Integer Sided Triangles with Trisectable Angles:

Their Perimeters and Residual

Sam Fischer

Mathematics
Whitman College
Walla Walla Washington
May 15, 2015

Dedicated to Mom, and Dad. Thank you so much. I'm so thankful to have the chance to make you happy and proud.

Abstract

Every triangle with integer sides and trisectible angles can be characterized by it's perimeter and a square free number called a residual. This paper describes the creation of a comprehensive list of triangles with trisectible angles and integer sides up to perimeter 10,000 and discusses how the perimeters of those triangles are related to their residuals.

Acknowledgements

Without the efforts of Professor R. A. Gordon and Professor P. Keef this paper, which is a culminating accomplishment for me, would not have been possible. To them and to the Whitman math department as a whole I am deeply indebted. Not only for classes like Data Structures, Mathematical Modeling, and Geometry, (taught by Professors A. Schueller, D. Hundley, and R. A. Gordon, respectively) which gave me the tools to take ownership for this problem, but for much deeper reasons as well. When I came to Whitman I wanted more than anything to grow into a person of outstanding character, but I don't think I understood what that meant until I began my classes, and more specifically, my classes with this math department. I realized in Professor D. Guichard's beginning calculus class my first semester if I didn't push myself to take ownership of the material I would surely fail. Translating this to a life lesson that I have tried to live out since then has been a process shepherded by my experiences in the classes of Olin hall. To the reader I would say one thing: if there are faults in this paper, then they are solely my own and I apologize for their obviousness, but, if there is a steady rightness to my character then it is at least in part to the credit of this department and I apologize that it might not have shown through more on these pages.

Contents

1 Introduction

In Integer Sided Triangles with Trisectible Angles *(A brief note on spelling. I have gone with the convention of 'constructable' and 'trisectible'.)* R. Gordon was able to determine methods for finding integer-sided triangles with three trisectible angles. One question which was opened through his research was how the residuals of the triangles are generally related to their perimeters. This paper aims to explore exactly that question. In doing so the paper will first lay down a theoretical foundation, moving through theory of constructability, trisectability, and then residuals. After doing so the paper will then describe the methods designed for researching this question which involved both programming and data analysis. After this discussion of implementation will come the results and then analysis of those results. To conclude the paper some further questions and topics are listed. *(A brief note on spelling. I have gone with the convention of 'constructable' and 'trisectible'.)*

2 constructable angles and trisectible angles

Before we can investigate triangles with trisectible angles, we have to define what it means for an angle to be trisectible. Doing so will lead us to a general definition of what it means for an angle to be constructable.

2.1 constructable Numbers

In geometry a length is constructable when it can be constructed with a finite number of steps using a compass and a straightedge. We formalize this in the following definition.

Definition 1. *A real number ρ is constructable if, given a line segment of length x, a line segment of length $|\rho|x$ can be constructed with a compass and unmarked straightedge.*

Now that we have a definition, we are motivated to determine which of the real numbers are constructable.

Theorem 1. *The integers are constructable*

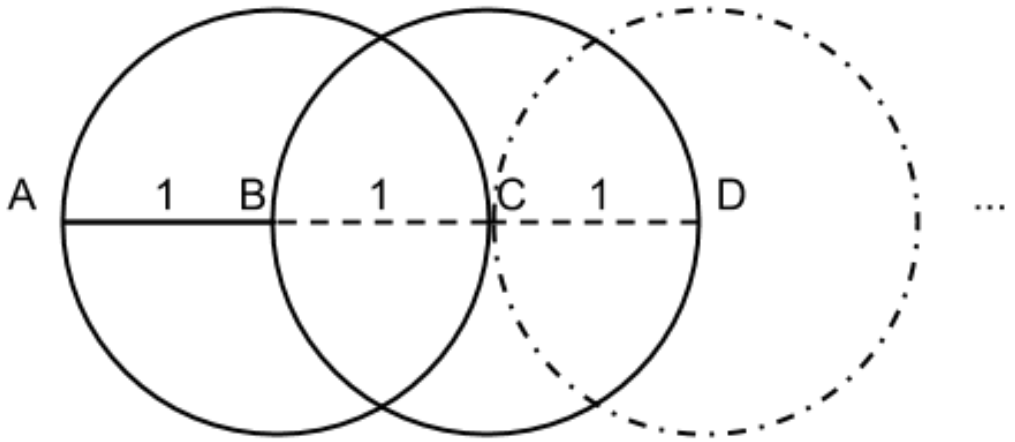


Figure 1: Constructability of the Integers

Proof. Let z be an integer. We begin with a line segment AB (see Figure 1) of length 1 and seek to construct from that segment a line segment of length $|z|$. Using our compass we take one endpoint, B, of our line segment of length 1 to be the center, and the other endpoint, A, to define the extent of our radius. We then draw a circle with radius 1. Using our straightedge we extend our line segment AB through to where it intersects with the circle at point C, giving us a diameter AC of length 2 (as $r = 1$). We repeat this process again, but with our new point of intersection, C as the center of the new circle, and with our old center, B as the extent of our radius. Extending AC to its point of intersection with the new circle at point D we have a line segment of length $2 + 1 = 3$. We can continue this process until we reach a line segment of length $|z|$, and therefore, as we have constructed a line segment of length $|z| \cdot 1$ we have shown that z is constructable. As z was picked arbitrarily, we conclude that this must be true for all integers z, and that therefore all integers are constructable numbers. □

To show the rational numbers, a larger subset of the real numbers which are closely related to the integers, are constructable would be a logical step to take. To do this we will first prove a helpful lemma.

Lemma 1 (Side-Splitter Theorem)**.** *A line drawn inside a triangle parallel to any of the sides creates two similar triangles with proportional sides.*

Proof. Suppose we have a triangle XYZ with side lengths A,C, and B and there is a line segment ST in the interior of the triangle which is parallel

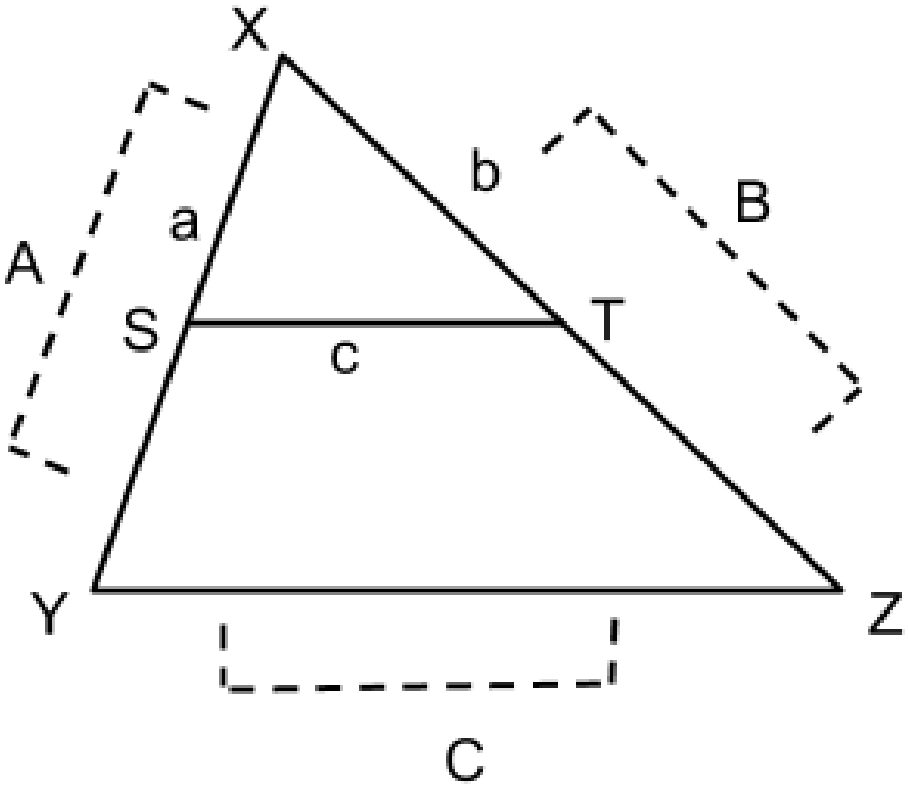


Figure 2: Side-Splitter Theorem

to one of the sides, say YZ (see Figure 2. By proposition 29 of Euclid's first book of the elements [3] we know that angle $\angle XST = \angle XYZ$ and $\angle XTS = \angle XZY$. By definition angle $\angle YXZ = \angle SXT$, and therefore that the two triangles ΔYXZ and ΔSXT are similar. By the definition of similarity provided by Euclid [3], we know that the ratio of side A to side a and side B to side b and side C to side c are all of the same proportion. □

Theorem 2. *The rational numbers are constructable*

Proof. Since we have already shown that the integers are constructable, all we have left is to show that the rational numbers which are not integers are constructable. Let r be a positive rational number that is not an integer. By definition there exist positive integers p and q such that $r = p/q$ with

$q \neq 1$ and either $p > q$ or $q > p$. Suppose $q > p$. We begin with a line segment of length 1 and using the methods described in the proof that the integers are constructable we can construct line segments of length $|p|$ and $|q|$. We label the line segment of length $|q|$ to be AE (see Figure 3). Using the methods described in proposition 3 of Euclid's first book of the elements [3] we cut a line segment of length $|p|$ on length AE and label this length AC. We intersect, at an arbitrary positive angle, a line segment through AE at point A. Using the methods describe in proposition 3 again, we cut a length of 1 on this segment and label it AD. We then connect points D and E with our straightedge and using the methods described in proposition 31 in Euclid's first book of the elements [3] we construct a parallel line to DE through point C until it intersects with AD at point B. By our lemma we know that these two triangles must be similar and therefore we know that the proportion AB/AD must be equal to the proportion AC/AE, but as AD is equal to length 1, and AE is equal to length $|q|$ and AC is equal to length $|p|$ we know that $AD = |p|/|q| = |p/q| = |r| \cdot 1$ as desired. In the case where $p > q$ it follows that we can create the same construction but where $AC = |q|$, $AE = |p|$, and $AB = 1$. Because r was picked arbitrarily out of the set of $\mathbb{Q}\backslash\mathbb{Z}$ we know that all rationals in the set $\mathbb{Q}\backslash\mathbb{Z}$ are constructable, and as we already proved all integers are constructable we conclude that all rationals are constructable. □

With a general idea of what a constructable number might look like, we explore the operations the constructable numbers generally closed under.

Theorem 3. *The constructable numbers are closed under addition and subtraction.*

Proof. Suppose we have two constructable numbers a and b and suppose they are of the same sign. It follows by the definition of a constructable number that we can construct a length $|b| \cdot 1$ and $|a| \cdot 1$ given a segment of length 1. To show that $a + b$ is constructable we must show that length $|a+b| \cdot 1$ is constructable given a segment of length 1. Because it is given that we can construct a line segment of length $|a|$ and length $|b|$ we begin with two segments of these lengths, AB and CD, respectively (see Figure 4). By Proposition 2 of Euclid's first book of the elements [3] we are able to place a straight line equal to a given straight line with one end at a given point, and therefore we are able to place a straight line of length $|b|$ at point B of line segment AB. By drawing a circle of radius $|b|$ centered at B and extending

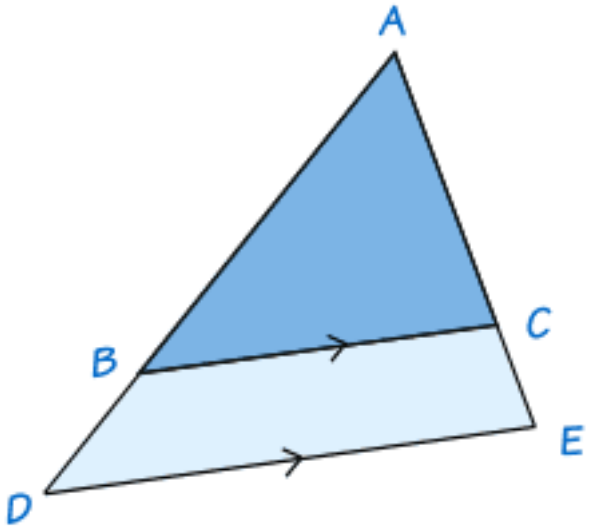


Figure 3: Proof that the rational numbers are constructable.

AB with our straightedge through this circle, we construct a line segment of length $|a| + |b| = |a + b|$. We have therefore shown that when a and b are constructable numbers with the same sign they are closed under addition. This also gives us a proof that the constructable numbers are closed under subtraction when one number, a, is positive and the other, b, is negative.

Now suppose we have two constructable numbers a and b and suppose they have opposite signs. Then, assuming $|b| > |a|$ it follows that $|a + b| = |b| - |a|$ and the construction for this is given by Euclid in the third proposition in his first book of elements [3]. This also gives us a proof that the constructable numbers are closed under subtraction when one number, a, is positive and the other, b is positive.

