Trade Costs

by Jeff Enos, David Kane, Arjun Ravi Narayan, Aaron Schwartz, Daniel Suo and Luyi Zhao

Introduction

Trade costs are the costs a trader must pay to implement a decision to buy or sell a security. Consider a single trade of a single equity security. Suppose on the evening of August 1, a trader decides to purchase 10,000 shares of IBM at $10, the *decision price* of the trade. The next day, the trader's broker buys 10,000 shares in a rising market and pays $11 per share, the trade's *execution price*.

How much did it cost to implement this trade? In the most basic ex-post analysis, trade costs are calculated by comparing the execution price of a trade to a benchmark price.[1] Suppose we wished to compare the execution price to the price of the security at the time of the decision in the above example. Since the trader's decision occurred at $10 and the broker paid $11, the cost of the trade relative to the decision price was $\$11 - \$10 = \$1$ per share, or $10,000 (9.1% of the total value of the execution).

Measuring costs relative to a trade's decision price captures costs associated with the delay in the release of a trade into the market and movements in price after the decision was made but before the order is completed. It does not, however, provide a means to determine whether the broker's execution reflects a fair price. For example, the price of $11 would be a poor price if most transactions in IBM on August 2 occurred at $10.50. For this purpose a better benchmark would be the day's volume-weighted average price, or VWAP. If VWAP on August 2 was $10.50 and the trader used this as her benchmark, then the trade cost would be $0.50 per share, or $5,000.

The first version of the **tradeCosts** package provides a simple framework for calculating the cost of trades relative to a benchmark price, such as VWAP or decision price, over multiple periods along with basic reporting and plotting facilities to analyse these costs.

Trade costs in a single period

Suppose we want to calculate trade costs for a single period. First, the data required to run the analysis must be assembled into three data frames.

The first data frame contains all trade-specific information, a sample of which is in the `trade.mar.2007` data frame:

```
> library("tradeCosts")
> data("trade.mar.2007")
> head(trade.mar.2007)

      period       id side exec.qty exec.price
1 2007-03-01 03818830    X    60600       1.60
2 2007-03-01 13959410    B     4400      32.21
3 2007-03-01 15976510    X    13600       7.19
4 2007-03-01 22122P10    X   119000       5.69
5 2007-03-01 25383010    X     9200       2.49
6 2007-03-01 32084110    B     3400      22.77
```

Trading data must include at least the set of columns included in the sample shown above: `period` is the (arbitrary) time interval during which the trade was executed, in this case a calendar trade day; `id` is a unique security identifier; `side` must be one of `B` (buy), `S` (sell), `C` (cover) or `X` (short sell); `exec.qty` is the number of shares executed; and `exec.price` is the price per share of the execution. The `create.trade.data` function can be used to create a data frame with all of the necessary information.

Second, trade cost analysis requires dynamic descriptive data, or data that changes across periods for each security.

```
> data("dynamic.mar.2007")
> head(dynamic.mar.2007[c("period", "id", "vwap",
+      "prior.close")])

      period       id   vwap prior.close
1 2007-03-01 00797520   3.88        3.34
2 2007-03-01   010015 129.35        2.53
3 2007-03-01   023282 613.57       12.02
4 2007-03-01 03818830   1.58        1.62
5 2007-03-01   047628 285.67        5.61
6 2007-03-01   091139 418.48        8.22
```

The `period` and `id` columns match those in the trading data. The remaining two columns in the sample are benchmark prices: `vwap` is the volume-weighted average price for the period and `prior.close` is the security's price at the end of the prior period.

The third data frame contains static data for each security.

```
> data("static.mar.2007")
> head(static.mar.2007)

           id symbol                        name sector
1301 00036020   AAON                    Aaon Inc    IND
2679 00036110    AIR                    Aar Corp    IND
3862 00040010   ABCB              Ameris Bancorp    FIN
406  00080S10   ABXA                 Abx Air Inc    IND
3239 00081T10    ABD            Acco Brands Corp    IND
325  00083310    ACA Aca Capital Hldgs Inc -redh    FIN
```

The `id` column specifies an identifier that can be linked to the other data frames. Because this data is static, there is no `period` column.

