ACO and GA for Portfolio Optimization

A. Faroux, C. Zeiter

Spring Semester 2025

Adjusted closing prices 2022 to 2024
Red: AAPL, Green: GOOG, Blue: MSFT, Orange: AMZN, Purple: META, Teal: TSLA

Our Problem

Goal: Allocate weights to n assets to maximize portfolio financial return *relative to risk*.

Why it's hard:

- **Combinatorial**: $w_i = 0$ or $w_i \in [w_{\min}, w_{\max}]$.
- **Non-convex objective**: Sharpe ratio is not convex nor differentiable.
- **Constraint-heavy**: Budget, bounds, no short-selling.

Implication: Traditional optimization fails — need population-based metaheuristics

Framing the Problem

Search space: possible weightings of annualized returns of each S&P 500 stock for the period 2022-24: $w = (w_1, \ldots, w_n)^\intercal \in \mathbb{R}^n$. The covariance matrix between the returns of the different assets is given by Σ and the annualized returns by μ.

$$
\begin{aligned}
\text{maximize} \quad & f(w) = \frac{w^\intercal \mu}{\sqrt{w^\intercal \Sigma w}} = S(w) \quad \text{(Sharpe ratio)} \\
\text{subject to} \quad & \sum_{i=1}^{n} w_i = 1 \quad \text{(budget constraint)} \\
& w_i = 0 \quad \text{or} \quad 0.05 \leq w_i \leq 0.3 \quad \forall i \quad \text{(no too small or too large investment)} \\
& \#\{i \mid w_i > 0\} \geq 10 \quad \text{(diversification constraint)} \\
& w_i \geq 0 \quad \forall i \quad \text{(no short selling)}
\end{aligned}
$$

Essentially, Sharpe measures returns compared to volatility of an asset.

ACO: Ant Colony Optimization

- Each ant represents a candidate portfolio allocation strategy.
- Pheromone trails τ_i encode the desirability of investing in each asset i.

Algorithm 1 Ant Colony Optimization for Portfolio Selection

Initialize: pheromones $\tau_i \leftarrow 1$ for all assets i, best Sharpe ratio $S^* \leftarrow -\infty$
while elapsed time $<$ `max_seconds` **do**
    for ant $k = 1$ to m **do**
        $w_k \sim$ Dirichlet$(\tau_i \cdot \alpha + 0.01)$ and enforce constraints
        Evaluate Sharpe ratio $S_k = f(w_k)$
        if $S_k > S^*$ **then** $S^* \leftarrow S_k$, $w^* \leftarrow w_k$
        end if
    end for
    Find best ant: $k_{best} = \arg\max_k S_k$
    Evaporate: $\tau_i \leftarrow (1-\rho)\tau_i$
    Deposit: $\tau_i \leftarrow \tau_i + w_{k_{best},i} \cdot \max(0, S_{k_{best}})$
end while
return w^*, S^*

GA: Genetic Algorithm

- Each individual represents a candidate portfolio allocation strategy.
- Individuals evolve through crossover, mutation, and tournament selection with elitism.

Algorithm 2 Genetic Algorithm for Portfolio Selection

```
Initialize: population P_0 ~ Dirichlet(1), Hall of Fame H, S* ← −∞
while elapsed time < max_seconds do
    Evaluate: S_i = f(w_i) for all w_i ∈ P_t
    Update H ← top n_elite individuals from P_t ∪ H
    Offspring ← Crossover(P_t, c_x) + Mutate(P_t, p_m, p_b)
    Evaluate offspring; enforce constraints
    P_{t+1} ← Tournament selection from offspring + elite(H)
    if max(H) > S* then S* ← max(H), w* ← arg max(H)
    end if
end while
return w*, S*
```

Enforcing Constraints: Intuition

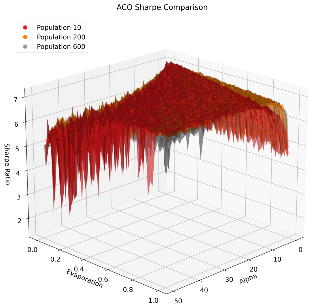
- For each iteration of the algorithms we receive portfolio weights $w = (w_1, \ldots, w_n)$, which may violate investment constraints.
- We enforce constraints sequentially to ensure valid portfolios:
  1. **Cap oversized weights:** If any $w_i > w_{\max}$, cap it at $w_{\max}$ and redistribute the excess to assets with smaller weights proportionally.
  2. **Remove tiny positions:** If $0 < w_i < w_{\min}$, set $w_i = 0$ and redistribute to remaining active positions proportionally.
  3. **Handle edge cases:** If all weights become zero (or nearly zero), randomly select $\min(\texttt{minactive}, n)$ assets and assign equal weights.
  4. **Ensure minimum diversification:** Guarantee that at least `minactive` assets have positive weights.
  5. **Normalize:** Rescale final weights so $\sum_{i=1}^{n} w_i = 1$ (budget constraint).

