Data Analysis for High-Frequency Trading

Assignment 2

Visualisation and Stylised Facts

Julian Albert
ALBJUL005

Department of Statistical Sciences
University of Cape Town
South Africa

August 30, 2019

Contents

Part I

Bar-data and Candle-stick Plots

In the first part of the assignment we are tasked with plotting the 10-minute and 1-minute volume-weighted average price (VWAP) and Mid-price/Micro-price data in the form of Open-High-Low-Close (OHLC) or candlesticks. To calculate this T minute bar and candlesticks we use the following

$$\text{VWAP}_{(t+1,t)} = \frac{\sum_{i=t}^{T} v_i p_i}{\sum_{i=t}^{T} v_i} \tag{1}$$

$$\text{MicroPrice}_{(t+1,t)} = \frac{\sum_{i=t}^{T} (a_i + b_i) m_i}{\sum_{i=t}^{T} (a_i + b_i)} \tag{2}$$

where t is the time in steps of 1 second, v_i is the volume at time i, p_i the price at time i, a_i and b_i are the bid and ask volumes at time i and m_i is the mid-price at time i. Note here we are defining the micro-price as the volume-weighted mid-price. Using Equation (1) we calculate out 1-minute and 10-minute bars which will be used for the rest of the analysis. In the proceeding sections "transaction" data refers to the prices (as bars) and VWAPs (points) whilst "quote" data refers to mid-prices (as bars) and micro-prices (as points). KOHL bars can be interpreted as show in Figure 1, where Green and Red bar are associated with closing prices being higher and lower than the opening price for the T minute bar respectively. We also plot a point at the VWAP from the T minute bar.

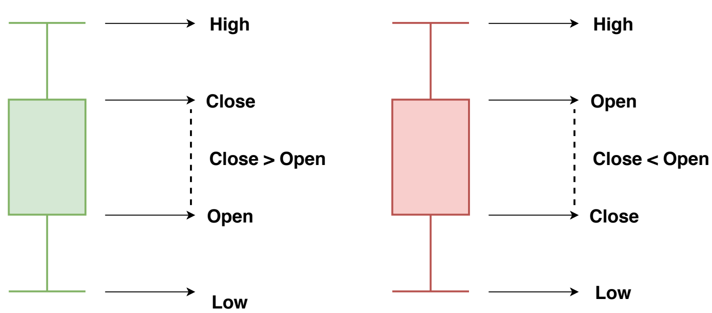


Figure 1: Breakdown of OHLC "candlestick" bars.

When analysing candlesticks we can define a bearish engulfing pattern as one which develops in an up trend when sellers outnumber buyers. This action is reflected by a long red candle engulfing a small green candle. The pattern indicates that sellers are back in control and that the price could continue to decline. An engulfing pattern on the bullish side of the market takes place when buyers outpace sellers. This is reflected in the chart by a long green candle engulfing a small red candle With bulls having established some control, the price could head higher.

Transaction Data

First we look at the 10-minute VWAP for AGL: Anglo American PLC in Figure 2. The top plot represents the OHLC data, whilst the bottom plot shows the volume for the 10-minute interval corresponding to the OHLC bar. We can see that the day opens with a big price shift downwards which is followed by a small up tick and then 20-minutes of decrease. Interesting here is that after about 2-hours we see 5 negative bars (Close < Open) followed by 3 positive bars (Close > Open). Overall for the day we can see that the bar seem almost "paired" indicating that there is some sort of persistence in the autocorrelation. That is up moves are followed by up moves and down moves are followed by down moves. We also see for the last 40 minutes of the day a big increase in the volume traded bringing the daily price close the the initial price at the start of the day. The large volume is also what drive the VWAP, where there are large volumes the VWAP shifts from the center of open close. I.e towards the end of the day we can see VWAP towards the closing, opening, opening, closing and closing prices of the 10-minute bar which may be indicative of an algorithm making decision at 10-minute intervals. We can also note the bearish engulfing pattern at around 11:15am which signals for increasing down moves.

Figure 2: Candlestick of OHLCV Transactions and Bar Graph for Volume - 10 Minute Bars.

Beyond the 10-minute VWAP we can analyse the 1-minute bars as illustrated in Figure 3. Here we see even more clearly the autocorrelation in price direction, however, of more interest is the sharp spike in volume that are clearly weighted down in the 10-minute interval at approximately 09:45am and 12:50pm. A more in-depth analysis is difficult as the plot is not very clear due to the VWAP points (the smallest point size in `GGPlot()` was used. There are however some blue OHLC candlesticks indicating that the price didn't close higher or lower than the opening prices in the 1-minute (referred to as a doji). These blue markers are seen at around 11:45am indicating what is known as a bearish harami cross (where an up candle is followed by a doji).The pattern shows indecision on the part of the buyers. If the price continues higher afterward, an up-trend may remain, but a down candle following this pattern indicates a further slide.

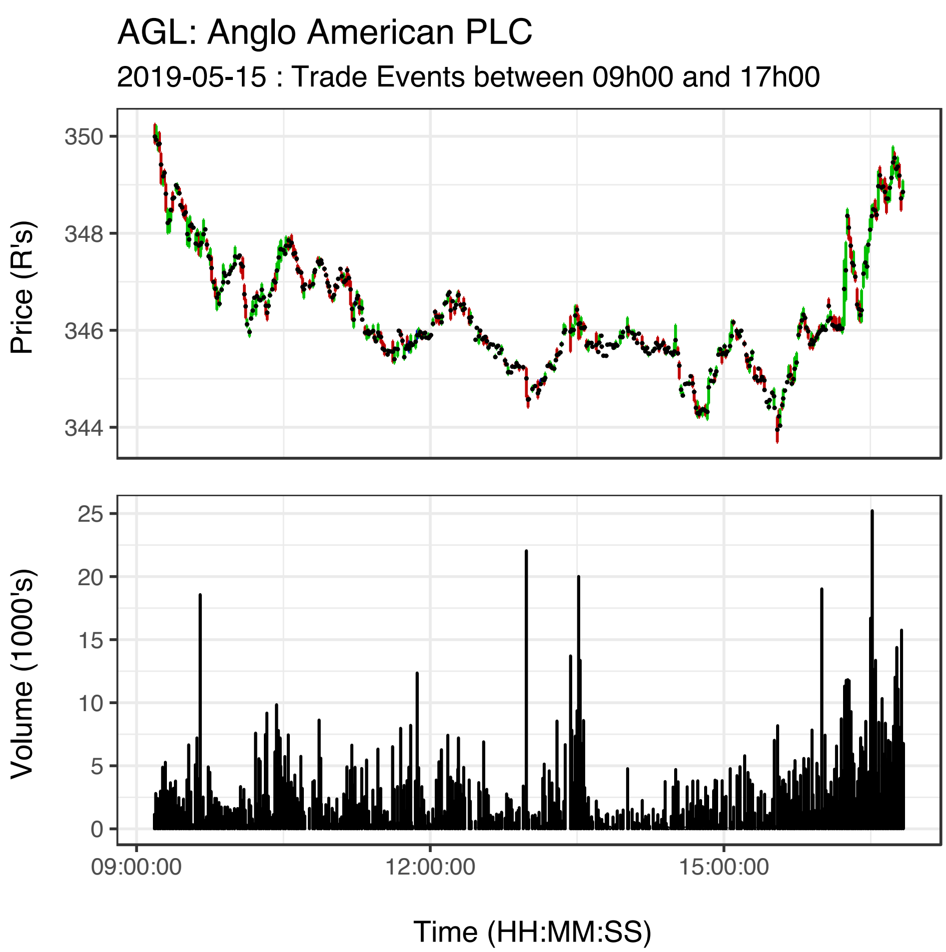


Figure 3: Candlestick of OHLCV Transactions and Bar Graph for Volume - 1 Minute Bars.

Quote Data

Next we can analyse the quote data, this is done for NPN: Naspers Ltd on a different day as analysing AGL: Anglo American PLC quotes on the same day would yield similar insights.

The first interesting insight to note is that in the 10-minute bars it would appear that the NPN quotes do not display many high/lows far beyond the opening and closing prices. This makes sense as we would expect the mid-prices to be a bit tighter than transaction prices. The volume for NPN is also consistently high. Again we see autocorrelation between the mid-price activities.

