Rule 605 Analysis

Bill Alpert, Barron's, william.alpert@barrons.com (mailto:william.alpert@barrons.com), 1.212.416.2742

Friday, December 26, 2014

Introduction

This is the second iteration of our analysis. If anything proves the value of showing my work to you, it's discovering mistakes. I found that I left out some parentheses in a formula, resulting in erroneous numbers for net price-improvement. This iteration of the scripts should produce better estimates.

Thanks for helping us do this story on trade execution. This document summarizes our analysis of your Form 605 reports. The story that results from this analysis — and lots of other reporting — will probably use *net price-improvement* as a measure to rank market makers and exchanges, and in turn, the brokers who route orders to them. We may also make mention of *effective-spread minus realized-spread*. We're sharing these files with you so that you can see why I chose that measure and how I calculate it. The included raw data and and scripts should allow you to *exactly* reproduce my analysis, using the free, open-source software that I describe in the included **"README.html"** file. The measures and graphics you see here are calculated from *your firm's* data.

I hope that you'll use these materials to alert me to errors and to re-analyze the data any way you like.

The file you're reading was created with the app *R Markdown* [1] and you can recreate or update it by using the open-source software **R** [2] and *RStudio* [3] to open the **"rule605_analysis.Rmd"** file in the **"./rule605/analysis" directory**, then clicking on the *"Knit"* button in the upper left.

The Data

We're using Form 605 reports to compare firms' execution quality because the reports are the only data that everyone files in comparable format. Indeed, firms use those data to benchmark themselves against the competition.

Unfortunately, many firms only leave the trailing three months' reports accessible to the public. So we're performing our analysis on the latest three months' reports. You'll see that we've placed the unaltered raw data files of your last three reports in the directory **"./rule605/data/f605_data"**, decompressing them first if necessary. We combine them into a single data frame (i.e., table) for analysis with the script you'll find in the **"form605_merge_data.R"** file, which is in the **"./rule605/data/gather_source"** directory.

Only two order types make sense to analyze for price-improvement: *market orders* (which Form 605 codes as **"11"** and we've renamed **"mkt_ordr"**) and *marketable limit orders* (which Form 605 codes as **"12"** and we've renamed **"mktbl_lmt_ordr"**). The merge script filters out other order types, as well as any rows reporting on a stock in which the market center executed no orders in the period reported.

Because we want to control our analysis for whether stocks are listed on the NYSE or Nasdaq, or indexed in the S&P 500 or Russell 1000, you'll find membership lists of those stocks in the directory **"./rule605/data/constituent_data"**. The merge script adds columns to our data frame to indicate whether a stock shown in the Form 605 is listed on an exchange or belongs to an index.

The data frame resulting from the merge script is called **f605_mktble_df**.

Analysis

This comparison zeros in on *net price-improvement*. We calculate all our measures with the script you'll find in the file **"form605_analysis.R"**, which is in the directory **"./rule605/analysis"**. If you believe we should focus on other measures, please suggest them.

The merge script in the file **"form605_merge_data.R"** reads the raw form 605 filings into a data frame table. Here are the column headings that correspond to each form 605 field.

Field Number	Field Description	My Table's Column Name
F1	Designated Participant	"participant"
F2	Market Center code	"market_center"
F3	Month and Year	"date"
F4	Ticker Symbol	"ticker"
F5	Order Type	"order_type"
F6	Order Size	"order_size"
F7	Total Covered Orders	"total_orders"
F8	Total Covered Shares	"total_shrs"
F9	Cancelled Shares	"cancelled_shrs"
F10	Market Center Executed Shares	"mc_exec_shrs"
F11	Away Executed Shares	"away_exec_shrs"
F12	Shares from 0 to 9 Seconds	"shrs_0to9sec"
F13	Shares from 10 to 29 Seconds	"shrs_10to29sec"
F14	Shares from 30 to 59 Seconds	"shrs_30to59sec"
F15	Shares from 60 to 299 Seconds	"shrs_60to299sec"

F16	Shares from 5 to 30 Minutes	“shrs_5to30min”
F17	Average Realized Spread	“avg_realzd_spread”
F18	Average Effective Spread	“avg_effec_spread”
F19	Price Improved Shares	“px_improved_shrs”
F20	Price Improved Avg Amount	“px_improved_avg_amt”
F21	Price Improved Avg Time (Secs)	“px_improved_avg_secs”
F22	At-the-Quote Shares	“at_quote_shrs”
F23	At-the-Quote Avg Time (Secs)	“at_quote_avg_secs”
F24	Outside-the-Quote Shares	“outside_quote_shrs”
F25	Outside-the-Quote Avg Amt ($)	“outside_quote_avg_amt”
F26	Outside-Quote Avg Time (secs)	“outside_quote_avg_sec”

In addition to the columns from the Form 605s, I’ve added columns that note a stock’s membership in the S&P 500 or the Russell 1000, and columns for listings on the NYSE, Nasdaq or Amex.

