Parallelised Time Series Spike Detection on Hadoop

**FlexiScore**: Identifying Energy Supply Points with Flexible Electricity Consumption

**Background**

In Great Britain's energy market, there are two roughly equal day rates for the cost of electricity supplied: (1) wholesale energy price and (2) transmission and distribution costs ("T&D").

Wholesale price varies at different times of the day. For instance, power price at peak periods can be over ten times more expensive than off-peak periods within the same day. This creates incentives for suppliers to encourage customers to shift their energy load to periods where wholesale prices are considerably lower.

With smart meters, energy consumption at every supply point is recorded at half-hourly intervals across Great Britain. Such data can be used to establish an individual load profile for each and every supply point. Supply points with flexible consumption can be identified if their consumption deviates from their usual behaviour (i.e. does not conform very well to its presumed-Broken symmetry). FlexiScore is used to quantify the degree of flexibility and allow comparison between different supply points across the entire portfolio.

**Data Preprocessing**

Data in the real world is rarely perfect. For instance, duplicate and missing values can be found in many places. This can be due to many legitimate reasons, such as meter removal, signal interruption, system upgrades, etc.

A step of data preprocessing is introduced to mitigate the problem of imperfect data. Duplicate data is systematically resolved and missing data is permutated. This process creates a continuous (i.e. gapless) stream of regularly spaced time series data.

**Metropolis Hastings Algorithm (MCMC sampler)**

Metropolis-Metropolis sampling combing algorithm can be used to draw samples from a distribution. The key feature of the sampling algorithm is that the resampled values imitate the original target distribution. This makes it a preferable tool to permute missing data. Afterwards, the algorithm is solved and missing data is permuted. This process creates a continuous (i.e. gapless) stream of regularly spaced time series data, seasonal models can be established and then the residuals can be passed on to the next step.

**Seasonal Model**

Using the regularly spaced time series data, seasonal models can be established and then the residuals can be passed on to the next step for further processing. Depending on the context, any time series technique can be used such as ARIMA models. Our implementation chose to use Fourier transforms instead due to its fast performance over large amount of data. Fourier transform is traditionally used in audio analysis but it can also be used to establish seasonal time series models.

The algorithm first calculates the proposal distribution using the original data (i.e. non-missing) data. Afterwards, the Metropolis-Hasting chain begins with a randomly selected position alongside a proposed jump. The proposed jump will be accepted if the randomly drawn number is smaller than the log acceptance ratio, otherwise the proposed jump will be rejected and the chain will remain unchanged. The Metropolis-Hasting chain is a Markov chain that is designed to produce a continuous stream of samples that are drawn from a stationary distribution. This process repeats iteratively until the percentage exceeds the predefined threshold. Once the iteration process finishes, final seasonal model is returned alongside the model residuals.

**FlexiScore**

The FlexiScore is defined as the percentage of spikey days within a given period. A spikey day indicates that the supply point is in a state where consumption exceeds some threshold.

FlexiScore is simply defined as the percentage of spikey days within a given period. A spikey day indicates that the supply point is in a state where consumption exceeds some threshold.

The FlexiScore is defined as:

\[
\text{FlexiScore} = \frac{\text{Number of spikey days}}{\text{Total number of days}} \times 100
\]

**Spike Detection**

The algorithm calculates the per-supply-point flexi-score, the score should be zero or one and free from any missing values. After the above steps, we would expect a graph with exactly 48 valid consumption values per day. This data can be reduced into lower granularity (i.e. fewer values) through use of aggregation. This serves several purposes.

First, it reduces the amount of data we are dealing with, which can drastically speed up the computation process. Secondly, by aggregating consumption values into a wider window (e.g. 1 hour, 4 hours, or even 24 hours) we can effectively adjust the granularity of flexibility measurement. Lastly, this is a prerequisite of the algorithm which is largely determined by the resolution of the business control. The current FlexiScore implementation uses data aggregated to hourly level (i.e. 48 values per day).

**Score Calculation**

The FlexiScore value is set and the first few members from the chain are discarded. This is because earlier members of the chain are likely sampled when the target distribution is far from the stationary distribution. This process repeats iteratively until the percentage exceeds the predefined threshold. Once the iteration process finishes, final seasonal model is returned alongside the model residuals.

**Technical Architecture**

The technical architecture consists of two main components: (1) the parallelised time series spike detection component and (2) the FlexiScore component. The parallelised time series spike detection component is responsible for identifying spikey days within a given period. The FlexiScore component is responsible for calculating the FlexiScore value for each supply point. The two components are integrated to provide a comprehensive solution for identifying and evaluating the flexibility of supply points.