

101公式alpha

祖拉Kakushadze^{§ †1}

[§] Quantigic® Solutions LLC,² 1127 High Ridge Road, #135, Stamford, CT 06905

[†] 第比利斯自由大学,商学院和物理学院240,David Agmashene
beli Alley,第比利斯,0159,格鲁吉亚

2015.12.9

“这个世界上有两种人:追求幸福的人和斗牛士。”

(加州祖拉Kakushadze, 90年代初)

摘要

我们提出了101个现实生活中的量化交易alpha的显式公式(也是计算机代码)。它们的平均持有期大约在0.6-6.4天之间。这些alpha的平均成对相关性很低, 为15.9%。收益率与波动率有很强的相关性, 但对周转率没有显著的依赖性, 直接证实了基于更间接的实证分析的早期结果。我们进一步从经验上发现, 换手率对alpha相关性的解释力较差。

¹ Zura Kakushadze, Ph.D., is the President and a Co-Founder of Quantigic® Solutions LLC and a Full Professor in the Business School and the School of Physics at Free University of Tbilisi. Email: zura@quantigic.com

² DISCLAIMER: This address is used by the corresponding author for no purpose other than to indicate his professional affiliation as is customary in publications. In particular, the contents of this paper are not intended as an investment, legal, tax or any other such advice, and in no way represent views of Quantigic® Solutions LLC, the website www.quantigic.com or any of their other affiliates.

³ Paraphrasing *Blondie's* (Clint Eastwood) one-liners from a great 1966 motion picture *The Good, the Bad and the Ugly* (directed by Sergio Leone).

1. 简介

在现代定量交易中有两种互补的——在某种意义上甚至是相互竞争的——趋势。一方面，越来越多的市场参与者(例如，定量交易者等)采用复杂的定量技术来挖掘阿尔法。⁴这导致了更微弱和更短暂的阿尔法。另一方面，技术进步允许基本上自动化(大部分)阿尔法收获过程。这就产生了越来越多的阿尔法，其数量可以达到数十万甚至数百万，随着这一领域的指数级增长，在我们意识到之前，它可能会达到数十亿……

这种阿尔法的扩散——尽管大多是微弱和短暂的——允许以一种复杂的方式将它们组合起来，以达到统一的“巨型阿尔法”。然后，实际交易的是这个“超级阿尔法”——而不是交易单个阿尔法——附带的奖励是交易的自动内部交叉(因此对节省交易成本等盈利至关重要)、阿尔法投资组合多样化(在任何给定时间段内对冲阿尔法投资组合的任何子集破产)，等等。组合阿尔法的挑战之一是常见的“变量太多，观察值太少”的困境。因此，alpha样本协方差矩阵具有严重的奇异性。

同样，自然地，量化交易是一个保密的领域，从业人员的数据和其他信息不容易获得。这无意中给现代量化交易制造了一个谜。例如，有如此大量的alpha，它们之间不是高度相关的吗？这些alpha是什么样的呢？它们主要是基于价格和成交量数据、均值回归、动量等吗？阿尔法收益如何取决于波动性、换手率等？

在之前的一篇论文中[Kakushadze and Tulchinsky, 2015]通过研究4000个现实生活中的alpha的一些经验属性，揭开了现代量化交易领域的神秘面纱。在这篇论文中，我们又迈出了一步，为101个现实生活中的量化交易阿尔法提供了显式公式——也是计算机代码。我们的公式化alpha(尽管大多数不一定都那么“简单”)的目的是让读者了解一些更简单的现实生活alpha是什么样子的。⁵它还使读者能够在历史数据上复制和测试这些alpha，并进行新的研究和其他实证分析。希望它能进一步激发(年轻的)研究人员提出新的想法，创造自己的阿尔法。

⁴ “An alpha is a combination of mathematical expressions, computer source code, and configuration parameters that can be used, in combination with historical data, to make predictions about future movements of various financial instruments.” [Tulchinsky *et al*, 2015] Here “alpha” –following the common trader lingo –generally means any reasonable “expected return” that one may wish to trade on and is not necessarily the same as the “academic” alpha. In practice, often the detailed information about how alphas are constructed may even not be available, e.g., the only data available could be the position data, so “alpha” then is a set of instructions to achieve certain stock (or other instrument) holdings by some times , , …(e.g., a tickers by holdings matrix for each).

⁵ We picked these alphas largely based on simplicity considerations, so they can be presented within the inherent limitations of a paper. There also exist myriad other, “non-formulaic” (coded and too-complex-to-present) alphas.

我们将在第2节中讨论我们的公式化alpha的一些一般特征。这些alpha主要是基于“价格-量”(每日收盘价回报, 开盘价, 收盘价, 高点, 低点, 成交量和vwap)的, 尽管在一些alpha中使用了“基本”输入, 包括一个利用市值的alpha, 以及一些采用某种二元行业分类的alpha, 如GICS, BICS, NAICS, SIC等。它们用于行业中和各种数量。⁶

与我们一起讨论波动性经验, 以及[我们的alpha和Tulchinsky的Kakushadze, 在2015年第3节]中的属性, 基于我们发现的数据, 单个缩放alpha夏普比率, 营业额和每股美分的经验, 以及样本协方差矩阵。平均持有期约为0.6至6.4天。这些alpha的平均(中位数)成对相关性较低, 为15.9%(14.3%)。收益率是强相关的

$$R \sim V^X \quad (1)$$

对于我们的101 alpha, ≈ 0.76 。进一步, 我们发现收益率对周转率没有显著的依赖性。这直接证实了[Kakushadze and Tulchinsky, 2015]基于更间接的实证分析得出的早期结果。⁷

我们进一步从经验上发现, 离职本身对alpha相关性的解释力很差。这并不是说周转不会增加价值, 例如, 通过因子模型对协方差矩阵建模。一个更精确的说法是成对相关的alpha (, 精=1, \cdots , 标记alpha, \neq)并不是与乘积 $\ln(s:2) \ln(e:2)$ (以)高度相关, 其中的“=”是“=”和“/”是一个先验的任意归一化常数。⁹

我们在第4节中简要地总结。附录A包含我们的公式化alpha, 以及其中使用的函数、运算符和输入数据的定义。附录B包含一些法律术语。

2. Formulaic alpha

在本节中, 我们将描述我们的101个公式化alpha的一些一般特征。alpha是WorldQuant LLC专有的, 在此使用时已获得其明确许可。在alpha的专有性质所施加的限制下, 我们尽可能多地提供细节。公式化表达式(也是计算机代码)在附录A中给出。

⁶ More precisely, depending on the alpha and industry classification used, neutralization can be w. r. t. sectors, industries, subindustries, etc. –different classifications use different nomenclature for levels of similar granularity.

⁷ In [Kakushadze and Tulchinsky, 2015] the alpha return volatility was not directly available and was estimated indirectly based on the Sharpe ratio, cents-per-share and turnover data. Here we use direct realized volatility data.

⁸ Depending on a construction, a priori the turnover might add value via the specific (idiosyncratic) risk for alphas.

⁹ Here we use log of the turnover as opposed to the turnover itself as the latter has a skewed, roughly log-normal distribution, while pair-wise correlations take values in $(-1, 1)$ (in fact, their distribution is tighter –see below).

