obAnalytics Guide

Phil Nocturne

2015-09-02

Contents

Overview

Overview to the package and this guide.

Loading data

lala

Expected format

lala

Processing, saving and loading

lala

Preprocessed example data

lala

Visualisation

lala

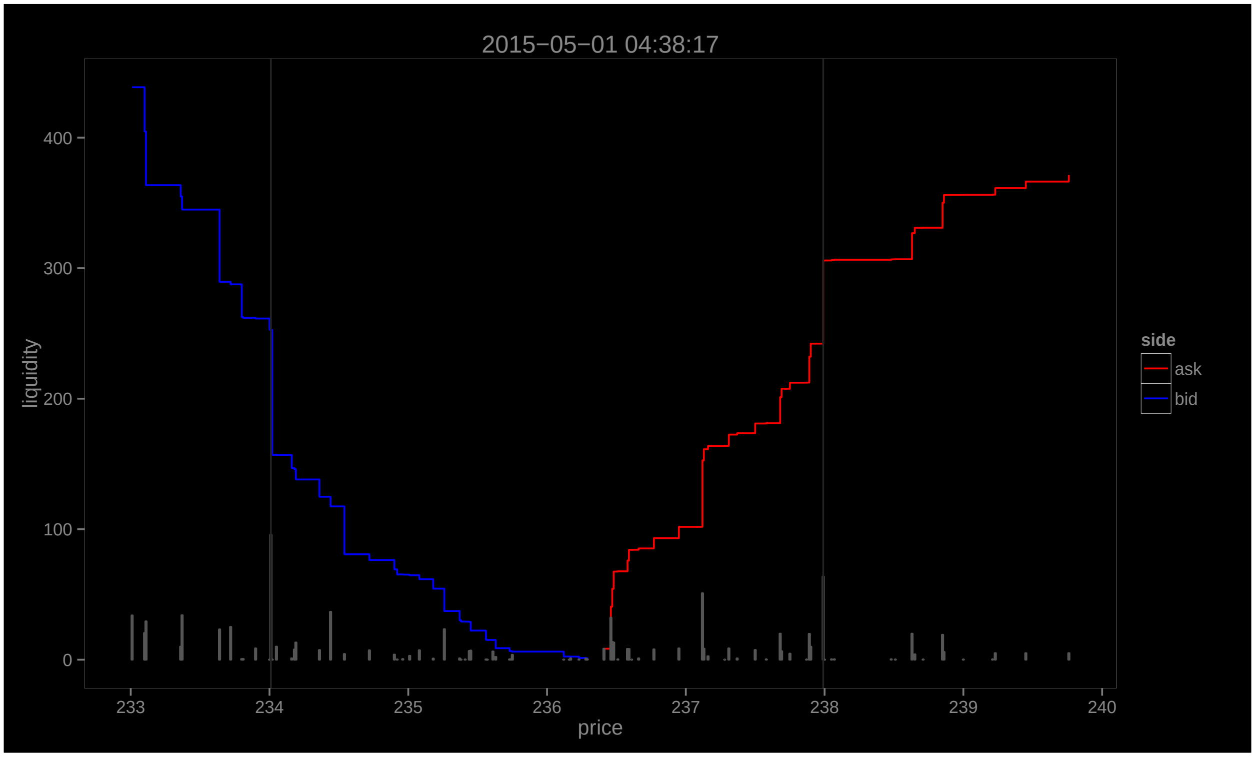
Order book shape

The purpose of the cumulative volume graph is to quickly identify the shape of the limit order book for the given point in time. The “shape” is defined as the cumulative volume available at each price level, starting at the best bid/ask.

Using this shape, it is possible to visually summarise order book imbalance and market depth.

```r
# get a limit order book for a specific point in time, limited to +- 150bps
# above/below best bid/ask price.
lob <- orderBook(lob.data$events,
    tp=as.POSIXct("2015-05-01 04:38:17.429", tz="UTC"), bps.range=150)

# visualise the order book liquidity.
plotCurrentDepth(lob, volume.scale=10^-8)
```



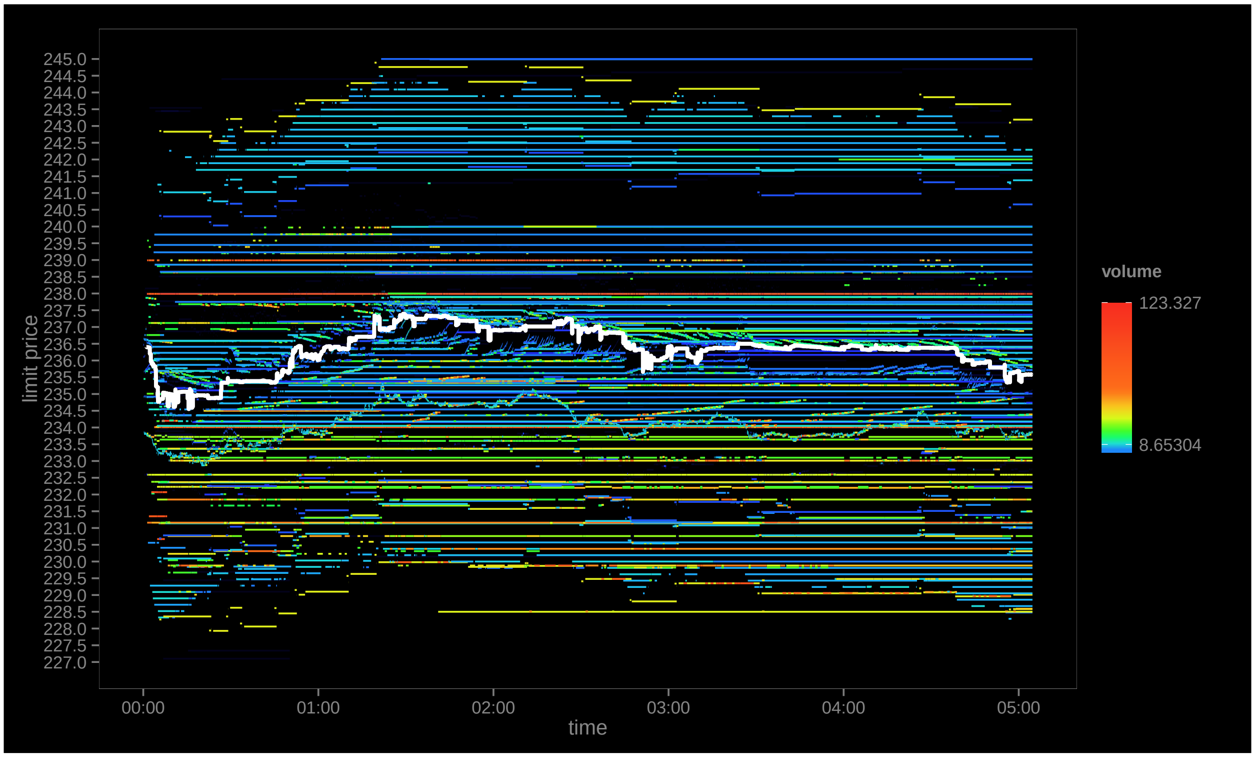
In the figure above, an order book has been reconstructed with the *orderBook* function for a specific point in time. The visualisation produced with the *plotCurrentDepth* function depicts a number of order book features. Firstly, the embedded bar chart at the bottom of the plot shows the amount of volume available at specific price levels ranging from the *bid* side on the left (blue) through to the *ask* side (red) on the right.

