

A polynomial counterexample to weak Markus–Yamabe in dimension eight

Joshua Christl

joshua.plus

Abstract. We give an explicit degree-eight polynomial vector field $X : \mathbb{R}^8 \rightarrow \mathbb{R}^8$ with Jacobian characteristic polynomial $(\lambda + 1)^8$ at every point and three distinct rational zeros. Thus X is a noninjective Hurwitz map. Appending coordinates $-u$ gives counterexamples in every dimension $n \geq 8$. The proposed improvement is from dimension fourteen to dimension eight. The certificate is checked in Lean, including the passage from polynomial partial derivatives to actual real coordinate derivatives. The complete project source needed for that certificate is printed in the appendix.

1 The previous paper and the proposed improvement

Castañeda, Honorato and Valenzuela–Henríquez’s August 5, 2026 preprint, *The weak Markus–Yamabe conjecture fails in dimension 14*, constructs a degree-seven counterexample on $\mathbb{R}^{14}$ with three rational zeros and Jacobian spectrum $\{-1\}$ everywhere. Its abstract explicitly leaves dimensions $3 \leq n \leq 13$ open.¹

	August paper	Present construction
Smallest dimension supplied	14	8
Polynomial degree	7	8
Jacobian spectrum everywhere	$\{-1\}$	$\{-1\}$
Distinct rational zeros exhibited	3	3
Dimensions covered by padding	$n \geq 14$	$n \geq 8$

This improves a dimension bound; it does not refute the paper’s theorem or improve its separate cubic-degree construction. Dimensions three through seven are not settled here. The claim concerns the *weak* conjecture about injectivity, rather than the classical formulation about global attraction.

Status of the claim. The explicit mathematical certificate below has been checked. The accompanying literature audit found no prior eight-dimensional construction as of September 10, 2026. Historical priority and independent expert review remain separate from the formal verification; no publication or acceptance is claimed by this note.

2 What a counterexample must establish

A C^1 map $X : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is *Hurwitz* when every complex eigenvalue of $JX(p)$ has negative real part for every $p \in \mathbb{R}^n$. The weak conjecture says that such a map must be injective. A counterexample therefore needs an explicit map, an everywhere-valid eigenvalue certificate, and two distinct inputs with equal outputs. A single constant matrix cannot serve: negative-real-part eigenvalues exclude zero, so that matrix would be invertible. The phenomenon here is global noninjectivity despite local invertibility everywhere.

¹Á. Castañeda, G. Honorato and F. Valenzuela–Henríquez, *The weak Markus–Yamabe conjecture fails in dimension 14*, arXiv:2608.05392v1, 5 August 2026. <https://arxiv.org/abs/2608.05392>. Relevant locations: Conjecture 1.1, Theorem 1.3, and Sections 3–5.

3 The explicit counterexample

Always use the input and output coordinate order (a, b, c, d, e, x, y, z) . Define

$$X(a, b, c, d, e, x, y, z) = \begin{pmatrix} -a - 2x^3y^2 + \frac{1}{2}x^3z \\ -b + d + 15x^2y^3 - 6x^2yz \\ -c + e + 9x^2y^4 - 3x^2y^2z \\ -d + 12x^3y^4 - 3x^3y^2z \\ -e + 4x^3y^5 - x^3y^3z \\ a + \frac{3}{2}x^2y - x \\ b - 3xz - y \\ c + 5xy^3 - 3xyz - z - \frac{1}{4} \end{pmatrix}. \quad (1)$$

Its degree is eight: the monomial $4x^3y^5$ occurs, and no monomial has higher ordinary total degree. Define

$$\begin{aligned} p_0 &= (0, 0, 0, 0, 0, 0, 0, -\frac{1}{4}), \\ p_+ &= (\frac{13}{4}, 45, -\frac{297}{8}, -\frac{351}{8}, \frac{351}{16}, 1, -\frac{3}{2}, \frac{31}{2}), \\ p_- &= (-\frac{13}{4}, -45, -\frac{297}{8}, \frac{351}{8}, \frac{351}{16}, -1, \frac{3}{2}, \frac{31}{2}). \end{aligned}$$

Theorem 1. *The polynomial map in (1) is smooth and Hurwitz, but is not injective. More precisely,*

$$\det(\lambda I_8 - JX(p)) = (\lambda + 1)^8 \quad (p \in \mathbb{R}^8, \lambda \in \mathbb{C}), \quad (2)$$

and $X(p_0) = X(p_+) = X(p_-) = 0$.

Proof. All components are polynomials, hence smooth. Differentiation and determinant expansion give (2); the complete matrix and exact Lean certificate are in Appendix A, and an explanatory derivation follows below. A characteristic root must therefore be -1 , whose real part is negative. Substitution of the three rational points into (1) gives zero in each component. Their x coordinates are respectively $0, 1, -1$, so they are pairwise distinct. Consequently the map is not injective. $\square$

Every higher dimension. For $n = 8 + k$, put $\widehat{X}(p, u) = (X(p), -u)$, where $u \in \mathbb{R}^k$. Its Jacobian is block diagonal with blocks $JX(p)$ and $-I_k$. Its characteristic polynomial is $(\lambda + 1)^{8+k}$, and the three points $(p_0, 0), (p_+, 0), (p_-, 0)$ still have the same image. This proves failure in every dimension $n \geq 8$.

