Delhi Technological University

Department of Applied Mathematics

M.Sc Mathematics

Fractal Image Compression: Banach Fixed Point Theorem

Submitted By :
Vidipt Vashist
2K20/MSCMAT/33
Bharat Varshney
2K20/MSCMAT/07

Submitted To :
Prof. Satyabrata
ADHIKARI

Fractal Image Compression: Banach Fixed Point Theorem

Vidipt Vashist (2K20/MSCMAT/33)
Bharat Varshney (2K20/MSCMAT/07)

November 23, 2021

Contents

1 INTRODUCTION **4**

2 THEORY **5**
2.1 Metric Space 5
2.2 Cauchy sequence 5
2.3 Complete Metric Space 5
2.4 Fractal 5
2.5 Feedback loop 6
2.6 Banach Fixed Point Theorem 6

3 APPLICATION **7**
3.1 Problem Statement 7
3.2 Principle 8
3.3 Mathematics Behind : Contractions for Images 8

4 FRACTAL IMAGE COMPRESSION **10**
4.1 Analogue 11
4.2 Affine Transformation 12
4.3 Working Process 13

4.4 Encoding Images 14
4.5 Decoding 15

5 MATHEMATICA : EXAMPLE SOLVING 15

6 CONCLUSION 17

7 REFERENCES 18

1 INTRODUCTION

With the advance of the information age the need for mass information storage and fast communication links grows. Storing images in less memory leads to a direct reduction in storage cost and faster data transmissions. These facts justify the efforts, of private companies and universities, on new image compression algorithms. Images are stored on computers as collections of bits (a bit is a binary unit of information which can answer "yes" or "no" questions) representing pixels or points forming the picture elements. Since the human eye can process large amounts of information (some 8 million bits), many pixels are required to store moderate quality images. These bits provide the "yes" and "no" answers to the 8 million questions that determine the image. Most data contains some amount of redundancy, which can sometimes be removed for storage and replaced for recovery, but this redundancy does not lead to high compression ratios. An image can be changed in many ways that are either not detectable by the human eye or do not contribute to the degradation of the image. The standard methods of image compression come in several varieties. The current most popular method relies on eliminating high frequency components of the signal by storing only the low frequency components (Discrete Cosine Transform Algorithm). This method is used on JPEG (still images), MPEG (motion video images), H.261 (Video Telephony on ISDN lines), and H.263 (Video Telephony on PSTN lines) compression algorithms. Fractal Compression was first promoted by M.Barnsley The purpose of the image compression is to remove these redundancies and thereby to reduce the size of the image to represent it for the suitable application. Broadly, all the image compression techniques are categorized as lossless and lossy. In lossless image compression, an image is reversible and in lossy image compression techniques it is irreversible. There are many popular image compression techniques.

2 THEORY

2.1 Metric Space

A distance on a set K is a function d : K $\times$ K $\rightarrow$ R satisfying

1. For all P, Q $\in$ K, d(P, Q) $\geq$ 0;
2. d(P, Q) = 0 if and only if P = Q;
3. For all P, Q, R $\in$ K, d(P, Q) $\leq$ d(P, R) + d(R, Q);

2.2 Cauchy sequence

1. A sequence P_n of elements of a metric space K is a Cauchy sequence if $\forall\ \epsilon > 0, thereexistsN \in$ N such that for all n, m ¿ N, then
$$d(P_n, P_m) < \epsilon$$
2. A sequence P_n of elements of a metric space K converges to a limit A $\in$ K if for all ϵ ¿ 0, there exists N $\in$ N such that for all n ¿ N, then
$$d(P_n, A) < \epsilon$$

2.3 Complete Metric Space

A metric space K is a complete metric space if any Cauchy sequence P_n of elements of K converges to an element A of K.

2.4 Fractal

Mandelbrot first coined the term fractal. Roughly a fractal is a geometric shape, every part of which is a reduced copy of the whole. The following are examples of fractals.

2.5 Feedback loop

A feedback loop is the part of a system in which some portion (or all) of the system's output is used as input for future operations.

2.6 Banach Fixed Point Theorem

Let K be a complete metric space in which the distance between two points P and Q is denoted d(P, Q). And let F : K $\rightarrow$ K be a contraction, i.e. there exists c $\in$ (0, 1) such that for all P, Q $\in$ K, then

$$d(F(P), F(Q)) \leq cd(P, Q)$$

Then F has a unique fixed point, i.e. there exists a unique A $\in$ K such that F(A) = A.

