$SU_5 \supset SU_4 \otimes U_1$ A4 Group Orbits & Their Polytope Hulls Using Quaternions

While verifying Koca' s work (et al.)

4 D Polytopes and Their Dual Polytopes of the Coxeter Group Represented by Quaternions

A note about using octonion math for quaternions....

Ignore the extra {0,0,0,0} at the end of these lists. I actually use octonion multiplication indicated by the SmallCircle (∘) below.

This is an 8D construct useful in further work with E8. The key for doing multiplication is to use one of the 480 possible octonions with the first (of 7) triads in the Fano plane being (123), which makes all upper-left quadrant multiplication a quaternion multiplication. I do this because my octonion→quaternion multiplication handles symbolics better than the built-in MTM quaternion code.

FYI - I chose $\varphi \to \left(\sqrt{5} - 1\right) / 2$=0.618... (small golden ratio vs. large 1.618...).

We need to pick an octonion that has the first triad = {1, 2, 3} (i.e. strict quaternion),
and strict Cayley Dickson construction where e_4 to e_7 quadrant multiplication remains within the quadrant.

So we find the one (of 48) with first triad = {1, 2, 3} and the one with Cayley – Dickson construction.

In[∘]:=

```
flip = True;
allOcts = Table[setFM[i, flip, 0]; triads, {i, 240}];
fn = position[allOcts, {{1, 2, 3}, {1, 4, 7}, {1, 6, 5}, {2, 4, 6}, {2, 5, 7}, {3, 5, 4}, {3, 6, 7}}]
```

Out[∘]=

```
85
```

In[ ]:=

```
(* Koca uses σ and τ for Golden ratios, but I use φ as described in the FYI above.
  This sets up the symbolic relationships. *)
Clear[σ, Φ, τ];
σ = -φ;
τ = Φ = 1 / φ;
```

In[ ]:=

```
(* This sets the type of octonion split for negative split numbers (where Abs@split selects a triad):
  False implies the first of 4 non-triad imaginary dimensions becomes real (or +1), giving a 1+3i + 1+3i octonion
  True implies the first 3 become real, giving a 1+3i + 3+1i octonion **)
split3Real = False;
setFM[(*)85**)fn(**), flip, split = 0];
triads
Row@{fanoPlane, fmDispN}
```

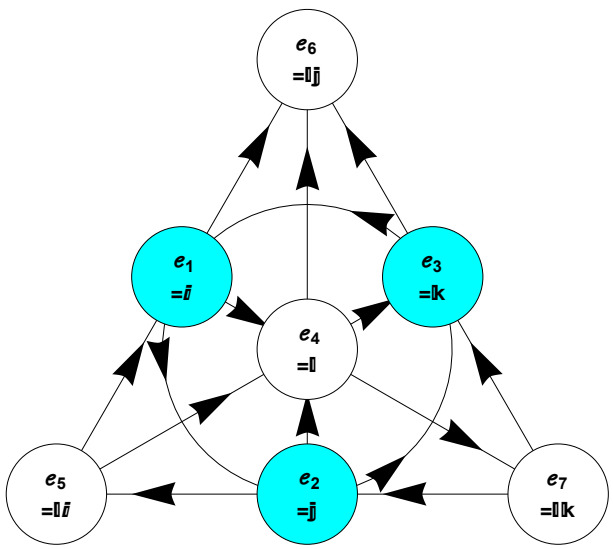
Out[ ]=

{{1, 2, 3}, {1, 4, 7}, {1, 6, 5}, {2, 4, 6}, {2, 5, 7}, {3, 5, 4}, {3, 6, 7}}

Out[ ]=

$$\begin{pmatrix} 0\mapsto 1 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & -1 & 3 & -2 & 7 & -6 & 5 & -4 \\ 2 & -3 & -1 & 1 & 6 & 7 & -4 & -5 \\ 3 & 2 & -1 & -1 & -5 & 4 & 7 & -6 \\ 4 & -7 & -6 & 5 & -1 & -3 & 2 & 1 \\ 5 & 6 & -7 & -4 & 3 & -1 & -1 & 2 \\ 6 & -5 & 4 & -7 & -2 & 1 & -1 & 3 \\ 7 & 4 & 5 & 6 & -1 & -2 & -3 & -1 \end{pmatrix}$$

Octonion Fano Plane



Define the Permutations

```
(* These are the 15 binary orbit permutations using {0,1}'s *)
perms4D = Flatten[Permutations /@ {{1, 0, 0, 0}, {1, 1, 0, 0}, {1, 1, 1, 0}, {1, 1, 1, 1}}, 1];
```

In[ ]:=

```
{#, perms4D[[#]]} & /@ Range@15 // MatrixForm
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} 1 & \{1, 0, 0, 0\} \\ 2 & \{0, 1, 0, 0\} \\ 3 & \{0, 0, 1, 0\} \\ 4 & \{0, 0, 0, 1\} \\ 5 & \{1, 1, 0, 0\} \\ 6 & \{1, 0, 1, 0\} \\ 7 & \{1, 0, 0, 1\} \\ 8 & \{0, 1, 1, 0\} \\ 9 & \{0, 1, 0, 1\} \\ 10 & \{0, 0, 1, 1\} \\ 11 & \{1, 1, 1, 0\} \\ 12 & \{1, 1, 0, 1\} \\ 13 & \{1, 0, 1, 1\} \\ 14 & \{0, 1, 1, 1\} \\ 15 & \{1, 1, 1, 1\} \end{pmatrix}$$

In[ ]:=

```
(* Functions to generate positional and ± sign permutations of lists *)
EvenPermutationQ@p_ := Signature@p == -1;
OddPermutationQ@p_ := Signature@p == 1;

pm@n_ := Flatten[Outer[List, Sequence @@ Table[{-1, 1}, {n}]], n - 1];
pm2@n_ := pm@n / 2;
```

In[ ]:=

```
EvenPermutations@p_List := Module[{(**)i, j(**)},
   i = Range@Length@p;
   j = Permutations@i;
   (j /. Inner[Rule, i, p, List])[[Flatten@Position[j, _?EvenPermutationQ, 1, Heads → False]]];
OddPermutations@p_List := Module[{(**)i, j(**)},
   i = Range@Length@p;
   j = Permutations@i;
   (j /. Inner[Rule, i, p, List])[[Flatten@Position[j, _?OddPermutationQ, 1, Heads → False]]];
```

In[ ]:=

```
Eperms@in_List := EvenPermutations[Inner[Times, in, Array[1 &, Length@in], List]];
Operms@in_List :=  OddPermutations[Inner[Times, in, Array[1 &, Length@in], List]];
```

```
(* Quaternion permutations in 3 or 4D Code in 14 types of permutations:
  "NoneNoSign",                 no permutation of position or sign,
  "None",                       no permutation of position (just ± sign permutations on each position),
  "NoneEsign",                  no permutation of position (just Even ± sign permutations on each position),
  "NoneOsign",                  no permutation of position (just Odd ± sign permutations on each position),
  "AllNoSign",                  all permutations of position and no ± sign permutations (just like Permutations[]),
  "All",                        full set of ± sign and positional permutations,
  "Rotate",                     RightRotations of {e1,e2,e3} with ± permutations on each position (cyclic with sign),
  "NoSign",                     no sign permutations, just RightRotation of positions            (cyclic only),
  "NoSignEpos" & "NoSignOpos", no sign permutations,  Even (or Odd) positional permutations,
  "Epos"       & "Opos",       full set of ± sign and Even (or Odd) positional permutations,
  "Esign"      & "Osign",      ± sign Permutation by count of Even (or Odd) + signs,
  "EsignEpos"  & "EsignOpos" & "OsignEPos"   & "OsignOPos"
                               ± sign Permutation by count of Even (or Odd) + signs and Even (or Odd) position permutations *)
permRotate@in_List := Module[{out = in},
   Join[{out}, (out = RotateRight@out) & /@ Range[Length@in - 1]]];
```

```
perms[in_List, select_List : {}, permType_String : "All"] := Module[{inLength, noSelect, signPerms, permList, outList},
   inLength = Length@in;
   (* Set noSelect if the permType needs all ± permutations *)
   noSelect = permType == "Epos" || permType == "Opos" || permType == "Rotate" || permType == "All" || permType == "None";
   (* If the permType contains "NoSign", return Array of +1s,
      If the permType doesn't require selecting Even or Odd sign counts, send all signPerms,
          otherwise do the Select *)
   signPerms = If[StringContainsQ[permType, "NoSign"], Array[1 &, inLength],
     If[noSelect, pm@inLength,
      Select[pm@inLength, If[StringContainsQ[permType, "Osign"], OddQ, EvenQ]@Count[Sign@#, 1] &]]];
   (* Do the position perms *)
   permList = Which[
      StringTake[permType, 3] == "Non", List,
      StringContainsQ[permType, "All"] || permType == "Esign" || permType == "Osign", Permutations,
      permType == "Rotate" || permType == "NoSign", permRotate,
      (* Eperms and Operms processes all positional permutations of Even or Odd Signature (+1,-1) *)
      StringContainsQ[permType, "Epos"], Eperms,
      StringContainsQ[permType, "Opos"], Operms]@in;
   (* Apply the sign perms and delete duplicates, not using Union due to it sorting, which may not be desired *)
   outList = DeleteDuplicates@FullSimplify@Flatten[Table[# permList[[i]] & /@ signPerms, {i, Length@permList}], 1];
   (* Apply the select list if needed *)
   outList[[If[select == {}, Range@Length@outList, select]]];
```

In[ ]:=

```
{#, outp = Union@perms[{x, y, z}, (*){1,2},**)#], Length@outp} & /@
  {"AllNoSign", "NoneNoSign", "None", "NoneEsign", "NoneOsign", "All", "Rotate", "NoSign", "NoSignEpos", "NoSignOpos",
   "Epos", "Opos", "Esign", "Osign", "EsignEpos", "EsignOpos", "OsignEpos", "OsignOpos"} // MatrixForm
```

Out[ ]//MatrixForm=

AllNoSign	
NoneNoSign	
None	
NoneEsign	
NoneOsign	
All	{{-x, -y, -z}, {-x, -y, z}, {-x, y, -z}, {-x, y, z}, {-x, -z, -y}, {-x, -z, y}, {-x, z, -y}, {-x, z, y}, {x, -y, -z}, {x, -y, z}, {x, y,
Rotate	
NoSign	
NoSignEpos	
NoSignOpos	
Epos	
Opos	
Esign	
Osign	
EsignEpos	
EsignOpos	
OsignEpos	
OsignOpos	

In[ ]:=

```
(**)
{#, outp = Union@perms[{x, y, z, w}, #], Length@outp} & /@
  {"AllNoSign", "NoneNoSign", "None", "NoneEsign", "NoneOsign", "All", "Rotate", "NoSign", "NoSignEpos", "NoSignOpos",
   "Epos", "Opos", "Esign", "Osign", "EsignEpos", "EsignOpos", "OsignEpos", "OsignOpos"} // MatrixForm
```

Out[ ]//MatrixForm=

(… 1 …)

Full expression not available (original memory size: 373.2 kB)

Set up the quaternion basis elements for W(A4)

```
(* 4D Polytopes and Their Dual Polytopes of the Coxeter Group Represented by Quaternions *)
A4oSet := (
   (* r1,r2,r3,r4,r0 are the generator sequence functions of α1...4, & α0, with  d=aStr<>"@"<>bStr where d^5=1
      where exponents of r use the rCycle function *)
   R1 = aStr = "r1@r3";
   R2 = bStr = "r2@r4";
   R1R2 = dStr = aStr <> "@" <> bStr;

