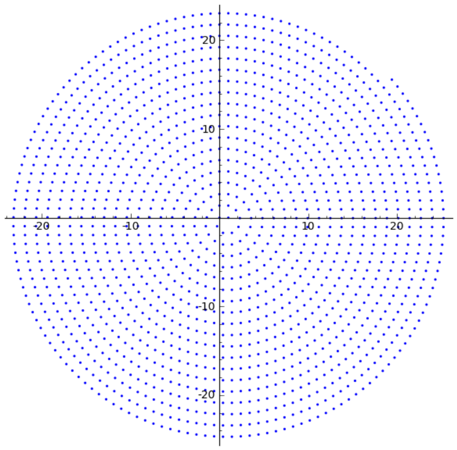
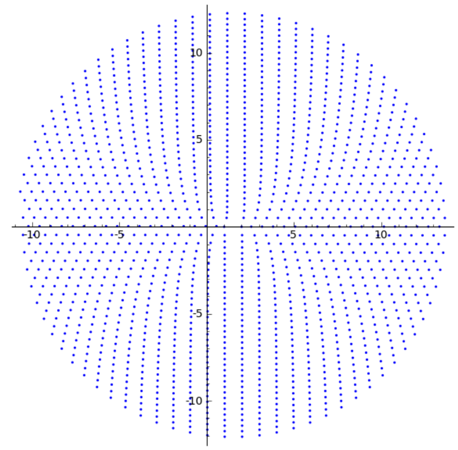
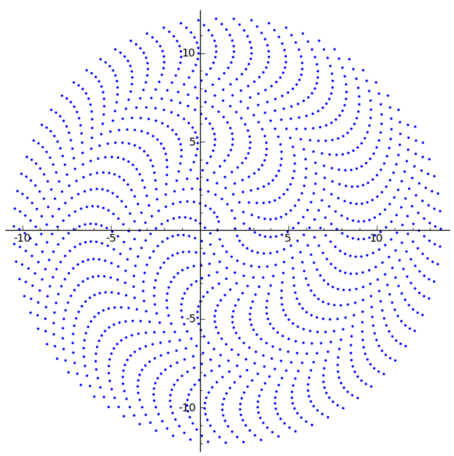
Figure 1. Partial sums of $\sum e\big(\alpha\sqrt{n}\big)$, $n < 1500$. Notation: $e(x) = e^{2\pi i x}$.