4 How eight coordinates were obtained

The starting point is the three-dimensional Keller map reproduced in Section 3 of the August paper. A Keller map has constant nonzero Jacobian determinant. After normalizing the component order and scale, and making the source shear $z_{\text{old}} = z - 4y^2$, write the resulting map as

$$F_i(v) = v_i + \sum_{j=1}^{\ell_i} H_{i,j}(v), \quad v = (x, y, z), \quad (\ell_1, \ell_2, \ell_3) = (2, 3, 3).$$

The terms are explicitly

$$\begin{aligned} H_{1,1} &= -\frac{3}{2}x^2y, & H_{1,2} &= 2x^3y^2 - \frac{1}{2}x^3z, \\ H_{2,1} &= 3xz, & H_{2,2} &= -15x^2y^3 + 6x^2yz, \\ H_{2,3} &= -12x^3y^4 + 3x^3y^2z, & & \\ H_{3,1} &= -5xy^3 + 3xyz, & H_{3,2} &= -9x^2y^4 + 3x^2y^2z, \\ H_{3,3} &= -4x^3y^5 + x^3y^3z. & & \end{aligned}$$

The source shear has determinant one and the normalized original map has determinant one. Thus $\det JF = 1$. Alternatively this identity is checked by differentiating the three displayed sums and expanding their determinant.

Give x, y, z weights $w = (1, 1, 2)$. Every $H_{i,j}$ has weighted degree $w_i + 2j$:

$$H_{i,j}(sx, sy, s^2z) = s^{w_i+2j}H_{i,j}(x, y, z).$$

Weights organize the terms differently from ordinary total degree. The source shear is what removes the obstructing term of relative weighted degree zero.

Introduce a chain $q_{i,1}, \dots, q_{i,\ell_i}$ for each component, with $q_{i,1} = v_i$ and $q_{i,\ell_i+1} = 0$. Take $\beta = (0, 0, -1/4)$ and define chain equations

$$\begin{aligned} Y_{i,1} &= \beta_i - q_{i,1} + q_{i,2} - H_{i,1}(v), \\ Y_{i,j} &= -q_{i,j} + q_{i,j+1} - H_{i,j}(v) \quad (j \geq 2). \end{aligned}$$

Identify the chain coordinates as

$$(q_{1,1}, q_{1,2}) = (x, a), \quad (q_{2,1}, q_{2,2}, q_{2,3}) = (y, b, d), \quad (q_{3,1}, q_{3,2}, q_{3,3}) = (z, c, e).$$

Reordering both coordinates and equations as (a, b, c, d, e, x, y, z) gives exactly (1). There are $2 + 3 + 3 = 8$ coordinates.

Why the collision survives. Solving the auxiliary equations gives $q_{i,j} = -\sum_{m=j}^{\ell_i} H_{i,m}(v)$ for $j \geq 2$. The remaining equations become $\beta - F(v) = 0$. The three base points are $(0, 0, -1/4)$, $(1, -3/2, 31/2)$ and $(-1, 3/2, 31/2)$, with common image β . Back substitution gives precisely the displayed p_0, p_+, p_- .

5 Why the Jacobian spectrum is constantly minus one

This argument explains the construction. The direct Lean determinant proof in the appendix is independent of it.

Step 1: a constant determinant. In chain order write $JY = -I_8 + M$. The five auxiliary equations are affine in the auxiliary coordinates; their Jacobian block A is triangular with diagonal -1 . Eliminating these equations leaves $\beta - F(v)$. If the full Jacobian is written in auxiliary/base block order as

$$JY = \begin{pmatrix} A & B \\ C & D \end{pmatrix},$$

then its Schur complement satisfies $D - CA^{-1}B = -JF(v)$. Hence

$$\det JY = \det A \det(-JF) = (-1)^5(-1)^3 = 1, \quad \det(I_8 - M) = 1.$$

Reordering coordinates and equations simultaneously does not change these determinants.

Step 2: weighted scaling. Assign $\nu_{i,j} = w_i + 2(j-1)$ to chain coordinate $q_{i,j}$, and let $D_s = \text{diag}(s^{\nu_{i,j}})$. The entries of M depend only on v . Its nonzero entries are chain constants 1, and terms $-\partial H_{i,j}/\partial v_r$. For a derivative entry, its weighted degree is $w_i + 2j - w_r = 2 + \nu_{i,j} - \nu_{r,1}$. For a chain constant in column $(i, j+1)$, this expression is zero. Consequently, for $s \neq 0$,

$$M(sx, sy, s^2z) = s^2 D_s M(x, y, z) D_s^{-1}.$$

Combining this with Step 1 gives

$$\det(I_8 - s^2 M(v)) = 1 \quad (s \neq 0).$$

For fixed real v , the polynomial $r \mapsto \det(I_8 - rM(v)) - 1$ vanishes at every positive real $r = s^2$. It is therefore identically zero. It follows that $\det(tI_8 - M(v)) = t^8$, initially for $t \neq 0$ and then for all t by polynomial identity. Finally,

$$\det(\lambda I_8 - JY) = \det((\lambda + 1)I_8 - M) = (\lambda + 1)^8.$$

This is (2); the identity holds over $\mathbb{C}$ because it is a polynomial identity with real coefficients.

What changed from the August construction. The existing method uses ordinary homogeneous pieces. The present construction uses the displayed source shear and weighted pieces, retaining only chain lengths $(2, 3, 3)$. The derivative weights make the same spectral mechanism work with fewer coordinates. The explicit map and its certificate establish the improvement even without a general theorem about weighted chains.

6 Lean verification and exact scope

The final Lean theorem is:

```
theorem weak_markus_yamabe_false_in_eight :
  ∃ F : (Fin 8 → ℝ) → (Fin 8 → ℝ),
    ContDiff ℝ (τ : ℕ) F ∧ IsHurwitz F ∧ ¬ Function.Injective F
```

Here `IsHurwitz` quantifies over every real input and every complex characteristic root of the *actual* coordinate-derivative matrix. Smoothness ensures these partial derivatives are the classical Jacobian. The proof uses the explicit polynomial witness, rather than postulating a matrix with the desired spectrum.

Module	Role
<code>MarkusEight</code>	Formula, displayed Jacobian, exact characteristic determinant.
<code>MarkusPolynomial</code>	Multivariate polynomial representation, all formal partial derivatives, evaluation and analytic derivative bridge.
<code>CalculusLibrary</code>	Evaluated formal partial derivatives are real coordinate derivatives.
<code>VectorLibrary</code>	Reduction lemmas for explicit eight-coordinate vectors.
<code>MarkusResult</code>	Three zeros, distinctness, smoothness, Hurwitz property and final existence theorem.
<code>Library</code>	Shared Mathlib import and project-owned supporting lemmas.