We have therefore shown that the constructable numbers are closed under addition and subtraction. □

Theorem 4. *The constructable numbers are closed under multiplication.*

Proof. Suppose we have two constructable numbers a and b. It follows by the definition of a constructable number that we can construct a length $|a| \cdot 1$ and $|b| \cdot 1$ given a segment of length 1. To show that ab is constructable (see Figure 5) we must show that length $|ab| \cdot 1$ is constructable given a line

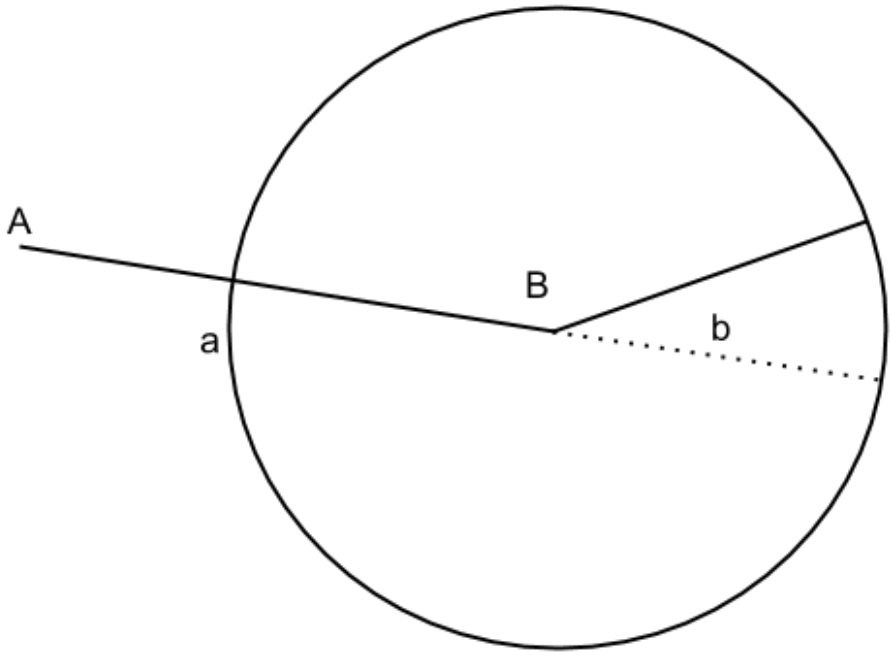


Figure 4: The constructable numbers are closed under addition

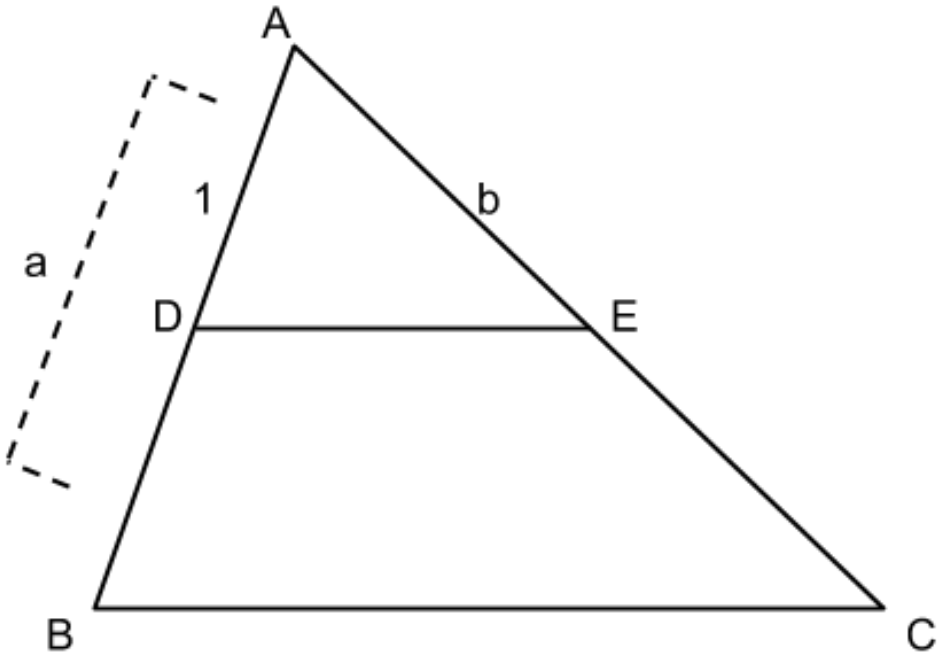


Figure 5: Proof that the constructable numbers are closed under multiplication

segment of length 1. The method of proof is the same as showing that the rational numbers are constructable. We begin with our line segment of length $|a|$, AB and then using proposition 3 from Euclid's first book of elements we cut a length 1 on AB and label the end point of this cut D. Using our straight edge we draw a line through point A which has a positive angle between it and line AB. Using proposition two of Euclid's first book of the elements we cut a length $|b|$ this line and the segment AE. We then connect D to E, and by proposition 31 in Euclid's first book of element's we construct a line parallel to DE from point B and then extend AE until it intersects with this parallel line at a point C. By the Side-Splitter Theorem we know that the proportion $AB/AD = AC/AE$. Because $AD = 1$, $AB = |a|$, and $AE = |b|$ then $|a| = AC/|b|$ or $|a||b| = AC$, or $|ab| = AC$. So we have shown that in the case of $a = b$ that ab is a constructable number. □

Theorem 5. *The constructable numbers are closed under division*

Proof. Suppose we have two constructable numbers a and b. It follows by the definition of a constructable number that we can construct a length $|a| \cdot 1$ and $|b| \cdot 1$ given a segment of length 1. To show that a/b is constructable we must show that length $|a/b| \cdot 1$ is constructable given a line segment of length 1. Note that in the case where $a = b$ we must construct the length $|a/b| = |a/a| = 1$ which is constructable as the integers are constructable and 1 is an integer. Therefore we consider the case where either $a < b$ or $b < a$. Without loss of generality suppose $a < b$. Then, as a and b are constructable we begin with line segments of length $|a|$ and $|b|$. Let line segment AB be of length $|b|$ (see Figure 6) and then using proposition 3 from Euclid's first book of elements [3] cut a length $|a|$ on AB and label the point D. Then using proposition 2 from Euclid's first book of elements connect a line segment of length 1 onto AB at any angle, so long as the two lines are not co-linear, and then label this line AE. Connect D and E. By proposition 31 of Euclid's first book of elements [3] construct a parallel line to DE through point B and extend AE until it intersects with this parallel line at a point C. By the Side-Splitter Theorem it follows that the two triangles ΔADE and ABC are proportional and that therefore the ratio of sides AB/AD and AC/AE are equal. Because $AD = |a|$ and $AB = |b|$ and $AE = 1$ it follows that $|b|/|a| = AC/1$ or $|b/a| = AC$. We have shown that given two constructable numbers a and b we can construct a length $|a/b| \cdot 1$ given a line segment of length 1, and therefore we have shown that a/b is a constructable number. As a and b were picked generically, and the decision to make $a < b$ does not let us lose generality it follows that the constructable numbers are closed under division. □

To prove that the constructable numbers are closed under the extraction of square roots we will prove a useful lemma: Thales' theorem.

Lemma 2. *If A, B and C are points on a circle where the line segment AC forms a diameter of that circle, then the angle $\angle ABC$ is a right angle.*

Proof. We with three points A, B, C on a circle (see Figure 7) where the only condition on those points is that the line segment AC form a diameter of that circle. By proposition 10 of Euclid's first book of elements we can bisect the diameter AC at point O creating two radii of equal length, OA and OC. Connecting center O to point on the circle B we form another radius such that $OA = OB = OC$. Connecting A to B to C we create two triangles isosceles triangles ΔAOB and ΔCOB. By proposition five of Euclid's first book of the

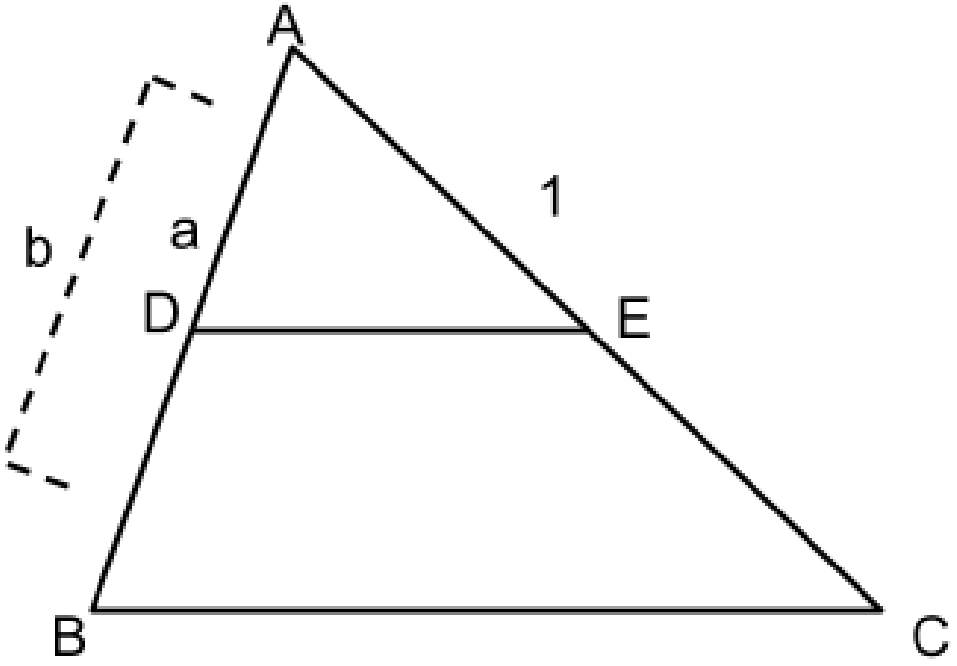


Figure 6: Closure of constructable numbers by division

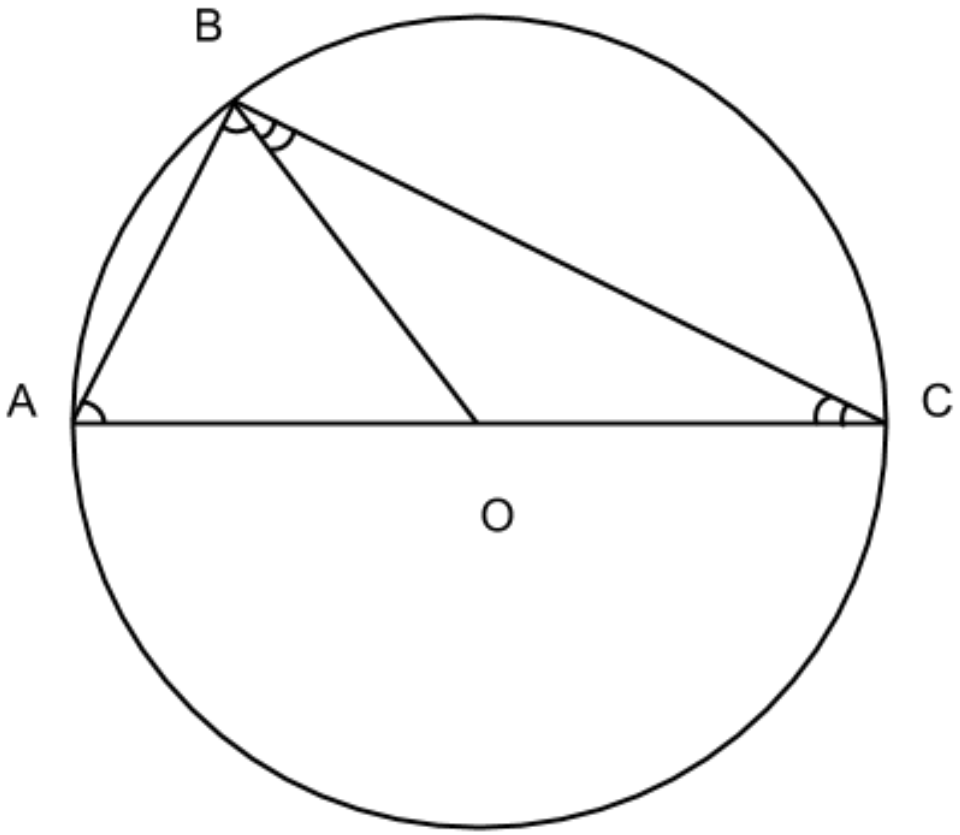


Figure 7: Thales Theorem

elements [3] it follows that the base angles of these respective triangles must be equal, and therefore $\angle OAB = \angle OBA$ and $\angle OCB = \angle OBC$. Because triangle ΔABC is a triangle we know that the sum of the interior angles must be equal to 180 degrees, and therefore $\angle ABC + \angle BCA + \angle CAB = 180°$. However we know that angle $\angle ABC = \angle BCA + \angle CAB$ and therefore that $\angle ABC + \angle BCA + \angle CAB = 2\angle ABC$ and therefore $\angle ABC = 90°$ □

Theorem 6. *The constructable numbers are closed under extraction of square roots.*

Proof. Suppose we have a constructable number a. It follows by the definition

of a constructable number that we can construct a length $|a| \cdot 1$ if given a segment of length 1. To show that $\sqrt{a}$ is constructable we must show that length $|\sqrt{a}| \cdot 1$ is constructable given a line segment of length 1.

We begin with our given length of 1, GF (see Figure 8), and by proposition 3 in Euclid's first book of elements [3] append onto it a segment HG of length $|a|$. We then construct a circle of diameter HF and using proposition 12 from Euclid's first book of the elements [3] draw a perpendicular line from G extending it to a point of intersection K with the circle. Notice that if we connect line segments HK and KF then we get three right triangles: ΔHKG, ΔFKG and ΔHKF. We know the first two triangles are right because they contain the perpendicular drawn at point G. Triangle ΔHKF is a right triangle because of Thales' theorem. From these similar triangles we get that the proportionality of the legs KG and GF in triangle ΔFKG must be equal to the proportionality of the legs HG and KG in the triangle ΔHKG. This means that $KG/GF = HG/GK$, and because GF is our given length of 1 we get that $KG^2 = HG$ or $KG = \sqrt{HG}$.

