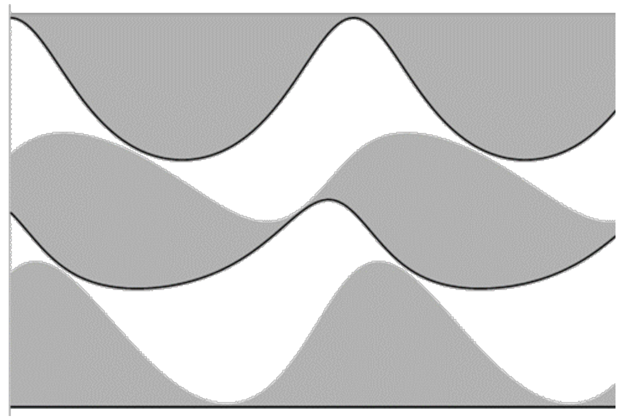
Properties of Polygonal Orbits in Elliptic Billiards

Dan S. Reznik
Ronaldo Garcia
Jair Koiller

Rio de Janeiro, Junho, 2019

To Do

- N=3,4,5: show other vertices' speed / speed of A.
- "Flat" view. Pin one side horizontally, as opposed to one vertex to (0,1)
- What do notable loci look like in "Vert" and "Flat" views?
- ETC: 101-1000?

Utils

In[1]:=
```
SetDirectory["C:\\Users\\drezn\\Dropbox\\Mathematica"];
```

In[2]:=
```
toDeg[r_] := 180. * r / π;
toRad[d_] := π * d / 180.;
negl = Compile[{{v, _Real}}, Abs[v] < 10^-18];
safeDiv = Compile[{{num, _Real}, {denom, _Real}}, If[negl[denom], 0, num / denom]];
magn2 = Compile[{{v, _Real, 1}}, v.v];
magn = Compile[{{v, _Real, 1}}, Sqrt[magn2[v]]];
flipY[{x_, y_}] := {x, -y};
flipX[{x_, y_}] := {-x, y};
perp[{x_, y_}] := {-y, x};
perpNeg[{x_, y_}] := {y, -x};
refl = Compile[{{v, _Real, 1}, {n, _Real, 1}}, 2 (v.n) n / magn2[n] - v];
norm = Compile[{{v, _Real, 1}}, v / magn[v]];
clamp[v_, max_: 100] := If[v > max, max, If[v < -max, -max, v]];
(*ray[p0_,phat_,d_]:=p0+phat*d;*)
ray = Compile[{{p0, _Real, 1}, {phat, _Real, 1}, {d, _Real}}, p0 + phat * d];
Clear@rot;
rot[p_, st_, ct_] := Module[{m},
   m = {{ct, -st}, {st, ct}};
   m.p];
rot[p_, t_] := rot[p, Sin@t, Cos@t];
getEquilateral[th_] := Module[{s = Sin[2 π / 3.], c = Cos[2 π / 3.]},
   NestList[rot[#, s, c] &, rot[{1., 0.}, Sin@th, Cos@th], 2]];
(*x0+nx*t=0= >t=-x0/nx;*)
interY[p0_, phat_] := Module[{t},
   t = -p0[[1]] / phat[[1]];
   ray[p0, phat, t]];
(*y0+ny*t=0= >t=-y0/ny;*)
interX[p0_, phat_] := Module[{t},
   t = -p0[[2]] / phat[[2]];
   ray[p0, phat, t]];
```

$-s\,(\mathrm{dl.dl}) + (p - \mathrm{l1})\,\mathrm{dl} == 0 \Rightarrow \boldsymbol{s = (p - \mathrm{l1}).\mathrm{dl} / \mathrm{dl}^\wedge 2}$

In[22]:=
```
closestPerp[p_, l1_, l2_] := Module[{dl = l2 - l1, s},
   s = ((p - l1).dl) / (dl.dl);
   ray[l1, dl, s]];
```

In[23]:=
```
closestDist[p_, l1_, l2_] := magn[p - closestPerp[p, l1, l2]]
```

In[24]:=
```
Clear@quadRoots;
quadRoots[a_, b_, c_] := Module[{det = b^2 - 4 a c, sqrtDet},
   If[det < 0, Print["quadRoots fail: {a,b,c}=" <> ToString@{a, b, c}];
    False,
    sqrtDet = Sqrt[det]; {-b - sqrtDet, -b + sqrtDet} / (2 a)]];
Clear@interRays;
interRays[p1_, n1_, p2_, n2_] := Module[{m, b, sols},
   m = Transpose[{n1, n2}]; (*{{nx1,-nx2},{ny1,-ny2}};*)
   If[negl[Det[m]], p1,
    b = p2 - p1; (*{x2-x1,y2-y1}];*)
    sols = LinearSolve[m, b];
    ray[p1, n1, sols[[1]]]]
  ];
```

In[28]:=
```
Second[l_] := l[[2]];
Third[l_] := l[[3]];
Fourth[l_] := l[[4]];
Fifth[l_] := l[[5]];
Sixth[l_] := l[[6]];
Seventh[l_] := l[[7]];
```

In[34]:=
```
Clear[getStats];
getStats[vals_] := Module[{mean, sd, z},
   mean = Mean@vals;
   sd = StandardDeviation@vals;
   z = safeDiv[sd, mean];
   {mean, sd, z, Min@vals, Max@vals, Median@vals, Length@vals}];
```

In[691]:=
```
Clear@getStatsLabeled;
getStatsLabeled[vals_] :=
 Thread[{"mean", "sd", "zscore", "min", "max", "median", "N"} -> getStats[vals]]
```

Triangles

In[36]:=
```
cosHalfAngle[cosAngle_] := Sqrt[(1. + cosAngle) / 2];(* careful: plus or minus *)
sinHalfAngle[cosAngle_] := Sqrt[(1. - cosAngle) / 2];(* careful: plus or minus *)
cosDoubleAngle[cosAngle_] := 2 * cosAngle * cosAngle - 1.; (* cA^2-sA^2 = 2 cA^2-1 *)
sinDoubleAngle[sinAngle_] := 2 * sinAngle * Sqrt[1. - sinAngle^2];(* 2 sA cA *)
(* gives cos(A) *)
lawOfCosines[a_, b_, c_] := (b^2 + c^2 - a^2) / (2. b c);
getSin[theCos_] := Sqrt[1. - theCos^2];
sinCosTripleAngle[s_, c_, s2_, c2_] := Module[{s3, c3},
   c3 = c2 c - s2 s;
   s3 = s2 c + s c2;
   {s3, c3}];
getSinCosApB[sa_, sb_, ca_, cb_] := {sa cb + sb ca , ca cb - sa sb};
getSinCosAmB[sa_, sb_, ca_, cb_] := {sa cb - sb ca, ca cb + sa sb};
getSinApmB[sa_, sb_, ca_, cb_] := {sa cb + sb ca, sa cb - sb ca};
```

In[46]:=
```
Clear@getTriBisectors;
getTriBisectors[p1_, p2_, p3_] := Module[{u12, u23, u31},
   u12 = norm[p2 - p1];
   u23 = norm[p3 - p2];
   u31 = norm[p1 - p3];
   {
    norm[u12 - u31],
    norm[u23 - u12],
    norm[u31 - u23]
   }
  ];
```

In[48]:=
```
getBisector[u_, v_] := norm[norm@u + norm@v];
getEllPs[a_, ts_] := {a Cos@#, Sin@#} & /@ ts;
reflAboutLine[p_, l1_, l2_] := refl[p - l1, l2 - l1] + l1;
```

Ellipse

In[51]:=
```
ellEqn[a_, x_, y_] := (x / a) ^2 + y^2 - 1;
ellEqnb[a_, b_, x_, y_] := (x / a) ^2 + (y / b) ^2 - 1;
ellGrad = Compile[{{a, _Real}, {x, _Real}, {y, _Real}}, -{x, y a^2}];
(* (x/a)^2+y^2==1 *)
ellGradb = Compile[{{a, _Real}, {b, _Real}, {x, _Real}, {y, _Real}}, -{x b^2, y a^2}];
(* (x/a)^2+(y/b)^2==1 *)
ellY = Compile[{{a, _Real}, {x, _Real}}, Sqrt[1 - (x / a) ^2]];
ellP = Compile[{{a, _Real}, {x, _Real}}, {x, Sqrt[1 - (x / a) ^2]}];
ellYb = Compile[{{a, _Real}, {b, _Real}, {x, _Real}}, b * ellY[a / b, x / b]];
ellError[a_, b_, {x_, y_}] := (x / a) ^2 + (y / b) ^2 - 1;
Clear@ellPb; ellPb[a_, b_, t_] := {a Cos@t, b Sin@t};
Clear@eccentricity; eccentricity[a_, b_] := Sqrt[1 - (b / a) ^2];
focalDistance = Compile[{{a, _Real}, {b, _Real}}, Sqrt[a^2 - b^2]];
getFoci = Compile[{{a, _Real}}, Module[{c}, If[a < 1, c = Sqrt[1 - a^2];
      {{0, c}, {0, -c}}, c = Sqrt[a^2 - 1];
      {{-c, 0}, {c, 0}}]]]];
```

In[63]:=
```
ellRayCoeffs = Module[{r, eqn, sols},
  r = ray[{x, y}, {nx, ny}, s];
  eqn = FullSimplify[ellEqn[a, r[[1]], r[[2]]],
    Assumptions → a > 0 && Abs[x] <= a && (nx^2 + ny^2 == 1)];
  FullSimplify[a^2 CoefficientList[eqn, s], a > 0]]
```

CompiledFunction::cfta : Argument {x, y} at position 1 should be a rank 1 tensor of machine-size real numbers. >>

Out[63]= $\{x^2 + a^2 (-1 + y^2),\ 2 (nx\,x + a^2\,ny\,y),\ nx^2 + a^2\,ny^2\}$

In[64]:=
```
Clear@ellInterRay;
ellInterRay[a_, {x_, y_}, {nx_, ny_}] := Module[{c2, c1, c0, ss},
    c2 = nx^2 + a^2 ny^2;
    c1 = 2 (nx x + a^2 ny y);
    c0 = x^2 + a^2 (-1 + y^2);
    ss = quadRoots[c2, c1, c0];
    If[ListQ[ss],
     ray[{x, y}, {nx, ny}, #] & /@ ss,
     {{x, y}, {x, y}}]
   ];
```

Used for Proofs

```
In[66]:= quadRootsUnprot[a_, b_, c_] := Module[{det = b^2 - 4 a c, sqrtDet},
           sqrtDet = Sqrt[det]; {-b - sqrtDet, -b + sqrtDet} / (2 a)];
         (*Clear@ellInterRayUnprot;
         ellInterRayUnprot[a_,{x_,y_},{nx_,ny_}]:=Module[{c2,c1,c0,ss},
           c2=nx^2+a^2 ny^2;
           c1=2 (nx x+a^2 ny y);
           c0=x^2+a^2 (-1+y^2);
           ss=quadRootsUnprot[c2,c1,c0];
           ray[{x,y},{nx,ny},#]&/@ss
          ];*)

In[67]:= quadRootsUnprotC =
           Compile[{{a, _Real}, {b, _Real}, {c, _Real}}, Module[{det = b^2 - 4 a c, sqrtDet},
             sqrtDet = Sqrt[det];
             {-b - sqrtDet, -b + sqrtDet} / (2 a)]];

In[68]:= Clear@ellInterRayUnprot;
         ellInterRayUnprot = Compile[{{a, _Real}, {p, _Real, 1}, {n, _Real, 1}},
            Module[{x, y, nx, ny, a2, c2, c1, c0, ss},
             {x, y} = p;
             {nx, ny} = n;
             a2 = a * a;
             c2 = nx^2 + a2 ny^2;
             c1 = 2 (nx x + a2 ny y);
             c0 = x^2 + a2 (-1 + y^2);
             ss = quadRootsUnprotC[c2, c1, c0];
             ray[p, n, #] & /@ ss
            ]];

In[70]:= (*Module[{a=1.5,alpha=.1,p,tc,t0,n=10000},
          p={1.5 Cos[alpha],Sin[alpha]};
          t0=Timing[Table[ellInterRayUnprot[a,p,ellGrad[a,Sequence@@p]],{i,n}]][[1]];
          tc=Timing[Table[ellInterRayUnprotC[a,p,ellGrad[a,Sequence@@p]],{i,n}]][[1]];
          {t0,tc}]*)

In[1215]:= Clear@getInterReflNonComp;
         getInterReflNonComp[a_, pfrom_, pto_] := Module[{norm, theRefl, pnext},
           norm = ellGrad[a, pto[[1]], pto[[2]]];
           theRefl = refl[pfrom - pto, norm];
           pnext = ellInterRayUnprot[a, pto, theRefl][[2]];
           pnext];
```

```
In[71]:= Clear@getInterRefl;
getInterRefl = Compile[
   {{a, _Real}, {pfrom, _Real, 1}, {pto, _Real, 1}}, Module[{norm, theRefl, pnext},
    norm = ellGrad[a, pto[[1]], pto[[2]]];
    theRefl = refl[pfrom - pto, norm];
    pnext = ellInterRayUnprot[a, pto, theRefl][[2]];
    pnext]];

In[73]:= Clear@bounceRay;
bounceRay[a_, p1_, p2_, bounces_] := Module[{p1v, p2v, p3v},
  p1v = p1; p2v = p2;
  {p1, p2, Sequence @@ Table[
     p3v = getInterRefl[a, p1v, p2v];
     p1v = p2v;
     p2v = p3v;
     p3v, {i, bounces}]}]

In[75]:= Clear@triSides;
triSides[vs_] := MapThread[(#1 - #2) &, {vs, RotateLeft@vs}];
Clear@triLengths; triLengths[vs_] := magn /@ triSides[vs];
triPer[tri_] := Total[triLengths@tri];
(* a,b,c are side lengths *)
Clear@triAreaHeron; triAreaHeron[{a_, b_, c_}] := Module[{s},
  s = (a + b + c) / 2; (* semi perimeter *)
  Sqrt[s * (s - a) * (s - b) * (s - c)]];
(* order by oposition to vertices *)
getMedians[t1_, t2_, t3_] := {t2 + t3, t1 + t3, t1 + t2} / 2;
getMediansV[vs_] := .5 (vs + RotateLeft@vs);
```

Centroids

```
In[81]:= PerimeterCentroid[vtx_] := Module[{sides, meds, per, perCentroid},
   sides = triLengths@vtx;
   meds = getMediansV @@ vtx;
   per = Total@sides;
   perCentroid = Sum[meds[[i]] * sides[[i]], {i, Length@vtx}] / per;
   perCentroid];

In[82]:= (* vtx mean, perimeter, area *)
getCentroids[vtx_] :=
  {Mean@vtx, PerimeterCentroid@vtx, RegionCentroid[Polygon@vtx]};
```

In[83]:=
```
Clear@getPolyCosines;
getPolyCosines[poly_] := Module[{us},
   us = MapThread[norm[#1 - #2] &, {poly, RotateLeft@poly}];
   MapThread[(-#1.#2) &, {us, RotateRight@us}]];
```

In[85]:=
```
Clear@polySides;
polySides[poly_] := MapThread[magn[#1 - #2] &, {poly, RotateLeft@poly}];
```

In[87]:=
```
getCircularity[locus_] := Module[{rads, mean, sd},
   rads = magn /@ locus;
   mean = Mean@rads;
   sd = StandardDeviation@rads;
   {
    "min" -> Min@rads,
    "max" -> Max@rads,
    "mean" → mean,
    "sd" → sd,
    "zscore" → sd / mean
   }];
```

In[88]:=
```
(* by zscore *)
getMinCircularity[loci_] := Module[{l, imin},
  l = Select["zscore" /. Quiet@(getCircularity /@ loci), NumberQ];
  imin = First@First@Position[l, Min[l]];
  {l[[imin]], centerNames[[imin]]}]
```

Formatting

In[89]:=
```
nfn[v_, n_] := ToString@NumberForm[v, {n + 2, n}];
nfne[v_, n_] := ToString@NumberForm[v, {n + 2, n},
    NumberFormat → (Row[{#1, If[#3 ≠ "", "*10^", ""], #3}] &)];
nf[v_] := ToString@NumberForm[v, {3, 2}];
nf1[v_] := ToString@NumberForm[v, {3, 1}];
nf0[v_] := ToString@IntegerPart[v];
Clear@padn;
padn[n_, pads_: 3] :=
  ToString@NumberForm[n, {pads, 0}, NumberPadding → {"0", ""}, NumberPoint → ""]
```

Ellipse Invariants

Constant Perimeter (Darboux, 1917)

$$P = \frac{2\sqrt{2\,\delta - a^2 - b^2}\,\left(a^2 + b^2 + \delta\right)}{a^2 - b^2}, \quad \delta = \sqrt{a^4 - a^2 b^2 + b^4}.$$

Jean - Gaston Darboux (1842 - 1917)

In[95]:=
```
deltaA[a_] := Sqrt[a^4 - a^2 + 1];
```

In[96]:=
```
Clear[darbouxP];
darbouxP[a_, b_ : 1] := Module[{a2, b2, delta, per},
   a2 = a^2;
   b2 = b^2;
   delta = Sqrt[a2^2 - a2 b2 + b2^2];
   per = 2 Sqrt[2 delta - a2 - b2] (a2 + b2 + delta) / (a2 - b2);
   per
  ];
```

Exit angle at x_1 which creates triangular orbit (Garcia, 2019)

$$\cos\alpha = \frac{a^2 b\sqrt{-a^2 - b^2 + 2\sqrt{a^4 - b^2 c^2}}}{c^2\sqrt{a^4 - c^2 x_1^2}}.$$

```
In[98]:= Clear@cosAlpha; cosAlpha[a_, x_] := Module[{x2, a2, a4, c2, denom},
           x2 = x^2;
           a2 = a^2;
           a4 = a2^2;
           c2 = a2 - 1;
           denom = c2 Sqrt[a4 - c2 x2];
           a2 Sqrt[-a2 - 1 + 2 Sqrt[a4 - c2]] / denom];

In[99]:= cosAlphaQuad[a_, x1_] := a^2 / Sqrt[a^6 + a^4 + x1^2 (1 - a^4)];
         cosAlphaQuadSelfInter[a_, x1_] := a^2 / Sqrt[(a^2 - 1) (a^4 + x1^2 (1 - a^2))];
```

Can I find an expression for the next orbit point P_2 in terms of “a” and x_1

```
In[101]:= Clear@getP2;
          getP2[a_, x1_] := Module[{y, p1, norm, ca, sa, p2, normRot, normRotNeg, p2Neg},
            y = -ellY[a, x1];
            p1 = {x1, y};
            ca = cosAlpha[a, x1] /. {Sqrt[1 - a^2 + a^4] → d} /. {-1 - a^2 + 2 d → d2};
            (* to simplify *)
            sa = Sqrt[1 - ca^2];
            norm = ellGrad[a, x1, y];
            normRot = rot[norm, sa, ca];
            normRotNeg = rot[norm, -sa, ca];
            p2 = ellInterRayUnprot[a, p1, normRot][[2]]; (* get 2nd solution *)
            p2Neg = ellInterRayUnprot[a, p1, normRotNeg][[2]];
            {p2, p2Neg}];
```

Triangular Orbits

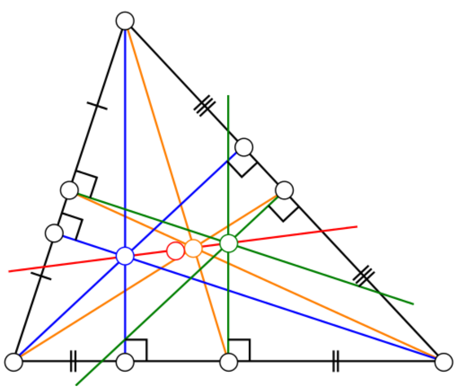
In[102]:=
```
Clear@getReflData; getReflData [a_, p0_, normal_, sinAlpha_, cosAlpha_] :=
 Module[{nRot, inter, interNorm, interRefl, nextInter, nextInterNorm},
  nRot = rot[normal, sinAlpha, cosAlpha];
  inter = ellInterRay[a, p0, nRot][[2]];
  interNorm = norm[ellGrad[a, inter[[1]], inter[[2]]]];
  interRefl = refl[norm[p0 - inter], interNorm];
  nextInter = ellInterRay[a, inter, interRefl][[2]];
  nextInterNorm = norm[ellGrad[a, nextInter[[1]], nextInter[[2]]]];
  { (* returns an "object" *)
   "nRot" → nRot,
   "inter" → inter,
   "interNorm" → interNorm,
   "interRefl" → interRefl,
   "nextInter" → nextInter,
   "nextInterNorm" → nextInterNorm
  }];
```

```
In[103]:= Options[getOrbitData] = {posY → False}; Clear@getOrbitData;
getOrbitData[a_, x_, OptionsPattern[]] :=
 Module[{y, o1, n1, ca, sa, sa2, ca2, halfAlpha, pos, neg},
  y = -ellY[a, x];
  If[OptionValue@posY, y = -y];
  o1 = {x, y};
  n1 = norm[ellGrad[a, x, y]];
  ca = cosAlpha[a, x];
  sa = Sqrt[1 - ca^2];
  pos = getReflData[a, o1, n1, sa, ca];
  neg = getReflData[a, o1, n1, -sa, ca]; (* orbit at -α *)
  (* half-angle to display α at bisector *)
  sa2 = Sqrt[(1 - ca) / 2];
  ca2 = Sqrt[(1 + ca) / 2];
  halfAlpha = rot[n1, sa2, ca2];
  {
   "ca" → ca,
   "sa" → sa,
   "halfAlpha" → halfAlpha,
   "nRot" → ("nRot" /. pos),
   "interRefl" → ("interRefl" /. pos),
   "orbit" → {o1, "inter" /. pos, "nextInter" /. pos},
   "normals" → {n1, "interNorm" /. pos, "nextInterNorm" /. pos},
   "nRotNeg" → ("nRot" /. neg),
   "interReflNeg" → ("interRefl" /. neg),
   "orbitNeg" → {o1, "inter" /. neg, "nextInter" /. neg},
   "normalsNeg" → {n1, "interNorm" /. neg, "nextInterNorm" /. neg}
  }
 ];

In[104]:= getOrbitData[2., 1., posY → False] // ColumnForm

Out[104]= ca → 0.549885
sa → 0.835241
halfAlpha → {-0.699945, 0.714197}
nRot → {-0.954984, 0.296658}
interRefl → {0.999046, 0.0436602}
orbit → {{1., -0.866025}, {-1.99582, 0.0646022}, {1.94314, 0.236743}}
normals → {{-0.27735, 0.960769}, {0.991722, -0.128403}, {-0.898933, -0.438085}}
nRotNeg → {0.649963, 0.759966}
interReflNeg → {-0.999046, -0.0436602}
orbitNeg → {{1., -0.866025}, {1.94314, 0.236743}, {-1.99582, 0.0646022}}
normalsNeg → {{-0.27735, 0.960769}, {-0.898933, -0.438085}, {0.991722, -0.128403}}
```

```
In[105]:= Clear@orbitNormals; orbitNormals[a_, t_] := Module[{ellP, orbitData},
    ellP = {N@a * Cos[N@t], Sin[N@t]};
    orbitData = getOrbitData[N[a], ellP[[1]], posY → (ellP[[2]] > 0)];
    {"orbit" /. orbitData, "normals" /. orbitData}
   ];

In[106]:= getAlpha[a_, t_] := Module[{x1, ca},
     x1 = a * Cos[t];
     ca = cosAlpha[a, a Cos[t]];
     ArcCos[ca]];

In[1903]:= Clear@getOrbitCosines;
getOrbitCosines[a_, t_] := Module[{orbit, normals},
     {orbit, normals} = orbitNormals[a, t];
     getPolyCosines@orbit];
```

Major Triangular Centers (Notables)

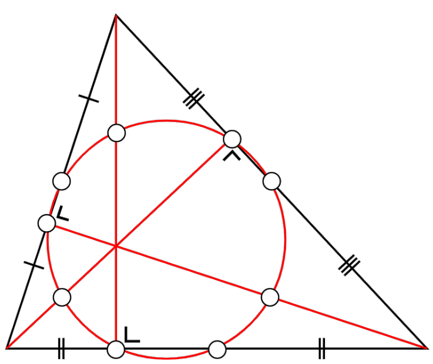
Linha de Euler: bari, ortho, circum, 9-pc all lie on a line (but not incenter)

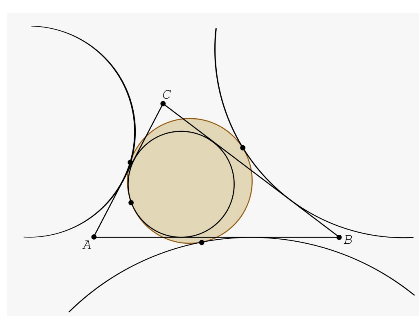


```
In[108]:= getAltBases[p1_, p2_, p3_] := Module[{sides},
     sides = {{p2, p3}, {p3, p1}, {p1, p2}};
     MapThread[closestPerp[#1, #2[[1]], #2[[2]]] &, {{p1, p2, p3}, sides}]];
```

In[109]:=
```
getIncenter[p1_, n1_, p2_, n2_] := interRays[p1, n1, p2, n2];
Clear@getBaricenter;
getBaricenter[p1_, p2_, p3_] := Module[{medians}, (*(p1+p2+p3)/3;*)
  medians = getMedians[p1, p2, p3];
  interRays[p1, medians[[1]](*23*) - p1, p2, medians[[2]](*31*) - p2]
 ];
getOrthocenter[p1_, p2_, p3_] := Module[{altBases},
   altBases = getAltBases[p1, p2, p3];
   interRays[p1, altBases[[1]] - p1, p2, altBases[[2]] - p2]
  ];
getCircumcenter[p1_, p2_, p3_] := Module[{medians, medianPerps},
   medians = getMedians[p1, p2, p3];
   medianPerps = {
     medians[[1]] + norm[perp[p2 - p3]],
     medians[[2]] + norm[perp[p3 - p1]],
     medians[[3]] + norm[perp[p1 - p2]]
    };
   interRays[medians[[1]](*2+3*),
    perp[p2 - p3], medians[[2]](*3+1*), perp[p1 - p3]]
  ];
```

Nine Point Circle: its center is the circumcenter of the medians, and is tangent to incircle (at Feuerbach point) and all 3 excircles



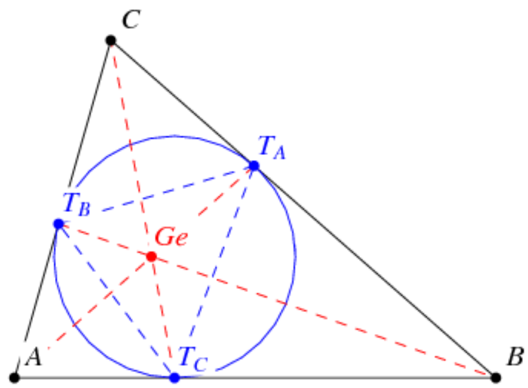


In[113]:=
```
getNinepointcenter[p1_, p2_, p3_] := Module[{medians},
   medians = getMedians[p1, p2, p3];
   getCircumcenter @@ medians
  ];
```

In[114]:=
```
(* shoot ray from incenter *)
getFeuerbachpoint[npc_, incenter_, inradius_] := Module[{inter},
   ray[incenter, norm[incenter - npc], inradius]
  ];
```

In[115]:=
```
getExcenters[orbit_, normals_] := Module[{perps, perpsNeg, exc},
   perps = perp /@ normals;
   perpsNeg = perpNeg /@ normals;
   exc = MapThread[interRays[#1, #2, #3, #4] &,
     {orbit, perps, RotateLeft@orbit, RotateLeft@perpsNeg}];
   exc];
```

Gergonne Point:
perspector of ABC and its contact triangle Ta, Tb, Tc



Trilinears

Trilinears x : y : z convert via triangle vertices A, B, C and sides a, b, c :

$$\underline{P} = \frac{ax}{ax+by+cz}\underline{A} + \frac{by}{ax+by+cz}\underline{B} + \frac{cz}{ax+by+cz}\underline{C}.$$

In[116]:=
```
Clear@trilinearToCartesian;
trilinearToCartesian[
   {A_, B_, C_}, (* vertices *)
   {a_, b_, c_}, (* side lengths *)
   {x_, y_, z_} (* trilinears *)
  ] := Module[{(*a,b,c,*)denom},
   (* may not need *)
   (*a=magn[C-B];b=magn[A-C];c=magn[B-A];*)
   denom = {a, b, c}.{x, y, z};
   (a x A + b y B + c z C) / denom];
```

In[118]:=
```
rotateSym[fabc_] := {fabc, fabc /. {a → b, b → c, c → a}, fabc /. {a → c, b → a, c → b}}
```

In[119]:=
```
(* trilinears: a:b:c *)
getIncenterTrilin[orbit_, sides_] := trilinearToCartesian[orbit, sides, {1, 1, 1}];
getCircumcenterTrilin[orbit_, {a_, b_, c_}] := Module[{a2 = a * a, b2 = b * b, c2 = c * c},
   trilinearToCartesian[orbit,
    {a, b, c}, {a (b2 + c2 - a2), b (c2 + a2 - b2), c (a2 + b2 - c2)}]];
getCircumcenterTrilin2[orbit_, {a_, b_, c_}] := Module[{cosA, cosB, cosC},
   cosA = lawOfCosines[a, b, c];
   cosB = lawOfCosines[b, a, c];
   cosC = lawOfCosines[c, a, b];
   trilinearToCartesian[orbit, {a, b, c}, {cosA, cosB, cosC}]];
getOrthocenterTrilin[orbit_, sides_] := Module[{cs},
   cs = getPolyCosines[orbit];
   trilinearToCartesian[orbit, sides, 1 / cs]
  ];
```

In[123]:=
```
getSymmedian[orbit_, sides_] := trilinearToCartesian[orbit, sides, sides];
getMitten[orbit_, {a_, b_, c_}] :=
  trilinearToCartesian[orbit, {a, b, c}, {b + c - a, c + a - b, a + b - c}];
getGergonne[orbit_, {a_, b_, c_}] := trilinearToCartesian[orbit,
   {a, b, c}, {b * c / (b + c - a), c * a / (c + a - b), a * b / (a + b - c)}];
getNagel[orbit_, {a_, b_, c_}] := trilinearToCartesian[orbit,
   {a, b, c}, {(b + c - a) / a, (c + a - b) / b, (a + b - c) / c}];
getSpieker[orbit_, {a_, b_, c_}] := trilinearToCartesian[
   orbit, {a, b, c}, {b c (b + c), c a (c + a), a b (a + b)}];
getCentroid[orbit_, {a_, b_, c_}] :=
  trilinearToCartesian[orbit, {a, b, c}, {b c, c a, a b}];
```

Generate Billiard as Locus

In[129]:=
```
getX100Trilin[orbit_, {a_, b_, c_}] :=
  trilinearToCartesian[orbit, {a, b, c}, 1 / {b - c, c - a, a - b}];
getX88Trilin[orbit_, {a_, b_, c_}] :=
  trilinearToCartesian[orbit, {a, b, c}, 1 / {b + c - 2 a, c + a - 2 b, a + b - 2 c}];
```

Get Triangular Center (Notable) Info

In[131]:=
```
Clear@getTouchPts; getTouchPts[orbit_, normals_] := Module[{inc, tps},
  inc = getIncenter[orbit[[1]], normals[[1]], orbit[[2]], normals[[2]]];
  tps = MapThread[closestPerp[inc, #1, #2] &, {orbit, RotateLeft@orbit}];
  tps];
```

In[132]:=
```
getAntiComplement[p_, bar_] := bar - 2 (p - bar);
```

```
In[133]:= Clear@getNotables;
getNotables[orbit_, normals_] :=
  Module[{inc, inradius, cir, circumradius, npc, exc, tps, ort,
    medians, npcradius, sides, bar, feu, antifeu, x88, perimeter},
   inc = getIncenter[orbit[[1]], normals[[1]], orbit[[2]], normals[[2]]];
   inradius = closestDist[inc, orbit[[1]], orbit[[2]]];
   tps = MapThread[closestPerp[inc, #1, #2] &, {orbit, RotateLeft@orbit}];
   medians = getMedians @@ orbit;
   cir = getCircumcenter @@ orbit;
   circumradius = magn[cir - orbit[[1]]];
   npc = getNinepointcenter @@ orbit;
   npcradius = magn[npc - medians[[1]]];
   exc = getExcenters[orbit, normals];
   sides = RotateLeft[triLengths@orbit];
   feu = getFeuerbachpoint[npc, inc, inradius];
   bar = getBaricenter @@ orbit;
   antifeu = getAntiComplement[feu, bar]; (* anticomplement *)
   x88 = getX88Trilin[orbit, sides];
   ort = getOrthocenter @@ orbit;
   perimeter = Total@sides;
   (* we think this is constant for excentral triangle *)
   {
    "inc" → inc,
    "bar" → bar, (* also: getCentroid[orbit,sides] *)
    "ort" → ort,
    "cir" → cir,
    "npc" → npc,
    "exc" → exc,
    "ex12" → exc[[1]],
    "ex23" → exc[[2]],
    "ex31" → exc[[3]],
    "feu" → feu,
    "antifeu" → antifeu,
    "x88" → x88,
    (* all thru trilinears, each computing sides unnecessarily *)
    "mit" → getMitten[orbit, sides],
    "sym" → getSymmedian[orbit, sides],
    "ger" → getGergonne[orbit, sides],
    "nag" → getNagel[orbit, sides],
    "spi" → getSpieker[orbit, sides],
    (* not really notables, aux info *)
    "incRadius" → inradius,
```

```
    "tps" → tps,
    "medians" → medians,
    "cirRadius" → circumradius,
    "npcRadius" → npcradius,
    "sides" → sides,
    "perimeter" → perimeter,
    "area" → triAreaHeron@sides
   }];
```

Excenters and Excircles

In[135]:=
```
Clear@showBisectors; showBisectors[vs_] := Module[{bs},
  bs = getTriBisectors @@ vs;
  Graphics[{FaceForm[None], EdgeForm[Black], Black, PointSize[Medium],
    Polygon[vs],
    Point[vs],
    MapThread[Text[#1, #2, {1.5, 0}] &, {{"P1", "P2", "P3"}, vs}],
    MapThread[Arrow[{#1, ray[#1, #2, .25]}] &, {vs, bs}]},
   ImageSize → Small]
 ];
```

In[136]:=
```
getExtouchPoints[orbit_, exc_] :=
  MapThread[closestPerp[#1, #2, #3] &, {exc, orbit, RotateLeft[orbit]}];
```

In[137]:=
```
Clear@getExcircleData;
getExcircleData[orbit_, exc_(*12,23,31*), npc_] := Module[
   {excFeet, altLengths, tps, radii, feus, nagel, medians, mitten, mittenPedals,
    mittenFeet, sides, cosABC, cosSum, cosProd, sideProd, altLengthsProd},
   (* side touch points to sides: 12,23,31 *)
   tps = getExtouchPoints[orbit, exc];
   excFeet = getAltBases @@ exc;
   altLengths = MapThread[magn[#1 - #2] &, {exc, excFeet}];
   radii = MapThread[magn[#1 - #2] &, {exc, tps}];
   feus = MapThread[ray[#1, norm[npc - #1], #2] &, {exc, radii}];
   (*nagel=
     interRays[orbit[[1]],tps[[2]]-orbit[[1]],orbit[[2]],tps[[3]]-orbit[[2]]];*)
   (* MapThread[Line[{#1,ray[#1,norm[#2-#1],10]}]&,{exc,RotateRight@medians}]}],
    {}],*)
   medians = getMedians @@ orbit;
   mitten =
    interRays[exc[[1]], medians[[3]] - exc[[1]], exc[[2]], medians[[1]] - exc[[2]]];
   mittenPedals = MapThread[closestPerp[mitten, #1, #2] &,
     {orbit, RotateLeft@orbit}];
   mittenFeet = MapThread[interRays[#1, #2 - #1, #3, #4 - #3] &,
```

```
      {exc, RotateRight@medians, RotateLeft@exc, RotateRight@exc}];
    sides = RotateLeft[triLengths@exc];
    cosABC = getPolyCosines[exc];
    cosSum = Total[cosABC];
    cosProd = cosABC[[1]] * cosABC[[2]] * cosABC[[3]];
    sideProd = sides[[1]] * sides[[2]] * sides[[3]];
    altLengthsProd = altLengths[[1]] * altLengths[[2]] * altLengths[[3]];
    {
     "sides" → sides,
     "perimeter" → Total[sides],
     "cosABC" → cosABC,
     "excFeet" → excFeet,
     "altLengths" → altLengths,
     "sumAltLengths" → Total@altLengths,
     "prodAltLengths" → altLengthsProd,
     "sumInvAltLengths" → Total[(1/#) & /@ altLengths],
     "cosSum" → cosSum,
     "cosProd" → cosProd,
     "sideProd" → sideProd,
     "tps" → tps, (* extouchpoints*)
     "radii" → radii,
     "sumRadiiSqr" → Total[#^2 & /@ radii],
     "sumInvRadiiSqr" → Total[(1/#^2) & /@ radii],
     "sumRadii" → Total@radii,
     "sumInvRadii" → Total[(1/#) & /@ radii],
     "feus" → feus,
     "mitten" → mitten,
     "mittenPedals" → mittenPedals,
     "mittenFeet" → mittenFeet (* BAD
       choice: intersecao do segm Vi-MITTEN com o lado oposto do extriangulo *)
    }
   ];
```

Trilinears: X(1)~X(100)

In[139]:=
```
Clear@getNewCenters;
getNewCenters[orbit_, {a_, b_, c_}, singles_: {}] := Module[{eqns,
    cosA, cosB, cosC, sinA, sinB, sinC,
    secA, secB, secC, cscA, cscB, cscC,
    tanA, tanB, tanC, cotA, cotB, cotC,
    cos2A, cos2B, cos2C, sin2A, sin2B, sin2C,
    sec2A, sec2B, sec2C, csc2A, csc2B, csc2C,
```

```
 cos3A, sin3A, cos3B, sin3B, cos3C, sin3C,
 sec3A, sec3B, sec3C, csc3A, csc3B, csc3C,
 cPi3 = Cos[π/3.], sPi3 = Sin[π/3.], cPi6 = Cos[π/6.], sPi6 = Sin[π/6.],
 sinApPi3, sinBpPi3, sinCpPi3, sinAmPi3, sinBmPi3, sinCmPi3,
 sinApPi6, sinBpPi6, sinCpPi6, sinAmPi6, sinBmPi6, sinCmPi6,
 cscApPi3, cscBpPi3, cscCpPi3, cscAmPi3, cscBmPi3, cscCmPi3,
 cscApPi6, cscBpPi6, cscCpPi6, cscAmPi6, cscBmPi6, cscCmPi6},
{cosA, cosB, cosC} =
 {lawOfCosines[a, b, c], lawOfCosines[b, a, c], lawOfCosines[c, a, b]};
{sinA, sinB, sinC} = getSin /@ {cosA, cosB, cosC};
{secA, secB, secC} = 1. / {cosA, cosB, cosC};
{cscA, cscB, cscC} = 1. / {sinA, sinB, sinC};
{tanA, tanB, tanC} = {sinA, sinB, sinC} / {cosA, cosB, cosC};
{cotA, cotB, cotC} = 1. / {tanA, tanB, tanC};
{cos2A, cos2B, cos2C} = cosDoubleAngle /@ {cosA, cosB, cosC};
{sec2A, sec2B, sec2C} = 1. / {cos2A, cos2B, cos2C};
{sin2A, sin2B, sin2C} = getSin /@ {cos2A, cos2B, cos2C};
{csc2A, csc2B, csc2C} = 1. / {sin2A, sin2B, sin2C};
{sin3A, cos3A} = sinCosTripleAngle[sinA, cosA, sin2A, cos2A];
{sin3B, cos3B} = sinCosTripleAngle[sinB, cosB, sin2B, cos2B];
{sin3C, cos3C} = sinCosTripleAngle[sinC, cosC, sin2C, cos2C];
{sec3A, sec3B, sec3C} = 1. / {cos3A, cos3B, cos3C};
{csc3A, csc3B, csc3C} = 1. / {sin3A, sin3B, sin3C};
{sinApPi3, sinAmPi3} = getSinApmB[sinA, sPi3, cosA, cPi3];
{sinBpPi3, sinBmPi3} = getSinApmB[sinB, sPi3, cosB, cPi3];
{sinCpPi3, sinCmPi3} = getSinApmB[sinC, sPi3, cosC, cPi3];
{sinApPi6, sinAmPi6} = getSinApmB[sinA, sPi6, cosA, cPi6];
{sinBpPi6, sinBmPi6} = getSinApmB[sinB, sPi6, cosB, cPi6];
{sinCpPi6, sinCmPi6} = getSinApmB[sinC, sPi6, cosC, cPi6];
{cscApPi3, cscBpPi3, cscCpPi3} = 1. / {sinApPi3, sinBpPi3, sinCpPi3};
{cscAmPi3, cscBmPi3, cscCmPi3} = 1. / {sinAmPi3, sinBmPi3, sinCmPi3};
{cscApPi6, cscBpPi6, cscCpPi6} = 1. / {sinApPi6, sinBpPi6, sinCpPi6};
{cscAmPi6, cscBmPi6, cscCmPi6} = 1. / {sinAmPi6, sinBmPi6, sinCmPi6};
eqns = {
  {"X(1)", Hold@{1, 1, 1}, "INCENTER"},
  {"X(2)", Hold@{b c, c a, a b}, "CENTROID"},
  {"X(3)", Hold@{cosA, cosB, cosC}, "CIRCUMCENTER"},
  {"X(4)", Hold@{secA, secB, secC}, "ORTHOCENTER"},
  {"X(5)", Hold@{cosB cosC + sinB sinC, cosC cosA + sinC sinA, cosA cosB + sinA sinB},
   "NINE-POINT CENTER"},
  {"X(6)", Hold@{a, b, c}, "SYMMEDIAN / LEMOINE / GREBE POINT"},
  {"X(7)", Hold@{b c / (b + c - a), c a / (c + a - b), a b / (a + b - c)},
   "GERGONNE POINT"},
```

```
{"X(8)", Hold@{(b+c-a) / a, (c+a-b) / b, (a+b-c) / c}, "NAGEL POINT"},
{"X(9)", Hold@{b+c-a, c+a-b, a+b-c}, "MITTENPUNKT"},
{"X(10)", Hold@{b c (b+c), c a (c+a), a b (a+b)}, "SPIEKER CENTER"},
{"X(11)", Hold@{1 - cosB cosC + sinB sinC, 1 - cosC cosA + sinC sinA,
   1 - cosA cosB + sinA sinB}, "FEUERBACH POINT"},
{"X(12)", Hold@{1 + cosB cosC + sinB sinC, 1 + cosC cosA + sinC sinA,
   1 + cosA cosB + sinA sinB}, "{X(1),X(5)}-HARMONIC CONJUGATE OF X(11)"},
{"X(13)", Hold@{cscApPi3, cscBpPi3, cscCpPi3},
 "1st ISOGONIC CENTER (FERMAT / TORRICELLI POINT)"},
{"X(14)", Hold@{cscAmPi3, cscBmPi3, cscCmPi3}, "2nd ISOGONIC CENTER"},
{"X(15)", Hold@{sinApPi3, sinBpPi3, sinCpPi3}, "1st ISODYNAMIC POINT"},
{"X(16)", Hold@{sinAmPi3, sinBmPi3, sinCmPi3}, "2nd ISODYNAMIC POINT"},
{"X(17)", Hold@{cscApPi6, cscBpPi6, cscCpPi6}, "1st NAPOLEON POINT"},
{"X(18)", Hold@{cscAmPi6, cscBmPi6, cscCmPi6}, "2nd NAPOLEON POINT"},
{"X(19)", Hold@{1 / (b^2 + c^2 - a^2), 1 / (a^2 + c^2 - b^2), 1 / (a^2 + b^2 - c^2)},
 "CLAWSON POINT"}, {"X(20)", Hold@
  {cosA - cosB cosC, cosB - cosC cosA, cosC - cosA cosB}, "DE LONGCHAMPS POINT"},
{"X(21)", Hold@{(b+c-a) / (b+c), (a+c-b) / (a+c), (a+b-c) / (a+b)},
 "SCHIFFLER POINT"},
{"X(22)", Hold@{a (b^4 + c^4 - a^4), b (c^4 + a^4 - b^4), c (a^4 + b^4 - c^4)},
 "EXETER POINT"}, {"X(23)", Hold@{a (b^4 + c^4 - a^4 - b^2 c^2),
   b (a^4 + c^4 - b^4 - a^2 c^2), c (b^4 + a^4 - c^4 - b^2 a^2)}, "FAR-OUT POINT"},
{"X(24)", Hold@{secA cos2A, secB cos2B, secC cos2C},
 "PERSPECTOR OF ABC AND ORTHIC-OF-ORTHIC TRIANGLE"},
{"X(25)", Hold@{a / (b^2 + c^2 - a^2), b / (c^2 + a^2 - b^2), c / (a^2 + b^2 - c^2)},
 "HOMOTHETIC CENTER OF ORTHIC AND TANGENTIAL TRIANGLES"},
{"X(26)", Hold@{a (b^2 cos2B + c^2 cos2C - a^2 cos2A),
   b (a^2 cos2A + c^2 cos2C - b^2 cos2B), c (a^2 cos2A + b^2 cos2B - c^2 cos2C)},
 "CIRCUMCENTER OF THE TANGENTIAL TRIANGLE"},
{"X(27)", Hold@{secA / (b+c), secB / (c+a), secC / (a+b)},
 "CEVAPOINT OF ORTHOCENTER AND CLAWSON CENTER"}, {"X(28)", Hold@
  {tanA / (b+c), tanB / (c+a), tanC / (a+b)}, "CEVAPOINT OF X(19) AND X(25)"},
{"X(29)", Hold@{secA / (cosB + cosC), secB / (cosC + cosA), secC / (cosA + cosB)},
 "CEVAPOINT OF INCENTER AND ORTHOCENTER"},
{"X(30)", Hold@{cosA - 2 cosB cosC, cosB - 2 cosC cosA, cosC - 2 cosA cosB},
 "EULER INFINITY POINT"},
{"X(31)", Hold@{a^2, b^2, c^2}, "2nd POWER POINT"},
{"X(32)", Hold@{a^3, b^3, c^3}, "3rd POWER POINT"},
{"X(33)", Hold@{1 + secA, 1 + secB, 1 + secC},
 "PERSPECTOR OF THE ORTHIC AND INTANGENTS TRIANGLES"},
{"X(34)", Hold@{1 - secA, 1 - secB, 1 - secC}, "X(4)-BETH CONJUGATE OF X(4)"},
{"X(35)", Hold@{1 + 2 cosA, 1 + 2 cosB, 1 + 2 cosC},
 "{X(1),X(3)}-HARMONIC CONJUGATE OF X(36)"}, {"X(36)",
```

```
 Hold@{1 - 2 cosA, 1 - 2 cosB, 1 - 2 cosC}, "INVERSE-IN-CIRCUMCIRCLE OF INCENTER"},
{"X(37)", Hold@{b + c, c + a, a + b}, "CROSSPOINT OF INCENTER AND CENTROID"},
{"X(38)", Hold@{b^2 + c^2, c^2 + a^2, a^2 + b^2},
 "CROSSPOINT OF X(1) AND X(75)"}, {"X(39)",
 Hold@{a (b^2 + c^2), b (c^2 + a^2), c (a^2 + b^2)}, "BROCARD MIDPOINT"},
{"X(40)", Hold@{cosB + cosC - cosA - 1, cosA + cosC - cosB - 1, cosA + cosB - cosC - 1},
 "BEVAN POINT"},
{"X(41)", Hold@{a^2 (b + c - a), b^2 (c + a - b), c^2 (a + b - c)},
 "X(6)-CEVA CONJUGATE OF X(31)"}, {"X(42)", Hold@{a (b + c), b (c + a), c (a + b)},
 "CROSSPOINT OF INCENTER AND SYMMEDIAN POINT"}, {"X(43)",
 Hold@{a b + a c - b c, b c + b a - c a, c a + c b - a b}, "X(6)-CEVA CONJUGATE OF X(1)"},
{"X(44)", Hold@{b + c - 2 a, c + a - 2 b, a + b - 2 c}, "X(6)-LINE CONJUGATE OF X(1)"},
{"X(45)", Hold@{2 b + 2 c - a, 2 c + 2 a - b, 2 a + 2 b - c},
 "X(9)-BETH CONJUGATE OF X(1)"},
{"X(46)", Hold@{cosB + cosC - cosA, cosA + cosC - cosB, cosA + cosB - cosC},
 "X(4)-CEVA CONJUGATE OF X(1)"}, {"X(47)", Hold@{cos2A, cos2B, cos2C},
 "X(110)-BETH CONJUGATE OF X(34)"}, {"X(48)",
 Hold@{tanB + tanC, tanA + tanC, tanA + tanB}, "CROSSPOINT OF X(1) AND X(63)"},
{"X(49)", Hold@{cos3A, cos3B, cos3C}, "CENTER OF SINE-TRIPLE-ANGLE CIRCLE"},
{"X(50)", Hold@{sin3A, sin3B, sin3C}, "X(74)-CEVA CONJUGATE OF X(184)"},
{"X(51)", Hold@{a^2 (cosB cosC + sinB sinC), b^2 (cosC cosA + sinC sinA),
   c^2 (cosA cosB + sinA sinB)}, "CENTROID OF ORTHIC TRIANGLE"},
{"X(52)", Hold@{cos2A (cosB cosC + sinB sinC), cos2B (cosC cosA + sinC sinA),
   cos2C (cosA cosB + sinA sinB)}, "ORTHOCENTER OF ORTHIC TRIANGLE"},
{"X(53)", Hold@{tanA (cosB cosC + sinB sinC), tanB (cosC cosA + sinC sinA),
   tanC (cosA cosB + sinA sinB)}, "SYMMEDIAN POINT OF ORTHIC TRIANGLE"},
{"X(54)", Hold@{1 / (cosB cosC + sinB sinC), 1 / (cosC cosA + sinC sinA),
   1 / (cosA cosB + sinA sinB)}, "KOSNITA POINT"},
{"X(55)", Hold@{a (b + c - a), b (c + a - b), c (a + b - c)},
 "INSIMILICENTER(CIRCUMCIRCLE, INCIRCLE)"},
{"X(56)", Hold@{a / (b + c - a), b / (c + a - b), c / (a + b - c)},
 "EXSIMILICENTER(CIRCUMCIRCLE, INCIRCLE)"},
{"X(57)", Hold@{1 / (b + c - a), 1 / (c + a - b), 1 / (a + b - c)},
 "ISOGONAL CONJUGATE OF X(9)"}, {"X(58)",
 Hold@{a / (b + c), b / (c + a), c / (a + b)}, "ISOGONAL CONJUGATE OF X(10)"},
{"X(59)", Hold@{1 / (1 - (cosB cosC + sinB sinC)), 1 / (1 - (cosC cosA + sinC sinA)),
   1 / (1 - (cosA cosB + sinA sinB))}, "ISOGONAL CONJUGATE OF X(11)"},
{"X(60)", Hold@{1 / (1 + (cosB cosC + sinB sinC)), 1 / (1 + (cosC cosA + sinC sinA)),
   1 / (1 + (cosA cosB + sinA sinB))}, "ISOGONAL CONJUGATE OF X(12)"}, {"X(61)",
 Hold@{(sinA cPi6 + sPi6 cosA), (sinB cPi6 + sPi6 cosB), (sinC cPi6 + sPi6 cosC)},
 "ISOGONAL CONJUGATE OF X(17)"}, {"X(62)",
 Hold@{(sinA cPi6 - sPi6 cosA), (sinB cPi6 - sPi6 cosB), (sinC cPi6 - sPi6 cosC)},
 "ISOGONAL CONJUGATE OF X(18)"},
```

```
{"X(63)", Hold@{cotA, cotB, cotC}, "ISOGONAL CONJUGATE OF X(19)"},
{"X(64)", Hold@{1 / (cosA - cosB cosC), 1 / (cosB - cosC cosA),
   1 / (cosC - cosA cosB)}, "ISOGONAL CONJUGATE OF X(20)"},
{"X(65)", Hold@{cosB + cosC, cosC + cosA, cosA + cosB},
 "ORTHOCENTER OF THE INTOUCH TRIANGLE"},
{"X(66)", Hold@{b c / (b^4 + c^4 - a^4), c a / (c^4 + a^4 - b^4),
   a b / (a^4 + b^4 - c^4)}, "ISOGONAL CONJUGATE OF X(22)"},
{"X(67)", Hold@{b c / (b^4 + c^4 - a^4 - b^2 c^2), c a / (c^4 + a^4 - b^4 - c^2 a^2),
   a b / (a^4 + b^4 - c^4 - a^2 b^2)}, "ISOGONAL CONJUGATE OF X(23)"},
{"X(68)", Hold@{cosA sec2A, cosB sec2B, cosC sec2C}, "PRASOLOV POINT"},
{"X(69)", Hold@{(cosA) / a^2, (cosB) / b^2, (cosC) / c^2},
 "SYMMEDIAN POINT OF THE ANTICOMPLEMENTARY TRIANGLE"},
{"X(70)", Hold@{b c / (b^2 cos2B + c^2 cos2C - a^2 cos2A),
   c a / (c^2 cos2C + a^2 cos2A - b^2 cos2B),
   a b / (a^2 cos2A + b^2 cos2B - c^2 cos2C)}, "ISOGONAL CONJUGATE OF X(26)"},
{"X(71)", Hold@{(b + c) cosA, (c + a) cosB, (a + b) cosC},
 "ISOGONAL CONJUGATE OF X(27)"}, {"X(72)",
 Hold@{(b + c) cotA, (c + a) cotB, (a + b) cotC}, "ISOGONAL CONJUGATE OF X(28)"},
{"X(73)", Hold@{secB + secC, secC + secA, secA + secB},
 "CROSSPOINT OF INCENTER AND CIRCUMCENTER"},
{"X(74)", Hold@{1 / (cosA - 2 cosB cosC), 1 / (cosB - 2 cosC cosA),
   1 / (cosC - 2 cosA cosB)}, "ISOGONAL CONJUGATE OF EULER INFINITY POINT"},
{"X(75)", Hold@{1 / a^2, 1 / b^2, 1 / c^2}, "ISOTOMIC CONJUGATE OF INCENTER"},
{"X(76)", Hold@{1 / a^3, 1 / b^3, 1 / c^3}, "3rd BROCARD POINT"},
{"X(77)", Hold@{1 / (1 + secA), 1 / (1 + secB), 1 / (1 + secC)},
 "ISOGONAL CONJUGATE OF X(33)"}, {"X(78)", Hold@
  {1 / (1 - secA), 1 / (1 - secB), 1 / (1 - secC)}, "ISOGONAL CONJUGATE OF X(34)"},
{"X(79)", Hold@{1 / (1 + 2 cosA), 1 / (1 + 2 cosB), 1 / (1 + 2 cosC)},
 "ISOGONAL CONJUGATE OF X(35)"},
{"X(80)", Hold@{1 / (1 - 2 cosA), 1 / (1 - 2 cosB), 1 / (1 - 2 cosC)},
 "REFLECTION OF INCENTER IN FEUERBACH POINT"},
{"X(81)", Hold@{1 / (b + c), 1 / (c + a), 1 / (a + b)},
 "CEVAPOINT OF INCENTER AND SYMMEDIAN POINT"},
{"X(82)", Hold@{1 / (b^2 + c^2), 1 / (c^2 + a^2), 1 / (a^2 + b^2)},
 "ISOGONAL CONJUGATE OF X(38)"},
{"X(83)", Hold@{b c / (b^2 + c^2), a c / (c^2 + a^2), a b / (a^2 + b^2)},
 "CEVAPOINT OF CENTROID AND SYMMEDIAN POINT"},
{"X(84)", Hold@{1 / (cosB + cosC - cosA - 1), 1 / (cosA + cosC - cosB - 1),
   1 / (cosA + cosB - cosC - 1)}, "ISOGONAL CONJUGATE OF X(40)"},
{"X(85)", Hold@{b^2 c^2 / (b + c - a), c^2 a^2 / (c + a - b), a^2 b^2 / (a + b - c)},
 "ISOTOMIC CONJUGATE OF X(9)"},
{"X(86)", Hold@{(b c) / (b + c), (c a) / (c + a), (a b) / (a + b)},
 "CEVAPOINT OF INCENTER AND CENTROID"},
```

```
    {"X(87)", Hold@{1/(a b + a c - b c), 1/(b c + b a - c a), 1/(c a + c b - a b)},
     "X(2)-CROSS CONJUGATE OF X(1)"},
    (* drawing billiard *)
    {"X(88)", Hold@{1/(b + c - 2 a), 1/(c + a - 2 b), 1/(a + b - 2 c)},
     "ISOGONAL CONJUGATE OF X(44)"},
    {"X(89)", Hold@{1/(2 b + 2 c - a), 1/(2 c + 2 a - b), 1/(2 a + 2 b - c)},
     "ISOGONAL CONJUGATE OF X(45)"}, {"X(90)",
     Hold@{1/(cosB + cosC - cosA), 1/(cosC + cosA - cosB), 1/(cosA + cosB - cosC)},
     "X(3)-CROSS CONJUGATE OF X(1)"},
    {"X(91)", Hold@{sec2A, sec2B, sec2C}, "ISOGONAL CONJUGATE OF X(47)"},
    {"X(92)", Hold@{csc2A, csc2B, csc2C},
     "CEVAPOINT OF INCENTER AND CLAWSON POINT"},
    {"X(93)", Hold@{sec3A, sec3B, sec3C}, "ISOGONAL CONJUGATE OF X(49)"},
    {"X(94)", Hold@{csc3A, csc3B, csc3C}, "ISOGONAL CONJUGATE OF X(50)"},
    {"X(95)", Hold@{b^2 c^2 1/(cosB cosC + sinB sinC),
       c^2 a^2 1/(cosC cosA + sinC sinA), a^2 b^2 1/(cosA cosB + sinA sinB)},
     "CEVAPOINT OF CENTROID AND CIRCUMCENTER"},
    {"X(96)", Hold@{sec2A 1/(cosB cosC + sinB sinC), sec2B 1/(cosC cosA + sinC sinA),
       sec2C 1/(cosA cosB + sinA sinB)}, "ISOGONAL CONJUGATE OF X(52)"},
    {"X(97)", Hold@{cotA 1/(cosB cosC + sinB sinC), cotB 1/(cosC cosA + sinC sinA),
       cotC 1/(cosA cosB + sinA sinB)}, "ISOGONAL CONJUGATE OF X(53)"}, {"X(98)",
     Hold@{b c/(b^4 + c^4 - a^2 b^2 - a^2 c^2), c a/(c^4 + a^4 - b^2 c^2 - b^2 a^2),
       a b/(a^4 + b^4 - c^2 a^2 - c^2 b^2)}, "TARRY POINT"},
    {"X(99)", Hold@{(b c)/(b^2 - c^2), (c a)/(c^2 - a^2), (a b)/(a^2 - b^2)},
     "STEINER POINT"},
    (* so called "antifeu" *)
    {"X(100)", Hold@{1/(b - c), 1/(c - a), 1/(a - b)},
     "ANTICOMPLEMENT OF FEUERBACH POINT"}
   };
  Chop[{
      #[[1]], #[[3]],
      trilinearToCartesian[orbit, {a, b, c}, N@ReleaseHold[#[[2]]]]} & /@
    (* if singles ≠ {}, use the list to only evaluate requested indices *)
    If[Length[singles] > 0, eqns[[singles]], eqns]
  ];

In[141]:= centerNames = (#[[1]] <> ": " <> #[[2]]) & /@
    Quiet@getNewCenters[{{0, 0}, {1, 0}, {1, 1}}, RotateLeft@{1, 1, Sqrt[2.]}];
```

In[142]:=
```
Clear@newCenters;
newCenters[a_, t_, singles_: {}] := Module[{orbit, normals, sides},
  {orbit, normals} = orbitNormals[a, t];
  orbit = Reverse@orbit;
  sides = RotateLeft[triLengths[orbit]];
  getNewCenters[orbit, sides, singles]
 ];
```

Basic Clrs and Drawing

In[143]:=
```
ptClrs = {
   "ell" → Black,
   "med" → Black,
   "bar" → Purple,
   "inc" → Darker@Green,
   "ort" → Orange,
   "cir" → Red,
   "npc" → RGBColor[1., 0.02, 0.3],
   "exc" → Darker@Green,
   "ex12" → Darker@Green,
   "ex23" → Darker@Green,
   "ex31" → Darker@Green,
   "feu" → RGBColor[0.5, 0.5, 0.27],
   "antifeu" → Orange,
   "x88" → Cyan,
   "nag" → Red,
   "mit" → Lighter@Blue,
   "sym" → Red,
   "ger" → Pink,
   "spi" → Darker@Cyan,
   "nap" → Red,
   "hex" → Pink,
   (* overrides *)
   "red" → Red,
   "green" → Green,
   "blue" → Blue
  };
```

In[144]:=
```
Clear@plotEll;
plotEll[a_, clr_: ("ell" /. ptClrs)] :=
  ParametricPlot[N@{a Cos[u], Sin[u]}, {u, 0.0, 2.0 π},
   PlotStyle → clr, PerformanceGoal → "Quality"];
Clear@plotEllb;
plotEllb[a_, b_, clr_: ("ell" /. ptClrs)] :=
 ParametricPlot[N@{a Cos[u], b Sin[u]}, {u, 0.0, 2.0 π},
  PlotStyle → clr, PerformanceGoal → "Quality"]
```

In[148]:=
```
drawArrow[p0_, phat_, l_] := Arrow[{p0, ray[p0, phat, l]}];
```

In[149]:=
```
Clear@txtSubscript;
txtSubscript[txt_, subscr_, size_, where_] :=
  Text[Style[Subscript[txt, subscr], size], where];
```

Draw Orbits

In[151]:=
```
Clear@drawOrbit;
drawOrbit[orbitData_, lgt_: .25, drawNormals_: True, psize_: 12] :=
  Module[{normals, orbit, halfAlpha},
   orbit = "orbit" /. orbitData;
   normals = "normals" /. orbitData;
   halfAlpha = "halfAlpha" /. orbitData;
   Graphics[{
     PointSize[Large], Arrowheads[Medium],
     {Black, Point@orbit},
     {FaceForm[GrayLevel[.95]], EdgeForm[Black], Polygon[orbit]},
     If[drawNormals,
      {Black,
       MapThread[drawArrow[#1, #2, lgt] &, {orbit, normals}],
       Dashed,
       drawArrow[orbit[[1]], "nRot" /. orbitData, lgt],
       drawArrow[orbit[[1]], "nRotNeg" /. orbitData, lgt],
       Text["α", ray[orbit[[1]], halfAlpha , lgt * .5]]}, {}],
     MapThread[txtSubscript["P", ToString@#1, psize, ray[#2, -#3, lgt * .5]] &,
      {{1, 2, 3}, orbit, normals}]}]
  ];
```

In[153]:=
```
Clear@drawOneNotable;
drawOneNotable[clr_, at_, txt_, dx_: 1.5, dy_: 0, fntSize_: 12] :=
 {clr, Point@at, Text[Style[txt, fntSize], at, {dx, dy}]};
circOneNotable[clr_, at_, rad_, txt_, dx_: 1.5, dy_: 0, fntSize_: 12] :=
  {clr, Circle[at, rad], Text[Style[txt, fntSize], at, {dx, dy}]};
```

```
Clear@drawNotableLines;
drawNotableLines[notables_] := Graphics@{Darker[Gray], Dashed,
   (* euler line: cir,bar,npc,ort *)
   Line[{"cir", "ort"} /. notables],
   (* npc to feuerbach: npc,inc,feu *)
   Line[{"npc", "feu"} /. notables],
   (* mittenpunkt to symmedian: mit,inc,sym *)
   Line[{"mit", "sym"} /. notables],
   (* gergonne to mit: ger,bar,mit *)
   Line[{"mit", "ger"} /. notables],
   (* nagel line: nag,bar,spi,inc *)
   Line[{"nag", "inc"} /. notables]};
Clear@drawNotables;
drawNotables[notables_, prefix_: ""] := Graphics[{PointSize[Large],
    drawOneNotable[(# /. ptClrs), (# /. notables), prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "feu", "ex12", "ex23",
      "ex31", "sym", "mit", "ger", "nag", "spi", "antifeu", "x88"}}];
drawNotablesSingleClr[notables_, clr_, prefix_: ""] := Graphics[{PointSize[Large],
    drawOneNotable[clr, (# /. notables), prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "feu", "ex12", "ex23",
      "ex31", "sym", "mit", "ger", "nag", "spi", "antifeu", "x88"}}];
Clear@circNotables;
circNotables[notables_, rad_, prefix_: ""] := Graphics[{PointSize[Large],
    circOneNotable[(# /. ptClrs), (# /. notables), rad, prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "feu", "ex12", "ex23",
      "ex31", "sym", "mit", "ger", "nag", "spi", "antifeu", "x88"}}];
Clear@circNotablesSingleClr;
circNotablesSingleClr[notables_, rad_, clr_, prefix_: ""] :=
  Graphics[{PointSize[Large],
    circOneNotable[clr, (# /. notables), rad, prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "feu", "ex12", "ex23",
      "ex31", "sym", "mit", "ger", "nag", "spi", "antifeu", "x88"}}];
Clear@drawSomeNotables;
drawSomeNotables[notables_, nots_, prefix_: ""] := Graphics[{PointSize[Large],
    drawOneNotable[(# /. ptClrs), (# /. notables), prefix <> #] & /@ nots}];
Clear@drawBasicNotables;
drawBasicNotables[notables_, prefix_: ""] := drawSomeNotables[
   notables, {"inc", "bar", "ort", "cir", "npc", "mit", "feu"}, prefix];
Clear@drawBasicNotablesSingleClr;
drawBasicNotablesSingleClr[notables_, clr_, prefix_: ""] :=
  Graphics[{PointSize[Large],
    drawOneNotable[clr, (# /. notables), prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "mit", "feu"}}];
```

```
Clear@circBasicNotables;
circBasicNotables[notables_, rad_, prefix_: ""] := Graphics[{PointSize[Large],
    circOneNotable[(# /. ptClrs), (# /. notables), rad, prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "mit", "feu"}}];
Clear@circBasicNotablesSingleClr;
circBasicNotablesSingleClr[notables_, rad_, clr_, prefix_: ""] :=
  Graphics[{PointSize[Large],
    circOneNotable[clr, (# /. notables), rad, prefix <> #] & /@
     {"inc", "bar", "ort", "cir", "npc", "mit", "feu"}}];
```

Poly Utils

```
In[173]:= Clear@polyVtx;
polyVtx[alphaT_, i_, fnVtx0_] := Module[{a, p1, t, alpha},
   (* ps={a Cos[toRad[#]],Sin[toRad[#]]}&/@("ts"/.alphaT); *)
   a = "a" /. alphaT;
   alpha = ("alphas" /. alphaT)[[i]];
   t = ("tsRad" /. alphaT)[[i]];
   p1 = {a Cos[t], Sin[t]};
   fnVtx0[a, p1, alpha]
  ];
```

```
In[175]:= Clear@polyError;
polyError[alphaT_, i_, fnVtx0_, fnErrorP_] := Module[{a, alpha, poly, err},
   a = "a" /. alphaT;
   alpha = ("alphas" /. alphaT)[[i]];
   poly = polyVtx[alphaT, i, fnVtx0];
   err = fnErrorP[a, poly[[1]], alpha];
   {"a" -> a, "poly" -> poly, "error" -> err}];
```

```
In[177]:= Clear@sumPolyCosines;
sumPolyCosines[alphaT_, i_, fnVtx0_] := Module[{poly},
   (* ps={a Cos[toRad[#]],Sin[toRad[#]]}&/@("ts"/.alphaT); *)
   poly = polyVtx[alphaT, i, fnVtx0];
   Total@getPolyCosines[poly]
  ];
```

In[179]:=
```
Clear@newCentersPoly;
newCentersPoly[alphaT_, i_, fnVtx0_, vtx_, singles_: {}] :=
  Module[{poly, tri, sides},
   poly = polyVtx[alphaT, i, fnVtx0];
   tri = Part[poly, vtx];
   tri = Reverse@tri;
   sides = RotateLeft@triLengths[tri];
   getNewCenters[tri, sides, singles]
  ];
```

In[181]:=
```
Clear@getLociTablePoly;
getLociTablePoly[alphaT_, vtxFn_, vtx_: {1, 2, 3}] := Module[{nc, pts, loci, lgt},
   lgt = Length["alphas" /. alphaT];
   loci = Table[
     nc = newCentersPoly[alphaT, i, vtxFn, vtx] // Quiet;
     pts = (#[[3]] & /@ nc);
     {i, pts},
     {i, lgt}];
   loci = Append[loci, ReplacePart[loci[[1]], 1 → lgt + 1]];
   Transpose[#[[2]] & /@ loci]];
```

In[183]:=
```
Clear@getCentroidPath;
getCentroidPath[alphaT_, fnVtx0_] := Module[{centroids, centroidNames, polys},
   polys = Table[polyVtx[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}];
   centroids = Transpose[getCentroids /@ polys];
   (*centroidNames={"Gvtx","Gper","Glam"};*)
   centroids];
```

In[185]:=
```
Clear@showCentroidPaths;
Options[showCentroidPaths] = {drLegend → True};
showCentroidPaths[alphaT_, fnVtx0_, OptionsPattern[]] :=
 Show[{ListLinePlot[MapThread[If[OptionValue@drLegend, Legended[#1, #2], #1] &,
     {getCentroidPath[alphaT, fnVtx0], {"vtx", "per", "area"}}]]},
  Frame → True, FrameStyle → Medium]
```

```
In[188]:= Clear@getCentroidRadialStats;
getCentroidRadialStats[alphaT_, fnVtx0_] := Module[{paths, rs, means, sds, zs},
   paths = getCentroidPath[alphaT, fnVtx0];
   rs = Map[magn, paths, {2}];
   means = Mean /@ rs;
   sds = StandardDeviation /@ rs;
   zs = MapThread[(#2 / #1) &, {means, sds}];
   {means, sds, zs}];
getCentroidRadialStatsTable[alphaT_, fnVtx0_] := Transpose@
    Prepend[getCentroidRadialStats[alphaT, fnVtx0], {"vtx", "perimeter", "area"}] //
   Prepend[#, {"type", "mean", "sd", "zscore"}] & // Grid[#, Frame → All] &

In[191]:= Clear@plotPolyAreas;
plotPolyAreas[ts_, polyAreas_, ps_] := Module[{polyMean, polySd(*,lab*)},
   polyMean = Mean@polyAreas;
   polySd = StandardDeviation@polyAreas;
   (*lab="mean="<>nfn[polyMean,3]<>", sd="<>nfn[polySd,3];*)
   ListLinePlot[Transpose@{ts, polyAreas}, Frame → True, AxesOrigin → {0, 0},
    (*PlotLabel→lab,*) PlotStyle → ps]
  ];

In[193]:= Clear@getPolyAreas;
getPolyAreas[alphaT_, fnVtx0_] := Module[{alphas, polys, polyAreas},
  polys = Table[polyVtx[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}];
  polyAreas = Area /@ Polygon /@ polys;
  MapThread[{#1, #2} &, {"tsRad" /. alphaT, polyAreas}]
 ];

In[194]:= Clear@showPolyArea;
showPolyArea[alphaT_, fnVtx0_, clr_] := Module[{a, polyA, meanA, sdA, lab},
  a = "a" /. alphaT;
  If[("a" /. alphaT) ≠ a, Print["error: as are diff"]];
  polyA = getPolyAreas[alphaT, fnVtx0];
  meanA = Mean[Second /@ polyA];
  sdA = StandardDeviation[Second /@ polyA];
  lab = Style["orbit area, a/b=" <>
     nfn[a, 2] <> ", sd/mean=" <> nfn[sdA / meanA, 4], {Black, 16}];
  Show[plotPolyAreas[First /@ polyA, Second /@ polyA, clr],
   (*PlotRange→{All,{meanA-3sdA,meanA+3sdA}},PlotLabel→lab,*)
   FrameStyle → Medium,
   FrameLabel → (Style[#, {Black, 14}] & /@ {"θ (rad)", "area"})]]
```

In[196]:=
```
Clear@reportPolyAreaStats;
reportPolyAreaStats[alphaT_, fnVtx0_] := Module[{polyA, step, mean, sd, degStep},
  polyA = getPolyAreas[alphaT, fnVtx0];
  degStep = ("tsDeg" /. alphaT)[[2]] - ("tsDeg" /. alphaT)[[1]];
  mean = Mean[Second /@ polyA];
  sd = StandardDeviation[Second /@ polyA];
  {"a" /. alphaT, degStep, Length@polyA, mean, sd, sd / mean}]
```

In[198]:=
```
Clear@getP2Alpha; getP2Alpha[a_, p1_, alpha_] :=
 Module[{norm, ca, sa, p2, normRot, normRotNeg, p2Neg},
  (*y=-ellY[a,x1];
  p1={x1,y};*)
  ca = Cos[alpha];
  sa = Sin[alpha];
  norm = ellGrad[a, p1[[1]], p1[[2]]];
  normRot = rot[norm, sa, ca];
  normRotNeg = rot[norm, -sa, ca];
  p2 = ellInterRayUnprot[a, p1, normRot][[2]]; (* get 2nd solution *)
  p2Neg = ellInterRayUnprot[a, p1, normRotNeg][[2]];
  {p2, p2Neg}];
```

```
In[1823]:= Clear@plotPolyCos;
Options[plotPolyCos] = {pert → .02, keep → All, cosDiv → 2,
   clrs -> {Black, Red, Green, Blue, Cyan, Magenta, Orange, Brown, Purple, Gray},
   polys0 → {}, ts0 → {}};
plotPolyCos[alphaT_, fnVtx0_, OptionsPattern[]] := Module[{a, ts, polys, cosPoly,
    theN, clrs0, clrs0k, plotData, ticksx, ticksxGrid, ticksy, keep0},
   a = "a" /. alphaT;
   polys = If[Length[OptionValue@polys0] > 0, OptionValue@polys0,
     Table[polyVtx[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}]];
   theN = Length[polys[[1]]];
   clrs0 = Take[OptionValue@clrs, 1 + theN];
   cosPoly = getPolyCosines /@ polys;
   keep0 = Part[Range[theN], OptionValue@keep];
   clrs0k = Part[clrs0, {1, Sequence @@ (keep0 + 1)}];
   ts = If[Length[OptionValue@ts0] > 0, OptionValue@ts0, "tsDeg" /. alphaT];
   ticksx = Table[i, {i, 0, Max[ts] + 1, 30}];
   ticksxGrid =
    ticksx /. {180 → {180, Directive[Thick, Black, Dashed, Opacity@.75]}};
   ticksy = Table[i, {i, -1.5, 1.5, .1}];
   plotData = Transpose /@
     {{ts, (Total /@ cosPoly) / OptionValue@cosDiv + OptionValue@pert},
      Sequence @@
       MapIndexed[{ts, #1 + RandomReal[{-OptionValue@pert, OptionValue@pert}]} &,
        Transpose[Part[#, keep0] & /@ cosPoly]]};
   Legended[ListLinePlot[plotData,
     Frame → True, FrameStyle → 12, PlotStyle → clrs0k,
     AspectRatio → .25,
     PlotRange → {{0, 360}, Automatic},
     FrameTicks → {{ticksy, None}, {ticksx, None}},
     GridLines → {ticksxGrid, ticksy}, ImageSize → 800,
     PlotLabel → Style["a" <> nfn[a, 2] <> ", N=" <> ToString@theN, {16, Black}]],
    LineLegend[Directive[{Thick, #}] & /@ clrs0k,
     Style[#, 16] & /@ Flatten@{"Σcos" <>
         If[OptionValue@cosDiv ≠ 1, "/" <> ToString[OptionValue@cosDiv], ""],
        ("cos(" <> # <> ")") & /@ Part[{"A", "B", "C", "D", "E",
           "F", "G", "H", "I"}, keep0]}]]];
```

Triangular Radii

In[202]:=
```
Clear@radiusIncenter;
radiusIncenter[orbit_, normals_] := Module[{ctr, foot, d},
   ctr = getIncenter[orbit[[1]], normals[[1]], orbit[[2]], normals[[2]]];
   foot = closestPerp[ctr, orbit[[1]], orbit[[2]]];
   d = magn[ctr - foot];
   {ctr, d}];
```

In[204]:=
```
Clear@radiusCircumcenter;
radiusCircumcenter[orbit_, normals_] := Module[{ctr, p1, d},
   ctr = getCircumcenter @@ orbit;
   p1 = orbit[[1]];
   d = magn[ctr - p1];
   {ctr, d}];
```

In[206]:=
```
Clear@radiusNPC;
radiusNPC[orbit_, normals_] := Module[{ctr, med12, d},
   ctr = getNinepointcenter @@ orbit;
   med12 = (orbit[[1]] + orbit[[2]]) / 2;
   (* distance to a median *)
   d = magn[ctr - med12];
   {ctr, d}];
```

In[208]:=
```
Clear@getRadii;
getRadii[a_, tmax_, radFn_] := Module[{orbit, normals, series},
   series = Table[{orbit, normals} = Evaluate[orbitNormals[a, t]];
     {t, radFn[orbit, normals][[2]]}, {t, 0, tmax, toRad[1.]}];
   series];
```

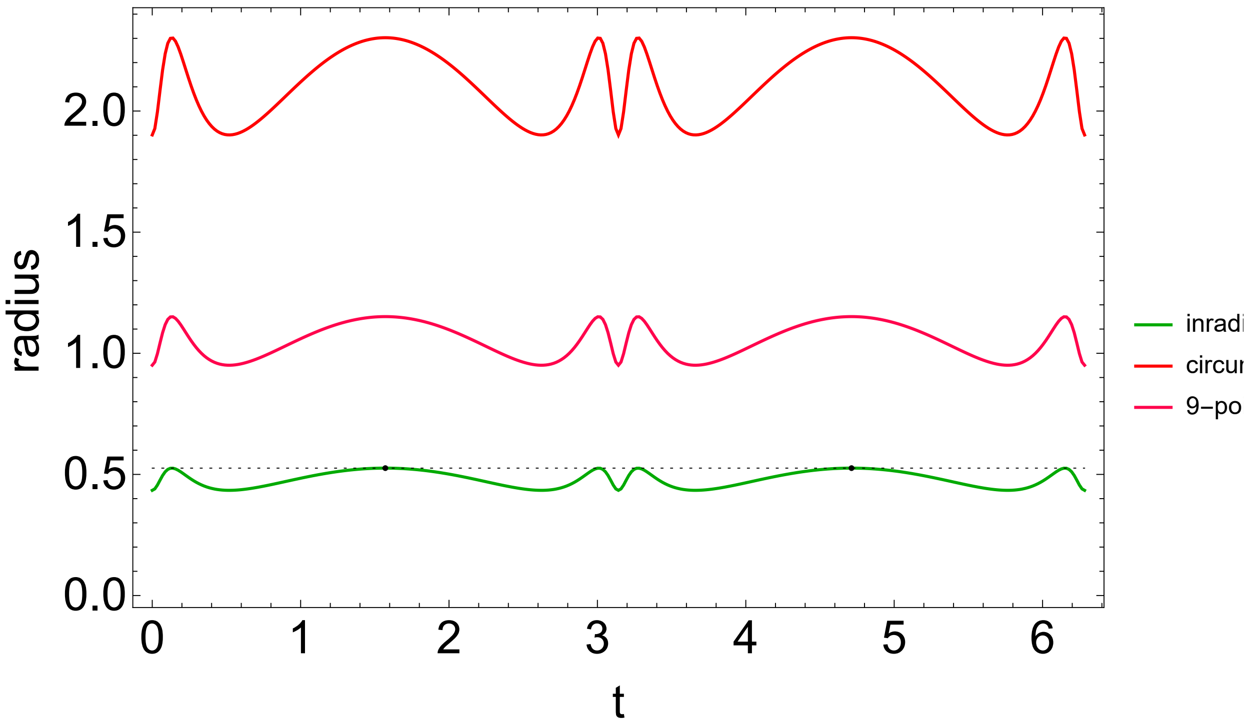
Pergunta : para qual t a area é maximizada e/ou o inradius? Parece estar ocorrendo em $\pi/2$

In[210]:=
```
inMax = Module[{rs, top},
  rs = getRadii[2, 2 π, radiusIncenter];
  top = Ordering[#[[2]] & /@ rs, {-2, -1}];
  rs[[top]]]
```

Out[210]=
```
{{1.5708, 0.525932}, {4.71239, 0.525932}}
```

In[211]:=
```
ListLinePlot[
 Map[getRadii[2, 2 π, #] &, {radiusIncenter, radiusCircumcenter, radiusNPC}],
 AxesLabel → {"t", "radius"},
 Epilog → {Point@inMax, Dotted, Line[{{0, inMax[[1, 2]]}, {2 π, inMax[[1, 2]]}}]},
 PlotLegends → {"inradius", "circumradius", "9-point-circle radius"}, Frame → True,
 ImageSize → Large, FrameStyle → Large,
 FrameLabel → {"t", "radius"},
 PlotStyle → Evaluate[(# /. ptClrs) & /@ {"inc", "cir", "npc"}]]
```

Out[211]=



* Area of Triangle A = inradius * semiperimeter = r * s

* Circumradius (R) is related to inradius (r) as follows :

$$R = \frac{a\,b\,c}{4\,r\,s}$$
$$= \frac{r}{\cos A + \cos B + \cos C - 1}$$
$$= \sqrt{\frac{a^2 + b^2 + c^2}{8\,(1 + \cos A \cos B \cos C)}},$$

In[212]:=
```
getOrbitCosines[2, 0]
```

Out[212]= {0.965424, 0.131483, 0.131483}

In[213]:=
```
getOrbitCosineSum[a_, t_] := Total[getOrbitCosines[a, t]];
```

In[214]:=
```
getOrbitCosineSum[1.01, π] - 1
```
Out[214]= 0.499889

In[215]:=
```
getOrbitCosineSum[1.5, π] - 1
```
Out[215]= 0.36266

In[216]:=
```
getOrbitCosineSum[2, π] - 1
```
Out[216]= 0.22839

Cos[A] + Cos[B] + Cos[C] - 1 is constant!