粗略地说，我们可以把 α 信号看作是基于均值回归或动量的信号。¹⁰均值回归的alpha信号的符号与它所依据的回报相反。例如，一个简单的均值回归alpha由

$$-\ln(\text{today's open} / \text{yesterday's close}) \quad (2)$$

在这里，如果除息日是今天，昨天的收盘价将根据任何拆分和股息进行调整。这里的想法(或希望)是股票将均值回归并收回部分收益(如果今天的开盘价高于昨天的收盘价)或弥补部分损失(如果今天的开盘价低于昨天的收盘价)。这就是所谓的“延迟-0”alpha。一般来说，“delay-0”是指alpha中使用的某些数据(例如价格)的时间与alpha打算交易的时间一致。例如，理想情况下alpha(2)将在或者更现实的情况下，尽可能接近今天的开盘时间进行交易。更广泛地说，这可以是其他时间，例如，收盘时间¹¹

关于动量alpha的一个简单例子是

$$\ln(\text{yesterday's close} / \text{yesterday's open}) \quad (3)$$

在这里，如果价格数字调整了几天或没有调整，它同样与2计数没有区别。这个想法数据(或使用的希望)是样本外的。这里的意思是，如果股票昨天涨(跌)，今天的趋势会继续，收益(损失)会进一步增加。这就是所谓的“延迟-1”阿尔法，如果你的意图是今天交易(例如，从开盘开始)。¹²一般来说，“delay-1”意味着alpha在计算它时使用的最新数据的日期之后的第二天进行交易。“delay-2”的alpha是

在复杂的alpha中，均值回归和动量的元素可以混合在一起，使它们在这方面不那么明显。然而，人们可以将这些alpha的较小构建块视为基于均值回归或动量。例如，附录A中的Alpha#101是一个延迟-1动量Alpha:如果股票在日内上涨(即收盘价>开盘，高点>低点)，第二天就会持有该股票的多头头寸。另一方面，附录A中的Alpha#42本质上是一个延迟-0均值回归Alpha:如果股票在下半日上涨(收盘价> vwap)¹³而不是下跌(收盘价< vwap)，则排名(vwap -close)较低。分母权重较低的是较富裕的股票。“反向”头寸是在接近收盘时采取的。

¹⁰ On longer horizons, for a discussion of mean-reversion (contrarian) and momentum (trend following) strategies, see, e.g., [Avellanida and Lee, 2010] and [Jegadeesh and Titman, 1993], respectively, and references therein.

¹¹ Four of our 101 alphas in Appendix A, namely, the alphas numbered 42, 48, 53 and 54, are delay-0 alphas. They are assumed to be traded at or as close as possible to the close of the trading day for which they are computed.

¹² On the other hand, if the alpha (3) is executed as close as possible to yesterday's close, then it is delay-0.

¹³ Here “vwap”，as usual, stands for “volume-weighted average price” .

3. alpha的数据和经验性质

在本节中，我们描述了基于WorldQuant LLC专有数据的公式化alpha的经验属性，在此使用已获得其明确许可。我们在此数据集专有性质的约束下提供了尽可能多的细节。

对于我们的alpha，我们采用年化的每日夏普比率 γ ，每日营业额和每股美分 δ 。让我们用指数($= 1, \dots, 101$)标记我们的阿尔法指数，其中 $= 101$ 是阿尔法指数的数量。对于每个alpha，通过定义3个，和4个

$$S_i = \sqrt{252} \frac{P_i}{V_i} \quad (4)$$

$$T_i = \frac{D_i}{I_i} \quad (5)$$

$$C_i = 100 \frac{P_i}{Q_i} \quad (6)$$

这里 γ 是平均每日损益(以美元计); V 为每日投资组合波动率; P_i 是 i -th alpha的日均股票交易量(买入加卖出); Q_i 是日均美元交易量;和 C_i 是所述alpha的总美元投资(实际多头加空头头寸，不含杠杆)。更准确地说， P_i 的本金是恒定的;但是， C_i 因每日损益而波动。因此，在式(4)中对 Q_i 和 C_i 进行了相应的调整(使 C_i 为常数)。收集该数据的时间段为2010年1月4日至2013年12月31日。对于同一时期，我们也取alpha的已实现日收益的样本协方差矩阵 Σ 。时间序列中的观测数为1006个， Σ 是非奇异的。从 Σ 我们读出日收益波动率 $\sigma^2 = \Sigma_{11}$ 和相关矩阵 $\rho = \Sigma^{-1}_{11}$ (其中 $\rho_{11} = 1$)，注意 $\rho_{11} = 1$ ，平均14日收益由 $\mu = \Sigma_{11}^{-1} \mu$ 给出。

表1和图1总结了年化夏普比率 γ ，日成交额 V ，平均持有期 T ，每股美分 δ ，日收益波动率 σ^2 ，年化平均日收益 μ ，和 ρ 对相关关系与 γ 。

3.1. 回报vs波动率&周转率

我们运行两个横截面回归，都具有截距， $\ln(V) / T$ 和 $\ln(\delta)$ 作为唯一的解释变量， $\ln(\rho_{11}) / T$ 和 $\ln(\gamma)$ 。结果总结在表2和表3中。与[Kakushadze and Tulchinsky, 2015]一致，我们没有统计

¹⁴ Here the average is over the time series of the realized daily returns.

这里显著依赖于周转率，而平均日收益与日收益波动率=强相关，我们具有缩放性质(1)， ≈ 0.76 。

3.2. 周转率解释相关性吗？

如果我们在阿尔法指数和股票之间进行类比，那么阿尔法指数的周转率就类似于股票的流动性，这通常是通过平均每日美元交易量(ADDV)来衡量的。¹⁵在多因素风险模型¹⁷中，通常使用Log of ADDV作为风格风险因子¹⁶来近似股票投资组合协方差矩阵结构，其主要目标是建模协方差矩阵的非对角线元素，即成对相关结构。¹⁸按照这个类比，我们可以问周转率——或者更准确地说，它的对数——是否对建模alpha相关性具有解释力。¹⁹很明显，由于高度倾斜(大致对数正态)的营业额分布(见图1)，直接使用营业额(而不是它的对数)将使我们无处可去。

要回答这个问题，回想一下，在因子模型中，协方差矩阵是通过

$$\Gamma_{ij} = \xi_i^2 \delta_{ij} + \sum_{A,B=1}^K \Omega_{iA} \varphi_{AB} \Omega_{jB} \quad (7)$$

这里:A是具体风险;EF是与L \ll 危险因素相对应的 $\times L$ 因子加载矩阵;JFH为因子协方差矩阵。在我们的案例中，我们感兴趣的是对相关矩阵进行建模，并确定营业额是否对成对相关具有解释力。波动率和换手率是否相关是另一个问题。

因此，我们的方法是将因子加载矩阵的其中一列取为ln()。更准确地说，先验地说，我们没有理由选择ln()而不是ln()'，其中'是某个归一化因子。为了处理这个问题，让我们对进行归一化，使ln()的横截面均值为零，并设N = 1为单位-向量(截距)。然后我们可以构造三个对称张量组合O = N N，P = N lnQR + N ln()'，S = ln()lnQR。现在让我们定义一个复合指数TUV = T(|>V，它取X = (-1)/2的值，也就是说，我们拉出一个一般的非对角线下三角元素

¹⁵ Perhaps a more precise analogy would be between the turnover and the ratio of ADDV and market cap; however, this is not going to be critical for our purposes here.

¹⁶ For liquidity as a style risk factor, see, e.g., [Pastor and Stambaugh, 2003] and references therein.

¹⁷ See, e.g., [Grinold and Kahn, 2000] and references therein.

¹⁸ Variances are relatively stable and can be computed based on historical data (sample variances). It is the off-diagonal elements of the sample covariance matrix –to wit, the correlations –that are out-of-sample unstable.

¹⁹ Log of the turnover as a factor for risk models for alpha portfolios was suggested in [Kakushadze, 2014].

