Iterated Function Systems and a Fractal Image Compression Method

Martin Pham
Supervisor: Ed Vrscay

July 2017

Table of Contents

Data Compression

- Lossy data compression can be considered as representing your signal in a space where you can reduce redundancy
- Iterated function systems (IFS) present another method for recovering some target signal by posing as inverse problem

Iterated Function Systems

- Let $(X, \|\cdot\|)$ be a complete metric space with $H(X)$ the set of non-empty subsets of X
- An N-map IFS on $(X, \|\cdot\|)$ is defined by N (affine) mappings

$$w_i(x) = S_i x + T_i$$

Let $0 < |S_i| < 1$ such that $\{w_i\}$ are contractive

- We define a "fractal transform" operator T that "combines" all N images of each mapping. For $([0,1], \|\cdot\|)$ this is

$$T(x) = \bigcup_{i=1}^{N} w_i(x) \quad \forall x \in H(X)$$

T can be shown to be contractive operator

This is important because...

Fractal Transform Attractors*

- Banach Contraction Principle: T is guaranteed to have a unique fixed point

$$\bar{x} = T(\bar{x}) = \bigcup_{i=1}^{N} w_i(\bar{x})$$

- Fixed point is said to be "self-similar": it is the union of contracted versions of itself
- This is a property of fractals, and images are known to be self-similar (think of clouds or trees)

A few examples for generating fractal-like objects...

Example: Cantor Set

- Consider the 2-map IFS on $([0,1], \|\cdot\|)$ given by

$$w_1(x) = \frac{1}{3}x$$

$$w_2(x) = 1 - \frac{1}{3}x$$

- We pick some initial $x_0 \in [0,1]$ and apply operator T repeatedly, keeping track of how many times each point on the interval is visited

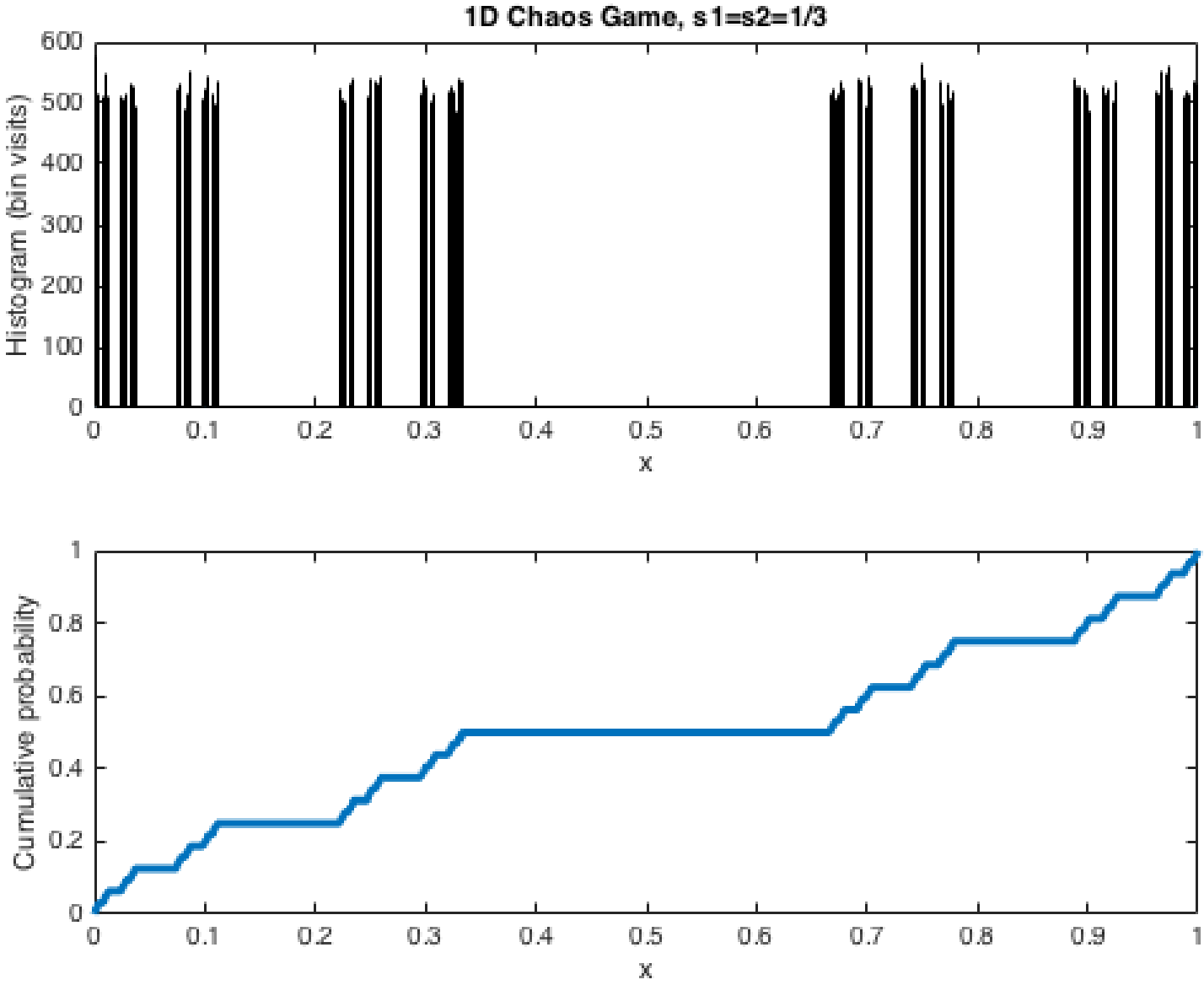

Figure: For large sampling, you get repeated structures

- We may also work on probability measures. Let $M(X)$ be all probability measures on X.