I’ve tried to name the columns in ways you’ll understand, but if you are confused, contact me. The raw Form 605s use arcane labels to code for the size of orders, so I re-coded the levels of the *order size* variable as: “100-499”, “500-1999”, “200-4999” and “5000+”. I re-coded the two levels that we’re using in the *order type* variable as: “mkt_ordr” and “mktbl_lmt_ordr”.

The row structure of the *f605_mktble_df* data frame reflects that of the Form 605, where each row summarizes the month’s orders for a particular stock, with a new row for every combination of five order types and four order sizes–combinations which can generate up to 20 rows per ticker. Here are a few rows and columns from the data frame:

	market_center	date	ticker	order_type	order_size	avg_realzd_spread	avg_effec_spread	px_improved_shrs	px_improved_avg_amt
1	TCDRG	201408	A	mkt_ordr	100-499	-0.0109	0.0067	50958.0000	0.0058
2	TCDRG	201408	A	mkt_ordr	500-1999	-0.0007	0.0053	53645.0000	0.0038
3	TCDRG	201408	A	mkt_ordr	2000-4999	-0.0319	0.0206	15327.0000	0.0012
4	TCDRG	201408	A	mkt_ordr	5000+	-0.1068	0.0236	9671.0000	0.0036
5	TCDRG	201408	A	mktbl_lmt_ordr	100-499	-0.0005	0.0154	24935.0000	0.0010
6	TCDRG	201408	A	mktbl_lmt_ordr	500-1999	0.0085	0.0139	16495.0000	0.0020
7	TCDRG	201408	A	mktbl_lmt_ordr	2000-4999	0.0831	0.0201	8200.0000	0.0002
8	TCDRG	201408	A	mktbl_lmt_ordr	5000+	-0.1606	0.0099	3600.0000	0.0001
9	TCDRG	201408	AA	mkt_ordr	100-499	0.0040	0.0042	580550.0000	0.0031
10	TCDRG	201408	AA	mkt_ordr	500-1999	0.0043	0.0038	1193196.0000	0.0034

Some Rows and Columns from the f605_mktble_df data frame

Net Price-Improvement

Net price-improvement (or net p-i) is one of more useful measures you can calculate from Form 605 data. For each stock, I define net p-i as the per-share price improvement minus the average outside-the-quote dis-improvement, measured in cents. A better measure results if we weight the results for each table row by the number of shares involved, so that improvement on a bunch of shares doesn’t get overly offset by dis-improvement on just a few shares. Shares executed at-the-quote zero out of the result.

The formula for the numerator of the calculation looks like this:

> 100 * ((px_improved_shrs * px_improved_avg_amt) + (at_quote_shrs * 0) - (outside_quote_shrs * outside_quote_avg_amt))

Multiplying by 100 gives us a result in pennies, not dollars. Each row of the Form 605 reports these values for a combination of order type and size in a particular stock during the month. **Since we’re given values at the row-level, we should perform a weighting for each row and then roll them up for a grand average**.

The most appropriate denominator for each row’s weighting calculation is the row’s value for Market Center Executed Shares:

> “mc_exec_shrs”

Dividing the numerator by this denominator gives us the share-weighted mean of the per-share price-improvements and dis-improvements, for all that period’s transactions in a stock within a particular order-type or -size. Because we’re doing a row-wise calculation, each ticker can yield up to 20 of these weighted averages. I put these in a new column:

> “net_pi_mc”.

Here’s what those input and output columns look like, for a few rows in the table:

	market_center	date	ticker	order_type	order_size	total_shrs	cancelled_shrs	mc_exec_shrs	away_exec_shrs	px_improved_shrs	px_improved_

