A SOFTWARE PACKAGE FOR DECOMPOSING AFFINE SEMIGROUPS

MADELINE CHEN

Abstract. An affine semigroup is a set of non-negative integer vectors that is closed under addition. Affine semigroups are central objects of study in algebraic combinatorics, discrete geometry, and optimization. Many noteworthy functions that are defined on an affine semigroup restrict to polynomials on subsets of the respective semigroup, called cones. Many of the proofs that such decompositions exist are existential, meaning that no precise collection of cones is known. By better understanding a precise decomposition into cones, we can gain further insight into its original function. This paper introduces AffineSemigroup, the software package, designed to ease the process of finding a semigroup decomposition as a union of cones. Utilizing the existing graphics system in Sage, an open source computer algebra system comparable to Mathematica or Matlab, our software package automates the process of defining cones one-by-one in succession until a complete decomposition of the semigroup is obtained. We provide a clear and intuitive interface allowing the user to choose the basepoint and generators that define each cone, at each step updating the graphic representation to aide the user in placing the next cone.

1. Introduction

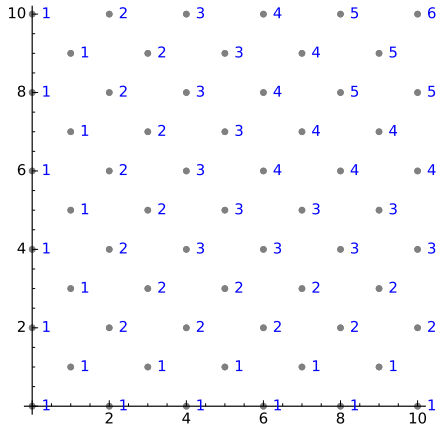
An affine semigroup is a set of non negative integer vectors that is closed under vector addition, and a cone is a translation of a subsemigroup of an affine semigroup generated by linearly independent vectors.

For example, consider the semigroup generated by $(2,0),(1,1),(0,2) \in \mathbb{Z}^2_{\geq 0}$. We can see in Figure 1*a* the elements of the semigroup $A = \langle (2,0),(1,1),(0,2) \rangle$ generated up to $(25,25)$. In this particular example, this figure also illustrates the number of factorizations associated with each element of the semigroup. A factorization of an element is a tuple indicating how many generators are used to represent the element. For instance, $(15,15) = 7(2,0) + 1(1,1) + 7(0,2)$ is an example of a factorization of the element $(15,15) \in A$. The number of factorization counts the number of different ways to represent the element using the generators. Following the example above, $(15,15) \in A$ has a total of 8 factorizations given by

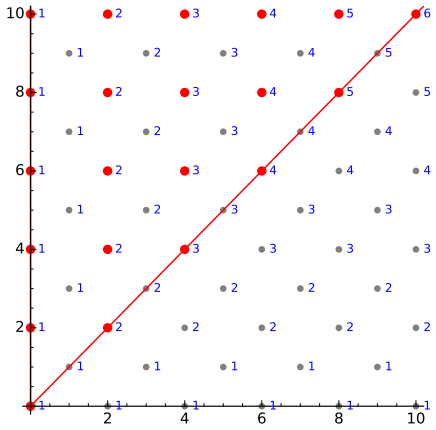
$$\{(7,1,7),(2,11,2),(0,15,0),(5,5,5),(6,3,6),(3,9,3),(4,7,4),(1,13,1)\}.$$

Date: June 14, 2018.

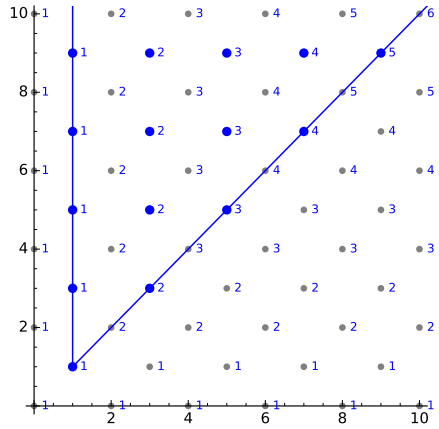
Key words and phrases. affine semigroup; computation; quasipolynomial.



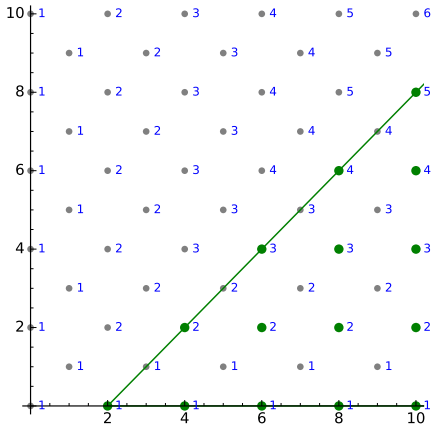
(A) $A = \langle (2,0), (1,1), (0,2) \rangle$



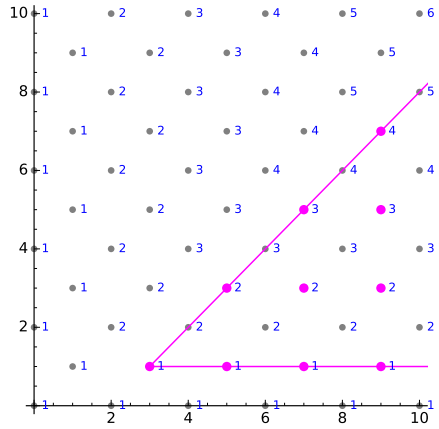
(B) $C_1 = \langle (0,0); (0,2), (2,2) \rangle$



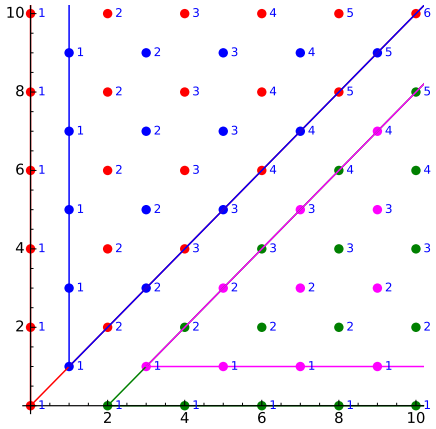
(C) $C_2 = \langle (1,1); (0,2), (2,2) \rangle$



(D) $C_3 = \langle (2,0); (2,0), (2,2) \rangle$



(E) $C_4 = \langle (3,1); (2,0), (2,2) \rangle$



(F) Decomposition

FIGURE 1

Utilizing the generators $(0,2)$ and $(2,2)$, we can restrict the semigroup to a cone $C_1 = \langle (0,0);(0,2),(2,2)\rangle$, and translating by $(1,1)$, we yield the cone $C_2 = \langle (1,1);(0,2),(2,2)\rangle$. Similarly, we can restrict the semigroup by $(2,0)$ and $(2,2)$, and translating by $(2,0)$ yielding the cone $C_3 = \langle (2,0);(2,0),(2,2)\rangle$ and and translating by $(1,1)$ yielding the cone $C_4 = \langle (3,1);(2,0),(2,2)\rangle$. Restricting the number of factorizations to each cone yields a polynomial $c_1x + c_2y + c_3 = f(x,y)$.

$$C_1 = \langle (0,0);(0,2),(2,2)\rangle \;\; ; \;\; \tfrac{1}{2}x + 0y + 1 = f(x,y)$$

$$C_2 = \langle (1,1);(0,2),(2,2)\rangle \;\; ; \;\; \tfrac{1}{2}x + 0y + \tfrac{1}{2} = f(x,y)$$

$$C_3 = \langle (2,0);(2,0),(2,2)\rangle \;\; ; \;\; 0x + \tfrac{1}{2}y + 1 = f(x,y)$$

$$C_4 = \langle (3,1);(2,0),(2,2)\rangle \;\; ; \;\; 0x + \tfrac{1}{2}y + \tfrac{1}{2} = f(x,y)$$

Notice how the union of C_1, C_2, C_3, and C_4 is equal to the A. This is known as a decomposition.

In this paper we introduce `AffineSemigroup`, a software package designed to ease the process of finding a semigroup decomposition as a union of cones. This current version supports the basic functionality of the algebraic structure affine semigroup by running needed functionalities through GAP. We also include plotting functions to help identify and spot patterns for relevant data pertaining to affine semigroup. This package also includes the functionalities of cones. In its own class definition, the `Cone` object has member functions that further benefits the decomposition process. In particular, unique member functions that gives the user more insight into the polynomials that exists behind each abstract object. Finally, this software package is currently focused towards the development of the `ConeDecomposition` class; specifically, to provide an automated process to allow the user to recursively define cones to obtain a complete decomposition of the semigroup desired.

This software package is available at

`http://github.com/coneill-math/affinesgps-sage`.

This paper will provided a summary of the relevant definitions that pertain to software package, in *Section 2*, then explore a comprehensive documentation of the software package. We explore in detail the functionalities of the object classes

`AffineSemigroup`, `Cone`, *and* `ConeDecomposition`

in *Section 3*. In addition, we give a detailed walkthrough of the automated Decomposition process provided by the `ConeDecomposition` class, showcasing the various options available in the interface in *Section 4*. As a result, we demonstrate an observation for 3-generated affine semigroup that was produced in the process of using the software package in *Section 5*. Finally, in *Section 6*, we conclude the paper with conjectures formulated from our observations.

2. Background

In this section, we recall several definitions used in this paper.

Definition 2.1. An *affine semigroup* A is a subsemigroup of $(\mathbb{Z}^d_{\geq 0}, +)$, $d \geq 2$, generated by some finitely many vectors. We write $A = \langle g_1, ..., g_n \rangle$, where $g_1, ..., g_n$ are the *generators* of A. The *factorization* of an element $a \in A$ is a tuple, $(x_1, ..., x_n)$ where $a = x_1 g_1 + ... + x_n g_n$. The *length* of a factorization $(x_1, ..., x_n)$ is $x_1 + ... + x_n$, i.e. the number of generators appearing in the factorization.

Example 2.2. Let $A = \langle (1,0), (1,1), (1,2), (1,3), (1,4), (1,5) \rangle$. Referring to Figure 2$a$ we can see the elements of A up to $(20, 20)$.

Example 2.3. Let $B = \langle (11,7), (17,23), (5,13) \rangle$. Referring to Figure 3$a$ we can see B represented up to the maximum of $(100, 100)$.

Definition 2.4. The *minimum factorization length* is defined as the minimum of the factorization set of an element. Similarly, the *maximum factorization length* is the maximum of the factorization set of an element.

Example 2.5. Referring to Figure 2c and 2d we can see the minimum and maximum factorization of elements in A represented up to $(20, 20)$. Similarly, we can see in Figure 3c and 3d, the the minimum and maximum factorization of elements in B represented up to $(100, 100)$.

Definition 2.6. Let $A = \langle g_1, ..., g_n \rangle$ be an affine semigroup.

1. A *relation* is a pair $\{(x_1, ..., x_n), (y_1, ..., y_n)\}$ of factorization for the same element $a \in A$.
2. The *minimal presentation* of A is a collection of minimal relations of its generators.