In[217]:=
```
Table[getOrbitCosineSum[1.5, t] - 1, {t, 0, 2. π, π/100}] // Mean
```
Out[217]= 0.36266

In[218]:=
```
Table[getOrbitCosineSum[2, t] - 1, {t, 0, 2. π, π/100}] // Mean
```
Out[218]= 0.22839

In[219]:=
```
Table[getOrbitCosineSum[2, t] - 1, {t, 0, 2. π, π/100}] // StandardDeviation // Chop
```
Out[219]= 0

Does the extriangle have a similar property, e.g., sum of sines is constant?

Algebraically, no

Constraint on side lengths if sum of cosines is constant

In[220]:=
```
Together[lawOfCosines[a, b, c] + lawOfCosines[b, c, a] + lawOfCosines[c, a, b]]
```
Out[220]= $-\frac{1}{a\,b\,c}0.5\left(1.\,a^3-1.\,a^2\,b-1.\,a\,b^2+1.\,b^3-1.\,a^2\,c-1.\,b^2\,c-1.\,a\,c^2-1.\,b\,c^2+1.\,c^3\right)$

Adding perimeter constancy doens' t help

In[221]:=
```
(Together[lawOfCosines[a, b, c] + lawOfCosines[b, c, a] + lawOfCosines[c, a, b]] /.
   {c → P - a - b}) // FullSimplify
```
Out[221]= $\left(0.+2.\,b^2\,P-2.\,b\,P^2+0.5\,P^3+a^2\,(-3.\,b+2.\,P)+a\left(-3.\,b^2+5.\,b\,P-2.\,P^2\right)\right)/(a\,b\,(a+b-1.\,P))$

An expression for P the perimeter

In[222]:=
```
darbouxP[a]
```
Out[222]= $$\frac{2\left(1+a^2+\sqrt{1-a^2+a^4}\right)\sqrt{-1-a^2+2\sqrt{1-a^2+a^4}}}{-1+a^2}$$

In[223]:=
```
getTriPerimeters[a_, t_] := Module[{orbit, normals, darboux, per},
  {orbit, normals} = orbitNormals[a, t];
  darboux = darbouxP[a];
  per = Total[magn /@ (orbit - RotateLeft[orbit])];
  {per, darboux}]
```

In[224]:=
```
getTriPerimeters[1.2, 0]
```

Out[224]= {5.76226, 5.76226}

In[225]:=
```
(*FullSimplify[(
   lawOfCosines[s1,s2,s3]+
    lawOfCosines[s2,s3,s1]+
    lawOfCosines[s3,s1,s2])/.
  {s3→((darbouxP[a]/.Sqrt[1-a^2+a^4]→d)-s2-s1)},
 a>0&d>0&s1>0&s2>0]*)
```

Identity: Cos[t]^2 = (cos[2 t] + 1)/2 => Cos[2t] = 2Cos[t]^2-1

In[226]:=
```
cosAlpha[a, x1]^2
```

Out[226]=
$$\frac{a^4\left(-1-a^2+2\sqrt{1-a^2+a^4}\right)}{\left(-1+a^2\right)^2\left(a^4-\left(-1+a^2\right)x1^2\right)}$$

Which means :

r / R = cos[A] + cos[B] + cos[C] - 1

In[227]:=
```
ratioInradiusCircumRadius[a_, t_] := Module[{orbit, normals, R, r},
   {orbit, normals} = orbitNormals[a, t];
   R = radiusCircumcenter[orbit, normals][[2]];
   r = radiusIncenter[orbit, normals][[2]];
   r / R];
```

Prova cosA+cosB+cosC=1+r/R=constante Prof. Ronaldo Garcia

Aplicando este resultado ao triângulo do bilhar elíptico (confirmado com cálculo simbólico) obtemos

$$\frac{r}{R} = \frac{2b^2(\delta - b^2)}{(a^2 - b^2)(a^2 + \delta)}, \quad \delta = \sqrt{a^4 - a^2 b^2 + b^4}.$$

$$\frac{r}{R} = \frac{2(\delta - b^2)(a^2 - \delta)}{a^2 - b^2}.$$

$$\cos A + \cos B + \cos C = \frac{(a^2 + b^2)(2\delta - a^2 - b^2)}{(a^2 - b^2)^2}$$

Correção 2 a expressão r/R :

$$\frac{r}{R} = \frac{2(\delta - b^2)(a^2 - \delta)}{(a^2 - b^2)^2}.$$

In[228]:=
```
getDelta[a_, b_: 1] := Module[{a2, b2},
   a2 = a^2;
   b2 = b^2;
   Sqrt[a2^2 - a2 b2 + b2^2]
  ];
getOrbitCosineSumRG[a_, b_: 1] := Module[{a2, b2, a2plusb2, delta},
   a2 = a^2;
   b2 = b^2;
   a2plusb2 = a2 + b2;
   delta = getDelta[a, b];
   a2plusb2 (2 delta - a2plusb2) / (a2 - b2) ^2];
ratioInrad2CircumRadRG[a_, b_: 1] := Module[{a2, b2, delta},
   a2 = a^2;
   b2 = b^2;
   delta = getDelta[a, b];
   2 (delta - b2) (a2 - delta) / (a2 - b2) ^2];
```

In[231]:=
```
ratioInrad2CircumRadRG0[a_, b_: 1] := Module[{a2, b2, delta},
   a2 = a^2;
   b2 = b^2;
   delta = getDelta[a, b];
   2 b2 (delta - b2) / ((a2 - b2) (a2 + delta))];
```

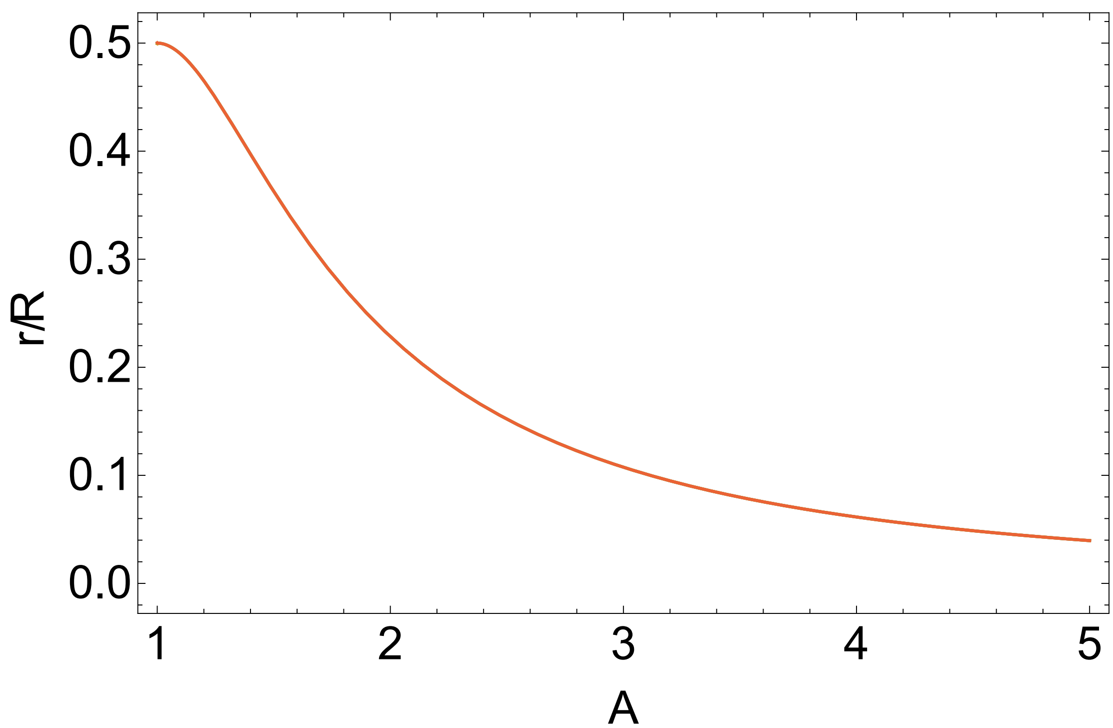
In[232]:=
```
getOrbitCosineSumRG[2.]
```
Out[232]= 1.22839

In[233]:=
```
ratioInrad2CircumRadRG[2.]
```
Out[233]= 0.22839

In[234]:=
```
ratioInrad2CircumRadRG0[2.]
```
Out[234]= 0.22839

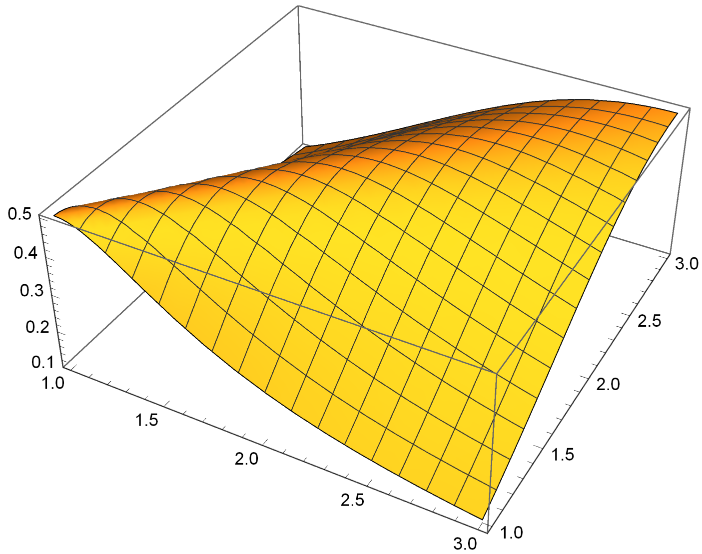
Plota razao constance : r/R = cosA + cosB + cosC - 1

In[235]:=
```
Plot[{
   getOrbitCosineSum[a, 0] - 1,
   getOrbitCosineSumRG[a] - 1,
   ratioInrad2CircumRadRG[a],
   ratioInradiusCircumRadius[a, 0]}, {a, 1, 5},
  FrameLabel → {"A", "r/R"},
  Frame → True, FrameStyle → Large, ImageSize → Large]
```
Out[235]=



In[236]:=
```
Plot3D[ratioInrad2CircumRadRG[a, b], {a, 1, 3}, {b, 1, 3}] // Quiet
```

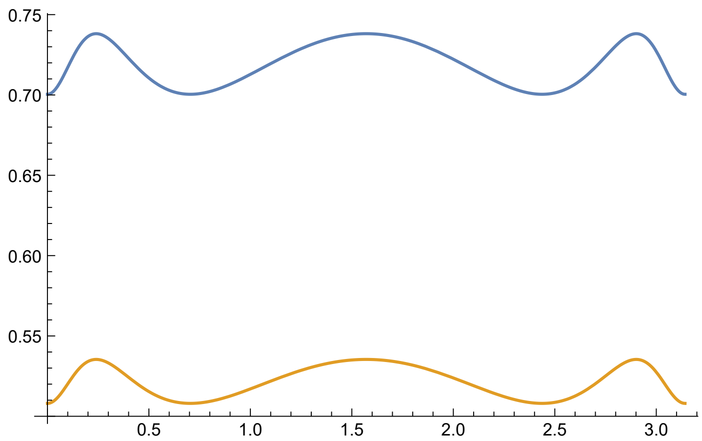
Out[236]=



In[237]:=
```
getIncNpcDist[a_, t_] := Module[{orbit, normals, inc, npc, feu},
    {orbit, normals} = orbitNormals[a, t];
    inc = radiusIncenter[orbit, normals];
    npc = radiusNPC[orbit, normals];
    feu = getFeuerbachpoint[npc[[1]], inc[[1]], inc[[2]]];
    magn[inc[[1]] - feu]
   ];
getFeuNpcDist[a_, t_] := Module[{orbit, normals, inc, npc, feu},
   {orbit, normals} = orbitNormals[a, t];
   inc = radiusIncenter[orbit, normals];
   npc = radiusNPC[orbit, normals];
   feu = getFeuerbachpoint[npc[[1]], inc[[1]], inc[[2]]];
   magn[npc[[1]] - feu]
  ]
```

In[239]:=
```
Plot[{getFeuNpcDist[1.5, t],
  getIncNpcDist[1.5, t]}, {t, 0, π}]
```

Out[239]=



Numerically Calculate Exit Angle: α

In[240]:=
```
Clear@getPolyAlphaTOneQuarter;
getPolyAlphaTOneQuarter[errFn_, a_, alpha0_, degStep_: 10, verbose_] :=
  Module[{lastAlpha, tRad, min, tab, ts, alphas, p1},
   lastAlpha = alpha0;
   tab = Table[
     tRad = toRad[tDeg];
     p1 = {a Cos[tRad], Sin[tRad]};
     min = Quiet@FindMinimum[errFn[a, p1, alpha], {alpha, lastAlpha},
        Method -> "PrincipalAxis"(*,WorkingPrecision→40*)];
     lastAlpha = (alpha /. min[[2]]);
     (*min=gradDescPentErr[a,p1,lastAlpha];*)
     (*lastAlpha=min[[3]];*)
     If[verbose, Print["t=", tDeg, "; min=", min]];
     {tDeg, tRad, lastAlpha, min},
     (* takes advantage of 4-arc symmetry *)
     {tDeg, 0, 90, degStep}];
   {
    "a" → a,
    "tsDeg" → First /@ tab,
    "tsRad" → Second /@ tab,
    "alphas" → Third /@ tab,
    "mins" → (#[[4]] & /@ tab)
   }];
```

Takes Advantage of 4-arc alpha symmetry (only computes fn from 0 to 90)

In[242]:=
```
Clear@doubleList; doubleList[vs_] := Join[vs, Rest[Reverse@vs]];
Clear@quadrupleList;
quadrupleList[vs_] := Most[doubleList@doubleList@vs];
```

In[244]:=
```
Clear@quadrupleT;
quadrupleT[theT_, doMins_: True] := Module[{degStep, radStep},
   degStep = ("tsDeg" /. theT)[[2]] - ("tsDeg" /. theT)[[1]];
   radStep = ("tsRad" /. theT)[[2]] - ("tsRad" /. theT)[[1]];
   {
    "a" → ("a" /. theT),
    "tsDeg" → Range[0, 360 - degStep, degStep],
    "tsRad" → Range[0, 2 π - radStep, radStep],
    "alphas" → quadrupleList["alphas" /. theT],
    "mins" → If[doMins, quadrupleList["mins" /. theT], "mins" /. theT]
   }];
```

In[246]:=
```
Clear@getPolyAlphaT;
getPolyAlphaT[errFn_, a_, startAlpha_, degStep_, verbose_] := Module[{theT},
   (* if the step angle is too large, the previous guess won't help *)
   theT = getPolyAlphaTOneQuarter[errFn, a, startAlpha, degStep, verbose];
   theT = quadrupleT@theT;
   theT];
```

In[248]:=
```
Clear@calcAlphaT;
calcAlphaT[doCalc_(*False for load*), errFn_, fname_,
   a_, startAlpha_, degStep_, verbose_: True] := Module[{theT},
   If[doCalc,
    (* if the step angle is too large, the previous guess won't help *)
    theT = getPolyAlphaT[errFn, a, startAlpha, degStep, verbose];
    Print["saving: ", Length["alphas" /. theT], " records to", fname];
    Save[fname, theT];
    theT,
    theT = Get[fname];
    Print["loaded: ", Length["alphas" /. theT], " records from", fname];
    theT]];
```

Tangents to Ellipse

In[250]:=
```
FullSimplify[{x, y} /.
   Solve[{ellEqnb[a, b, x, y] == 0, ellGradb[a, b, x, y].{px - x, py - y} == 0}, {x, y}],
  a > 0 && b > 0 && px ∈ Reals && py ∈ Reals]
```

CompiledFunction::cfsa : Argument a at position 1 should be a machine−size real number. >>

Out[250]=
$$\left\{\left\{\frac{a^2\left(b^2\,px-\sqrt{b^2\,px^2+a^2\left(-b^2+py^2\right)}\;Abs[py]\right)}{b^2\,px^2+a^2\,py^2},\right.\right.$$
$$\left.\frac{b^2\left(a^2\,py^2+px\sqrt{b^2\,px^2+a^2\left(-b^2+py^2\right)}\;Abs[py]\right)}{b^2\,px^2\,py+a^2\,py^3}\right\},$$
$$\left\{\frac{a^2\left(b^2\,px+\sqrt{b^2\,px^2+a^2\left(-b^2+py^2\right)}\;Abs[py]\right)}{b^2\,px^2+a^2\,py^2},\right.$$
$$\left.\left.\frac{b^2\left(a^2\,py^2-px\sqrt{b^2\,px^2+a^2\left(-b^2+py^2\right)}\;Abs[py]\right)}{b^2\,px^2\,py+a^2\,py^3}\right\}\right\}$$

In[251]:=
```
Clear@ellTangentsb;
ellTangentsb = Compile[{{a, _Real}, {b, _Real}, {p, _Real, 1}},
   Module[{a2, b2, px, py, px2, px3, py2, py3, radicand, numFact, denomx, denomy},
    {px, py} = p;
    a2 = a * a;
    b2 = b * b;
    px2 = px * px; py2 = py * py;
    px3 = px * px2; py3 = py * py2;
    denomx = b2 px2  + a2 py2;
    denomy = b2 px2 py + a2 py3;
    radicand =  b2 px2 + a2 (py2 - b2);
    numFact = Sqrt[radicand] * Abs[py];
    (*Print["radicand=",radicand,
      ", numFact=",numFact,", dx=",denomx,", dy=",denomy];*)
    (* NOTE: Reverse[] so first one is Clockwise, the 2nd one is CCW *)
    Reverse@
      {{a2 safeDiv[b2 px - numFact, denomx], b2 safeDiv[a2 py2 + px numFact, denomy]},
       {a2 safeDiv[b2 px + numFact, denomx], b2 safeDiv[a2 py2 - px numFact, denomy]}}
   ]];
```

In[253]:=
```
FullSimplify[
 {x, y} /. Solve[{ellEqn[a, x, y] == 0, ellGrad[a, x, y].{px - x, py - y} == 0}, {x, y}],
 a > 0 && px ∈ Reals && py ∈ Reals]
```

CompiledFunction::cfsa : Argument a at position 1 should be a machine-size real number. >>

Out[253]=
$$\left\{\left\{\frac{a^2 \left(px^2 + py \sqrt{px^2 + a^2 \left(-1 + py^2\right)} \text{ Abs}[px]\right)}{px^3 + a^2\, px\, py^2}, \frac{a^2\, py - \sqrt{px^2 + a^2 \left(-1 + py^2\right)} \text{ Abs}[px]}{px^2 + a^2\, py^2}\right\}, \left\{\frac{a^2 \left(px^2 - py \sqrt{px^2 + a^2 \left(-1 + py^2\right)} \text{ Abs}[px]\right)}{px^3 + a^2\, px\, py^2}, \frac{a^2\, py + \sqrt{px^2 + a^2 \left(-1 + py^2\right)} \text{ Abs}[px]}{px^2 + a^2\, py^2}\right\}\right\}$$

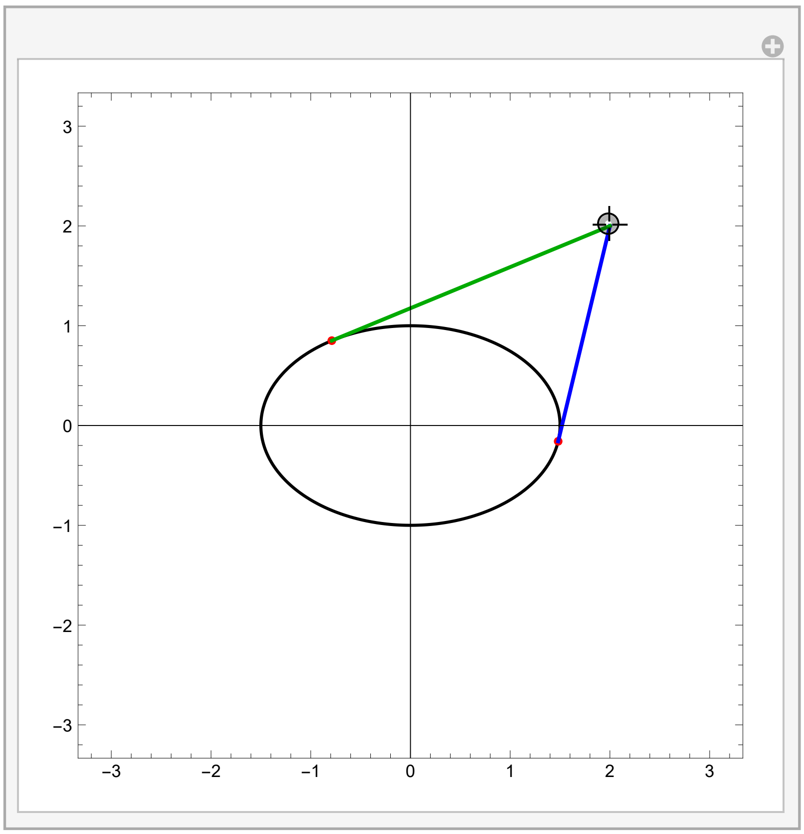
In[254]:=
```
Clear@ellTangents;
ellTangents = Compile[{{a, _Real}, {p, _Real, 1}},
   Module[{a2, px, py, px2, px3, py2, radicand, numFact, denomx, denomy},
    {px, py} = p;
    a2 = a * a;
    px2 = px * px; py2 = py * py;
    px3 = px * px2;
    denomx = px3 + a2 px py2;
    denomy = px2 + a2 py2;
    radicand = px2 + a2 (py2 - 1);
    numFact = Sqrt[radicand] * Abs[px];
    (*["radicand=",radicand,
     ", numFact=",numFact,", dx=",denomx,", dy=",denomy];*)
    (* NOTE: the first one is Clockwise, the 2nd one is CCW *)
    {{a2 safeDiv[px2 + py numFact, denomx], safeDiv[a2 py - numFact, denomy]},
     {a2 safeDiv[px2 - py numFact, denomx], safeDiv[a2 py + numFact, denomy]}}
   ]];
```

In[256]:=
```
ellTangentsProt[a_, p_] :=
  If[a > 1, ellTangents[a, p], a * perp[ellTangents[1 / a, perp@p]]];
```

In[257]:=
```
Clear@ellTangentsAB;
ellTangentsAB[a_, b_, p1_] := b * ellTangents[a / b, p1 / b];
```

In[259]:=
```
Module[{a = 1.5, tangs, ell, gr, maxX = 3, maxY = 3},
 ell = ParametricPlot[N@{a Cos[u], Sin[u]}, {u, 0.0, 2.0 π},
   PlotStyle → Black, PerformanceGoal → "Quality"];
 Manipulate[
  tangs = ellTangentsAB[a, 1, ploc];
  gr = Graphics[{PointSize@Medium, Red, Point@tangs, Thick, Blue,
     Line[{ploc, tangs[[1]]}], Darker@Green, Line[{ploc, tangs[[2]]}]}];
  Show[{ell, gr}, PlotRange → {{-maxX, maxX}, {-maxY, maxY}}, Frame -> True],
  {{ploc, {2, 2}}, Locator}]]
```

Out[259]=



Numerically Calculate α Using Caustics (Tangents to Ellipse)

In[260]:=
```
Clear@getOneCosAlphaCaustic;
getOneCosAlphaCaustic[a_, p1_, acaustic_, bcaustic_] := Module[{tangs, n},
   tangs = ellTangentsAB[acaustic, bcaustic, p1];
   n = norm[ellGrad[a, Sequence @@ p1]];
   (* 2nd tangent is clockwise,
   but it doesn't matter as the normal is bisected *)
   n.norm[(tangs[[1]] - p1)]];
```

This may take some time for high-order N-odd nonagon

In[262]:=
```
Clear@getAlpha0;
getAlpha0[a_, errCausticFn_, x1guess_] :=
  Module[{min, c2, acaustic, bcaustic, p1, p2, tangs},
   min = Quiet@FindMinimum[errCausticFn[a, x1],
      {x1, x1guess}, Method -> "PrincipalAxis"];
   c2 = a^2 - 1; (* b=1 *)
   acaustic = -(x1 /. Second[min]);
   (*acaustic^2-bcaustic^2==c^2*)
   bcaustic = Sqrt[acaustic^2 - c2];
   tangs = ellTangentsAB[acaustic, bcaustic, {a, 0}];
   p1 = {a, 0};
   p2 = ellInterRayUnprot[a, p1, tangs[[2]] - p1][[2]];
   {acaustic, bcaustic, tangs, p1, p2, min}];
```

In[264]:=
```
Clear@getCosAlphasCaustic;
getCosAlphasCaustic[a_, acaustic_, bcaustic_, step_: 1] :=
  Module[{tDegs, tRad, cossa},
   tDegs = Range[0, 90, 1];
   cossa = (tRad = toRad[#];
        getOneCosAlphaCaustic[a, {a Cos[tRad], Sin[tRad]},
         acaustic, bcaustic]) & /@ tDegs;
   cossa];
```

In[266]:=
```
Clear@getPolyAlphaCausticTOneQuarter;
getPolyAlphaCausticTOneQuarter[errCausticFn_, a_, degStep_: 10, verbose_] :=
  Module[{tsDeg, tsRad, alphas, cosAlphas, acaustic, bcaustic, min},
   tsDeg = Range[0, 359, degStep];
   tsRad = toRad /@ tsDeg;
   {acaustic, bcaustic, min} = Part[getAlpha0[a, errCausticFn, -a + .1], {1, 2, 6}];
   cosAlphas = getCosAlphasCaustic[a, acaustic, bcaustic];
   alphas = ArcCos /@ cosAlphas;
   {
    "a" → a,
    "tsDeg" → tsDeg,
    "tsRad" → tsRad,
    "alphas" → alphas,
    "min" → min
   }];
```

In[268]:=
```
Clear@getPolyAlphaCausticT;
getPolyAlphaCausticT[errCausticFn_, a_, degStep_, verbose_] := Module[{theT},
   (* if the step angle is too large, the previous guess won't help *)
   theT = getPolyAlphaCausticTOneQuarter[errCausticFn, a, degStep, verbose];
   theT = quadrupleT[theT, False];
   theT];
```

In[270]:=
```
Clear@calcAlphaCausticT;
calcAlphaCausticT[doCalc_(*False for load*),
   errCausticFn_, fname_, a_, degStep_, verbose_: True] := Module[{theT},
   If[doCalc,
    (* if the step angle is too large, the previous guess won't help *)
    theT = getPolyAlphaCausticT[errCausticFn, a, degStep, verbose];
    Print["saving: ", Length["alphas" /. theT], " records to", fname];
    Save[fname, theT];
    theT,
    theT = Get[fname];
    Print["loaded: ", Length["alphas" /. theT], " records from", fname];
    theT]];
```

Relaxation Algm for Alpha w/ Caustics

In[272]:=
```
Clear@normErr;
normErr[p_, pl_, pr_, norm_] := getBisector[pl - p, pr - p].perp[norm];
```

```
In[274]:= getRelaxErr[a_, ts_] := Module[{ps, lower, psAug, norms, bis},
    ps = getEllPs[a, ts];
    lower = flipY[Last@ps];
    psAug = {{a, 0}, Sequence @@ ps, lower};
    norms = norm /@ (ellGrad[a, #[[1]], #[[2]]] & /@ ps);
    bis = Table[normErr[psAug[[i]], psAug[[i - 1]], psAug[[i + 1]], norms[[i - 1]]]^2,
      {i, 2, Length[psAug] - 1}];
    Total@
     bis];
```

What is an ellipse' s parameters which pass thru point x0, y0 and has focal length c

```
In[275]:= Clear@getConfocalEll; getConfocalEll[c_, {x0_, y0_}] = Module[{eqn1, eqn2, ab},
   eqn1 = ac^2 - bc^2 == c^2;
   eqn2 = ellError[ac, bc, {x0, y0}] == 0;
   ab = {ac, bc} /. Quiet@Solve[{eqn1, eqn2}, {ac, bc}, Reals]]
```

Out[275]=

$$\left\{\left\{\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,1\right],\right.\right.$$
$$\left.-\sqrt{-c^2+\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,1\right]^2}\right\},$$
$$\left\{\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,1\right],\right.$$
$$\left.\sqrt{-c^2+\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,1\right]^2}\right\},$$
$$\left\{\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,4\right],\right.$$
$$\left.-\sqrt{-c^2+\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,4\right]^2}\right\},$$
$$\left\{\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,4\right],\right.$$
$$\left.\left.\sqrt{-c^2+\text{Root}\left[c^2\,x0^2+\left(-c^2-x0^2-y0^2\right)\#1^2+\#1^4\,\&,\,4\right]^2}\right\}\right\}$$

```
In[1913]:= Clear@getTangCW;
getTangCW[a_, p1_, acaustic_, bcaustic_] := Module[{tangs, norm},
    tangs = ellTangentsAB[acaustic, bcaustic, p1];
    norm = ellGrad[a, Sequence @@ p1];
    If[(tangs[[1]] - p1).perp[norm] > 0, tangs[[1]], tangs[[2]]]];
```

```
In[278]:= Clear@getInitTs;
getInitTs[a_, n_] := Module[{tstep, ts},
    tstep = π / ((n + 1) / 2.);
    ts = Table[i * tstep, {i, 1, (n - 1) / 2}];
    ts];
```

In[280]:=
```
Clear@getInitTs2;
getInitTs2[a_, n_] := Module[{tstep, ts, c, xmin, xmax, xstep, xs},
   c = If[a < 1, .25 * a, Sqrt[a^2 - 1]];
   xmax = (c + a) / 2;
   xmin = -xmax;
   xstep = (xmax - xmin) 2 / (n - 1);
   xs = Table[xmax - i * xstep, {i, 1, (n - 1) / 2}];
   ts = ArcCos[# / a] & /@ xs;
   ts];
```

In[282]:=
```
getInitTs[1.5, 7]
```

Out[282]=
```
{0.785398, 1.5708, 2.35619}
```

In[283]:=
```
getInitTs2[1.5, 7]
```

Out[283]=
```
{1.27564, 1.86596, 2.63146}
```

In[284]:=
```
Clear@getCausticAxes;
getCausticAxes[a_, p1_, p2_] := Module[{f1, f2, f2p, pb, causticAxes},
   {f1, f2} = getFoci[a];
   f2p = reflAboutLine[f2, p1, p2];
   (*f2foot=.5(f2+f2p);*)
   pb = interRays[f1, f2p - f1, p1, p2 - p1];
   causticAxes =
    First@Select[getConfocalEll[Sqrt[a^2 - 1], pb], (#[[1]] > 0 && #[[2]] > 0) &];
   causticAxes];
```

```
In[286]:= Clear@getCausticOddN;
getCausticOddN[a_, n_] := Module[{ts, ps, p1, lower, norms, tsyms,
    min, p2, f1, f2, (*f2foot,*)f2p, pb, causticAxes, poly},
  ts = getInitTs2[a, n];
  ps = getEllPs[a, ts];
  lower = flipY[Last@ps];
  p1 = {a, 0};
  norms = norm /@ (ellGrad[a, #[[1]], #[[2]]] & /@ps);
  tsyms = Table[Symbol["t" <> ToString[i]], {i, Length@ps}];
  min = Quiet@FindMinimum[getRelaxErr[a, tsyms],
     MapThread[{#1, #2} &, {tsyms, ts}]];
  p2 = getEllPs[a, tsyms /. min[[2]]][[1]];
  {f1, f2} = getFoci[a];
  (*f2foot=closestPerp[f2,p1,p2];
  f2p=f2+2(f2foot-f2);*)
  f2p = reflAboutLine[f2, p1, p2];
  (*f2foot=.5(f2+f2p);*)
  pb = interRays[f1, f2p - f1, p1, p2 - p1];
  causticAxes =
   First@Select[getConfocalEll[Sqrt[a^2 - 1], pb], (#[[1]] > 0 && #[[2]] > 0) &];
  poly = Chop@bounceRay[a, p1, p2, n - 1];
  {"a" → a,
   "n" → n,
   "causticAB" → causticAxes,
   "poly" → poly,
   "error" → min[[1]],
   (* for geometric drawing *)
   "f1" → f1,
   "f2" → f2,
   "f2p" → f2p,
   "pb" → pb,
   "guessPoly" → {p1, Sequence @@ ps, lower},
   "guessNorms" → norms}
 ];
```

In[287]:=
```
getRelaxErrEven[a_, ts_, mod4_] := Module[{ps, left, psAug, norms, bis},
   ps = getEllPs[a, ts];
   left = If[mod4, {0, 1}, flipX[Last@ps]];
   psAug = {{a, 0}, Sequence @@ ps, left};
   norms = norm /@ (ellGrad[a, #[[1]], #[[2]]] & /@ ps);
   bis = Table[normErr[psAug[[i]], psAug[[i - 1]], psAug[[i + 1]], norms[[i - 1]]]^2,
     {i, 2, Length[psAug] - 1}];
   Total@
    bis];
```

In[288]:=
```
Clear@getInitTsEven;
getInitTsEven[a_, n_] := Module[{slices, tstep, ts},
   slices = Floor[(n / 2 + 1) / 2];
   tstep = (π / 2.) / slices;
   ts = Table[i * tstep, {i, 1, slices - 1}];
   ts];
```

```
In[290]:= Clear@getCausticEvenN;
getCausticEvenN[a_, n_] := Module[{mod4, ts, ps, p1, left, norms,
    tsyms, min, p2, f1, f2, (*f2foot,*)f2p, pb, causticAxes, poly},
   mod4 = Mod[n, 4] == 0;
   p1 = {a, 0};
   If[n > 4,
    ts = getInitTsEven[a, n];
    ps = getEllPs[a, ts];
    left = If[mod4, {0, 1}, flipX[Last@ps]];
    norms = norm /@ (ellGrad[a, #[[1]], #[[2]]] & /@ps);
    (* parou aqui: será que otimizacao esta correta *)
    tsyms = Table[Symbol["t" <> ToString[i]], {i, Length@ps}];
    min = Quiet@FindMinimum[getRelaxErrEven[a, tsyms, mod4],
       MapThread[{#1, #2} &, {tsyms, ts}]];
    p2 = getEllPs[a, tsyms /. min[[2]]][[1]],
    (* n=4 special case *)
    ps = {{0, 1}};
    norms = {{-1, 0}};
    p2 = {0, 1};
    left = {-a, 0};
    min = {0, {}};
   ];
   {f1, f2} = getFoci[a];
   f2p = reflAboutLine[f2, p1, p2];
   pb = interRays[f1, f2p - f1, p1, p2 - p1];
   causticAxes =
    First@Select[getConfocalEll[Sqrt[a^2 - 1], pb], (#[[1]] > 0 && #[[2]] > 0) &];
   poly = Chop@bounceRay[a, p1, p2, n - 1];
   {"a" → a,
    "n" → n,
    "causticAB" → causticAxes,
    "poly" → poly,
    "error" → min[[1]],
    (* for geometric drawing *)
    "f1" → f1,
    "f2" → f2,
    "f2p" → f2p,
    "pb" → pb,
    "guessPoly" → {p1, Sequence @@ps, left},
    "guessNorms" → norms}
  ];
```

In[291]:=
```
Clear@rotCaustic; rotCaustic[caustic_] := Module[{a},
  a = 1 / ("a" /. caustic);
  {
   "a" → a,
   "n" → ("n" /. caustic),
   "causticAB" → a * Reverse["causticAB" /. caustic],
   "poly" → a * (perp /@ ("poly" /. caustic)),
   "error" → ("error" /. caustic),
   (* for geometric drawing *)
   "f1" → a * perp["f1" /. caustic],
   "f2" → a * perp["f2" /. caustic],
   "f2p" → a * perp["f2p" /. caustic],
   "pb" → a * perp["pb" /. caustic],
   "guessPoly" → a * (perp /@ ("guessPoly" /. caustic)),
   "guessNorms" → a * (perp /@ ("guessNorms" /. caustic))
  }];
```

In[292]:=
```
Clear@getCausticBoth;
getCausticBoth[a_, n_] := Module[{a0, caustic},
   a0 = If[a < 1, 1 / a, a];
   caustic = If[EvenQ[n], getCausticEvenN[a0, n], getCausticOddN[a0, n]];
   If[a < 1, rotCaustic[caustic], caustic]];
```

Do caustic touchpoints configure orbit within caustic?

In[1915]:=
```
Clear@getPolyCaustic; getPolyCaustic[a_, n_, tRad_, causticAB_] :=
 Module[{p1rot, tang, p2rot, bouncedRot, normsRot},
  p1rot = {a Cos[tRad], Sin[tRad]};
  (* may not be always picking the right tangent *)
  tang = getTangCW[a, p1rot, Sequence @@ causticAB];
  p2rot = ellInterRayUnprot[a, p1rot, tang - p1rot][[2]];
  bouncedRot = bounceRay[a, p1rot, p2rot, n - 1];
  normsRot = norm /@ (ellGrad[a, #[[1]], #[[2]]] & /@ bouncedRot);
  {bouncedRot, normsRot}
 ];
```

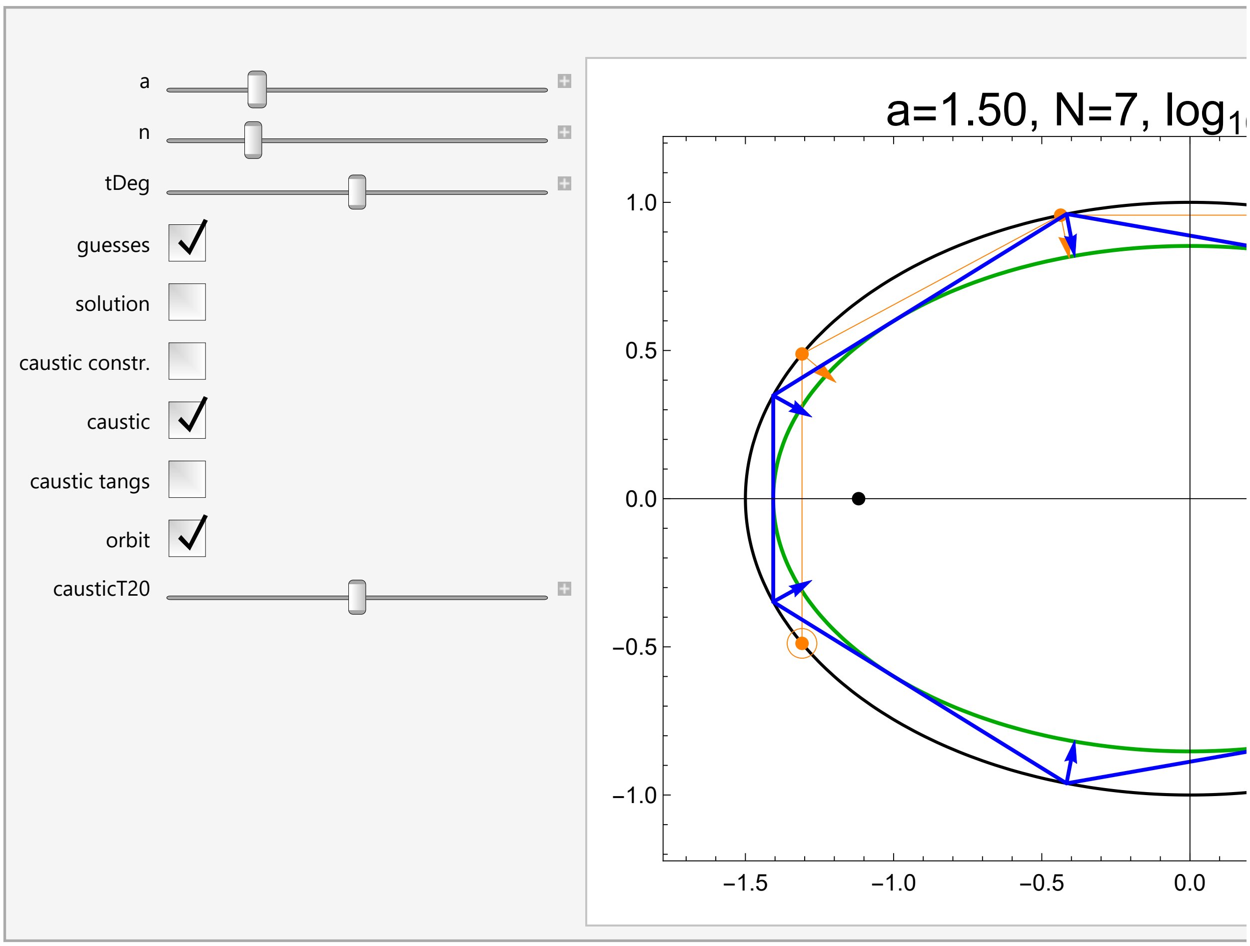
In[295]:=
```
Clear@getCausticTangs;
getCausticTangs[f1_, f2_, orbit_] :=
  MapThread[Module[{f2p, pb},
     f2p = reflAboutLine[f2, #1, #2];
     pb = interRays[f1, f2p - f1, #1, #2 - #1];
     pb] &, {Most@orbit, Rest@orbit}];
```

```
In[297]:= bounceRayAB[a_, b_, p1_, p2_, n_] :=
  Module[{A = a / b, p1b = p1 / b, p2b = p2 / b, bounced},
   bounced = bounceRay[A, p1b, p2b, n];
   b * bounced];
 ellInterAB[a_, b_, from_, dir_] := b * ellInterRayUnprot[a / b, from / b, dir];

In[299]:= Clear@drawCausticOddN;
 Options[drawCausticOddN] =
   {drSol → False, drCaustic → False, drCausticConstr → False, drGuesses → False,
    drOrbit → False, drPlotLabel → False, drCausticTangs → False, causticT2 → π / 4};
 drawCausticOddN[a_, n_, tDeg_, OptionsPattern[]] :=
  Module[{causticData, bouncedRot, normsRot, causticAB, ps,
    norms, psAug, bounced, normsSol, causticTangs, causticTangPoly,
    (*ts,ps,f1,f2,f2foot,pb,f2p,psSol,psAug,gr,p1,min,norms,normsSol,
    bis,lower,bounced,normsRot,bouncedRot,tsyms,causticAxes,*)
    causticEll, p1Caustic, p2Caustic, polyCaustic, lab, lgt = .15, circRad = .05},
   causticData = getCausticBoth[a, n];
   causticAB = "causticAB" /. causticData;
   psAug = "guessPoly" /. causticData;
   ps = Rest@Most@psAug;
   norms = "guessNorms" /. causticData;
   bounced = "poly" /. causticData;
   normsSol = norm /@ (ellGrad[a, #[[1]], #[[2]]] & /@ bounced);
   {bouncedRot, normsRot} = getPolyCaustic[a, n, toRad@tDeg, causticAB];
   causticTangs =
    getCausticTangs["f1" /. causticData, "f2" /. causticData, bouncedRot];
   causticTangPoly = bounceRayAB[Sequence @@ causticAB,
     First@causticTangs, Second@causticTangs, n - 1];
   p1Caustic = {causticAB[[1]], 0};
   p2Caustic = {causticAB[[1]] Cos[OptionValue@causticT2],
     causticAB[[2]] Sin[OptionValue@causticT2]};
   (*ellInterAB[Sequence@@causticAB,n1CausticInter,
      bouncedRot[[2]]-bouncedRot[[1]]][[2]];*)
   polyCaustic = bounceRayAB[Sequence @@ causticAB, p1Caustic, p2Caustic, n - 1];
   causticEll = If[OptionValue@drCaustic,
     plotEllb[Sequence @@ causticAB, {Thick, Darker@Green}], {}];
   gr = Graphics[{PointSize@Large, Arrowheads@Medium,
      If[OptionValue@drGuesses, {Orange, Point@psAug, Line[psAug],
        Circle[First@psAug, circRad], Circle[Last@psAug, circRad],
        MapThread[Arrow[{#1, ray[#1, #2, lgt]}] &, {ps, norms}]}, {}],
      If[OptionValue@drSol,
       {{Blue, Thick, {If[OptionValue@drOrbit, Dashed, {}], Line@bounced},
         Point@bounced, If[OptionValue@drOrbit, {},
```

```
      MapThread[Arrow[{#1, ray[#1, #2, lgt]}] &, {bounced, normsSol}]]},
     Red, Point@Take[bounced, {1 + (n + If[EvenQ[n], 0, 1]) / 2, n}]}, {}],
   {Black, Point[{"f1", "f2"} /. causticData]},
   If[OptionValue@drCausticConstr,
    {Thick, Blue,
     Point[{bounced[[1]], bounced[[2]]}], Line[{bounced[[1]], bounced[[2]]}],
     {Black, Dashed, Line[{"f2", "f2p"} /. causticData],
      Line[{"f1", "f2p"} /. causticData]},
     {Darker@Green, Point["pb" /. causticData]},
     {Black, Point["f2p" /. causticData]},
     {Black, MapThread[Text[Style[#1, 16], #2, {0, -1.75}] &,
       {{"f₁", "f₂", "f₂'", "P₁", "P₂", "b"},
        {Sequence @@ ({"f1", "f2", "f2p"} /. causticData),
         bounced[[1]], bounced[[2]], "pb" /. causticData}}]}}, {}],
   If[OptionValue@drOrbit, {Blue, Thick, Line@bouncedRot,
     Point@First[bouncedRot],
     MapThread[Arrow[{#1, ray[#1, #2, lgt]}] &, {bouncedRot, normsRot}]}, {}],
   If[OptionValue@drCausticTangs, {Darker@Green, Point@causticTangs, Circle[
      First@causticTangs, circRad], (*Dashed,Red,Line@causticTangPoly,*)
     Magenta, Point[flipX /@ {p1Caustic, p2Caustic}],
     (*Line@polyCaustic}*)Line[flipX /@ polyCaustic] (*,
     Dotted,Line[{bouncedRot[[(n+1)/2]],{0,0}}]*)}, {}]
  }];
lab = "a=" <> nfn[a, 2] <> ",  N=" <> ToString@n <>
   ",  log₁₀(err)=" <> nfn[Log10["error" /. causticData], 1];
Show[{plotEll[a, Black], causticEll, gr},
 If[OptionValue@drPlotLabel, PlotLabel → Style[lab, {Black, Large}], {}],
 Frame → True, FrameStyle → Medium,
 ImageSize → Large, PlotRange → {{-a - .1, a + .1}, {-1.1, 1.1}}]]
```

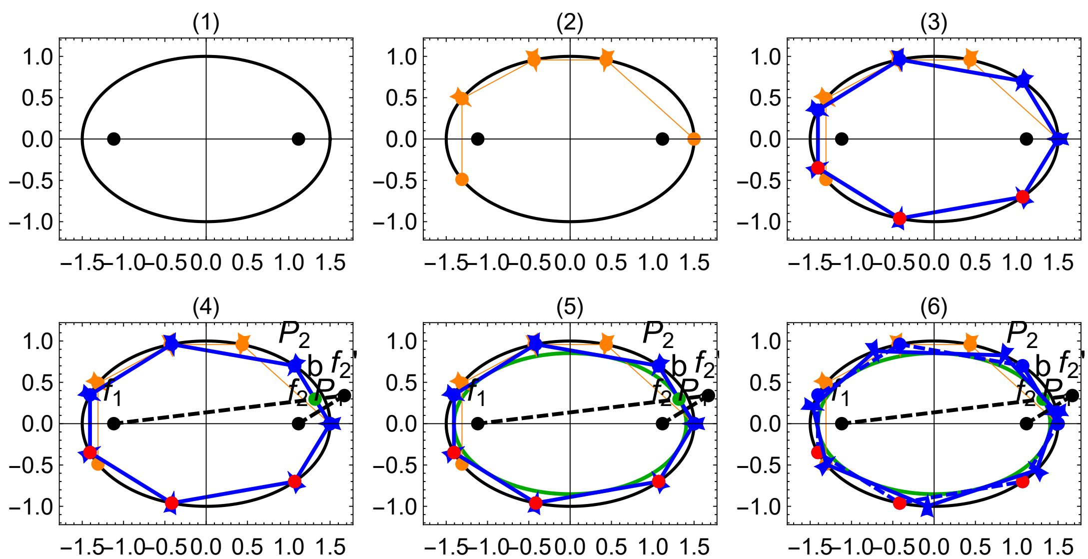
In[302]:=
```
Manipulate[drawCausticOddN[a, n, tDeg, drGuesses → drGuesses0, drSol → drSol0,
   drCausticConstr → drCausticConstr0,
   drCaustic → drCaustic0, drCausticTangs → drCausticTangs0,
   causticT2 → causticT20, drOrbit → drOrbit0, drPlotLabel → True],
  {{a, 1.5}, 1.1, 3, .01},
  {{n, 7}, 3, 23, 1},
  {{tDeg, 0}, -360, 360, 1},
  {{drGuesses0, True, "guesses"}, {True, False}},
  {{drSol0, False, "solution"}, {True, False}},
  {{drCausticConstr0, False, "caustic constr."}, {True, False}},
  {{drCaustic0, True, "caustic"}, {True, False}},
  {{drCausticTangs0, False, "caustic tangs"}, {True, False}},
  {{drOrbit0, True, "orbit"}, {True, False}},
  {{causticT20, π / 4.}, 0, π/2., π/100.}]
```

Out[302]=



In[303]:=
```
Module[{a = 1.5, tDeg = 8, n = 7, params},
 params =
  Table[Flatten[{ConstantArray[True, i], ConstantArray[False, 5 - i]}], {i, 0, 5}];
 Grid@Partition[Row[#, Spacer[20], Alignment → {Right, Center}] & /@
    MapIndexed[{Show[drawCausticOddN[a, n, tDeg,
          Thread[{drGuesses, drSol, drCausticConstr, drCaustic, drOrbit} → #1]],
         ImageSize → Small,
         PlotLabel → Style["(" <> ToString[First@#2] <> ")", {Black, Medium}]]} &,
     params], 3]]
```

Out[303]=



Pencil of Caustics and Tangents

In[304]:=
```
Clear@drawCausticPencil;
drawCausticPencil[a_, nmin_, nmax_, ps_: {Darker@Green}] :=
  Show[MapThread[plotEllb[#1, #2, ps] &,
    Transpose@Table["causticAB" /. getCausticBoth[a, n], {n, nmin, nmax}]]];
```

In[306]:=
```
Clear@drawCausticPencilAB;
drawCausticPencilAB[causticsAB_, ps_: {Darker@Green}] :=
  Show[plotEllb[Sequence @@ #, ps] & /@ causticsAB];
```

In[308]:=
```
getCausticY[a_, n_, x_] := Module[{acaustic, bcaustic},
   {acaustic, bcaustic} = ("causticAB" /. getCausticBoth[a, n]);
   ellYb[acaustic, bcaustic, x]];
```

In[309]:=
```
Clear@ellEqn5t;
ellEqn5t[{x_, y_}] :=
  (b2 ((x - x0) Cos@t - (y - y0) Sin@t) ^2 + a2 ((y - y0) Cos@t + (x - x0) Sin@t) ^2 - a2 b2) ^
   2;
```

In[311]:=
```
Clear@solveEll5t;
solveEll5t[ps_] := Quiet@NMinimize[{Total[ellEqn5t /@ps], a2 > 1 && b2 > 1},
   {a2, b2, x0, y0, t} ∈ Reals];
```

In[312]:=
```
solveEll5t[{{-2, 0}, {0, -1}, {2, 0}, {0, 1}, {2 Cos[π/2.], Sin[π/2]}}]
```

Out[312]= $\{5.80069\times10^{-15}, \{a2\to4., b2\to1., x0\to-9.81809\times10^{-18}, y0\to-1.94332\times10^{-9}, t\to-1.90542\times10^{-9}\}\}$

Show parametric plot of tangent points if a > b

In[313]:=
```
osculatingCircle[a_, b_, t_] := Module[{s = Sin@t, c = Cos@t, frac},
   frac = ((a s) ^2 + (b c) ^2) / (a b);
   {c (a - b frac), s (b - a frac)}];
```

In[314]:=
```
osculatingCircle[a, b, 0]
```

Out[314]= $\left\{a-\frac{b^2}{a}, 0\right\}$

In[315]:=
```
FullSimplify[(a + osculatingCircle[a, b, 0][[1]]) / 2, a > 0 && b > 0]
```

Out[315]= $a-\frac{b^2}{2\,a}$

In[316]:=
```
osculatingCircle[a, b, π/2]
```

Out[316]= $\left\{0, -\frac{a^2}{b}+b\right\}$

In[317]:=
```
FullSimplify[(b + osculatingCircle[a, b, π/2][[2]]) / 2, a > 0 && b > 0]
```

Out[317]= $-\frac{a^2}{2\,b}+b$

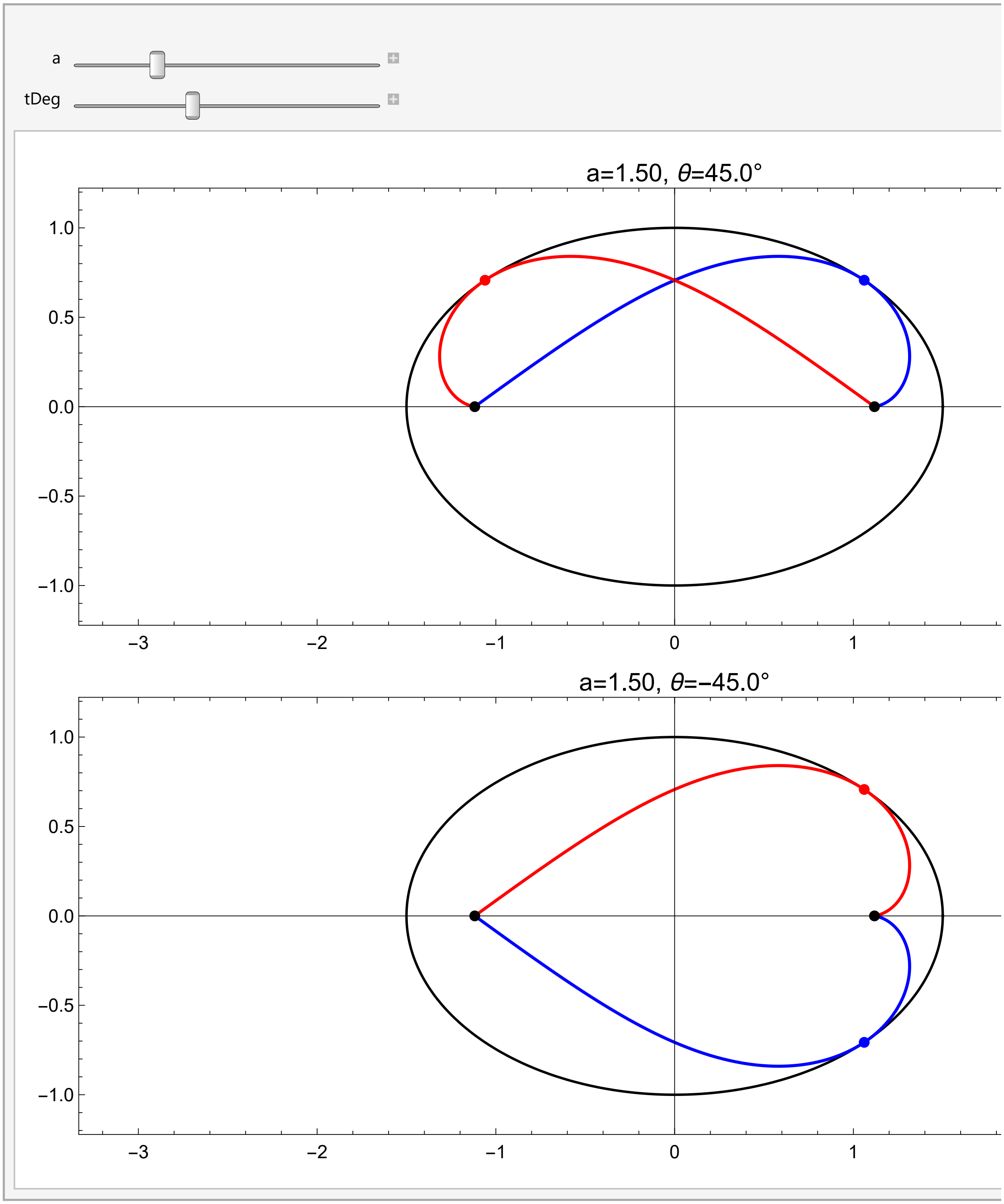
Radius of osculating circles = diameter of each of the circular loci of caustic tangents!

