w1]], " = 0"]], Spacer[6],
Row[{"P2(w1, t) = ",
TraditionalForm[Collect[FullSimplify[BricardsEquation[anglesDeg[[2]] Degree,
sigmas[[2]] Degree, w1, t]], w1]], " = 0"]], Spacer[6],
Row[{"P3(w2, t) = ",
TraditionalForm[Collect[FullSimplify[BricardsEquation[anglesDeg[[3]] Degree,
sigmas[[3]] Degree, w2, t]], w2]], " = 0"]], Spacer[6],
Row[{"P4(w2, z) = ", TraditionalForm[Collect[FullSimplify[BricardsEquation[
anglesDeg[[4]] Degree, sigmas[[4]] Degree, w2, z]], w2]], " = 0"]]
}]

(*discriminant*)
Disc[t_] := (Sin[sigma1 Degree] Sin[(sigma1 - alpha1 - beta1) Degree] +
t^2 Sin[(sigma1 - alpha1) Degree] Sin[(sigma1 - beta1) Degree])
(Sin[(sigma1 - gamma1) Degree] Sin[(sigma1 - delta1) Degree] + t^2 Sin[
(sigma1 - beta1 - gamma1) Degree] Sin[(sigma1 - beta1 - delta1) Degree]);

(*=====
FLEXION 1=====*)
e0 = 1;
W2[t_] := t;
Z[t_] := (-t Sin[beta1 Degree] + e0 Sqrt[Disc[t]]) /
(Sin[(sigma1 - delta1) Degree] Sin[(sigma1 - alpha1 - beta1) Degree] +
t^2 Sin[(sigma1 - alpha1) Degree] Sin[(sigma1 - beta1 - delta1) Degree]);
U[t_] := (t Sin[beta1 Degree] + e0 Sqrt[Disc[t]]) /
(Sin[(sigma1 - delta1) Degree] Sin[(sigma1 - alpha1 - beta1) Degree] +
t^2 Sin[(sigma1 - alpha1) Degree] Sin[(sigma1 - beta1 - delta1) Degree]);
W1[t_] := e0 Sqrt[Disc[t]] /
(Sin[(sigma1 - gamma1) Degree] Sin[(sigma1 - delta1) Degree] +
t^2 Sin[(sigma1 - beta1 - gamma1) Degree]
Sin[(sigma1 - beta1 - delta1) Degree]);

(*=====
FLEXION 2=====*)
e00 = -1;
W22[t_] := t;
Z2[t_] := (-t Sin[beta1 Degree] + e00 Sqrt[Disc[t]]) /
(Sin[(sigma1 - delta1) Degree] Sin[(sigma1 - alpha1 - beta1) Degree] +

```

```

t^2 Sin[(sigma1 - alpha1) Degree] Sin[(sigma1 - beta1 - delta1) Degree]];
U2[t_] := (t Sin[beta1 Degree] + e00 Sqrt[Disc[t]]) /
(Sin[(sigma1 - delta1) Degree] Sin[(sigma1 - alpha1 - beta1) Degree] +
t^2 Sin[(sigma1 - alpha1) Degree] Sin[(sigma1 - beta1 - delta1) Degree]);
W12[t_] := e00 Sqrt[Disc[t]] /
(Sin[(sigma1 - gamma1) Degree] Sin[(sigma1 - delta1) Degree] +
t^2 Sin[(sigma1 - beta1 - gamma1) Degree]
Sin[(sigma1 - beta1 - delta1) Degree]);

```

(\*Step 2: Checking that formula from Theorem 1 simplifies to (E.1)\*)

```

Column[
{(*Header*)TextCell[Style["===== FLEXIONS =====",
Darker[Red], Bold, 16], "Text"], (*Explanatory note*)
TextCell[Style["Solutions to Bricard's equations under a free parameter
t ∈ ℂ using Theorem 1:", GrayLevel[0.3], 13], "Text"], Spacer[12],
(*Heading for results*)TextCell[Style["Solution 1:", Bold, 14], "Text"],
Spacer[6], (*Traditional form results*)
Row[{"w1(t) = ", TraditionalForm[FullSimplify[Z[t]]}], Spacer[6],
Row[{"t(t) = ", TraditionalForm[FullSimplify[W2[t]]}], Spacer[6],
Row[{"w2(t) = ", TraditionalForm[FullSimplify[U[t]]}], Spacer[6],
Row[{"z(t) = ", TraditionalForm[FullSimplify[W1[t]]}], Spacer[12],
(*Heading for results*)TextCell[Style["Solution 2:", Bold, 14], "Text"],
Spacer[6], (*Traditional form results*)
Row[{"w1(t) = ", TraditionalForm[FullSimplify[Z2[t]]}], Spacer[6],
Row[{"t(t) = ", TraditionalForm[FullSimplify[W22[t]]}], Spacer[6],
Row[{"w2(t) = ", TraditionalForm[FullSimplify[U2[t]]}], Spacer[6],
Row[{"z(t) = ", TraditionalForm[FullSimplify[W12[t]]]}]
}]

```

(\*Step 3: Checking that (E.1) solves  $P_2(w_1, t) = 0$  and  $P_3(w_2, t) = 0$  even when  $\pm$  signs do NOT agree\*)

```

Column[
{(*Header*)TextCell[Style["===== Equations:  $P_2(w_1, t) = 0$ 
and  $P_3(w_2, t) = 0$  =====",
Orange, Bold, 16], "Text"], (*Explanatory note*)TextCell[
Style["Let  $w_{1s1}$  and  $w_{1s2}$  be formulas for  $w_1(t)$  from solutions
1 and 2, respectively. Similarly, let  $w_{2s1}$  and  $w_{2s2}$  be
the formulas for  $w_2(t)$ . We show that all four pairs -
 $(w_{1s1}, w_{2s1})$ ,  $(w_{1s1}, w_{2s2})$ ,  $(w_{1s2}, w_{2s1})$ , and  $(w_{1s2}, w_{2s2})$ 
- solve equations  $P_2(w_1, t) = 0$  and  $P_3(w_2, t) = 0$ .",
GrayLevel[0.3], 13], "Text"], Spacer[12],
(*Heading for results*)TextCell[Style["Pair 1:", Bold, 14], "Text"],
Spacer[6], (*Traditional form results*)
Row[{" $P_2(w_{1s1}, t) =$ ", TraditionalForm[Collect[FullSimplify[
BricardsEquation[anglesDeg[[2] Degree, sigmas[[2] Degree, Z[t], t]], w1]],

```



```

" ", Style["✓", Darker[Green], Bold]]], Spacer[6],
Row[{"P3(w2s1, t) = ", TraditionalForm[Collect[FullSimplify[
  BricardsEquation[anglesDeg[[3]] Degree, sigmas[[3]] Degree, U[t], t]], w2]],
" ", Style["✓", Darker[Green], Bold]]], Spacer[12],

TextCell[Style["Pair 2:", Bold, 14], "Text"],
Spacer[6], (*Traditional form results*)
Row[{"P2(w1s1, t) = ", TraditionalForm[Collect[FullSimplify[
  BricardsEquation[anglesDeg[[2]] Degree, sigmas[[2]] Degree, Z[t], t]], w1]],
" ", Style["✓", Darker[Green], Bold]]], Spacer[6],
Row[{"P3(w2s2, t) = ", TraditionalForm[Collect[FullSimplify[
  BricardsEquation[anglesDeg[[3]] Degree, sigmas[[3]] Degree, U2[t], t]], w2]],
" ", Style["✓", Darker[Green], Bold]]], Spacer[12],

TextCell[Style["Pair 3:", Bold, 14], "Text"],
Spacer[6], (*Traditional form results*)
Row[{"P2(w1s2, t) = ", TraditionalForm[Collect[FullSimplify[BricardsEquation[
  anglesDeg[[2]] Degree, sigmas[[2]] Degree, Z2[t], t]], w1]],
" ", Style["✓", Darker[Green], Bold]]], Spacer[6],
Row[{"P3(w2s1, t) = ", TraditionalForm[Collect[FullSimplify[
  BricardsEquation[anglesDeg[[3]] Degree, sigmas[[3]] Degree, U[t], t]], w2]],
" ", Style["✓", Darker[Green], Bold]]], Spacer[12],

TextCell[Style["Pair 4:", Bold, 14], "Text"],
Spacer[6], (*Traditional form results*)
Row[{"P2(w1s2, t) = ", TraditionalForm[Collect[FullSimplify[BricardsEquation[
  anglesDeg[[2]] Degree, sigmas[[2]] Degree, Z2[t], t]], w1]],
" ", Style["✓", Darker[Green], Bold]]], Spacer[6],
Row[{"P3(w2s2, t) = ", TraditionalForm[Collect[FullSimplify[
  BricardsEquation[anglesDeg[[3]] Degree, sigmas[[3]] Degree, U2[t], t]], w2]],
" ", Style["✓", Darker[Green], Bold]]]
}]

(*Step 4: Checking that (E.1) does NOT satisfy w1(t)/w2(t) =
c and w1(t)w2(t) =
c when c is less or greater -1 even when +- signs do NOT agree*)
tMin = 0;
tMax = 5;

expressions = {Z[t] * U[t], Z[t] / U[t], Z[t] * U2[t], Z[t] / U2[t],
  Z2[t] * U[t], Z2[t] / U[t], Z2[t] * U2[t], Z2[t] / U2[t]};
labels = {"w1s1 * w2s1", "w1s1 / w2s1", "w1s1 * w2s2", "w1s1 / w2s2",
  "w1s2 * w2s1", "w1s2 / w2s1", "w1s2 * w2s2", "w1s2 / w2s2"};
Column[
{TextCell[Style["===== Plots of w1(t)w2(t) and w1(t)/w2(t) For All
  Pairs =====", Darker[Orange], Bold, 16], "Text"],

```

