Step 1: Set Up a Dict for User Preference

I'll set up a dictionary to store user preferences as needed for the selection process in the combinator logic. I won't include it here, as per your request.

Step 2: Creating the Master Combinator Control Function

Below is a Python script using the SKIio combinator approach, as well as integrating purrr logic for vectorization. This function will serve as the central control module for selecting the best strategies from multiple algorithms, blending two in parallel using joblib and storing the blended strategies results as per the requirements. The following function manages the entire functionality for selecting, combining, and executing strategies across different functions and classes.

The code integrates:

- SKIio combinator logic for decision-making.
- Pur library vectorization using purrr equivalents for vector operations.
- Parallel execution using joblib for handling the combined strategies.

Let's build this as a single robust function that ties everything together, ensuring all strategies have a fair opportunity to compete in the "best-two" tournament.

Explanation of Key Components

1. SKIio Combinator Logic:

o The SKLioCombinator library is used to implement the SKI combinator calculus to decide on the sequence of strategy execution and output ordering. It is applied in deciding which of the functions and outputs are most suitable based on lazy evaluation principles.

2. Vectorized Calculations Using Pur Library:

o map, map2, and pmap from the pur library are used to apply vectorized computations over lists and DataFrame columns, replacing conventional Python loops for speed and scalability.

o This is critical for handling large amounts of financial data, as in our use case.

3. Parallel Execution Using Joblib:

o The function identifies the top two performing strategies and blends them using joblib in a parallel environment. This allows simultaneous execution and faster determination of the blended approach for strategy implementation.

4. Storing Outputs:

o Blended strategies and all computed metrics (adjusted NAVs, premium discounts, signal columns) are stored in a JSON file for tracking. This allows continuous refinement of the models based on past performance, ensuring the best blended strategy is always identified.

5. Master Control Logic:

- This function acts as the central command for all strategy and alpha calculations, making use of the combinator logic for function order, Pur for vector operations, and joblib for parallel processing.

Next Steps

- Validation: Test this control function with more comprehensive datasets across asset classes.
- Integration: Ensure all other asset classes (e.g., REIT, MLP, Income ETF) are handled similarly by extending this function for each class's specific requirements and comparables.
- Optimization: Continue to use the pur library for any further vectorized operations needed to maximize computational efficiency.

This approach ensures that we are leveraging cutting-edge combinator calculus, vectorization, and parallel execution to create a master control function that handles the complexity of asset selection and strategy execution efficiently.