Lecture 35

Iterated Function Systems (cont’d)

Application of Banach’s Fixed Point Theorem to IFS (conclusion)

In the previous lecture, we focussed on the idea of the “parallel operator” \( \hat{f} \) associated with an \( N \)-map iterated function system as a mapping from sets to sets in \( \mathbb{R}^n \). It was therefore necessary to discuss the idea of a distance function or metric between sets in \( \mathbb{R}^n \). Such a metric is the “Hausdorff metric.” We ended the lecture by simply stating the main result, which we repeat below.

**Theorem:** Let \( f \) be an \( N \)-map IFS composed of contraction mappings, \( f_k : D \to D \), \( 1 \leq k \leq N \), where \( D \subset \mathbb{R}^n \). Let \( c_k \in [0, 1) \), \( 1 \leq k \leq N \) denote the contractivity factors of the IFS maps, \( f_k \), and define \( c = \max_{1 \leq k \leq N} c_k \). (Note that \( 0 < c < 1 \).)

Now let \( \mathcal{H}(D) \) denote an appropriate space of subsets of \( D \) which is a complete metric space with respect to the Hausdorff metric \( h \). (Details provided in the Appendix to this lecture.) Let \( \hat{f} \) be the “parallel” operator associated with this IFS defined as follows: For any set \( S \in \mathcal{H}(D) \),

\[
\hat{f}(S) = \bigcup_{k=1}^{N} \hat{f}_k(S).
\]

Then for any two sets \( S_1, S_2 \in \mathcal{H}(D) \),

\[
h(\hat{f}(S_1), \hat{f}(S_2)) \leq c h(S_1, S_2).
\]

In other words, the mapping \( \hat{f} : \mathcal{H}(D) \to \mathcal{H}(D) \) is contractive (with respect to the Hausdorff metric).

**Corollary:** From Banach’s Fixed Point Theorem, there exists a unique set \( A \in \mathcal{H}(D) \) which is the fixed point of \( \hat{f} \), i.e.,

\[
A = \hat{f}(A) = \bigcup_{k=1}^{N} \hat{f}_k(A).
\]

The set \( A \) is the attractor of the IFS. Furthermore, it is self-similar in that it may be expressed as a union of contracted copies of itself.

The Appendix at the end of the notes for this lecture contains copies of handwritten notes by the instructor (from a course taught by him years ago) in which a proof of the above Theorem is presented. A number of results must be proved on the way, however. These, as well as the final proof,
are rather detailed and technical, and are presented for purposes of information. They are considered to be supplementary to the course.

**Using IFS attractors to approximate sets, including natural objects**

*Note:* Much of the following section was taken from the instructor’s article, *A Hitchhiker’s Guide to “Fractal-Based” Function Approximation and Image Compression*, a slightly expanded version of two articles which appeared in the February and August 1995 issues of the UW Faculty of Mathematics Alumni newspaper, *Math Ties*. It may be downloaded from the instructor’s webpage.

As a motivation for this section, we revisit the “spleenwort fern” attractor, shown below, due to Prof. Michael Barnsley and presented earlier in the course (Week 12, Lecture 33).

![Spleenwort Fern – the attractor of a four-map IFS in $\mathbb{R}^2$.](image)

As mentioned later in that lecture, with the creation of these fern-type attractors in 1984 came the idea of using IFS to approximate other shapes and figures occurring in nature and, ultimately, images in general. The IFS was seen to be a possible method of *data compression*. A high-resolution picture of a shaded fern normally requires on the order of one megabyte of computer memory for storage. Current compression methods might be able to cut this number by a factor of ten or so. However,
as an attractor of a four map IFS with probabilities, this fern may be described totally in terms of only 28 IFS parameters! This is a staggering amount of data compression. Not only are the storage requirements reduced but you can also send this small amount of data quickly over communications lines to others who could then “decompress” it and reconstruct the fern by simply iterating the IFS “parallel” operator \( \hat{f} \).

However, not all objects in nature – in fact, very few – exhibit the special self-similarity of the spleenwort fern. Nevertheless, as a starting point there remains the interesting general problem to determine how well sets and images can be approximated by the attractors of IFS. We pose the so-called **inverse problem** for geometric approximation with IFS as follows:

Given a “target” set \( S \), can one find an IFS \( f = \{f_1, f_2, \ldots, f_N\} \) whose attractor \( A \) approximates \( S \) to some desired degree of accuracy in an appropriate metric “\( D \)” (for example, the Hausdorff metric \( h \))?

At first, this appears to be a rather formidable problem. How does one start? By selecting an initial set of maps \( \{f_1, f_2, \ldots, f_N\} \), iterating the associated parallel operator \( \hat{f} \) to produce its attractor \( A \) and then comparing it to the target set \( S \)? And then perhaps altering some or all of the maps in some ways, looking at the effects of the changes on the resulting attractors, hopefully zeroing in on some final IFS?

If we step back a little, we can come up with a strategy. In fact, it won’t appear that strange after we outline it, since you are already accustomed to looking at the self-similarity of IFS attractors, e.g., the Sierpinski triangle in this way. Here is the strategy.

Given a target set \( S \), we are looking for the attractor \( A \) of an \( N \)-map IFS \( f \) which approximates it well, i.e.,

\[
S \approx A. \tag{4}
\]

By “\( \approx \)”, we mean that the \( S \) and \( A \) are “close” – for the moment “visually close” will be sufficient. Now recall that \( A \) is the attractor of the IFS \( f \) so that

\[
A = \bigcup_{k=1}^{N} \hat{f}_k(A). \tag{5}
\]
Substitution into Eq. (4) yields

\[ S \approx \bigcup_{k=1}^{N} \hat{f}_k(A). \]  

(6)

But we now use Eq. (4) to replace \( A \) on the RHS and arrive at the final result,

\[ S \approx \bigcup_{k=1}^{N} \hat{f}_k(S). \]  

(7)

In other words, in order to find an IFS with attractor \( A \) which approximates \( S \), we look for an IFS, i.e., a set of maps \( f = \{f_1, f_2, \cdots f_n\} \), which, under the parallel action of the IFS operator \( \hat{f} \), map the target set \( S \) as close as possible to itself. In this way, we are expressing the target set \( S \) as closely as possible as a union of contracted copies of itself.

This idea should not seem that strange. After all, if the set \( S \) is self-similar, e.g., the attractor of an IFS, then the approximation in Eq. (7) becomes an equality.

The basic idea is illustrated in the figure below. At the left, a leaf – enclosed with a solid curve – is viewed as an approximate union of four contracted copies of itself. Each smaller copy is obtained by an appropriate contractive IFS map \( f_i \). If we restrict ourselves to affine IFS maps in the plane, i.e. \( f_i(x) = Ax + b \), then the coefficients of each matrix \( A \) and associated column vector \( b \) – a total of six unknown coefficients – can be obtained from a knowledge of where three points of the original leaf \( S \) are mapped in the contracted copy \( \hat{f}_i(S) \). We then expect that the attractor \( A \) of the resulting IFS \( \hat{f} \) lies close to the target leaf \( S \). The attractor \( A \) of the IFS is shown on the right.

In general, the determination of optimal IFS maps by looking for approximate geometric self-similarities in a set is a very difficult problem with no simple solutions, especially if one wishes to automate the process. Fortunately, we can proceed by another route by realizing that there is much more to a picture than just geometric shapes. There is also shading. For example, a real fern has veins which may be darker than the outer extremeties of the fronds. Thus it is more natural to think of a picture as defining a function: At each point or pixel \((x, y)\) in a photograph or a computer display (represented, for convenience, by the region \( X = [0,1]^2 \)) there is an associated grey level \( u(x,y) \), which may assume a finite nonnegative value. (In practical applications, i.e. digitized images, each pixel can assume one of only a finite number of discrete values.) In the next section, we show one way in which shading can be produced on IFS attractors. It won’t, however, be the ideal method of performing image approximation. A better method for images will involve a “collaging” of the graphs of functions, leading to an effective method of approximating and compressing images. This will be
Approximating a leaf as a “collage”, i.e. a union of contracted copies of itself. The attractor $A$ of the four-map IFS obtained from the “collage” procedure on the left.

discussed in the next lecture.

