Signal enhancement:

Signals in raw form are not always usable, we need to pass them through the filtering layer to ensure the bare minimum quality for further analysis. So first let us try to understand signal filtering and its significance, and post that will understand moving average filtering methods.

Signal Filtering

Signal filtering is a primary pre-processing step that is used in most signal-processing applications. The raw signal is not always in a usable form to perform advanced analysis i.e. various noises are present in the raw signal. We have to apply a filter in order to reduce the noise in the signal as a part of the pre-processing step.
Signal filtering is a process used to manipulate or modify a signal to extract desired information or remove unwanted components. It involves applying a filter, which is a mathematical algorithm or system, to the input signal. Signal filtering finds applications in various fields, including telecommunications, audio processing, image processing, biomedical signal processing, and control systems.

There are many pre-processing steps applied one of which is de-noising which is essential when signals are sampled from the surrounding environment. The moving average filter is one such filter that is used to reduce random noise in most of the signals in the time domain.

Now that we have understood the significance of signal filtering, let us understand the moving average filtering.

Moving Average Filter

Moving Average Filter is a Finite Impulse Response (FIR) Filter smoothing filter used for smoothing the signal from short-term overshoots or noisy fluctuations and helps in retaining the true signal representation or retaining sharp step response. It is a simple yet elegant statistical tool for de-noising signals in the time domain.

These filters are a favourite for most Digital Signal Processing (DSP) applications dealing with time-series data. It is simple, fast, and shows amazing results by suppressing noise and retaining the sharp step response. This makes it one of the optimal choices for time-domain encoded signals.

The Moving Average filter is a good smoothing filter in the time domain but a terrible filter in the frequency domain. In applications where only time-domain processing is present Moving average filters shine, but in applications where information is encoded in both time and frequency or the frequency domain solely it can be a terrible option to choose.

Time Domain Signals: Signals are represented as Amplitude vs Time, refer to the below figure 2.
Frequency Domain Signals: Signals are represented as Amplitude vs Frequency, refer to the below figure 2.

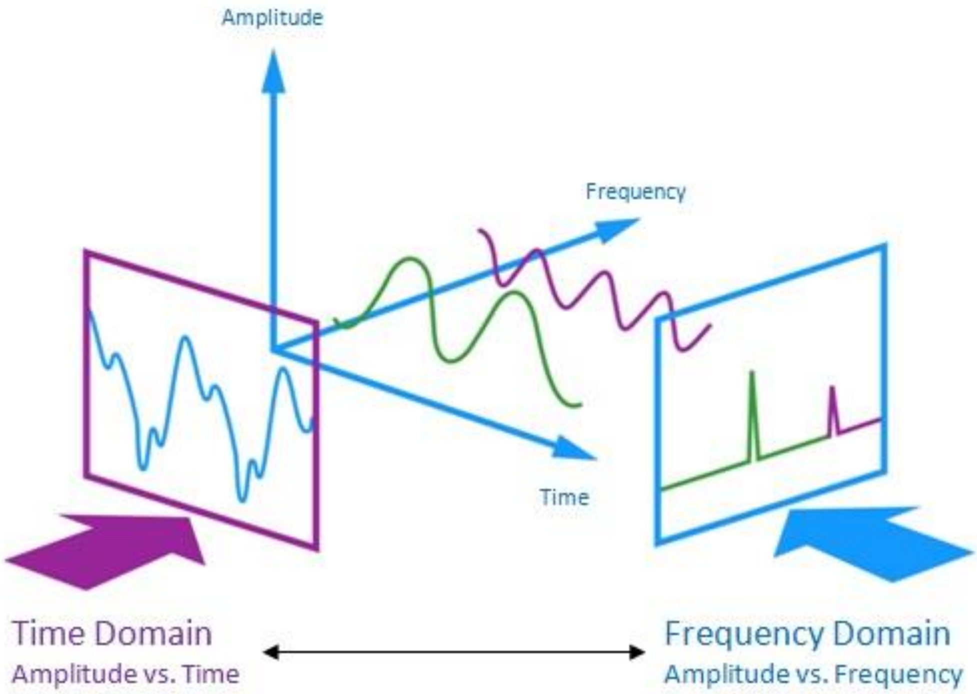

Fig. 2: Time-Frequency Domain Analysis

Types of Moving Average Filter

There are various types of moving average filters but on a broader level simple, cumulative moving average, weighted moving average, and exponentially weighted average filters form the basic block for most of the other variants. There are many moving average filter variants, more or less the fundamental structure boils down to four core types illustrated below figure.

