Fair Exponential Smoothing with Small Alpha

(fessa)

Juha Reunanen

juha.reunanen@tomaattinen.com

October 10, 2015

Abstract

This short paper shows that standard exponential smoothing with a small α may not give the desired results $s(t)$ for small values of t, because the first sample $x(0)$ has such a huge impact on the outcome. Fortunately, the paper also proposes a remedy to the problem.

1 Introduction

Exponential smoothing[1] is a simple and effective technique for low-pass filtering a time series. In its most basic form, it is defined like this:

$$\begin{aligned} s(0) &= x(0) \\ s(t) &= \alpha \cdot x(t) + (1-\alpha) \cdot s(t-1) \end{aligned}$$

where $x(t)$ is the raw input signal, $s(t)$ is the filtered signal, and α is a parameter $(0 < \alpha < 1)$.

The technique has many nice properties, one of which is that even with a small α (i.e., heavy filtering), finding $s(t)$ is very fast — compared to calculating a moving average using a long window. This is very nice in case $x(t)$ is a sequence of large images, for instance.

2 Problem

One problem with using exponential smoothing, in particular with a small α, is that if $x(0)$ happens to be an outlier, it affects $s(t)$ even for pretty large values of t.

Denote with $w(k,t)$ the effect that $x(k)$ has over $s(t)$. Clearly:

$$\begin{aligned} w(0,0) &= 1 \\ w(0,1) &= 1-\alpha \\ w(1,1) &= \alpha \\ w(0,2) &= (1-\alpha)^2 \\ w(1,2) &= \alpha(1-\alpha) \\ w(2,2) &= \alpha^2 \\ &\dots \end{aligned}$$

Note that $w(k,t) = 0$ for any $k > t$, because we have a causal filter (i.e., events from the future cannot affect the current filtered value).

Now we can see the problem: the effect of the first sample, $w(0,t) = (1-\alpha)^t$, is quite significant even for large t, if α is small.

[1] https://en.wikipedia.org/wiki/Exponential_smoothing

Consider for instance $\alpha = 0.001$. You need to wait until $t = 693$ before the relative weight of the first sample drops below 50%! So the 691 samples $x(1), \ldots, x(691)$, all newer than $x(0)$, together weigh less than $x(0)$ does alone! And what if you want to use an even smaller α? It does not appear far-fetched to say that this can be quite unfair, depending of course on what exactly it is that you are doing.

3 Challenge

So we want to define a new, fairer scheme for $w(k,t)$. What properties should we expect?

Clearly we need to have $\sum_{k=0}^{t} w(k,t) = 1$ for any $t \geq 0$. Let's call this our first requirement.

Also, we would like to keep this asymptotic property of standard exponential smoothing:

$$w(k-1,t) = (1-\alpha) \cdot w(k,t)$$

But why not just require it for any k and t? Standard exponential smoothing caters for this only when $k > 2$ (any $t \geq k$) and $t \to \infty$ (any k). Yes, let's try and call this our second requirement.

Finally, we would like to be able to compute a new $s(t)$ using a small number of operations only.

4 Solution

Putting together the first and the second requirement, we have:

$$\begin{cases} \displaystyle\sum_{k=0}^{t} w(k,t) = 1 \\ w(k-1,t) = (1-\alpha) \cdot w(k,t) \end{cases} \tag{1}$$

Solving, it appears that we get:

$$w(k,t) = \frac{(1-\alpha)^{t-k}}{\sum_{\tau=0}^{t}(1-\alpha)^{\tau}}$$

This holds for $k \leq t$. For $k > t$, we have $w(k,t) = 0$, as pointed out earlier.

For the fessa method being introduced[2], we are looking for an $\alpha_{\mathsf{fessa}}(t)$ that we can use exactly as we use α in the standard method. Now we can define $\alpha_{\mathsf{fessa}}(t)$ as follows:

$$\alpha_{\mathsf{fessa}}(t) = w(t,t) = \frac{1}{\sum_{\tau=0}^{t}(1-\alpha)^{\tau}} = \frac{1}{d(t)}$$

Turns out that we can update α_{fessa} recursively:

$$\begin{aligned} a(0) = d(0) = \alpha_{\mathsf{fessa}}(0) &= 1 \\ a(t) &= a(t-1) \cdot (1-\alpha) \\ d(t) &= d(t-1) + a(t) \\ \alpha_{\mathsf{fessa}}(t) &= \frac{1}{d(t)} \end{aligned}$$

The filtering itself works as previously — we just use $\alpha_{\mathsf{fessa}}(t)$ instead of α:

$$s(t) = \alpha_{\mathsf{fessa}}(t) \cdot x(t) + (1 - \alpha_{\mathsf{fessa}}(t)) \cdot s(t-1)$$

[2] fessa stands for *fair exponential smoothing with small alpha*.