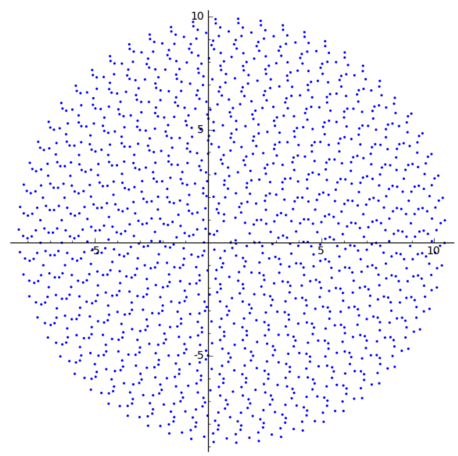
$\alpha = 1/2$



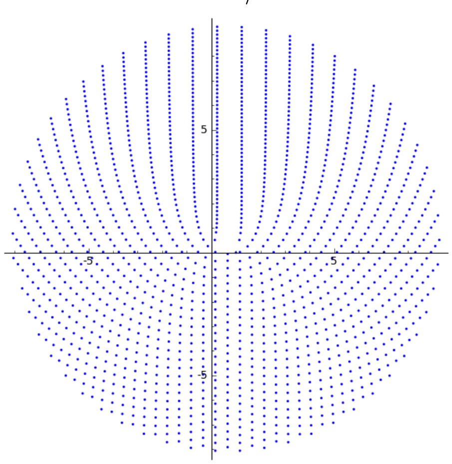
$\alpha = 1$



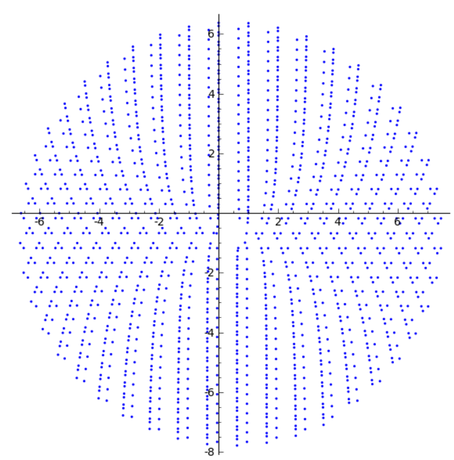
$\alpha = 65/64$



$\alpha = 1.3$

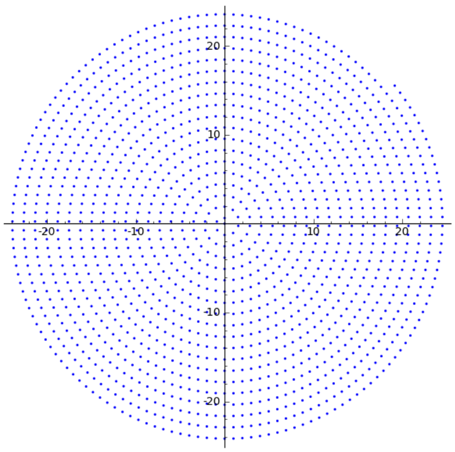


$\alpha = \sqrt{2}$

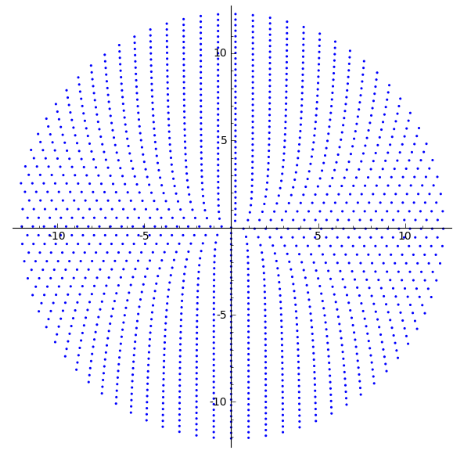


$\alpha = \sqrt{3}$

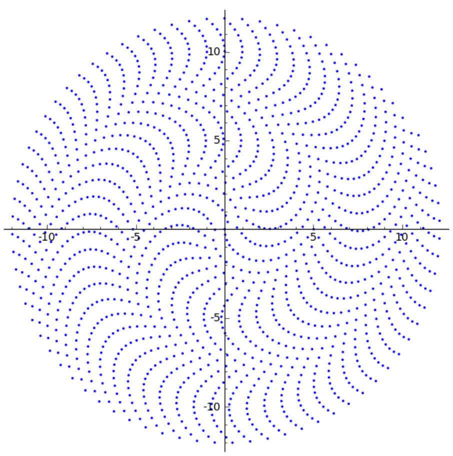
Figure 2. $\left\{\frac{m}{\pi\alpha} e(\alpha\sqrt{m} - 1/4)\right\}_{0 \le m < 1500}$.