   (* Embedding A3 into A4 5 ways with sets of 3 generators *)
   in = {r1, r2, r3, r4, r0};
   A3inA4 = {in[[ ;; 3]]};
   Do[in = RotateLeft[in]; AppendTo[A3inA4, in[[ ;; 3]]], 4];
   inStr = ToString /@ in;
   A3inA4Str = ToString@# & /@ # & /@ A3inA4;

   (* Set up for doW(in) with α1,α2,α3,α4,α00  eq. 12 *)
   α1 = -1;
   α2 = 1/2 (1 + e1 + e2 + e3);
   α3 = -e1;
   α4 = 1/2 (e1 - σ e2 - τ e3);
   α5 = α00 = - (α1 + α2 + α3 + α4);

   (* W(A4)=prq[I,p,prq[c*,switchστ[I]*,c]] ⊕ prq[I,p,prq[c*,switchστ[I]*,c]]*] with p∈I and c∈T', between eq. 13, 14 and 18,19 *)
   oc0 = (e3 - e2) / Sqrt[2];

   αα = octSimplify[(-σ + e2 + τ e3) / 2];
   ββ = octSimplify@prq[oc0*, switchστ[αα]*, oc0];

   (* the (Cartan) Schlafli Matrix  eq. 15 *)
   csm = csmA4 = {{2, -1, 0, 0}, {-1, 2, -1, 0}, {0, -1, 2, -1}, {0, 0, -1, 2}};

   (* Quaternion & Dual basis vectors  eq. 16 *)
   Clear[ωω, ω0, ω1, ω2, ω3, ω4];
   ωω = {ω0, ω1, ω2, ω3, ω4};
   O1000Str = ToString[# /. ω0 → 0, TraditionalForm] & /@ Inner[Subtract, RotateLeft[ωω], ωω, List];
   O1000Str = ReplacePart[ToString[# /. ω0 → 0, TraditionalForm] & /@ Inner[Subtract, RotateLeft[ωω], ωω, List], 4 → "(ω4-ω3)/2"];
```

```
ω0 = 0;
ωωRule = {
  ω1 → (- √5 + τ e2 - σ e3) / √10 ,
  ω2 → 2 (τ e2 - σ e3) / √10 ,
  ω3 → (- √5 e1 + τ^2 e2 - σ^2 e3) / √10 ,
  ω4 → -2 ( e3 - e2) / √10 (*=-2 oc0/ √5 *)};
ωωList = oct2List /@ ωω /. ωωRule;
O1000 = oct2Quat@# & /@ Inner[Subtract, RotateLeft[ωω /. ωωRule], ωω /. ωωRule, List];

(* Λ Quaternion & Dual basis {α0,β0,γ0,ϱ0} which is used in Λ1[[aa1,aa2,aa3,aa4] which drives xxOct[x1,x2,x3,x4] eq. 19, and aωΛ00 after eq. 19
aωΛ0 := αβγϱω =   {aa1 ω1 ,                    aa2 ω2 ,                    aa3 ω3,                              aa4 ω4};
aωΛ1 := {- (aa1 + aa2) ω1, (aa1 + aa2 + aa3 + aa4) ω2 ,     - (aa2 + aa3 + aa4) ω3,                    (aa2 + aa3) ω4};
aωΛ2 := {- (aa3 + aa4) ω1 ,                    aa3 ω2 ,     - (aa1 + aa2 + aa3) ω3,                              aa1 ω4};
aωΛ3 := {          aa4 ω1 ,     - (aa2 + aa3 + aa4) ω2,                    aa2 ω3 ,                    - (aa1 + aa2) ω4};
aωΛ4 := {  (aa2 + aa3) ω1 ,     - (aa1 + aa2 + aa3) ω2, (aa1 + aa2 + aa3 + aa4) ω3,                    - (aa3 + aa4) ω4};
aωΛOrbit = {aωΛ0, aωΛ1, aωΛ2, aωΛ3, aωΛ4} /. {ω1 → 1, ω2 → 1, ω3 → 1, ω4 → 1};
aωΛ04 = {aωΛ0, aωΛ1, aωΛ2, aωΛ3, aωΛ4} /. ωωRule;
aωΛ = Total /@ aωΛ04;

(* 3D Basis vectors v in terms of ω eq. 21, 22, 25 *)
Clear[v04, v0, v1, v2, v3, v4];
v04 = {v0, v1, v2, v3, v4};
Ov1000Str = ToString[# /. v0 → 0, TraditionalForm] & /@ Inner[Subtract, RotateLeft[v04], v04, List];

v0 = 0;
vvRule = {
  v1 → ω1 - 1 ω4 / 4,
  v2 → ω2 - 2 ω4 / 4,
  v3 → ω3 - 3 ω4 / 4,
  v4 →       ω4 / 4};
vvList = oct2List /@ v04 /. vvRule;
Ov1000 = oct2Quat@# & /@ Inner[Subtract, RotateLeft[v04 /. vvRule /. ωωRule], v04 /. vvRule /. ωωRule, List];

(* eq. 22 - The vΛ alternate 3D quaternion (p0..4) basis {αv0,βv0,γv0,ϱv0} which is used in vΛ1[[aa1,aa2,aa3,aa4]
     which drives xxOct[x1,x2,x3,x4] and avΛ0 after eq. 19  *)
avΛ0 := αβγϱv =  { aa1 v1,                    aa2 v2,                    aa3  v3,            (aa1 + 2 aa2 + 3 aa3 + 4 aa4) v4};
avΛ1 := {          aa1 v1,                    aa2 v2,           (aa3 + aa4) v3,            (aa1 + 2 aa2 + 3 aa3 -  aa4) v4};
avΛ2 := {          aa1 v1,            (aa2 + aa3) v2,                   aa4 v3,            (aa1 + 2 aa2 - 2 aa3 -  aa4) v4};
avΛ3 := { (aa1 + aa2) v1,                   aa3 v2,                   aa4 v3,            (aa1 - 3 aa2 - 2 aa3 -  aa4) v4};
avΛ4 := {          aa2 v1,                    aa3 v2,                   aa4 v3,           (4 aa1 + 3 aa2 + 2 aa3 +  aa4) v4};
avΛOrbit = {avΛ0, avΛ1, avΛ2, avΛ3, avΛ4} /. {v1 → 1, v2 → 1, v3 → 1, v4 → 1};
```

```
av∧04 = {av∧0, av∧1, av∧2, av∧3, av∧4} /. vvRule /. ωωRule;
av∧ = Total /@ av∧04;

(* 3D basis projections *)
p0 = oc0;
p1 = p0∘e1;
p2 = p0∘e2;
p3 = p0∘e3;
(* The following changes to p2 & p3, which generate the Real e0 part of the quaternion basis,
have eq. 28 pp evaluate to the same prescribed values in O1000 (changing dims 1↔2),
but it breaks the 3D hulls when compared to A3 constructions in Archimedian/Catalan solids (**)
  p2= p0∘e3;
  p3=-p0∘e2;**)
p4 := -p0/(2 √5) (aa1 + 2 aa2 + 3 aa3 + 4 aa4);

pp := {p0, p1, p2, p3, p4};

(* eq. 25,26 **)
αavp∧0 := (aa1 - aa3) / 2;
βavp∧0 := (aa1 + aa3) / 2;
γavp∧0 := (aa1 + 2 aa2 + aa3) / 2;
ϱavp∧0 := 1;
αβγϱavp∧0 := {αavp∧0, βavp∧0, γavp∧0, ϱavp∧0};

avp := αβγϱavp∧0 pp[[2 ;;]];

(*)This is the octonion→quaternion that gets used in av∧ by using αβγϱv **)
αβγϱv0 = {αv0, βv0, γv0, ϱv0} = oct2Quat@Total[αβγϱv /. vvRule /. ωωRule];

(* This is the normal octonion→quaternion ω basis that gets used in aω∧ *)
αβγϱ0 = {α0, β0, γ0, ϱ0} = oct2Quat@Total@αβγϱω /. ωωRule);
```

In[ ]:=

```
(* Resolving the {aa1,aa2,aa3} using Weyl orbit weights *)
aa := {aa1, aa2, aa3, aa4};
aRule@orbit_List := Inner[Rule, aa, PadRight[orbit, 4], List];
```

In[ ]:=

```
doW@"A4"
Total@% /. aRule@{1, 0, 0, 0}
oct2List@%
octonion@%
```

Out[ ]=

$$\left\{-\sqrt{\frac{2}{5}}\ \text{aa4}\ (-e_2+e_3)+\sqrt{\frac{2}{5}}\ \text{aa2}\left(\frac{e_2}{\varphi}+\varphi\, e_3\right)+\frac{\text{aa1}\left(-\sqrt{5}+\frac{e_2}{\varphi}+\varphi\, e_3\right)}{\sqrt{10}}+\frac{\text{aa3}\left(-\sqrt{5}\ e_1+\frac{e_2}{\varphi^2}-\varphi^2\, e_3\right)}{\sqrt{10}},\ 0,\ 0,\ 0\right\}$$

Out[ ]=

$$\frac{-\sqrt{5}+\frac{e_2}{\varphi}+\varphi\, e_3}{\sqrt{10}}$$

Out[ ]=

$$\left\{-\frac{1}{\sqrt{2}},\ 0,\ \frac{1}{\sqrt{10}\ \varphi},\ \frac{\varphi}{\sqrt{10}},\ 0,\ 0,\ 0,\ 0\right\}$$

Out[ ]=

$$-\frac{1}{\sqrt{2}}+\frac{e_2}{\sqrt{10}\ \varphi}+\frac{\varphi\, e_3}{\sqrt{10}}$$

In[ ]:=

