

# Notes on trilinear coordinates of a triangle

David Eppstein

Being a *Mathematica* notebook wherein the author calculates trilinear coordinates related to the coincidence of two triples of lines in a compass-and-straightedge construction of the Apollonian gasket, as described and illustrated on his web page <http://www.ics.uci.edu/~eppstein/junkyard/tangencies/apollonian.html>

## ■ Basic Notation

Conventions:  $a$ ,  $b$ , and  $c$  always represent the edge lengths (or edges) of a triangle.  $A$ ,  $B$ , and  $C$  always represent the angles (or vertices) of the triangle. All points are represented in trilinear coordinates: a vector  $\{p,q,r\}$  represents a point for which the distances from lines  $a,b,c$  are in proportion  $p:q:r$ . All lines are represented similarly by a triple:  $\{s,t,u\}$  represents the line  $ps+qt+ru=0$ . Therefore the formulae for meets and incidences between points and lines are completely self-dual, so we don't need separate functions for lines and for points.

General-purpose functions will be capitalized, but particular points or lines of construction (even though they look syntactically like functions) will be lowercased.

```
In[5]:= Meet[{p_, q_, r_}, {s_, t_, u_}] :=
      {q u - r t, r s - p u, p t - q s}
```

```
In[6]:= IncidentQ[p_, L_] := Simplify[p . L] == 0
```

```
In[7]:= CoincidentQ[p_, q_, r_] := IncidentQ[Meet[p, q], r]
```

The three triangle vertices:

```
In[8]:= vertex[a] = {1, 0, 0};
      vertex[b] = {0, 1, 0};
      vertex[c] = {0, 0, 1};
```

And the three triangle sides:

```
In[9]:= side[a] = Meet[vertex[b], vertex[c]];
      side[b] = Meet[vertex[a], vertex[c]];
      side[c] = Meet[vertex[a], vertex[b]];
```

## ■ Midpoints and Infinity

Midpoints of triangle edges. The trilinears are equal to half the altitude lengths, which are inversely proportional to side lengths (since  $\text{altitude} \cdot \text{side} = \text{area}/2$ ).

```
In[1]:= midpoint[a, b] = {1/a, 1/b, 0};
        midpoint[a, c] = {1/a, 0, 1/c};
        midpoint[b, c] = {0, 1/b, 1/c};
        midpoint[b, a] = {1/a, 1/b, 0};
        midpoint[c, a] = {1/a, 0, 1/c};
        midpoint[c, b] = {0, 1/b, 1/c};
```

Lines through pairs of midpoints determine parallels to the sides. The extra factor of  $abc$  clears denominators and simplifies the expressions.

```
In[2]:= midline[x_, y_, z_] := xyz Meet[midpoint[x, y], midpoint[x, z]]
```

Define the "slope" of a line to be the point where the given line meets the line at infinity. Since we have a parallel to each side, we can compute their slopes.

```
In[3]:= sideslope[a] = Meet[side[a], midline[a, b, c]];
        sideslope[b] = Meet[side[b], midline[b, c, a]];
        sideslope[c] = Meet[side[c], midline[c, a, b]];
```

These three points determine the line at infinity. The factor of  $-c$  normalizes this to a symmetric representation.

```
In[11]:= CoincidentQ[sideslope[a], sideslope[b], sideslope[c]]
```

```
Out[11]= True
```

```
In[14]:= infiniteLine = Meet[sideslope[a], sideslope[b]] / -c
```

```
Out[14]= {a, b, c}
```

```
In[15]:= Slope[L_] := Meet[L, infiniteLine]
```

## ■ Altitudes and Orthocenter

Ok, I cheated to compute the pedal points (where the altitudes cross the sides) — it's much easier to compute in Cartesian coordinates and then convert to trilinears.

```
In[19]:= pedal[a] = {0, c (a^2 + b^2 - c^2), b (a^2 - b^2 + c^2)};
        pedal[b] = {c (a^2 + b^2 - c^2), 0, a (b^2 + c^2 - a^2)};
        pedal[c] = {b (a^2 - b^2 + c^2), a (b^2 + c^2 - a^2), 0};
```

Once the pedals are known, the altitudes themselves are easy to find.

```
In[21]:= altitude[x_] := Meet[vertex[x], pedal[x]]
```

The orthocenter is the point where the three altitudes meet.

```
In[22]:= CoincidentQ[altitude[a], altitude[b], altitude[c]]
```

```
Out[22]= True
```

```
In[24]:= orthocenter = Meet[altitude[a], altitude[b]]
```

```
Out[24]= {b c (a^2 + b^2 - c^2) (a^2 - b^2 + c^2), a c (a^2 + b^2 - c^2) (-a^2 + b^2 + c^2), a b (a^2 - b^2 + c^2) (-a^2 + b^2 + c^2)}
```

Finally, in order to compute the contact points, we need the slopes of the altitudes.

```
In[25]:= perpendicularSlope[x_] := Slope[altitude[x]]
```

## ■ Angle Bisectors, Incenter, and Contact Points

The angle bisectors are easy to give trilinear coordinates for, because they are the lines equidistant from two triangle sides.

```
In[16]:= bisector[a] = {0, 1, -1};
```

```
        bisector[b] = {1, 0, -1};
```

```
        bisector[c] = {0, 1, -1};
```

```
In[18]:= incenter = -Meet[bisector[a], bisector[b]]
```

```
Out[18]= {1, 1, 1}
```

The inscribed circle meets the triangle sides on radii parallel to the altitudes. The common factor of  $-x$  is removed to simplify the resulting expressions.

```
In[37]:= contact[x_] := Meet[side[x], Meet[incenter, perpendicularSlope[x]]] / -x
```

