Computational Physics
Problem Set 2

Saleh Shamloo Ahmadi
Student Number: 98100872

September 27, 2021

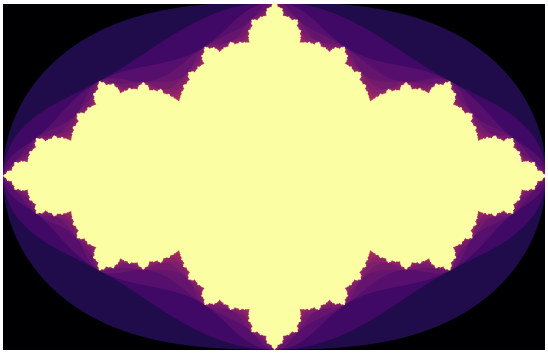
1 Julia Set

Informally, The Julia set of a complex mapping function consists of the points in the complex plane that do not diverge after repeatedly applying the mapping function to them. That is, if $z_{n+1} = f(z_n)$ the Julia set $J(f)$ consists of the points where the sequence z_n converges if z_0 is the point.

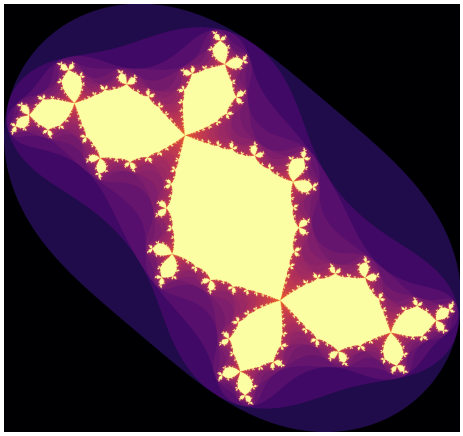
To find $J(f)$, we first find a lower bound for the abolute value of numbers that diverge (R) (this just has to be *a* lower bound, not *the smallest* lower bound). Then we repeatedly apply the mapping $f(z)$ to the points $|z| < R$ and stop when $|z_n| > R$. For better visuallization of the function's behavior, we color each point based on the iteration in which the point diverged. Algorithm 1 demonstrates this in practice.

The quadratic polynomial $f(z) = z^2 + c$ is of particular interest. It is related to the mandelbrout set. The polynomials in this report are all generated using this mapping function.