Table 1: Table of effective $w(k,t)$ in standard exponential smoothing for different k and $t \leq 10$, when $\alpha = 0.001$. The numbers are rounded to four significant digits. It is clear how $w(0,t)$ dominates — which means that the value of $x(0)$ has a significant impact on $s(t)$ when t is small.

		t										
		0	1	2	3	4	5	6	7	8	9	10
	0	1.000	.9990	.9980	.9970	.9960	.9950	.9940	.9930	.9920	.9910	.9900
	1		.0010	.0010	.0010	.0010	.0010	.0010	.0010	.0010	.0010	.0010
	2			.0010	.0010	.0010	.0010	.0010	.0010	.0010	.0010	.0010
	3				.0010	.0010	.0010	.0010	.0010	.0010	.0010	.0010
	4					.0010	.0010	.0010	.0010	.0010	.0010	.0010
k	5						.0010	.0010	.0010	.0010	.0010	.0010
	6							.0010	.0010	.0010	.0010	.0010
	7								.0010	.0010	.0010	.0010
	8									.0010	.0010	.0010
	9										.0010	.0010
	10											.0010

Table 2: Table of effective $w(k,t)$ in the fessa method for different k and $t \leq 10$, when $\alpha = 0.001$. Compared to Table 1, we can see that for each t, all samples received so far have an almost equal significance, however a more recent sample always weighs slightly more than do the earlier values of x.

		t										
		0	1	2	3	4	5	6	7	8	9	10
	0	1.000	.4997	.3330	.2496	.1996	.1663	.1424	.1246	.1107	.0996	.0905
	1		.5003	.3333	.2499	.1998	.1664	.1426	.1247	.1108	.0997	.0905
	2			.3337	.2501	.2000	.1666	.1427	.1248	.1109	.0997	.0906
	3				.2504	.2002	.1667	.1429	.1249	.1110	.0998	.0907
	4					.2004	.1669	.1430	.1251	.1111	.0999	.0908
k	5						.1671	.1431	.1252	.1112	.1000	.0909
	6							.1433	.1253	.1113	.1001	.0910
	7								.1254	.1114	.1003	.0911
	8									.1116	.1004	.0912
	9										.1005	.0913
	10											.0914

5 Examples

For $\alpha = 0.001$, Tables 1 and 2 show the effective $w(k,t)$ for small values of k and t for both the standard method and the proposed fessa approach, respectively. Based on Table 1, it is clear that in standard exponential smoothing the value of $x(0)$ has a huge impact on $s(t)$, when t is small. On the other hand, based on Table 2, the properties of the fessa method appear to be much nicer: at any time t, all samples processed so far have roughly the same weight, however the more recent the sample, the (slightly) more it weighs.

As another simple illustration, consider $x(t)$ drawn from a uniform distribution between 0 and 100. Have a look at Figures 1–3. It is clear that with such a small alpha (0.001), whatever $x(0)$ happens to be affects the results of the standard exponential smoothing method for a very long time — too long, one could probably say. On the other hand, the fessa method being proposed finds its way near the expected value quite soon.