The kernel-checked determinant proof uses `eval_det` and `ring`. It does not assume a symbolic algebra program’s answer as an axiom. The independent SymPy script differentiates the map and verifies that $JX + I_8$ has characteristic polynomial t^8 , then checks all three rational zeros.

Replay checked on September 10, 2026. The named Lean build and axiom audit completed successfully; the exact SymPy replay passed. Lean and Mathlib are pinned to version 4.33.1. The audited declarations depend only on `propext`, `Classical.choice`, and `Quot.sound`; they do not use `sorryAx` or a custom axiom.

The existence theorem formally closes the dimension-eight analytic claim. The ordinary-degree count and the padding argument for $n > 8$ are the elementary written arguments above. The general weighted-chain construction and the historical novelty claim are not additional conclusions of the Lean theorem.

7 Reproducing the certificate

The accompanying source archive contains the six complete Lean modules, pinned Lake configuration, an audit file, the exact symbolic checker, and source hashes. With Lean/elan installed, run in its source directory:

```
lake exe cache get
lake build ProofExplorer.Jacobian.MarkusResult
lake env lean Audit.lean
# Optional independent calculation, with SymPy installed:
python3 check_symbolic.py
```

The explicit target is intentional: the configuration’s default target belongs to the larger Proof Explorer repository. No omitted project module is needed for this named target. The appendix prints the exact source snapshot; wrapped lines and line numbers are presentation aids, so use the archive for compilation.

8 Source and claim checklist

1. **Previous result.** Castañeda, Honorato and Valenzuela–Henríquez, *The weak Markus–Yamabe conjecture fails in dimension 14*, arXiv:2608.05392v1, 5 August 2026. <https://arxiv.org/html/2608.05392v1>. The three-dimensional starting map is attributed there to Alpöge.
2. **Proposed contribution.** The explicit eight-dimensional polynomial field, obtained from a source shear and weighted chain lengths $(2, 3, 3)$, with its exact derivative, determinant and collision certificate.
3. **Formal source.** The appendix is a snapshot of `ProofExplorer/Jacobian/` in the Proof Explorer repository. Its six files are reproduced without proof edits. The larger shared `Library.lean` contains coefficient lemmas unrelated to this result; it is included in full to preserve the exact import closure.
4. **What remains outside the claim.** No result for dimensions three through seven, no claim of the smallest possible degree, and no independent priority or peer-review certification.

Audit command and output

```
import ProofExplorer.Jacobian.MarkusResult

#print axioms ProofExplorer.Jacobian.MarkusEight.characteristic_determinant
#print axioms ProofExplorer.Jacobian.MarkusEight.actual_jacobian
#print axioms ProofExplorer.Jacobian.MarkusEight.three_zeros
#print axioms ProofExplorer.Jacobian.MarkusEight.weak_markus_yamabe_false_in_eight

'ProofExplorer.Jacobian.MarkusEight.characteristic_determinant' depends on axioms: [propext,
Classical.choice,
Quot.sound]
'ProofExplorer.Jacobian.MarkusEight.actual_jacobian' depends on axioms: [propext, Classical.choice,
Quot.sound]
'ProofExplorer.Jacobian.MarkusEight.three_zeros' depends on axioms: [propext, Classical.choice, Quot.sound]
'ProofExplorer.Jacobian.MarkusEight.weak_markus_yamabe_false_in_eight' depends on axioms: [propext,
Classical.choice,
Quot.sound]
```

A MarkusEight

ProofExplorer/Jacobian/MarkusEight.lean

```

1 import ProofExplorer.Jacobian.Library
2
3 /-! An eight-dimensional polynomial Hurwitz candidate, ordered (a,b,c,d,e,x,y,z).
4 The determinant and collision certificates are independent of the chain argument. -/
5 namespace ProofExplorer.Jacobian.MarkusEight
6 noncomputable section
7 variable {K : Type*} [Field K] [CharZero K]
8
9 def field (a b c d e x y z : K) : Fin 8 → K :=
10   ![-a - 2*x^3*y^2 + x^3*z/2,
11     -b + d + 15*x^2*y^3 - 6*x^2*y*z,
12     -c + e + 9*x^2*y^4 - 3*x^2*y^2*z,
13     -d + 12*x^3*y^4 - 3*x^3*y^2*z,
14     -e + 4*x^3*y^5 - x^3*y^3*z,
15     a + 3*x^2*y/2 - x,
16     b - 3*x*z - y,
17     c + 5*x*y^3 - 3*x*y*z - z - 1/4]
18
19 def jacobian (x y z : K) : Matrix (Fin 8) (Fin 8) K :=
20   ![-1, 0, 0, 0, 0, -6*x^2*y^2 + 3*x^2*z/2, -4*x^3*y, x^3/2;
21     0, -1, 0, 1, 0, 30*x*y^3 - 12*x*y*z, 45*x^2*y^2 - 6*x^2*z, -6*x^2*y;
22     0, 0, -1, 0, 1, 18*x*y^4 - 6*x*y^2*z, 36*x^2*y^3 - 6*x^2*y*z, -3*x^2*y^2;
23     0, 0, 0, -1, 0, 36*x^2*y^4 - 9*x^2*y^2*z, 48*x^3*y^3 - 6*x^3*y*z, -3*x^3*y^2;
24     0, 0, 0, 0, -1, 12*x^2*y^5 - 3*x^2*y^3*z, 20*x^3*y^4 - 3*x^3*y^2*z, -x^3*y^3;
25     1, 0, 0, 0, 0, 3*x*y - 1, 3*x^2/2, 0;
26     0, 1, 0, 0, 0, -3*z, -1, -3*x;
27     0, 0, 1, 0, 0, 5*y^3 - 3*y*z, 15*x*y^2 - 3*x*z, -3*x*y - 1]
28
29 def characteristicMatrix (t x y z : K) : Matrix (Fin 8) (Fin 8) K :=
30   ![t + 1, 0, 0, 0, 0, 6*x^2*y^2 - 3*x^2*z/2, 4*x^3*y, -x^3/2;
31     0, t + 1, 0, -1, 0, -30*x*y^3 + 12*x*y*z, -45*x^2*y^2 + 6*x^2*z, 6*x^2*y;
32     0, 0, t + 1, 0, -1, -18*x*y^4 + 6*x*y^2*z, -36*x^2*y^3 + 6*x^2*y*z, 3*x^2*y^2;
33     0, 0, 0, t + 1, 0, -36*x^2*y^4 + 9*x^2*y^2*z, -48*x^3*y^3 + 6*x^3*y*z, 3*x^3*y^2;
34     0, 0, 0, 0, t + 1, -12*x^2*y^5 + 3*x^2*y^3*z, -20*x^3*y^4 + 3*x^3*y^2*z, x^3*y^3;
35     -1, 0, 0, 0, 0, t - 3*x*y + 1, -3*x^2/2, 0;
36     0, -1, 0, 0, 0, 3*z, t + 1, 3*x;
37     0, 0, -1, 0, 0, -5*y^3 + 3*y*z, -15*x*y^2 + 3*x*z, t + 3*x*y + 1]
38
39 set_option maxRecDepth 10000 in
40 set_option maxHeartbeats 0 in
41 theorem characteristic_determinant (t x y z : K) :
42   (characteristicMatrix t x y z).det = (t+1)^8 := by
43   unfold characteristicMatrix
44   eval_det
45   ring
46
47 end
48 end ProofExplorer.Jacobian.MarkusEight