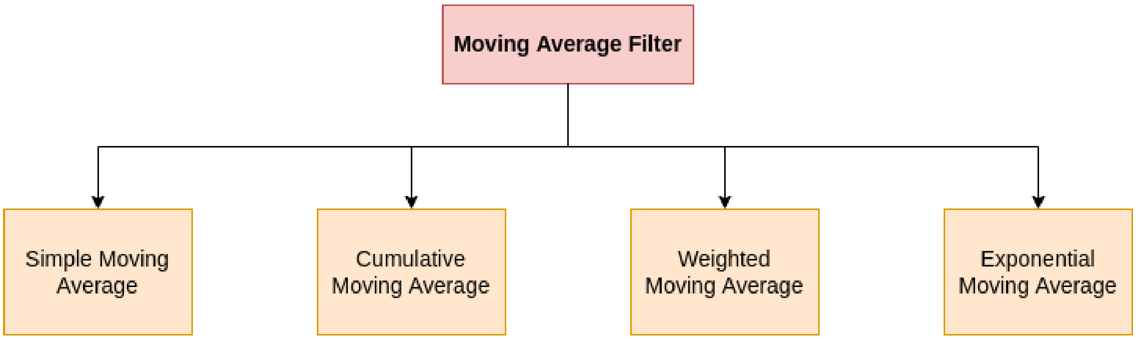

Fig. 3: Types of Moving Average Filter.

This article will cover most of them on a broader level and will show use cases of it, we will also understand their variants which we had used in our research use case, and boosted performance metrics.

Now let us, deep-dive, into each of these types along with code snippets and some basic mathematical formulation.

Don’t get intimidated by coding and mathematics I have tried to keep it short and simple.

Simple Moving Average (SMA)

This is one of the simplest forms of moving average filter that is easy to understand and apply to the desired application. The main advantage of the simple moving average is that we don't need exorbitant mathematics to understand it i.e it can be interpreted by its formula itself.

The con of SMA is that it gives equal weightage to all samples due to which it does not suppress the noisy signal effectively.

Let's take an example.

Let's say we have an array of numbers $a1,a2,...,an$

If we take periodicity (window length) as k, then the average of 'k' elements would be

For easy understanding, let's assume k = 4

$$SMA1=a1+a2+a3+a4$$

$$SMA_2=a_2+a_3+a_4+a_5$$

$$SMA_{N-k+1}=a_n+a_{n-1}+a_{n-2}+a_{n-3}$$

So simple moving average filter equation result would be

SMA Array = $[SMA_1, SMA_2, \cdots, SMA_{N-k+1}]$ which contains 'N-K+1' elements.

Let us implement this simple moving average filter using Python. We will be using the convolution concept for convolving the input signal with all ones with the given window size.

Convolution is a mathematical way of combining two signals to form a third signal.
We will choose a simple sine wave and superimpose random noise and demonstrate how effective is a simple moving average filter for reducing noise and restoring to the original signal waveform.

Following is the plot showing the effectiveness of the simple moving average filter for random noise reduction:-