Example: Sierpinski Gasket

- Consider the 3-map IFS on $([0,1]^2, \|\cdot\|)$ with probabilities p_i

$$w_1(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} x, \quad p_1 = \frac{1}{3}$$

$$w_2(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} x + \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}, \quad p_2 = \frac{1}{3}$$

$$w_3(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} x + \begin{bmatrix} 0.5 \\ \frac{\sqrt{3}}{4} \end{bmatrix}, \quad p_3 = \frac{1}{3}$$

- For $S \in H(X)$ and $\mu \in M(X)$, operator is

$$(T\mu)(S) = \sum_{i=1}^{N} p_i \mu(w_i^{-1}(S))$$

Sierpinski Gasket

https://youtu.be/Wy_UfwfvhUg

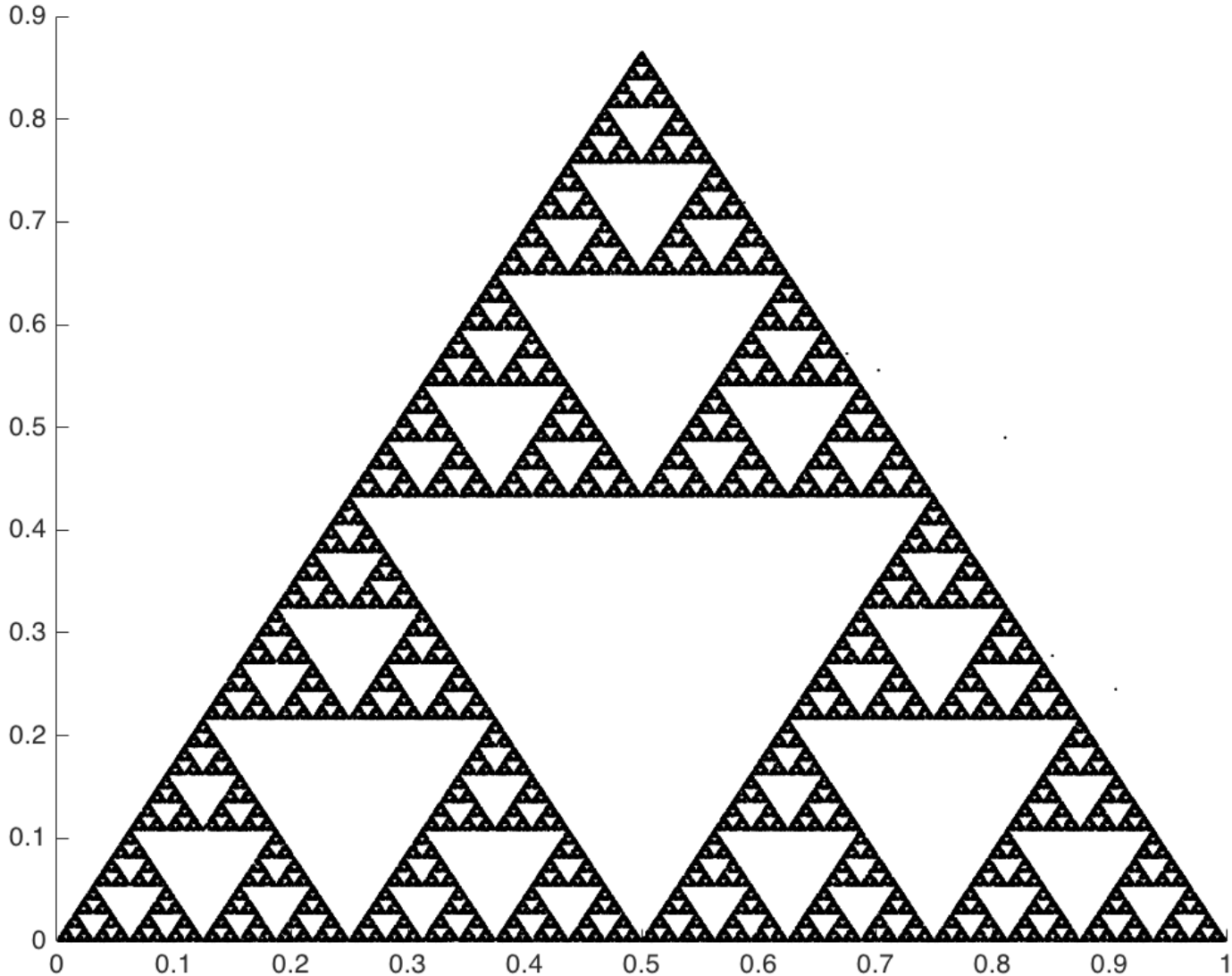

Figure: 100000 samples

Chaos Game

- We can play the "chaos game" by changing some values and observing the resultant attractor. Consider the slightly different IFS

$$w_1(x) = \begin{bmatrix} 0.5 & 0.42 \\ 0 & 0.5 \end{bmatrix} x, \quad p_1 = \frac{1}{3}$$

$$w_2(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} x + \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}, \quad p_2 = \frac{1}{3}$$

$$w_3(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} x + \begin{bmatrix} 0.5 \\ \frac{\sqrt{3}}{4} \end{bmatrix}, \quad p_3 = \frac{1}{3}$$

