APPENDIX

Theoretically, there should be no difference between non-parametrically estimating a function $f(x)$ or $f\{T(x)\}$, where $T(x)$ is a known transformation of the domain of the function $x \to T(x)$. Since $f(\cdot)$ is not specified, $h = f \circ T$ is not specified and should be as easy to estimate non-parametrically as $f(\cdot)$. Unfortunately, in theory and in practice, this is not the case and the transformation of the x-space leads to substantially different estimators. To be concrete, consider the case when one is interested in estimating the smooth function $f(\cdot)$ from pairs of observed data (x_i, Y_i). A standard problem to solve in this case is to minimize the penalized criterion

$$\sum_{i=1}^{n}\{Y_i - f(x_i)\}^2 + \lambda \int \{f''(x)\}^2 dx \ ,$$

where λ is a smoothing parameter applied to the integral of the square of the second derivative of $f(\cdot)$. Of course other penalties could be used, but here we focus on the standard one. The following ideas can easily be generalized to other penalties. Consider the case when the x_i's are transformed into $z_i = T(x_i)$. In this case, one estimates a function $g(\cdot)$ such that the penalty becomes

$$\sum_{i=1}^{n}\{Y_i - g(z_i)\}^2 + \lambda \int \{g''(z)\}^2 dz \ ,$$

or, re-writing in terms of x

$$\sum_{i=1}^{n}\{Y_i - g(T(x_i))\}^2 + \lambda \int \{g''\{T(x)\}\}^2 dx \ .$$

If we wanted the exact same smoother then we would need to penalize the square of the second derivative of $g \circ T$, which is equal to

$$g''\{T(x)\}\{T'(x)\}^2 + g'\{T(x)\}T''(x) \neq g''\{T(x)\} \ .$$

The equality holds if the transformation $z_i = T(x_i)$ is linear, but does not typically hold for other types of transformations. In short, transforming the function domain, then smoothing on the new domain and transforming back to the original domain is not equivalent to smoothing on the original domain unless the transformation is linear. One could obtain the exact same estimate if the penalty term were changed to match the transformation, but almost never is this the case in practice. Below we show how the curve estimate can change dramatically via the domain transformation and back transformation after smoothing.

To illustrate this point, consider the class of functions,

$$m(x,j) = \sqrt{x(1-x)}\ sin\left\{\frac{2\pi(1+2^{(9-4j)/5})}{x+2^{(9-4j)/5}}\right\} \text{ for } x \in [0,1] \tag{1}$$

and the class of transformations of the domain $T(x) = x^{\alpha}$ for $\alpha > 0$. Figure 1 shows how the function $m(x,6)$ changes as the domain is transformed by $T(x) = x^{\alpha}$ for four different levels, $\alpha = (0.1, 0.5, 2, 10)$. When $\alpha > 1$, the points are shifted to the left and compacted to be close to 0, but when $\alpha < 1$, the points are shifted to the right and compacted close to 1. This class of functions has been used extensively to illustrate adaptive smoothing and is considered to be a difficult class of functions to fit.