```
O1000;
(* The function biQuaternion takes a 4D quaternion and creates a (L)eft/(R)ight (and/or split) octonion .
  Here R is {0,0,0,0} due to the input being Real (ℝ).*)
O1000octList = biQuaternion /@ %;
O1000oct = octonion /@ %;
O1000ListRnd = rndMat[# /. φωRep] & /@ %%;
O1000octRnd = octonion /@ %%;
checkVertices[O1000oct, True, True, True, True, True, False]
```

Out[ ]=

List length= 5 and it is symbolic octonion

$$\text{Symbolic}=\begin{pmatrix} 1 & -\frac{1}{\sqrt{2}}+\frac{e_2+\varphi^2 e_3}{\sqrt{10}\,\varphi} \\ 2 & \frac{1}{\sqrt{2}}+\frac{e_2+\varphi^2 e_3}{\sqrt{10}\,\varphi} \\ 3 & -\frac{e_1}{\sqrt{2}}+\frac{(1-2\varphi)\,e_2-\varphi^3\,(2+\varphi)\,e_3}{\sqrt{10}\,\varphi^2} \\ 4 & \frac{e_1}{\sqrt{2}}+\frac{\left(2-\frac{1}{\varphi^2}\right)e_2}{\sqrt{10}}+\frac{\left(-2+\varphi^2\right)e_3}{\sqrt{10}} \\ 5 & \sqrt{\frac{2}{5}}\,(-e_2+e_3) \end{pmatrix} \quad \text{Math} = \begin{pmatrix} 1 & -\frac{1}{\sqrt{2}}+\frac{\sqrt{\frac{2}{5}}\left(e_2+\frac{1}{4}\left(-1+\sqrt{5}\right)^2 e_3\right)}{-1+\sqrt{5}} \\ 2 & \frac{1}{\sqrt{2}}+\frac{\sqrt{\frac{2}{5}}\left(e_2+\frac{1}{4}\left(-1+\sqrt{5}\right)^2 e_3\right)}{-1+\sqrt{5}} \\ 3 & -\frac{e_1}{\sqrt{2}}+\frac{2\sqrt{\frac{2}{5}}\left(\left(2-\sqrt{5}\right)e_2-\frac{1}{8}\left(-1+\sqrt{5}\right)^3\left(2+\frac{1}{2}\left(-1+\sqrt{5}\right)\right)e_3\right)}{\left(-1+\sqrt{5}\right)^2} \\ 4 & \frac{e_1}{\sqrt{2}}+\frac{\left(2-\frac{4}{\left(-1+\sqrt{5}\right)^2}\right)e_2}{\sqrt{10}}+\frac{\left(-2+\frac{1}{4}\left(-1+\sqrt{5}\right)^2\right)e_3}{\sqrt{10}} \\ 5 & \sqrt{\frac{2}{5}}\,(-e_2+e_3) \end{pmatrix} \quad \text{Numeric} = \begin{pmatrix} 1 & -0.7071+0.5117\,e_2+0.1954\,e_3 \\ 2 & 0.7071+0.5117\,e_2+0.1954\,e_3 \\ 3 & -0.7071\,e_1-0.1954\,e_2-0.5117\,e_3 \\ 4 & 0.7071\,e_1-0.1954\,e_2-0.5117\,e_3 \\ 5 & -0.6325\,e_2+0.6325\,e_3 \end{pmatrix}\text{List}$$

Norm'd distances:

From origin symbolic= $\left\{\frac{\sqrt{5+\frac{1}{\varphi^2}+\varphi^2}}{\sqrt{10}},\ \frac{\sqrt{5+\frac{1}{\varphi^2}+\varphi^2}}{\sqrt{10}},\ \frac{\sqrt{5+\frac{(1-2\varphi)^2}{\varphi^4}+\varphi^2\,(2+\varphi)^2}}{\sqrt{10}},\ \frac{\sqrt{13+\frac{1}{\varphi^4}-\frac{4}{\varphi^2}-4\,\varphi^2+\varphi^4}}{\sqrt{10}},\ \frac{2}{\sqrt{5}}\right\}$

From origin math = $\left\{\frac{2}{\sqrt{5}},\ \frac{2}{\sqrt{5}},\ \frac{2}{\sqrt{5}},\ \frac{2}{\sqrt{5}},\ \frac{2}{\sqrt{5}}\right\}$

From origin numeric = {0.894427, 0.894427, 0.894427, 0.894427, 0.894427}

From adjacent vertices symbolic= $\left\{\sqrt{2},\ \frac{\sqrt{1+\varphi\left(-6+9\,\varphi+\varphi^3\left(10+\varphi^2\,(3+\varphi)^2\right)\right)}}{\sqrt{10}\,\varphi^2},\ \frac{\sqrt{2+2\,\varphi\,(-2+\varphi\,(-1+\varphi\,(2+\varphi\,(7+(-1+\varphi)\,\varphi\,(1+\varphi)\,(2+\varphi))))))}}{\sqrt{5}\,\varphi^2},\ \frac{\sqrt{37+\frac{1}{\varphi^4}-\frac{8}{\varphi^2}-8\,\varphi^2+\varphi^4}}{\sqrt{10}},\ \frac{\sqrt{13+\frac{1}{\varphi^2}+\frac{4}{\varphi}-4\,\varphi+\varphi^2}}{\sqrt{10}}\right\}$

From adjacent vertices math = $\left\{\sqrt{2},\ \sqrt{2},\ \sqrt{2},\ \sqrt{2},\ \sqrt{2}\right\}$

From adjacent vertices numeric = {1.41421, 1.41421, 1.41421, 1.41421, 1.41421}

In[ ]:=

```
(*  Λ processes 3D orbit specifications.
  It defaults to the irregular A3 Icosahedron, on A3, we take default perms as "Osign" *)
Λ[f_, group_(*)String|group_List**) : "A3", orbit_List : {1, 1, 1}, permType_String : "Rotate"] := f[doW@group /. aRule@orbit, permType];
```

In[ ]:=

```
(*  Λ0 defaults to the irregular A3 Icosahedron, on A4, we take default perms as "Osign" *)
ΛnD[Λin_, f_, grp_, orb_List : {1, 1, 1}, permType_String : "Rotate"] :=
  Module[{Λinput, ωΛin, vΛin, validΛin, function, group, validGroupName, groupL, orbit, output, octΛinput, octppωω},
   print["Λ=", Λin, " f=", f, " grp=", grp, " orb=", orb, " perm=", permType];

   (* If missing inputs, correct assignments based on patterns *)
   (* If Λin is defined correctly it will a Member of the two valid Λin types.
     Otherwise set it to Null because it does not need Λinput options in processing the permutation. *)
   ωΛin = MemberQ[aωΛ, Λin];
   vΛin = MemberQ[avΛ, Λin];
   validΛin = ωΛin || vΛin;

   (* If orbit got a 3D default due to lack of inputs, but Λinput indicates it is 4D, change the orbit to 4D. *)
   orbit = orb;
   If[orbit == {1, 1, 1} && validΛin, orbit = {1, 1, 1, 1}];
   print["orbit=", orbit];

   (* If f is a string, it got assigned what is supposed to be a group. *)
   group = If[StringQ@f, f, grp];
   (* If the group is a valid name, take its length from its rank, otherwise infer it from the orbit length. *)
   validGroupName = StringQ@group && (StringLength@group < 2 || StringLength@group > 3);
   groupL = If[validGroupName, ToExpression[StringTake[group, -1]], Length@orbit];
   (* If the group is not a valid name, give it a default based on groupL=Length@orbit *)
   group = If[! validGroupName, If[groupL == 4, "A4", "A3"]];
   print["group=", group, " groupL=", groupL];

   (* If it is a valid group, set up the group  *)
   If[validGroupName, doW@group];

   (* If f is not a valid function, default it to oShow. *)
   function = If[! (f === oPerms || f === oE8List || f === oCalc || f === oShow), oShow, f];
   print["function=", function];

   Λinput = Which[
     (* If Λin in a valid 4D group, just set Λinput=Λin. *)
     ωΛin, Λin,
     vΛin, Λin,
     (* If f is a string, it got assigned what is supposed to be a group, so that means Λin=Null.
```

```
      But if there is a valid group name with length 4 (e.g. A4), we may assign it to avΛ0 *)
  StringQ@f && Length@orbit == 4 ||
   (* or if Λin=Null and grp is a valid 4D string, set it to Λinput=avΛ0. *)
   Λin === Null && validGroupName && groupL == 4, avΛ[[1]],
  (* It isn't defined, so set it to Null. *)
  True, Null];
print["ωΛin=", ωΛin, " vΛin=", vΛin, " validΛin=", validΛin, " Λinput=", Λinput];

(* Generate the permutation pattern(s):
 For 4D vertex generation,
   oct2Quat@Λin is a 4D list of quaternion coefficients.
   It is converted from Λin (which is one of 5 (avΛ0...4) sums of quaternion (octonion) elements,
     but only after applying the aRule@orbit replacments for aa1...4

     pp[[;;4]] is a scaled list of the first 4 of 5 quaternions  p0...p3,
        but you can also use aωΛ0...3 and ωω[[;;4]] instead of pp

   If the Λinput is Null, it is either 3D or already processed into an nD list by ΛnD, so treat it as an nD list just to be permuted.
    Otherwise it must be a 4D group to be processed with Λinput. *)
output = If[Length@orbit == 3 || Λinput === Null,
  function[doW@group /. aRule@orbit, permType],
  (* For 4D, generate the oScaleListOut string that can also be used as input for a call to oScaleListDo. *)
  (* The comments switch between quaternion/octonion math to Real *)
  octΛinput = oct2List[Λinput /. aRule@orbit];
  octppωω = oct2List[Total@If[vΛin, pp[[ ;; 4]], (ωω /. ωωRule)] /. aRule@orbit];
  print["octΛinput=", octΛinput, " octppωω=", octppωω];
  oScaleListIn = {
    If[doOctΛ, (* Quaternion/Octonion *)
     (* If Λinput is in avΛ use pp, otherwise use ωω  *)
     {oct2List[octΛinput ∘ octppωω][[If[doPP3D, 2, 1] ;; 4]], permType},
     (* Real *)
     {oct2List[Inner[Times, oct2Quat@Λinput,
          (* If Λinput is in avΛ use pp, otherwise use ωω *)
          If[vΛin, pp[[ ;; 4]], (ωω /. ωωRule)[[ ;; 4]]],
          Plus] /. aRule@orbit][[If[doPP3D, 2, 1] ;; 4]], permType}]};
  (* Process it into vertex permutations that can be fed to hulls3D (after trimming to 3D) or hulls3DPerms up to 8D. *)
  (**)print["oScaleListIn=", oScaleListIn];
  (* If ΛnD is called from oScaleListDo, this oScaleListOut will be replaced after the Join in oScaleListDo. *)
  oScaleListOut = DeleteDuplicates[(*)octSym/@**)Λ[oE8List, Sequence @@ First@oScaleListIn][[All, If[doPP3D, 2, 1] ;; 4]]];
  (**)print["oScaleListOut=", oScaleListOut]; (**)
  (* Output the vertices if it doesn't need to be visualized. *)
  If[function =!= oShow, oScaleListOut,
   (* Generate the convex hulls of the 3 or 4D vertices *)
```

```
      TableForm@{Row@{" f=", function, " group=", group, " orbit=", orbit, " perm=", permType},
        If[doPP3D,
         hulls3D@oScaleListOut,
         hulls3DPerms["oScaleListOut", False, , 4]]}]];
    output];
```

These are functions involved in generating the Graphics3D visualization of nD orbit hulls using quaternion math for a given group (e.g. A3, B3, H3, A4, C4, D4, F4, H4)

In[ ]:=