Secondly, the blue and red lines show the *cumulative* volume of the bar chart for the bid and ask sides of the order book respectively. Finally, the two subtle vertical lines at price points $234 and $238 show the position of the top 1% largest limit orders.

Price level volume

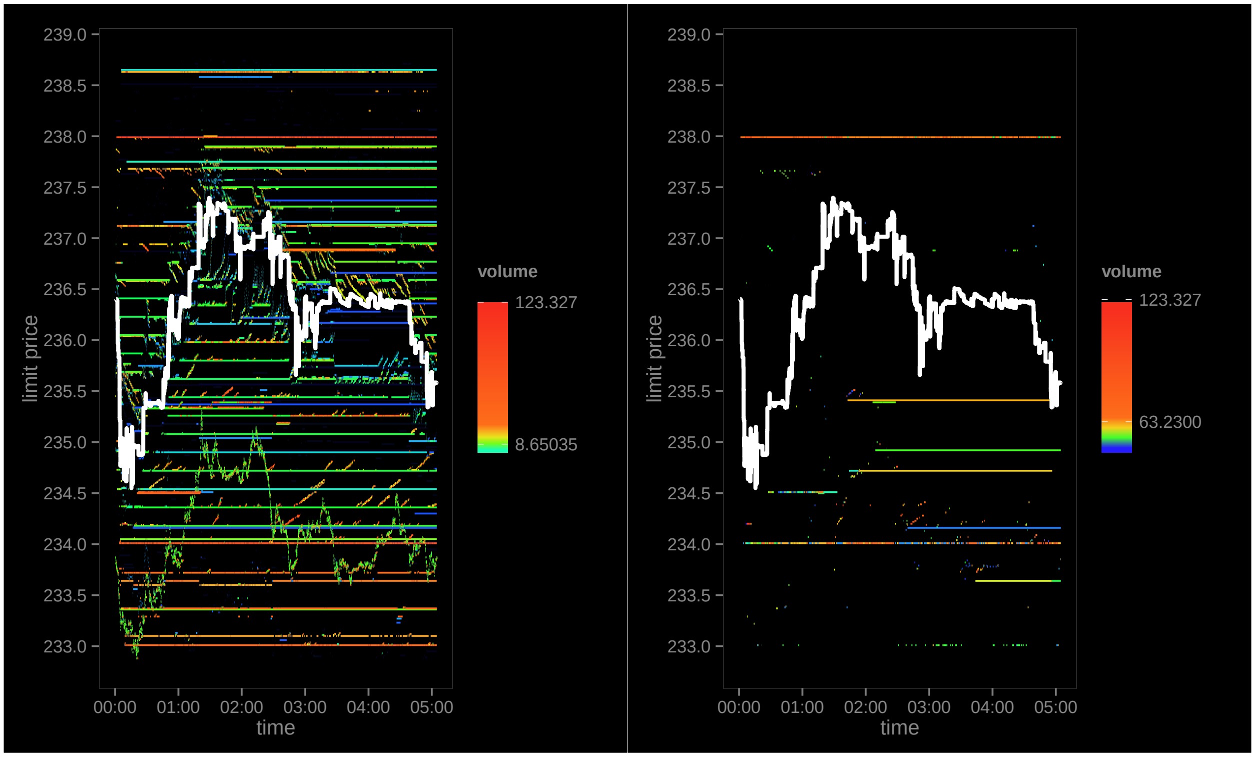
All lob.data:

```
# plot all lob.data price level volume between $247 and $245 and overlay the
# market midprice.
spread <- getSpread(lob.data$depth.summary)
plotPriceLevels(lob.data$depth, spread, price.from=227, price.to=245,
    volume.scale=10^-8, col.bias=0.25, show.mp=T)
```



