

Exponential Smoothing

Philipp K. Janert
(Dated: February 2006)

We look at different ways to produce a “smoothed” curve from noisy data and introduce single and double exponential smoothing as simple and particularly expedient smoothing algorithms.

TIME SERIES

Whenever we want to follow the development of some random quantity over time, we are dealing with a *Time Series*. Time series are very common, and are familiar from the general media: charts of stock prices, popularity ratings of politicians, and temperature curves are all examples. Whenever somebody uses the word “trend”, you know we are dealing with a time series.

Note that studying the time development of some stochastic (i.e. random) quantity over time is different, and more subtle, than just studying the averages of some quantity: To know that some stock cost *on average* \$50.- last year does not help you at all if you bought it at its maximum for \$100.- and sold it at its minimum for \$10.-. To take another example: the average temperature at some location is going to vary drastically, both on a rather long timescale (winter vs. summer), as well as on a much shorter (day vs. night) timescale. Giving only the average means missing out on a lot of relevant action. (Don’t laugh: analyses of this sort are much more common than anybody would want to admit.)

One more thing: To speak of a time series, some form of randomness has to be present. The fully predictable position of a ball bearing rolling down an incline, or of a pendulum, regularly swinging back and forth, also are examples of some quantity changing over time, but we would probably not be referring to either as a time series. On the other hand, as soon as some noise enters the system, because of friction, or oscillations of the support, or through some other random process, the term applies again. Clearly, the boundaries are somewhat fluid.

The first thing that comes to mind when we follow the behavior of a noisy signal over time is to ask for some way to be able to distinguish the “important” trends from the “noise”. Colloquially, this is known as “smoothing”.

Important Note: For the rest of this article, we will assume all observations to be gathered at equally space time intervals!

FLOATING AVERAGES

The best well-known and most commonly applied smoothing technique is the *Floating Average*. The idea is very simple: for any odd number of consecutive points, replace the center-most value with the average of the other points:

$$x_i \rightarrow \frac{1}{2k+1} \sum_{i=-k}^k x_{i+k}$$

In this formula, all the x_i have the same weight, but it may be reasonable to require points towards the center of the smoothing interval to be more important relative to the others. We can introduce weight factors into the sum to obtain a *weighted floating average*:

$$x_i \rightarrow \sum_{i=-k}^k w_k x_{i+k} \quad \sum_{i=-k}^k w_k = 1$$

The weights are usually chosen symmetrically around the mid-point, for instance $(\frac{1}{4}, \frac{1}{2}, \frac{1}{4})$ for $k = 1$ or $(\frac{1}{16}, \frac{1}{4}, \frac{3}{8}, \frac{1}{4}, \frac{1}{16})$ for $k = 2$. More complicated sets of weights exist for special applications, such as the 15-point “Spencer” moving average, which is used to calculate mortality statistics for the insurance industry and which contains *negative* weights as well as positive ones.

Straightforward as this approach is, it has nevertheless several problems:

- The first and last a points cannot be smoothed. While the beginning of a time series is usually not of great interest, it can be a real drawback not to have a smoothed value at the leading edge, where all the action is happening — in particular if a large k is required to achieve the desired smoothness.
- The calculation is slightly tedious, since for each smoothed point the entire interval has to be summed again, in particular if we use unequal weights. This implies that we must keep track of a sufficient amount of history, even if all we want to do is update the leading edge of the smoothed curve.
- It is not possible to extract something like a trend from the floating average, making extrapolation of the time series impossible.

Interestingly, there exists a surprisingly unknown, simple little scheme called *Exponential Smoothing*, which addresses all of these points. There are three different forms of exponential smoothing, known as single, double (*Holt-*), and triple (*Holt-Winters-* exponential smoothing. The

