Gaussian Filtering for Stable Parameter Selection

Jeff Tuche :)

February 14, 2025

1 Introduction

After performing Random Forest (RF) analysis to identify robust parameter sets based on Train Sharpe, Test Sharpe, and Sharpe Degradation, we encountered a significant issue:

Some of the selected parameters were far from each other in parameter space, making direct averaging or simple selection unreliable.

To address this, we apply **Gaussian Filtering** using Kernel Density Estimation (KDE), which ensures that our final selected parameters are located in high-density clusters, eliminating isolated outliers.

2 Random Forest Selection Recap

The RF model was trained to predict three key performance metrics:

- **Train Sharpe**: Measures strategy performance on training data.
- **Test Sharpe**: Measures how well the strategy generalizes on unseen test data.
- **Sharpe Degradation**: Measures the stability of the parameter set between train and test periods.

The output of RF was a filtered list of parameter sets that were predicted to perform well. However, RF does not account for the spatial distribution of parameters in the search space. This means that some selected parameters may be **far apart and not part of any robust cluster**.

2.1 Example of RF Output

Problem: In the above table, the first two parameter sets are similar and close to each other, but the third set (with btc_ma_window = 30) is significantly different. If we compute a simple average, we may introduce misleading results.

btc_ma_window	btc_size	btc_envelope_1	btc_envelope_2	Train Sharpe	Test Sharpe
10	0.06	0.07	0.12	2.4	2.0
14	0.08	0.08	0.14	2.1	2.0
30	0.05	0.06	0.15	2.5	2.2

Table 1: Examples of selected parameter sets after RF filtering

3 Why Gaussian Filtering?

Gaussian Filtering via Kernel Density Estimation (KDE) solves this issue by:

- Detecting **clusters of stable parameters**.
- Eliminating **isolated outliers** that were selected by RF but do not belong to a robust group.
- Ensuring the final parameter set is **surrounded by other robust parameters**, making it less likely to be an overfitted selection.

4 How Gaussian Filtering Works

4.1 Step 1: Standardization and PCA

To ensure all parameter scales are comparable, we standardize the selected parameters:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma} \tag{1}$$

If the number of parameters is large, we apply Principal Component Analysis (PCA) to reduce dimensionality:

$$X_{\text{PCA}} = PX_{\text{scaled}} \tag{2}$$

where P is the projection matrix that retains the top components explaining the most variance.

4.2 Step 2: Estimating Parameter Density

We then estimate the density of parameter distributions using KDE:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right) \tag{3}$$

where:

- K is the Gaussian kernel function,
- h is the bandwidth parameter,
- X_i are the sampled data points.

4.3 Step 3: Selecting Stable Parameters

A threshold is applied to select the top 30% densest points:

$$\text{stability\_threshold} = \text{percentile}(\text{density\_scores}, 70) \tag{4}$$

Only parameter sets satisfying:

$$\text{density\_score} \geq \text{stability\_threshold} \tag{5}$$

are kept.

4.4 Step 4: Computing Final Stability Ranges

For each parameter, we compute:

- **Mean and Standard Deviation:** Center and spread of stable values.
- **Interquartile Range (IQR):** The middle 50% of values to remove outliers.
- **Median:** More robust than mean in the presence of skewed data.

Example Output:

Parameter	Stable Range (Q25-Q75)	Mean	Std Dev
btc_ma_window	[7, 11]	9.2	3.0
btc_size	[0.06, 0.09]	0.075	0.024
btc_envelope_1	[0.07, 0.08]	0.072	0.011

Table 2: Stable parameter ranges after Gaussian Filtering

5 Example Walkthrough

Before Gaussian Filtering: Selected parameter values from RF:

$$[5, 7, 8, 10, 12, 30, 32] \tag{6}$$

The average of all values would misleadingly be 14.85, which is far from the stable cluster.

After Gaussian Filtering:

- KDE finds a high-density cluster around $[7, 8, 10, 12]$.
- It removes outliers $5, 30, 32$.
- The final stable range is computed only on the cluster.

6 Conclusion

Gaussian Filtering is an essential post-processing step after RF selection because it ensures that:

- Only parameter sets forming a **dense, stable cluster** are used.
- We avoid selecting isolated high-performing but non-robust parameter sets.
- The final parameter ranges are computed based on reliable, high-density areas.

Next Steps: With these filtered parameters, we can:

- Perform Bayesian Optimization within these stable regions.
- Validate the final parameter sets on additional out-of-sample data.