# Pointillist Graphing of Iterated Function Systems

Risto A. Paju
Jyväskylä, Finland
risto.a.paju@iki.fi
http://algoristo.com/

## Abstract

I demonstrate simple graphing methods for functions in the 2D plane that capture essential features of the functions while being aesthetically pleasing. Instead of solid curves or geometric shapes, my approach generates dithered shades directly by mapping large numbers of individual points. Multi-colour composites of these graphs can be used for more elaborate constructs, such as iterated function systems.

## Introduction

Iterated function systems (IFS) are a common tool for visualizing Julia sets. In the classic example of the complex polynomial $p_c(z) = z^2 + c$, it is first inverted to form the IFS

$$\begin{cases} f_1(z) = \sqrt{z-c} \\ f_2(z) = -\sqrt{z-c} \end{cases} \tag{1}$$

Random points iterated with this IFS will then converge towards points in $J_c$, the Julia set of the polynomial $p_c$. Known as the Inverse Iteration Method (IIM), this approach is noted for its speed, but the uneven point density in the resulting graphs is often regarded as a drawback. Methods such as Modified IIM have been developed to overcome this issue and produce solid, connected graphs. [2], [6, pp. 173-178]

However, in my algorithmic art (for example Figure 1) I celebrate unevenness. I appreciate the delicate, veil-like textures of IIM over a solid technical look. In fact, the aesthetic power of image dithering or halftoning has been known for centuries, from the art of mezzotint to computer graphics [1]. Besides, as the point density reflects the properties of the functions themselves, such graphs may also have educational value.

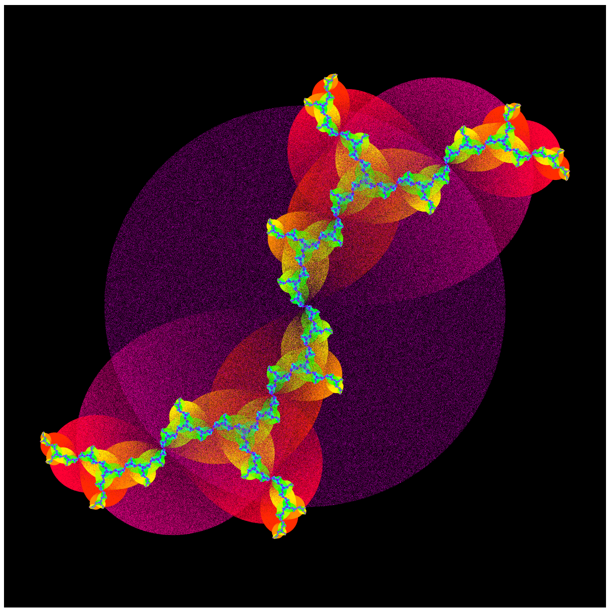


**Figure 1** : *IFS iterations of the Julia set $J_{-i}$ with the initial disc $B(0,1)$*

As I exploit the varying point density for artistic purposes, I also use multiple colours to demonstrate the different stages of iteration in a single graph. In addition, I explore the effects of different initial sets on the final works. This paper outlines all of these steps I use to create my art.

Throughout the text, I use the Julia set IFS (1) with $c = -i$ as an example. The resulting graphs (Figures 1, 4(a), and 4(b)) represent not only the set $J_{-i}$, but also the processes of getting there from the various initial sets.

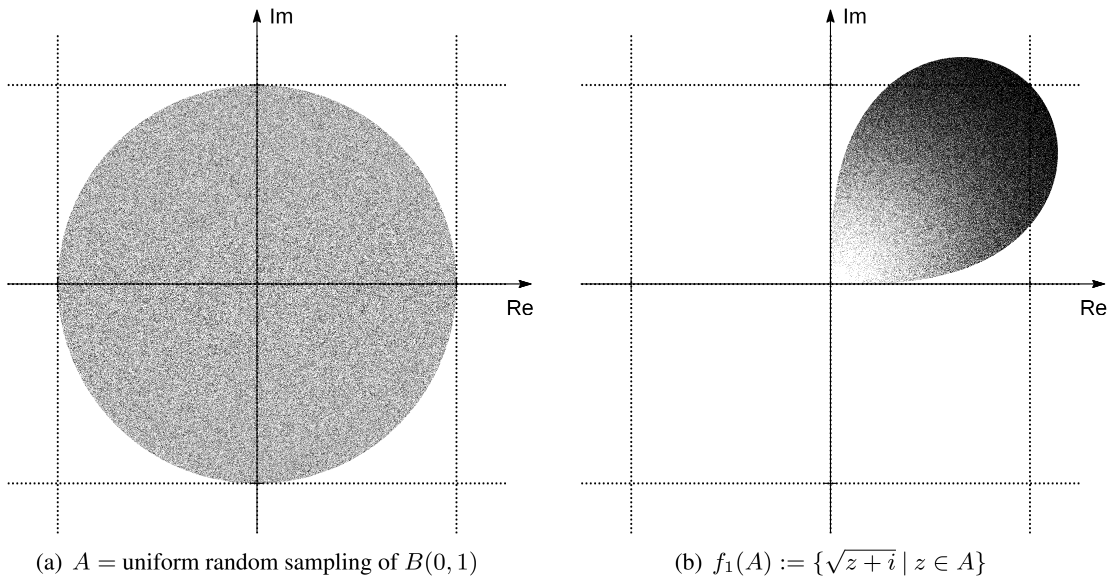
While this paper only refers to Julia sets, my method is readily applicable to any IFS. In fact, it is not limited to complex numbers or even the 2D plane.

