ANALYSIS OF FIBONACCI NUMBERS IN PHYLLOTAXIS

ALLAN NEWMAN

Abstract. Fibonacci numbers occur naturally in the number of clockwise and anticlockwise spirals present in the formations of leaves and seeds of many plants. The physical structure of these natural arrangements can be modelled by deformable unit area cylindrical lattices with a generating helix where lattice points represent the location of the plants' leaves or seeds. The number of clockwise and anticlockwise spirals (m, n) are called the parastichy pair of this lattice. All possible lattices can be generated by two parameters, x and y, which are dependent on the divergence angle of the generating helix, the circumference of the cylinder and chosen height intervals. We thus have a moduli space of all possible lattices which is tessellated by the parastichy pairs. The energy function of lattices is a function on this moduli space, and we can analyze local minimum trajectories on this function to indicate lattices which are naturally occurring. In this report, we further discuss the mechanics of this model and present potential future steps to further understand the natural phenomenon driving these pattern formations.

1. Introduction

The Fibonacci numbers are formed by a recursive sequence that is defined as follows:

$$a_0 = 0, \quad a_1 = 1, \quad a_n = a_{n-1} + a_{n-2}$$

Thus yielding the sequence $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...$

The Fibonacci sequence of numbers has been widely observed in the natural world. The spirals of leaves and seeds on plants is just one example of this. The study of the arrangements of leaves on plant stems is called **phyllotaxis**. There exists many types of phyllotaxis, but in this report, we will only observe spiral phyllotaxis.

Among the plants that demonstrate spiral phyllotaxis are pine cones, sunflowers and pineapples. When examining these plants, one can observe their spirals through many contexts but for our research, we examine the fact that there will be certain numbers of spirals in the clockwise and anticlockwise direction. Figure 1 below demonstrates how we will be counting these spirals and one occurrence of Fibonacci numbers in the number of spirals. The red lines follow the anticlockwise spirals, of which there are 13, and the blue lines follow the clockwise spirals, of which there are 8. Since this pine cone has 8 spirals in one direction and 13 spirals in the other, we will label this pine cone with the **parastichy pair** $(8, 13)$, which you may note to be consecutive Fibonacci numbers.

Parastichy pairs will be how we label all structures that we want to count the number of spirals on. In all parastichy pairs (m, n), we will order the numbers in the pair such that $m \leq n$. For our purposes, if the number of clockwise and anticlockwise spirals are switched,

Date: August 10, 2024.

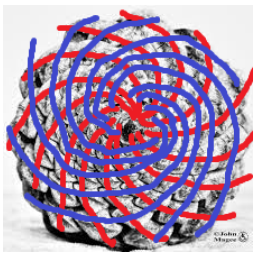

FIGURE 1. Pine cone with spirals shown in red and blue

we still want to label those structures the same. Our claim that we seek to find results for is that consecutive Fibonacci number parastichy pairs occur naturally due to the structures with these pairs being the ones which locally minimize the energy required to form them. It is reasonable to imagine that growth by some amount left or right will require the same amount of energy, so we can label all structures by their number of spirals, regardless of which direction has more spirals.

The observation of Fibonacci numbers in the number of spirals in plant leaf formation is not a recent discovery. Leonardo da Vinci observed this pattern in one of his journals in 1503. More recently, D'Arcy Thompson, a Scottish mathematical biologist, made further inquiries in his 1917 book *On Growth and Form*.[1] In 1998, Hyun-Woo Lee and Leonid Levitov, physicists from MIT, published a journal article where they propose a method by which this behavior can be modelled. It was our goal to use their model to develop mathematical tools and to prove results about this phenomenon.

2. MODEL

Lee and Levitov's model starts by modelling the stem of a plant by a cylinder. We place a helix upon this cylinder, this helix represents the path along which new leaves grow. Thus you can imagine that when the first leaf of on a plant grows, it will be at the base of cylinder on the helix. Then as each subsequent leaf grows, the previous leaves will move along this helix, spiralling up the cylinder. We can place points along the helix at certain height intervals, such that the height difference between each set of consecutive points is equal.[1] Figure 2 below demonstrates one such cylinder.