In[318]:=
```
Clear@showOneCTC; showOneCTC[a_, b_, tDeg_] :=
 Module[{t, c2, llp, llpTop, bconf, tangs, p1, p1Top, fs, gr, lab},
  t = toRad@tDeg;
  c2 = If[a > b, a^2 - b^2, b^2 - a^2];
  p1 = {a Cos@t, Sin@t};
  p1Top = {a Cos[π / 2 + t], Sin[π / 2 + t]};
  (*llp=Quiet@ListLinePlot[
      Transpose@Append[Table[
          bconf=Sqrt[aconf^2-c2];
          tangs=ellTangentsAB[aconf,bconf,p1];
          tangs,
          {aconf,Sqrt[c2],a,step}],{p1,p1}],PlotStyle→Thick];*)
  llp = Quiet@ParametricPlot[
     bconf = Sqrt[aconf^2 - c2];
     tangs = ellTangentsAB[aconf, bconf, p1];
     tangs,
     {aconf, Sqrt[c2], a}, PlotStyle → {Thick, Blue}];
  llpTop = Quiet@ParametricPlot[
     bconf = Sqrt[aconf^2 - c2];
     tangs = ellTangentsAB[aconf, bconf, p1Top];
     tangs,
     {aconf, Sqrt[c2], a}, PlotStyle → {Thick, Red}];
  lab = "a=" <> nfn[a, 2] <> ", θ=" <> nfn[toDeg[t], 1] <> "°";
  fs = getFoci[a / b];
  gr = Graphics@{PointSize@Large, Point@fs, Blue, Point@p1, Red, Point@p1Top};
  Show[{plotEllb[a, b], llp, llpTop, gr}, Frame → True, FrameStyle → Medium,
   PlotLabel → (Style[lab, {Black, 16}]),
   PlotRange → {{-a, a}, {-b, b}}]];
```

```
In[319]:= Clear@tangentPathFrame;
Options@tangentPathFrame = {showLeft → True, showRight → True, imgSize → 400};
tangentPathFrame[a_, tDegSmall_, tDeg_, xmax_, OptionsPattern[]] := Module[{pr, sz},
   sz = OptionValue@imgSize;
   pr = {{-xmax, xmax}, {-1.1, 1.1}};
   Grid[Transpose@{
      If[OptionValue@showLeft,
       {Show[showOneCTC[a, 1, 0], PlotRange → pr, ImageSize → sz],
        Show[showOneCTC[a, 1, tDegSmall], PlotRange → pr, ImageSize → sz]},
       Sequence @@ {}],
      If[OptionValue@showRight, {Show[showOneCTC[a, 1, 90 + tDeg],
         PlotRange → pr, ImageSize → sz],
        Show[showOneCTC[a, 1, tDeg], PlotRange → pr, ImageSize → sz]}, Sequence @@ {}]
     }]];
```

In[322]:=
```
Manipulate[tangentPathFrame[a, 5, tDeg, 3, showLeft → False, imgSize → 800],
 {{a, 1.5}, 1, 3., .01},
 {{tDeg, -45}, -180, 180, 1}]
```

Out[322]=

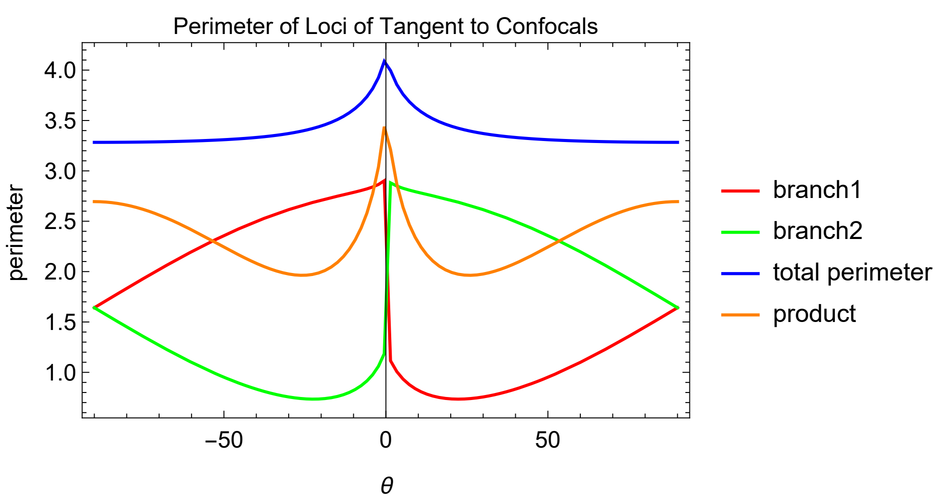


Investigate Perimeter of Family

In[323]:=
```
Clear@perimeterCTC; perimeterCTC[a_, b_, t_] := Module[{c2, bconf, tangs, p1, locus,
    branch1, per1, branch2, per2},
  c2 = If[a > b, a^2 - b^2, b^2 - a^2];
  p1 = {a Cos@t, Sin@t};
  locus = Append[Table[bconf = Sqrt[aconf^2 - c2];
     tangs = ellTangentsAB[aconf, bconf, p1];
     tangs,
     {aconf, Sqrt[c2], a, .01}], {p1, p1}];
  branch1 = First /@ locus;
  branch2 = Second /@ locus;
  per1 = Total@MapThread[magn[#1 - #2] &, {Rest@branch1, Most@branch1}];
  per2 = Total@MapThread[magn[#1 - #2] &, {Rest@branch2, Most@branch2}];
  {per1, per2, per1 + per2, per1 * per2}
 ];
```

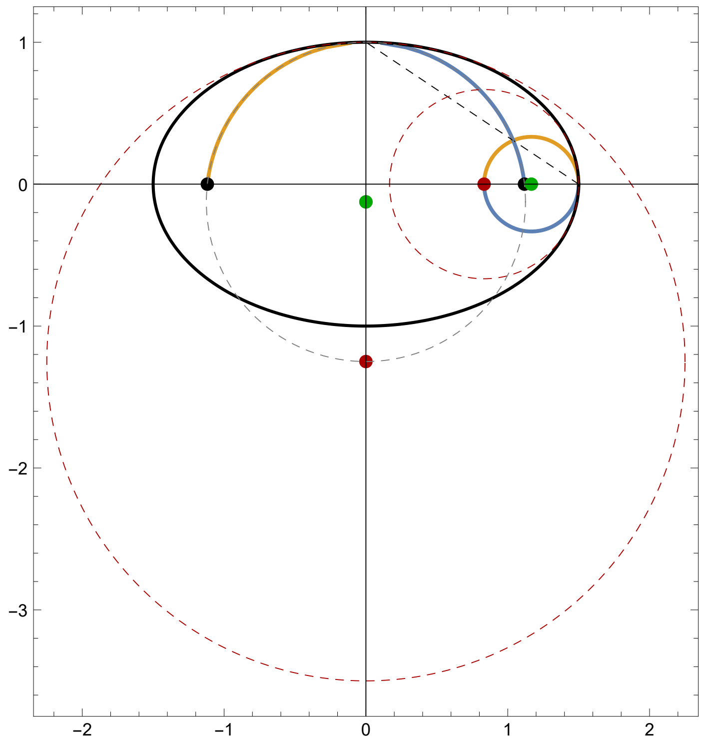
In[324]:=
```
Plot[Evaluate@Quiet@perimeterCTC[1.5, 1., toRad@t],
 {t, -90, 90}, PlotStyle → {Red, Green, Blue, Orange},
 PlotLegends → {"branch1", "branch2", "total perimeter", "product"},
 PerformanceGoal → "Speed", Frame → True, FrameStyle → Medium,
 FrameLabel → {"θ", "perimeter"}, MaxRecursion → 2,
 PlotLabel → Style["Perimeter of Loci of Tangent to Confocals", {Black, Medium}]]
```

Out[324]=



```
In[325]:= Module[{a = 1.5, b = 1, c2, bconf, tangs, lpRight, lpTop, fs, gr,
    i = 0, oscR, oscRrad, oscT, oscTrad, circumT, circumTrad, cr, ct},
   oscR = osculatingCircle[a, b, 0];
   oscRrad = a - oscR[[1]];
   oscT = osculatingCircle[a, b, π/2];
   oscTrad = b - oscT[[2]];
   fs = getFoci[a / b];
   circumT = getCircumcenter[{0, b}, fs[[1]], fs[[2]]];
   circumTrad = b - circumT[[2]];
   cr = {(2 a^2 - b^2) / (2 a), 0};
   ct = {0, (2 b^2 - a^2) / (2 b)};
   gr = Graphics[{PointSize@Large, Point@fs,
      {Dashed, Line[{{a, 0}, {0, 1}}]},
      {Darker@Red, Dashed,
       MapThread[{Point@#1, Circle[#1, #2]} &, {{oscR, oscT}, {oscRrad, oscTrad}}]},
      {Gray, Dashed, Point@circumT, Circle[circumT, circumTrad]},
      {Darker@Green, Point@cr, Point@ct}
      (*,{Darker@Green,Dashed,
       MapThread[Circle[#1,#2]&,{{cr,ct},{oscRrad/2,oscTrad/2}}]}*)}];
   c2 = If[a > b, a^2 - b^2, b^2 - a^2];
   lpRight = ListLinePlot[
     i = 0;
     Transpose@Append[Table[bconf = Sqrt[aconf^2 - c2];
        tangs = ellTangentsAB[aconf, bconf, {a, 0}];
        tangs,
        {aconf, Sqrt[c2], a, .001}], {{a, 0}, {a, 0}}], PlotStyle → Thick];
   lpTop = ListLinePlot[
     i = 0;
     Transpose@Table[bconf = Sqrt[aconf^2 - c2];
       tangs = ellTangentsAB[aconf, bconf, {0.001, 1.}];
       tangs,
       {aconf, Sqrt[c2], a, .01}], PlotStyle → Thick];
   Show[{lpRight, lpTop, plotEllb[a, b], gr}, PlotRange → All,
    AxesOrigin → {0, 0}, Frame → True, AspectRatio → Automatic]]
```

Out[325]=



In[326]:=

In[327]:=
```
ellTangentsAB[1.2, Sqrt[1.2^2 - (1.5^2 - 1)], {0, 1}]
```

Out[327]=
```
{{0., 0.19}, {0., 0.19}}
```

In[328]:=
```
Clear@drawCausticTangs;
Options[drawCausticTangs] =
  {drCircleT → False, drCircles → False, nmax → 7, drEll → False};
drawCausticTangs[a_, tDeg_, OptionsPattern[]] := Module[{c, tRad, fs, x1,
    x1Fixed = True, ns, caustics, tangs, inters, p1, gr, normal, lgt = .2,
    cirLeft, cirLeftRad, cirTop, cirTopRad, cirT, cirTRad, ellSol},
   c = If[a < 1, Sqrt[1 - a^2], Sqrt[a^2 - 1]];
   fs = getFoci[a];
   ns = Range[3, OptionValue@nmax];
   caustics = ("causticAB" /. getCausticBoth[a, #]) & /@ns;
   tRad = toRad[tDeg];
   p1 = {a Cos[tRad], Sin[tRad]};
   normal = norm@ellGrad[a, Sequence @@ p1];
   tangs = Flatten[ellTangentsAB[Sequence @@ #, p1] & /@ caustics, 1];
   If[OptionValue@drCircleT,
    cirT = getCircumcenter[p1, Sequence @@ Part[tangs, {-1, -3}]];
    cirTRad = magn[cirT - p1]];
   If[OptionValue@drCircles, Module[{tangsLeft, tangsTop},
     tangsLeft = ellTangentsAB[Sequence @@ Last[caustics], {a, 0}];
     cirLeft = getCircumcenter[{a, 0}, Sequence @@ tangsLeft];
```

```
    cirLeftRad = magn[cirLeft - {a, 0}];
    tangsTop = ellTangentsAB[Sequence @@ Last[caustics], {10^-6, 1}];
    cirTop = getCircumcenter[{0, 1}, Sequence @@ tangsTop];
    cirTopRad = magn[cirTop - {0, 1}]]];
  inters = Chop[Re[#]] & /@ ellInterRayUnprot[a, p1, # - p1][[2]] & /@ tangs;
  ellSol = If[OptionValue@drEll,
    solveEll5t[{p1, Sequence @@ Part[tangs, {-1, -3, -5, -7, -9}]}], {0, 0}];
  gr = Graphics[{PointSize@Medium,
      {Blue, Line[{p1, #}] & /@ Drop[inters, 2], Point /@ Drop[tangs, 2]},
      {Darker@Red, Line[{p1, #}] & /@ Take[inters, 2], Point /@ Take[tangs, 2]},
      {Black, MapThread[Text[Style["N=" <> ToString@#, Medium, Background → White],
          x1 = If[x1Fixed, a / 4, (#1 - 2) / (OptionValue@nmax - 2)];
          {x1, ellYb[Sequence @@ #2, x1]}] &, {ns, caustics}]},
      If[OptionValue@drCircleT, {Darker@Red, PointSize@Large, Point@cirT,
        Opacity@.1, EdgeForm[Darker@Black], Disk[cirT, cirTRad]}, {}],
      If[OptionValue@drCircles, MapThread[{PointSize@Large, Dashed,
          Point@#1, Opacity@.1, EdgeForm[Darker@Black], Disk[#1, #2]} &,
        {{cirLeft, cirTop}, {cirLeftRad, cirTopRad}}], {}],
      If[OptionValue@drEll, {Darker@Red, Rotate[Circle[{x0, y0} /. ellSol[[2]],
          Sqrt /@ ({a2, b2} /. ellSol[[2]])], -t /. ellSol[[2]]]}, {}],
      {Black, PointSize@Large, Point@fs, Point@p1, Arrowheads@Medium,
       Arrow[{p1, p1 + lgt * normal}]}}];
  Show[{plotEll[a],
    drawCausticPencilAB[caustics],
    (*drawCausticPencil[a,4,OptionValue@nmax],*)
    plotEllb[Sequence @@ First[caustics], Darker@Red],
    gr},
   PlotRange → All, ImageSize → Large, Frame → True, FrameStyle → Medium,
   PlotLabel → Style["a=" <> nfn[a, 2] <> ", θ=" <> nfn[tDeg, 1], {Black, Large}]]]
```

```
In[330]:= Manipulate[Show[drawCausticTangs[1.5, tDeg, drCircleT → drCircleT0,
    drCircles → drCircles0, drEll → drEll0, nmax → 9], ImageSize → 800],
  {{tDeg, 0}, -360, 360, 1, Appearance → "Labeled"},
  {{drCircleT0, False, "circleT"}, {True, False}},
  {{drCircles0, False, "circles"}, {True, False}},
  {{drEll0, False, "ellipse"}, {True, False}},
  SaveDefinitions → True]
```

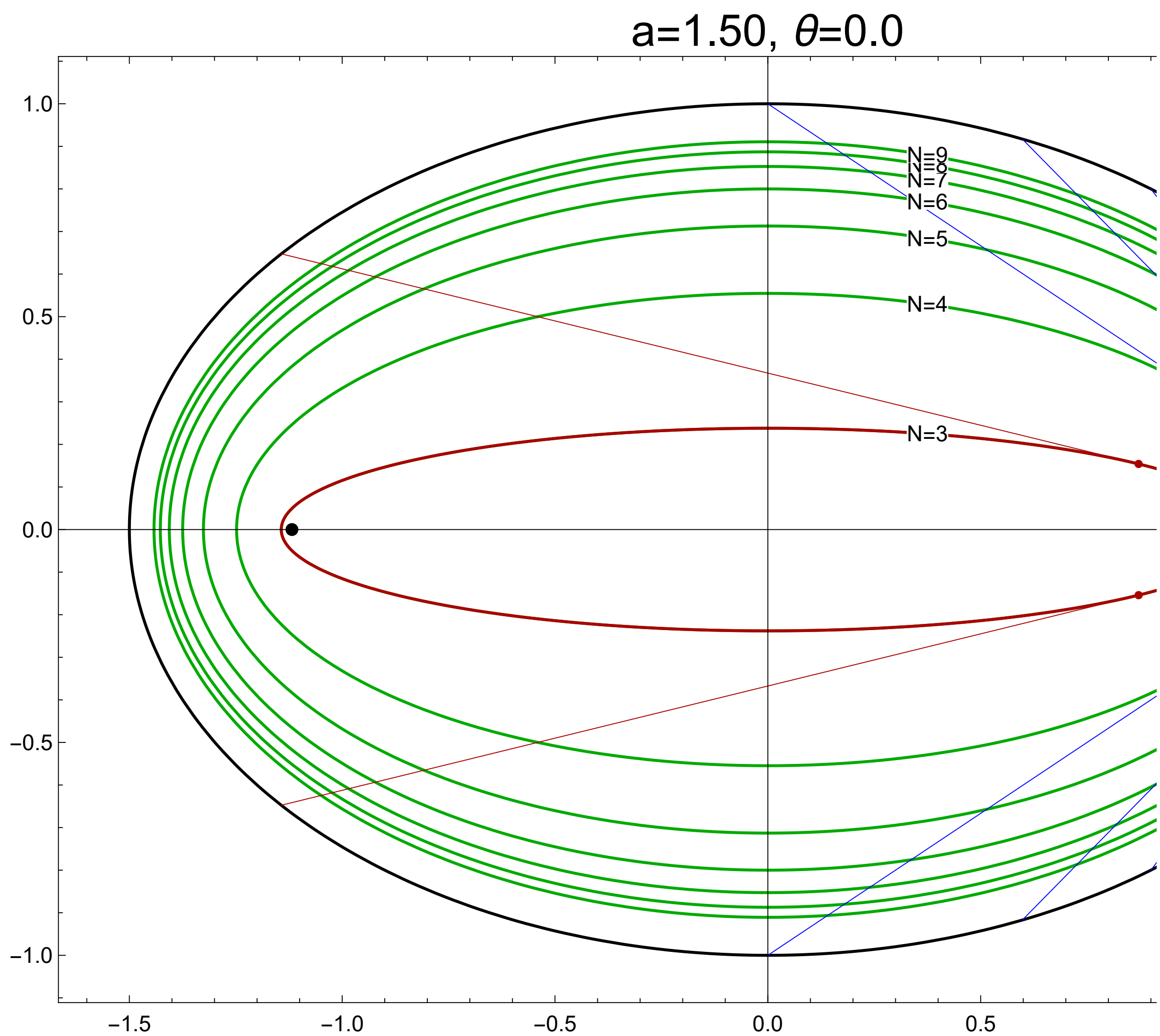
tDeg 0

circleT

circles

ellipse

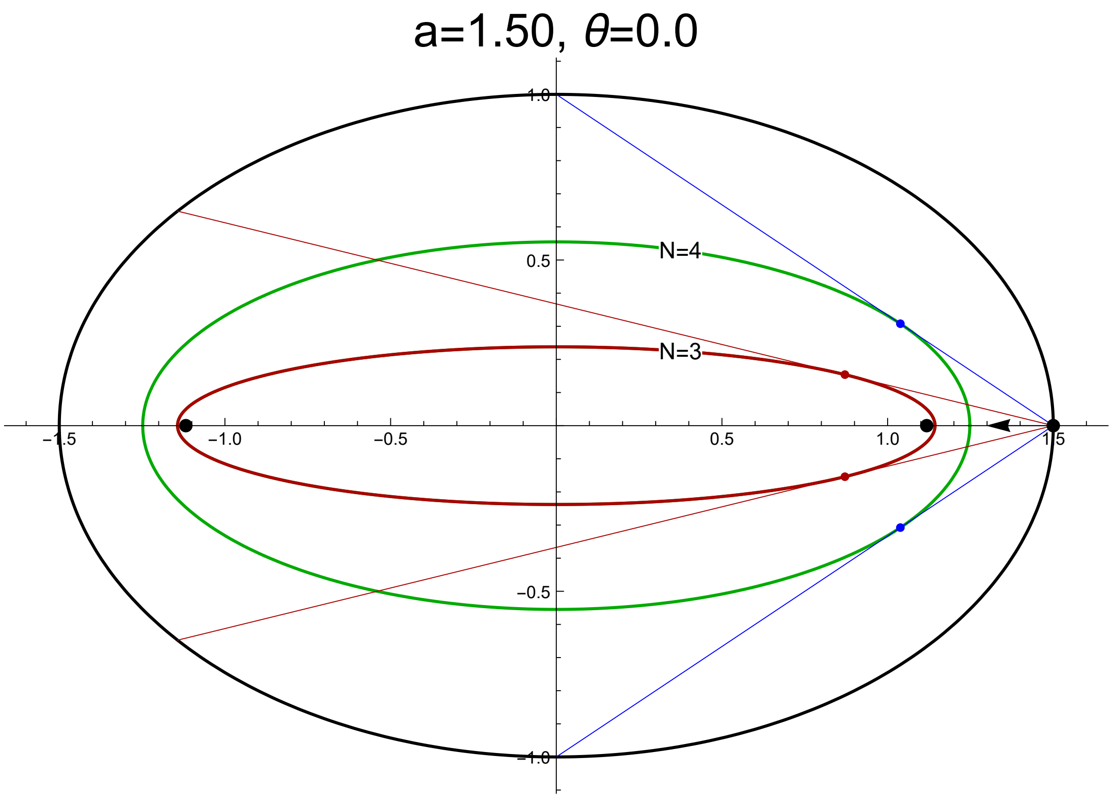
Out[330]=



Ideia : será que uma orbita tangente a alguma caustica da familia acima é orbita de outra caustica?

In[331]:= `Show[drawCausticTangs[1.5, 0, nmax → 4], Frame → False, Axes → True]`

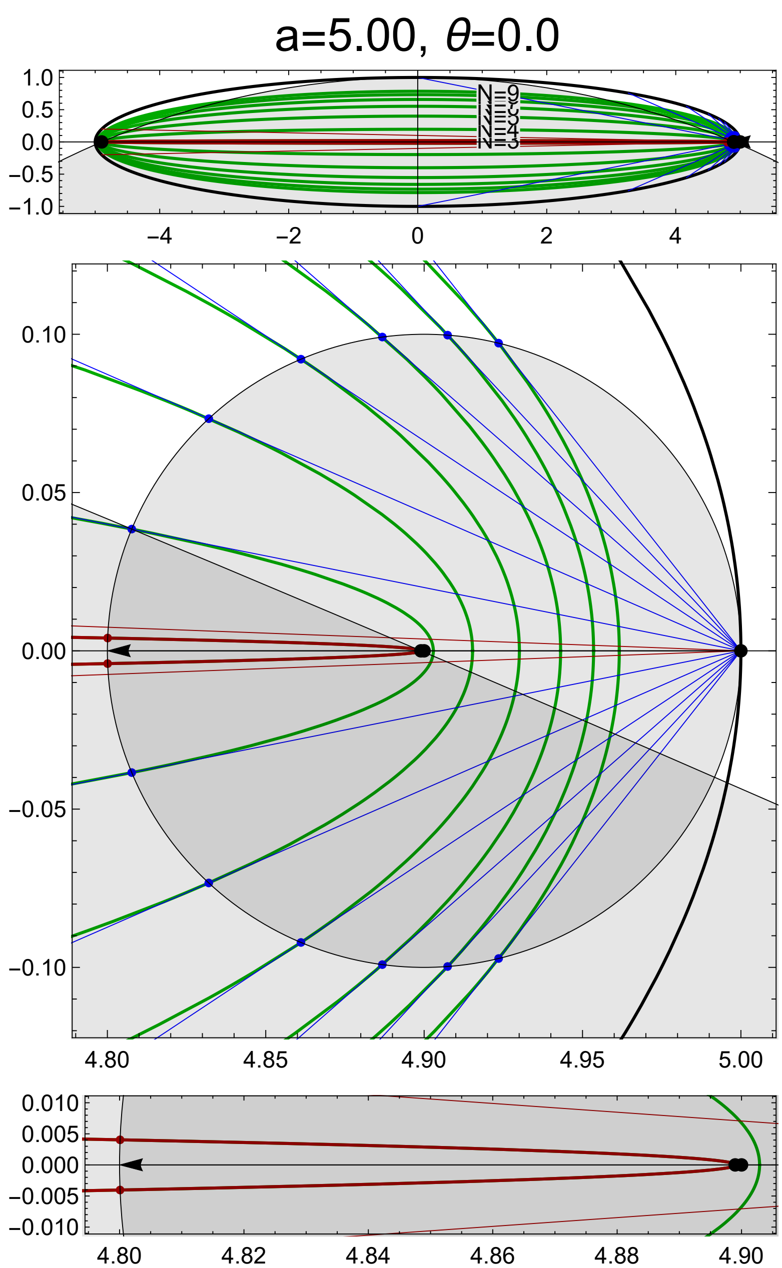
Out[331]=



Take a/b to a bigger value : center of tangents-circle always stays between N=3 and N=4

In[332]:=
```
Column@{Show[drawCausticTangs[5, 0, drCircles → True, nmax → 9],
    ImageSize → 400, PlotRange → {{-5, 5}, {-1, 1}}],
  Show[drawCausticTangs[5, 0, drCircles → True, nmax → 9], ImageSize → 400,
    PlotLabel → None, PlotRange → {{4.8, 5}, {-.11, .11}}],
  Show[drawCausticTangs[5, 0, drCircles → True, nmax → 9], ImageSize → 400,
    PlotLabel → None, PlotRange → {{4.8, 4.9}, {-.01, .01}}]}
```

Out[332]=



Hyperbola: Locus of Tangents to Confocals

```
In[333]:= hypC[a_, b_] := a^2 + b^2;
hypEqn[a_, b_, {x_, y_}] := (x / a) ^2 - (y / b) ^2 == 1
hypGrad[a_, b_, {x_, y_}] := -{x b^2, -y a^2};
```

```
In[335]:= hypParam[a_, b_, t_] := Module[{x, y},
    x = a Cosh[t];
    y = b Sinh[t];
    {{x, y}, {-x, -y}}];
```

```
In[336]:= FullSimplify[{x, y} /.
   Solve[{hypEqn[a, b, {x, y}], hypGrad[a, b, {x, y}].{px - x, py - y} == 0}, {x, y}],
  a > 0 && px ∈ Reals && py ∈ Reals]
```

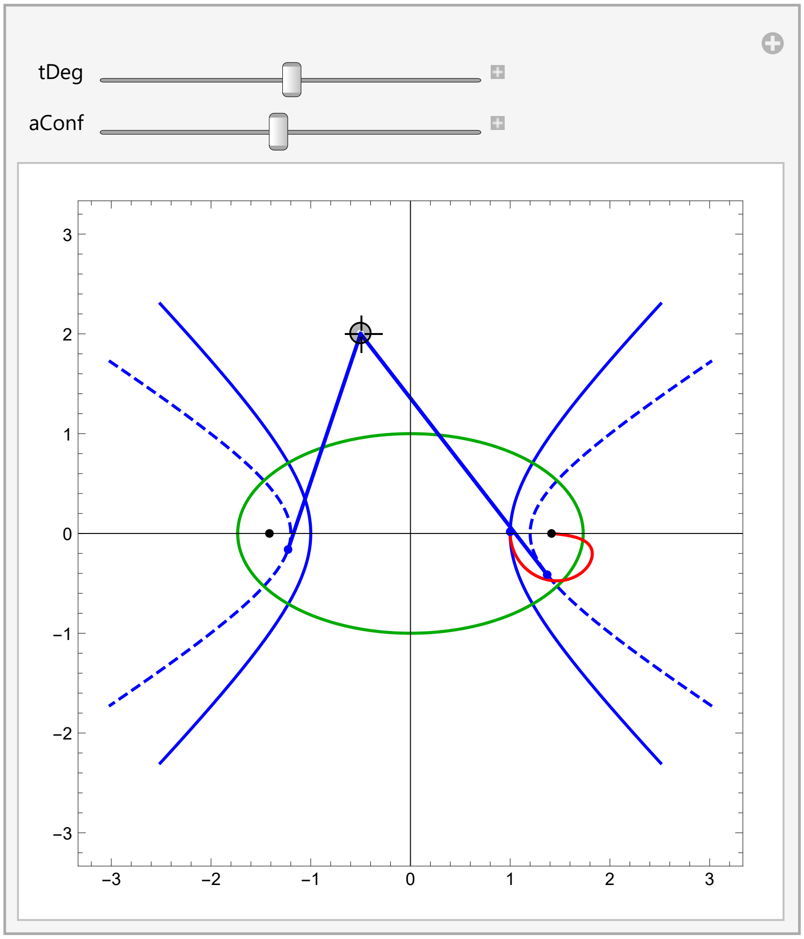
Out[336]=
$$\left\{\left\{-\frac{a^2\left(b^2\,px+\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{-b^2\,px^2+a^2\,py^2},\frac{b^2\left(a^2\,py^2+px\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{b^2\,px^2\,py-a^2\,py^3}\right\},\left\{\frac{a^2\left(b^2\,px-\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{b^2\,px^2-a^2\,py^2},\frac{b^2\left(a^2\,py^2-px\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{b^2\,px^2\,py-a^2\,py^3}\right\}\right\}$$

```
In[337]:= Clear@hypTangentsAB;
hypTangentsAB = Compile[{{a, _Real}, {b, _Real}, {p, _Real, 1}},
   Module[{a2, b2, px, py, px2, px3, py2, py3, radicand, numFact, denomx, denomy},
    {px, py} = p;
    a2 = a * a;
    b2 = b * b;
    px2 = px * px; py2 = py * py;
    px3 = px * px2; py3 = py * py2;
    denomx = b2 px2  - a2 py2;
    denomy = b2 px2 py - a2 py3;
    radicand =  -b2 px2 + a2 (py2 + b2);
    numFact = Sqrt[radicand] * Abs[py];
    Reverse@{
      {a2 safeDiv[b2 px + numFact , denomx], b2 safeDiv[a2 py2 + px numFact, denomy]},
      {a2 safeDiv[b2 px - numFact, denomx], b2 safeDiv[a2 py2 - px numFact, denomy]}}
   ]];
```

In[339]:=
```
drawTangs[p_, tangs_, clr_] := {PointSize@Medium, clr,
   Point@tangs, Thick, Line[{p, tangs[[1]]}], Line[{p, tangs[[2]]}]};
```

In[340]:=
```
Module[{a = 1, b = 1, ellB = 1, maxX = 3, bConf, ppHyp, ppEll,
   ppHypConf, hypLocus, ellA, ellTangs, hypTangs, pt, gr, c2, c, fs},
  c2 = a^2 + b^2;
  c = Sqrt@c2;
  fs = {{c, 0}, {-c, 0}};
  ellA = Sqrt[c2 + ellB^2];
  ppHyp = ParametricPlot[hypParam[a, b, t], {t, -π/2, π/2}, PlotStyle → Blue];
  ppEll = ParametricPlot[ellPb[ellA, ellB, t], {t, -π, π}, PlotStyle → Darker@Green];
  Manipulate[
   bConf = Sqrt[c2 - aConf^2];
   ppHypConf = ParametricPlot[hypParam[aConf, bConf, t],
     {t, -π/2, π/2}, PlotStyle → {Dashed, Blue}];
   pt = hypParam[a, b, toRad[tDeg]][[1]];
   (*ellTangs=ellTangentsAB[ellA,ellB,ellLoc];*)
   hypTangs = hypTangentsAB[aConf, bConf, hypLoc];
   hypLocus =
    Quiet@ParametricPlot[hypTangentsAB[aLocus, Sqrt[c2 - aLocus^2], pt][[1]],
      {aLocus, 1, c}, PlotStyle → Red];
   gr = Graphics[{PointSize@Medium,
      {Blue, Point@pt},
      {Black, Point@fs}, (*,
      drawTangs[ellLoc,ellTangs,Darker@Green],*)
      drawTangs[hypLoc, hypTangs, Blue]}];
   Show[{ppHyp, ppHypConf, ppEll, hypLocus, gr}, ImageSize → Medium,
    Frame → True, PlotRange → {{-maxX, maxX}, {-maxX, maxX}}],
   {{tDeg, 1}, -90, 90, .01},
   {{aConf, 1.2}, .5, 2, .01}, (*,
   {{ellLoc,{0,2}},Locator},*)
   {{hypLoc, {-.5, 2}}, Locator}]]
```

Out[340]=



Sum Of Cosines: Caustic & Relaxation

In[1885]:=

```
(* slow, since needs to recompute caustic everytime *)
(* use getCausticOrbits for efficiency *)
Clear@getCausticOrbit;
getCausticOrbit[a_, n_, t_] :=
  Module[{causticData, causticAB, poly, norms, logErr},
   causticData = getCausticBoth[a, n];
   logErr = Log10["error" /. causticData];
   If[logErr > -6,
    Print["a=", nfn[a, 2], ", n=", n, ": log10(error)=", nfn[logErr, 2]]];
   causticAB = "causticAB" /. causticData;
   Most@getPolyCaustic[a, n, t, causticAB][[1]]];
```

In[1887]:=

```
(* slow, use getPolyCosinesCaustic for efficiency *)
Clear@getCausticOrbitCosines;
getCausticOrbitCosines[a_, n_, t_] := getPolyCosines@getCausticOrbit[a, n, t];
```

```
In[341]:= Clear@getCausticOrbits;
getCausticOrbits[a_, n_, tDegs_] :=
  Module[{causticData, causticAB, poly, norms, logErr},
   causticData = getCausticBoth[a, n];
   logErr = Log10["error" /. causticData];
   If[logErr > -6,
    Print["a=", nfn[a, 2], ", n=", n, ": log10(error)=", nfn[logErr, 2]]];
   causticAB = "causticAB" /. causticData;
   Most[getPolyCaustic[a, n, toRad@#, causticAB][[1]]] & /@ tDegs
  ];
```

```
In[1473]:= Clear@getPolyCosinesCaustic;
getPolyCosinesCaustic[a_, n_, step_: 1] :=
  getPolyCosines /@ getCausticOrbits[a, n, Range[0, 359, step]];
```

```
In[1601]:= insertNum[val_, list_] := {val, #} & /@ list;
```

```
In[1602]:= insertNum[1, {a, b, c}]
Out[1602]= {{1, a}, {1, b}, {1, c}}
```

```
In[1603]:= MapThread[insertNum[#1, #2] &,
  {{1, 2, 3}, Transpose[Accumulate /@ {{1, 1, 2}, {10, 10, 11}, {200, 11, 201}}]}]
Out[1603]= {{{1, 1}, {1, 10}, {1, 200}}, {{2, 2}, {2, 20}, {2, 211}}, {{3, 4}, {3, 31}, {3, 412}}}
```

```
In[1981]:= Clear@cosineSumStackedPlotLowLevel;
Options[cosineSumStackedPlotLowLevel] = {note -> "", ps → Thick};
cosineSumStackedPlotLowLevel[a_, n_, coss_, tDegs_, OptionsPattern[]] :=
 Module[{cossAcc, cossAccAngs, legs, filling},
  cossAcc = Transpose[Accumulate /@ coss];
  cossAccAngs =
   Transpose@MapThread[insertNum[#1, #2] &, {tDegs, Transpose@cossAcc}];
  legs = Table[Subscript["∑cosθ", If[i > 1, "1→", ""] <> ToString@i], {i, n}];
  filling = {1 → Axis, Sequence @@ Table[(i + 1 → {i}), {i, n - 1}]};
  ListLinePlot[cossAccAngs,
   Filling → filling, (*{1→Axis,2→{1},3→{2},4→{3},5→{4}},*)
   PlotLegends → legs, PlotTheme -> "Scientific",
   PlotStyle → OptionValue@ps, ImageSize → Large, Frame → True,
   FrameStyle → Large,
   PlotRange → {{Min@tDegs, Max@tDegs}, Automatic},
   PlotLabel → Style["∑cos(θi) stacked: a=" <> nfn[a, 2] <>
      ", N=" <> ToString@n <> OptionValue@note, {Black, Large}]]]
```

In[1984]:=
```
Clear@cosineSumStackedPlot;
cosineSumStackedPlot[a_, n_, tDegs_, opts : OptionsPattern[]] :=
 Module[{coss, cossAcc, legs, filling},
  coss = getPolyCosinesCaustic[a, n, 1];
  cosineSumStackedPlotLowLevel[a, n, coss, tDegs, opts]
 ];

(* &&& *)
```

Cusps: cosines in caustics is generating cusps!

Do both orbit and caustics generate the same curves?

In[1985]:=
```
getOrbitCosines[1.5, 0]
```

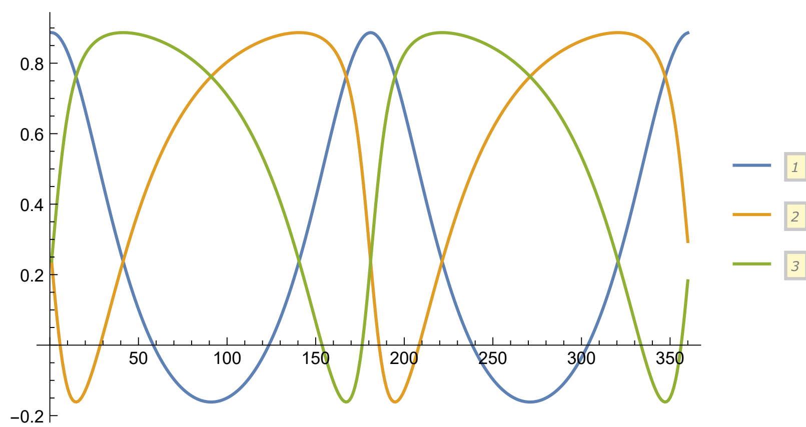
Out[1985]= {0.88676, 0.23795, 0.23795}

In[1986]:=
```
getCausticOrbitCosines[1.5, 3, 0]
```

Out[1986]= {0.88676, 0.23795, 0.23795}

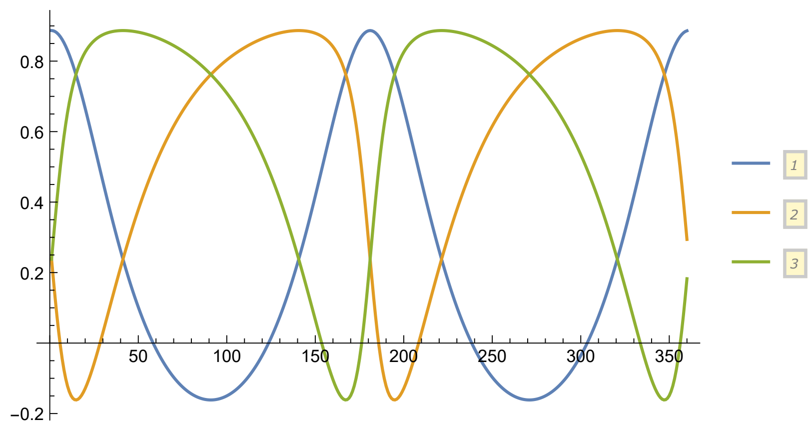
In[1987]:=
```
ListLinePlot[Transpose[getOrbitCosines[1.5, toRad@#] & /@ Range[0, 359, 1]],
 PlotLegends → Automatic]
```

Out[1987]=



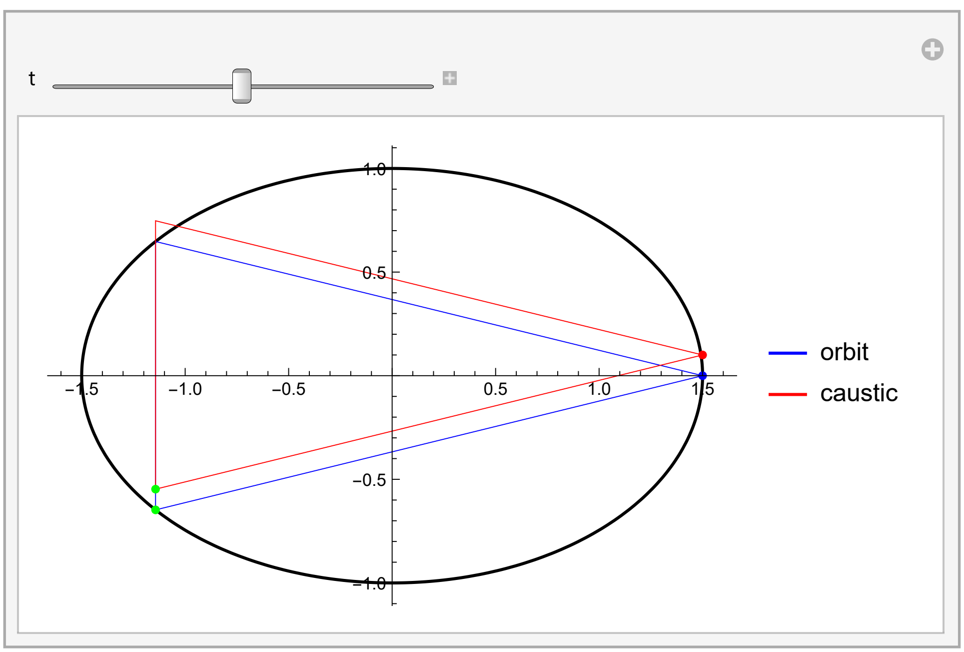
In[1988]:=
```
ListLinePlot[
 Transpose[getCausticOrbitCosines[1.5, 3, toRad@#] & /@ Range[0, 359, 1]],
 PlotLegends → Automatic]
```

Out[1988]=



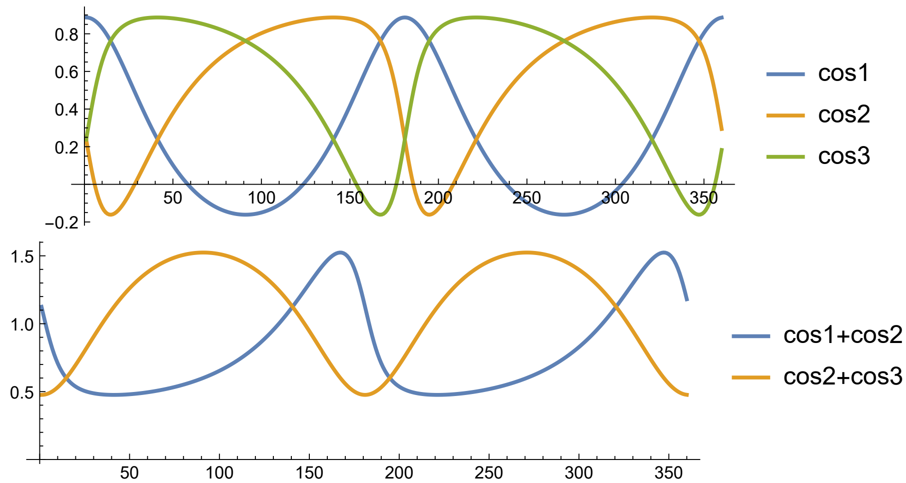
In[1989]:=
```
Module[{a = 1.5, n = 3, triOrb, triCau, ell, gr1, gr2},
 ell = plotEll[a];
 Manipulate[
  triOrb = First@orbitNormals[a, toRad@t];
  triCau = {#[[1]], #[[2]] + .1} & /@ getCausticOrbit[a, n, toRad@t];
  gr1 = Graphics[{FaceForm@None, EdgeForm@Blue, Polygon@triOrb, Blue,
     PointSize@Medium, Point[First@triOrb], Green, Point[Second@triOrb]}];
  gr2 = Graphics[{FaceForm@None, EdgeForm@Red, Polygon@triCau, Red,
     PointSize@Medium, Point[First@triCau], Green, Point[Second@triCau]}];
  Legended[Show[{ell, gr1, gr2}], LineLegend[{Blue, Red}, {"orbit", "caustic"}]],
  {{t, 0}, -180, 180}]]
```

Out[1989]=



In[1990]:=
```
Module[{cossT, l123, l1p2, ar = .33},
 cossT = Transpose@getPolyCosinesCaustic[1.5, 3, 1];
 l123 = ListLinePlot[cossT, AspectRatio → ar, PlotStyle → Thick,
   ImageSize → Medium, PlotLegends → {"cos1", "cos2", "cos3"}];
 l1p2 = ListLinePlot[{
    Legended[cossT[[1]] + cossT[[2]], "cos1+cos2"],
    Legended[cossT[[2]] + cossT[[3]], "cos2+cos3"]
    (*Legended[cossT[[1]]+cossT[[3]],"cos1+cos3"]*)},
   AspectRatio → ar, ImageSize → Medium, PlotStyle → Thick];
 Grid[Transpose@{{l123, l1p2}}]]
```

Out[1990]=

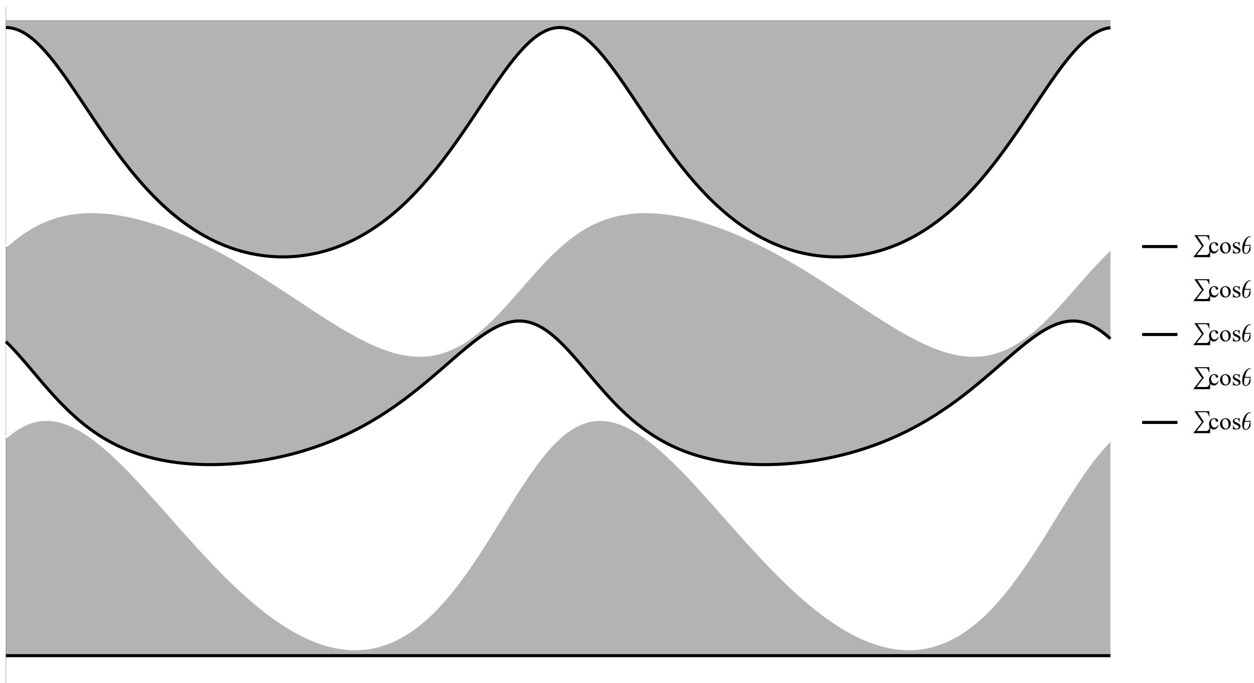


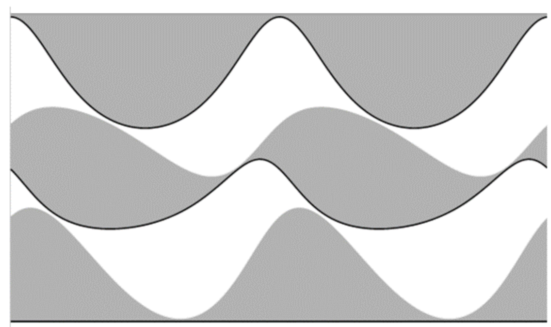
In[1991]:=

Copacabana Princesinha do Mar

In[2002]:=
```
Show[cosineSumStackedPlot[1.5, 5, Range[0, 359, 1],
  ps → ({Opacity@1, EdgeForm@None, #} & /@ {Black, White, Black, White, Black})],
 PlotLabel -> "", Axes → None, Frame → None]
```

Out[2002]=





In[1476]:=
```
Clear@getPolyCosineStats;
getPolyCosineStats[a_, n_] := getStats[Total /@ getPolyCosinesCaustic[a, n]];
```

In[345]:=
```
Save["cosSumCaustic.m", cosSumCaustic];
```

In[1471]:=
```
(* slow computation *)
cosSumCaustic = If[False, Flatten[
    Table[{a, n, getPolyCosineStats[a, n]}, {a, 1.1, 3.0, .1}, {n, 3, 21, 1}], 1];
   Save["cosSumCaustic.m", cosSumCaustic],
   Get["cosSumCaustic.m"]];
```

Show (a, n) with highest z-scores, ignore N=4

In[347]:=
```
Select[Sort[Select[cosSumCaustic, #[[2]] ≠ 4 &], #1[[3, 3]] > #2[[3, 3]] &],
  Abs[#[[3, 3]]] > 10^-9 &] // Grid
```

Out[347]=

2.5	5	$\{-1.40384, 1.67906\times10^{-9}, -1.19604\times10^{-9}, -1.40384, -1.40384, -1.40384\}$
2.4	12	$\{-10.2574, 1.38909\times10^{-8}, -1.35423\times10^{-9}, -10.2574, -10.2574, -10.2574\}$
2.7	10	$\{-7.8964, 1.16016\times10^{-8}, -1.46923\times10^{-9}, -7.8964, -7.8964, -7.8964\}$
2.8	7	$\{-4.12911, 6.13308\times10^{-9}, -1.48533\times10^{-9}, -4.12911, -4.12911, -4.12911\}$
2.5	6	$\{-2.81633, 4.90386\times10^{-9}, -1.74123\times10^{-9}, -2.81633, -2.81633, -2.81633\}$
2.2	8	$\{-5.51568, 1.07162\times10^{-8}, -1.94285\times10^{-9}, -5.51568, -5.51568, -5.51568\}$

Check Cosine Sum for Hexagons vs Model

In[1455]:=
```
hexagonHorizCosineSum[a_] := Module[{ca, c2a, cb},
  ca = a / (1 + a);
  c2a = 2 ca^2 - 1;
  cb = - ca;
  2 c2a + 4 cb]
```

In[1456]:=
```
hexagonHorizCosineSum[1]
```

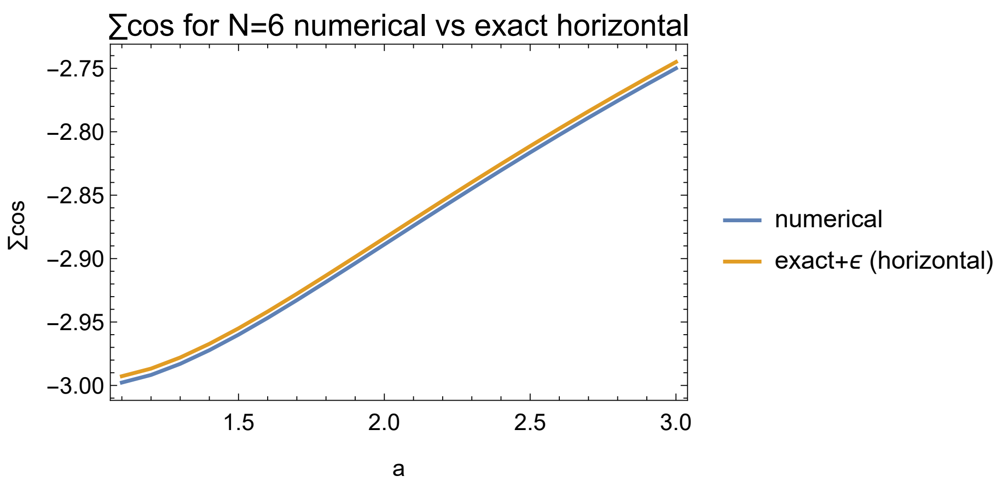
Out[1456]= -3

In[1457]:=
```
6 * Cos[(2 / 3 π)]
```

Out[1457]= -3

In[1458]:=
```
Module[{cosHex, cosHexHoriz},
 cosHex = {#[[1]], #[[3, 1]]} & /@ Select[cosSumCaustic, #[[2]] == 6 &];
 cosHexHoriz = {#, .005 + hexagonHorizCosineSum@#} & /@ (First /@ cosHex);
 ListLinePlot[
  {Legended[cosHex, "numerical"], Legended[cosHexHoriz, "exact+ϵ (horizontal)"]},
  PlotLabel → Style["∑cos for N=6 numerical vs exact horizontal", {Black, 16}],
  Frame → True, FrameStyle → Medium,
  FrameLabel → (Style[#, Medium] & /@ {"a", "∑cos"}), PlotStyle → Thick]]
```

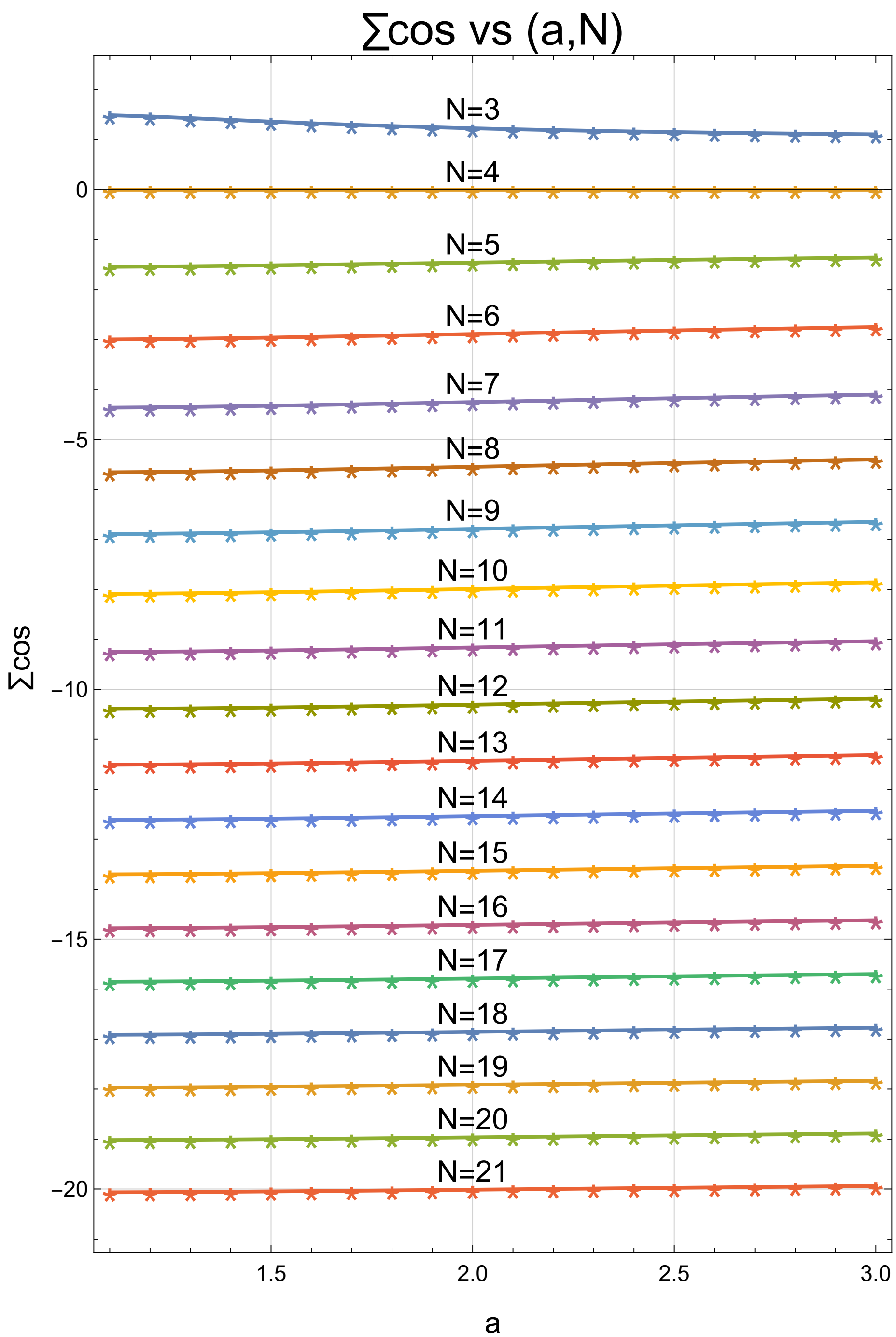
Out[1458]=



Plot Cosine Sum vs a and n

In[348]:=
```
constCosPlot = Module[{uniqueNs, groups},
  groups = GroupBy[cosSumCaustic, #[[2]] &];
  uniqueNs = Keys[groups];
  ListPlot[
   Table[{#[[1]], #[[3, 1]]} & /@ groups[uniqueNs[[i]]], {i, Length@uniqueNs}],
   PlotStyle → Thick, Joined → True, PlotMarkers → Style["*", Large],
   GridLines → Automatic, Epilog → {Text[Style["N=" <> ToString@#, 16],
        {2.0, Mean[#[[3, 1]] & /@ groups[#]]}, {0, -1}] & /@ uniqueNs},
   Frame → True, ImageSize → Large, FrameStyle → Directive[Black, Medium],
   FrameLabel → (Style[#, 16] & /@ {"a", "∑cos"}), AspectRatio → 1.5,
   PlotLabel → Style["∑cos vs (a,N)", {Black, Large}]]]
```

Out[348]=



In[349]:= **Length@cosSumCaustic**

Out[349]= 380

Get Stats of ∑cos Z-Score without N = 4

In[350]:=
```
getStats[#[[3, 3]] & /@ Select[cosSumCaustic, #[[2]] ≠ 4 &]]
```

Out[350]= $\{-5.53386\times10^{-11}, 2.2158\times10^{-10}, -4.00409, -1.94285\times10^{-9}, 8.40426\times10^{-11}, -3.51529\times10^{-14}, 360\}$

Get Stats of ∑cos standard dev without N = 4

In[351]:=
```
getStats[#[[3, 2]] & /@ cosSumCaustic]
```

Out[351]= $\{3.7288\times10^{-10}, 1.48712\times10^{-9}, 3.98819, 4.29674\times10^{-17}, 1.38909\times10^{-8}, 2.4233\times10^{-13}, 380\}$

Draw Poly

In[352]:=
```
Clear@polyExcentral;
polyExcentral[alphaT_, i_, fnVtx0_] := Module[{a, poly, polyTangs, polyExc},
   a = "a" /. alphaT;
   poly = polyVtx[alphaT, i, fnVtx0];
   polyTangs = perp[ellGrad[a, #[[1]], #[[2]]]] & /@ poly;
   polyExc = MapThread[interRays[#1, #2, #3, #4] &,
     {poly, polyTangs, RotateLeft@poly, RotateLeft@polyTangs}];
   {poly, polyExc}];
```

In[354]:=
```
Clear@drawPoly;
Options[drawPoly] =
  {drNotables → False, drSubtri → False, drLoci → False, drMedians → False,
   drExcentral → False, drCentroids → False, drCentroidLabels → False,
   drError → False, drCircs → False, drIncs → False, vtx → {1, 2, 3}, plotAll → False};
drawPoly[polyErr_, notableLoci_, OptionsPattern[]] :=
  Module[{poly, a, centroids, centroidNames, lab, pnames, meds,
     polyTangs, polyInters, polyInterNames, tri1, notables, circs, incs,
     lgt = .25, fnt = 14},
   poly = "poly" /. polyErr;
   a = "a" /. polyErr;
   centroids = getCentroids[poly];
   centroidNames = {"Gvtx", "Gper", "Glam"};
   lab = "error:  " <> nfn[N@("error" /. polyErr), 3];
   pnames =
    Take[{"A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K"}, Length@poly];
   meds = getMediansV@poly;
   polyTangs = perp[ellGrad[a, #[[1]], #[[2]]]] & /@ poly;
   polyInters = MapThread[interRays[#1, #2, #3, #4] &,
```

```
    {poly, polyTangs, RotateLeft@poly, RotateLeft@polyTangs}];
  polyInterNames = If[OptionValue@drExcentral,
    MapThread[(#1 <> "'" <> #2 <> "'") &, {pnames, RotateLeft@pnames}], {}];
  (* *** *)
  If[OptionValue@drCircs, Module[{sides, tri, cir},
    circs = Table[
       tri = Take[RotateLeft[poly, i - 1], 3];
       cir = getCircumcenterTrilin[tri, RotateLeft@triLengths@tri];
       (*cir=getCircumcenter@@tri;*)
       {cir, magn[cir - tri[[1]]]}, (* returns circumcenter and circumradius *)
       {i, Length@poly}];
    (*circs=Take[circs,1];*)
   ]];
  If[OptionValue@drIncs, Module[{sides, tri, inc},
    incs = Table[
       tri = Take[RotateLeft[poly, i - 1], 3];
       inc = getIncenterTrilin[tri, RotateLeft@triLengths@tri];
       (*cir=getCircumcenter@@tri;*)
       {inc, closestDist[inc, tri[[1]], tri[[2]]]},
       (* returns circumcenter and circumradius *)
       {i, Length@poly}];
    (*circs=Take[circs,1];*)
   ]];
  If[OptionValue@drNotables || OptionValue@drSubtri, Module[{normals},
    tri1 = Part[poly, OptionValue@vtx];
    normals = getTriBisectors @@ tri1;
    notables = getNotables[tri1, normals];
   ]];
  gr = Graphics[{PointSize@Large, Point@poly, FaceForm@None,
     {EdgeForm[{Black, Thick}],
      FaceForm@Gray, Opacity@.1, Polygon@poly, Black, Point@poly},
     {Black, Arrowheads[Medium], MapThread[
       drawArrow[#1, norm[perpNeg[#2]], lgt] &, {poly, polyTangs}]},
     If[OptionValue@drCentroids, {Black, MapThread[
        {Point@#1, If[OptionValue@drCentroidLabels, Text[Style[#2, 14],
            #1, {0, -1.5}], {}]} &, {centroids, centroidNames}]}, {}],
     If[OptionValue@drExcentral, {EdgeForm@Darker@Green,
       Polygon@polyInters, Darker@Green, Point@polyInters,
       MapThread[Text[Style[#1, fnt], #2, {-1.25, -1.25}] &,
        {polyInterNames, polyInters}]}, {}],
     {Black, MapThread[Text[Style[#1, fnt], ray[#2, norm[perp[#3]], lgt/2]] &,
       {pnames, poly, polyTangs}]},
     If[OptionValue@drMedians, {{Dashed, Blue, Line[{#, {0, 0}}] & /@polyInters},
```

```
          {Blue, Point@meds, Point@{0, 0}}}, {}],
        If[OptionValue@drCircs, {Red, {Dashed, Circle[#[[1]], #[[2]]] & /@ circs},
          Point[First /@ circs], Circle[circs[[1, 1]], .05]}, {}
        ],
        If[OptionValue@drIncs, {Darker@Green, Dashed,
          Circle[#[[1]], #[[2]]] & /@ incs, Point[First /@ incs]}, {}
        ],
        If[OptionValue@drNotables,
         {List @@ drawSomeNotables[notables, First@notableLoci],
          EdgeForm@None, FaceForm@Red, Opacity@.1, Polygon@tri1}, {}],
        If[OptionValue@drSubtri, {EdgeForm@None, FaceForm@Red,
          Opacity@.2, Polygon@tri1}, {}]
       }];
     Show[{plotEll[a], gr, If[OptionValue@drLoci, Second@notableLoci, {}]},
      If[OptionValue@drError, PlotLabel → lab, {}],
      PlotRange → (If[OptionValue@plotAll, All, {{-2, 2}, {-1, 1}}]),
      Frame → True, FrameStyle → Medium,
      AxesStyle → Directive[{Dotted, Gray}]]];
```

```
In[357]:= Clear@showOnePoly;
showOnePoly[alphaT_, i_, notableLoci_, fnVtx0_,
   fnErrorP_, opts : OptionsPattern[]] := Module[{polyErr},
   polyErr = polyError[alphaT, i, fnVtx0, fnErrorP];
   drawPoly[polyErr, notableLoci, FilterRules[{opts}, Options[drawPoly]]]];
```

```
In[359]:= Clear@getPolyNotableLoci;
getPolyNotableLoci[alphaT_, fnVtx0_, vtx_: {1, 2, 3},
   nots_: {"inc", "bar", "ort"}] := Module[{polys, normals, tris, notables},
   (*{"cir","npc","mit","feu"};*)
   polys = Table[polyVtx[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}];
   polys = Append[polys, First@polys];
   tris = Part[#, vtx] & /@ polys;
   normals = getTriBisectors @@ # & /@ tris;
   notables = MapThread[getNotables[#1, #2] &, {tris, normals}];
   {nots, ListLinePlot[Transpose[nots /. notables], PlotStyle → (nots /. ptClrs)]}
  ];
```

```
In[361]:= makeVtxLabs[vtxList_] := StringJoin /@ Map[ToString, vtxList, {2}];
```

In[362]:=
```
Clear@manipulatePolyVtx;
manipulatePolyVtx[alphaT_, notableLociList_, fnVtx0_, fnErrorP_,
   vtxList_: {{1, 2, 3}}] := Module[{vtxLabs, vtxMatch, lociMatch},
   vtxLabs = makeVtxLabs@vtxList;
Manipulate[
    vtxMatch = First@FirstPosition[vtxLabs, tri];
    lociMatch = notableLociList[[vtxMatch]];
    Show[
     showOnePoly[alphaT, i,
      lociMatch, fnVtx0, fnErrorP,
      drNotables → drNotables0, drSubtri → drSubtri0, drLoci → drLoci0,
      drExcentral → drExcentral0, drCentroids → drCentroids0,
      drCentroidLabels → drCentroidLabels0, drCircs → drCircs0,
      drIncs → drIncs0, plotAll → plotAll0,
      vtx → vtxList[[vtxMatch]]], ImageSize → Large],
    {{i, 1}, 1, Length["alphas" /. alphaT], 1, Appearance → "Labeled"},
    {{drExcentral0, False, "excentral"}, {True, False}},
    {{drSubtri0, False, "subtri"}, {True, False}},
    {{drNotables0, True, "notables"}, {True, False}},
    {{drLoci0, True, "loci"}, {True, False}},
    {{drCentroids0, True, "centroids"}, {True, False}},
    {{drCentroidLabels0, False, "centroidLabels"}, {True, False}},
    {{drCircs0, False, "circs"}, {True, False}},
    {{drIncs0, False, "incs"}, {True, False}},
    {{tri, vtxLabs[[1]]}, vtxLabs},
    {{plotAll0, False, "plotAll"}, {True, False}}, SaveDefinitions → True]];
```

In[364]:=
```
Clear@showOneLocusVtx;
Options[showOneLocusVtx] =
  {filling → None, fillingStyle → {Opacity@.1}, plotAll → False};
showOneLocusVtx[a_, i_, lociList_, vtxList_, vtx_, OptionsPattern[]] :=
  Module[{vtxLabs},
   vtxLabs = makeVtxLabs@vtxList;
   Show[{plotEll[a],
     ListLinePlot[
      MapIndexed[If[vtx == "all" || vtx == #1, Legended[
          lociList[[First@#2, i]], vtxLabs[[First@#2]]], {{0, 0}}] &, vtxLabs],
      Filling → (OptionValue@filling), FillingStyle → (OptionValue@fillingStyle),
      PlotStyle → Thick
      (*PlotStyle→{Blue,Red,Darker@Green}*)]}, Frame → True, FrameStyle → Medium,
    PlotRange → (If[OptionValue@plotAll, All, {{-2, 2}, {1, 1}}]),
    PlotLabel → Style[centerNames[[i]] <> ", a/b=" <> nfn[a, 2], {Black, 16}],
    ImageSize → Large]];
```

In[367]:=
```
Clear@manipulateLociVtx;
manipulateLociVtx[alphaT_, lociList_,
   vtxList_: {{1, 2, 3}}, opts : OptionsPattern[]] := Module[{a, vtxLabs},
   a = "a" /. alphaT;
   vtxLabs = makeVtxLabs@vtxList;
   Manipulate[showOneLocusVtx[a, i, lociList, vtxList, vtx,
     FilterRules[{opts, plotAll → plotAll0}, Options[showOneLocusVtx]]],
    {{i, 1}, 1, Length[centerNames], 1, Appearance → "Labeled"},
    {{vtx, "all"}, {"all", Sequence @@ vtxLabs}},
    {{plotAll0, True, "plotAll"}, {True, False}}, SaveDefinitions → True]];
```

In[369]:=
```
makeOneLociRow[alphaT_, lociList_, vtxList_, i_,
   showAll_: True, opts : OptionsPattern[]] := Module[{a, vtxLabs},
   vtxLabs = makeVtxLabs@vtxList;
   a = "a" /. alphaT;
   Grid[{showOneLocusVtx[a, i, lociList, vtxList,
        #, FilterRules[{opts}, Options[showOneLocusVtx]]] & /@
      If[showAll, Prepend[vtxLabs, "all"], vtxLabs]}]];
```

Poly Vert

In[370]:=
```
Clear@rotPoly;
rotPoly[a_, poly_] := Module[{tangs, bis, mtx, polyRot, bisRot},
   tangs = perp[ellGrad[a, #[[1]], #[[2]]]] & /@ poly;
   bis = norm[perpNeg[#]] & /@ tangs;
   mtx = {perp@bis[[1]], -bis[[1]]};
   polyRot = (mtx.# + {0, 1}) & /@ ((# - poly[[1]]) & /@ poly);
   bisRot = (mtx.#) & /@ bis;
   {polyRot, bis, bisRot}
  ];
```

```
In[372]:= Clear@drawPolyVert;
drawPolyVert[polyErr_, drBack_: True] := Module[{a, poly, centroid,
    lab, pnames, polyRot, polyBisectors, polyBisectorsRot, centroidRot,
    lgt = .33, fnt = 14},
   a = "a" /. polyErr;
   poly = "poly" /. polyErr;
   centroid = RegionCentroid@Polygon@poly;
   lab = "error: " <> nfn[N@("error" /. polyErr), 3];
   pnames =
    Take[{"A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K"}, Length@poly];
   {polyRot, polyBisectors, polyBisectorsRot} = rotPoly[a, poly];
   centroidRot = RegionCentroid@Polygon@polyRot;
   (*polyInters=MapThread[interRays[#1,#2,#3,#4]&,
     {poly,polyTangs,RotateLeft@poly,RotateLeft@polyTangs}];
   polyInterNames={"p12","p23","p34","p45","p51"};*)
   gr = Graphics[{PointSize@Large, FaceForm@None, Arrowheads[Medium],
      {EdgeForm[{Red, Thick}], FaceForm@Gray, Opacity@.3, Polygon@polyRot},
      {Black, Point@polyRot, Black, drawArrow[{0, 1}, {0, -1}, 1.5 * lgt], "bar" /.
        ptClrs, Point@centroidRot, Text[Style["G'", 14], centroidRot, {0, -1.5}]},
      MapThread[Text[Style[#1 <> "'", {Lighter@Red, fnt}], ray[#2, -#3, lgt / 2]] &,
       {pnames, polyRot, polyBisectorsRot}],
      If[drBack,
       {{EdgeForm[{Gray, Dashed}], Polygon@poly, Gray, Point@poly,
         Point@centroid, Text[Style["G", 14], centroid, {0, -1.5}]},
        {Lighter@Gray, MapThread[drawArrow[#1, #2, lgt] &,
          {poly, polyBisectors}]},
        MapThread[Text[Style[#1, {Lighter@Gray, fnt}], ray[#2, -#3, lgt / 2]] &,
         {pnames, poly, polyBisectors}]}, {}]
     }];
   Show[{plotEll[a, Lighter@Gray], gr}, AxesStyle → Directive[Lighter@Gray],
    Frame → True, PlotRange → {{-2, 2}, {-1.2, 1.2}}]];
```

```
In[374]:= Clear@getPolyVertLoci;
getPolyVertLoci[alphaT_, fnVtx0_] :=
  Module[{a, polys, centroids, polyRots, centroidsRot, toPlot},
   a = "a" /. alphaT;
   polys = Table[polyVtx[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}];
   centroids = RegionCentroid[Polygon[#]] & /@ polys;
   polyRots = First /@ (rotPoly[a, #] & /@polys);
   (*polyRots = rotpoly[a, #] & /@ polys;*)
   centroidsRot = RegionCentroid[Polygon[#]] & /@ polyRots;
   (*toPlot={Second /@ polyRots,
      Third /@ polyRots, Fourth /@ polyRots, Fifth /@ polyRots};*)
   toPlot = Rest@Transpose[polyRots];
   ListLinePlot[{Sequence @@ toPlot, centroids, centroidsRot}, Axes -> False]
  ];

In[376]:= Clear@showOnePolyVert;
showOnePolyVert[alphaT_, i_,
   polyVertLoci_, fnVtx0_, fnErrorP_, drBack_: True] :=
  Module[{a, p1, t, alpha, polyErr},
   polyErr = polyError[alphaT, i, fnVtx0, fnErrorP];
   Show[{drawPolyVert[polyErr, drBack], polyVertLoci},
    PlotRange -> {{-2, 2}, {-2, 2}},
    Frame -> True, FrameStyle -> Medium,
    AxesStyle -> Directive[{Dotted, Gray}]]];
```

Triangle (Simulate AlphaT)

```
In[378]:= Clear@getEllVtx0;(* not for error purposes *)
getEllVtx0[a_, p1_, alpha_] := {p1};
triErrorP[a_, p1_, alpha_] := 0;

In[381]:= getEllAlphaT[a_] := Module[{tsDeg, tsRad, alphas},
   tsDeg = Range[360] - 1;
   tsRad = toRad /@ tsDeg;
   alphas = ConstantArray[0, Length@tsRad];
   {"a" → a, "tsDeg" → tsDeg,
    "tsRad" → tsRad,
    "alphas" → alphas}];

In[382]:= ellAlphaT15 = getEllAlphaT[1.5];
(*Save["data/ellAlphaT_a15.m",ellAlphaT15];*)
```

In[383]:=
```
getTriAlphaT[a_] := Module[{tsDeg, tsRad, alphas},
    tsDeg = Range[360] - 1;
    tsRad = toRad /@ tsDeg;
    alphas = getAlpha[a, #] & /@ tsRad;
    {"a" → a, "tsDeg" → tsDeg,
     "tsRad" → tsRad,
     "alphas" → alphas}];
```

In[384]:=
```
triAlphaT15 = getTriAlphaT[1.5];
(*Save["data/triAlphaT_a15.m",triAlphaT15];*)
```

In[385]:=
```
Clear@getTriVtx0;(* not for error purposes *)
getTriVtx0[a_, p1_, alpha_] := Module[{p2, p2Neg, p3},
    {p2, p2Neg} = getP2Alpha[a, p1, alpha];
    {p1, p2, p2Neg}];
```

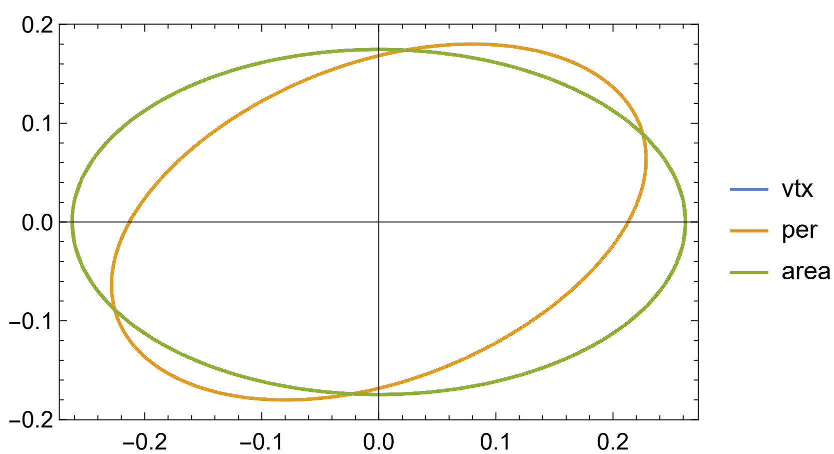
In[387]:=
```
getCentroidRadialStatsTable[triAlphaT15, getTriVtx0]
```

Out[387]=

type	mean	sd	zscore
vtx	0.220564	0.0308089	0.139682
perimeter	0.20213	0.0310537	0.153632
area	0.220564	0.0308089	0.139682

In[388]:=
```
showCentroidPaths[triAlphaT15, getTriVtx0]
```

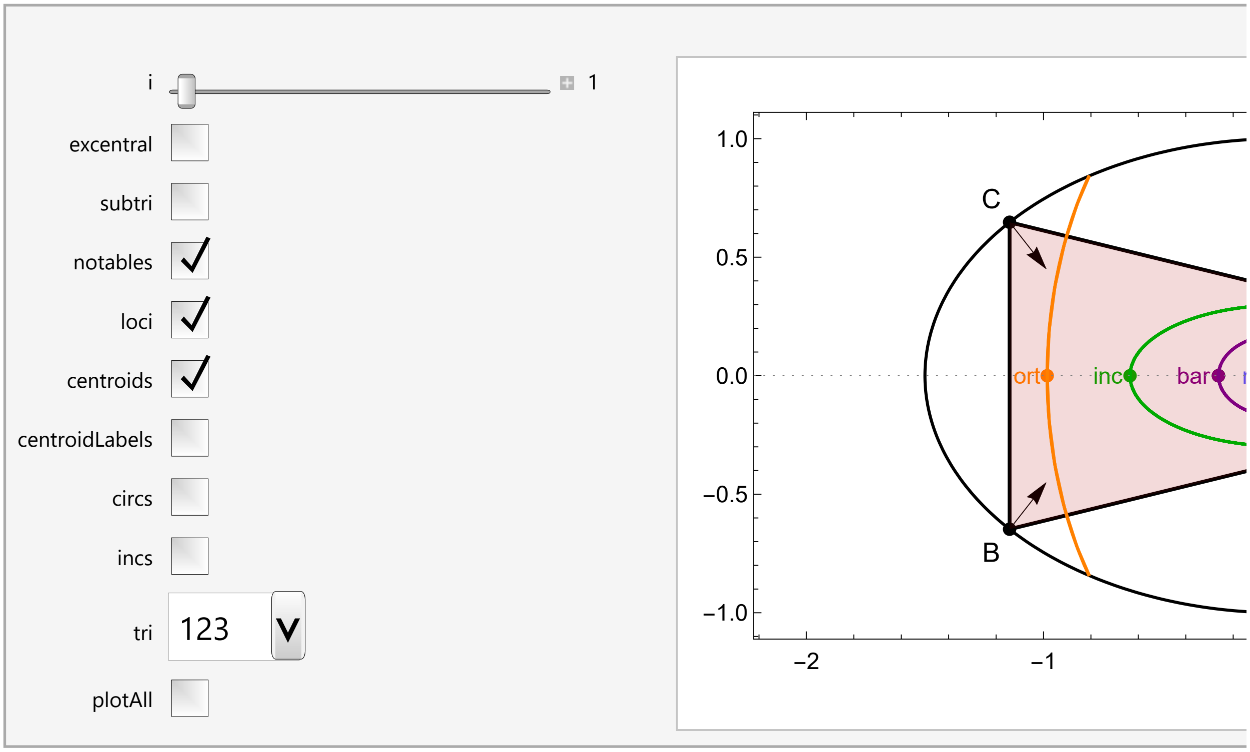
Out[388]=



In[389]:=
```
triNotableLoci15 = getPolyNotableLoci[
    triAlphaT15, getTriVtx0, {1, 2, 3}, {"inc", "bar", "ort", "mit"}];
```

In[390]:=
```
manipulatePolyVtx[triAlphaT15, {triNotableLoci15}, getTriVtx0, triErrorP]
```

Out[390]=

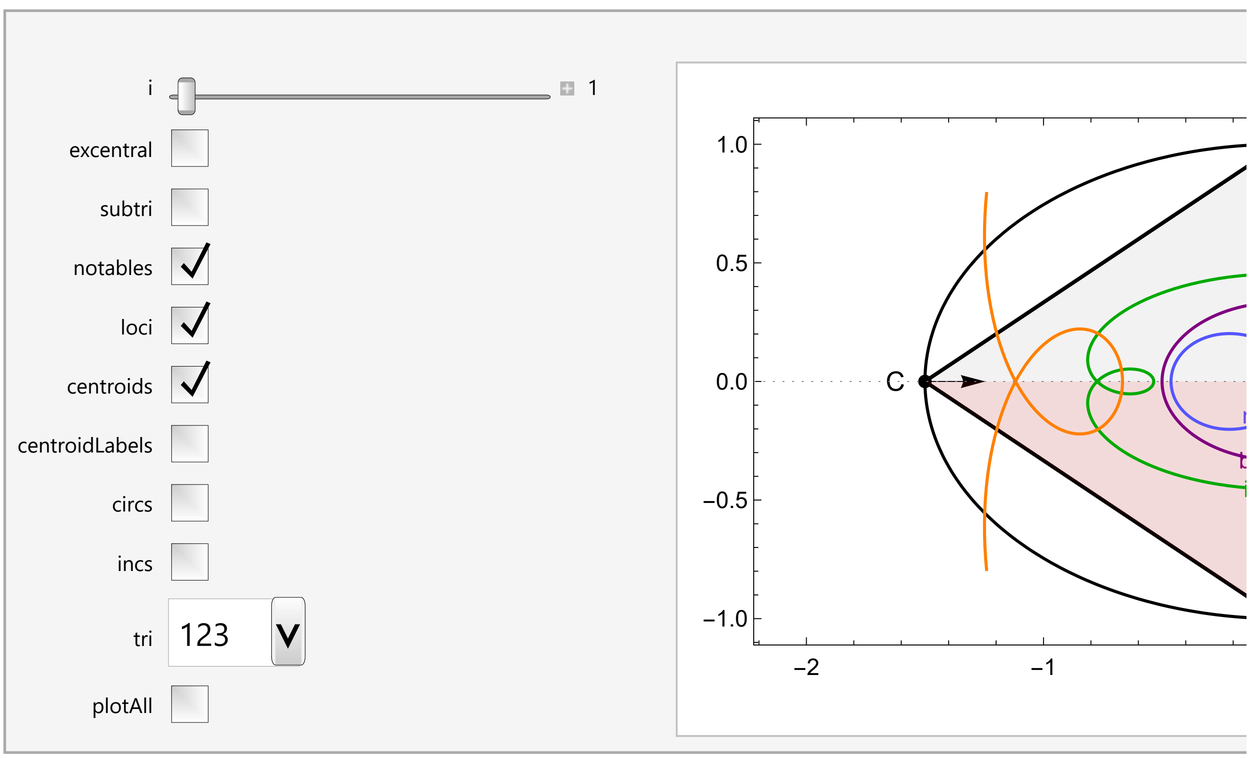


Quadrangle

In[391]:=
```
Clear@getQuadVtx;
getQuadVtx[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p3Neg = getInterRefl[a, p1, p2Neg];
   {p2, p2Neg, p3, p3Neg}];
Clear@getQuadVtx0;(* not for error purposes *)
getQuadVtx0[a_, p1_, alpha_] := Module[{p2, p2Neg, p3},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   {p1, p2, p3, p2Neg}];
```

```
In[395]:= Clear@quadError;
quadError[a_, t_, alpha_] := Module[{p1, qv},
   p1 = {a Cos[t], Sin[t]};
   qv = getQuadVtx[a, p1, alpha];
   (* p3Neg - p3 *)
   magn2[qv[[3]] - qv[[4]]]];
quadErrorP[a_, p1_, alpha_] := Module[{qv},
   qv = getQuadVtx[a, p1, alpha];
   (* p3Neg - p3 *)
   magn2[qv[[3]] - qv[[4]]]];

In[398]:= Clear@quadAlphaT15;
quadAlphaT15 =
  calcAlphaT[False, quadErrorP, "data/quadAlphaT_a15.m", 1.5, .794, 1, False];

loaded: 360 records fromdata/quadAlphaT_a15.m

In[400]:= quadNotableLoci15 = getPolyNotableLoci[
   quadAlphaT15, getQuadVtx0, {1, 2, 3}, {"inc", "bar", "ort", "mit"}];

In[401]:= manipulatePolyVtx[quadAlphaT15, {quadNotableLoci15}, getQuadVtx0, quadErrorP]
```

Out[401]=



Centroid Invariance: zscores seem high because mean is so small

In[402]:=
```
getCentroidRadialStatsTable[quadAlphaT15, getQuadVtx0]
```

Out[402]=

type	mean	sd	zscore
vtx	6.20891×10^{-9}	2.04343×10^{-8}	3.29113
perimeter	4.71547×10^{-9}	1.63338×10^{-8}	3.46387
area	3.82033×10^{-9}	1.32516×10^{-8}	3.4687

Sum of Cosines of Quad Internal Angles: CONSTANT

In[403]:=
```
Module[{sumCosT},
 sumCosT =
  Table[sumPolyCosines[quadAlphaT15, i, getQuadVtx0], {i, Length@quadAlphaT15}];
 {Mean@sumCosT, StandardDeviation@sumCosT}]
```

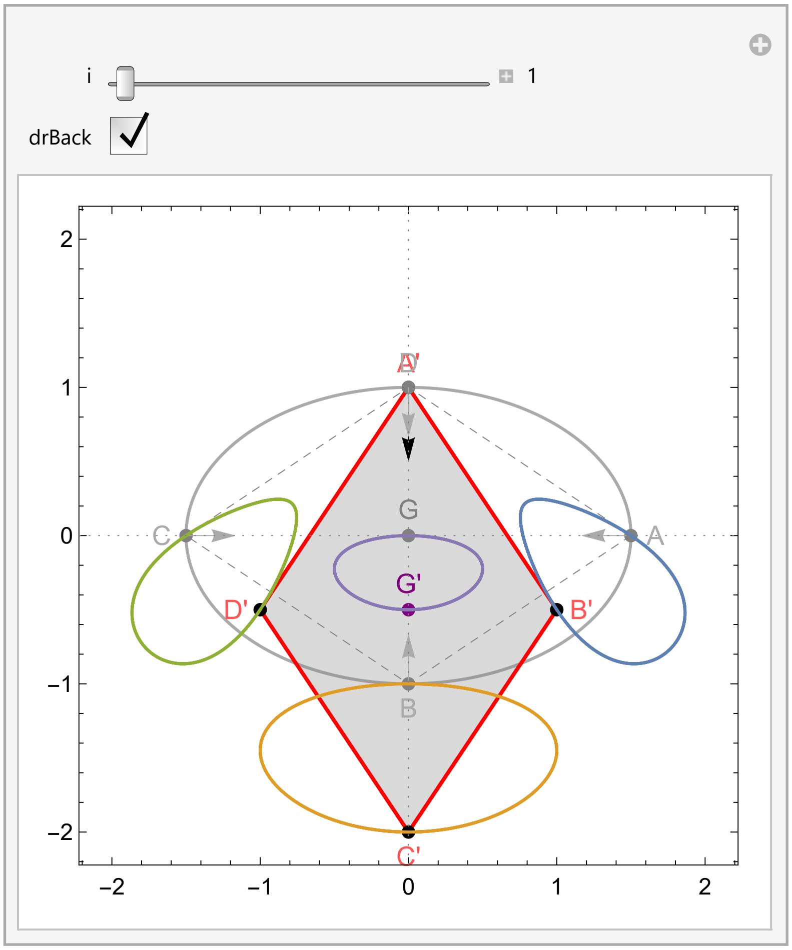
Out[403]= $\{-2.66454\times10^{-16}, 2.19953\times10^{-16}\}$

Vertical Quadrangle

In[404]:=
```
quadVertLoci15 = getPolyVertLoci[quadAlphaT15, getQuadVtx0];
```

In[405]:=
```
Manipulate[showOnePolyVert[quadAlphaT15,
   i, quadVertLoci15, getQuadVtx0, quadErrorP, drBack],
 {{i, 1}, 1, Length["alphas" /. quadAlphaT15], 1, Appearance -> "Labeled"},
 {{drBack, True}, {True, False}},
 SaveDefinitions -> True]
```

Out[405]=

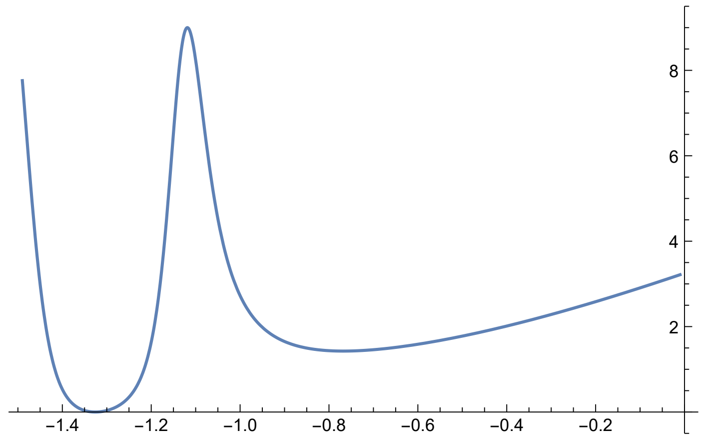


Pentagon Alpha w/ Caustic

In[406]:=
```
Clear@pentErrCaustic;
pentErrCaustic[a_, x1_] := Module[{p3, p2, p1},
   p3 = {x1, ellY[a, x1]};
   p2 = getInterRefl[a, {x1, 0}, p3];
   p1 = getInterRefl[a, p3, p2];
   magn2[p1 - {a, 0}]
  ];
```

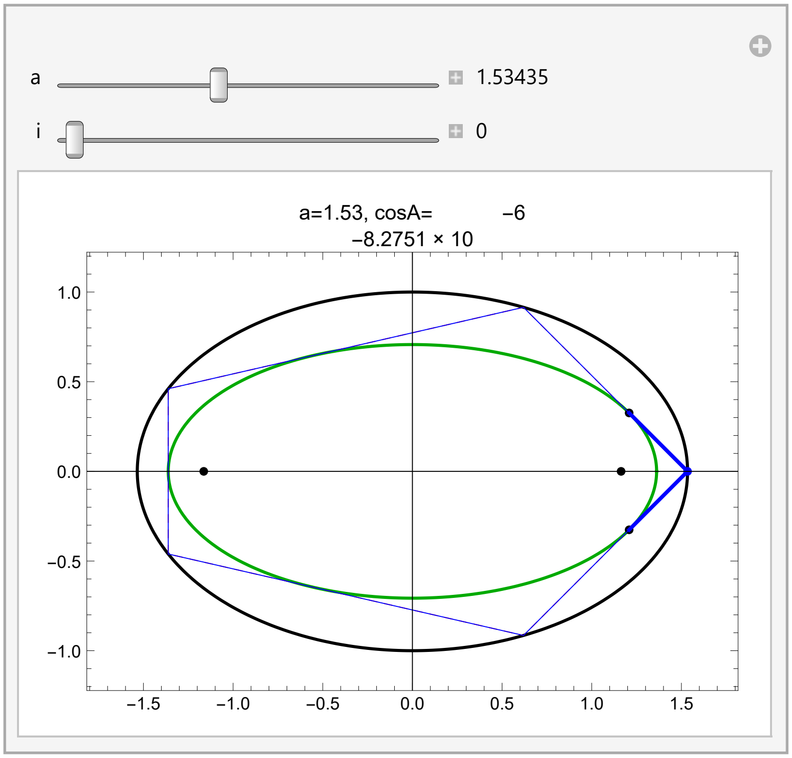
In[408]:=
```
Plot[pentErrCaustic[1.5, x1], {x1, -1.49, -.01}, PlotRange → All]
```

Out[408]=



In[409]:=
```
Manipulate[DynamicModule[
  {acaustic, bcaustic, tangs, p1, p2, p1rot, p2rot, n1, ca, fs, sa, n1rot, min, ell,
   caustic, gr, maxX, maxY = 1.1, poly, polyRot, tRad, cosAlphas, cosA, lab},
  {acaustic, bcaustic, tangs, p1, p2, min} = getAlpha0[a, pentErrCaustic, -a + .1];
  cosA = norm[tangs[[1]] - p1].norm[tangs[[2]] - p1];
  lab = "a=" <> nfn[a, 2] <> ",  cosA=" <> nfn[cosA, 4];
  ell = plotEll[a];
  cosAlphas = getCosAlphasCaustic[a, acaustic, bcaustic];
  caustic = plotEllb[acaustic, bcaustic, Darker@Green];
  poly = bounceRay[a, p1, p2, 4];
  maxX = a + .1;
  fs = getFoci[a];
  Dynamic[tRad = toRad[N@i];
   p1rot = {a Cos[tRad], Sin[tRad]};
   n1 = norm[ellGrad[a, Sequence @@ p1rot]];
   ca = cosAlphas[[i + 1]];
   sa = Sqrt[1 - ca^2];
   n1rot = rot[n1, sa, ca];
   p2rot = ellInterRayUnprot[a, p1rot, n1rot][[2]];
   polyRot = bounceRay[a, p1rot, p2rot, 4];
   gr = Graphics[{PointSize@Medium, Black, Point@tangs, Point@fs,
      {Red, Dashed, Line@poly},
      {Blue, Line@polyRot, Point[polyRot[[1]]]},
      {Thick, Blue, Line[{p1, tangs[[1]]}], Line[{p1, tangs[[2]]}]}}];
   Show[{ell, caustic, gr},
    PlotRange → {{-maxX, maxX}, {-maxY, maxY}}, Frame -> True, PlotLabel → lab]]],
 {{a, 1.53435}, .5, 3, .001, Appearance → "Labeled"},
 {{i, 0}, 0, 90(*length of cosAlphas*), 1, Appearance → "Labeled"}]
```

Out[409]=



What is the a for which the exit angle is 90 degrees?