Once assembled, these three data frames can be analysed by the `trade.costs` function:

[1] For an in-depth discussion of both ex-ante modeling and ex-post measurement of trade costs, see Kissell and Glantz (2003).

```
> result <- trade.costs(trade = trade.mar.2007,
+     dynamic = dynamic.mar.2007,
+     static = static.mar.2007,
+     start.period = as.Date("2007-03-01"),
+     end.period = as.Date("2007-03-01"),
+     benchmark.price = "vwap")
```

The `trade`, `dynamic`, and `static` arguments refer to the three data frames discussed above. `start.period` and `end.period` specify the period range to analyse. This example analyses only one period, March 1, 2007, and uses the `vwap` column of the `dynamic` data frame as the benchmark price. `result` is an object of class `tradeCostsResults`.

```
> summary(result)

Trade Cost Analysis

Benchmark Price: vwap

Summary statistics:
 Total Market Value:        1,283,963
 First Period:             2007-03-01
 Last Period:              2007-03-01
 Total Cost:                   -6,491
 Total Cost (bps):                -51

Best and worst batches over all periods:
                          batch.name exec.qty   cost
1 22122P10 (2007-03-01 - 2007-03-01)  119,000 -3,572
2 03818830 (2007-03-01 - 2007-03-01)   60,600 -1,615
3 88362320 (2007-03-01 - 2007-03-01)   31,400 -1,235
6 25383010 (2007-03-01 - 2007-03-01)    9,200     33
7 13959410 (2007-03-01 - 2007-03-01)    4,400    221
8 32084110 (2007-03-01 - 2007-03-01)    3,400    370

Best and worst securities over all periods:
        id exec.qty   cost
1 22122P10  119,000 -3,572
2 03818830   60,600 -1,615
3 88362320   31,400 -1,235
6 25383010    9,200     33
7 13959410    4,400    221
8 32084110    3,400    370

NA report:
           count
id             0
period         0
side           2
exec.price     0
exec.qty       0
vwap           1
```

The first section of the report provides high-level summary information. The total unsigned market value of trades for March 1 was around $1.3 million. Relative to VWAP, these trades cost -$6,491, indicating that overall the trades were executed at a level "better" than VWAP, where better buys/covers (sells/shorts) occur at prices below (above) VWAP. This total cost is the sum of the signed cost of each trade relative to the benchmark price. As a percentage of total executed market value, this set of trades cost -51 bps relative to VWAP.

The next section displays the best and worst *batches* over all periods. We will discuss batches in the next section. For now, note that when dealing with only one period, each trade falls into its own batch, so this section shows the most and least expensive trades for March 1. The next section displays the best and worst securities by total cost across all periods. Because there is only one trade per security on March 1, these results match the best and worst batches by cost.

Calculating the cost of a trade requires a non-`NA` value for `id`, `period`, `side`, `exec.price`, `exec.qty` and the benchmark price. The final section shows a count for each type of `NA` in the input data. Rows in the input data with `NA`'s in any of these columns are removed before the analysis is performed and reported here.

Costs over multiple periods

Calculating trade costs over multiple periods works similarly. Cost can be calculated for each trade relative to a benchmark price which either varies over the period of the trade or is fixed at the decision price.