Iterated Function Systems with Probabilities

Let us recall our definition of an iterated function system from the past couple of lectures:

**Definition (Iterated function system (IFS)):** Let $f = \{f_1, f_2, \ldots, f_N\}$ denote a set of $N$ contraction mappings on a closed and bounded subset $D \subset \mathbb{R}^n$, i.e., for each $k \in \{1, 2, \ldots, N\}$, $f_k : D \to D$ and there exists a constant $0 \leq C_k < \infty$ such that

$$d(f_k(x), f_k(y)) \leq C_k d(x, y) \quad \text{for all } x, y \in D.$$  

(8)

Associated with this set of contraction mappings is the “parallel set-valued mapping” $\hat{f}$, defined as follows: For any subset $S \subset D$,

$$\hat{f}(S) = \bigcup_{k=1}^{n} \hat{f}_k(S),$$

(9)

where the $\hat{f}_k$ denote the set-valued mappings associated with the mappings $f_k$. The set of maps $f$ with parallel operator $\hat{f}$ define an $N$-map **iterated function system** on the set $D \subset \mathbb{R}^n$. 

414
Theorem: There exists a unique set \( A \subset D \) which is the “fixed point” of the parallel IFS operator \( \hat{f} \), i.e.,
\[
A = \hat{f}(A) = \bigcup_{k=1}^{N} \hat{f}_k(A) .
\] (10)
Consequently, the set \( A \) is self-similar, i.e., \( A \) is the union of \( N \) geometrically-contracted copies of itself.

We’re now going to return to an idea that was used in Problem Set No. 5 to introduce you to IFS, namely the association of a set of probabilities \( p_i \) with the IFS maps \( f_i \), as defined below.

Definition (Iterated function system with probabilities (IFSP)): Let \( f = \{f_1, f_2, \cdots, f_N\} \) denote a set of \( N \) contraction mappings on a closed and bounded subset \( D \subset \mathbb{R}^n \), i.e., for each \( k \in \{1, 2, \cdots, N\} \), \( f_k : D \to D \) and there exists a constant \( 0 \leq C_k < \) such that
\[
d(f_k(x), f_k(y)) \leq C_k d(x, y) \quad \text{for all } x, y \in D .
\] (11)
Associated with each map \( f_k \) is a probability \( p_k \in [0, 1] \) such that
\[
\sum_{k=1}^{N} p_k = 1 .
\] (12)
Then the set of maps \( f \) with associated probabilities \( p = (p_1, p_2, \cdots, p_N) \) is known as an \( N \)-map iterated function systems with probabilities on the set \( D \subset \mathbb{R}^n \) and will be denoted as \((f, p)\).

As before, the “IFS part” of an IFSP, i.e., the maps \( f_k, 1 \leq k \leq N \), will determine an attractor \( A \) that satisfies the self-similarity property in Eq. (10). But what about the probabilities \( p_k \)? What role do they play?

The answer is that they will uniquely determine a measure that is defined on the attractor \( A \). It is beyond the scope of this course to discuss measures, which are intimately related to the theory of integration, in any detail. (As such, you are referred to a course such as PMATH 451, “Measure and Integration.”) Here we simply mention how these measures can be visualized with the help of the random iteration algorithm that you examined in Problem Set No. 5. We actually have encountered
measures earlier in this course – although they weren’t mentioned explicitly – when we examined the
distribution of iterates of a chaotic dynamical system. And this is how we are going to discuss them
in relation to iterated function systems with probabilities. They will determine the distribution of
iterates produced by the random iteration algorithm, which is also known as the “Chaos Game”.

Example No. 1: To illustrate, we consider the following two-map IFS on the interval \([0, 1]\):

\[ f_1(x) = \frac{1}{2} x, \quad f_2(x) = \frac{1}{2} x + \frac{1}{2}. \tag{13} \]

It should not be difficult to see that the attractor of this IFS is the interval \(X = [0, 1]\) since

\[ \hat{f}_1 : [0, 1] \to [0, \frac{1}{2}], \quad \hat{f}_2 : [0, 1] \to [\frac{1}{2}, 1], \tag{14} \]

so that

\[ \hat{f}([0, 1]) = \hat{f}_1([0, 1]) \cup \hat{f}_2([0, 1]) \]
\[ = [0, \frac{1}{2}] \cup [\frac{1}{2}, 1] \]
\[ = [0, 1]. \tag{15} \]

We now let \(p_1\) and \(p_2\) be probabilities associated with the IFS maps \(f_1\) and \(f_2\), respectively, such that

\[ p_1 + p_2 = 1. \tag{16} \]

We now consider the following random iteration algorithm involving these two maps and their prob-
abilities: Starting with an \(x_0 \in [0, 1]\), define

\[ x_{n+1} = f_{\sigma_n}(x_n), \quad \sigma_n \in \{1, 2\}, \tag{17} \]

where \(\sigma_n\) is chosen from the set \(\{1, 2\}\) with probabilities \(p_1\) and \(p_2\) respectively, i.e.,

\[ P(\sigma_n = 1) = p_1 \quad P(\sigma_n = 2) = p_2. \tag{18} \]

Case No. 1: Equal probabilities, i.e., \(p_1 = p_2 = \frac{1}{2}\).

At each step of the algorithm in Eq. (17) there is an equal probability of choosing map \(f_1\) or \(f_2\).
No matter where we start, i.e., what \(x_0\) is, there is a 50% probability that \(x_1\) will be located in \([0, \frac{1}{2}]\)
and a 50% probability that it will be located in \([\frac{1}{2}, 1]\). And since this is the case, there should be a
50% probability that \( x_2 \) will be located in \([0, \frac{1}{2}]\), etc.

Let us now perform the following experiment, very much like the one that we performed to analyze the distribution of iterates of chaotic dynamical systems. Once again, for an \( n \) sufficiently large, we divide the interval \([0, 1]\) into \( n \) subintervals \( I_k \) of equal length using the partition points,

\[
x_k = k\Delta x, \quad 0 \leq k \leq n, \quad \text{where} \quad \Delta x = \frac{1}{n}.
\]

(19)

(In our section on chaotic dynamical systems, we used “\( N \)” instead of \( n \). Unfortunately, \( N \) is now reserved for the number of maps in our IFS.) We now run the random iteration algorithm in (17) for a large number of iterations, \( M \), counting the number of times, \( n_k \), that each subinterval \( I_k \) is visited. We then define the following numbers,

\[
p_k = \frac{n_k}{M}, \quad 1 \leq k \leq n.
\]

(20)

which are once again the fraction of total iterates \( \{x_n\}_{n=1}^M \) found in each subinterval \( I_k \).

In the figure below we present a plot of the \( p_k \) obtained using a partition of \( n = 1000 \) points and \( M = 10^7 \) iterates.

![Plot of p_k](image)

The distribution of the iterates \( x_n \) seems to be quite uniform, in accordance with our earlier discussion.
**Case No. 2:** Unequal probabilities, e.g., \( p_1 = \frac{2}{5}, p_2 = \frac{3}{5} \).

At each step of the algorithm in Eq. (17) there is now a greater probability of choosing map \( f_2 \) over \( f_1 \). No matter where we start, i.e., what \( x_0 \) is, there is a 60% probability of finding \( x_1 \) in the interval \([\frac{1}{2}, 1]\) and a 40% probability of finding \( x_1 \) in the interval \([0, \frac{1}{2}]\). As such, we might expect the distribution of the iterates to be somewhat “slanted” toward the right, possibly like this,

![Graph 1](image1)

But wait! If there is a higher probability of finding an iterate in the second half-interval \([\frac{1}{2}, 1]\) than the first, this is going to make itself known in each half interval as well. For example, there should be a higher probability of finding an iterate in the subinterval \([\frac{1}{4}, \frac{1}{2}]\) than in the subinterval \([0, \frac{1}{4}]\), etc., something like this,

![Graph 2](image2)
The reader should begin to see that there is no end to this analysis. There should be higher and lower probabilities for the one-eighth intervals, something like this,

If we run the random iteration algorithm for this case, using a partition of \( n = 1000 \) points and \( M = 10^7 \) iterates, the following plot of the \( p_k \) fractions is obtained.

This is a histogram approximation of the so-called invariant measure, which we’ll denote as \( \bar{\mu} \) and which is associated with the IFS,

\[
f_1(x) = \frac{1}{2} x, \quad f_2(x) = \frac{1}{2} x + \frac{1}{2},
\]  

with associated probabilities,

\[
p_1 = \frac{2}{5}, \quad p_2 = \frac{3}{5}.
\]  

(21)
**Example No. 2:** We now consider the following two-map IFS on \([0, 1]\),

\[
    f_1(x) = \frac{3}{5}x, \quad f_2(x) = \frac{3}{5}x + \frac{2}{5}.
\]  

The fixed point of \(f_1\) is \(\bar{x}_1 = 0\). And the fixed point of \(f_2\) is \(\bar{x}_2 = 1\). The attractor of this IFS is once again the interval \([0, 1]\) since

\[
    \hat{f}_1 : [0, 1] \rightarrow \left[0, \frac{3}{5}\right], \quad \hat{f}_2 : [0, 1] \rightarrow \left[\frac{2}{5}, 1\right],
\]

so that

\[
    \hat{f}(0, 1) = \hat{f}_1([0, 1]) \cup \hat{f}_2([0, 1]) = \left[0, \frac{3}{5}\right] \cup \left[\frac{2}{5}, 1\right] = [0, 1].
\]

This doesn’t seem to be so interesting, but the fact that the images of \([0, 1]\) under the action of \(f_1\) and \(f_2\) overlap will make things interesting in terms of the underlying invariant measure.