```

B MarkusPolynomial

ProofExplorer/Jacobian/MarkusPolynomial.lean

```

1 import ProofExplorer.Jacobian.MarkusEight
2 import ProofExplorer.Jacobian.CalculusLibrary
3
4 namespace ProofExplorer.Jacobian.MarkusEight
5 open MvPolynomial
6 noncomputable section
7 set_option maxHeartbeats 0
8 set_option maxRecDepth 10000
9 variable {K : Type*} [Field K] [CharZero K]
10
11 def polynomialMap : Fin 8 → MvPolynomial (Fin 8) K :=
12   ![(C (-1)) * (X 0) + (C (1/2)) * (X 7) * ((X 5) ^ 3) + ((C (-2)) * ((X 5) ^ 3) * ((X 6) ^ 2)),
13     ((X 3) + ((C (-1)) * (X 1)) + ((C (15)) * ((X 5) ^ 2) * ((X 6) ^ 3) + ((C (-6)) * (X 6) * (X 7) * ((X
14       5) ^ 2))),
15     ((X 4) + ((C (-1)) * (X 2)) + ((C (9)) * ((X 5) ^ 2) * ((X 6) ^ 4) + ((C (-3)) * (X 7) * ((X 5) ^ 2) *
16       ((X 6) ^ 2))),
17     (((C (-1)) * (X 3)) + ((C (12)) * ((X 5) ^ 3) * ((X 6) ^ 4) + ((C (-3)) * (X 7) * ((X 5) ^ 3) * ((X 6)
18       ^ 2))),
19     (((C (-1)) * (X 4)) + ((C (4)) * ((X 5) ^ 3) * ((X 6) ^ 5) + ((C (-1)) * (X 7) * ((X 5) ^ 3) * ((X 6) ^
20       3))),
21     ((X 0) + ((C (-1)) * (X 5)) + ((C (3/2)) * (X 6) * ((X 5) ^ 2))),
22     ((X 1) + ((C (-1)) * (X 6)) + ((C (-3)) * (X 5) * (X 7))),
23     ((C (-1/4)) + (X 2) + ((C (-1)) * (X 7)) + ((C (5)) * (X 5) * ((X 6) ^ 3) + ((C (-3)) * (X 5) * (X 6) *
24       (X 7)))]
25
26 def polynomialJacobian : Matrix (Fin 8) (Fin 8) (MvPolynomial (Fin 8) K) :=
27   ![(C (-1)), (C (0)), (C (0)), (C (0)), (C (0)), ((C (-6)) * ((X 5) ^ 2) * ((X 6) ^ 2) + ((C (3/2)) * (X
28     7) * ((X 5) ^ 2))), ((C (-4)) * (X 6) * ((X 5) ^ 3)), ((C (1/2)) * ((X 5) ^ 3));
29     (C (0)), (C (-1)), (C (0)), (C (1)), (C (0)), (((C (30)) * (X 5) * ((X 6) ^ 3) + ((C (-12)) * (X 5) *
30     (X 6) * (X 7))), (((C (-6)) * (X 7) * ((X 5) ^ 2) + ((C (45)) * ((X 5) ^ 2) * ((X 6) ^ 2))), ((C (-6))
31     * (X 6) * ((X 5) ^ 2));
32     (C (0)), (C (0)), (C (-1)), (C (0)), (C (1)), (((C (18)) * (X 5) * ((X 6) ^ 4) + ((C (-6)) * (X 5) * (X
33     7) * ((X 6) ^ 2))), (((C (36)) * ((X 5) ^ 2) * ((X 6) ^ 3) + ((C (-6)) * (X 6) * (X 7) * ((X 5) ^ 2))),
34     ((C (-3)) * ((X 5) ^ 2) * ((X 6) ^ 2));
35     (C (0)), (C (0)), (C (0)), (C (-1)), (C (0)), (((C (36)) * ((X 5) ^ 2) * ((X 6) ^ 4) + ((C (-9)) * (X
36     7) * ((X 5) ^ 2) * ((X 6) ^ 2))), (((C (48)) * ((X 5) ^ 3) * ((X 6) ^ 3) + ((C (-6)) * (X 6) * (X 7) *
37     ((X 5) ^ 3))), ((C (-3)) * ((X 5) ^ 3) * ((X 6) ^ 2));
38     (C (0)), (C (0)), (C (0)), (C (0)), (C (-1)), (((C (12)) * ((X 5) ^ 2) * ((X 6) ^ 5) + ((C (-3)) * (X
39     7) * ((X 5) ^ 2) * ((X 6) ^ 3))), (((C (20)) * ((X 5) ^ 3) * ((X 6) ^ 4) + ((C (-3)) * (X 7) * ((X 5) ^
40     3) * ((X 6) ^ 2))), ((C (-1)) * ((X 5) ^ 3) * ((X 6) ^ 3));
41     (C (1)), (C (0)), (C (0)), (C (0)), (C (0)), ((C (-1)) + ((C (3)) * (X 5) * (X 6))), ((C (3/2)) * ((X 5)
42     ^ 2)), (C (0));
43     (C (0)), (C (1)), (C (0)), (C (0)), (C (0)), ((C (-3)) * (X 7)), (C (-1)), ((C (-3)) * (X 5));
44     (C (0)), (C (0)), (C (1)), (C (0)), (C (0)), (((C (5)) * ((X 6) ^ 3) + ((C (-3)) * (X 6) * (X 7))),
45     (((C (-3)) * (X 5) * (X 7)) + ((C (15)) * (X 5) * ((X 6) ^ 2))), ((C (-1)) + ((C (-3)) * (X 5) * (X
46     6)))]
47
48 set_option maxHeartbeats 0 in
49 theorem polynomial_jacobian_entries (i j : Fin 8) :
50   pderiv j (polynomialMap (K := K) i) = polynomialJacobian i j := by
51   apply MvPolynomial.funext
52   intro u
53   fin_cases i <=> fin_cases j <=>
54     simp [polynomialMap, polynomialJacobian, Derivation.leibniz, smul_eq_mul,
55       Library.mv_pderiv_numeral, map_ofNat] <=> ring
56
57 theorem polynomialMap_evaluation (p : Fin 8 → K) (i : Fin 8) :
58   eval p (polynomialMap i) = field (p 0) (p 1) (p 2) (p 3) (p 4) (p 5) (p 6) (p 7) i := by
59   fin_cases i <=> simp [polynomialMap, field] <=> ring
60
61