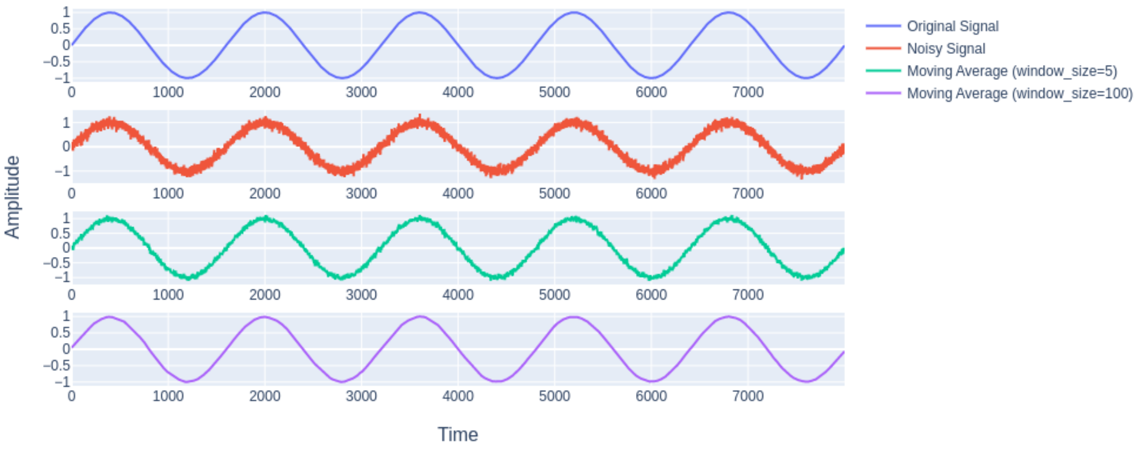

Fig. 4: Effectiveness of the simple moving average filter for random noise reduction.

One characteristic of the SMA is that if the data has a periodic fluctuation, then applying an SMA of that period will eliminate that variation (the average always contains one complete cycle). But a perfectly regular cycle is rarely encountered.

Cumulative Moving Average (CMA)

CMA is a bit deviated from other types in moving average family and the usability of this filter in noise reduction is nil. One benefit of CMA is that it also accounts for past data considerably by also accounting for the recent data point, unlike SMA which will be just an average of past data points within a defined sliding window size and equal weights.

The simple moving average has a sliding window of constant size k, contrary to the Cumulative Moving Average in which the window size becomes larger as time passes during computation.

$$CMA_n = n x_1 + x_2 + \cdots + x_n$$

To reduce computational overhead we use a generalised form:-

$$CMA_{n+1} = n+1 x_{n+1} + n.CMA_n$$

This is called cumulative since the ‘n+1’ terms will also account for the cumulative of ‘n’ previous data points while averaging the points.

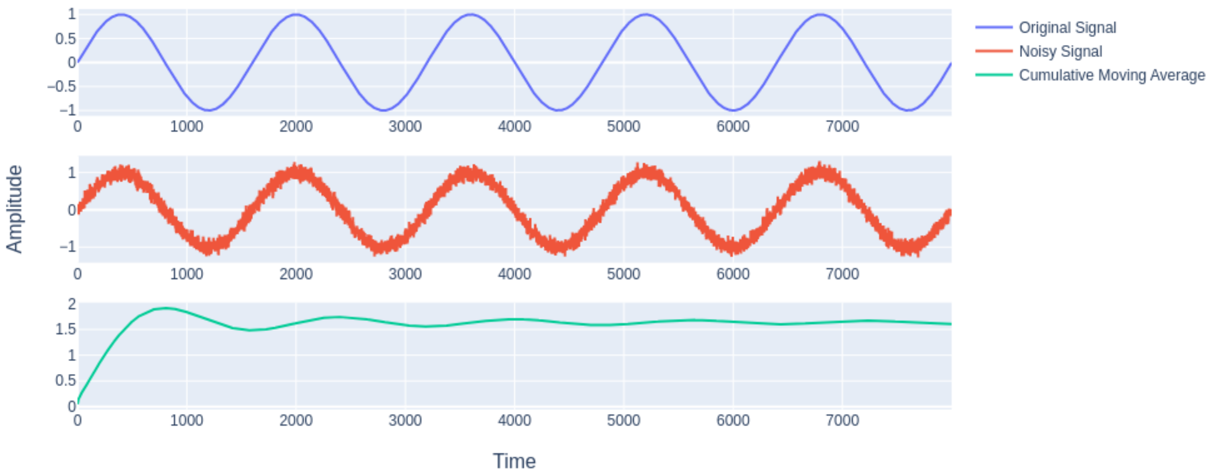

Fig. 5: Illustration of Cumulative Moving Average.

It is very evident that CMA is not at all useful for reducing noise since the retrieved signal is nowhere as pristine in shape as the original signal. This filter is worst to be used for reducing noise pursuit.

But what we can learn from its formula and concept is we can give certain weights not necessarily equal i.e. playing with weights and adjustments that was a setback in the case of SMA. This is where our next topic will come into play Weighted Moving Average.