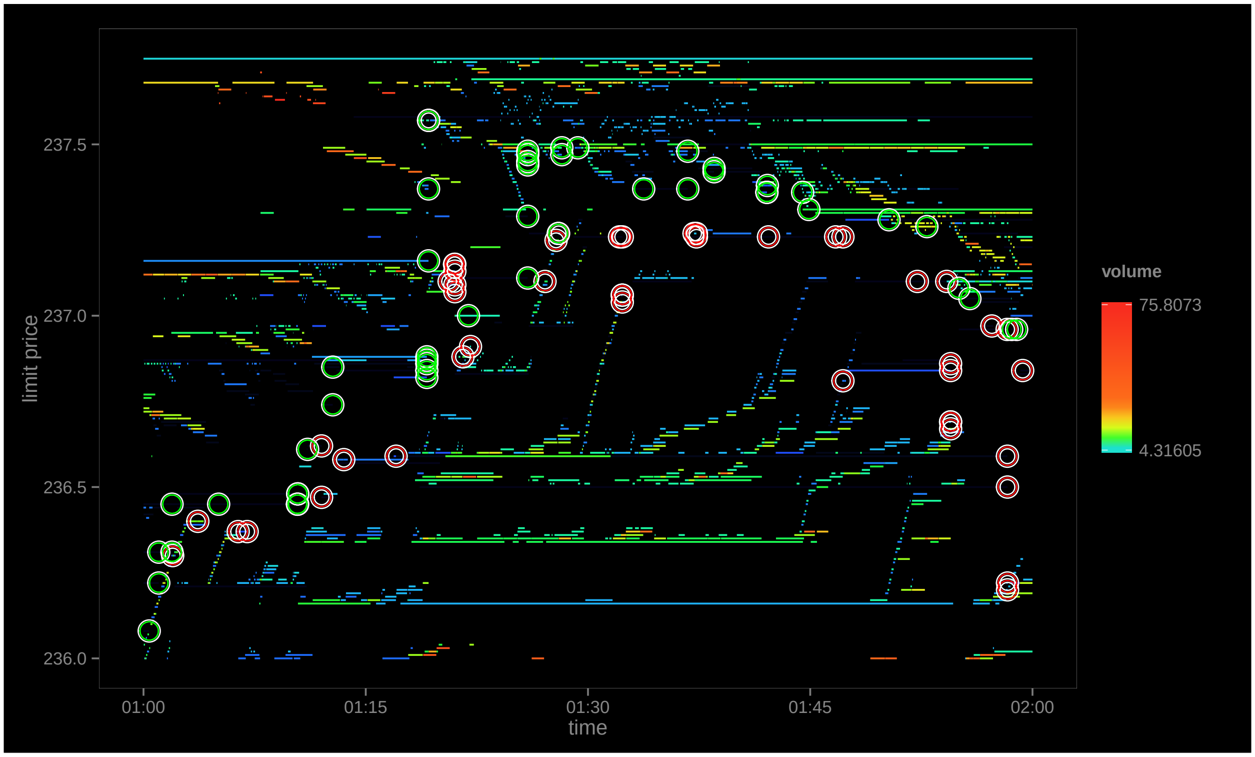
plot all depth levels, rescaling the volume by 10ˆ-8. produce 2 plots side-by-side: second plot contains depth levels with > 50 units of volume.

```
spread <- with(lob.data, getSpread(depth.summary))
p1 <- with(lob.data, plotPriceLevels(depth, spread, col.bias=0.1, volume.scale=10^-8))
p2 <- with(lob.data, plotPriceLevels(depth, spread, col.bias=0.1, volume.scale=10^-8, volume.from=50))
library(grid)
pushViewport(viewport(layout=grid.layout(1, 2)))
print(p1, vp=viewport(layout.pos.row=1, layout.pos.col=1))
print(p2, vp=viewport(layout.pos.row=1, layout.pos.col=2))
```



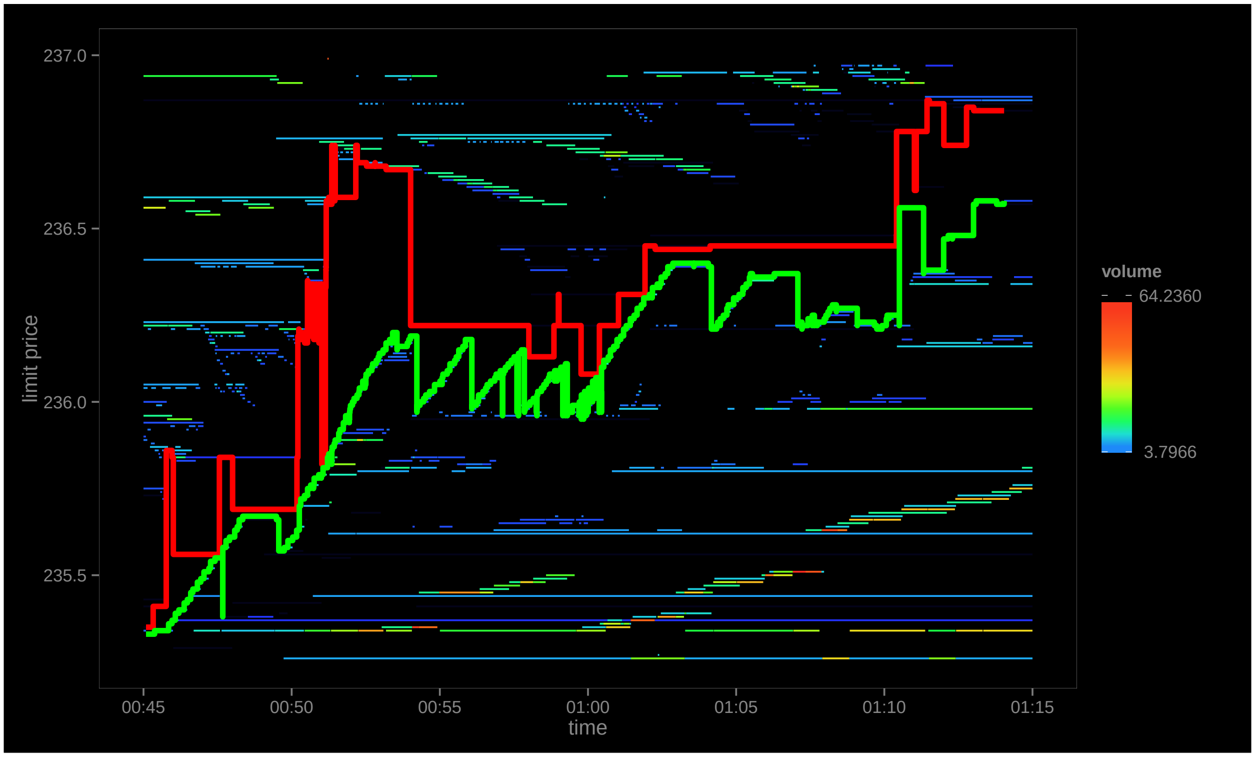
with trades:

```
# plot 1 hour of trades centred around the bid/ask spread.
plotPriceLevels(lob.data$depth, trades=lob.data$trades,
    price.from=236, price.to=237.75, volume.scale=10^-8, col.bias=0.2,
    start.time=as.POSIXct("2015-05-01 01:00:00.000", tz="UTC"),
    end.time=as.POSIXct("2015-05-01 02:00:00.000", tz="UTC"))
```



