Image Processing with Wavelets and Multifractal Analysis

Franklin Mendivil
Acadia University

franklin.mendivil@acadiau.ca

Overview

- Wavelet based methods
  - Wavelets for image representation and JPEG 2000
  - Non-separable wavelets
  - Wavelet Denoising
- IFS based methods
  - IFS fractal basics – IFS on images
  - IFS image compression – basics and variations
  - Fractal denoising
  - Mixing IFS and wavelets
  - IFS in Content Based Image Retrieval
- Multifractal based methods
  - Basic introduction
  - Edge detection
  - Denoising

Wavelet Based Methods

Wavelet image representations

The ability of wavelets to localize position and frequency is very useful in representing images.

This allows us to isolate a region of the image or to examine the image on different scales.

Decomposing an image in a wavelet basis tends to concentrate most of the energy of the image into a few (large) coefficients. This is referred to as *energy compaction*.

One reason for this is that images tend to have large smooth regions. In these regions the wavelet coefficients decay rapidly.

This also means that the places where the wavelet coefficients fail to decay rapidly (usually) correspond to edges or texture.

JPEG2000

The new JPEG2000 standard uses wavelet coding for image compression (both lossless and lossy).

In the lossless mode, an integer-to-integer wavelet transform is used. This mode can also be lossy if we cut the progressive bitstream at some point.

The coefficients are quantized and then each bitplane is coded independently.

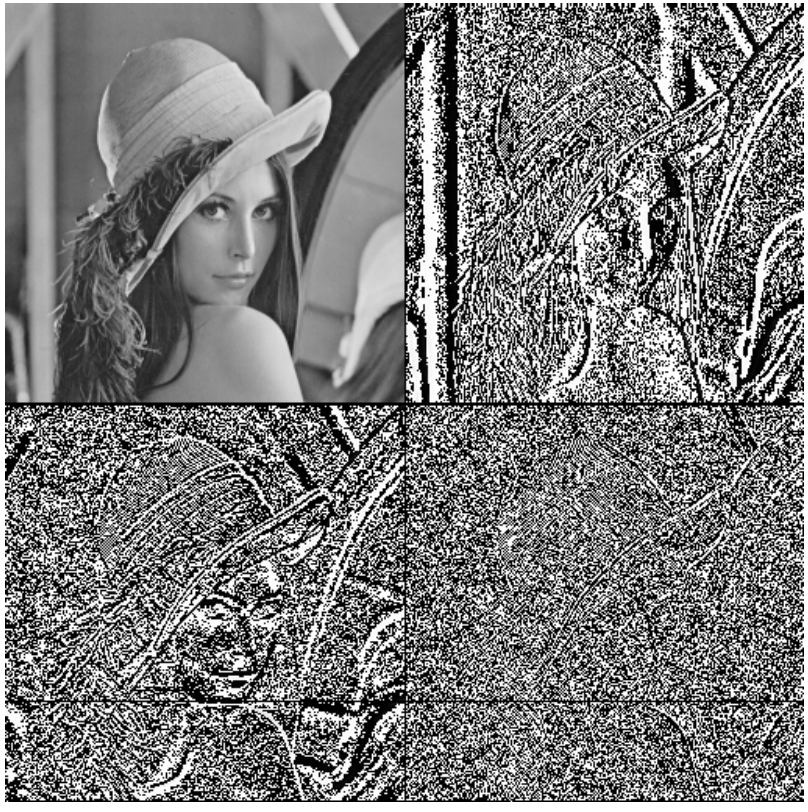
When a particular bit (in a bitplane) is encoded depends on its “significance”. Thus, even within a bitplane the stream can be progressive.

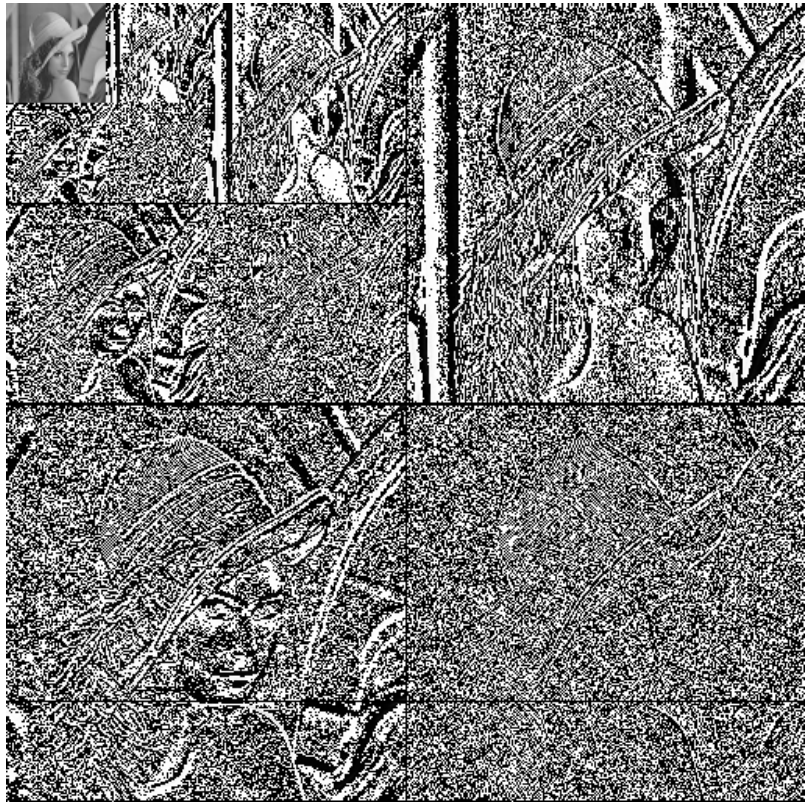
A context-dependent binary arithmetic coder is used for entropy coding.

The progressive local improvement data is collected in “packets” which form “layers” (a global quality improvement).

The scale and resolution capability of wavelets is essential for this progressive encoding.

Separable decomposition





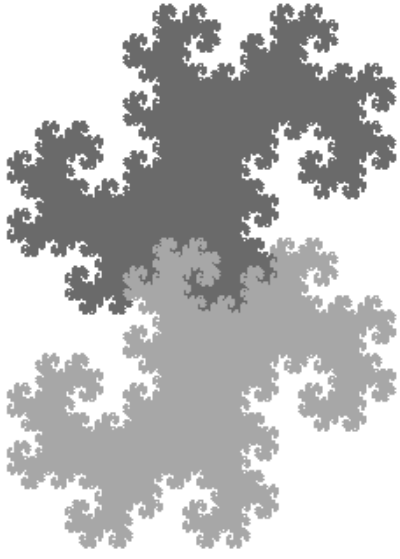
Non-separable wavelets

The scaling function for a non-separable wavelet satisfies a matrix dilation equation of the form

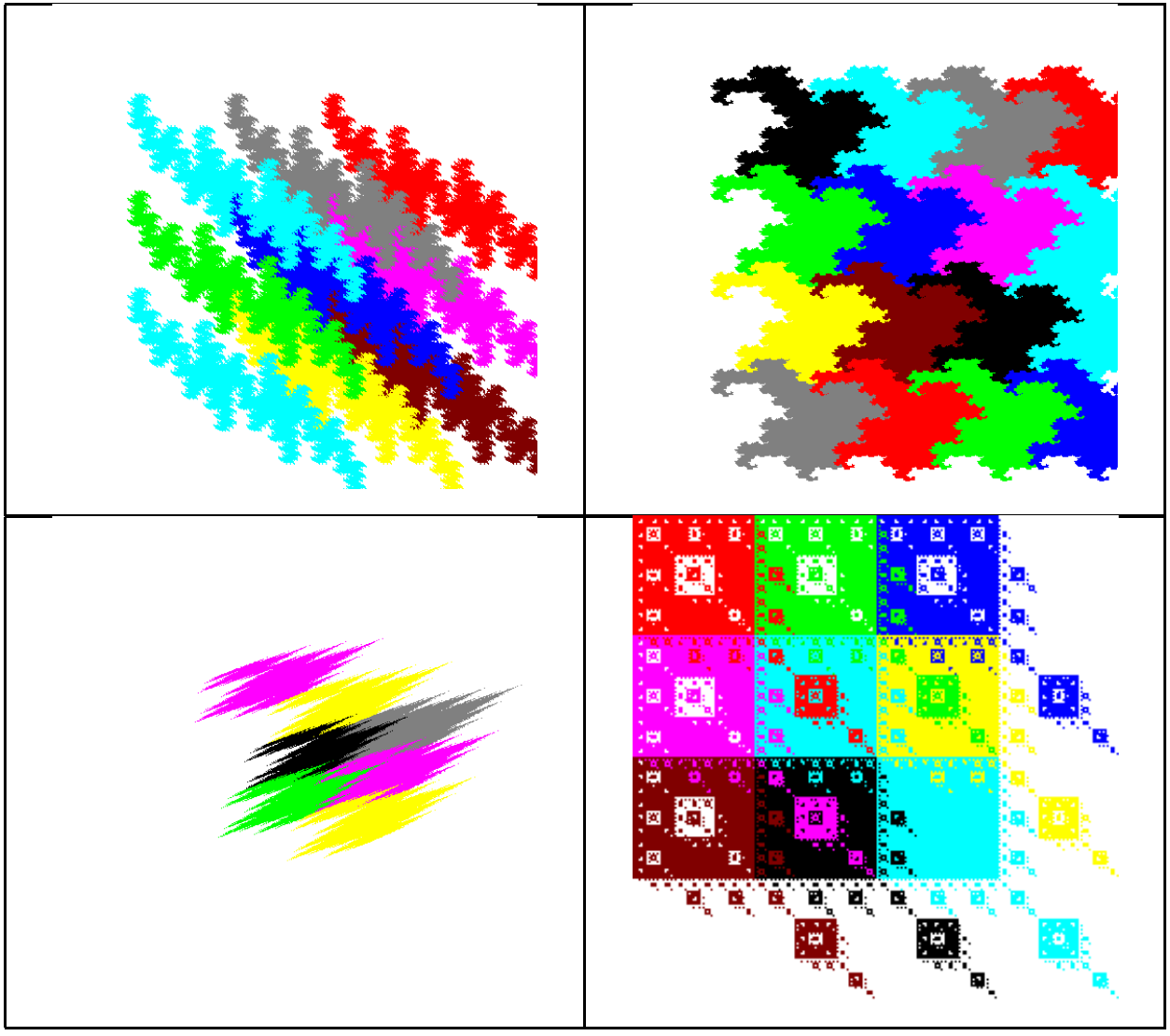
$$\phi(x) = \sum_{i \in \mathbb{Z}^n} c_i \phi(Ax - i)$$

where A is a *dilation matrix* (expansive integer matrix).

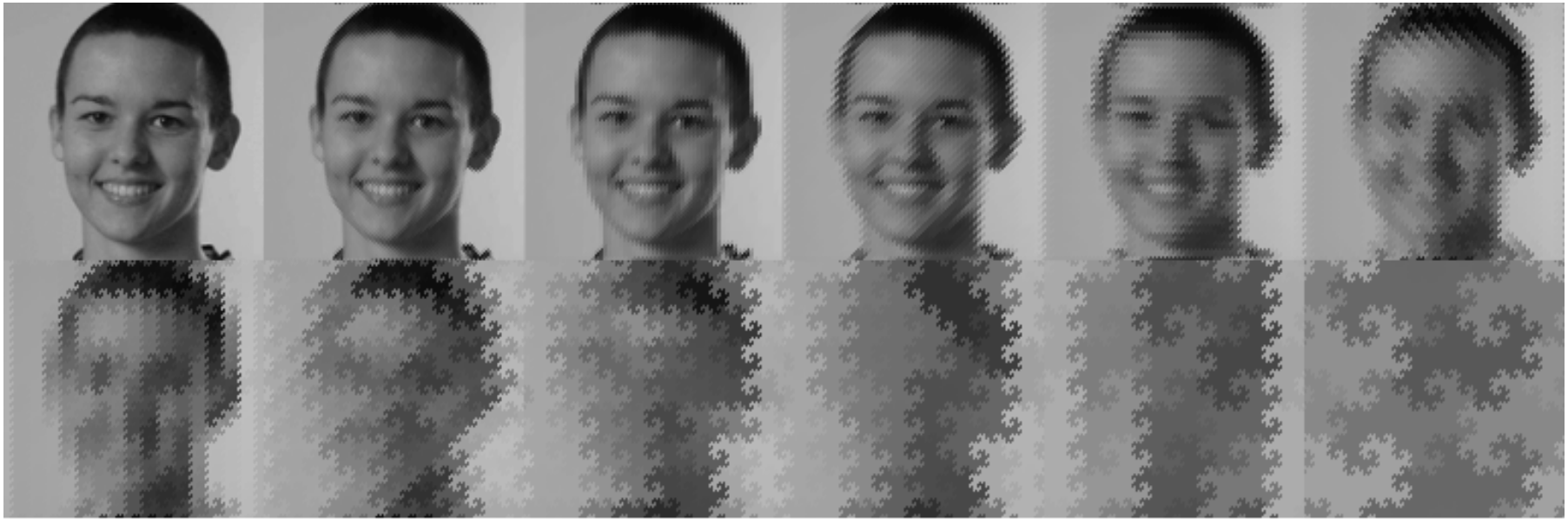
The simplest examples of non-separable wavelets are Haar wavelets based on self-affine tilings of $\mathbb{R}^2$.



