IMPLEMENTATION OF ALGORITHMS RELATED TO CONVEX HULLS AND TROPICAL VARIETIES FOR VISUALIZATION

A Thesis

by

EMMA OWUSU KWAAKWAH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2016

Major Subject: Mathematics

ABSTRACT

The study of the zero sets of polynomials is one of the main themes in Algebraic Geometry. In the study of the zero sets, the Archimedean tropical variety can be a helpful tool in approximating the norms of the zeros of a given polynomial.

The main goal of this thesis is to implement the algorithm for graphically constructing the Archimedean tropical variety of polynomials in two variables (bivariate polynomials) using Sage, a free open-source mathematics software. In the process of graphically constructing the Archimedean tropical variety of bivariate polynomials, the convex hull of points in 3D will be required. This will lead us to write a Sage function based on methods and functions in Sage that will return the convex hull of points in 3D in Sage.

DEDICATION

I dedicate this work to my parents, Mr. Benjamin Owusu Kwaakwah and the late Mrs. Joyce Owusu Kwaakwah.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. J. Maurice Rojas, without whose support and guidance, this work would not have been possible.

To my "personal person", Ebenezer Poku Dwobeng, I appreciate all the support and the intelligent discussions you had with me about the convex hull problem. I'm always thankful for you. Also, my deepest appreciation to my entire family for their prayers, support and encoragement.

To Dr. Timo de Wolff and Dr. Chris O'Neill, thank you for bringing me aboard the Patchworking project, it helped hone my Sage skills. Also, thank you for helping me get my Sage files back when I lost them and for pointing me to some of the Sage functions used in this work.

Dr. Kaitlyn Phillipson, my defense buddy, your help during my proposal submittal stage and my defense preparation is really appreciated.

And finally, to my committee members, it was an honour to have you serve on my committee.

NOMENCLATURE

2D	The two-dimensional space
3D	The three-dimensional space
ArchTrop(f)	The Archimedean Tropical Variety of $f(x)$
Archnewt(f)	The Archimedean Newton Polytope of $f(x)$
$\mathbb{C}$	The field or complex numbers
conv	The convex hull
Newt(f)	The Newton Polytope of $f(x)$
$\mathbb{Q}$	The field of rational numbers
$\mathbb{R}$	The field of real numbers
Sage	Software for Algebra and Geometry
Z(f)	The zero set of $f(x)$

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iii

ACKNOWLEDGEMENTS iv

NOMENCLATURE v

TABLE OF CONTENTS vi

LIST OF FIGURES viii

1. INTRODUCTION 1

   1.1 Definition of Some Key Terms 2
   1.2 Problem Statement 3

2. BACKGROUND ON ALGEBRAIC AND TROPICAL GEOMETRY . . . 5

   2.1 The Relation between the Newton Polygon, Amoeba and the ArchTrop of a Polynomial 5
   2.2 Differences Between the ArchTrop and Amoeba of a Polynomial . . . 8
      2.2.1 Some Properties of the ArchTrop of a Polynomial 8
      2.2.2 Some Properties of the Amoeba of a Polynomial 8

3. CONVEX HULL OF POINTS IN THREE-DIMESIONAL SPACE USING SAGE AND THE ARCHTROP OF A GIVEN BIVARIATE POLYNOMIAL 10

   3.1 Steps to Derive the Convex Hull of Points in 3D in Sage 10
   3.2 The ArchTrop Algorithm for a Given Bivariate Polynomial 12
   3.3 Implementation of the ArchTrop Algorithm for a Bivariate Polynomial in Sage 13
   3.4 Conclusion 20

4. CONCLUSIONS 21

REFERENCES 22

APPENDIX A. 3D CONVEX HULL FUNCTION AND OTHER AUXILLIARY FUNCTIONS IN SAGE 23

APPENDIX B. THE ARCHTROP FUNCTION 25

LIST OF FIGURES

FIGURE Page

2.1 The Newton polygon of $1+x_1+x_1^2+x_1^3+x_1^2x_2^3+10x_1x_2+12x_1^2x_2+10x_1^2x_2^2$ 6

2.2 The Amoeba of $1+x_1+x_1^2+x_1^3+x_1^2x_2^3+10x_1x_2+12x_1^2x_2+10x_1^2x_2^2$. 6

2.3 The ArchTrop of $1+x_1+x_1^2+x_1^3+x_1^2x_2^3+10x_1x_2+12x_1^2x_2+10x_1^2x_2^2$ 7

3.1 A plot of the points (0,0,0), (1,1,1), (0,1,0), (1,0,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (2,2,2) in 3D 11

3.2 A plot of the convex hull of the points (0,0,0), (1,1,1), (0,1,0), (1,0,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (2,2,2) 12

3.3 The vertices of the Archnewt of $1+x^3+y^2-10x_1x_2$ 15

3.4 The vertices of the ArchTrop of $1+x^3+y^2-10x_1x_2$ 16

3.5 The vertices and edges of the ArchTrop of $1+x^3+y^2-10x_1x_2$ . . . 16

3.6 The ArchTrop of $1+x^3+y^2-10x_1x_2$ 17

3.7 The Amoeba of $1+x^3+y^2-10x_1x_2$ 18

3.8 The ArchTrop of $100+x+y+xy+100x^2+y^2$ 19

3.9 The Amoeba of $100+x+y+xy+100x^2+y^2$ 20

1. INTRODUCTION

The study of the roots of polynomials is one of the main themes in Algebraic Geometry. Given a polynomial, $f(x) \in \mathbb{R}[x]$ some of the questions that can be asked about it are:

- does $f(x)$ have solutions in $\mathbb{R}$?
- how many solutions does $f(x)$ have in $\mathbb{Q}$?
- what are the roots of $f(x)$ in $\mathbb{R}$?