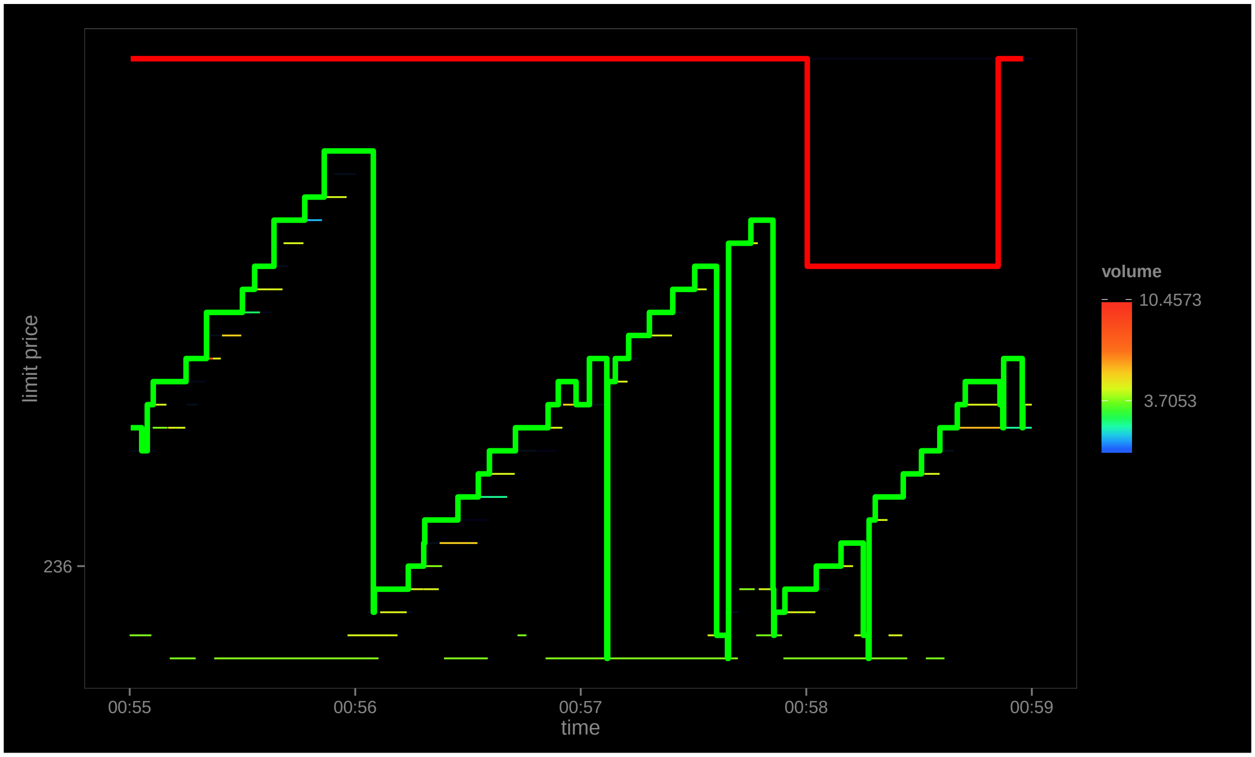
30 minute zoom:

```
# zoom in to 30 minutes of bid/ask quotes.
spread <- getSpread(lob.data$depth.summary)
plotPriceLevels(lob.data$depth, spread, price.from=235.25, price.to=237,
    start.time=as.POSIXct("2015-05-01 00:45:00.000", tz="UTC"),
    end.time=as.POSIXct("2015-05-01 01:15:00.000", tz="UTC"),
    volume.scale=10^-8, col.bias=0.5, show.mp=F)
```



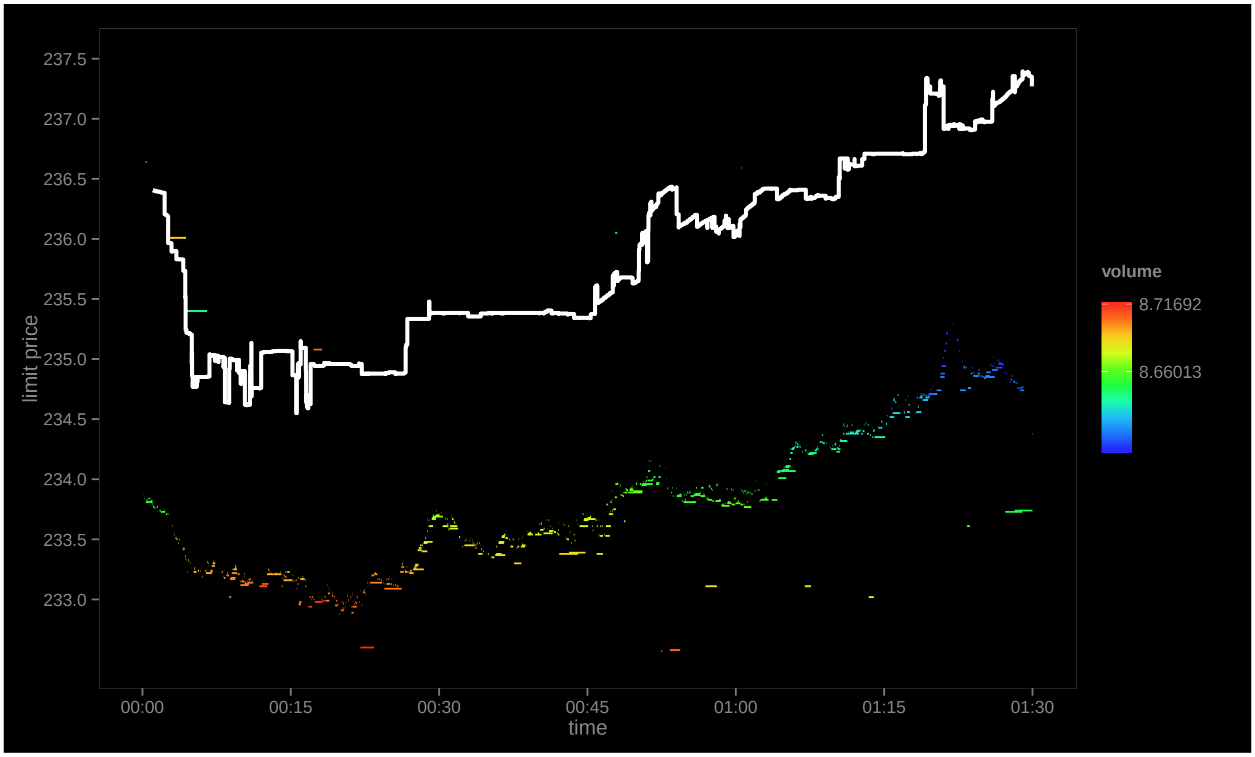
4 minute zoom:

```
# zoom in to 4 minutes of bid/ask quotes.
spread <- getSpread(lob.data$depth.summary)
plotPriceLevels(lob.data$depth, spread, price.from=235.90, price.to=236.25,
    start.time=as.POSIXct("2015-05-01 00:55:00.000", tz="UTC"),
    end.time=as.POSIXct("2015-05-01 00:59:00.000", tz="UTC"),
    volume.scale=10^-8, col.bias=0.5, show.mp=F)
```