Other self-affine tilings



Non-separable subsampling



Smooth non-separable wavelets

The construction of continuous or smooth non-separable wavelets is considerably more difficult than Haar wavelets.

There are several obstructions to using the one-dimensional techniques to construct such wavelets.

At the present time, not many families of smooth non-separable wavelets are known. It is known that arbitrarily smooth non-separable compactly supported wavelets exist.

Truncation: separable vs. non-separable

Separable truncations 3,4,5,6 levels

Non-separable truncations 6,8,10,12 levels

Truncation: separable vs. non-separable

Separable truncations 2,3,4,5 levels

Non-separable truncations 4,6,8,10 levels

Wavelet Denoising

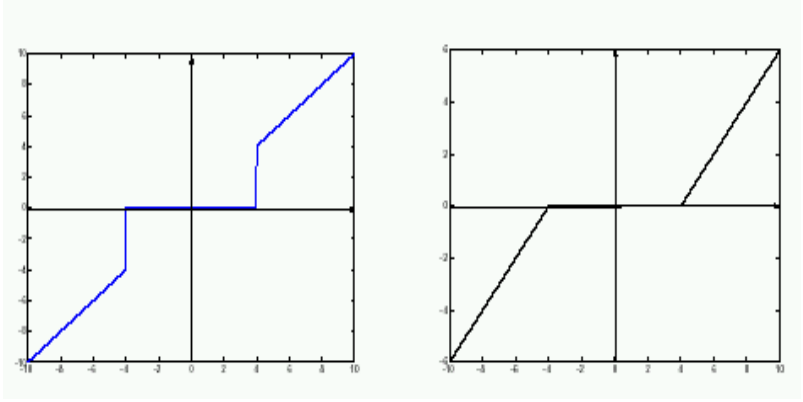
Suppose we have an image X that is corrupted with noise N so we observe $Y = X + N$. We want to find an estimate $\hat{X}$ for X.

The energy compaction property of wavelets means that Y will have a few large coefficients and many small coefficients.

The basic idea in wavelet denoising is to find these small coefficients and to shrink them (either zero them or shrink them towards zero), thus reducing the noise.

Hard vs. Soft Thresholding

There are several different methods for setting the threshold. One basic distinction is *hard* versus *soft* thresholding.



Soft thresholding has the advantage of being continuous (so as not to introduce artificial discontinuities).

Soft thresholding has also been shown to achieve a near-optimal minimax rate over several Besov spaces.

Soft thresholding also has some other theoretical advantages.

Denoising (cont)

BayesShrink: the threshold is σ^2/σ_X. This threshold is derived by assuming that the distribution of the wavelet coefficients in any scale is given by a Generalized Gaussian Distribution (GGD). The problem is to estimate the GGD parameters for each subband. These parameters are estimated for each subband from the noisy image data.

SureShrink: the threshold is computed as either a SURE threshold (derived from minimizing Stein's unbiased risk estimate) or the Universal threshold ($\sigma\sqrt{2\log(M)}$, which is asymptotically optimal in the minimax sense).

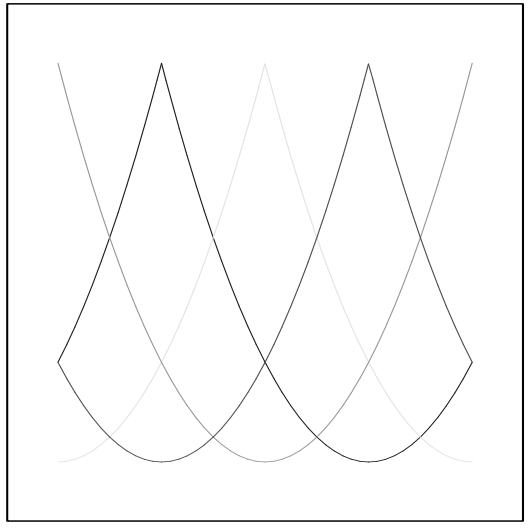
The SURE threshold can be viewed as another (and data-driven) way to minimize the Bayes risk given a GGD prior on the coefficients.

NormalShrink: the threshold is $\beta\sigma^2/\sigma_Y$, where $\beta = \sqrt{\log(L_k/J)}$ (L_k is the length of the kth subband and J is the number of subbands), σ is the estimated noise variance (as before) and σ_Y is the variance of the particular subband of the noisy signal.

Cycle Spinning

The discrete wavelet transform has a lattice of preferred locations. These locations might not be the correct ones for denoising a particular image.

One way to deal with this problem is via *cycle spinning* – denoising shifted versions of the image and then averaging the various shifts.

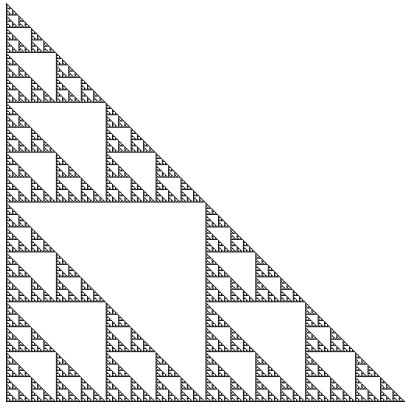


The idea is that the noise at different locations is independent, so the averaging will tend to smooth out the noise.

IFS Based Methods

IFS Fractal Basics

We wish to formalize the notion of a set being *self-similar*.



Consider the three affine maps

$$w_0(x,y) = (\frac{x}{2}, \frac{y}{2}), \quad w_1(x,y) = (\frac{x+1}{2}, \frac{y}{2}), \quad w_2(x,y) = (\frac{x}{2}, \frac{y+1}{2}).$$

We see that if S is the Sierpinski gasket (above) then

$$S = w_0(S) \cup w_1(S) \cup w_2(S).$$

In fact, S is the unique such set.

Definition of IFS

Let (X, d) be a complete metric space.

An *IFS on* X is a finite set of contractive self-maps $\{w_i : X \to X : i = 0, \ldots N\}$.

The *attractor* of the IFS is a non-empty compact set $A \subset X$ so that $A = \bigcup_i w_i(A)$.

We endow the space

$$\mathcal{H}(X) = \{K \subset X : \emptyset \neq K \text{ is compact}\}$$

with the Hausdorff metric d_h making $(\mathcal{H}(X), d_h)$ a complete metric space.

For our purposes, we usually take $X = [0, 1]^2$.

The map $W : \mathcal{H}(X) \to \mathcal{H}(X)$ defined by

$$W(B) = \bigcup_i w_i(B)$$

is a contraction if $\{w_i\}$ is an IFS.

Thus

$$B_n := W^n(B) \Rightarrow A$$

for any non-empty compact set B.

Collage Theorem

Theorem: (Collage Theorem) Let $T : X \to X$ be a contractive map with contractivity factor $s < 1$ and x_T be its fixed point. Then

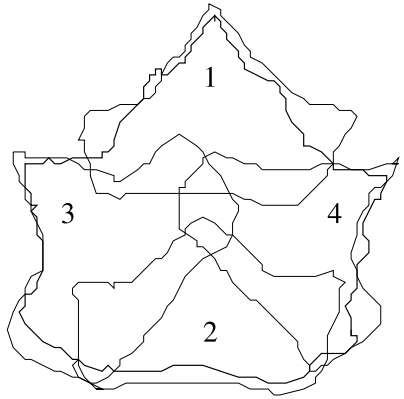
$$d(y, x_T) \leq \frac{d(y, T(y))}{1 - s}.$$

In the context of an IFS $\{w_i\}$, this says that if a set B almost self-tiles under $\{w_i\}$ then B is "close" to the attractor of the IFS.

This is used in the inverse problem: *Given a set A, find an IFS whose fixed point is A.*

Often the quantity $d(y, T(y))$ is referred to as the *collage distance.*

Collage Theorem (cont)



A collage of a leaf with copies of itself.

The attractor of the resulting IFS.

IFS on Function Spaces (IFSM)

We consider an image as a function $f : X \to \mathbb{R}$, so we need an IFS operator on functions.

Let $\phi_i : \mathbb{R} \to \mathbb{R}$ be maps. We usually want them to be Lipschitz.

Define $T(f)(x) = \sum_i \phi_i \left(f(w_i^{-1}(x)) \right)$ so T maps functions on X to functions on X.

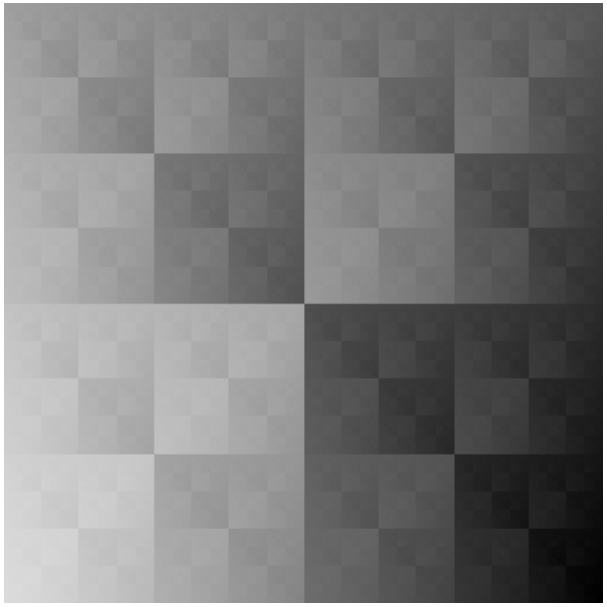
Under suitable conditions on the w_i's (contractivity) and on the ϕ_i's we know that T will map $L^p(X)$ to $L^p(X)$.

Notice that the w_i's perform a mapping on the "base space" X and the ϕ_i's map on the "vertical" space $\mathbb{R}$. The ϕ_i's are usually called the *grey-scale maps*.

Most often we take

$$w_i(x) = a_i x + s_i \quad \text{and} \quad \phi_i(t) = \alpha_i t + \beta_i.$$

IFSM example



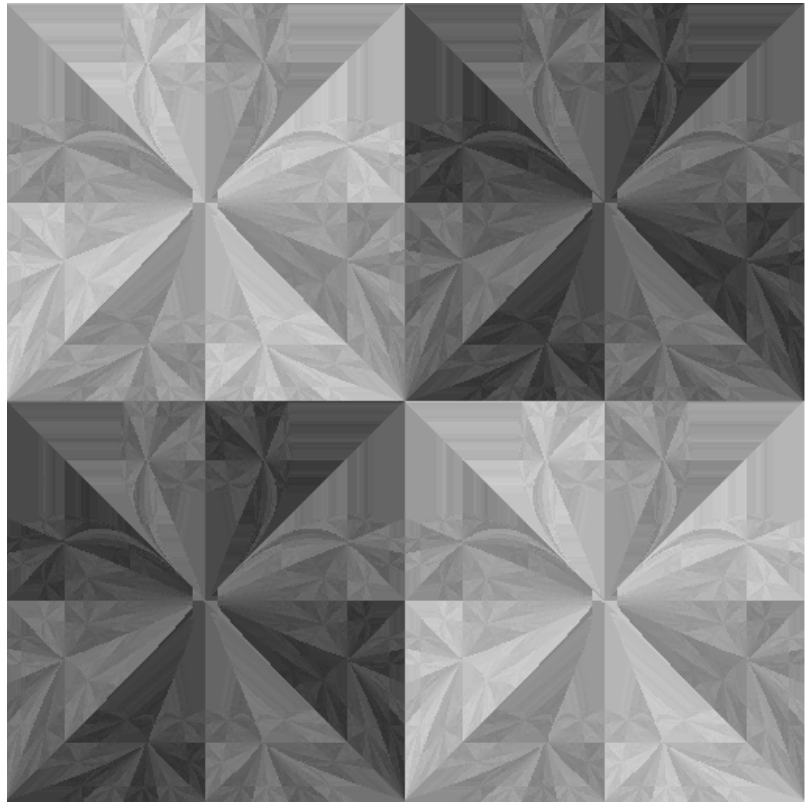
There is one parent and four children.