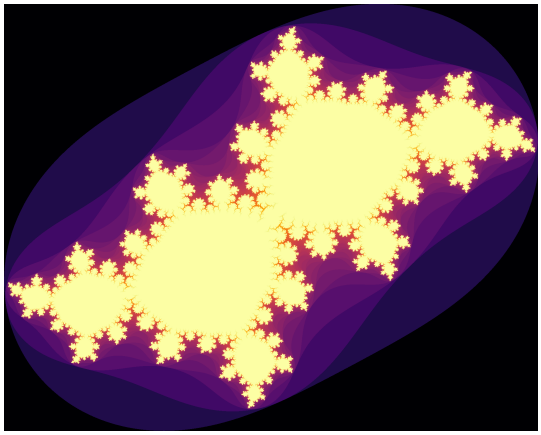
$c = 0.6$



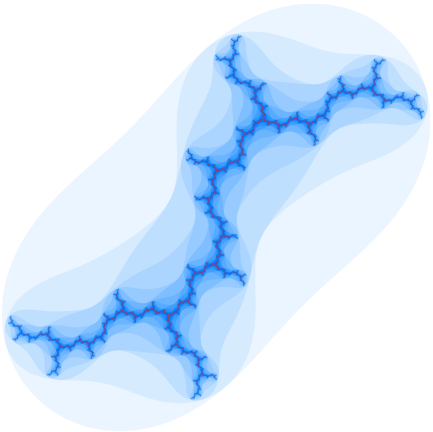
$c = -0.123 + 0.745i$



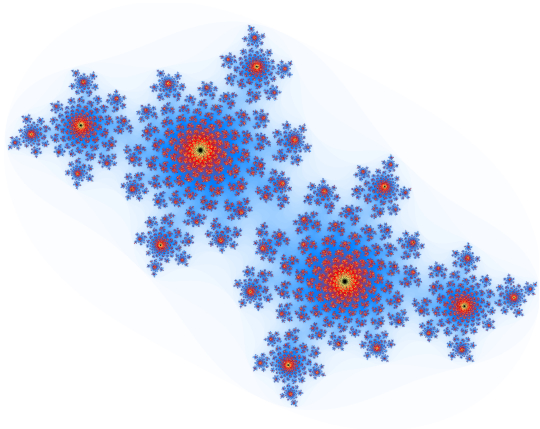