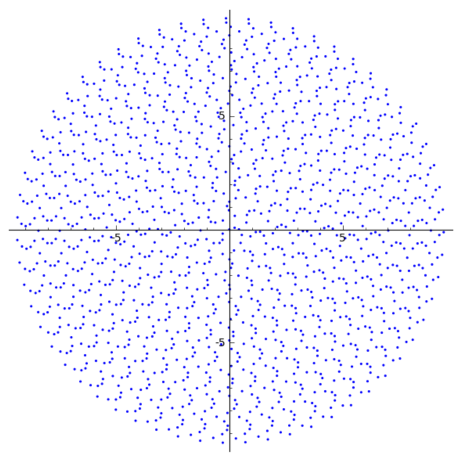
$\alpha = 1/2$



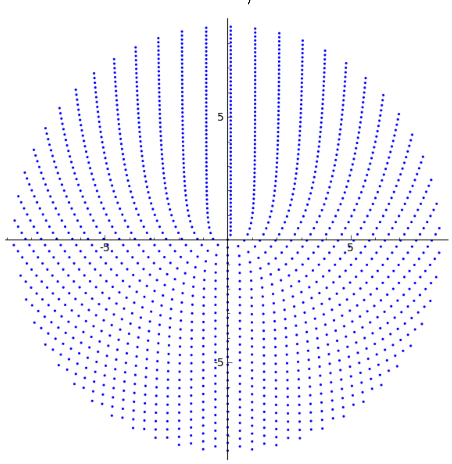
$\alpha = 1$



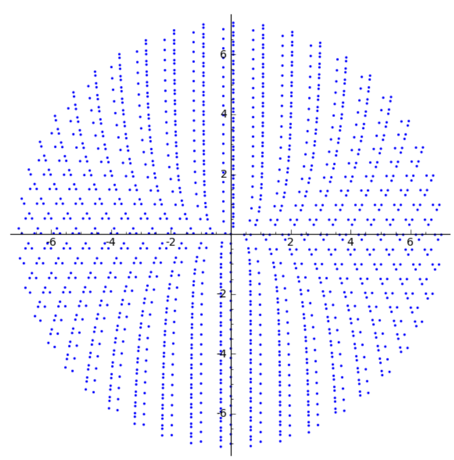
$\alpha = 65/64$



$\alpha = 1.3$

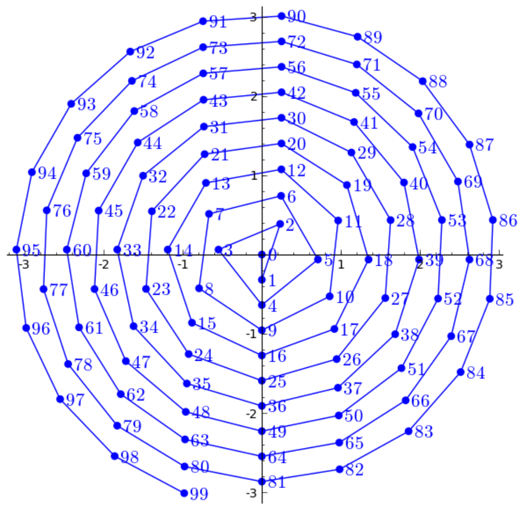


$\alpha = \sqrt{2}$

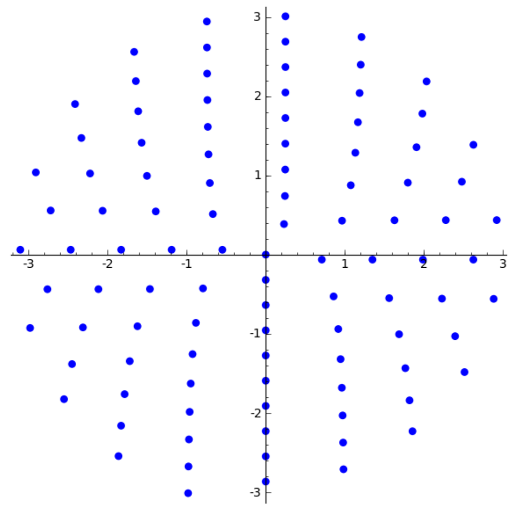


$\alpha = \sqrt{3}$

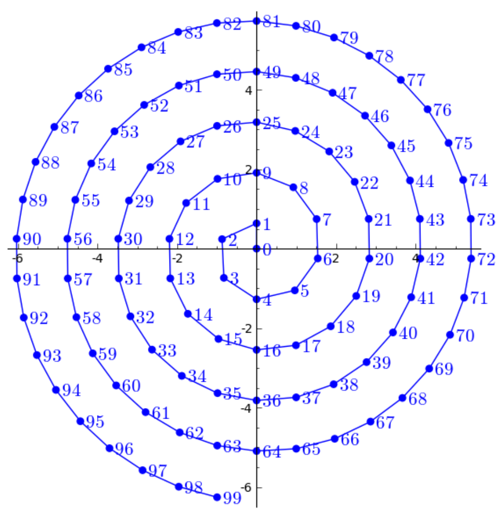
Figure 3. Joined points.