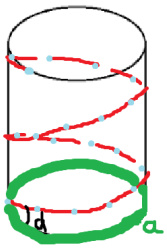

FIGURE 2. Cylindrical lattice demonstrating model

This creates a cylindrical lattice, with parameters d, r, and a. d is the measure of the divergence angle of the helix, in other words, the angle between the ground and the helix.

r is the rise rate, meaning the height increase from each lattice point to the next. a is the circumference of the cylinder.

Using this cylindrical lattice, we can convert to a planar lattice by a procedure of rolling the cylinder out along the generating helix. This yields lattice points in a line upon a plane. The m^{th} lattice point along the generating helix can be found from the following formula:

$$r_m = \frac{adm}{2\pi}\boldsymbol{i} + rm\boldsymbol{j}$$

In order to create a full rank lattice, we extend the lattice points from the generating helix left and right by a units, this yields a lattice which can be analyzed properly as it can be observed along the generating helix and within the range of $[0, a)$, since by extending by a left and right, within any range of length a, there will be copies of the original lattice points that will be placed as if you were to cut and place the cylindrical lattice flat onto the plane. Within these a length intervals, we can observe the flatten cylinder with the extended lattice points representing lattice points that occur after one or more revolutions of the cylinder. This extended lattice also requires a modified formula for lattice points:

$$r_{pm} = (\frac{adm}{2\pi} + ap)\boldsymbol{i} + rm\boldsymbol{j}$$

where r_{pm} is the p^{th} extension by a of the m^{th} lattice point along the generating helix.

As we are using a planar lattice, it is useful to use Cartesian coordinates and thus we change parameters from d, r, and a to x, y, and A, where x is the relative row-to-row displacement of the generating helix, y is the height-to-circumference ratio and A is the unit cell area, meaning the area formed by the parallelogram with vertices $(0, 0)$, the endpoints of two basis vectors and the endpoint of the sum of the basis vectors.[1] These parameters have the following relations to d, r, and a:

$$x = \frac{-d}{2\pi}, \quad y = \frac{r}{a}, \quad A = ar$$

. With this change of parameters, we also have a different formula for lattice points:

$$r_{pm} = \sqrt{A}(\frac{p - mx}{\sqrt{y}})\boldsymbol{i} + m\sqrt{Ay}\boldsymbol{j}$$

This formula for lattice points can generate all possible lattices by simply changing x, y, and A. We choose to add the condition that for all lattices $A = 1$, this means that we re-scale the lattice, without changing any ratios of the generating vectors, to make $A = 1$.[2] Since x and y parameterize every possible lattice, it leads to the introduction of a moduli space of all possible lattices, with each pair (x, y) representing a unique lattice. It also gives a final lattice point formula:

$$r_{pm} = \frac{p - mx}{\sqrt{y}}\boldsymbol{i} + m\sqrt{y}\boldsymbol{j}$$

In Figure 3, there is one example of a lattice which we have generated using the circumference and generating helix. The red line segments indicate the basis vectors, which have been re-scaled to ensure $A = 1$.

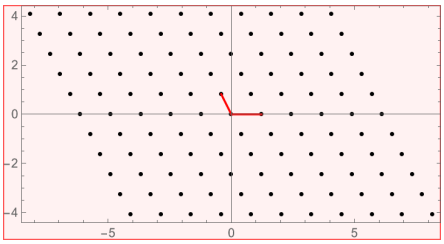

FIGURE 3. Planar lattice with generating helix vector and circumference vector

3. Main Content

3.1. **Identifying Parastichy Pairs.** As was previously established, the plants we are studying have parastichy pairs for the number of spirals in their leaf formations. We also observe parastichy pairs in lattices. For each lattice, we observe its shortest and second shortest vector which start from the origin and extend to a lattice point. The one additional condition we observe is that these vectors must be linearly independent.