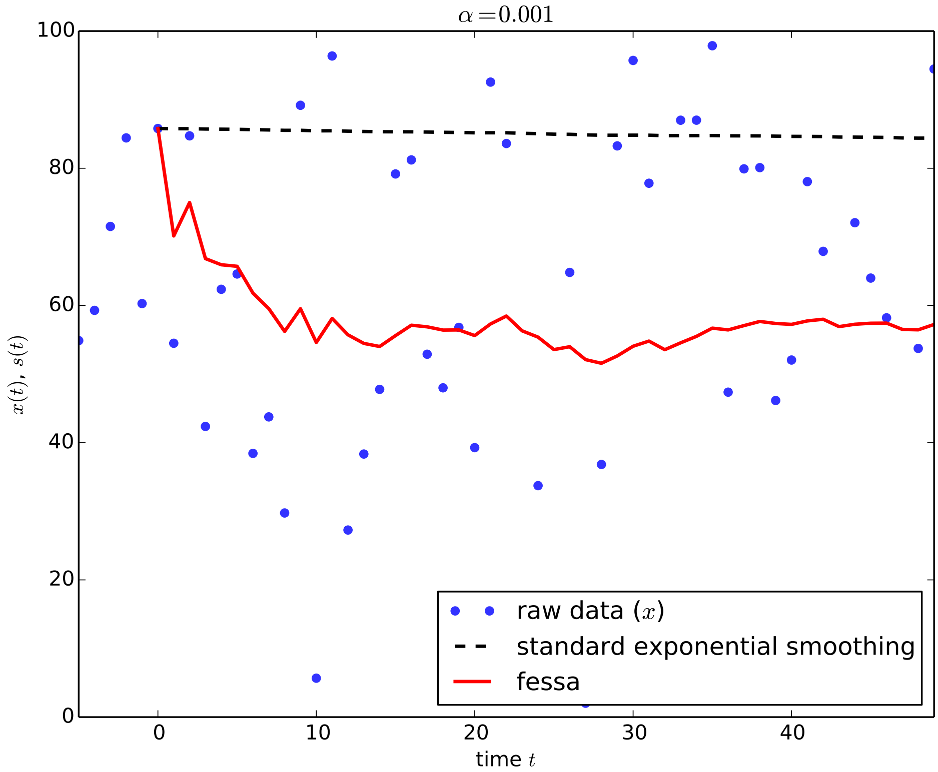

Figure 1: Comparison between standard exponential smoothing and the proposed fessa method for small t. Note that the standard method gets seriously stuck with the $x(0)$, even though it is not really even an outlier.

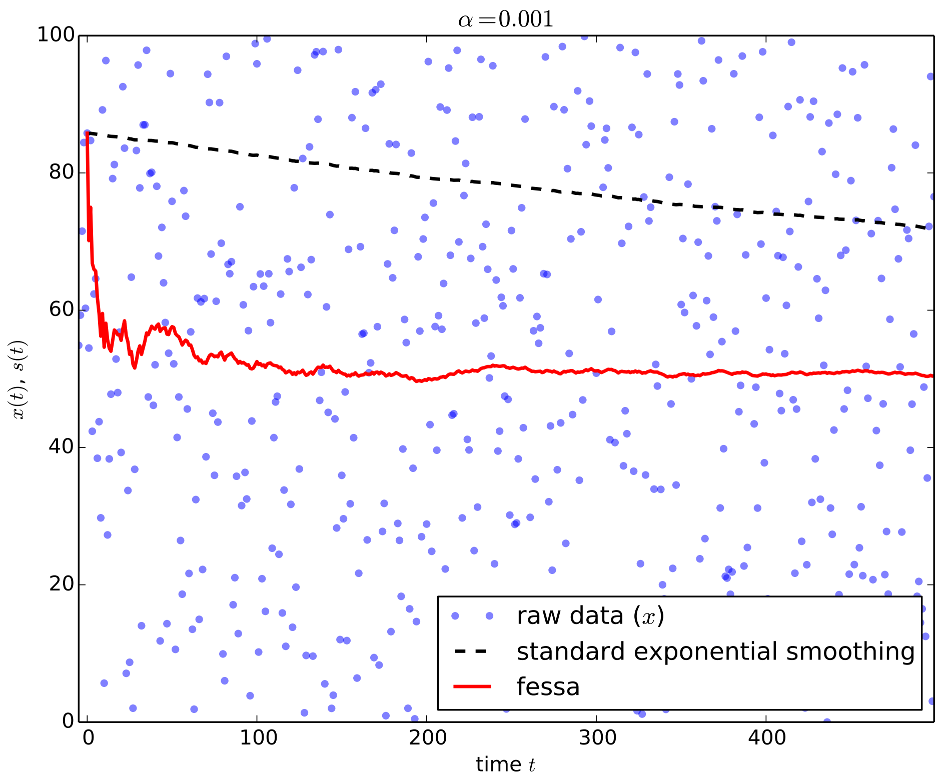

Figure 2: Comparison between standard exponential smoothing and the proposed fessa method for a medium range of t. Given that the expected value is 50, the standard method is still far away, even though the direction is clearly right.