Proof

First, let us proof that the sequence (u_n) defined by $\{$ is convergent for all $x \in E$.

For all $m < n \in N$:

$$
\begin{aligned}
d(u_m, u_n) &= d(f^m(x), f^n(x)) \\
&\leq s^m d(x, f^{n-m}(x)) \text{ because } f \text{ is a contraction} \\
&\leq s^m \left(\sum_{i=0}^{n-m-1} d(f^i(x), f^{i+1}(x)) \right) \text{ by the triangular inequality} \\
&\leq s^m \left(\sum_{i=0}^{n-m-1} s^i d(x, f(x)) \right) \text{ because } f \text{ is a contraction} \\
&= s^m \left(\frac{1 - s^{n-m}}{1 - s} d(x, f(x)) \right) \\
&\leq \frac{s^m}{1-s} d(x, f(x)) \underset{m \to \infty}{\rightarrow} 0
\end{aligned}
$$

So (u_n) is a Cauchy sequence and as E is complete, (u_n) is convergent. Let x_0 bet its limit.

Moreover, as a contraction is Lipschitz-continuous, it is also continuous so $f(u_n) \rightarrow f(x_0)$. Thus, if we let n tend to infinity in $u_{n+1} = f(u_n)$, we have that $x_0 = f(x_0)$. So x_0 is a fixed point of f.

We have shown that f has a fixed point. Let us show by contradiction that it is unique. Let y_0 be another fixed point, then:

$$d(x_0, y_0) = d(f(x_0), f(y_0)) \leq sd(x_0, y_0) < d(x_0, y_0)$$

Contradiction.

3 APPLICATION

3.1 Problem Statement

The main problem with the digital images is that the large number of bytes are essential to represent them. For example a digital image of size 1920 × 1200 with 24 bits per pixel requires about 1.46 MB of a computer memory. Transmission of the image using a 19200 bits per second modem takes 5.5 minutes. This much time consumption is off course not affordable for many applications.

3.2 Principle

The original image is segmented into parts such that each part is nearly same as a reduced copy of the original image. The union of all the segments is then close enough to the original image. Thus the images with global self similarity are encoded with extreme efficiency This method, introduced by Barnsley, has been called iterated function systems. The underlying idea of the method is to approximate an image by geometric objects.

3.3 Mathematics Behind : Contractions for Images

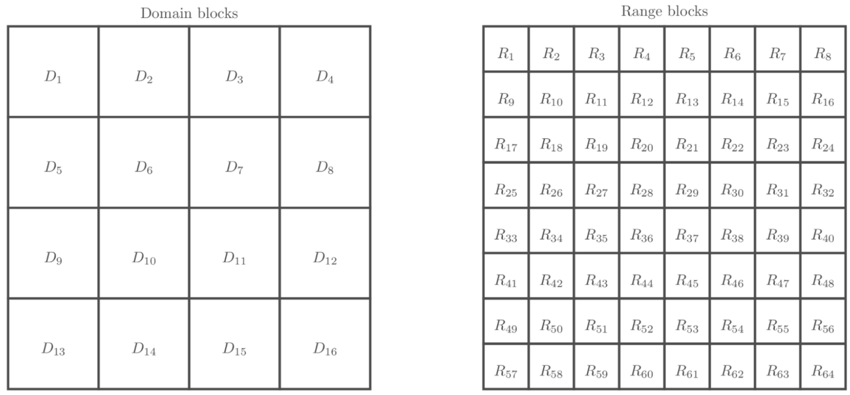
Firstly, let us define the image set and a distance. We choose $E = [0,1]^{h\times w}$. E is the set of matrices with h rows, w columns and with coefficients in $[0,1]$. Then we take

$$d(x,y) = \left(\sum_{i=1}^{h} \sum_{j=1}^{w} (x_{ij} - y_{ij})^2 \right)^{0.5} .d$$

is the distance obtained from the Frobenius norm.

