Finite Time Growth (Non-Linear Analysis)

Wessel et al (2010)'s Finite Time Growth is,

$$\lambda_k^{(n,\tau,T)} = \frac{1}{T} ln \frac{\| X_k^T - \bar{X}_k^T \|_2}{\| X_k - \bar{X}_k \|_2}, k = 0, \dots N - (n-1)\,\tau,$$

where, λ = growth rate; T = 1 (the number of steps), N = 9 (The embedding dimension), and τ = 1(the delay used)

This multi-dimensional analysis required more calculation than any of the simpler equations given above. Thus, to understand our workings we would require the reader to examine the Excel program we have written and included in the digital appendix alongside the following reading. The program cannot be printed here as the first spreadsheet alone comprises an array of 144,000 equations to be performed simultaneously. At the smallest print setting possible, with each block ≤1.23cm across the spreadsheet comprises 2168 pages so would be an unreasonable addition to the printed version of this thesis.

We began by taking the first set of nine points as the starting coordinate. These become a_1, b_1, c_1, d_1, e_1, f_1, g_1, h_1 and i_1. Then find the euclidean distance (d) between the nine-dimensional point framed by those coordinates and every other point framed by a set of nine coordinates e.g. a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_1 and i_2, by performing the equation :

$$distance = \sqrt{\begin{matrix}(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2 + (d_1 - d_2)^2 + (e_1 - e_2)^2 \\ +(f_1 - f_2)^2 + (g_1 - g_2)^2 + (h_1 - h_2)^2 + (i_1 - i_2)^2\end{matrix}}$$

We compare all distances to find, for every nine-dimensional set, which other distances are nearest neighbours, and what are the mean neighbour distances.

The instructions for using the spreadsheet "FTG ARRAY" are as follows:

1. Copy a single column of numbers representing all consecutive period data in question.

2. Go to the first tab entitled "Step 1". This is a 356 column by 121 row grids.

3. Paste the column into box "II2". This is the 243rd column on the second row.

4. Go to the second tab entitled "Auto-arrange" and simply press "Cmd C" to copy the entire highlighted array without changing any element within the tab. A block comprising rows 2 through 121 and columns B through AUI112 is pre-highlighted.

5. Go to the third tab entitled "Step 2" and in node A2 paste values. Make sure to paste values only, not functions. The paste function can take several minutes.

6. Node A1 gives the number of 9 dimensional strings in the array. Take this as "n".

At this point the user can choose to perform the next task the slow way, or the fast way:

a. The slow way

In the first row, observe that an option is given at the top of each set of values to sort the values with a small grey button with an upward pointing arrow. Press this arrow for each of n columns and press "ascending". This takes a while, causes severe eye strain, and saving often is recommended.

b. The fast way

Go to computer system preferences > keyboard > tick “Use F1, F2… as standard function keys”. This will nullify the use of function keys for processes such as adjusting the screen dimness or volume but instead allows for the use of quick functions we have programmed into the spreadsheet. Much of the spreadsheet has been collapsed for its use. Click on the second visible node in the second row of “step 2”. Then press the series of keys “F1” “Enter” “next arrow” n times. As this does not require carefully looking at the screen to press the various ‘sort’, ‘ascending’ and ‘OK’ buttons as they pop up, it means that the “F1” “Enter” “next arrow” sequence can be done with the user’s eyes shut at high speed. We found that the processing speed of the spreadsheet was about 3x slower than the hands could work so, rather than trying to keep a count of how many times the sequence had been types, the user could stop typing the sequence when the program had reached around n=30 (for example for a 90n string) and take a small break (e.g. take a sip of tea) while the spreadsheet caught up with the remaining functions. Going over n series of button presses does not matter as Excel simply does not function when no more n’s exist and instead beeps loudly, at which point the user can press ‘escape’ and move on.

7. Copy rows 2 through 7 onto the clipboard.

8. Go to the fourth tab entitled “Step 3”. Enter into node A1 and paste values. Make sure to paste values only, not functions. While this is still highlighted, copy again. Enter into node A8 and paste transpose. While this is still highlighted, copy again.

9. Go to the fifth and final tab entitled “Step 4”. Enter into node A12 and paste.

10. Return to the first tab “Step 1” and copy the original column of period values.

11. Return again to “Step 4”, enter into node A2 and paste transpose.

12. “Find and Replace” all nonsense “§” symbols with the “=” sign.

13. Near the top of the spreadsheet in columns H and I (the row changes slightly between to between 18, and 22, according to how many dimensions are being calculated) is a value which will read “ J = ”. This value is the final value in the column of results given by the nearest neighbour equations. A slight adjustment of the spreadsheet will be needed by checking the nodes entitled “Mean” and “SD” which will have functions which read, “=SUM(J13:J---)/COUNT(J13:J---)” and “=STDEV(J13:J---)” respectively. For these functions replace “---” with the number given by “J = ”. This will give the mean Finite Time Growth value and standard deviation for the data set in question.

* End of Instructions*

To interrogate the spreadsheet, in “step 1”, the first 122 columns automatically repeat column “ii” in a vertical orientation, and the 123rd to the 242nd columns transpose that same array horizontally. Columns 244 and above call up this horizontal and vertically displayed repeated data, and apply the equation for distance across an 120 x 120 equation array of 144,000 equations. The “Auto-arrange” tab is not a named ‘step’ because absolutely nothing should be inserted into, or adjusted, regarding the content of this tab. It automatically gathers and rearranges answers from “step 1” in a format that can be pasted into “step 2” for further application of the equations. Answers to equations from “Step 1” giving Euclidean distances are interspersed with columns coded with reference to nodes in “step 4” along with the intentionally non-sensical notation §. This nonsense notation was especially chosen as it would not perform a function on the data at this point. “Step 2” is there to arrange all the nearest neighbours for every nine-dimensional array into order. It is interspersed with nonsensical references that will be converted to sense in “step 4.” “Step 3” is simply there as a stop gap in order to make sure values are copy pasted in the correct orientation and free from formulas. Replacing § with = in “step 4” will force the tab to refer to its own columns of 9 dimensional vectors. Functions in this tab automatically take sets of mean Nearest Neighbour values and square the difference between values from each 9-dimensional array and the first nearest neighbour value of each array. For all 9 values the sum of the square root is divided by the sum of the square root of the previous set of Nearest Neighbour values. Then, the natural logarithm LN is applied, a logarithm to the base e, for each n to n+1 comparison.