Chaos Game on Sierpinski Gasket

https://youtu.be/i79-CJY_MYQ

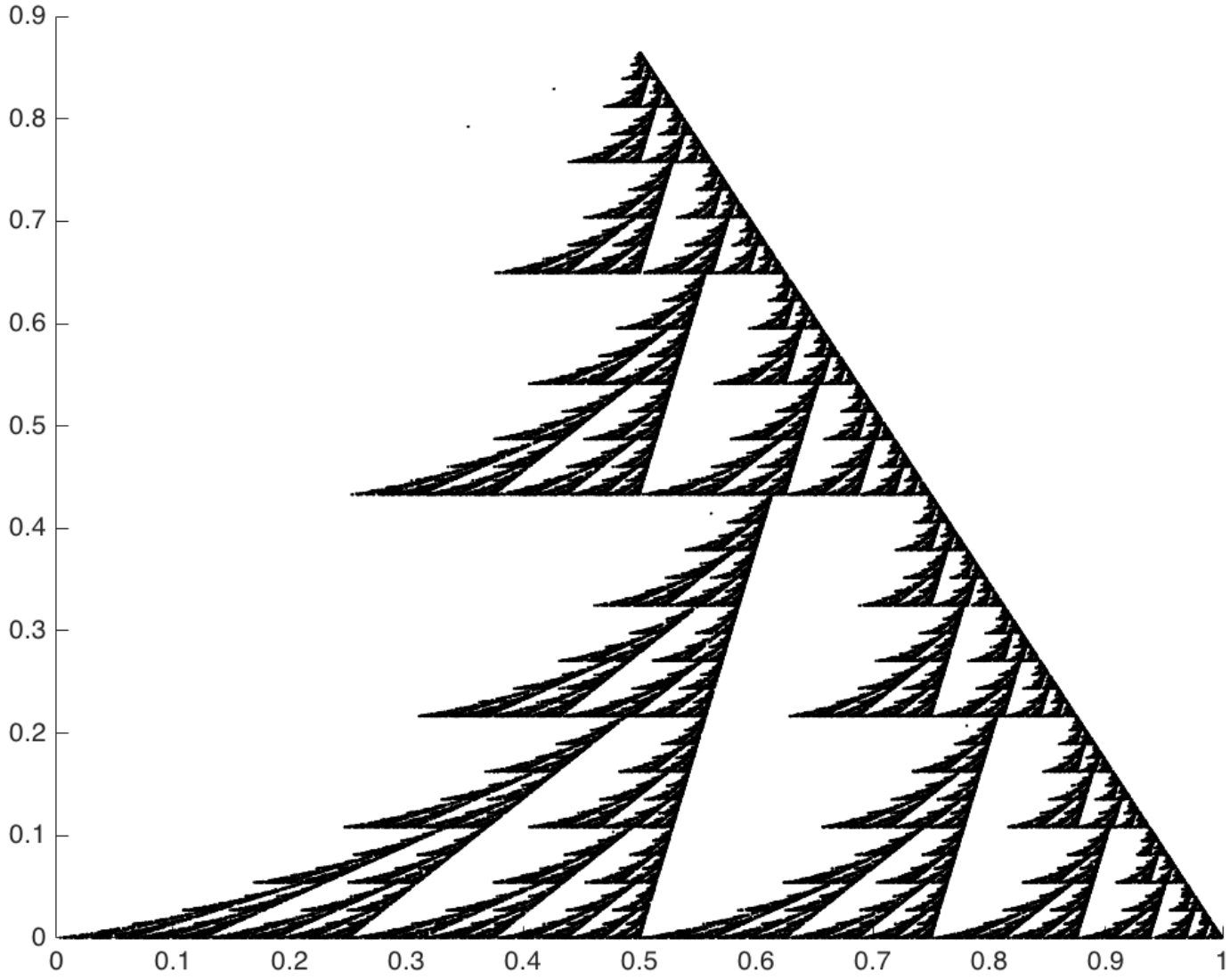

Figure: 100000 samples

Example: Barnsley Fern

- Consider 4-map IFS with probabilities on $([-3,3]\times[0,10], \|\cdot\|)$

$$w_1(x) = \begin{bmatrix} 0 & 0 \\ 0 & 0.16 \end{bmatrix} x, \quad p_1 = 0.01$$

$$w_2(x) = \begin{bmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}, \quad p_2 = 0.85$$

$$w_3(x) = \begin{bmatrix} 0.2 & -0.26 \\ 0.23 & 0.22 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix}, \quad p_3 = 0.07$$

$$w_4(x) = \begin{bmatrix} -0.15 & 0.28 \\ 0.26 & 0.24 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0.44 \end{bmatrix}, \quad p_4 = 0.07$$

Barnsley Fern

https://youtu.be/RriynAPyWY4

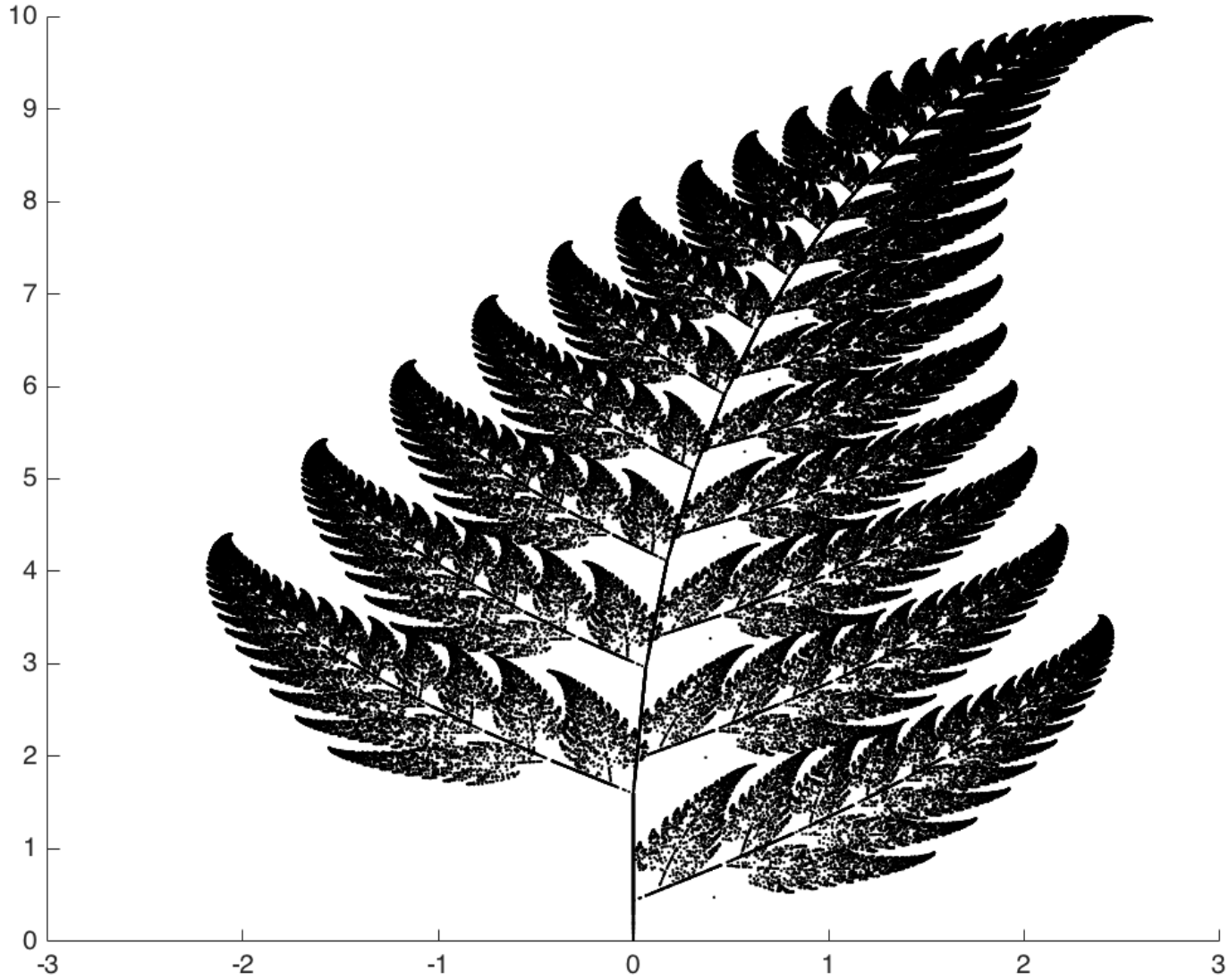

Figure: 300000 samples

We can play the chaos game...

Chaos Game on Barnsley Fern

- Consider 4-map IFS with probabilities on $([-3,3]\times[0,10],\|\cdot\|)$

$$w_1(x)=\begin{bmatrix}0 & 0\\ 0 & 0.16\end{bmatrix}x, \quad p_1=0.05$$

$$w_2(x)=\begin{bmatrix}0.85 & -0.04\\ 0.04 & 0.85\end{bmatrix}x+\begin{bmatrix}0\\ 1.6\end{bmatrix}, \quad p_2=0.6$$

$$w_3(x)=\begin{bmatrix}0.2 & -0.26\\ 0.23 & 0.22\end{bmatrix}x+\begin{bmatrix}0\\ 1.6\end{bmatrix}, \quad p_3=0.175$$

$$w_4(x)=\begin{bmatrix}-0.15 & 0.28\\ 0.26 & 0.24\end{bmatrix}x+\begin{bmatrix}0\\ 0.44\end{bmatrix}, \quad p_4=0.175$$

Chaos Game on Barnsley Fern

https://youtu.be/D61mmlqZ7Yo

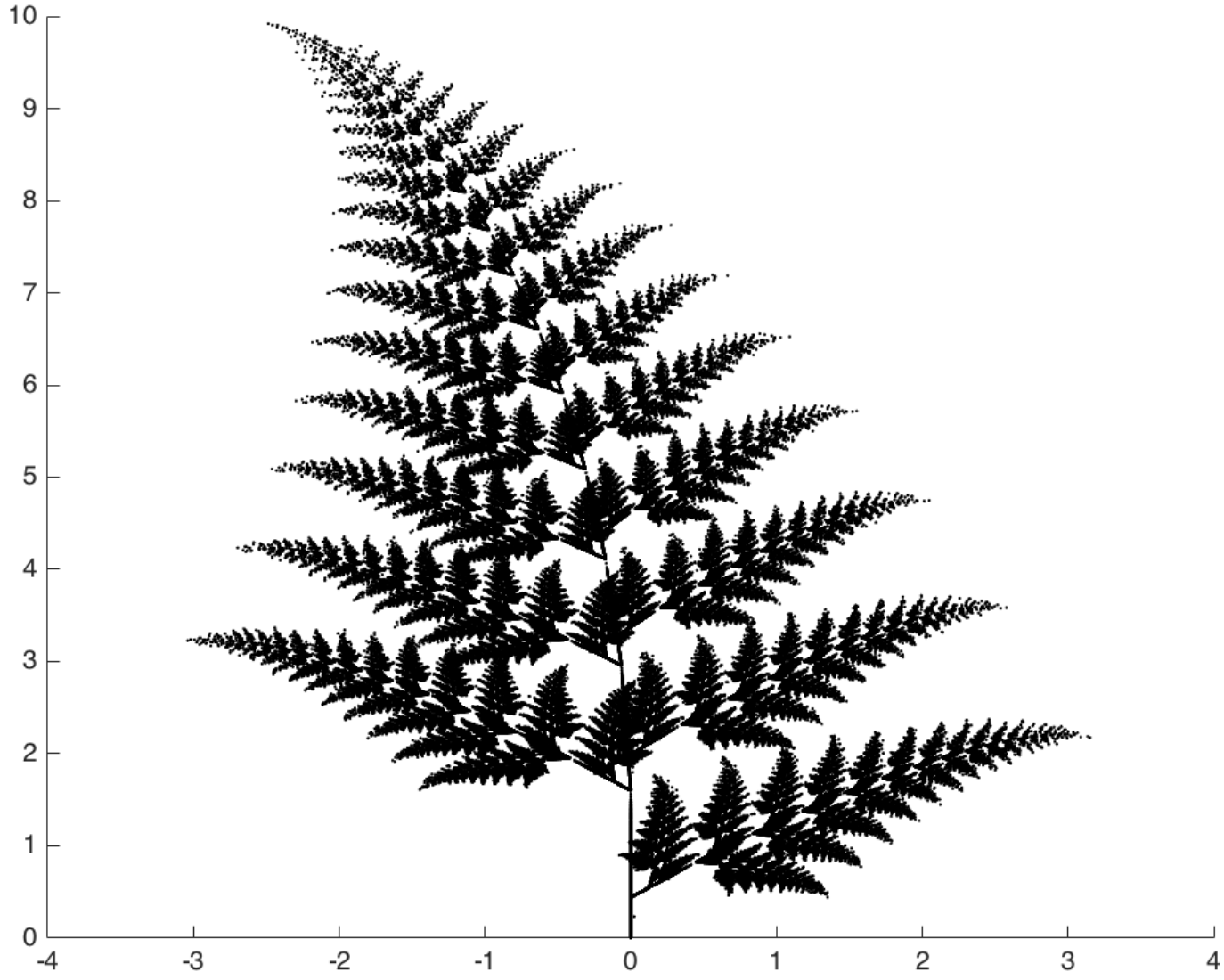

Figure: 300000 samples

Relation to Images

- Greyscale images may be thought of as two dimensional functions. Let $M(X)$ be functions on $X = [0,1]^2$.
- Instead of p_i we introduce (affine) range-modifying mappings

$$\phi_i(t) = \alpha_i t + \beta_i$$

For $u \in M(X)$, the operator T is

$$(Tu)(x) = \sum_{i=1}^{N}{}' \phi_i(u(w_i^{-1}(x)))$$

with fixed point

$$\bar{u} = \sum_{i=1}^{N}{}' \phi_i(\bar{u}(w_i^{-1}(x)))$$

Important theorem...

Collage Theorem*

- Let u be some target image
- Given some IFS operator T with contractive constant c, then Collage Theorem states:

$$\|u - \bar{u}\| \leq \frac{1}{1-c}\|u - Tu\|$$

That is, finding T such that it maps u close to itself means the fixed point $\bar{u}$ well approximates u

Proof of Collage Theorem

Consider:

$$\begin{aligned}
\|u - \bar{u}\| &= \|u - T\bar{u}\| \quad \text{(fixed point)} \\
&= \|u - Tu + Tu - T\bar{u}\| \quad \text{(add zero)} \\
&\leq \|u - Tu\| + \|Tu - T\bar{u}\| \quad \text{(triangle inequality)} \\
&\leq \|u - Tu\| + c\|u - \bar{u}\| \quad \text{(contractive)}
\end{aligned}$$

Rearranging:

$$\|u - \bar{u}\| \leq \frac{1}{1-c}\|u - Tu\|$$

This is the heart of the encoding scheme

Local IFS Encoding

- In practice, we only use subsections of an image to approximate other subsections of that image:
  - Partition image non-overlapping into 16×16 pixel domain blocks D_j and 8×8 pixel range blocks R_i
  - For each R_i, find mapping ϕ_i and domain block D_j that minimizes "collage error"

$$\Delta_{ij} = \|u(R_i) - \phi_i(u(D_j))\|$$

  - Encode indices and parameter values

$$(i, j, \alpha_i, \beta_i)$$

- This is a very computationally expensive global search, but when decoding we can start at any image and iterate towards an approximation of our target

Local IFS Encoding

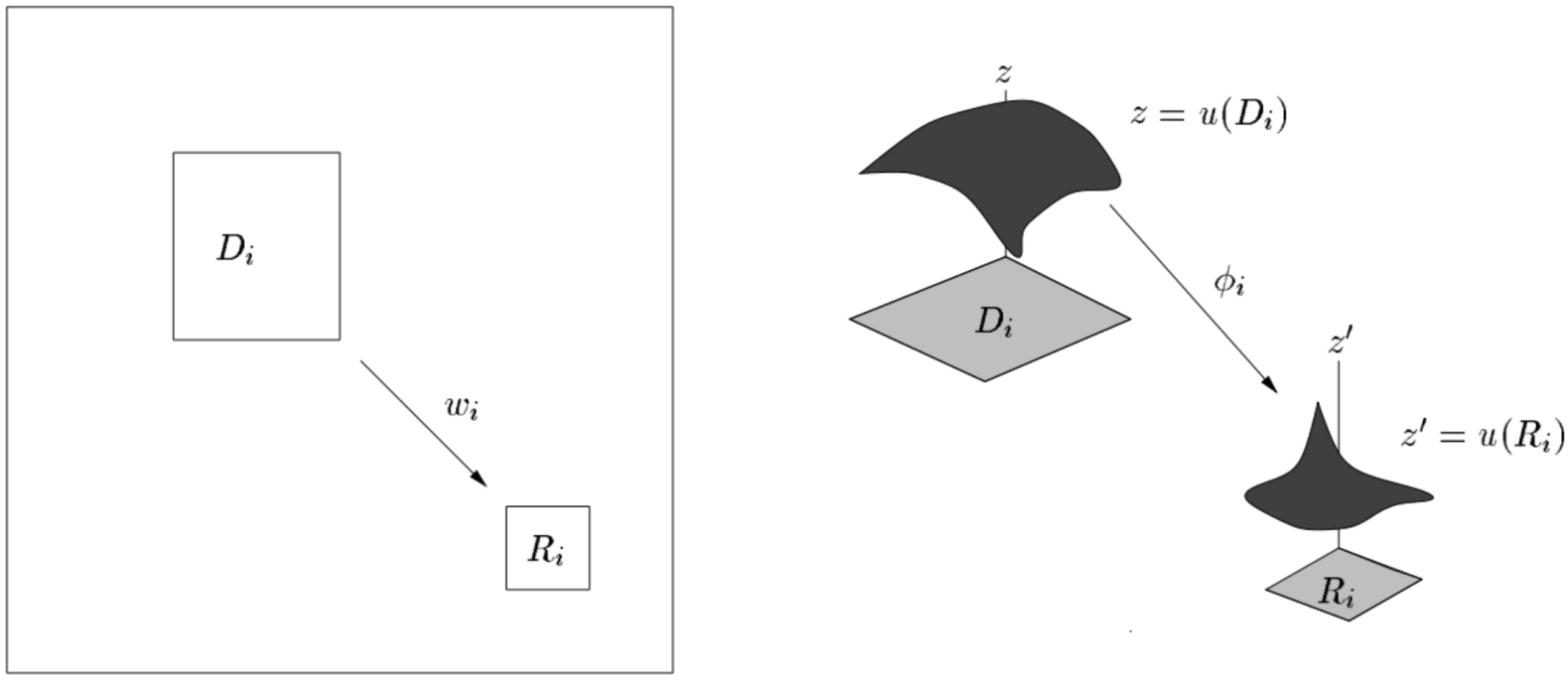

Figure: w_i the spatial component, ϕ_i the greyscale component

Given a 256×256 pixel image, we have $32 \times 32 = 1024$ range blocks and $16 \times 16 = 256$ domain blocks

Target Image

Figure: Lena

Attractor Image

Lena Compressed

https://youtu.be/6dAFFc8Tf3Y

Comparison

Figure: Uncompressed

Figure: Compressed

Compression Results

- Original:

$$(256 \times 256) \text{ pixels} \times 8 \text{ bits} = 524288 \text{ bits}$$

- Compressed:

$$(32 \times 32) \text{ range blocks} \times (10 + 8 + 8 + 8) \text{ bits} = 34816 \text{ bits}$$

So

$$\text{Compression ratio} = \frac{\mathit{Uncompressed}}{\mathit{Compressed}} = 15.06$$

$$\text{Space savings} = 1 - \frac{\mathit{Compressed}}{\mathit{Uncompressed}} = 0.93$$

The End

ASIDE (Time Permitting):

IFS methods can be also be used to generate Hilbert curves

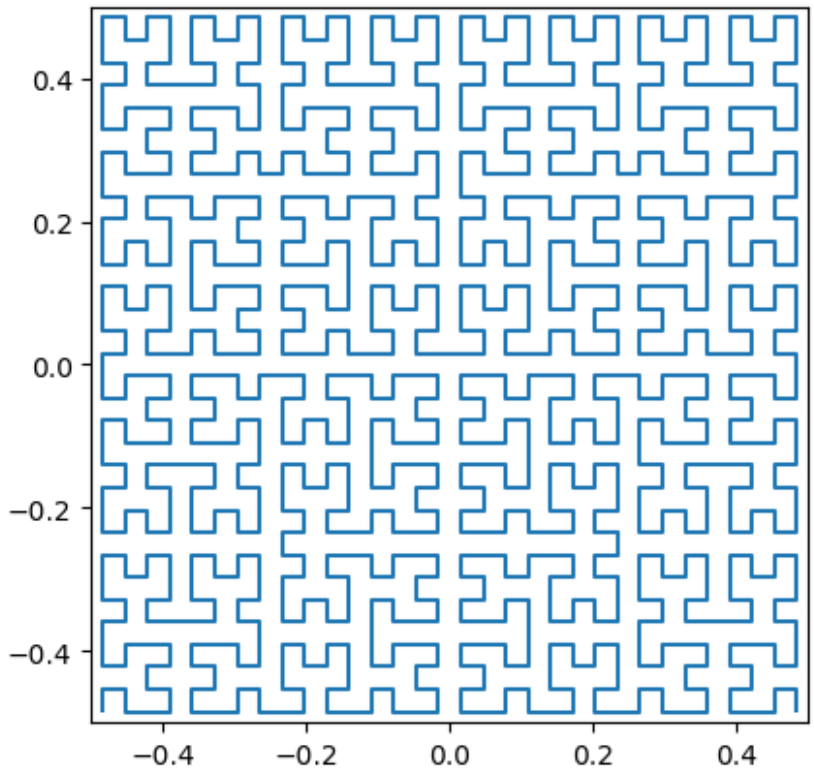

Figure: 2D Hilbert curve, 5 iterations

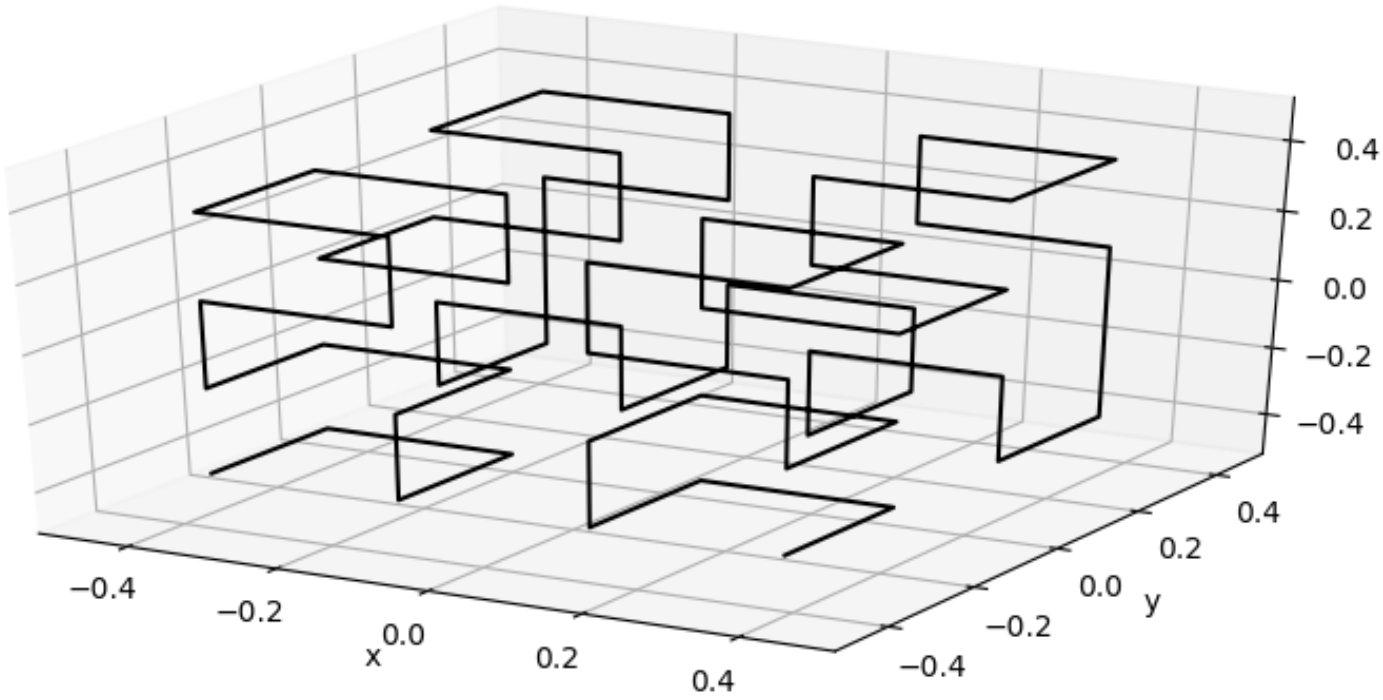

Figure: 3D Hilbert curve, 3 iterations

Soundcloud Artwork Collaging

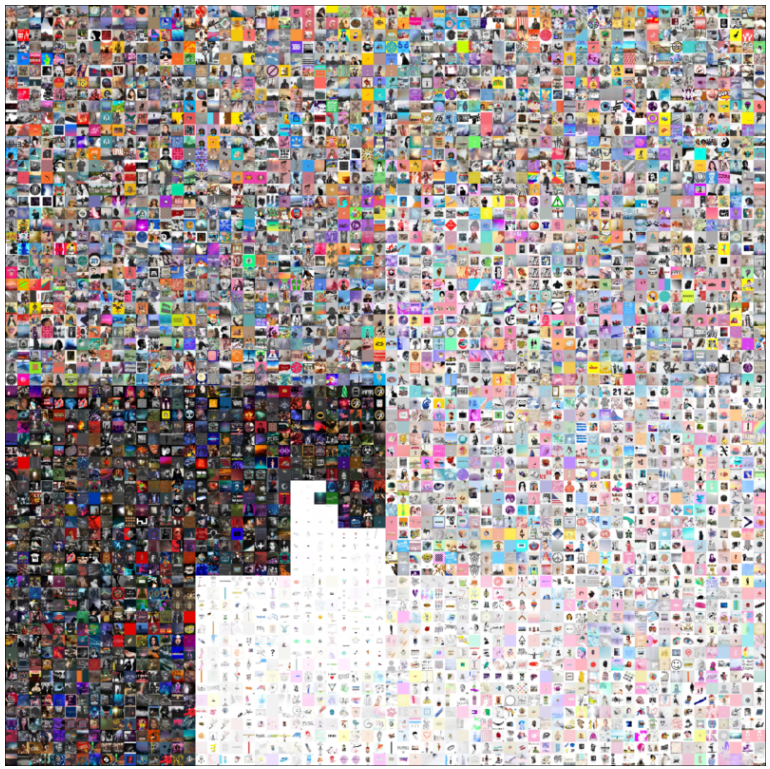

Figure: Mean pixel characterization

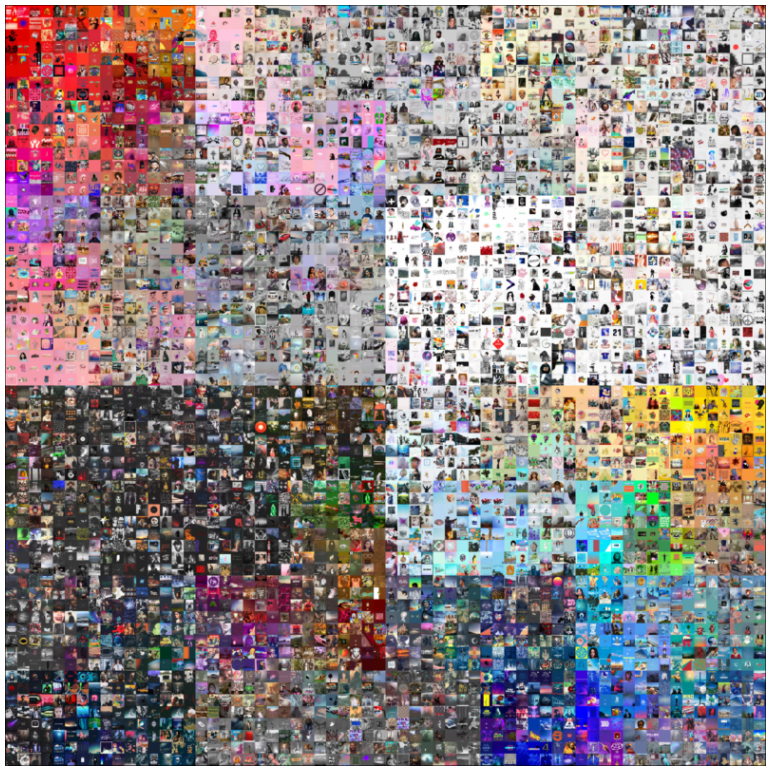

Figure: Modal pixel characterization