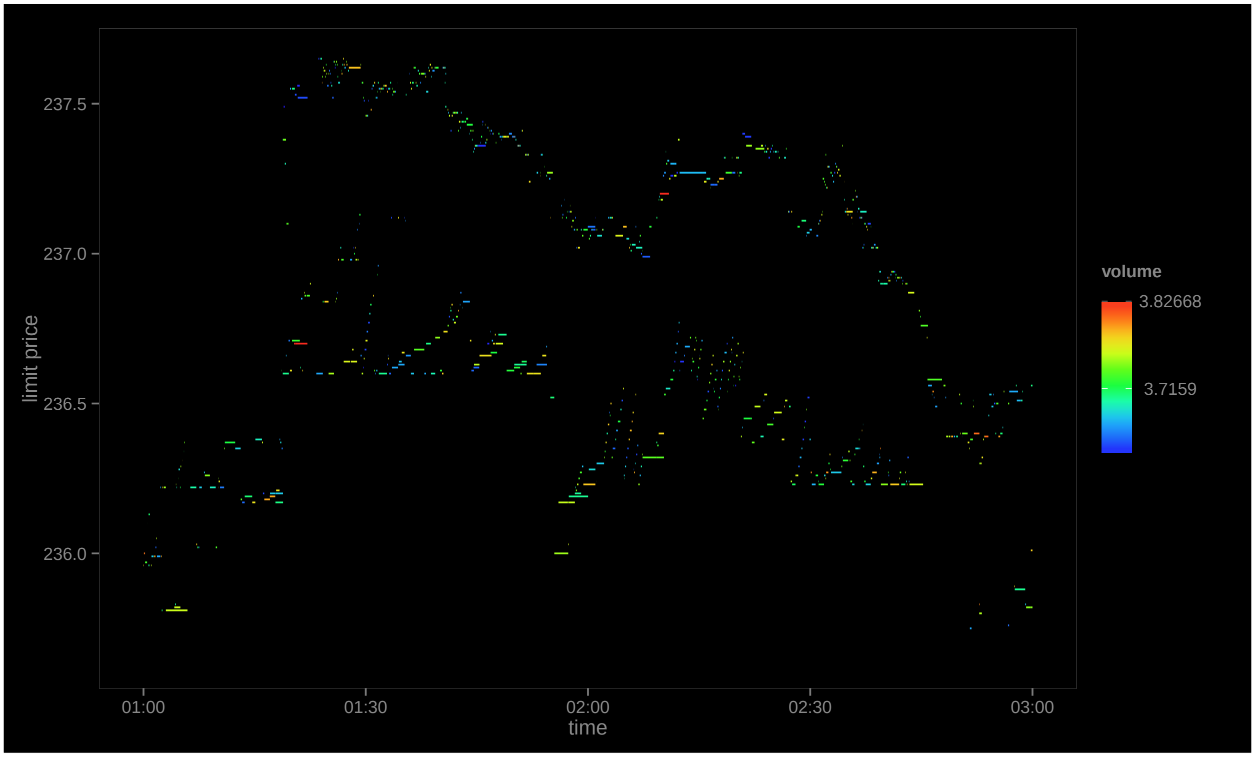


individual market paricipant:

```
#
spread <- getSpread(lob.data$depth.summary)
plotPriceLevels(lob.data$depth, spread, price.from=232.5, price.to=237.5,
    volume.scale=10^-8, col.bias=1, show.mp=T,
    end.time=as.POSIXct("2015-05-01 01:30:00.000", tz="UTC"),
    volume.from=8.59, volume.to=8.72)
```



```
#
plotPriceLevels(lob.data$depth, price.from=235.65, price.to=237.65,
    volume.scale=10^-8, col.bias=1,
    start.time=as.POSIXct("2015-05-01 01:00:00.000", tz="UTC"),
    end.time=as.POSIXct("2015-05-01 03:00:00.000", tz="UTC"),
    volume.from=3.63, volume.to=3.83)
```