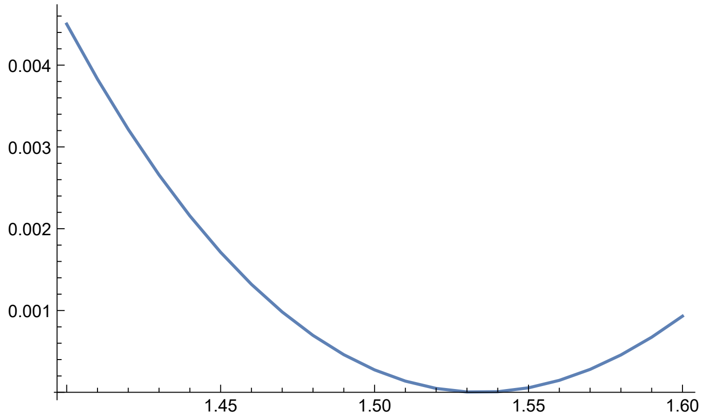
In[410]:=
```
Clear@pentCosA;
pentCosA[a_] := Module[{tangs, p1, cosA, guess},
    guess = .1 - a;
    {tangs, p1} = Part[getAlpha0[a, pentErrCaustic, guess], {3, 4}];
    cosA = norm[tangs[[1]] - p1].norm[tangs[[2]] - p1];
    cosA];
```

In[412]:=
```
pentCosA[1.534352]
```

Out[412]= -7.32339×10^{-6}

In[413]:=
```
ListPlot[Table[{a, pentCosA[a]^2}, {a, 1.4, 1.6, .01}], Joined → True]
```

Out[413]=



In[414]:=
```
Clear@getPentVtx;
getPentVtx[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p3Neg = getInterRefl[a, p1, p2Neg];
   p4Neg = getInterRefl[a, p2Neg, p3Neg];
   {p2, p2Neg, p3, p3Neg, p4, p4Neg}];

Clear@getPentVtx0;(* not for error purposes *)
getPentVtx0[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p3Neg = getInterRefl[a, p1, p2Neg];
   {p1, p2, p3, p3Neg, p2Neg}];
```

In[418]:=
```
Clear@pentErrorP;
pentErrorP[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg},
   {p2, p2Neg, p3, p3Neg, p4, p4Neg} = getPentVtx[a, p1, alpha];
   (* p3Neg - p3 *)
   magn2[p3Neg - p4] + magn2[p3 - p4Neg]];
Clear@pentErrorPabs;
pentErrorPabs[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg},
  {p2, p2Neg, p3, p3Neg, p4, p4Neg} = getPentVtx[a, p1, alpha];
  (* p3Neg - p3 *)
  magn[p3Neg - p4] + magn[p3 - p4Neg]];
Clear@pentError;
pentError[a_, t_, alpha_] := Module[{p1, pv},
   p1 = {a Cos[t], Sin[t]};
   pentErrorP[a, p1, alpha]];
```

Pentagon Exit Angle Table, Use Caustics!

In[423]:=
```
Clear@pentAlphaT125;
pentAlphaT125 = calcAlphaCausticT[False(*False for load*),
   pentErrCaustic, "data/pentAlphaCausticT_a125.m", 1.25, 1];
```

```
loaded: 360 records fromdata/pentAlphaCausticT_a125.m
```

In[425]:=
```
Clear@pentAlphaT15;
pentAlphaT15 = calcAlphaCausticT[False(*False for load*),
   pentErrCaustic, "data/pentAlphaCausticT_a15.m", 1.5, 1];
```

```
loaded: 360 records fromdata/pentAlphaCausticT_a15.m
```

In[427]:=
```
Clear@pentAlphaT20;
pentAlphaT20 = calcAlphaCausticT[False(*False for load*),
   pentErrCaustic, "data/pentAlphaCausticT_a20.m", 2.0, 1];
```

```
loaded: 360 records fromdata/pentAlphaCausticT_a20.m
```

In[429]:=
```
Clear@pentAlphaT30;
pentAlphaT30 = calcAlphaCausticT[False(*False for load*),
   pentErrCaustic, "data/pentAlphaCausticT_a30.m", 3.0, 1];
```

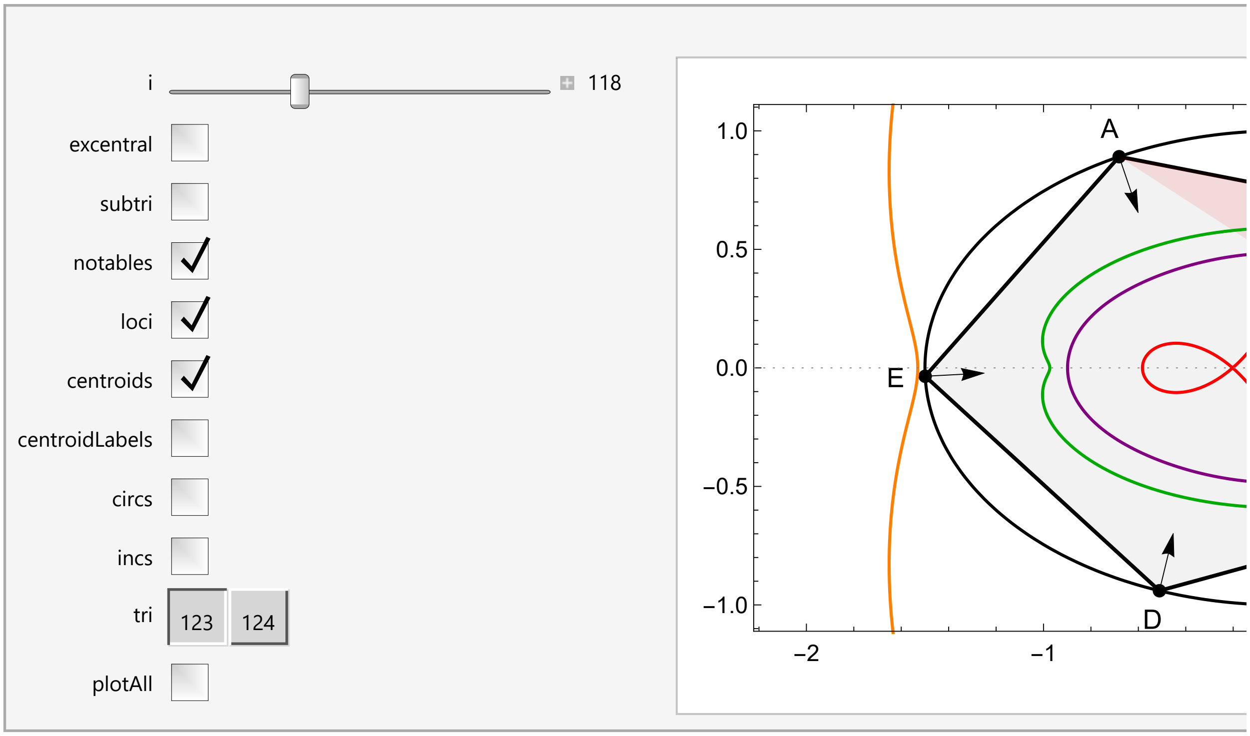
```
loaded: 360 records fromdata/pentAlphaCausticT_a30.m
```

Draw Pentagon With Loci

In[431]:=
```
pentVtxList = {{1, 2, 3}, {1, 2, 4}};
pentNotableLociList =
  getPolyNotableLoci[pentAlphaT15, getPentVtx0, #, {"bar", "inc", "cir", "ort"}] & /@
   pentVtxList;
```

In[433]:=
```
manipulatePolyVtx[pentAlphaT15,
 pentNotableLociList, getPentVtx0, pentErrorP, pentVtxList]
```

Out[433]=



Centroid Invariance: zscores seem high because mean is so small

In[434]:=
```
getCentroidRadialStatsTable[pentAlphaT15, getPentVtx0]
```

Out[434]=

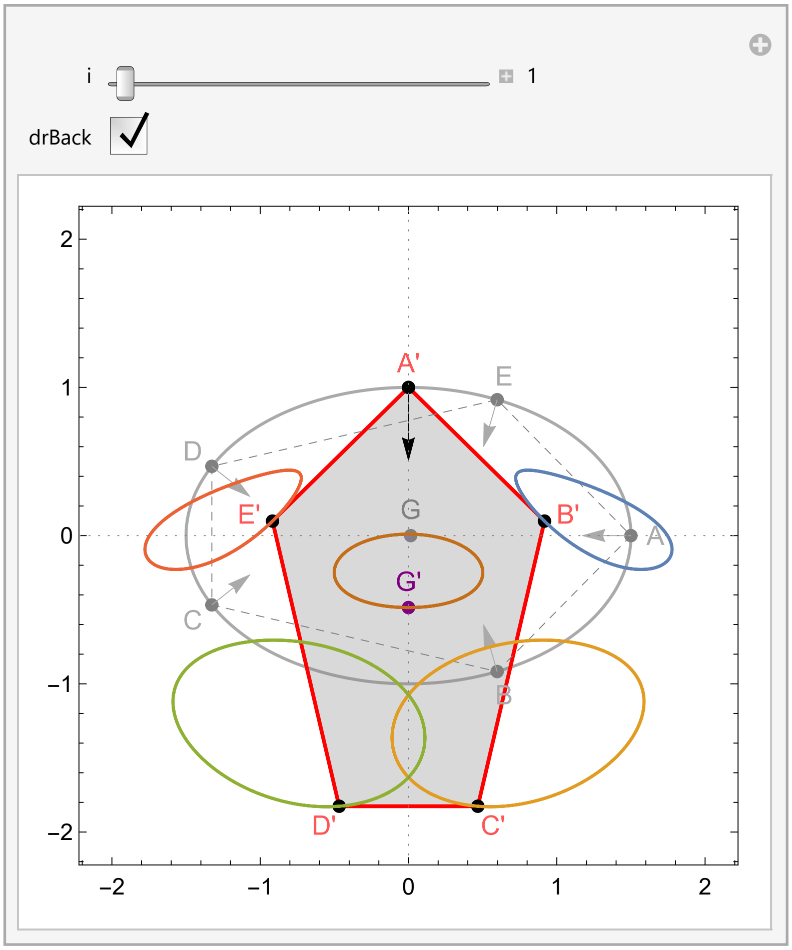
type	mean	sd	zscore
vtx	0.00738912	0.00103213	0.139682
perimeter	0.0249026	0.00347864	0.139689
area	0.0123152	0.00172021	0.139682

Vertical Pentagon

In[435]:=
```
pentVertLoci15 = getPolyVertLoci[pentAlphaT15, getPentVtx0];
```

In[436]:=
```
Manipulate[showOnePolyVert[pentAlphaT15,
   i, pentVertLoci15, getPentVtx0, pentErrorP, drBack],
 {{i, 1}, 1, Length["alphas" /. pentAlphaT15], 1, Appearance -> "Labeled"},
 {{drBack, True}, {True, False}},
 SaveDefinitions -> True]
```

Out[436]=



Pentagon: Area and Centroid Invariance: Near Miss

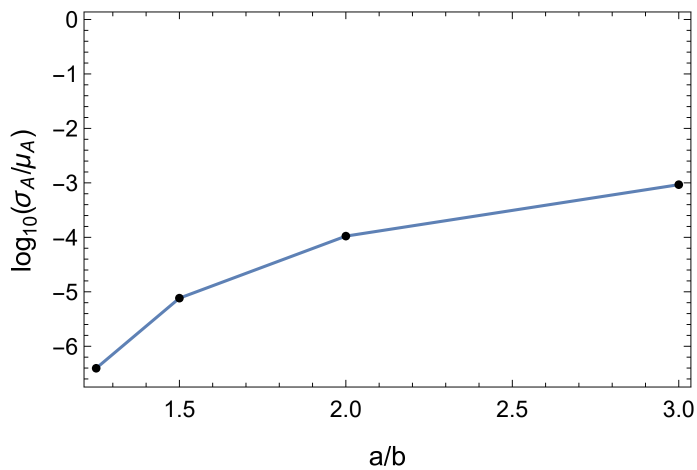
In[437]:=
```
Prepend[reportPolyAreaStats[#, getPentVtx0] & /@
   {pentAlphaT125, pentAlphaT15, pentAlphaT20, pentAlphaT30},
  {"a", "N", "degStep", "μ", "sd", "sd/μ"}] // Grid[#, Frame → All] &
```

Out[437]=

a	N	degStep	μ	sd	sd/μ
1.25	1	360	2.95269	1.16449×10^{-6}	3.94384×10^{-7}
1.5	1	360	3.49121	0.0000266885	7.64447×10^{-6}
2.	1	360	4.47678	0.000471248	0.000105265
3.	1	360	6.19187	0.00573595	0.000926368

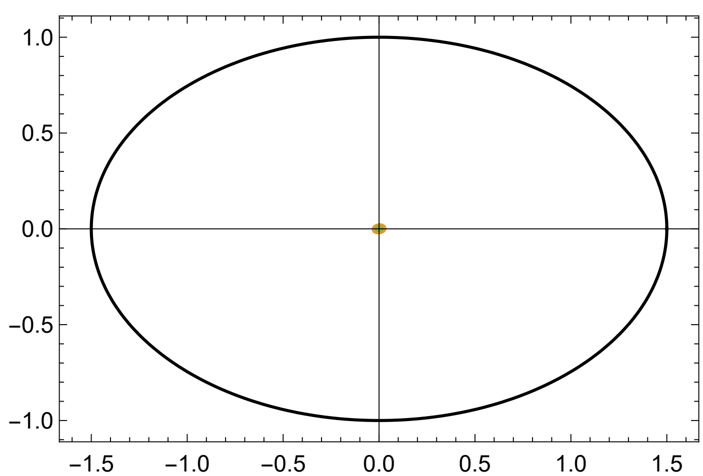
In[438]:=
```
Module[{pts},
  pts = {#[[1]], Log10[#[[6]]]} & /@ (reportPolyAreaStats[#, getPentVtx0] & /@
      {pentAlphaT125, pentAlphaT15, pentAlphaT20, pentAlphaT30});
  ListLinePlot[pts, Epilog → {PointSize@Medium, Point@pts}, Frame → True,
   FrameStyle → Medium, FrameLabel → (Style[#, 14] & /@ {"a/b", "log10(σA/μA)"})]]
```

Out[438]=



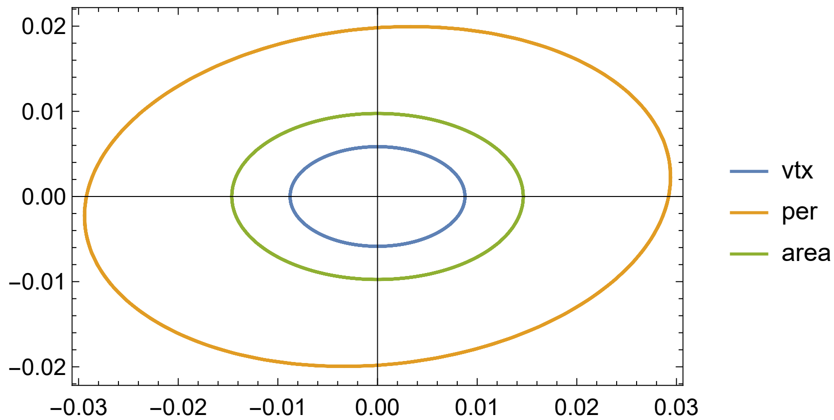
In[439]:=
```
Show[{plotEll["a" /. pentAlphaT15],
   ListLinePlot@getCentroidPath[pentAlphaT15, getPentVtx0]},
  Frame → True, FrameStyle → Medium]
```

Out[439]=



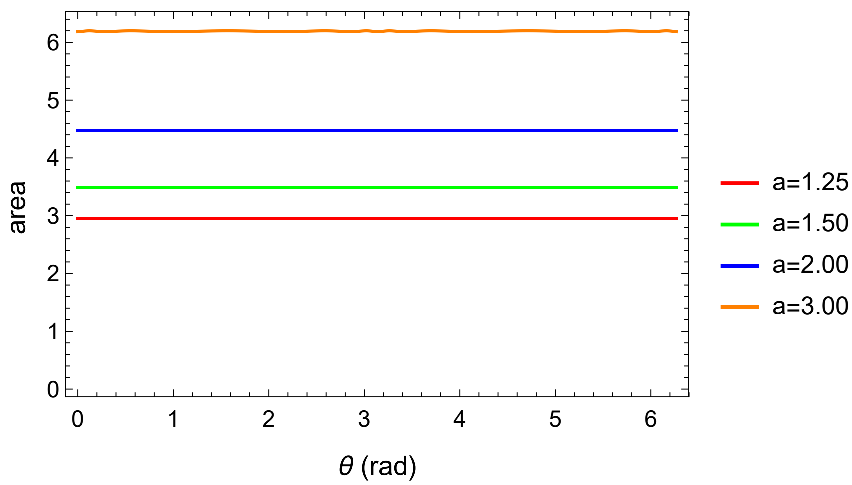
In[440]:=
```
showCentroidPaths[pentAlphaT15, getPentVtx0]
```

Out[440]=



In[441]:=
```
Module[{clrs = {Red, Green, Blue, Orange},
  pentAlphaTs = {pentAlphaT125, pentAlphaT15, pentAlphaT20, pentAlphaT30}},
 Legended[Show[MapThread[showPolyArea[#1, getPentVtx0, #2] &, {pentAlphaTs, clrs}],
   PlotRange → All, ImageSize → Medium],
  LineLegend[Directive[{Thick, #}] & /@ clrs,
   ("a=" <> nfn["a" /. #, 2]) & /@ pentAlphaTs]]]
```

Out[441]=



Hexagon

In[442]:=
```
Clear@getHexVtx;
getHexVtx[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p3Neg = getInterRefl[a, p1, p2Neg];
   p4Neg = getInterRefl[a, p2Neg, p3Neg];
   {p2, p2Neg, p3, p3Neg, p4, p4Neg}];

Clear@getHexVtx0;(* not for error purposes *)
getHexVtx0[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p4, p3Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p3Neg = getInterRefl[a, p1, p2Neg];
   {p1, p2, p3, p4, p3Neg, p2Neg}];
```

In[446]:=
```
Clear@hexErrorP; hexErrorP[a_, p1_, alpha_] := Module[{p4, p4Neg},
  {p4, p4Neg} = Part[getHexVtx[a, p1, alpha], {5, 6}];
  (* p3Neg - p3 *)
  magn2[p4 - p4Neg]];
Clear@hexErrorPabs;
hexErrorPabs[a_, p1_, alpha_] := Sqrt@hexErrorP[a, p1, alpha];
```

In[448]:=
```
Clear@hexError;
hexError[a_, t_, alpha_] := Module[{p1, pv},
   p1 = {a Cos[t], Sin[t]};
   hexErrorP[a, p1, alpha]];
```

In[450]:=
```
Clear@getHexVtxHalf;
getHexVtxHalf[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p1sym, p2sym, p3sym},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p1sym = -p1;
   p2sym = -p2Neg;
   {p3, p2sym}];
Clear@hexErrorPhalf;
hexErrorPhalf[a_, p1_, alpha_] := Module[{p3, p2sym},
  {p3, p2sym} = getHexVtxHalf[a, p1, alpha];
  (* p3Neg - p3 *)
  magn2[p3 - p2sym]];
```

Hexagon Exit Angle Table

In[453]:=
```
Clear@hexAlphaT15;
hexAlphaT15 =
  calcAlphaT[False, hexErrorPhalf, "data/hexAlphaT_a15.m", 1.5, .9273, 1, False];
```

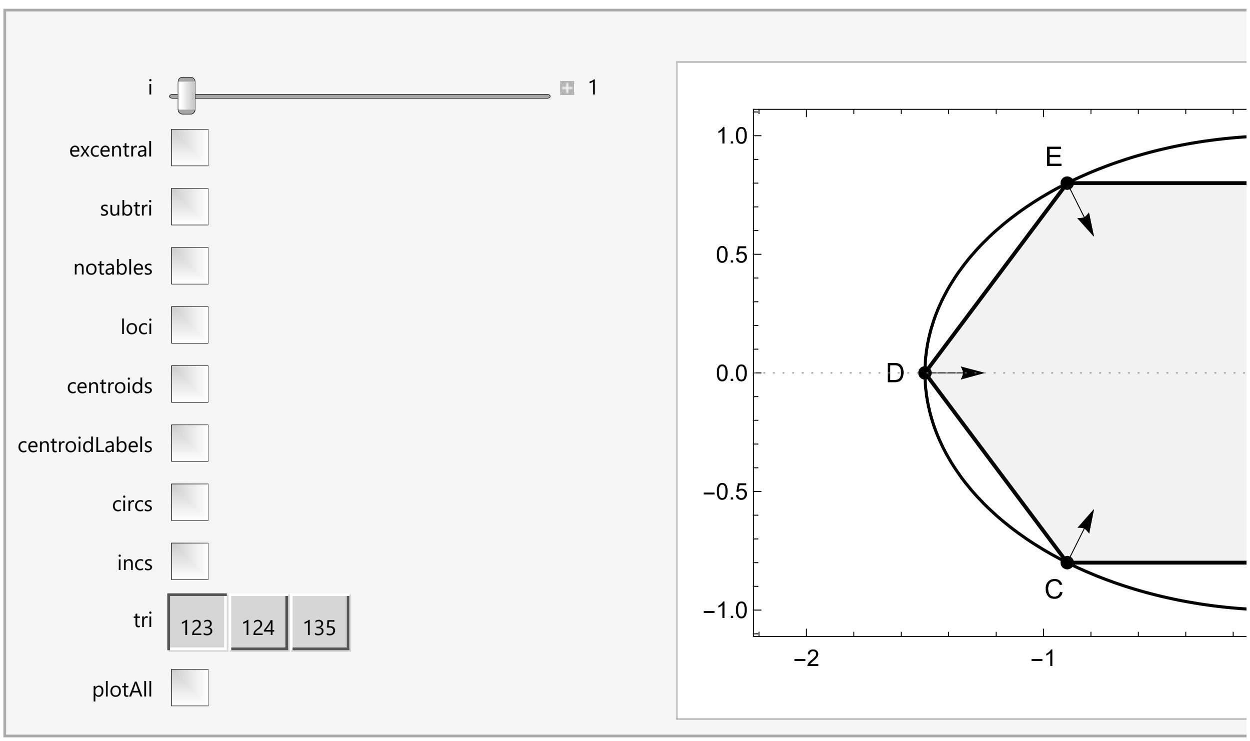
```
loaded: 360 records fromdata/hexAlphaT_a15.m
```

Draw Hexagon With Loci

In[455]:=
```
hexVtxList = {{1, 2, 3}, {1, 2, 4}, {1, 3, 5}};
hexNotableLociList =
  getPolyNotableLoci[hexAlphaT15, getHexVtx0, #, {"bar", "inc", "cir", "ort"}] & /@
   hexVtxList;
```

In[457]:=
```
manipulatePolyVtx[hexAlphaT15,
 hexNotableLociList, getHexVtx0, hexErrorP, hexVtxList]
```

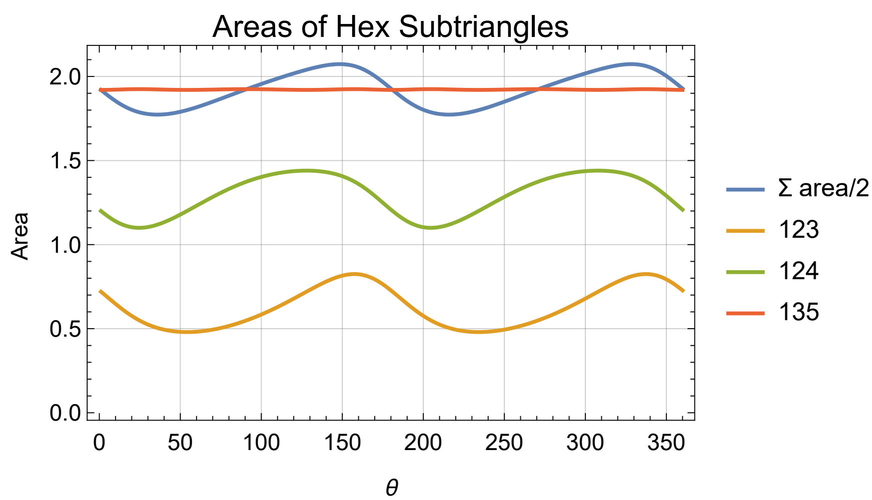
Out[457]=



Area of (1,3,5) subtriangle very stable

In[458]:=
```
Module[{polys, n, areas, vtx = {{1, 2, 3}, {1, 2, 4}, {1, 3, 5}}},
 n = Length["alphas" /. hexAlphaT15];
 polys = Table[polyVtx[hexAlphaT15, i, getHexVtx0], {i, n}];
 areas = Table[Area[Polygon[Part[polys[[i]], #]]] & /@ vtx, {i, n}];
 (* add totals *)
 areas = Transpose[Prepend[#, Total[#] / 2] & /@ areas];
 ListLinePlot[
  MapThread[Legended[#1, #2] &, {areas, {"Σ area/2", "123", "124", "135"}}],
  PlotStyle → Thick, GridLines → Automatic, Frame → True, FrameStyle → "Medium",
  PlotLabel → Style["Areas of Hex Subtriangles", Directive[Black, 16]],
  FrameLabel → {"θ", "Area"}]
]
```

Out[458]=



In[459]:=

Hexagon: Centroid Invariance: zscores seem high because mean is so small

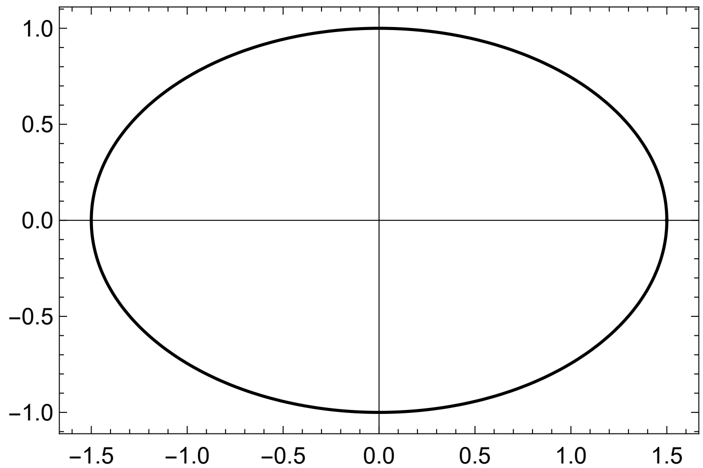
In[460]:=
```
getCentroidRadialStatsTable[hexAlphaT15, getHexVtx0]
```

Out[460]=

type	mean	sd	zscore
vtx	4.41845×10^{-8}	1.83075×10^{-7}	4.14341
perimeter	2.15659×10^{-8}	1.02857×10^{-7}	4.7694
area	1.40399×10^{-8}	6.13123×10^{-8}	4.36702

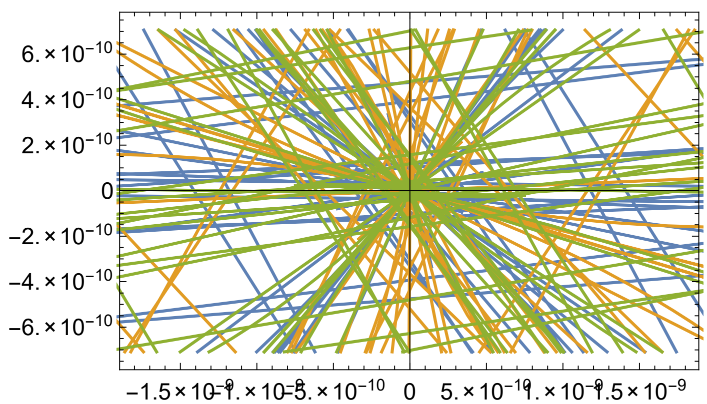
In[461]:=
```
Show[{plotEll["a" /. hexAlphaT15],
  ListLinePlot[getCentroidPath[hexAlphaT15, getHexVtx0],
   Epilog → {PointSize@Large, Blue, Point[{0, 0}], Point /@ hexCentroidMeans}]},
 Frame → True, FrameStyle → Medium]
```

Out[461]=



In[462]:=
```
Show[{ListLinePlot[getCentroidPath[hexAlphaT15, getHexVtx0]]},
 Frame → True, FrameStyle → Medium]
```

Out[462]=



Heptagon w/ Caustic

In[463]:=
```
Clear@getHeptVtx;
getHeptVtx[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg, p5, p5Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p5 = getInterRefl[a, p3, p4];
   p3Neg = getInterRefl[a, p1, p2Neg];
   p4Neg = getInterRefl[a, p2Neg, p3Neg];
   p5Neg = getInterRefl[a, p3Neg, p4Neg];
   {p2, p2Neg, p3, p3Neg, p4, p4Neg, p5, p5Neg}];

Clear@getHeptVtx0;(* not for error purposes *)
getHeptVtx0[a_, p1_, alpha_] := Module[{p2, p3, p4, p2Neg, p3Neg, p4Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p3Neg = getInterRefl[a, p1, p2Neg];
   p4Neg = getInterRefl[a, p2Neg, p3Neg];
   {p1, p2, p3, p4, p4Neg, p3Neg, p2Neg}];
```

In[467]:=
```
Clear@heptErrorP;
heptErrorP[a_, p1_, alpha_] := Module[{p4, p4Neg, p5, p5Neg},
  {p4, p4Neg, p5, p5Neg} = Part[getHeptVtx[a, p1, alpha], {5, 6, 7, 8}];
  magn2[p4 - p5Neg] + magn2[p5 - p4Neg]];
Clear@heptErrorPabs;
heptErrorPabs[a_, p1_, alpha_] := Sqrt@heptErrorP[a, p1, alpha];
Clear@heptError;
heptError[a_, t_, alpha_] := Module[{p1, pv},
   p1 = {a Cos[t], Sin[t]};
   heptErrorP[a, p1, alpha]];
```

In[471]:=
```
Clear@heptErrCaustic;
heptErrCaustic[a_, x1_] := Module[{p4, p3, p2, p1},
   p4 = {x1, ellY[a, x1]};
   p3 = getInterRefl[a, {x1, 0}, p4];
   p2 = getInterRefl[a, p4, p3];
   p1 = getInterRefl[a, p3, p2];
   magn2[p1 - {a, 0}]
  ];
```

Pentagon Exit Angle Table, Use Caustics!

In[473]:=
```
Clear@heptAlphaT125;
heptAlphaT125 = calcAlphaCausticT[False(*False for load*),
   heptErrCaustic, "data/heptAlphaCausticT_a125.m", 1.25, 1];
```

```
loaded: 360 records fromdata/heptAlphaCausticT_a125.m
```

In[475]:=
```
Clear@heptAlphaT15;
heptAlphaT15 = calcAlphaCausticT[False(*False for load*),
   heptErrCaustic, "data/heptAlphaCausticT_a15.m", 1.5, 1];
```

```
loaded: 360 records fromdata/heptAlphaCausticT_a15.m
```

In[477]:=
```
Clear@heptAlphaT20;
heptAlphaT20 = calcAlphaCausticT[False(*False for load*),
   heptErrCaustic, "data/heptAlphaCausticT_a20.m", 2.0, 1];
```

```
loaded: 360 records fromdata/heptAlphaCausticT_a20.m
```

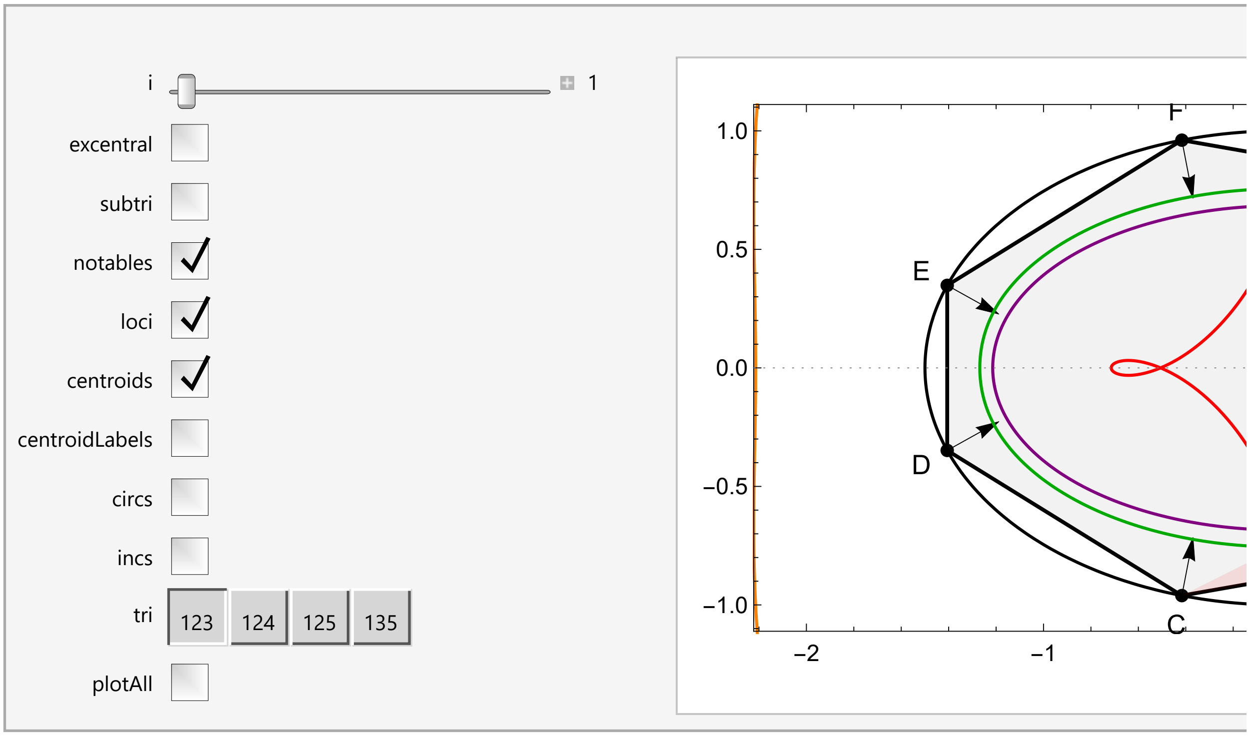
In[479]:=
```
Clear@heptAlphaT30;
heptAlphaT30 = calcAlphaCausticT[False(*False for load*),
   heptErrCaustic, "data/heptAlphaCausticT_a30.m", 3.0, 1];
```

```
loaded: 360 records fromdata/heptAlphaCausticT_a30.m
```

In[481]:=
```
heptVtxList = {{1, 2, 3}, {1, 2, 4}, {1, 2, 5}, {1, 3, 5}};
heptNotableLocis =
  getPolyNotableLoci[heptAlphaT15, getHeptVtx0, #, {"bar", "inc", "cir", "ort"}] & /@
   heptVtxList;
```

In[483]:=
```
manipulatePolyVtx[heptAlphaT15,
 heptNotableLocis, getHeptVtx0, heptErrorP, heptVtxList]
```

Out[483]=



Heptagon: Centroid Invariance

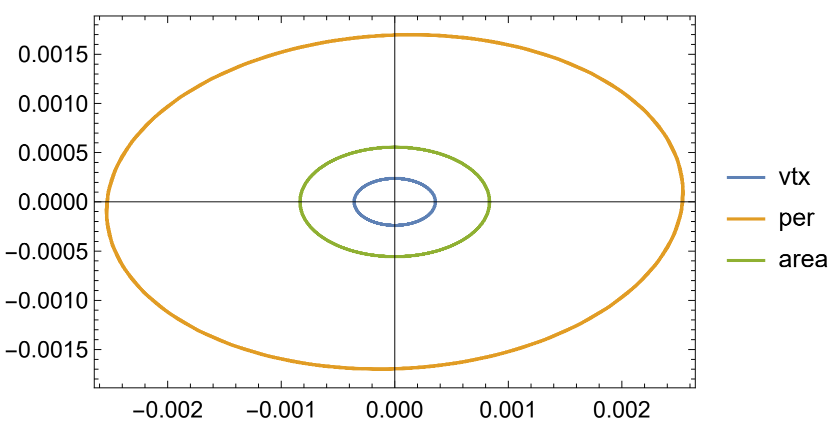
In[484]:=
```
getCentroidRadialStatsTable[heptAlphaT15, getHeptVtx0]
```

Out[484]=

type	mean	sd	zscore
vtx	0.000301348	0.000042093	0.139682
perimeter	0.00214133	0.000299106	0.139682
area	0.000703145	0.0000982168	0.139682

In[485]:=
```
showCentroidPaths[heptAlphaT15, getHeptVtx0]
```

Out[485]=



Octagon

In[486]:=
```
Clear@getOctVtx;
getOctVtx[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg, p5, p5Neg},
    {p2, p2Neg} = getP2Alpha[a, p1, alpha];
    p3 = getInterRefl[a, p1, p2];
    p4 = getInterRefl[a, p2, p3];
    p5 = getInterRefl[a, p3, p4];
    p3Neg = getInterRefl[a, p1, p2Neg];
    p4Neg = getInterRefl[a, p2Neg, p3Neg];
    p5Neg = getInterRefl[a, p3Neg, p4Neg];
    {p2, p2Neg, p3, p3Neg, p4, p4Neg, p5, p5Neg}];
Clear@getOctVtx0;(* not for error purposes *)
getOctVtx0[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p4, p5, p3Neg, p4Neg},
    {p2, p2Neg} = getP2Alpha[a, p1, alpha];
    p3 = getInterRefl[a, p1, p2];
    p4 = getInterRefl[a, p2, p3];
    p5 = getInterRefl[a, p3, p4];
    p3Neg = getInterRefl[a, p1, p2Neg];
    p4Neg = getInterRefl[a, p2Neg, p3Neg];
    {p1, p2, p3, p4, p5, p4Neg, p3Neg, p2Neg}];
Clear@octErrorP; octErrorP[a_, p1_, alpha_] := Module[{p5, p5Neg},
   {p5, p5Neg} = Part[getOctVtx[a, p1, alpha], {7, 8}];
   (* p3Neg - p3 *)
   magn2[p5 - p5Neg]];
```

In[491]:=
```
Clear@getOctVtxHalf;
getOctVtxHalf[a_, p1_, alpha_] := Module[{p2, p2Neg, p3, p1sym, p2sym, p3sym},
    {p2, p2Neg} = getP2Alpha[a, p1, alpha];
    p3 = getInterRefl[a, p1, p2];
    p1sym = -p1;
    p2sym = -p2Neg;
    p3sym = getInterRefl[a, p1sym, p2sym];
    {p3, p3sym}];
Clear@octErrorPhalf;
octErrorPhalf[a_, p1_, alpha_] := Module[{p3, p3sym},
   {p3, p3sym} = getOctVtxHalf[a, p1, alpha];
   (* p3Neg - p3 *)
   magn2[p3 - p3sym]];
Clear@octErrorPabs;
octErrorPabs[a_, p1_, alpha_] := Sqrt@octErrorP[a, p1, alpha];
```

In[495]:=
```
Clear@octError;
octError[a_, t_, alpha_] := Module[{p1, pv},
   p1 = {a Cos[t], Sin[t]};
   octErrorP[a, p1, alpha]];
```

Octagon Exit Angle Table

In[497]:=
```
Clear@octAlphaT15;
octAlphaT15 =
  calcAlphaT[False, octErrorPhalf, "data/octAlphaT_a15.m", 1.5, 1.091, 1, True];
```

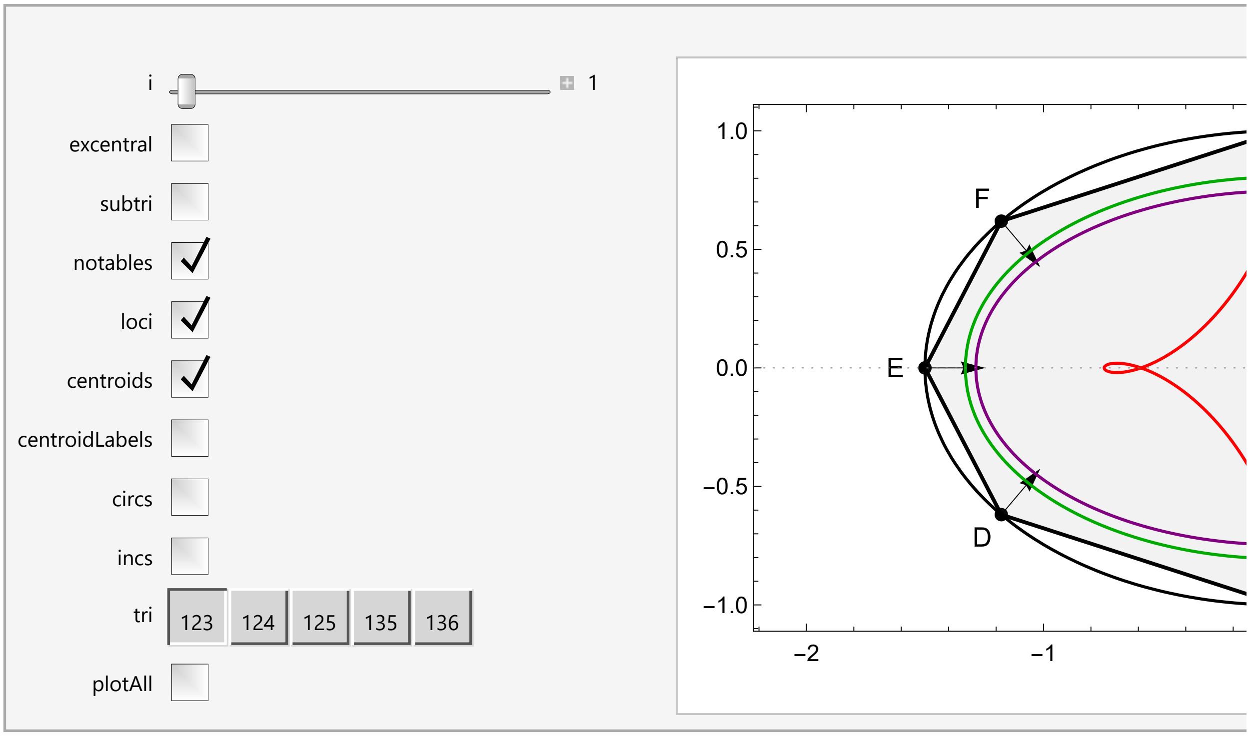
```
loaded: 360 records fromdata/octAlphaT_a15.m
```

Draw Octagon With Loci

In[499]:=
```
octVtxList = {
   {1, 2, 3}, (* sym: {1,2,8},*)
   {1, 2, 4}, (* sym: {1,2,7},{1,3,8},{1,4,8}*)
   {1, 2, 5}, (* sym: {1,2,6},refl {1,4,5}*)
   {1, 3, 5}, (* sym: {1,3,7},*)
   {1, 3, 6} (* sym: {1,4,7},{1,4,6}*)
  };
octNotableLociList =
  getPolyNotableLoci[octAlphaT15, getOctVtx0, #, {"bar", "inc", "cir", "ort"}] & /@
   octVtxList;
```

In[501]:=
```
manipulatePolyVtx[octAlphaT15,
 octNotableLociList, getOctVtx0, octErrorP, octVtxList]
```

Out[501]=



Octagon: Centroid Invariance: zscores seem high because mean is so small

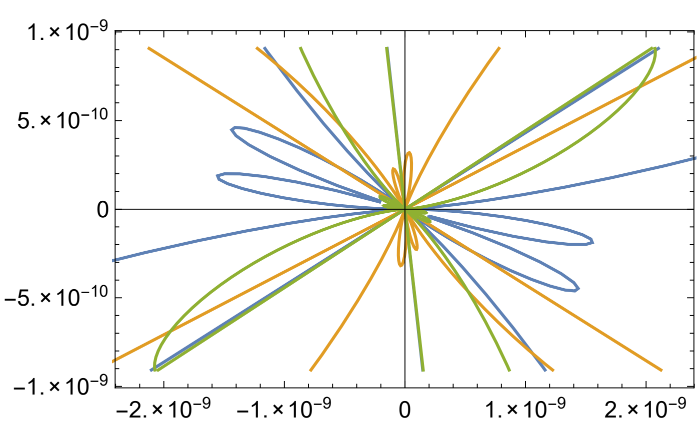
In[502]:=
```
getCentroidRadialStatsTable[octAlphaT15, getOctVtx0]
```

Out[502]=

type	mean	sd	zscore
vtx	3.01909×10^{-9}	4.42362×10^{-9}	1.46522
perimeter	1.35613×10^{-9}	2.51046×10^{-9}	1.8512
area	6.26108×10^{-10}	1.00745×10^{-9}	1.60906

In[503]:=
```
Show[{ListLinePlot[getCentroidPath[octAlphaT15, getOctVtx0]]},
 Frame → True, FrameStyle → Medium]
```

Out[503]=



Nonagon w/ Caustic

In[504]:=
```
Clear@getNonaVtx;
getNonaVtx[a_, p1_, alpha_] :=
  Module[{p2, p2Neg, p3, p3Neg, p4, p4Neg, p5, p5Neg, p6, p6Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p5 = getInterRefl[a, p3, p4];
   p6 = getInterRefl[a, p4, p5];
   p3Neg = getInterRefl[a, p1, p2Neg];
   p4Neg = getInterRefl[a, p2Neg, p3Neg];
   p5Neg = getInterRefl[a, p3Neg, p4Neg];
   p6Neg = getInterRefl[a, p4Neg, p5Neg];
   {p5, p5Neg, p6, p6Neg}];

Clear@getNonaVtx0;(* not for error purposes *)
getNonaVtx0[a_, p1_, alpha_] := Module[{p2, p3, p4, p5, p2Neg, p3Neg, p4Neg, p5Neg},
   {p2, p2Neg} = getP2Alpha[a, p1, alpha];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p5 = getInterRefl[a, p3, p4];
   p3Neg = getInterRefl[a, p1, p2Neg];
   p4Neg = getInterRefl[a, p2Neg, p3Neg];
   p5Neg = getInterRefl[a, p3Neg, p4Neg];
   {p1, p2, p3, p4, p5, p5Neg, p4Neg, p3Neg, p2Neg}];
```

In[508]:=
```
Clear@nonaErrorP;
nonaErrorP[a_, p1_, alpha_] := Module[{p5, p5Neg, p6, p6Neg},
  {p5, p5Neg, p6, p6Neg} = getNonaVtx[a, p1, alpha];
  magn2[p5 - p6Neg] + magn2[p6 - p5Neg]];
Clear@nonaErrorPabs;
nonaErrorPabs[a_, p1_, alpha_] := Sqrt@nonaErrorP[a, p1, alpha];
Clear@nonaError;
nonaError[a_, t_, alpha_] := Module[{p1, pv},
   p1 = {a Cos[t], Sin[t]};
   nonaErrorP[a, p1, alpha]];
```

In[512]:=
```
Clear@nonaErrCaustic;
nonaErrCaustic[a_, x1_] := Module[{p5, p4, p3, p2, p1},
   p5 = {x1, ellY[a, x1]};
   p4 = getInterRefl[a, {x1, 0}, p5];
   p3 = getInterRefl[a, p5, p4];
   p2 = getInterRefl[a, p4, p3];
   p1 = getInterRefl[a, p3, p2];
   magn2[p1 - {a, 0}]
  ];
```

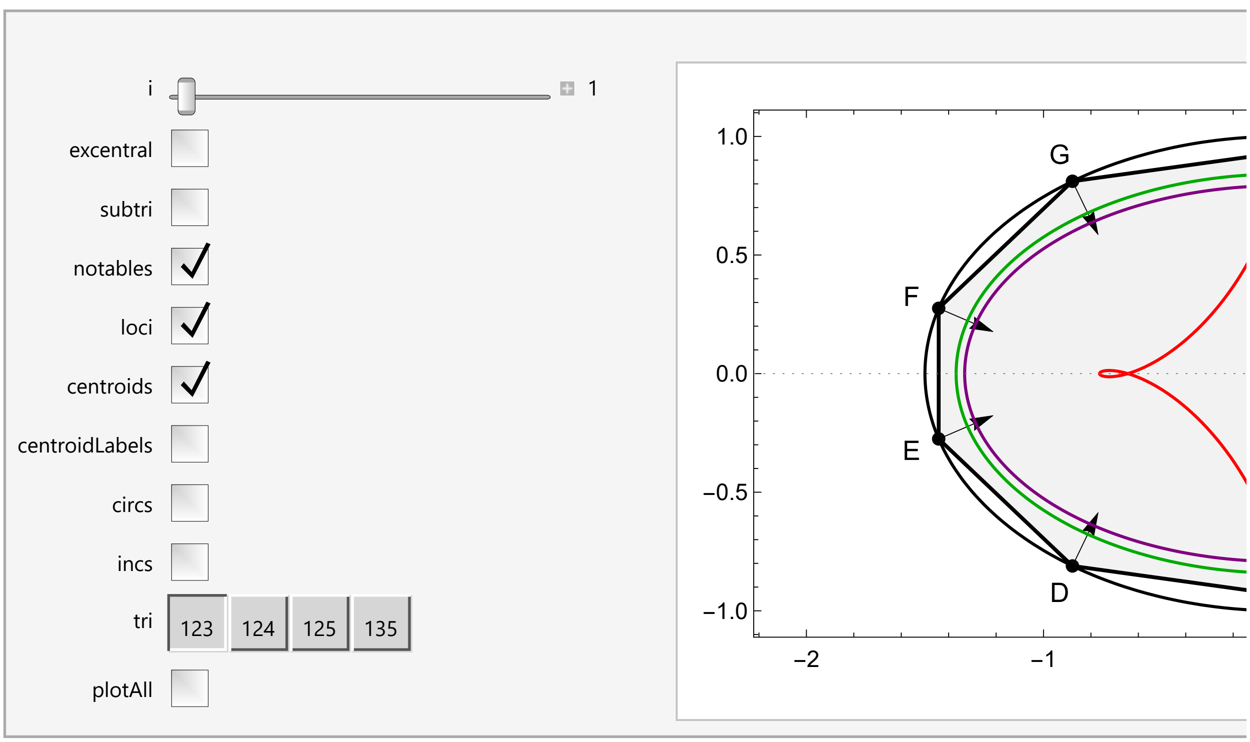
In[514]:=
```
Clear@nonaAlphaT15;
nonaAlphaT15 = calcAlphaCausticT[False(*False for load*),
   nonaErrCaustic, "data/nonaAlphaCausticT_a15.m", 1.5, 1];
```

```
loaded: 360 records fromdata/nonaAlphaCausticT_a15.m
```

In[516]:=
```
nonaVtxList = {{1, 2, 3}, {1, 2, 4}, {1, 2, 5}, {1, 3, 5}};
nonaNotableLocis =
  getPolyNotableLoci[nonaAlphaT15, getNonaVtx0, #, {"bar", "inc", "cir", "ort"}] & /@
   nonaVtxList;
```

In[518]:=
```
manipulatePolyVtx[nonaAlphaT15,
 nonaNotableLocis, getNonaVtx0, nonaErrorP, nonaVtxList]
```

Out[518]=



Nonagon: Centroid Invariance

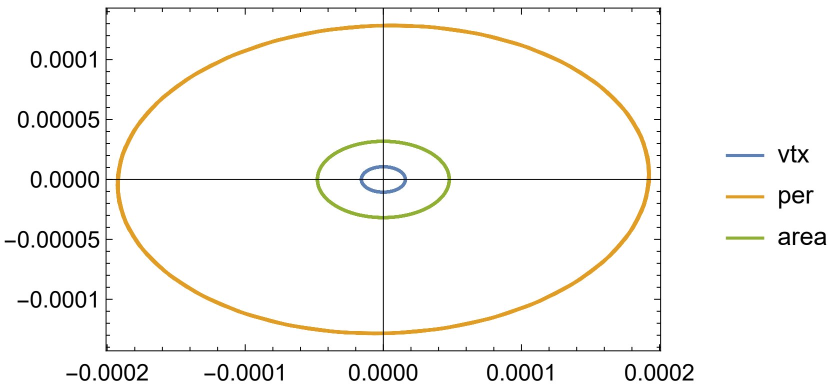
In[519]:=
```
getCentroidRadialStatsTable[nonaAlphaT15, getNonaVtx0]
```

Out[519]=

type	mean	sd	zscore
vtx	0.0000134198	1.87451×10^{-6}	0.139682
perimeter	0.000162304	0.0000226709	0.139682
area	0.0000402595	5.62354×10^{-6}	0.139682

In[520]:=
```
showCentroidPaths[nonaAlphaT15, getNonaVtx0]
```

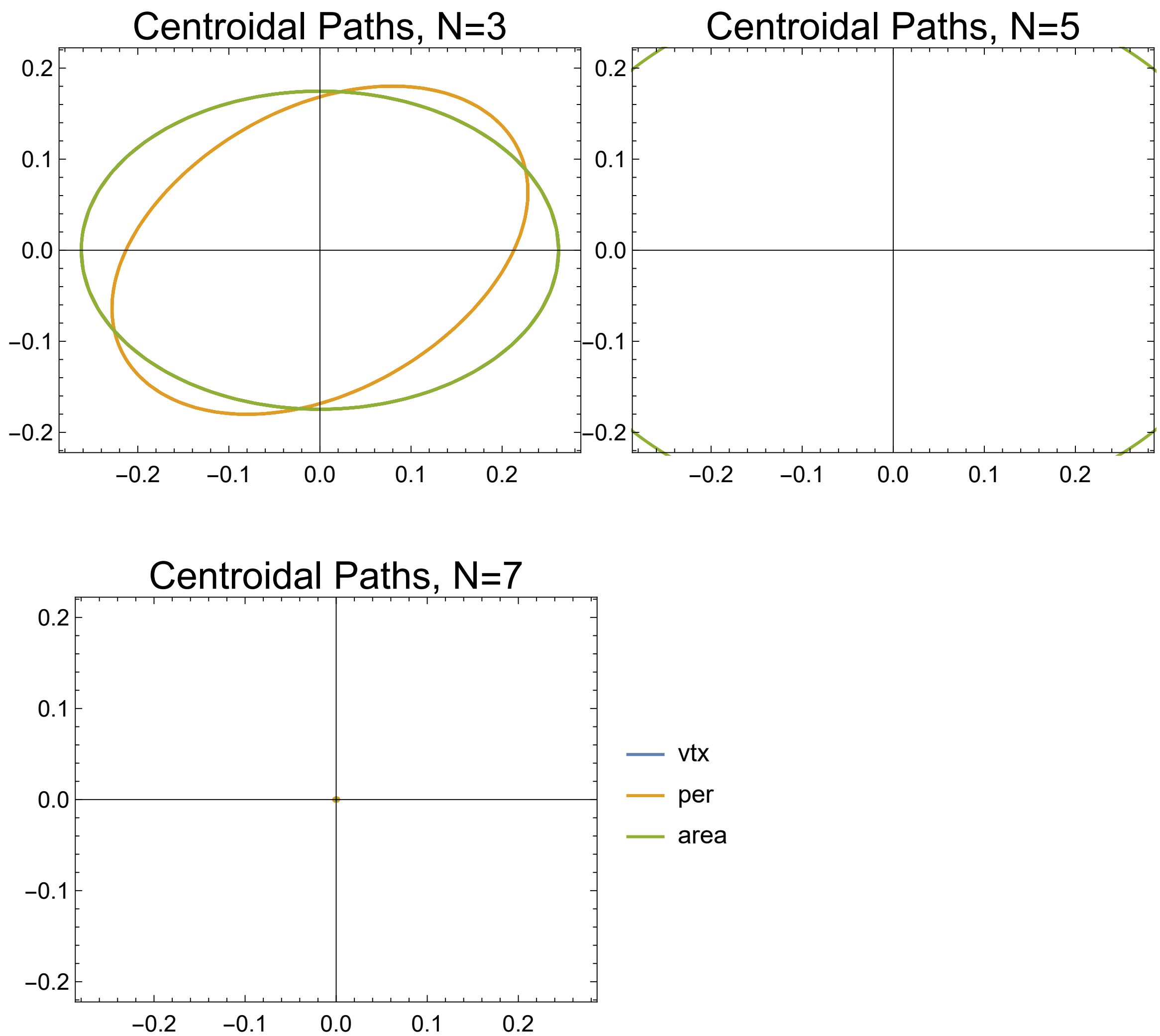
Out[520]=



Centroidal Loci N=3,5,7,9

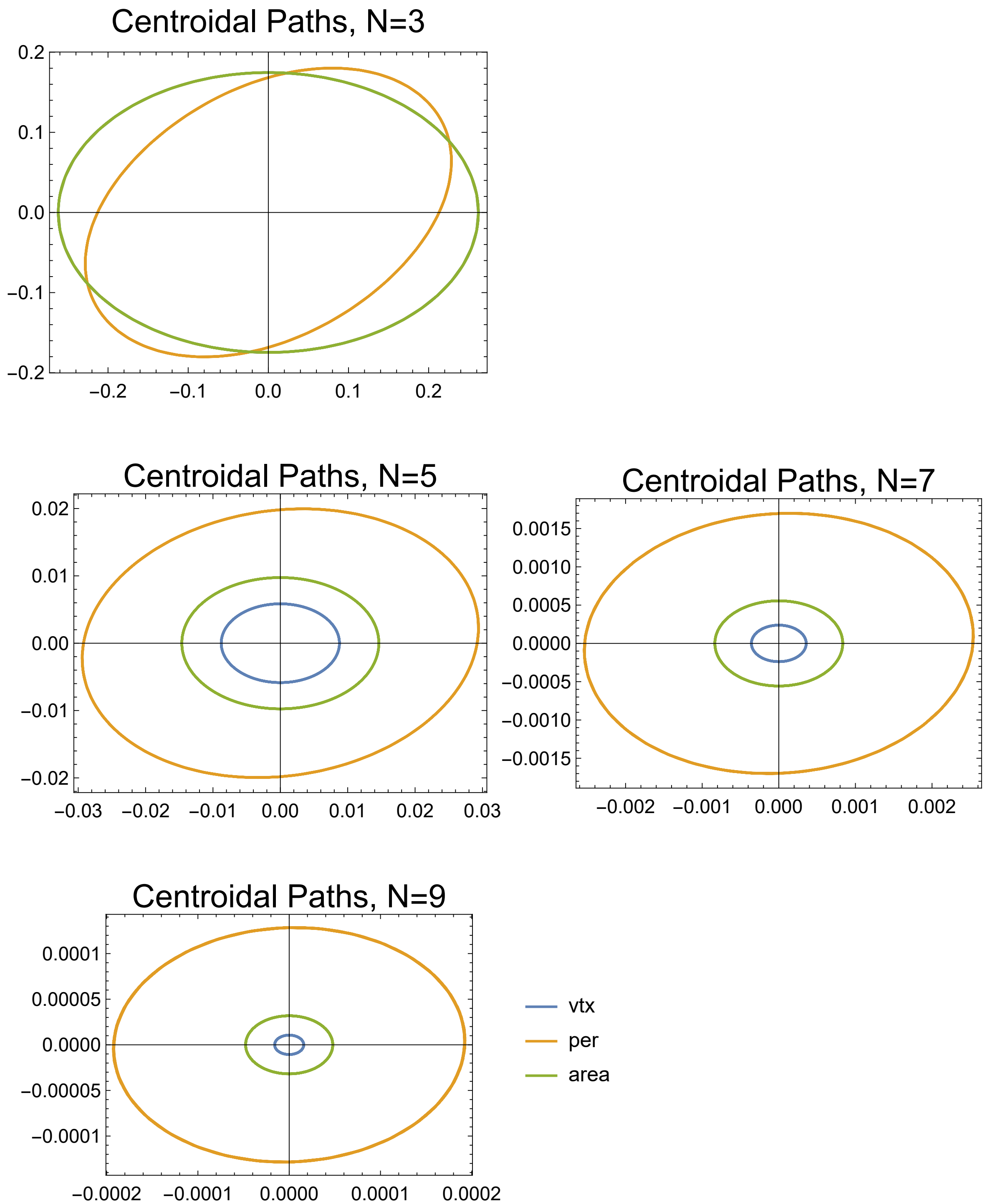
In[521]:=
```
Row[MapThread[Show[showCentroidPaths[#1, #2, drLegend → #4],
    ImageSize → 300, PlotLabel → Style["Centroidal Paths, " <> #3, {Black, 20}],
    PlotRange → {{-.275, .275}, {-.2, .2}}, AspectRatio → Automatic] &,
  {{triAlphaT15, heptAlphaT15, heptAlphaT15}, {getTriVtx0, getPentVtx0,
    getHeptVtx0}, {"N=3", "N=5", "N=7"}, {False, False, True}}]]
```

Out[521]=



In[522]:=
```
Row[MapThread[Show[showCentroidPaths[#1, #2, drLegend → #4],
    ImageSize → 300, PlotLabel → Style["Centroidal Paths, " <> #3, {Black, 20}],
    PlotRange → All, AspectRatio → Automatic] &,
  {{triAlphaT15, pentAlphaT15, heptAlphaT15, nonaAlphaT15},
   {getTriVtx0, getPentVtx0, getHeptVtx0, getNonaVtx0},
   {"N=3", "N=5", "N=7", "N=9"}, {False, False, False, True}}]]
```

Out[522]=


Centroidal Paths, N=3
0.2
0.1
0.0
−0.1
−0.2
−0.2
−0.1
0.0
0.1
0.2
Centroidal Paths, N=5
0.02
0.01
0.00
−0.01
−0.02
−0.03
−0.02
−0.01
0.00
0.01
0.02
0.03
Centroidal Paths, N=7
0.0015
0.0010
0.0005
0.0000
−0.0005
−0.0010
−0.0015
−0.002
−0.001
0.000
0.001
0.002
Centroidal Paths, N=9
0.0001
0.00005
0.0000
−0.00005
−0.0001
−0.0002
−0.0001
0.0000
0.0001
0.0002
vtx
per
area

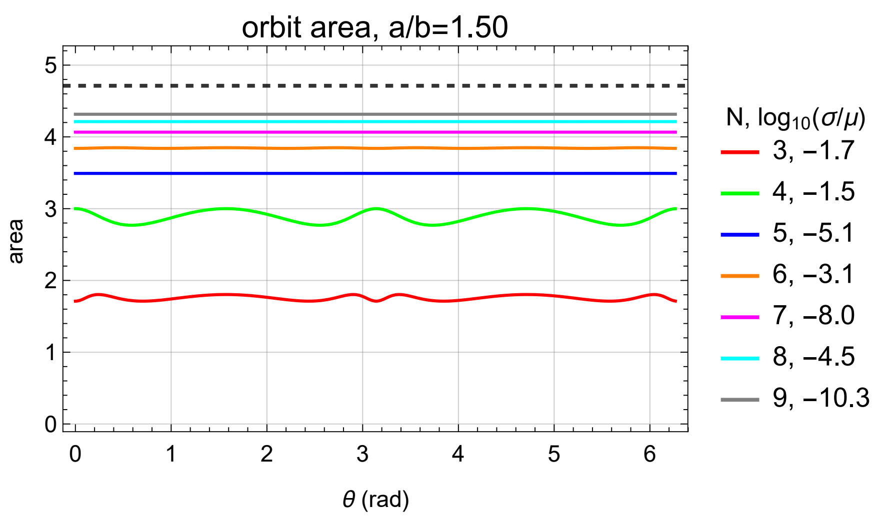
Areas: N=3,4,5,6,7,8,9

In[523]:=
```
getTriAreas[a_] := Module[{ts, tris},
   ts = Table[toRad@t, {t, 0, 359, 1}];
   tris = orbitNormals[a, #][[1]] & /@ ts;
   MapThread[{#1, Area[Polygon[#2]]} &, {ts, tris}]];
```

In[524]:=
```
getPolyMeanAreas[a_] := Module[{triA, otherAs, areas, means, sds, zs},
   triA = getTriAreas[a];
   otherAs = MapThread[getPolyAreas[#1, #2] &,
     Transpose@{
       {quadAlphaT15, getQuadVtx0},
       {pentAlphaT15, getPentVtx0},
       {hexAlphaT15, getHexVtx0},
       {heptAlphaT15, getHeptVtx0},
       {octAlphaT15, getOctVtx0},
       {nonaAlphaT15, getNonaVtx0}
      }];
   areas = {triA, Sequence @@ otherAs};
   means = Mean /@ ((Second /@ #) & /@ areas);
   sds = StandardDeviation /@ ((Second /@ #) & /@ areas);
   zs = MapThread[safeDiv[#1, #2] &, {sds, means}];
   {areas, means, sds, zs}
  ];
```

In[525]:=
```
Module[{a = 1.5, areas, means, sds, zs, labs,
   clrs = {Red, Green, Blue, Orange, Magenta, Cyan, Gray}},
  {areas, means, sds, zs} = getPolyMeanAreas[a];
  (*Print[means,sds];*)
  labs = MapThread[(ToString[#1] <> ",  " <> nfn[Log10[#2], 1]) &,
    {Range[3, 2 + Length@means], zs}];
  Legended[Show[MapThread[plotPolyAreas[First /@ #1, Second /@ #1, #2] &,
      {areas, clrs}],
    PlotRange → {All, {0, 5}}, FrameStyle → Medium,
    GridLines → {Automatic,
      {0, 1, 2, 3, 4, 5, {π * a, Directive[Black, Dashed, Thick, Opacity@.8]}}},
    FrameLabel -> {"θ  (rad)", "area"}, PlotLabel →
     Style["orbit area, a/b=" <> nfn[a, 2], {Black, 16}]],
   LineLegend[Directive[Thick, #] & /@ clrs, Style[#, 14] & /@ labs,
    LegendLabel → "N, log10 (σ/μ)"]]]
```

Out[525]=

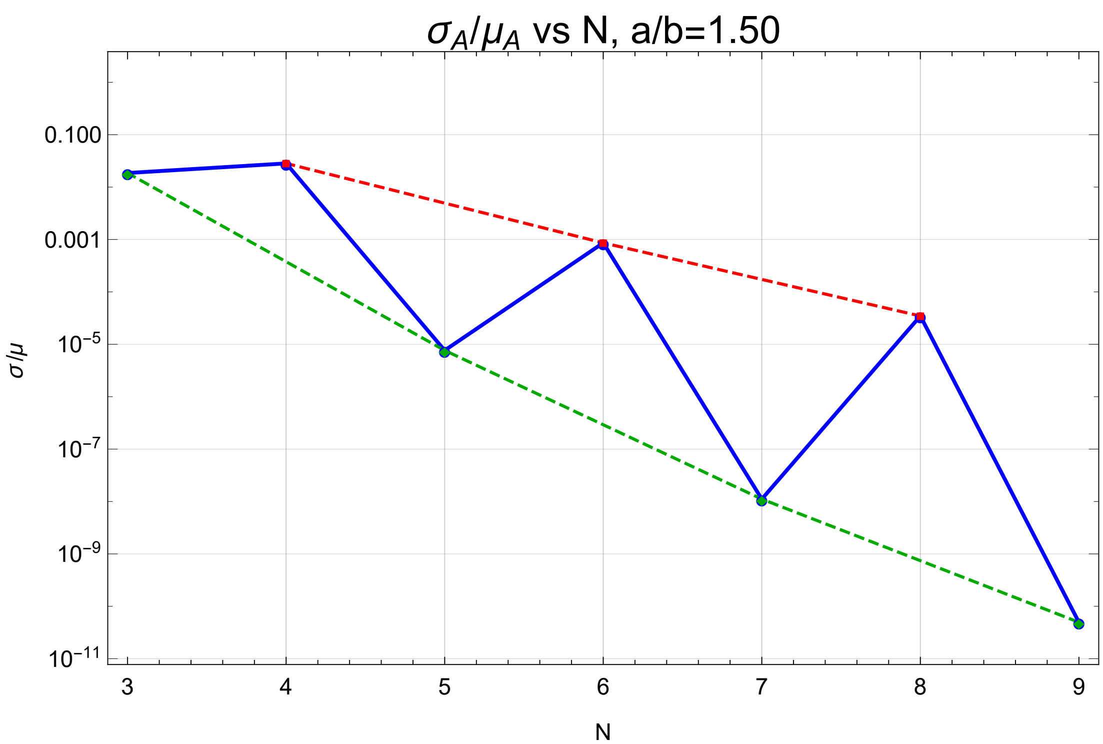


SLOW!

