1. Way-of-work
   (1) Have fully understanding of the paper's section and related papers' sections
   (2) Find if there any similar implementation on Github.
   (3) Implement unit function needed for implementation via brute force algorithm, and verify some test cases' answers with hand-calculated answers.
   (4) Implement unit function needed for implementation via more efficient algorithms, and verify test cases' answers with brute force algorithm answers.
   (5) Integrate functions into a class and simplify code as much as possible.
   (6) Add built-in exceptions for the class.
   (7) Write Jupyter Notebook for demo.
2. Design choices
   (1) The paper didn't clearly describe how to determine the top N most highly correlated stocks. I simply take into account top N stocks with the highest Spearman's ρ.
   (2) For calculation of double sum in implementation of extended approach. I use an O(n) runtime algorithm, inspired by this post.
   (3) To speed up calculation of sum of all perpendicular distances in implementation of geometric approach, I use element wise tensor dot product to calculate all dot product at once. It shows out to be about 100 times faster.
   (4) Because the analytical formulas for extremal approach is too complicated, I choose to generate general formulas using SymPy. Although the efficiency will be reduced by doing so, it can be directly expanded to any dimensions.
3. Learnings.
   (1) 4 different ways to describe multivariate correlation.
   (2) Copula. It is a very new thing to me. When I implemented the extremal approach, it took me a long time to figure out what to calculate.
   (3) In order to speed up the efficiency, I searched and learned lots of element wise calculation skills.
4. UML Class Diagram