The example below is considered in light of the questions above.

Example 1. *Consider the polynomial*

$$f(x) = x^3 - 2x^2 - 5x + 6$$

To answer the first question for $f(x)$, we can use Descartes' Rule of Signs which gives a bound on the number of real roots a polynomial can have. In our example, $f(x)$ has two sign changes so this means that it has either 0 or 2 positive real roots. Also, $f(-x) = -x^3 - 2x^2 + 5x + 6$ has only one sign change so that implies that $f(x)$ has 1 negative real root. So, by the the above-mentioned rule, we know that $f(x)$ has at least one real root. For the second question, we can use the Rational Root Test. The factors of the constant term of $f(x)$ are $\pm 1, \pm 2$ and ± 3 while the only factors of the leading coefficient of $f(x)$ are ± 1. So, by the rational root test, the choices of values we can test are $x = \pm 1, x = \pm 2$ and $x = \pm 3$. Of these choices, we clearly see that only $x = 1, x = -2$ and $x = 3$ satisfy $f(x) = 0$. Hence, the answer for the third question can easily be given as $x = 1, x = -2$ or $x = 3$.

In the above example, we see that finding the exact roots of a polynomial in one variable(univariate polynomial) can sometimes be easy since there are a lot of algebraic techniques that can be used. As the number of variables increase, the difficulty in finding the exact roots may increase and that makes approximating roots of polynomials the next best option.

1.1 Definition of Some Key Terms

Before stating the problem we seek to answer, let us state the following definitions.

Definition 1. *A set $X \in \mathbb{R}^n$ is said to be a convex set if for any two points, $x_1, x_2 \in X$, $\lambda x_1 + (1-\lambda)x_2 \in X$ for each $\lambda \in [0,1]$.*

Definition 2. *Given the polynomial*

$$f(x) \quad = \quad \sum_{i=1}^{t} c_i x^{a_i}$$

where $c_i \neq 0 \, \forall i$ we define the Support, Newt, Archnewt and Amoeba of f as:

$$\begin{aligned} \mathit{Support}(f) &:= \{(a_i)\} \\ \mathit{Newt}(f) &:= \mathit{conv}(\{(a_i)\}_{i\in[t]}) \\ \mathit{Archnewt}(f) &:= \mathit{conv}(\{(a_i, -\log|c_i|)\}) \\ \mathit{Amoeba}(f) &:= \{(\log|z_1|, \log|z_2| \cdots, \log|z_n|) : (z_1, z_2, \cdots, z_n) \in Z(f)\} \end{aligned}$$

In addition, we give two equivalent definitions of ArchTrop(f) below;

$$\begin{aligned} \mathit{ArchTrop}(f) := \{w \in \mathbb{R}^n | (w, -1) & \text{ is an outer normal of a face of } \mathit{Archnewt}(f) \\ & \text{of positive dimension}\} \end{aligned}$$

$$\mathit{ArchTrop}(f) := \{w : \mathit{max}_{i\in\{1,\cdots,t\}} |c_i e^{wa_i}| \text{is attained for at least two distinct } i\}$$

In the next section, we will discuss the above-defined terms and their relationships with one another.

1.2 Problem Statement

Given a polynomial in two variables(bivariate polynomial), one can solve for the roots using techniques from Numerical Algebraic Geometry. Sometimes, getting the exact roots of such a polynmial can be cumbersome and so finding approximate roots will be the best alternative. The problem we seek to answer in this thesis is to graphically approximate the zeros of bivariate polynomials with a focus on graphically obtaining their ArchTrops. This is important because traditional algebraic methods are too slow for convenient visualization, but the tropical approximation methods we use are probably much faster. Also, we know that if we have a graphical representation of the ArchTrop of a polynomial $f(x)$, there is a formula that can be used to approximate the norms of the zeros of $f(x)$ and vice versa [3] and furthermore, approximate roots can be refined at a later stage via Newton's method.

The convex hull of points in 3D is needed so that finding the vertices, edges and rays of the ArchTrop will be easy. So, as part of this thesis, instead of implementing an algorithm to find the convex hull of points in 3D in Sage, we will write a Sage function that is based on functions and methods in Sage that will return the convex hull of points in 3D.

This thesis will begin with the problem we seek to answer and some definitions, its second section will discuss some background on Algebraic and Tropical Geometry, the relation between the Newton polygon, Amoeba and ArchTrop of a polynomial. Also, the properties of the two latter objects will be discussed as well. In the final section, we will discuss how to get the convex hull of points in 3D in Sage and also how we get the ArchTrop of a given bivariate polynomial. In the appendices will be the Sage functions that return convex hull of points in 3D and the ArchTrop of a given bivariate polynomial respectively.

2. BACKGROUND ON ALGEBRAIC AND TROPICAL GEOMETRY

Algebraic Geometry is the study of the solutions of algebraic equations, but it is interesting to know that norms of the zeros of polynomials can be estimated using polyhedral geometry. Isaac Newton was the first to use polygons to determine the series expansion of univariate polynomials over the Puiseux field [5].

2.1 The Relation between the Newton Polygon, Amoeba and the ArchTrop of a Polynomial

Since 1676 when Isaac Newton first came up with the relation between polygons and series expansions of polynomials, Newton's work has been extended over other fields [3] and work on the relatonship between polyhedral geometry and estimation of the norms of the solutions of polynomials has also progressed over the centuries. The Newton polygon of the polynomial f mentioned in the previous section defines the geometry of the polynomial and is related to its Amoeba.

Consider the polynomial

$$f(x) = 1 + x_1 + x_1^2 + x_1^3 + x_1^2x_2^3 + 10x_1x_2 + 12x_1^2x_2 + 10x_1^2x_2^2$$

The Newton polygon of this polynomial is the convex hull of the points $(0,0), (1,0)$, $(2,0), (3,0), (2,3), (1,1), (2,1)$ and $(2,2)$ which is the triangle in Figure 2.1 with vertices $(0,0), (3,0)$ and $(2,3)$.