In[529]:=
```
zs15 = getPolyMeanAreas[1.5][[4]];
```

In[530]:=
```
Module[{a = 1.5, pts},
 pts = Transpose@{Range[Length@zs15] + 2, zs15};
 ListLogPlot[{pts, Part[pts, {2, 4, 6}], Part[pts, {1, 3, 5, 7}]},
  PlotMarkers → {Automatic, Small},
  PlotStyle → {{Thick, Blue}, {Dashed, Red}, {Dashed, Darker@Green}},
  Frame → True, FrameLabel → {"N", "σ/μ"},
  FrameStyle → Directive[Black, Medium], Joined → True, GridLines → Automatic,
  PlotLabel → Style["σA/μA vs N, a/b=" <> nfn[a, 2], {20, Black}], ImageSize → Large]]
```

Out[530]=



Elliptic Perimeter

The arc length of the ellipse is

$$\begin{aligned} s(t) &= a\,E(t, e) \\ &= a\,E\left(t, \sqrt{1-\frac{b^2}{a^2}}\right) \\ &= b\,E\left(t, \sqrt{1-\frac{a^2}{b^2}}\right), \end{aligned}$$

where $E(t, e)$ is an incomplete elliptic integral of the second kind with elliptic modulus e (the eccentricity).

In[546]:=
```
getEllArc[a_, t_] := Module[{ecc2 = 1 - 1/a^2},
   a EllipticE[t, ecc2]];
```

```
getEllArc[a_, t1_, t2_] := Module[{ecc2 = 1 - 1/a^2},
  a (EllipticE[t2, ecc2] - EllipticE[t1, ecc2])]
```

$$p \approx \pi \left[3(a+b) - \sqrt{(3a+b)(a+3b)}\right]$$

```
In[543]:= ellPerRamanujan1[a_, b_] := π (3 * (a + b) - Sqrt[(3 a + b) (a + 3 b)]);

In[544]:= ellPerRamanujan1[1.5, 1]

Out[544]= 7.93272

In[545]:= getEllArc[1.5, 2 π]

Out[545]= 7.93272

In[584]:= getEllArc[1.5, 2 π - π/10, π/10]

Out[584]= -6.99876

In[531]:= Clear@ellV; ellV[a_, t_] = Module[{v},
  v = D[{a Cos[t], Sin[t]}, t];
  Sqrt[v.v]]
```

Out[531]= $\sqrt{\text{Cos}[t]^2 + a^2\ \text{Sin}[t]^2}$

Tan[t]/a == y0/x0 = > t = ArcTan[a y0/x0]

a Cos[t1] == x1, a Cos[t2] == x2, a Cos[(t1 + t2)/2] == a (Cos[t1/2] Cos[t2/2] - Sin[t1/2] Sin[t2/2])

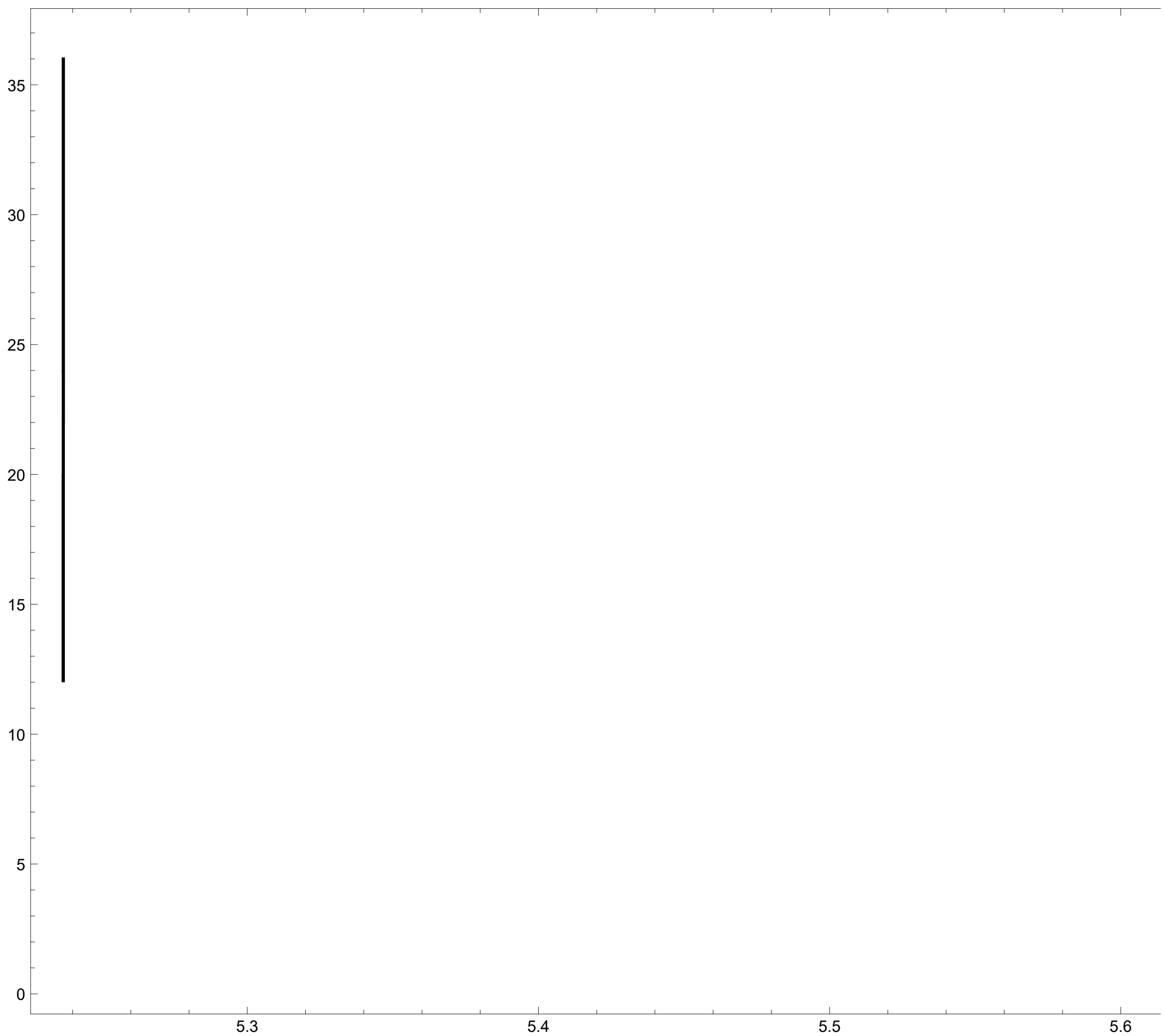
```
In[532]:= Clear@solEllT; solEllT[a_, {x0_, y0_}] := ArcTan[x0, a y0];

In[533]:= Clear@getEllMidPoints;
getEllMidPoints[a_, ps_] := Module[{mids, midsInter, t1, t2, tavgs, midsT},
  (*mids=MapThread[(#1+#2)/2&,{ps,RotateLeft@ps}];
  midsInter=ellInterRayUnprot[a,{0,0},#][[2]]&/@mids;*)
  tavgs = MapThread[(t1 = solEllT[a, #1];
      t2 = solEllT[a, #2];
      (t1 + t2) / 2) &, {Most@ps, Rest@ps}];
  midsT = {a Cos@#, Sin@#} & /@ tavgs]

In[623]:= getDistStats[ps_] := getStats@MapThread[magn[#1 - #2] &, {ps, RotateLeft@ps}];
getDist2Stats[ps_] := getStats@MapThread[magn2[#1 - #2] &, {ps, RotateLeft@ps}];
getEllArcStats[a_, ts_] := getStats@MapThread[
    Module[{t1, t2},
      t1 = #1; t2 = #2;
      (* avoid negative values *)If[t2 < t1, t2 += 2 π];
      getEllArc[a, t1, t2]^2] &, {ts, RotateLeft@ts}];

In[626]:= getEllArc[1.5, 2 π - .1, 2 π + .1]

Out[626]= 0.299723
```

```
In[627]:= getPerimeter[a_, ts_] := Module[{ps},
    ps = {a Cos@#, Sin@#} & /@ ts;
    Total@MapThread[magn[#1 - #2] &, {ps, RotateLeft@ps}]];

In[614]:= getDistZ[a_, ts_] := Module[{ps},
    ps = {a Cos@#, Sin@#} & /@ ts;
    getDistStats[ps][[3]]];

In[615]:= getDistZ2[a_, ts_] := Module[{ps},
    ps = {a Cos@#, Sin@#} & /@ ts;
    getDist2Stats[ps][[3]]];

In[616]:= Clear@getEllArcZ;
getEllArcZ[a_, ts_] := getEllArcStats[a, ts][[3]];

In[618]:= labStr[ps_, ds_] := nfn[ds, 5] <> "  (N=" <> ToString@Length@ps <> ")"

In[628]:= getEvenEllipseSampling[a_, steps_, zfn_: getDistZ] :=
  Module[{ps, psZ, ts, tsyms, minSol, tsMin, psMin, minZ},
   ts = Range[0, 2 π - π / steps, 2 π / steps];
   ps = {1.5 Cos@#, Sin@#} & /@ ts;
   psZ = getDistStats[ps][[3]]; (* zscore *)
   (* this one works best *)
   tsyms = Table[Symbol["t" <> ToString[i]], {i, Length@ps}];
   minSol = Quiet@FindMinimum[zfn[a, tsyms],
      MapThread[{#1, #2} &, {tsyms, ts}]];
   tsMin = tsyms /. (minSol[[2]]);
   psMin = {a Cos@#, Sin@#} & /@ tsMin;
   minZ = getDist2Stats[psMin][[3]];
   {
    "ts" → ts;
    "ps" → ps,
    "psZ" → psZ,
    "tsMin" → tsMin,
    "err" → First@minSol,
    "psMin" → psMin,
    "minZ" → minZ}];

If[False,
  evenEllSamples = getEvenEllipseSampling[1.5, 3599.(*,getEllArcZ*)];
  Save["evenEllSamples.m", evenEllSamples],
  evenEllSamples = Get["evenEllSamples.m"]];

Out[702]= $Aborted
```

```
In[816]:= getBinCounts[vals_, binN_] := Module[{g, bs, cs},
   {g, {{cs}}} = Reap[Histogram[vals, binN, Function[{bins, counts}, Sow[counts]]]];
   {g, {{bs}}} = Reap[Histogram[vals, binN, Function[{bins, counts}, Sow[bins]]]];
   MapThread[{#1[[1]] + #1[[2]] / 2, #2} &, {bs, cs}]];

In[817]:= getBinCounts[Range[1, 100, 1], 10]

Out[817]= {{5, 9}, {20, 10}, {35, 10}, {50, 10}, {65, 10},
  {80, 10}, {95, 10}, {110, 10}, {125, 10}, {140, 10}, {155, 1}}

In[827]:= Clear@getAreaDistrComb;
getAreaDistrComb[a_, ns_, step_: 1.] := Module[{ts, polys, areas, stats, hs, clrs},
   (*ts=toDeg["tsMin"/.evenEllSamples];*)
   ts = Range[0, 360 - step, step];
   polys = getCausticOrbits[a, #, ts] & /@ ns;
   areas = Table[Area /@ Polygon /@ polys[[i]], {i, Length@polys}];
   clrs = Take[ColorData[3, "ColorList"], Length@areas];
   (*MapThread[Histogram[#1,20,"Probability",ChartStyle→#2]&,{areas,clrs}]*)
   Show[MapThread[ListLinePlot[getBinCounts[#1, 20], PlotStyle → #2] &,
     {areas, clrs}], Frame → True, PlotRange → All]
  ];
```

In[832]:=
```
getAreaDistrComb[1.5, Range[5, 6, 1], 1]
```

Out[832]=

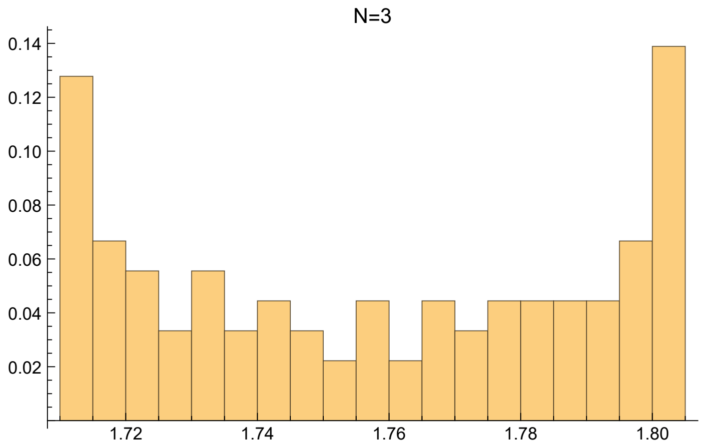


In[726]:=
```
Clear@getAreaDistr;
getAreaDistr[a_, n_, step_: 1.] := Module[{ts, polys, areas, stats},
  (*ts=toDeg["tsMin"/.evenEllSamples];*)
  ts = Range[0, 360 - step, step];
  polys = getCausticOrbits[a, n, ts];
  areas = Area /@ Polygon /@ polys;
  {Histogram[areas, 20, "Probability", ImageSize → Medium,
    PlotLabel → "N=" <> ToString@n], getStatsLabeled[areas]}]
```

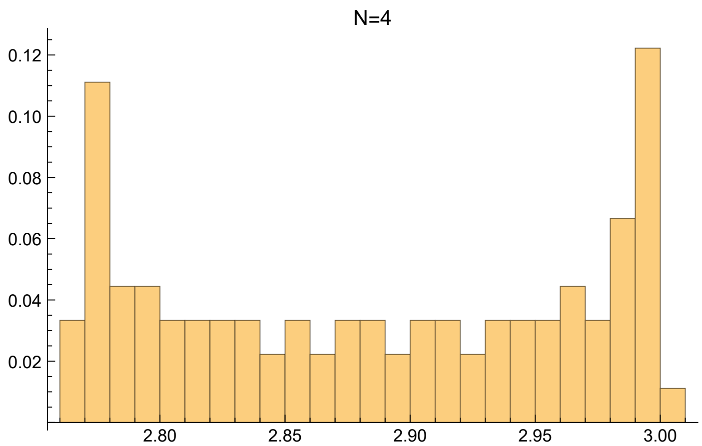
In[689]:=
```
zs15
```

Out[689]= $\{0.0185589, 0.0283081, 7.64447\times10^{-6}, 0.000862265, 1.11151\times10^{-8}, 0.0000346931, 4.95942\times10^{-11}\}$

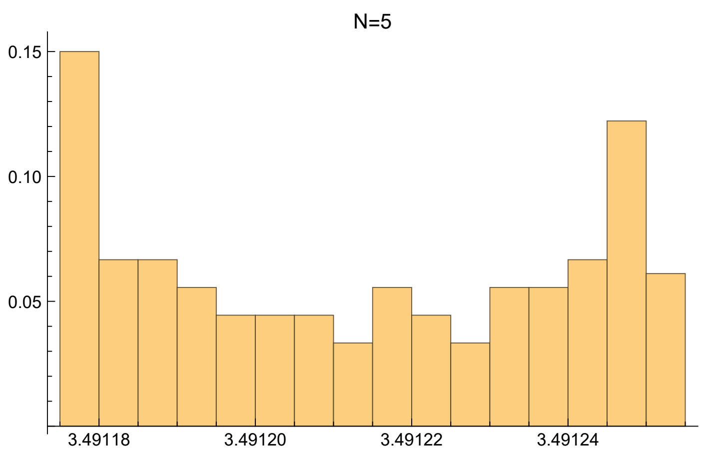
In[736]:=
```
Grid[getAreaDistr[1.5, #, 1] & /@ Range[3, 9, 1], Alignment → Left]
```




```
{mean → 1.75782, sd → 0.0326232,
 zscore → 0.0185589, min → 1.71144,
 max → 1.80359, median → 1.75799, N → 3
```

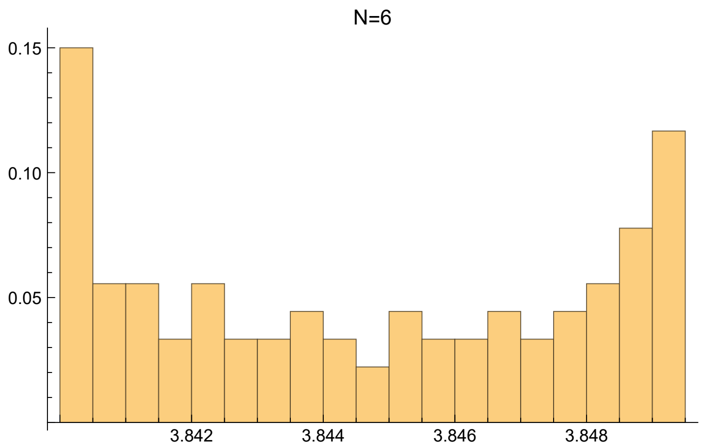



```
{mean → 2.88577, sd → 0.0816905,
 zscore → 0.0283081, min → 2.76926,
 max → 3., median → 2.88623, N → 360}
```

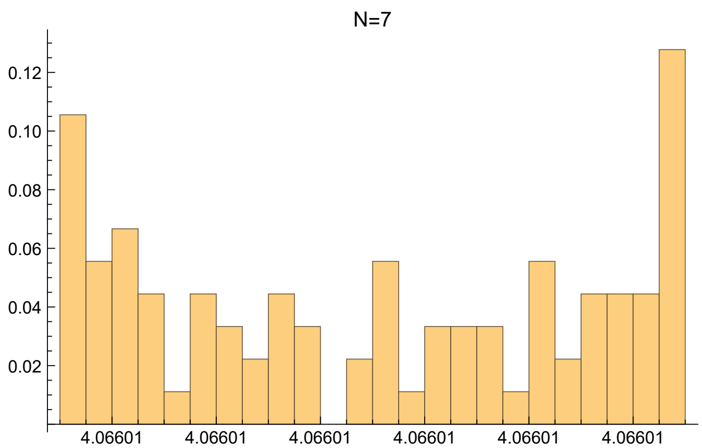


{mean → 3.49121, sd → 0.0000266885,
 zscore → 7.64447×10^{-6}, min → 3.49118,
 max → 3.49125, median → 3.49121, N → 3

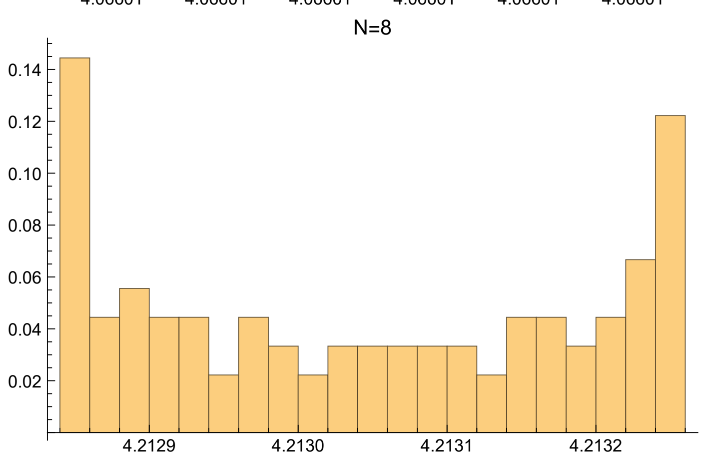
Out[736]=



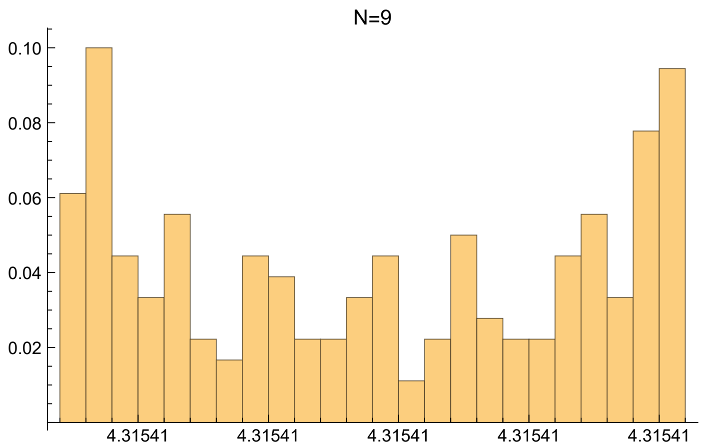
{mean → 3.84468, sd → 0.00331513,
zscore → 0.000862265, min → 3.84,
max → 3.84936, median → 3.84471, N → 3



{mean → 4.06601, sd → 4.14383×10^{-8},
zscore → 1.01914×10^{-8}, min → 4.06601,
max → 4.06601, median → 4.06601, N → 3



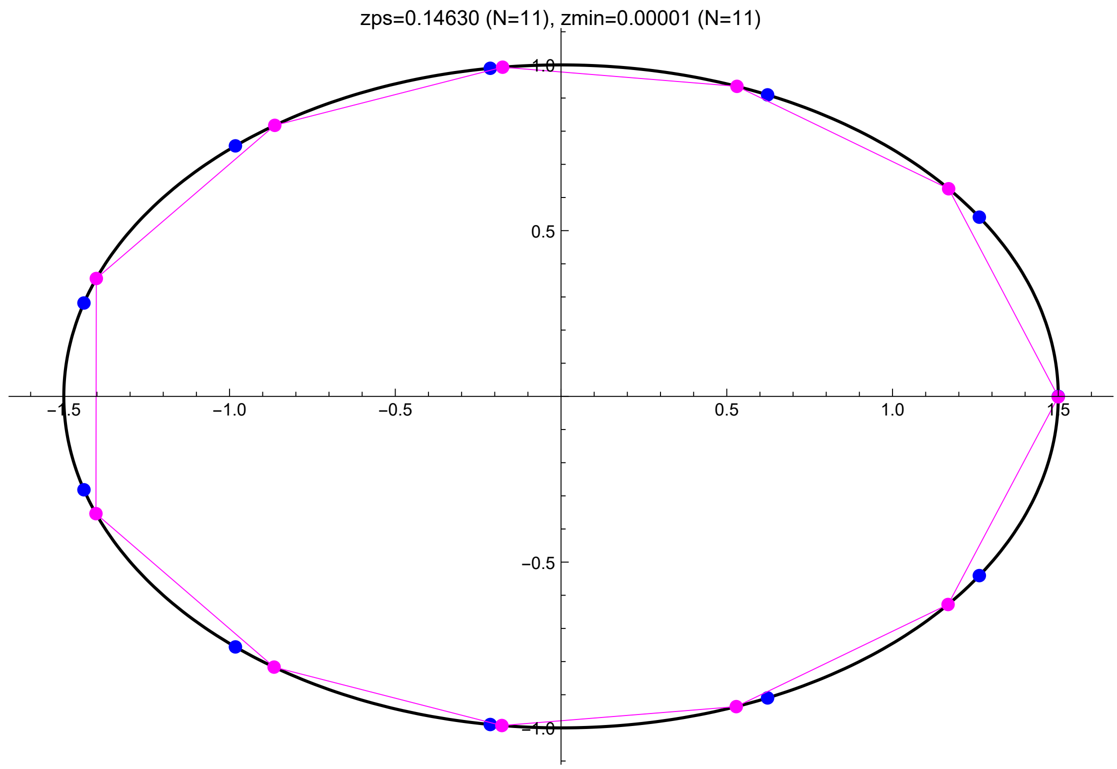
{mean → 4.21305, sd → 0.000146163,
zscore → 0.0000346931, min → 4.21284,
max → 4.21325, median → 4.21304, N → 3



{mean → 4.31541, sd → 8.00186 × 10^{-11},
zscore → 1.85425 × 10^{-11}, min → 4.31541
max → 4.31541, median → 4.31541, N → 3

In[632]:=
```
Module[{a = 1.5, steps = 11, ell, evens, ps, psZ, psMin, minZ, gr, lab},
 ell = plotEll[a];
 evens = getEvenEllipseSampling[a, steps(*,getEllArcZ*)];
 ps = "ps" /. evens;
 psZ = "psZ" /. evens;
 psMin = "psMin" /. evens;
 minZ = "minZ" /. evens;
 lab = "zps=" <> labStr[ps, psZ] <> ", zmin=" <> labStr[psMin, minZ];
 gr = Graphics[{PointSize@Large,
    {Blue, Point@ps},
    (*{Red,Point[{0,.0}+#&/@mids]},
    {Darker@Green,Point@psMax,Line@psMax},*)
    {Magenta, Point@psMin, FaceForm@None, EdgeForm@Magenta, Polygon@psMin}}];
 Show[{ell, gr}, PlotLabel → lab, ImageSize → Large]]
```

Out[632]=




```
Module[{a = 1.5, areas, means, sds, zs, labs,
   clrs = {Red, Green, Blue, Orange, Magenta, Cyan, Gray}},
  {areas, means, sds, zs} = getPolyMeanAreas[a];
  (*Print[means,sds];*)
  labs = MapThread[(ToString[#1] <> ",  " <> nfn[Log10[#2], 1]) &,
    {Range[3, 2 + Length@means], zs}];
  Legended[Show[MapThread[plotPolyAreas[First /@ #1, Second /@ #1, #2] &,
      {areas, clrs}],
    PlotRange → {All, {0, 5}}, FrameStyle → Medium,
    GridLines → {Automatic,
      {0, 1, 2, 3, 4, 5, {π * a, Directive[Black, Dashed, Thick, Opacity@.8]}}},
    FrameLabel -> {"θ (rad)", "area"}, PlotLabel →
     Style["orbit area, a/b=" <> nfn[a, 2], {Black, 16}]],
   LineLegend[Directive[Thick, #] & /@ clrs, Style[#, 14] & /@ labs,
    LegendLabel → "N, log10(σ/μ)"]]]
```

Sum of Cosines: Regular Polygon

In[870]:=
```
sumIntAng[n_] := (n - 2) * π;
regularAngle[n_] := sumIntAng[n] / n;
sumCosRegAng[n_] := n * Cos[regularAngle[n]];
(* passes thru (3,1.5), and (4,0) *)
graphRegAngSum[n_] := 6 - (3/2) n;
```

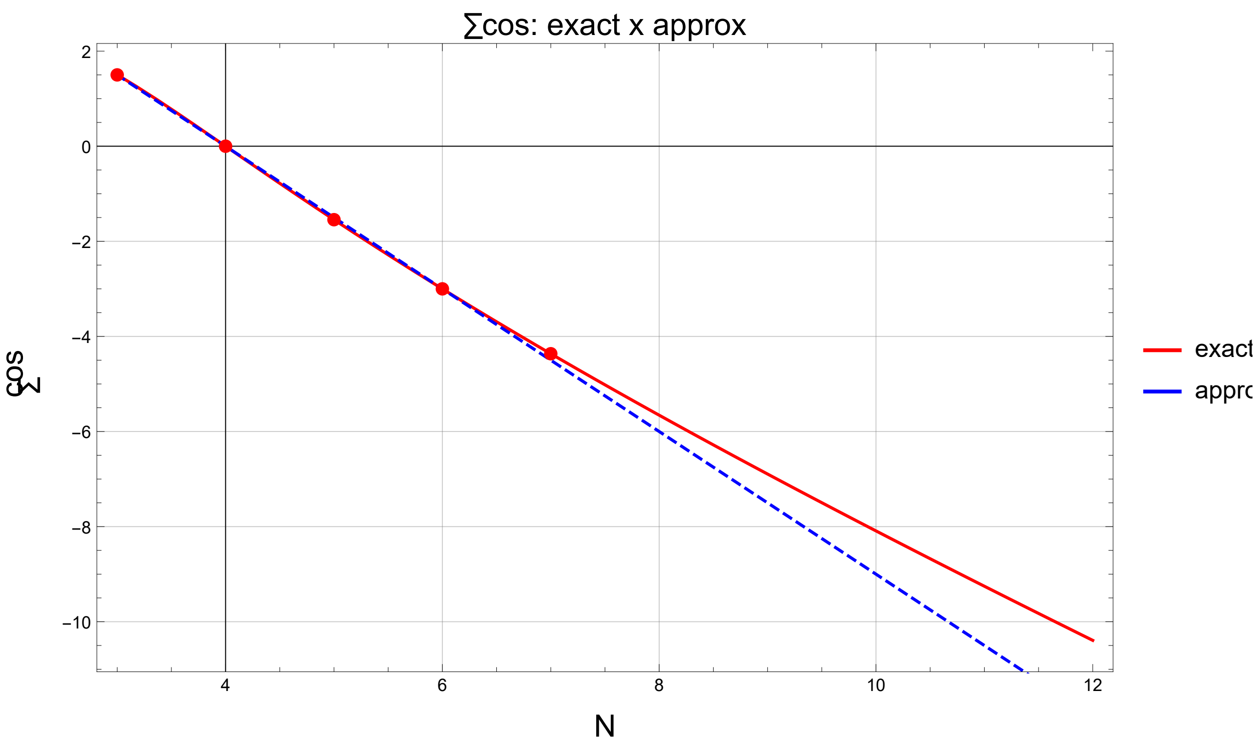
In[874]:=
```
Clear@sumRegSimple; sumRegSimple[n_] = FullSimplify[sumCosRegAng[n]]
```

Out[874]= $-n \operatorname{Cos}\left[\frac{2\pi}{n}\right]$

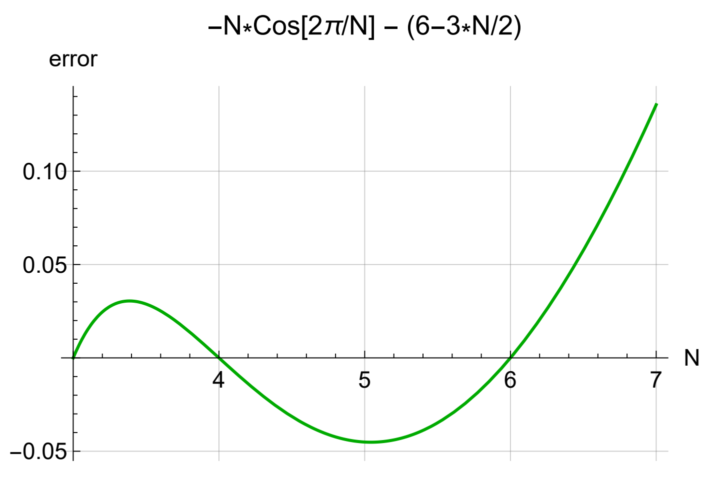
```
Module[{sumCos},
 sumCos = N /@ Table[{i, sumCosRegAng[i]}, {i, 3, 12, 1}];
 sumCos // Prepend[#, {"N", "Σcos"}] & // Grid[#, Frame → All, Alignment → Left] &
]
```

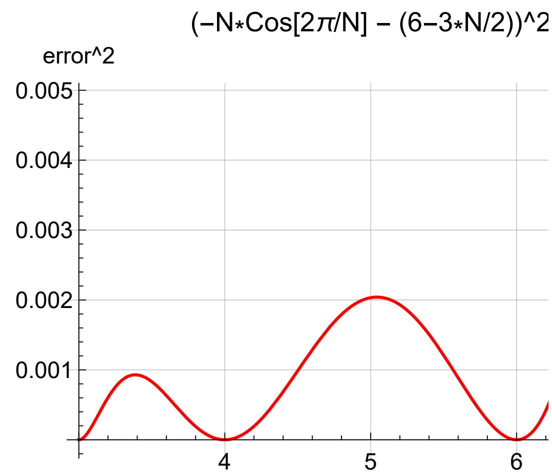
N	Σcos
3.	1.5
4.	0.
5.	-1.54508
6.	-3.
7.	-4.36443
8.	-5.65685
9.	-6.8944
10.	-8.09017
11.	-9.25379
12.	-10.3923

```
Module[{sumCos, ll, pl, pl2, clrs = {Red, Blue}, ps},
 ps = Table[{i, sumCosRegAng[i]}, {i, 3, 7}];
 sumCos = N /@ Table[{i, sumCosRegAng[i]}, {i, 3, 12, 1}];
 pl = Plot[sumCosRegAng[x], {x, 3, 12}, PlotStyle → (clrs[[1]]),
   Epilog → {PointSize@Large, Red, Point@ps}];
 pl2 = Plot[graphRegAngSum[x], {x, 3, 12}, PlotStyle → {Dashed, clrs[[2]]}];
 Legended[Show[{pl, pl2},
   GridLines → Automatic, ImageSize → Large,
   Frame → True, FrameLabel → (Style[#, {Black, 16}] & /@ {"N", "Σcos"}),
   PlotLabel → Style["∑cos: exact x approx", {Black, 16}]],
  LineLegend[Directive[Thick, #] & /@ clrs,
   {"exact[N]  =  -N*Cos[2π/N]", "approx[N]  =  6-(3/2)*N"}]]]
```



```
Module[{plErr, plErr2},
 plErr = Plot[sumCosRegAng[x] - graphRegAngSum[x],
   {x, 3, 7}, PlotStyle → Darker@Green, Axes → True, AxesStyle → Medium,
   AxesLabel → {"N", "error"}, GridLines → Automatic, ImageSize → Medium,
   PlotLabel → Style["-N*Cos[2π/N]  -  (6-3*N/2)", {Black, 14}]];
 plErr2 = Plot[(sumCosRegAng[x] - graphRegAngSum[x])^2, {x, 3, 7},
   PlotStyle → Red, Axes → True, AxesStyle → Medium,
   AxesLabel → {"N", "error^2"}, GridLines → Automatic, ImageSize → Medium,
   PlotLabel → Style["(-N*Cos[2π/N]  -  (6-3*N/2))^2", {Black, 14}]];
 Grid[{{plErr, plErr2}}]]
```





Sums of Cosines: N=3,4,5,6,7,8,9

Sum of Cosines of Internal Angles: CONSTANT

```
In[875]:= getMeanSumCos[alphaT_, fnVtx0_] := Module[{sumCosT, mu, sd, z},
   sumCosT =
    Table[sumPolyCosines[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}];
   mu = Mean@sumCosT;
   sd = StandardDeviation@sumCosT;
   z = Abs@safeDiv[sd, mu];
   {mu, sd, z}];

In[876]:= allSumCos15 = MapThread[{#1, Sequence @@ getMeanSumCos[#2, #3]} &,
   {Range[3, 9],
    {triAlphaT15, quadAlphaT15, pentAlphaT15,
     hexAlphaT15, heptAlphaT15, octAlphaT15, nonaAlphaT15},
    {getTriVtx0, getQuadVtx0, getPentVtx0, getHexVtx0,
     getHeptVtx0, getOctVtx0, getNonaVtx0}}];
```

In[877]:=
```
allSumCos15 //
  Prepend[#, {"N", "μ(∑cos)", "σ(∑cos)", "z=σ/μ"}] & // Grid[#, Frame → All] &
```

Out[877]=

N	μ(∑cos)	σ(∑cos)	z=σ/μ
3	1.36266	3.08113×10^{-15}	2.26112×10^{-15}
4	-1.42418×10^{-16}	1.11661×10^{-15}	7.84032
5	-1.51287	8.93732×10^{-11}	5.90753×10^{-11}
6	-2.96	3.55601×10^{-8}	1.20136×10^{-8}
7	-4.3237	1.17056×10^{-8}	2.70731×10^{-9}
8	-5.61767	1.60936×10^{-9}	2.86481×10^{-10}
9	-6.85745	2.328×10^{-10}	3.39485×10^{-11}

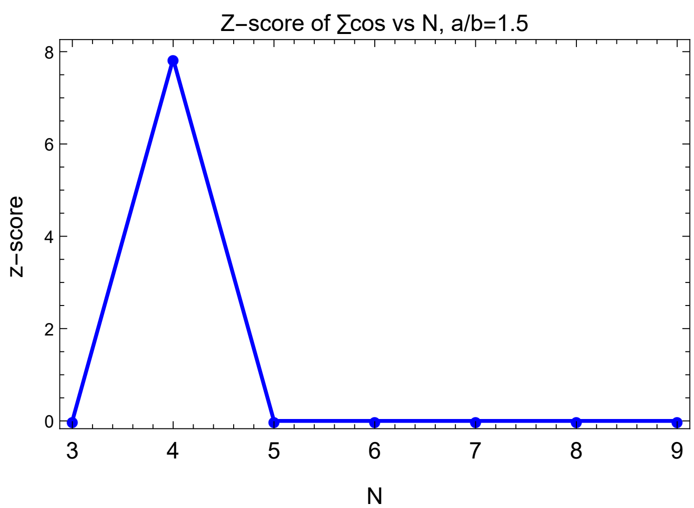
Plot Z-Scores. Is this sum truly constant?

In[878]:=
```
Part[#, {1, 4}] & /@ allSumCos15
```

Out[878]= $\{\{3, 2.26112\times10^{-15}\}, \{4, 7.84032\}, \{5, 5.90753\times10^{-11}\}, \{6, 1.20136\times10^{-8}\}, \{7, 2.70731\times10^{-9}\}, \{8, 2.86481\times10^{-10}\}, \{9, 3.39485\times10^{-11}\}\}$

In[879]:=
```
ListPlot[Part[#, {1, 4}] & /@ allSumCos15, PlotMarkers → {Automatic, Small},
 Joined → True, PlotStyle → {Thick, Blue}, Frame → True,
 FrameLabel → (Style[#, {Black, Medium}] & /@ {"N", "z-score"}), PlotRange → All,
 PlotLabel → Style["Z-score of ∑cos vs N, a/b=1.5", {Black, Medium}],
 FrameStyle → {Black, Medium}]
```

Out[879]=

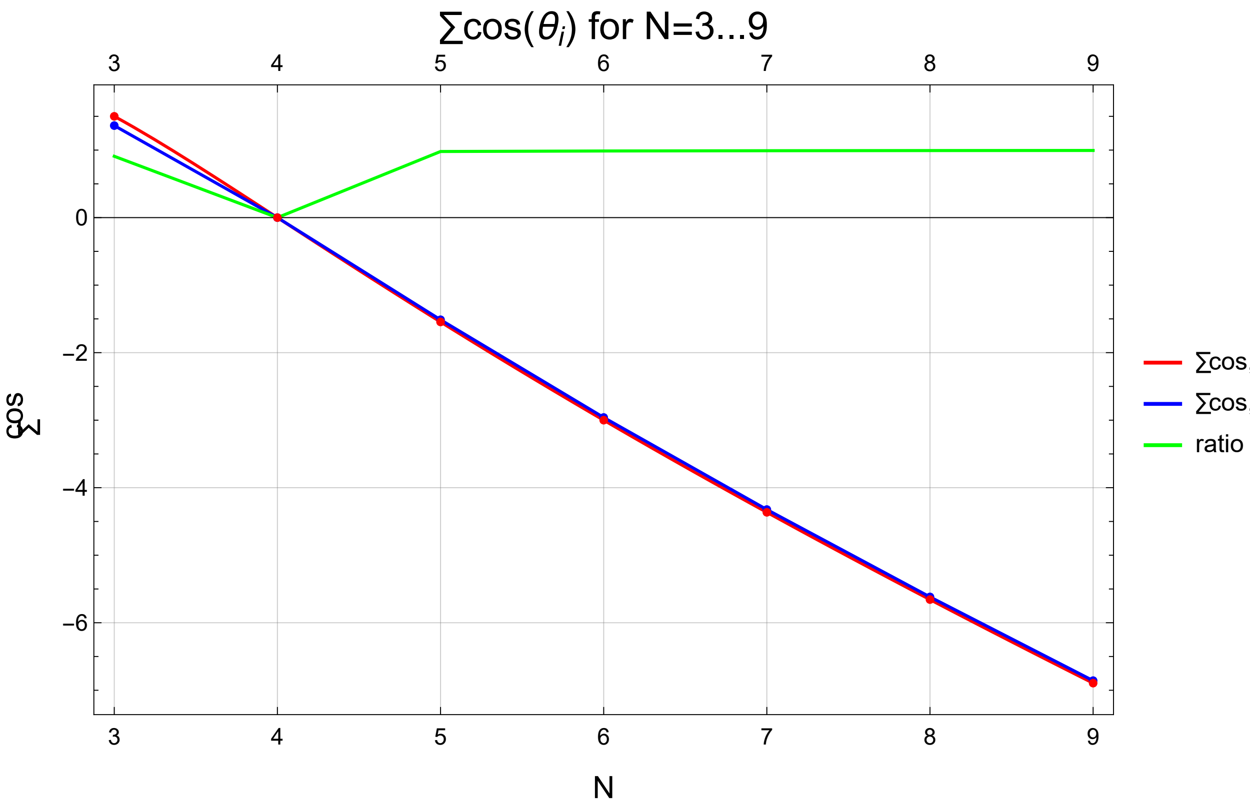


Cosine Sum vs N

***Jair : outros a/b

In[880]:=
```
sumCos15 = Module[{pts10, pts15, ptsRatio,
   pl10, pl15, gr15, plRatio, nmax = 9,
   clrs = {Red, Blue, Green}},
  pts10 = Table[{i, sumCosRegAng[i]}, {i, 3, nmax}];
  pl10 = Plot[sumCosRegAng[x], {x, 3, nmax},
    PlotStyle → (clrs[[1]]),
    Epilog → {PointSize@Medium, clrs[[1]], Point@pts10}];
  pts15 = MapThread[{#1, #2} &, {Range[3, nmax, 1], Second /@ allSumCos15}];
  pl15 = ListLinePlot[pts15, Joined → True,
    PlotStyle → (clrs[[2]])];
  ptsRatio = MapThread[{#1, safeDiv[#2, #3]} &,
    {First /@ pts15, Second /@ pts15, Second /@ pts10}];
  plRatio = ListLinePlot[ptsRatio, PlotStyle → (clrs[[3]])];
  (* ListLinePlot[] not drawing epilog *)
  gr15 = Graphics[{PointSize@Medium, clrs[[2]], Point@pts15}];
  Legended[Show[{pl10, pl15, gr15, plRatio},
    PlotLabel → Style["∑cos(θi) for N=3..." <> ToString@nmax, {Black, 20}],
    FrameLabel → (Style[#, {Black, 16}] & /@ {"N", "Σcos"}),
    FrameStyle → Directive@{Black, Medium}, Frame → True,
    ImageSize → Large, GridLines → {Range[3, nmax], Automatic},
    FrameTicks → {Range[3, nmax], Automatic}],
   LineLegend[Directive[Thick, #] & /@ clrs,
    {"∑cos, a/b=1 (regular, exact)", "∑cos, a/b=1.5 (numeric)", "ratio"}]]]
```

Out[880]=



In[881]:=

```
Module[{nmax = 9},
  {N@Table[sumCosRegAng[i], {i, 3, nmax}], Chop[Second /@ allSumCos15]} //
       Transpose //
      Map[Function[x, Append[x, safeDiv[x[[2]], x[[1]]]]], #] & //
     MapThread[Function[{x, y}, Prepend[x, y]], {#, Range[3, nmax]}] & //
    Prepend[#, {"N", "∑cos|a/b=1", "∑cos|a/b=1.5", "ratio"}] & //
   Grid[#, Frame → All, ItemStyle → {Black, 16}] &]
```

Out[881]=

N	$\sum\cos\|_{a/b=1}$	$\sum\cos\|_{a/b=1.5}$	ratio
3	1.5	1.36266	0.90844
4	0.	0	0.
5	-1.54508	-1.51287	0.979149
6	-3.	-2.96	0.986667
7	-4.36443	-4.3237	0.990668
8	-5.65685	-5.61767	0.993073
9	-6.8944	-6.85745	0.994641

Tabachnikov' s request : cosine sum will be constant if sum 1/((x/a^2)^2 + y^2) is constant

In[924]:=
```
Clear@tabachnikovSum;
tabachnikovSum[a_, poly_, n_: 2] :=
  Total[(1 / Sqrt[(#[[1]] / a^2) ^2 + #[[2]] ^2] ^n) & /@ poly];
```

In[926]:=
```
tabachnikovSum[1.5, {{-1, 0}, {1, 0}, {1, 1}}, 2]
```

Out[926]=
```
10.9601
```

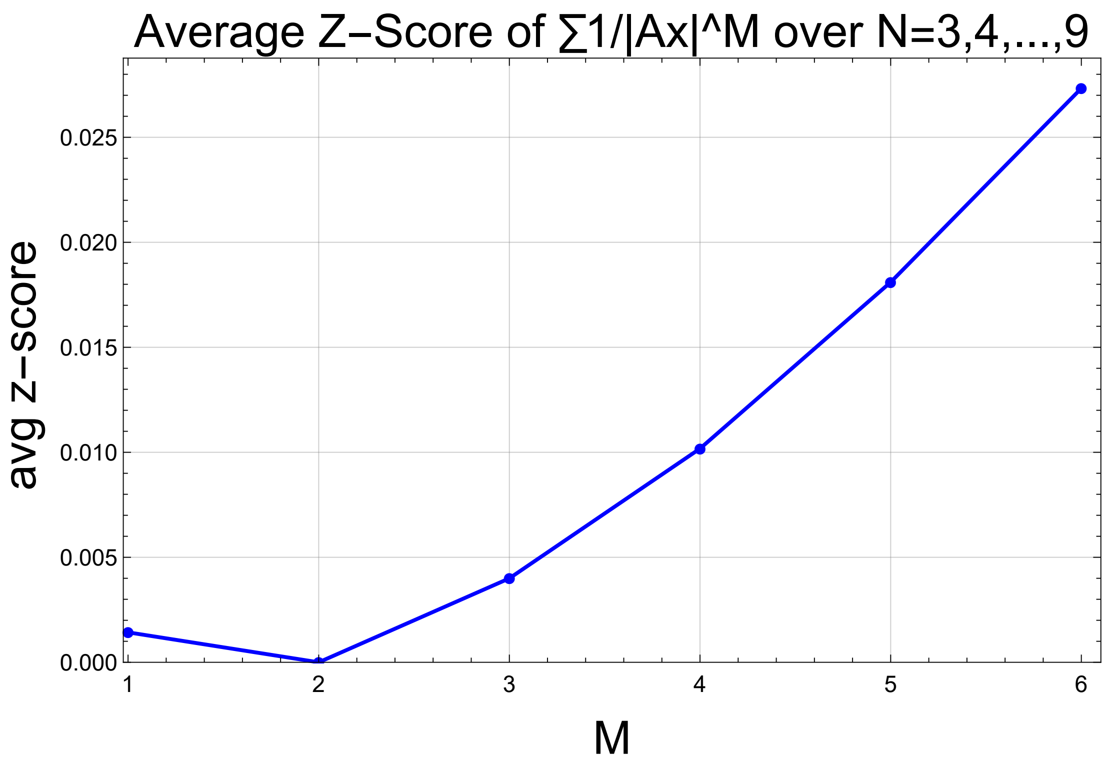
In[1000]:=
```
Clear@tabachnikovSums;
tabachnikovSums[alphaT_, fnVtx0_, n_: 2] := Module[{poly, a, sums, sd, mean, z},
   a = "a" /. alphaT;
   sums = Table[tabachnikovSum[a, polyVtx[alphaT, i, fnVtx0], n],
     {i, Length["alphas" /. alphaT]}];
   mean = Mean@sums;
   sd = StandardDeviation@sums;
   {mean, sd, sd / mean}];
```

In[1002]:=
```
allTabachnikovSumsAvgZ[n_] := Module[{sums},
  sums = MapThread[tabachnikovSums[#1, #2, n] &,
    {{triAlphaT15, quadAlphaT15, pentAlphaT15,
      hexAlphaT15, heptAlphaT15, octAlphaT15, nonaAlphaT15},
     {getTriVtx0, getQuadVtx0, getPentVtx0, getHexVtx0,
      getHeptVtx0, getOctVtx0, getNonaVtx0}}];
  Mean[Third /@ sums]]
```

In[1003]:=
```
tabSumsAvgZ = {#, allTabachnikovSumsAvgZ@#} & /@ Join[Range[1, 6]];
```

In[1004]:=
```
ListLinePlot[tabSumsAvgZ,
 PlotMarkers → Automatic, PlotStyle → {Thick, Blue},
 Frame → True, FrameStyle → Medium,
 PlotLabel → Style["Average Z-Score of ∑1/|Ax|^M over N=3,4,...,9", Large],
 FrameLabel → (Style[#, Large] & /@ {"M", "avg z-score"}),
 GridLines → Automatic, AxesOrigin → {1, 0},
 ImageSize → Large, PlotRange → {{.975, Automatic}, {-0, Automatic}}]
```

Out[1004]=



In[1005]:=
```
allTabachnikovSums15 = MapThread[tabachnikovSums[#1, #2, 2] &,
   {{triAlphaT15, quadAlphaT15, pentAlphaT15,
     hexAlphaT15, heptAlphaT15, octAlphaT15, nonaAlphaT15},
    {getTriVtx0, getQuadVtx0, getPentVtx0, getHexVtx0,
     getHeptVtx0, getOctVtx0, getNonaVtx0}}];
Grid[Prepend[Transpose@Prepend[Transpose@allTabachnikovSums15, Range[3, 9, 1]],
  {"N", "μ", "σ", "σ/μ"}], Frame → All]
```

Out[1006]=

N	μ	σ	σ/μ
3	5.20256	1.99096×10^{-14}	3.82688×10^{-15}
4	6.5	4.04384×10^{-8}	6.2213×10^{-9}
5	7.97812	1.78431×10^{-9}	2.23651×10^{-10}
6	9.5	5.65423×10^{-7}	5.95182×10^{-8}
7	11.0383	1.20683×10^{-7}	1.09331×10^{-8}
8	12.5846	1.46109×10^{-8}	1.16102×10^{-9}
9	14.1353	2.59278×10^{-9}	1.83426×10^{-10}

In[1007]:=
```
allTabachnikovSums15n4 = MapThread[tabachnikovSums[#1, #2, 4] &,
    {{triAlphaT15, quadAlphaT15, pentAlphaT15,
      hexAlphaT15, heptAlphaT15, octAlphaT15, nonaAlphaT15},
     {getTriVtx0, getQuadVtx0, getPentVtx0, getHexVtx0,
      getHeptVtx0, getOctVtx0, getNonaVtx0}}];
Grid[Prepend[Transpose@Prepend[Transpose@allTabachnikovSums15n4, Range[3, 9, 1]],
   {"N", "μ", "σ", "σ/μ"}], Frame → All]
```

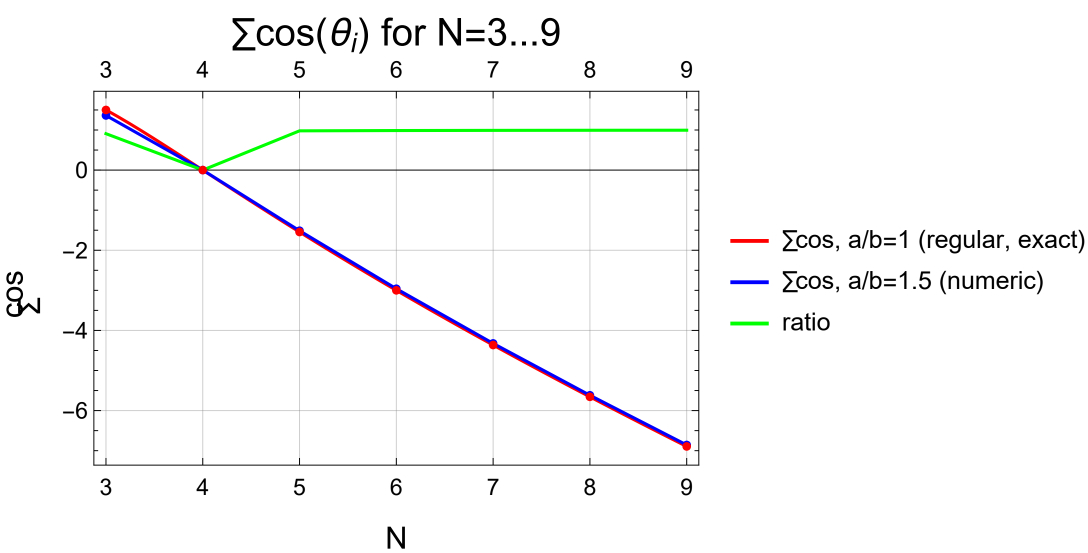
Out[1008]=

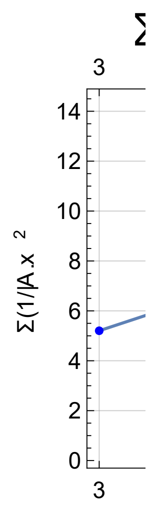
N	μ	σ	σ/μ
3	9.63141	0.144915	0.0150461
4	11.375	0.552753	0.0485937
5	13.716	0.000867046	0.0000632142
6	16.2187	0.110639	0.00682169
7	18.7759	5.11055×10^{-6}	2.72186×10^{-7}
8	21.359	0.0152742	0.000715116
9	23.9566	2.79964×10^{-8}	1.16863×10^{-9}

In[909]:=
```
sumTabachnikov15 = Module[{pts, nmax = 9},
   pts = MapThread[{#1, #2} &, {Range[3, nmax, 1], First /@ allTabachnikovSums15}];
   ListLinePlot[pts, Epilog → {PointSize@Medium, Blue, Point@pts}, PlotLabel →
     Style["∑ 1/|A.xi|^2 for N=3..." <> ToString@nmax <> ", a/b=1.5", {Black, 20}],
    FrameLabel → {"N", "Σ(1/|A.x|^2"}, FrameStyle → Directive@{Black, Medium},
    Frame → True, GridLines → {Range[3, nmax], Automatic},
    FrameTicks → {Range[3, nmax], Automatic}]];
```

In[910]:=
```
Grid[{Show[#, ImageSize → Medium] & /@ {sumCos15, sumTabachnikov15}}]
```

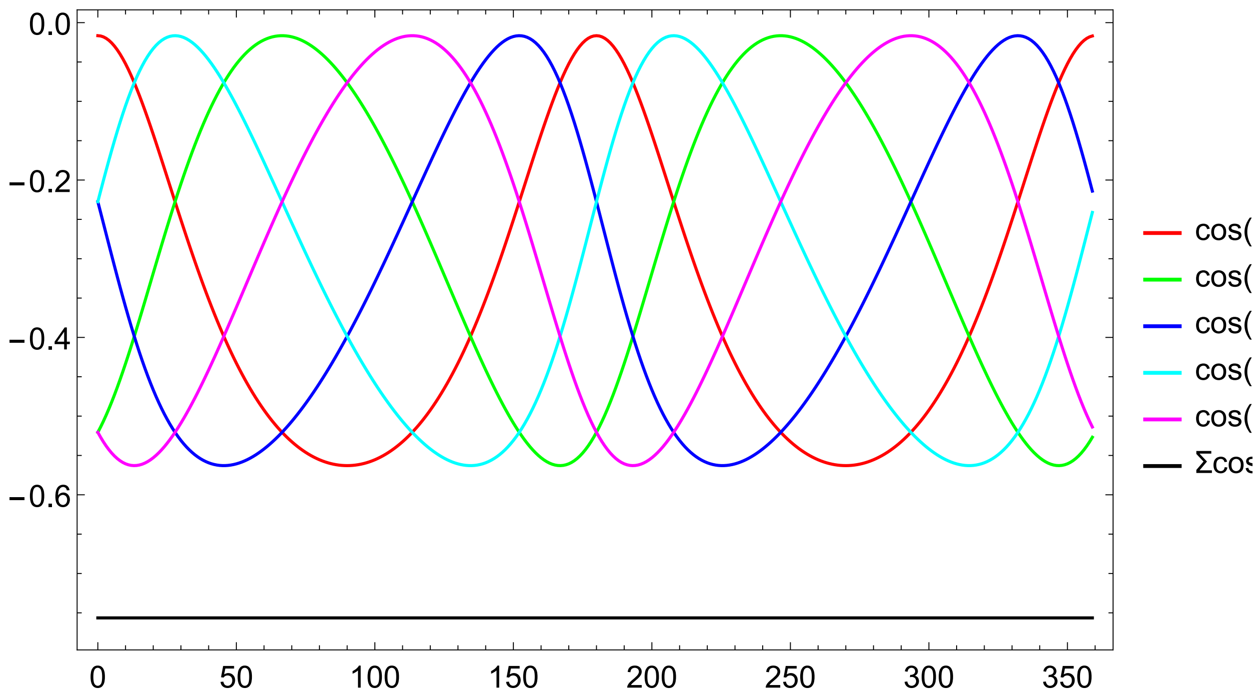
Out[910]=





In[890]:=
```
Module[{ts, clrs = {Red, Green, Blue, Cyan, Magenta, Black}, pents, cosPent15},
 ts = "tsDeg" /. pentAlphaT15;
 pents = Table[polyVtx[pentAlphaT15, i, getPentVtx0],
   {i, Length["alphas" /. pentAlphaT15]}];
 cosPent15 = getPolyCosines /@ pents;
 Legended[ListLinePlot[Transpose /@ {
     {ts, First /@ cosPent15},
     {ts, Second /@ cosPent15},
     {ts, Third /@ cosPent15},
     {ts, Fourth /@ cosPent15},
     {ts, Fifth /@ cosPent15},
     {ts, .5 (Total /@ cosPent15)}}, Frame → True,
   FrameStyle → 16, PlotStyle → clrs, ImageSize → Large],
  LineLegend[Directive[{Thick, #}] & /@ clrs,
   Style[#, 16] & /@
    Flatten@{("cos(" <> # <> ")") & /@ {"A", "B", "C", "D", "E"}, "Σcos/2"}]]]
```

Out[890]=



Plot N=3

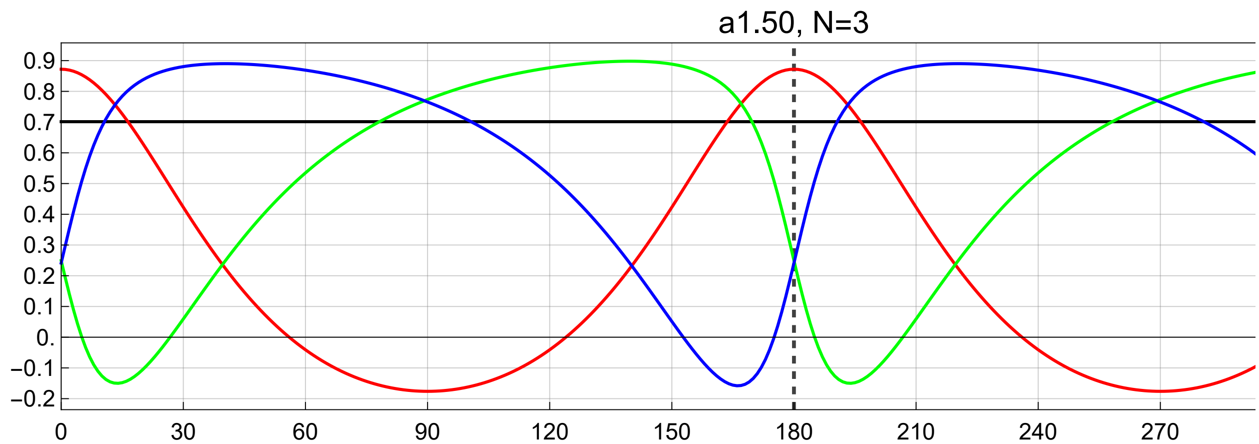
In[891]:=
```
plotCosN3 = Module[{a = 1.5, ts, cosTri15,
    clrs = {Red, Green, Blue, Black}, ticksx, ticksy, ticksxGrid},
   ts = Table[t, {t, 0, 359, 1}];
   cosTri15 = getOrbitCosines[a, toRad@#] & /@ ts;
   ticksx = Table[i, {i, 0, Max[ts] + 1, 30}];
   ticksxGrid =
    ticksx /. {180 → {180, Directive[Thick, Black, Dashed, Opacity@.75]}};
   ticksy = Table[i, {i, -.5, 1.5, .25}];
   Legended[ListLinePlot[Transpose /@ {
        {ts, First /@ cosTri15},
        {ts, Second /@ cosTri15},
        {ts, Third /@ cosTri15},
        {ts, (Total /@ cosTri15)}
      }, Frame → True, FrameStyle → 12, PlotLabel → Style["N=3", {16, Black}],
     PlotRange → {{0, 360}, Automatic}, PlotStyle → clrs,
     AspectRatio → .25, FrameTicks → {{ticksy, None}, {ticksx, None}},
     GridLines → {ticksxGrid, ticksy}, ImageSize → 800],
    LineLegend[Directive[{Thick, #}] & /@ clrs,
     Style[#, 16] & /@ Flatten@{("cos(" <> # <> ")") & /@ {"A", "B", "C"}, "Σcos"}]]];
```

Plot N=4

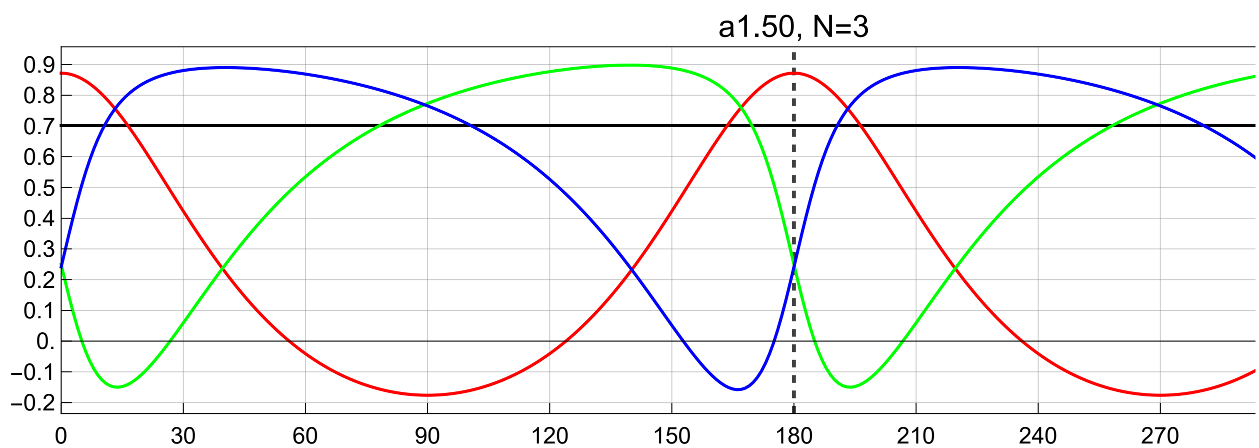
In[1826]:=
```
pTriAll = plotPolyCos[triAlphaT15, getTriVtx0];
pQuadAll = plotPolyCos[quadAlphaT15, getQuadVtx0, pert -> .015];
pPentAll = plotPolyCos[pentAlphaT15, getPentVtx0];
pHexAll = plotPolyCos[hexAlphaT15, getHexVtx0, pert -> 0.02, cosDiv → 5];
pHeptAll = plotPolyCos[heptAlphaT15, getHeptVtx0, cosDiv → 5];
pOctAll = plotPolyCos[octAlphaT15, getOctVtx0, pert → .02, cosDiv → 7];
pNonaAll = plotPolyCos[nonaAlphaT15, getNonaVtx0, cosDiv → 8];
```

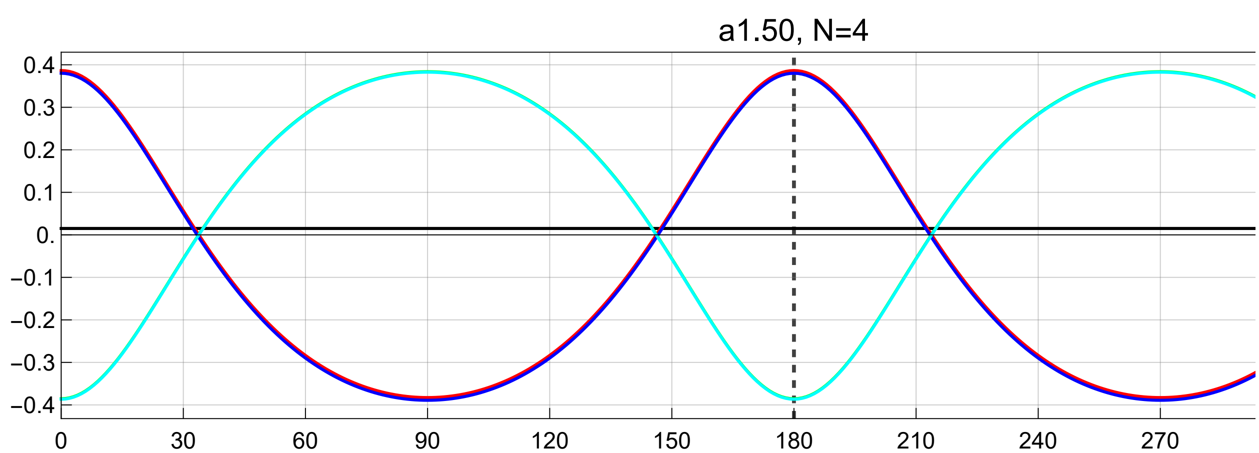
In[1834]:= `pTriAll`

Out[1834]=

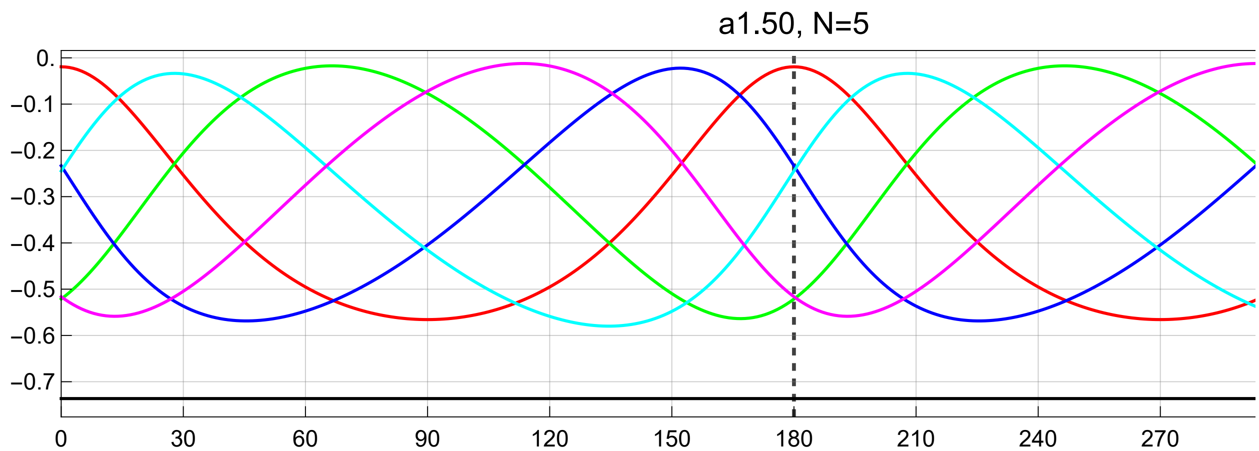


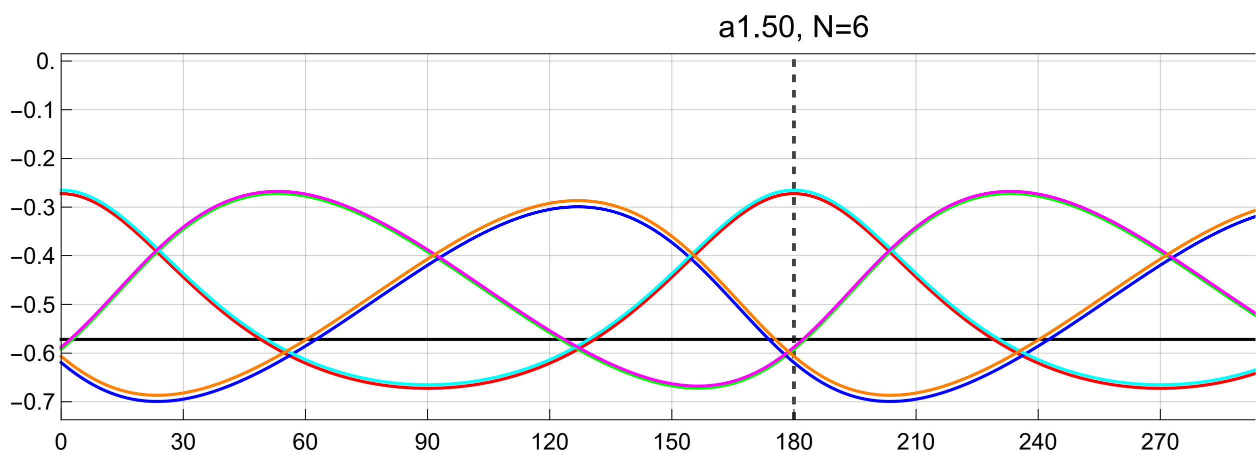
In[1833]:= `Column[{pTriAll, pQuadAll, pPentAll, pHexAll, pHeptAll, pOctAll, pNonaAll}]`

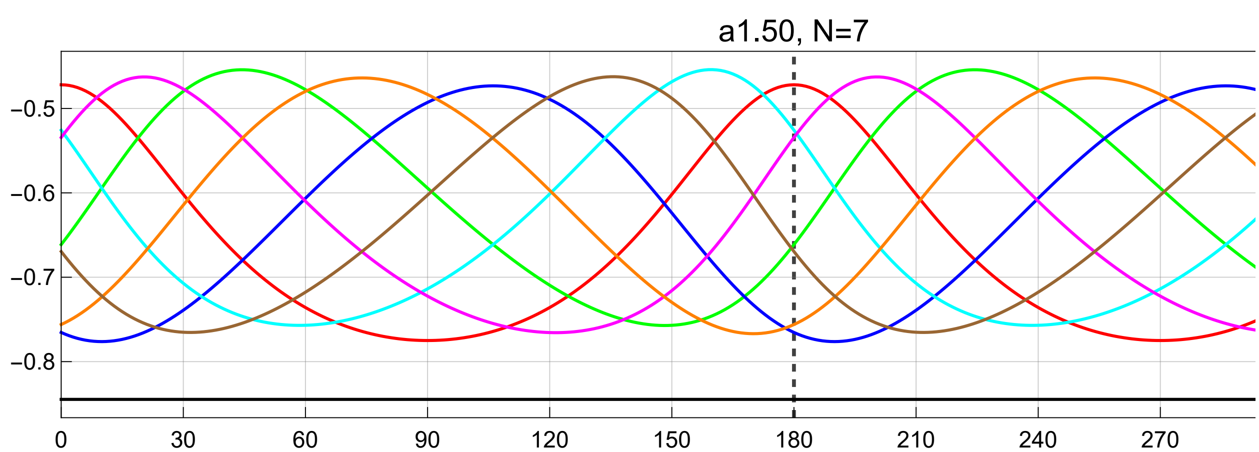


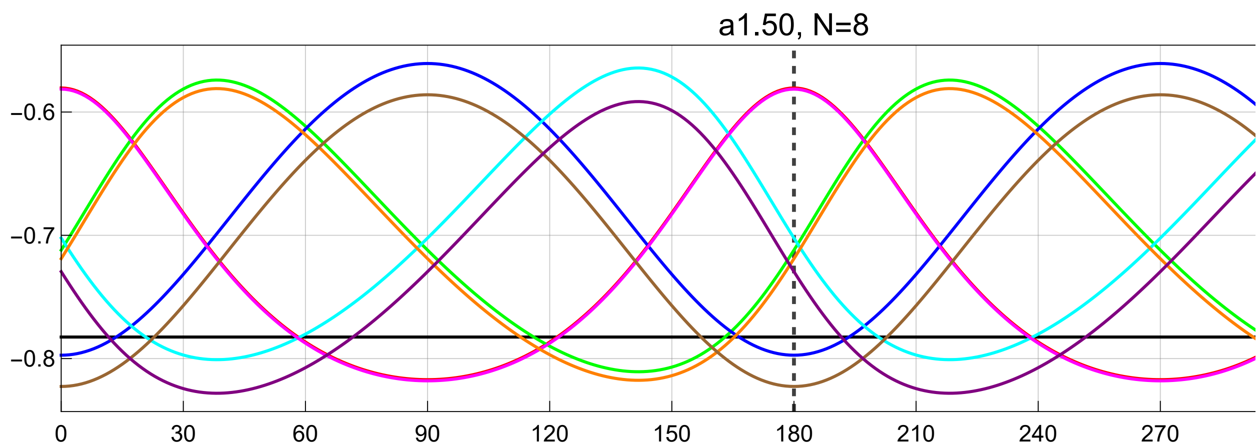


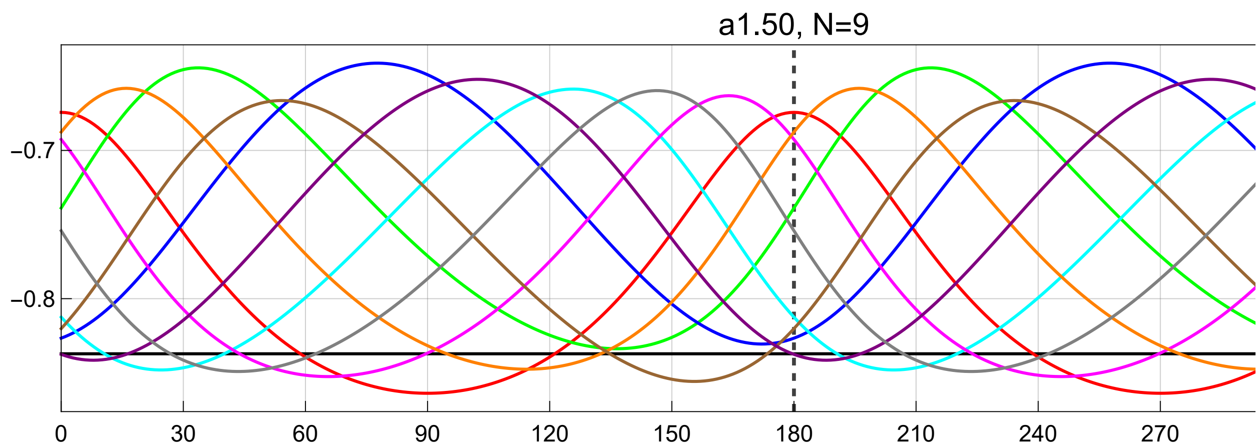
Out[1833]=


a1.50, N=5
0.
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
0
30
60
90
120
150
180
210
240
270


a1.50, N=6
0.
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
0
30
60
90
120
150
180
210
240
270


a1.50, N=7
−0.5
−0.6
−0.7
−0.8
0
30
60
90
120
150
180
210
240
270

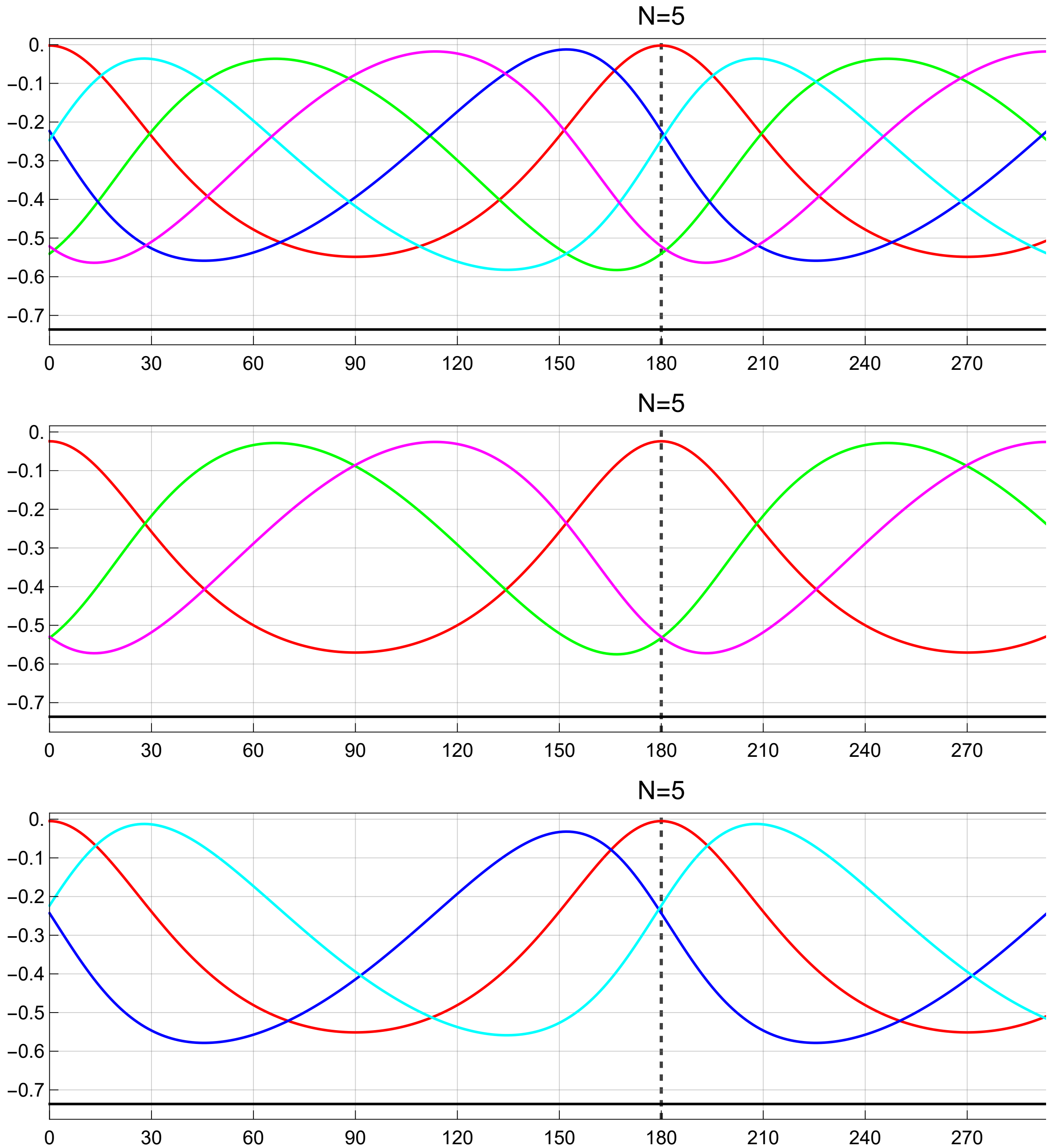




In[900]:=
```
pPentABE = plotPolyCos[pentAlphaT15, getPentVtx0, keep -> {1, 2, 5}];
pPentACD = plotPolyCos[pentAlphaT15, getPentVtx0, keep -> {1, 3, 4}];
```

In[902]:= `Column[{pPentAll, pPentABE, pPentACD}]`

Out[902]=



Is Cos(α) | $_{N>3}$ a perfect multiple of Cos(α) | $_{N=3}$???

```
Clear@getAlphaRatio;
getAlphaRatio[alphaT_, fnVtx0_] := Module[{ts, a, cosTri, polys, cosPoly, meanTri,
    meanPoly, ratios, meanRatios, sdRatios, cosTriTransf, k1, k2, lm},
   ts = "tsDeg" /. alphaT;
   a = "a" /. alphaT;
   cosTri = getOrbitCosines[a, toRad@#] & /@ ts;
   meanTri = Mean[First /@ cosTri];
   polys = Table[polyVtx[alphaT, i, fnVtx0], {i, Length["alphas" /. alphaT]}];
   cosPoly = getPolyCosines /@ polys;
   meanPoly = Mean[First /@ cosPoly];
   ratios =
    MapThread[((#2 - meanPoly) / (#1 - meanTri)) &, {First /@ cosTri, First /@ cosPoly}];
   meanRatios = Mean@ratios;
   sdRatios = StandardDeviation@ratios;
   cosTriTransf = (First /@ cosTri - meanTri) * meanRatios + meanPoly;
   k1 = meanRatios;
   k2 = -meanTri * meanRatios + meanPoly;
   (*Print["sd=",sdRatios,", k1=",k1," k2=",k2];*)
   lm =
    LinearModelFit[MapThread[{#1, #2} &, {First /@ cosTri, First /@ cosPoly}], x, x];
   {k1, k2, sdRatios / meanRatios, Normal@lm, lm["RSquared"], cosPoly, cosTriTransf}];
```

```
Take[getAlphaRatio[pentAlphaT15, getPentVtx0], 5]
```

{0.521235, -0.478765, 5.46546×10^{-14}, -0.478765 + 0.521235 x, 1.}

***Jair : perform experiments with a/b = 1.25, 2.0,

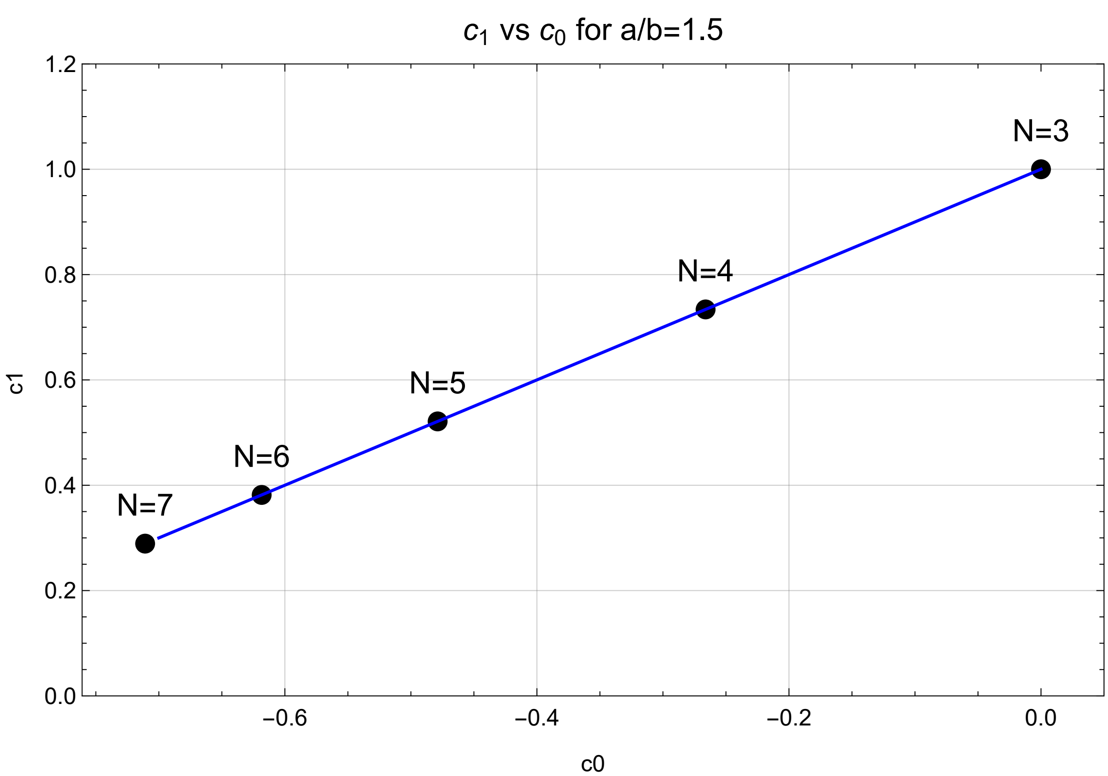
```
linearRatios = MapThread[Prepend[Take[getAlphaRatio[#1, #2], 5], #3] &,
     {{triAlphaT15, quadAlphaT15, pentAlphaT15, hexAlphaT15, heptAlphaT15},
      {getTriVtx0, getQuadVtx0, getPentVtx0, getHexVtx0, getHeptVtx0},
      {3, 4, 5, 6, 7}}] // Prepend[#, {"N", "k1", "k2", "sd/mu", "lm", "rsquared"}] &;
linearRatios // TableForm
```

N	k1	k2	sd/mu	lm	rsquared
3	1.	-2.33147×10^{-15}	3.24618×10^{-14}	-1.35752×10^{-15} + 1. x	1.
4	0.733859	-0.266141	1.44453×10^{-7}	-0.266141 + 0.733859 x	1.
5	0.521235	-0.478765	5.46546×10^{-14}	-0.478765 + 0.521235 x	1.
6	0.381607	-0.618393	2.60202×10^{-6}	-0.618393 + 0.381607 x	1.
7	0.289133	-0.710867	3.82655×10^{-14}	-0.710867 + 0.289133 x	1.