The parameters are

α	0.5	0.5	0.4	0.5
β	80	40	132	0

Another example

This is the same type of IFSM operation, except with a twist.



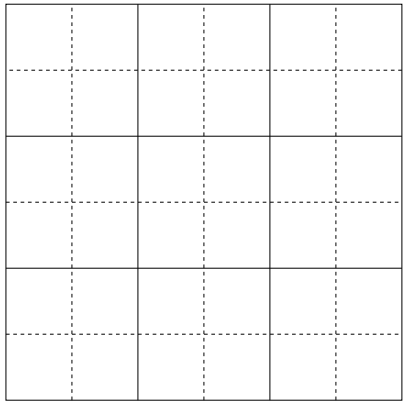
Here we have allowed non-linear maps w_i.

Fractal Image Compression

Given a picture (a function f), the basic idea is to try to find an IFSM operator T whose fixed point x_T closely matches f.

If we can find a T so that $d(T(f), f)$ is small, then we know (by the *Collage Theorem*) that $d(f, x_T)$ will be small as well.

The standard fractal compression method is the Block-Encoding method.



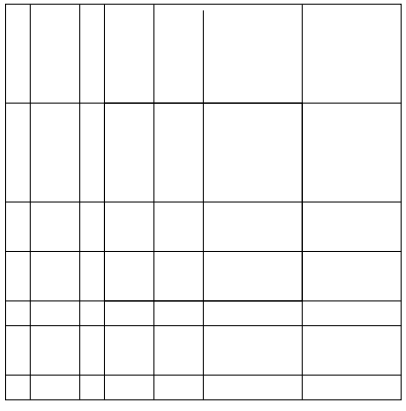
The maps w_i are found by the searching while the α_i and β_i are (usually) calculated to minimize mean square error.

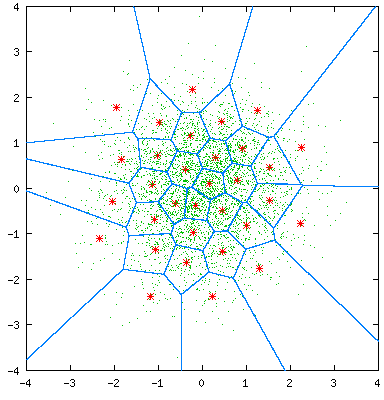
Fractal Compression as Self-VQ

Vector Quantization (VQ) is one form of data compression for multi-dimensional data.

In basic VQ you map each data vector into an appropriate (usually closest) *codevector* from a *codebook*.

This allows you to quantize vector data which might have dependencies between the components.





IFS block-coding is like a Self-VQ in that the image acts as its own codebook.

Why are we interested in compression?

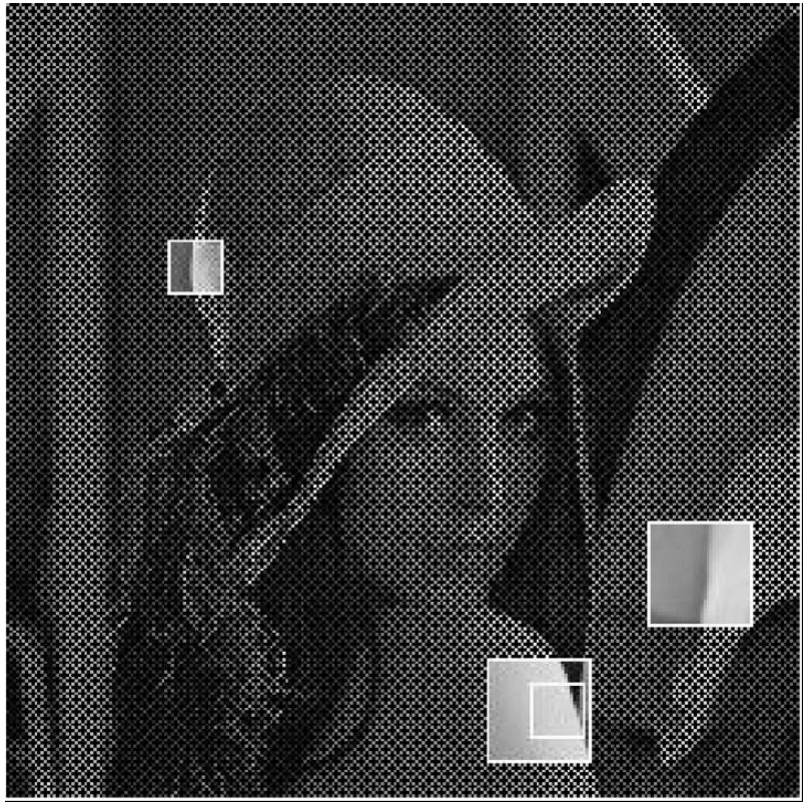
Compression is approximation – we attempt to approximate the image by the fixed point of an IFSM.

In doing this, we analyze the “fractal” nature of the image.

The starting point of the IFS-based methods for image processing is this fractal analysis.

Local IFS

This type of IFS can be formalized as a *Local* IFS.



Basic Fractal Compression: example

Reconstruction: one, two, three, four, ten iterations and the original image.

Extensions to the basic algorithm

Many modifications and extensions to the basic algorithm have been proposed.

These include adaptive partitions (both top down and bottom up), more complicated grey-level maps (the ϕ_i's) and merging fractal and wavelet or other transform techniques.

A significant problem is the problem of searching for the best match. This has also been a large focus of the research in this area.

Methods to speed up the search include classifying the blocks, limiting to a local search, using transforms or convolution, and doing the search in a hierarchical fashion (multi-scale).

Postprocessing

One problem with the iteration scheme is that edges (and noise) can get replicated to finer scales.

One can either apply a smoothing filter to the end result or (better) apply some smoothing operation for each iteration.

Varying the partition

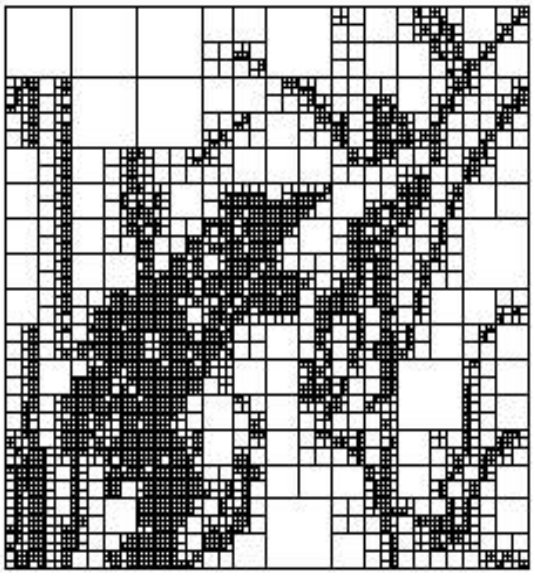
Many different types of partitions have been tried, including triangular partitions, rectangular partitions and more general partitions.

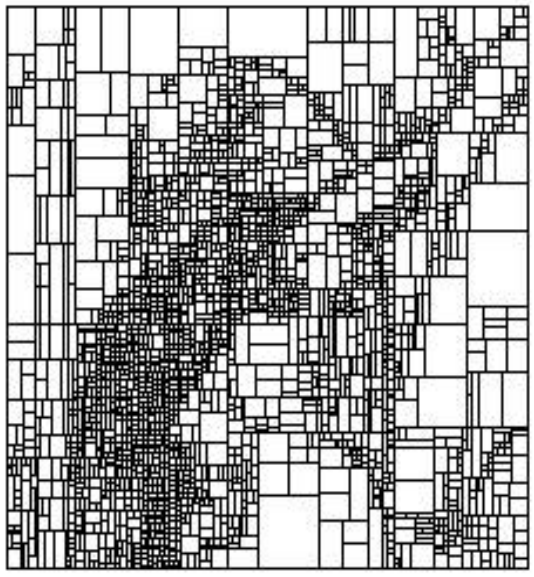
Most of these partitioning schemes are also adaptive.

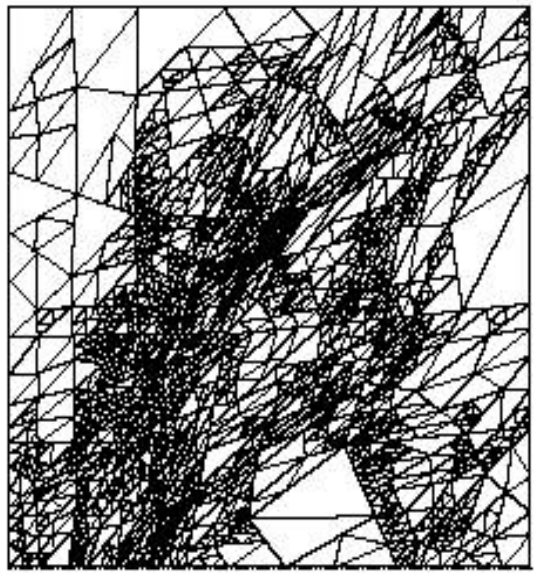
The benefit is the flexibility to match image features.

The cost is both the extra algorithmic complexity and the extra storage necessary for encoding the partition.

Examples of three partitions





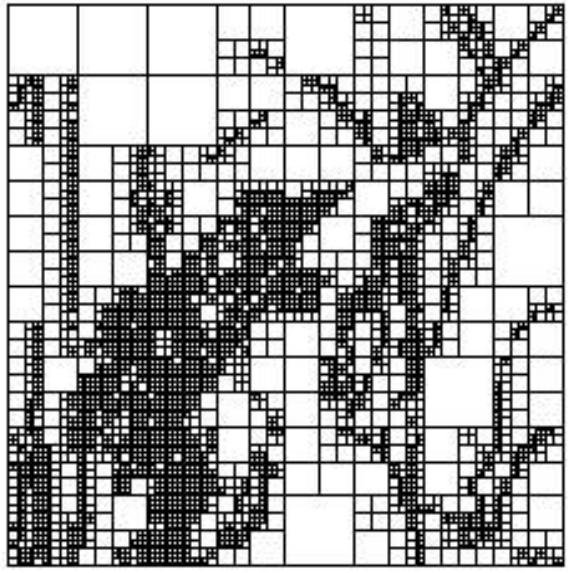


Quadtree partition

This is the simplest adaptive partitioning scheme.

Start with large(ish) child blocks and try to find a match, subdividing into four smaller blocks if some error tolerance is not met.

The tree structure allows simpler storage of the partition.

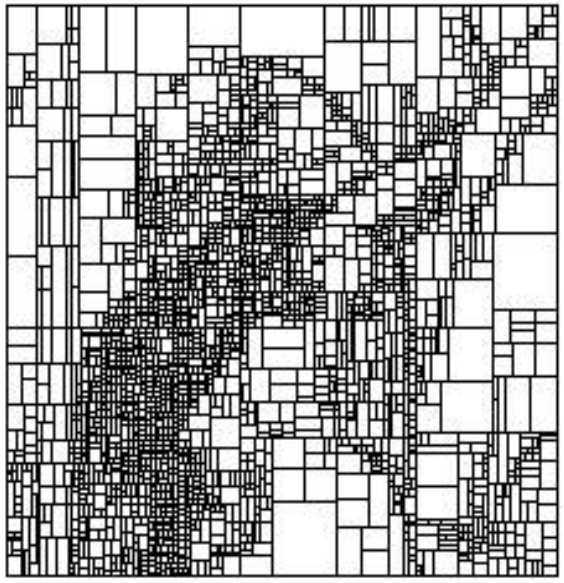


HV partitions (Horizontal, Vertical)

There are several split criteria.

We could try to find an edge and split on the edge.

