# Computing high order dependencies between the different levels and features in limit order books

Matthew Newton

## Introduction

This project investigates various methods of computing dependencies between the different levels and features for limit order book data [redacted] A total of eight methods were created, each of which are to compute a dependency matrix that would summarise the dependency between features.

The standard linear correlation method provides a basic and fast tool for computing dependence, it does not however consider the asynchronous nature of the time-series or any sort of higher order dependence. Consequently, the data is both resampled and filtered (using a second order Butterworth filter) in order to deal with this asynchronicity, this obtains two more dependency matrices. Two further methods aim to deal with the asynchronicity in different ways. The method by Takaki Hayashi and Nakahiro Yoshida[1] resynchronises the data by looking at the changes between two features overlap with each other, whereas Paul Malliavin and Maria Elvira Mancino's method [2][3][4] uses the Fourier transform to resynchronise the data.

Whilst all the above methods produce a good insight into the pairwise dependencies between features, none of them consider a higher order dependency. A student-t copula fit provides the basis for the final three dependency matrices, considering the raw, resampled and filtered data. [5][6][7]

It is also worth mentioning that a lot of the contextual information for this project was obtained from a paper providing an overview on limit order books. [8]

## Overview of how the code works

All of the code within this project has been heavily commented, any issues with regard to how the functions works should be explained within the main script and functions themselves.

The MATLAB code is separated into five folders:

"data" contains the csv files as well as the code to import the csv files into MATLAB, the directory may need to be altered and it is recommended that the csv file should be in the same folder as the "import_data" function.

"estimator functions" contains the four different estimators used to calculate dependency. They all are designed such that the only inputs required are the data to be evaluated and the features that have been selected. The data that is input is dictated by the functions

“calculate_all_dependecies…” All estimator functions will output a nxn matrix, where n is the number of selected features.

“global functions” are the more general functions, some are purely designed to simplify the code and make it tidier, whereas others serve a much greater purpose and do the bulk of the overall computation.

“plotting functions” as the name suggests, all create a separate figure to represent the data and results. The inputs to these functions are produced by the “global” and “estimator” functions.

“scripts” used to contain various scripts that calculated and plotted different results. However, all these have been merged into a single script entitled “main_script_all”, making there only file in the folder.

There are three main ways that the data is evaluated: in distinct separate time intervals, in time intervals but progressively getting longer throughout time (effectively a cumulative time) and with a varying resampling frequency. The names of the functions and various options reflect this, which should make understanding the code and selecting the outputs more intuitive.

## Overview of how to use the code

For the code to work the only file that needs to be open is the main_script_all, from here the user should select the parameters they wish. A description of what each variable does is commented above the variable, not all the variables are relevant for each plot and will have no effect on the respected plots. It is “select outputs” section that determine which plots and results will be produced. A “0” means don’t plot and a “1” is do plot. From this section down, there are no other user inputs that are required as the rest of the code is composed of self-sufficient functions. However, the code has been heavily commented to allow the user to gain an understanding as to how the functions work. There are also some sections of unused code that have been commented out in case the user wishes to reincorporate them. As the code runs there will be messages in the command window indicating the run progress, it provides a good measure for predicting the overall run time.

## Result example

All the results were calculated for the first five levels in the limit order book, any more and the results would be difficult to determine as there would be 1600 values of dependency whereas there are only 400 values when the first five levels are used. The plots have been exported as eps files with their MATLAB figure files, as well as the respective workspaces.

It is important to be able to visualise the data that is being used. Figures 1 shows the bid and ask prices of the first five levels for one day on 3$^{rd}$ October 2016, where the size of each point represents the relative bid and ask sizes for each level. The vertical lines represent the intervals that were taken when computing the dependency matrices. It is clear to see when

the auctioning period takes place as all the bid and ask prices behave erratically between the period of 07:30 and 09:30 before converging to around 88 for the remainder of the day.

Figure 2 uses the same data as figure 1, however it shows each bid and ask price on a separate subplot. The axes are aligned so the differences between each level are more easy to spot. It is worth noting that these plots only show the data for a single day and the market throughout the month will vary massively. However, figure 1 and 2 still provide a good insight into the structure of the books.

As mentioned above this project investigates the effect of resampling and filtering the data. Figure 3 provides a visual aid to see what resampling and filtering the data actually does. The resampling frequency used was one second and the lines in the auctioning period look significantly smoother than before, with the incorporation of filtering seeking to match the discontinuities together.