Let us once again assume equal probabilities, i.e., \(p_1 = p_2 = \frac{1}{2}\). Given an \(x_0 \in [0, 1]\), there is an equal probability of choosing either \(f_1\) or \(f_2\) to apply to \(x_0\). This means that there is an equal opportunity of finding \(x_1\) in \([0, \frac{3}{5}]\) and \([\frac{2}{5}, 1]\). But notice now that these two intervals OVERLAP. This means that there are two ways for \(x_0\) to get mapped to the interval \([\frac{2}{5}, \frac{3}{5}]\). This implies that there should be a slightly greater probability of finding \(x_1\) in this subinterval than in the rest of \([0, 1]\), something like this,

![Diagram](image-url)

But this means that the middle parts of the smaller subintervals will be visited more often than their outer parts, etc.. If we run the random iteration algorithm for \(n = 1000\) and \(M = 10^7\) iterates, the following distribution is obtained.
This is a histogram approximation of the so-called invariant measure, $\bar{\mu}$, associated with the IFS,

$$f_1(x) = \frac{3}{5}x, \quad f_2(x) = \frac{3}{5}x + \frac{2}{5}. \quad (26)$$

with associated probabilities,

$$p_1 = p_2 = \frac{1}{2}. \quad (27)$$

Following the landmark papers by J. Hutchinson and M. Barnsley/S. Demko (see references at the end of the next lecture), researchers viewed invariant measures of IFSP as a way to produce “shading” in a set. After all, a set itself is not a good way to represent an object, unless one is interested only in its shape, in which case a binary, black-white, image is sufficient. That being said, if one is going to use IFSP measures to approximate shaded objects, then one would like to have a little more control on the shading than what is possible by the invariant measure method shown above. Such methods to achieve more control essentially treat measures more like functions. As such, it is actually advantageous to devise IFS-type methods on functions, which is the subject of the next lecture.
Note: We are going to want the Hausdorff distance \( h \) to be a metric in an appropriate space. From previous discussions, e.g. the Cantor set on \([0,1]\), it would appear that this space would consist of all nonempty subsets of \( X \). However, it is desirable that \( h \) be a metric and not a pseudo-metric. For example:
\[
 h([0,1], [0,1]) = h([0,1], (0,1]) = h([0,1], [0,1]) = \ldots
\]

For this reason, as well as the fact that the usual "fractal" sets are closed, it would seem desirable to consider only closed subsets. However, questions of convergence of sets are also involved. For this reason, the sets should be compact.

Let \((X,d)\) be a compact metric space. Let \(\mathcal{H}(x)\) denote the set of all nonempty compact subsets of \(X\).

Then: \((\mathcal{H}(x), h)\) is a complete metric space.

(see Barnsley, "Fractals Everywhere", Sects. 2.6, 2.7; Falmer, "Geometry of Fractal Sets")

We have now arrived at our "appropriate" metric space \((X,d)\) for the IFS set-valued mapping \( W \) associated with \( W = (w_1, w_2, \ldots, w_n) \), \( w_i \in \mathcal{C} \mathcal{R}_n(x) \).

Note that "points" in \(\mathcal{H}(x)\) are nonempty compact subsets of \(X\) (e.g. Cantor set \( C \) on \([0,1]\)).
Now, to establish the connection between \( \mathcal{H}(x) \) and contraction maps on \( X \). From our observation on the actions of \( \hat{w}_1 \) and \( \hat{w}_2 \) on subsets of \( [0,1] \), we expected the following result:

**Theorem:** 
\((X,d)\) complete metric space; \((\mathcal{H}(x), \mathcal{h})\) as defined above. Let \( f \in \text{Con}(x) \) and define \( \hat{f}(s) = \{ f(x), \forall x \in s \} \).

Then:
1. \( \hat{f} : \mathcal{H}(x) \to \mathcal{H}(x) \)
2. \( \hat{f} \) is a contraction mapping on \((\mathcal{H}(x), \mathcal{h})\).

**Proof:**
1. Let \( A \subseteq \mathcal{H}(x) \), i.e. \( A \) is nonempty and compact subset of \( X \). Then \( \hat{f}(A) \) is nonempty. Now show that \( \hat{f}(A) \) is compact. Let \( \{x_n\}_{n=1}^{\infty} \) be an infinite sequence of points in \( A \) and let \( y_n = f(x_n) \). Since \( A \) is compact, there exists an infinite subsequence \( \{x_{i_m}\}_{m=1}^{\infty} \), denoted by \( \{x_{i_m}\}_{m=1}^{\infty} \), which possesses a limit point \( x' \in A \), i.e. \( \lim_{m \to \infty} x_{i_m} = x' \).

But since \( f \) is continuous (since \( f \in \text{Con}(x) \)), it follows that
\[
\lim_{m \to \infty} y_{i_m} = \lim_{m \to \infty} f(x_{i_m}) = f\left( \lim_{m \to \infty} x_{i_m} \right) = f(x').
\]
Since \( x' \in A \), it follows that \( y' = \lim_{m \to \infty} y_{i_m} = f(x') \) lies in \( \hat{f}(A) \).

Thus, the infinite sequence \( y_n \in \hat{f}(A) \) contains a subsequence which converges to a point in \( \hat{f}(A) \). Therefore \( \hat{f}(A) \) is compact.

Now to show that \( \hat{f} \) is contractive.
let $A, B \in \mathcal{H}(x)$. Recall that
\[
\hat{h}(A, B) = \max \left[ \sup_{x \in A} d(x, B), \sup_{y \in B} d(y, A) \right]
\]

Now consider $\hat{h}(\hat{f}(A), \hat{f}(B))$:
\[
\hat{h}(\hat{f}(A), \hat{f}(B)) = \max \left[ \sup_{x \in \hat{f}(A)} d(x, \hat{f}(B)), \sup_{y \in \hat{f}(B)} d(y, \hat{f}(A)) \right]
\]

Look at the first entry:
\[
\sup_{x \in \hat{f}(A)} d(x, \hat{f}(B)) = \sup_{x \in \hat{f}(A)} \inf_{y \in \hat{f}(B)} d(x, y) \quad x \in \hat{f}(A) \Rightarrow x = f(p), p \in A
\]
\[
= \sup_{p \in A} \inf_{q \in B} d(f(p), f(q))
\]
\[
\leq \sup_{p \in A} \inf_{q \in B} d(p, q)
\]
\[
= c f \sup_{p \in A} d(p, B)
\]

Similarly:
\[
\sup_{y \in \hat{f}(B)} d(y, \hat{f}(A)) \leq c f \sup_{q \in B} d(q, A)
\]

Combining these results we have
\[
\hat{h}(\hat{f}(A), \hat{f}(B)) \leq c f \max \left[ \sup_{p \in A} d(p, B), \sup_{q \in B} d(q, A) \right]
\]
\[
= c f \hat{h}(A, B).
\]

Since $c f \in [0, 1)$, it follows that $\hat{f}$ is contraction in $(\mathcal{H}(x), \hat{d})$. \hfill \blacksquare
Examples: Return to $X = [0, 1]$, $W_1(x) = \frac{1}{2}x$. Since $W_1 \in C_X(x)$, it follows, from previous theorem, that the associated set-valued mapping $\hat{W}_1$ is contractive on $(X(x), k)$.

Therefore, there exists a unique set $A \in X(x)$ such that $\hat{W}_1(A) = A$. We noted earlier that $A = \{0\}$ (which is non-empty and compact).

Recall the iteration sequence $I_0 = [0, 1]$, $I_{n+1} = \hat{W}_1(I_n)$, $n = 0, 1, 2, \ldots$

\[0 \rightarrow 1 \rightarrow 0 \frac{1}{3} \rightarrow 0 \frac{1}{9} \rightarrow \ldots\]

$I_n = [0, \frac{1}{3^n}]$ Note that $\hat{W}_1(I_n, \{0\}) = \frac{1}{3^n} \rightarrow 0$ as $n \rightarrow \infty$.