```

(*Explanatory text*)
TextCell[Style["Checking that all four pairs -  $(w_{1s1}, w_{2s1}), (w_{1s1}, w_{2s2}),$ 
 $(w_{1s2}, w_{2s1}),$  and  $(w_{1s2}, w_{2s2})$  - does NOT satisfy  $w_1(t)/w_2(t) =$ 
 $c$  and  $w_1(t)w_2(t) = c$  when  $c \neq -1.$ ", GrayLevel[0.3]], "Text"],
Spacer[12],

(*Plots panel*)
Panel[GraphicsGrid[Partition[Table[Plot[expressions[[i]], {t, tMin, tMax},
PlotLabel → Style[labels[[i]], Bold, 14], PlotRange → All,
AxesLabel → {"t", None}, ImageSize → 250], {i, Length[expressions]}], 2],
Spacings → {2, 2}], Background → Lighter[Gray, 0.95], FrameMargins → 15]
]]

(*Step 5: Solving  $P_1(w_1, z) = 0$  and  $P_4(w_2, z) = 0$  for  $w_1$  and  $w_2$ *)
(*Define the first Bricard equation  $P_1(Z, z) = 0$ *)
P1 = BricardsEquation[anglesDeg[[1]] Degree, sigmas[[1]] Degree, Z, z];
(*Solve for Z1, Z2*)
solutionZExpressions = (Z /. FullSimplify[Solve[P1 == 0, Z]]);
Zz1[z_] := solutionZExpressions[[1]];
Zz2[z_] := solutionZExpressions[[2]];

(*Define the fourth Bricard equation  $P_4(U, z) = 0$ *)
P4 = BricardsEquation[anglesDeg[[4]] Degree, sigmas[[4]] Degree, U, z];
(*Solve for U1, U2*)
solutionUExpressions = (U /. FullSimplify[Solve[P4 == 0, U]]);
Uu1[z_] := solutionUExpressions[[1]];
Uu2[z_] := solutionUExpressions[[2]];

Column[
{(*Header*)TextCell[Style["===== Equations:  $P_1(w_1, z) = 0$ 
and  $P_4(w_2, z) = 0$  =====",
Darker[Cyan], Bold, 16], "Text"], (*Explanatory note*)
TextCell[Style["We solve  $P_1(w_1, z) = 0$  and  $P_4(w_2, z) = 0$  for  $w_1$  and  $w_2$ .",
GrayLevel[0.3], 13], "Text"], Spacer[12],
(*Heading for results*)Row[{" $w_{1,1}(z) =$ ", TraditionalForm[Zz1[z]]}],
Row[{" $w_{1,2}(z) =$ ", TraditionalForm[Zz2[z]]}],
Spacer[6], Row[{" $w_{2,1}(z) =$ ", TraditionalForm[Uu1[z]]}],
Row[{" $w_{2,2}(z) =$ ", TraditionalForm[Uu2[z]]}]
]

(*Step 6: Checking that pairs  $(w_{1,1}(z), w_{2,1}(z)), (w_{1,1}(z), w_{2,2}(z)),$ 
 $(w_{1,2}(z), w_{2,1}(z)),$  and  $(w_{1,2}(z), w_{2,2}(z))$  does NOT satisfy  $w_1(z)/w_2(z) =$ 
 $c$  and  $w_1(z)w_2(z) = c$  when  $c$  is less or greater 1*)
zMin = Sqrt[(3 - Sqrt[3]) / (3 + Sqrt[3])];
zMax = 1;

```

```

expressions = {Zz1[z] * Uu1[z], Zz1[z] / Uu1[z], Zz1[z] * Uu2[z], Zz1[z] / Uu2[z],
  Zz2[z] * Uu1[z], Zz2[z] / Uu1[z], Zz2[z] * Uu2[z], Zz2[z] / Uu2[z]};
labels = {"w1,1 * w2,1", "w1,1 / w2,1", "w1,1 * w2,2",
  "w1,1 / w2,2", "w1,2 * w2,1", "w1,2 / w2,1", "w1,2 * w2,2", "w1,2 / w2,2"};
Column[
  {TextCell[Style["===== Plots of w1(z)w2(z) and w1(z)/w2(z) For All
    Pairs =====", Magenta, Bold, 16], "Text"],

  (*Explanatory text*)
  TextCell[Style["Checking that all four pairs - (w1,1(z),
    w2,1(z)), (w1,1(z), w2,2(z)), (w1,2(z), w2,1(z)), and
    (w1,2(z), w2,2(z)) - does NOT satisfy w1(z)/w2(z) = c and
    w1(z)w2(z) = c when c ≠ +1.", GrayLevel[0.3]], "Text"],
  Spacer[12],

  (*Plots panel*)
  Panel[GraphicsGrid[Partition[Table[Plot[expressions[[i]], {z, zMin, zMax},
    PlotLabel → Style[labels[[i]], Bold, 14], PlotRange → All,
    AxesLabel → {"t", None}, ImageSize → 250], {i, Length[expressions]}], 2],
    Spacings → {2, 2}], Background → Lighter[Gray, 0.95], FrameMargins → 15]
  ]],

  (*Compute and print all P_i for flexion 1*)
  TextCell[
    Style["===== FLEXIBILITY (Double Checking) =====",
      Darker[Magenta], Bold, 16], "Text"]
  funcs = {{w1, z}, {w1, t}, {w2, t}, {w2, z}};
  TextCell[Style["Solution 1:", Bold, 14], "Text"]
  Do[angles = anglesDeg[[i]] Degree;
    sigma = sigmas[[i]] Degree;
    {α, β, γ, δ} = angles;
    poly = Which[i == 1, BricardsEquation[{α, β, γ, δ}, sigma, Z[t], W1[t]],
      i == 2, BricardsEquation[{α, β, γ, δ}, sigma, Z[t], W2[t]],
      i == 3, BricardsEquation[{α, β, γ, δ}, sigma, U[t], W2[t]],
      i == 4, BricardsEquation[{α, β, γ, δ}, sigma, U[t], W1[t]]];
    Print[Row[{Subscript["P", i], "("}, funcs[[i, 1]],
      ", ", funcs[[i, 2]], ") = ", FullSimplify[poly]]], {i, 1, 4}];

  (*Compute and print all P_i for flexion 2*)
  TextCell[Style["Solution 2:", Bold, 14], "Text"]
  funcs = {{w1, z}, {w1, t}, {w2, t}, {w2, z}};
  Do[angles = anglesDeg[[i]] Degree;
    sigma = sigmas[[i]] Degree;
    {α, β, γ, δ} = angles;
    poly = Which[i == 1, BricardsEquation[{α, β, γ, δ}, sigma, Z2[t], W12[t]],
      i == 2, BricardsEquation[{α, β, γ, δ}, sigma, Z2[t], W22[t]],
      i == 3, BricardsEquation[{α, β, γ, δ}, sigma, U2[t], W22[t]],

```

```

i = 4, BricardsEquation[{ $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\delta$ }, sigma, U2[t], W12[t]]];
Print[Row[{Subscript["P", i], "("}, funcs[[i, 1],
  ", ", funcs[[i, 2], ") = "], FullSimplify[poly]]], {i, 1, 4}];

(*=====
====*)
(*=====NOT LINEAR COMPOUND=====*)
(*=====
====*)
(*Define domain limits for t*)
tMin = 2;
tMax = 5;
(*{TraditionalForm[cot[Subscript[ $\theta$ ,1]/2]],
  TraditionalForm[cot[Subscript[ $\theta$ ,2]/2]],
  TraditionalForm[cot[Subscript[ $\theta$ ,3]/2]],
  TraditionalForm[cot[Subscript[ $\theta$ ,4]/2]]};*)

(*=====
FLEXION 1=====*)
(*List of expressions& labels*)
expressions =
{Z[t] * U[t], Z[t] / U[t], W1[t] * W2[t], W1[t] / W2[t], Z[t] * W2[t], Z[t] / W2[t],
  U[t] * W1[t], U[t] / W1[t], Z[t] * W1[t], Z[t] / W1[t], W2[t] * U[t], W2[t] / U[t]};
labels = {"w1 * w2", "w1 / w2", "z * t", "z / t", "w1 * t", "w1 / t",
  "w2 * z", "w2 / z", "w1 * z", "w1 / z", "w2 * t", "t / w2"};

Column[{TextCell[Style["===== NOT LINEAR COMPOUND =====",
  Darker[Brown], Bold, 16], "Text"],
  (*Explanatory text*)
  TextCell[Style["Above we consider the first pair of equations ( $P_2(w_1, t) = 0$  and  $P_3(w_2, t) = 0$ ). Solving them as quadratic equations in  $w_1$  and  $w_2$ , respectively we parametrize the solutions by the first three expressions in Solutions 1 and 2 in a neighborhood of any point  $(w_1, t, w_2)$  such that  $2 + 6t^2 + 3t^4 \neq 0$ . Here, we choose any continuous branch of the square root in this neighborhood, and the signs in  $\pm$  need Not agree (this means we consider all 4 pairs we describe above). We conclude that one one-dimensional irreducible component of the solution set of the first pair of Bricard's equations satisfies  $w_1/w_2=-1$ , and NO other component satisfies  $w_1/w_2=\text{const}$  NOR  $w_1w_2=\text{const}$ ."], "Text"]}];