1	TCDRG	201408	A	mkt_ordr	100-499	64035.0000	5977.0000	58058.0000	0.0000	50958.0000	
2	TCDRG	201408	A	mkt_ordr	500-1999	61826.0000	0.0000	61826.0000	0.0000	53645.0000	
3	TCDRG	201408	A	mkt_ordr	2000-4999	25662.0000	0.0000	25662.0000	0.0000	15327.0000	
4	TCDRG	201408	A	mkt_ordr	5000+	15961.0000	0.0000	15961.0000	0.0000	9671.0000	
5	TCDRG	201408	A	mktbl_lmt_ordr	100-499	1047302.0000	782836.0000	264150.0000	0.0000	24935.0000	
6	TCDRG	201408	A	mktbl_lmt_ordr	500-1999	563671.0000	487808.0000	73968.0000	0.0000	16495.0000	
7	TCDRG	201408	A	mktbl_lmt_ordr	2000-4999	516712.0000	497269.0000	16265.0000	0.0000	8200.0000	
8	TCDRG	201408	A	mktbl_lmt_ordr	5000+	394133.0000	385204.0000	8929.0000	0.0000	3600.0000	
9	TCDRG	201408	AA	mkt_ordr	100-499	637370.0000	3900.0000	633470.0000	0.0000	580550.0000	
10	TCDRG	201408	AA	mkt_ordr	500-1999	1315216.0000	1000.0000	1314216.0000	0.0000	1193196.0000	

The Columns Used to Calculate Net Price-Improvement

Alternative share-weighting denominators for net price-improvement

Some market makers say they count away-executed shares in their net price-improvement reports, so I tried a couple of other denominators, as well. Dividing by the denominator **(mc_exec_shrs + away_exec_shrs)** yields a measure I call **“net_pi_mc_away”**, while dividing by **(total_shrs - cancelled_shrs)** yields a measure I call **“net_pi_mc_tot”**.

Each of these measures of net price-improvement gets its own column:

	market_center	date	ticker	order_type	order_size	net_pi_numerator	net_pi_mc	net_pi_mc_away	net_pi_mc_tot
1	TCDRG	201408	A	mkt_ordr	100-499	29362.9200	0.5058	0.5058	0.5058
2	TCDRG	201408	A	mkt_ordr	500-1999	20385.1000	0.3297	0.3297	0.3297
3	TCDRG	201408	A	mkt_ordr	2000-4999	364.2400	0.0142	0.0142	0.0142
4	TCDRG	201408	A	mkt_ordr	5000+	-1729.4400	-0.1084	-0.1084	-0.1084
5	TCDRG	201408	A	mktbl_lmt_ordr	100-499	1670.5000	0.0063	0.0063	0.0063
6	TCDRG	201408	A	mktbl_lmt_ordr	500-1999	2799.0000	0.0378	0.0378	0.0369
7	TCDRG	201408	A	mktbl_lmt_ordr	2000-4999	164.0000	0.0101	0.0101	0.0084
8	TCDRG	201408	A	mktbl_lmt_ordr	5000+	36.0000	0.0040	0.0040	0.0040
9	TCDRG	201408	AA	mkt_ordr	100-499	179905.5000	0.2840	0.2840	0.2840
10	TCDRG	201408	AA	mkt_ordr	500-1999	401606.6400	0.3056	0.3056	0.3056

Row measures of net price-improvement

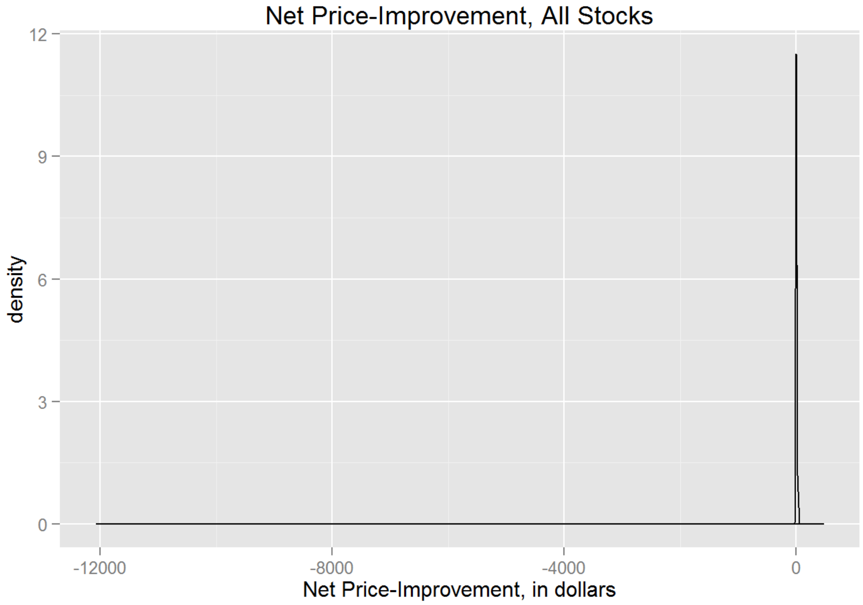
The rows’ weighted averages can then be averaged for all stocks during the period, or examined for such subsets as Russell 1000 stocks or orders above 5,000 shares.