$c = -0.391 - 0.587i$



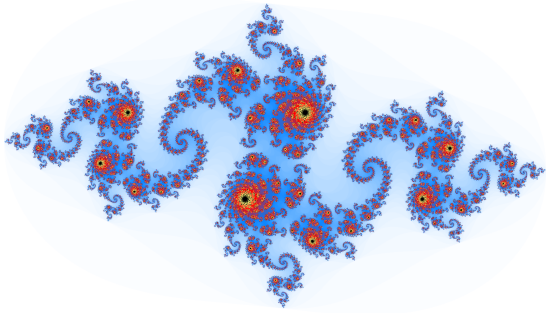
$c = -i$



$c = -0.4 + 0.6i$

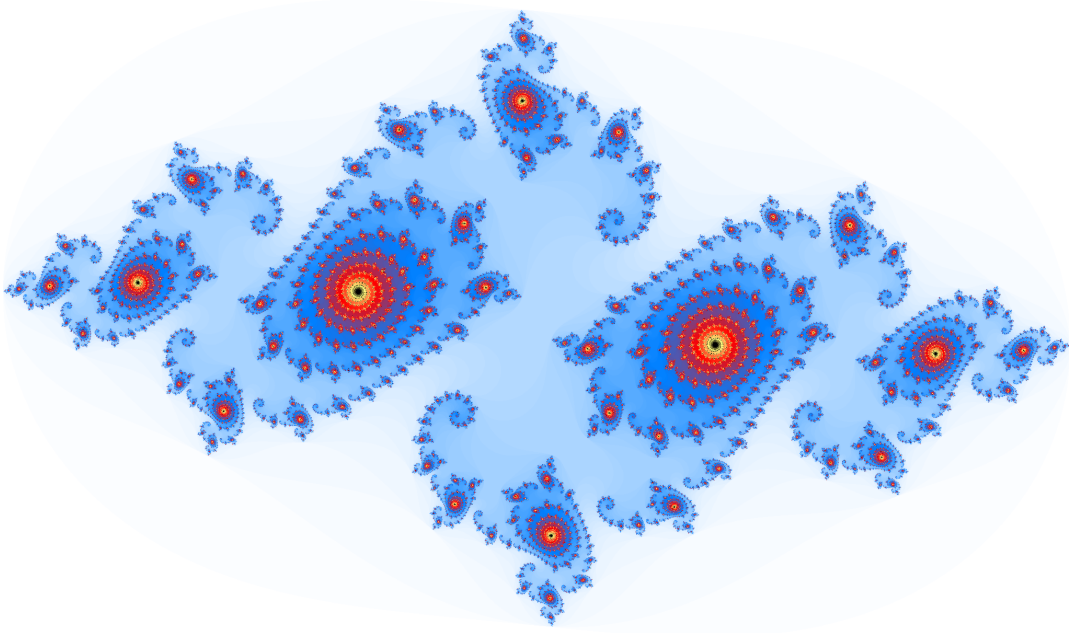


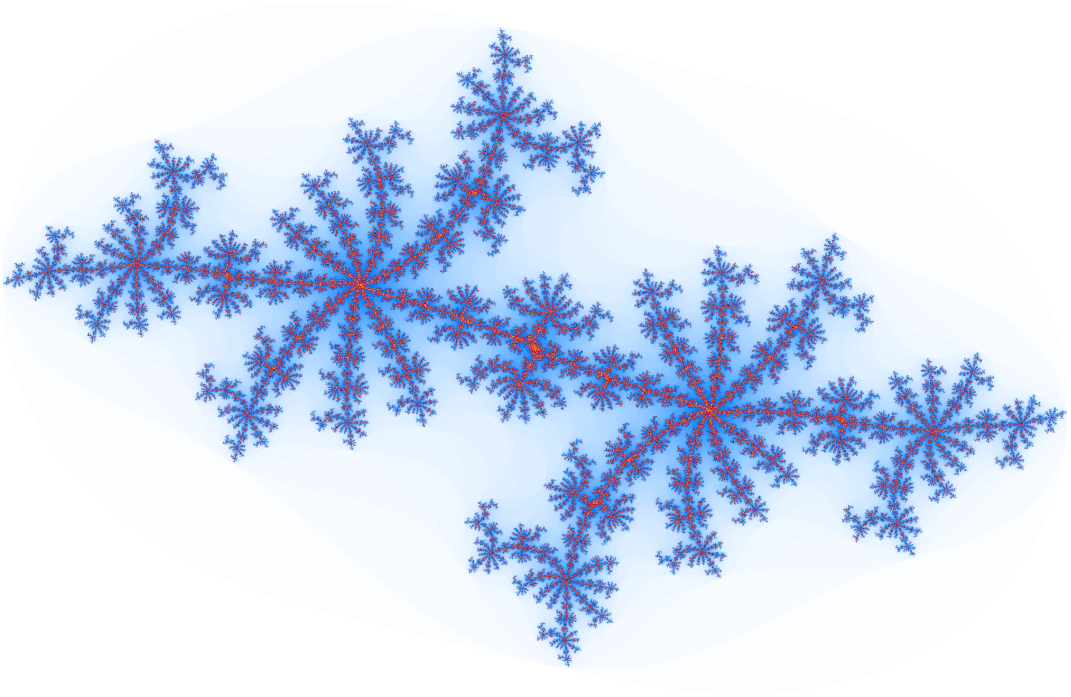