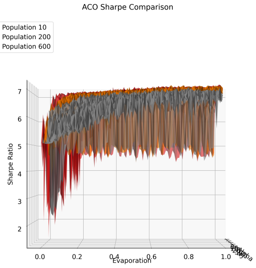
Hyperparameters Tuned per Algorithm

Algorithm	**Hyperparameters Tuned**
Ant Colony Optimization (ACO)	α (pheromone influence), ρ (evaporation rate), m (number of ants)
Genetic Algorithm (GA)	c_x (crossover probability), p_m (mutation probability), p_b (bitwise mutation probability), P (population size)

- Fixed the other parameters for GA after initial experiments to: $n_e = 3$ size of elitary group and $n_t = 3$ tournament size.
- Runs on the ETH Euler cluster to find optimal parameters for our problem by trying various combinations of parameters.

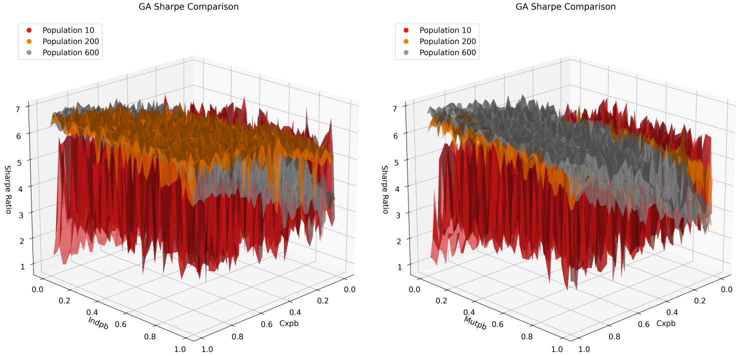
ACO Parameter Tuning





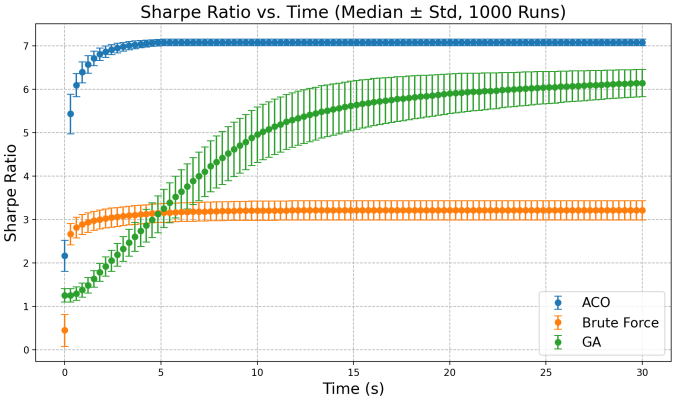
Based on the results, $\alpha = 20$, $\rho = 0.7$, and $m = 200$ were selected as the final parameters. Although a smaller number of ants can achieve slightly higher Sharpe ratios, $m = 200$ was found to yield the most consistently good results.
The same experiment was also conducted for $m \in \{10, 100, 200, 300, 600, 1000\}$. Only the most illustrative examples are shown for clarity.

GA Parameter Tuning



Like in ACO $P \in \{10, 100, 200, 300, 600, 1000\}$ was tested. The population size of 200 also returned the most consistent results with the highest Sharpe ratios.
Based on these results, the optimal parameters $c_x = 0.7$, $p_m = 0.1$, $p_b = 0.3$, and $P = 200$ were chosen.

Sharpe vs Time



ACO converges much faster than GA. Brute force (used as a benchmark algorithm) is much less efficient. Note that in the first few seconds brute force performs better than GA.