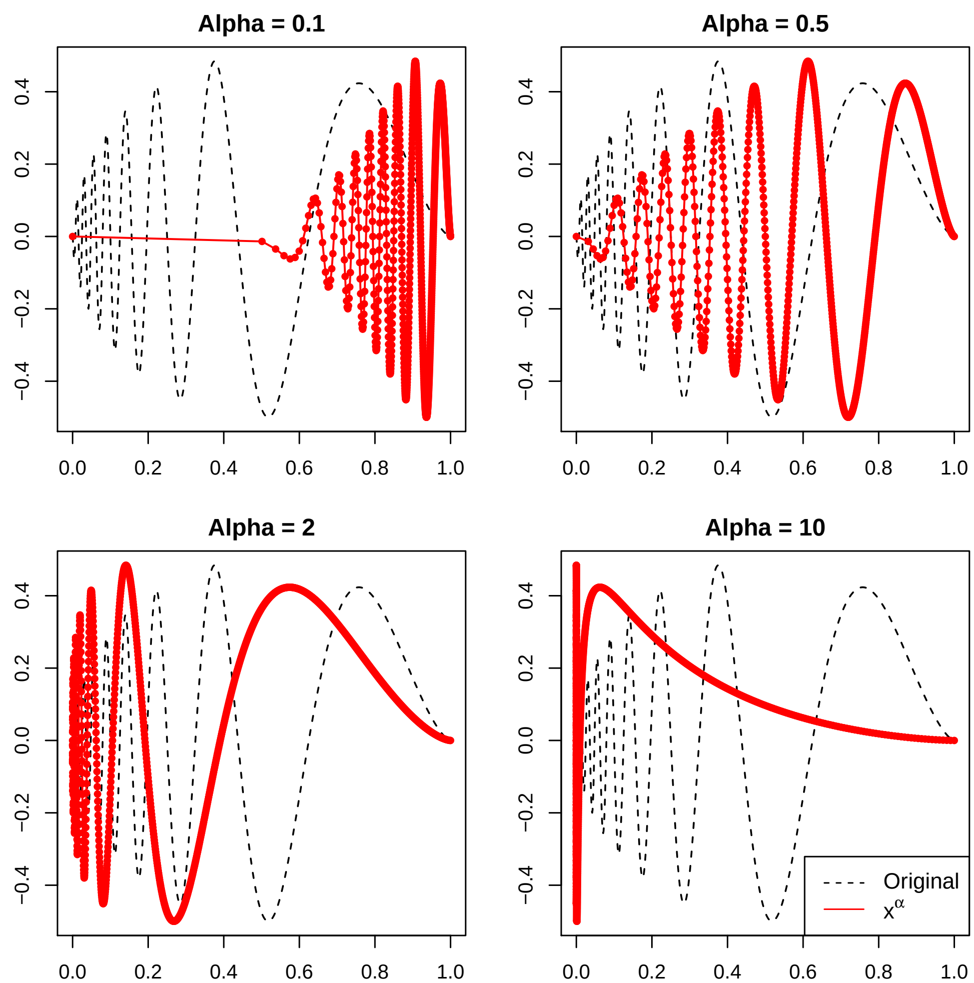

Figure 1: The black dashed line shows the original function $m(x,j)$ from Equation (8) where $j = 6$. The red line and corresponding points show the same function with the transformed domain $T(x) = x^{\alpha}$ for $\alpha = (0.1, 0.5, 2, 10)$.

Figure 2 shows the result of smoothing the function on the original domain (shown in red) compared to the smoothed results of the transformation $T(x) = x^{\alpha}$ for $\alpha = 0.01$ and 20 (shown in green and blue respectively). The original data is shown in gray. These results were obtained by fitting a cubic smoothing spline using the function smooth.spline in R (R Core Team, 2018). Each spline had 150 knots and the smoothing parameter was determined by generalized cross-validation. After fitting the data, everything is transformed back to the original scale. Note the substantial differences among these fits. Indeed, when the transformation is $T(x) = x^{20}$ (shown in blue) the entire first part of the function is missed, while the second part of the function has an extremely unsmooth pattern. When there is no transformation before smoothing (curve in red), the first part of the function fits relatively well, but the second part still displays a strong under-smoothed pattern. Conversely, after the transformation corresponding to $\alpha = 0.01$, the estimator performs exceptionally well, even though we are applying a standard smoother. There are three important conclusions here: 1) transforming the x-space, smoothing using standard techniques, and re-transforming back to the original scale of the data is not invariant to non-linear transformations; 2) the more one understands the underlying structure of the x-space, the better the smoothing results; 3) transformation of the x-space should be coupled with the corresponding transformation of the penalty term on the unspecified smooth function.

We would also like to highlight a completely unexpected finding. Indeed, these functions have been introduced in the literature as examples of hard functions to smooth and have been used extensively to highlight the power of adaptive smoothing. More precisely, methods have been proposed to either increase or decrease the number of knots for splines or allow for a different smoothing parameter as a function of the domain (Crainiceanu, Ruppert, Carroll, Joshi, & Goodner, 2007). Thus, we are showing that for this class of functions, these approaches are actually not necessary and a careful transformation of the x-domain can provide extraordinary improvements in the smoothing approach without requiring advanced adaptive smoothing techniques. In other words, we have achieved adaptive smoothing using standard smoothing plus a simple transformation of the x-space. This further highlights that transforming the x-space can have substantial, unexpected, and strong effects on the smoothing estimators.