□

Knowing the constructable numbers are closed under the five above operations, and the rational numbers are constructable, we are motivated to prove this general theorem about constructable numbers.

Theorem 7. *A number is constructable if and only if it can be obtained from rational numbers a finite sequence of additions, subtractions, multiplications, divisions, and square roots.*

Proof. If a number can be obtained from a finite sequence of additions, subtractions, multiplications, divisions, and square roots of rational numbers then by theorem's 2 through 6 we know that the number must be able to be geometrically constructed using only a compass and straightedge.

To show that a constructable number must always be constructed in this way, we imagine a Cartesian plane (with the rational numbers as points) where all our constructions take place. With a compass and a straightedge we are able to construct all needed constructable numbers. It follows then that all constructable numbers can be described as the intersection of a line and a line, a line and a circle, or a circle and a circle. In the second and third case what will result is a quadratic equation whose roots will be the desired constructable number. Sometimes the root of this quadratic will be rational, however, the root will never have a radical greater than a square root.

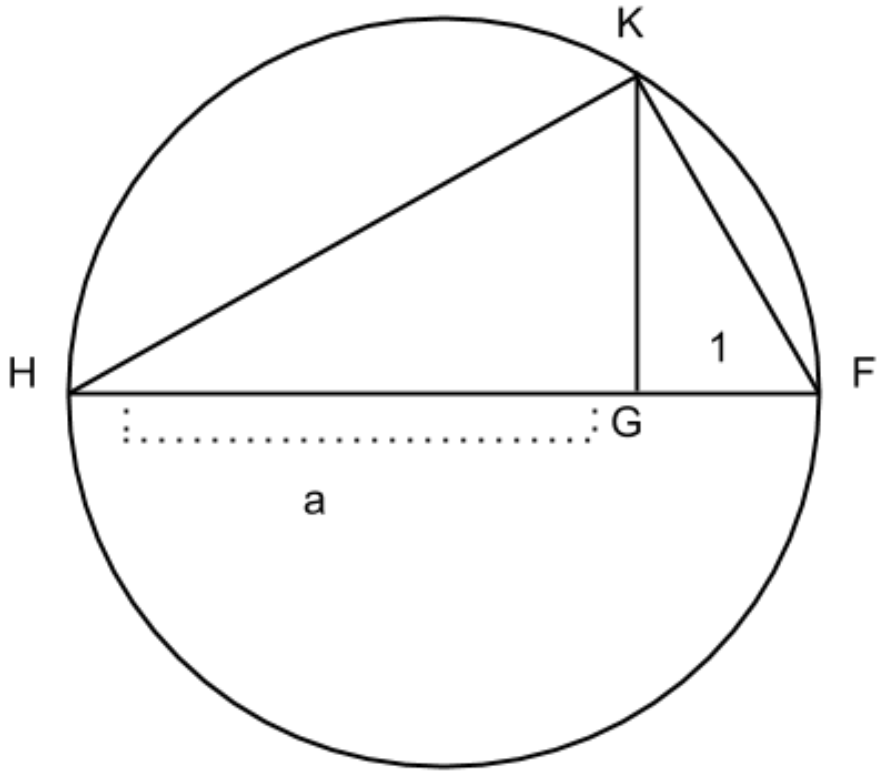


Figure 8: Closure of the constructable numbers by the extraction of square roots

Now if we there are produced points with square roots, we can begin the process of intersecting circles with lines again and produce points which can be either rational or involve a square root, which would give us a square root of a square root. In this way we see that all constructable numbers are formed by a finite sequence of addition, subtraction, multiplication, division, and extraction of square roots on the rational numbers.

□

Just as we began our discussion of construct ability with geometric description, and then gave a arithmetic defintion so will we now prove a theorem about the constructability of numbers in terms of being roots of a cubic equation. This will require some lemmas and definitions.

Lemma 3 (The Factor Theorem)**.** *If r is a root of a polynomial $p(x)$, then the factor $(x - r)$ divides $p(x)$ evenly.*

Proof. Let $p(x)$ be a polynomial of positive degree and let r be a root of $p(x)$. It follows by the [help me professor!] (division property?) that we can express $p(x)$ as the product of $(x - r)$ times a quotient polynomial $q(x)$ plus a remainder polynomial of degree less than the degree of $(x-r)$. Because the degree of $(x - r)$ is one it follows that our remainder must be a constant R. We then have the equality $p(x) = (x-r)q(x)+R$. Plugging in $x = r$ into this equation we see that $0 = 0 * q(x) + R$ or that $R = 0$. Because our remainder was zero, the factor divided without remainder into the polynomial. □

Lemma 4. *If the roots of a cubic equation $x^3 + bx^2 + cx + d = 0$ are r_1, r_2, and r_3, then $b = -(r_1 + r_2 + r_3)$.*

Proof. By the factor theorem and the fact that the coefficient of x^3 is one, it follows that each factor will evenly divide our polynomial, and therefore

$$x^3 + bx^2 + cx + d = 0$$

is the same as $(x - r_1)(x - r_2)(x - r_3) = 0$. Multiplying these factors out gives us that the coefficient of x^2 is $-(r_1 + r_2 + r_3)$ and therefore $b = -(r_1 + r_2 + r_3)$. □

At this point in the proof it is nessessary to introduce some notation.

Definition 2 (Conjugate)**.** *Let x be an element in the set $\{a + b\sqrt{r}|a, b, r \in \mathbb{Q}$ and r is square free$\}$. The conjugate of x, denoted as $\overline{x}$, is given as $a-b\sqrt{r}$ where $x = a + b\sqrt{r}$.*

Lemma 5. *The conjugate of the sum of two elements in the set*

$$\{a + b\sqrt{r} | a, b, r \in \mathbb{Q} \text{ and } r \text{ is square free}\}$$

is equal to the sum of the conjugates, and the product of two elements of that set is the product of the conjugates.

Proof. Let $a, b, c, d \in \mathbb{Q}$ and observe that

$$\overline{(a + b\sqrt{r}) + (c + d\sqrt{r})} = \overline{(a + c) + (b + d)\sqrt{r}} = (a + c) - (b + d)(\sqrt{r})$$

$$= (a - b\sqrt{r}) + (c - d\sqrt{r}) = \overline{(a + b\sqrt{r})} + \overline{c + d\sqrt{r}}$$

which proves the case for addition, and for multiplication observe

$$\overline{(a + b\sqrt{r})(c + d\sqrt{r})} = \overline{(ac + rbd) + (ad + bc)\sqrt{r}} = (ac + rbd) - (ad + bc)\sqrt{r}$$

and

$$\overline{(a + b\sqrt{r})} \cdot \overline{(c + d\sqrt{r})} = (a - b\sqrt{r}) \cdot (c - d\sqrt{r}) = (ac + bdr) - (ad + bc)\sqrt{r}$$

and therefore

$$\overline{(a + b\sqrt{r})(c + d\sqrt{r})} = \overline{(a + b\sqrt{r})} \cdot \overline{(c + d\sqrt{r})}.$$

□

Lemma 6. *Let $R = a + b\sqrt{r}$ where $a, b, r \in \mathbb{Q}$ and r is square free. If R is the root of a polynomial with rational coefficients then its conjugate $\overline{R} = a - b\sqrt{r}$ is also a root of the polynomial.*

Proof. Suppose we have some polynomial

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0$$

where $a_i \in \mathbb{Q}$ for all $0 \leq i \leq n$ such that

$$p(a + b\sqrt{r}) = a_n(a + b\sqrt{r})^n + a_{n-1}(a + b\sqrt{r})^{n-1} + ... + a_1(a + b\sqrt{r}) + a_0 = 0.$$

If we take the conjugate of both sides of this equation, we see that because the conjugate of zero is zero

$$\overline{a_n(a + b\sqrt{r})^n + a_{n-1}(a + b\sqrt{r})^{n-1} + ... + a_1(a + b\sqrt{r}) + a_0} = 0.$$

From the previous lemma, which says that the conjugate of a sum is equal to the sum of the conjugates, it follows that we can express the left side of the equation as

$$\overline{a_n(a+b\sqrt{r})^n} + \overline{a_{n-1}(a+b\sqrt{r})^{n-1}} + ... + \overline{a_1(a+b\sqrt{r}) + a_0}.$$

Now since the conjugate of every rational number is just itself and a_i for $0 \leq i \leq n$ is a rational number we know that the conjugate of $a + b\sqrt{r}$ is a root of our polynomial, as none of the coefficients have changed. □

One result from number theory

Lemma 7. *An irrational and a rational cannot sum to make a rational.*

Proof. By hypothesis let $a \in \mathbb{Q}$ and $s \notin \mathbb{Q}$. This will be a proof by contradiction. Suppose that the sum $a + s$ was a rational number b. By definition there exist integers p and q such that $p/q = a$ and integers m and n such that $m/n = b$. The equation $a + s = b$ leads us to the equation $p/q + s = m/n$ which implies $s = m/n - p/q = (mq - np)/(nq)$ which, as m, n, p, and q were supposed to be integers, would make s a rational number. However this is a contradiction as s was supposed to be an irrational number. Therefore we conclude that $a + s$ cannot be equal to a rational number when a is a rational number and s is not. □

Now onto our main theorem

Theorem 8. *A cubic equation with rational coefficients has a constructable root if and only if the equation has a rational root.*

Proof. By hypothesis let $p(x)$ be a cubic polynomial with rational coefficients. Dividing through by the leading coefficient if necessary, we can assume that the leading coefficient is one. By hypothesis we suppose that $p(x)$ has a constructable root $R_1 = a + b\sqrt{r}$ where $a, b, r \in \mathbb{Q}$. Then by our lemma 6 we know that there exists a second root R_2 such that $R_2 = \overline{R_1} = a - b\sqrt{r}$. Let the third root of our cubic equation be s. By lemma 4, the sum of R_1, R_2, and s is the rational coefficient of our cubic equation, and therefore $a + b\sqrt{r} + a + b - \sqrt{r} + s$ or $2a + s$ is equal to our rational coefficient. By lemma 7 it follows that as a is rational s must also be rational, and therefore our cubic equation with rational coefficients has a rational root s when it has a constructable root.

If $p(x)$ has a rational root, r, to begin with, then by theorem 2 it follows that $p(x)$ has a constructable root. □

2.2 constructable Angles

With a general theory of constructable numbers and our sights set on describing trisectible angles we are motivated to begin a discussion on the theory of constructable angles. Because lengths and angles are related well by trigonometric functions we begin our theory there.

Theorem 9. *An angle θ is constructable if and only if* $\cos(\theta)$ *is a constructable number. In particular, if* $\cos(\theta)$ *is a rational number then θ is constructable.*

Proof. By hypothesis construct an angle θ which occurs at the intersection of two line α and β at a point C. By proposition 11 from Euclid's first book of the elements [3] we can construct a line from one of the points (A) perpendicular to α. If we call this line γ we can extend γ until it intersects with β at point B. The result is a right triangle, where the cosine of θ measures the quotient of side CA to CB. By theorem 5 it follows that CA/CB is a constructable number and therefore the $\cos(\theta)$ is a constructable number as well.

If $\cos(\theta)$ is a constructable length then by theorem 6, 4, and theorem 3 we know that the length $\sqrt{1-\cos^2(\theta)}$ must also be constructable. If we begin, by hypothesis, with a line segment AB of length $\cos(\theta)$ it follows by this fact and proposition 11 of Euclid's first book of elements [3] that we can construct a perpendicular line segment BC of length $\sqrt{1-\cos^2(\theta)}$. Connecting line segment AC we see that we have a triangle whose hypotenuse is one. Therefore if we take the cosine of the angle α located at the intersect of lines AB and AC we see that it is equal to $\cos(\theta)/1$. Therefore $\cos(\alpha) = \cos(\theta)$. Now because we have constructed this triangle, we can construct a congruent triangle which is essentially the current one flipped over a vertical axis going through point A. It follows that the angle α or the angle created by adding α to the angle between the two hypothesis will be the angle θ and therefore θ is constructable.

An alternate way to do this is to construct a unit circle (a circle of radius one), construct two perpendicular axis' (using proposition 11 from Euclid's first book of the elements [3]), cut lengths of $\cos(\theta)$ on both sides of one of the axis (using proposition 3 [3]), draw perpendiculars up to the circle (using proposition 11 [3]), and then complete the triangles by connecting line segments back to the center. One of the resulting triangles is guaranteed to contain an angle equal to θ. □

2.3 Trisectible Angles

Because in this study we are interested only in triangles with integer sides, by the law of cosines which says that

$$c^2 = a^2 + b^2 - 2ab\cos(C)$$

we will only be interested in interior angles which result in rational cosine values. With that in mind we come to this critical theorem.