```
(* Using perms (above) to generate solids with octonion (quaternion) orbits as input *)
oE8List[in_List, select_List : {}, permType_String : "Rotate"] := Module[{out},
   DeleteDuplicates[
    (* Use oScale to scale "in" without having to change the VisibLie_E8 UI.
       If the permType "NoneNoSign" indicates it is already a large list or the list is >8  or oScale is not a list just use "in",
        otherwise multiply "in" by oScale (usually an array of 1's. using the correct dimensions
       (e.g. if it a 3D list, take oScale dims 2;;4 otherwise 1;;Length@in *)
    out = perms[in
       If[permType == "NoneNoSign" || ! ListQ@oScale || Length@in > 8, 1,
        If[Length@in == 3, oScale⟦2 ;; 4⟧, oScale⟦ ;; Length@in⟧]], select , permType];
    (* Rejoin the permutations output to an 8D oct2List *)
    If[Length@# == 3,
       Join[{0}, #, {0, 0, 0, 0}],
       Join[#, Array[0 &, 8 - Length@# /. slRep]]] & /@ If[permType == "NoneNoSign" || Length@out⟦1⟧ > 8, Flatten[out, 1], out]]];
oPerms[in_List, select_List : {}, permType_String : "Rotate"] := octonion /@ oE8List[in, select, permType];
oCalc[in_List, select_List : {}, permType_String : "Rotate"] := oct2ImQuat@# & /@ oPerms[in, select, permType];
oShow[in_List, select_List : {}, permType_String : "Rotate"] := hulls3D[oCalc[in, select, permType]] (*) ⟦1,2,1⟧//ColumnForm**) // MatrixForm(**);
```

Define the SU5 4D Solids

In[ ]:=
```
doPP3D = False;
addNullPoints = True;
```

In[ ]:=
```
doW@"A4";
```

Define the 16 orbits & their duals for A4

Single Ring Orbits

In[ ]:=
```
(*) {1,0,0,0} **)
FiveCell = perms4D[[1]];
(*) Negation Dual **)
Dual5Cell = -TriRectified5Cell;
(*) -{0,0,0,1} **)
```

In[ ]:=
```
(*) {0,1,0,0} **)
Rectified5Cell = perms4D[[2]];
(*) Negation Dual **)
DualRectified5Cell = -BiRectified5Cell;
(*) -{0,0,1,0} **)
```

In[ ]:=
```
(*) {0,0,1,0} **)
BiRectified5Cell = perms4D[[3]];
(*) Negation Dual **)
DualBiRectified5Cell = -Rectified5Cell;
(*) -{0,0,1,0} **)
```

In[ ]:=
```
(*) {0,0,0,1} **)
TriRectified5Cell = perms4D[[4]];
(*) Negation Dual **)
DualTriRectified5Cell = -FiveCell;
(*)-{0,0,0,1} **)
```

Dual Ring Orbits

In[ ]:=
```
(* {1,1,0,0} *)
Truncated5Cell = perms4D[[5]];
(*) Negation Dual **)
DualTruncated5Cell = DualTruncated5Cell;
(*)-{0,0,1,1}=**)
```

In[ ]:=
```
(* {1,0,1,0} *)
Cantellated5Cell = perms4D[[6]];
(*) Negation Dual **)
DualCantellated5Cell = -DualCantellated5Cell;
(*)-{1,0,1,0} **)
```

In[ ]:=
```
(*) {1,0,0,1} **)
Runcinated5Cell = perms4D[[7]];
(*) Self Dual **)
DualRuncinated5Cell = -Runcinated5Cell;
```

In[ ]:=
```
(*) {0,1,1,0} **)
BiTruncated5Cell = perms4D[[8]];
(*) Self Dual **)
DualBiTruncated5Cell = -BiTruncated5Cell;
```

In[ ]:=
```
(*) {0,1,0,1} **)
BiCantellated5Cell = perms4D[[9]];
(*) Negation Dual **)
DualBiCantellated5Cell = Cantellated5Cell;
(*)-{0,1,0,1} **)
```

In[ ]:=
```
(*) {0,0,1,1} **)
TriTruncated5Cell = perms4D[[10]];
(*) Negation Dual **)
DualTriTruncated5Cell = -Truncated5Cell;
(*)-{0,0,1,1} **)
```

Triple Ring Orbits

In[ ]:=
```
(*) {1,1,1,0} **)
CantiTruncated5Cell = perms4D[[11]];
(*) Negation Dual **)
DualCantiTruncated5Cell = -CantiRuncinated5Cell;
(*)-{0,1,1,1} **)
```

In[ ]:=
```
(*) {1,1,0,1} **)
RunciTruncated5Cell = perms4D[[12]];
(*) Negation Dual **)
DualRunciTruncated5Cell = -BiCantiTruncated5Cell;
(*)-{1,0,1,1} **)
```

In[ ]:=
```
(*) {1,0,1,1} **)
BiCantiTruncated5Cell = perms4D[[13]];
(*) Negation Dual **)
DualBiCantiTruncated5Cell = -RunciTruncated5Cell;
(*)-{1,1,0,1} **)
```

In[ ]:=
```
(*) Not listed {0,1,1,1} **)
CantiRuncinated5Cell = perms4D[[14]];
(*) Negation Dual **)
DualCantiRuncinated5Cell = -CantiTruncated5Cell;
(*)-{1,1,1,0}  **)
```

Quad Ring Orbit

In[ ]:=
```
(*) {1,1,1,1} **)
OmniTruncated5Cell = perms4D[[15]];
(*) Self Dual **)
DualOmniTruncated5Cell = -OmniTruncated5Cell;
```

Pre-evaluate multiOrbit 4D Solids

In[ ]:=
```
multiOrbitList4D = {{},
   FiveCell, Rectified5Cell, BiRectified5Cell, TriRectified5Cell,
   Truncated5Cell, Cantellated5Cell, Runcinated5Cell, BiTruncated5Cell, BiCantellated5Cell, TriTruncated5Cell,
   CantiTruncated5Cell, RunciTruncated5Cell, BiCantiTruncated5Cell, CantiRuncinated5Cell, OmniTruncated5Cell,

   Dual5Cell, DualRectified5Cell, DualBiRectified5Cell, DualTriRectified5Cell, DualTruncated5Cell,
   DualCantellated5Cell, DualRuncinated5Cell, DualBiTruncated5Cell, DualBiCantellated5Cell, DualTriTruncated5Cell,
   DualCantiTruncated5Cell, DualRunciTruncated5Cell, DualBiCantiTruncated5Cell, DualCantiRuncinated5Cell, DualOmniTruncated5Cell};
```

In[ ]:=

```
multiOrbitList4DStr = {"None", "5-Cell", "Rectified 5-Cell", "BiRectified 5-Cell", "TriRectified 5-Cell", "Truncated 5-Cell",
    "Cantellated 5-Cell", "Runcinated 5-Cell", "BiTruncated 5-Cell", "BiCantellated 5-Cell", "TriTruncated 5-Cell",
    "CantiTruncated 5-Cell", "RunciTruncated 5-Cell", "BiCantiTruncated 5-Cell", "CantiRuncinated 5-Cell", "OmniTruncated 5-Cell",

    "Dual 5-Cell", "Dual Rectified 5-Cell", "Dual BiRectified 5-Cell",
    "Dual TriRectified 5-Cell", "Dual Truncated 5-Cell", "Dual Cantellated 5-Cell", "Dual Runcinated 5-Cell",
    "Dual BiTruncated 5-Cell", "Dual BiCantellated 5-Cell", "Dual TriTruncated 5-Cell", "Dual CantiTruncated 5-Cell",
    "Dual RunciTruncated 5-Cell", "Dual BiCantiTruncated 5-Cell", "Dual CantiRuncinated 5-Cell", "Dual OmniTruncated 5-Cell"};

(**)
If[MemberQ[paneList, "Dynkin"],
  (*)doPrint=True;**)
  multiOrbitList4DHulls  = Join[{""}, (Print["Starting ", #];
       oScaleListOut = DeleteDuplicates@∧nD[av∧⟦1⟧, oCalc, "A4", multiOrbitList4D⟦#⟧, "Osign"];
       hulls3DPerms["oScaleListOut", False, , (**)1(*)4**)];
       Row@{combinedHull, overallHull}) & /@ Range[2, Length@multiOrbitList4D]];
  doPrint = False];
```

showGroups produces this $SU_5 \supset SU_4 \otimes U_1$ verts ↦ Graphics3D list

In[ ]:=
```
getSU4Hull[perm4DNum_, ΛNum_] := (ΛnD[avΛ⟦ΛNum⟧, oShow, "A4", perms4D⟦perm4DNum⟧, "Osign"]; overallHull);
getSU5Hull@orbitNum_ := multiOrbitList4DHulls⟦1 + orbitNum⟧;
```

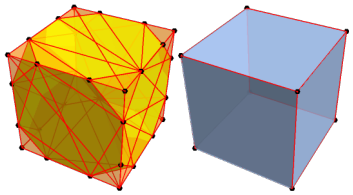
In[ ]:=
```
(* This uses the RepName function from GroupMath *)
showGroups[in_, mult_ : 1] := Module[{(**)allSU4Orbits, uniqueSU4Orbits, SU5sum, SU4sum, SU5net, u1List, orbit, u1(**)}, {
      SU5sum = getRepNumber[RepName[SU5, mult perms4D⟦#⟧]⟦1⟧];
      allSU4Orbits = Table[Abs /@ in⟦i, ;; 3⟧ /. aRule[mult perms4D⟦#⟧], {i, 5}];
      uniqueSU4Orbits = Union@allSU4Orbits;
      SU4sum = Total[getRepNumber[RepName[SU4, #]⟦1⟧] & /@ uniqueSU4Orbits];
      SU5net = SU5sum - SU4sum;
      u1List = Table[Round[Total[If[i < 5, -1, 1] in⟦i, 4⟧ /. aRule[mult perms4D⟦#⟧]]], {i, 5}];
      Row@{"SU5 O(", Style[Row[mult perms4D⟦#⟧], Orange], ")          SU5 ⊃ SU4 ⊗ U1 ", getSU5Hull@#},
      Row@{"SU5-ΣSU4=", Style[SU5sum, Orange], "-", Style[SU4sum, Blue], "=", Style[SU5net, Black], " missing"},
      Row@{"ΣU1 ", Style[u1List, Red], "=", Total@u1List},
      ColumnForm@Table[
        orbit = Abs /@ in⟦i, ;; 3⟧ /. aRule[mult perms4D⟦#⟧];
        u1 = Round[Total[If[i < 5, -1, 1] in⟦i, 4⟧ /. aRule[mult perms4D⟦#⟧]]];
        repName = RepName[{SU5, SU4, U1}, {mult perms4D⟦#⟧, orbit, u1}];
        Row@{"Λ'(", i - 1, ")  SU4 O(", Style[Row@orbit, Blue], ")  ",
          Style[repName⟦1⟧, Bold, Orange], "⊃", Style[repName⟦2⟧, Bold, Blue], "⊗",
          Style[repName⟦3⟧, Bold, If[ToExpression@Last@repName == u1, Red, Black]],
          " verts ↦", getSU4Hull[#, i]}, {i, 5}]} & /@ Range@Length@perms4D];
```

In[2766]:=