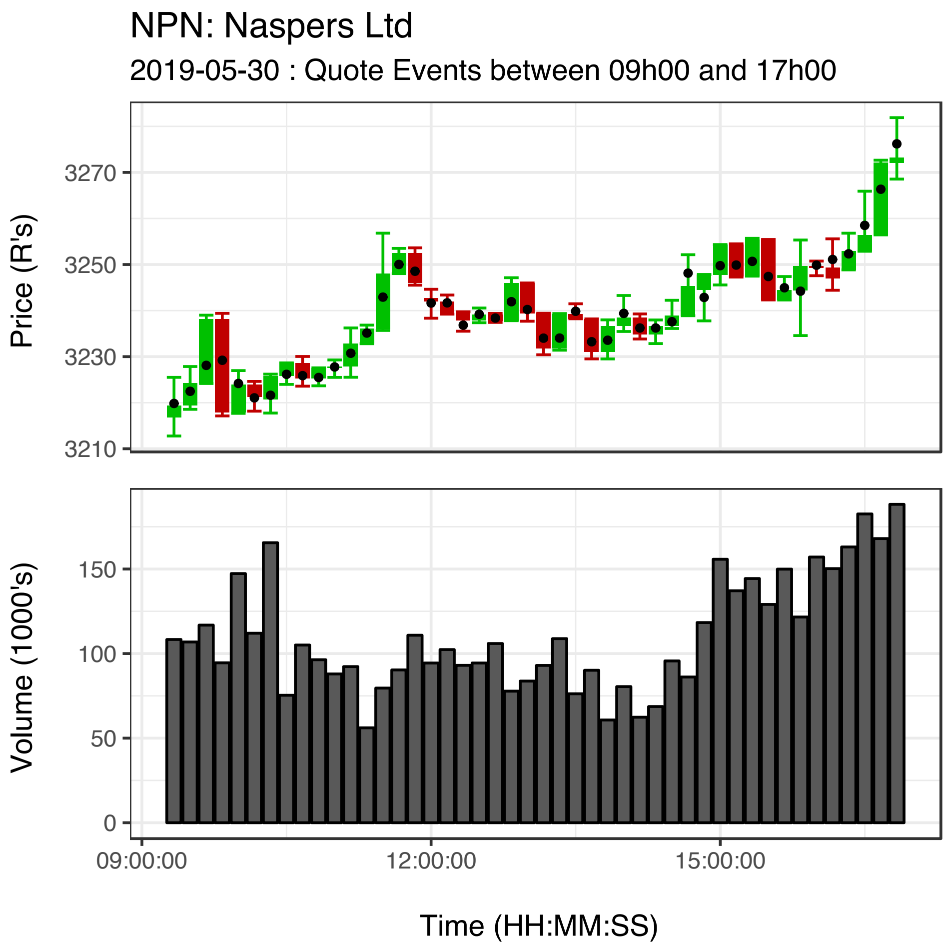


Figure 4: Candlestick of OHLCV Transactions and Bar Graph for Volume - 10 Minute Bars.

Once again for the 1-minute bar data we obviously observe the same trend, however, we see a really large volume on bid and offer at 16:46pm (this could again be an algorithm designed to kick in 15 minutes before auction). We also observe more frequent groups of upward move (green candles) than downward moves in the NPN mid-prices.

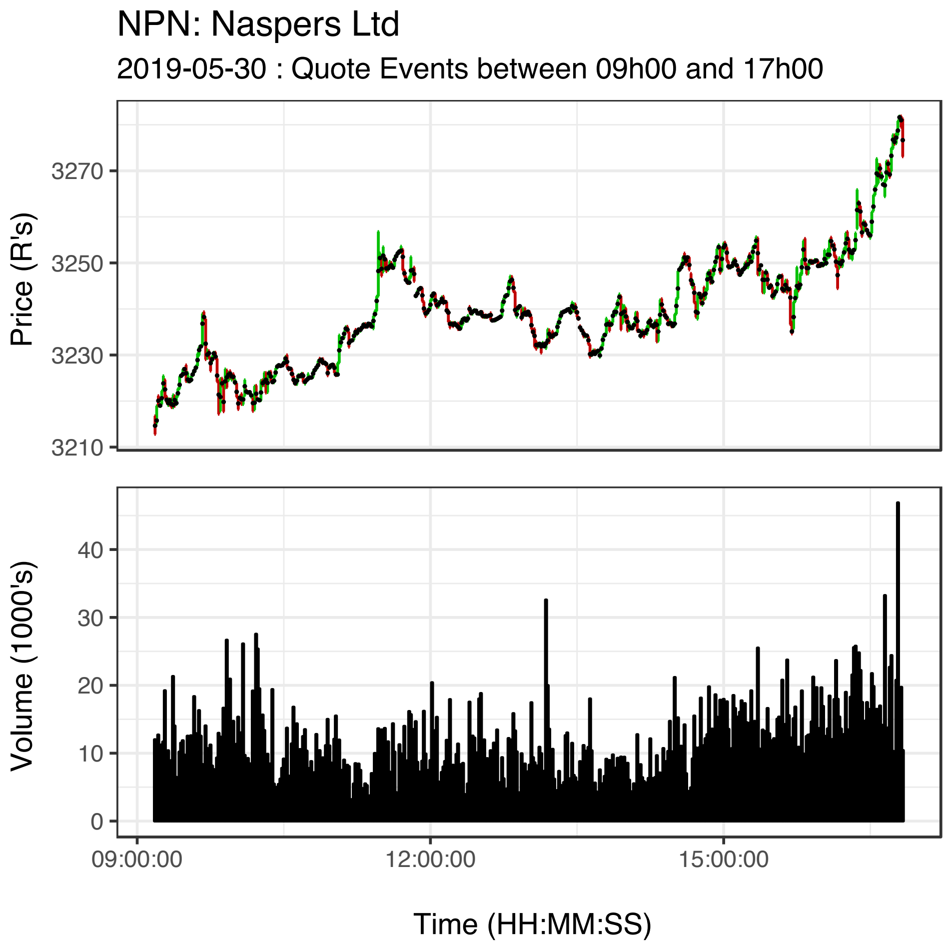


Figure 5: Candlestick of OHLCV Transactions and Bar Graph for Volume - 1 Minute Bars.

Time-series stylized facts

In this section we are interested in the sample distributions for the VWAPs and micro-prices (1-minute bars), in theory we would expect them to behave similarly as they are determined by similar market forces, however, there are more likely to be deviations in the transactions (VWAP) due to OTC transactions and private arrangements between two parties.

Frequency plots

The first step in understanding the sample distributions is plotting the frequencies for the 1-minute VWAPs. We can see from Figure 6 where the leftmost plot represents all of the VWAPs whilst the middle Figure is all the VWAPs excluding the abnormality of VWAP > 41000. When comparing this to the rightmost plot (micro-price frequencies) we see similar distributions, of course we cannot simply ignore the outlier but for the purposes of exploratory data analysis and our hypothesis about the similarities excluding the data point to validate our expectation seems fair.

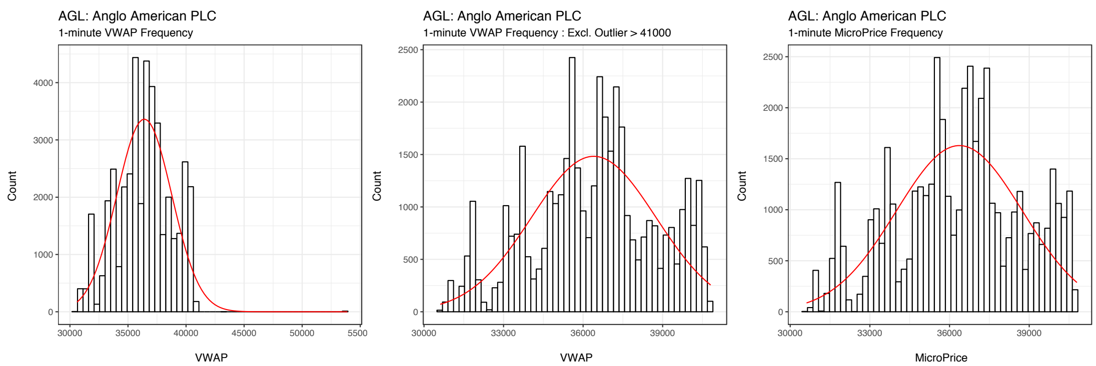
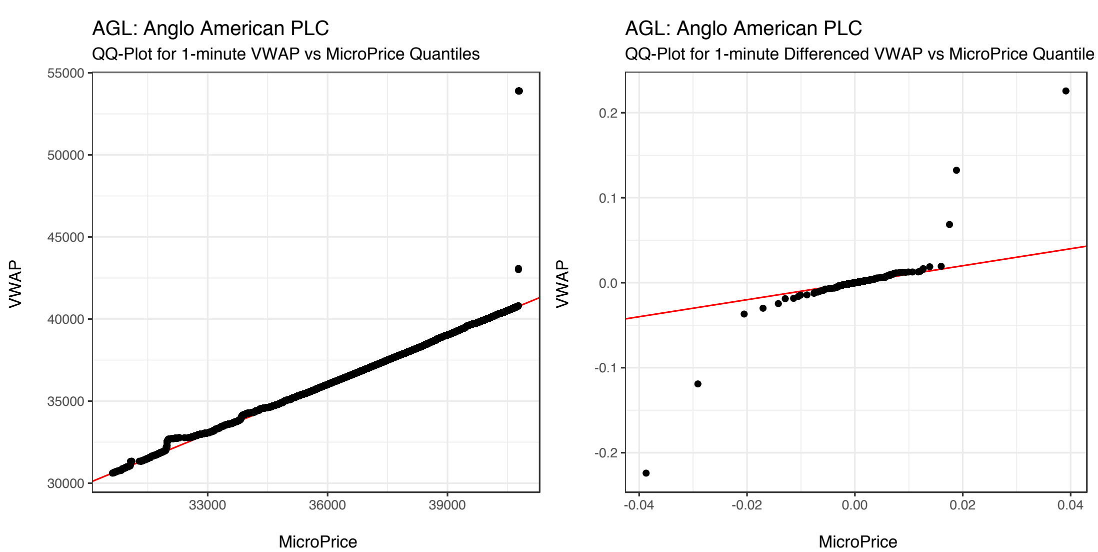


Figure 6: Frequency Plots for VWAP and MicroPrices - 1 Minute Bars.