Theorem 10. *Suppose that θ is an angle such that $\cos\theta$ is a rational number. The angle θ can be trisected if and only if there exists a rational number $t \in (-1, 1)$ such that $\cos\theta = 4t^3 - 3t$.*

Proof. By hypothesis let θ be an angle such that $\cos\theta$ is a rational number. By theorems 2 and 9 it follows that θ is a constructable angle. The triple angle formula, which is given as

$$\cos(3\theta) = 4\cos^3(\theta) - 3\cos\theta,$$

can be reformulated as

$$\cos(\theta) = 4\cos^3(\theta/3) - 3\cos(\theta/3).$$

which suggests that $\cos(\theta/3)$ is a root of the polynomial

$$0 = 4t^3 - 3t - \cos(\theta).$$

By theorems 9 and 8 we obtain the following set of equivalences.

The angle θ is trisectible if and only if

1. the angle $\theta/3$ is constructable
2. $\cos(\theta/3)$ is a constructable number
3. the polynomial $4x^3 - 3x - \cos\theta$ has a rational root
4. there is a rational number $t \in (-1, 1)$ such that $\cos(\theta) = 4t^3 - 3t$

We limit t to the range $(-1, 1)$ because $f(1)$ and $f(-1)$ map to points in the range of $f(x)$ over the domain of $(-1, 1)$. To seee this observe that if $f(x)$ is the polynomial $4t^3 - 3t - \cos(\theta)$, then

$$f(-1) = -4 + 3 - \cos(\theta) = -1 - \cos(\theta) = 1/2 - 3/2 - \cos(\theta) = f(1/2)$$

and

$$f(1) = 4 - 3 - \cos(\theta) = 1 - \cos(\theta) = -1/2 + 3/2 - \cos(\theta) = f(-1/2).$$

Therefore, rather than writing the domain of $f(x)$ as $[-1, 1]$ we can just write it as $(-1, 1)$.

□

For our study, a theorem relating the trisectability of two interior angles to the third interior angle of a triangle would be useful and such a theorem is given below.

Theorem 11. *If two angles of a triangle are trisectible, then the third angle is as well.*

Proof. Let α, β and γ be the three angles of a triangle, and by hypothesis let α and β be trisectible. Note that $\alpha + \beta + \gamma = \pi$ and that therefore $\gamma = \pi - \alpha - \beta$. Note also that $\cos(\pi/3)$ is a rational number–1/2–and therefore by theorem 2 $\cos(\pi/3)$ is a constructable number and by theorem 9 $\pi/3$ is a constructable angle. By hypothesis α and β are trisectible, and it therefore follows by theorem 10 and its equivalences that $\alpha/3$ and $\beta/3$ must also be constructable. By the fact that $\gamma/3 = \pi/3 - \alpha/3 - \beta/3$, and theorem 3 and 9, we conclude that $\gamma/3$ must be a constructable angle. By theorem 10 this means that γ is trisectible. □

We can also derive a similar result about an angle θ and its supplementary angle.

Theorem 12. *An angle θ is trisectible if and only if its supplementary angle $\pi - \theta$ is trisectible.*

Proof. By hypothesis suppose that θ is a trisectible angle. By theorem 10 and its equivalences it follows that $\theta/3$ is a constructable angle. Since $\pi/3$ is a constructable angle (see the proof of theorem 11) it follows that the difference of $\pi/3 - \theta/3$ is also constructable. Since $\pi/3 - \theta/3$ is constructable, it follows by theorem 10 and its equivalences that $\pi - \theta$ is trisectible. □

This leads us to be able to condense one part of theorem 10.

Theorem 13. *An angle θ is trisectible if $|\cos(\theta)| = |4t^3 - 3t|$ for some rational number $t \in (0, 1)$.*

Proof. Note that the function $|\cos(\theta)|$ can be written piecewise as

$$\cos(\theta) = \begin{cases} \cos(\theta) & : 0 \leq \theta \leq \pi/2 \\ \cos(\pi - \theta) & : \pi/2 \leq \theta \leq \pi \end{cases}$$

as we showed in the previous theorem that θ is trisectible if and only if $\pi - \theta$ is trisectible. This means that $|\cos(\theta)|$ is trisectible if there exists a rational number $t \in (0, 1)$ such that $|\cos(\theta)| = |4t^3 - 3t|$. □

In order to build a list of constructable cosine values (which we can search through in our program if need be) we are led to insert a rational value y/z where $1 \leq y < z$ and y and z are relatively prime. It follows that we can describe the rational cosine values in three distinct cases based on three distinct ways to form z.

1. z is odd
2. z is even and w is odd with $z = 2w$
3. z is even and w is even with $z = 2w$

Through analyse of these three cases we find that we can describe the trisectible cosines in the way proved below.

Theorem 14. *the angle θ is trisectible if and only if $\cos(\theta) = m/n$ where m and n are relatively prime and are constructed in one of*

1. *$m = y|4y^2 - 3z^2|, n = z^3$, with z being odd, the GCD(y,z) = 1, and $1 \leq y < z$*
2. *$m = \frac{1}{2}y|y^2 - 3w^2|, n = w^3$, with w and y being odd, the GCD(y,w) = 1, and $1 \leq y < 2w$*
3. *$m = y|y^2 - 3w^2|, n = 2w^3$, with w being even, the GCD(y,w) = 1, and $1 \leq y < 2w$*

Proof. Plugging in y/z for t in our equation for cosine we get that $\cos(\theta) = |4(y/z)^3 - 3(y/z)| = y|4y^2 - 3z^2|/z^3$ We examine each of the three cases individually

1. By hypothesis, z and y are relatively prime and therefore $\gcd(z, y) = 1$. It follows therefore that $\gcd(z^3, y) = 1$ as well, as if there was a common divisor to z^3 and y greater than 1 it would be the gcd of z and y which is a contradiction. All we have left to check is whether z is relatively prime to $4y^2 - 3z^2$. Again, by the fact that z and y are relatively prime, $\gcd(z^3, y^2) = 1$ as the latter having a greater common divisor would imply the falsity of the former causing a contradiction. Furthermore $\gcd(z^3, 4) = 1$ as z is odd and cubing z could not introduce a 2 into it's prime factorization. Therefore $\gcd(z^3, 4y^2) = 1$. From this we conclude that $\gcd(z^3, 4y^2 - 3z^2) = 1$. Therefore in the case that z is odd, with $\gcd(y, z) = 1$ and $1 \leq y < z$

$$m = y|4y^2 - 3z^2|, n = z^3 \text{ with } z \text{ odd, } \gcd(y, z) = 1, \text{ and } 1 \leq y < z.$$

2. By hypothesis suppose that z is even and w is odd with $z = 2w$. Then we have that

$$\frac{y|4y^2 - 3z^2|}{z^3} = \frac{y|4y^2 - 12w^2|}{8w^3} = \frac{y|y^2 - 3w^2|}{2w^3}.$$

Now as $\gcd(z, y) = 1$ and $z = 2w$ it follows that $\gcd(2w, y) = 1$, and y must necessarily be odd, else the gcd of $2w$ and y would be at least 2. From this we also know that $\gcd(w, y) = 1$ as $\gcd(2, y) = 1$ and $\gcd(2w, y) = 1$. It therefore also follows that $\gcd(w, y^2) = 1$ and that therefore $\gcd(w^3, y^2) = 1$. This means that $\gcd(w^3, y^2 - 3w^2) = 1$ as if it was equal to some greater integer it would lead to a contradiction. Therefore in the case that z is odd, and w is odd with $z = 2w$ and $\gcd(y, z) = 1$ and $1 \leq y < z$

$$m = \frac{1}{2}y|y^2 - 3w^2|, n = w^3 \text{ with } w, y \text{ odd, } \gcd(y, w) = 1, \text{ and } 1 \leq y < 2w.$$

3. By hypothesis suppose that z is even and w is even with $z = 2w$. It follows that

$$\frac{y|4y^2 - 3z^2|}{z^3} = \frac{y|4y^2 - 12w^2|}{8w^3} = \frac{y|y^2 - 3w^2|}{2w^3}.$$

To show that the numerator and denominator of this fraction are relatively prime observe that as $\gcd(2w, y) = 1$ it follows that $\gcd(w, y) = 1$

and that therefore the $\gcd(2w^3, y) = 1$ and that $\gcd(2w^3, y^2) = 1$ as if there did exist a gcd greater than 1 it would have to divide both $2w$ and y which is a contradiction. Therefore we know that the $\gcd(2w^3, y^2 - 3w^2) = 1$ as any divisor greater than 1 would have to divide $2w$ and $y^2 - 3w^2$ which means it would have to divide $2w$ and y^2 which is a contradiction. Therefore in the case that z is even, and w is even with $z = 2w$ and $\gcd(y, z) = 1$ and $1 \leq y < z$

$$m = \frac{1}{2}y|y^2 - 3w^2|, n = w^3 \text{ with } w \text{ even, } \gcd(y, w) = 1, \text{ and } 1 \leq y < 2w.$$

□

Having, for our purposes, exhausted trisectability we move on to a principle part of this study: residuals.

3 Residuals

Suppose that we have a triangle with integer sides and we pick two angles θ and ϕ, it follows by the law of cosines that the cosines of the angles θ and ϕ must be be rationally valued. Furthermore, because these are always two acute angles in a triangle two interior angles to a triangle we can limit the values of θ and ϕ to be $0 < \theta < \phi < \pi/2$. If we take $\cos(\theta) = m/n$ and $\cos(\phi) = p/q$ then we compute the cosine of the final angle like so:

$$\cos(\pi - (\theta - \phi)) = -\cos(\theta + \phi) = sin(\theta)sin(\phi) - \cos(\theta)\cos(\phi)$$

$$= \frac{\sqrt{(n^2 - m^2) * (q^2 - p^2)} - mp}{nq}.$$

For this cosine value to be rational $\sqrt{(n^2 - m^2) * (q^2 - p^2)}$ must be rational, and therefore $(n^2 - m^2) * (q^2 - p^2)$ must be a perfect square. By the fundamental theorem of arithmetic, these two quantities can be expressed as the product of a perfect square and a square free number. We write this as $n^2 - m^2 = j^2 r$ and $q^2 - p^2 = k^2 s$ where r and s are square free. It follows that $(n^2 - m^2) * (q^2 - p^2)$ will be a perfect square when $r = s$. This result is clearly important, as our triangles being integer sided hinges on it, and we will therefore more rigorously define and name these square free numbers.

Definition Let $\theta \in (0, \pi)$ be an angle whose cosine is rational and write $\cos(\theta) = m/n$, where m and n are integers. Express the number $n^2 - m^2$ as $j^2 r$ where j and r are positive integers and r is square free. We call r the residual of the angle θ.

For convenience, we will list a theorem we proved early in the discussion.

Theorem 15. *If θ and ϕ are two angles in an integer-sided triangle then θ and ϕ have the same residual r.*

Because we are motivated to measure the perimeter of a triangle against it's residual, we must prove that a triangle (and therefore it's interior angles) can be characterized by one such residual. The theorem which follows after the lemmas below accomplishes this.

Lemma 8. *If $\cos(\theta)$ is rational (where $\theta \in (0, \pi)$) and the residual of θ is r, then we calculate that $\sin(\theta) = w\sqrt{r}$, where w is a positive rational number. Conversely, if $\cos(\theta)$ is rational and $\sin(\theta) = w\sqrt{r}$ (where r is a square free integer) for some positive rational number w, then the residual of θ is r.*

Proof. By hypothesis suppose that $\cos(\theta) = m/n$ and the residual of θ is r. It follows that $\cos^2(\theta) = m^2/n^2 = 1 - \sin^2(\theta)$ and therefore $\sin(\theta) = \sqrt{1 - m^2/n^2} = \frac{\sqrt{n^2 - m^2}}{n}$. By hypothesis there exists an integer j such that $n^2 - m^2 = j^2 r$. It follows then that $\sin(\theta) = \frac{j}{n}\sqrt{r} = w\sqrt{r}$. Where w is a rational number equal to j/n.