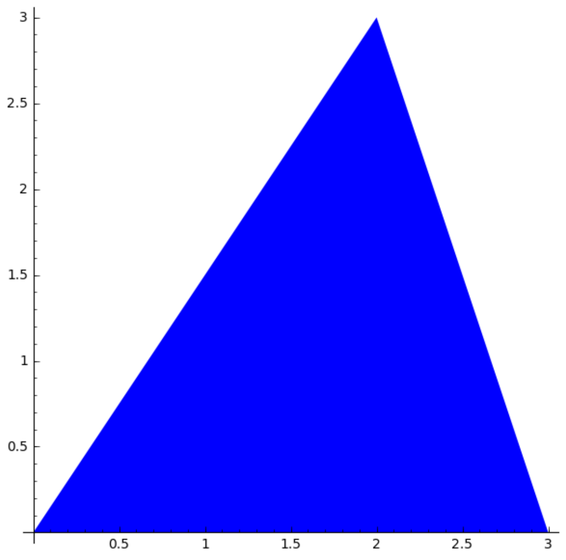

Figure 2.1: The Newton polygon of $1+x_1+x_1^2+x_1^3+x_1^2x_2^3+10x_1x_2+12x_1^2x_2+10x_1^2x_2^2$

Observe that the outernormals to the sides of the of the Newton polygon of f are in the same direction as the tentacles of Amoeba(f) in [1] pictured in Figure 2.2

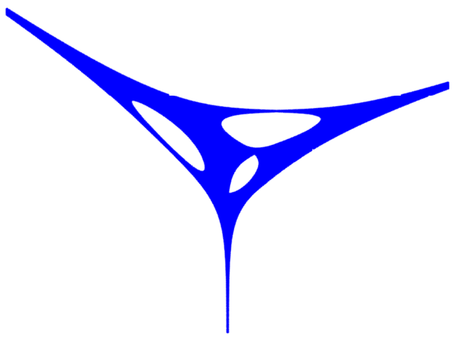

Figure 2.2: The Amoeba of $1+x_1+x_1^2+x_1^3+x_1^2x_2^3+10x_1x_2+12x_1^2x_2+10x_1^2x_2^2$

Just as we observed that there is a relation between Newt(f) and Amoeba(f), there is a similar connection between Amoeba(f) and ArchTrop(f). Each point of one of them is close to some point of the other and in fact, the complements of both sets are topologically similar [2]. Also, the tentacles of Amoeba(f) correspond the rays of ArchTrop(f). Lastly, both the ArchTrop and the Amoeba can be useful in speeding up polynomial system solving [2]. Below is a picture of ArchTrop(f).

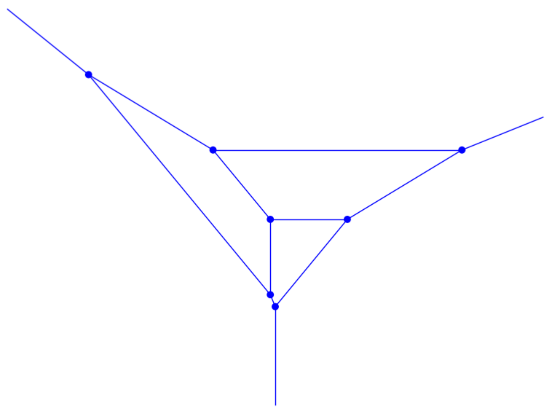

Figure 2.3: The ArchTrop of $1 + x_1 + x_1^2 + x_1^3 + x_1^2x_2^3 + 10x_1x_2 + 12x_1^2x_2 + 10x_1^2x_2^2$

The late 20th century has seen a lot more work being done in both Algebraic and Tropical geometry. The amoeba of a polynomial was introduced in the mid 90s by I.M. Gelfand, M.M. Kapranov and A.V. Zelevinsky in their book [4].

In 2000, Oleg Viro's work related the Amoeba and ArchTrop of a univariate polynomial, and a bound on the distance between the two was attained [7]. In 2016, Avendaño et. al. in [3], building on the work of Oleg Viro, found that for a given polynomial $f \in \mathbb{C}[x_1^{\pm 1}, \cdots, x_n^{\pm 1}]$ with $k \geq 2$ monomials and Newt(f) having dimension d, then $1 \leq d \leq \min\{n, k-1\}$ and the following results hold:

- if f, is a monomial or binomial, then Amoeba(f) equals ArchTrop(f)
- if $k = d + 1$, then Archtrop(f) $\subset$ Amoeba(f) and both are contractible
- if $\phi(x) = 1 + x_1 + \cdots + x_{k-1}$ and $\psi(x) = (x_1 + 1)^{k-d} + x_2 + \cdots + x_d$, then the Amoeba($\phi$) contains a point that is $\log(k-1)$ away from ArchTrop(ϕ) and ArchTrop(ψ) also contains points that approach Amoeba(ψ) from a distance of $\log(k-d)$

2.2 Differences Between the ArchTrop and Amoeba of a Polynomial

Given that there is a close relation between the ArchTrop and Amoeba of a polynomial, each has its own unique characteristics and we list below, some of the unique characteristics as found in [2].

2.2.1 Some Properties of the ArchTrop of a Polynomial

The ArchTrop of a polynomial

- is a piecewise-linear curve
- is a polyhedral complex which means that it is a union of polyhedra which intersect only along common edges
- is not always a subset of the Amoeba of the same polynomial
- partitions $\mathbb{R}^n$ into a finite number of polyhedral cells with each having dimension bewtween 0 and n

In addition, the ArchTrop of 0 is defined as $\mathbb{R}^n$.