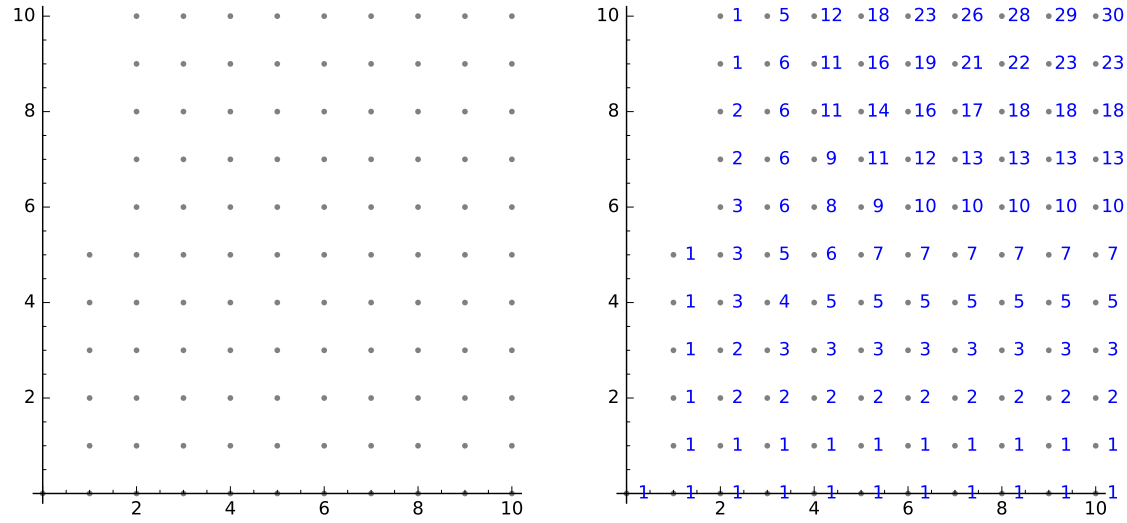
Definition 2.7. The *Cone* generated by $\alpha_1, ..., \alpha_n$ translated by β is the set

$$C = \langle \beta; \alpha_1, ..., \alpha_n \rangle = \left\{ \beta + \sum_{j=1}^{n} c_j \alpha_j : c_1, ..., c_n \in \mathbb{Q} \right\} \subset A.$$

Definition 2.8. Let A be an Affine Semigroup. We define a **decomposition** of A is the union of a finite set of cones $C_1, ..., C_n$ such that $A = \bigcup_i C_i$.

Example 2.9. Let $X = \langle (2,0), (2,2), (0,2) \rangle$. We can define the following cones, $C_1 = \langle (0,0); (0,2), (1,1) \rangle$ and $C_2 = \langle (2,0); (2,0), (1,1) \rangle$, such that $C_1, C_2 \subset X$ and $X = C_1 \cup C_2$ forms a decomposition of X. Refer to Figure 4.

Definition 2.10. Fix an affine semigroup A, function $f : A \to \mathbb{R}$ is *eventually quasipolynomial* if there exists a decomposition for A so that f restricts to a polynomial on each cone. For more information on quasipolynomial, we direct the reader to [2].



(A) The affine semigroup A in Example 2.2.

(B) Number of factorizations

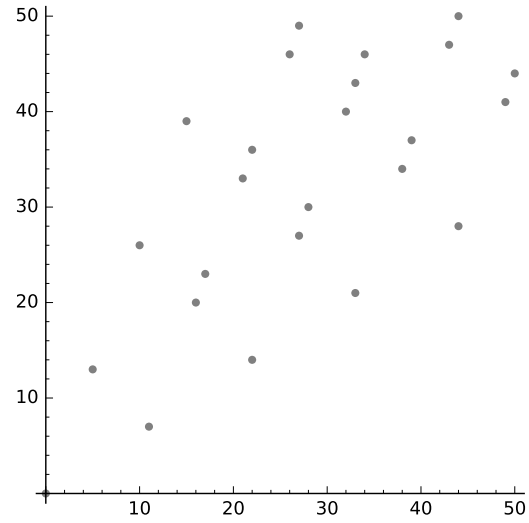
(C) Minimum factorization length

(D) Maximum factorization length

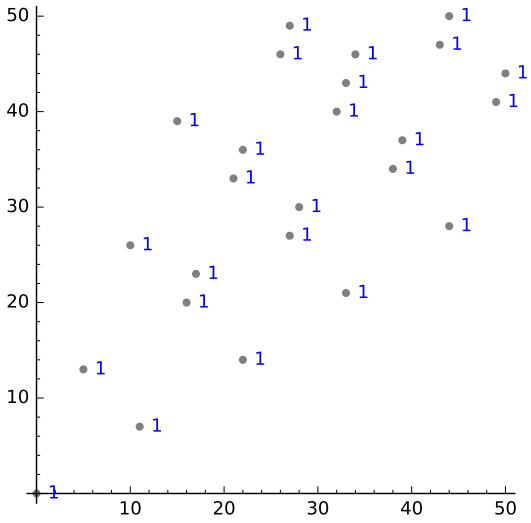
FIGURE 2

3. Class Documentation

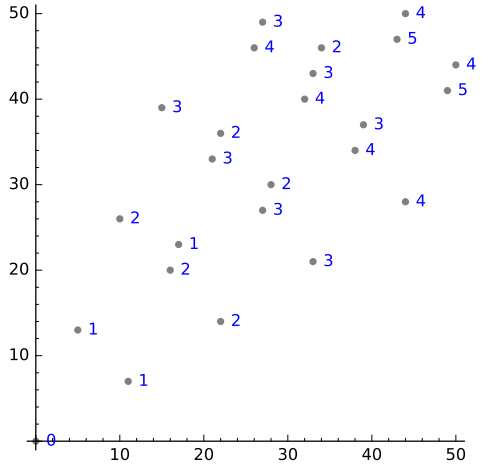
`AffineSemigroup` is a software package that consists of three main object definitions: `AffineSemigroup`, `Cone`, and `ConeDecomposition`. In this section, we demonstrate the functionality that each class provides.



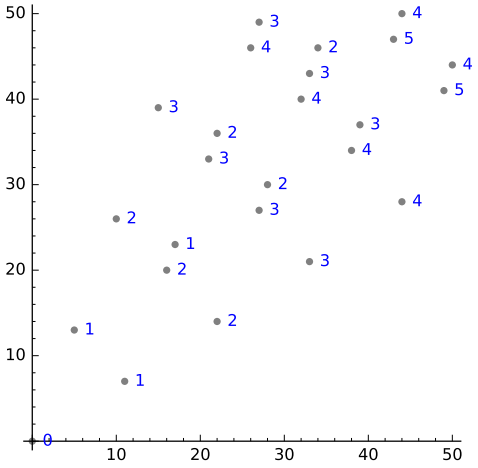
(A) The affine semigroup B from Example 2.3.



(B) Number of Factorization



(C) Minimum Factorization

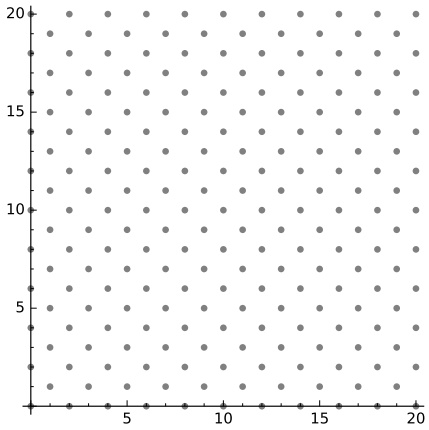


(D) Maximum Factorization

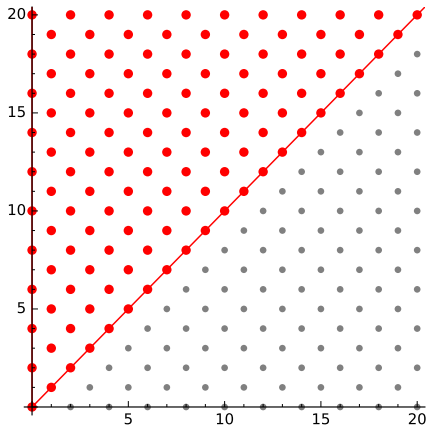
FIGURE 3

3.1. **The `AffineSemigroup` Class.** An object class that supports the basic functionality of the algebraic structure, Affine Semigroup, by running the need functionalities through GAP.

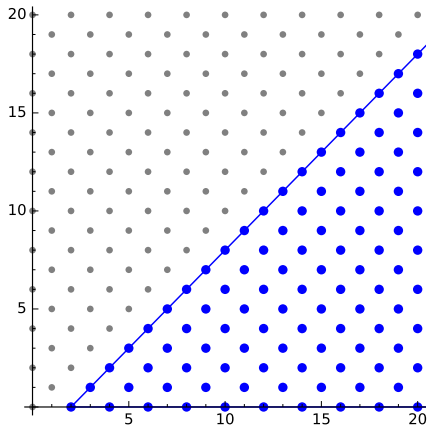
```
AffineSemigroup(self,gens)
```



(A) The affine semigroup X from Example 2.9.



(B) Cone1



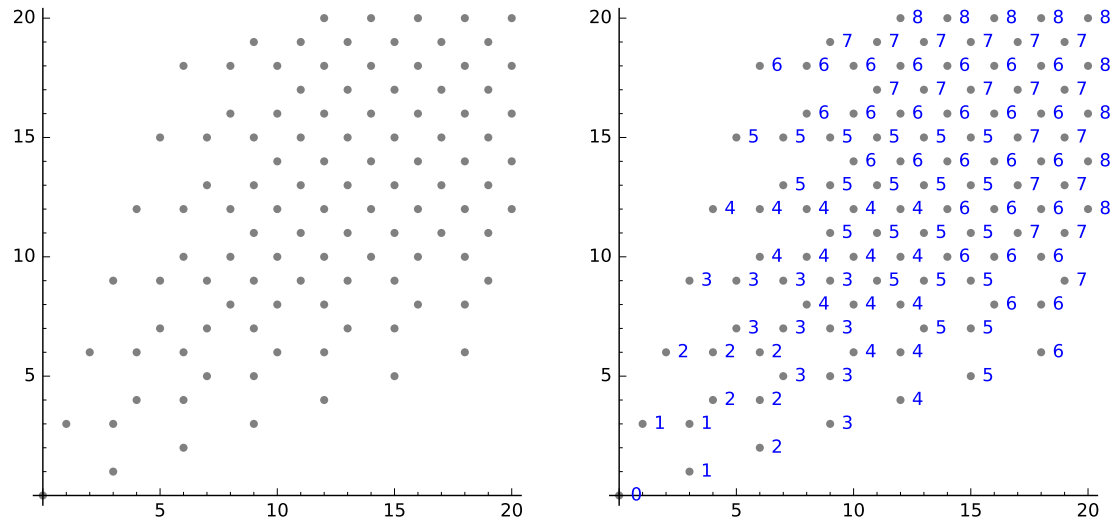
(C) Cone 2

FIGURE 4

The object constructor initializes a `AffineSemigroup` object by defining its generators, `gens`.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: A.gens
[[1, 3], [3, 1], [3, 3]]
sage: B = AffineSemigroup([[1, 3], [4, 5], [7, 9]])
sage: B.gens
[[1, 3], [4, 5], [7, 9]]
```

`__contains__(self,other)`



(A) `A.Plot(25,25)` (B) `A.Plot(25,25,MaxFact)`

FIGURE 5

A helper function that returns `True` if `other` is the element of the semigroup and returns `False` otherwise.