Now, let $x \in E$ the image we want to compress. We will segment twice the image in blocks:

- Firstly, we partition the image in destination or range blocks $R_1, ..., R_L$. These blocks are disjoint and they cover the whole image.
- Then, we segment the image in source or domain blocks $D_1, ..., D_K$. These blocks are not necessarily disjoint and neither they necessarily cover the image.



Then, for each range block R_l, we will choose a domain block D_{k_l} and a mapping $f_l : [0,1]^{D_{k_l}} \to [0,1]^{R_l}$.

Finally, we can define our function f as:

$$f(x)_{ij} = f_l(x_{D_{k_l}})_{ij} \text{ if } (i,j) \in R_l$$

Claim: If all f_l are contractions then f is a contraction.

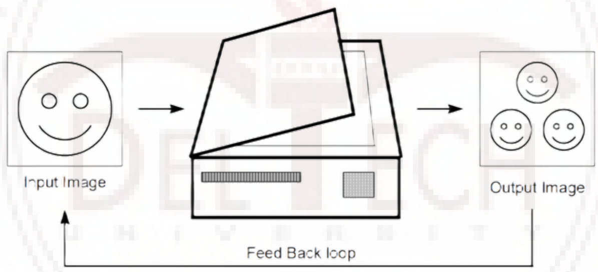
$$
\begin{aligned}
d(f(x), f(y))^2 &= \sum_{i=1}^{h} \sum_{j=1}^{w} (f(x)_{ij} - f(y)_{ij})^2 \text{ by definition of } d \\
&= \sum_{l=1}^{L} \sum_{(i,j)\in R_l} (f(x)_{ij} - f(y)_{ij})^2 \text{ because } (R_l) \text{ is a partition} \\
&= \sum_{l=1}^{L} \sum_{(i,j)\in R_l} (f_l(x_{D_{k_l}})_{ij} - f_l(y_{D_{k_l}})_{ij})^2 \text{ by definition of } f \\
&= \sum_{l=1}^{L} d(f_l(x_{D_{k_l}}), f_l(y_{D_{k_l}}))^2 \text{ by definition of } d \\
&\leq \sum_{l=1}^{L} s_l^2 d(x_{D_{k_l}}, y_{D_{k_l}})^2 \text{ because the } (f_l) \text{ are contractions} \\
&\leq \max_l s_l^2 \sum_{l=1}^{L} d(x_{D_{k_l}}, y_{D_{k_l}})^2 \\
&= \max_l s_l^2 \sum_{l=1}^{L} \sum_{(i,j)\in R_l} (x_{ij} - y_{ij})^2 \text{ by definition of } d \\
&= \max_l s_l^2 \sum_{i=1}^{h} \sum_{j=1}^{w} (x_{ij} - y_{ij})^2 \text{ because } (R_l) \text{ is a partition} \\
&= \max_l s_l^2 d(x, y)^2 \text{ by definition of } d
\end{aligned}
$$

4 FRACTAL IMAGE COMPRESSION

Fractal Compression was first promoted by M.Barnsley, who founded a company based on fractal image compression technology. Barnsley suggested that perhaps storing images as collections of transformations could lead to image compression.

4.1 Analogue

- Imagine a special type of photocopying machine that reduces the image to be copied by half and reproduces it three times on the copy
- We can observe that all the copies seem to converge to the same final image, the one in
- Since the copying machine reduces the input image, any initial image placed on the copying machine will be reduced to a point as we repeatedly run the machine



machine that makes three reduced copies of

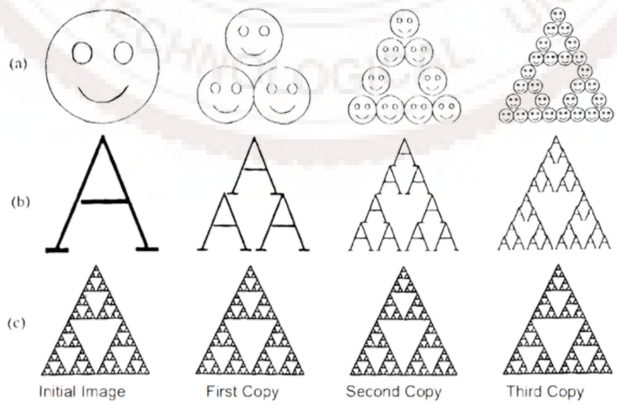


Figure 2: The first three copies generated on the copying machine Figure 1. [Y]

4.2 Affine Transformation

Affline Transformation 1

Any transformation that preserves collinearity (i.e., all points lying on a line initially still lie on a line after transformation) and ratios of distances (e.g., the midpoint of a line segment remains the midpoint after transformation) is called affline .