2.2.2 Some Properties of the Amoeba of a Polynomial

- The amoeba of a polynomial is a closed set

- An amoeba in 2D has tentacles
- Any connected component of the complement of the amoeba of a polynomial is convex

3. CONVEX HULL OF POINTS IN THREE-DIMESIONAL SPACE USING SAGE AND THE ARCHTROP OF A GIVEN BIVARIATE POLYNOMIAL

As discussed in the previous section, to be able to visualize the ArchTrop of a given bivariate polynomial, we would first need the Archnewt which is the convex hull of points in 3D. From the fact that we would need the outer normal of facets of the Archnewt, we would define the convex hull in such a way that it returns the facets in the convex hull, each, as a list of points. So, in this section, we will discuss the process of getting the 3D convex hull in Sage.

3.1 Steps to Derive the Convex Hull of Points in 3D in Sage

Given a list of points in 3D, the following steps describe how to get their convex hull in Sage and we call the function convex_hull3D:

- use the `Polyhedron` class in Sage to create a polyhedron whose vertices are the given points and in 3D
- use the `faces()` method of the Polyhedron class to get the 2-dimensional faces (facets) of the polyhedron in an auxilliary class
- use the `as_polyhedron()` method to change each facet in the auxilliary class into a polyhedron
- use the `vertices_list()` method to get the facets which have each been converted into a polyhedron as a list of lists
- convert each facet of the polyhedron which is a list of lists into a list of vectors

From the above sequence of steps, we can obtain the convex hull of any given list of points in 3D as a list of lists of vectors. Let us consider the following example

Example 2. *Consider the following points* $(0,0,0), (1,1,1), (0,1,0), (1,0,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (2,2,2)$ *in 3D, we will use the function* `convex_hull3D` *to find their convex hull.*

Input

`convex_hull3D`$([(0,0,0), (1,1,1), (0,1,0), (1,0,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (2,2,2)])$

Below is a plot of the points in 3D.

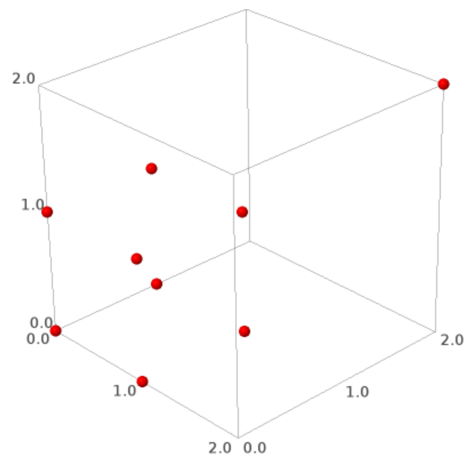

Figure 3.1: A plot of the points (0,0,0), (1,1,1), (0,1,0), (1,0,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (2,2,2) in 3D

Output

$[[(0,0,0), (0,0,1), (0,1,0), (0,1,1)], [(0,0,0), (0,0,1), (1,0,0), (1,0,1)], [(0,0,0), (0,1,0), (1,0,0), (1,1,0)], [(0,0,1), (0,1,1), (2,2,2)], [(0,1,0), (0,1,1), (2,2,2)], [(1,0,0), (1,0,1), (2,2,2)], [(0,0,1), (1,0,1), (2,2,2)], [(1,0,0), (1,1,0), (2,2,2)], [(0,1,0), (1,1,0), (2,2,2)]]$

See below a picture of the polyhedron whose facets are the output of

`convex_hull3D`$([(0,0,0),(1,1,1),(0,1,0),(1,0,0),(0,0,1),(1,1,0),(1,0,1),(0,1,1),$ $(2,2,2)])$

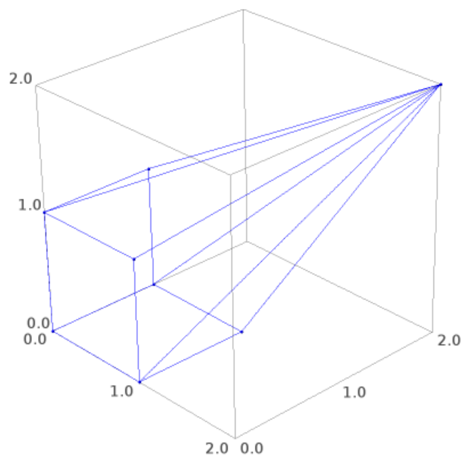

Figure 3.2: A plot of the convex hull of the points (0,0,0), (1,1,1), (0,1,0), (1,0,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (2,2,2)

3.2 The ArchTrop Algorithm for a Given Bivariate Polynomial

Given a bivariate polynomial $f(x) = \sum_{i=1}^{t} c_i\, x^{a_i}$, where $c_i \neq 0\, \forall i$ the following steps describe the process of obtaining its ArchTrop:

- let $P = \text{conv}(\{(a_i, -\log|c_i|)\}$
- consider the outernormals of the facets of P with negative last coordinate
- the vertices of the ArchTrop of $f(x)$ are $\{(\frac{-w_1}{w_3}, \frac{-w_2}{w_3}) | (w_1, w_2, w_3)$ is an outernormal of P with $w_3 < 0\}$

- vertex i and vertex j from above define an edge if the facets of P that they come from are adjacent
- a ray (a line perpendicular to an edge of the Newt(f)) emanates from a vertex of the ArchTrop if the edge of Newt(f) forms a part of the boundary of the facet from which the vertex of the ArchTrop comes from

3.3 Implementation of the ArchTrop Algorithm for a Bivariate Polynomial in Sage

Given that we have a 3D convex hull function in Sage that returns the vertices of each facet of the convex hull as a list of vectors, we proceed to describe the main result of this thesis, namely; the Sage function that returns a graphical representation of the ArchTrop of a given bivariate polynomial. In the process of doing so, we will use the 2D convex hull code presented in [6]
For a given bivariate polynomial f, the following steps describe how we get the ArchTrop of f in Sage:

- the `exponents()` method is used to get the support of f
- use the `newton_polytope()` method to get the Newton polygon of f
- use the `coefficients()` method to get the coefficients of f
- append the negative log of the coefficients to the respective support to get the points in 3D used to obtain the lifted support
- use the `convex_hull3D` function on the lifted support to get the facets of the Archnewt
- use the `normal_vect` function we wrote to find the normal vector of a face. To check to see if the normal vector obtained points outward, take a point in

the Archnewt that is not a vertex of the facet under consideration and call this point p. Find the position vector between the first vertex(call this vertex p_1) of the facet and p. If the dot product of the position vector from p to p_1 and the normal vector is positive, then multiply the normal vector by -1, else, maintain the normal vector since it is an outer normal

- project the facets of the Archnewt onto 2D
- obtain the edges of each facet of the projected Archnewt as a list of list of lists
- reverse the order of the edges of each facet of the projected Archnewt
- obtain the vertices of ArchTrop from the facets of the Archnewt whose outer-normals have a negative last coordinate
- using the steps in the ArchTrop Algorithm, obtain the vertices, edges and rays of the ArchTrop of f

In the following example, we will show how the Sage code from the above steps are used to obtain the vertices, edges and rays of the ArchTrop.

Example 3. *Let* $f(x) = 1 + x_1^3 + x_2^2 - 10x_1x_2$. *The support of this polynomial is* $(0,0),(3,0),(0,2)$ *and* $(1,1)$. *The points in the lifted support are* $(0,0,0),(3,0,0),$ $(0,2,0)$ *and* $(1,1,\frac{-239263565}{103910846})$. *The facets of the Archnewt, each as a list of points are,* $[(0,0,0),(0,2,0),(3,0,0)],[(0,0,0),(0,2,0),(1,1,\frac{-239263565}{103910846})],[(0,0,0),(3,0,0),$ $(1,1,\frac{-239263565}{103910846})],[(0,2,0),(3,0,0),(1,1,\frac{-239263565}{103910846})]$. *Figure 3.3 below is the Archnewt(f).*

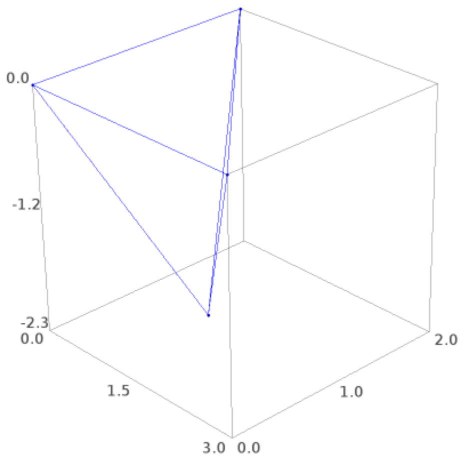

Figure 3.3: The vertices of the Archnewt of $1 + x^3 + y^2 - 10x_1x_2$

The outer normals of the facets are $(0, 0, 6), (\frac{-239263565}{51955423}, 0, -2), (0, \frac{-717790695}{103910846}, -3), (\frac{239263565}{51955423}, \frac{717790695}{103910846}, -1)$ *respectively. Since only the second, third and fourth outer normals have negative third components, it implies that ArchTrop(f) has three vertices, namely;* $(\frac{-239263565}{103910846}, 0), (0, \frac{-239263565}{103910846})$*, and* $(\frac{239263565}{51955423}, \frac{717790695}{103910846})$*. Below is a plot of the vertices of ArchTrop(f).*

Figure 3.4: The vertices of the ArchTrop of $1 + x^3 + y^2 - 10x_1x_2$

From the facets of the Archnewt as lists, we notice that any pairwise comparison between the second, third and fourth facets has an edge in common, that is, any two facets have two vertices in common. From this result, any two of the three vertices of ArchTrop(f*) will have an edge in common. The vertices and edges of the ArchTrop of* f *are represented below*

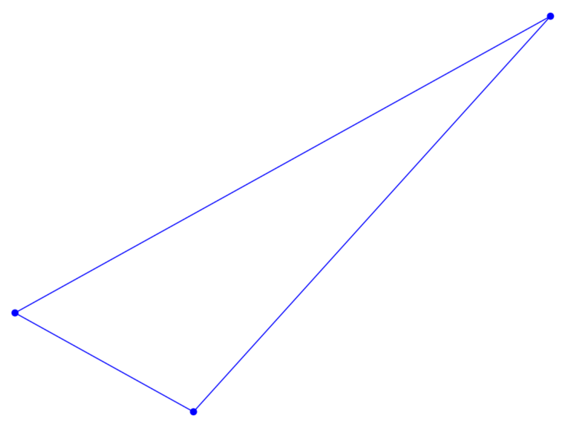

Figure 3.5: The vertices and edges of the ArchTrop of $1 + x^3 + y^2 - 10x_1x_2$

Now that we have the vertices and edges of ArchTrop(f), we complete the ArchTrop by obtaining the rays. For the first vertex of the ArchTrop, the only edge of Newt(f) that is part of the first vertex's corresponding facet is $[(0,0),(0,2)]$ *and the outer normal to this edge is* $(-2,0)$. *The edge of Newt(f) that is part of the boundary of the facet corresponding to the second vertex of ArchTrop(f) is* $[(0,0),(3,0)]$ *and has outer normal (0, -3). Lastly, for the third vertex of ArchTrop(f), the only edge of Newt(f) that is a part of the boundary of its corresponding facet is* $[(0,2),(3,0)]$ *and has* $(2,3)$ *as the outer normal to the edge. So, from these edges and outer normals, we obtain ArchTrop(f). Below is a diagram of ArchTrop(f). The ArchTrop of the above polynomial has* 3 *vertices,* 3 *edges and* 3 *rays.*