Once we have the two shortest linearly independent vectors, we observe what lattice points are the endpoints of these vectors. For each of these vectors, the lattice point with be labelled $r_{p'm'}$, the value of m' indicates which point along the original generating helix, this point is an extension of. This value m' tells how many spirals there are in one direction, this is due to the fact that, there must be m' lines within an a length interval such that if you were to place a copy of that shortest (or second shortest) vector upon each lattice point, the number of lines formed by these vectors that intersect the $\boldsymbol{i}$-axis will be m'. Alternatively, since we start at the origin (r_{00}), we know the points it will lie on a line with will have m values equal to multiples of m' so if you we to continue along the generating helix, the first point which is a multiple of m' would be m', so all of the lattice points before $r_{0m'}$ along the helix will be part of a different line, however we want to only consider the lattice points within an a length interval, however, we can simply just use the extended lattice points for any m such that r_{0m} is not in the desired interval. Thus if we have the shortest and second shortest vectors, we can find the parastichy pair of any possible lattice.

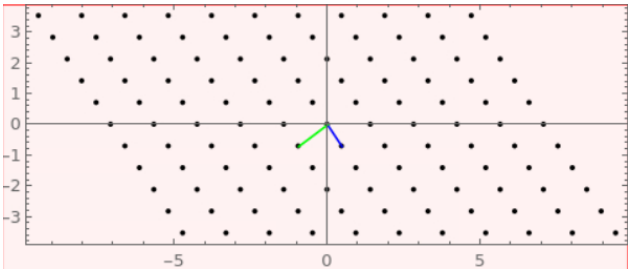

FIGURE 4. Lattice with shortest and second shortest vectors labelled

This can begin to be seen in Figure 4 above, which shows the shortest and second shortest vectors from the origin, if you we to place these vectors on every lattice point, you would yield two sets of parallel lines which represent the spirals on a plant.

3.2. **Parastichy Pair Detection Algorithm.** In order to determine the parastichy pairs for all possible lattice points, we develop a Mathematica algorithm which can perform the process for us.

```
xPoints = Subdivide[0, 1, 100];
yPoints = Subdivide[1/100, 1/2, 49];
gridPoints = Tuples[{xPoints, yPoints}];
mnSet = {};
For[i=1, i <= Length[gridPoints], i++,
    v1 = {-gridPoints[[i,1]] / Sqrt[gridPoints[[i,2]]],
        Sqrt[gridPoints[[1,2]]]};
    v2 = {1 / Sqrt[gridPoints[[1,2]]],0};
    range = 100;
    latticePoints = Flatten[Table[m v1 + n v2, {m, -range, range},
        {n, -range, range}], 1];
    distanceFromOrigin[latticePoint_] :=
        latticePoint[[1]]^2 + latticePoint[[2]]^2;
    distances = distanceFromOrigin[#] & /@ latticePoints;
    sortedIndices = Ordering[distances];
    sortedPoints = latticePoints[[sortedIndices]];
    matrix = {sortedPoints[[2]], sortedPoints[[3]]};
    k = 4;
    While[Det[matrix] == 0,
        matrix = {sortedPoints[[2]], sortedPoints[[k]]}; k++];
    k = k-1;
    mn = If[Abs[sortedPoints[[2,2]] / v1[[2]]] <=
            Abs[sortedPoints[[k,2]] / v1[[2]]],
        If[Abs[sortedPoints[[2,2]] / v1[[2]]] == 0,
            {1, Abs[sortedPoints[[k,2]] / v1[[2]]]},
        {Abs[sortedPoints[[2,2]] / v1[[2]]],
            Abs[sortedPoints[[k,2]] / v1[[2]]]}],
        If[Abs[sortedPoints[[k,2]] / v1[[2]]] == 0,
            {1, Abs[sortedPoints[[2,2]] / v1[[2]]]},
        {Abs[sortedPoints[[k,2]] / v1[[2]]],
            Abs[sortedPoints[[2,2]] / v1[[2]]]}]];
    mnSet = Join[mnSet, {mn}];
];
gridMNSet = Transpose@{gridPoints, mnSet};
```

The above code finds parastichy pairs for each (x, y) with x and y being values equal to an integer divided by 100 for $x \in [0, 1]$ and $y \in (\frac{1}{100}, \frac{1}{2}]$. This is just a chosen grid that we do the parastichy pair detection on, but can be expanded or shrunk depending upon the results you would like to yield. Additionally, the above code has a range of 100, this means when generating the lattice, it extends the lattice 100 points along the generating helix forward and backwards as well as extending the lattice 100 points left and right by the circumference.