In the discussion below, unless otherwise stated, **we’ll use the first-mentioned share-weighting measure, “net_pi_mc”**, calculated like so:

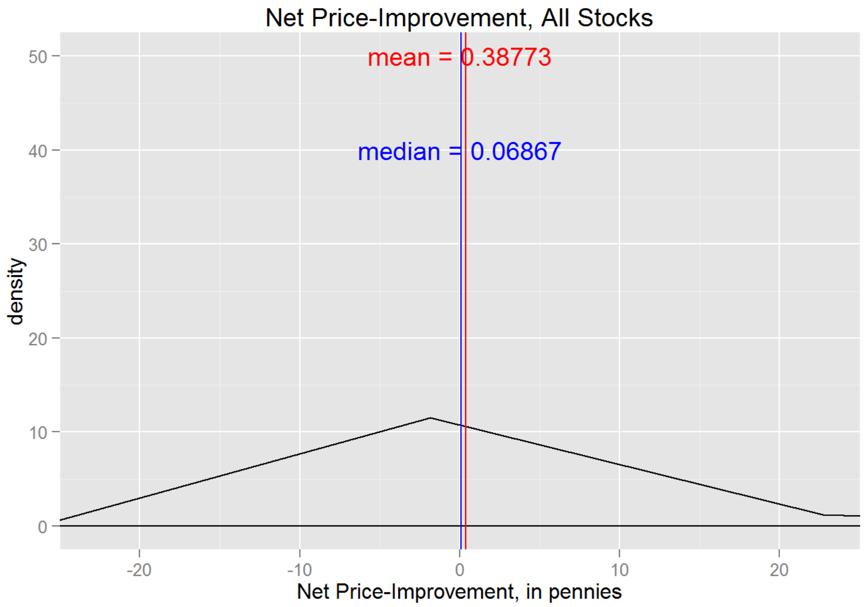
> 100 * ((px_improved_shrs * px_improved_avg_amt) + (at_quote_shrs * 0) - (outside_quote_shrs * outside_quote_avg_amt)) / mc_exec_shrs

Mean or Median?

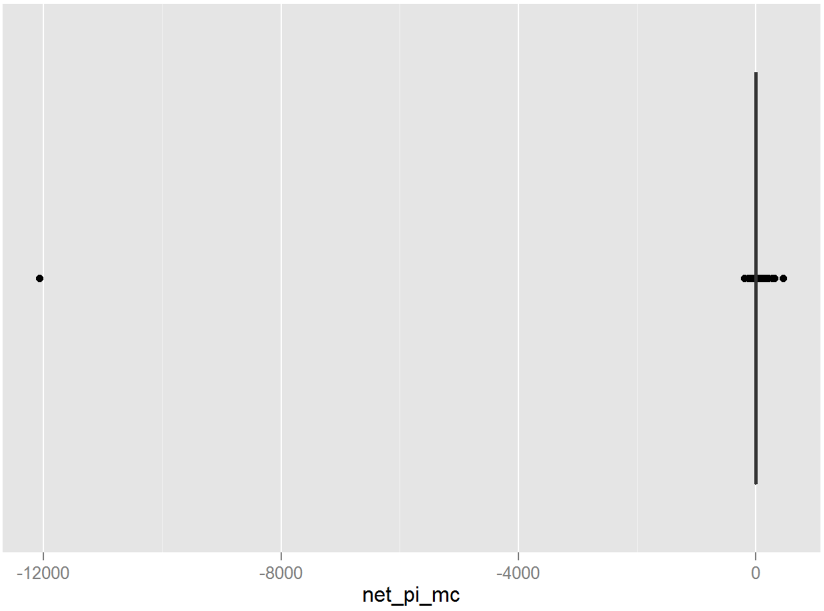
A problem with averaging a measure like net price-improvement across stocks is that the *mean* can be overly-influenced by a stock with a nominally-large share prices and spreads, e.g. Priceline or NVR, whose shares have topped $1,000 apiece. Data entry errors can also influence a mean. The graphic, below, of the distribution of net p-i across the stocks in your Form 605 reports allows you to see such “outliers”.



Zooming in on the majority of the stocks, where net price-improvement is under a penny, enables you to better see how the measure looks across stocks.