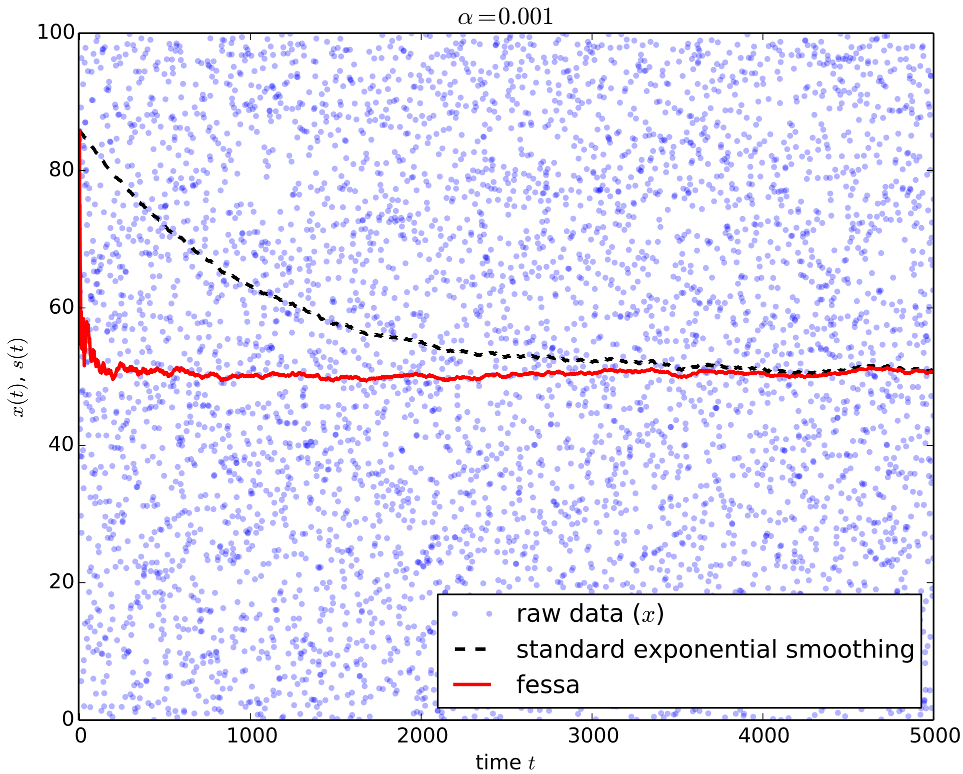

Figure 3: Comparison between standard exponential smoothing and the proposed fessa method for large t. Now even the standard method has reached the expected value. From around here on, the two methods are essentially identical.

6 Runtime performance

Because asymptotically ($t \to \infty$) the fessa method is equivalent to standard exponential smoothing, any implementation can switch to the standard method as soon as $(1 - \alpha)^t$ gets really small (close to machine epsilon or so). Then, for a large t the computational cost is on the same level with standard exponential smoothing.

With a small t, compared to the standard method there is additional multiplication (to update a), addition (to update d) and division[3] (to calculate α_{fessa}), so do not use fessa if the overhead is too much for you.[4] Otherwise, feel free to use the proposed method in place of standard exponential smoothing.

7 Missing proofs

It would be nice to prove that $\lim_{t\to\infty} \alpha_{\mathsf{fessa}}(t) = \alpha$.

It is quite obvious that $w(k,t) = \frac{(1-\alpha)^{t-k}}{\sum_{\tau=0}^{t}(1-\alpha)^{\tau}}$ satisfies (1), but should it be shown that the solution is unique?

[3] Actually this is just the reciprocal, which may be faster to calculate than generic division.

[4] But in this case please let me know what it is that you are working with — it sounds quite interesting already!

8 Disclaimer

The technique presented in this paper may well be a well-known one, but nevertheless I had fun figuring it out — not to mention coming up with the name. Indeed I think I need the method in practice, since I have a few times hesitated to set a low α because of being afraid of a very noisy $x(0)$. So if after this we can set lower αs and get smoother signals, it is quite enough for me, regardless of whether someone came up with this method already in the 1600s or so.

9 Code

See `https://github.com/reunanen/fessa` for a simple implementation of the fessa technique.