As for the actual mechanics of the algorithm, after the grid has been decided on and the range has been determined. The algorithm subdivides the interval given then pairs all x-values and y-values together to have ordered pairs to operate on. It then initializes the mnSet for later use. A for loop is then began which will operate on each grid point. From the grid point, the generating vectors are produced, this is done by plugging x and y into the lattice point formula, with v_1 being the vector along the generating helix and v_2 being the vector which extends by the circumference. These two vectors are then used to create the lattice which extends 100 (the range) points up, down, left and right from the origin. A distance function is then defined, which is then used to sort all distances between the origin and lattice points. The corresponding indices to the distances are then sorted, which then allows for the sorting of the lattice points by distance. Since the origin is one of the lattice points, we start by looking at the second and third shortest distances. We place the points into a matrix and then check the determinant. If the determinant is nonzero, then these two vectors are the shortest and second shortest linearly independent vectors. If not then, we replace the third shortest distances with the fourth then fifth and so forth until we have linear independence. The while loop used does continue to add to our counter k once after we achieve linear independence, so we must reduce k by 1 to get the correct index for the second shortest vector. The algorithm then checks the height of the two vectors which, when divided by the initial height rise from the lattice points on the generating helix, tells what the m value of the endpoints will be. It then puts these points into a pair for storage in mnSet. This process runs through all of the grid points and then finally combines the grid point set and mnSet to yield a table which associates each grid point to a parastichy pair.

(x, y)	(m, n)	Coloring Assignment
$(\frac{1}{3}, \frac{1}{2})$	$(1, 1)$	Blue X
$(\frac{1}{2}, \frac{1}{2})$	$(1, 1)$	Blue X
$(\frac{3}{4}, \frac{1}{4})$	$(1, 2)$	Black Dot
$(\frac{2}{3}, \frac{1}{10})$	$(1, 3)$	Green X
$(\frac{7}{12}, \frac{1}{10})$	$(1, 4)$	Red X
$(\frac{99}{100}, \frac{1}{25})$	$(1, 6)$	Orange Dot
$(\frac{5}{6}, \frac{1}{100})$	$(1, 6)$	Orange Dot
$(\frac{11}{12}, \frac{1}{100})$	$(1, 12)$	Purple Dot
$(\frac{5}{12}, \frac{1}{500})$	$(5, 12)$	Green Dot
$(\frac{7}{16}, \frac{1}{500})$	$(7, 16)$	Red Dot

FIGURE 5. Table of Parastichy Pairings with Coloring Convention

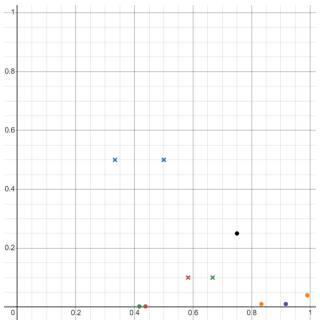

FIGURE 6. Sample Plane Coloring Based Upon Parastichy Pairs

3.3. **Coloring of the Plane Using Parastichy Pairs.** This algorithm that we developed is the first step in a greater goal of determining what parastichy pairs are naturally occurring. This can be conducted by a process requiring two objectives to be complete. As established, there is a moduli space of all possible lattices, since our algorithm can produce the parastichy pair for every lattice based upon its (x, y) pair, we create a coloring scheme for all parastichy pairs. It was previously established in Lee and Levitov's paper that lattices with the same parastichy pair appear in regions, specifically regions in the shape of hyperbolic quadrilaterals, which will tessellate the plane.[1] We call these regions, the parastichy pair domains because within each the same parastichy pair will be present for all lattices. In our research, a full algorithm that could complete this coloring was not fully developed but a sample coloring can be produced via using the parastichy pair detection algorithm for some set of grid points and then producing a coloring manually. Above are a table of sample points with a coloring scheme (Figure 5) and an image of said points graphed with coloring (Figure 6), these figures illustrate an intermediate step towards the complete coloring of the plane based upon parastichy pairs.