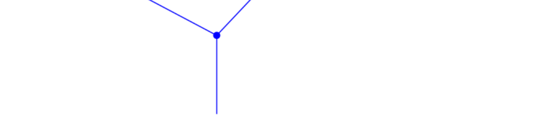

Figure 3.6: The ArchTrop of $1 + x^3 + y^2 - 10x_1x_2$

As discussed in Section 2*, we again see the relation between the ArchTrop and Amoeba of* f. *Comparing Figures 3.6 and 3.7 we see that the Amoeba and ArchTrop of* f *have a hole each and the three tentacles of the Amoeba corresond to the three rays of the ArchTrop.*

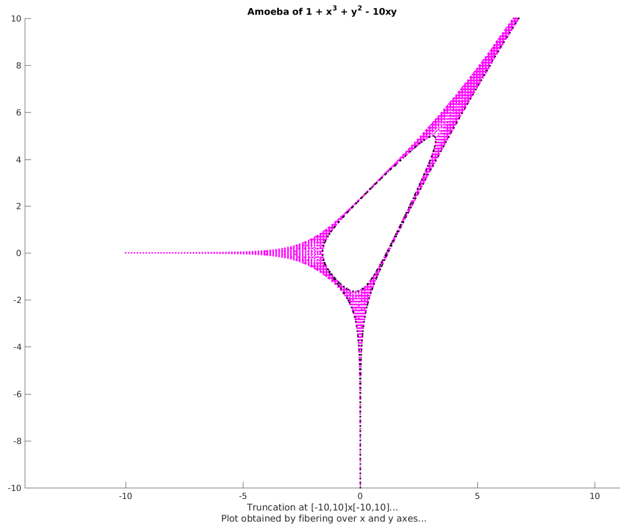

Figure 3.7: The Amoeba of $1 + x^3 + y^2 - 10x_1x_2$

In the above example, we saw the steps used in getting the ArchTrop of the function $f(x) = 1 + x^3 + y^2 - 10x_1x_2$ and compared it with its Amoeba. In the following, we will consider a slightly different example where the Amoeba and ArchTrop have no holes.

Example 4. *Consider the polynomial* $g(x) = 100+x+y+xy+100x^2+y^2$*. The support of* g *is* $(0,0),(1,0),(0,1),(1,1),(2,0)$ *and* $(0,2)$*. The points in the lifted support are* $(2,0,\frac{-239263565}{51955423}),(1,1,0),(0,2,0),(1,0,0),(0,1,0)$ *and* $(0,0,\frac{-239263565}{51955423})$*. The facets of the Archnewt of* g *are* $[(0,0,\frac{-239263565}{51955423}),(0,1,0),(0,2,0)],[(0,0,\frac{-239263565}{51955423}),(0,1,0),(1,0,0)],[(0,1,0),(0,2,0),(1,0,0),(1,1,0)],[(1,0,0),(1,1,0),(2,0,\frac{-239263565}{51955423})],$ $[(0,0,\frac{-239263565}{51955423}),(1,0,0),(2,0,\frac{-239263565}{51955423})],[(0,2,0),(1,1,0),(2,0,\frac{-239263565}{51955423})]$ *and* $[(0,0,\frac{-239263565}{51955423}),(0,2,0),(2,0,\frac{-239263565}{51955423})]$.

For the above facets, the following are their respective outer normals; $(\frac{-239263565}{51955423},0,0),(\frac{-239263565}{51955423},\frac{-239263565}{51955423},1),(0,0,1),(\frac{239263565}{51955423},0,1),(0,\frac{-478527130}{51955423},0),$

$(\frac{239263565}{51955423}, \frac{239263565}{51955423}, 0)$ *and* $(0, \frac{478527130}{51955423}, -4)$.

Of these outer normals, there is only one whose third coordinate is negative. This implies that the ArchTrop of g *will have only one vertex. The coordinates of the vertex of ArchTrop(g) is* $(0, \frac{239263565}{103910846})$.

Projecting the facet $[(0, 0, \frac{-239263565}{51955423}), (0, 2, 0), (2, 0, \frac{-239263565}{51955423})]$ *of Archnewt(g) down to 2D, we get a triangle with vertices* $(0, 0), (2, 0)$ *and* $(0, 2)$. *But this triangle is precisely Newt(g). The outer normals of the edges of Newt(g) are* $(0, -2), (2, 2)$ *and* $(-2, 0)$. *So, potting these vectors together with the point* $(0, \frac{239263565}{103910846})$ *gives ArchTrop(g). Below is a picture of ArchTrop(g).*

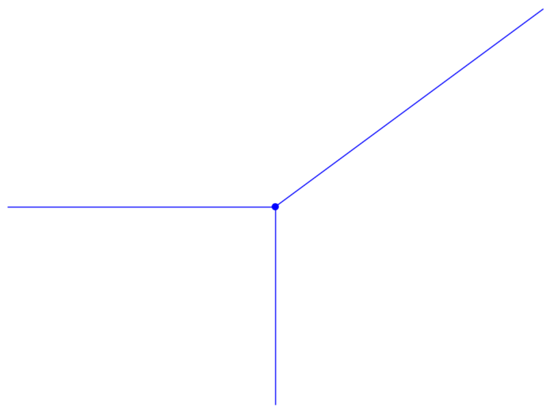

Figure 3.8: The ArchTrop of $100 + x + y + xy + 100x^2 + y^2$

As we compared the Amoebas and ArchTrops of polynomials in section 2 and in the previous example, we will do same here. Below is a picture of the Amoeba of g.