A boxplot shows outliers more clearly:



These are the 15 extreme values at the bottom and top of your net price improvement:

	market_center	date	ticker	order_type	order_size	net_pi_mc
1	TCDRG	201410	SEB	mkt_ordr	100-499	-12067.0314
2	TCDRG	201409	SEB	mktbl_lmt_ordr	100-499	-194.2264
3	TCDRG	201410	DFVS	mkt_ordr	2000-4999	-128.9502
4	TCDRG	201408	DJCO	mktbl_lmt_ordr	500-1999	-107.9074
5	TCDRG	201410	FORTY	mkt_ordr	500-1999	-104.5669
6	TCDRG	201410	MHR E	mkt_ordr	2000-4999	-94.4460
7	TCDRG	201410	BKEPP	mkt_ordr	2000-4999	-92.2641
8	TCDRG	201409	ISRL	mkt_ordr	100-499	-86.7850
9	TCDRG	201408	MYOS	mkt_ordr	2000-4999	-76.6070
10	TCDRG	201410	ASRVP	mkt_ordr	5000+	-74.8660

Bottom extreme values of net price-improvement

	market_center	date	ticker	order_type	order_size	net_pi_mc
146754	TCDRG	201408	ALX	mkt_ordr	100-499	123.1000
146755	TCDRG	201408	NVR	mkt_ordr	100-499	127.8700
146756	TCDRG	201409	CCZ	mktbl_lmt_ordr	100-499	128.1400
146757	TCDRG	201408	DHIL	mkt_ordr	500-1999	138.0000
146758	TCDRG	201408	AAMC	mkt_ordr	100-499	138.3111
146759	TCDRG	201410	MAYS	mkt_ordr	100-499	161.0500

146760	TCDRG	201409	GRID	mkt_ordr	500-1999	168.0000
146761	TCDRG	201408	VBFC	mkt_ordr	100-499	168.6800
146762	TCDRG	201408	NVR	mktbl_lmt_ordr	2000-4999	174.2701
146763	TCDRG	201410	ATRI	mkt_ordr	100-499	178.0000

Top extreme values of net price-improvement

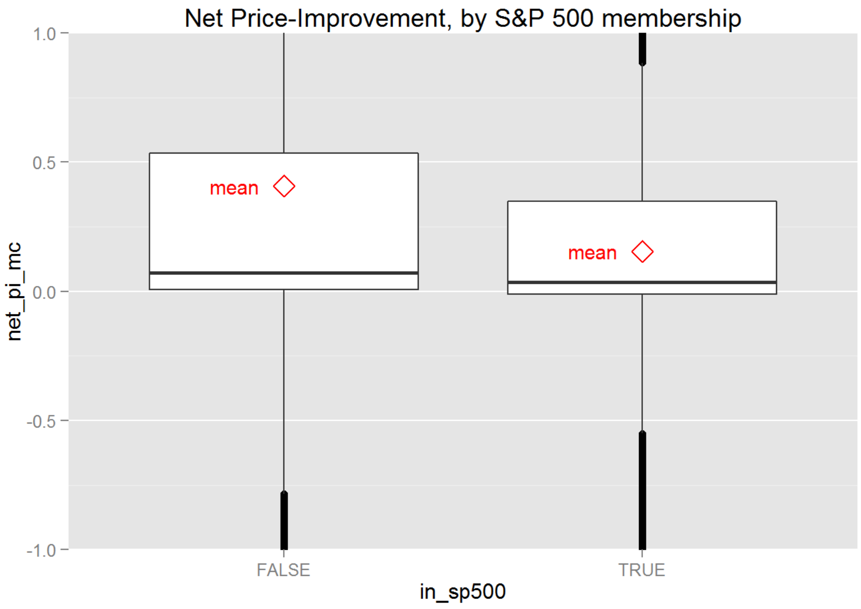
Please glance at these values to alert me to any that seem to be data-entry errors in the raw Form 605.

Because of the possible presence of extreme outliers, I wonder if the median is a more robust summary measure than the mean, for showing our readers what sort of price-improvement they can expect on a typical order ? One can argue that some kind of trimmed-mean could exclude a few extreme outliers, but such decisions can raise suspicions of cherry-picking.

