Paper 3: CVaR Adjusted Update Module

Formulation

The Conditional Value-at-Risk (CVaR) adjustment explicitly targets tail risk beyond Value-at-Risk (VaR). The adjustment factor is defined as:

$$\text{Adj} = e^{-\text{CVaR}}$$

Updated weights:

$$w' = \frac{w \times \text{Adj}}{\sum w \times \text{Adj}}$$

Where:

- CVaR: Average loss beyond a specified quantile (tail).
- w: Current portfolio weights.

Why this Works

The exponential damping factor penalizes higher expected losses in the tail, shifting portfolio allocation toward safer or lower-risk assets when tail risk is high.

Algorithm Steps

1. Simulate or observe return samples.
2. Compute losses as $L = -R$, where R are returns.
3. Calculate VaR at desired level α.
4. Identify losses exceeding VaR and compute their mean (CVaR).
5. Apply exponential adjustment to current weights.
6. Normalize the weights to ensure they sum to 1.

Code Equivalent

```
def cvar_adjusted_update(weights, simulated_returns, alpha=0.05):
    losses = -simulated_returns
    var = np.percentile(losses, 100 * (1 - alpha))
    cvar = losses[losses >= var].mean()
    adjustment = np.exp(-cvar)
    new_weights = weights * adjustment
    new_weights = np.maximum(new_weights, 1e-5)
    new_weights /= np.sum(new_weights)
    return new_weights

# Example usage:
simulated_rets = returns.iloc[t].values @ weights
weights = cvar_adjusted_update(weights, simulated_rets, alpha=0.05)
```

Summary

- Explicitly controls downside tail risk.
- Encourages stability in extreme market scenarios.
- Widely supported in robust optimization literature.