```

```

44 theorem polynomialJacobian_evaluation (p : Fin 8 → K) (i j : Fin 8) :
45   eval p (polynomialJacobian i j) = jacobian (p 5) (p 6) (p 7) i j := by
46   fin_cases i <=> fin_cases j <=> simp [polynomialJacobian, jacobian] <=> ring
47
48 /-- Analytic coordinate derivatives of the actual real polynomial field. -/
49 theorem analytic_jacobian (p : Fin 8 → ℝ) (i j : Fin 8) :
50   HasDerivAt (fun t => let q := Function.update p j t
51     field (q 0) (q 1) (q 2) (q 3) (q 4) (q 5) (q 6) (q 7) i)
52     (jacobian (p 5) (p 6) (p 7) i j) (p j) := by
53   have h := CalculusLibrary.eval_hasDerivAt (polynomialMap i) p j
54   simpa only [polynomial_jacobian_entries, polynomialJacobian_evaluation,
55     polynomialMap_evaluation] using h
56
57 end
58 end ProofExplorer.Jacobian.MarkusEight

```

C MarkusResult

ProofExplorer/Jacobian/MarkusResult.lean

```

1 import ProofExplorer.Jacobian.MarkusPolynomial
2 import ProofExplorer.Jacobian.VectorLibrary
3
4 namespace ProofExplorer.Jacobian.MarkusEight
5 noncomputable section
6 variable {K : Type*} [Field K] [CharZero K]
7 set_option maxHeartbeats 0
8 set_option maxRecDepth 10000
9
10 /-- The real vector field on its eight-dimensional coordinate space. -/
11 def vectorField (p : Fin 8 → K) : Fin 8 → K :=
12   field (p 0) (p 1) (p 2) (p 3) (p 4) (p 5) (p 6) (p 7)
13
14 def zeroPoint : Fin 8 → K := ![0,0,0,0,0,0,0,-1/4]
15 def positivePoint : Fin 8 → K := ![13/4,45,-297/8,-351/8,351/16,1,-3/2,31/2]
16 def negativePoint : Fin 8 → K := ![-13/4,-45,-297/8,351/8,351/16,-1,3/2,31/2]
17
18 theorem three_zeros :
19   vectorField (zeroPoint (K := K)) = 0 ∧
20   vectorField (positivePoint (K := K)) = 0 ∧
21   vectorField (negativePoint (K := K)) = 0 := by
22   constructor
23   · funext i; fin_cases i <=> norm_num [vectorField, field, zeroPoint, Matrix.cons_val]
24   constructor
25   · funext i; fin_cases i <=> norm_num [vectorField, field, positivePoint, Matrix.cons_val]
26   · funext i; fin_cases i <=> norm_num [vectorField, field, negativePoint, Matrix.cons_val]
27
28 theorem three_points_distinct :
29   zeroPoint (K := K) ≠ positivePoint ∧ zeroPoint (K := K) ≠ negativePoint ∧
30   positivePoint (K := K) ≠ negativePoint := by
31   constructor
32   · intro h; have := congrFun h 5; norm_num [zeroPoint, positivePoint, Matrix.cons_val] at this
33   constructor
34   · intro h; have := congrFun h 5; norm_num [zeroPoint, negativePoint, Matrix.cons_val] at this
35   · intro h; have := congrFun h 5; norm_num [positivePoint, negativePoint, Matrix.cons_val] at this
36
37 theorem not_injective : ¬ Function.Injective (vectorField (K := K)) := by
38   intro h
39   exact three_points_distinct.1 (h (three_zeros.1.trans three_zeros.2.1.symm))
40
41 theorem characteristicMatrix_eq (t x y z : K) :
42   characteristicMatrix t x y z = Matrix.scalar (Fin 8) t - jacobian x y z := by
43   apply Matrix.ext
44   intro i j
45   fin_cases i <=> fin_cases j <=>
46     simp [characteristicMatrix, jacobian, Matrix.scalar, Matrix.diagonal] <=> ring
47
48 /-- Every complex characteristic root is exactly minus one. -/
49 theorem characteristic_root (t x y z : ℂ) :
50   (Matrix.scalar (Fin 8) t - jacobian x y z).det = 0 ↔ t = -1 := by
51   rw [← characteristicMatrix_eq, characteristic_determinant]
52   constructor
53   · intro h
54     have hz : t+1=0 := eq_zero_of_pow_eq_zero h
55     linear_combination hz
56   · intro h; simp [h]
57
58
59 /-- Matrix of actual real coordinate derivatives, for any map on ℝ^8. -/
60 def analyticJacobian (F : (Fin 8 → ℝ) → (Fin 8 → ℝ)) (p : Fin 8 → ℝ) :