This type of "interval calculus" gives us information about the behaviour of the iteration sequence $x_0 \in [0, 1]$ $x_{n+1} = W_1(x_n)$ considered earlier.

If $x_0 \in [0, 1] = I_0$, then $x_1 \in \hat{W}_1(I_0) = I_1$. This limits the possible location of $x_1$. Likewise $x_n \in I_n = [0, \frac{1}{3^n}]$. You can see that as $n \rightarrow \infty$, $x_n$ has no other alternative but to approach $\bar{x} = 0$.

Likewise for $W_2(x) = \frac{1}{3}x + \frac{2}{3}$, $W_2(1) = 1$. If $I_0 = [0, 1]$, $I_{n+1} = \hat{W}_2(I_n)$ then $I_n = \left[1 - \frac{1}{3^n}, 1\right]$ and $\hat{W}_2(I_n, \{1\}) = \frac{1}{3^n} \rightarrow 0$ as $n \rightarrow \infty$.

$\hat{W}_2(\{0\}) = \{0\}$. 
Some other important properties that we shall need:

**Lemma 1:** Let $A, B, C \in \mathcal{H}(x)$. Then $d(A \cup B, C) = \max \{d(A, C), d(B, C)\}$

**Proof:**

$$d(A \cup B, C) = \sup_{x \in A \cup B} d(x, C) = \max \{\sup_{x \in A} d(x, C), \sup_{x \in B} d(x, C)\}$$

The distance from $A \cup B$ to $C$ is the greater of the distances from $A$ to $C$ and $B$ to $C$.

**Lemma 2:** $d(C, A \cup B) \leq \min \{d(C, A), d(C, B)\}$

**Proof:**

$$d(C, A \cup B) = \sup_{x \in C} d(x, A \cup B) = \sup_{x \in C} \inf_{y \in A \cup B} d(x, y)$$

Now

$$\sup_{x \in C} \inf_{y \in A \cup B} d(x, y) \leq \sup_{x \in C} \inf_{y \in A} d(x, y) = d(C, A)$$

and

$$\sup_{x \in C} \inf_{y \in A \cup B} d(x, y) \leq \sup_{x \in C} \inf_{y \in B} d(x, y) = d(C, B)$$

Since restricting the set on which an "inf" is taken can push the lower bound upward.

The desired result follows.

From these results, it follows that

$$h(A \cup B, C) \leq \max \{h(A, C), h(B, C)\}$$

**Example:** $X = [0, 1]$  $A = [0, 1/3]$  $B = [1/3, 1]$  $C = [0, 1] = A \cup B$

$h(A \cup B, C) = 0$ but $h(A, C) = 2/3$  $h(B, C) = 1/3$
Lemma 3: \( A, B, C, D \in \mathcal{H}(x) \). Then
\[
\h (A \cup B, C \cup D) \leq \max \left\{ \h (A, C), \h (B, D) \right\}
\]

**Proof:**
\[
\h (A \cup B, C \cup D) = \max \left\{ \h (A \cup B, C \cup D), \h (C \cup D, A \cup B) \right\}
\]
\[
= \max \left\{ \h (A, C \cup D), \h (B, C \cup D), \h (C, A \cup B), \h (D, A \cup B) \right\}
\]

From Lemma 2, \( \h (A, C \cup D) \leq \min \left\{ \h (A, C), \h (A, D) \right\} \leq \h (A, C) \)

Since \( \h (A, C) = \max \left\{ \h (A, C), \h (C, A) \right\} \)

Likewise, \( \h (B, C \cup D) \leq \min \left\{ \h (B, C), \h (B, D) \right\} \leq \h (B, D) \)

\( \h (C, A \cup B) \leq \min \left\{ \h (C, A), \h (C, B) \right\} \leq \h (A, C) \)

\( \h (D, A \cup B) \leq \min \left\{ \h (D, A), \h (D, B) \right\} \leq \h (B, D) \)

The desired result follows.

An inductive argument produces the following important result.

Lemma 4: Let \( A_i, B_i \in \mathcal{H}(x), \ i = 1, 2, \ldots, N \)

Then
\[
\h \left( \bigcup_{i=1}^{N} A_i, \bigcup_{j=1}^{N} B_j \right) \leq \max_{1 \leq i \leq N} \h (A_i, B_i)
\]
We now arrive at the most important result involving IFS in $\mathcal{H}(x)$. It was originally proved by J. Hutchinson ("Fractals and Self-Similarity," Indiana Univ. Math. J. 30 No. 5 (1981), pp. 713-747).

Theorem: Let $(x, d)$ be a complete metric space and $(\mathcal{H}(x), \mathcal{H})$ defined as above. Let $\mathcal{H} \in \text{Con}(x)$ with contraction factors $c_i \in [0, 1)$.

Then the set-valued mapping $\hat{\mathcal{H}} : \mathcal{H}(x) \to \mathcal{H}(x)$, whose action is defined as

$$\hat{\mathcal{H}}(S) = \bigcup_{i=1}^{N} \hat{\mathcal{H}}_i(S), \quad S \in \mathcal{H}(x)$$

is contractive in $(\mathcal{H}(x), \mathcal{H})$, with contraction factor $c = \max_{1 \leq i \leq N} c_i$.

This result is a consequence of Lemma 4 on the previous page and the results of the Theorem on page 28: For $A, B \in \mathcal{H}(x)$,

$$\lambda(\hat{\mathcal{H}}(A), \hat{\mathcal{H}}(B)) = \lambda\left(\bigcup_{i=1}^{N} \hat{\mathcal{H}}_i(A), \bigcup_{j=1}^{N} \hat{\mathcal{H}}_j(B)\right)$$

$$\leq \max_{1 \leq i \leq N} \lambda(\hat{\mathcal{H}}_i(A), \hat{\mathcal{H}}_i(B)) \quad (\text{Lemma 4})$$

$$\leq \max_{1 \leq i \leq N} c_i \lambda(A, B) \quad (\text{Theorem, pg. 29})$$

$$= c \lambda(A, B).$$
Corollary: There exists a unique compact and non-empty set \( A \subseteq \mathcal{H}(x) \), such that

1) \( \hat{W}(A) = A = \bigcup_{i=1}^{n} \hat{W}_i(A) \)

2) \( \chi(\hat{W}_n(s), A) \to 0 \) as \( n \to \infty \) \( \forall s \in \mathcal{H}(x) \).

This result follows immediately from Banach's CFP.

Comments: From 1) \( A \) is a fixed point of \( \hat{W} \).

\( A \) is a union of "shrunken" copies of itself.

"A is tiled with shrunken copies of itself."

From 2) \( A \) is an attractor.

Examples: 1. \( X = [0,1] \)

\[ W_1(x) = \frac{x}{3}, \quad W_2(x) = \frac{x}{3} + \frac{2}{3} \]

\( A = C \) the Cantor middle-3 set in \([0,1]\)

2. \( X = [0,1] \)

\[ W_1(x) = \frac{x}{2}, \quad W_2(x) = \frac{x}{2} + \frac{1}{2} \]

\[ \hat{W} : [0,1] \to [0, \frac{1}{2}] \cup \left[ \frac{1}{2}, 1 \right] = [0,1] \]

\( c_1 = c_2 = \frac{1}{2} \)

\( A = [0,1] \)

3. \( X = [0,1] \)

\[ W_1(x) = Sx, \quad W_2(x) = Sx + (1-S) \quad s \in [0,1) \]

\[ W_1(0) = 0, \quad W_2(1) = 1 \]

\[ c_1 = c_2 = s \]

\[ \hat{W} : [0,1] \to [0, s] \cup [1-s, 1] \]

Case 1: If \( s > \frac{1}{2} \), then \( \hat{W} : [0,1] \to [0,1] \quad A = [0,1] \)

Case 2: If \( 0 < s < \frac{1}{2} \) then \( \hat{W} : \frac{1}{1} \to 0 \quad \overline{0} \)

produces a "sierpinski" in \([0,1]\).

\( A \) is a Cantor-like set in \([0,1]\)

\[ \hat{W} : \frac{1}{0} \quad \frac{1}{s} \to \frac{1}{s} \quad \frac{2}{s} \quad \frac{2}{s} \quad \frac{3}{s} \quad \frac{3}{s} \quad \frac{4}{s} \quad \frac{4}{s} \]

\[ 0 \quad 1 \quad 0 \quad 5 \quad 1 \quad s \quad 1 \quad s \]
4. \( X = [0,1] \quad w_1(x) = s_1 x \quad w_2(x) = s_2 x + (1-s_2) \quad s_1, s_2 \in [0,1] \)

1) \( s_1 + s_2 > 1 \) then \( A = [0,1] \)

2) \( s_1 + s_2 < 1 \)

\[ w : 0 \rightarrow 0 \quad s_1 \quad 1 - s_2 \quad 1 \]

\( A \) is a Cantor-like set.