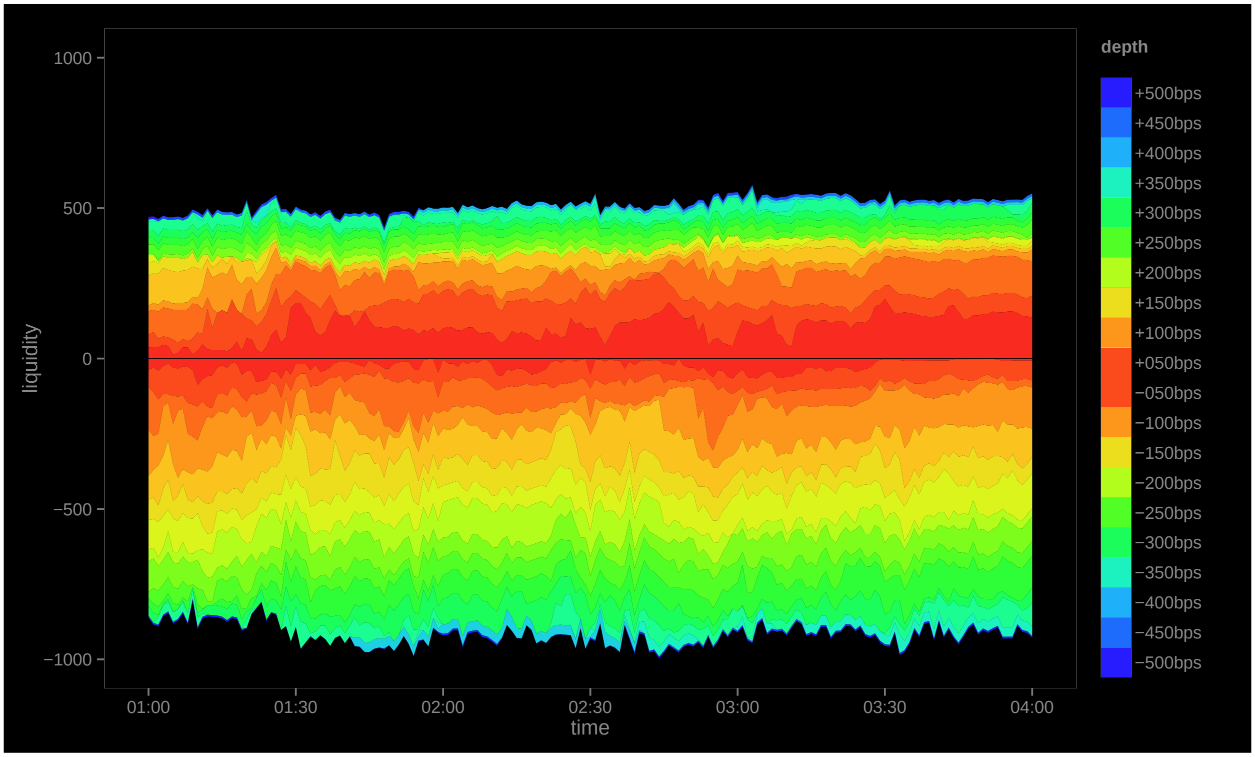


The available volume at each price level is colour coded according to the range of volume at all price levels. The colour coding follows the visible spectrum, such that larger amounts of volume appear "hotter" than smaller amounts, where cold = blue, hot = red. Since the distribution of limit order size exponentially decays, it can be difficult to visually differentiate: most values will appear to be blue. The function provides price, volume and a colour bias range to overcome this.

Liquidity

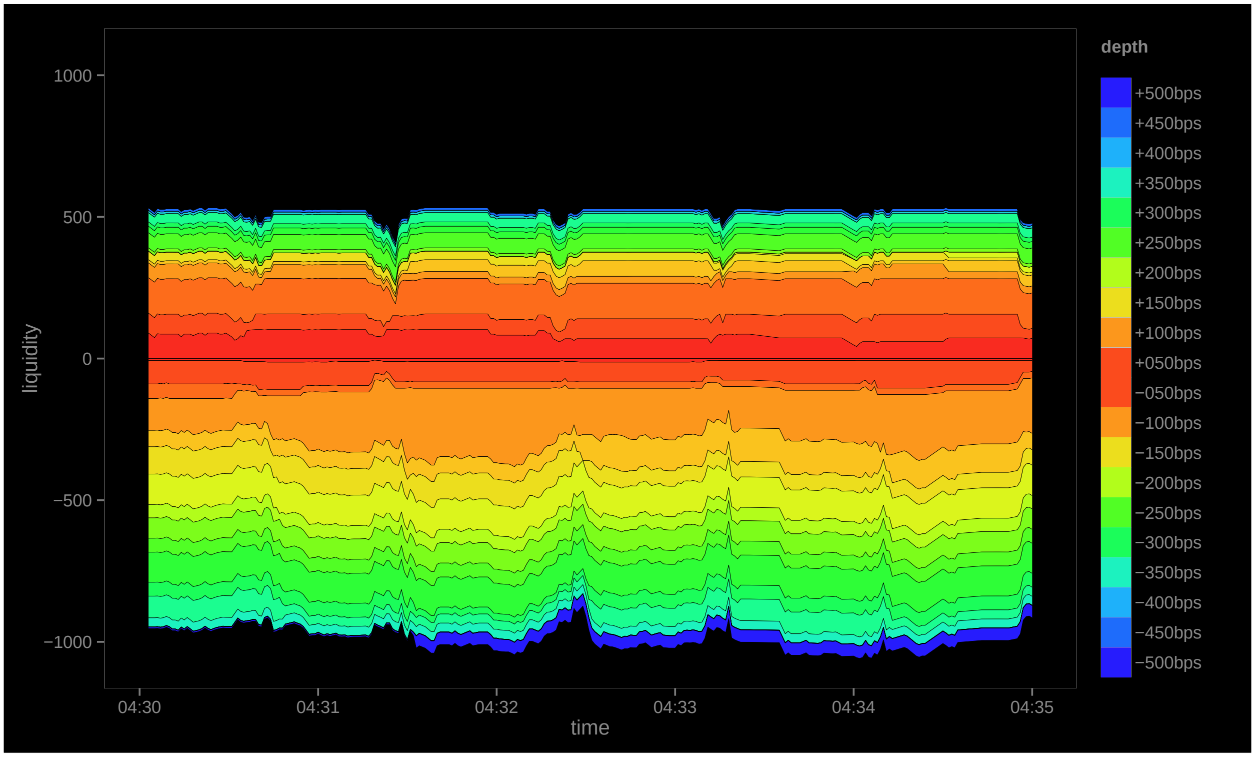
liquidity…

```
plotVolumePercentiles(lob.data$depth.summary, volume.scale=10^-8, perc.line=F, start.time=as.POSIXct("20
    end.time=as.POSIXct("2015-05-01 04:00:00.000", tz="UTC"))
```