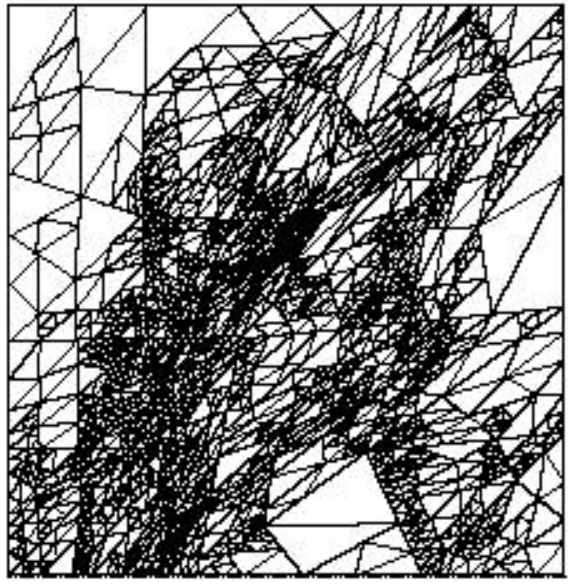
We could split to minimize the error (often mean-square).



Triangular partition

Triangular partitions are very flexible but complex.

The partition requires a lot of information to encode.



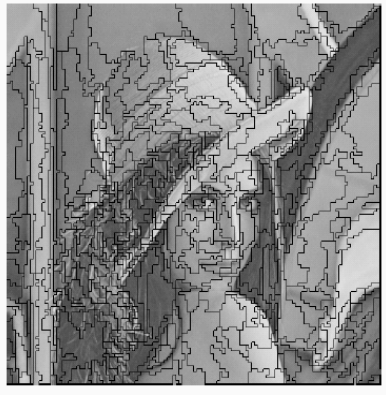
Region growing

In this approach, we start with a partition into small atomic regions, usually blocks. These blocks function as the child regions.

We selectively merge two regions according to an error criteria.

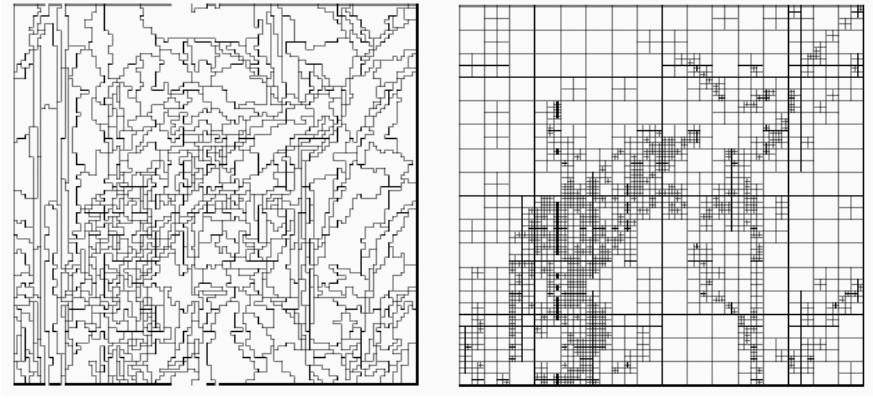
We must also merge the corresponding parent regions.

Thus, we keep a list of active child and parent regions and possible candidates for merging.



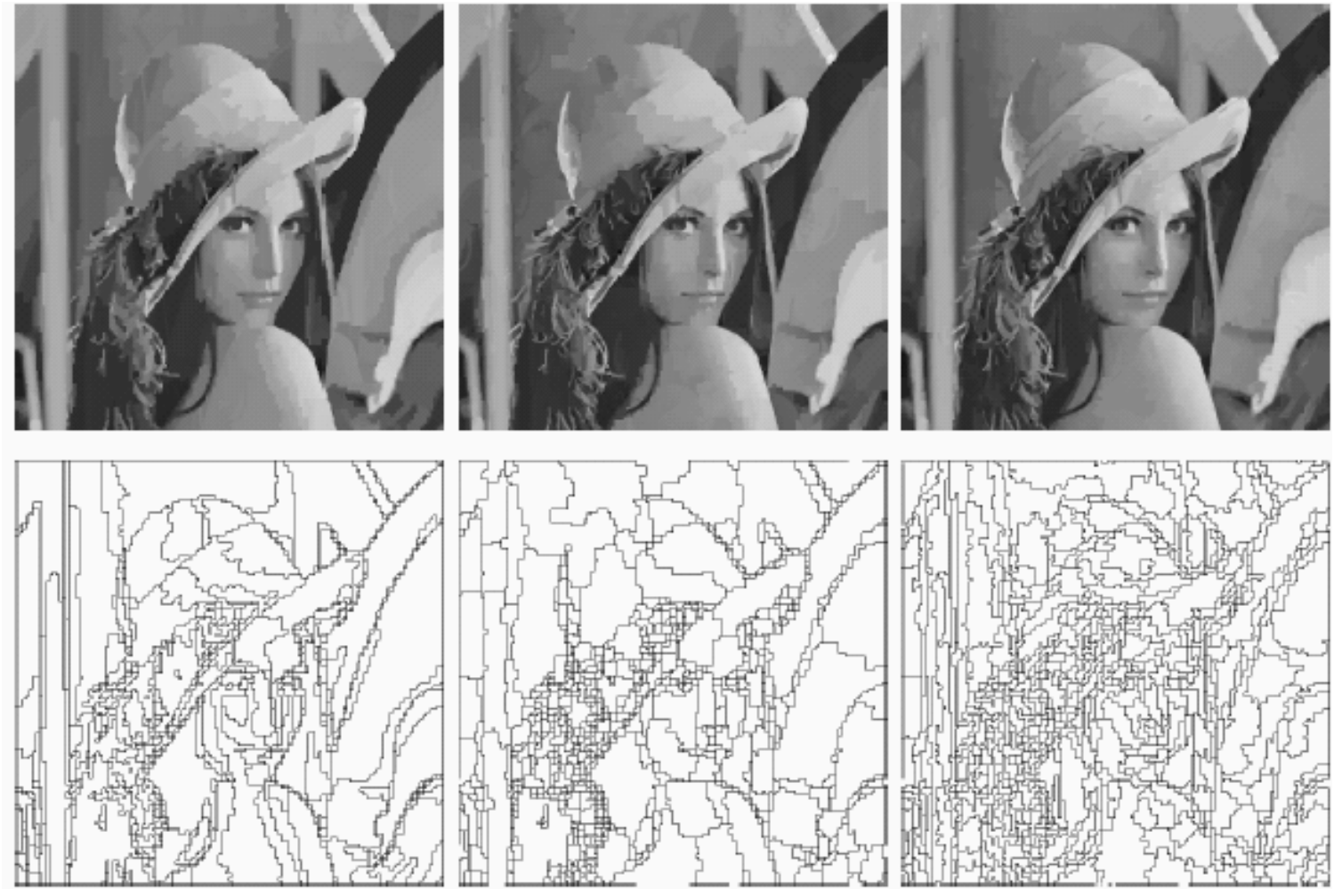
Region growing example

Region growing and quadtree fractal compression



Region growing partition and quadtree partition

Region growing example



Variance based splitting criteria

Two collage based splitting criteria

Beyond collage coding

The IFS image approximation usually uses the collage distance as a measure of error.

$$d(y, x_T) \leq \frac{d(y, T(y))}{1 - s}$$

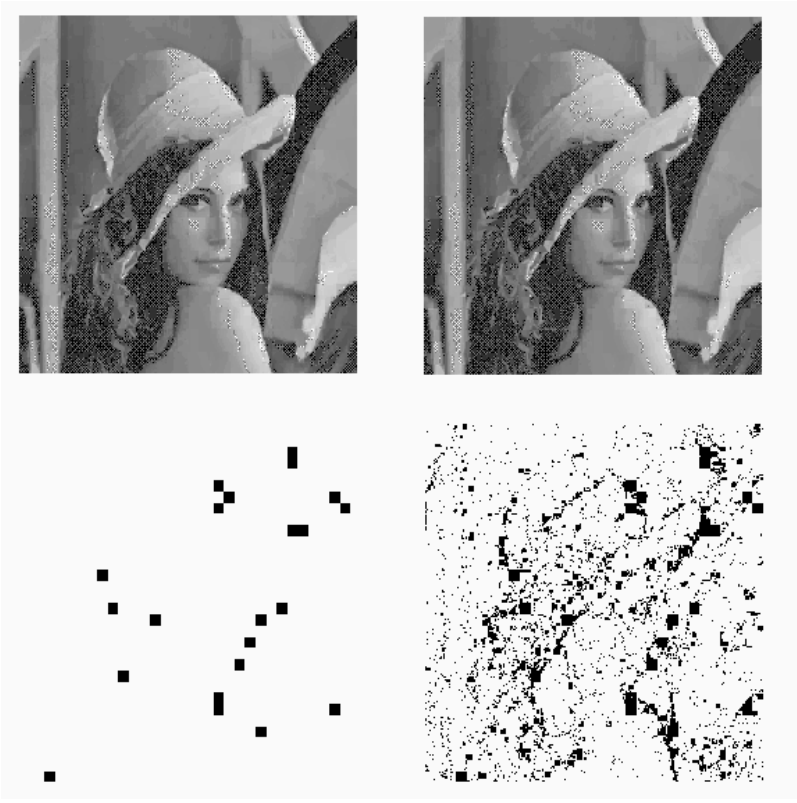
This provides a sub-optimal solution, since we really want to minimize $d(y, x_T)$.

This is done to simplify the computational problem.

There are various ways of trying to improve on this.

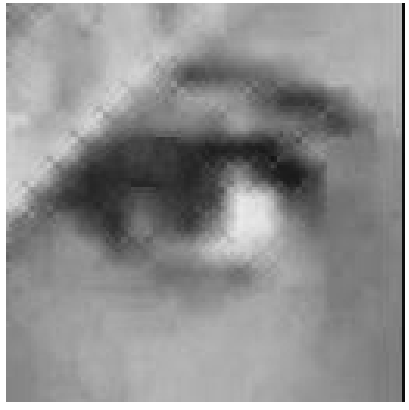
These usually involve starting with the collage solution and doing some type of “local” search.

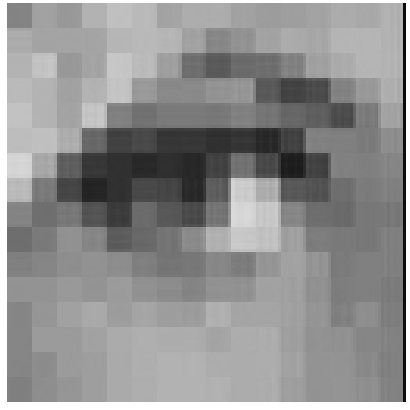
Gradient descent optimizing

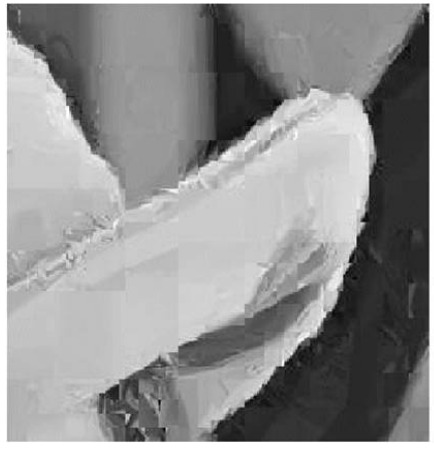


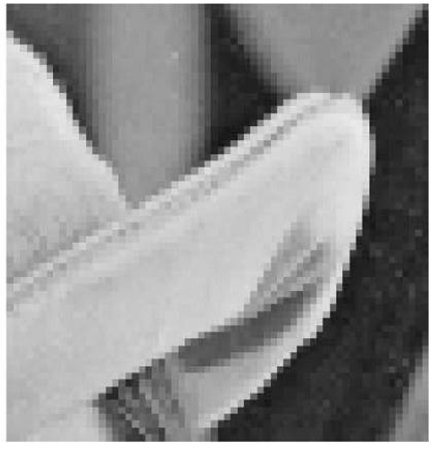
Unfortunately, gradient descent doesn’t really pay off in terms of image quality and it is very computationally expensive

Fractal zoom









Fractal Denoising

Most approaches to denoising involve some type of local smoothing, with the assumption that the noise is additive and zero mean.

Using fractal compression as a basis for denoising is natural since fractal compression involves some local smoothing.

