The **Fractional Dimensions** pop-up menu

Jacques Lévy Véhel

12 January 2001

This text presents a brief explanation of the functionalities of the **Fractional Dimensions** pop-up menu.

Contents

1 Overview

Fractional dimensions are the best known part of fractal analysis. A huge number of dimensions have been defined in various fields. In the current implementation of **Fraclab**, only two dimensions can be computed: the box dimension using a plain box method, and the regularization dimension.

2 Regularization Dimension

This dimension is defined is the following way: One first computes smoother and smoother versions of the original signal, obtained simply through convolution with a kernel. Now, if the original signal is "fractal", its graph has infinite length, while all regularized versions have finite length. When the smoothing parameter tends to 0, the smoothed version tends to the original signal, and its length will tend to infinity. The regularization dimension measures the speed at which this convergence to infinity takes place. In many cases, this will coincide with the usual box dimension. In general, it can be shown that the regularization dimension is more precise than the box dimension, in the sense that it is always smaller, but still larger than the Hausdorff dimension. In addition, the regularization dimension lends to more robust estimation procedures for various reasons. One of them is that we may choose the regularization kernel. Also, the smoothed versions are adaptive by construction. Finally, the smoothing parameter can be varied in very small steps, as box sizes have to undergo sudden changes. Another advantage is that, due to the fully analytical definition of the regularization dimension, it is easy to derive an estimator in the presence of noise.

Check first, as usual, the **Input data name**. The current implementation of the regularization dimension allows to deal with both 1D and 2D signals, in a way which is transparent to the user: Just input your signal, and **Fraclab** will recognize its type.

Second, you need to decide on the minimum and maximum amount of smoothing. This is done by specifying the corresponding sizes for the kernel, using the **Nmin** and **Nmax** parameters. This is expressed in sample units, i.e. a value of 5 means that your kernel will have a "width" of 11 sample points (the precise definition of the width depends on the kernel). You then choose a **Kernel** shape among **Gaussian** and **Rectangular**. The width in the Gaussian case in simply the standard deviation, while it is the number of non zero coefficients in the case of the rectangular kernel. The **Voices** parameter lets you choose how many smoothed curves you want to compute. A value of, e.g., 64, means that 64 smoothed signals will be generated with smoothing parameters regularly spaced between **Nmin** and **Nmax**. As alluded to above, the estimation of the regularization dimension can accommodate for the presence of additive white Gaussian noise in the data. If you want to use this feature, just enter the standard deviation of the noise in the **StD** box below the **Noise** heading (estimate your noise using any classical method, or use the built-in estimation available in the Denoising menu). As usual, the dimension will be estimated through regression, and you may choose which type of regression you will use, i.e. **Least Square Regression**, **Weighted Least Square**, **Penalized Least Square**, **Maximum Likelihood** or **Lepskii Adaptive Procedure**. All these methods are well-known except the last one, for which you may consult reference (2). If you select **Range Specify**, you will be able to choose interactively a region where an approximate linear behaviour holds (the next paragraph details how to do this). Otherwise, set the **Range** to **Automatic**. It is sometimes instructive to look at the **Regularized graphs**: when this option is checked, you will get a graphic window displaying all the regularized version of your signal (these will be contour plots in case you are dealing with an image). **No regularized graphs** is the default. In case you are interested in keeping the regularized versions for further processing, check the **Save regularized graphs** button. **Forget regularized graphs** is the default (note that this option is disabled if **No regularized graphs** has been checked). Be careful that, if you use a large number of voices and you keep the regularized graphs, you will add a large number of variables to your environment (each regularized graph is a distinct structure).