对称矩阵Y变成一个向量YZ。这样我们就可以构造四个x向量Z, OZ, PZ和SZ。现在我们可以运行Z / OZ, PZ和SZ的线性回归。请注意, OZ = 1只是截距(单位x向量), 因此这是Z / PZ和SZ与截距的回归。结果总结在表4中。很明显, 线性和双线性(在 $\ln()$ 中)变量PZ和SZ对成对相关Z的解释能力很差, 而OZ(截距)只是对平均相关平均值(Z)进行建模。回想一下, 通过构造, PZ和SZ与OZ正交, 并且这三个解释变量彼此独立。

让我们强调 $\ln(=)$ 超过 $\ln()$, 我们的结论(与截距)并不一定显示非零平均相关性, 在因素模型上下文中, 营业额之间没有增加这些值, 它只意味着营业额本身似乎无助于对 α 相关性的建模。上述分析并没有解决周转是否增加了建模方差的解释价值, 例如, 特定风险。²⁰因此, 一个线性变量(见表5), 虽然不是很强。要查看周转是否通过, 例如, 特定风险增加价值, 需要使用本文范围之外的某些专有方法²¹

4. 结论

我们强调, 我们在这里展示的101个alpha不是“玩具”alpha, 而是在生产中使用的真实交易alpha。事实上, 在撰写本文时, 这些alpha中有80个已经投入生产。²²据我们所知, 这是文献中第一次出现如此大量的现实生活中的显式公式化alpha。这应该不足为奇:自然地, 量化交易是高度专有和保密的。我们在里的目标是让人们一窥现代和不断发展的量化交易的复杂世界, 并在任何可能的程度上帮助揭开它的神秘面纱。

如今的技术进步使阿尔法采矿自动化成为可能。量化交易阿尔法是迄今为止数量最多的可用交易信号, 可以转化为交易策略/投资组合。在一个(美元中性的)投资组合中, 个人股票持有有无数的排列, 例如, 2000只流动性最强的美国股票, 可以在高频和中频的时间范围内产生正回报。此外, 这些阿尔法中的许多都是短暂的, 它们的范围是非常不稳定的。要挖掘数十万、数百万甚至数十亿的阿尔法, 并将它们组合成一个统一的“巨型阿尔法”(mega-alpha), 然后进行交易, 由于交易的自动内部交叉, 还可以节省相当大的执行成本, 这是一个额外的好处。

²⁰ Suppressing alpha weights by the turnover can add value but be highly correlated with volatility suppression.

²¹ Roughly speaking, when the specific risk is computed via nontrivial (proprietary) methods, the column in the factor loadings matrix corresponding to the turnover is no longer proportional to $\ln()$ but is a more complex function of the turnover, the specific risk also depends on the turnover nontrivially and is not quadratic in $\ln()$.

²² For proprietary reasons, we are not at liberty to state precisely which ones.

本着这种精神，我们以1832年俄罗斯诗人米哈伊尔·莱蒙托夫(Mikhail Lermontov)的一首诗作为本文的结尾(由Zura Kakushadze翻译自俄语，约1993年):

《风帆》

孤独的帆似乎是白色的,在朦胧
的雾霾中,蓝色的海,是外国大
风寻求力量?它为何逃回家乡
的港湾?

帆的弯桅吱吱作响,风浪在前方呼啸,
它追求的不是幸福,它逃离的也不是
幸福!

下面是奔流的ázure小溪,上面是闪
耀的金光,但船帆似乎希望风暴,仿
佛在风暴中它认为是和平。

附录A:公式化的阿尔法

在本附录的小节A.1中，我们提供了101个公式化的alpha。一旦定义了函数和运算符，这些公式也是代码。alpha中使用的函数和运算符在第A.2小节中定义。第A.3款详细说明了输入数据。

A.1. alpha的公式表达式

Alpha#1:
$$\text{rank}(\text{Ts_ArgMax}(\text{SignedPower}(((\text{returns} < 0)) ? \text{Stddev}(\text{returns}, 20) : \text{close}), 2), 5)) - 0.5$$

Alpha#2:
$$(-1 * \text{correlation}(\text{rank}(\text{delta}(\log(\text{volume}), 2)), \text{rank}(((\text{close} - \text{open}) / \text{open})), 6))$$

Alpha#3:
$$(-1 * \text{correlation}(\text{rank}(\text{open}), \text{rank}(\text{volume}), 10))$$

Alpha#4:
$$(-1 * \text{Ts_Rank}(\text{rank}(\text{low}), 9))$$

Alpha#5:
$$(\text{rank}((\text{open} - (\text{sum}(\text{vwap}, 10) / 10))) * (-1 * \text{abs}(\text{rank}((\text{close} - \text{vwap}))))))$$

Alpha#6:
$$(-1 * \text{correlation}(\text{open}, \text{volume}, 10))$$

Alpha#7:
$$((\text{adv20} < \text{volume}) ? ((-1 * \text{ts_rank}(\text{abs}(\text{delta}(\text{close}, 7)), 60)) * \text{sign}(\text{delta}(\text{close}, 7))) : (-1 * 1))$$

Alpha#8: (-1 * rank(((sum(open, 5) * sum(returns, 5)) - delay((sum(open, 5) * sum(returns, 5)), 10))))

Alpha#9: ((0 < ts_min(delta(close, 1), 5)) ? delta(close, 1) : ((ts_max(delta(close, 1), 5) < 0) ? delta(close, 1) : (-1 * delta(close, 1))))

Alpha#10: rank(((0 < ts_min(delta(close, 1), 4)) ? delta(close, 1) : ((ts_max(delta(close, 1), 4) < 0) ? delta(close, 1) : (-1 * delta(close, 1)))))

Alpha#11: ((rank(ts_max((vwap - close), 3)) + rank(ts_min((vwap - close), 3))) * rank(delta(volume, 3)))

Alpha#12: (sign(delta(volume, 1)) * (-1 * delta(close, 1)))

Alpha#13: (-1 * rank(covariance(rank(close), rank(volume), 5)))

Alpha#14: ((-1 * rank(delta(returns, 3))) * correlation(open, volume, 10))

Alpha#15: (-1 * sum(rank(correlation(rank(high), rank(volume), 3)), 3))

Alpha#16: (-1 * rank(covariance(rank(high), rank(volume), 5)))

Alpha#17: (((-1 * rank(ts_rank(close, 10))) * rank(delta(delta(close, 1), 1))) * rank(ts_rank((volume / adv20), 5)))

Alpha#18: (-1 * rank(((stddev(abs((close - open)), 5) + (close - open)) + correlation(close, open, 10))))

Alpha#19: ((-1 * sign(((close - delay(close, 7)) + delta(close, 7)))) * (1 + rank((1 + sum(returns, 250)))))

Alpha#20: (((-1 * rank((open - delay(high, 1)))) * rank((open - delay(close, 1)))) * rank((open - delay(low, 1))))

Alpha#21: (((((sum(close, 8) / 8) + stddev(close, 8)) < (sum(close, 2) / 2)) ? (-1 * 1) : (((sum(close, 2) / 2) < ((sum(close, 8) / 8) - stddev(close, 8))) ? 1 : (((1 < (volume / adv20)) || ((volume / adv20) == 1)) ? 1 : (-1 * 1))))

Alpha#22: (-1 * (delta(correlation(high, volume, 5), 5) * rank(stddev(close, 20)))) Alpha#23: (((sum(high, 20) / 20) < high) ? (-1 * delta(high, 2)) : 0)

Alpha#24: (((((delta((sum(close, 100) / 100), 100) / delay(close, 100)) < 0.05) || ((delta((sum(close, 100) / 100), 100) / delay(close, 100)) == 0.05)) ? (-1 * (close - ts_min(close, 100))) : (-1 * delta(close, 3)))