```
sage: A = AffineSemigroup([[2, 3], [3, 5], [5, 3]])
sage: [0,0] in A
True
sage: [2,3] in A
True
sage: [4,3] in A
False
sage: [4,4] in A
False
```

`GenericGapCallGlobal(self, gapfuncname)`
Given a gap object, this function returns the corresponding sage object. This functionality serves a a wrapper functions to call the existing GAP functions and convert them to Sage objects.

`Plot(self,xmax,ymax,ptsize=None,func=None)`
A member function that returns a plot that maps the members of the AffineSemigroup from [0,0] to [xmax,ymax].

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: A.Plot(20,20)
```

```
Launched png viewer for Graphics object consisting of 1 graphics
   primitive
sage: A.Plot(20,20,MaxFact)
Launched png viewer for Graphics object consisting of 182 graphics
   primitives
```

Refer to Figure 5.

`Factorizations(self,v)`
A member function that returns the factorization of the element v.

```
sage: A.Factorizations([40,50])
[[13, 8, 1], [10, 5, 5], [7, 2, 9]]
sage: B.Factorizations([40,50])
[[0, 10, 0]]
```

`FactorizationsUpToElement(self, v)`
A member function that utilizes a dynamic programming algorithm to generate the factorizations of every element up to the element v. For each element leading up to v, this function saves the factorization of the each element interally.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: time A.Factorizations([50,50])
CPU times: user 33.1 ms, sys: 6.31 ms, total: 39.4 ms
Wall time: 41.5 ms
[[11, 11, 2], [8, 8, 6], [5, 5, 10], [2, 2, 14]]
sage: time A.FactorizationsUpToElement([50,50])
CPU times: user 3.58 s, sys: 599 ms, total: 4.18 s
Wall time: 4.66 s
sage: time A.Factorizations([50,50])
CPU times: user 1.2 ms, sys: 703  s , total: 1.9 ms
Wall time: 1.46 ms
[[11, 11, 2], [8, 8, 6], [5, 5, 10], [2, 2, 14]]
```

`LengthSet(self,v)`
A member function that returns the length of the element v.

```
sage: A.LengthSet([1,3])
[1]
sage: A.LengthSet([15,15])
[5, 7]
sage: A.LengthSet([40,50])
[18, 20, 22]
sage: B.LengthSet([7,9])
[1]
```

```
sage: B.LengthSet([40,50])
[10]
```

`MinimalPresentation(self)`
A member function that returns the minimal presentation of the semigroup.

```
sage: A.MinimalPresentation()
[[[3, 3, 0], [0, 0, 4]]]
sage: B.MinimalPresentation()
[[[1, 12, 0], [0, 0, 7]]]
```

`ElementOfMinimalPresentation(self)`
A member function that returns the elements that corresponds to each set of elements in its minimal presentation.

```
sage: A.ElementOfMinimalPresentation()
[[12, 12]]
sage: B.ElementOfMinimalPresentation()
[[49, 63]]
```

`ElementsUpToElement(xmax,ymax)`
A helper function that takes in a semigroup, `semigroup` and returns a list of points up to `[xmax,ymax]` that are members of the `semigroup`.

```
sage: A = AffineSemigroup([[1, 2], [2, 1], [1, 5]])
sage: A.ElementsUpToElement(5,5)
[[0, 0], [1, 2], [1, 5], [2, 1], [2, 4], [3, 3], [4, 2], [4, 5], [5, 4]]
```

`Elasticity(self,v=None)`
`OmegaPrimality(self,v=None)`
`CatenaryDegree(self,v=None)`
`DeltaSet(self,v)`
Member functions for semigroup theoretic functions. See for precise definitions.

```
sage: A = AffineSemigroup([[5, 1], [1, 3], [2, 5]])
sage: A.Elasticity()
12/7
sage: A.Elasticity([10,25])
1.0
sage: A.OmegaPrimality()
24
sage: A.OmegaPrimality([10,25])
24
sage: A.CatenaryDegree()
```

```
24
sage: A.CatenaryDegree([10,25])
0
sage: A.DeltaSet([10,25])
[]
```

3.2. **The `Cone` Class.** An object class that supports the functionalities of cones, specifically focused on member functions to give more insight into the polynomials that each cone restricts to.

`Cone(self,basepoint,gens,semigroup,degree=None,func=None)`
This object constructor initializes a Cone object by taking in the basepoint, `basepoint`, generator list, `gens`, relevant to the specified semigroup, `semigroup`, with the option to specify a degree, `degree`, or unique function, `func`, to calculate the necessary polynomial.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: cone = Cone([0,0],[[1,3],[3,3]],A)
sage: cone.basepoint
[0, 0]
sage: cone.gens
[[1, 3], [3, 3]]
```

`InitDegree(self,coeff)`
A helper function that runs internally in the constructor. Defines the matrix of the polynomial function defined with the given degree,`self.degree`, and function, `self.func`, specified initially by the user in the Cone construction.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: cone = Cone([0,0],[[1,3],[3,3]],A,2)
sage: cone.matrix
[ 0  0  0  0  0  1]
[ 1  3  9  1  3  1]
[ 9  9  9  3  3  1]
[ 4 12 36  2  6  1]
[16 24 36  4  6  1]
[36 36 36  6  6  1]
sage: cone = Cone([0,0],[[1,3],[3,3]],A,3)
sage: cone.matrix
[  0   0   0   0   0   0   0   0   0   1]
[  1   3   9  27   1   3   9   1   3   1]
[ 27  27  27  27   9   9   9   3   3   1]
[  8  24  72 216   4  12  36   2   6   1]
[ 64  96 144 216  16  24  36   4   6   1]
```

```
[216 216 216 216  36  36  36   6   6   1]
[343 441 567 729  49  63  81   7   9   1]
[729 729 729 729  81  81  81   9   9   1]
```

`DrawCone(self,plot,xmax,ymax,color)`
`HighlightCone(self,plot,xmax,ymax,color)`
`PlotCone(self,plot,xmax,ymax,color)`
Member functions that returns a plot of the semigroup with the elements of the cone up to [xmax,ymax] in the color, `color`, specified.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: plot = A.Plot(15,15)
sage: cone = Cone([0,0],[[1,3],[3,3]],A)
sage: cone.DrawCone(plot,15,15,"magenta")
sage: cone.HighlightCone(plot,15,15,"magenta")
sage: cone.PlotCone(plot,15,15,"magenta")
```

Refer to Figure 6.

`ContainsPoints(self,point)`
A member function that returns `True` if `point` is a member of the cone, `False` otherwise.

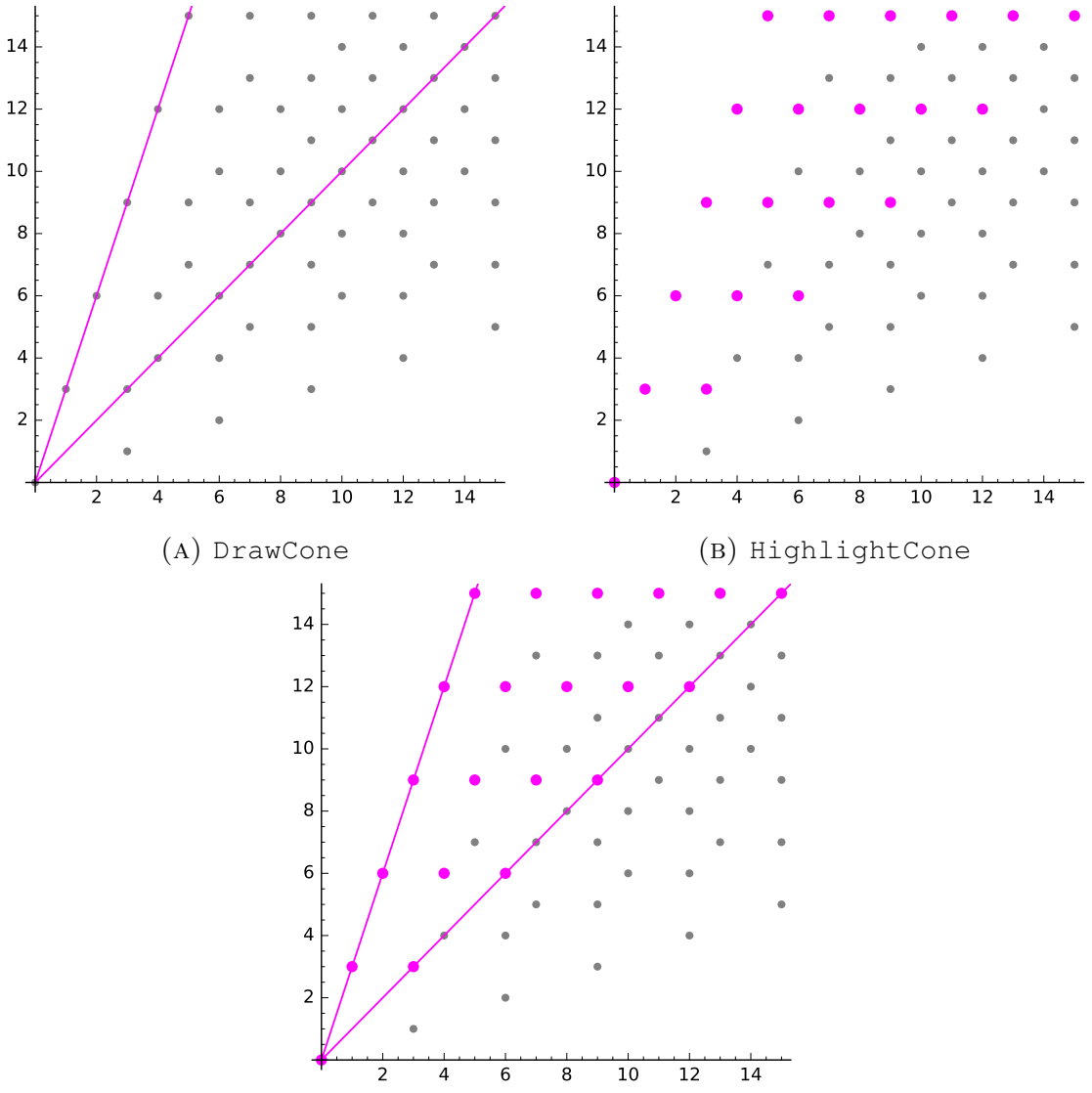
```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: cone = Cone([0,0],[[1,3],[3,3]],A,1)
sage: cone.ContainsPoint([2,2])
False
sage: cone.ContainsPoint([9,9])
True
sage: cone.ContainsPoint([100,100])
False
sage: cone.ContainsPoint([99,99])
True
```

4. The ConeDecomposition Class

The main functionality of this `ConeDecomposition` class is to provide an automated process to decompose affine semigroups. Using the member function

`Decomposition()`,

the user can recursively define cones, by their basepoints and generators, in order to construct a semigroup decomposition. This function returns a list of cones of the respective decomposition.



(A) `DrawCone` (B) `HighlightCone`

(C) `PlotCone`

FIGURE 6

The object constructor,

```
ConeDecomposition(self,semigroup,xmax,ymax,func=None),
```

takes in the maximum coordinates, `xmax` and `ymax`, for the corresponding semigroup, `semigroup`, with the option to specify a function, `func`, to calculate the necessary polynomial for decomposition.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: disjoint =  ConeDecomposition(A,30,30)
```

In order to aid the decomposition process, the function first prints the minimal presentation and elements of the minimal presentation of the underlying affine semigroup.

```
sage: conelist = disjoint.Decomposition(30,30)
Beginning Disjoint Decomposition for Affine Semigroup generated by [[1,
    3], [3, 1], [3, 3]]
Minimal Presentation = [[[3, 3, 0], [0, 0, 4]]] @ [[12, 12]]
```

At each iteration of defining a cone, the function gives a list of suggested generators, using the original generators of the affine semigroup. Cone definition is done by running the cone object previously defined. Furthermore, the function gives suggested basepoints, based off of the remaining points in the function, utilizing the `RemainingPts` function.