This smoothing arises from two sources: the mean-square fit in the region matching and the contractivity in the grey-scale maps, ϕ_i.

We can also try to estimate the noise-free fractal parameters and use these in the fractal denoising.

Any of the fractal compression schemes can be used for denoising.

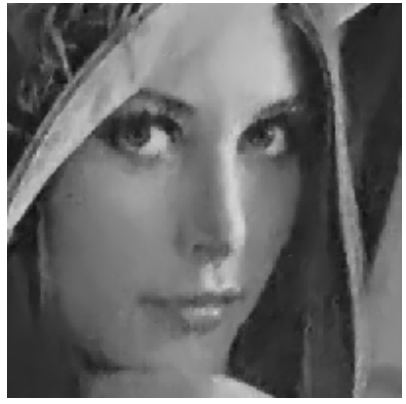
For this application, we don't really care about compression so approximation schemes which expand are ok.

Example: Fractal Denoising

Original

Standard scheme, cycle spinning with 16 shifts

Noisy



Quadtree scheme, cycle spinning with 16 shifts

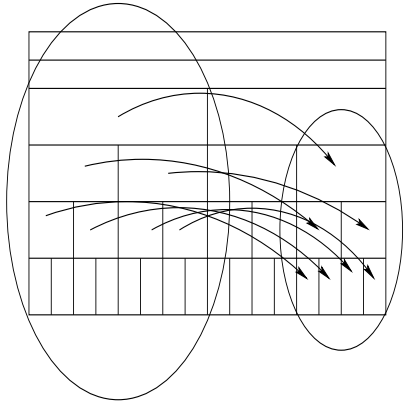
Mixing IFS and wavelets (IFSW)

The key observation in mixing IFS and wavelets is the fact that wavelets have a natural notion of scale.

We expand our signal in some wavelet basis and then try to do an IFS type compression on the wavelet coefficients.

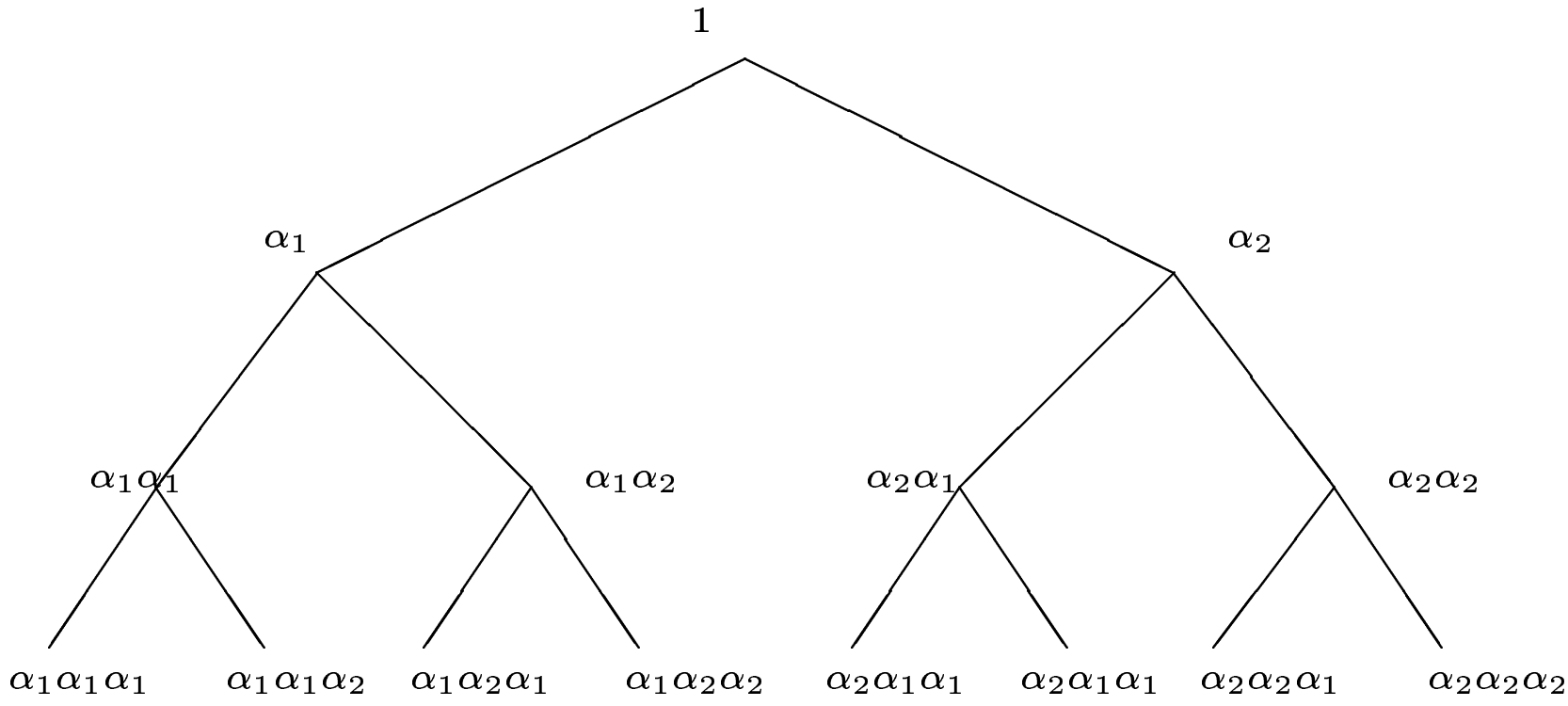
For simplicity, we restrict our discussion to one dimension. The extension to two dimensions is simple.

The type of transform considered is illustrated as



Basically we are trying to extrapolate the fine resolution wavelet coefficients from the coarse resolution wavelet coefficients.

Usually one will store all the expansion coefficients up to some level and then try to use an IFS to predict the higher frequency ones.



One of the main motivations for this is that the wavelets overlap, thus the “blocks” in the “wavelet coefficient space” are overlapping in “picture space”.

However, since the wavelets are orthogonal, the different “blocks” still represent independent information.

This helps to eliminate some of the blockiness from the fractal block algorithm.

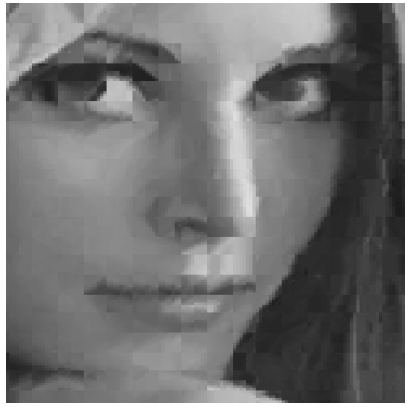
Wavelets are localized in both frequency and space, thus different “blocks” correspond to different parts (spatially) of the picture.

If we use the separable Haar basis, IFSW compression is exactly the same as IFS compression.

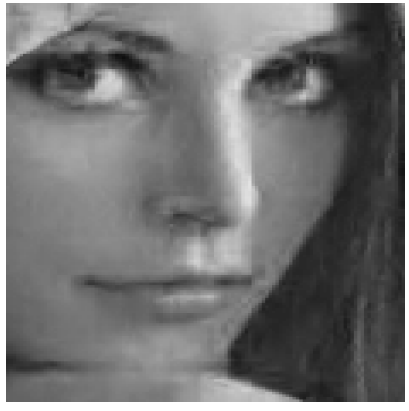
For other wavelets, the relationship is more complicated.

One could also try to do an IFS-type compression scheme in other transform spaces (or using other orthonormal basis).

Comparison: Fractal/Fractal-Wavelet

Standard IFSM scheme

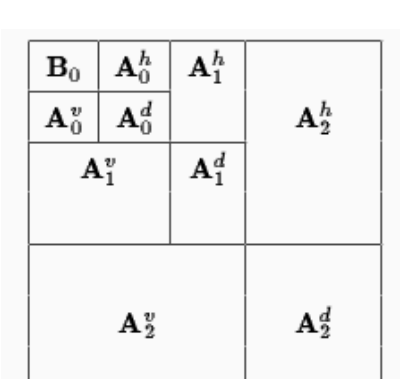



IFSW scheme

Extensions to IFSW

There are several natural ways to generalize the basic IFSW transform.

The 2D wavelet coefficients are



We can picture a general 2D IFSW as

$$M : \begin{array}{|c|c|}\hline B_{000} & A^h_{000} \\ \hline A^v_{000} & A^d_{000} \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|}\hline b_{000} & a^h_{000} & \alpha^h_{00}A^h_{000} & \alpha^h_{01}A^h_{000} \\ \hline a^v_{000} & a^d_{000} & \alpha^h_{10}A^h_{000} & \alpha^h_{11}A^h_{000} \\ \hline \alpha^v_{00}A^v_{000} & \alpha^v_{01}A^v_{000} & \alpha^d_{00}A^d_{000} & \alpha^d_{01}A^d_{000} \\ \hline \alpha^v_{10}A^v_{000} & \alpha^v_{11}A^v_{000} & \alpha^d_{10}A^d_{000} & \alpha^d_{11}A^d_{000} \\ \hline \end{array}$$

Introduction to CBIR

Content Based Image Retrieval is an application where you query an image database and the database returns the N most "visually similar" images.

Of course, the major difficulty is defining what "similar" means – this is an ill-posed problem.

The other problem is one of speed – the query should take only a couple of minutes even with databases of 100 000 images or more. This speed restriction severely constrains the possible image analysis.

The usual method is to calculate "image descriptors" – these are typically vectors of floating point numbers.

These same descriptors are computed for the query image and compared to those for the images in the database. A score is calculated for each image and the top N images are returned.

The mathematical challenge is to discover descriptors and metrics to compare the descriptors.

Example: Color Histogram

As an example, a simple image descriptor is a *color histogram*.

To compute this descriptor, choose some number, say 24, of representative colors.

Map all the colors in an image to one of these representative colors.

Scan through all the pixels in the image and record how many times each representative color is used.

We now must compare two color histograms to compute a distance between two images.

We could use an l_p metric, but this has disadvantages. For some purposes, an *Earth-Movers* distance is better.

All l_p metrics would measure these distances as the same, but in color space blue might be closer to purple than yellow.

Thus, we want a metric that measures both the “horizontal” difference as well “vertical” difference.

Fractal Descriptors

During fractal compression, a lot of data about the image is generated.

This information can be used to try to classify the image.

A very simple example of this would be an overall collage error. That is, once the final fractal "code" T has been found for the image y, compute $d(y, T(y))$.

This gives a measure of how self-similar the image is (at least under this class of IFS).

One could also compute a histogram of "local collage errors"– that is, the matchings for each parent-child block matching.

Other possibilities include statistics of relative location of parent-child matchings, statistics of α and β parameters, etc.