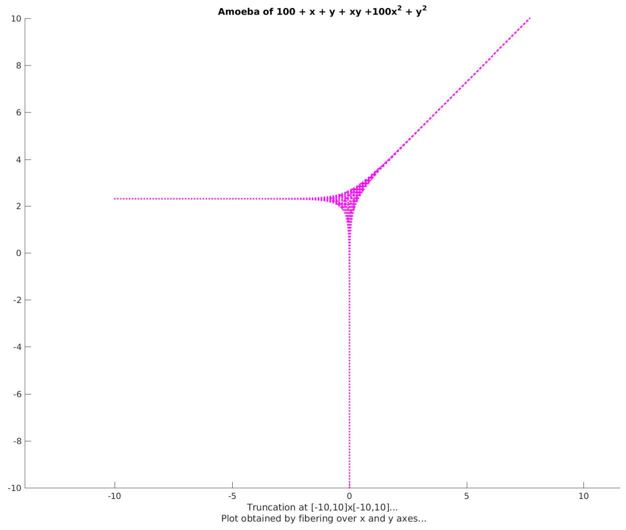

Figure 3.9: The Amoeba of $100 + x + y + xy + 100x^2 + y^2$

3.4 Conclusion

From all the discussions above, we see the relation between the Newton polygon, Amoeba and ArchTrop of a polynomial. Also, we see how the Sage functions can make the computation of the ArchTrop of a given bivariate polynomial so easy to compute. It will be a good exercise in future to extend this work to polynomials in three or more variables, bearing in mind that the convex hull function will be in 4D or even in a dimension higher.

4. CONCLUSIONS

From all the discussions above, we see the relation between the Newton polygon, Amoeba and ArchTrop of a polynomial. Also, we see how the Sage functions can make the computation of the ArchTrop of a given bivariate polynomial so easy to compute. It will be a good exercise in future to extend this work to polynomials in three or more variables, bearing in mind that the convex hull function will be in 4D or even in a dimension higher.

REFERENCES

[1] Oleg Alexandrov. A Photo of the Amoeba of $1 + x_1 + x_1^2 + x_1^3 + x_1^2 x_2^3 + 10x_1x_2 + 12x_1^2x_2 + 10x_1^2x_2^2$. `https://en.wikipedia.org/wiki/Amoeba_%28mathematics%29#/media/File:Amoeba3.svg`, 2007 (accessed May, 2016).

[2] Eleanor Anthony, Sheridan Grant, Peter Gritzmann, and J. Maurice Rojas. Polynomial-time Amoeba neighborhood Membership and Faster Localized Solving. In Fabien Vivodtzev Janine Bennett and Valerio Pascucci, editors, *Topological and Statistical Methods for Complex Data – Tackling Large-Scale, High-Dimensional, and Multivariate Data Sets*, chapter 15. Springer-Verlag, 2014.

[3] Martín Avendaño, Roman Kogan, Mounir Nisse, and J. Maurice Rojas. Metric estimates and membership Complexity for Archimedean Amoebae and Tropical Hypersurfaces. *Submitted for publication*, 2016.

[4] Israel M. Gelfand, Mikhail M. Kapranov, and Andrei V. Zelevinsky. *Discriminants, Resulatants, and Multidimensional determinants.* Birkhäuser Boston, 1994.

[5] Isaac Newton. Letter to Oldenburg dated 1676 Oct 24, the correspondence of Isaac Newton II. *Cambridge University Press.*

[6] Tom Switzer. 2D Convex Hulls. `http://tomswitzer.net/2010/03/graham-scan/`, 2010 (accessed March, 2016).

[7] Oleg Viro. Dequantization of Real Algebraic Geometry on a Logarithmic paper. In Joan Verdera Carles Casacuberta, Rosa Maria Miro-Roig and Sebastia Xambo-Descamps, editors, *European Congress of Mathematics*, volume 1. Barcelona, July 10-14 2000.

APPENDIX A

3D CONVEX HULL FUNCTION AND OTHER AUXILLIARY FUNCTIONS IN SAGE

```
#The convex_hull3D function takes a list of points as its
#argument and returns their convex hull as list of lists
#of points.
def convex_hull3D(points):
    Poly = Polyhedron(points)
    n = dim(Poly)-1
    # In the next piece of code, we get the
    #2-dimensional faces (facets of Poly)
    #as vertices
    twodimfaces_polTet=Poly.faces(n)
    facets_polTet = []
    for face in twodimfaces_polTet:
        face_as_poly = face.as_polyhedron()
        face_as_vert = face_as_poly.vertices_list()
        facets_polTet.append(face_as_vert)
    polTet_facets_as_vectors =[]
    for face in facets_polTet:
        temp_list_vect = []
        for vert in face:
            vert_as_vector = vector(vert)
            temp_list_vect.append(vert_as_vector)
        polTet_facets_as_vectors.append(temp_list_vect)
    return polTet_facets_as_vectors
# The following returns True if two facets are adjacent
def face_adjacent(vector1,vector2):
    count = 0
    for i in range(len(vector1)):
        for j in range(len(vector2)):
            if vector1[i] == vector2[j]:
                count= count + 1
    if count >=2:
        return True
    else:
        return False
def pairs(list1):
    temp1 = []
    for i in range(len(list1)):
        if i != len(list1)-1:
            temp2 = [list1[i], list1[i+1]]
            temp1.append(temp2)
        else:
            temp2 = [list1[-1], list1[0]]
            temp1.append(temp2)
    return temp1
```

In the above block of code, we add the 2D convex hull code from [6].