QQ-Plots

The similarities between VWAPs and micro-prices are clearly seen by the QQ plot in Figure 7 where the quantiles for both statistics are incredibly correlated, we see that at high values the two diverge. To more easily observe the uncorrelated differences between VWAP and micro-price we can take the log-difference to "de-trend" the time series. We can see that VWAP has fatter tails than the micro-prices, this is expected as there is arguably more volatility in transactions to the quotes and - as said previously - the mid-prices should not exhibit as large movements as transaction prices.



(a) Trend

(b) De-trend

Figure 7: Quantile-Quantile Plot for VWAP vs MicroPrice - 1 Minute Bars.

VWAP Tail Distributions

Now we can examine the tails of the two statistics. First Figure 8 shows the lower (5%) and upper (95%) tails for the VWAP data. Most noteworthy is the fat upper tail due to the extreme event, other than this we can see that the upper tail is similar to a left skewed normal distribution. However, the lower tail shows a dip at a VWAP of 31250, the tail seems more fat than that of a normal distribution.

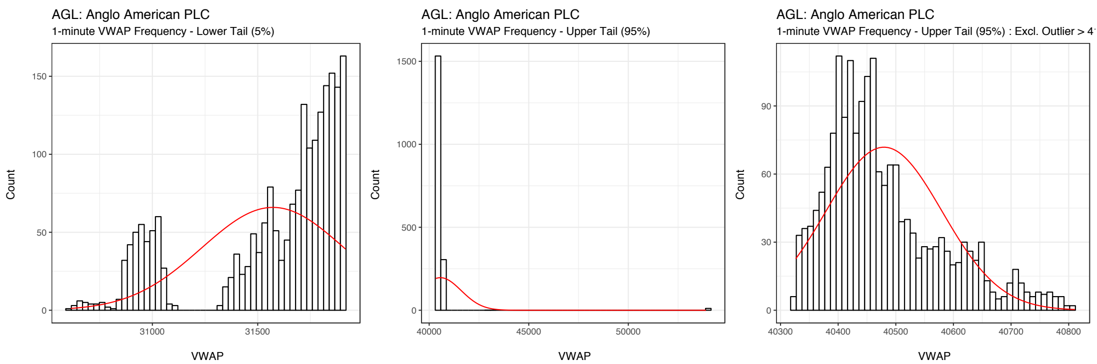


Figure 8: Tail Distributions (5% and 95%) VWAP - 1 Minute Bars.

MicroPrice Tail Distributions

Here the analysis is much the same as before, again this is expected as the two statistics are assumed highly correlated as they are both dependent on the same features from the TAQ data. In both instances we see that the data is empirically more leptokurtotic than the normal distribution.

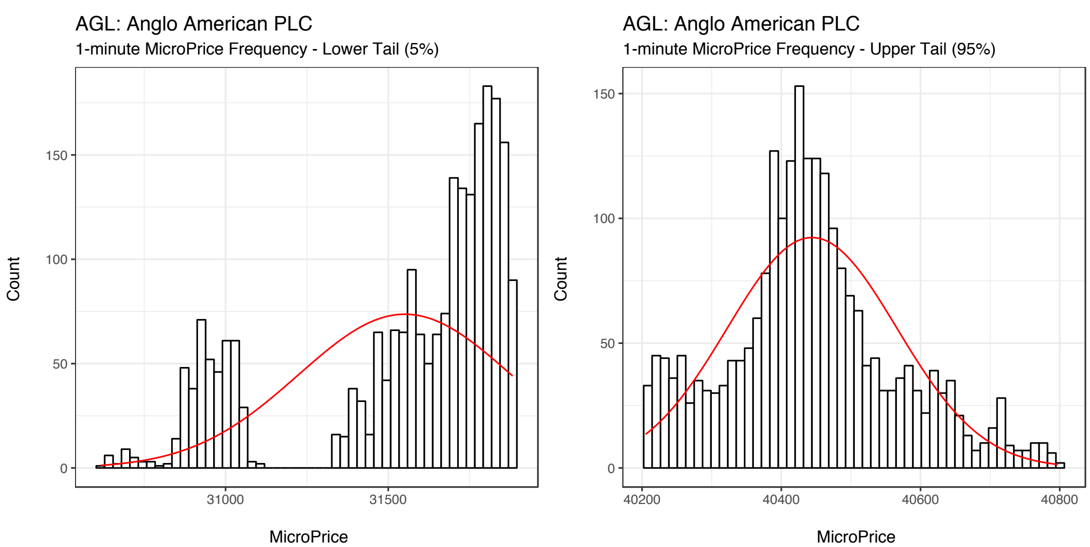
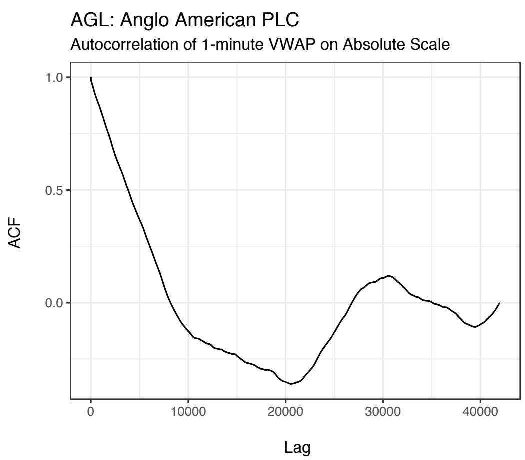


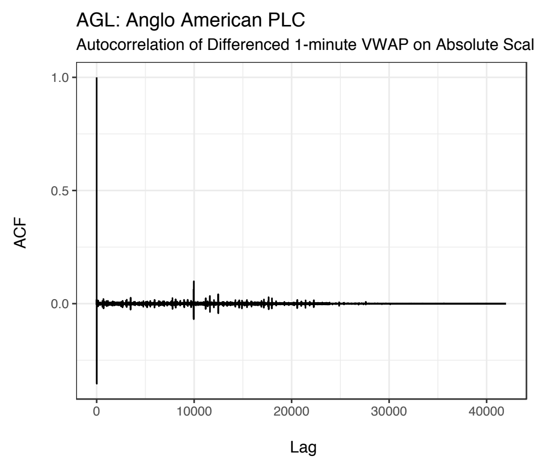
Figure 9: Tail Distributions (5% and 95%) MicroPrices - 1 Minute Bars.

Autocorrelations for VWAP and MicroPrice

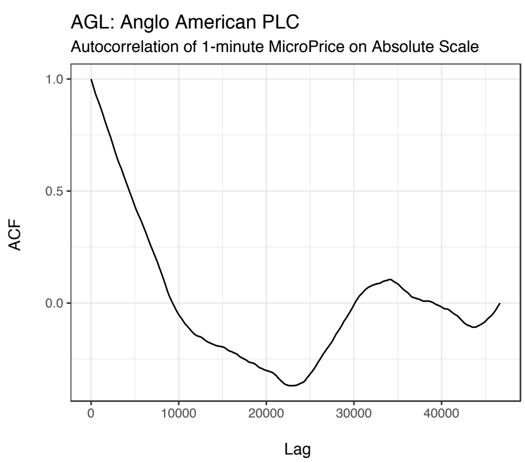
The ACF function indicates that if one observes a VWAP or micro-price now, based on this information alone there is some [non]vanishing predictability of the market time series. Both the top panels and lower panel in Figure 10 show similar behaviour where the autocorrelations die out relatively slow until lag 8000 before becoming negative and reverting upwards at approximately 20 000 where it proceeds to oscillate. This is a known characteristic of the autocorrelations of the prices - represented by VWAP - and hence the "homogeneous" micro-prices where the sinusoidal shape is indicative of seasonality. The log-differenced series in comparison dies out relatively fast, for VWAP we see a strong negative autocorrelation for small lags but then nothing too significant. Overall from the de-trended series we can see that the autocorrelations die out indicating that there is some vanishing predictability of the time series.



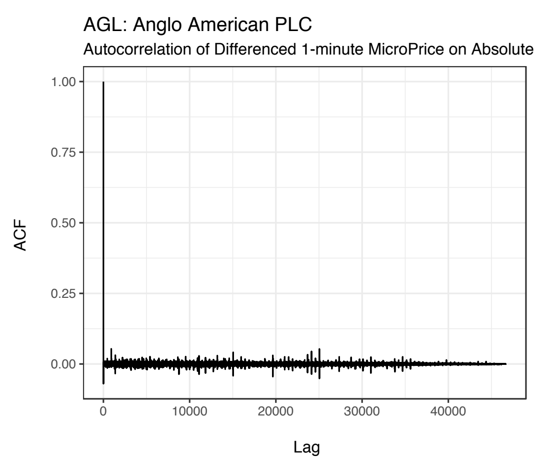
(a) Trend



(b) De-trend



(c) Trend



(d) De-trend

Figure 10: ACF Plots for Trended and De-trended VWAP and MicroPrices - 1 Minute Bars.

Part II

Price Impact

Price impact refers to the correlation between an incoming order and the subsequent price change [1]. A buyer initiated order should push the prices up while a seller initiated order should push the prices down. Big mid-price moves can be the result of fat-finger trades or the result of an order-book sweep when an aggressive buyer (seller) executes a large MO that walks the order book against multiple price levels on the ask (bid) - this can often be the result of an unintentional stop-loss or other mechanistic trading triggers. We plot the average price impact as

$$\Delta p = p(t_{k+1}) - p(t_k)$$

with average normalised volume

$$\omega^* = \frac{v_{ij}}{\bar{v}}$$

where $p(t_{k-1})$ and $p(t_k)$ are respectively the log-midquote price immediately prior and immediately after the trade event [2]. This change in the log-mid-price quantifies immediate price response to a trade of volume v_{ij}. The average normalised volume is ω^*.