```
Defining Cone 1:
        Cone generators Options-  0:[1, 3]  1:[3, 1]  2:[3, 3]
        Please enter generators:[0,1]
        Set Basepoint: (enter vector or 1 to keep set @ [0, 0])  1
        Preview -- Cone generate by [[1, 3], [3, 1]] with basepoint @ [0,
            0]
Launched png viewer for Graphics object consisting of 180 graphics
   primitives
        *** Warning -- Cone Overlap @ []
        0:Keep Cone / 1:Remove Current Cone / 2:Redefine Previous Cone(s)
            / 3: Add Cone & End Program / 4: End Program
        Choose option: 1

Defining Cone 1:
        Cone generators Options-  0:[1, 3]  1:[3, 1]  2:[3, 3]
        Please enter generators:[0,1]
        Set Basepoint: (enter vector or 1 to keep set @ [0, 0])  [3,3]
        Preview -- Cone generate by [[1, 3], [3, 1]] with basepoint @ [3,
            3]
Launched png viewer for Graphics object consisting of 180 graphics
   primitives
        *** Warning -- Cone Overlap @ []
        0:Keep Cone / 1:Remove Current Cone / 2:Redefine Previous Cone(s)
            / 3: Add Cone & End Program / 4: End Program
        Choose option: 3
        Finished with 1 cone.
```

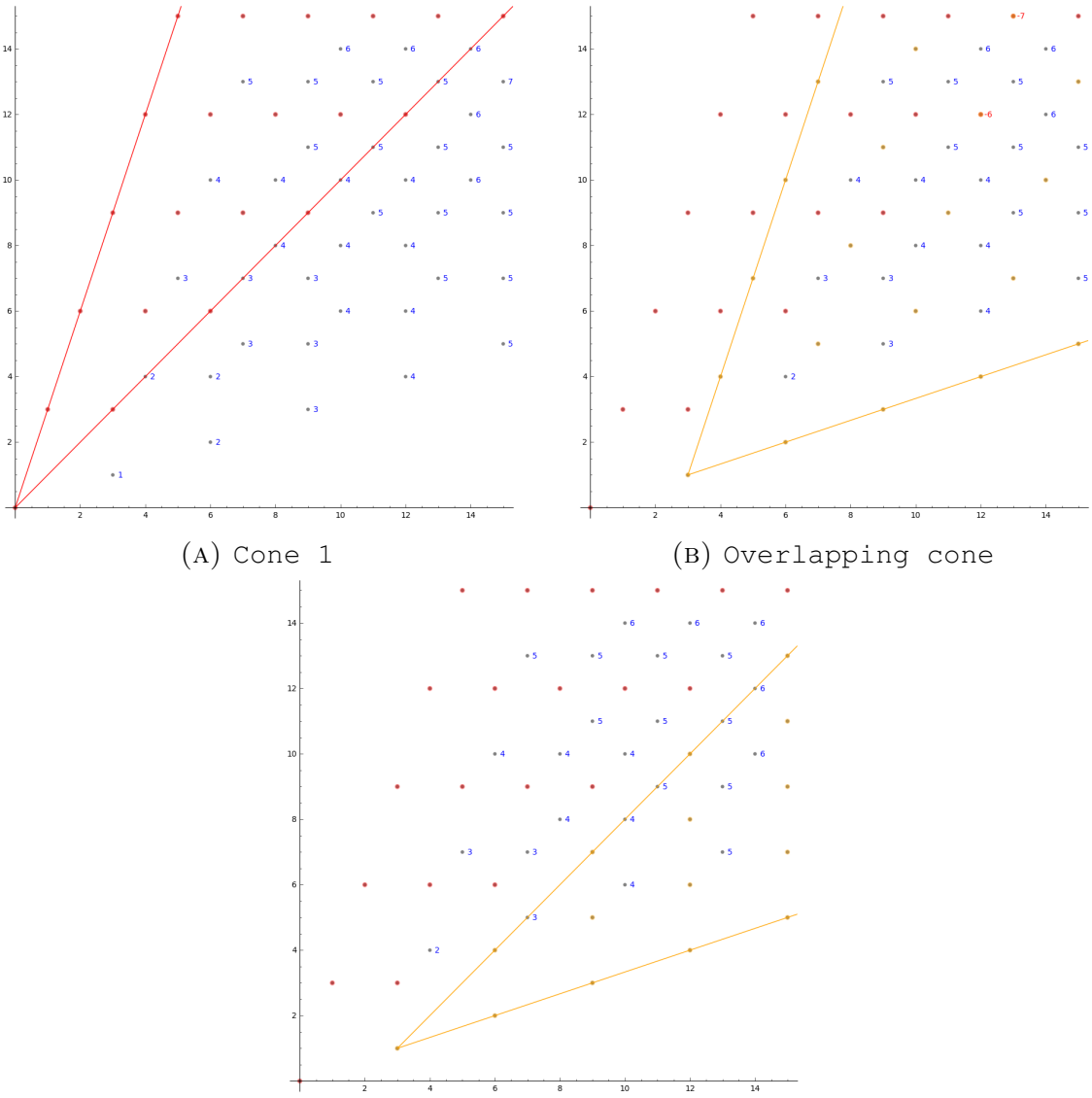
The `Decomposition()` function also supports the input of unique functions.

```
sage: maxFact = ConeDecomposition(A,15,15,MaxFact)
sage: conelist =maxFact.Decomposition(15,15)
```

```
Begining Decomposition for Affine Semigroup generate by [[1, 3], [3, 1],
   [3, 3]]
Minimal Presentation = [[[3, 3, 0], [0, 0, 4]]] @ [[12, 12]]

Defining Cone 1:
        Cone generators Options-  0:[1, 3]  1:[3, 1]  2:[3, 3]
        Please enter generators:[0,2]
        Set Basepoint: (enter vector or 1 to keep set @ [0, 0])  1
        Preview -- Cone generate by [[1, 3], [3, 3]] with basepoint @ [0,
            0]
Launched png viewer for Graphics object consisting of 204 graphics
   primitives
        0:Keep Cone / 1:Remove Current Cone / 2:Redefine Previous Cone(s)
            / 3: Add Cone & End Program / 4: End Program
        Choose option: 0
        Cone generate by [[1, 3], [3, 3]] with basepoint @ [0, 0]
```

With each iteration of defining a new cone, the user has the option to: 0) keep the current cone, 1) remove the current cone, 2) redefine the current cone, 3) add the current cone to the list and end the program, 4) end the program. By providing a plot preview of the current decomposition process, this gives the user a visual insight to where they should place the next cone. In addition, with the given options at each cone iteration, this interface gives the user flexibility to redefine previous cones.

```
Defining Cone 2:
        Cone generators Options-  0:[1, 3]  1:[3, 1]  2:[3, 3]
        Please enter generators:[0,1]
        Set Basepoint: (enter vector or 1 to keep set @ [3, 1])  1
        Preview -- Cone generate by [[1, 3], [3, 1]] with basepoint @ [3,
            1]
Launched png viewer for Graphics object consisting of 148 graphics
   primitives
        *** Warning -- Cone Overlap @ [[12, 12], [13, 15], [14, 18], [15,
            21], [16, 24], [17, 27], [18, 30], [24, 24], [25, 27], [26,
           30]]
        0:Keep Cone / 1:Remove Current Cone / 2:Redefine Previous Cone(s)
            / 3: Add Cone & End Program / 4: End Program
        Choose option: 1

Defining Cone 2:
        Cone generators Options-  0:[1, 3]  1:[3, 1]  2:[3, 3]
        Please enter generators:[1,2]
        Set Basepoint: (enter vector or 1 to keep set @ [3, 1])  1
        Preview -- Cone generate by [[3, 1], [3, 3]] with basepoint @ [3,
            1]
```



(A) Cone 1

(B) Overlapping cone

(C) Cone 2

FIGURE 7

```
Launched png viewer for Graphics object consisting of 150 graphics
   primitives
        0:Keep Cone / 1:Remove Current Cone / 2:Redefine Previous Cone(s)
            / 3: Add Cone & End Program / 4: End Program
        Choose option: 0
```

Refer to Figure 7.

5. Data Results for 3 Generated Affine Semigroups

In our efforts to determine the leading coefficients of the quasipolynomials of Affine Semigroups, we began by restricting our focus to decomposing Affine Semigroups generated by 3 generators over the constant 1 function. Utilizing the Affine Semigroup packaged for decomposition, we made the following observations:

Example 5.1. Let A be an Affine Semigroup such that $A = \langle (1,3),(3,1),(3,3) \rangle$. The minimal presentation is

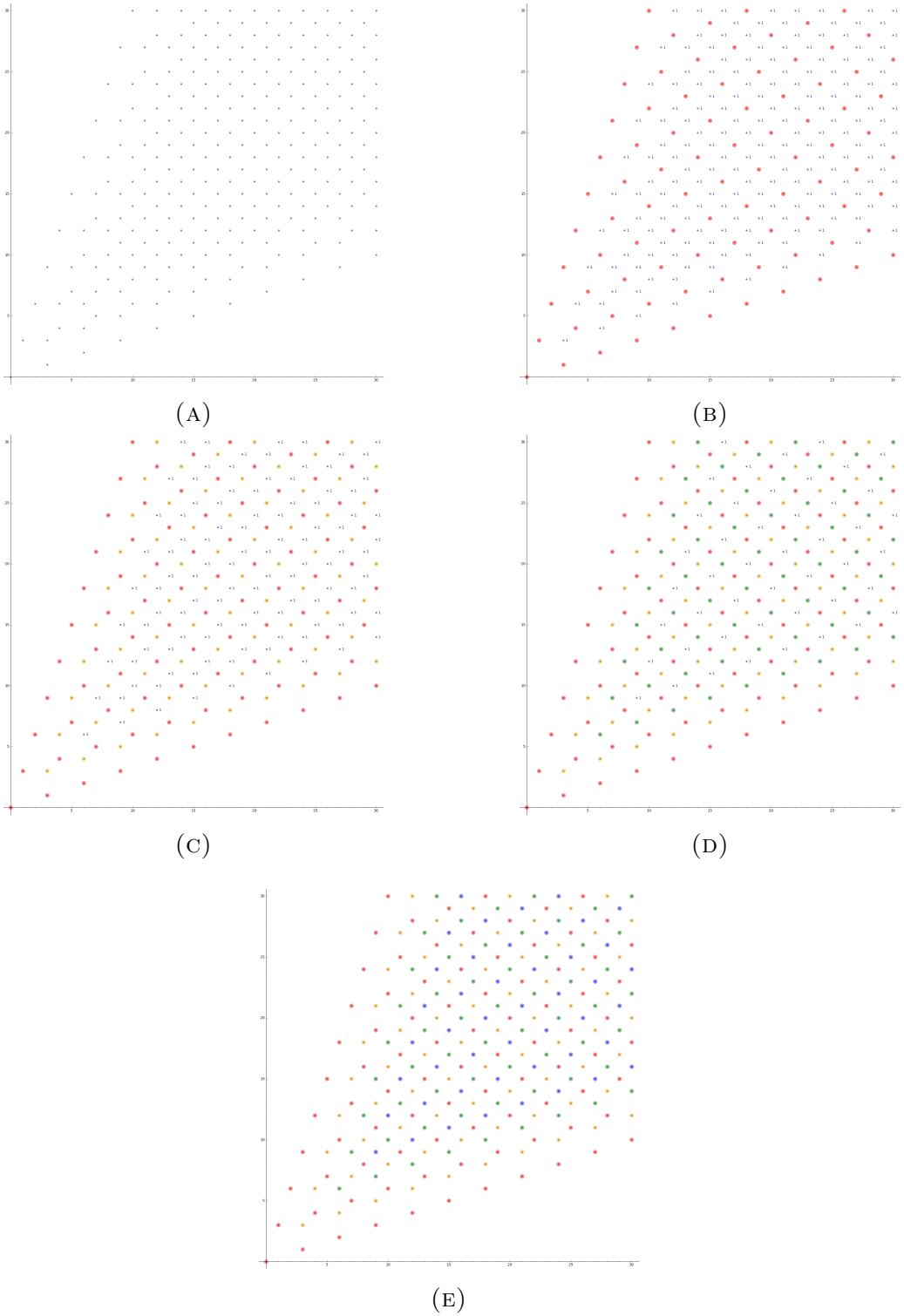
$$MinPres(A) = \{(3,3,0),(0,0,4)\}.$$

We can decompose the semigroup using only the middle generator to achieve a decomposition utilizing 4 cones generated by $(1,3)$ and $(3,1)$ with a translation factors of $(3,3)$. Refer to Figure 8.