```

```

61   Matrix (Fin 8) (Fin 8) ℝ :=
62   fun i j => deriv (fun t => F (Function.update p j t) i) (p j)
63
64   /-- The usual Hurwitz condition, expressed through complex characteristic roots. -/
65   def IsHurwitz (F : (Fin 8 → ℝ) → (Fin 8 → ℝ)) : Prop :=
66     ∀ p (t : ℂ),
67       (Matrix.scalar (Fin 8) t - (analyticJacobian F p).map Complex.ofReal).det = 0 → t.re < 0
68
69   theorem actual_jacobian (p : Fin 8 → ℝ) :
70     analyticJacobian vectorField p = jacobian (p 5) (p 6) (p 7) := by
71     apply Matrix.ext
72     intro i j
73     exact (analytic_jacobian p i j).deriv
74
75   theorem complexified_jacobian (x y z : ℝ) :
76     (jacobian x y z).map Complex.ofReal = jacobian (x : ℂ) (y : ℂ) (z : ℂ) := by
77     apply Matrix.ext
78     intro i j
79     fin_cases i <=> fin_cases j <=> simp [jacobian, Matrix.map_apply, Matrix.cons_val] <=> push_cast <=> ring
80
81   theorem vectorField_hurwitz : IsHurwitz vectorField := by
82     intro p t ht
83     rw [actual_jacobian, complexified_jacobian] at ht
84     have h := (characteristic_root t (p 5) (p 6) (p 7)).mp ht
85     rw [h]
86     norm_num
87
88   theorem vectorField_smooth : ContDiff ℝ (τ : ℕ) (vectorField (K := ℝ)) := by
89     apply contDiff_pi.2
90     intro i
91     fin_cases i <=> dsimp [vectorField, field] <=> fun_prop
92
93   /-- An end-to-end analytic counterexample in dimension eight. -/
94   theorem weak_markus_yamabe_false_in_eight :
95     ∃ F : (Fin 8 → ℝ) → (Fin 8 → ℝ),
96       ContDiff ℝ (τ : ℕ) F ∧ IsHurwitz F ∧ ¬ Function.Injective F :=
97     (vectorField, vectorField_smooth, vectorField_hurwitz, not_injective)
98
99   end
100   end ProofExplorer.Jacobian.MarkusEight

```

D CalculusLibrary

ProofExplorer/Jacobian/CalculusLibrary.lean

```

1 import ProofExplorer.Jacobian.Library
2
3 namespace ProofExplorer.Jacobian.CalculusLibrary
4 open MvPolynomial
5
6 set_option backward.isDefEq.respectTransparency false in
7 /-- Formal partial derivatives agree with analytic coordinate derivatives. -/
8 theorem eval_hasDerivAt {n : ℕ} (p : MvPolynomial (Fin n) ℝ)
9   (u : Fin n → ℝ) (j : Fin n) :
10   HasDerivAt (fun t => eval (Function.update u j t) p)
11     (eval u (pderiv j p)) (u j) := by
12   induction p using MvPolynomial.induction_on with
13   | C c => simp using hasDerivAt_const (u j) c
14   | add p q hp hq =>
15     convert! hp.add hq using 1 <=> simp [Pi.add_apply] <=> rfl
16   | mul_X p i hp =>
17     by_cases hij : j = i
18     · subst i
19       convert! hp.mul (hasDerivAt_id (u j)) using 1 <=>
20         simp [Derivation.leibniz, smul_eq_mul, Pi.mul_apply, mul_comm, add_comm] <=> rfl
21     · convert! hp.mul (hasDerivAt_const (u j) (u i)) using 1 <=>
22       simp [Derivation.leibniz, smul_eq_mul, hij, Function.update_of_ne (Ne.symm hij), Pi.mul_apply,
23         mul_comm] <=> rfl
24 end ProofExplorer.Jacobian.CalculusLibrary

```

E VectorLibrary

ProofExplorer/Jacobian/VectorLibrary.lean

```

1 import ProofExplorer.Jacobian.Library
2
3 namespace ProofExplorer.Jacobian.VectorLibrary
4
5 @[simp] theorem eight_0 {K : Type*} (a b c d e f g h : K) :
6   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 0 = a := rfl
7
8 @[simp] theorem eight_1 {K : Type*} (a b c d e f g h : K) :
9   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 1 = b := rfl
10
11 @[simp] theorem eight_2 {K : Type*} (a b c d e f g h : K) :
12   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 2 = c := rfl
13
14 @[simp] theorem eight_3 {K : Type*} (a b c d e f g h : K) :
15   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 3 = d := rfl
16
17 @[simp] theorem eight_4 {K : Type*} (a b c d e f g h : K) :
18   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 4 = e := rfl
19
20 @[simp] theorem eight_5 {K : Type*} (a b c d e f g h : K) :
21   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 5 = f := rfl
22
23 @[simp] theorem eight_6 {K : Type*} (a b c d e f g h : K) :
24   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 6 = g := rfl
25
26 @[simp] theorem eight_7 {K : Type*} (a b c d e f g h : K) :
27   (![a,b,c,d,e,f,g,h] : Fin 8 → K) 7 = h := rfl
28
29 end ProofExplorer.Jacobian.VectorLibrary