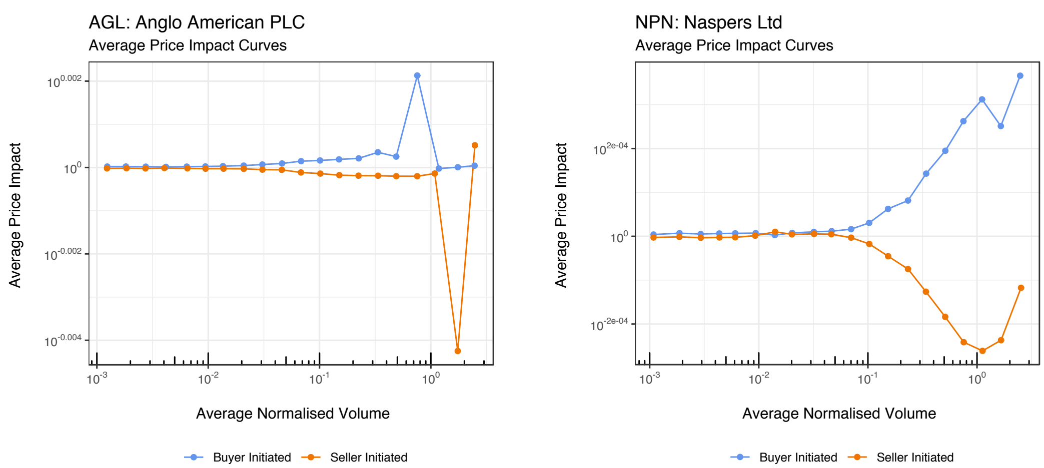


Figure 11: Price Impact Curves with Logarithmically Spaced Bins for $\omega^* \in [10^{-3}, 10^{0.5}]$ for the Period 2019-01-17 until 2019-06-14

From Figure 11 we see that for low volumes the price impact is not severe, however for AGL we see a large buyer-initiated price impact which is led by a large seller initiated price impact which could be due to a delayed feedback. In NPN we see similar behaviour for small volumes, however, the buyer and seller initiated price impacts diverge nearly symmetrically. As NPN is a more liquid stock we could be seeing that for less liquid stocks the market players react more asymmetrically whereas with NPN (liquid and large market capitalisation) both sides are matched.

Order-Book Seasonality

To analyse the order-book seasonality is to identify intraday trends that may be driving the order-book activities. To do this we consider three components

- Volume transacted
- Absolute Returns
- Spread

in all three cases we need to normalise our data such that it is comparable between days, we do this by using Equation (3)

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{3}$$

Volume

First we want to plot volume as a function of time across the day (intraday volume curves) as normalised by the daily volume. To do this we can compute and normalise by day to give us normalised daily volumes. We can then take volume traded at time t (grouped by minute) and aggregate across all days (i.e. take the 09:01:00 minute group of normalised data for all days and aggregate with a mean) to give us our average normalised trading volume for each time step. The result for both equities is illustrated in Figure 12.

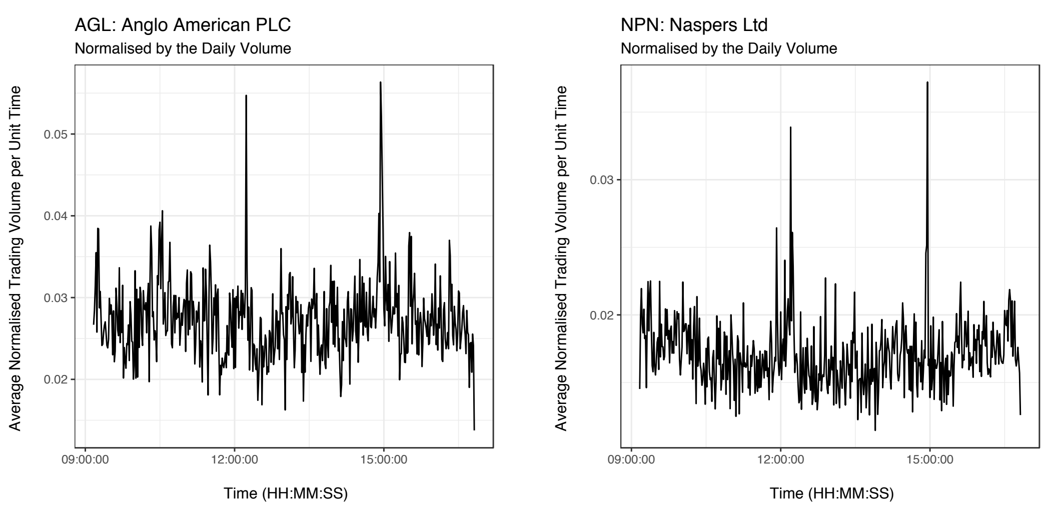


Figure 12: Intraday Volume Curve for the Period 2019-01-17 until 2019-06-14

Interestingly we see two definitive spikes at approximately 12:15pm and 14:55pm, the reason for these spikes is unclear as it appears in both equities and neither times having any prevalence to the HKE. A possibility is the $\approx$ 15:00pm impact may be due to American Markets opening, whilst 12:00pm may be manager making big trades before going to lunch.

Absolute Returns

Below we have defined the absolute return to be

$$r_t = |\log(P_{t+1}) - \log(P_t)|$$

We can see an expected result whereby the morning (opening) and afternoon (closing) periods are high with relative lows everywhere else. This is more intuitive than the volumes discussed previously as this is when managers tend to trade the most (not necessarily the highest volume but rather the activity for return yield is high). The result is a parabolic curve that is more "properly" shaped for NPN the more liquid of the two equities.

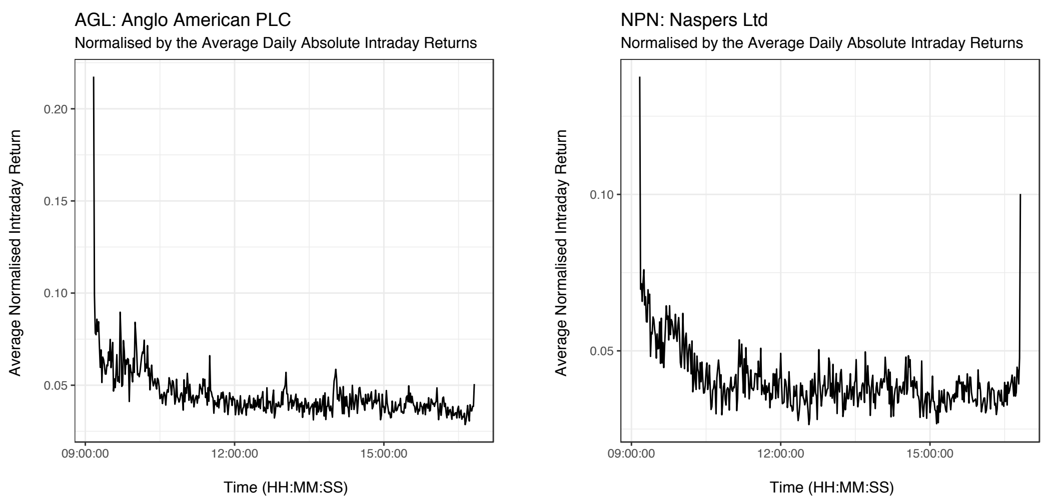


Figure 13: Intraday Normalised Absolute Log Returns for the Period 2019-01-17 until 2019-06-14

Spread

Although more subtle than the parabolic shape see in absolute returns, we have a "similar" behaviour where spreads are high at market open (close for NPN). Market makers likely force the spread to be wide in the beginning of the day when price are (arguably) at their most uncertain, as the day develops the spread will get tighter with an end of day spike hike due to volatility increases. Interestingly we see a more exponential behaviour for AGL compared to NPN which could be a function of market capitalisation. Where spread acts as a proxy for liquidity we can determine that mornings are relatively illiquid whilst for high-cap stocks end of day also displays low levels of liquidity that are preceded by "normal" liquidity.