A transformations that run in a loop back mode is that for a given initial image each image is formed from a transformed (and reduced) copies of itself, and hence it must have detail at every scale. That is, the images are fractals.

we must constrain these transformations, with the limitation that the transformations must be contractive , that is, a given transformation applied to any two points in the input image must bring them closer in the copy.

we choose transformations of the form

$$\begin{aligned} T_1(x,y) &= \left(\frac{x}{2}, \frac{y}{2}\right), \\ T_2(x,y) &= \left(\frac{x}{2} + \frac{1}{2}, \frac{y}{2}\right), \\ T_3(x,y) &= \left(\frac{x}{2} + \frac{1}{4}, \frac{y}{2} + \frac{1}{2}\right). \end{aligned}$$

s sufficient to generate interesting transformations called affine transformations of the plane. Each can skew, stretch, rotate, scale and translate an input image.

A common feature of these transformations that run in a loop back mode is that for a given initial image each image is formed from a transformed (and reduced) copies of itself, and hence it must have detail at every scale. That is, the images are fractals

argument is that as follows: the image looks complicated yet it is generated from only 4 affine transformations

Each transformation wi is defined by 6 numbers, ai, bi, ci, di, ei, and fi , see eq(1), which do not require much memory to store on a computer (4 transformations x 6 numbers / transformations x 32 bits /number = 768 bits). Storing the image as a collection of pixels, however required much more memory (at least 65,536 bits for the resolution . So if we wish to store a picture of a fern, then we can do it by storing the numbers that define the affine transformations and simply generate the fern whenever we want to see it.
Now suppose that we were given any arbitrary image, say a face. If a small number of affine transformations could generate that face, then it too could be stored compactly.

4.3 Working Process

All the images in nature contain a considerable amount of affine redundancy. The affine redundancy means, large segments of the image look like the small segments of the same image. Large segments are known as domain blocks whereas small segments as range blocks. We can find an affine transformation (a combination of rotation, reflection, scaling and shifting transformation) that transforms a domain block to the suitable range block. The parameters of the transformation constitutes a fractal code. Thus a range block is approximated by applying an affine transformation on suitably chosen domain block. Since the mappings reduces the size of the domain block, it is a contractive mapping. Fractal image compression works as follows:

1. The image is partitioned into non-overlapping range blocks.
2. The same image is partitioned into overlapping domain blocks. Domain blocks are larger in size than the range blocks in order to maintain contractive condition.
3. Finally the image is encoded by using a suitable affine transformation which maps a domain block to a best fitted range block.

4. To achieve the decompression, exactly opposite is done. Inverse affine transform is applied to recover the image. Usually 8 to 9 inverse iterations are applied on the encoded image to decode the image. The iteration starts with any arbitrary image. Successive application of the affine map gives the sequence of images that ultimately converge to a fixed image (by fixed point theorem of Banach).

4.4 Encoding Images

Contraction Transformation 1

A transformation w is said to be contractive if for any two points P1, P2, the distance

$$d(w(P1), w(P2)) \leq sd(P1, P2)$$

for some s $\leq$ 1, where d = distance. This formula says the application of a contractive map always brings points closer together (by some factor less than 1).

Contractive Mapping Fixed Point Theorem 1

This theorem says something that is intuitively obvious: if a transformation is contractive then when applied repeatedly starting with any initial point, we converge to a unique fixed point.

If X is a complete metric space and W: X→X is contractive, then W has a unique fixed point.

Above theorems tells us that transformation W will have a unique fixed point in the space of all images. That is, whatever image (or set) we start with, we can repeatedly apply W to it and we will converge to a fixed image.

Example

Suppose we are given an image f that we wish to encode. This means we want to find a collection of transformations $w_1, w_2, ..., w_N$ and want f to be the fixed point of the map W. In other words, we want to partition f into pieces to which we apply the transformations w_i , and get back the original image f.