3.4. **Energy Minimizing Lattices.** Once the coloring of the plane is complete, there is one other major form of analysis we want to complete on these lattices. For the set of lattices, we define an energy functional as follows:

$$E_{total} = \frac{1}{2} \sum U(|r_{pm} - r_{p'm'}|)$$

with $U(r)$ being a generic repulsive interaction. The issue that arises with this energy functional is that it will diverge, so we can instead look at the energy density on each cell, A, which as we established, will always be 1, thus we can yield the following density functional:

$$E(x, y) = \sum U(|r_{pm}|)$$

We do have to consider one condition to ensure this sum converges. $U(r)$ must be some function which will decay sufficiently such so that the sum converges. For our purposes, this will mean choosing functions such as exponential decay functions and power functions.

Once the summation is set up, we can then run the following code to determine the energy density for all lattices.

```
ContourPlot[Sum[Sum[Exp[-5 Sqrt[ (p-m*x)^2 / y + m^2 * y]], {p, 0, 100}],
```

```
{m, 0, 100}], {x, 0, 1}, {y, 0, 1}]
```

This code chooses to run the sum for p and m up to 100, but any suitable large choice will yield the necessary result. It also looks at the $0 \leq x \leq 1$ and $0 \leq y \leq 1$. The figure below shows just the domain for $0.5 \leq x \leq 1$ and $0 \leq y \leq 0.5$ only summing m and p to 20.

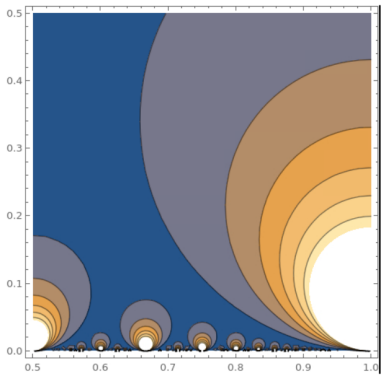

FIGURE 7. Contour Plot for Energy of Lattices with $x \in [\frac{1}{2}, 1], y \in [0, \frac{1}{2}]$

Once the energy for all lattice has been computed, we seek to find the minimum trajectory for energy on all lattices. This means for y-value, we want to find the local minima, meaning for every y, we want to find each x where the following is true:

$$\frac{\partial E}{\partial x} = 0, \ \frac{\partial^2 E}{\partial x^2} > 0$$

In order to do this, one would need to create an algorithm which does the following:

(1) For given y determine x-values for all local minima
(2) Store each determined x-value in an list of ordered pairs, paired with the y-value given
(3) Repeat for all y-values in chosen domain
(4) Graph all stored ordered pairs, resulting in a trajectory showing all energy minima

We want to determine the minimum trajectory because this trajectory is going to show all of the lattices which require the least energy locally to maintain. In the physical setting using pine cones, this will mean that if a pine cone was growing more scales upon its stem and as it grows it is going to tend towards the structure of these minimum energy lattices. Thus, once the above algorithm is complete, it will show the lattices which are likely to naturally occur.

This leads to the final step in determining naturally occurring parastichy pairs. If we overlay the minimum trajectory over the parastichy pair coloring of the plane, we will see that the trajectory moves only through certain parastichy pair domains, so we determine that the parastichy pairs associated to these domains are the naturally occurring parastichy pairs, which we expect to find, will be the domains associated with parastichy pairs consisting of consecutive Fibonacci numbers.

4. Conclusions

4.1. **Final Remarks.** There is still a fair amount to be completed to have a fully automated process for determining which parastichy pairs are naturally occurring but the parastichy pair detection algorithm serves as a major step in completing this process.

Acknowledgements. This report is based on work supported by NSF grant DMS-2051032, which we gratefully acknowledge. I would also like to express my thanks to the Mathematics department of Indiana University for hosting the program, Dr. Wai-Tong (Louis) Fan for directing this program and most of all my mentor Dr. Sanjana Agarwal for all of the help in understanding and working on this problem.

References

[1] Hyun Woo Lee and Leonid Levitov. Universality in phyllotaxis: A mechanical theory. *Series in Mathematical Biology and Medicine*, 1998.
[2] Oded Regev. Lecture 1 introduction 1 lattices.

Department of Mathematics and Computer Science, Indiana State University, Terre Haute, Indiana 47809 United States

Email address: atnewman2002@gmail.com