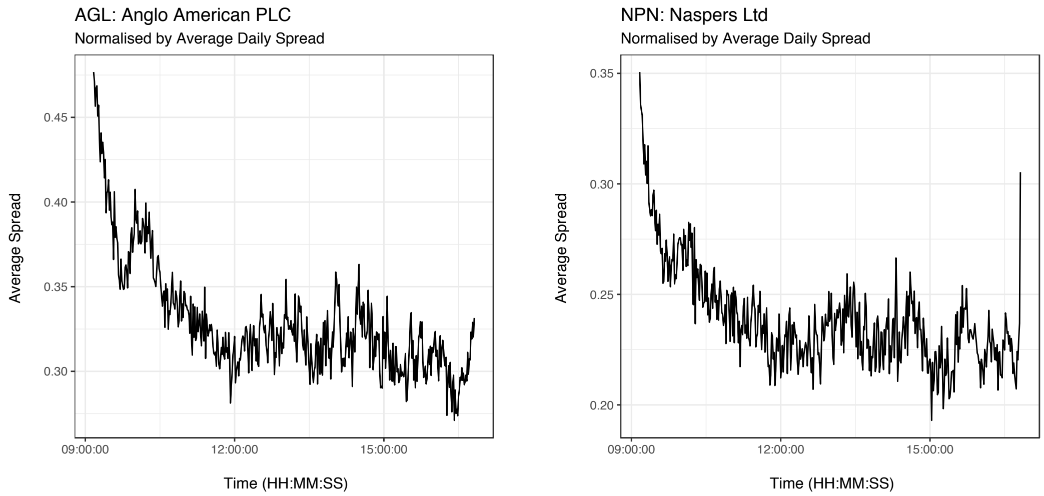


Figure 14: Intraday Spreads for the Period 2019-01-17 until 2019-06-14

- END -

References

[1] Bouchaud, J.-P., Farmer, J. D., and Lillo, F. How markets slowly digest changes in supply and demand. In *Handbook of financial markets: dynamics and evolution*. Elsevier, 2009, pp. 57–160.

[2] Harvey, M., Hendricks, D., Gebbie, T., and Wilcox, D. Deviations in expected price impact for small transaction volumes under fee restructuring. *Physica A: Statistical Mechanics and its Applications 471* (2017), 416–426.