APPENDIX B

THE ARCHTROP FUNCTION

```
#The archtrop function below returns the Archimedean Tropical
#Variety of a given bivariate polynomial
def archtrop(f):
    supp = []
    supp.append(f.exponents())
    suppor = supp[0]
    newt=f.newton_polytope()
    newt_center = newt.center()
    newt_poly_bd = pairs(convex_hull(suppor))
    newt_poly_bdry = []
    for edge in newt_poly_bd:
        temp = []
        for pt in edge:
            temp.append(vector(pt))
        newt_poly_bdry.append(temp)
    coeffs= f.coefficients();
    new_coeff = []
    for i in range(len(suppor)):
        new_coeff.insert(i,coeffs[suppor.index(suppor[i])])
    support = []
    for i in range(len(suppor)):
        pts_for_archnewt = tuple(suppor[i]) +
        (QQ(float(-log(abs(new_coeff[i])))),)
        support.append(pts_for_archnewt)
    #Poly is the polyhedron in Sage from the "lifted support"
    Poly = Polyhedron(support)
    # facets is the ArchNewt, ie, the convex hull of the
    #"lifted support"
    facets = convex_hull3D(support)
    #facets_proj2d is a projection of the 3D facets of
    #ArchNewt in 2D
    facets_proj2d = []
    for face in facets:
        each_face = []
        for j in range(len(face)):
            face_as_list = list(face[j])
            del face_as_list[-1]
            each_face.append(vector(face_as_list))
        facets_proj2d.append(each_face)
    # facets_proj2d_edges gives the 2D projection of the
    #facets of ArchNewt as edges
    facets_proj2d_edges = []
    for face in facets_proj2d:
        edges = pairs(face)
        facets_proj2d_edges.append(edges)
    Vert=Poly.vertices_list()
    Vertices_of_Poly = []
    for v in Vert:
        Vertices_of_Poly.append(vector(v))
    Outward_normals =[]
    for face in facets:
        norm_vect = normal_vect(face)
        for vert in Vertices_of_Poly:
            if vert not in face:
                pos_vect = vert - face[0]
                if vector(norm_vect).dot_product(vector(pos_vect))>0:
                    Outward_normals.append(-1*(norm_vect))
                    break
                else:
                    Outward_normals.append(norm_vect)
                    break
            else:
                continue
    #vert_arch stores the vertices of the Archtrop
    verts_arch = []
    for vert in Outward_normals:
        if vert[-1]< 0:
            divide_by_last_coord = (-vert[0]/vert[-1], \\
            -vert[1]/vert[-1])
            verts_arch.append(divide_by_last_coord)
        else:
            verts_arch.append([])
    rev_facets_proj2d_edges_list = []
    facets_proj2d_edges_copy = facets_proj2d_edges[:]
```

```
for i in range(len(facets_proj2d_edges_copy)):
    temp_rev_faces_list = []
    for j in range(len(facets_proj2d_edges_copy[i])):
        temp_list = []
        for k in range(len(facets_proj2d_edges_copy[i][j])):
            pt_as_list = list(facets_proj2d_edges_copy[i][j][k])
            temp_list.append(pt_as_list)
            temp_list.reverse()
        temp_rev_faces_list.append(temp_list)
    rev_facets_proj2d_edges_list.append(temp_rev_faces_list)
rev_facets_proj2d_edges = []
for i in range(len(rev_facets_proj2d_edges_list)):
    rev_face = []
    for j in range(len(rev_facets_proj2d_edges_list[i])):
        rev_edge = []
        for k in range(len(rev_facets_proj2d_edges_list[i][j])):
            rev_edge.append(vector\\
            (rev_facets_proj2d_edges_list[i][j][k]))
        rev_face.append(rev_edge)
    rev_facets_proj2d_edges.append(rev_face)
arch_edges = []
# Here, if an element of verts_arch is not empty(ie, !=[]), then
#it means its 3rd coord is -ve, so we plot those pts
for vert in verts_arch:
    if vert != []:
        arch_edges.append(point(vert))
```

```
#The code below compares any 2 facets of ArchNewt, if any two are
#adjacent and their corresponding verts_archs are not empty, then
#we draw a line to connect the two verts_arch's
for i in range(len(facets)):
    for j in range(len(facets)):
        if i < j and face_adjacent(facets[i],facets[j])== True \\
         and verts_arch[i]!=[] and verts_arch[j]!=[]:
            L= line([verts_arch[i], verts_arch[j]])
            arch_edges.append(L)
#Here, we look at the edges of the ArchNewt projected onto the 2D
#plane. If an edge or its reverse is part of the bdry of the
#Newton polygon, then we draw a line that is normal to the edge
#from the corresponding verts_arch with the start point of the
#ray being verts_arch and the end pt is the position vector of
#the normal added to the verts_arch
for k in range(len(facets_proj2d_edges)):
    if verts_arch[k] != []:
        for l in range(len(facets_proj2d_edges[k])):
            if facets_proj2d_edges[k][l] in newt_poly_bdry:
                norm2newtpolybdry=\\
                normal_vect2d(facets_proj2d_edges[k][l])
                norm2newtpolybdry_ritedrxn = -1*norm2newtpolybdry
         new_position_as_tuple  = tuple(new_position )
                L1= line([verts_arch[k],new_position_as_tuple])
                arch_edges.append(L1)
            elif rev_facets_proj2d_edges[k][l] in newt_poly_bdry:
                norm2newtpolybdry=\\
                normal_vect2d(rev_facets_proj2d_edges[k][l])
                norm2newtpolybdry_ritedrxn =\\
                 -1*norm2newtpolybdry
                new_position  = vector(verts_arch[k]) + \\
                norm2newtpolybdry_ritedrxn
                new_position_as_tuple = tuple(new_position)
                L2= line([tuple(verts_arch[k]),\\
                 new_position_as_tuple])
                arch_edges.append(L2)
    else:
        continue
show(sum(arch_edges), axes=False)
```