Fractal Descriptors (cont)

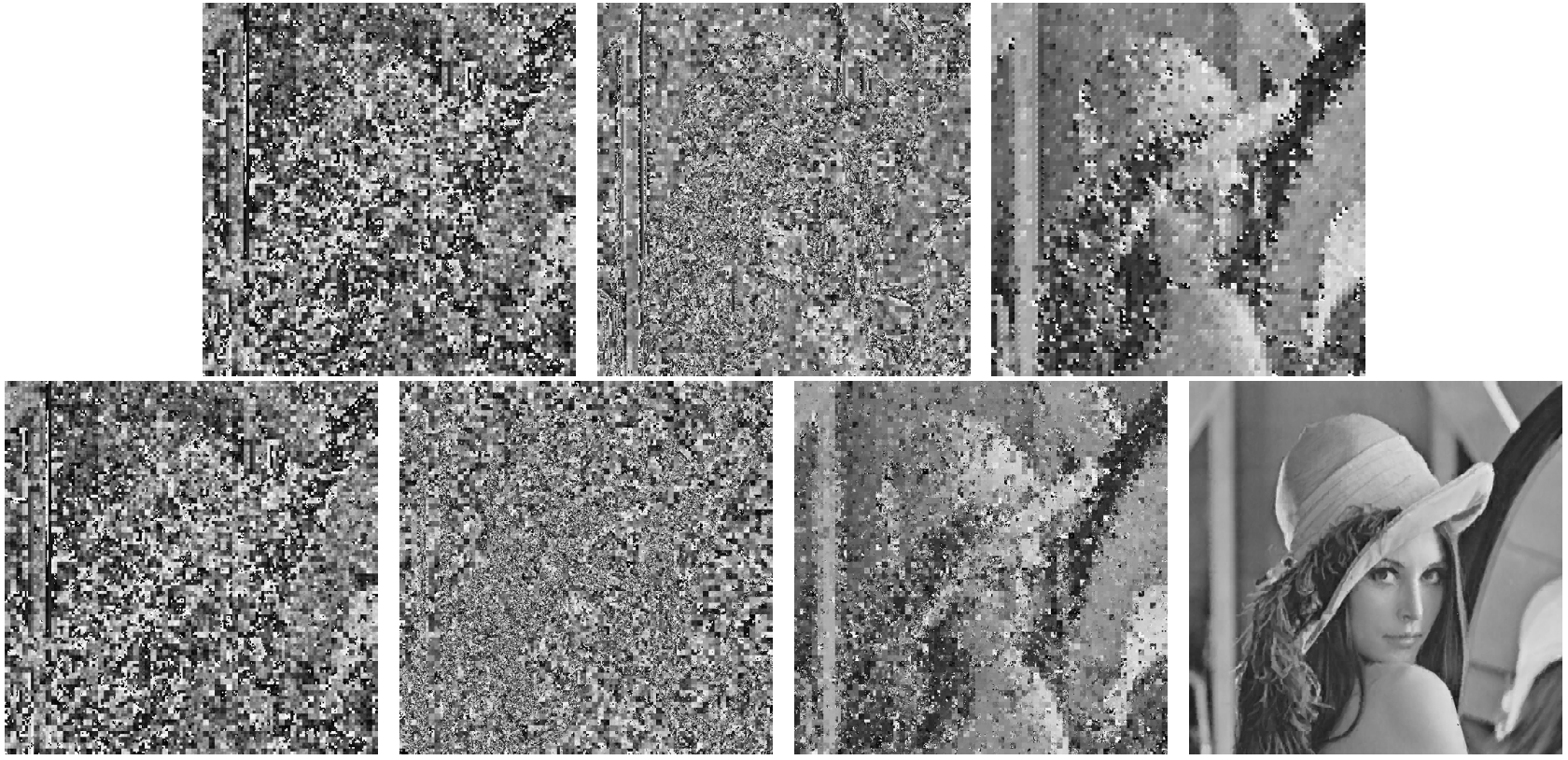
The meaning of these fractal descriptors is not completely clear, but they do provide one way of measuring similarity.

Which metric to use for which descriptor is also not at all clear.

Edge information is partially stored in the parent-child matchings.

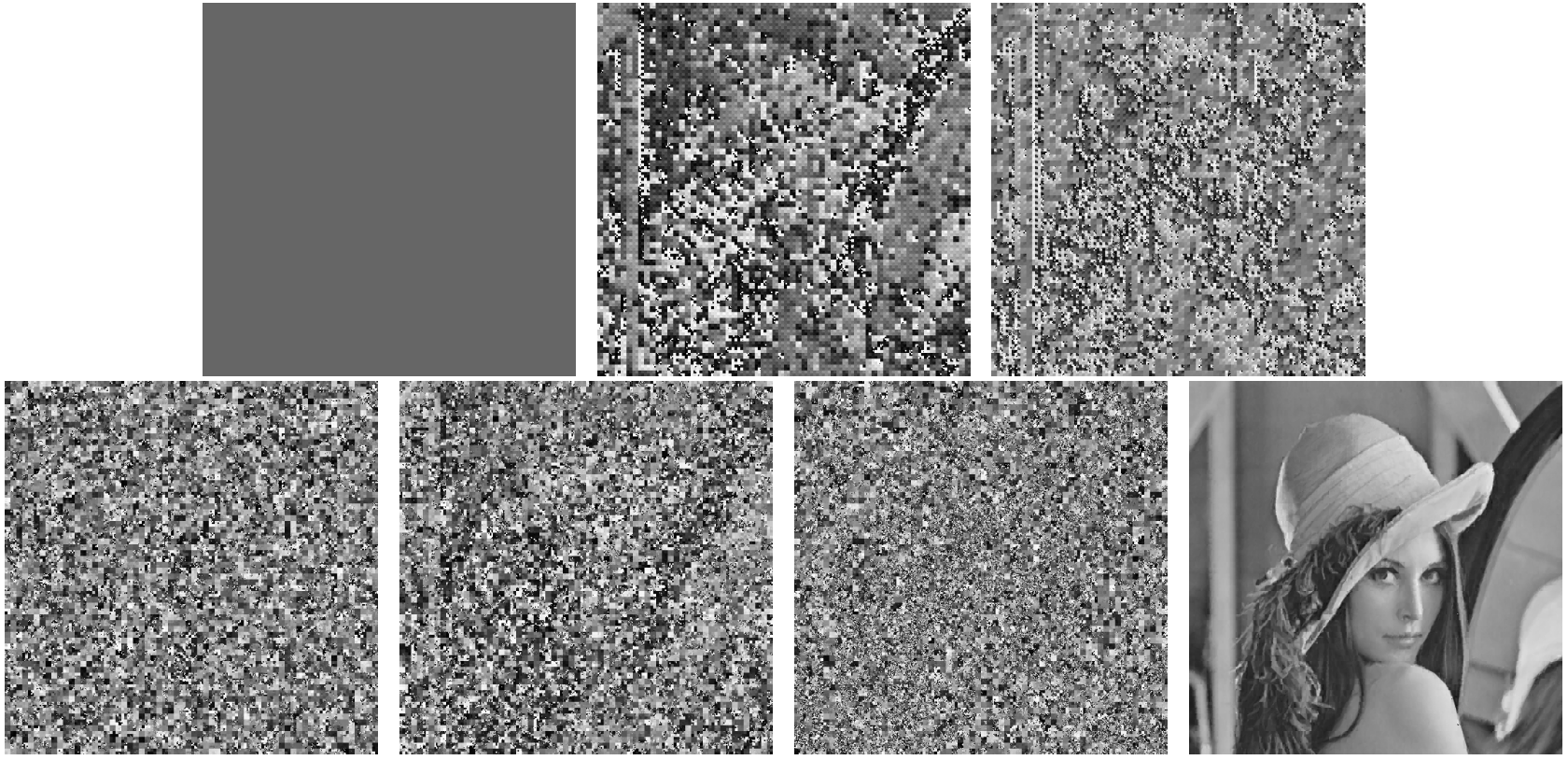
The β's partially encode the "global look" of the image. They represent something like the average value of the image over the child block.

Meaning of IFS parameters



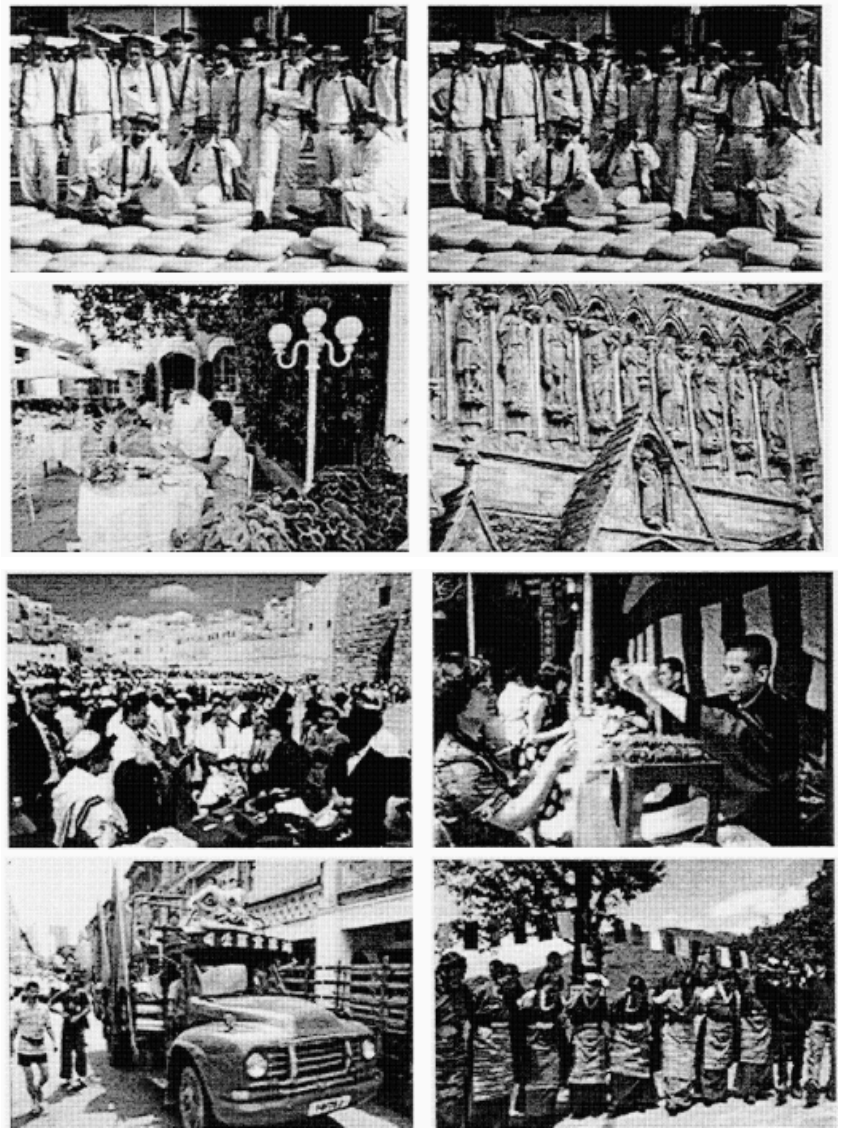
Fixed α, β, index,
random α, β, index, reconstructed.

IFS parameters (cont)

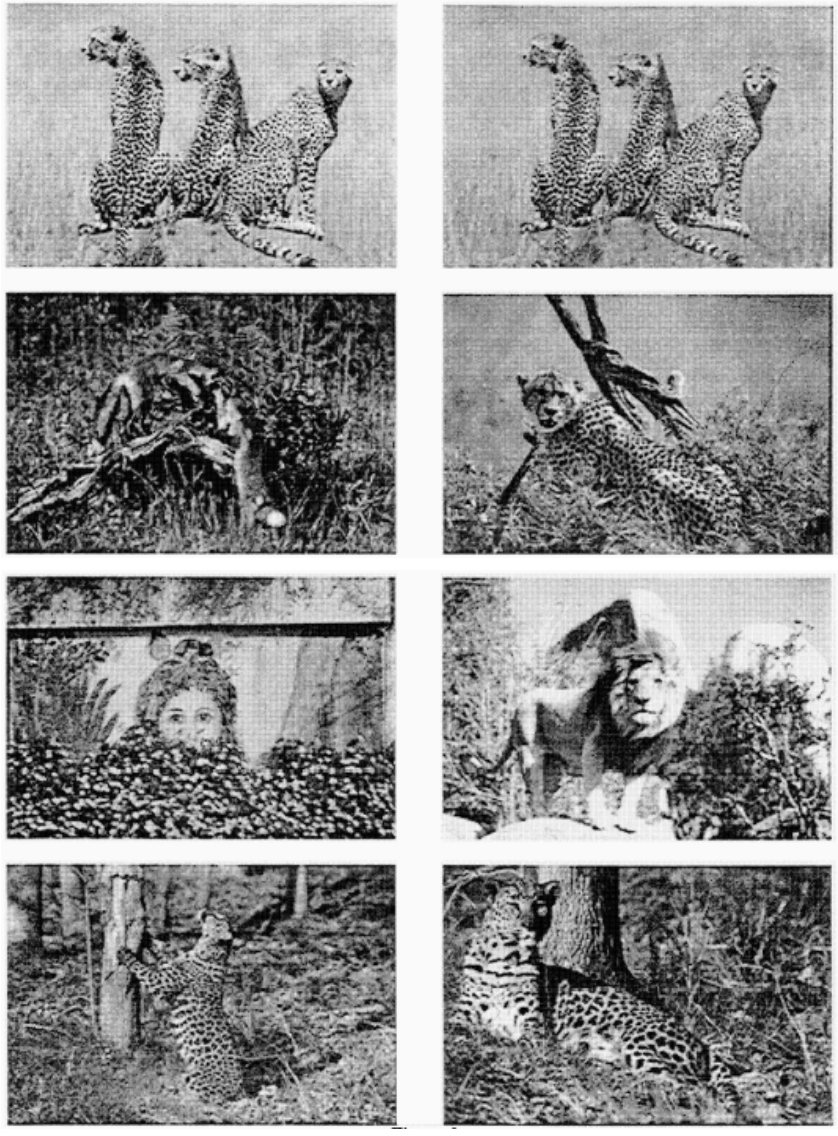


Fixed (α, β), (α,index), (β,index),
random (α, β), (α,index), (β,index), reconstructed