Analogously, one one-dimensional irreducible component of the solution set of the other pair of equations ( $P_1(w_1, z) = 0$  and  $P_4(w_2, z) = 0$ ) satisfies  $w_1/w_2=+1$ , and NO other component satisfies  $w_1/w_2=\text{const}$  NOR  $w_1w_2=\text{const}$ . As a result, NO component of the solution set of all four equations satisfies

```

```

w1/w2=const NOR w1w2=const. So, our example does not belong
to the linear compound class, even after switching the
boundary strips.", GrayLevel[0.3]], "Text"],
Spacer[6],
TextCell[
Style["Below, we also present the plots of functions w1w2, w1/w2, zt,
z/t, w1t, w1/t, w2z, w2/z, w1z, w1/z,
w2t, and t/w2.", GrayLevel[0.3]], "Text"],
Spacer[6],
TextCell[Style["Solution 1:", Bold, 14], "Text"], Spacer[12],
Panel[GraphicsGrid[Partition[Table[Plot[expressions[[i]], {t, tMin, tMax},
PlotLabel → Style[labels[[i]], Bold, 14], PlotRange → All,
AxesLabel → {"t", None}, ImageSize → 250], {i, Length[expressions]}], 2],
Spacings → {2, 2}], Background → Lighter[Gray, 0.95], FrameMargins → 15]
]]

```

```

(*=====
FLEXION 2=====*)
(*List of expressions& labels*)
expressions = {Z2[t] * U2[t], Z2[t] / U2[t], W12[t] * W22[t], W12[t] / W22[t],
Z2[t] * W22[t], Z2[t] / W22[t], U2[t] * W12[t], U2[t] / W12[t],
Z2[t] * W12[t], Z2[t] / W12[t], W22[t] * U2[t], W22[t] / U2[t]};
labels = {"w1 * w2", "w1 / w2", "z * t", "z / t", "w1 * t", "w1 / t",
"w2 * z", "w2 / z", "w1 * z", "w1 / z", "w2 * t", "t / w2"};

Column[{ TextCell[Style["Solution 2:", Bold, 14], "Text"], Spacer[12],
(*Plots panel*)
Panel[GraphicsGrid[Partition[Table[Plot[expressions[[i]], {t, tMin, tMax},
PlotLabel → Style[labels[[i]], Bold, 14], PlotRange → All,
AxesLabel → {"t", None}, ImageSize → 250], {i, Length[expressions]}], 2],
Spacings → {2, 2}], Background → Lighter[Gray, 0.95], FrameMargins → 15]
]]

```

```

(*=====
=====*)
(*=====
NOT TRIVIAL=====*)
(*=====
=====*)
(*Define domain limits for t*)
tMin = 0;
tMax = 5;

(*=====
FLEXION 1=====*)

```

```

(*List of expressions to plot*)
expressions = {Z[t], W2[t], U[t], W1[t]};
labels = {"w1", "t", "w2", "z"};

Column[{TextCell[
  Style["===== NOT TRIVIAL (FLEXION 1) =====",
    Darker[Pink], Bold, 16], "Text"],
  (*Explanatory text*)
  TextCell[Style["This configuration does not belong to the trivial class –
    even after switching the boundary strips – since none of the
    functions w1, t, w2, or z is constant.", GrayLevel[0.3]], "Text"],
  Spacer[12],
  (*Plots in a light panel*)
  Panel[GraphicsGrid[Partition[Table[Plot[expressions[[i]], {t, tMin, tMax},
    PlotLabel → Style[labels[[i]], Bold, 14], PlotRange → All,
    AxesLabel → {"t", None}, ImageSize → 250], {i, Length[expressions]}], 2],
    Spacings → {2, 2}], Background → Lighter[Gray, 0.95], FrameMargins → 15]

  ]}]

(*=====
FLEXION 2=====*)
(*List of expressions to plot*)
expressions = {Z2[t], W22[t], U2[t], W12[t]};
labels = {"w1", "t", "w2", "z"};

Column[{TextCell[
  Style["===== NOT TRIVIAL (FLEXION 2) =====",
    Darker[Green], Bold, 16], "Text"],

  (*Explanatory text*)
  TextCell[Style["This configuration does not belong to the trivial class –
    even after switching the boundary strips – since none of the
    functions w1, t, w2, or z is constant.", GrayLevel[0.3]], "Text"],
  Spacer[12],

  (*Plots in a light panel*)
  Panel[GraphicsGrid[Partition[Table[Plot[expressions[[i]], {t, tMin, tMax},
    PlotLabel → Style[labels[[i]], Bold, 14], PlotRange → All,
    AxesLabel → {"t", None}, ImageSize → 250], {i, Length[expressions]}], 2],
    Spacings → {2, 2}], Background → Lighter[Gray, 0.95], FrameMargins → 15]

  ]}]

(*=====
=====*)

```

```

(*=====
SWITCHING BOUNDARY STRIPS=====*)
(*=====
=====*)
SwitchingRightBoundaryStrip[anglesDeg_List] := Module[{modified = anglesDeg},
  (*Indices:{row,column}={1,2},{1,3},{4,2},{4,3}*)
  modified[[1, 2]] = 180 - anglesDeg[[1, 2]]; (*β1*)
  modified[[1, 3]] = 180 - anglesDeg[[1, 3]]; (*γ1*)
  modified[[4, 2]] = 180 - anglesDeg[[4, 2]]; (*β4*)
  modified[[4, 3]] = 180 - anglesDeg[[4, 3]]; (*γ4*)
  modified]

SwitchingLeftBoundaryStrip[anglesDeg_List] := Module[{modified = anglesDeg},
  (*Indices:{row,column}={2,2},{2,3},{3,2},{3,3}*)
  modified[[2, 2]] = 180 - anglesDeg[[2, 2]]; (*β2*)
  modified[[2, 3]] = 180 - anglesDeg[[2, 3]]; (*γ2*)
  modified[[3, 2]] = 180 - anglesDeg[[3, 2]]; (*β3*)
  modified[[3, 3]] = 180 - anglesDeg[[3, 3]]; (*γ3*)
  modified]

SwitchingLowerBoundaryStrip[anglesDeg_List] := Module[{modified = anglesDeg},
  (*Indices:{row,column}={1,1},{1,2},{2,1},{2,2}*)
  modified[[1, 1]] = 180 - anglesDeg[[1, 1]]; (*α1*)
  modified[[1, 2]] = 180 - anglesDeg[[1, 2]]; (*β1*)
  modified[[2, 1]] = 180 - anglesDeg[[2, 1]]; (*α2*)
  modified[[2, 2]] = 180 - anglesDeg[[2, 2]]; (*β2*)
  modified]

SwitchingUpperBoundaryStrip[anglesDeg_List] := Module[{modified = anglesDeg},
  (*Indices:{row,column}={3,1},{3,2},{4,1},{4,2}*)
  modified[[3, 1]] = 180 - anglesDeg[[3, 1]]; (*α3*)
  modified[[3, 2]] = 180 - anglesDeg[[3, 2]]; (*β3*)
  modified[[4, 1]] = 180 - anglesDeg[[4, 1]]; (*α4*)
  modified[[4, 2]] = 180 - anglesDeg[[4, 2]]; (*β4*)
  modified]

(*=====
=====*)
(*=====NOT CONIC & NOT CHIMERA & NOT LINEAR
CONJUGATE & NOT ISOGONAL=====*)
(*=====
=====*)
Column[{TextCell[
  Style["===== NOT CONIC & NOT CHIMERA & NOT LINEAR CONJUGATE
& NOT ISOGONAL=====", Blue, Bold, 16], "Text"],
TextCell[Style[

```

```

"Condition (N.0) is satisfied for all  $i=1,\dots,4 \Rightarrow$  NOT equimodular-conic,
NOT chimera, NOT isogonal and NOT linear conjugate.
Applying any boundary-strip switch still preserves
(N.0), so no conic, no chimera, no isogonal and no
linear conjugate form emerges.", GrayLevel[0.3]], "Text"]
}]

(*Now the exact same Module for checking all switch combinations...*)
Module[{angles = anglesDeg, switchers, combinations, results},
  (*Define switch functions*)
  switchers = <|"Right" → SwitchingRightBoundaryStrip,
    "Left" → SwitchingLeftBoundaryStrip, "Lower" → SwitchingLowerBoundaryStrip,
    "Upper" → SwitchingUpperBoundaryStrip|>;
  (*Generate all combinations of switches (from size 1 to 4)*)
  combinations = Subsets[Keys[switchers], {1, Length[switchers]}];
  (*Evaluate condition after each combination of switches*) results = Table[
    Module[{switched = angles, name, passQ}, name = StringRiffle[combo, " + "];
    Do[switched = switchers[sw][switched], {sw, combo}];
    passQ = And@@ (checkConditionN0Degrees /@ switched);
    (*Print the result after switching*)
    (*Print["\nSwitch combination: ", name];
    Print["Switched anglesDeg:"];
    Print[MatrixForm[switched]];*)
    {name, passQ}], {combo, combinations}];
  (*Display results*)
  Column[Prepend[Table[Module[{comboName = res[[1]], passQ = res[[2]]},
    Row[{Style[comboName <> ": ", Bold],
      If[passQ,
        Style["Condition (N.0) is still satisfied.", Darker[Green]],
        Style["Condition (N.0) fails.", Red, Bold]
      ]
    }, {res, results}], TextCell[
    Style["CONDITION (N.0) AFTER SWITCHING BOUNDARY STRIPS", 14], "Text"]]]],

  (*=====
  =====*)
  (*=====
  NOT ORTHODIAGONAL=====*)
  (*=====
  =====*)

  (*Column[
    {TextCell[Style["===== NOT ORTHODIAGONAL =====",
      Darker[Blue], Bold, 16], "Text"],
    TextCell[Style[

```