$c = -0.8 + 0.16i$

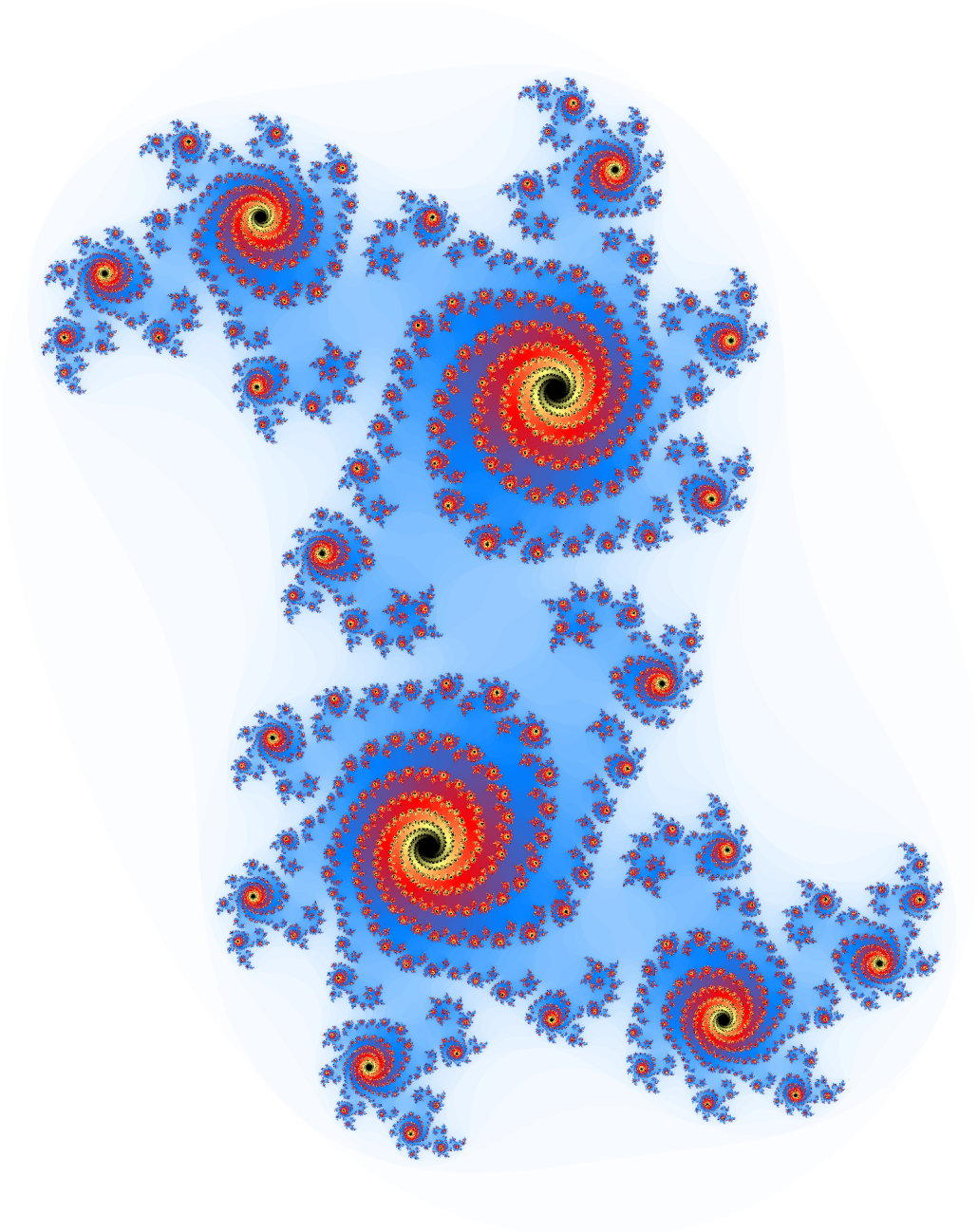


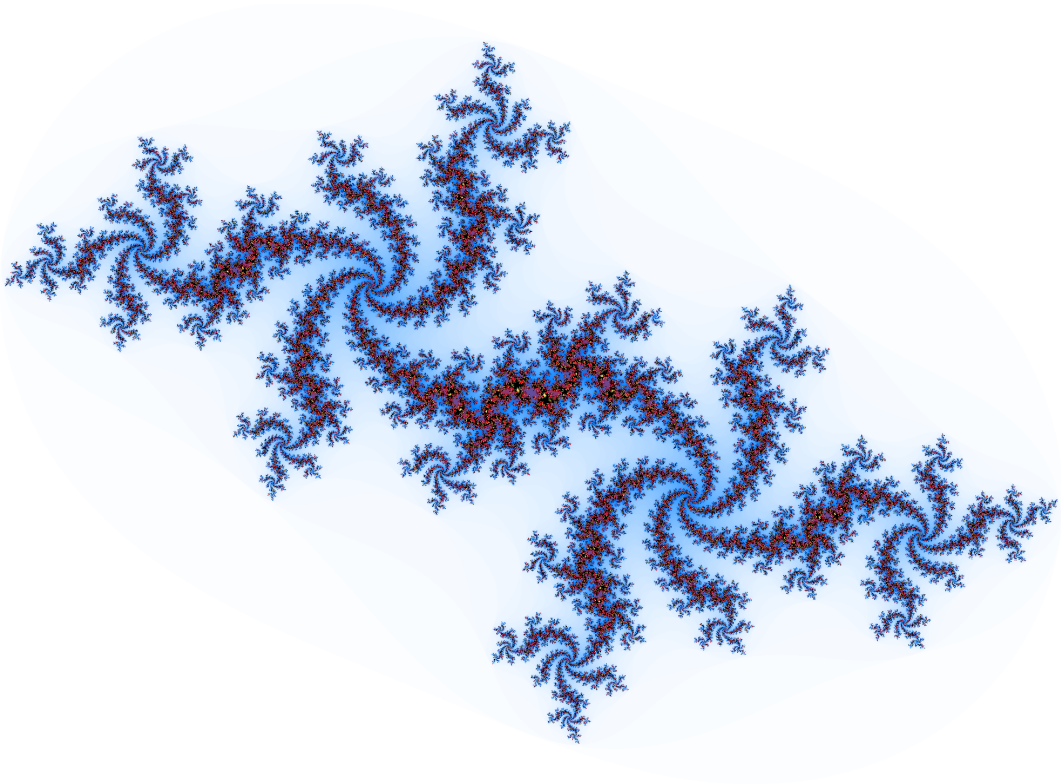
Algorithm 1 Fractal Generation by Relative Mapping