Suppose, for example, that the trader decided to short a stock on a particular day, but he wanted to trade so many shares that it took several days to complete the order. For instance, consider the following sequence of trades in our sample data set for Progressive Gaming, PGIC, which has id `59862K10`:

```
> subset(trade.mar.2007, id %in% "59862K10")

        period       id side exec.qty exec.price
166 2007-03-13 59862K10    X    31700       5.77
184 2007-03-15 59862K10    X    45100       5.28
218 2007-03-19 59862K10    X   135800       5.05
259 2007-03-20 59862K10    X    22600       5.08
```

How should we calculate the cost of these trades? We could calculate the cost for each trade separately relative to a benchmark price such as `vwap`, exactly as in the last example. In this case, the cost of each trade in PGIC would be calculated relative to VWAP in each period and then added together. However, this method would ignore the cost associated with spreading out the sale over several days. If the price of the stock had been falling over the four days of the sale, for example, successive trades appear less attractive when compared to the price at the time of the decision. The trader can capture this cost by grouping the four short sales into a *batch* and comparing the execution price of each trade to the batch's original decision price.

Performing this type of multi-period analysis using **tradeCosts** requires several modifications to the previous single period example. Note that since no period range is given, analysis is performed over the entire data set:

```
> result.batched <- trade.costs(trade.mar.2007,
+     dynamic = dynamic.mar.2007,
+     static = static.mar.2007,
+     batch.method = "same.sided",
+     benchmark.price = "decision.price")
```

First, `trade.costs` must be instructed how to group trades into batches by setting the `batch.method` parameter. This version of **tradeCosts** provides a single multi-period sample batch method, `same.sided`, which groups all consecutive same-sided orders into a single batch. Provided there were no buys in between the four sales in PGIC, all four trades would be grouped into the same batch. Second, setting `benchmark.price` to `decision.price` sets the benchmark price to the prior closing price of the first trade in the batch. Running `summary` on the new result yields the following:

```
> summary(result.batched)

Trade Cost Analysis

Benchmark Price: decision.price

Summary statistics:
 Total Market Value:        47,928,402
 First Period:              2007-03-01
 Last Period:               2007-03-30
 Total Cost:                   587,148
 Total Cost (bps):                 123

Best and worst batches over all periods:
                            batch.name exec.qty    cost
1   04743910 (2007-03-19 - 2007-03-19)   17,800 -82,491
2   31659U30 (2007-03-09 - 2007-03-13)   39,800 -33,910
3   45885A30 (2007-03-13 - 2007-03-19)  152,933 -31,904
274 49330810 (2007-03-13 - 2007-03-30)   83,533  56,598
275 15649210 (2007-03-15 - 2007-03-28)   96,900  71,805
276 59862K10 (2007-03-13 - 2007-03-20)  235,200 182,707

Best and worst securities over all periods:
          id exec.qty    cost
1   04743910   17,800 -82,491
2   31659U30   51,400 -32,616
3   45885A30  152,933 -31,904
251 49330810   83,533  56,598
252 15649210  118,100  73,559
253 59862K10  235,200 182,707

NA report:
            count
id              0
period          0
side            6
exec.price      0
exec.qty        0
prior.close     2
```

This analysis covers almost $50 million of executions from March 1 to March 30, 2007. Relative to decision price, the trades cost $587,148, or 1.23% of the total executed market value.

The most expensive batch in the result contained the four sells in PGIC (`59862K10`) from March 13 to March 20, which cost $182,707.

Plotting results

The **tradeCosts** package includes a `plot` method that displays bar charts of trade costs. It requires two arguments, a `tradeCostsResults` object, and a character string that describes the type of plot to create.

The simplest plot is a time series of total trade costs in basis points over each period:

```
> plot(result.batched, "time.series.bps")
```

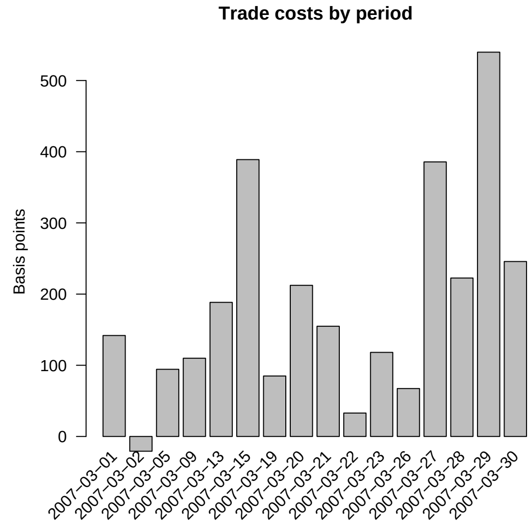

Figure 1: A time series plot of trade costs.