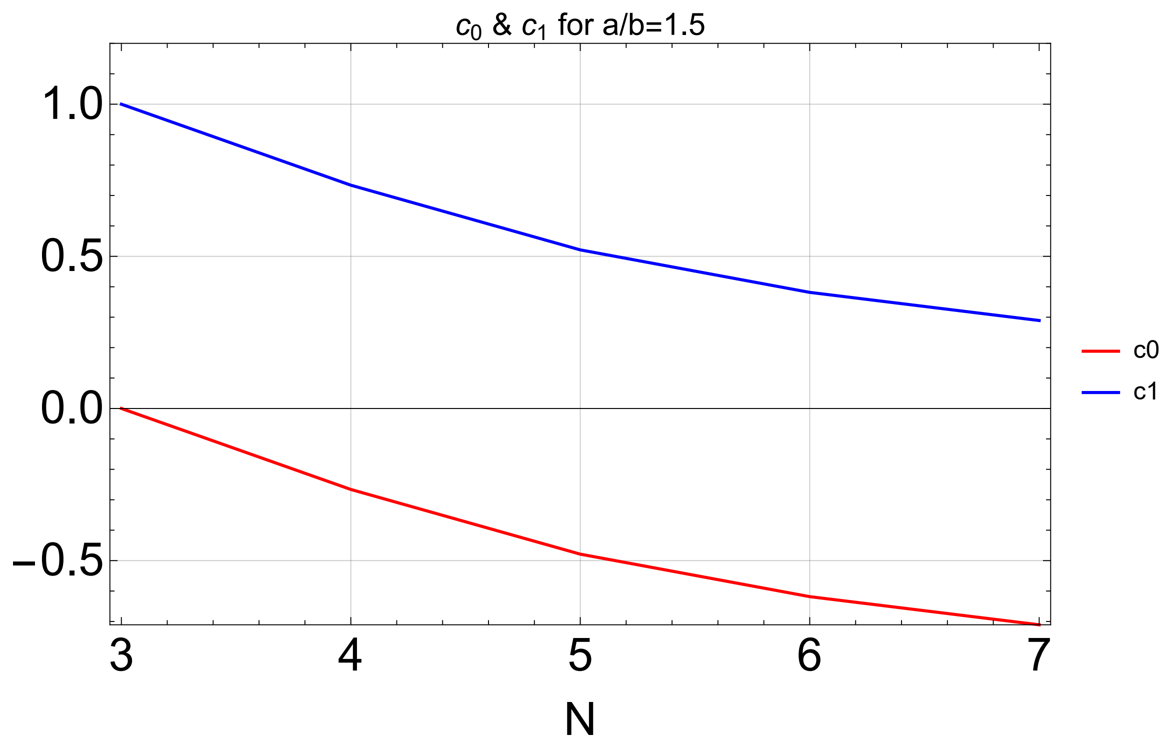
```
Prepend[{#[[1]], nfn[Chop@Coefficient[#[[2]], x, 0], 3],
      nfn[Coefficient[#[[2]], x, 1], 3]} & /@ Rest[Part[#, {1, 5}] & /@ linearRatios],
   {"N", "c0|a=1.5", "c1|a=1.5"}] // Grid[#, Frame → All,
    ItemStyle → {FontFamily → "Helvetica", Medium}, Spacings → {1, 1}] &
```

N	$c_0 \vert_{a=1.5}$	$c_1 \vert_{a=1.5}$
3	0.000	1.000
4	−0.266	0.734
5	−0.479	0.521
6	−0.618	0.382
7	−0.711	0.289

```
Module[{coeffs},
 coeffs =
  CoefficientList[#[[2]], x] & /@ Chop@Rest[Part[#, {1, 5}] & /@ linearRatios];
 Show[{ListPlot[coeffs, PlotStyle → Black, Epilog → MapThread[
       Text[Style["N=" <> ToString@#1, 16], #2, {0, -2.}] &, {Range[3, 7], coeffs}]],
    Plot[x + 1, {x, -.7, 0}, PlotStyle → Blue]}, Frame → True, GridLines → Automatic,
  PlotRangePadding → {{0.05, .05}, {0, .2}}, FrameStyle → Medium,
  PlotLabel → Style["c1 vs c0 for a/b=1.5", {Black, 16}],
  ImageSize → Large, FrameLabel → {"c0", "c1"}]]
```

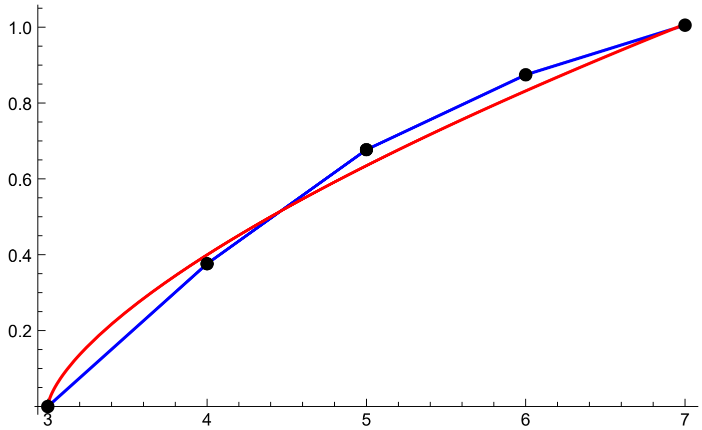


```
Module[{coeffs},
 coeffs =
  CoefficientList[#[[2]], x] & /@ Chop@Rest[Part[#, {1, 5}] & /@ linearRatios];
 Show[{ListPlot[
    {Legended[MapThread[{#1, #2} &, {Range[3, 7], First /@ coeffs}], "c0"],
     Legended[MapThread[{#1, #2} &, {Range[3, 7], Second /@ coeffs}], "c1"]},
    Joined → True, PlotStyle → {Red, Blue}, Epilog → MapThread[
      Text[Style["N=" <> ToString@#1, 16], #2, {0, -2.}] &, {Range[3, 7], coeffs}]]},
  Frame → True, GridLines → Automatic, PlotRangePadding → {{0.05, .05}, {0, .2}},
  FrameStyle → Large, PlotLabel → Style["c₀ & c₁ for a/b=1.5", {Black, 16}],
  ImageSize → Large, FrameLabel → {"N", ""}]]
```

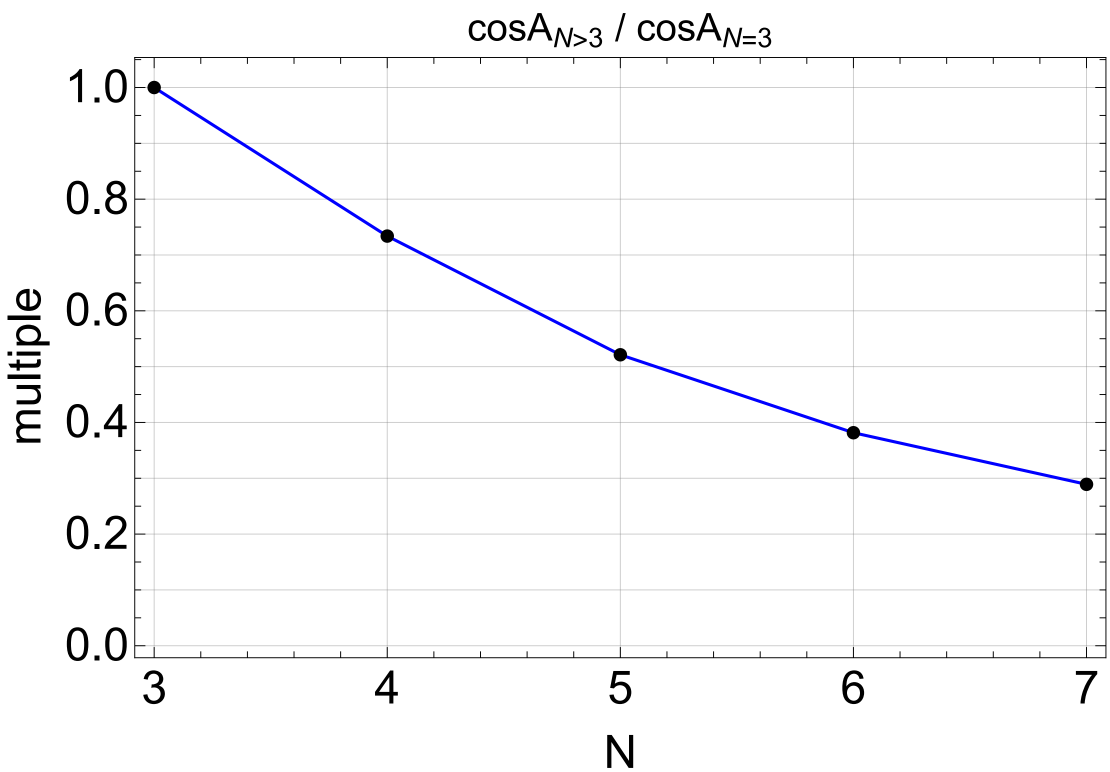


Try to model by distance to (0, 1) point

```
Module[{coeffs, ds, ll, pl},
 coeffs =
  CoefficientList[#[[2]], x] & /@ Chop@Rest[Part[#, {1, 5}] & /@ linearRatios];
 ds = MapThread[{#1, #2} &, {Range[3, 7], magn[# - {0, 1}] & /@ coeffs}];
 pl = Plot[.4 (x - 3) ^ (2 / 3), {x, 3, 7}, PlotStyle → Red];
 ll = ListLinePlot[ds, PlotStyle → Blue, Epilog → {PointSize@Large, Point@ds}];
 Show[{ll, pl}]]
```

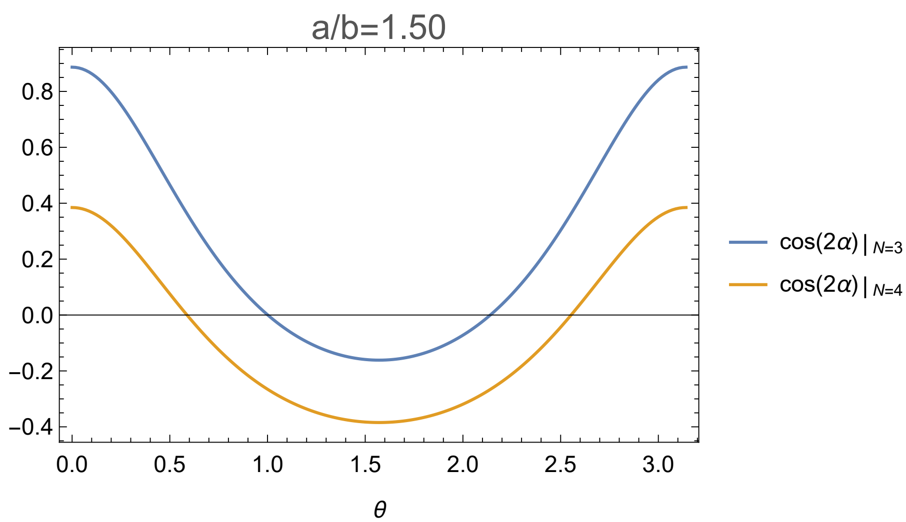


```
Module[{pts, ticks},
 ticks = {Range[3, 7, 1], Range[0, 1, .1]};
 pts =
  Rest[Transpose@{First /@ linearRatios, Coefficient[Fifth /@ linearRatios, x, 1]}];
 ListPlot[pts, Joined → True, Epilog → {PointSize@Large, Point@pts},
  PlotStyle → Blue,
  Frame → True, FrameStyle → Large,
  GridLines → ticks, FrameLabel → {"N", "multiple"},
  ImagePadding → {{Automatic, Automatic}, {Automatic, Scaled[0.01]}},
  ImageSize → Large,
  PlotLabel → Style["cosA_{N>3} / cosA_{N=3}", Directive[Black, 20]]]]
```



Ronaldo proveu formula para Cos[alpha] | N = 4. O que estamos vendo é uma razao linear entre os cossenos de 2A.

```
Module[{a = 1.5, t, pl, lm, c3, c4},
 c3 = Table[t = toRad[tDeg];
   {t, cosDoubleAngle[cosAlpha[a, a Cos[t]]]}, {tDeg, 0, 180}];
 c4 = Table[t = toRad[tDeg];
   {t, cosDoubleAngle[cosAlphaQuad[a, a Cos[t]]]}, {tDeg, 0, 180}];
 pl = ListLinePlot[{Legended[c3, "cos(2α)|N=3"], Legended[c4, "cos(2α)|N=4"]},
   Frame → True, FrameStyle → Medium, FrameLabel → {"θ"},
   PlotLabel → Style["a/b=" <> nfn[a, 2], {Darker@Gray, 18}]];
 Print[pl];
 lm = LinearModelFit[MapThread[{Second[#1], Second[#2]} &, {c3, c4}], x, x];
 {Normal@lm, lm["RSquared"]}]
```




```
{-0.266141 + 0.733859 x, 1.}
```

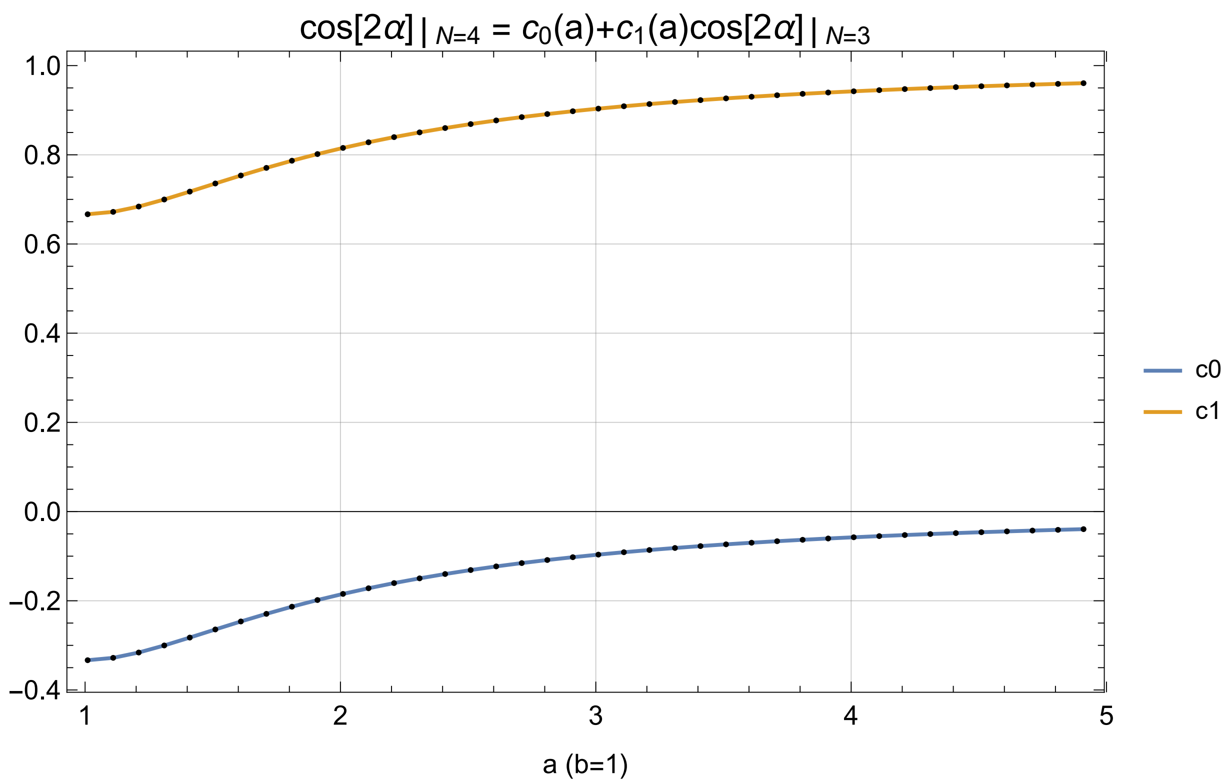
```
getCos4LinearModel[a_] := Module[{t, pl, lm, c3, c4, norm},
  c3 = Table[t = toRad[tDeg];
    {t, cosDoubleAngle[cosAlpha[a, a Cos[t]]]}, {tDeg, 0, 180}];
  c4 = Table[t = toRad[tDeg];
    {t, cosDoubleAngle[cosAlphaQuad[a, a Cos[t]]]}, {tDeg, 0, 180}];
  lm = LinearModelFit[MapThread[{Second[#1], Second[#2]} &, {c3, c4}], x, x];
  norm = Normal@lm;
  {Coefficient[norm, x, 0], Coefficient[norm, x, 1], lm["RSquared"]}]

cos4lms = Table[{a, Sequence @@ getCos4LinearModel[a]}, {a, 1.01, 5, .1}];
```

R^2 all perfect!

```
Fourth /@ cos4lms
```

```
{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., 1., 1., 1., 1., 1., 1., 1., 1.}
```

```
Module[{tr, c0, c1}, tr = Transpose[Take[#, 3] & /@ cos4lms];
 c0 = MapThread[{#1, #2} &, {tr[[1]], tr[[2]]}];
 c1 = MapThread[{#1, #2} &, {tr[[1]], tr[[3]]}];
 ListLinePlot[{Legended[c0, "c0"], Legended[c1, "c1"]}, PlotStyle → Thick,
  Epilog → {Point@c0, Point@c1}, Frame → True, FrameStyle → Directive[Black, 14],
  PlotLabel → Style["cos[2α]|N=4 = c0(a)+c1(a)cos[2α]|N=3", {Black, 18}],
  FrameLabel → (Style[#, {Black, 14}] & /@ {"a (b=1)"}),
  GridLines → Automatic, ImageSize → Large,
  ImagePadding → {{Automatic, Automatic}, {Automatic, Scaled[0.005]}}]]
```



Closed – Form : Cos[2 α] $|_{N=4}$ = c0 + c1 Cos[2 α] $|_{N=3}$

```
c4ronaldo[a_] := {2 (-a^2 - 1 + Sqrt[a^4 - a^2 + 1]) / (3 * a^2 + 3),
  (a^2 + 1 + 2 Sqrt[a^4 - a^2 + 1]) / (3 a^2 + 3)}
```

```
c4ronaldo[a]
```

$$\left\{\frac{2\left(-1-a^2+\sqrt{1-a^2+a^4}\right)}{3+3\,a^2},\ \frac{1+a^2+2\sqrt{1-a^2+a^4}}{3+3\,a^2}\right\}$$

```
c4ronaldo[1.5]
```

```
{-0.266141, 0.733859}
```

```
Module[{c4, c5},
 c4 = -0.2661410422611972 + 0.7338589577388027 c3;
 c5 = FullSimplify[-0.2661410422611972 + 0.7338589577388027 c4];
 {c4, c5}]
```

```
{-0.266141 + 0.733859 c3, -0.461451 + 0.538549 c3}
```

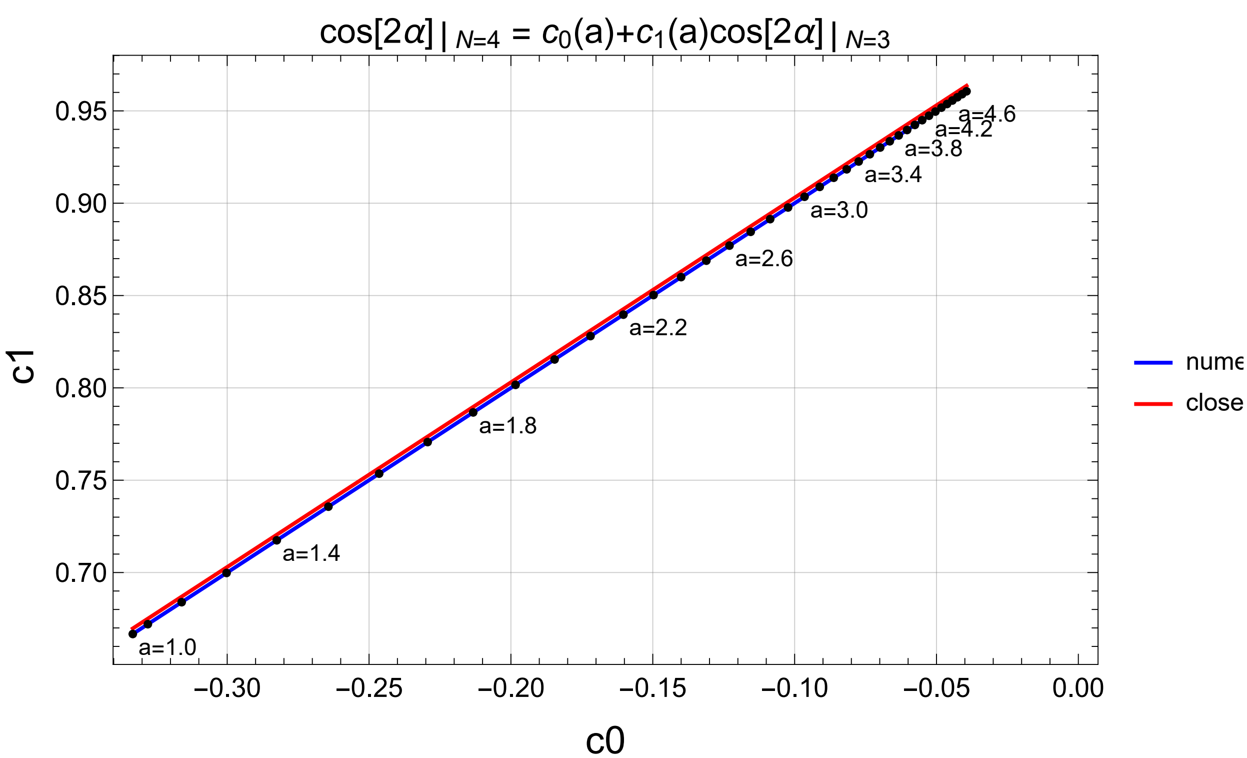
```
FullSimplify[c4ronaldo[a][[1]] - 1 - c4ronaldo[a][[2]]]
```

```
-2
```

```
cos4lmsRonaldo = Table[{a, Sequence @@ c4ronaldo[a]}, {a, 1.01, 5, .1}];
```

Relationship between c0 and c1 is linear as well: c0 = 1+ c1

```
Module[{tr, trRon, c01, c01ron},
 tr = Transpose[Take[#, 3] & /@ cos4lms];
 trRon = Transpose[Take[#, 3] & /@ cos4lmsRonaldo];
 c01 = MapThread[{#1, #2} &, {tr[[2]], tr[[3]]}];
 c01ron = MapThread[{#1, #2} &, {tr[[2]], tr[[3]] + .003}];
 ListLinePlot[{Legended[c01, "numeric"], Legended[c01ron, "closed-form"]},
  PlotStyle → {{Thick, Blue}, {Thick, Red}}, Epilog → {PointSize@Medium, Point@c01,
    MapThread[Text[Style["a=" <> nfn[#1, 1], {Black, 12}], #2, {-1.2, .8}] &,
     Take[#, {1, -1, 4}] & /@ {First /@ cos4lms, c01}]},
  Frame → True, FrameStyle → Directive[Black, 14],
  PlotLabel → Style["cos[2α]|N=4 = c0(a)+c1(a)cos[2α]|N=3", {Black, 18}],
  FrameLabel → (Style[#, {Black, 20}] & /@ {"c0", "c1"}),
  GridLines → Automatic, ImageSize → Large,
  ImagePadding → {{Automatic, Automatic}, {Automatic, Scaled[0.005]}}]]
```



Algorithm para N = 4, dado um "a"

1) calculamos $c_1 = \frac{2\left(-1-a^2+\sqrt{1-a^2+a^4}\right)}{3+3\,a^2}$

2) calculamos $c_0 = 1 + c_1$

3) Cos[2 α] | $_{N=4}$ = c_0 + c_1 Cos[2 α] | $_{N=3}$ = 1 + c_1 (1 + Cos[2 α] | $_{N=3}$)

```
c4da = FullSimplify[cosDoubleAngle[cosAlphaQuad[a, a ct]], a > 1]
```

$$-1. + \frac{2\,a^2}{a^2 + ct^2 - a^4\,\left(-1 + ct^2\right)}$$

```
c3da = FullSimplify[cosDoubleAngle[cosAlpha[a, a ct]], a > 1]
```

$$-1. + \frac{2\,a^2\,\left(1 + a^2 - 2\,\sqrt{1 - a^2 + a^4}\right)}{\left(-1 + a^2\right)^2\,\left(-ct^2 + a^2\,\left(-1 + ct^2\right)\right)}$$

```
(c3da /. {ct -> Cos[π / 6]})
```

$$-1. + \frac{2\,a^2\,\left(1 + a^2 - 2\,\sqrt{1 - a^2 + a^4}\right)}{\left(-\frac{3}{4} - \frac{a^2}{4}\right)\,\left(-1 + a^2\right)^2}$$

```
(c0 + c1 (c3da /. {ct -> Cos[π / 6]}))
```

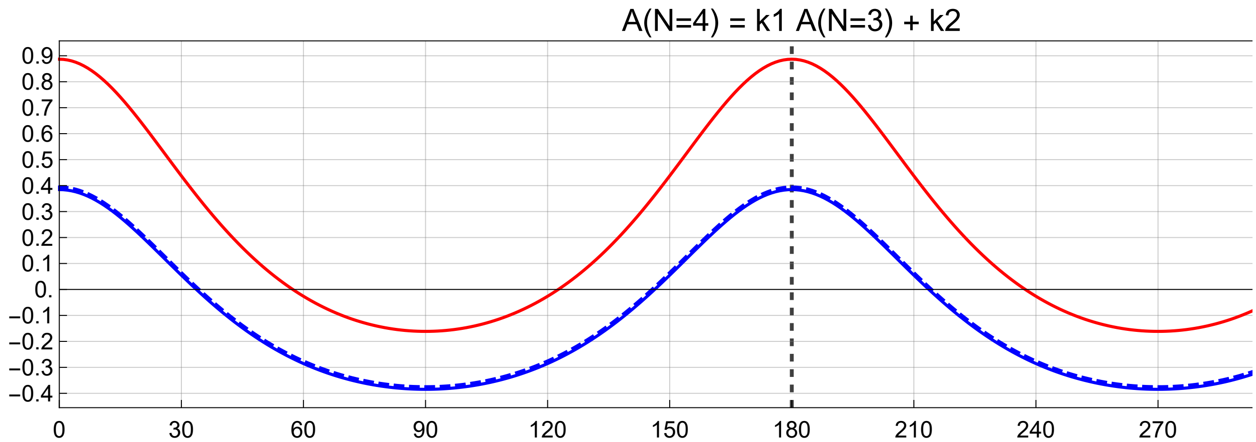
$$c0 + \left(-1. + \frac{2\,a^2\,\left(1 + a^2 - 2\,\sqrt{1 - a^2 + a^4}\right)}{\left(-\frac{3}{4} - \frac{a^2}{4}\right)\,\left(-1 + a^2\right)^2}\right)\,c1$$

```
solveC4C3Exact[a0_] := Module[{eqn1, eqn2},
  eqn1 = (c4da == c0 + c1 c3da) /. {ct → ct11, a → a0};
  eqn2 = (c4da == c0 + c1 c3da) /. {ct → ct22, a → a0};
  First[{c0, c1} /. FullSimplify[Solve[{eqn1, eqn2}, {c0, c1}], ct11 > 0 & ct22 > 0]]
 ]
```

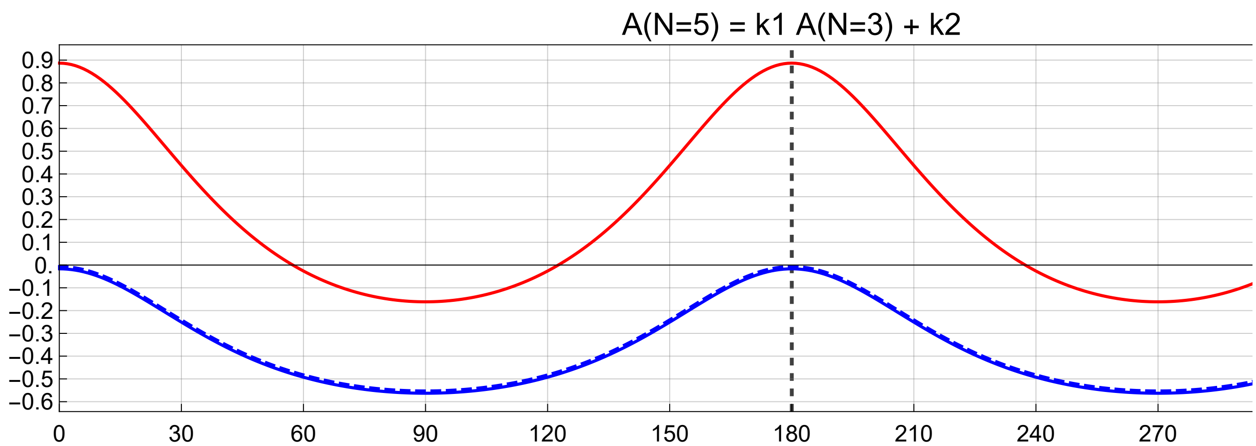
Alpha Ratios

```
Clear@drawAlphaRatio;
drawAlphaRatio[alphaT_, fnVtx0_] :=
  Module[{ts, a, cosTri, k1, k2, cosPoly, cosTriTransf, theN,
     ticksx, ticksxGrid, ticksy, strN, lab,
     clrs = {Red, Blue, {Dashed, Blue}}},
    ts = "tsDeg" /. alphaT;
    a = "a" /. alphaT;
    cosTri = getOrbitCosines[a, toRad@#] & /@ ts;
    (*{k1,k2,sdRatios/meanRatios,Normal@lm,lm["RSquared"],cosPoly,cosTriTransf}*)
    {k1, k2, cosPoly, cosTriTransf} =
     Part[getAlphaRatio[alphaT, fnVtx0], {1, 2, 6, 7}];
    theN = Length[cosPoly[[1]]];
    ticksx = Table[i, {i, 0, Max[ts] + 1, 30}];
    ticksxGrid =
     ticksx /. {180 → {180, Directive[Thick, Black, Dashed, Opacity@.75]}};
    ticksy = Table[i, {i, -1, 2, .1}];
    strN = "N=" <> ToString@theN;
    lab = "A(" <> strN <> ")  = k1  A(N=3)  +  k2";
    Legended[ListLinePlot[Transpose /@ {
         {ts, First /@ cosTri},
         {ts, First /@ cosPoly},
         {ts, cosTriTransf + .01}
       }, Frame → True, FrameStyle → 12, PlotStyle → clrs, AspectRatio → .25,
      PlotRange → {{0, 360}, Automatic}, FrameTicks → {{ticksy, None}, {ticksx, None}},
      GridLines → {ticksxGrid, ticksy},
      ImageSize → 800, PlotLabel → Style[lab, {16, Black}]],
     LineLegend[Directive[{Thick, #}] & /@ clrs,
      Style[#, 16] & /@ {"N=3", strN, strN <> " adj"}]]];
```

```
drawAlphaRatio[quadAlphaT15, getQuadVtx0]
```




```
drawAlphaRatio[pentAlphaT15, getPentVtx0]
```



Sum of Cosines and Cos(α) multiple via Caustics

N	$c_0 \mid_{a=1.5}$	$c_1 \mid_{a=1.5}$
3	0.000	1.000
4	−0.266	0.734
5	−0.479	0.521
6	−0.618	0.382
7	−0.711	0.289

In[1459]:=
```
Clear@getAlphaRatioCaustic;
getAlphaRatioCaustic[a_, n_] := Module[{ts, cosTri, meanTri, polys, cosPoly, lm},
   ts = Range[360] - 1.;
   cosTri = getOrbitCosines[a, toRad@#] & /@ ts;
   meanTri = Mean[First /@ cosTri];
   polys = getCausticOrbits[a, n, ts];
   cosPoly = getPolyCosines /@ polys;
   lm =
    LinearModelFit[MapThread[{#1, #2} &, {First /@ cosTri, First /@ cosPoly}], x, x];
   {Normal@lm, lm["RSquared"]}];
```

In[1461]:=
```
cosAlphaRatios = Module[{fname = "cosAlphaRatios.m", theT},
   If[False,
    theT = Flatten[
      Table[{a, n, getAlphaRatioCaustic[a, n]}, {a, 1.1, 3.0, .1}, {n, 4, 21, 1}], 1];
    Save[fname, theT];
    Print["saved ", Length@theT, " lines to ", fname];
    theT,
    theT = Get[fname];
    Print["loaded ", Length@theT, " lines from ", fname];
    theT]];
```

```
loaded 360 lines from cosAlphaRatios.m
```

Let' s look at R^2

In[1462]:=
```
getStats[#[[3, 2]] & /@ cosAlphaRatios]
```

Out[1462]= $\{1., 5.24093\times10^{-17}, 5.24093\times10^{-17}, 1., 1., 1., 360\}$

In[1463]:=
```
Take[cosAlphaRatios, 3]
```

Out[1463]=
```
{{1.1, 4, {-0.328849 + 0.671151 x, 1.}},
 {1.1, 5, {-0.535502 + 0.464498 x, 1.}}, {1.1, 6, {-0.663663 + 0.336337 x, 1.}}}
```

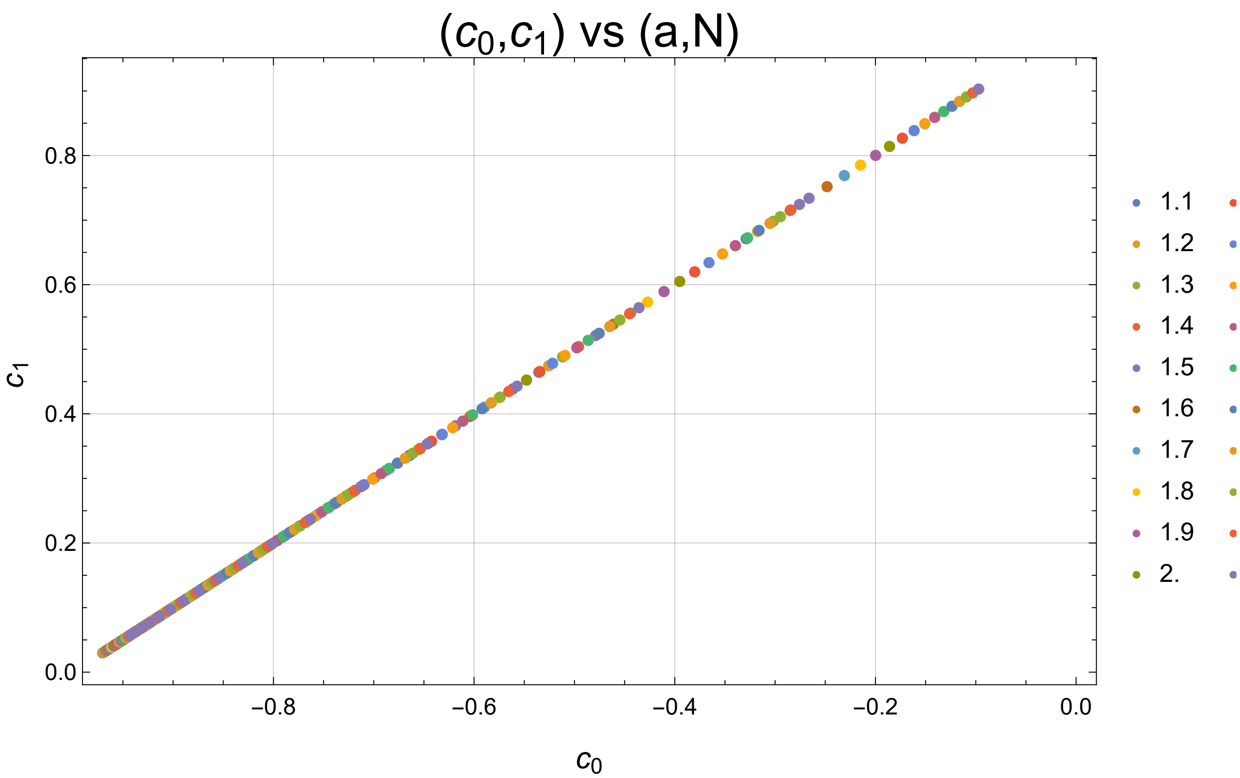
In[1464]:=
```
getARCoeffs[ar_] :=
  {ar[[1]], ar[[2]], Coefficient[ar[[3, 1]], x, 0], Coefficient[ar[[3, 1]], x, 1]};
```

In[1465]:=
```
getARCoeffs[cosAlphaRatios[[1]]]
```

Out[1465]=
```
{1.1, 4, -0.328849, 0.671151}
```

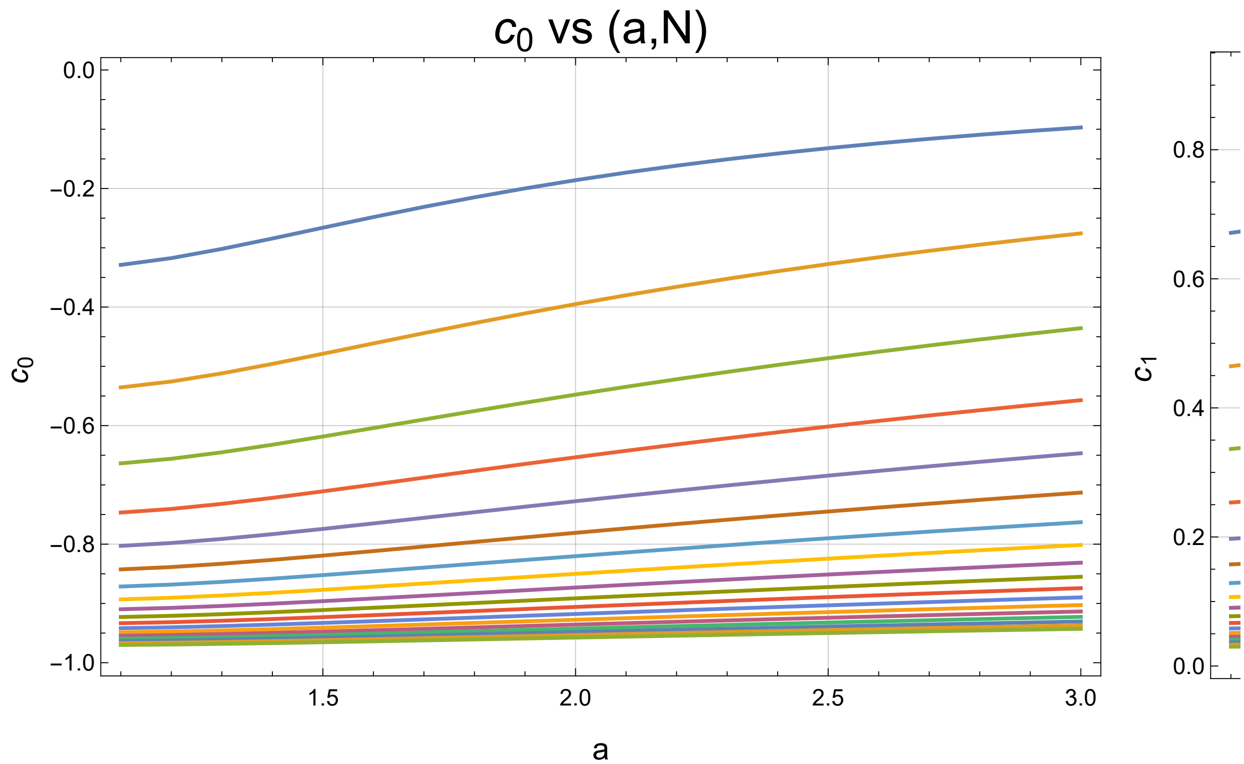
In[1466]:=
```
Module[{uniqueNs, groups},
 groups = GroupBy[getARCoeffs /@ cosAlphaRatios, #[[1]] &];
 (* grouping by a *)
 uniqueNs = Take[Keys[groups], All];
 ListPlot[
  Table[{#[[3]], #[[4]]} & /@ groups[uniqueNs[[i]]], {i, Length@uniqueNs}],
  PlotStyle → Thick, GridLines → Automatic,
  (*Epilog→{Text[Style["N="<>ToString@#,16],
      {2.0,Mean[#[[3,1]]&/@groups[#]]},{0,-1}]&/@uniqueNs},*)
  Frame → True, ImageSize → Large, FrameStyle → Directive[Black, Medium],
  FrameLabel → (Style[#, 16] & /@ {"c₀", "c₁"}),
  PlotLegends → uniqueNs,
  PlotLabel → Style["(c₀,c₁) vs (a,N)", {Black, Large}]]]
```

Out[1466]=



```
In[1467]:= Module[{uniqueNs, groups, plc0, plc1},
  groups = GroupBy[getARCoeffs /@ cosAlphaRatios, #[[2]] &];
  (* grouping by n *)
  uniqueNs = Keys[groups];
  plc0 = ListLinePlot[
    Table[{#[[1]], #[[3]]} & /@ groups[uniqueNs[[i]]], {i, Length@uniqueNs}],
    PlotStyle → Thick, GridLines → Automatic,
    (*Epilog→{Text[Style["N="<>ToString@#,16],
         {2.0,Mean[#[[3,1]]&/@groups[#]]},{0,-1}]&/@uniqueNs},*)
    Frame → True, ImageSize → Large, FrameStyle → Directive[Black, Medium],
    FrameLabel → (Style[#, 16] & /@ {"a", "c0"}),
    (*PlotLegends→uniqueNs,*)
    PlotLabel → Style["c0 vs (a,N)", {Black, Large}]];
  plc1 = ListLinePlot[
    Table[{#[[1]], #[[4]]} & /@ groups[uniqueNs[[i]]], {i, Length@uniqueNs}],
    PlotStyle → Thick, GridLines → Automatic,
    (*Epilog→{Text[Style["N="<>ToString@#,16],
         {2.0,Mean[#[[3,1]]&/@groups[#]]},{0,-1}]&/@uniqueNs},*)
    Frame → True, ImageSize → Large, FrameStyle → Directive[Black, Medium],
    FrameLabel → (Style[#, 16] & /@ {"a", "c1"}),
    PlotLegends → uniqueNs,
    PlotLabel → Style["c1 vs (a,N)", {Black, Large}]];
  Grid[{{plc0, plc1}}]]
```

Out[1467]=

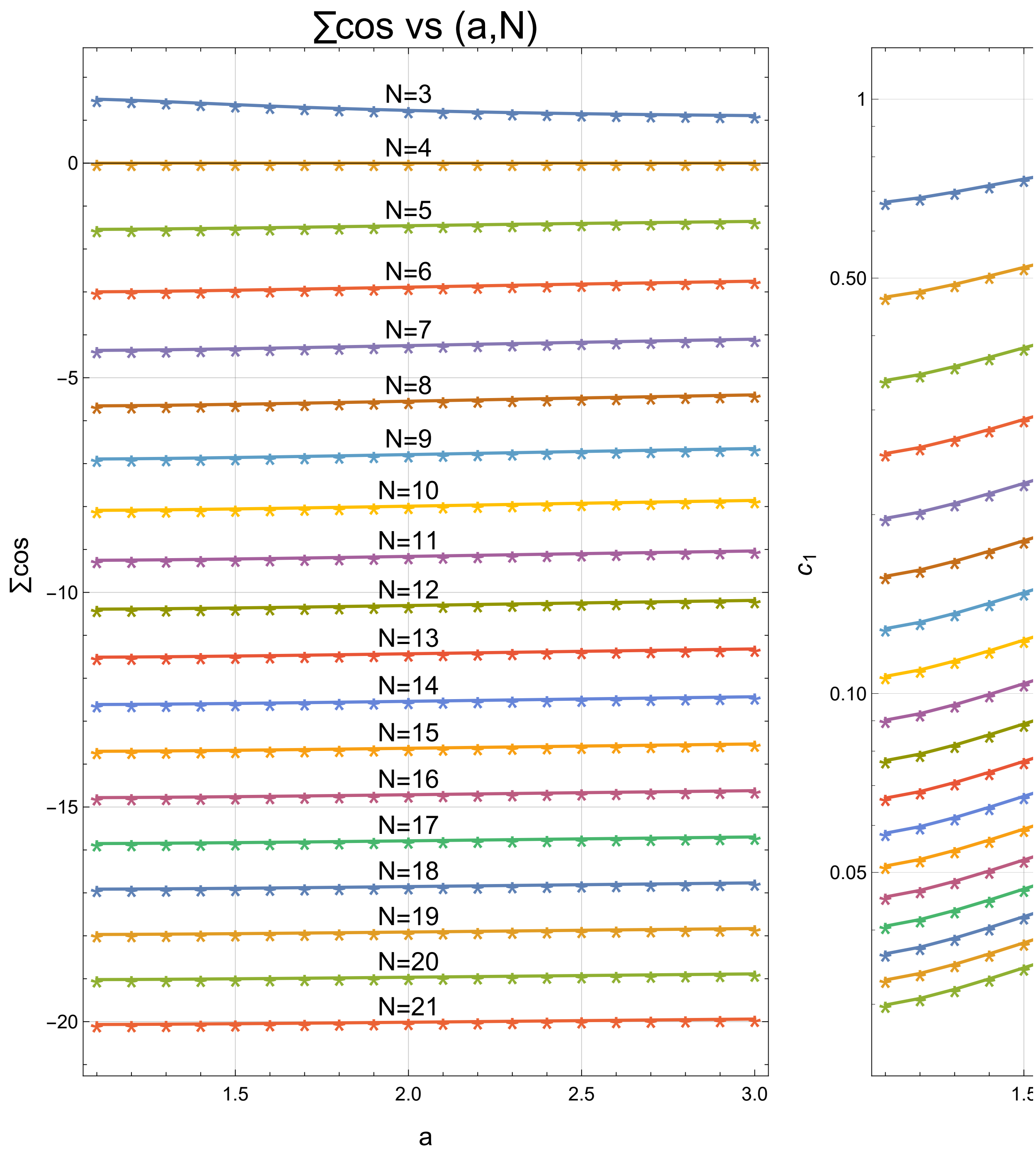


In[1468]:=

```
c1plot = Module[{uniqueNs, groups, plc0, plc1},
  groups = GroupBy[getARCoeffs /@ cosAlphaRatios, #[[2]] &];
  (* grouping by n *)
  uniqueNs = Keys[groups];
  plc1 = ListLogPlot[
    Table[{#[[1]], #[[4]]} & /@ groups[uniqueNs[[i]]], {i, Length@uniqueNs}],
    PlotStyle → Thick, Joined → True, GridLines → Automatic,
    (*Epilog→{Text[Style["N="<>ToString@#,16],
          {2.0,Mean[#[[3,1]]&/@groups[#]]},{0,-1}]&/@uniqueNs},*)
    PlotMarkers → Style["*", 1 Large],
    Epilog → {Text[Style["N=" <> ToString@#, 16],
          {2.0, Log[Mean[#[[4]] & /@ groups[#]]]}, {0, 1}] & /@ uniqueNs},
    Frame → True, ImageSize → Large, FrameStyle → Directive[Black, Medium],
    AspectRatio → 1.5,
    FrameLabel → (Style[#, 16] & /@ {"a", "c₁"}),
    (*PlotLegends→uniqueNs,*)
    PlotLabel → Style["c₁ vs (a,N)", {Black, Large}]];
  plc1]
```

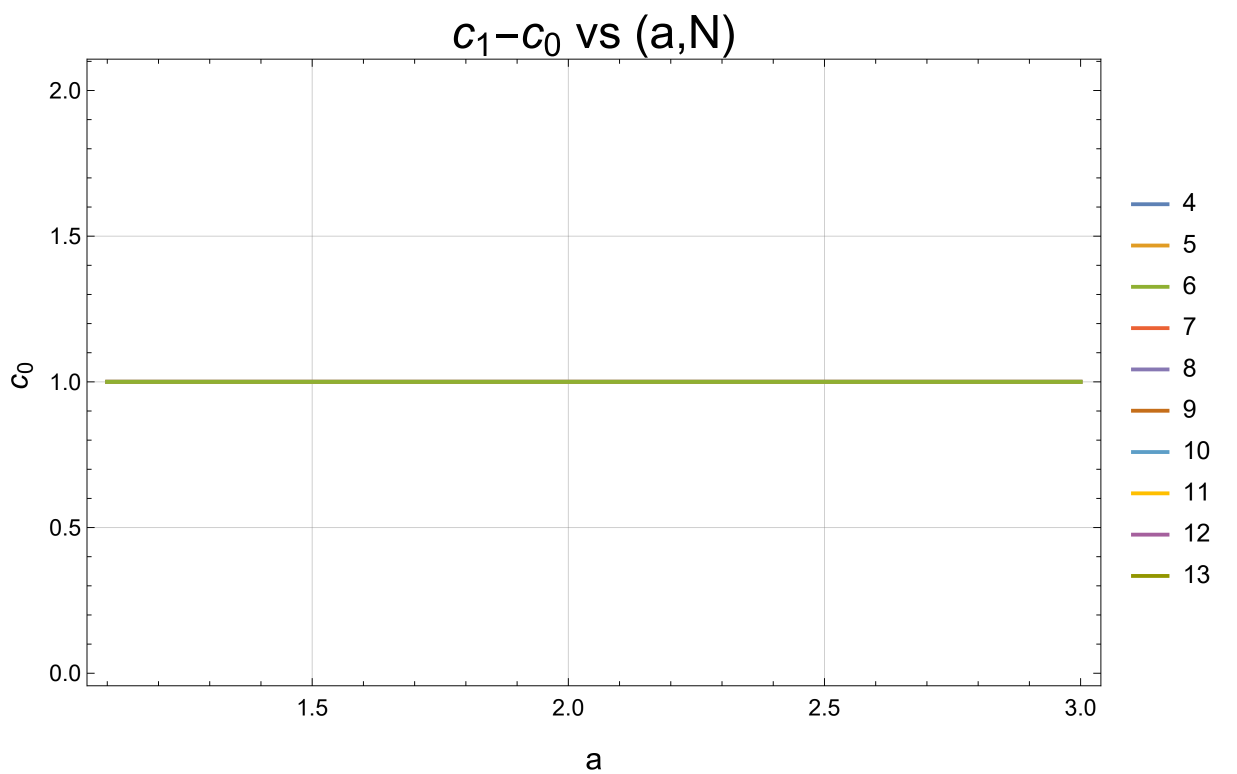
In[1469]:= `Grid[{{constCosPlot, c1plot}}]`

Out[1469]=



In[1470]:=
```
Module[{uniqueNs, groups, plc0, plc1},
 groups = GroupBy[getARCoeffs /@ cosAlphaRatios, #[[2]] &];
 (* grouping by n *)
 uniqueNs = Keys[groups];
 plc0 = ListLinePlot[Table[
     {#[[1]], #[[4]] - #[[3]]} & /@ groups[uniqueNs[[i]]], {i, Length@uniqueNs}],
   PlotStyle → Thick, GridLines → Automatic, (*Epilog→
     {Text[Style["N="<>ToString@#,16],{2.0,Mean[#[[3,1]]&/@groups[#]]},{0,-1}]&/@
       uniqueNs},*)
   Frame → True, ImageSize → Large, FrameStyle → Directive[Black, Medium],
   FrameLabel → (Style[#, 16] & /@ {"a", "c0"}),
   PlotLegends → uniqueNs,
   PlotLabel → Style["c1-c0 vs (a,N)", {Black, Large}]];
 plc0]
```

Out[1470]=



Hyperbola

```
hypC[a_, b_] := a^2 + b^2;
hypEqn[a_, b_, {x_, y_}] := (x / a) ^2 - (y / b) ^2 == 1
hypGrad[a_, b_, {x_, y_}] := -{x b^2, -y a^2};
```

In[1185]:=
```
hypParam[a_, b_, t_] := Module[{x, y},
   x = a Cosh[t];
   y = b Sinh[t];
   {{x, y}, {-x, -y}}];
```

In[1186]:=
```
FullSimplify[{x, y} /.
   Solve[{hypEqn[a, b, {x, y}], hypGrad[a, b, {x, y}].{px - x, py - y} == 0}, {x, y}],
  a > 0 && px ∈ Reals && py ∈ Reals]
```

Out[1186]=

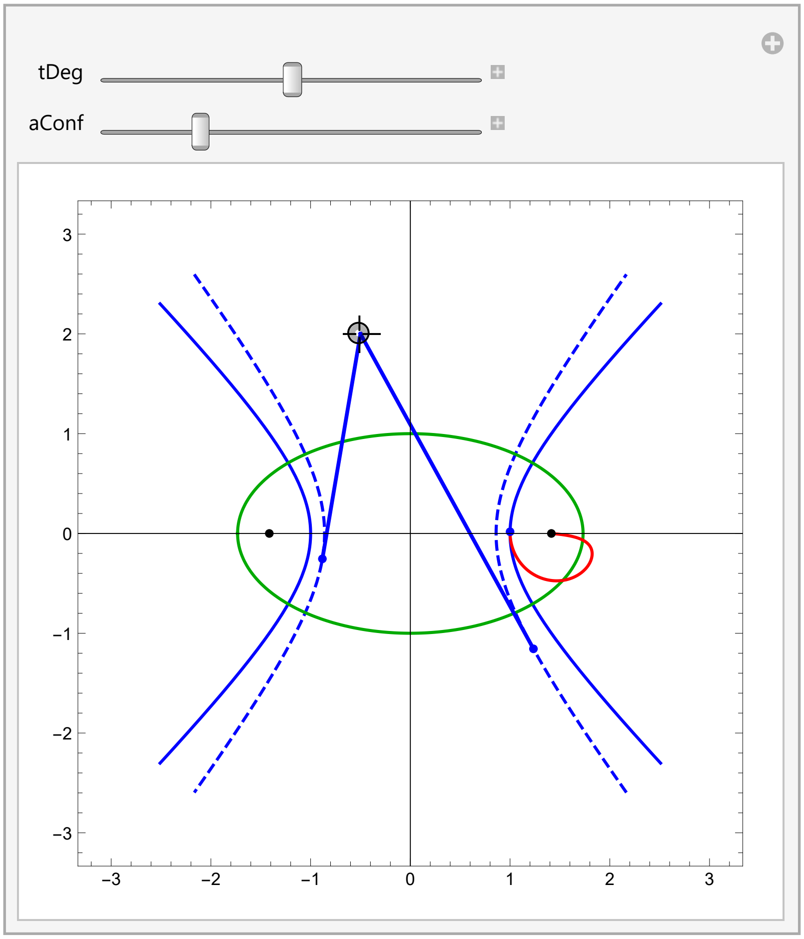
$$\left\{\left\{-\frac{a^2\left(b^2\,px+\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{-b^2\,px^2+a^2\,py^2},\ \frac{b^2\left(a^2\,py^2+px\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{b^2\,px^2\,py-a^2\,py^3}\right\},\right.$$
$$\left.\left\{\frac{a^2\left(b^2\,px-\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{b^2\,px^2-a^2\,py^2},\ \frac{b^2\left(a^2\,py^2-px\sqrt{-b^2\,px^2+a^2\left(b^2+py^2\right)}\;\mathrm{Abs}[py]\right)}{b^2\,px^2\,py-a^2\,py^3}\right\}\right\}$$

In[1187]:=
```
Clear@hypTangentsAB;
hypTangentsAB = Compile[{{a, _Real}, {b, _Real}, {p, _Real, 1}},
   Module[{a2, b2, px, py, px2, px3, py2, py3, radicand, numFact, denomx, denomy},
    {px, py} = p;
    a2 = a * a;
    b2 = b * b;
    px2 = px * px; py2 = py * py;
    px3 = px * px2; py3 = py * py2;
    denomx = b2 px2  - a2 py2;
    denomy = b2 px2 py - a2 py3;
    radicand =  -b2 px2 + a2 (py2 + b2);
    numFact = Sqrt[radicand] * Abs[py];
    Reverse@{
      {a2 safeDiv[b2 px + numFact , denomx], b2 safeDiv[a2 py2 + px numFact, denomy]},
      {a2 safeDiv[b2 px - numFact, denomx], b2 safeDiv[a2 py2 - px numFact, denomy]}}
   ]];
```

In[1189]:=
```
drawTangs[p_, tangs_, clr_] := {PointSize@Medium, clr,
   Point@tangs, Thick, Line[{p, tangs[[1]]}], Line[{p, tangs[[2]]}]};

drawHyp[a_, b_, ps_: {Dashed, Blue}] :=
  ParametricPlot[hypParam[a, b, t], {t, -π/2, π/2}, PlotStyle → ps];
```

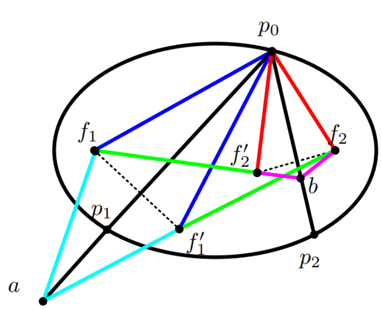
In[1192]:=
```
Module[{a = 1, b = 1, ellB = 1, maxX = 3, bConf, ppHyp, ppEll,
  ppHypConf, hypLocus, ellA, ellTangs, hypTangs, pt, gr, c2, c, fs},
 c2 = a^2 + b^2;
 c = Sqrt@c2;
 fs = {{c, 0}, {-c, 0}};
 ellA = Sqrt[c2 + ellB^2];
 ppHyp = ParametricPlot[hypParam[a, b, t], {t, -π/2, π/2}, PlotStyle → Blue];
 ppEll = ParametricPlot[ellPb[ellA, ellB, t], {t, -π, π}, PlotStyle → Darker@Green];
 Manipulate[
  bConf = Sqrt[c2 - aConf^2];
  ppHypConf = drawHyp[aConf, bConf];
  pt = hypParam[a, b, toRad[tDeg]][[1]];
  (*ellTangs=ellTangentsAB[ellA,ellB,ellLoc];*)
  hypTangs = hypTangentsAB[aConf, bConf, hypLoc];
  hypLocus =
   Quiet@ParametricPlot[hypTangentsAB[aLocus, Sqrt[c2 - aLocus^2], pt][[1]],
     {aLocus, 1, c}, PlotStyle → Red];
  gr = Graphics[{PointSize@Medium,
     {Blue, Point@pt},
     {Black, Point@fs}, (*,
     drawTangs[ellLoc,ellTangs,Darker@Green],*)
     drawTangs[hypLoc, hypTangs, Blue]}];
  Show[{ppHyp, ppHypConf, ppEll, hypLocus, gr}, ImageSize → Medium,
   Frame → True, PlotRange → {{-maxX, maxX}, {-maxX, maxX}}],
  {{tDeg, 1}, -90, 90, .01},
  {{aConf, 1.2}, .5, 2, .01}, (*,
  {{ellLoc,{0,2}},Locator},*)
  {{hypLoc, {-.5, 2}}, Locator}]]
```

Out[1192]=



Self-Intersecting N=4 (bowtie/borboleta)

Construction from: https://pdfs.semanticscholar.org/3d98/090d1023f48be37d534682ad989a89cb0042.pdf



Ronaldo : cos do quadrangulo e da borboleta

In[1432]:=
```
cosAlphaQuadSelfInter[a, x1]
```

Out[1432]=
$$\frac{a^2}{\sqrt{\left(-1+a^2\right)\left(a^4+\left(1-a^2\right)x1^2\right)}}$$

In[1433]:=
```
cosAlphaQuadSelfInter[1.5, 0]
```

Out[1433]= 0.894427

In[1434]:=
```
FullSimplify[cosAlphaQuadSelfInter[a, 0], a > 0]
```

Out[1434]=
$$\frac{1}{\sqrt{-1+a^2}}$$

In[1435]:=
```
Select[a /. Solve[cosAlphaQuadSelfInter[a, 0.] == 1, a], negl[Im[#]] &]
```

Out[1435]= {-1.41421, 1.41421}

In[1436]:=
```
Clear@maxX1QuadSelfInter;
maxX1QuadSelfInter[a_] =
 FullSimplify[x1 /. Solve[cosAlphaQuadSelfInter[a, x1] == 1, x1], a > 1]
```

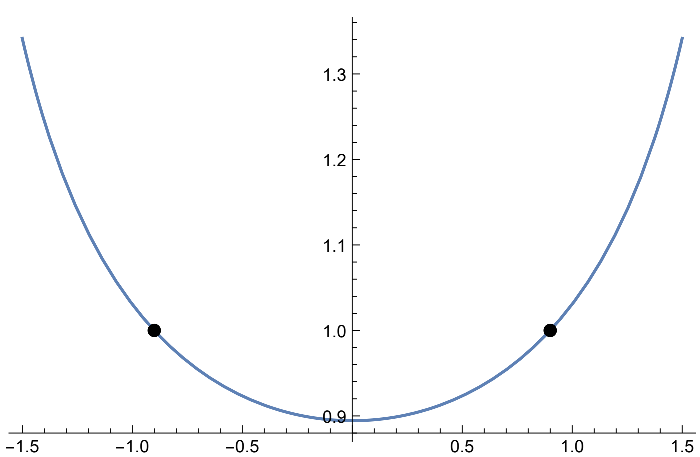
Out[1437]=
$$\left\{\frac{a^2\sqrt{-2+a^2}}{1-a^2},\ \frac{a^2\sqrt{-2+a^2}}{-1+a^2}\right\}$$

In[1438]:=
```
maxX1QuadSelfInter[1.5]
```

Out[1438]= {-0.9, 0.9}

In[1439]:=
```
Module[{a = 1.5, maxX1, maxPs},
 maxX1 = maxX1QuadSelfInter[a];
 maxPs = {#, cosAlphaQuadSelfInter[a, #]} & /@ maxX1;
 Plot[cosAlphaQuadSelfInter[a, x1], {x1, -a, a}, Epilog → {PointSize@Large,
    Point /@ maxPs}]]
```

Out[1439]=



In[1440]:=
```
Module[{ca, p1, n1, sa, n1rot, p2},
 Quiet[
  ca = cosAlphaQuadSelfInter[a, x1];
  p1 = {x1, ellY[a, x1]};
  n1 = ellGrad[a, Sequence @@ p1];
  sa = Sqrt[1 - ca^2];
  n1rot = rot[n1, -sa, ca];
  p2 = ellInterRayUnprot[a, p1, n1rot][[2]];
 ];
 x1 /. Solve[{FullSimplify[p1[[1]] - p2[[1]] == 0, a > 0 && x1 < a && -1 < ca < 1],
    a > 1, x1 < a}, x1, Reals]]
```

Out[1440]= $\left\{\text{ConditionalExpression}\left[-\sqrt{\frac{a^4}{-1+a^2}}, 1<a<\sqrt{2} \,||\, a>\sqrt{2}\right], \text{ConditionalExpression}\left[-\sqrt{\frac{-2a^2+a^4}{-1+a^2}}, a>\sqrt{2}\right]\right\}$

In[1441]:= $\text{x1StraightDownQuadSelfInter[a\_]} := \left\{-\sqrt{\frac{a^4}{-1+a^2}}, -\sqrt{\frac{-2a^2+a^4}{-1+a^2}}\right\}$

In[1442]:=
```
x1StraightDownQuadSelfInter[1.5]
```

Out[1442]= {-2.01246, -0.67082}

In[1443]:=
```
Clear@solVorbit; solVorbit[a_] = Module[{n, p, y, nperp, vtop, eqn},
  y = -Quiet@ellY[a, x];
  n = Quiet@ellGrad[a, x, y];
  vtop = {0, 1};
  p = {x, y};
  nperp = perp[n];
  eqn = (nperp.(p - vtop) == 0);
  First@Normal[x /. Solve[eqn && a > 0 && 0 < x < a, x, Reals]]];
```