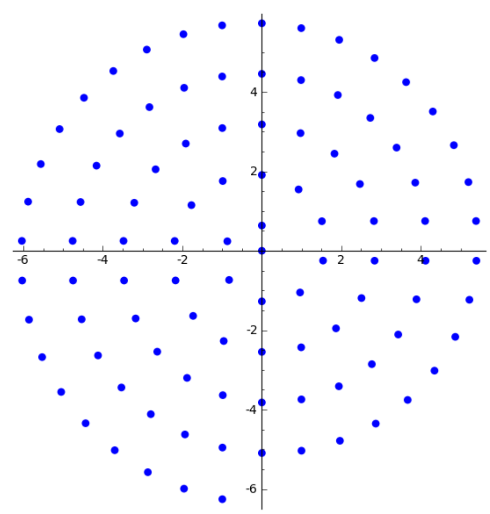
$\alpha = 1,\ N = 100$



$\alpha = 1,\ N = 100$

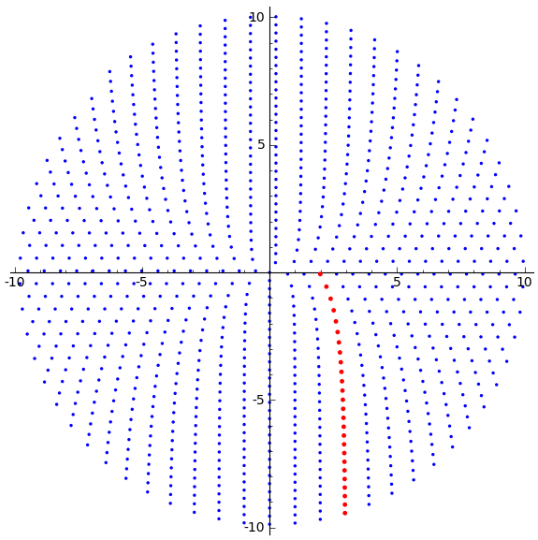


$\alpha = 1/2,\ N = 100$

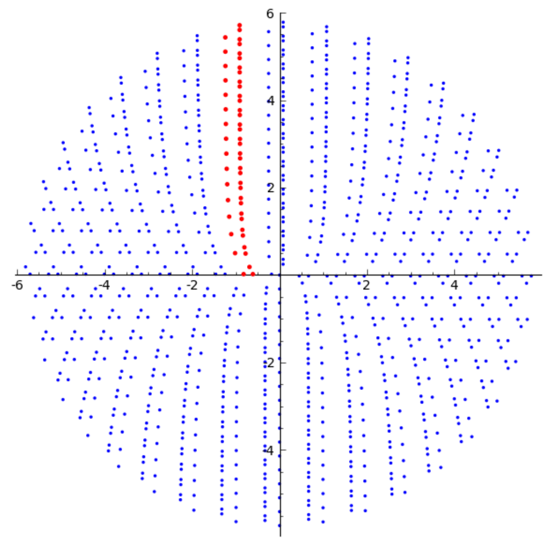


$\alpha = 1/2,\ N = 100$

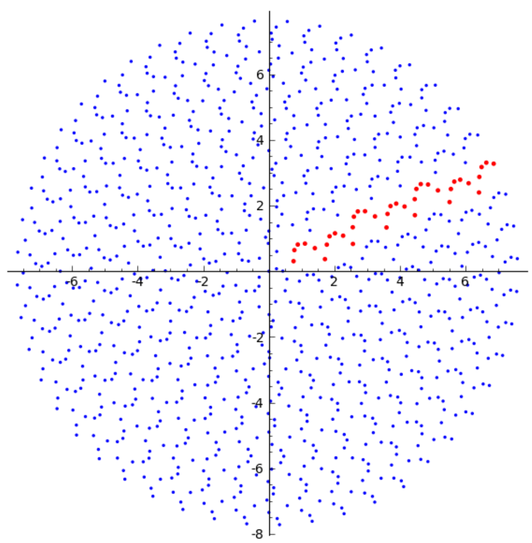
Figure 4. Orbits.