```
sage: A = AffineSemigroup([[1,3],[3,3],[3,1]])
sage: decomposeA = ConeDecomposition(A,30,30)
sage: disjointA_1 = decomposeA.Decomposition(30,30)
Beginning Decomposition for Affine Semigroup generate by [[1, 3], [3, 1],
    [3, 3]]
Minimal Presentation = [[[3, 3, 0], [0, 0, 4]]] @ [[12, 12]]
...
Finished with 4 cones
sage: disjointA_1

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 1]] with
   polynomial function 0x + 0y + 1>,
 <Cone with basepoint [3, 3] generated by [[1, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [6, 6] generated by [[1, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [9, 9] generated by [[1, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>]
```

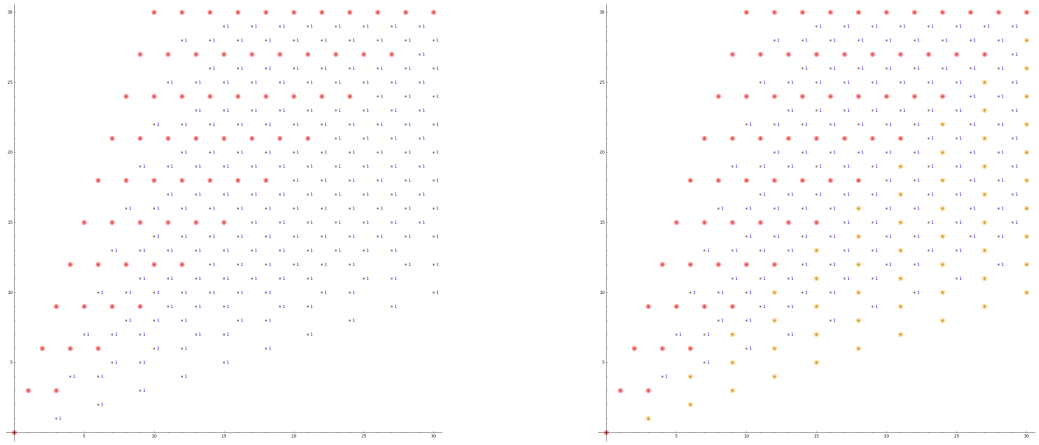
We can also decompose the semigroup using only the two outer generators to obtain a decomposition with 6 cones: 3 cones generated by $(1,3)$ and $(3,3)$ with a translation factor of $(1,3)$ and 3 cones generated by $(3,3)$ and $(3,1)$ with a translation factor of $(3,1)$. Refer to Figure 9.

```
sage: disjointA_2 = A_decomp.Decomposition(30,30)
Beginning Decomposition for Affine Semigroup generate by [[1, 3], [3, 1],
    [3, 3]]
Minimal Presentation = [[[3, 3, 0], [0, 0, 4]]] @ [[12, 12]]
...
Finished with 6 cones
sage: disjointA_2
```



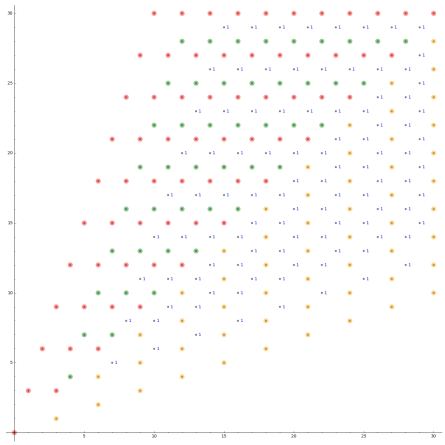
(A) (B) (C) (D) (E)

FIGURE 8. Example A Decomposition 1

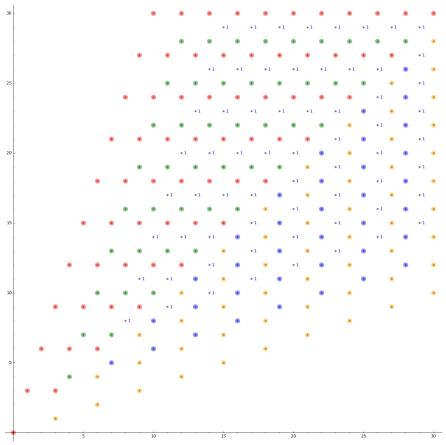


(A)

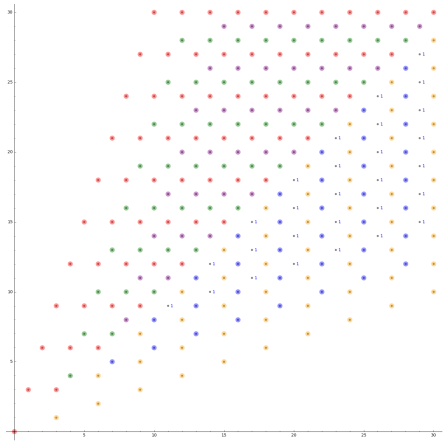
(B)



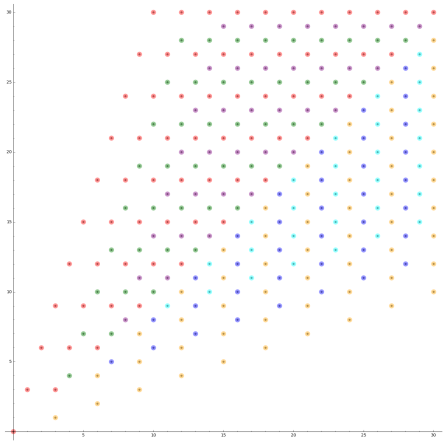
(C)



(D)



(E)



(F)

FIGURE 9. Example A Decomposition 2

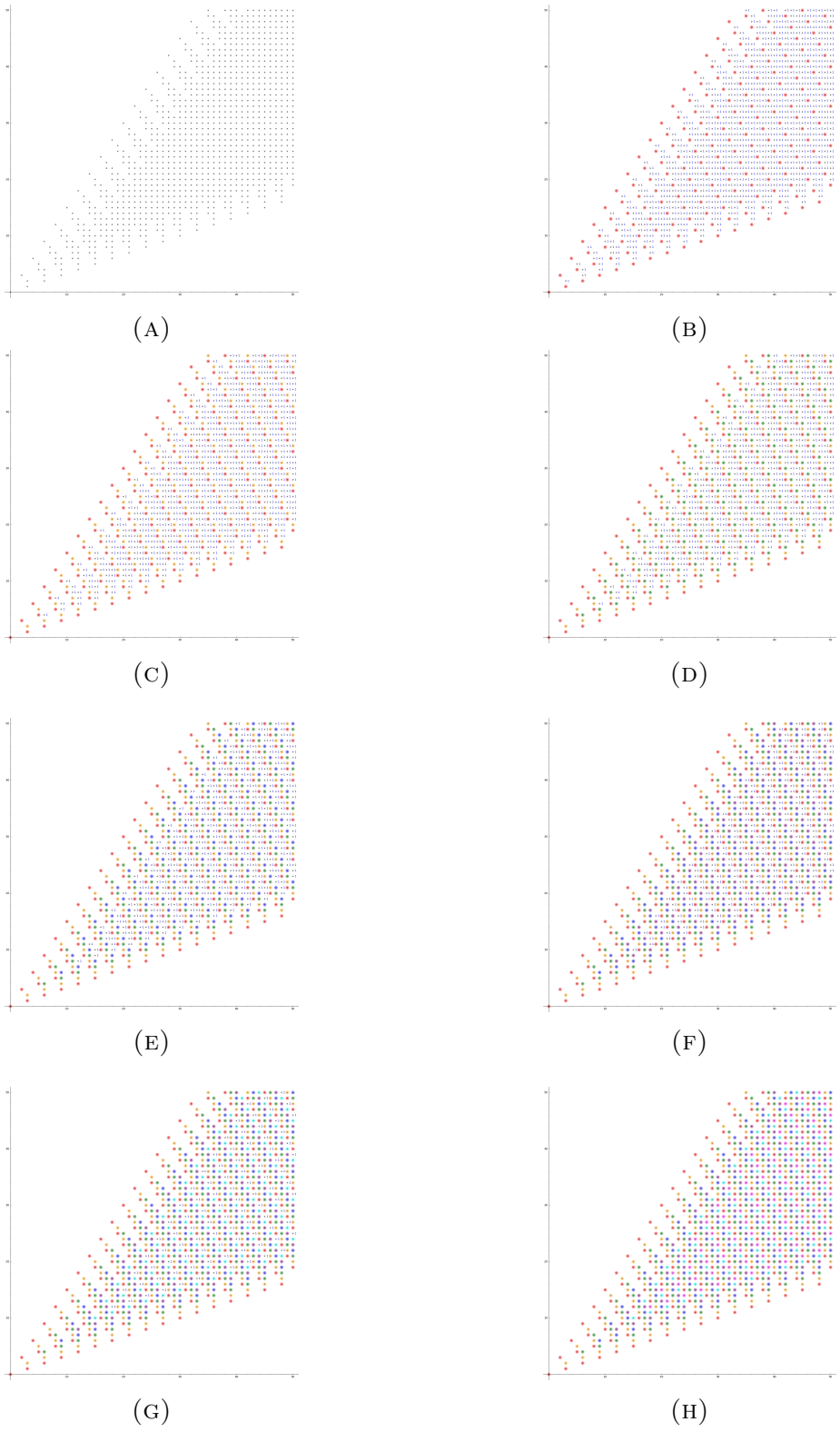
```
[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 3]] with
   polynomial function 0x + 0y + 1>,
 <Cone with basepoint [3, 1] generated by [[1, 3], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [6, 2] generated by [[1, 3], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [9, 3] generated by [[3, 1], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [10, 6] generated by [[3, 1], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [11, 9] generated by [[3, 1], [3, 3]] with
    polynomial function 0x + 0y + 1>]
```

Example 5.2. Let $B = \langle (2,3), (3,1), (3,2) \rangle$. The minimal presentation is

$$MinPres(B) = \{(3,5,0), (0,0,7)\}.$$

We can decompose the semigroup using only the middle generator to achieve a decomposition utilizing 7 cones generated by $(2,3)$ and $(3,1)$ with a translation factor of $(3,2)$. Refer to Figure 10.