In[1444]:=
```
Clear@solXorbit; solXorbit = Quiet@Module[{n, p1, p3, eqn},
   p1 = {x, Quiet@ellY[a, x]};
   p3 = FullSimplify[getInterReflNonComp[a, p1, -p1], a > 0 && (-a < x < 0)];
   eqn = FullSimplify[(p3 + p1)[[1]] == 0, a > 0 && (-a < x < 0)];
   FullSimplify[x /. Solve[eqn, x], a > 0]]
```

Out[1444]= $\left\{-a\sqrt{\frac{-2+a^2}{-1+a^2}}, a\sqrt{\frac{-2+a^2}{-1+a^2}}\right\}$

In[1445]:=
```
solXorbit /. {a → 1.5}
```

Out[1445]= {-0.67082, 0.67082}

In[1446]:= solXorbit[a, x]

Out[1446]= $\left\{-a\sqrt{\frac{-2+a^2}{-1+a^2}}, a\sqrt{\frac{-2+a^2}{-1+a^2}}\right\}[a, x]$

```
Clear@getQuadSelfInter;
getQuadSelfInter[a_, p1_] := Module[{ca, sa, bounce, p2, n1, n1rot, fs, gr},
   ca = cosAlphaQuadSelfInter[a, p1[[1]]];
   n1 = ellGrad[a, Sequence @@ p1];
   sa = Sqrt[1 - ca^2];
   n1rot = rot[n1, -sa, ca];
   p2 = ellInterRayUnprot[a, p1, n1rot][[2]];
   bounce = bounceRay[a, p1, p2, 3];
   bounce];
```

In[1449]:= hypEqn[a, b, {x, y}]

Out[1449]= $\frac{x^2}{a^2} - \frac{y^2}{b^2} == 1$

In[1450]:= FullSimplify[Solve[hypEqn[a, b, {x, 0}], a, Reals]]

Out[1450]= $\left\{\left\{a \to -\sqrt{x^2}\right\}, \left\{a \to \sqrt{x^2}\right\}\right\}$

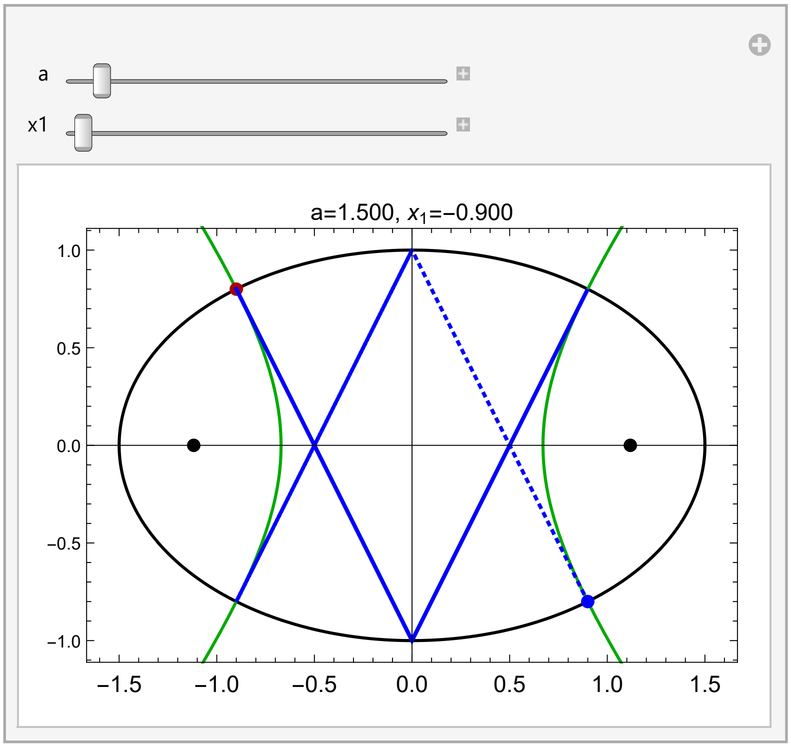
In[1451]:= Solve[hypEqn[a, b, {0, y}], b]

Out[1451]= $\{\{b \to -i\, y\}, \{b \to i\, y\}\}$

```
In[1452]:= Clear@showQuadSelfInter;
showQuadSelfInter[theA_, x1_] := Module[{ca, sa, bounce, c2, p1,
    p2, n1, n1rot, dqx, dq, bounceDq, aConf, bConf, hyp, solX, fs, gr},
   p1 = {x1, ellY[theA, x1]};
   bounce = If[theA < Sqrt[2], {p1}, getQuadSelfInter[theA, p1]];
   dqx = solVorbit[theA];
   dq = {dqx, -ellY[theA, dqx]};
   bounceDq = bounceRay[theA, dq, {0, 1}, 2];
   fs = getFoci[theA];
   solX = solXorbit[[1]] /. {a → theA};
   aConf = Abs@solX;
   c2 = theA^2 - 1; (*ellipse*)
   (* for hyp: c2=a2+b2 => b=sqrt(c2-a2) *)
   bConf = Sqrt[c2 - aConf^2];
   hyp = drawHyp[aConf, bConf, Darker@Green];
   gr = Graphics[{PointSize@Large, FaceForm@None, Thick,
      {Point@fs, Darker@Red, Point@p1},
      {Blue, Point@dq, Dotted, Line@bounceDq},
      (*{Darker@Green,Point@{solX,0}},*)
      {Blue, Line@bounce}}];
   Show[{plotEll[theA], hyp, gr}, Frame → True,
    PlotLabel → Style["a=" <> nfn[theA, 3] <> ",  x₁=" <> nfn[x1, 3], {Black, Medium}],
    FrameStyle → {Black, Medium}]];
```

In[1454]:=
```
Module[{x1max = maxX1QuadSelfInter[1.5][[2]]},
 Manipulate[
  showQuadSelfInter[a, x1],
  {{a, 1.5}, Sqrt[2.], 3, .001},
  {{x1, 0}, -x1max, x1max, .01}]]
```

Out[1454]=



In[1022]:=
```
Clear@getQuadSelfInterCosines;
getQuadSelfInterCosines[a_, x1_] := Module[{p1, bounce},
  p1 = {x1, ellY[a, x1]};
  bounce = getQuadSelfInter[a, p1];
  (*Print@Chop@bounce;*)
  getPolyCosines[Most@bounce]];
```

In[1023]:=
```
Prepend[#, 1.5] & /@ Take[getQuadSelfInterCosines[1.5, 0], 2]
```

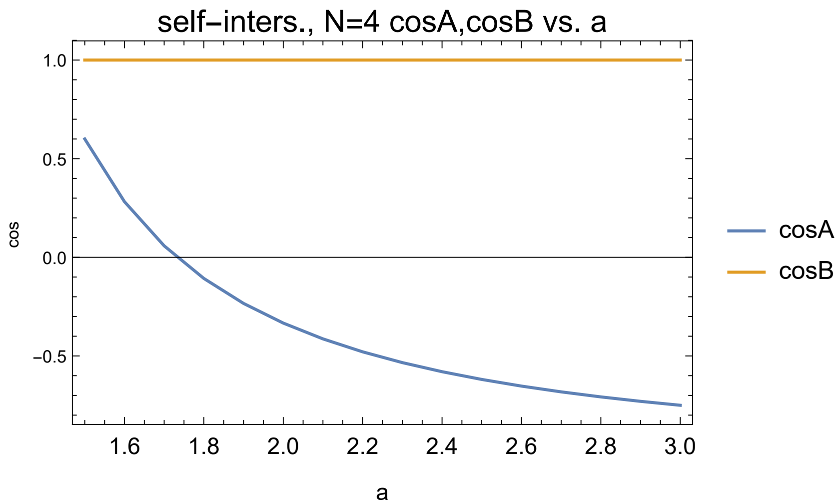
Prepend::normal : Nonatomic expression expected at position 1 in Prepend[0.6, 1.5]. ≫

Prepend::normal : Nonatomic expression expected at position 1 in Prepend[1., 1.5]. ≫

Out[1023]=
```
{Prepend[0.6, 1.5], Prepend[1., 1.5]}
```

In[1024]:=
```
ListLinePlot[
 Transpose@Table[{a, #} & /@ Take[getQuadSelfInterCosines[a, 0], 2], {a, 1.5, 3, .1}],
 Frame → True, FrameLabel → {"a", "cos"}, FrameStyle → {Black, Medium},
 PlotLegends → {"cosA", "cosB"}, PlotStyle → Automatic,
 PlotLabel → Style["self-inters., N=4 cosA,cosB vs. a", {Black, 16}]]
```

Out[1024]=



In[1025]:=

In[1026]:=
```
getStats[Module[{a = #, x1max, x1StrDown},
    x1max = maxX1QuadSelfInter[a][[2]];
    Table[Total@getQuadSelfInterCosines[a, x1],
     {x1, -x1max, x1max, 2 x1max / 1000}]]] & /@ {1.5, 2.0, 3.0} // ColumnForm
```

Out[1026]=
```
{3.2, 4.51641 × 10^-14, 1.41138 × 10^-14, 3.2, 3.2, 3.2, 1001}
{1.33333, 6.11394 × 10^-15, 4.58546 × 10^-15, 1.33333, 1.33333, 1.33333, 1001}
{0.5, 1.11386 × 10^-15, 2.22773 × 10^-15, 0.5, 0.5, 0.5, 1001}
```

Ronaldo : orbita dupla tem a soma de cossenos :

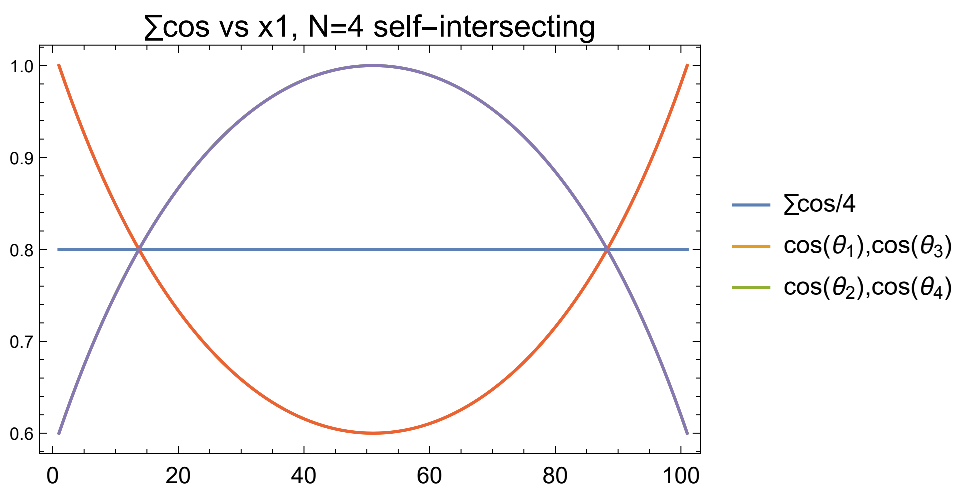
In[1027]:=
```
somaCossenosBorboleta[a_] := 4 (a^2 - 2) / (a^2 - 1);
```

In[1028]:=
```
somaCossenosBorboleta /@ {1.5, 2.0, 3.0}
```

Out[1028]=
```
{0.8, 2.66667, 3.5}
```

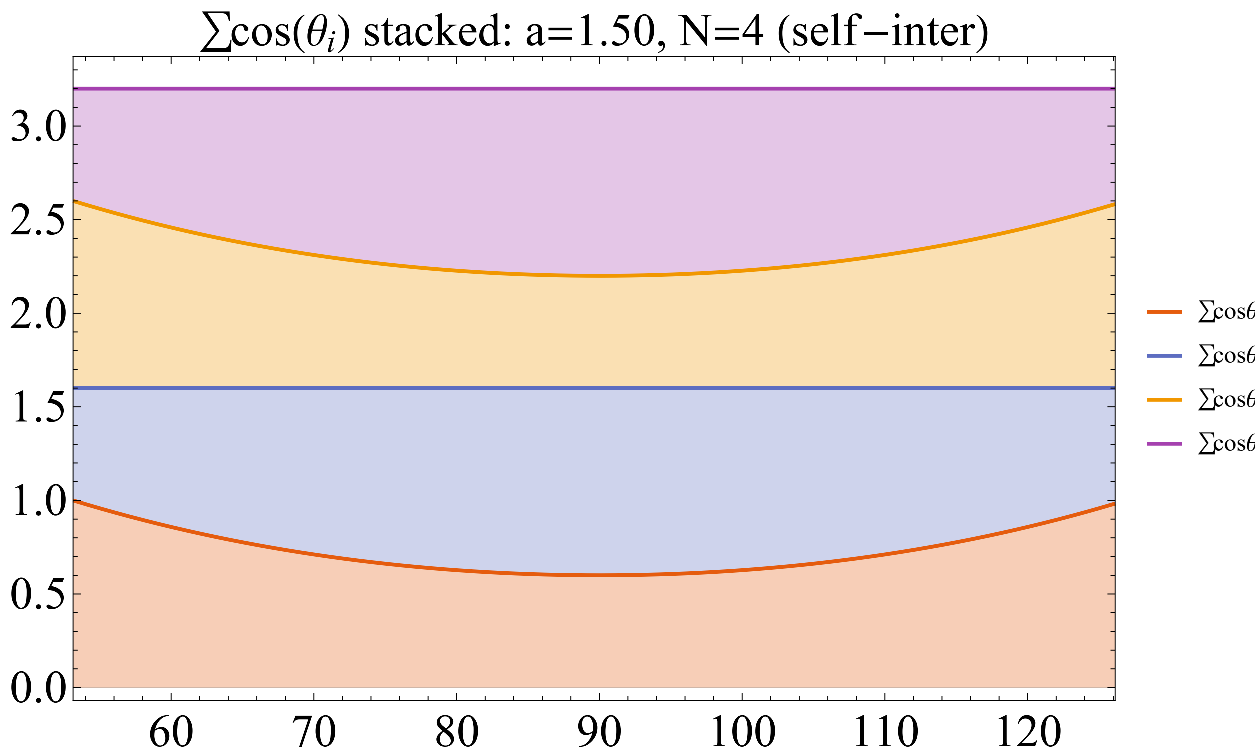
In[1029]:=
```
Module[{a = 1.5, x1max, x1StrDown, cosTab, cosSum, x1s},
 x1max = maxX1QuadSelfInter[a][[2]];
 x1s = Range[-x1max, x1max, x1max / 50];
 cosTab = getQuadSelfInterCosines[a, #] & /@ x1s;
 cosSum = Total /@ cosTab;
 ListLinePlot[Transpose@MapThread[Prepend[#1, #2 / 4] &, {cosTab, cosSum}],
  PlotLegends → {"∑cos/4", "cos(θ1),cos(θ3)", "cos(θ2),cos(θ4)"},
  Frame → True, FrameStyle → {Black, Medium},
  PlotLabel → Style["∑cos vs x1, N=4 self-intersecting", {Black, 16}]]]
```

Out[1029]=



In[1966]:=
```
Module[{a = 1.5, x1max, x1StrDown, coss, cosSum, x1s, tmin, tmax, ts, cossAcc},
 x1max = maxX1QuadSelfInter[a][[2]];
 x1s = Range[-x1max, x1max, x1max / 50];
 tmin = ArcCos[x1max / a];
 tmax = ArcCos[-x1max / a];
 ts = Range[tmin, tmax, toRad[1.]];
 coss = getQuadSelfInterCosines[a, a Cos[#]] & /@ ts;
 cosineSumStackedPlotLowLevel[a, 4, coss,
  Range[toDeg@tmin, toDeg@tmax, 1], note -> " (self-inter)"]]
```

Out[1966]=

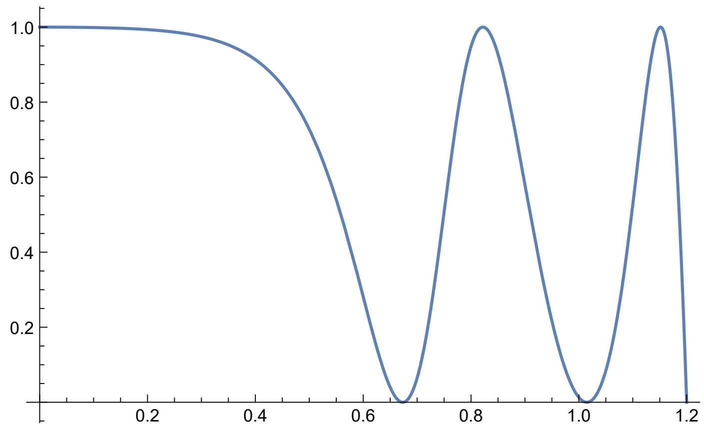


Self-Intersecting N=5 (pentagram)

In[1678]:=
```
Clear@pentagramErrCaustic;
pentagramErrCaustic[a_, x1_] := Module[{p1, p2, p3, p4},
   p1 = {x1, -ellY[a, x1]};
   p2 = flipY[p1];
   p3 = getInterRefl[a, p1, p2];
   p4 = getInterRefl[a, p2, p3];
   p4[[2]]^2
  ];
```

In[1680]:=
```
Plot[pentagramErrCaustic[1.2, x1], {x1, 0, 1.2}, PlotRange → All]
```

Out[1680]=



In[1681]:=
```
pentagramX1 = x1 /. Quiet@Second@FindMinimum[pentagramErrCaustic[1.2, x1], {x1, .6}]
```

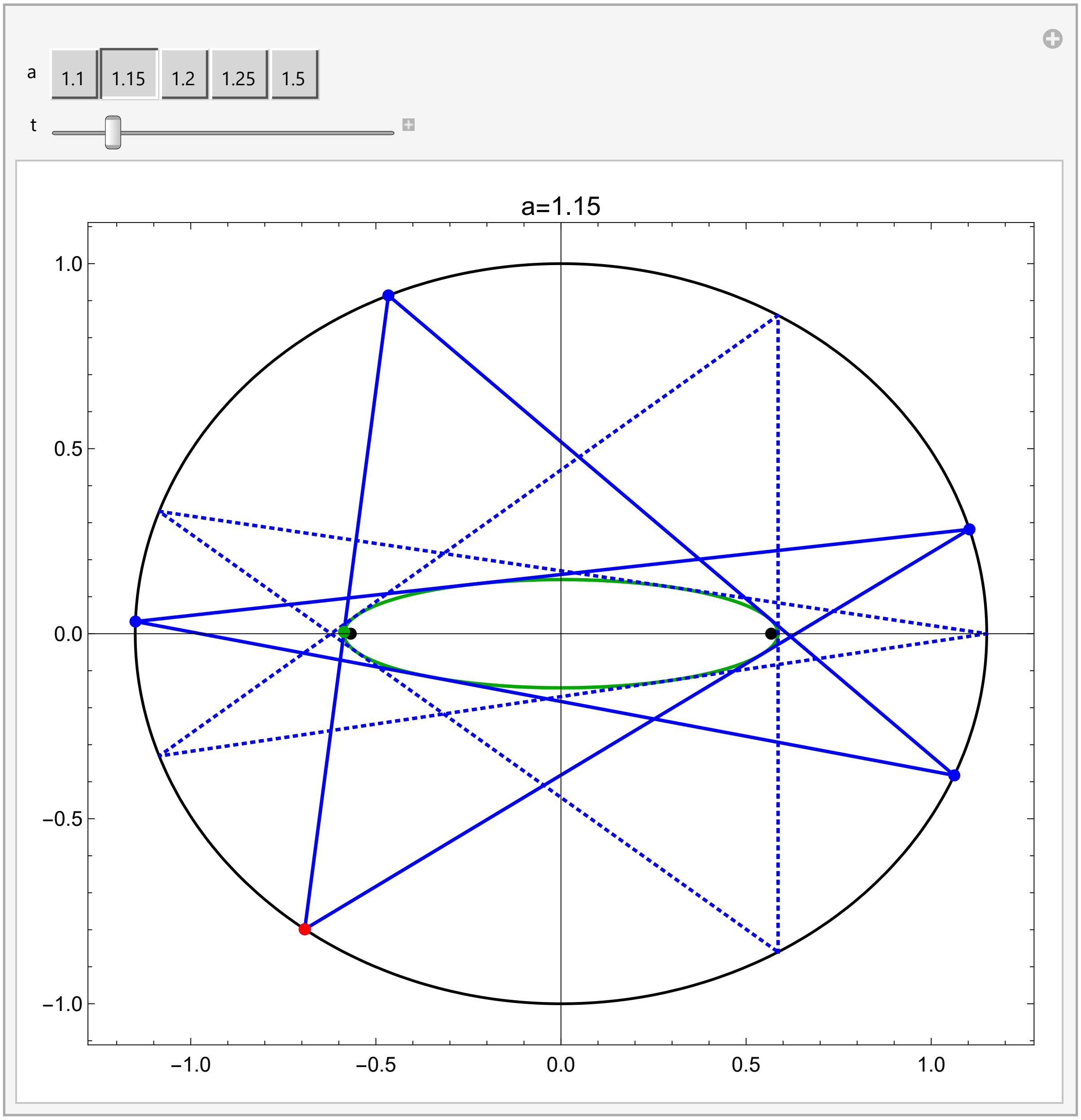
Out[1681]=
```
0.673304
```

In[1682]:=
```
pentagramMins = <|1.1 → .5, 1.15 → .6, 1.2 → .6, 1.25 → .7, 1.5 → 1.1|>;
```

In[1938]:=
```
getPentagrams[a_] :=
  Module[{x0, y0, gr, p1, p2, causticAB, pstart, pend, tRad, tangCW, bounceT},
   x0 = x1 /. Quiet@Second@FindMinimum[
        pentagramErrCaustic[a, x1], {x1, pentagramMins[a]}];
   y0 = ellY[a, x0];
   p1 = {x0, -y0};
   p2 = flipY@p1;
   causticAB = getCausticAxes[a, p1, p2];
   Table[
    tRad = toRad[tDeg];
    pstart = {a Cos@tRad, Sin@tRad};
    (* needs to get the correct tang &&& *)
    tangCW = getTangCW[a, pstart, Sequence @@ causticAB];
    (*tangs=ellTangentsAB[Sequence@@causticAB,pstart];*)
    pend = ellInterRayUnprot[a, pstart, tangCW - pstart][[2]];
    bounceT = bounceRay[a, pstart, pend, 4];
    bounceT, {tDeg, 0, 359}]];
```

```
In[1951]:= Manipulate[
  DynamicModule[{bounce, x0, y0, gr, p1, p2, fs,
    causticAB, ell, ellCaustic, pstart, pend, tRad, tangCW, bounceT},
   x0 = x1 /. Quiet@Second@FindMinimum[
        pentagramErrCaustic[a, x1], {x1, pentagramMins[a]}];
   y0 = ellY[a, x0];
   p1 = {x0, -y0};
   p2 = flipY@p1;
   bounce = bounceRay[a, p1, p2, 4];
   ell = plotEll[a];
   causticAB = getCausticAxes[a, p1, p2];
   ellCaustic = plotEllb[Sequence @@ causticAB, {Thick, Darker@Green}];
   fs = getFoci[a];
   Dynamic[tRad = toRad[t];
    pstart = {a Cos@tRad, Sin@tRad};
    (*tangs=ellTangentsAB[Sequence@@causticAB,pstart];*)
    tangCW = getTangCW[a, pstart, Sequence @@ causticAB];
    pend = ellInterRayUnprot[a, pstart, tangCW - pstart][[2]];
    bounceT = bounceRay[a, pstart, pend, 4];
    gr = Graphics[{PointSize@Large, EdgeForm@None, Thick,
       {Black, Point@fs},
       {Blue, Dotted, Line@bounce},
       {Blue, Line@bounceT, Point@bounceT},
       {Darker@Green, PointSize@Large, Point@tangCW},
       {Red, Point@pstart}}];
    Show[{ell, ellCaustic, gr}, PlotLabel → Style["a=" <> nfn[a, 2], {Black, 15}],
     Frame → True, FrameStyle → Medium, ImageSize → Large]]],
  {{a, 1.15}, Keys@pentagramMins},
  {{t, 30}, -180, 180, 1}, SynchronousUpdating → False]
```

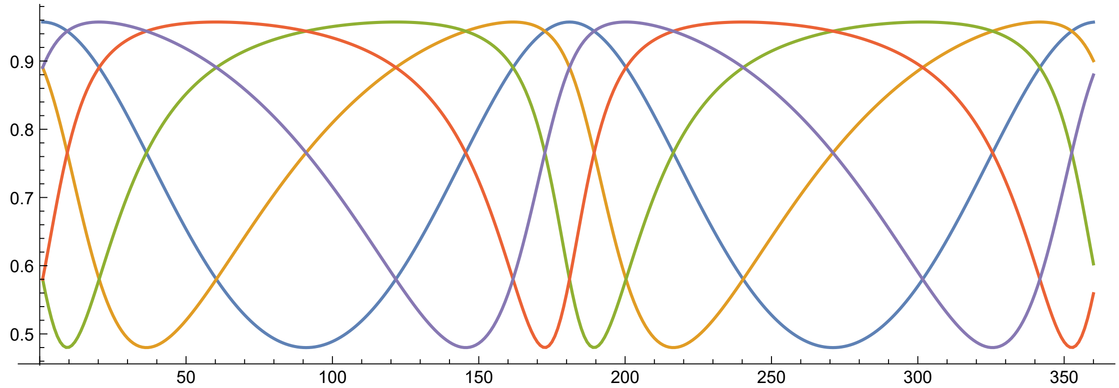
Out[1951]=



In[1940]:=
```
Clear@getPentagramCosines;
getPentagramCosines[a_] := Module[{},
   If[MemberQ[Keys@pentagramMins, a],
    getPolyCosines /@ Most /@ getPentagrams[a]]];
```

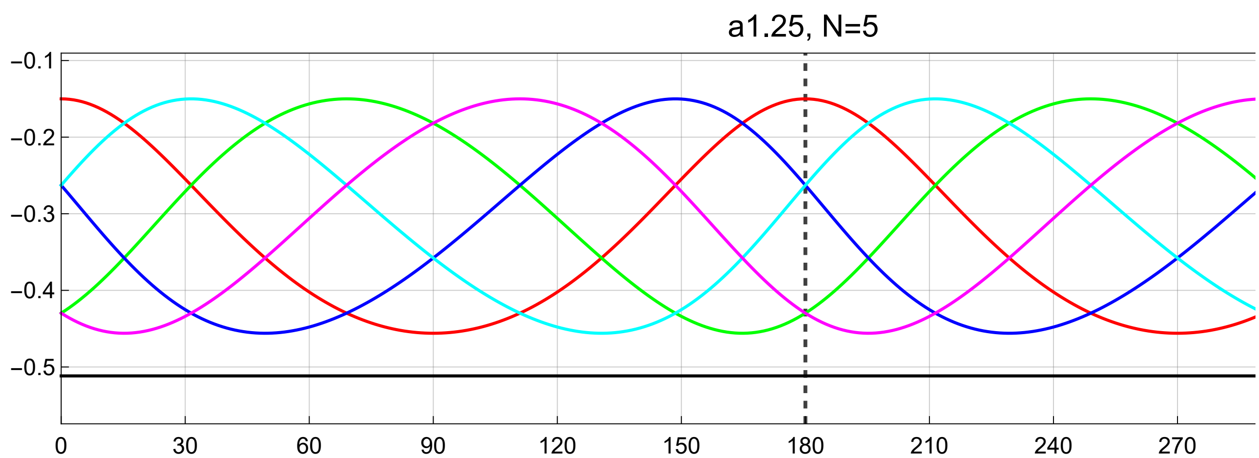
In[1942]:=
```
ListLinePlot[Transpose@getPentagramCosines[1.15],
 AspectRatio → .33, ImageSize → Large]
```

Out[1942]=



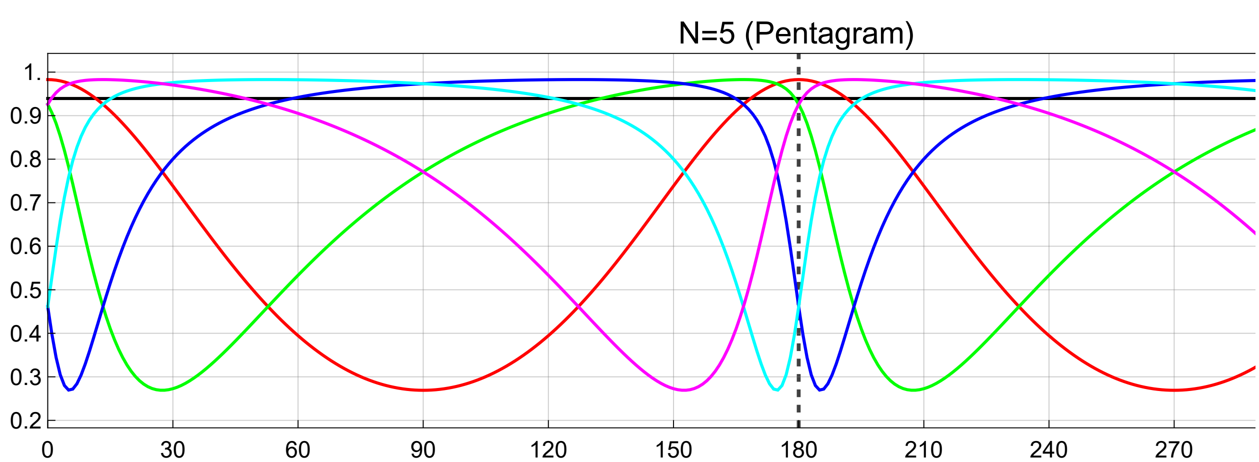
In[1943]:=
```
Show[plotPolyCos[pentAlphaT125, getPentVtx0, pert → 0, cosDiv → 3],
 PlotRange → {All, {-.55, -.1}}]
```

Out[1943]=



In[1944]:=
```
Show[plotPolyCos[{}, {}, polys0 → (Most /@ getPentagrams[1.25]),
   ts0 → Range[0, 359], cosDiv → 4, pert → 0],
 PlotLabel → Style["N=5  (Pentagram)", {Black, 16}], PlotRange → {All, {.2, 1}}]
```

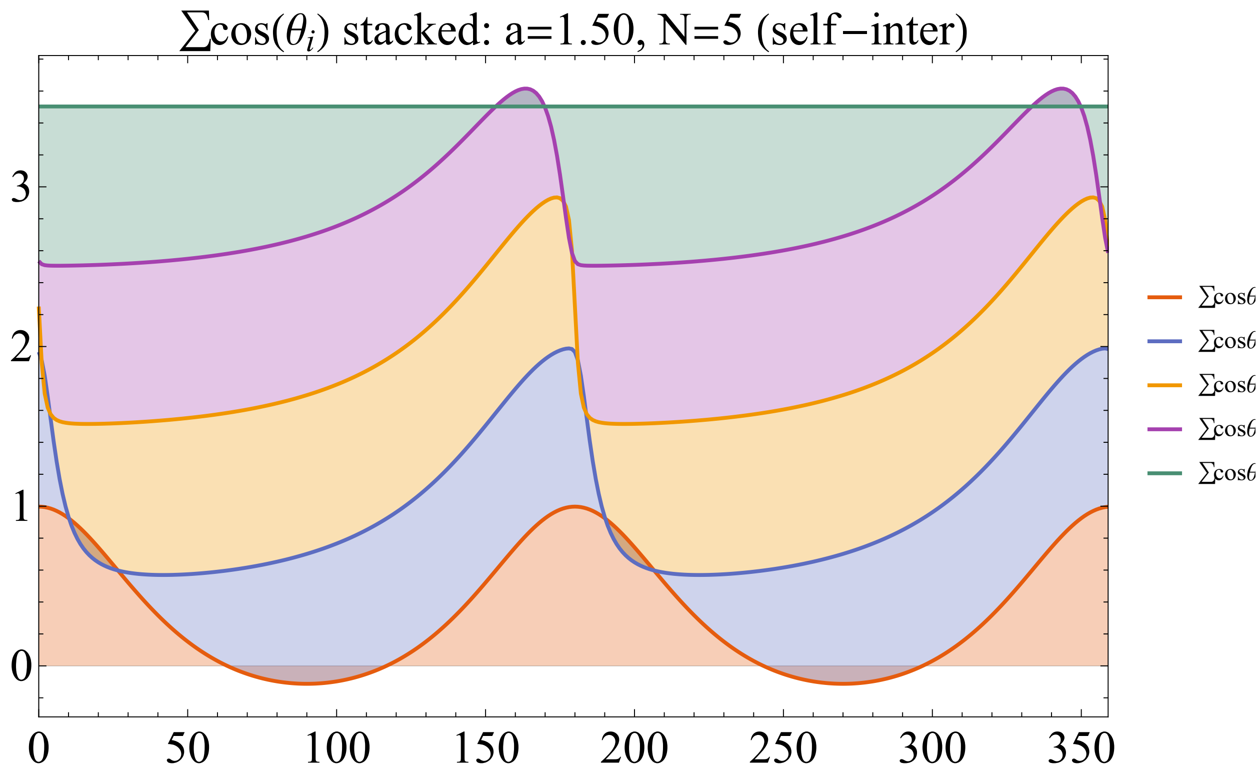
Out[1944]=



In[1695]:=
```
getPentagramCosines[1.15]
```

In[1967]:=
```
Module[{a = 1.5, x1max, x1StrDown, coss, cosSum, x1s, tmin, tmax, ts, cossAcc},
 coss = getPentagramCosines[a];
 cosineSumStackedPlotLowLevel[
  a, 5, coss, Range[0, 359, 1], note -> "  (self-inter)"]]
```

Out[1967]=



In[1690]:=
```
First[getPentagramCosines[1.15]]
```

Out[1690]= {0.957215, 0.890614, 0.580133, 0.580133, 0.890614}

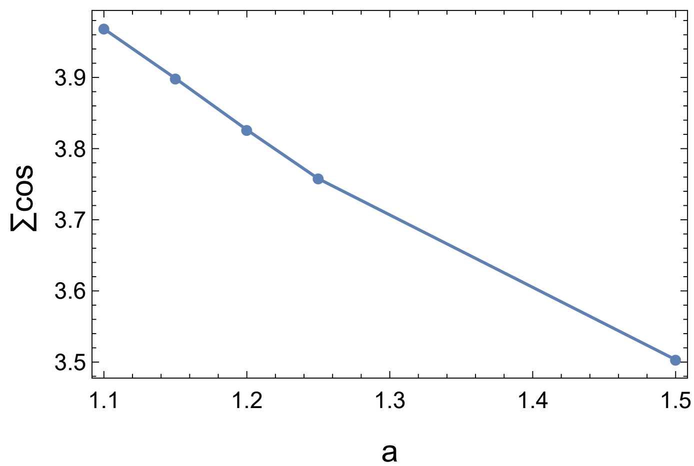
In[1946]:=
```
pentagramCosineStats =
  getStats[Total /@ getPentagramCosines[#]] & /@ Keys@pentagramMins;
pentagramCosineStats // ColumnForm
```

Out[1947]=

$\{3.96836, 1.36471\times10^{-14}, 3.43898\times10^{-15}, 3.96836, 3.96836, 3.96836, 360\}$
$\{3.89871, 1.89843\times10^{-13}, 4.86937\times10^{-14}, 3.89871, 3.89871, 3.89871, 360\}$
$\{3.82645, 2.13989\times10^{-11}, 5.59237\times10^{-12}, 3.82645, 3.82645, 3.82645, 360\}$
$\{3.75772, 5.57328\times10^{-12}, 1.48316\times10^{-12}, 3.75772, 3.75772, 3.75772, 360\}$
$\{3.50329, 4.97287\times10^{-12}, 1.41949\times10^{-12}, 3.50329, 3.50329, 3.50329, 360\}$

In[1948]:=
```
ListLinePlot[Transpose@{Keys@pentagramMins, First /@ pentagramCosineStats},
 PlotMarkers → Automatic, Frame → True, FrameStyle → Medium,
 FrameLabel → (Style[#, {Black, 16}] & /@ {"a", "∑cos"})]
```

Out[1948]=



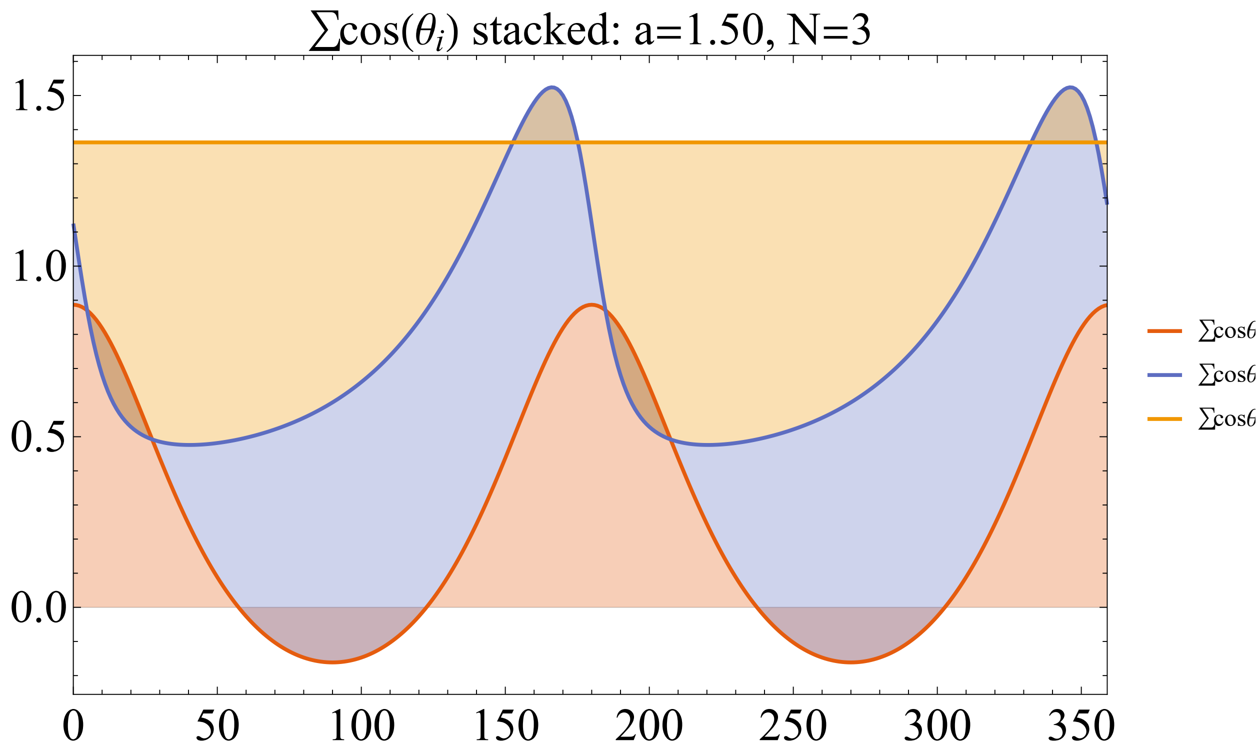
In[1949]:=
```
pentagramFrames[a_, step_] :=
  Module[{bounce, x0, y0, gr, p1, p2, fs, causticAB,
    ell, ellCaustic, pstart, pend, tRad, tangs, bounceT, imgs},
   x0 = x1 /. Quiet@Second@FindMinimum[pentagramErrCaustic[a, x1],
        {x1, pentagramMins[a]}];
   y0 = ellY[a, x0];
   p1 = {x0, -y0};
   p2 = flipY@p1;
   bounce = bounceRay[a, p1, p2, 4];
   ell = plotEll[a];
   causticAB = getCausticAxes[a, p1, p2];
   ellCaustic = plotEllb[Sequence @@ causticAB, {Thick, Darker@Green}];
   fs = getFoci[a];
   imgs = Table[
     tRad = toRad[t];
     pstart = {a Cos@tRad, Sin@tRad};
     tangs = ellTangentsAB[Sequence @@ causticAB, pstart];
     pend = ellInterRayUnprot[a, pstart, tangs[[1]] - pstart][[2]];
     bounceT = bounceRay[a, pstart, pend, 4];
     gr = Graphics[{PointSize@Large, EdgeForm@None, Thick,
        {Black, Point@fs},
        {Blue, Dotted, Line@bounce},
        {Blue, Line@bounceT, Point@bounceT},
        {Darker@Green, PointSize@Medium, Point@tangs},
        {Red, Point@pstart}}];
     Show[{ell, ellCaustic, gr}, PlotLabel →
       Style["a=" <> nfn[a, 2] <> ", θ=" <> ToString@t <> "°", {Black, 15}],
      Frame → True, FrameStyle → Medium, ImageSize → 800], {t, 0, 360, step}];
   imgs];
```

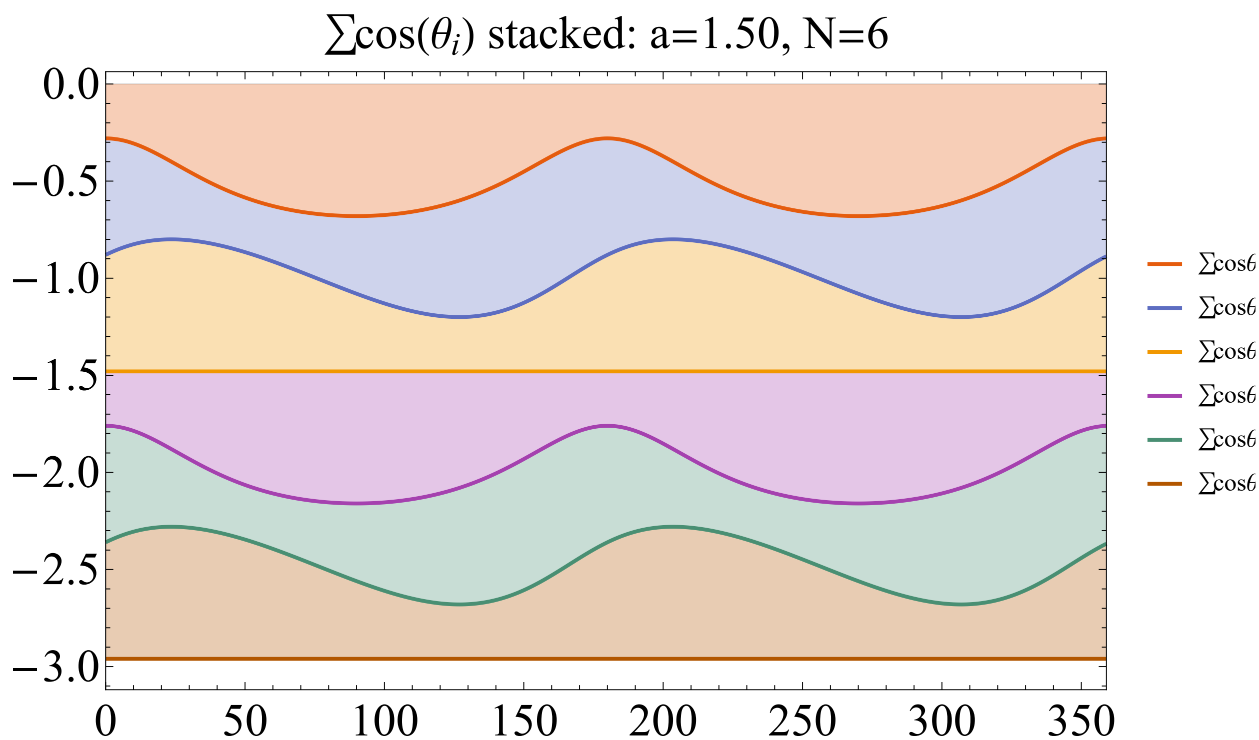
Stacked Cosine Sum Plots

```
In[1968]:= stackedBowtie =
  Module[{a = 1.5, x1max, x1StrDown, coss, cosSum, x1s, tmin, tmax, ts, cossAcc},
   x1max = maxX1QuadSelfInter[a][[2]];
   x1s = Range[-x1max, x1max, x1max / 50];
   tmin = ArcCos[x1max / a];
   tmax = ArcCos[-x1max / a];
   ts = Range[tmin, tmax, toRad[1.]];
   coss = getQuadSelfInterCosines[a, a Cos[#]] & /@ ts;
   cosineSumStackedPlotLowLevel[a, 4, coss,
    Range[toDeg@tmin, toDeg@tmax, 1], note -> " (self-inter)"]];

In[1969]:= stackedPentagram =
  Module[{a = 1.5, x1max, x1StrDown, coss, cosSum, x1s, tmin, tmax, ts, cossAcc},
   coss = getPentagramCosines[a];
   cosineSumStackedPlotLowLevel[
    a, 5, coss, Range[0, 359, 1], note -> " (self-inter)"]];

In[1970]:= stacked3645 = cosineSumStackedPlot[1.5, #, Range[0, 359, 1]] & /@ {3, 6, 4, 5};
```

In[1971]:=
```
Grid[Transpose@Partition[Join[stacked3645, {stackedBowtie, stackedPentagram}], 2]]
```

Out[1971]=

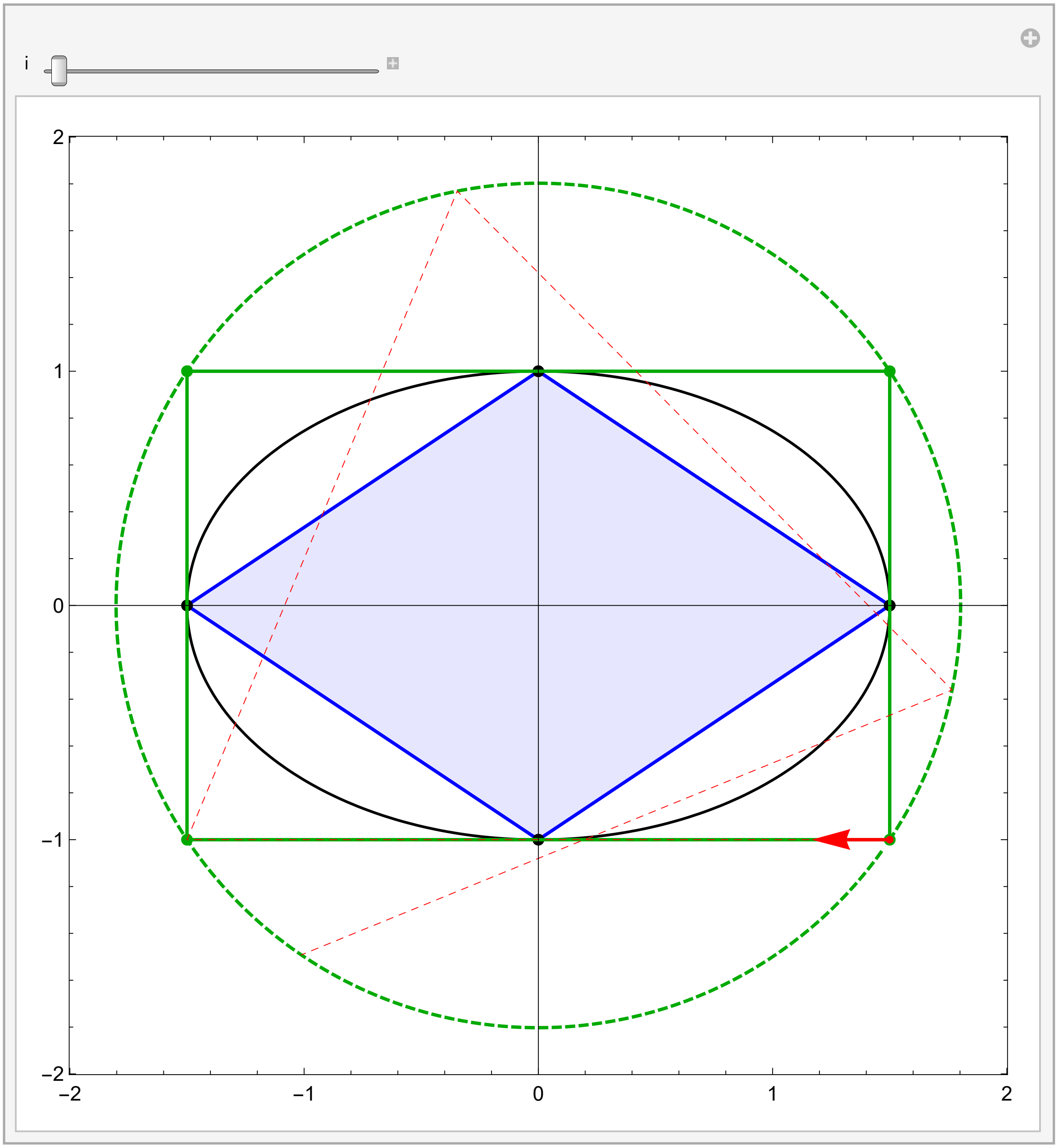




Excentral Loci

In[2241]:=
```
showPolyExcentralLocus;
showPolyExcentralLocus[alphaT_, fnVtx0_] :=
 Module[{a, poly, exc, gr, n, excLocus, excTab, bounce, excMaxX, excMaxY, excPts},
  a = "a" /. alphaT;
  n = Length["alphas" /. alphaT];
  excTab = Table[polyExcentral[alphaT, i, fnVtx0], {i, n}];
  excPts = First /@ (Second /@ excTab);
  excMaxX = Max[First /@ excPts];
  excMaxY = Max[Second /@ excPts];
  Manipulate[
   {poly, exc} = polyExcentral[alphaT, i, fnVtx0];
   bounce = bounceRayAB[excMaxX, excMaxY, exc[[1]], exc[[2]], Length@poly - 1];
   gr = Graphics[{FaceForm@None, PointSize@Large,
      {EdgeForm[{Blue, Thick}], FaceForm@Blue, Opacity@.1, Polygon@poly},
      {Black, Point@poly},
      {EdgeForm[{Darker@Green, Thick}], Polygon@exc},
      {Darker@Green, Point@exc},
      {Darker@Green, Dashed, Thick, Line[excPts]},
      {Red, Thick, Arrow[
        {bounce[[1]], ray[bounce[[1]], norm[bounce[[2]] - bounce[[1]]], .33]}]},
      {Red, Dashed, Line@bounce, PointSize@Medium, Point@bounce[[1]]}}];
   Show[{plotEll[1.5], gr}, PlotRange → All,
    ImageSize → Large, Frame → True, FrameStyle → Medium], {{i, 1}, 1, n, 1}]]
```

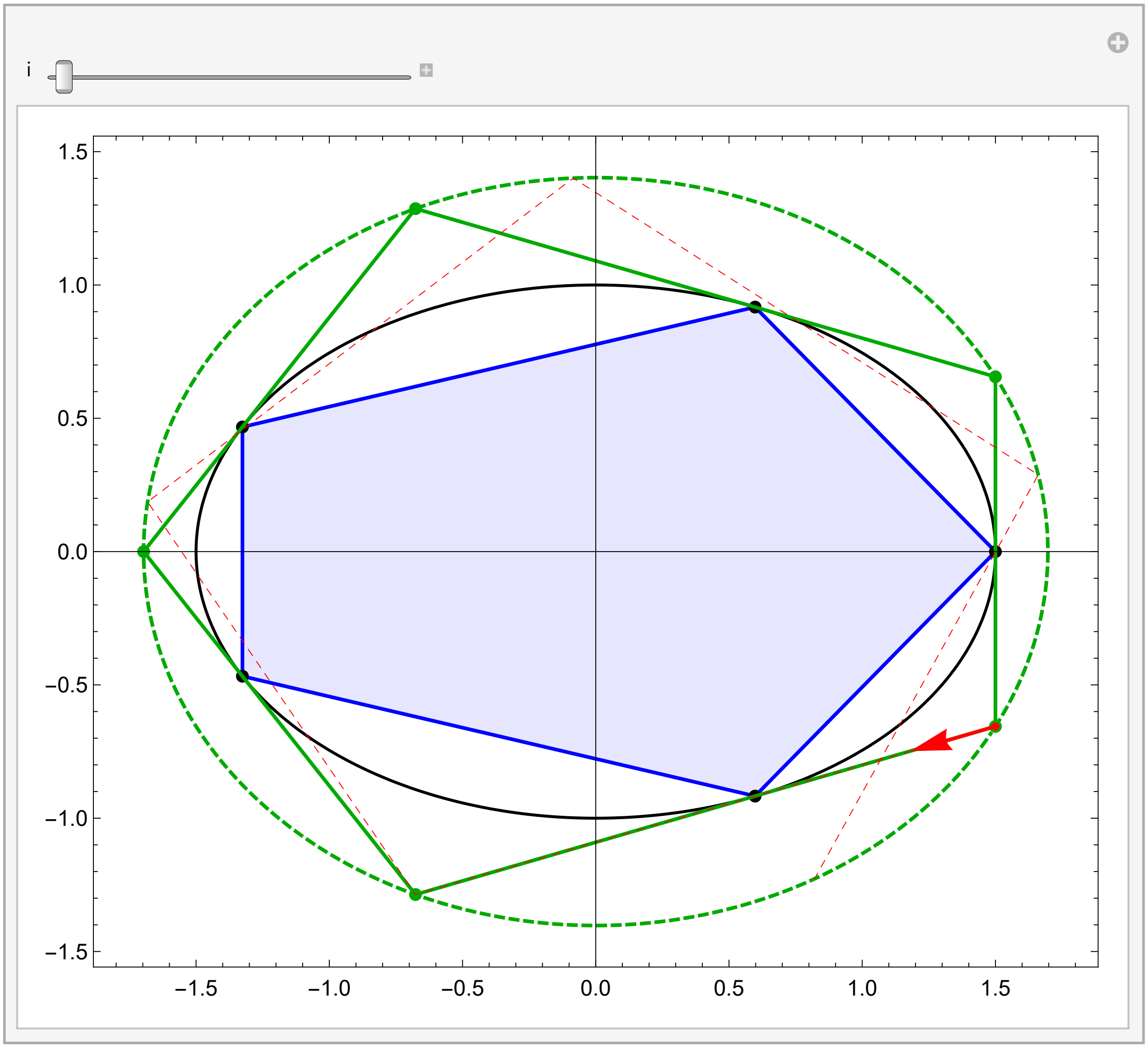
In[2243]:= `showPolyExcentralLocus[quadAlphaT15, getQuadVtx0]`



Out[2243]=

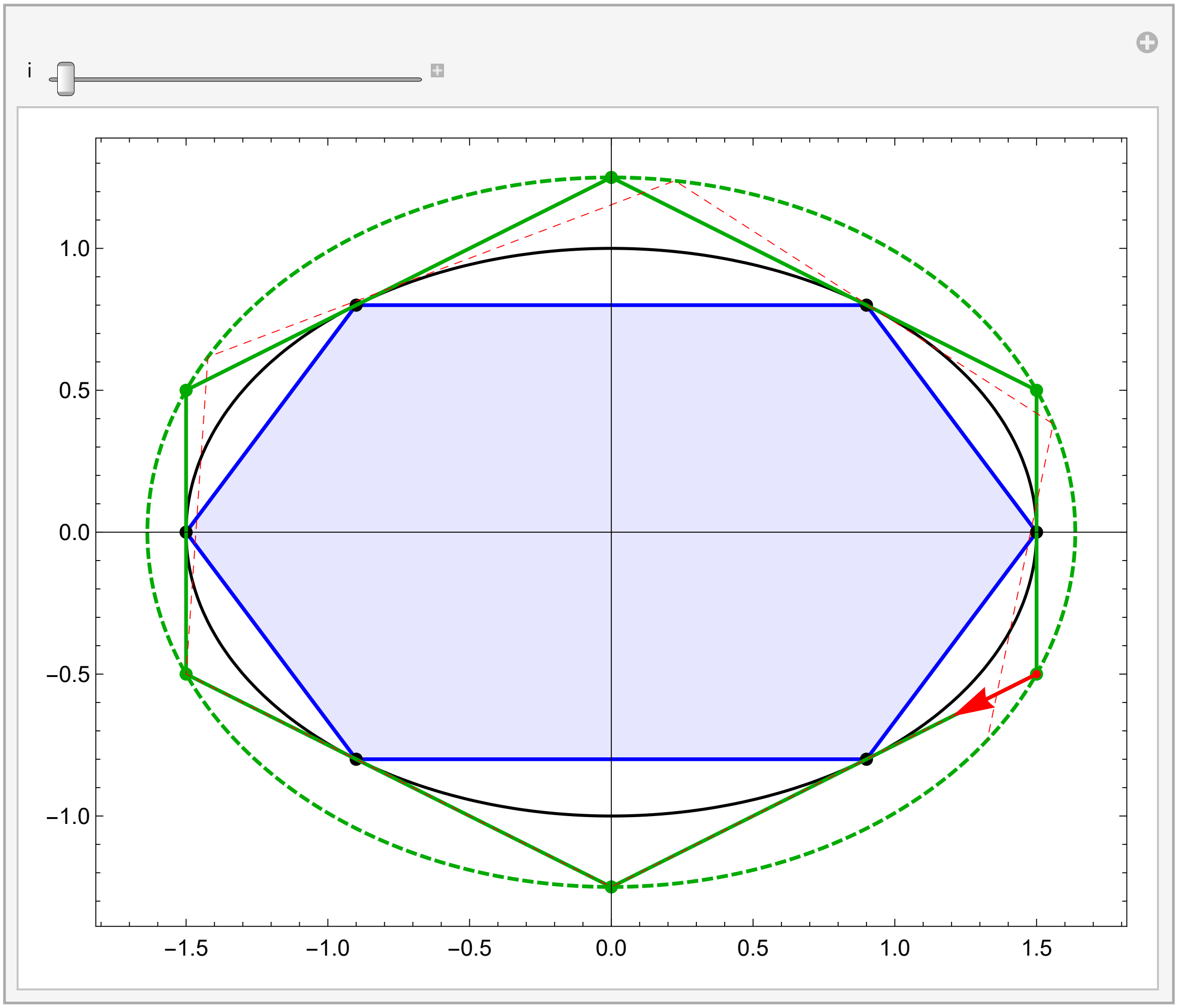
In[2244]:= `showPolyExcentralLocus[pentAlphaT15, getPentVtx0]`

Out[2244]=



In[2245]:= `showPolyExcentralLocus[hexAlphaT15, getHexVtx0]`

Out[2245]=



Export Polys to XLS

```
fnames = FileNames["*15.m", "data"]
```

```
{data\ellAlphaT_a15.m, data\heptAlphaCausticT_a15.m,
 data\heptAlphaT_a15.m, data\hexAlphaT_a15.m, data\nonaAlphaCausticT_a15.m,
 data\octAlphaT_a15.m, data\pentAlphaCausticT_a15.m,
 data\pentAlphaT_a15.m, data\quadAlphaT_a15.m, data\triAlphaT_a15.m}
```

```
{"data\\ellAlphaT_a15.m", "data\\pentAlphaT_a15.m", "data\\hexAlphaT_a15.m",
 "data\\heptAlphaT_a15.m", "data\\quadAlphaT_a15.m", "data\\triAlphaT_a15.m"}
```

```
{data\ellAlphaT_a15.m, data\pentAlphaT_a15.m, data\hexAlphaT_a15.m,
 data\heptAlphaT_a15.m, data\quadAlphaT_a15.m, data\triAlphaT_a15.m}
```

```
chgExt[fname_, newExt_] :=
  StringReplace[fname, RegularExpression["[^\\.]+?$"] → newExt];
Clear@
 remExt;
remExt[fname_] := StringReplace[fname, RegularExpression["\\.[^\\.]+$"] -> ""];

Clear@prepDF;
prepDF[fname_, fnVtx0_] := Module[{a, df, alphaT, polys, n, vnames},
   alphaT = Get[fname];
   a = "a" /. alphaT;
   df = Transpose[{"tsDeg", "tsRad", "alphas"} /. alphaT];
   polys = Table[polyVtx[alphaT, i, fnVtx0], {i, Length@df}];
   n = Length[polys[[1]]];
   vnames = Flatten@Table[{"x" <> ToString@i, "y" <> ToString@i}, {i, n}];
   df = Prepend[Chop[#, 10^-7] & /@ MapThread[
        Join[(*{#3},*)#1, Sequence @@ #2] &, {df, polys(*,Range[Length@df]*)}],
     {(*"row",*)"t_deg", "t_rad", "alpha_rad", Sequence @@ vnames}];
   Print["Converted " <> ToString[Length@df] <> " records from file " <> fname];
   df];
exportDFs;
exportDFs[xlsOut_, fnames_, fnVtx0s_, sheetNames_] := Module[{dfs, rules},
   dfs = MapThread[prepDF[#1, #2] &, {fnames, fnVtx0s}];
   rules = MapThread[(#1 → #2) &, {sheetNames, dfs}];
   Export[xlsOut, "Sheets" → rules, "Rules"]];

exportDFs[
 "data/orbitPolys_a15.xlsx",
 {"data/ellAlphaT_a15.m",
  "data/triAlphaT_a15.m",
  "data/quadAlphaT_a15.m",
  "data/pentAlphaT_a15.m",
  "data/hexAlphaT_a15.m",
  "data/heptAlphaT_a15.m"},
 {getEllVtx0, getTriVtx0, getQuadVtx0, getPentVtx0, getHexVtx0, getHeptVtx0},
 {"ell a=1.5", "N=3", "N=4", "N=5", "N=6", "N=7"}]
```

```
Converted 361 records from file data/ellAlphaT_a15.m
Converted 361 records from file data/triAlphaT_a15.m
Converted 361 records from file data/quadAlphaT_a15.m
Converted 361 records from file data/ellAlphaT_a15.m
Converted 361 records from file data/triAlphaT_a15.m
Converted 361 records from file data/quadAlphaT_a15.m
Converted 361 records from file data/pentAlphaT_a15.m
Converted 361 records from file data/hexAlphaT_a15.m
Converted 361 records from file data/heptAlphaT_a15.m
data/orbitPolys_a15.xlsx
Converted 361 records from file data/heptAlphaT_a15.m
Converted 361 records from file data/hexAlphaT_a15.m
Converted 361 records from file data/heptAlphaT_a15.m
data/orbitPolys_a15.xlsx
```

SLOW: Pentagon: α at rest position?

In[1717]:=
```
Clear@pentEqn0;
pentEqn0 = Quiet@Module[{p3, p2, p1},
   p3 = ellP[a, x3];
   (*n3= ellGrad[a,Sequence@@p3];*)
   p2 = FullSimplify[getInterRefl[a, flipY[p3], p3], a > 1 && -a < x3 < 0];
   p1 = FullSimplify[getInterRefl[a, p3, p2], a > 1 && -a < x3 < 0];
   FullSimplify[{p1[[1]] == a, p1[[2]] == 0}, a > 1 && -a < x3 < 0]]
```

Out[1718]=
$$\{a\,(a^{11}+2\,a^6\,(-4+3\,a^2)\,x3+a^5\,(-4+3\,a^2-3\,a^4)\,x3^2+a\,(-4+7\,a^2-6\,a^4+3\,a^6)\,x3^4+2\,(-1-2\,a^2+3\,a^4)\,x3^5)==12\,a^5\,(-1+a^2)\,x3^3+(-1+a^2)^3\,x3^6,\; a^{24}+a^{12}\,(208-368\,a^2+191\,a^4-30\,a^6+15\,a^8)\,x3^4+a^4\,(-1+a^2)^2\,(16+16\,a^2+191\,a^4-30\,a^6+15\,a^8)\,x3^8+(-1+a^2)^6\,x3^{12}==2\,a^2\,x3^2\,(a^{12}\,(32-48\,a^2+22\,a^4-3\,a^6+3\,a^8)+2\,a^8\,(-1+a^2)^2\,(66-5\,a^2+5\,a^4)\,x3^4+(-1+a^2)^2\,(6+a^2+31\,a^4-9\,a^6+3\,a^8)\,x3^8)\}$$

In[1719]:=
```
pent0sols = x3 /. Normal[Solve[pentEqn0[[2]], x3, Reals]]
```

Out[1719]=
$$\{-\sqrt{\text{Root}[a^{24}+(-64\,a^{14}+96\,a^{16}-44\,a^{18}+6\,a^{20}-6\,a^{22})\,\#1+(208\,a^{12}-368\,a^{14}+191\,a^{16}-30\,a^{18}+15\,a^{20})\,\#1^2+(-264\,a^{10}+548\,a^{12}-324\,a^{14}+60\,a^{16}-20\,a^{18})\,\#1^3+(16\,a^4-16\,a^6+175\,a^8-396\,a^{10}+266\,a^{12}-60\,a^{14}+15\,a^{16})\,\#1^4+(-12\,a^2+22\,a^4-70\,a^6+140\,a^8-104\,a^{10}+30\,a^{12}-6\,a^{14})\,\#1^5+(1-6\,a^2+15\,a^4-20\,a^6+15\,a^8-6\,a^{10}+a^{12})\,\#1^6\,\&,\,1]},$$
$$\sqrt{\text{Root}[a^{24}+(-64\,a^{14}+96\,a^{16}-44\,a^{18}+6\,a^{20}-6\,a^{22})\,\#1+}$$

$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 1\Big],$$

$$-\sqrt{\text{Root}\Big[a^{24} + \left(-64\, a^{14} + 96\, a^{16} - 44\, a^{18} + 6\, a^{20} - 6\, a^{22}\right) \#1 +}$$
$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 2\Big],$$

$$\sqrt{\text{Root}\Big[a^{24} + \left(-64\, a^{14} + 96\, a^{16} - 44\, a^{18} + 6\, a^{20} - 6\, a^{22}\right) \#1 +}$$
$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 2\Big],$$

$$-\sqrt{\text{Root}\Big[a^{24} + \left(-64\, a^{14} + 96\, a^{16} - 44\, a^{18} + 6\, a^{20} - 6\, a^{22}\right) \#1 +}$$
$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 3\Big],$$

$$\sqrt{\text{Root}\Big[a^{24} + \left(-64\, a^{14} + 96\, a^{16} - 44\, a^{18} + 6\, a^{20} - 6\, a^{22}\right) \#1 +}$$
$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 3\Big],$$

$$-\sqrt{\text{Root}\Big[a^{24} + \left(-64\, a^{14} + 96\, a^{16} - 44\, a^{18} + 6\, a^{20} - 6\, a^{22}\right) \#1 +}$$
$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 4\Big],$$

$$\sqrt{\text{Root}\Big[a^{24} + \left(-64\, a^{14} + 96\, a^{16} - 44\, a^{18} + 6\, a^{20} - 6\, a^{22}\right) \#1 +}$$
$$\left(208\, a^{12} - 368\, a^{14} + 191\, a^{16} - 30\, a^{18} + 15\, a^{20}\right) \#1^{2} +$$
$$\left(-264\, a^{10} + 548\, a^{12} - 324\, a^{14} + 60\, a^{16} - 20\, a^{18}\right) \#1^{3} +$$
$$\left(16\, a^{4} - 16\, a^{6} + 175\, a^{8} - 396\, a^{10} + 266\, a^{12} - 60\, a^{14} + 15\, a^{16}\right) \#1^{4} +$$
$$\left(-12\, a^{2} + 22\, a^{4} - 70\, a^{6} + 140\, a^{8} - 104\, a^{10} + 30\, a^{12} - 6\, a^{14}\right) \#1^{5} +$$
$$\left(1 - 6\, a^{2} + 15\, a^{4} - 20\, a^{6} + 15\, a^{8} - 6\, a^{10} + a^{12}\right) \#1^{6}\ \&,\ 4\Big]\Big\}$$

In[1720]:= **pent0sols /. {a → 1.5}**

Out[1720]= {-1.11862, 1.11862, -1.326, 1.326, -1.75242, 1.75242, -15.7638, 15.7638}

This one works!

In[1721]:=
```
Clear@pent0X3;
pent0X3[a_] = pent0sols[[3]]
```

Out[1722]=
$$-\sqrt{\text{Root}\left[a^{24}+\left(-64\,a^{14}+96\,a^{16}-44\,a^{18}+6\,a^{20}-6\,a^{22}\right)\#1+\left(208\,a^{12}-368\,a^{14}+191\,a^{16}-30\,a^{18}+15\,a^{20}\right)\#1^2+\left(-264\,a^{10}+548\,a^{12}-324\,a^{14}+60\,a^{16}-20\,a^{18}\right)\#1^3+\left(16\,a^4-16\,a^6+175\,a^8-396\,a^{10}+266\,a^{12}-60\,a^{14}+15\,a^{16}\right)\#1^4+\left(-12\,a^2+22\,a^4-70\,a^6+140\,a^8-104\,a^{10}+30\,a^{12}-6\,a^{14}\right)\#1^5+\left(1-6\,a^2+15\,a^4-20\,a^6+15\,a^8-6\,a^{10}+a^{12}\right)\#1^6\;\&,\;2\right]}$$

Where is the root?

In[1723]:=
```
Root[
  a^24 + (-64 a^14 + 96 a^16 - 44 a^18 + 6 a^20 - 6 a^22) #1 + (208 a^12 - 368 a^14 + 191 a^16 - 30 a^18 + 15 a^20)
     #1^2 + (-264 a^10 + 548 a^12 - 324 a^14 + 60 a^16 - 20 a^18) #1^3 +
    (16 a^4 - 16 a^6 + 175 a^8 - 396 a^10 + 266 a^12 - 60 a^14 + 15 a^16) #1^4 +
    (-12 a^2 + 22 a^4 - 70 a^6 + 140 a^8 - 104 a^10 + 30 a^12 - 6 a^14) #1^5 +
    (1 - 6 a^2 + 15 a^4 - 20 a^6 + 15 a^8 - 6 a^10 + a^12) #1^6 &, 2] /. {a → 1.5}
```

Out[1723]= 1.75828

Where is x3?

In[1724]:=
```
N[pent0X3[1.5], 10]
```

Out[1724]= -1.326

In[1725]:=
```
First[pent0X3[a][[2, 1]]][x]
```

Out[1725]=
$$a^{24}+\left(-64\,a^{14}+96\,a^{16}-44\,a^{18}+6\,a^{20}-6\,a^{22}\right)x+\left(208\,a^{12}-368\,a^{14}+191\,a^{16}-30\,a^{18}+15\,a^{20}\right)x^2+\left(-264\,a^{10}+548\,a^{12}-324\,a^{14}+60\,a^{16}-20\,a^{18}\right)x^3+\left(16\,a^4-16\,a^6+175\,a^8-396\,a^{10}+266\,a^{12}-60\,a^{14}+15\,a^{16}\right)x^4+\left(-12\,a^2+22\,a^4-70\,a^6+140\,a^8-104\,a^{10}+30\,a^{12}-6\,a^{14}\right)x^5+\left(1-6\,a^2+15\,a^4-20\,a^6+15\,a^8-6\,a^{10}+a^{12}\right)x^6$$

Divide exponents by 2

In[1726]:=
```
pent0X3[a] /. {a^n_ :> (a₂)^(n/2)}
```

Out[1726]=

$$-\sqrt{\text{Root}\Big[a_2^{12} + \#1^6\left(1 - 6\,a_2 + 15\,a_2^2 - 20\,a_2^3 + 15\,a_2^4 - 6\,a_2^5 + a_2^6\right) + \#1^5\left(-12\,a_2 + 22\,a_2^2 - 70\,a_2^3 + 140\,a_2^4 - 104\,a_2^5 + 30\,a_2^6 - 6\,a_2^7\right) + \#1^4\left(16\,a_2^2 - 16\,a_2^3 + 175\,a_2^4 - 396\,a_2^5 + 266\,a_2^6 - 60\,a_2^7 + 15\,a_2^8\right) + \#1^3\left(-264\,a_2^5 + 548\,a_2^6 - 324\,a_2^7 + 60\,a_2^8 - 20\,a_2^9\right) + \#1^2\left(208\,a_2^6 - 368\,a_2^7 + 191\,a_2^8 - 30\,a_2^9 + 15\,a_2^{10}\right) + \#1\left(-64\,a_2^7 + 96\,a_2^8 - 44\,a_2^9 + 6\,a_2^{10} - 6\,a_2^{11}\right)\ \&,\ 2\Big]}$$

In[1727]:=
```
N[pent0X3[1.5], 20]
```

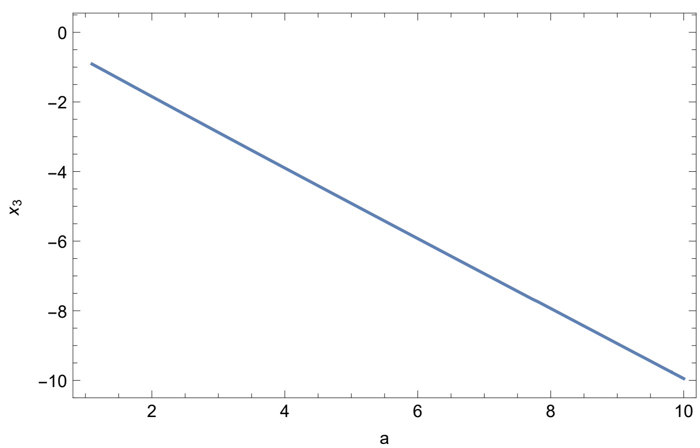
Out[1727]= -1.326

In[1728]:=
```
Since relation is almost linear do a linear model !
```

Out[1728]= a almost do is linear^2 relation Since model !

In[1729]:=
```
Plot[pent0X3[a], {a, 1.1, 10}, Frame → True,
 AxesOrigin → {1, 0}, FrameLabel → {"a", "x₃"}]
```

Out[1729]=



In[1730]:=
```
pent0X3[.5]
```

Out[1730]= -0.318395

In[1731]:=
```
lmPent0X3 = LinearModelFit[Table[{a, pent0X3[a]}, {a, 1, 5, .01}], x, x]
```

Out[1731]= FittedModel[$0.20999 - 1.02709\,x$]

In[1732]:=
```
nlmPent0X3 = NonlinearModelFit[
  Table[{a, pent0X3[a]}, {a, 1, 5, .01}], {c1 + c2 x + c3 x^3}, {c1, c2, c3}, x]
```

Out[1732]= FittedModel[$0.23207 - 1.0403\,x + 0.000448996\,x^3$]

In[1733]:=
```
nlmPent0X3["RSquared"]
```

Out[1733]= 1.