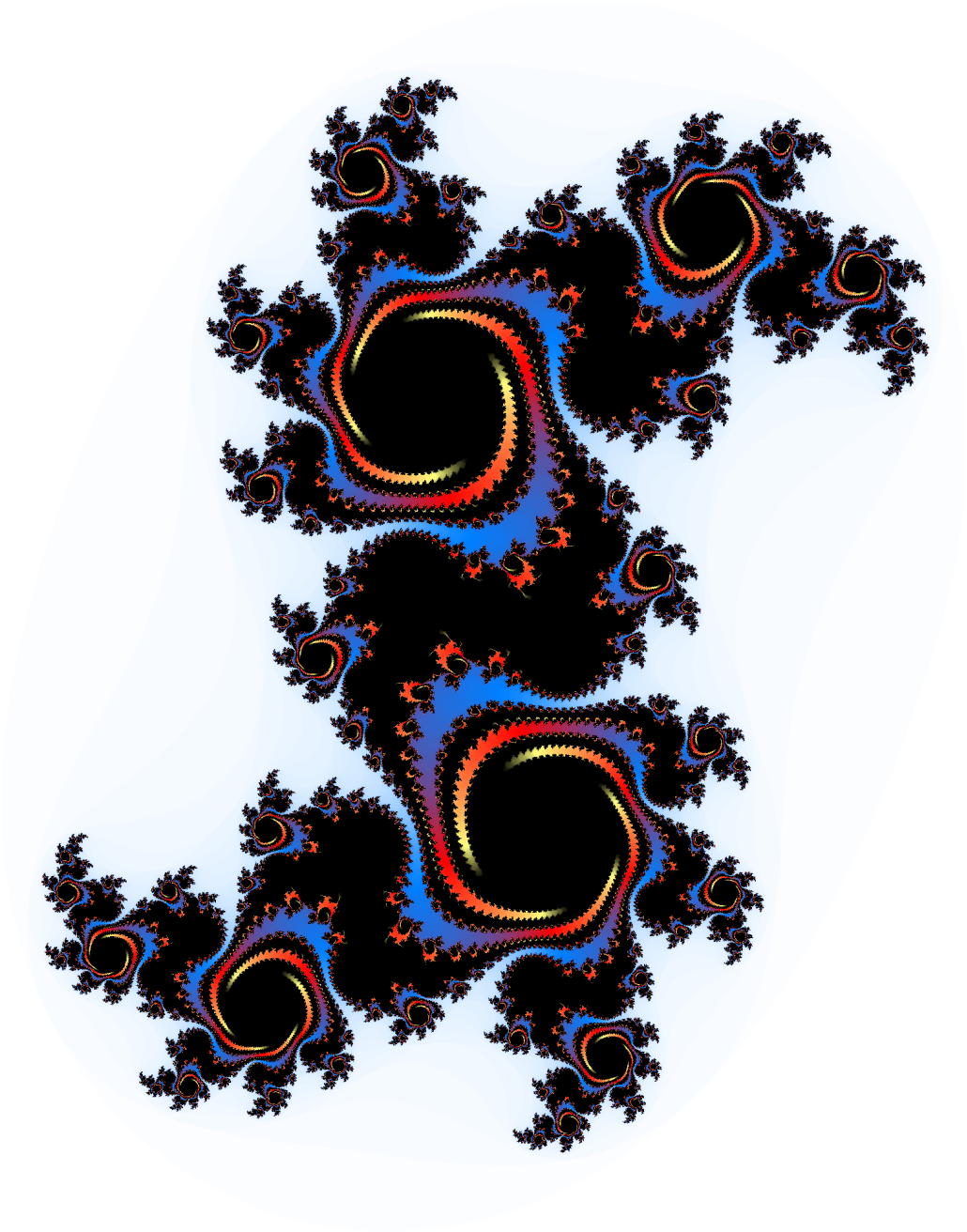
function FRACTAL(f, i_{max}, R) ▷ f is the mapping function and i_{max} is the maximum iteration depth
  for all z in the complex plane, with a given resolution **do**
    $i \leftarrow 0$
    while $i < i_{max}$ **and** $|z| < R$ **do**
      $z = f(z)$
      $i \leftarrow i + 1$
    end while
    $J[z] \leftarrow i$ ▷ J is the array holding the divergence iteration for each point of the complex plane and $J[z]$ is the element corresponding to the complex number z
  end for
  return J
end function