One of the interesting applications is in the stock market where data stream arrives in an orderly manner, an investor may want the average price of all trades for a particular financial instrument up until the current time. As each new transaction occurs, the average price at the time of the transaction can be calculated for all transactions up to that point using the cumulative average

Fig. 6: Illustration of CMA application

Weighted Moving Average

Weighted Average works similarly to that of a simple moving average filter, where we are considering the relative proportion of each data point within a window length. SMA is unweighted means while WMA contains weight for relative proportion adjustments which makes it even more captivating for application.

Let's take an example to get a better understanding of this. Let's say we have an array of numbers $X1, X2, \cdots, Xn$. Let's keep the periodicity or window length the same as in the previous example i.e k =4

The general Weighted Average formula is

$$\sum_{i=11}^{N} W_i x_i$$

The purpose of giving weights is to give more importance to a few data points over the other data points. And this weight depends on the application. One of the applications is in image processing to specify the filters we use. Those filter coefficients are nothing but the weights and the local patch of pixel values to which the filters are being applied serves as the data points and the combination serves as the weighted average. There are many filters for example blurring filters (nothing but equally weighted filters).

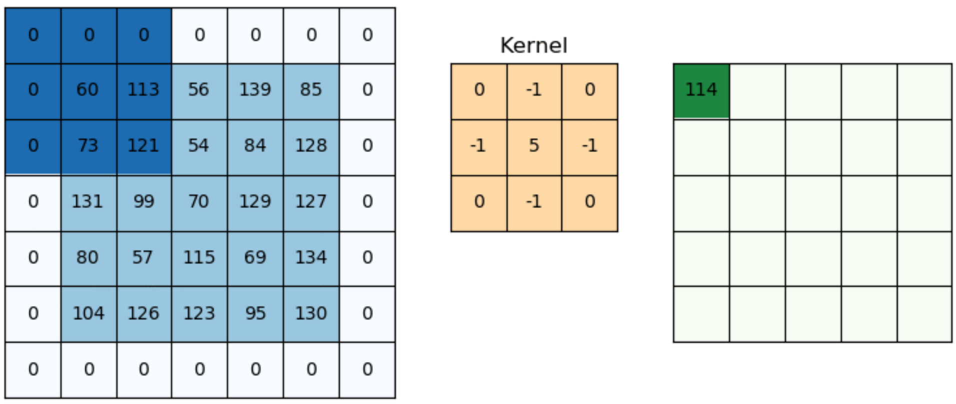

Fig. 7: Illustration of weights (3x3 matrix) applied to the image(i.e. input samples).

The characteristics of weights will decide what effects we want given the input samples. It is quite interesting to understand this part as it is some transformation it does with input samples to get a different perspective of input samples in a much better format and the result is better interpretability.

These WMA techniques are also being used extensively in stock markets but more than this its better variant is used like EWA or EWMA.

Exponentially Weighted Average (EWA) or Exponential Weighted Moving Average (EWMA)

An exponentially weighted average gives more weightage to recent data points and less to previous data points overall. This ensures the trend is maintained by still accounting for a decent portion of the reactive nature of recent data points. In comparison with SMA, this filter suppresses noisy components very well i.e. effective and optimal in de-noising signals in the time domain. It is 1st order infinite impulse response(IIR) filter.

The general formula is

$$EMA_t = \begin{cases} x_0 & t = 0 \\ \beta \cdot x_t + (1-\beta) \cdot EMA_{t-1} & t \geq 1 \end{cases}$$

For different applications, the beta coefficient definition changes but overall the meaning remains the same giving more importance to recent data points and less to previously averaged data points.

The EWMA is a recursive function, recursive property leads to exponentially decaying weights.

$$EMA_t = \beta \cdot x_t + (1-\beta) \cdot EMA_{t-1}$$

$$EMA_t = \beta \cdot x_t + (1-\beta) \cdot (\beta \cdot x_{t-1} + (1-\beta) \cdot EMA_{t-2})$$

$$EMA_t = \beta \cdot x_t + \beta \cdot x_{t-1} - \beta^2 \cdot x_{t-1} + (1-\beta)^2 \cdot EMA_{t-2}$$

$(1-\beta)^k(1-\beta)^k$ will keep on exponentially decaying as shown in the figure below.