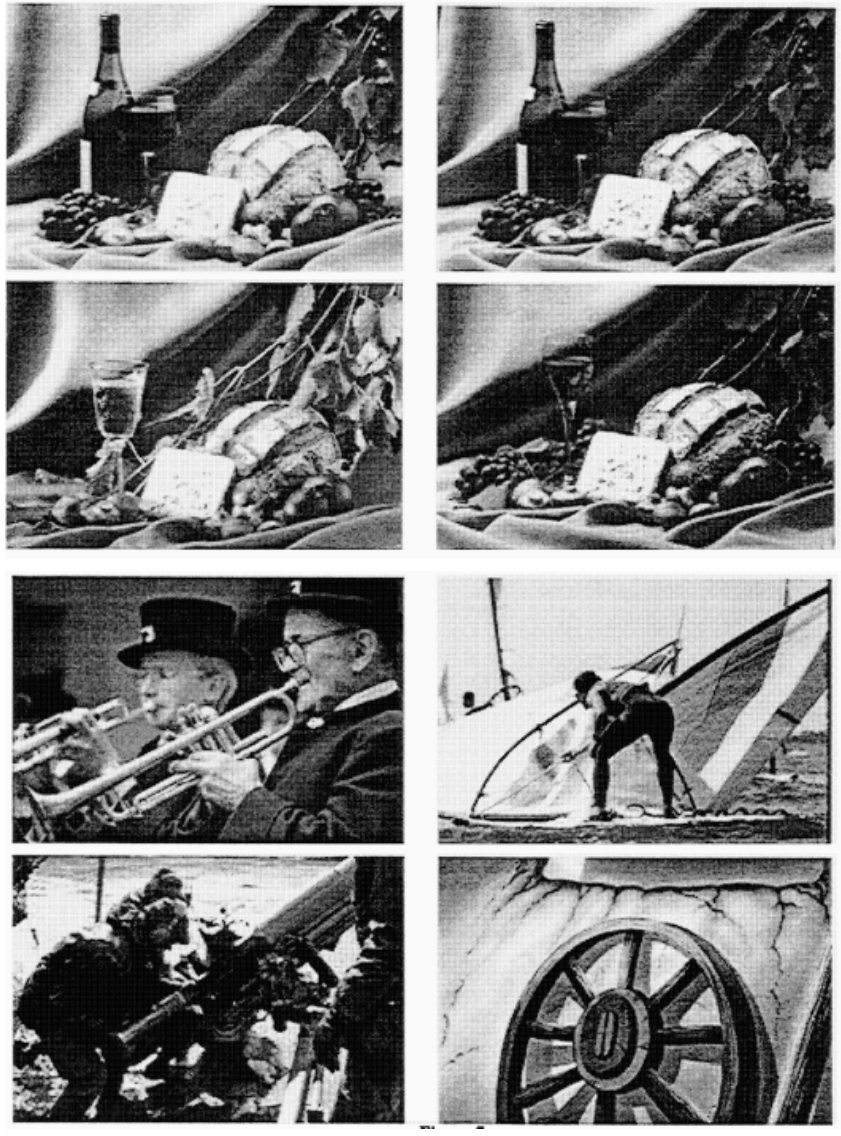
Example: Image Matching



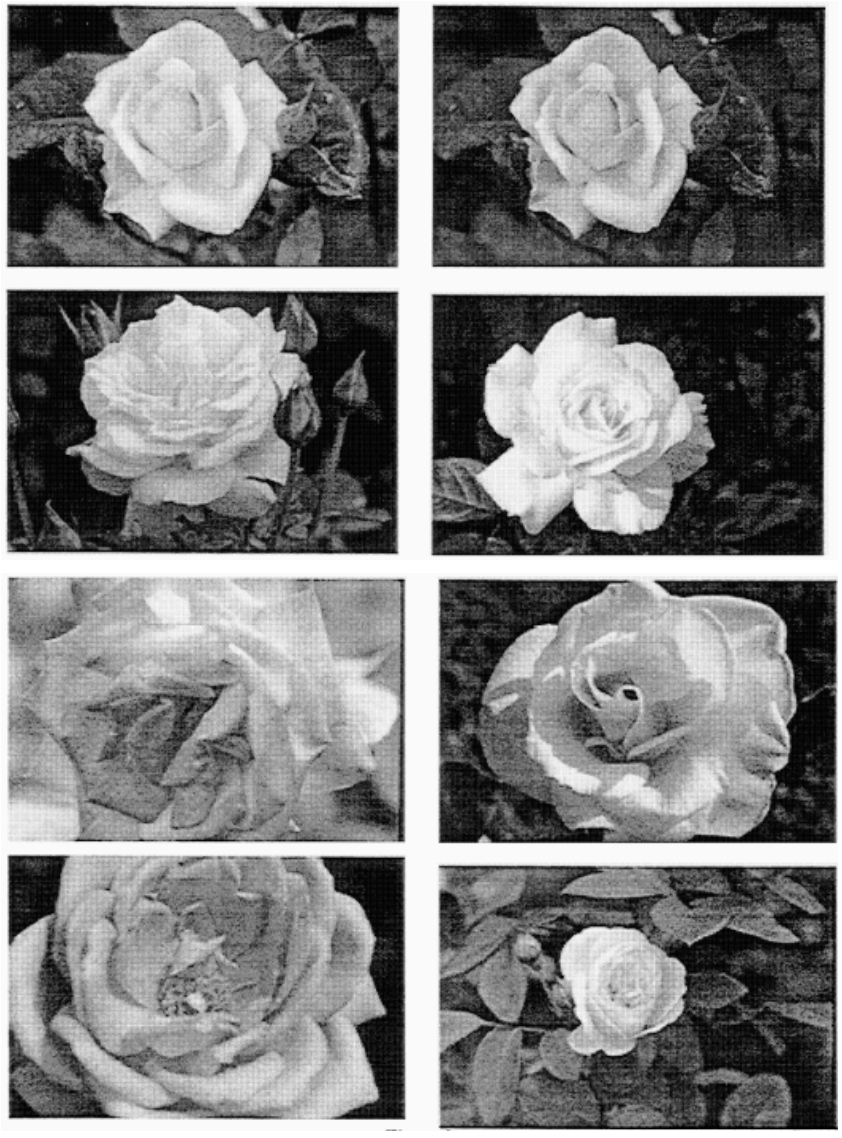
Example: Image Matching



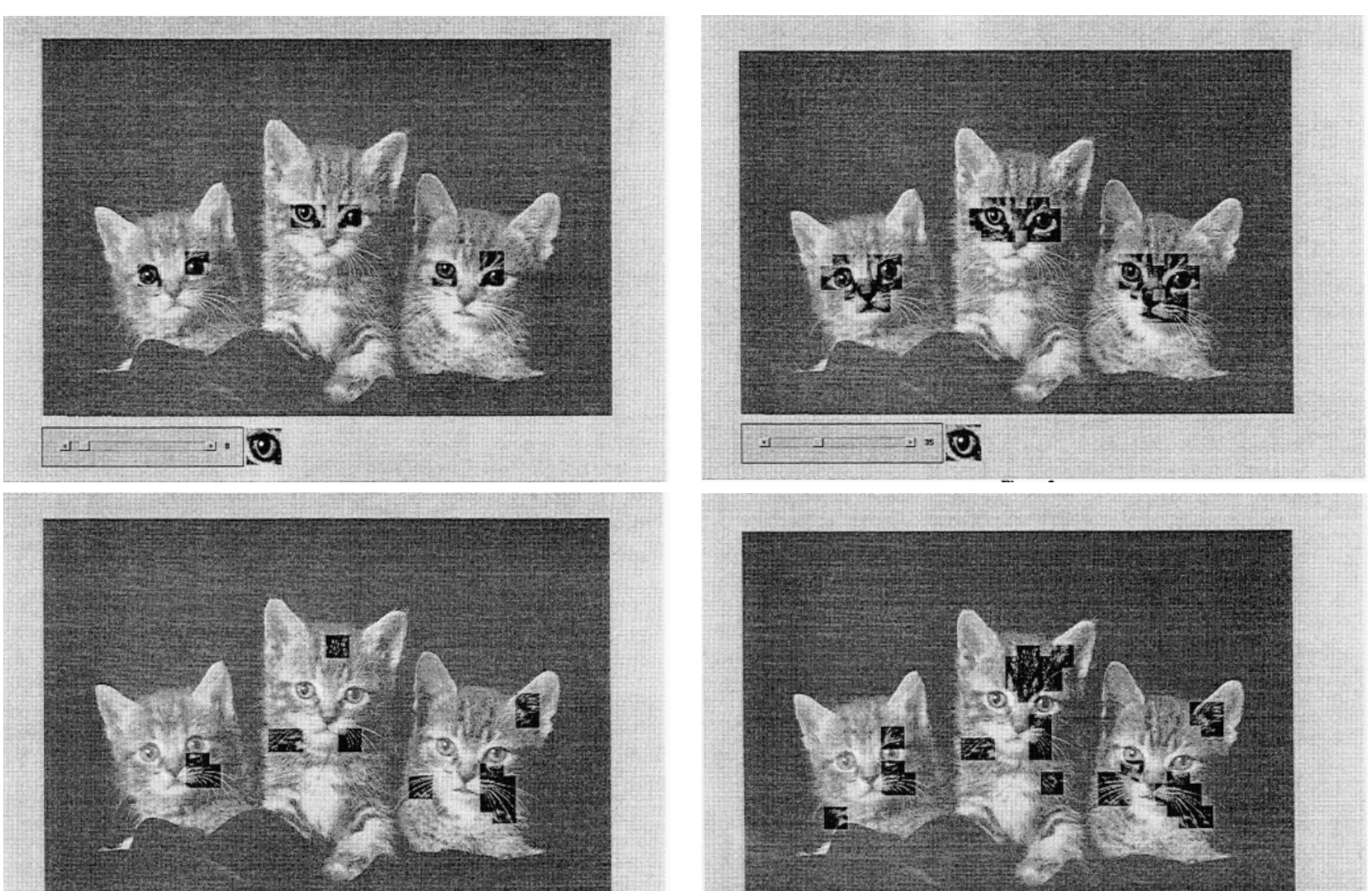
Example: Image Matching



Example: Image Matching



Finding subimages



Multifractal Based Methods

Basics

"The multifractal analysis of images consists in defining a sequence of capacities on the image, computing its multifractal spectrum, and classifying each point according to the corresponding value of $(\alpha, f(\alpha))$, both in a geometric and a probabilistic fashion." – Jacques Levy-Vehel

A point on an edge (piecewise smooth curve) in the image belongs to a set E_α whose dimension is 1.

A point in a homogeneous region has $f(\alpha) = 2$.

The idea is that edge points should be relatively rare points – too many edge points in a region should be classified as a texture, not a collection of edges.

This allows us to analyze the image in several ways.