```

"cos( $\alpha_i$ )·cos( $\gamma_i$ )  $\neq$  cos( $\beta_i$ )·cos( $\delta_i$ ) for each i = 1  $\Rightarrow$  NOT orthodiagonal.
Switching boundary strips does not
correct this.", GrayLevel[0.3]], "Text"]
}]

Module[{angles=anglesDeg,switchers,combinations,results},
(*Define switch functions*)switchers=<|"Right"→SwitchingRightBoundaryStrip,
"Left"→SwitchingLeftBoundaryStrip,"Lower"→SwitchingLowerBoundaryStrip,
"Upper"→SwitchingUpperBoundaryStrip|>;
(*Helper function:compute and print difference only*)
formatOrthodiagonalCheck[quad_List]:=
Module[{vals},vals=Table[Module[{a,b,c,d,lhs,rhs,diff},{a,b,c,d}=quad[[i]];
lhs=FullSimplify[Cos[a Degree] Cos[c Degree]];
rhs=FullSimplify[Cos[b Degree] Cos[d Degree]];
diff=Chop[lhs-rhs];
Style[Row[{"cos( $\alpha$ "<>ToString[i]<>")·cos( $\gamma$ "<>ToString[i]<>") - ",
"cos( $\beta$ "<>ToString[i]<>")·cos( $\delta$ "<>ToString[i]<>") = ",
NumberForm[diff,{5,3}]]],If[diff==0,Red,Black]]],{i,Length[quad]}];
Column[vals]];
(*Orthodiagonal check for anglesDeg before any switching*)
Print[TextCell[Style["\nInitial anglesDeg (no switches):",Bold]]];
Print[MatrixForm[angles]];
Print[TextCell[Style[
"Orthodiagonal check: cos( $\alpha_i$ )·cos( $\gamma_i$ ) - cos( $\beta_i$ )·cos( $\delta_i$ ) for i = 1..4",
Italic]]];
Print[formatOrthodiagonalCheck[angles]];
(*Generate all combinations of switches (from size 1 to 4)*)
combinations=Subsets[Keys[switchers],{1,Length[switchers]}];
(*Evaluate condition after each combination of switches*)results=
Table[Module[{switched=angles,name,passQ},name=StringRiffle[combo," + "];
Do[switched=switchers[sw][switched],{sw,combo}];
passQ=And@@(checkConditionN0Degrees/@switched);
Print[Style["\nSwitch combination: ", Bold],name];
Print[Style["Switched anglesDeg:", Italic]];
Print[MatrixForm[switched]];
Print[
TextCell[Style["Orthodiagonal check: cos( $\alpha_i$ )·cos( $\gamma_i$ ) - cos( $\beta_i$ )·cos( $\delta_i$ )
for i = 1..4",Italic]]];
Print[formatOrthodiagonalCheck[switched]];
{name,passQ}],{combo,combinations}]]];*)

Column[
{TextCell[Style["===== ORTHOGONALITY CHECK =====",
Darker[Blue], Bold, 16], "Text"],
TextCell[Style["cos( $\alpha_i$ )·cos( $\gamma_i$ )  $\neq$  cos( $\beta_i$ )·cos( $\delta_i$ ) for at least
one i = 1,..., 4  $\Rightarrow$  NOT orthodiagonal. Switching boundary

```

```

strips does not correct this.", GrayLevel[0.3]], "Text"]]]

(*Helper
function:Returns True if at least one cosine product difference is non-
zero.Returns False if all differences are zero.*)
isNotOrthodiagonal[quad_List] :=
  Or @@ Table[Module[{a, b, c, d, diff}, {a, b, c, d} = quad[[i]];
    diff = Chop[Cos[a Degree] Cos[c Degree] - Cos[b Degree] Cos[d Degree]];
    diff != 0], {i, Length[quad]}];

(*First,check the initial,unswitched angles*)
Print[TextCell[Style["\nInitial anglesDeg (no switches):", Bold]]];
If[isNotOrthodiagonal[anglesDeg], Print[Style[
  " -> Condition met: At least one difference is non-zero.", Darker@Green]],
  Print[Style[" -> Condition NOT met: All differences are zero.", Red]]];

(*Now,use your desired module to check all switch combinations*)
Module[{angles = anglesDeg, switchers, combinations, results},
  (*Define switch functions*)
  switchers = <|"Right" -> SwitchingRightBoundaryStrip,
    "Left" -> SwitchingLeftBoundaryStrip, "Lower" -> SwitchingLowerBoundaryStrip,
    "Upper" -> SwitchingUpperBoundaryStrip|>;
  (*Generate all combinations of switches (from size 1 to 4)*)
  combinations = Subsets[Keys[switchers], {1, Length[switchers]}];
  (*Evaluate condition after each combination
  of switches and store in 'results'*)results = Table[
    Module[{switched = angles, name, passQ}, name = StringRiffle[combo, " + "];
    Do[switched = switchers[sw][switched], {sw, combo}];
    (* ***THIS IS THE KEY CHANGE*****) (*Set passQ using our
    new helper function*)passQ = isNotOrthodiagonal[switched];
    {name, passQ}], {combo, combinations}];
  (*Display results in the specified column format*)
  Column[Prepend[Table[Module[{comboName = res[[1]], passQ = res[[2]]},
    Row[{Style[comboName <> ": ", Bold], If[passQ,
      Style["Condition met (at least one difference is non-zero).", Darker[
        Green]], Style["Condition NOT met (all differences are zero).",
        Red, Bold]}]], {res, results}], TextCell[
    Style["\nNON-ORTHOGONALITY CHECK AFTER SWITCHING BOUNDARY STRIPS", 14],
    "Text"]]]]

(*=====
====*)
(*=====
NOT CONJUGATE-MODULAR=====*)
(*=====

```

```

====*)
(*Column[{TextCell[
  Style["===== NOT CONJUGATE-MODULAR =====",
    Purple,Bold,16],"Text"],
  TextCell[Style["M1 = M2 = M3 = M4 = M
    and M ≠ 2 ⇒ NOT conjugate-modular. Boundary-strip
    switches preserve this.",GrayLevel[0.3]],"Text"]
}]
Ms=FullSimplify[Times@@@results];
allEqualQ=Simplify[Equal@@Ms];

Module[{angles=anglesDeg,switchers,combinations,results,
  computeConjugateModularInfo},(*Define switch functions*)
switchers=<|"Right"→SwitchingRightBoundaryStrip,
  "Left"→SwitchingLeftBoundaryStrip,"Lower"→SwitchingLowerBoundaryStrip,
  "Upper"→SwitchingUpperBoundaryStrip|>;
(*Computes Mi and pi and prints them,
with classification*)computeConjugateModularInfo[quad_List]:=
Module[{abcdList,Ms,summary},abcdList=computeABCD/@quad;
Ms=FullSimplify[Times@@@abcdList];
summary=If[Simplify[Equal@@Ms]&&Ms[[1]]!=2,
  Style["M1 = M2 = M3 = M4 = M and M ≠ 2",Bold],
  Style["M1 = M2 = M3 = M4 = M and M = 2",Red,Bold]];
Column[{Style["Mi values:",Bold],Row[{"M1 = ",Ms[[1],
  ", M2 = ",Ms[[2]],", M3 = ",Ms[[3]],", M4 = ",Ms[[4]]},summary]}}];
(*Original anglesDeg check*)
Print[
  TextCell[Style["\nInitial configuration (no switches applied):",Bold]]];
Print[MatrixForm[angles]];
Print[computeConjugateModularInfo[angles]];
(*Generate all switch combinations (from size 1 to 4)*)
combinations=Subsets[Keys[switchers],{1,Length[switchers]}];
(*Evaluate each switched configuration*)results=
Table[Module[{switched=angles,name,passQ},name=StringRiffle[combo," + "];
  Do[switched=switchers[sw][switched],{sw,combo}];
  passQ=And@@(checkConditionN0Degrees/@switched);
  Print[Style["\nSwitch combination: ", Bold],name];
  Print[Style["Switched anglesDeg:", Italic]];
  Print[MatrixForm[switched]];
  Print[computeConjugateModularInfo[switched]];
  {name,passQ}},{combo,combinations}];*)

Column[{TextCell[
  Style["===== CONJUGATE-MODULAR CHECK =====",
    Purple, Bold, 16], "Text"],
  TextCell[Style["M1 = M2 = M3 = M4 = M and M ≠ 2 ⇒ NOT conjugate-modular.

```

```

Boundary-strip switches preserve this.", GrayLevel[0.3]], "Text"]]]

(*Helper Function:Returns True if all M_i values are equal
AND their common value is not 2. Returns False otherwise.*)
isNotConjugateModular[quad_List] :=
Module[{abcdList, Ms}, abcdList = computeABCD /@ quad;
Ms = FullSimplify[Times@@@abcdList];
(*The condition is met if they are all equal AND the value isn't 2*)
Simplify[Equal@@Ms] && (Ms[[1]] ≠ 2)];

(*First,check the initial,unswitched angles*)
Print[TextCell[Style["\nInitial anglesDeg (no switches):", Bold]]];
If[isNotConjugateModular[anglesDeg], Print[
Style[" -> Condition met: All Mi are equal and M ≠ 2.", Darker@Green]],
Print[Style[" -> Condition NOT met.", Red]]];