5. \( X = [0,1]^2 \quad w_1(x,y) = (\frac{1}{2}x, \frac{1}{2}y) \quad w_2(x,y) = (\frac{1}{2}x + \frac{1}{4}, \frac{1}{2}y + \frac{\sqrt{3}}{4}) \quad w_3(x,y) = (\frac{1}{2}x + \frac{1}{4}, \frac{1}{2}y + \frac{\sqrt{3}}{4}) \)

Contractive factors \( c_1 = c_2 = c_3 = \frac{1}{2} \)

\( w_1(0,0) = (0,0), \quad w_2(1,0) = (1,0), \quad w_3 \left( \frac{1}{2}, \frac{\sqrt{3}}{2} \right) = \left( \frac{1}{2}, \frac{\sqrt{3}}{2} \right) \)

Fixed points form vertices of an equilateral triangle \( T \)

Let us examine the action of \( \hat{w} \) on the region \( T \in H(x) \) composed of triangle + interior

\( w_1 : \)

\( w_2 : \)

\( w_3 : \)

Then \( \hat{w} : \)

"middle fourth" missing

Apply \( \hat{w} \) again: \( \hat{w} : \)

The fixed point attractor \( A \) of \( \hat{w} \) is the so-called "Sierpinski gasket"

Note that \( A = w_1(A) \cup w_2(A) \cup w_3(A) \)
Exercise: The "Koch Curve" is constructed by iterating the following procedure:

\[
\begin{align*}
1 & \rightarrow \frac{\sqrt{3}}{3} \quad \frac{\sqrt{3}}{3} \\
\text{i.e., remove (open) middle third of any interval and add two sides with length one-third of interval.}
\end{align*}
\]

The first few generations are:

- \( I_0 = [0,1] \)
- \( I_1 \)
- \( I_2 \)
- \( I_3 \)

Construct an IFS \( \mathcal{W} = (W_1, \ldots, W_n) \) whose attractor \( A = \lim_{n \to \infty} I_n \).

Other notable examples:
- "Sierpinski fern"
- "Twin dragon"
Attractors for IFS \( \{ C, T_x = 5x \pm 1, s \in C \} \)
"MANDELBROT SET" \( \{ z \in C, A(z) \text{ is disconnected} \} \)
Lecture 36

Iterated function systems for functions: “Fractal transforms” and “fractal image coding”

The previous lecture concluded with the comment that we should regard a picture as being more than merely geometric shapes. There is also shading. As such, it is more natural to think of a picture as defining a function: At each point or pixel \((x, y)\) in a photograph – assumed to be black-and-white for the moment – there is an associated “grey level” \(u(x, y)\) which assumes a finite and nonnegative value. (Here, \((x, y) \in X = [0, 1]^2\), for convenience.) For example, consider Figure 1 below, a standard test case in image processing studies named “Boat”. The image is a 512 × 512 pixel array. Each pixel assumes one of 256 shades of grey (0 = white, 255 = black). From the point of view of continuous real variables \((x, y)\), the image is represented as a piecewise constant function \(u(x, y)\). If the grey level value of each pixel is interpreted as a value in the \(z\) direction, then the graph of the image function \(z = u(x, y)\) is a surface in \(\mathbb{R}^3\), as shown on the right. The red-blue spectrum of colours in the plot is used to characterize function values: Higher values are more red, lower values are more blue.

![Figure 1](image)

**Figure 1.** Left: The standard test-image, *Boat*, a 512 × 512-pixel digital image, 8 bits per pixel. Right: The *Boat* image, viewed as a non-negative image function \(z = u(x, y)\).
Our goal is to set up an IFS-type approach to work with non-negative functions \( u : X \rightarrow \mathbb{R}^+ \) instead of sets. Before writing any mathematics, let us illustrate schematically what can be done. For ease of presentation, we consider for the moment only one-dimensional images, i.e. positive real-valued functions \( u(x) \) where \( x \in [0, 1] \). An example is sketched in Figure 2(a). Suppose our IFS is composed of only two contractive maps \( f_1, f_2 \). Each of these functions \( f_i \) will map the “base space” \( X = [0, 1] \) to a subinterval \( \hat{f}_i(X) \) contained in \( X \). Let’s choose

\[
\hat{f}_1(x) = 0.6x, \quad \hat{f}_2(x) = 0.6x + 0.4. \tag{28}
\]

For reasons which will become clear below, it is important that \( \hat{f}_1(X) \) and \( \hat{f}_2(X) \) are not disjoint - they will have to overlap with each other, even if the overlap occurs only at one point.

The first step in our IFS procedure is to make two copies of the graph of \( u(x) \) which are distorted to fit on the subsets \( \hat{f}_1(X) = [0, 0.6] \) and \( \hat{f}_2(X) = [0.4, 1] \) by “shrinking” and translating the graph in the \( x \)-direction. This is illustrated in Figure 2(b). Mathematically, the two “component” curves \( a_1(x) \) and \( a_2(x) \) in Figure 2(b) are given by

\[
a_1(x) = u(f_1^{-1}(x)) \quad x \in \hat{f}_1(X), \quad a_2(x) = u(f_2^{-1}(x)) \quad x \in \hat{f}_2(X), \tag{29}
\]

It is important to understand this equation. For example, the term \( f_1^{-1}(x) \) is defined only for those \( x \in X \) at which the inverse of \( f_1 \) exists. For the inverse of \( f_1 \) to exist at \( x \) means that one must be able to get to \( x \) under the action of the map \( f_1 \), i.e., there exists a \( y \in X \) such that \( f_1(y) = x \). But this means that \( y = f_1^{-1}(x) \). It also means that \( x \in \hat{f}_1(X) \), where

\[
\hat{f}_1(X) = \{ f_1(y) : y \in X \}. \tag{30}
\]

Furthermore, note that since the map \( f_1(x) \) is a contraction map, it follows that the function \( u_1(x) \) is a **contracted** copy of \( u(x) \) which is situated on the set \( \hat{f}_1(X) \). All of the above discussion also applies to the map \( f_2(x) \).

We’re not finished, however, since some additional flexibility in modifying these curves would be desirable. Suppose that are allowed to modify the \( y \) (or grey level) values of each component function \( a_i(x) \). For example, let us

1. multiply all values \( a_1(x) \) by 0.5 and add 0.5,
2. multiply all values \( a_2(x) \) by 0.75.
The modified component functions, denoted as \( b_1(x) \) and \( b_2(x) \), respectively, are shown in Figure 2(c). What we have just done can be written as

\[
\begin{align*}
b_1(x) &= \phi_1(a_1(x)) = \phi_1(u(f_1^{-1}(x))) \quad x \in \hat{f}_1(X), \\
b_2(x) &= \phi_2(a_2(x)) = \phi_2(u(f_2^{-1}(x))) \quad x \in \hat{f}_2(X),
\end{align*}
\]

(31)

where

\[
\phi_1(y) = 0.5y + 0.5, \quad \phi_2(y) = 0.75y, \quad y \in \mathbb{R}^+.
\]

(32)

The \( \phi_i \) are known as grey-level maps: They map (nonnegative) grey-level values to grey-level values.

We now use the component functions \( b_i \) in Figure 2(c) to construct a new function \( v(x) \). How do we do this? Well, there is no problem to define \( v(x) \) at values of \( x \in [0,1] \) which lie in only one of the two subsets \( \hat{f}_i(X) \). For example, \( x_1 = 0.25 \) lies only in \( \hat{f}_1(X) \). As such, we define \( v(x_1) = b_1(x) = \phi_1(u(f_1^{-1}(x))) \). The same is true for \( x_2 = 0.75 \), which lies only in \( \hat{f}_2(X) \). We define \( v(x_2) = b_2(x) = \phi_2(u(f_2^{-1}(x))) \).

Now what about points that lie in both \( \hat{f}_1(X) \) and \( \hat{f}_2(X) \), for example \( x_3 = 0.5 \)? There are two possible components that we may use to define our resulting function \( v(x_3) \), namely \( b_1(x_3) \) and \( b_2(x_3) \). How do we suitably choose or combine these values to produce a resulting function \( v(x) \) for \( x \) in this region of overlap?