```
sage: B = AffineSemigroup([[2,3],[3,1],[3,2]])
sage: decomposeB = ConeDecomposition(B,30,30)
sage: disjoint_B_1 = decomposeB.Decomposition(30,30)
Beginning Decomposition for Affine Semigroup generate by [[2, 3], [3, 1],
    [3, 2]]
Minimal Presentation = [[[3, 5, 0], [0, 0, 7]]] @ [[21, 14]]
...
Finished with 7 cones
sage: disjoint_B_1

[<Cone with basepoint [0, 0] generated by [[2, 3], [3, 1]] with
   polynomial function 0x + 0y + 1>,
 <Cone with basepoint [3, 2] generated by [[2, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [6, 4] generated by [[2, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [9, 6] generated by [[2, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [12, 8] generated by [[2, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [15, 10] generated by [[2, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [18, 12] generated by [[2, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>]
```



(A) (B) (C) (D) (E) (F) (G) (H)

FIGURE 10. Example B Decomposition 1

We can also decompose the semigroup using only the two outer generators to obtain a decomposition with 8 cones: 3 cones generated by $(3,1)$ and $(3,2)$ with a translation factor of $(2,3)$ and 5 cones generated by $(2,3)$ and $(3,2)$ with a translation factor of $(3,1)$. Refer to Figure 11.

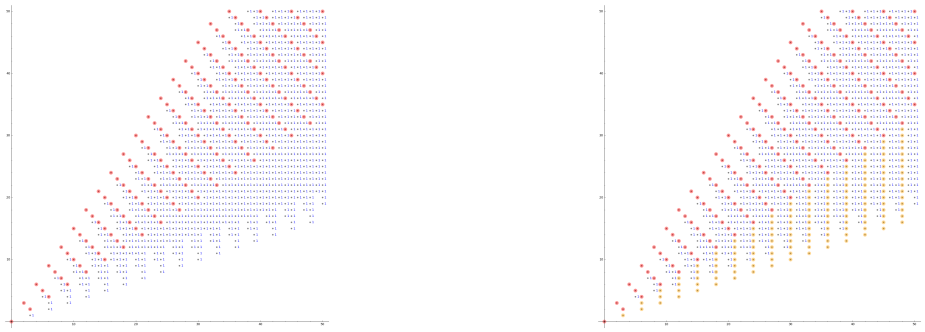
```
sage: disjoint_B_2 = decomposeB.Decomposition(50,50)
Beginning Decomposition for Affine Semigroup generate by [[2, 3], [3, 1],
    [3, 2]]
Minimal Presentation = [[[3, 5, 0], [0, 0, 7]]] @ [[21, 14]]
...
Finished with 8 cones
sage: disjoint_B_2

[<Cone with basepoint [0, 0] generated by [[2, 3], [3, 2]] with
   polynomial function 0x + 0y + 1>,
 <Cone with basepoint [3, 1] generated by [[2, 3], [3, 2]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [6, 2] generated by [[2, 3], [3, 2]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [9, 3] generated by [[2, 3], [3, 2]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [12, 4] generated by [[2, 3], [3, 2]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [15, 5] generated by [[3, 1], [3, 2]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [17, 8] generated by [[3, 1], [3, 2]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [19, 11] generated by [[3, 1], [3, 2]] with
    polynomial function 0x + 0y + 1>]
```

6. Conjectures and Further Data

Upon further examination into the observations made, we noticed an overlying pattern for the 3 generated affine semigroups. We will examine the decomposition for number of factorization, minimum factorization length, and maximum factorization length.

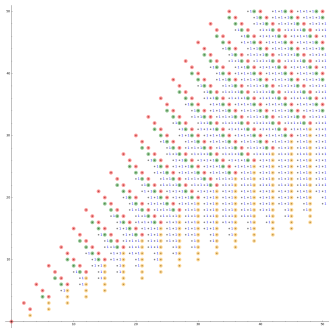
Recall we define $A = \langle (1,3), (3,1), (3,3) \rangle$. In the following decompositions, we follow the form defined in **Claim 5.2**. We can see by these data results that for the 3 generated Affine Semigroups, that the coefficients for the linear terms are constant across all of the cones in the respective decomposition.

```
numFact = ConeDecomposition(A,25,25,NumFact)
```

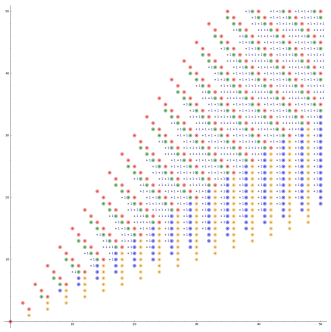


(A)

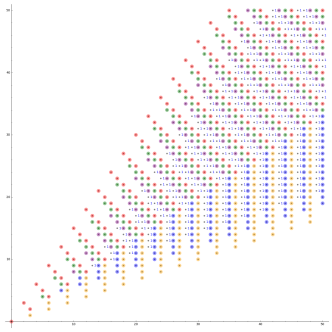
(B)



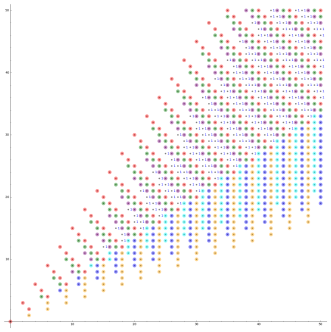
(C)



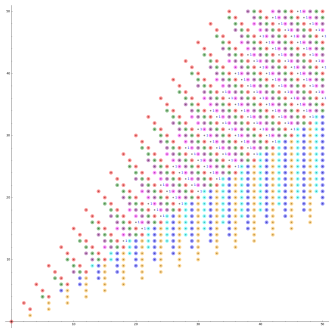
(D)



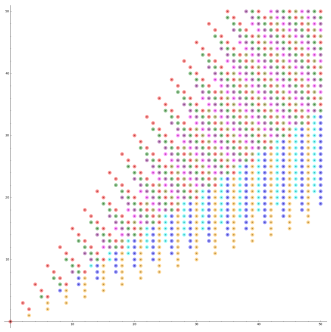
(E)



(F)



(G)



(H)

FIGURE 11. Example B Decomposition 2

```
sage: numFact_1 = numFact.Decomposition(25,25)
sage: numFact_1

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 1]] with
   polynomial function 0x + 0y + 1>,
 <Cone with basepoint [3, 3] generated by [[1, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [6, 6] generated by [[1, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [9, 9] generated by [[1, 3], [3, 1]] with
    polynomial function 0x + 0y + 1>]
```

Refer to Figure 12.

```
sage: numFact_2 = numFact.Decomposition(25,25)
sage: numFact_2

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 3]] with
   polynomial function 0x + 0y + 1>,
 <Cone with basepoint [3, 1] generated by [[3, 1], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [4, 4] generated by [[1, 3], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [7, 5] generated by [[3, 1], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [8, 8] generated by [[1, 3], [3, 3]] with
    polynomial function 0x + 0y + 1>,
 <Cone with basepoint [11, 9] generated by [[3, 1], [3, 3]] with
    polynomial function 0x + 0y + 1>]
```

Refer to Figure 13.

Minimum Factorization Length.

```
minFact = ConeDecomposition(A,25,25,MinFact)
```

```
sage: minFact_1 = minFact.Decomposition(25,25)
sage: minFact_1

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 1]] with
   polynomial function 1/4x + 1/4y + 0>,
 <Cone with basepoint [3, 3] generated by [[1, 3], [3, 1]] with
    polynomial function 1/4x + 1/4y -1/2>,
 <Cone with basepoint [6, 6] generated by [[1, 3], [3, 1]] with
    polynomial function 1/4x + 1/4y -1>,
```

```
 <Cone with basepoint [9, 9] generated by [[1, 3], [3, 1]] with
    polynomial function 1/4x + 1/4y -3/2>]
```

Refer to Figure 14.

```
sage: minFact_2 = minFact.Decomposition(25,25)
sage: minFact_2

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 3]] with
   polynomial function 0x + 1/3y + 0>,
 <Cone with basepoint [3, 1] generated by [[3, 1], [3, 3]] with
    polynomial function 1/3x + 0y + 0>,
 <Cone with basepoint [4, 4] generated by [[1, 3], [3, 3]] with
    polynomial function 0x + 1/3y + 2/3>,
 <Cone with basepoint [7, 5] generated by [[3, 1], [3, 3]] with
    polynomial function 1/3x + 0y + 2/3>,
 <Cone with basepoint [8, 8] generated by [[3, 1], [3, 3]] with
    polynomial function 1/3x + 0y + 4/3>,
 <Cone with basepoint [9, 11] generated by [[1, 3], [3, 3]] with
    polynomial function 0x + 1/3y + 4/3>]
```

Refer to Figure 15.

Maximum Factorization Length.

```
maxFact = ConeDecomposition(A,25,25,MaxFact)
```

```
sage: maxFact_1 = maxFact.Decomposition(25,25)
sage: maxFact_1

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 1]] with
   polynomial function 1/4x + 1/4y + 0>,
 <Cone with basepoint [3, 3] generated by [[1, 3], [3, 1]] with
    polynomial function 1/4x + 1/4y -1/2>,
 <Cone with basepoint [6, 6] generated by [[1, 3], [3, 1]] with
    polynomial function 1/4x + 1/4y -1>,
 <Cone with basepoint [9, 9] generated by [[1, 3], [3, 1]] with
    polynomial function 1/4x + 1/4y -3/2>]
```

Refer to Figure 16.

```
sage: maxFact_2 = maxFact.Decomposition(25,25)
sage: maxFact_2

[<Cone with basepoint [0, 0] generated by [[1, 3], [3, 3]] with
   polynomial function 0x + 1/3y + 0>,
 <Cone with basepoint [3, 1] generated by [[3, 1], [3, 3]] with
    polynomial function 1/3x + 0y + 0>,
```