By hypothesis suppose that $\sin(\theta) = w\sqrt{r}$ it follows that $\sin^2(\theta) = w^2 r = 1 - \cos^2(\theta)$ and therefore $\cos^2(\theta) = 1 - w^2 r$. By hypothesis, $\cos(\theta)$ is a rational number, so for integers m and n whose gcd is 1 let $\cos(\theta) = m/n$. It follows that $1 - w^2 r = m^2/n^2$ and therefore $n^2 - n^2 w^2 r = m^2$ and finally $n^2 - m^2 = (nw)^2 r$. By the fundamental theorem of arithmetic the prime factorization of a number is unique, and therefore it's expression into the product of a square free number and a perfect square is unique and we conclude that r must be the residual of θ. □

Lemma 9. *Suppose that θ and ϕ are angles with rational cosines and common residual r. Then the angle $\pi - \theta - \phi$ has residual r.*

Proof. We may assume that $\pi - \theta - \phi$ lies in the interval $(0, \pi)$. It follows from trigonometric identities that

$$\sin(\pi - \theta - \phi) = \sin(\theta + \phi) = \sin(\theta)\cos(\phi) + \cos(\theta)\sin(\phi).$$

By hypothesis there exists m, n, p, and, q such that $\cos(\theta) = m/n$ and $\cos(\phi) = p/q$ and the residual is common and is r. By the previous lemma we can assume that there exist rational numbers w_1 and w_2 such that $\sin(\theta) = w_1\sqrt{r}$ and $\sin(\phi) = w_2\sqrt{r}$. Since

$$\sin(\pi - \theta - \phi) = \sin(\theta + \phi) = \sin(\theta)\cos(\phi) + \cos(\theta)\sin(\phi)$$

it follows that

$$\sin(\pi-\theta-\phi) = w_1 p\sqrt{r}/q + w_2 m\sqrt{r}/ = \frac{(npw_1 + mqw_2)\sqrt{r}}{nq} = \frac{npw_1 + mqw_2}{nq}\sqrt{r}$$

and so by the previous lemma the residual of $\pi - \theta - \phi$ is $\sqrt{r}$. □

Theorem 16. *If θ, ϕ, and γ are the three angles of an integer-sided triangle, then the angles θ, ϕ, and γ have the same residual.*

Proof. by relabeling the angles if necessary we may assume that θ and ϕ are acute. Since the triangle is integer-sided it follows that the cosines of all three angles must be rational. By theorem 15 it follows that θ and ϕ must have the same residual r. By 16 it follows that γ must have the same residual as θ and ϕ as $\gamma = \pi - \theta - \phi$. □

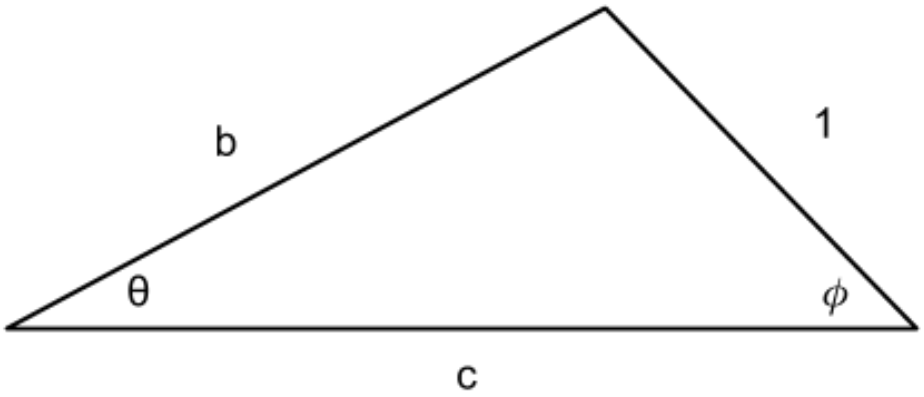
With the conception that our triangles can be represented by a unique residual, we are motivated to work the other way around. If we have two angles with rational cosines and a common residual, does there exist a triangle containing those angles?

Theorem 17. *Suppose that θ and ϕ are angles with rational cosines that satisfy $0 < \theta < \phi < \pi/2$. If θ and ϕ have the same residual, then there exists an integer sided triangle containing the angles θ and ϕ.*

Proof. Suppose that the angles θ and ϕ have the same resiudal. We can write $\cos(\theta) = m/n, n^2 - m^2 = j^2 r, \cos(\phi) = p/q$, and $q^2 - p^2 = k^2 r$ where all of the integers are positive and r is square-free. Consider the triangle with angles θ and ϕ and with sides as indicated in figure 3

Using the law of sines we compute

$$\frac{\sin(\theta)}{1} = \frac{\sin(\phi)}{b} = \frac{\sin(\theta + \phi)}{c}$$



and it follows that

$$b = \frac{\sin(\phi)}{\sin(\theta)} = \frac{n\sqrt{q^2 - p^2}}{q\sqrt{n^2 - m^2}} = \frac{nk}{qj};$$

$$c = \frac{\sin(\theta + \phi)}{\sin(\theta)} = \frac{n}{\sqrt{n^2 - m^2}}\left(\frac{m\sqrt{q^2 - p^2}}{nq} + \frac{p\sqrt{n^2 - m^2}}{nq}\right) = \frac{mk + pj}{qj}.$$

And therefore the triangle $(qj, nk, mk + pj)$ is an integer-sided triangle containing the angle θ and ϕ. □

These past two theorems together give us this final theorem.

Theorem 18. *Acute angles θ and ϕ with rational cosines have the same residual if and only if there exists an integer-sided triangle containing the angles θ and ϕ.*

4 A General Formula for an ISTTA

With all the previous theorems in our arsenal we culminate the paper with this theorem.

Theorem 19. *Let θ and ϕ be trisectible angles with rational cosines that satisfy $0 < \theta < \phi < \pi/2$. Suppose that θ and ϕ have the same residual r and write $\cos(\theta) = m/n, n^2 - m^2 = j^2 r, \cos(\phi) = p/q$, and $p^2 - q^2 = k^2 r$, where all of the integers are positive. Then each of the triangles $(qj, nk, mk + pj)$ and $(qj, nk, mk - pj)$ are ISTTA.*

Proof. The formula $(qj, nk, mk+pj)$ was determined in the proof of theorem 3. By considering the acute angles θ and $\phi - \theta$ and using the same approach, we obtain the formula $(qj, nk, mk - pj)$. Note that this second triangle contains an obtuse angle. □

With the above theorem we are able to generate infinitely many integer sided triangles with trisectible angles. However, proofs about the residuals of these angles and their trisections are needed as a theoretical basis for our investigations. Those theorems and their proofs follow below.

Lemma 10. *Suppose that θ is an angle with a rational cosine value. Then the angle 3θ has a common residual with θ.*

Proof. By the triple angle formula for sine it follows that

$$\sin(3\theta) = 3\sin(\theta) - 4\sin^3(\theta).$$

By lemma 8 it follows that if r is the residual of angle θ there exists a rational number w such that $\sin(\theta) = w\sqrt{r}$. Substituting this into our equation above we see that

$$\sin(3\theta) = 3w\sqrt{r} - 4w^3 r\sqrt{r} = (3w - 4w^3 r)\sqrt{r}.$$

Since w and r are rational numbers it follows that $(3w - 4w^3 r)$ is a rational number and that by lemma 8 3θ has residual r. □

Theorem 20. *Suppose that t is a rational number in the interval $(-1, 1)$. If angles θ and ϕ in the interval $(0, \pi)$ satisfy $\cos\theta = t$ and $\cos\phi = 4t^3 - 3t$, then θ and ϕ have the same residual.*

Proof. By the triple angle formula for cosine, we find that $\cos(3\theta) = \cos(\phi)$. It follows that the angle 3θ is either $\phi, 2\pi + \phi$, or $2\pi - \phi$. As the cosine values are equal it follows that ϕ must have the same residual as 3θ. By lemma 10 it follows that ϕ and θ have the same residual. □

Our final theorem, which concludes the building of the theoretical foundation for the research of this thesis.

Theorem 21. *For each square-free integer r, there exist an infinite number of distinct scalene ISTTAs for which the common residual of the angles is r.*

Proof. Suppose first that r is an odd square-free integer. For each positive integer n, consider the rational number $a_n = |4n^2 - r|/(4n^2 + r)$. Choose a positive integer N such that $N > 2\sqrt{r}$. The sequence $\{a_n\}_{n=N}^{\infty}$ is increasing. Note that if a, b are integers greater than zero where $a > b$ and n is an integer less than both of them, then $2na > 2nb$ and $ab + na - nb - n^2 > ab - na + nb - n^2$ and $(a - n)(b + n) > (b + n)(b - n)$ and finally $\frac{a-n}{a+n} > \frac{b-n}{b+n}$. If we let $a_k = |4k^2 - r|/(4k^2 + r)$ be an element in our sequence $\{a_n\}_{n=N}^{\infty}$ and a_{k+1} be the succeeding element $|4(k + 1)^2 - r|/(4(k + 1)^2 + r)$ we see that $4(k + 1)^2 > 4(k)^2$ and therefore it follows that $a_{k+1} > a_k$ so the sequence must be increasing. Also note that $\sqrt{3}/2 < a < 1$ for all $n \geq N$. Noting further that

$$(4n^2 + r)^2 - (4n^2 - r)^2 = (4n)^2 r$$

and referring to theorem 20, we find that the angle θ_n which satisfies $\cos(\theta_n) = f(a_n)$ is an acute angle that has residual r. Since the sequence is increasing to 1 and the function f is increasing on the interval $(\sqrt{3}/2, 1)$, the sequence $\{\theta_n\}_{n=N}^{\infty}$ is decreasing to 0. From the infinite collection of angles, we can take pairs of θ values and use them (as in theorem 19) to form an infinite number of distinct ISTTAs whose angles have residual r. Using θ_q as our base angle, where $\theta_q < \pi/4$, the angles θ_q and θ_n for $n > q$ generate triangles that have two unequal acute angles and an obtuse angle. Therefore, there exist an infinite number of distinct scalene ISTTAs whose angles have residual r. Now suppose that r is an even square-free integer. For each positive integer n, consider the rational number $b_n = (n^2 r - 1)/(n^2 r + 1)$. Using the same reasoning as above we can confirm that $\{b_n\}$ is an increasing sequence and that $\sqrt{3}/2 < b_n < 1$ for all $n \geq 3$. Noting that

$$(n^2 + 1)^2 - (n^2 r - 1)^2 = (2n)^2 r$$

and referring to theorem 20, we find that the angle ϕ_n which satisfies $\cos(\phi_n) = f(b_n)$ is an acute angle that has residual r. The rest of the proof for this case now follows the lines of the previous paragraph. □

With this theory in our hands we are ready to move into the main investigation of our paper.

5 Goals of This Investigation

As stated above, this paper seeks to investigate the relationship between the perimeters of integer sided triangles with trisectible angles and their residuals. Rather than conduct a theoretical investigation, exploring related theorems etc..., this paper instead seeks to generate and analysis the data in the hopes that it points us in a theoretical direction. Rather than generate the data by hand, it would be far more advantageous for a number of reasons to create a computer program which can do this. A computer program would not only free the user to do other work rather than just computations, but by it's nature it would allow other users to get a copy of the code and tinker with it as they see fit so they too can generate data and conduct investigations. From all this the goals of this investigation may be summed up as the following

1. Design a program which can generate and store the side and angle triples, perimeter, and residual of integer sided triangles with trisectible angles.

2. Design that program be comprehensive, i.e., be able to generate all relevant data of all integer sided triangles with trisectible angles within a given range of perimeters.

3. Design that program to be able to generate all relevant data for integer sided triangles with trisectible angles for, at least, perimeters 1-4,000.

4. Hopefully, design a program which could be able to increase the perimeter range ceiling to 10,000.

5. Meaningfully map the results and analyze.

Professor Gordon had generously provided me with the Maple code which I have provided below (see Figure 9).

6 Implementation

6.1 Gordon's Code

One of the first decisions to make was which language to implement the code in. Maple would be an obvious choice as it would require no translation of Gordon's code into another language. Despite this, the program I would

```
S:={16, 27, 125, 128, 343, 432, 729, 1024, 1331, 2000, 2197, 3375, 3456, 4913, 5488, 6859, 8192,
9261, 11664}:
for P from 1 to 153 do
low := trunc((P+5)/3): high:= trunc((P-1)/2):
for c from low to high do
y:= P-2*c+1: z:=trunc((P-c-1)/2):
for a from y to z do
b:=P-a-c:
r[a]:=(b^2+c^2-a^2)/(2*b*c):
s[a]:=(a^2+c^2-b^2)/(2*a*c):
t[a]:=(b^2+a^2-c^2)/(2*a*b):
if(evalb(denom(r[a]) in S) and evalb(denom(s[a]) in S) and evalb(denom(t[a]) in S) and igcd
(a,b,c) = 1)then print(a,b,c,r[a],s[a],t[a]); end if;
od: od: od:
```

Figure 9: Russel Gordon's Maple Code

design would be implemented in python. Python was chosen not only because I was familiar with it (I was taking a 200 level computer science class in python at the time), but also because it was dynamic (plenty of packages and modules), free (anyone, other students or researchers, could take my code and run it through an IDE in minutes and be free to change it in anyway), and one of the more legible languages to someone with little background in computer science. I also had little familiarity with Maple, and although I was fairly fluent in Matlab, the free and legible nature of python (as well as my taking a class in python) was too alluring to turn down.

Implementing Gordon's code in another language meant first translating what it did into the English language. Such a translation is given below.

1. Store a list of valid cosine denominators for integer sided triangles with trisectible angles in an array S
2. Then for perimeter, P, from 1 through 153
   (a) Set a floor of what a side value c of this triangle could be to be $\lfloor (P+5)/3 \rfloor$
   (b) Set a ceiling of what a side value c of this triangle could be to be $\lfloor (P-1)/2 \rfloor$
   (c) Then for every side value c in the range of $[\lfloor (P+5)/3 \rfloor, \lfloor (P-1)/2 \rfloor]$
      i. Set a floor of what a second side value a of this triangle could be to be $P - (2 * c) - 1$
      ii. Set a ceiling of what a second side value a of this triangle could be to be $\lfloor (P-c-1)/2 \rfloor$

iii. Then for every side value a in the range of

$$[P - (2 * c) - 1, \lfloor (P - c - 1)/2 \rfloor]$$

A. Compute the third side b by setting $b = P - a - c$
B. Compute the cosines of the three angles of the triangle using the law of cosines
C. Print the triangle if the denominator of these cosines are in the array S and if the sides are all relatively prime.

Even with this English translation there are still particulars that need to be sorted out for this program to be properly understood. Particularly the range of the values of the cosines and the sides.