To make a long story short, this is a rather complicated mathematical issue and was a subject of research, in particular at Waterloo. There are many possibilities of combining these values, including (1) adding them, (2) taking the maximum or (3) taking some weighted sum, for example, the average. In what follows, we consider the first case, i.e. we simply add the values. The resulting function \( v(x) \) is sketched in Figure 3(a). The observant reader may now be able to guess why we demanded that the subsets \( \hat{f}_1([0,1]) \) and \( \hat{f}_2([0,1]) \) overlap, touching at least at one point. If they didn’t, then the union \( \hat{f}_1(X) \cup \hat{f}_2(X) \) would have “holes”, i.e. points \( x \in [0,1] \) at which no component functions \( a_i(x) \), hence \( b_i(x) \), would be defined. (Remember the Cantor set?) Since want our IFS procedure to map functions on \( X \) to functions on \( X \), the resulting function \( v(x) \) must be defined for all \( x \in X \).
**Figure 2(a):** A sample “one-dimensional image” $u(x)$ on [0,1].

**Figure 2(b):** The component functions given in Eq. (29).

**Figure 2(c):** The modified component functions given in Eq. (31).
The 2-map IFS \( f = \{ f_1, f_2 \}, f_i : X \rightarrow X \), along with associated grey-level maps \( \Phi = \{ \phi_1, \phi_2 \}, \phi_i : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \), is referred to as an **Iterated Function System with Grey-Level Maps** (IFSM), \((f, \Phi)\). What we did above was to associate with this IFSM an operator \( T \) which acts on a function \( u \) (Figure 2(a)) to produce a new function \( v = Tu \) (Figure 3(a)). Mathematically, the action of this operator may be written as follows: For any \( x \in X \),

\[
v(x) = (Tu)(x) = \sum_{i=1}^{N} \phi_i(u(f_i^{-1}(x))).
\]  

(33)

The prime on the summation signifies that for each \( x \in X \) we sum over only those \( i \in \{1, 2\} \) for which a “preimage” \( f_i^{-1}(x) \) exists. (Because of the “no holes” condition, it guaranteed that for each \( x \in X \), there exists at least one such \( i \) value.) For \( x \in [0, 0.4) \), \( i \) can be only 1. Likewise, for \( x \in (0.6, 1] \), \( i = 2 \). For \( x \in [0.4, 0.6] \), \( i \) can assume both values 1 and 2. The extension to a general \( N \)-map IFSM is straightforward.

There is nothing preventing us from applying the \( T \) operator to the function \( v \), so let \( w = Tv = T(Tu) \). Again, we take the graph of \( v \) and “shrink” it to form two copies, etc. The result is shown in Figure 3(b). As \( T \) is applied repeatedly, we produce a sequence of functions which converges to a function \( \bar{u} \) in an appropriate metric space of functions, which we shall simply denote as \( \mathcal{F}(X) \). In most applications, one employs the function space \( L^2(X) \), the space of real-valued square-integrable functions on \( X \), i.e.,

\[
L^2(X) = \left\{ f : X \rightarrow \mathbb{R}, \| f \|_2 \equiv \left[ \int_X |f(x)|^2 dx \right]^{1/2} < \infty \right\}.
\]  

(34)

In this space, the distance between two functions \( u, v \in L^2(X) \) is given by

\[
d_2(u, v) = \| u - v \|_2 = \left[ \int_X |u(x) - v(x)|^2 dx \right]^{1/2}.
\]  

(35)

The function \( \bar{u} \) is sketched in Figure 3(c). (Because it has so many jumps, it is better viewed as a histogram plot.)

In general, under suitable conditions on the IFS maps \( f_i \) and the grey-level maps \( \phi_i \), the operator \( T \) associated with an IFSM \((w, \Phi)\) is contractive in the space \( \mathcal{F}(X) \). Therefore, from the Banach Contraction Mapping Theorem, it possesses a unique “fixed point” function \( \bar{u} \in \mathcal{F}(X) \). This is precisely the case with the 2-map IFSM given above. Its attractor is sketched in Figure 3(c). Note
Figure 3(a): The resulting “fractal transform” function \( v(x) = (Tu)(x) \) obtained from the component functions of Figure 2(c).

Figure 3(b): The function \( w(x) = T(Tu)(x) = (T^2u)(x) \): the result of two applications of the fractal transform operator \( T \).

Figure 3(c): The “attractor” function \( \bar{u} = T\bar{u} \) of the two-map IFSM given in the text.
that from the fixed point property $\bar{u} = T\bar{u}$ and Eq. (33), the attractor $\bar{u}$ of an $N$-map IFSM satisfies the equation

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

(36)

In other words, the graph of $\bar{u}$ satisfies a kind of “self-tiling” property: it may be written as a sum of distorted copies of itself.

Before going on, let’s consider the three-map IFSM composed of the following IFS maps and associated grey-level maps:

$$f_1(x) = \frac{1}{3}x, \quad \phi_1(y) = \frac{1}{2}y,$$

$$f_2(x) = \frac{1}{3}x + \frac{1}{3}, \quad \phi_2(y) = \frac{1}{2},$$

$$f_3(x) = \frac{1}{3}x + \frac{2}{3}, \quad \phi_3(y) = \frac{1}{2}y + \frac{1}{2},$$

(37)

Notice that $\hat{f}_1(X) = [0, \frac{1}{3}]$ and $\hat{f}_2(X) = [\frac{1}{3}, 1]$ overlap only at one point, $x = \frac{1}{3}$. Likewise, $\hat{f}_2(X)$ and $\hat{f}_3(X)$ overlap only at $x = \frac{2}{3}$. The fixed point attractor function $\bar{u}$ of this IFSM is sketched in Figure 4. It is known as the “Devil’s Staircase” function. You can see that the attractor satisfies a self-tiling property: If you shrink the graph in the $x$-direction onto the interval $[0, \frac{1}{3}]$ and shrink the in $y$-direction by $\frac{1}{3}$, you obtain one piece of it. The second copy, on $[\frac{1}{3}, \frac{2}{3}]$, is obtained by squashing the graph to produce a constant. The third copy, on $[\frac{2}{3}, 1]$, is just a translation of the first copy by $\frac{2}{3}$ in the $x$-direction and $\frac{1}{2}$ in the $y$-direction. (Note: The observant reader can complain that the function graphed in Figure 6 is not the fixed point of the IFSM operator $T$ as defined in Eq. (37): The value $v(\frac{1}{3})$ should be $\frac{3}{2}$ and not $\frac{1}{2}$, since $x = \frac{1}{3}$ is a point of overlap. In fact, this will also happen at $x = \frac{2}{3}$ as well as an infinity of points obtained by the action of the $f_i$ maps on $x = \frac{1}{3}$ and $\frac{2}{3}$. What a mess! Well, not quite, since the function in Figure 7 and the true attractor differ on a countable infinity of points. Therefore, the the $L^2$ distance between them is zero! The two functions belong to the same equivalence class in $L^2([0,1])$.)

Now we have an IFS-method of acting on functions. Along with a set of IFS maps $f_i$ there is a corresponding set of grey-level maps $\phi_i$. Together, Under suitable conditions, the determine a unique attracting fixed point function $\bar{u}$ which can be generated by iterating operator $T$, defined in
Figure 4: The “Devil’s staircase” function, the attractor of the three-map IFSM given in Eq. (37).

Eq. (vTu). As was the case with the “geometrical IFS” earlier, we are naturally led to the following inverse problem for function (or image) approximation:

Given a “target” function (or image) \( v \), can we find an IFSM \((f, \Phi)\) whose attractor \( \bar{u} \) approximates \( v \), i.e.,

\[
u \approx v?
\]

(38)

We can make this a little more mathematically precise:

Given a “target” function (or image) \( v \) and an \( \epsilon > 0 \), can we find an IFSM \((f, \Phi)\) whose attractor \( \bar{u} \) approximates \( v \) to within \( \epsilon \), i.e. satisfies the inequality \( \| v - \bar{u} \| < \epsilon \)?

Here, \( \| \cdot \| \) denotes an appropriate norm for the space of image functions considered.

For the same reason as in the previous lecture, the above inverse problem may be reformulated as follows:

Given a target function \( v \), can we find an IFSM \((f, \Phi)\) with associated operator \( T \), such that

\[
u \approx Tu?
\]

(39)

In other words, we look for a fractal transform \( T \) that maps the target image \( u \) as close as possible to itself. Once again, we can make this a little more mathematically precise:
Given a target function $u$ and an $\delta > 0$, can we find an IFSM $(f, \Phi)$ with associated operator $T$, such that

$$\| u - Tu \| < \delta? \quad (40)$$

This basically asks the question, “How well can we ‘tile’ the graph of $u$ with distorted copies of itself (subject to the operations given above)?” Now, you might comment, it looks like we’re right back where we started. We have to examine a graph for some kind of “self-tiling” symmetries, involving both geometry (the $f_i$) as well as grey-levels (the $\phi_i$), which sounds quite difficult. The response is “Yes, in general it is.” However, it turns out that an enormous simplification is achieved if we give up the idea of trying to find the best IFS maps $f_i$. Instead, we choose to work with a fixed set of IFS maps $f_i, 1 \leq i \leq N$, and then find the “best” grey-level maps $\phi_i$ associated with the $f_i$.