Differential entropy in this context is a measure of the off-diagonal mass of a dependency matrix and is given by the log determinate of the inverse dependency matrix. From figure 4 is it shown that this metric varies throughout the day, being significantly greater after auctioning than before. The different methods of computing the dependency matrix are also compared, they all vary throughout the day but certain methods tend to be greater than other. There are certain values where the metric blows up to infinity, this is because two or more of the rows are identical. These values have been set to 1000, so there are no errors in the plot.

Figure 5 is similar to figure 4, however the data is the cumulative data instead of distinct intervals. The two plots are similar but there are subtle differences in their shape.

One of the most interesting results in this project is how the merit depends on sampling frequency. Figure 6 uses the data for an entire day and shows how the sampling frequency between 0.01 and 0.1 per second. Unfortunately, no results were saved outside this range, however it was observed that the merits converged to a value at around 5-10 per second.

The information that is contained in the workspace is not clearly laid out and does not give a good representation of the results. There are summary functions that are used to present the results for each method in a standalone MATLAB figure. The figure contains the dependency matrix in a table format, with the row and column names stating which features relate to the dependency. The interval time or frequency value are also stated, along with the differential entropy for that dependency matrix, as well as a list of the most significant dependencies and which features relate to which. No screenshots of the summaries are included in this report but instead there are MATLAB figures available.

As the summarise can only display one method at a time, a visualisation of all the dependency matrices for the different methods has been created. The visualise matrices functions creates an animation of the matrices through time. Each time interval or frequency is updated every three seconds, to create a side by side comparison of the different dependency matrices. The animation can only be created by running the function, meaning no MATLAB files could be included.

## Future developments

The standout task to further the progression of this project is to obtain more results. Currently the script has only been run over the first week due to the limited time available, however it would be very interesting to see how the results vary throughout the month or year. Comparing the different time frames and the length of the time window would provide further insight into how each method performs.

As the functions have only been tested on the raw data, the reliability of each dependency method is left unknown. By simulating data with a built-in dependency structure and then testing it using the current code, an idea of how well the different methods are at recovering the dependencies can be created.

Whilst all the code in this project is complete and free from bugs, there is potential to improve the efficiency and conciseness of the code, as current run times are large.