## Basic Method and the Gradient Effect

To find the image of a geometric shape under a function, the following simple method turns out to have surprising benefits. It automatically generates variable, dithered colours based on the function in question.

1. Choose an initial shape, such as the origin-centred unit disc $B(0,1)$
2. Fill the shape with random points of a uniform distribution
3. Compute the image points with the given function or IFS. For the latter, I usually prefer the random-game method: one function of the IFS is randomly chosen for each point.

From our IFS (1) with $c = -i$, consider the first function $f_1(z) = \sqrt{z+i}$. Its derivative increases as we approach the point $z = -i$, so the images of nearby points end up further apart. Given their initial uniform distribution, this generates colour intensities inversely proportional to the derivative (Figure 2). In computer graphics, such varying shades would be called gradients; here they have a simple relation to the gradient or derivative of the original function.



(a) $A =$ uniform random sampling of $B(0,1)$ (b) $f_1(A) := \{\sqrt{z+i} \mid z \in A\}$

**Figure 2** : *Density variations in the complex plane due to the square root function*

Illustrations of complex functions in 2D are never without compromises. A complete $\mathbb{C} \to \mathbb{C}$ plot would require four dimensions, or possibly a range of colours for the phase information. However, this method can express some of that complex information in a simple monochrome plot.

## Colouring Iterations

To visualize iterated function systems, I simply repeat step 3. Using the Julia set IFS (1) with $c = -i$, the set $J_{-i}$ begins to take form after a few iterations (Figure 3).

For colouring the results, I was inspired by the escape-time method of fractal graphics, where colours are assigned by the number of iterations (also known as the Level Set Method [2]). Here the iterations carry a very different meaning, but the same idea applies. I simply lay the graphs of iterates on top of each other, using a different colour for each.

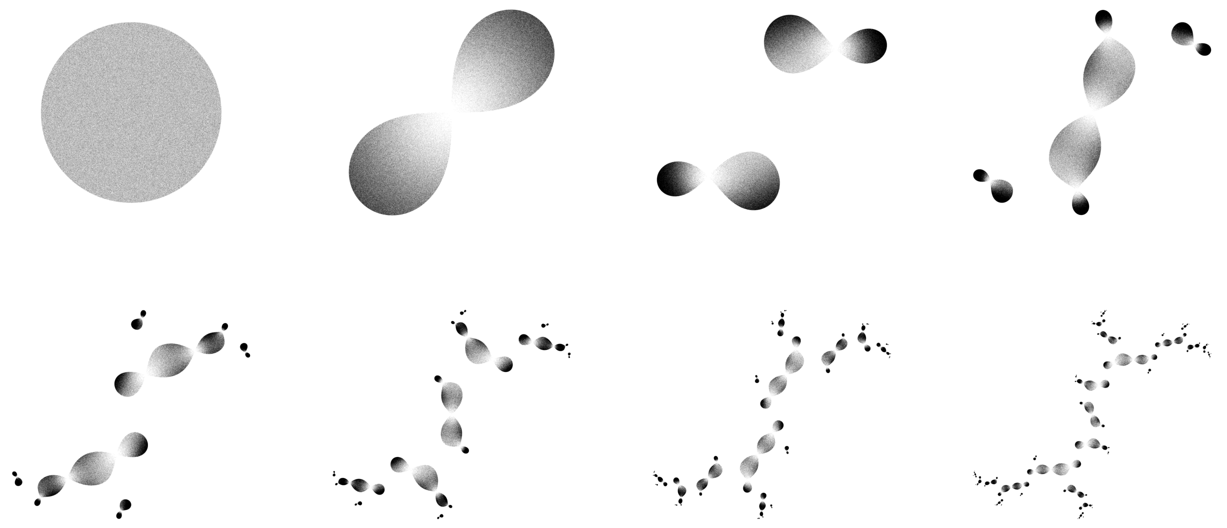


**Figure 3**: $B(0,1)$ *and its first iterations under the Julia set IFS of* $J_{-i}$

In the resulting composite graph (Figure 1), the early stages of the iteration form a kind of shadow or halo around the final form, while tracing its course of evolution. This is made possible by the contractive nature of the functions: the later stages cover a smaller effective area. Meanwhile, as the random-game method maintains the number of points between iterations, colours are more concentrated in the final stages.