**Question:** What are these “best” grey-level maps?

**Answer:** They are the $\phi_i$ maps which will give the best “collage” or tiling of the function $v$ with contracted copies of itself using the fixed IFS maps, $w_i$.

To illustrate, consider the target function $v = \sqrt{x}$. Suppose that we work with the following two IFS maps on $[0,1]$: $f_1(x) = \frac{1}{2} x$ and $f_2(x) = \frac{1}{2} x + \frac{1}{2}$. Note that $\hat{f}_1(X) = [0, \frac{1}{2}]$ and $\hat{f}_1(X) = [\frac{1}{2}, 1]$. The two sets $\hat{f}(X)$ overlap only at $x = \frac{1}{2}$.

(Note: It is very convenient to work with IFS maps for which the overlapping between subsets $\hat{f}_i(X)$ is minimal, referred to as the “nonoverlapping” case. In fact, this is the usual practice in applications. The remainder of this discussion will be restricted to the nonoverlapping case, so you can forget all of the earlier headaches involving “overlapping” and combining of fractal components.)

We wish to find the best $\phi_i$ maps, i.e. those that make $\| v - Tv \|$ small. Roughly speaking, we would like that

$$v(x) \approx (Tv)(x), \quad x \in [0, 1], \quad (41)$$

or at least for as many $x \in [0, 1]$ as possible. Recall from our earlier discussion that the first step in the action of the $T$ operator is to produce copies of $v$ which are contracted in the $x$-direction onto the subsets $\hat{f}_i(X)$. These copies, $a_i(x) = v(f_i^{-1}(x)), i = 1, 2$, are shown in Figure 5(a) along with
the target \( v(x) \) for reference. The final action is to modify these functions \( a_i(x) \) to produce functions \( b_i(x) \) which are to be as close as possible to the pieces of the original target function \( v \) which sit on the subsets \( f_i(X) \). Recall that this is the role of the grey-level maps \( \phi_i \) since \( b_i(x) = \phi_i(a_i(x)) \) for all \( x \in f_i(X) \). Ideally, we would like grey-level maps that give the result

\[
v(x) \approx b_i(x) = \phi_i(v(f_i^{-1}(x))), \quad x \in f_i(X).
\] (42)

Thus if, for all \( x \in f_i(X) \), we plot \( v(x) \) vs. \( v(f_i^{-1}(x)) \), then we have an idea of what the map \( \phi_i \) should look like. Figure 5(b) shows these plots for the two subsets \( f_i(X) \), \( i = 1, 2 \). In this particular example, the exact form of the grey level maps can be derived: \( \phi_1(t) = \frac{1}{\sqrt{2}} t \) and \( \phi_2(t) = \frac{1}{\sqrt{2}} \sqrt{t^2 + 1} \). I leave this as an exercise for the interested reader.

In general, however, the functional form of the \( \phi_i \) grey level maps will not be known. In fact, such plots will generally produce quite scattered sets of points, often with several \( \phi(t) \) values for a single \( t \) value. The goal is then to find the “best” grey level curves which pass through these data points. But that sounds like least squares, doesn’t it? In most such “fractal transform” applications, only a straight line fit of the form \( \phi_i(t) = \alpha_i t + \beta_i \) is assumed. For the functions in Figure 5(b), the “best” affine grey level maps associated with the two IFS maps given above are:

\[
\begin{align*}
\phi_1(t) &= \frac{1}{\sqrt{2}} t, \\
\phi_2(t) &\approx 0.35216 t + 0.62717.
\end{align*}
\] (43)

The attractor of this 2-map IFSM, shown in Figure 5(c), is a very good approximation to the target function \( v(x) = \sqrt{x} \).

In principle, if more IFS maps \( w_i \) and associated grey level maps \( \phi_i \) are employed, albeit in a careful manner, then a better accuracy should be achieved. The primary goal of IFS-based methods of image compression, however, is not necessarily to provide approximations of arbitrary accuracy, but rather to provide approximations of acceptable accuracy “to the discerning eye” with as few parameters as possible. As well, it is desirable to be able to compute the IFS parameters in a reasonable amount of time.
Figure 5(a): The target function $v(x) = \sqrt{x}$ on $[0,1]$ along with its contractions $a_i(x) = v(w_i^{-1}(x))$, $i = 1, 2$, where the two IFS maps are $w_1(x) = \frac{1}{2}x$, $w_2(x) = \frac{1}{2}x + \frac{1}{2}$.

Figure 5(b): Plots of $v(x)$ vs $a_i(x) = v(w_i^{-1}(x))$ for $x \in w_i(X)$, $i = 1, 2$. These graphs reveal the grey level maps $\phi_i$ associated with the two-map IFSM.

Figure 5(c): The attractor of the two-map IFSM with grey level maps given in Eq. (43).
“Local IFSM”

That all being said, there is still a problem with the IFS method outlined above. It works fine for the examples that were presented but these are rather special cases – all of the examples involved monotonic functions. In such cases, it is reasonable to expect that the function can be approximated well by combinations of spatially-contracted and range-modified copies of itself. In general, however, this is not guaranteed to work. A simple example is the target function \( u(x) = \sin \pi x \) on \([0,1]\), the graph of which is sketched in Figure 6 below.

![Figure 6: Target function \( u(x) = \sin \pi x \) on \([0,1]\)](image)

Suppose that we try to approximate \( u(x) = \sin \pi x \) with an IFS composed with the two maps,

\[
\begin{align*}
f_1(x) &= \frac{1}{2}x \\
f_2(x) &= \frac{1}{2} + \frac{1}{2}.
\end{align*}
\]

(44)

It certainly does not look as if one could express \( u(x) = \sin \pi x \) with two contracted copies of itself which lie on the intervals \([0, 1/2]\) and \([1/2, 1]\). Nevertheless, if we try it anyway, we obtain the result shown in Figure 7. The best “tiling” of \( u(x) \) with two copies of itself is the constant function, \( \bar{u}(x) = \frac{2}{\pi} \), which is the mean value of \( u(x) \) over \([0,1]\).

If we stubbornly push ahead and try to express \( u(x) = \sin \pi x \) with four copies of itself, i.e., use the four IFS maps,

\[
\begin{align*}
f_1(x) &= \frac{1}{4}x \\
f_2(x) &= \frac{1}{4}x + \frac{1}{4} \\
f_3(x) &= \frac{1}{4}x + \frac{1}{2} \\
f_4(x) &= \frac{1}{4}x + \frac{3}{4},
\end{align*}
\]

(45)

then the attractor of the “best four-map IFS” is shown in Figure 8. It appears to be a piecewise constant function as well.

Of course, we can increase the number of IFS maps to produce better and better piecewise constant approximations to the target function \( u(x) \). But we really don’t need IFS to do this. A better strategy,
which follows a method A significant improvement, which follows a method introduced in 1989 by A. Jacquin, then a Ph.D. student of Prof. Barnsley, is to break up the function into “pieces”, i.e., consider it as a collection of functions defined over subintervals of the interval $X$. Instead of trying to express a function as a union of copies of spatially-contracted and range-modified copies of itself, the modified method, known as “local IFS,” tries to express each “piece” of a function as a spatially-contracted and range-modified copie of larger “pieces” of the function, not the entire function. We illustrate by considering once again the target function $u(x) = \sin \pi x$. It can be viewed as a union of two monotonic functions which are defined over the intervals $[0,1/2]$ and $[1/2,1]$. But neither of these “pieces” can, in any way, be considered as spatially-contracted copies of other monotone functions extracted from $u(x)$. As such, we consider $u(x)$ as the union of four “pieces,” which are supported on the so-called “range” intervals,

$$I_1 = [0, 1/4], \quad I_2 = [1/4, 1/2], \quad I_3 = [1/2, 3/4], \quad I_4 = [3/4, 1].$$

We now try to express each of these pieces as spatially-contracted and range-modified copies of the
two larger “pieces” of \( u(x) \) which are supported on the so-called “domain” intervals,

\[
J_1 = [0, 1/2] \quad J_2 = [1/2, 1].
\] (47)

In principle, we can find IFS-type contraction maps which map each of the \( J_k \) intervals to the \( I_l \) intervals. But we can skip these details. We’ll just present the final result. Figure 9 shows the attractor of the IFS that produces the best “collage” of \( u(x) = \sin \pi x \) using this 4 domain block/2 range block method. It clearly provides a much better approximation than the earlier four-IFS-map method.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig9}
\caption{IFSM attractor obtained by trying to approximate \( u(x) = \sin \pi x \) on [0,1] with four copies of itself.}
\end{figure}

**Fractal image coding**

We now outline a simple block-based fractal coding scheme for a greyscale image function, for example, 512 × 512 pixel Boat image shown back in Figure 1(a).

