

# HOMWORK 3: DUAL LISTING ARBITRAGE

FRANCISCO GARCÍA FLÓREZ, JORIS VAN LAMMEREN, WOUTER VARENKAMP

**Abstract.** In this homework we study pure arbitrage as a way of making risk-free profits by trading some volume between two markets. Even though the profits per trade are low compared to the amount of cash involved, it is still possible to make a significant profit over time.

## 1 Trading robot

The trading algorithm is implemented in three separate steps, executed every time there is a book update. These three steps are *offers and bids checking*, *trading* and *book updating*, each of them explained in the following sections.

We consider each book to be a set of price and volume pairs, that we denote as  $(P_i^{a,b}, V_i^{a,b})$ , where the superscript  $a, b$  refer to *ask* and *bid* respectively, and  $i \in [1, N_{a,b}]$  is the index of the entry. In a more general setting we would also want to specify the market, however in this case there are only two options: buying CHI\_AKZA and selling EUR\_AKZA or the opposite, so references to bids or asks are implicitly connected to the right market.

**Offers and bids checking** In the first place, right after the robot receives a book update, it computes a matrix of logical values (**True** or **False**) representing which trades are profitable. This matrix  $T$  is computed as follows:

$$T_{ij} = P_i^a < P_j^b, \quad \forall i, j \in [1, N_a] \times [1, N_b] .$$

Since not every time we receive a book update there are going to be any profitable trades, we need to check that at least one of the elements of  $T$  is **True**, which we could check by computing the product of its components:

$$\text{check} = \bigoplus_{i,j}^{N_a, N_b} T_{ij} ,$$

thus if **check** is **True**, we proceed with the next step, in which we compute the data we need for the *trading* part. This data is contained in two matrices, each element composed of a pair of prices and volumes:

$$\begin{aligned} V_{ij} &= (V_i^a, V_j^b), \quad \forall i, j \in [1, N_a] \times [1, N_b] \\ P_{ij} &= (P_i^a, P_j^b), \quad \forall i, j \in [1, N_a] \times [1, N_b] \end{aligned} .$$

However we still need to do one extra computation with  $V$ , since the volume that can be traded is limited by either  $V_i^a$  or  $V_j^b$  the final volume we can trade is given by  $\tilde{V}_{ij} = \min(V_{ij,1}, V_{ij,2})$ , where the subscript 1 and 2 reference each element of the pair.

Finally, selecting components  $\tilde{V}_{ij}$  and  $P_{ij}$  for which  $T_{ij} = \text{True}$ , we can proceed to the next step.

**Trading** Now, taking the values selected in the last step and iterating over them

## 2 Results

As we can see in Table ??

## 3 Discussion

Our main conclusion is that it takes a high investment to make a small profit. This seems like it is not a good investment, but there is no chance on making a loss. So we make a small risk-free profit without the risk of making a loss. The strength of this system is that it only does immediate trades. This minimizes the risk to zero. So you have a guaranteed profit. This is also the weakness of this system. It might pass on a really good price for the share because it can not be sold immediately. So the system could be more profitable if we could implement statistics that can tell the robot that an asset can be very profitable in the nearby future.

## References

- [1] P. Wilmott et al, *The Mathematics of Financial Derivatives*, 1995.

## 4 Appendix

Table 1: Cash moved for each feed.

#	CHI (Buy)	CHI (Sell)	EUR (Buy)	EUR (Sell)	Profit
1	-37407.95	74623.63	-74613.81	37412.86	14.73
2	-57272.87	24620.31	-24617.09	57280.36	10.71
3	-37686.14	29292.16	-29288.31	37691.10	8.81
4	-19650.57	85786.14	-85774.92	19653.14	13.79
5	-66249.23	57756.53	-57748.97	66259.82	18.15

Table 2: Assets moved for each feed.

#	CHI (Buy)	CHI (Sell)	EUR (Buy)	EUR (Sell)	Position
1	491	-982	982	-491	0
2	749	-322	322	-749	0
3	496	-385	385	-496	0
4	257	-1122	1122	-257	0
5	866	-756	756	-866	0