Appendix

R Code

```r
##
#
# Author: Julian Albert
# Date: 20 August 2019
#
# Description:
# HFT Assignment 2 - Basically want to work with TAQ data and perform EDA to
# better understand the data and the trading environment.
#
#------------------------------------------------------------------------------#

# 0. Clean Workspace, Directory and Library ----

## Clean Workspace
rm(list=ls())
dev.off() # Close Plots
setwd("~") # Clear Path to User

## Locations
project_folder <- "/Documents/UCT/Coursework/HFT"
loc_script <- "/Assignment_2/UCT_Assignment/Code"
loc_figs <- "/Assignment_2/UCT_Assignment/Figs"
loc_data <- "/Data"

## Directories
dir_script <- paste("~", project_folder, loc_script, sep = '')
dir_figs <- paste("~", project_folder, loc_figs, sep = '')
dir_data <- paste("~", project_folder, loc_data, sep = '')

## Filenames
filen_dat <- "/dat_TAQ.rds"

## Set Working Directory to Script Location
setwd(dir_script)

## Libraries - Lazyload
if (!require("pacman")) install.packages("pacman")
p_load(tidyverse, lubridate, zoo, data.table, Cairo, viridis,
       quantmod, grid, scales, pracma)

## Pull Clean Data
dat_TAQ <- readRDS(file = paste(dir_data, filen_dat, sep = ""))

## Options
options(digits.secs = 3)

# 1. Variables Needed ----

DayStartTime <- "09:00:00"
DayEndTime <- "17:00:00"
timezone <- "Africa/Johannesburg"

Stock1 <- "AGL: Anglo American PLC"
Stock2 <- "NPN: Naspers Ltd"

# 2. Functions ----

## Function to draw candlestick plots
func.draw_candles <- function(df, title_param, subtitle_param, alpha_param = 1)
{

  df$change <- ifelse(df$Close > df$Open, "up",
                      ifelse(df$Close < df$Open, "down", "flat"))

  # So let us instead find delta (seconds) between 1st and 2nd row and just
  # use it for all other rows. We check 1st 3 rows to avoid larger "weekend gaps"
```

```
  width_candidates <- c(as.numeric(difftime(df$DateTimeL[2], df$DateTimeL[1]), units = "secs"),
                        as.numeric(difftime(df$DateTimeL[3], df$DateTimeL[2]), units = "secs"),
                        as.numeric(difftime(df$DateTimeL[4], df$DateTimeL[3]), units = "secs"))

  df$width_s = min(width_candidates)
  # define the vector of candle colours either by name or by rgb()
  #candle_colors = c("down" = "red", "up" = "green", "flat" = "blue")
  candle_colors = c("down" = rgb(192 , 0, 0, alpha = 255, maxColorValue = 255),
                    "up" = rgb(0, 192, 0, alpha = 255, maxColorValue = 255),
                    "flat" = rgb(0, 0, 192, alpha = 255, maxColorValue = 255))

  # Candle chart:
  g <- ggplot(df, aes(x = DateTimeL)) +
    geom_errorbar(aes(ymin = Low/100, ymax = High/100, colour = change),
                  alpha = alpha_param, width = (df$width_s * 0.9)) +
    theme_bw() +
    labs(title = title_param, subtitle = subtitle_param,
         x = "\n Time (HH:MM:SS)", y = "Price (R's) \n") +
    scale_x_datetime(date_labels = "%H:%M:%S",
                     timezone = "Africa/Johannesburg") +
    geom_rect(aes(xmin = DateTimeL - width_s/2 * 0.9,
                  xmax = DateTimeL + width_s/2 * 0.9,
                  ymin = pmin(Open/100, Close/100),
                  ymax = pmax(Open/100, Close/100),
                  fill = change), alpha = alpha_param) + # candle recatngle
    guides(fill = FALSE, colour = FALSE) +
    scale_color_manual(values = candle_colors) + # color for line
    scale_fill_manual(values = candle_colors) # color for candle fill

  # Handle special cases: flat bar and Open == close:
  if (any(df$change == "flat"))
  {
    g <- g + geom_segment(data = df[df$change == "flat",],
                          aes(x = DateTimeL - width_s / 2 * 0.9,
                              y = Close/100, yend = Close/100,
                              xend = DateTimeL + width_s / 2 * 0.9,
                              colour = change),
                          alpha = alpha_param)
  }

  return(g)

}

## Function to subset data for plotting
func.plot_subset <- function(Data, Date_for_plot)
{

  ## Define the Day
  tmp_ymd <- paste(year(Date_for_plot),
                   month(Date_for_plot),
                   day(Date_for_plot), sep = "-")

  ## Need to subset for the Trading Hours
  date_start <- as.POSIXct(paste(tmp_ymd, DayStartTime, sep = " "), tz = timezone)
  date_end <- as.POSIXct(paste(tmp_ymd, DayEndTime, sep = " "), tz = timezone)
  plot_interval <- interval(date_start, date_end)

  dat_hourly <- Data[Data$DateTimeL %within% plot_interval, ]
  return(dat_hourly)
}


## Function to calc by groups and return candlestick plots
func.by_interval_calcs <- function(TAQ_data_1_equity, # Take in data for one equity
         interval_numeric = 10,
         interval_time_unit = "min", # time for intervals i.e 1, "min"
         DayStartTime = "09:00:00",
         DayEndTime = "17:00:00",
         Type = "Trade",
```

```r
        Stock_name,
        Date_for_plot = Date_for_plot)
{

  ## Specify our time interval to group data by
  interval <- paste(interval_numeric, interval_time_unit, sep = " ")
  ## Group Data by time interval >> initialise OHLC columns
  dat_interval <- TAQ_data_1_equity %>%
    group_by(Int = cut(DateTimeL, interval)) %>%
    ungroup() %>%
    mutate(Open = NA, High = NA, Low = NA, Close = NA)

  ## We are going to want the plots for both transaction data and for
  ## Quote data, make function that deals with this
  if(Type == "Quote"){

  ## If we want quote data: initialise trade data columns for merge
  tmp.interval_OHLCV <- dat_interval %>%
    filter(Type != "Quote") %>%
    mutate(Tot_Volume_OHLCV = NA, MicroPrice_OHLCV = NA)
  ## Calculate required measures for the Quote data by interval
  tmp.interval_OHLCV_calcs <- dat_interval %>%
    filter(Type == "Quote") %>%
    group_by(Int) %>%
    mutate(Open = first(na.omit(MidPrice)),
           High = max(na.omit(MidPrice)),
           Low = min(na.omit(MidPrice)),
           Close = last(na.omit(MidPrice)),
           Tot_Volume_OHLCV = Volume.Bid + Volume.Ask,
           MicroPrice_OHLCV = weighted.mean(MidPrice, Tot_Volume_OHLCV)) %>%
    ungroup()

  ## Bind Trades and Quotes together to give us calcs. by interval
  df <- bind_rows(tmp.interval_OHLCV, tmp.interval_OHLCV_calcs) %>%
    arrange(DateTimeL)

  ## For the Plot we can Seperate data into Quotes only >> subset for a day
  dat.plot_candles <- df %>%
    filter(Type == "Quote") %>%
    func.plot_subset(Date_for_plot)

  } else{

  ## If we want trade data: initialise quote data columns for merge
  tmp.interval_OHLCV <- dat_interval %>%
    filter(Type != "Trade") %>%
    mutate(Volume_OHLCV = NA, VWAP = NA)

  ## Calculate required measures for the Trade data by interval
  tmp.interval_OHLCV_calcs <- dat_interval %>%
    filter(Type == "Trade") %>%
    group_by(Int) %>%
    mutate(Open = first(Price),
           High = max(Price),
           Low = min(Price),
           Close = last(Price),
           Volume_OHLCV = sum(Volume.Trade),
           VWAP = weighted.mean(Price, Volume.Trade)) %>%
    ungroup()

  ## Bind Trades and Quotes together to give us calcs. by interval
  df <- bind_rows(tmp.interval_OHLCV, tmp.interval_OHLCV_calcs) %>%
    arrange(DateTimeL)

  ## For the Plot we can Seperate data into Trades only >> subset for a day
  dat.plot_candles <- df %>%
    filter(Type == "Trade") %>%
    func.plot_subset(Date_for_plot)

  }

  ## Generic Titles
  time.title <- paste(Type, " Events between ",
```

```r
                        unlist(strsplit(DayStartTime, ":"))[1],
                        "h", unlist(strsplit(DayStartTime, ":"))[2], " and ",
                        unlist(strsplit(DayEndTime, ":"))[1],
                        "h", unlist(strsplit(DayEndTime, ":"))[2], sep = "")

  date.title <- date(dat.plot_candles$DateTimeL[1])

  plot_subtitle <- paste(date.title, time.title, sep = " : ")

  ## Plot
  candle_plot <- dat.plot_candles %>%
    group_by(Int) %>%
    slice(n()) %>%
    func.draw_candles(title_param = Stock_name,
                        subtitle_param = plot_subtitle)

  return(list(Data_used = df,
              Plot = candle_plot))

}

## Function to get micro-prices and VWAP as points for plot
func.plot_other <- function(Data, Type = "Trade", Date_for_plot)
{

  if(Type == "Trade"){

    ## For trade
    point_data <- Data %>%
      filter(Type == "Trade",
             date(DateTimeL) == date(Date_for_plot)) %>%
      group_by(Int) %>%
      slice(n())

    points <- point_data$VWAP/100

    hist_data <- Data %>%
      filter(Type == "Trade",
             date(DateTimeL) == date(Date_for_plot)) %>%
      group_by(Int) %>%
      slice(n())

  } else{

    ## For trade
    point_data <- Data %>%
      filter(Type == "Quote",
             date(DateTimeL) == date(Date_for_plot)) %>%
      group_by(Int) %>%
      slice(n())

    points <- point_data$MicroPrice_OHLCV/100

    hist_data <- Data %>%
      filter(Type == "Quote",
             date(DateTimeL) == date(Date_for_plot)) %>%
      group_by(Int) %>%
      mutate(Volume_OHLCV = sum(Tot_Volume_OHLCV)) %>%
      slice(n())

  }

  return(list(Points = points,
              HistData = hist_data))

}

## Final Plot for candlesticks and histogram
func.final_plot <- function(dat_event_by_interval,
                            points_event,
                            hist_event,
                            point_size = 1)
{
```

```r

  plot_candles <- dat_event_by_interval$Plot +
    geom_point(aes(y = points_event), size = point_size) +
    labs(x = "", y = "Price (R's) \n") +
    theme(axis.title.x=element_blank(),
          axis.text.x=element_blank(),
          axis.ticks.x=element_blank())

  plot_hist <- ggplot(hist_event, aes(DateTimeL, Volume_OHLCV/1000)) +
    geom_bar(stat = "identity", color = "black")  +
    theme_bw() +
    labs(x = "\n Time (HH:MM:SS)", y = "Volume (1000's) \n") +
    scale_x_datetime(date_labels = "%H:%M:%S",
                     timezone = "Africa/Johannesburg")

  grid.newpage()
  grid.draw(rbind(ggplotGrob(plot_candles),
                  ggplotGrob(plot_hist), size = "last"))

}

## Function to specify scientific notation 10^# for plots
func.fancy_scientific_labels <- function(l)
{
  index_zero <- which(l == 0)
  # turn in to character string in scientific notation
  l <- format(l, scientific = TRUE)
  # quote the part before the exponent to keep all the digits
  l <- gsub("^(.*)e", "'\\1'e", l)
  # turn the 'e+' into plotmath format
  l <- gsub("e", "%*%10^", l)
  # return this as an expression
  l[index_zero] <- "0"
  parse(text=l)
}

# i.e. ggplot() + scale_y_continuous(labels = func.fancy_scientific_labels)

# 3. Plot Candlesticks for AGL 10 minutes ----

idx <- ceiling(runif(1, 0, NROW(dat_TAQ$AGL$DateTimeL)))
Date_for_plot <- dat_TAQ$AGL$DateTimeL[idx]

## Plots for the Transactions
dat_AGL_10min_trades <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$AGL,
                                               interval_numeric = 10,
                                               Type = "Trade",
                                               Stock_name = Stock1,
                                               Date_for_plot = Date_for_plot)

dat_AGL_10min_trades_other <- func.plot_other(dat_AGL_10min_trades$Data_used,
                                              Type = "Trade",
                                              Date_for_plot = Date_for_plot)

dat_AGL_1min_trades <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$AGL,
                                              interval_numeric = 1,
                                              Type = "Trade",
                                              Stock_name = Stock1,
                                              Date_for_plot = Date_for_plot)

dat_AGL_1min_trades_other <- func.plot_other(dat_AGL_1min_trades$Data_used,
                                             Type = "Trade",
                                             Date_for_plot = Date_for_plot)