```

F Library

ProofExplorer/Jacobian/Library.lean

```

1 import Mathlib
2
3 namespace ProofExplorer.Jacobian.Library
4 open Polynomial
5 variable {R : Type*} [CommRing R]
6
7 theorem coeff_mul_0 (p q : R[X]) : (p*q).coeff 0 = p.coeff 0 * q.coeff 0 := by
8   simp [Polynomial.coeff_mul, Finset.Nat.sum_antidiagonal_succ]
9   all_goals ring
10
11 theorem coeff_mul_1 (p q : R[X]) : (p*q).coeff 1 = p.coeff 0 * q.coeff 1 + p.coeff 1 * q.coeff 0 := by
12   simp [Polynomial.coeff_mul, Finset.Nat.sum_antidiagonal_succ]
13   all_goals ring
14
15 theorem coeff_mul_2 (p q : R[X]) : (p*q).coeff 2 = p.coeff 0 * q.coeff 2 + p.coeff 1 * q.coeff 1 + p.coeff 2
  * q.coeff 0 := by
16   simp [Polynomial.coeff_mul, Finset.Nat.sum_antidiagonal_succ]
17   all_goals ring
18
19 theorem coeff_mul_3 (p q : R[X]) : (p*q).coeff 3 = p.coeff 0 * q.coeff 3 + p.coeff 1 * q.coeff 2 + p.coeff 2
  * q.coeff 1 + p.coeff 3 * q.coeff 0 := by
20   simp [Polynomial.coeff_mul, Finset.Nat.sum_antidiagonal_succ]
21   all_goals ring
22
23 theorem coeff_pow_2_0 (p : R[X]) : (p^2).coeff 0 = 1*(p.coeff 0)^2 := by
24   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
25   norm_num [Polynomial.coeff_one]
26   all_goals ring
27
28 theorem coeff_pow_2_1 (p : R[X]) : (p^2).coeff 1 = 2*(p.coeff 0)^1*(p.coeff 1)^1 := by
29   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
30   norm_num [Polynomial.coeff_one]
31   all_goals ring
32
33 theorem coeff_pow_2_2 (p : R[X]) : (p^2).coeff 2 = 1*(p.coeff 1)^2 + 2*(p.coeff 0)^1*(p.coeff 2)^1 := by
34   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
35   norm_num [Polynomial.coeff_one]
36   all_goals ring
37
38 theorem coeff_pow_2_3 (p : R[X]) : (p^2).coeff 3 = 2*(p.coeff 1)^1*(p.coeff 2)^1 + 2*(p.coeff 0)^1*(p.coeff
  3)^1 := by
39   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
40   norm_num [Polynomial.coeff_one]
41   all_goals ring
42
43 theorem coeff_pow_3_0 (p : R[X]) : (p^3).coeff 0 = 1*(p.coeff 0)^3 := by
44   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
45   norm_num [Polynomial.coeff_one]
46   all_goals ring
47
48 theorem coeff_pow_3_1 (p : R[X]) : (p^3).coeff 1 = 3*(p.coeff 0)^2*(p.coeff 1)^1 := by
49   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
50   norm_num [Polynomial.coeff_one]
51   all_goals ring
52
53 theorem coeff_pow_3_2 (p : R[X]) : (p^3).coeff 2 = 3*(p.coeff 0)^1*(p.coeff 1)^2 + 3*(p.coeff 0)^2*(p.coeff
  2)^1 := by
54   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
55   norm_num [Polynomial.coeff_one]
56   all_goals ring

```

```

57
58 theorem coeff_pow_3_3 (p : R[X]) : (p^3).coeff 3 = 1*(p.coeff 1)^3 + 6*(p.coeff 0)^1*(p.coeff 1)^1*(p.coeff
59 2)^1 + 3*(p.coeff 0)^2*(p.coeff 3)^1 := by
60   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
61   norm_num [Polynomial.coeff_one]
62   all_goals ring
63
64 theorem coeff_pow_4_0 (p : R[X]) : (p^4).coeff 0 = 1*(p.coeff 0)^4 := by
65   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
66   norm_num [Polynomial.coeff_one]
67   all_goals ring
68
69 theorem coeff_pow_4_1 (p : R[X]) : (p^4).coeff 1 = 4*(p.coeff 0)^3*(p.coeff 1)^1 := by
70   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
71   norm_num [Polynomial.coeff_one]
72   all_goals ring
73
74 theorem coeff_pow_4_2 (p : R[X]) : (p^4).coeff 2 = 6*(p.coeff 0)^2*(p.coeff 1)^2 + 4*(p.coeff 0)^3*(p.coeff
75 2)^1 := by
76   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
77   norm_num [Polynomial.coeff_one]
78   all_goals ring
79
80 theorem coeff_pow_4_3 (p : R[X]) : (p^4).coeff 3 = 4*(p.coeff 0)^1*(p.coeff 1)^3 + 12*(p.coeff 0)^2*(p.coeff
81 1)^1*(p.coeff 2)^1 + 4*(p.coeff 0)^3*(p.coeff 3)^1 := by
82   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
83   norm_num [Polynomial.coeff_one]
84   all_goals ring
85
86 theorem coeff_pow_5_0 (p : R[X]) : (p^5).coeff 0 = 1*(p.coeff 0)^5 := by
87   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
88   norm_num [Polynomial.coeff_one]
89   all_goals ring
90
91 theorem coeff_pow_5_1 (p : R[X]) : (p^5).coeff 1 = 5*(p.coeff 0)^4*(p.coeff 1)^1 := by
92   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
93   norm_num [Polynomial.coeff_one]
94   all_goals ring
95
96 theorem coeff_pow_5_2 (p : R[X]) : (p^5).coeff 2 = 10*(p.coeff 0)^3*(p.coeff 1)^2 + 5*(p.coeff 0)^4*(p.coeff
97 2)^1 := by
98   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
99   norm_num [Polynomial.coeff_one]
100   all_goals ring
101
102 theorem coeff_pow_5_3 (p : R[X]) : (p^5).coeff 3 = 10*(p.coeff 0)^2*(p.coeff 1)^3 + 20*(p.coeff
103 0)^3*(p.coeff 1)^1*(p.coeff 2)^1 + 5*(p.coeff 0)^4*(p.coeff 3)^1 := by
104   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
105   norm_num [Polynomial.coeff_one]
106   all_goals ring
107
108 theorem coeff_pow_6_0 (p : R[X]) : (p^6).coeff 0 = 1*(p.coeff 0)^6 := by
109   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
110   norm_num [Polynomial.coeff_one]
111   all_goals ring
112
113 theorem coeff_pow_6_1 (p : R[X]) : (p^6).coeff 1 = 6*(p.coeff 0)^5*(p.coeff 1)^1 := by
114   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
115   norm_num [Polynomial.coeff_one]