Alpha#25: rank(((((-1 * returns) * adv20) * vwap) * (high - close)))
 Alpha#26: (-1 * ts_max(correlation(ts_rank(volume, 5), ts_rank(high, 5), 5), 3)) Alpha#27: ((0.5 * rank((sum(correlation(rank(volume), rank(vwap), 6), 2) / 2.0))) ? (-1 * 1) : 1) Alpha#28: scale(((correlation(adv20, low, 5) + (high + low) / 2)) - close))
Alpha#29: (min(product(rank(rank(scaling(log(sum(ts_min(rank(rank((-1 * rank(delta((close - 1), 5))))), 2), 1)))), 1), 5) + ts_rank(delay((-1 * returns), 6), 5))
 Alpha#30: (((1.0 - rank(((sign((close - delay(close, 1))) + sign((delay(close, 1) - delay(close, 2)))) + sign((delay(close, 2) - delay(close, 3)))))) * sum(volume, 5)) / sum(volume, 20)) Alpha#31: ((rank(rank(rank(decay_linear((-1 * rank(rank(delta(close, 10))))), 10))) + rank((-1 * delta(close, 3)))) + sign(scale(correlation(adv20, low, 12))))
Alpha#32: (scale(((sum(close, 7) / 7) - close)) + (20 * scale(correlation(vwap, delay(close, 5), 230))))
 Alpha#33: rank((-1 * ((1 - (open / close))¹)))
Alpha#34: rank(((1 - rank((stddev(returns, 2) / stddev(returns, 5)))) + (1 - rank(delta(close, 1)))))) Alpha#35: ((Ts_Rank(volume, 32) * (1 - Ts_Rank(((close + high) - low), 16))) * (1 - Ts_Rank(returns, 32)))
Alpha#36: (((((2.21 * rank(correlation((close - open), delay(volume, 1), 15))) + (0.7 * rank((open - close)))) + (0.73 * rank(Ts_Rank(delay((-1 * returns), 6), 5)))) + rank(abs(correlation(vwap, adv20, 6)))) + (0.6 * rank(((sum(close, 200) / 200) - open) * (close - open)))))
Alpha#37: (rank(correlation(delay((open - close), 1), close, 200)) + rank((open - close)))
 Alpha#38: ((-1 * rank(Ts_Rank(close, 10))) * rank((close / open)))
Alpha#39: ((-1 * rank((delta(close, 7) * (1 - rank(decay_linear((volume / adv20), 9)))))) * (1 + rank(sum(returns, 250))))
Alpha#40: ((-1 * rank(stddev(high, 10))) * correlation(high, volume, 10))

Alpha#41: (((high * low)^{0.5}) - vwap)

Alpha#42: (rank((vwap - close)) / rank((vwap + close)))
Alpha#43: (ts_rank((volume / adv20), 20) * ts_rank((-1 * delta(close, 7)), 8))

Alpha#44: (-1 * correlation(high, rank(volume), 5))

Alpha#45: (-1 * ((rank((sum(delay(close, 5), 20) / 20)) * correlation(close, volume, 2)) * rank(correlation(sum(close, 5), sum(close, 20), 2))))

Alpha#46: (((0.25 < (((delay(close, 20) - delay(close, 10)) / 10) - ((delay(close, 10) - close) / 10))) ? (-1 * 1) : (((((delay(close, 20) - delay(close, 10)) / 10) - ((delay(close, 10) - close) / 10)) < 0) ? 1 : ((-1 * 1) * (close - delay(close, 1)))))

Alpha#47: (((((rank((1 / close)) * volume) / adv20) * ((high * rank((high - close))) / (sum(high, 5) / 5))) - rank((vwap - delay(vwap, 5))))

Alpha#48: (indneutralize(((correlation(delta(close, 1), delta(delay(close, 1), 1), 250) * delta(close, 1)) / close), IndClass.subindustry) / sum(((delta(close, 1) / delay(close, 1))^2), 250)) Alpha#49: (((((delay(close, 20) - delay(close, 10)) / 10) - ((delay(close, 10) - close) / 10)) < (-1 * 0.1)) ? 1 : ((-1 * 1) * (close - delay(close, 1))))

Alpha#50: (-1 * ts_max(rank(correlation(rank(volume), rank(vwap), 5)), 5))

Alpha#51: (((((delay(close, 20) - delay(close, 10)) / 10) - ((delay(close, 10) - close) / 10)) < (-1 * 0.05)) ? 1 : ((-1 * 1) * (close - delay(close, 1))))

Alpha#52: ((((-1 * ts_min(low, 5)) + delay(ts_min(low, 5), 5)) * rank(((sum(returns, 240) - sum(returns, 20)) / 220))) * ts_rank(volume, 5))

Alpha#53: (-1 * delta(((close - low) - (high - close)) / (close - low)), 9))

Alpha#54: ((-1 * ((low - close) * (open^5))) / ((low - high) * (close^5)))

Alpha#55: (-1 * correlation(rank(((close - ts_min(low, 12)) / (ts_max(high, 12) - ts_min(low, 12)))), rank(volume), 6))

Alpha#56: (0 - (1 * (rank((sum(returns, 10) / sum(sum(returns, 2), 3))) * rank((returns * cap))))) Alpha#57: (0 - (1 * ((close - vwap) / decay_linear(rank(ts_argmax(close, 30)), 2)))) Alpha#58: (-1 * Ts_Rank(decay_linear(correlation(IndNeutralize(vwap, IndClass.sector), volume, 3.92795), 7.89291), 5.50322))

Alpha#59: (-1 * Ts_Rank(decay_linear(correlation(IndNeutralize(((vwap * 0.728317) + (vwap * (1 - 0.728317))), IndClass.industry), volume, 4.25197), 16.2289), 8.19648))

Alpha#60: (0 - (1 * ((2 * scale(rank((((close - low) - (high - close)) / (high - low)) * volume)))) - scale(rank(ts_argmax(close, 10)))))

Alpha#61: (rank((vwap - ts_min(vwap, 16.1219))) < rank(correlation(vwap, adv180, 17.9282))) Alpha#62: ((rank(correlation(vwap, sum(adv20, 22.4101), 9.91009)) < rank(((rank(open) + rank(open)) < (rank(((high + low) / 2)) + rank(high)))))) * -1)

Alpha#63: ((rank(decay_linear(delta(IndNeutralize(close, IndClass.industry), 2.25164), 8.22237)) - rank(decay_linear(correlation(((vwap * 0.318108) + (open * (1 - 0.318108))), sum(adv180, 37.2467), 13.557), 12.2883))) * -1)

Alpha#64: ((rank(correlation(sum(((open * 0.178404) + (low * (1 - 0.178404))), 12.7054), sum(adv120, 12.7054), 16.6208)) < rank(delta((((high + low) / 2) * 0.178404) + (vwap * (1 - 0.178404))), 3.69741))) * -1)

Alpha#65: ((rank(correlation(((open * 0.00817205) + (vwap * (1 - 0.00817205))), sum(adv60, 8.6911), 6.40374)) < rank((open - ts_min(open, 13.635)))) * -1)

Alpha#66: ((rank(decay_linear(delta(vwap, 3.51013), 7.23052)) + Ts_Rank(decay_linear((((low * 0.96633) + (low * (1 - 0.96633))) - vwap) / (open - ((high + low) / 2))), 11.4157), 6.72611)) * -1) Alpha#67: ((rank((high - ts_min(high, 2.14593)))^rank(correlation(IndNeutralize(vwap, IndClass.sector), IndNeutralize(adv20, IndClass.subindustry), 6.02936))) * -1)

Alpha#68: ((Ts_Rank(correlation(rank(high), rank(adv15), 8.91644), 13.9333) < rank(delta(((close * 0.518371) + (low * (1 - 0.518371))), 1.06157))) * -1)

Alpha#69: ((rank(ts_max(delta(IndNeutralize(vwap, IndClass.industry), 2.72412), 4.79344))^Ts_Rank(correlation(((close * 0.490655) + (vwap * (1 - 0.490655))), adv20, 4.92416), 9.0615)) * -1)

Alpha#70: ((rank(delta(vwap, 1.29456))^Ts_Rank(correlation(IndNeutralize(close, IndClass.industry), adv50, 17.8256), 17.9171)) * -1)

Alpha#71: max(Ts_Rank(decay_linear(correlation(Ts_Rank(close, 3.43976), Ts_Rank(adv180, 12.0647), 18.0175), 4.20501), 15.6948), Ts_Rank(decay_linear((rank(((low + open) - (vwap + vwap)))^2), 16.4662), 4.4388))

Alpha#72: (rank(decay_linear(correlation(((high + low) / 2), adv40, 8.93345), 10.1519)) / rank(decay_linear(correlation(Ts_Rank(vwap, 3.72469), Ts_Rank(volume, 18.5188), 6.86671), 2.95011)))

Alpha#73: (max(rank(decay_linear(delta(vwap, 4.72775), 2.91864)), Ts_Rank(decay_linear(((delta(((open * 0.147155) + (low * (1 - 0.147155))), 2.03608) / ((open * 0.147155) + (low * (1 - 0.147155)))) * -1), 3.33829), 16.7411)) * -1)

Alpha#74: ((rank(correlation(close, sum(adv30, 37.4843), 15.1365)) < rank(correlation(rank(((high * 0.0261661) + (vwap * (1 - 0.0261661)))), rank(volume, 11.4791))) * -1)

Alpha#75: (rank(correlation(vwap, volume, 4.24304)) < rank(correlation(rank(low), rank(adv50), 12.4413)))

Alpha#76: (max(rank(decay_linear(delta(vwap, 1.24383), 11.8259)), Ts_Rank(decay_linear(Ts_Rank(correlation(IndNeutralize(low, IndClass.sector), adv81, 8.14941), 19.569), 17.1543), 19.383)) * -1)

Alpha#77: min(rank(decay_linear((((high + low) / 2) + high) - (vwap + high)), 20.0451)), rank(decay_linear(correlation((high + low) / 2), adv40, 3.1614), 5.64125)))

Alpha#78: (rank(correlation(sum(((low * 0.352233) + (vwap * (1 - 0.352233))), 19.7428), sum(adv40, 19.7428), 6.83313))^rank(correlation(rank(vwap), rank(volume), 5.77492))) Alpha#79: (rank(delta(IndNeutralize(((close * 0.60733) + (open * (1 - 0.60733))), IndClass.sector), 1.23438)) < rank(correlation(Ts_Rank(vwap, 3.60973), Ts_Rank(adv150, 9.18637), 14.6644)))

Alpha#80: ((rank(Sign(delta(IndNeutralize(((open * 0.868128) + (high * (1 - 0.868128))), IndClass.industry), 4.04545)))^Ts_Rank(correlation(high, adv10, 5.11456), 5.53756)) * -1) Alpha#81: ((rank(Log(product(rank((rank(correlation(vwap, sum(adv10, 49.6054), 8.47743))^4)), 14.9655))) < rank(correlation(rank(vwap), rank(volume), 5.07914))) * -1) Alpha#82: (min(rank(decay_linear(delta(open, 1.46063), 14.8717)), Ts_Rank(decay_linear(correlation(IndNeutralize(volume, IndClass.sector), ((open * 0.634196) + (open * (1 - 0.634196))), 17.4842), 6.92131), 13.4283)) * -1)

Alpha#83: ((rank(delay(((high - low) / (sum(close, 5) / 5)), 2)) * rank(rank(volume))) / (((high - low) / (sum(close, 5) / 5)) / (vwap - close)))

Alpha#84: SignedPower(Ts_Rank((vwap - ts_max(vwap, 15.3217)), 20.7127), delta(close, 4.96796))

Alpha#85: (rank(correlation(((high * 0.876703) + (close * (1 - 0.876703))), adv30, 9.61331))^rank(correlation(Ts_Rank(((high + low) / 2), 3.70596), Ts_Rank(volume, 10.1595), 7.11408)))

Alpha#86: ((Ts_Rank(correlation(close, sum(adv20, 14.7444), 6.00049), 20.4195) < rank(((open + close) - (vwap + open)))) * -1)

Alpha#87: $(\max(\text{rank}(\text{decay_linear}(\text{delta}(((\text{close} * 0.369701) + (\text{vwap} * (1 - 0.369701))), 1.91233), 2.65461)), \text{Ts_Rank}(\text{decay_linear}(\text{abs}(\text{correlation}(\text{IndNeutralize}(\text{adv81}, \text{IndClass.industry}), \text{close}, 13.4132)), 4.89768), 14.4535)) * -1)$

Alpha#88: $\min(\text{rank}(\text{decay_linear}(((\text{rank}(\text{open}) + \text{rank}(\text{low})) - (\text{rank}(\text{high}) + \text{rank}(\text{close}))), 8.06882)), \text{Ts_Rank}(\text{decay_linear}(\text{correlation}(\text{Ts_Rank}(\text{close}, 8.44728), \text{Ts_Rank}(\text{adv60}, 20.6966), 8.01266), 6.65053), 2.61957))$

Alpha#89: $(\text{Ts_Rank}(\text{decay_linear}(\text{correlation}(((\text{low} * 0.967285) + (\text{low} * (1 - 0.967285))), \text{adv10}, 6.94279), 5.51607), 3.79744) - \text{Ts_Rank}(\text{decay_linear}(\text{delta}(\text{IndNeutralize}(\text{vwap}, \text{IndClass.industry}), 3.48158), 10.1466), 15.3012))$

Alpha#90: $((\text{rank}((\text{close} - \text{ts_max}(\text{close}, 4.66719)))^{\wedge} \text{Ts_Rank}(\text{correlation}(\text{IndNeutralize}(\text{adv40}, \text{IndClass.subindustry}), \text{low}, 5.38375), 3.21856)) * -1)$

Alpha#91: $((\text{Ts_Rank}(\text{decay_linear}(\text{decay_linear}(\text{correlation}(\text{IndNeutralize}(\text{close}, \text{IndClass.industry}), \text{volume}, 9.74928), 16.398), 3.83219), 4.8667) - \text{rank}(\text{decay_linear}(\text{correlation}(\text{vwap}, \text{adv30}, 4.01303), 2.6809))) * -1)$

Alpha#92: $\min(\text{Ts_Rank}(\text{decay_linear}(((\text{high} + \text{low}) / 2) + \text{close}) < (\text{low} + \text{open})), 14.7221), 18.8683), \text{Ts_Rank}(\text{decay_linear}(\text{correlation}(\text{rank}(\text{low}), \text{rank}(\text{adv30}, 7.58555), 6.94024), 6.80584))$

Alpha#93: $(\text{Ts_Rank}(\text{decay_linear}(\text{correlation}(\text{IndNeutralize}(\text{vwap}, \text{IndClass.industry}), \text{adv81}, 17.4193), 19.848), 7.54455) / \text{rank}(\text{decay_linear}(\text{delta}(((\text{close} * 0.524434) + (\text{vwap} * (1 - 0.524434))), 2.77377), 16.2664)))$

Alpha#94: $((\text{rank}((\text{vwap} - \text{ts_min}(\text{vwap}, 11.5783)))^{\wedge} \text{Ts_Rank}(\text{correlation}(\text{Ts_Rank}(\text{vwap}, 19.6462), \text{Ts_Rank}(\text{adv60}, 4.02992), 18.0926), 2.70756)) * -1)$

Alpha#95: $(\text{rank}((\text{open} - \text{ts_min}(\text{open}, 12.4105))) < \text{Ts_Rank}((\text{rank}(\text{correlation}(\text{sum}((\text{high} + \text{low}) / 2), 19.1351), \text{sum}(\text{adv40}, 19.1351), 12.8742))^{\wedge} 5), 11.7584))$

Alpha#96: $(\max(\text{Ts_Rank}(\text{decay_linear}(\text{correlation}(\text{rank}(\text{vwap}), \text{rank}(\text{volume}), 3.83878), 4.16783), 8.38151), \text{Ts_Rank}(\text{decay_linear}(\text{Ts_ArgMax}(\text{correlation}(\text{Ts_Rank}(\text{close}, 7.45404), \text{Ts_Rank}(\text{adv60}, 4.13242), 3.65459), 12.6556), 14.0365), 13.4143)) * -1)$

Alpha#97: $((\text{rank}(\text{decay_linear}(\text{delta}(\text{IndNeutralize}(((\text{low} * 0.721001) + (\text{vwap} * (1 - 0.721001))), \text{IndClass.industry}), 3.3705), 20.4523)) - \text{Ts_Rank}(\text{decay_linear}(\text{Ts_Rank}(\text{correlation}(\text{Ts_Rank}(\text{low}, 7.87871), \text{Ts_Rank}(\text{adv60}, 17.255), 4.97547), 18.5925), 15.7152), 6.71659)) * -1)$

Alpha#98: (rank(decay_linear(correlation(vwap, sum(adv5, 26.4719), 4.58418), 7.18088)) - rank(decay_linear(Ts_Rank(Ts_ArgMin(correlation(rank(open), rank(adv15), 20.8187), 8.62571), 6.95668), 8.07206)))

Alpha#99: ((rank(correlation(sum((high + low) / 2), 19.8975), sum(adv60, 19.8975), 8.8136)) < Rank (correlation(low, volume, 6.28259))) * -1)

Alpha#100: (0 - (1 * ((1.5 * scale) (indneutralize(indneutralize(rank((((close - low) - (high - .) close) / (high - low)) * volume)), IndClass.subindustry), IndClass.subindustry Scale (indneutralize((correlation(close, rank(adv20), 5) - rank(ts_argmin(close, 30))), IndClass.subindustry))) * (volume / adv20))))

Alpha#101: ((close - open) / ((high - low) + .001))

A.1. Functions and Operators

(下面的 “{}” 表示占位符。所有表达式不区分大小写。)

Abs (x), log(x), sign(x) =标准定义;运算符 “+”、 “-”、 “*”、 “/”、 “>”、 “<”、 “==”、 “||”、 “x ?”
Y: z”

Rank (x) =横断面Rank

延迟(x, d) = x d天前的值

相关性(x, y, d) =过去d天x和y的时间序列相关性

协方差(x, y, d) =过去d天x和y的时间序列协方差

Scale (x, a) =重新缩放的x, 使得sum(abs(x)) = a(默认为a = 1)

Delta (x, d) =今天x的值减去d天前x的值

Signedpower (x, a) = x^a

Decay_linear (x, d) =过去d天的加权移动平均值, 其权重为线性衰减的d, d -1, ..., 1(重新缩放为总和为1)

Indneutralize (x, g) =x对组g(子行业、行业、部门等)进行横截面中和, 即x在每个组g内被横截面贬低

ts_{O}(x, d) =在过去d天的时间序列中应用的算子O;将非整数天数d转换为楼层(d)

Ts_min (x, d) =过去d天的时间序列min

Ts_max (x, d) =过去d天的时间序列最大值

Ts_argmax (x, d) = ts_max(x, d)发生在ts_argmin(x, d)的哪一天= ts_min(x, d)发生在哪一天

Ts_rank (x, d) =过去d天的时间序列排名

Min (x, d) = ts_min(x, d)

Max (x, d) = ts_max(x, d)

Sum (x, d) =过去d天的时间序列和

Product (x, d) =过去d天的时间序列乘积

Stddev (x, d) =过去d天的移动时间序列标准差

A.2. 输入数据

回报=每日近距离回报

开盘价、收盘价、高点、低点、成交量=每日价格和成交量数据的标准定义vwap =每日成交量加权均价

Cap =市值

Adv {d} =过去d天的日均美元交易量

在indneutralize(x, IndClass.level)中， IndClass =二进制行业分类(如GICS, BICS, NAICS, SIC等)的通用占位符， 其中level = sector, industry, subindustry等。同一alpha中的多个IndClass不必对应于相同的行业分类。

附录B:免责声明

在语境需要的地方， 阳性包括阴性和/或中性， 单数形式包括复数， 反之亦然。本文作者 (“作者”)及其关联公司， 包括但不限于Quantigic®Solutions LLC(“作者关联公司” 或 “其关联公司”)不作任何暗示或明示保证或任何其他声明， 包括但不限于对适销性和适合特定目的的暗示保证。与本文内容相关或与本文内容相关， 包括但不限于本文中包含的任何公式、代码或算法(“内容”)。

读者可自行承担使用内容的风险，读者不得向作者或其关联公司提出任何索偿，而作者及其关联公司对读者或任何第三方的任何损失、费用、机会成本、损害赔偿或任何其他与读者使用内容有关或因使用内容而产生的不利影响，包括但不限于：任何直接、间接、附带、特殊、后果性或任何其他由读者招致的损害赔偿，无论其原因如何，或根据任何责任理论；任何利润损失（无论直接或间接产生）、任何商誉或声誉损失、任何数据损失、采购替代商品或服务的成本，或任何其他有形或无形的损失；读者对内容的完整性、准确性或存在的任何依赖，或使用内容的任何其他效果；以及读者在使用内容时可能遇到的任何和所有其他逆境或负面影响，无论作者或其关联公司是否已经或应该已经意识到此类逆境或负面影响。

本协议附录A中包含的公式和代码是在WorldQuant LLC明确许可的情况下提供的。WorldQuant LLC保留对本协议附录A中包含的公式和代码的所有权利、所有权和利益及其任何和所有版权。

参考文献

- Avellaneda, M. and Lee, J.H. “Statistical arbitrage in the U.S. equity market.” Quantitative Finance 10(7) (2010), pp. 761-782.
- Grinold, R.C. and Kahn, R.N. “Active Portfolio Management.” New York, NY: McGraw-Hill, 2000. Jegadeesh, N. and Titman, S. “Returns to buying winners and selling losers: Implications for stock market efficiency.” Journal of Finance 48(1) (1993), pp. 65-91.
- Kakushadze, Z. “Factor Models for Alpha Streams.” The Journal of Investment Strategies 4(1) (2014), pp. 83-109.
- Kakushadze, Z. and Tulchinsky, I. “Performance v. Turnover: A Story by 4,000 Alphas.” Journal of Investment Strategies (forthcoming). Available online: <http://ssrn.com/abstract=2657603> (September 7, 2015).
- Pastor, L. and Stambaugh, R.F. “Liquidity Risk and Expected Stock Returns.” The Journal of Political Economy 111(3) (2003), pp. 642-685.
- Tulchinsky, I. *et al.* “Finding Alphas: A Quantitative Approach to Building Trading Strategies.” New York, NY: Wiley, 2015.

表格

Quantity	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
S	1.238	1.929	2.224	2.265	2.498	4.162
T	0.1571	0.3429	0.4752	0.5456	0.6474	1.604
$1 / T$	0.6235	1.545	2.104	2.391	2.916	6.365
C	0.1324	0.3125	0.3969	0.4814	0.5073	2.031
$10^3 \times \sigma$	0.9318	1.194	1.395	1.747	2.019	10.44
$100\% \times \tilde{R}$	3.285	4.4	5.441	6.015	6.296	28.72
$100\% \times \Psi_{ij}$	-15.09	7.457	14.31	15.86	22.91	87.33

表1。总结(使用R函数总结())年化夏普比率3, 日年化平均日收益>, 以及与>的配对相关性(见第3节)。成交量, , 平均持仓期1 /, 每股美分4, 日收益波动率=, 业绩数据不包括任何交易或交易成本。价格影响等因素。

	Estimate	Standard error	t-statistic	Overall
Intercept	-3.509	0.295	-11.88	
$\ln(\sigma)$	0.761	0.046	16.65	
Mult./Adj. R-squared				0.737 / 0.734
F-statistic				277.2

表2。摘要(使用R函数Summary (lm()))对 $\ln()$ / $\ln(=)$ 与截距的横截面回归。有关详细信息, 请参见小节3.1。另见图2。

	Estimate	Standard error	t-statistic	Overall
Intercept	-3.435	0.324	-10.60	
$\ln(\sigma)$	0.775	0.052	14.84	
$\ln(T)$	-0.023	0.040	-0.57	
Mult./Adj. R-squared				0.738 / 0.732
F-statistic				137.8

表3。 $\ln()$ / $\ln(=)$ 和 $\ln()$ 与截距的截面回归总结。有关详细信息, 请参见小节3.1。

	Estimate	Standard error	t-statistic	Overall
Intercept	0.1587	0.0017	95.18	
y_a	0.0067	0.0023	2.907	
z_a	0.0474	0.0063	7.537	
Mult./Adj. R-squared				0.0127 / 0.0123
F-statistic				32.55

表4。总结了Z对PZ和SZ截距的截面回归。有关详细信息, 请参见小节。另见图3。

	Estimate	Standard error	t-statistic	Overall
Intercept	-6.174	0.062	-100.1	
$\ln(T)$	0.368	0.068	5.412	
Mult./Adj. R-squared				0.228 / 0.221
F-statistic				29.29

表5。 $\ln(\text{=}) / \ln(\text{0})$ 与截距的截面回归总结。有关详细信息，请参见小节。另见图4。

图

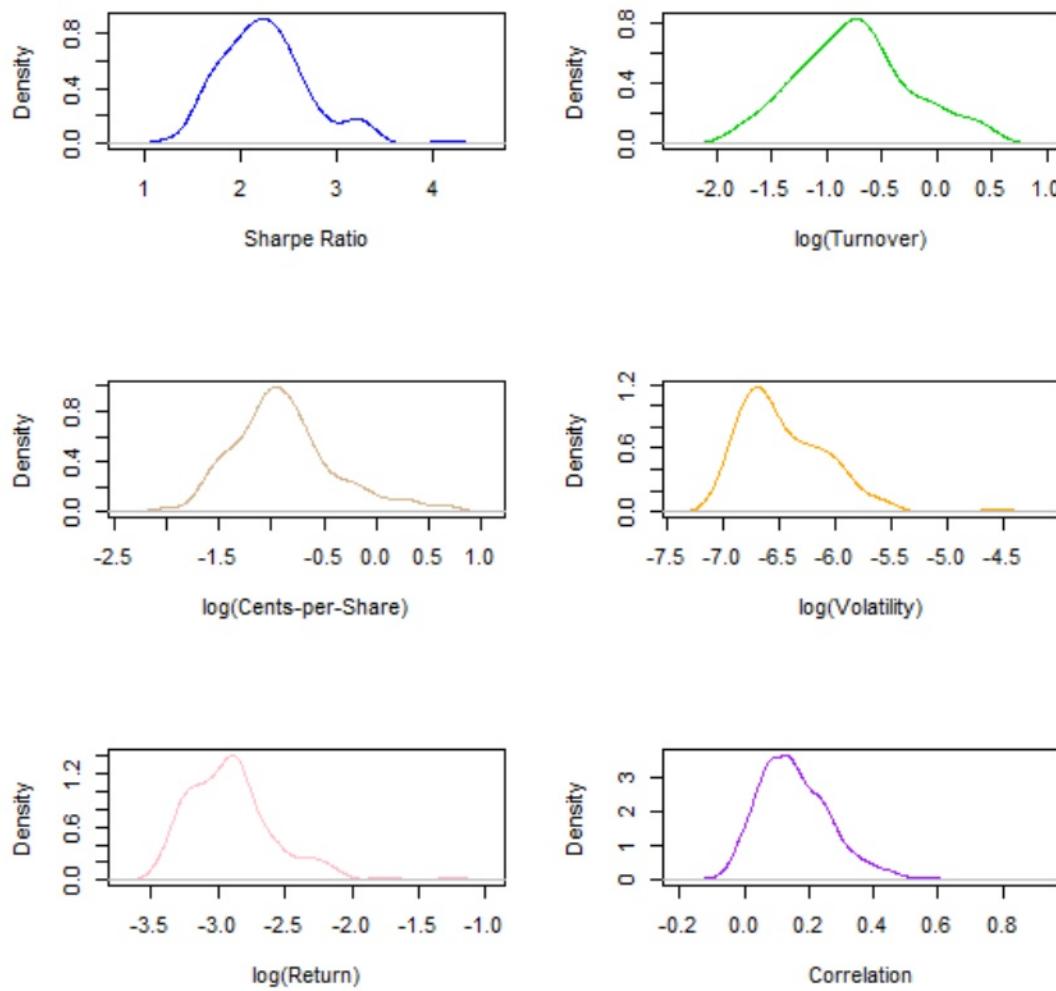


图1。密度(使用R函数密度())绘制了年化夏普比率3、日成交量、每股收益4、日收益波动率=、年化平均日收益>以及与>的配对相关性(见表1和第3节)。3、=和>中的“极端”异常值是由于延迟-0 alpha(见第2节)。

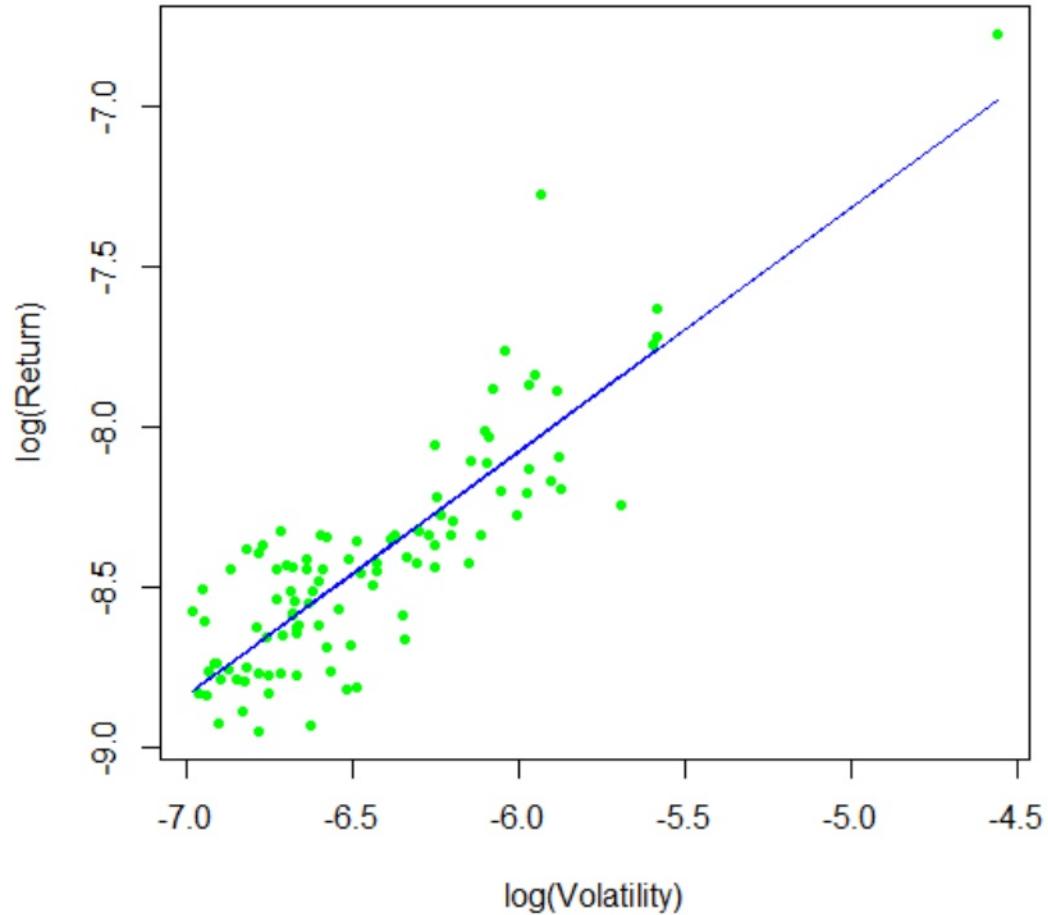


图2。横轴: $\ln(=)$;纵轴: $\ln()$ 。点表示数据点。直线绘制线性回归拟合 $\ln() \approx -3.509 + 0.761 \ln(=)$ 。见表2。

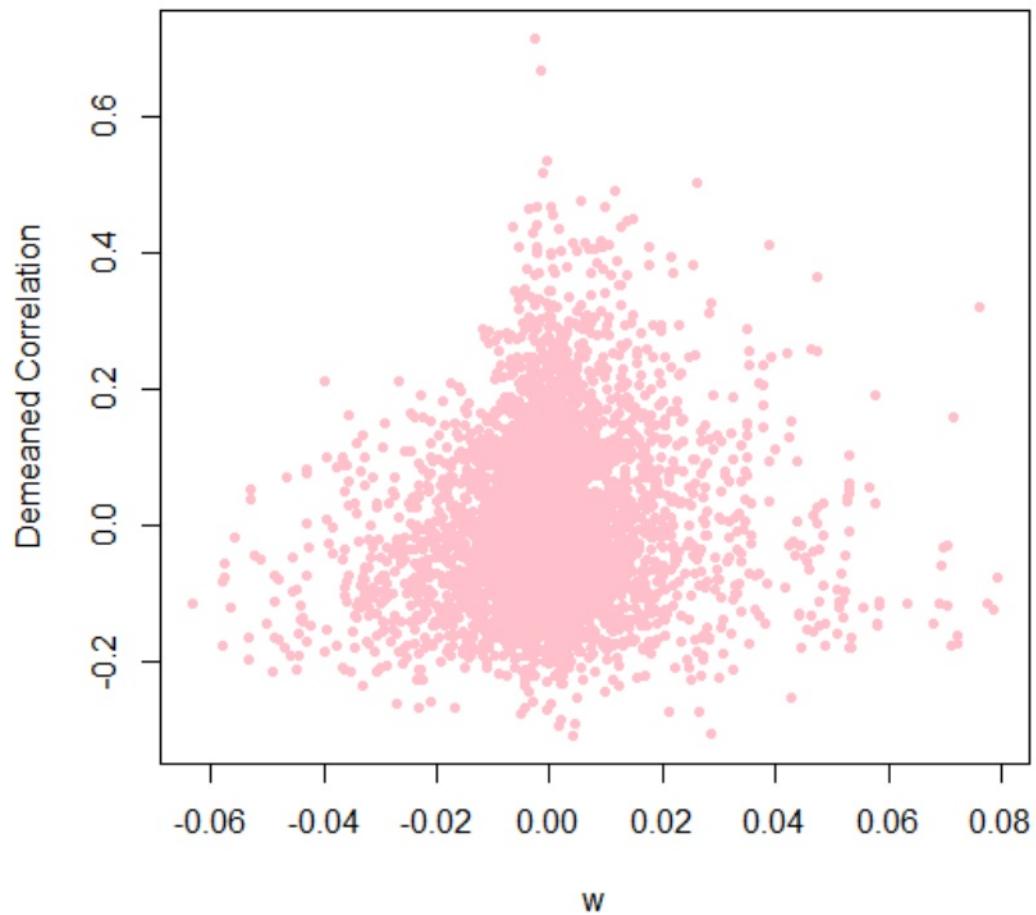


图3。横轴: $Z = 0.0067 PZ + 0.0474 SZ$;纵轴: $Z - \text{Mean}(Z)$ 。见表4和3.2小节。数值系数为表4中的回归系数。

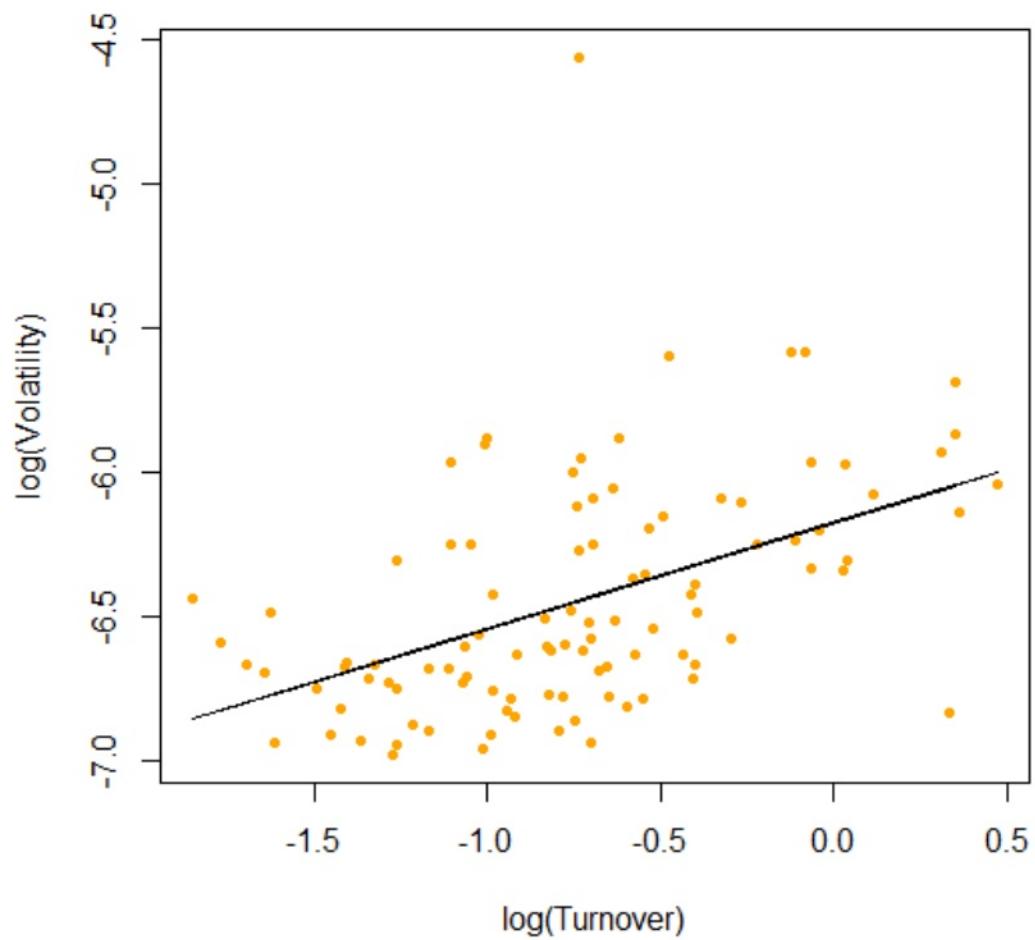


图4。横轴: $\ln()$;纵轴: $\ln(=)$ 。点表示数据点。直线绘制线性回归拟合 $\ln(=) \approx -6.174 + 0.368 \ln()$ 。见表5。