You are now ready to hit the **Compute** button. If you decided to use the **Automatic** range, then you will simply get the estimated dimension to the right of the box **Regularization Dimension=**. If you selected **Range Specify**, then a graphic window will pop up. In both graphs of this window, abscissa represents the logarithm of the smoothing parameter. In the lower graph, the ordinate represents the logarithm of the length of the regularized versions. If the parameter **StD** is 0, the upper graph simply represents the increments of the lower graph. This device is useful because it allows to emphasize more clearly a possible linear behaviour (linearity in the lower graph translates into constancy in the upper graph). Otherwise (if **StD** is positive), the upper graph displays a curve related to the relative strength of the noise and the signal at all scales (see reference (1) for more). Using the cross that appear when you point inside the graphic window, choose a region where approximate linearity holds: Select this region by clicking on its endpoints. A red line showing the regression will appear, and the corresponding estimated dimension will be displayed above the lower graph. Repeat this selection operation until you are satisfied, then hit **return** on your keyboard. The cross will disappear, and the final estimated dimension will be displayed to the right of the box **Regularization Dimension=**, at the very bottom of the **Regularization Dimension** window.

Note that if you select both **Range Specify** and **Regularized graphs**, the graphic window displaying the regularized graphs will appear first, then the window for the regression range selection will appear on the top of it and will mask it. Just move it if you want to inspect the regularized graphs.

3 Box Dimension: Box method

This is the well-known box dimension. The interface works exactly in the same way as the regularization dimension one, except that the **Nmin** and **Nmax** parameters now refer to the minimum and maximum box sizes. And, of course, there are no kernels, nor regularized graphs to visualize or to save. Please keep in mind that the computations can get very long if **Nmax** is chosen larger than 8. Note finally that only the 1D version has been implemented so far. The reason why the box dimension method is not so much developped in **Fraclab** is that the regularization dimension is better tool, both from a theoretical and computationnal point of view: indeed, the regularization dimension is always a lower bound to the box dimension, and, as you'll quickly check by manipulating **Fraclab**, estimations on numerical samples are in almost all cases less accurate for the box dimension.

4 Homework

Try experimenting on some simple curves, such as the graph of a Weierstrass function or a path of a fractional Brownian motion. Check that the regularization dimension estimator always gives superior results as compared to the box dimension estimator.

Try also the following: synthesize a deterministic Weierstrass function with the default parameters, except that you put **Sample Size** = 4096. Then add to the output signal, *Wei0*, a white Gaussian noise (type first "x= randn(4096,1);" in your matlab window, then "y = Wei0 +0.2*x;". Import y to **Fraclab**'s workspace by clicking on **Scan Workspace** and selecting y in the window titled **Import Data from Matlab Workspace** that appears). Compare visually *Wei0* and y, then compute the regularized dimensions of those two signals: Use first the default options. In the case of *Wei0*, the points in the graph showing the logarithms of the length of the regularized versions are reasonably well aligned at least between abscissa 2 and 4, and, by selecting this region with the cross, you get an estimated dimension of around 1.52, which is not too bad. If you analyze y, however, you see that there is not significant region in the graph showing the logarithms of the length of the regularized versions where a linear behaviour holds. Set then **StD** = 0.2, and estimate again on y. You'll see as previously on the graph the small black circles corresponding to y, and, in addition, red stars that show the estimator corrected to take into account the noise. The red stars are close to the circles to the right of the graph, then depart from them significantly as we move to the left. Here is why: Points on the right correspond to a large amount or smoothing, or "low frequencies", for which the signal will dominate over the noise. In this region, there is not much to compensate for, as the lengths of the smoothed original and the smoothed noisy signal should not be too different. On the extreme left, however, we are looking at high frequencies (i.e. almost no smoothing), and we mainly analyze noise: The length of the original regularized signal is here significantly smaller that the observed one: this is why the estimated "true" length (the red stars) are well below the measured length (the black circles). The red stars in the region between abscissa 2 and 4 should again be roughly aligned, and selecting this range with the cross should still give you an estimated dimension not too far from 1.5.

5 References

(1) F. Roueff, J. Lévy Véhel, *A Regularization Approach to Fractional Dimension Estimation*, Proceedings of Fractals 98, Malta, October 1998.

(2) C. Canus, *Robust Large Deviation Multifractal Spectrum Estimation*, Proceedings of International Wavelets Conference, Tangier, April 1998.