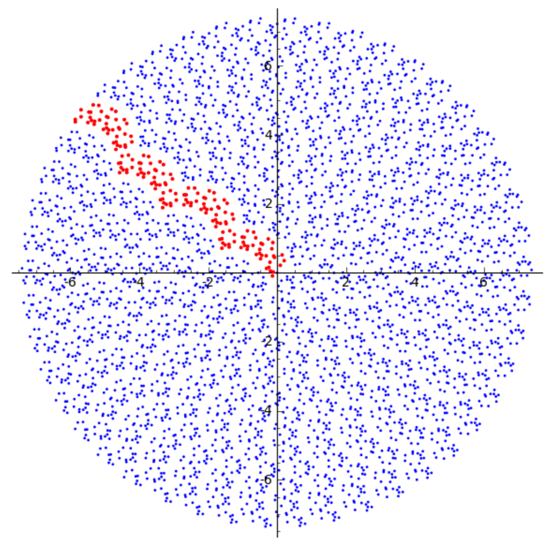
$m_0 = 39$, $\alpha = 1$, $N = 1000$



$m_0 = 11$, $\alpha = \sqrt{3}$, $N = 1000$



$m_0 = 11$, $\alpha = 1.3$, $N = 1000$



$m_0 = 1$, $\alpha = e$, $N = 4000$

Programs (SAGE). Listings.

```
##########
# FIGURE 1
##########
for alpha in [1/2, 1, 65/64, 1.3, sqrt(2), sqrt(3)]:
    P = exact_plot(alpha,2000)
    P.show()
```

```
##########
# FIGURE 2
##########
for alpha in [1/2, 1, 65/64, 1.3, sqrt(2), sqrt(3)]:
```

```
    P = plot_model(alpha,2000)
    P.show()
```

```
##########
# FIGURE 3
##########
P = plot_modeltj(1/2,100)
P.show()
P = plot_model(1/2,100, 40)
P.show()
P = plot_modeltj(1,100)
P.show()
P = plot_model(1,100, 40)
P.show()
```

```
##########
# FIGURE 4
##########
P = orbit(39, 1, 1000, psize = 15)
P.show()
P = orbit(11, sqrt(3), 1000, psize = 15)
P.show()
P = orbit(11, 1.3, 1000, psize = 15)
P.show()
P = orbit(1, exp(1), 4000, psize = 10)
P.show()
```

```
%cython
import math
def l_sqr(double alpha, long long int N):
    cdef double sx = 0.0
    cdef double sy = 0.0
    cdef double t
    cdef double n
    Lt = []
    for n in range(N):
        t = 6.283185307179586476925286*alpha*math.sqrt(n)
        sx += math.cos(t)
        sy += math.sin(t)
        Lt.append( (sx,sy) )
    return Lt
```

```
# exact calculations
def exact_plot(alph,n_points):
    L = l_sqr(alph,n_points)
    P = list_plot(L, size = 5)
    P.set_aspect_ratio(1)
    return P
```

```
# model
def plot_model(alph, n_points, psize = 5):
    L = []
    for k in range(n_points):
```

```
        ang = (6.283185307179586476925286*(alph*sqrt(k)-0.25) ).n()
        rad = (sqrt(k)/3.1415926535/alph).n()
        L.append( (rad*cos(ang),rad*sin(ang)) )
    P = list_plot(L, size = psize)
    P.set_aspect_ratio(1)
    return P
```

```
# model calculations text
def plot_modeltj(alph,n_points):
    L = []
    P = point((0,0), size =0)
    for k in range(n_points):
        ang = (6.283185307179586476925286*(alph*sqrt(k)-0.25) ).n()
        rad = (sqrt(k)/3.1415926535/alph).n()
        L.append( (rad*cos(ang),rad*sin(ang)) )
        P += text(" $"+str(k)+"$",L[k],horizontal_alignment='left',fontsize=15)
    P += list_plot(L, size = 40)
    P += list_plot(L, thickness = 1, plotjoined = True)

    P.set_aspect_ratio(1)
    return P
```

```
# orbit over model
def orbit(n_0, alph, n_points, psize = 14):
        t_n = n_0
        Or = []
        P = plot_model(alph, n_points, psize/2)
        while t_n < n_points:
                theta_n = (alph*sqrt(t_n)).n()
                rad = theta_n/pi/alph/alph
                Or.append( ( (rad*cos(2*pi*theta_n-pi/2)).n(),
                                            (rad*sin(2*pi*theta_n-pi/2)).n()) )
                t_n += round( 2*sqrt(t_n)/alph +1/alph^2 )
        P += list_plot(Or, size = psize, color = 'red', zorder =60)
        P.set_aspect_ratio(1)
        return P
```