Exercise Week 3. Basic arbitrage trading: dual listings, PnL calculations, position visualization

Data:

You will receive *5* .csv files for the fictive stocks OTZ, BSF, RLI, RMCC, HWG. In each file is a dataset of 5-minutely observations of bid and offer price and volumes on two fictive exchanges: I-XCHNG and Z-XCHNG. On one of the two exchanges the stock is more liquidly traded than in the other, as is visible from the market volumes and widths.

The first column gives the time of the observation, the next four columns, respectively contain the bid price, bid volume, ask price and ask volume on I-XCHNG. The last four columns give the equivalent values on Z-XCHNG.

Assignment:

Design and backtest a basic arbitrage trading strategy. Assume you can trade with immediate-or-cancel (IOC) orders on both exchanges, where you can sell against the bid price, and buy from the offer price, in the volume that is available. You can assume a 100% hit rate, all volume that is available may be traded. However, it is also allowable at any moment to choose less volume to trade. It is further allowed to short sell a stock, e.g. enter into a negative position on an exchange.

Trade only when it is possible to lock in a guaranteed profit. Match bids in one exchange to offers in the other, and trade on both exchanges when they cross.

Keep track of your position on both exchanges as it evolves over time and visualize it. Choose a way to evaluate the value of your portfolio of longs and shorts at each instant and explain why you chose that method. Combine your portfolio valuation and trading results to calculate total profit-and-loss at each time.

Incorporate the following maximum outstanding positions per exchange:

	I-XCHNG	Z-XCHNG
OTZ	250 lots	250 lots
BSF	250 lots	250 lots
RLI	250 lots	250 lots
RMCC	250 lots	250 lots
HWG	250 lots	250 lots

Deliverables:

Graphs of the positions per exchange and profit-and-loss per exchange over time. Visualize both the whole dataset as well as a specific interesting zoomed in subset of the data, e.g. a single day.

Files for each of the stocks in .csv format according to the following example:

Timestamp	Position-I-XCHNG	Position-Z-XCHNG	Profit-and-Loss
2018-01-01 00:00:05	5	-5	50.0
2018-01-01 00:00:10	10	-10	80.5
2018-01-01 00:00:15	15	-15	99.5

Ensure the values are valid **after** applying all the trades done at that timestamp.

The script generating these outputs, cleaned up and able to run without any further modifications.

Extra Materials:

Skeleton Python notebook with starting points.

NOTE: This guideline provides just a single possible method to solve the questions posed in the exercise. If your Python is already strong or you have other ideas, feel free to deviate where you want or to not follow the steps in the guideline at all.

Step-by-step guideline for Python implementation:

- Write the script for a single stock, but make it easy to configure which input file/stock your script is run on.

- Start by exploring the data:
  - Read the dataset for the stock into a pandas DataFrame.
  - Visualize the columns that are in this DataFrame by plotting each of them individually.
  - Are there too many data points to clearly see the behaviour? That can be resolved by limiting the observations of your dataset, e.g. the first or last 250, and/or by increasing the size of your plot.
  - It makes sense to combine the bid and offer prices on both exchanges into a single graph, as the values will be close together. Do so. Choose your line colours or line styles to be able to distinct them. Is it clearly visible when you would want to make an arbitrage trade?

- Check that your code is generic:
  - Change the input dataset to another stock. It might have a different amount of observations, does all your code still function?
  - Restart your notebook, resetting all stored variables, and rerun it from the start. Does everything work as is?

- Write the algorithm:
  - As your algorithm will need to keep track of the position limits per exchange, its behaviour will need to depend on the trades it did before. For this reason it will be easiest to implement the algorithm one instant at a time by looping over each of the observations and timestamps in the input data.
  - We want to keep track of the position at each instant. To store this, you can use a new DataFrame which has the same index as the market data (all the timestamps) and has two columns, one for each exchange. Initially you can leave it empty, or fill it with zeroes/nan's.

  - Now, for each step in time do the following:
    - Obtain the current available prices and volumes on both exchanges, your previous position, and any other relevant numbers.
    - Determine your trading decision based on these numbers. Is there an arbitrage trade available? In what size?
    - If there is a trade available, and you would do it in the full available size, would you still wind up within your limits? Ensure you don't breach them, you might need to limit your trading!
    - Considering all of this, determine what the new position per exchange should be and store it.

- Visualize the trading algorithms position taking alongside the initial dataset, does it do what you expected it to do?

- After you have determined your positions per exchange for each timestamp, use them to calculate your PnL At each instant, this PnL should consist of two parts:
  - The total money you have paid/received for the trades you have done so far (cumulative sum of all trading PnL)
  - The current valuation of the position you are holding
  - Calculate both of these numbers for each timestamp, add them together.

- Visualize all data together in a few plots: Market prices, volumes, positions taken by the algorithm, and PnL. Is it all consistent?

- Finally, collect your positions and PnLs in a single DataFrame and write it out to .csv.