## Effects of Initial Shapes

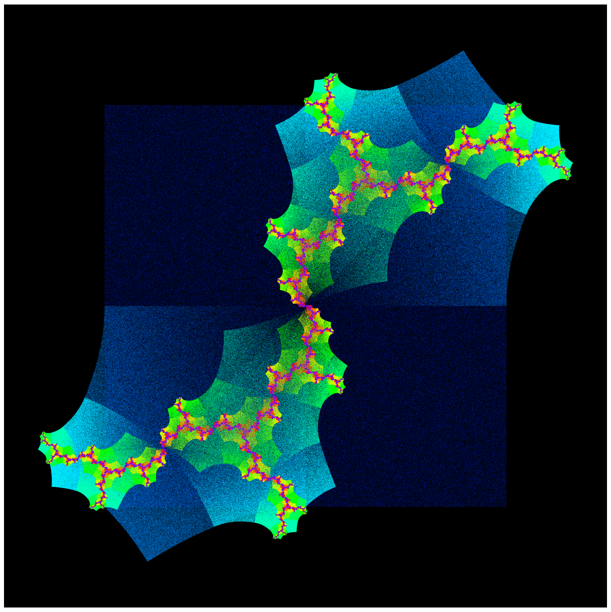
As shown by Hutchinson [4], the limit set of an IFS is independent of the initial set, provided the IFS is contractive and the initial set is compact. This is also true at a practical level, with finite iterations and a suitably bounded IFS. However, the choice of the initial set is very apparent in the early iterations. For example, the square $[-1,1]^2$ becomes rather jagged under the example IFS (Figure 4(a)).

## Colour Mixing Effects

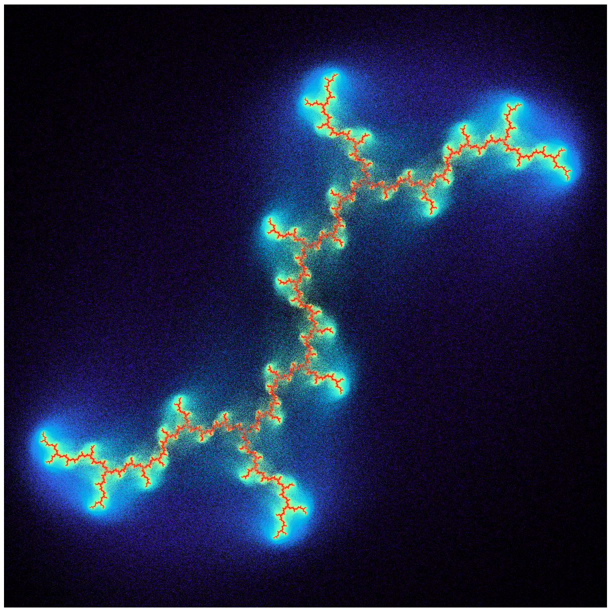
A Gaussian distribution of points makes a particularly interesting initial set. The lack of a defined edge is maintained throughout iterations with smooth functions such as the Julia set IFS. When combined into a single graph, the individual iterations are no longer discernible from the fluid mass of colours (Figure 4(b)). It may bring to mind the smoothly coloured escape-time graphs of Julia sets, while being rather different in spirit. This effect of mixing colours is also apparent with simpler initial sets, especially when the point density varies smoothly over a large area.

## Summary

A simple pointwise approach generates powerful visualizations of functions in two or more dimensions. Gradients of functions turn into image gradients simply by mapping individual points, without any additional algorithms. This method can also be applied in succession to generate IFS fractals. For further instructive and artistic depth, several iteration stages can be overlaid together in colour. Different initial sets provide yet another dimension to explore.



(a) Square initial set $[-1,1]^2$



(b) Gaussian initial set centred at origin with unit std. deviation

**Figure 4** : *IFS iterations of the Julia set $J_{-i}$*

## Acknowledgements

My art has been greatly inspired by Antti Käenmäki through his series of lectures [5]. A number of free software projects, including the Julia language [3] and the gnuplot graphing package [7], have provided invaluable tools for practical implementation. Finally, I would like to thank the reviewers for their helpful suggestions.

## References

[1] Lee Daniel Crocker, Paul Boulay, and Mike Morra. Digital halftoning, 1991. `http://www.efg2.com/Lab/Library/ImageProcessing/DHALF.TXT` (as of Apr. 19, 2016).

[2] Vasileios Drakopoulos. Comparing rendering methods for Julia sets. *Journal of WSCG*, 10:155–161, 2002.

[3] Jeff Bezanson et al. The Julia language. `http://julialang.org/` (as of Jan. 5, 2016).

[4] John E. Hutchinson. Fractals and self similarity. *Indiana Univ. Math. J.*, 30 (5):713–747, 1981.

[5] Antti Käenmäki. Johdatus fraktaaligeometriaan (course handout). Department of Mathematics and Statistics, University of Jyväskylä, 2015.

[6] Heinz-Otto Peitgen and Dietmar Saupe, editors. *The Science of Fractal Images*. Springer, 1988.

[7] T. Williams and C. Kelley. gnuplot – an interactive plotting program. `http://gnuplot.info/` (as of Jan. 5, 2016).