6.2 Particular Ranges of the Program

That we only need a finite number of cosine values to check a finite number of perimeters is not obvious, however, it is true that the maximum value of the denominator of the cosine value can only get as large as $\sqrt[3]{P^2/2}$ where P is the maximum allowed perimeter. To see why this is consider the cosine of any angle in a triangle. By the law of cosines we know that the cosine of this angle must be equal to $(a^2 + b^2 - c^2)/2ab$ where a, b, and c are the necessary sides. By the triangle inequality we know that any one side must be less than the sum of the other two sides, which means that a, at most, could not be more than one half the perimeter. The same bound would be on b as well, which means that a rough bound on the denominator would be $2(P/2)^2$ or $P^2/2$. We introduce the cube because the denominator of our triple angle formula is always either twice an even cube or an odd cube (see 14).

While the availability for limits to the values for our sides may be apparent (triangle inequality, etc...), the exact numbers not be initially scrutable. To see why we have set the limits we have suppose we have a triangle with sides a, b, c where $a < b < c$. It follows by the triangle inequality that $c < a + b$ and therefore

$$2c = c + c < a + b + c = P.$$

However because a, b, and c are all integers with a strictly less than relationship it follows that $a + b + c \leq c - 2 + c - 1 + c = 3c - 3$. Therefore we can

say that $2c < P \leq 3c - 3$ or equivalently that $\frac{P+3}{3} \leq c < \frac{P}{2}$. If we consider the equations $f(P) = \frac{P+3}{3} = P/3 + 1$ and $g(P) = \frac{P+5}{3} = P/3 + 5/3$ we can tell that the $\lceil f(p) \rceil = \lfloor g(P) \rfloor$ as the lines will differ by a constant that is less than one: 2/3rds. This means that we can safely set the lower limit of our greatest side c to be $\lfloor \frac{P+5}{3} \rfloor$. Because we are dealing with integers it follows that $c < \lfloor P/2 \rfloor$ if $c < P/2$. Again, since we are dealing with integer values $c \leq \lfloor P/2 \rfloor - 1$ if $c < \lfloor P/2 \rfloor$. Because $-1 = \lfloor -1/2 \rfloor$ it follows that

$$c \leq \lfloor P/2 \rfloor + \lfloor -1/2 \rfloor = \lfloor P/2 - 1/2 \rfloor = \lfloor (P-1)/2 \rfloor.$$

Now that we have established an upper and lower integer bound for c we can say that $\lfloor \frac{P+5}{3} \rfloor < c \leq \lfloor (P-1)/2 \rfloor$.

To establish the range values for our second side observe that $2a = a + a < a + b = P - c$ and that $a + b < a + c$ so $2a < P - c < a + c$. With some arithmetic we get that $P - 2c < a < \frac{P-c}{2}$ which is equivalent to $P - 2c + 1 \leq a \leq \lfloor (P - c - 1)/2 \rfloor$ for similar reasons as above.

6.3 First Implementation of Modified-Gordon Code

In accordance with the first goal of this investigation, I set out to improve on this framework by automating more tasks to the computer program. Professor Gordon's maple code required a lot of work from the user. The program, for example, would generate a list of triangles with valid cosine denominators only, and require that the numerators still had to be checked by hand at the end of the process to see if they were actually valid triangles. The same was true of residuals as well, and as the reader can see (figure 9) the valid cosine denominators must be put into the program by the user one by one in the array S.

The first task I assigned myself was to create an implementation to generate the denominator values. Knowing that the maximum value I could generate would be $P^2/2$ I could create a simple program which would generate twice even cubes and odd cubes so long as the cubes would be less than $P^2/2$ (see 10).

Now that I had automated a process to create the denominators I set about to design a process to generate and check for correct numerators. From theorem 14 we know that there are three separate numerator cases depending on what kind of denominator we have. It would then be necessary, if we wanted to check the validity of the numerator, to first identify which of

```
trisectableCosineDen = []
maxperimeter = 1000
den = 0
maxnumber = maxperimeter**2/2
while den < maxnumber:
    if den%2 == 1:
     den = den**3
    else:
     den = 2*den**3
    trisectableCosineDen.append(den)
```

Figure 10: Code for generating the cosine denominator fraction list

the three cases the denominator satisfies and then generate a list of possible values for our numerator and then check to see if any of the values satisfy our known value of the numerator. Because we have an equation which describes the denominator in terms of a cube of an integer z which is relatively prime to an integer y in the numerator, I decided the best route was to generate a list of relatively prime integers to z and to let these be the set of possible values for y. Then, picking the numerator equation that corresponds to our denominator, I would plug in z and y and see if it matched the numerator value we were testing. If any of the y's combined with a z in the formula for the numerator to give us the numerator value we were testing for, then the value would be confirmed and the angle would be trisectible. By theorem 11 We would only have to do this for two of the angles in the triangle to know that every angle would be trisectible or not. Should any of them fail, the triangle would be discarded.

To have this algorithm work I needed to design a program to generate all of the numbers relatively prime to a given integer. I developed an algorithm (see Figure 11)which used a GCD function to run through every

```
def GenRelPrime(n):
    i=1
    RelPrimeList = []
    while i < n:
        if gcd(n-i,n) == 1:
            RelPrimeList.append(n-i)
        i=i+1
    return RelPrimeList
```

Figure 11: Code for generating relatively prime sets

integer less than the desired integer and returned the ones whose greatest common denominator was 1. The GCD function itself runs off of the Euclidean Algorithm which is roughly bounded by a linear curve of complexity. This means that creating the list of inputs for the numerator has, at the very worst, a computation complexity $O(n^2)$. Because this is nestled within an $O(n)$ function already, this method of checking would have computation complexity of $O(n^2 + n)$.

R. Gordon suggested that we use the fact that if these cosine values are trisectible, we should be able to use them as the last coefficient in our cubic polynomial and have one of the roots of that cubic polynomial be returned as rational (see theorem 8). One problem is that root general finding algorithms which I could find were all of computational complexity of at least $O(n^2)$. This could be circumvented by the use of the cubic formula. Although it's a gross formula for manual computation, it would be computational complexity $O(1)$ for a computer to compute. However, there is a fundamental problem with this algorithm: *computers cannot return irrational numbers.* An irrational computed in decimal notation would never end but computers necessarily cut off repeating decimals somewhere down the line because they have finite memories. Therefore any answer we got would necessarily be in a finite decimal notation, and would therefore be able to be represented as a ratio of two integers. Although this algorithm would be far cleaner, we bumped into a basic hardware limitation. And although there are computer software programs which can output radicals, the software I was using, python, was not ideal for this.

Because of this limitation it became apparent that creating a list of valid

numerators was necessary. Furthermore, if we were going to be committed to generating numerators, we might as well do so before the main body of the program, and if we have numerators and denominators, we might as well just generate a list of good cosine values in general. This, although nasty sounding, was not that hard to do to bring to implementation (see Figure 12 for the code).

6.4 Analysis of Performance of First Implementation

Armed with the numerator and denominator list I was able to modify R.Gordon's program to generate only the desired triangles. However, this did come with an associated cost: computing the triangles for perimeters $1 - 1,000$ took about five minutes. The three for loops (Perimeter p, Side c, Side a) give us a computational complexity of $O(n^3)$. Within the deepest shell of those for loops we call to see if our cosine value is in the list which is an order $O(n)$ operation, which brings us to a total computational complexity of $O(n^4)$. Therefore to compute the triangles for perimeters $1 - 4,000$ would take 256 times longer. Although I added on a 'write' function so that this kind of calculation need only be made once, (the results would be saved to a text file) getting through the first $4,000$ perimeters (one of our primary goals) would take about 21 and a half hours on single terminal, and getting to the ideal goal of $10,000$ perimeters would take about 35 days. The program needed to be improved.

6.5 Second Implementation

One way to shave off some time would be to improve the way we generated cosine values at the beginning of the program. Rather than using theorem 14 to generate our fractions, we can use theorem 13 by plugging in all the rational numbers from zero to $n - 1/n$ where n is the maximum value our denominator can take on according to the maximum perimeter we set. This is cheaper than generating sets of relatively prime integers because arithmetic, unlike the gcd function, is of computational complexity $O(1)$. Although it would have some repeated values, one simple way is to run two nested loops, with the outer one incrementing the denominator from 0 to n (where n is the maximum value our denominator can take on according to the maximum perimeter we set), and the inner one incrementing the numerator from 0 to the denominator value and then plugging in the resulting fraction each

```
while True:
    n=(2*i)
    denominator = 2*((n)**3)
    denominators.append(denominator)
    Yvalues = GenRelPrime(2*n)
    while True:
        y = Yvalues.pop()
        numerator = y*abs((y**2)-3*(n**2))
        ComDiv = gcd(numerator,denominator)
        numerator = numerator/ComDiv
        denominator = denominator/ComDiv
        TrisectableCosineValues.append((numerator,denominator))
        if not Yvalues:
            break
    i=i+1
    if int(denominator) > int(maximum):
        break
while True:
    n = (2*j)+1
    denominator = (n)**3
    denominators.append(denominator)
    Yvalues = GenRelPrime(n)
    while True:
        y = Yvalues.pop()
        if y%2 == 1:
            numerator = int(.5*(y*abs(y**2-3*n**2)))
            ComDiv = gcd(numerator,denominator)
            numerator = numerator/ComDiv
            denominator = denominator/ComDiv
            TrisectableCosineValues.append((numerator,denominator))
        else:
            numerator = y*abs(4*(y**2)-3*(n**2))
            ComDiv = gcd(numerator,denominator)
            numerator = numerator/ComDiv
            denominator = denominator/ComDiv
            TrisectableCosineValues.append((numerator,denominator))
        if not Yvalues:
            break
    j=j+1
    if denominator > maximum:
        break
```


Figure 12: Code to generate the cosine values of trisectible angles

```
fractionlist = []
for i in range(0,200):
    for j in range(1,200):
        num = i
        den = j
        gcdnd = gcd(num,den)
        num = num/gcdnd
        den = den/gcdnd
        fractionlist.append((num,den))
```

Figure 13: A better way to generate valid cosines

time into the equation from 13 and storing the result. The code for this implementation is given below (see Figure 13).

However, this still left us with the problem that we had a main program which was running at a computational complexity of $O(n^4)$. In searching for a way to remove the duplicates out of the list of cosine values, so that searching through the list would take less time, I stumbled across an extremely helpful data-type: sets. A set is a data type which has unordered collections of unique elements accessed by a hashing function. A gentle introduction into data types, and hashing is included in the appendix, but the big takeaway is that a set will not only not allow for duplicates, but it also has a membership look-up function of computational complexity $O(1)$.

Making the trisectible cosine values ordered pairs (numerator,denominator) of objects in a set meant that searching for cosine values in that set would be a constant time operation regardless of how many cosine values there were to search through. This also meant reducing the computational complexity of my program from $O(n^4)$ to $O(n^3)$.

In terms of real world gains, this translated to being able to calculate 4,000 perimeters in a half hour and by splitting the work amongst separate programs 10,000 perimeters were computed in a little over three hours.This made it possible to hit desired objectives 1-5 in a reasonable amount of time. The only part left to program was finding the residual of each valid triangle, storing that data, and then mapping it against the perimeters.

6.6 Residuals

Due to theorem 16 it is necessary to compute only one angles residual in the triangle. To compute the residuals of these triangles we needed to create a prime factorization of the difference of the numerator squared and the denominator squared, and then create an integer by multiplying together every integer which had an odd power representation in the prime factorization. In terms of implementation this meant having a function to create a prime factorization.

Creating a prime factorization function meant taking an integer and returning a list of pairs, a prime and the power it is raised to. To do this the prime factorization function was segmented into a part that would factor the function into a list of primes, and a function that would condense that list into a pair with a count. The code for these functions is displayed below (see Figures 14, and 15).

Taking one of our cosine values for each valid triangle, we apply the factor and condense function to that cosine value to get a list of tuples of it's prime factors and their corresponding powers. Once we have this list we can easily create another list from it which only contains the factors with odd powers. Then, creating a product function, (see Figure 16), we multiply all the primes in that list together to get the square free residual for that cosine. The code all together is given below (see Figure 17)

This gave me all the data I needed to pursue the main question of this paper

7 Results

Below is a plot of the perimeters against the residuals of all triangles with relatively prime integer sides and trisectible angles for perimeters 1-10,000 that was generated by my code (see Figure 18). In Figure 18 each red circle represents such a triangle, and the coordinate pair of each red circle represents the (perimeter,residual) pairing for that triangle. The two blue circles are coordinates of data cluster centers I implemented with a k-means algorithm. The details of this algorithm and it's findings will be in the appendix and data analyzing sections of this paper respectively. I also put together a tracker of the total number of residuals and have included the text file which recorded the sides, cosines, perimeter, and residual, of each triangle in one line of text.

```
def factorize(n):
    def isPrime(n):
     return not [x for x in xrange(2,int(math.sqrt(n)))
                 if n%x == 0]
    primes = []
    candidates = xrange(2,n+1)
    candidate = 2
    while not primes and candidate in candidates:
     if n%candidate == 0 and isPrime(candidate):
         primes = primes + [candidate] + factorize(n/candidate)
     candidate += 1
    return primes
```

Figure 14: Factoring function for taking an integer and outputting a list of factors

```
def condense(L):
  prime,count,list=0,0,[]
  for x in L:
    if x == prime:
      count = count + 1
    else:
      if prime != 0:
     list.append((prime,count))
      prime,count=x,1
  list.append((prime,count))
  return list
```

Figure 15: Condense function for putting a list of factors into a list of pairs (prime, power)

```
def prod(listdata):
    if len(listdata) == 1:
        return listdata[0]
    else:
        return listdata[0]*prod(listdata[1:])
```

Figure 16: Function which takes a list of integers and returns the product of those integers as an integer

7.1 Analysis

Although the highest residuals do occur at the end of the data, it's also clear with a closer look at the perimeter interval 9,000-10,000 (see Figure 19) we see that these residuals are preceded closely behind by a string of 6 triangles with residual under 100, and one triangle between the highest two with residual less than fifty.