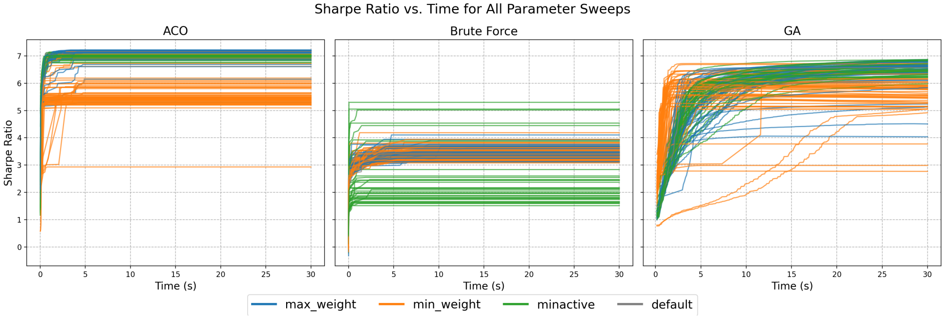
Iterations Speeds

Algorithm	Iterations / Second
Brute Force	1279
ACO	6.6
GA	1.0

This table contrasts with the previous slide.

- Brute unsurprisingly has many iterations per second.
- ACO and GA in the previous plot were quite close.
- If the implementation of GA was more efficient, we would probably reach even closer performance.

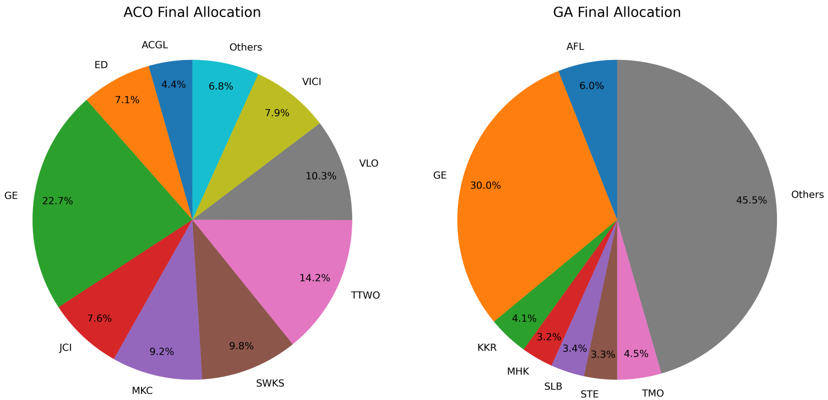
Impact of Boundary Conditions



We run each algorithm with the initial boundary conditions and vary one of them.

- ACO is only considerably impacted when varying the minimal weight boundary condition.
- Brute Force performs generally worse than ACO and GA. Relaxing the minimal required number assets greatly improves performance.
- While the convergence time of GA is strongly impacted by the minimal number of chosen assets, the Sharpe ratio after convergence is limited more by the minimal weight constraint.

Final Asset Allocation



Even though we get similar Sharpe ratios, the allocations are very different (stock tickers written next to pie chart).

What we Learned

- **ACO Outperforms on Historical Data**
  $\Rightarrow$ ACO's pheromone reinforcement suits historical optimization, because successful assets get stronger selection bias. This would probably not work as well on future expected returns.
  $\Rightarrow$ Interestingly our custom implementation of ACO had considerably better performance than GA (written with DEAP library which one would expect to be optimized for performance).
- **Constraints and Enforcement Overhead**
  $\Rightarrow$ The choice of boundary conditions can not only have an impact on the quality of a result, but also on the performance of the algorithm.
  $\Rightarrow$ Dirichlet sampling frequently violates weight constraints, forcing repeated constraint corrections in ACO (not in GA). This additional overhead could probably be reduced by using a better weight-distribution logic. Future research could focus on improving ACO in this regard.
- **Solution Quality vs. Diversity**
  $\Rightarrow$ Both algorithms achieve comparable Sharpe ratios but discover fundamentally different portfolio allocations, suggesting complementary exploration strategies.

References

- Kamolsin, Chiranun and Visutsak, Porawat - Solving Portfolio Optimization Problem for Long-term Stocks Investment using Ant Colony Optimization
- Haqiqi, Kambiz Forqandoost and Kazemi, Tohid - Ant Colony Optimization Approach to Portfolio Optimization – A Lingo Companion
- Steven A., Hertono G. F., Handari B. D. - Clustered stocks weighting with ant colony optimization in portfolio optimization