Some Examples of Capacities

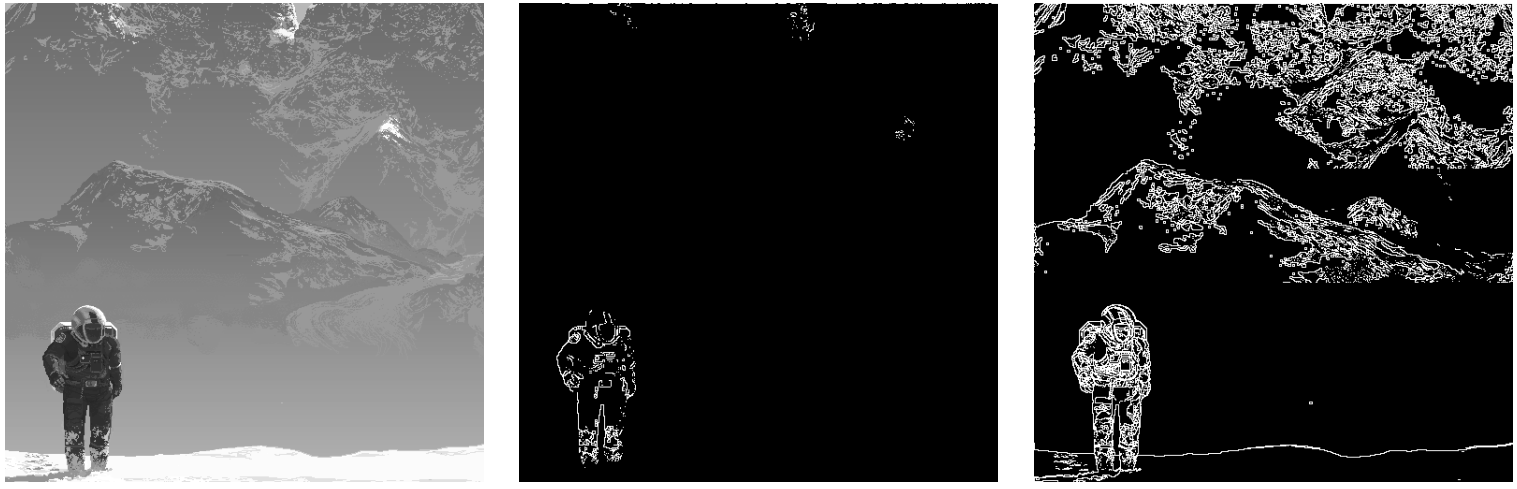
Sum measure: For a set Ω we simply sum over the image grey level values in the set.

Max capacities: We can define these at different resolutions. Basically, we compute the maximum of the pixels in a region Ω or the maximum block (at some resolution) which intersects Ω.

Iso capacities: We can again define these at different resolutions. These are defined to be the size of the largest level set in Ω at some resolution.

The max capacities depend only on the grey level values while the iso capacities depend only on the grey level distribution. Thus, these give different information.

Edge Detection: Example



Original Image, Smooth Edges obtained with *max* capacity, Irregular Edges obtained with *max* capacity.

We define a point to be on a “smooth” edge if it belongs to a set E_α whose dimension is 1.

We define a point to be on an “irregular” edge if it belongs to some set E_α whose dimension is between two values, say 1.1 and 1.5.

Edge points are defined using both local information ($\alpha(x)$) and global information ($f(\alpha)$).

Multifractal Denoising

The basic idea in using multifractal analysis in image denoising is that we want to increase the Hölder function α of the image.

We want to do this in such a way as to preserve the important features of the image, particularly edges.

The denoised signal $\hat{X}$ is thus viewed as the image closest to Y which has the desired Hölder function, α_X.

Usually we take $\alpha_X = \alpha_Y + \delta$, where $\delta > 0$ is some function (so that $\hat{X}$ everywhere has greater smoothness than Y).

An immediate problem arrises: if the noise is additive white noise then $\alpha_Y = -1/2$ a.s. everywhere. Thus there is no way to extract information about α_X from α_Y.

The solution to this is that we have only a limited number of resolutions. We craft our measures of regularity to agree with our intuition at these resolutions.

Given this, we still have to be able analyze and prescribe the local Hölder regularity.

One solution to both problems involves wavelets. There are simple estimates on the wavelet coefficients which guarantee a certain local Hölder regularity:

$$|d_{i,j}| \leq C 2^{-j(\alpha+1/2)}.$$

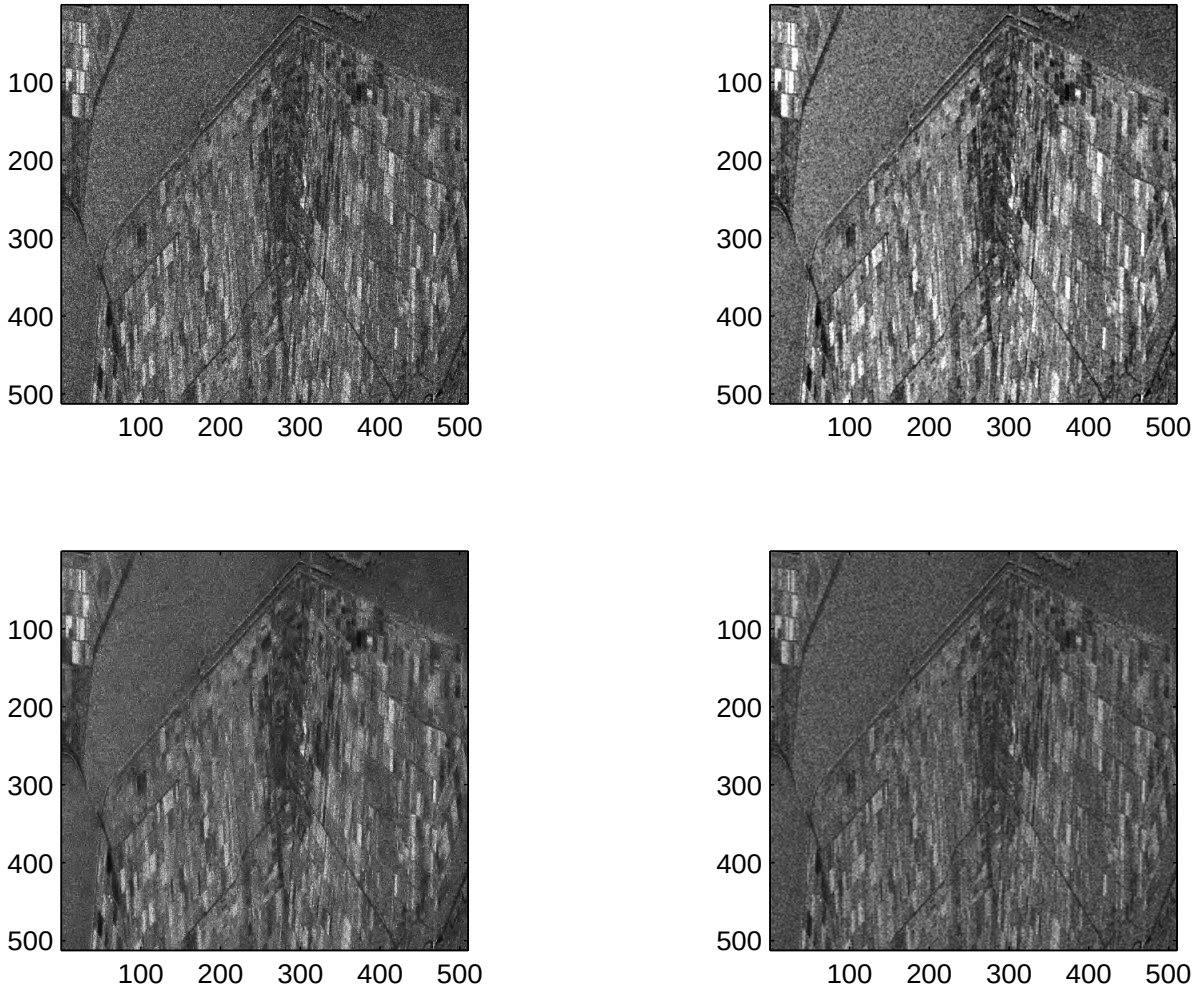
Thus, given the wavelet coefficents $d_{i,j}$ of Y we modify them to obtain the coefficients $c_{i,j}$ of $\hat{X}$, hopefully with the desired local smoothness and still close to Y.

Prescribing the Hölder Function

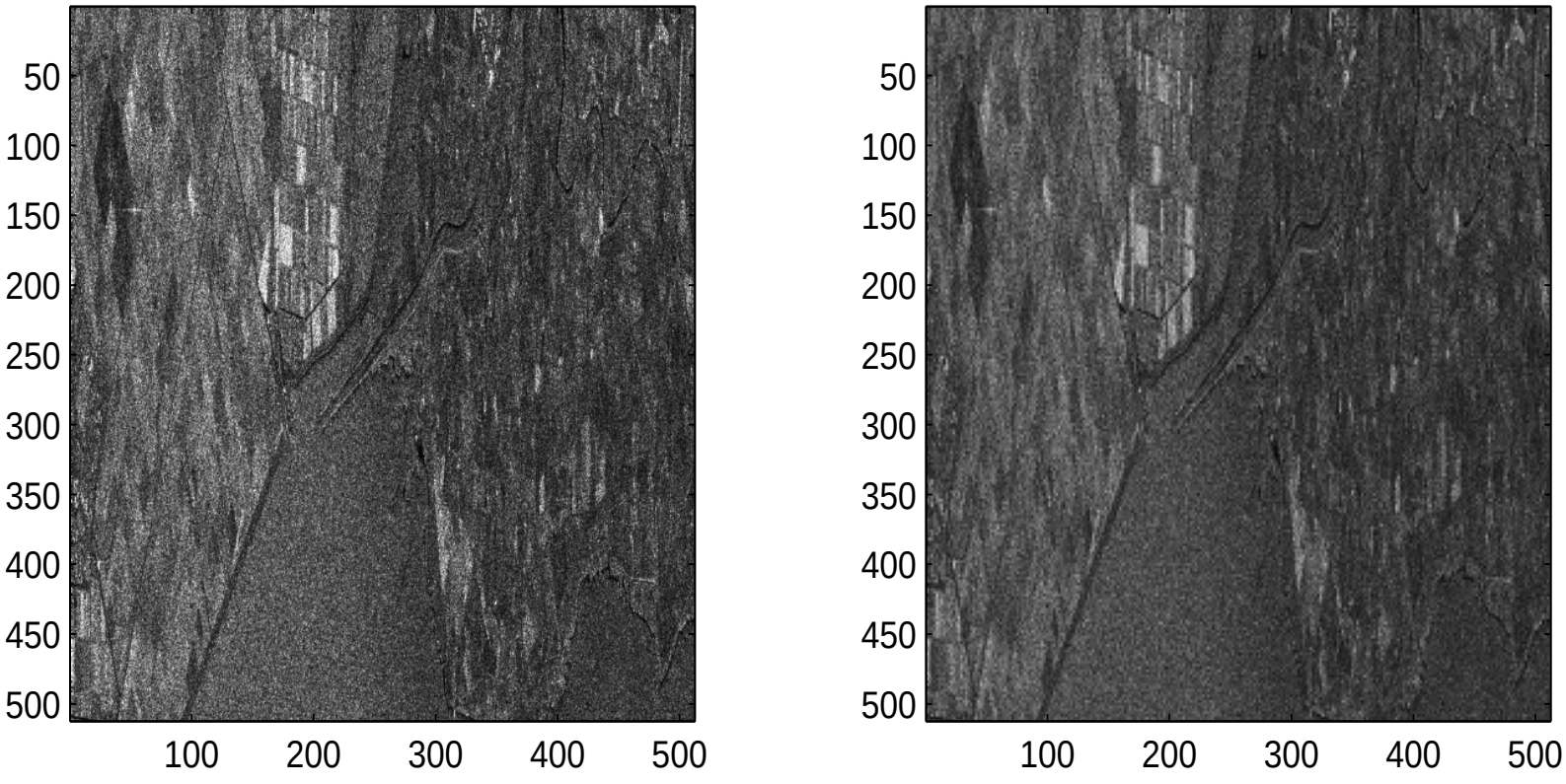
Each wavelet coefficient at a “large” resolution influences many points. For this reason, the minimization problem is a global one – we must solve simultaneously for all the wavelet coefficients as one large optimization problem.

We obtain a system of constraint equations which is linear in the logarithm of the desired wavelet coefficients. Our objective function is the risk $E(\|\hat{X} - Y\|)$.

To simplify the situation, we could assume that $c_{i,j} = B_j d_{i,j}$ with $0 < B_j < 1$. Thus, we shrink the coefficients of Y to get those of X. Notice we allow ourselves to shrink each resolution by a different amount.



Original SAR image, Kuan median filtering, wavelet shrinkage, local regularity based denoising.



Original SAR image and local regularity based denoised version.

Resources

- *Fractal Image Compression*, Yuval Fisher, editor, Springer-Verlag, 1995.
- The Konstantz Paper Collection, `www.inf.uni-konstanz.de/cgip/fractal2/html/index.html`
- Waterloo Fractal Compression Project, `links.uwaterloo.ca`
- Jacques Levy-Vehel's web page at `fractales.inria.fr`.
- "A wavelet tour of signal processing", Stéphane Mallat, Academic Press.
- "Introduction to the multifractal analysis of images", Jacques Levy-Vehel.