This chart displays the cost for each day in the previous example. According to this chart, all days had positive cost except March 2.

The second plot displays trade costs divided into categories defined by a column in the `static` data frame passed to `trade.costs`. Since `sector` was a column of that data frame, we can look at costs by company sector:

```
> plot(result.batched, "sector")
```

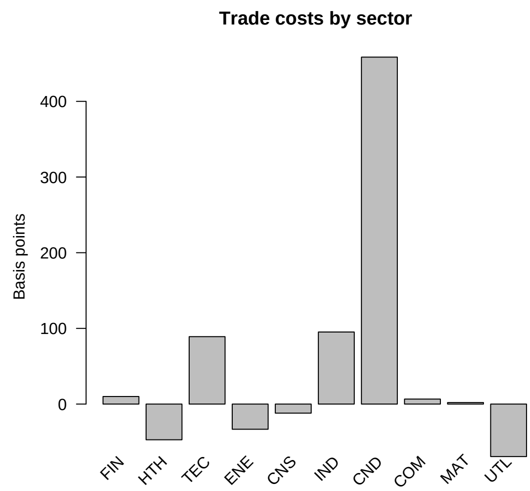

Figure 2: A plot of trade costs by sector.

Over the period of the analysis, trades in `CND` were especially expensive relative to decision price.

The last plot applies only to `same.sided` batched trade cost analysis as we performed in the multi-period example. This chart shows cost separated into the different periods of a batch. The cost of the first batch of PGIC, for example, contributes to the first bar, the cost of the second batch to the second bar, and so on.

```
> plot(result.batched, "cumulative")
```

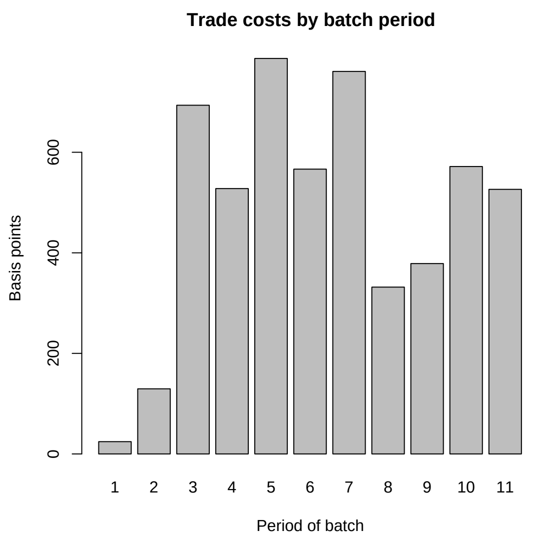

Figure 3: Costs by batch period, in bps.

As one might expect, the first and second trades in a batch are the cheapest with respect to decision price because they occur closest to the time of the decision.

Conclusion

tradeCosts currently provides a simple means of calculating the cost of trades relative to a benchmark price over multiple periods. Costs may be calculated relative to a period-specific benchmark price or, for trades spanning multiple periods, the initial decision price of the trade. We hope that over time and through collaboration the package will be able to tackle more complex issues, such as ex-ante modeling and finer compartmentalization of trade costs.

Bibliography

R. Kissell and M. Glantz. *Optimal Trading Strategies*. American Management Association, 2003.

Jeff Enos, David Kane, Arjun Ravi Narayan, Aaron Schwartz, Daniel Suo, Luyi Zhao
Kane Capital Management
Cambridge, Massachusetts, USA
`jeff@kanecap.com`