4.5 Decoding

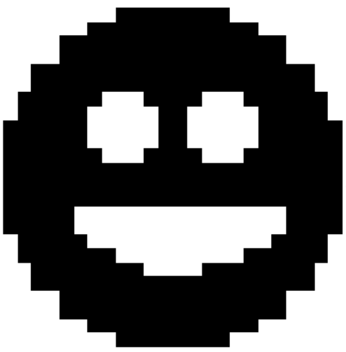
The reconstruction process of the original image consists on the applications of the transformations describe. In other words, we want to partition f into pieces to which we apply the transformations w_i , and get back the original image f.

5 MATHEMATICA : EXAMPLE SOLVING

CODE LINK 1

Detail Code is available at :https://www.wolframcloud.com/obj/44310500-a205-4f60-9437-84a9887c3f53.

In[3]:= imgR =



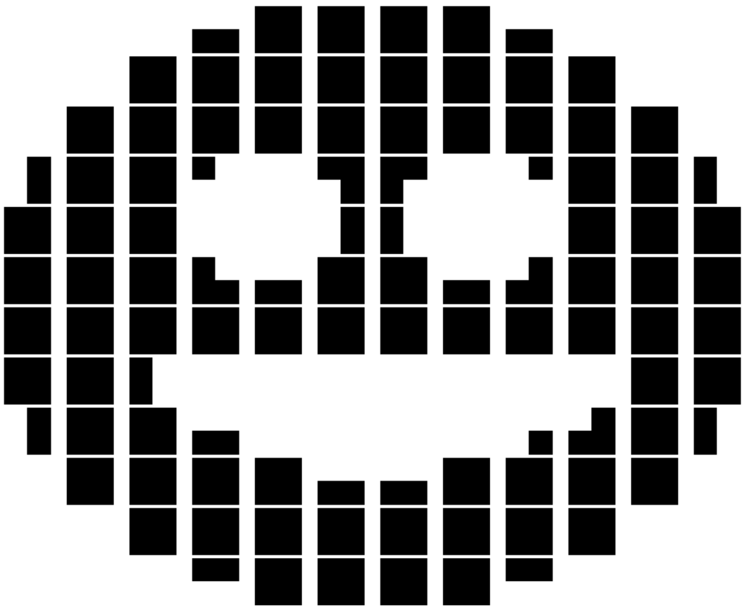
;

```
In[4]:= imgD = ImageResize[imgR, {12, 12}] // Round TexForm
Out[•]= (Round TexForm)[ ]
```

```
In[∘]:= imgRP = ImagePartition[imgR, 2];

In[∘]:= Grid[imgRP, Spacings → {0.8, 0.1}]
```

Out[∘]=

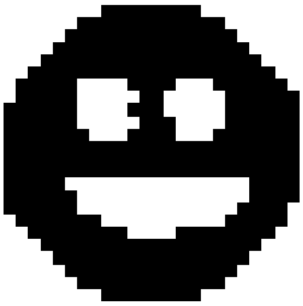


```
In[∘]:= g = GraphicsGrid[Partition[z, 3]]
```

Out[∘]=

```
In[ ]:= gc = Closing[z // Last, DiskMatrix[0.9]] // Round
```

Out[ ]=



6 CONCLUSION

To store images on a computer, there are many compression scheme that do exist to accomplish this task. The main characters in this industry are JPEG, JPEG 200, SVG and PNG. However, there is a fifth one which is interesting because it uses fractals to compress images. What we have experienced inside that vignette is how starting from a simple game we can discover very powerful ideas that may lead to mathematical and technological breakthroughs. When we are looking for a unique solution of a problem, it has now become a standard method in many domains of mathematics to try to see if the solution of the problem can be characterized as the unique fixed point of an operator especially constructed for that purpose

7 REFERENCES

1. M. F. Barnsley, Fractals everywhere, San Diego, Academic Press, 1988

2. R.D. Boss, E.W. Jacobs, “Fractals-Based Image Compression,” NOSC Technical Report 1315, Sept. 1898. Naval Ocean Systems Center, San Diego CA 92152-5000.

3. https://pvigier.github.io/2018/05/14/fractal-image-compression.html

4. https://github.com/popa13/Fractal-Image-Compression

5. ISSN: 2395-4213 (O) Reviewers: (1) Laurence BoxerPDF U P Dolhare, V V Nalawade et al. Asian Journal of Mathematics and Computer Research, 25, 1, 2018