(*Now,use the clean module to check all switch combinations*)
Module[{angles = anglesDeg, switchers, combinations, results},
(*Define switch functions*)
switchers = <|"Right" → SwitchingRightBoundaryStrip,
"Left" → SwitchingLeftBoundaryStrip, "Lower" → SwitchingLowerBoundaryStrip,
"Upper" → SwitchingUpperBoundaryStrip|>;
(*Generate all combinations of switches (from size 1 to 4)*)
combinations = Subsets[Keys[switchers], {1, Length[switchers]}];
(*Evaluate condition after each combination
of switches and store the result*)results = Table[
Module[{switched = angles, name, passQ}, name = StringRiffle[combo, " + "];
Do[switched = switchers[sw][switched], {sw, combo}];
(*Set passQ using our new helper function for this check*)
passQ = isNotConjugateModular[switched];
{name, passQ}], {combo, combinations}];
(*Display results in the specified column format*)
Column[Prepend[Table[Module[{comboName = res[[1]], passQ = res[[2]]},
Row[{Style[comboName <> ": ", Bold], If[passQ,
Style["Condition met (All Mi are equal and M ≠ 2).", Darker[Green]],
Style["Condition NOT met.", Red, Bold]]}], {res, results}],
TextCell[Style["\nCONJUGATE-MODULAR CHECK AFTER SWITCHING BOUNDARY STRIPS",
14], "Text"]]]]

```

Out[5460]=