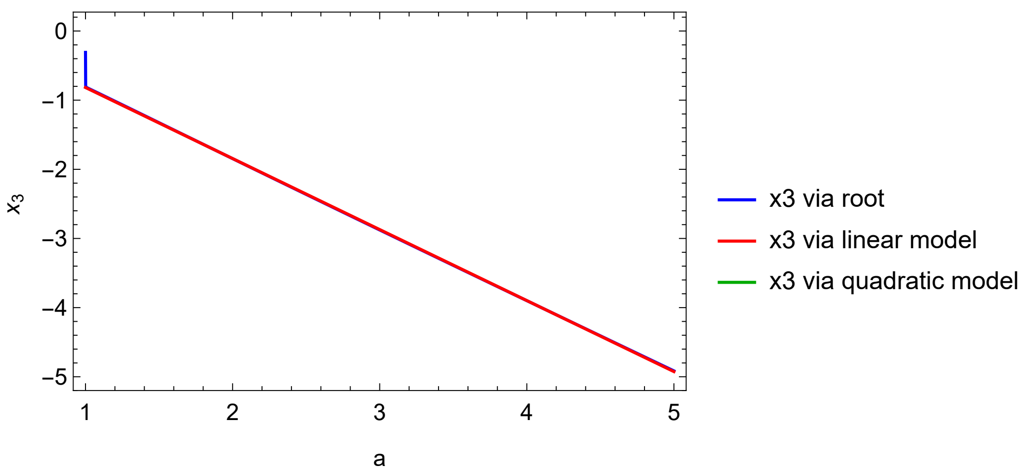
In[1734]:=
```
lmPent0X3["RSquared"]
```
Out[1734]= 0.999983

In[1735]:=
```
pent0X3[1.5]
```
Out[1735]= -1.326

In[1736]:=
```
lmPent0X3[1.5]
```
Out[1736]= -1.33065

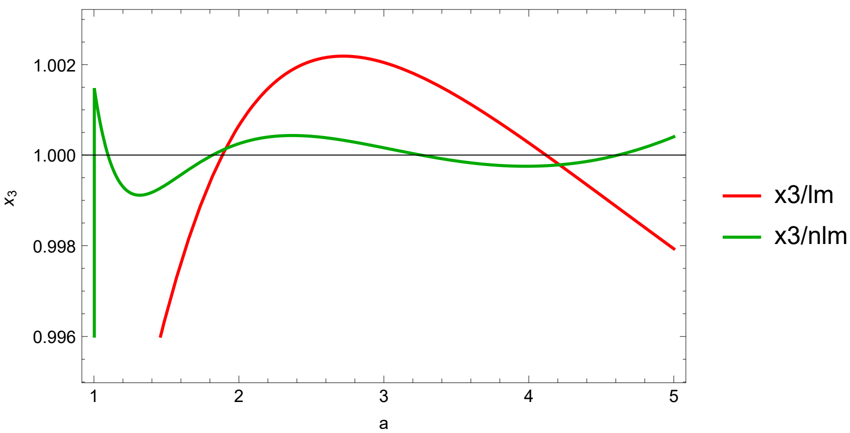
In[1737]:=
```
nlmPent0X3[1.5]
```
Out[1737]= -1.32686

In[1738]:=
```
Plot[{pent0X3[a], lmPent0X3[a], nlmPent0X3},
 {a, 1, 5}, PlotStyle → {Blue, Red, Darker@Green},
 PlotLegends → {"x3 via root", "x3 via linear model", "x3 via quadratic model"},
 Frame → True, AxesOrigin → {1, 0}, FrameStyle → Medium,
 FrameLabel → (Style[#, Medium] & /@ {"a", "x₃"})]
```
Out[1738]=



In[1739]:=
```
LogPlot[{pent0X3[a] / lmPent0X3[a], pent0X3[a] / nlmPent0X3[a]}, {a, 1, 5},
 PlotStyle → {Red, Darker@Green}, PlotLegends → {"x3/lm", "x3/nlm"},
 AxesOrigin → {1, 1}, Frame → True, FrameLabel → {"a", "x₃"}]
```

Out[1739]=



Ronaldo : eqn de x3 e x2 em fn de x3

In[2024]:=
```
Clear@pentX2ronaldo;
pentX2ronaldo[a_, x2_] = a^12 + (2 * a^9 - 4 * a^11) * x2 + (-9 * a^8 + 5 * a^10) * x2^2 +
   (-8 * a^5 + 8 * a^7) * x2^3 + (-4 * a^2 + 3 * a^4 + 6 * a^6 - 5 * a^8) * x2^4 +
   (-2 * a + 8 * a^3 - 10 * a^5 + 4 * a^7) * x2^5 + (1 - 3 * a^2 + 3 * a^4 - a^6) * x2^6;
```

In[2020]:=
```
pentX3ronaldo[a_, x3_] =
  a^12 + (-8 * a^7 + 6 * a^9) * x3 + (-4 * a^6 + 3 * a^8 - 3 * a^10) * x3^2 +
   (12 * a^5 - 12 * a^7) * x3^3 + (-4 * a^2 + 7 * a^4 - 6 * a^6 + 3 * a^8) * x3^4 +
   (-2 * a - 4 * a^3 + 6 * a^5) * x3^5 + (1 - 3 * a^2 + 3 * a^4 - a^6) * x3^6;
```

In[2236]:=
```
Select[x2 /. NSolve[pentX2ronaldo[1.5, x2], x2], # ∈ Reals &]
```

Out[2236]=
```
{-1.49112, 0.597948, 2.25176, 3.88258}
```

```
(* x2: a>=1: [[2]], else [[3]] *)
```

In[2235]:=
```
Table[{a, Select[x2 /. NSolve[pentX2ronaldo[a, x2], x2], # ∈ Reals &]},
  {a, .98, 1.02, .01}] // ColumnForm
```

Out[2235]=
```
{0.98, {-48.515, -0.762715, 0.298619, 14.0778}}
{0.99, {-98.5075, -0.786066, 0.303805, 21.062}}
{1., {-0.809017, 0.309017}}
{1.01, {-0.831495, 0.314256, 15.416, 101.508}}
{1.02, {-0.853442, 0.31952, 9.71298, 51.515}}
```

```
(* x3: a>=1: [[1]], else[[2]] *)
```

In[2217]:=
```
Table[{a, Select[x3 /. NSolve[pentX2ronaldo[a, x3], x3], # ∈ Reals &]},
   {a, .98, 1.02, .01}] // ColumnForm
```

Out[2217]=
```
{0.98, {-48.515, -0.762715, 0.298619, 14.0778}}
{0.99, {-98.5075, -0.786066, 0.303805, 21.062}}
{1., {-0.809017, 0.309017}}
{1.01, {-0.831495, 0.314256, 15.416, 101.508}}
{1.02, {-0.853442, 0.31952, 9.71298, 51.515}}
```

In[2189]:=
```
Select[x2 /. NSolve[pentX2ronaldo[.99, x2], x2], # ∈ Reals &]
```

Out[2189]=
```
{-98.5075, -0.786066, 0.303805, 21.062}
```

In[2040]:=
```
Select[x3 /. NSolve[pentX3ronaldo[1.5, x3], x3], # ∈ Reals &]
```

Out[2040]=
```
{-1.326, 1.11862, 1.75242, 15.7638}
```

In[2175]:=
```
Select[x3 /. NSolve[pentX3ronaldo[.99, x3], x3], # ∈ Reals &]
```

Out[2175]=
```
{-3.04103, -0.798796, 0.291494, 19701.5}
```

In[2176]:=
```
Select[x3 /. NSolve[pentX3ronaldo[1.01, x3], x3], # ∈ Reals &]
```

Out[2176]=
```
{-0.819246, 0.326923, 3.08423, 20301.5}
```

In[2010]:=
```
pentX2frX3[a_, x3_] := Module[{f1, f2, denom},
    f1 = (2 * a^3 + a^4 - x3^2 - 2 * a * x3^2 - a^2 * x3^2);
    f2 = (-2 * a^3 + a^4 - x3^2 + 2 * a * x3^2 - a^2 * x3^2);
    denom = a^8 + 4 * a^2 * x3^2 - 2 * a^4 * x3^2 -
      2 * a^6 * x3^2 - 3 * x3^4 + 2 * a^2 * x3^4 + a^4 * x3^4;
    (x3 * f1 * f2) / denom];
```

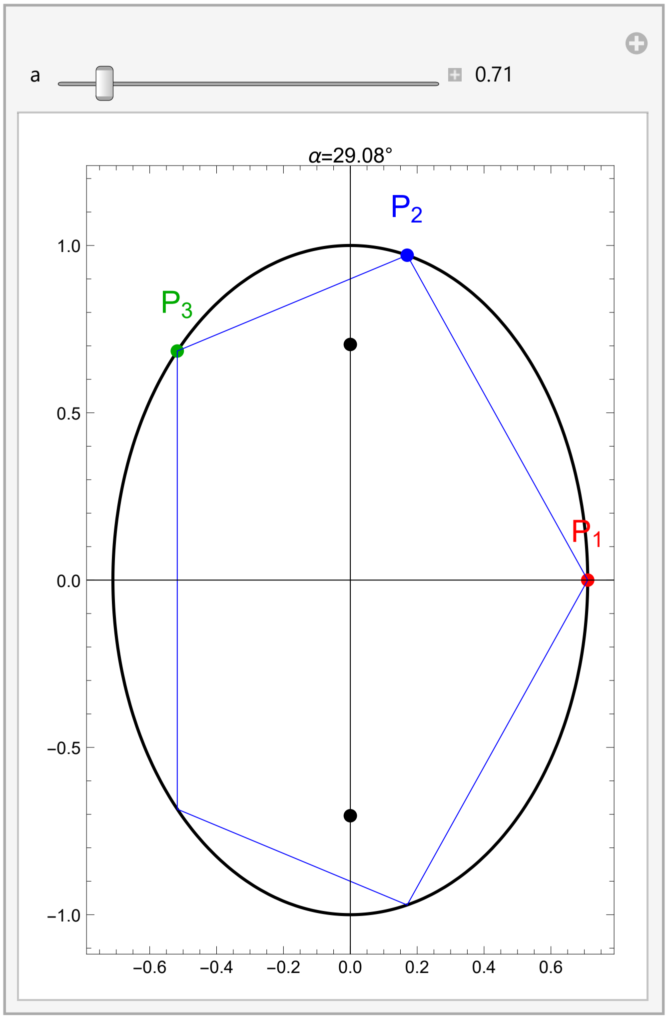
In[2043]:=
```
pentX2frX3[1.5, -1.326]
```

Out[2043]=
```
0.597953
```

In[2237]:=
```
Clear@showPentEqnX2ronaldo;
Options@showPentEqnX2ronaldo = {drRoot → True, drFoci → True};
showPentEqnX2ronaldo[a_, OptionsPattern[]] :=
  Module[{fs, x2, x3sols, x3, p1, p2, x2sols, p3, p21, ell,
    gr, ps, bounce, alpha, clrs = {Red, Blue, Darker@Green},
    lgt = .33},
   ell = plotEll[a];
   fs = getFoci[a];
   p1 = {a, 0};
   (* x2: a>=1: [[2]], else [[3]] *)
   x2sols = Select[x2 /. NSolve[pentX2ronaldo[a, x2], x2], # ∈ Reals &];
   x2 = If[a >= 1, x2sols[[2]], x2sols[[3]]];
   p2 = ellP[a, x2];
   (* x3: a>=1: [[1]], else[[2]] *)
   x3sols = Select[x3 /. NSolve[pentX3ronaldo[a, x3], x3], # ∈ Reals &];
   x3 = If[a >= 1, x3sols[[1]], x3sols[[2]]];
   p3 = ellP[a, x3];
   bounce = bounceRay[a, p1, p2, 4];
   p21 = p2 - p1;
   ps = {p1, p2, p3};
   alpha = toDeg[ArcTan[p21[[2]], -p21[[1]]]];
   gr = Graphics[{PointSize@Large,
      MapThread[{#1, Point@#2, txtSubscript["P", ToString@#3, 16, #2 + {0, .15}]} &,
       {clrs, ps, Range[3]}],
      If[OptionValue@drFoci, {Black, Point@fs}, {}],
      If[OptionValue@drRoot, {Blue, Line@bounce}, {}]}];
   Show[{ell, gr}, Frame → True,
    PlotLabel → "α=" <> nfn[alpha, 2] <> "°", PlotRange → All]];
```

In[2240]:=
```
Manipulate[showPentEqnX2ronaldo[a], {{a, 1.5}, .5, 3, .01, Appearance → "Labeled"}]
```

Out[2240]=

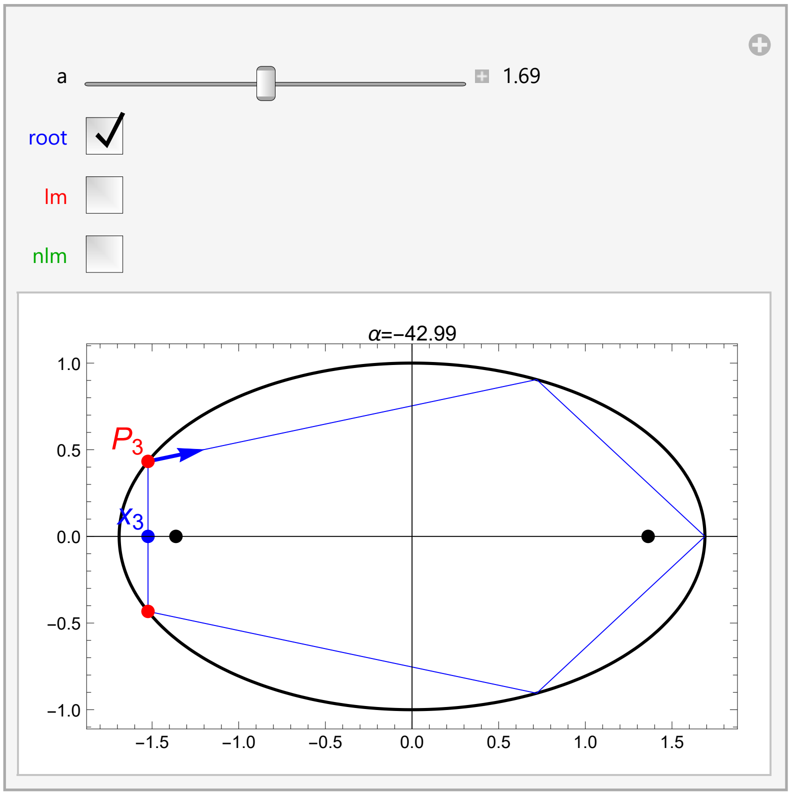


Show pentX3 solutions, root and quad model

In[1740]:=
```
Clear@showPentEqnX3;
Options@showPentEqnX3 =
  {drRoot → True, drLm → False, drNlm → False, scalex → 1, drFoci → True};
showPentEqnX3[a_, OptionsPattern[]] :=
 Module[{fs, x3, p3, p4, ell, gr, bounce, bounce34, alpha,
   x3lm, p3lm, bouncelm, x3nlm, p3nlm, bouncenlm,
   lgt = .33},
  ell = plotEll[a];
  fs = getFoci[a];
  x3 = pent0X3[a];
  p3 = ellP[a, x3];
  p4 = flipY[p3];
  bounce = bounceRay[a, p4, p3, 4];
  bounce34 = bounce[[3]] - bounce[[4]];
  alpha = toDeg[ArcTan[bounce34[[2]] / bounce34[[1]]]];
  x3lm = lmPent0X3[a];
  p3lm = ellP[a, x3lm];
  bouncelm = bounceRay[a, flipY@p3lm, p3lm, 4];
  x3nlm = nlmPent0X3[a];
  p3nlm = ellP[a, x3nlm];
  bouncenlm = bounceRay[a, flipY@p3nlm, p3nlm, 4];
  gr = Graphics[{PointSize@Large,
     If[OptionValue@drFoci, {Black, Point@fs}, {}],
     If[OptionValue@drRoot, {Blue, Line@bounce}, {}],
     If[OptionValue@drLm, {Red, Line@bouncelm}, {}],
     If[OptionValue@drNlm, {Darker@Green, Line@bouncenlm}, {}],
     {Blue, Thick, Arrow[{p3, p3 + lgt * norm[bounce[[3]] - p3]}]},
     {Red, Point@p4, Point@p3, Text[Style["P₃", 16], p3, {1.2, -1.2}]},
     {Blue, Point@{x3, 0}, Text[Style["x₃", 16], {x3, 0}, {1.2, -1.2}]}}];
  Show[{ell, gr}, Frame → True, PlotLabel → "α=" <> nfn[alpha, 2]]]
```

In[1743]:=
```
Manipulate[showPentEqnX3[a, drRoot → root, drLm → lm, drNlm → nlm],
 {{a, 1.5}, .5, 3, .001, Appearance → "Labeled"},
 {{root, True, Style["root", Blue]}, {True, False}},
 {{lm, False, Style["lm", Red]}, {True, False}},
 {{nlm, False, Style["nlm", Darker@Green]}, {True, False}}
]
```

Out[1743]=

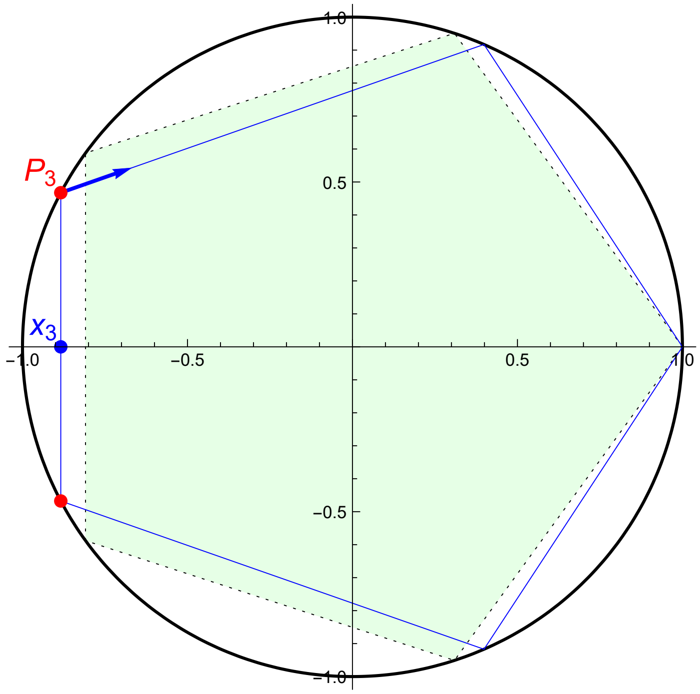


In[1744]:=
```
Range[0, 2 π, 2 π / 5]
```

Out[1744]= $\left\{0, \frac{2\pi}{5}, \frac{4\pi}{5}, \frac{6\pi}{5}, \frac{8\pi}{5}, 2\pi\right\}$

In[1745]:=
```
Show[{Graphics@{Thick, Dashed, Green, Opacity@.1,
    EdgeForm[{Black, Dotted}],
    Polygon[rot[{1, 0}, Sin@#, Cos@#] & /@ Range[0, 2 π - 2 π / 5, 2 π / 5]]},
  Graphics@Map[Scale[#, {1 / 1.5, 1}, {0, 0}] &,
    List @@ showPentEqnX3[1.5, drFoci → False]]}, Axes → True]
```

Out[1745]=



In[1746]:=
```
showPentEqnX3[1.5], showPren
```

Obtains interior of Root[], in terms of “x” (a sextic), whose coefficients are polynomials in $a_2 = a^2$

In[1746]:=
```
pent0poly = First[pent0X3[a][[2, 1]]][x] /. {a^n_ :> (a₂)^(n/2)}
```

Out[1746]=

$$a_2^{12} + x^6 \left(1 - 6 a_2 + 15 a_2^2 - 20 a_2^3 + 15 a_2^4 - 6 a_2^5 + a_2^6\right) + x^5 \left(-12 a_2 + 22 a_2^2 - 70 a_2^3 + 140 a_2^4 - 104 a_2^5 + 30 a_2^6 - 6 a_2^7\right) + x^4 \left(16 a_2^2 - 16 a_2^3 + 175 a_2^4 - 396 a_2^5 + 266 a_2^6 - 60 a_2^7 + 15 a_2^8\right) + x^3 \left(-264 a_2^5 + 548 a_2^6 - 324 a_2^7 + 60 a_2^8 - 20 a_2^9\right) + x^2 \left(208 a_2^6 - 368 a_2^7 + 191 a_2^8 - 30 a_2^9 + 15 a_2^{10}\right) + x \left(-64 a_2^7 + 96 a_2^8 - 44 a_2^9 + 6 a_2^{10} - 6 a_2^{11}\right)$$

In[1747]:=
```
-Sqrt[pent0poly /. {a₂ → 1.5^2, x → 1.35}]
```

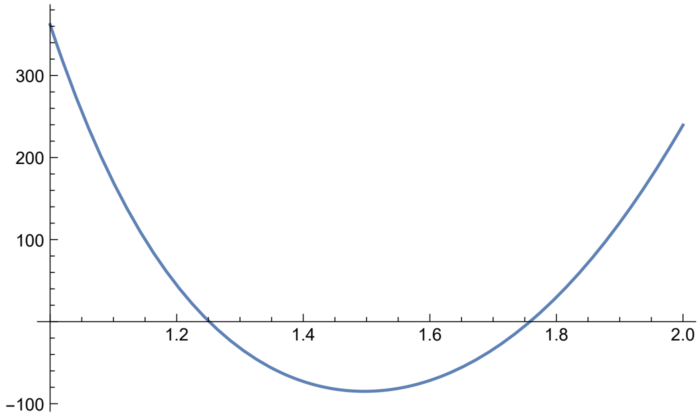
Out[1747]= 0. − 7.51587 i

In[1748]:=
```
pent0poly /. {a₂ → (1.5^2), x → -1.3260005019839656}
```

Out[1748]= 263 463.

In[1749]:=
```
Plot[pent0poly /. {a₂ → (1.5^2), x → theX}, {theX, 1, 2}]
```

Out[1749]=



To get X3 I need the negative of the square root of the second root of the following sextic :

In[1750]:=
```
Grid[Prepend[
  Transpose@{{1, x, x^2, x^3, x^4, x^5, x^6}, CoefficientList[pent0poly, x]},
  {"term", "coeff"}], Alignment → Right,
 ItemStyle → Directive[FontSize -> 20, FontFamily -> "Courier"], Frame → All]
```

Out[1750]=

term	coeff
1	a_2^{12}
x	$-64\, a_2^7 + 96\, a_2^8 - 44\, a_2^9 + 6\, a_2^{10} - 6\, a_2^{11}$
x^2	$208\, a_2^6 - 368\, a_2^7 + 191\, a_2^8 - 30\, a_2^9 + 15\, a_2^{10}$
x^3	$-264\, a_2^5 + 548\, a_2^6 - 324\, a_2^7 + 60\, a_2^8 - 20\, a_2^9$
x^4	$16\, a_2^2 - 16\, a_2^3 + 175\, a_2^4 - 396\, a_2^5 + 266\, a_2^6 - 60\, a_2^7 + 15\, a_2^8$
x^5	$-12\, a_2 + 22\, a_2^2 - 70\, a_2^3 + 140\, a_2^4 - 104\, a_2^5 + 30\, a_2^6 - 6\, a_2^7$
x^6	$1 - 6\, a_2 + 15\, a_2^2 - 20\, a_2^3 + 15\, a_2^4 - 6\, a_2^5 + a_2^6$

In[1751]:=
```
Table[a^n, {n, 0, 12}]
```

Out[1751]= $\{1, a, a^2, a^3, a^4, a^5, a^6, a^7, a^8, a^9, a^{10}, a^{11}, a^{12}\}$

In[1752]:=
```
pentAcoeffs =
  Table[Coefficient[#, a₂, a2pow] & /@ CoefficientList[pent0poly, x], {a2pow, 0, 12}]
```

Out[1752]=
```
{{0, 0, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 0, -12, -6}, {0, 0, 0, 0, 16, 22, 15},
 {0, 0, 0, 0, -16, -70, -20}, {0, 0, 0, 0, 175, 140, 15}, {0, 0, 0, -264, -396, -104, -6},
 {0, 0, 208, 548, 266, 30, 1}, {0, -64, -368, -324, -60, -6, 0},
 {0, 96, 191, 60, 15, 0, 0}, {0, -44, -30, -20, 0, 0, 0},
 {0, 6, 15, 0, 0, 0, 0}, {0, -6, 0, 0, 0, 0, 0}, {1, 0, 0, 0, 0, 0, 0}}
```

In[1753]:=
```
Transpose[pentAcoeffs].pentAcoeffs // MatrixForm
```

Out[1753]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 15320 & 43298 & 27376 & 5280 & 384 & 0 \\ 0 & 43298 & 216294 & 245276 & 80273 & 8448 & 208 \\ 0 & 27376 & 245276 & 478976 & 270652 & 45840 & 2132 \\ 0 & 5280 & 80273 & 270652 & 262534 & 75496 & 5827 \\ 0 & 384 & 8448 & 45840 & 75496 & 36880 & 4556 \\ 0 & 0 & 208 & 2132 & 5827 & 4556 & 924 \end{pmatrix}$$

In[1754]:=
```
Sqrt@Det[Transpose[pentAcoeffs].pentAcoeffs]
```

Out[1754]= $1088\sqrt{9642191725036764242}$

In[1755]:=
```
Grid[Prepend[Transpose@Prepend[pentAcoeffs, Table[x^(ToString@n), {n, 0, 6}]],
   Prepend[Table[(a_2)^ToString@n, {n, 0, 12}], ""]],
  Frame → All, Alignment → {"Bottom", "Center"},
  ItemStyle → Directive[FontSize -> 16, FontFamily -> "Courier"]]
```

Out[1755]=

	a_2^0	a_2^1	a_2^2	a_2^3	a_2^4	a_2^5	a_2^6	a_2^7	a_2^8	a_2^9	a_2^{10}	a_2^{11}	a_2^{12}
x^0	0	0	0	0	0	0	0	0	0	0	0	0	1
x^1	0	0	0	0	0	0	0	-64	96	-44	6	-6	0
x^2	0	0	0	0	0	0	208	-368	191	-30	15	0	0
x^3	0	0	0	0	0	-264	548	-324	60	-20	0	0	0
x^4	0	0	16	-16	175	-396	266	-60	15	0	0	0	0
x^5	0	-12	22	-70	140	-104	30	-6	0	0	0	0	0
x^6	1	-6	15	-20	15	-6	1	0	0	0	0	0	0

In[1756]:=
```
Transpose[pentAcoeffs] // MatrixForm
```

Out[1756]//MatrixForm=

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -64 & 96 & -44 & 6 & -6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 208 & -368 & 191 & -30 & 15 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -264 & 548 & -324 & 60 & -20 & 0 & 0 & 0 \\ 0 & 0 & 16 & -16 & 175 & -396 & 266 & -60 & 15 & 0 & 0 & 0 & 0 \\ 0 & -12 & 22 & -70 & 140 & -104 & 30 & -6 & 0 & 0 & 0 & 0 & 0 \\ 1 & -6 & 15 & -20 & 15 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

How many non zero?

In[1757]:=
```
pentX3counts = Sort[Tally[Flatten@pentAcoeffs], #1[[2]] > #2[[2]] &]
```

Out[1757]=
```
{{0, 54}, {15, 4}, {-6, 4}, {1, 3}, {-20, 2}, {6, 1}, {-30, 1}, {-44, 1},
 {60, 1}, {191, 1}, {96, 1}, {-60, 1}, {-324, 1}, {-368, 1}, {-64, 1},
 {30, 1}, {266, 1}, {548, 1}, {208, 1}, {-104, 1}, {-396, 1}, {-264, 1},
 {140, 1}, {175, 1}, {-70, 1}, {-16, 1}, {22, 1}, {16, 1}, {-12, 1}}
```

In[1758]:=
```
(* sum of counts *) Total[#[[1]] * #[[2]] & /@ pentX3counts]
```

Out[1758]= 5

In[1759]:=
```
Grid[Prepend[MapThread[Prepend[#2, #1] &,
    {Range[1, Length@pentX3counts - 1], Select[pentX3counts, #[[1]] ≠ 0 &]}],
   {"row", "number", "count"}], Frame → All]
```

Out[1759]=

row	number	count
1	15	4
2	-6	4
3	1	3
4	-20	2
5	6	1
6	-30	1
7	-44	1
8	60	1
9	191	1
10	96	1
11	-60	1
12	-324	1
13	-368	1
14	-64	1
15	30	1
16	266	1
17	548	1
18	208	1
19	-104	1
20	-396	1
21	-264	1
22	140	1
23	175	1
24	-70	1
25	-16	1
26	22	1
27	16	1
28	-12	1

In[1760]:=
```
Total[If[# ≠ 0, 1, 0] & /@ (Flatten@pentAcoeffs)]
```

Out[1760]= 37

Sum of Lines/Cols

In[1761]:=
```
(* cols *) Total /@ pentAcoeffs
```

Out[1761]= {1, -18, 53, -106, 330, -770, 1053, -822, 362, -94, 21, -6, 1}

In[1762]:=
```
(* lines *) Total /@ Transpose@pentAcoeffs
```

Out[1762]= {1, -12, 16, 0, 0, 0, 0}

In[1763]:=
```
(* all *) Total[Total /@ pentAcoeffs]
```

Out[1763]= 5

In[1764]:=
```
RealDigits[126, 2, 7]
```

Out[1764]= {{1, 1, 1, 1, 1, 1, 0}, 7}

In[1765]:=
```
CoefficientList[pent0poly, x] /. {a2 → 1}
```

Out[1765]=
```
{1, -12, 16, 0, 0, 0, 0}
```

In[1766]:=
```
N[Root[pent0poly /. a2 → (3/2)^2, 2], 50]
```

Out[1766]=
```
1.7582773312617253313511911949828310018815690822691
```

In[1767]:=
```
Clear@encryptX3;
encryptX3[theA_, msg_] := Module[{sexticCoeffs, cc, ccNorm, x3s, digs, dotProd},
  sexticCoeffs = CoefficientList[pent0poly, x] /. {a2 → theA^2};
  cc = ToCharacterCode[msg];
  ccNorm = 1 + cc/256;
  x3s = N /@ (pent0X3 /@ ccNorm);
  digs = First[RealDigits[#, 2, 7]] & /@ cc;
  dotProd = (pentAcoeffs.#) & /@ digs;
  {"sexticCoeffs" → sexticCoeffs,
   "cc" → cc,
   "digs" → digs,
   "dotProd" → dotProd,
   "dotTotals" → Total /@ dotProd,
   "minDot" → Min@dotProd,
   "maxDot" → Max@dotProd,
   "ccNorm" → ccNorm,
   "x3s" → x3s}];
```

In[1768]:=
```
"sexticCoeffs" /. encryptX3[3/2, "hello"]
```

Out[1768]=
$$\left\{\frac{282\,429\,536\,481}{16\,777\,216}, -\frac{95\,616\,333\,279}{2\,097\,152}, \frac{53\,011\,771\,191}{1\,048\,576}, -\frac{1\,891\,044\,225}{65\,536}, \frac{550\,360\,575}{65\,536}, -\frac{8\,038\,575}{8192}, \frac{15\,625}{4096}\right\}$$

Solving a sextic w/ two cubics (Raghavendra G. Kulkarni):

http://euclid.trentu.ca/aejm/V3N1/Kulkarni.V3N1.pdf

$$a_6 = \sqrt{(5a_5^4/64) - (3a_4a_5^2/8) + (a_4^2/4) + (a_3a_5/2) - a_2} \qquad (18)$$

$$a_7 = (a_3a_4/4) + (a_4a_5^3/16) - (a_4^2a_5/8) - (a_3a_5^2/16) - (a_5^5/128) - (a_1/2)$$

$$a_0 = [(a_3/2) + (a_5^3/16) - (a_4a_5/4)]^2 - (a_7/a_6)^2 \qquad (24)$$

```
In[1769]:= Clear@kulkarniCond;
kulkarniCond[a5_, a4_, a3_, a2_, a1_, a0_] := Module[{a6, a7, a0cond},
  a6 = Sqrt[5 a5^4 / 64 - 3 a4 a5^2 / 8 + a4^2 / 4 + a3 a5 / 2 - a2];
  a7 = a3 a4 / 4 + a4 a5^3 / 16 - a4^2 a5 / 8 - a3 a5^2 / 16 - a5^5 / 128 - a1 / 2;
  a0cond = (a3 / 2 + a5^3 / 16 - a4 a5 / 4)^2 - (a7 / a6)^2;
  (* Must be equal *)
  {a0, a0cond}];
```

His example, note that if I flip signs it doesn' t work

test : $x^6 - 8x^5 + 32x^4 - 78x^3 + 121x^2 - 110x + 50 = 0$

```
In[1770]:= N[kulkarniCond @@ ((# * 1) & /@ {-8, 32, -78, 121, -110, 50})]
Out[1770]= {50., 50.}

In[1771]:= N[kulkarniCond @@ ((# * -1) & /@ {-8, 32, -78, 121, -110, 50})]
Out[1771]= {-50., 12 115.3}

In[1772]:= CoefficientList[pent0poly, x]
```

Out[1772]= $\{a_2^{12}, -64 a_2^7 + 96 a_2^8 - 44 a_2^9 + 6 a_2^{10} - 6 a_2^{11}, 208 a_2^6 - 368 a_2^7 + 191 a_2^8 - 30 a_2^9 + 15 a_2^{10},$
$-264 a_2^5 + 548 a_2^6 - 324 a_2^7 + 60 a_2^8 - 20 a_2^9, 16 a_2^2 - 16 a_2^3 + 175 a_2^4 - 396 a_2^5 + 266 a_2^6 - 60 a_2^7 + 15 a_2^8,$
$-12 a_2 + 22 a_2^2 - 70 a_2^3 + 140 a_2^4 - 104 a_2^5 + 30 a_2^6 - 6 a_2^7, 1 - 6 a_2 + 15 a_2^2 - 20 a_2^3 + 15 a_2^4 - 6 a_2^5 + a_2^6\}$

```
In[2027]:= testKulkarni[theA_] := Module[{a6toa0, a5toa0(*a6=1*), a6},
    a6toa0 = Reverse[CoefficientList[pent0poly, x] /. {a2 → theA^2}];
    a6 = First@a6toa0;
    (* to force a6 = 1 *)
    a5toa0 = Drop[a6toa0, 1] / a6;
    kulkarniCond @@ a5toa0];
```

In[2030]:=
```
testKulkarniRonaldo[a_, fn_] := Module[{a6toa0, a5toa0(*a6=1*), a6},
    a6toa0 = Reverse[CoefficientList[fn[a, x], x]];
    a6 = First@a6toa0;
    (* to force a6 = 1 *)
    a5toa0 = Drop[a6toa0, 1] / a6;
    kulkarniCond @@ a5toa0];
```

In[2038]:=
```
testKulkarniRonaldo[1.5, pentX2ronaldo]
```

Out[2038]=
```
{-66.4301, -89.7679}
```

In[2039]:=
```
testKulkarniRonaldo[1.5, pentX3ronaldo]
```

Out[2039]=
```
{-66.4301, 42894.4}
```

Doesn't work!

In[1774]:=
```
N[testKulkarni[1.5]]
```

Out[1774]= $\{4412.96, 7.05286\times10^{11}\}$

In[1775]:=
```
"dotTotals" /. encryptX3["Jair and Ronaldo are Great Mathematicians"]
```

ReplaceAll::reps : {encryptX3[Jair and Ronaldo are Great Mathematicians]} is
neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing. >>

Out[1775]=
```
dotTotals /. encryptX3[Jair and Ronaldo are Great Mathematicians]
```

In[1776]:=
```
ListLinePlot[
  ("dotTotals" /. encryptX3["Jair and Ronaldo are Great Mathematicians"])]
```

ReplaceAll::reps : {encryptX3[Jair and Ronaldo are Great Mathematicians]} is
neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing. >>

ListLinePlot::lpn : dotTotals /. encryptX3[Jair and Ronaldo are Great Mathematicians] is not a list of numbers or pairs of numbers. >>

Out[1776]=
```
ListLinePlot[dotTotals /. encryptX3[Jair and Ronaldo are Great Mathematicians]]
```

Grab Diagonals

In[1777]:=
```
pentAcoeffsDiags = Module[{mtx = Transpose[pentAcoeffs], diag, i},
   Table[mtx[[Length[mtx] - i, i + diag]],
    {diag, 1, Length[mtx]}, {i, 0, Length[mtx] - 1}]]; % // MatrixForm
```

Out[1777]//MatrixForm=
```
ListLinePlot[dotTotals /. encryptX3[Jair and Ronaldo are Great Mathematicians]]
```

In[1778]:=
```
Total /@ pentAcoeffsDiags
```

Out[1778]=
```
{5, 0, 0, 0, 0, 0, 0}
```

The ones below don' t work

In[1779]:=
```
Clear@pentEqn;
pentEqn = Quiet@Module[{p1, r1, p3, n3, r3, p2fr1, p2fr3},
    p1 = {a, 0};
    r1 = {-ca, Sqrt[1 - ca^2]};
    p2fr1 = FullSimplify[ellInterRayUnprot[a, p1, r1][[2]], a > 1 && 0 <= ca <= 1];
    p3 = ellP[a, x3];
    n3 = ellGrad[a, Sequence @@ p3];
    r3 = FullSimplify[refl[{0, -1}, n3], a > 1 && -a < x3 < 0];
    p2fr3 = FullSimplify[ellInterRayUnprot[a, p3, r3][[2]], a > 1 && -a ≤ x3 ≤ 0];
    (*pnext=ellInterRayUnprot[a,pto,theRefl][[2]];*)
    {"p1" -> p1,
     "p2fr1" -> p2fr1,
     "p3" -> p3,
     "n3" -> n3,
     "r3" -> r3,
     "p2fr3" → p2fr3}];
pentEqn // ColumnForm
```

Out[1781]=

$$\text{p1} \to \{a, 0\}$$

$$\text{p2fr1} \to \left\{\frac{a\,(ca^2+a^2\,(-1+ca^2))}{-ca^2+a^2\,(-1+ca^2)}, \frac{2\,a\,ca\,\sqrt{1-ca^2}}{ca^2-a^2\,(-1+ca^2)}\right\}$$

$$\text{p3} \to \left\{x3, \sqrt{1-\frac{x3^2}{a^2}}\right\}$$

$$\text{n3} \to \left\{-x3, -a^2\sqrt{1-\frac{x3^2}{a^2}}\right\}$$

$$\text{r3} \to \left\{-\frac{2\,a\,x3\,\sqrt{(a-x3)\,(a+x3)}}{a^4+x3^2-a^2\,x3^2}, \frac{-a^4+(1+a^2)\,x3^2}{a^4+x3^2-a^2\,x3^2}\right\}$$

$$\text{p2fr3} \to \left\{\frac{a^6\,(-4+a^2)\,x3-2\,a^4\,(-3+a^2)\,x3^3+(-1+a^2)^2\,x3^5}{a^8-2\,a^2\,(-2+a^2+a^4)\,x3^2+(-3+2\,a^2+a^4)\,x3^4}, -\frac{\sqrt{(a-x3)\,(a+x3)}\,\left(a^8-2\,a^2\,(2-a^2+a^4)\,x3^2+(-1+a^2)^2\,x3^4\right)}{a^9-2\,a^3\,(-2+a^2+a^4)\,x3^2+a\,(-3+2\,a^2+a^4)\,x3^4}\right\}$$

In[1782]:=
```
pentEqn /. {a -> 1.5, ca -> Cos[toRad[50]], x3 -> -1}
```

Out[1782]= $\{\text{p1} \to \{1.5, 0\}, \text{p2fr1} \to \{0.784969, 0.852141\}, \text{p3} \to \{-1, 0.745356\}, \text{n3} \to \{1, -1.67705\}, \text{r3} \to \{0.879764, -0.47541\}, \text{p2fr3} \to \{1.3008, -0.497958\}\}$

In[1783]:=
```
Quiet@FindMinimum[
  Module[{p2fr3, p2fr1},
   {p2fr3, p2fr1} =
    ({"p2fr3", "p2fr1"} /. pentEqn /. {a → 1.5, ca → theCa, x3 → theX3});
   magn2[p2fr3 - p2fr1]],
  {{theCa, Cos@toRad[30.]}, {theX3, -1.3}}]
```

Out[1783]= $\{1.55461\times10^{-29}, \{\text{theCa} \to 0.778137, \text{theX3} \to -1.35267\}\}$

In[1784]:= `pentSys = Thread[("p2fr1" /. pentEqn) == ("p2fr3" /. pentEqn)]`

Out[1784]=
$$\left\{\frac{a\left(ca^2+a^2\left(-1+ca^2\right)\right)}{-ca^2+a^2\left(-1+ca^2\right)}==\frac{a^6\left(-4+a^2\right)x3-2a^4\left(-3+a^2\right)x3^3+\left(-1+a^2\right)^2x3^5}{a^8-2a^2\left(-2+a^2+a^4\right)x3^2+\left(-3+2a^2+a^4\right)x3^4},\right.$$
$$\left.\frac{2\,a\,ca\sqrt{1-ca^2}}{ca^2-a^2\left(-1+ca^2\right)}==-\left(\left(\sqrt{(a-x3)\,(a+x3)}\,\left(a^8-2a^2\left(2-a^2+a^4\right)x3^2+\left(-1+a^2\right)^2x3^4\right)\right)\Big/\left(a^9-2a^3\left(-2+a^2+a^4\right)x3^2+a\left(-3+2a^2+a^4\right)x3^4\right)\right)\right\}$$

In[1785]:= `pentSols = Solve[Join[pentSys, {0 ≤ ca ≤ 1, a > 1}], {ca, x3}, Reals]`

Solve::svars : Equations may not give solutions for all "solve" variables. >>

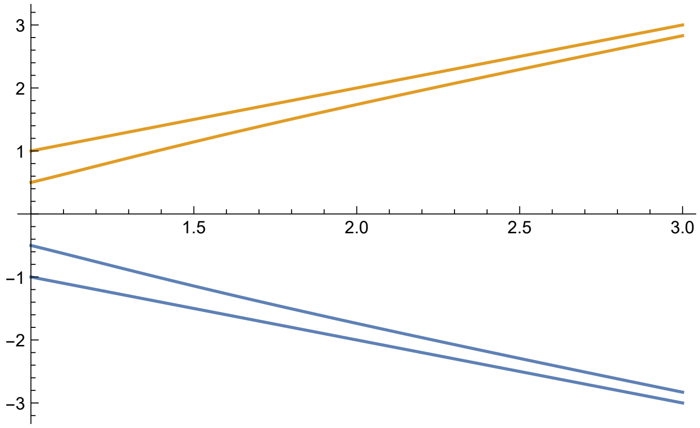
Out[1785]=
$$\Big\{\Big\{ca\to\text{ConditionalExpression}\Big[\sqrt{\left(\left(a^{11}+4a^8x3-a^{10}x3+4a^5x3^2-2a^7x3^2-2a^9x3^2-6a^6x3^3+2a^8x3^3-3a^3x3^4+2a^5x3^4+a^7x3^4-a^2x3^5+2a^4x3^5-a^6x3^5\right)\Big/\left(a^9+a^{11}-4a^6x3+5a^8x3-a^{10}x3+4a^3x3^2+2a^5x3^2-4a^7x3^2-2a^9x3^2+6a^4x3^3-8a^6x3^3+2a^8x3^3-3a\,x3^4-a^3x3^4+3a^5x3^4+a^7x3^4+x3^5-3a^2x3^5+3a^4x3^5-a^6x3^5\right)\right)},$$
$$\left(-a<x3<\frac{a}{-1+a^2}-\sqrt{\frac{a^2-a^4+a^6}{\left(-1+a^2\right)^2}}\ \&\&\ a>1\right)\ ||\ \left(-\frac{a}{-1+a^2}+\sqrt{\frac{a^2-a^4+a^6}{\left(-1+a^2\right)^2}}<x3<a\ \&\&\ a>1\right)\Big]\Big\},$$
$$\{ca\to\text{ConditionalExpression}[0,a>1],\ x3\to\text{ConditionalExpression}[a,a>1]\},$$
$$\left\{ca\to\text{ConditionalExpression}[0,a>1],\ x3\to\text{ConditionalExpression}\left[\frac{a}{-1+a^2}-\sqrt{\frac{a^2-a^4+a^6}{\left(-1+a^2\right)^2}},a>1\right]\right\},$$
$$\{ca\to\text{ConditionalExpression}[1,a>1],\ x3\to\text{ConditionalExpression}[-a,a>1]\},$$
$$\left\{ca\to\text{ConditionalExpression}[1,a>1],\ x3\to\text{ConditionalExpression}\left[-\frac{a}{-1+a^2}+\sqrt{\frac{a^2-a^4+a^6}{\left(-1+a^2\right)^2}},a>1\right]\right\}\Big\}$$

These solutions are unfortunately for triangles and not for pentagons!

In[1786]:= `pentX3sols = FullSimplify[x3 /. Normal[pentSols], a > 1]`

Out[1786]=
$$\left\{x3,\ a,\ \frac{a-a\sqrt{1-a^2+a^4}}{-1+a^2},\ -a,\ \frac{a^3}{1+\sqrt{1-a^2+a^4}}\right\}$$

In[1787]:=
```
Show[{Plot[Evaluate@Part[pentX3sols, {3, 5}], {a, 1, 3}, PlotRange → All],
  Plot[{-x, x}, {x, 1, 3}]}]
```

Out[1787]=


Take 2nd cosine solution as most reasonable

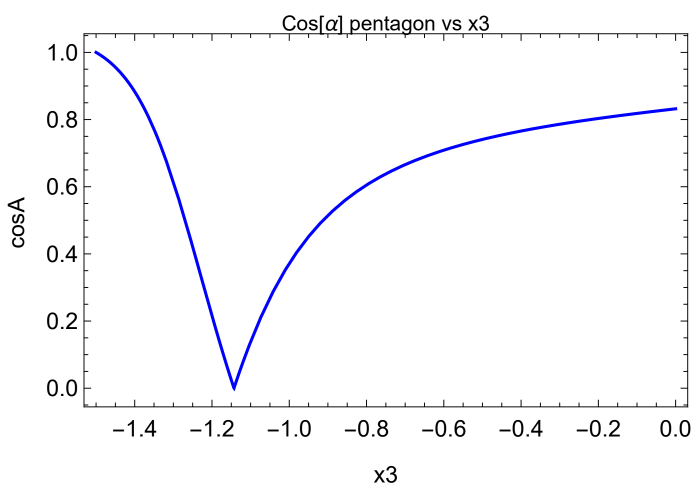
In[1788]:=
```
pentCaSol = (ca /. Normal@Solve[pentSys, {ca, x3}, Reals]);
```

Solve::svars : Equations may not give solutions for all "solve" variables. >>

In[1789]:=
```
(# /. {a -> 1.5, x3 -> -1.4}) & /@pentCaSol
```

Out[1789]=
```
{-0.883114, 0.883114, -1, -1, -1, 0, 0, 0, 1, 1., 1.}
```

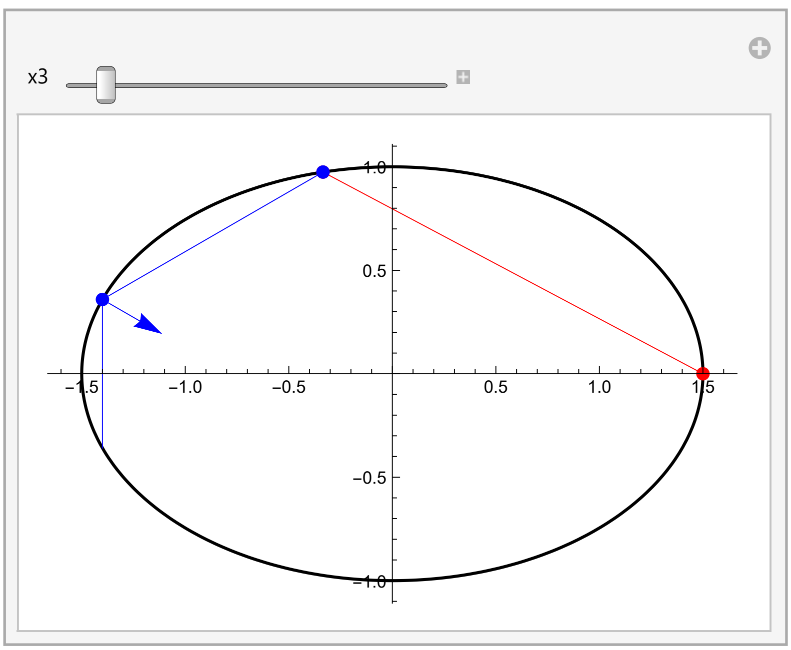
In[1790]:=
```
Plot[pentCaSol[[2]] /. {a → 1.5, x3 → theX3},
 {theX3, -1.5, 0}, PlotStyle → {Red, Blue},
 Frame → True, FrameStyle → Medium, FrameLabel → {"x3", "cosA"},
 PlotLabel → "Cos[α] pentagon vs x3"]
```

Out[1790]=


In[1791]:=
```
Clear@showPentEqn;
showPentEqn[theA_, theX3_] :=
 Module[{gr, theCa, pe, ell, p1, p3, p3n, p2fr1, n3, p2fr3, lgt = .33},
  theCa = (pentCaSol[[2]] /. {a → theA, x3 → theX3});
  pe = pentEqn /. {a → theA, ca → theCa, x3 → theX3};
  p1 = "p1" /. pe;
  p2fr1 = "p2fr1" /. pe;
  p3 = "p3" /. pe;
  n3 = norm["n3" /. pe];
  p3n = flipY[p3];
  p2fr3 = "p2fr3" /. pe;
  ell = plotEll[theA];
  gr = Graphics[{PointSize@Large,
     {Red, Point@p1, Point@p2fr1, Line[{p1, p2fr1}]},
     {Blue, Point@p3, Line[{p3n, p3}], Arrow[{p3, p3 + lgt * n3}],
      Point@p2fr3, Line[{p3, p2fr3}]}}];
  Show[{ell, gr}]]
```

In[1793]:=
```
Manipulate[showPentEqn[1.5, x3], {{x3, -1.4}, -1.5, 0, .01}]
```

Out[1793]=



Second Trial : p1, p2, p3, p4. p4x == 0

In[1794]:=
```
Clear@pentEqn2;
pentEqn2[a_, ca_] = Quiet@Module[{p1, r1, p2, p3, p4, p34eqn},
    p1 = {a, 0};
    r1 = {-ca, Sqrt[1 - ca^2]};
    p2 = FullSimplify[ellInterRayUnprot[a, p1, r1][[2]], a > 1 && 0 <= ca <= 1];
    p3 = FullSimplify[getInterRefl[a, p1, p2], a > 1 && 0 <= ca <= 1];
    p4 = FullSimplify[getInterRefl[a, p2, p3], a > 1 && 0 <= ca <= 1];
    (*pnext=ellInterRayUnprot[a,pto,theRefl][[2]];*)
    p34eqn = FullSimplify[p3[[1]] - p4[[1]] == 0, a > 1 && 0 <= ca <= 1];
    {"p1" → p1,
     "p2" → p2,
     "p3" → p3,
     "p4" → p4,
     "p34eqn" → p34eqn}];
```

In[1796]:=
```
Clear@solPentCa; solPentCa = ca /. Solve[("p34eqn" /. pentEqn2[a, ca]) /.
    {a^2 → a2, a^4 → a2^2, a^6 → a2^3, a^8 → a2^4, a^10 → a2^5, a^12 → a2^6}, ca];
```

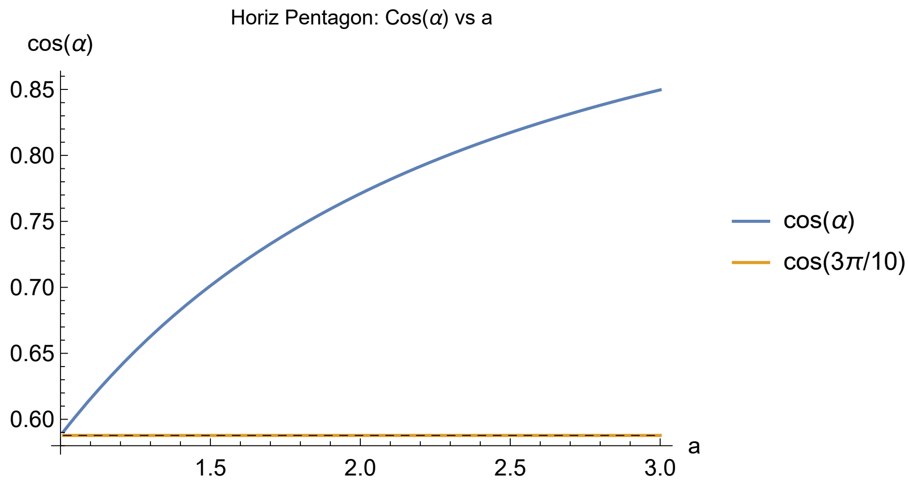
In[1797]:=
```
solPentCa
```

Out[1797]=
```
{0, -√Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
      (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 1],
 √Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
     (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
     (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
     (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 1],
 -√Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
      (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 2],
 √Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
     (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
     (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
     (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 2],
 -√Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
      (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 3],
 √Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
     (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
```

```
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 3],
  -√Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
       (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
       (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
       (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 4],
  √Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
      (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 4],
  -√Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
       (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
       (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
       (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 5],
  √Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
      (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 5],
  -√Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
       (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
       (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
       (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 6],
  √Root[5 a2^6 + (-10 a2^5 - 10 a2^6) #1 + (-9 a2^4 + 34 a2^5 - 9 a2^6) #1^2 +
      (36 a2^3 - 36 a2^4 - 36 a2^5 + 36 a2^6) #1^3 + (-29 a2^2 + 4 a2^3 + 50 a2^4 + 4 a2^5 - 29 a2^6) #1^4 +
      (6 a2 + 14 a2^2 - 20 a2^3 - 20 a2^4 + 14 a2^5 + 6 a2^6) #1^5 +
      (1 - 6 a2 + 15 a2^2 - 20 a2^3 + 15 a2^4 - 6 a2^5 + a2^6) #1^6 &, 6]}
```

In[1798]:=
```
Module[{c3p10 = Cos[3 π / 10.]},
  Plot[{Select[N[solPentCa /. {a2 → (a)^2}], # ∈ Reals &][[3]], c3p10},
   {a, 1.01, 3}, Epilog → {Dashed, Line[{{1.01, c3p10}, {3, c3p10}}]},
   PlotLabel → "Horiz Pentagon: Cos(α) vs a", AxesStyle → Medium,
   AxesLabel → {"a", "cos(α)"}, PlotLegends → {"cos(α)", "cos(3π/10)"}]]
```

Out[1798]=



In[1799]:= **N@Cos[3 π / 10]**

Out[1799]= 0.587785

In[1800]:= **pentEqn2pi4 = pentEqn2[a, Cos[π / 4]]; pentEqn2pi4 // ColumnForm**

Out[1800]=

$$p1 \to \{a,\ 0\}$$

$$p2 \to \left\{\frac{a\left(\frac{1}{2}-\frac{a^2}{2}\right)}{-\frac{1}{2}-\frac{a^2}{2}},\ \frac{a}{\frac{1}{2}+\frac{a^2}{2}}\right\}$$

$$p3 \to \left\{\frac{a\left(\frac{1}{16}-\frac{a^2}{4}+\frac{11 a^4}{8}-\frac{9 a^6}{4}+\frac{a^8}{16}\right)}{\frac{1}{16}+\frac{a^2}{4}-\frac{13 a^4}{8}+\frac{9 a^6}{4}+\frac{a^8}{16}},\ \frac{2 a\left(-\frac{1}{2}-\frac{a^2}{2}\right)\left(-\frac{1}{4}+a^2-\frac{3 a^4}{4}\right)}{\frac{1}{16}+\frac{a^2}{4}-\frac{13 a^4}{8}+\frac{9 a^6}{4}+\frac{a^8}{16}}\right\}$$

$$p4 \to \left\{\frac{a\left(\frac{1}{512}-\frac{9 a^2}{512}+\frac{33 a^4}{128}-\frac{189 a^6}{128}-\frac{895}{128}\left(-1+\frac{1}{\sqrt{2}}\right)\left(1+\frac{1}{\sqrt{2}}\right)a^8+\frac{863}{128}\left(-1+\frac{1}{\sqrt{2}}\right)\left(1+\frac{1}{\sqrt{2}}\right)a^{10}+\frac{237 a^{12}}{128}-\frac{401 a^{14}}{128}+\frac{1225 a^{16}}{512}-\frac{a^{18}}{512}\right)}{\left(-\frac{1}{2}-\frac{a^2}{2}\right)\left(\frac{1}{256}+\frac{a^2}{32}-\frac{41 a^4}{64}+\frac{127 a^6}{32}+\frac{1437}{64}\left(-1+\frac{1}{\sqrt{2}}\right)\left(1+\frac{1}{\sqrt{2}}\right)a^8+\frac{503 a^{10}}{32}-\frac{745 a^{12}}{64}+\frac{153 a^{14}}{32}+\frac{a^{16}}{256}\right)},\ -\frac{a\left(\frac{-1+a}{\sqrt{2}}+a\right)\left(\frac{1}{\sqrt{2}}+a+\right.}{}$$

$$p34eqn \to \frac{5 a^{12}}{\sqrt{2}}+\frac{\left(-1+a^2\right)^6}{64\sqrt{2}}+\frac{9 a^6\left(-1+a^2\right)^2\left(1+a^2\right)}{2\sqrt{2}}+\frac{a^2\left(-1+a^2\right)^2\left(1+a^2\right)\left(3+a^2\right)\left(1+3 a^2\right)}{16\sqrt{2}}+\frac{a^8\left(-9+34 a^2-9 a^4\right)}{4\sqrt{2}} == \frac{5 a^{10}\left(1+a^2\right)}{\sqrt{2}}+$$

A solução para o angulo que faz cosA = 0 é uma raíz de sextica no quadrado de a :

In[1801]:= **pentEqn2pi4p34x = FullSimplify["p34eqn" /. pentEqn2pi4, a > 0] /.**
{a^2 → a2, a^4 → a2^2, a^6 → a2^3, a^12 → a2^6}

Out[1801]= $1 + 6\ a2 + 244\ a2^3 + 41\ a2^6 == a2^2\left(73 + 257\ a2^2 + 26\ a2^3\right)$

In[1802]:= **a2Sols = a2 /. Solve[pentEqn2pi4p34x, a2, Reals]**

Out[1802]= {Root[1 + 6 #1 − 73 #1^2 + 244 #1^3 − 257 #1^4 − 26 #1^5 + 41 #1^6 &, 1],
Root[1 + 6 #1 − 73 #1^2 + 244 #1^3 − 257 #1^4 − 26 #1^5 + 41 #1^6 &, 2],
Root[1 + 6 #1 − 73 #1^2 + 244 #1^3 − 257 #1^4 − 26 #1^5 + 41 #1^6 &, 3],
Root[1 + 6 #1 − 73 #1^2 + 244 #1^3 − 257 #1^4 − 26 #1^5 + 41 #1^6 &, 4]}

In[1803]:= **Sqrt /@ N[a2Sols, 10]**

Out[1803]= {1.6310422658 ⅈ, 0.2762409560 ⅈ, 0.7079534549, 1.5343675814}

Hex Eqn

In[1804]:=
```
Clear@HexEqn;
HexEqn[a_, ca_] = Quiet@Module[{p1, r1, p2, p3, p23eqn},
    p1 = {a, 0};
    r1 = {-ca, Sqrt[1 - ca^2]};
    p2 = FullSimplify[ellInterRayUnprot[a, p1, r1][[2]], a > 1 && 0 <= ca <= 1];
    p3 = FullSimplify[getInterRefl[a, p1, p2], a > 1 && 0 <= ca <= 1];
    p23eqn = FullSimplify[p2[[2]] - p3[[2]] == 0, a > 1 && 0 <= ca <= 1];
    {"p1" → p1,
     "p2" → p2,
     "p3" → p3,
     "p23eqn" → p23eqn}];
```

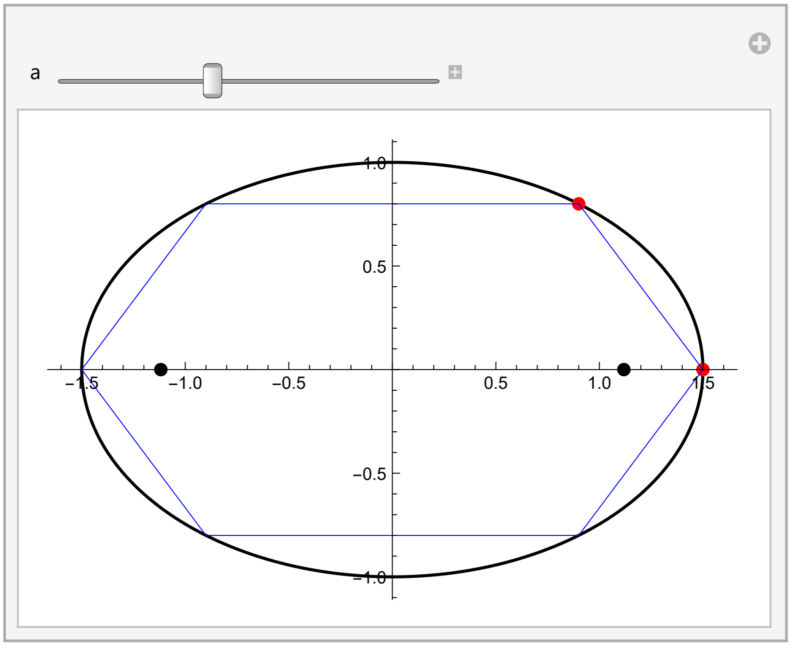
In[1806]:=
```
Solve["p23eqn" /. HexEqn[a, ca], ca]
```

Out[1806]= $\left\{\{ca \to -1\}, \{ca \to 0\}, \{ca \to 1\}, \left\{ca \to \frac{a}{1+a}\right\}\right\}$

In[1807]:=
```
Clear@showHexEqn;
showHexEqn[a_] := Module[{fs, ca, p1, r1, p2, ell, gr, bounce, lgt = .33},
  fs = getFoci[a];
  p1 = {a, 0};
  ca = a/(1 + a);
  r1 = {-ca, Sqrt[1 - ca^2]};
  p2 = ellInterRayUnprot[a, p1, r1][[2]];
  ell = plotEll[a];
  bounce = bounceRay[a, p1, p2, 5];
  gr = Graphics[{PointSize@Large,
     {Black, Point@fs},
     {Red, Point@p1, Point@p2},
     {Blue, Line@bounce}}];
  Show[{ell, gr}]]
```

```
In[1809]:= Manipulate[showHexEqn[a], {{a, 1.5}, .5, 3, .01}]
```

Out[1809]=



Oct Eqn

```
In[1810]:= Clear@OctEqn;
OctEqn[a_, ca_] = Quiet@Module[{p1, r1, p2, p3, p3eqn},
    p1 = {a, 0};
    r1 = {-ca, Sqrt[1 - ca^2]};
    p2 = FullSimplify[ellInterRayUnprot[a, p1, r1][[2]], a > 1 && 0 <= ca <= 1];
    p3 = FullSimplify[getInterRefl[a, p1, p2], a > 1 && 0 <= ca <= 1];
    p3eqn = FullSimplify[p3[[1]] == 0, a > 1 && 0 <= ca <= 1];
    {"p1" → p1,
     "p2" → p2,
     "p3" → p3,
     "p3eqn" → p3eqn}];

In[1812]:= Clear@octSols; octSols[a_] = ca /. Solve["p3eqn" /. OctEqn[a, ca], ca];

In[1813]:= Select[octSols[5.], # ∈ Reals &]

Out[1813]= {-0.680397, 0.680397, -0.999998, 0.999998}
```

```
In[1814]:= Clear@HexEqn;
HexEqn[a_, ca_] = Quiet@Module[{p1, r1, p2, p3, p23eqn},
    p1 = {a, 0};
    r1 = {-ca, Sqrt[1 - ca^2]};
    p2 = FullSimplify[ellInterRayUnprot[a, p1, r1][[2]], a > 1 && 0 <= ca <= 1];
    p3 = FullSimplify[getInterRefl[a, p1, p2], a > 1 && 0 <= ca <= 1];
    p23eqn = FullSimplify[p2[[2]] - p3[[2]] == 0, a > 1 && 0 <= ca <= 1];
    {"p1" → p1,
     "p2" → p2,
     "p3" → p3,
     "p23eqn" → p23eqn}];
```

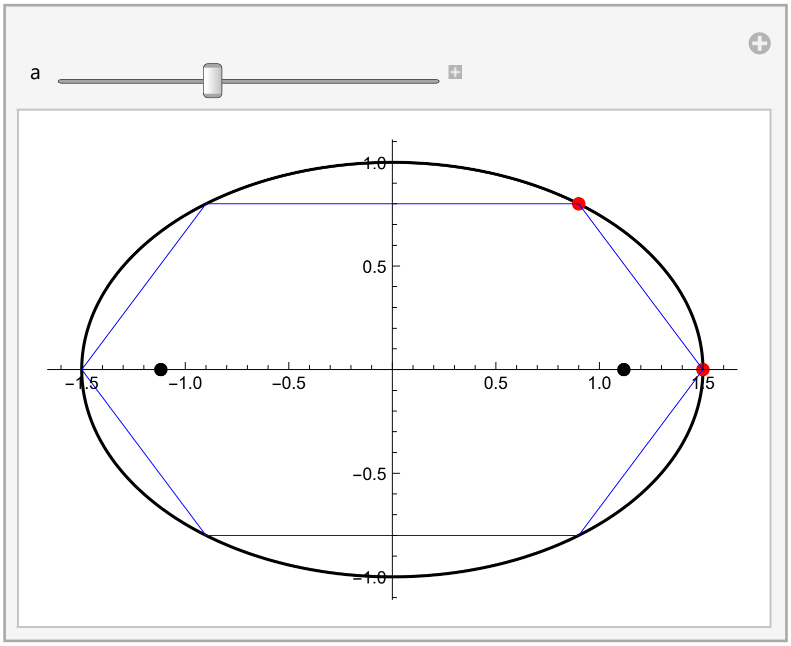
```
In[1816]:= Solve["p23eqn" /. HexEqn[a, ca], ca]
```

Out[1816]= $\left\{\{ca \to -1\}, \{ca \to 0\}, \{ca \to 1\}, \left\{ca \to \frac{a}{1 + a}\right\}\right\}$

```
In[1817]:= Clear@showOctEqn;
showOctEqn[a_] := Module[{fs, ca, p1, r1, p2, ell, gr, bounce, lgt = .33},
   fs = getFoci[a];
   p1 = {a, 0};
   ca = a/(1 + a);
   r1 = {-ca, Sqrt[1 - ca^2]};
   p2 = ellInterRayUnprot[a, p1, r1][[2]];
   ell = plotEll[a];
   bounce = bounceRay[a, p1, p2, 5];
   gr = Graphics[{PointSize@Large,
      {Black, Point@fs},
      {Red, Point@p1, Point@p2},
      {Blue, Line@bounce}}];
   Show[{ell, gr}]]
```

In[1819]:=
```
Manipulate[showHexEqn[a], {{a, 1.5}, .5, 3, .01}]
```

Out[1819]=



Make Movies I

In[1171]:=
```
Clear@makeMovie; makeMovie[fname_, frames_, repeats_: 3] := Module[{framesConcat},
   framesConcat = Flatten@ConstantArray[frames, 3];
   Print["Exporting " <> ToString@Length[framesConcat] <> " frames to " <> fname];
   Export[fname, framesConcat, "VideoEncoding" → "MPEG-4 Video"];
  ];
```

Quadrangle

```
doImgsQuadrangle = False;
If[doImgsQuadrangle,
  Module[{imgs, op},
   imgs = Table[Show[showOneQuad[quadAlphaT15, i, True, True], ImageSize → 800],
     {i, Length["alphas" /. quadAlphaT15]}];
   makeMovie["quadrangle3.mov", imgs]]];
```

```
Exporting 1083 frames to quadrangle3.mov
```

Pentagon

```
doImgPents = False;
If[doImgPents, Module[{imgs, op},
  imgs = Table[Show[{showOnePent[pentAlphaT15, i, pentNotableLoci, True, True]},
     ImageSize → 800], {i, Length["alphas" /. pentAlphaT15]}];
  makeMovie["pents2.mov", imgs]]];
```

```
doImgPents124 = False;
If[doImgPents124, Module[{imgs, op},
  imgs = Table[
    Show[{showOnePent[pentAlphaT15, i, pentNotableLoci124, True, True, {1, 2, 4}]},
     ImageSize → 800], {i, Length["alphas" /. pentAlphaT15]}];
  makeMovie["pents2_124.mov", imgs]]];
```

```
Exporting 1083 frames to hepts2_124.mov
```

```
doImgPentsBoth = False;
If[doImgPentsBoth, Module[{imgs, op},
  imgs = Table[GraphicsRow[{
      showOnePent[pentAlphaT15, i, pentNotableLoci, True, True, {1, 2, 3}],
      showOnePent[pentAlphaT15, i, pentNotableLoci124, True, True, {1, 2, 4}]},
     ImageSize → 1200],
    {i, Length["alphas" /. pentAlphaT15]}];
  makeMovie["pents2_123_side_124.mov", imgs]]];
```

```
Exporting 1083 frames to hepts2_123_side_124.mov
```

```
doImgPentVert = True;
If[doImgPentVert, Module[{imgs, op},
  imgs = Join@{
     Table[Show[showOnePentVert[pentAlphaT15, i, pentVertLoci], ImageSize → 800],
      {i, Length["alphas" /. pentAlphaT15]}],
     Table[Show[showOnePentVert[pentAlphaT15, i, pentVertLoci, False],
       ImageSize → 800], {i, Length["alphas" /. pentAlphaT15]}]};
  makeMovie["pentVert2.mov", imgs]]];
```

```
Exporting 2160 frames to heptVert2.mov
```

Hexagon

```
doImgHex = True;
If[doImgHex, Module[{imgs, op},
   imgs = Table[
     Show[showOnePoly[hexAlphaT15, i, hexNotableLoci135, getHexVtx0, hexErrorP,
       drNotables → True, drLoci → True, drCentroids → True, drCentroidLabels → False,
       vtx -> {1, 3, 5}], PlotRange → {{-2, 2}, {-1.5, 1.5}},
      ImageSize → 800], {i, Length["alphas" /. hexAlphaT15]}];
   makeMovie["hexOrbits.mov", imgs]]];
```

```
Exporting 1080 frames to hexOrbits.mov
```

Pencil of Caustics

```
doImgPencil = False;
If[doImgPencil, Module[{imgs, op},
   imgs =
    Table[Show[drawCausticTangs[1.5, tDeg], ImageSize → 800], {tDeg, 0, 359, .25}];
   makeMovie["pencilOfCaustics.mov", imgs]]];
```

```
Exporting 4311 frames to pencilOfCaustics.mov
```

```
doImgPencilCircle = False;
If[doImgPencilCircle, Module[{imgs, op},
  imgs = Table[Show[drawCausticTangs[a, 0, nmax → 9, drLeftCircle → True],
     ImageSize → 800, PlotRange → {{-2, 2}, {-1.1, 1.1}}], {a, .7, 2, .01}];
  makeMovie["circleOfTangents.mov", Join[imgs, Reverse@imgs]]]]
```

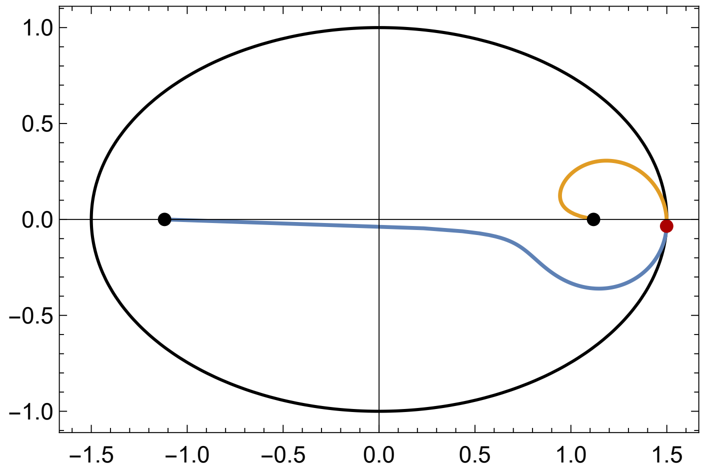
```
Exporting 786 frames to circleOfTangents.mov
```

Locus of Tangents to Confocals

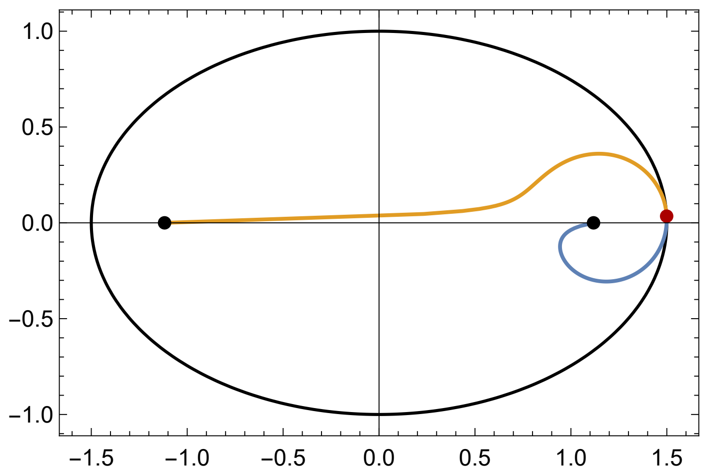
```
doImgConfocalLoci = False;
If[doImgConfocalLoci, Module[{imgs, op},
  imgs = Table[Show[showOneCTC[1.5, 1, tDeg], ImageSize → 800], {tDeg, -45, 45, .1}];
  makeMovie["lociOfConfocalTangentsShort2.mov", Join[imgs, Reverse@imgs]]]]
```

```
Exporting 5406 frames to lociOfConfocalTangentsShort2.mov
```

```
showOneCTC[1.5, 1, -2]
```




```
showOneCTC[1.5, 1, 2]
```



Tangent Locus vs a

```
doImgTangLocusVsA = False;
If[doImgTangLocusVsA, Module[{imgs, op},
  imgs = Table[tangentPathFrame[a, 5, -45, 3], {a, 1, 3, .01}];
  makeMovie["imgTangLocusVsA2.mov", Join[imgs, Reverse@imgs]]]]
```

```
Exporting 1206 frames to imgTangLocusVsA2.mov
```

Self-Inter Quad

```
doQuadSelf = False;
If[doQuadSelf, Module[{a = 1.5, x1max, imgs},
   x1max = maxX1QuadSelfInter[a][[2]];
   imgs = Show[showQuadSelfInter[a, #], ImageSize → 800] & /@
     Range[-x1max, x1max, (2. x1max) / 360];
   makeMovie["quadself15.mov", Join[imgs, Reverse@imgs]]]];
```

```
Exporting 2166 frames to quadself15.mov
```

Pentagram

```
doImgPentagram = False;
If[doImgPentagram, Module[{imgs, op},
  imgs = pentagramFrames[1.15, 1];
  makeMovie["pentagram115.mov", Join[imgs, Reverse@imgs]]]]
```

```
Exporting 2166 frames to pentagram115.mov
```