```
doPP3D = True;
MatrixForm@Transpose[showGroups[avΛOrbit, #] & /@ Range@2]
```

Out[2767]//MatrixForm=

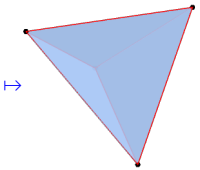
SU_5 O(1000) $SU_5 \supset SU_4 \otimes U_1$



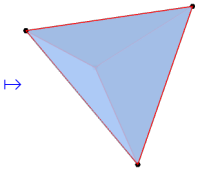
$SU_5 - \Sigma SU_4 = 5 - 5 = 0$ missing

$\Sigma U1$ {-1, -1, -1, -1, 4}=0

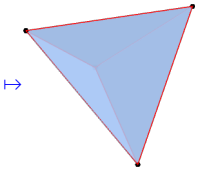
Λ'(0) SU_4 O(100) 5⊃4⊗-1 verts ↦



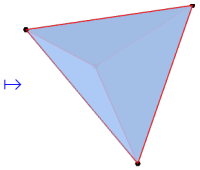
Λ'(1) SU_4 O(100) 5⊃4⊗-1 verts ↦



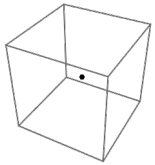
Λ'(2) SU_4 O(100) 5⊃4⊗-1 verts ↦



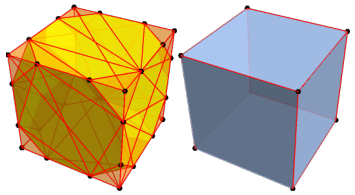
Λ'(3) SU_4 O(100) 5⊃4⊗-1 verts ↦



Λ'(4) SU_4 O(000) 5⊃1⊗4 verts ↦



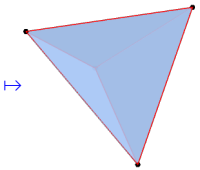
SU_5 O(2000) $SU_5 \supset SU_4 \otimes U_1$



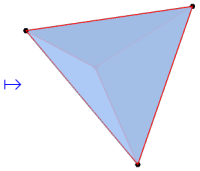
$SU_5 - \Sigma SU_4 = 15 - 11 = 4$ missing

$\Sigma U1$ {-2, -2, -2, -2, 8}=0

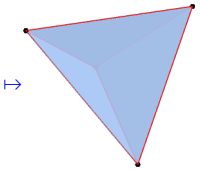
Λ'(0) SU_4 O(200) 15⊃10⊗-2 verts ↦



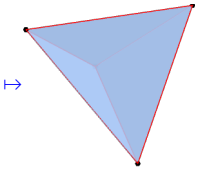
Λ'(1) SU_4 O(200) 15⊃10⊗-2 verts ↦



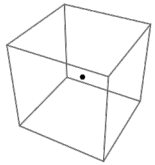
Λ'(2) SU_4 O(200) 15⊃10⊗-2 verts ↦



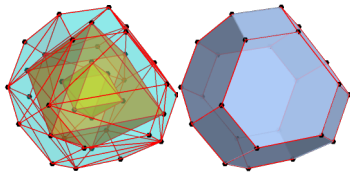
Λ'(3) SU_4 O(200) 15⊃10⊗-2 verts ↦



Λ'(4) SU_4 O(000) 15⊃1⊗8 verts ↦



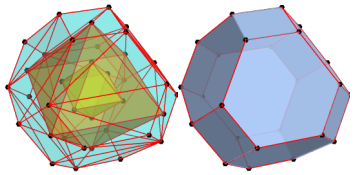
SU_5 O(0100) $SU_5 \supset SU_4 \otimes U_1$



$SU_5 - \Sigma SU_4 = 10 - 10 = 0$ missing

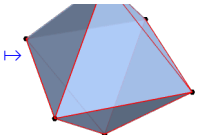
$\Sigma U1$ {-2, -2, -2, 3, 3}=0

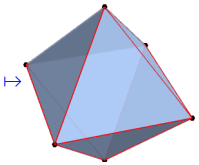
SU_5 O(0200) $SU_5 \supset SU_4 \otimes U_1$

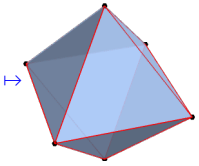


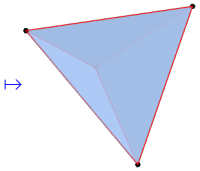
$SU_5 - \Sigma SU_4 = 50 - 30 = 20$ missing

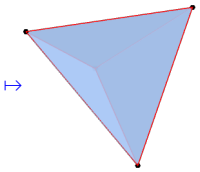
$\Sigma U1$ {-4, -4, -4, 6, 6}=0

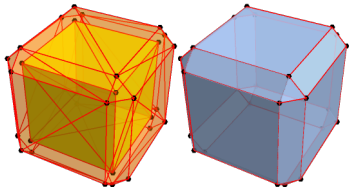
Λ'(0) SU_4 O(010) 10⊃6⊗-2 verts ↦ 

Λ'(1) SU_4 O(010) 10⊃6⊗-2 verts ↦ 

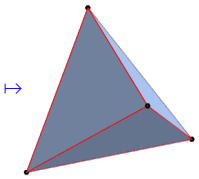
Λ'(2) SU_4 O(010) 10⊃6⊗-2 verts ↦ 

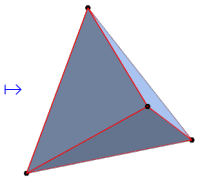
Λ'(3) SU_4 O(100) 10⊃4⊗3 verts ↦ 

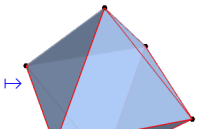
Λ'(4) SU_4 O(100) 10⊃4⊗3 verts ↦ 

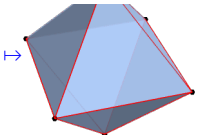
SU_5 O(0010) SU_5 ⊃ SU_4 ⊗ U_1 

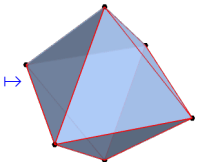
$SU_5-\Sigma SU_4=10-10=0$ missing
$\Sigma U1$ {-3, -3, 2, 2, 2}=0

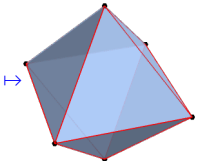
Λ'(0) SU_4 O(001) $\overline{10}$⊃$\overline{4}$⊗-3 verts ↦ 

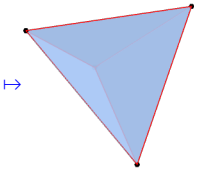
Λ'(1) SU_4 O(001) $\overline{10}$⊃$\overline{4}$⊗-3 verts ↦ 

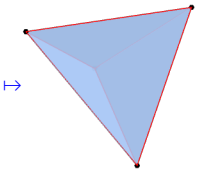
Λ'(2) SU_4 O(010) $\overline{10}$⊃6⊗2 verts ↦ 

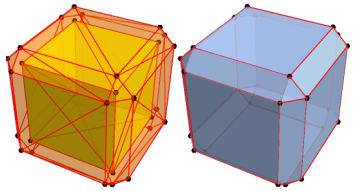
Λ'(0) SU_4 O(020) $\overline{50}$⊃20'⊗-4 verts ↦ 

Λ'(1) SU_4 O(020) $\overline{50}$⊃20'⊗-4 verts ↦ 

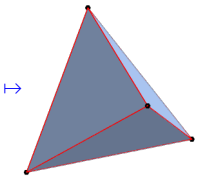
Λ'(2) SU_4 O(020) $\overline{50}$⊃20'⊗-4 verts ↦ 

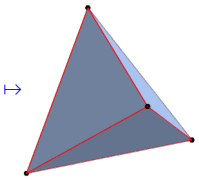
Λ'(3) SU_4 O(200) $\overline{50}$⊃10⊗6 verts ↦ 

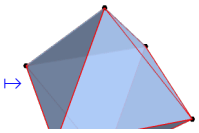
Λ'(4) SU_4 O(200) $\overline{50}$⊃10⊗6 verts ↦ 

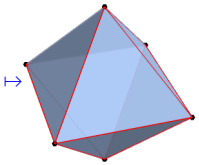
SU_5 O(0020) SU_5 ⊃ SU_4 ⊗ U_1 

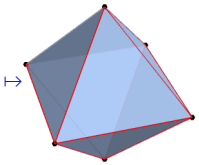
$SU_5-\Sigma SU_4=50-30=20$ missing
$\Sigma U1$ {-6, -6, 4, 4, 4}=0

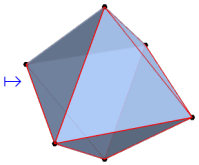
Λ'(0) SU_4 O(002) 50⊃$\overline{10}$⊗-6 verts ↦ 

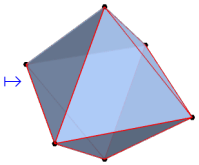
Λ'(1) SU_4 O(002) 50⊃$\overline{10}$⊗-6 verts ↦ 

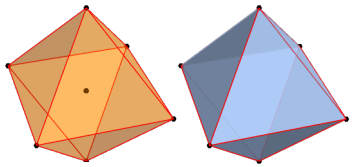
Λ'(2) SU_4 O(020) 50⊃20'⊗4 verts ↦ 

Λ'(3) SU_4 O(010) $\overline{10}\supset 6\otimes 2$ verts ↦ 

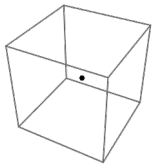
Λ'(4) SU_4 O(010) $\overline{10}\supset 6\otimes 2$ verts ↦ 

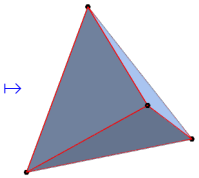
Λ'(3) SU_4 O(020) $50\supset 20'\otimes 4$ verts ↦ 

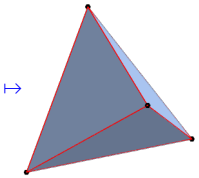
Λ'(4) SU_4 O(020) $50\supset 20'\otimes 4$ verts ↦ 

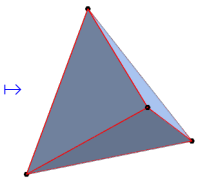
SU_5 O(0001) $SU_5 \supset SU_4 \otimes U_1$ 

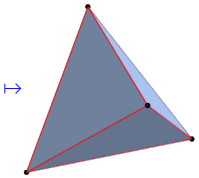
$SU_5-\Sigma SU_4=5-5=0$ missing
$\Sigma U1$ {-4, 1, 1, 1, 1}=0

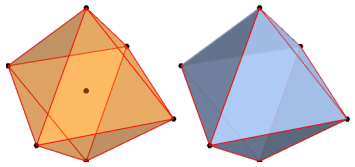
Λ'(0) SU_4 O(000) $\overline{5}\supset 1\otimes -4$ verts ↦ 

Λ'(1) SU_4 O(001) $\overline{5}\supset \overline{4}\otimes 1$ verts ↦ 

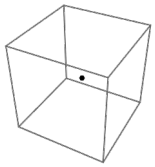
Λ'(2) SU_4 O(001) $\overline{5}\supset \overline{4}\otimes 1$ verts ↦ 

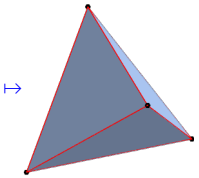
Λ'(3) SU_4 O(001) $\overline{5}\supset \overline{4}\otimes 1$ verts ↦ 

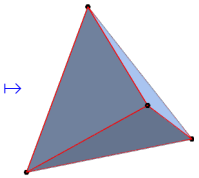
Λ'(4) SU_4 O(001) $\overline{5}\supset \overline{4}\otimes 1$ verts ↦ 

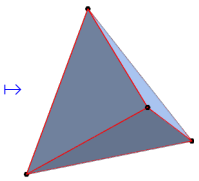
SU_5 O(0002) $SU_5 \supset SU_4 \otimes U_1$ 

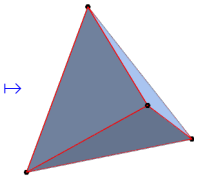
$SU_5-\Sigma SU_4=15-11=4$ missing
$\Sigma U1$ {-8, 2, 2, 2, 2}=0

Λ'(0) SU_4 O(000) $\overline{15}\supset 1\otimes -8$ verts ↦ 

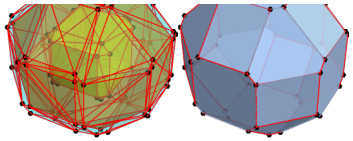