## Collation of Plots

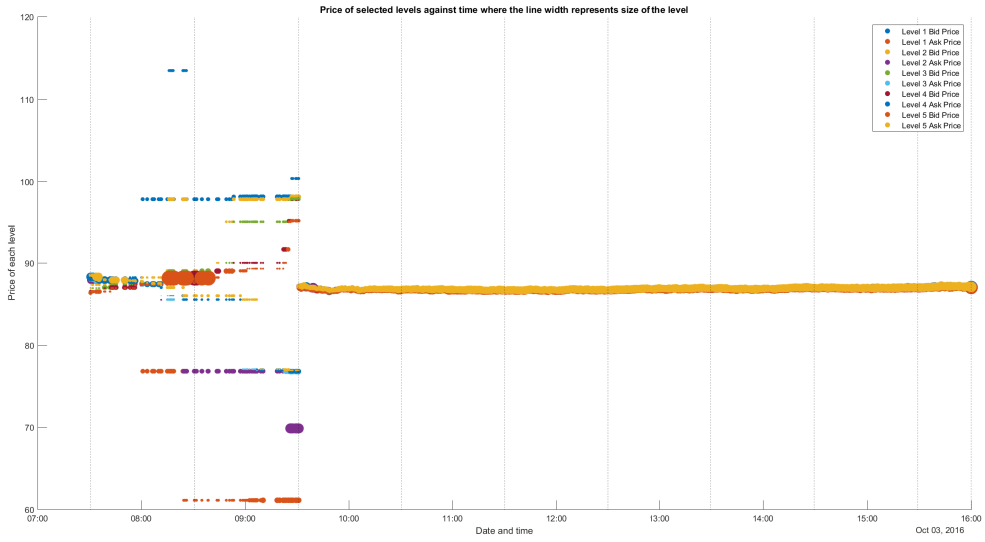


Figure 1

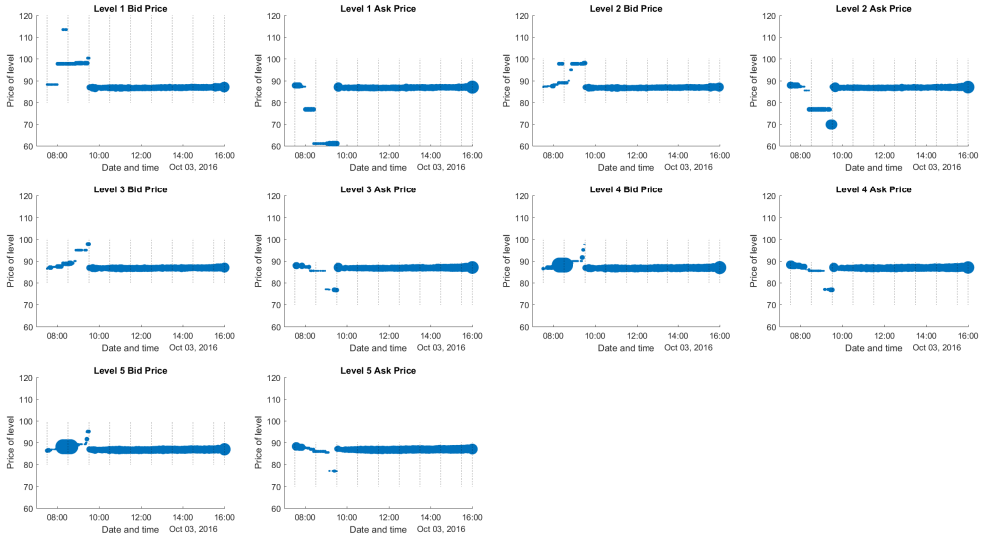


Figure 2

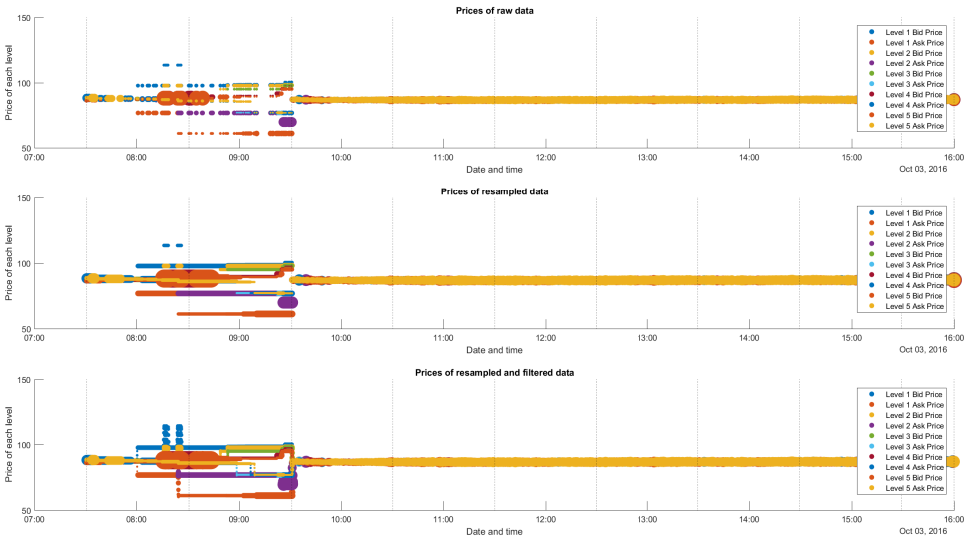


Figure 3

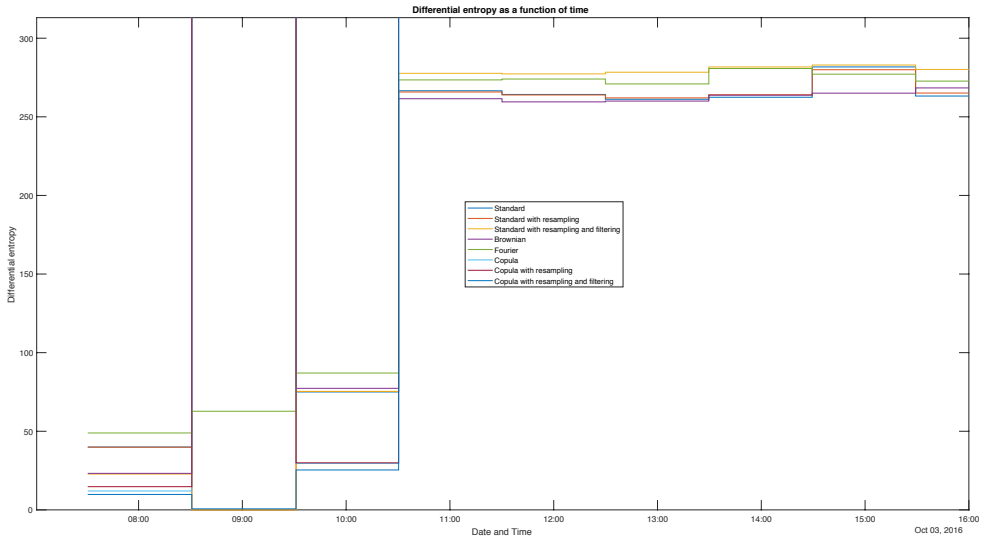


Figure 4

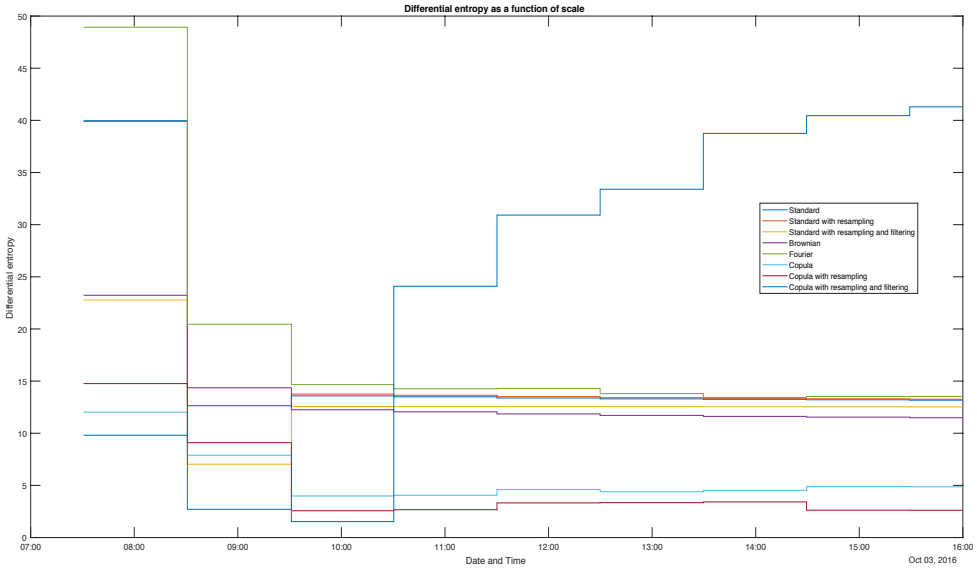


Figure 5

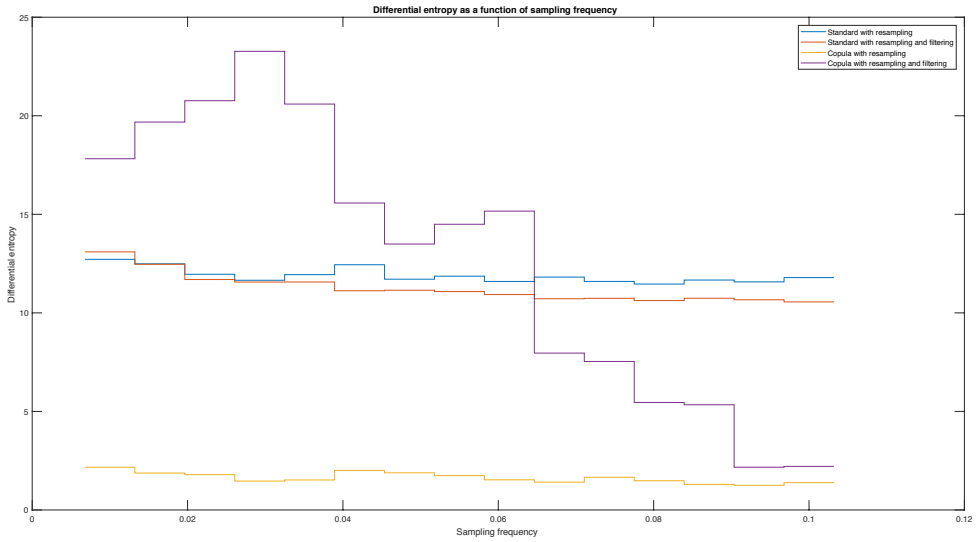


Figure 6

## References

[1] “On covariance estimation of non-synchronously observed diffusion processes” – Takaki Hayashi and Nakahiro Yoshida
[2] “Fourier transform method for nonparametric estimation of multivariate volatility” – Paul Malliavin and Maria Elvira Mancino
[3] “Fourier series method for measurement of multivariate volatilities” - Paul Malliavin and Maria Elvira Mancino
[4] “Fourier method for the univariate and multivariate volatility in the presence if high frequency data” – Chanel Malherbe
[5] “Copulas: A personal view” – Paul Embrechts
[6] “Modelling dependencies: An Overview” – Martyn Dorey and Phil Joubert
[7] “A review of copula models for economic time series” – Andrew Patton
[8] “Limit order books’ – Martin Gould, Mason Porter, Stacy Williams, Mark McDonald, Daniel Fenn and Sam Howison.