```

```

116   all_goals ring
117
118 theorem coeff_pow_6_3 (p : R[X]) : (p^6).coeff 3 = 20*(p.coeff 0)^3*(p.coeff 1)^3 + 30*(p.coeff
119   0)^4*(p.coeff 1)^1*(p.coeff 2)^1 + 6*(p.coeff 0)^5*(p.coeff 3)^1 := by
120   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
121   norm_num [Polynomial.coeff_one]
122   all_goals ring
123
124 theorem coeff_pow_7_0 (p : R[X]) : (p^7).coeff 0 = 1*(p.coeff 0)^7 := by
125   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
126   norm_num [Polynomial.coeff_one]
127   all_goals ring
128
129 theorem coeff_pow_7_1 (p : R[X]) : (p^7).coeff 1 = 7*(p.coeff 0)^6*(p.coeff 1)^1 := by
130   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
131   norm_num [Polynomial.coeff_one]
132   all_goals ring
133
134 theorem coeff_pow_7_2 (p : R[X]) : (p^7).coeff 2 = 21*(p.coeff 0)^5*(p.coeff 1)^2 + 7*(p.coeff 0)^6*(p.coeff
135   2)^1 := by
136   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
137   norm_num [Polynomial.coeff_one]
138   all_goals ring
139
140 theorem coeff_pow_7_3 (p : R[X]) : (p^7).coeff 3 = 35*(p.coeff 0)^4*(p.coeff 1)^3 + 42*(p.coeff
141   0)^5*(p.coeff 1)^1*(p.coeff 2)^1 + 7*(p.coeff 0)^6*(p.coeff 3)^1 := by
142   simp only [pow_succ, pow_zero, coeff_mul_0, coeff_mul_1, coeff_mul_2, coeff_mul_3]
143   norm_num [Polynomial.coeff_one]
144   all_goals ring
145
146 /-- Numeral rule for partial derivatives, in any finite number of variables. -/
147 theorem mv_pderiv_numeral {K : Type*} [CommRing K] {n : ℕ}
148   (i : Fin n) (m : ℕ) [m.AtLeastTwo] :
149   MvPolynomial.pderiv i (no_index (OfNat.ofNat m)) : MvPolynomial (Fin n) K = 0 := by
150   exact (MvPolynomial.pderiv i).map_natCast m
151
152 /-- Coefficient extraction after multiplication by a variable. -/
153 theorem mv_coeff_X_mul {K : Type*} [CommSemiring K] {n : ℕ}
154   (m : Fin n → ℕ) (i : Fin n) (p : MvPolynomial (Fin n) K) :
155   MvPolynomial.coeff m (MvPolynomial.X i * p) =
156   if i ∈ m.support then MvPolynomial.coeff (m-Finsupp.single i 1) p else 0 :=
157   MvPolynomial.coeff_X_mul' m i p
158
159 end ProofExplorer.Jacobian.Library

```

G Independent exact arithmetic check

This Python program requires SymPy. It is supplementary evidence; the Lean proof does not depend on its output.

```

1  """Exact symbolic certificate and reproducible Lean certificate generation."""
2  import sympy as s
3  from pathlib import Path
4  import json
5  zv=s.symbols('a b c d e x y z')
6  a,b,c,d,e,x,y,z=zv
7  t=s.Symbol('t')
8  F=s.Matrix([-a-2*x**3*y**2+x**3*z/2,-b+d+15*x**2*y**3-6*x**2*y*z,-c+e+9*x**2*y**4-3*x**2*y**2*z,-d+12*x**3*
y**4-3*x**3*y**2*z,-e+4*x**3*y**5-x**3*y**3*z,a+s.Rational(3,2)*x**2*y-x,b-3*x*z-y,c+5*x*y**3-3*x*y*z-z-s.R
ational(1,4)])
9  J=F.jacobian(zv)
10 points=[[0,0,0,0,0,0,0,-s.Rational(1,4)],[s.Rational(13,4),45,-s.Rational(297,8),-s.Rational(351,8),s.Ratio
nal(351,16),1,-s.Rational(3,2),s.Rational(31,2)],[-s.Rational(13,4),-45,-s.Rational(297,8),s.Rational(351,8)
],s.Rational(351,16),-1,s.Rational(3,2),s.Rational(31,2)]]
11 assert (J+s.eye(8)).charpoly(t).as_expr()==t**8
12 assert all(F.subs(dict(zip(zv,p))))==s.zeros(8,1) for p in points)
13 Path(__file__).with_suffix('.json').write_text(json.dumps({'order':list(map(str,zv)), 'field':list(map(str,F
)), 'shifted_charpoly':str(t**8), 'zeros':[[str(v) for v in p] for p in
points]}),indent=2)+'\n')
14 print('Verified the characteristic polynomial and three rational zeros.')
```

Recorded output

Verified the characteristic polynomial and three rational zeros.