Λ'(1) SU_4 O(002) $\overline{15}\supset \overline{10}\otimes 2$ verts ↦ 

Λ'(2) SU_4 O(002) $\overline{15}\supset \overline{10}\otimes 2$ verts ↦ 

Λ'(3) SU_4 O(002) $\overline{15}\supset \overline{10}\otimes 2$ verts ↦ 

Λ'(4) SU_4 O(002) $\overline{15}\supset \overline{10}\otimes 2$ verts ↦ 

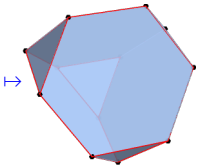
SU_5 O(1100) $SU_5 \supset SU_4 \otimes U_1$



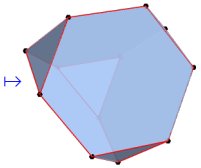
$SU_5 - \Sigma SU_4 = 40 - 34 = 6$ missing

$\Sigma U1$ {-3, -3, -3, 2, 7}=0

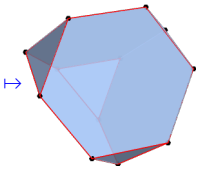
$\Lambda'(0)$ SU_4 O(110) $\overline{40} \supset \overline{20} \otimes -3$ verts ↦



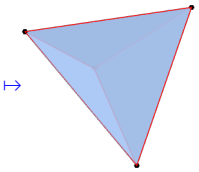
$\Lambda'(1)$ SU_4 O(110) $\overline{40} \supset \overline{20} \otimes -3$ verts ↦



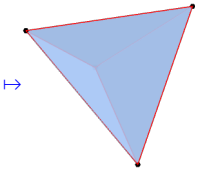
$\Lambda'(2)$ SU_4 O(110) $\overline{40} \supset \overline{20} \otimes -3$ verts ↦



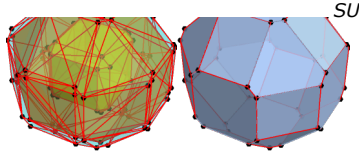
$\Lambda'(3)$ SU_4 O(200) $\overline{40} \supset 10 \otimes 2$ verts ↦



$\Lambda'(4)$ SU_4 O(100) $\overline{40} \supset 4 \otimes 7$ verts ↦



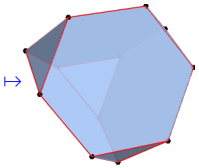
SU_5 O(2200) $SU_5 \supset SU_4 \otimes U_1$



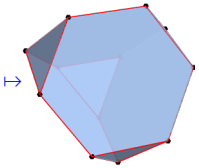
$SU_5 - \Sigma SU_4 = 420 - 171 = 249$ missing

$\Sigma U1$ {-6, -6, -6, 4, 14}=0

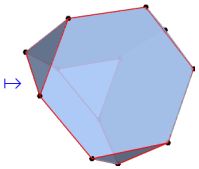
$\Lambda'(0)$ SU_4 O(220) $420 \supset 126 \otimes -6$ verts ↦



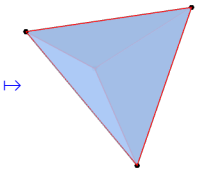
$\Lambda'(1)$ SU_4 O(220) $420 \supset 126 \otimes -6$ verts ↦



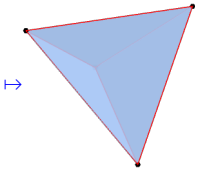
$\Lambda'(2)$ SU_4 O(220) $420 \supset 126 \otimes -6$ verts ↦



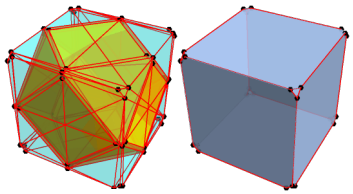
$\Lambda'(3)$ SU_4 O(400) $420 \supset 35 \otimes 4$ verts ↦



$\Lambda'(4)$ SU_4 O(200) $420 \supset 10 \otimes 14$ verts ↦



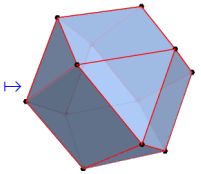
SU_5 O(1010) $SU_5 \supset SU_4 \otimes U_1$

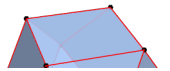


$SU_5 - \Sigma SU_4 = 45 - 41 = 4$ missing

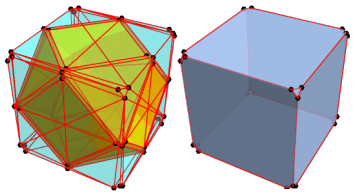
$\Sigma U1$ {-4, -4, 1, 1, 6}=0

$\Lambda'(0)$ SU_4 O(101) $\overline{45} \supset 15 \otimes -4$ verts ↦





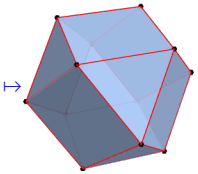
SU_5 O(2020) $SU_5 \supset SU_4 \otimes U_1$

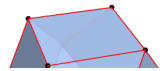


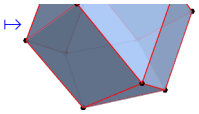
$SU_5 - \Sigma SU_4 = 560 - 230 = 330$ missing

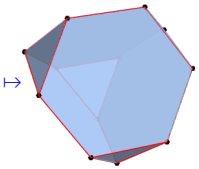
$\Sigma U1$ {-8, -8, 2, 2, 12}=0

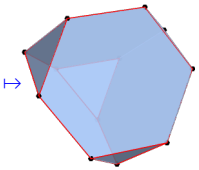
$\Lambda'(0)$ SU_4 O(202) $\overline{560} \supset 84 \otimes -8$ verts ↦

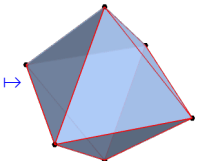


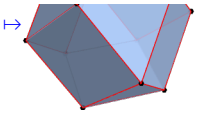


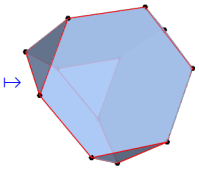
Λ'(1) SU_4 O(101) $\overline{45}\supset 15\otimes -4$ verts ↦ 

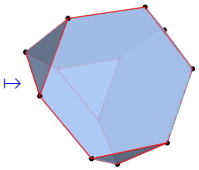
Λ'(2) SU_4 O(110) $\overline{45}\supset \overline{20}\otimes 1$ verts ↦ 

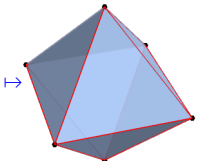
Λ'(3) SU_4 O(110) $\overline{45}\supset \overline{20}\otimes 1$ verts ↦ 

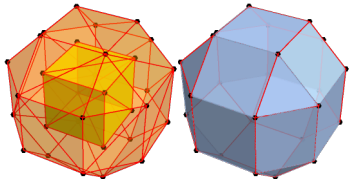
Λ'(4) SU_4 O(010) $\overline{45}\supset 6\otimes 6$ verts ↦ 

Λ'(1) SU_4 O(202) $\overline{560}\supset 84\otimes -8$ verts ↦ 

Λ'(2) SU_4 O(220) $\overline{560}\supset 126\otimes 2$ verts ↦ 

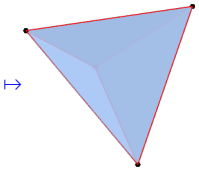
Λ'(3) SU_4 O(220) $\overline{560}\supset 126\otimes 2$ verts ↦ 

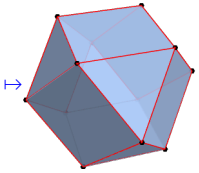
Λ'(4) SU_4 O(020) $\overline{560}\supset 20'\otimes 12$ verts ↦ 

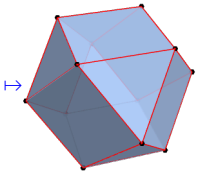
SU_5 O(1001) $SU_5 \supset SU_4 \otimes U_1$ 

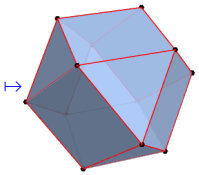
$SU_5-\Sigma SU_4=24-23=1$ missing

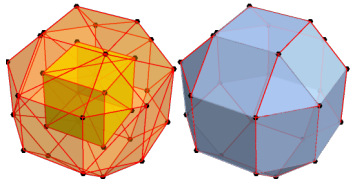
$\Sigma U1$ $\{-5, 0, 0, 0, 5\}=0$

Λ'(0) SU_4 O(100) $24\supset 4\otimes -5$ verts ↦ 

Λ'(1) SU_4 O(101) $24\supset 15\otimes 0$ verts ↦ 

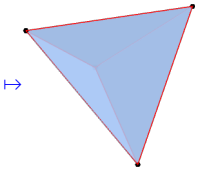
Λ'(2) SU_4 O(101) $24\supset 15\otimes 0$ verts ↦ 

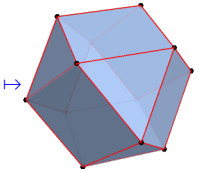
Λ'(3) SU_4 O(101) $24\supset 15\otimes 0$ verts ↦ 

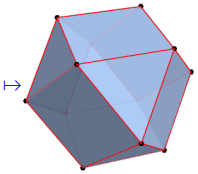
SU_5 O(2002) $SU_5 \supset SU_4 \otimes U_1$ 

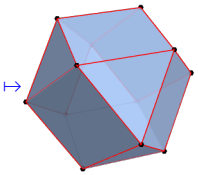
$SU_5-\Sigma SU_4=200-104=96$ missing

$\Sigma U1$ $\{-10, 0, 0, 0, 10\}=0$

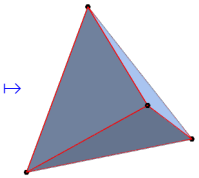
Λ'(0) SU_4 O(200) $200\supset 10\otimes -10$ verts ↦ 

Λ'(1) SU_4 O(202) $200\supset 84\otimes 0$ verts ↦ 

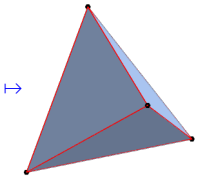
Λ'(2) SU_4 O(202) $200\supset 84\otimes 0$ verts ↦ 

Λ'(3) SU_4 O(202) $200\supset 84\otimes 0$ verts ↦ 

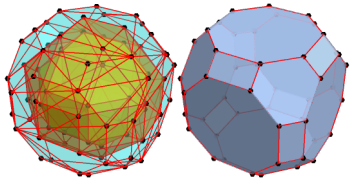
Λ'(4) SU_4 O(001) 24⊃$\overline{4}$⊗5 verts ↦



Λ'(4) SU_4 O(002) 200⊃$\overline{10}$⊗10 verts ↦

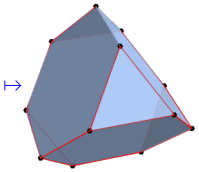


SU_5 O(0110) SU_5 ⊃ SU_4 ⊗ U_1

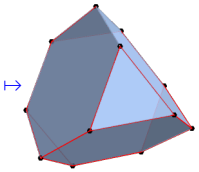


SU_5−ΣSU_4=75−60=15 missing
ΣU1 {−5, −5, 0, 5, 5}=0

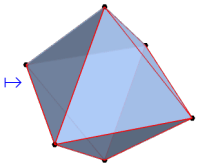