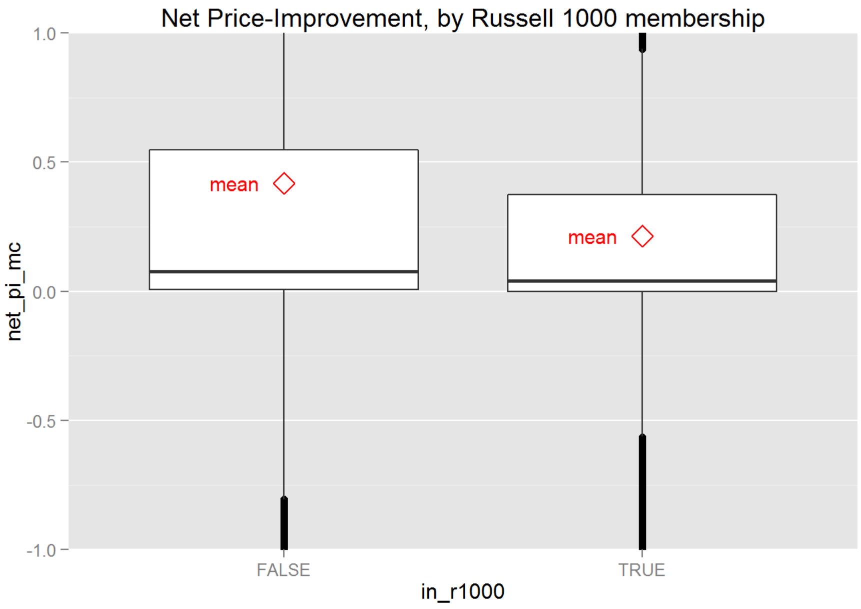
Comparing Categories

We can see whether the median net price-improvement is much affected by a stock’s exchange listing or its size (for which, we use the proxy of its membership in the S&P 500 or the Russell 1000).

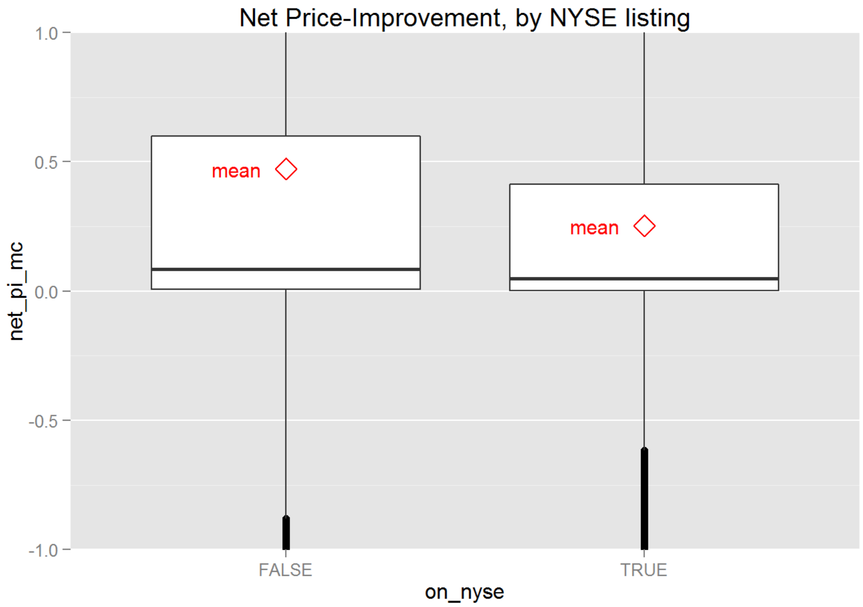
Here, I compare S&P 500 members vs. non-members. The horizontal line across the middle of the boxplots shows the median net-pi for each group, and I mark the mean with a red diamond:



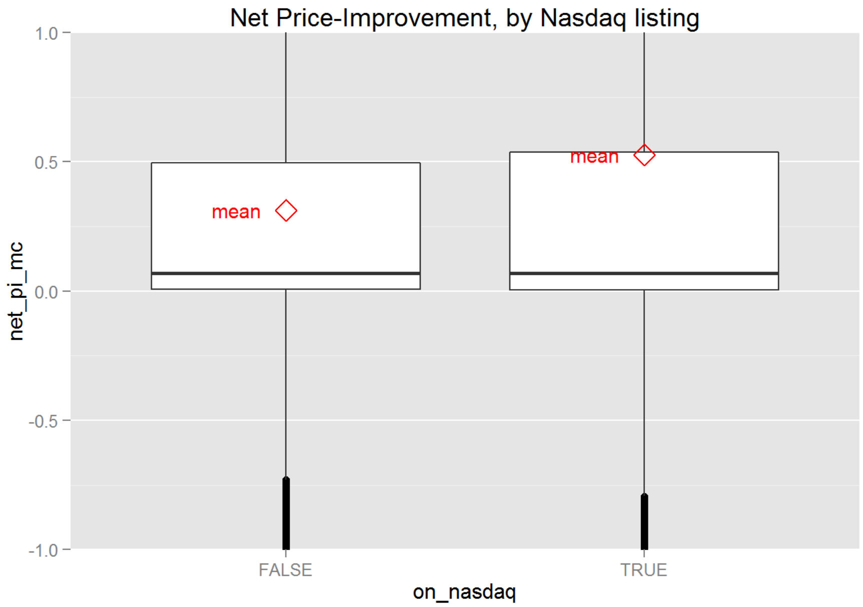
Here’s the net p-i comparison for Russell 1000 membership, which many folks consider a more objective and comprehensive way of defining large-cap stocks:



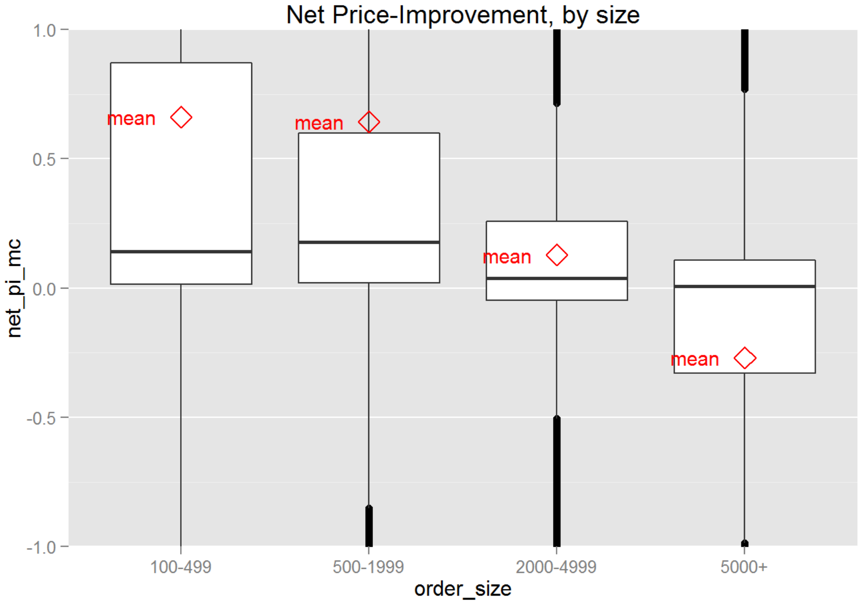
Here’s the net p-i for NYSE listing:



And for Nasdaq listing:

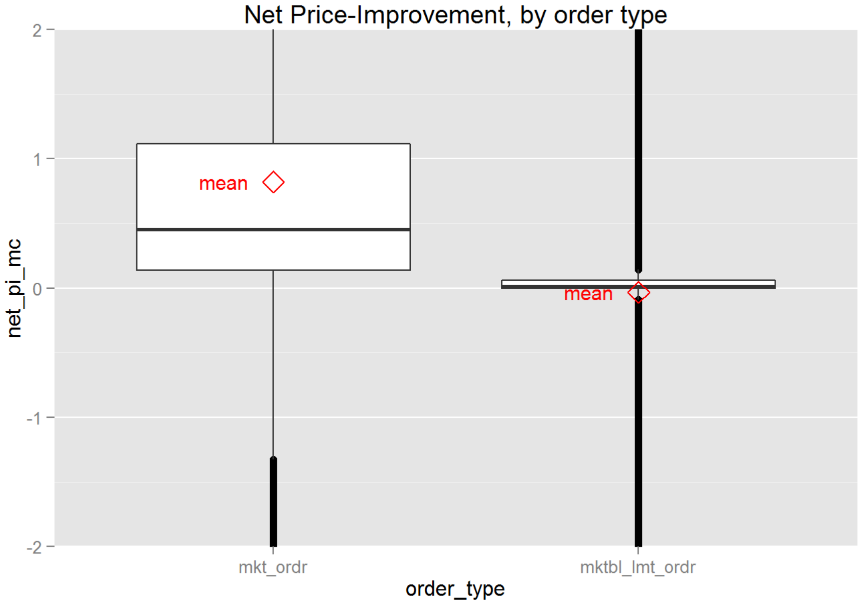


We can also compare the affects of order-size. The boxplots below show there are size differences. I'm told that the typical retail order falls in the 500-to-1999 bracket, but I'm inclined to use the average across all order sizes, to avoid suspicions of cherry-picking:



Order Type

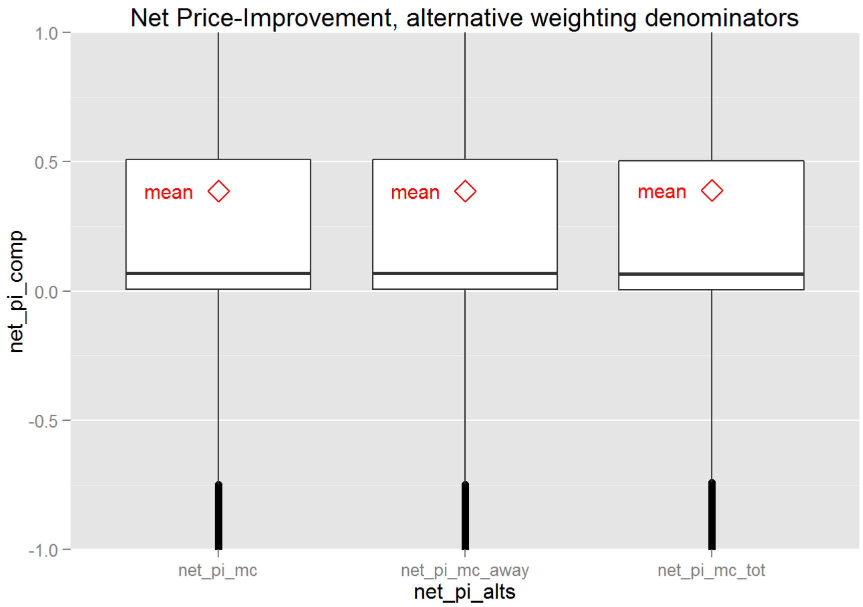
We initially filtered the raw Form 605 data to retain only the two order types that make sense to study for price improvement: market orders and marketable limit orders. Let's see how the two order types differ in their average net price-improvement:



Price improvement seems to differ dramatically between market orders and marketable limit orders. That makes me wonder if we should use a measure that combines the two order types, or perhaps **just use the price improvement on market orders**.

Alternative weighting schemes

Some firms apparently count all executions, those at the market center and away, in their price improvement measures. Others only count market center executions. This may have ramifications for the weighting formula. So here, I compare a measure of price improvement with various denominators:



The Net Price-Improvement Measures We're Thinking of Using

The **median or mean of the share-weighted net price-improvement per share, in all stocks and order sizes** seem to be good measures for comparing market centers.

Here are the results if you count both market orders and marketable limit orders. Based on the acccompanying data, your firm's median was 0.06867 cents per share and the mean was 0.38773 cents per share.

And counting just market orders, instead of also counting marketable limit orders, those measures are median 0.45246 cents per share and mean 0.82113 cents per share.

Reproducing this analysis

You can reproduce this analysis by using RStudio to open the file **"form605_merge_data.R"**, which you'll find in the directory **"./data/gather_source"**. Click on the "Source" button you'll see to the upper right of the file window and the script will automatically run through the data-cleaning and analysis that I performed. It will also save MS Excel-readable files with the calculated numbers, in the directory **".analysis/results_data"**, including a large spreadsheet file that has the Form 605 data that's been cleaned up and formatted.

We've undertaken the effort to give you these "replication files" so that you can see exactly what we've done, so please take a look at our work and critique it.

Thanks!

Bill Alpert 212.416.2742 (w) or 201.9223.8214 (m) william.alpert@barrons.com (mailto:william.alpert@barrons.com)

```
sessionInfo()
```

```
## R version 3.1.2 (2014-10-31)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## 
## locale:
## [1] LC_COLLATE=English_United States.1252 
## [2] LC_CTYPE=English_United States.1252   
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C                          
## [5] LC_TIME=English_United States.1252    
## 
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods  
## [8] base     
## 
## other attached packages:
##  [1] reshape2_1.4.1    car_2.0-22        digest_0.6.7     
##  [4] xtable_1.7-4      magrittr_1.5      tidyr_0.2.0      
##  [7] dplyr_0.3.0.2     knitr_1.8         rmarkdown_0.4.2  
## [10] easyGgplot2_1.0.0 plyr_1.8.1        ggplot2_1.0.0    
## [13] devtools_1.6.1   
## 
## loaded via a namespace (and not attached):
##  [1] assertthat_0.1   bitops_1.0-6     colorspace_1.2-4 DBI_0.3.1       
##  [5] evaluate_0.5.5   formatR_1.0      gtable_0.1.2     htmltools_0.2.6 
##  [9] httr_0.6.0       labeling_0.3     lazyeval_0.1.9   MASS_7.3-35     
## [13] munsell_0.4.2    nnet_7.3-8       parallel_3.1.2   proto_0.3-10    
## [17] Rcpp_0.11.3      RCurl_1.95-4.5   scales_0.2.4     stringr_0.6.2   
## [21] tools_3.1.2      yaml_2.1.13
```

1. https://rmarkdown.rstudio.com (https://rmarkdown.rstudio.com)↩
2. https://www.r-project.org (https://www.r-project.org)↩
3. https://www.rstudio.com (https://www.rstudio.com)↩