Similar results are apparent for the other triangle with residual over 600 which occurs in the perimeter range 7,500-9,000 (see Figure 20). It is followed, not by more higher residual triangles as our hypothesis would predict, but instead by a collection of four triangles with residuals all under 50.

Looking back at the data for all the triangles, it becomes apparent that regardless of how large our perimeter is residuals under 100 seem to dominate. In total, there are only 11 triangles with residual greater than 100. However, it may also be the case that this general residual bound (100) may be increasing over time. In order to test this hypothesis I implemented a data clustering algorithm called the k-means algorithm.

```
def prod(listdata):
    if len(listdata) == 1:
        return listdata[0]
    else:
        return listdata[0]*prod(listdata[1:])

def factorize(n):
    def isPrime(n):
        return not [x for x in xrange(2,int(math.sqrt(n)))
                    if n%x == 0]
    primes = []
    candidates = xrange(2,n+1)
    candidate = 2
    while not primes and candidate in candidates:
        if n%candidate == 0 and isPrime(candidate):
            primes = primes + [candidate] + factorize(n/candidate)
        candidate += 1
    return primes

def condense(L):
  prime,count,list=0,0,[]
  for x in L:
    if x == prime:
      count = count + 1
    else:
      if prime != 0:
        list.append((prime,count))
      prime,count=x,1
  list.append((prime,count))
  return list


savetriangles = list(triangles)

while True:
    triangledata = triangles.pop()
    temp = []
    cosvalue1 = triangledata[3]
    cosvalue2 = triangledata[4]
    cosvalue3 = triangledata[5]
    temp.append(abs(cosvalue1[0]**2-cosvalue1[1]**2))
    temp.append(abs(cosvalue2[0]**2-cosvalue2[1]**2))
    temp.append(abs(cosvalue3[0]**2-cosvalue3[1]**2))
    leastval = min(temp)
    leastval = condense(factorize(leastval))
    squarefree = []
    while True:
        testforsquare = leastval.pop()
        testforsquarevalue = testforsquare[0]
        testforsquare = testforsquare[1]
        testforsquare = testforsquare%2
        if testforsquare !=0:
            squarefree.append(testforsquarevalue)
        if not leastval:
            break
    if squarefree:
        residuals.append(prod(squarefree))
    if not triangles:
        break
residuals.reverse()
print (len(residuals))
print(len(savetriangles))
for i in range(len(residuals)):
    f.write(str(savetriangles[i]) + ' ')
    f.write(str(residuals[i])+"\n")


f.write("----END OF RUN-----")
f.close
```

Figure 17: Segment of program for computing residuals

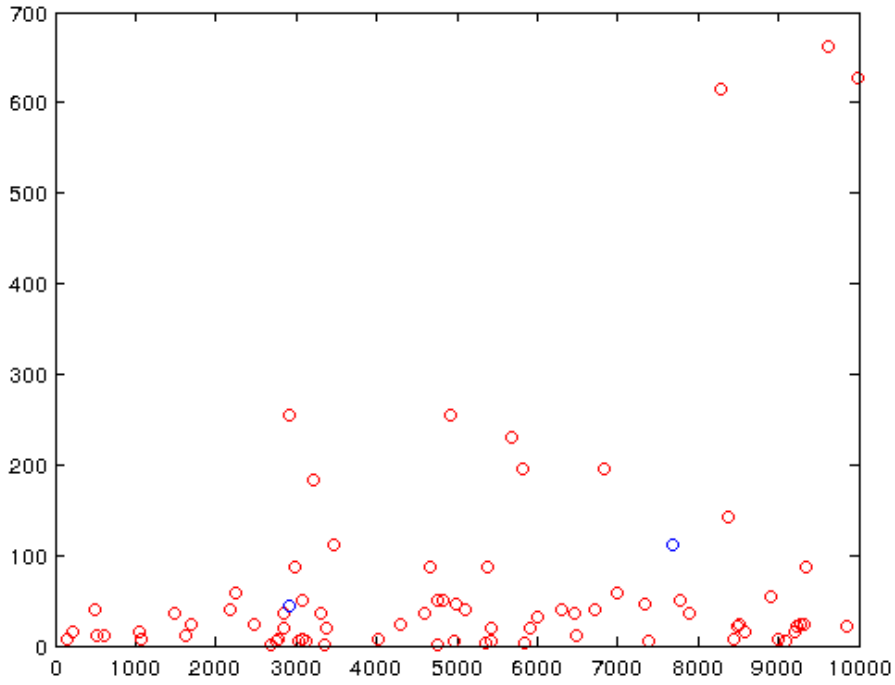


Figure 18: Perimeter vs. Residual for triangles with relatively prime integer sides and trisectible angles

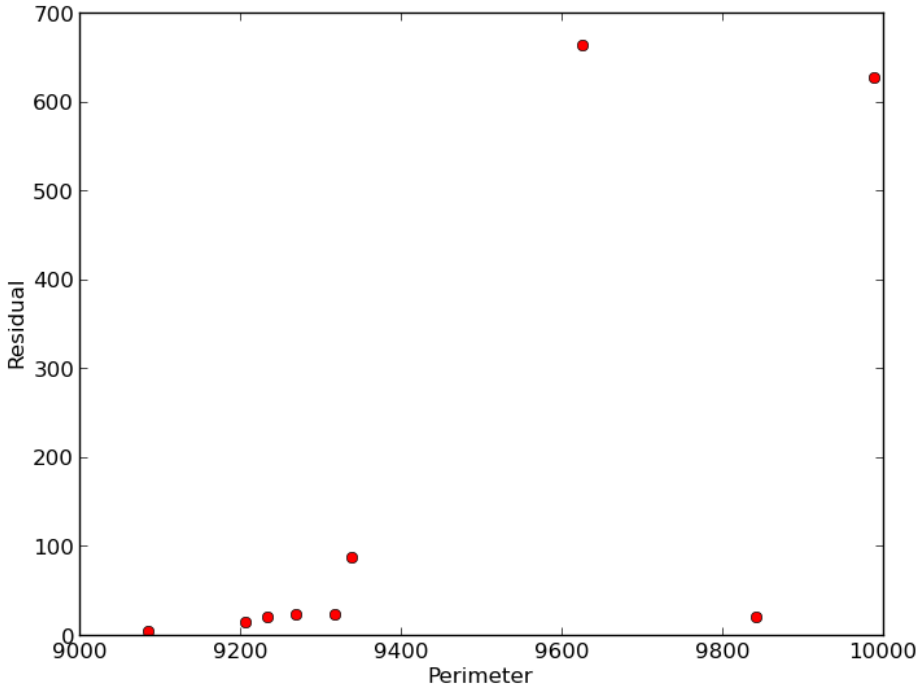


Figure 19: Perimeter vs. Residual for triangles with relatively prime integer sides and trisectible angles and perimeters from 9-10 thousand

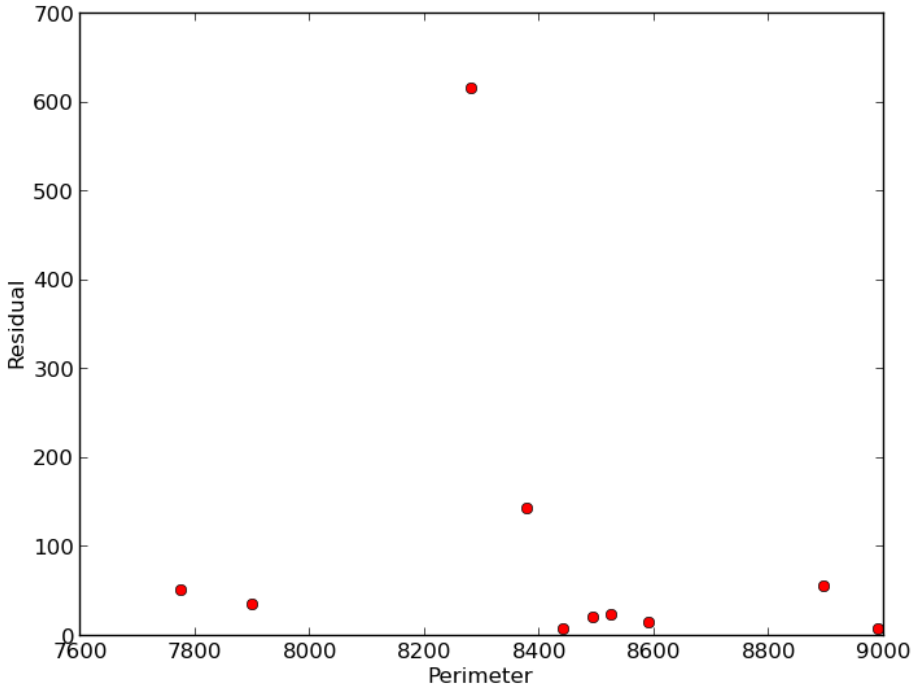


Figure 20: Perimeter vs. Residual for triangles with relatively prime integer sides and trisectible angles and perimeters from 7.5-9 thousand

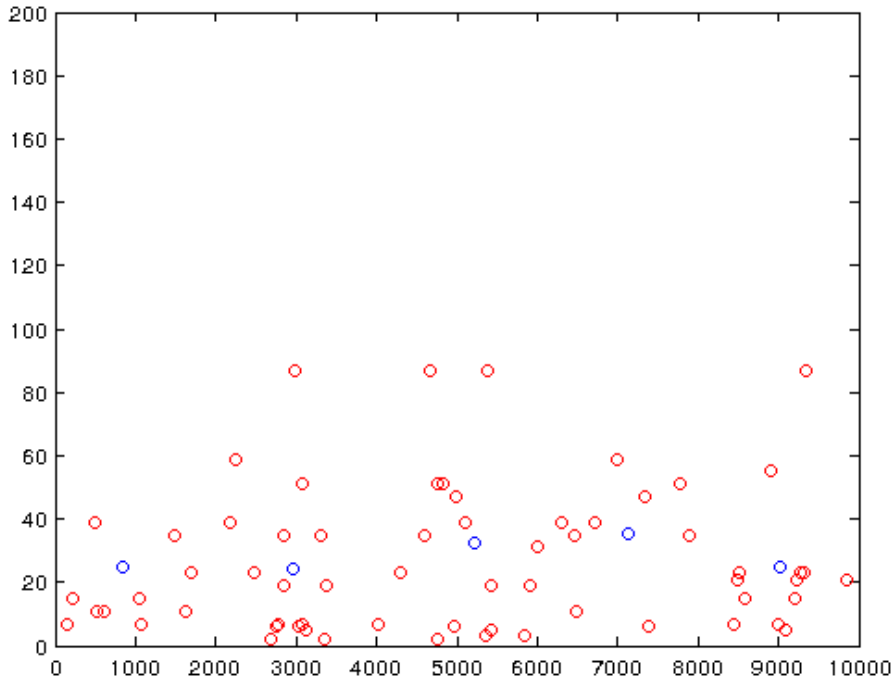


Figure 21: A typical run of the k-means algorithm on the data set

The goal of a k-means algorithm is to partition data into k clusters where each data point belongs to the cluster closest to it. By implementing k-means on this data set I hoped to see if there was a general rising trend with the data.

Upon implementing the code (with the help of D. Hundley) on the entire data set I realized that keeping the extreme residual values (the ones over 100) was skewering these centers. In order to answer the question of what is happening to the data points for residual less than one hundred, I needed to fix the data by removing all residuals greater than one hundred.

Doing so, and then implementing the k-means algorithm with five centers gave me the following result (see Figure 21).

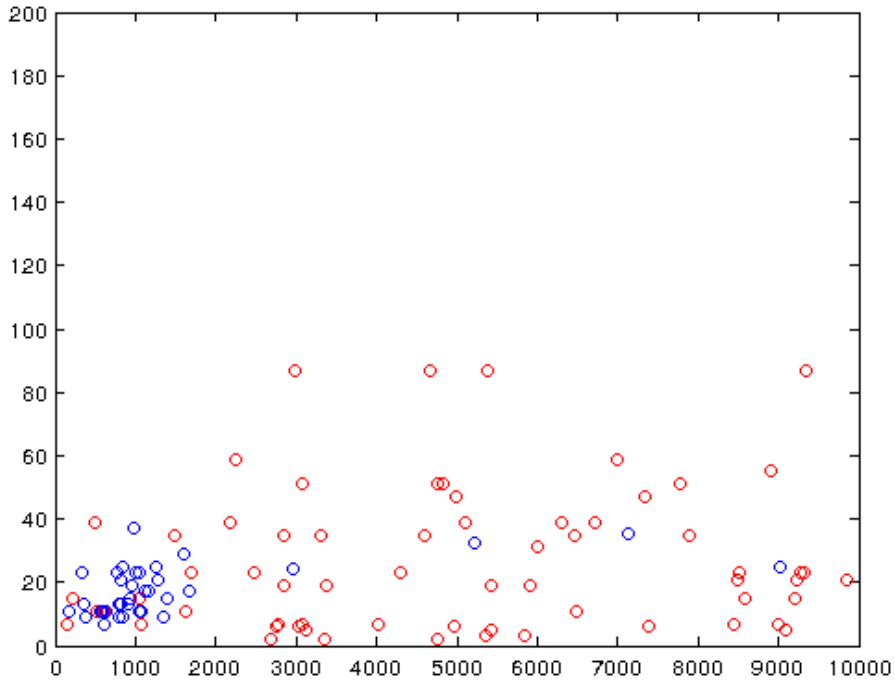


Figure 22: Twenty runs of the k-means algorithm on the data set with the centers recorded each time

Although this looks like a good result, it's important to test the algorithm and see that the program is converging to these centers. To do so I set up a program to run the k-means algorithm 20 times and record all of the centers. The result is recorded in the figure below (see Figure **??**).