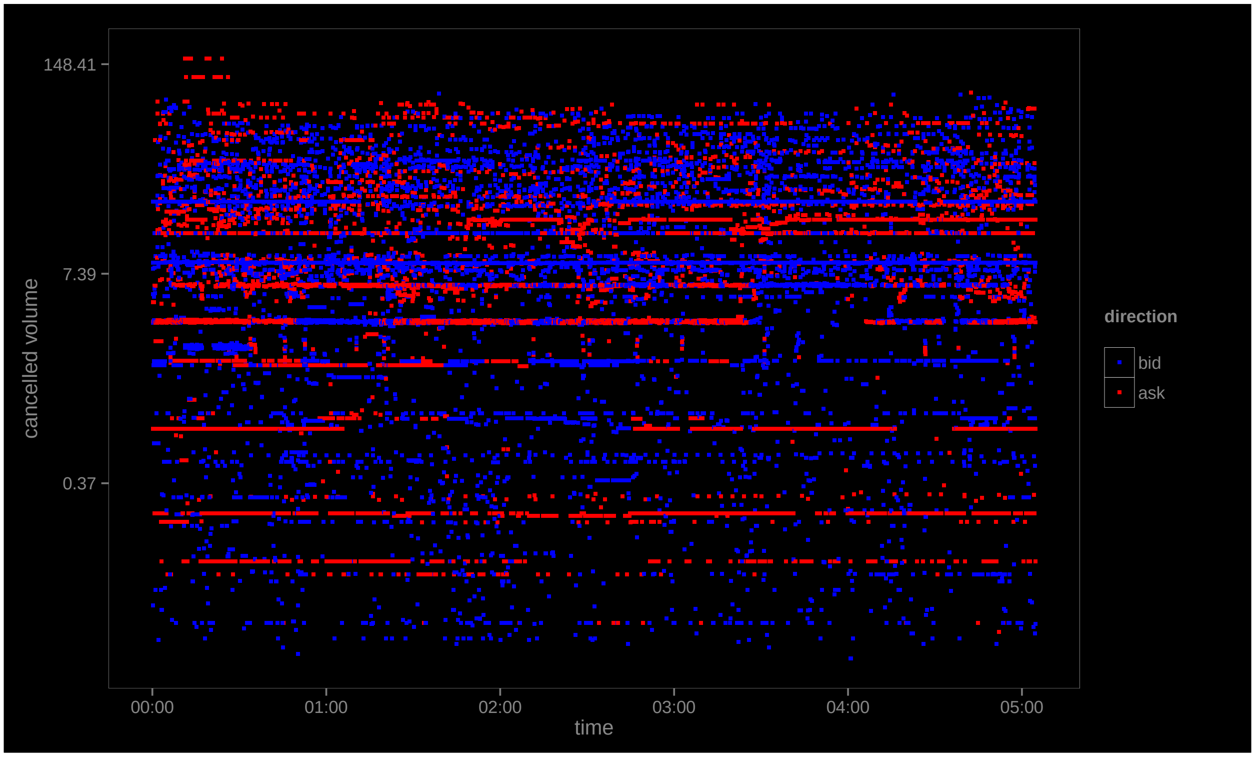
another. . .

```r
# visualise 15 minutes of order book liquidity.
# data will be aggregated to second-by-second resolution.
plotVolumePercentiles(lob.data$depth.summary,
start.time=as.POSIXct("2015-05-01 04:30:00.000", tz="UTC"),
end.time=as.POSIXct("2015-05-01 04:35:00.000", tz="UTC"),
volume.scale=10^-8)
```