setwd(dir_figs)
cairo_pdf("HFT_Ass2_fig_AGL10min_trades.pdf", height = 5, width = 5)
func.final_plot(dat_AGL_10min_trades,
                dat_AGL_10min_trades_other$Points,
                dat_AGL_10min_trades_other$HistData)
dev.off()

cairo_pdf("HFT_Ass2_fig_AGL1min_trades.pdf", height = 5, width = 5)
func.final_plot(dat_AGL_1min_trades,
```

```
                 dat_AGL_1min_trades_other$Points,
                 dat_AGL_1min_trades_other$HistData,
                 point_size = 0.1)
dev.off()
setwd(dir_script)

## Plots for the Quotes
dat_AGL_10min_quotes <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$AGL,
                                                interval_numeric = 10,
                                                Type = "Quote",
                                                Stock_name = Stock1,
                                                Date_for_plot = Date_for_plot)

dat_AGL_10min_quotes_other <- func.plot_other(dat_AGL_10min_quotes$Data_used,
                                               Type = "Quote",
                                               Date_for_plot = Date_for_plot)

dat_AGL_1min_quotes <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$AGL,
                                               interval_numeric = 1,
                                               Type = "Quote",
                                               Stock_name = Stock1,
                                               Date_for_plot = Date_for_plot)

dat_AGL_1min_quotes_other <- func.plot_other(dat_AGL_1min_quotes$Data_used,
                                              Type = "Quote",
                                              Date_for_plot = Date_for_plot)

setwd(dir_figs)
cairo_pdf("HFT_Ass2_fig_AGL10min_quotes.pdf", height = 5, width = 5)
func.final_plot(dat_AGL_10min_quotes,
                 dat_AGL_10min_quotes_other$Points,
                 dat_AGL_10min_quotes_other$HistData)
dev.off()

cairo_pdf("HFT_Ass2_fig_AGL1min_quotes.pdf", height = 5, width = 5)
func.final_plot(dat_AGL_1min_quotes,
                 dat_AGL_1min_quotes_other$Points,
                 dat_AGL_1min_quotes_other$HistData,
                 point_size = 0.1)
dev.off()
setwd(dir_script)

# 4. Plot Candlesticks for NPN  10 minutes  ----

idx <- ceiling(runif(1, 0, NROW(dat_TAQ$NPN$DateTimeL)))
Date_for_plot <- dat_TAQ$NPN$DateTimeL[idx]

## Plots for the Transactions
dat_NPN_10min_trades <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$NPN,
                                                interval_numeric = 10,
                                                Type = "Trade",
                                                Stock_name = Stock2,
                                                Date_for_plot = Date_for_plot)

dat_NPN_10min_trades_other <- func.plot_other(dat_NPN_10min_trades$Data_used,
                                               Type = "Trade",
                                               Date_for_plot = Date_for_plot)

dat_NPN_1min_trades <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$NPN,
                                               interval_numeric = 1,
                                               Type = "Trade",
                                               Stock_name = Stock2,
                                               Date_for_plot = Date_for_plot)

dat_NPN_1min_trades_other <- func.plot_other(dat_NPN_1min_trades$Data_used,
                                              Type = "Trade",
                                              Date_for_plot = Date_for_plot)

setwd(dir_figs)
cairo_pdf("HFT_Ass2_fig_NPN10min_trades.pdf", height = 5, width = 5)
func.final_plot(dat_NPN_10min_trades,
                 dat_NPN_10min_trades_other$Points,
                 dat_NPN_10min_trades_other$HistData)
```

```r
dev.off()

cairo_pdf("HFT_Ass2_fig_NPN1min_trades.pdf", height = 5, width = 5)
func.final_plot(dat_NPN_1min_trades,
                dat_NPN_1min_trades_other$Points,
                dat_NPN_1min_trades_other$HistData,
                point_size = 0.1)
dev.off()
setwd(dir_script)

## Plots for the Quotes
dat_NPN_10min_quotes <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$NPN,
                                               interval_numeric = 10,
                                               Type = "Quote",
                                               Stock_name = Stock2,
                                               Date_for_plot = Date_for_plot)

dat_NPN_10min_quotes_other <- func.plot_other(dat_NPN_10min_quotes$Data_used,
                                              Type = "Quote",
                                              Date_for_plot = Date_for_plot)

dat_NPN_1min_quotes <- func.by_interval_calcs(TAQ_data_1_equity = dat_TAQ$NPN,
                                              interval_numeric = 1,
                                              Type = "Quote",
                                              Stock_name = Stock2,
                                              Date_for_plot = Date_for_plot)

dat_NPN_1min_quotes_other <- func.plot_other(dat_NPN_1min_quotes$Data_used,
                                             Type = "Quote",
                                             Date_for_plot = Date_for_plot)

setwd(dir_figs)
cairo_pdf("HFT_Ass2_fig_NPN10min_quotes.pdf", height = 5, width = 5)
func.final_plot(dat_NPN_10min_quotes,
                dat_NPN_10min_quotes_other$Points,
                dat_NPN_10min_quotes_other$HistData)
dev.off()

cairo_pdf("HFT_Ass2_fig_NPN1min_quotes.pdf", height = 5, width = 5)
func.final_plot(dat_NPN_1min_quotes,
                dat_NPN_1min_quotes_other$Points,
                dat_NPN_1min_quotes_other$HistData,
                point_size = 0.1)
dev.off()
setwd(dir_script)

# 5. Time Series Stylised facts ----

func.freq_plot <- function(data)
{

    n <- length(data)
    mean <- mean(data)
    sd <- sd(data)
    bins <- seq(min(data), max(data), length = 50)
    binwidth <- diff(bins)[1]

    ggplot(as.data.frame(data),
           aes(x = data, mean = mean, sd = sd, binwidth = binwidth, n = n)) +
      geom_histogram(binwidth = binwidth, colour = "black", fill = "white") +
      stat_function(fun = function(x) dnorm(x, mean = mean, sd = sd)*n*binwidth,
                    color = "red", size = 0.5) +
      theme_bw()
}

## Frequency plot >> VWAP
AGL_freq_trades <- dat_AGL_1min_trades$Data_used %>%
  filter(Type == "Trade") %>%
  group_by(Int) %>%
  slice(n())

setwd(dir_figs)
cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_Freq.pdf", width = 5, height = 5)
```

```r
func.freq_plot(AGL_freq_trades$VWAP) +
  labs(x = "\n VWAP", y = "Count \n", title = Stock1,
       subtitle = "1-minute VWAP Frequency")
dev.off()

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_Freq_lt41000.pdf", width = 5, height = 5)
func.freq_plot(AGL_freq_trades$VWAP[AGL_freq_trades$VWAP < 41000]) +
  labs(x = "\n VWAP", y = "Count \n", title = Stock1,
       subtitle = "1-minute VWAP Frequency : Excl. Outlier > 41000")
dev.off()

## Frequency plot >> Micro-Price
AGL_freq_quotes <- dat_AGL_1min_quotes$Data_used %>%
  filter(Type == "Quote" & !is.na(MicroPrice_OHLCV)) %>%
  group_by(Int) %>%
  slice(n())

cairo_pdf("HFT_Ass2_fig_AGL1min_MicroPrice_Freq.pdf", width = 5, height = 5)
func.freq_plot(AGL_freq_quotes$MicroPrice_OHLCV) +
  labs(x = "\n MicroPrice", y = "Count \n", title = Stock1,
       subtitle = "1-minute MicroPrice Frequency")
dev.off()

## QQ plot does this so we use it for GGplot because different lengths
sx <- sort(AGL_freq_quotes$MicroPrice_OHLCV)
sy <- sort(AGL_freq_trades$VWAP)
lenx <- length(sx)
leny <- length(sy)
if (leny < lenx)sx <- approx(1L:lenx, sx, n = leny)$y
if (leny > lenx)sy <- approx(1L:leny, sy, n = lenx)$y

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_MicroPrice_QQ.pdf", width = 5, height = 5)
ggplot() +
  geom_abline(intercept = 0, slope = 1, col = "red") +
  geom_point(aes(x=sx, y=sy)) +
  theme_bw() +
  labs(y = "VWAP \n", x = "\n MicroPrice", title = Stock1,
       subtitle = "QQ-Plot for 1-minute VWAP vs MicroPrice Quantiles") ## Highly correlated
dev.off()

## Detrend
sx <- sort(diff(log(AGL_freq_quotes$MicroPrice_OHLCV)))
sy <- sort(diff(log(AGL_freq_trades$VWAP)))
lenx <- length(sx)
leny <- length(sy)
if (leny < lenx)sx <- approx(1L:lenx, sx, n = leny)$y
if (leny > lenx)sy <- approx(1L:leny, sy, n = lenx)$y

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_MicroPrice_QQ_DiffLog.pdf", width = 5, height = 5)
ggplot() +
  geom_abline(intercept = 0, slope = 1, col = "red") +
  geom_point(aes(x=sx, y=sy)) +
  theme_bw() +
  labs(y = "VWAP \n", x = "\n MicroPrice", title = Stock1,
       subtitle = "QQ-Plot for 1-minute Differenced VWAP vs MicroPrice Quantiles") ## MP
            fatter tails
dev.off()

## Tails >> VWAP

### Lower
qlower <- quantile((AGL_freq_trades$VWAP), 0.05)
sample_qlower <- (AGL_freq_trades$VWAP)[(AGL_freq_trades$VWAP) <= qlower]

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_Freq_TailLower.pdf", width = 5, height = 5)
func.freq_plot(sample_qlower)  +
  labs(x = "\n VWAP", y = "Count \n", title = Stock1,
       subtitle = "1-minute VWAP Frequency - Lower Tail (5%)")
dev.off()

### Upper
qupper <- quantile((AGL_freq_trades$VWAP), 0.95)
sample_qupper <- (AGL_freq_trades$VWAP)[(AGL_freq_trades$VWAP) >= qu]
```

```r

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_Freq_TailUpper.pdf", width = 5, height = 5)
func.freq_plot(sample_qupper) +
  labs(x = "\n VWAP", y = "Count \n", title = Stock1,
       subtitle = "1-minute VWAP Frequency - Upper Tail (95%)")
dev.off()

length(which(sample_qupper > 41000))/length(sample_qupper)
cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_Freq_TailUpper_lt41000.pdf", width = 5, height = 5)
func.freq_plot(sample_qupper[sample_qupper < 41000])   +
  labs(x = "\n VWAP", y = "Count \n", title = Stock1,
       subtitle = "1-minute VWAP Frequency - Upper Tail (95%) : Excl. Outlier > 41000 ")
dev.off()

## Tails >> MicroPrice

### Lower
qlower <- quantile((AGL_freq_quotes$MicroPrice_OHLCV), 0.05)
sample_qlower <- (AGL_freq_quotes$MicroPrice_OHLCV)[
  (AGL_freq_quotes$MicroPrice_OHLCV) <= qlower]

cairo_pdf("HFT_Ass2_fig_AGL1min_MicroPrice_Freq_LowerTail.pdf", width = 5, height = 5)
func.freq_plot(sample_qlower) +
  labs(x = "\n MicroPrice", y = "Count \n", title = Stock1,
       subtitle = "1-minute MicroPrice Frequency - Lower Tail (5%)")
dev.off()

### Upper
qupper <- quantile((AGL_freq_quotes$MicroPrice_OHLCV), 0.95)
sample_qupper <- (AGL_freq_quotes$MicroPrice_OHLCV)[
  (AGL_freq_quotes$MicroPrice_OHLCV) >= qupper]

cairo_pdf("HFT_Ass2_fig_AGL1min_MicroPrice_Freq_UpperTail.pdf", width = 5, height = 5)
func.freq_plot(sample_qupper)  +
  labs(x = "\n MicroPrice", y = "Count \n", title = Stock1,
       subtitle = "1-minute MicroPrice Frequency - Upper Tail (95%)")
dev.off()

## ACFs
func.acf_plot <- function(data)
{

  acfres <- acf(data, main = "", lag.max = length(data), plot = F)

  afcs <- tibble(ACF = as.numeric(acfres$acf),
                 Lag = as.numeric(acfres$lag))

  ggplot(afcs, aes(x = Lag, y = ACF)) +
    geom_line() +
    labs(y = "ACF \n", x = "\n Lag") +
    theme_bw() +
    theme(aspect.ratio = 0.75)

}

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_ACF.pdf", width = 5, height = 5)
func.acf_plot(AGL_freq_trades$VWAP) +
  labs(title = Stock1,
       subtitle = "Autocorrelation of 1-minute VWAP on Absolute Scale")
dev.off()

cairo_pdf("HFT_Ass2_fig_AGL1min_VWAP_ACF_DiffLog.pdf", width = 5, height = 5)
func.acf_plot(diff(log(AGL_freq_trades$VWAP))) +
  labs(title = Stock1,
       subtitle = "Autocorrelation of Differenced 1-minute VWAP on Absolute Scale")
dev.off()

cairo_pdf("HFT_Ass2_fig_AGL1min_MicroPrice_ACF.pdf", width = 5, height = 5)
func.acf_plot(AGL_freq_quotes$MicroPrice_OHLCV) +
  labs(title = Stock1,
       subtitle = "Autocorrelation of 1-minute MicroPrice on Absolute Scale")
dev.off()