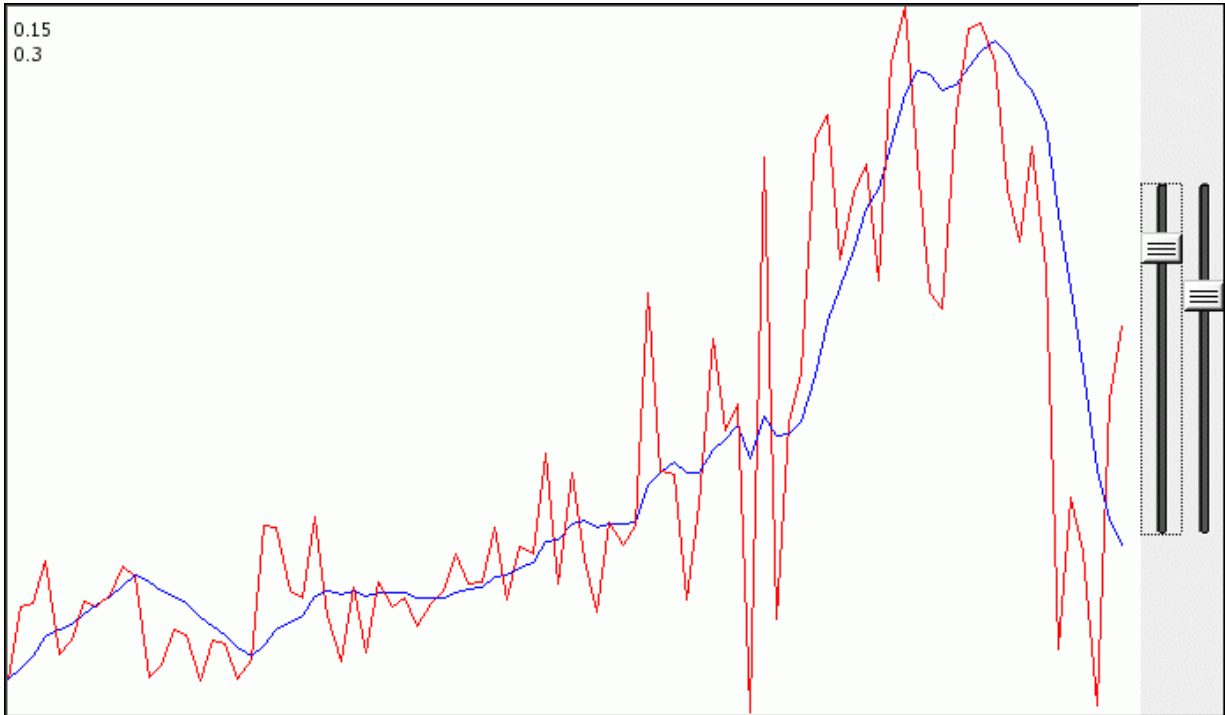


FIG. 1: A time series with a double-exponential smoothed curve

first one finds a smooth approximation to a noisy signal, the second also allows to extract a linear trend, and the third one takes into account periodic (i.e. regularly recurring) variations.

EXPONENTIAL SMOOTHING

All exponential smoothing methods are conveniently written as *recurrence relations*: the next value is calculated from the previous one (or ones). For single expo-

ponential smoothing, the formula is very simple (x_i is the noisy data, s_i is the corresponding “smoothed” value):

$$x_i \rightarrow s_i = \alpha x_i + (1 - \alpha)s_{i-1} \quad 0 < \alpha \leq 1$$

The parameter α controls the amount of smoothing — if $\alpha = 1$, the curve is not smoothed at all; if $\alpha = 0$, the curve is absolutely smooth, in fact, it shows no variation at all (i.e. it is a flat line)!!

Why is this method called *exponential* smoothing? To see this, it is useful to expand the recursion:

$$\begin{aligned} s_i &= \alpha x_i + (1 - \alpha)s_{i-1} \\ &= \alpha x_i + (1 - \alpha)[\alpha x_{i-1} + (1 - \alpha)s_{i-2}] \\ &= \alpha x_i + (1 - \alpha)[\alpha x_{i-1} + (1 - \alpha)[\alpha x_{i-2} + (1 - \alpha)s_{i-3}]] \\ &= \dots \\ &= \alpha [x_i + (1 - \alpha)x_{i-1} + (1 - \alpha)^2 x_{i-2}] + (1 - \alpha)^3 s_{i-3} \\ &= \dots \\ &= \alpha \sum_{j=0}^i (1 - \alpha)^j x_{i-j} \end{aligned}$$

Now the name becomes clear: *all* previous observations

contribute to the smoothed value, but the contribution

is suppressed by increasing powers of the parameter α . The fact that observations further in the past are suppressed multiplicatively is characteristic for exponential behavior. In a way, exponential smoothing is like a floating average with *infinite memory, but with exponential fall-off*. (Also note the fact that the sum of the weights: $\sum_j \alpha(1-\alpha)^j$ sums to 1 as required, by virtue of the geometric series: $\sum_i q^i = 1/(1-q)$ for $q < 1$.)

Simple exponential smoothing as described above works well for time series without an overall trend. However, in the presence of an overall trend, the smoothed values tend to lag behind the raw data, unless α is chosen to be close to 1 — however, in this case the resulting curve is not sufficiently smoothed.

This is where *double* exponential smoothing comes in. In double exponential smoothing, we propagate two values from time step to time step: the actual value, and its “trend”, where the “trend” is really the *change* in the data from time step to time step.

The two equations that define double exponential smoothing are:

$$\begin{aligned} s_i &= \alpha x_i + (1-\alpha)(s_{i-1} + u_{i-1}) \\ u_i &= \gamma(s_i - s_{i-1}) + (1-\gamma)u_{i-1} \end{aligned}$$

Here the smoothed value s_i is the raw data, smoothed as before, but now with an additional contribution from the trend u_i . The trend itself is the single exponentially smoothed change in the main variable (i.e. $s_i - s_{i-1}$). Note that we now have a second parameter, conventionally labeled γ , which controls the smoothing of the trend.

The astute reader will have noticed that we have been silent on the proper way to start the recursion, i.e. the behavior for $i = 0$. In the literature, one can find discussions of this point — possible choices for single exponential smoothing include setting $s_0 = x_0$ or possibly to set s_0 to an average over $x_0 \dots x_3$. For double-exponential smoothing, a value for u_0 has to be chosen as well. Some experimentation is required, but overall I take the pragmatic point of view that if the choice of startup matters, then the entire time series is probably too short anyway!