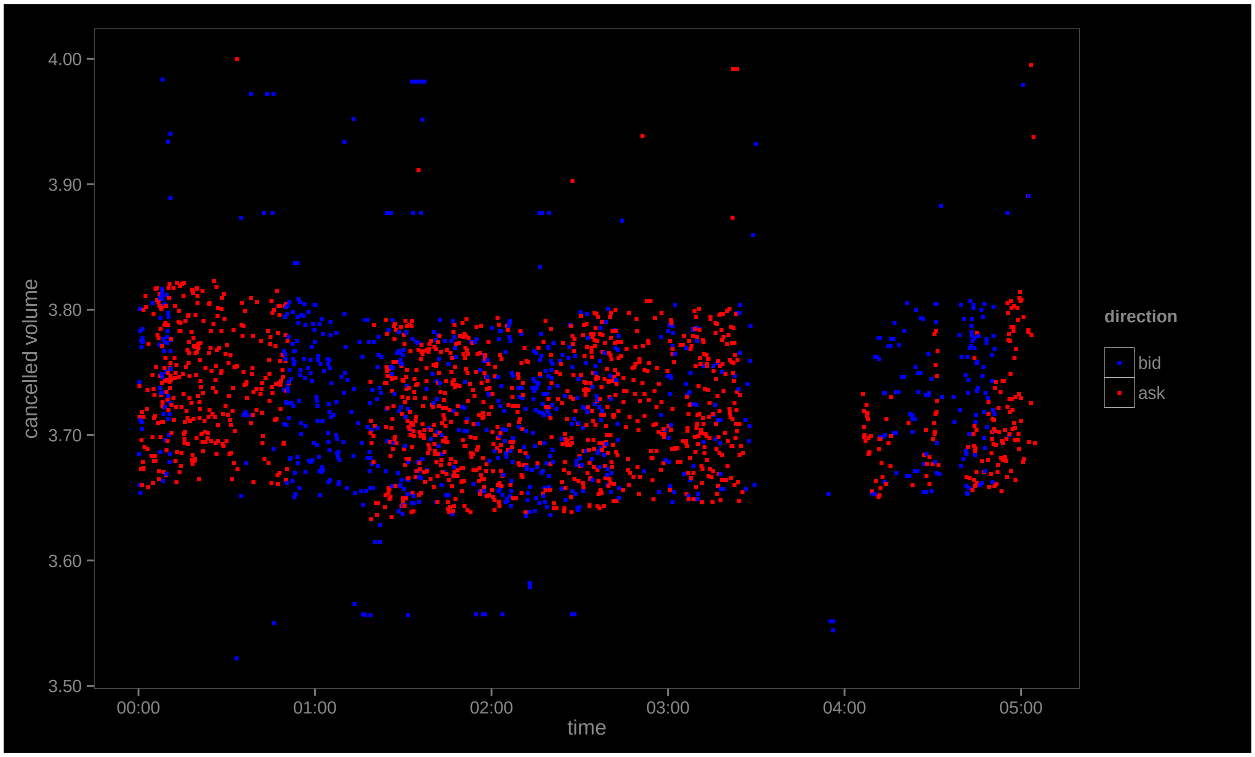


Fleeting orders

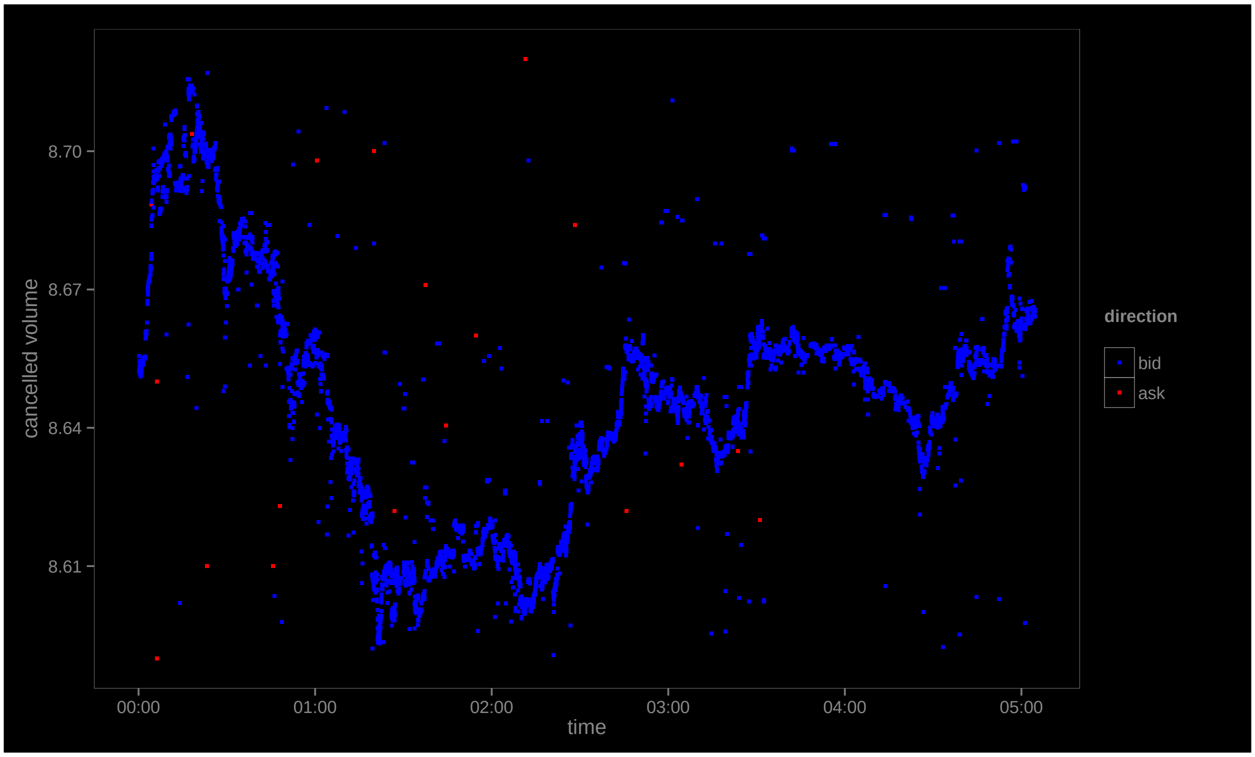
```
plotVolumeMap(lob.data$events, volume.scale=10^-8, log.scale = T)
```



```
plotVolumeMap(lob.data$events, volume.scale=10^-8, volume.from=3.5, volume.to=4)
```



```
plotVolumeMap(lob.data$events, volume.scale=10^-8, volume.from=8.59, volume.to=8.72)
```



Analysis

lala.

Order book reconstruction

lala.

```
tp <- as.POSIXct("2015-05-01 04:25:15.342", tz="UTC")
ob <- orderBook(lob.data$events, max.levels=10)
with(ob, {
  asks$liquidity <- round(asks$liquidity*10^-8, 2)
  bids$liquidity <- round(bids$liquidity*10^-8, 2)
  cols <- c("id", "timestamp", "price", "liquidity")
  print(asks[, cols], row.names=F)
  print(bids[, cols], row.names=F)
})
```

```
##        id           timestamp  price liquidity
##  65619449 2015-05-01 04:55:15 236.05     86.75
##  65619719 2015-05-01 04:58:04 235.90     79.94
##  65619475 2015-05-01 04:55:25 235.87     79.90
##  65620112 2015-05-01 05:04:19 235.85     73.11
##  65620138 2015-05-01 05:04:41 235.84     50.29
##  65620086 2015-05-01 05:03:45 235.81     34.31
##  65620107 2015-05-01 05:04:17 235.80     21.11
##  65619914 2015-05-01 05:01:00 235.72      7.91
##  65620140 2015-05-01 05:04:42 235.71      7.70
##  65620105 2015-05-01 05:04:16 235.71      3.91
##        id           timestamp  price liquidity
##  65619912 2015-05-01 05:03:13 235.45      0.16
##  65620122 2015-05-01 05:04:26 235.12      1.10
##  65620109 2015-05-01 05:04:18 235.10      2.03
##  65618028 2015-05-01 05:00:08 235.01      4.57
##  65619358 2015-05-01 04:54:21 234.95      4.67
##  65598930 2015-05-01 00:39:56 234.92      4.80
##  65620023 2015-05-01 05:02:33 234.74      5.88
##  65620062 2015-05-01 05:03:28 234.73     16.86
##  65619669 2015-05-01 04:57:31 234.72     23.99
##  65597424 2015-05-01 00:23:05 234.54     28.36
```

Trades

trades…

```
with(lob.data, {
  trades.ex <- head(trades[, c("timestamp", "price", "volume", "direction", "maker", "taker")], 10)
  trades.ex$volume <- round(trades.ex$volume*10^-8, 2)
  print(trades.ex, row.names=F)
})
```

```
##             timestamp  price volume direction    maker    taker
##  2015-05-01 00:00:06 236.47   1.79      sell 65595247 65595250
##  2015-05-01 00:00:50 236.61   2.11       buy 65595284 65595314
##  2015-05-01 00:02:14 236.27   0.21      sell 65595325 65595444
##  2015-05-01 00:02:56 235.92   2.00      sell 65595446 65595491
##  2015-05-01 00:02:59 236.01   0.21       buy 65595466 65595494
##  2015-05-01 00:03:35 236.01   0.07       buy 65595466 65595549
##  2015-05-01 00:04:00 236.00   0.21       buy 65595550 65595587
##  2015-05-01 00:04:24 235.45   2.00       buy 65595610 65595614
##  2015-05-01 00:04:42 235.04   2.12      sell 65595607 65595653
##  2015-05-01 00:04:42 235.02  10.37      sell 65595603 65595653
```

Market impacts

lala.

```r
# get impacts data.frame from trades data.
impacts <- tradeImpacts(lob.data$trades)
# impacts (in bps)
sell.bps <- with(impacts[impacts$dir == "sell", ], {
(max.price-min.price)/max.price
})
10000*summary(sell.bps[sell.bps > 0])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4215  1.8060  7.8130 12.0000 18.7800 37.8600
```