In what follows, let \( X \) be an \( n_1 \times n_2 \) pixel array on which the image \( u \) is defined.

- Let \( R^{(n)} \) denote a set of \( n \times n \)-pixel range subblocks \( R_i \), \( 1 \leq i \leq N_{R^{(n)}} \), which cover \( X \), i.e., \( X = \bigcup_i R_i \).

- Let \( D^{(m)} \) denote a set of \( m \times m \)-pixel domain \( D_j \), \( 1 \leq j \leq N_{D^{(m)}} \), where \( m = 2n \). (The \( D_i \) are not necessarily non-overlapping, but they should cover \( X \).) These two partitions of the image are illustrated in Figure 10.

- Let \( w_{ij} : D_j \to R_i \) denote the affine geometric transformations that map domain blocks \( D_j \) to \( R_i \). There are 8 such constraction maps: 4 rotations, 2 diagonal flips, vertical and horizontal
flips, so the maps should really be indexed as \( w_{ij}^k \), \( 1 \leq k \leq 8 \). In many cases, only the zero rotation map is employed so we can ignore the \( k \) index, which we shall do from here on for simplicity.

Since we are now working in the discrete domain, i.e., pixels, as opposed to continuous spatial variables \((x, y)\), some kind of “decimation” is required in order to map the larger \( 2n \times 2n \)-pixel domain blocks to the smaller \( n \times n \)-pixel range blocks. This is usually accomplished by a “decimation procedure” in which nonoverlapping \( 2 \times 2 \) square pixel blocks of a domain block \( D_j \) are replaced with one pixel. This definition of the \( w_{ij} \) maps is a formal one in order to identify the spatial contractions that are involved in the fractal coding operation.

The decimation of the domain block \( D_j \) is accompanied by a decimation of the image block \( u(D_j) \) which is supported on it, i.e., the \( 2n \times 2n \) greyscale values that are associated with the pixels in \( D_j \). This is usually done as follows: The greyscale value assigned to the pixel replacing four pixels in a \( 2 \times 2 \) square is the average of the four greyscale values over the square that has been decimated. The result is an \( n \times n \)-pixel image, to be denoted as \( \tilde{u}(D_j) \), which is the “decimated” version of \( u(D_j) \).

- For each range block \( R_i \), \( 1 \leq i \leq N_{R(n)} \), compute the errors associated with the approximations,

\[
u(R_i) \approx \phi_{ij}(u(w_{ij}^{-1}(R_i))) = \phi_{ij} \tilde{u}(D_j), \quad \text{for all } 1 \leq j \leq N_{D(n)}, \tag{48}\]

where, for simplicity, we use affine greyscale transformations,

\[
\phi(t) = \alpha t + \beta. \tag{49}\]

The approximation is illustrated in Figure 11.

In each such case, one is essentially determining the best straight line fit through \( n^2 \) data points \( (x_k, y_k) \in \mathbb{R}^2 \), where the \( x_k \) are the greyscale values in image block \( \tilde{u}(D_j) \) and the \( y_k \) are the

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{image.png}
\caption{Partitioning of an image into range and domain blocks.}
\end{figure}
\[ z = u|D_j(x, y) \]

\[ z' = u|R_i(x, y) \]

**Figure 11.** Left: Range block \( R_i \) and associated domain block \( D_j \). Right: Greyscale mapping \( \phi \) from \( u(D_j) \) to \( u(R_i) \).

The corresponding greyscale values in image block \( u(R_i) \). (Remember that you may have to take account of rotations or inversions involved in the mapping \( w_{ij} \) of \( D_j \) to \( R_j \)). This can be done by the method of least squares, i.e., finding \( \alpha \) and \( \beta \) which minimize the total squared error,

\[ \Delta^2(\alpha, \beta) = \sum_{k=1}^{n} (y_i - \alpha x_i + \beta)^2 . \]  

(50)

As is well known, minimization of \( \Delta^2 \) yields a system of linear equations in the unknowns \( \alpha \) and \( \beta \).

Now let \( \Delta_{ij}, 1 \leq j \leq \mathcal{D}^{(m)} \) denote the approximation error \( \Delta \) associated with the approximations to \( u(R_i) \) in Eq. (48). Choose the domain block \( j(i) \) that yields the lowest approximation error.

The result of the above procedure: You have fractally encoded the image \( u \). The following set of parameters for all range blocks \( R_i, 1 \leq i \leq N_{R(n)} \),

\[ j(i), \quad \text{index of best domain block}, \]

\[ \alpha_i, \beta_i, \quad \text{affine greyscale map parameters}, \]  

(51)

comprises the **fractal code** of the image function \( u \). The fractal code defines a fractal transform \( T \).

The fixed point \( \bar{u} \) of \( T \) is an approximation the image \( u \), i.e.,

\[ u \approx \bar{u} = T\bar{u} . \]  

(52)

This is happening for the same reason as for our IFSM function approximation methods outlined in the previous section. Minimization of the approximation errors in Eq. (48) is actually minimizing the “tiling error”

\[ \| u - T\bar{u} \|, \]  

(53)

437
originally presented in Eq. (40). We have found a fractal transform operator that maps the image $u$—in “pieces,” i.e., in blocks—close to itself.

**Moral of the story:** You store the fractal code of $u$ and generate its approximation $\bar{u}$ by iterating $T$, as shown in the next example.

In Figure 12, are shown the results of the above block-based IFSM procedure as applied to the 512 $\times$ 512 Boat image. 8 $\times$ 8-pixel blocks were used for the range blocks $R_i$ and 16 $\times$ 16-pixel blocks for the domain blocks $D_j$. As such, there are 4096 range blocks and 1024 domain blocks.

The bottom left image of Figure 12 is the fixed point attractor $\bar{u}$ of the fractal transform defined by the fractal code obtained in this procedure.

You may still be asking the question, “How to we iterate the fractal transform $T$ to obtain its fixed point attractor?” Very briefly, we start with a “seed image,” $u_0$, which could be the zero image, i.e., an image for which the greyscale value at all pixels is zero. You then apply the fractal operator $T$ to $u_0$ to obtain a new image $u_1$, and then continue with the iteration procedure,

$$u_{n+1} = Tu_n, \ n \geq 0.$$  \hfill (54)

After a sufficient number of iterations (around 10-15) for 8 $\times$ 8 range blocks, the above iteration procedure will have converged.

But perhaps we haven’t answered the question completely. At each stage of the iteration procedure, i.e, at step $n$, when you wish to obtain $u_{n+1}$ from $u_n$, you must work with each of its range blocks $R_i$ separately. One replaces image block $u_n(R_i)$ supported on $R_i$ with a suitably modified version of the image $u_n(D_{j(i)})$ on the domain block $D_{j(i)}$ as dictated by the fractal code. The image block $u_n(D_{j(i)})$ will first have to be decimated. (This can be done at the start of each iteration step, so you don’t have to be decimating each time.) It is also important to make a copy of $u_n$ so you don’t modify the original while you are constructing $u_{n+1}$. Remember that the fractal code is determined by approximating $u_n$ with parts of itself!

There are still a number of other questions and points that could be discussed. For example, better approximations to an image can be obtained by using smaller range blocks, $R_i$, say 4 $\times$ 4-pixel
Figure 12. Clockwise, starting from top left: Original Boat image. The iterates $u_1$ and $u_2$ and fixed point approximation $\bar{u}$ obtained by iteration of fractal transform operator. ($u_0 = 0.$) $8 \times 8$-pixel range blocks. $16 \times 16$-pixel domain blocks. But that means small domain blocks $D_j$, i.e., $8 \times 8$ blocks, which means greater searching to find an optimal domain block for each range block. The searching of the “domain pool” for optimal blocks is already a disadvantage of the fractal coding method.

That being said, various methods have been investigated and developed to speed up the coding time by reducing the size of the “domain pool.” This will generally produce less-than-optimal ap-
proximations but in many cases, the loss in fidelity is almost non-noticeable.

Some references (these are old!)

Original research papers:


Books:


Expository papers:
