2017 Fall Competition
Case Packet

traders@mit.edu
traders.mit.edu

August 7, 2017

Contents

0.1 Overview

1 Introduction

Traders@MIT welcomes you to our 10th Annual Intercollegiate Trading Competition! We are excited to be holding the competition again this year and are confident that it will provide a rich and challenging learning experience. The competition will be held on 11/11, and competitors will also be required to attend networking events held on 11/10 (Veterans Day). All travel to and from the competition will be fully reimbursed.

Our competition consists of electronic trading. Teams will be ranked overall based on their total weighted rankings from each of the three cases. You will also have several opportunities to speak with our attending sponsors.

More logistical details will be forthcoming closer to competition day.

Our sponsors for this year's competition are:

Platinum: DRW Trading, Wolverine Trading, Town Square Trading, Optiver, Consolidated Trading, SevenEight Capital, Barclays, SIG, Five Rings Capital

Gold: D.E. Shaw, Hudson River Trading, Bank of America Merrill Lynch, Jane Street Capital, Group One Trading, Goldman Sachs, IMC Financial Markets, Old Mission Capital, BNP Paribas, Flow Traders, Belvedere Trading,

Silver: Tech Square Trading, Virtu, Vatic Labs, TD Securities, Element Capital, 3Red Trading, Volant Trading

2 Dark Pools and Foreign Exchange

2.1 Overview

Dark pools are private exchanges on which confidential (and often massive) trades can be conducted by large financial institutions. The order book in a dark pool is hidden; further, fulfilled orders are often revealed to the public with a delay, or even never revealed at all. As a result, high-volume trades can be executed in a dark pool with little to no market impact on public exchanges.

The foreign exchange (FX) market is the global platform for trading currencies and currency products. FX is the largest and most liquid market in the world: about $5.1 trillion is traded per day. The real-world FX market is heavily dependent on macroeconomic news and events.

In this case, you will design an algorithm to trade on the foreign exchange market on both public and dark exchanges. You will be expected to discover and capitalize upon arbitrage opportunities. **Unlike previous years, you will NOT be allowed to click trade for this case.** Information about the TradersBot API will be provided, which will allow you to pull in pricing information and also submit trades programmatically.

2.1.1 Foreign Exchange

FX quotes are given in terms of currency pairs. The first currency listed is the base currency, and the second currency listed is known as the quote currency. When you are going "long" or buying a currency pair, you are buying the base currency and selling the quote currency. The quote currency is quoted in terms of its value relative to the other currency in the pair. For example, if the USD/JPY exchange rate is given as

$$\text{USD/JPY} = 106.78,$$

this means that US$1 = 106.78 Japanese yen. In the next example, USD is the quote currency and EUR is the base currency. If the exchange rate is the following:

$$\text{EUR/USD} = 1.2508,$$

then 1 Euro = US$1.2508.

In the FX market, these currency pairs can be bought and sold at the given bid/ask price. The bid is the price the counterparty is willing to buy the pair from you, and the ask is the

price they are willing to sell the pair to you. Remember, "buying the pair" means buying the base currency and selling the quote currency, while "selling the pair" means selling the base currency and buying the quote currency.

2.2 Case Information

In this case, you will be trading pairs of the following 5 underlying currencies: USD (US dollar), EUR (Euro), JPY (Japanese yen), CHF (Swiss franc), and CAD (Canadian dollar).

You will be able to trade 8 currency pairs, listed below:

Exchange	Currency Pairs			
Public	USD/CAD	USD/EUR	USD/CHF	USD/JPY
Dark	EUR/CAD	EUR/JPY	EUR/CHF	CHF/JPY

Note that this is not all $\binom{5}{2} = 10$ possible currency pairs.

Case details:

Starting Endowment Per Round	US$100,000
Rounds of Trading	3
Length of Round	15 min/round
Fees	$0.0001/contract
Fines	Trading with yourself: $0.008/contract
Constraints	Position limits will be enforced; see below. Orders that would cause you to go outside your position limit will be rejected. The maximum transaction size is USD$10,000. You may only send 25 pieces of information a second, otherwise your orders will get rejected. More information is available in the Mangocore API section.

Position limits for the underlying currencies:

Currency	Position Limits
USD	[0, ∞]
EUR	$\pm$100,000
JPY	$\pm$10,000,000
CHF	$\pm$100,000
CAD	$\pm$100,000

Note: Position limits for underlying currencies do not imply position limits on each currency pair.

2.3 Trading Tips and Strategies

2.3.1 Triangular Arbitrage

We will provide servers to backtest your code on. If your code is consistently profitable when traded on historical data, there is a better chance it will be profitable during the competition.

We now explain the concept of triangular arbitrage. The motivation behind triangular arbitrage is to take advantage of price discrepancies between three currencies. For example, theoretically, the price of the pair EUR/JPY should equal the price of the pair EUR/USD times the price of the pair USD/JPY. If not, then an arbitrage opportunity exists. As an example, suppose that

$$\begin{gathered}\text{EUR/JPY} = 134 \\ \text{EUR/USD} = 1.12 \\ \text{USD/JPY} = 119\end{gathered}$$

Note that $1.12 \cdot 119 = 133.28 < 134$. We can make money by buying the two currency pairs EUR/USD and USD/JPY, and selling EUR/JPY. Explicitly, we can convert 13328 yen to US$112 by buying 112 contracts of USD/JPY. Next, we can can convert US$112 to 100 Euro by buying 100 contracts of EUR/USD. Finally, we can convert 100 Euro to 13400 yen by selling 100 contracts of EUR/JPY. Overall, we made a profit of $13400 - 13328 = 72$ yen. Our profit from this trade is US$72/119 $\approx$ US$0.61.

2.3.2 News Items

In this case, you will be provided news items, which you are encouraged to incorporate into your algorithm.

- Name of a large market entrant (one of $\{A, B, C, D\}$).
- A dark currency pair (e.g., EUR/CHF)
- A timeframe $[a,\ a + dt]$ where a, dt are at least 5 and at most 30 ticks from the current time.

Each entrant is associated with a parameter $\sigma \in [0.52, 0.68, 0.82, 0.95]$. σ is the likelihood that after a participant sells a given currency pair, the value of the pair goes down. It is up to you to determine the correct mapping. Note that you may want to change your strategy for providing fills based on this parameter; it may be prudent to "fade" – i.e., provide less strict fills – for the entrants that are more strongly correlated with market trends. For example, if $\sigma_A = 0.95$, filling an order from entrant A may not be strictly beneficial, as the likelihood of your fill decreasing in value is close to 1.

The purpose of this news item is to inform you that the entrant will submit a sell order for the pair on the dark exchange somewhere within the timeframe. This order will be very large (in particular, it will be larger than the transaction limit by some constant factor), and given

that liquidity in dark pools is very low, we expect that teams with winning strategies will take advantage of these trades.

A sample news bulletin would look as follows: "TRADE ALERT: Entrant A will submit a sell order for EUR/CHF between 10 and 18 ticks from current time."

2.4 Position Close-Out, Scoring, and Illegal Activity

Any non-zero position in a currency will be closed out at the end of the trading period with the closing price, with a small penalty (0.01USD/share). Your PNL during the case will be displayed in US dollars—our grading script will make the necessary currency conversions in real time. If your PNL drops below a certain threshold (bottom 10% of all competitors) your trades will be blocked for the remainder of the round.

Within each of the 3 rounds, you will be ranked by your PNL. Your PNL will reset to 0 at the beginning of each of the rounds. To determine final rankings, we will consider your average rank across all three rounds.

Frontrunning, or operating on advanced knowledge of pending orders from other brokers, is illegal. **You cannot trade the same direction as the market entrant in the news bulletin**; you are restricted to providing liquidity. Competitors that do this will be disqualified.

3 Options

3.1 Overview

In this case, you will algorithmically trade European options on an index based on three different equities. Your ability to earn profits will depend on your ability to:

1. exploit inconsistencies in implied volatility across different strike prices
2. market make when spreads are large, without becoming overly exposed to risk
3. hedge positions to reduce risk

This case is relatively complex. We strongly recommend investing time in understanding the relevant concepts and building a useful model before the competition.

For the rest of this case description, we will simply refer to European options as options.

3.2 Options Basics

Options come in two flavors: A *call* option confers the right to buy a given product (the underlying) at a fixed price (the strike price) on – but not before – a given date (the expiration date). Likewise, a *put* option gives the right to sell the underlying at the strike price on the expiration date.

For example, imagine that an investor is holding an Apple call option at a strike of $100 and expiry of November 15th. (S)he would exercise the right to buy if the price of Apple was greater than $100 on November 15th.

The price or value of an option depends on the volatility of the underlying. *Volatility* is the standard deviation of the logarithmic return distribution of the underlying. Intuitively, it is a way to quantify how much the price will fluctuate. There are two relevant measures of volatility:

- Realized volatility is the volatility of the underlying over some past time period and is generally estimated as the sample standard deviation of the annualized log returns during the period.
- Implied volatility is the volatility of the underlying that when used in a pricing model returns the current market price of the option. We will explore this concept more later.

Options can be either in-the-money, out-of-the-money, or at-the-money. In-the-money refers to an option that is profitable if it were to be exercised. Out-of-the-money is defined analogously. At-the-money refers to an option that is whose strike price is at the spot price; i.e. the option would break even if it was exercised today.

3.3 Options Pricing Models

Given the current price of the underlying (the *spot* price) S, strike price K, risk-free interest rate r, volatility σ, current time t, and expiration time T, the Black-Scholes formula for the value C of a call option on a non-dividend-paying asset is

$$C = N(d_1)S - N(d_2)Ke^{-r(T-t)}$$

where

$$d_1 = \frac{ln(\frac{S}{K}) + (r + \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}}$$

$$d_2 = d1 - \sigma\sqrt{T-t}$$

and $N(x)$ is the standard normal cumulative distribution function. A derivation of this model can be found here: http://www.math.cuhk.edu.hk/~rchan/teaching/math4210/chap08.pdf. Briefly, the Black-Scholes formula and related extensions assume the spot price follows a geometric Brownian motion, which is equivalent to claiming that incremental logarithmic returns are normally-distributed and have a constant standard deviation. In reality, returns are not typically normally distributed. They tend to have a higher probability of extreme events than would be the case under a normal distribution ("fat tails"), particularly for extreme negative events. The Black-Scholes formula does give a fast and relatively accurate approximation, and is widely used as a basic model. The value P for a corresponding put option can be obtained from put-call parity, which for European options is as follows:

$$C + Ke^{-r(T-t)} = P + S$$

The intuition behind this formula is as follows: Ignoring for the moment the interest rate (which will be $r = 0$ during the competition), the asset package on the left side of this equation (i.e. one call option plus the dollar value of the strike) will pay out a guaranteed K, plus an additional bonus value if the option is in-the-money equal to the difference between the price of the underlying and the strike price. Therefore the left side will pay out the larger of K and the price of the underlying at expiry. Meanwhile, the right side (comprised of a put P and a future S) will pay out the value of the underlying, plus additionally the difference between the strike price and the underlying if the put option is in-the-money. Take a moment to understand why these are equal; the interest rate factor simply accounts for the time value of money.

3.4 Volatility Curves, Smile and Skew

Under the assumption that the options pricing follows Black-Scholes, we can invert the above formula for a given strike price and expiry time to deduce the implied volatility. Perhaps surprisingly, options with different strike prices often have different implied volatilities. Typically

options that are more out-of-the-money or in-the-money will have higher implied volatilities, because the actual distribution of the returns of the underlying is more fat-tailed than the normal distribution. This effect is known as the *volatility smile*. Moreover, options with strike prices much lower than the current spot price often exhibit higher implied volatility than options with strike prices equally higher than the current spot price. This effect is known as the *volatility skew*. The best way to visualize this is by plotting the implied volatility against the strike price; this is known as the volatility curve. Figure 1 shows the effect of the October 1987 stock market crash on the volatility curve.

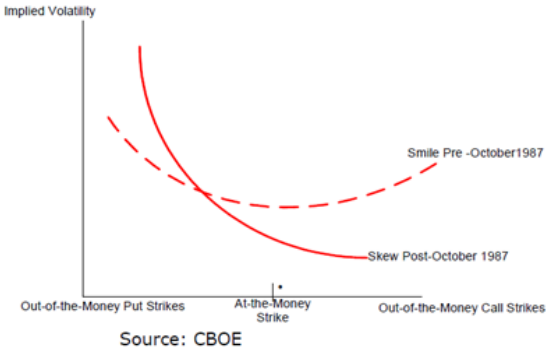
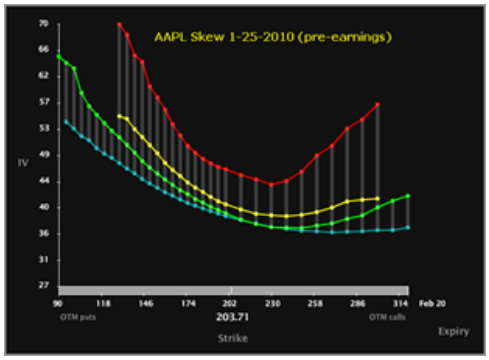
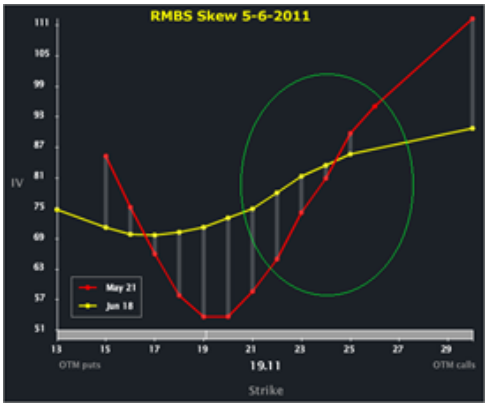


Figure 1: The S&P 500 Implied Volatility Curve Pre- and Post-1987

As seen above, an increase in the perceived likelihood of extreme negative events can increase volatility skew. Similarly, a belief of increased likelihood in extreme events in either direction compared to more typical events unchanged can make the volatility smile more pronounced. Note however, that these effects are *not* equivalent to that of belief that price movements will on average be larger than they have been in the recent past; this instead causes a parallel shift upward in the implied volatilities of at all strike prices. These effects can occur simultaneously.

Below, we show two examples of changing volatility curves. The left is from just prior to an earnings release for a stock. Note the pronounced curvature and asymmetry in the front-month options (red). Also note that the overall level of volatility for the front-month options is higher than for longer-dated options, indicating that most of the variability in price is expected to occur in the near-term:





In contrast, the chart on the right comes from a company that engages in numerous patent lawsuits. A particular one is expected to reach a decision soon, and the decision will be either very good or very bad for the future of the company. The front-month options again exhibit a substantial volatility smile, but it is more symmetrical this time. Also, subsequent volatility

remains high in this example given the risks of other ongoing litigation and R&D. However, these later risks are expected to be less binary than the impending decision: market participants predict the later return distribution will have thinner tails, so the implied volatility surface is flatter.

3.5 Greeks and Hedging

Option prices are a function of multiple variables, including the price of the underlying, the volatility of the underlying and time. We can differentiate with respect to these variables to get various quantities, commonly called the “Greeks." The Greeks describe the effects of changes in various parameters on the price of the option. We can think of the Greeks as representing exposure to market risk along different axes. Hedging is the process of minimizing such risk across each Greek.

The simplest and most important Greeks are:

- Delta: the rate of change of an option's value relative to a change in the underlying. It is always positive for calls and always negative for puts. Traders can think of delta as the impact of the movement of the underlying on the value of their portfolio; therefore, it is also equivalent to how many shares of the underlying to sell in order to hedge their option against the movement of the underlying. Having a portfolio with a small (in absolute value) delta corresponds to not having an opinion as to whether the underlying will move up or down. The value of delta for a call option is always between 0 and 1, and the values of delta for put and call options at the same strike price can be related via put-call parity (since the derivative of the underlying with respect to itself is clearly 1). Intuitively, for call options we should expect that delta should approach 1 for very in-the-money options and 0 for very out-of-the-money options; this is indeed the case.

- Gamma: the rate of change of delta relative to a change in the underlying. Since delta increases from 0 to 1 as the underlying passes the strike price, gamma is always positive for call options; since the delta for put options is simply 1 less than that for call options via put-call parity, gamma is the same for put and call options. Gamma determines how quickly you must adjust a delta hedge when price changes.

- Vega: the rate of change of an option's value relative to the (implied) volatility of the underlying. Vega is positive for every option. Intuitively, this is because extremely out-of-the-money options result in a fixed loss, whereas extremely in-the-money options can have large payouts proportional to exactly how in-the-money they are; hence, as volatility and hence the likelihood of both extremely good and extremely bad events increases, the options price should only reflect the added value of the extremely good outcomes and will therefore increase. Vega tends to be higher for options with later expiry dates and for options with a strike price near the current spot. A trader holding a portfolio with a small (in absolute value) vega has no opinion about whether the volatility of the underlying will go up or down across all strike prices; however, (s)he may still have an opinion about the shape of the volatility curve (which will be discussed in the next section). The vega for the underlying will be zero.

Expressions for the Greeks from differentiating Black-Scholes are below, assuming an interest rate of 0. Note these can become unstable for options very close to expiry: where $N(x)$ is the

Table 3.1: Formulas for Greeks

Greek	Calls	Puts
Delta	$N(d_1)$	$N(d_1) - 1$
Gamma	$\frac{N'(d_1)}{S\sigma\sqrt{T-t}}$	$\frac{N'(d_1)}{S\sigma\sqrt{T-t}}\ N(d_1)$
Vega	$SN'(d_1)\sqrt{T-t}$	$SN'(d_1)\sqrt{T-t}$

standard normal cumulative distribution function and $N'(x)$ is the standard normal probability density function. Successful competitors will be able to compute the Greeks for each product using a model receiving live updates from MangoCore, making use of the implied volatilities computed by inverting Black-Scholes.

3.6 Volatility Arbitrage

Volatility arbitrage aims to take advantage of inconsistent future and current volatilities. For a certain underlying P, profits can come from differences between the anticipated future realized volatility of P and current implied volatilities of options for P. Profits can also come from differences between current and future implied volatilities, and from the differences in implied volatilities across different strike prices.

For example, suppose a trader believes that the future realized volatility for a given product will be higher than the current implied volatility of a call option on that product across all strike prices. Then the trader believes the option is cheaper than it should be, so (s)he buys it, hedges delta by shorting some amount of the underlying, and adjusts the hedge as the price of the underlying changes. Note that this trade has a positive vega, because the trader believes that the implied volatility will increase.

One of the most common types of volatility arbitrage is called volatility surface relative value trading. It involves attempting to earn profits by predicting changes in the shape of the volatility smile, independent of beliefs about vega.

Suppose you believe the volatility smile is relatively flat, compared to what you believe to be the true shape of the curve. Then you could buy options far out on the curve, perhaps hedge some vega exposure by shorting options with strike prices closer to the current spot price (at-the-money options) if you have no view on the future overall level of volatility, and then hedge your delta exposure by holding a position in the underlying equal and opposite to the remaining delta of your options portfolio. This leaves you exposed only to changes in the curvature of the smile. A completely analogous strategy could be employed if you believe that the skew of the volatility smile will increase. Successful competitors in this case will be able to identify inconsistencies in the volatility smile to exploit such arbitrage opportunities.

3.7 Market Making

Another possible way to make money in options is market making, because options spreads are often relatively large. However, this runs the risk of adverse selection. For example, if you are offering 100 call options, you should be willing to do so *given that someone is willing to trade against you*. Someone who has a better grasp of the market, or better information, could take advantage of an overzealous market maker offering tiny spreads.

Successful competitors in this case will be able to market make when spreads are high, without exposing themselves to too much additional risk, such as by hedging.

3.8 Case Information

The tradable instruments are options and futures on the T@MIT index, where the futures are solely for hedging. Both are cash-settled. Note the options underlying is the index, not the futures contract.

There are 82 options in each round of the case: 41 puts and 41 calls, in range(80, 121). Options expire at the end of each round and their tickers are subsequently re-used. Five example tickers are listed below; other tickers follow the same naming conventions:

Table 3.2: Tradable Options

Strike Price	**Put Option Ticker**	**Call Option Ticker**
$90	T90P	T90C
$95	T95P	T95C
$100	T100P	T100C
$105	T105P	T105C
$110	T110P	T110C

The ticker for futures on the index is “TMXFUT”. The ticker for the T@MIT index itself is "TMX". The index is viewable but not tradable.

To successfully trade this case, you should make a pricing model to show the current volatility curve, your current portfolio Greeks, and any refinements or derived metrics that help you trade faster and better. This will allow you to quickly pursue arbitrage opportunities and to hedge your positions. In addition to pursuing such arbitrage opportunities when spreads are low, you should also recognize potential market making opportunities at times when spreads are higher, where the potential gain may outweigh the adverse selection risk. Competitors should read MangoCore API to understand how to get live MangoCore updates programatically and then build models off of them.

3.9 Case Specifications

There will be four 7.5-minute periods. Each period represents one month of trading. One team member will trade while the other monitors a pricing model and advises the trader. Team

members will switch roles after each period. Trading activity by both partners during any given round is grounds for disqualification. Unlike the SEC, we have granular audit logs – please don't make us use them.

All tradable instruments expire at the end of each round. Each round begins with the T@MIT index at 100. The continuously compounded risk-free interest rate is **0%**. No dividends will be paid out during case periods.

The contract multiplier for **options** on the T@MIT index is 1. Options have a trading fee of \$0.005 per contract per transaction. There is a net limit of 5,000 options contracts.

The contract multiplier for **futures** on the T@MIT index is 1. Futures have a trading fee of \$0.005 per contract per transaction. There is a gross trading limit of 2,500 futures contracts, and a net limit of 2,500 futures contracts. All future contracts are cash-settled upon expiry.

There will also be delta and vega **portfolio limits** of $\pm 10,000$ and $\pm 10,000$ respectively. One share of the underlying will have a delta of 100. Upon exceeding either limit, competitors will be fined proportionally to the square of the difference between their Greek and the limit, at a rate of $\$10^{-7}$ per squared excess per second.

All outstanding positions at the end of each period will be automatically closed out at their fair prices, and options will be automatically exercised if they are in-the-money. In each round, your rank will be calculated based on your P&L (including any and all fees) relative to other competitors; winners will be determined by the best average rank across the four rounds.

4 MangoCore API

4.1 Overview

MangoCore interfaces with external clients with a JSON API over a websocket connection. Users can use this API to programmatically query our simulator servers which may be helpful for cases that are suited for higher frequency trading. This guide details how to set up a local server for testing and provides information about the TradersBot Python package (available on pip), which allows easy interfacing with Mangocore's JSON messages.

For all cases, competitors should expect to pull price paths in addition to other information relevant to each case from MangoCore and analyze this data with real-time models. Ideally information from these models should inform competitors' trades. The only case for which traders will be allowed to *submit* orders programmatically through MangoCore is the algorithmic trading case (FX), where competitors can both pull price paths and submit orders for various securities in real-time.

4.2 TradersBot Python Package

As MangoCore interfaces with external clients with a JSON API over a websocket connection, theoretically it is possible to interact with the client through any programming language of choice, provided the user knows how to set up a websocket connection. However, we *highly* recommend that competitors use the TradersBot Python package, which provides a Python wrapper to the MangoCore API that allows for easy websocket setup to interface with the Mangocore server. TradersBot's installation, usage and general documentation are extensively detailed at `http://mangocore-client.readthedocs.io`, which provides instructive examples along with a thorough description and purpose of each feature. This URL is subsequently referred to as the *Documentation*.

In this section, we will broadly describe the functions provided by TradersBot as well as the types of messages that the MangoCore server sends to TradersBot. We also provide details on how to run a local instance of MangoCore to allow for local testing.

4.3 Server setup

MangoCore is run on a server; during competition day, we will be hosting the MangoCore server and provide the necessary IP/URL for competitors to connect to. However, competitors might want to run MangoCore on a local server in order to test their strategies. To run the

server locally, grab the executable for your operating system from the competitor Dropbox and follow the instructions in the instructional PDF in the Dropbox folder. The server will open up a port on localhost:10914 and you can connect to it by opening up a websocket connection to `ws://localhost:10914/#trader0`, although if you are using TradersBot, the websocket connection will be setup for you. *Note: you cannot use your given competition traderid locally because the provided binaries default to a small set of recognized traderids that will not include your assigned traderid.*

To run your algorithm on our server for competition day and practice sessions, switch the host from `localhost:10914` to `m.angocore.com` and `trader0` to your given trader ID.

4.4 TradersBot Setup

There are several parameters when connecting with a MangoCore server: the MangoCore server IP/URL (which will be provided to you on competition day for the competition server and can be run locally with information located in the section Server Setup), a team username, a password, and an optional token.

TradersBot will automatically establish a connection with the server when you instantiate it with necessary parameters. A thorough description of how to use TradersBot, including instantiation, is in the documentation.

4.5 Messages

MangoCore has several types of messages that it will deliver to TradersBot. We describe the types of messages below; the exact information they contain and the message format is described in more technical detail in the documentation. In general, you can expect that each message is delivered as some sort of Python dictionary with keys corresponding to relevant information for that type of message. For example, a trade message might contain a dictionary with a key “ticker" that denotes which stock ticker the trade is referring to.

For each type of message, you can implement a function in your Python script that takes in as input the `message`, and parses the information in the message, possibly analyzing it and/or updating any internal variables you have with the parsed information. TradersBot has an internal function on receiving each type of message that we can set to a user defined function. An example TradersBot use case is in the documentation, which implements functions for each type of message.

We describe the content and purpose of each message below. Again, for specific details about the message format and the dictionary that comes with each message, refer to the documentation. This is meant to outline and guide competitors on the purpose of each type of message.

4.6 Passive Information

4.6.1 onMarketUpdate

Sometimes, you will receive market updates for securities. This generally happens every 500ms as opposed to following every API call that may modify the state of the order books. However, you will receive every trade notification so you may be able to maintain an orderbook estimate that can be more accurate.

Additionally, you can ping the market with micro-orders to discover the top of the book. You will want to do this sparingly because of the trading limits (see the last section).

4.6.2 onTraderUpdate

As with market updates, you will generally receive trader updates every 500ms. However, you will be able to calculate your precise trader state at any moment in time based on other information you receive.

4.6.3 onTrade

You will receive a notification following every trade (not just the ones concerning your orders) immediately (after network/processing latencies of course). You can use this information to keep an accurate internal order book and also have more precise information about the state of the market.

4.6.4 onNews

For quant outcry, you will receive news periodically. Note that for other cases you will also receive news, which will also be sent through the MangoCore API, but unlike the news for quant outcry, this news will not be machine parseable, and likely be of no use to deal with programatically.

4.7 Submitting and Canceling Orders

The only case for which submitting or cancelling orders is allowed through the Mangocore API is the FX algorithmic trading case. In this case, users can submit and cancel orders with the Mangocore API. The rest of this section is in the context of the FX case.

For each message type, the user should have some specified function that takes in as parameters a `message` and also an object `order`, which initially starts off as an empty order. If the user wishes to buy a stock, they should use the method `addBuy` with several parameters, including the ticker of the stock, quantity, price and an optional unique token that identifies the order. If the user wishes to sell a stock, use the method `addSell` with the same parameters. If the user wishes to cancel an already existing order, then use the method `addCancel` with the unique token identifier of the order that they wish to cancel.

More detail about these functions is given in the documentation, along with a clarifying example.

4.8 Message Limits

There are limits for how frequently you may send messages: you may send a maximum of 25 pieces of information per second, where a piece of information is defined as a message, order, or cancel. For example, a message with 6 orders and 2 cancels counts as $1 + 6 + 2 = 9$ pieces of information. Any piece of information past this limit of 25 gets discarded. Also to note, messages are either entirely processed or entirely discarded. In other words, if you have sent 20 pieces of information in the last second and you attempt to send 10 more pieces in your next message, nothing in your next message will be processed (but we will record that you have tried to send 30 pieces of information in this past second).

If you send over 250 pieces of information in any second, your connection will be immediately killed (this isn't because we don't like you; we just don't want you to overload our network and starve other connections).

We will be monitoring server load and if it seems that our limits are too strict, we will relax them.

4.9 More Information

You will run your algorithms on your computers - we will not run them for you. Our server will be running somewhere (possibly in a public cloud, possibly in a box somewhere on MIT, etc). You may wish to optimize latency by trying to find where our server is.

If your client falls more than 5 seconds behind in receiving information, we will kill the connection.

4.10 Backtesting

A brief note about back-testing: profits that you see in your backtesting may not be the same as profits you may see during the actual competition. This may be due to competition (other market makers decreasing your edge), increased latencies (your algorithm will not be running on the same machine as the exchange), and other factors. You may want to simulate these pessimistic conditions in your backtesting to make it more rigorous.