```

```r
cairo_pdf("HFT_Ass2_fig_AGL1min_MicroPrice_ACF_DiffLog.pdf", width = 5, height = 5)
func.acf_plot(diff(AGL_freq_quotes$MicroPrice_OHLCV)) +
  labs(title = Stock1,
       subtitle = "Autocorrelation of Differenced 1-minute MicroPrice on Absolute Scale")
dev.off()

# Part II ----

# Price Impact ----

## Price Impact and seasonality
dat_AGL_all <- dat_TAQ$AGL
dat_NPN_all <- dat_TAQ$NPN
dev.off()

# 20 log normal bins
bins <- logspace(-3, 0.5, n = 21)

## for each bin we want the mean(delta price) and sum(volume)/TotalVolume

func.price_impact_curve <- function(data, stock, bins)
{

dat_Price_impact <- data %>%
  mutate(BuyerSellerInitiated = lag(data$Trade.Sign, 1),
         VolumeTraded = lag(data$Volume.Trade, 1)) %>%
  select(DateTimeL, Mid.Price.Change,
         BuyerSellerInitiated, VolumeTraded) %>%
  filter(BuyerSellerInitiated != "0") %>%
  mutate(BuyerSellerInitiated = as.factor(BuyerSellerInitiated),
         NormalisedVolume = VolumeTraded/mean(VolumeTraded, na.rm = TRUE))

dat_Price_impact$bins <- cut(dat_Price_impact$NormalisedVolume, bins)
dat_Price_impact <- na.omit(dat_Price_impact)

dat_Price_impact <- dat_Price_impact %>%
  filter(BuyerSellerInitiated != "0") %>%
  mutate(BuyerSellerInitiated = as.factor(BuyerSellerInitiated)) %>%
  group_by(bins, BuyerSellerInitiated) %>%
  mutate(AvgMidPriceChange = mean(Mid.Price.Change),
         AvgNormalisedVolume = mean(NormalisedVolume)) %>%
  ungroup()

Price_impact_plot <- dat_Price_impact %>%
  group_by(bins, BuyerSellerInitiated) %>%
  slice(n())

ggplot(Price_impact_plot, aes(y = AvgMidPriceChange,
                              x = AvgNormalisedVolume,
                              col = BuyerSellerInitiated)) +
  geom_line() +
  geom_point() +
  theme_bw() +
  scale_color_manual(labels = c("Buyer Initiated", "Seller Initiated"),
                     values = c("cornflowerblue", "darkorange2")) +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
                labels = trans_format("log10", math_format(10^.x))) +
  labs(x = "\n Average Normalised Volume",
       y = "Average Price Impact \n",
       title = stock,
       subtitle = "Average Price Impact Curves",
       color = "")  +
  scale_y_continuous(label = math_format()) +
  annotation_logticks(sides  = "b") +
  theme(aspect.ratio = 0.75,
        legend.direction = 'horizontal',
        legend.position = "bottom")
}

setwd(dir_figs)
cairo_pdf("HFT_Ass2_fig_AGL_PriceImpact.pdf", height = 5, width = 5)
func.price_impact_curve(dat_AGL_all, Stock1, bins)
dev.off()
```

```
cairo_pdf("HFT_Ass2_fig_NPN_PriceImpact.pdf", height = 5, width = 5)
func.price_impact_curve(dat_NPN_all, Stock2, bins)
dev.off()

 # Order Book Sesonality ----

# Average the volumes traded across all the days in your data and
# plot the aggregate volume as a function of time across the day
# (intraday volume curves) as normalised by the daily volume.

func.normalise <- function(data)
{

  num <- data - min(data, na.rm = TRUE)
  denom <- max(data, na.rm = TRUE) - min(data, na.rm = TRUE)

  dat_norm <- num/denom
  return(dat_norm)

}

func.norm_trade_vol_plot <- function(data, stock)
{

  ## Do we want to get a daily average and take the timevol/daily average
  dat_NormTradeVol_by_DailyVol <- data %>%
    filter(Type == "Trade") %>% # trade data
    mutate(date = as.Date(DateTimeL),
           time = format(DateTimeL,"%H:%M")) %>% # seperate date and time
    group_by(date) %>%
    mutate(NormVol = func.normalise(Volume.Trade)) %>% # daily volume
    ungroup() %>%
    group_by(time) %>%
    mutate(AvgNormVol = mean(NormVol, na.rm = TRUE)) %>% # avg of vol at t/dayvol
    slice(n()) %>% # only need one row from each time
    ungroup() %>%
    mutate(time = as.POSIXct(strptime(time, format = "%H:%M"), na.rm = TRUE))

  # as.POSIXct(strptime(test1$time, format = "%H:%M:%S"))

  ggplot(dat_NormTradeVol_by_DailyVol, aes(x = time, y = AvgNormVol)) +
    geom_line() +
    scale_x_datetime(date_labels = "%H:%M:%S",
                     timezone = "Africa/Johannesburg") +
    labs(x = "\n Time (HH:MM:SS)",
         y = "Average Normalised Trading Volume per Unit Time \n", title = stock,
         subtitle = "Normalised by the Daily Volume") +
    theme_bw()

}

cairo_pdf("HFT_Ass2_fig_AGL_NormTradeVol.pdf", height = 5, width = 5)
func.norm_trade_vol_plot(dat_AGL_all, Stock1)
dev.off()

cairo_pdf("HFT_Ass2_fig_NPN_NormTradeVol.pdf", height = 5, width = 5)
func.norm_trade_vol_plot(dat_NPN_all, Stock2)
dev.off()

func.norm_abs_ret_plot <- function(data, stock)
{

  ## Do we want to get a daily average and take the timevol/daily average
  dat_NormIntradayRet_by_DailyAbsRet <- data %>%
    filter(Type == "Trade") %>% # trade data
    mutate(date = as.Date(DateTimeL),
           time = format(DateTimeL,"%H:%M")) %>% # seperate date and time
    mutate(AbsReturns = c(0, abs(diff(log(Price))))) %>% # log returns
    group_by(date) %>%
    mutate(NormRet = func.normalise(AbsReturns)) %>% # normalise
    ungroup() %>%
    group_by(time) %>%
```

```r
    mutate(AvgNormRet = mean(NormRet, na.rm = TRUE)) %>% # avg
    slice(n()) %>% # only need one row from each time
    ungroup() %>%
    mutate(time = as.POSIXct(strptime(time, format = "%H:%M"), na.rm = TRUE))

  # as.POSIXct(strptime(test1$time, format = "%H:%M:%S"))

  ggplot(dat_NormIntradayRet_by_DailyAbsRet, aes(x = time, y = AvgNormRet)) +
    geom_line() +
    scale_x_datetime(date_labels = "%H:%M:%S",
                     timezone = "Africa/Johannesburg") +
    labs(x = "\n Time (HH:MM:SS)",
         y = "Average Normalised Intraday Return \n", title = stock,
         subtitle = "Normalised by the Average Daily Absolute Intraday Returns") +
    theme_bw()

}

cairo_pdf("HFT_Ass2_fig_AGL_NormAbsRet.pdf", height = 5, width = 5)
func.norm_abs_ret_plot(dat_AGL_all, Stock1)
dev.off()

cairo_pdf("HFT_Ass2_fig_NPN_NormAbsRet.pdf", height = 5, width = 5)
func.norm_abs_ret_plot(dat_NPN_all, Stock2)
dev.off()

func.spreads_plot <- function(data, stock)
{

  ## Do we want to get a daily average and take the timevol/daily average
  dat_spreads <- data %>%
    filter(Type == "Quote") %>% # trade data
    mutate(date = as.Date(DateTimeL),
           time = format(DateTimeL,"%H:%M")) %>% # seperate date and time
    group_by(date) %>%
    mutate(DailySpread = L1.Ask - L1.Bid,
           NormDailySpread = func.normalise(DailySpread)) %>% # Daily returns
    ungroup() %>%
    group_by(time) %>%
    mutate(AvgNormIntradaySpread = mean(NormDailySpread, na.rm = TRUE)) %>% # avg ret @ t
    slice(n()) %>% # only need one row from each time
    ungroup() %>%
    mutate(time = as.POSIXct(strptime(time, format = "%H:%M"), na.rm = TRUE))

  # as.POSIXct(strptime(test1$time, format = "%H:%M:%S"))

  ggplot(dat_spreads, aes(x = time, y = AvgNormIntradaySpread)) +
    geom_line() +
    scale_x_datetime(date_labels = "%H:%M:%S",
                     timezone = "Africa/Johannesburg") +
    labs(x = "\n Time (HH:MM:SS)",
         y = "Average Spread \n", title = stock,
         subtitle = "Normalised by Average Daily Spread") +
    theme_bw()

}

cairo_pdf("HFT_Ass2_fig_AGL_AvgSpreads.pdf", height = 5, width = 5)
func.spreads_plot(dat_AGL_all, Stock1)
dev.off()

cairo_pdf("HFT_Ass2_fig_NPN_AvgSpreads.pdf", height = 5, width = 5)
func.spreads_plot(dat_NPN_all, Stock2)
dev.off()
```

Plagiarism Declaration Form

A copy of this form, completed and signed, to be attached to all coursework submissions to the Statistical Sciences Department.

COURSE CODE:	**STA5091Z**
COURSE NAME:	**Data Analysis for High-Frequency Trading**
STUDENT NAME:	**Julian Albert**
STUDENT NUMBER:	**ALBJUL005**
GROUP NUMBER:	**2**

PLAGIARISM DECLARATION

- I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
- I have used a generally accepted citation and referencing style. Each contribution to, and quotation in, this tutorial/report/project from the work(s) of other people has been attributed, and has been cited and referenced.
- This tutorial/report/project is my own work.
- I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.
- I acknowledge that copying someone else's assignment or essay, or part of it, is wrong, and declare that this is my own work.
- Agreement to this statement does not exonerate me from the University's plagiarism rules.

Signature: Date: August 30, 2019