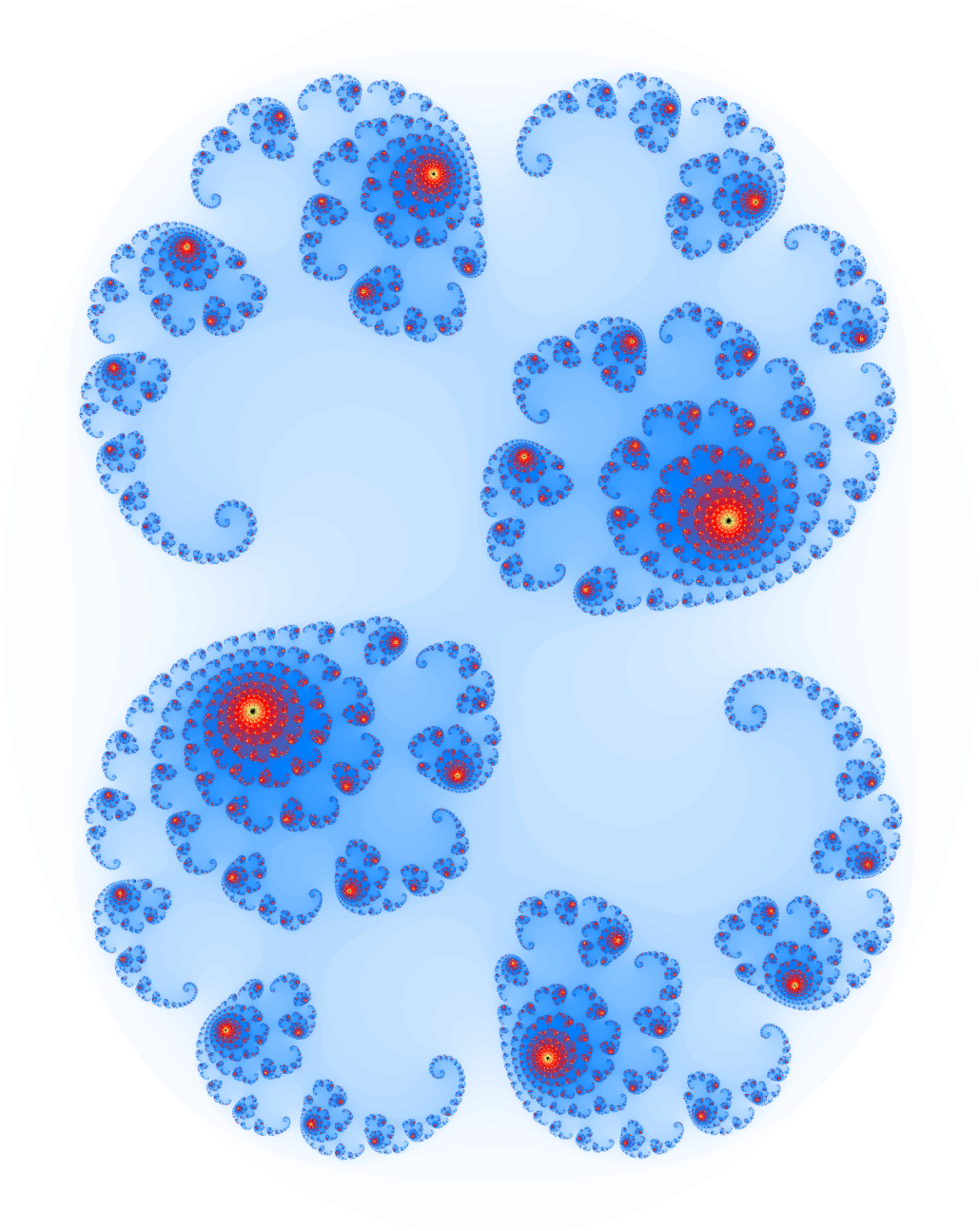






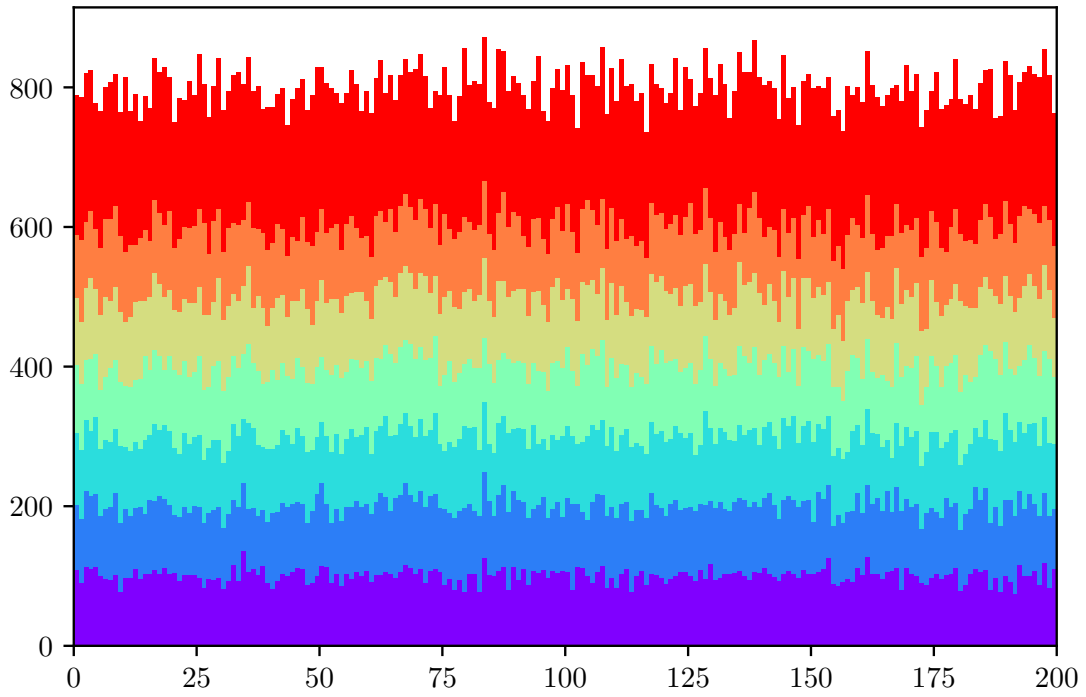






2 Random Ballistic Desposition

The algorithm for random ballistic desposition is trivial; Just add points particles to different points (buckets) randomly and save your data in consistent intervals.



To check the validity of our data, we can use simple linear regression. If the particles are randomly deposited, their final distribution must be uniform. So

$$\bar{h}(t) = \frac{\sum_{i=0}^{L} h_i(t)}{L} = \frac{t}{L}, \tag{1}$$

where L is the length of the 1D surface, $h_i(t)$s are the heights (number of deposited particles) of each point at time t, and $\bar{h}(t)$ is the average height at time t.

We can also calculate the *dynamic growth exponent*; In the following equation, β is the dynamic growth exponent (SD is the standard deviation):

$$\mathrm{SD}_h(t) \sim t^{\beta} \tag{2}$$

After running the simulation multiple times and also with larger number of points and larger time, it is obvious that β oscillates around 0.5.