Another question concerns the best choice of values for α (and γ). Again, we need to define what we mean by “best” in each specific situation. If we require smoothing mostly for visualization of data, some experimentation helps to find the trade-off between roughness and responsiveness of the smoothed graph.

IMPLEMENTATION NOTES

The beauty of exponential smoothing lies in its extreme simplicity. For single exponential smoothing, the entire operation can be done inline, for instance through a simple awk-script, which can be entered on the command line:

```
> awk 'BEGIN {a=0.05} NR==1 {s=$1} { s=a*$1 + (1-a)*s; print $1, s }' data
```

This program reads the noisy signal from the file called `data` and prints out the original signal, together with the smoothed value. Here, `a` controls the smoothing and we set its value in the `BEGIN` block before reading any of the input lines. The following block, which is only executed for the first data line read (i.e. when the *number of records* `NR` equals one), initializes the smoothed variable — this minimizes transient behavior at the beginning of the time series. The final block is executed for every input line read and performs the smoothing operation and output.

The interplay between the α and γ in double-exponential smoothing is hard to visualize. The example program is a rudimentary tool to try out the effect of different values for the smoothing parameters interactively. It is written in Python and uses the PyQt bindings for the Qt GUI toolkit.

The program should be very straightforward to understand — most of the code lines serve only to set up and arrange the GUI elements: a canvas to draw on, two sliders to adjust α and γ , and a bunch of canvas line elements, one for each consecutive pair of data points.

Every Qt application requires exactly one `QApplication` instance. This class to provide the necessary event loop to listen for user events. In our example, we define `App` as a subclass of `QApplication` and override its constructor to set up the main window with its elements. We also connect the sliders with the appropriate messages, which will force a redraw of the graphics whenever a value of one of the smoothing

parameters has been changed.

The actual smoothing operation is performed in the `redraw` method, which also repaints the canvas. The `redraw` operation is called with the new value of α or γ whenever one of the sliders has been moved.

Finally, we instantiate the `App` class and pass control to the event loop by calling `exec_loop`. The application now waits for user interface events, which Qt will dispatch to the appropriate updating method.

FORECASTING, SEASONALITY, AND ALL THAT

So far, we have merely attempted to replace a noisy signal with a curve that is less “bumpy”. Can we use the

same method to forecast future values of the underlying signal?

In principle, the answer is yes. However, we need to understand that whenever we attempt to predict future behavior of a system from its past, we implicitly make the assumption that there is an underlying model which governs the development of the system. If we *know* the behavior to be totally random, with each step being fully independent of all previous ones, we will know better than to attempt making forecasts.

The problem with single and double exponential smoothing is that either one makes a totally different assumption about the underlying model: while single exponential smoothing assumes that the system will stay steady at the last (smoothed) value (i.e. the predicted value k steps in the future is $s_{i+k} = s_i = \alpha x_i + (1 - \alpha)s_{i-1}$), double exponential smoothing assumes that the system will continue to grow linearly at the most recent (smoothed) rate (so that $s_{i+k} = s_i + ku_i$). Both will miss drastic departures from the previous behavior.

However, when we have good reason to believe that either of these two models applies to our system, exponential smoothing can be helpful in predicting future observations. There is even a third variant of exponential smoothing, not surprisingly known as *triple* or *Holt-Winters* exponential smoothing, which also takes into account a known seasonality.

Seasonality means that we know the system to undergo periodic changes, in addition to any linear trends that may exist — such as yearly patterns in consumer purchasing behavior, also known as Christmas shopping. (Of course many other examples of seasonal variations exist.) Note that the seasonal change can be either multiplicative or additive (different smoothing formulas apply in either case).

I will not repeat the formulas for triple exponential smoothing here, since they are somewhat messy. The main application area of triple exponential smoothing is forecasting in situations where we have good reason to believe that the underlying model is well approximated by a linear trend in addition to the seasonal change, we know the period of the seasonality ahead of time, *and* we have at least one full season of data points to bootstrap the recurrence relation! For mere visual smoothing of noisy data, triple exponential smoothing is usually not the most appropriate. The formulas, and application examples, can easily be found in the references at the end of this article.

FURTHER READING

The book **The Analysis of Time Series** by Chris Chatfield (Chapman and Hall, 6th ed., 2004) is a very handy, practical introduction to the field. It covers not only simple time series models such as exponential smoothing, but also more advanced topics state-space models and spectral methods. The treatment is practical throughout, but stresses understanding, instead of lapsing into a “cookbook recipe” approach.

A very good introduction to exponential smoothing specifically can be found in chapter 6.4 of the **Engineering Statistics Handbook** available online from NIST (the National Institute of Standards and Technology): <http://www.itl.nist.gov/div898/handbook/index.htm>.

Data sets to play with are available from **StatLib**: <http://lib.stat.cmu.edu>. The datasets *Andrews* and *hipel-mcleod* for instance contain a number of time series. The book by Chapman gives additional references.

© 2006 by Philipp K. Janert. All rights reserved.