```

===== CONDITION (N.0) =====
✓ All vertices satisfy (N.0).

```

Out[5463]=

```

===== CONDITION (N.3) =====
✓ M1 = M2 = M3 = M4 =  $-1 + \sqrt{3}$ 

```

Out[5469]=

===== CONDITION (N.4) =====  
 $\checkmark \mathbf{r1} = \mathbf{r2} = \frac{2}{\sqrt{3}}; \checkmark \mathbf{r3} = \mathbf{r4} = \frac{2}{\sqrt{3}}$   
 $\checkmark \mathbf{s1} = \mathbf{s4} = 3 - \sqrt{3}; \checkmark \mathbf{s2} = \mathbf{s3} = \frac{1}{\sqrt{3}}$

Out[5479]=

===== CONDITION (N.5) =====

$\triangle$  Approximate validation using  
 $\epsilon$ -tolerance. For rigorous proof, see the referenced paper.

 **Valid Combination Found (M < 1):**

$\mathbf{e1} = -1, \mathbf{e2} = -1, \mathbf{e3} = 1$   
 $\mathbf{t1} = 0.K + 0.446521iK'$   
 $\mathbf{t2} = 1.K + 0.446521iK'$   
 $\mathbf{t3} = 3.K + 0.446521iK'$   
 $\mathbf{t4} = 0.K + 0.446521iK'$   
 $\mathbf{t1} + \mathbf{e1}*\mathbf{t2} + \mathbf{e2}*\mathbf{t3} + \mathbf{e3}*\mathbf{t4} = -4.K + 0.iK'$

Out[5481]=

===== OTHER PARAMETERS =====

$\mathbf{u} = 2 - \sqrt{3}$   
 $\sigma1 = 165^\circ, \sigma2 = 165^\circ, \sigma3 = 195^\circ, \sigma4 = 165^\circ$   
 $\mathbf{f1} = \frac{1}{2} (1 + \sqrt{3}), \mathbf{f2} = 4 - 2\sqrt{3}, \mathbf{f3} = 4 - 2\sqrt{3}, \mathbf{f4} = \frac{1}{2} (1 + \sqrt{3})$   
 $\mathbf{z1} = 1 + \sqrt{3}, \mathbf{z2} = -1 - \frac{2}{\sqrt{3}}, \mathbf{z3} = -1 - \frac{2}{\sqrt{3}}, \mathbf{z4} = 1 + \sqrt{3}$   
 $\mathbf{x1} = 3 + 2\sqrt{3}, \mathbf{x2} = 3 + 2\sqrt{3}, \mathbf{x3} = 3 + 2\sqrt{3}, \mathbf{x4} = 3 + 2\sqrt{3}$   
 $\mathbf{y1} = 2 + \sqrt{3}, \mathbf{y2} = \frac{1}{-1 + \frac{1}{\sqrt{3}}}, \mathbf{y3} = \frac{1}{-1 + \frac{1}{\sqrt{3}}}, \mathbf{y4} = 2 + \sqrt{3}$   
 $\mathbf{p1} = \sqrt{-1 + \frac{2}{\sqrt{3}}}, \mathbf{p2} = \sqrt{-1 + \frac{2}{\sqrt{3}}}, \mathbf{p3} = \sqrt{-1 + \frac{2}{\sqrt{3}}}, \mathbf{p4} = \sqrt{-1 + \frac{2}{\sqrt{3}}}$   
 $\mathbf{q1} = \sqrt{2 - \sqrt{3}}, \mathbf{q2} = i\sqrt{1 - \frac{1}{\sqrt{3}}}, \mathbf{q3} = i\sqrt{1 - \frac{1}{\sqrt{3}}}, \mathbf{q4} = \sqrt{2 - \sqrt{3}}$   
 $\mathbf{p1} \cdot \mathbf{q1} = \sqrt{-4 + \frac{7}{\sqrt{3}}}, \mathbf{p2} \cdot \mathbf{q2} = i\sqrt{-\frac{5}{3} + \sqrt{3}}, \mathbf{p3} \cdot \mathbf{q3} = i\sqrt{-\frac{5}{3} + \sqrt{3}}, \mathbf{p4} \cdot \mathbf{q4} = \sqrt{-4 + \frac{7}{\sqrt{3}}}$   
 $\overline{\alpha 1} = 60^\circ, \overline{\beta 1} = 150^\circ, \overline{\gamma 1} = 45^\circ, \overline{\delta 1} = 75^\circ$

Out[5483]=

## ===== Bricard's System of Equations =====

We introduce new notation for the

cotangents of half of the dihedral angles. Denote  $w_1 :=$

$$\cot\left(\frac{\theta_1}{2}\right), \quad t := \cot\left(\frac{\theta_2}{2}\right), \quad w_2 := \cot\left(\frac{\theta_3}{2}\right), \quad \text{and} \quad z := \cot\left(\frac{\theta_4}{2}\right)$$

$$P_1(w_1, z) = \frac{w_1^2 ((3 + \sqrt{3}) z^2 + 2\sqrt{3})}{4\sqrt{2}} - \frac{(3 + \sqrt{3}) w_1 z}{2\sqrt{2}} + \frac{2z^2 + \sqrt{3} - 1}{4\sqrt{2}} = 0$$

$$P_2(w_1, t) = \frac{1}{4} (-3t^2 - \sqrt{3} - 1) w_1^2 + \frac{1}{4} (t^2 + \sqrt{3} - 1) - \frac{(\sqrt{3} - 1) t w_1}{\sqrt{2}} = 0$$

$$P_3(w_2, t) = \frac{1}{4} (3t^2 + \sqrt{3} + 1) w_2^2 + \frac{1}{4} (-t^2 - \sqrt{3} + 1) - \frac{(\sqrt{3} - 1) t w_2}{\sqrt{2}} = 0$$

$$P_4(w_2, z) = \frac{w_2^2 ((3 + \sqrt{3}) z^2 + 2\sqrt{3})}{4\sqrt{2}} - \frac{(3 + \sqrt{3}) w_2 z}{2\sqrt{2}} + \frac{2z^2 + \sqrt{3} - 1}{4\sqrt{2}} = 0$$

Out[5495]=

## ===== FLEXIONS =====

Solutions to Bricard's equations under a free parameter  $t \in \mathbb{C}$  using Theorem 1:

### Solution 1:

$$w_1(t) = \frac{\sqrt{3t^4 + 6t^2 + 2} - \sqrt{6}t + \sqrt{2}t}{3t^2 + \sqrt{3} + 1}$$

$$t(t) = t$$

$$w_2(t) = \frac{\sqrt{3t^4 + 6t^2 + 2} + \sqrt{2}(\sqrt{3} - 1)t}{3t^2 + \sqrt{3} + 1}$$

$$z(t) = \frac{\sqrt{3t^4 + 6t^2 + 2}}{\sqrt{3}t^2 + \sqrt{3} + 1}$$

### Solution 2:

$$w_1(t) = -\frac{\sqrt{3t^4 + 6t^2 + 2} + \sqrt{2}(\sqrt{3} - 1)t}{3t^2 + \sqrt{3} + 1}$$

$$t(t) = t$$

$$w_2(t) = -\frac{\sqrt{3t^4 + 6t^2 + 2} - \sqrt{6}t + \sqrt{2}t}{3t^2 + \sqrt{3} + 1}$$

$$z(t) = -\frac{\sqrt{3t^4 + 6t^2 + 2}}{\sqrt{3}t^2 + \sqrt{3} + 1}$$

Out[5496]=

===== Equations:  $P_2(w_1,$   
 $t) = 0$  and  $P_3(w_2, t) = 0$  =====

Let  $w_{1s1}$  and  $w_{1s2}$  be formulas for  $w_1(t)$  from solutions 1 and 2, respectively.