The last four are stable, but the earliest cluster center is a little more unpredictable. However, we are confident that it is bound in that general region. If we add a linear trend line to a typical run we are able to see that there appears to be a slight relationship between the size of the perimeter and the residual (see Figure 23).

One major drawback of the k-means algorithm is that the user has to set the number of data cluster centers before the program runs. This means that

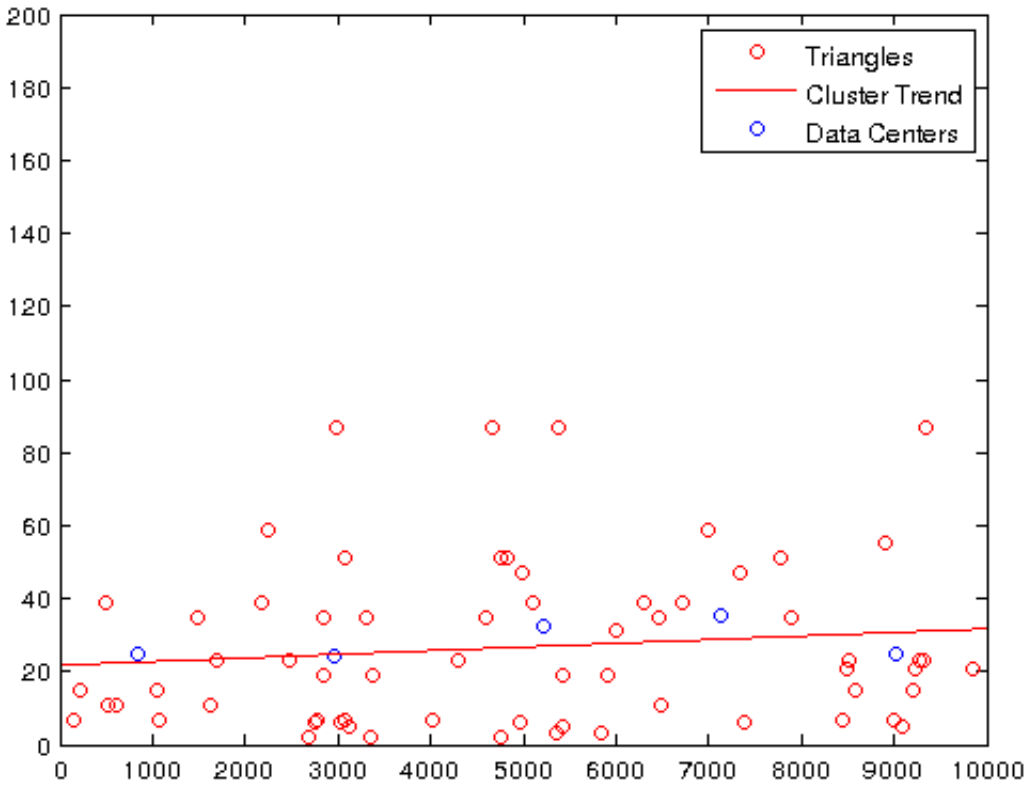


Figure 23: A typical run of the k-means algorithm with a trend line on the data clusters

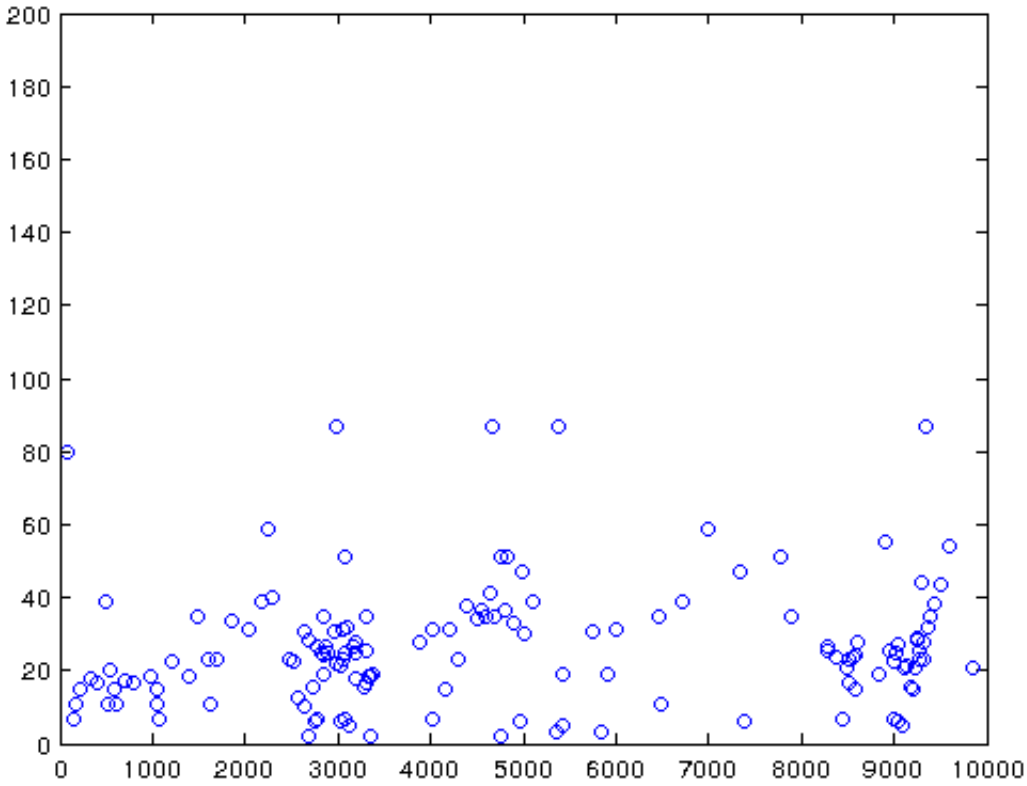


Figure 24: 100 iterations of the LBG algorithm on the data set. Cluster centers only displayed.

there may be a more representative partitioning of the data which doesn't get produced simply because there aren't enough cluster centers. In order to overcome this drawback I designed and implemented (with the help of D. Hundley) a Linde-Buzo-Gray algorithm. The LBG algorithm is identical to the k-means algorithm, except that it will automatically delete the least well partitioning cluster centers (as measured by the average distance of the points in the cluster to the cluster center) and create two new cluster centers in it's place. The process of pruning and generating data cluster centers continues until the total error of all the centers (mean distance of points in cluster to cluster center) goes below a preset level by the user. In order to ensure that the algorithm doesn't just return a data cluster center for each point I implemented a pruning tool whereby if there were more than 10 total cluster centers only the best ten would survive into the next iteration. Testing the LBG algorithm in a manner similar to k-means gave somewhat disappointing results. Although the total error from the number of centers was less, the cluster centers seemed to converge far less (see Figure 24)

8 Conclusions and Questions for Further Study

Our initial hypothesis that the residuals of our triangles would be proportional to our perimeters seems only partly true. In general, it seems that most triangles with trisect able angles and integer sides fit on a rising trendline, however it is certainly not the case that just because triangle B has larger perimeter than triangle A that triangle B has a larger residual than triangle A.

This study leaves open many avenues for exploration. In terms of implementation we can look for ways to reduce the computational complexity of our program to generate more data, as well as trying an implementation that is not comprehensive like ours was. Similarly, with more data, different kinds of analysis can be run on the triangles and their residuals and different iterations of the LBG or k-means algorithm can be run.

Answering questions about the density of these triangles, how to find the triangle with the least perimeter for a given residual, and how the fractions which generate these triangles relate to the residuals of the triangles could

all be fruitful paths as well.

Appendices

Abstract Data Type

An abstract data type, sometimes abbreviated ADT, is a logical description of how we view the data and the operations that are allowed without regard to how they will be implemented. This means that we are concerned only with what the data is representing and not with how it will eventually be constructed. By providing this level of abstraction, we are creating an encapsulation around the data. The idea is that by encapsulating the details of the implementation, we are hiding them from the user's view. This is called information hiding.

Computational Complexity

When trying to characterize an algorithm's efficiency in terms of execution time, independent of any particular program or computer, it is important to quantify the number of operations or steps that the algorithm will require. If each of these steps is considered to be a basic unit of computation, then the execution time for an algorithm can be expressed as the number of steps required to solve the problem. Deciding on an appropriate basic unit of computation can be a complicated problem and will depend on how the algorithm is implemented.

A good basic unit of computation for comparing the summation algorithms shown earlier might be to count the number of assignment statements performed to compute the sum. In a summation function, it makes sense to use the number of terms in the summation to denote the size of the problem. We can then say that the sum of the first 100,000 integers is a bigger instance of the summation problem than the sum of the first 1,000. Because of this, it might seem reasonable that the time required to solve the larger case would be greater than for the smaller case. Our goal then is to show how the algorithm's execution time changes with respect to the size of the problem.

Computer scientists prefer to take this analysis technique one step further. It turns out that the exact number of operations is not as important as determining the most dominant part of the function. In other words, as the problem gets larger, some portion of the function tends to overpower the rest. This dominant term is what, in the end, is used for comparison. The order of magnitude function describes the part of the function that

f(n)	Name
1	Constant
$\log n$	Logarithmic
n	Linear
$n \log n$	Log Linear
n^2	Quadratic
n^3	Cubic
2^n	Exponential

Figure 25: Common Functions for Big O notation

increases the fastest as the value of n increases. Order of magnitude is often called Big-O notation (for "order") and written as $O(f(n))$. It provides a useful approximation to the actual number of steps in the computation. The function f(n) provides a simple representation of the dominant part of the original function.

Although we do not see this in the summation example, sometimes the performance of an algorithm depends on the exact values of the data rather than simply the size of the problem. For these kinds of algorithms we need to characterize their performance in terms of best case, worst case, or average case performance. The worst case performance refers to a particular data set where the algorithm performs especially poorly. Whereas a different data set for the exact same algorithm might have extraordinarily good performance. However, in most cases the algorithm performs somewhere in between these two extremes (average case). A number of very common order of magnitude functions are shown in Figure 25

Data Type

The implementation of an abstract data type is often referred to as a data structure and will require that we provide a physical view of the data using some collection of programming constructs and primitive data types.

Hashing Function

A hash function is a way of mapping an object to a place in memory

allocation. It uses properties about the object as numbers and a function to compute the address. Data types which use hashing functions to store their objects have membership functions of roughly constant order, as accessing an object is as easy as inputting that object into a function.

K-means Algorithm

A k-means algorithm partitions data into k clusters where each data point belongs to the cluster center closest to it. At the beginning of the k-means algorithm a number of cluster centers are input by the user into the algorithm. The algorithm will then partition the data according to these centers, evaluate the mean distance of the points in each cluster to the center, and then generate new cluster centers for the partitions which are performing the worst. The process repeats again (partitioning and evaluating the partition) until the total error (the mean distance of the points in each cluster to the center) goes below a certain level, or until the centers stabilize. Stabilization is checked for by keeping a copy of the old centers and checking if the new and old centers match.

Linde-Buzo-Gray Algorithm

An algorithm to partition data into a variable amount of clusters. It runs just like the k-means algorithm but allows for automated pruning and generating of cluster centers.

Ordered and Unordered Data Type

When a data type is described as ordered or unordered, we are referencing whether or not the relative position of each object to each other is stored. A list is an example of ordered data type: there is a first element, a second element, a third element, etc..., and the position of each element in the list references another position. A set is an example of an unordered data type as the placement of each object is recorded by a hashing function not by it's placement relative to another object.

Primitive Data Type

A primitive data type is a data type that is provided by a programming language as an elementary building block. Most languages allow more complicated composite data types to be constructed from these primitive data types.

References

[1] Brad Miller and David Ranum, *Problem Solving with Algorithms and Data Structures using Python*, Runestone Interactive, 2014.

[2] Daniel Rosenthal, David Rosenthal, and Peter Rosenthal, *A Readable Introduction to Real Mathematics*, Springer, 2014.

[3] Euclid and Thomas L. Heath, *The Thirteen Books of the Elements, Vol. 1: Books 1-2*, Dover Publications, London, 1956.

[4] J. Gallian, *Contemporary Abstract Algebra, 5th ed.*, Houghton Mifflin, 2002.

[5] Russel A. Gordon, *Integer Sided Triangles with trisectible Angles*, Mathematics Magazine (2015), 198–211.

[6] *Python Software Foundation [US]* (2015), `https://docs.python.org/3/`. Accessed May 13th 2015.