```
<Cone with basepoint [4, 4] generated by [[1, 3], [3, 3]] with
   polynomial function 0x + 1/3y + 2/3>,
<Cone with basepoint [7, 5] generated by [[3, 1], [3, 3]] with
   polynomial function 1/3x + 0y + 2/3>,
<Cone with basepoint [8, 8] generated by [[3, 1], [3, 3]] with
   polynomial function 1/3x + 0y + 4/3>,
<Cone with basepoint [9, 11] generated by [[1, 3], [3, 3]] with
   polynomial function 0x + 1/3y + 4/3>]
```

Refer to Figure 17.

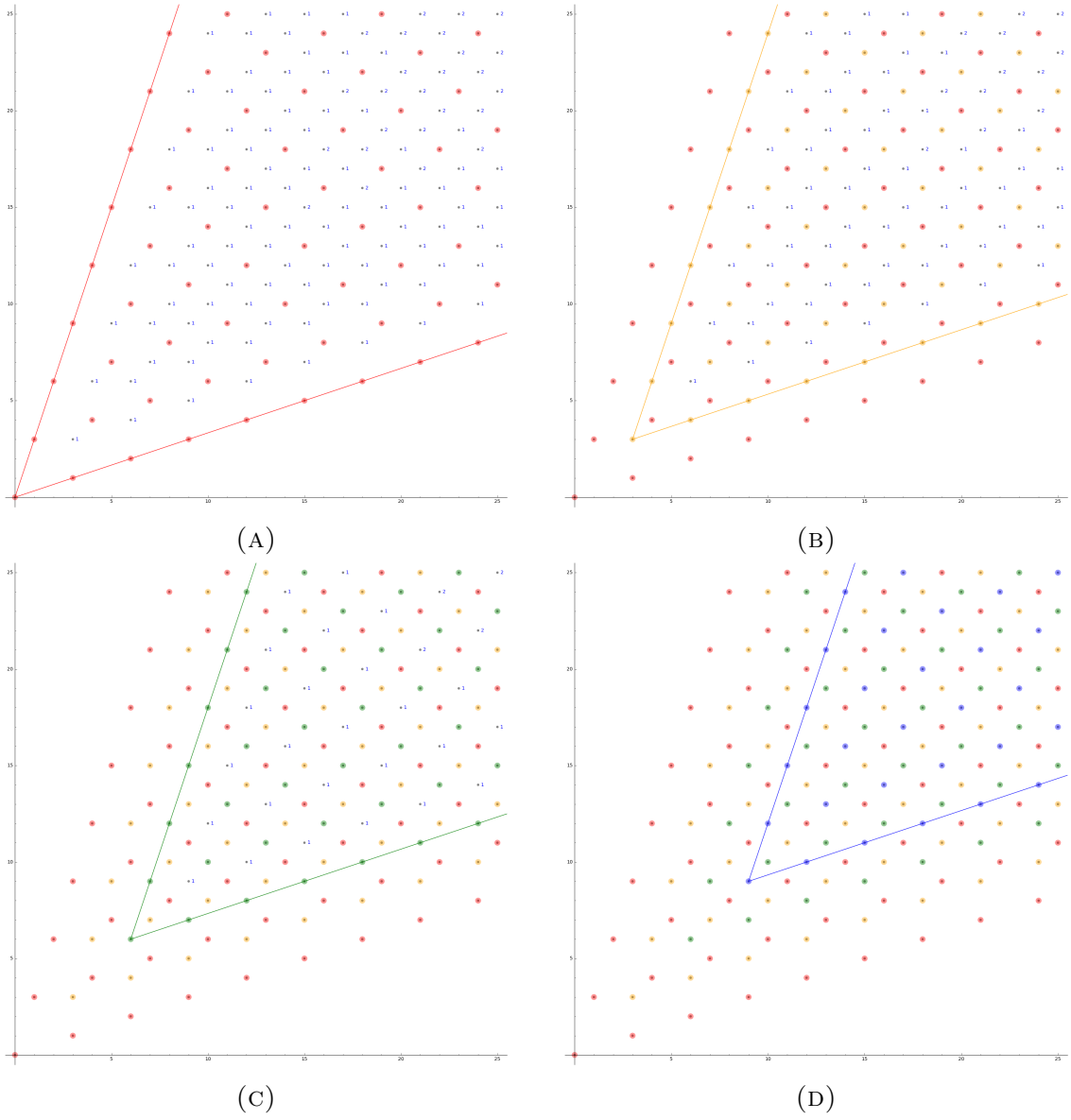
References

[1] J. Rosales and P. García-Sánchez, *Numerical semigroups*, Developments in Mathematics, Vol. 20, Springer-Verlag, New York, 2009.

[2] C. O'Neill, *On factorization invariants and Hilbert functions*, Journal of Pure and Applied Algebra **221** (2017), no. 12, 3069–3088. Available at arXiv:math.AC/1503.08351.