Similarly, let  $w_{2s1}$  and  $w_{2s2}$  be the formulas for  $w_2(t)$ . We show  
 that all four pairs –  $(w_{1s1}, w_{2s1})$ ,  $(w_{1s1}, w_{2s2})$ ,  $(w_{1s2}, w_{2s1})$ , and  
 $(w_{1s2}, w_{2s2})$  – solve equations  $P_2(w_1, t) = 0$  and  $P_3(w_2, t) = 0$ .

#### Pair 1:

$$P_2(w_{1s1}, t) = 0 \quad \checkmark$$

$$P_3(w_{2s1}, t) = 0 \quad \checkmark$$

#### Pair 2:

$$P_2(w_{1s1}, t) = 0 \quad \checkmark$$

$$P_3(w_{2s2}, t) = 0 \quad \checkmark$$

#### Pair 3:

$$P_2(w_{1s2}, t) = 0 \quad \checkmark$$

$$P_3(w_{2s1}, t) = 0 \quad \checkmark$$

#### Pair 4:

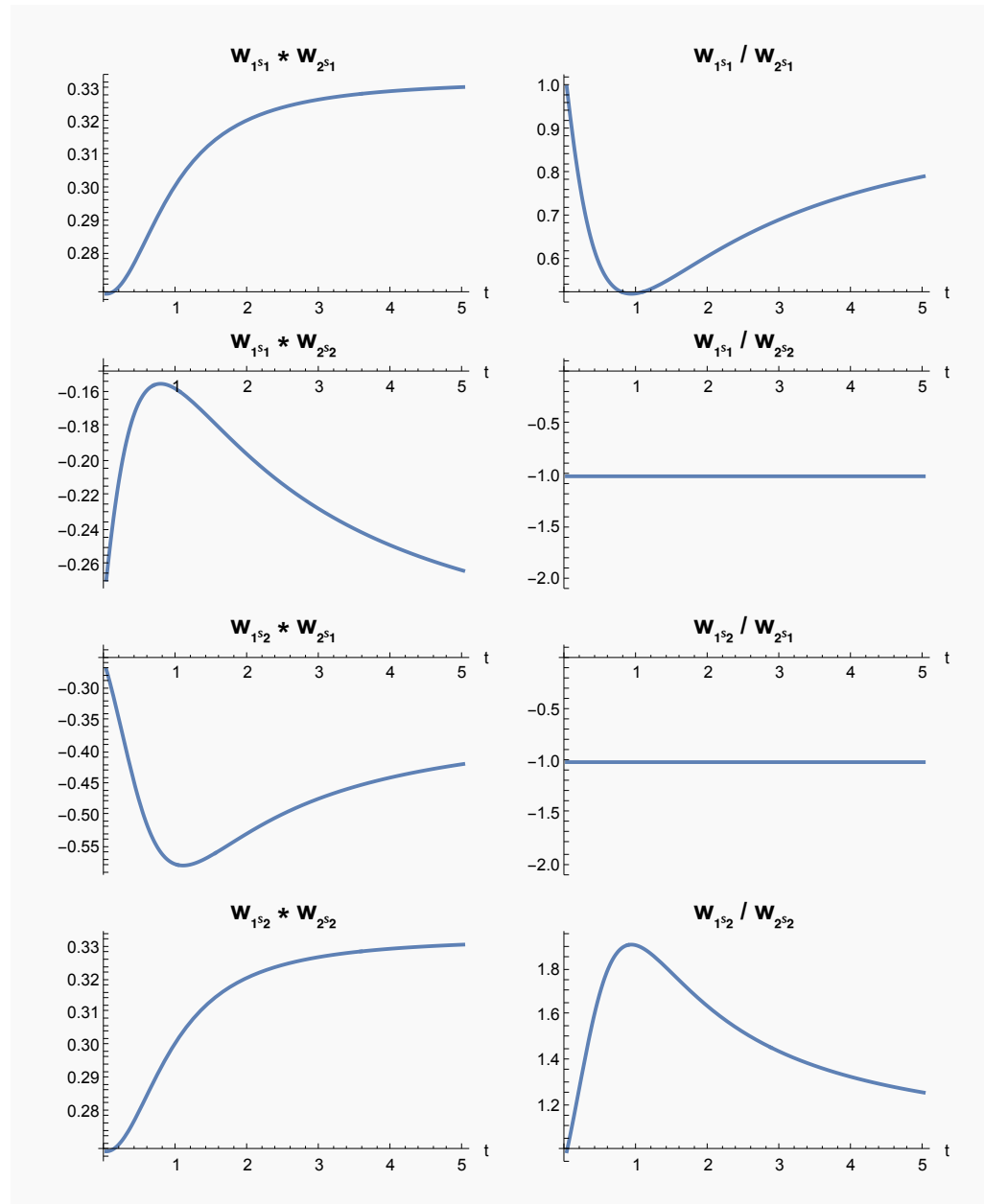
$$P_2(w_{1s2}, t) = 0 \quad \checkmark$$

$$P_3(w_{2s2}, t) = 0 \quad \checkmark$$

Out[5501]=

===== Plots of  $w_1(t)w_2(t)$   
and  $w_1(t)/w_2(t)$  For All Pairs =====

Checking that all four pairs –  $(w_{1s1}, w_{2s1})$ ,  
 $(w_{1s1}, w_{2s2})$ ,  $(w_{1s2}, w_{2s1})$ , and  $(w_{1s2}, w_{2s2})$  – does NOT  
satisfy  $w_1(t)/w_2(t) = c$  and  $w_1(t)w_2(t) = c$  when  $c \neq -1$ .





Out[5510]=

===== Equations:  $P_1(w_1, z) = 0$  and  $P_4(w_2, z) = 0$  =====

We solve  $P_1(w_1, z) = 0$  and  $P_4(w_2, z) = 0$  for  $w_1$  and  $w_2$ .

$$w_{1,1}(z) = \frac{(3 + \sqrt{3})z - \sqrt{2} \sqrt{-(3 + \sqrt{3})z^4 + 6z^2 + \sqrt{3} - 3}}{(3 + \sqrt{3})z^2 + 2\sqrt{3}}$$

$$w_{1,2}(z) = \frac{\sqrt{2} \sqrt{-(3 + \sqrt{3})z^4 + 6z^2 + \sqrt{3} - 3} + (3 + \sqrt{3})z}{(3 + \sqrt{3})z^2 + 2\sqrt{3}}$$

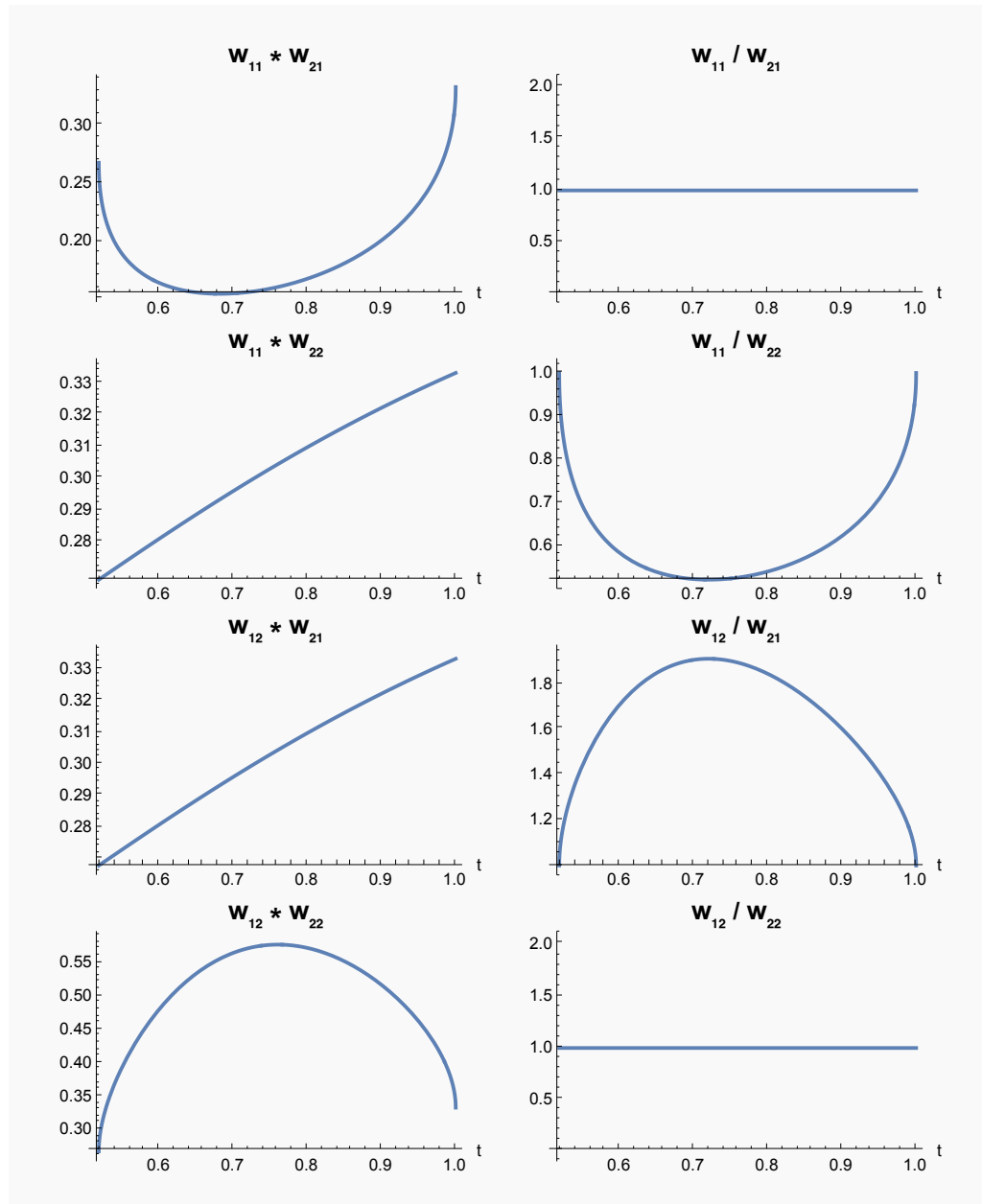
$$w_{2,1}(z) = \frac{(3 + \sqrt{3})z - \sqrt{2} \sqrt{-(3 + \sqrt{3})z^4 + 6z^2 + \sqrt{3} - 3}}{(3 + \sqrt{3})z^2 + 2\sqrt{3}}$$

$$w_{2,2}(z) = \frac{\sqrt{2} \sqrt{-(3 + \sqrt{3})z^4 + 6z^2 + \sqrt{3} - 3} + (3 + \sqrt{3})z}{(3 + \sqrt{3})z^2 + 2\sqrt{3}}$$

Out[5515]=

===== Plots of  $w_1(z)w_2(z)$   
and  $w_1(z)/w_2(z)$  For All Pairs =====

Checking that all four pairs –  $(w_{11}(z), w_{21}(z))$ ,  $(w_{11}(z), w_{22}(z))$ ,  $(w_{12}(z), w_{21}(z))$ , and  $(w_{12}(z), w_{22}(z))$  – does NOT satisfy  $w_1(z)/w_2(z) = c$  and  $w_1(z)w_2(z) = c$  when  $c \neq +1$ .



Out[5516]=

===== FLEXIBILITY  
(Double Checking) =====

Out[5518]=

**Solution 1:**

$$P_1(w_1, z) = 0$$

$$P_2(w_1, t) = 0$$

$$P_3(w_2, t) = 0$$

$$P_4(w_2, z) = 0$$

Out[5520]=

**Solution 2:**

$$P_1(w_1, z) = 0$$

$$P_2(w_1, t) = 0$$

$$P_3(w_2, t) = 0$$

$$P_4(w_2, z) = 0$$

Out[5527]=

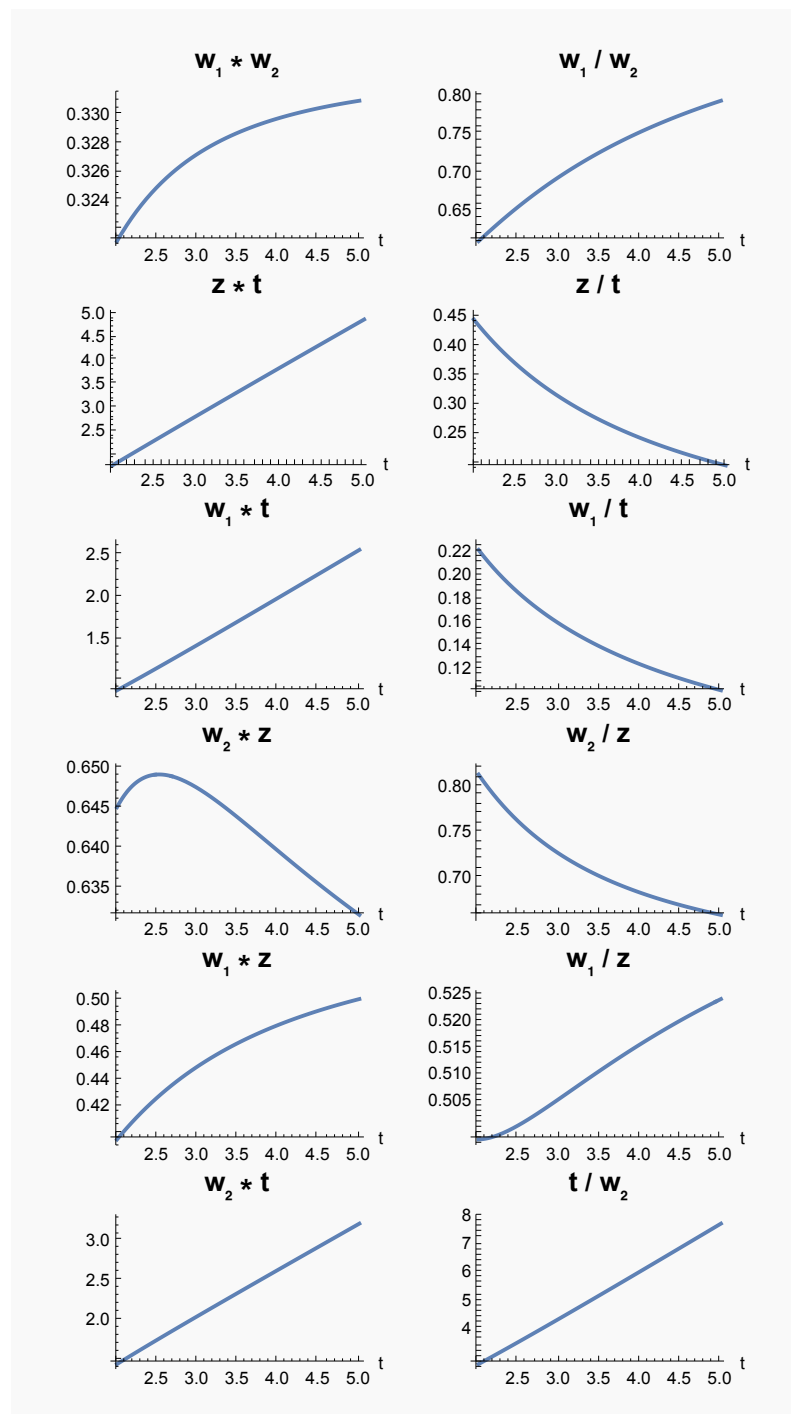
**===== NOT LINEAR COMPOUND =====**

Above we consider the first pair of equations ( $P_2(w_1, t) = 0$  and  $P_3(w_2, t) = 0$ ). Solving them as quadratic equations in  $w_1$  and  $w_2$ , respectively we parametrize the solutions by the first three expressions in Solutions 1 and 2 in a neighborhood of any point  $(w_1, t, w_2)$  such that  $2 + 6t^2 + 3t^4 \neq 0$ . Here, we choose any continuous branch of the square root in this neighborhood, and the signs in  $\pm$  need Not agree (this means we consider all 4 pairs we describe above). We conclude that one one-dimensional irreducible component of the solution set of the first pair of Bricard's equations satisfies  $w_1/w_2 = -1$ , and NO other component satisfies  $w_1/w_2 = \text{const}$  NOR  $w_1 w_2 = \text{const}$ .

Analogously, one one-dimensional irreducible component of the solution set of the other pair of equations ( $P_1(w_1, z) = 0$  and  $P_4(w_2, z) = 0$ ) satisfies  $w_1/w_2 = +1$ , and NO other component satisfies  $w_1/w_2 = \text{const}$  NOR  $w_1 w_2 = \text{const}$ . As a result, NO component of the solution set of all four equations satisfies  $w_1/w_2 = \text{const}$  NOR  $w_1 w_2 = \text{const}$ . So, our example does not belong to the linear compound class, even after switching the boundary strips.

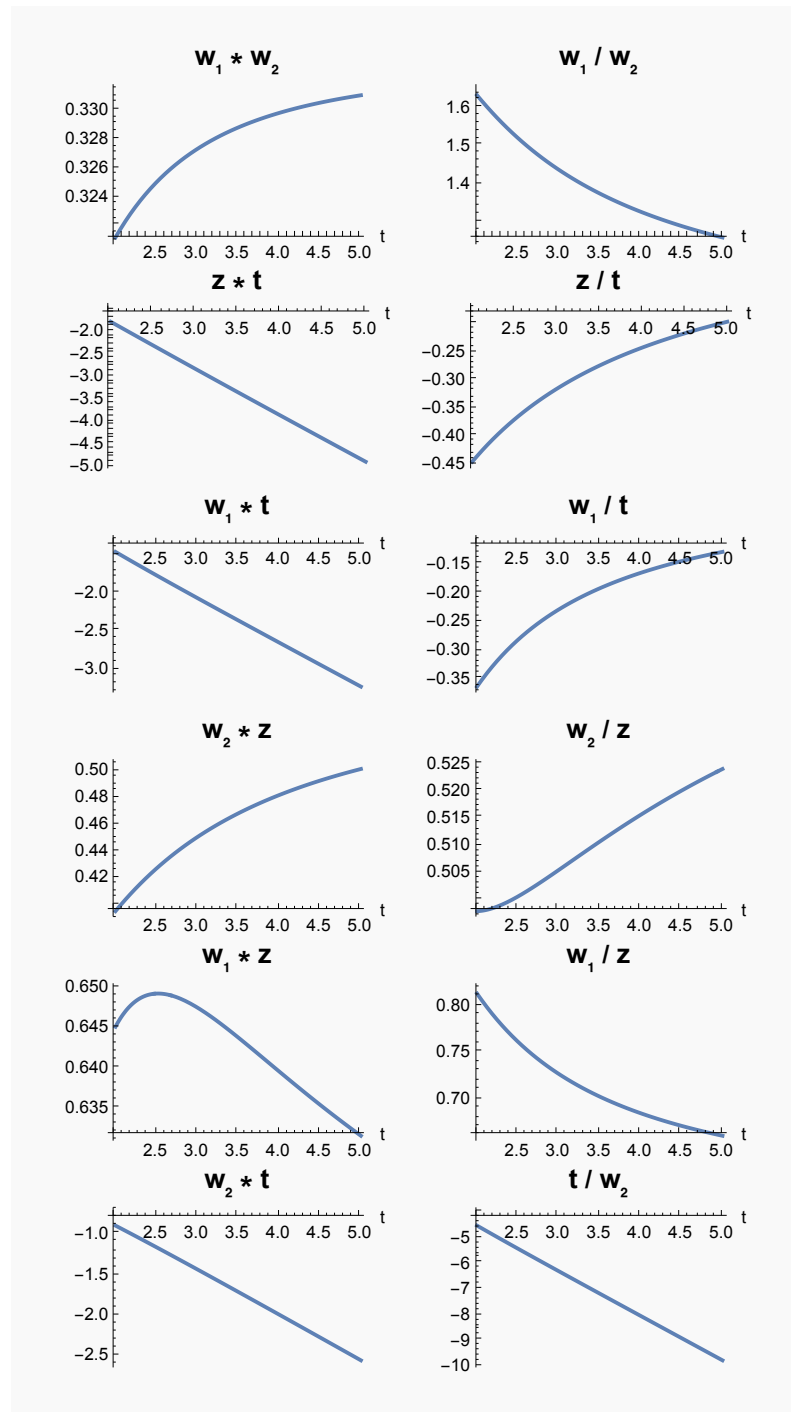
Below, we also present the plots of functions  $w_1 w_2$ ,  $w_1/w_2$ ,  $zt$ ,  $z/t$ ,  $w_1 t$ ,  $w_1/t$ ,  $w_2 z$ ,  $w_2/z$ ,  $w_1 z$ ,  $w_1/z$ ,  $w_2 t$ , and  $t/w_2$ .

**Solution 1:**



Out[5530]=

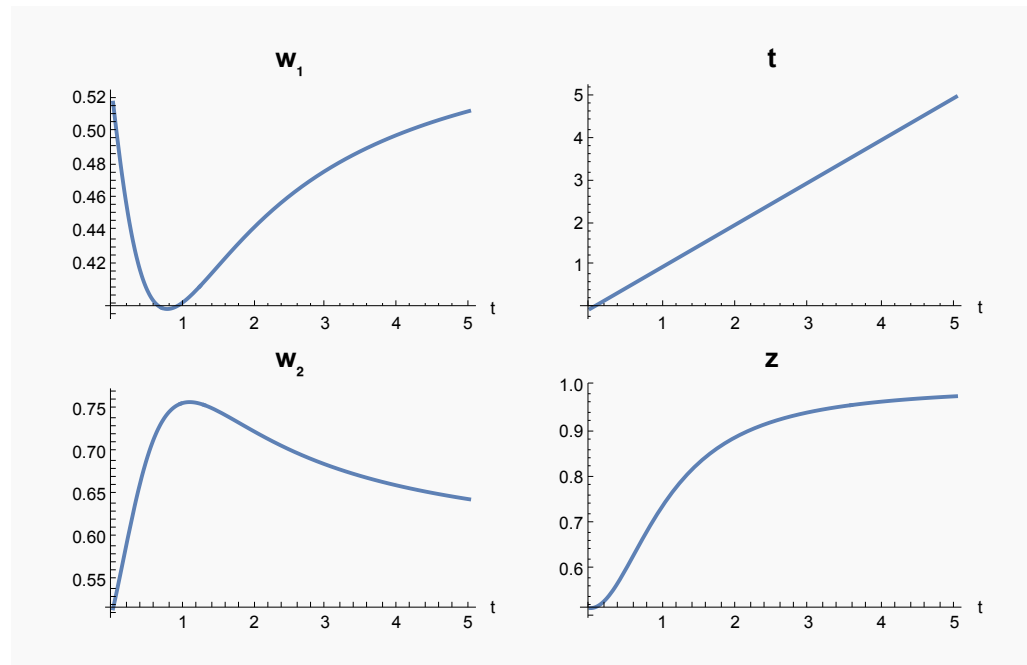
## Solution 2:



Out[5535]=

===== NOT TRIVIAL (FLEXION 1) =====

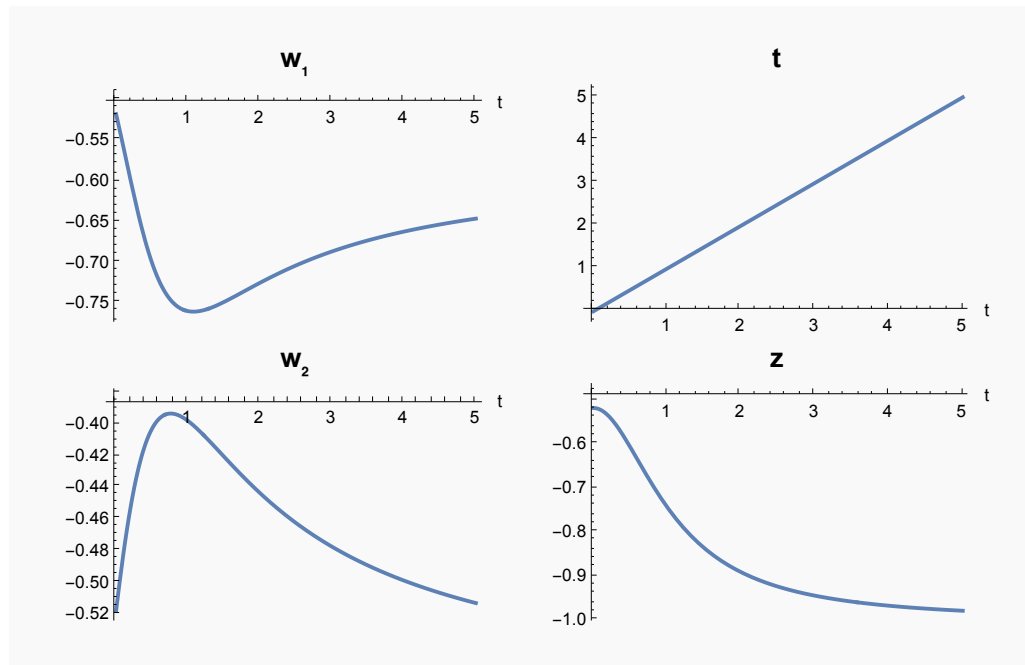
This configuration does not belong to the trivial class – even after switching the boundary strips – since none of the functions  $w_1$ ,  $t$ ,  $w_2$ , or  $z$  is constant.