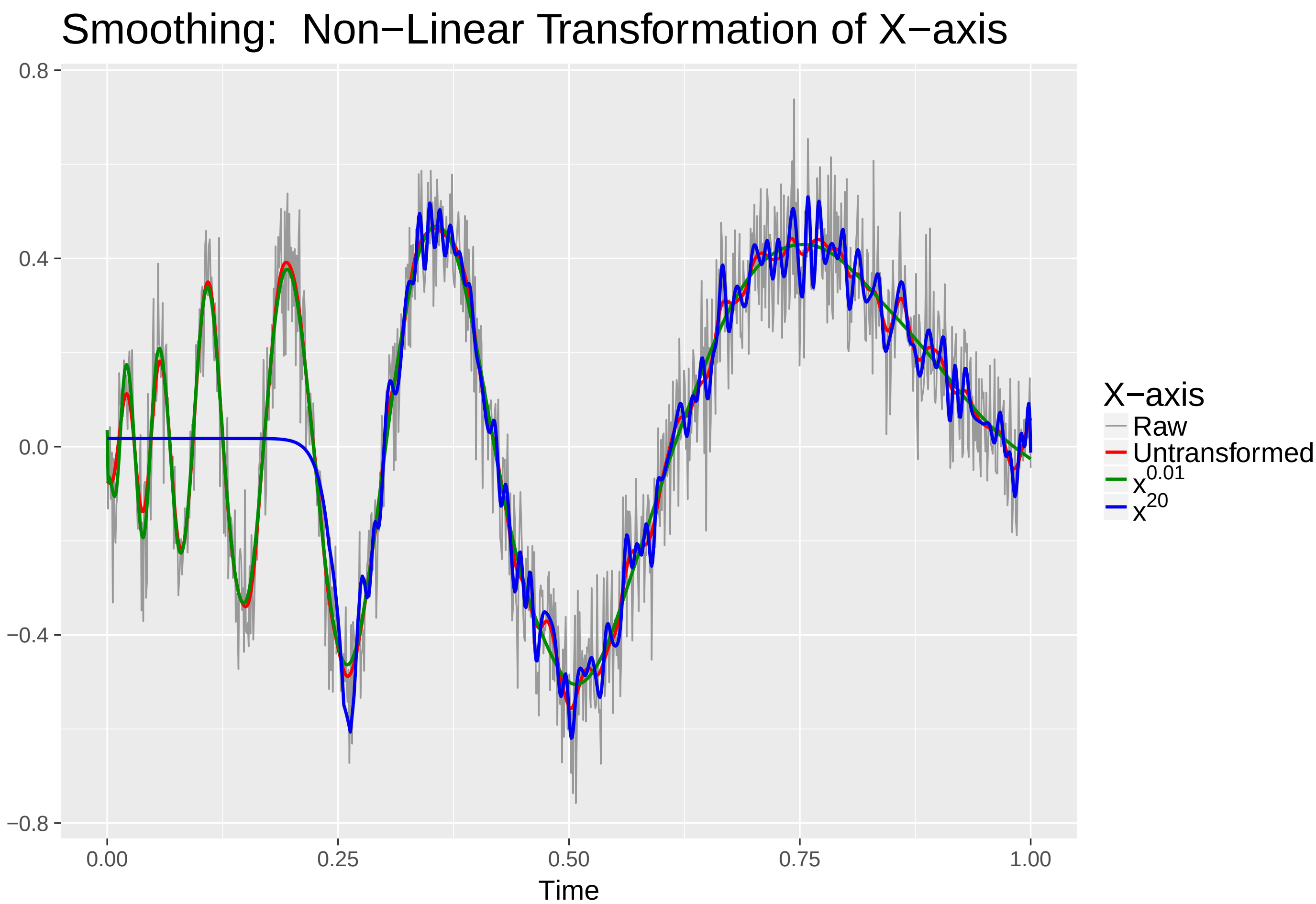

Figure 2: Smooth estimates of a function with and without transformation of the domain. Raw data is shown in dark gray and the smoothed data on the original domain is shown in red. The green and blue curves show the smooth estimates after transformation of the domain, x^{α} for $\alpha = 0.01$ and 20 respectively.

References

Crainiceanu, C., Ruppert, D., Carroll, R. J., Joshi, A., & Goodner, B. (2007). Spatially adaptive bayesian penalized splines with heteroscedastic errors. *Journal of Computational and Graphical Statistics*, *16*(2), 265-288.

R Core Team. (2018). *R: A language and environment for statistical computing.* Vienna, Austria. URL https://www.R-project.org/.