[3] M. Delgado, P. García-Sánchez, J. Morais, *NumericalSgps, A package for numerical semigroups*, Version 0.980 dev (2013), (GAP package), `http://www.fc.up.pt/cmup/mdelgado/numericalsgps/`.

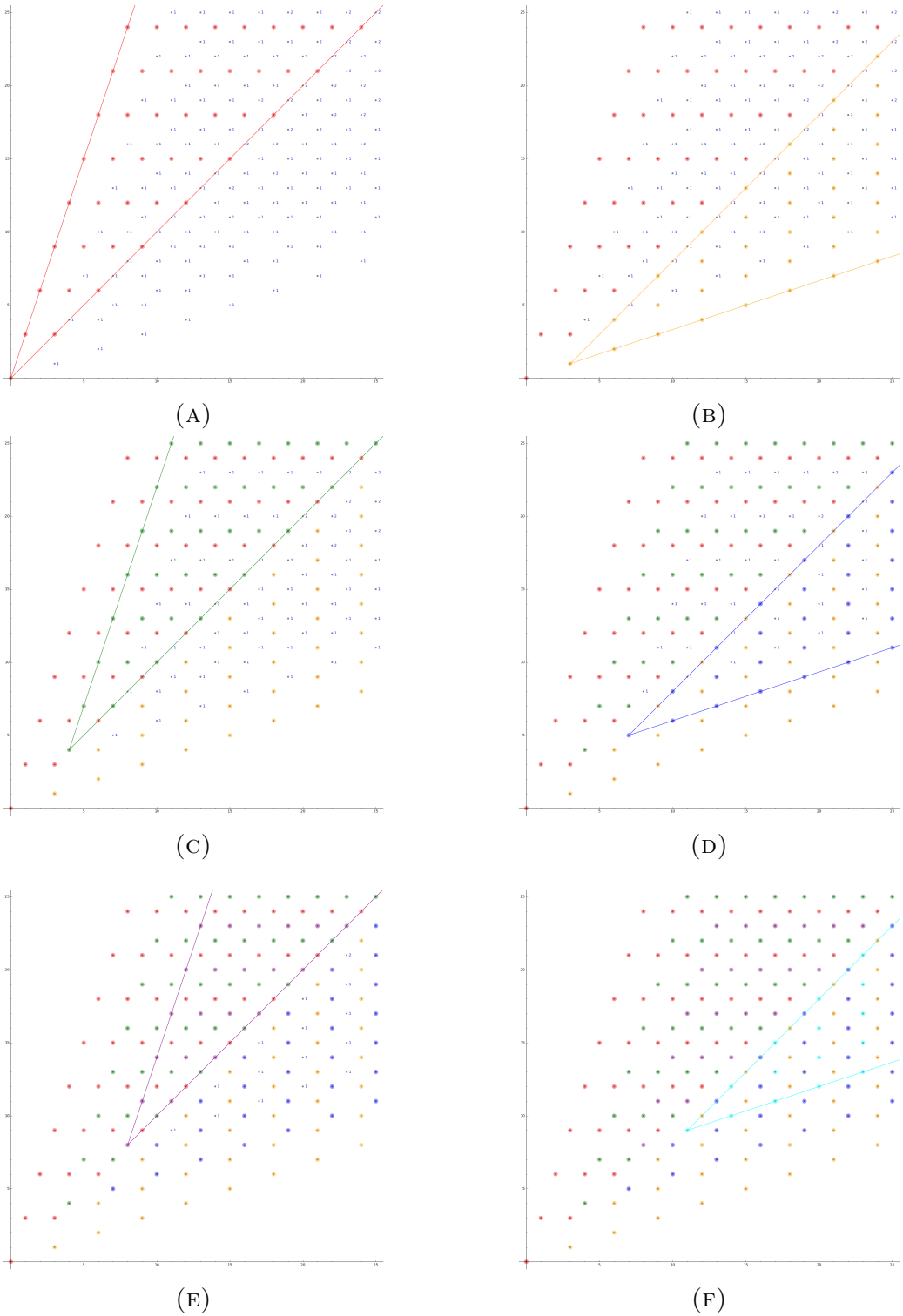
Mathematics Department, University of California Davis, Davis, CA 95616
E-mail address: `macchen@ucdavis.edu`



(A) (B)

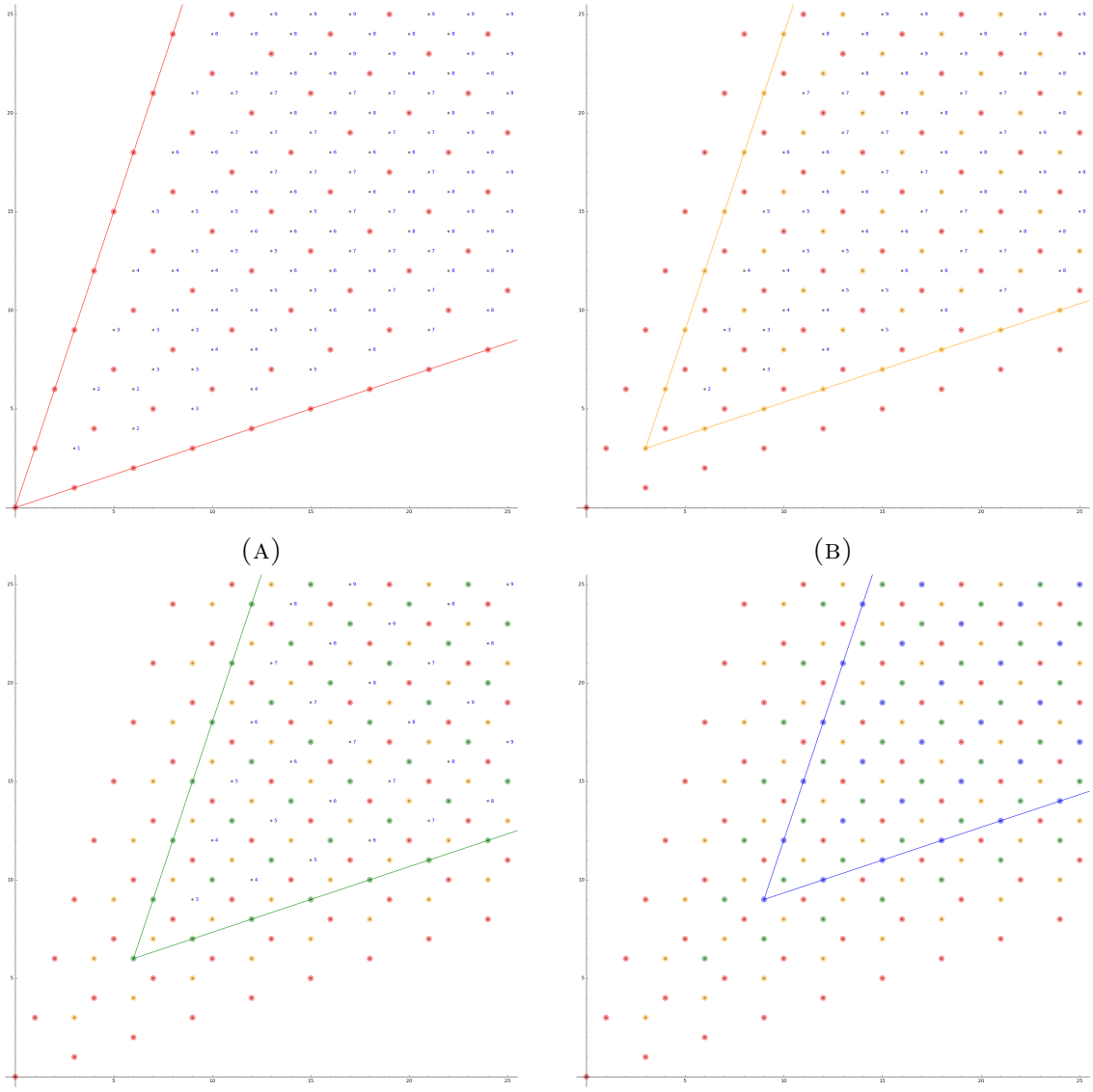
(C) (D)

FIGURE 12. Number of factorizations decomposition 1



(A) (B) (C) (D) (E) (F)

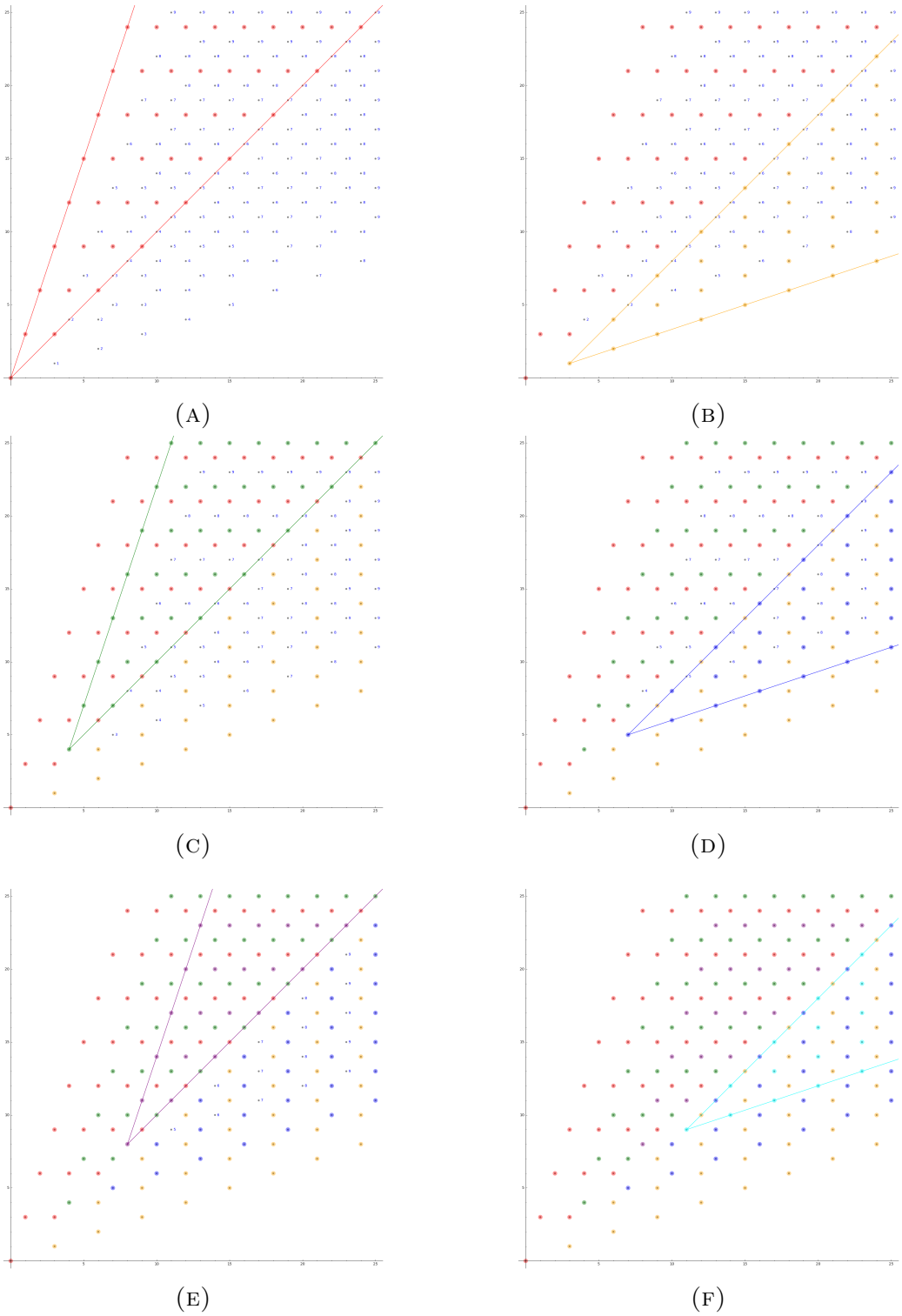
FIGURE 13. Number of factorizations decomposition 2



(A) (B)

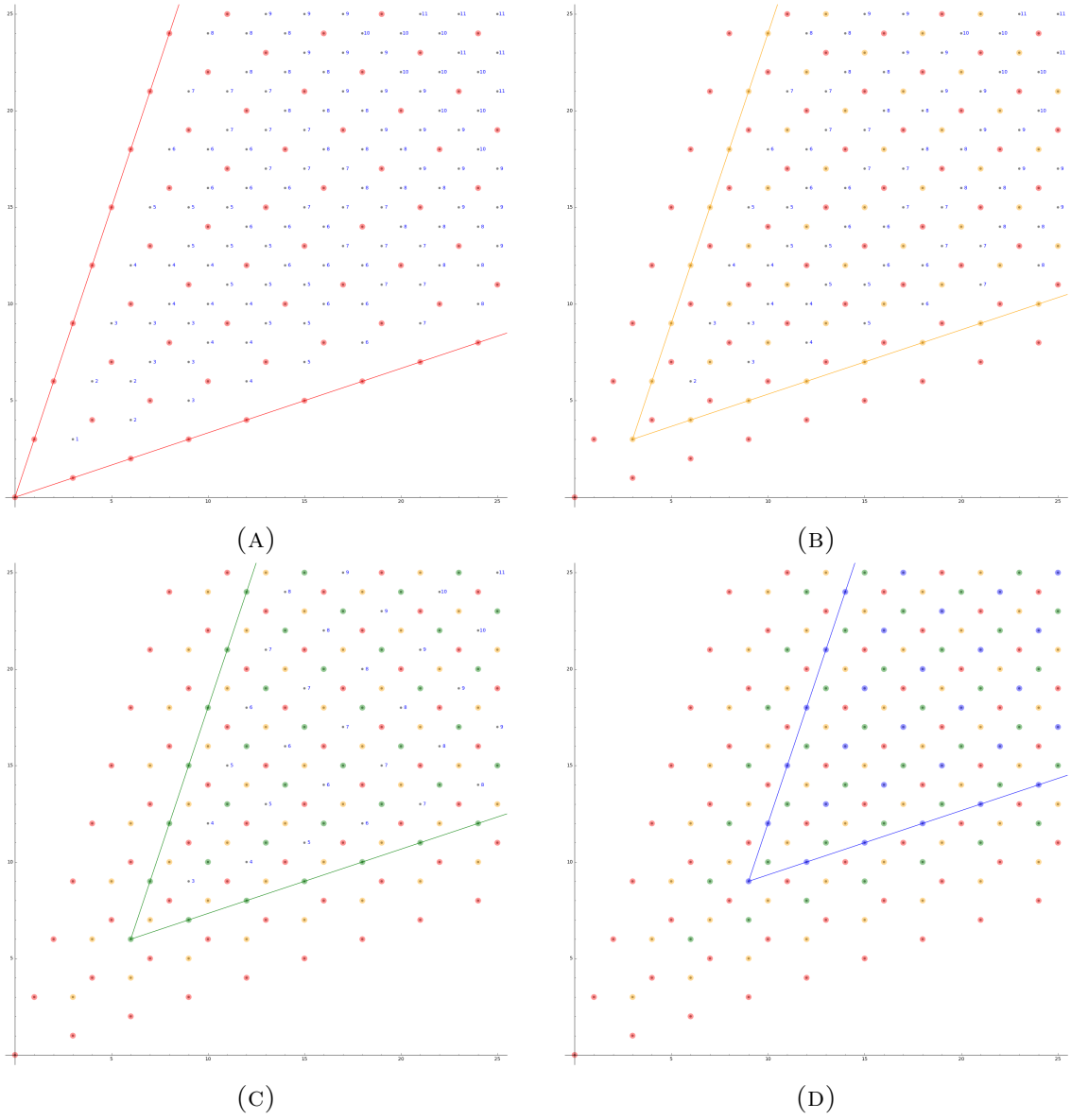
(C) (D)

FIGURE 14. Minimum factorization length decomposition 1



(A) (B) (C) (D) (E) (F)

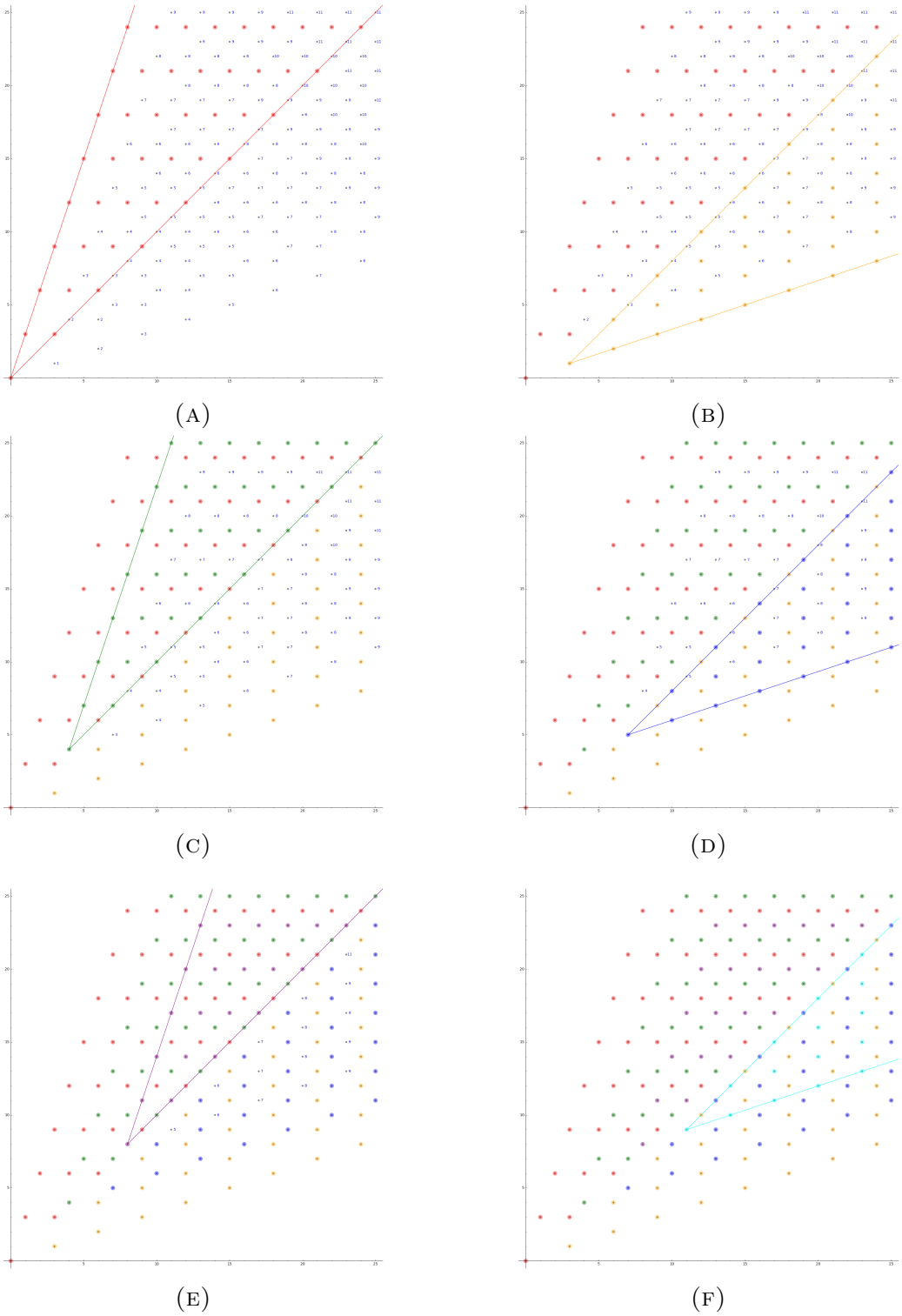
FIGURE 15. Minimum factorization length decomposition 2



(A) (B)

(C) (D)

FIGURE 16. Maximum factorization length decomposition 1



(A) (B) (C) (D) (E) (F)

FIGURE 17. Maximum factorization length decomposition 2