Out[5538]=

===== NOT TRIVIAL (FLEXION 2) =====

This configuration does not belong to the trivial class – even after switching the boundary strips – since none of the functions  $w_1$ ,  $t$ ,  $w_2$ , or  $z$  is constant.



Out[5543]=

===== NOT CONIC & NOT CHIMERA & NOT  
LINEAR CONJUGATE & NOT ISOGONAL=====

Condition (N.0) is satisfied for all  $i=1,\dots,4$   
 $\Rightarrow$  NOT equimodular-conic, NOT chimera, NOT isogonal  
 and NOT linear conjugate. Applying any boundary-strip  
 switch still preserves (N.0), so no conic, no chimera,  
 no isogonal and no linear conjugate form emerges.

Out[5544]=

CONDITION (N.0) AFTER SWITCHING BOUNDARY STRIPS

**Right:** Condition (N.0) is still satisfied.

**Left:** Condition (N.0) is still satisfied.

**Lower:** Condition (N.0) is still satisfied.

**Upper:** Condition (N.0) is still satisfied.

**Right + Left:** Condition (N.0) is still satisfied.

**Right + Lower:** Condition (N.0) is still satisfied.

**Right + Upper:** Condition (N.0) is still satisfied.

**Left + Lower:** Condition (N.0) is still satisfied.

**Left + Upper:** Condition (N.0) is still satisfied.

**Lower + Upper:** Condition (N.0) is still satisfied.

**Right + Left + Lower:** Condition (N.0) is still satisfied.

**Right + Left + Upper:** Condition (N.0) is still satisfied.

**Right + Lower + Upper:** Condition (N.0) is still satisfied.

**Left + Lower + Upper:** Condition (N.0) is still satisfied.

**Right + Left + Lower + Upper:** Condition (N.0) is still satisfied.

Out[5545]=

===== ORTHOGONALITY CHECK =====

$\cos(\alpha_i) \cdot \cos(\gamma_i) \neq \cos(\beta_i) \cdot \cos(\delta_i)$  for  
 at least one  $i = 1, \dots, 4 \Rightarrow$  NOT orthodiagonal.  
 Switching boundary strips does not correct this.

**Initial anglesDeg (no switches):**

-> Condition met: At least one difference is non-zero.

Out[5549]=

NON-ORTHOGONALITY CHECK AFTER SWITCHING BOUNDARY STRIPS

**Right:** Condition met (at least one difference is non-zero).

**Left:** Condition met (at least one difference is non-zero).

**Lower:** Condition met (at least one difference is non-zero).

**Upper:** Condition met (at least one difference is non-zero).

**Right + Left:** Condition met (at least one difference is non-zero).

**Right + Lower:** Condition met (at least one difference is non-zero).

**Right + Upper:** Condition met (at least one difference is non-zero).

**Left + Lower:** Condition met (at least one difference is non-zero).

**Left + Upper:** Condition met (at least one difference is non-zero).

**Lower + Upper:** Condition met (at least one difference is non-zero).

**Right + Left + Lower:** Condition met (at least one difference is non-zero).

**Right + Left + Upper:** Condition met (at least one difference is non-zero).

**Right + Lower + Upper:** Condition met (at least one difference is non-zero).

**Left + Lower + Upper:** Condition met (at least one difference is non-zero).

**Right + Left + Lower + Upper:**  
 Condition met (at least one difference is non-zero).

Out[5550]=

===== CONJUGATE-MODULAR CHECK =====

$M_1 = M_2 = M_3 = M_4 = M$  and  $M \neq 2 \Rightarrow$  NOT  
 conjugate-modular. Boundary-strip switches preserve this.



**Initial anglesDeg (no switches):**

-> Condition met: All  $M_i$  are equal and  $M \neq 2$ .

Out[5554]=

CONJUGATE-MODULAR CHECK AFTER SWITCHING BOUNDARY STRIPS

**Right:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Left:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Lower:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Left:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Lower:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Left + Lower:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Left + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Lower + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Left + Lower:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Left + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Lower + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Left + Lower + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).  
**Right + Left + Lower + Upper:** Condition met (All  $M_i$  are equal and  $M \neq 2$ ).