Λ'(0) SU_4 O(011) 75⊃20⊗−5 verts ↦



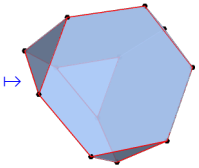
Λ'(1) SU_4 O(011) 75⊃20⊗−5 verts ↦



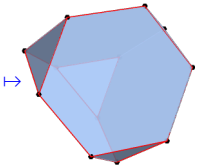
Λ'(2) SU_4 O(020) 75⊃20'⊗0 verts ↦



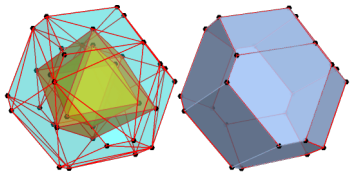
Λ'(3) SU_4 O(110) 75⊃$\overline{20}$⊗5 verts ↦



Λ'(4) SU_4 O(110) 75⊃$\overline{20}$⊗5 verts ↦

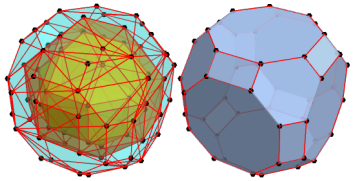


SU_5 O(0101) SU_5 ⊃ SU_4 ⊗ U_1



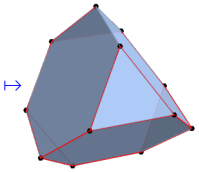
SU_5−ΣSU_4=45−41=4 missing
ΣU1 {−6, −1, −1, 4, 4}=0

SU_5 O(0220) SU_5 ⊃ SU_4 ⊗ U_1

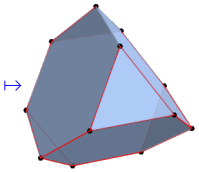


SU_5−ΣSU_4=1176−357=819 missing
ΣU1 {−10, −10, 0, 10, 10}=0

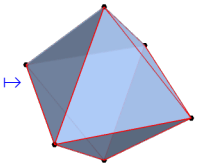
Λ'(0) SU_4 O(022) 1176⊃$\overline{126}$⊗−10 verts ↦



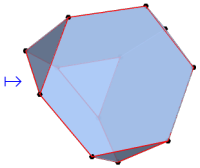
Λ'(1) SU_4 O(022) 1176⊃$\overline{126}$⊗−10 verts ↦



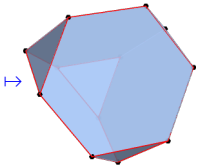
Λ'(2) SU_4 O(040) 1176⊃105⊗0 verts ↦



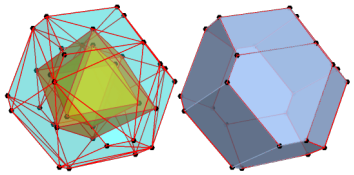
Λ'(3) SU_4 O(220) 1176⊃126⊗10 verts ↦



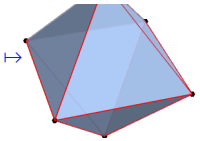
Λ'(4) SU_4 O(220) 1176⊃126⊗10 verts ↦

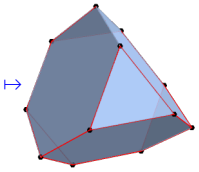


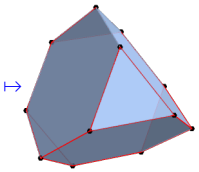
SU_5 O(0202) SU_5 ⊃ SU_4 ⊗ U_1

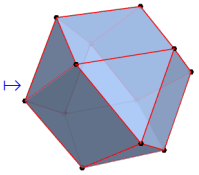


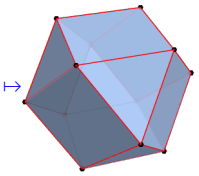
SU_5−ΣSU_4=560−230=330 missing
ΣU1 {−12, −2, −2, 8, 8}=0

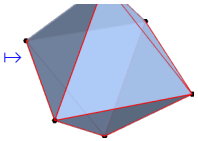
Λ'(0) SU_4 O(010) 45⊃6⊗-6 verts ↦ 

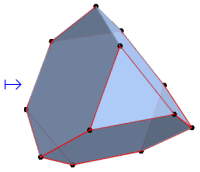
Λ'(1) SU_4 O(011) 45⊃20⊗-1 verts ↦ 

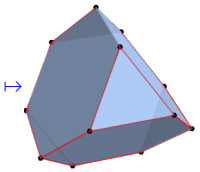
Λ'(2) SU_4 O(011) 45⊃20⊗-1 verts ↦ 

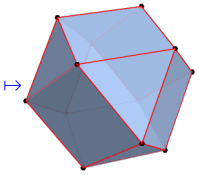
Λ'(3) SU_4 O(101) 45⊃15⊗4 verts ↦ 

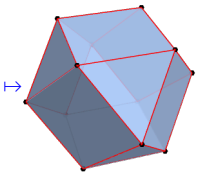
Λ'(4) SU_4 O(101) 45⊃15⊗4 verts ↦ 

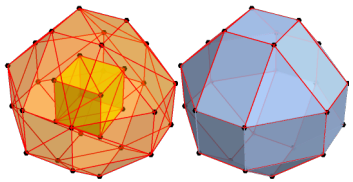
Λ'(0) SU_4 O(020) 560⊃20'⊗-12 verts ↦ 

Λ'(1) SU_4 O(022) 560⊃$\overline{126}$⊗-2 verts ↦ 

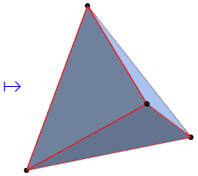
Λ'(2) SU_4 O(022) 560⊃$\overline{126}$⊗-2 verts ↦ 

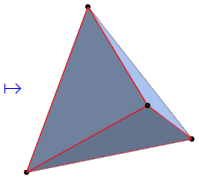
Λ'(3) SU_4 O(202) 560⊃84⊗8 verts ↦ 

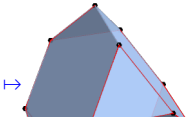
Λ'(4) SU_4 O(202) 560⊃84⊗8 verts ↦ 

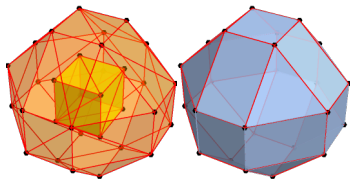
SU_5 O(0011) SU_5 ⊃ SU_4 ⊗ U_1 

SU_5-ΣSU_4=40-34=6 missing
ΣU1 {-7, -2, 3, 3, 3}=0

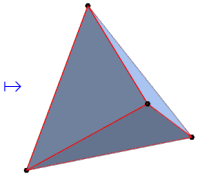
Λ'(0) SU_4 O(001) 40⊃$\overline{4}$⊗-7 verts ↦ 

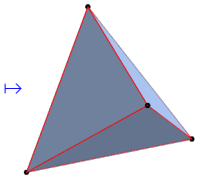
Λ'(1) SU_4 O(002) 40⊃$\overline{10}$⊗-2 verts ↦ 

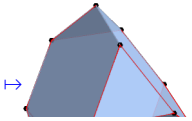
Λ'(2) SU_4 O(011) 40⊃20⊗3 verts ↦ 

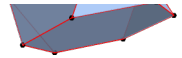
SU_5 O(0022) SU_5 ⊃ SU_4 ⊗ U_1 

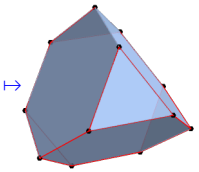
SU_5-ΣSU_4=420-171=249 missing
ΣU1 {-14, -4, 6, 6, 6}=0

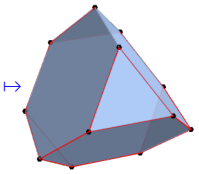
Λ'(0) SU_4 O(002) $\overline{420}$⊃$\overline{10}$⊗-14 verts ↦ 

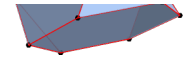
Λ'(1) SU_4 O(004) $\overline{420}$⊃$\overline{35}$⊗-4 verts ↦ 

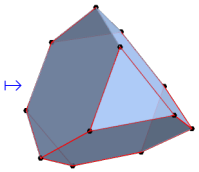
Λ'(2) SU_4 O(022) $\overline{420}$⊃$\overline{126}$⊗6 verts ↦ 

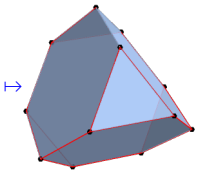


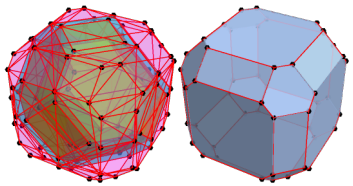
Λ'(3) SU_4 O(011) 40⊃20⊗3 verts ↦ 

Λ'(4) SU_4 O(011) 40⊃20⊗3 verts ↦ 

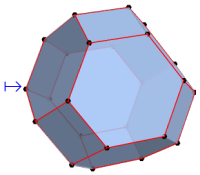


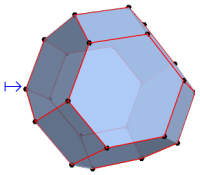
Λ'(3) SU_4 O(022) $\overline{420}$⊃$\overline{126}$⊗6 verts ↦ 

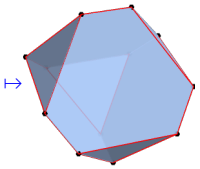
Λ'(4) SU_4 O(022) $\overline{420}$⊃$\overline{126}$⊗6 verts ↦ 

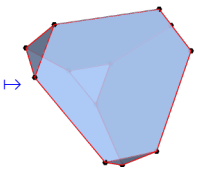
SU_5 O(1110) SU_5 ⊃ SU_4 ⊗ U_1 

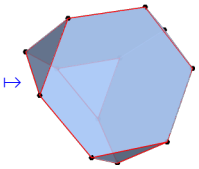
$SU_5-\Sigma SU_4=280-189=91$ missing
$\Sigma U1$ {-6, -6, -1, 4, 9}=0

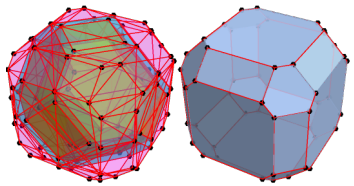
Λ'(0) SU_4 O(111) 280⊃64⊗-6 verts ↦ 

Λ'(1) SU_4 O(111) 280⊃64⊗-6 verts ↦ 

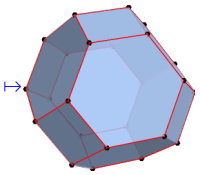
Λ'(2) SU_4 O(120) 280⊃60⊗-1 verts ↦ 

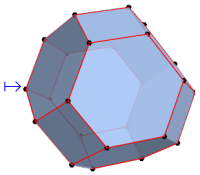
Λ'(3) SU_4 O(210) 280⊃45⊗4 verts ↦ 

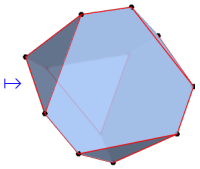
Λ'(4) SU_4 O(110) 280⊃$\overline{20}$⊗9 verts ↦ 

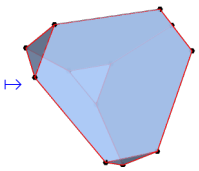
SU_5 O(2220) SU_5 ⊃ SU_4 ⊗ U_1 

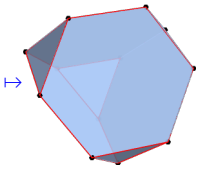
$SU_5-\Sigma SU_4=8505-1755=6750$ missing
$\Sigma U1$ {-12, -12, -2, 8, 18}=0

Λ'(0) SU_4 O(222) 8505⊃729⊗-12 verts ↦ 

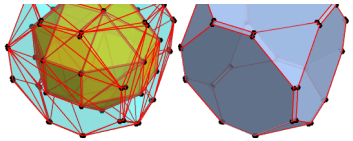
Λ'(1) SU_4 O(222) 8505⊃729⊗-12 verts ↦ 

Λ'(2) SU_4 O(240) 8505⊃540'⊗-2 verts ↦ 

Λ'(3) SU_4 O(420) 8505⊃360'⊗8 verts ↦ 

Λ'(4) SU_4 O(220) 8505⊃126⊗18 verts ↦ 

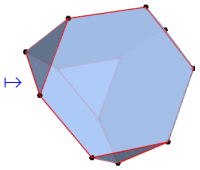
SU_5 O(1101) $SU_5 \supset SU_4 \otimes U_1$



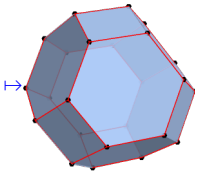
$SU_5 - \Sigma SU_4 = 175 - 135 = 40$ missing

$\Sigma U1$ {-7, -2, -2, 3, 8}=0

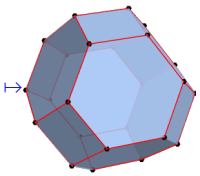
Λ'(0) SU_4 O(110) 175⊃$\overline{20}$⊗-7 verts ↦



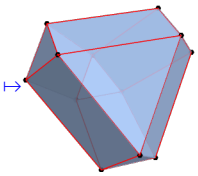