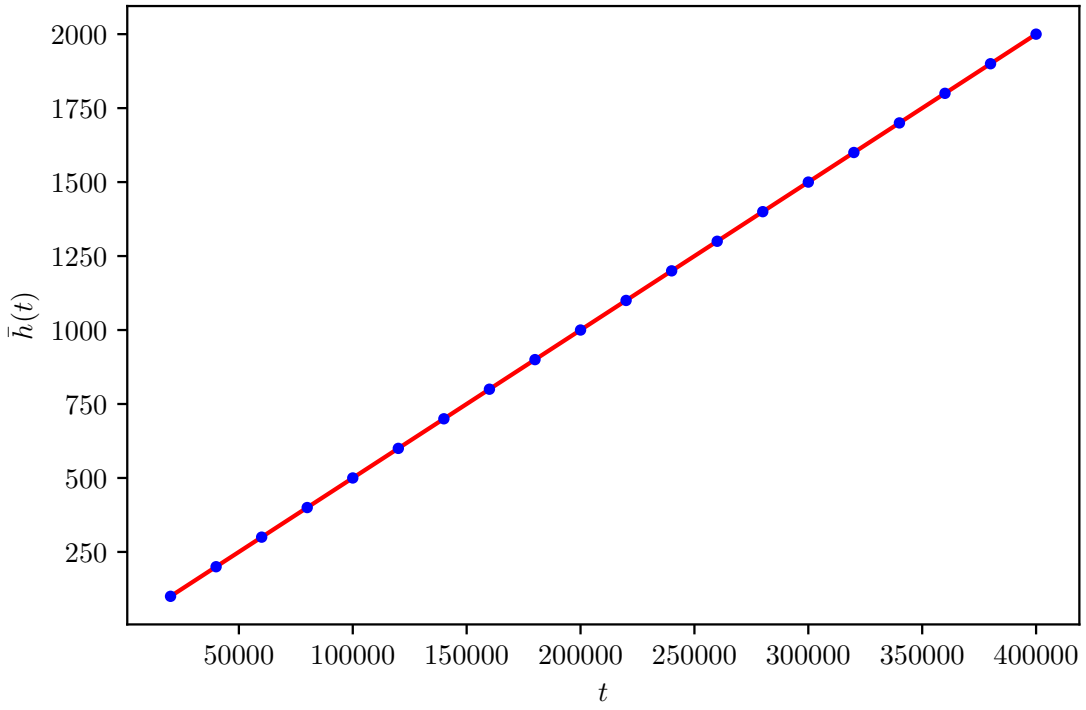


Figure 1: $L = 200$, slope $= 0.005 \pm 5 \times 10^{-19}$

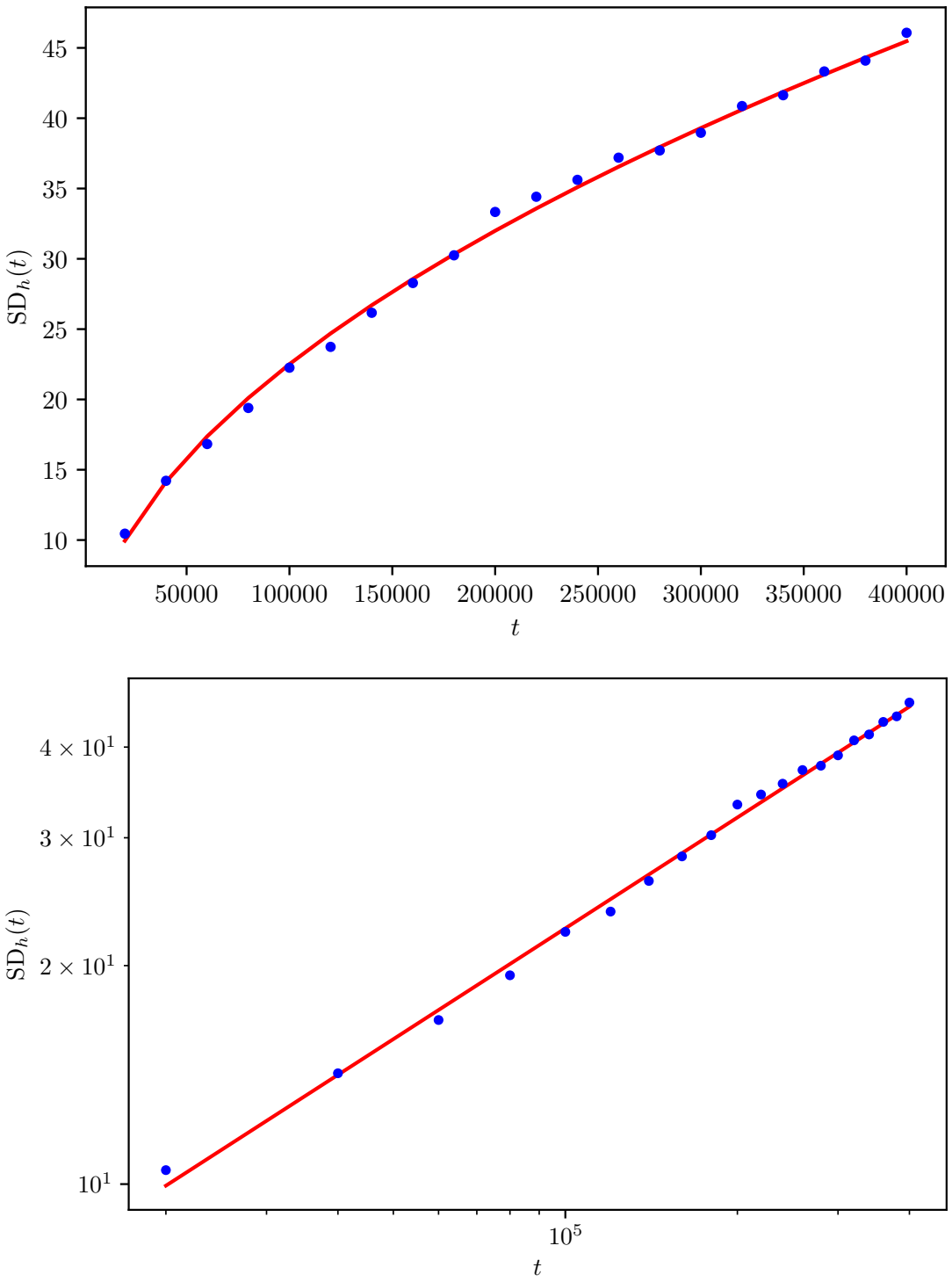


Figure 2: $\beta = 0.507 \pm 0.007$