## ■ Trigonometry, Area, and Height

We now digress from trilinears for a moment to work through some basic formulae of triangle measurement. First, the law of cosines:

```
In[39]:= cosine[A] = (b^2 + c^2 - a^2) / (2 b c);
```

```
        cosine[B] = (a^2 - b^2 + c^2) / (2 a c);
```

```
        cosine[C] = (a^2 + b^2 - c^2) / (2 a b);
```

Next, the radii of the three tangent circles centered on the vertices of the triangle:

```
In[50]:= semiperimeter = (a + b + c) / 2;
         radius[x_] := Simplify[semiperimeter - x]
```

Heron's formula for area from side lengths:

```
In[42]:= area = Simplify[Sqrt[semiperimeter radius[a] radius[b] radius[c]]]
```

```
Out[42]=  $\frac{1}{4} \sqrt{(a+b-c)(a-b+c)(-a+b+c)(a+b+c)}$ 
```

The formula for area from height and base length:

```
In[43]:= height[x_] := (2 area) / x
```

Finally, the law of sines:

```
In[101]:= sine[A] = 2 * area / (b c);
          sine[B] = 2 * area / (a c);
          sine[C] = 2 * area / (a b);
```

## ■ The New Triangle Centers

First we find the points where the altitudes cross the tangent circles. The trilinears below are all actual distances.

```
In[54]:= outcross[a] = {height[a] + radius[a], -radius[a] cosine[C], -radius[a] cosine[B]};
         outcross[b] = {-radius[b] cosine[C], height[b] + radius[b], -radius[b] cosine[A]};
         outcross[c] = {-radius[c] cosine[B], -radius[c] cosine[A], height[c] + radius[c]};
```

```
In[55]:= incross[a] = {height[a] - radius[a], radius[a] cosine[C], radius[a] cosine[B]};
         incross[b] = {radius[b] cosine[C], height[b] - radius[b], radius[b] cosine[A]};
         incross[c] = {radius[c] cosine[B], radius[c] cosine[A], height[c] - radius[c]};
```

Define the lines from the contact points to the crossing points. These lines also pass through the tangencies with the inner and outer Apollonian circles (hence the switch in terminology from outer to inner).

```
In[56]:= outerline[x_] := Meet[contact[x], incross[x]];
         innerline[x_] := Meet[contact[x], outcross[x]]
```

These lines really do form center points:

```
In[59]:= CoincidentQ[outerline[a], outerline[b], outerline[c]]
         CoincidentQ[innerline[a], innerline[b], innerline[c]]
```

```
Out[59]= True
```

```
Out[60]= True
```

Find the new centers! Output omitted to spare sensitive readers...

```
In[61]:= outercenter = Meet[outerline[a], outerline[b]];
        innercenter = Meet[innerline[a], innerline[b]];
```

Note: if you want to see the actual trilinears, the expressions above will not give them to you in symmetric form—you need to divide by the appropriate normalizing factor, or add the three forms given by the formula above and its symmetric counterparts with b,c and c,a.

Each trilinear has the form  $p+q\cdot\text{area}$  where p and q are big ugly polynomials. To get the inner center from the outer one or vice versa, just negate q. (This makes sense because the area involves a square root by Heron's formula; we are just taking the other branch of the square root.)

## ■ Incidence with Gergonne-Incenter Line

Clark Kimberling observed numerically that the two new points seem to be the incidences of the line L(1,7) with the two lines L(226,402) and L(226,403), where (in his notation) L(i,j) stands for Meet[Xi,Xj], X1 is the incenter, X7 is the Gergonne point, and the other three points are not so famous but have reasonably simple trilinears. He then asked if one could prove that this observation is not merely a numerical coincidence. We first deal with L(1,7).

The Gergonne point is the triple incidence between the lines connecting contact points to vertices.

```
In[62]:= gergonneLine[x_] := Meet[vertex[x], contact[x]]

In[64]:= CoincidentQ[gergonneLine[a], gergonneLine[b], gergonneLine[c]]

Out[64]= True

In[69]:= gergonne = Simplify[Meet[gergonneLine[a], gergonneLine[b]]];
```

Now define L(1,7):

```
In[73]:= L17 = Simplify[Meet[incenter, gergonne]];
```

The new centers really are on L(1,7):

```
In[76]:= IncidentQ[outercenter, L17]
        IncidentQ[innercenter, L17]

Out[76]= True

Out[77]= True
```

## ■ Incidence with Other Lines

```
In[119]:= X226 = {bc (b + c) / (b + c - a),  
              ca (c + a) / (c + a - b),  
              ab (a + b) / (a + b - c)};
```

```
In[102]:= tri402[x_] :=  
          Simplify[TrigExpand[Sec[x - Pi / 4]] /. {Cos[x] -> cosine[x], Sin[x] -> sine[x]}]
```

```
In[115]:= X402 = {tri402[A], tri402[B], tri402[C]} / (2 Sqrt[2]);
```

```
In[121]:= CoincidentQ[X226, X402, outercenter]
```

```
Out[121]= True
```

```
In[127]:= tri403[x_] :=  
          Simplify[TrigExpand[Sec[x + Pi / 4]] /. {Cos[x] -> cosine[x], Sin[x] -> sine[x]}]
```

```
In[129]:= X403 = {tri403[A], tri403[B], tri403[C]} / (2 Sqrt[2]);
```

```
In[130]:= CoincidentQ[X226, X403, innercenter]
```

```
Out[130]= True
```