Λ'(1) SU_4 O(111) 175⊃64⊗-2 verts ↦



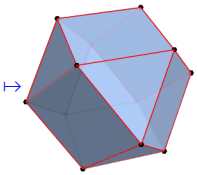
Λ'(2) SU_4 O(111) 175⊃64⊗-2 verts ↦



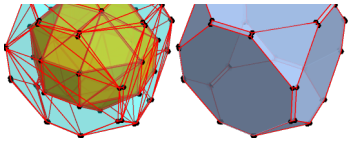
Λ'(3) SU_4 O(201) 175⊃36⊗3 verts ↦



Λ'(4) SU_4 O(101) 175⊃15⊗8 verts ↦



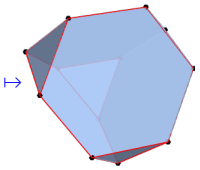
SU_5 O(2202) $SU_5 \supset SU_4 \otimes U_1$



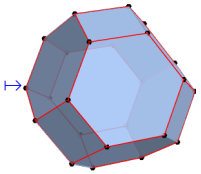
$SU_5 - \Sigma SU_4 = 4410 - 1209 = 3201$ missing

$\Sigma U1$ {-14, -4, -4, 6, 16}=0

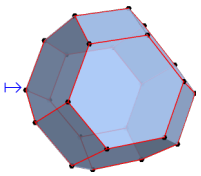
Λ'(0) SU_4 O(220) $\overline{4410}$⊃126⊗-14 verts ↦



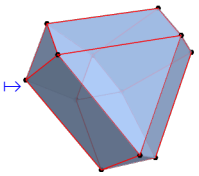
Λ'(1) SU_4 O(222) $\overline{4410}$⊃729⊗-4 verts ↦



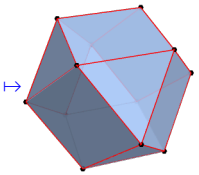
Λ'(2) SU_4 O(222) $\overline{4410}$⊃729⊗-4 verts ↦



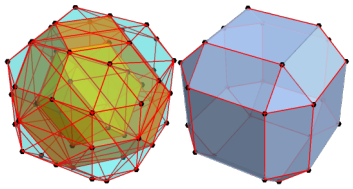
Λ'(3) SU_4 O(402) $\overline{4410}$⊃270⊗6 verts ↦



Λ'(4) SU_4 O(202) $\overline{4410}$⊃84⊗16 verts ↦



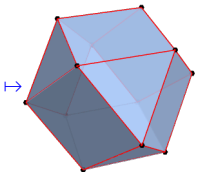
SU_5 O(1011) $SU_5 \supset SU_4 \otimes U_1$



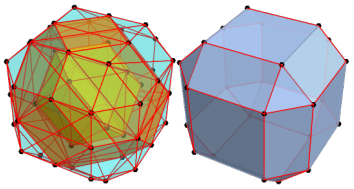
$SU_5 - \Sigma SU_4 = 175 - 135 = 40$ missing

$\Sigma U1$ {-8, -3, 2, 2, 7}=0

Λ'(0) SU_4 O(101) $\overline{175}$⊃15⊗-8 verts ↦



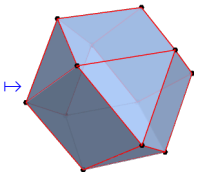
SU_5 O(2022) $SU_5 \supset SU_4 \otimes U_1$

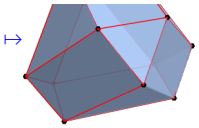


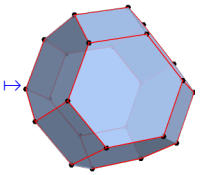
$SU_5 - \Sigma SU_4 = 4410 - 1209 = 3201$ missing

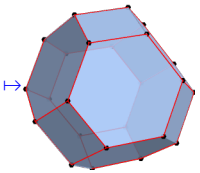
$\Sigma U1$ {-16, -6, 4, 4, 14}=0

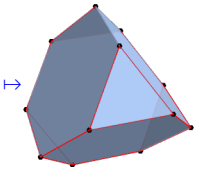
Λ'(0) SU_4 O(202) 4410⊃84⊗-16 verts ↦

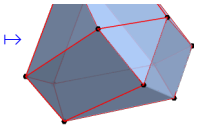


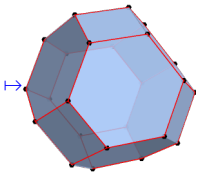
Λ'(1) SU_4 O(102) $\overline{175}\supset\overline{36}\otimes$-3 verts ↦ 

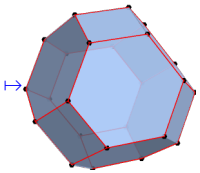
Λ'(2) SU_4 O(111) $\overline{175}\supset 64\otimes 2$ verts ↦ 

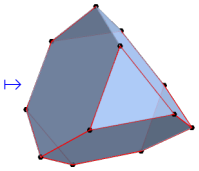
Λ'(3) SU_4 O(111) $\overline{175}\supset 64\otimes 2$ verts ↦ 

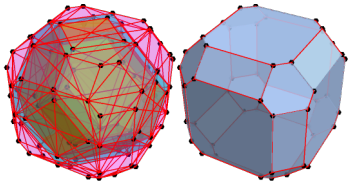
Λ'(4) SU_4 O(011) $\overline{175}\supset 20\otimes 7$ verts ↦ 

Λ'(1) SU_4 O(204) $4410\supset\overline{270}\otimes$-6 verts ↦ 

Λ'(2) SU_4 O(222) $4410\supset 729\otimes 4$ verts ↦ 

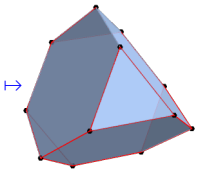
Λ'(3) SU_4 O(222) $4410\supset 729\otimes 4$ verts ↦ 

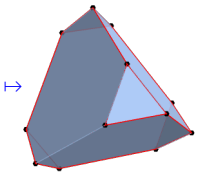
Λ'(4) SU_4 O(022) $4410\supset\overline{126}\otimes 14$ verts ↦ 

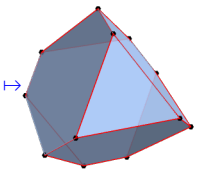
SU_5 O(0111) $SU_5 \supset SU_4 \otimes U_1$ 

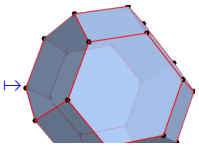
$SU_5-\Sigma SU_4=280-189=91$ missing

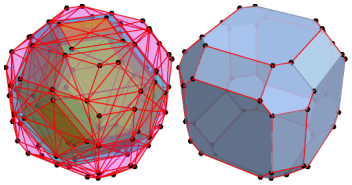
$\Sigma U1$ {-9, -4, 1, 6, 6}=0

Λ'(0) SU_4 O(011) $\overline{280}\supset 20\otimes$-9 verts ↦ 

Λ'(1) SU_4 O(012) $\overline{280}\supset\overline{45}\otimes$-4 verts ↦ 

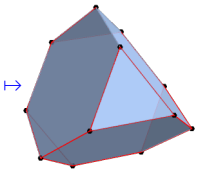
Λ'(2) SU_4 O(021) $\overline{280}\supset\overline{60}\otimes 1$ verts ↦ 

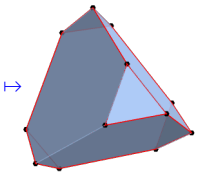
Λ'(3) SU_4 O(111) $\overline{280}\supset 64\otimes 6$ verts ↦ 

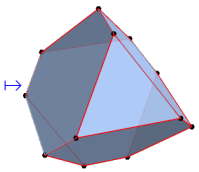
SU_5 O(0222) $SU_5 \supset SU_4 \otimes U_1$ 

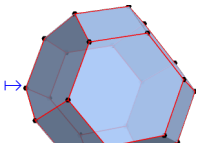
$SU_5-\Sigma SU_4=8505-1755=6750$ missing

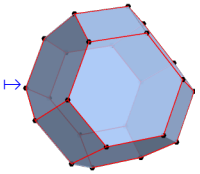
$\Sigma U1$ {-18, -8, 2, 12, 12}=0

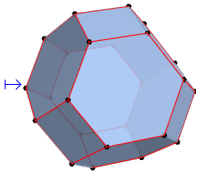
Λ'(0) SU_4 O(022) $\overline{8505}\supset\overline{126}\otimes$-18 verts ↦ 

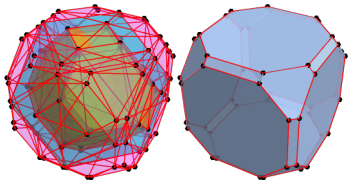
Λ'(1) SU_4 O(024) $\overline{8505}\supset\overline{360'}\otimes$-8 verts ↦ 

Λ'(2) SU_4 O(042) $\overline{8505}\supset\overline{540'}\otimes 2$ verts ↦ 

Λ'(3) SU_4 O(222) $\overline{8505}\supset 729\otimes 12$ verts ↦ 

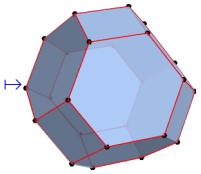
Λ'(4) SU_4 O(111) $\overline{280}$⊃64⊗6 verts ↦ 

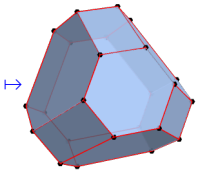
Λ'(4) SU_4 O(222) $\overline{8505}$⊃729⊗12 verts ↦ 

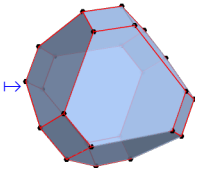
SU_5 O(1111) $SU_5 \supset SU_4 \otimes U_1$ 

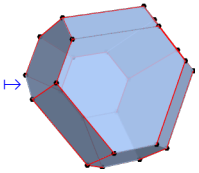
$SU_5 - \Sigma SU_4 = 1024 - 519 = 505$ missing

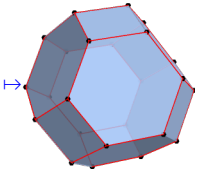
$\Sigma U1$ {-10, -5, 0, 5, 10}=0

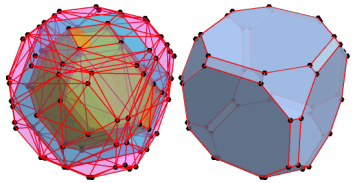
Λ'(0) SU_4 O(111) 1024⊃64⊗-10 verts ↦ 

Λ'(1) SU_4 O(112) 1024⊃140⊗-5 verts ↦ 

Λ'(2) SU_4 O(121) 1024⊃175⊗0 verts ↦ 

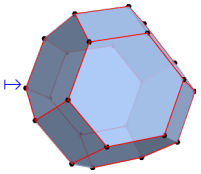
Λ'(3) SU_4 O(211) 1024⊃$\overline{140}$⊗5 verts ↦ 

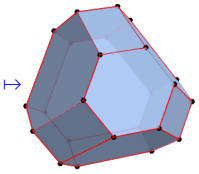
Λ'(4) SU_4 O(111) 1024⊃64⊗10 verts ↦ 

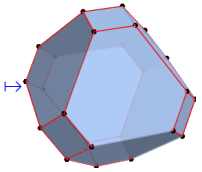
SU_5 O(2222) $SU_5 \supset SU_4 \otimes U_1$ 

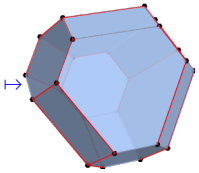
$SU_5 - \Sigma SU_4 = 59\,049 - 7329 = 51\,720$ missing

$\Sigma U1$ {-20, -10, 0, 10, 20}=0

Λ'(0) SU_4 O(222) 59049⊃729⊗-20 verts ↦ 

Λ'(1) SU_4 O(224) 59049⊃$\overline{1980}$⊗-10 verts ↦ 

Λ'(2) SU_4 O(242) 59049⊃2640⊗0 verts ↦ 

Λ'(3) SU_4 O(422) 59049⊃1980⊗10 verts ↦ 

Λ'(4) SU_4 O(222) 59049⊃729⊗20 verts ↦